

pdfToolbox

callas pdfToolbox

callas

Table of Contents

1. Installation, activation, deactivation, updates 15

1.1 Installation of pdfToolbox Desktop or Server on Mac 16

1.2 Installation of pdfToolbox Desktop or Server on Windows 25

1.3 Activation and Deactivation of pdfToolbox 30

1.4 Multi-user system (Activating additional users on the same system) 40

1.5 Migrating the pdfToolbox to another computer (Deactivation) 42

1.6 Uninstalling the pdfToolbox Acrobat plug-in 47

1.7 Change language of user interface (desktop only) 48

1.8 Using pdfToolbox and Preflight in Acrobat Pro parallel 50

1.9 Notifications about available updates 59

1.10 Run as a service (Windows) 64

1.11 Create a Daemon using Mac 71

1.12 Run as an autostart systemd service (Linux) 72

1.13 System paths for preferences 76

1.14 How to upload large files to support 78

2. callas pdfToolbox Basics 80

2.1 The Switchboard: easy-to-use Actions 81

2.2 Profiles, Checks, Fixups, Process Plans and Libraries 86

2.3 Checks and Fixups 87

2.4 Duplicate and edit Checks and Fixups 94

2.5 Creating Profiles 101

2.6 Duplicate and edit Profiles 114

2.7 Protect Profiles, Checks, Fixups, Process Plans and Libraries against changes 122

2.8 Running a Profile and examining results 125

2.9 Export Profiles to a previous pdfToolbox version 130

2.10 Preferences 134

2.11 Syntax Checks 138

2.12 Examining page content 140

2.13 Requirements for conversions to PDF 143

2.14 Add your own Output Intents....155
2.15 Checking the ISO standards using callas Desktop products....157
2.16 Compatibility between pdfToolbox and Acrobat....160
2.17 Deleting multiple Fixups or Checks at once....162
2.18 Automatic optimization of PDFs....164
2.19 Keyboard shortcuts and additional Hotkeys....166
2.20 Page selector and Split scheme....170
3. Process Plans in detail....176
3.1 When Profiles are not enough: Process Plans....177
3.2 Process Plans: tips and tricks....186
3.3 Using Quick Check as a step in a Process Plan....191
3.4 Process Plan step "Create PDF copy"....208
3.5 Process Plan step "File pick up"....213
3.6 Process Plan step "Rename"....218
3.7 Process Plan step "Add files"....222
3.8 Using Process Plans in Process Plans....226
3.9 Using Process Plans in Process Plans to loop over pages....230
3.10 Switch on/off a Process Plan step via a Variable....234
3.11 Modifying the result (return code) of a Process Plan with the Check property "Create a hit (or not)"....237
3.12 Modifying structures in a Process Plan....238
4. Protected Profiles and Process Plans....239
4.1 What are protected Profiles and Process Plans....240
4.2 Protecting one or more Profiles or Process Plans....242
4.3 Modifying one or more protected Profile or Process Plan....246
4.4 Unprotecting one or more Profiles or Process Plans....249
5. Actions and their use in callas' products....251
5.1 Introduction....252
5.2 Actions for automation....256
5.3 Modify images....272
5.4 Create file (from process plan)....275

6. Libraries 277
6.1 Libraries - Overview 278
6.2 Create new, Manage, Import or Export Libraries 282
6.3 Search in or across Libraries 292
7. Color conversion 296
7.1 Color conversion overview 297
7.2 Setting up a "Convert colors" Fixup 299
7.3 Convert colors: Advanced settings (previously "Policies") 307
7.4 Policies (deprecated, available till v10.2) 322
7.5 Processing black objects with Advanced settings (v11.0) 345
7.6 Processing black objects (deprecated, available till v10.2) 351
7.7 Convert colors to PSO Coated v3 (ECI) 357
7.8 Adjust tone values and apply gradation curve to selected objects 363
7.9 How to convert CMYK in DeviceN to DeviceCMYK 380
7.10 DeviceLink conversion 387
7.11 Use included DeviceLink profile to convert ISO Coated v2 ↔ PSO Coated v3 (ECI) 397
7.12 Replace existing ICC profile 404
7.13 Convert spot color names to UTF-8 419
7.14 Update all spot colors in a PDF using a spot color Swatch library 429
7.15 Internal spot color library 433
7.16 Convert RGB to CMYK using custom tolerance for gray (v11.0) 440
7.17 Extract CMYK from DeviceN 444
8. PDF/X-5 and n-channel color 445
8.1 Color convert with n-channel ICC profiles 446
8.2 Convert to PDF/X-5n 454
8.3 Validate against PDF/X-5n standard 469
8.4 Softproof PDF/X-5n files 473
9. Spectral color and CxF 478
9.1 Introduction: CxF and spectral data 479
9.2 Embed CxF data (import) 480

9.3 Extract and remove CxF information 484
9.4 Analyze CxF information 489
10. Spotify - Derive spot colors from content (v11.0) 493
10.1 Why Spotify? 494
10.2 How does Spotify work 495
10.3 Spotify in callas pdfToolbox 498
10.4 Spotify parameters 509
11. Fonts and Text 518
11.1 Embedding fonts 519
11.2 Embedding fonts: Font substitution file 521
11.3 Subset Fonts 527
11.4 Unembed all fonts 528
11.5 Convert fonts to outlines 531
11.6 Convert Type 3 fonts to outlines (10.2) 534
11.7 Search text 541
12. Images 545
12.1 Resample to JPEG2000 546
12.2 Upsample/downsample images 548
13. Prepare PDF documents for production 552
13.1 Generate bleed from page content 553
13.2 Generate bleed for irregular shapes 563
13.3 Check and fix bleed 568
13.4 Create a dieline and bleed for irregular shapes (v11.0) 570
13.5 Create a dieline and bleed for irregular shapes with gaps in outer border (v11.0) 573
13.6 Change page size with the help of the “Set page geometry boxes” Fixup 577
13.7 Outline page geometry boxes in a specified tint value of a spot color 581
13.8 Create pre-separated pages 595
13.9 How to 'Split or reorder' PDF 597
13.10 Create white underlays for printing on transparent foil using "Create spot color plate based on ink amount" 600
13.11 Derive spot color from softmasks 616

13.12 Ink coverage Check properties 619

14. Large format 640

14.1 Adding grommets using a Fixup 641

14.2 Add ink layer 644

14.3 Adding grommets 649

14.4 Tiling 655

14.5 Add borders 659

14.6 Add bleed 663

14.7 Create pole pocket banner 666

14.8 Smart resolution to limit memory consumption 673

15. Variable Data Print (VDP) 675

15.1 Create VDP files from PDF templates 676

15.2 Fast VDP mode 684

15.3 DPart metadata injection 687

15.4 Distribute form XObjects on layers 688

16. Debugging of Profiles and Process plans 695

16.1 Why a test mode? 696

16.2 How to use test mode 697

16.3 Purpose of logging feature 703

16.4 Activating logging 705

16.5 How to create a detailed log when executing Process Plans (or Profiles, Checks or Fix-ups) 708

16.6 The JSON log files 717

17. Variables and JavaScript, Ask-at-runtime dialog 730

17.1 Simple variables 731

17.2 Using Variables for resources 743

17.3 Variables using JavaScript: Overview 745

17.4 JavaScript in pdfToolbox: Custom data objects and methods 759

17.5 Interpretation of the app.requires values 765

17.6 Using variables in a property - The TrimBox example 766

17.7 Extracting information from an XML Report file via XPath 774

17.8 Using an external JSON jobticket file 776
17.9 Defining variables using app.requires with closed choice of allowed values....778
17.10 Process pages differently in a Process Plan using a Check and JavaScript....780
17.11 Using "trigger" values to adjust processing in a Process Plan 784
17.12 How to branch processing in a Process Plan using JavaScript 788
17.13 Debugging JavaScript Variables 790
17.14 Use RegEx in variables....796
17.15 Using object coordinates from a hit in a Process Plan 803
17.16 Map (spot and process) colors using script variables 805
17.17 Ask-at-runtime Dialog: Introduction....812
17.18 Adjust ask-at-runtime dialog....816
17.19 Working with ask-at-runtime templates....822
17.20 Arbitrary JavaScript controlled Fixups....826
17.21 Referencing resources in arbitrary JavaScript controlled Fixups 831
17.22 Using JavaScript for pop up values in "Arbitrary JavaScript controlled Fixups" 835
17.23 Using JavaScript in an (apply to) filter 844
17.24 Arbitrary JavaScript controlled Checks....848
17.25 Execute external application via JavaScript function 856
17.26 Autocomplete while writing JavaScript 860
17.27 Editing JavaScript in Visual Studio Code 862
18. Advanced configuration of Profiles 867
18.1 Boolean logic and conditions in preflight checks 868
18.2 Combining different checks to solve complex problems 870
18.3 Negating Check results....872
19. Reports for Profiles....876
19.1 PDF reports (using masks or layers) 877
19.2 HTML based custom reports 884
19.3 HTML based custom reports – generate optional information 891
19.4 HTML based custom reports – control custom parameters via JavaScript 906
19.5 Dump variables report (using JSON) 912
19.6 Command line options for multi-language support....914

19.7 Modify strings for internal syntax checks using Custom Dicts 917
19.8 XML report: Convert pt to mm....919
19.9 JSON report – based on JS object 921
19.10 JSON report – similar to XML report 926
20. Place content....931
20.1 Place dynamic page numbers....932
20.2 Place dynamic text 938
20.3 Place individual text per page....945
20.4 Place barcodes and matrix codes 948
20.5 Place any content: Basics (“Place Content” via HTML templates....954
20.6 Place any content: Preparation 963
20.7 Place any content: Positioning content....975
20.8 Place any content: Simple example placing content in the upper left 980
20.9 Using Variables in a Place Content HTML template to place text 986
20.10 Place any content: Use information about the PDF document....989
20.11 Place barcode: Using an HTML-template for an extended configuration....999
20.12 Advanced 2D code use cases....1005
20.13 Place content transparently or opaquely....1010
20.14 Overview of pdfChip versions in pdfToolbox 1013
21. Shapes 1014
21.1 Shapes: An overview....1015
21.2 Defining shapes 1017
21.3 Applying shapes....1036
21.4 Extended "Shapes" features 1049
21.5 Efficiently creating varnish or white background (requires at least v9.1)....1055
21.6 Use shapes to visualize small distances between objects or inside of outlined objects 1062
22. Optical character recognition (OCR) 1067
22.1 Create invisible text via OCR 1068
22.2 OCR support for additional languages 1072
22.3 Partial OCR (filtering page content)....1077

23. Barcode recognition 1081
23.1 Find barcodes 1082
23.2 Supported Barcode symbologies 1087
23.3 Read Barcode or Matrix code and determine properties 1090
24. Context aware object detection: Sifter 1095
24.1 Beyond classic preflighting: Context aware object detection (Sifter) 1096
24.2 Finding the right "Context aware object detection" property 1103
24.3 Proximity: Object reaches into edge area of a shape 1110
24.4 Proximity: Object crosses shape 1120
24.5 Proximity: Objects close to each other 1123
24.6 Above versus below: Object on top of other object(s) 1126
24.7 Above versus below: Object not on top of any other object 1128
24.8 Above versus below: Object below other object 1130
24.9 Above versus below: Object not below any other object 1134
24.10 Above versus below: Object covers other object 1137
24.11 Above versus below: Object covered by other object 1140
24.12 Inside versus outside: Object inside other object 1143
24.13 Inside versus outside: Object outside other object 1146
24.14 Inside versus outside: Object inside shape 1149
24.15 Inside versus outside: Object outside shape 1153
24.16 Inside versus outside: Object crosses other object 1157
24.17 Visibility: Object is invisible 1161
24.18 Visibility: Object is visible 1164
24.19 Visibility: Object is partially obliterated 1165
24.20 Visibility: Object is completely obliterated 1167
24.21 Visibility: Object is partially clipped 1169
24.22 Visibility: Object is completely clipped 1171
24.23 Advanced "Context aware object detection" property 1173
24.24 "Shapes" Property for use in "Context aware object detection" checks 1177
24.25 How to debug Sifter Checks and see their results 1183
25. Processing Steps 1188

25.1 Design and more 1189
25.2 Using metadata for standardisation 1192
25.3 Working with processing steps metadata for a layer 1194
25.4 Predefined "Processing Steps" library 1198
25.5 Checking processing steps information 1203
25.6 Fixing processing steps data 1207
25.7 Overview of predefined groups and types for processing steps 1211
25.8 Configurable Checks and Fixups for Processing Steps 1215
25.9 Split PDFs based on Processing Steps 1216
25.10 Index layer names if initial visibility or Processing Steps metadata is different 1218
26. PDF 2.0 in prepress 1221
26.1 Check for print and prepress related PDF 2.0 features 1222
26.2 Display DPart metadata 1225
26.3 Use DPart metadata in a Process Plan via QuickCheck 1228
26.4 DPart metadata injection 1230
27. Interactively analyse and explore PDF documents 1231
27.1 Visually inspect PDF files 1232
27.2 Display ink coverage information for all separations 1236
27.3 View ink coverage per separation 1239
27.4 View safety zone in PDF 1243
27.5 Heat map for «Out of gamut» visualization 1244
27.6 Examining page content: The Object Inspector 1247
27.7 Examining page content: Filter in the Object Inspector 1253
27.8 Compare documents 1256
27.9 Log comparison based information 1260
27.10 Explore Metadata 1262
27.11 Explore Layers 1264
27.12 Explore PDF 1267
27.13 Explore Fonts 1281
27.14 Explore tagging 1283
27.15 XMP Metadata reports 1285

27.16 Analyze the space usage of a PDF 1298

28. Quick Check 1301

28.1 Quick Check – Introduction 1302

28.2 Using Quick Check as a step in a Process Plan 1305

28.3 Quick Check configuration syntax 1322

28.4 All "aggregated" Quick Check objects and output 1327

28.5 "direct" data structures and output 1424

28.6 "status" data structure and output 1432

28.7 Using Quick Check directly on the command line 1434

28.8 Error codes and Return codes for Quick Check Results 1438

29. Quick Fix 1440

29.1 Quick Fix overview 1441

29.2 Overview Quick Fixes with corresponding Fixups 1448

29.3 Quick Fix configuration essentials: String comparison operators and page selection expressions 1450

29.4 Quick Fix features 1455

29.5 Using Quick Fix on the command line 1502

29.6 Reorder pages using Quick Fix 1504

29.7 JavaScript based configuration Quick Fix 1509

29.8 Apply transformation matrix 1515

30. Imposition 1518

30.1 Overview of imposition in pdfToolbox 1519

30.2 Simple imposition - how to use 1522

30.3 Simple imposition - available methods 1527

30.4 Simple imposition - defining sheet sizes 1530

30.5 Getting started with imposition configurations 1534

30.6 Imposition configuration terminology - general 1538

30.7 Imposition configuration terminology - shingling 1543

30.8 Imposition configuration editing and debugging 1547

30.9 Getting imposed sheet information after imposition 1552

30.10 JavaScript imposition - getting started 1554

30.11 JavaScript imposition - building an imposition configuration 1562
30.12 JavaScript imposition - step by step examples....1565
30.13 JavaScript imposition - inserting imposition snippets....1568
30.14 JavaScript imposition - convenience functions....1571
30.15 JavaScript imposition - object definitions 1574
30.16 JavaScript imposition - using Place Content during imposition 1588
30.17 Old-style imposition - building blocks....1595
30.18 Old-style imposition - controlling the process....1605
30.19 Old-style imposition - runlists....1607
30.20 Old-style imposition - token engine 1619
30.21 Old-style imposition - an example imposition configuration....1635
30.22 Old-style imposition - using variables during imposition 1648
30.23 Old-style imposition - creating variable-size impositions....1649
31. Downloadable and adjustable solutions 1654
31.1 Process Plan: Color picker to derive a specific color 1655
31.2 Process Plan: Determine text in custom area....1658
31.3 Process Plan: Extract text using OCR....1661
31.4 Process Plan: Scale page excluding Processing Steps....1663
31.5 Process Plan: Create bookmarks from headings 1666
31.6 Process Plan: Use QR Codes to place icons and link annotations 1669
31.7 Process Plan: Single out page objects....1672
32. callas pdfToolbox CLI (command line interface) 1678
32.1 Introduction to pdfToolbox CLI....1679
32.2 Installation and activation of pdfToolbox Server/CLI....1684
32.3 Hints and troubleshooting & Displaying program information 1690
32.4 Processing....1695
32.5 Using Profiles 1700
32.6 General command line options....1706
32.7 Converting office documents to PDF or PDF/A....1712
32.8 Additional command line options and response files 1715
32.9 Creating a report using Profiles 1719

32.10 Enumerate Profiles 1729
32.11 Results (Return codes, Error codes and Reason codes) 1732
32.12 Commands related to Arrange 1736
32.13 Commands related to Large format printing 1754
32.14 Commands related to Present 1756
32.15 Commands related to Document 1759
32.16 Command to add bookmark structure 1775
32.17 Commands related to Colors 1777
32.18 Commands related to Layers 1779
32.19 Commands related to Reports 1782
32.20 DeviceLink Conversion 1790
32.21 Run as a Server 1791
32.22 Distributed Processing 1794
32.23 Running pdfToolbox via Webservices (SOAP) 1804
32.24 Activating logging 1808
32.25 Predefined Profiles 1813
32.26 Modifying structure of bookmarks and DPart 1869
32.27 Handling Licensing through the License Server 1878
32.28 Creating ZUGFeRD / Factur-X invoices 1879
33. Using pdfToolbox in cloud environments 1882
33.1 Using the callas pdfToolbox CLI Docker Image from Docker Hub 1883
33.2 Docker Snippets for callas pdfToolbox SDK 1886
33.3 Using the License Server 1887
34. callas pdfToolbox SDK 1889
34.1 Activation and Deactivation of pdfToolbox SDK 1890
34.2 callas pdfEngine SDK: First steps 1893
34.3 Help? Displaying program information for pdfToolbox SDK 1896
34.4 File components and their use in the SDK 1897
34.5 .Net Core migration guide 1901
34.6 Predefined Profiles 1906
35. Server 1962

35.1 Introduction to pdfToolbox Server 1963
35.2 Connect with Remote Server 1968
35.3 Job settings 1970
35.4 Using Job Tickets 1974
35.5 Sidecar files and their use when processing files with a Server-Job 1978
35.6 How to upgrade pdfToolbox Server successfully (without losing server jobs)? 1982
35.7 pdfToolbox Server integration in automation systems 1983
36. Enfocus Switch 1988
36.1 Using pdfToolbox in Enfocus Switch 1989
36.2 Setting the path for the pdfToolbox configurators 1991
36.3 Setting the path for the pdfToolbox Switch apps 1993
36.4 Deprecated Switch apps 1995

1. Installation, activation, deactivation, updates

1.1 Installation of pdfToolbox Desktop or Server on Mac

You can download the latest version of pdfToolbox Desktop or pdfToolbox Server from our website by requesting a trial:

1. pdfToolbox Desktop: pdftoolboxdesktop
2. pdfToolbox Server: pdftoolboxserver

You can also register on our website for easy access to all pdfToolbox download links. Once registered and logged in, you can access the list of download links:

1. pdfToolbox Desktop: pdftoolboxdesktop
2. pdfToolbox Server: pdftoolboxserver

System requirements & hardware recommendations

Please refer to the callas website for system requirements and hardware recommendations for pdfToolbox Desktop and pdfToolbox Server:

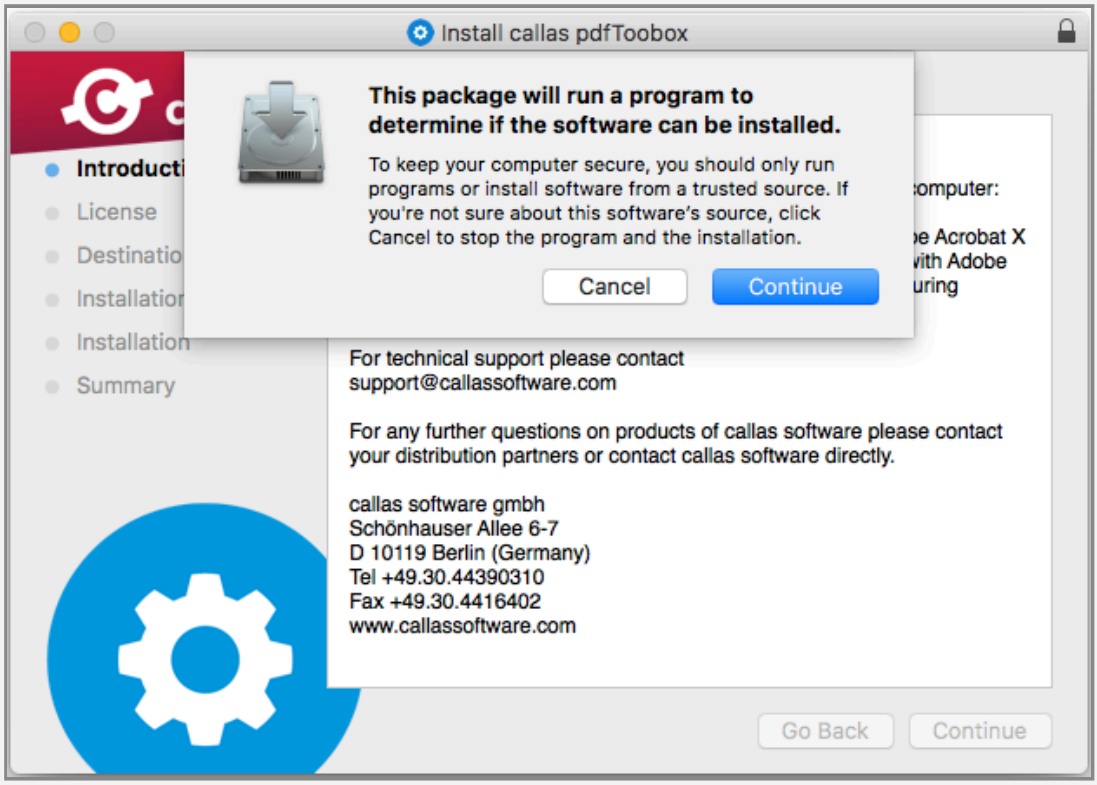
- System requirements

Installation on Mac



To install the application, double click the pdfToolbox icon.

System Check



The package will then check the integrity of the installer.

pdfToolbox 9 or lower **DO NOT WORK** on Mac OS 10.15 (Catalina).

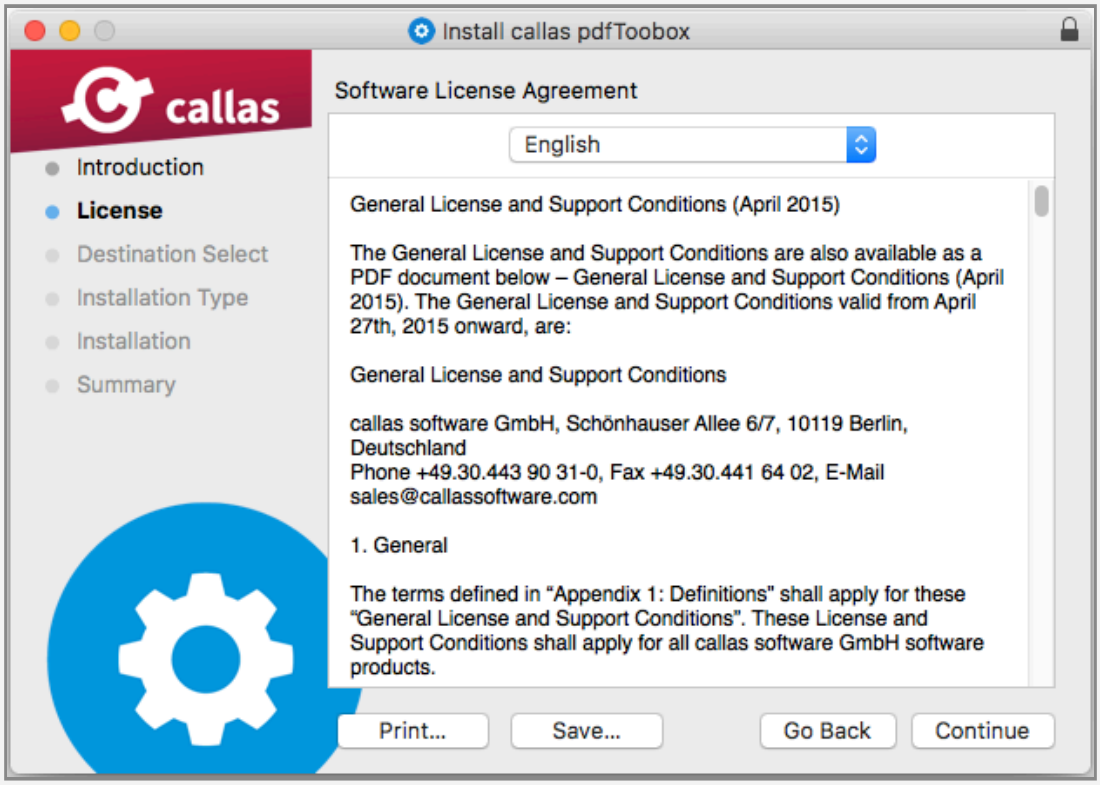
A workaround for this is to install the pdfToolbox 9 PlugIn on Acrobat (Mac OS 10.15 (Catalina)). Please write to support@callassoftware.com for the Plug-In.

Introduction Screen



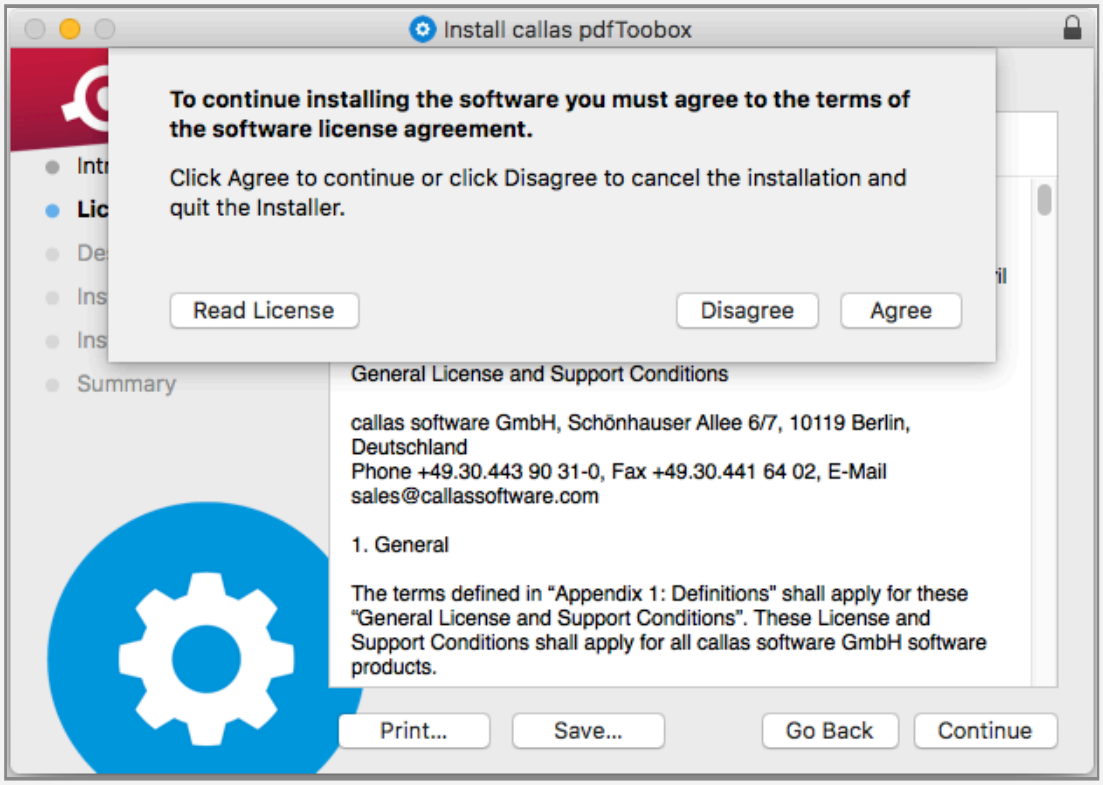
Please check that your system supports the requirements

License Dialog



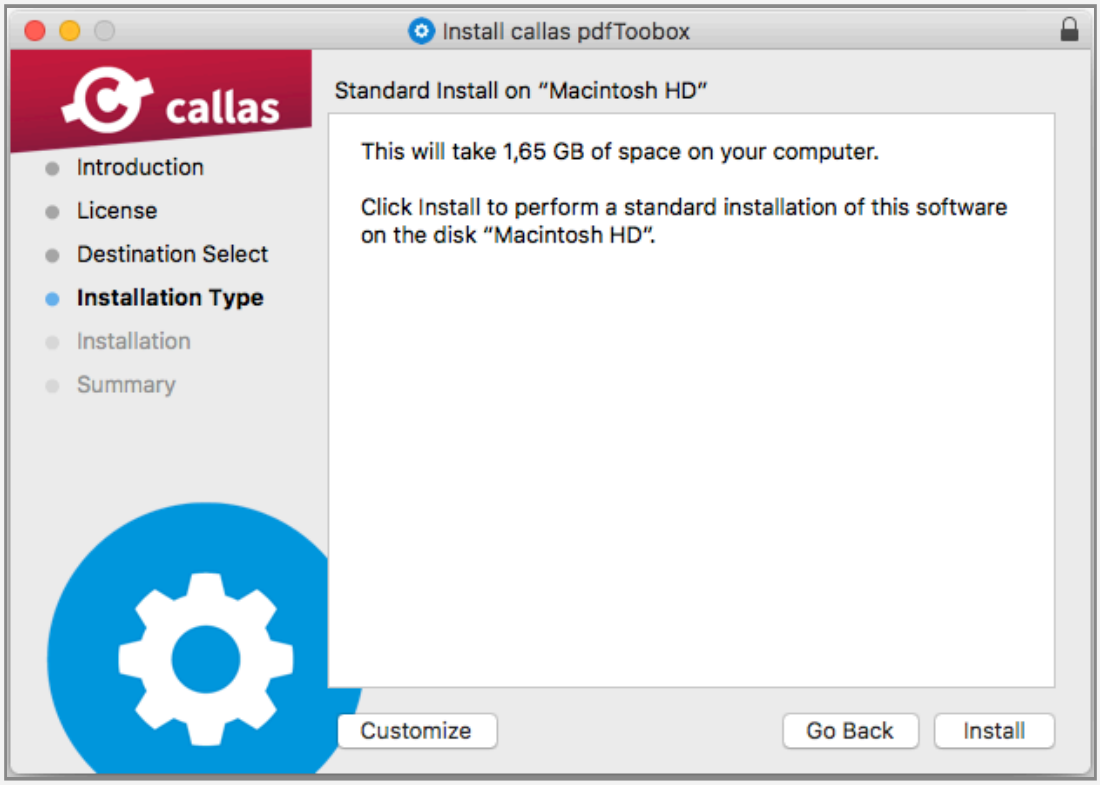
Please read the License agreement

Agree to the terms of agreement

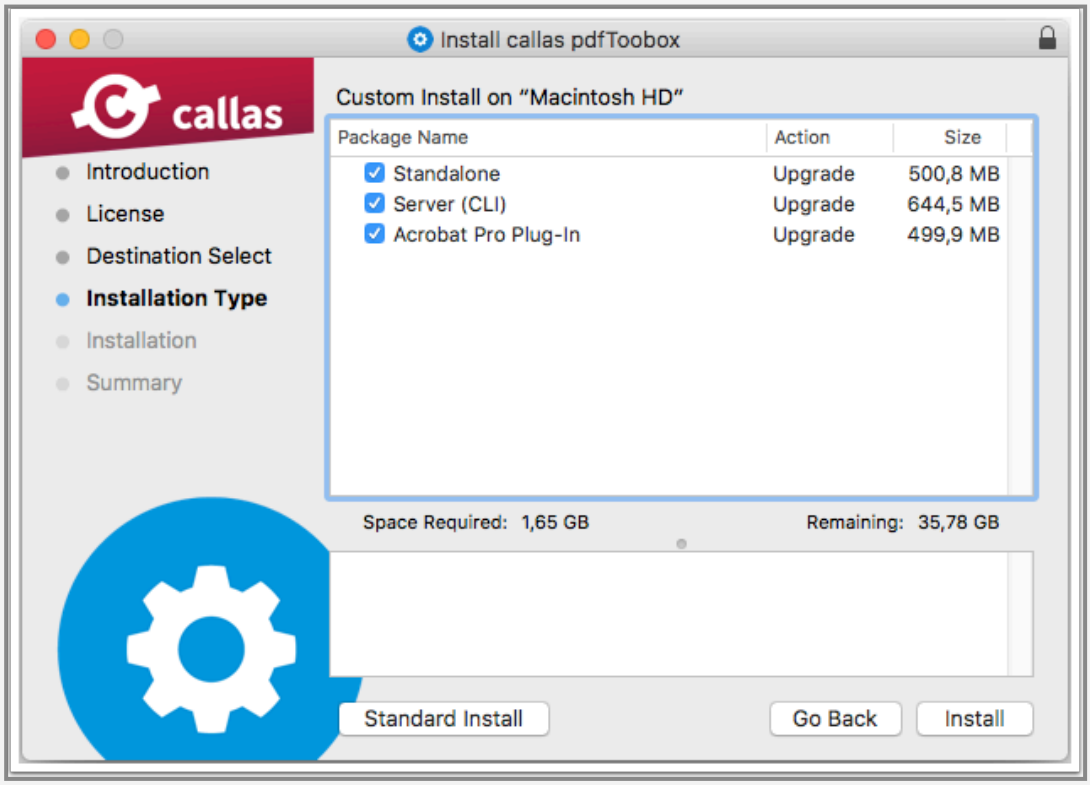


Click on "Continue" if you wish to accept the terms and and install the software.

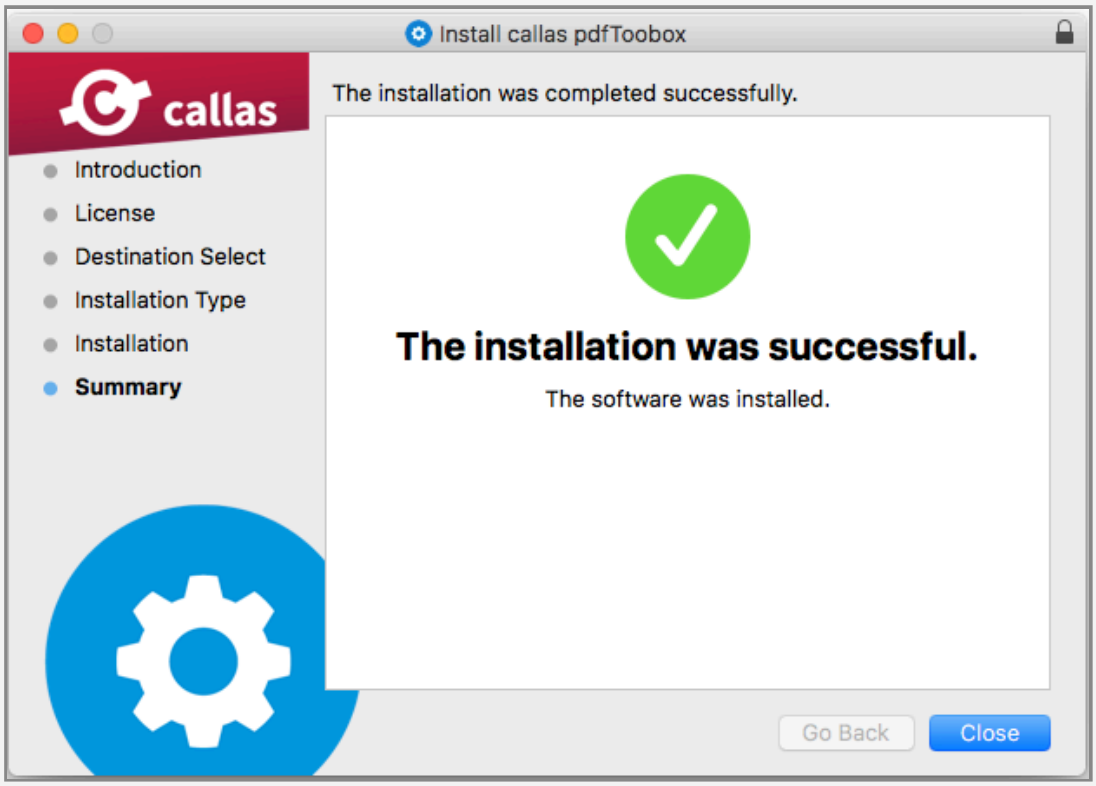
Installation type



It is possible to Install the Desktop version and the Plugin version together or only the Standalone version. If you do not wish to install the Plugin, click on "Customize" and uncheck the "Plugin" installation.

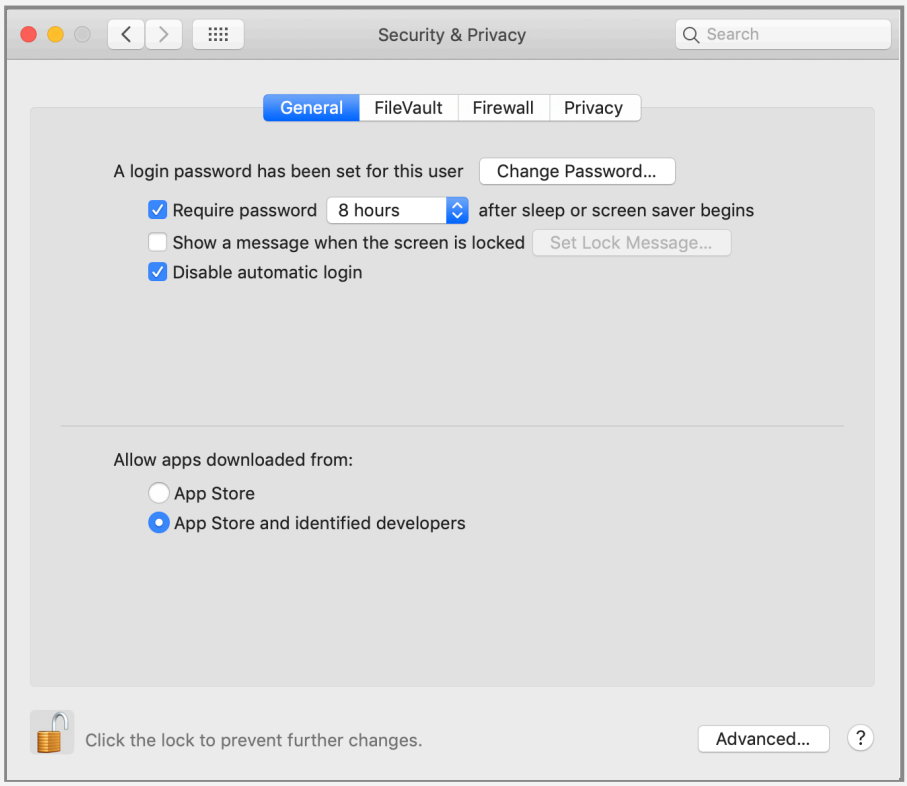


Installation was successful



Identify developer on Mac

callas software is a registered developer at Apple for MacOS. Due to some quite strict security settings starting with MacOS 10.7 only software downloaded from the Mac App Store can be installed directly. Just change the system settings in 'Security and Privacy' to 'Mac App Store and identified developers' before installing a callas software product. You can set it back to 'Mac App Store' after the installation ofcourse.



1.2 Installation of pdfToolbox Desktop or Server on Windows

You can download the latest version of pdfToolbox Desktop or pdfToolbox Server from our website by requesting a trial:

1. pdfToolbox Desktop: pdftoolboxdesktop
2. pdfToolbox Server: pdftoolboxserver

You can also register on our website for easy access to all pdfToolbox download links. Once registered and logged in, you can access the list of download links:

1. pdfToolbox Desktop: pdftoolboxdesktop
2. pdfToolbox Server: pdftoolboxserver

System requirements & hardware recommendations

Please refer to the callas website for system requirements and hardware recommendations for pdfToolbox Desktop and pdfToolbox Server:

- System requirements

Unattended installation

It is also possible to run the installtion without any user interaction.

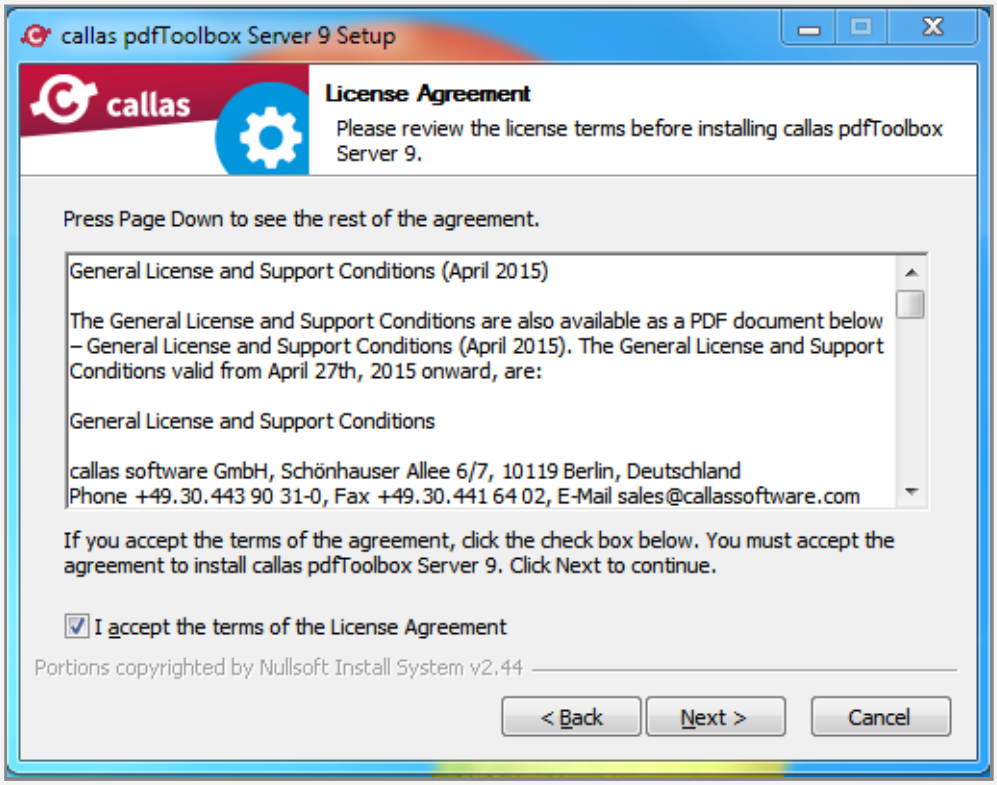
With the additional parameter /S the applicatin will be installed at the default location on Program Files.

```
C:\>"C:\Users\Administrator\Desktop\callas pdfToolbox 9.exe" /S
```

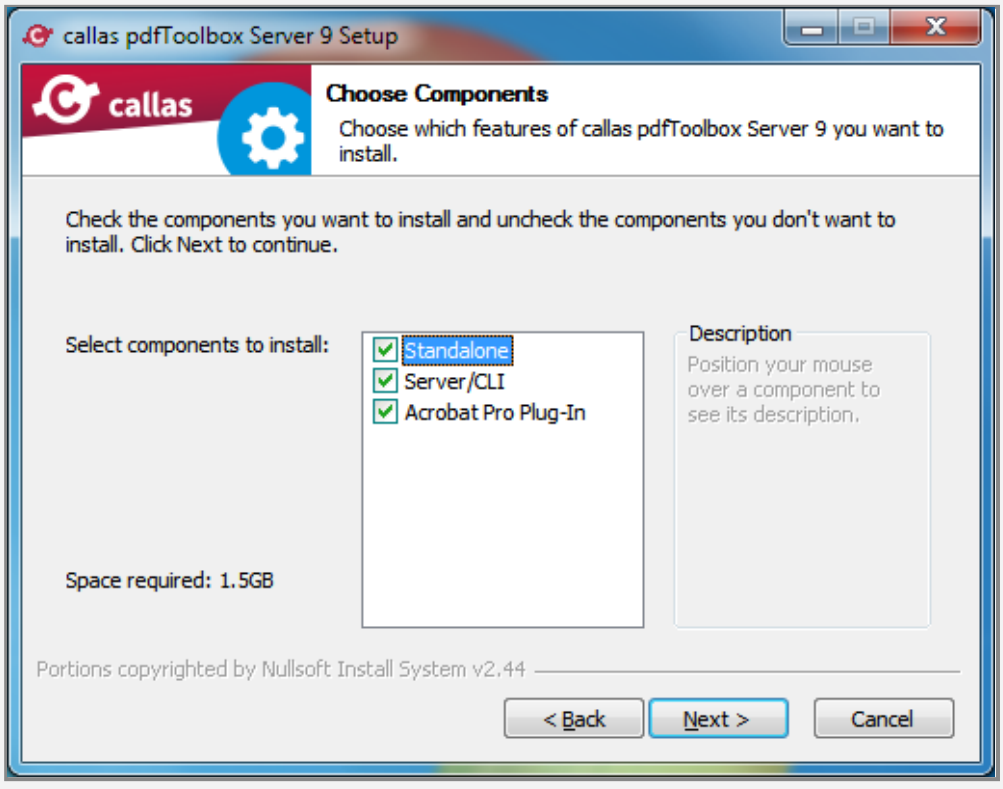
Installation with user interaction



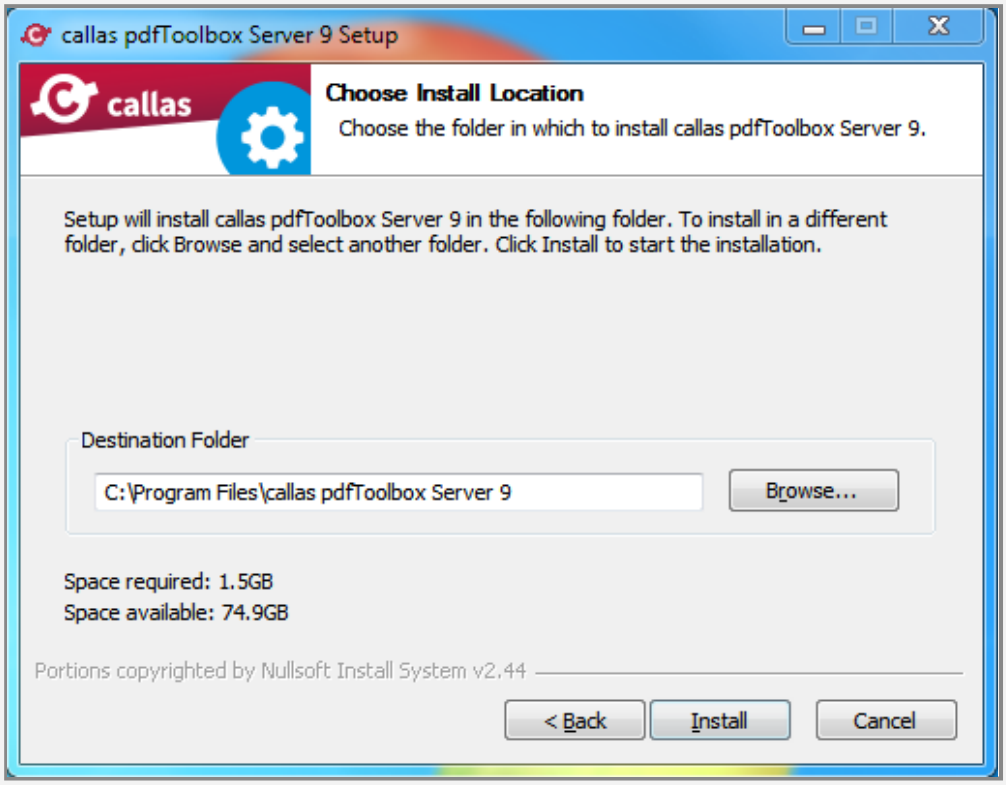
Short Introduction, continue with "Next".



Please read the License and Support Conditions carefully and click on "Next" if you agree.



Now select the components you want to install.



You can install the application on the default location in program files, else this can be changed by clicking on "Browse..." and setting path to the custom location.



Installation of pdfToolbox is now finished. Clicking on "Finish" will run pdfToolbox Desktop Standalone.

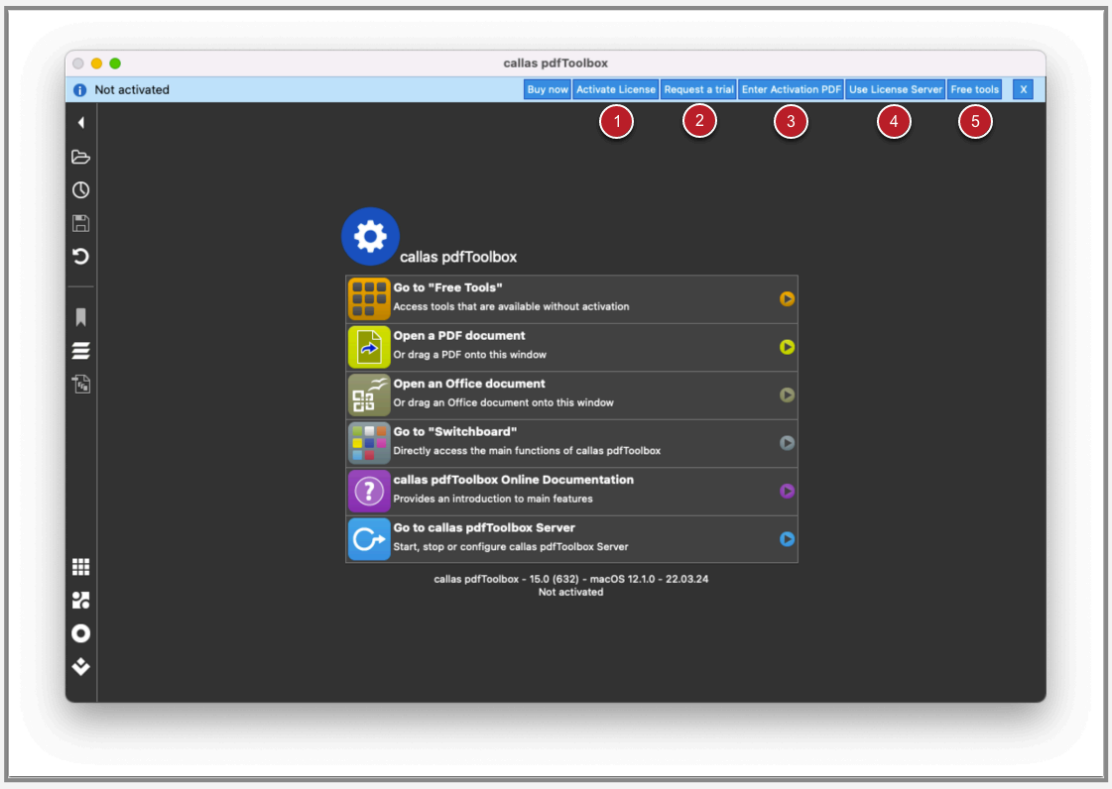
1.3 Activation and Deactivation of pdfToolbox

Main application window

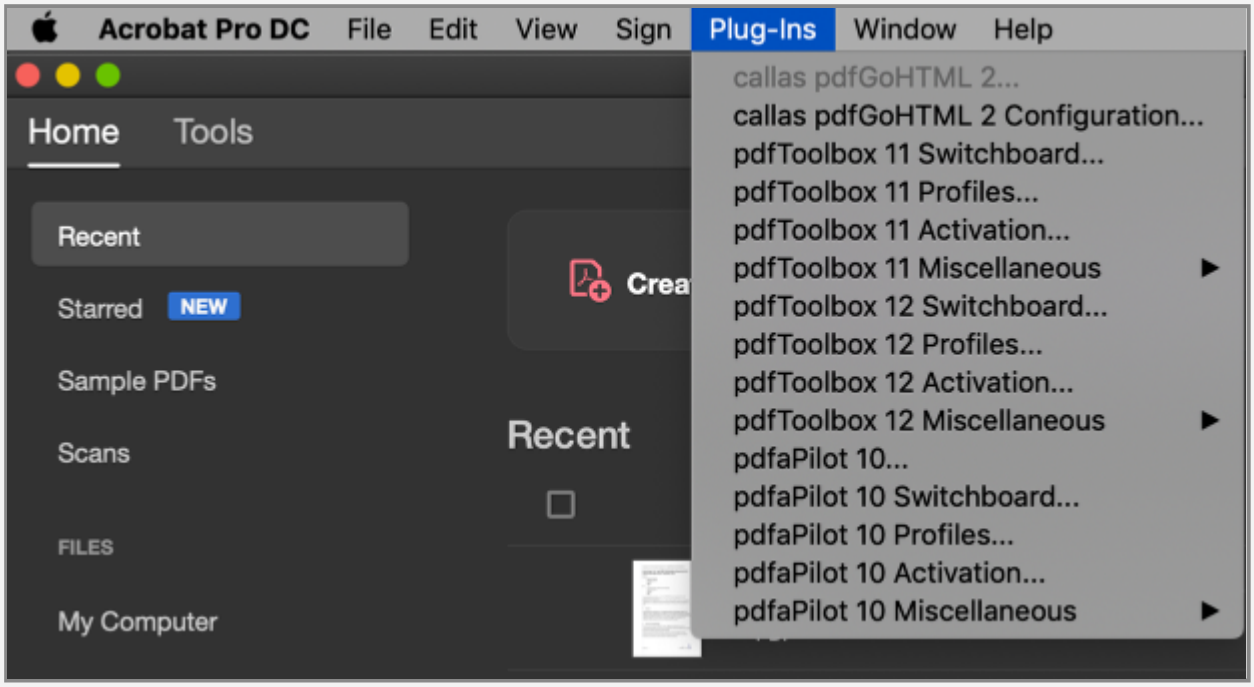
To use any callas product on a computer, you must first activate it. This is true if you have purchased a license key, but also if you want to run the trial software. This article explains how the normal activation process for pdfToolbox Desktop works and which steps you have to go through.

Launch the application. You will see the main window with a blue bar at the top (If you do not see a blue bar, go to Help > Activate callas pdfToolbox). You have the following options:

1. Activate License (if you have purchased a license key)
2. Request a trial (2-week trial license)
3. Enter Activation PDF
4. Use License Server
5. Free tools



Alternatively, activation with the Acrobat Plugin

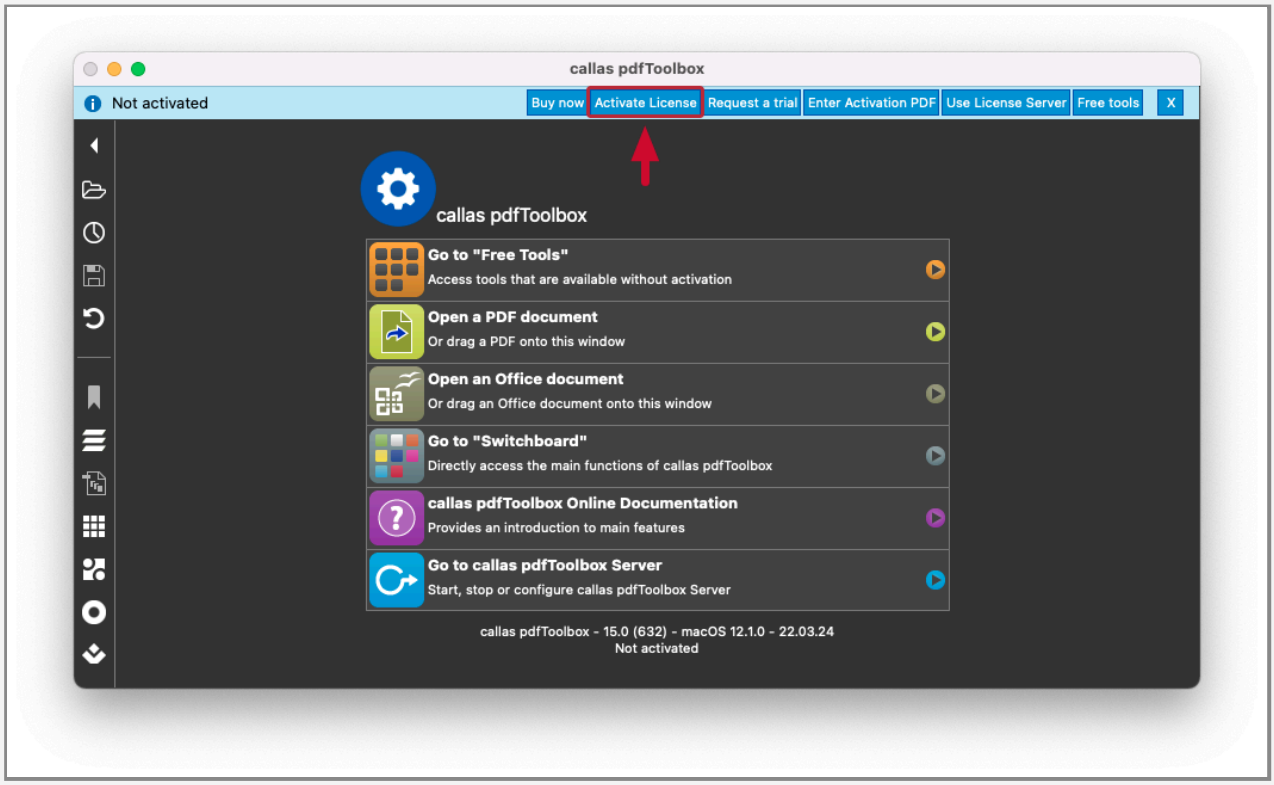


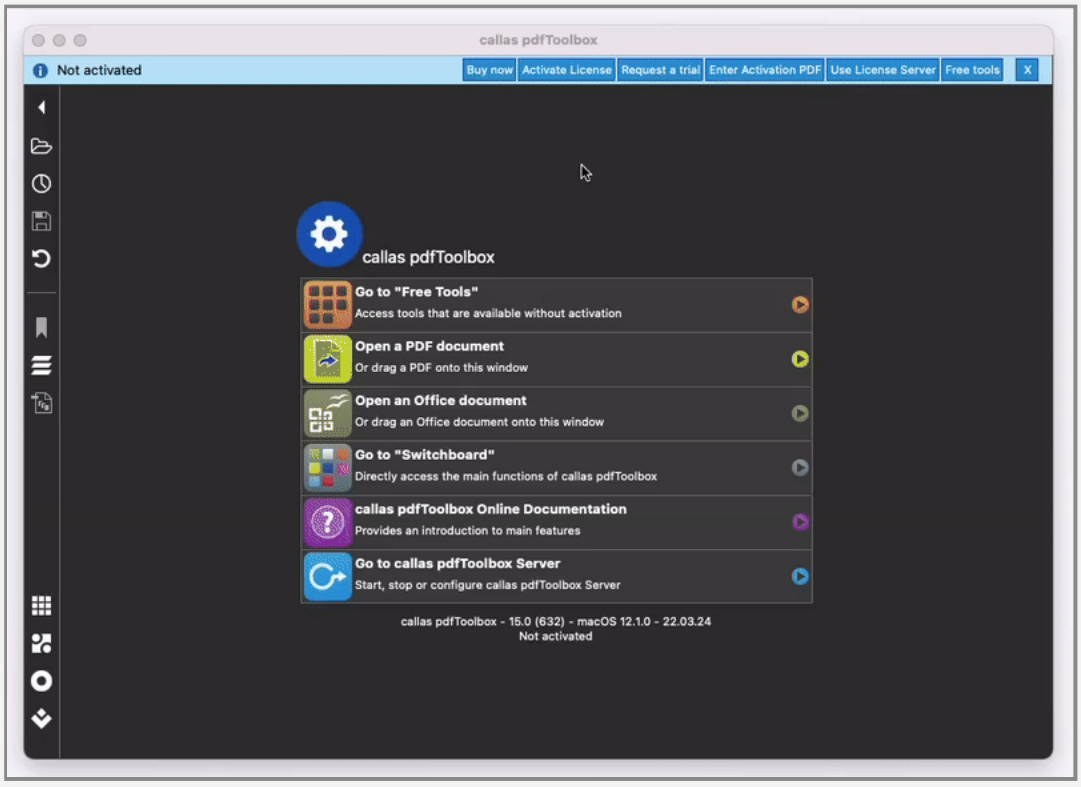
1. Activate the full version

If you have bought a license for callas pdfToolbox Desktop, click on the "Activate License" button. To active the full version you will need a valid license key that can be found on your License.pdf.

After clicking "Activate License" the Activation Window appears. Here you have to fill out your Name, Company, Email address and a valid License key. It is important to enter a valid email because the Activation PDF will be sent to that email address. You can enter the License Key by dragging and dropping your License PDF onto the lighter gray area.

Afterwards click the "Next" button. This will sent your activation request to the activation server. Please continue now with step 3.

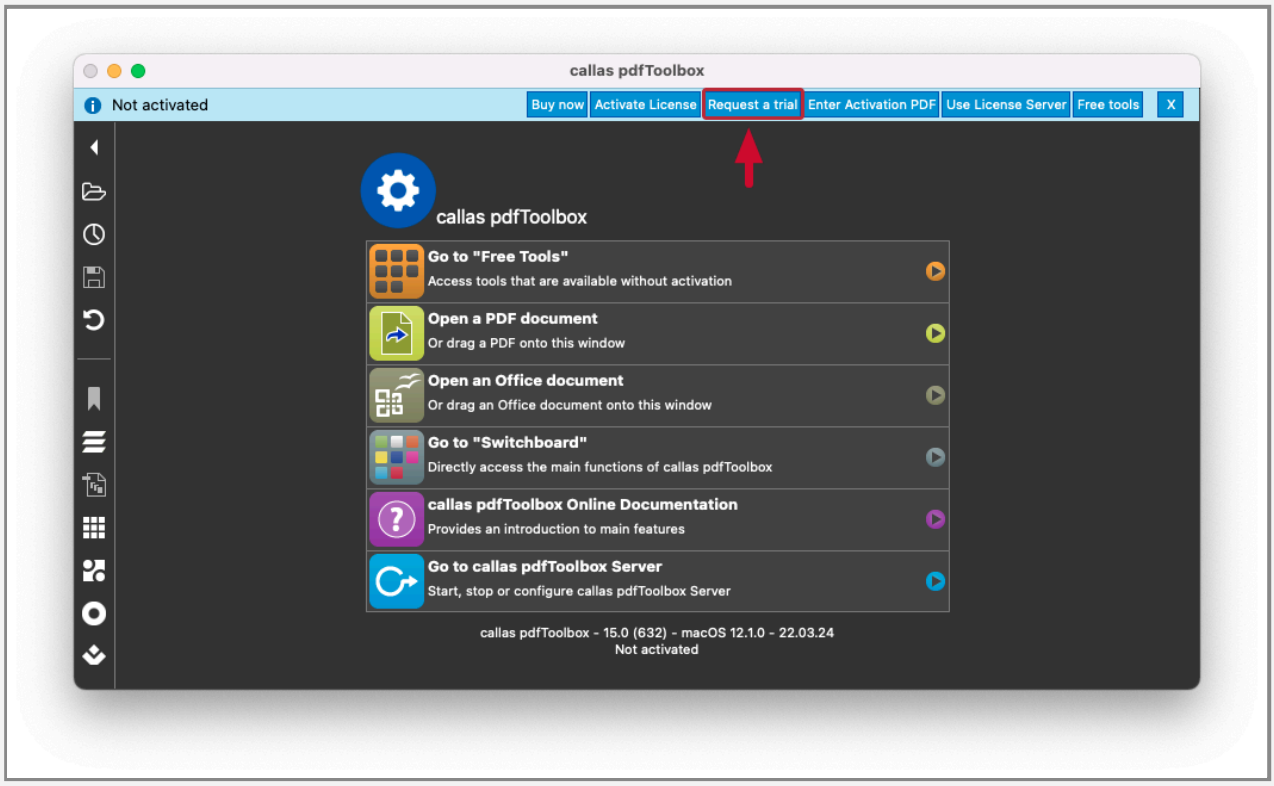

callas pdfToolbox
Not activated
Buy now
Activate License
Request a trial
Enter Activation PDF
Use License Server
Free tools
X
callas pdfToolbox
Go to "Free Tools"
Access tools that are available without activation
Open a PDF document
Or drag a PDF onto this window
Open an Office document
Or drag an Office document onto this window
Go to "Switchboard"
Directly access the main functions of callas pdfToolbox
callas pdfToolbox Online Documentation
Provides an introduction to main features
Go to callas pdfToolbox Server
Start, stop or configure callas pdfToolbox Server
callas pdfToolbox - 15.0 (632) - macOS 12.1.0 - 22.03.24
Not activated

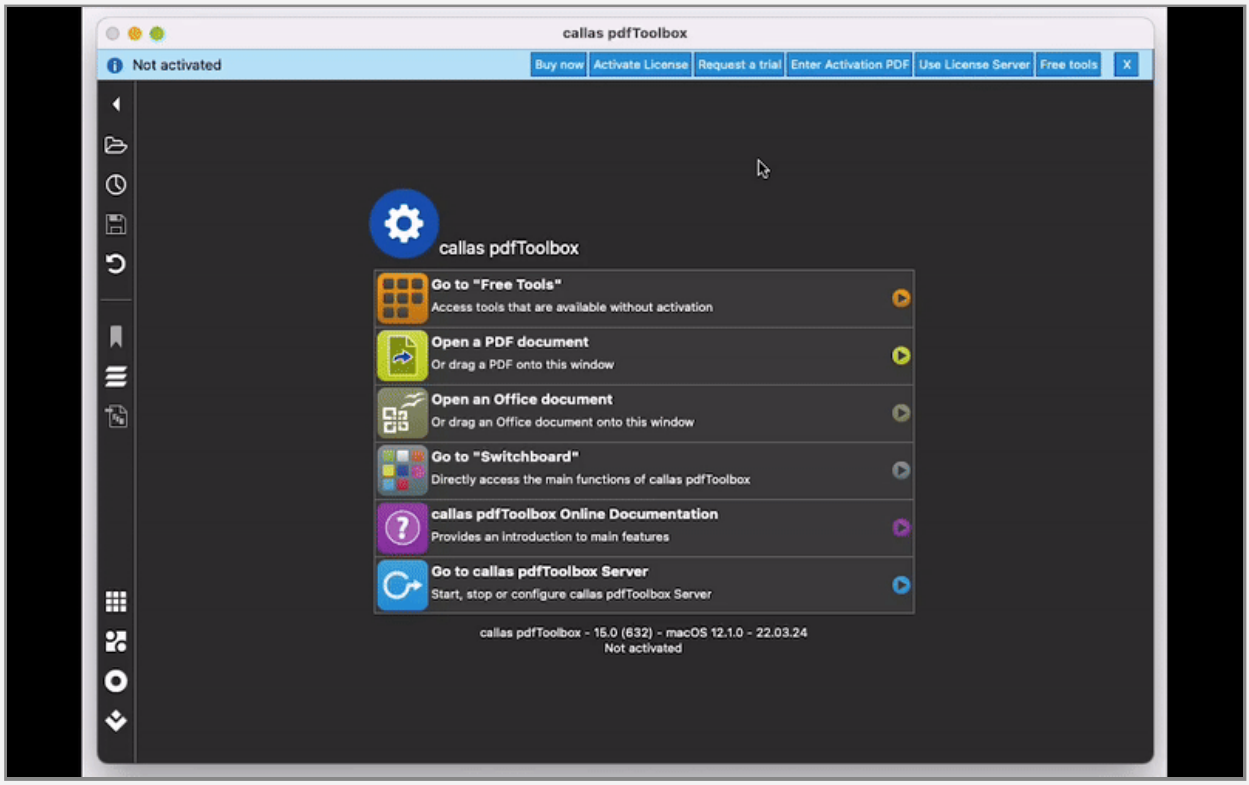

callas pdfToolbox
Not activated
Buy now
Activate License
Request a trial
Enter Activation PDF
Use License Server
Free tools
X
callas pdfToolbox
Go to "Free Tools"
Access tools that are available without activation
Open a PDF document
Or drag a PDF onto this window
Open an Office document
Or drag an Office document onto this window
Go to "Switchboard"
Directly access the main functions of callas pdfToolbox
callas pdfToolbox Online Documentation
Provides an introduction to main features
Go to callas pdfToolbox Server
Start, stop or configure callas pdfToolbox Server
callas pdfToolbox - 15.0 (632) - macOS 12.1.0 - 22.03.24
Not activated

2. Request a trial version

If you simply want to try the product, click on "Request a trial". The Activation Window appears. Here you have to fill out your Name, Company and Email address. You can select which version of the software you want to activate a trial for. It is important to enter a valid email because the Activation PDF for your trial will be sent to that email address.

Afterwards click the "Next" button. This will sent your activation request to the activation server. Please continue now with step 3.





3. Activate pdfToolbox

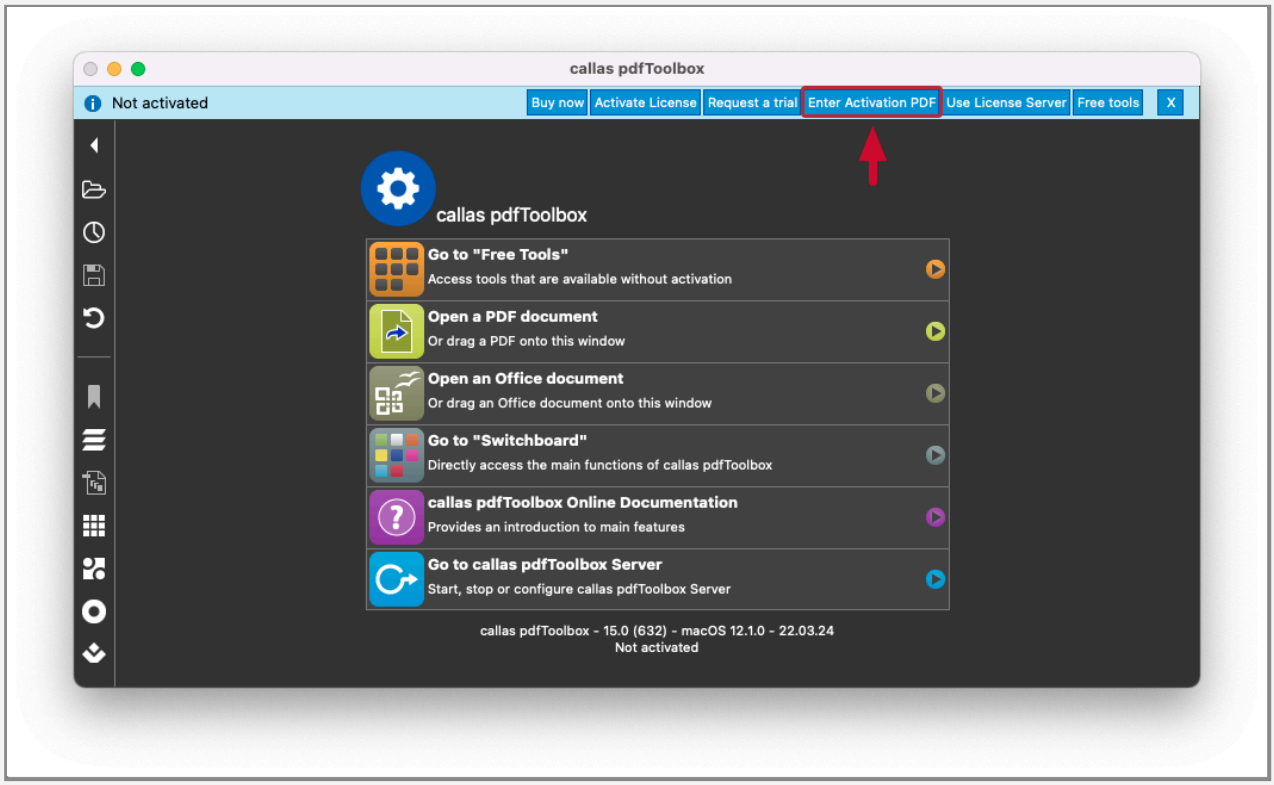
After submitting your details (and clicked on the "Next" button), it will typically take only a few seconds to receive an email from the callas activation server containing an attachment called "Activation.pdf".

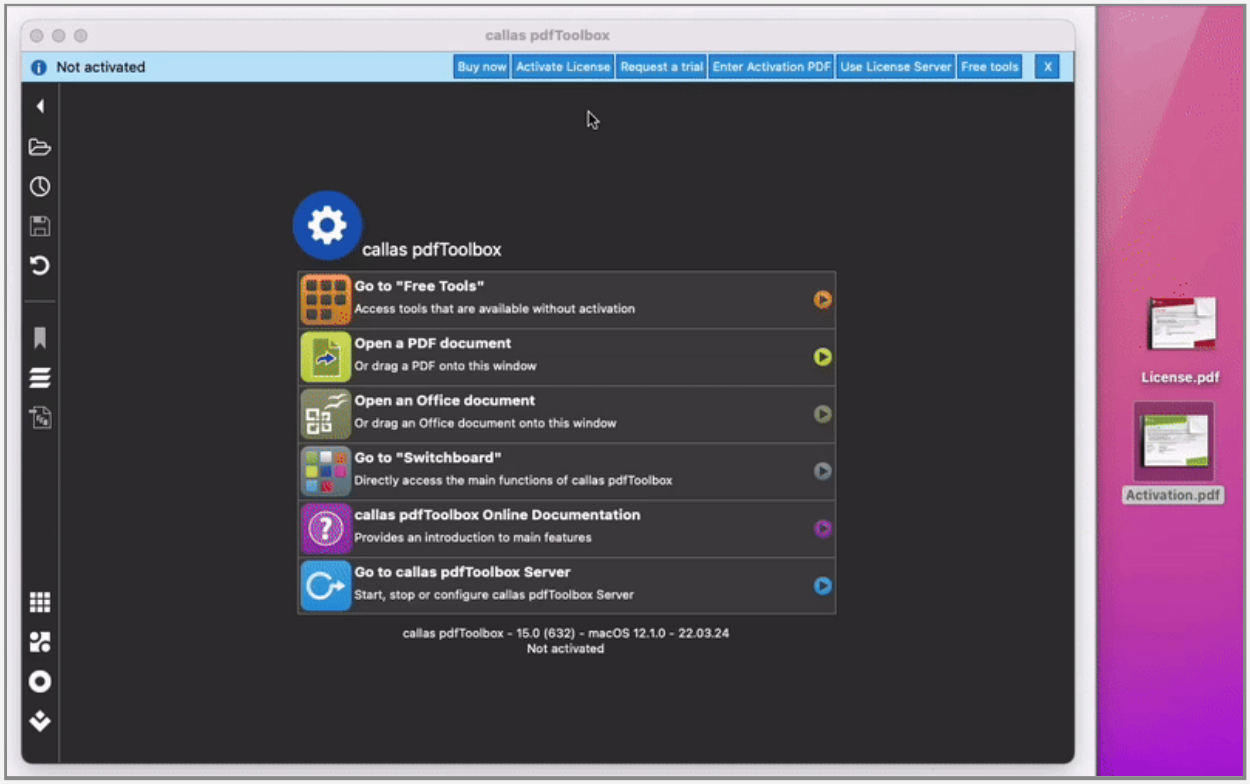
If no response is received or in the event of an error, please contact support@callassoftware.com to determine the exact cause.

NOTE: The Activation.pdf is only **valid for 48 hours** after being requested. After this timeframe, a new Activation.pdf has to be requested from the activation server.

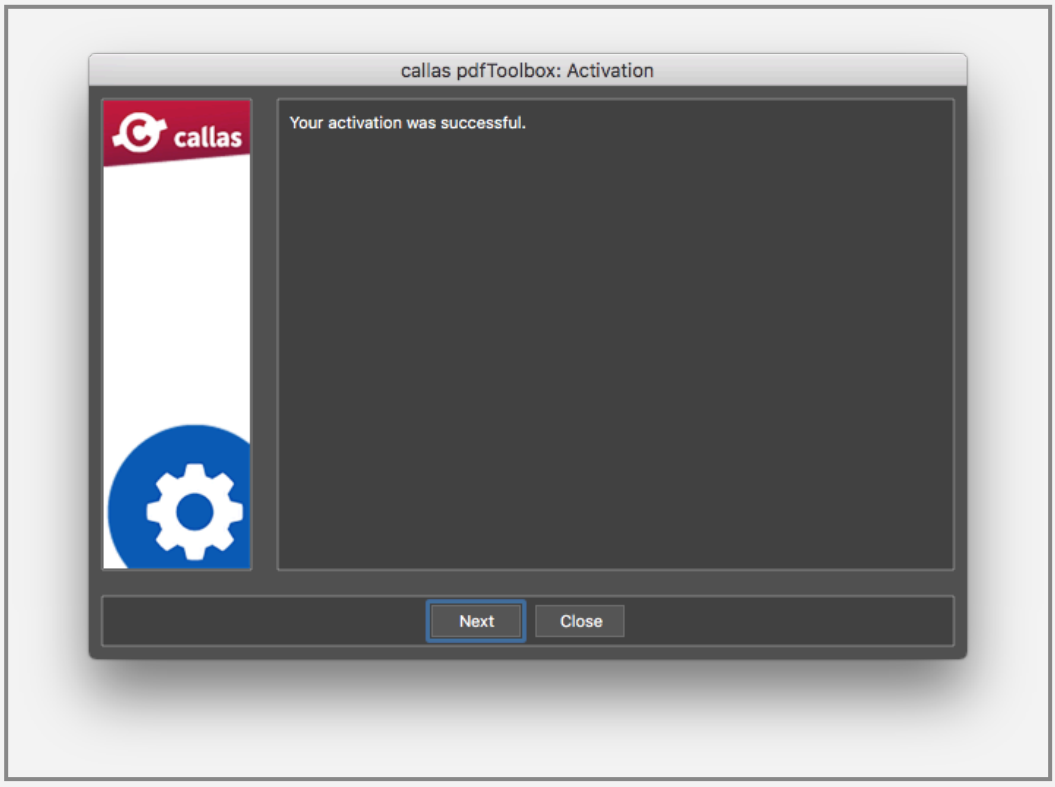
You can either drag and drop the Activation.pdf file onto the lighter gray area or copy and paste the information from the email into the field below. It is generally safer to drag and drop the file than to copy and paste the information. Drag-

ging and dropping the file will automatically fill in the information in the field.





Click "Next" and your callas pdfToolbox Desktop will be activated. You can click "Close" and start using the software.



Activate pdfToolbox without internet connection

You can also install and activate callas software on a system that doesn't have an Internet connection. In order to make this work, you can do an off-line activation. Follow to normal activation procedure until the point where you have entered your details:

1. Click on this "Details" button to get the activation information.
2. Click on the "Send via E-Mail" button to reveal the full email text that needs to be sent to the activation server.
3. Copy and paste this text into a text document and take it to a system where you have email access. Create a new email with the subject, content and "To" address as described.
4. Once you receive the response email with the attached “Activation.pdf”, take this file back to the system without

Internet access and use it to go through the rest of the procedure.

Deactivation

As the activation (and the resulting license file) is bound to the hardware, it is necessary to deactivate a license on one machine before an activation takes place on a new machine.

1. Launch the application.
2. Go to "Help > About callas pdfToolbox".
3. Click "Deactivate".
4. Choose the product "callas pdfToolbox (Desktop)".
5. Fill out your email address.
6. Click "Next".
7. Click "Yes".

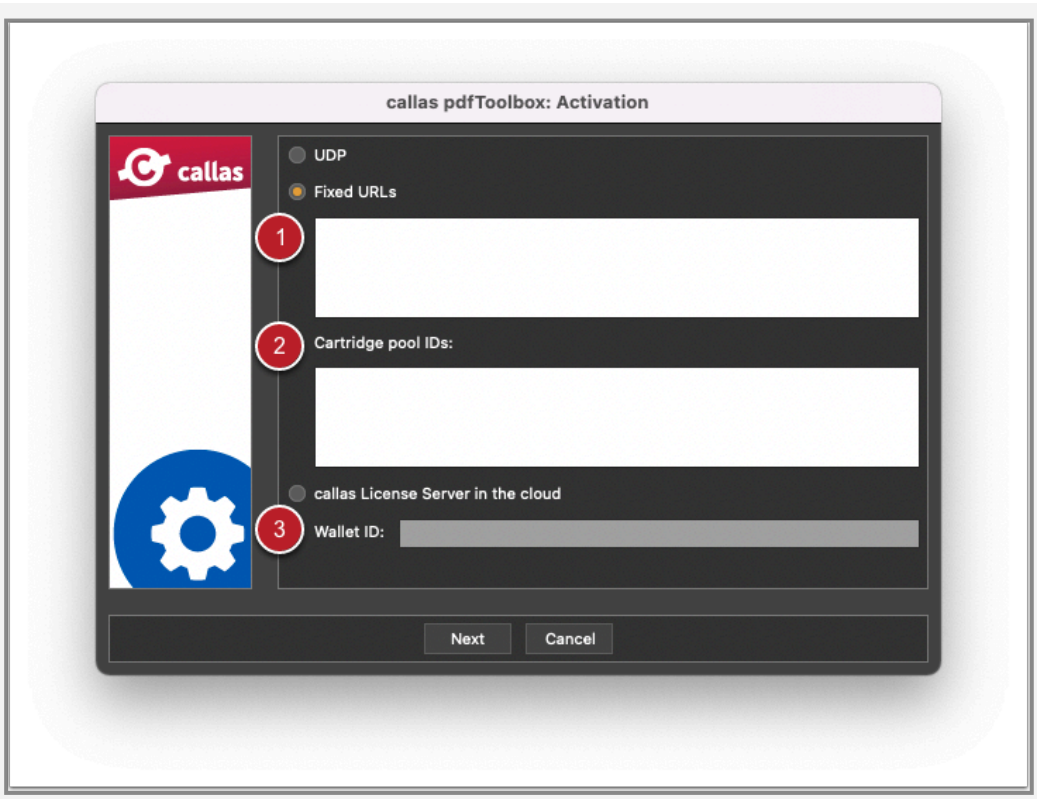
After the procedure, your callas pdfToolbox Desktop license will be deactivated.

NOTE: The deactivation procedure is the same for DeviceLink Add-on Desktop (in step 3 you have to select the product "callas DeviceLink Add-on (Desktop)").

4. Use License Server

To run pdfToolbox Desktop without activating a hardware-bound licence, you can use the Licence Server. Here you have several options:

1. To use the License Server on-premise to activate pdfToolbox Desktop, you must enter the IP address of the server here. More information on how to activate the License Server on-premise can be found here.
2. Input field to specifying a cartridge pool. More information can be found here.
3. To use the License Server (SaaS solution) to activate pdfToolbox Desktop, you must enter the Wallet ID here. More information on how to use the License Server can be found here.



5. Free tools

pdfToolbox also includes free tools that can be used without activation. The tools allow detailed insight into PDF files, for example into the internal structure, fonts, metadata, existing DPart or tagging structures or the reading of barcodes and QR codes. pdfToolbox also contains free tools that can be used without activation. The tools allow detailed insight into PDF files, e.g. the internal structure, fonts, metadata, existing DPart or tagging structures or the reading of barcodes and QR codes.


callas pdfToolbox
Free tools
Prepress, Color and Transpi
Explore PDF
Explore Fonts
Explore Metadata
Explore Layers
Explore Tagging
Explore DPart
Test mode
Read barcode
Licensed to Sarah Nitschmann (callas software)

1.4 Multi-user system (Activating additional users on the same system)

pdfToolbox can be operated by multiple users on the same computer.

We'll show you where to copy the license file to in order to allow multiple users to access pdfToolbox.

License file for individual users on the same system

pdfToolbox licenses are only bound to a given system, not to specific users.

During activation, the license file is copied to the user preferences of the currently active user.

Please make sure that the user has sufficient write access to this directory.

In order for other users to use this license, the License.txt file must be copied manually to the user preferences of the other users.

To allow other users to work with this license, you must manually copy the **License.txt** file into the User Preferences for the other user.

The path to the folder where the License.txt is copied to make it available to other individual users is as follows:

MacOS:
/Users/<USERNAME>/Library/Preferences/callas software/callas pdfToolbox <VERSION>

Windows:
%AppData%\callas software\callas pdfToolbox *<VERSION>*

It is recommended to store the License.txt in the following path if pdfToolbox should be activated for all users:

MacOS:

/Library/Preferences/callas software/callas pdfToolbox *<VERSION>*

Windows:

%allusersprofile%\callas software\callas pdfToolbox *<VERSION>*

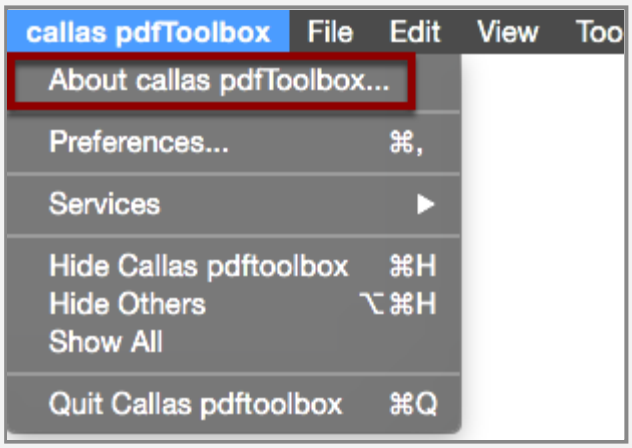
pdfToolbox CLI/Server activation for all users on one system

If the pdfaPilot CLI/Server version shall be activated for all users on a system, there are various possibilities. Details can be found in the CLI manual under "Activating pdfToolbox CLI".

1.5 Migrating the pdfToolbox to another computer (Deactivation)

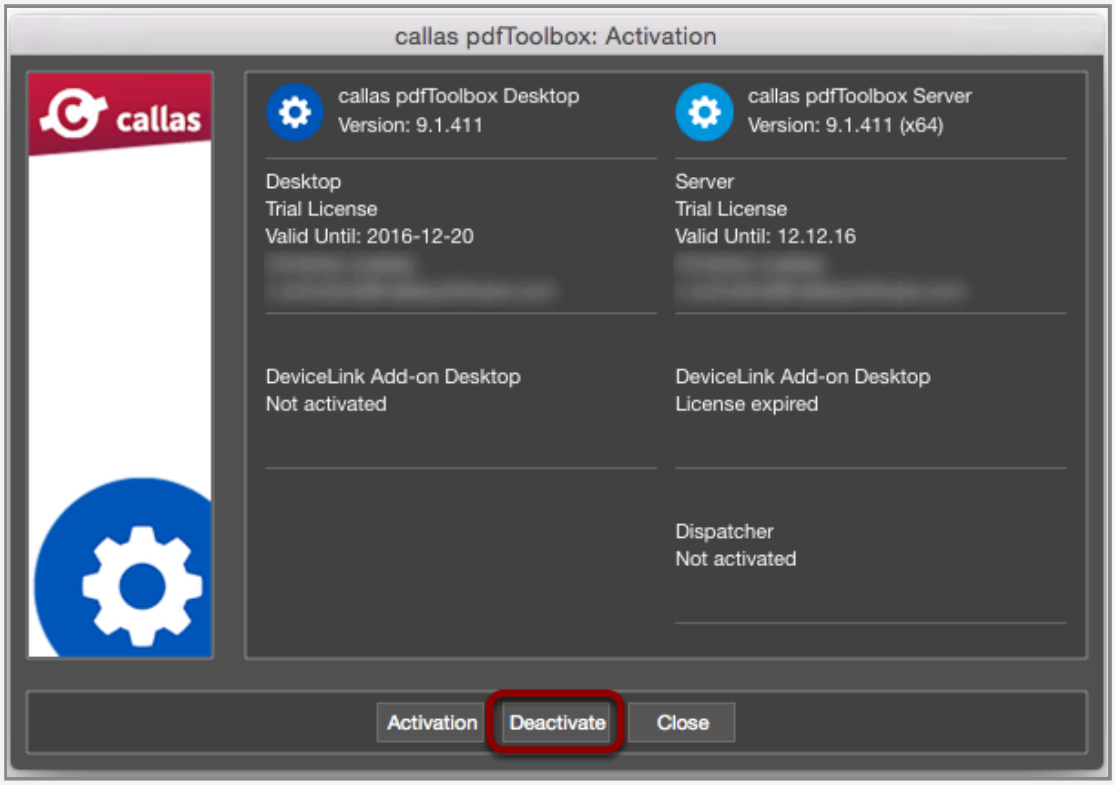
If you wish to use pdfToolbox on another computer - for example, because you have acquired a new device - you must first deactivate the application on the last computer used.

Open the “About pdfToolbox” menu item



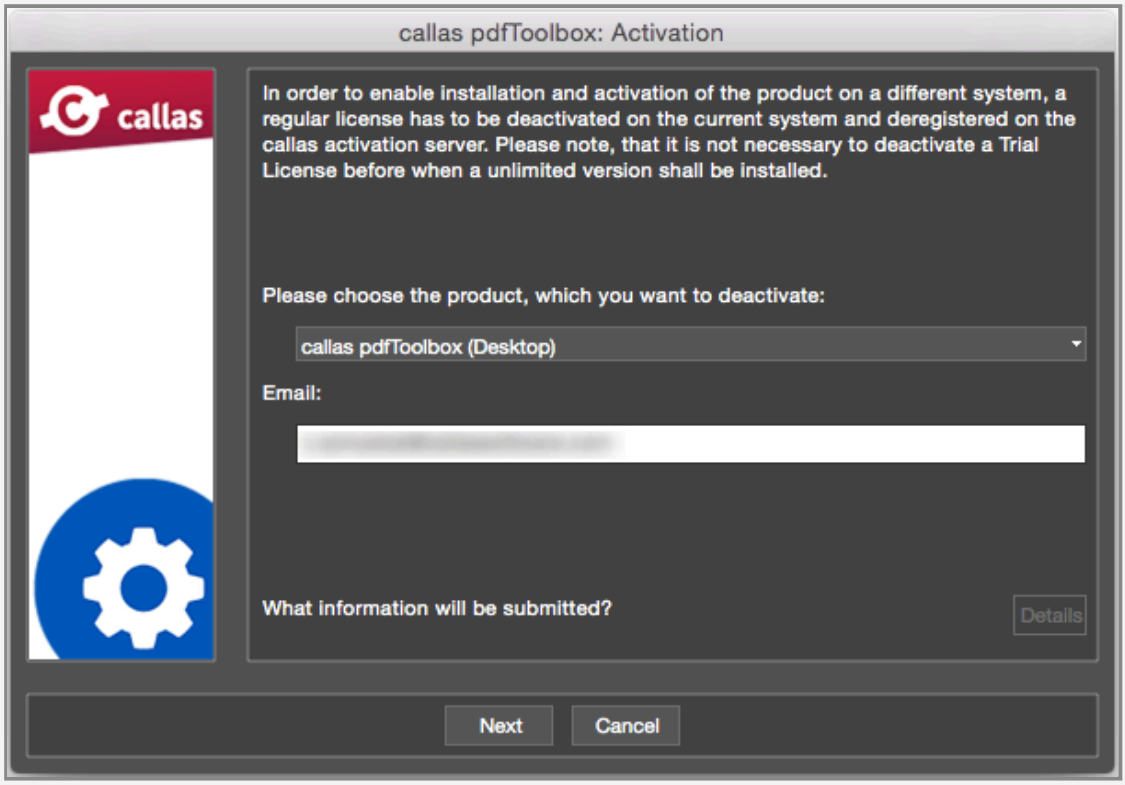
Options to **Activate** and **Deactivate** pdfToolbox can be found under the following menu item: **About callas pdfToolbox...**

The “Activate” window



In the Activate window, you can deactivate the pdfToolbox by clicking **Deactivate**.

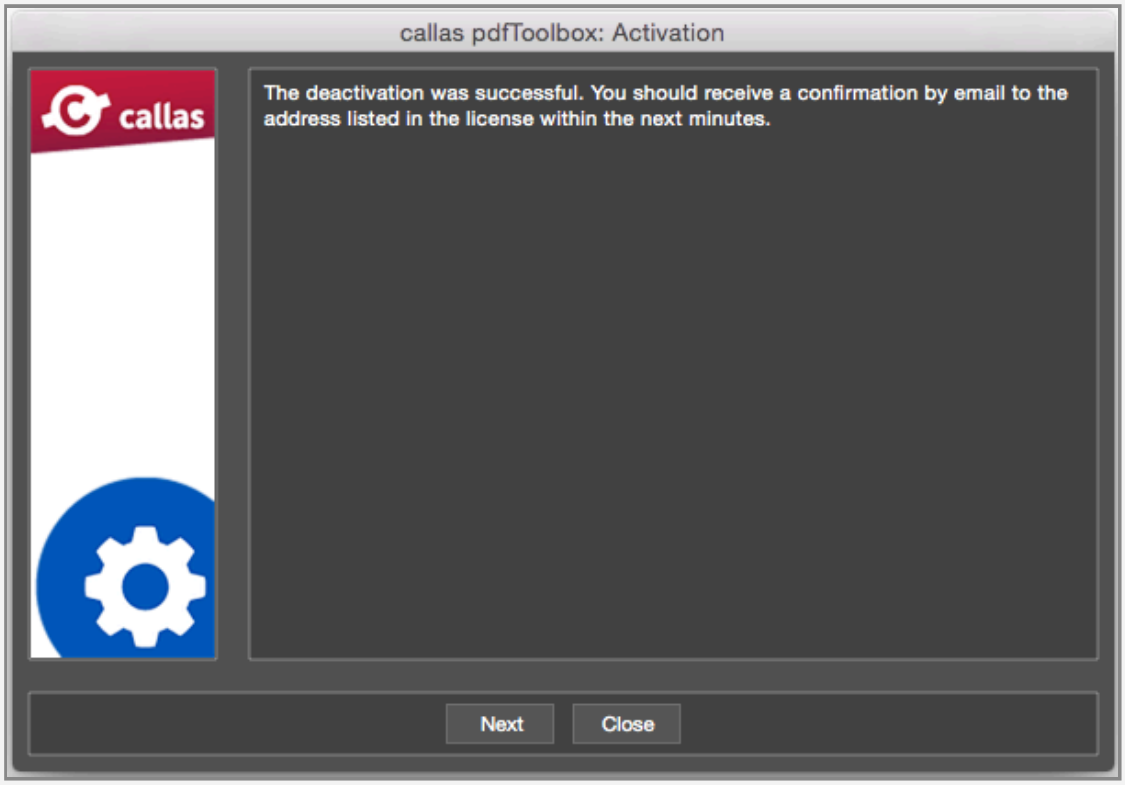
Deactivating license: Product and email



In the dialog which opens, you can select the **software to deactivate** (if you use multiple callas software products) and, if necessary, change the **email address** to which the confirmation message will be sent.

Click **Continue** to start the process.

Notification and deactivation email



pdfToolbox will report **Successful deactivation** and inform you that an **email confirmation** has been sent.

After you have received this deactivation notification by email, you can activate pdfToolbox on another machine using the original "License.pdf" which has to be valid for the installed version.
Please note: a previously received "Activation.pdf" can only be used on the machine where it has been requested from and only within 48 hours after receiving it.

To learn how to activate pdfToolbox, visit the support article Activation procedure: callas pdfToolbox Desktop.

Move your pdfToolbox settings to another machine

If you have created a number of own Profiles etc., the most easiest way to move your settings is to export the respective

Library as described here and copy the created .kfpl file to the new machine.

However, there might be cases, where a direct access to the stored user data is needed.
Your pdfToolbox settings are stored in the following location:

Windows:
C:\Users\USERNAME\AppData\Roaming\callas software\
callas pdfToolbox ...

Mac OS X:
/Users/USERNAME/Library/Preferences/callas software/
callas pdfToolbox ...

Replace "USERNAME" in the path above with your username, and copy the contents of this folder to the same location on other computer.

1.6 Uninstalling the pdfToolbox Acrobat plug-in

The pdfToolbox plugin can be easily deleted in just a few steps. We'll show you where the relevant packages can be found depending on the operating system and Acrobat version.

Removing the pdfToolbox Plug-in on MacOS

The pdfToolbox plug-in for Acrobat can be easily removed by deleting the **pdfToolbox__.acroplugin** package in the callas software subfolder in the plug-ins directory for Acrobat, under Mac.
Note: The file name contains the respective pdfToolbox version number at the "__" position.

Acrobat DC

Please note that as of Acrobat DC, Adobe has changed the folder in which it stores third-party plug-ins:
/Library/Application Support/Adobe/Acrobat/DC/Plug-ins/callas software

Acrobat XI and older

For Acrobat XI and older, meanwhile, the plug-in can be found in the following folder:
/Applications/Adobe Acrobat XI Pro/Adobe Acrobat Pro.app/Contents/Plug-ins/callas software

Removing the pdfToolbox Plug-in on Windows

On **Windows**, the plugin can be found in the subfolder of the plugin directory for the current **Acrobat** version:
Acrobat\plug_ins\callas software

The folders and files contained within can be identified by their name, e.g. *"pdfToolbox 9" or "pdfToolbox 13".*

Delete them to remove the plug-in.

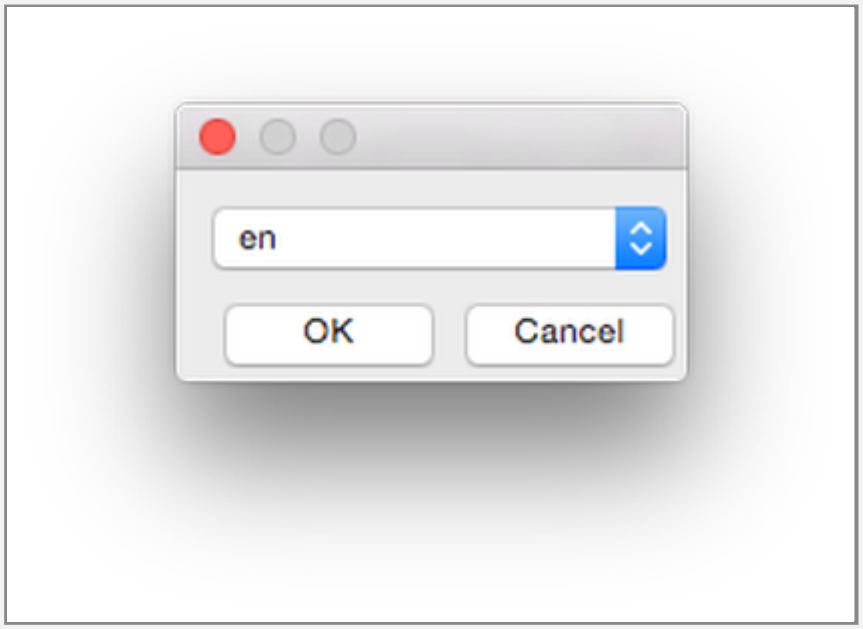
1.7 Change language of user interface (desktop only)

Normally, pdfToolbox Desktop will launch in the same language in which the the operating system is running (or English if the language in which the operating system running is not supported by pdfToolbox). It is possible to switch the language of the user interface to any other language supported by pdfToolbox Desktop. In order to do so, **press the Cmd/Ctrl key while launching pdfToolbox**.

A small dialog will appear, with a popup menu indicating the codes of the available languages, for example "de" for German or "fr" for French.

After selecting the desired language, simply click OK, and pdfToolbox Desktop will launch in the selected language.



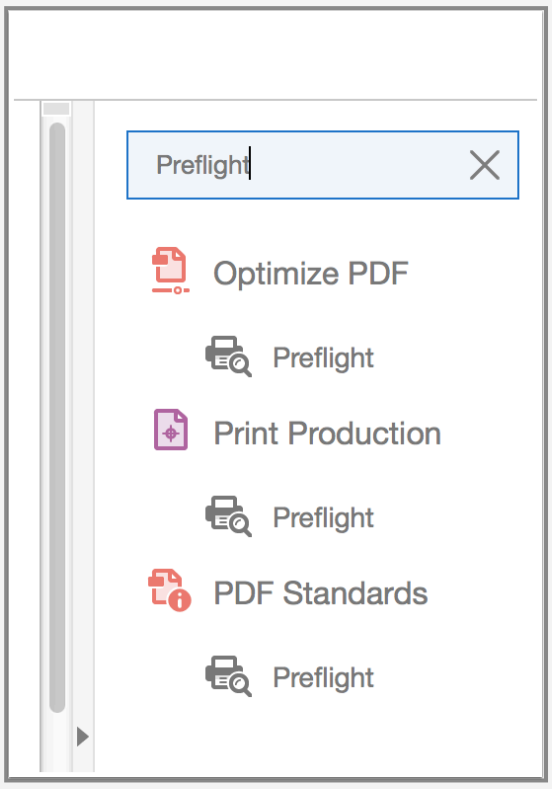

en
OK
Cancel

1.8 Using pdfToolbox and Preflight in Acrobat Pro parallel

**A lot of users want to use the Acrobat Preflight functionality as well as the high number of additional tools from pdfToolbox simultaneously.
This tutorial explains, how this parallel usage of Acrobat Preflight (a development of callas software, which has been integrated by Adobe as a part of Adobe Acrobat Pro since 2003) and callas pdfToolbox works.**

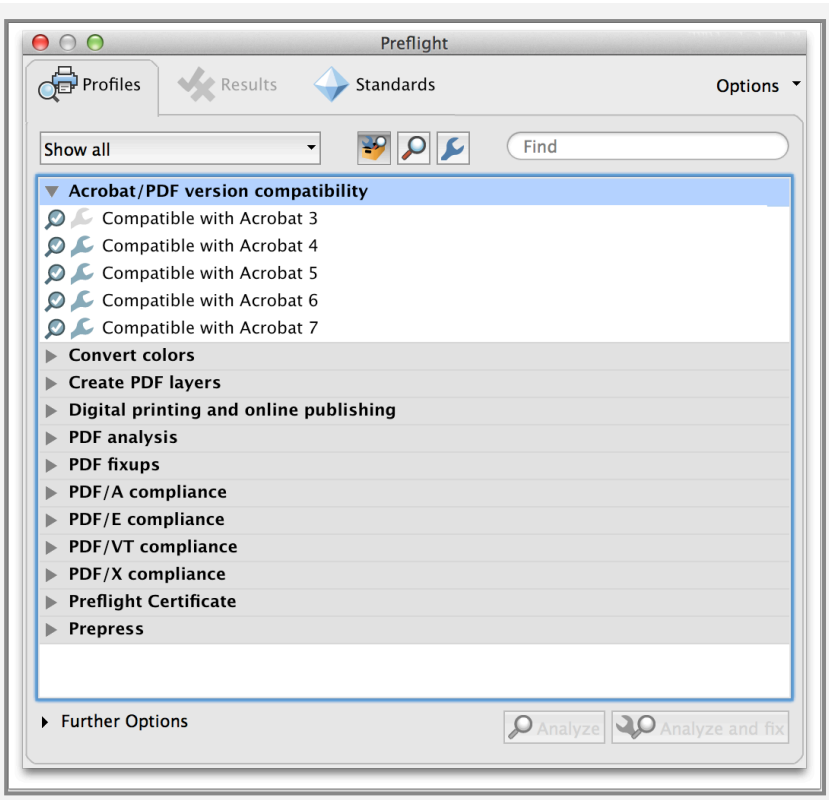
Open Acrobat-build-in Preflight

The Preflight functionality in Acrobat Pro can be found using the "Tools" sidebar for example or with the shortcut "CMD, Shift + X" (in MacOS) or "CTRL, Shift + X" (using Windows).



The Preflight Profiles window

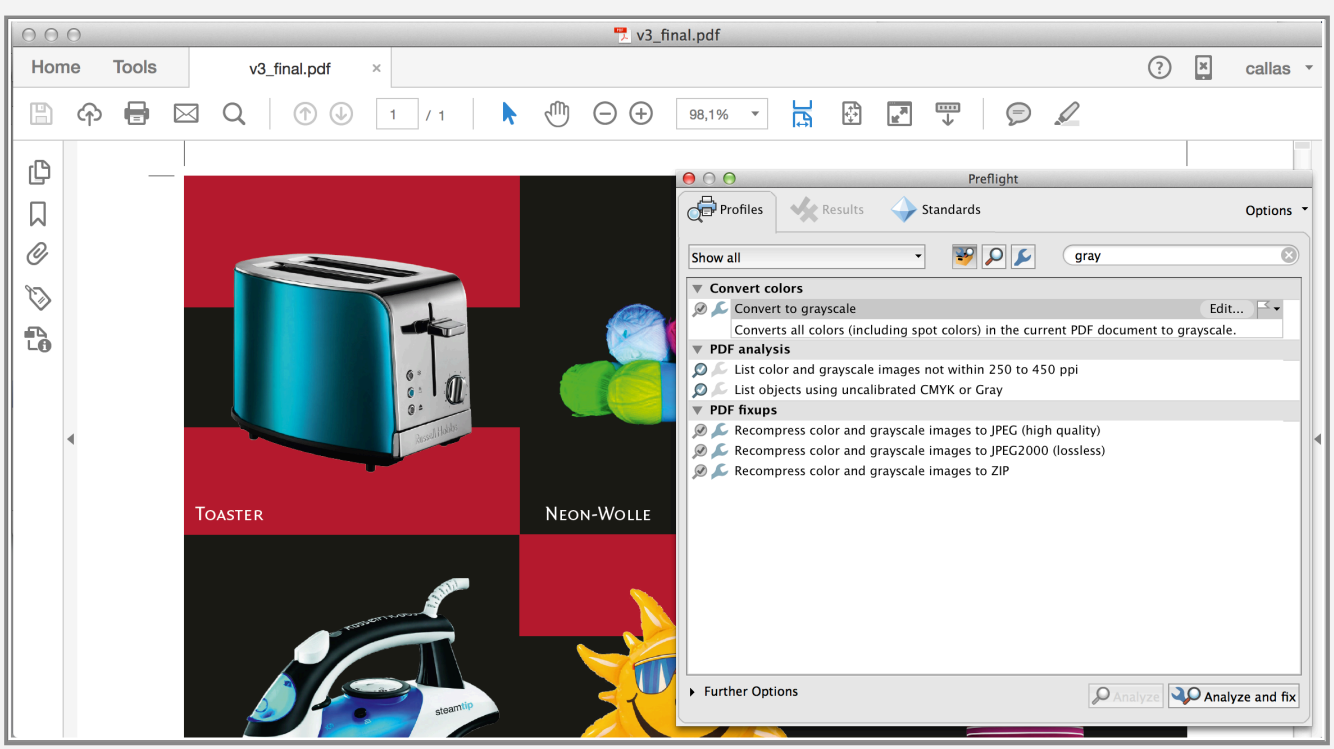
The Profile window lists a number of Profiles, Checks and Fix-ups. Also a separate "Standards" area gives easy access for conversion of PDF files into PDF-based ISO standards.



Process a PDF

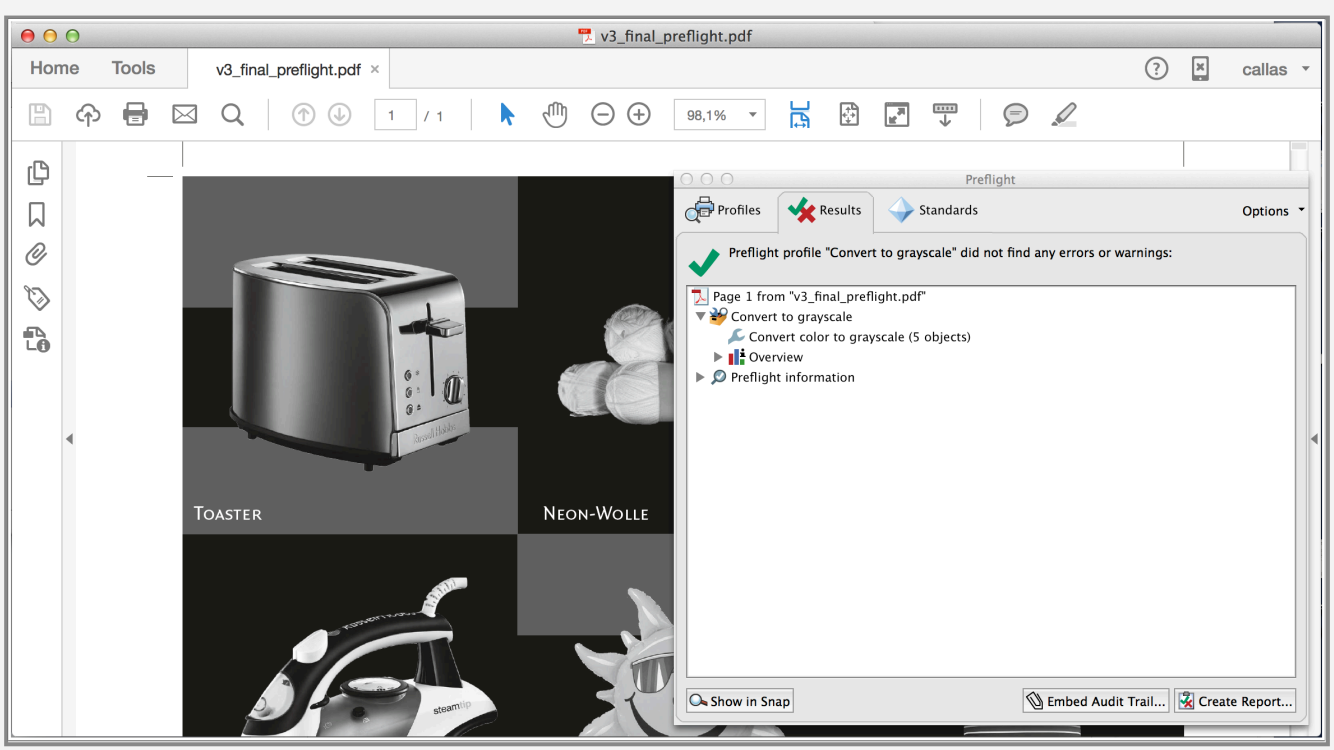
When a PDF is open, a fix or an analysis can be executed on that file.

Using Profiles, which only contain checks and/or validation of standards (and with Checks within the respective group), only the "Analyze" button is active. When a Profile contains Fixups and conversions into a PDF standard (and for single Fixups of course), "Analyze and fix" can be used.



Result dialog

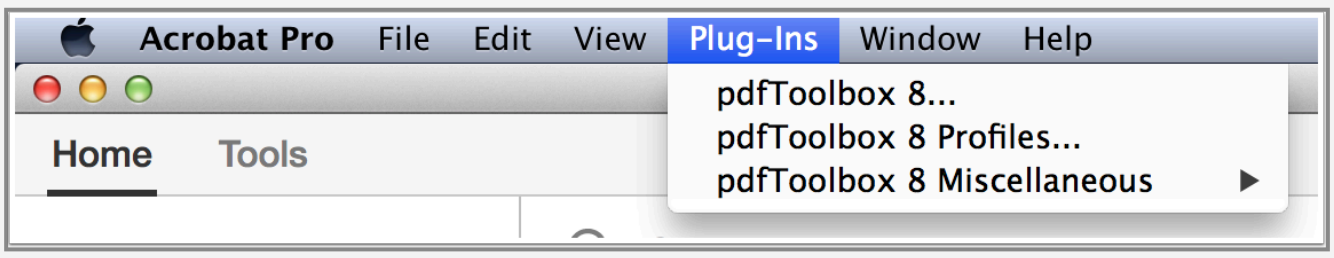
After processing, the result is shown and can be reviewed.
A click on the "Profiles" tab within this window allows further processing or analysis.



pdfToolbox as an Acrobat Pro Plug-In

Using the pdfToolbox installer, a Desktop version and a Plug-In for Acrobat Pro can be installed (the latter of of course only if Acrobat Pro is installed on the respective system).

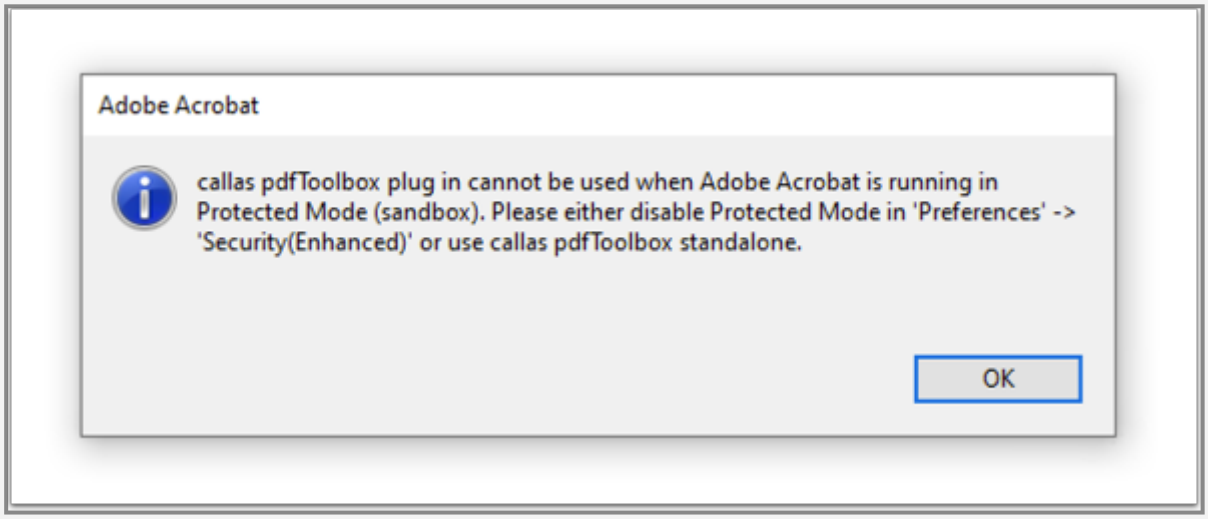
pdfToolbox can be found using the "Plug-Ins" menu or within the "Tools" sidebar.

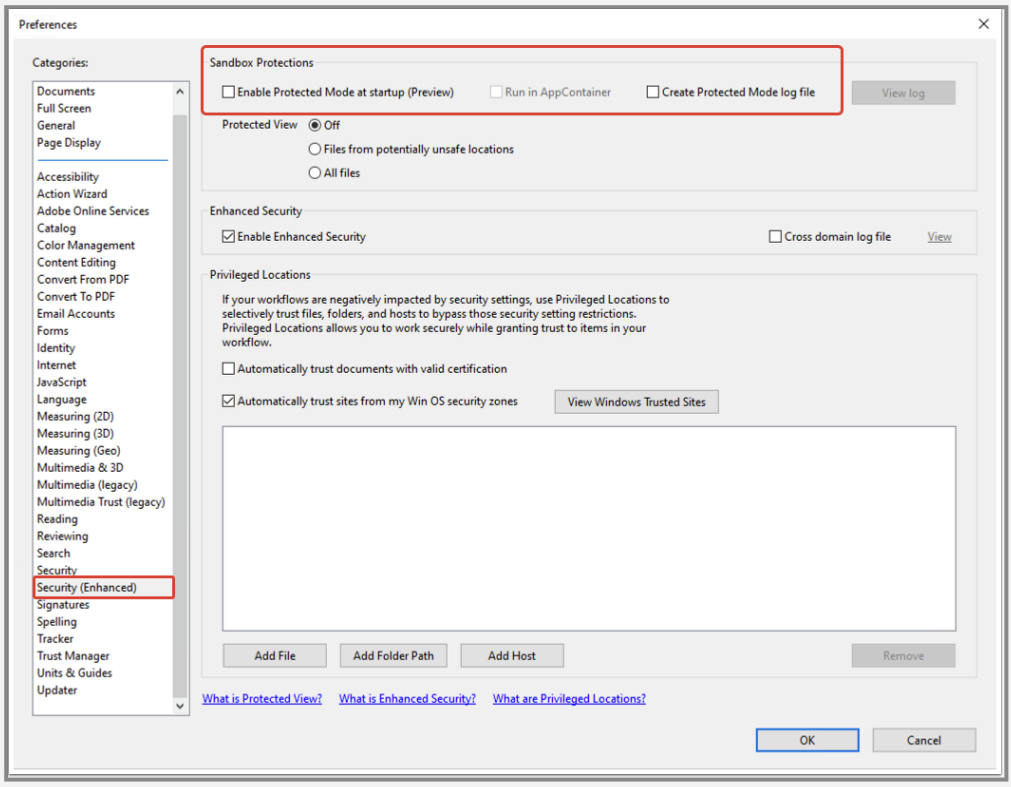


Using the pdfToolbox Plug-In on Windows

If the pdfToolbox Plug-In is not displayed in Adobe Acrobat Pro after installation, the security settings

should be checked. **The Plug-In for Acrobat cannot be used in protected mode (sandbox mode).** This mode must first be deactivated under "Preferences" > "Security (Enhanced)" (see 2nd screenshot). Then Acrobat must be restarted so that the Plug-In can be used.

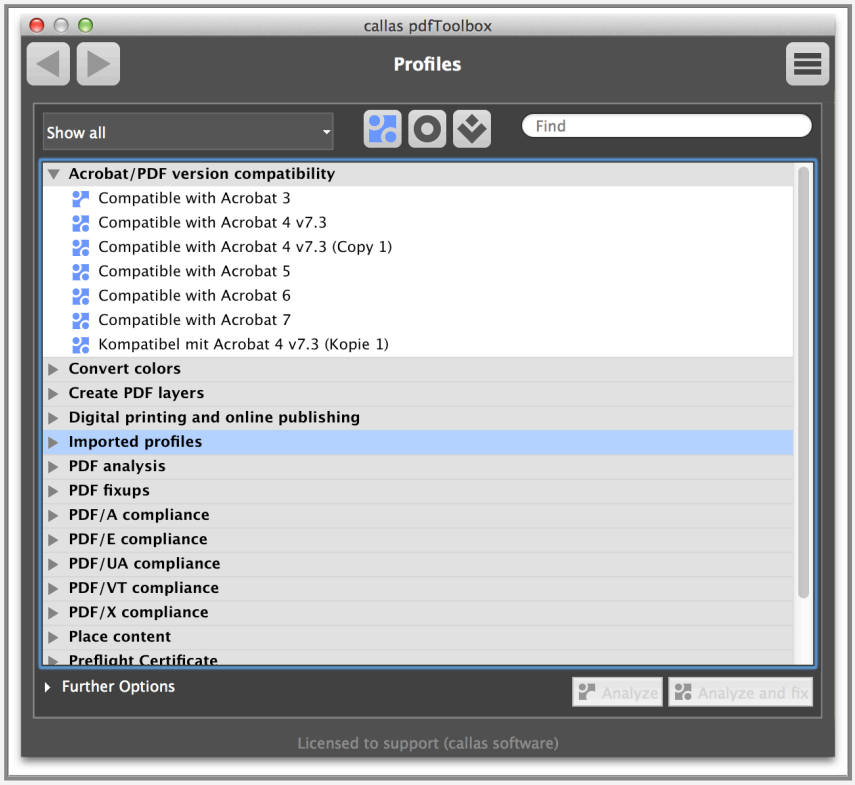




Open the pdfToolbox Profiles window

A click on the "pdfToolbox Profiles" entry opens a window with a number of predefined Profiles, Checks and Fixups (as well as ProcessPlans).

You will notice that the Preflight Profile window looks quite similar to the one of pdfToolbox.
The reason for this is quite simple: Adobe uses technology from callas software for the Preflight functionality in Acrobat Pro.



Why using pdfToolbox AND Acrobat Preflight?

There are several functions available in pdfToolbox (Plug-In and Standalone) only, which are not part of Acrobat Preflight:

- ProcessPlans
- Custom reports (based on HTML-Template)
- Quick Checks
- Quick Fixes

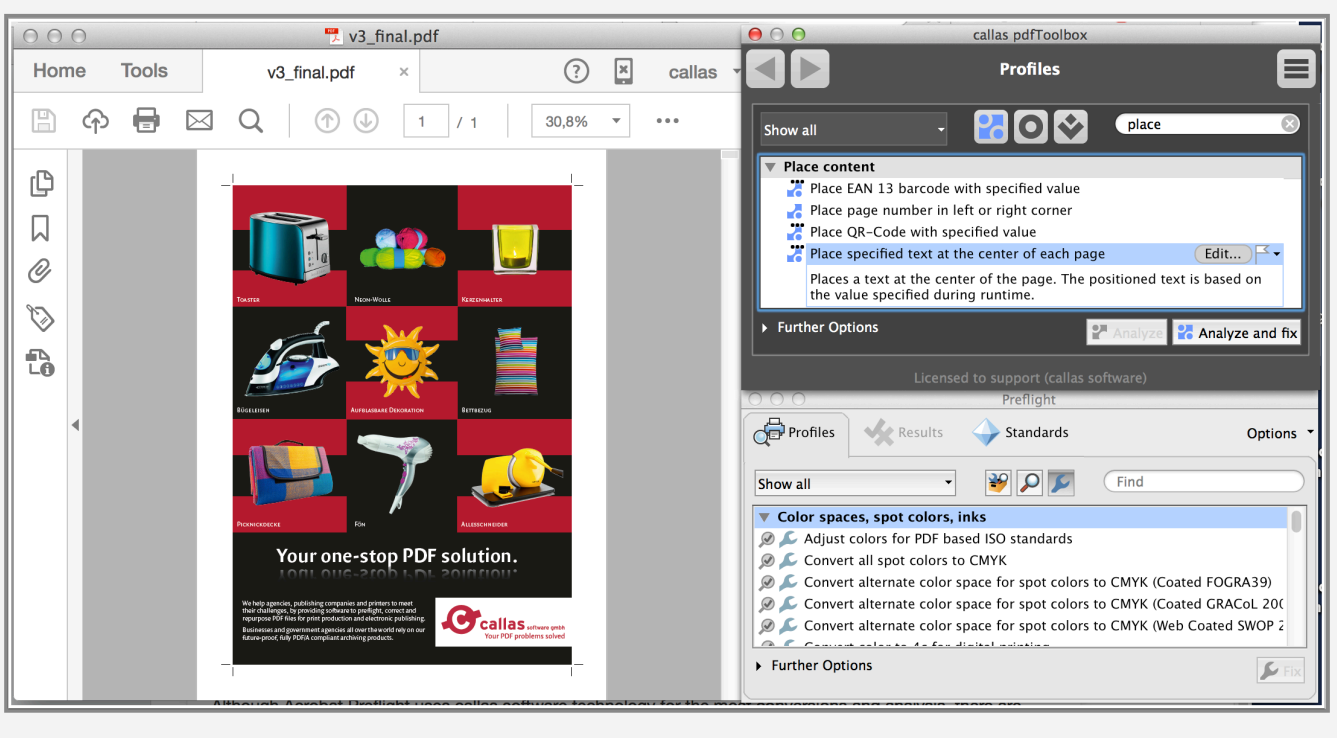
- Actions

- **Several Fixups like:**
- Place text
- Place barcode
- Place page number
- Place content (based on HTML-Template)
- Convert colors to n-channel

In general, pdfToolbox has a more frequent update interval than Acrobat Preflight.
Therefore new features, adjustments and fixes are earlier available in callas software products.

Running Preflight and pdfToolbox in parallel

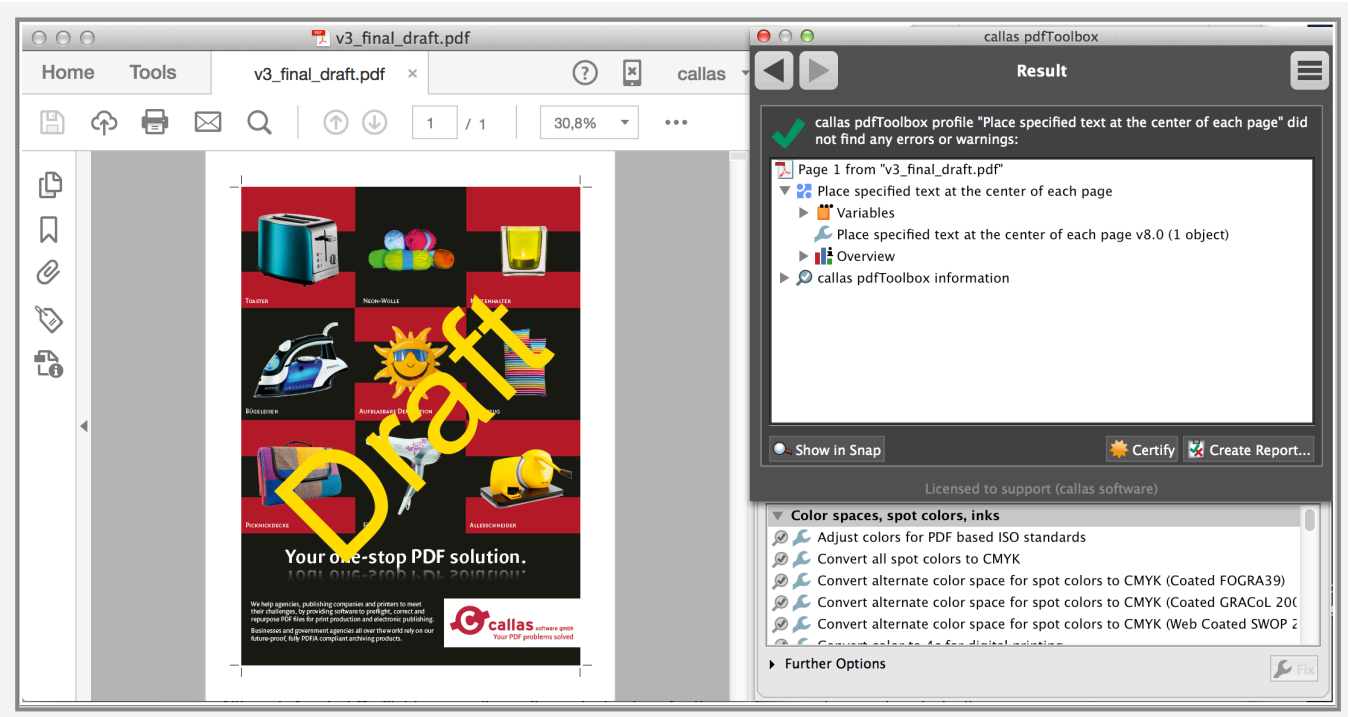
You can use Acrobat Preflight and callas pdfToolbox Plug-In in parallel.



Using pdfToolbox-only features

Using the pdfToolbox Plug-In allows to use pdfToolbox-only features like placing custom content using a Fixup.

For example placing a text, a barcode, numbering pages or a custom content (using a HTML template) as additional content into the PDF.



Upgrade Adobe Acrobat on Windows so that callas pdfToolbox is still working

When upgrading Adobe Acrobat on Windows (for example to go from version 9 to Acrobat X), Adobe Acrobat is completely reinstalled and the old version of the software is removed by the Adobe installer. To ensure pdfToolbox remains working, follow the following steps:

- Your preferences should not be affected by this upgrade. But it is always wise to have a backup of your preferences before doing big changes, so head on over to the callas preferences folder and make a backup.
- Make sure you have the pdfToolbox installer handy. If necessary you can download it from the callas download page.
- Run the Adobe Acrobat installer to upgrade your Acrobat installation.
- When Adobe Acrobat is installed and properly working, run the callas pdfToolbox installer to reinstall the pdfToolbox plugin in your new version of Adobe Acrobat.

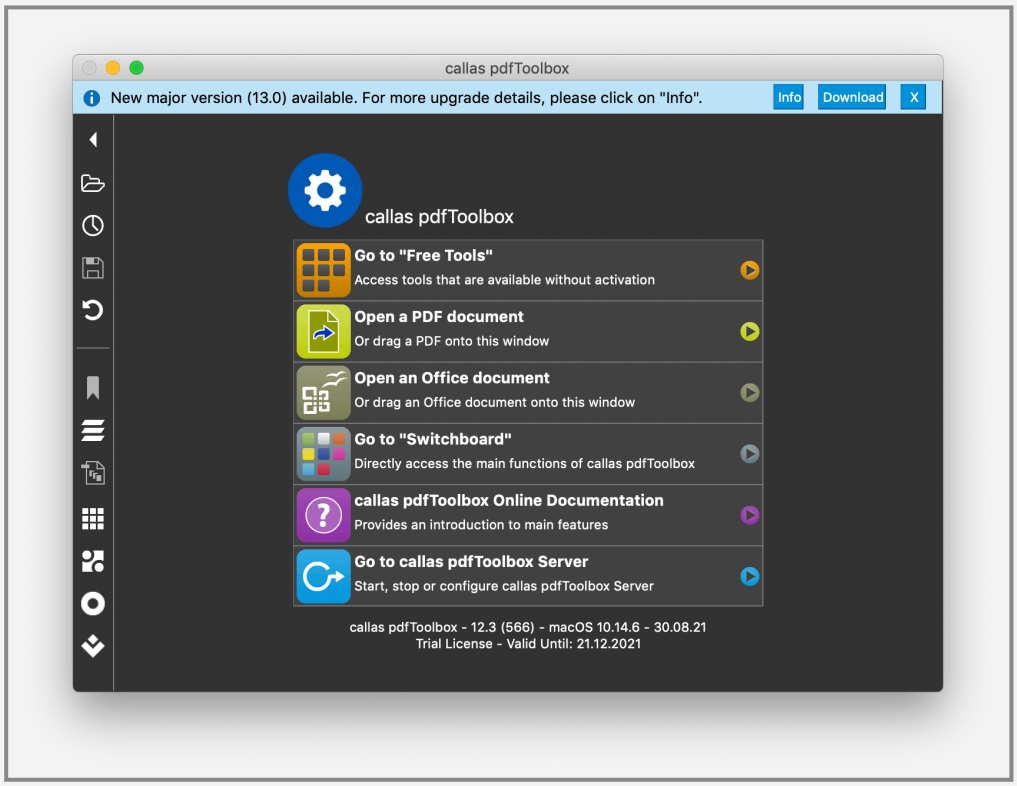
Remarks

- Your preferences or custom profiles will not be affected by running the Adobe or pdfToolbox installer.
- You should not need to reactivate pdfToolbox, it will continue to recognize your license.
- You can avoid having to run the pdfToolbox installer, but going into the plug-ins folder of Adobe Acrobat and making a copy of the callas pdfToolbox folder. Then, after upgrading to the new Adobe Acrobat version simply copy the pdfToolbox folder back into the plug-ins folder of Adobe Acrobat.

1.9 Notifications about available updates

**In pdfToolbox Standalone we inform you about available software updates.
How can you customize these notifications? We will show you in this tutorial.**

Update notification for major versions



As soon as a new major version is available for download, a notification bar in the main window of the Standalone version is shown. This functionality is not available in the Acrobat Pro PlugIn.

You can either download this update directly using the "Download" button or hide this notification using the "X". When you hide the notification you'll be asked if you want to be notified with the next start of the software or if you do not

want any information about this update anymore. These settings can also be set in the preferences (more information further down in this tutorial).

Additionally, you can switch to the callas software product page for more information about the update.

Additional information about the installation of new major versions

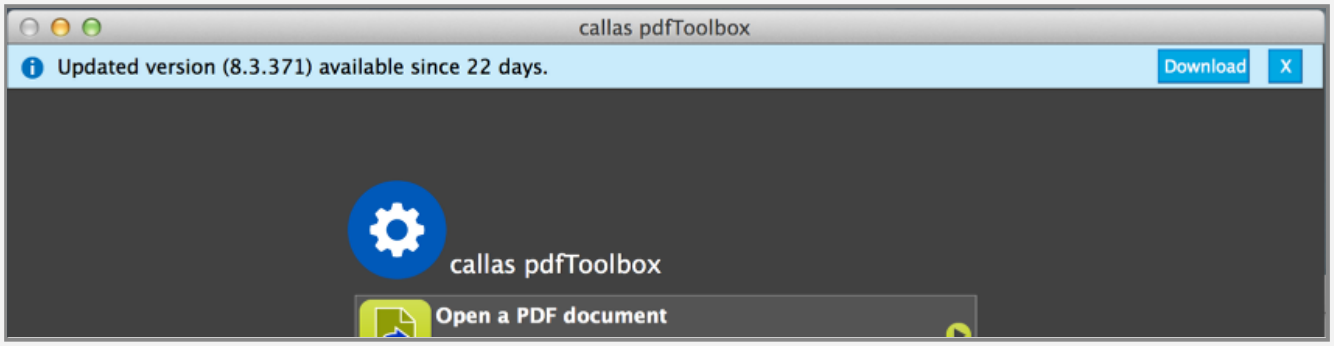
New major versions contain a bigger number of new functions as well as significant improvements regarding processing.
Major versions are installed in parallel to existing versions - so you can continue to use them.

A new license key is necessary for new major versions as well as a new activation.
Owners of a valid software maintenance agreement (SMA) will receive a new keycode from your reseller or integration partner.
If no SMA exists, you can buy an update on the callas software website or at a local reseller. More information is shown using the "Info" button.

After the first start of the software after the update you'll be asked if you want to import your settings (e.g. own Profiles, ProcessPlans, Checks and Fixups) into the new version.

Please note: Profiles or Libraries exported from new major versions cannot be imported in older software versions.
Profiles or Libraries from older versions can be imported into newer versions of course.
If you want to have the possibility to switch back to a previous version, we recommend a backup of your Profiles.
More information about this feature in "Export Libraries" further down.

Update notification for minor versions



As soon as a new minor version is available for download, a notification bar in the main window of the Standalone version is shown. This functionality is not available in the Acrobat Pro PlugIn.

Minor versions are software updates, which not only contain bugfixes and performance improvements but also new functions.
User of the respective major version of the Desktop software can update free of charge. User of the Server/CLI version needs to have a valid SMA.

You can either download the version using the "Download" button or hide the notication with the "X" button.
In case you hide the notification, you can choose if you want a reminder at the next software start or if you want to skip this version update. These settings can also be defined in the preferences of the software (more about this feature later in this tutorial).

Additional information about the installation of new minor versions

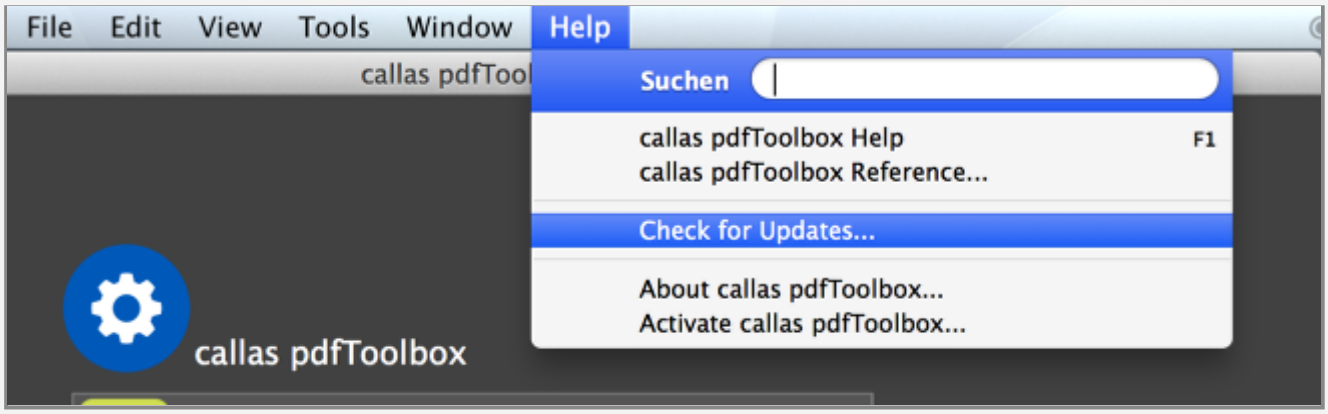
New minor versions can offer new Checks and Fixups as well as new settings for existing Checks and Fixups.
Sometimes also new, predefined Profiles, Checks and Fixups are contained.

Therefore it is not possible, in most cases, to switch back to a previous minor version (e.g. from version 8.3 back to 8.2). In-

tensive internal regression tests ensure that a customer should not have any problem with an update.

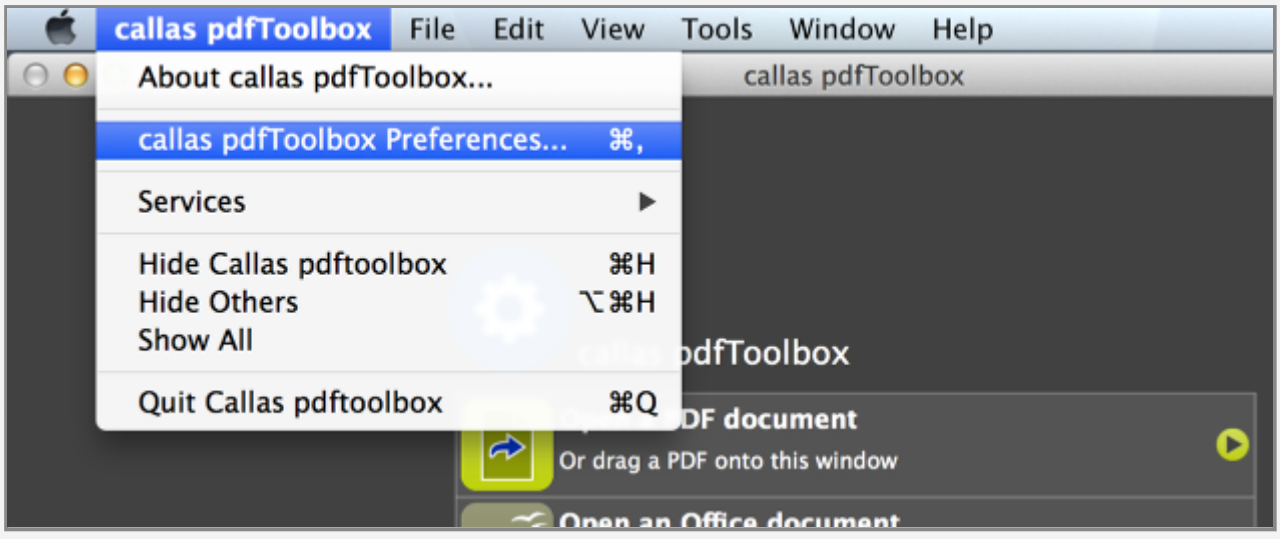
If you want to keep the possibility to switch back to a previous minor version, we recommend a backup of your Profiles. More information in the article part "Export Libraries".

Manual check for new updates



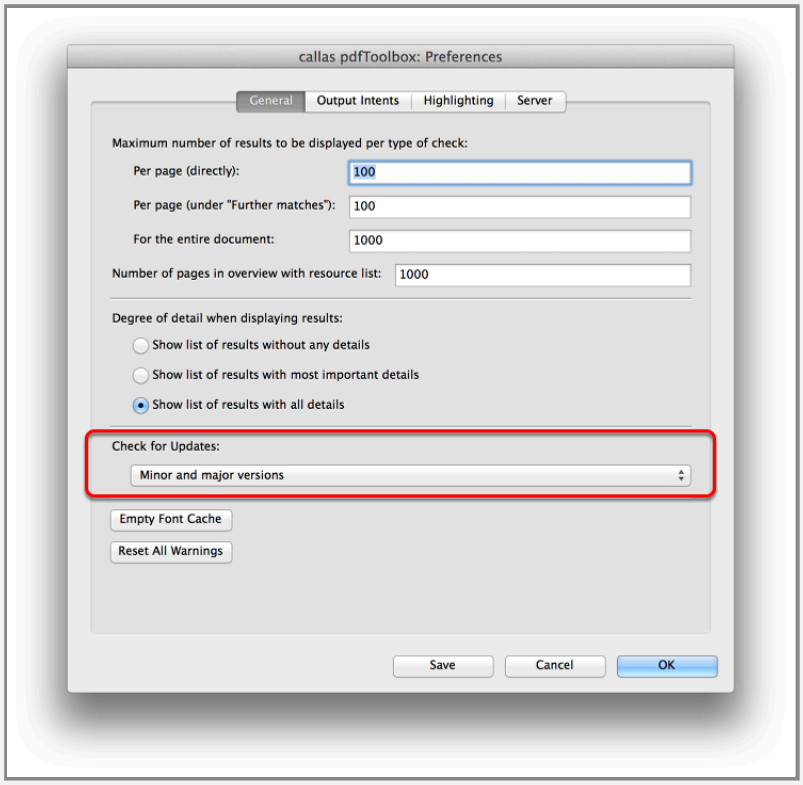
In the "Help" menu you'll find "Check for updates", which will check for the latest available versions.

Go to the update check in the preferences



Using the menu bar, you can open the software preferences.

Configure the update check in the preferences



In the "General" tab in the preferences, you can easily define if the software shall check for updates. And if yes: whether it should happen for major versions, minor versions or both.

1.10 Run as a service (Windows)

Start server, dispatcher or satellite as a service

The callas pdfToolbox Services application is only available for Windows at the moment.

For Mac or Linux a Deamon can be used:

- Create a Deamon using Mac
- Create a Deamon using Linux

Installation

1. Ensure there is an installation of callas pdfToolbox Server/CLI on the system and the application has been activated successfully.

2. A special executable, which is needed to run pdfToolbox as a service, is located in /cli/var/Service.

Copy the executable into the subfolder "/cli" of the application folder of the server installation.

3. To install, the following command has to be executed on the command line:

```
pdfToolboxService.exe --install
```

Click to copy
Please confirm the security question of Windows if shown.

4. Open the "Services dialog" of Windows. This dialog can easily opened by typing the following string into the search field of the Windows start menu or use the following command on the command line:

Services.msc

5. There should show up 3 new services:

- callas pdfToolbox Server
- callas pdfToolbox Satellite
- callas pdfToolbox Dispatcher

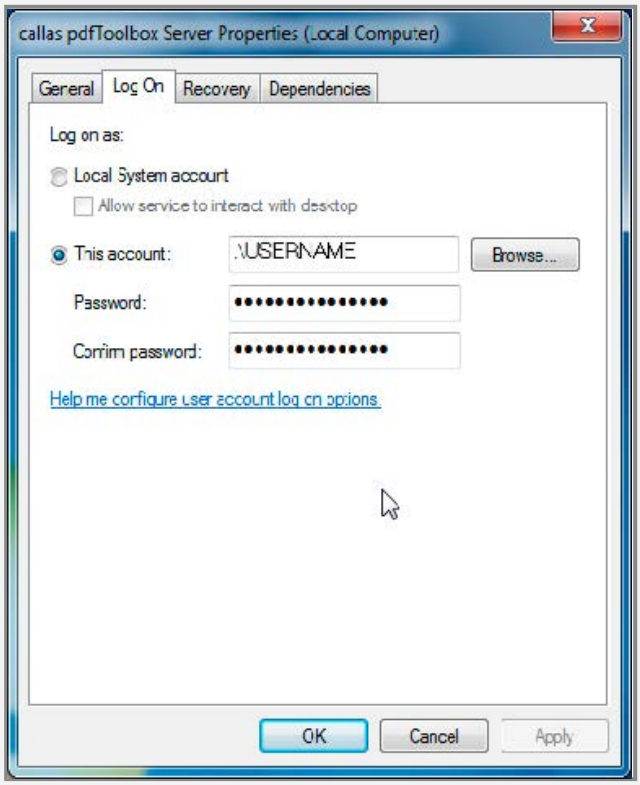


6. Select "pdfToolbox Server" and use the right-click menu item "Action" [or "Properties" depending on the Windows-version] in order to use this service and set in "General" the "Startup Type" to "Automatic" or "Manual".

When "Automatic" is choosen, every started job will continue processing, even when no user is logged on the system. It will even start processing, when the operating system is started.

When using "Automatic", also user details have to be entered into the "Log On" tab.

It must be ensured, that all folders used in the jobsettings can be accessed by the defined user (especially when network paths shall be used by the job).



Configuration of a job

Now a job can be configured using the ServerUI, which can be accessed using the Standalone version (Menu: Tools - Server).

When a job is started, pdfToolbox Standalone can be closed. The services apllication ensures, that the processing will continue even when the user is logging off.

If a job is using network paths for the used folders, it is recommended to use UNC paths (e.g. "\\192.168.1.22\hotfolder\...") as an assigned drive letter (e.g. "H:\hotfolder\...") is user-specific.

Access by remote

It is possible to connect to a Server running as a service by remote via the local network.

Start pdfToolbox standalone, select menu: "Tools" - "Server" and choose

"Connect with remote server".

Enter the IP of the remote server where the service is running.

After connecting all jobs on the remote server are shown and can be started, stopped or even modified. (Hotfolder paths of any server jobs (IN, OUT, etc.) have to be configured so that they are valid from the service's perspective (the system where the service is running) - and not from the perspective of the controlling standalone application).

Uninstall or change to a new major version

If the Service is no longer needed and the entry shall be removed or when the Service entry has to be updated because of a new major version (e.g. when upgrading from version 9 to version 10), it is recommended to uninstall the service using the following command:

```
pdfToolboxService.exe --uninstall
```

Click to copy
The pdfToolboxService.exe has to be in the same directory like the pdfToolbox.exe (like when installing the Service).

Troubleshooting

If network paths are used for processing jobs, the user should have sufficient rights to access them.

There may be special requirements for converting Office files to PDF when using pdfToolbox as a service. Check http://www.callassoftware.com/goto/tbx_ENU_topdf for the latest details.

Note: In general it is recommended to grant the respective service user administrator privileges. If this level of rights can not set due to internal regulations, some additional settings within the operating system are recommended. Additional settings with limited user rights

Additional settings for Office conversion with restricted user permissions

If Office conversion is needed out of services using Windows it is recommended to grant the respective service user administrator privileges. If this level of rights can not set due to internal regulations, some additional settings within the operating system are recommended.

The following folders should allow the user the respective access right:

For 64-bit machines	
C:\Windows\Temp	Modify
C:\Windows\syswow64\config	Read
C:\Windows\syswow64\config\systemprofile	Read
C:\Windows\syswow64\config\systemprofile\AppData	Modify
C:\Windows\syswow64\config\systemprofile\Desktop	Modify (Create it, if it does not exist)
C:\Windows\syswow64\config\systemprofile\AppData\Local\Mi-	Modify (Create it, if it does

For 64-bit machines	
crosoft\Windows\INetCache	not exist)

For 32-bit machines	
C:\Windows\Temp	Modify
C:\Windows\system32\config	Read
C:\Windows\system32\config\systemprofile	Read
C:\Windows\system32\config\systemprofile\AppData	Modify
C:\Windows\system32\config\systemprofile\Desktop	Modify (Create it, if it not exist)
C:\Windows\system32\config\systemprofile\AppData\Local\Microsoft\Windows\INetCache	Modify (Create it, if it not exist)

Additional settings:

- Set the 32-bit folder preferences (for details see above) in addition to the 64-bit preferences on 64-bit systems running 64-bit versions of Microsoft Office
- Set the default printer to XPS Document Writer

DCOM settings:

- Launch DCOMCNFG by using:

```
C:\WINDOWS\SysWOW64> mmc comexp.msc /32
```

- Go to Computers MyComputer DCOM Config.
- Right-click the application that you want to automate.
  The application names are listed in the table below:

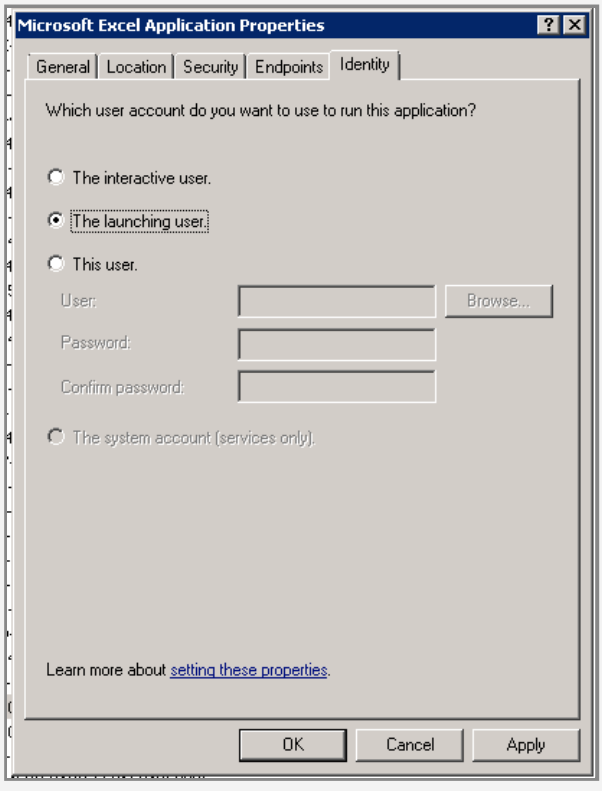
Application	DCOM Name
Microsoft Access 2007/2010/2013/2016	Microsoft Access Application
Microsoft Excel 2007/2010/2013/2016	Microsoft Excel Application
Microsoft Office Word 2007	Microsoft Office Word 97 - 2003 Document
Microsoft Word 2010/2013/2016	Microsoft Word 97 - 2003 Document

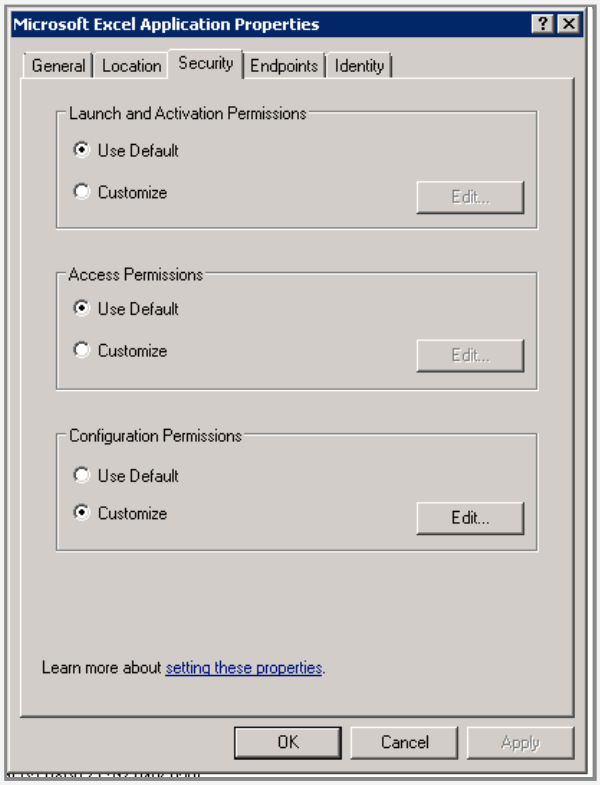
- On some systems Microsoft Word is not displayed and you will have to use `{00020906-0000-0000-C000-000000000046}` instead.
- Click Properties to open the property dialog box for this application.
- Verify Identity and Security tabs

Note

For Microsoft Excel, Microsoft Word and Microsoft PowerPoint the DCOM Identity "The launching user" must be selected.

For Microsoft Visio the DCOM Identity "The interactive user" needs to be selected.




Microsoft Excel Application Properties
General
Location
Security
Endpoints
Identity
Launch and Activation Permissions
Use Default
Customize
Edit...
Access Permissions
Use Default
Customize
Edit...
Configuration Permissions
Use Default
Customize
Edit...
Learn more about setting these properties.
OK
Cancel
Apply

1.11 Create a Daemon using Mac



com.callassoftware.pdftoolboxserver.agent.plist

Other than a Service using Windows, a so-called Daemon using Mac can either be started when a User is logged in or (alternatively) when the system is started:

Copy the attached plist file to the following location:

/Library/LaunchAgents

(for run at Login)

or to

/Library/LaunchDaemons

(for run at Boot)

The agent will be activated with the follwing call:

Terminal:

```
launchctl load /Library/LaunchAgents/
com.callassoftware.pdftoolboxserv-
er.agent.plist
```

Using the following call, the agent can be deactivated:

Terminal:

```
launchctl unload /Library/LaunchAgents/
com.callassoftware.pdftoolboxserv-
er.agent.plist
```

If a Server-Job is active, this Job will start as soon as the sytem is started and files in the respective hot folders will be processed.

1.12 Run as an autostart systemd service (Linux)

The ReadMe as well as the template for running pdfToolbox as an autostart systemd service under Linux can also be found in the installation package in the following folder:

etc/systemd

systemd callas service usage/installation example

Note: Requires root permissions.

Note: The service description must *not* contain a privateTemp attribute.

Note: The whole <install_dir> path including all higher level directories need to have at least read-and-execute permissions.

Recommendation: Unpack the installer below a non-userspecific directory (such as e.g. /opt/callas).

Note: A `--cachefolder` option *must* be used, *even* if the user executing the service actually has a home directory. That is because per default a subdirectory of '/usr/share/callas software' is used as the storage folder and creating the needed directory structure would require permissions to run with the root user account.

Note: The <install_dir> and the <cache folder> must be owned by the user executing the service.

Recommendation: Choose a top level folder to contain the <install_dir> and the <cache folder>. Then change ownership for the whole top level folder, e.g. for pdfToolbox running in dispatcher mode:

```
sudo mkdir -p /opt/callas/dispatcher

sudo chown -R nobody:daemon /opt/callas/dispatcher
```

Click to copy

Recommendation: to make future updates easier it is recommended to have a download folder, e.g. /opt/callas/dispatcher/download. A symlink would then point to the concrete pdfToolbox version to be used:

```
sudo -u nobody bash
cd /opt/callas/dispatcher
mkdir downloads
cd downloads
wget https://www.callassoftware.com/extranet/callas_pdfToolboxCLIandServer/
callas_pdfToolboxCLI_x64_Linux_14-3-616.tar.gz
tar xvpf callas_pdfToolboxCLI_x64_Linux_14-3-616.tar.gz
cd ..
ln -s downloads/callas_pdfToolboxCLI_x64_Linux_14-3-616 callas_pdfToolbox_CLI
```

Click to copy

Preparation

1. Use one of the prepared service templates such as ...
   - callas-dispatcher.service
   - callas-hotfolder.service
   - callas-satellite.service
2. Adjust the service description to fit your needs (e.g. change real installation directory and/or specifiy a `--licenseserver` option)
3. Setup a cache folder (Note: The cache folder must be owned/writable by the executing user, e.g. by running 'sudo -R chown nobody:daemon <given_cache_folder>')
4. Activate the license. This needs to be done with the user account that is executing the service and the specified `--cachefolder`

```
sudo -u nobody bash
cd downloads/callas_pdfToolboxCLI_x64_Linux_14-3-616
./pdfToolbox        --keycode myname mycompany mylicense.pdf        --cachefold-
er=/opt/callas/dispatcher/cache
./pdfToolbox        --activate myactivation.pdf        --cachefolder=/opt/callas/
dispatcher/cache
```

Click to copy

Installation

Copy the service description to the systemd service directory (as an example we are going to use the callas-dispatcher.service)

Note: /lib/systemd/system/callas-dispatcher.service must be a regular file and *not* a symbolic link.

Note: A 'sudo systemctl daemon-reload' is needed whenever /lib/systemd/system/callas-dispatcher.service is changed.

```
sudo cp callas-dispatcher.service to /lib/systemd/system
```

Click to copy

Check if the service is working as expected:

```
sudo systemctl daemon-reload
sudo systemctl start callas-dispatcher
systemctl status callas-dispatcher
```

Click to copy

Controlling the service

Check the service status:

```
systemctl status callas-dispatcher
```

Click to copy

Stop the service:

```
sudo systemctl stop callas-dispatcher
systemctl status callas-dispatcher
```

Click to copy

Start the service:

```
sudo systemctl start callas-dispatcher
systemctl status callas-dispatcher
```

Click to copy

Restart the service:

```
sudo systemctl restart callas-dispatcher
systemctl status callas-dispatcher
```

Click to copy

Set callas dispatcher service to auto start on reboot:

```
sudo systemctl enable callas-dispatcher
```

Click to copy

To disable the callas dispatcher service on next reboot:

```
sudo systemctl disable callas-dispatcher
```

Click to copy

1.13 System paths for preferences

Whether it be for comparison related information logged in a 'Compare.log' file or allowing other users to work on a machine where the callas license is hardware-bound, a user needs to find the system paths. The information in this article helps with finding such paths.

Below is the general path for the Preferences folder:

Win: %APPDATA%\callas software\callas pdfToolbox CLI 13

(can easily be reached by entering "%allusersprofile%" into the address bar of the Windows Explorer)
Mac: ~/Library/Preferences/callas software/callas pdfToolbox CLI 13
Linux: $HOME/.callas software/callas pdfToolbox CLI 13

Below you will find usage-based system paths:

Multi-user system (Activating additional users on the same system)

The folder path from which the **License.txt** file can be retrieved and to which it should be copied is as follows:

MacOS: */Users/<USERNAME>/Library/Preferences/callas software/callas pdfToolbox <VERSION>*

Windows: *C:\Users\<USERNAME>\AppData\Roaming\callas software\callas pdfToolbox <VERSION>*

Log comparison based information

Server/CLI

MacOS: */Users/<USERNAME>/Library/Preferences/callas software/callas pdfToolbox CLI <VERSION>/Logging/Compare.log*

Windows: *C:\Users\<USERNAME>\AppData\Roaming\callas software\callas pdfToolbox CLI <VERSION>\Logging\Compare.log*

Linux*: /home/<USERNAME>/.callas software/callas pdfToolbox CLI <VERSION>/Logging/Compare.log*

Desktop

MacOS: */Users/<USERNAME>/Library/Preferences/callas software/callas pdfToolbox <VERSION>/Logging/Compare.log*

Windows: *C:\Users\<USERNAME>\AppData\Roaming\callas software\callas pdfToolbox <VERSION>\Logging\Compare.log*

License Server: Support for 'configuration file' with defined IPs

MacOS: */Users/<USERNAME>/Library/Preferences/callas software/callas pdfToolbox CLI <VERSION>/LicenseServer.txt*

Windows: *C:\Users\<USERNAME>\AppData\Roaming\callas software\callas pdfToolbox CLI <VERSION>\LicenseServer.txt*

Linux: */home/<USERNAME>/.callas software/callas pdfToolbox CLI <VERSION>/LicenseServer.txt*

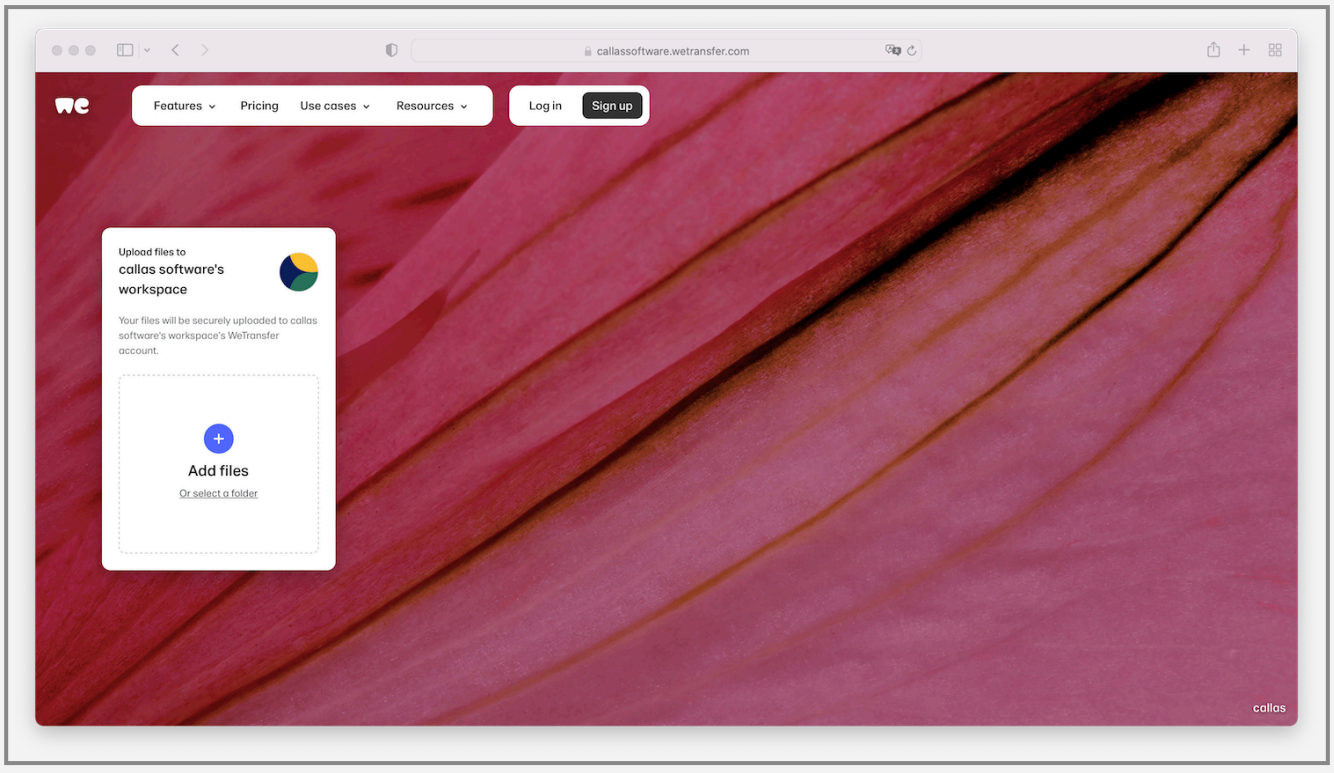
1.14 How to upload large files to support

When reporting problems to support or asking questions about a specific functionality, it is always helpful to provide as much information as possible:

- Profiles or Process Plans when working with pdfToolbox or pdfaPilot
- Example documents 'before' and 'after' to explain the problem or question
- Screenshots explaining the specific problem

If such files are larger than 2MB, it is more reliable and faster to send them to us via WeTransfer.

callassoftware.wetransfer.com



1. Go to callassoftware.wetransfer.com
2. Click on 'Add files' and select the files that you

want to upload (can be one or more).

3. Enter your email address.
4. If there is already a support ticket, please enter the relevant ticket number in the 'Message' field so that the files can be easily merged into the appropriate support case.
5. Your files will be uploaded to the callas software workspace and sent to our support team.

You will receive an email confirming your files were uploaded succesfully.

2. callas pdfToolbox Basics

2.1 The Switchboard: easy-to-use Actions

The Switchboard is the first window you should explore in pdfToolbox Desktop. pdfToolbox offers a number of predefined Actions. Many of these Actions are quickly accessible from the Switchboard. They facilitate the use of frequently used processes such as imposition or color conversions. Because they are easy to use, they are limited in the way they can be configured.

Actions started as items on the Switchboard, but over the years they have migrated to different parts of pdfToolbox. For more information on where to find actions in pdfToolbox and an overview of all available actions, please refer to the chapter Actions and their use in callas' products.

If you want to perform more complex and customizable tasks with pdfToolbox, you should use a Fixup, Profile or Process Plan. See the next article for more information.

The Switchboard window

To open the Switchboard if it is not visible, use the "Tools" > "Switchboard" menu item or click on the Switchboard icon which is located on the left side panel of the main window.





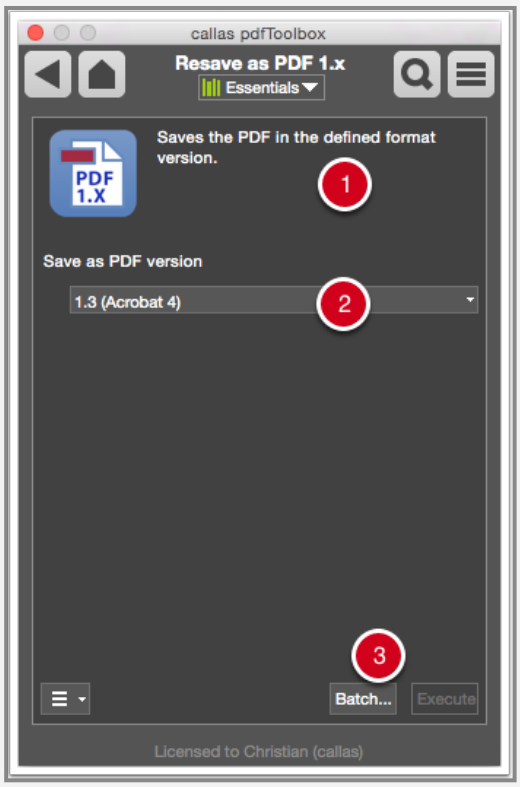
The following section briefly describes how to get to a particular action.

1. The left screenshot shows the Switchboard main window with the available functionality groups.
2. Clicking on a specific group icon will display all actions related to that group.
3. Depending on the Action you have selected, options may appear in the Switchboard window; use them to configure exactly how you want the action to be performed.
4. The "Execute" button can be used to perform the Switchboard action on the currently displayed PDF document.
5. The "Batch" mode allows you to process multiple PDF documents in a folder.

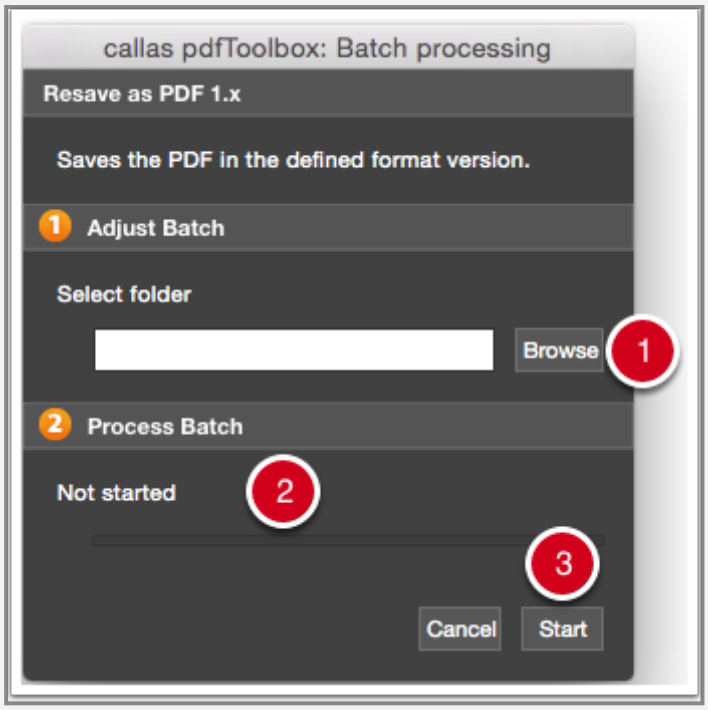
Batch Processing

pdfToolbox Desktop can do more than just checking or fixing one file at a time; the Batch Processing mode allows it to process multiple PDF documents within a folder. The Switchboard allows you to apply any action to all PDF files in a given folder.

The Batch function in pdfToolbox can process **up to 100 documents at a time**. For a high volume approach using hot folders, pdfToolbox Server is recommended.

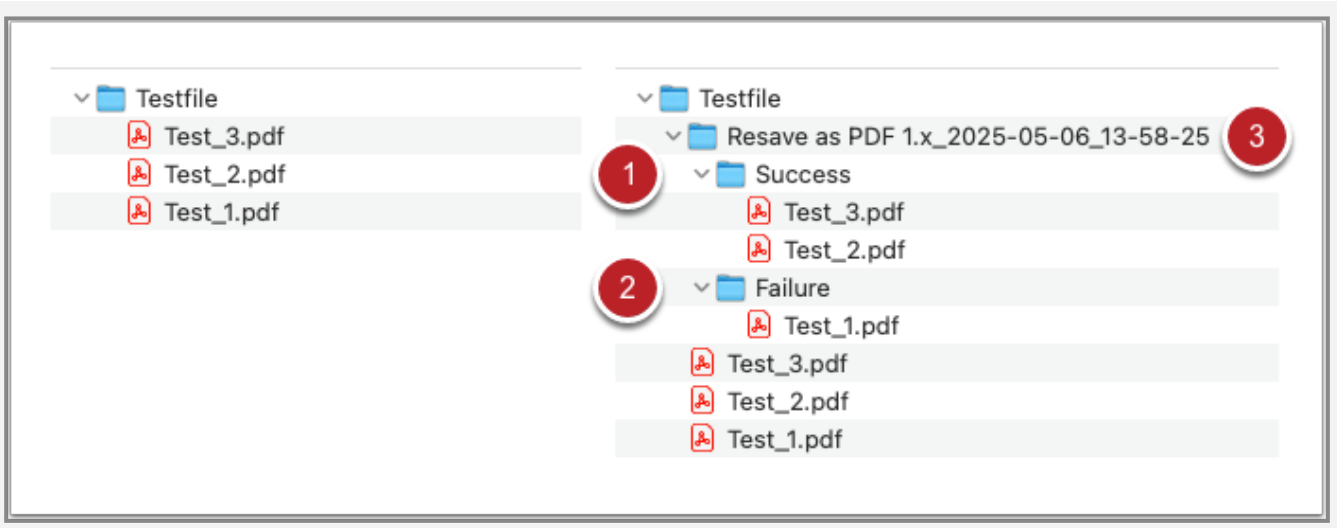


1. Select the desired action
2. Apply the desired settings.
3. Click on the **Batch** option to specify additional batch processing settings.



1. Select the folder containing the PDF files you want to process.
2. Under "2 Process Batch", the processing progress will be displayed.
3. Click on the "Start" button to start the batch processing.

Result after batch processing



Depending on the processing results, the files are then saved either to a folder with successful (Success - 1) or unsuccessful (Failure - 2) results.

These are located in a folder named with the *timestamp* - 3 for the processing point which will be automatically shown after processing is complete.

2.2 Profiles, Checks, Fixups, Process Plans and Libraries

callas pdfToolbox uses Profiles, Checks, Fixups, Process Plans and Libraries. These can be briefly summarized as follows:

- **Checks can be used to search PDF files for specific factors, and Fixups allow you to alter files in line with specific criteria.**
- **Profiles are designed to link Checks and Fixups together.**
- **Process Plans can be used to ensure that multiple Profiles, Checks or Fixups run one after another in a controlled fashion.**
- **Finally, Libraries enable users to bring all pre-specified categories together into personalized collections for a better overview.**

2.3 Checks and Fixups

With callas pdfToolbox you can use Fixups and Checks. **Checks** can be used to search PDF files for specific factors, and **Fixups** allow you to alter files in line with specific criteria. That means that Checks inspect PDF documents, but do not alter them. Fixups allow you to correct or modify your PDF file according to certain criteria. For example, if you want to flatten transparencies, do a color conversion or place content on the PDF file, you can do this with the corresponding Fix-up.

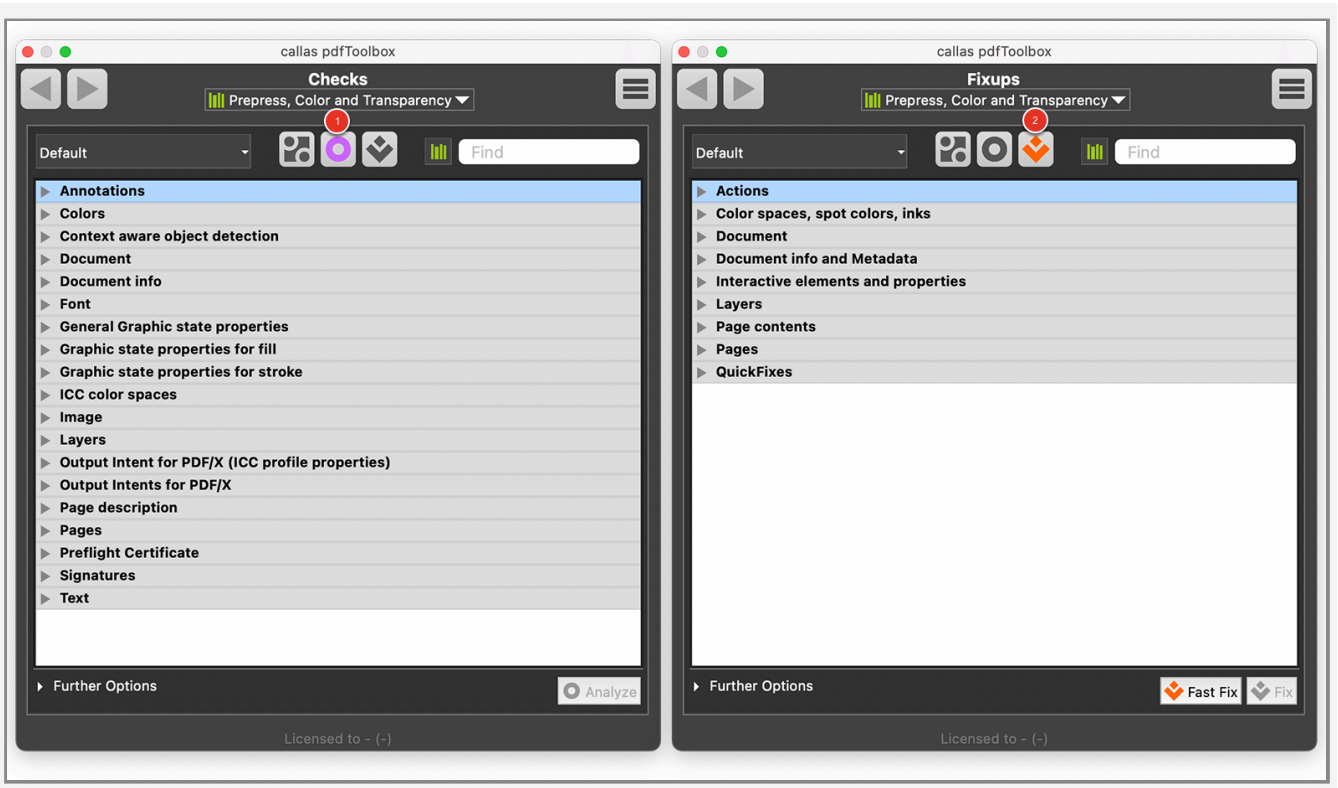
Checks and Fixups are set up in a similar way in pdfToolbox. There are three different ways to apply Checks or Fixups in pdfToolbox:

1. A predefined Check/Fixup corresponds exactly to the requirements and only needs to be executed.
2. A predefined Check/Fixup only requires a few parameters to be adjusted.
3. A Check/Fixup is created from scratch and all parameters are set individually.

Where can I find Checks and Fixups?

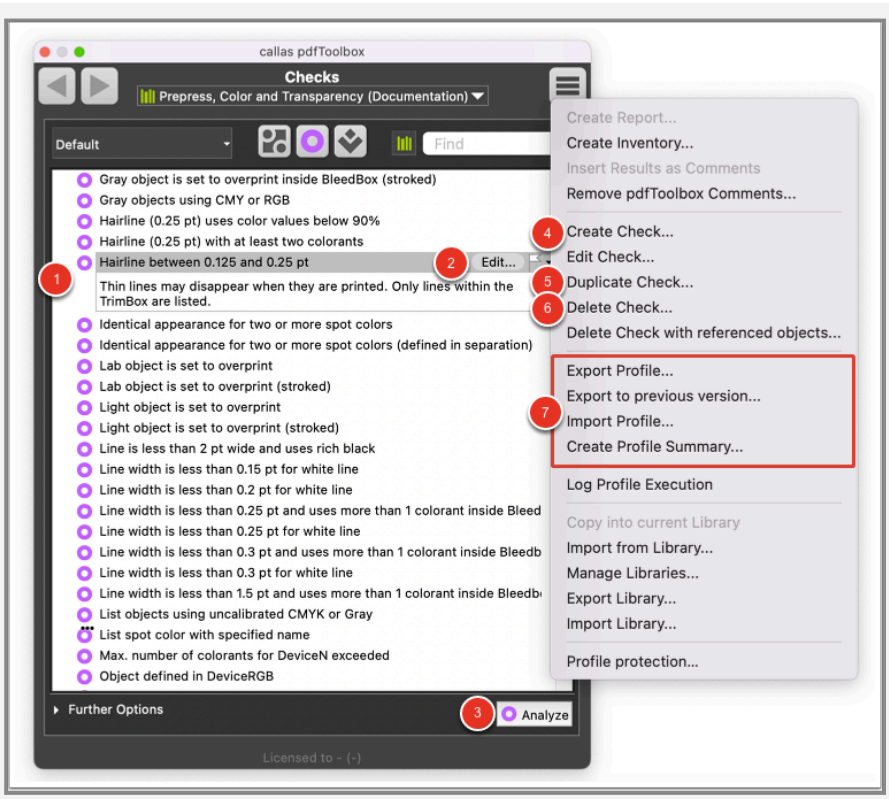
To access Checks and Fixups you can use the sidebar on the left side of the main window. Otherwise, the "Tools" menu can be used. Click either on Checks or Fixups. This will open the Profile window. The highlighted icon on the top shows you in which tab you are:

1. The highlighted icon indicates that we are in the **"Checks"** section of the profile window.
2. The highlighted icon indicates that we are in the **"Fixups"** section of the profile window.



The Profile Window with predefined Checks

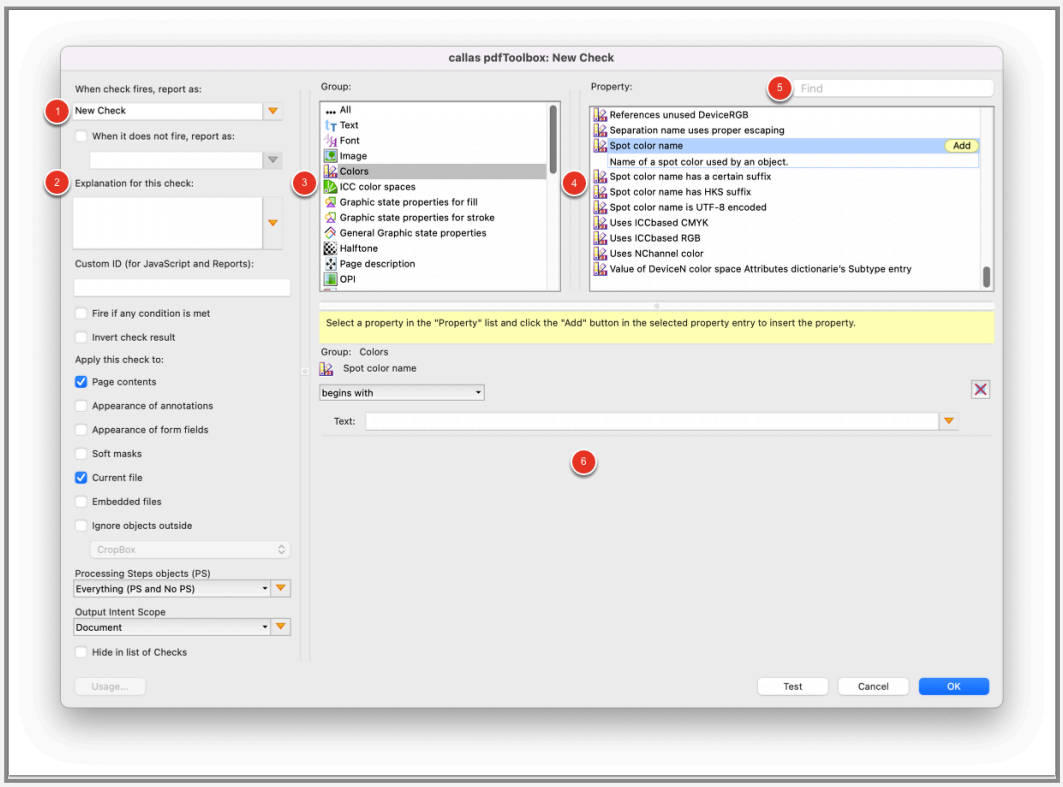
The Profile window contains all predefined Checks. Let's take a closer look at the Checks section:



1. A number of different Checks are supplied as standard. Clicking on an entry in the list will provide a short description of the Check.
2. After clicking on this button the "Edit Check" Window will be opened. In this window you can adjust the predefined Check.
3. This button will execute the selected Check on the currently displayed PDF document.
4. Checks can be created from scratch (see the section "Create a new Check" below).
5. It is possible to duplicate an existing Check and alter specific elements as necessary (for example changing the resolution or color space). The next article explains how this works.
6. Users can also delete Checks.
7. Checks can also be exported and imported. This may be of interest if you need to forward or distribute them within your organization, or if you work with external partners.

Create a new Check

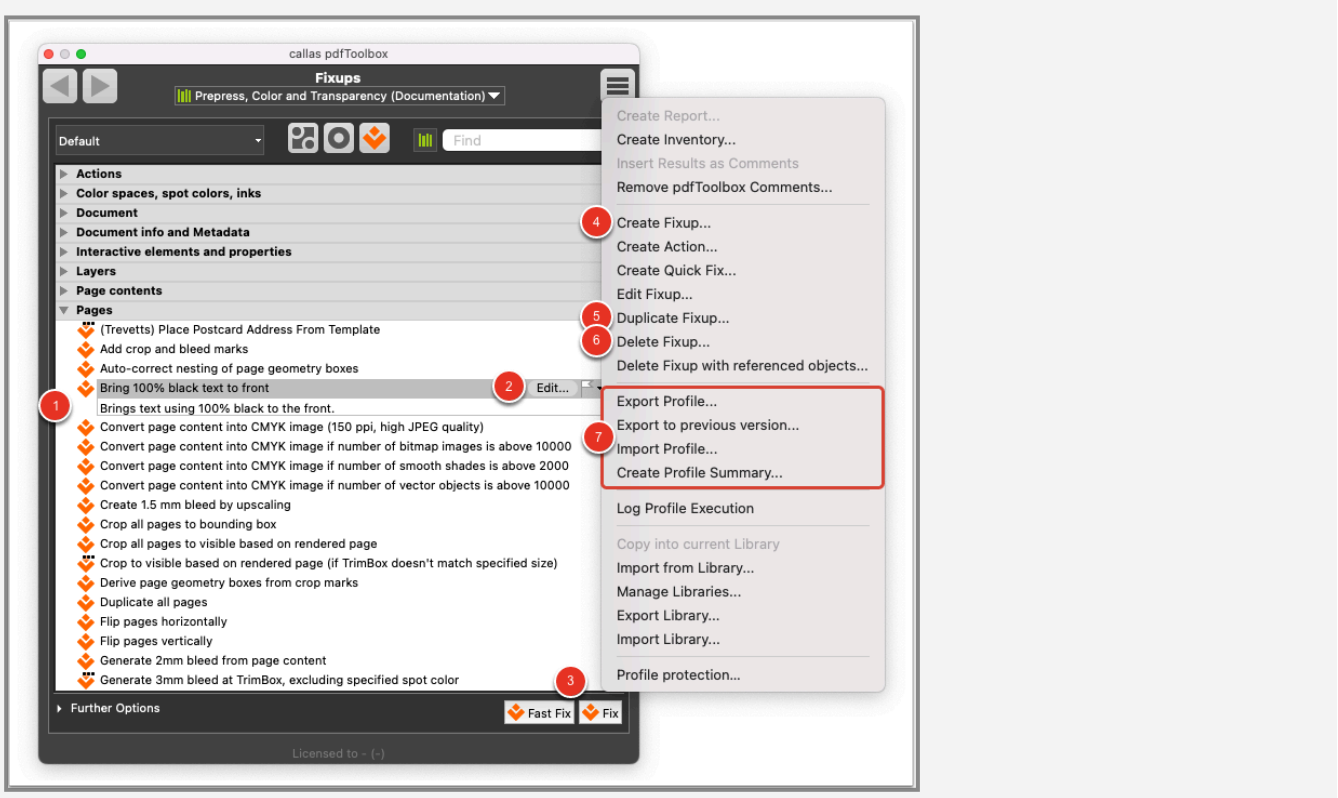
Watch this short video to learn how to create your first preflight Check in pdfToolbox.



1. A name for the Check can be defined here.
2. A short explanation for the Check can be defined.
3. All Check properties are divided into several groups.
4. List of all Check properties of the selected group.
5. In the search filed you can look for a specific Check property.
6. After clicking the "Add" button of a Check property, this area displays all the possible options and parameters for the selected Check property.

The Profile Window with predefined Fixups

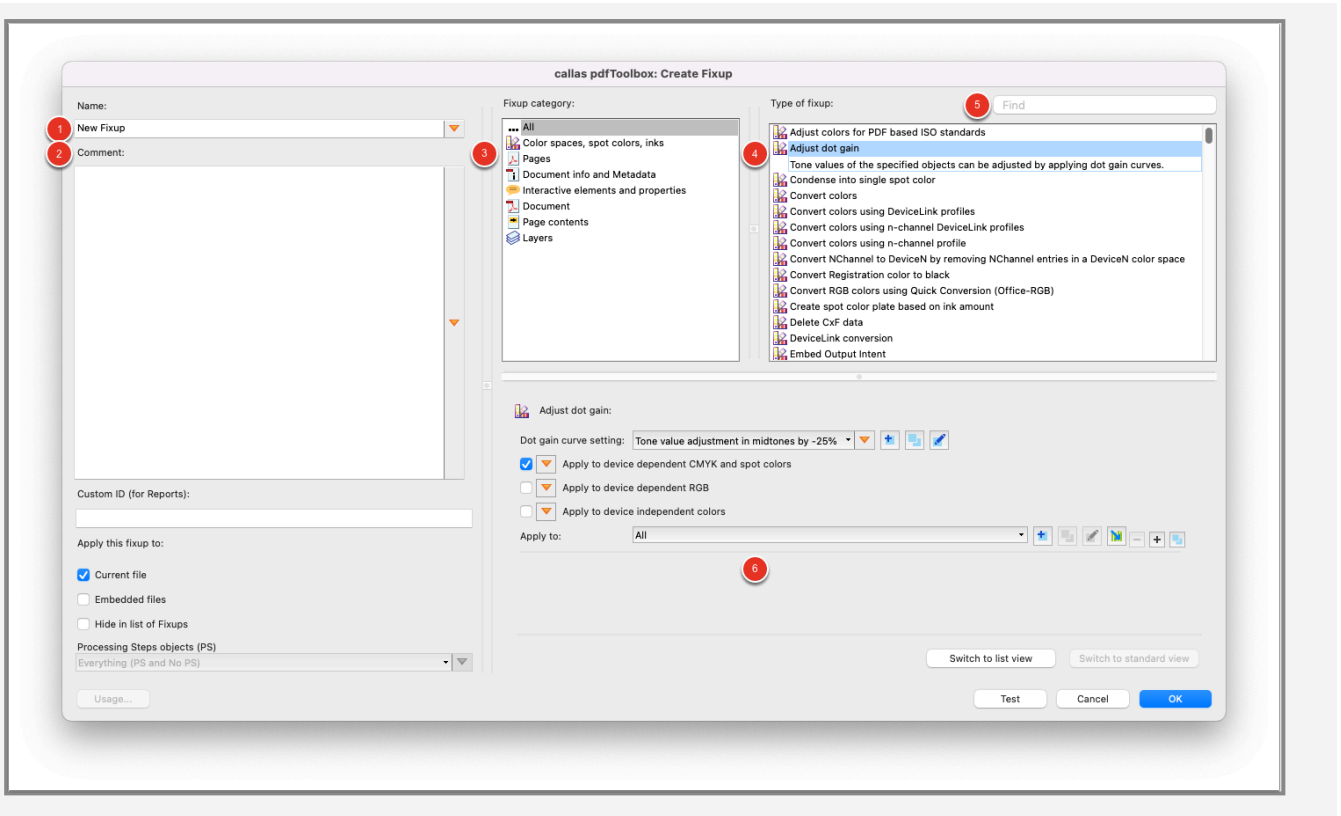
The Profile window contains all predefined Fixups. Let's take a closer look at the Fixup section:



1. A number of different Fixups are supplied as standard. Clicking on an entry in the list will provide a short description of the Fixup.
2. After clicking on this button the "Edit Fixup" Window will be opened. In this window you can adjust the predefined Fixup.
3. These buttons will execute the selected Fixup on the currently displayed PDF document.
4. Fixups can be created from scratch (see the section "Create a new Fixup" below).
5. It is possible to duplicate an existing Fixup and alter specific elements as necessary (for example changing the resolution or color space). The next article explains how this works.
6. Users can also delete Fixups.

7. Fixups can also be exported and imported. This may be of interest if you need to forward or distribute them within your organization, or if you work with external partners.

Create a new Fixup



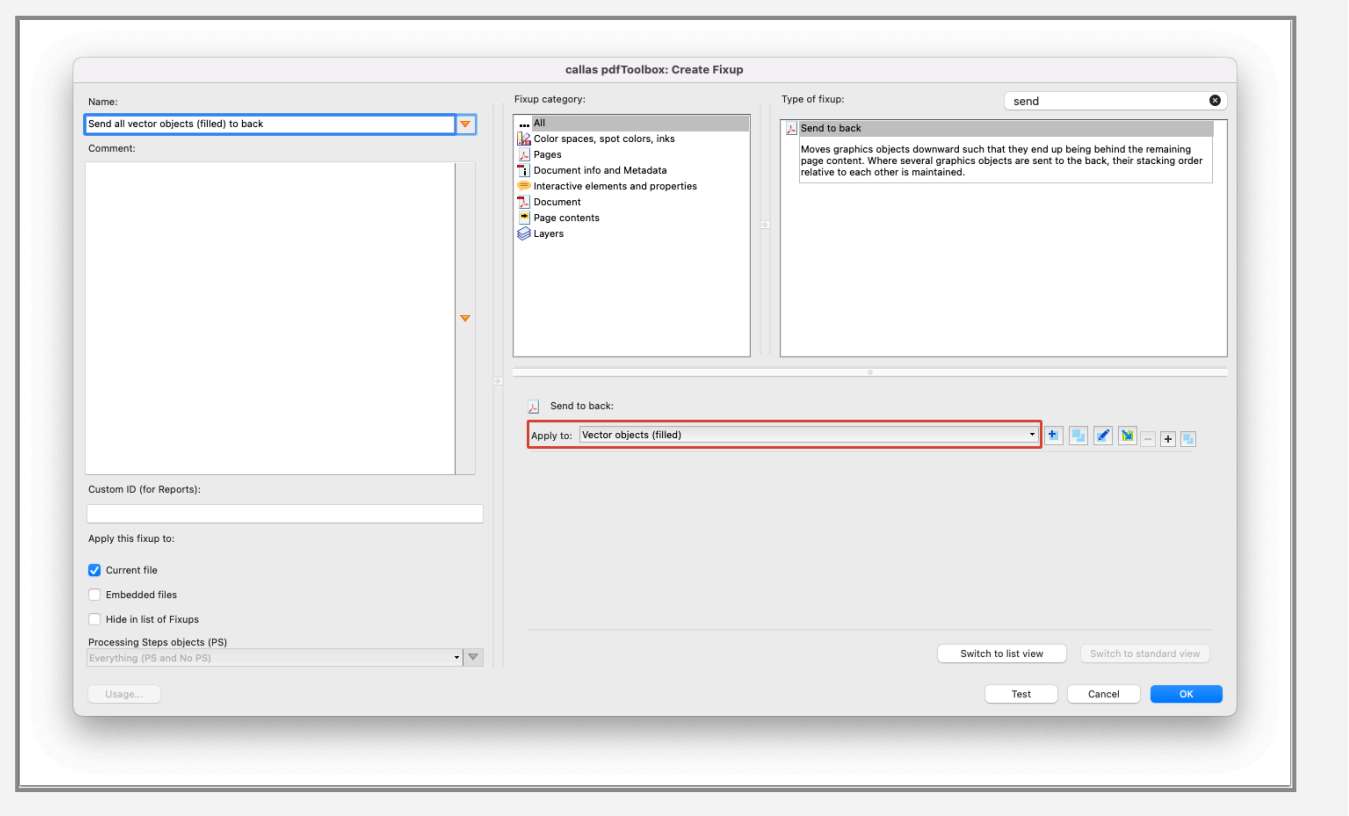
1. A name for the Fixup can be defined here.
2. A short explanation for the Fixup can be defined.
3. All types of Fixups are grouped into different categories.
4. List of all Fixup Types of the selected category.
5. In the search filed you can look for a specific Fixup type.
6. After selecting a Fixup, this area displays all the possible options and parameters.

Use Checks as filter in a Fixup

Sometimes you want to apply Fixups to the entire document. But there are also cases, where you only want to change specific objects in the PDF file. Therefore, it is sometimes necessary to apply a filter to the selected Fixup. This can be done by using a Check that can be stored in a Fixup so that it is only applied to certain objects.

Some Fixups already have parameters that limits the execution to certain objects. Other Fixups don't have this option. In those cases you might still want to limit the Fixup to specific objects or parameters. For those Fixups you can finde an "apply to" parameter.

In the "apply to" option you will have a list of all Checks that are part of the current library. When selecting a Check, you limit the execution of the Fixup to the objects that hits the Check. The following Image shows an example of the "apply to" Parameter.



The Fixup in the screenshot sends all filled vector objects to the background. "Vector objects (filled)" is a configured Check that identifies filled vector objects.

Watch this short video to get an overview of the terminology used around Checks and Fixups in pdfToolbox so that it becomes easier to understand the documentation and find your way around the software.

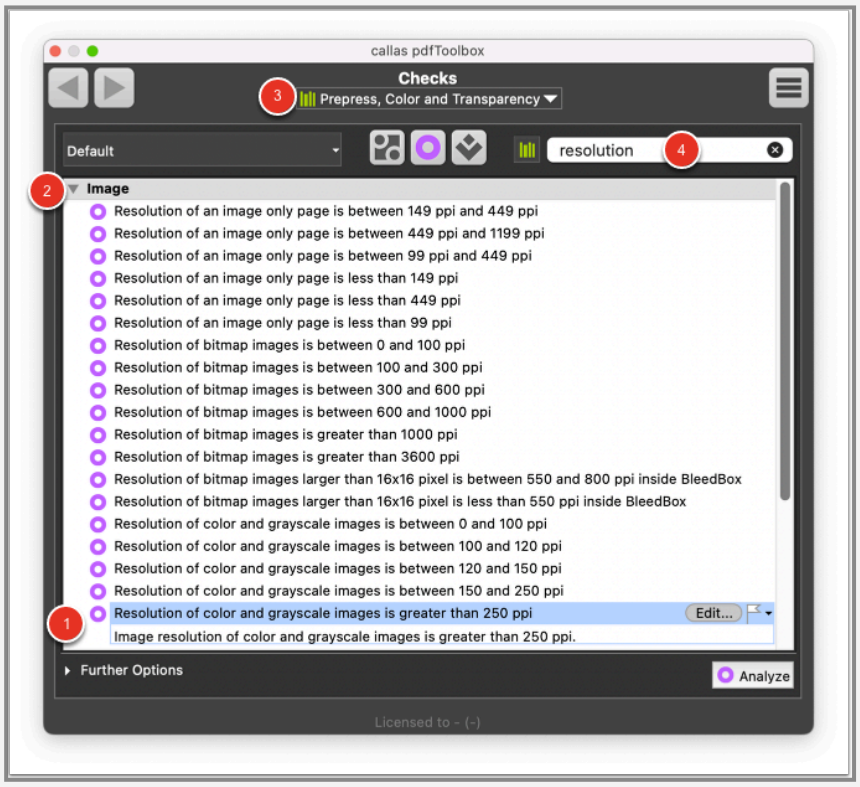
2.4 Duplicate and edit Checks and Fixups

Creating and setting up a duplicate of a Check or Fixup

In some cases, it may make sense to create a new Check or Fixup based on an existing one, as an existing Check/Fixup may be quite well suited to a new task but require changes to certain settings within its own workflow.

The starting point can be one of the Checks/Fixups supplied as standard or a custom one with specific details that need to be altered. The user can duplicate an existing Check/Fixup and alter specific elements as necessary. The following example shows step by step how to duplicate and edit a predefined Check. The same procedure is used for Fixups.

Selecting a Check for the duplicate

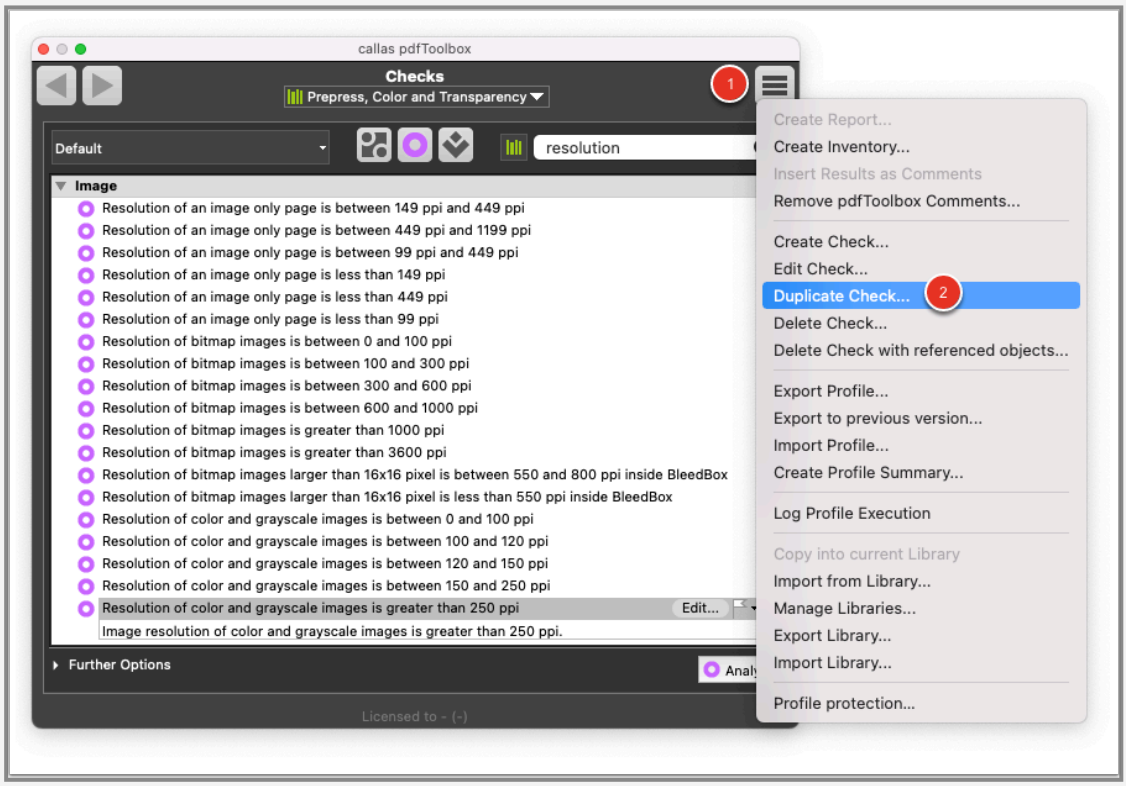


In the example shown, the predefined Check “Resolution of color and grayscale images is greater than *250 ppi*” (1) needs to be changed, raising the threshold to *350 ppi*.

The Check can be found under the **“Image”** (2) category and is a part of the **“Prepress, Color and Transparency”** (3) library.

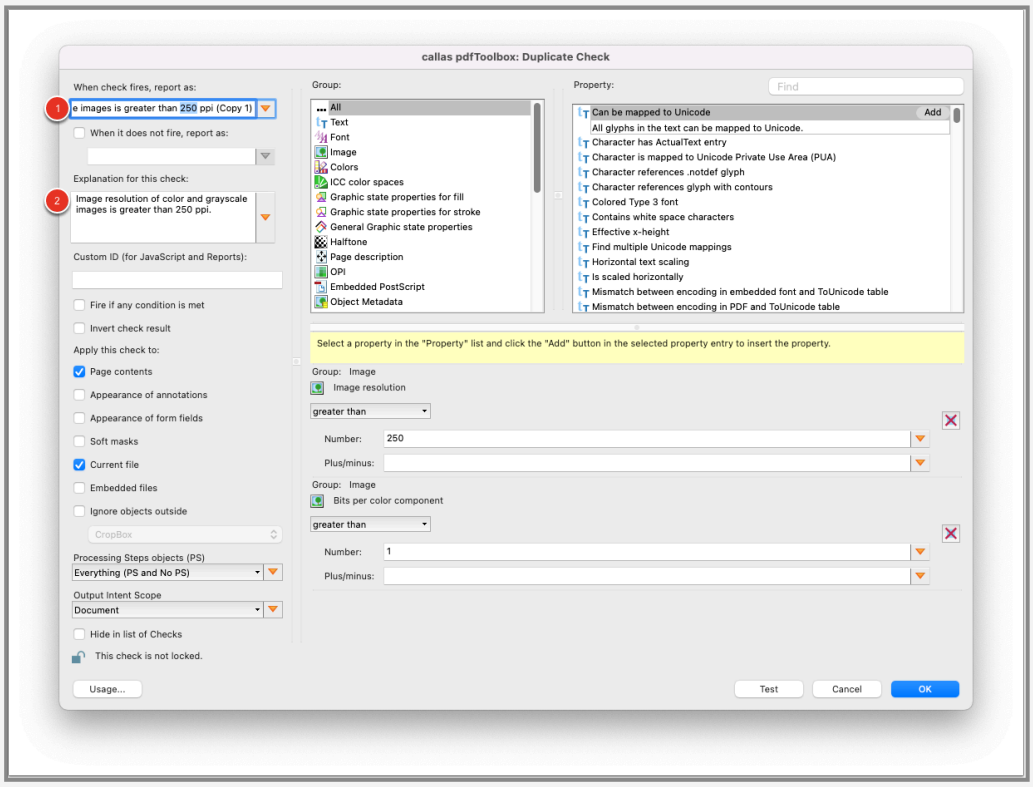
You can also use the **search field** (4) to search specifically for the Check.

Duplicating the Check



To duplicate a Check, first select it. In the flyout menu to the upper right (1), select “Duplicate Check” (2).

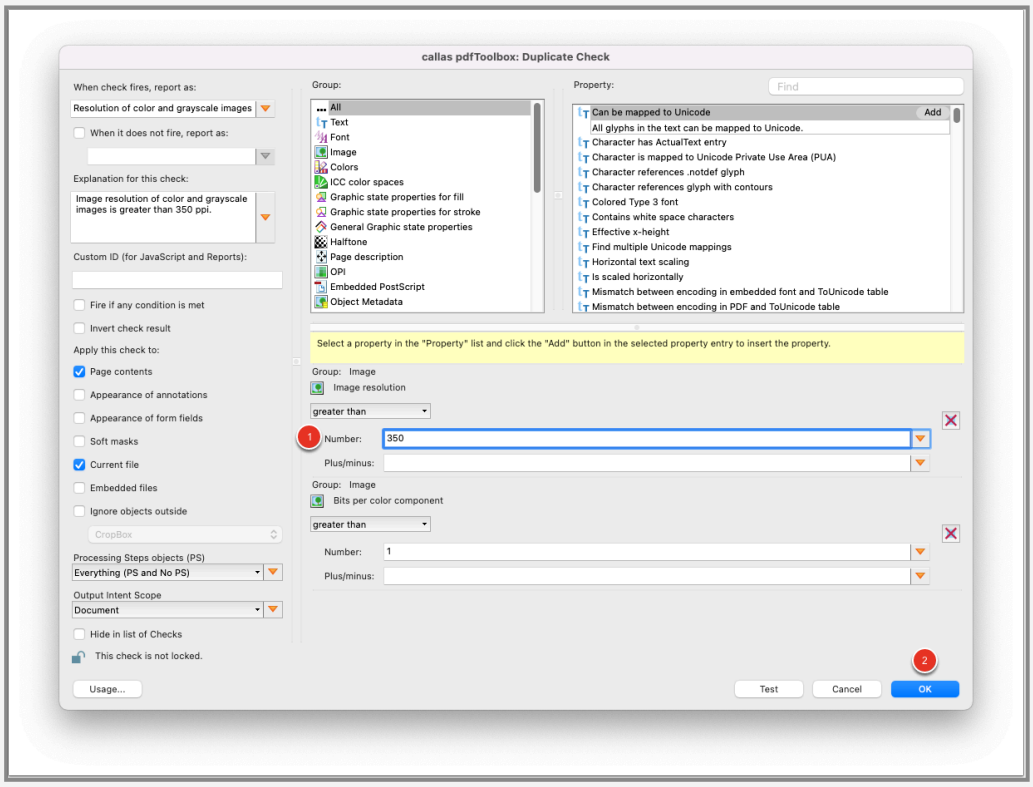
Changing settings for the duplicated Check



First, we should change the description of the Check so that its new function is clear. The name should be changed to: “Resolution of color and grayscale images is greater than *350 ppi*” (1).

We should also update the text in the Explanation field (2).

Editing properties

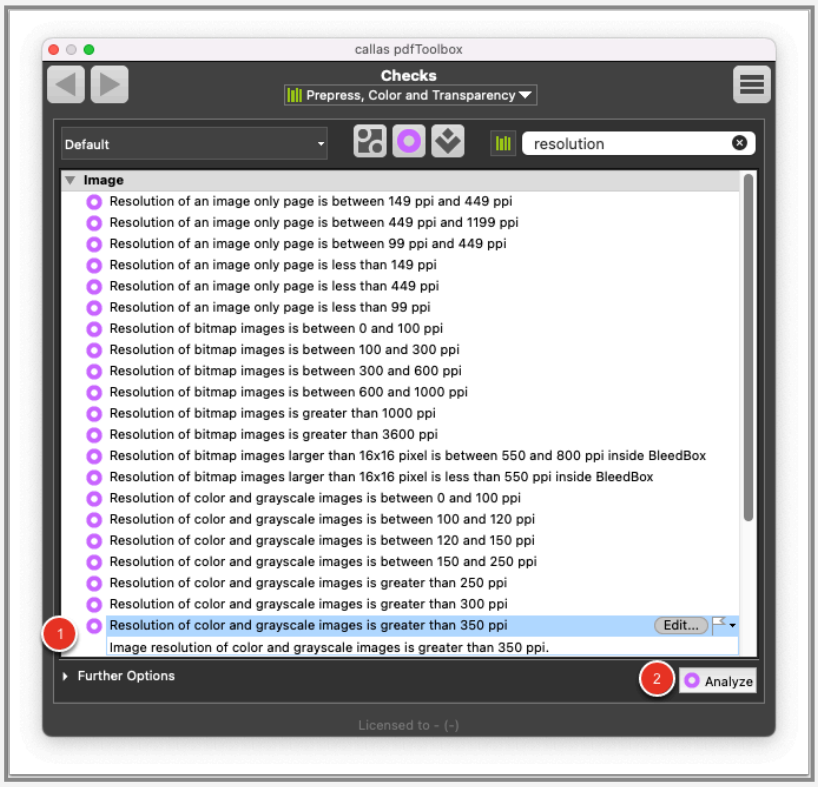


The image resolution for the duplicate is now raised from *250 ppi* to ***350 ppi*** (1).

> This Check consists of a combination of two Checks:
>
> 1. **Image resolution:** checks if the image resolution is greater than 350 dpi.
>
> 2. **Bits per color component:** checks if the image has more than 1 bit per color component. This means that only grayscale and color images are taken into account.

Click on the **OK** button (2) to save your changes.

The new Check in the Profile window



The new Check is now shown in the list **(1)**.

Click on **Analyze (2)** to find and report any applicable images in a PDF document with a resolution greater than 350 ppi.

The result window

This window shows the result of the Check. In this case a red cross indicating there is an error issued by the preflight Check. This means, that at least one image in the document has a higher resolution than 350 ppi. By double-clicking on the error message, pdfToolbox will show you the affected object by highlighting the image with a red frame. If you want to know the exact value that caused the error message, you can inspect the "Trigger values" (in the example, the image has a resolution of 400 ppi and 8 bits per color component).


callas pdfToolbox
Result
Prepress, Color and Transparency
callas pdfToolbox profile "Resolution of color and grayscale images is greater than 350 ppi" found the following errors:
Pages 1 - 3 from "Testfile.pdf"
Resolution of color and grayscale images is greater than 350 ppi
Resolution of color and grayscale images is greater than 350 ppi (1 match on 1 page)
Summary
Page 3: Color image 280.08x604.8 pt 400.0 ppi CMYK overprint: off
Images
Graphic properties
Trigger values
Bits per color component (bit): 8.0
Image resolution (ppi): 400.0
Overview
callas pdfToolbox information
Show in Snap
Certify
Create Report...
Licensed to - (-)

2.5 Creating Profiles

If you want to apply a number of Checks and/or Fixups to a document, it would be very time-consuming to perform each step individually. For these use cases, you should use Profiles. A Profile is a collection of any number of Checks and/or Fixups.

If the Profile contains both – Checks and Fixups – all Fixups are executed first. After modifying the document, pdfToolbox will run the Checks. This means that a Profile always executes the Fixups first and then the Checks.

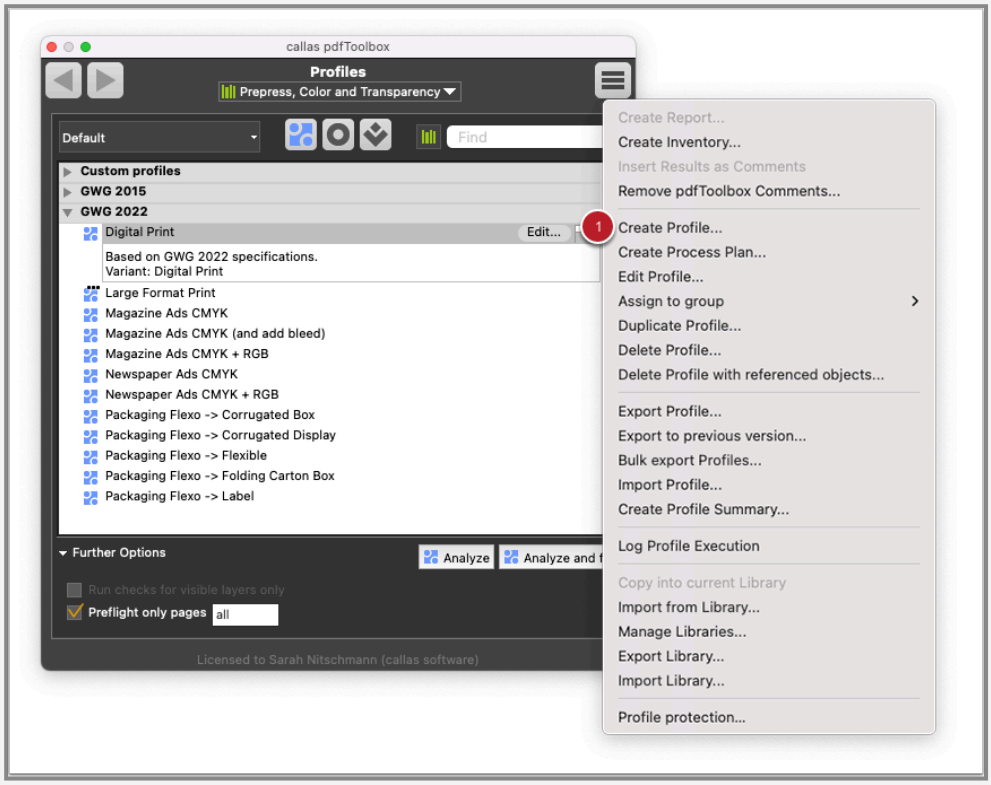
The Profile window

pdfToolbox provides a wide range of Profiles from a number of different categories, including digital printing, prepress, PDF analysis, PDF Fixups and PDF standards.



1. A number of predefined Profiles are supplied as standard. Clicking on an entry in the list will provide a short description of the Profile.
2. The highlighted icon indicates we're in the Profile section of the Profiles window.
3. After clicking on the "Edit…" button the "Edit Profile" window will be opened. In this window you can adjust the predefined Profile.
4. The "**Analyze**" button only does a quality control and will never change the PDF document: in the Profile you have selected, it will only execute the Checks. The "**Analyze and fix**" button also runs the Fixups of the selected Profile in addition to the Checks. The document may be modified in this case.
5. Profiles can be created from scratch. The following section is a step-by-step description of how to create a new Profile.
6. One or more profiles (multiple selection) can be assigned to another or a new group.
7. It is possible to duplicate an existing Profile and alter specific elements as necessary.
8. Profiles can be deleted. Either you delete only the Profile (all Checks and Fixups contained in it remain) or you delete the Profile together with all referenced Checks and Fixups.
9. Profiles can also be exported and imported. This may be of interest if you need to forward or distribute them within your organization, or if you work with external partners. **Bulk export Profiles:** Allows you to export multiple profiles at once. It opens a new dialog box where a selection of profiles can be made to be exported all at once.

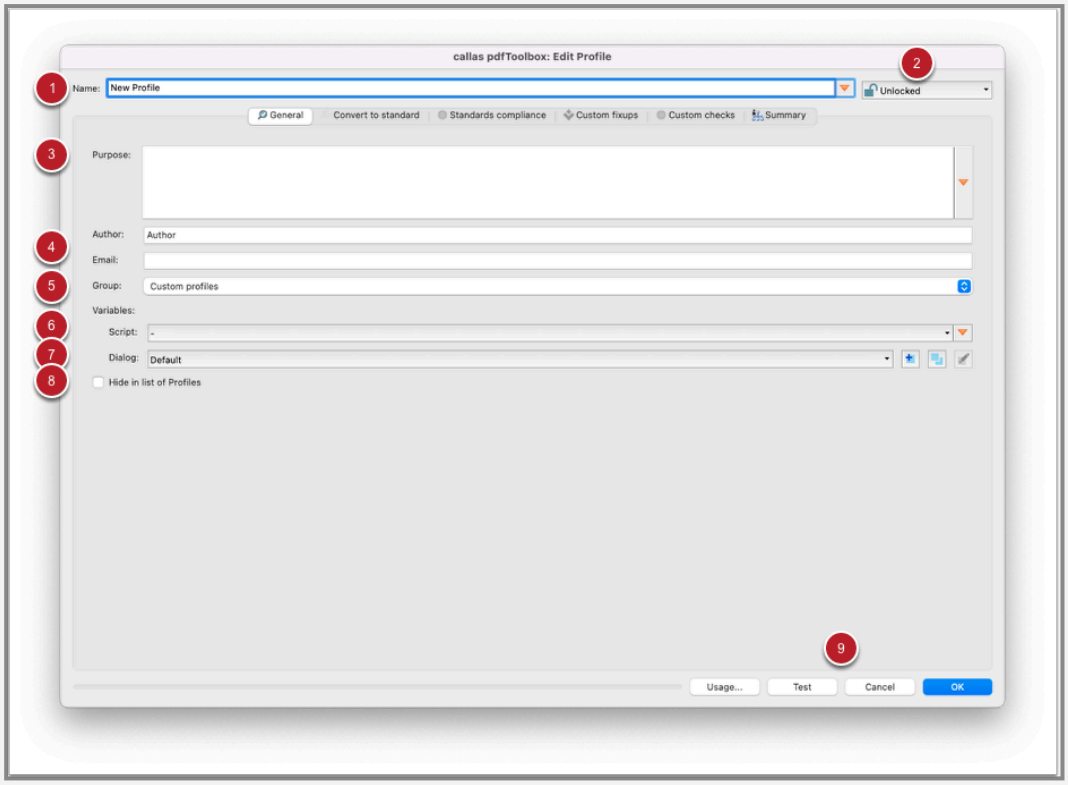
How to create a new Profile?



1. Click on the “Create Profile...” in the pull-down menu.

The “Edit Profile” window: an overview

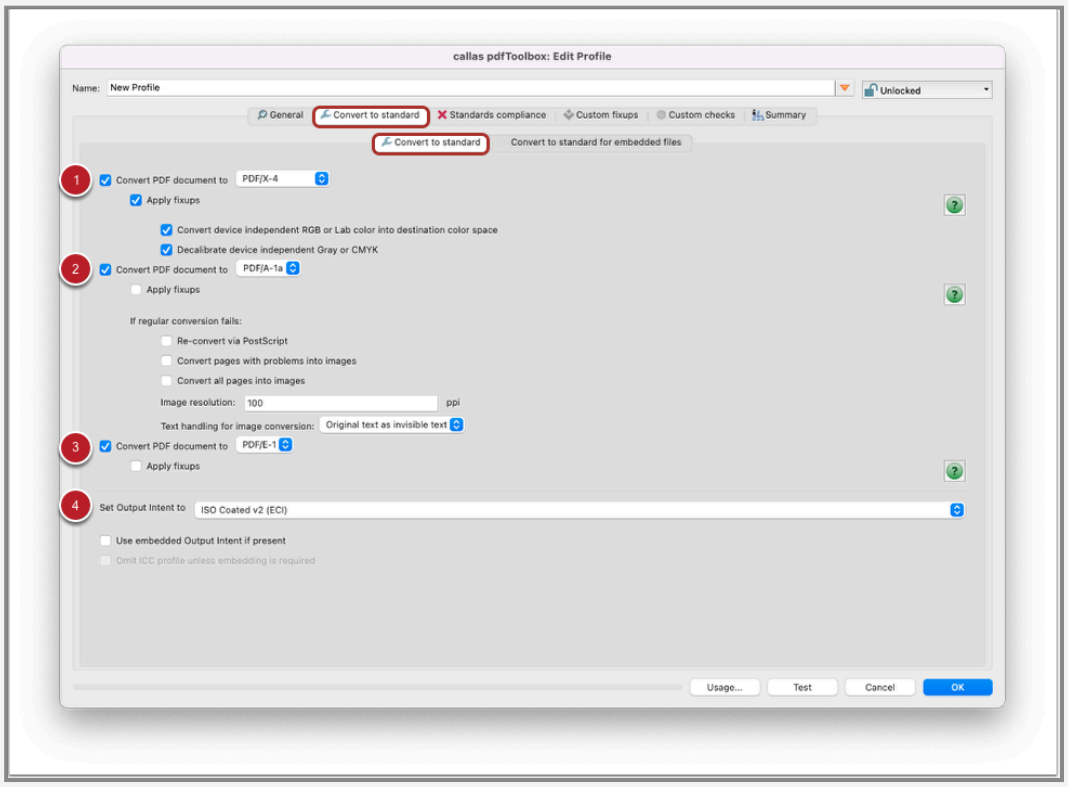
Edit Profile: General information



1. The **name** should be specified according to the purpose of the profile. The orange triangle (here and elsewhere) allows you to include variables and scripts.
2. A Profile can be **unlocked**, **locked** or password protected.
3. The **Purpose** field allows you to add explanatory text that will later be displayed in the profile list when a profile is selected.
4. **Author** and **Email** are intended to provide information about the author of the Profile.
5. Under **Group**, you can specify whether the profile should be stored under Custom Profiles or in another group.
6. The Profile can also be supplied with a **script**.
7. If checked, the Profile will be **hidden** in the list of Profiles in the Library.
8. At the lower right part of the window, you will find the **Usage...** button (Info in which Process Plan this Profile is

used), as well as Test (lets you test the Profile within their respective editor), **Cancel** and **OK** button **(4)**.

Edit Profile: Convert to standard



The **Convert to standard** section is particularly extensive.

It is divided into conversion parameters for PDF/X (prepress), PDF/A (long-term archiving) and PDF/E (digital construction drawings).

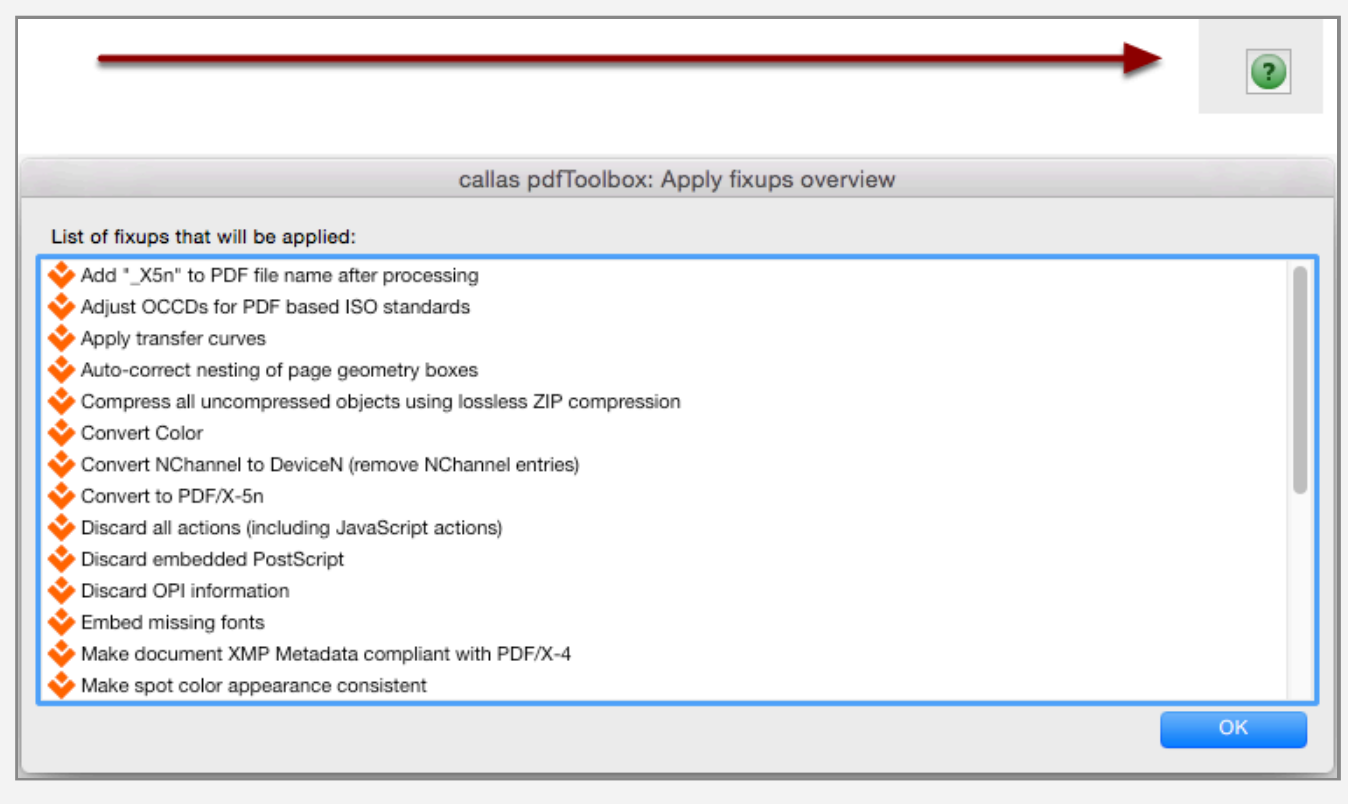
This section converts and fixes PDF files, rather than simply checking them.

1. Under **Convert PDF document to PDF/X**, you can choose from a range of PDF/X versions from a pull-down menu (PDF/X-1 - PDF/X-6). You can optionally **Apply Fixups** by checking the box. Depending on the standard selected, two other options will be available: **Convert device independent RGB or Lab color into destination color space** and **Decalibrate device independent Gray or CMYK**.
2. Under **Convert PDF document to PDF/A**, you can choose from a range of PDF/A versions from a pull-down menu. The **Apply Fixups** box can be checked. **If regular conver-**

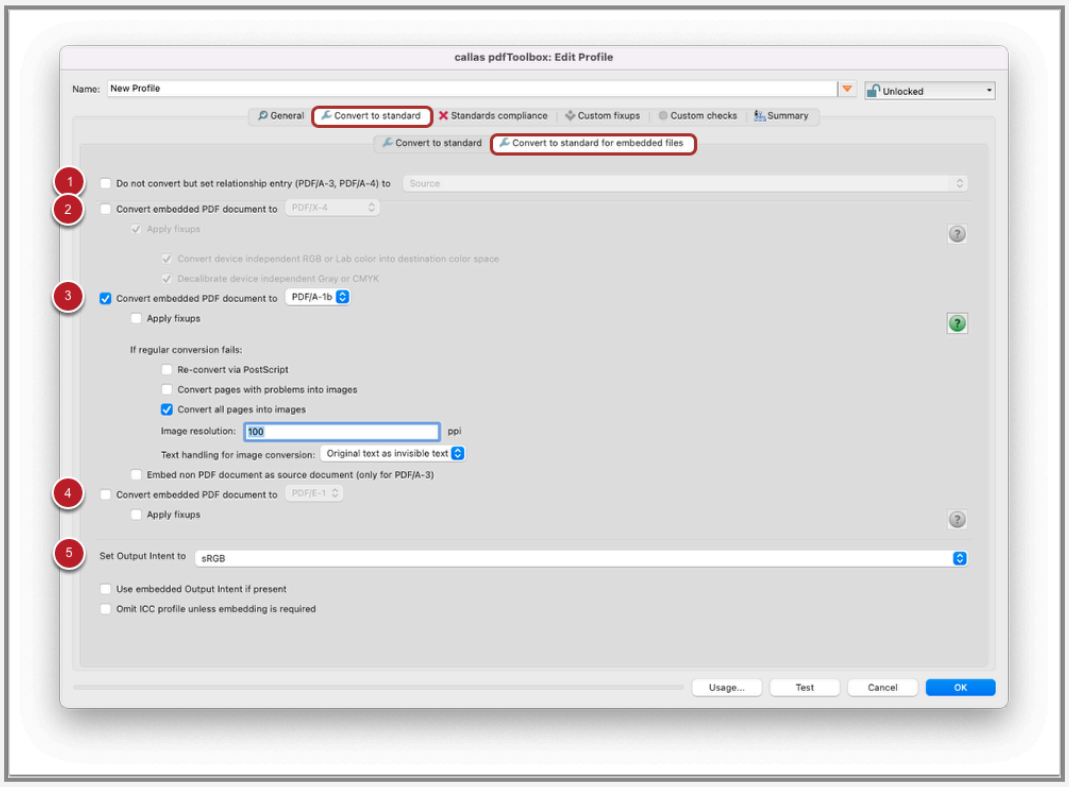
sion fails, a number of additional options are available. **Re-convert via PostScript**, **Convert pages with problems into images**, or **Convert all pages into images**. The last two of these options allow you to specify an **Image resolution**.

3. The third standard is **PDF/E**. You can again **Apply fixups** here.
4. Finally, you can also select the **Output Intent** from a pull-down menu. Two options are available here: **Use embedded Output Intent if present** and **Omit ICC profile unless embedding is required** (e.g. with modern Output Intents; this allows you to reduce the size of the resulting file to the size of the Output Intent).

To learn more about the **Fixups** that will be applied during each conversion type, click on the **green question mark button** to open the corresponding list. (The button is gray, meaning inactive, when a conversion type is also inactive.)



Edit Profile: Convert to standard for embedded files

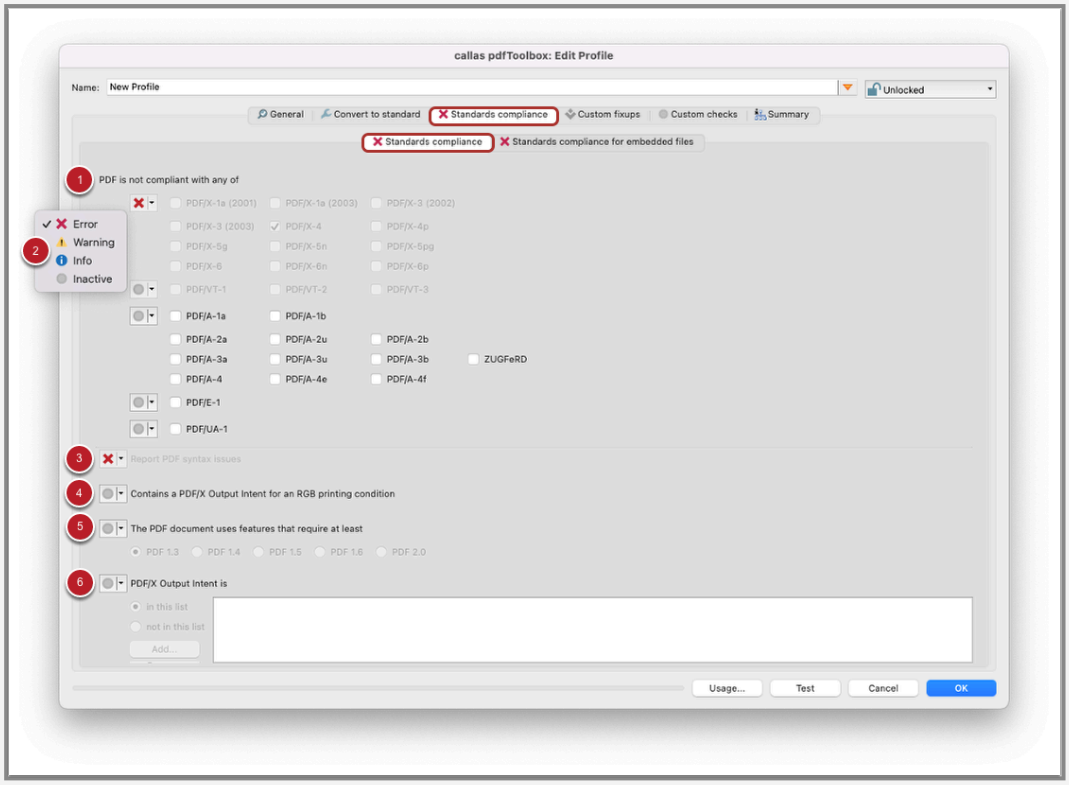


Convert to standard for embedded files offers very similar parameters to those shown in the previous step, but in this case they relate to embedded PDF files.

1. The first section deals with the special case of PDF/A-3 and PDF/A-4. These PDF standards also allow you to embed files left in their original state (not just PDF, but also other formats). **Do not convert but set relationship entry (PDF/A-3, PDF/A-4) to ...** allows you to define the relationship in one of a number of different ways, including **Source**, **Data**, **Supplement** and others.
2. **Convert embedded PDF document to PDF/X** enables a number of different options as well as **optional Fixups**.
3. **Convert embedded PDF document to PDF/A** enables a number of different options as well as **optional Fixups**. In addition, there are also the same backup options as before to determine how to proceed if regular conversion fails.
4. **Convert embedded PDF document to PDF/E** plus **optional Fixups**.

5. **Output Intent** settings.

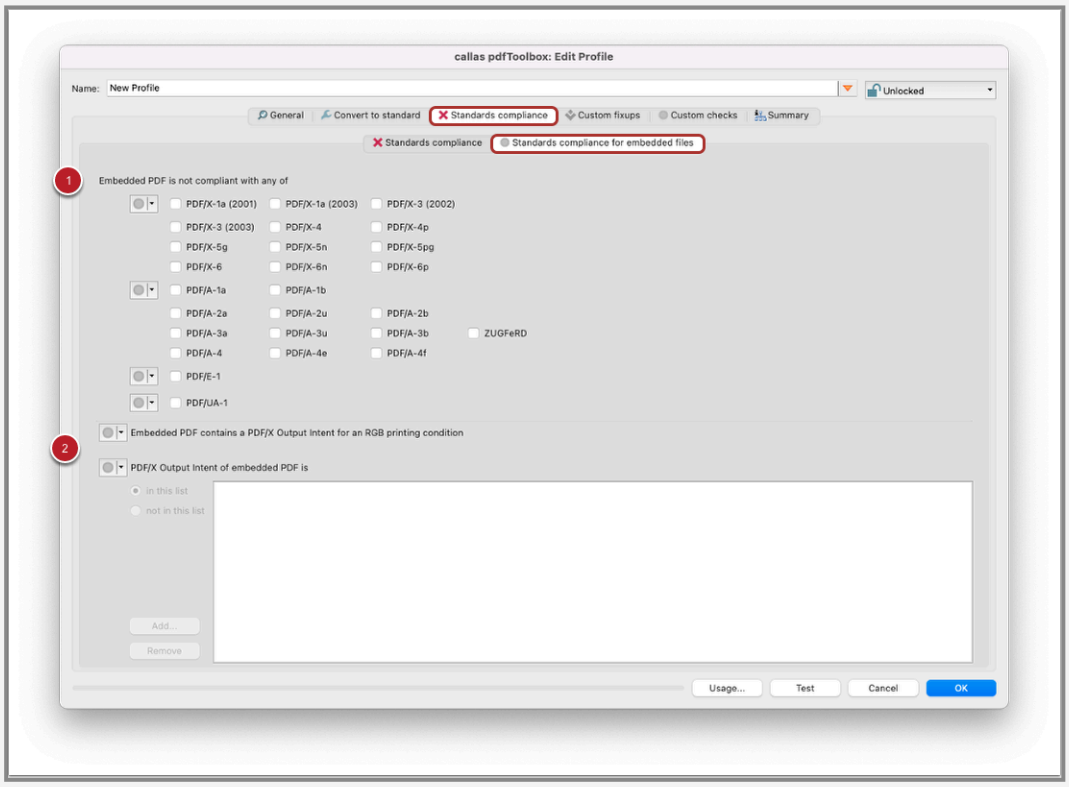
Edit Profile: Standards compliance



The **Standards compliance** section relates to the various ISO standards available for the Portable Document Format.

1. The category **PDF is not compliant with any of** allows you to integrate all currently available PDF standards into the query.
2. Using the pull-down menu (as with the other entries) you can specify how to report anomalies - as an **error**, as a **warning**, or as **info**. By default, this setting is **inactive**.
3. "**Report PDF syntax issues**" indicates any issues with the PDF syntax like missing color space or missing XObjects etc.
4. Here, you can determine whether a PDF contains a **PDF/X Output Intent for an RGB printing condition**.
5. The first item affects **PDF functions** depending on the **PDF version**: “The PDF document uses features that require at least Acrobat 4.0 (PDF 1.3) and others.
6. An embedded PDF/X Output Intent can also be compared against a custom **list** of Output Intents.

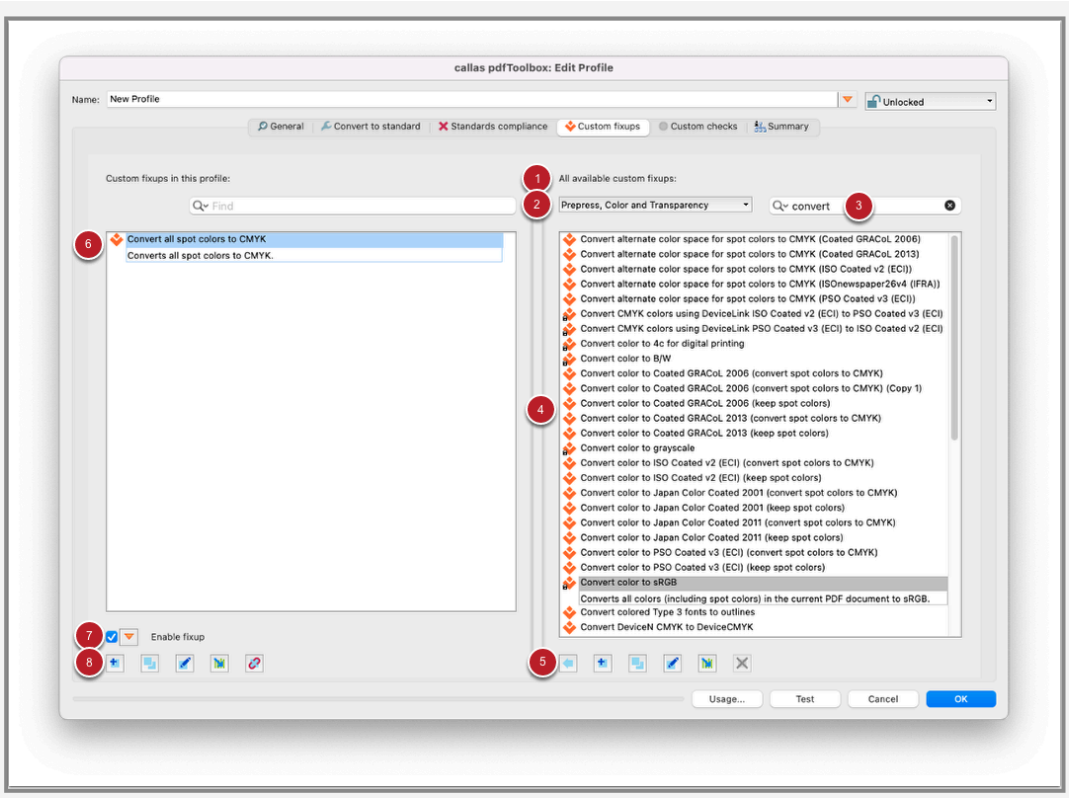
Edit Profile: Standards compliance for embedded files



The **Standards compliance for embedded files** section offers the same parameters as in the previous section. In this case, however, they relate to embedded PDF files:

1. The **PDF is not compliant with any of** option again relates to all currently available PDF standards.
2. The second section again turns to the **PDF/X** standard for PDF printing.

Edit Profile: Custom Fixups



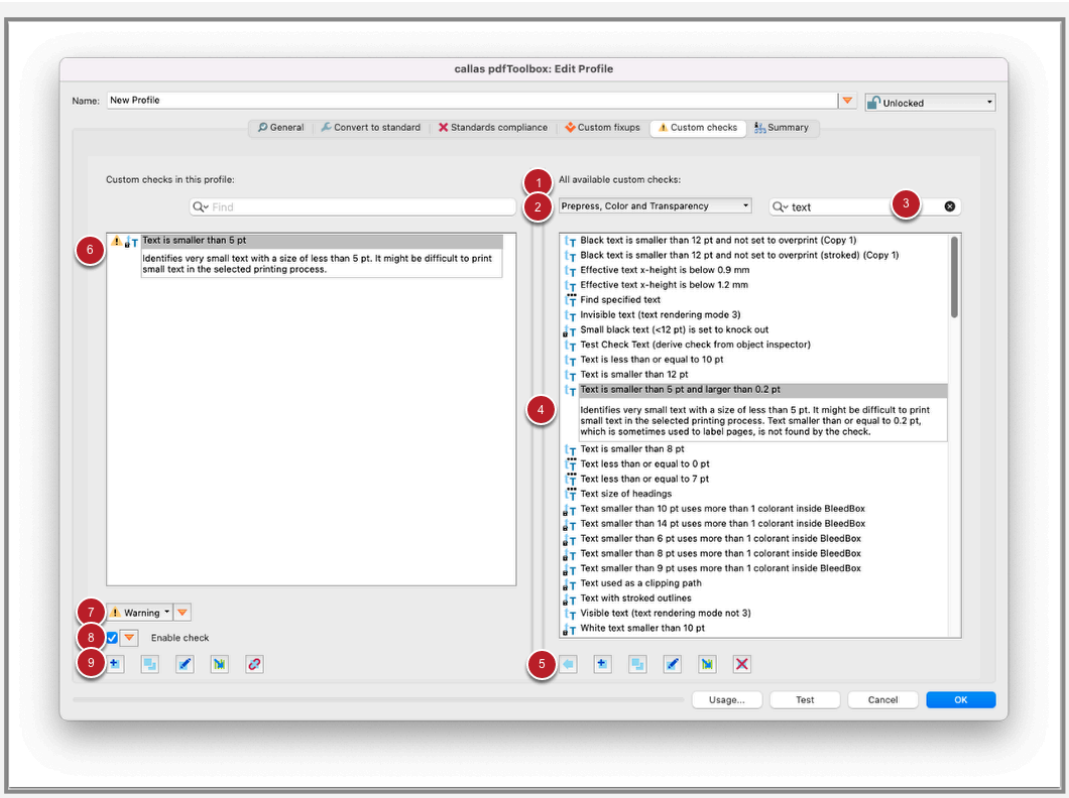
Under **Custom Fixups**, you can select entries from the list of all available pdfToolbox Fixups and integrate them into the new Profile.

The window is structured as follows:

1. The right hand side shows the **list of Fixups available** in pdfToolbox. It can be considered the “inventory” available to you.
2. The Fixups shown will be those available in the **currently selected Library** (in this example, Essentials).
3. If you know the name of a given Fixup, you can use the **Search** tool to make it easier to find.
4. Click on a Fixup in the list to see more **detailed information**.
5. Click on the **blue arrow symbol to the left** to move the selected Fixup to the middle column. This will add it to the Profile you are currently creating or editing.
6. The middle column shows **all Fixups** used in the **current Profile**.
7. You can enable or disable this Fixup.

8. The buttons allow you to **add**, **duplicate**, **edit**, **import** and **delete** Fixups.

Edit Profile: Custom Checks



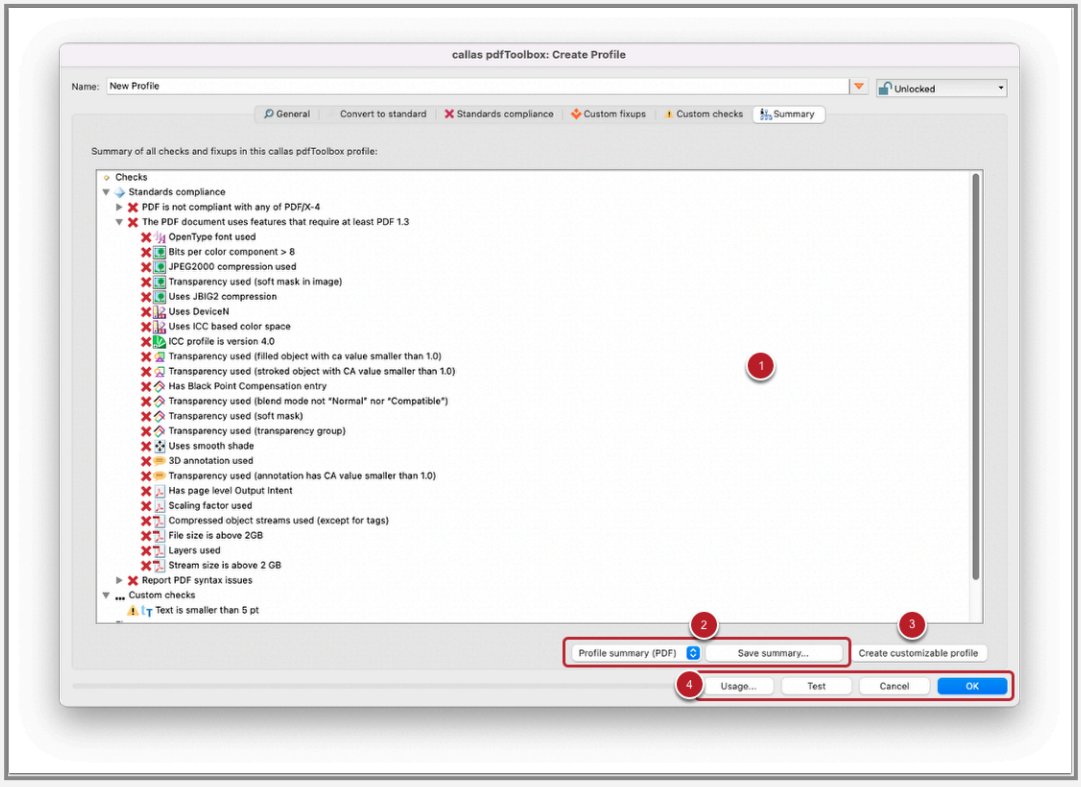
Under **Custom Checks**, you can select entries from the list of all available pdfToolbox checks and integrate them into the new Profile.

The window is structured as follows:

1. The right hand side shows the **list of Checks available** in pdfToolbox. It can be considered the “inventory” available to you.
2. The Checks shown will be those available in the **currently selected library** (in this example, Prepress, Color and Transparency).
3. If you know the name of a given Check, you can use the **Search** tool to make it easier to find.
4. Click on a Check in the list to see more **detailed information**.

5. Click on the **blue arrow symbol to the left** to move the selected Check to the middle column. This will add it to the Profile you are currently creating or editing.
6. The middle column shows **all Checks** used in the **current Profile**.
7. You can use the pull-down menu to specify whether the Check should return an **Error**, a **Warning** or an **Info** message. You can also use the orange triangle to add and manage **variables and scripts**.
8. You can enable or disable this Check.
9. The buttons allow you to **add**, **duplicate**, **edit**, **import** and **delete** Checks.

Edit Profile: Summary



In the **Summary**, you can see a detailed overview of all Checks and Fixups included in the newly created (or edited) Profile.

1. The entries are grouped by section, i.e. **Document**, **Pages**, **Images**, **Colors**, **Fonts** and so on.

2. *Optional:* You can save the Profile **summary**. Starting with **pdfToolbox 16** the following file formats are supported for the Profile summary:
   - Profile summary (PDF): PDF file with all Fixups and Checks that are part of the Profile
   - Profile summary (CSV): comma-separated file with all Fixups and Checks that are part of the Profile
   - Variable summary (CSV): comma-separated file with all Variables (simple and script variables) that are part of the Profile
3. *Optional:* You can also **generate a customizable profile** based on the current Profile - a duplicate, in other words - which you can configure further.
4. At the lower right of the window, you will find the **Usage...** button (Info in which Process Plan this Profile is used), as well as Test (lets you test the Profile within their respective editor), **Cancel** and **OK** button (**4**).

2.6 Duplicate and edit Profiles

Profiles consist of a combination of Checks and/or Fix-ups. With pdfToolbox, a number of predefined Profiles are supplied as standard.

The Profile window



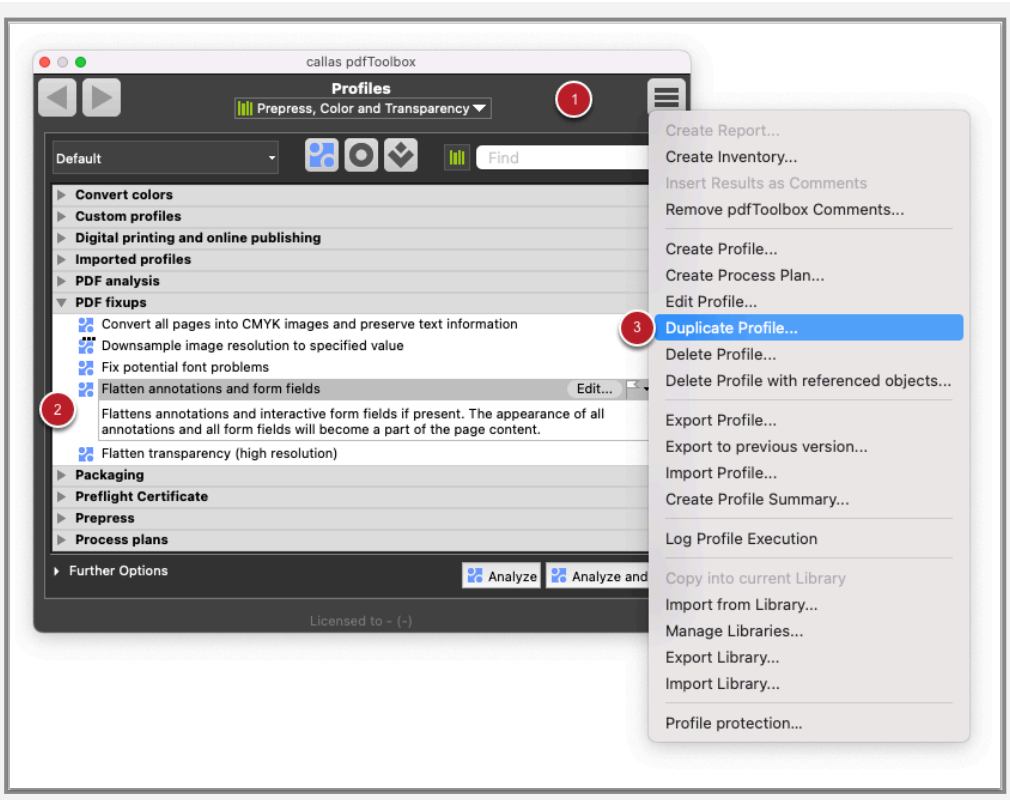
Clicking on an entry in the list containing the Profiles will provide a short description of the relevant details (1).

If custom tasks are needed, users can create their own Profiles. These can be altered versions of existing Profiles (see section below) - in which case you can work with duplicates (2) - or new Profiles can be created from scratch (3).

Creating and setting up a duplicate of a Profile

In some cases, it may make sense to create a new Profile based on an existing one, as an existing Profile may be quite well suited to a new task but requires changes to certain settings. In that case you can duplicate an existing Profile and adapt it to your needs. We'll show you how this works step by step in this article.

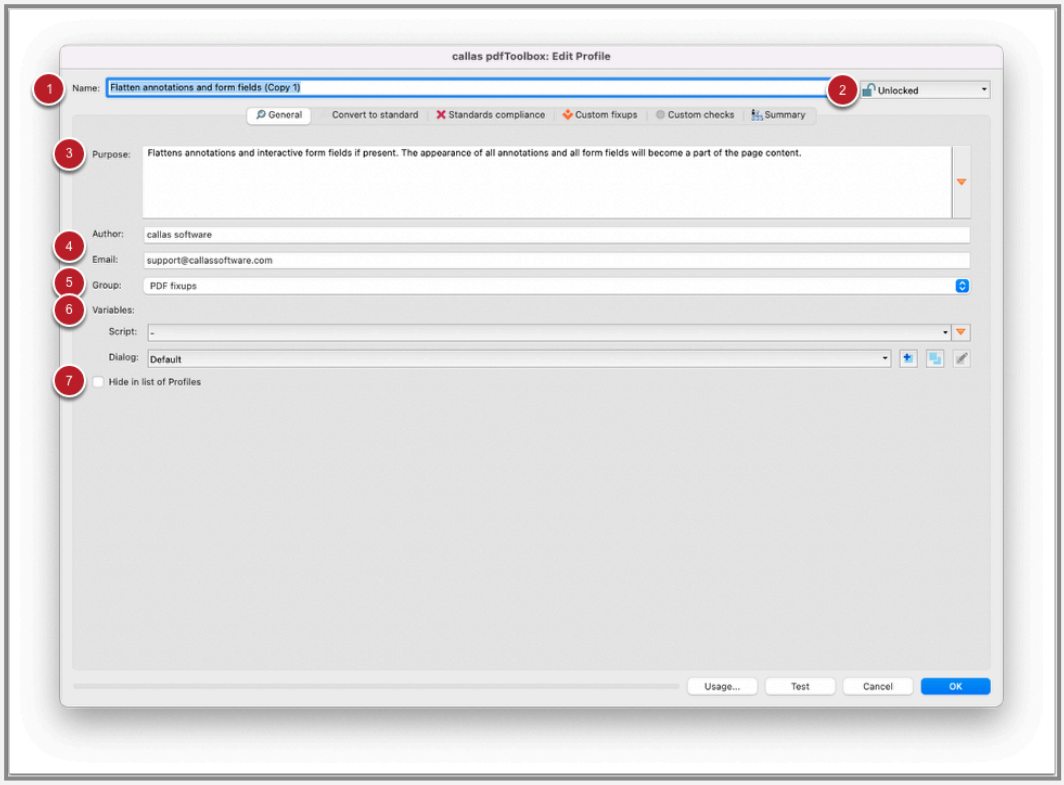
Selecting a source Profile for the duplicate



In the example shown, the Profile "Flatten annotations and form fields" (1) needs to be altered. The new Profile should only flatten form fields and move annotations out of the BleedBox. The Profile can be found in the "PDF Fixups" category and is a part of the "Prepress, Color and Transparency" library.

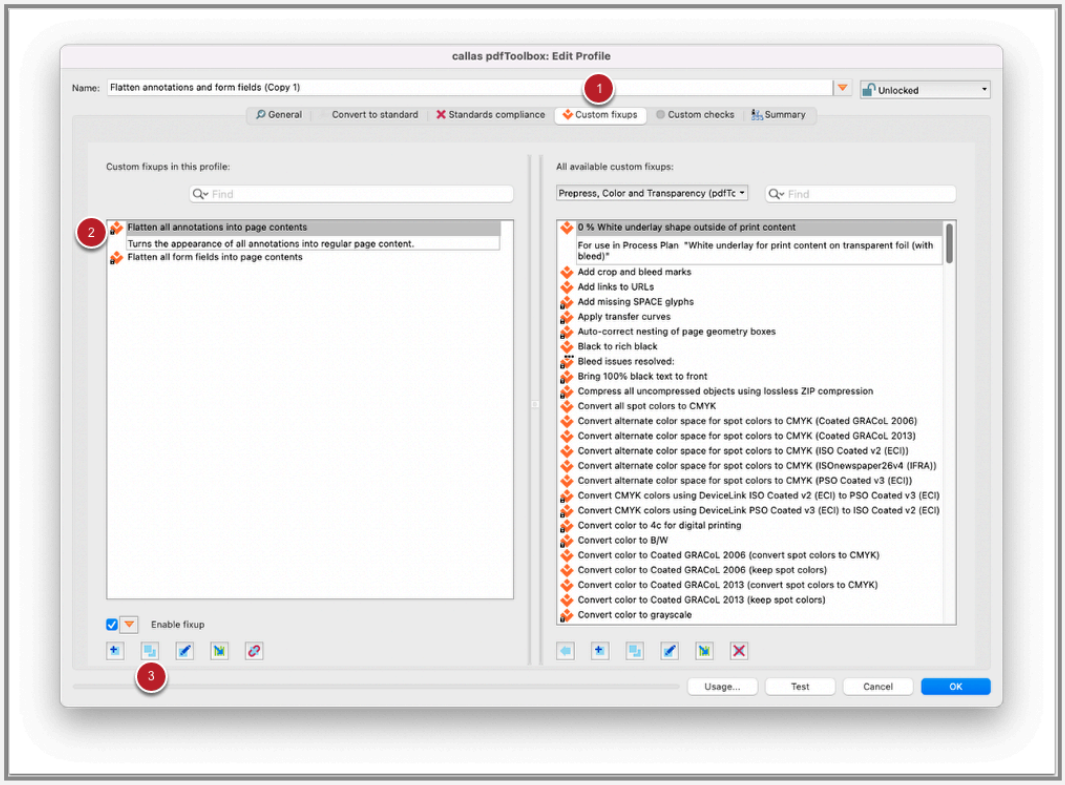
To duplicate a Profile, first select it. In the flyout menu to the upper right (2), select "Duplicate Profile" (3).

Changing settings for the duplicated Profile



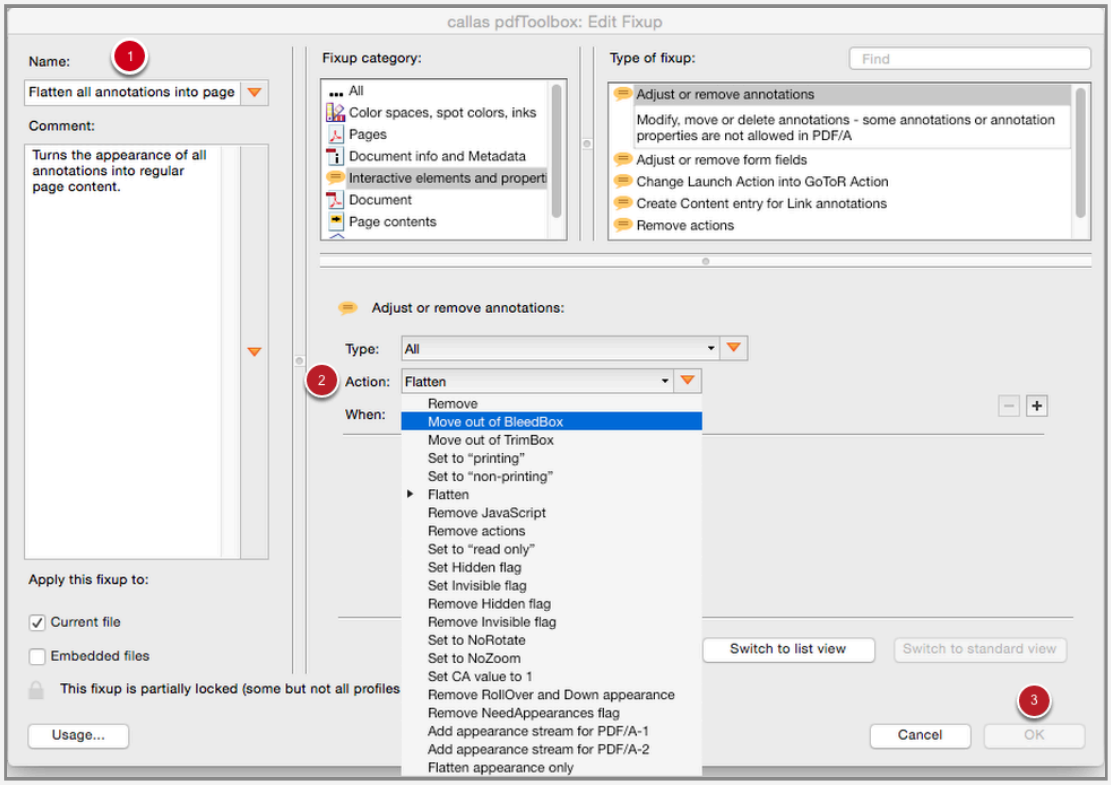
1. We should change the **name** of the Profile so that its new function is clear. In the example shown, the automatically generated text “Flatten annotations and form fields **(Copy1)**” can be changed to “Move annotations and flatten form fields” in the relevant box.
2. You can **protect** the Profile from being changed by clicking on the **lock symbol**.
3. We should also update the text in the **Comment** field.
4. Provide your own details in the **Author** and **Email** fields.
5. As this Profile is a duplicate, the “PDF Fixups” **group** is still suitable.
6. No **Variables** are used for this Profile.
7. We don't want to hide the Profile in the list of the Profile window.

Edit Profile



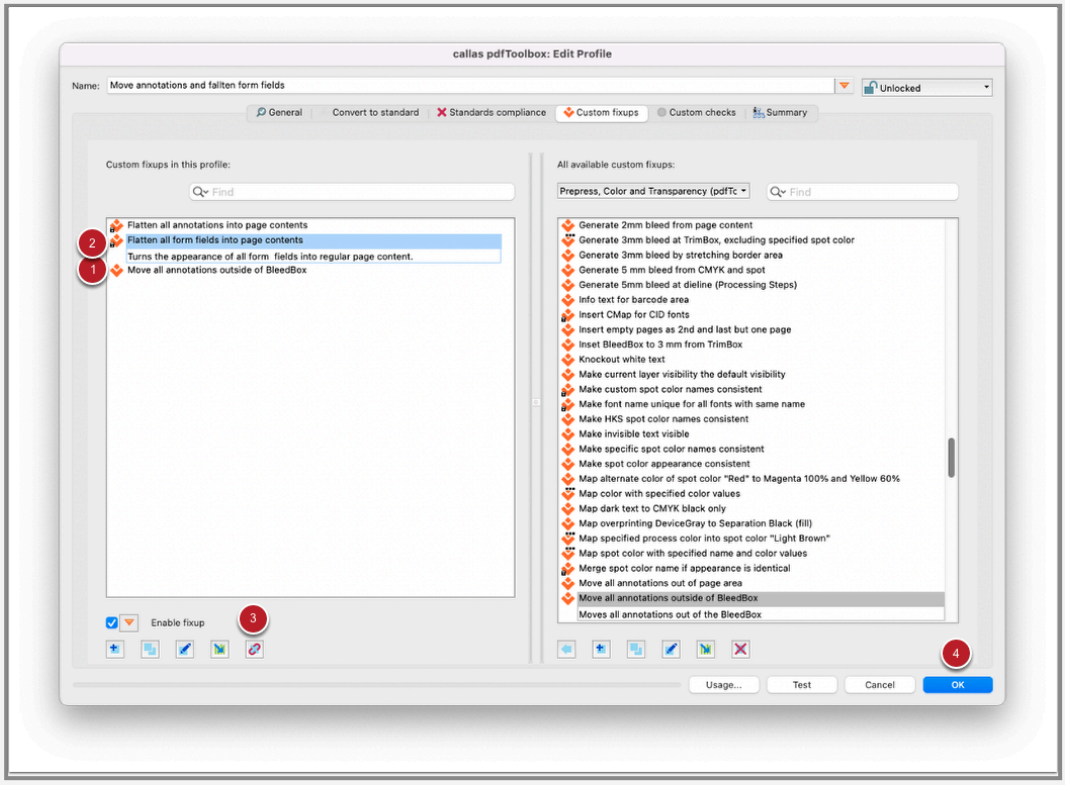
1. The settings to be changed in this example can be found under **“Custom Fixups.”**
2. For the original Fixup, we will select “Flatten all annotations into page contents.”
3. We will then duplicate the Fixup by clicking on the **duplicate symbol**.

Settings under “Duplicate this Fixup”



1. For this new duplicated Fixup, we should again edit the **name** and comment in the left column.
2. The new Profile should move annotations instead of flattening them. We therefore select the appropriate entry from the pull-down menu under **“Action”**.
3. Click **OK** to accept changes.

Results and advanced settings under “Edit Profile”



1. The **new Fixup** is now listed in the middle column.
2. The **previous Fixup** must now be **removed**. To do so, select the corresponding item in the list.
3. To remove it, click on the **broken link symbol** to break the link between the selected Fixup and the Profile.
4. Click **OK** to save and close the altered Profile.

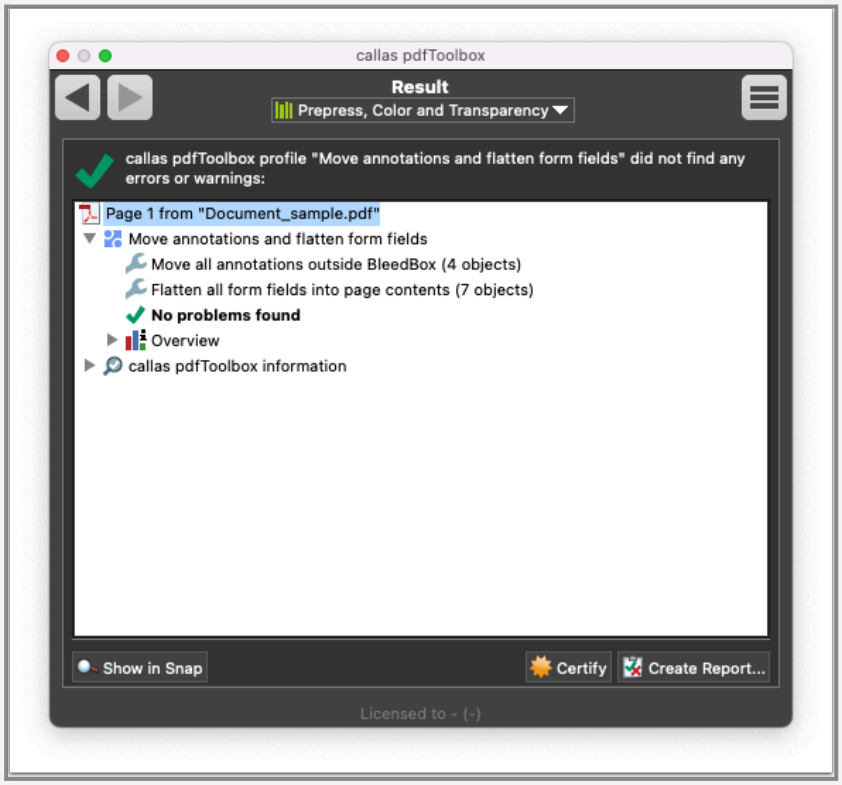
The new Profile in the Overview window



The new Profile is now shown in the list of Profiles (1). Click **Analyze and fix** (2) to execute the Profile.

The result window

This window shows the result of the Profile. All Fixups performed by the Software are listed here. The green check indicates that the Profile has been successfully executed and there are no problems found. 4 annotations has been moved outside the BleedBox and 7 form fields has been flattened into the page content.


callas pdfToolbox
Result
Prepress, Color and Transparency
callas pdfToolbox profile "Move annotations and flatten form fields" did not find any errors or warnings:
Page 1 from "Document_sample.pdf"
Move annotations and flatten form fields
Move all annotations outside BleedBox (4 objects)
Flatten all form fields into page contents (7 objects)
No problems found
Overview
callas pdfToolbox information
Show in Snap
Certify
Create Report...
Licensed to - (-)

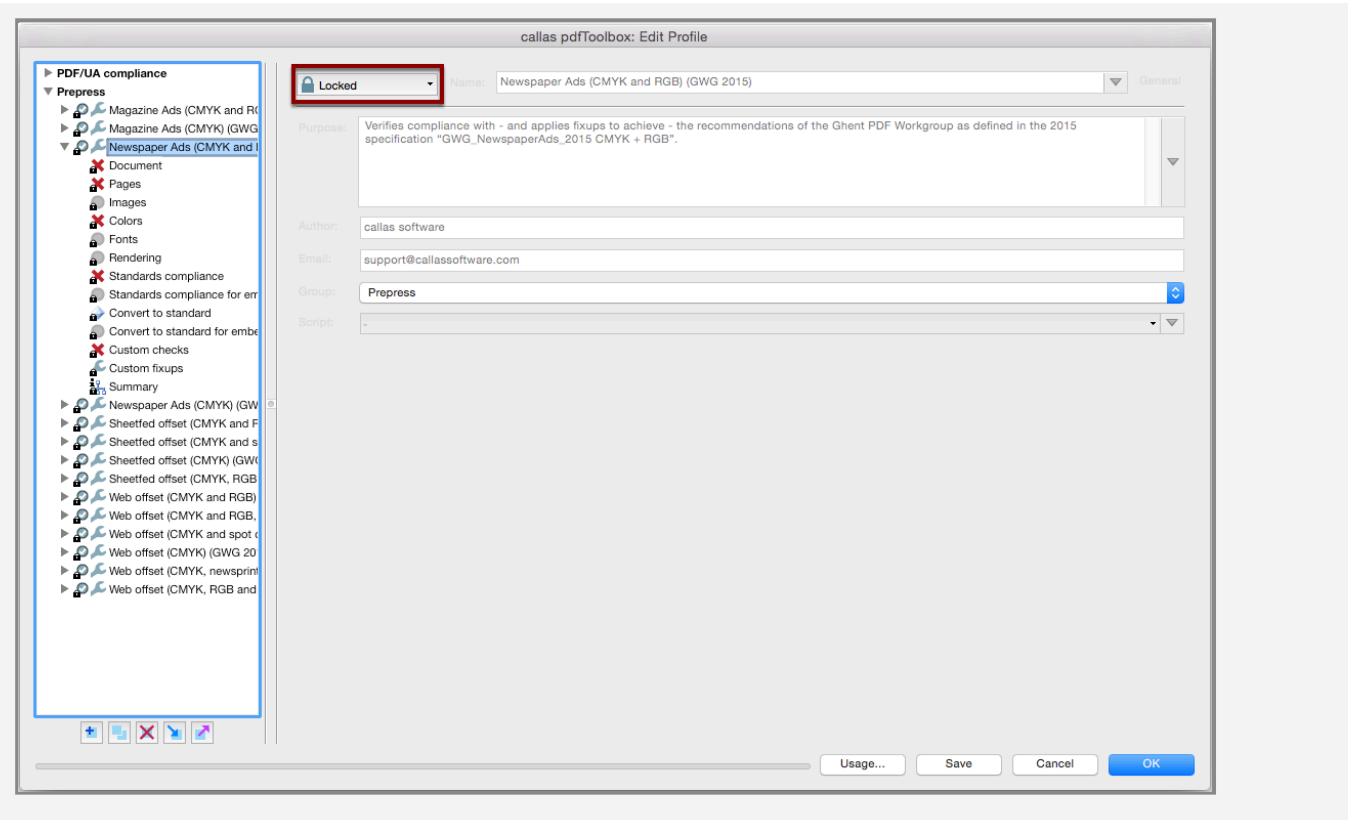
2.7 Protect Profiles, Checks, Fixups, Process Plans and Libraries against changes

You can prevent unplanned changes to Fixups, Checks, Profiles, Process Plans and Libraries by locking them - i.e. by making them write-protected.

Checks and Fixups cannot be individually locked; instead, they are automatically locked whenever a Profile which uses them is itself locked.

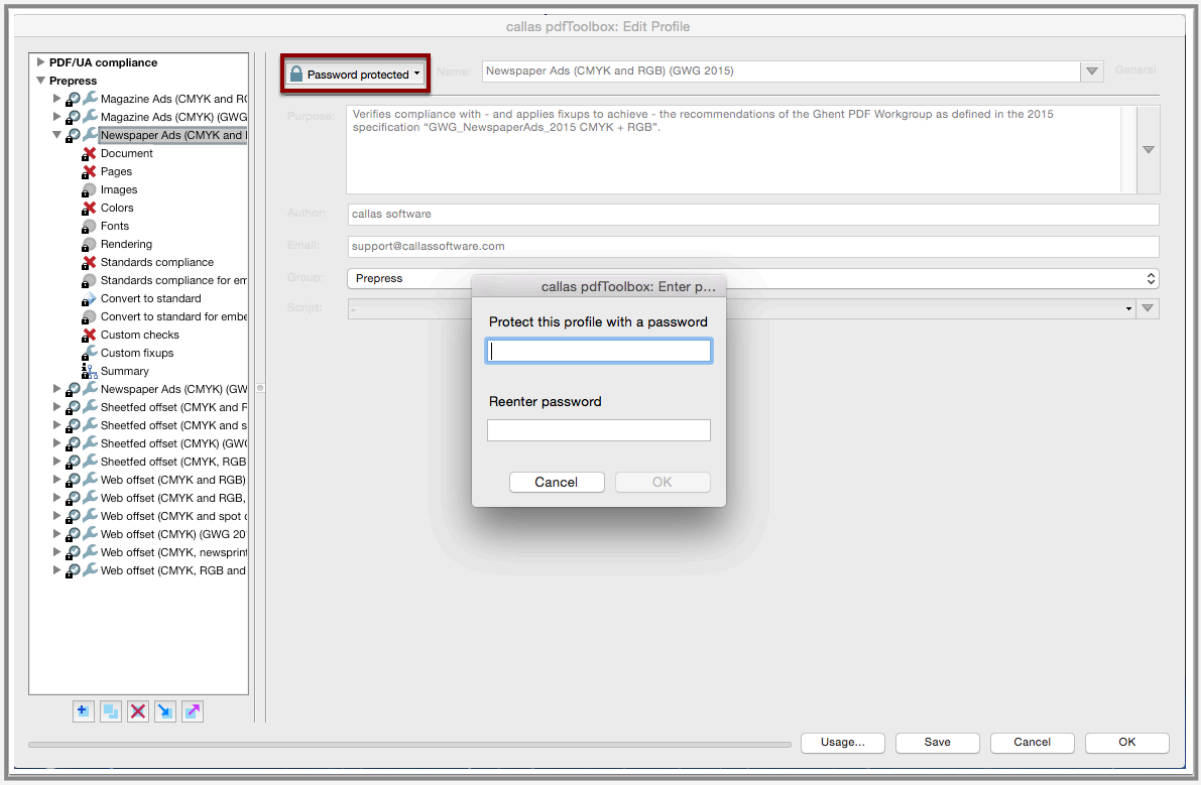
Some of the predefined Fixups, Checks and Profiles are protected by default. They can be identified by looking for the lock icon.

Locked Profile



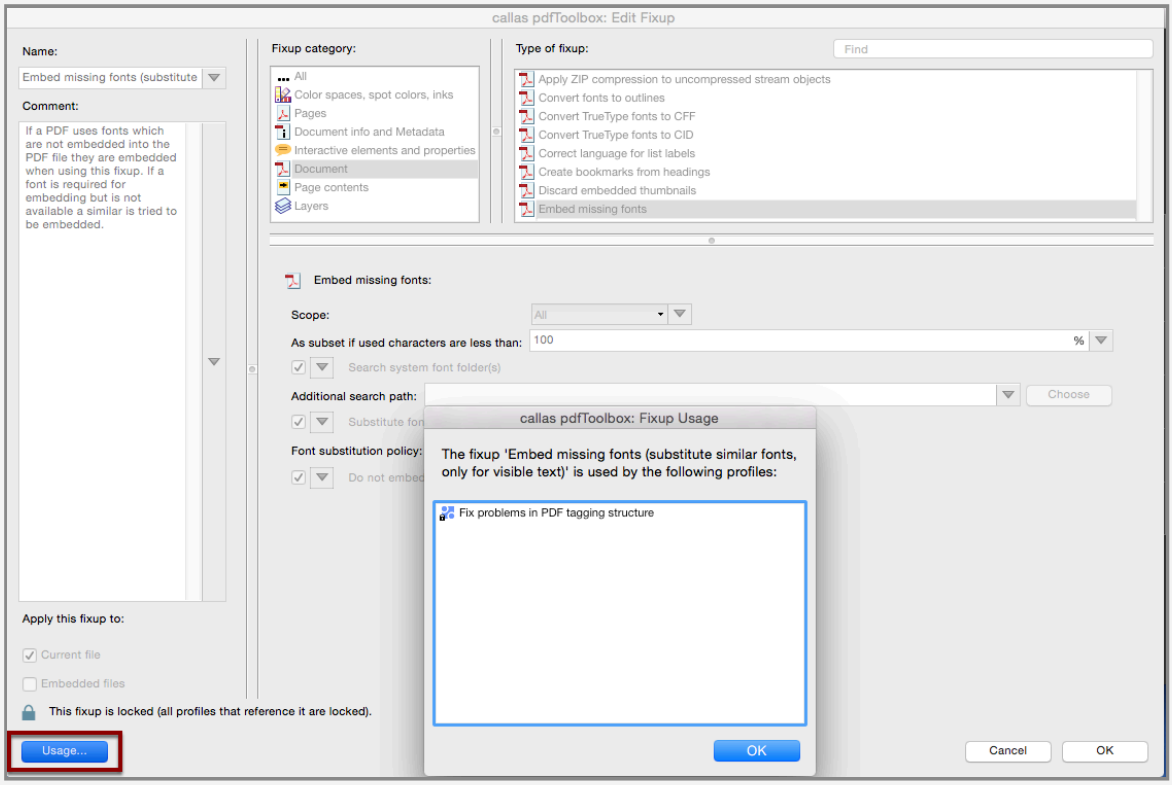
A locked profile is protected against changes.

Password-protected Profile



Alternately, a Profile can be assigned a password.

Fixup used in other Profiles or Process Plans



A Fixup (or a Check) can only be changed if it is not being used in a locked Profile or Process Plan.

You can see how it is being used in the corresponding interface marked “Usage...” within the same dialog.

Call up the listed Profiles as required, click “Edit...” and change the lock setting to “Open”.

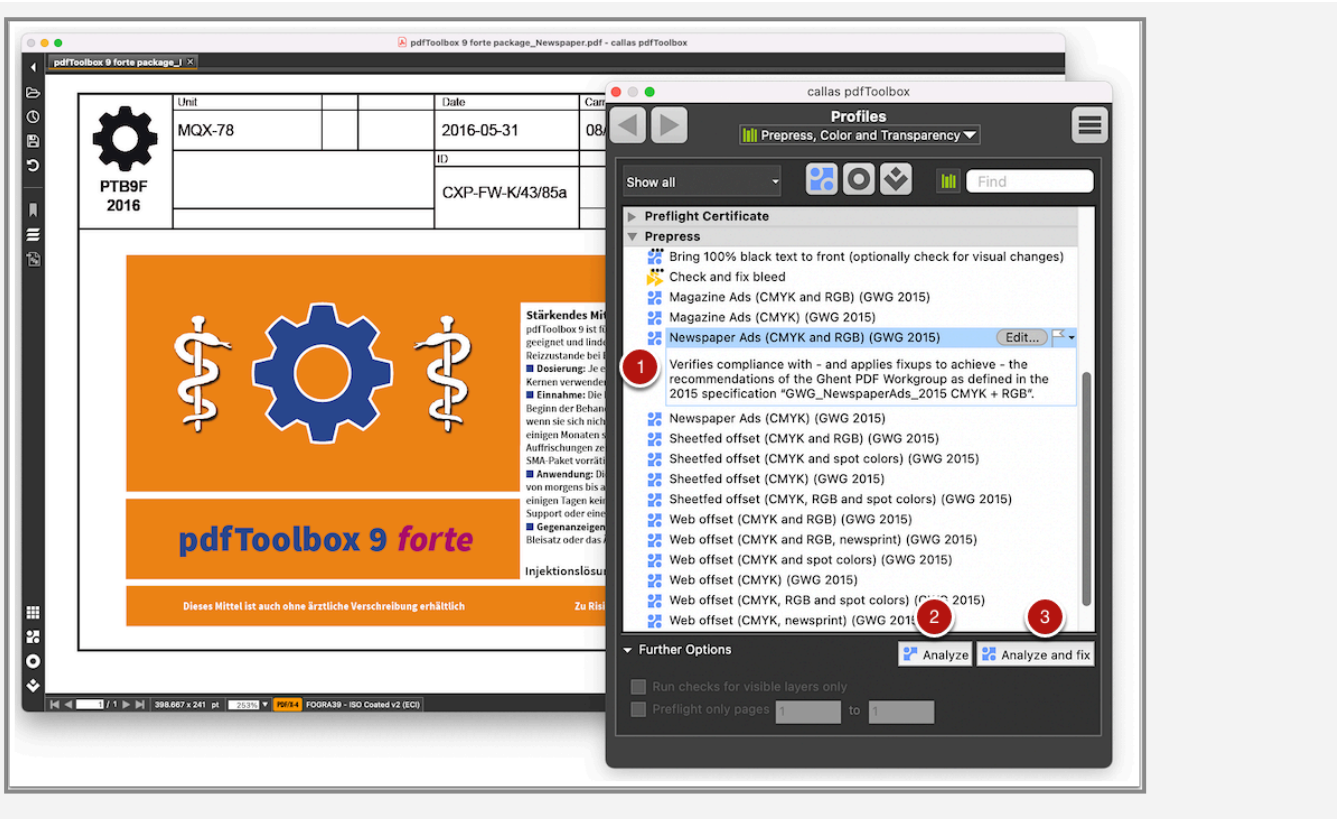
You can then edit the Fixup (or the Check).

2.8 Running a Profile and examining results

pdfToolbox uses Profiles to perform quality control and/or to fix PDF documents. This article explains how you use those Profiles and how you can drill-down in the results of using such a Profile.

Running a Profile to analyze or fix a PDF document

To use a Profile, first open the Profiles Window by using the "Tools" > "Profiles" menu item.

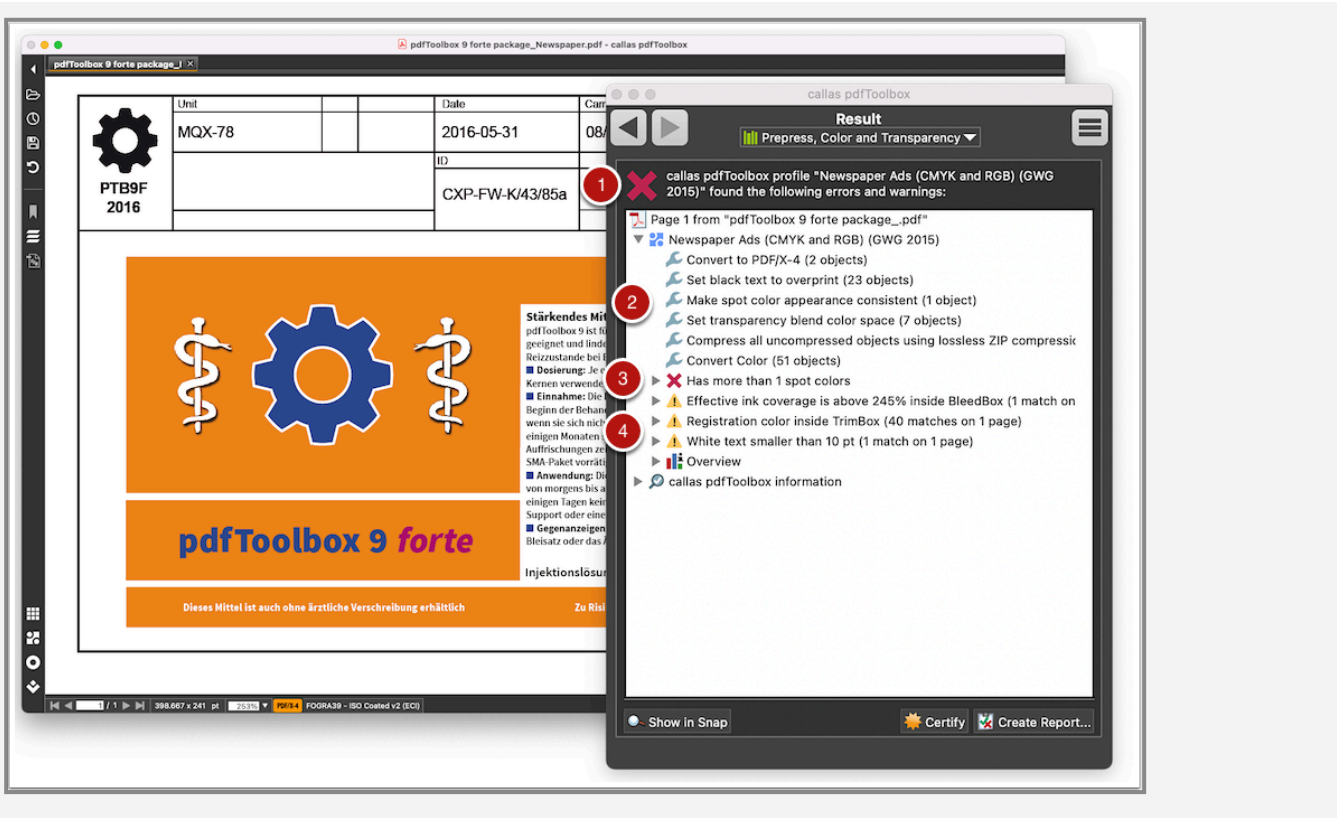


1. Select the Profile you want to use.
2. The "**Analyze**" button only does quality control and will never change the PDF document: in the Profile you have selected, it will only execute the Checks. This can be used to check whether a PDF document is good or not.

3. The "**Analyze and fix**" button also runs the Fixups of the selected Profile in addition to the quality control. The document may be modified in this case.

The Result window – examining processing results

While pdfToolbox Desktop is processing the PDF document, you'll see a progress window that informs you of which Fix-ups have been run or what messages are reported about the PDF document. When processing is done, this progress window becomes the "Result" window you see below.



The Result window shows:

1. The result of processing. In this case a red cross indicating there was at least one error detected.
2. All Fixups performed by the Software are listed. It could be assumed that this list only displays the Fixups that were necessary for the profile execution. This is very often the case. However, there are a small number of Fixups that are always listed in the results window. The reason

for this will be explained in the next section "Fixups that are always listed in the Result window".

3. An error issued by preflight Checks in the Profile. An error is something that likely will be a problem later on. It's something that needs to be fixed.
4. A list of all warnings issued by preflight Checks in the Profile. A warning is an item that may be an issue but doesn't have to be. It's something that needs to be checked.

When pdfToolbox processes a PDF document using a Profile, it will first run all Fixups in the Profile and then run the preflight Checks. If something in the PDF document was wrong, but it could be fixed by a Fixup, it won't appear in the warnings or errors. If a warning or an error is reported, it's because they could not be fixed by the Profile and they need to be dealt with otherwise.

Fixups that are always listed in the Result window

When a profile is applied to a document, all Fixups performed by the software are listed in the Results window. This list often assumes, that it is meaningful in terms of whether the execution of a Fixup was necessary. Of course, this assumption is true in most cases. However, there are a small number of Fixups that are always executed by the software, regardless of whether they are necessary or not. These Fixups are therefore displayed each time in the Results window. For example, single Fixups from the "Fonts" or "XMP Metadata" area may be affected.

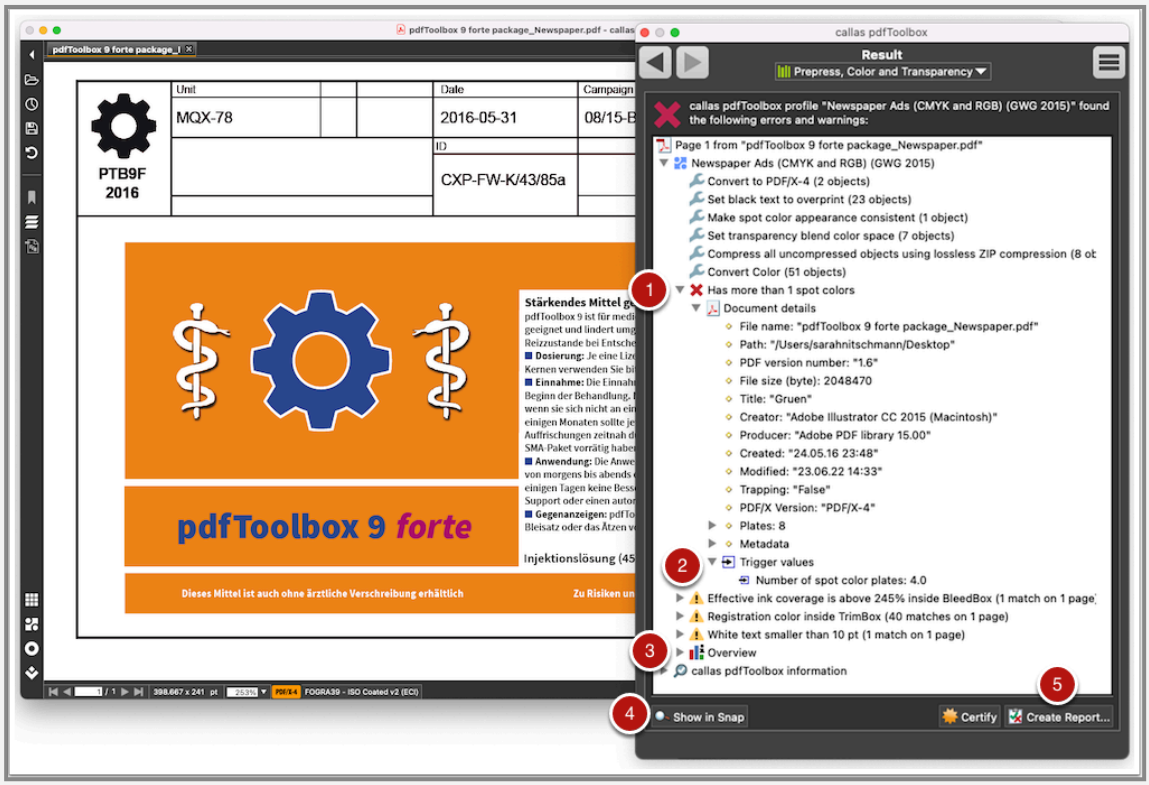
The list of executed Fixups can therefore not always be interpreted as a reliable information about actual changes to the PDF.

The technical background is as follows: As already mentioned in chapter 2.1, an internal analysis is performed before a Fixup is executed, which checks whether a Fixup is necessary. This analysis requires a certain amount of time. To keep the processing time of the profile as short as possible, certain Fixups are always applied to the PDF file immediately, since the previous analysis would take more time than simply executing the Fixup.

To find out, whether the PDF file has been modified by a Fix-up and the result file matches all requirements, a Check can be configured which verifies the result.

Drilling down to what causes warnings and errors

It's often helpful to know exactly where on the page a warning or error occurs.

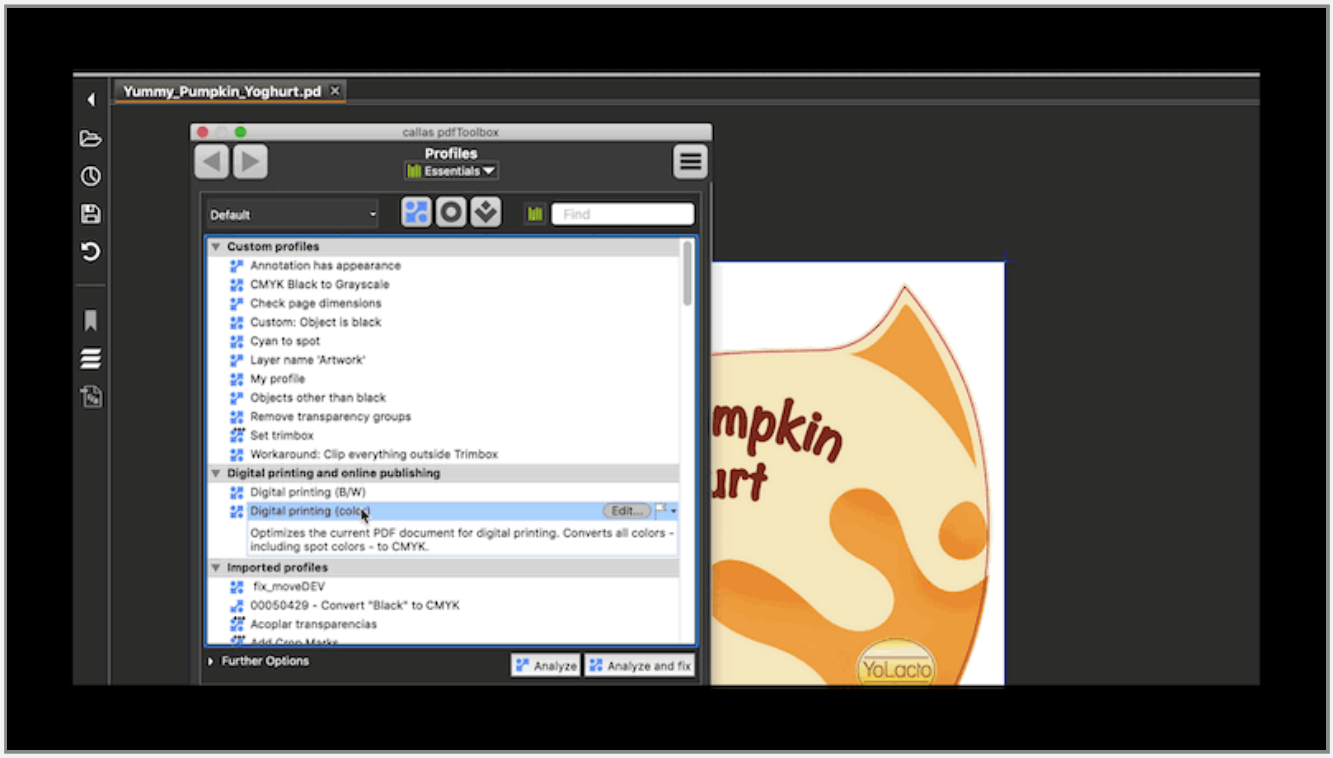


1. Via the triangle next to the warning or error message, further details can be viewed. Different information appears depending on the category.
2. Under "Trigger values" a list of the objects that caused the respective warning or error message is displayed.
3. The Overview is always part of the Results window and can be viewed independently of the Fixups and Checks. It contains detailed information about the PDF document (page geometry, color spaces, fonts, images and much more).
4. You can click the "Show in Snap" button to open a separate window where just the object causing the warning or error is shown.

5. A preflight report can be generated using the "Create Report..." button. You can find more information about this topic here.

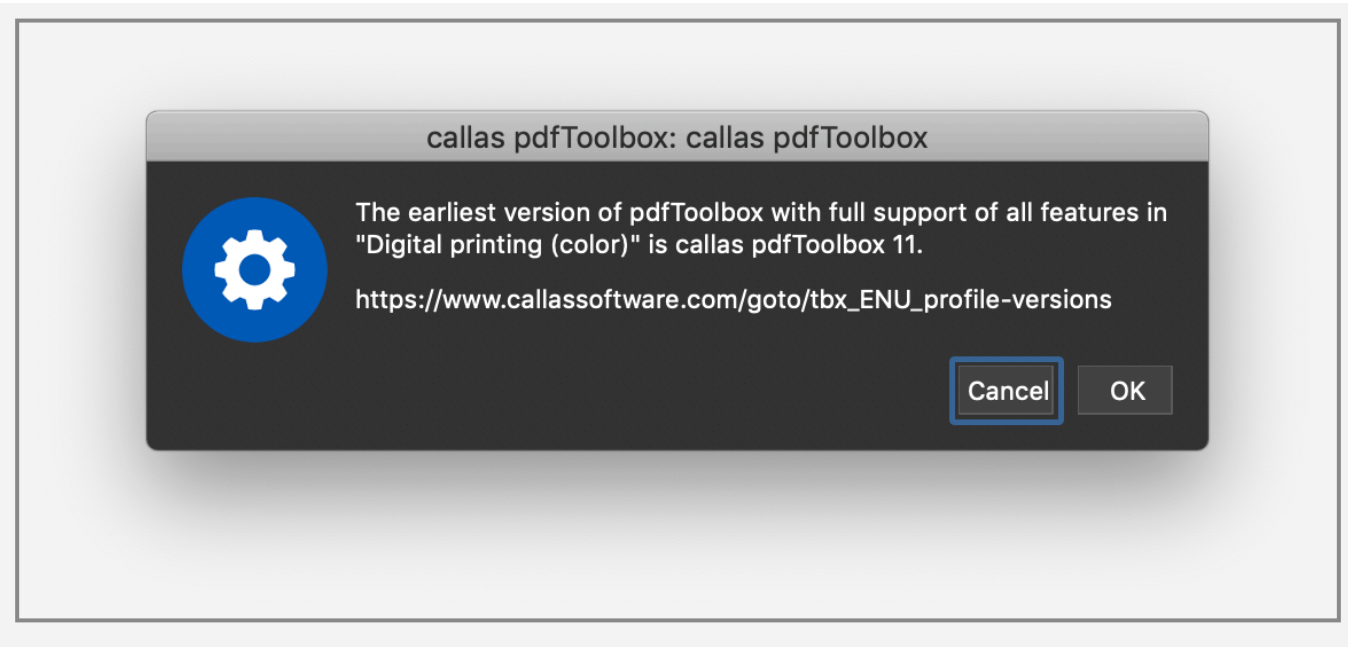
2.9 Export Profiles to a previous pdfToolbox version

pdfToolbox offers support to export Profiles/Checks/Fixups/ Process Plans to previous versions of pdfToolbox.



Steps to export

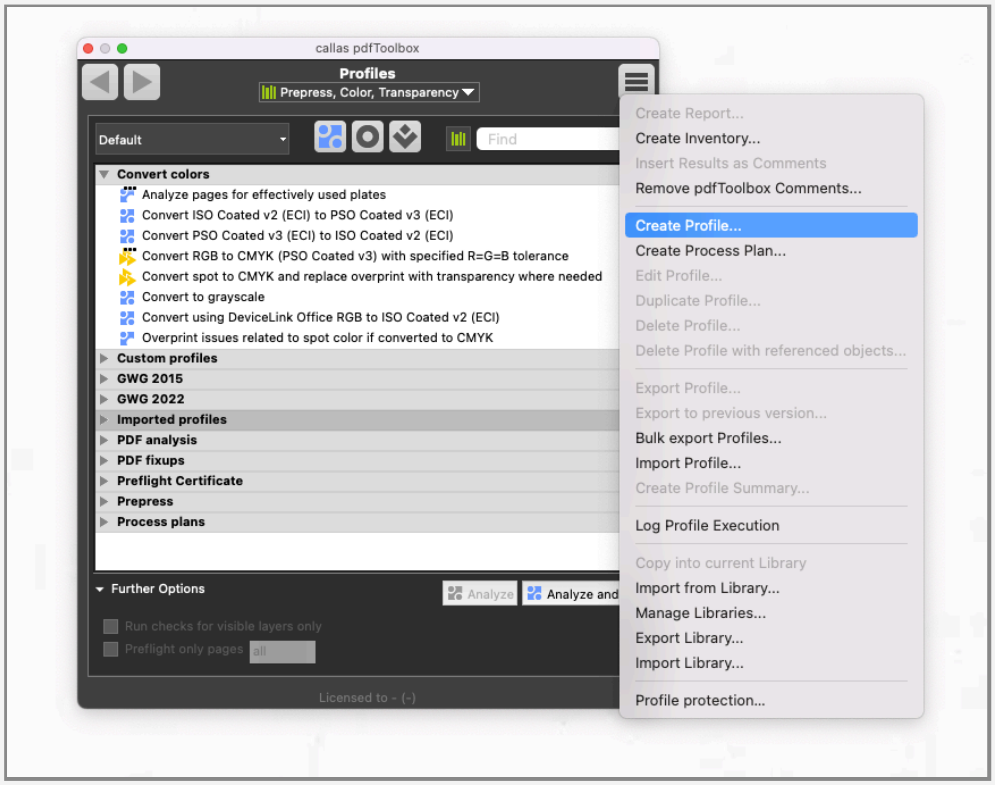
1. To export, select an 'ITEM' which can be a Check, Fixup, Profile or a Process Plan
2. Go to Options --- Export to previous version...
3. A pop-up window opens with the following information:
   - “The earliest version of pdfToolbox with full support of all features in <name of 'ITEM'> is pdfToolbox <version>"
   - And a link to the online manual with information to determine what features in a Profile prevent it to be saved in an earlier version



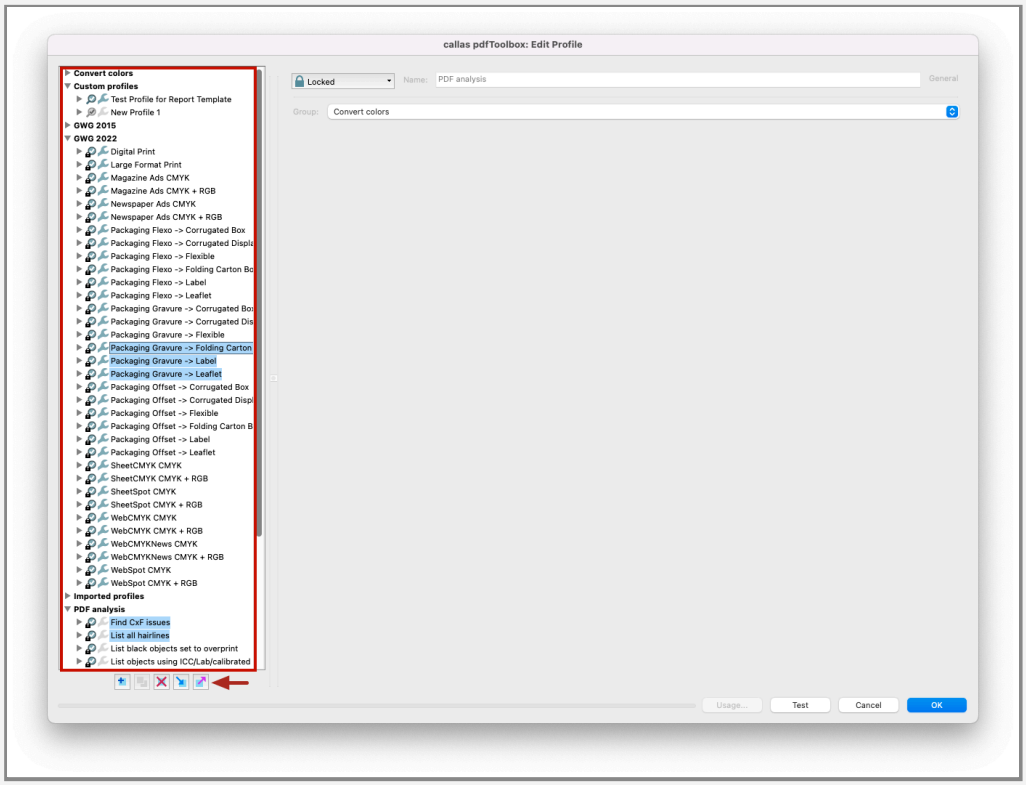
How to export multiple Profiles at once

It is quite easy to export multiple Profiles at once.

1. Just go to the "Edit" dialog of the profiles (or choose "Create Profile")



2. Mark all Profiles, which shall be exported in the left list and click the right icon below the list.

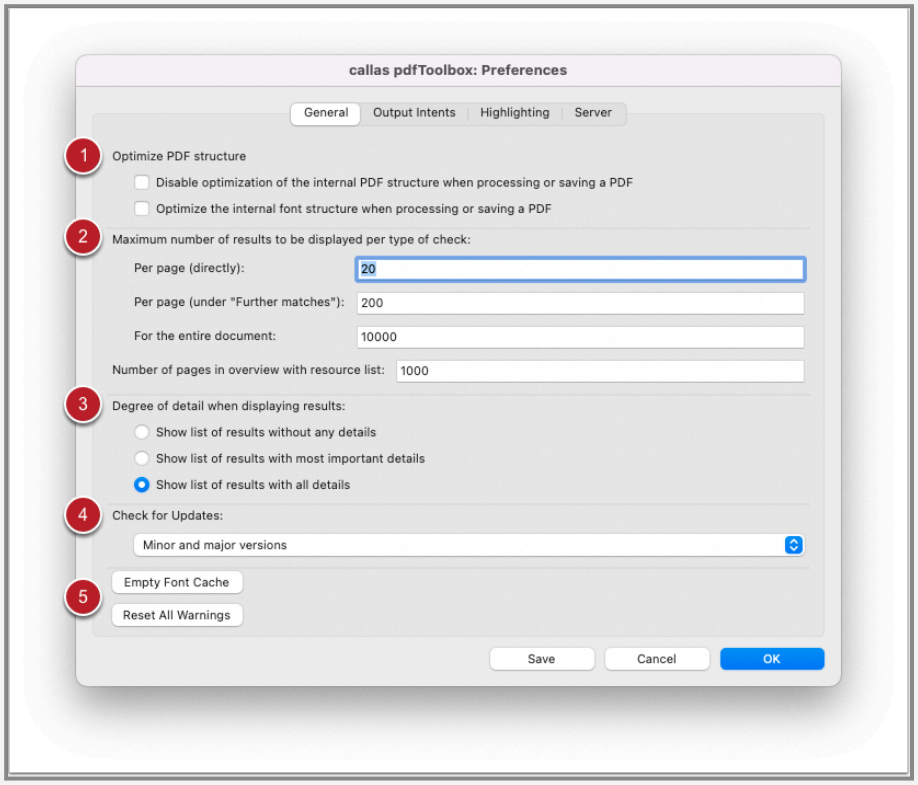


3. Choose a destination folder, where the selected Profiles shall be saved.

2.10 Preferences

The Preferences dialog offers level-of-detail settings for customizing power-tool results, lets you specify how to highlight problems when generating a report, and provides an editor for creating and adjusting output conditions (OutputIntents).

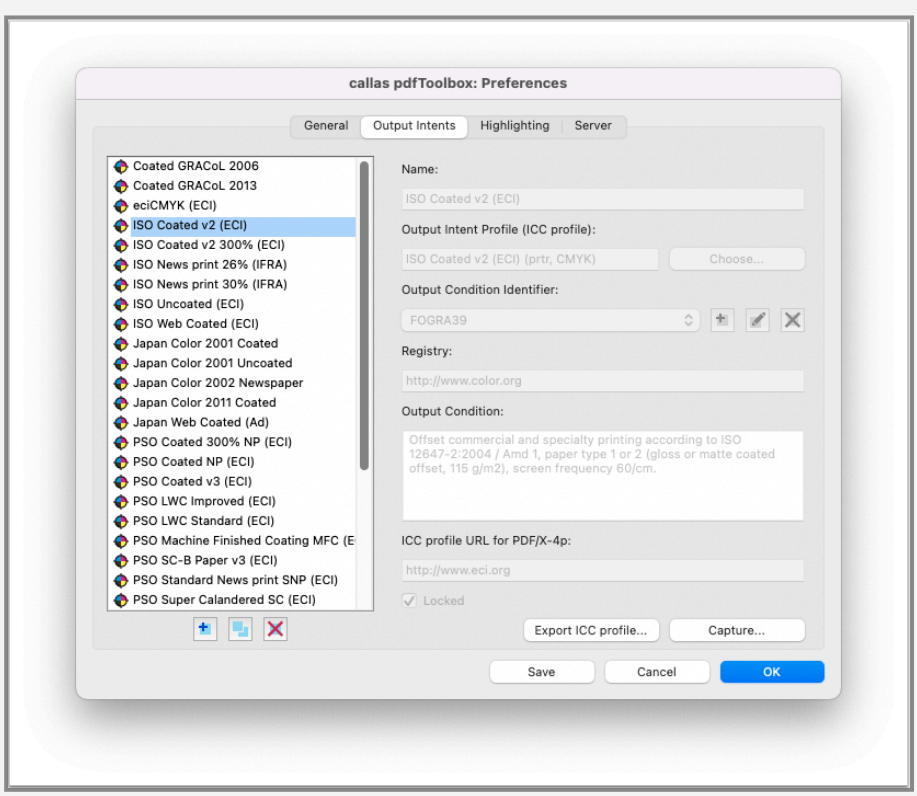
General



1. Here you can choose whether or not to automatically optimise your PDF files when they are being processed.
2. You can limit the maximum number of results to show per page or per document.
3. The level of detail when displaying results can be customized.
4. You can choose whether or not to check for updates.
5. You can also empty the font cache and reset all warnings within pdfToolbox.

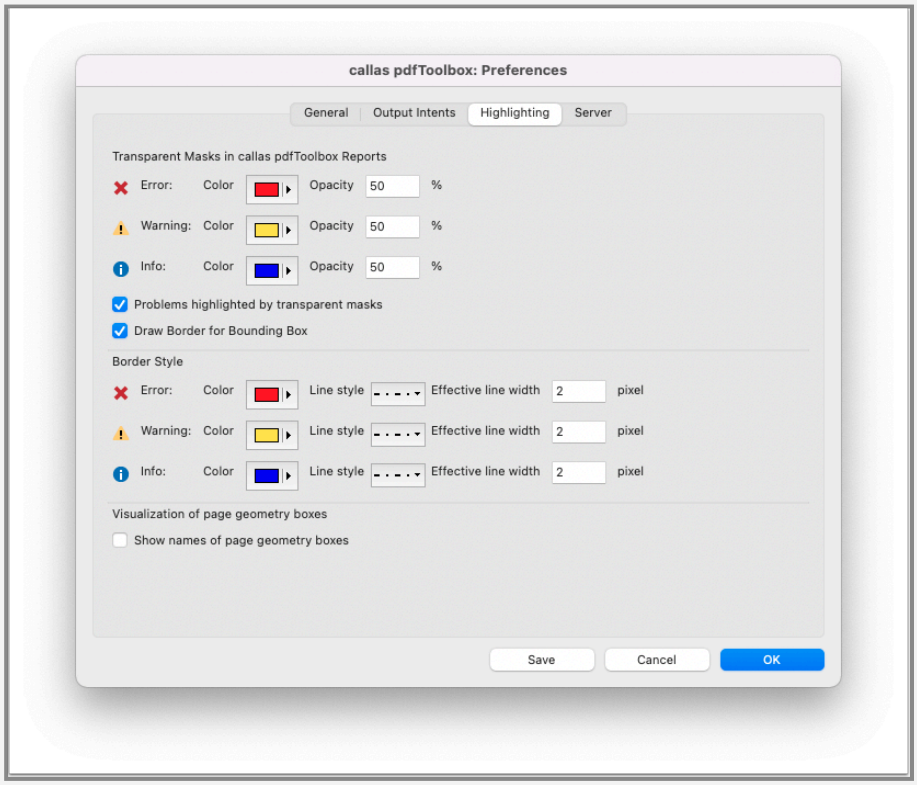
OutputIntents

Here you can create, edit or export OutputIntents (output conditions) as well as extracting ICC profiles.



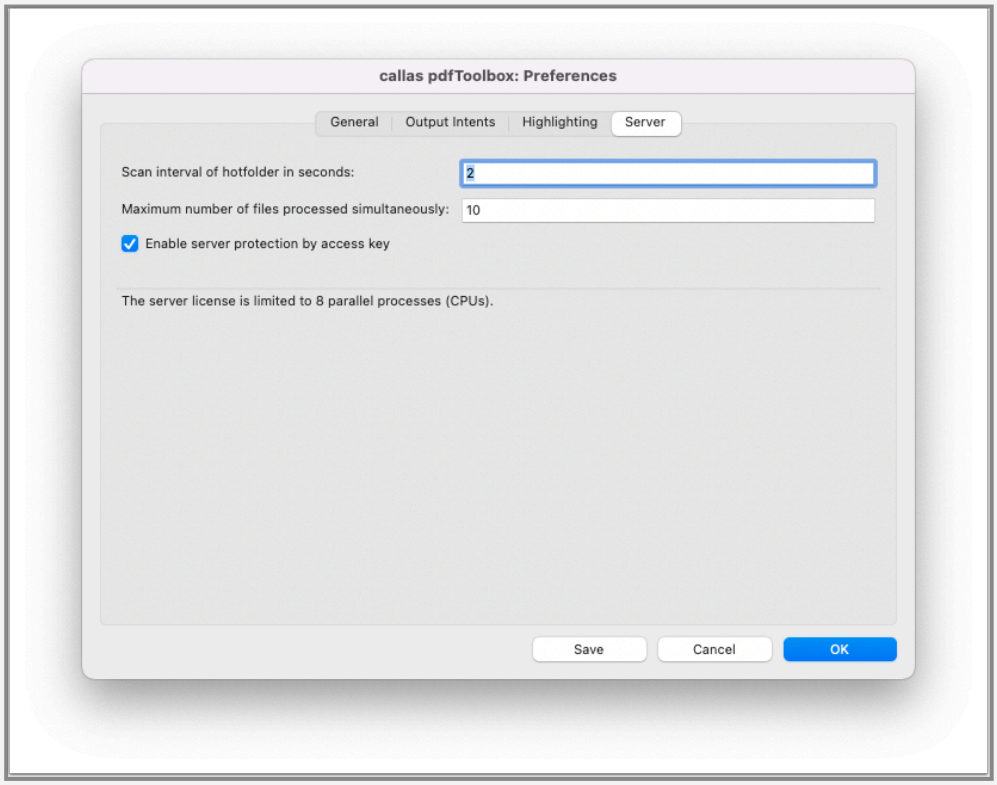
Highlighting

This tab lets you specify the color and style for highlighted objects within a PDF document or the transparent masks in a PDF mask report.



Server

This tab lets you configure the settings for the server version of pdfToolobx. These server settings will only be applied when a server is started from pdfToolbox Desktop, they will not affect the settings of an already running server.

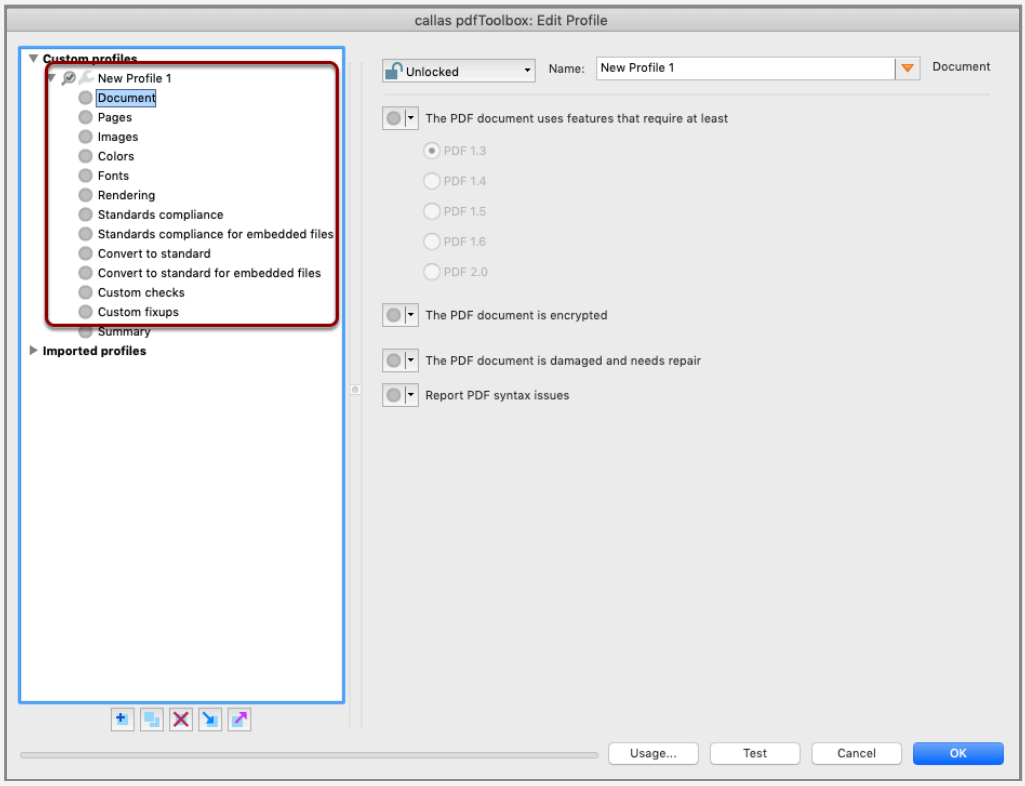

callas pdfToolbox: Preferences
General
Output Intents
Highlighting
Server
Scan interval of hotfolder in seconds:
2
Maximum number of files processed simultaneously:
10
Enable server protection by access key
The server license is limited to 8 parallel processes (CPUs).
Save
Cancel
OK

2.11 Syntax Checks

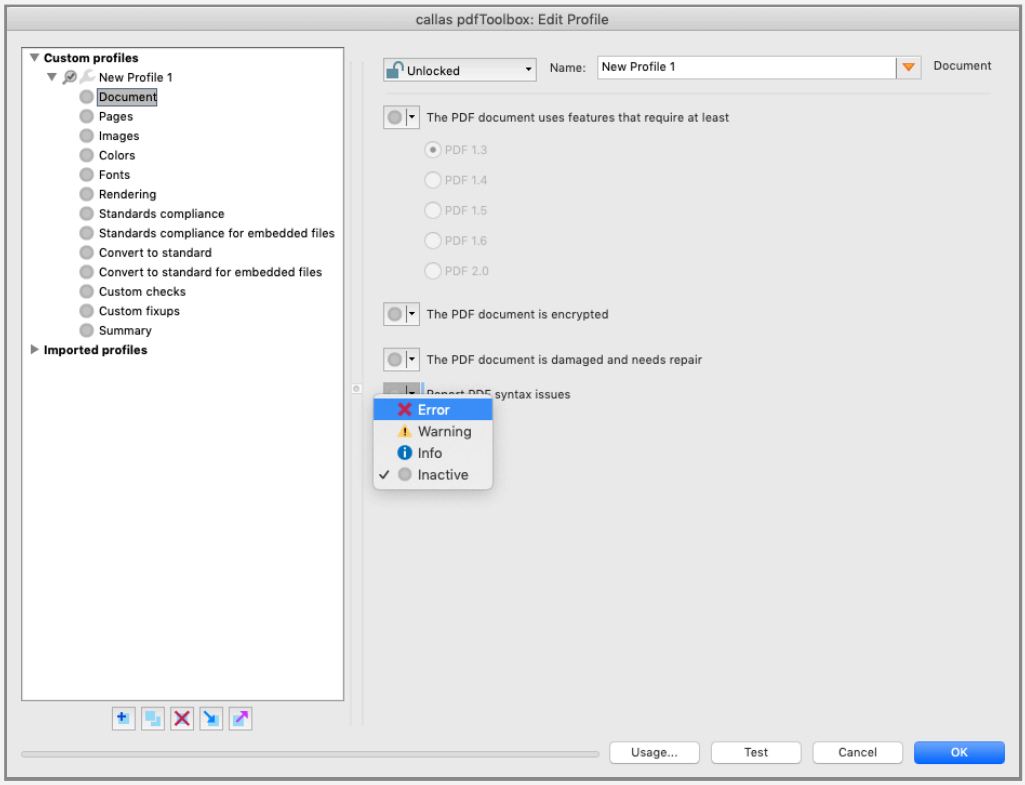
Syntax checks, as the name suggests, indicates any issues with the PDF syntax and acts like some sort of a master Profile to check syntax issues in your PDF. Switching it on to indicate such issues in the form of errors or mere warnings can count as a good practice.

How to enable/disable 'Syntax Checks'

On creating/editing a Profile, you can find a wide range of settings that are available across a number of categories like 'Document', 'Pages', 'Images', 'Colors' etc for a Profile, like shown below.



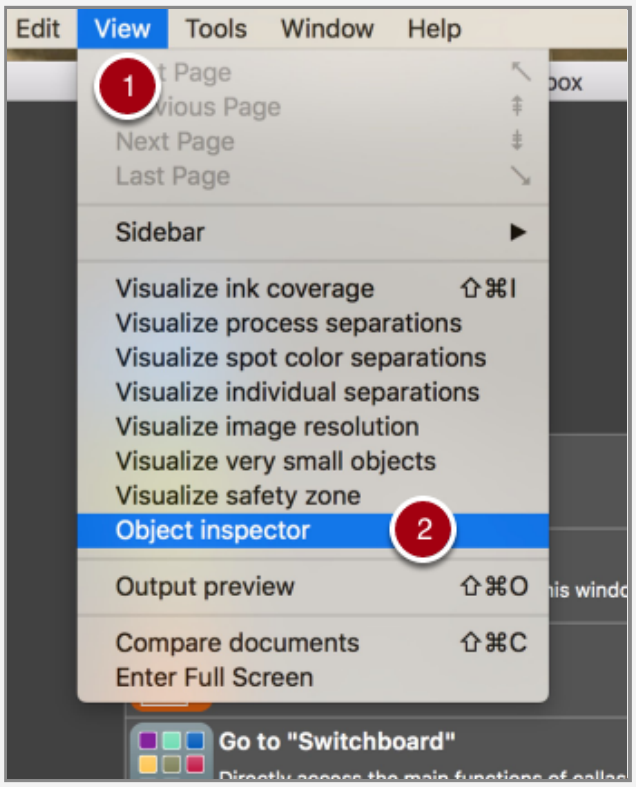
Under 'Document', you can find 'Report PDF syntax issues' amongst other Checks. Simply click on the dropdown next to it to enable/disable it, like shown below.


callas pdfToolbox: Edit Profile
Custom profiles
New Profile 1
Document
Pages
Images
Colors
Fonts
Rendering
Standards compliance
Standards compliance for embedded files
Convert to standard
Convert to standard for embedded files
Custom checks
Custom fixups
Summary
Imported profiles
Unlocked
Name:
New Profile 1
Document
The PDF document uses features that require at least
PDF 1.3
PDF 1.4
PDF 1.5
PDF 1.6
PDF 2.0
The PDF document is encrypted
The PDF document is damaged and needs repair
Error
Warning
Info
Inactive
Usage...
Test
Cancel
OK

2.12 Examining page content

In the visualizer section of pdfToolbox Desktop, the Object Inspector allows identifying page content, both viewing which objects form a PDF page, and what their attributes are. Two additional concepts: "Wireframe viewing" and "Object type filtering" have been implemented here.

Showing the object inspector

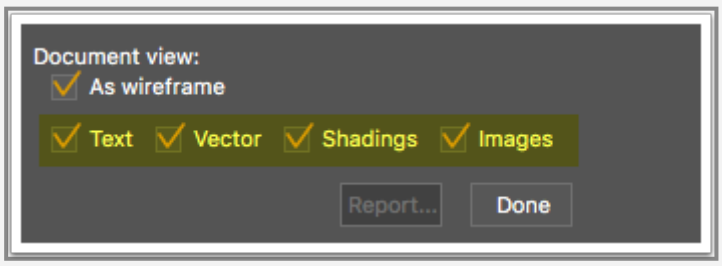


1. Click on the "View" menu
2. Select the "Object inspector" menu item

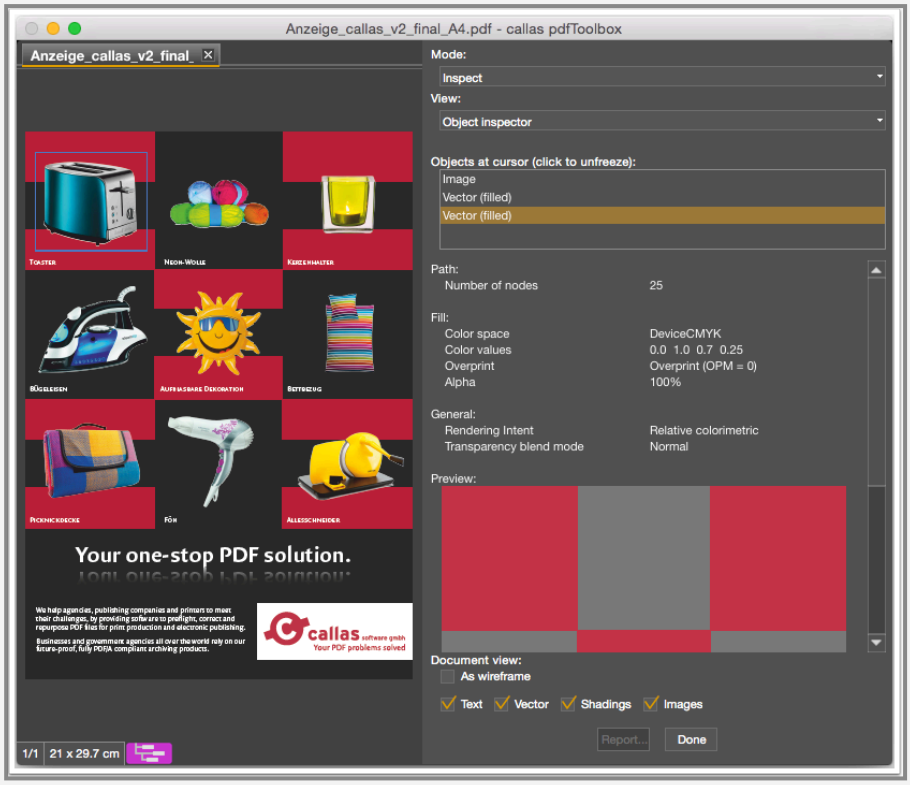
Object type filtering

By default, the object inspector shows all PDF objects on the page. This can be changed by disabling or enabling the

checkboxes at the bottom of the Object inspector area. Deselecting "Text" for example, will cause all text objects on the page to be hidden. This allows examining whether objects are actually text for example, or allows viewing what is behind other objects (and normally hidden from view).



Object properties



Every single object can be selected by clicking on it inside the view on the left hand side. A click will fix the selection on the right hand side. Another click will release it.

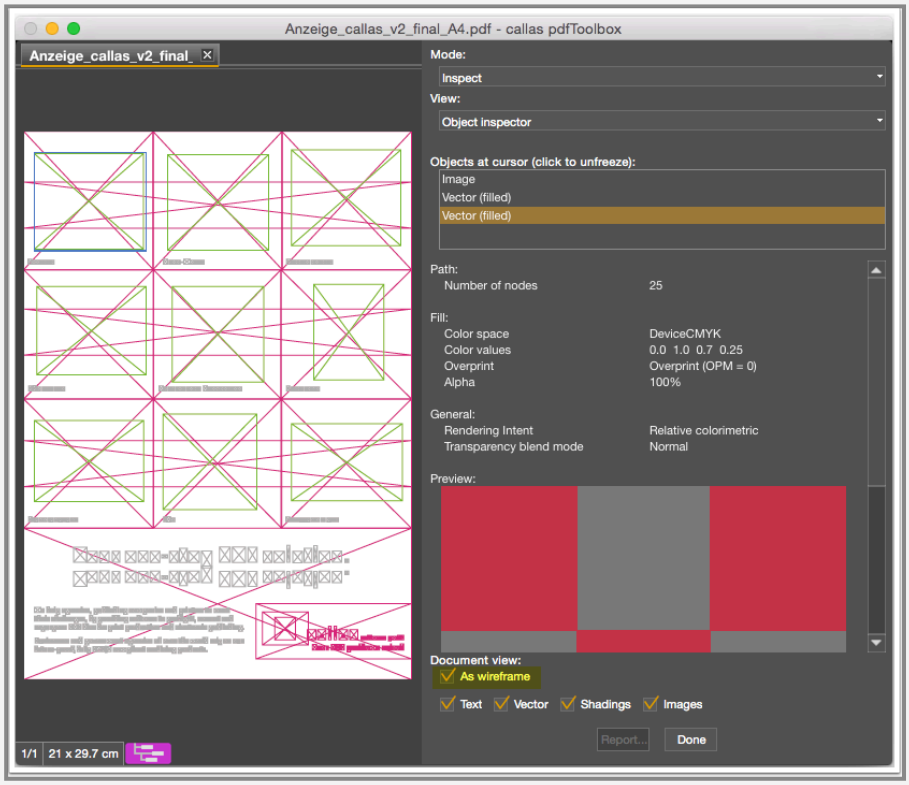
Selected objects are also visible in the preview on the right hand side.

Under "Objects at cursor", you will see a stack of all the objects in the same order as they appear in the PDF.

The topmost object is the topmost in the PDF. You can click at objects below the topmost object to see it's properties.

Wireframe viewing

At the bottom of the Object inspector area, check the "As Wireframe" checkbox, to show the displayed PDF page as wireframe. In this mode, all objects are shown with differently color rectangle outlines. This allows seeing the structure of the page (which objects are on the page, how they are layered etc...).



2.13 Requirements for conversions to PDF

pdfToolbox offers the possibility to convert several file formats to PDF. Some file formats can be converted directly via pdfToolbox. This means, that no external application is required for the conversion of these file types.

If a file format cannot be converted directly via pdfToolbox, pdfToolbox is using the applications in which the files have been created for this conversion.

Supported applications and files

Direct conversion via pdfToolbox

Filetype	Filename	Operating system	Note
PostScript files	.ps	All platforms	
	.eps		
	.prn		
XPS files	.xps	Windows	
CF2 files	.cf2	All platforms	Since pdfToolbox 15.0
HTML files	.html	All platforms	
	.htm		
	.svg		
Image files	.tiff	All platforms	
	.jpeg		
	.png		
	.gif		

Filetype	Filename	Operating system	Note
	.psd		
	.psb		Since pdfToolbox 11.0
	.heic		Since pdfToolbox 16.0

Conversion via external application

Application	Filename	Operating system	Note
Microsoft Word 2007 - 2024	.doc .docx .dot .dotx .dotm .rtf .txt .xml .vcf .ics .wpd	Windows	
Microsoft Excel 2007 - 2024	.xls .xlsx .xlsm .xlt .xltx .xltm	Windows	
Microsoft PowerPoint 2007 - 2024	.ppt .pptx .pps .ppsx .pot .potx .potm .ppa .ppam	Windows	
Microsoft Visio 2007 - 2021	.vsd .vdx .vdw .vss .vsx .vtx .vsdx	Windows	Since pdfToolbox 11.0
Microsoft Project 2010 - 2021	.mpp .mpt .mpd .mpw .mpx	Windows	
Microsoft Publisher 2007 - 2021	.pub	Windows	
LibreOffice Writer 4.x - 7.x	.odt .ott .sxw .stw .doc .rtf .sdw .vor .txt .pdb .xml .psw .docx .docm .dotx .dotm .602 .wpd .hwp	All platforms	
LibreOffice Calc 4.x - 7.x	.ods .ots .sxc .stc .dif .dbf .xls .xlt .sdc .vor .slk .csv .pxl .uos .xlc .xlm .xlw .sdc .xlsb .xlsm .xlsx .xltm	All platforms	

Application	Filename	Operating system	Note
	.xltx .wb2		
LibreOffice Impress 4.x - 7.x	.odp .otp .sxi .sti .ppt .pot .sxd .sda .sdd .vor .uop .odg .cgm .pptm .pptx .potm .potx	All platforms	
LibreOffice Draw 4.x - 7.x	.bmp .emf .pcx .pgm .wmf	All platforms	Since pdfToolbox 10.2

Note: Please note that 'Email to PDF conversion' is supported by callas pdfaPilot. Futher information here.

General

- All Office-applications should be installed with the default options, in order to guarantee that all needed components are available.
- Install all the latest updates.
- Start the application at least once and make sure that the setup wizards have been executed successfully.
- pdfToolbox need to be activated. This means, a valid activation request must have been sent and the software must have been activated with the activation file received afterwards.
- The activation needs to be performed for the user who wants to execute the file conversion.
- On Windows systems, install the Microsoft .NET Framework Version 4 or newer - this is not automatically installed with the regular Windows Update but belongs to the optional updates.
- Please ensure, that the Microsoft Visual C++ Redistributable Packages for Visual Studio (2013, 2017, 2019) were properly installed by the callas installation package. Available here.
- While conversions to PDF are performed, no other users on this machine should use the applications needed for this process.
- In particular the file to be processed should not be opened in the respective application.

- Only one instance of the application that creates PDF must be used at the same time. Therefore in pdfToolbox Server or in Enfocus Switch the number of instances needs to be set to 1.

- Although some file types can be processed with several applications it is recommended to process documents with the application where they were created with.
- Microsoft Office files are not processed via OpenOffice/LibreOffice by default under Windows and MacOS if no Microsoft Office is installed on the system. For Server/CLI, processing of such MS Office files by OpenOffice/LibreOffice can be forced using the `--topdf_forceopenoffice` parameter.

Microsoft applications

- Please make sure that Microsoft Office is activated. In case Microsoft Office runs as an evaluation version, a warning dialog might pop up during conversion.
- Install the latest service pack for Microsoft Office.
- When updating the Microsoft Office Suite it is recommended to deinstall previously installed version before to ensure all registry entries are removed and properly set when the new installation of Office takes place. Especially when installing Microsoft Office 2010 on a system with an existing Microsoft Office 2007 such registry entries seems to remain. See Microsoft knowledge base for details how to uninstall.

Word

Word documents may contain annotations from the function “Track changes”. They are output into the PDF with the setting “Screen”. When outputting with setting “Print” they are not output. In the Desktop version of pdfToolbox the setting “Print” is default so that annotations are not output. Whereas in pdfaPilot Desktop the setting “Screen” is default so that annotations are output. On the Mac the output of annotations can only be omitted when running the english language version of Word. Moreover “Enable access for assistive devices” need to be enabled in “Universal Access” of the “System preferences”.

Excel

- If the contents exceed the “print area”, only the parts covered by this “print area” will be output whilst the other parts will not be output.
- The page format of the output PDF is optimized in order to fit as much content as possible in a well readable way onto a page. For large sheets the content may be spread over several PDF pages.
- To ensure conversion, the print spooler has to be active. This is the case when the Excel preview shows the content. Otherwise the spooler has to be started in the “Services” configuration of the system settings. After doing this, the “Microsoft XPS Document Writer” shows up in the print dialog of Excel.

PowerPoint

It is not recommended to use parallel conversion of PowerPoint files with MS Office, as this can sometimes lead to problems, instead sequential processing works stable.

Mac

- A page range can not be specified, the parameter Page range (pagerange) does not have an influence.
- Images are exported with a resolution of 72 ppi, the parameter Print (topdf_print) does not have an influence.
- If Microsoft Office 2008 and Microsoft Office 2011 are installed the Office application that is set as default for opening Office documents in the system or the Office application that is already open is used to process Office documents.

Dialog handling

- Under Windows dialogs appearing in the Office application during processing can be handled.
- Therefore an empty file named MTPGuiActions.log has to be placed in the user folder here:
- <user>\AppData\Roaming\axaio software\MS Office 12\MadeToPrint\Logfiles\
- If the Office document that opens a dialog is processed now an ID of the dialog is logged in MTPGuiActions.log
- This ID needs to be entered in MTPGuiActions.cfg which can be found in the etc/PDFOfficeTool folder next to the executable of pdfToolbox Desktop or Server

- Within MTPGuiActions.cfg the options of the dialog handling is described

OpenOffice and LibreOffice

- Close all open OpenOffice and LibreOffice instances before processing with pdfToolbox.
- Please keep in mind the known restrictions of OpenOffice.org regarding cases where umlauts from Type1 and OpenType fonts might not be output correctly. You will find more information on the current state of this issue on OpenOffice.org.
- Only LibreOffice or OpenOffice may be installed.
- For Windows only the 32-bit versions of LibreOffice and OpenOffice are supported (before pdfToolbox 11/ pdfEngine 11).
- *From pdfToolbox 11/pdfEngine 11 onwards with the 64bit variant on Windows, the Libre/OpenOffice has to be 64bit as well (and 32bit when the 32bit variant of pdfToolbox/ pdfEngine is installed).*
- For Non-Windows systems (Mac, Linux..) the JDK needs to be installed: https://www.oracle.com/java/technologies/javase-jdk14-downloads.html

Mac

Please verify that OpenOffice is installed inside the default installation path (e.g. ”/Applications/OpenOffice.org” on Mac OS X systems).

If it is not, please re-install it to this location.

On Mac the following default installation/location are supported (in this order - first wins):

- /Applications/LibreOffice.app
- /Applications/OpenOffice.app
- /Applications/OpenOffice.org.app

Linux

- Support for Office Conversion has been tested thoroughly on Ubuntu 14.04 LTS (Server) with the default installation of LibreOffice and also with an OpenOffice installation.
- On Debian based Linux distributions (like Ubuntu or Mint), the required LibreOffice packages are installed via

the package manager or via the command line as shown below:
  - sudo aptitude install libreoffice
  - (please substitute 'libreoffice' by 'openoffice' if you prefer OpenOffice)

- On other linux distributions or with other OS versions it might be required to search for packages with different names. E.g install everything found with the following search terms:
  - libreoffice (or openoffice)
  - uno
  - ure
  - headless

- Some distributions, like e.g. Debian 10 ("Buster") may require to install the additional "libreoffice-java-common" package:
  - `sudo apt install libreoffice-java-common`

iWork '09 Pages

- A page range can not be specified as it is also via manual PDF export, the parameter Page range (pagerange) does not have an influence.
- Images are exported with a resolution of 72 ppi as it is also via manual PDF export, the parameter Print (topdf_print) does not have an influence.
- Documents of iWork '08 Pages can not be processed with pdfToolbox Server application because of a changed file format (file packages of Pages version 3 and earlier).

PostScript and EPS files

- ICC profiles referenced in a PDF settings file (.joboptions) need to be copied into the operating system folder for ICC profiles, e.g.:
  - Windows:
    C:\Windows\system32\spool\drivers\color
  - MacOS:
    /MacOS HD/Library/ColorSync/Profiles

- The application will also look into the following folders for ICC profiles:
  - /Library/Application Support/Adobe/Color/Profiles/Recommended
  - /Library/Application Support/Adobe/Color/Profiles
  - /System/Library/ColorSync/Profiles
- Alternatively, you can put ICC-files for PostScript to PDF in the subfolder of the application:
  .../etc/PDFPSTool/ICCProfiles
- A Color settings file (.csf) that is referenced in the PDF settings file is not necessary for the processing.

Usage when running as a service

Sometimes, problems may occur when running pdfToolbox as a service, especially in scenarios involving automation without direct user interaction.

In general, it is recommended to grant the respective service user administrator privileges. If this level of rights cannot be set due to internal regulations, some additional settings within the operating system are recommended (the following workaround has been successful in most cases):

Create the following folders (with appropriate permissions) on your system:

For 64-bit machines	
C:\Windows\Temp	Modify
C:\Windows\syswow64\config	Read
C:\Windows\syswow64\config\systemprofile	Read
C:\Windows\syswow64\config\systemprofile\AppData	Modify
C:\Windows\syswow64\config\systemprofile\Desktop	Modify (Create it, if it does not exist)
C:\Windows\syswow64\config\systemprofile\	Modify (Create it, if it does not exist)

For 64-bit machines	
AppData\Local\Microsoft\Windows\ INetCache	

For 32-bit machines	
C:\Windows\Temp	Modify
C:\Windows\system32\config	Read
C:\Windows\system32\config\systemprofile	Read
C:\Windows\system32\config\systemprofile\AppData	Modify
C:\Windows\system32\config\systemprofile\ Desktop	Modify (Create it, if it does not exist)
C:\Windows\system32\config\systemprofile\ AppData\Local\Microsoft\Windows\ INetCache	Modify (Create it, if it does not exist)

Please NOTE:

- For smooth operation, the folders named above need to be restored after a Windows Update.
- For the Office/Excel conversion in a service setting, it is necessary to set up a default printer, especially when working with the "System" account.
- For Microsoft Excel, Microsoft Word and Microsoft PowerPoint the **DCOM Identity** "The launching user" must be selected. For Microsoft Visio the DCOM Identity "The interactive user" needs to be selected.

Office conversion

Additional settings

- Set the 32-bit folder preferences (detailed above) in addition to the 64-bit preferences on 64-bit systems running 64-bit versions of Microsoft Office
- Set the default printer to XPS Document Writer

DCOM settings

- Launch DCOMCNFG by using:C:\WINDOWS\SysWOW64> mmc comexp.msc /32
- Go to Computers > MyComputer > DCOM Config.
- Right-click the application that you want to automate.The application names are listed in the table below:

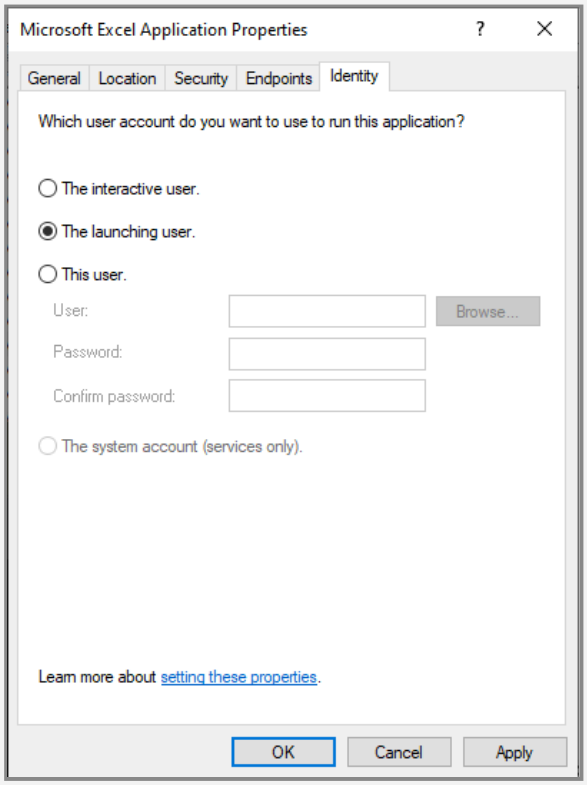
Application	DCOM Name
Microsoft Access 2007/2010/2013/2016	Microsoft Access Application
Microsoft Excel 2007/2010/2013/2016	Microsoft Excel Application
Microsoft Office Word 2007	Microsoft Office Word 97 - 2003 Document
Microsoft Word 2010/2013/2016	Microsoft Word 97 - 2003 Document

- On some systems Microsoft Word is not displayed and you will have to use {00020906-0000-0000-C000-000000000046} instead.
- Click Properties to open the property dialog box for this application.
- Verify Identity and Security tabs

Note

For Microsoft Excel, Microsoft Word and Microsoft PowerPoint the DCOM Identity "The launching user" must be selected.

For Microsoft Visio the DCOM Identity "The interactive user" needs to be selected.



Slow conversion on systems without Internet connection

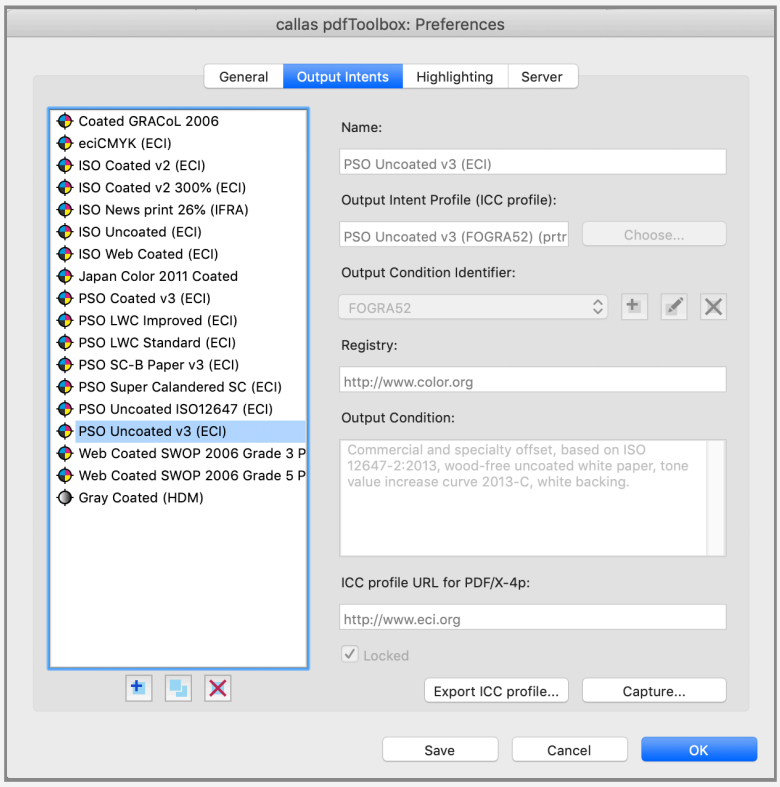
Some customers experienced long processing times on maschines without an Internet connection, especially when using MS Office 365 or MS Office 2016/2019.
The following registry key did the trick for them:
`[HKEY_CURRENT_USER\Software\Microsoft\Office\Common\ClientTelemetry] "DisableTelemetry"=dword:00000001` This might be related to:
https://docs.microsoft.com/en-US/DeployOffice/compat/manage-the-privacy-of-data-monitored-by-telemetry-in-office
We haven't found if there are official settings available - therefore we recommend an intensive testing on such systems before this key is used in a production environment as we can not estimate if there are any side effects.

Still problems when converting?

In case you are having trouble with 'Convert to PDF', please contact our support team and simply fill out this form.

2.14 Add your own Output Intents

Numerous predefined Output Intents are already delivered with pdfToolbox. But if certain output intents are missing or if you need further Output Intents that are not yet available, you can simply define and create them using Preferences settings.



To add your own Output Intent, simply click on the small "+" symbol below the list on the left.

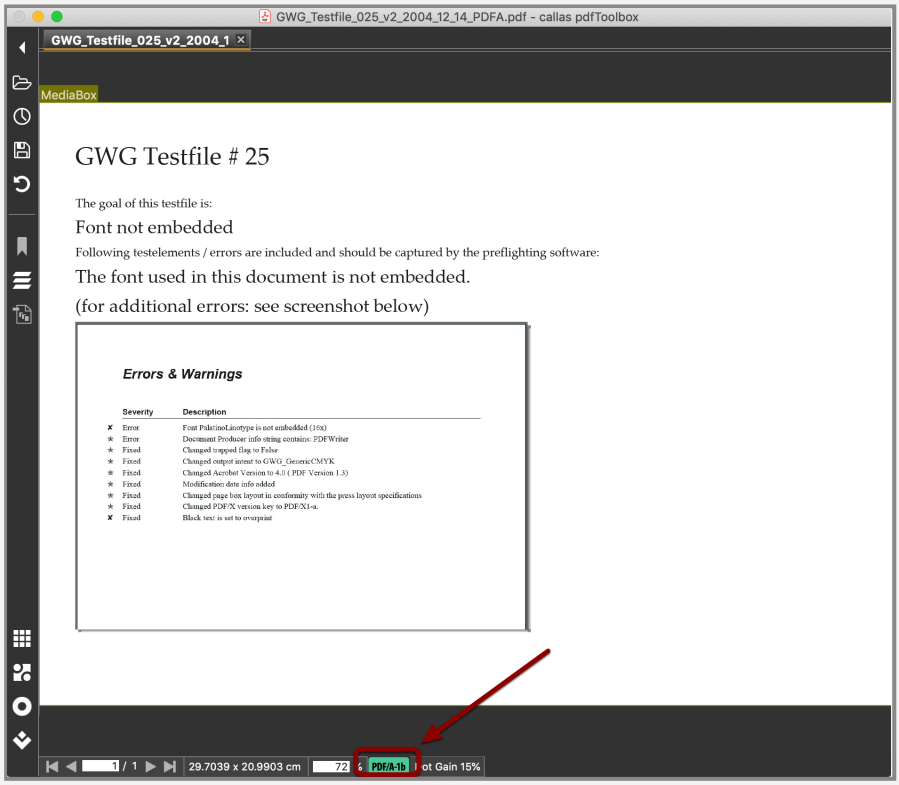
Then assign a name, select the ICC target profile and select the "Registered Name (OutputConditionIdentifier). If the latter is not yet available, you can create it yourself using the "+" symbol. Make sure that the Registration URL and the definition of the output condition are entered correctly. These are embedded as meta information in the PDF files later provided with this Output Intent.

If you now create a new Fixup (Fixup type "Embed Output Intent on page level"), the new Output Intent defined in the last step should be selectable in the list.

Output_Intent_PSO_Uncoated_v3_(ECI)_embedded.kfpx

2.15 Checking the ISO standards using callas Desktop products

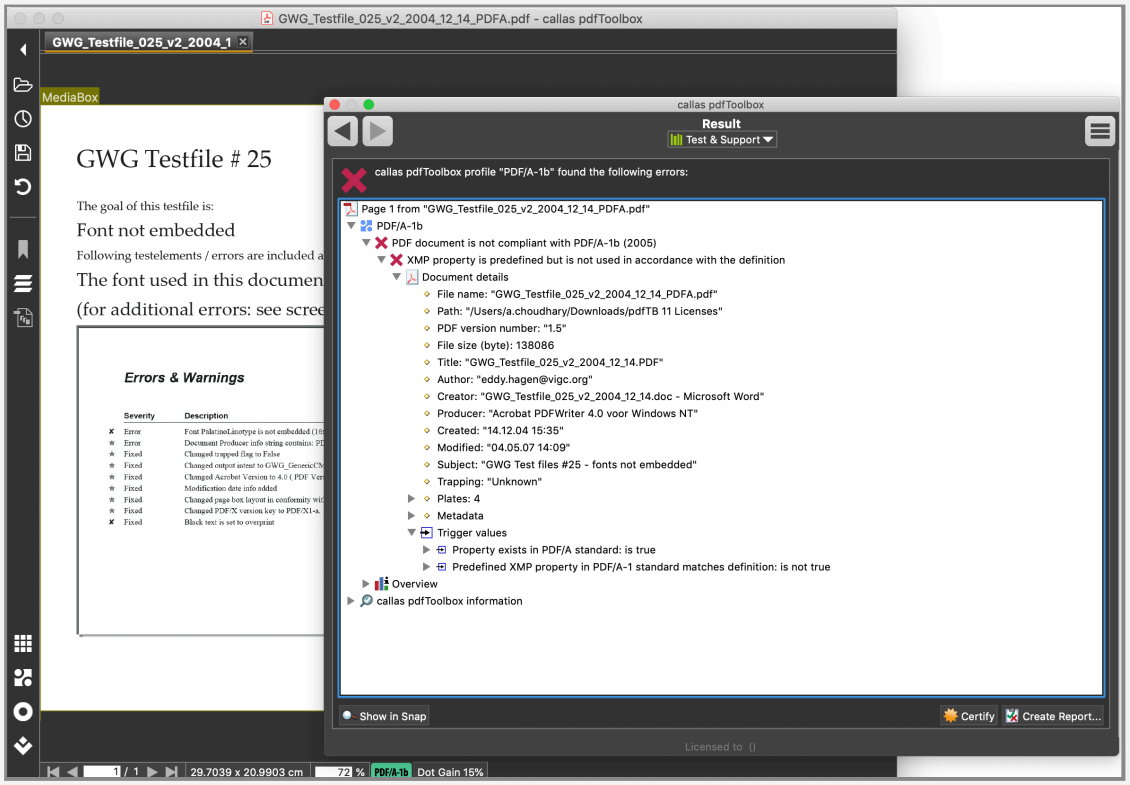
This article explains how you can check if a PDF file is ISO standard compliant or not.



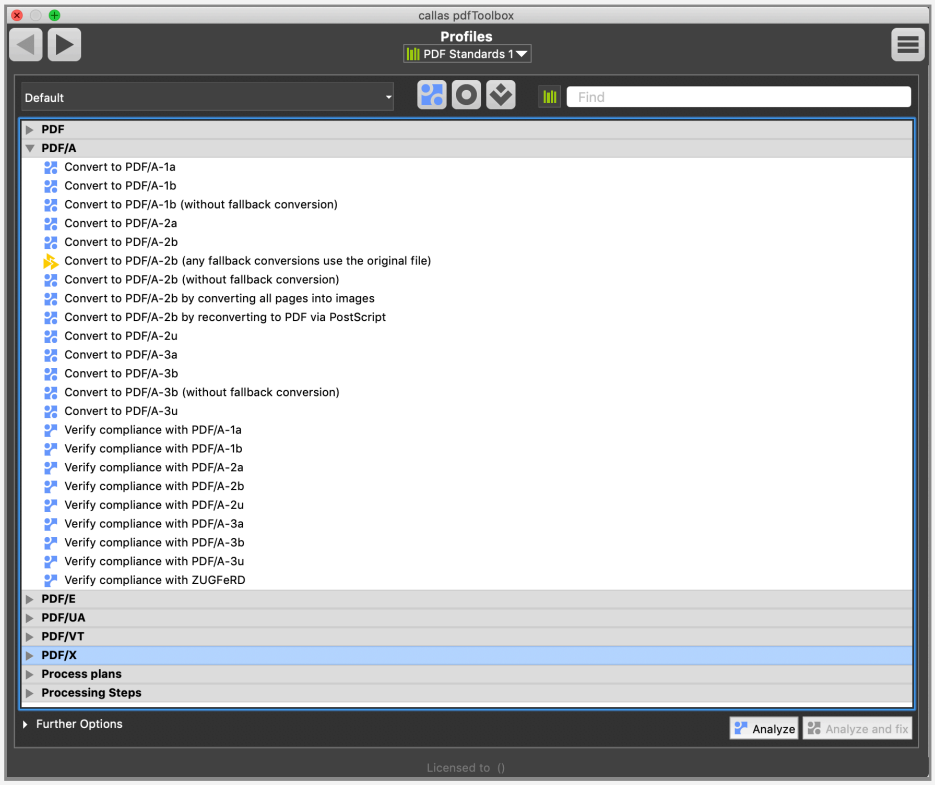
Upon opening a file in a callas Desktop product (pdfToolbox/ pdfaPilot), the application will notify you at the bottom if the document is marked with a standard or not (marked as a rectangular selection in the screenshot above).

In the example above, the PDF document is marked as PDF/ A1-b compliant. Upon clicking the coloured (green in this case but the color depends on the type of standard) button, a result window will pop up showing the compliance or non compliance result, like in the image below.

A PDF file with multiple standard compliance will be shown with multiple buttons, coloured individually based on the standard.



Another way to check the standards compliance is via the predefined Profiles. Under the 'PDF Standards' library, the desktop application comes with different Profiles for checking compliance with or even converting to a standard (PDF/A, PDF/X, PDF/UA etc).


callas pdfToolbox
Profiles
PDF Standards 1
Default
Find
PDF
PDF/A
Convert to PDF/A-1a
Convert to PDF/A-1b
Convert to PDF/A-1b (without fallback conversion)
Convert to PDF/A-2a
Convert to PDF/A-2b
Convert to PDF/A-2b (any fallback conversions use the original file)
Convert to PDF/A-2b (without fallback conversion)
Convert to PDF/A-2b by converting all pages into images
Convert to PDF/A-2b by reconverting to PDF via PostScript
Convert to PDF/A-2u
Convert to PDF/A-3a
Convert to PDF/A-3b
Convert to PDF/A-3b (without fallback conversion)
Convert to PDF/A-3u
Verify compliance with PDF/A-1a
Verify compliance with PDF/A-1b
Verify compliance with PDF/A-2a
Verify compliance with PDF/A-2b
Verify compliance with PDF/A-2u
Verify compliance with PDF/A-3a
Verify compliance with PDF/A-3b
Verify compliance with PDF/A-3u
Verify compliance with ZUGFeRD
PDF/E
PDF/UA
PDF/VT
PDF/X
Process plans
Processing Steps
Further Options
Analyze
Analyze and fix
Licensed to ()

2.16 Compatibility between pdfToolbox and Acrobat

There is no 100% compatibility between Preflight and pdfToolbox as the underlying engines are different in some areas. Please find the differences overview below:

Fixups existing only in Preflight (Acrobat)

All Preflight Fixups that are based on Acrobat's PDF Optimizer are listed below:

- Repair damaged document
- Discard all comments and form fields
- Discard all external cross references
- Discard all form submission, import and reset actions
- Discard all JavaScript actions
- Discard document structure
- Discard embedded thumbnails
- Discard all alternate images
- Detect and merge image fragments
- Remove unreferenced named destinations
- Remove invalid links
- Normalize spot color appearance
- Stamp page

Features existing only in pdfToolbox

pdfToolbox is updated more frequently and therefore more Fixups are available. Some Fixups also have more settings as in the Preflight version, such as "Convert colors", "Flatten transparency" etc. Below is the list of all such Fixups (linked to their respective documentation articles):

- Test Mode
- Context aware object detection (Sifter)
- JavaScript Variables
- Quick Check
- Quick Fix
- Process Plans
- Actions
- Create and apply shapes

- Place Content / pdfChip
- Place Barcode
- Create appearances for annotations
- Custom Template Reports
- Advanced color conversion policies
- DeviceLink profiles
- N channel color profile
- Embed missing Fonts
- Profile/Library Comparison
- Create and apply Shapes
- Variable Data Print (VDP)
- JSON report
- Create invisible text via OCR

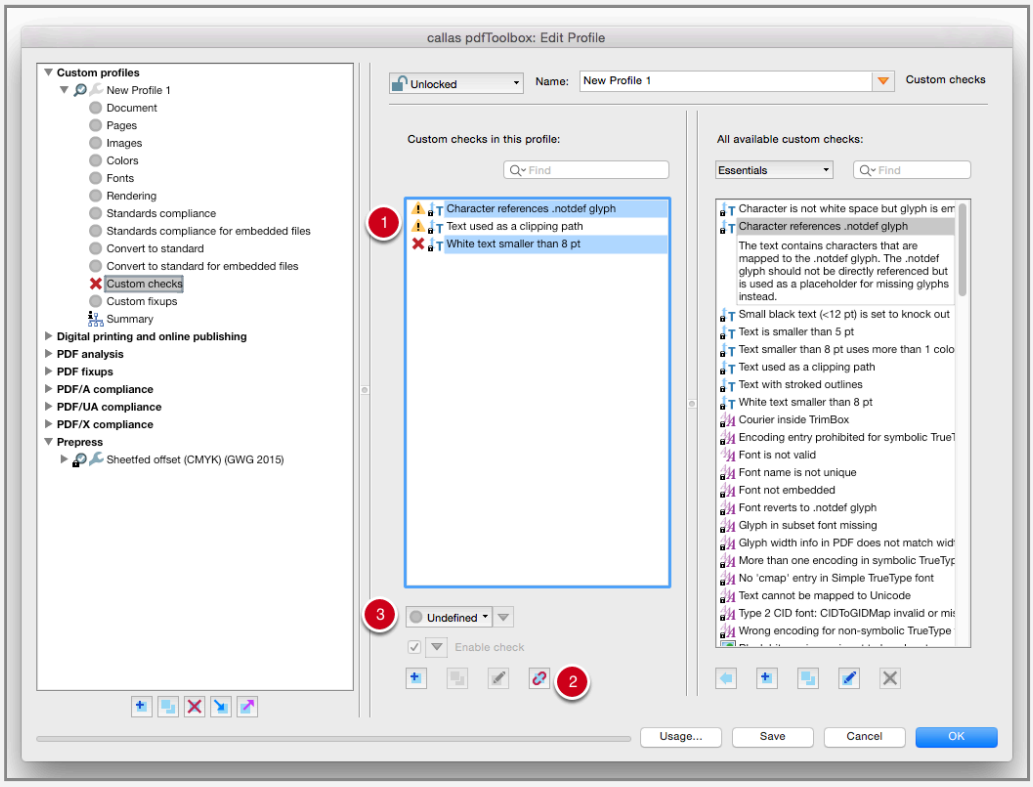
Fallbacks during execution

Internally, "Embed fonts" in Acrobat is automatically assigned to pdfToolbox's "Embed missing fonts". The same happens with Preflight's "Convert to spot color" which will be assigned automatically to pdfToolbox's "Map colors".

2.17 Deleting multiple Fixups or Checks at once

When you edit a profile, you can select multiple Fixups or Checks, deleting them all at once or adding more.

Selecting multiple Fixups or Checks in the “Edit Profile” window



When **setting up or modifying a profile**, you can select and then delete (or change, or add) multiple Checks or Fixups at once.

1. To do so, simply select the desired Checks/Fixups in the desired column by pressing and holding **Shift** (for consecutive items) or **Ctrl** (for non-consecutive items).
2. You can then **Add / Delete** the selected items.

3. When selecting multiple Checks at once, you can also **change** the **message type** (Info, Warning, Error) for all of them at once.

2.18 Automatic optimization of PDFs

PDF files processed by pdfToolbox are automatically optimized when changes have been applied (e.g. via a Fixup or an Action) and file is saved. These optimizations are only applied when the PDF is saved - so this is not the case when an analysis-only processing is performed.
Even a normal re-save of a PDF will trigger the optimization.

All variants (Standalone, Server/CLI and SDK) of pdfToolbox are doing the following optimizations by default:

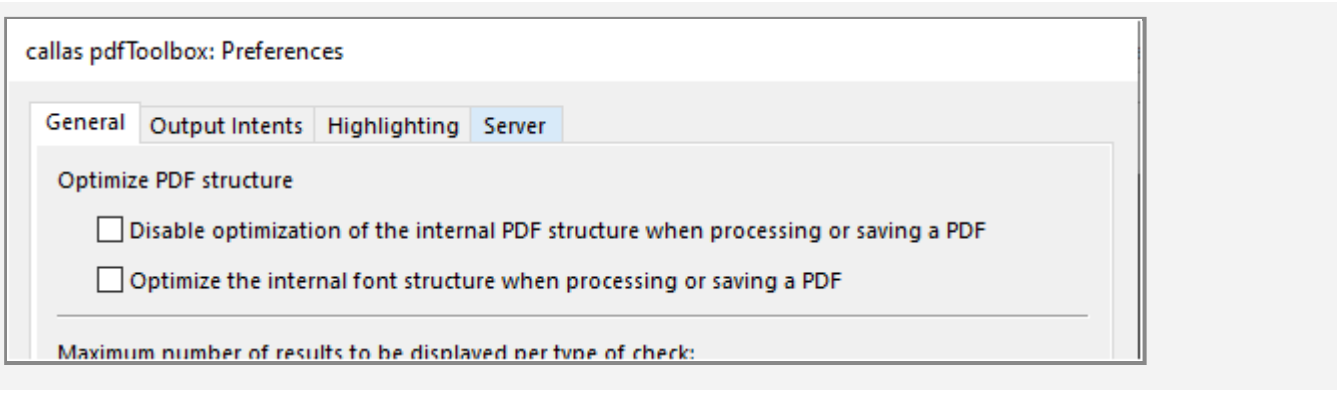
- Remove unreferenced objects
- Optimize content stream (generating substreams that can be shared)
- Merge identical forms and images (determined by an MD5 hash of their contents)

It is also possible to:

- Merge identical fonts and encodings
  and linearize the PDF for optimized web display
- Disable the optimization completely

Adjust optimization in pdfToolbox Desktop

The way a PDF is going to be optimized in Standalone can be adjusted in the preferences:



Adjust optimization in pdfToolbox Server/CLI

- The additional merge of identical font descriptors and encodings can be requested by the additional option `--`

`optimizepdf` within a CLI command (as long as this command creates an output PDF, e.g. when running Profiles).

- Disable the optimization completely can be requested by the additional option `--nooptimization` within a CLI command (as long as this command creates an output PDF).

When using the hotfolder processing in Server, these options have to be set a an additional CLI parameters within each job.

For obvious reasons, the 2 options can not be combined.

The previously available action `--optimizepdf` has been deprecated, but can easily be substituted by:
`--topdf --optimizepdf`
as --optimizepdf has become a general file processing option on CLI.

Optimization in pdfToolbox 11 and older versions

In pdfToolbox 11 and previous versions, the automatic optimization also optimizes the font structure by default.

2.19 Keyboard shortcuts and additional Hotkeys

For some functionality callas pdfToolbox offers the possibility to use keyboard shortcuts.

callas pdfToolbox

Function	Shortcut	
	Mac	Windows
callas pdfToolbox Preferences	Cmd+,	
Hide callas pdfToolbox	Cmd+H	
Hide Others	Alt+Cmd+H	
Quit callas pdfToolbox	Cmd+Q	

File

Function	Shortcut	
	Mac	Windows
Open	Cmd+O	Ctrl+O
Save	Cmd+S	Ctrl+S
Save as	Cmd+Shift+S	Ctrl+Shift+S
Revert	Cmd+R	Ctrl+R
Close	Cmd+W	Ctrl+W
Exit	Cmd+Q	Alt+F4

Edit

Function	Shortcut	
	Mac	Windows
Cut	Cmd+X	Ctrl+X
Copy	Cmd+C	Ctrl+C
Paste	Cmd+V	Ctrl+V
Select All	Cmd+A	Ctrl+A

View

Function	Shortcut	
	Mac	Windows
First Page	<-	Home
Previous Page	↟	Page up
Next Page	↡	Page down
Last Page	->	End
Visualize ink coverage	Shift+Cmd+I	Ctrl+Shift+I
Output preview	Shift+Cmd+O	Ctrl+Shift+O
Compare documents	Shift+Cmd+C	Ctrl+Shift+C

Tools

Function	Shortcut	
	Mac	Windows
Switchboard	Cmd+2	Ctrl+2

Function	Shortcut	
Profiles	Cmd+3	Ctrl+3
Checks	Cmd+4	Ctrl+4
Fixups	Cmd+5	Ctrl+5
Organize pages	Shift+Cmd+M	Ctrl+Shift+M
Explore Metadata	Cmd+6	Ctrl+6
Explore Layers	Cmd+7	Ctrl+7
Explore PDF	Cmd+8	Ctrl+8
Explore Fonts	Cmd+9	Ctrl+9
Server	Cmd+0	Ctrl+0
Server Checkpoint Files	Shift+Cmd+0	Ctrl+Shift+0
View in default PDF application	Cmd+D	Ctrl+D

Window

Function	Shortcut	
	Mac	Windows
Loupe	Cmd+L	Ctrl+L
Show next document	^⇥	Ctrl+TAB
Show previous document	^Shift⇥	Ctrl+Shift+TAB

Help

Function	Shortcut	
	Mac	Windows
callas pdfToolbox Help	F1	F1

Additional hotkeys help in pdfToolbox 12

Function	Shortcut	
Profile window		
	Mac	Windows
Opens the Check/Fixup/Profile in Test mode	Option + Edit button	Alt + Edit button
Opens the Check/Fixup/Profile in Test mode with 'Log Profile execution'	Option + Shift + Click Edit	Alt + Shift + Edit button
Suppress “Ask at runtime” dialog and use default values (also works in Test mode)	Cmd + Execute Profile	Alt Gr + Execute Profile
Add a temporary variable to enable a Fixup in a Profile (with default value **true**). This allows to switch off selected Fixups without modifying the profile beforehand.	Option + Analyze/Analyze and Fix button	Alt + Analyze/ Analyze and Fix button
Add a temporary variable to enable a Fixup in a Profile (with default value **false**). This allows to switch on selected fixups without modifying the profile beforehand.	Option + Shift + Analyze/ Analyze and Fix	Alt + Shift + Analyze/ Analyze and Fix
Ask at runtime window		
	Mac	Window
Toggle between the labels and key of the variable	Cmd + K	Alt Gr + K

2.20 Page selector and Split scheme

For some features, it is very important to have a flexible option to determine which pages to process and which pages not to process (e.g. all even pages, all odd pages, only the first two and the last pages or every 4th page, etc.). For this purpose, a flexible syntax is used to express the desired groups of pages or page ranges.

A page selector expression can be a list e.g. 1,2,3. It can also be a simple expression, e.g. 3--3 will find all pages except the first and last two (last page being -1). Or it uses more advanced syntax: e.g. *3 will select every third page. More information below.

```
"page_selector": "all|even|odd|<splitscheme_expression>"
```

Click to copy

Split scheme is a TYPE of 'page selector'

Usage on CLI

The “Page selector” functionality can be used on CLI in the following way:

```
--pagerange=<expression>
```

Click to copy

This is the same syntax as used for split schemes (`--splitscheme=<expression>`)

Usage in the SDK

The “Page selector” functionality can be used in the SDK where the following parameter is listed (with slightly different types in other languages).

```
const PTB_sys_char_t*          pageselector          /*!< Use page selector syntax,
might be NULL for all pages */
```

Click to copy

Split Scheme

A split scheme expression may be a number with an asterisk "*" or a more complex string. If it is a number with an asterisk "*", it creates groups of pages where each group has the defined number of pages. E.g. if the expression is "3*" it would group the PDF into groups of 3 pages.

Expressions

Type	Syntax	Example	
Simple expression	number[-number]	1-5	Page 1 to 5: [1,2,3,4,5]
		5-1	Page 5 to 1: [5,4,3,2,1]
		8	Only page 8
		-1	Last page
		-3--1	Last 3 pages: [n, n-1, n-2]
		-1--3	Last 3 pages in reverse order: [n-2, n-1, n]
		-1-3	Last n - 2 pages in reverse

Type	Syntax	Example	
			order: [n, n-1, ... ,3]

Type	Syntax	
Simple expression with Simple Range	number[-number]_number[-number][","joker$]	

Type	Syntax
Multipage expression	even_pages
	uneven_pages
	Package number* [(start_page)]

Type	Syntax
	*number [(start_page)]
	*number [(start_page,end_page)]

Type	Syntax
Simple ex- pression list	simple_expression {"," simple_expres- sion}["," joker$]

Type	Syntax

Joker

```
<expression>,$
```

Click to copy
Can be combined with other expressions (has to be the last item in a list) in order to group all pages that are not part of any other expression into their own group.

Example

```
1-5,8,-3--1,$
```

Click to copy
would create groups of pages with pages 1-5, page 8, the last 3 pages and the rest of the pages of the PDF.

Gathering pages into PDF files

When splitting a PDF into multiple files, there is a difference between the comma "," and underscore "_" in the page selector. Whenever a comma is used, a new PDF file is created. Whenever an underscore is used, the current PDF file is continued. Examples:

```
1-5,7-9
1-5_7-9
```

Click to copy

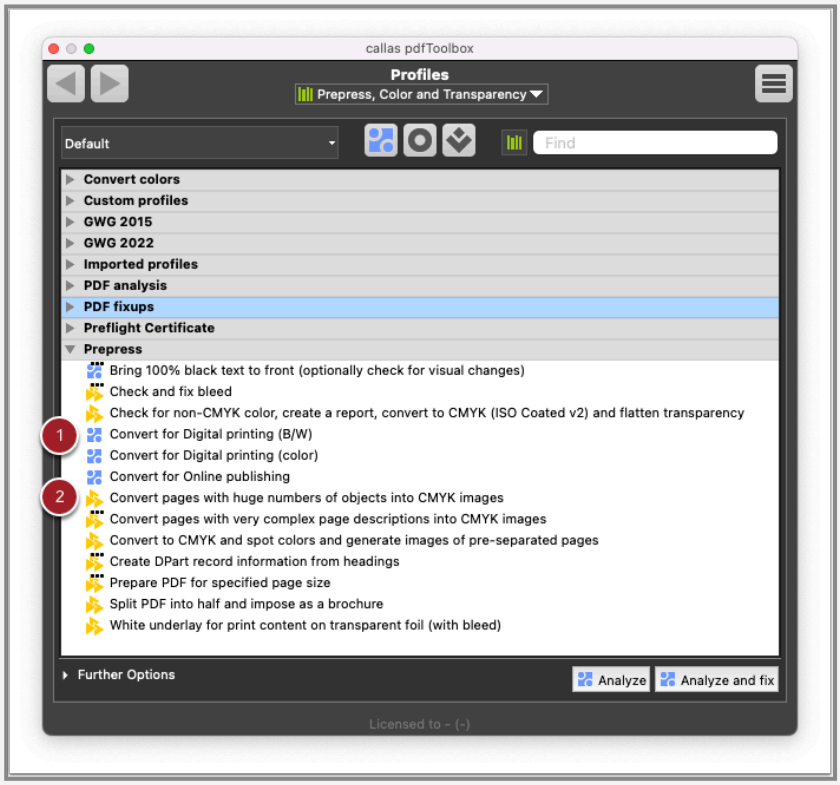
The first page selector generates two PDF files: one with pages 1 through 5, the second one with pages 7 through 9.

The second page selector generates only one PDF file with pages 1 through 5 and 7 through 9.

3. Process Plans in detail

3.1 When Profiles are not enough: Process Plans

When you look at the Profile window, you'll see that the list of Profiles contains two different types of items.



1. Regular Profiles, recognizable by their blue icons.
2. Process Plans, recognizable by their yellow icons.

Process Plans behave like Profiles in most ways. They can be run on PDF documents, you can import and export them. But Process Plans are built differently and they can be used to solve different problems.

Why Process Plans?

Process plans were invented to solve two very distinct problems:

- In a Profile you can have Checks and Fixups, but you cannot control the order in which these are executed. For Checks that is not a problem, Checks do not change the

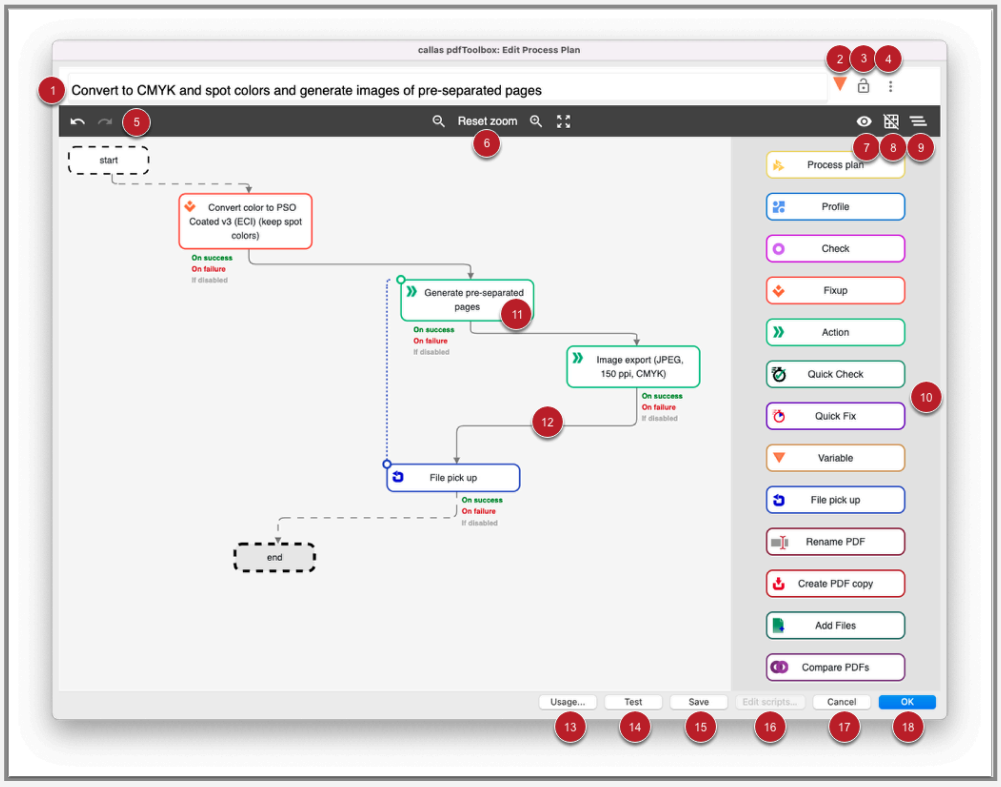
PDF document they are run on so their order is irrelevant. But for Fixups this can be a real problem. Sometimes you *need* to do things in a certain order.

- Sometimes you need to be a little careful when you execute a Fixup; you really only want to perform a fix if a certain condition is met.

Process plans make both of these things possible; you can control the order of execution and you can do conditional processing based on the result of a Check or Profile.

Anatomy of a Process plan

Users get a visual editor in which the steps can be clearly arranged and linked. The figure shows the editor "Edit Process Plan":



Area at the top: Basic information on the Process Plan

1. Field for naming the Process Plan (appears later in the "Profiles" window).

2. Button for adding a variable to the name of the Process Plan.

3. Button for locking or unlocking the Process Plan.

4. Pop-up menu: Here you can store metadata on the author of the Process plan including e-mail, password and description. This description is then displayed in the "Profiles" window and helps to communicate the purpose of the Process Plan.

The horizontal bar below: Editing and View Modes

5. Undo / Restore / (Delete connection, not visible in this view).

6. Reduce view / Reset zoom / Enlarge view / Adjust view to window.

7. Show or hide connection info.

8. Enable or disable automatic alignment to the grid.

9. Generate layout automatically (Note: The auto-generated layout can be undone using the Undo arrow in the top right corner (5)).

Vertical area on the right: List of elements that can be included in Process plans

10. Repertoire of sequence steps from a Process plan: Profile, Check, Fixup, Process Plan, Action, Quick Check, Quick Fix, Variable, File pick up, Rename PDF, Create PDF copy, Add files, Compare PDFs.

Main field in the middle: Area for arranging sequences

11. Example of a sequence step.

12. Example for connection type.

Horizontal area below: Completing the set-up Process plan

13. Check the usage (where this Process Plan is used) using the 'Usage' button.

14. Before saving, you can Check the Process Plan in practice by clicking on the "Test" button. More about the test mode in pdfToolbox can be found in the chapter "How to use the test mode".

15. A Process Plan that is not yet finished can be saved.

16. All the JavaScripts used in a Process Plan can be edited at the same time in Visual Studio Code. The "Edit Scripts" but-

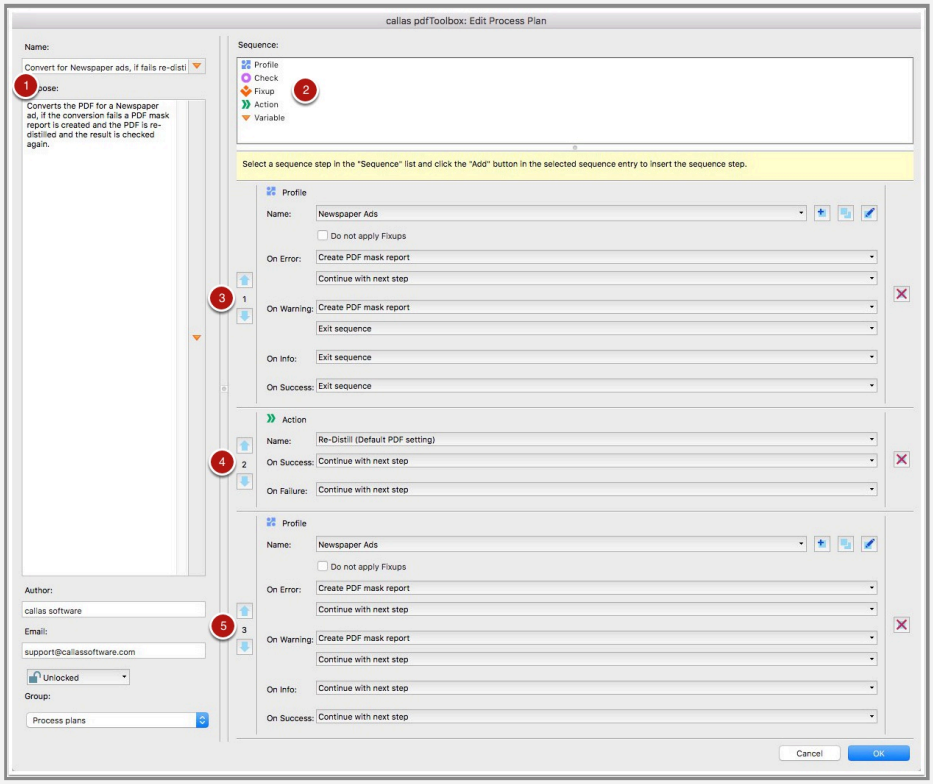
ton is only displayed, if the Visual Studio Code command line tools are installed. Learn more here.

17. "Cancel" editing.

18. Press "OK" to save the Process Plan.

Process plans before pdfToolbox 11

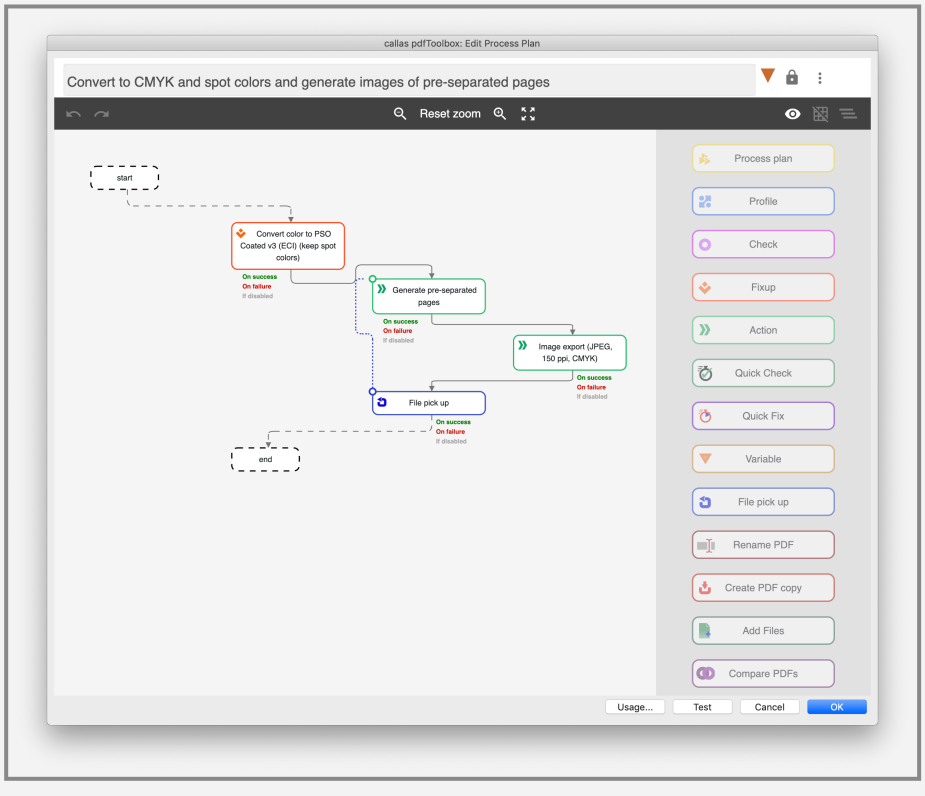
We introduced the NextGenUI Process plans in pdfToolbox 11. While the usage is the same as before, the UI looks totally different from the previous one. Below is the Process Plan editor before pdfToolbox 11:



1. Process plans of course have a name and description, just as Profiles.
2. This area at the top details what you can use as the steps in a Process Plans. Profiles, Checks and Fixups are described in this manual, actions and variables go beyond the scope of this manual but are described elsewhere in the online help.

3. The first step in this Process Plan is always executed; in this example that is a Profile. Notice the up and down arrows to allow changing the order of the steps in the Process Plan. This allows determining what exactly the order is going to be of all of the steps in the Process Plan. The "On Error", "On Warning", ... options allow you to specify what needs to happen based on the result of this first step. This could simply be executing the next step, but the options can also be used to jump to a specific step in the Process Plan or exit the sequence altogether.
4. This second step is an action, which was created in the Switchboard and which will re-distill the file.
5. The last step in this example is the same as the first step. This is to re-execute the same Profile if the first step wasn't successful and thus the re-distill action was used.

How is a Process plan structured?

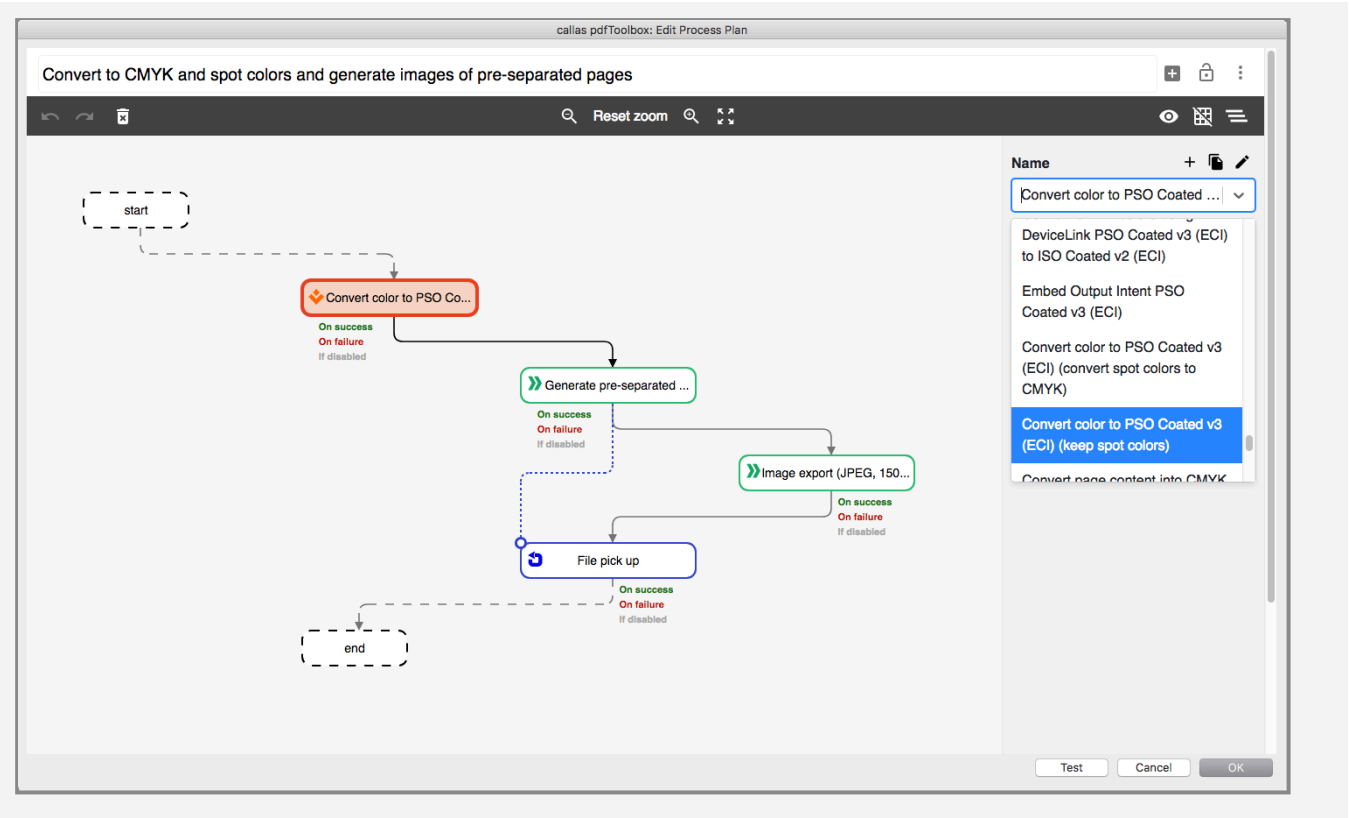


Process plans always have a start point, at least one sequence step and one end point.

Between start and end, sequence steps from the repertoire can be placed from the right-hand side. Sequence steps are possible in the following categories:

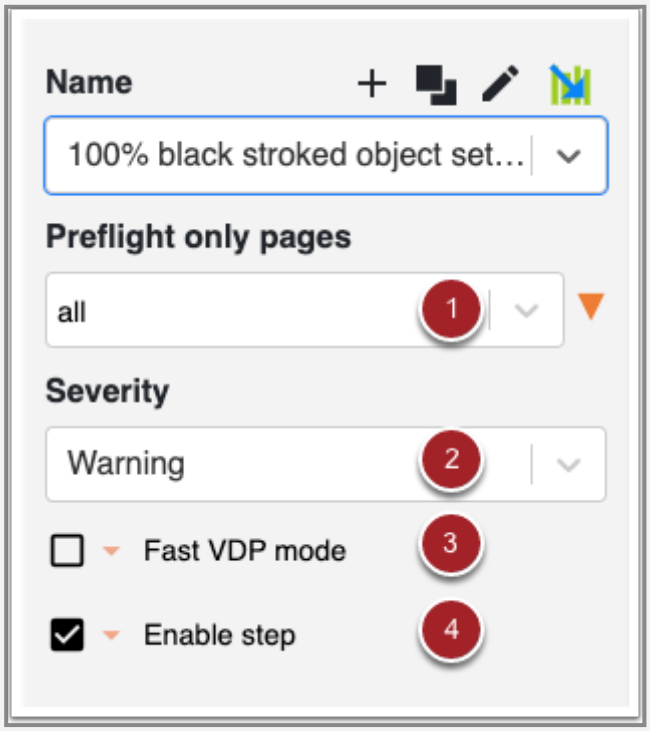
- Process plan
- Profile
- Check
- Fixup
- Action
- Quick Check
- Quick Fix
- Variable
- File pick-up
- Rename PDF
- Create a PDF copy
- Add Files
- Compare PDFs

Set up Sequence steps



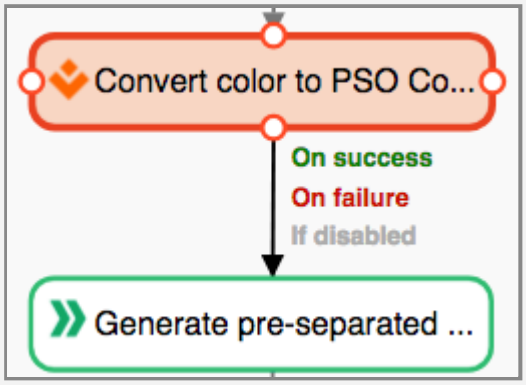
If you click on a placed sequence step, a pop-up menu is displayed on the right side, in which all Profiles / Checks / Fixups

/ etc. available in pdfToolbox can be selected - in each case suitable for the type of sequence step.



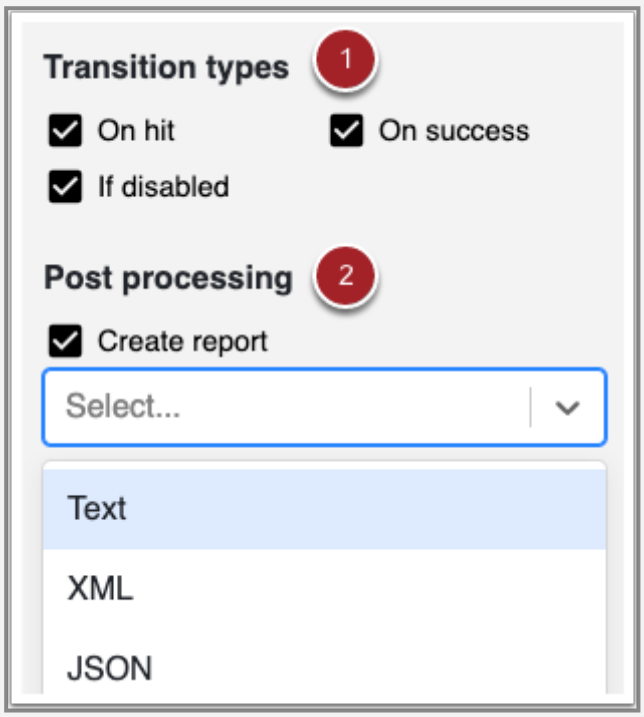
1. Since pdfToolbox 14 there is a page selector for Profiles, Fixups and Checks. Either a predefined option "all, equal or unequal" can be selected, or you apply a page selector expression. Further information can be found here in the article Page selector and Split scheme. If a page selector is defined, only pages referenced from the page selector will be executed by the sequence step.
2. Since pdfToolbox 14, a severity for a Check can be specified (Info, Error, Warning or a variable) which defines how a hit should be reported.
3. The "Fast VDP mode" checkbox only works for files representing variable data. Is it activated, XObjects are processed only once ignoring context. Read this article for further information.
4. A Sequence step can be activated or deactivated via the control field "Enable step". Deactivation is useful in cases where a step is to be skipped.

Connecting Sequence Steps



Sequence steps have four nodes that can be used to connect individual steps. To do this, click on the area of the initial step and lead the displayed connecting line to the desired next step.

Connection type



1. If you click on a connection line, you can use the Check-boxes displayed on the right-hand side to specify the con-

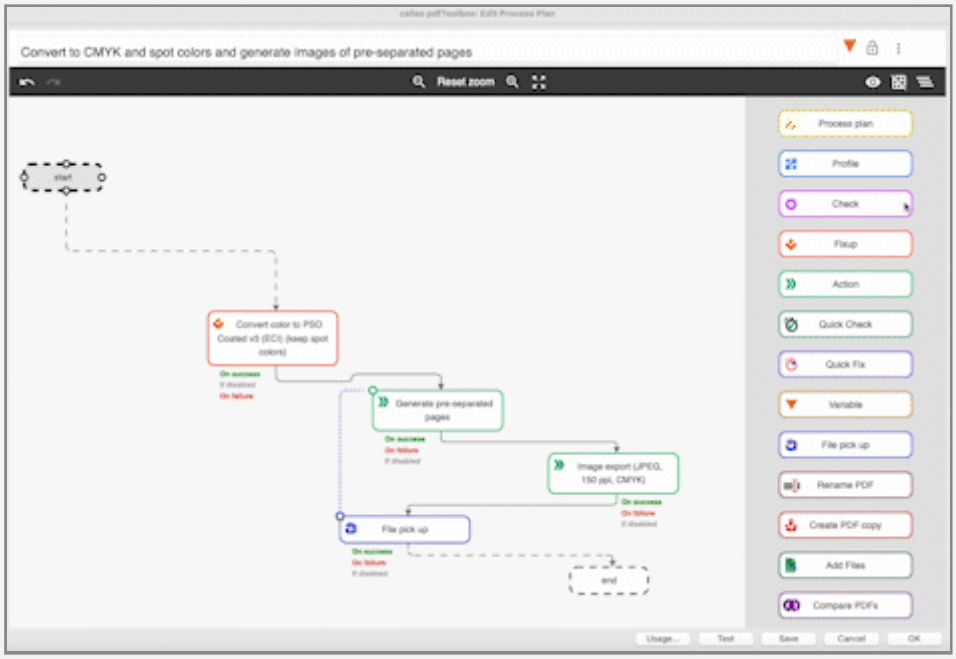
ditions under which a subsequent and connected step is to be triggered (not all connection types mentioned here will necessarily be visible, as they are context-dependent):

- On success
- On failure
- On hit
- On success
- If disable

2. You can also create a report in postprocessing, whereby the report type can be set (for example XML, Text, JSON or PDF report) . This option is only available for certain sequence types.

Add a new sequence step

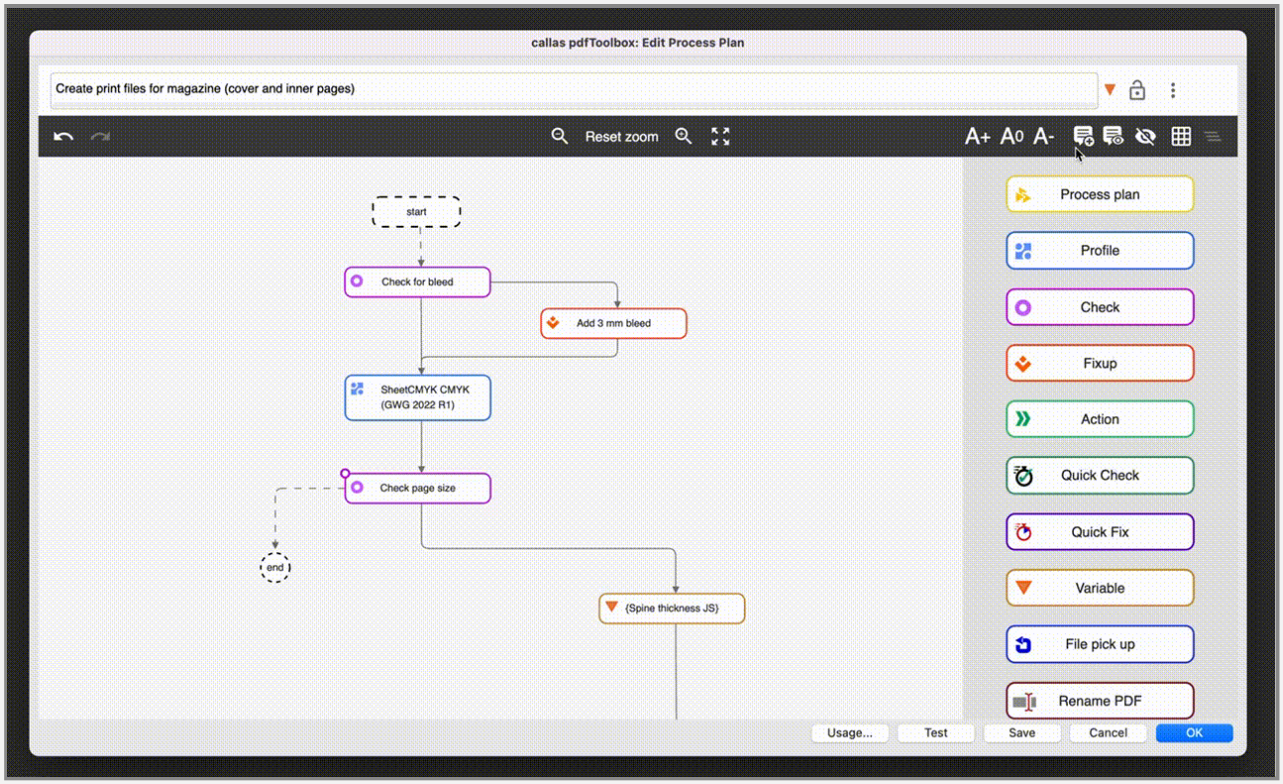
Adding a new sequence step to a process plan is done by drag and drop. The required sequence step is selected from the right column and then dragged to the desired position in the process plan. When the connection line lights up blue, the step can be added.



3.2 Process Plans: tips and tricks

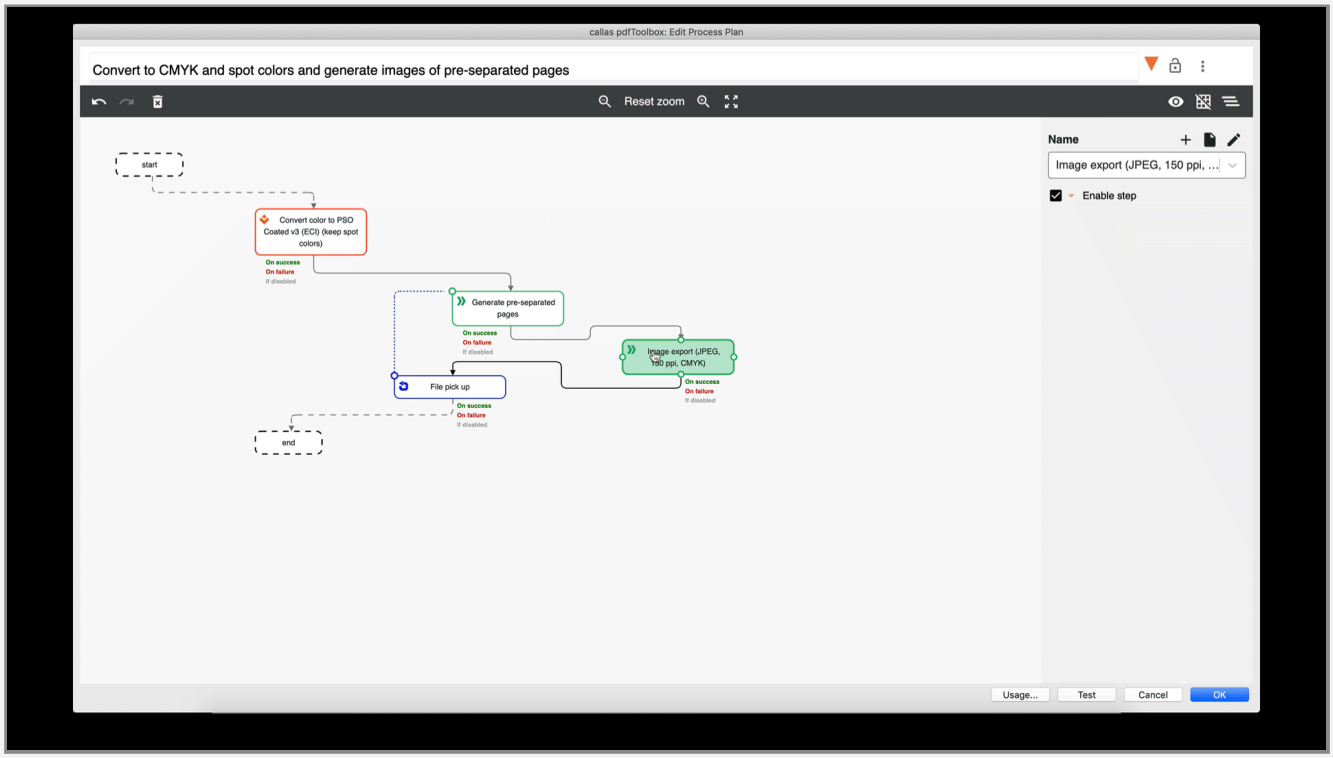
Annotations within Process Plan

Starting with pdfToolbox 16, the Process Plan editor supports annotations to help you better document your work. The information added to the "Text" parameter is displayed within the annotation. The "Comment" parameter can be used to provide additional information that is only visible when the annotation is selected. Use the color picker to change the color of the annotation and the text. There is also an icon in the top menu bar to toggle the visibility of annotations.



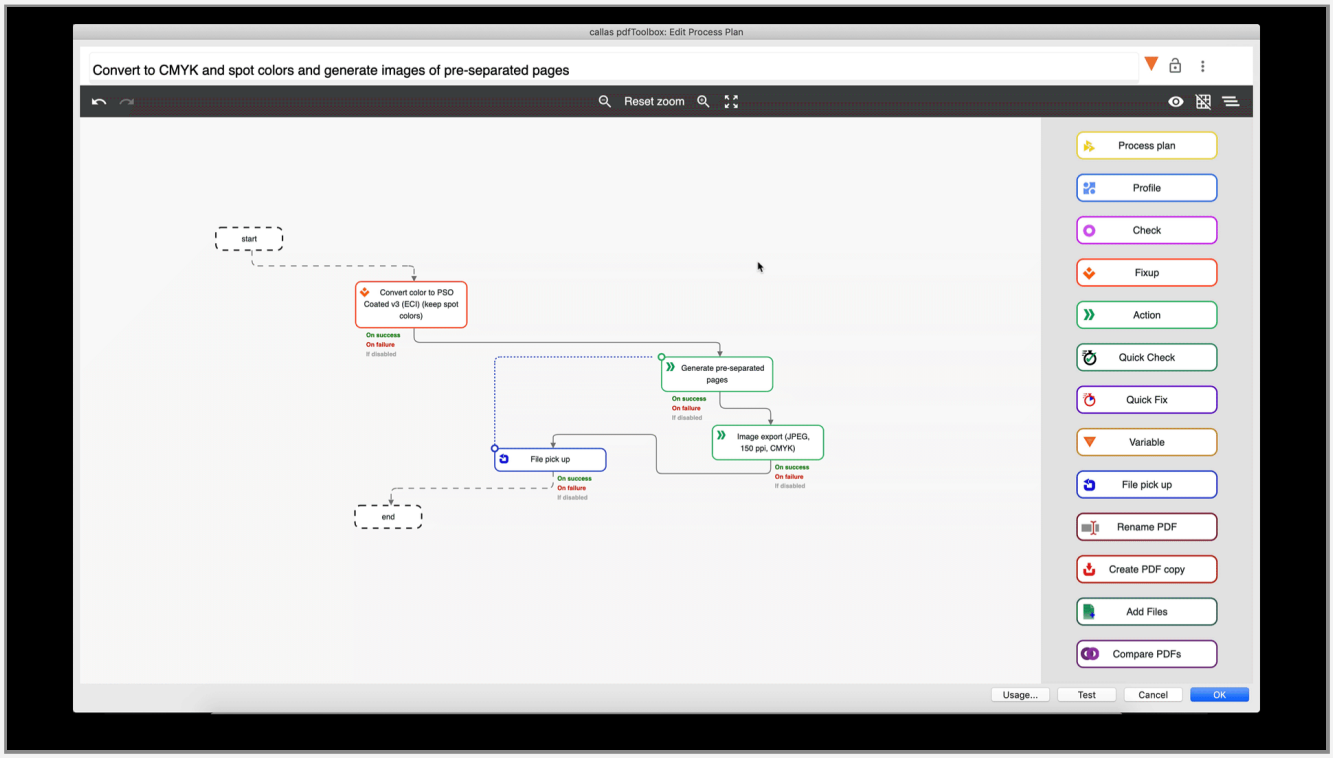
Multiple selections of Process Plan steps

Starting with pdfToolbox 12.2, Process Plans allow multiple selections of steps with **shift+click** in order to re-arrange steps in a process plan. Move all the selected items together wich **click+drag**.



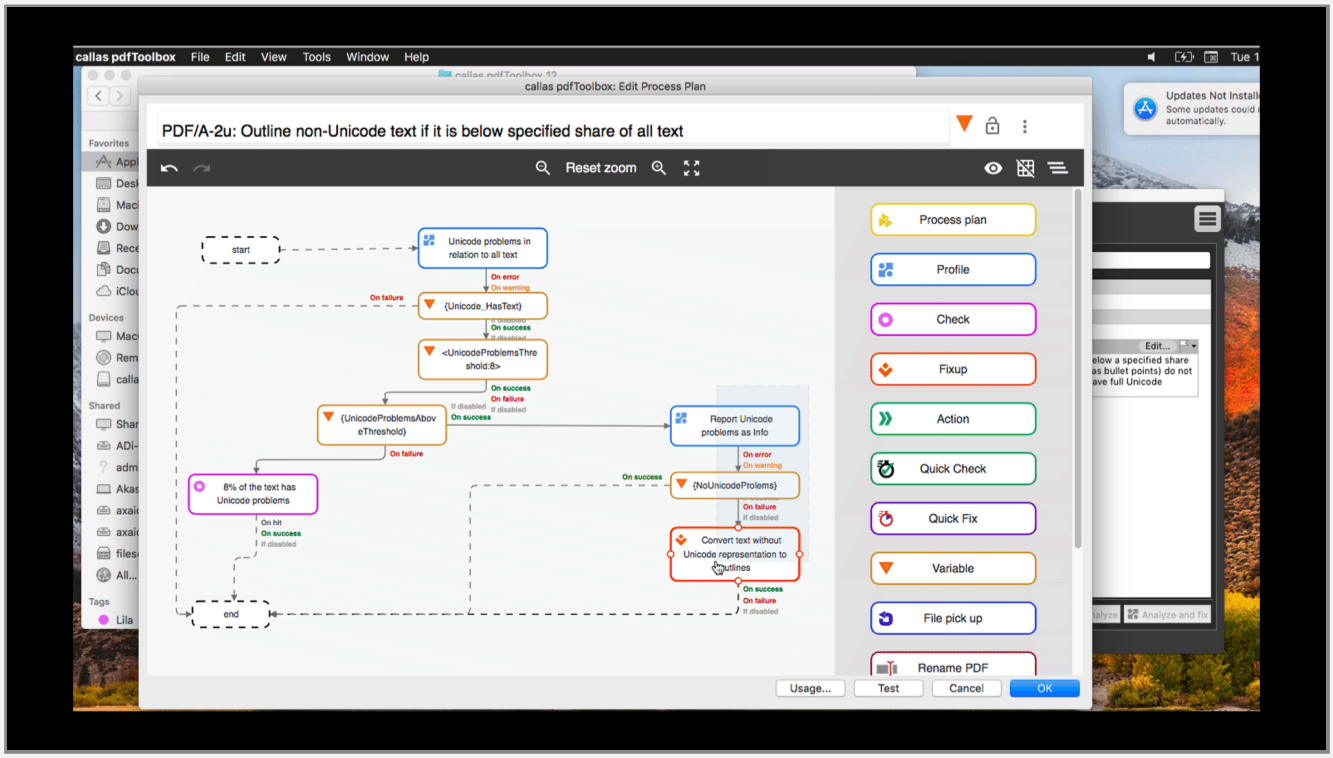
Move canvas

Prior to pdfToolbox 12.2, the canvas could be moved by click + drag inside the canvas. Since click + drag is now used for multiple selection of steps, the gesture for moving the canvas has been changed to **SPACE + Click + Drag**.



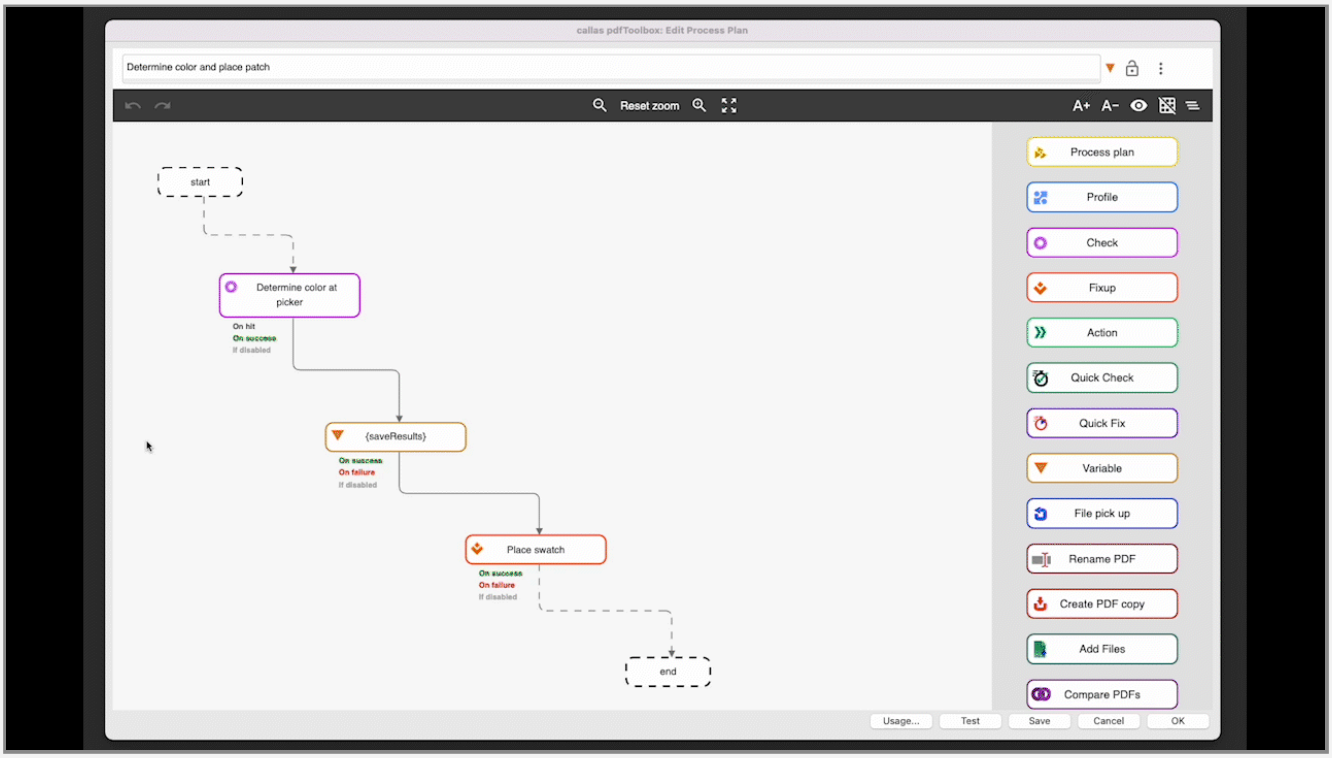
Auto-scrolling

pdfToolbox 12.2 onwards, you can select Process Plan items and move them out of the canvas to allow auto-scrolling.



Alignment of Process Plan steps

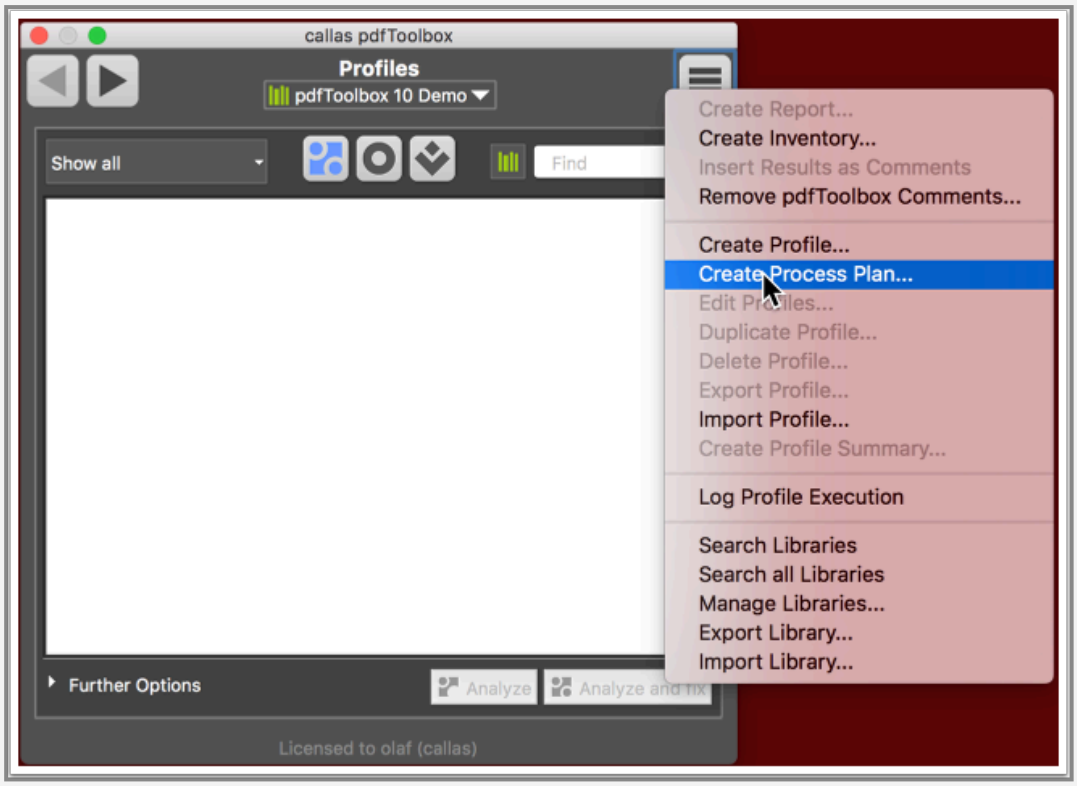
Since pdfToolbox 15, you can select Process Plan steps and align them horizontally or vertically.


callas pdfToolbox: Edit Process Plan
Determine color and place patch
Reset zoom
start
Determine color at picker
On hit
On success
If disabled
{saveResults}
On success
On failure
If disabled
Place swatch
On success
On failure
If disabled
end
Process plan
Profile
Check
Fixup
Action
Quick Check
Quick Fix
Variable
File pick up
Rename PDF
Create PDF copy
Add Files
Compare PDFs
Usage...
Test
Save
Cancel
OK

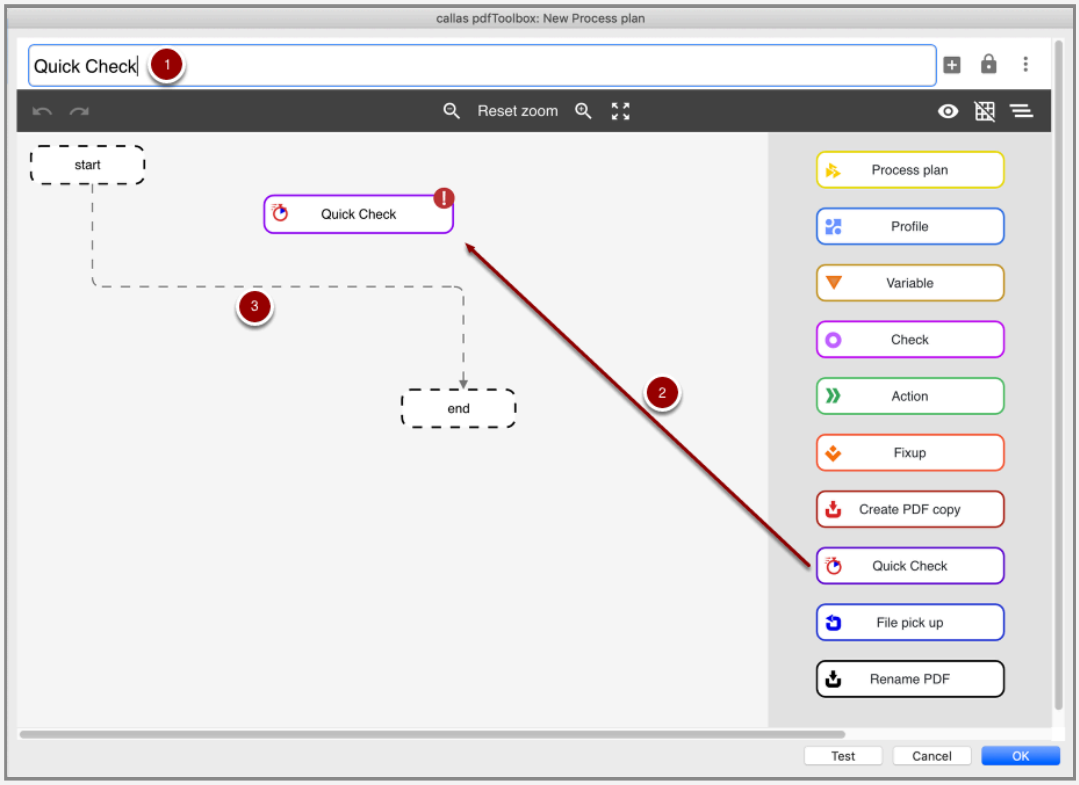
3.3 Using Quick Check as a step in a Process Plan

When creating or editing a Process Plan, it is possible to insert a Quick Check step in the Process Plan.

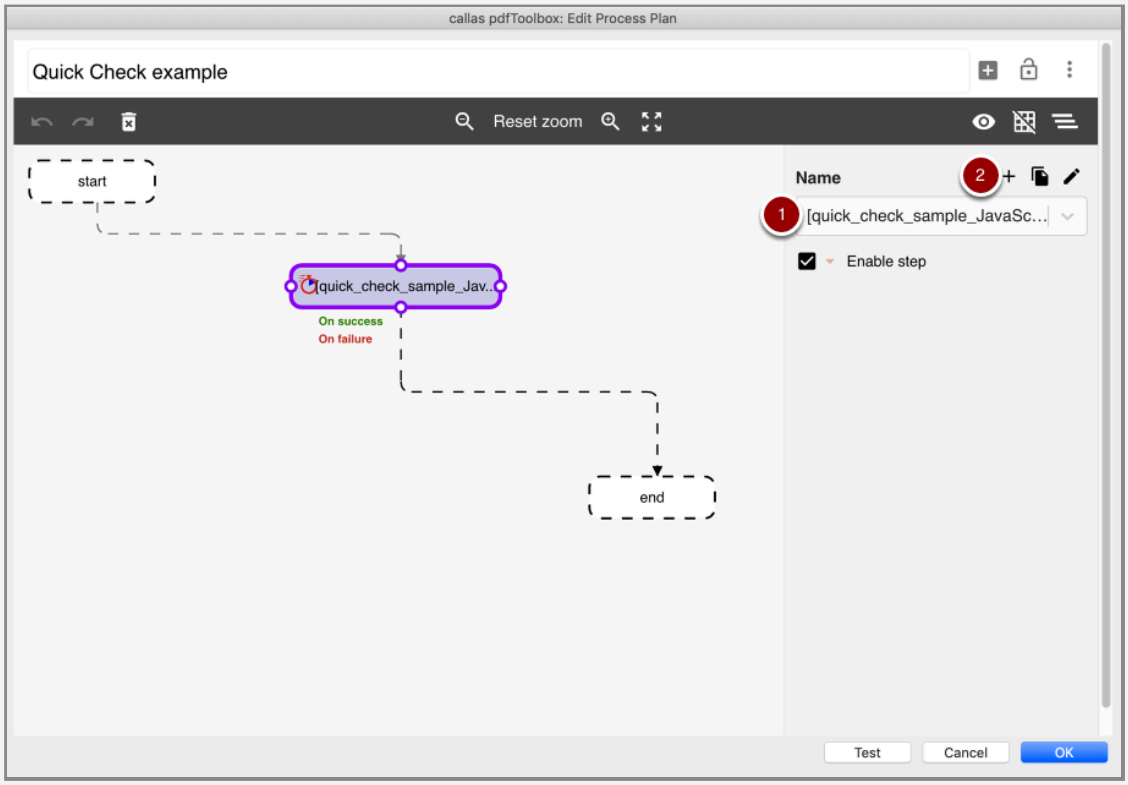
For the purpose of this article, a new Process Plan is created:



Add Quick Check:



1. In the top left field you can enter a suitable name for the new Process plan
2. Select "Quick Check" from the "Sequence" list on the right and drag this step onto the area of the Process plan editor
3. Delete the default connection type between "Start" and "End" (dashed line) and connect "Start" with "Quick Check" and "Quick Check" again with "End" (as shown below)

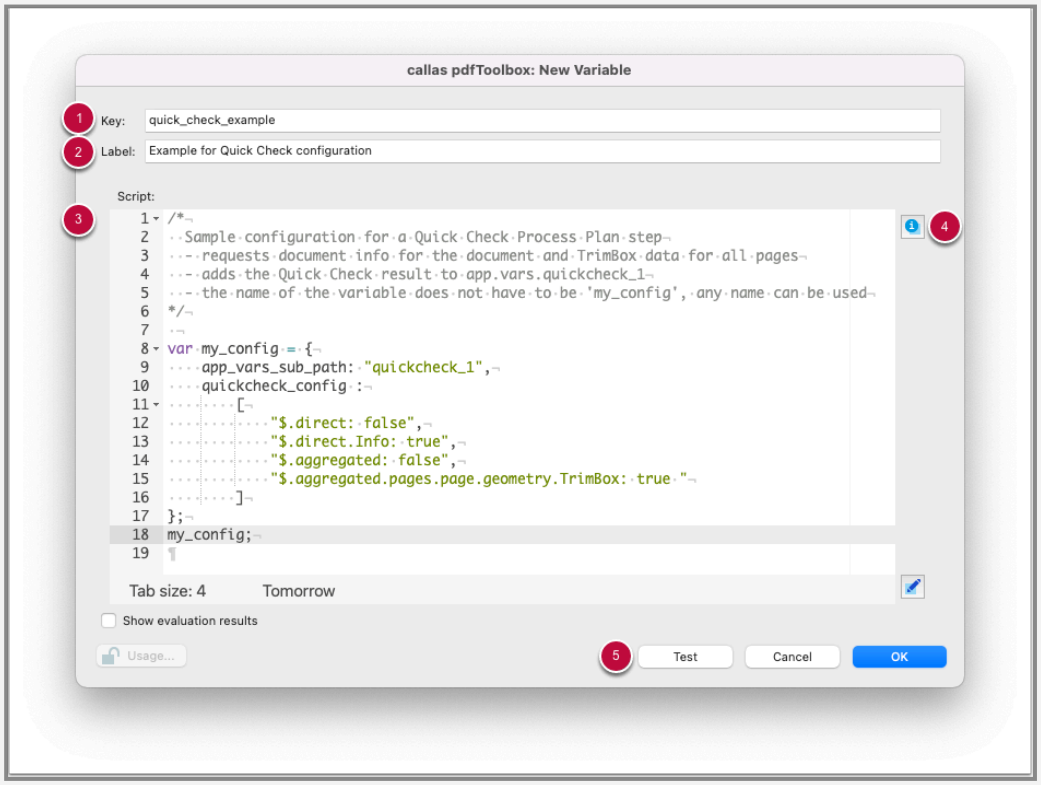


The Quick Check step has now been inserted into the process plan as an element.

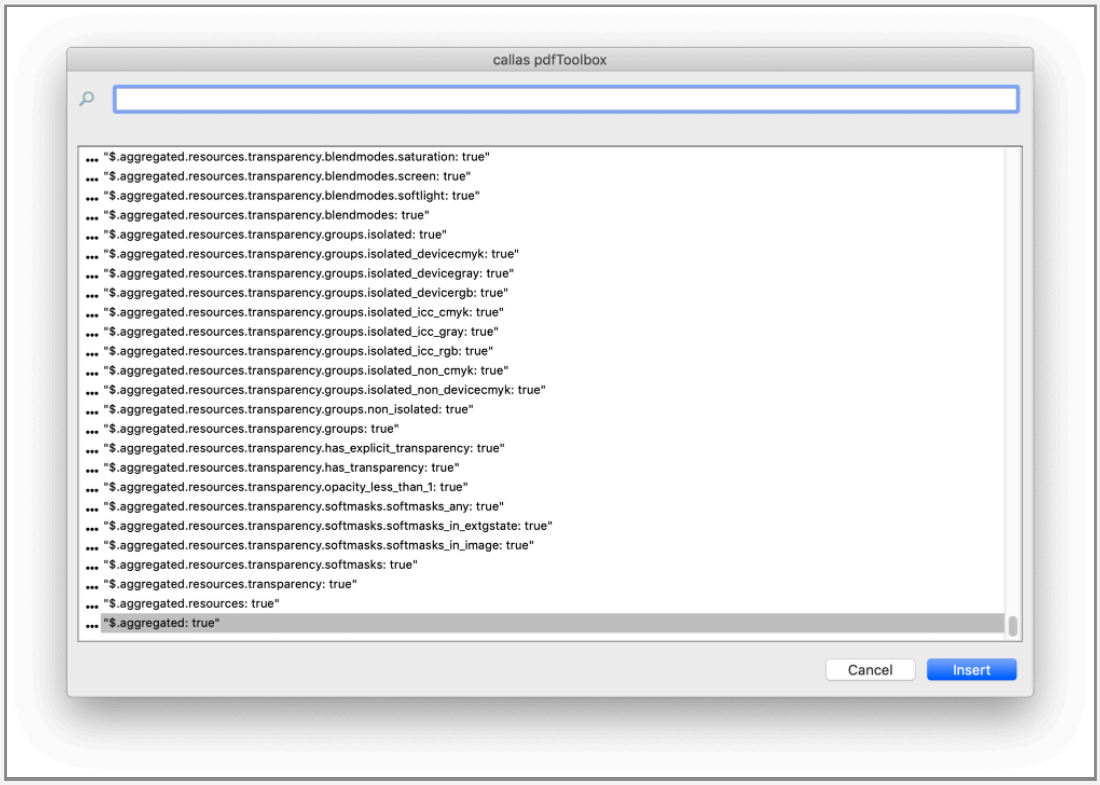
1. In order to set up what the Quick Check step should contribute as a result, either a suitable Quick Check JavaScript variable is selected from the "Name" drop-down menu,
2. or a new Quick Check JavaScript variable is created using the "+" symbol (explained below).

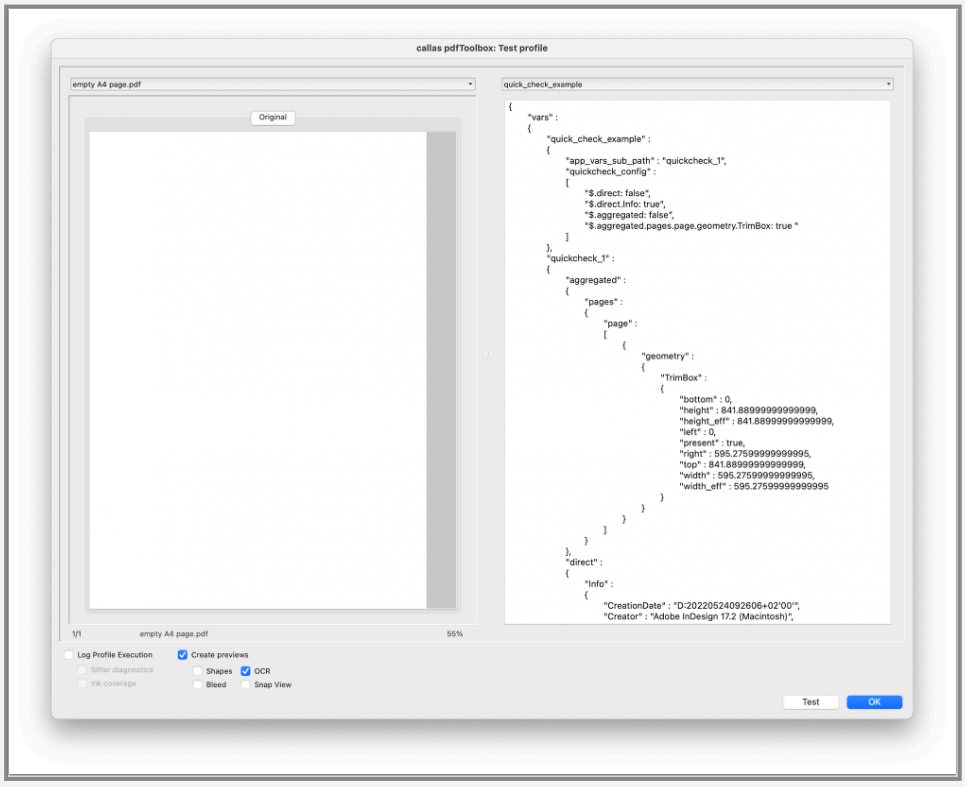
New Quick Check JavaScript variable

When creating a new Quick Check-JavaScript variable, the Edit Quick Check window is opened, as shown below.



1. Enter suitable value for the "Key" (by which the variable can be referenced in the `app.vars` object)
2. Enter suitable value for the "Label" (shown in the user interface when displaying a reference to the variable)
3. Edit the "Script" (what is shown above is the default script that requests the TrimBox for all pages)
4. To view the list of all Quick Check objects, click 'i' and 'Insert Quick Check objects' (a new window opens that is shown below)
5. The "Test" button will open the "Test profile" window which shows the output of the Quick Check Variable (here the size of the TrimBox)


callas pdfToolbox
"$.aggregated.resources.transparency.blendmodes.saturation: true"
"$.aggregated.resources.transparency.blendmodes.screen: true"
"$.aggregated.resources.transparency.blendmodes.softlight: true"
"$.aggregated.resources.transparency.blendmodes: true"
"$.aggregated.resources.transparency.groups.isolated: true"
"$.aggregated.resources.transparency.groups.isolated_devicecmyk: true"
"$.aggregated.resources.transparency.groups.isolated_devicegray: true"
"$.aggregated.resources.transparency.groups.isolated_devicergb: true"
"$.aggregated.resources.transparency.groups.isolated_icc_cmyk: true"
"$.aggregated.resources.transparency.groups.isolated_icc_gray: true"
"$.aggregated.resources.transparency.groups.isolated_icc_rgb: true"
"$.aggregated.resources.transparency.groups.isolated_non_cmyk: true"
"$.aggregated.resources.transparency.groups.isolated_non_devicecmyk: true"
"$.aggregated.resources.transparency.groups.non_isolated: true"
"$.aggregated.resources.transparency.groups: true"
"$.aggregated.resources.transparency.has_explicit_transparency: true"
"$.aggregated.resources.transparency.has_transparency: true"
"$.aggregated.resources.transparency.opacity_less_than_1: true"
"$.aggregated.resources.transparency.softmasks.softmasks_any: true"
"$.aggregated.resources.transparency.softmasks.softmasks_in_extgstate: true"
"$.aggregated.resources.transparency.softmasks.softmasks_in_image: true"
"$.aggregated.resources.transparency.softmasks: true"
"$.aggregated.resources.transparency: true"
"$.aggregated.resources: true"
"$.aggregated: true"
Cancel
Insert


callas pdfToolbox: Test profile
empty A4 page.pdf
quick_check_example
Original
{
"vars" :
{
"quick_check_example" :
{
"app_vars_sub_path" : "quickcheck_1",
"quickcheck_config" :
[
"$.direct: false",
"$.direct.Info: true",
"$.aggregated: false",
"$.aggregated.pages.page.geometry.TrimBox: true "
]
},
"quickcheck_1" :
{
"aggregated" :
{
"pages" :
{
"page" :
[
{
"geometry" :
{
"TrimBox" :
{
"bottom" : 0,
"height" : 841.88999999999999,
"height_eff" : 841.88999999999999,
"left" : 0,
"present" : true,
"right" : 595.27599999999995,
"top" : 841.88999999999999,
"width" : 595.27599999999996,
"width_eff" : 595.27599999999995
}
}
}
]
}
},
"direct" :
{
"Info" :
{
"CreationDate" : "D:20220524092606+02'00'",
"Creator" : "Adobe InDesign 17.2 (Macintosh)",
1/1
empty A4 page.pdf
55%
Log Profile Execution
Create previews
Sifter diagnostics
Ink coverage
Shapes
OCR
Bleed
Snap View
Test
OK

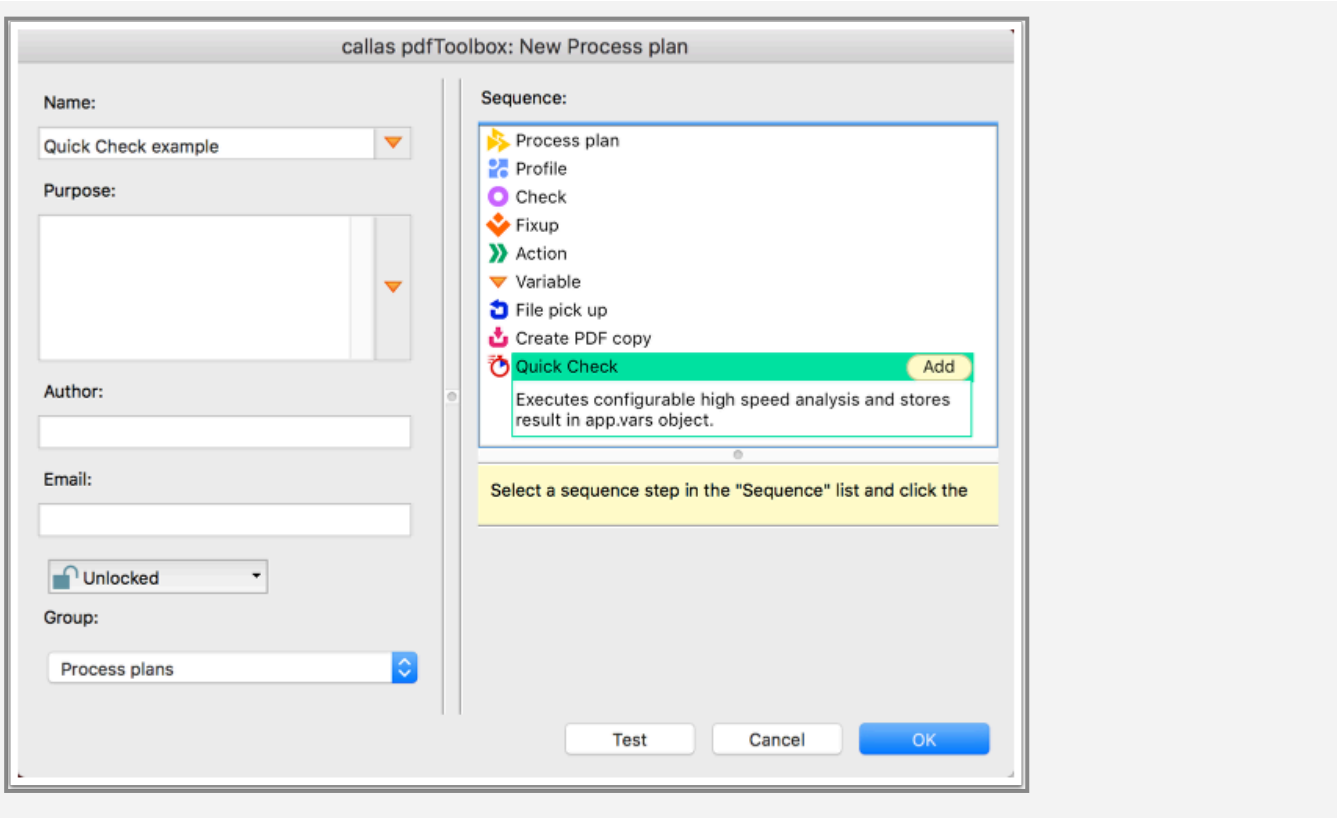
After inserting the required Quick Check object, click OK to save this Quick Check-JavaScript variable, and click OK again to save the new Process Plan.

Whenever this Process Plan is executed, the TrimBox of all pages gets stored in the `app.vars` object under `quickcheck_1`.

Obviously, in this simple form the Process Plan is not very useful. The next part of the article shows a slightly extended variant of this Process Plan, that actually makes some interesting use of the information retrieved by the Quick Check step.

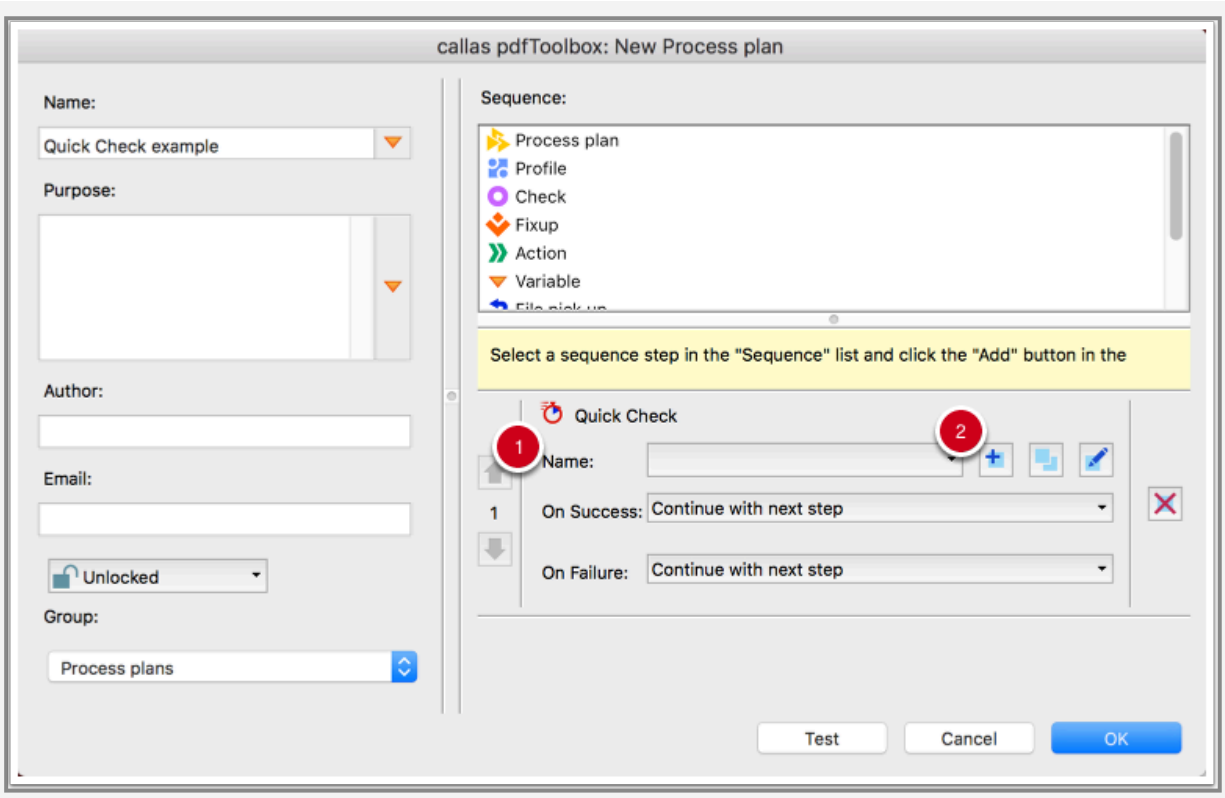
Old Process Plan editor

After entering a suitable name for the new Process Plan, select "Quick Check" from the "Sequence" list and click on the "Add" button in that list entry:



A new "Quick Check" step gets inserted as the first item in the list of Process Plan steps.

1. In order to configure what the "Quick Check" step is to provide as a result, either a suitable Quick Check-JavaScript variable has to be chosen in the "Name" pop-up list,
2. or a new Quick Check-JavaScript variable has to be created using the "+" icon.



How to use information retrieved by a Quick Check step in a Process Plan

In order to try out the Process Plan presented below, please download it here:

Quick_Check_example.kfpx

After importing the downloaded "Quick_Check_example.kfpx" file and importing it into pdfToolbox Desktop, it should show up as can be seen below:



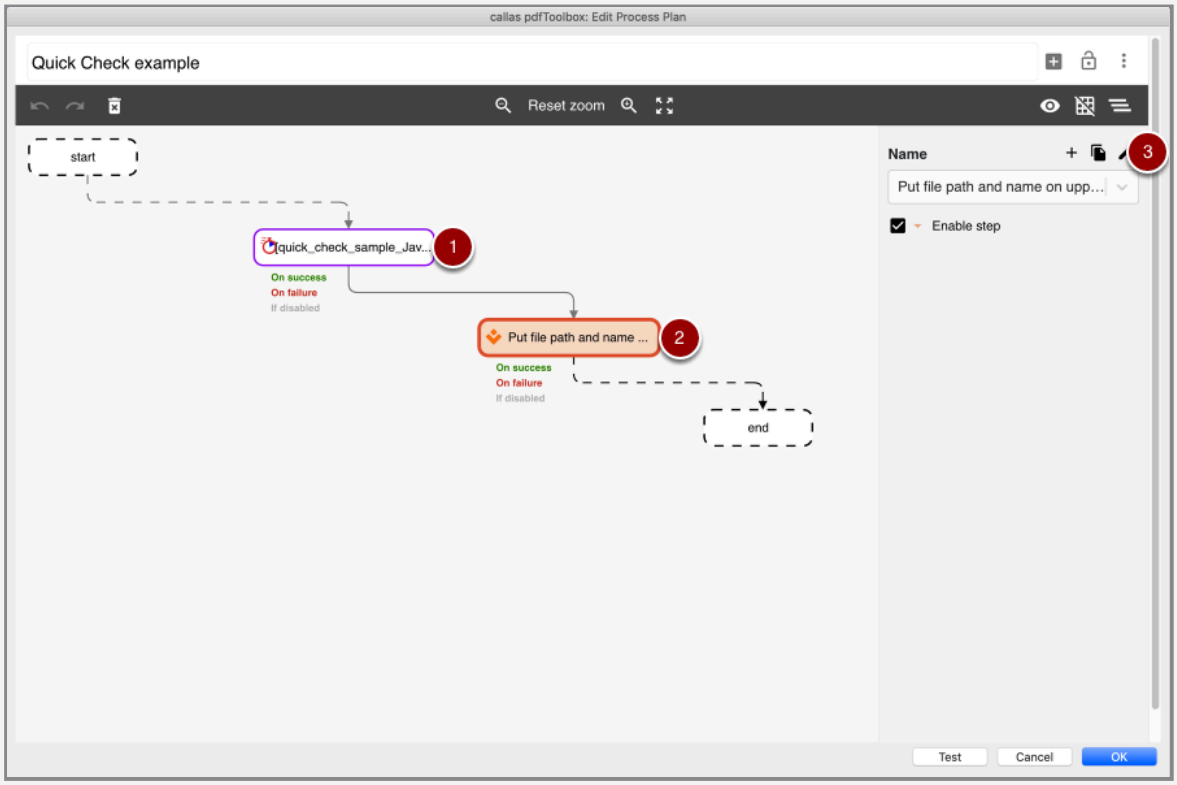
In order to explore how this Process Plan has been constructed, click on the "Edit..." button next to the "Quick Check example" entry.

The "Edit Process Plan" dialog will open. It contains two steps:

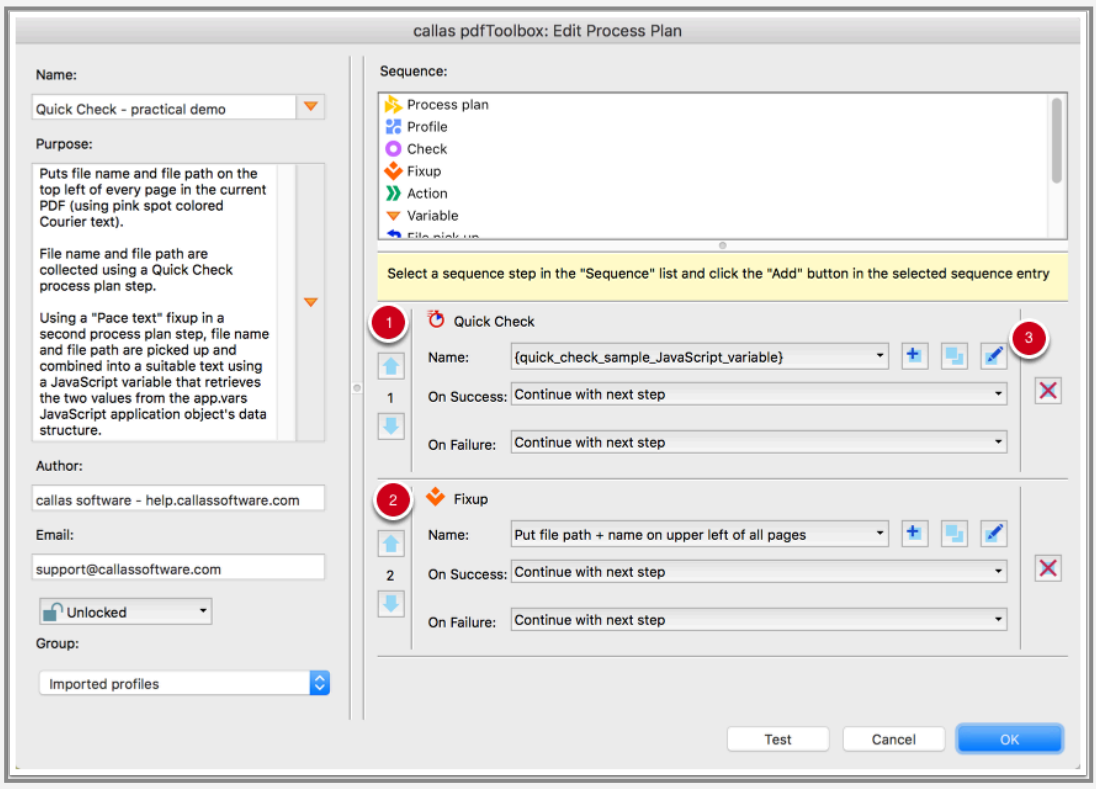
1. Quick Check step using a variable named "quick_check_sample_JavaScript_variable" that collects information from the current PDF
2. Fixup step "Put file name and file path on upper left of all pages" that picks up the collected information and uses it to place text on each page of the current PDF

In order to have a look at the way the JavaScript variable has been set up,

3. click on the Edit button to the right of the "Name" popup.



Alternate image for the older Process Plan UI:

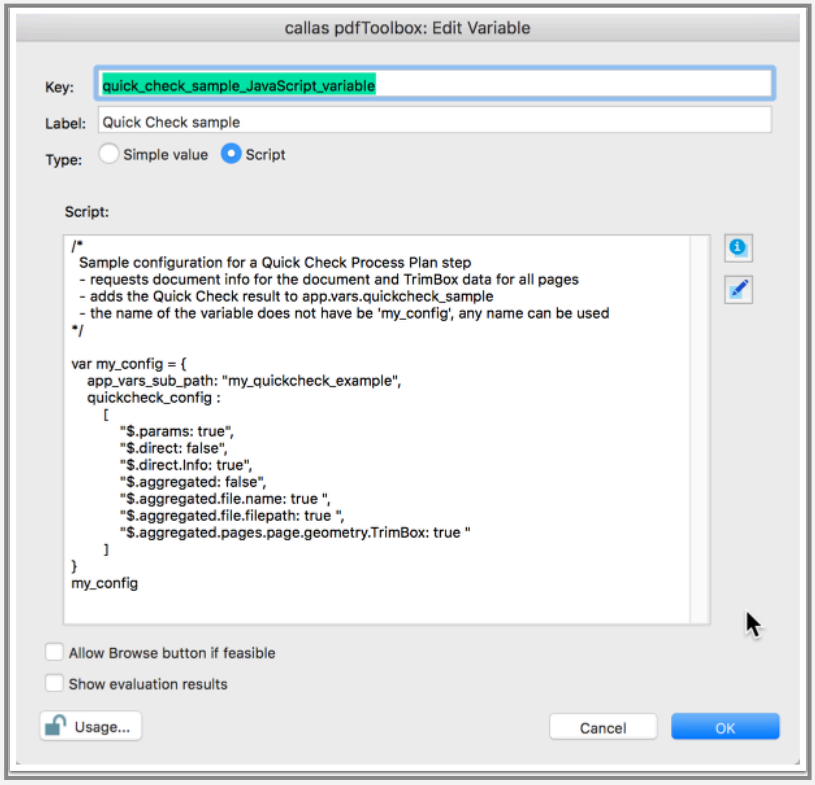


The "Edit Variable" window will open, revealing the setup of the variable "quick_check_sample_JavaScript_variable".

The relevant part of the configuration are these two lines:

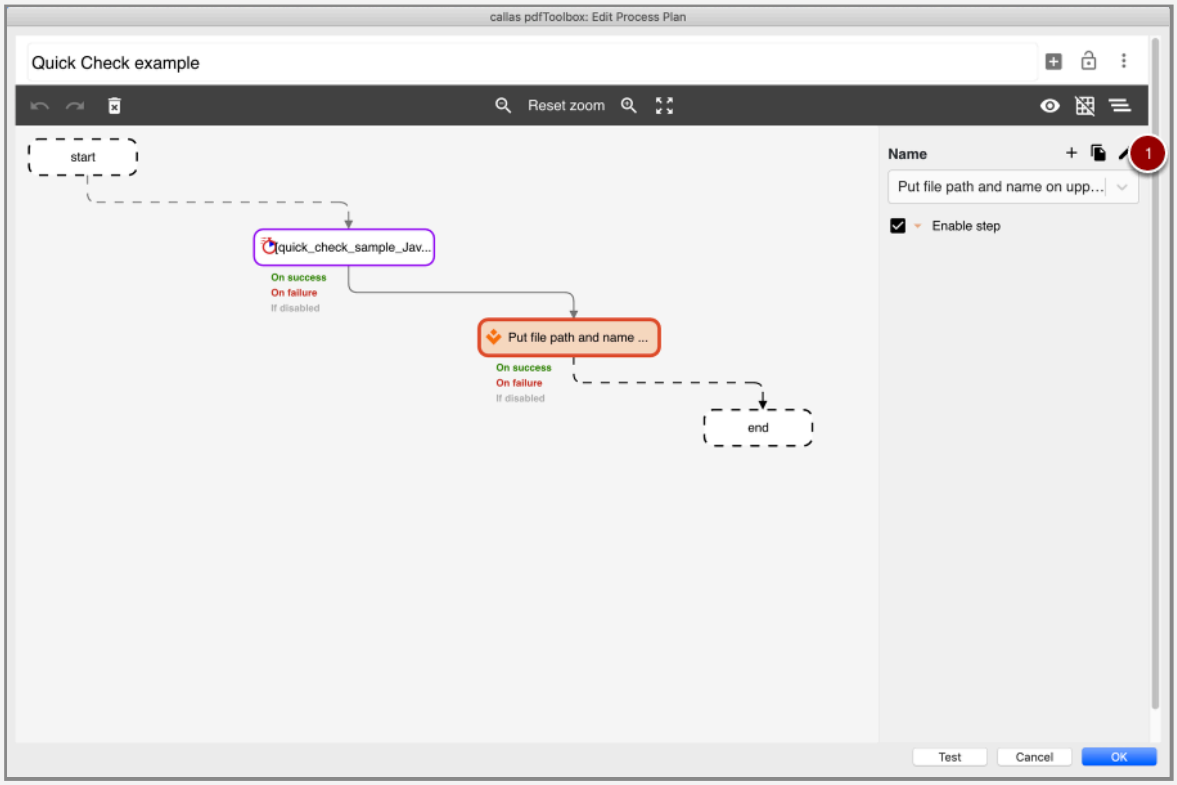
```
            "$.aggregated.file.name: true ",
            "$.aggregated.file.filepath: true ",
```

which request that Quick Check retrieves information about the file name and the file path of the current PDF. For a detailed description of the configuration set up please check out the article Quick Check configuration syntax.

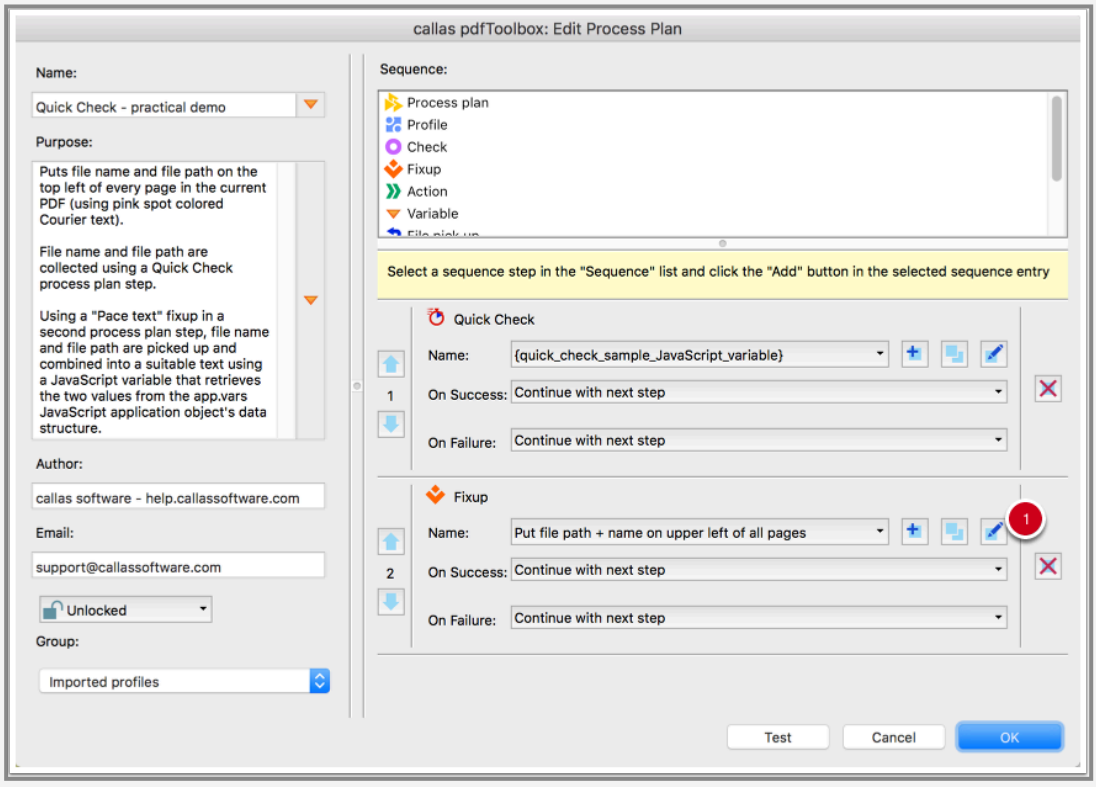


Close the "Edit Variable" window, and thus go back to the Edit window for the Process Plan.

1. Click on the Edit icon to the right of the Name entry for the Fixup step.



Alternate image for the older Process Plan UI:

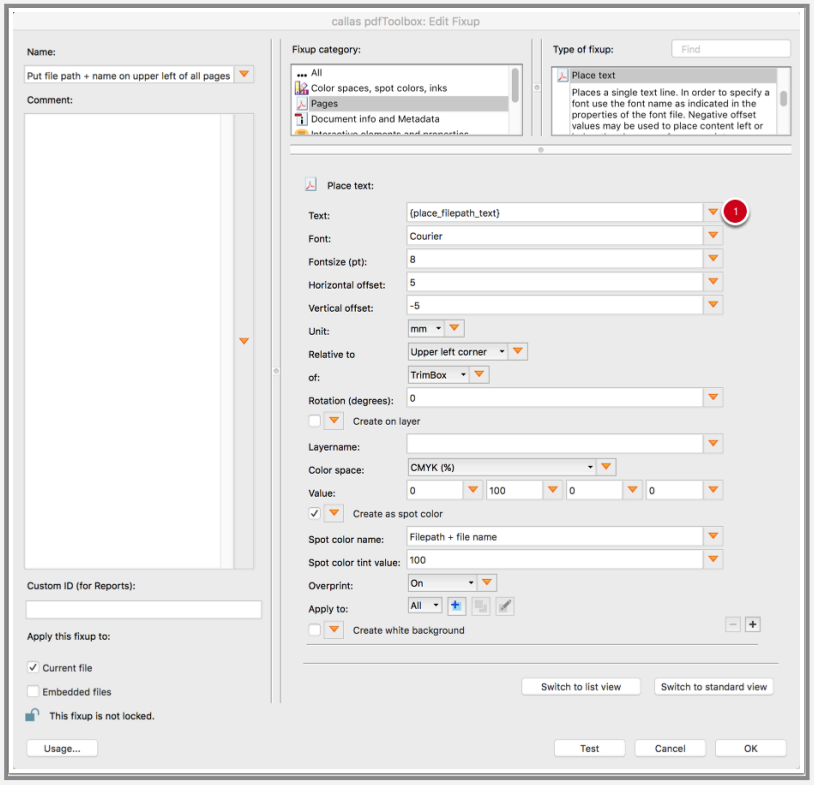


The "Edit Fixup" window for the "Put file path and name on upper left of all pages" Fixup will open.

The numerous fields in this Fixup are filled with values that will make the contents of the "Text" field show up in the upper left of all pages of the current page using Courier font in pink spot color.

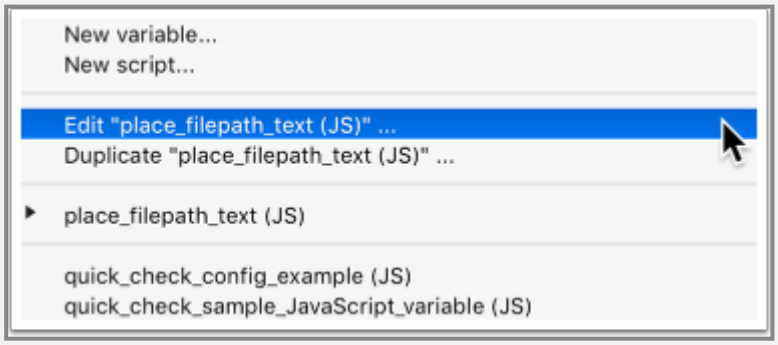
The more interesting part of this Fixup at least in this context is how the contents of the "Text" is defined.

1. Click on the orange triangle popup to the right of the "Text" field.



A list of already defined variables is now shown, plus various options regarding existing or new variables.

1. Click on 'Edit "place_filepath_text"...'



The "Edit Variable" window for the variable "place_filepath_text" will open.

Important: this not the same variable as the JavaScript variable defined in the QuickCheck step (which configures that Quick Check step), rather, it is a separate JavaScript variable that makes use of the information retrieved by the Quick Check step.

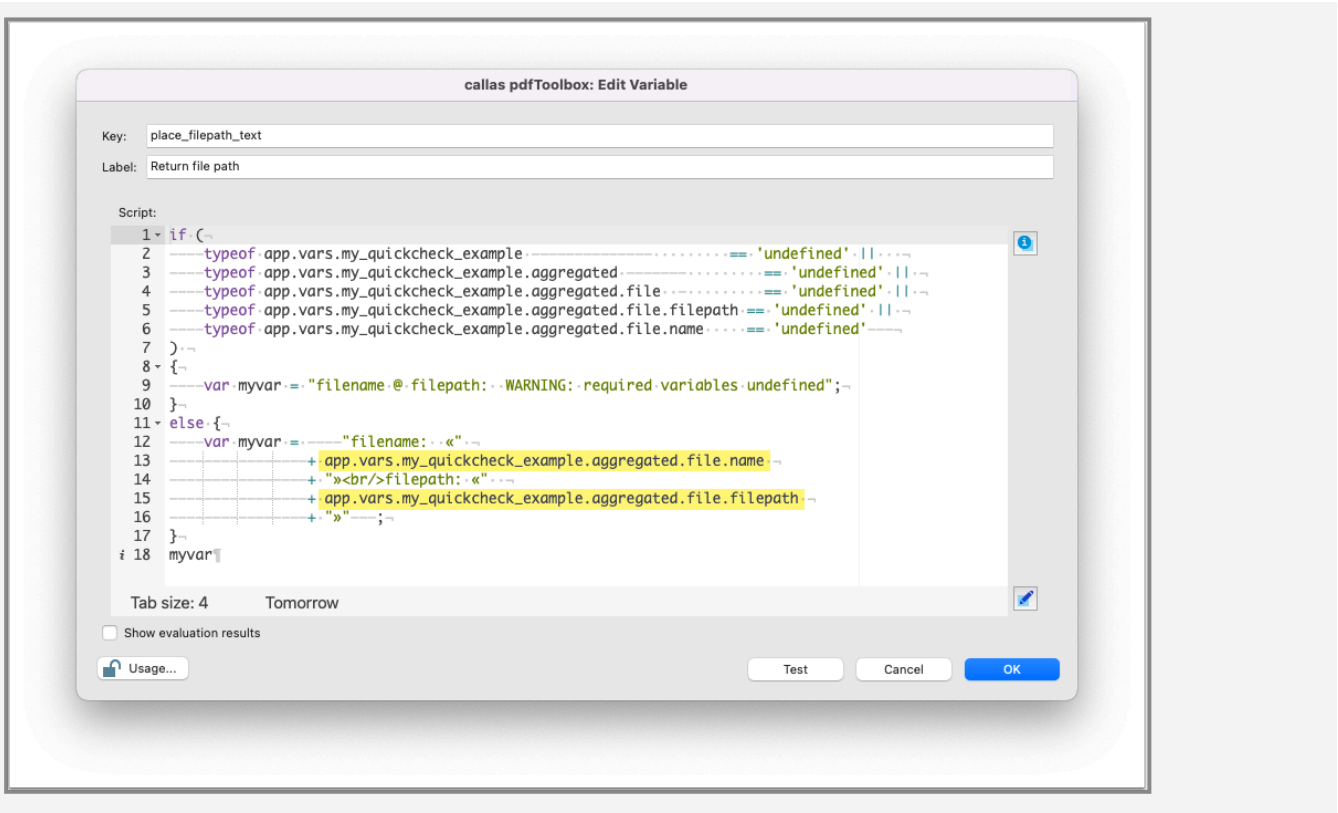
As that information will only be available if the Fixup gets executed as part of a Process Plan, where in a previous step a suitable Quick Check step has been executed. Some precautions are taken in the JavaScript code that give a kind of error message "filename @ filepath: WARNING: required variables undefined" if the Fixup is executed on its own or in a different Process Plan or Profile. This is managed by the `if` branch of the JavaScript code:

```
if (
        typeof app.vars.my_quickcheck_example
           == 'undefined' ||
        typeof app.vars.my_quickcheck_example.aggregated
 == 'undefined' ||
        typeof app.vars.my_quickcheck_example.aggregated.file                == 
'undefined' ||
        typeof app.vars.my_quickcheck_example.aggregated.file.filepath  == 'unde-
fined' ||
        typeof app.vars.my_quickcheck_example.aggregated.file.name      == 'unde-
fined'
)
{
        var myvar = "filename @ filepath:  WARNING: required variables undefined";
}
```

The intended main portion of the JavaScript variable is contained in the `else` branch:

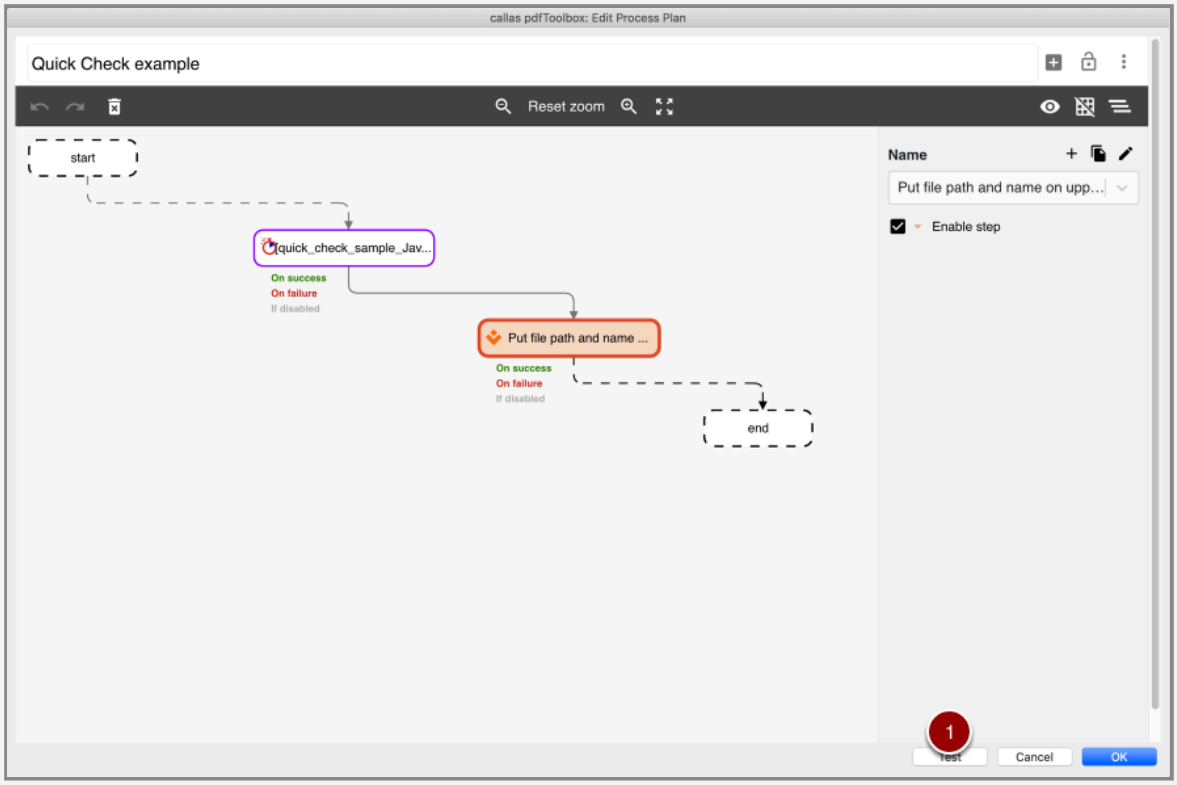
```
else {
        var myvar =         "filename:  «"
                                + app.vars.my_quickcheck_example.aggregated.file.
name
                                + "»<br/>filepath: «"
                                + app.vars.my_quickcheck_example.aggregated.file.
filepath
                                + "»"        ;
}
```

This else branch creates a string built from two variables, `app.vars.my_quickcheck_example.aggregated.file.name` and `app.vars.my_quickcheck_example.aggregated.file.filepath`, concatenated with other bits of text into a nicely formatted string.

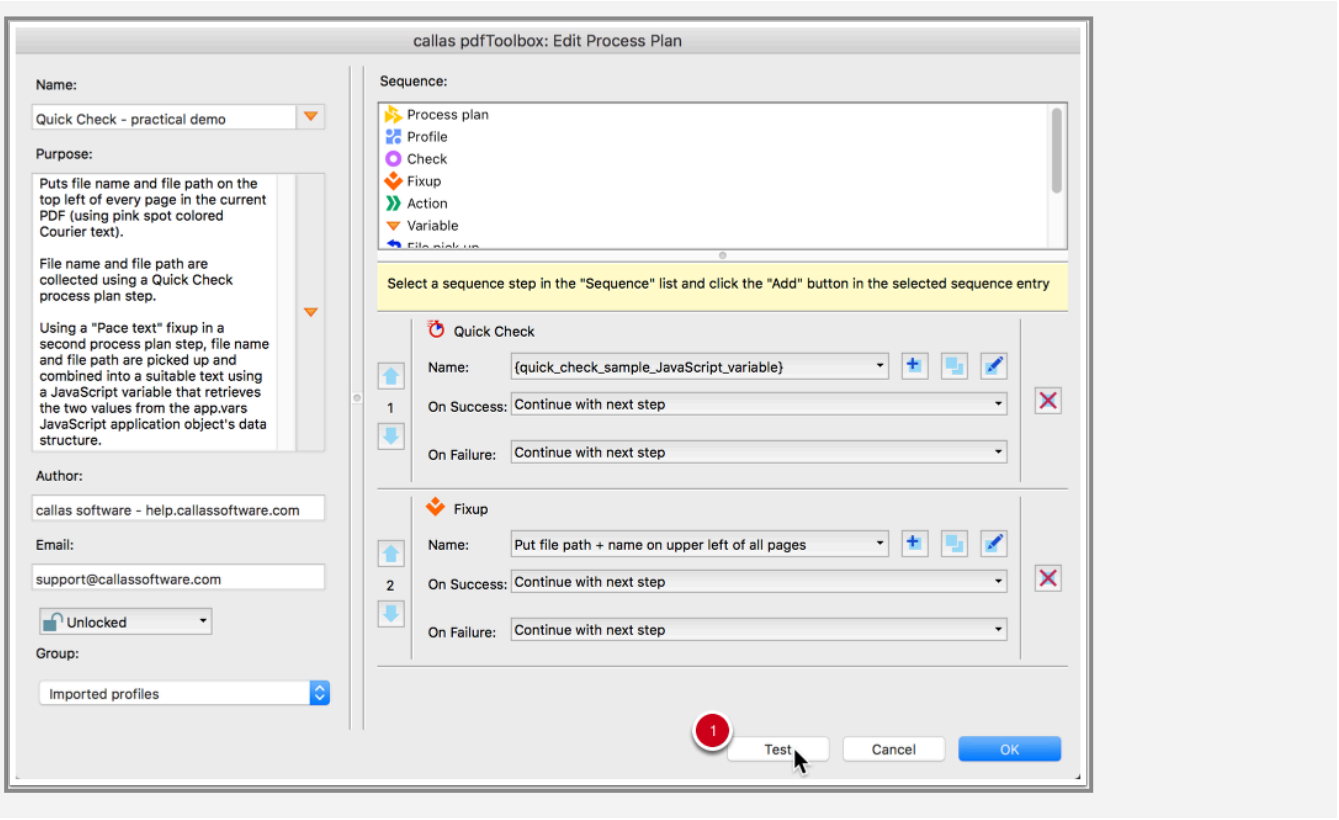


After closing this "Edit Variable" window, and then also the "Edit Fixup" window, the "Edit Process Plan" window will be shown again. Instead of also closing this window and trying the Process Plan on one of your files,

1. use the "Test" feature by clicking on the "Test" button

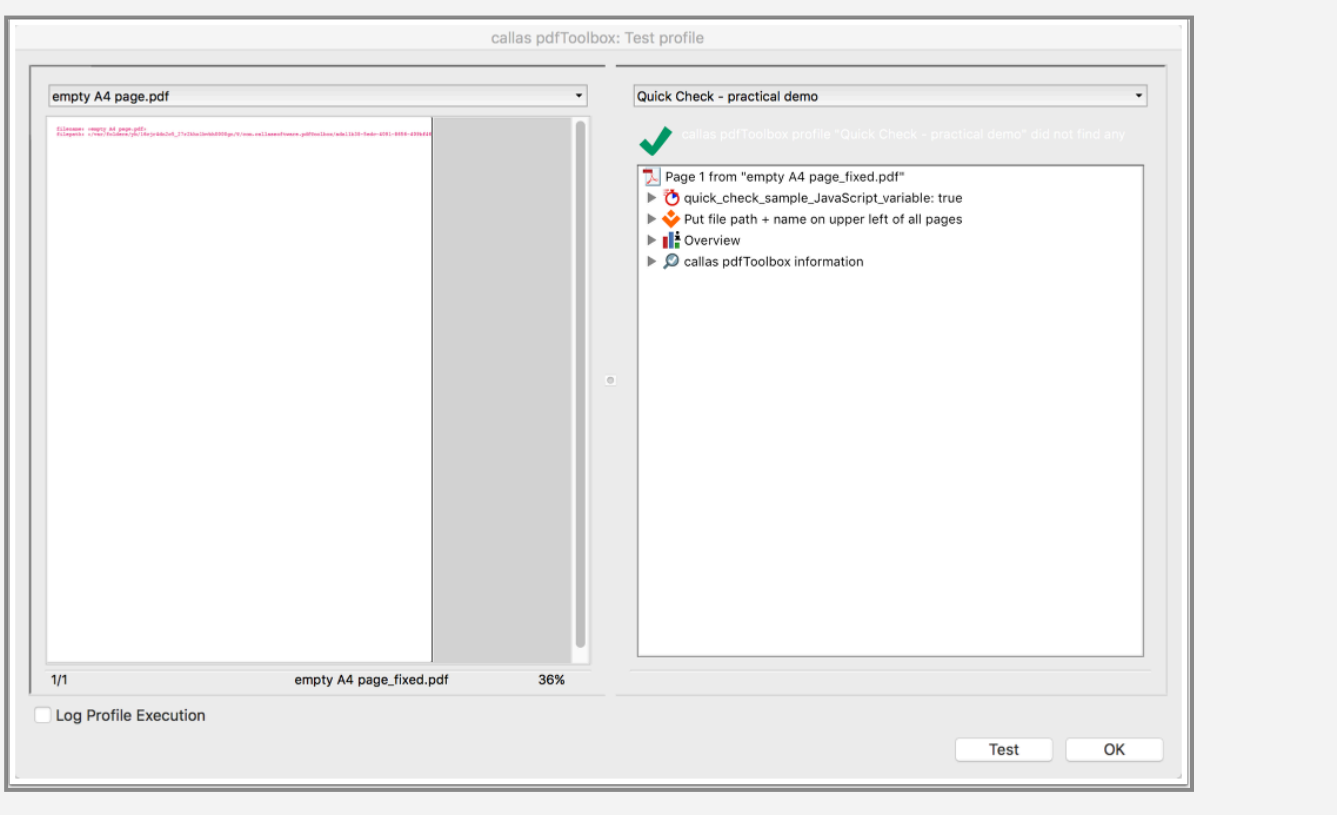


Alternate image for the older Process Plan UI:



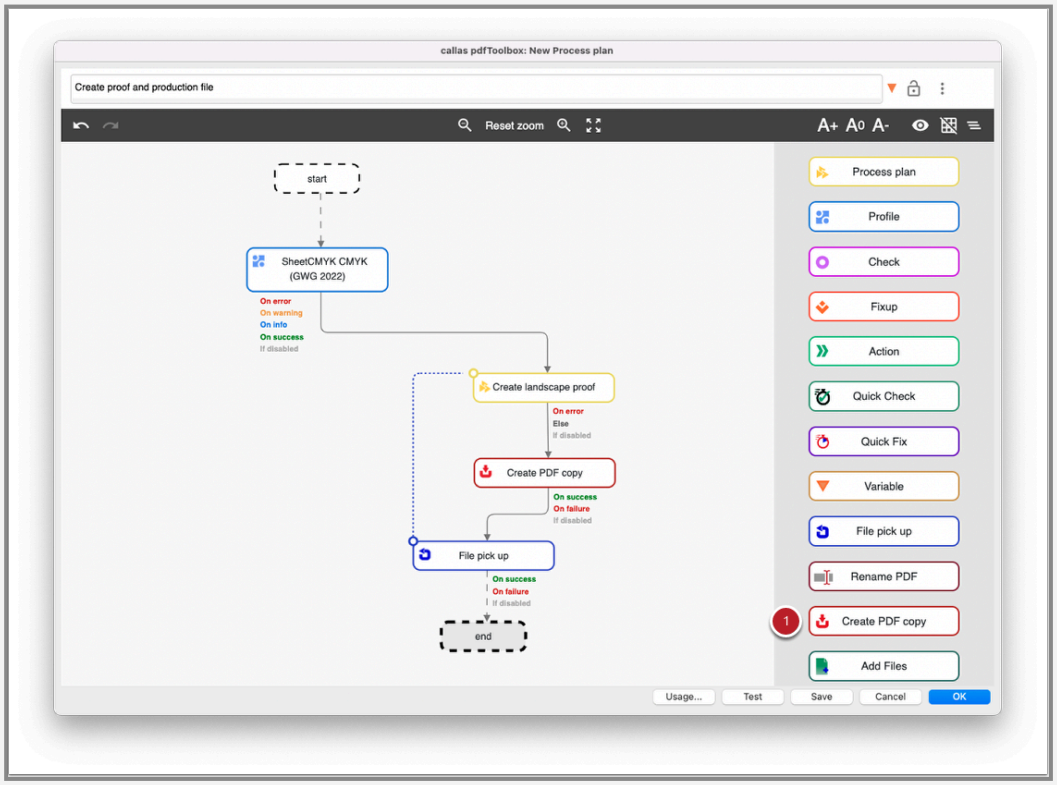
This will open the "Test profile" window – a very convenient environment to try Process Plans (or Profiles, Fixups and Checks) on a copy of a currently open PDF file without the risk of inadvertently breaking your files ("Test" processing is always carried out on temporary copies of your PDF files), and without tedious open/edit/close/apply cycles, and without any need to clean up processed copies of your PDF files.

As you can see in the screen shot some text is showing up in the upper left of the sample file used here (a PDF consisting of an empty A 4 sized page, and named accordingly).

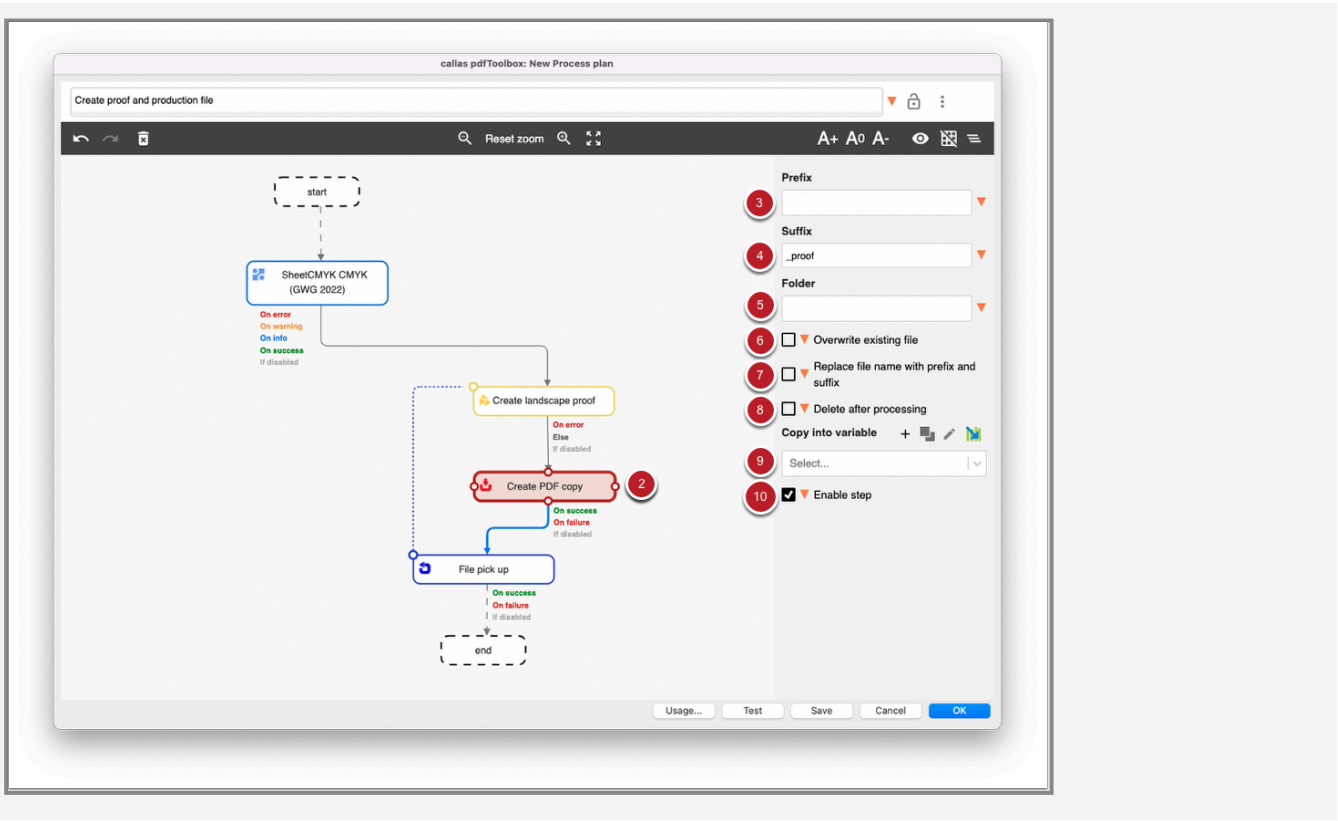

callas pdfToolbox: Test profile
empty A4 page.pdf
Quick Check - practical demo
callas pdfToolbox profile "Quick Check - practical demo" did not find any
Page 1 from "empty A4 page_fixed.pdf"
quick_check_sample_JavaScript_variable: true
Put file path + name on upper left of all pages
Overview
callas pdfToolbox information
1/1
empty A4 page_fixed.pdf
36%
Log Profile Execution
Test
OK

3.4 Process Plan step "Create PDF copy"

pdfToolbox 10 onwards, you can add a new Process Plan step to generate additional PDF output files from a Process Plan. Where before, a single Process Plan would always have a single PDF output file, using this new "Create PDF copy" step, a single Process Plan can now save additional PDF output files. See how it works in pdfToolbox 11 (go below for pdfToolbox 10 UI)



1. The Process Plan step is called "Create PDF copy". As with all other steps, the step can be added via the list with the sequence steps on the right using Drag&Drop.



2. After dragging and dropping the step, it is included in the Process Plan, as shown in the figure. By default, this step generates an additional PDF file with the same name as the main output file, saved in the same folder.

For "Create PDF copy" a number of properties are provided in the column on the right that can be set.

3. The "Prefix" property allows you to optionally specify a prefix for the name of the PDF document created by this step.

4. The "Suffix" property allows you to optionally specify a suffix for the name of the PDF document generated by this step.

5. The "Folder" property allows you to specify an optional folder where the PDF copy will be saved. This folder is created at the same level as the main output file.

Example: If `copy` is specified under "Folder", a new folder named "copy" will be created next to the output PDF file during Process Plan execution and the PDF copy will be saved in this folder.
Note: It is not possible to specify a complete file path, e.g. `c:\temp\copy`, but only a relative folder at the level of the output file.

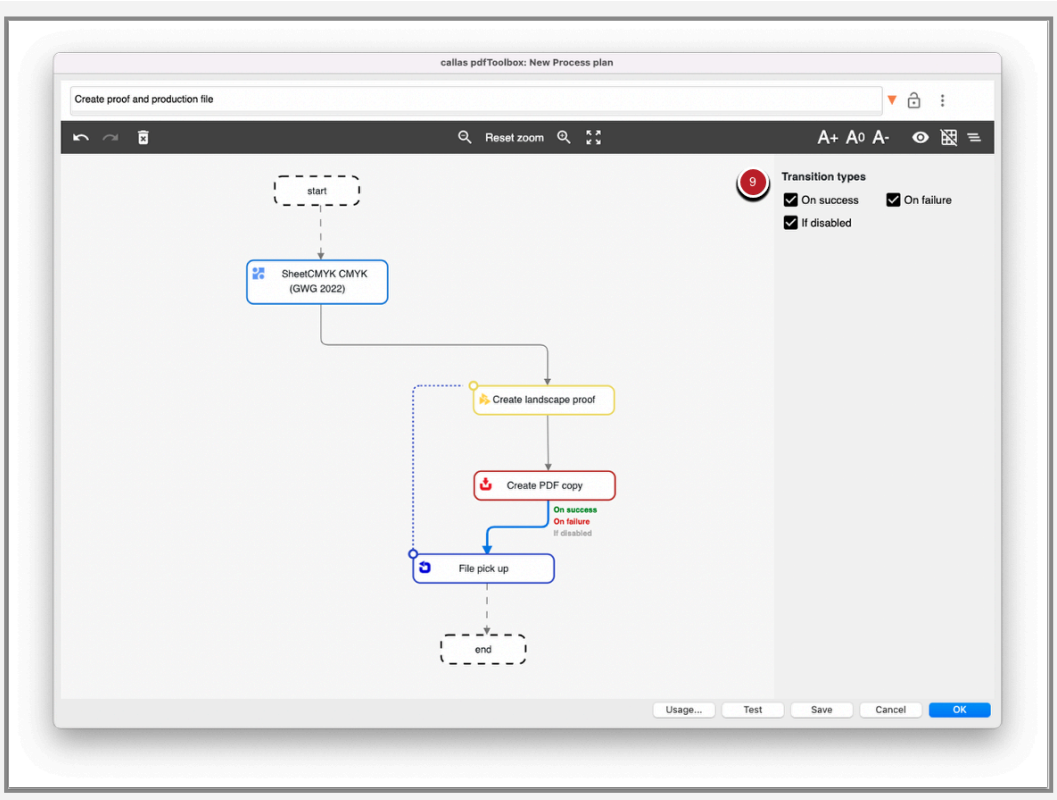
6. The "Overwrite existing file" checkbox allows existing files to be overwritten by files of the same name created in this step. Variables are possible for this (orange triangle).

7. The option "Replace file name with prefix and suffix" allows changes to the file name. Variables are possible for this (orange triangle).

8. "Delete after processing" allows the creation of intermediate results, ensuring that the generated copies are automatically deleted once the Process Plan execution is complete.

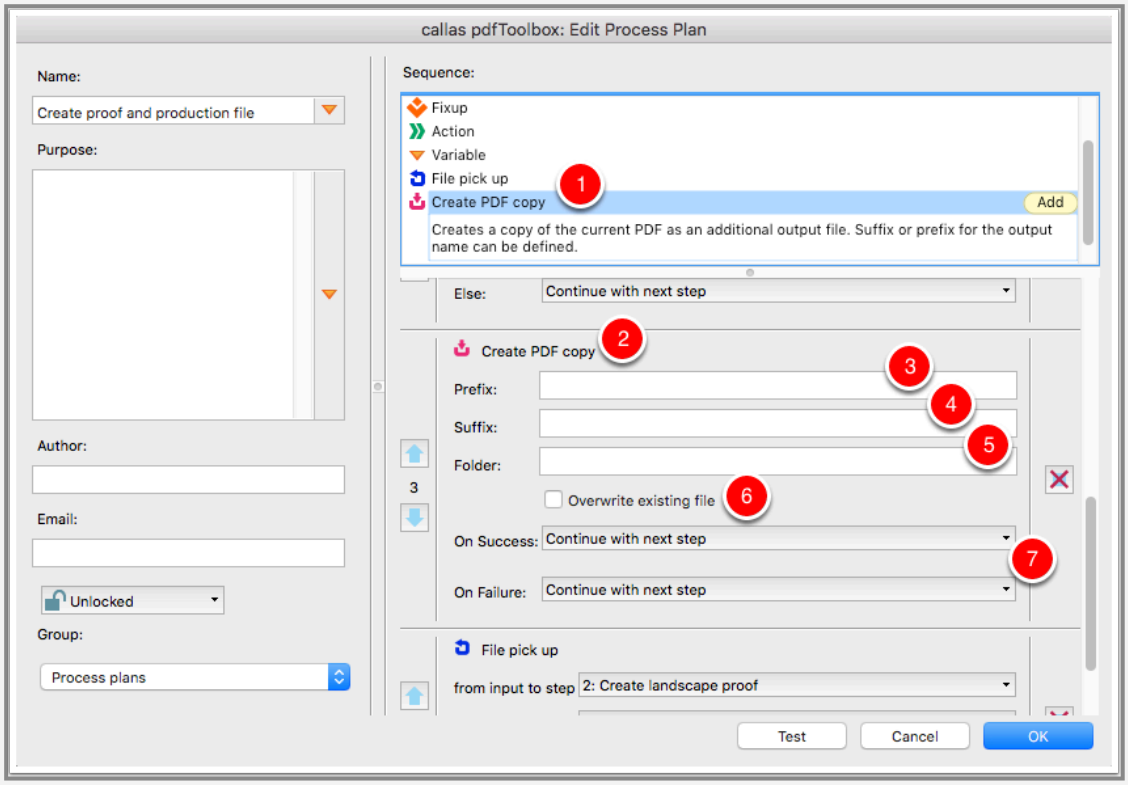
9. **Available since pdfToolbox 16:** The "Copy into variable" option allows you to store the file path of the generated PDF copy in a simple variable. This makes it easy to access the file copy in later steps of the Process Plan, such as a FilePickup step.

10. Finally, the step in this category can be activated or deactivated.



9. By clicking on the connection type, you can define the conditions under which the next step will be executed. The available options are: "On success," "On failure," and "If disabled."

Alternatively in pdfToolbox 10 or lower



1. The Process Plan step is called "Create PDF copy". As all other steps, double-clicking or clicking the "Add" button the right of the name, inserts the step at the end of your Process Plan.
2. This is how this new step looks in a Process Plan. By default, this step outputs an additional PDF file with the same name as the regular result of the complete Process Plan and next to it (in the same folder as where the regular output file is generated).
3. This property allows specifying an optional prefix that will be added to the name of the PDF file generated by this step.
4. This property allows specifying an optional suffix that will be added to the name of the PDF file generated by this step.
5. This property allows specifying an optional folder for the output file generated by this step. If used, a folder with the specified name will be created and the output file generated by this step will be saved into this folder.

6. Select this checkbox to overwrite any file with the same name as the file generated by this step.
7. These options allows setting up what needs to happen next based on success or failure of this step.

Remarks

- The Process Plan always generates an output PDF files, whether or not you use such a step. Using this step simply generates an *extra* output PDF file.
- You can use this step multiple times in the same Process Plan if you need to generate multiple output files at different stages of your Process Plan.

3.5 Process Plan step "File pick up"

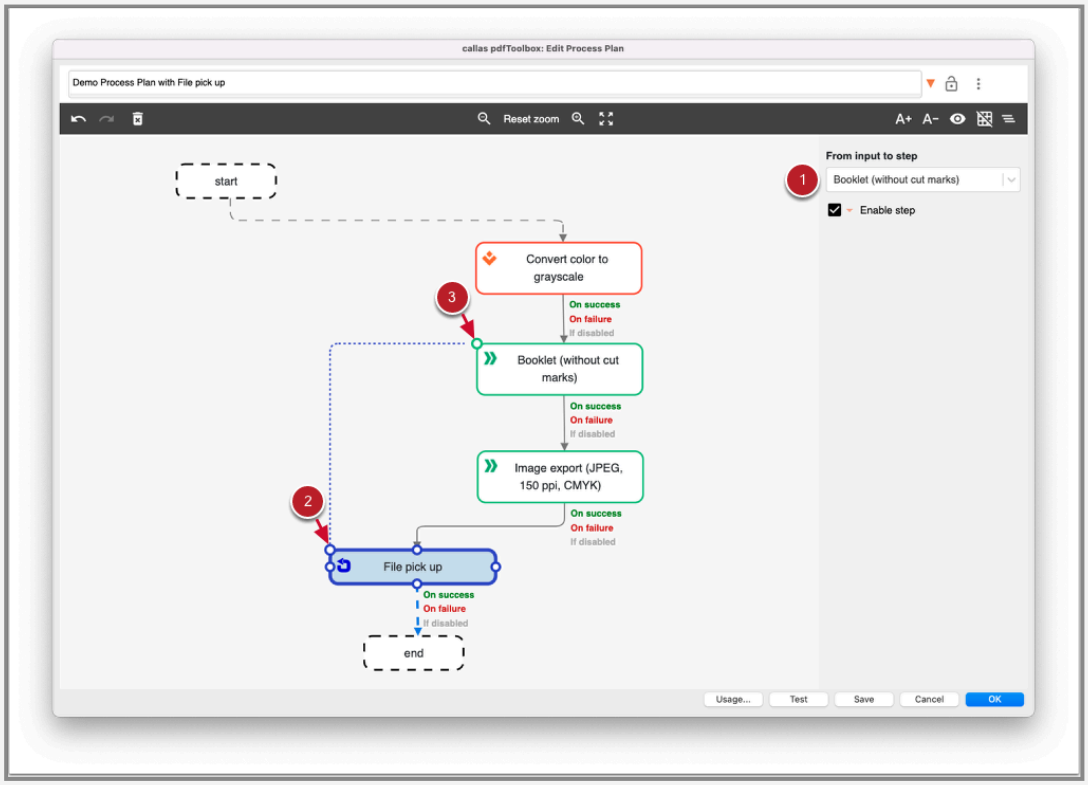
The "File pick up" step within a Process Plan allows you to revert to a previous version of a PDF file that was used as input in a previous step. This replaces the current PDF file with the initial input PDF file from a specified previous step within the Process Plan. The purpose of the "File pick up" step is to rewind to a specific state of a PDF document as it existed at an earlier stage in the process.

How to insert a "File pick up" step into a Process Plan

Select the "File pick up" sequence step from the right column and drag it to the desired position in the Process Plan (when the connection line is blue, the step can be added). You can use this step multiple times in the same Process Plan.

To return to an earlier state of the PDF file, the File pick up step needs to be connected with a previous Process Plan step. To connect the steps, you have two options:

- Select a corresponding sequence step in the dropdown menue on the right (1) (a blue dashed connection line will appear).
- Drag a connection line from the upper left corner of the "File pick up" step (2) to the circle in the upper left corner of the corresponding Process Plan step (3).



This Process Plan illustrates how the “File pick up” works:

1. Convert color to grayscale: The PDF document is converted to grayscale.
2. Booklet (without cut marks): Prepares the PDF document for double sided printing.
3. Image export (JPEG, 150 ppi): Exports the booklet as a low-resolution JPEG.
4. File Pick up: Since the "File pick up" step is associated with the "Booklet" Action (as indicated by the blue dashed line), the current PDF file, which is a grayscale imposed booklet at this stage, will be replaced by the input PDF file before the booklet Action which will result in the input PDF file that was converted to grayscale.

Different return codes between pdfToolbox 14 and pdfToolbox 15

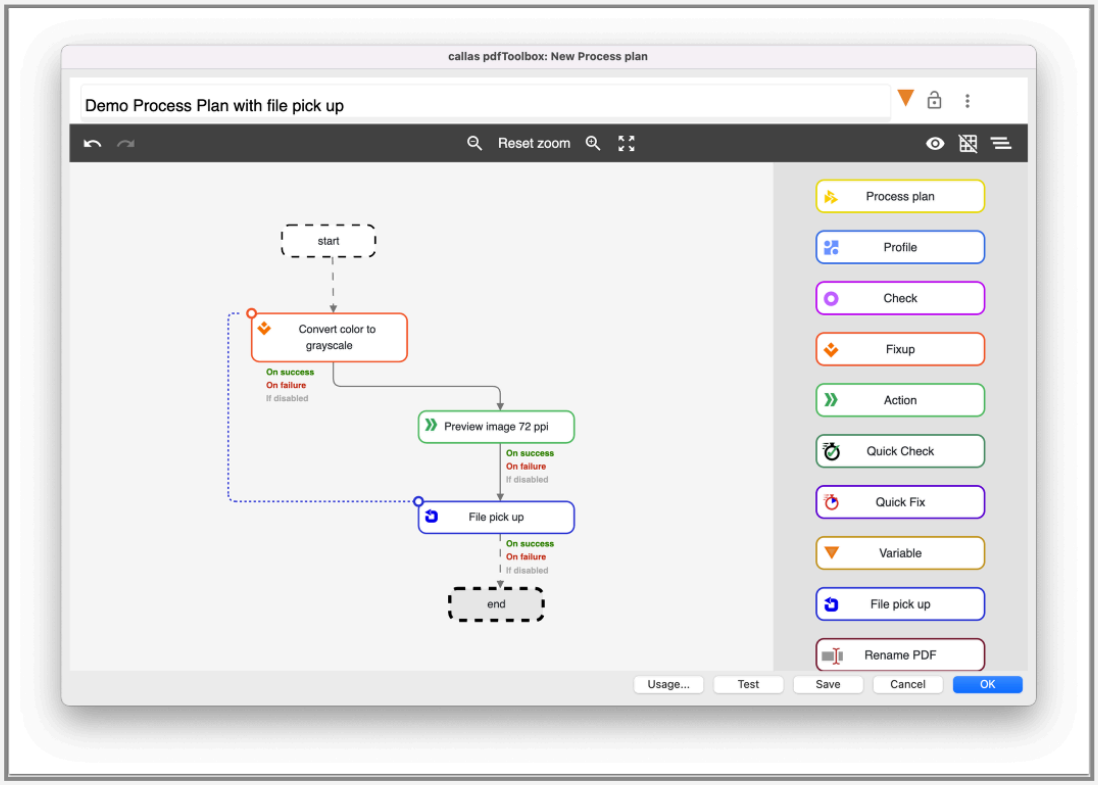
WARNING: Different return codes between pdfToolbox 14 and pdfToolbox 15!

Since pdfToolbox 15, the "File pick up" step not only restores the original file, but also the status at the selected pick-up time. This can lead to different return codes when using pdfToolbox 14 and pdfToolbox 15 on the CLI.

Example for the Process Plan shown below:

The "File pick up" step is associated with the first sequence step of the Process Plan, which has the same effect as returning to the original PDF file from which the Process Plan was started.

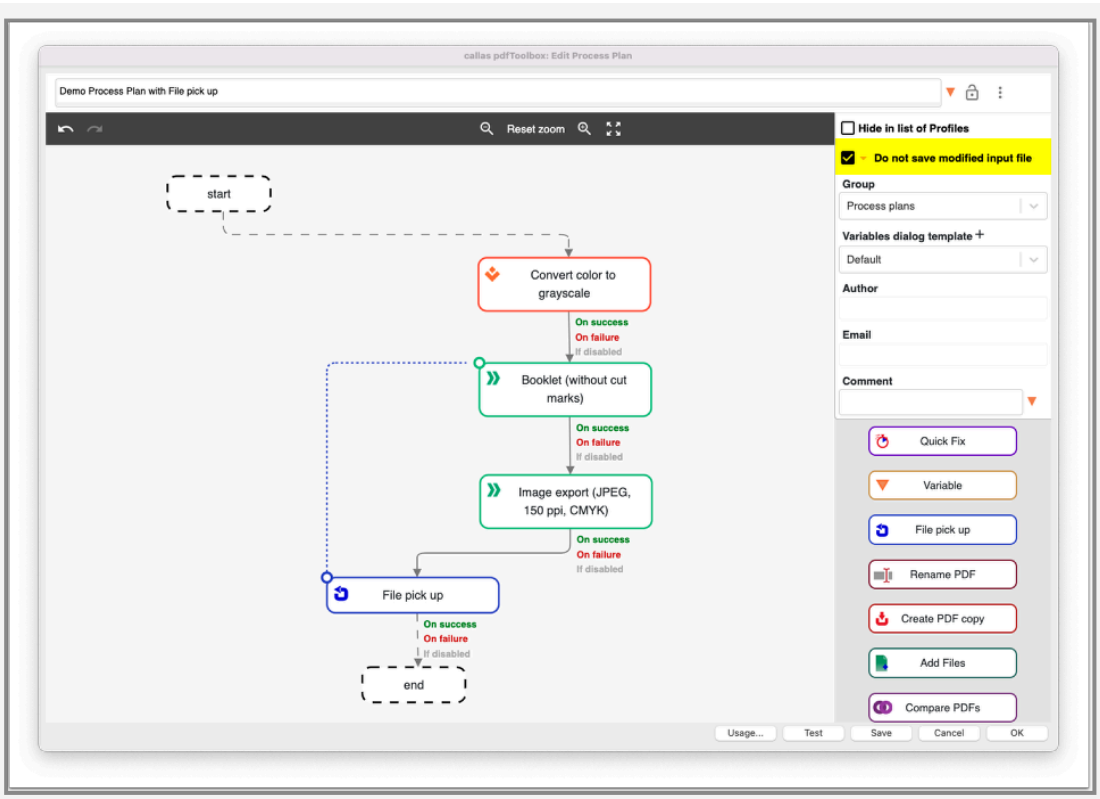
pdfToolbox 14: Return code on CLI after executing the Process Plan: **5** (No hit, Fixups have been executed)
pdfToolbox 15: Return code on CLI after executing the Process Plan : **0** (No hit, no Fixup executed)



Do not save modified input file

With pdfToolbox 15, a new checkbox has been introduced within Process Plans that allows you to bypass saving the modified input file. This checkbox is useful if the modifications made to the PDF file within the process plan are not required as a final output. When this option is enabled and a process plan is executed, the "Do not save modified input file" option acts internally as a file pickup step that references the first step of the process plan. This means that if the checkbox is checked, pdfToolbox will always restore the original input file as a result.

Therefore it's important to enable this checkbox only when you are certain that you do not need to retain the results of the Process Plan.



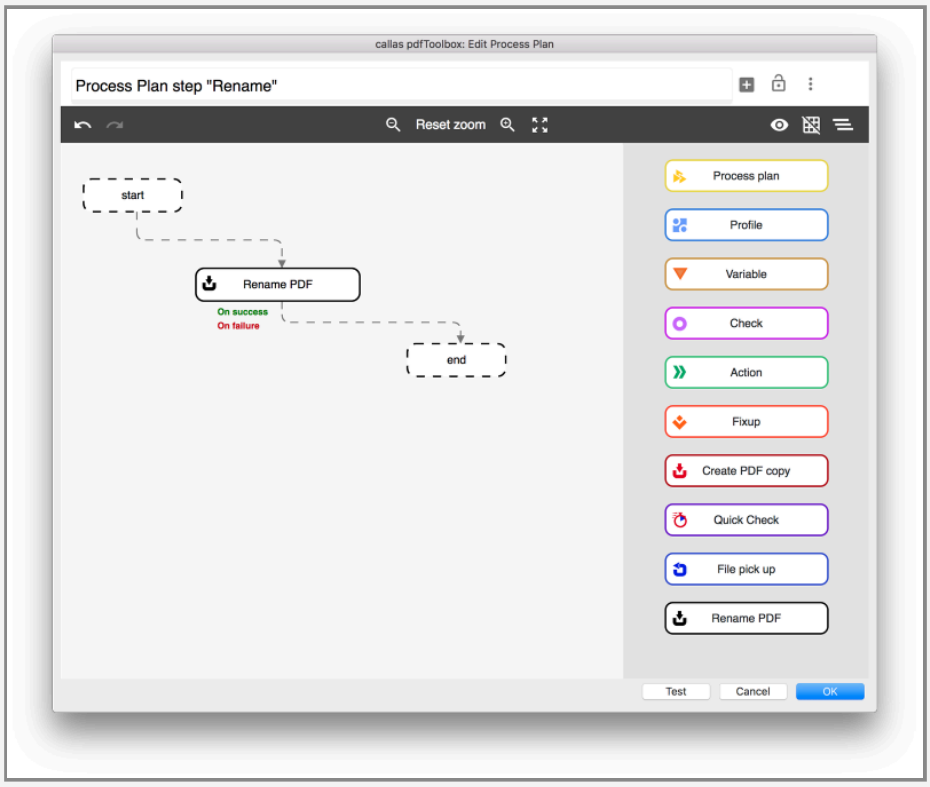
For better understanding, here is an example based on the above process map:

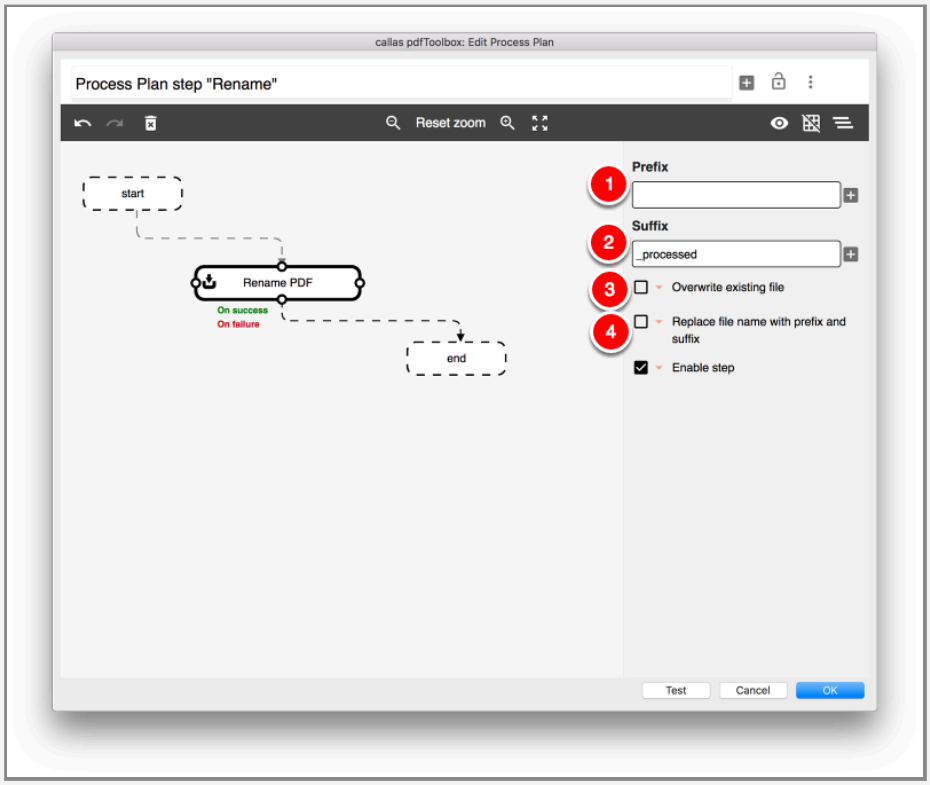
1. **"Do not save modified input file" is unchecked**: Because of the File Pickup step at the end, the output PDF is converted to grayscale, but not imposed into a booklet.

2. **"Do not save modified input file" is enabled**: The original input file will be restored (the result will not be converted to grayscale or imposed into a booklet).

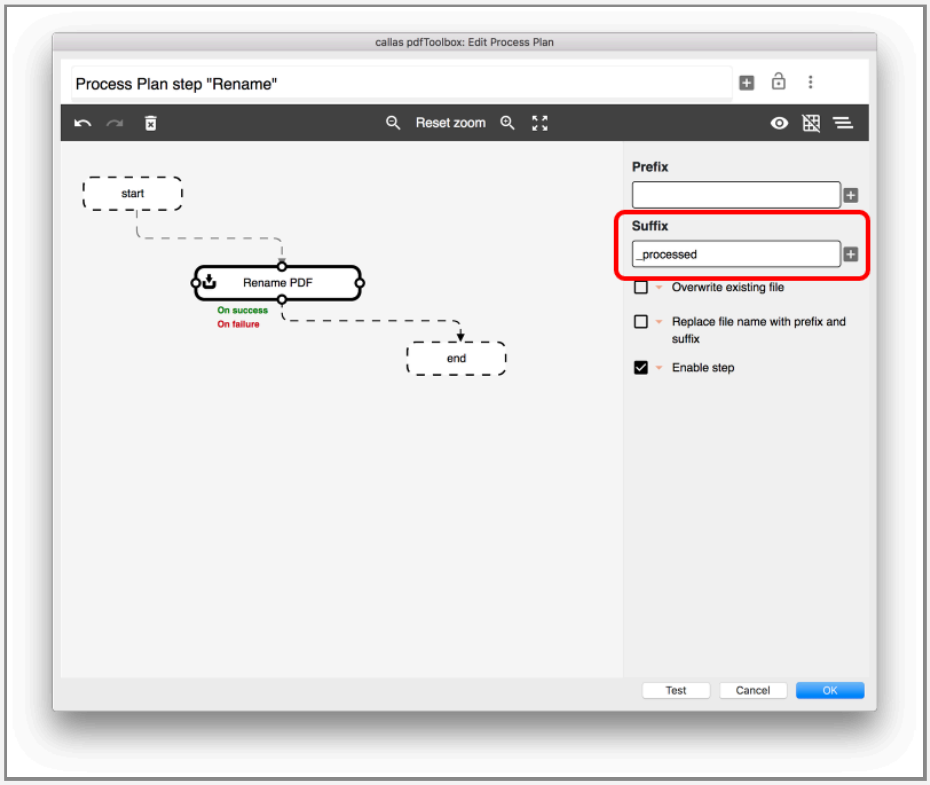
3.6 Process Plan step "Rename"

pdfToolbox 11 introduced a new Process plan step 'Rename' which, as the name suggests, renames the current PDF in the Process plan workflow.



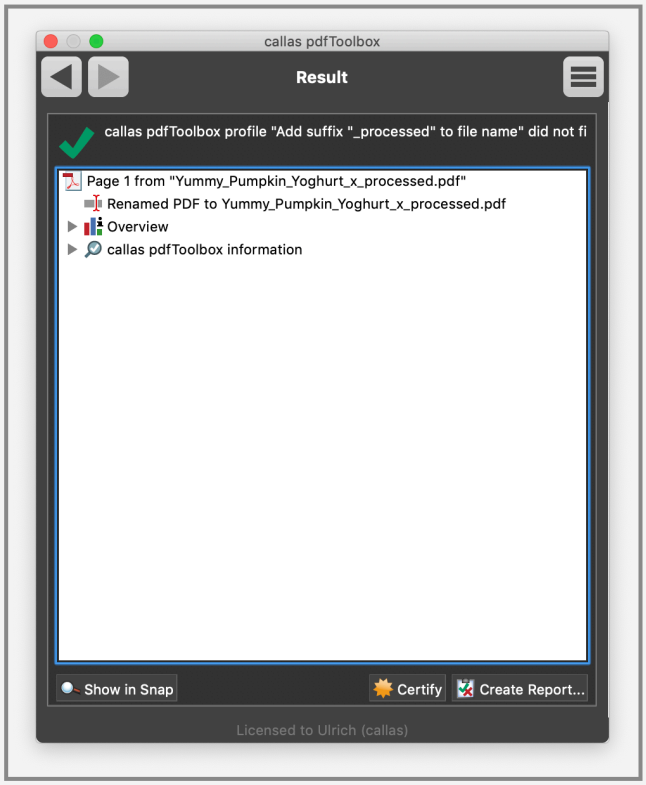


1. Prefix: A defined Prefix is prepended to the PDF file name
2. Suffix: A defined Suffix is appended to the PDF file name
3. Replace file name with Prefix and suffix: If set, the current file name is not included in the new file name, i.e. the new file name will be Prefix + Suffix
4. Overwrite existing file: If set, the destination file is over-written, if it exists



So a setting like the one above would add "_processed" behind the file name, but before the file type suffix.

Using a file called "Yummy_Pumpkin_Yoghurt_x.pdf" would result into "Yummy_Pumpkin_Yoghurt_x_processed.pdf" like on the screenshot of the result dialog below:



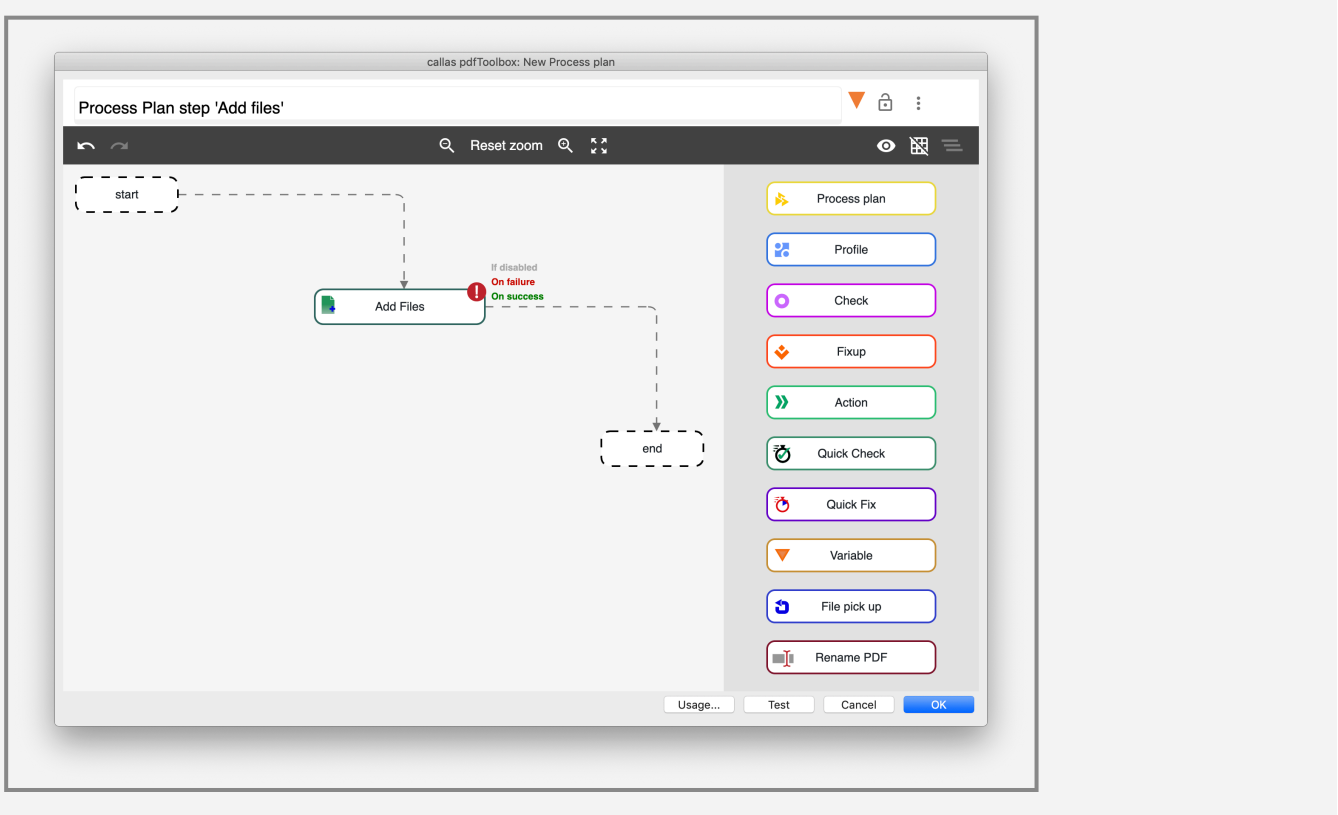
Our CTO Ulrich Frotscher talks about 'Rename' step in Process Plans:

3.7 Process Plan step "Add files"

The Process Plan step 'Add files' supports three different use cases:

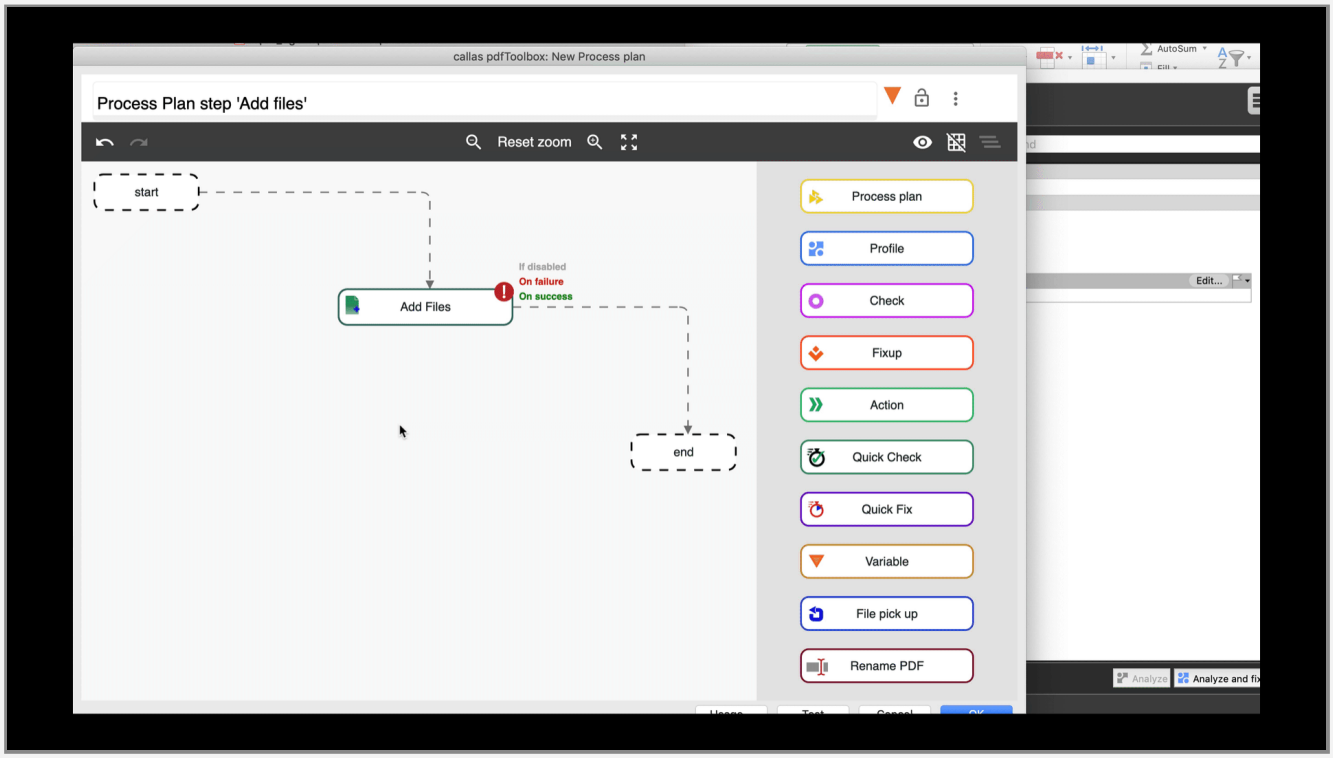
- **Merge PDF Files**: Allows you to add additional pages to an existing PDF within the Process Plan workflow.
- **Embed Files**: Enables embedding external files into the PDF being processed (e.g., for archival workflows).
- **Replace File**: Replaces the current PDF with a different PDF, making it the active file in the workflow.

Earliest version with full support for the "Replace file" option is pdfToolbox 16

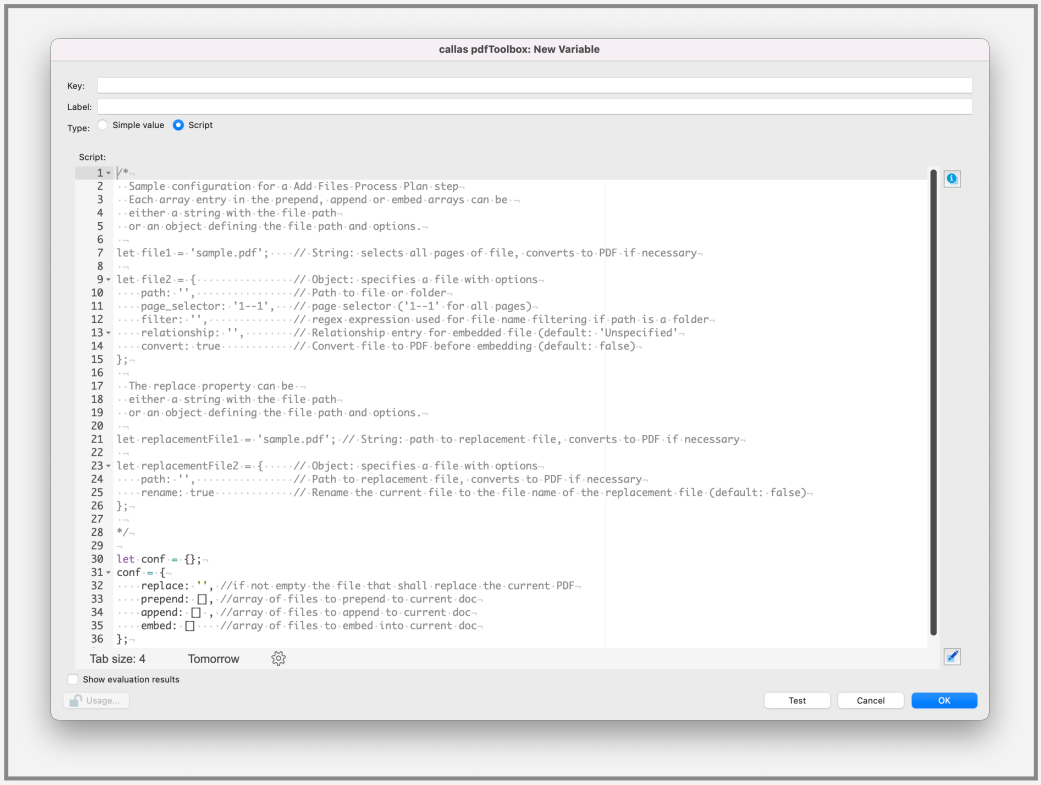


The Process Plan itself is configured using a script:

- After drag&dropping the step onto the canvas, click on the step
- Click the '+' button in the right side panel to start working on the configuration script



The sample script looks like below:



Example script: Add a cover page

Below is a sample script to add a cover page (1 pager) to the front of a PDF file:

```
const conf = {
    prepend: [], //array of files to prepend to current doc
    append: [] , //array of files to append to current doc
    embed: []    //array of files to embed into current doc
};

const CoverPath = '/Users/admin/Downloads/callas_cover_Part1.pdf';

conf.prepend.push({
        path: CoverPath,
        first_page:0,
        num_pages:1
});

conf;
```

Click to copy

Fancy a video about 'Add files' Process Plan step?

Replace PDF file

The `replace` property can be either:

A **string** that specifies the full path to the document that will replace the current file.

```
let conf = {};
conf = {
        replace: "/Users/Desktop/AddFiles/test.pdf"
};
```

Click to copy

An **object** that defines the full path to the replacement document and offers the option to rename the file based on the name of the replacement file.

```
let conf = {};
conf = {
        replace:
        {
                path: "/Users/Desktop/AddFiles/test.pdf",
                rename: true
        }
};
```

Click to copy

Example: Replace current file with result of previous Process Plan step

Below is a downloadable Process Plan that replaces the current PDF file with the result file from the previous process plan step:

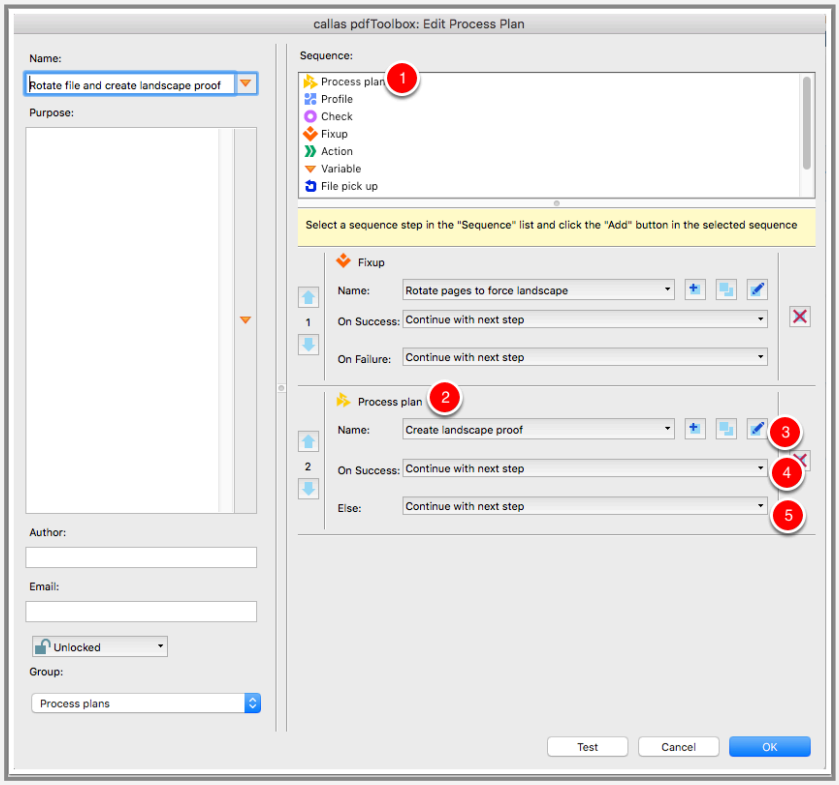


Replace current PDF with topographic ink coverage view.kfpx

(earliest version with full support is **pdfToolbox 16**)

3.8 Using Process Plans in Process Plans

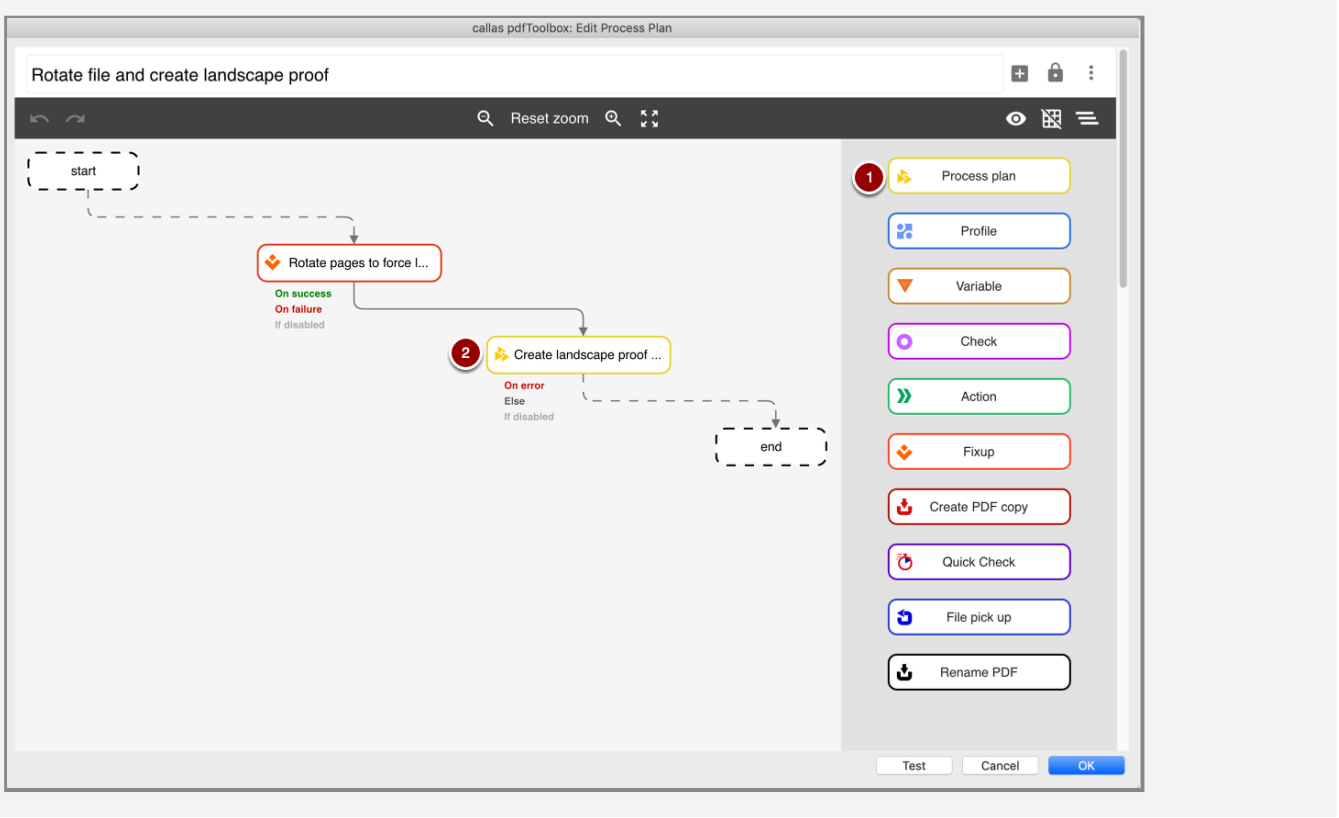
pdfToolbox 10 added a new type of step to Process Plans to allow running a Process Plan inside of another Process Plan.



1. Double-clicking "Process Plan" or clicking the "add" button next to it, adds a Process Plan step to the Process Plan you're editing.
2. A Process Plan step is identified by the yellow Process Plan icon and its title.
3. The "Name" property has a pull - down list showing all Process Plans in the current library. This allows you to select the Process Plan you want to use. The buttons on the right make it easy to create a new Process Plan, duplicate the currently showing Process Plan or edit the currently showing Process Plan.
4. If the result after running this Process Plan step is "success", this options allows you to choose what needs to happen next. This is similar to all other Process Plan steps.

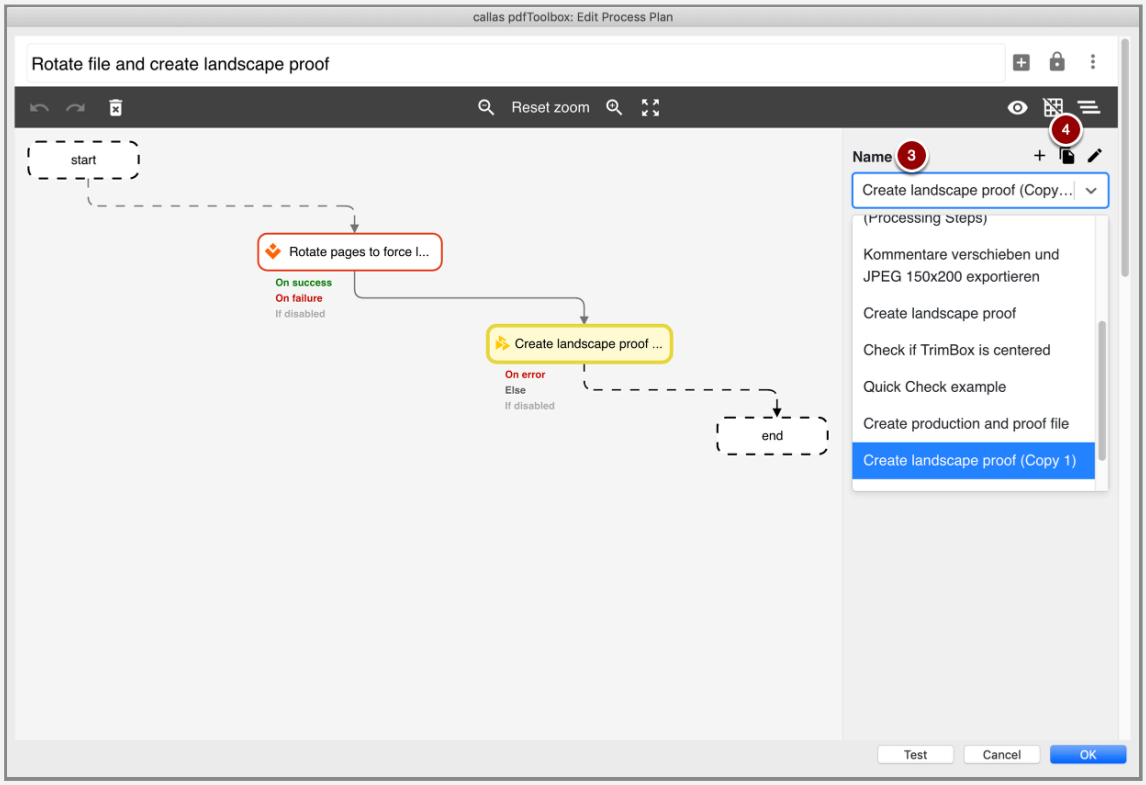
5. If the result after running this Process Plan step is not "success", this options allows you to choose what needs to happen next. This is similar to all other Process Plan steps.

Alternatively in pdfToolbox 11



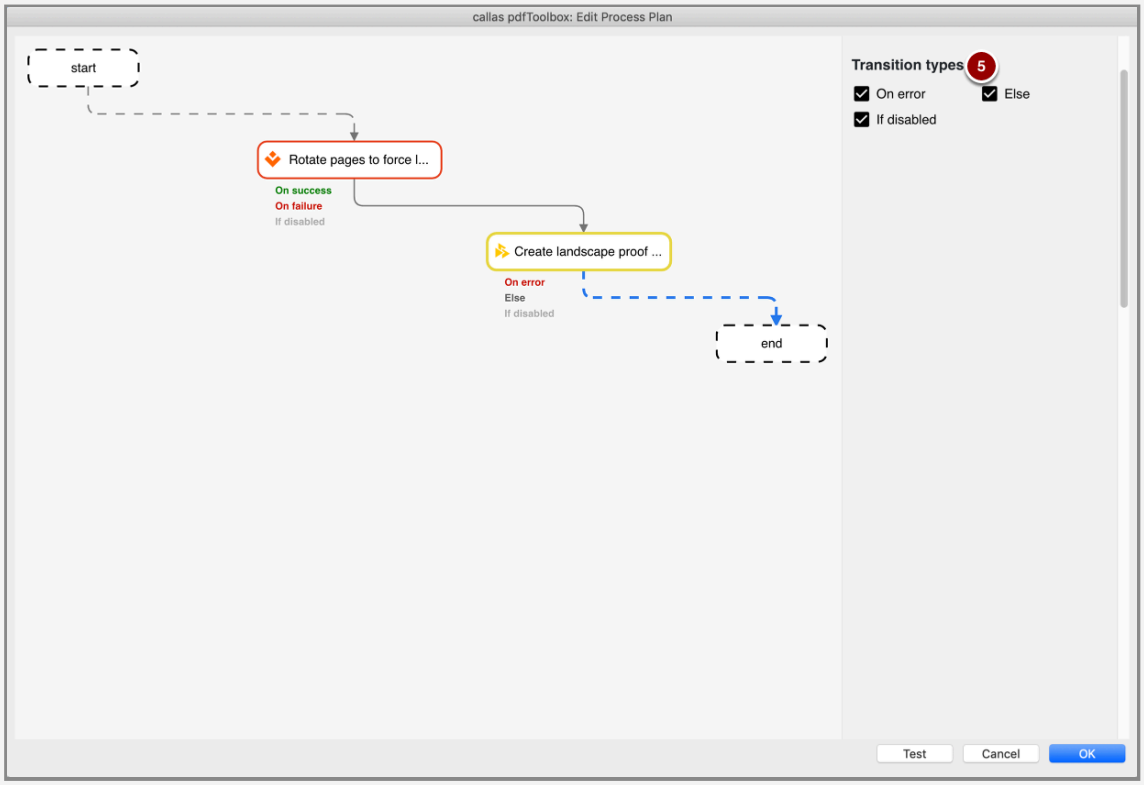
1. The type "Process plan" is also listed in the list of sequence steps. This step "Process Plan" can be added via the list with the sequence steps on the right by drag&drop.

2. A process plan step can be recognized by the yellow process plan icon.



3. The Name property has a pop-up menu that displays the list of all process plans that comprise the selected library. This allows you to select the Process plan you want to use.

4. The buttons above allow you to easily create a new Process plan, duplicate the currently displayed Process plan, or edit the currently displayed Process plan.



5. After the Process plan is set up, you can set the connection types to determine what is to happen afterwards. This is similar to other Process plan steps.

Remarks

- This feature allows you to put frequently used steps into their own Process Plan and re-use them easily in other Process Plans.
- Take care not to create infinite loops.

3.9 Using Process Plans in Process Plans to loop over pages

Why looping over pages?

Normally Fixups and Checks work just fine for multi-page PDFs. That is even the case if pages are very different, e.g. when their page sizes differ. All Fixups, where that is relevant, can be set up to work according to the current page size (page geometry box size). However, that is not always the case. E.g. if you want to add different objects to each page or if you want to use the current Trim Box in JavaScript you may want to process each page individually. That was aways possible in Process Plans using JavaScript, but it becomes much easier with pdfToolbox 16.

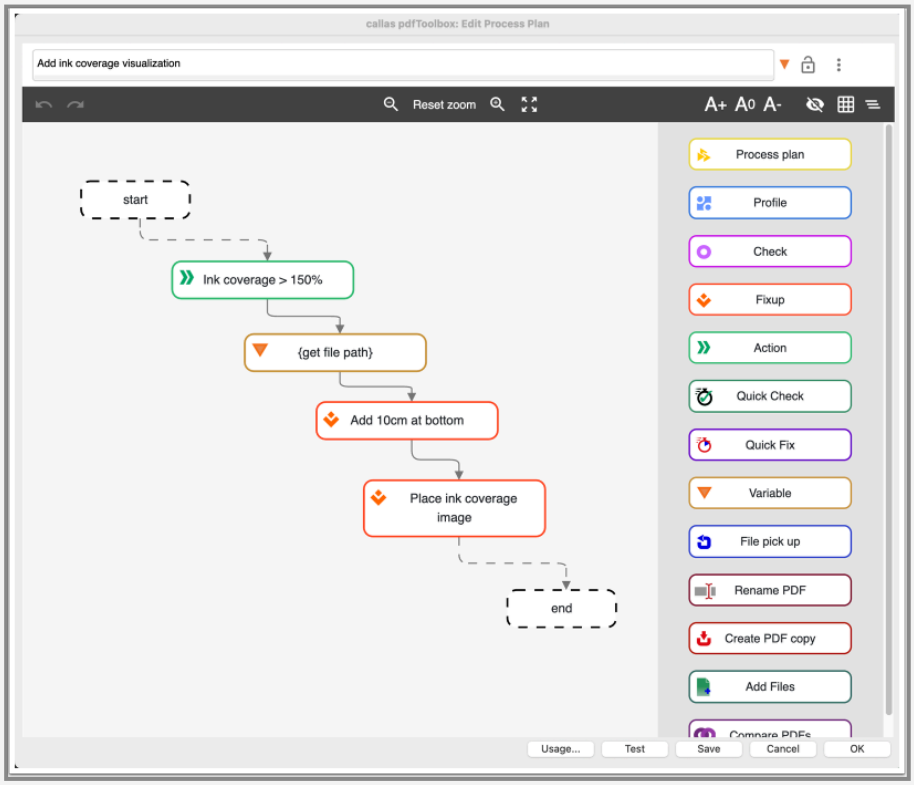
Example

As an example we want to place an ink amount visualization on each page of a PDF file. The Process Plan below does that for a single page in a PDF file. A variable is used to specify that page.



Add ink coverage visualization.kfpx

(earliest version with full support is **pdfToolbox 16**)



The Process Plan creates an ink coverage visualization for the specified page, gets the file path from the JavaScript object, adds 10cm at the bottom of the specified page and places the ink amount visualization in that area.

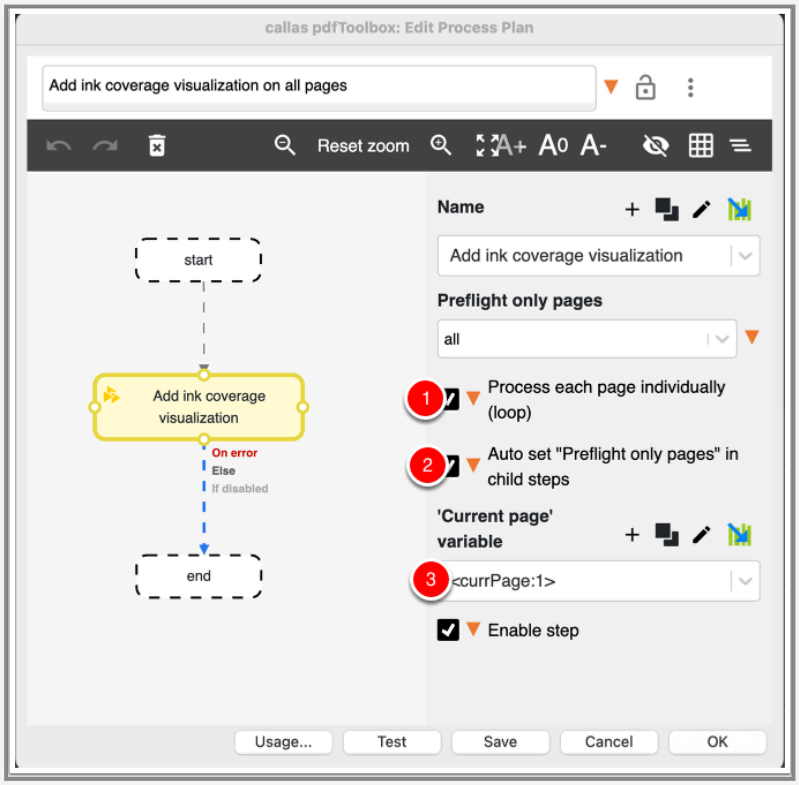
It would not be impossible to do this for all pages of a document using the "Place content" technology, but it would be a complex task to create a proper template.

Applying a Process Plan to each page individually using a parent Process Plan

It is much easier but a little slower to do that using a parent Process Plan.

Add ink coverage visualization on all pages.kfpx

(earliest version with full support is **pdfToolbox 16**)



In the parent Process Plan there is a "Process each page individually (loop)" checkbox (1). If that is active the PDF is provided to the child Process Plan several times, depending on the number of pages in it. That of course slows down processing, but allows you to process the PDF page by page achieving things that would otherwise be much more complex.

In the child Process Plan you then have to make sure that only the current page is processed. There are two tools that make that more convenient:

With (2) 'Auto set "Preflight only pages" in child steps' the parent Process Plan automatically sets all steps in the child Process Plan that have a "Preflight only pages" input field to use a variable for the current page. This input field is available for all Check, Fixup and Profile steps in the child Process Plan.

With (3) you can specify a variable that is set to the current page number.

In our example, we are using the variable "currPage" to tell the Action in the child Process Plan that creates the ink amount visualization which page to process. We use the

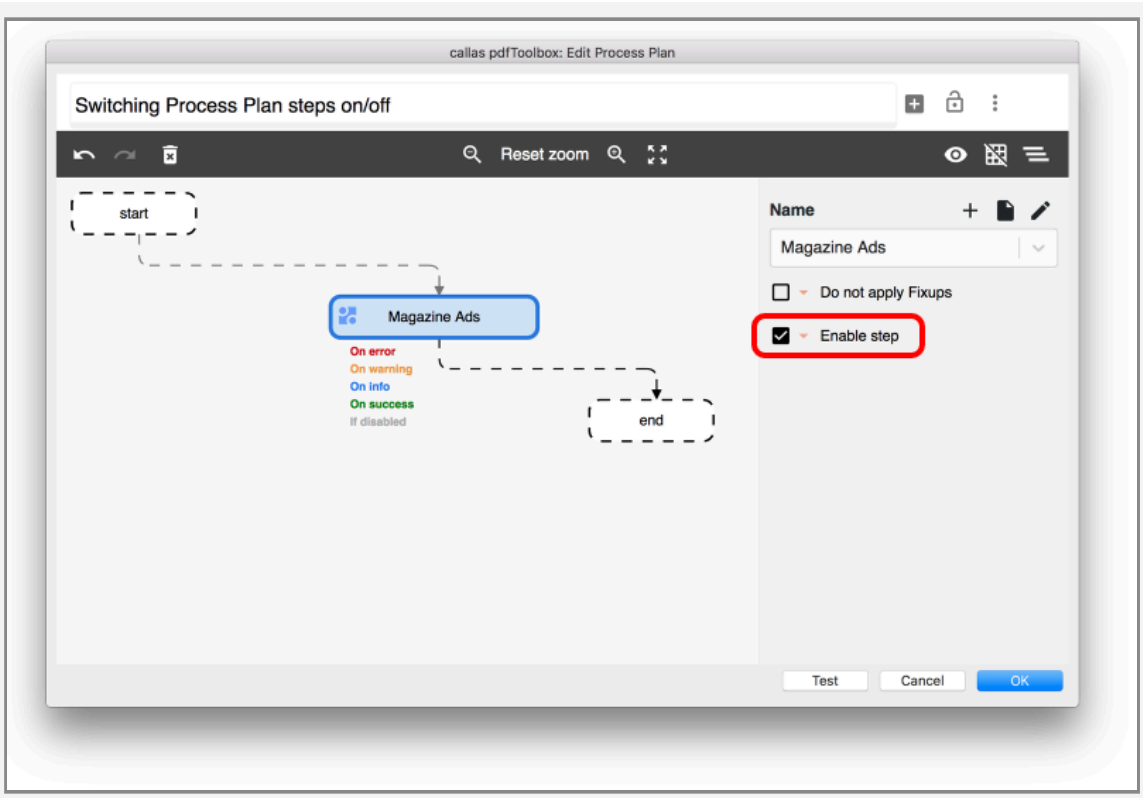
same variable for the two Fixups in there, although that would not be necessary with the 'Auto set "Preflight only pages" in child steps' checkbox. But this way we can also use the child process plan directly.



One thing to note is that the "'Current page' variable" is 1 based while the page geometry boxes array in JavaScript is 0 based. So, if you want to use the current page variable in the JavaScript of the child Process Plan you will have to subtract 1.

3.10 Switch on/off a Process Plan step via a Variable

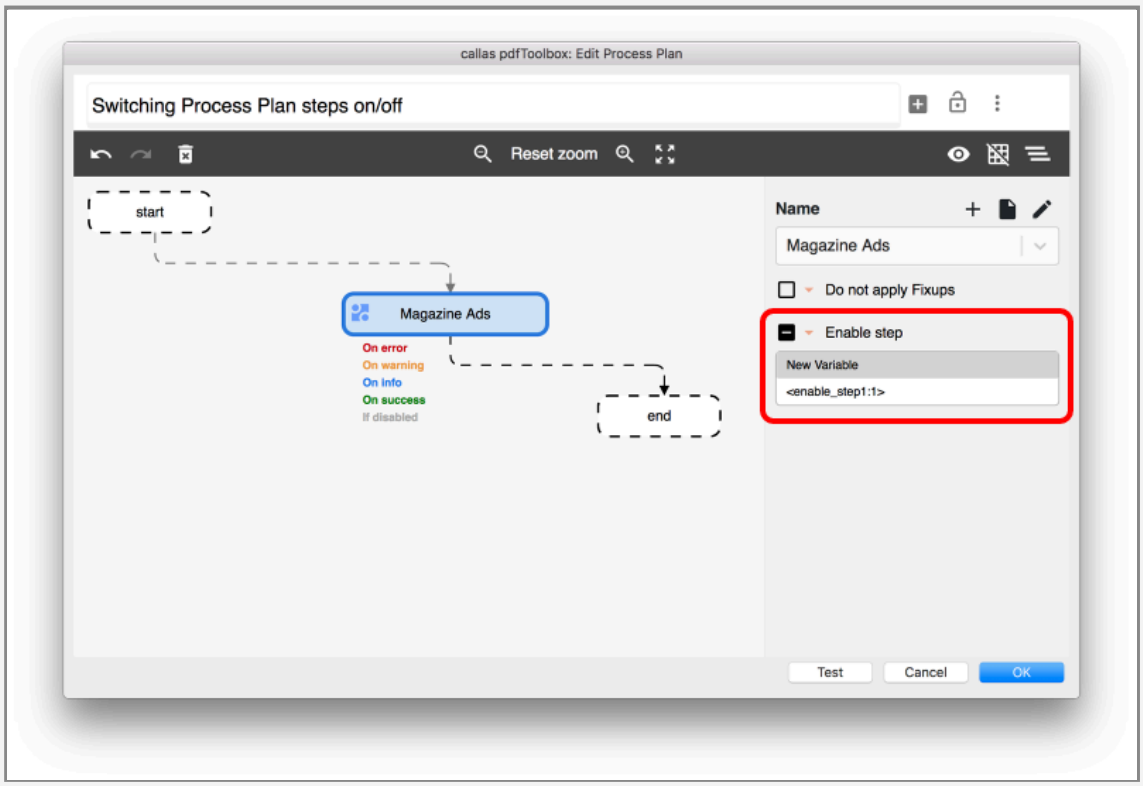
As of pdfToolbox 11, you can switch on/off a Process Plan step using a Variable or a simple checkbox. Instead of adding an additional Variable as a Process Plan step to manipulate the steps, it can be achieved using the variable inside the Process Plan step for convenience.



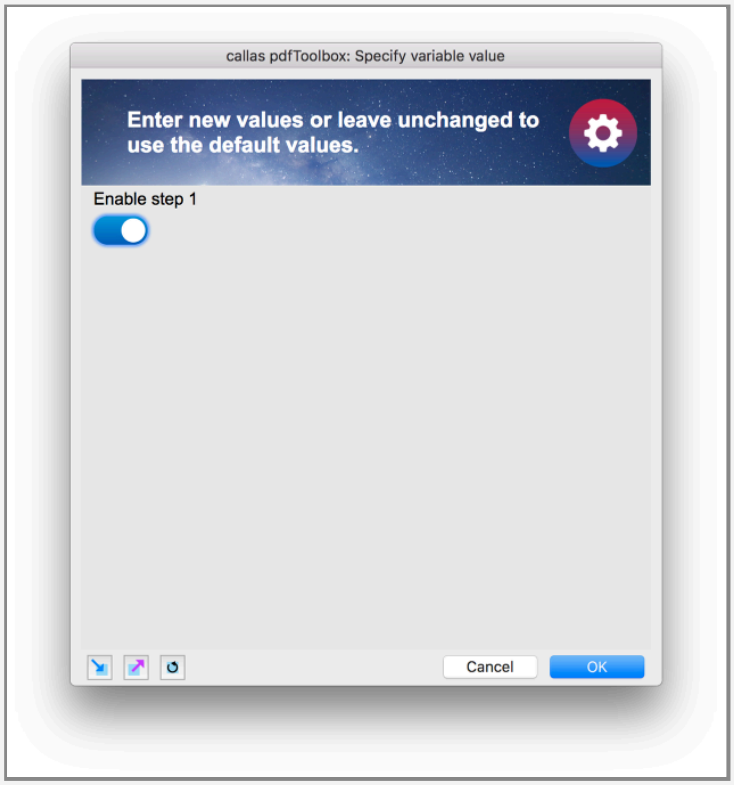
In the screenshot above, the Profile 'Sheetfed offset (CMYK and RGB) (GWG 2015)' is the first step of our example Process Plan. pdfToolbox 11 offers a checkbox/Variable option to switch on or off this Process Plan step (marked with a red rectangle). In this example with a checked checkbox, the Process Plan step (Profile) is switched ON (Obviously, an unchecked checkbox will lead to this Process Plan step being switched off).

Assigning a variable to the Process Plan step

You can assign a variable to the Process Plan step (using the inverted orange triangle next to the checkbox). When a variable is selected, the checkbox cannot be used (a dash [-] appears in the checkbox) as shown in the screenshot below. The screenshot also shows the menu with available Variables, which is shown when you click on the orange Variables symbol between the checkbox and "Enable step".



If you run this Process Plan, it will prompt you to switch on or off the first step of the Process plan as shown in the screenshot below.



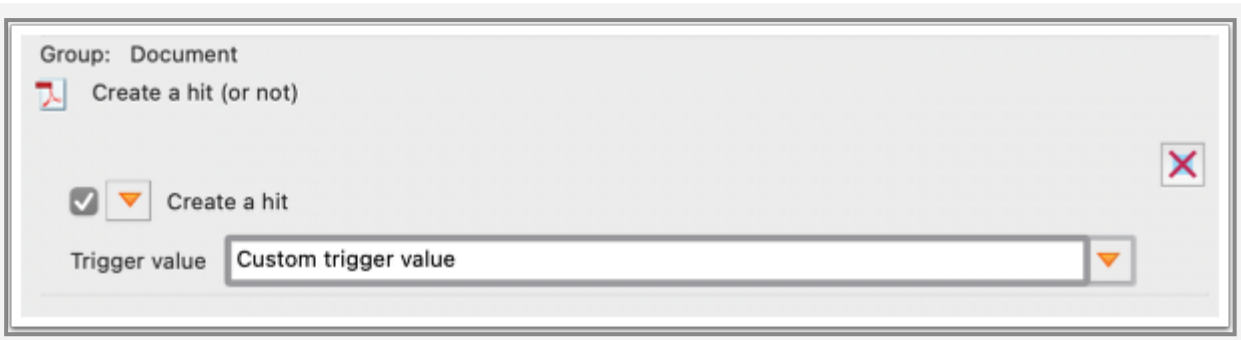
Watch an extended explanation of 'Switch on/off a Process Plan step via a Variable' here:

3.11 Modifying the result (return code) of a Process Plan with the Check property "Create a hit (or not)"

Beginning with pdfToolbox 14 a special Check property "Create a hit (or not)" is available that can be set to always or never create a hit. This can be used in a Process Plan with switches to explicitly specify the result (a hit or no hit). On command line via the return code or in the SDK it can be used to control workflows.

It is in addition possible to specify the trigger value.

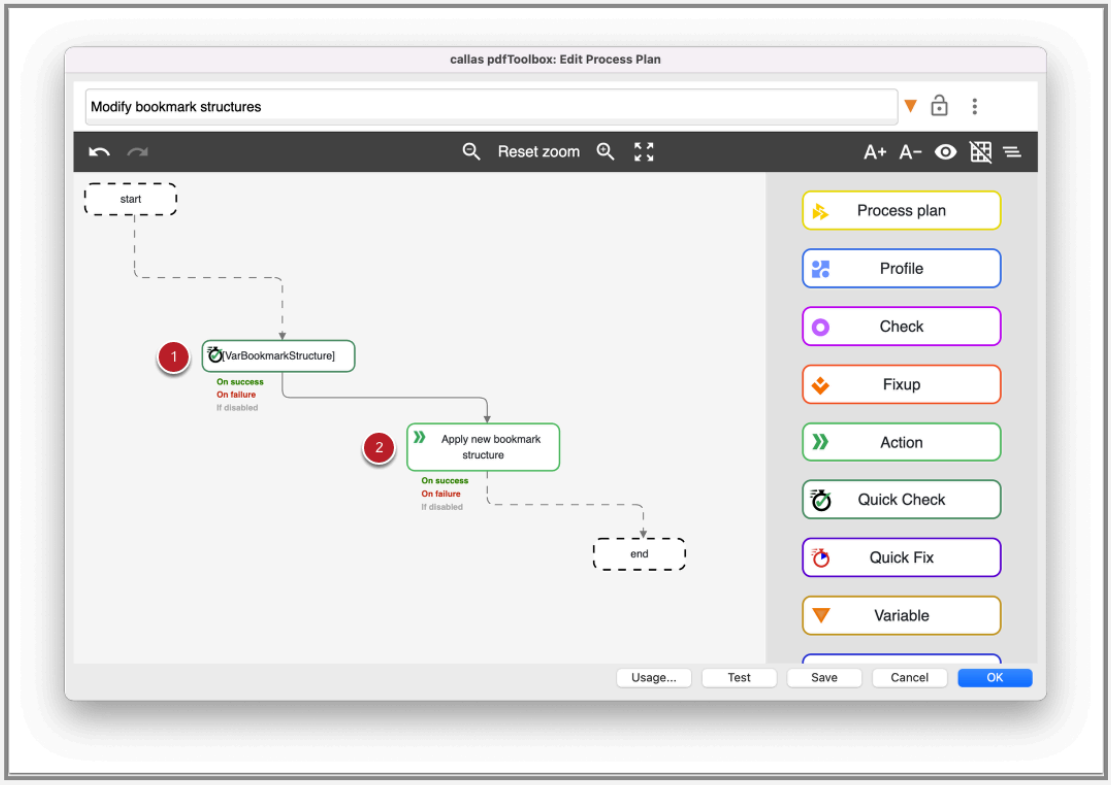
Since all parameters can be controlled via variables, it is easily possible to make the result of a Process Plan depending on variables in the pdfToolbox JavaScript object.



3.12 Modifying structures in a Process Plan

pdfToolbox can extract existing DPart Metadata or bookmark structures in a JSON format so that you can modify them and inject them back into the PDF document.

This functionality is mainly used on the CLI with the --modifystructures parameter, but it is also possible to modify the structures within pdfToolbox Desktop. Here you need to create a Process Plan where you extract the DPart or Bookmark structures using Quick Check and import them using the "Apply Structures" Action.



1. Quick Check to extract the existing structure to JSON. Relevant Quick Check filter expression:
   - DPart: "$.aggregated.doc.dpartroot: true"
   - Bookmarks: "$.aggregated.bookmarks: true"
2. "Apply structures" Action: JavaScript can be used to modify the existing structure from the Quick Check result and inject it into the PDF again.

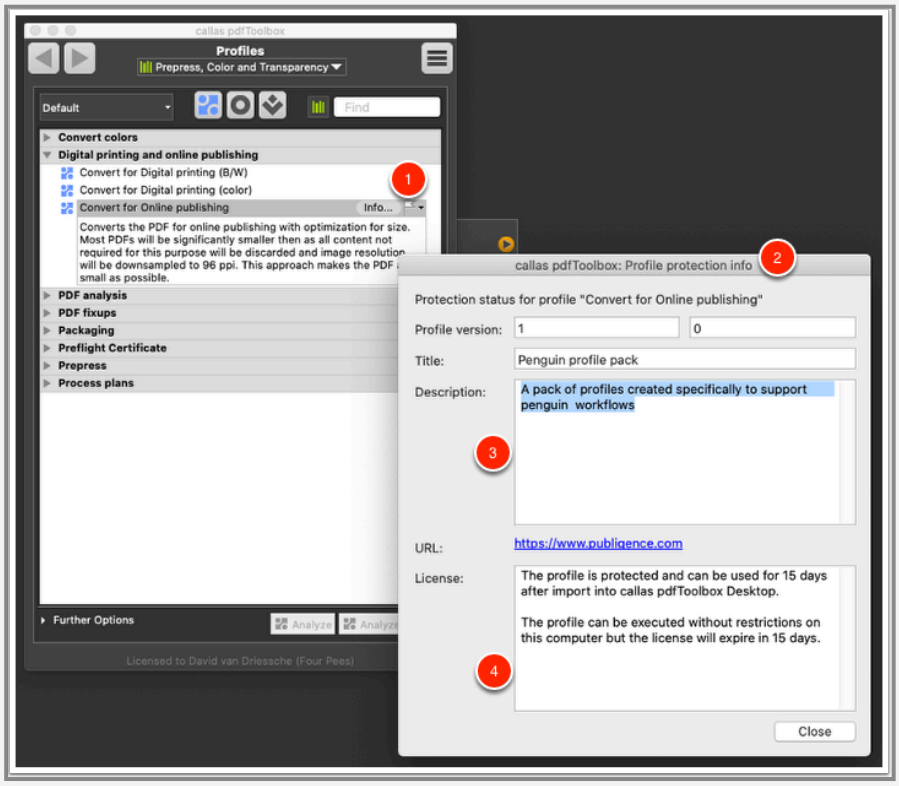
4. Protected Profiles and Process Plans

4.1 What are protected Profiles and Process Plans

Profiles and Process Plans can contain a lot of complexity and intellectual property, in the included script variables, which Fixups and Checks are used, how they are configured etc... In some situations, you want to share such Profiles or Process Plans with others, but do not want them to be able to see or modify your work. You might even want to sell such Profiles or Process Plans and have them only work with certain pdfToolbox Desktop or CLI installations.

Starting with pdfToolbox 14, the ability to protect Profiles and Process Plans was included in the product, especially to handle this situation.

Recognising protected Profiles



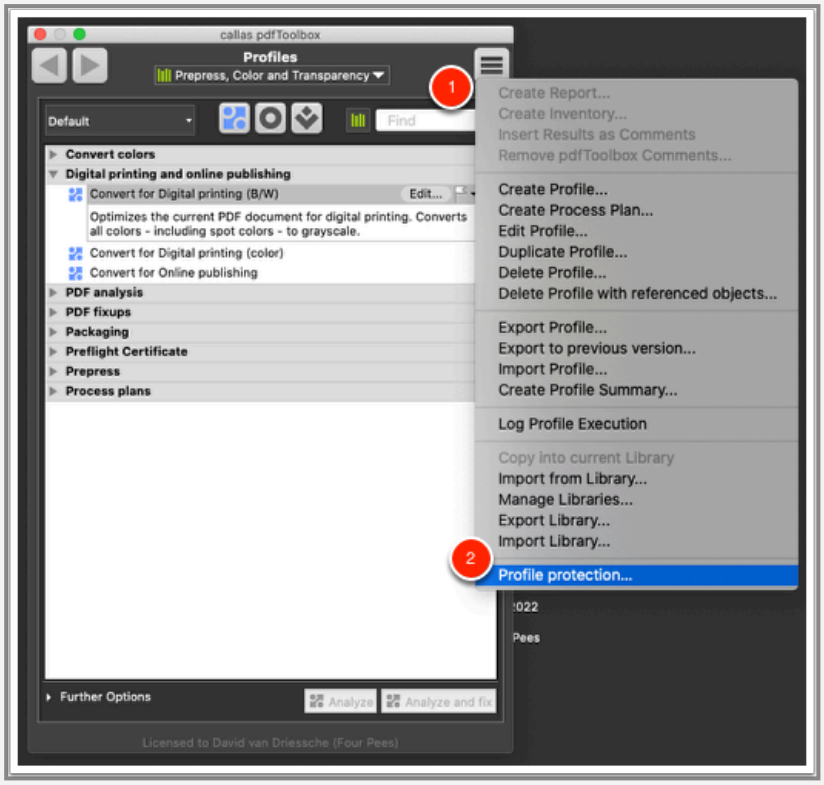
Regular Profiles appear in the profile list with an "Edit..." button. Clicking this opens up the Profile or Process Plan editor and allows seeing and modifying the Profile or Process Plan.

A protected Profile has an "Info..." button instead. Clicking that opens up an information dialog window:

1. The "Info..." button for this protected Profile.
2. The protected Profile information dialog window.
3. Information provided by the person who protected the Profile or Process Plan. A URL may be present and can be clicked when present to get more information about this Profile or Process Plan.
4. Information on whether this Profile or Process Plan can be used on this computer. In the example shown, there is a trial license available that allows using the Profile for another 15 days.

4.2 Protecting one or more Profiles or Process Plans

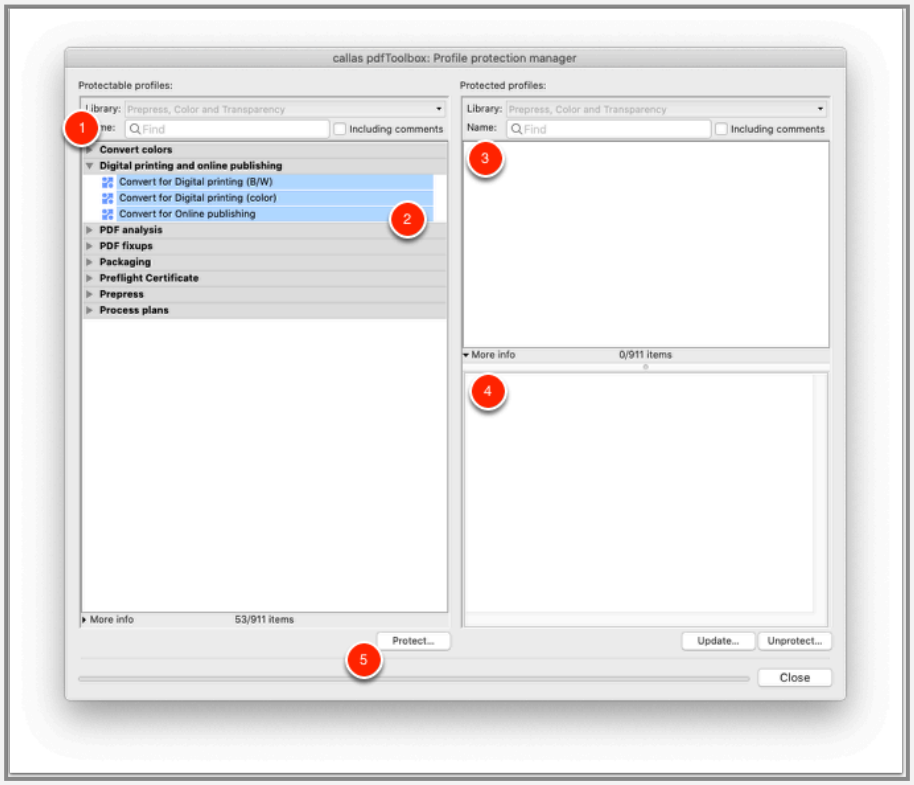
Protecting Profiles or Process Plans starts from the Profiles window in pdfToolbox Desktop (accessible through the Tools > Profiles menu item).



Click on the action button in the right hand top corner (lovingly called the "Hamburger" button), and in the menu that appears, click on "Profile protection...". You do not have to select the Profiles or Process Plans you want to protect yet; you'll do that in the dialog window that appears next.

Selecting the Profiles you want to protect

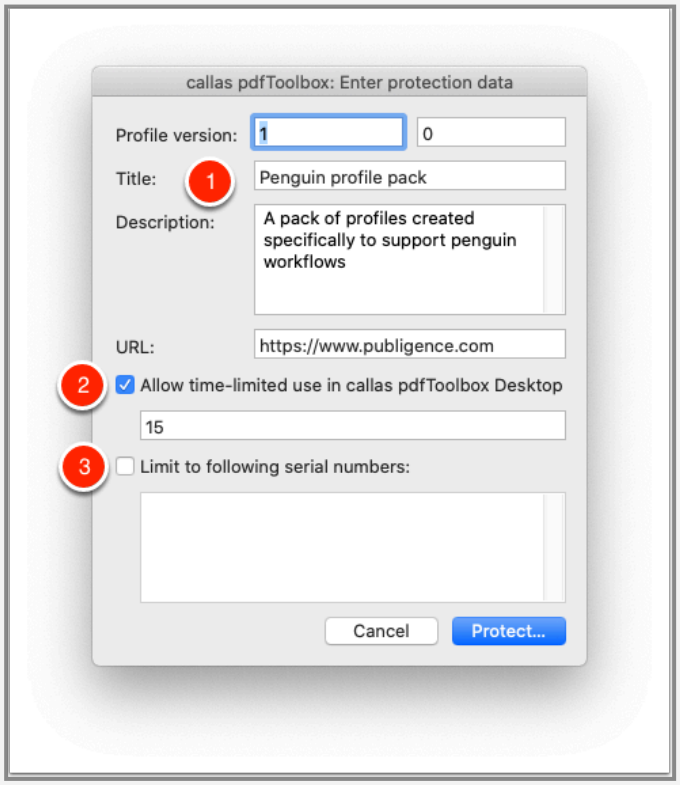
In the Profile protection manager window that appears, you can start the process:



1. The list on the left-hand side shows all Profiles and Process Plans in the current library.
2. You can select any number of Profiles and Process Plans to protect them.
3. If your library already contains protected Profiles, they will be shown in the list of protected Profiles on the right-hand side.
4. Selecting a protected Profile shows that profile's information in the "More info" area.
5. Once you selected the Profiles you want to protect, click the "Protect..." button.

Protecting profiles

After clicking "Protect..." in the previous dialog window, the "Enter protection data" dialog appears. The information you enter here, is used to protect all of the Profiles you selected in the previous step.



1. The top of the dialog contains information about the protected Profiles. This information is shown when a user clicks the "Info..." button on a protected Profile.
   a. A major and minor version number
   b. A title for the protected Profiles (for example identifying the collection of Profiles you are sharing)
   c. A longer description for the protected Profiles
   d. A URL the user can click to learn more about the Profiles
2. If you want to allow users to import your Profiles and use them without limitations for a number of days, make sure this checkbox is enabled and fill in the number of days the trial should last. The trial starts when a user imports the Profiles into pdfToolbox Desktop
3. If you want to specify a list of serial numbers that allow full use of the protected Profiles, enter them here. Any user that doesn't have serial number in this list will be able to use the trial period (if you specify one), but will not be able to use the Profiles after the trial period.

When all information is filled in, click "Protect...".

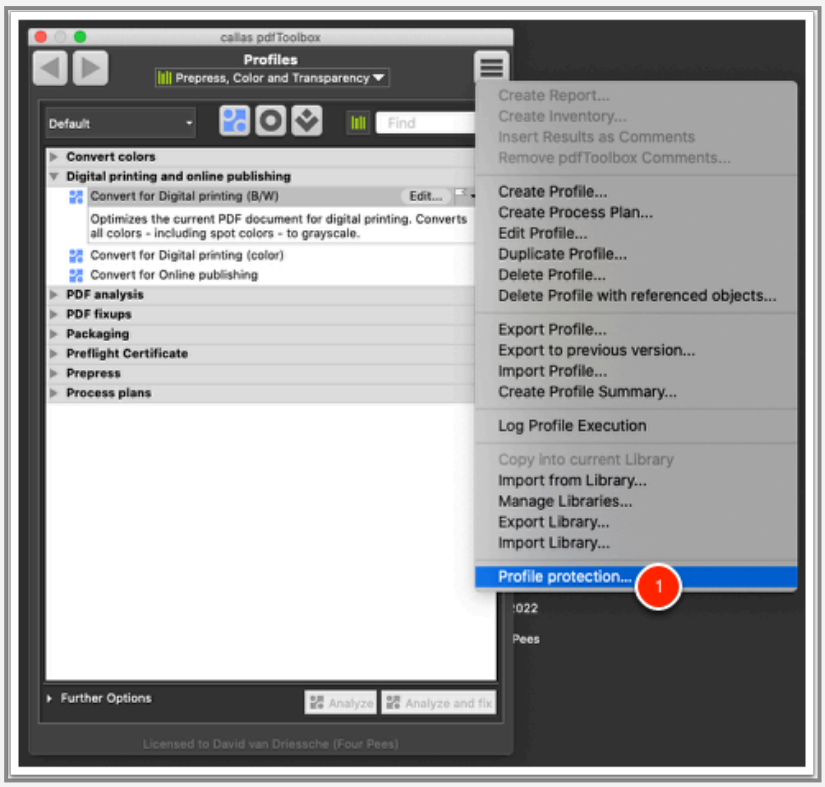


In the password dialog window, enter your password and confirm it. Then click OK.

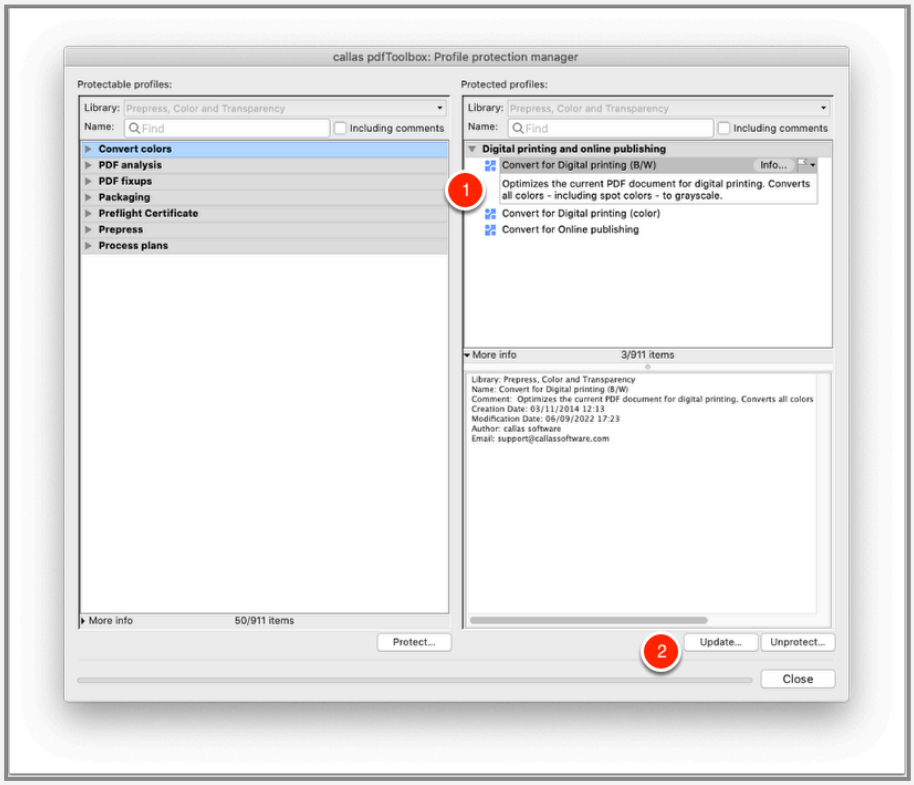
Do not forget your password. callas is not able to retrieve a password you no longer remember.

4.3 Modifying one or more protected Profile or Process Plan

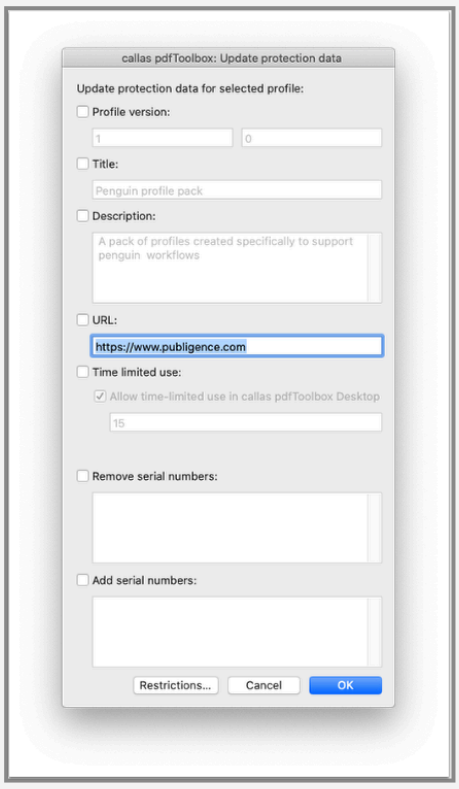
Modifying one or more protected Profiles, follows much of the same steps as protecting them in the first place. Start by clicking on the "Profile protection..." menu item once more.



In the list of protected Profile, select the Profile or Profiles you wish to update. Then click the "Update..." button



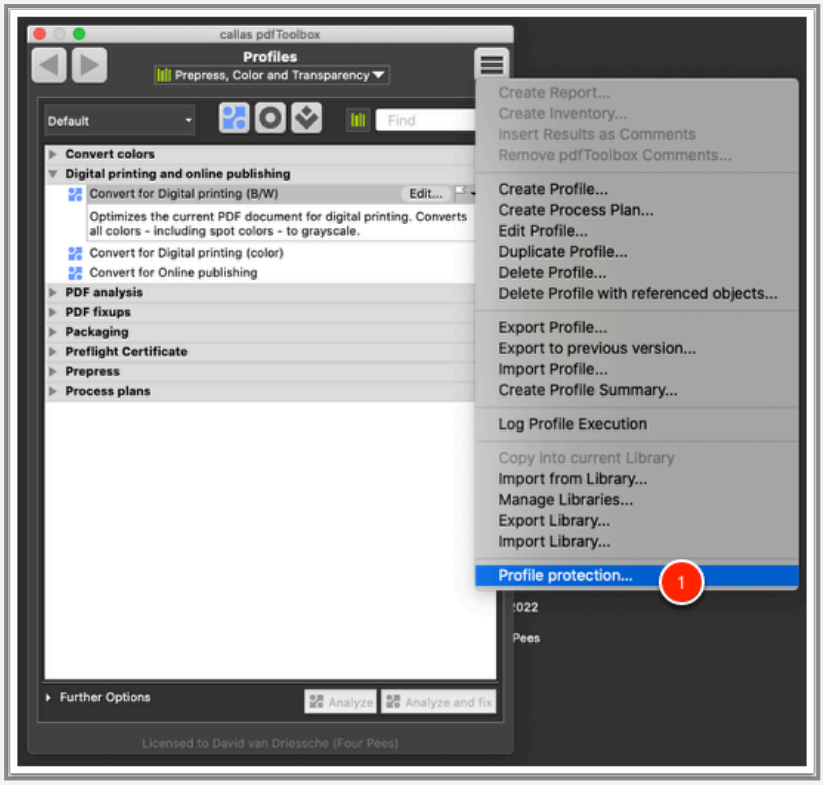
This brings up the "Update protection data" dialog window.



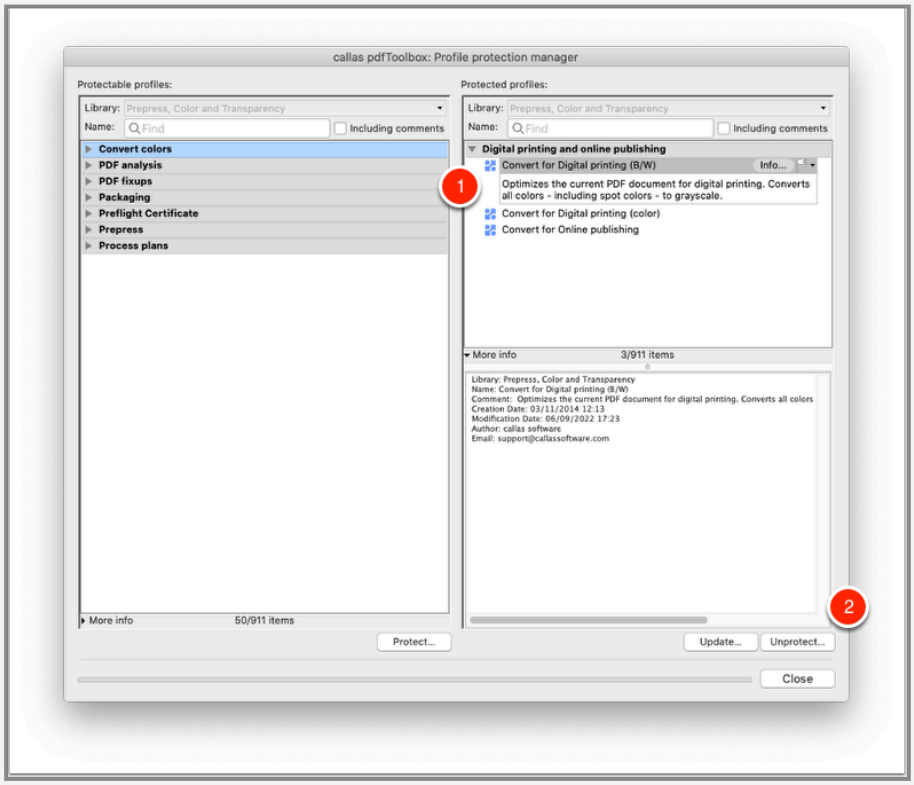
Now enable the checkboxes in front of any information you wish to change, and then modify the information. Note that you can remove or add serial numbers from the list of serial numbers the protected Profiles are enabled for. When you filled in everything you want to change, click "OK" and your Profiles are updated.

4.4 Unprotecting one or more Profiles or Process Plans

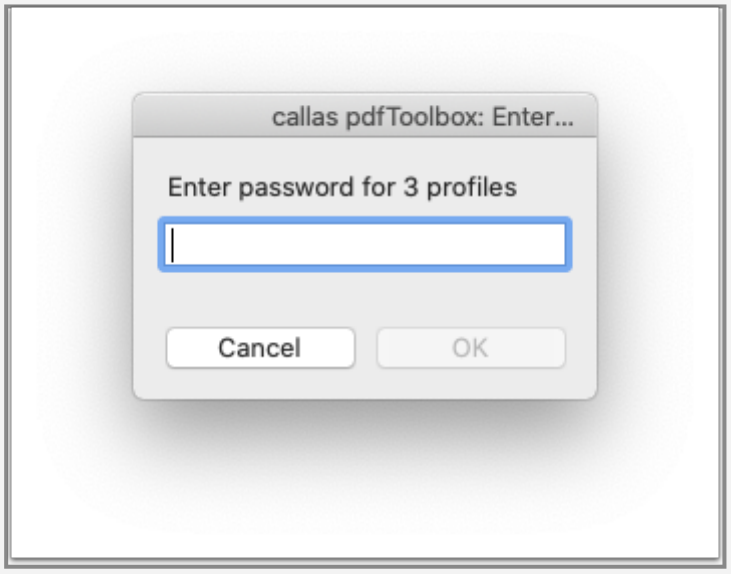
Unprotecting one or more protected Profiles, follows much of the same steps as protecting them in the first place. Start by clicking on the "Profile protection..." menu item once more.



In the list of protected Profiles, select the Profile or Profiles you wish to unprotect. Then click the "Unprotect..." button.



This brings up the password dialog window for those Profiles. Enter the correct password and click "OK". If the password checks out, your Profiles are now unprotected.



5. Actions and their use in callas' products

5.1 Introduction

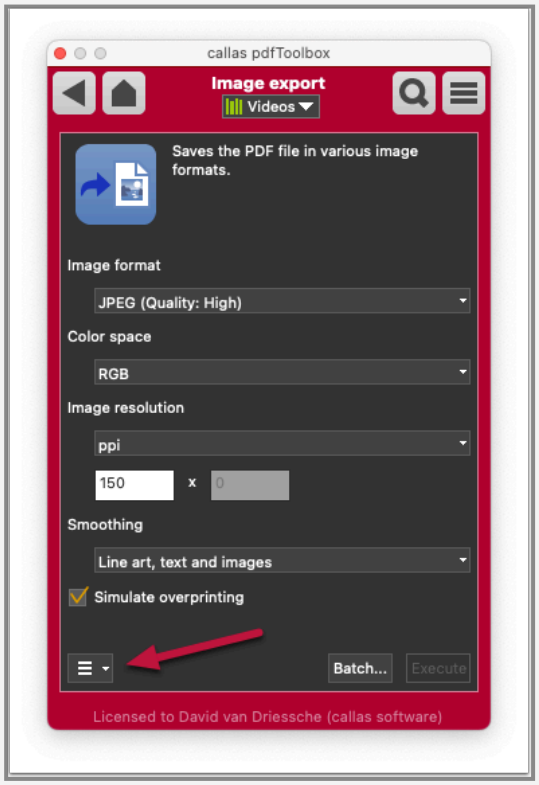
Actions in pdfToolbox are different entities than fixups. Generally they provide access to functionality that wouldn't be easy to integrate into a preflight profile; often because they work with non-PDF files, or work with multiple files on the input or output side. Examples are:

- Save as image
- Convert to PDF
- Split PDF
- Merge PDF
- ...

Actions started out as the items on the Switchboard, but over the years have migrated to different parts of pdfToolbox. This is where they can be found today.

Switchboard

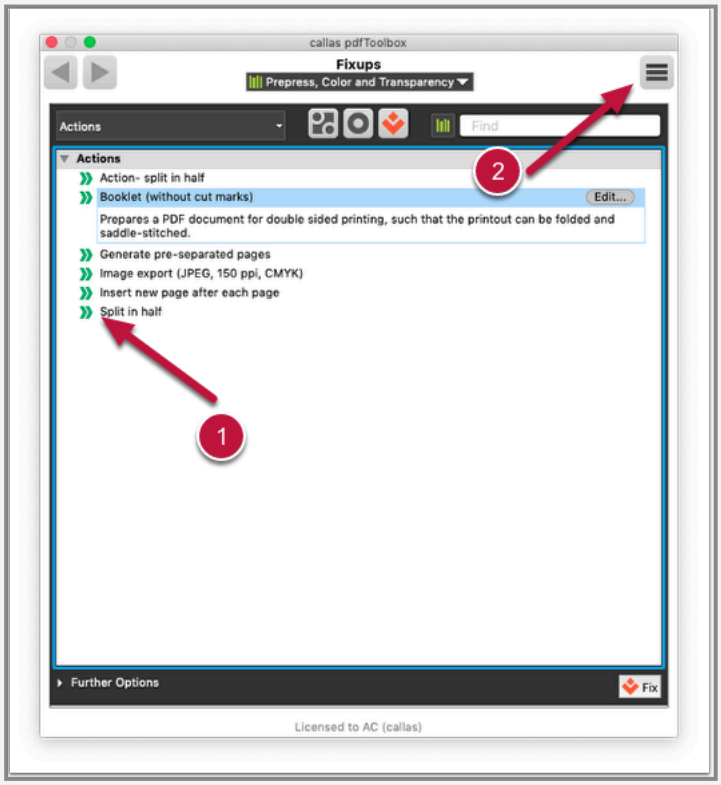
Many actions can be created on the Switchboard. In the example below, the "Image export" pane of the Switchboard is used. Clicking on the hamburger menu at the bottom left, opens a menu. The menu item "Create a new Switchboard action..." allows saving the current Switchboard settings into an action. If you have already created actions of this type, they show up in the menu as well and you can rename or delete them.



While the Switchboard can still be used for this, it is no longer the best way to work with actions.

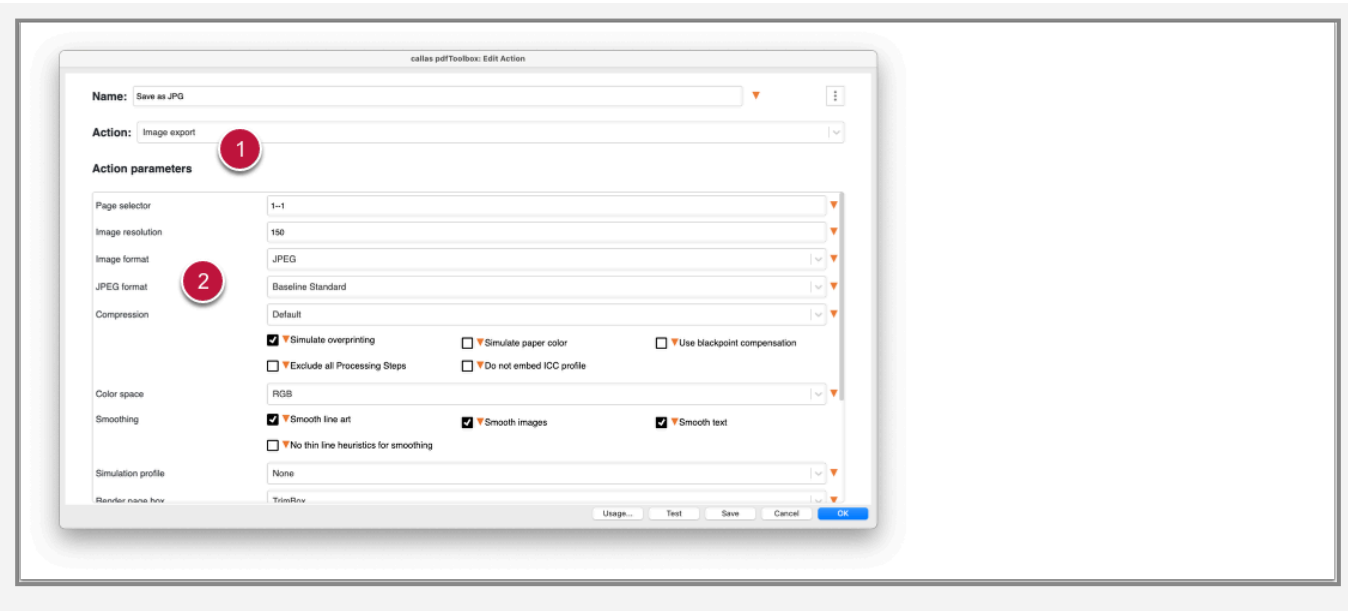
Fixups window

In the Fixups window, all actions in the current library show up with a green icon in front of the name of the action.



You can use the hamburger menu in the top right corner of the fixups window to:

- Create a new action
- Duplicate an existing action
- Delete an action
- Delete an action together with all resources it may be using



When creating a new action, or editing an existing one, the action editor dialog gives access to all action types (1). Each action type then comes with its own set of properties.

Command line or SDK

Most actions in pdfToolbox are also available in pdfToolbox CLI and pdfToolbox SDK. The following article gives an overview of all action types and details on their availability and usage in both the CLI and the SDK.

5.2 Actions for automation

As mentioned in the previous article, actions were originally introduced as items on the Switchboard. However, Switchboard actions are limited to manual PDF modifications within pdfToolbox Desktop and cannot be used in automated workflows.

This article provides a simple overview of all action types, and indicates whether (and how) they are available in pdfToolbox CLI and pdfToolbox SDK for the use in automated workflows.

Action types

Name	Description	Switchboard	CLI	SDK
Add bleed	Adds bleed around the trimbox of the PDF.	Large Format Printing > Add bleed	Fixup: ‘Generate bleed at page edges’	
Add borders	Adds white space around the PDF with an optional fold line.	Large Format Printing > Add borders	Fixup: ’Set page geometry boxes’ Fixup: ‘Create and apply shapes'	
Add ink layer	Adds an additional ink layer under or over a selected portion of the PDF.	Large Format Printing > Add ink layer	Fixup: ‘Create and apply shapes'	
Add marks	Adds marks for the TrimBox and/or BleedBox to the PDF.	Prepress > Add marks	Fixup: ‘Add marks'	
Add missing layer	Put all objects that currently are not on a	Layers > Add missing layer	Fixup: ‘Put objects on layer'	

Name	Description	Switchboard	CLI	SDK
	layer yet, on a new or existing layer.			
Booklet	Prepares the PDF for double-sided printing, such that the printout can be folded and saddle-stitched.	Arrange > Booklet	--booklet	PTB_Booklet
Brightness	Adjusts the brightness of all objects on the page, and is most suitable for B/W pages or where color PDFs have to be printed in black and white.	Colors > Brightness	Fixup: 'Adjust dot gain'	
Browse layers	Allows you to see and modify layers and OCCDs.	Layers > Browse layers	-	
Browse metadata	Allows you to review document and object-level metadata in the PDF.	Reports > Browse metadata	--extractxmpmetadata	PTB_ExtractXMPMetada
Colors	Creates a detailed report about the colors of the PDF.	Reports > Colors	--report=INVENTORY,COLORS...	PTB_PreflightReport (wi PTB_eInventory as PTB_EReportType)
Comments	Puts com-	Layers >	-	

Name	Description	Switchboard	CLI	SDK
by author	ments from all authors on separate layers	Comments by author		
Compare documents	Allows visual comparison between two PDFs.	Reports > Compare documents	--compare	PTB_Compare
Content export	Extracts the content of the PDF as plain text or an XML with individual words or characters.	Text > Content export	--extractcontent Action: ‘Contect export'	Action: ‘Contect export'
Convert to RGB	Converts PDF files to RGB (sRGB)	Colors > Convert to RGB	Profile: ‘Convert to RGB'	
Convert to B/W	Converts PDF files to black and white	Colors > Convert to B/W	Profile: ‘Convert to grayscale'	
Correct 4c Black	Restores accidentally separated black.	Colors > Correct 4c black	Fixup: ‘Correct 4c black'	
Create file package	Creates a new PDF file containing all files from the selected folder. The file package has a summary and bookmarks for direct access to the embedded files.	Document > Create file package	--collection	-
Create Trim-	Inserts a Trim-	Prepress >	Fixup: ‘Derive page geometry boxes from crop marks'	

Name	Description	Switchboard	CLI	SDK
Box	Box and/or BleedBox based on the printer marks in the PDF.	Create Trim-box		
Crop to visible	Crops the page to the visible area.	Pages > Crop to visible	Fixup: ‘Crop all pages to bounding box'	
DeviceLink conversion	Converts colors in the PDF based on DeviceLink profiles.	Colors > DeviceLink conversion	Fixup: ‘Convert colors using DeviceLink profiles'	
Downsample	Reduces the resolution of images.	Images > Downsample	Fixup: 'Downsample/compress XXX images'	
Duplicate page	Duplicates pages in a PDF.	Arrange > Duplicate page	--duplicate page	Fixup ‘Insert page'
Embed CxF data	Embeds CxF data in the PDF.	Prepress > Embed Cxf data	--cxf --embed	PTB_EmbedCxF
Embed files	Embeds the selected files into the current PDF. All file types can be embedded.	Document > Embed files	--collection	-
Embed font	Embeds missing fonts in the PDF if they are installed on the system.	Text > Embed font	Fixup: ‘Embed missing fonts'	
Enlarge	Enlarges the page area without modifying the page	Pages > Enlarge	Fixup: ’Set page geometry boxes'	

Name	Description	Switchboard	CLI	SDK
	content.			
Enumerate layer	Creates separate layers for the selected object types.	Layers > Enumerate layer	--enumeratelayers	PTB_EnumerateLayers
EPS export	Saves the PDF file as one or more single page EPS files. Transparency is flattened using the highest setting.	Document > EPS export	--createeps	PTB_SaveAsEPS
Extract CxF data	Extracts CxF data from the PDF.	Prepress > Extract CxF data	--cxf --extract	PTB_ExtractCxF
Extract dieline	Extracts a dueling from objects identified by a single check	Document > Extract dieline	--extractdieline	-
Extract files	Extracts files that are embedded in the PDF to the selected folder. Existing folder structures are reconstructed.	Document > Extract files	--extractembeddedfiles	PTB_ExtractEmbeddedFi
Extract ICC profiles	Extracts ICC profiles from the PDF file (either the output intent profile or source profiles).	Prepress > Extract ICC profiles Colors > Extract ICC profiles	--extracticcprofiles	PTB_ExtractICCProfile

Name	Description	Switchboard	CLI	SDK
Fill page	Puts several pages on a larger imposed sheet. Distributes pages across / down as space permits.	Arrange > Fill page	--fillpage	PTB_FillPage
Flatten layers	Flattens all layers. Objects on invisible layers are removed.	Layers > Flatten layers	Fixup: 'Discard hidden layer content and flatten visible layers'	
Flatten transparency	Flattens transparency in the PDF.	Prepress > Flatten transparency	Fixup: 'Flatten transparency'	
Flip	Flips the pages horizontally or vertically	Pages > Flip	Fixup: 'Flip pages'	
Font to outlines	Converts text to outlines so their fonts are no longer needed in the PDF.	Text > Font to outline	Fixup: 'Convert font to outlines'	
Fonts	Creates a detailed report about the fonts of the PDF.	Reports > Fonts	--report=INVENTORY,FONTS...	PTB_PreflightReport (with PTB_eInventory as PTB_EReportType)
Form fields by type	Puts form fields based on their type on separate layers.	Layers > Form fields by type	-	
Generate bleed	Generates bleed for	Pages > Generate bleed	Fixup: 'Generate bleed by upscaling' Fixup: 'Generate bleed at page edges'	

Name	Description	Switchboard	CLI	SDK
	pages either by upscaling or by mirroring page content as an image.			
Grommets by Number	Adds a specific number of grommets to the PDF.	Large Format Print > Grommets by number	Fixup: 'Add grommets'	
Grommets by distance	Adds grommets at specific distances around the PDF.	Large Format Print > Grommets by distance	Fixup: 'Add grommets'	
Handout	Creates handouts from a PDF presentation.	Present > Handout	--handout	PTB_Handout
Image export	Saves the PDF in various image formats	Document > Image export	--saveasimg	PTB_SaveAsImage
Images	Creates a detailed report about the images of the PDF.	Reports > Images	--report=INVENTORY,IMAGES...	PTB_PreflightReport (wit PTB_eInventory as PTB_EReportType)
Import as layer	Imports the selected PDF document as a layer in the PDF.	Layers > Import as layer	--importaslayer	PTB_ImportAsLayer
Impose	Offers flexible imposition based on imposition templates.	Arrange > Impose	--impose	PTB_Impose

Name	Description	Switchboard	CLI	SDK
Increase line width	Increases line width of thin lines to ensure printability	Prepress > Increase line width	Fixup: ‘Increase line width'	
Light table	Puts several pages on one new page to give the impression of a light table.	Present > Light table	--lighttable	PTB_LightTable
Make layer	Creates a new layer in the PDF with the objects identified using a preflight check.	Layers > Make layer	Fixup: ‘Put objects on layer'	
Modify images	Extracts all images from a PDF into a folder, together with a descriptive JSON file and replaces the images in the PDF with the images in the folder.	Images > Extract	--modifyimages --extract	PTB_ModifyImagesExtrac
		Images > Replace	--modifyimages --replace	PTB_ModifyImagesRepla
Move content	Moves the content of the page by a given distance.	Pages > Move content	Quickfix: ‘Apply transformation matrix'	
New page	Creates an empty page in the PDF document.	Arrange > New page	Fixup: ‘Insert page'	
N-Up	Puts several pages on a	Arrange > N-Up	--nup	PTB_NUp

Name	Description	Switchboard	CLI	SDK
	larger imposed sheet.			
OCR	Adds invisible text to the page via OCR.	Text > OCR	Fixup: ‘Create invisible text via OCR'	
Office PDF to CMYK	Converts PDF files created by office applications to the selected printing condition.	Colors > Office PDF to CMYK	Profile: ‘Convert to CMYK, Office conversion'	
Office to PDF	Converts office documents directly to PDF.	Document > Office to PDF	Use office documents as input files	
Optimize PDF	Optimizes the internal structure of the PDF and saves it for fast web view.	Document > Optimize PDF	--optimizepdf	-
Organize pages	Allows for reorganizing the pages of one or more PDF documents.	Pages > Organize pages	Quickfix: ‘Reorder pages’ *(partial)*	
Overprint & Knockout	Fixes overprint issues for black and white objects.	Prepress > Overprint & knockout	Fixup: ’Set overprint & knockout'	
Outline page boxes	Adds vector lines on top of the page geometry boxes.	Prepress > Outline page boxes	Fixup: ‘Outline page geometry boxes'	
Output in-	Sets, replaces	Prepress >	Fixup: ‘Embed output intent’, ‘Remove all output intent(s	

Name	Description	Switchboard	CLI	SDK
tent	or removes an output intent from the PDF.	Output intent	and 'Set ICC profile for output intent(s)'	
Overlay	Places the content of a PDF on top of the active PDF.	Document > Overlay	--overlay	PTB_Overlay
Passe partout	Adds a background border around the current page content.	Present > Passe partout	--passepartout	PTB_PassePartout
PDF/X-1a	Converts a PDF to the PDF/X-1a ISO standard.	Standards > PDF/X-1a	Profile 'Convert to PDF/X-1a'	
PDF/X-3	Converts a PDF to the PDF/X-3 ISO standard.	Standards > PDF/X-3	Profile 'Convert to PDF/X-3'	
PDF/X-4	Converts a PDF to the PDF/X-4 ISO standard.	Standards > PDF/X-4	Profile 'Convert to PDF/X-4'	
PDF/X-6	Converts a PDF to the PDF/X-6 ISO standard.	Standards > PDF/X-6	Profile 'Convert to PDF/X-6'	
Pole banner	Prepares the PDF for production of a pole banner.	Large Format Printing > Pole banner	Process Plan: 'Pole banner'	
PostScript export	Saves the PDF file into a PostScript file. Transparency	Document > PostScript export	--createps	PTB_SaveAsPostScript

Name	Description	Switchboard	CLI	SDK
	is flattened using the highest setting.			
PostScript to PDF	Converts PostScript documents into PDF.	Document > PostScript to PDF	--topdf	PTB_SaveAsPDF2
Pre-separated pages	Splits pages in the PDF into pre-separated pages (e.g. 4 pages for CMYK pages).	Prepress > Pre-separated pages	Action: 'Pre-separated pages'	
Presentation	Optimizes the PDF for slide presentations.	Present > Presentation	--presentation	PTB_Presentation
Process conversion	Prepares the PDF for the chosen printing condition and carries out the necessary color conversion.	Colors > Process conversion	Fixup: 'Convert colors'	
Process to spot	Maps process colors into spot colors	Colors > Process to spot	Fixup: 'Map spot colors' Fixup: 'Map spot and process colors'	
Rasterize	Rasterizes the content of large (heavy) pages into images.	Pages > Rasterize	Fixup: 'Convert page content into image'	
Reader spreads	Converts a PDF with single pages into a document with reader	Arrange > Reader spreads	--readerspreads	PTB_ReaderSpreads

Name	Description	Switchboard	CLI	SDK
	spreads.			
Recompress	Changes the compression type of the selected images.	Images > Recompress	Fixup: 'Downsample/compress XXX images'	
Re-Distill	Recreates the PDF via PostScript.	Document > Re-Distill	--redistill	PTB_Redistill
Registration color	Converts registration color (spot color ‘all’) to black only.	Colors > Registration color	Profile: ‘Convert registration color to CMYK black only'	
Remove CxF data	Removes CxF data from the PDF.	Prepress > Remove CxF data	Fixup: ‘Delete Cxf data'	
Remove ICC profiles	Removes source ICC profiles from tagged CMYK color spaces.	Colors > Remove ICC profiles	Fixup: ‘Conver colors’ using ‘Decalibrate’ option.	
Remove layer	Removes a layer together with all objects on that layer.	Layers > Remove layer	Fixup: ‘Remove layer'	
Replace font	Replaces an embedded font in the PDF with another font.	Text > Replace font	-	
Replace text	Replaces text in the document (using regular expressions that may contain	Text > Replace text	Quickfix: ’Search and replace text'	

Name	Description	Switchboard	CLI	SDK
	back references).			
Resample to JPEG2000	Resample images and compress them using JPEG 2000 compression.	Images > Resample to JPEG2000	Fixup: 'Downsample/compress XXX images'	
Resave as PDF 1.x	Saves the document in the defined PDF format.	Document > Resave as PDF 1.x	Fixup: ’Set PDF version'	
Rotate	Rotates pages by a specified angle	Pages > Rotate	Fixup: ‘Rotate pages’ Quickfix: ‘Apply transformation matrix'	
Sandwich	Places the selected PDF on top of the processed PDF.	Arrange > Sandwich	--overlay	PTB_Overlay
Scale by percent	Scales the PDF to a given percentage.	Pages > Scale by percent	Fixup: ’Scale pages'	
Scale page content only	Scales the page content to a given percentage without changing the page dimensions.	Pages > Scale page content only	Quickfix: ‘Apply transformation matrix'	
Scale page format only	Adjust the page dimensions without changing the page content.	Pages > Scale page format only	Fixup: ’Set page geometry boxes'	
Scale to format	Scales the PDF propor-	Pages > Scale to for-	Fixup: ’Scale pages'	

Name	Description	Switchboard	CLI	SDK
	tionally to fit on the specified paper size.	mat		
Sharpen images	Performs unsharp masking for images, which allows to increase contrast and density of images.	Images > Sharpen images	Fixup: 'Unsharp masking of images'	
Slice	Splits a PDF into two documents: one with the selected objects, the second one with the rest.	Arrange > Slice	--slice	PTB_Slice
Split layers	For each layer view (or layer if no layer views are present), a new file is created with just the content of that layer (view).	Layers > Split layers	--splitlayers	PTB_SplitLayers
Split or reorder	Splits the PDF into 'chunks' of pages. Optionally merges them again.	Arrange > Split or reorder	--splitpdf	PTB_Splitpdf
Split PDF at mark	Splits the PDF into parts based on the	Arrange > Split PDF at mark	--splitatmark	-

Name	Description	Switchboard	CLI	SDK
	result of a check.			
Split in half	Takes a PDF that has reader spreads and splits those into single pages.	Arrange > Split in half	--splithalf	PTB_Splithalf
Spot colors	Modifies spot colors as desired.	Colors > Spot color	Fixup: ‘Map spot colors’ Fixup: ‘Map spot and process colors'	
Spotify	Reduces page content to a specific number of spot colors.	Prepress > Spotify	--spotify	Fixup: ’Spotify'
Spot to process	Converts all spot colors in the PDF to process colors.	Colors > Spot to process	Fixup: ‘Convert all spot colors to CMYK'	
Step & Repeat	Puts each page of a PDF multiple times on a larger imposed sheet.	Arrange > Step & Repeat	--steprepeat	PTB_StepRepeat
Subset font	Creates subsets of fully embedded fonts to minimize file size.	Text > Subset font	Fixup: ’Subset fonts'	
Tiling by number	Cuts the PDF into a specific number of tiles horizontally and vertically.	Large Format Printing > Tiling by number	--tiling	-

Name	Description	Switchboard	CLI	SDK
Tiling by distance	Cuts the PDF into tiles of a specific width and height.	Large Format Printing > Tiling by distance	--tiling	-
Tone value adjustment	Adjusts tone values by a given percentage.	Colors > Tone value adjustment	Fixup: 'Adjust dot gain'	
Unembed all fonts	Removes all embedded fonts from the document.	Text > Unembed all fonts	Action: 'Unembed all fonts'	
Visualizer	Allows for visualizing print-relevant characteristics of the PDF.	Reports > Visualizer	--visualizer	PTB_Visualizer

5.3 Modify images

⚠ This action is available starting with pdfToolbox 16

The modify images action is designed to extract images from a PDF, modify them outside of pdfToolbox with a tool of your choice, and then re-insert them into the original PDF with pdfToolbox, replacing the original images. Images are re-inserted exactly where they come from, and retain their cropping, Z-order, and so on.

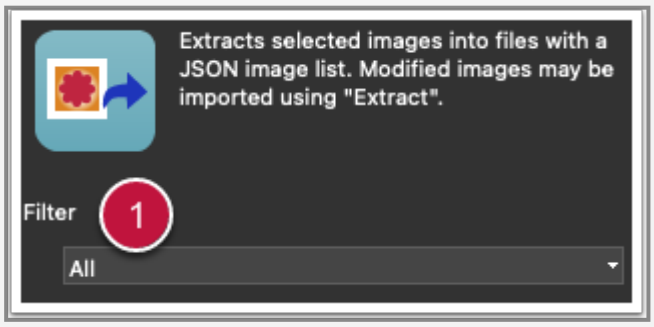
A possible use case would be to extract all images; use an AI-based upsampling or image improvement application, and replace the original images with the improved ones in the PDF. The functionality can be used manually using the Switchboard, or automatically using a pdfToolbox CLI command.

Switchboard



In the Switchboard, click on the "Images" group. To use this action, you have to use "Extract" and "Replace"

Extract



The "Extract" action on the Switchboard extracts all, or the selected images (using a preflight check as a filter) into a folder, together with a JSON file that identifies the images in the PDF.

Replace

You can modify any of the images that were previously extracted, but you should not modify their names. If you do modify their names, you'll need to also modify the exported JSON file so that the names of the images in the JSON and the names on disk are identical.

To replace images, make sure the original PDF is open in pdfToolbox Desktop. Then "Execute" the "Replace" action. pdfToolbox Desktop asks for the location of the JSON file. Select it, and the modified images are inserted into the PDF.

pdfToolbox CLI

In pdfToolbox CLI, the "Modify images" action is implemented as a command-line command called: "--modifyimages". It offers an additional parameter to specify whether you want to extract images or replace images:

```
./pdfToolbox --modifyimages --extract <pdf>
```

```
./pdfToolbox --modifyimages --extract=<kfpx> <pdf>



./pdfToolbox --modifyimages --replace <json> <pdf>
```

Click to copy

The extract command takes a PDF file and exports all images from it, together with the descriptive JSON file. Optionally, the "--extract" parameter can have the path to a pdfToolbox profile (.kfpx file). If that is the case, the profile should contain a single check that filters out those images you want to extract from the PDF.

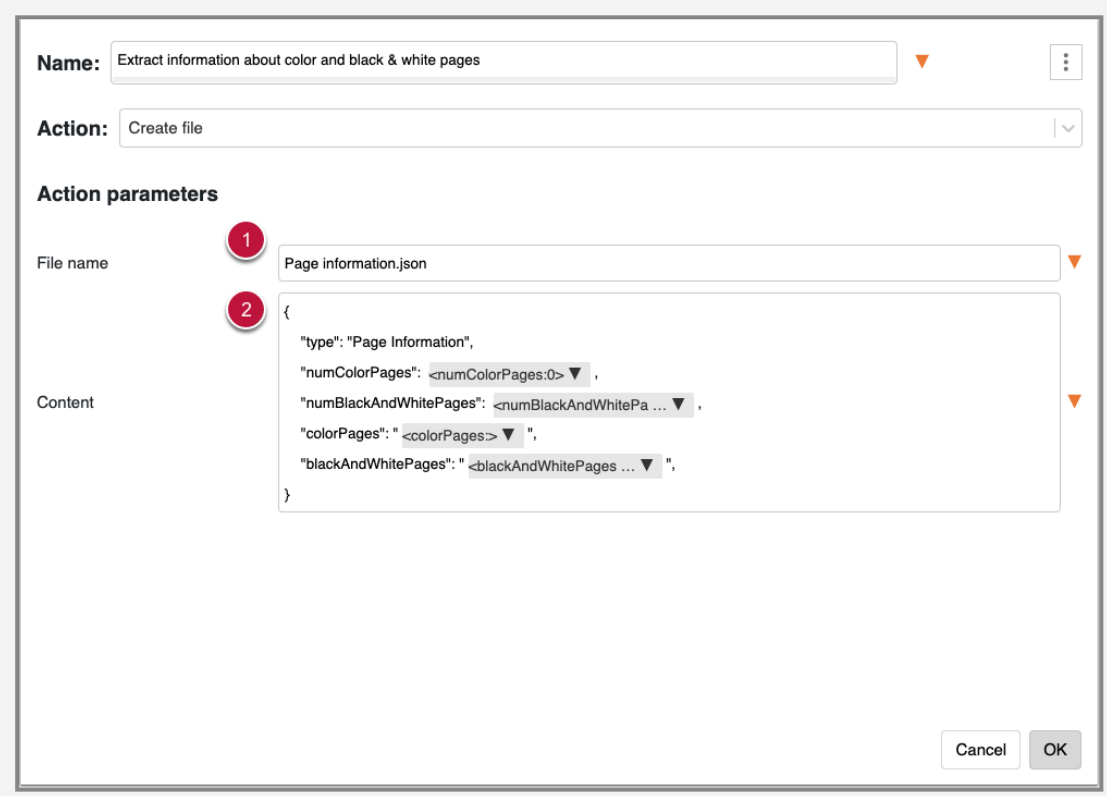
The replace command requires both the path to the JSON file (potentially updated if necessary), and the original PDF file. The output is the modified PDF file with updated images.

5.4 Create file (from process plan)



This action is available starting with pdfToolbox 16

This action allows creating a (text-based) file from a process plan. The action type is "Create file":

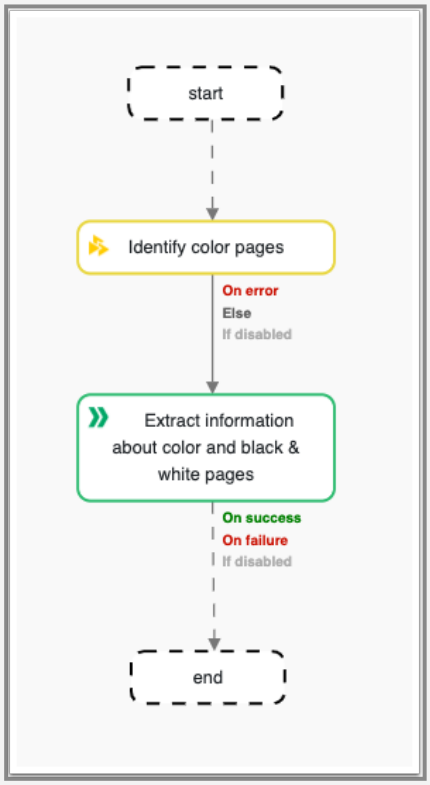


The action has only 2 parameters:

1. The name of the file that is created. By default, the file is created in the same folder as the output of the process plan. As pdfToolbox can't know which type of file you are writing, you have to specify both the name and extension of the output file.
2. The content of the file. The example above uses plain text that includes some simple variables. Of course it's also possible to first create the whole file content in one variable using an earlier JavaScript step of the process plan, and then just to specify that single variable here.

Example

Download the example below to get a better idea of how this works.



The example runs a sub process plan to calculate the number of black-and-white and color pages in a PDF. It then outputs a JSON file with that information using the Create File action and the variables calculated by the previous step.

Output JSON with page color information.kfpx

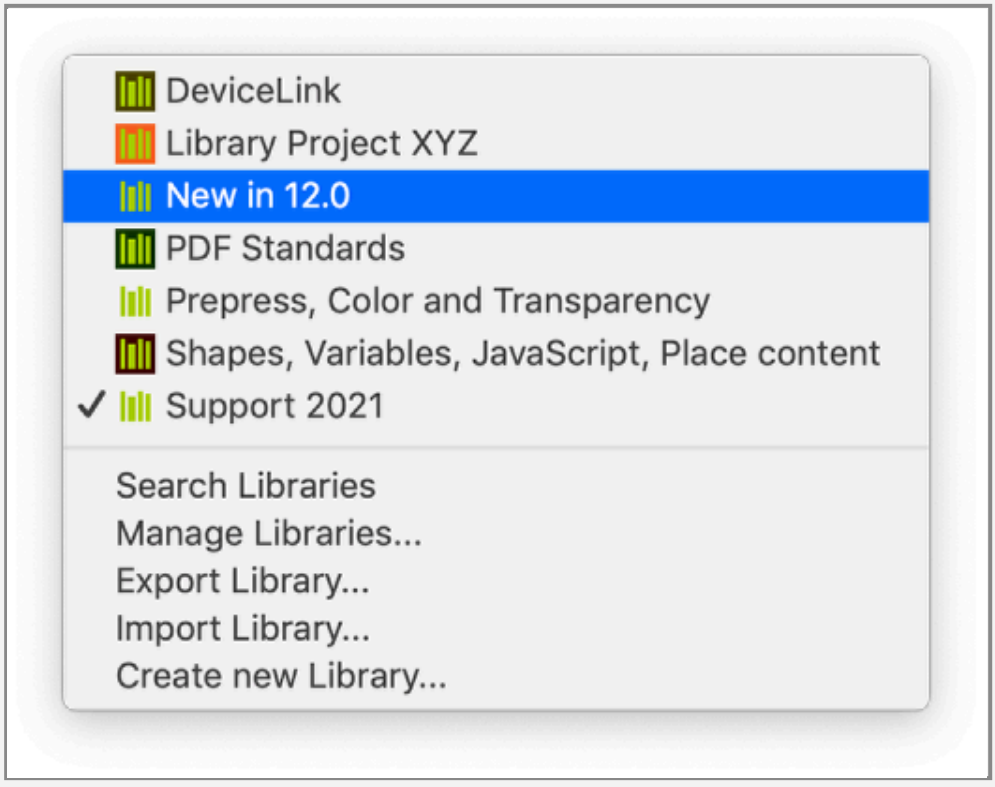
6. Libraries

6.1 Libraries - Overview

What are Libraries?

Libraries are repositories for all the different items you can create in pdfToolbox (such as checks, fixups, profiles, process plans, imposition layouts...).

You can use them to keep yourself organized on different projects, separate development from production profiles and much more.



Watch this short video for an introduction about libraries in pdfToolbox and how they can make you more productive.

Use Libraries for versioning and project management

With pdfToolbox libraries, versions and projects can easily be managed. Supposedly, a printing company has compiled records with profiles, checks and corrections for a particular workflow or customer. Now it is possible to spread them and manage different versions across multiple workstations via libraries.

1. For a customer or a workflow a library was created.
2. If the requirements change or if new pdfToolbox features are integrated, the existing library can simply be duplicated, a new version number (possibly also with the current date) can be given and the additional checks, profiles and corrections can be added.
3. The new, additional library could then be called "customer-v1.1-2016-06-26" or "Printer v1.1-2016-06-26".
4. Further updates will be processed the same.

In this way, it is non-destructive and any changes in the procedure are tracked.

Use Libraries for task oriented workspaces

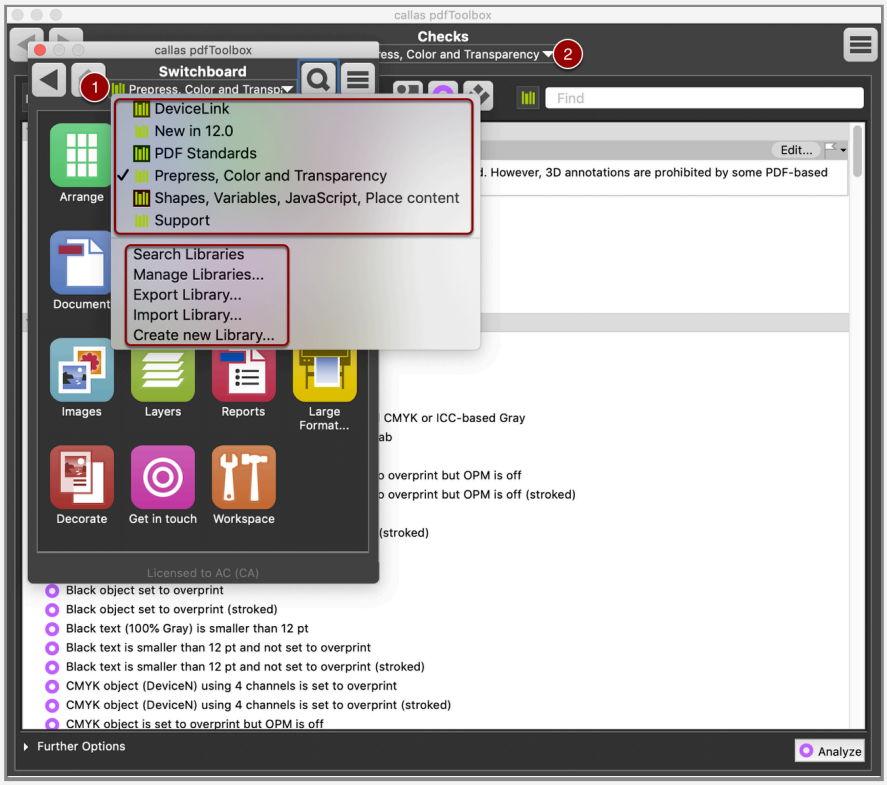
Libraries can be very useful for different employees in a company or for clients with whom you work, to get customized sets of profiles, checks and fixups.

- It may be useful to first create an extensive library, which you can adjust for the required workstations or partners.
- Smaller sets can be created as a library for employees who require only certain functions.
- According to certain workflows libraries can be set up, such as for "Offset", "Digital", "Online PDF", "Large Format", and many more.

Where are the Libraries located?

You can access the Libraries via the dropdown (as shown below) in:

1. The 'Switchboard'
2. The main Profile pane



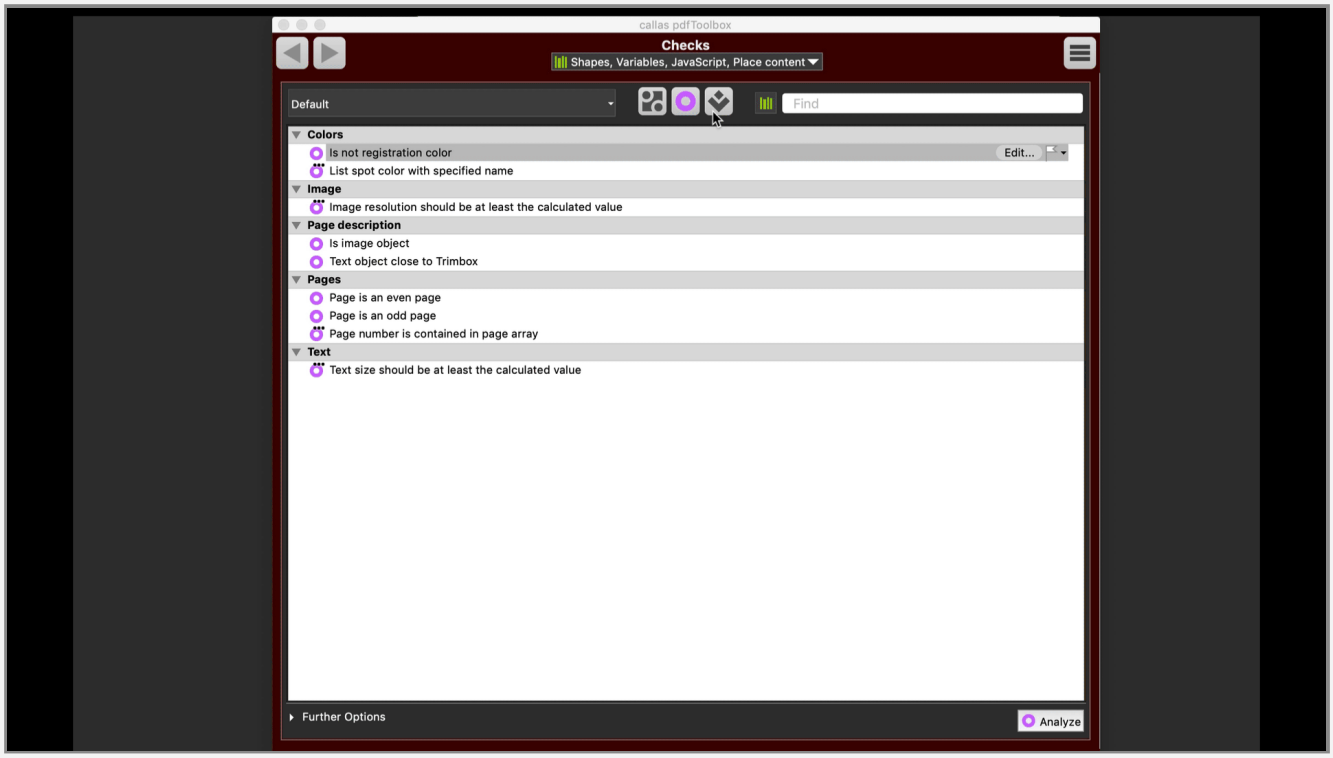
Apart from all the Libraries like 'PDF Standards' etc, you will see other options:

- Search Libraries (explained here)

The rest explained in the next Chapter

- Manage Libraries
- Export Libraries
- Import Libraries
- Create new Library

Switch between different Libraries



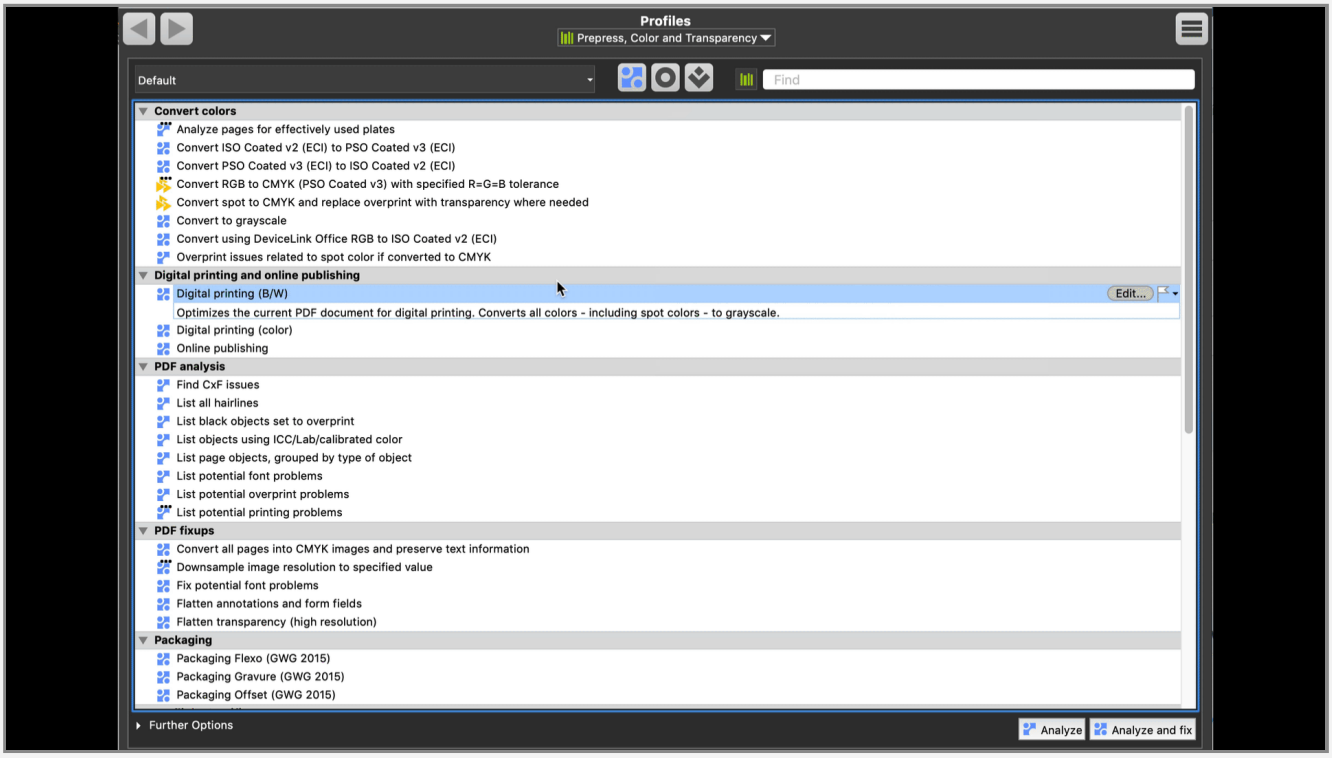
Watch the 10 minute video to know more:

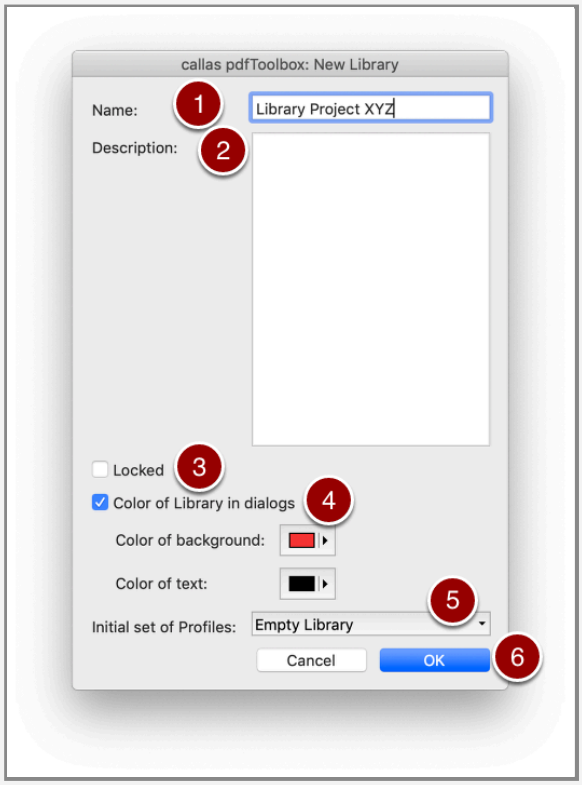
6.2 Create new, Manage, Import or Export Libraries

Create a new Library

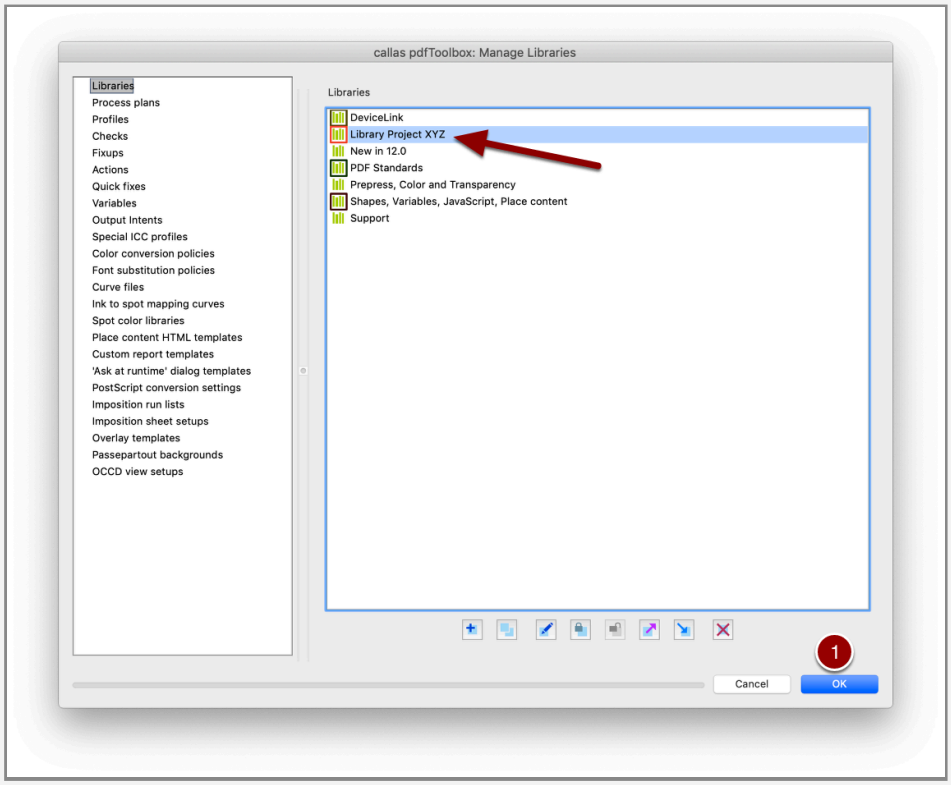
Watch this short video and learn how to create your own libraries in pdfToolbox.

Follow the below steps to create a new Library:





1. In the "**Name**" field save the new Library as "Library Project XYZ".
2. If necessary you can add a **description** for the new Library. No extra information is needed for this Library.
3. The new Library can be **locked** in order to protect them from changes. It is not done in this example.
4. For an improved Library overview, the **color** of the **background** and **text dialogs** can be customised.
5. You can prefer to start with an **empty library, empty library with resources** or using the **entire library with the default settings**. This example starts with a empty Library.
6. Click "**OK**" to save the new library.

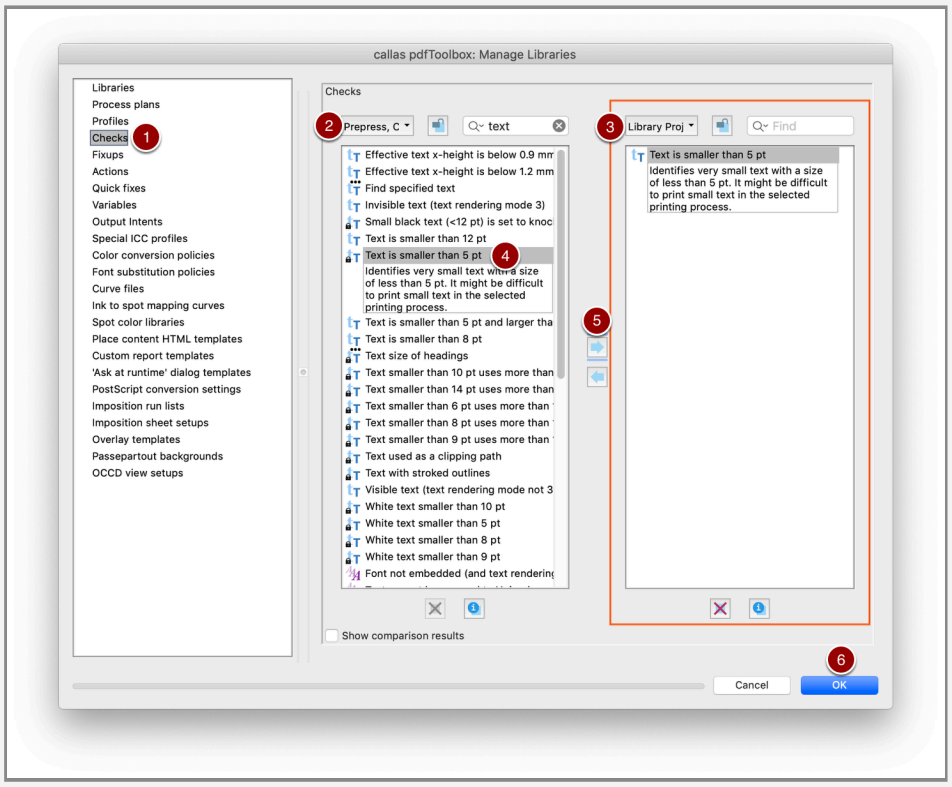


The new library "Library Project XYZ" is shown in the "Manage Libraries" list.

1. Click "OK".

You can now select this new Library in Switchboard and Profile pane.

Add a Check to created library



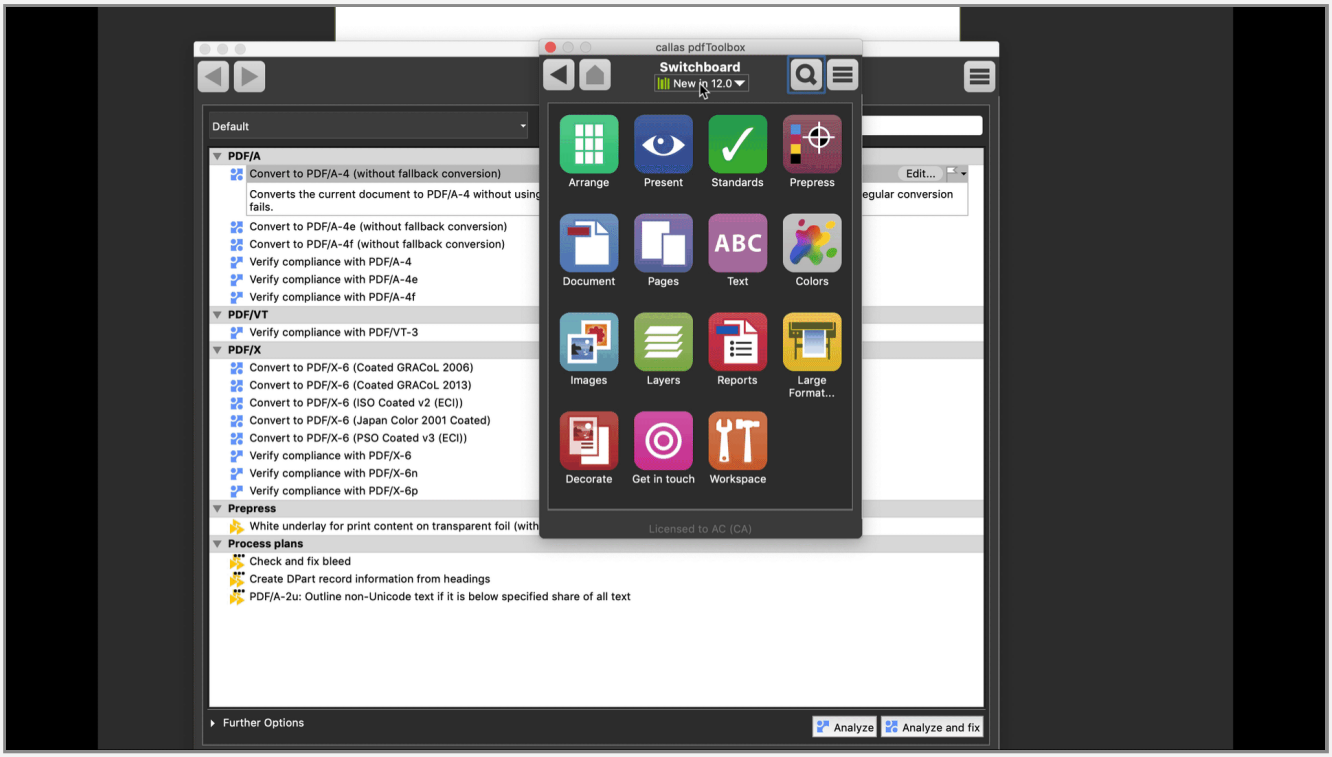
Now we want to investigate the created Library "Library Project XYZ". The new Library was constructed as an empty library. You can derive Process Plans, Profiles, Checks, Fixups and Actions from any other Library to the empty Library.

In this example we add the predefined Check "Text is smaller than 5 pt" from the 'Prepress, color and transparency' Library:

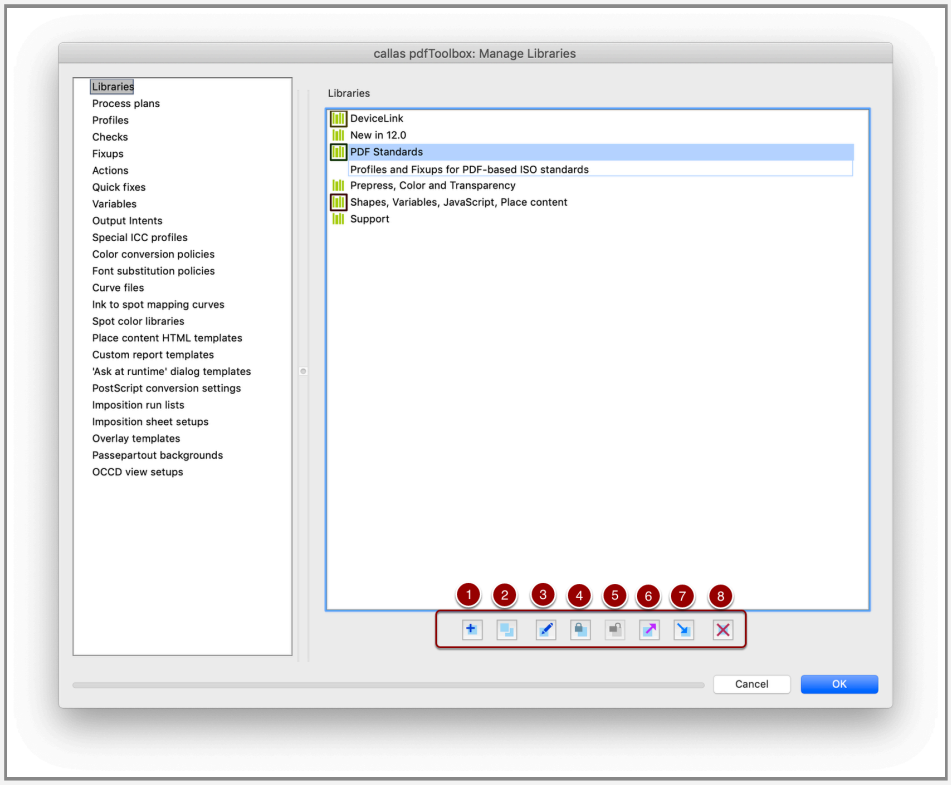
1. Go to 'Manage libraries' in Library dropdown > Select "Checks".
2. In the (left) dropdown list, select "a Library" from where you want to select the Check.
3. In the (right) dropdown list, select "Library Project XYZ" where you want to copy the Check.
4. Select the Check "Text is smaller than 5 pt".
5. Click on the blue arrow button to add the Check in "Library Project XYZ".
6. Click "OK".

Open 'Manage Libraries' dialog

1. Via Switchboard dropdown
2. Via Profile window dropdown
3. 'More options' button (top right)



Manage Libraries dialog

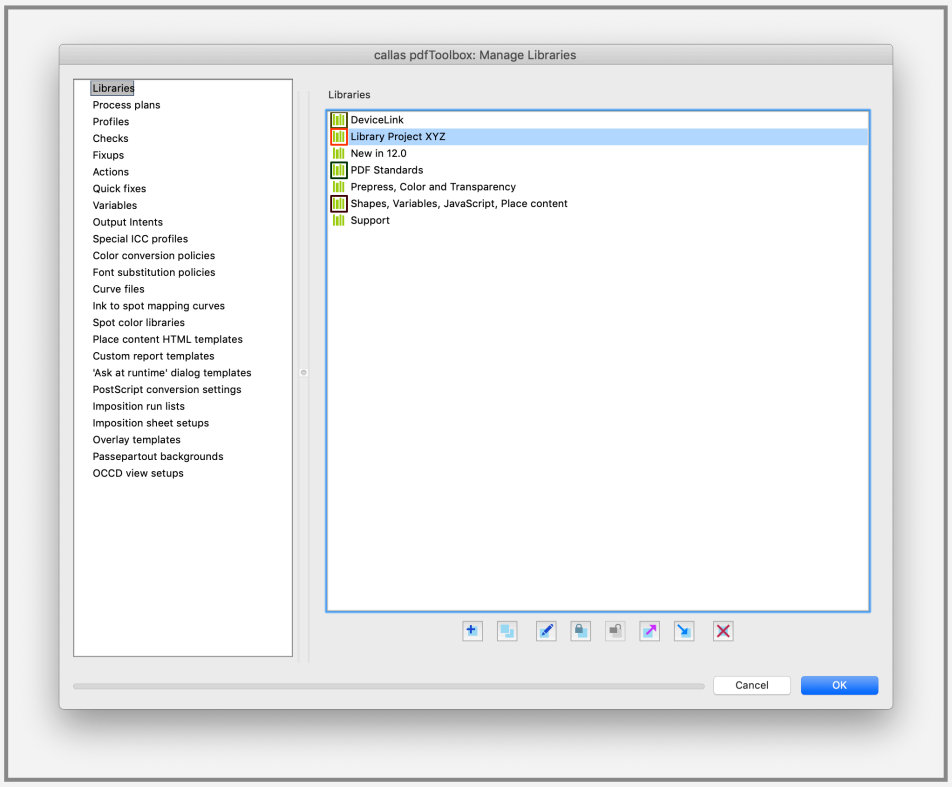


In the Libraries list above, all the default Libraries along with a custom Library 'Support' are shown.

In the lower part of the dialog, buttons for various functions can be found:

1. Adding a new Library.
2. Duplicate an existing Library.
3. Edit a selected Library.
4. Protect a selected Library.
5. Unprotect a selected Library.
6. Export a selected Library.
7. Import a Library.
8. Delete a selected Library.

Analyze the Manage Libraries dialog (left pane)



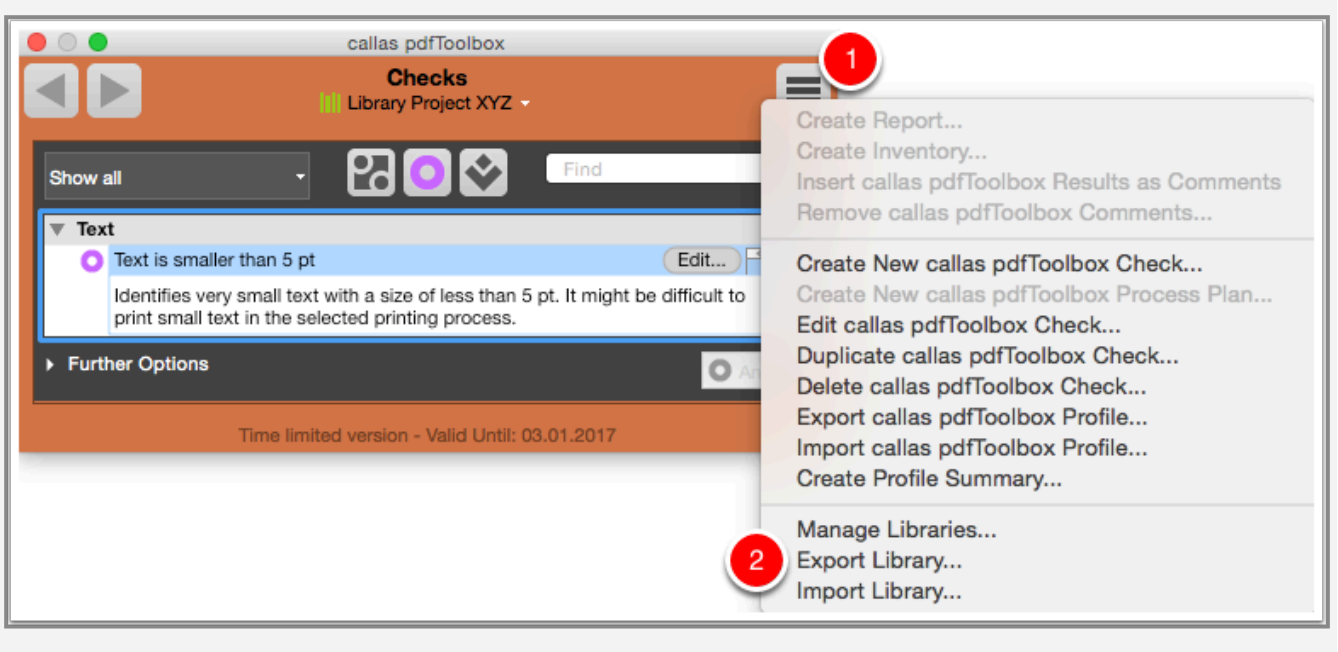
In the left column, under the entry "Libraries" all areas are listed that can be set up in the selected library.

Here are the details:

- **Libraries:** A list of available standard or custom libraries
- **Process plans:** Combined Profiles, Checks, Fixups and Actions to apply on a PDF file. Here the sequence is important
- **Profiles:** pdfToolbox Profiles that combined Checks and Fixups to apply on a PDF file. For example to convert a PDF to the PDF/X-4 standard
- **Checks:** pdfToolbox Checks that investigate a PDF file according to some criteria. For example the thickness of text
- **Fixups:** pdfToolbox Fixups that fixed a PDF file. For example converting all the text into outlines
- **Actions:** pdfToolbox Actions apply an action on a PDF file. For example creating an imposed PDF file.
- **Quick fixes:** Faster PDF manipulation feature

- **Variables:** Values given at run-time instead of predefined values
- **Output Intents:** The output conditions
- **ICC profiles:** ICC color profiles
- **Color conversion policies:** Policies to convert colors in a PDF file. For example convert to "Office RGB"
- **Font substitution policies:** Policies about missing fonts
- **Curve files:** Curve settings for tone value adjustments
- **Ink to spot mapping curves:** Mapping curve that looks at the ink amount values across the page and uses them to look up the tint value to use for the spot color plate
- **Spot color libraries:** A custom spot color library that stores spot colors and your alternate color space definitions
- **Place content HTML templates:** HTML templates to add PDF content. For example adding the file name.
- **Custom report templates:** A custom report templates for report generation.
- **'Ask at runtime' dialog templates:** A custom template to adjust the 'Ask at runtime' dialog to specify variables
- **PostScript conversion settings:** Quality levels.
- **Imposition run lists:** Imposition scheme plans. For example to plan 8 business cards, double sided.
- **Imposition sheet setups:** Sheet configuration plans. For example 8 business cards on a A4 page size.
- **Overlay templates:** Overlay templates such as "Draft" or "Watermark".
- **Passepartout backgrounds:** Passepartout backgrounds such as "Sand paper" or "Stone".
- **OCCD view setups:** Template to view Optional Content Configuration DIctionary

Export library



1. Click on the Action button (of the Library dropdown).
2. Click "Export Library".

And save the Library in your system.

Export multiple libraries

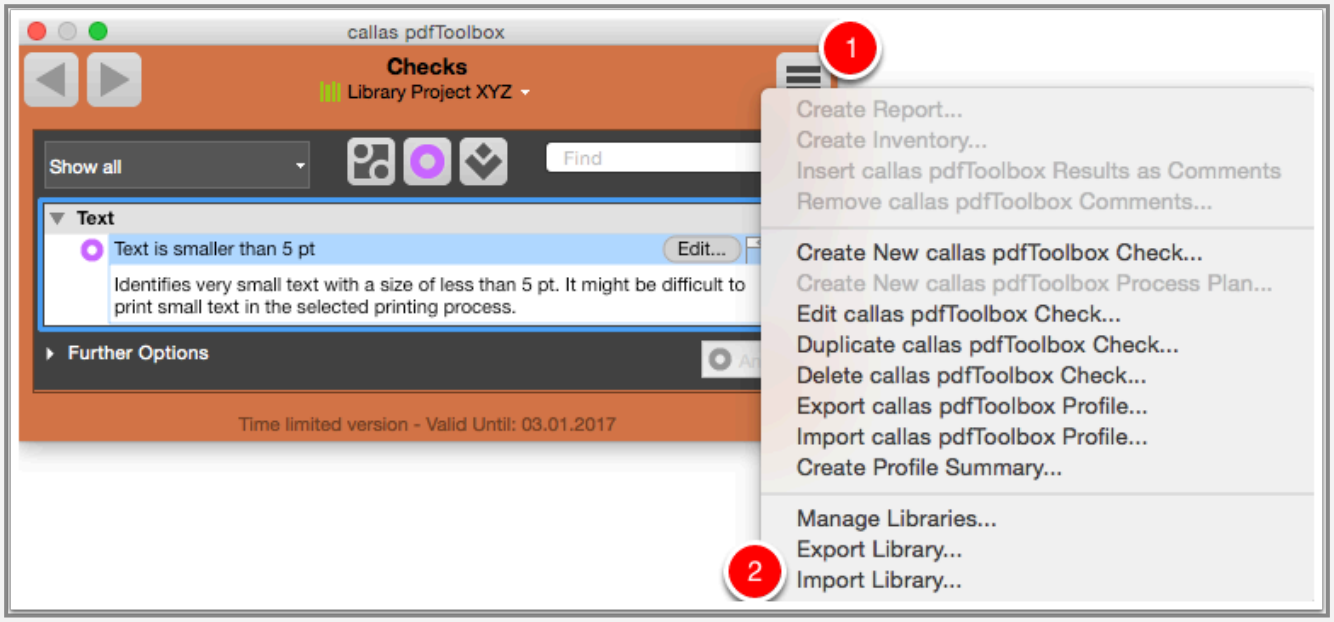
Earliest version with full support for exporting multiple libraries is pdfToolbox 16

In the **"Manage library"** dialog, multiple libraries can be exported at the same time. To do this:

1. Press ⌘ (Cmd) on Mac or Ctrl on Windows to select multiple libraries.
2. Click the "Export Library" button.
3. Choose a folder where the selected libraries should be saved.

The libraries will then be exported to the specified location.

Import library



1. Click on the Action button (or the Library dropdown).
2. Click "Import Library".

Select a Library (.kfpl) from your system and import.

Watch this 10 minute video:

6.3 Search in or across Libraries

Libraries can contain a huge number of Profiles, Fixups, Checks or even Variables and Actions. Various search options help to find certain elements easily.

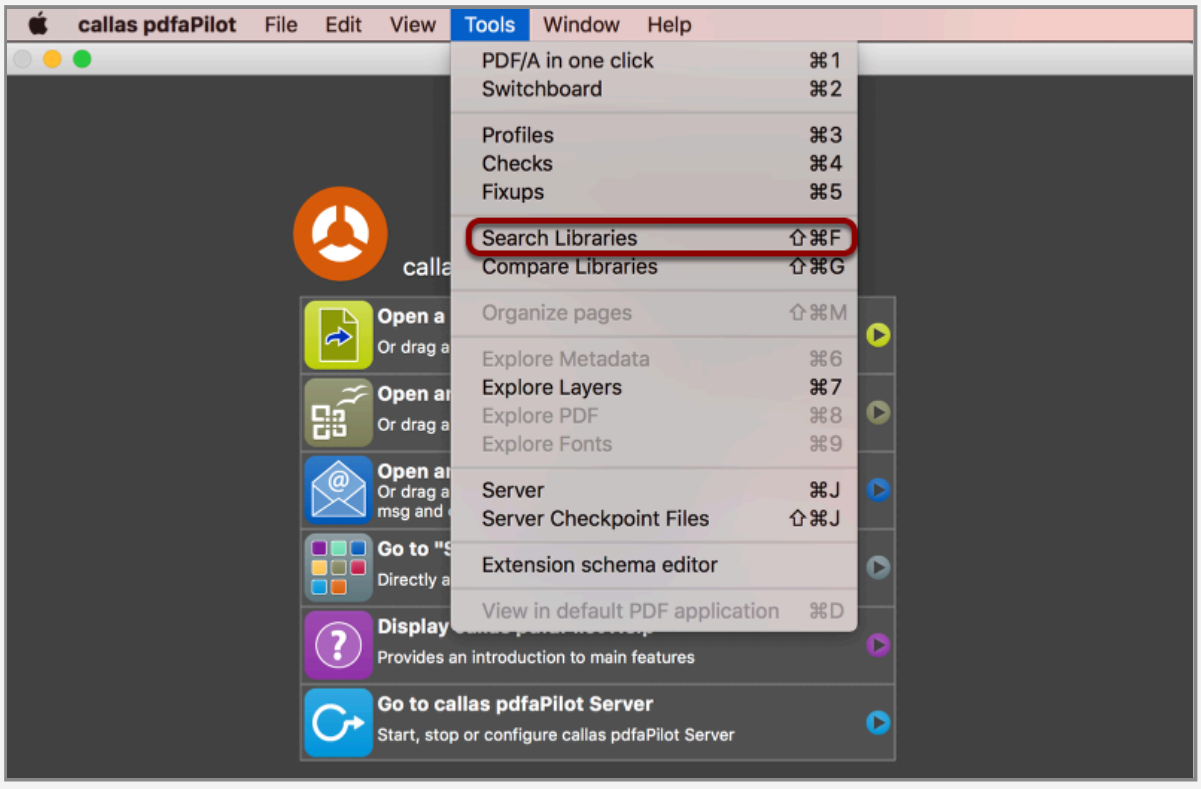
The illustrations show the related product pdfaPilot. The functionality is identical to that of pdfToolbox.

Global Search

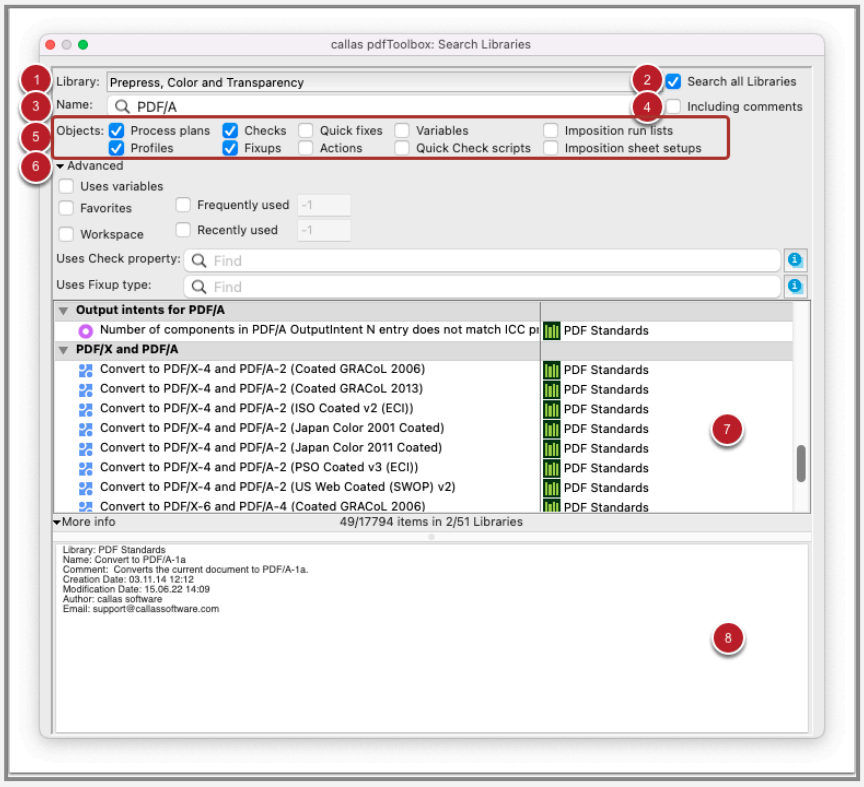
To search for Profiles, Fixups, Checks, Variables or Actions across the entire collection, use the 'Browse Libraries' menu item. You can access the function via Menu: Tools -> Search libraries.

Alternatively, you can use the keyboard shortcut Shift-Command-F.

You can browse libraries using the menu button in the Profiles window.



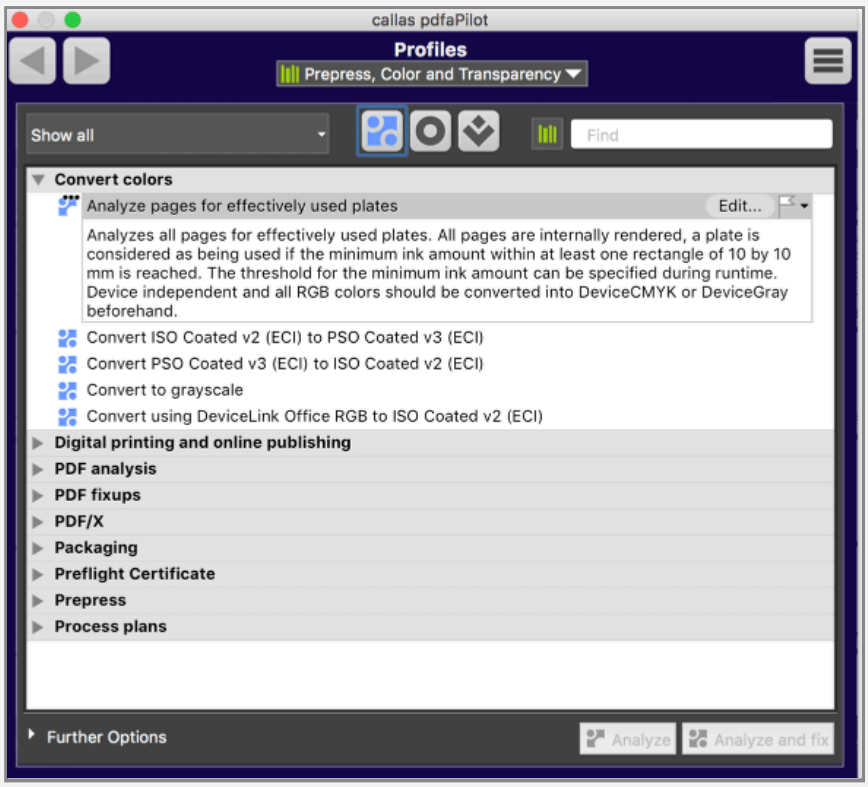
The search window contains various setting and filter options:



1. The Library to be searched can be selected from the pull-down menu.
2. The 'Search All Libraries' checkbox is used to search across Libraries.
3. Enter the search term under Name. As soon as you start typing, the first hits are displayed.
4. You can also include the comments used in the individual profiles etc. for the search.
5. Under 'Objects', you can specify the search individually for Process Plans, Checks, Variables, Profiles, Fixups, Quick fixes, Actions, Quick Check scrips, imposition run lists and imposition sheet setups
6. Under 'Advanced', the search can be set in more detail (Favourites, uses Variables, Workplace as well as frequently or last used). The use of Check properties and Fix-up types can also be taken into account.
7. The hits are displayed in the list display, specifying the library.
8. Under 'More info' you will also find the creation date, author and other info of the selected entry.

Library- Specific search in the Profiles/Fixups/ Checks

You can also search in the windows for Profiles, Fixups or Checks.



Only profiles that are in the currently selected library are displayed.

Cross-library search for Profiles/Fixups/Checks



If you want to search all Libraries, first click on the Library icon next to the search field. The search term in the "Search" field will be searched across Libraries.

7. Color conversion

7.1 Color conversion overview

Color conversion fixups

You will find all fixups for performing color conversions in the group "Color spaces, spot colors, inks". The most powerful one – "**Convert colors**" – is described in the section " Setting up a color conversion ".

The other available fixups are:

Adjust dot gain

Tone values of the specified objects can be adjusted by applying dot gain curves. For more details please see section " Adjusting tone values ".

Convert colors using DeviceLink profiles

DeviceLink profiles are especially intended to convert colors within one color model, e.g. to convert from one CMYK color space to another.

Convert RGB colors using Quick Conversion (Office-RGB)

The Quick Conversion method performs a transformation of RGB colors to CMYK as specified in the PostScript Reference Language Manual. You can define UCR (Under Color Removal) and BG (Black Generation) for both RGB vector/text and images.

Map colors

This fixup searches for a specified color and replaces it with a new color. When working within one color model (like from CMYK to CMYK), single pixels of images can be mapped as well. Moreover, intermediate values can be mapped, e.g. if there are lighter and darker gradations of a specific color.

For use with JavaScript a Fixup "Map colors using script variables" is available that can be configured via a JSON expression. Further information can be found here.

Map spot colors

Spot colors can be modified when applying this fixup. The name, alternate color space and overprint status of spot colors can be modified individually. Furthermore, spot colors can be converted to CMYK.

For use with JavaScript a Fixup "Map spot and process colors using script variables" is available that can be configured via a JSON expression. Further information can be found here.

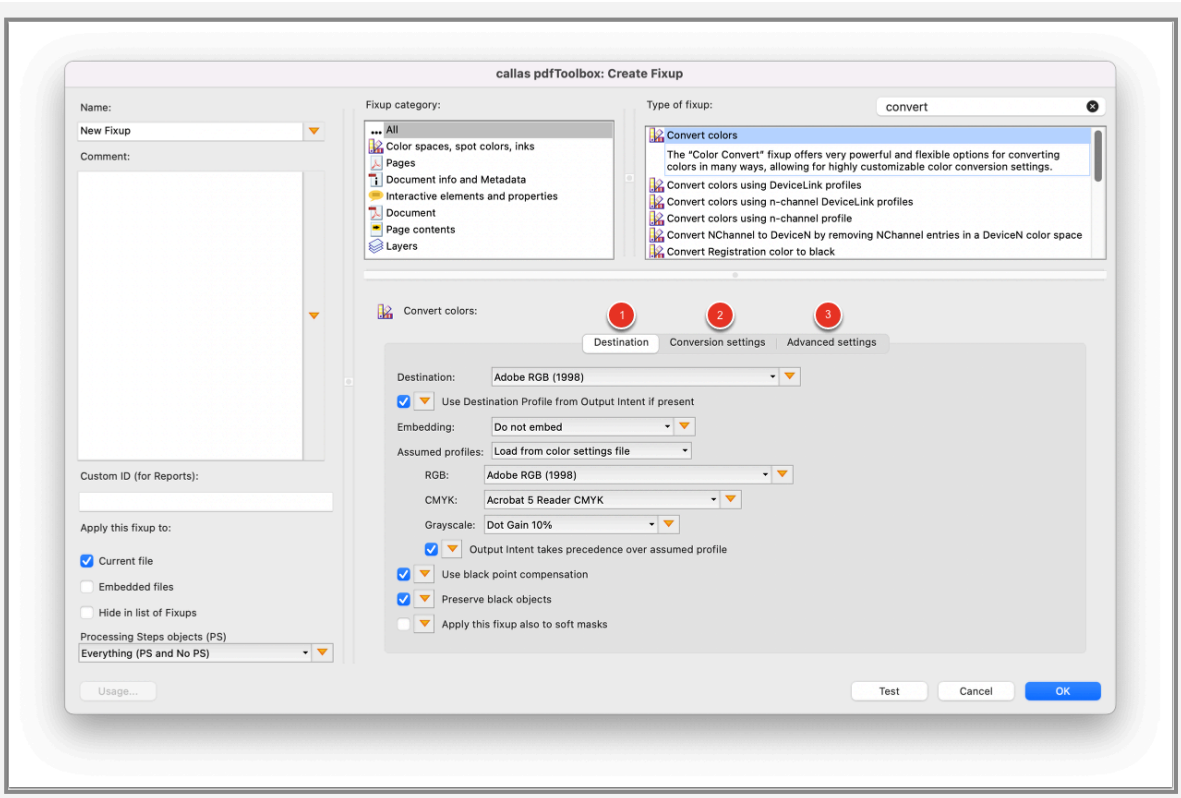
Combining color conversion fixups

If you combine more than one color conversion fixup in a profile, these will be processed in the folllowing order:

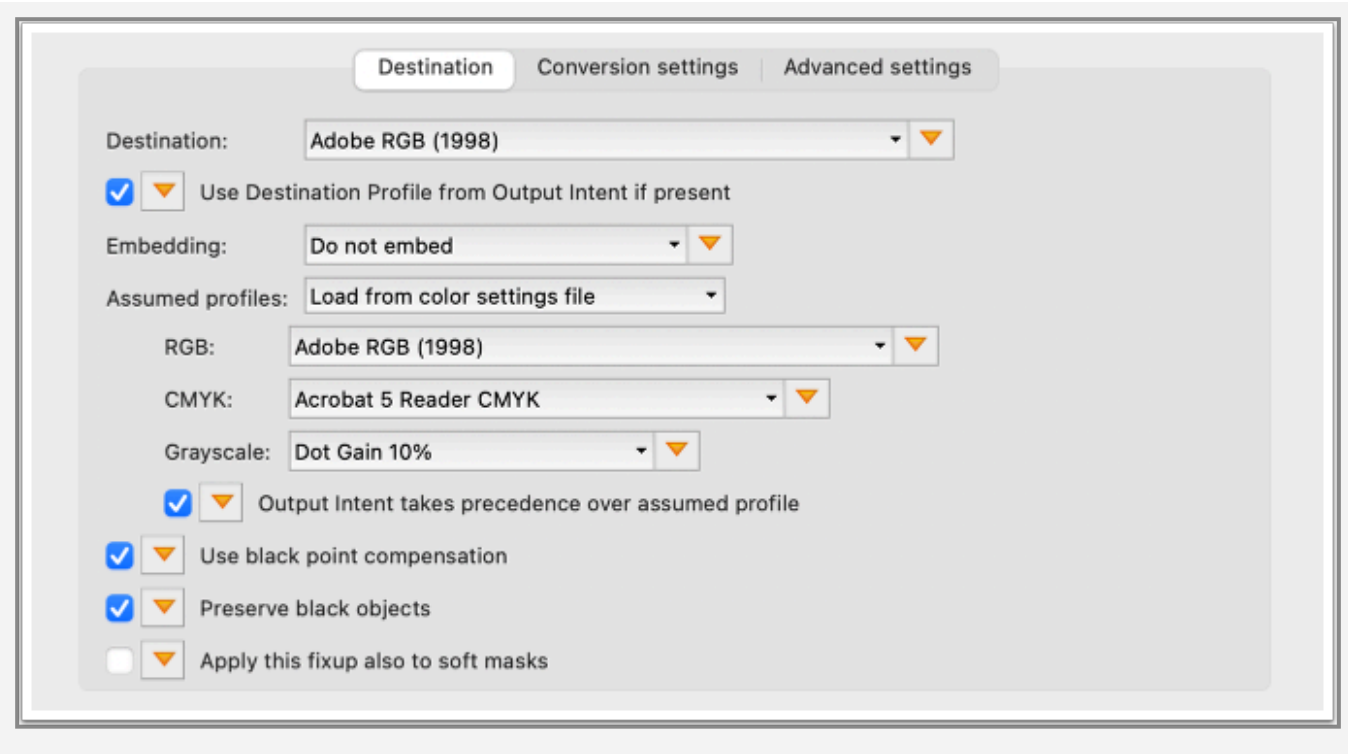
- Adjust Dot Gain
- Map spot colors
- Map colors
- DeviceLink Conversion
- Quick Color conversion
- Convert Colors
- Note: Objects that have already been converted within a profile are excluded from any further processing in that run. E.g. if CMYK objects are being adjusted by an "Adjust Dot Gain" fixup, they will remain untouched by a following "DeviceLink Conversion" fixup, when defined in one profile. If it is required to run both fixes on the same objects, the processing has to be done by two separate profiles, possibly as individual steps in a Process Plan.

7.2 Setting up a "Convert colors" Fixup

The "Convert colors" Fixup offers very powerful and flexible options for converting colors in many ways, allowing for highly customizable color conversion settings e.g. how different types of page objects are handled during color conversion and what the destination Profile of the color conversion process is. The Fixup consists of three sections. Let's take a look at each one.



1. Destination



Destination

In this list you can specify the destination profile that shall be used as the destination for the color conversion. In any case, this will only have an effect if 'Convert to destination' or 'Convert only alternate color space to destination' is chosen as Conversion in the "Conversion settings" area.

Use Destination Profile from Output Intent if present

An Output Intent has two meanings in color conversion:

1. Definition of the destination profile for color conversion (except DeviceLink)

2. Definition of the Output Intent for the resulting file

If this checkbox is activated and the processed PDF file contains an Output Intent, the profile chosen in the Destination list will be ignored and instead the one embedded in the Output Intent will be used.

Embedding

Do not embed

Converted objects are not tagged at all.

Embed as source profile

All converted objects will be tagged with the destination profile. This will hardly ever be useful for conversions to CMYK but is almost always very useful for conversions to RGB.

Embed as Output Intent for PDF/X

An Output Intent will be created for the PDF with the destination profile embedded as Output Intent profile.

- Note: This is highly recommended for any conversions to CMYK, as recent versions of Adobe Acrobat use that information by default for displaying on screen whereas at the same time this ICC profile in the Output Intent does not trigger any unwanted further color conversions.
- Note: Please be aware that only ICC profiles of type 'prtr' (output profiles) are allowed in an Output Intent for PDF/X files. Embedding for example an sRGB profile (which is a 'mntr' or display profile) in a PDF/X Output Intent will make it impossible to turn that file into a valid PDF/X file (or make it an invalid PDF/X file if it already had been a PDF/X file).

Assumed profiles

With this list you can specify the default profiles that will be used as the object's source profile if the object does not have a source profile. In any case, this will only have effect if "Convert to destination" or "Convert only alternate color space to destination" is chosen in the Conversion list under "Conversion settings". It is necessary to have valid ICC profiles specified for all color spaces. Selecting a predefined Adobe color

setting here will change the settings for the color spaces automatically.

- Note: If you import this Fixup and the used ICC profiles are not present on your system, you need to execute the Fixup once (otherwise the ICC profile may be displayed in "(...)", as it is not present on the current system yet).

RGB

List of available ICC profiles to be assumed for uncalibrated RGB colors.

CMYK

List of available ICC profiles to be assumed for uncalibrated CMYK colors.

Grayscale

List of available ICC profiles to be assumed for uncalibrated grayscale colors.

Output Intent takes precedence over assumed profile

If there is an Output Intent embedded in the processed file, the selected profile for the color space defined by the assumed Profile will be ignored and instead the settings will be adapted to the Output Intent profile.

Use black point compensation

If this checkbox is activated, black point compensation will be used for color conversion using the relative colorimetric rendering intent.

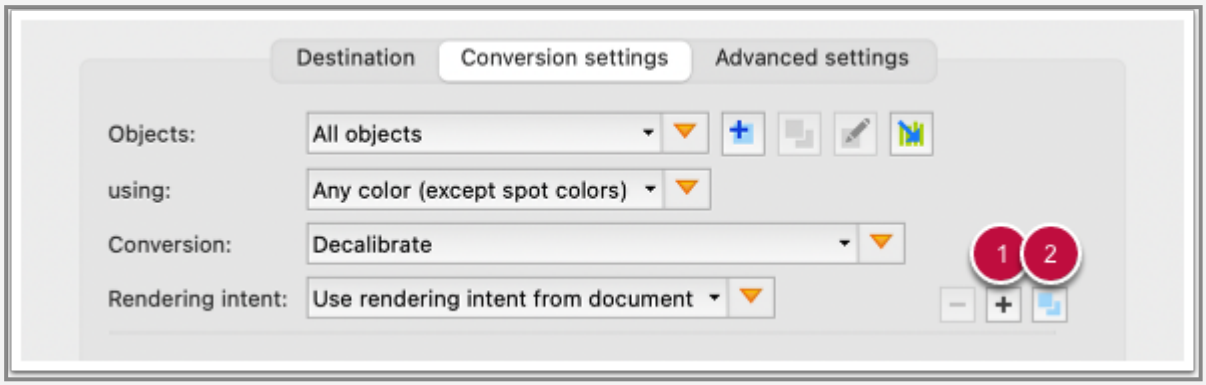
Preserve black objects

If this checkbox is activated, color definitions using just the black channel will remain black. Furthermore, RGB and Lab color definitions for gray will get converted to Black only

where applicable. For more information about the treatment of black objects please read the article: Processing black objects.

2. Conversion settings

You can define additional conversion settings for the various color spaces or object types by using the "Add setting" button (1). Since pdfToolbox 14 it is also possible to duplicate the respective set of configured parameters with the “Duplicate setting” button (2).



Objects

Here you can limit the execution of color conversion to a certain object type defined by a Check.

using

List of color spaces for which the defined conversion settings should be applied.

Conversion

You can choose one of the following options:

Decalibrate

Removes the source profile from a calibrated color space.

Note:

When pdfToolbox decalibrates ICCbased CMYK it will also disable the Overprint Mode (OPM) for the objects (if active beforehand). Background is that the OPM parameter has no effect on ICCbased CMYK and therefore disabling it avoids unexpected changes.

This approach has one limitation: If source profile and destination profile for an ICCbased CMYK object are identical the OPM parameter may actually have an effect. In such cases the above approach is not ideal and may require to establish OPM after the operation again.

Convert to destination

Objects are converted to the destination color space as specified by the destination settings.

Decalibrate, then convert

Removes the source profile from a calibrated color space first and converts the resulting, uncalibrated color into the destination color space.

Convert only alternate color space to destination

Only the alternate color space of spot colors is converted to the destination color space as specified by the destination settings.

Tag with ICC profile

Tags an object with the ICC profile of the destination if the source color space matches.

Do not convert

Objects will not be converted. This might be useful if you want to exclude a specific object type or color space from the color conversion.

Rendering intent

This list lets you define which Rendering Intent is used for a color conversion:

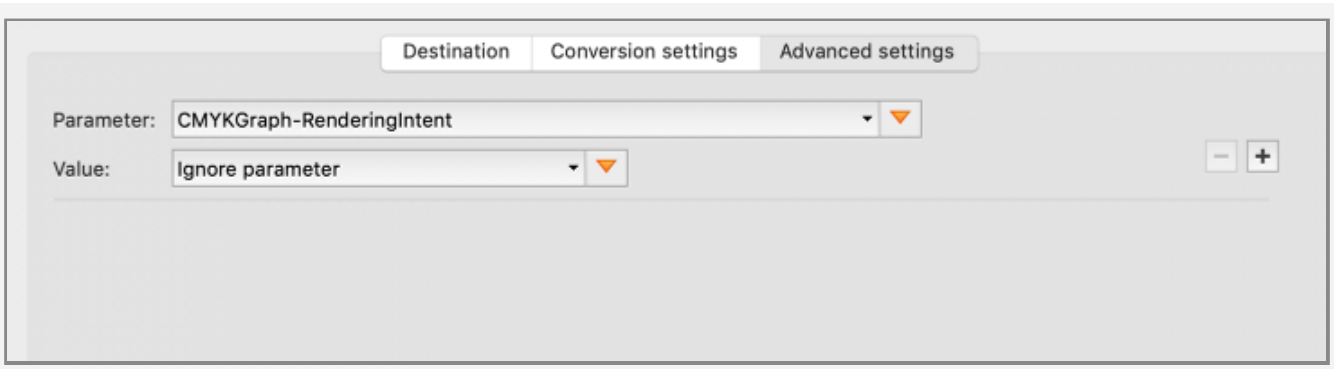
- Use rendering intent from document
- Perceptual
- Relative colorimetric
- Absolute colorimetric
- Saturation

Note:

- If there is no Rendering Intent explicitely defined in the PDF the default is Relative colorimetric.
- The setting has no effect for the conversion options "Decalibrate" and "Tag with ICC profile".

3. Advanced settings

The regular ICC based color conversion is extended by some useful options. This includes for example obtainment of pure black, recognition of RGB black or adaption of tone value increase. For more information about the treatment of black objects please read the article: Convert colors: Advanced settings.



Handling of process colors in Separation and Device N

As soon as you run a "Convert colors" Fixup, the alternate values of the following separation color spaces will be adapted automatically, to prevent problems in the output process:

Source	Destination
Separation/DeviceN Black	0% DeviceGray
Separation/DeviceN Cyan	100/0/0/0 DeviceCMYK
Separation/DeviceN Magenta	0/100/0/0 DeviceCMYK
Separation/DeviceN Yellow	0/0/100/0 DeviceCMYK
Separation "All"	0% DeviceGray
Separation “None"	100% DeviceGray

7.3 Convert colors: Advanced settings (previously "Policies")

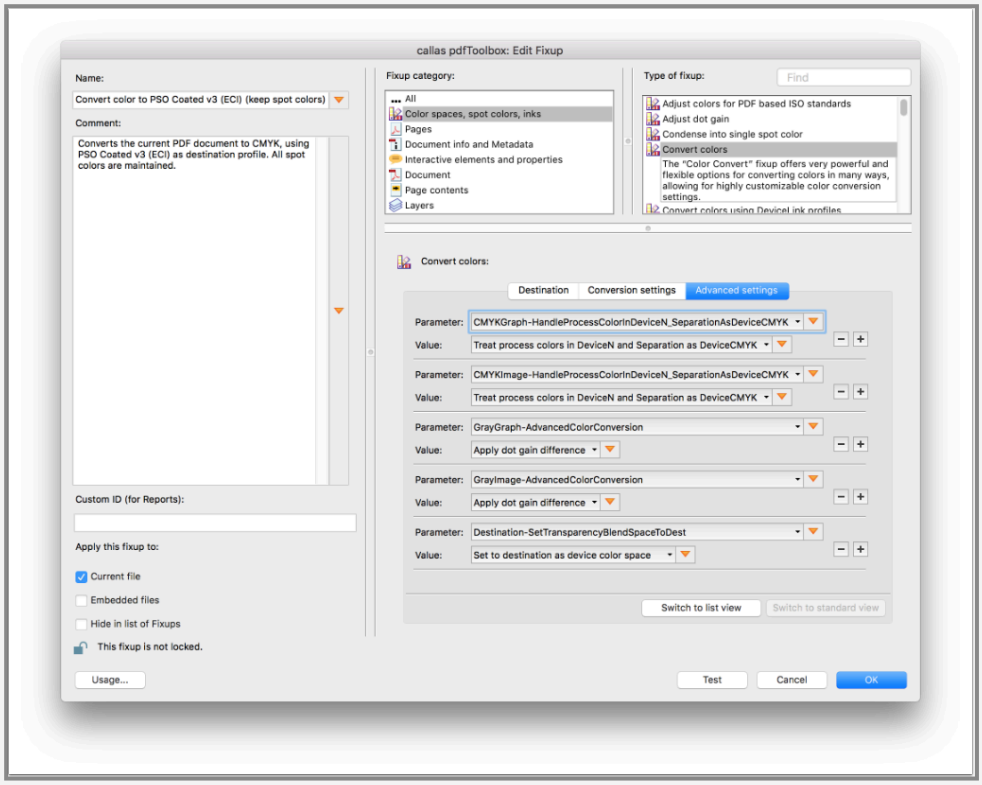
The regular ICC based color conversion is extended by some useful options. This includes obtainment of pure black, recognition of RGB black or adaption of tone value increase. Additional parameters for controlling these properties can be defined in the "Advanced settings" tab, that can be selected in the "Convert colors" Fixup.

You will find details on parameters and example configurations on the following pages.

Old, deprecated configuration using Policy files

Before pdfToolbox 11, these options were defined by config files, which were completely text based and could be modified with a simple text editor. Exported color conversion profiles were containing these additional files.
You'll find the documentation of this deprecated implementation here: Policies

Overview



All parameters defined in the "Advanced settings"-tab take precedence over all settings defined in the user interface. Please note that nonetheless only objects triggered due to the "Conversion settings" in the user interface will be changed.

The following parameters are available (please consult section Parameter for a detailed explanation):

- RenderingIntent
- C_eq_M_eq_Y_is_Black
- C_eq_M_eq_Y_is_Black_ExcludeBlendModes
- R_eq_G_eq_B_is_Black
- R_eq_G_eq_B_is_Black_tolerance
- R_eq_G_eq_B_is_Black_ExcludeBlendModes
- SetGrayColorSpaceTo
- a_eq_b_eq_0_is_Black
- a_eq_b_eq_0_is_Black_ExcludeBlendModes
- HandleProcessColorInDeviceN_SeparationAsDeviceCMYK
- AdvancedColorConversion
- CompressionMethod

- JPEGQuality
- SetTransparencyBlendSpaceToDest

Parameters

RenderingIntent

Keys

- CMYKGraph-RenderingIntent
- CMYKImage-RenderingIntent
- RGBGraph-RenderingIntent
- RGBImage-RenderingIntent
- GrayGraph-RenderingIntent
- GrayImage-RenderingIntent
- LabGraph-RenderingIntent
- LabImage-RenderingIntent

Values

Use Rendering Intent of document	Rendering Intent specified in the PDF is used (default)
RelativeColorimetic	Relative colorimetric
AbsoluteColorimetric	Absolute colorimetric
Perceptual	Perceptual
Saturation	Saturation

Description

This parameter defines which Rendering Intent is used for a color conversion. This parameter overrides the option Rendering Intent of the conversion settings.

- Note: If there is no Rendering Intent explicitely defined in the PDF the default is "Relative Colorimetric".

C_eq_M_eq_Y_is_Black

Keys

- CMYKGraph-C_eq_M_eq_Y_is_Black
- CMYKImage-C_eq_M_eq_Y_is_Black

Values

Ignore parameter	Convert as defined for color space CMYK (default)
DeviceGray	Use DeviceGray
DeviceCMYK Black	Use DeviceCMYK Black
Separation "Black"	Use Separation Black

If only the black channel is used, the object will be converted according to the settings for color space Gray and stored in the defined color space afterwards. This parameter overrides the option Preserve black objects of the destination settings for CMYK.

C_eq_M_eq_Y_isBlack_ExcludeBlendModes

Keys

- CMYKGraph-C_eq_M_eq_Y_is_Black_ExcludeBlendModes
- CMYKImage-C_eq_M_eq_Y_is_Black_ExcludeBlendModes

Values

<List of all Blend modes>	Name of Blend modes to exclude from color conversion To exclude multiple Blend modes, one setting for each blend mode has to be configured

Description

Objects, which are using a Blend mode listed in this parameter, will be processed using the normal color conversion and no preservation of black objects will take place.
This might be appropriate for objects (with empty colorant channels), that are using certain blend mode, which are using these empty channels to achieve a special effect with underlying objects.
If these empty channels would be replaced by a single colorant color space, these effects would not work anymore, so they must be preserved in some certain cases.

Please see article "Processing black objects" for further details.

R_eq_G_eq_B_is_Black

Keys

- RGBGraph-R_eq_G_eq_B_is_Black
- RGBImage-R_eq_G_eq_B_is_Black

Values

Ignore parameter	Convert as defined for color space RGB (default)

DeviceGray	Use DeviceGray
DeviceCMYK Black	Use DeviceCMYK Black
Separation "Black"	Use Separation Black

Description

This parameter only applies to device dependent RGB color – either if the object uses DeviceRGB or if it uses ICC based RGB or CalRGB but the parameter StripSourceProfile is set to On. It only has effect if ConvertToDestination is set to On. If not set to NoChange, instead of carrying out an ICC based color conversion, RGB colors where each of the Red Green and Blue values are equal, will be converted directly to the corresponding black tint value. For images this will only have an effect if all pixels in the image have equal Red, Green and Blue values.
The color conversion engine uses an internal tolerance of 3%, this mean the differences between R and G, G and B as well as for R and B must be <=3%.

R_eq_G_eq_B_is_Black_tolerance

Keys

- RGBGraph-R_eq_G_eq_B_is_Black_tolerance_light
- RGBImage-R_eq_G_eq_B_is_Black_tolerance_light
- RGBGraph-R_eq_G_eq_B_is_Black_tolerance_mid
- RGBImage-R_eq_G_eq_B_is_Black_tolerance_mid
- RGBGraph-R_eq_G_eq_B_is_Black_tolerance_dark
- RGBImage-R_eq_G_eq_B_is_Black_tolerance_dark

Values

<List of availible tolerance values>	Tolerance values between R and G, G and B and R and B interpreted as percentage points: +/- 0%, +/- 0.5%, +/- 1%, +/- 1.5%, +/- 2%, +/- 2.5%,

	+/- 3%, +/- 3.5%, +/- 4%, +/- 4.5%, +/- 5%

Description

These settings can be used for an improved and more controlled color conversion of gray RGB images or graphs to CMYK by allowing you to set individual tolerances to specify which RGB values should be considered neutral gray and which should not.

Because color differences are easier to see in lighter shades of RGB gray than in darker shades, separate tolerances can be set for light (0-20%), mid (40-60%), and dark (80-100%) shades. Everything in between will be interpolated. RGB grays below the tolerance are converted to separation black (100% K only), while RGB grays above the tolerance are converted to CMYK to preserve color representation.

R_eq_G_eq_B_is_Black_ExcludeBlendModes

Keys

- RGBImage-R_eq_G_eq_B_is_Black_ExcludeBlendModes
- RGBGraph-R_eq_G_eq_B_is_Black_ExcludeBlendModes

Values

Blend mode	Name of Blend modes to exclude from color conversion To exclude multiple Blend modes, one setting for each blend mode has to be configured

Description

Objects, which are using a Blend mode listed in this parameter, will be processed using the normal color conversion and no preservation of black objects will take place.
This might be appropriate for objects (with empty colorant channels), that are using certain blend mode, which are using these empty channels to achieve a special effect with underlying objects.
If these empty channels would be replaced by a single col-

orant color space, these effects would not work anymore, so they must be preserved in some certain cases.

Please see article "Processing black objects" for further details.

SetGrayColorSpaceTo

Keys

- GrayGraph-SetGrayColorSpaceTo
- GrayImage-SetGrayColorSpaceTo

Values

Keep color space for Gray	Convert as defined for color space Gray (default)
DeviceGray	Use DeviceGray
DeviceCMYK Black	Use DeviceCMYK Black
Separation "Black"	Use Separation Black

Description

This parameter applies to gray vector and image objects. It indicates which color space to use for gray objects after the conversion. For example, it may be necessary to encode a DeviceGray object as Separation Black object, so that overprinting also works for subjacent CMYK objects.

- Note: DeviceGray cannot overprint CMYK, even if overprint is set to true, whereas Separation Black does overprint Cyan, Magenta and Yellow once overprint is set to true.

- Note that both DeviceGray and Separation Black do overprint spot colors if overprint is set to true.

a_eq_b_eq_0_is_Black

Keys

- LabGraph-a_eq_b_eq_0_is_Black
- LabImage-a_eq_b_eq_0_is_Black

Values

Ignore parameter	Convert as defined for color space Lab (default)
DeviceGray	Use DeviceGray
DeviceCMYK Black	Use DeviceCMYK Black
Separation "Black"	Use Separation Black

Description

If a and b is 0 the luminosity value is directly transferred to the designated color space.
The color conversion engine uses an internal tolerance of 1%, this means the differences between L and a as well as for L and b must be <= 1%.

a_eq_b_eq_0_is_Black_ExcludeBlendModes

Keys

- LabGraph-a_eq_b_eq_0_is_Black_LimitToBlendModes
- LabImage-a_eq_b_eq_0_is_Black_LimitToBlendModes

Values

Blend mode	Name of Blend modes to exclude from color conversion To exclude multiple Blend modes, one setting for each blend mode has to be configured

Description

Objects, which are using a Blend mode listed in this parameter, will be processed using the normal color conversion and no preservation of black objects will take place.
This might be appropriate for objects (with empty colorant channels), that are using certain blend mode, which are using these empty channels to achieve a special effect with underlying objects.
If these empty channels would be replaced by a single colorant color space, these effects would not work anymore, so they must be preserved in some certain cases.

Please see article "Processing black objects" for further details.

HandleProcessColorInDeviceN_SeparationAsDeviceCMYK

Keys

- CMYKGraph-HandleProcessColorInDeviceN_SeparationAsDeviceCMYK
- CMYKImage-HandleProcessColorInDeviceN_SeparationAsDeviceCMYK

Values

Treat process colors defined in DeviceN or Separation as spot colors	Treat process colors defined in DeviceN or Separation as spot

	colors (default)
Treat process colors defined in DeviceN or Separation as DeviceCMYK	Treat process colors defined in DeviceN or Separation as DeviceCMYK

Description

This parameter defines the way process colors (Cyan, Magenta, Yellow, Black) in DeviceN or Separation are treated. If set to On, they are treated as DeviceCMYK objects. Not defined color channels are set to 0% in the respective color. To keep the overprint settings, the original color space (Separation or DeviceN) is recreated after the conversion if possible.

Use the AdvancedColorConversion parameters to define the treatment of newly added color channels.

AdvancedColorConversion

Keys

- CMYKGraph-AdvancedColorConversion
- CMYKImage-AdvancedColorConversion
- GrayGraph-AdvancedColorConversion
- GrayImage-AdvancedColorConversion

Values

No special treatment	Convert as defined for destination color space
Apply dot gain difference	Apply dot gain difference

Description

If set to "No special treatment", tints in colorants that were originally 0 are preserved.
Handles all black objects. Black objects are using only the K channel of CMYK or DeviceGray or Separation Black or DeviceN with only one channel named Black.
If necessary, new channels will be added for DeviceN (Separation color space becomes DeviceN after the conversion).
With AdjustDotGain the dot gain difference between source and destination profile is calcuated and all color values are modified so that the dot gain difference is compensated. This is useful mainly for standard PCS conversions.

CompressionMethod

Keys

- Destination-CompressionMethod

Values

Keep compression method	Keep compression method of original (default)
Compress all to ZIP	Compress all handled images with ZIP
Compress all to JPEG	Compress all handled images using JPEG

Description

Specifies the method for recompression of converted images.

- Note: All processed images are decompressed for color conversion tasks. If recompressing with JPEG the quality level is to be defined with the parameter `Destination-JPEGQuality`. Please keep in mind that every JPEG compression leads to the loss of quality.

JPEGQuality

Keys

- Destination-JPEGQuality

Values

Minimum	Minimum JPEG quality compression (20)
Low	Low JPEG quality compression (40)
Medium	Medium JPEG quality compression (60)
High	High JPEG quality compression (80)
Maximum	Maximum JPEG quality compression (100)

Description

Allows for specifying the quality level in which JPEG objects are saved after the conversion.

- Note: The predefined values are interpreted as a percentage declaration between 0 and 100. The value in brackets are giving a rough comparision to the equivalent setting in Adobe Photoshop.

SetTransparencyBlendSpaceToDest

Keys

- Destination-SetTransparencyBlendSpaceToDest

Values

Leave unchanged	Leaves transparency blend color space unchanged

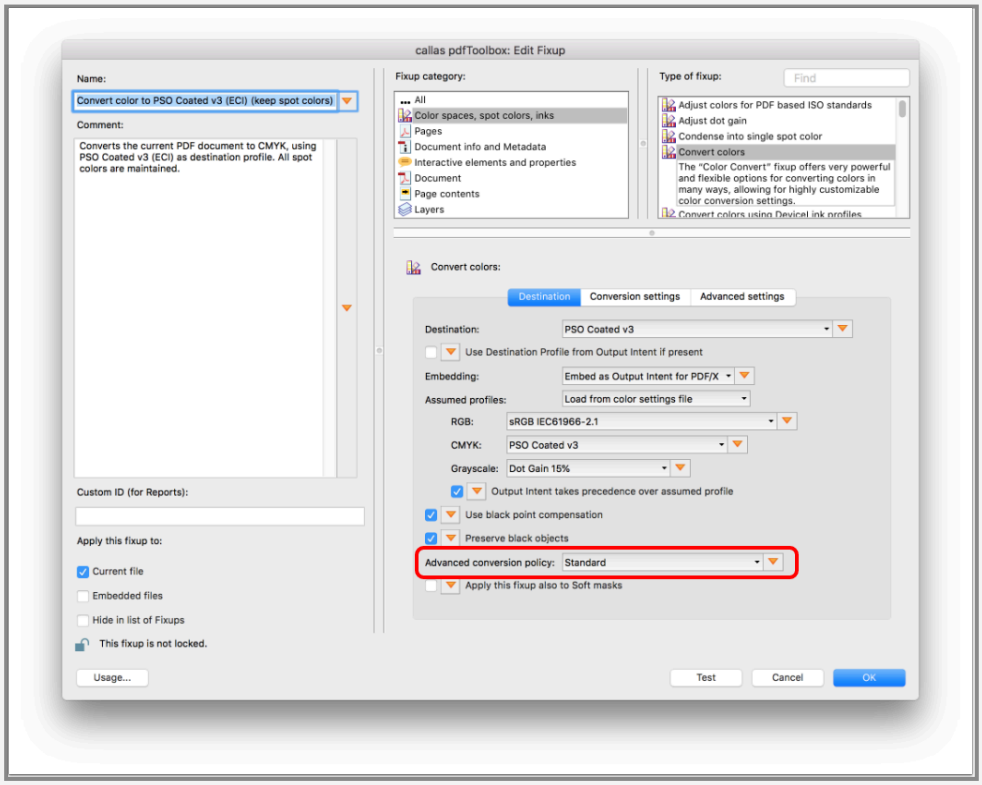
Set to destination ICC profile	Sets destination color space as transparency blend color space
Set to destination as device color space	Sets destination color space as transparency blend color space; if destination color space is ICC based, the respective device color space is set as transparency blend color space
Set to destination if equal to Output Intent	Sets destination color space as transparency blend color space; if destination color space is ICC based and the output intent matches the destination profile, the respective device color space is set as transparency blend color space

Description

Allows for specifying the transparency blend color space after the color conversion.

Implementation notes

When importing old "Convert colors"-Fixups, the contained Policy file will show up in the "Advanced conversion policy" Pop-Up in the "Destination"-tab.



It is possible to edit the Policy file using the parameters described on this page. You can even select another existing Policy file and export such Fixups

To avoid contradictionary settings, no configuration can be done in the "Advanced settings"-tab in such Fixups.

No matter if the old, deprecated Policy files or the new "Advanced settings" are used in the Fixup, both will work in pdfToolbox 11 and later.

Watch Dietrich von Seggern talk about 'Convert colors: Advanced settings' in the video below:

7.4 Policies (deprecated, available till v10.2)

The regular ICC based color conversion is extended by some useful options. This includes obtainment of pure black, recognition of RGB black or adaption of tone value increase. Additional parameters for controlling these properties can be defined in "Policy" files, that can be selected in the "Convert colors" Fixup. These config files are completely text based and can be modified with a simple text editor. Exported color conversion profiles contain these additional files. You can find the available Policy files in the following directories:

```
%PROGRAMFILES%\callas pdfToolbox <version>\ColorConversion\Policies
```

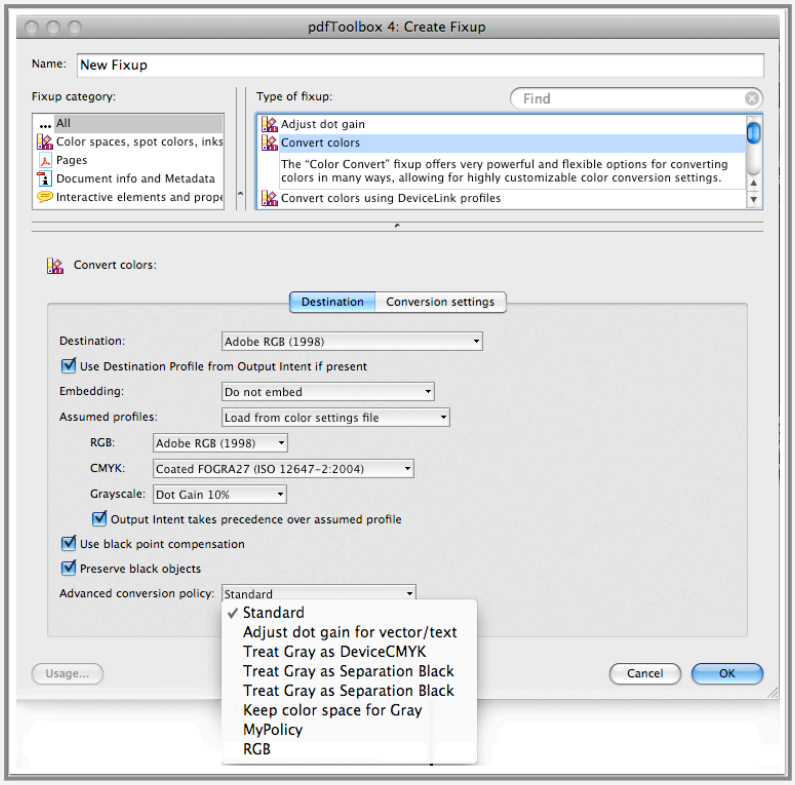
```
%PROGRAMFILES%\callas pdfaPilot <version>\ColorConversion\Policies
```

You will find details on parameters and example configurations on the following pages. When editing a color policy file, please make sure to a `<tab>` between the name of the parameter and the value. The use of a `<space>` character will result in a color conversion failure. A "`#`" at the beginning of a line indicates a comment and prevents the line from being processed.

Excerpt from a policy file:

```
DisplayName <tab> 1 <tab> MyPolicy
CMYKGraph-StripSourceProfile <tab> 1
CMYKGraph-C_eq_M_eq_Y_is_Black <tab> CMYK_Black
CMYKImage-PassThroughDeviceColor <tab> 1
GrayGraph-AdvancedColorConversion <tab> AdjustDotGain
GrayGraph-SetGrayColorSpaceTo <tab> Separation_Black
...
```

Overview



All parameters defined in color policy files take precedence over all settings defined in the user interface. Please note that nonetheless only objects triggered due to the Conversion settings in the user interface will be changed.

The following parameters are available (please consult section Parameter for a detailed explanation):

- StripSourceProfile
- ConvertToDestination
- RenderingIntent
- BlackPointCompensation
- ProfileSourceSelector
- PassThroughDeviceColor
- C_eq_M_eq_Y_is_Black
- C_eq_M_eq_Y_is_Black_ExcludeBlendModes
- R_eq_G_eq_B_is_Black
- R_eq_G_eq_B_is_Black_ExcludeBlendModes
- SetGrayColorSpaceTo
- a_eq_b_eq_0_is_Black
- a_eq_b_eq_0_is_Black_ExcludeBlendModes

- HandleProcessColorInDeviceN_SeparationAsDeviceCMYK
- AdvancedColorConversion_Primaries
- AdvancedColorConversion_Secondaries
- AdvancedColorConversion_Tertiaries
- AdvancedColorConversion_All
- AdvancedColorConversion
- CompressionMethod
- JPEGQuality
- SetTransparencyBlendSpaceToDest

Conversion parameters can be influenced by modifying a static color policy file. The location of this file is:

```
%PROGRAMFILES%\callas pdfToolbox version\etc\ColorConversion\Policies DeviceLink\
Policy_DL_Default.cfg
```

```
%PROGRAMFILES%\callas pdfToolbox Server version\cli\etc\ColorConversion\Policies
DeviceLink\Policy_DL_Default.cfg
```

This applies for all DeviceLink conversions and is not part of the kfpx.

Example of a policy file

Requirements

- Convert line art elements defined in RGB (calibrated and uncalibrated) using Quick Conversion
- Decalibrate calibrated CMYK
- Convert images using RGB to CMYK (Iso Coated v2 (ECI)), use sRGB as source profile uncalibrated RGB

Fixup settings

Convert Colors

Destination

- Destination: ISOcoated v2 (ECI)
- Assumed CMYK Profile: ISOcoated v2 (ECI)
- Assumed RGB Profile: sRGB
- Assumed Grayscale Profile: Dot Gain 15%
- Use black point compensation
- Preserve black objects
- Policy: Office_Conversion_Policy

Conversion Setting

- Objects: All Objects
- using: Any color (except spot colors)
- Conversion: Convert to destination

Settings of the Office_Conversion_Policy file

```
DisplayName 1 Office_Conversion_Policy

# Source parameters for CMYK vector objects
CMYKGraph-StripSourceProfile 1
CMYKGraph-ConvertToDestination
CMYKGraph-HandleProcessColorInDeviceN_SeparationAsDeviceCMYK 1
CMYKGraph-PassThroughDeviceColor 1

# Source parameters for CMYK images CMYKImage-StripSourceProfile 1 CMYKImage-Con-
vertToDestination 0
CMYKImage-HandleProcessColorInDeviceN_SeparationAsDeviceCMYK 1
CMYKImage-PassThroughDeviceColor 1

# Source parameters for RGB vector objects
RGBGraph-StripSourceProfile 1
RGBGraph-MakeQuickConversion 1
RGBGraph-UndercolorRemoval 0.5
```

```
RGBGraph-BlackGeneration 0.5

# Source parameters for RGB images
RGBImage-StripSourceProfile 0
RGBImage-ConvertToDestination 1

# Source parameters for Gray vector objects
GrayGraph-ConvertToDestination 0
GrayGraph-AdvancedColorConversion Off
GrayGraph-SetGrayColorSpaceTo 0

# Source parameters for Gray images
GrayImage-ConvertToDestination 0
GrayImage-AdvancedColorConversion Off
GrayImage-SetGrayColorSpaceTo 0
```

Parameters

StripSourceProfile

Keys

- CMYKGraph-StripSourceProfile
- CMYKImage-StripSourceProfile
- RGBGraph-StripSourceProfile
- RGBImage-StripSourceProfile
- GrayGraph-StripSourceProfile
- GrayImage-StripSourceProfile

Values

Off	Leave source profiles unchanged (default)
On	Remove source profiles

Description

If set to On source profiles are removed during conversion. This also applies if ConvertToDestination is set to Off. This parameter overrides the conversion option for calibrated colors of the conversion settings.

- Note: This parameter has a special implication for Lab color spaces:

Strictly speaking the 'source profile' for a Lab color space cannot be stripped or removed. Nevertheless, if StripSourceProfile is set to On, the Lab profile specified in Assumed profiles will be used instead – so this is a means to replace a Lab color space for an object with some other Lab color space.

ConvertToDestination

Keys

- CMYKGraph-ConvertToDestination
- CMYKImage-ConvertToDestination
- RGBGraph-ConvertToDestination
- RGBImage-ConvertToDestination
- GrayGraph-ConvertToDestination
- GrayImage-ConvertToDestination
- LabGraph-ConvertToDestination
- LabImage-ConvertToDestination

Values

Off	Do not convert to destination color space (default)
On	Convert to destination color space

Description

If set to On the object is converted to the destination color space as specified in the settings. If the object does not have a source profile, or if the parameter StripSourceProfile is set to On, the profile specified in Assumed profiles will be used as the source profile. If the object has a source profile, and StripSourceProfile is set to Off, the object's source profile will be used.

RenderingIntent

Keys

- CMYKGraph-RenderingIntent
- CMYKImage-RenderingIntent
- RGBGraph-RenderingIntent
- RGBImage-RenderingIntent
- GrayGraph-RenderingIntent
- GrayImage-RenderingIntent
- LabGraph-RenderingIntent
- LabImage-RenderingIntent

Values

UseDoc	Rendering Intent specified in the PDF is used (default)
RelCol	Relative colorimetric
AbsCol	Absolute colorimetric
Perc	Perceptual
Satur	Saturation

Description

This parameter defines which Rendering Intent is used for a color conversion. This parameter overrides the option Rendering Intent of the conversion settings.

- Note: If there is no Rendering Intent explicitely defined in the PDF the default is relative colorimetric.

BlackPointCompensation

Keys

- CMYKGraph-BlackPointCompensation
- CMYKImage-BlackPointCompensation
- RGBGraph-BlackPointCompensation
- RGBImage-BlackPointCompensation
- GrayGraph-BlackPointCompensation
- GrayImage-BlackPointCompensation
- LabGraph-BlackPointCompensation
- LabImage-BlackPointCompensation

Values

Off	Do not use Black Point Compensation
On	Use Black Point Compensation (default)

Description

If the parameter is set to On, Black Point Compensation is used for the color conversion when performing an ICC based color conversion using the relative colorimetric Rendering Intent. This parameter overrides the option Use black point compensation of the destination settings.

ProfileSourceSelector

Keys

- CMYKGraph-ProfileSourceSelector
- CMYKImage-ProfileSourceSelector

Values

FromDefaultProfile	Use default profile (default)
FromEmbeddedOutputIntent	Use profile from embedded Output Intent

Description

If set to FromEmbeddedOutputIntent the source profile will be taken from an embedded Output Intent for DeviceCMYK objects. (This parameter overrides the parameter Assumed profiles). If no Output Intent is embedded an error is returned. This parameter overrides the option Output Intent takes precendence over assumed profiles of the destination settings.

PassThroughDeviceColor

Keys

- CMYKGraph-PassThroughDeviceColor
- CMYKImage-PassThroughDeviceColor
- RGBGraph-PassThroughDeviceColor
- RGBImage-PassThroughDeviceColor
- GrayGraph-PassThroughDeviceColor
- GrayImage-PassThroughDeviceColor
- LabGraph-PassThroughDeviceColor

- LabImage-PassThroughDeviceColor

Values

Off	Convert all objects as defined in the conversion settings (default)
On	Convert only objects using a calibrated color space

Description

This parameter disables color conversion for objects that do not have a source profile. Thus, if both ConvertToDestination and PassThroughDeviceColor are set to On, only objects that have a source profile will be converted. Please be aware, that in this regard if StripSourceProfile is set to On the object will be considered as having no source profile (in essence setting both StripSourceProfile and PassThroughDeviceColor to On has the same effect as setting ConvertToDestination to Off). If ConvertToDestination is set to Off, PassThroughDeviceColor does not have any effect.

C_eq_M_eq_Y_is_Black

Keys

- CMYKGraph-C_eq_M_eq_Y_is_Black
- CMYKImage-C_eq_M_eq_Y_is_Black

Values

NoChange	Convert as defined for color space CMYK (default)
DeviceGray	Use DeviceGray
CMYK_Black	Use DeviceCMYK Black

Separation_Black	Use Separation Black

If only the black channel is used, the object will be converted according to the settings for color space Gray and stored in the defined color space afterwards. This parameter overrides the option Preserve black objects of the destination settings for CMYK.

C_eq_M_eq_Y_isBlack_ExcludeBlendModes

Keys

- CMYKGraph-C_eq_M_eq_Y_is_Black_ExcludeBlendModes
- CMYKImage-C_eq_M_eq_Y_is_Black_ExcludeBlendModes

Values

Blend mode	Name of Blend modes to exclude from color conversion, multiple values have to be <TAB> separated, e.g.: ColorDodge ColorBurn

Description

Objects, which are using a blend mode listed in this parameter, will be processed using the normal color conversion and no preservation of black objects will take place.
This might be appropriate for objects (with empty colorant channels), that are using certain blend mode, which are using these empty channels to achieve a special effect with underlying objects.
If these empty channels would be replaced by a single colorant color space, these effects would not work anymore, so they must be preserved in some certain cases.

Please see article "Processing black objects" for further details.

R_eq_G_eq_B_is_Black

Keys

- RGBGraph-R_eq_G_eq_B_is_Black
- RGBImage-R_eq_G_eq_B_is_Black

Values

NoChange	Convert as defined for color space RGB (default)
DeviceGray	Use DeviceGray
CMYK_Black	Use DeviceCMYK Black
Separation_Black	Use Separation Black

Description

This parameter only applies to device dependent RGB color – either if the object uses DeviceRGB or if it uses ICC based RGB or CalRGB but the parameter StripSourceProfile is set to On. It only has effect if ConvertToDestination is set to On. If not set to NoChange, instead of carrying out an ICC based color conversion, RGB colors where each of the Red Green and Blue values are equal, will be converted directly to the corresponding black tint value. For images this will only have an effect if all pixels in the image have equal Red, Green and Blue values.
The color conversion engine uses an internal tolerance of 3%, this mean the differences between R and G, G and B as well as for R and B must be <=3%.

R_eq_G_eq_B_is_Black_ExcludeBlendModes

Keys

- RGBImage-R_eq_G_eq_B_is_Black_ExcludeBlendModes
- RGBGraph-R_eq_G_eq_B_is_Black_ExcludeBlendModes

Values

Blend mode	Name of Blend modes to exclude from color conversion, multiple values have to be <TAB> separated, e.g.: ColorDodge ColorBurn

Description

Objects, which are using a blend mode listed in this parameter, will be processed using the normal color conversion and no preservation of black objects will take place.
This might be appropriate for objects (with empty colorant channels), that are using certain blend mode, which are using these empty channels to achieve a special effect with underlying objects.
If these empty channels would be replaced by a single colorant color space, these effects would not work anymore, so they must be preserved in some certain cases.

Please see article "Processing black objects" for further details.

SetGrayColorSpaceTo

Keys

- GrayGraph-SetGrayColorSpaceTo
- GrayImage-SetGrayColorSpaceTo

Values

NoChange	Convert as defined for color space Gray (default)
DeviceGray	Use DeviceGray
CMYK_Black	Use DeviceCMYK Black
Separation_Black	Use Separation Black

Description

This parameter applies to gray vector and image objects. It indicates which color space to use for gray objects after the conversion. For example, it may be necessary to encode a DeviceGray object as Separation Black object, so that overprinting also works for subjacent CMYK objects.

- Note: DeviceGray cannot overprint CMYK, even if overprint is set to true, whereas Separation Black does overprint Cyan, Magenta and Yellow once overprint is set to true.
- Note that both DeviceGray and Separation Black do overprint spot colors if overprint is set to true.

a_eq_b_eq_0_is_Black

Keys

- LabGraph-a_eq_b_eq_0_is_Black
- LabImage-a_eq_b_eq_0_is_Black

Values

NoChange	Convert as defined for color space Lab (default)
DeviceGray	Use DeviceGray
CMYK_Black	Use DeviceCMYK Black
Separation_Black	Use Separation Black

Description

If a and b is 0 the luminosity value is directly transferred to the designated color space.
The color conversion engine uses an internal tolerance of 1%, this means the differences between L and a as well as for L and b must be <= 1%.

a_eq_b_eq_0_is_Black_ExcludeBlendModes

Keys

- LabGraph-a_eq_b_eq_0_is_Black_LimitToBlendModes
- LabImage-a_eq_b_eq_0_is_Black_LimitToBlendModes

Values

Blend mode	Name of Blend modes to exclude from color conversion, multiple values have to be <TAB> separated, e.g.: ColorDodge ColorBurn

Description

Objects, which are using a blend mode listed in this parameter, will be processed using the normal color conversion and no preservation of black objects will take place.
This might be appropriate for objects (with empty colorant channels), that are using certain blend mode, which are using these empty channels to achieve a special effect with underlying objects.
If these empty channels would be replaced by a single colorant color space, these effects would not work anymore, so they must be preserved in some certain cases.

Please see article "Processing black objects" for further details.

HandleProcessColorInDeviceN_SeparationAsDeviceCMYK

Keys

- CMYKGraph-HandleProcessColorInDeviceN_SeparationAsDeviceCMYK
- CMYKImage-HandleProcessColorInDeviceN_SeparationAsDeviceCMYK

Values

Off	Treat process colors defined in DeviceN or Separation as spot colors (default)
On	Treat process colors defined in DeviceN or Separation as DeviceCMYK

Description

This parameter defines the way process colors (Cyan, Magenta, Yellow, Black) in DeviceN or Separation are treated. If set to On, they are treated as DeviceCMYK objects. Not defined color channels are set to 0% in the respective color. To keep the overprint settings, the original color space (Separation or DeviceN) is recreated after the conversion if possible.

Use the AdvancedColorConversion parameters to define the treatment of newly added color channels.

AdvancedColorConversion_Primaries

Keys

- CMYKGraph-AdvancedColorConversion_Primaries
- CMYKImage-AdvancedColorConversion_Primaries

Values

Off	No special treatment for objects using only one colorant (not K) (default)
AdjustDotGain	Apply dot gain difference
KeepZeroChannels	Restores zero channels after conversion

Description

If set to Off, tints in colorants that were originally 0 are preserved. If necessary, new channels will be added for DeviceN (Separation color space becomes DeviceN after the conversion). With AdjustDotGain the dot gain difference between source and destination profile is calcuated and all color values are modified so that the dot gain difference is compensated. This is usefull mainly for standard PCS conversions. KeepZeroChannels causes all channels which were 0 before the conversion to be set back to 0. This might lead to a color shift.

AdvancedColorConversion_Secondaries

Keys

- CMYKGraph-AdvancedColorConversion_Secondaries
- CMYKImage-AdvancedColorConversion_Secondaries

Values

Off	No special treatment for objects using only two colorants (default)
AdjustDotGain	Apply dot gain difference
KeepZeroChannels	Restores zero channels after conversion

Description

Same as AdvancedColorConversion_Primaries but for all objects using exactly two colorants of CMYK.

AdvancedColorConversion_Tertiaries

Keys

- CMYKGraph-AdvancedColorConversion_Tertiaries
- CMYKImage-AdvancedColorConversion_Tertiaries

Values

Off	No special treatment for objects using only three colorants (default)
AdjustDotGain	Apply dot gain difference
KeepZeroChannels	Restores zero channels after conversion

Description

Same as AdvancedColorConversion_Primaries but for all objects using exactly three colorants of CMYK.

AdvancedColorConversion_All

Keys

- CMYKGraph-AdvancedColorConversion_All
- CMYKImage-AdvancedColorConversion_All

Values

Off	No special treatment for objects using all

	four colorants (default)
AdjustDotGain	Apply dot gain difference

Description

Same as AdvancedColorConversion_Primaries but for all objects using all channels of CMYK (none of the channels is zero).

AdvancedColorConversion

Keys

- GrayGraph-AdvancedColorConversion
- GrayImage-AdvancedColorConversion

Values

Off	Convert as defined for color space Gray (default)
AdjustDotGain	Apply dot gain difference

Description

Same as AdvancedColorConversion_Primaries but for all black objects. Black objects are using only the K channel of CMYK or DeviceGray or Separation Black or DeviceN with only one channel named Black.

CompressionMethod

Keys

- Destination-CompressionMethod

Values

KeepCompressionMethod	Keep compression method of original (default) Also "0" can be used instead of the string
AllToZip	All to Zip Also "1" can be used instead of the string
AllToJpeg	All to Jpeg Also "2" can be used instead of the string

Description

Specifies the method for recompression of converted images.

- Note: All processed images are decompressed for color conversion tasks. If recompressing with JPEG the quality level is to be defined with the parameter `JPEGQuality`. Please keep in mind that every JPEG compression leads to the loss of quality.

JPEGQuality

Keys

- Destination-JPEGQuality

Values

0..100 (Highest image	(default value: 80)

quality: 100)	

Description

Allows for specifying the quality level in which JPEG objects are saved after the conversion.

- Note: The given value is interpreted as a percentage declaration between 0 and 100. Following you find the possible values and the equivalent setting in Adobe Photoshop:

20	Minimum
40	Low
60	Medium
80	High
100	Maximum

Other values are not supported and will be treated like the next higher supported value.

SetTransparencyBlendSpaceToDest

Keys

- Destination-SetTransparencyBlendSpaceToDest

Values

LeaveUnchanged	Leaves transparency blend color space unchanged
DestWithProfile	Sets destination color space as transparency blend color space
DestWithoutProfile	Sets destination color space as transparency blend color space; if destination color space is ICC based, the respective device color space is set as transparency blend color space

DestWithoutProfileIfMatchingOutputIntent	Sets destination color space as transparency blend color space; if destination color space is ICC based and the output intent matches the destination profile, the respective device color space is set as transparency blend color space

Description

Allows for specifying the transparency blend color space after the color conversion.

7.5 Processing black objects with Advanced settings (v11.0)

General

Black and tinted black (Gray) objects in PDF files can be rendered in different ways.

Common color spaces

DeviceGray

Gray color space with 1 channel

0% represents black in this color space

DeviceCMYK

CMYK color space with 4 channels

C, M and Y are 0 %, where only the black (K) channel is used

Separation Black

Color space with 1 channel

Uses only the K channel of the process colors for output

DeviceN Black

Color space with 1 channel

Uses only the K channel of the process colors for output

Other color spaces

There also is a special form of black in the following color spaces:

DeviceRGB

If all color components have the same value, the visual representation is tinted black.

Lab

If the color values for a and b equal 0, the visual representation is tinted black.

ICC based color spaces

Additionally there are the following color spaces:

ICCbasedGray, ICCbasedCMYK, ICCbasedRGB

When processing black it is recommended to ignore ICC profiles, since this would lead to a conversion of black using the PCS of the ICC profile resulting in rich black defined in CMYK (assuming the destination profile to be a CMYK profile).

Default handling

Depending on the color space the black is defined in, it will normally be processed like all the other colors in this color space; e.g. a black in DeviceCMYK will be processed like the other CMYK values.

Within the scope of print production though it is preferable to treat every black the same, no matter how it is defined internally. To process black elements properly according to the settings in the gray conversion, the following parameters are available:

- Gray....._SetGrayColorSpace
- RGB....._R_eq_G_eq_B_is_Black
- Lab....._a_eq_b_eq_0_is_Black
- CMYK....._C_eq_M_eq_Y_is_Black

Note:

Please keep in mind that some of the parameters mentioned in this article have a prefix (e.g. "RGBImage-..." or "CMYK-Graph-...") which will be shown in the "Advanced settings"-

tab of the "Convert colors"-Fixup.
Details can be found in the article: Advanced settings for Convert colors

When is black considered as black?

The color conversion engine uses a small tolerance when detecting black objects. So the objects considered as "black" might have a slightly different colorant value, but as their visual appearance will be gray when printed, they'll be handled the same way as if there colorant values were completely the same (also called "gray balance").

The used tolerances for the different colorspaces are:

Lab:
Tolerance = 1%
((|L - a| <= 0.01) & (|L - b| <= 0.01)) = Gray

RGB and calRGB:
Tolerance = 3%
((|R - G| <= 0.03) & (|G - B| <= 0.03) & (|R - B| <= 0.03)) = Gray

CMYK:
Tolerance = 0%
(C == 0) & (M == 0) & (Y == 0) = Gray

Setting up

Behavior of "Black" defined in common color spaces during color conversion

In the usual cases it is desirable, that black/gray elements are only represented in the K channel after any conversion.
Therefore you may use the option "Preserve Black" in the Fixup "Convert colors". This guarantees that all black objects are converted according to the gray treatment as required.

The parameters set by the option "Preserve Black" are:

- GrayGraph_SetGrayColorSpaceTo Separation "Black"
- GrayImage_SetGrayColorSpaceTo Separation "Black"
- CMYKGraph_C_eq_M_eq_Y_is_Black Separation "Black"

- CMYKImage_C_eq_M_eq_Y_is_Black Separation "Black"

The parameters set in most predefined Convert colors Fixups in pdfToolbox (overruling the parameters set by the option "Preserve Black") are:

- GrayGraph_SetGrayColorSpace Keep color space for Gray
- GrayImage_SetGrayColorSpace Keep color space for Gray
- Gray_AdvancedColorConversion Apply dot gain difference

Elements defined in DeviceGray remain DeviceGray after conversion. Gray elements defined in DeviceCMYK/Separation Black or DeviceN Black are mapped to Separation Black after the conversion.

NOTE:
It is recommended to define the 3 parameters also when setting up a new, own "Convert colors"-Fixup with the activated "Preserve Black" option.

Conversion of RGB/Lab Black/Gray to CMYK

The parameters set by the option "Preserve Black" are:

- RGBGraph_R_eq_G_eq_B_is_Black Separation "Black"
- RGBImage_R_eq_G_eq_B_is_Black Separation "Black"
- LabGraph_a_eq_b_eq_0_is_Black Separation "Black"
- LabImage_a_eq_b_eq_0_is_Black Separation "Black"

The option "Preserve Black" converts any RGB and/or Lab black to Separation Black, if objects of this color space are converted to CMYK. This conversion applies for images as well as for text and vector objects. During this conversion RGB gray scales are mapped 1:1 to Separation Black. This means, a 50% RGB gray (127/127/127) will be 50% Separation Black after conversion.

Tone value adjustments for conversions amongst different printing conditions

If files are converted amongst different printing conditions the different dot gain might be counterbalanced. The follow-

ing parameters (which should be used in most cases and are contained in almost all predefined "Convert colors"-Fixups) allow to compensate the different dot gain:

- GrayGraph-AdvancedColorConversion Apply dot gain difference
- GrayImage-AdvancedColorConversion Apply dot gain difference

The difference in dot gain between the gray source ICC profile and the destination ICC profile will be compensated. 100% Black remains unchanged.

Changing the destination color space of Gray

By changing the parameters in the "Advanced settings"-tab, it is possible to define the destination color space of gray elements with the following parameters:

- Gray....._SetGrayColorSpace
- RGB....._R_eq_G_eq_B_is_Black
- Lab....._a_eq_b_eq_0_is_Black
- CMYK....._C_eq_M_eq_Y_is_Black

Ignore parameter	Element will be converted according to the settings for the corresponding color space
DeviceGray	Element will be converted according to the settings for the color space "Gray" and stored in DeviceGray
DeviceCMYK Black	Element will be converted according to the settings for the color space "Gray" and stored in DeviceCMYK – Black channel only
Separation "Black"	Element will be converted according to the settings for the color space "Gray" and stored in Separation Black
Keep color space for Gray	Convert as defined for color space Gray (default) (only available for "Gray....._SetGrayColorSpaceTo")

- Attention: In DeviceGray, a defined Overprinting flag will have no influence on subjacent CMYK elements. These will always be knocked out.

Preserve black: Exclude certain BlendModes from special

handling (pdfToolbox 9.2)

Some special transparency effects are based on the interaction of color channels and therefore, a conversion of objects using a gray balance of a process color (e.g. RGB, CMYK or Lab) to pure gray scale may result in visually changed results. As these effects differ from the "construction", the used BlendMode and color space of a PDF a general solution is not possible and excluding certain BlendModes depending on the input files is necessary.

- RGBImage-R_eq_G_eq_B_is_Black_ExcludeBlendModes
- RGBGraph-R_eq_G_eq_B_is_Black_ExcludeBlendModes
- CMYKGraph-C_eq_M_eq_Y_is_Black_ExcludeBlendModes
- CMYKImage-C_eq_M_eq_Y_is_Black_ExcludeBlendModes
- LabGraph-a_eq_b_eq_0_is_Black_ExcludeBlendModes
- LabImage-a_eq_b_eq_0_is_Black_ExcludeBlendModes

All BlendModes can be excluded, which means they are converted to a process color space. Normally the gray balance is preserved by converting it to grayscale, but this can result in a changed visual appearance with some special usage of certain BlendModes (e.g. ColorDodge).

Possible values are: Normal, Compatible, Multiply, Screen, Overlay, Darken, Lighten, ColorDodge, ColorBurn, HardLight, SoftLight, Difference, Exclusion, Hue, Saturation, Color or Luminosity.
If more than one BlendMode shall be excluded, the values have to be separated either by a space, a comma or a semicolon.

7.6 Processing black objects (deprecated, available till v10.2)

General

Black and tinted black (Gray) objects in PDF files can be rendered in different ways.

Common color spaces

DeviceGray

Gray color space with 1 channel

0% represents black in this color space

DeviceCMYK

CMYK color space with 4 channels

C, M and Y are 0 %, where only the black (K) channel is used

Separation Black

Color space with 1 channel

Uses only the K channel of the process colors for output

DeviceN Black

Color space with 1 channel

Uses only the K channel of the process colors for output

Other color spaces

There also is a special form of black in the following color spaces:

DeviceRGB

If all color components have the same value, the visual representation is tinted black.

Lab

If the color values for a and b equal 0, the visual representation is tinted black.

ICC based color spaces

Additionally there are the following color spaces:

ICCbasedGray, ICCbasedCMYK, ICCbasedRGB

When processing black it is recommended to ignore ICC profiles, since this would lead to a conversion of black using the PCS of the ICC profile resulting in rich black defined in CMYK (assuming the destination profile to be a CMYK profile).

Default handling

Depending on the color space the black is defined in, it will normally be processed like all the other colors in this color space; e.g. a black in DeviceCMYK will be processed like the other CMYK values.

Within the scope of print production though it is preferable to treat every black the same, no matter how it is defined internally. To process black elements properly according to the settings in the gray conversion, the following parameters are available:

- SetGrayColorSpace
- R_eq_G_eq_B_is_Black
- a_eq_b_eq_0_is_Black
- C_eq_M_eq_Y_is_Black

Note:

Please keep in mind that some of the parameters mentioned in this article have to be supplemented with a prefix (e.g. "RGBImage-..." or "CMYKGraph-...") before they can be used

in the policy file to ensure correct processing.
Details can be found in the article: Policies

When is black considered as black?

The color conversion engine uses a small tolerance when detecting black objects. So the objects considered as "black" might have a slightly different colorant value, but as their visual appearance will be gray when printed, they'll be handled the same way as if there colorant values were completely the same (also called "gray balance").

The used tolerances for the different colorspaces are:

Lab:
Tolerance = 1%
((|L - a| <= 0.01) & (|L - b| <= 0.01)) = Gray

RGB and calRGB:
Tolerance = 3%
((|R - G| <= 0.03) & (|G - B| <= 0.03) & (|R - B| <= 0.03)) = Gray

CMYK:
Tolerance = 0%
(C == 0) & (M == 0) & (Y == 0) = Gray

Setting up

Behavior of "Black" defined in common color spaces during color conversion using policy "Default"

In the usual cases it is desirable, that black/gray elements are only represented in the K channel after any conversion. Therefore you may use the option "Preserve Black" in the fix-up "Convert colors". This guarantees that all black objects are converted according to the gray treatment defined in the color policy file.

The parameters set by the option "Preserve Black" are:

- SetGrayColorSpace Separation_Black
- C_eq_M_eq_Y_is_Black Separation_Black

The parameters set in the "Standard" color policy file (overruling the parameters set by the option "Preserve Black") are:

- SetGrayColorSpace NoChange
- Gray_AdvancedColorConversion AdjustDotGain

Elements defined in DeviceGray remain DeviceGray after conversion. Gray elements defined in DeviceCMYK/Separation Black or DeviceN Black are mapped to Separation Black after the conversion.

Conversion of RGB/Lab Black/Gray to CMYK

The parameters set by the option "Preserve Black" are:

- R_eq_G_eq_B_is_Black Separation_Black
- a_eq_b_eq_0_is_Black Separation_Black

The option "Preserve Black" converts any RGB and/or Lab black to Separation Black, if objects of this color space are converted to CMYK. This conversion applies for images as well as for text and vector objects. During this conversion RGB gray scales are mapped 1:1 to Separation Black. This means, a 50% RGB gray (127/127/127) will be 50% Separation Black after conversion.

Tone value adjustments for conversions amongst different printing conditions

If files are converted amongst different printing conditions the different dot gain might be counterbalanced. The following parameters – seen as defined in the policy "Standard" – allow to compensate the different dot gain:

- GrayGraph-AdvancedColorConversion AdjustDotGain
- GrayImage-AdvancedColorConversion AdjustDotGain

The difference in dot gain between the gray source ICC profile and the destination ICC profile will be compensated. 100% Black remains unchanged.

Changing the destination color space of Gray

By changing the color policy file it is possible to define the destination color space of gray elements with the following parameters:

- SetGrayColorSpace
- R_eq_G_eq_B_is_Black

- a_eq_b_eq_0_is_Black
- C_eq_M_eq_Y_is_Black

NoChange	Element will be converted according to the settings for the corresponding color space
DeviceGray	Element will be converted according to the settings for the color space "Gray" and stored in DeviceGray
CMYK_Black	Element will be converted according to the settings for the color space "Gray" and stored in DeviceCMYK – Black channel only
Separation_Black	Element will be converted according to the settings for the color space "Gray" and stored in Separation Black

- Attention: In DeviceGray, a defined Overprinting flag will have no influence on subjacent CMYK elements. These will always be knocked out.

Preserve black: Exclude certain BlendModes from special handling (pdfToolbox 9.2)

Some special transparency effects are based on the interaction of color channels and therefore, a conversion of objects using a gray balance of a process color (e.g. RGB, CMYK or Lab) to pure gray scale may result in visually changed results. As these effects differ from the "construction", the used BlendMode and color space of a PDF a general solution is not possible and excluding certain BlendModes depending on the input files is necessary.

- RGBImage-R_eq_G_eq_B_is_Black_ExcludeBlendModes
- RGBGraph-R_eq_G_eq_B_is_Black_ExcludeBlendModes
- CMYKGraph-C_eq_M_eq_Y_is_Black_ExcludeBlendModes
- CMYKImage-C_eq_M_eq_Y_is_Black_ExcludeBlendModes
- LabGraph-a_eq_b_eq_0_is_Black_ExcludeBlendModes
- LabImage-a_eq_b_eq_0_is_Black_ExcludeBlendModes

All BlendModes can be excluded, which means they are converted to a process color space. Normally the gray balance is preserved by converting it to grayscale, but this can result in a changed visual appearance with some special usage of certain BlendModes (e.g. ColorDodge).

Possible values are: Normal, Compatible, Multiply, Screen, Overlay, Darken, Lighten, ColorDodge, ColorBurn, HardLight,

SoftLight, Difference, Exclusion, Hue, Saturation, Color or Luminosity.
If more than one BlendMode shall be excluded, the values have to be separated either by a space, a comma or a semicolon.

7.7 Convert colors to PSO Coated v3 (ECI)

The most common color conversion is the one with ICC profiles; given a source and destination ICC profile, the color engine converts all colors from source to destination.

pdfToolbox provides predefined Fixups for color conversion, for example:

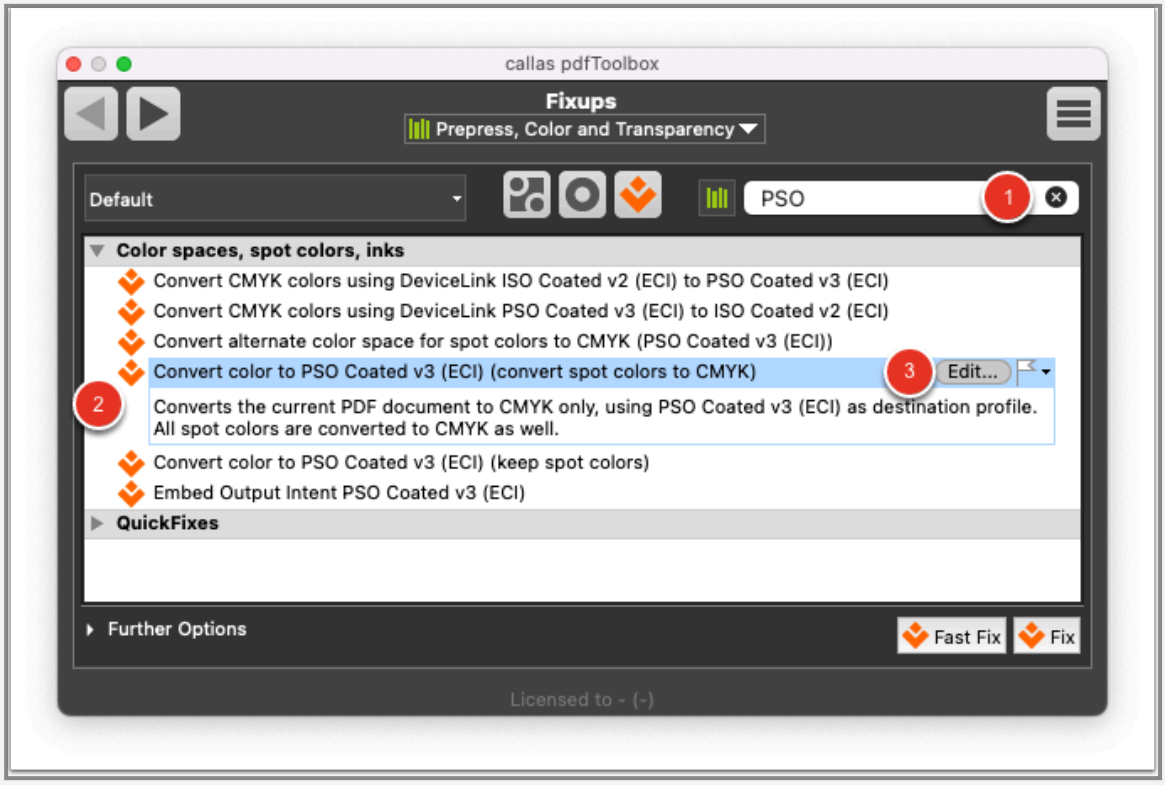
- **Convert color to PSO coated V3 (ECI)**
- **Convert color to ISO Coated v2 (ECI)**
- **Convert color to Japan Color Coated 2001**

Since "Convert Colors" is a very powerful Fixup in pdfToolbox, this article explains all relevant settings used for color conversion to PSO Coated V3 (ECI). The attached sample file contains CMYK objects, RGB images and spot colors.



Advertisement_callas.pdf

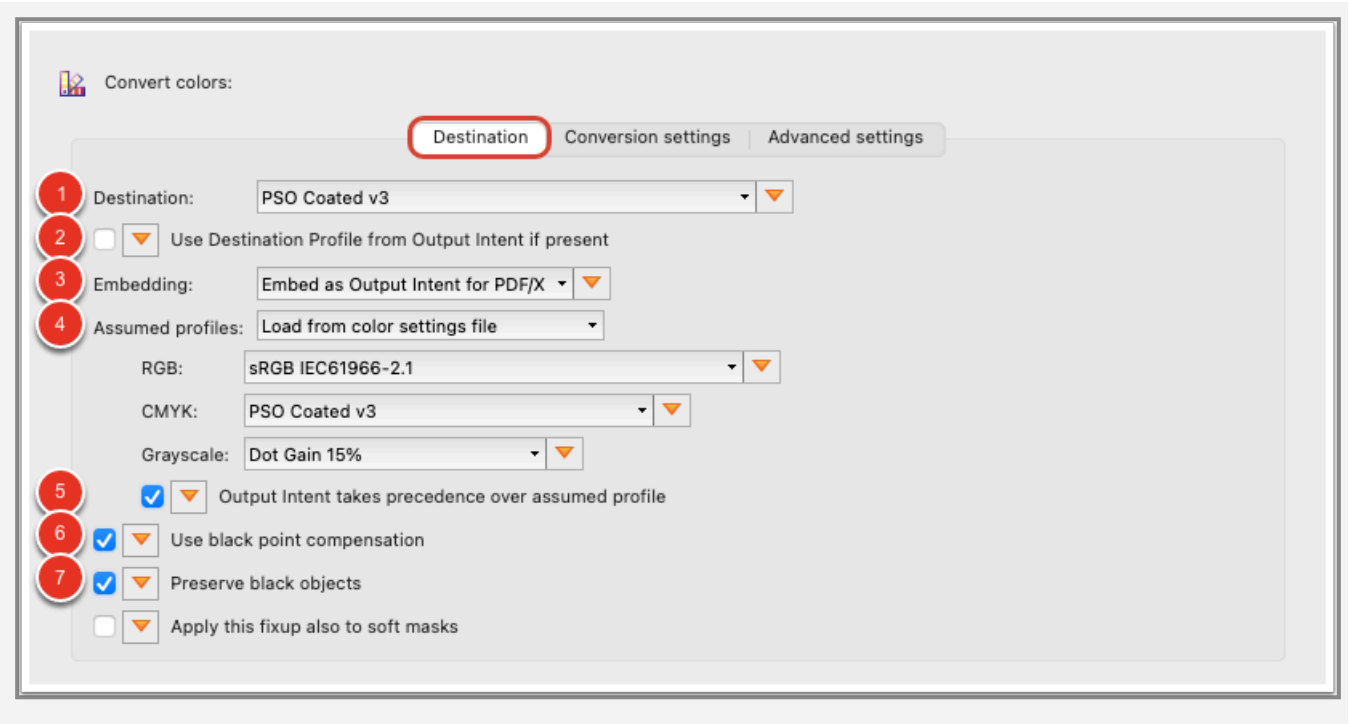
"Convert colors to PSO Coated v3 (ECI) (convert spot colors to CMYK)" Fixup



1. In the search field search for "PSO".
2. Select the Fixup "Convert color to PSO Coated v3 (ECI) (convert spot colors to CMYK)".
3. Click "Edit" to review the Fixup.

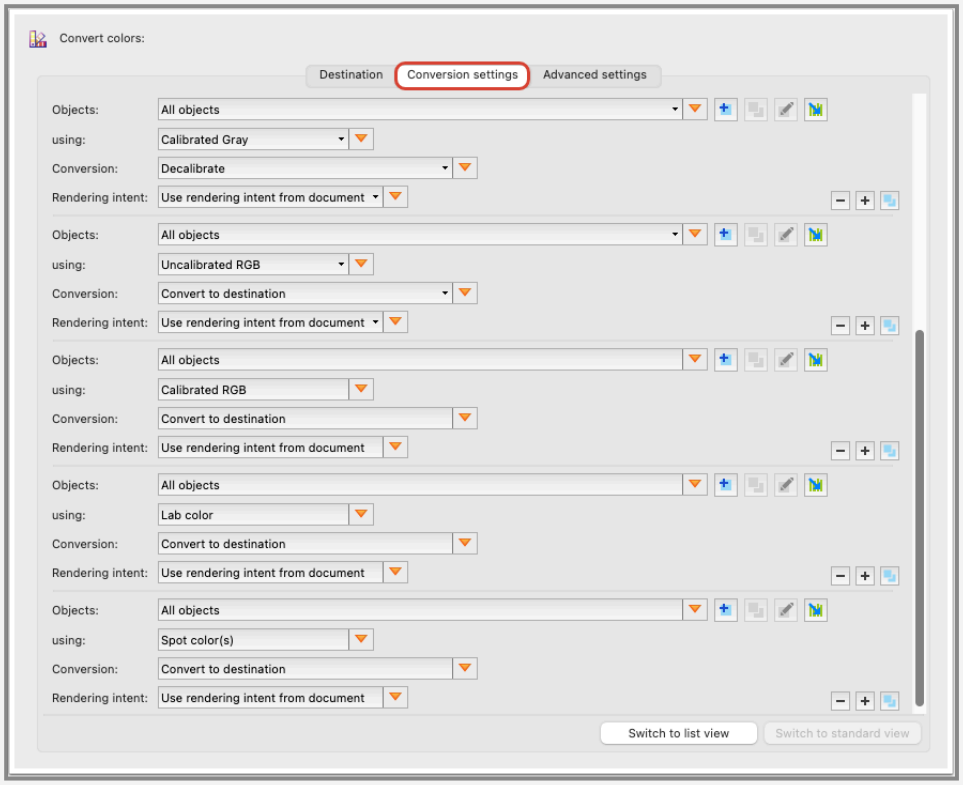
NOTE: pdfToolbox provides two Fixups for the color conversion to PSO Coated v3. The first Fixup transforms the spot colors to CMYK, the second keeps the spot colors.

Inspect the Fixup – The "Destination" tab



1. The target profile "PSO Coated v3" must be selected.
2. The check box must be disabled. If it is enabled, the Output Intent of the PDF document will be used as the target profile.
3. The target profile should be embedded in the document as Output Intent.
4. If the document has no Output Intent, you have to assume the source profile of the document. sRGB, Dot Gain 15% and PSO coated v3 are very common in the European area, so they are a good choice.
5. If an Output Intent is present in the document, it will be used as the source profile instead of the assumed profile.
6. Black point compensation should be used.
7. During color conversion pure black objects remain completely black.

The "Conversion settings" tab



This tab defines a list of different settings. The reason for this is that different types of color spaces should be handled in different ways. If you want to convert to PSO Coated V3 (ECI), the following settings are recommended:

Uncalibrated objects (no ICC Profile attached) should remain the same:

- Uncalibrated CMYK and Gray should use the conversion setting: **Do not convert**

To remove an associated ICC Profile from a calibrated CMYK or Gray objects:

- Calibrated CMYK and Gray should use the conversion setting: **Decalibrate**

RGB or Lab objects should be converted to CMYK:

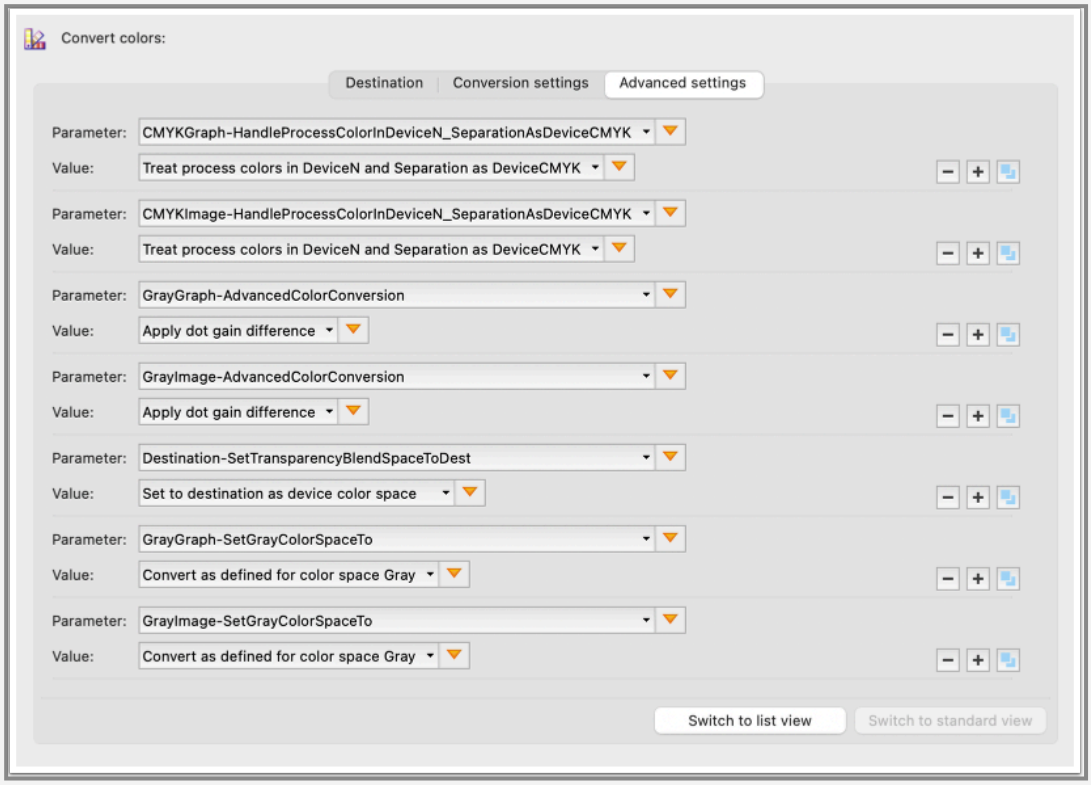
- Uncalibrated RGB, calibrated RGB and LAB should use the conversion setting: **Convert to destination**

We want to convert all spot colors to CMYK as well:

- Spot color(s) should use the Conversion setting: **Convert to destination**

The "Advanced settings" tab

Additional parameters for color conversion have been defined here. Since these are very specific, they are not explained further here. If you want to learn more about the advanced settings, read this article: Convert colors: Advanced settings.



If you want to create your own "Convert Colors" Fixup, it is recommended to duplicate an existing "Convert to …" Fixup. This will apply the predefined set of settings for the "Advanced settings" area.

Apply the Fixup and inspect the result



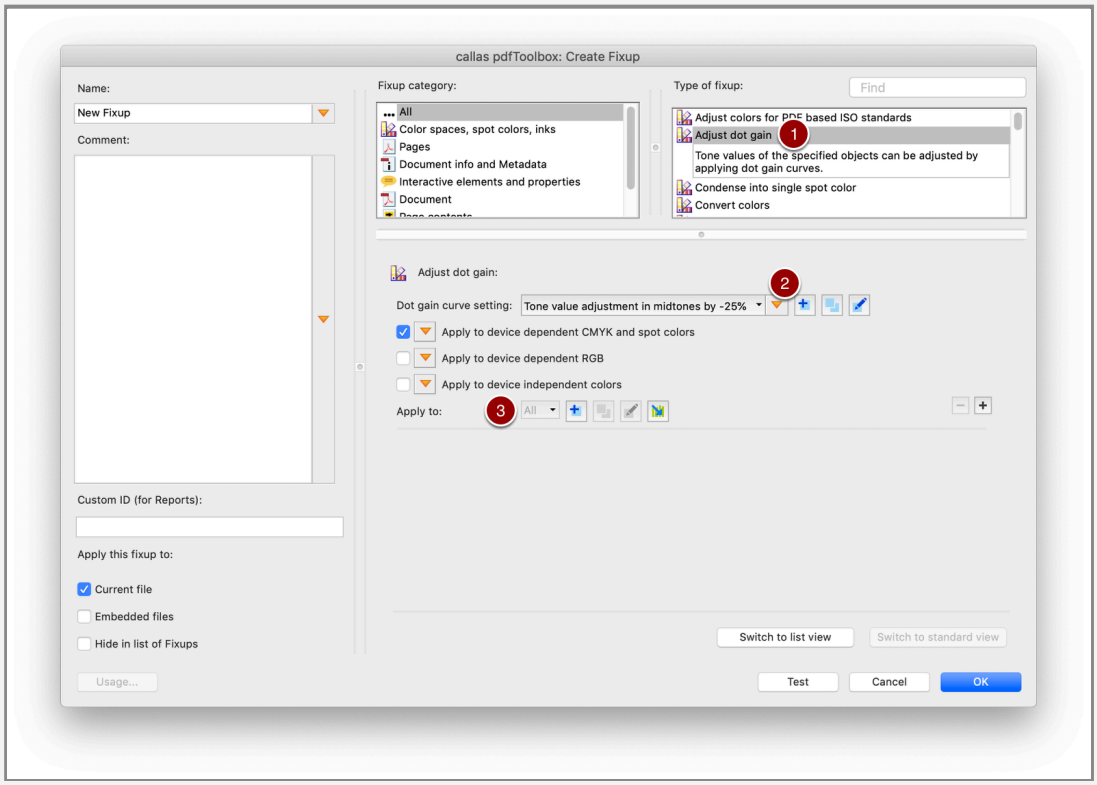
1. A green check mark is shown in the result window, refers to a successful color conversion. All spot color and RGB elements in the PDF file were converted to CMYK "PSO Coated v3".
2. The Document has now an embedded Output Intent "PSO Coated v3".

7.8 Adjust tone values and apply gradation curve to selected objects

Adjusting tone values

Besides the ICC based color conversion, it is possible to adjust the tone values in a PDF file. This can be performed with a different configuration settings, called curves or curvefiles.

Applying dot gain adjustment



1. Create a Fixup with type "Adjust dot gain".
2. A predefined tone value could be used OR the setting for the tone value can be changed to a customized setting by duplicating an existing curve file (explained later).
3. Here you have to specify a filter that determines which objects have to be 'fixed' or- adjusted for dot gain.

Apply gradation curve to select objects

The tutorial below demonstrates:

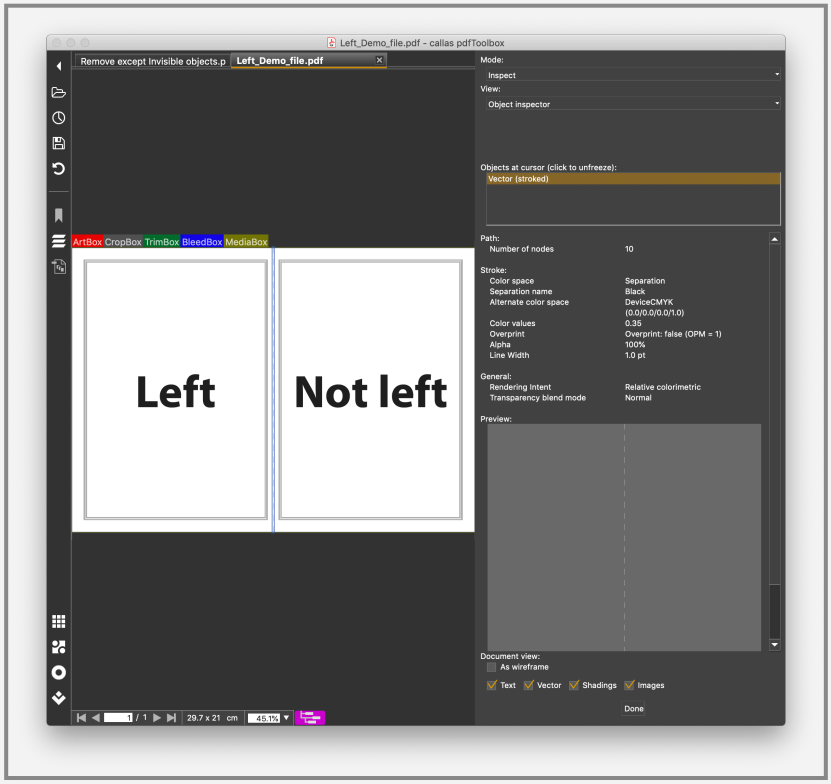
- **Setting up the appropriate filter**
- **Configuring a level for special requirements**
- **Performing levels**
- **And finally checking the result**

The purpose of this tutorial is to make a narrow black line with a tone value of 35% a little bit darker. The line that has to be correct uses a particular type of color coding, know as "Separation Black". Other lines in the PDF file are thinner or thicker and has a different tone value.



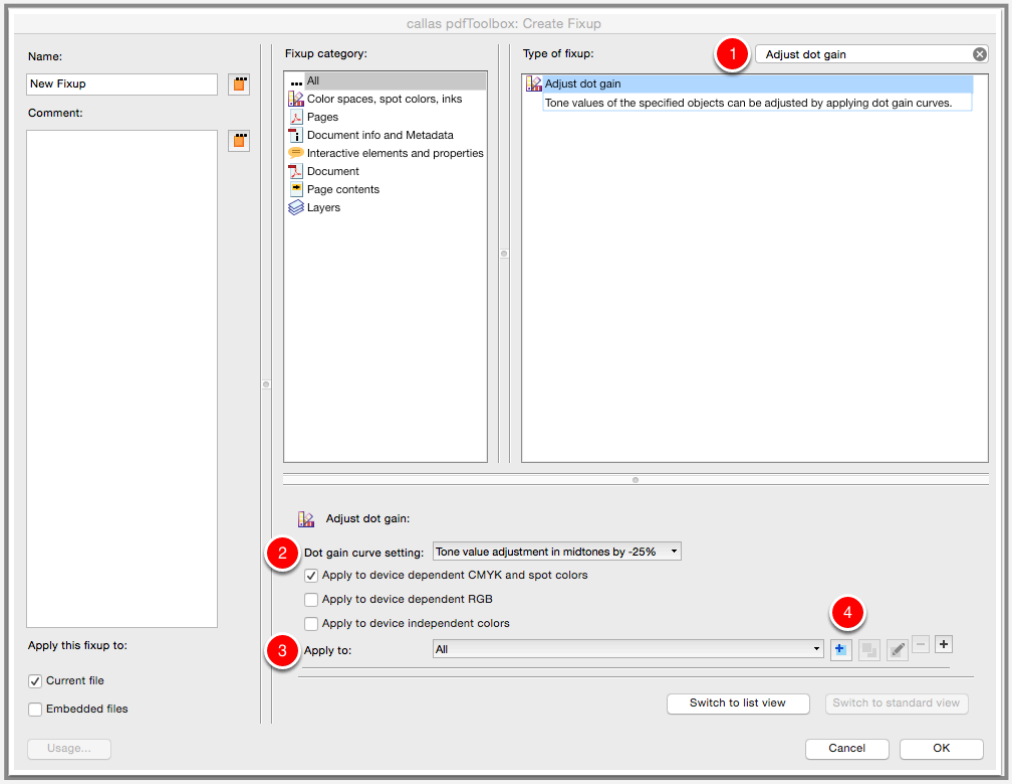
Left_Demo_file.pdf

Inspect the demo file



Upon inspecting (View>Object Inspector) the middle dashed line in the attached PDF, you will find that the dashed line appears too light in the print. The tone has to be changed from 35% to 55%.

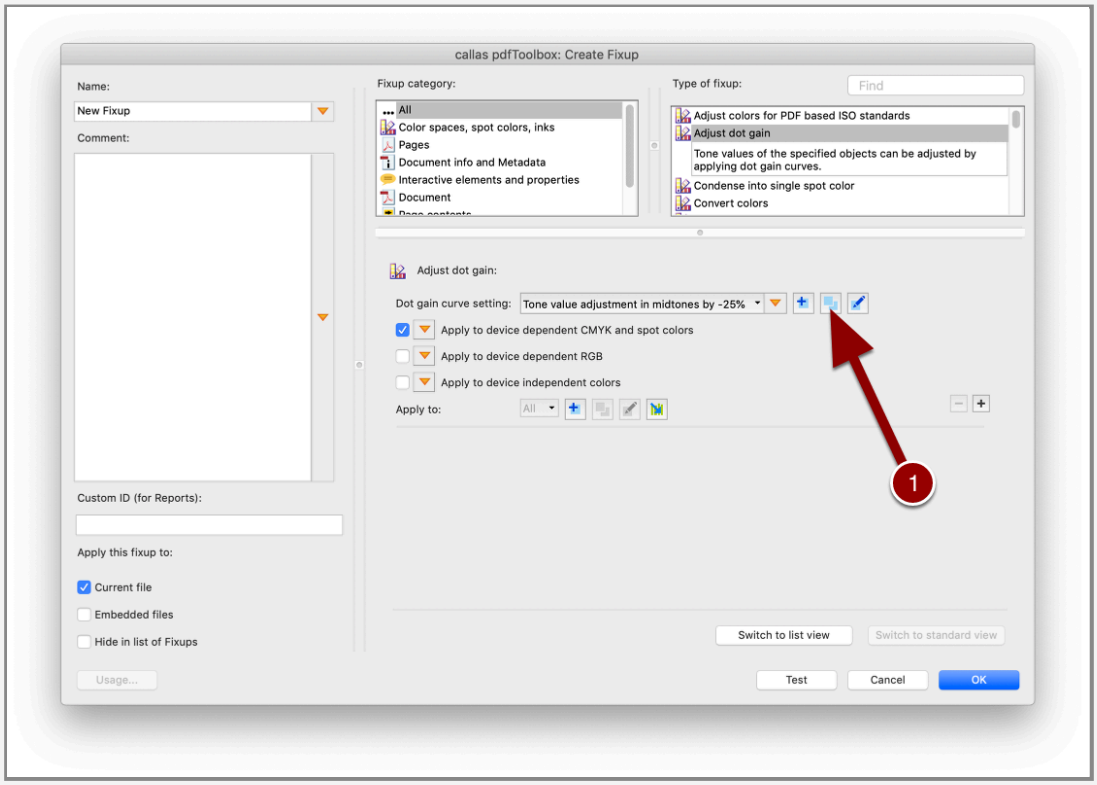
Create a Fixup



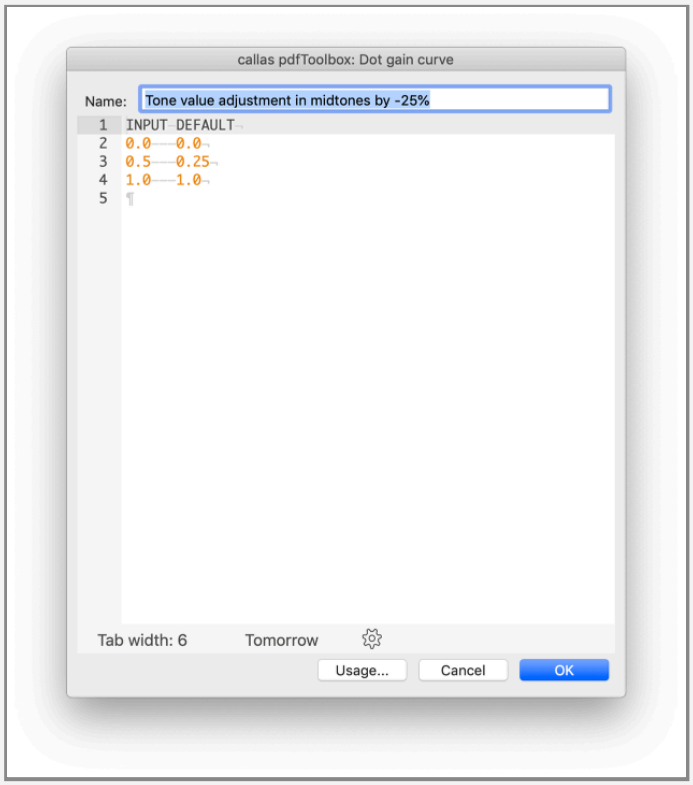
1. Create a Fixup with type "Adjust dot gain".
2. The setting for the tone value has to be changed to a customized setting (explained later).
3. Here you have to specify a filter that determines which objects have to be 'fixed' (explained next).
4. Click on the "plus" button to open a new customized Check.

Adjust the dot gain setting

The following steps are showing the current way of editing or creating a dot gain curve setting.
Before pdfToolbox 13, this had to be done via modifying an external configuration file. This procedure is described here.



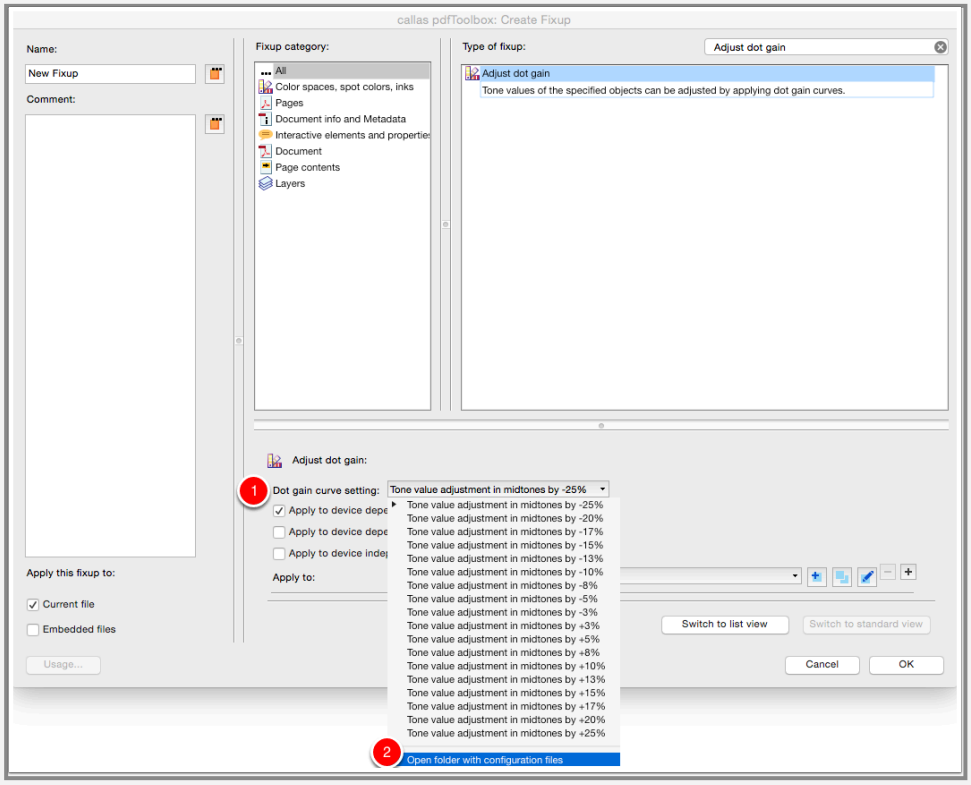
1. Best is to duplicate an existing setting. Duplication will open the window as shown below.



1. Change the name to "Change to 55 percent".
2. In this case the target value should be 55% (written as **0.55**).
3. Press OK to save this setting.

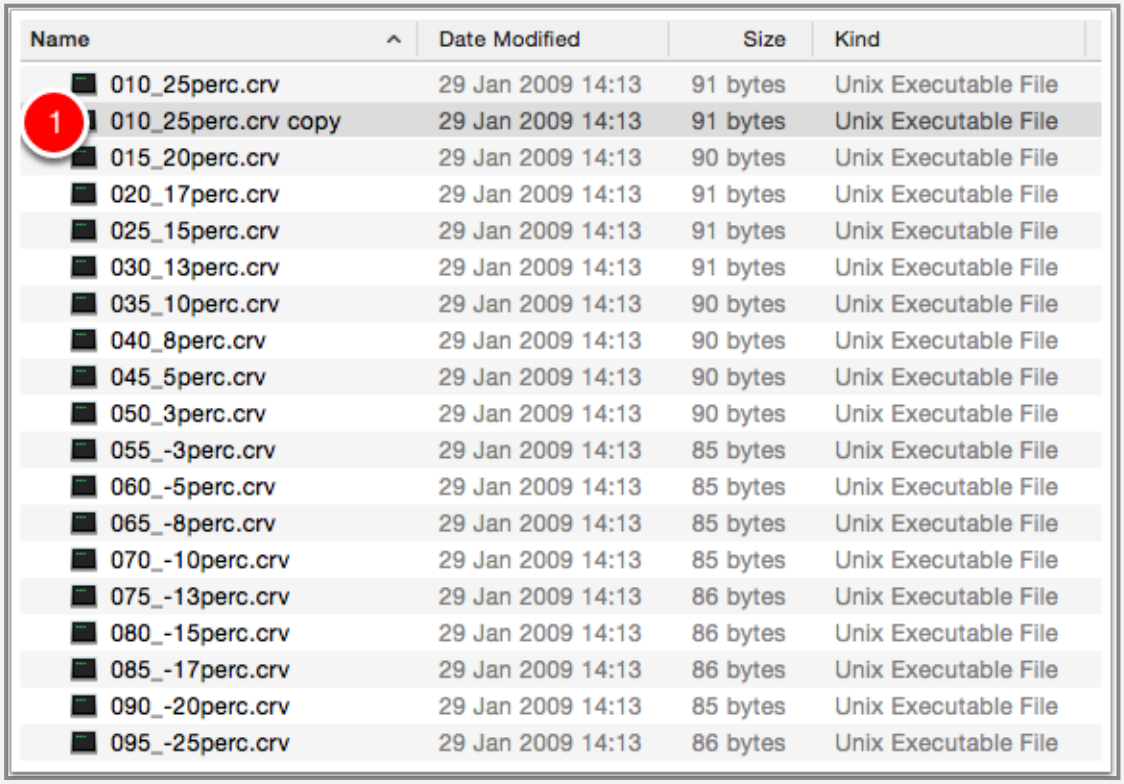


Open the dot gain curve setting folder (until pdfToolbox 13)

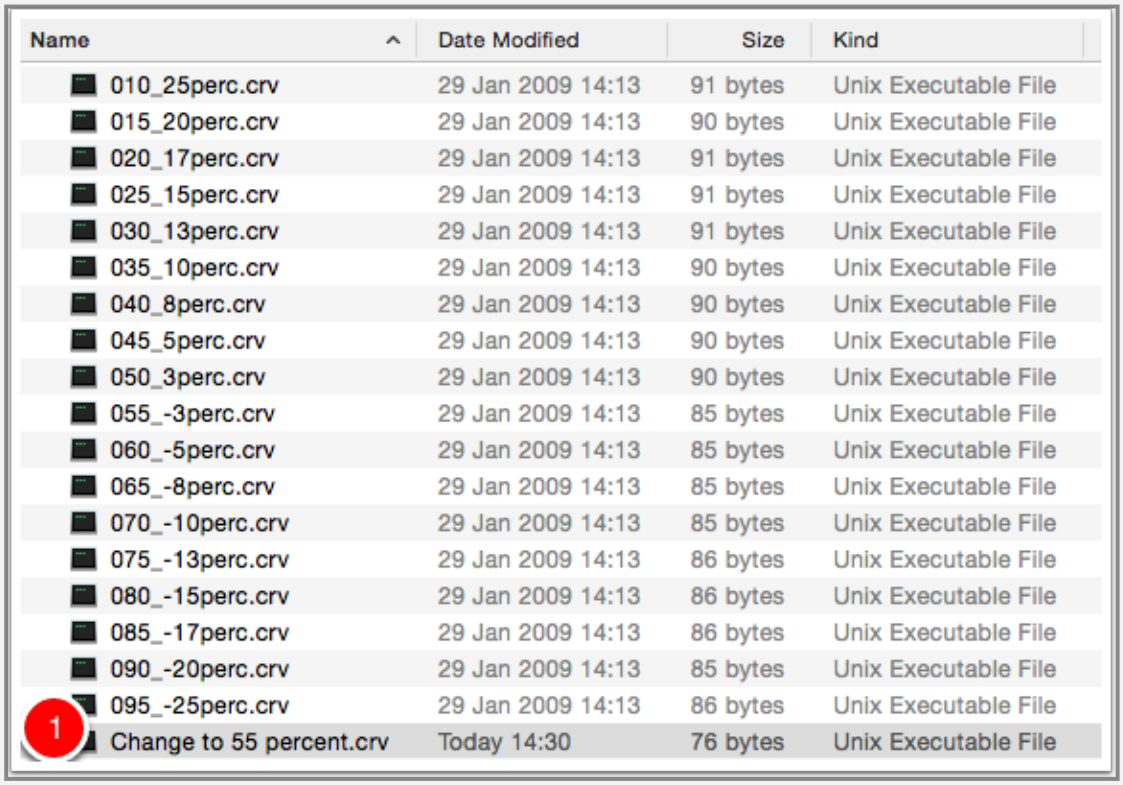


1. Open the "Dot gain curve setting list".
2. In the list select "Open folder with configuration files".

Modify existing dot gain curve setting file



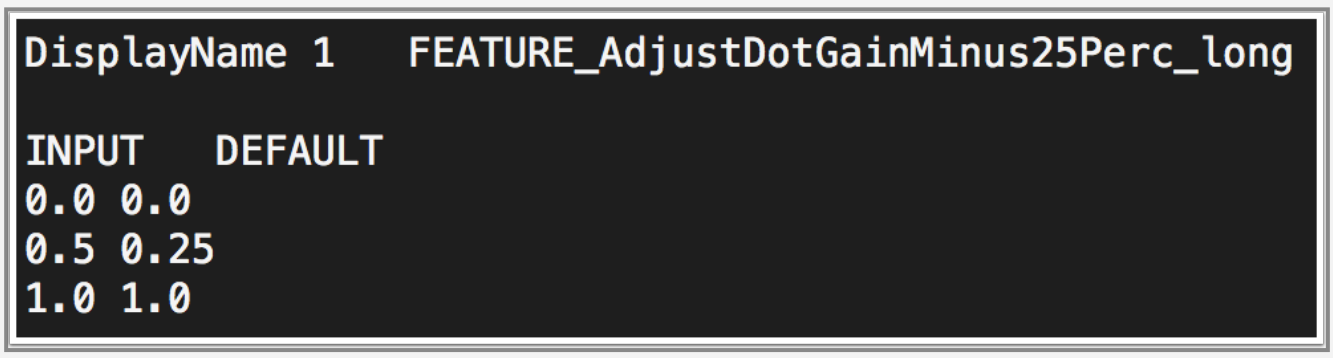
1. Select and make a copy of the dot gain curve setting file "010_25perc.crv".



Name	Date Modified	Size	Kind
010_25perc.crv	29 Jan 2009 14:13	91 bytes	Unix Executable File
015_20perc.crv	29 Jan 2009 14:13	90 bytes	Unix Executable File
020_17perc.crv	29 Jan 2009 14:13	91 bytes	Unix Executable File
025_15perc.crv	29 Jan 2009 14:13	91 bytes	Unix Executable File
030_13perc.crv	29 Jan 2009 14:13	91 bytes	Unix Executable File
035_10perc.crv	29 Jan 2009 14:13	90 bytes	Unix Executable File
040_8perc.crv	29 Jan 2009 14:13	90 bytes	Unix Executable File
045_5perc.crv	29 Jan 2009 14:13	90 bytes	Unix Executable File
050_3perc.crv	29 Jan 2009 14:13	90 bytes	Unix Executable File
055_-3perc.crv	29 Jan 2009 14:13	85 bytes	Unix Executable File
060_-5perc.crv	29 Jan 2009 14:13	85 bytes	Unix Executable File
065_-8perc.crv	29 Jan 2009 14:13	85 bytes	Unix Executable File
070_-10perc.crv	29 Jan 2009 14:13	85 bytes	Unix Executable File
075_-13perc.crv	29 Jan 2009 14:13	86 bytes	Unix Executable File
080_-15perc.crv	29 Jan 2009 14:13	86 bytes	Unix Executable File
085_-17perc.crv	29 Jan 2009 14:13	86 bytes	Unix Executable File
090_-20perc.crv	29 Jan 2009 14:13	85 bytes	Unix Executable File
095_-25perc.crv	29 Jan 2009 14:13	86 bytes	Unix Executable File
1 Change to 55 percent.crv	Today 14:30	76 bytes	Unix Executable File

1. Change the name to "Change to 55 percent".

Review existing dot gain curve setting data

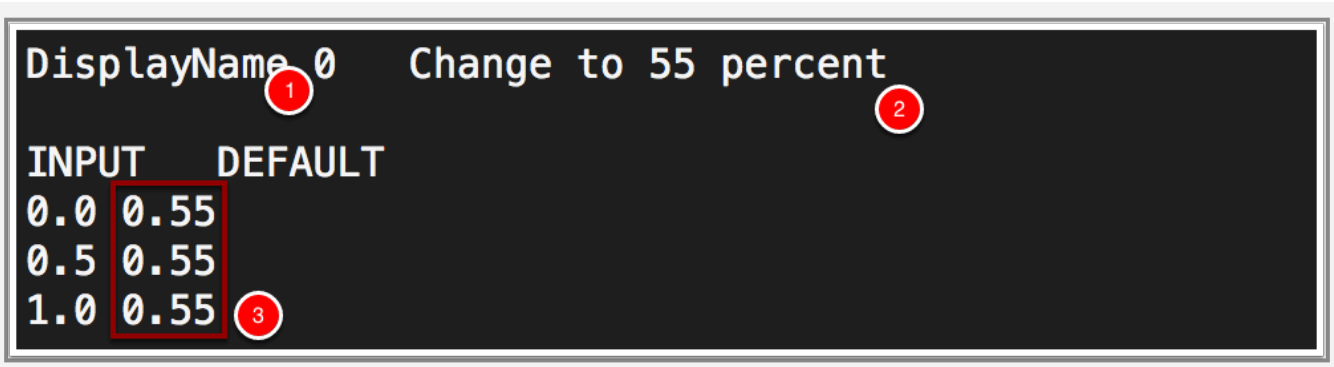


```
DisplayName 1   FEATURE_AdjustDotGainMinus25Perc_long

INPUT   DEFAULT
0.0 0.0
0.5 0.25
1.0 1.0
```

Open the dot gain curve setting file "010_25perc.crv" in a text editor to review the data. Most text editors are free. For example on Mac you have "TextEdit" and "TextWrangler"; on Windows "Notepad" and "Notepad ++".

Modify the new 'Change to 55 percent.crv'



Open the dot gain curve setting file "Change to 55 percent.crv" in a text editor to modify.

NOTE: When you edit or change the curve setting file, you have to make absolute sure to maintain the existing gaps between the values as **tabs**, and not to replace them with single spaces.

1. If you want to specify a different name, this value must be set from 1 to **0**.
2. The name must be set to an appropriate value that is used. Here "FEATURE_AdjustDotGainMinus25Perc_long" should be "**Change to 55 percent**".
3. The desired target values for levels must be registered in the rows affected by tabs. In this case the target value should be 55% (written as **0.55**).

Please make sure the text file uses the suffix ".crv".

Syntax of a dot gain curve

The curve file needs to have a **tab delimited** structured as following:

```
DisplayName 1 Example
```

When using the build-in editor in pdfToolbox 13 or later, syntax issues will be highlighted automatically.

INPUT	DEFAULT	Cyan	Magenta	Yellow	Black	REGEX::.*Spot.*
0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.1	0.1	0.1	0.12	0.1	0.1	0.11
0.2	0.3	0.3	0.3	0.3	0.3	0.3
0.3	0.4	0.4	0.4	0.44	0.4	0.42
0.4	0.5	0.5	0.52	0.5	0.45	0.5
0.5	0.6	0.6	0.6	0.6	0.6	0.6
0.6	0.7	0.7	0.7	0.7	0.66	0.68
0.7	0.79	0.8	0.8	0.8	0.76	0.78
0.8	0.85	0.85	0.86	0.85	0.8	0.82
0.9	0.9	0.9	0.9	0.92	0.9	0.91
1.0	1.0	1.0	1.0	1.0	1.0	1.0

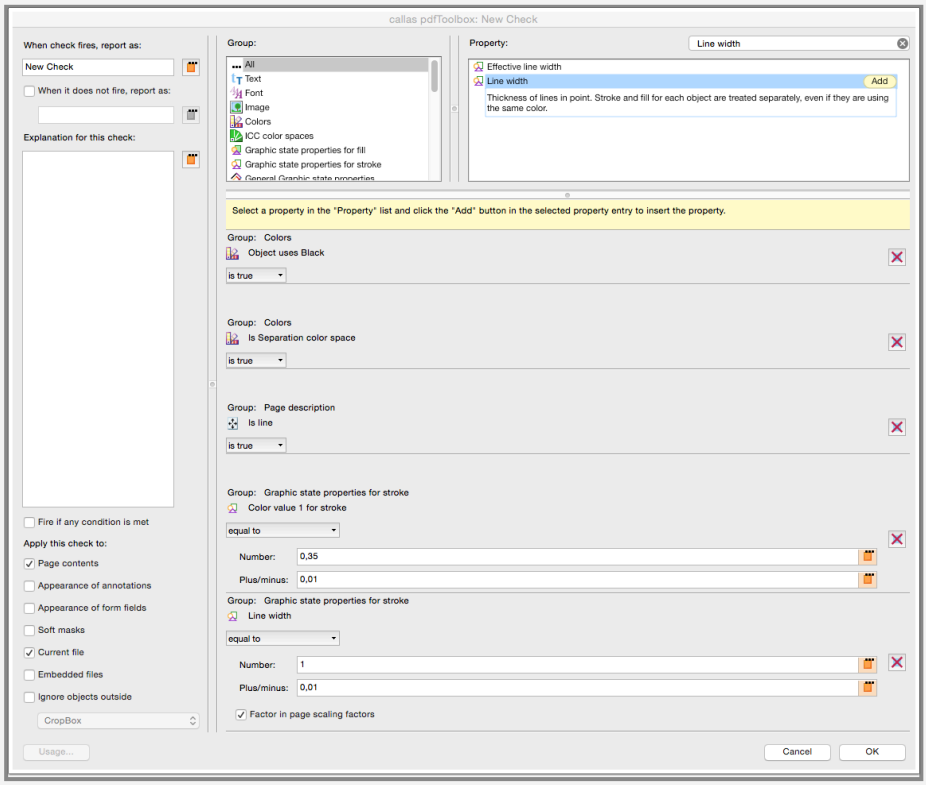
Not all values have to be listed in this file. For intermediate values not present a spline based interpolation is used.

Besides adjusting the tone values for each CMYK separation, you could also use the column title "`All`" to adjust all separations to the same value or "RGB::Red", "RGB::Green" and "RGB::Blue" for RGB as well as "LAB::L" for Lab color spaces. Every other column title will be considered as a spot color name. In order to be able to address a group of spot color channels regular expressions are supported as follows: "REGEX::_some_regular_expression_".

Add and configure the Checks for the 'Apply to' filter

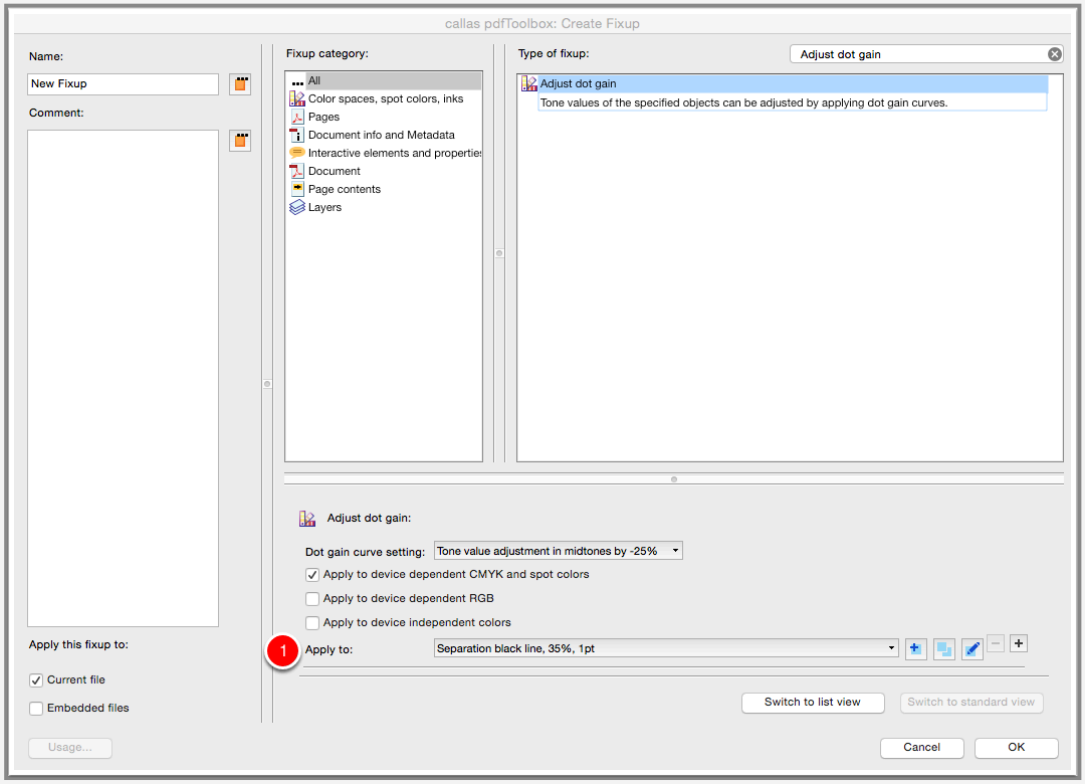
- Check property: "Object uses Black"
- Check property: "Is Separation color space"
- Check property: "Is line"
- Check property: "Color value 1 for stroke"
  - Number: 0,35
    Plus/minus: 0,01
- Check property: "Line width"

- Number: 1
  Plus/minus: 0,01



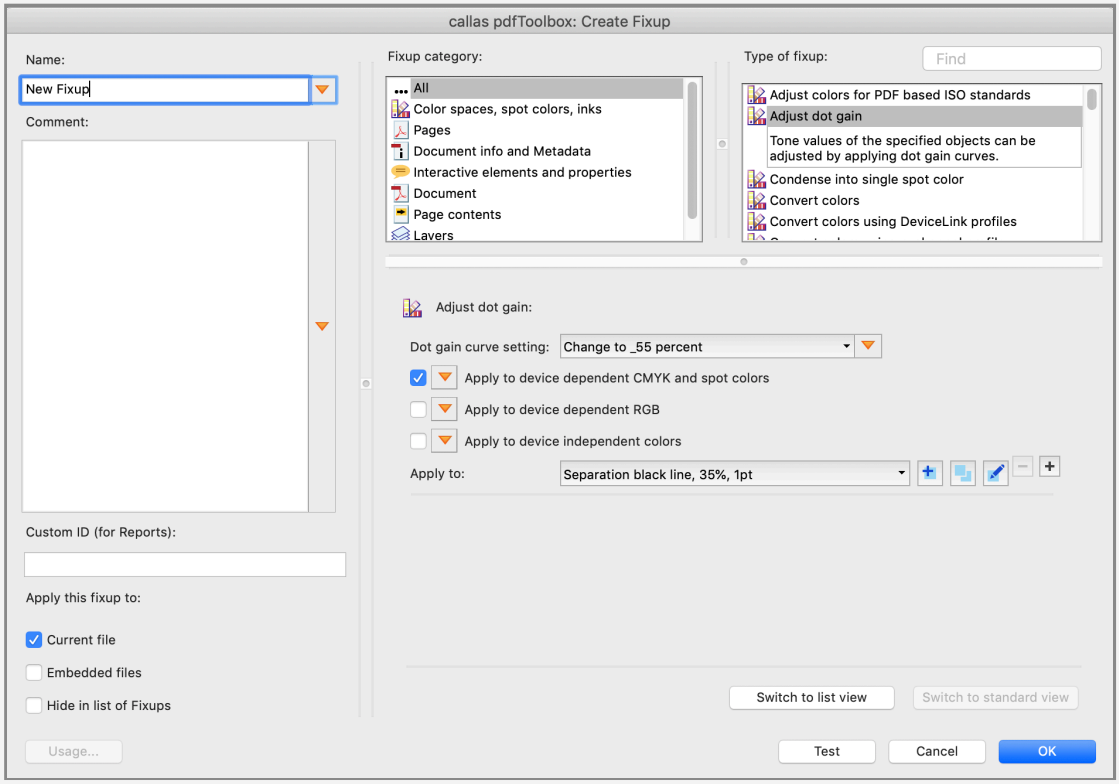
Finally, name the Check 'Separation black line, 35%, 1pt'

Apply a customized Check in the 'Adjust dot gain' Fixup



1. In the 'Apply to' filter, select "Separation black line, 35%, 1pt".

Apply dot gain curve setting

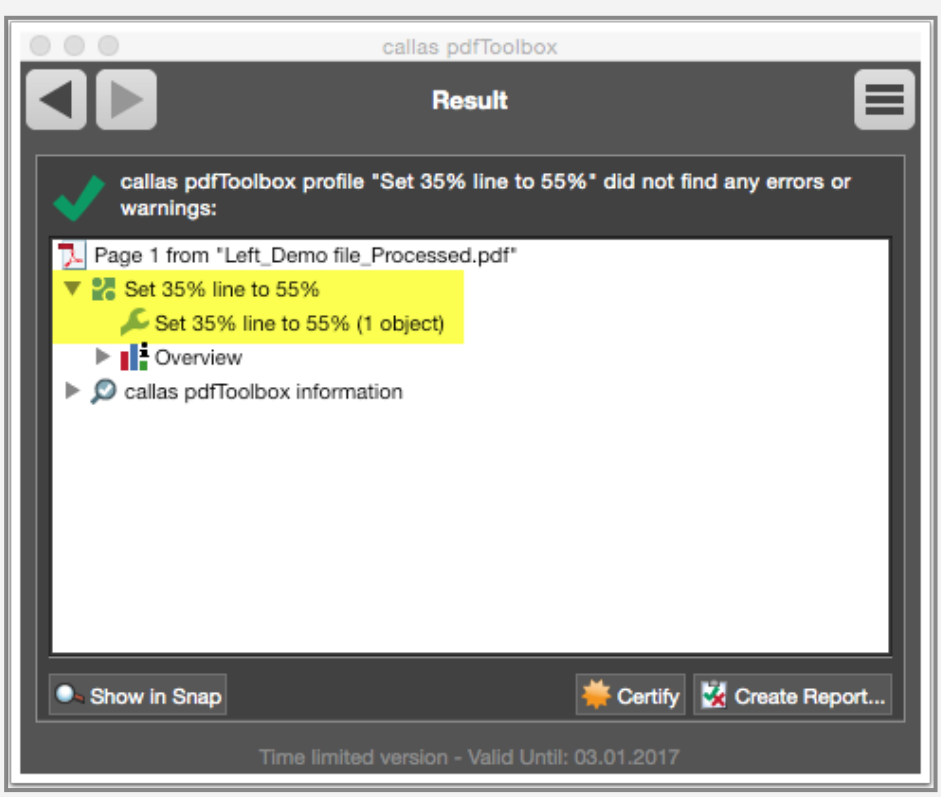


NOTE: After saving the new dot gain curve setting file, the setting is not in the tone configuration list. You have to close the dialog and open again to see the new setting in the list.

1. In the list click "Change to 55 percent".

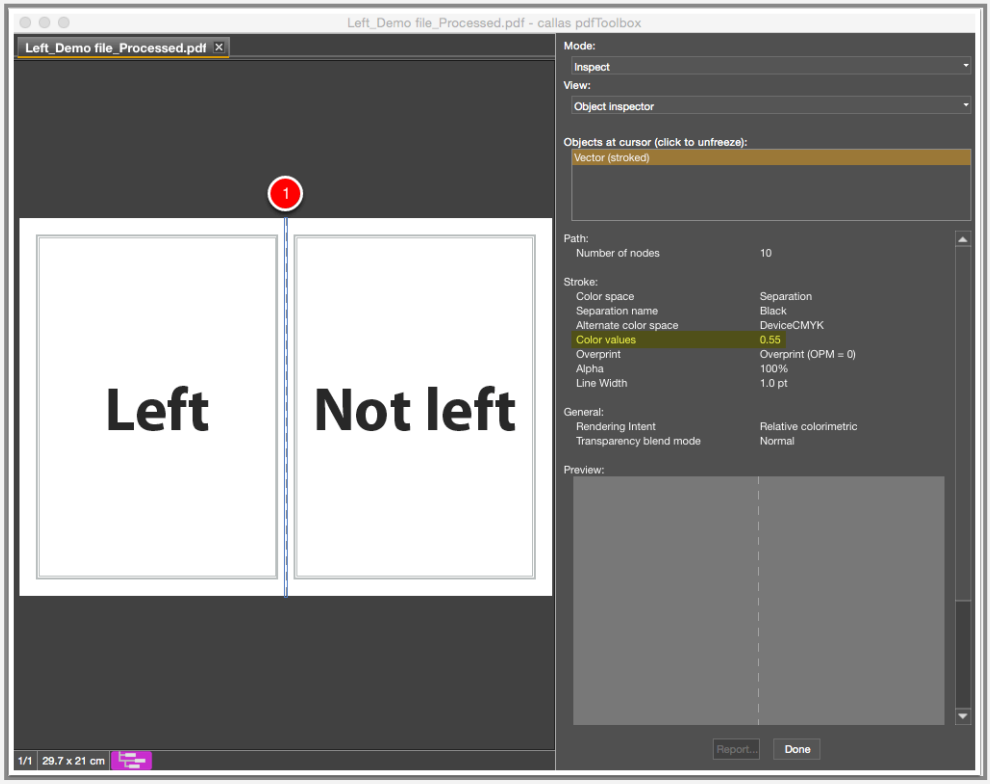
Rename the Fixup: "Set 35% line to 55%"

Apply Fixup on PDF file, save the Resultant PDF and review the report



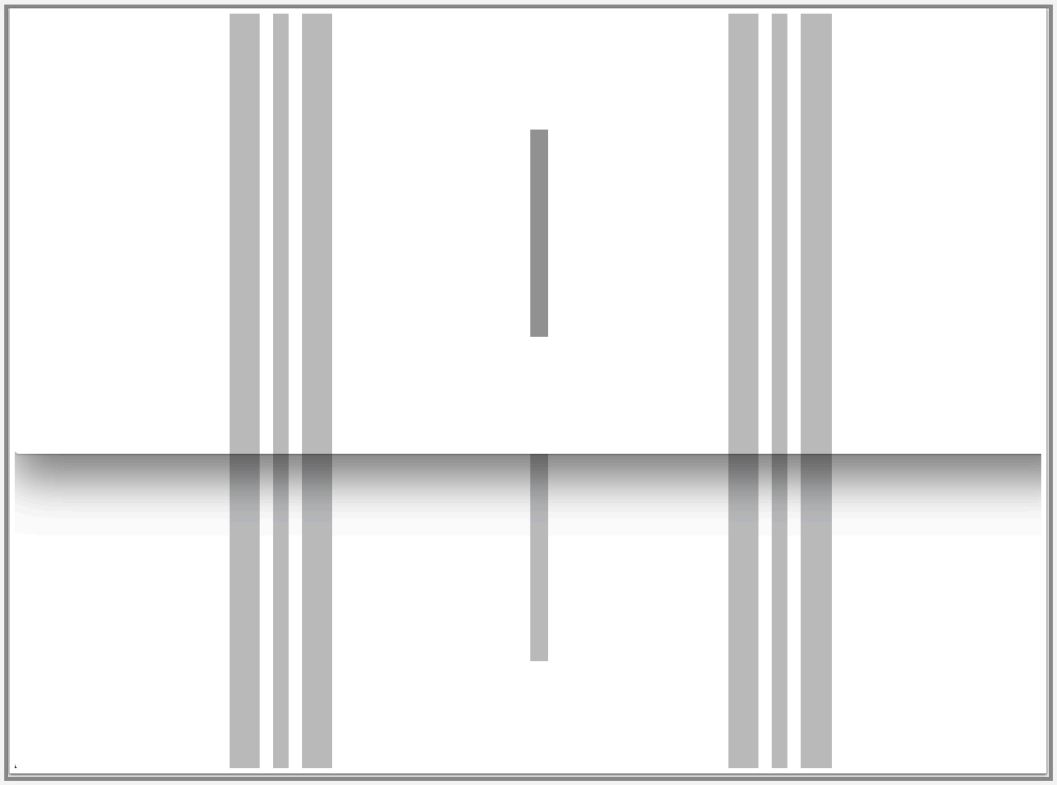
The 35% line tone is changed to 55%.

Inspect the processed PDF file "Left_Demo file_Processed"



1. Select the dashed line.

The tone of the dashed line is changed from 35% to 55%.

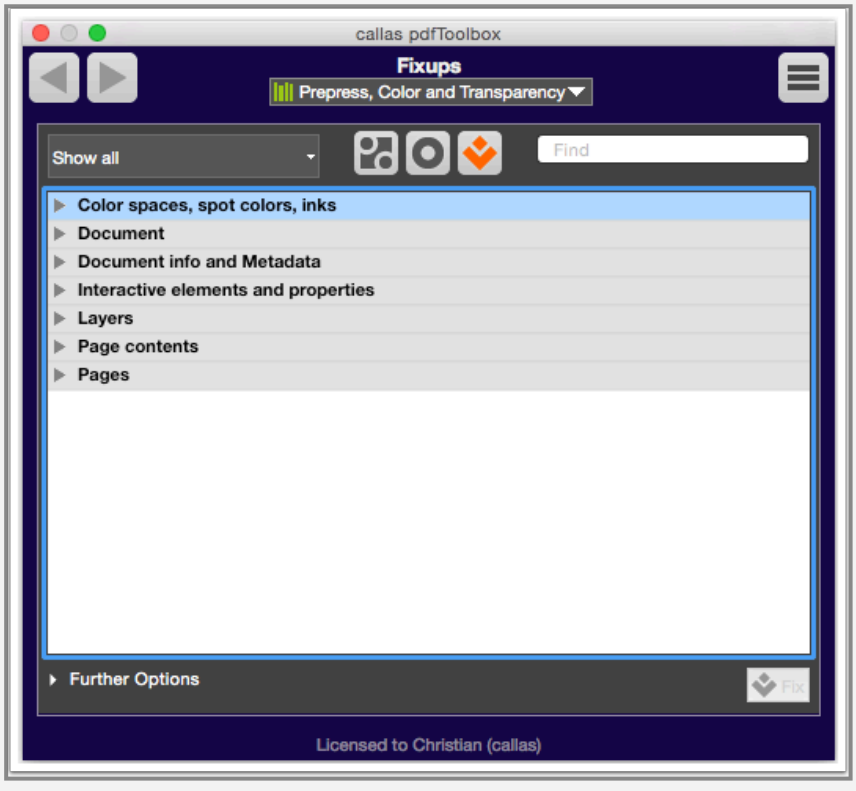


At the top is the processed PDF file where the tone of the dashed line is changed to 55%. At the bottom, the original dashed line with the lighter tone of 35%. In both PDF files the remaining lines are unchanged.

7.9 How to convert CMYK in DeviceN to DeviceCMYK

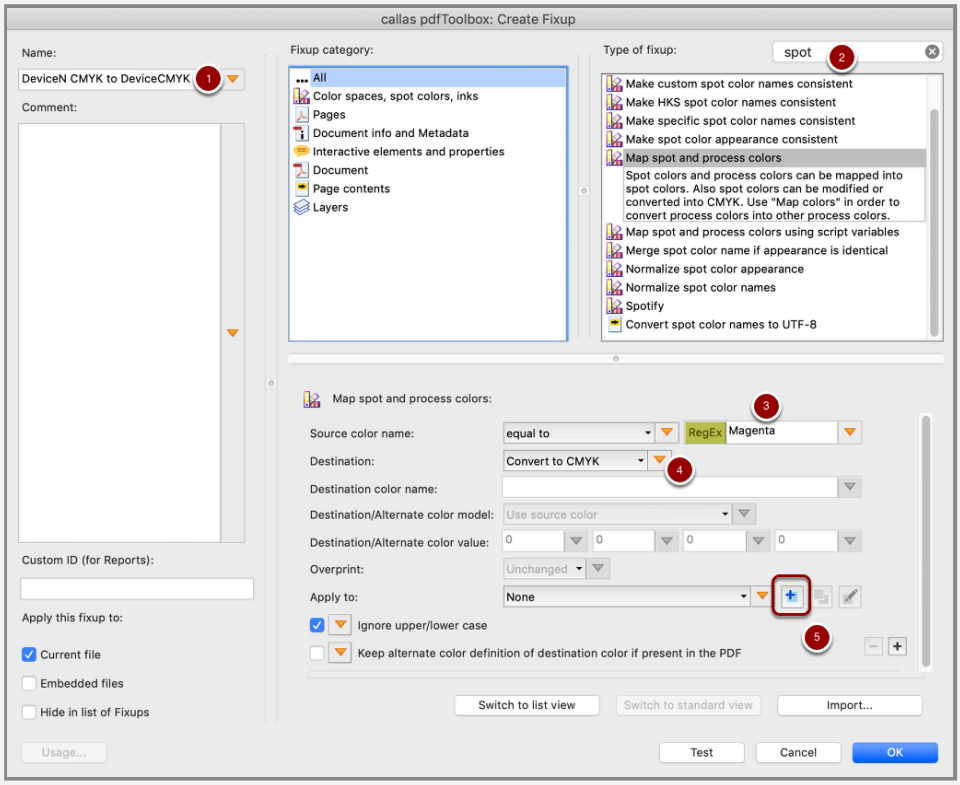
Sometimes it is necessary to convert a CMYK color space stored in DeviceN to DeviceCMYK. Some output devices treat DeviceN color spaces as special colors - even when they simply use CMYK in practice. In order to avoid an unwanted color conversion, it is therefore advisable to convert the CMYK channels stored in DeviceN to a simple DeviceCMYK format.

Create a new callas pdfToolbox Fixup



Create a new Fixup in the “Fixups” category inside the profile window. To do so, select “Create Fixup...” in the Options menu.

Configure Fixup



1) First, give the Fixup its own name, e.g. “DeviceN CMYK to DeviceCMYK”.

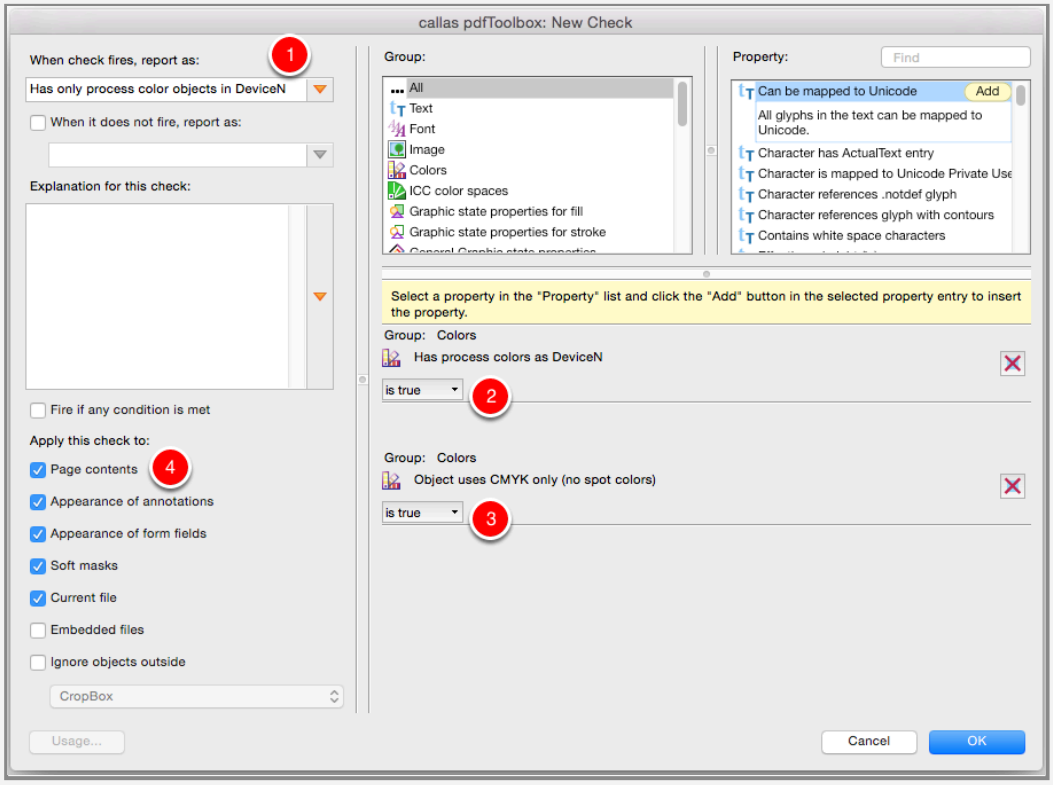
2) Next, find the Fixup type “Map spot and process colors”.

3) Type in “Magenta”.

4) Select “Convert to CMYK”.

5) To limit conversion to objects that use CMYK, you will need to create a new Check. Click on the highlighted icon to do so.

Create a check which restricts the Fixup to DeviceN and pure CMYK



1) Give the Check a suitable name, such as “Has only process color objects in DeviceN”.

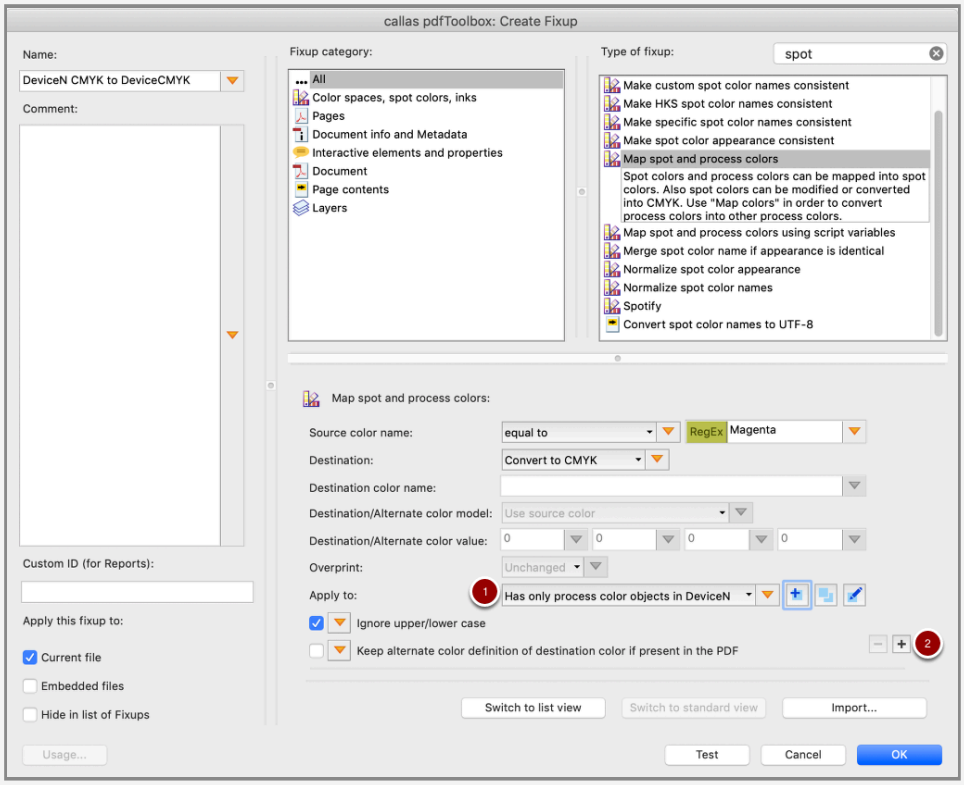
2) Search for “Has process colors as DeviceN” in the list of available properties in the upper right, then click “Add” and select “is true”.

3) Search again in the list of available properties, this time for “Object uses CMYK only (no spot colors)”, click “Add” and select “is true”.

4) Make sure that the same object types are enabled.

Click “OK” to save these settings.

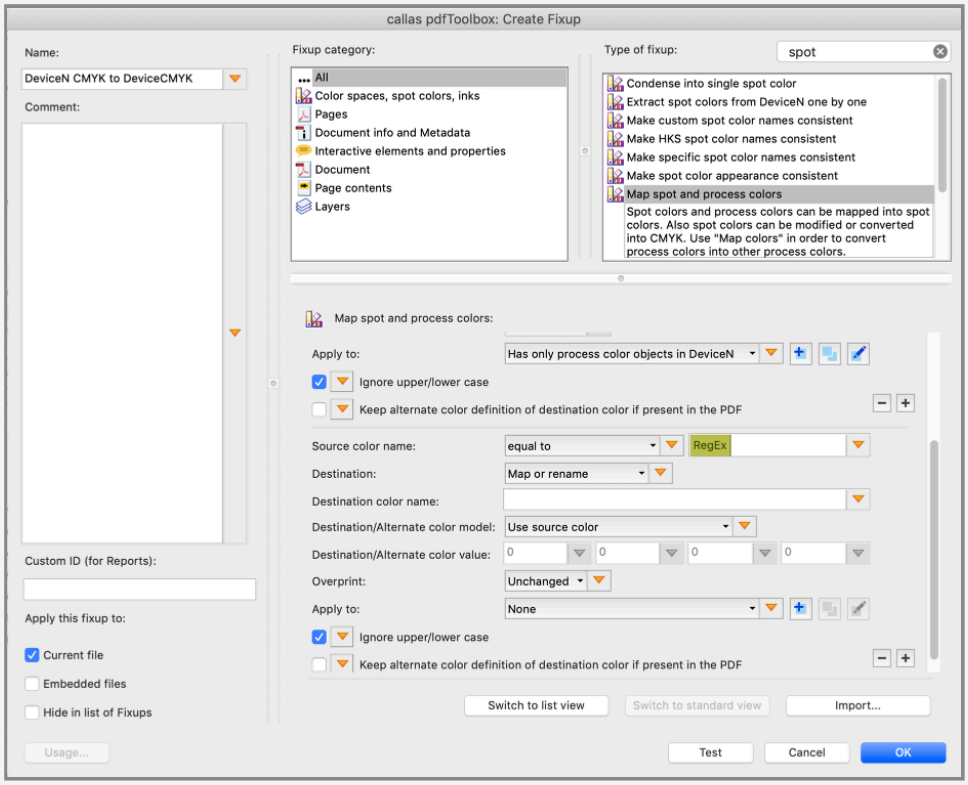
Continue creating the Fixup



1) The newly created Fixup should now be selected.

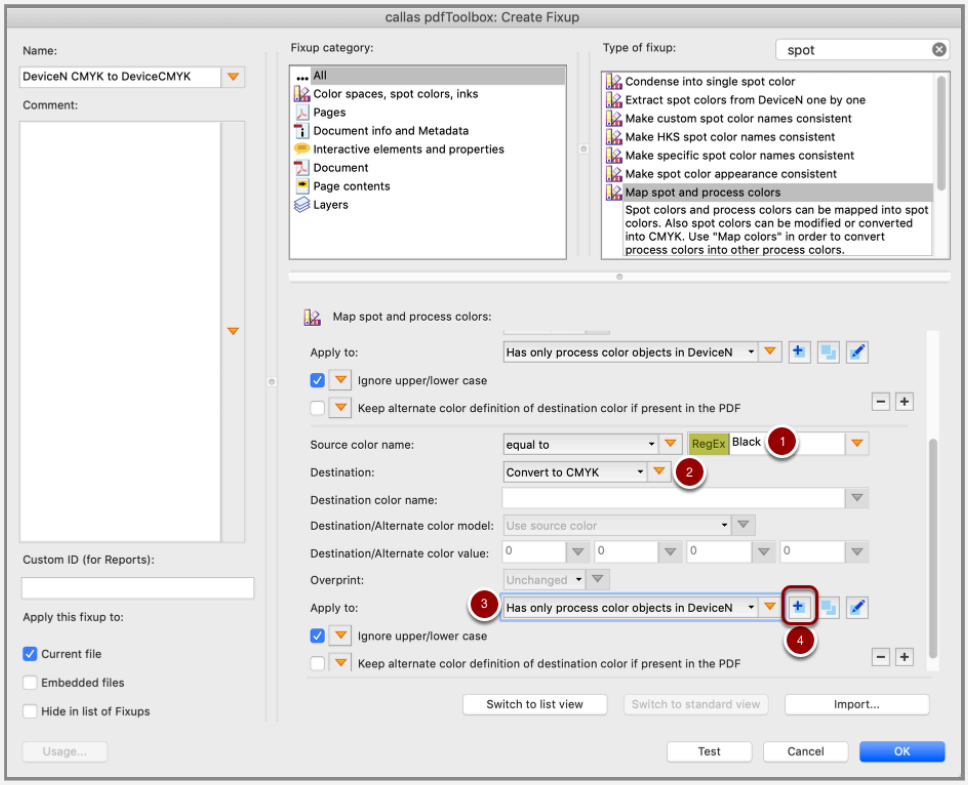
2) To add further conversion types to also cover cyan, yellow and black, click on the “+” symbol.

Add additional colors to the conversion process



A new area for additional conversion settings will be shown.

Add additional conversion types



Configuration settings for the next 4 CMYK colors:

1) Enter the color name, e.g. “Black”.

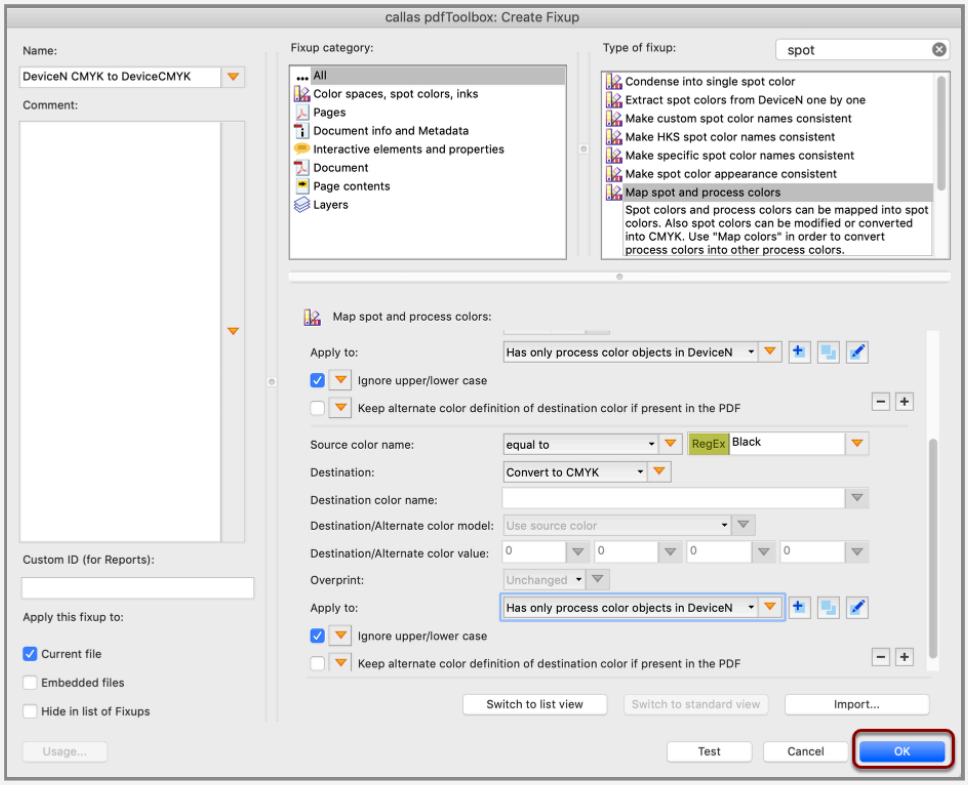
2) Select “Convert to CMYK”.

3) Select the newly created Fixup.

4) Repeat with similar settings for cyan and yellow.

(Please note that these settings can be entered in any order.)

Configure Fixup



After providing the settings for all 4 color values, save the new Fixup by clicking “OK”.

The new Fixup can now be used in any of the following ways: as an independent Fixup, within a Profile, or as part of a Process Plan.

7.10 DeviceLink conversion

Simple_TAC_tests.pdf

Simple_TAC_tests_DeviceLink300.pdf

The purpose of DeviceLink conversion

DeviceLink Profiles complement the use of standard ICC Profiles, countering the weaknesses (for certain purposes) of color conversion based on ICC Profiles. These weaknesses primarily involve CMYK-CMYK transformation and generally lead to loss of data relating to black generation and the number (and configuration/ purity) of color channels used.

A CMYK-CMYK transformation using ICC Profiles, however, always uses the device-independent lab color space, which leads to complete re-separation of the data and sometimes to unexpected and undesirable results.

DeviceLink Profiles prevent this outcome: they allow direct control of the color composition without the extra work of using an intermediate color space.

There are also DeviceLink Profiles with differing source and target color spaces.

The callas DeviceLink Library includes a set of the most commonly used DeviceLink Profiles.
However, you can also install your own DeviceLink Profiles. More on this issue can be found in this chapter.

Using your own DeviceLink Profiles

You can very easily use your own Profiles for DeviceLink conversion with the pdfToolbox. Simply click on the “Import”

button in the DeviceLink Action or in the “Convert colors with DeviceLink Profiles” Fixup, then select a Profile from your system.

An XML file with descriptive text and a PNG file with the icon (all three with identical names, differing only by the file extension) will be automatically generated. These can be customized according to your exact needs.

Please note that Profiles must end with the “icc” file extension in order to be imported.

Example XML file

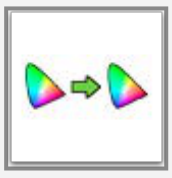
```
<?xml version=”1.0” encoding=”UTF-8” ?>
<devicelink>
<profile lang=”ENU”>
<header>DeviceLink_ISOcoatedV2_280_GCR70</header>
<explanation>This is the descriptive text for the DeviceLink profile.</explanation>
</profile>
</devicelink>
```

If you make any changes to the XML file, make sure to save it in UTF-8 format.

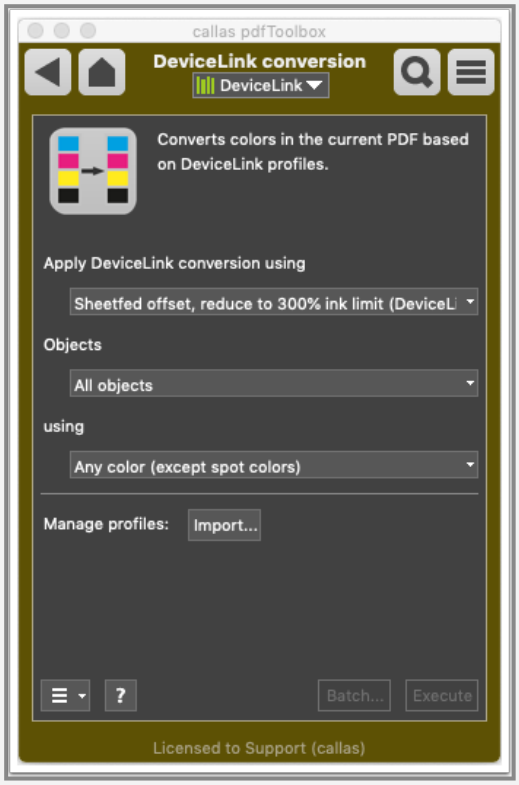
You can also create different sets of descriptive text for different languages: simply copy the area designated “ENU” and mark it with e.g. “DEU”. If only the “ENU” section exists, it will be used for all languages.

Example PNG file

The PNG file should be 64x64 pixels in size. It is only used in the Switchboard and only needs to be changed if you wish to use your own icon.



DeviceLink in the Switchboard



Import and use paths

You can easily select and import DeviceLink Profiles using the “Import” button.

The Profile will then be stored in your user preferences (in the current library from Version 9 onwards).

The files to be used for your own DeviceLink Profiles will be stored under the following paths:

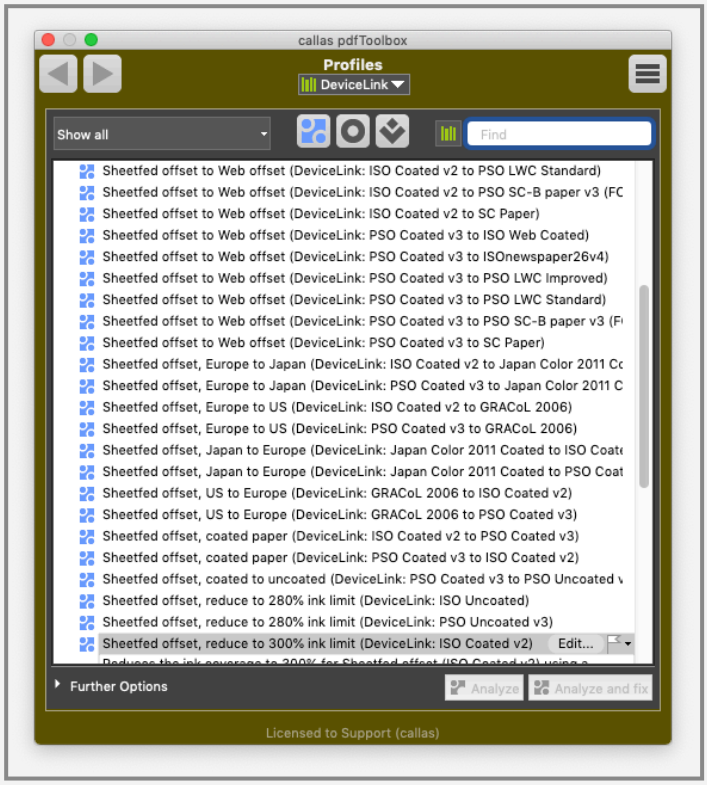
Mac:

/Users/<USERNAME>/Library/Preferences/callas software/callas pdfToolbox <VERSION>/Repositories/<CURRENT LIBRARY>/DeviceLink

Windows:

%AppData%\callas software\callas pdfToolbox <VERSION>\Repositories\<CURRENT LIBRARY>\DeviceLink

DeviceLink as Fixup

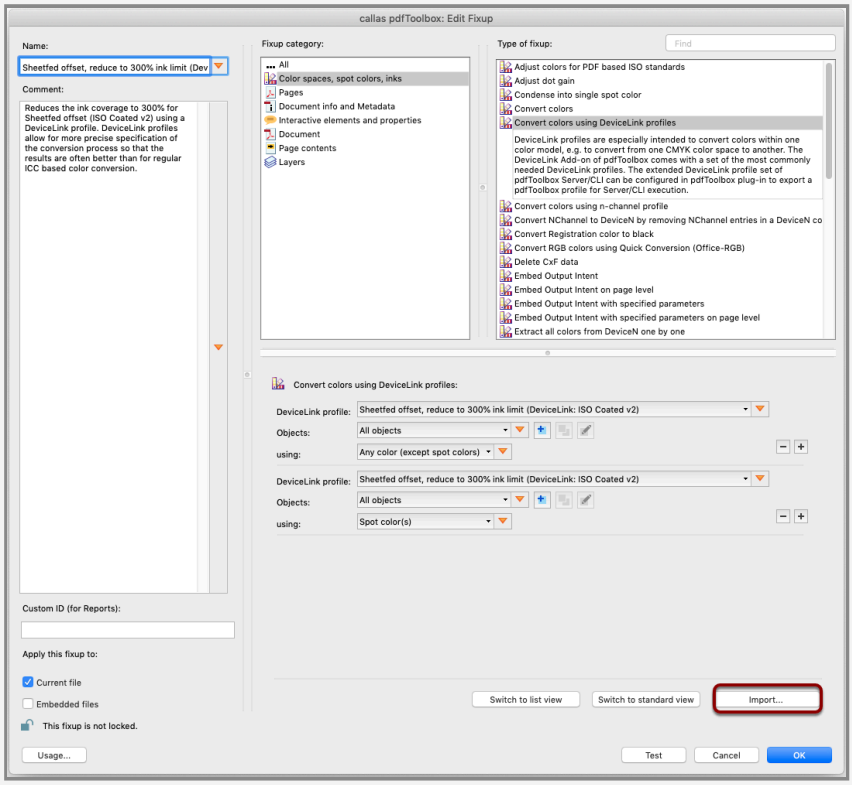


The pdfToolbox is supplied with a range of DeviceLink conversions as standard.
From Version 9 onwards, these can be found in the “DeviceLink” Library.
Starting with version 10.1, the previously required add-on license is no longer necessary and a large set of new DeviceLink profiles is included (old set of DeviceLink Profiles are attached just below).

You can also use your own DeviceLink profiles with pdfToolbox, of course.

DeviceLink_(old_set).kfpl

Editing a DeviceLink Fixup



To use your own DeviceLink Profiles, simply create a new Fixup.

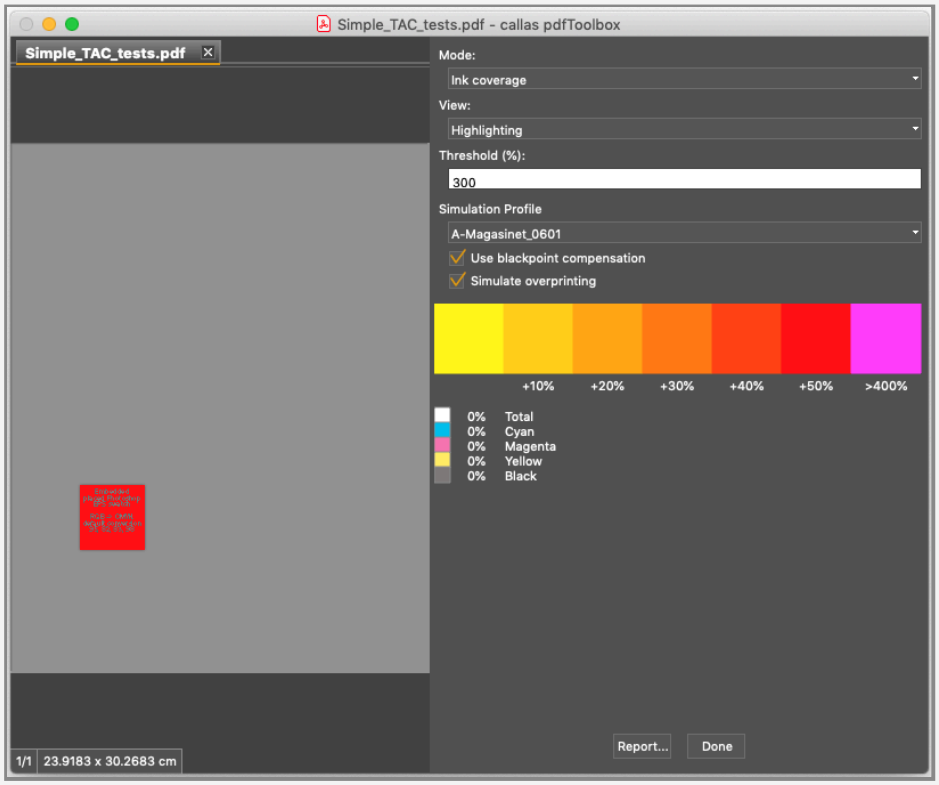
Alternately, you can edit existing Fixups and create duplicates in order to e.g. produce variants depending on the objects to be converted.

To import your own DeviceLink Profiles, simply click on the “Import” button.

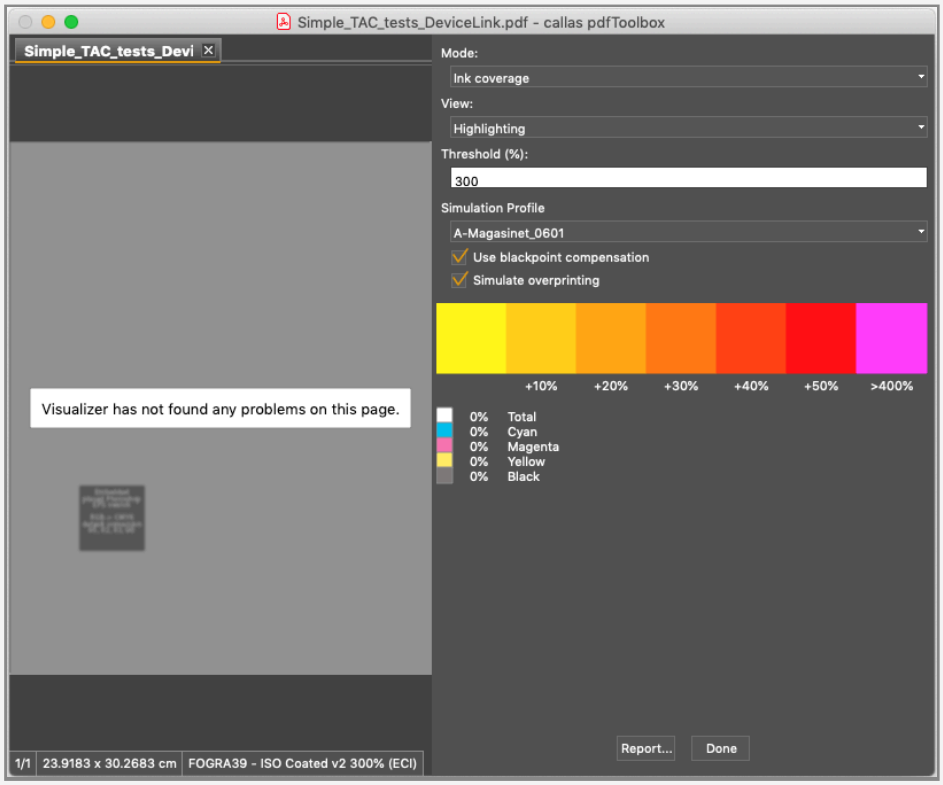
Use a DeviceLink Fixup to reduce color limit to 300%

The input file’s color limit is over 300%.

This can be visualized in the standalone version using the “Highlight total color limit” tool in the “View” menu.



After applying the Fixup named “Convert colors using DeviceLink sheetfed offset, reduce to 300% ink limit (ISO)”, the output file has a color limit below 300%.



List of integrated DeviceLink profiles

RGB to CMYK

Source	Target
Adobe RGB (1998)	ISO Coated v2
Adobe RGB (1998)	PSO Coated v3
sRGB	ISO Coated v2
sRGB	PSO Coated v3

Conversions between standards - Europe

Source	Target
ISO Coated v2	PSO Coated v3
PSO Coated v3	ISO Coated v2
PSOsc b_paper v3 (FOGRA54)	ISO Coated v2

Source	Target
PSOsc b_paper v3 (FOGRA54)	PSO Coated v3
ISO Coated v2	eciCMYK
PSO Coated v3	eciCMYK
eciCMYK	ISO Coated v2
eciCMYK	PSO Coated v3
ISO Coated v2	PSOsc b_paper v3 (FOGRA54)
PSO Coated v3	PSOsc b_paper v3 (FOGRA54)
PSO Uncoated v2	PSO Uncoated v3
PSO Uncoated v3	PSO Uncoated v2
PSO Coated v3	PSO Uncoated v3
ISO Coated v2	ISO Newsprint
ISO Coated v2	PSO LWC Standard
ISO Coated v2	PSO LWC Improved
ISO Coated v2	ISO Web Coated
ISO Coated v2	SC Paper
ISO Coated v3	ISO Newsprint
ISO Coated v3	PSO LWC Standard
ISO Coated v3	PSO LWC Improved
ISO Coated v3	ISO Web Coated
ISO Coated v3	SC Paper
PSO LWC Improved	ISO Newsprint
ISO Web Coated	ISO Newsprint

Conversions between standards – Europe/Japan

Source	Target
ISO Coated v2	Japan Color 2001 Coated
PSO Coated v3	Japan Color 2001 Coated
Japan Color 2001 Coated	ISO Coated v2
Japan Color 2001 Coated	PSO Coated v3

Conversions between standards – Europe/US

Source	Target
ISO Coated v2	Coated GRACoL 2006
PSO Coated v3	Coated GRACoL 2006
Coated GRACoL 2006	ISO Coated v2
Coated GRACoL 2006	PSO Coated v3
Coated GRACoL 2006	Web Coated SWOP grade 3
Coated GRACoL 2006	Web Coated SWOP grade 5
Web Coated SWOP grade 3	Web Coated SWOP grade 5

TAC reduction

Source and target profile	TAC percentage
ISO Coated v2	330
ISO Coated v2	300
PSO Coated v3	300
PSO Uncoated v3	280
ISO Uncoated	280
ISO Web Coated	300
PSO LWC Standard	300
PSO LWC Improved	300

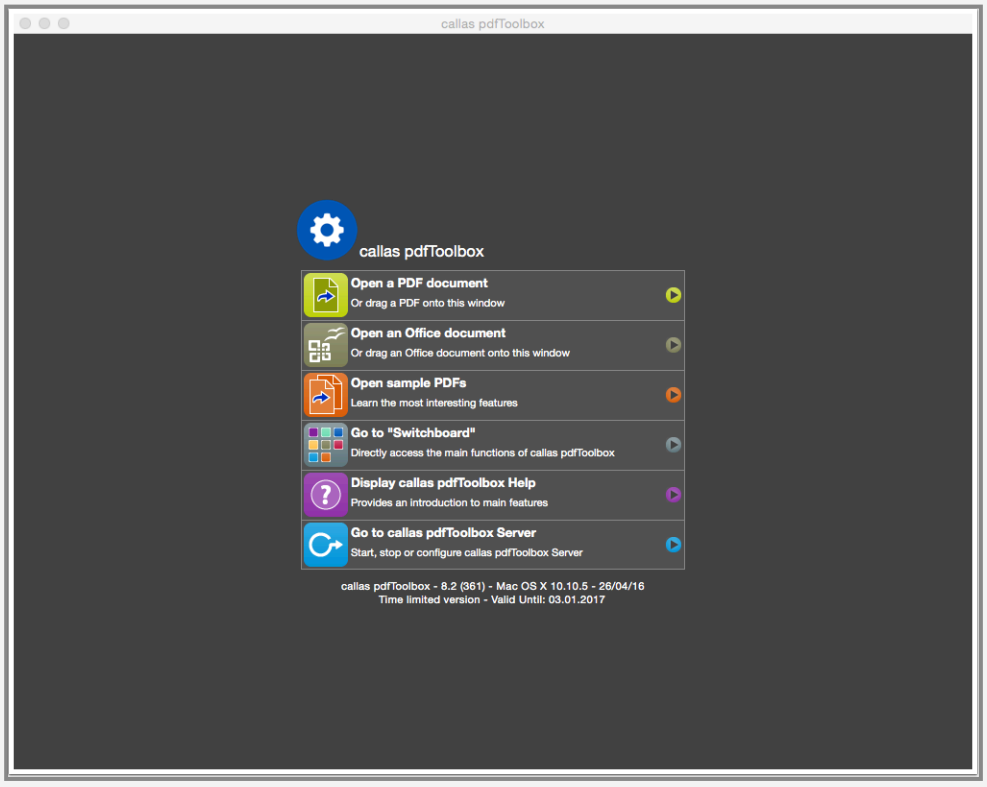
Source and target profile	TAC percentage
Coated GRACoL 2006	320
SWOP 3	300
SWOP 5	280
Japan Color 2011	340
ISO News	240

7.11 Use included DeviceLink profile to convert ISO Coated v2 ↔ PSO Coated v3 (ECI)

pdfToolbox includes a profile for color conversion and a device link for PSO Coated v3 ICC profile. The third version of Process Standard Offset was introduced in September 2015.

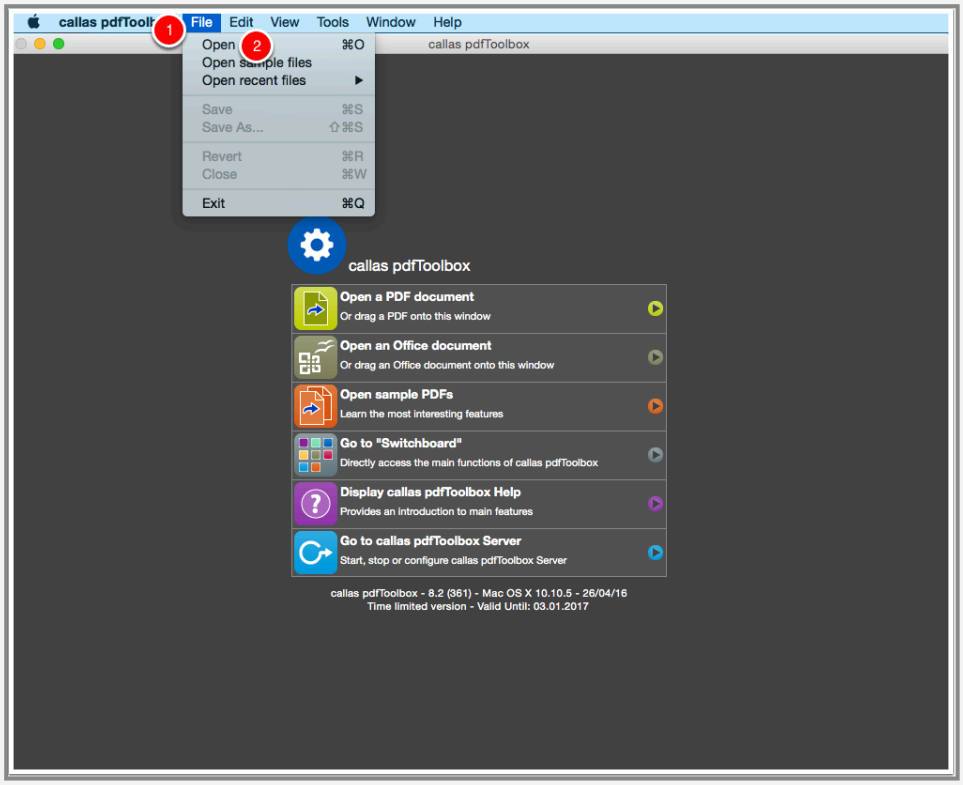
There are two DeviceLink Fixups, which can convert color from "ISO Coated v2 (ECI)" to "PSO Coated v3 (ECI)" or convert in reverse direction "PSO Coated v3 (ECI)" to "ISO Coated v2 (ECI)".

Launch pdfToolbox Desktop



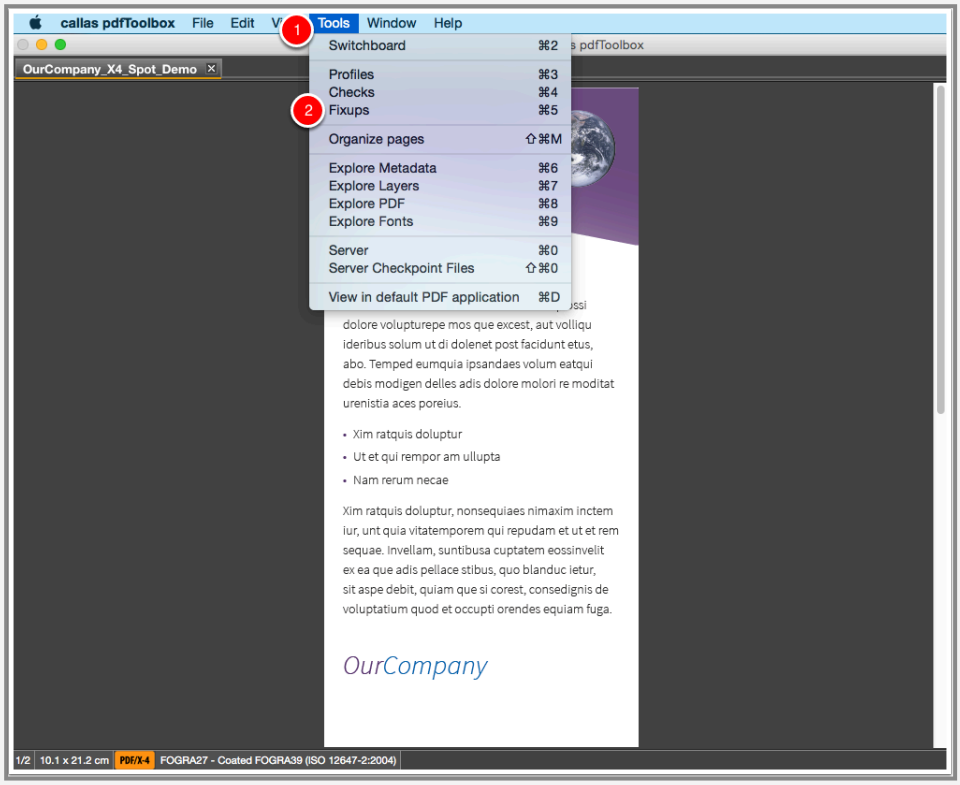
Open the PDF file "OurCompany_X4_Spot_Demo

file.pdf"



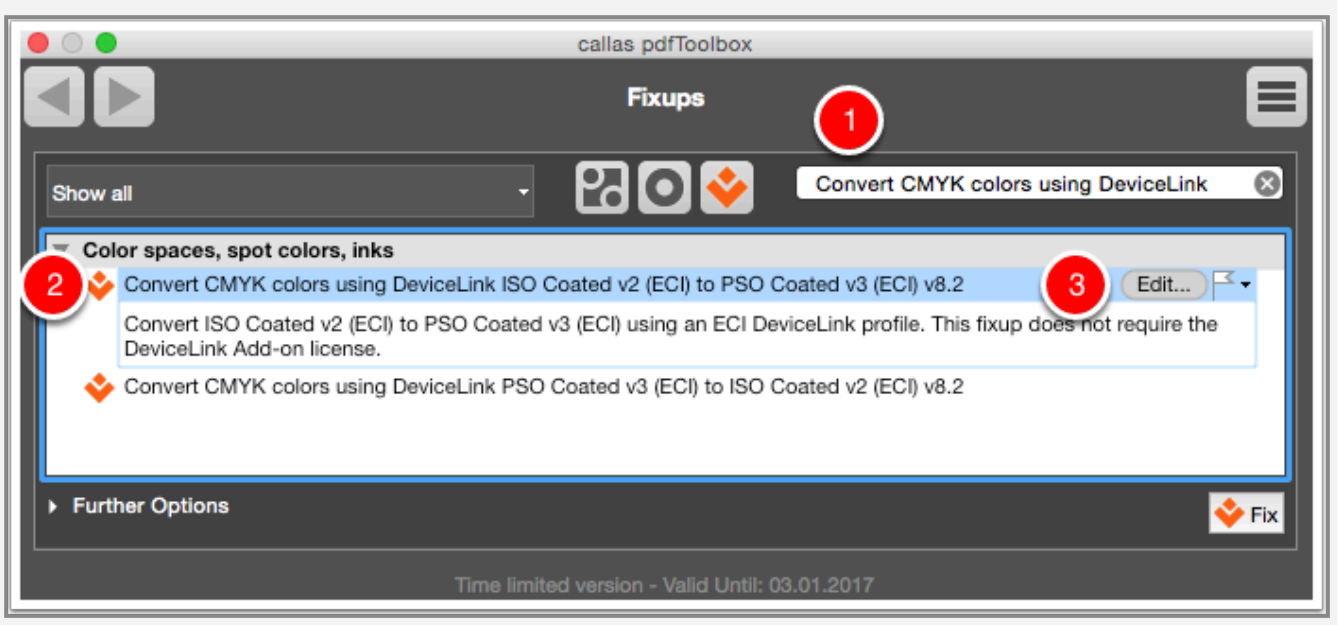
1. Go to "File".
2. Click "Open".

Open the Fixups dialog



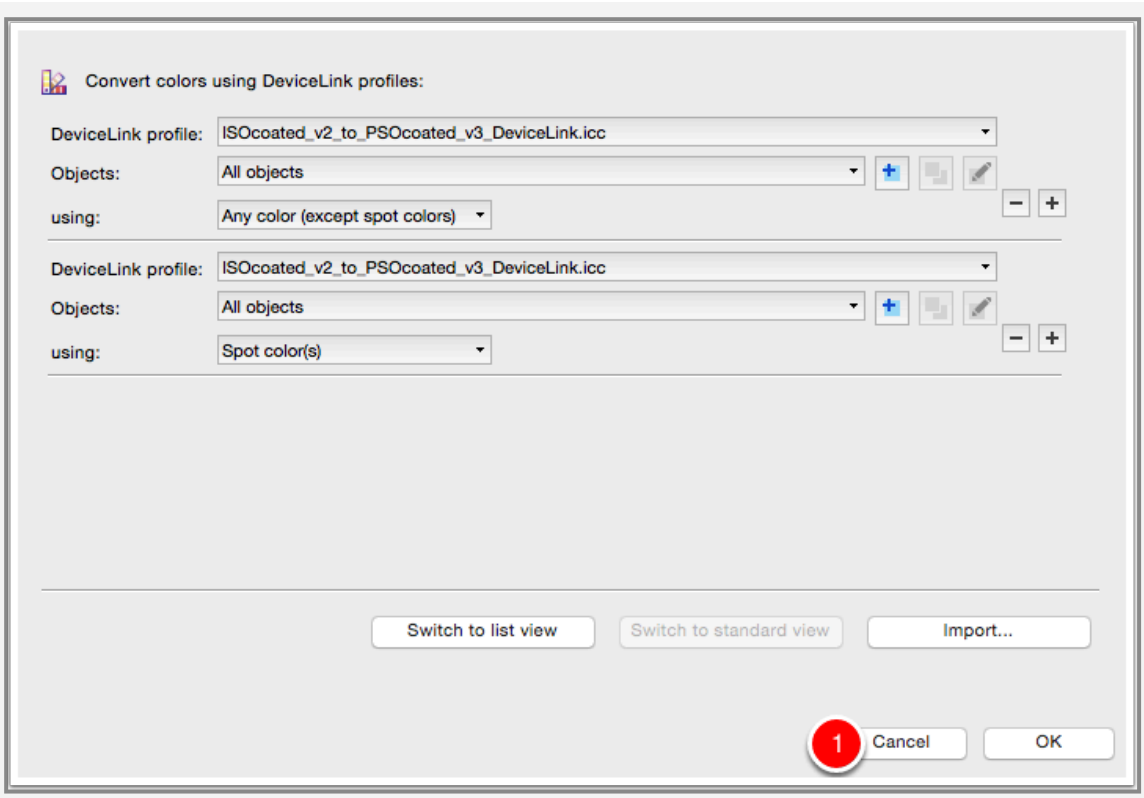
1. Go to "Tools".
2. Click "Fixups".

Search to Fixup



1. In the search field search to "Convert CMYK colors using DeviceLink".
2. Select the Fixup "Convert CMYK colors using DeviceLink ISO Coated v2 (ECI) to PSO Coated v3 (ECI) v8.2".
3. Click "Edit" to inspect the Fixup.

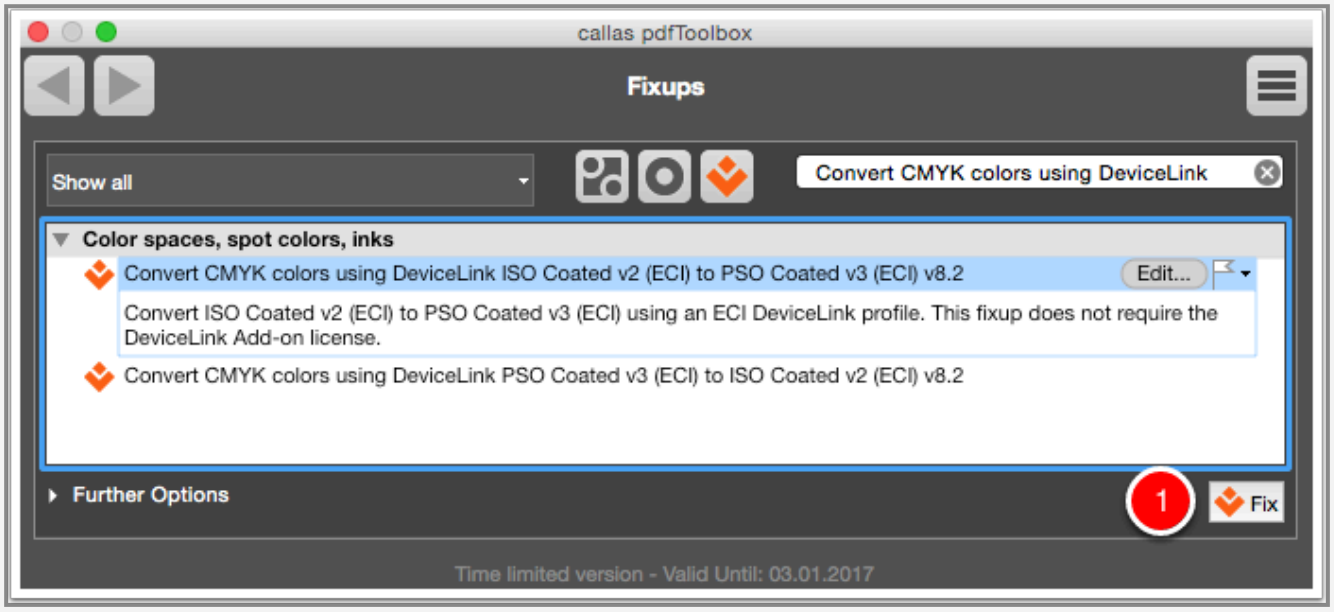
Inspect the Fixup



The PDF file uses an ISO Coated v2 profile (Coated FOGRA39) that will be converted by an ECI DeviceLink profile to PSO Coated v3. This will have only effect on CMYK colors.

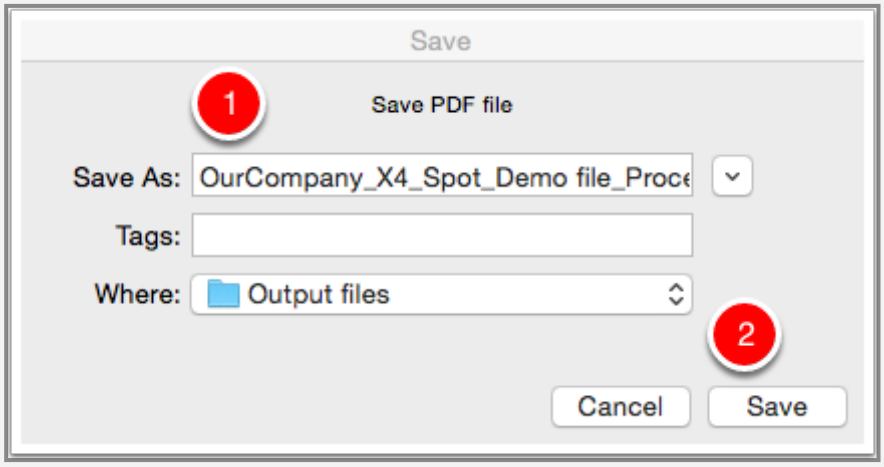
1. Click "Cancel".

Apply the Fixup



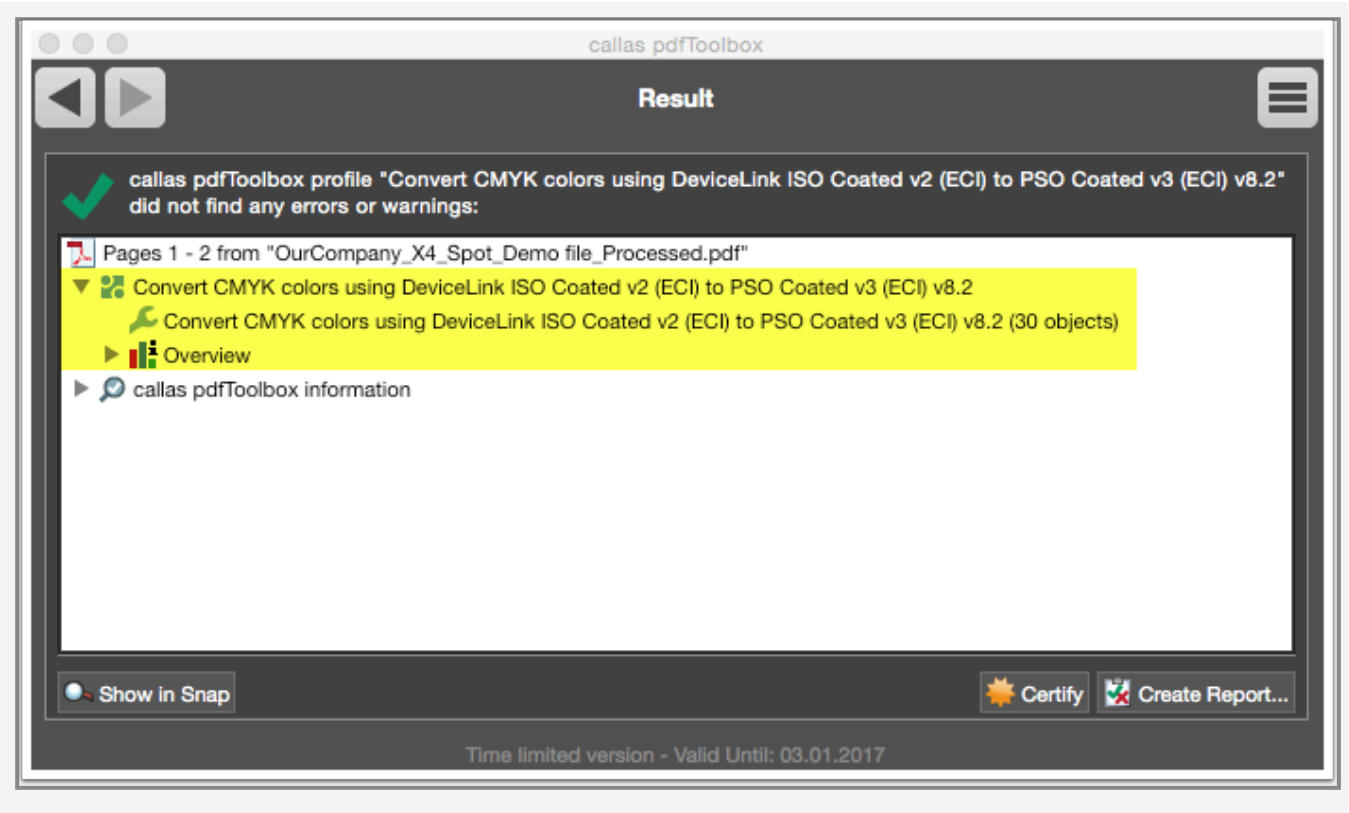
1. Click "Fix".

Save the output PDF file



1. Save the output PDF file as "OurCompany_X4_Spot_Demo file_Processed".
2. Click "Save".

Inspect the preflight report



A green check mark is shown, refers to a successful correction. All CMYK colors in the PDF file with an ISO Coated v2 profile (Coated FOGRA39) are converted to PSO Coated v3 by an ECI DeviceLink profile.

7.12 Replace existing ICC profile

pdfToolbox has the functionality to replace ICC profiles. This tutorial shows how to tag four color images with different ICC source profiles to one existing ICC profile. That is only possible when the same color space is used (DeviceCMYK or DeviceRGB) in the PDF document.

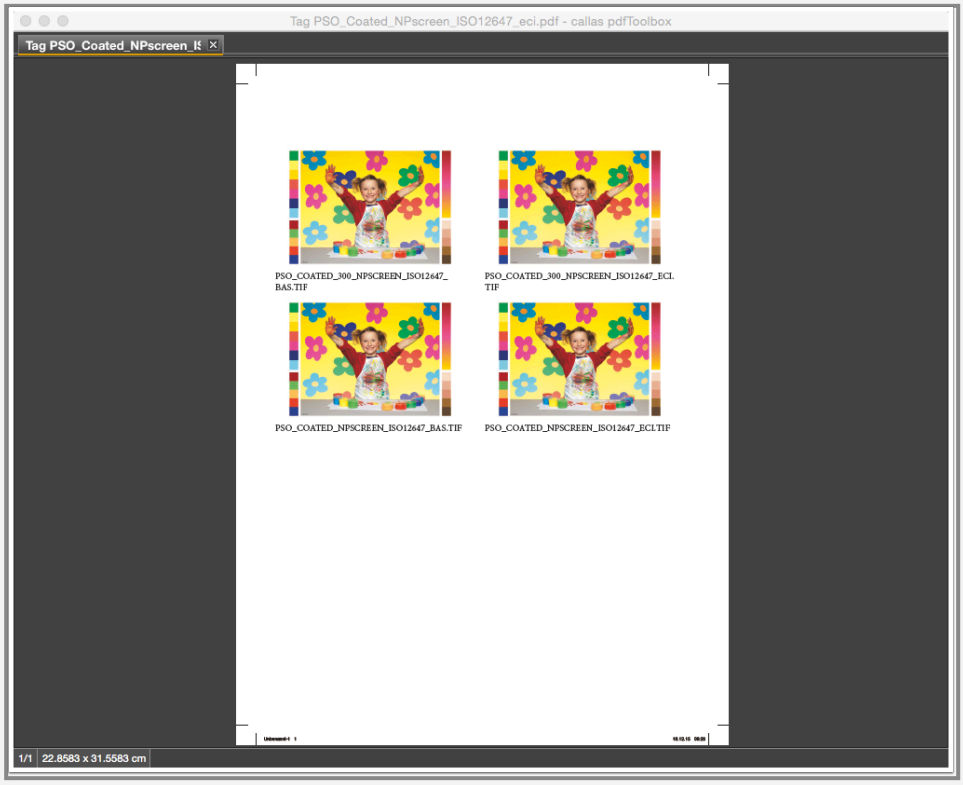


Tag_PSO_Coated_NPscreen_ISO12647_eci.pdf



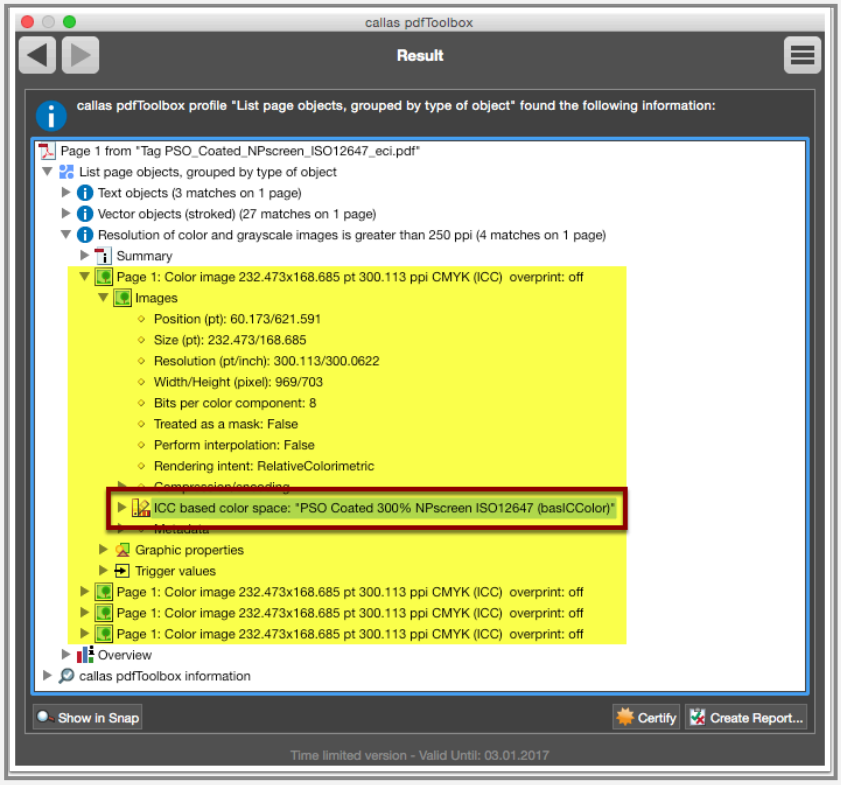
Tag_PSO_Coated_NPscreen_ISO12647_eci.kfpx

Open the attached PDF document in pdfToolbox (File > Open)



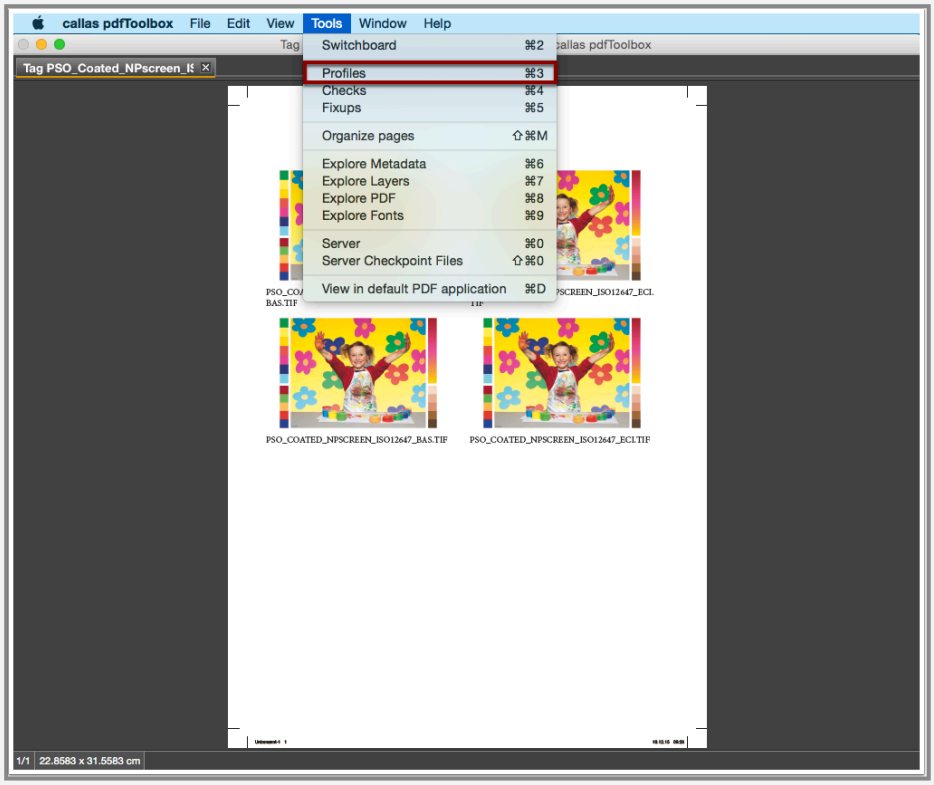
The PDF document has four images, each with their own ICC source profile.

PDF analysis preflight report of the original PDF document

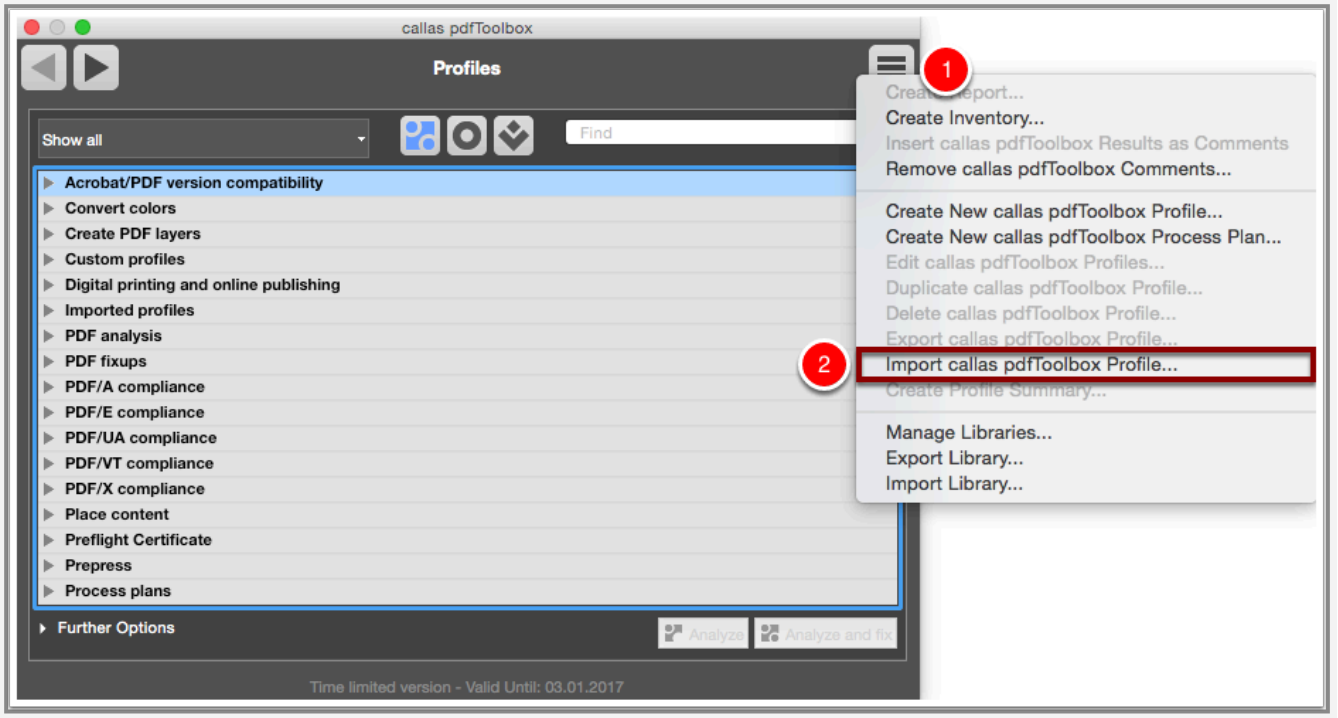


The predefined PDF analysis profile "List page objects, grouped by type of object" provides the user PDF object information. It can be executed to get text, vector, image, ... information used in the PDF document. In the original attached PDF document the image in the top left has the ICC source profile "PSO Coated 300% NPscreen ISO12647 (baslCColor)".

Open the Profiles dialog (Tools > Profiles)

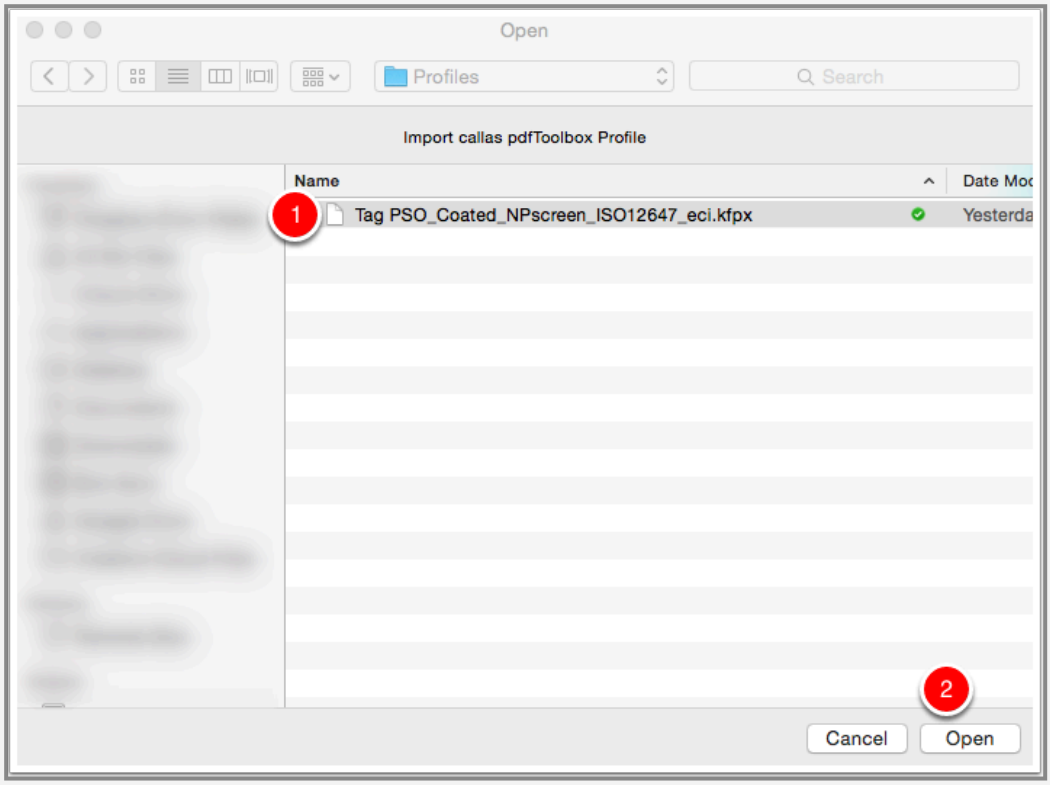


Import the attached Profile



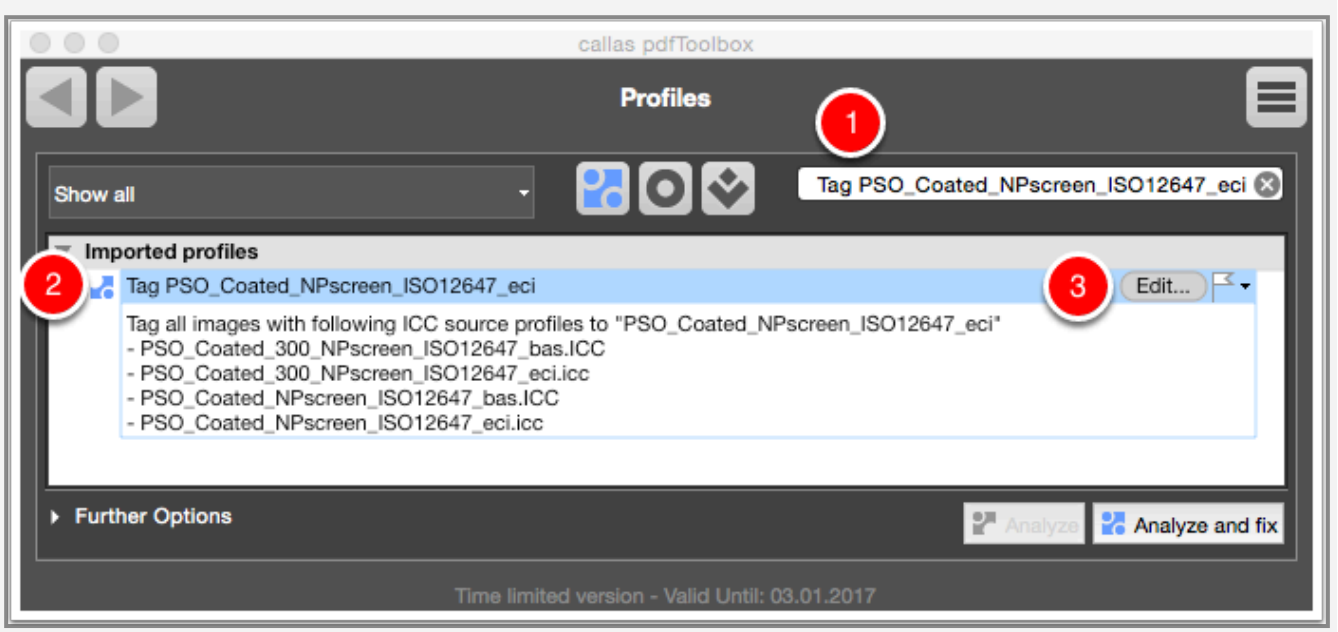
1. Click on the Profile action button.
2. Choose "Import callas pdfToolbox Profile".

Browse to the attached Profile



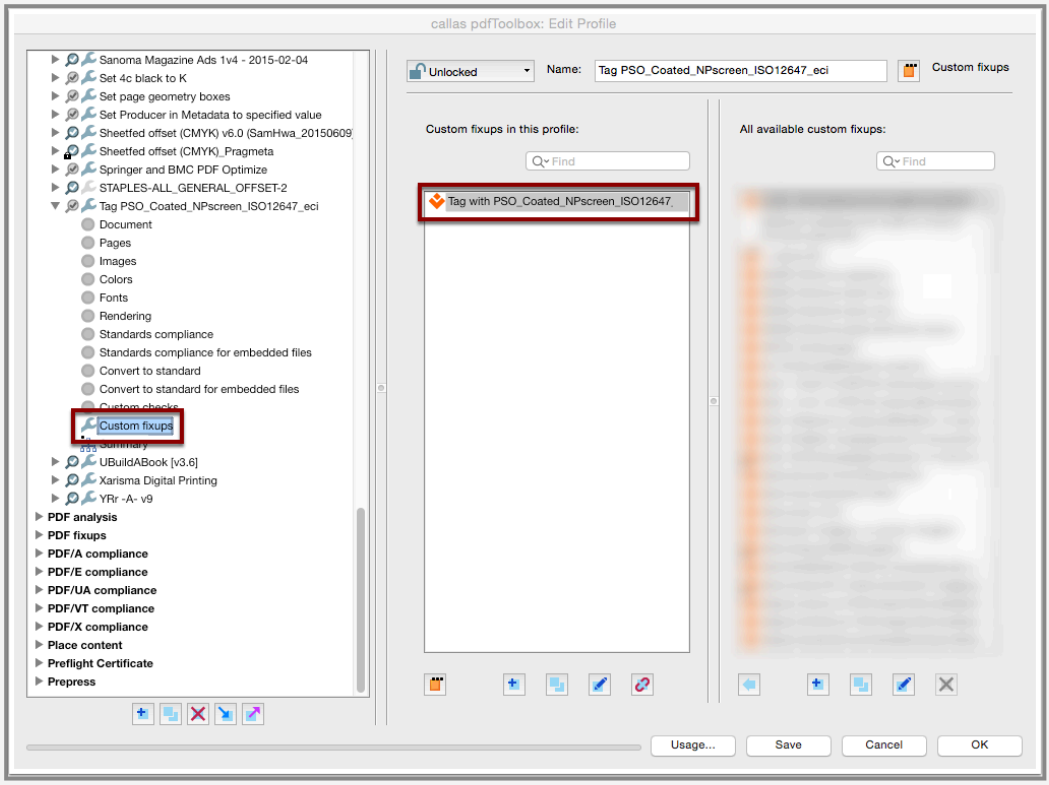
1. Select the Profile "Tag PSO_Coated_NPscreen_ISO12647_eci.kfpx".
2. Click "Open".

Go to the open Profiles dialog



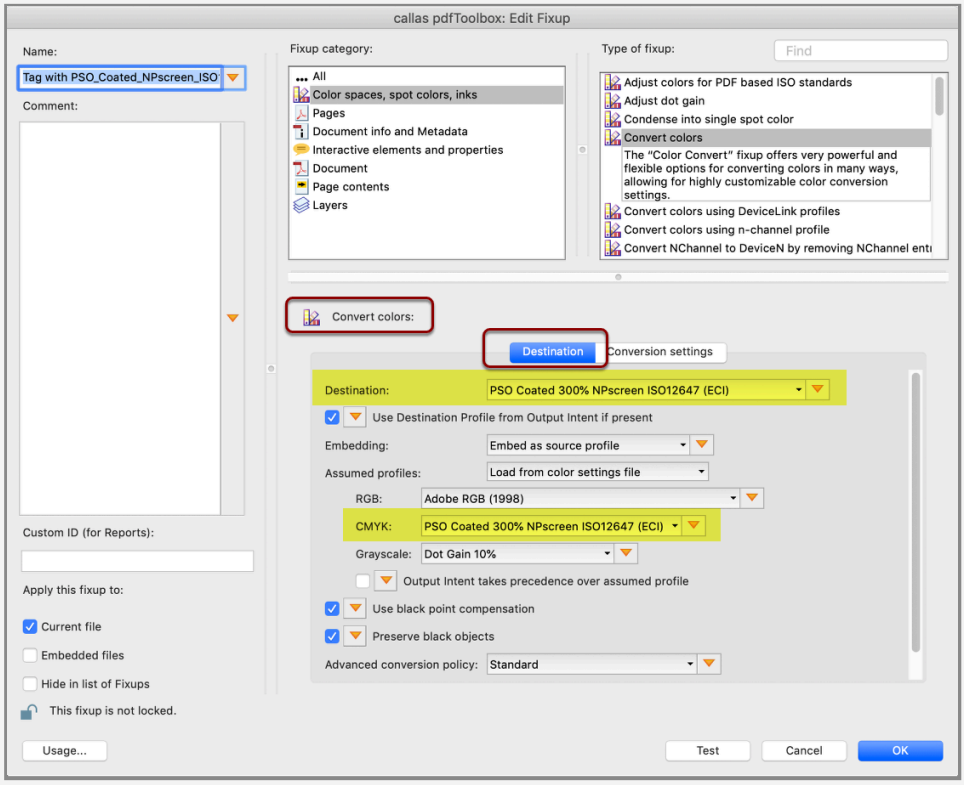
1. In search field search to "Tag PSO_Coated_NPscreen_ISO12647_eci".
2. Select the "Tag PSO_Coated_NPscreen_ISO12647_eci" Profile.
3. Click "Edit" to analyze the imported Profile.

Analyze imported Profile: Custom fixups



The Profile has one custom Fixup "Tag PSO_Coated_NPscreen_ISO12647_eci".

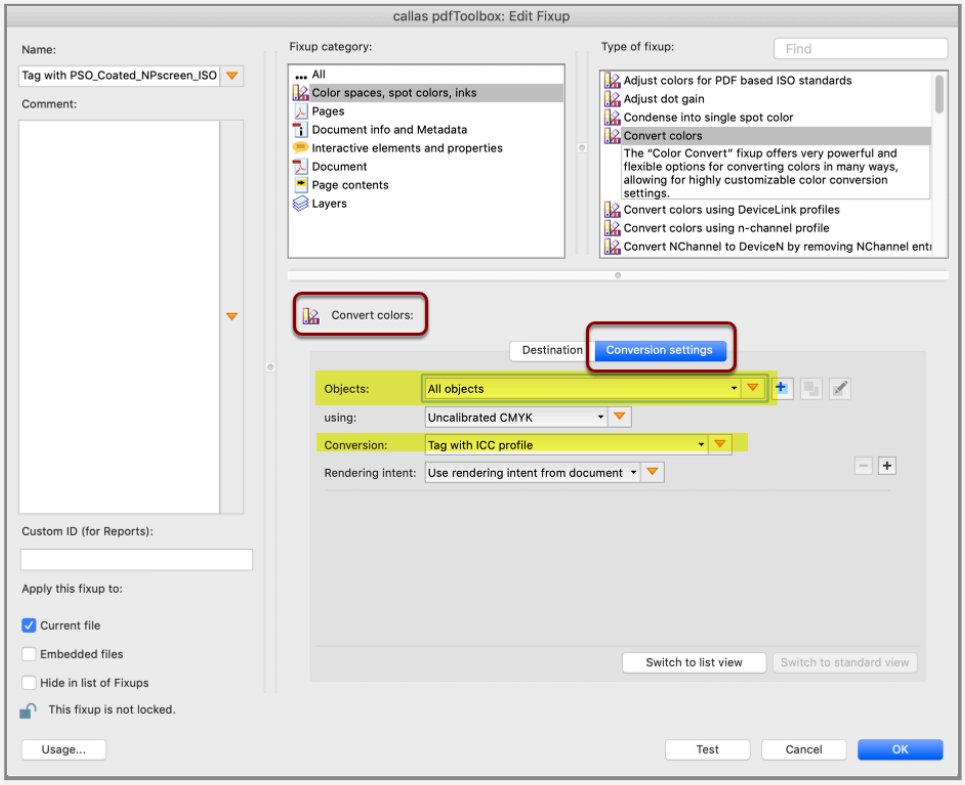
Analyze imported Profile: Custom fixup "Convert colors"



In the custom Fixup the Fixup "Convert colors" is used. The settings in the "Destination" tab are:

- Destination: "PSO Coated 300% NPscreen ISO12647 (ECI)" ICC profile
- CMYK: "PSO Coated 300% NPscreen ISO12647 (ECI)" ICC profile

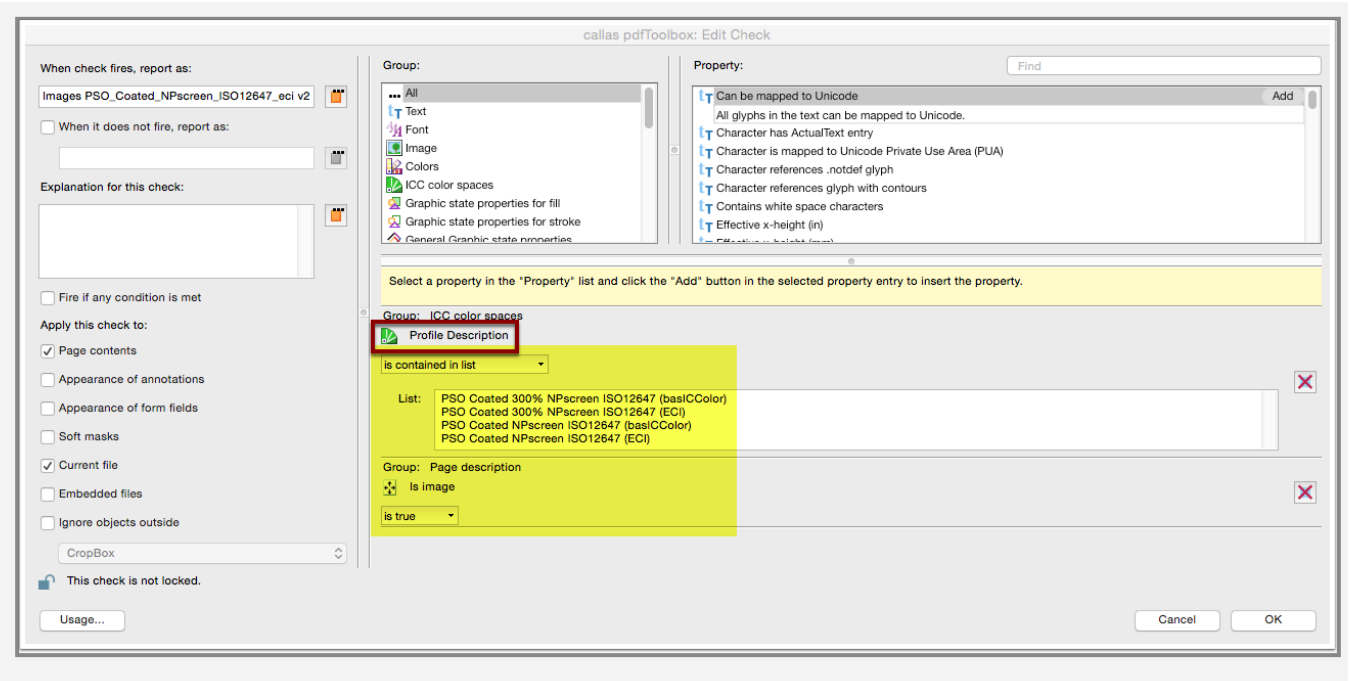
Analyze imported Profile: Custom fixup "Convert colors" conversion settings



In the custom Fixup the Fixup "Convert colors" is used. The settings in the "Conversion settings" tab are:

- Objects: "Images PSO_Coated_NPscreen_ISO12647_eci v2" check
- Conversion: Tag with ICC profile

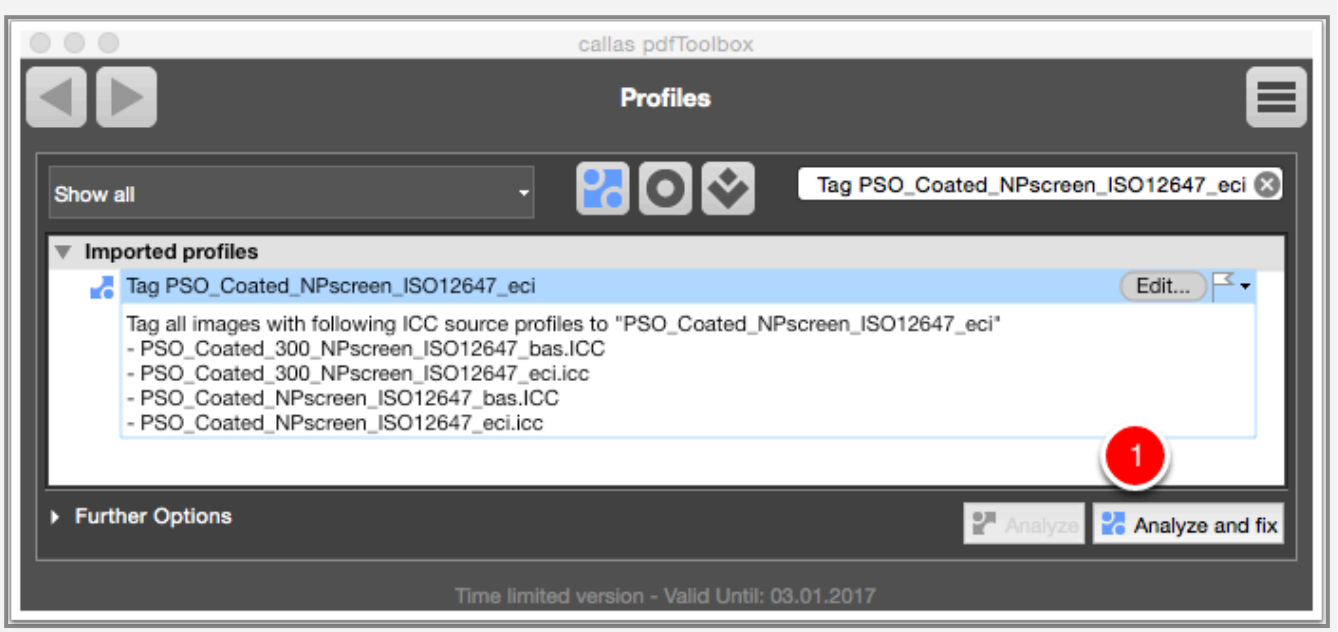
Analyze imported Profile: Custom fixup "Convert colors" objects check



The Fixup "Convert colors" uses a custom Check that has the Checks "Profile Description" and "Is image". The custom Check is configured to find images that have one of the following ICC source profiles:

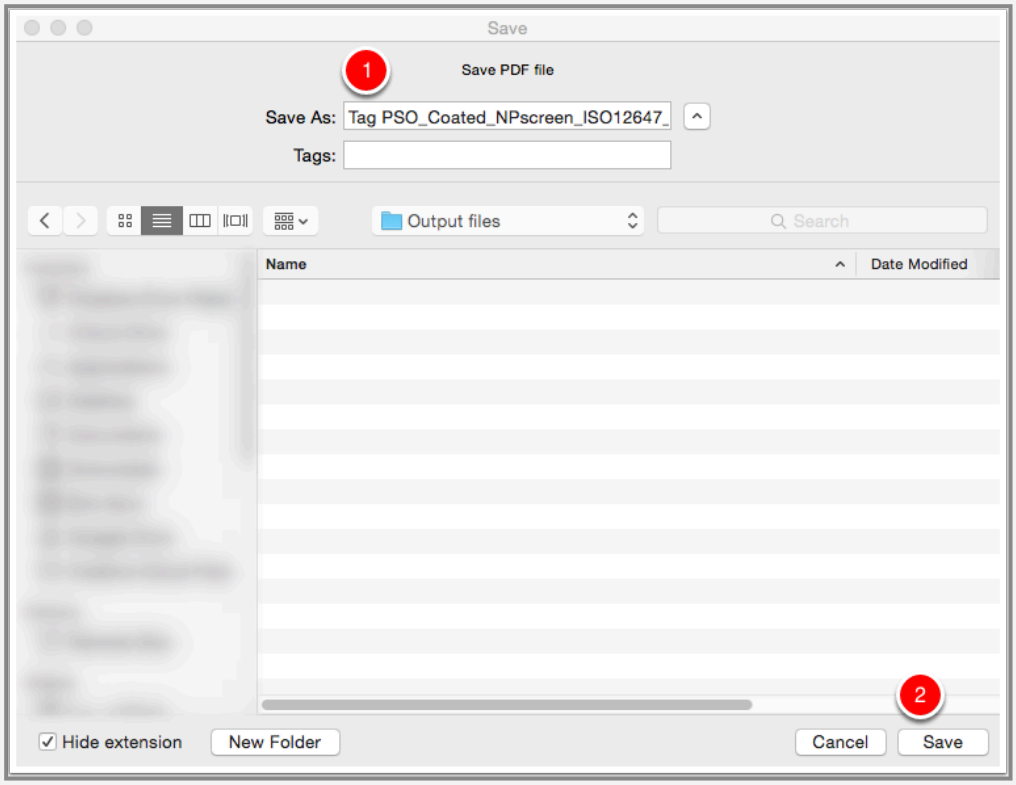
- PSO Coated 300% NPscreen ISO12647 (basICColor)
- PSO Coated 300% NPscreen ISO12647 (ECI)
- PSO Coated NPscreen ISO12647 (basICColor)
- PSO Coated NPscreen ISO12647 (ECI)

Go to the open Profiles dialog



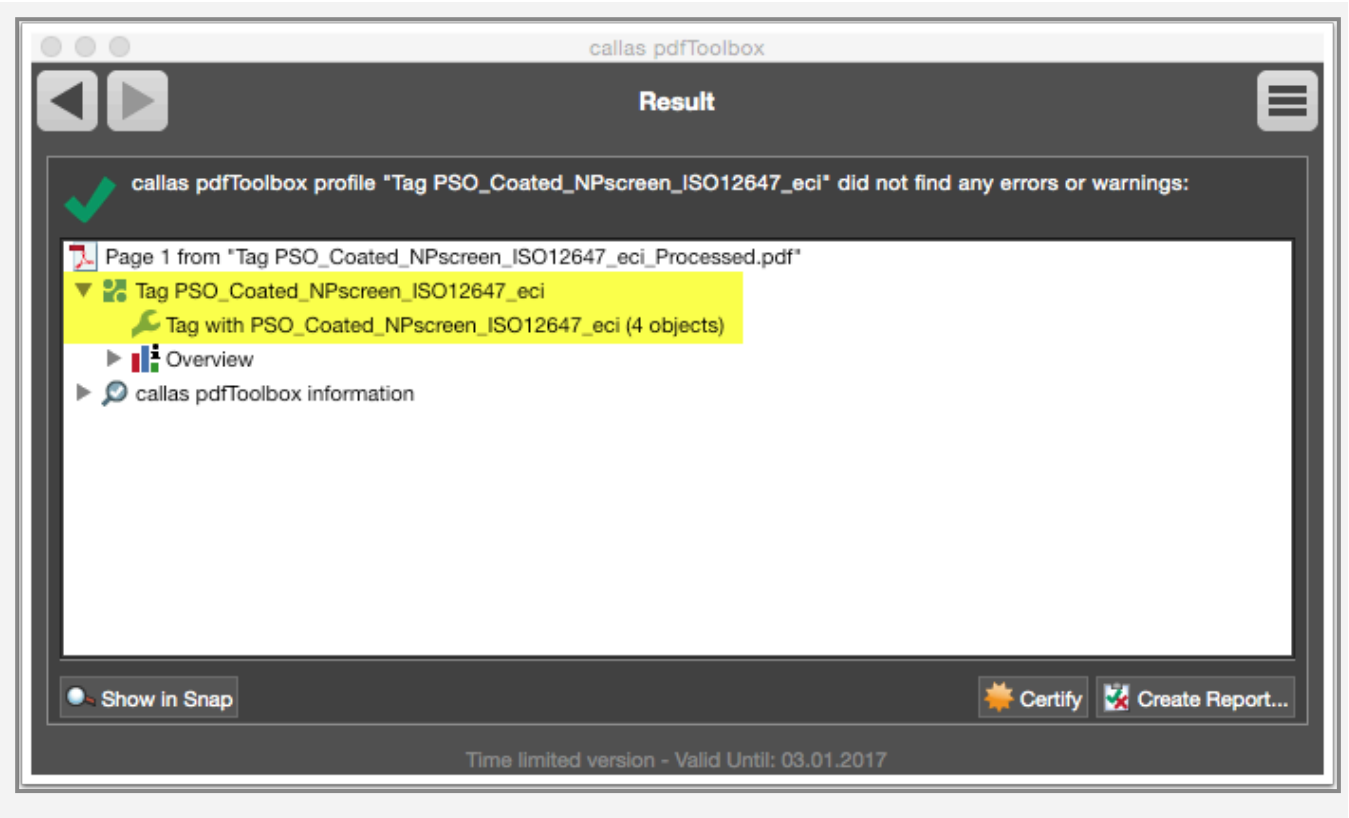
1. Click "Analyze and fix" to apply the Profile on the PDF document loaded in pdfToolbox.

Save the output PDF document



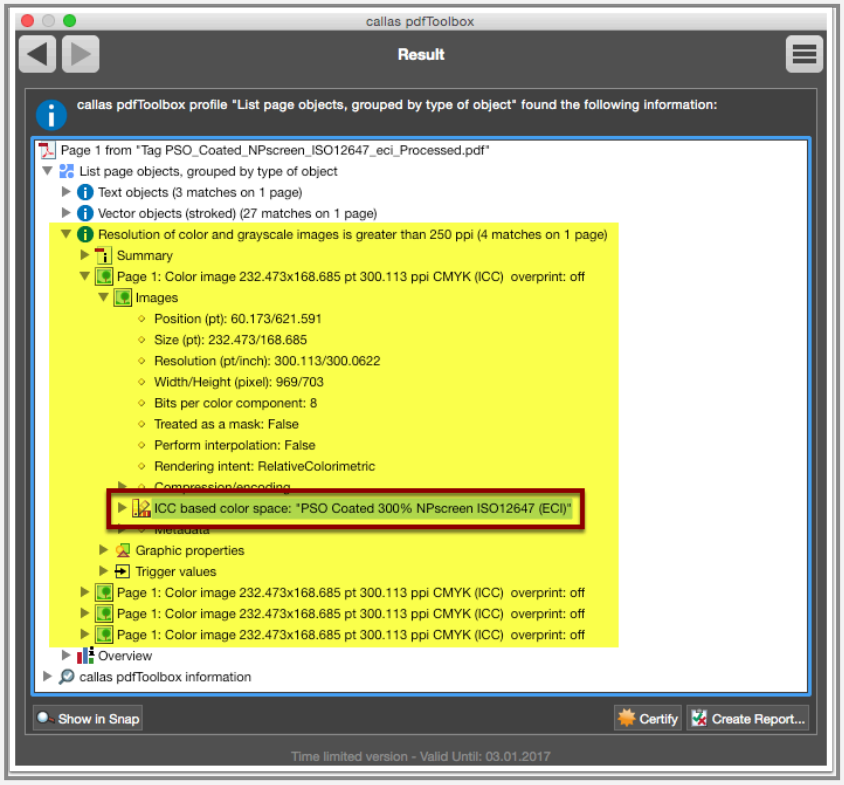
1. Save the output PDF document as "Tag PSO_Coated_NPscreen_ISO12647_eci_Processed".
2. Click "Save".

The preflight report



The ICC source profile of the four images are replaced by the existing ICC profile "PSO Coated 300% NPscreen ISO12647 (ECI)".

PDF analysis preflight report of the processed PDF document



The image in the top left has the ICC profile "PSO Coated 300% NPscreen ISO12647 (ECI)". The same for the other images in the PDF document.

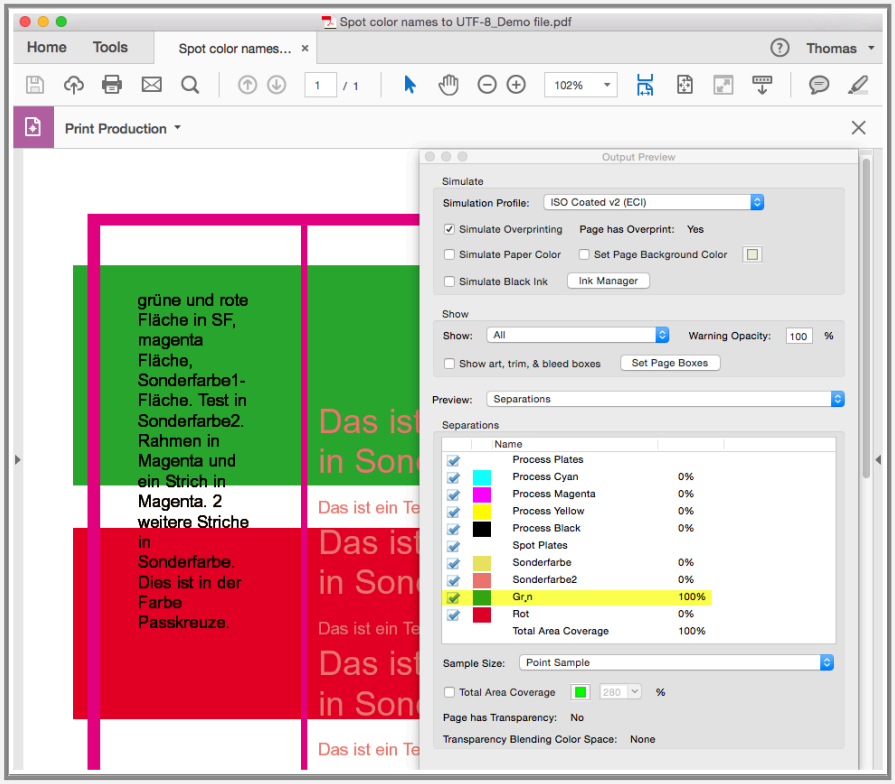
7.13 Convert spot color names to UTF-8

To make spot color names UTF-8 compliant, pdfToolbox 8.3 introduced the Fixup "Convert spot color names to UTF-8".

This tutorial explaines how to make the spot color "Grün" UTF-8 compliant.

Spot_color_names_to_UTF-8_Demo_file.pdf

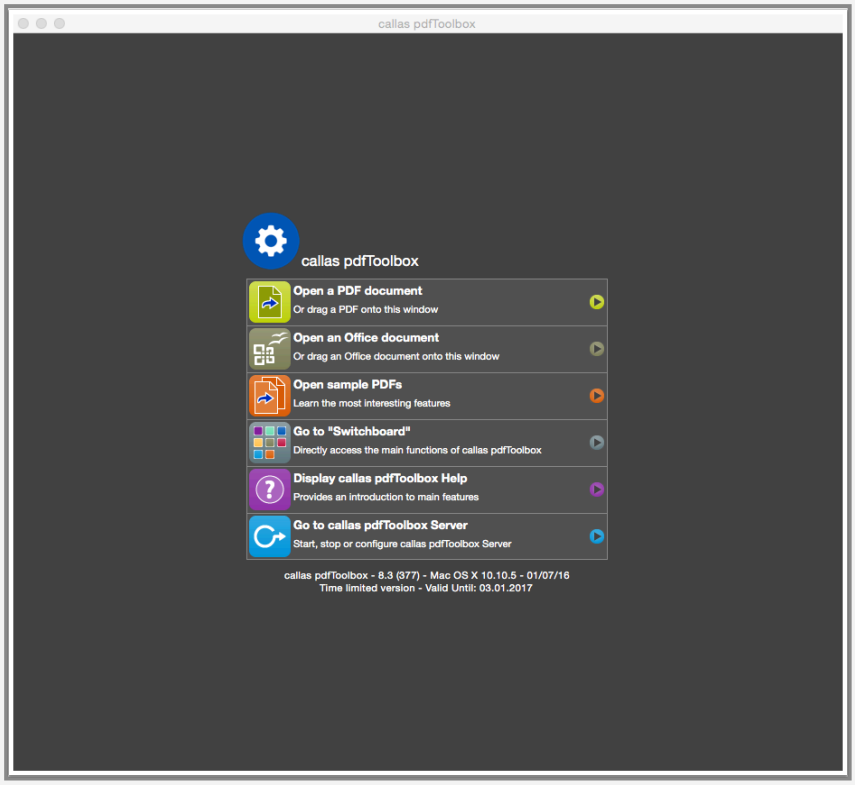
Analyze the PDF file "Spot color names to UTF-8_Demo file.pdf" in Adobe Acrobat Output Preview



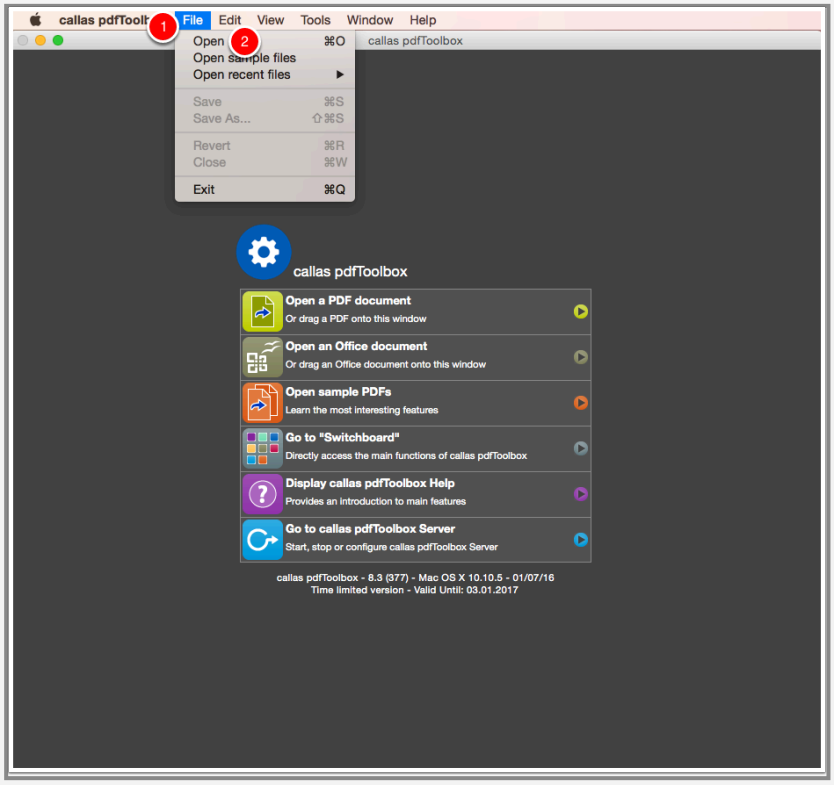
The spot color name "Grün" isn't UTF-8 compliant. So in Adobe Acrobat Output Preview the spot color name isn't correct shown.

Especially the use of umlauts, accented letters or letters from other writing systems, it is necessary to represent all characters by Unicode for having a clear display and processing. In PDF files, this is done for font and spot color names by using UTF-8.

Launch pdfToolbox Desktop

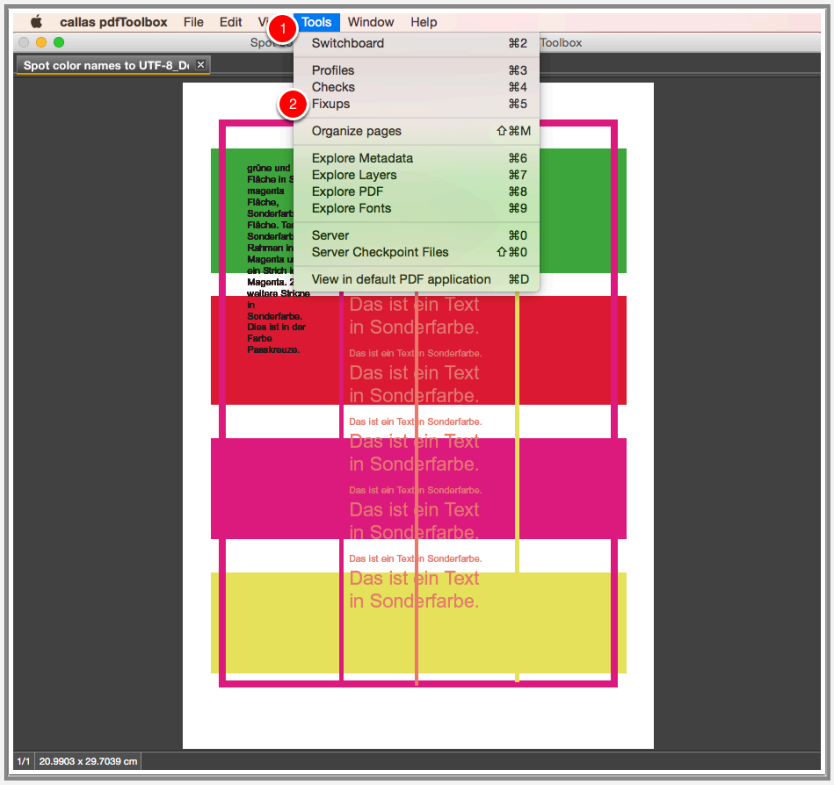


Open the PDF file "Spot color names to UTF-8_Demo file.pdf"



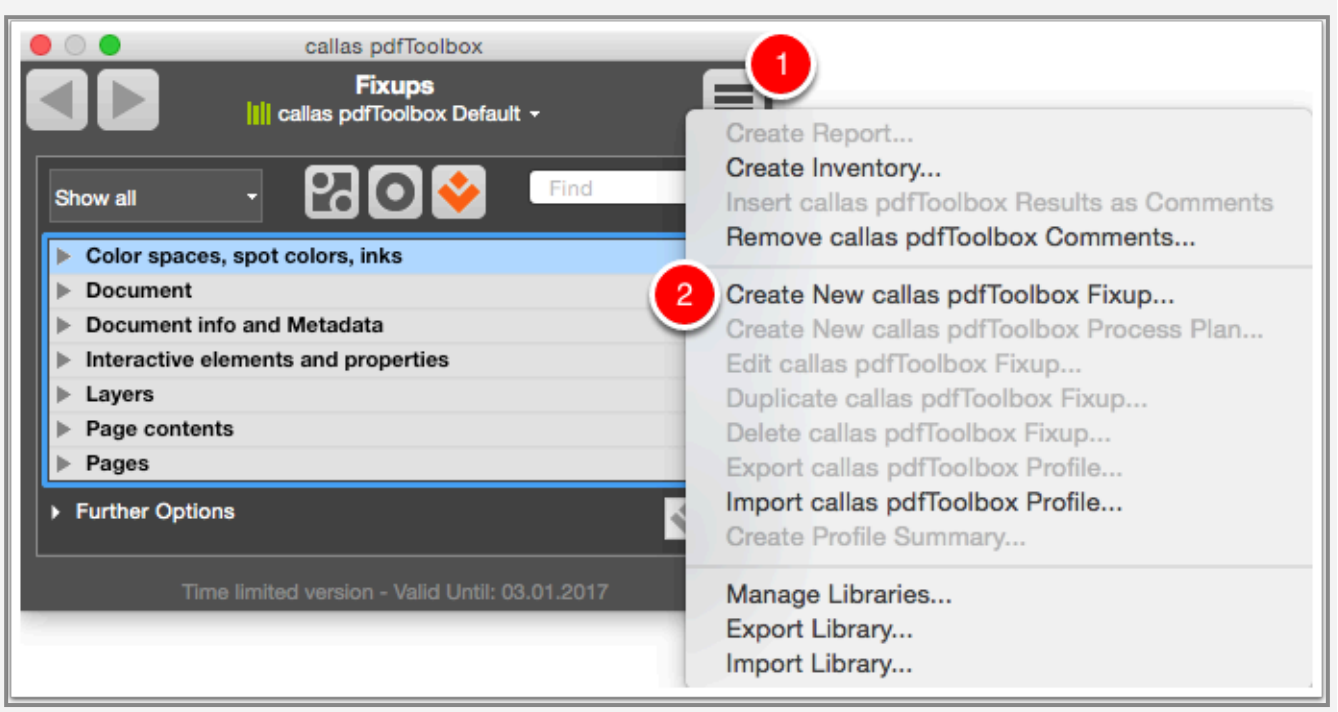
1. Go to "File".
2. Click "Open".

Open the Fixups dialog



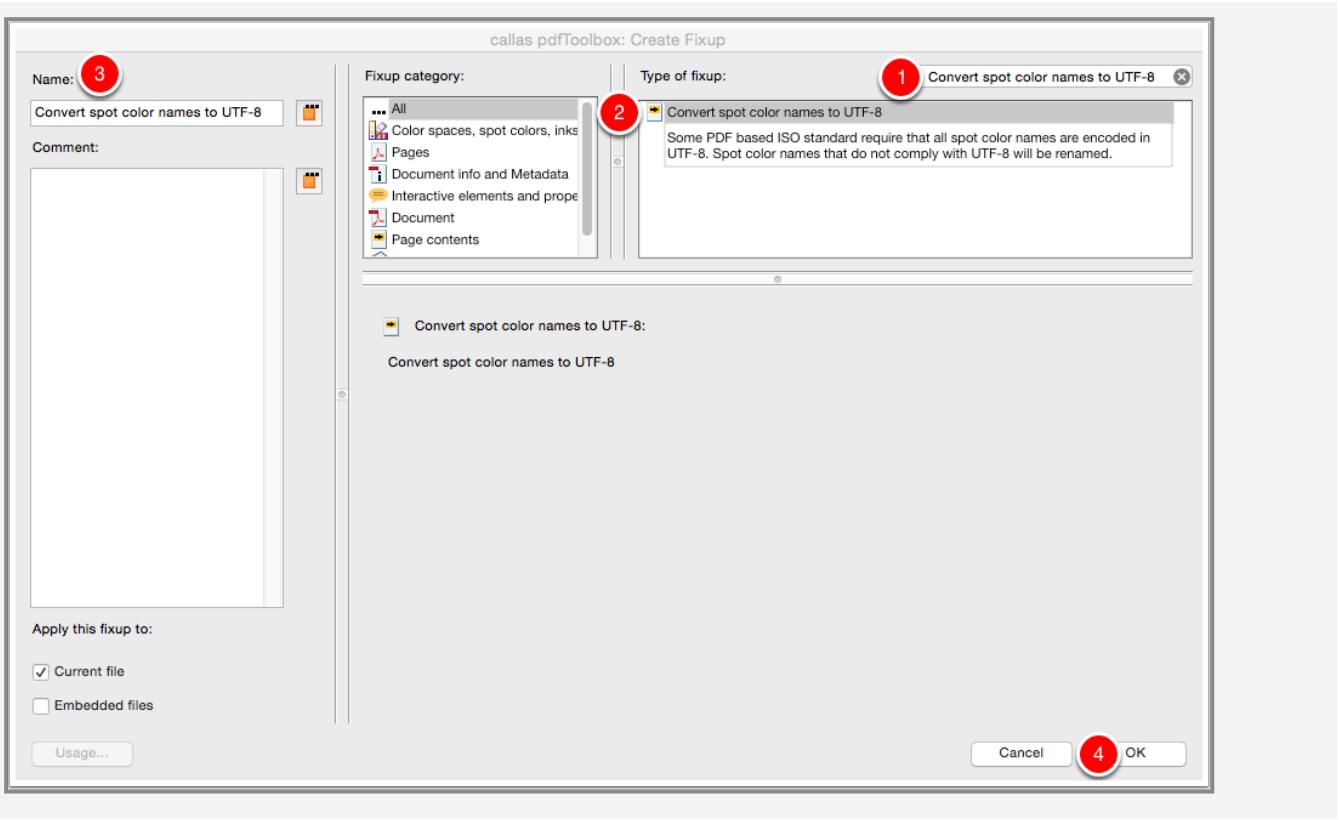
1. Go to "Tools".
2. Click "Fixups".

Create a new callas pdfToolbox Fixup



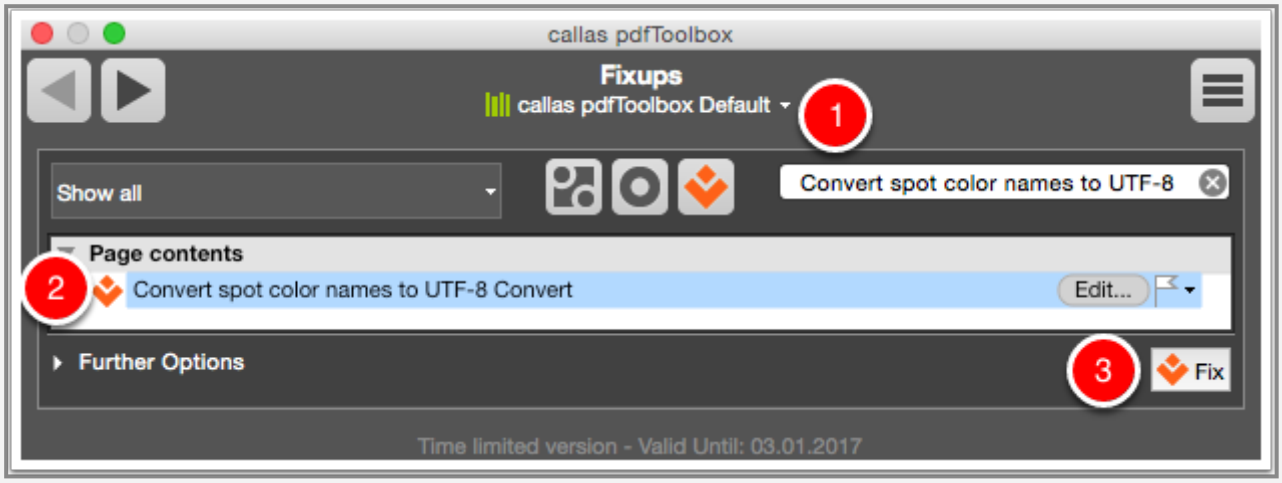
1. Click on the Fixup action button.
2. Click "Create New callas pdfToolbox Fixup".

Customize the Fixup



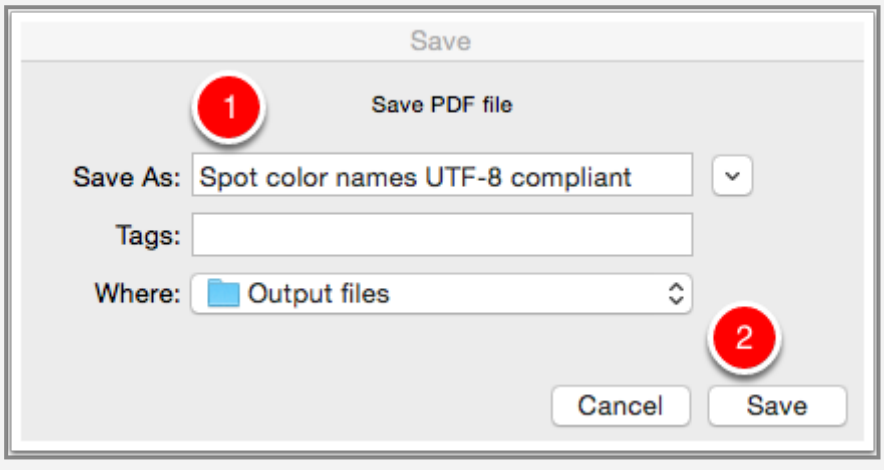
1. In the search field search to "Convert spot color names to UTF-8".
2. Select the Fixup "Convert spot color names to UTF-8".
3. Change the Fixup name to "Convert spot color names to UTF-8".
4. Click "OK".

Apply Fixup



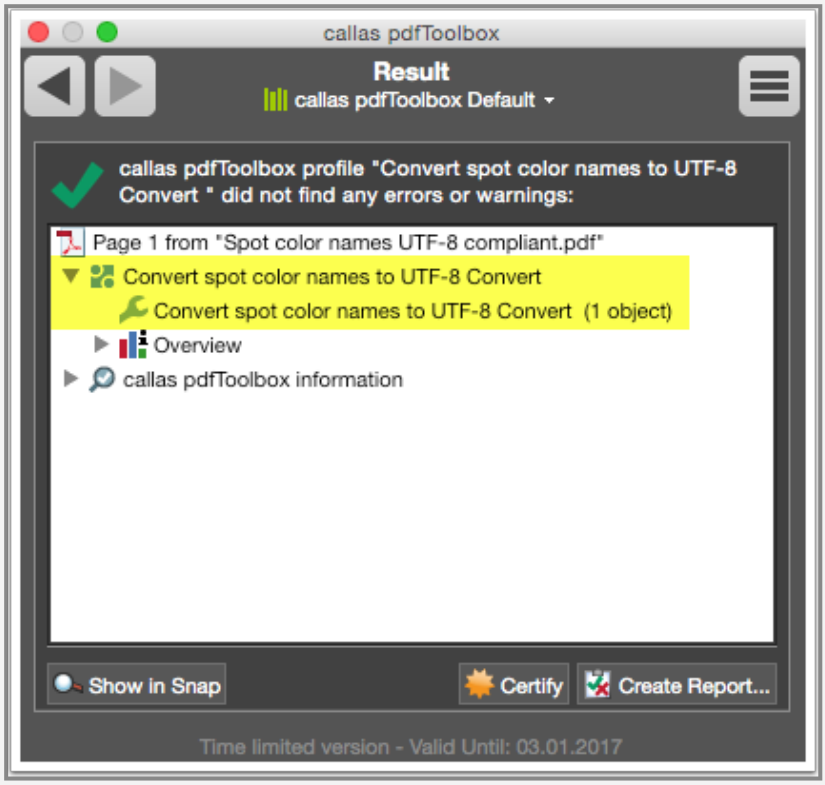
1. In the search field search to "Convert spot color names to UTF-8".
2. Select the Fixup "Convert spot color names to UTF-8".
3. Click "Fix".

Save the PDF file



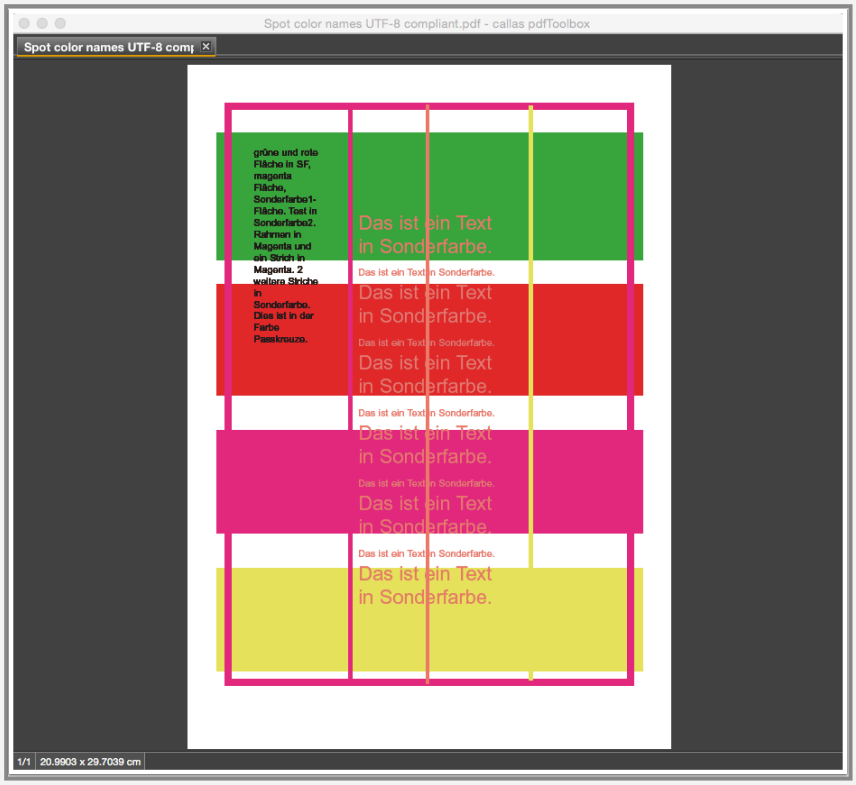
1. Save the PDF file as "Spot color names UTF-8 compliant".
2. Click "Save".

Analyze the preflight report

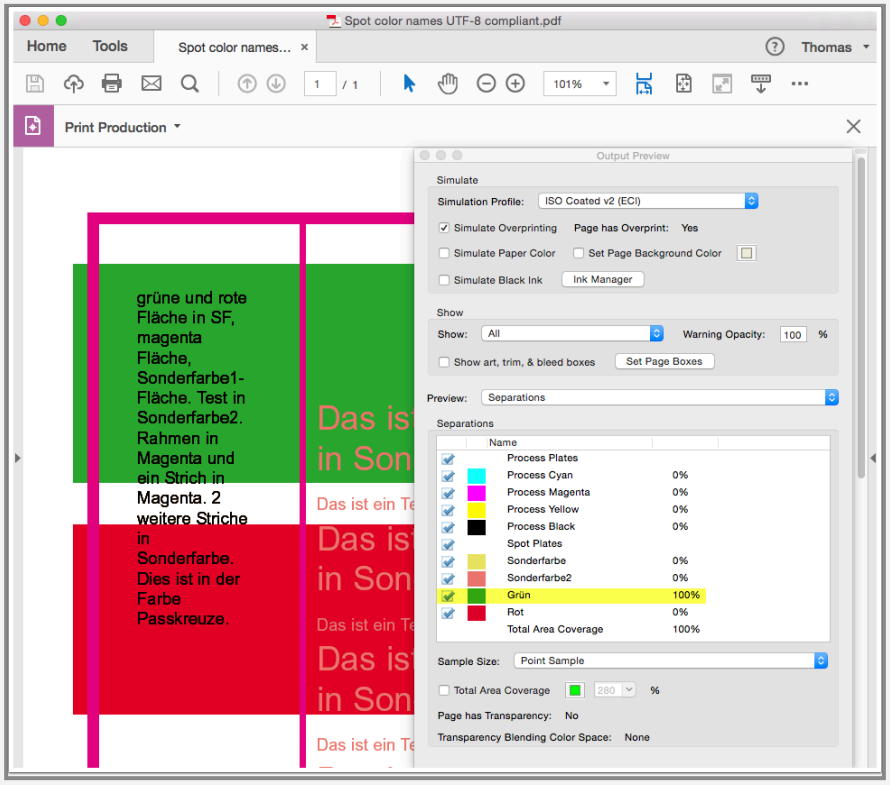


A green tick shows, the PDF file is successful preflighted.

Analyse the processed PDF file in pdfToolbox



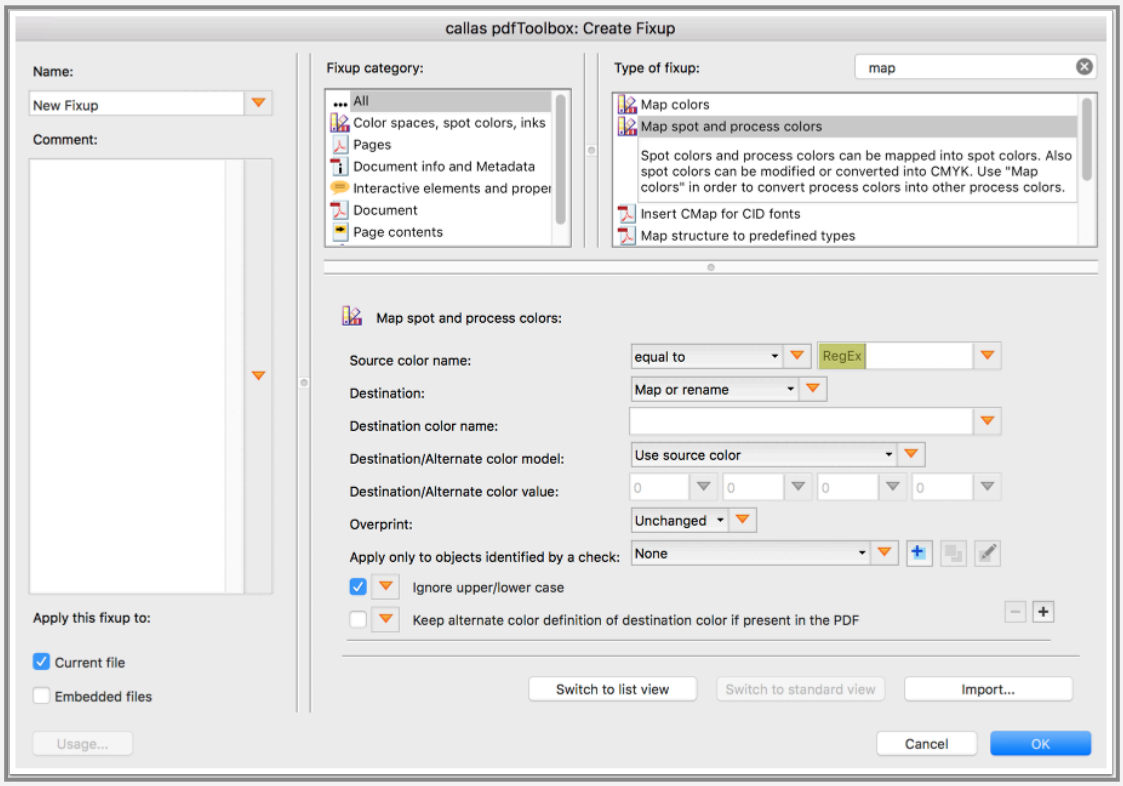
Analyze the processed PDF file in Adobe Acrobat Output Preview



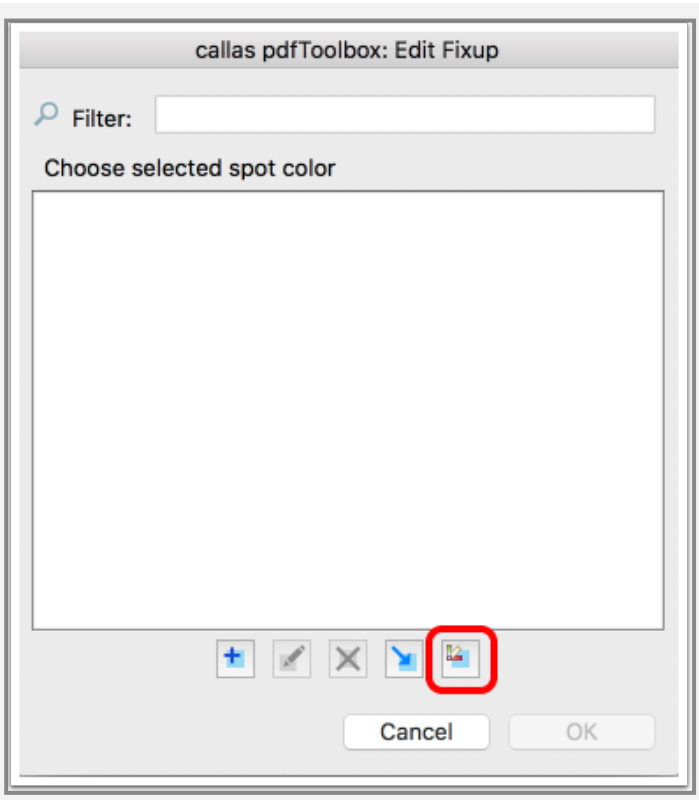
The spot color name "Grün" is now UTF-8 compliant, it shows correctly in Adobe Acrobat Output Preview.

7.14 Update all spot colors in a PDF using a spot color Swatch library

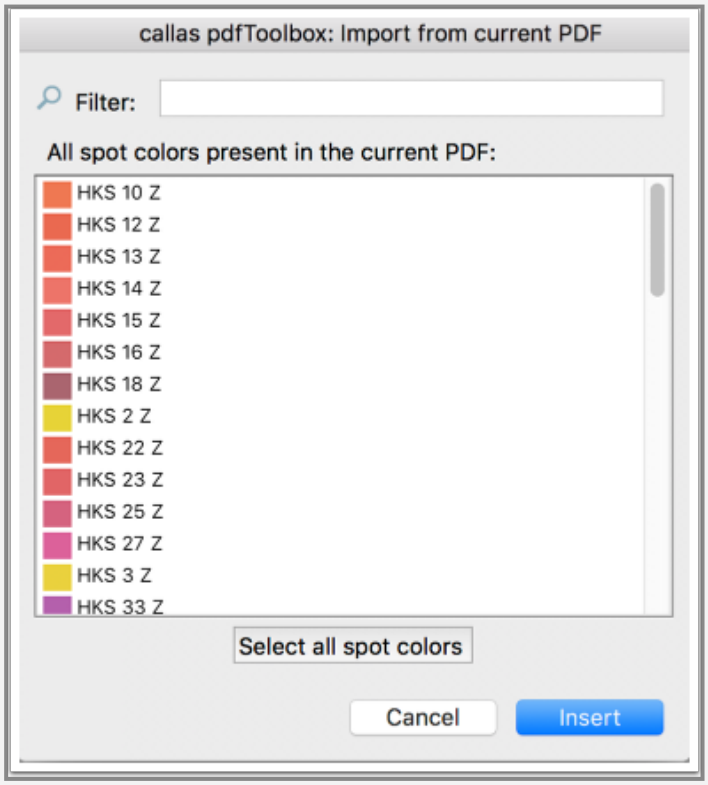
Using the Fixup 'Map spot and process colors', you can map spot colors and process colors into spot colors. Also spot colors can be modified or converted into CMYK. While doing so, you can import a color 'Swatch' file using the Import button in the bottom right corner.



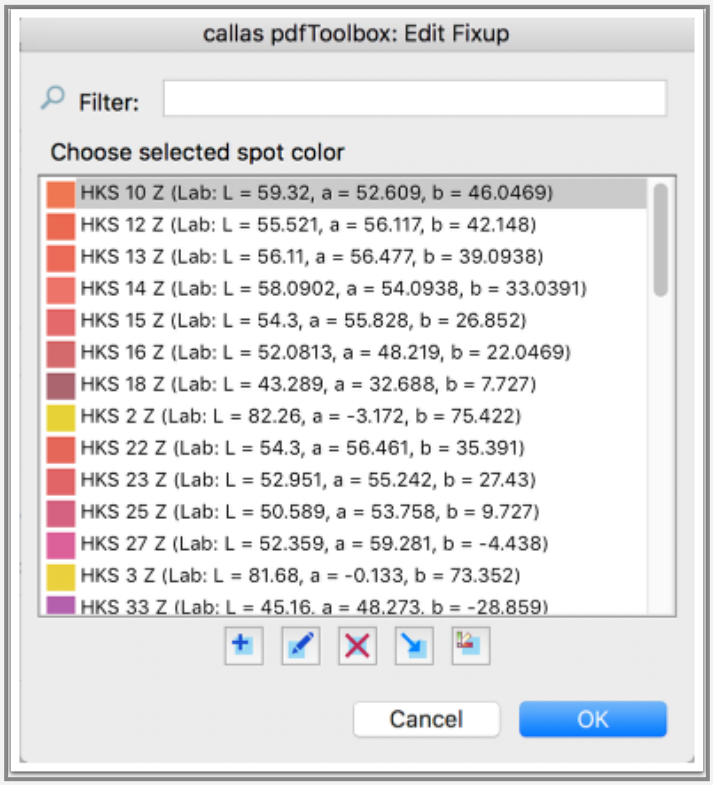
Click the rightmost button in order to import spot colors from standard spot color libraries.



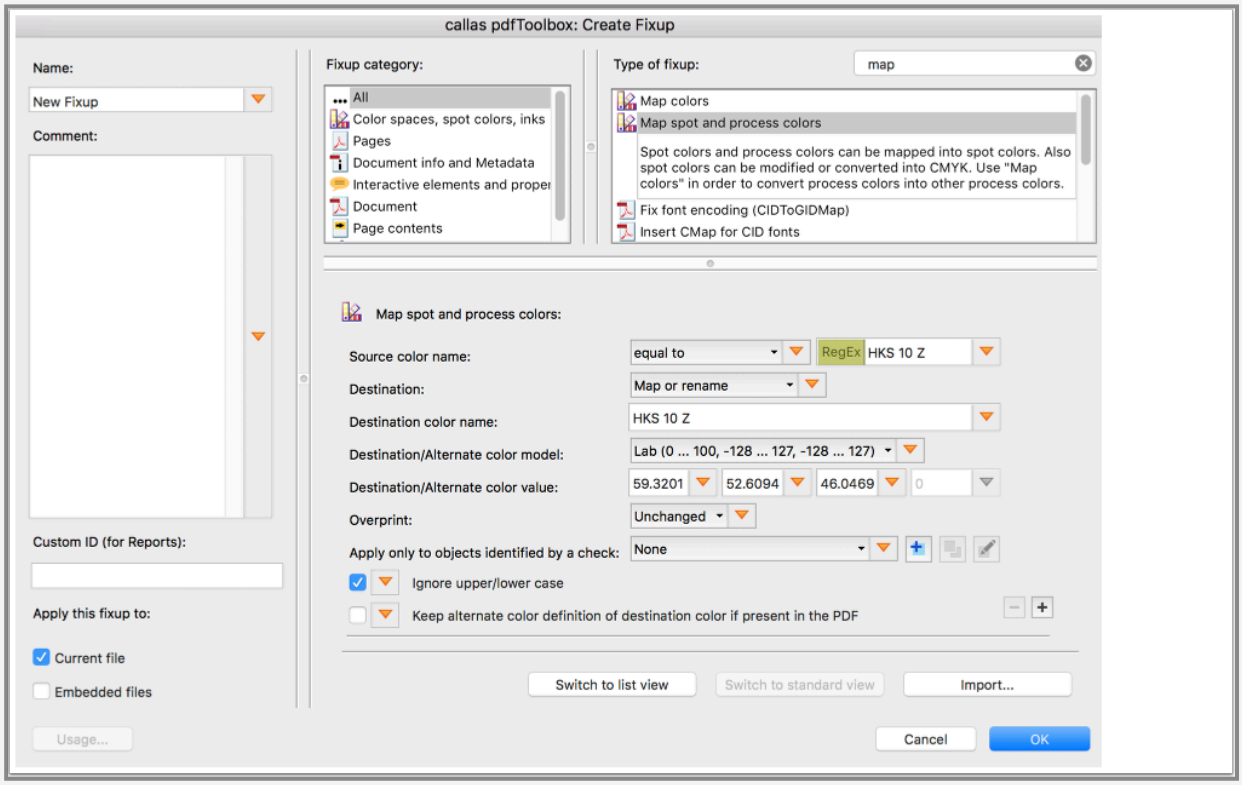
Next, you can browse and select an ASE file. A file with the ASE file extension is an Adobe Swatch Exchange file used for saving a collection of colors accessed through the Swatches palette of some Adobe products like Photoshop. The format makes it easy to share colors between programs.



After importing the ASE file, you can insert the spot colors present in the current PDF file to the fixup.



Select one or more spot colors from the 'Edit Fixup' window and click OK. Note that the colors from the ASE file will be added to an internal spot color library.



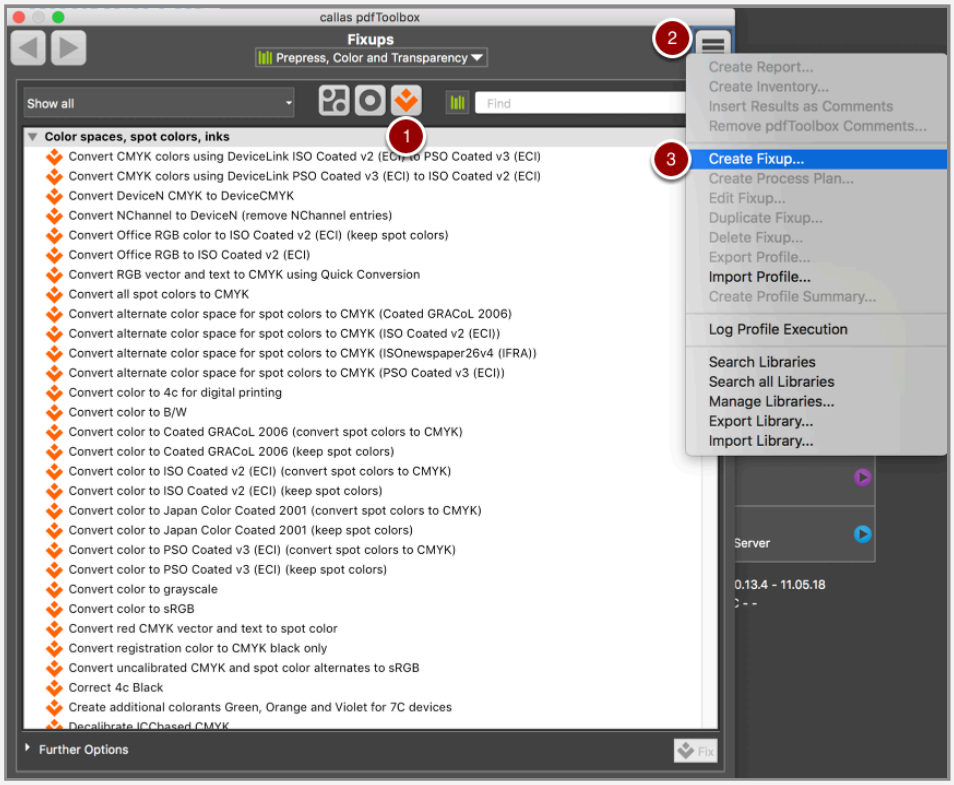
Voila! The color has been shared to the New Fixup using the Swatch file. Select the colors to be added to the fixup from the list.

7.15 Internal spot color library

pdfToolbox allows you to create a custom spot color library that stores spot colors and your alternate color space definitions.

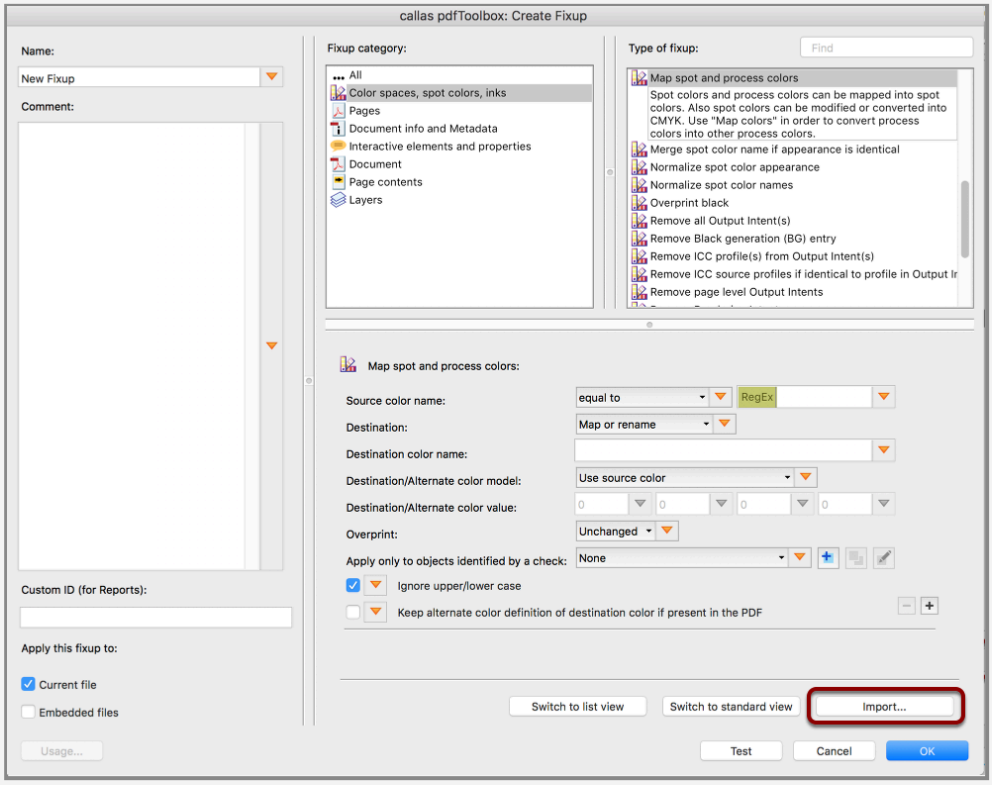
These entries can then be used in the corrections "Map colors" and "Map spot and process colors" to quickly and easily use the previously defined color values in the corrections for configuration.

Executing Fixups and creating new Fixups



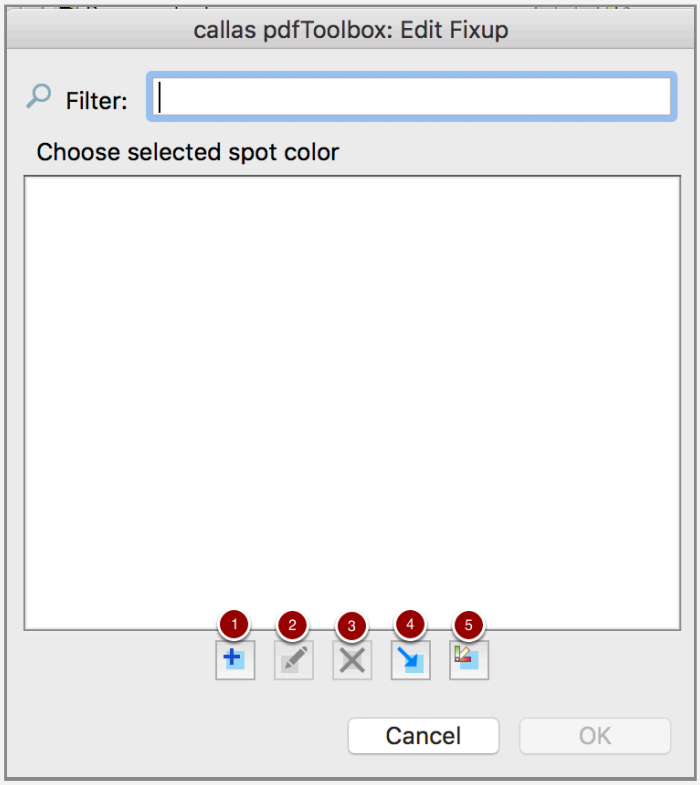
1. Select Fixups
2. Click the options menu
3. "Select "Create Fixup...".

Select Fixup type: "Map spot colors and process colors"



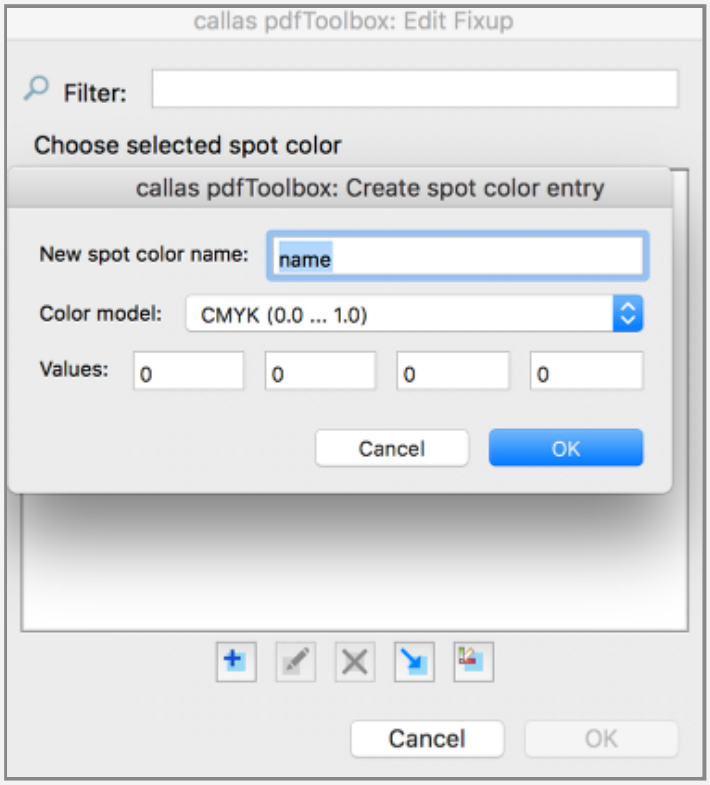
The search field in the upper right corner can also be used to quickly find the Fixup type, e.g. by entering "Spot color".

Click on "Import" to open a new dialog for selecting spot colors from the library.



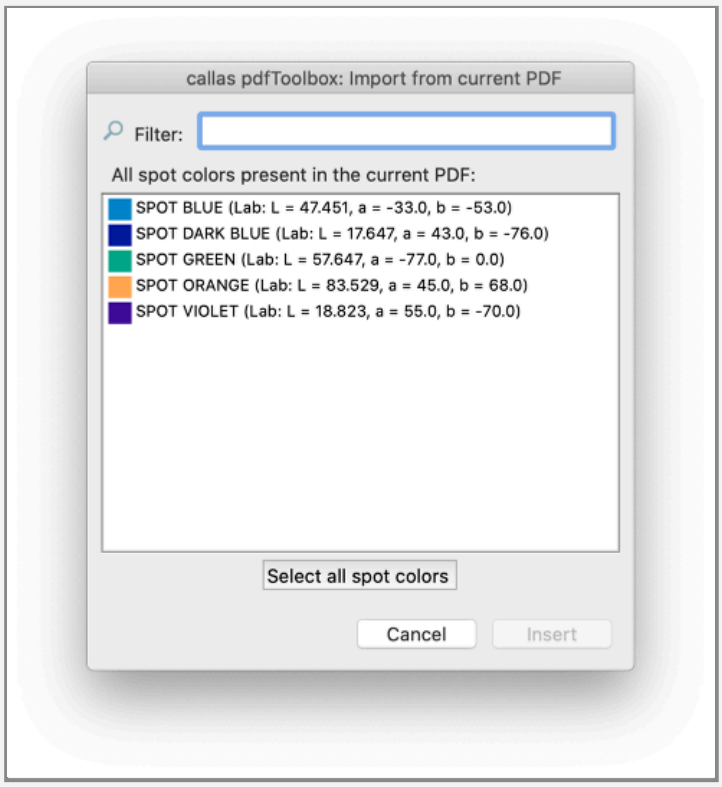
1. Create a new spot color entry
2. Edit selected spot color entry
3. Delete selected spot color entries
4. Import spot colors from the currently opened PDF file
5. Import spot colors from an ASE file (these can be created e.g. by Adobe InDesign and Illustrator)

Create a new spot color entry

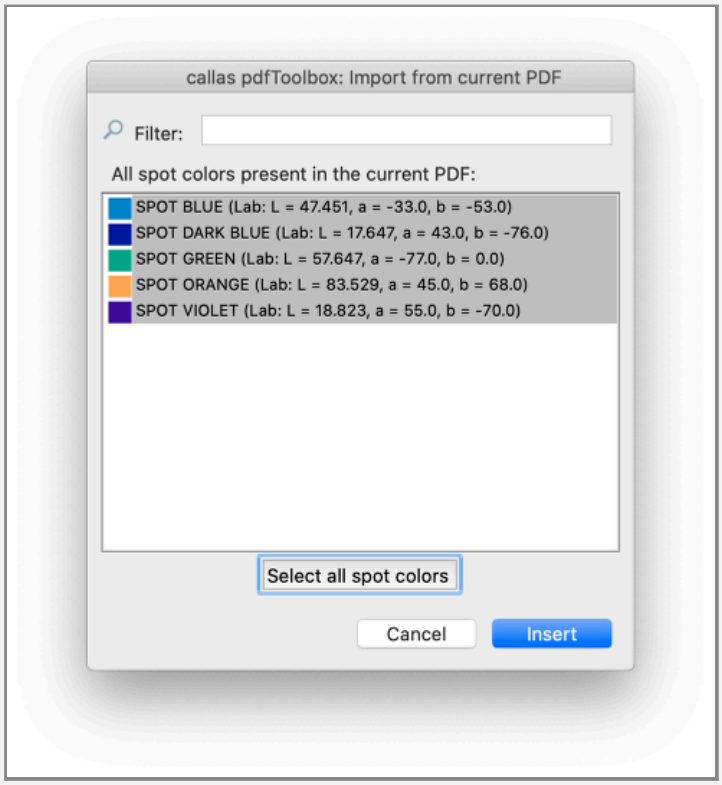


To manually create a spot color, the name, the color model of the alternative color space, and the corresponding color values must be defined.

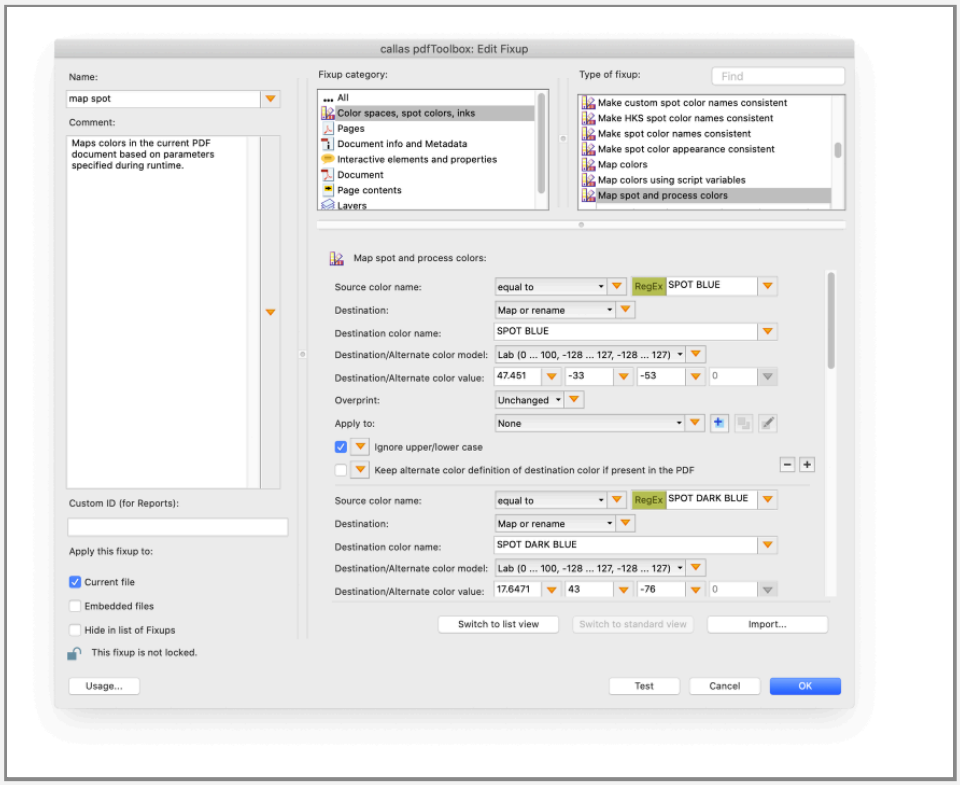
Import spot colors from the currently opened PDF file



If a PDF is open, the included spot colors are displayed and can be selected individually or entirely. These spot color definitions are then transferred to the library.



The spot colors can then be selected individually or entirely by the library and transferred to the Fixup for use.

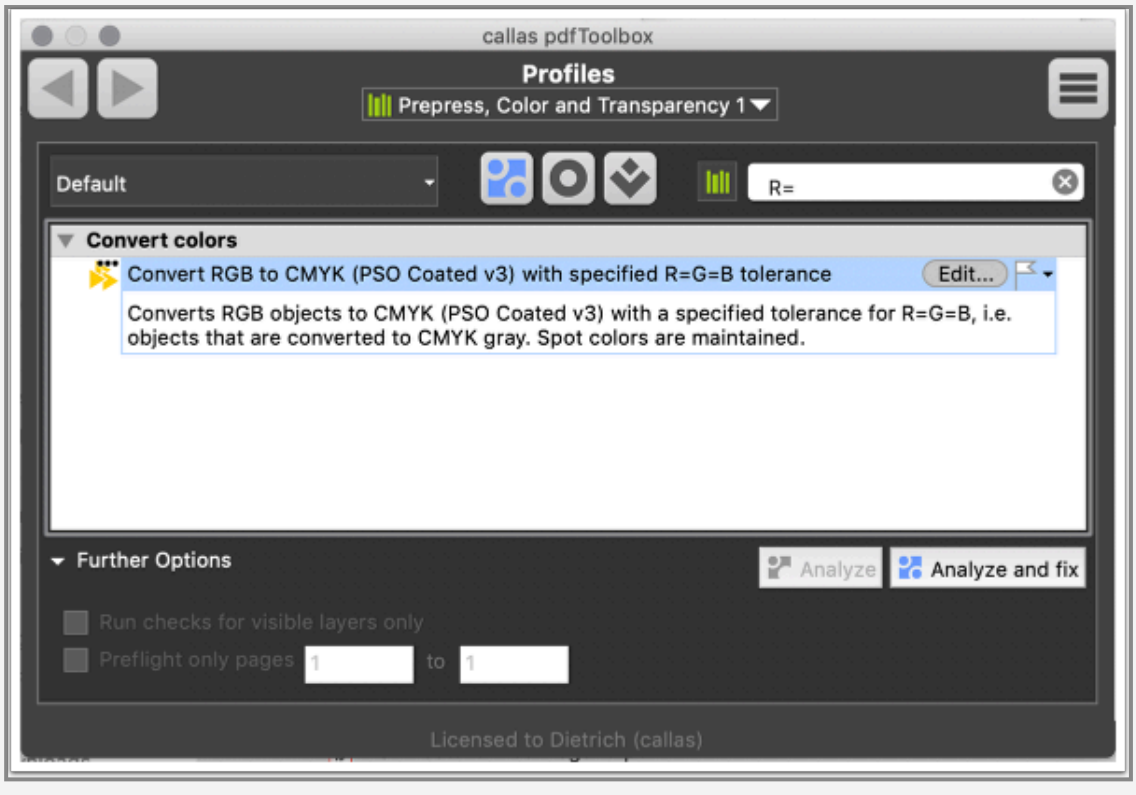


The imported spot color values are adopted and can now be further adjusted or used.

7.16 Convert RGB to CMYK using custom tolerance for gray (v11.0)

In RGB gray is indicated by same (or similar) color values for all three colorants. The tolerance is important when you convert RGB to CMYK, at least when you want to keep gray so that RGB gray would only use the K colorant (or DeviceGray or Separation Gray depending on the other settings in the Convert colors Fixup). This is controlled via the "Preserve black objects" checkbox in Convert colors.

The internal tolerance in Convert colors for that is 2%. But pdfToolbox 11 introduces a new Process Plan "Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance" which allows you to modify this tolerance.



Via a variable you can specify the tolerance.

How it works

The first step uses the Convert colors Fixup with a filter Check. The filter uses the "Difference between colorant channels" property with a variable that defines the maximum difference between the R,G and B colorant to be matched by the filter. All such colors are converted to gray (Dot Gain 15%).

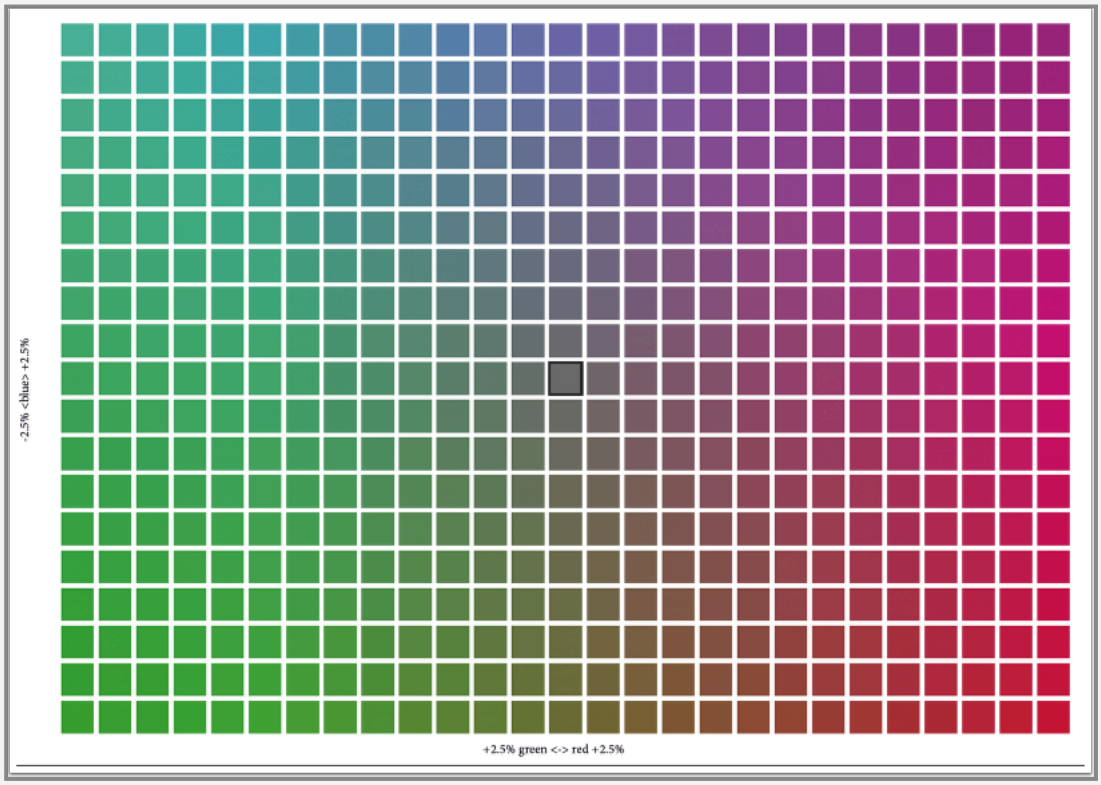
In the second step the remaining colors are converted to CMYK without preserving black objects. that means the tolerance can be either above or below 2% since what colors are considered to be gray is determined in the first step.

Example

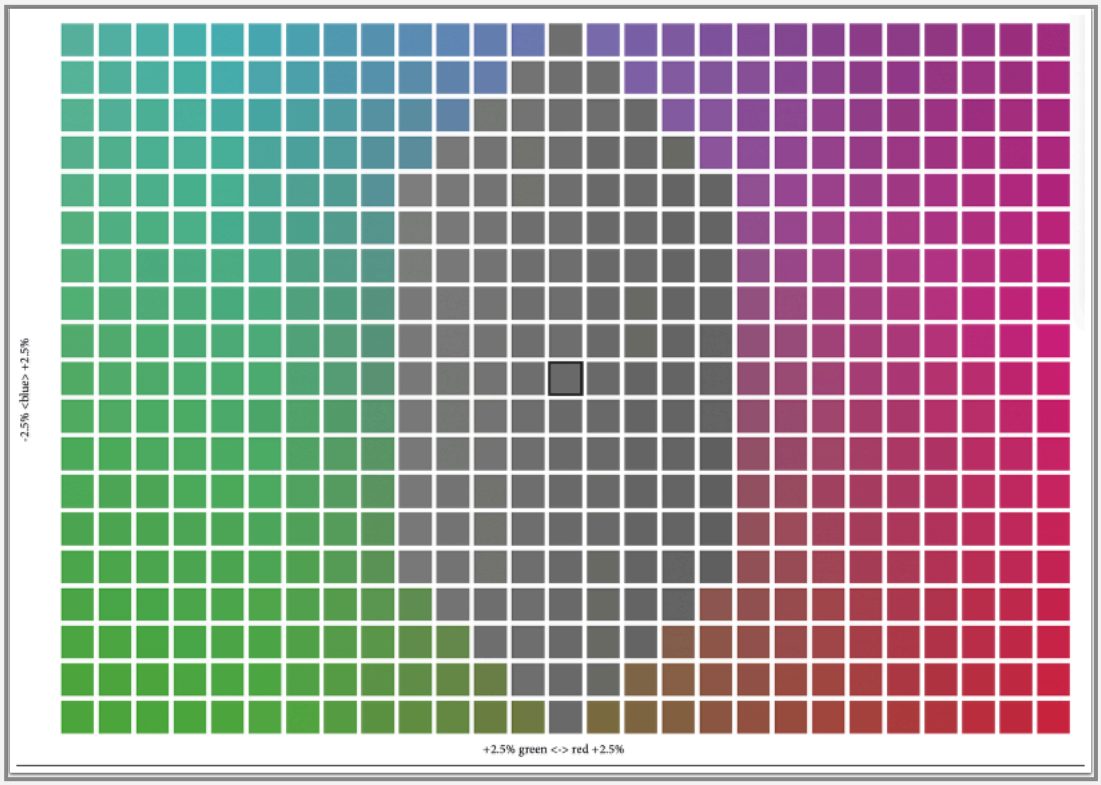
This test chart has a R=G=B=50% patch in the center (indicated with a black stroke) and color shifts of 2.5% towards the outer edges. This can be used to visualize the effects of the Process Plan.



RGB_Testchart_box2pt.pdf



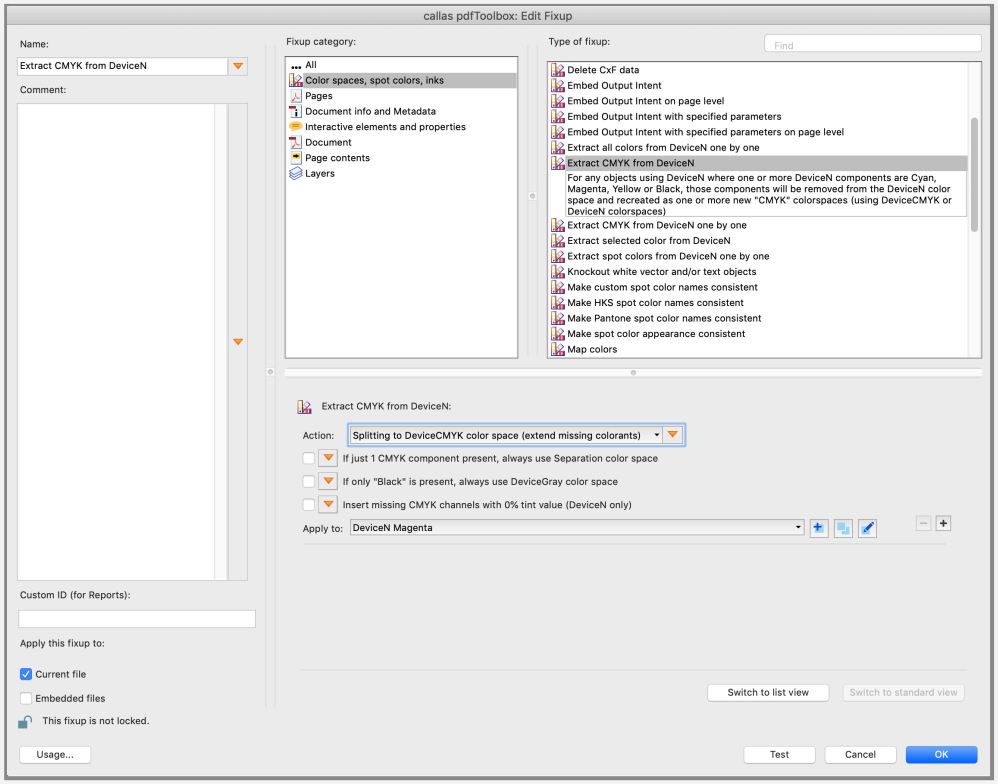
When you apply the Process Plan with a tolerance value of 25% the result looks like this.



Watch Dietrich von Seggern talk about RGB to Gray conversion in the video below:

7.17 Extract CMYK from DeviceN

This Fixup will take any CMYK colorants in the DeviceN color space and put them into their own color space, using DeviceCMYK, DeviceN or Separation (depending on situation/as configured).



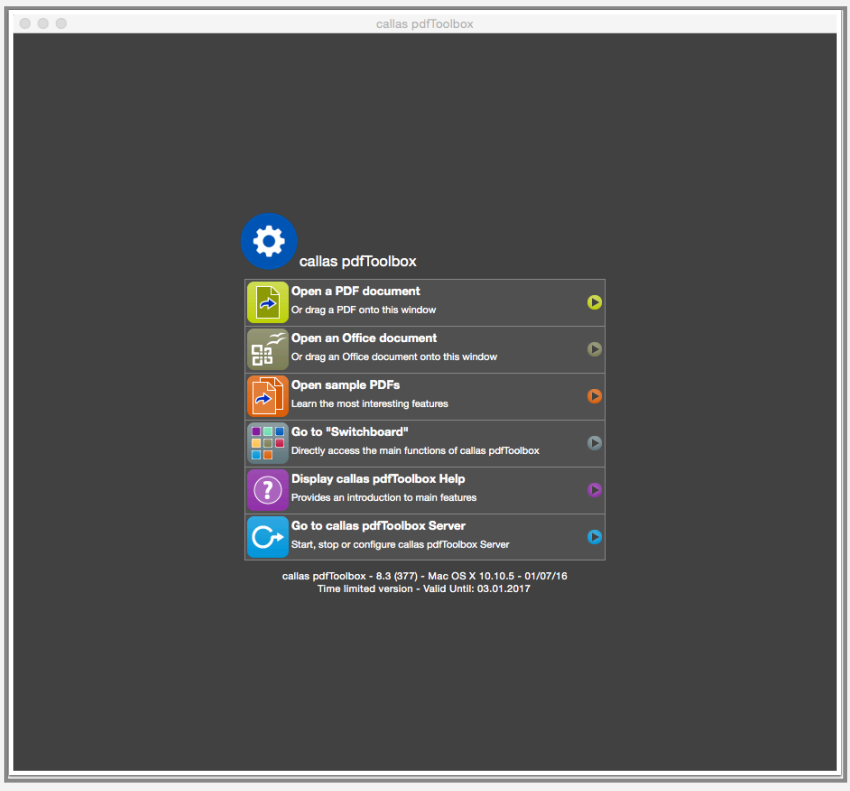
8. PDF/X-5 and n-channel color

8.1 Color convert with n-channel ICC profiles

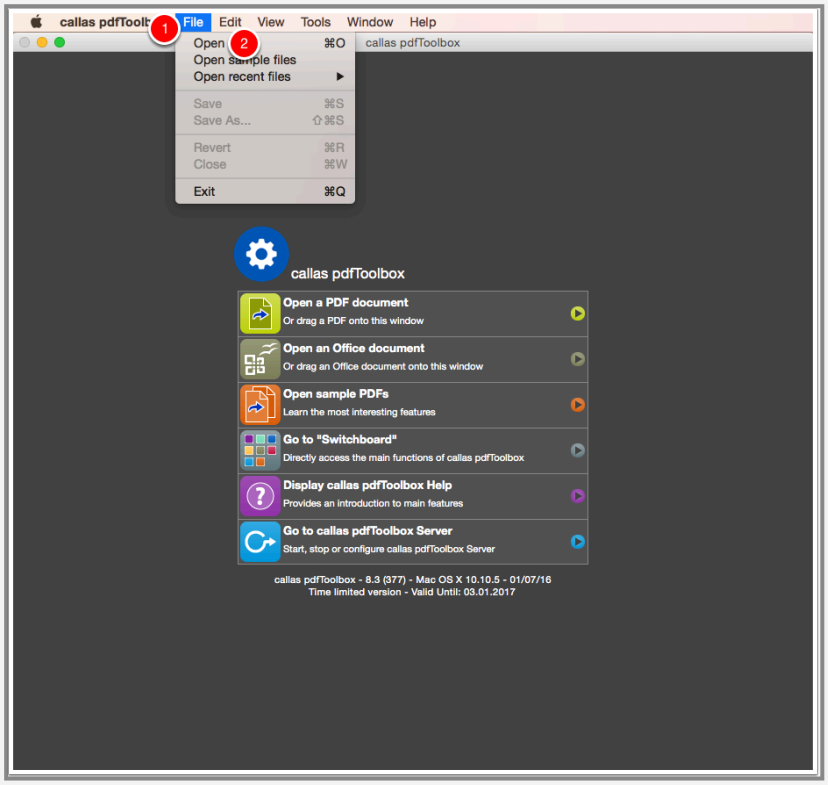
Multicolor spaces are allowed by the additional use of printing inks such as Orange, Green and Violet to get brilliant color results.

pdfToolbox has the possibility to convert a PDF file into a multicolor space by using an ICC profile. For example RGB and CMYK content convert to hexachrome, CMYK + Red or any other multicolor scenarios.

In the pdfToolbox Desktop Window

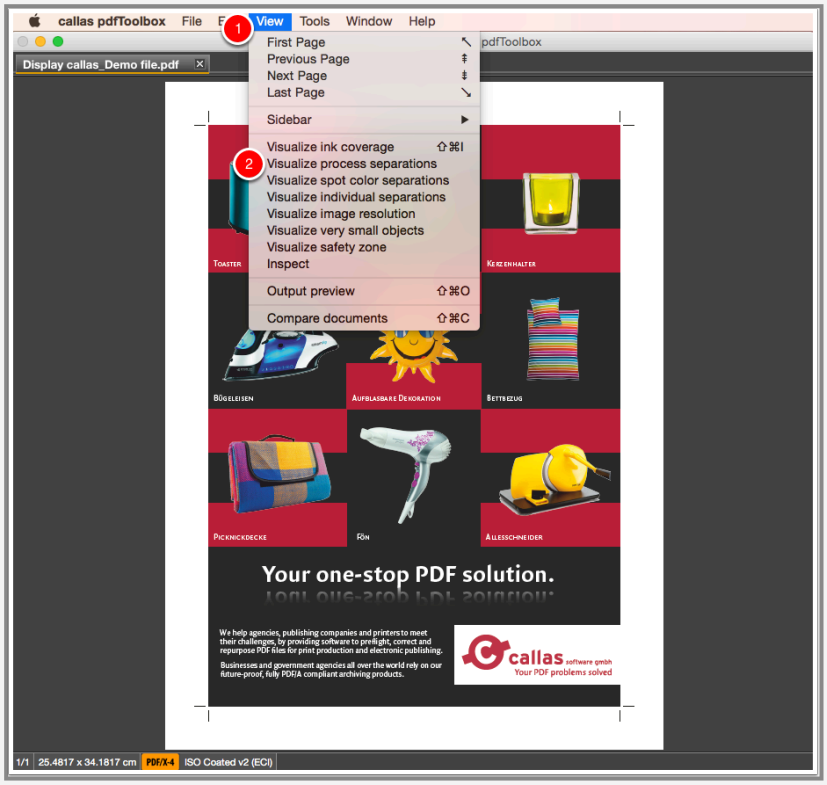


Open the PDF file "Display callas_Demo file.pdf"



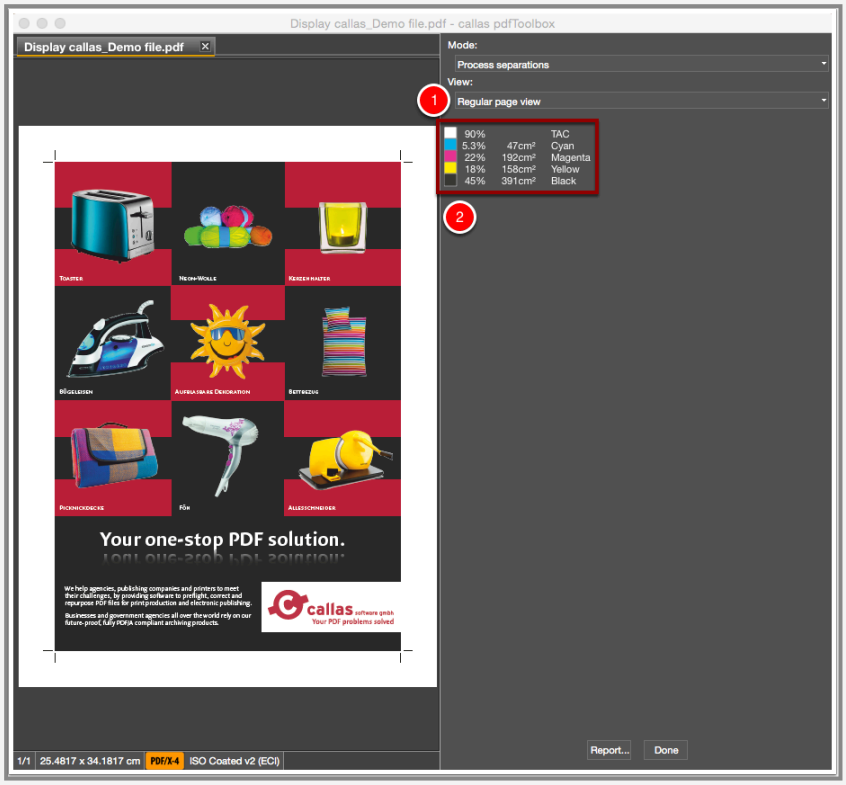
1. Go to "File".
2. Click "Open" and browse to the file "Display callas_Demo file.pdf".

Open the "Process separations" view mode



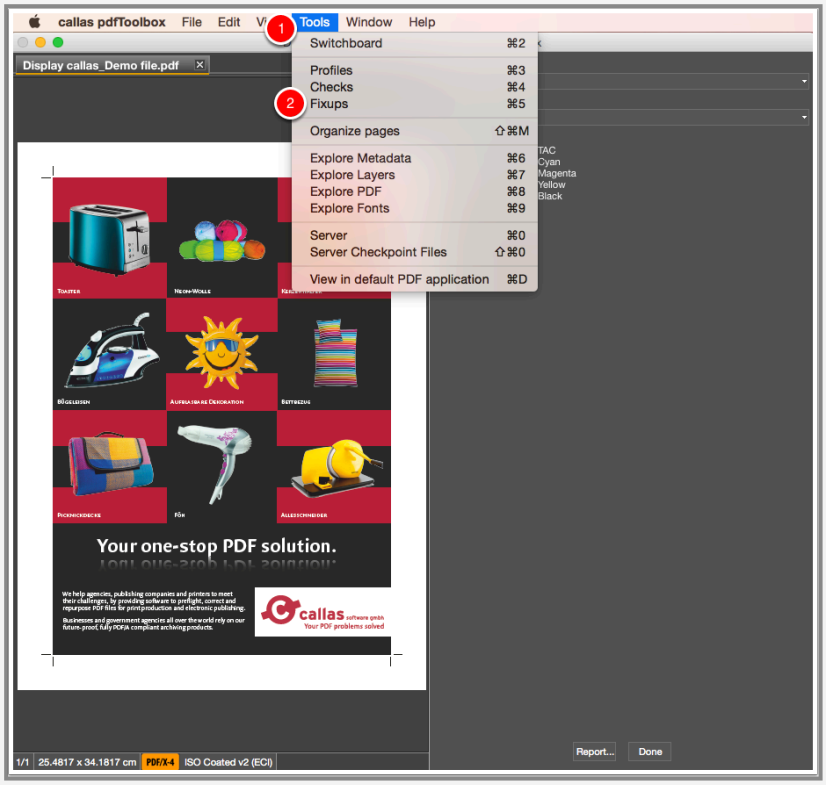
1. Go to "View".
2. Click "Visualize process separations".

Analyze the PDF file



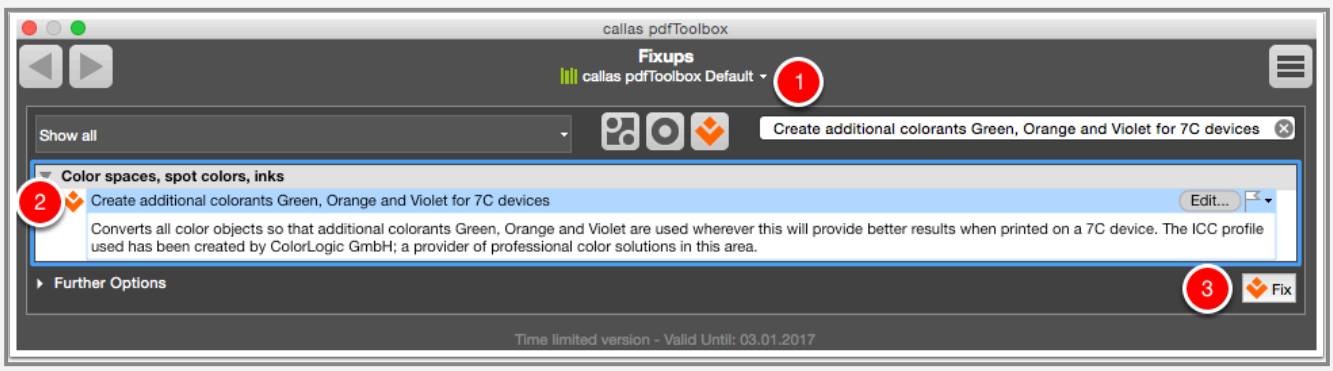
1. Select "Regular page view".
2. The PDF has a four color space (CMYK).

Open the Fixups dialog



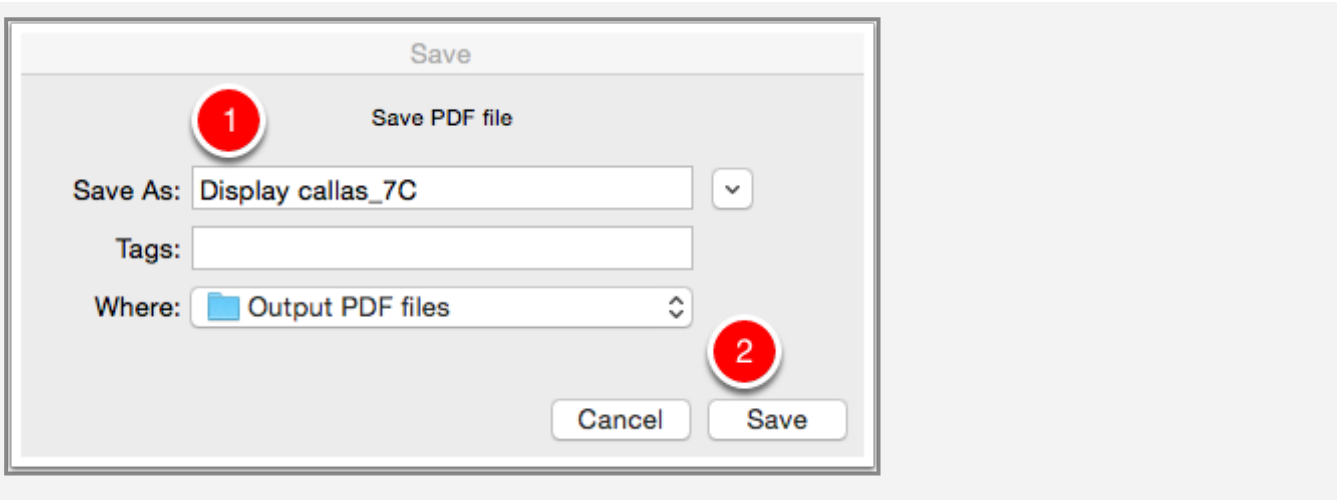
1. Go to "Tools" and pull down the menu.
2. Select "Fixups" to display the Fixup menu.

Apply the Fixup "Create additional colorants Green, Orange and Violet for 7C devices"



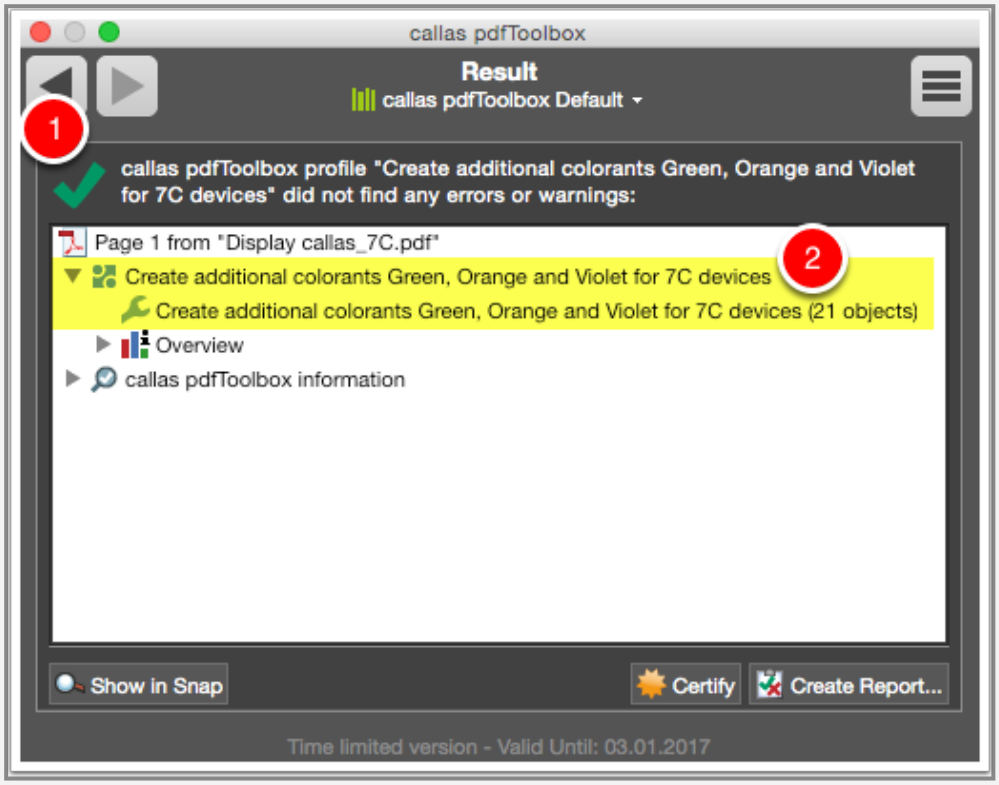
1. In the search field, search to "Create additional colorants Green, Orange and Violet for 7C devices".
2. Select the Fixup "Create additional colorants Green, Orange and Violet for 7C devices".
3. Click "Fix".

Save the output PDF file



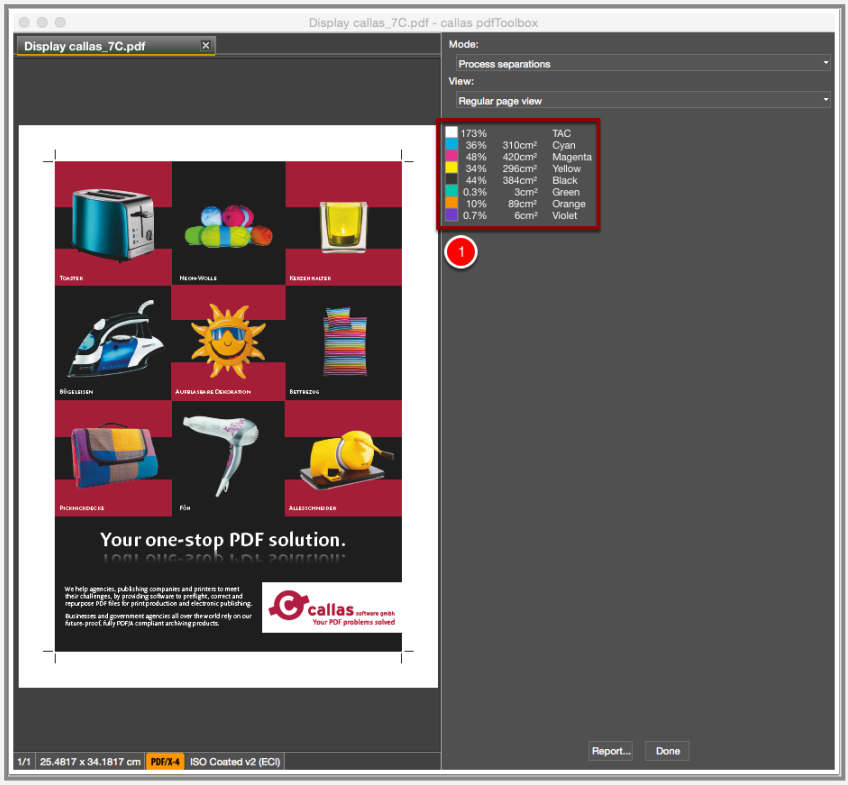
1. Save the output PDF file as "Display callas_7C".
2. Click "Save".

Review the preflight report



1. A green check-mark is displayed, indicating that no problems are found.
2. Descriptive text that details what the profile performs upon completion.

Analyze the processed PDF file



1. Note: The additional channels "Green", "Orange" and "Violet" have been added, in addition to the original process colors "CMYK".

8.2 Convert to PDF/X-5n

pdfToolbox is able to convert and validate PDF files to the PDF/X-5n ISO standard.

pdfToolbox has the possibility to convert a PDF file into a multicolor space by using an ICC profile. For example RGB and CMYK content convert to hexachrome, CMYK + Red or any other multicolor scenarios.

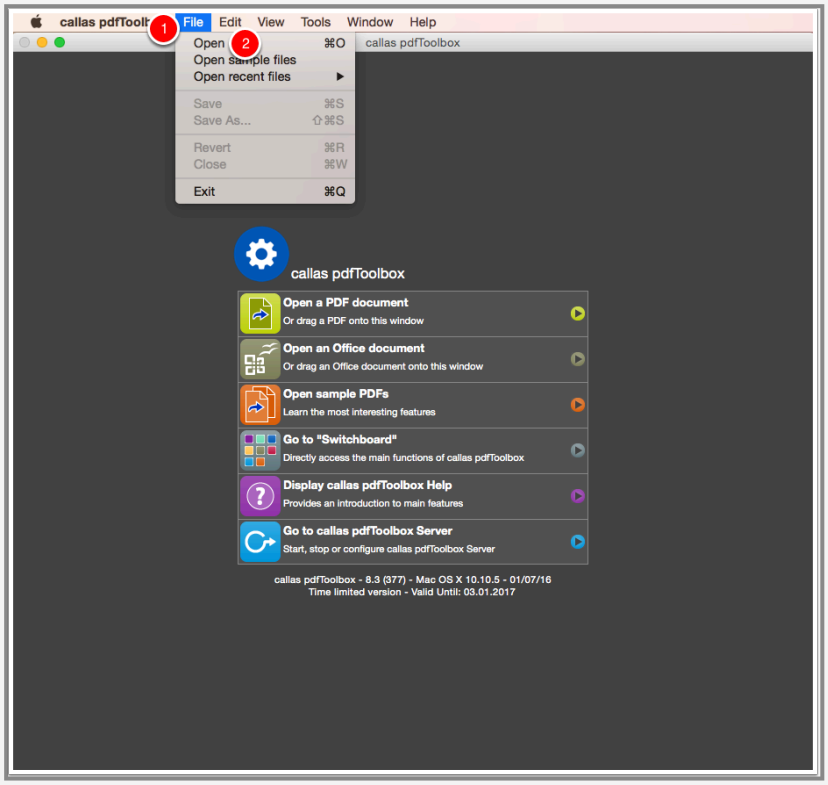
This article shows how to use a default 7-color profile and setup pdfToolbox to use your own multi-color profiles.

Right below, the demo file "Display callas_Demo file.pdf" used in this article can be dowbloaded.



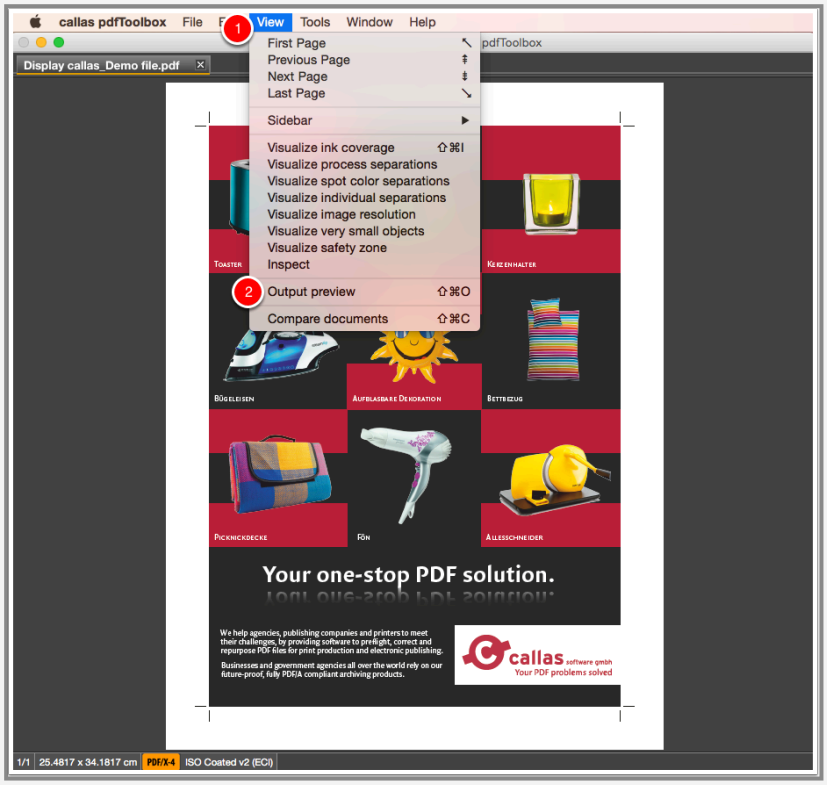
Display_callas_Demo_file.pdf

Open the PDF file "Display callas_Demo file.pdf"



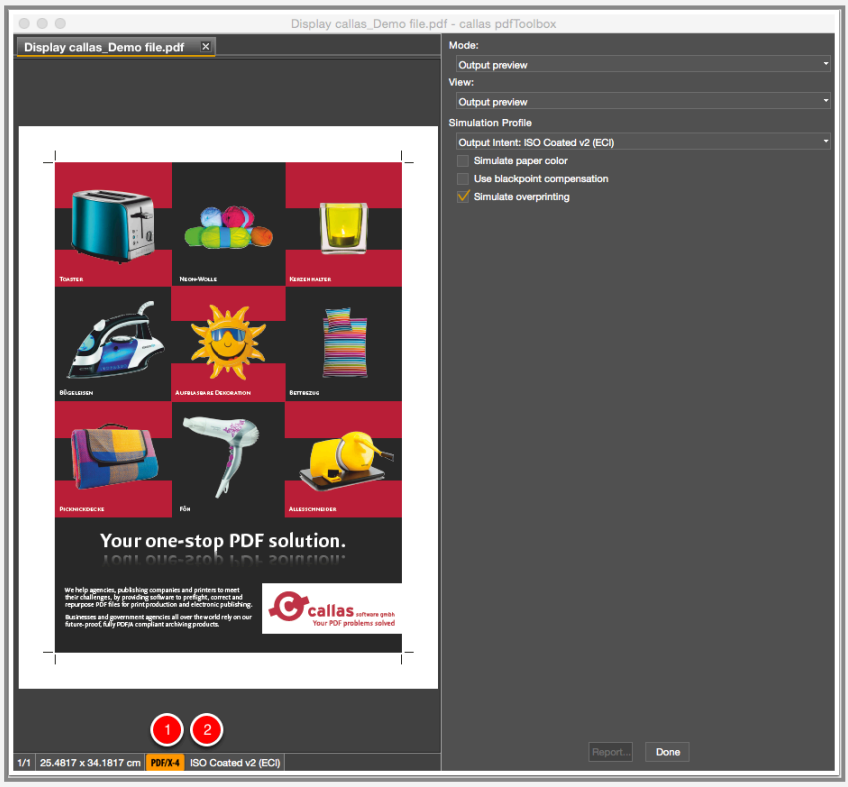
1. Go to "File".
2. Click "Open".

Open the Output preview mode



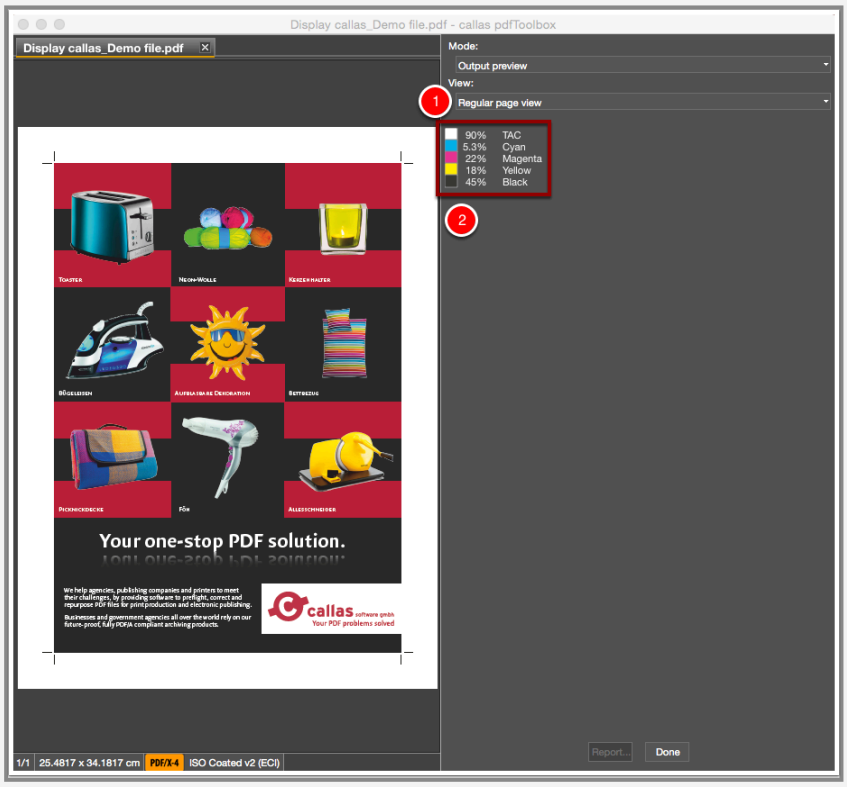
1. Go to "View".
2. Click "Output preview".

Analyze the Output preview



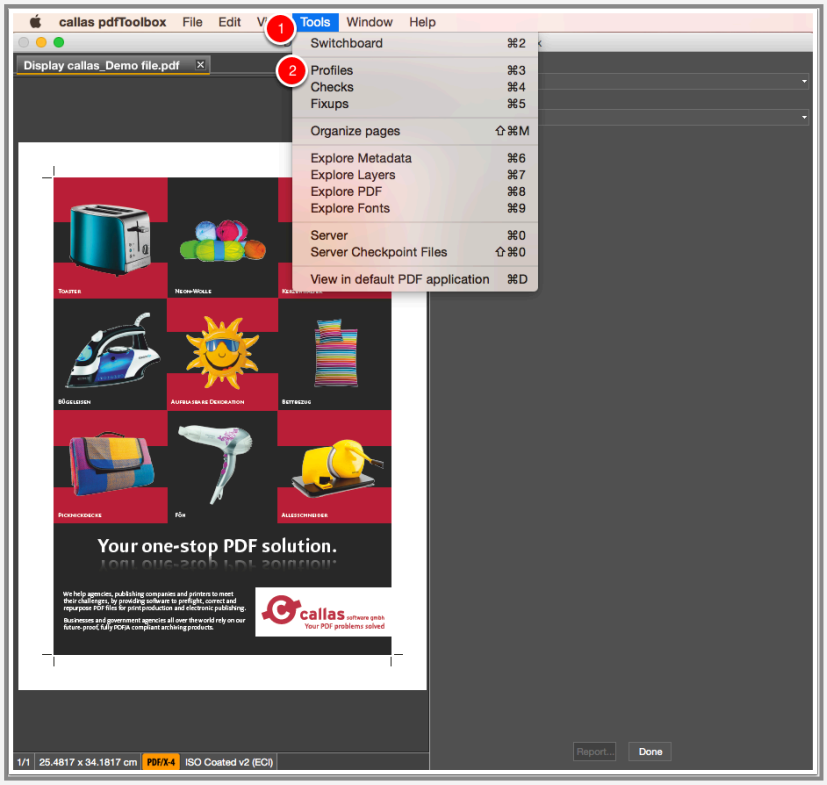
1. The PDF is identified as a PDF/X-4 standard.
2. The PDF has the output intent "ISO Coated v2 (ECI).

Analyze the Regular page view



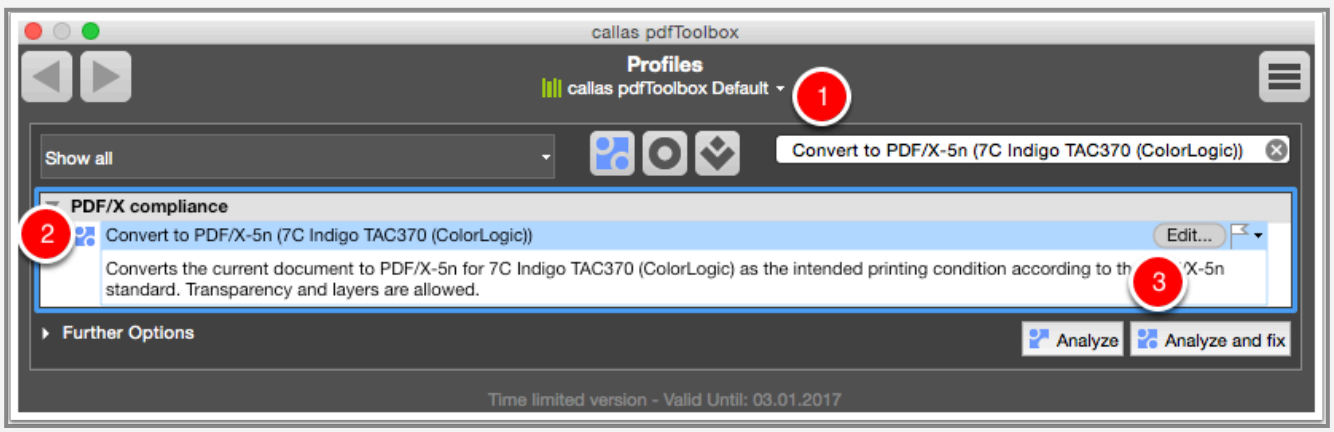
1. Select "Regular page view".
2. The PDF has a four color space (CMYK).

Open the Profile dialog



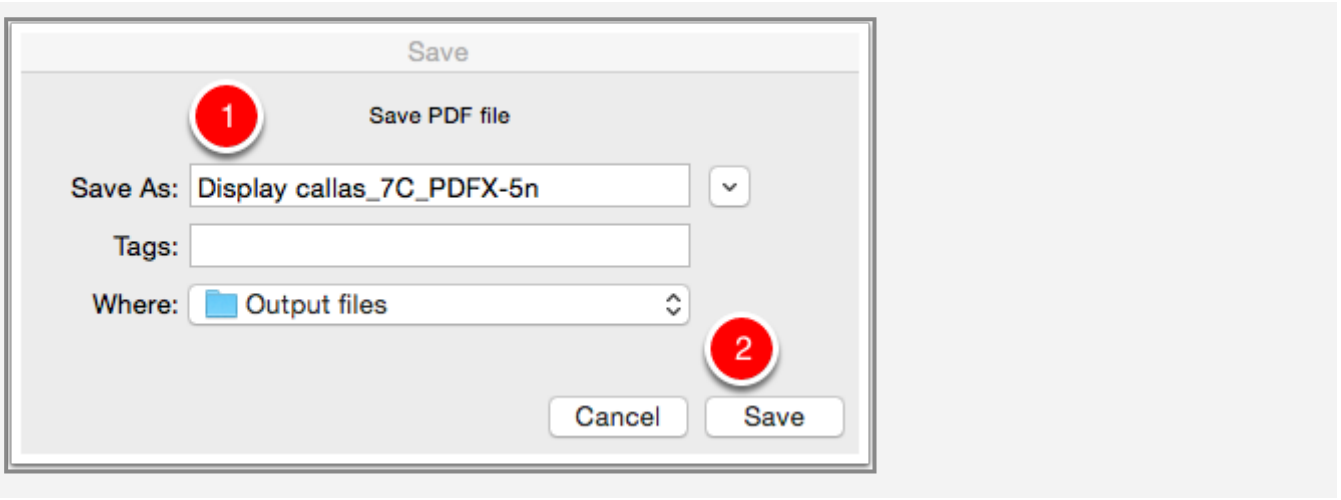
1. Go to "Tools".
2. Click "Profiles".

Apply the Profile "Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))"



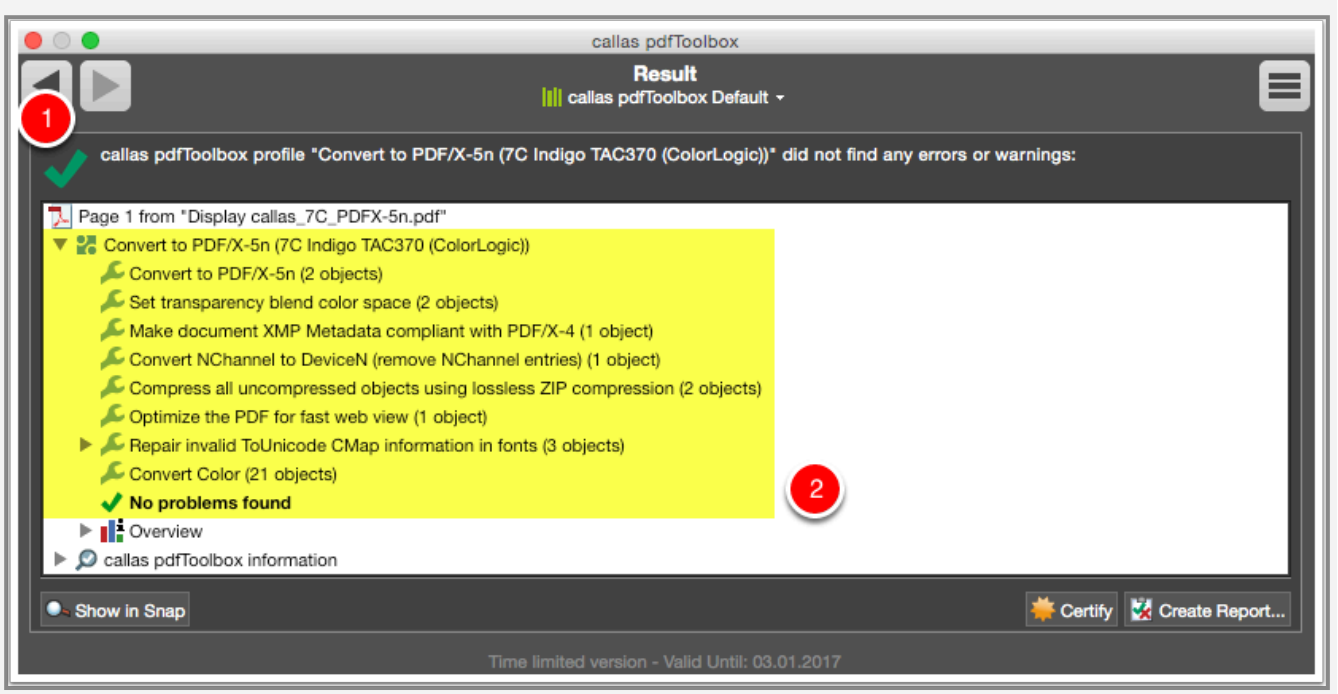
1. In the search field search to "Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))"
2. Select the Fixup "Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))".
3. Click "Analyze and fix".

Save the output PDF file



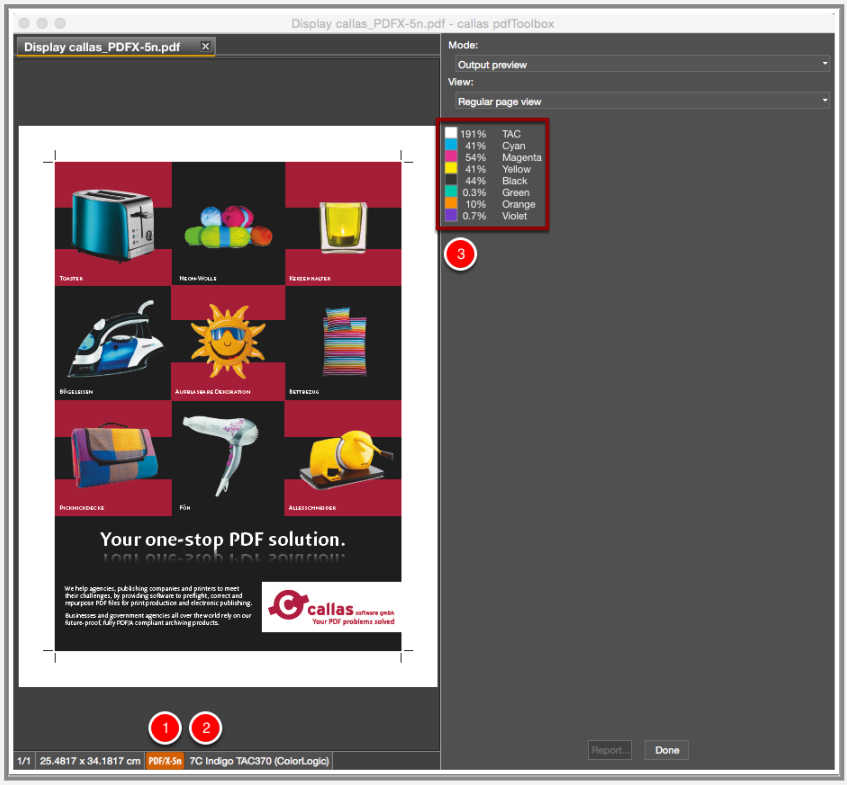
1. Save the PDF as "Display callas_7C_PDFX-5n".
2. Click "Save".

Review the preflight report



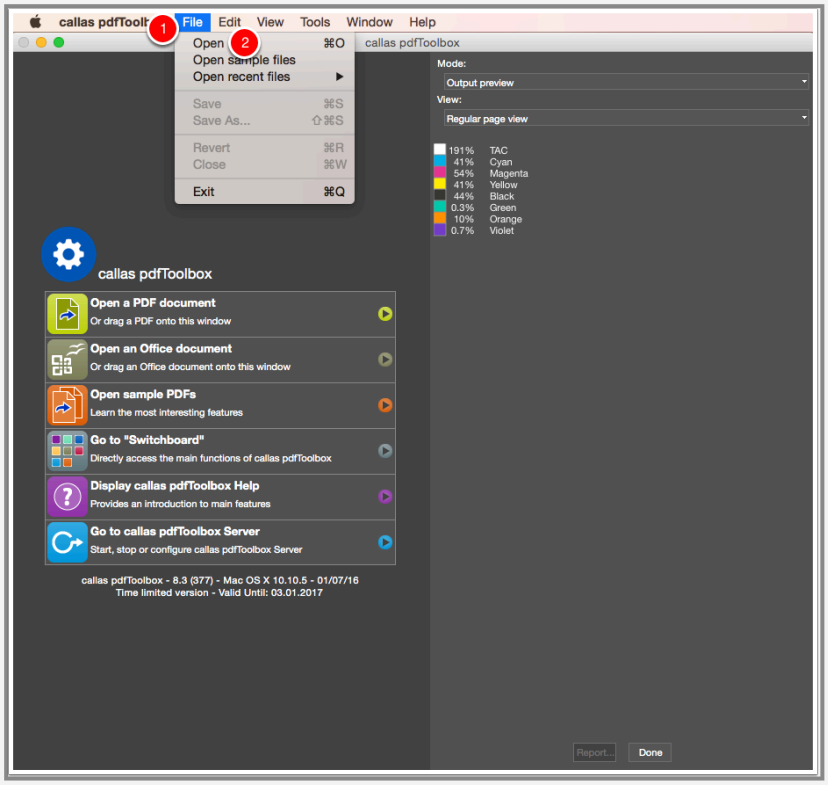
1. A green checkmark shows, no errors occurred.
2. More details what steps are carried out.

Analyze the output PDF file



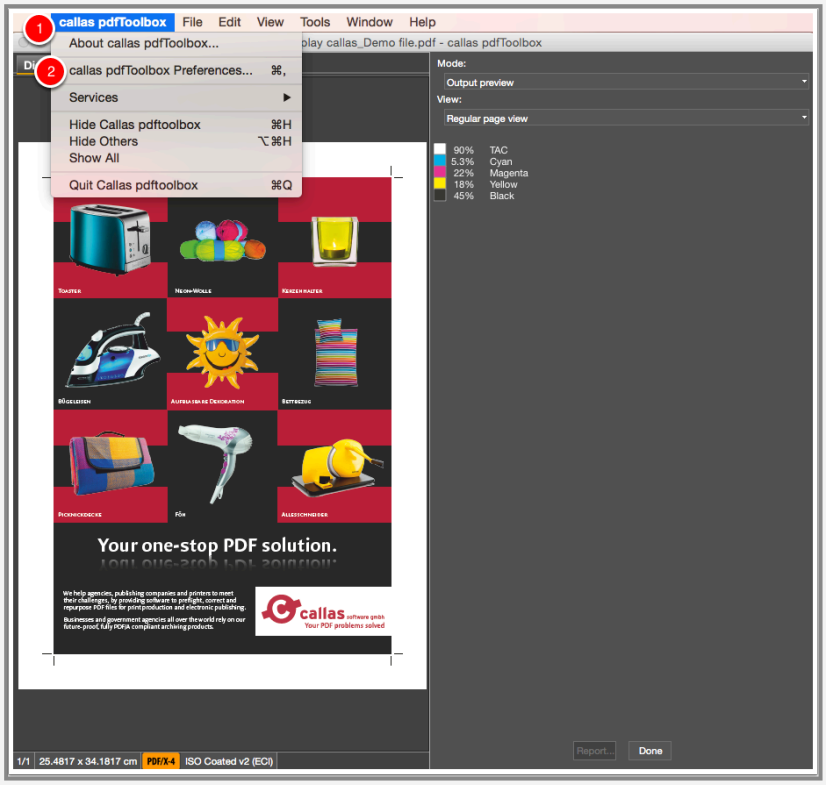
1. The PDF is identified as a PDF/X-5n standard.
2. The PDF has the output intent "7C Indigo TAC370 (Color-Logic)"
3. In the PDF three extra color separations are added: "Green"; "Orange" and "Violet". Now it is a seven color PDF file (CMYKOGV).

Open the PDF file "Display callas_Demo file.pdf"

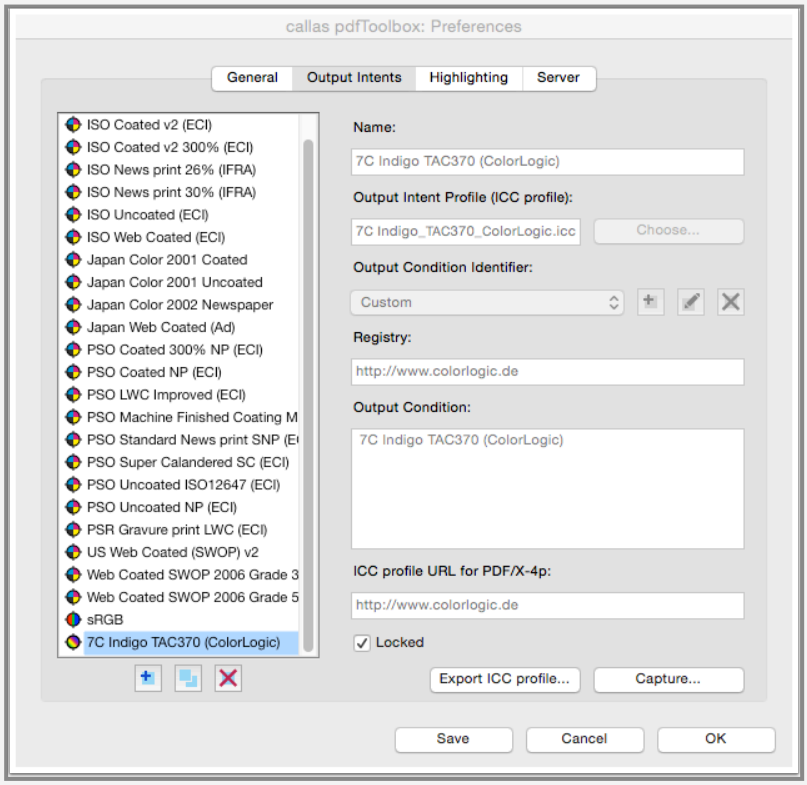


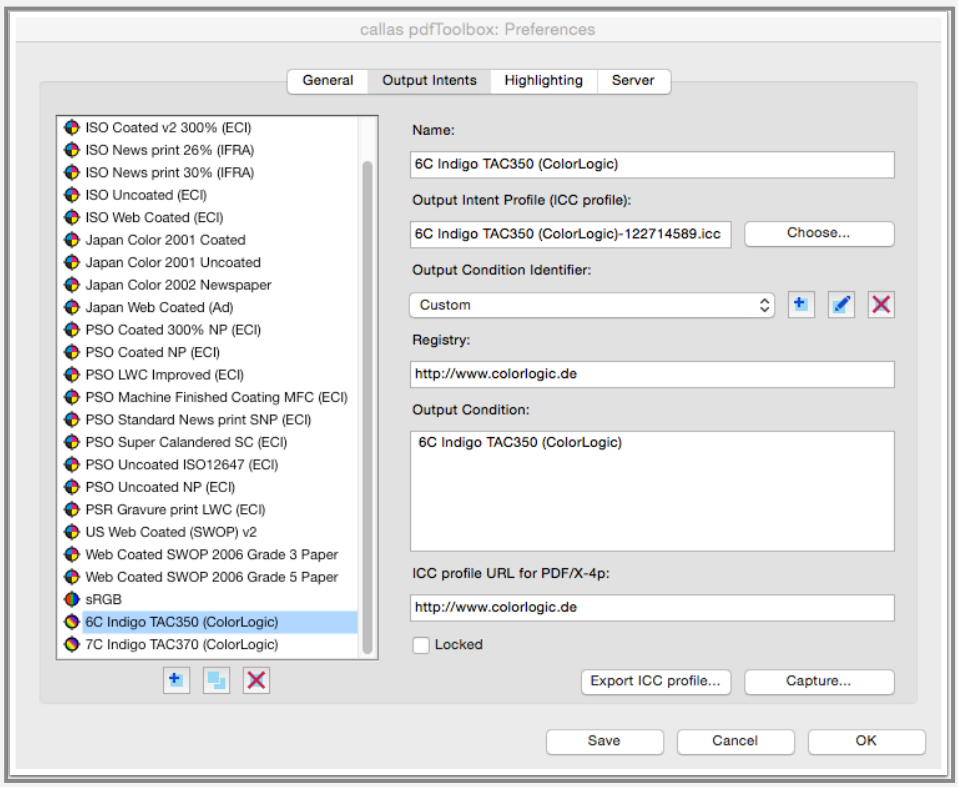
1. Go to "File".
2. Click "Open".

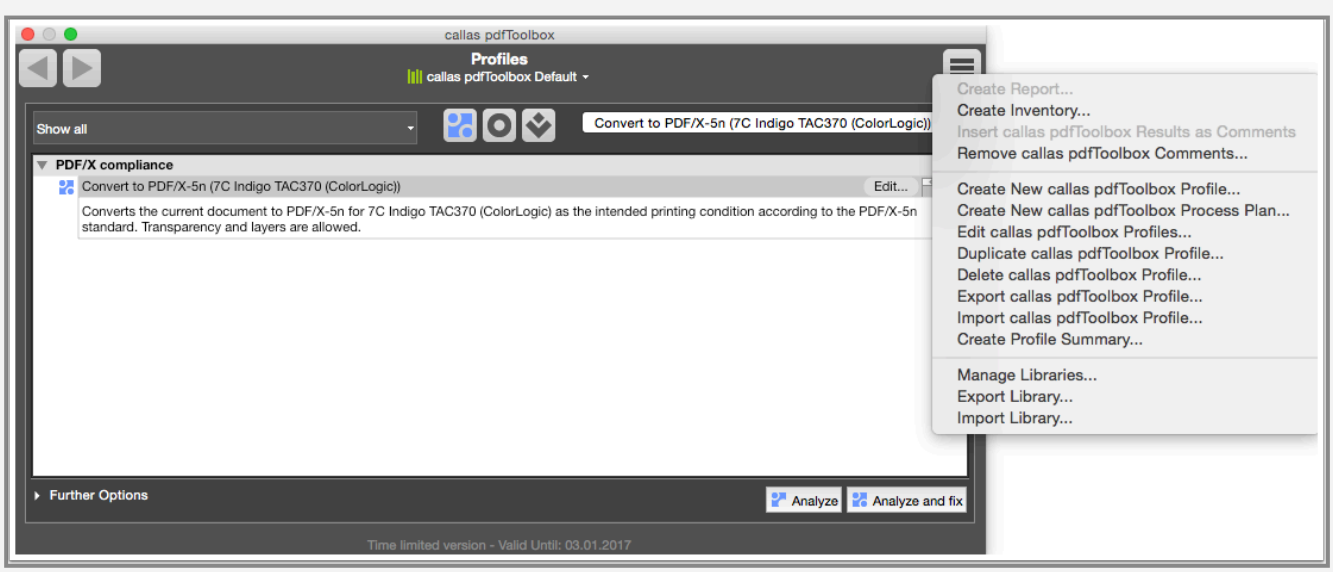
Open the callas pdfToolbox Preferences

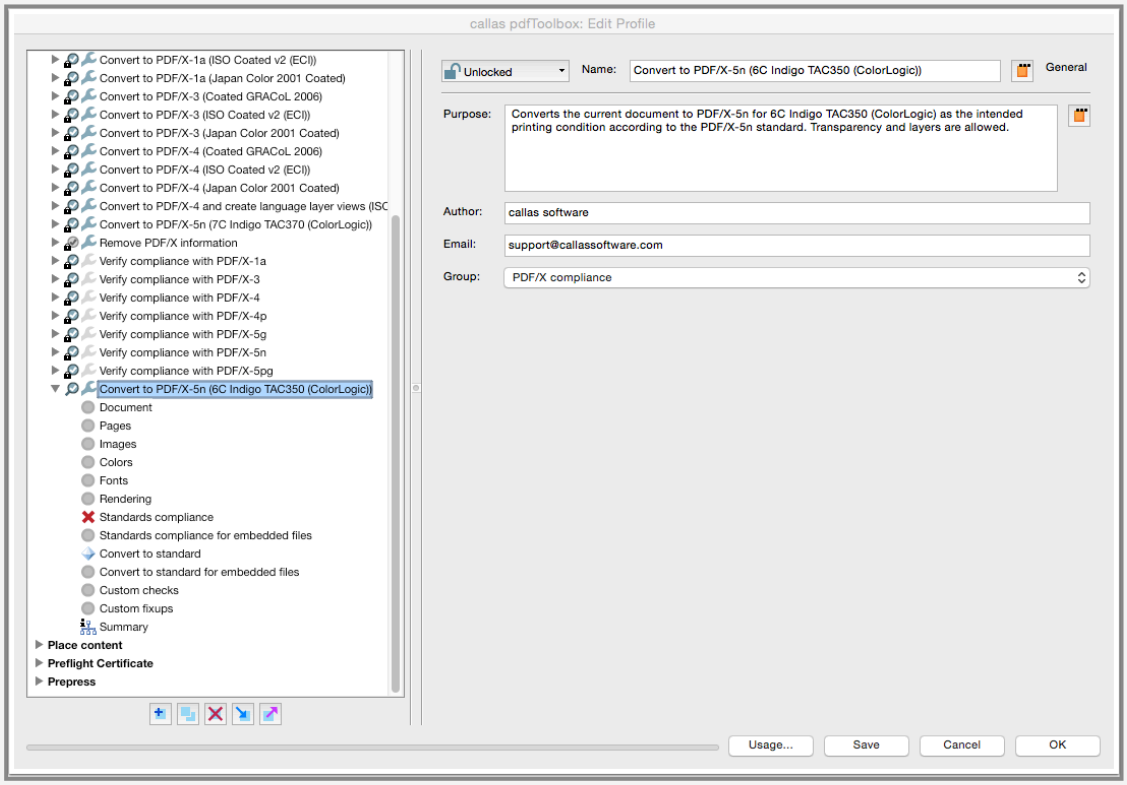


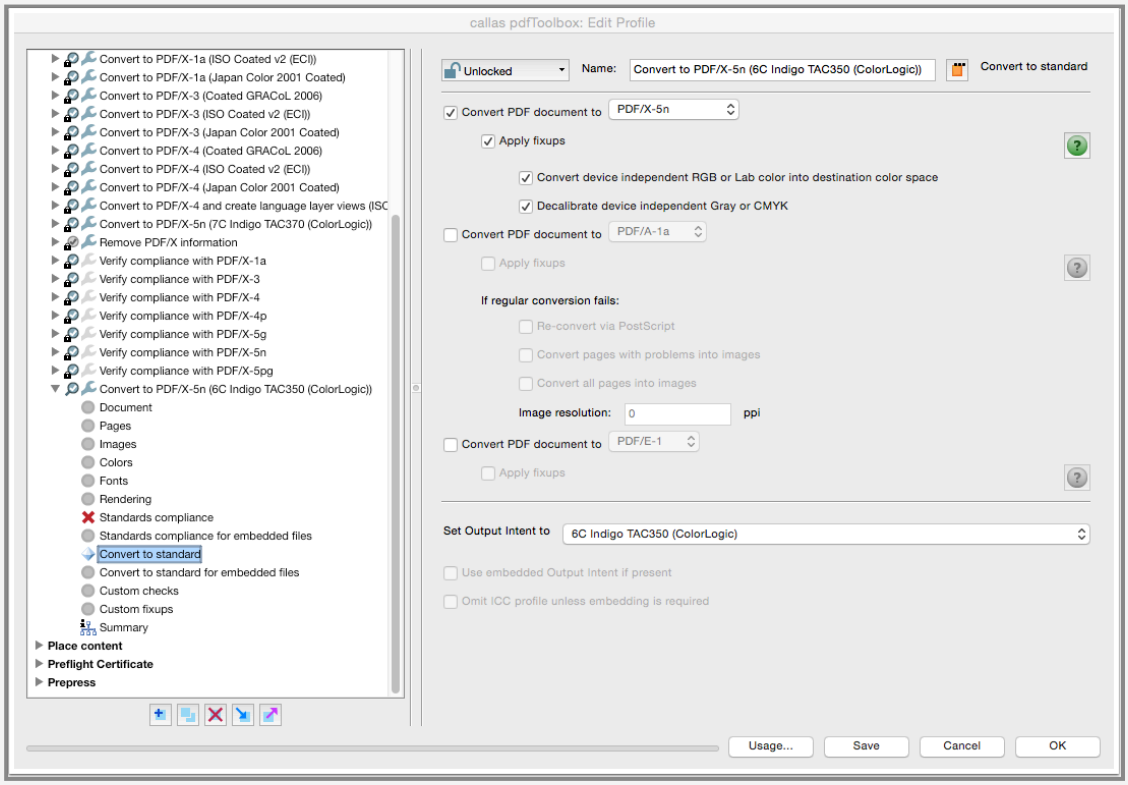
1. Go to "callas pdfToolbox".
2. Click "callas pdfToolbox Preferences".

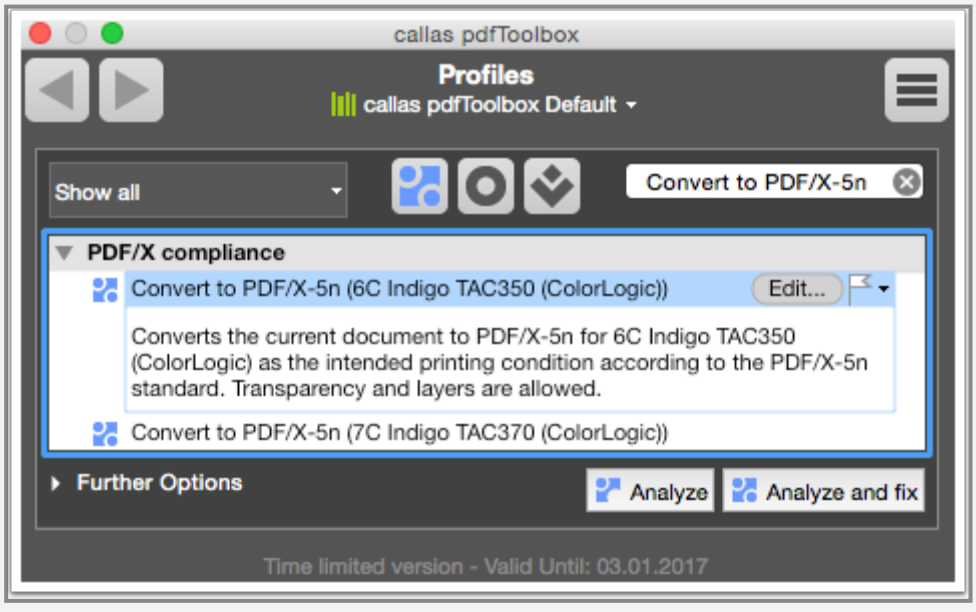

callas pdfToolbox: Preferences
General
Output Intents
Highlighting
Server
ISO Coated v2 (ECI)
ISO Coated v2 300% (ECI)
ISO News print 26% (IFRA)
ISO News print 30% (IFRA)
ISO Uncoated (ECI)
ISO Web Coated (ECI)
Japan Color 2001 Coated
Japan Color 2001 Uncoated
Japan Color 2002 Newspaper
Japan Web Coated (Ad)
PSO Coated 300% NP (ECI)
PSO Coated NP (ECI)
PSO LWC Improved (ECI)
PSO Machine Finished Coating M
PSO Standard News print SNP (E
PSO Super Calandered SC (ECI)
PSO Uncoated ISO12647 (ECI)
PSO Uncoated NP (ECI)
PSR Gravure print LWC (ECI)
US Web Coated (SWOP) v2
Web Coated SWOP 2006 Grade 3
Web Coated SWOP 2006 Grade 5
sRGB
7C Indigo TAC370 (ColorLogic)
Name:
7C Indigo TAC370 (ColorLogic)
Output Intent Profile (ICC profile):
7C Indigo_TAC370_ColorLogic.icc
Choose...
Output Condition Identifier:
Custom
Registry:
http://www.colorlogic.de
Output Condition:
7C Indigo TAC370 (ColorLogic)
ICC profile URL for PDF/X-4p:
http://www.colorlogic.de
Locked
Export ICC profile...
Capture...
Save
Cancel
OK

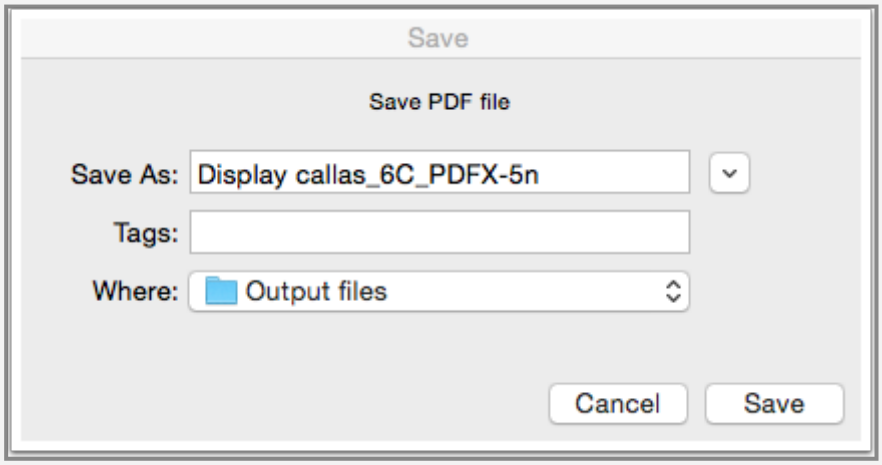

callas pdfToolbox: Preferences
General
Output Intents
Highlighting
Server
ISO Coated v2 300% (ECI)
ISO News print 26% (IFRA)
ISO News print 30% (IFRA)
ISO Uncoated (ECI)
ISO Web Coated (ECI)
Japan Color 2001 Coated
Japan Color 2001 Uncoated
Japan Color 2002 Newspaper
Japan Web Coated (Ad)
PSO Coated 300% NP (ECI)
PSO Coated NP (ECI)
PSO LWC Improved (ECI)
PSO Machine Finished Coating MFC (ECI)
PSO Standard News print SNP (ECI)
PSO Super Calandered SC (ECI)
PSO Uncoated ISO12647 (ECI)
PSO Uncoated NP (ECI)
PSR Gravure print LWC (ECI)
US Web Coated (SWOP) v2
Web Coated SWOP 2006 Grade 3 Paper
Web Coated SWOP 2006 Grade 5 Paper
sRGB
6C Indigo TAC350 (ColorLogic)
7C Indigo TAC370 (ColorLogic)
Name:
6C Indigo TAC350 (ColorLogic)
Output Intent Profile (ICC profile):
6C Indigo TAC350 (ColorLogic)-122714589.icc
Choose...
Output Condition Identifier:
Custom
Registry:
http://www.colorlogic.de
Output Condition:
6C Indigo TAC350 (ColorLogic)
ICC profile URL for PDF/X-4p:
http://www.colorlogic.de
Locked
Export ICC profile...
Capture...
Save
Cancel
OK


callas pdfToolbox
Profiles
callas pdfToolbox Default
Show all
Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))
PDF/X compliance
Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))
Edit...
Converts the current document to PDF/X-5n for 7C Indigo TAC370 (ColorLogic) as the intended printing condition according to the PDF/X-5n standard. Transparency and layers are allowed.
Further Options
Analyze
Analyze and fix
Time limited version - Valid Until: 03.01.2017
Create Report...
Create Inventory...
Insert callas pdfToolbox Results as Comments
Remove callas pdfToolbox Comments...
Create New callas pdfToolbox Profile...
Create New callas pdfToolbox Process Plan...
Edit callas pdfToolbox Profiles...
Duplicate callas pdfToolbox Profile...
Delete callas pdfToolbox Profile...
Export callas pdfToolbox Profile...
Import callas pdfToolbox Profile...
Create Profile Summary...
Manage Libraries...
Export Library...
Import Library...


callas pdfToolbox: Edit Profile
Convert to PDF/X-1a (ISO Coated v2 (ECI))
Convert to PDF/X-1a (Japan Color 2001 Coated)
Convert to PDF/X-3 (Coated GRACoL 2006)
Convert to PDF/X-3 (ISO Coated v2 (ECI))
Convert to PDF/X-3 (Japan Color 2001 Coated)
Convert to PDF/X-4 (Coated GRACoL 2006)
Convert to PDF/X-4 (ISO Coated v2 (ECI))
Convert to PDF/X-4 (Japan Color 2001 Coated)
Convert to PDF/X-4 and create language layer views (ISC
Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))
Remove PDF/X information
Verify compliance with PDF/X-1a
Verify compliance with PDF/X-3
Verify compliance with PDF/X-4
Verify compliance with PDF/X-4p
Verify compliance with PDF/X-5g
Verify compliance with PDF/X-5n
Verify compliance with PDF/X-5pg
Convert to PDF/X-5n (6C Indigo TAC350 (ColorLogic))
Document
Pages
Images
Colors
Fonts
Rendering
Standards compliance
Standards compliance for embedded files
Convert to standard
Convert to standard for embedded files
Custom checks
Custom fixups
Summary
Place content
Preflight Certificate
Prepress
Unlocked
Name:
Convert to PDF/X-5n (6C Indigo TAC350 (ColorLogic))
General
Purpose:
Converts the current document to PDF/X-5n for 6C Indigo TAC350 (ColorLogic) as the intended printing condition according to the PDF/X-5n standard. Transparency and layers are allowed.
Author:
callas software
Email:
support@callassoftware.com
Group:
PDF/X compliance
Usage...
Save
Cancel
OK


callas pdfToolbox: Edit Profile
Convert to PDF/X-1a (ISO Coated v2 (ECI))
Convert to PDF/X-1a (Japan Color 2001 Coated)
Convert to PDF/X-3 (Coated GRACoL 2006)
Convert to PDF/X-3 (ISO Coated v2 (ECI))
Convert to PDF/X-3 (Japan Color 2001 Coated)
Convert to PDF/X-4 (Coated GRACoL 2006)
Convert to PDF/X-4 (ISO Coated v2 (ECI))
Convert to PDF/X-4 (Japan Color 2001 Coated)
Convert to PDF/X-4 and create language layer views (ISC
Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))
Remove PDF/X information
Verify compliance with PDF/X-1a
Verify compliance with PDF/X-3
Verify compliance with PDF/X-4
Verify compliance with PDF/X-4p
Verify compliance with PDF/X-5g
Verify compliance with PDF/X-5n
Verify compliance with PDF/X-5pg
Convert to PDF/X-5n (6C Indigo TAC350 (ColorLogic))
Document
Pages
Images
Colors
Fonts
Rendering
Standards compliance
Standards compliance for embedded files
Convert to standard
Convert to standard for embedded files
Custom checks
Custom fixups
Summary
Place content
Preflight Certificate
Prepress
Unlocked
Name:
Convert to PDF/X-5n (6C Indigo TAC350 (ColorLogic))
Convert to standard
Convert PDF document to
PDF/X-5n
Apply fixups
Convert device independent RGB or Lab color into destination color space
Decalibrate device independent Gray or CMYK
Convert PDF document to
PDF/A-1a
Apply fixups
If regular conversion fails:
Re-convert via PostScript
Convert pages with problems into images
Convert all pages into images
Image resolution:
0
ppi
Convert PDF document to
PDF/E-1
Apply fixups
Set Output Intent to
6C Indigo TAC350 (ColorLogic)
Use embedded Output Intent if present
Omit ICC profile unless embedding is required
Usage...
Save
Cancel
OK


callas pdfToolbox
Profiles
callas pdfToolbox Default
Show all
Convert to PDF/X-5n
PDF/X compliance
Convert to PDF/X-5n (6C Indigo TAC350 (ColorLogic))
Edit...
Converts the current document to PDF/X-5n for 6C Indigo TAC350 (ColorLogic) as the intended printing condition according to the PDF/X-5n standard. Transparency and layers are allowed.
Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))
Further Options
Analyze
Analyze and fix
Time limited version - Valid Until: 03.01.2017

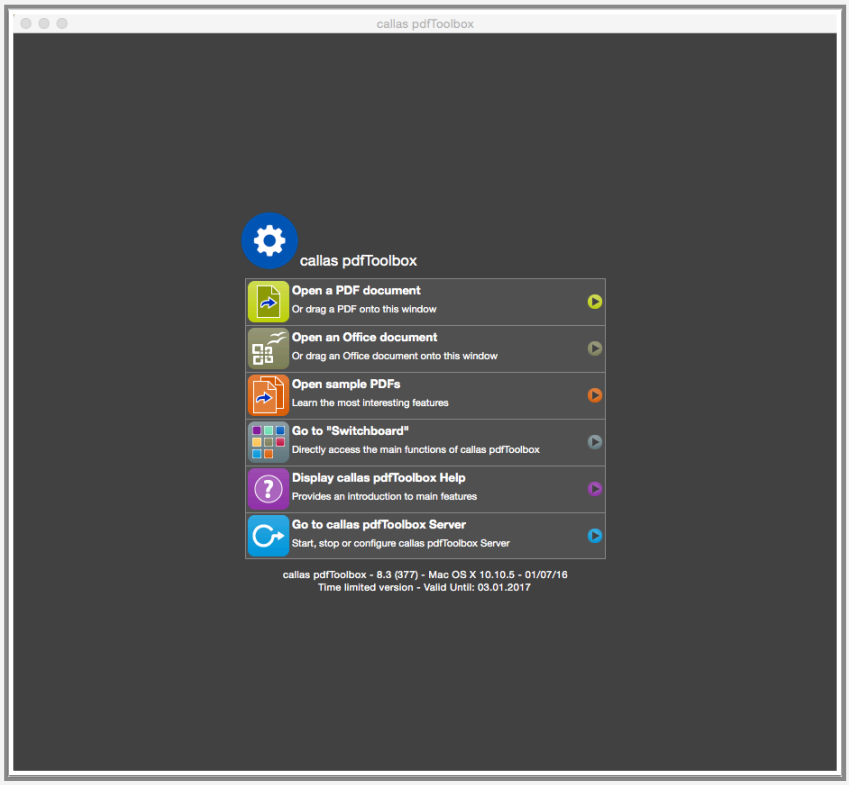

Save
Save PDF file
Save As: Display callas_6C_PDFX-5n
Tags:
Where: Output files
Cancel
Save

8.3 Validate against PDF/X-5n standard

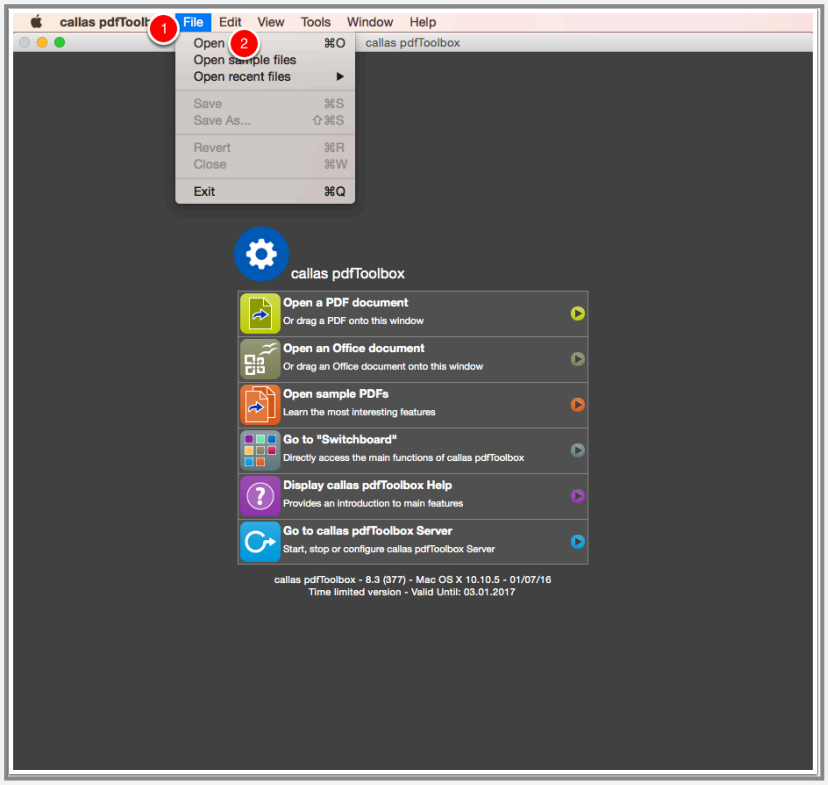
From pdfToolbox 8.0 onwards, you are able to convert PDF files to PDF/X-5n and validate the standard too.

Display_callas_Demo_file.pdf

Launch pdfToolbox Desktop

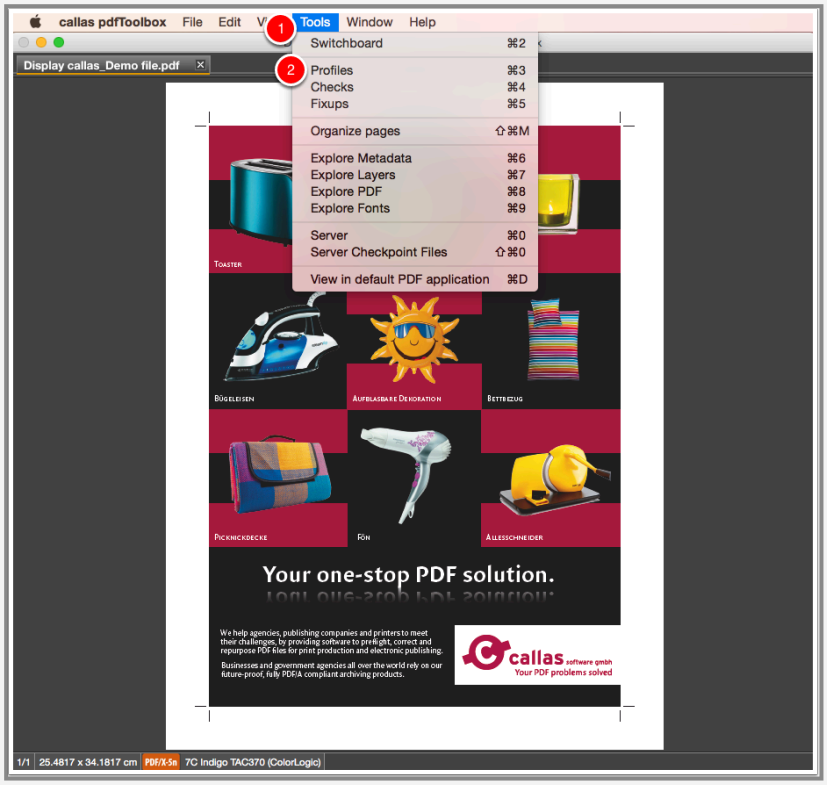


Open the PDF file "Display callas_Demo file.pdf"



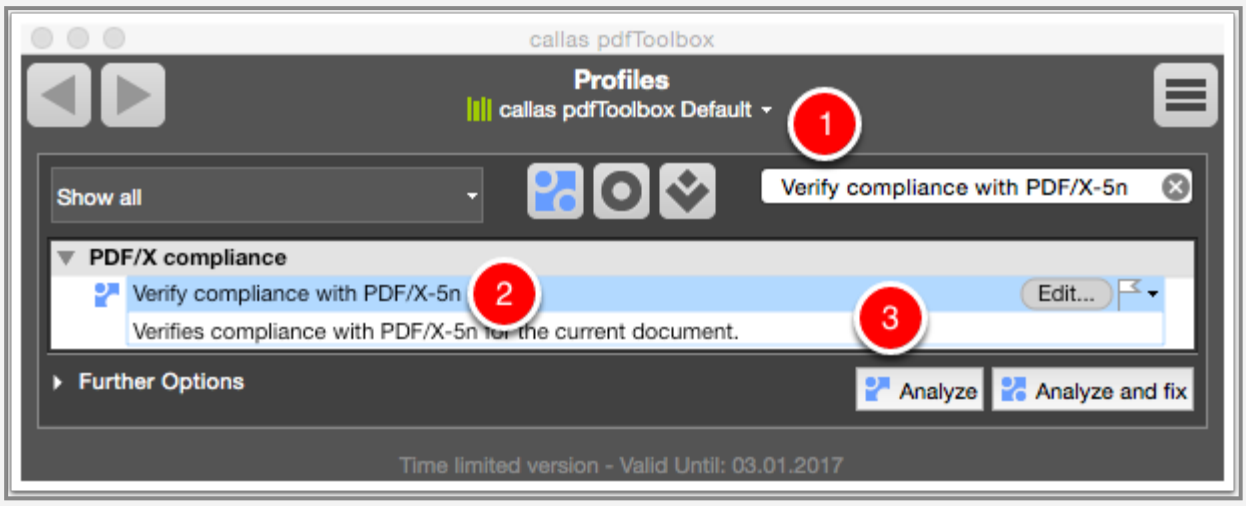
1. Go to "File".
2. Click "Open" to launch the file load dialog box and navigate to the folder where the input PDF file "Display callas_Demo file" is located.

Open the Profile dialog



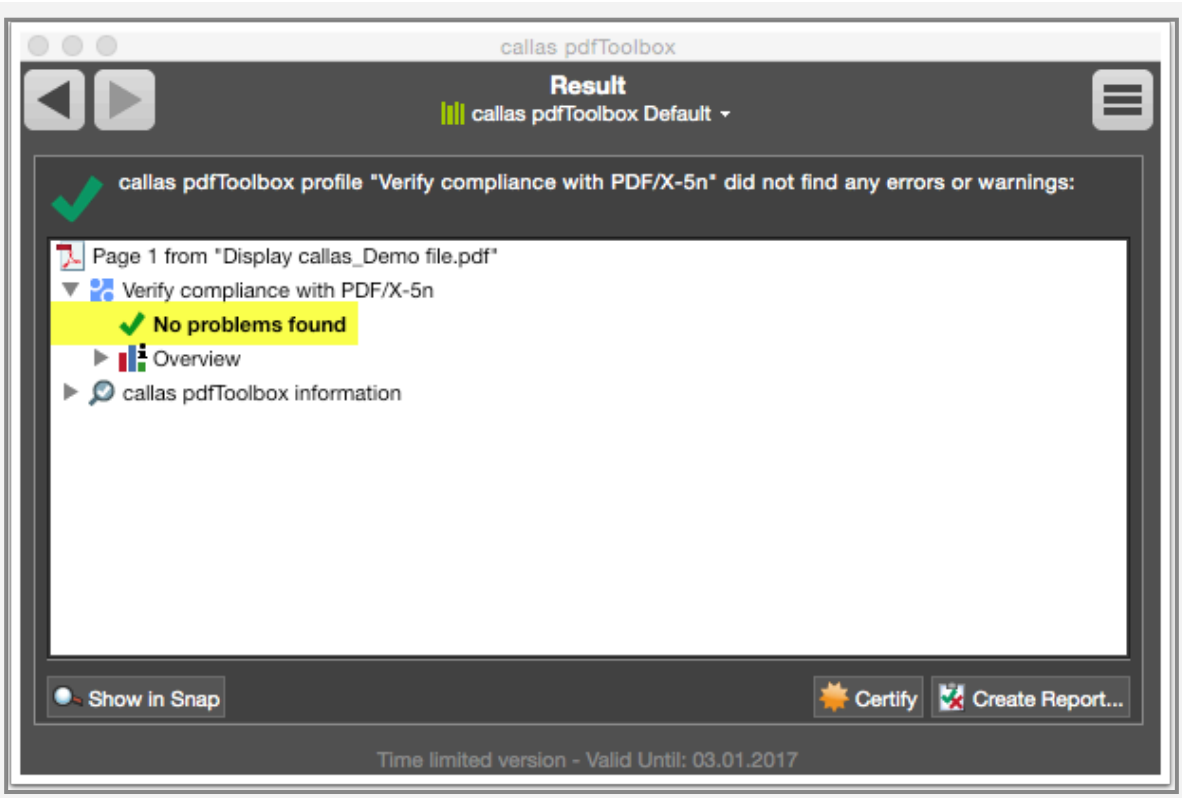
1. Go to "Tools".
2. Click "Profiles".

Verify compliance with PDF/X-5n



1. In the search field search to "Verify compliance with PDF/X-5n".
2. Select Profile "Verify compliance with PDF/X-5n".
3. Click "Analyze".

Review the preflight report



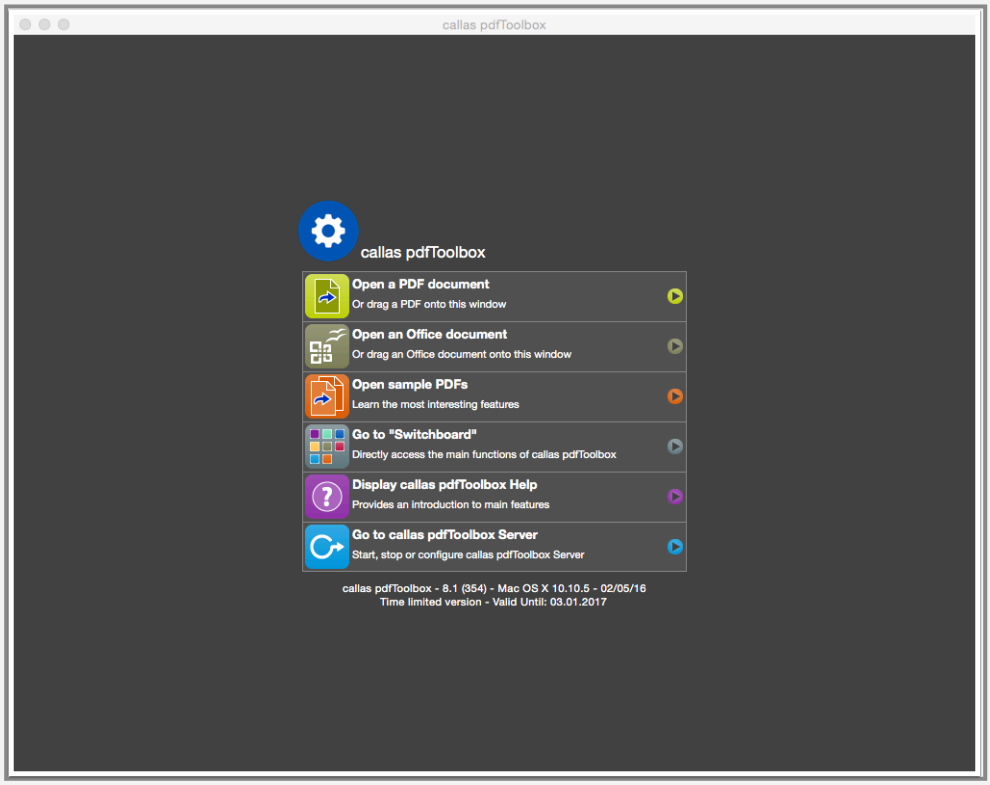
A green tick shows. No problems are found.

8.4 Softproof PDF/X-5n files

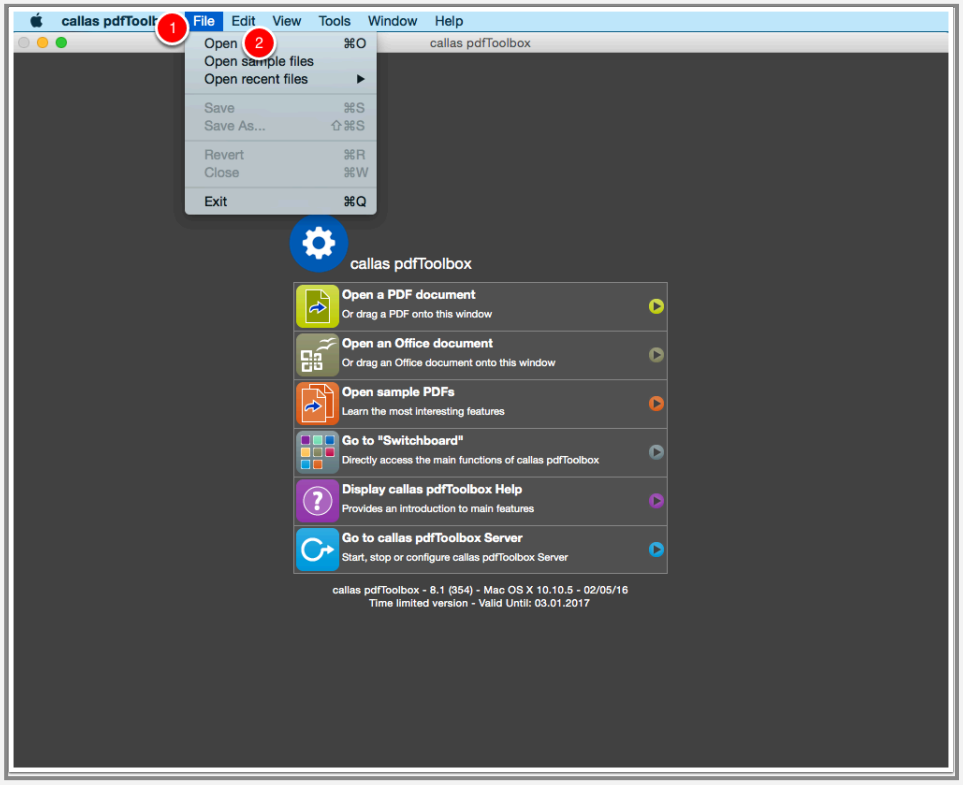
pdfToolbox 8.1 onwards, PDF/X-5n files are shown correctly in the Output Preview with the appropriate output intent profile.

OurCompany_X5n_Demo_file.pdf

Launch pdfToolbox Desktop

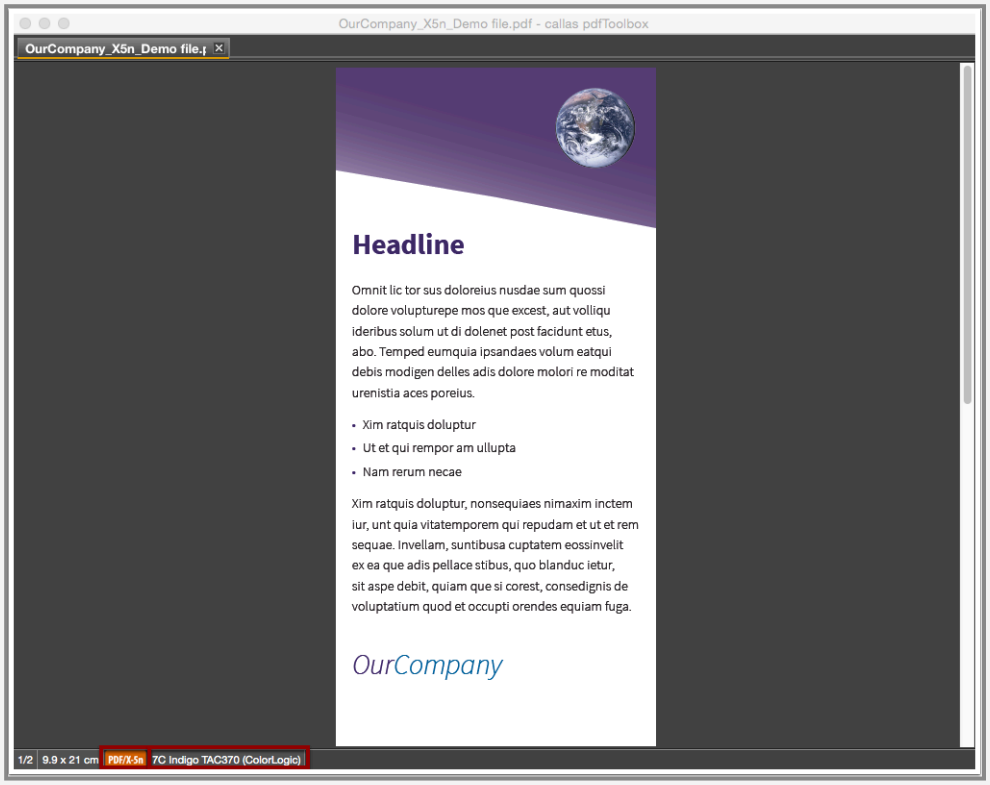


Open the PDF file "OurCompany_X5n_Demo file.pdf"



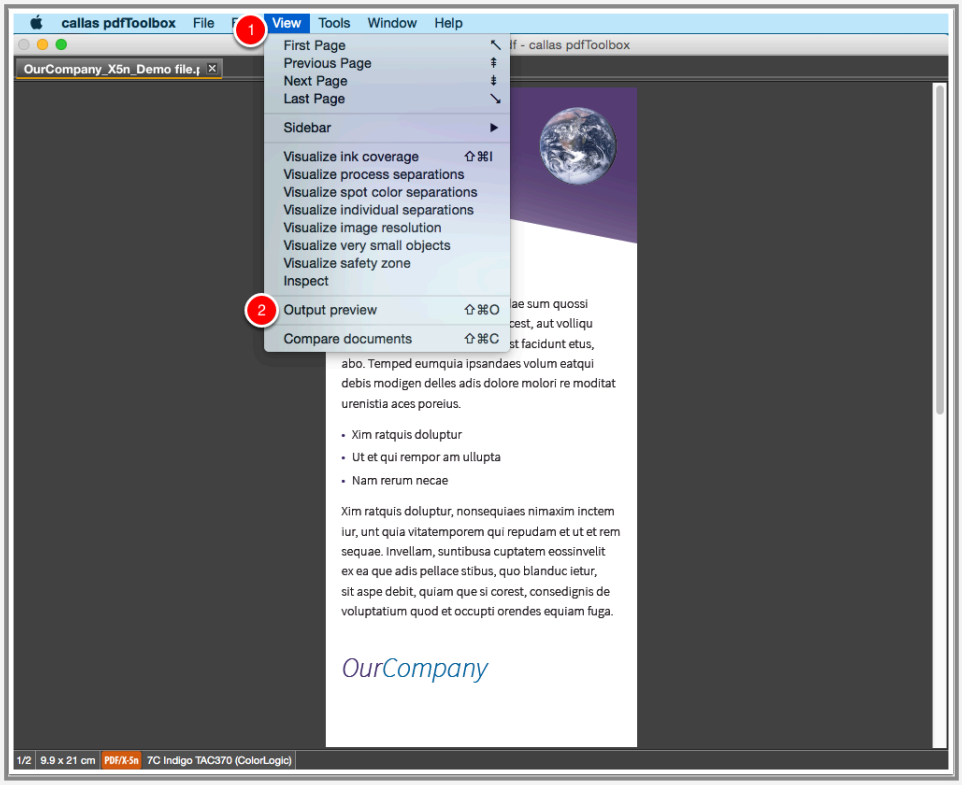
1. Go to "File".
2. Click "Open" to launch the file load dialog box and navigate to the folder where the input PDF file "OurCompany_X5n_Demo file.pdf" is located.

Inspect the PDF file



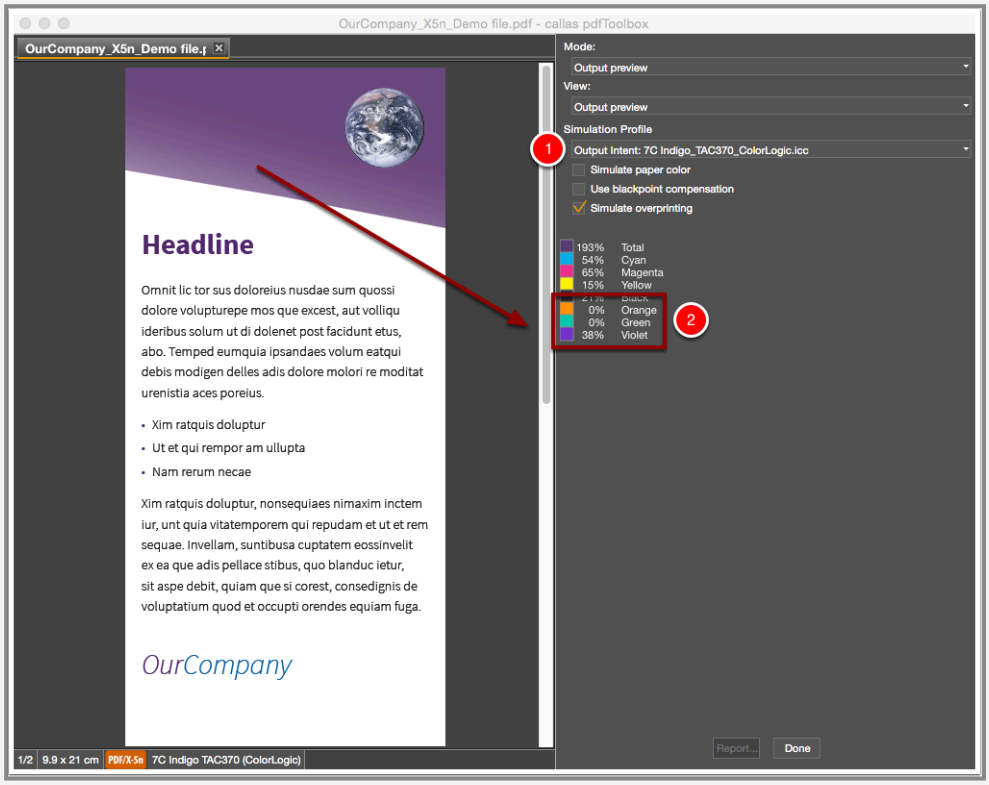
The PDF/X-5n label is represents in the main window. Also the output intent profile "7C Indigo TAC370 (ColorLogic)" is showed.

Open Output preview panel



1. Go to "View".
2. Click "Output preview".

Inspect PDF file



The Output Preview panel simulates the printing using the correct simulation profile "7C Indigo TAC370 (ColorLogic)".

1. Automatically the output intent profile "7C Indigo TAC370 (ColorLogic)" is showed in the Simulation Profile box.
2. Click in the picture on the purple area. You see the value of the three additional channels for the 7-color printing.

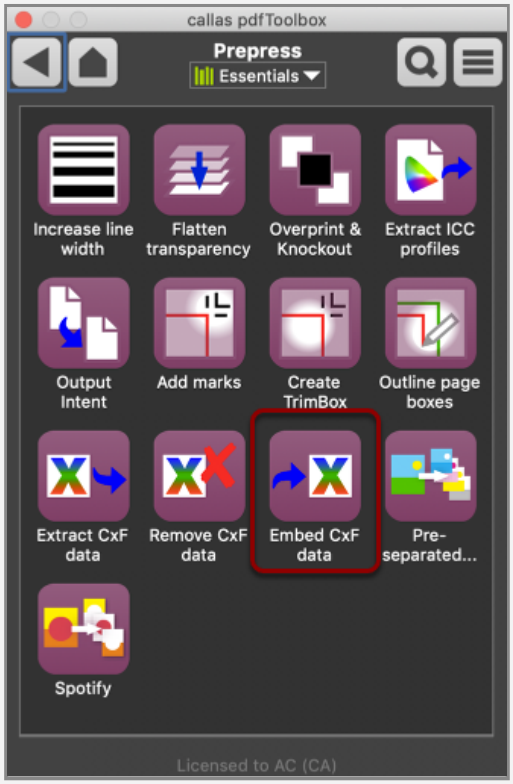
9. Spectral color and CxF

9.1 Introduction: CxF and spectral data

9.2 Embed CxF data (import)

In order to import CxF information CxF XML files need to be present in a folder and their name has to be the spot color name that it represents.

Open Switchboard -> Prepress -> Embed CxF data

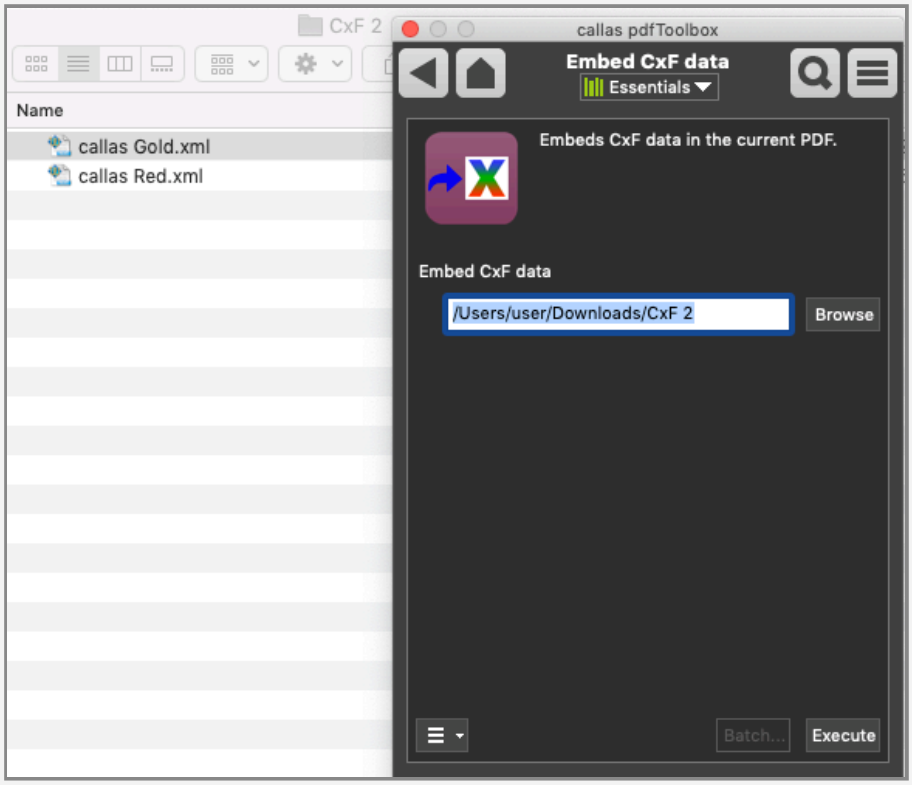


Open a PDF/X file



The PDF needs to be a PDF/X file or has to have at least an PDF/X Output Intent entry, since the CxF information is being embedded into the Output Intent entry. If there is no Output Intent entry present the "Execute" button in pdfToolbox cannot be hit.

Select a folder



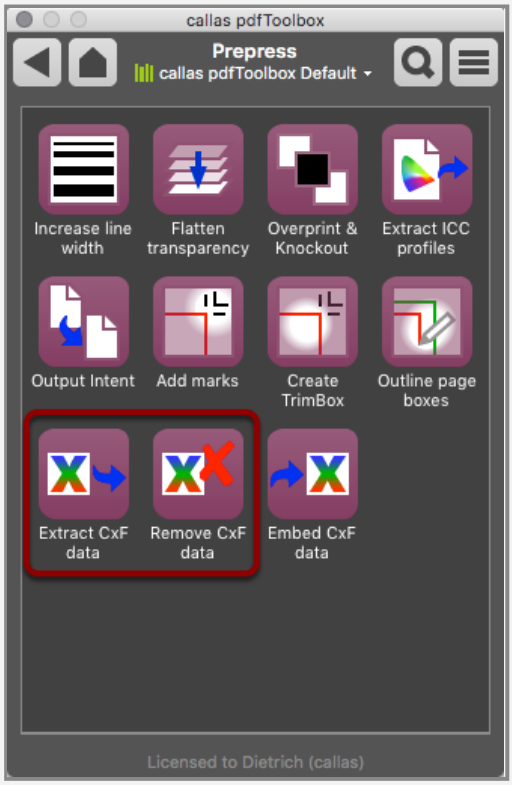
Click on Browse and select a folder that contains CxF XML files. Click on Execute in order to embed the CxF XML files.

The presence of CxF information in the result PDF is indicated by a CxF button at the bottom of the pdfToolbox window



9.3 Extract and remove CxF information

In order to extract or remove CxF information go to Switchboard -> Prepress



Open a PDF that has CxF information attached



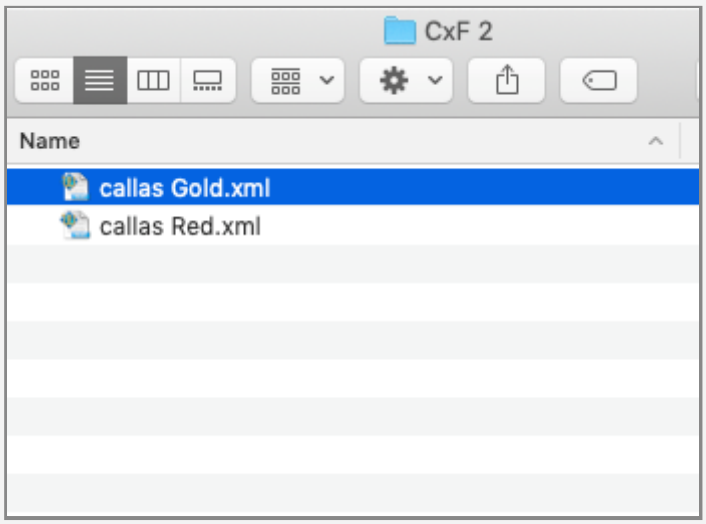
The presence of CxF information is indicated at the bottom of the pdfToolbox window.

Extracting CxF data



If you extract CxF data you are asked for a folder in your file system. For each CxF information in the PDF an XML file is created in that folder that has the name of the spot color that it represents.

A folder with CxF information as extracted from a PDF file



Removing CxF data from a PDF



You will be asked for a location to save the new PDF to.

The CxF indicator disappears from the pdfToolbox window



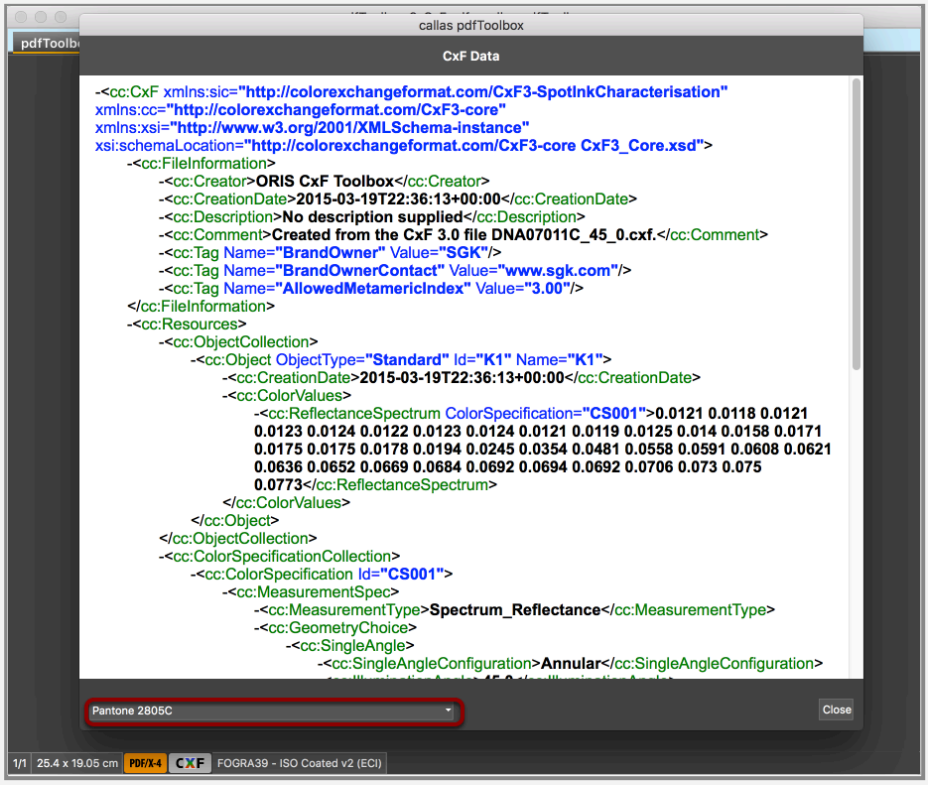
9.4 Analyze CxF information

Analyzing CxF information in a PDF file is easily possible...



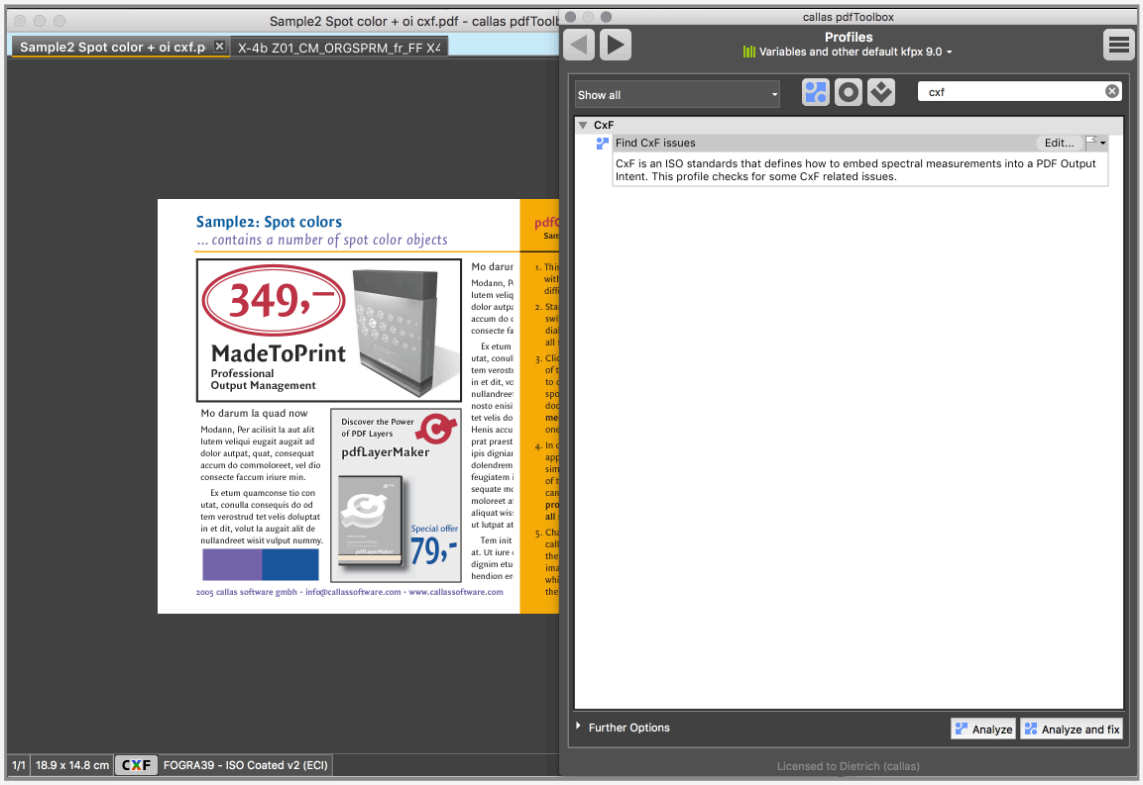
...by clicking on the CxF indicator at the bottom of the window.

A windows opens that displays the CxF data for the first spot color in its XML structure



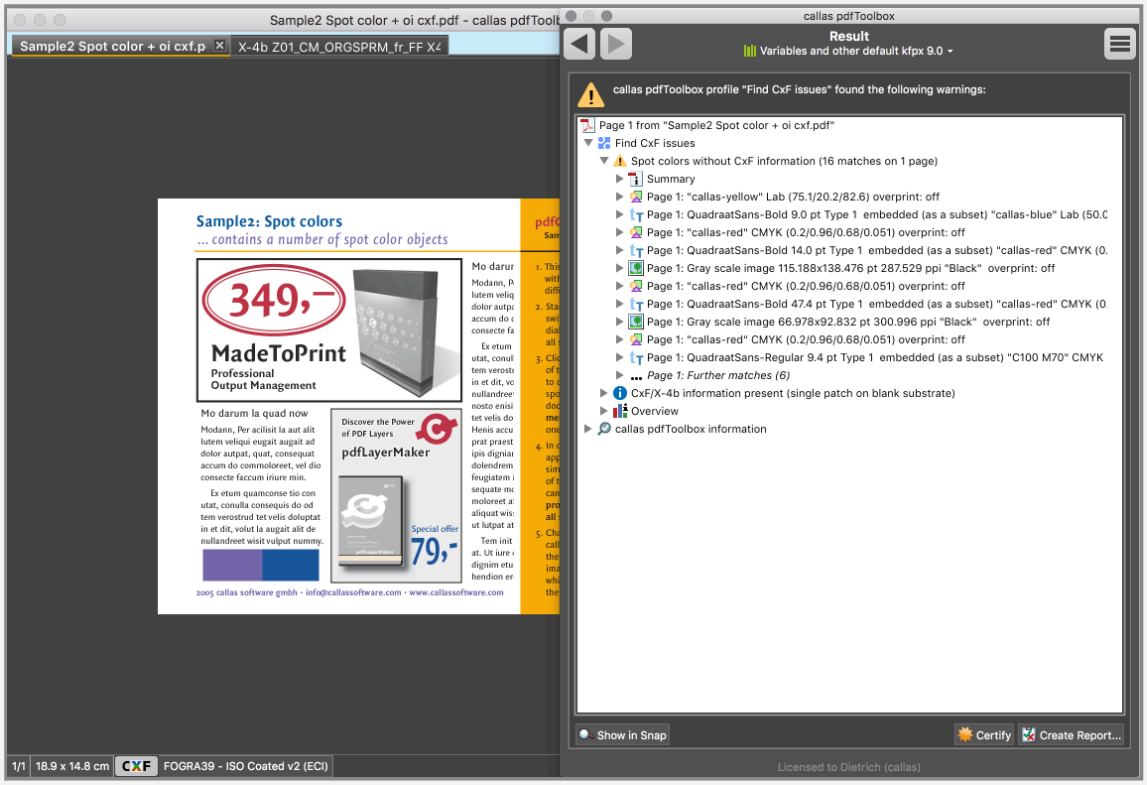
A pop up at the bottom of the windows allows you to select the spot color that you want to see.

It is also possible to run a profile that checks for various parameters of a PDF in combination with the embedded CxF information



Open the Profiles window and search for "CxF"

The CxF information profile "Find CxF issues"...



... reports for example if a PDF uses spot colors for which no CxF information is present.

10. Spotify - Derive spot colors from content (v11.0)

10.1 Why Spotify?

Some printing processes do not use CMYK process inks at all, instead only spot inks are available, and sometimes even a relatively small number. Furthermore, in some cases halftoning does not work at all, and thus iit may not be feasible to overprint spots with each other to achieve mixed ink colors.

This may be due to the way the printing process works, but could also go back to cost considerations, for example when imprints on small give-away items must be as inexpensive as possible and thus use only two spot colors.

For such scenarios, print ready PDFs should only use a specific number of spot colors, and always at a 100% tint value.

How to go from arbitrary images or PDFs to print-ready PDFs using only a small number of spot colors?

As any print service provider will try not to decline a print job just because the print file has not been prepared perfectly well, the question in this context is: how to turn an arbitrary PNG or JPEG image, or an arbitrary PDF, into a print ready PDF file that only uses a specified number of spot colors, none of them overprinting each other, and each of the spot colors with a tint value of 100% – while maintaining the overall appearance of the orginal image or PDF as well as possible?

This is where the new Spotify feature in pdfToolbox comes into play. Based on a complex analysis process, and controlled by a number of user configurable parameters, it derives the spot colors that offer the best appearance match, and assigns these to the respective areas of the original PDF, in order to provide a clean looking and pleasing result.

Everything you need to know about Spotify right here in this video:

10.2 How does Spotify work

Spotify uses a combination of techniques to allow a user to produce pleasing results:

- identify areas in an image that are to be ignored
- handle artifacts, whether created by anti-aliasing, JPEG artefacts or 'impurities' in the image
- find the specified number of individual colors that best represent the whole image
- assign the best matching color to each pixel (or leave it transparent by masking it out)

Areas to ignore

Areas in an image may have to be ignored for a number of reasons:

- in most cases, white areas are to be ignored; this may include areas that are 'almost white'
- almost always, fully transparent areas are to be ignored; sometimes this also applies 'almost fully transparent' areas
- in rare cases, gray areas are also to be ignored (this is only available in Spotify mode on the command line)
- in rare cases, black areas are also to be ignored (this is only available in Spotify mode on the command line)

Before going through the actual processing, and depending on how parameters are set, pixels in certain areas will not be taken into account (and will later on be masked out).

Artefacts

Even for digitally created – and seemingly very 'clean' – images there will be artefacts, most notably going back to antialiasing and – when JPEG compression is applied – to JPEG compression block artefacts.

The biggest issue with both is that in these areas colors will be present that are not representartive of the image appearance. Just envision an image with blue square next to a yellow square. If antialising kicks in, the pixels at the border between the two will use some shade(s) of green. Nonetheless

the representative colors for both areas still are blue and yellow, and not green.

Now envision a yellow background with some very thin text (e.g. using Helvetice Neue Light) on it using blue: there will be many yellow pixels (from the background), some but not many blue pixels (for the text) – and a sometimes surprisingly large number of green (or greenish) pixels – for the border between the blue text and the yellow background. In some real world files, the number of green pixels my be larger than the number of plain blue pixels.

Spotify has options – based on edge detection algorithms – that make it possible to ignore such border pixels when looking for the best matching colors, but to still to take these pixels into acount when assigning the best matching colors (and in this example assigning either blue or yellow to the green or hreenish pixels).

Determine best matching colors

Spotify uses k-means clustering algorithms to identify those (usually few) colors that best represent an image. Research has shown that first looking for more colors than needed, and then condensing the list of found colors into a smaller list of just those few desired colors almost always gives better results than looking directly for the desired number of colors.

As k-means clustering by its very nature has random aspects to it, running the algorithm twice (or more often) on the same data set will yield slightly (or even not so slightly) different results each time. As a consequence, Spotify will go through several iterations and the pick the iteration with lowest aggregated difference between original image and image represented by the result colors.

Assigning spot colors to pixels

Regardless whether Spotify is used on images or PDFs containing not just images but also text and vector, the result will always be an image on a PDF page, where that image extends over the whole page area. That image

- has one channel for each spot color

- each channel is bitonal, that is it is either 100% of the given spot color, or 0%
- for each pixel, only one of the channels will be set to 100%
- for every pixel, where all channels are set to 0%, a mask is applied to make the image transparent at th given pixel.

As the only way to encode such images in PDF is to use a DeviceN color space, the created image will be a DeviceN colored multi-channel bitonal image.

Names for the spot colors are either taken from a spot color library provided by the user, or generated from the RGB values that represent the appearance of the spot color.

The alternate space for the spot colors is by default encoded using sRGB. For the Spotify mode on the command line it is also possible to use DeviceRGB or Lab as the alternate space. Using Lab has proven to cause rendering problems on at least some viewers.



This video below goes directly to 'how Spotify works':

10.3 Spotify in callas pdfToolbox

Spotify features are available in the following ways:

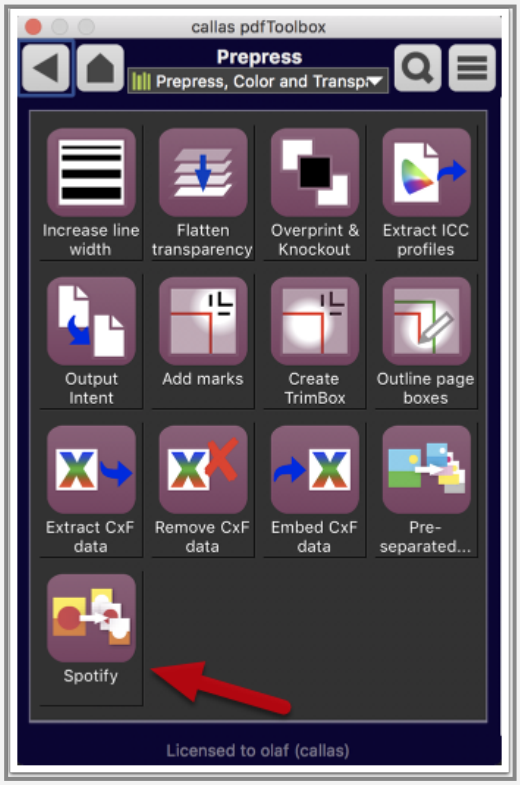
- as an action, which works in the same way as the fixup, but offers access only to the two most relevant parameters and thus is easier to use
- as a fixup, with access to most parameters that can be configured
- as an inspection tool, with access to most parameters, offering a way to play around with the parameters in an ad hoc manner and immediately see the effect of each parameter
- for command line versions of pdfToolbox in the form of a specific Spotify mode (providing access to the complete set of Spotify parameters)

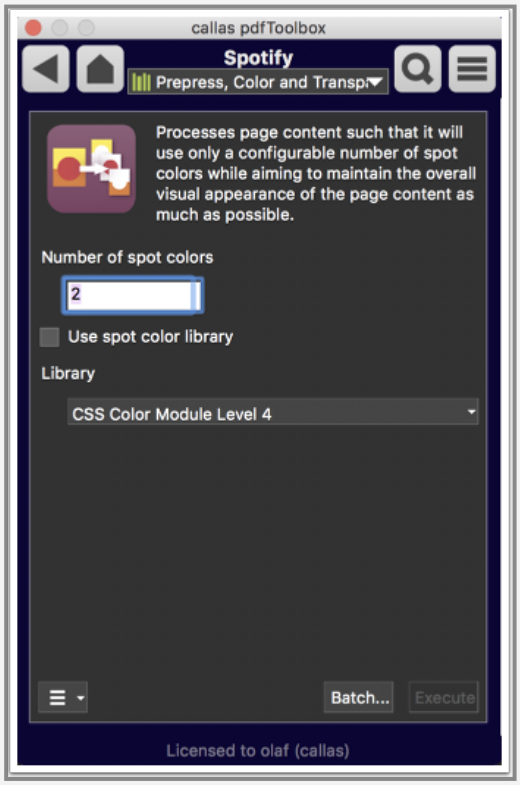
Meaning of Spotify parameters

The description of the various parameter available in the different ways Spotify may be used are available in a separate article Spotify parameters.

Spotify action in the Switchboard

You can find Spotify action in Switchboard under the category 'Prepress'.


callas pdfToolbox
Prepress
Prepress, Color and Transp
Increase line width
Flatten transparency
Overprint & Knockout
Extract ICC profiles
Output Intent
Add marks
Create TrimBox
Outline page boxes
Extract CxF data
Remove CxF data
Embed CxF data
Pre-separated...
Spotify
Licensed to olaf (callas)


callas pdfToolbox
Spotify
Prepress, Color and Transp
Processes page content such that it will use only a configurable number of spot colors while aiming to maintain the overall visual appearance of the page content as much as possible.
Number of spot colors
2
Use spot color library
Library
CSS Color Module Level 4
Batch...
Execute
Licensed to olaf (callas)

Managing spot color libraries

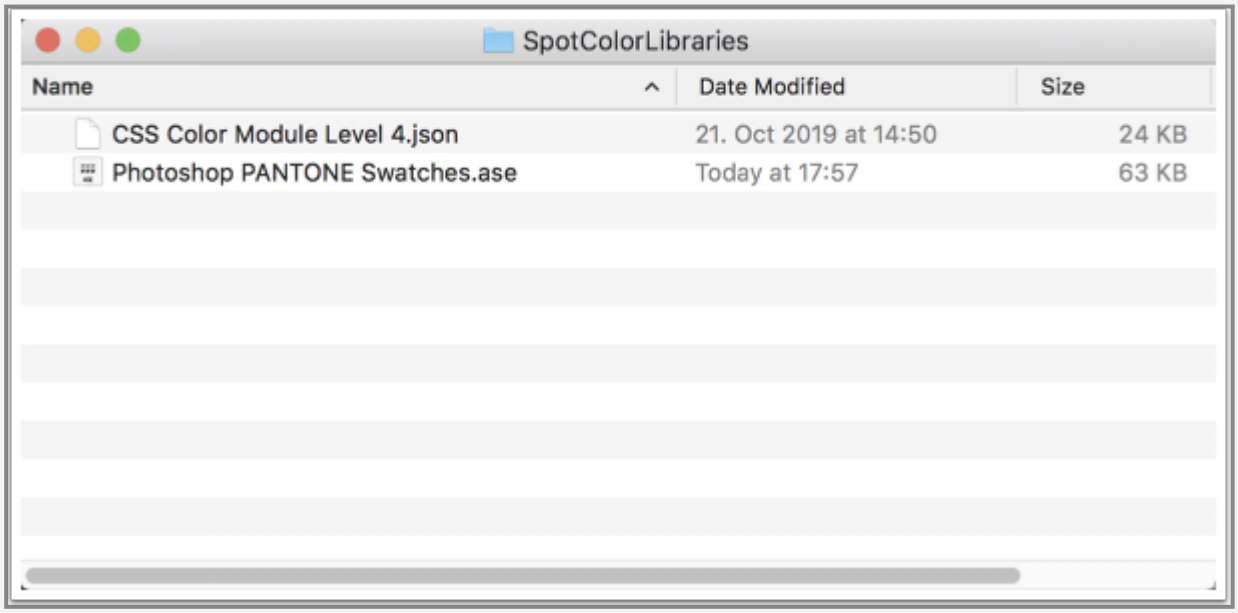
By default, we use 'CSS Color Module Level 4' as the spot color library. More info like the list of web colors here. You can select spot color library to use for Spotify action using the dropdown as shown below.



Setting up your own spot color libraries



Set up your own spot color libraries by clicking on the options icon on the bottom left of the Switchboard window and then selecting 'Managing spot color libraries'.

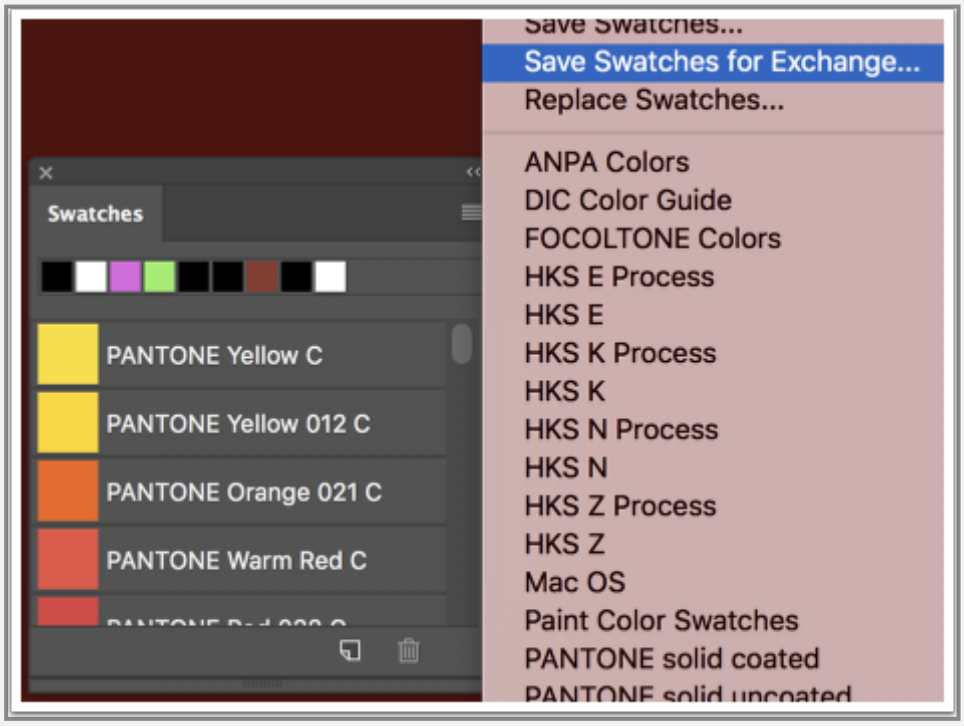


You can copy your own color libraries in 'SpotColorLibraries' folder (for example ASE files exported from Adobe Photoshop as in the next screenshot).

Export spot color libraries ("Swatches") from Adobe products

ACO (Adobe color file) and ASE (Adobe Swatch Exchange) files can be exported from Adobe products like Photoshop. These formats makes it easy to share colors between programs.

Also, a file with the ACB file extension is an Adobe Photoshop Color Book file. They're used to provide an easy way to comply with particular color standards, like for if you're printing an image versus using it for on-screen purposes.



Editing spot color libraries

You can edit the spot color libraries (in json format) that are found under:

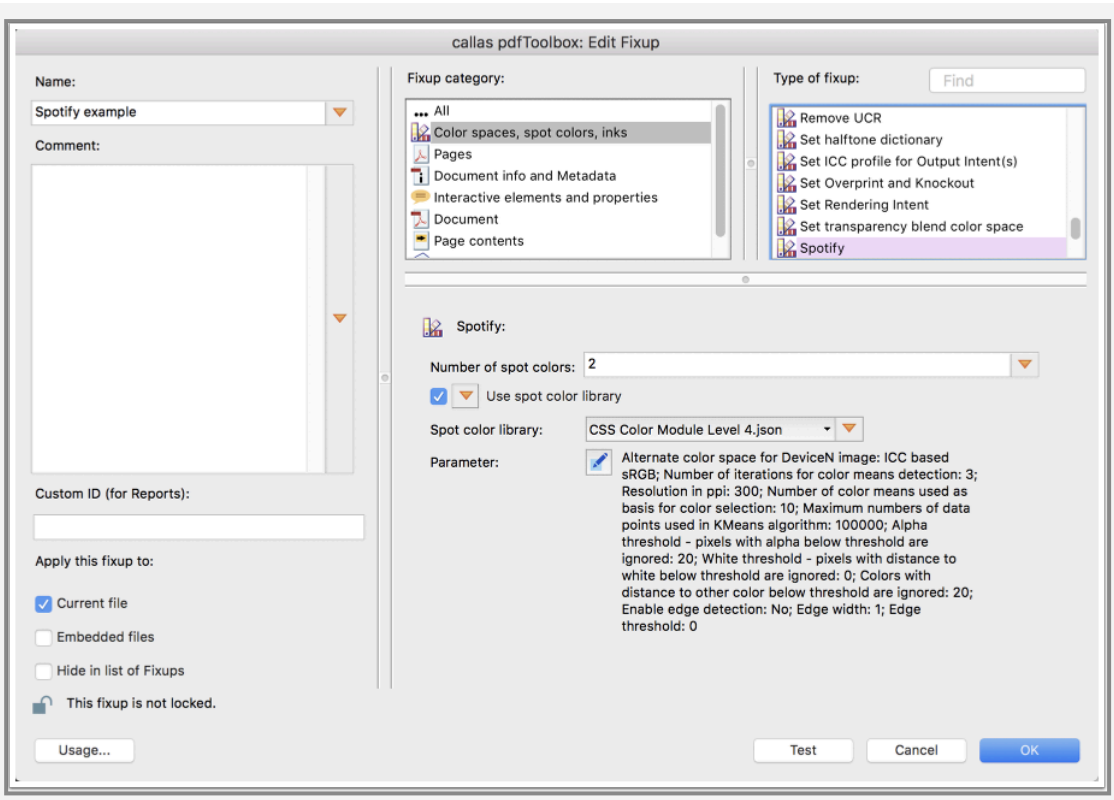
C:\Users\admin\AppData\Roaming\callas software\callas pdfToolbox 11\Repositories\Custom\20200622_145335\Spot-ColorLibraries

Color components in ASE files are stored using float values, for instance in RGB component values goes from 0 to 1 (instead of 0 to 255). For RGB and Gray colors, an convenient hex field with the HTML hex color format is included.

Please note:
RGB + L (Lab) values are from 0 to 1
a, b from Lab values are from -127 to 128

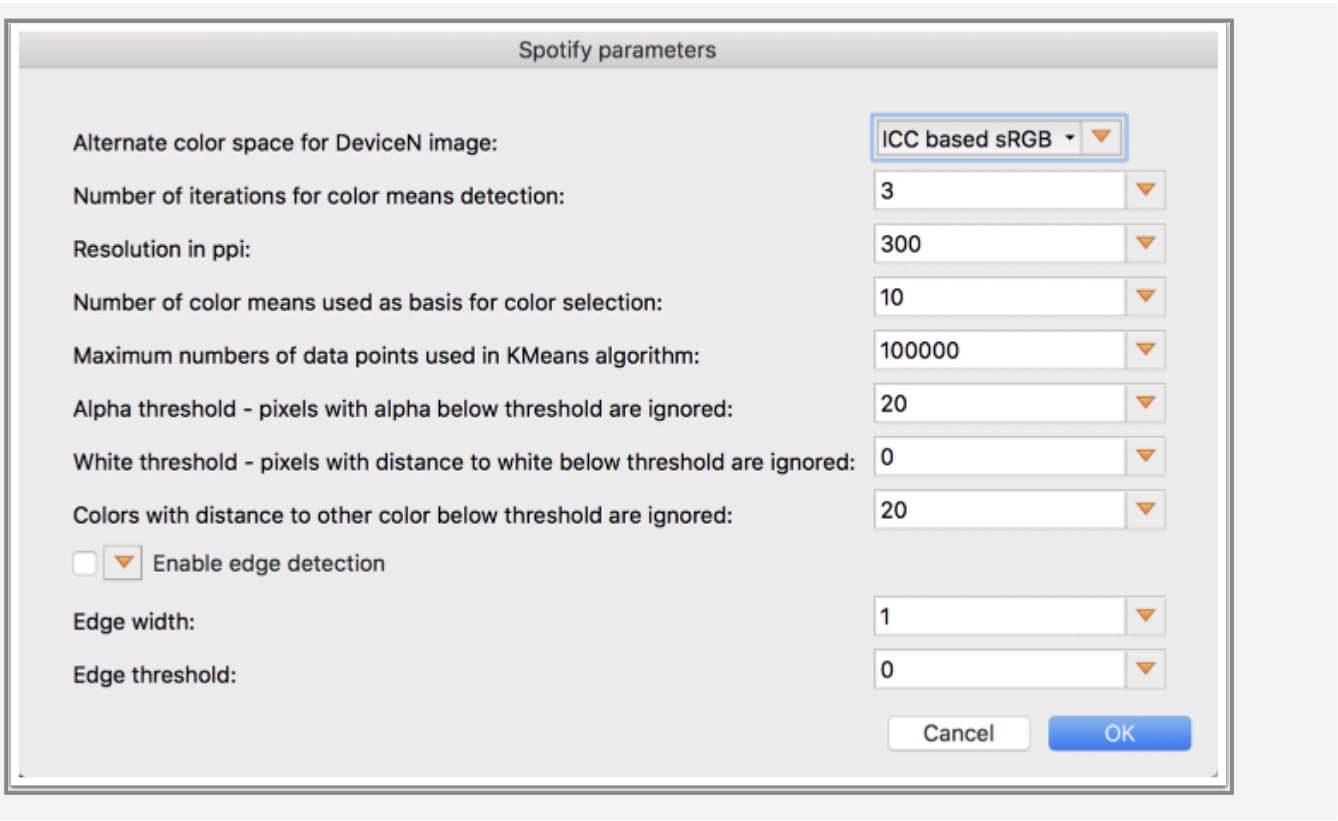
Spotify Fixup

To use Spotify in a Fixup, you can either create a new Fixup or edit an already existing Fixup. Find Spotify as the type of Fixup (under Fixup category 'Color spaces, spot colors, inks').



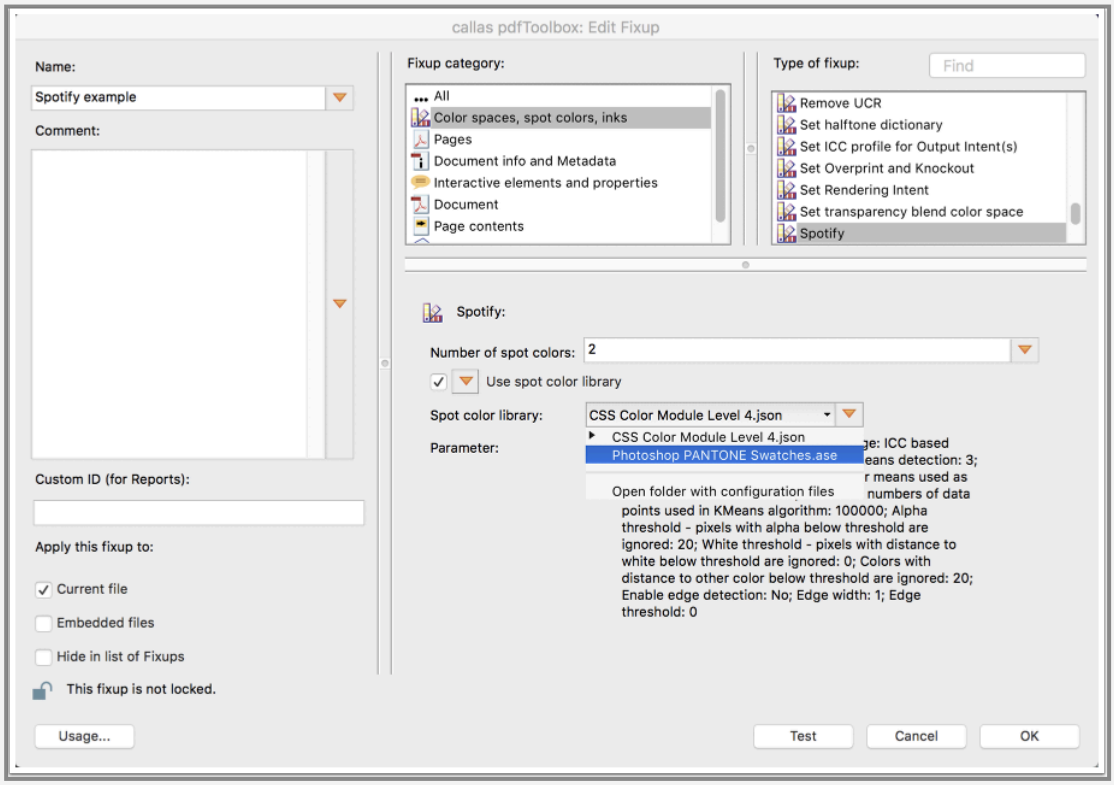
Spotify parameters

Explained here

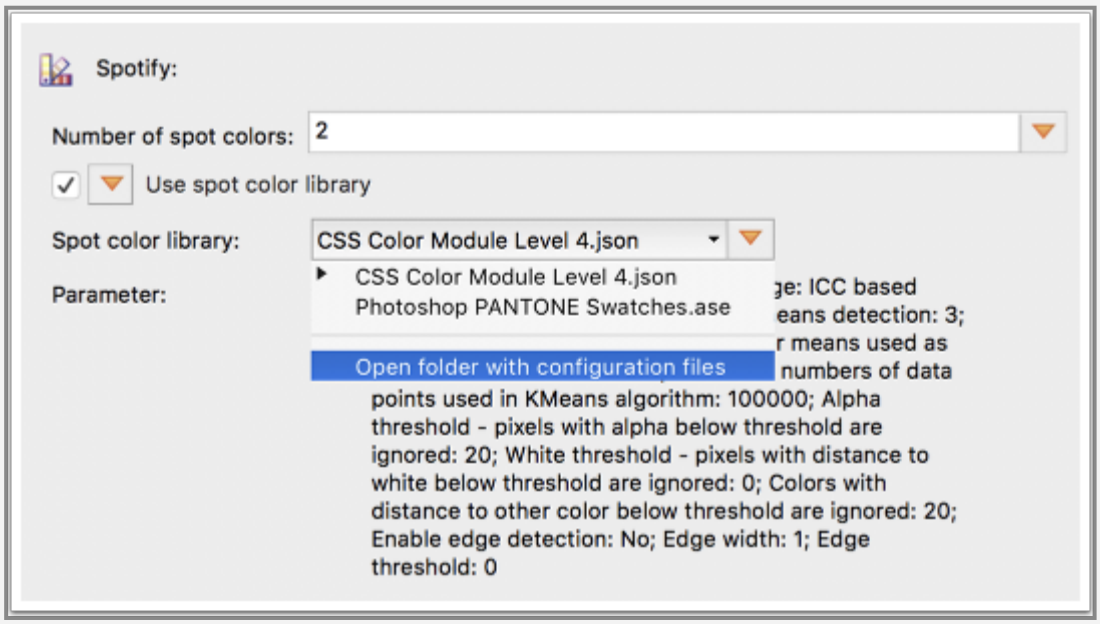


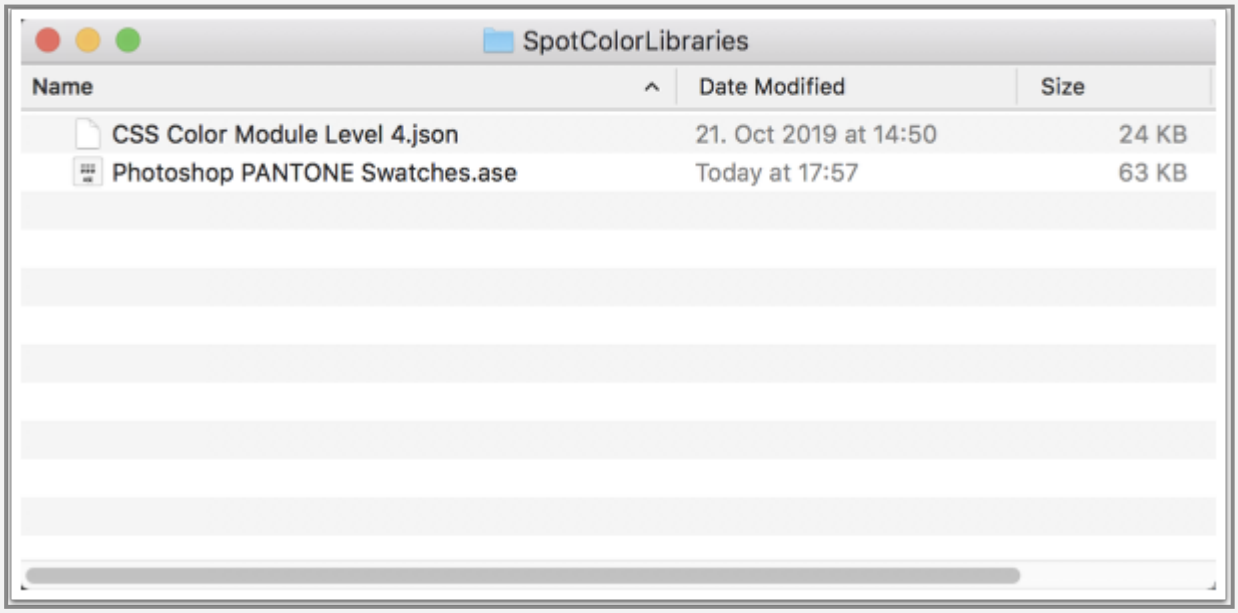
Managing spot color libraries

You can select spot color library to use for Spotify Fixup using the dropdown as shown below.



Setting up your own spot color libraries

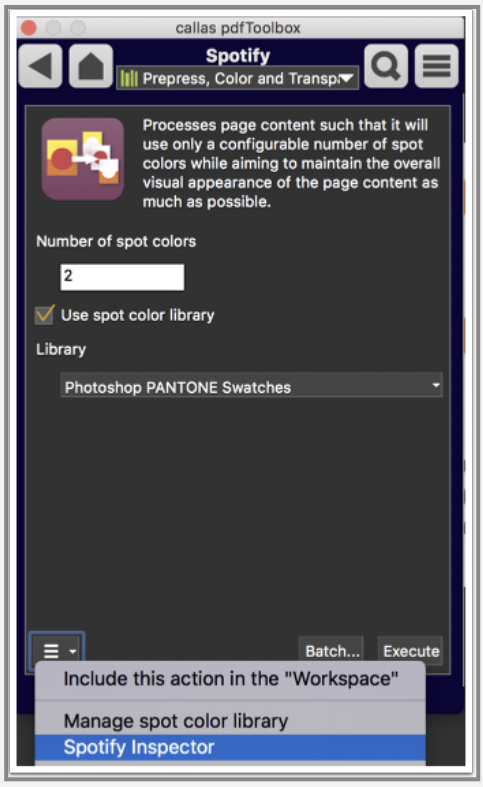




You can copy your own color libraries in 'SpotColorLibraries' folder (for example ASE files exported from Adobe Photoshop as explained above).

Spotify Inspector

Open Spotify Inspector via the menu at bottom of Spotify action in the Switchboard



Spotify Inspector window



Spotify mode in the CLI version

For a full list of options and parameters available for the Spotify mode of the command line, use

```
./pdfToolbox --help spotify
```

Click to copy

Please not that some of the parameters listed by `--help spotify` are specific to the Spotify mode, whereas others are general parameters applicable also to other modes.

Spotify usage in pdfToolbox Switchboard, Fixups and command line here:

10.4 Spotify parameters

Quite a number of parameters control how Spotify operates. The degree to which these parameters are accessible and can be configured varies, depending on the context where the Spotify functionality is used. The Spotify Switchboard action only offers access to two parameters: number of spot colors to create, and whether and which spot color library to work against. The Spotify fixup, the Spotify Inspector and the Spotify mode available on the command line on the other hand offer access to all or most parameters. There are a few parameters to which access is only provided on the command line, as they only can be put to good use in command line mode. Below a list of all parameter descriptions can be found.

Iterations

Number of iterations for color means detection (default: 3)

Spotify Switchboard action: n.a.

Spotify inspector: “KMeans:” → “Iterations”

Spotify fixup: “Number of iterations for color means detection”

Spotify mode in CLI version: --iterations

Max. runs

Maximum number of runs inside one iteration for color means detection (default: 10)

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --maxruns

Resolution

Resolution in ppi (pixels per inch) (default: 72) or "Width x Height" in pixel

Spotify Switchboard action: n.a.

Spotify inspector: Resolution

Spotify fixup: Resolution in ppi

Spotify mode in CLI version: --resolution

Max. samples

Maximum number of data points to be used by KMeans algorithm. Only has an effect if the number of pixels to be processed (possibly already constrained by --resolution) exceeds this value. Data points are selected randomly.

Spotify Switchboard action: n.a.

Spotify inspector: Max samples

Spotify fixup: Maximum number of data points to be used by KMeans algorithm.

Spotify mode in CLI version: --maxsamples

Alpha threshold

Alpha threshold. Pixels with alpha < threshold are ignored. An alpha value of 0 means *fully transparent*, and a value of 255 means *fully opaque* (default: 0)

Spotify Switchboard action: n.a.

Spotify inspector: Ignore pixels with alpha below threshold

Spotify fixup: Alpha threshold - pixels with alpha below threshold are ignored

Spotify mode in CLI version: --alphathreshold

White threshold

White threshold. Pixels with distance to white below threshold are ignored. If value is negative the threshold is applied to gray instead of RGB (default: 10).

For threshold values prepended with a minus sign, computation is done via a gray value derived by the formula YCbCr Luma = 0.299*r + 0.587*g + 0.114*b, values can range from 0 to 255 (for the comparison, the minus sign is actually discarded).

For threshold values not prepended by a minus sign, the comparison is done based on the L component of the Lab value of the color against an L value of 100 for white. Values can thus range from 0 to 100.

Spotify Switchboard action: n.a.

Spotify inspector: Ignore pixels with distance to white below threshold

Spotify fixup: White threshold - pixels with distance to white below threshold are ignored

Spotify mode in CLI version: --whitethreshold

Black threshold

Pixels with distance to black below threshold are mapped to black (default: 0)

Distance is computed as dE value in Lab between color value and a Lab value of 0,0,0 for black.

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --blackthreshold

Gray threshold

Pixels with distance to gray below threshold are mapped to gray (default: 0)

Computation is done based on the a and b components of the Lab value: a < threshold && b < threshold. Values can thus range from 0 to 128.

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --graythreshold

Color threshold

Colors with distance to other color below threshold are ignored (default: 20). Distance is computed as dE value in Lab.

Spotify Switchboard action:

Spotify inspector: dE Threshold

Spotify fixup: Colors with distance to other color below threshold are ignored

Spotify mode in CLI version: --colorthreshold

Number of spot colors as result

Number of spot colors to create (default: 2)

Spotify Switchboard action: Number of spot colors

Spotify inspector: Colors

Spotify fixup: Number of spot colors

Spotify mode in CLI version: --colors

Number of spot colors for analysis

Number of color means used as basis for color selection (default: 10)

Spotify Switchboard action: n.a.

Spotify inspector: Colors → out of

Spotify fixup: Number of color means used as basis for color selection

Spotify mode in CLI version: --colormeans

Spot color library

Use colors from an Adobe color book file (*.acb or *.aco) or JSON format (*.json) and match colors to nearest colors from color book

Spotify Switchboard action: Use spot color library → Library popup menu

Spotify inspector: Match colors → checkbox and popup menu

Spotify fixup: Use spot color library → Spot color library popup menu

Spotify mode in CLI version: --spotcolorlibrary

Number of gray colors

Number of gray means used as basis for color selection (default: 0)

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --graymeans

Edge detection

Enable edge detection.

For more information about the edge detection kernels supported and how they work, find some useful details at https://en.wikipedia.org/wiki/Edge_detection and http://www.cse.usf.edu/~r1k/MachineVisionBook/MachineVision.files/MachineVision_Chapter5.pdf

Spotify Switchboard action: n.a.

Spotify inspector: Enable edge detection

Spotify fixup: Enable edge detection

Spotify mode in CLI version: --edgedetection

Edge kernel

Edge kernel to use (default: *sobel*):

- *laplace*; 3x3 Laplace operator with weight 4
- *laplace2*; 3x3 Laplace operator with weight 8
- *log*: 5x5 Laplacian of Guassian operator
- *sobel*; 3x3 Sobel operator (coefficients: 1,2,1)
- *sobelopt*; 3x3 Sobel operator (coefficients: 5,10,5)
- *scharr*; 3x3 Scharr operator
- *canny*; 3x3 Canny operator

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --edgekernel

Edge width

Edge width in pixel (default: 1)

Spotify Switchboard action: n.a.

Spotify inspector: Enable edge detection → Width

Spotify fixup: Edge width

Spotify mode in CLI version: --edgewidth

Edge mode

Edge fill mode (default: *grow*):

- lab; fill edge pixels with color of mean with smallest distance in Lab (dE)
- hue; fill edge pixels with color of mean with smallest distance in Hue
- hsv; fill edge pixels with color of mean with smallest distance in HSV
- grow; fill edge pixels with color of nearest non-edge pixel with smallest dE (default)

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --edgemode

Edge threshold

Edge threshold (default: 0)

Suitable value ranges depend on the chose edge kernel.

Spotify Switchboard action: n.a.

Spotify inspector: Enable edge detection → Threshold

Spotify fixup: Edge threshold

Spotify mode in CLI version: --edgethreshold

Edge threshold 2

Edge threshold (default: 0)

This second edge threshold is used only for the Canny edge detection algorithm, for details see https://de.wikipedia.org/wiki/Canny-Algorithmus

Value ranges cannot easily be given since the thresholds are applied to the first or second derivation of the image brightness, thus the possible range is [0..∞]

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --edgethreshold2

Alternate color space (--alternatecs)

Alternate color space to use for the DeviceN image in the created PDF (default: *srgb*):

- *rgb*: Use DeviceRGB as the alternate color space for the DeviceN image in the created PDF
- *srgb*: Use sRGB as the alternate color space for the DeviceN image in the created PDF
- *lab*: Use Lab as the alternate color space for the DeviceN image in the created PDF

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --alternatecs

Output (--output)

- pdf: Create a PDF file with one page containing a single DeviceN image using spot colors
- png: Create an RGB PNG image representing the appearance of the PDF that has been or would be created using the ‘pdf’ option.
- json: Create a JSON formatted log file, representing the same output as seen in the log section at the bottom of the Spotify inspector window
- blob: A binary representation of the Spotify analysis result which may be used at a later stage for creation of a PDF file as would have been created using the ‘pdf’ option.

Spotify Switchboard action: n.a.

Spotify inspector: n.a.

Spotify fixup: n.a.

Spotify mode in CLI version: --output



Spotify via command line in this 5 minute video:

11. Fonts and Text

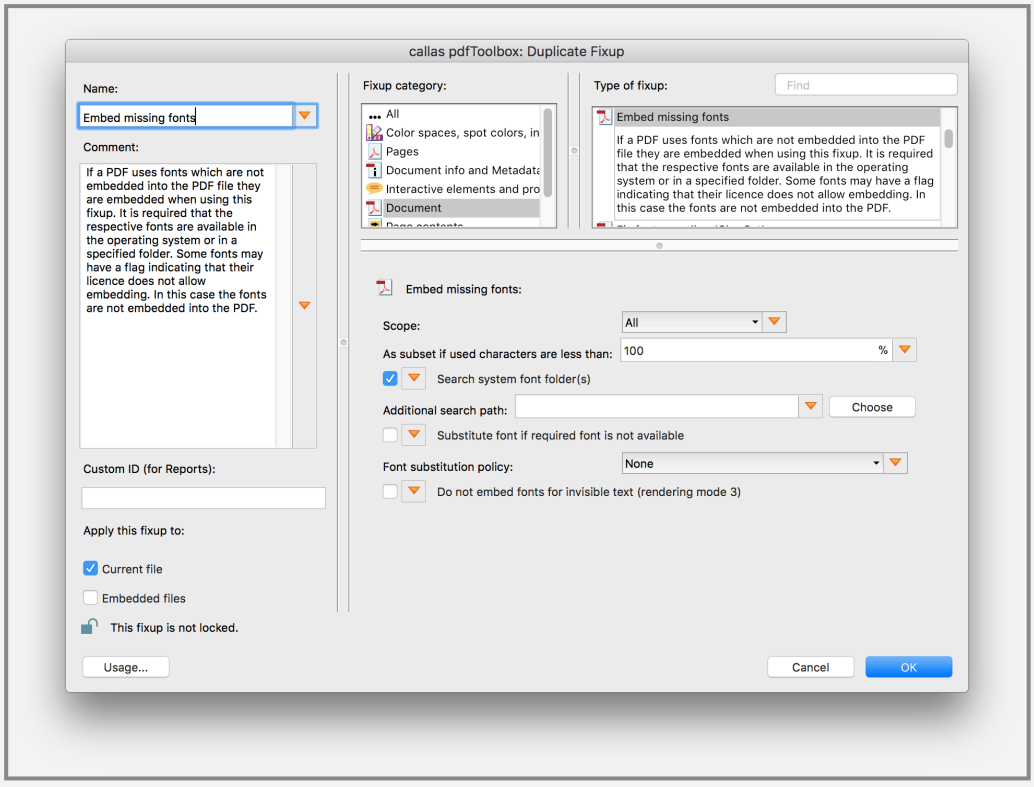
11.1 Embedding fonts

A PDF file does not necessarily have to contain all used fonts. If a font is used inside a PDF for a text, it is referenced by its name. The font can either be contained - then the contained font is usually used when the PDF is displayed, or cannot be contained - then the respective PDF viewer can search for a font with the referenced name on the system, and (if such a font is not found) use another font for displaying the text.

As several PDF-based ISO standards require used fonts to be embedded, the Fixup "Embed missing fonts" is already present in various Profiles for converting files into a standard-compliant PDF.

There are also predefined, single Fixups contained in the Libaries of the product.

The Fixup "Embed missing fonts"



Besides the possiblity to set the "Scope" (complete document, page contents and/or comments and/or forms), it is al-

so possible to embed the fonts as a "subset" (which means only necessary fonts will become embedded).

By default, several system font folders will be search for fonts to embed. You may either add an additional search path or limit the search to this specific folder.

Recursively it will search also in all sub-folders.

As sometimes not all needed fonts are available or only in a variant of the name (e. g. "Arial" ./. "Arial MT"), a substitution of fonts can make sense.
Sometimes it is also more important to ensure the content is maintained and smaller visual differences (caused by using a slightly different font face) are acceptable.
You'll find a detailed explanation how to set up a Font substitution policy file in the respective article.

There is also the possibility to exclude invisible fonts (using the so called "rendering mode 3", usually used for OCR documents) from embedding.

Supported file formats

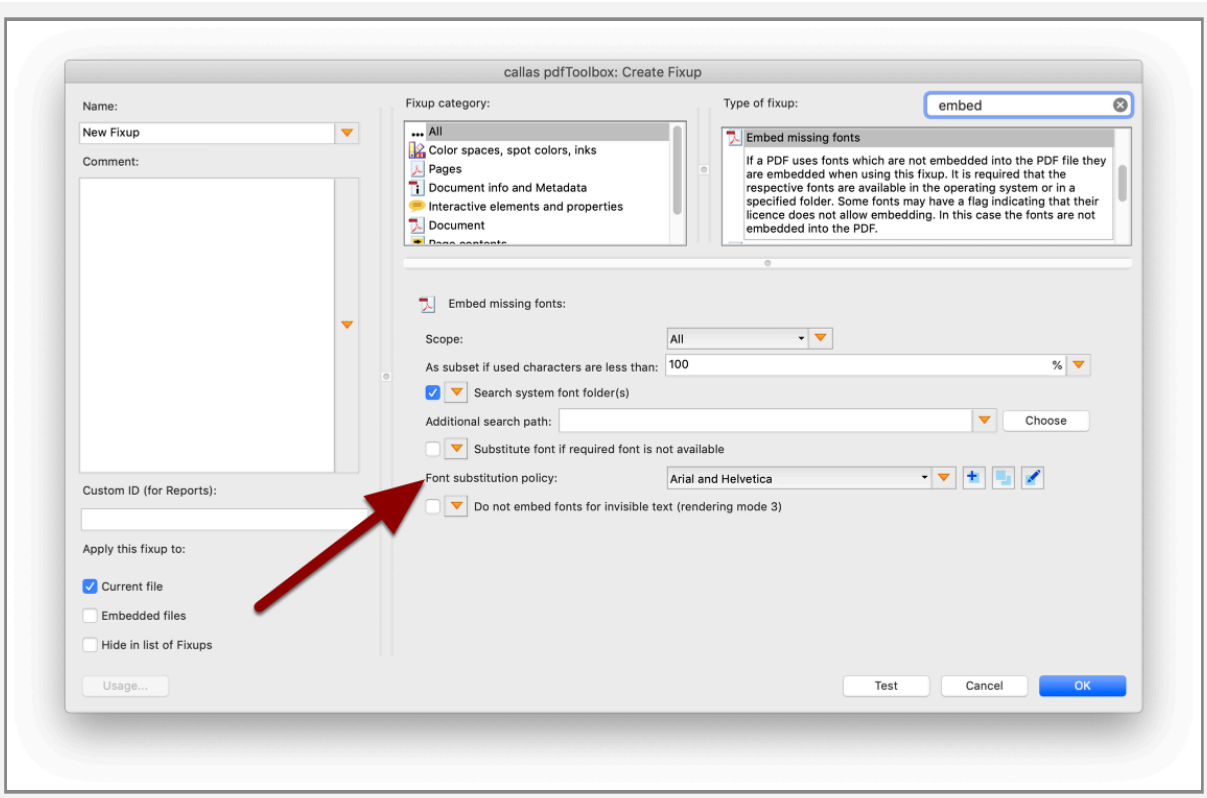
The following font formats are supported:

- PostScript Type1 Fonts
- CFF Fonts
- CFF-CID Fonts
- OpenType Fonts
- TrueType Fonts
- TrueType Collections

In addition, various wrapper formats are recognized, e. g. Mac resource' sfnt' files.

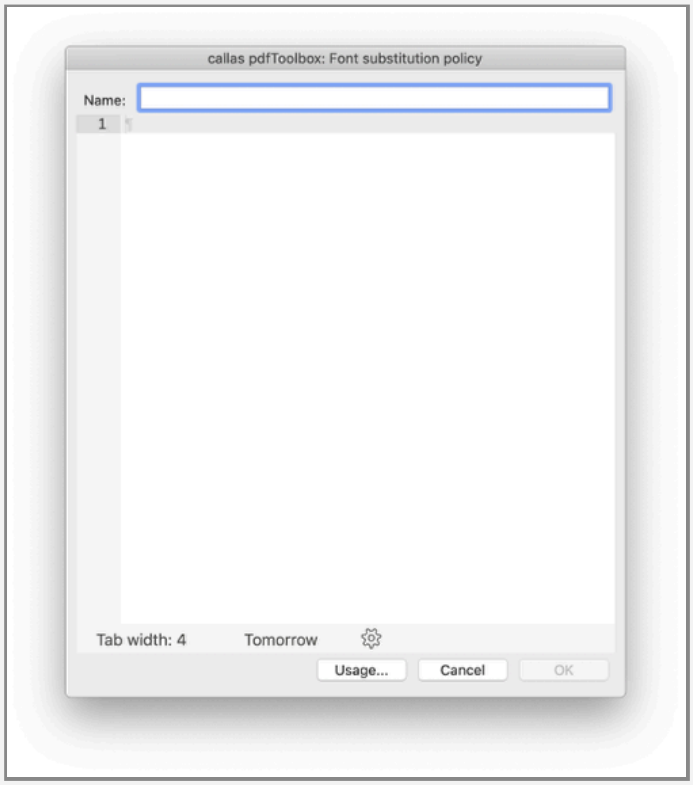
11.2 Embedding fonts: Font substitution file

The Fixup "Embed missing fonts" offers the possiblity to define a "Font substitution file", which can contain one or more lines of text, defining rules of how fonts (which are not available on the respective system) shall be replaced with other fonts.



Besides other options, the "Font substitution policy" can be defined in the Fixup.

Apart from choosing the options from the drop down, own substitution policies can be created based on the following syntax. Clicking the + button opens the following window where substitution policies can be defined.



Syntax for font substitution files

Please use a simple text editor to edit or create the policy files and save them as UTF-8.
All values have to be tabulator-separated. Hash signs "#" at the beginning of a line means they are comments and will not be taken into account by the engine.

The following substitution modes are available:

SubstituteFirst

The first font of a list of font names is substituted by the following fonts. The order of the possible replacement fonts is regarded.
Notation:

```
SubstituteFirst<tab>font substituted<tab>fontname<tab>fontname<tab>fontname ...
```

Regular Expressions

It is possible to use RegEx in SubstituteFirst lines. Example:

```
SubstituteFirst   '(.*)Bold(.*)Italic'
```

```
'Arial(.*)Bold(.*)Italic'   '\1Regular'
'\1'
```

Any font name that uses "Bold" and "Italic" with any additional characters will be replaced by any font that uses "Arial", "Bold" and "Italic" (with any additional characters in between of these phrases). If that cannot be found any font that has the same characters at the beginning (indicated by \1) and then "Regular" and if that is also not present any font that only has the same beginning.
Font substitution with RegEx will be used if normal substitution (without RegEx) was not successful.

Substitute All

This mode allows to define a list of fonts in which all fonts can substitute each other.
If any of the fonts of a list is missing it will be substituted with the next font in the list.
If this next font is also missing the previously listed font name will be searched.
If that is also missing, the font after the next font is tried and so on.
Notation:
`SubstituteAll<tab>fontname<tab>fontname<tab>fontname ...`

Regular Expressions

In SubsituteAll RegularExpressions are not supported for performance reasons.



The engine ignores all characters other than a-z, A-Z and 0-9. That means "Helvetica Bold", "Helvetica-Bold" and "HelveticaBold" are all the same. It might still make sense to have them all in a configuration file for documentation and clarity (and that is the reason why they are all present in the predefined fontsubsitution.cfg).

Example

The entry below defines that Arial is substituted with Helvetica and Helvetica with Arial.
At the end of this example, there are further entries allowing more comprehensive font substitutions.
To use them the '#' at the beginning of the line needs to be removed.

```
SubstituteAll        Arial        ArialMT        Helvetica        Helvetica
Neue        Futura        Helvetica Neue        Microsoft Sans Serif        MS
PGothic        Trebuchet MS        Verdana
SubstituteAll        Arial Bold        Arial-Bold        ArialMT,Bold        Hel-
vetica Bold        Futura-Bold        Futura Bold        Helvetica-Bold
SubstituteAll        Arial Italic        Arial-Italic        ArialMT,Italic
Helvetica Italic        Helvetica-Italic        Futura-Italic        Futura Ital-
ic        Arial Oblique        Arial-Oblique        Helvetica Oblique        Hel-
vetica-Oblique        Futura-Oblique        Futura Oblique
SubstituteAll        Arial Bold Italic        Arial-Bold-Italic        Ar-
ialMT,BoldItalic        Helvetica Bold Italic        Futura-Bold-Italic        Fu-
tura Bold Italic        Arial Bold Oblique        Arial-Bold-Oblique        Arial-
BoldOblique        Helvetica Bold Oblique        Helvetica-BoldOblique        Futu-
ra-Bold-Oblique        Futura Bold Oblique
SubstituteAll        Arial Black        Arial-Black        Helvetica Black
Helvetica-Black        Futura-Bold        Futura Bold        Arial-Bold
SubstituteAll        Arial Narrow        Helvetica Narrow
SubstituteAll        Arial Narrow Bold        Helvetica Narrow Bold        Arial
Narrow Fett        Helvetica Narrow Fett
SubstituteAll        Arial Narrow Bold Italic        Helvetica Narrow Bold Ital-
ic        Arial Narrow Fett Kursiv        Helvetica Narrow Fett Kursiv
SubstituteAll        Times-Roman        Times        Times New Roman        Gara-
mond
SubstituteAll        Times-Bold        Times-Roman Bold        Times Bold
Times New Roman Bold        Times Fett        Times New Roman Fett
SubstituteAll        Times-Italic        Times-Roman Italic        Times New Roman
Italic        Times Italic        Times Kursiv        Times New Roman Kursiv
SubstituteAll        Times-Bold-Italic        Times-Roman Bold Italic        Times
Bold Italic        Times New Roman Bold Italic        Times Fett Kursiv
Times New Roman Fett Kursiv
SubstituteAll        Courier        Courier New        Courier Std New        Luci-
da Console        MS Gothic
SubstituteAll        CourierStd-Bold        Courier Std Bold        Courier
```

```
Bold        Courier-Bold        Courier New Bold        Courier New-Bold
SubstituteAll        CourierStd-Oblique        Courier Std Oblique        Courier-
Std-Italic        Courier Std Italic        Courier Italic        Courier-Ital-
ic        Courier New Italic        Courier New-Italic        Courier
Oblique        Courier-Oblique        Courier New Oblique        Courier New-
Oblique
SubstituteAll        CourierStd-BoldOblique        Courier Std Bold Oblique
Courier BoldItalic        Courier-BoldItalic        Courier New Bold Italic
Courier New-Bold-Italic        Courier BoldOblique        Courier-
BoldOblique        Courier New Bold Oblique        Courier New-Bold-Oblique
#SubstituteFirst        '(.*)Regular'        Arial        Helvetica        '\1Regu-
lar'        '\1'
#SubstituteFirst        '(.*)Medium'        Arial        Helvetica        '\1Regu-
lar'        '\1'
#SubstituteFirst        '(.*)Bold'        'Arial(.*)Bold'        'Helvetica(.
*)Bold'        '\1Regular'        '\1'
#SubstituteFirst        '(.*)Italic'        'Arial(.*)Italic'        'Helvetica(.
*)Italic'        '\1Regular'        '\1'
#SubstituteFirst        '(.*)Black'        'Arial(.*)Black'        'Helvetica(.
*)Black'        '\1Regular'        '\1'        Arial(.*)Bold'        'Helvetica(.
*)Bold'
#SubstituteFirst        '(.*)Bold(.*)Italic'        'Arial(.*)Bold(.*)Ital-
ic'        'Helvetica(.*)Bold(.*)Italic'        '\1Regular'        '\1'
#SubstituteFirst        '(.*)Bold(.*)'        'Arial(.*)Bold'        'Helvetica(.
*)Bold'        '\1Regular'        '\1'
#SubstituteFirst        '(.*)Italic(.*)'        'Arial(.*)Italic'        'Helveti-
ca(.*)Italic'        '\1Regular'        '\1'
#SubstituteFirst        '(.*)'        '\1Regular'        '\1Medium'
'\1'        Arial        Helvetica
```

Syntax for glyph substitution in font substitution files

For font embedding the font needs to have glyphs for all character codes in the PDF.
Some PDFs use malformed character names in which case font embedding will fail. The `SubstituteGlyphName` entry allows for defining mapping of malformed character names to correct character names.
The first entry is the correct character name, after that an unlimited number of malformed entries can be specified.
As a result all malformed character names will be corrected

in the PDF and the result will be taken into account for font embedding.

```
SubstituteGlyphName <tab> nonbreakingspace
<tab> no_break_space <tab> non_break-
ing_space
```

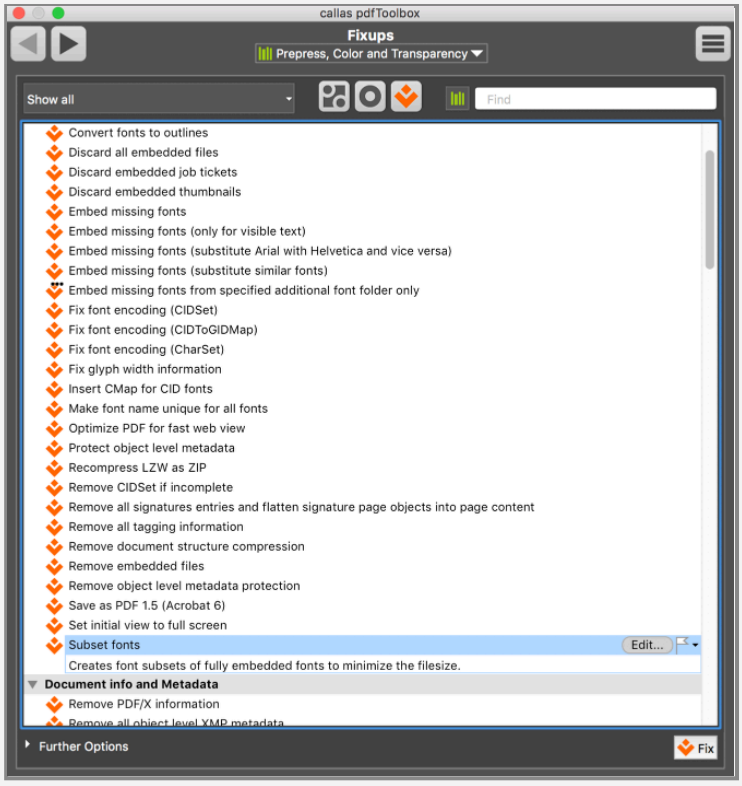
```
SubstituteGlyphName <tab> Hungarumlaut
<tab> hungarumlaut__now_576
```

11.3 Subset Fonts

Often the fonts used are completely embedded in a PDF - i.e. the font embedded in the PDF is included with all available characters, although only a selection may be used.

The Fixup "Subset fonts" reduces the font in the PDF to the characters actually used. This procedure can often also reduce the file size by a considerable amount. In the attached, very simple example, for example, from 70 kB to only 19 kB.

The fonts are completely restructured, which often also corrects some font problems.



MyriadPro_FullyEmbedded.pdf

11.4 Unembed all fonts

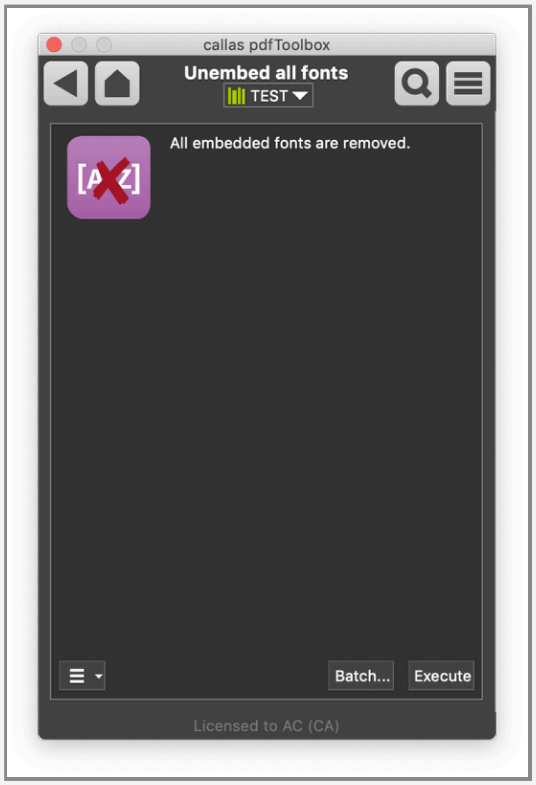
A PDF file does not necessarily have to contain all used fonts.

If a font is used inside a PDF for a text, it is referenced by its name. The font can either be contained - then the contained font is usually used when the PDF is displayed, or cannot be contained - then the respective PDF viewer can search for a font with the referenced name on the system, and (if such a font is not found) use another font for displaying the text.

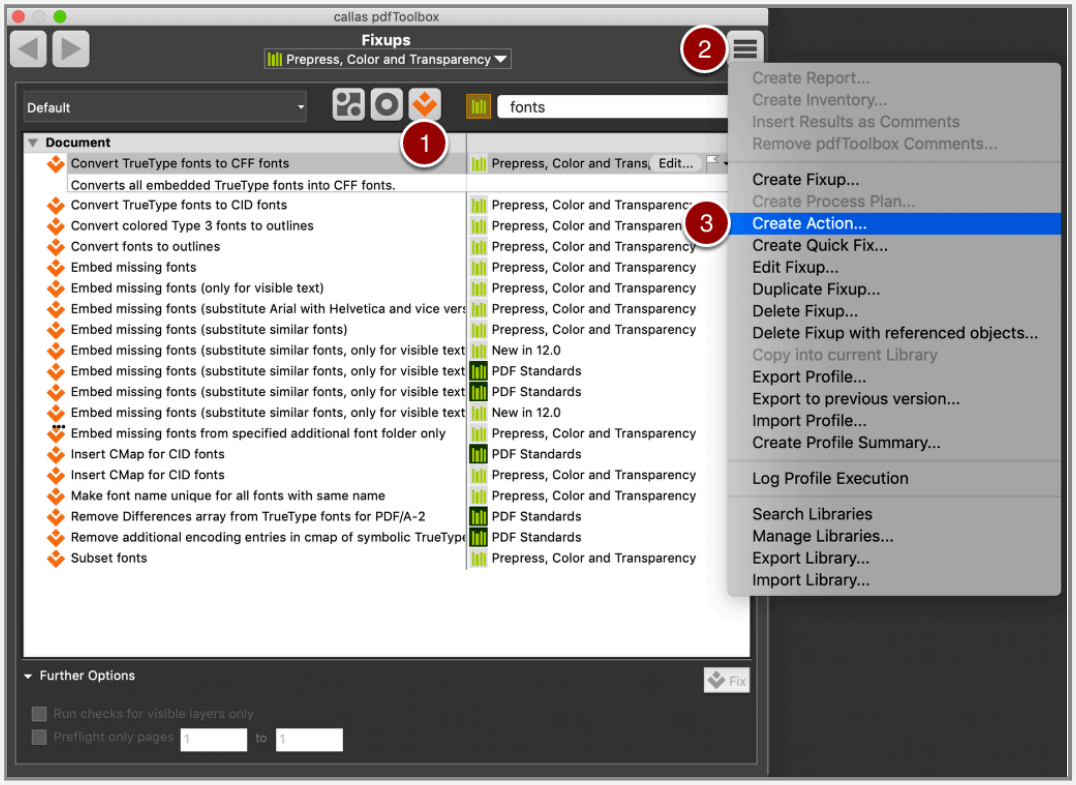
Starting pdfToolbox 13, it is now possible to remove all embedded fonts from a PDF file.

Switchboard Action: Unembed all fonts

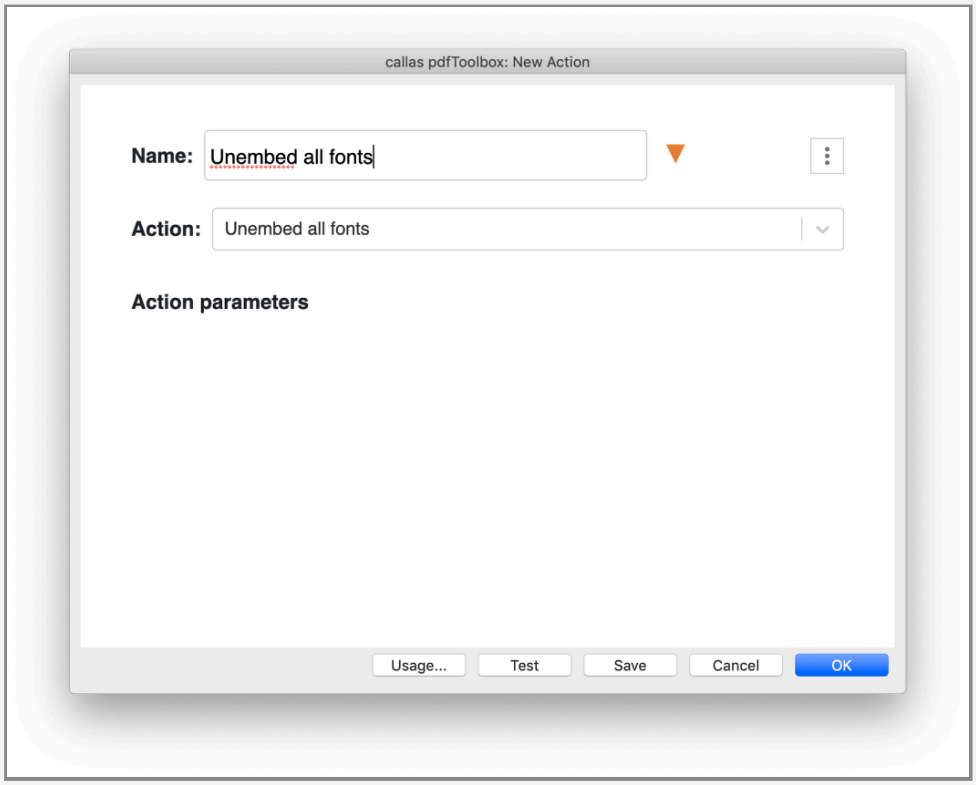
The Switchboard action 'Unembed all fonts' has no parameters and will simply remove all embedded fonts from a PDF file.



Action in Process Plans



1. Go to 'Fixups'
2. Click on the upper right hand side 'Menu'
3. Create new Action


callas pdfToolbox: New Action
Name:
Unembed all fonts
Action:
Unembed all fonts
Action parameters
Usage...
Test
Save
Cancel
OK

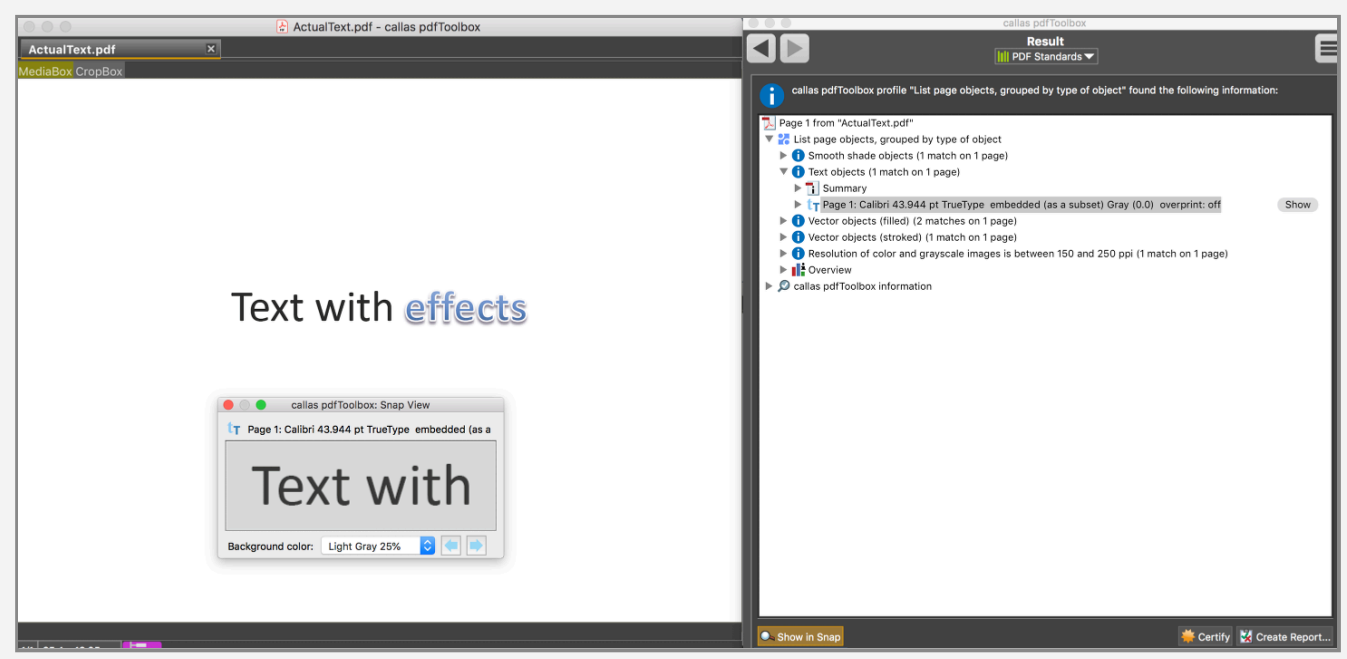
11.5 Convert fonts to outlines

After converting fonts to paths, the output system no longer needs to load fonts, since the glyphs only exist as vectors in the PDF.

Although correctly embedded fonts in a PDF should generally not cause any problems, it is possible that only text previously converted into paths is required for the transfer of the files. This is still common in packaging and label printing. Under certain circumstances, a font may also prohibit embedding based on the license restrictions it contains.

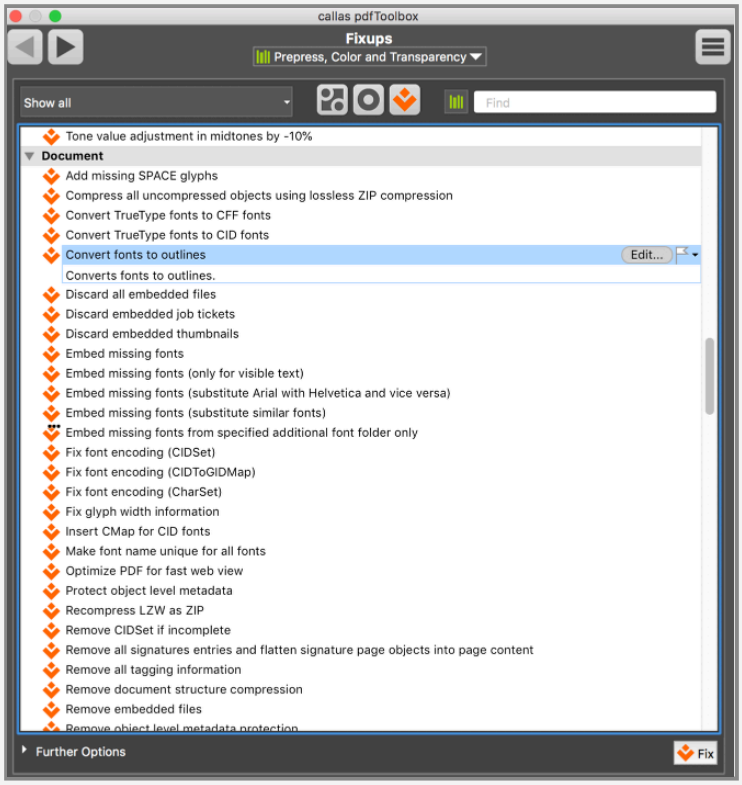
In order to convert the fonts into paths, a predefined Fixup is provided.

Text exists in the document as text objects



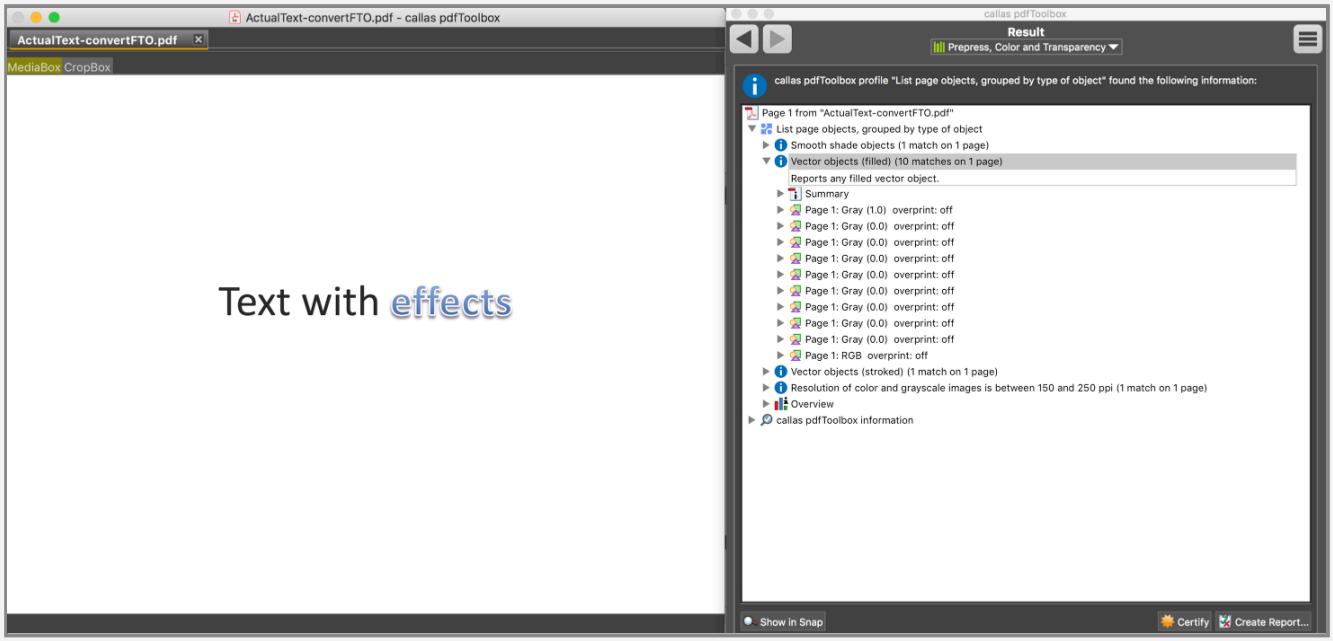
A Check for the contained objects shows that the black text exists as text objects.

Call Fixup



Open the Fixup window, search for "Convert fonts to Outlines" and execute the Fixup.

Text converted to vector format



A new analysis of the updated PDF no longer shows any text objects.

ActualText.pdf

ActualText-outlined.pdf

11.6 Convert Type 3 fonts to outlines (10.2)

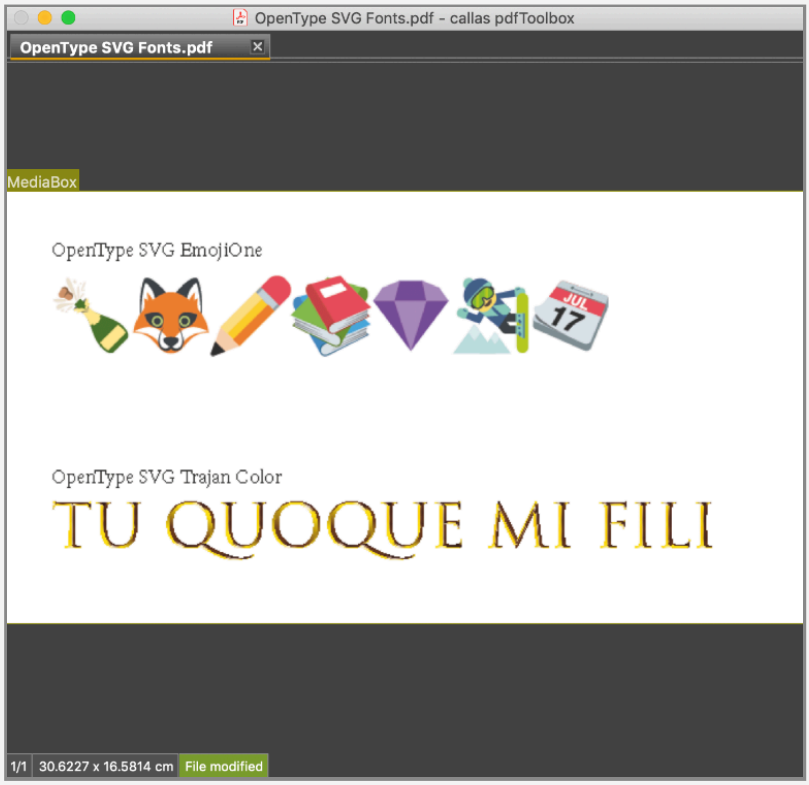
Compared to other fonts that can be used in PDF, Type 3 fonts represent some kind of a speciality.
Glyphs in Type 3 fonts of type 'd0' can contain any graphic object such as lines, areas or images, which in turn can use all properties and resources possible in PDF, such as colors, transparency, shading or fill patterns.

However, there are also Type 3 fonts of the type 'd1', which - like other fonts - only describe contours, whereby besides lines and areas also masks can be used. With both types of Type 3 fonts, however, it is not possible to optimize the font presentation, especially Hinting. A direct conversion into outlines is not possible with both types of Type 3 fonts, or only possible to a limited extent; instead, the existing graphic objects must be included as such in the page description.

Typical use cases for Type 3 fonts are, in addition to ordinary "monochrome" fonts, Emoji fonts (see illustration below) or generally "colorful" fonts with graphic properties varying within a glyph. Since 2018, this type of font has become more widespread because SVG fonts have been added to the OpenType format. These new possibilities are increasingly also supported in applications such as QuarkXPress or Adobe InDesign. SVG (*scalable vector graphics*) allows the use of almost all graphic possibilities as they are supported in PDF. Since SVG fonts cannot be used directly in PDF files, they are instead converted to the Type 3 format which is supported in PDF during PDF generation.

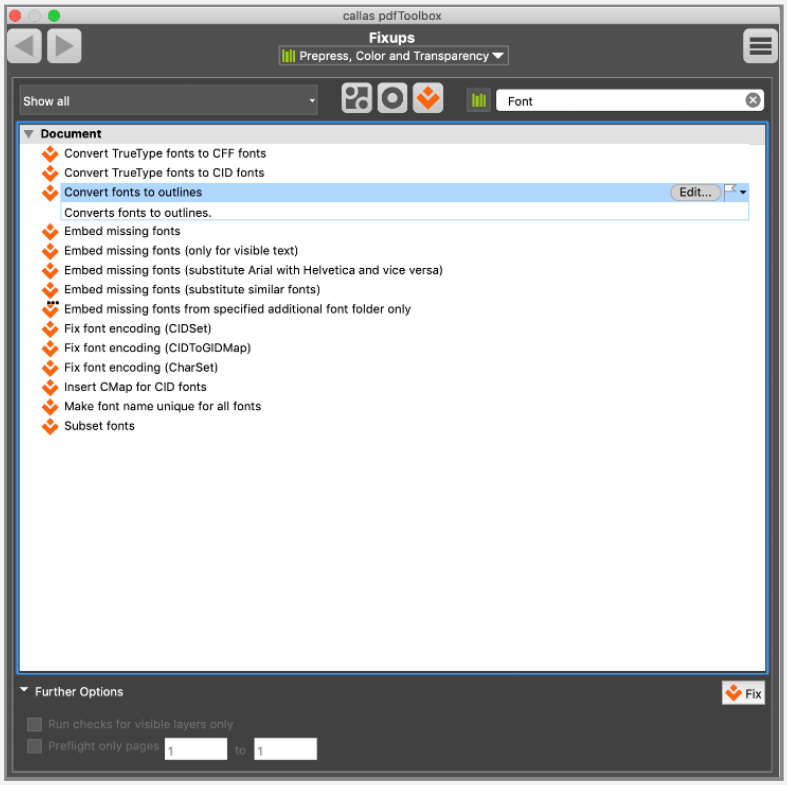
The support of SVG fonts and the Type 3 fonts generated from them must currently still be described as very mixed. For this reason, it may be worthwhile to convert Type 3 fonts into normal graphical content to avoid unpleasant surprises during output. It is not necessary to convert text in other fonts - which usually don't cause any problems - as well.

Starting with pdfToolbox 10.2, Type 3 fonts can also be converted to normal page content. Although the term "convert fonts to outlines" is not strictly accurate, the Fixup is called this way, because most users are most likely to look for it under this expression.



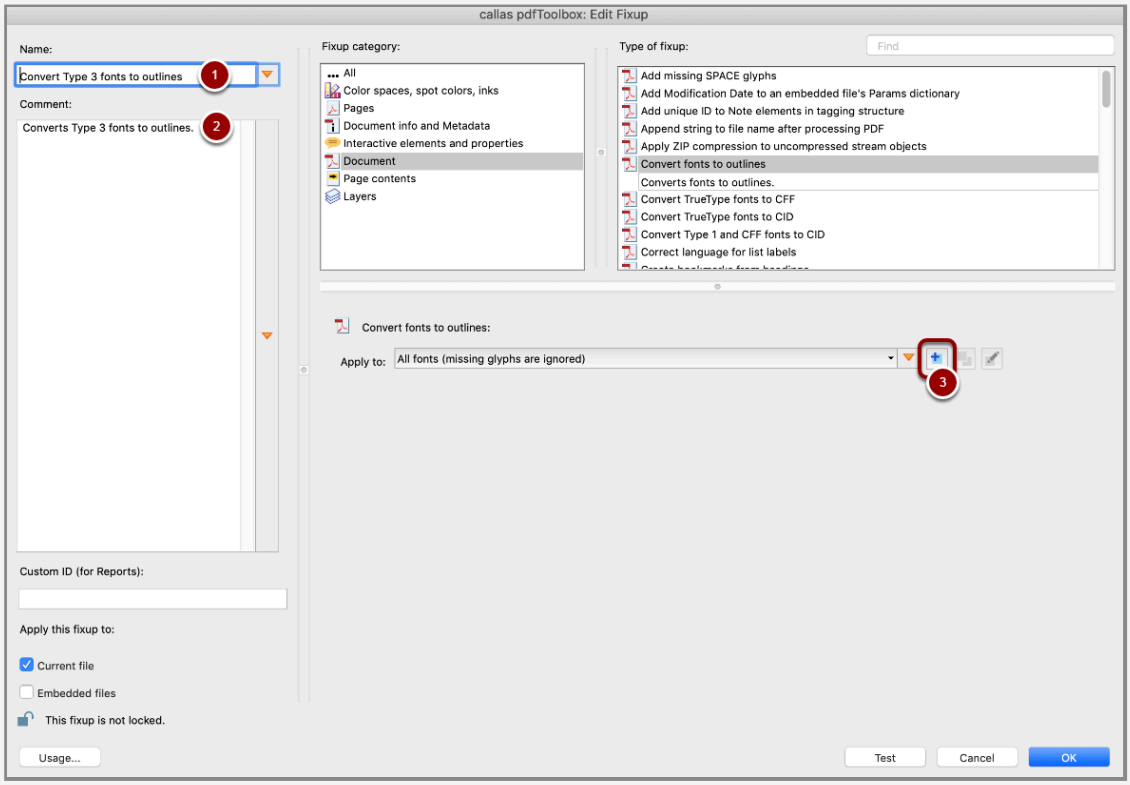
Adjusting the "Fonts to outlines" Fixup

The predefined Fixup "Convert fonts to outlines" will convert all fonts in the PDF to paths by default.



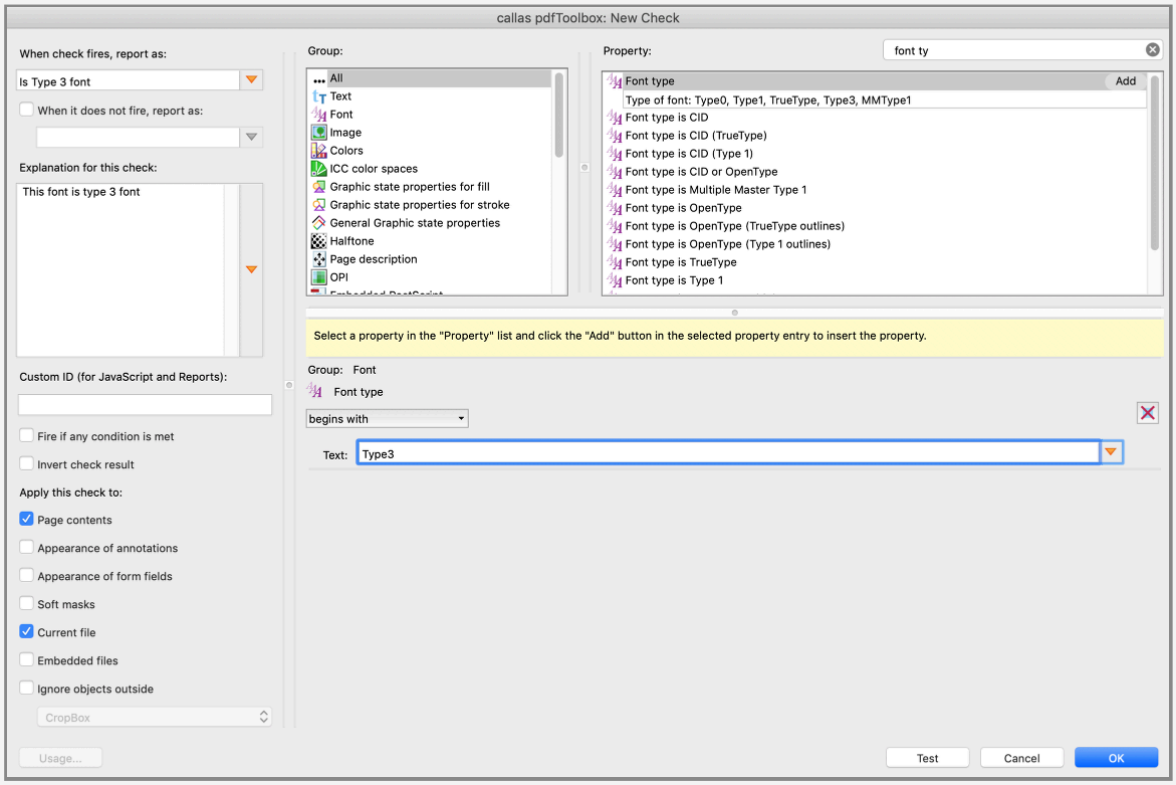
It is recommended to simply create a copy (via the menu button in the upper right corner - "Duplicate Fixup") of the existing Fixup, as shown below.

In the created copy you should first change the name (1) and comment (2).

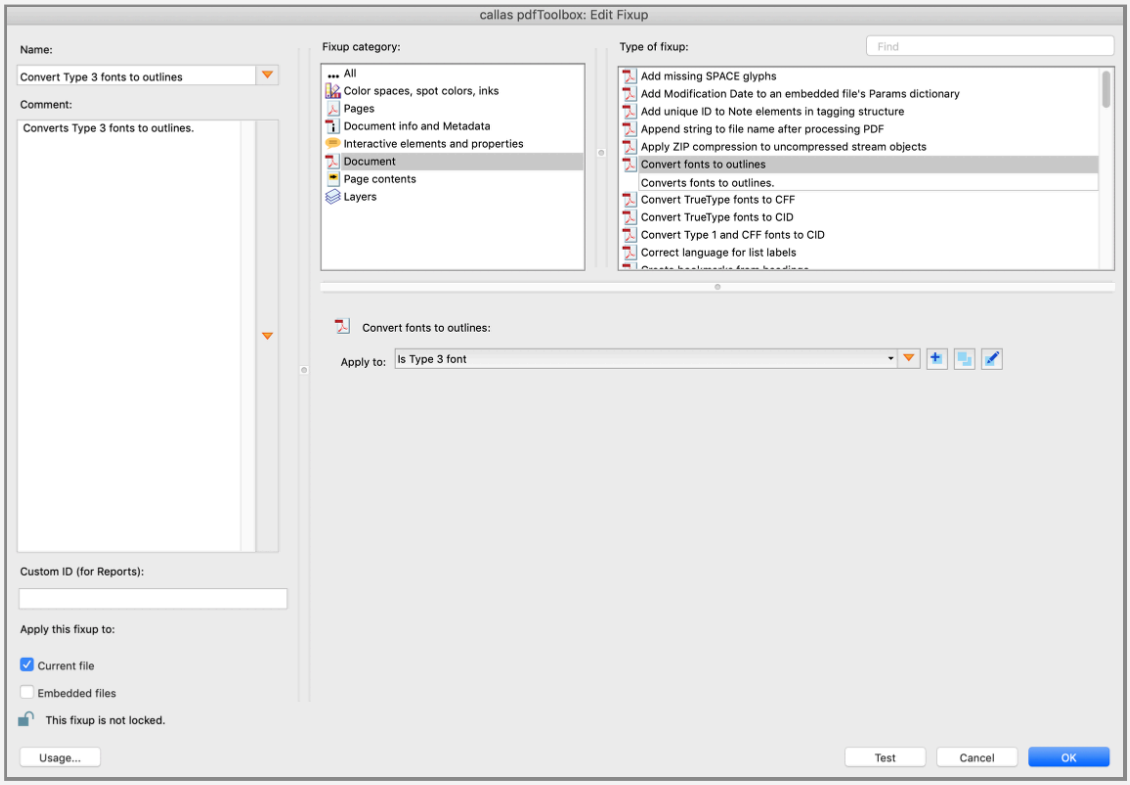


Then (3) create a new Check for the font "Type3", so that you can limit the correction to this font.

First you have to assign a suitable name and if necessary, a comment for this new Check. As property, search for "font", then enter "Type3" as value (the possible values are listed in the comment of the property) and select "begins with" as operator. The new Check can then be saved.



The new check is automatically preselected in the previously created Fixup (Convert Type 3 fonts to outlines) under "Apply to".



Preconfigured Fixups

Convert_Type_3_fonts_to_outlines.kfpx

As most problem-causing fonts are using colored Type 3 font, this Fixup can even be limited to those colored fonts using the following correction.

Convert_colored_Type_3_fonts_to_outlines.kfpx

Preconfigured Check to check if Type 3 font has all required resources

Type_3_font_has_all_required_resources.kfpx

This Check can be used with Acrobat Preflight:

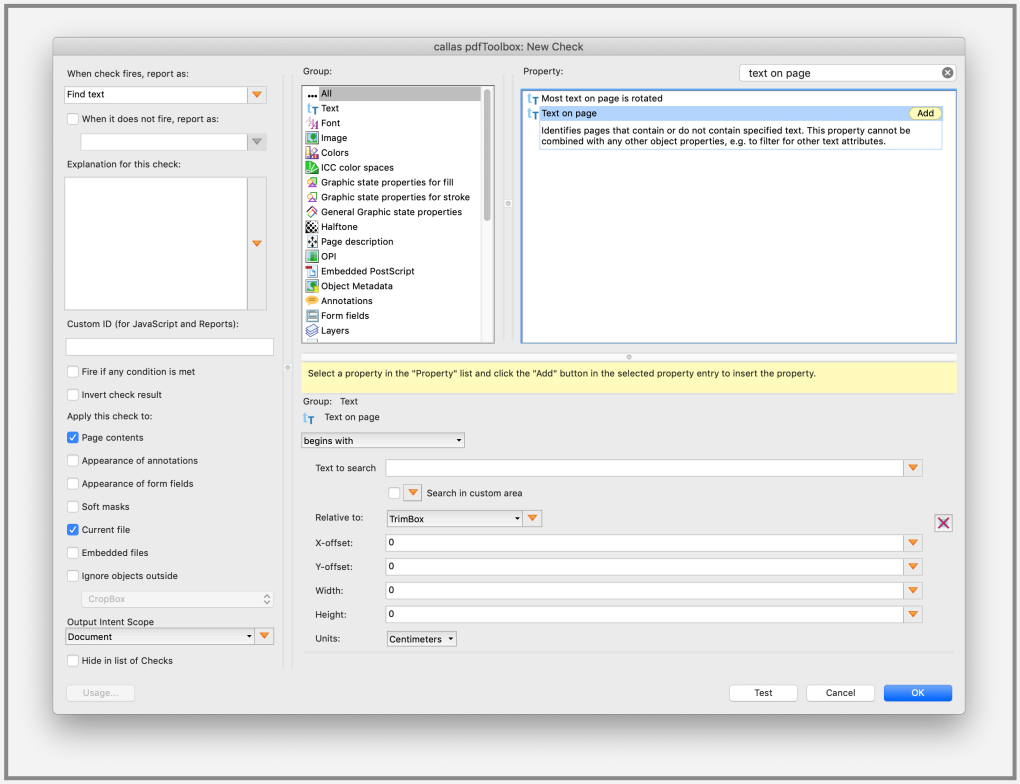
Type_3_font_misses_resources.kfp

Sample file

callas_"I_love_SVG_fonts".pdf

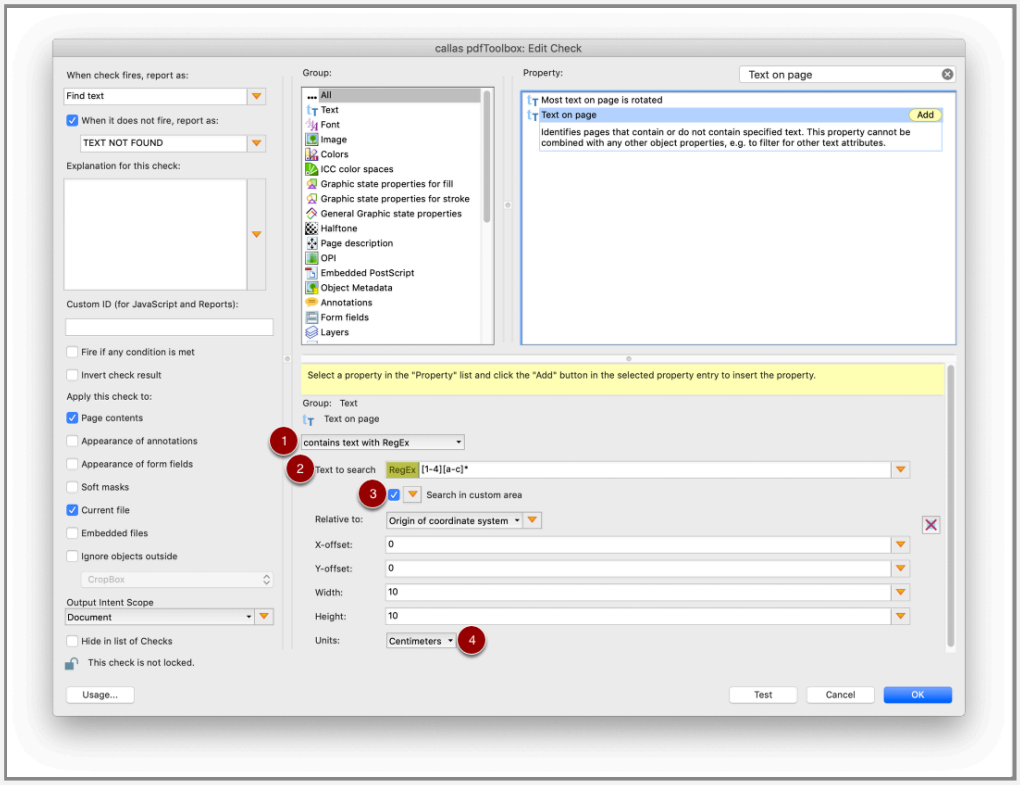
11.7 Search text

The Check property 'Text on page' identifies pages that contain or do not contain specified text. Please keep in mind that this property cannot be combined with any other object properties.



Text search using the Check property 'Text on page' works on the basis of the pdfToolbox Action --extracttext, which extracts text from a PDF file.

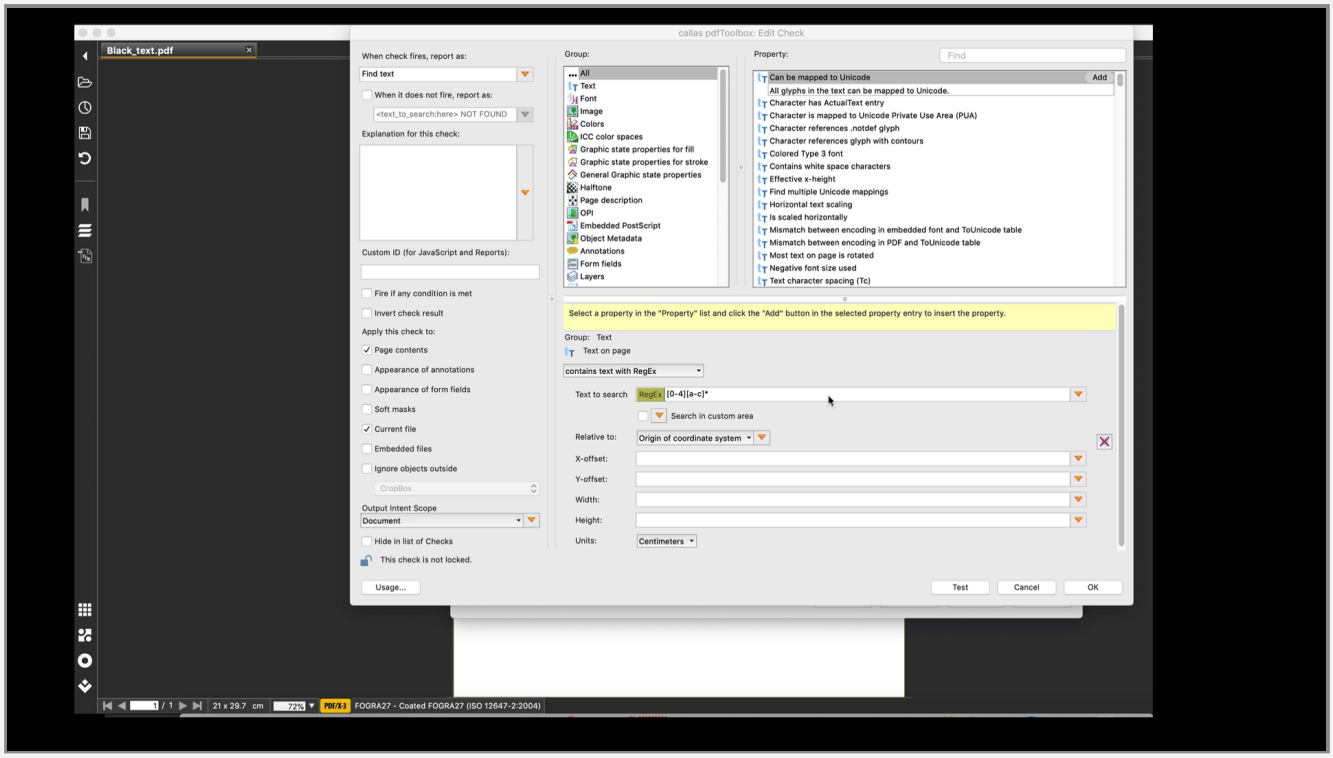
Usage



1. Matching criteria: Text can be searched based on different search criterion from the drop down, some examples below:
   - begins with
   - contains
   - contains text with Regex
   - ends with
   - equal to
   - matches with Regex
   - ...
2. Text to search: Based on the 'Matching criteria', text to be searched is to be entered (Regex in the screenshot below)
3. Search in custom area: Defines the position where text has to be searched. Important to note:
   - If you want to search on the whole page, "Search in custom area" must be deactivated)
   - Positive and negative numbers are allowed

- If a 0 (zero) is entered in "Width" or "Height", this is interpreted as "no value" and the orginal width and height of the pagebox remains

4. Units: Defines the 'unit' of the custom page dimensions in
   - Centimetres
   - Millimetres
   - Inches
   - Points
   - Picas

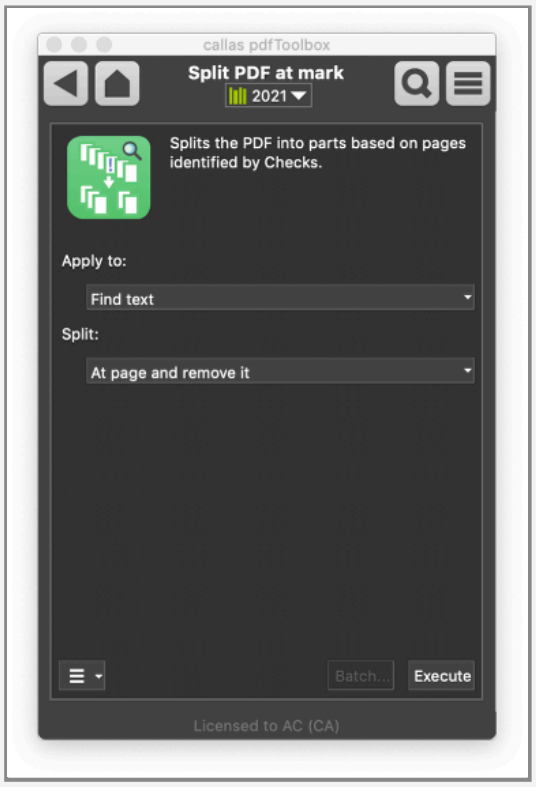


Careful consideration has to be made while opting from different operators which are used for different use cases. For example:

For the search word 'Black' on a PDF, a regex 'Ba*' and operator:

- 'Contains text with Regex': will give a hit (The word Black contains B followed by zero or more a's)
- 'Matches with Regex': will not give a hit because the regex doesn't entirely MATCH with the word 'Black'

This Check property can be used with the Action (and corresponding CLI parameter) 'Split PDF at mark'/--splitatmark

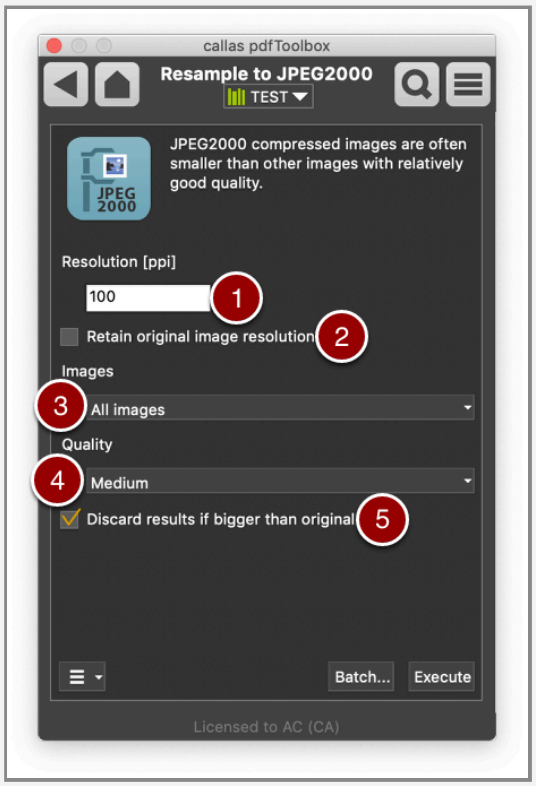


12. Images

12.1 Resample to JPEG2000

This Actions allows to resample images (changing the number of pixels in the image) to JPEG2000 with different set of parameters. JPEG2000 compressed images are often smaller than other images with relatively good quality.

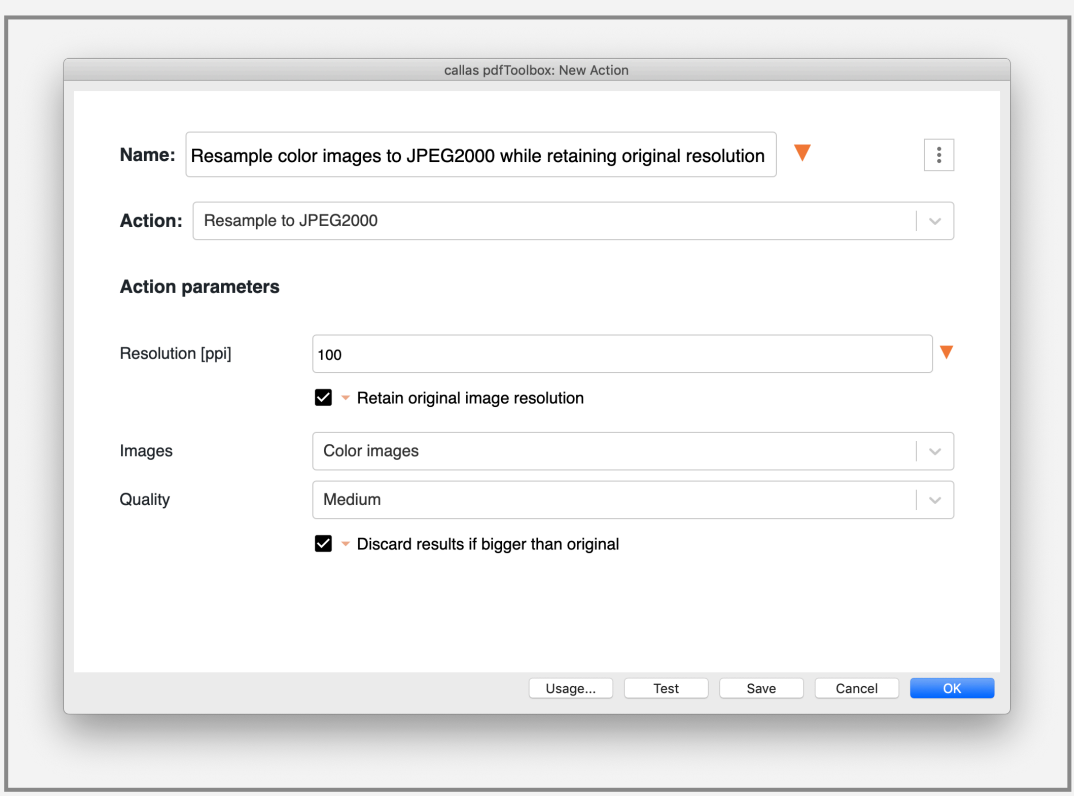
Switchboard Action



1. **Resolution**: In ppi. Default is 100
2. **Retain original image resolution**: Retains original image resolution and overwrites the resolution in the above field if checked.
3. **Images**: All images OR color images only OR Grayscale images only
4. **Quality**: Minimum OR low OR medium OR high OR maximum OR lossless
5. **Discard results if bigger than original**

Action in Process Plans

It is possible to use 'Resample to JPEG2000' Action in a Process Plan.



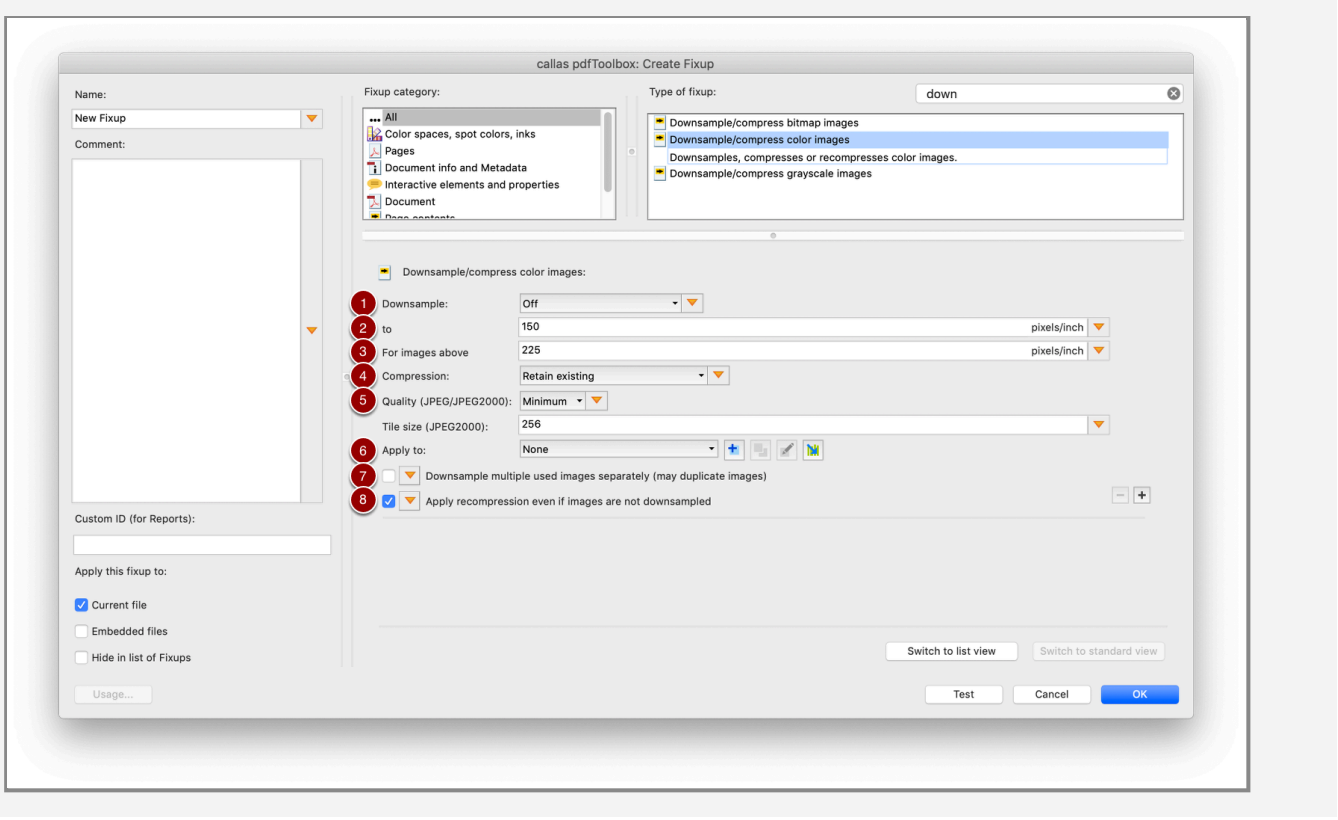
12.2 Upsample/downsample images

Sometimes it is required by a printer that all images in a PDF have at least a certain resolution. This can be difficult, especially if the low-res images are not available in a higher resolution, or the other way around.

With pdfToolbox you have the possibility to downsampling or upsampling images by using the following Fixup types:

- Downsample/compress images
- Upsample/compress images

Downsample/compress images



1. You can choose between 5 different downsampling methods. Which method should be used for recalculation depends on the images contained in the PDF and should therefore be determined individually (The quality after resampling with Lanczos is in most cases the best).
   The available downsampling methods are:
   - Average downsampling
   - Subsampling

- Bicubic downsampling
- Lanczos downsampling
- BlackmanSinc downsampling

2. "To": The target resolution in ppi
3. "For images above": Image resolutions that are above the specified threshold will be downsampled.
4. You can choose between different compressions:
   - Retain exising
   - Retain existing, but change quality
   - JPEG2000
   - JPEG
   - ZIP
5. "Quality": Allows for specifying the quality level in which JPEG objects are saved after the conversion.
   Note: The given value is interpreted as a percentage declaration between 0 and 100. The table below shows the possible values and the corresponding setting in Adobe Photoshop.
6. "Apply to": Here you can set a filter to exclude certain images from the resampling process.
7. "Downsample multiple used images separately": If an image file is contained multiple times in a document in different sizes or reflections, setting this check box will inspect each image reference separately and downsample it if necessary.
8. "Apply recompression even if images are not downsampled": If the checkbox is activated, all images contained in the document will be compressed with the selected compression.

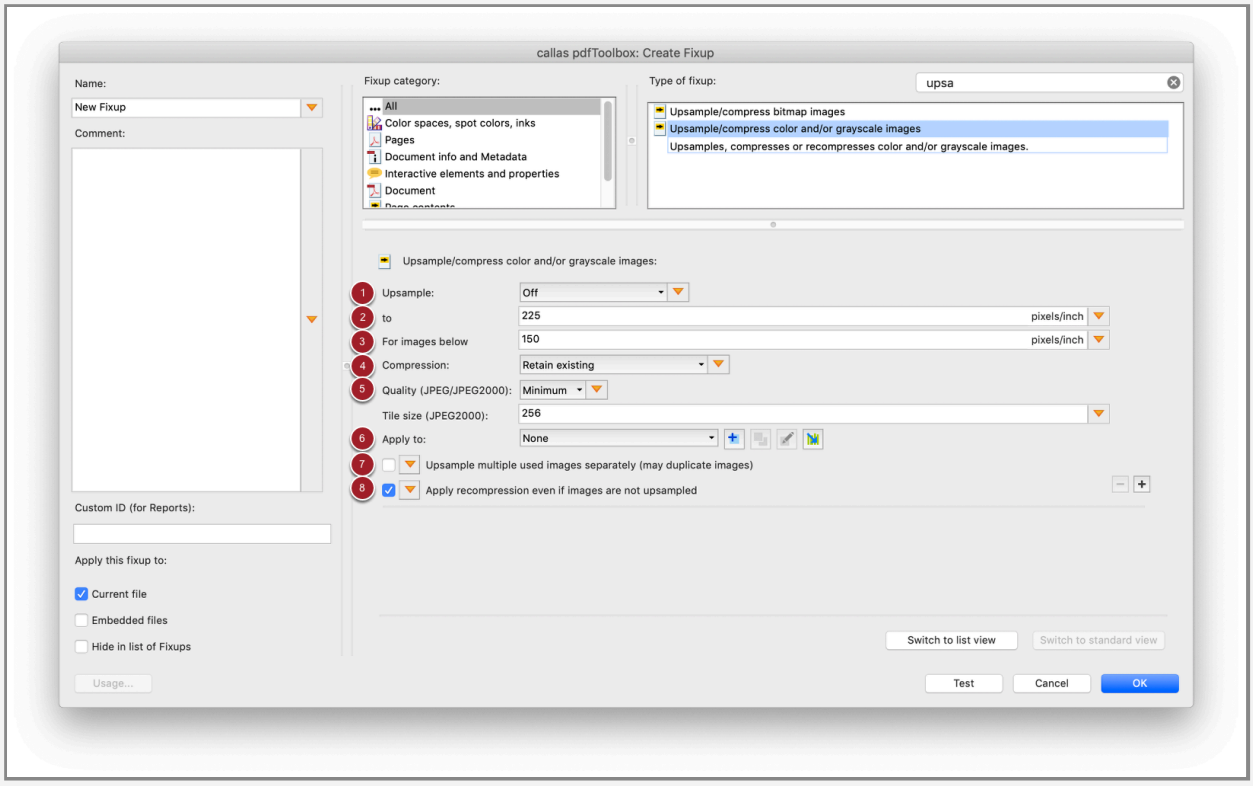
Possible values and corresponding setting in Adobe Photoshop

Adobe photoshop values	JPEG quality compression
20	Minimum
40	Low
60	Medium
80	High

Adobe photoshop values	JPEG quality compression
100	Maximum

Upsample/compress images

Upsampling images increases the pixel resolution of the image, but does not result in a visual improvement.



1. You can choose between 4 different upsampling methods. Which method should be used for recalculation depends on the images contained in the PDF and should therefore be determined individually (The quality after resampling with Lanczos is in most cases the best).
   The available upsampling methods are:
   - Average upsampling
   - Bicubic upsampling
   - Lanczos upsampling
   - BlackmanSinc upsampling

2. "To": The target resolution in ppi
3. "For images below": Image resolutions that are below the specified threshold are upsampled.
4. You can choose between different compressions:
   - Retain exising
   - Retain existing, but change quality
   - JPEG2000
   - JPEG
   - ZIP
5. "Quality": Allows for specifying the quality level in which JPEG objects are saved after the conversion.
   Note: The given value is interpreted as a percentage declaration between 0 and 100. The table above shows the possible values and the corresponding setting in Adobe Photoshop.
6. "Apply to": Here you can set a filter to exclude certain images from the resampling process.
7. "Upsample multiple used images separately": If an image file is contained multiple times in a document in different sizes or reflections, setting this check box will inspect each image reference separately and upsample it if necessary.
8. "Apply recompression even if images are not upsampled": If the checkbox is activated, all images contained in the document will be compressed with the selected compression.

More about the methods

When to use which method totally depends on the user. Here are some external links to:

- Lanczos resampling
- BlackmanSinc resampling: 'Blackman' window applied to a 'Sinc' filter for image resampling
- Bicubic resampling

13. Prepare PDF documents for production

13.1 Generate bleed from page content

With pdfToolbox, you are able to create bleed in a number of ways:

- **by mirroring page content as an image at the edges of the page**
- **by mirroring page objects at the edges of the page (without rasterisation)**
- **by repeating pixels at the edges of the page**
- **by stretching the content at the edges of the page**
- **in all cases bleed may be generated**
  - **just for the edges (and not for the corners)**
  - **for the edges and for the corners**

The "Generate bleed on page edges" can be configured very flexibly. The following article explains all the settings you can make in the Fixup.

How to create bleed by mirroring page content at the edges of the page

The following section contains a tutorial that shows how to generate bleed by means of mirroring the page content at the edges and corners of the page.



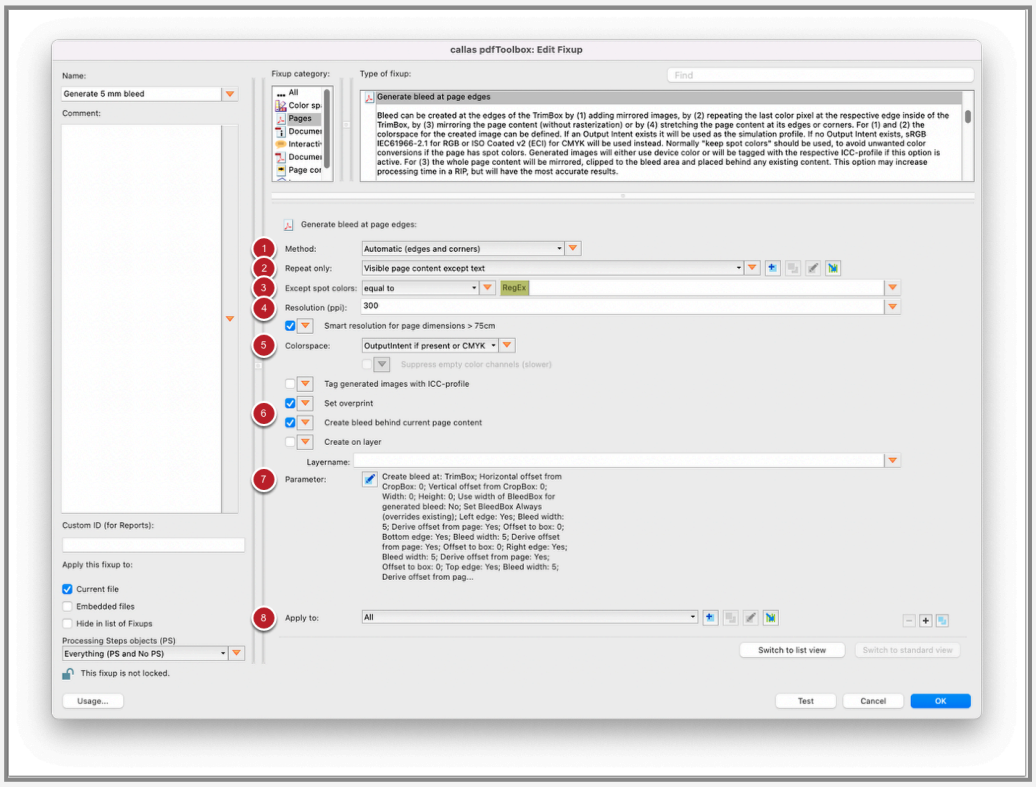
Bleed_Demo_file_1.pdf

Open the sample PDF and the "Generate bleed at page edges" dialog

Open the attached PDF "Bleed_Demo_file_1.pdf" (or your own document) and create a new Fixup.

- Fixup category: Pages
- Fixup type: Generate bleed at page edges

Configure the "Generate bleed at page edges" Fixup



1. Method

In the "**Method**" drop down list, you can choose between 5 different methods. In our example the method "**Automatic (edges and corners)**" is used.

The following methods can be used for the bleed generation:

Method	Explanation
Automatic (edges and corners)	**Available starting with pdfToolbox 16.** Should be used by default. Analyzes page edges of a PDF page and selects the best bleed generation method for each edge.
Mirror as image (edges)	Should be used, when images or patterns are close to the Trim Box. Using this method, the colorspace for the created im-

Method	Explanation
OR Mirror as image (edges and corners)	age needs to be defined.
Repeat last pixel as image (edges) OR Repeat last pixel as image (edges and corners)	With this method, only the last pixel is used for bleed generation. It can be used, if text objects are close to the Trim Box, to avoid that the text objects are included in the bleed. If images or patterns are close to the Trim Box, this method should not be used. Using this method, the colorspace for the created image needs to be defined.
Mirror page objects (edges) OR Mirror page objects (edges and corners)	More precise result than the "Mirror as image" method, because the bleed is not generated as an image. With this method all page objects are mirrored individually (vector objects are mirrored as vector objects etc.). However, this means that more document

Method	Explanation
	overhead is created. A colorspace is not defined, the mirrored page content simply keeps its existing colorspace(s).
Stretch border area as image (edges) OR Stretch border area as image (edges and corners)	Good solution when edges should not be visible. For bleed generation also the page content inside the Trim Box is changed, because it is used for stretching. This method will often be used for large format documents, as it is less noticeable at large page sizes. Using this method, the colorspace for the created image needs to be defined.

2. Repeat only

Here you can define what content you want to repeat in order to generate bleed. You can use a filter to limit the bleed generation to certain objects or properties.

3. Except spot colors

Specified spot colors are excluded. Here you can set, for example, that the Cutline is not taken into account for the bleed generation.

4. Resolution

Here you can specify a resolution in ppi for the generated bleed. 300 ppi is the default and should be used for print products.

The Checkbox **"Smart resolution for page dimensions > 75 cm"** enables auto adjustment of the resolution for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.

5. Colorspace

The bleed can be generated in a number of color spaces. In our example, the colorspace setting "CMYK and spot colors" is selected because the orange stripes have the spot color "HKS 10 N" and we want to keep the spot color in the bleed as well.

The following settings for the "Colorspace" parameter are available:

- OutputIntent if present or CMYK (CMYK will be used if no OutputIntent is present)
- CMYK (any spot colors will be converted to CMYK)
- CMYK, keep spot colors (A DeviceN color space is created so that spot colors are preserved. Should always be used when objects with a spot color are close to the Trim Box)
- RGB (sRGB IEC61966-2.1) , all colors that are neither sRGB nor plain RGB will be converted to sRGB
- Grayscale, all non-grayscale colors will be converted to grayscale

The checkbox **"Suppress empty color channels (slower)"** will only be enabled if **"CMYK, keep spot colors"** is selected as color space. If this checkbox is checked, the Device N color space of the generated bleed will contain only the colors that appear in the rendered image. Using this option may slow down processing.

6. Checkboxes

The following checkboxes are available:

Tag generated images with ICC-profile: If a document does not contain an output intent, the bleed can be tagged with the default ICC profile used for rendering.

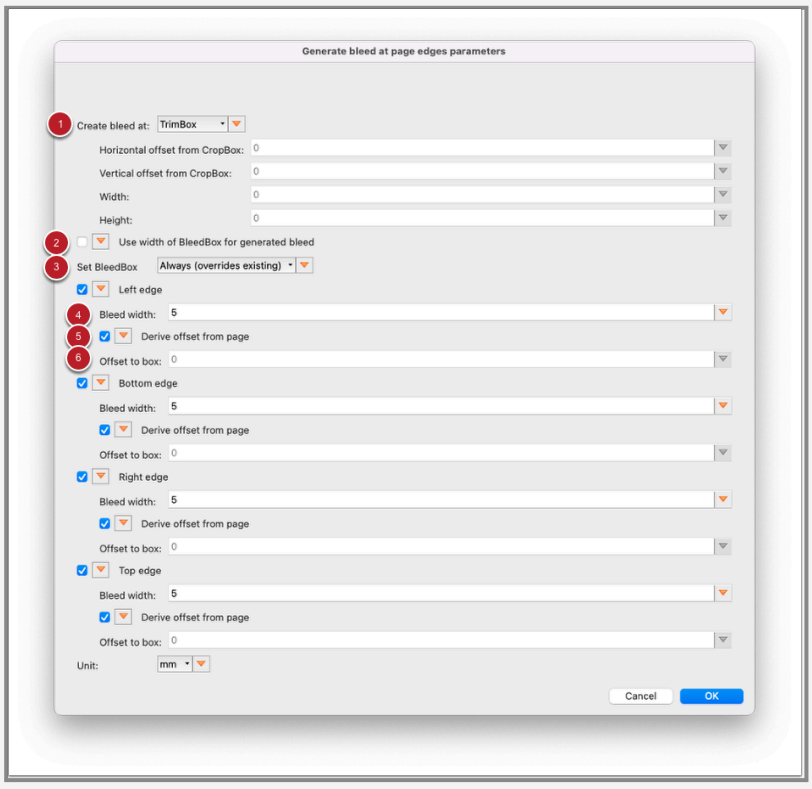
Set overprint: This checkbox should remain unchecked in most cases. It should only be activated if a cut contour is included in the document. By activating the checkbox, the new bleed will be set to overprint so that the cut contour is not covered.

Create bleed behind current page content: This checkbox should only be activated if bleed already exists in the document and you just want to extend it.

Create on layer: The bleed can be created on a new layer.

7. Parameters

Click on the edit button (pen icon) to configure the desired bleed geometry in a separate "Generate bleed at page edges parameters" dialog:



1. **Create bleed at**: Here you can select which page geometry box shall be used as source for the edges used for bleed generation; usually the TrimBox is used here (if available),

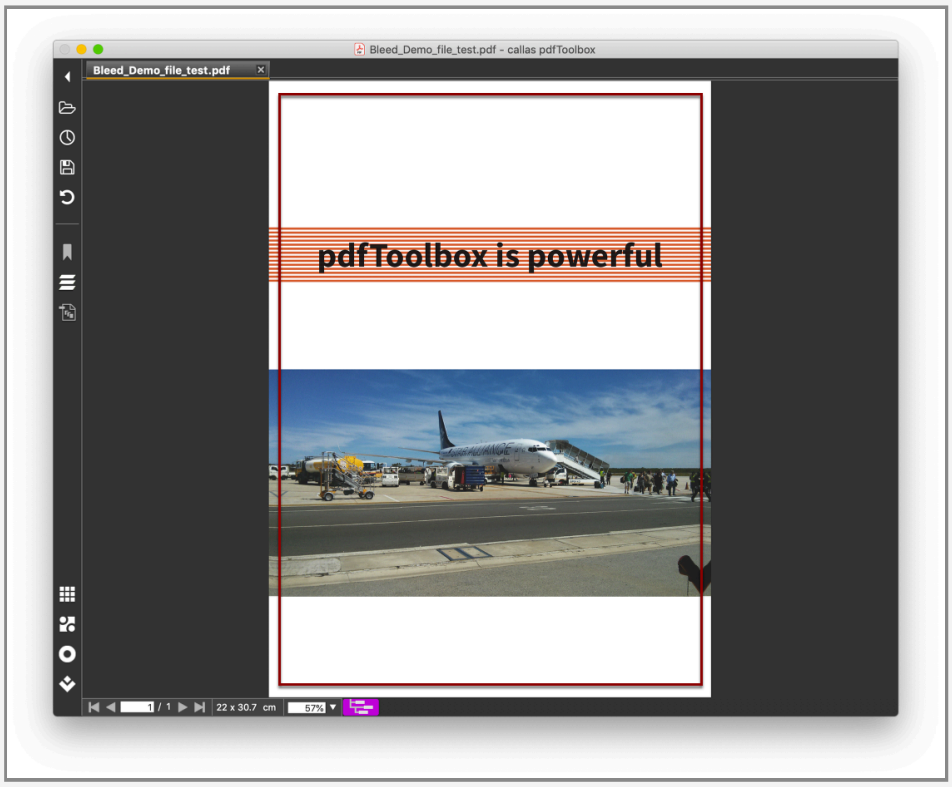
or the CropBox. It is also possible to define a "CustomBox" in which case the four parameters just below this option are enabled, and suitable values can be entered.

2. **Use width of BleedBox for generated bleed:** If the document already contains a BleedBox that only needs to be filled with content, this checkbox can be activated. Thus, not the value specified under "Bleed width" is used but the current width of the BleedBox (the specified offset is taken into account).
3. **Set BleedBox:** Here you can specify whether the BleedBox should be set again after bleed generation. If the BleedBox is already correctly defined in the document, the setting "ignore" can be used. If there is no bleed in the document yet, "Always (override existing)" should be set, or if there is no BleedBox at all, select "If missing".
4. **Bleed width**: The "amount" of bleed to generate.
5. **Derive offset from page: Since pdfToolbox 15** it is possible to automatically determine if an offset is needed. If the existing page content does not go straight to the TrimBox (ends close to the TrimBox), pdfToolbox will detect this and go inside the TrimBox to the edge where the page content actually ends and generate the bleed at that position. This method avoids thin white lines between the generated bleed and the page content. And it also works if there is already an existing bleed that is not big enough. pdfToolbox will detect the existing bleed and complete the missing bleed by generating the new bleed after the existing one.
6. **Offset to box**: Up to pdftoolbox 15 it is possible to set a fixed offset that will always be used for bleed generation to adjust pages where the page content ends just before the TrimBox. A negative offset value moves the source area for the bleed to the center of the page, a positive value moves the area to the outside of the page. A negative offset is useful when the page content ends just before the page edges, leaving a thin visible white line after the bleed is added. A small offset (such as -0.2 mm) can be specified for this use case.

8. Apply to

Here you can use a filter to specify to which pages the bleed generation is applied.

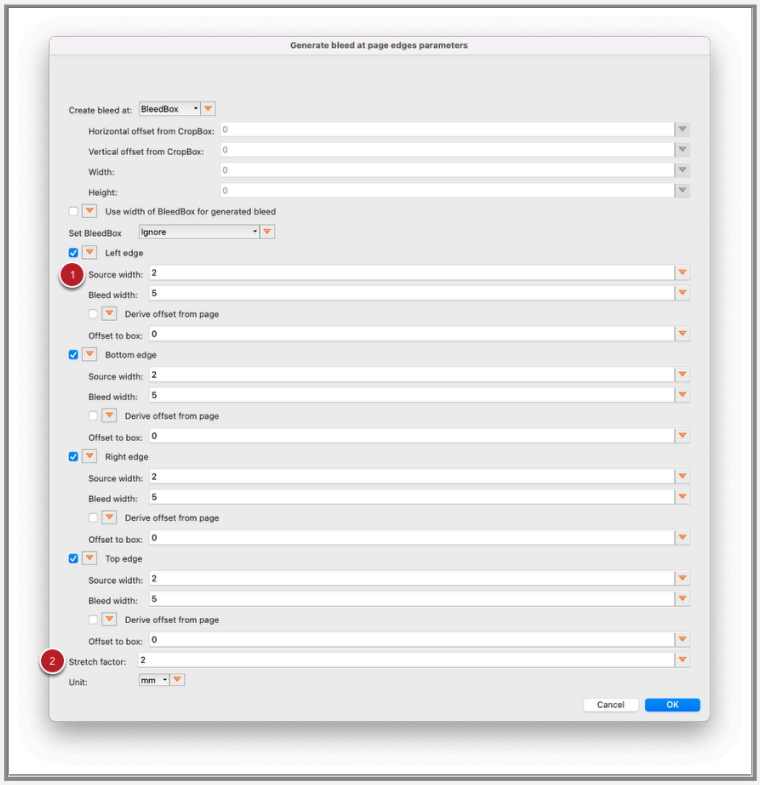
Review the processed PDF file in pdfToolbox



5 mm bleed is added on all sides of the PDF file by mirroring the content as an image.

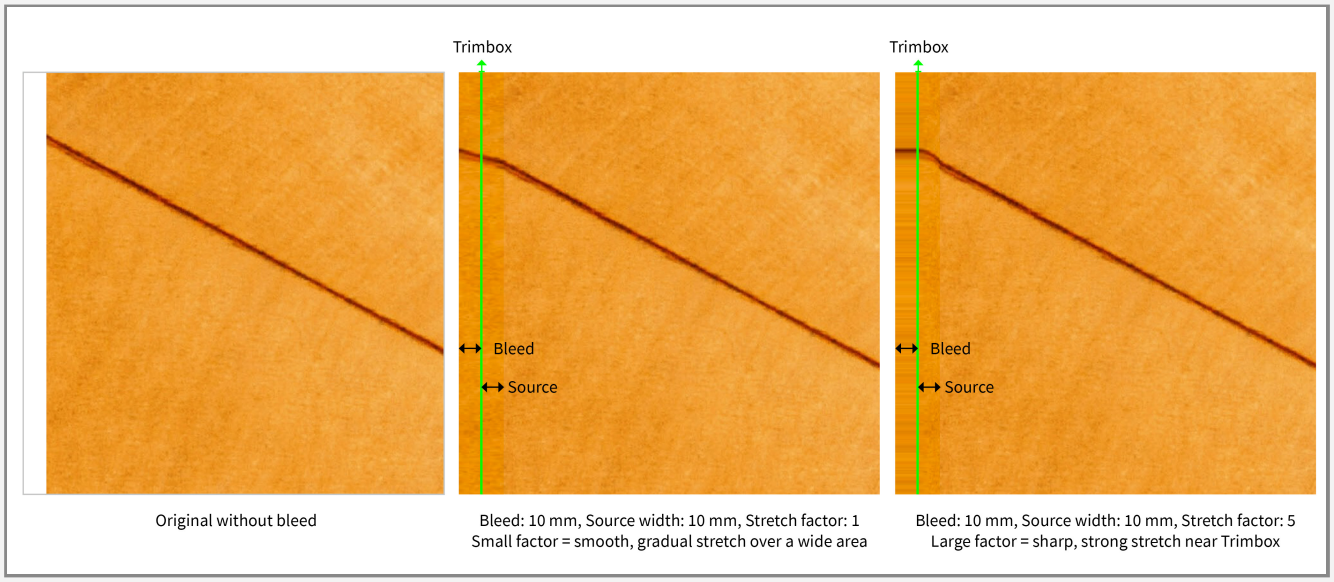
Additional parameters for "Stretch border area as image"

In case the "**Stretch border area as image**" mode is used for generating bleed from page content, two additional parameters are shown that need to be configured:



1. **Source width**: Width of the area within the TrimBox to be used for stretching. This source area is converted into an image that is then stretched to generate the bleed.
2. **Stretch factor**: This factor specifies the exponential function to calculate an increasing stretch effect starting from the inner edge of the Source width. The same stretch factor is used for all four sides. The slope of the exponential "stretch curve" is controlled by the stretch factor:

    1. **Small stretch factor:** The difference between the area outside the Trimbox and inside the Trimbox is smaller, resulting in a smoother transition. However, the stretching effect inside of the Trimbox is stronger. This is illustrated in the middle image below.
    2. **Large stretch factor:** The difference between the area outside the Trimbox and inside the Trimbox is larger, resulting in a sharper transition near the Trimbox. The original content inside the Trimbox remains closer to its original appearance. This is illustrated in the right image below.



Original without bleed

Bleed: 10 mm, Source width: 10 mm, Stretch factor: 1
Small factor = smooth, gradual stretch over a wide area

Bleed: 10 mm, Source width: 10 mm, Stretch factor: 5
Large factor = sharp, strong stretch near Trimbox

13.2 Generate bleed for irregular shapes

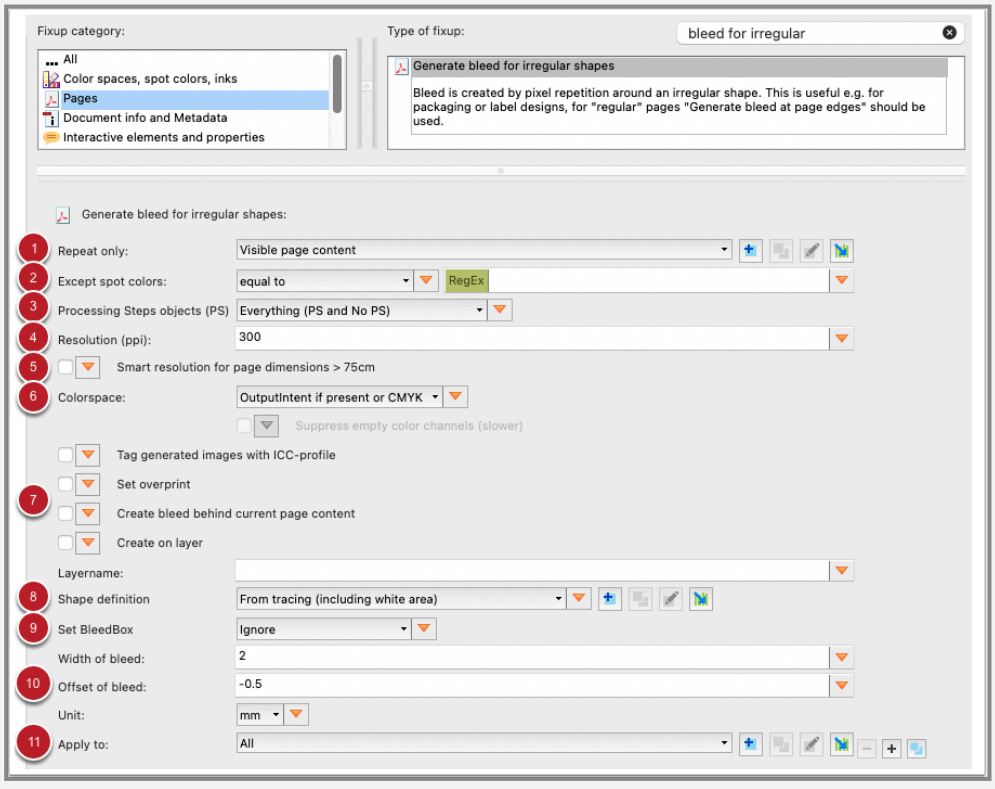
In pdfToolbox 10, you were able to create bleed for 'regular' pages by various methods like:

- Adding mirrored images
- Repeating the last color pixel at the edges inside Trimbox
- Mirroring the page content

pdfToolbox 11 introduced a Fixup for generating bleed for irregular pages. Using the Fixup 'Generate bleed for irregular shapes', you can create a shape from the settings specified in the "Shape definition" setting. The Pixel Repetition method is always used here. It uses the last pixel of a PDF document to generate the bleed.

A predefined Fixup is available in the "Prepress, Color and Transparency" library: "Generate 5mm bleed at dieline (Processing Steps)".

Fixup "Generate bleed for irregular shapes"



1. Repeat only

Here you can specify which content you want to repeat to generate the bleed. Please note: The Fixup always uses Pixel Repetition to generate the bleed. In the screenshot above, the last pixel of the 'Visible page content' would be repeated.

2. Except spot colors

Specified spot colors are excluded. Here you can set, for example, that the Cutline is not taken into account for the bleed generation.

3. Processing Steps objects (PS)

Here you can specify whether or not objects on a Processing Steps layer should be taken into account for the bleed gerneration or not.

4. Resolution (ppi)

Here you can specify a resolution in ppi for the generated bleed. 300 ppi is the default and should be used for print products.

5. Smart Resolution for page dimensions > 75 cm

Enables auto adjustment of the resolution for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.

6. Colorspace

The bleed can be generated in a number of color spaces. The following settings for the "Colorspace" parameter are available:

- OutputIntent if present or CMYK (CMYK will be used if no OutputIntent is present)
- CMYK (any spot colors will be converted to CMYK)
- CMYK, keep spot colors (A DeviceN color space is created so that spot colors are preserved. Should always be used when objects with a spot color are close to the Trim Box)
- RGB (sRGB IEC61966-2.1) , all colors that are neither sRGB nor plain RGB will be converted to sRGB
- Grayscale, all non-grayscale colors will be converted to grayscale

The checkbox **"Suppress empty color channels (slower)"** will only be enabled if **"CMYK, keep spot colors"** is selected as color space. If this checkbox is checked, the Device N color space of the generated bleed will contain only the colors that appear in the rendered image. Using this option may slow down processing.

7. Checkboxes

Tag generated images with ICC-profile: If a document does not contain an output intent, the bleed can be tagged with the default ICC profile used for rendering.
Set overprint: This checkbox should remain unchecked in most cases. It should only be activated if a cut contour is included in the document. By activating the checkbox, the new bleed will be set to overprint so that the cut contour is not covered.
Create bleed behind current page content: This checkbox should only be activated if bleed already exists in the document and you just want to extend it.
Create on layer: The bleed can be created on a new layer.

8. Shape definition

Define the configuration of the shape for creating the bleed. 4 predefined "Shape Definitions" are available in the drop-down list, which use the rendered appearance of the page content to generate the bleed around it (you also have the opportunity to configure your own custom shape definition).

- **"From tracing page content (including white areas)"**: White areas are considered to be part of the rendered page content (not transparent areas that appear to be

white because they show through the white background). Bleed is created only at the outer boarders of a shape.

- **"From tracing page content (including white areas) with inner boarders"**: White areas are considered to be part of the rendered page content (not transparent areas that appear to be white because they show through the white background). Bleed is created at the inner and outer boarders of a shape.
- **"From tracing page content (excluding white areas)"**: White areas are not considered to be part of the rendered page content (not transparent areas that appear to be white because they show through the white background). Bleed is created only at the outer boarders of a shape.
- **"From tracing page content (excluding white areas) with inner boarders"**: White areas are not considered to be part of the rendered page content (not transparent areas that appear to be white because they show through the white background). Bleed is created at the inner and outer boarders of a shape.

9. Set BleedBox

Here you can specify whether the BleedBox should be set again after bleed generation. If the BleedBox is already correctly defined in the document, the setting "ignore" can be used. If there is no bleed in the document yet, "Always (override existing)" should be set, or if there is no BleedBox at all, select "If missing".

10. Offset of bleed

How much the source area for the bleed generation should be adjusted. Negative values move the edges towards the center of the page, positive values move the edges outside the page. An offset is useful when the page content ends just before the page edges, leaving a thin visible white line after the bleed is added. For this use case, a small offset (such as -0.2 mm) can be specified.

11. Apply to

Here you can use a filter to specify to which pages the bleed generation is applied.

Example

Demofile_callas_label.pdf

2.5 mm bleed for irregular shapes.kfpx

(earliest version with full support is **pdfToolbox 15**)



You can watch all this and more about 'bleed' in the video below:

13.3 Check and fix bleed

pdfToolbox 11 introduces a new Process Plan for identifying and fixing bleed issues. Limitation: It will only work with PDF files where all pages have the same size.

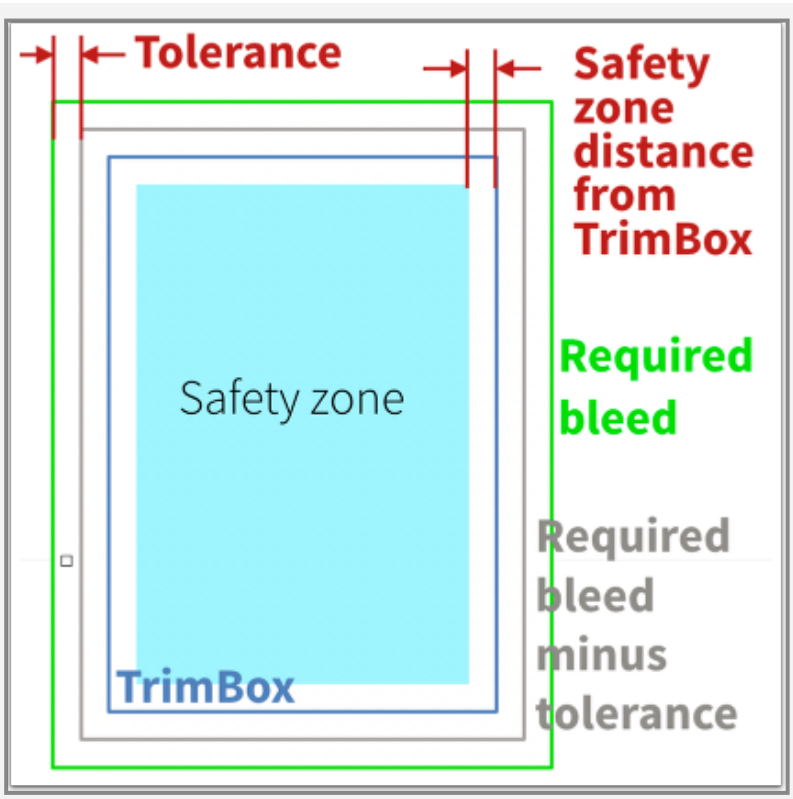
This Process Plan is based on many years prepress experience and has been developed by calibrate (office@calibrate.at). All JavaScripts in the Process Plan are protected and cannot be displayed.



When you start the Process Plan you are asked for:

- Required bleed
- Tolerance [%]
- Safety zone distance from TrimBox
- Unit
- Create required bleed (mirroring)
- Page type
- Spot colors to exclude (RegEx)

The first three input fields are illustrated by this diagram.



During processing pages are analyzed on all four edges. A page edge is classified as requiring bleed if

- there are objects in the "safety zone distance from TrimBox"
  AND
- the required bleed zone is empty or the required bleed zone is not empty but at the edge of the tolerance there are no objects

The respective page edge is then either reported or - if "Create required bleed" is selected - bleed is added via mirroring page content.

The method to add bleed can be adjusted in the Process Plan.

You can watch all this and more about 'bleed' in the video below:

13.4 Create a dieline and bleed for irregular shapes (v11.0)

When you create bleed for irregular shapes you will usually have a dieline or a similar object that defines the border of the shape that requires bleed.

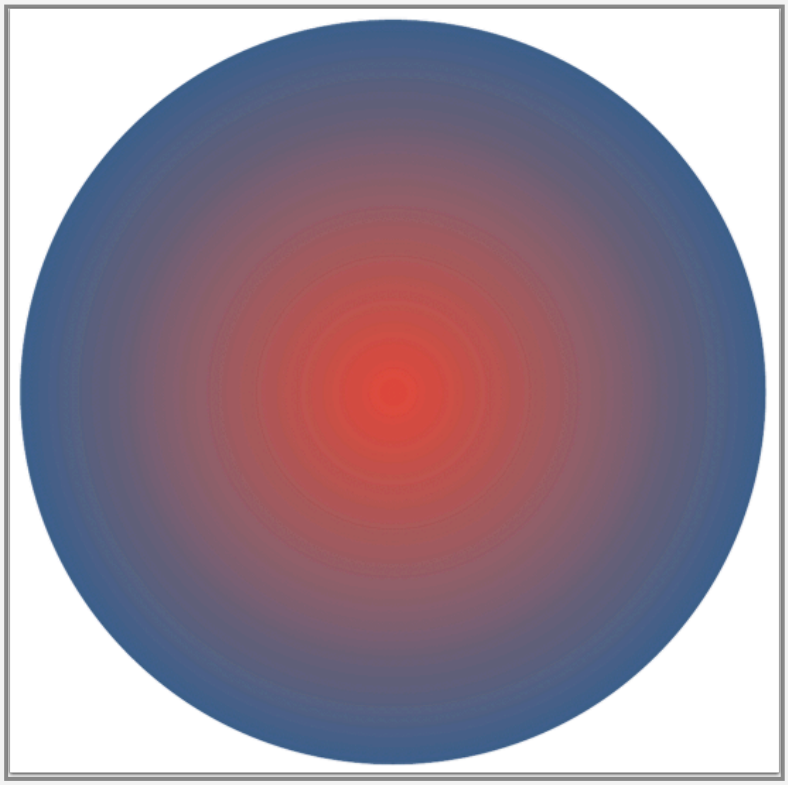
However, if that is missing you may want to create such a dieline on the fly and you can use pdfToolbox' shapes technology for that. The attached Process Plan adds a dieline to the outer border of the shape generated from the combined objects on a page and adds bleed to it. The dieline is created as a spot color (Dieline) on a layer (Dieline) and the layer is associated with Processing Steps metadata (Structural:Cutting).

Make_bleed_and_dieline_for_irregular_shapes.kfpx

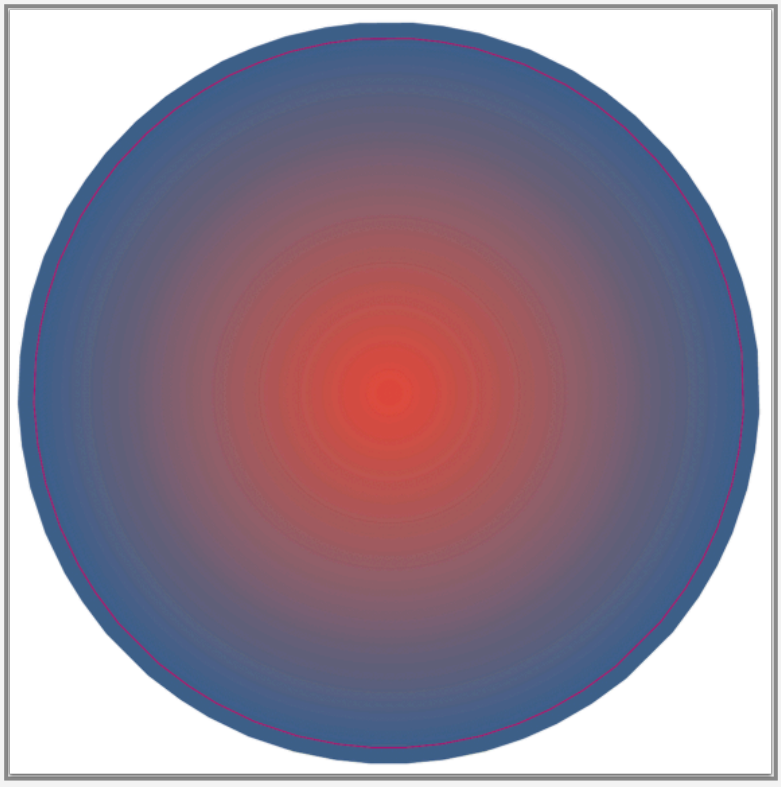
You can use the Process Plan with any PDF and any shape, but you may as well use this one.

Logo.pdf

Original



Result (Dieline has been colord pink)



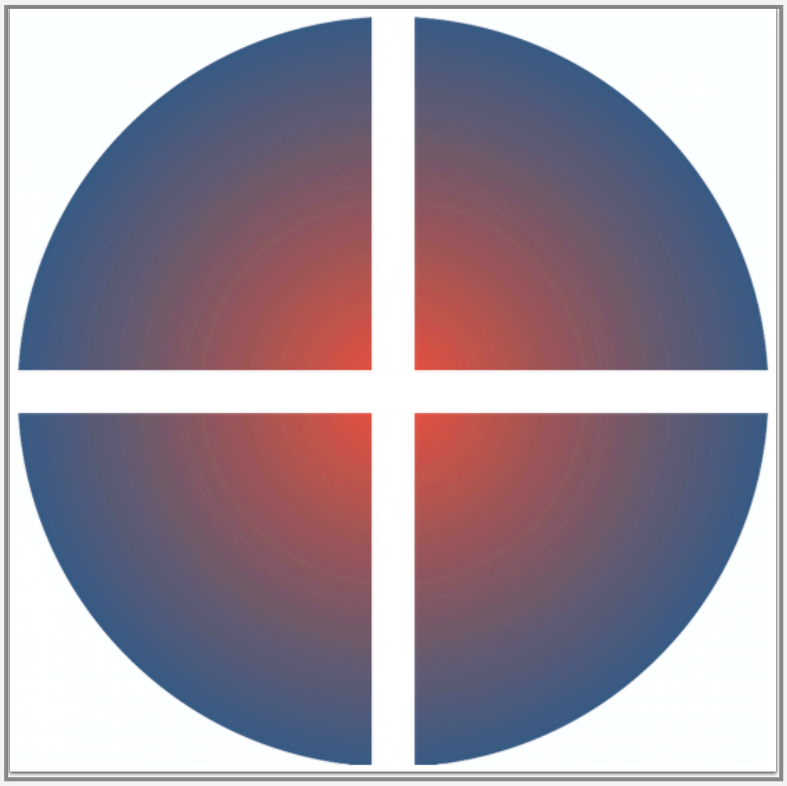
In this article you can read what you can do when the shape has gaps.

You can watch how to 'create a dieline and bleed for irregular shapes in case its missing' in the video below:

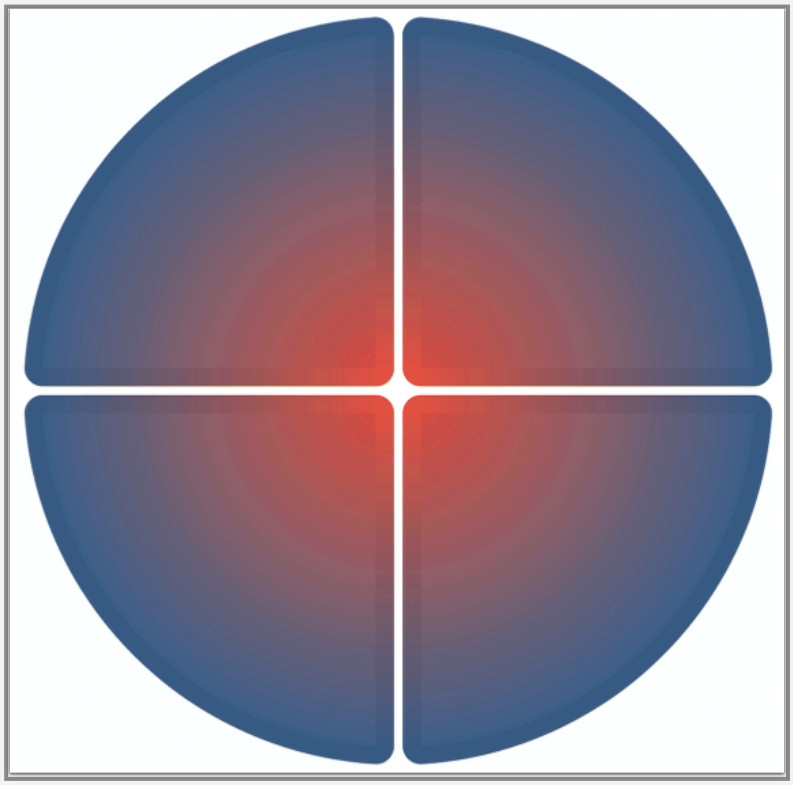
13.5 Create a dieline and bleed for irregular shapes with gaps in outer border (v11.0)

If a dieline is missing and the outer border of the shape that needs bleed has gaps creating bleed does not easily work.

When the original looks like this:



Normal bleed creation would generate something like

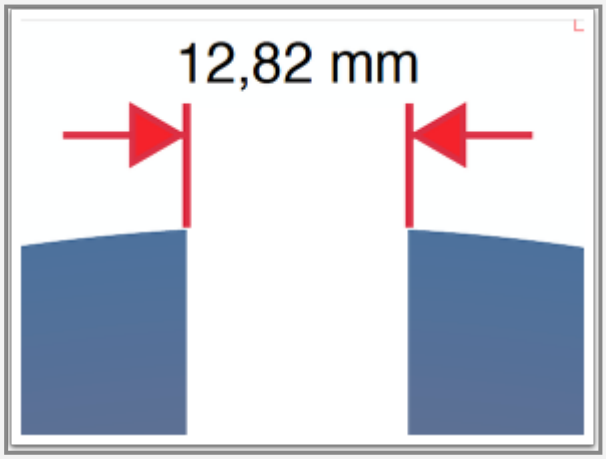


Nice, but not was is needed.

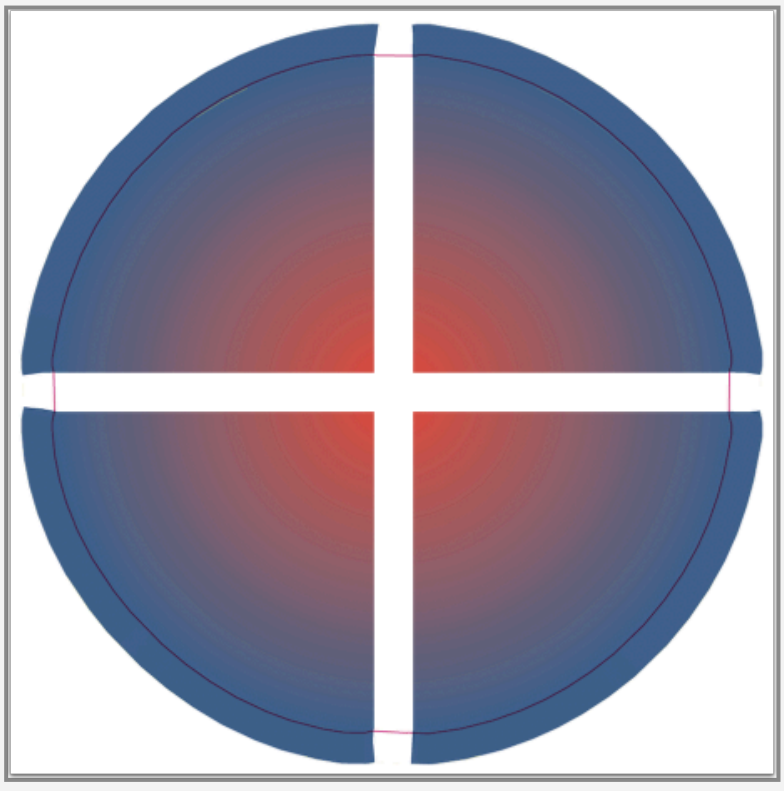
This Process Plan creates a shape of all objects on a page and adds a line to each single object. The width of the line can be adjusted.

Make_bleed_if_dieline_is_missing_and_there_ar.kfpx

In this example the size of the gaps is roughly 13 mm.



The Process Plan creates lines with a thickness of 4 x size of the gaps as a "HelpShape" on a layer and then creates a dieline 4 x size of the gaps inside of HelpShape. It then switches the HelpShape layer off and creates bleed at the dieline.



You can watch how to 'create a dieline and bleed for irregular shapes with gaps in outer borders' in the video below:

13.6 Change page size with the help of the “Set page geometry boxes” Fixup

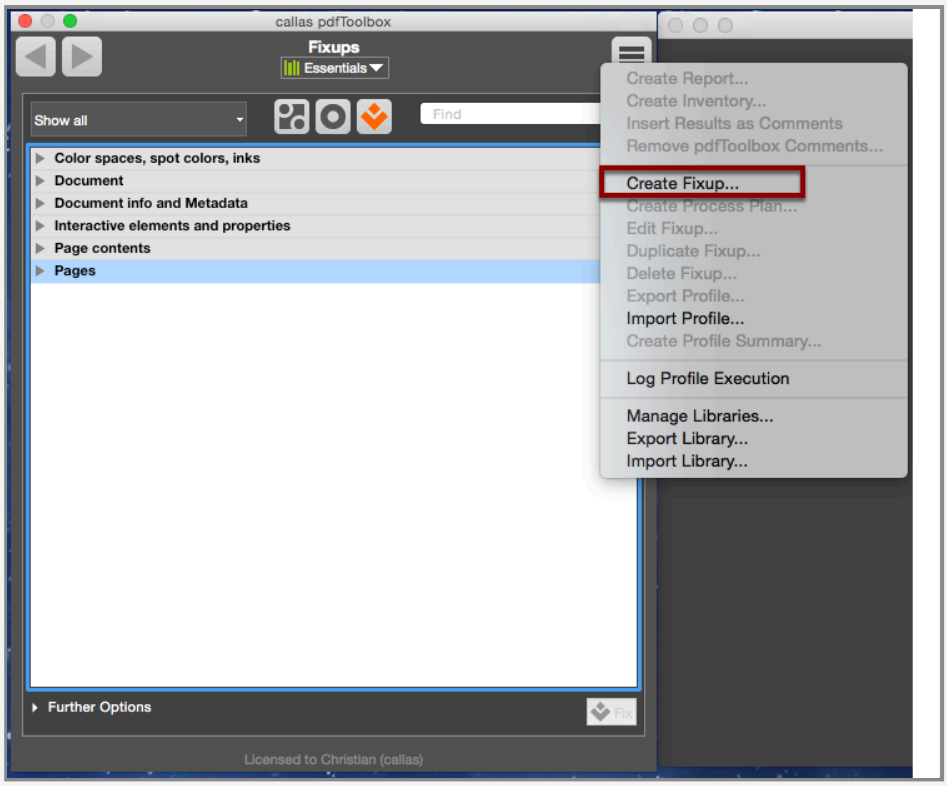
If you need to change page sizes within PDFs, you can use the “Set page geometry boxes” Fixup. This Fixup can even be applied to only even- or odd-numbered pages.

1-12.pdf

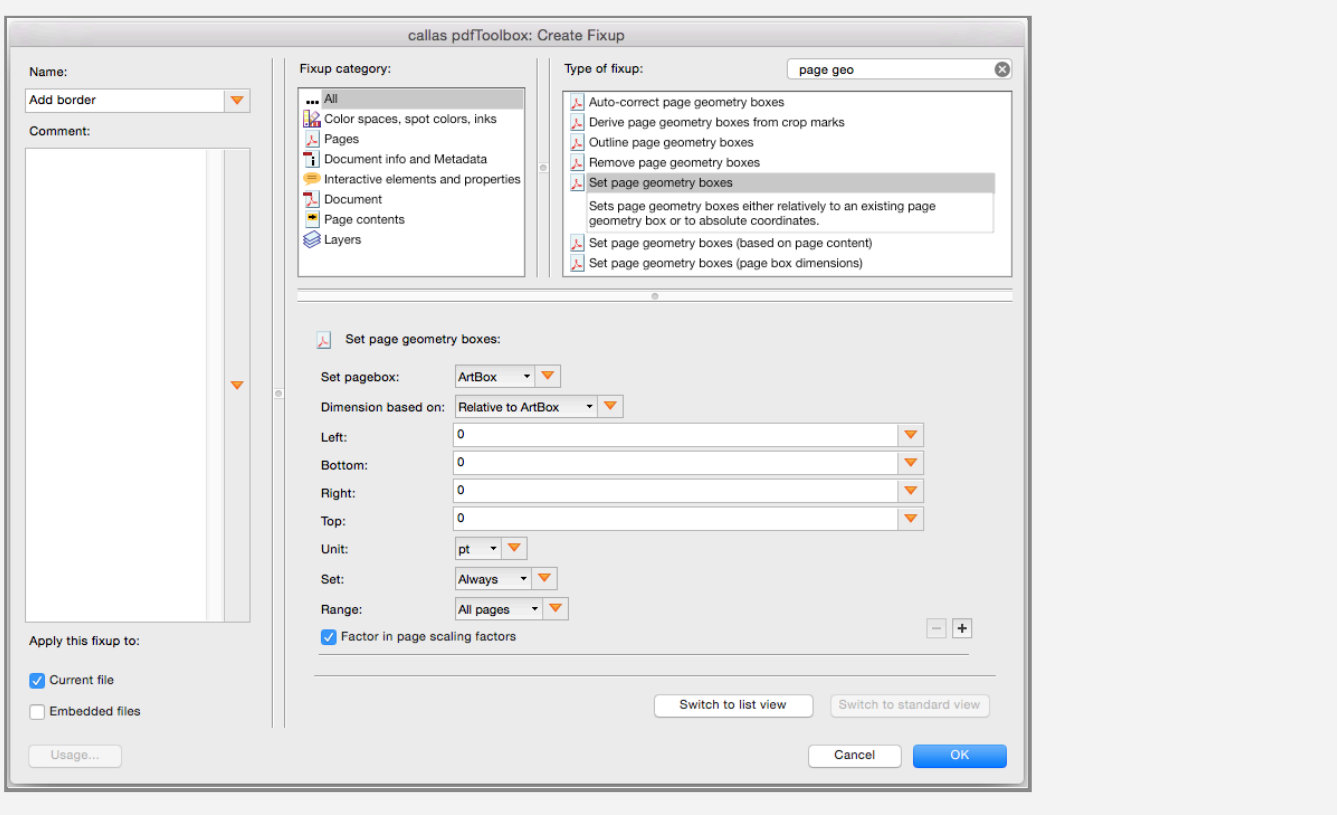
1-12_fix.pdf

Add_border.kfpx

First, we call up the list of Fixups and create a new Fixup.



Now, in the list of available Fixups, select “Set page geometry boxes” and configure it accordingly.

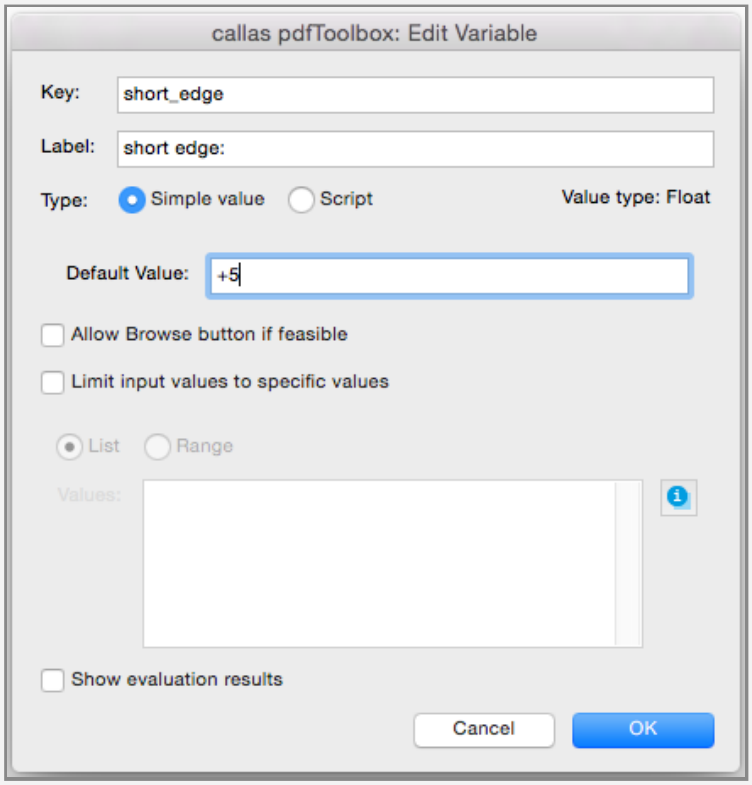


In this specific example, we will add +10 mm to the left and +5 mm to the right of even-numbered pages. For odd-numbered pages, we will add +5 mm to the left and +10 mm to the right.

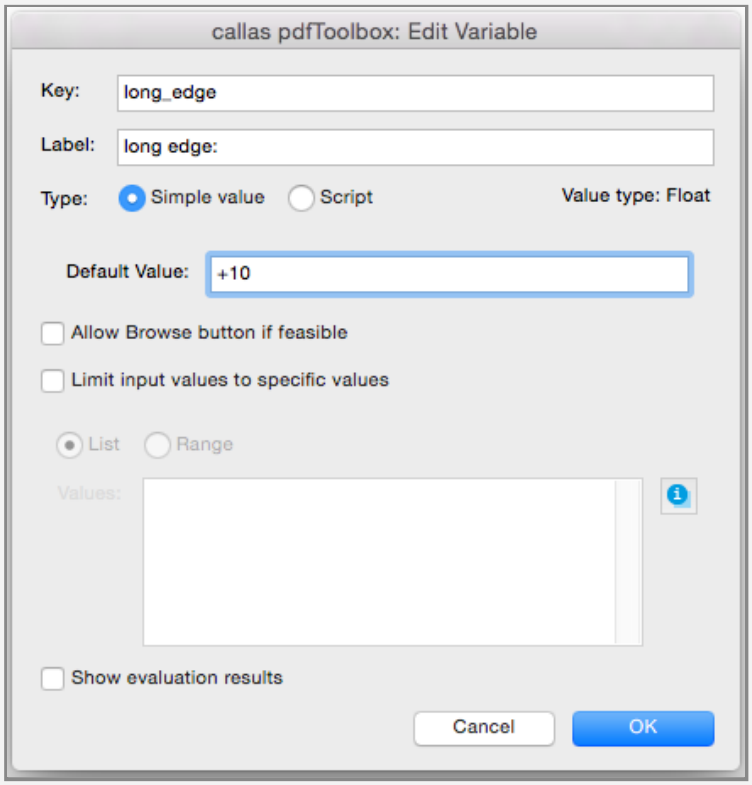
By default, distances are measured in pt; this should be changed to mm.

So that we only have to enter one detail for both longer and shorter pages (for even and odd pages), we will use variables.

To add variables, click on the orange triangle and select “New variable...”.

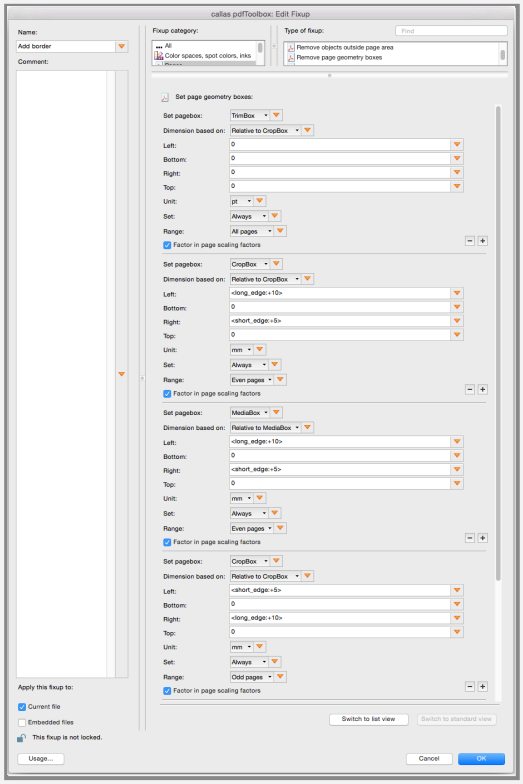


Please note here that variable names must not contain spaces.



The finished Fixup should look like the one shown here.

There are as many steps for long edges as for short edges.



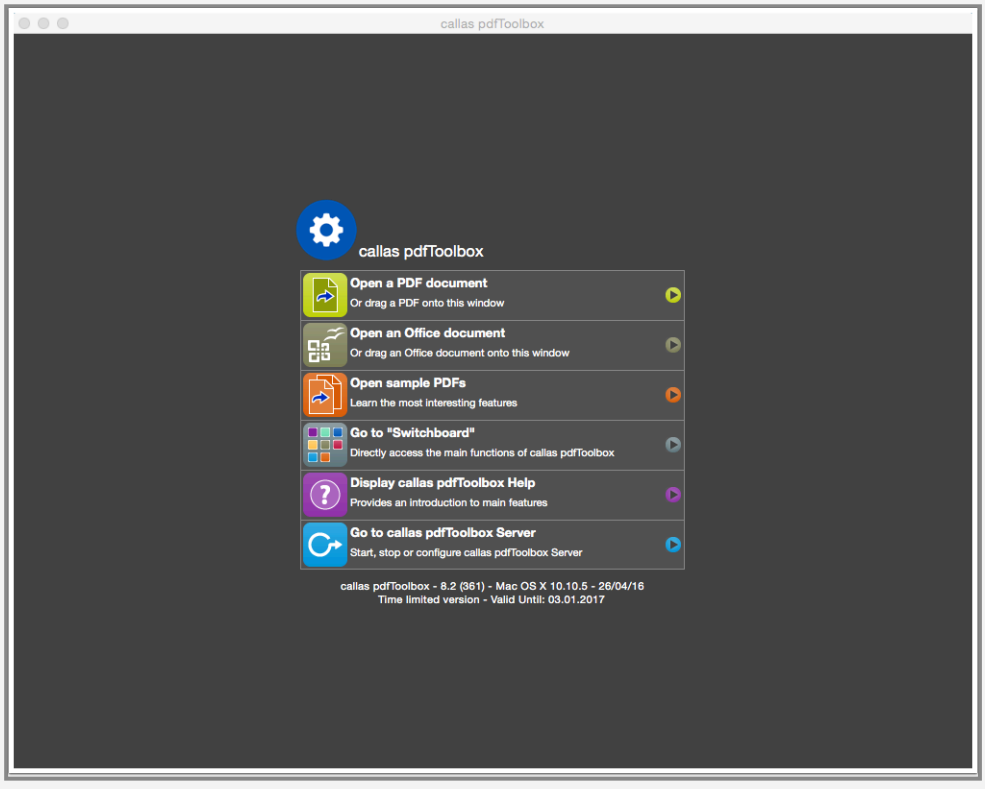
The finished profile is also attached to this tutorial, making it easy to import.

13.7 Outline page geometry boxes in a specified tint value of a spot color

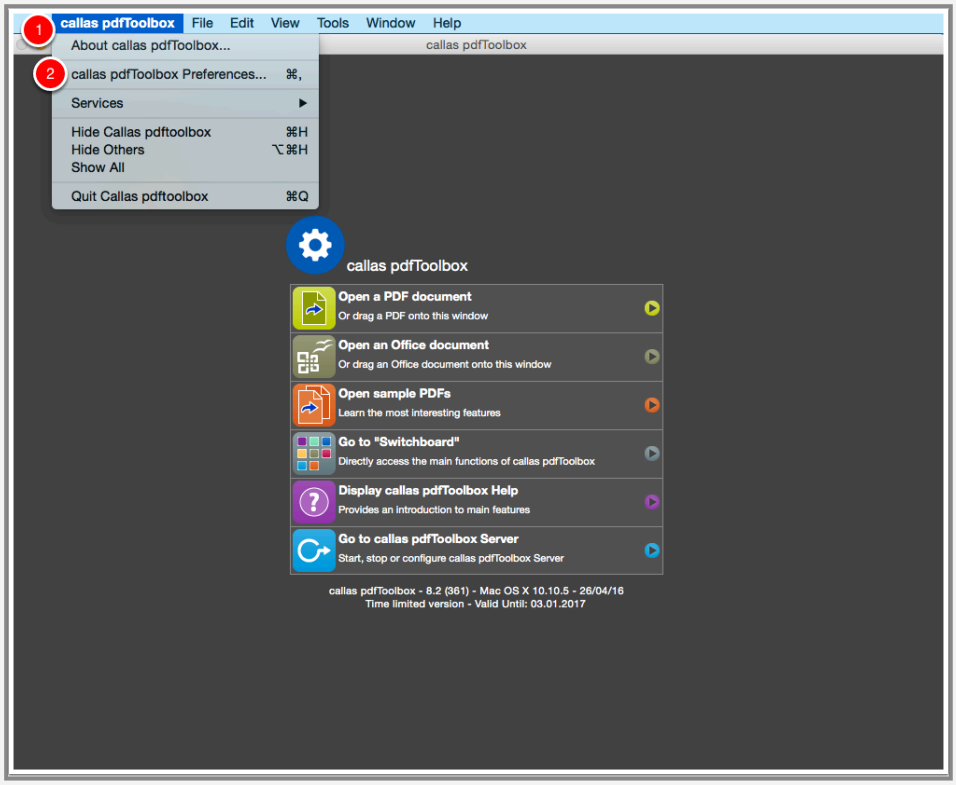
Starting pdfToolbox 8.2, it is possible to trace the page geometry frame BleedBox, ArtBox and TrimBox with lines in special colors and on separate layers.

PDF-boxes_Demo_file.pdf

Launch pdfToolbox Desktop



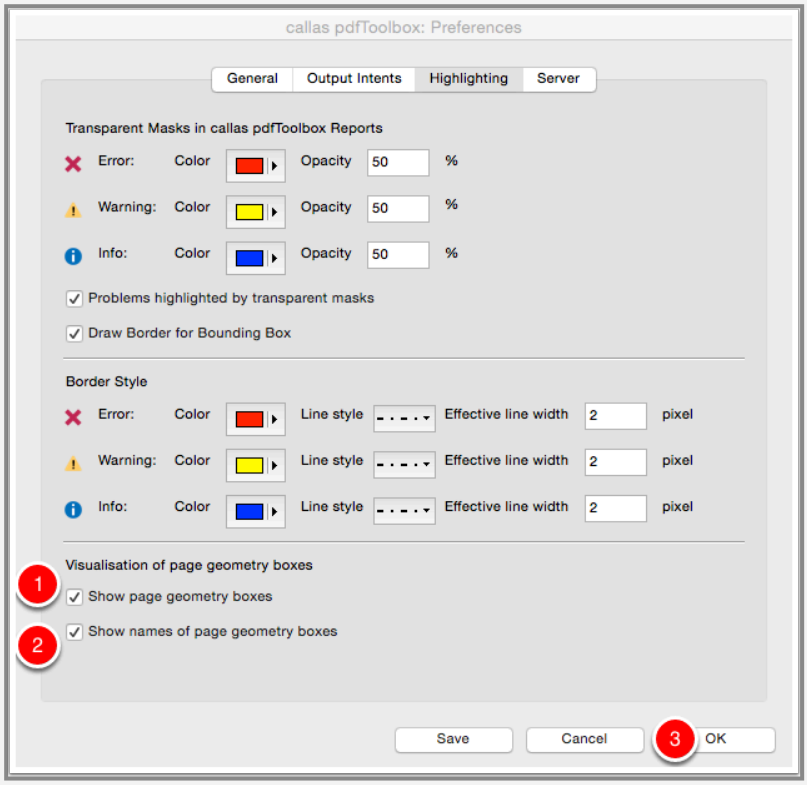
Open pdfToolbox Preferences



1. Go to "callas pdfToolbox".
2. Click "callas pdfToolbox Preferences".

NOTE: On windows you have to go to "Edit > callas pdfToolbox Preferences".

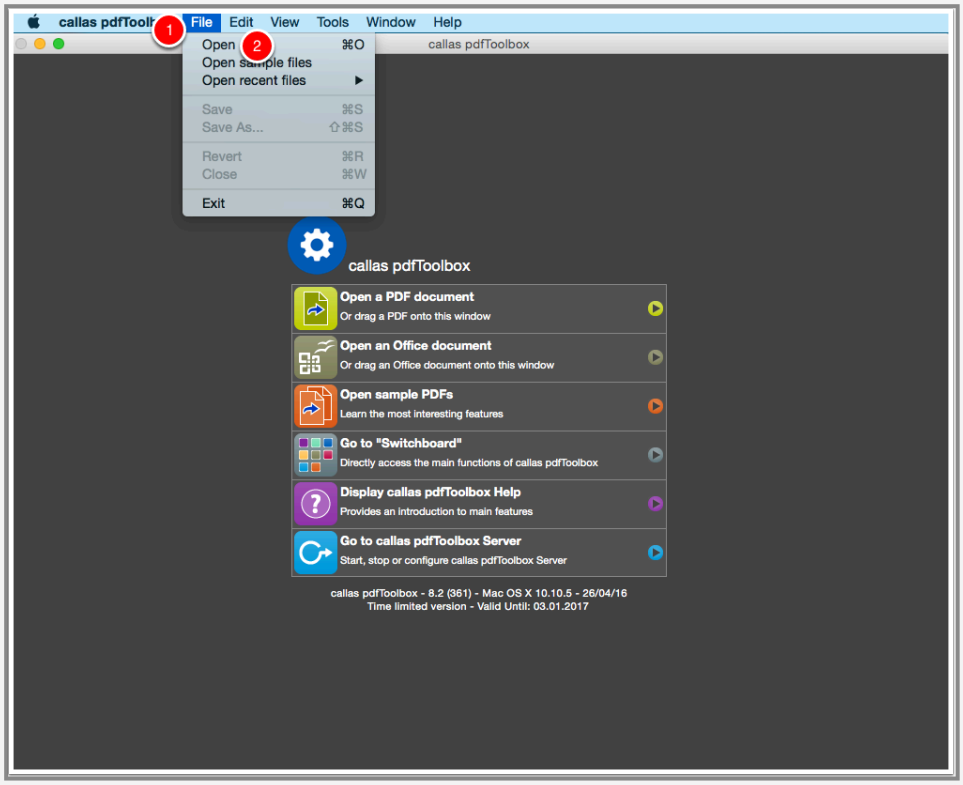
Change page geometry boxes preferences



You can show the page geometry boxes in pdfToolbox when you open a PDF file. In addition, the names of the boxes can be show too.

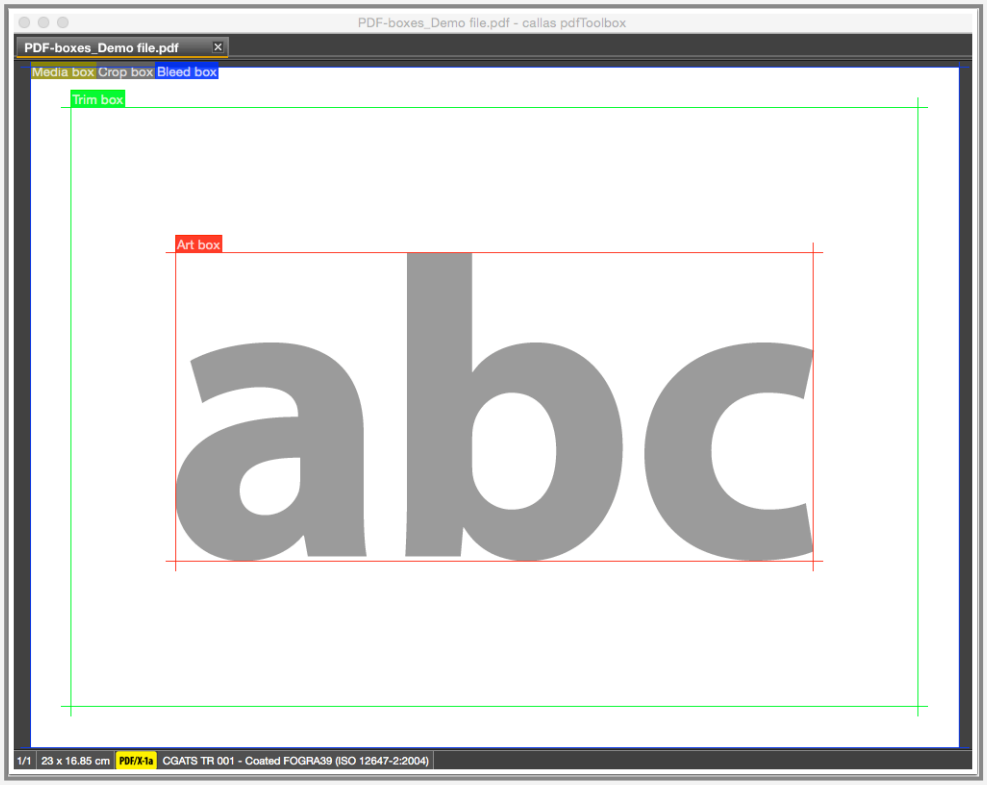
1. Select "Show page geometry boxes".
2. Select "Show names of page geometry boxes".
3. Click "OK".

Open PDF file "PDF-boxes_Demo file.pdf"



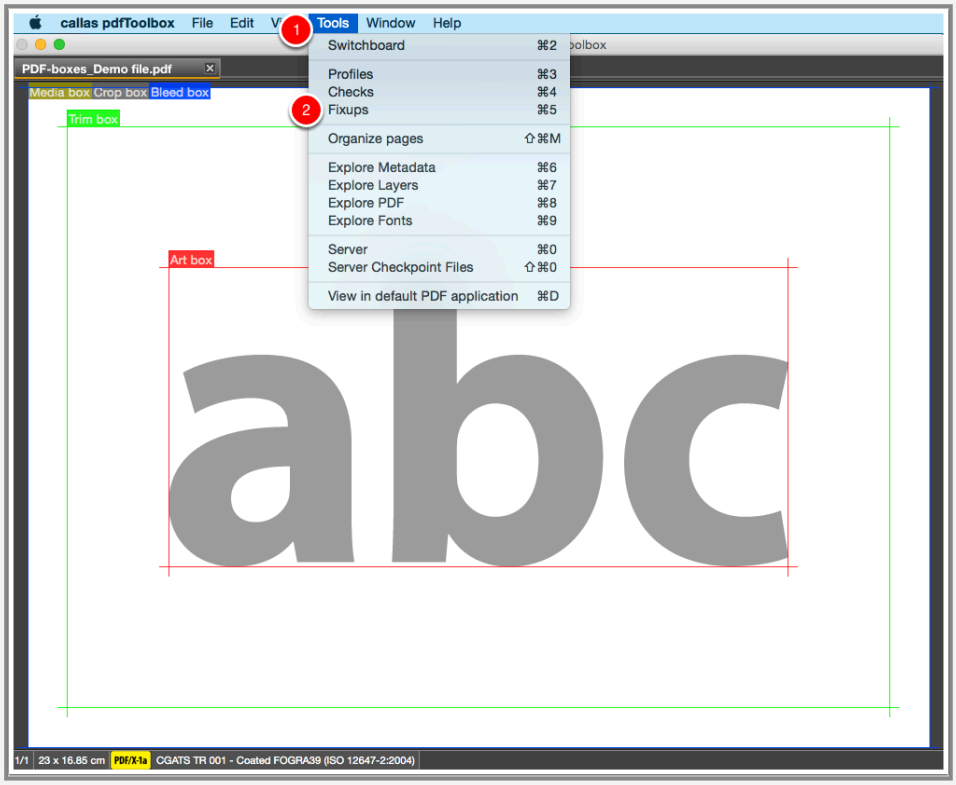
1. Go to "File".
2. Click "Open".

Inspect the page geometry boxes



The page geometry boxes with names are shown when open the PDF file in pdfToolbox.

Open the Fixups dialog



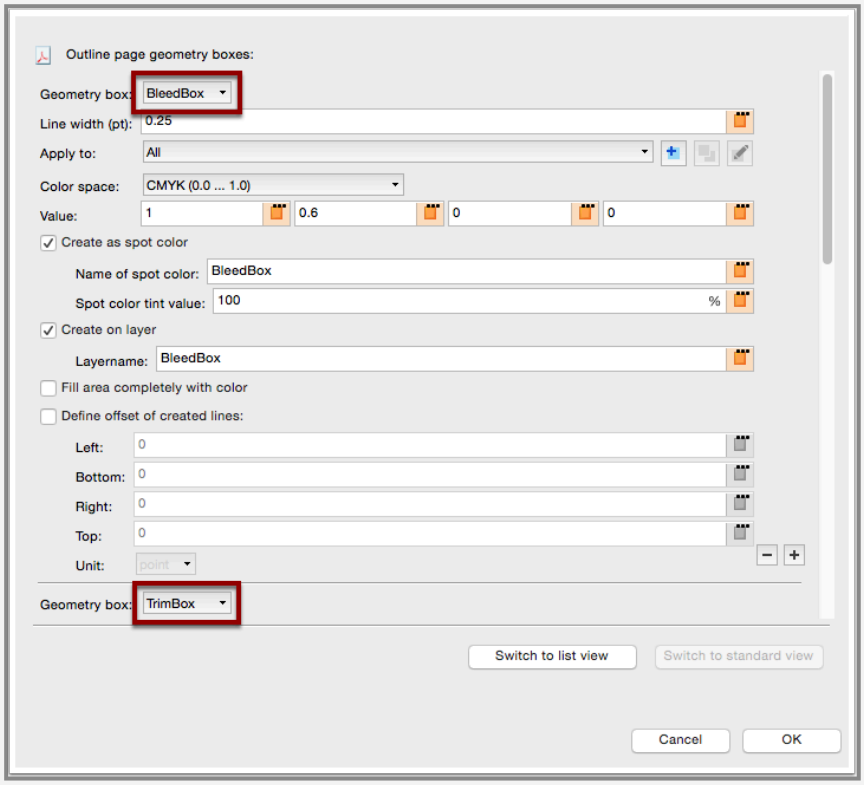
1. Go to "Tools".
2. Click "Fixups".

Search to Fixup



1. In the search field search to "Outline page geometry boxes".
2. Select the Fixup "Outline page geometry boxes".
3. Click "Edit".

Inspect the Fixup



The Fixup add three page geometry frames to the PDF file.

- BleedBox (crop rectangle)
- TrimBox (TrimBox)
- ArtBox (object frame)

These elements are presented in three different spot colors (blue, green and red) on three different layers.

To work organised, these new elements are each inserted into newly created levels.

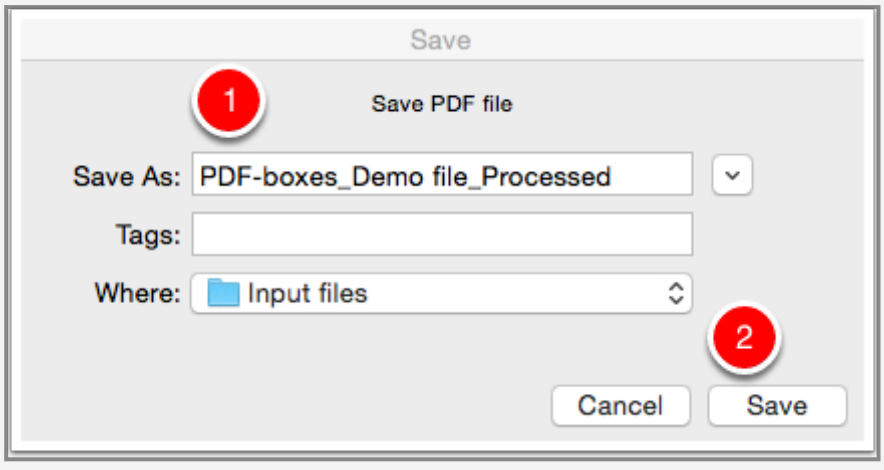
NOTE: The Fixup can be adjusted. Make a copy of the predefined Fixup and change the settings. By clicking on the "OK" button, the changes are applied.

Apply the Fixup



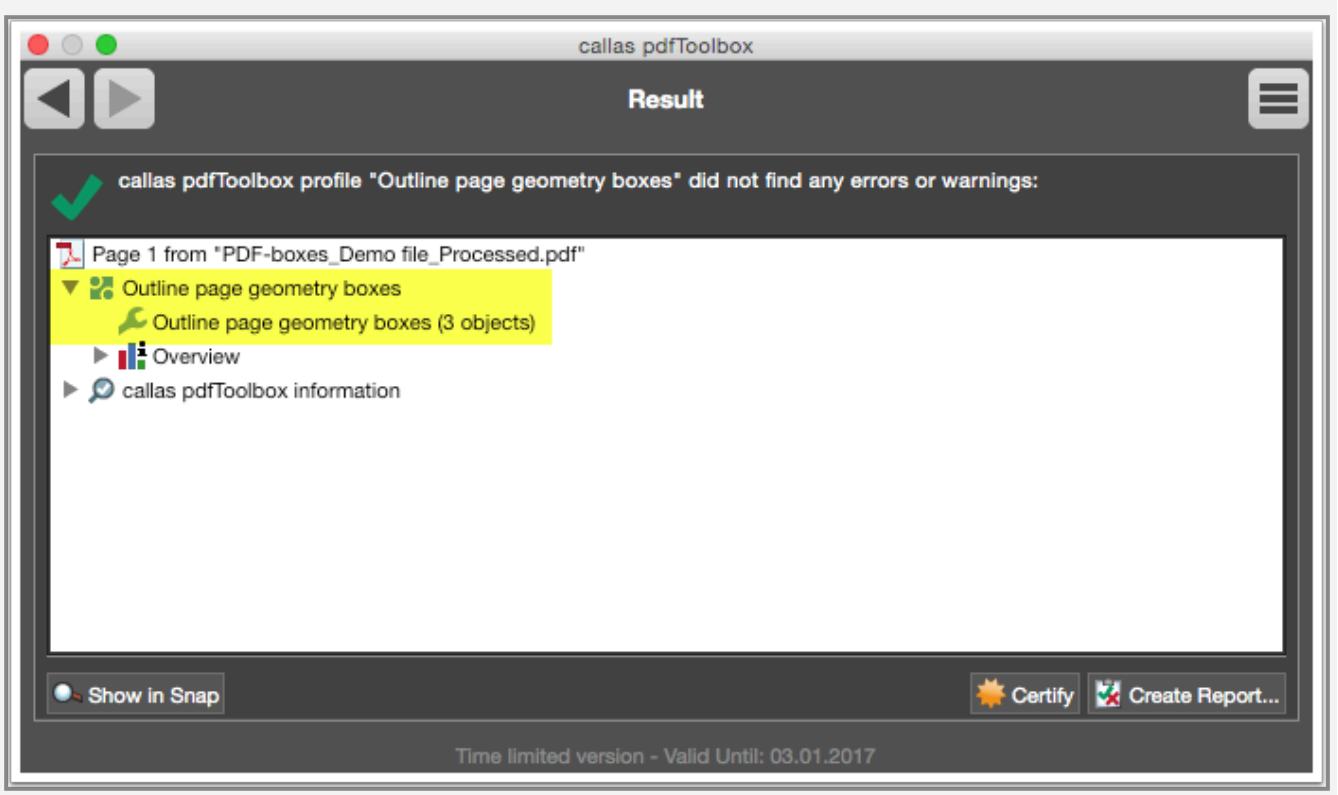
1. Click "Fix".

Save the output PDF file



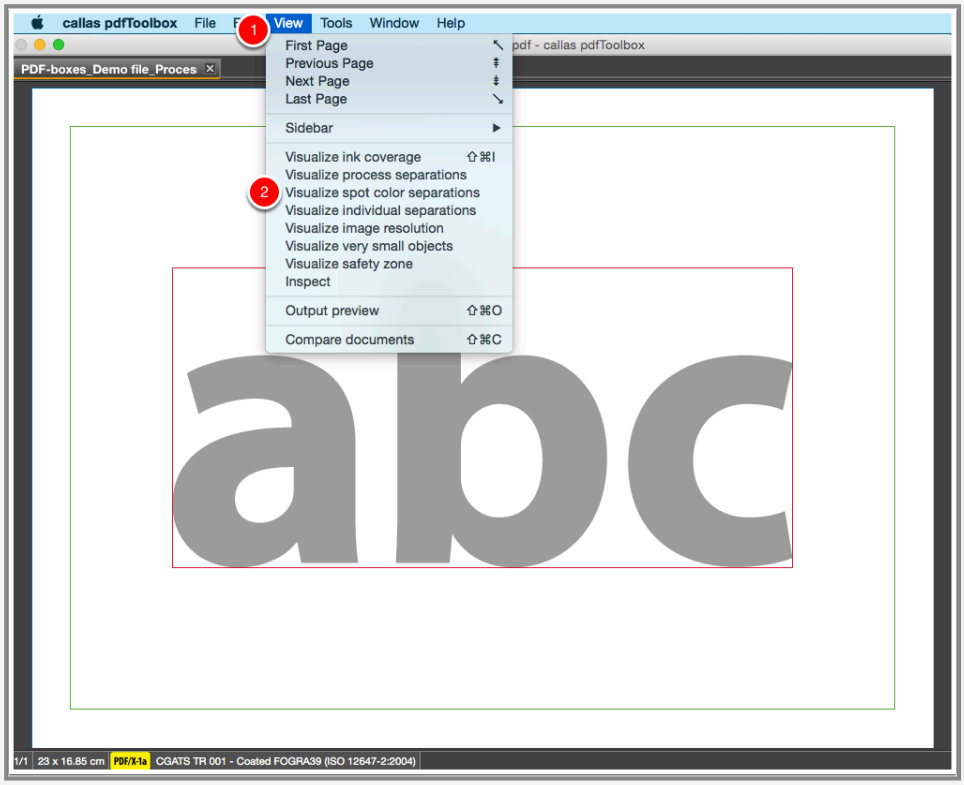
1. Save the output PDF file as "PDF-boxes_Demo file_Processed".
2. Click "Save".

Inspect the preflight report



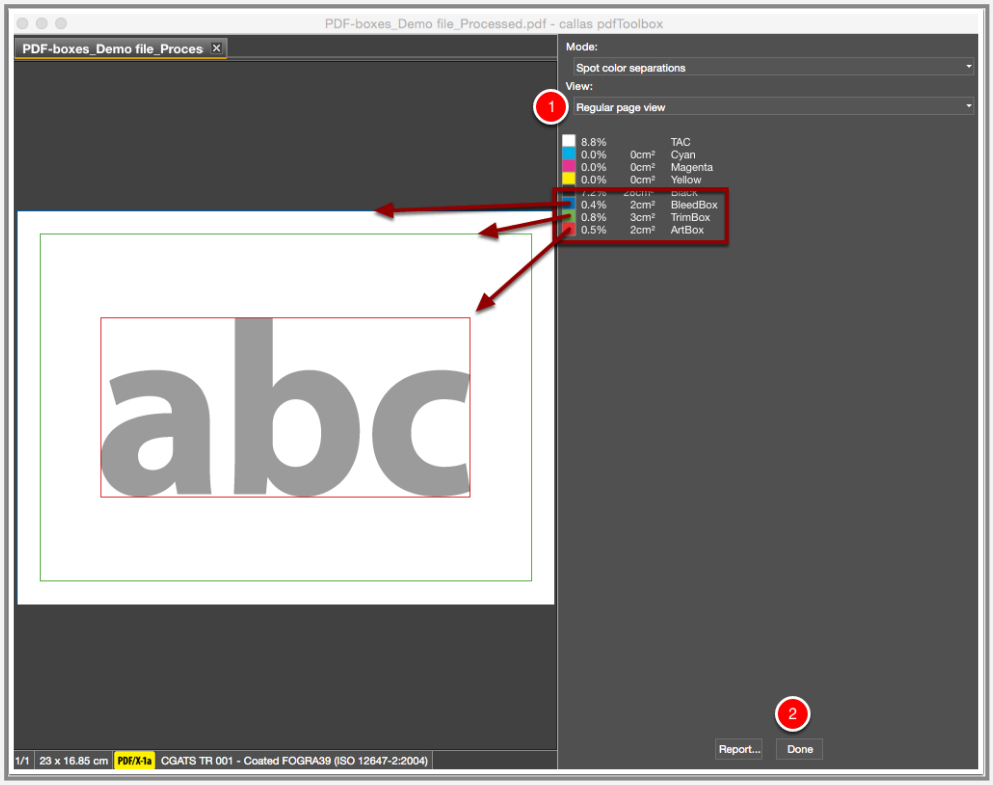
A green check mark is shown, refers to a successful correction. The three page geometry frames (BleedBox, TrimBox and ArtBox) are added in the PDF file.

Open Visualize spot color separations panel



1. Go to "View".
2. Click "Visualize spot color separations".

Inspect the processed PDF file

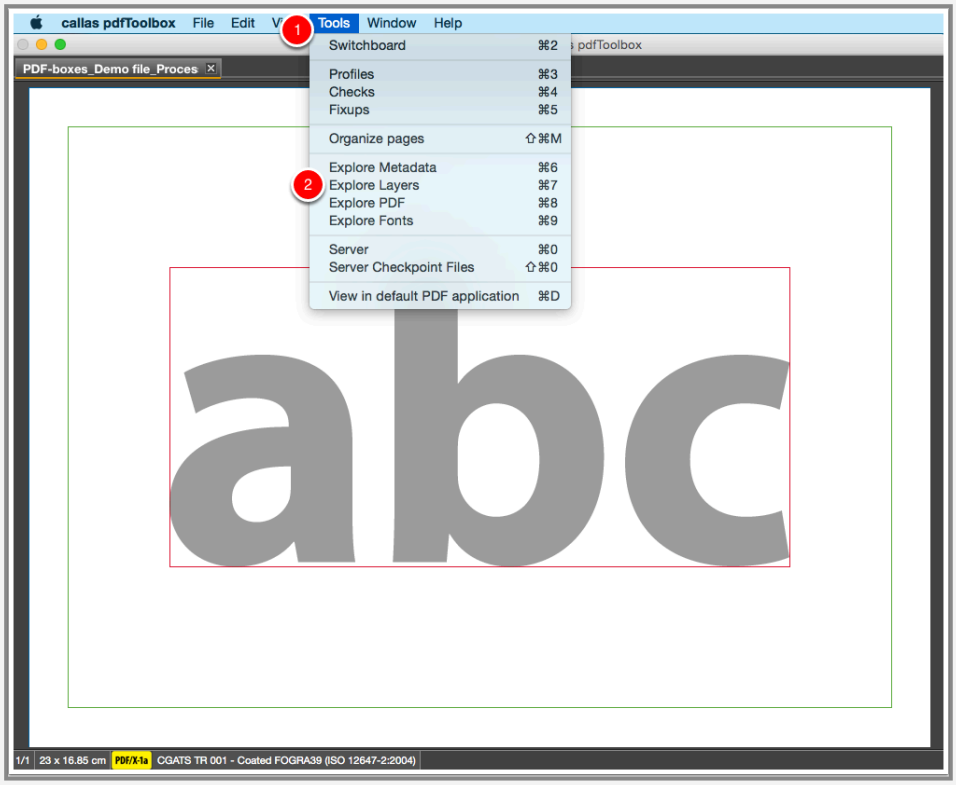


1. Select "Regular page view".

   The three page geometry frames (BleedBox, TrimBox and ArtBox) are presented in three different spot colors.

   NOTE: Too see the page geometry frames, you have to switch off "Show page geometry boxes" and "Show names of page geometry boxes" in the pfdToolbox preferences. Reopen the PDF file too see the changes.
2. Click "Done".

Open Explore Layers dialog



1. Go to "Tools".
2. Click "Explore Layers".

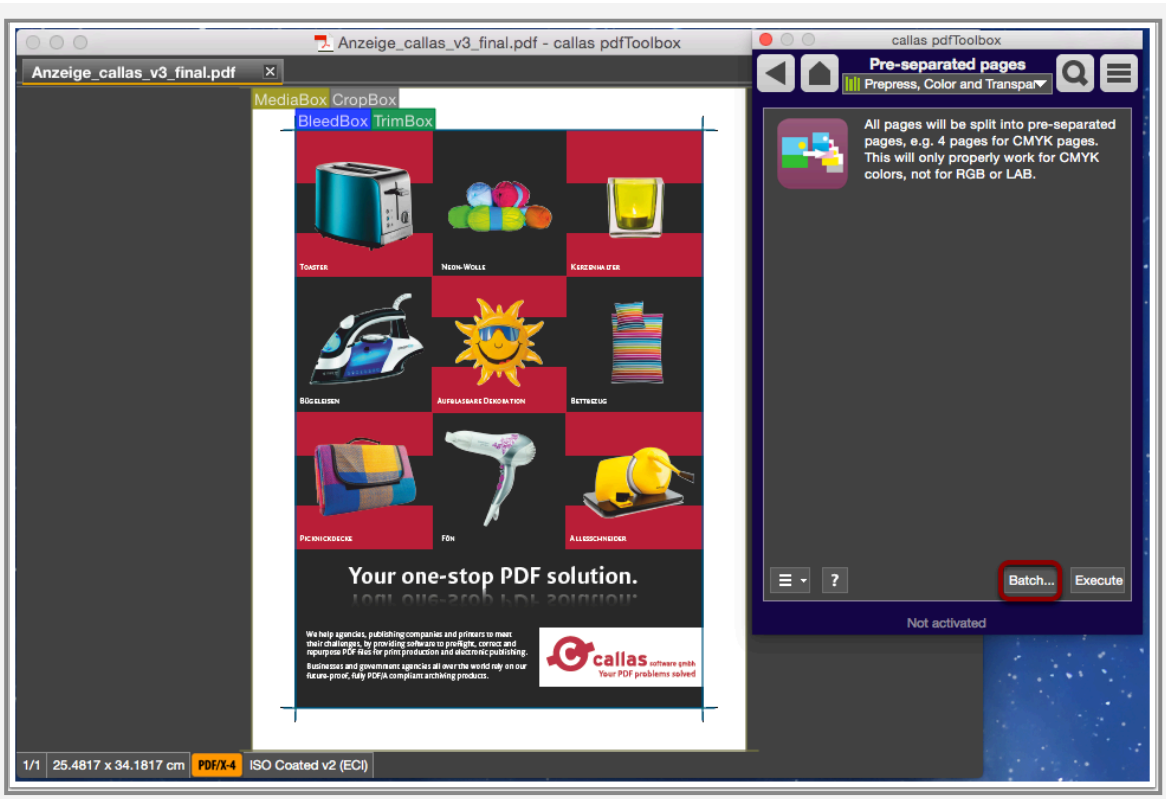
Inspect the layers



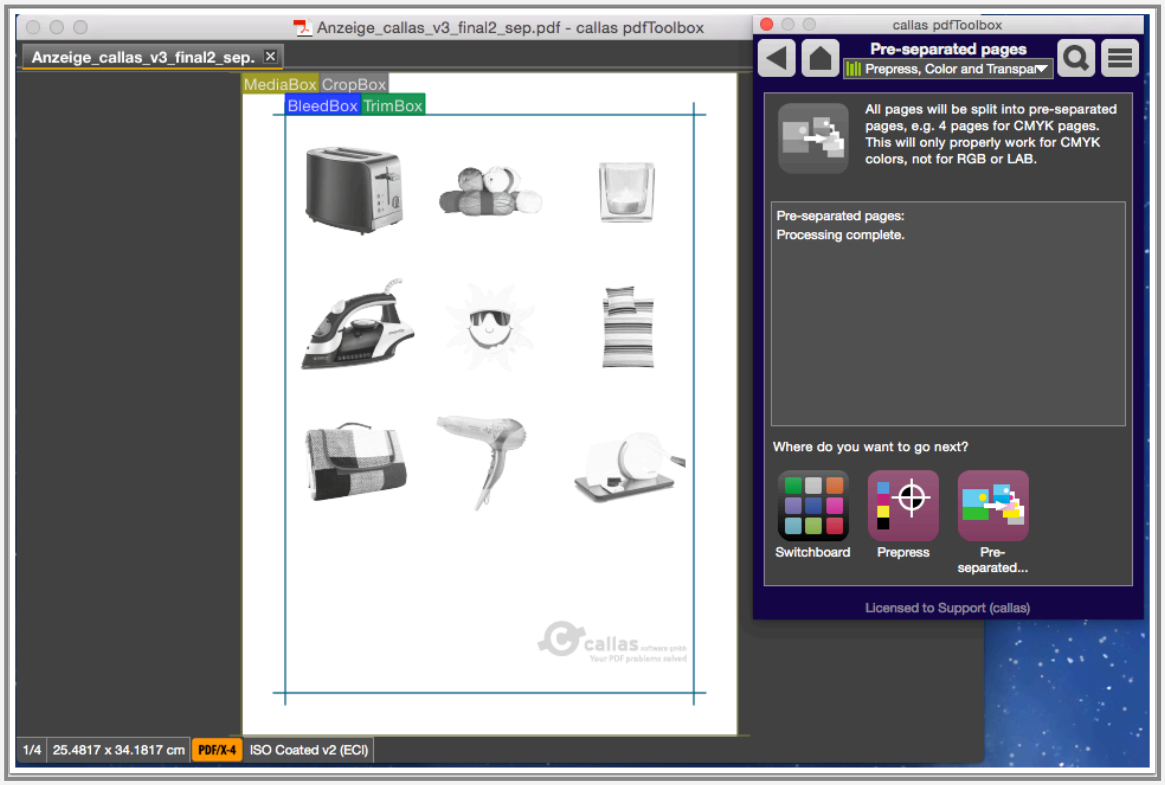
The three page geometry frames (BleedBox, TrimBox and ArtBox) are presented on three different layers.

13.8 Create pre-separated pages

The Switchboard Action “Pre-separated pages” under the “Prepress” group lets you convert one page into individual pre-separated pages. For example, this can be used to turn 1 CMYK page into 4 individual pages, one for each separation. Spot colors are also supported.
However, it is not possible to process other color spaces such as RGB or Lab.



To begin this Switchboard action, simply click on the corresponding button.



The result will contain the required number of pre-separated pages for each input page. This example shows the cyan separation.
The file will also contain a corresponding internal entry at the page level (“SeparationInfo” and “DeviceColorant”) to allow output devices (such as imagesetters for printing plates) to identify and output each separation.

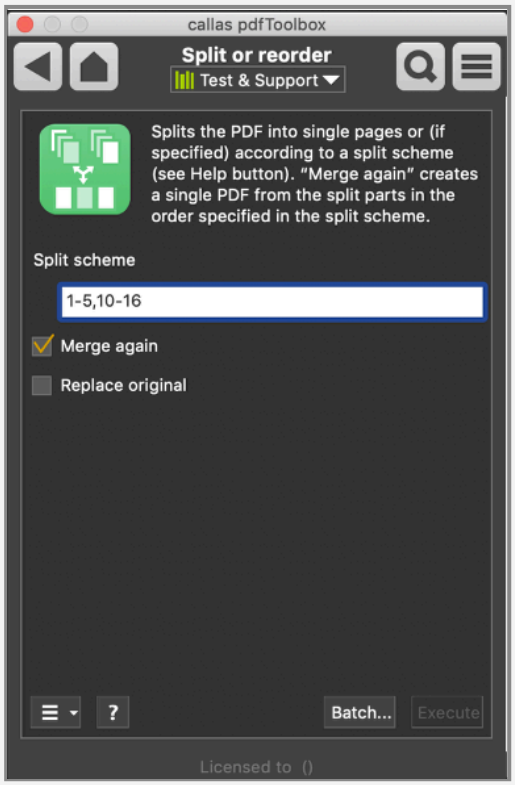
13.9 How to 'Split or reorder' PDF

Split or reorder using the Switchboard

The Switchboard Action in the 'Arrange' group allows you to split your multi-page documents into smaller packages according to a split scheme, with an additional option to merge the individual packages back together again. You can also replace the original document with the newly derived "split and merged" document.



This Action can also be used inside a Process Plan. Within a Process Plan, the generated PDFs will be handled as separate output files (like e.g. images), so they will not be handled in further steps of the Process Plan.



In the example above, a 16 page document is being split into 2 packages. The first package contains 5 pages numbered 1 to 5 whereas the second package contains 7 pages from page

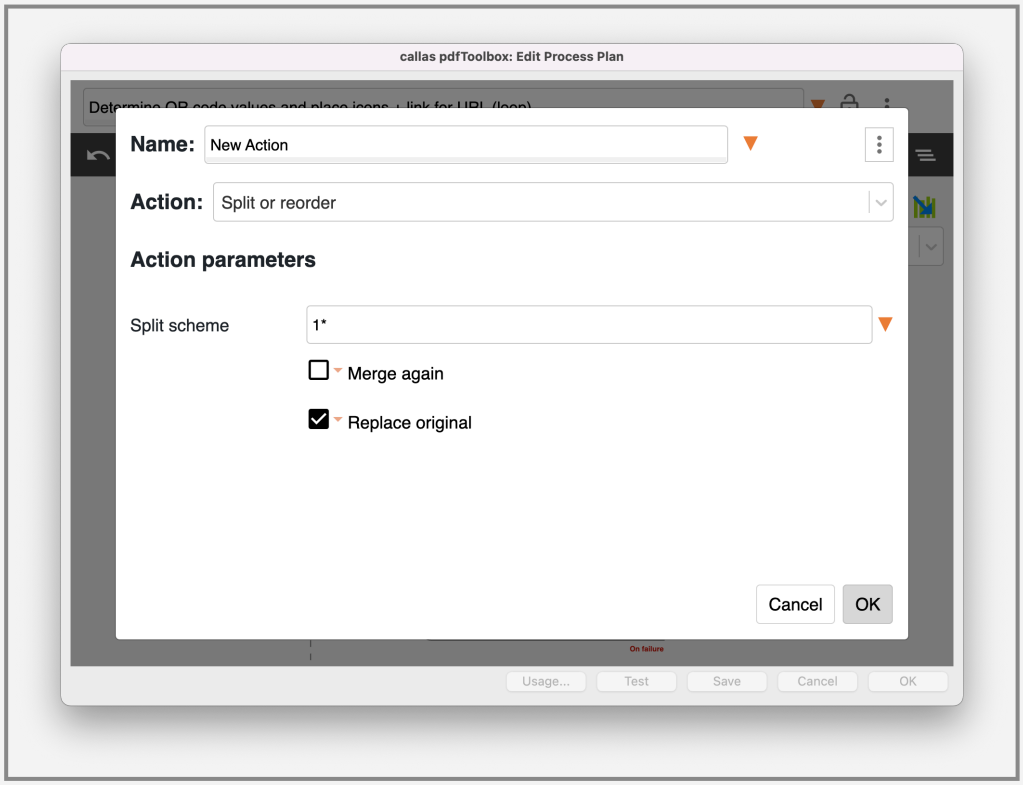
number 10 to 16. A user can optionally check the 'Merge again' checkbox to merge the individual packages again, as done in this example.

Checking the 'Replace original' (can be activated only upon checking 'Merge again') will replace the original document which is now merged with the split scheme '1-5,10-16'.

Split Scheme expression may be a number with an asterisk "*" or a more complex string. If it is a number with an asterisk "*" it creates PDF files with the defined number of pages. E.g. if the number is 3*, it would create 3 packages with 3 pages and one package with one page from a 10 page file. An overview of the different split scheme expressions can be found in this article: Page selector and Split scheme.

Split or reorder in a Process Plan

With the 'Replace original' option of the 'Split or reorder' Action, it is possible to use one merged file as the new input file for the later steps since it replaces the original file.



Split PDF on the CLI

The `--splitpdf` command on the CLI allows you to split multi-page documents into smaller packages. You'll find a list of the full syntax in the following article: Commands related to Arrange – Split PDF

Reorder pages using QuickFix

There are two QuickFixes in pdfToolbox to reorder pages:

- Reorder pages
- Invert page order

13.10 Create white underlays for printing on transparent foil using "Create spot color plate based on ink amount"

Background

When printing on transparent substrates – such as transparent foil for shrink sleeves – it can become essential to print a white primer onto the substrate before printing the actual print content. If all of the surface of thee substrate is treated with a white primer or white underlay, it would not make any sense to use a transparent substrate to begin with.

Thus it is desirable to print a white primer only where some print content is printed afterwards. In addition, the degree to which the ink amount of the print content increases, less – if any at all – white primer might be needed. As in most printing architectures, a white primer is relatively expensive, it would be nice to be able to limit the amount of white primer used as much as possible. Page areas printed with 200% of ink or more do not tend to gain anything anymore from printing white below them beforehand (of course this will depend a lot on the opacity of the inks used).

The fixup "**Create spot color plate based on ink amount**" makes it possible to create a spot color plate where at each position on the page its tint value is derived from the ink amount of the actual print content at that position – using a simple curve file that turns the input value of ink amount into a tint value of the newly generated spot color plate.

The tutorial described in the section below illustrates how the "Create spot color plate based on ink amount" may be used. The tutorial is divided into two parts: first, a simple use case is illustrated. Second, a more complex scenario is explained where the "Create spot color plate based on ink amount" is combined with using the "Create shape" feature to flexibly address the distinction between white and transparent parts of a page.

Create white underlay for packaging label printed on transparent foil (Part 1)

Example file "Peppery Pumpkin Yoghurt"

Peppery_Pumpkin_Yoghurt_2020-11-04__light_yell.pdf



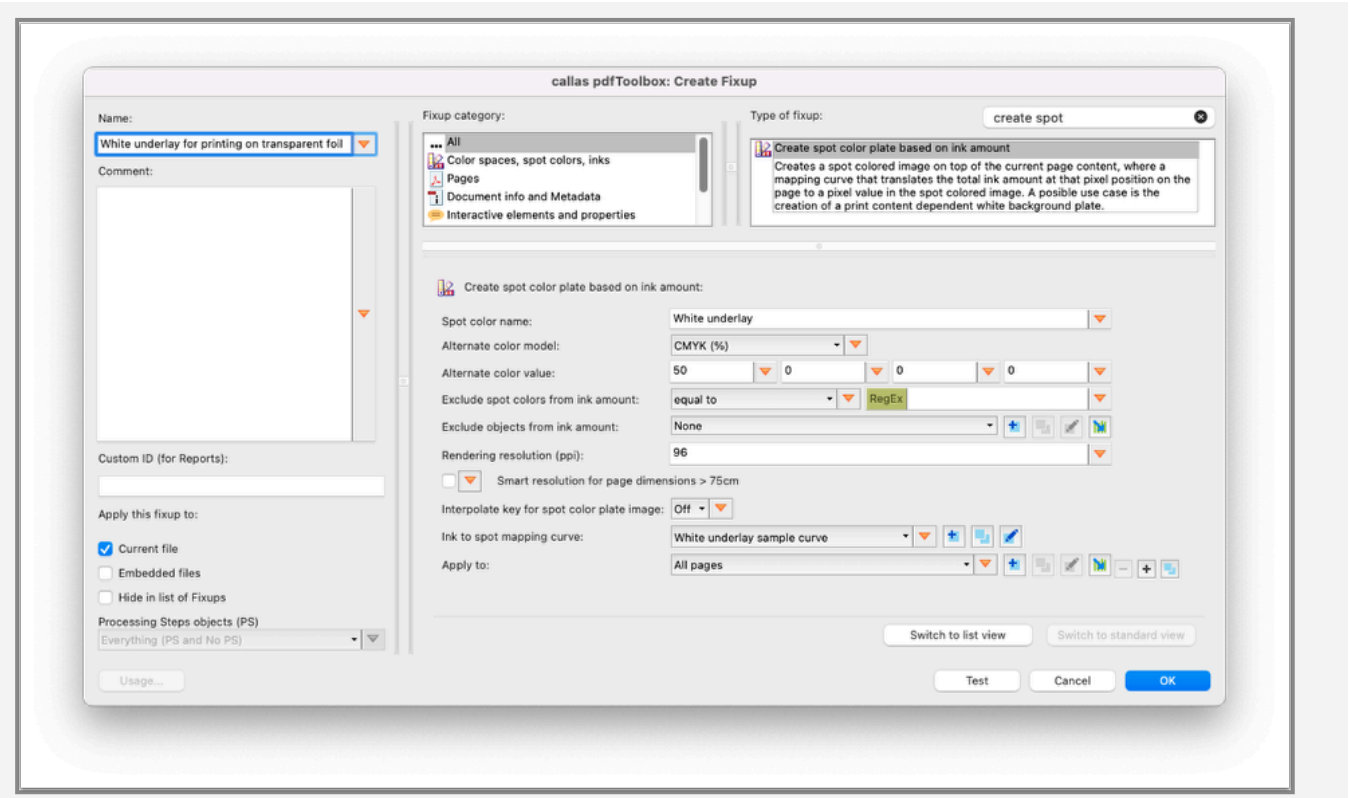
The sample label used in this tutorial is meant for printing on transparent foil, to be used as a glued on lid for yoghurt cups. The goal is to only print a white underlay in areas, where actual print content is present, and to only print as much of the white primer as actually needed – more on light areas of the print content, less in darker areas with a higher ink amount. To keep this tutorial simple we are ignoring different opacities of different inks.

The brownish red areas and the small green part have an ink coverage of just above 200%, whereas the orange areas sit at just above 100%. The remainder of the lid lingers mostly between 25% and 50%. Our assumption is that once ink cover-

age rreaches 200%, no white underlay is needed. Very light areas up to around 50% should be printed on 100% white primer, the range from 50% to 200% should go from 100% to 0% primer.

Create the "Create white underlays for printing on transparent foil" fixup

Create a new fixup, by choosing "Create spot color plate based on ink amount" as the fixup type (entering 'plate' in the search field will make locating it very easy), giving it a suitable name, and going through the configuration options, setting them to the values as illustrated below:



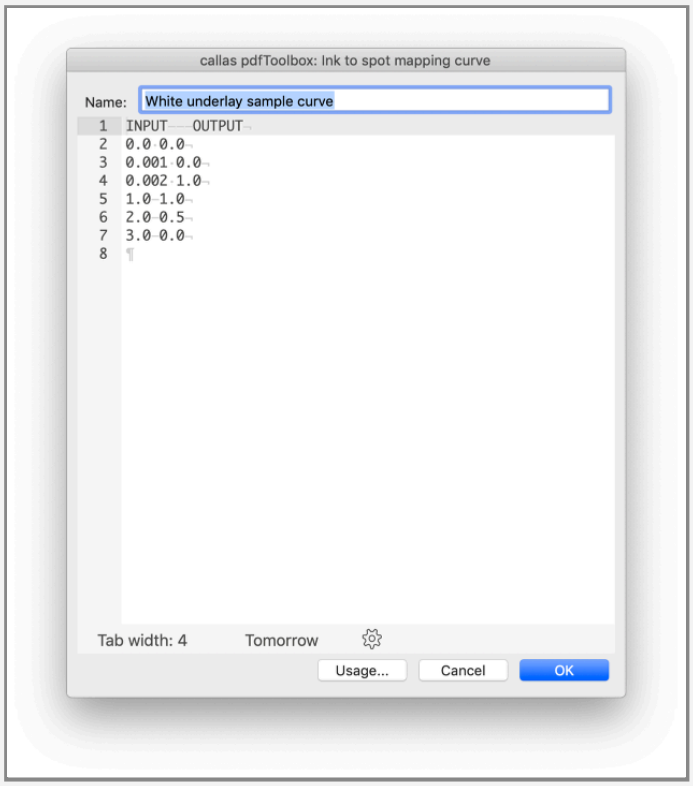
Regardless how the appearance of the spot color to be created is defined, the new spot color plate will be created on top of all existing page content, and will be set to overprint, so that it does not disturb the existing page content.

To make the spot color plate's content as smooth as possible, it is recommended to set the "Interpolate key for spot plate image" to "On".

Setting up the ink amount to spot mapping curve

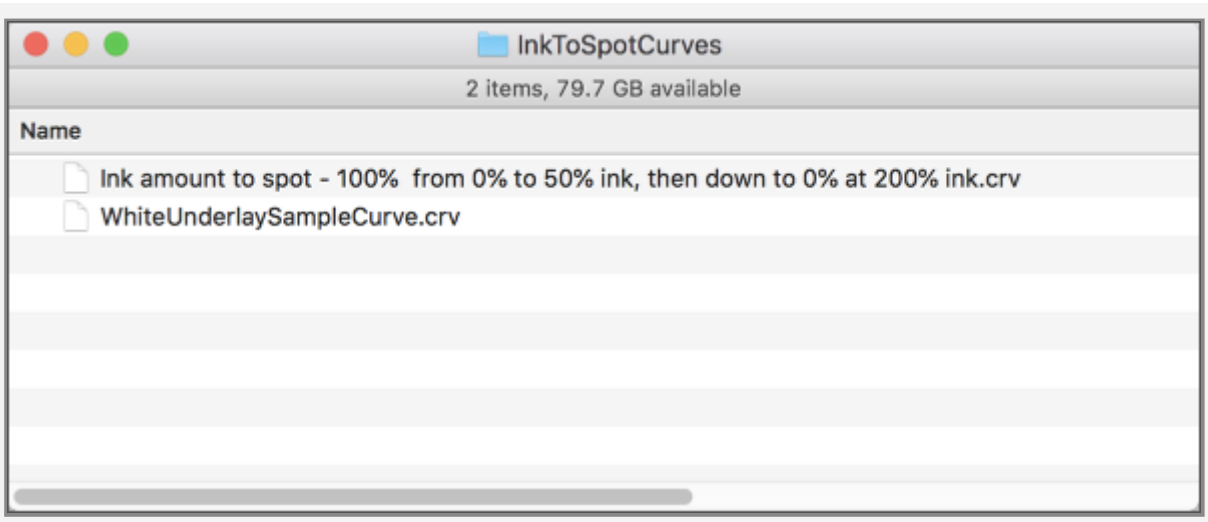
In order for this Fixup to work, a mapping curve is necessary that looks at the ink amount values across the page (at the configured resolution – this example uses 96 ppi; for crisper results value up to 300ppi may make sense) and uses them to look up the tint value to use for the spot color plate.

Click on the '+' or 'duplicate' button next to the "Ink to spot mapping curve" drop down:



This will take you to a mapping curve window. The easiest way to creating your own is to duplicate an existing one (like done above) (make sure to give a meaningful file name). Curve files can be found under preferences:

Users/User/Library/Preferences/callas\ software/callas\ pdfToolbox\ 13/Repositories/Custom/20210921_100038_0001/InkToSpotCurves/WhiteUnderlaySampleCurve.crv



In the example shown below the following has been done:

- Put the name of this curve file – as it should be displayed in the "Ink to spot mapping curve" drop down list – after "**DisplayName [tab] 0 [tab]**", the example uses "**Ink amount to spot - 100% from 0% to 50% ink, then down to 0% at 200% ink**"
- below the line with "INPUT [tab] [OUTPUT]" put three lines with the following content:
  - **0.0 [tab] 1.0**
    - → 0 % ink amount in the page will result in a 100% tint value on the spot color plate
  - **0.5 [tab] 1.0**
    - → 50 % ink amount in the page will result in a 100% tint value on the spot color plate; in addition,
      - all values between 0% and 50% ink amount will also result in a 100% tint value on the spot color plate
  - **2.0 [tab] 0.0**
    - → 200 % ink amount in the page will result in a 0% tint value on the spot color plate; in addition
      - all values between 50% and 200% ink amount will result in a tint value on the spot color plate going from 100% down to 0% (via linear interpolation)
      - all values above 200% ink amount will also result in a 0% tint value on the spot color plate

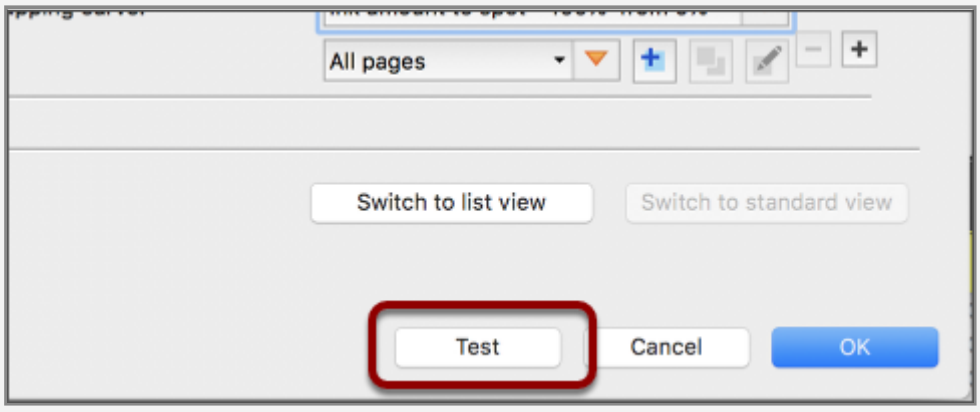
```
DisplayName 0   Ink amount to spot - 100% from 0% to 50% ink, then down to 0% at 200% ink

INPUT   OUTPUT
0.0 1.0
0.5 1.0
2.0 0.0
```

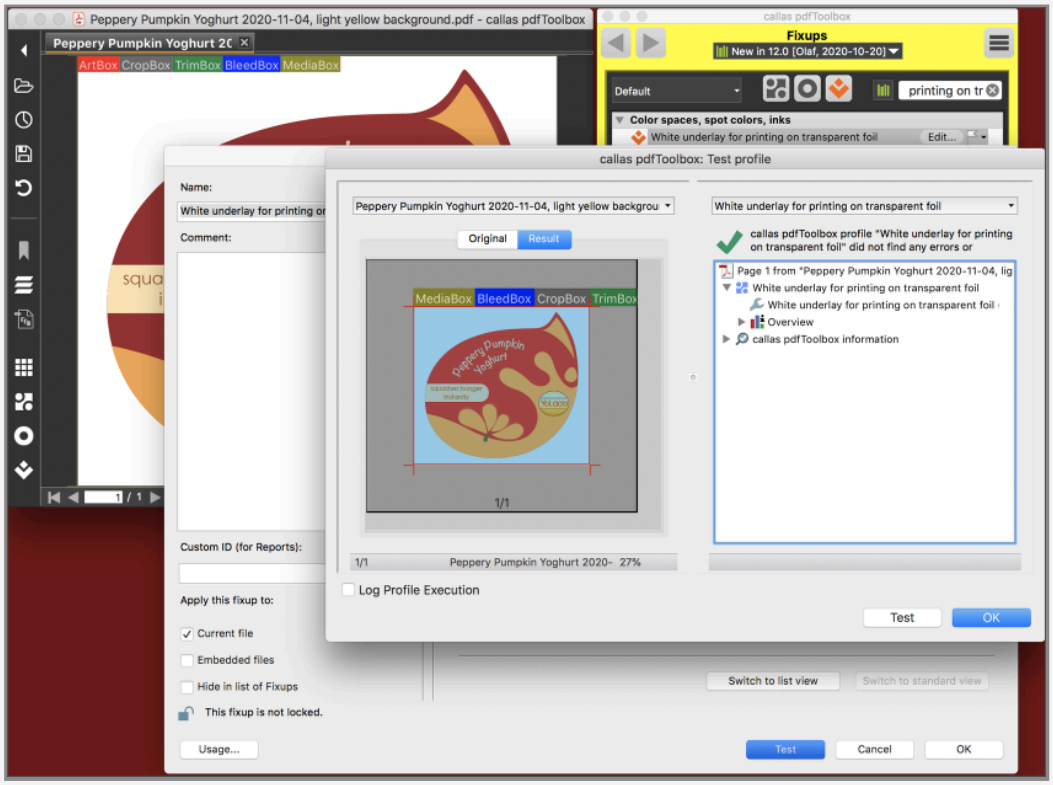
Once the new curve file has ben created, go back to pdfToolbox, save the fixup as configured so far and then reopen the fixup editing dialog – this is necessary to update the "Ink to spot mapping curve" drop down list. Now select the ink to spot mapping curve.

Testing the newly created "Create white underlays for printing on transparent foil" fixup

Instead of closing the fixup edit dialog and running the fixup on the example file – or any other file – it is recommended t use the "Test" mode as then you can stay in the fixup editing context and are in a position to easily adjust the configuration without having to close and reopen the dialog and saving out modified files to your hard disk again and again.

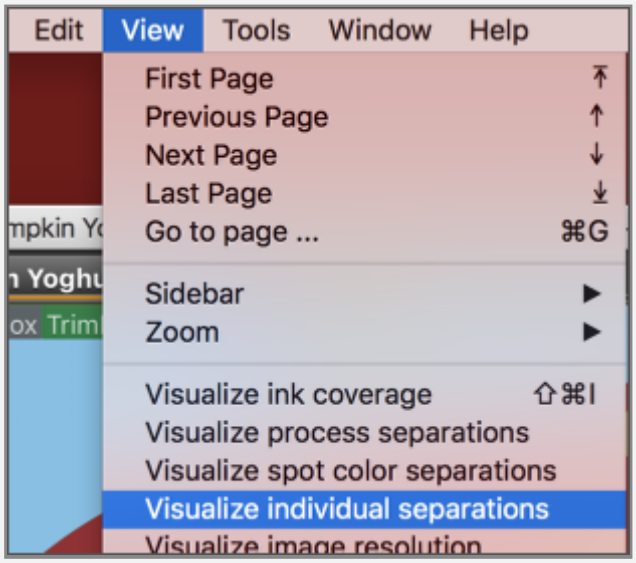


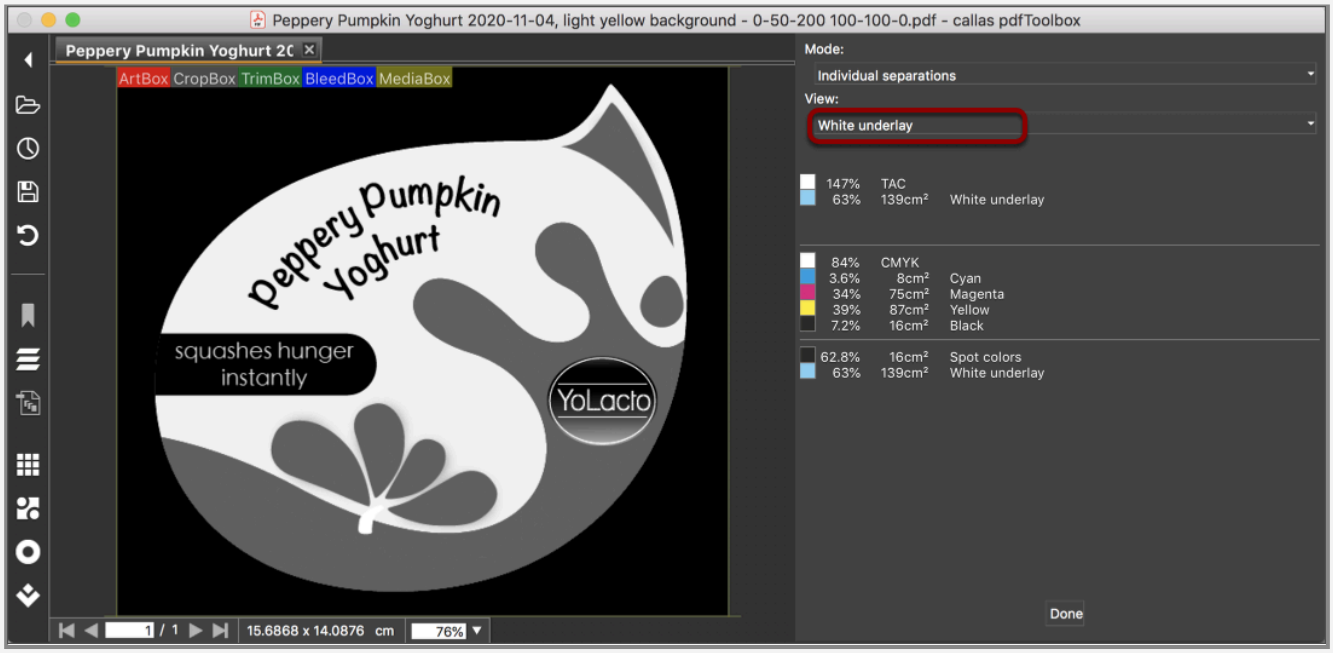
The result is displayed to the left – here the new spot color plate shows up in light Cyan colored overlay.



Additional control step: execute the fixup and inspect the new spot color plate using the "Individual separations" view

To be able to control the newly created spot color plate on its own, execute the fixup on the example file – or any other file – and inspect the result in the "Individual separations" view:





The "White underlay" spot color appears as a somehow inverted page image, with the darkest parts in areas where the page content is relatively light, and vice versa.

Create white underlay for packaging label printed on transparent foil (Part 2)

The approach explained in Part 1 runs into problems, if

- the print content contains white areas (e.g. white 'background' for inverted type) and transparent areas (e.g. content outside of the actual label, where nothing is to be printed at all)
- the desire is not to put any white underlay in areas where there is no print content at all but at the same time there are areas inside the print content where a white background is necessary; when printing on transparent foil (as opposed to more less white paper or more or less white paper like substrates) there is no pre-existing background. Instead depending on whatever the transparent foil will sit on will become the background, often in a not suitable manner.

In order to be able to distinguish between transparent areas and white areas, we ned to combine the fixup as illustrated above with a something that suppresses the "White underlay" in transparent areas of the page. This is achieved by created a shape reflecting the transparent areas of the page and applying a fill color – set to overprint – of 0% "White underlay". This will knock out the "White underlay" spot color (in transparent areas of the page) but will disturb any other content. The steps below illustrate how to set up the Shape based fixup.

As a further refinement, the shape generation can be configured such that the"White underlay" bleeds into the transparent areas by a few millimeters to compensate for registration issues or to make the print content stand out more on the transparent foil.

As this additional shape generation fixup needs to be applied right after creating the spot color plate for the "White underlay", it is best not use it on its own but instead to create a simple Process Plan that applies both fixups one after the other. The Process Plan is attached below.

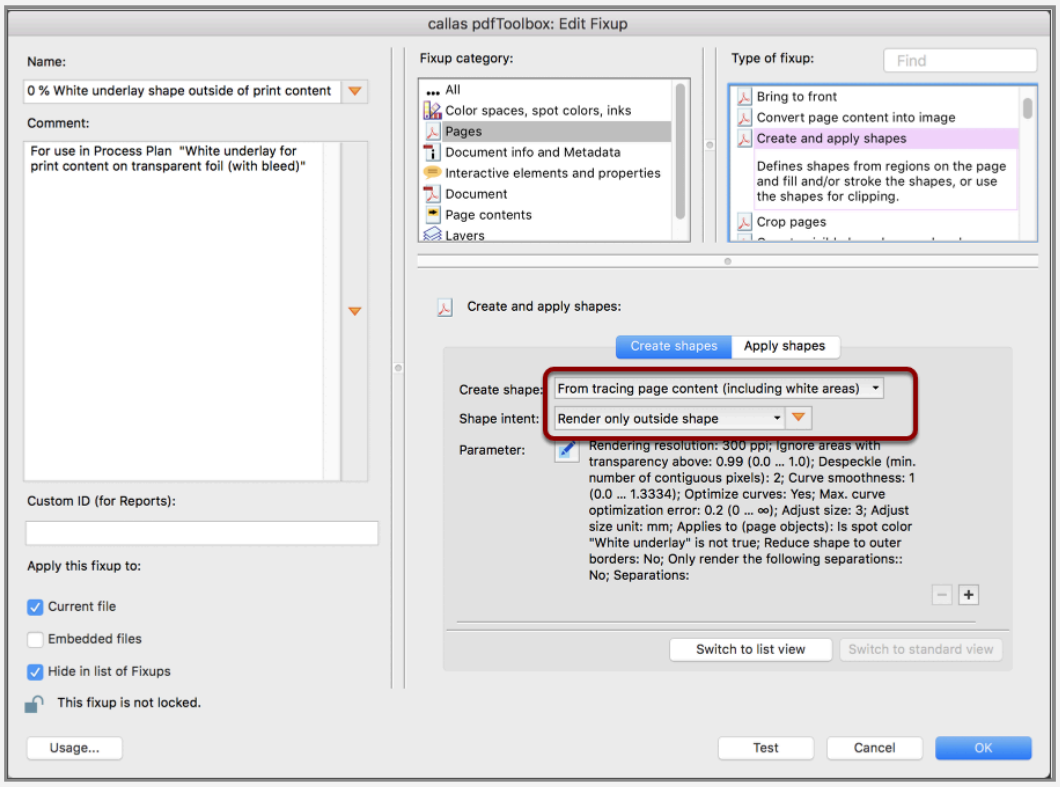
Also, a slightly different version of the "Peppery Pumpkin Yoghurt" label is attached below – it has white background in some parts of the print content, while the page is transparent outside of the intended print content.



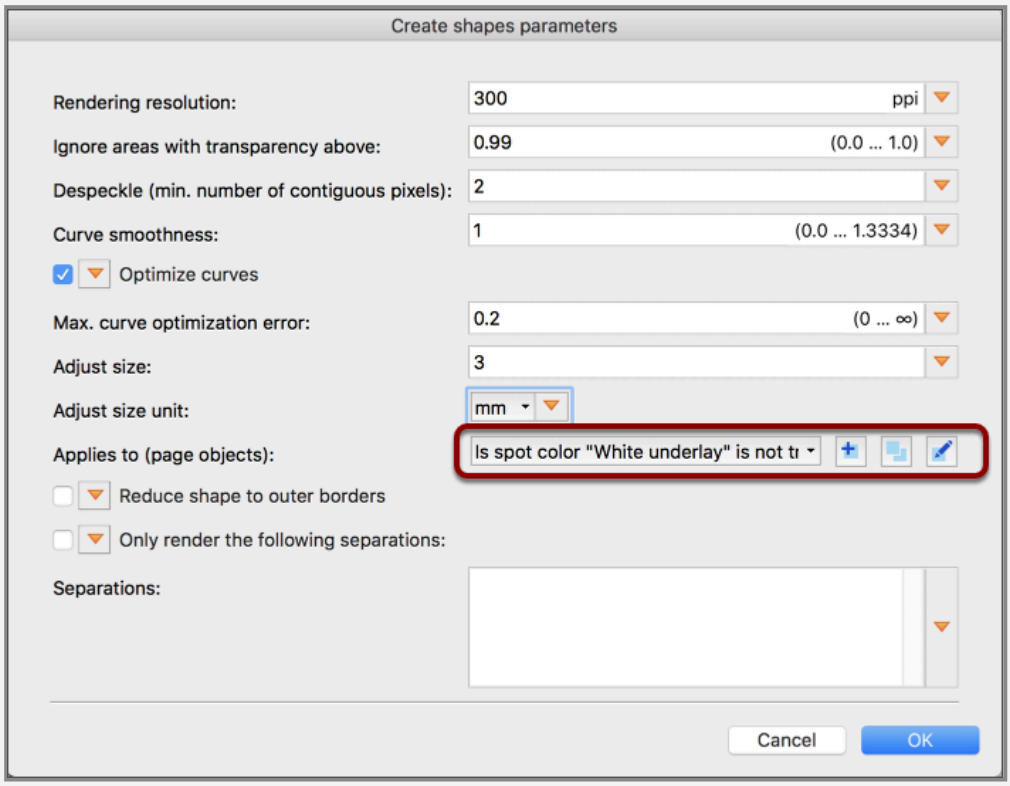
Peppery_Pumpkin_Yoghurt_2020-11-04__white_back.pdf

White_underlay_for_print_content_on_transpare.kfpx

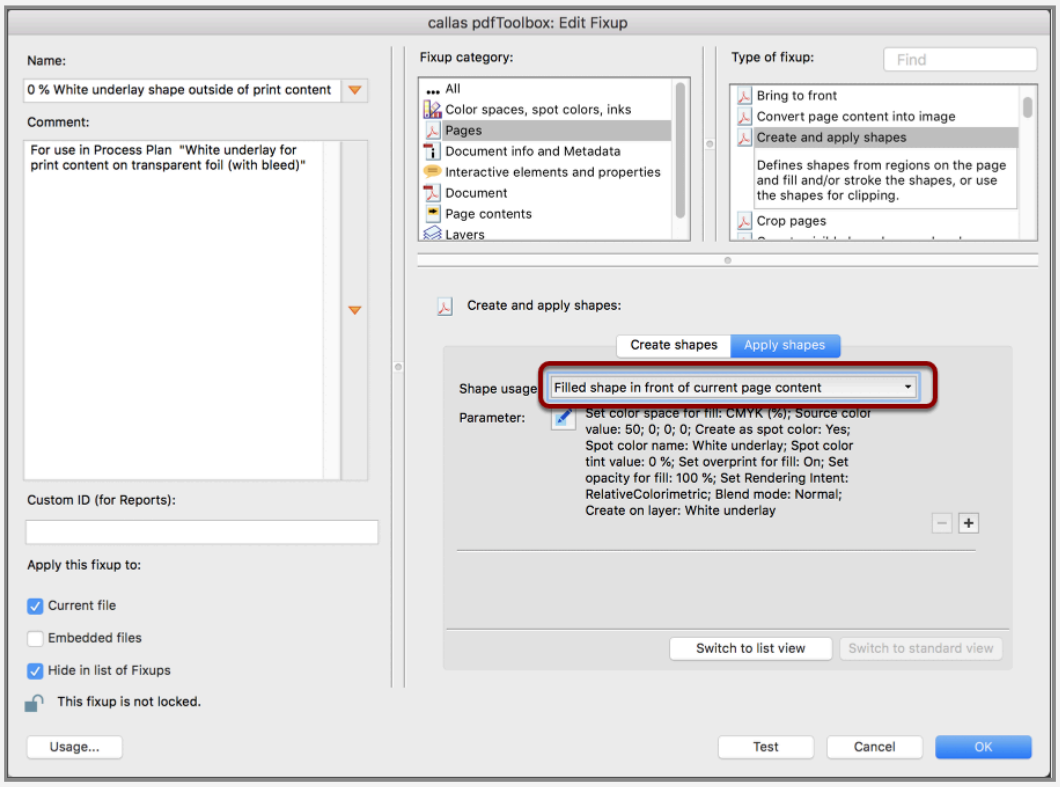
Create a new fixup for creating a Shape reflecting transparent areas of the page content



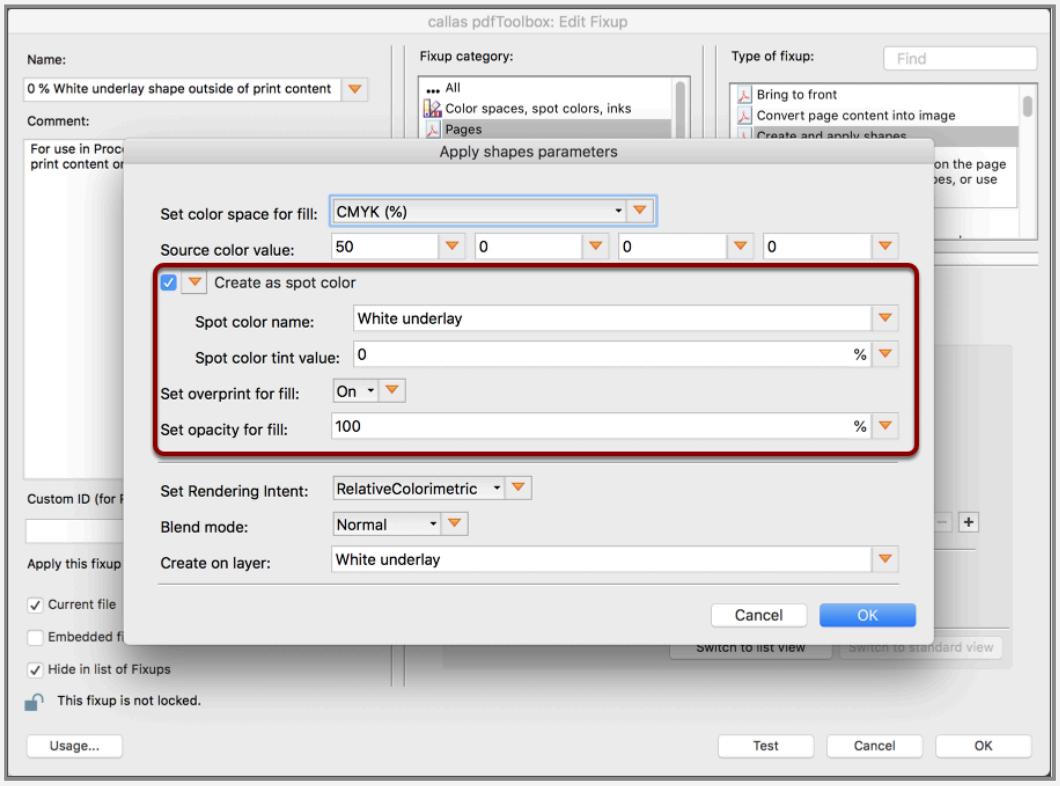
Make sure to exclude the already created "White underlay" spot color when determining transparent areas



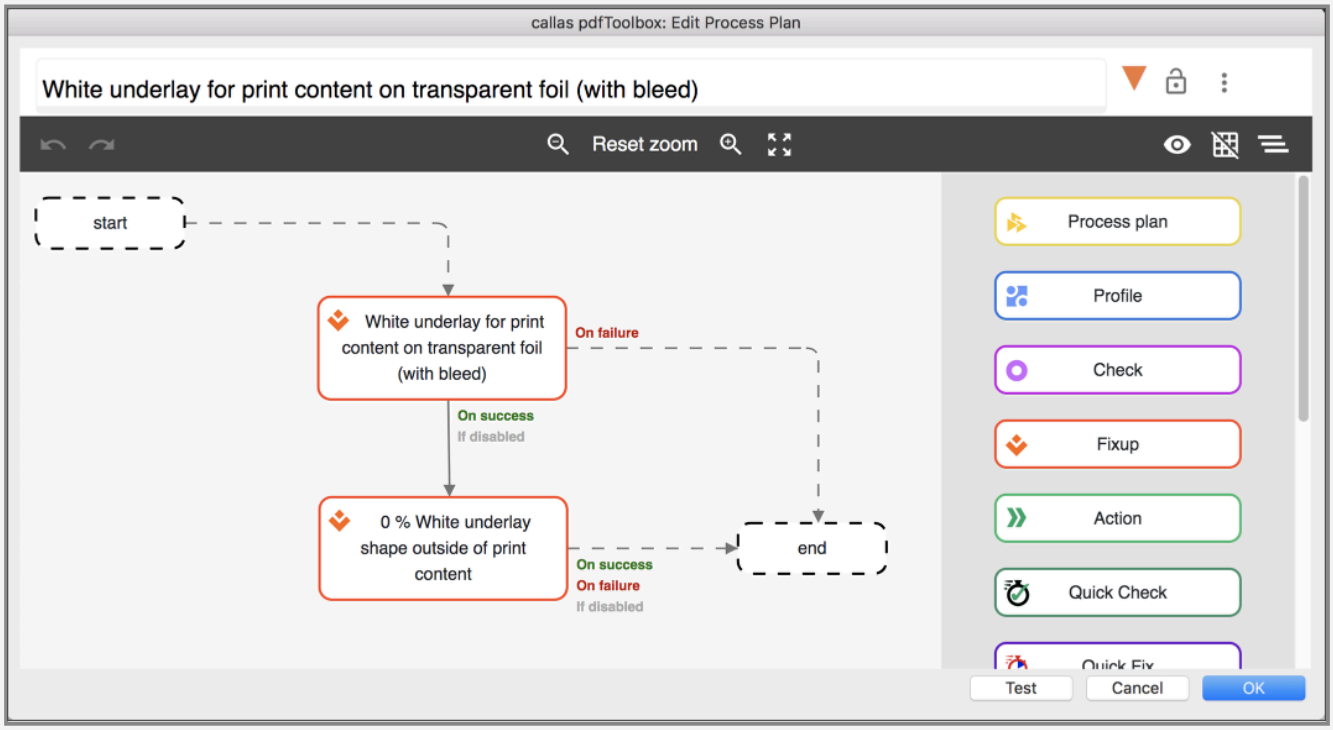
Configure how to fill the shape



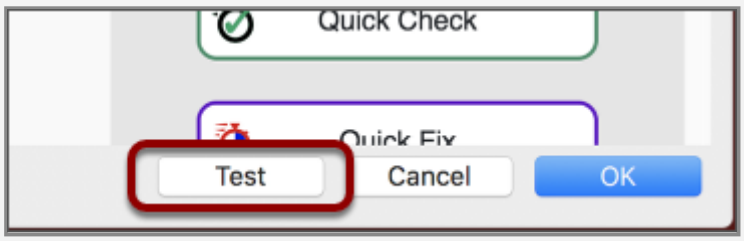
Fill the shape with 0% of "White underlay" spot color

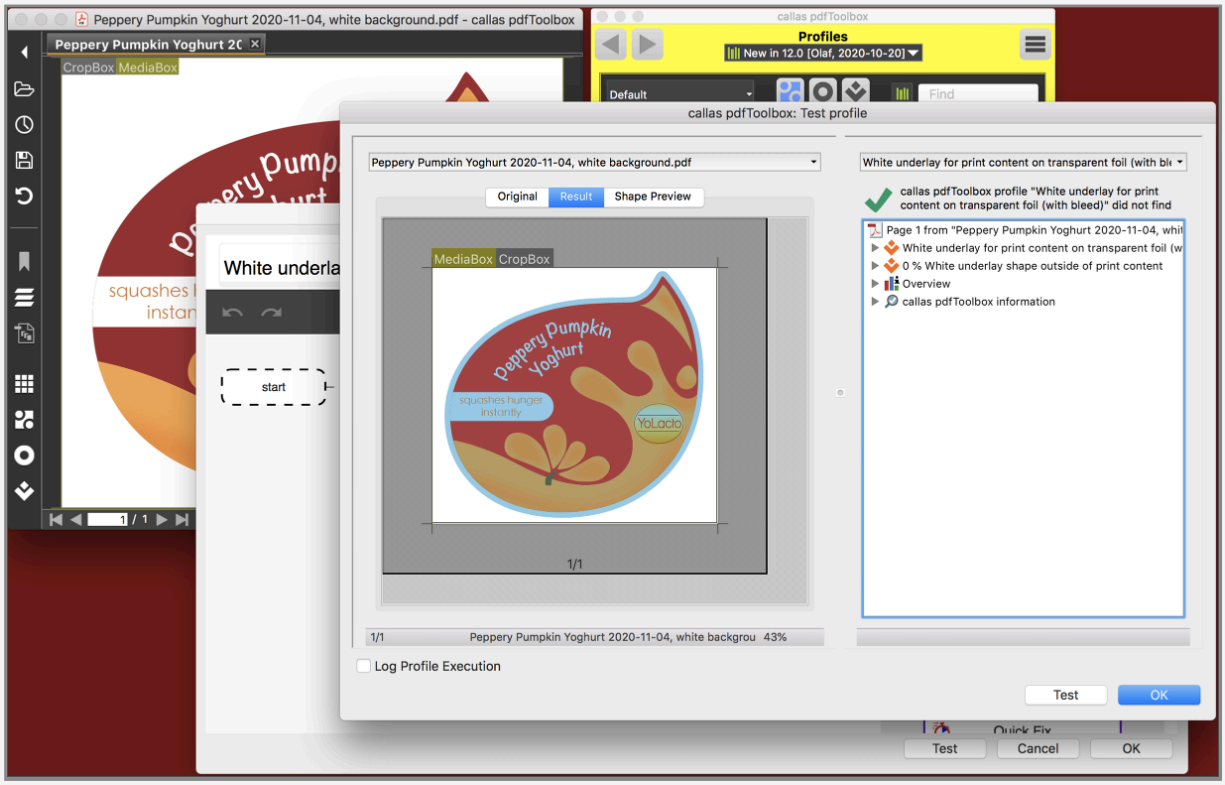


Bring it all together in a Process Plan



Testing the new Process Plan using the Test feature





Using the Process Plan on the label variant with white areas in the print content and a transparent area around the print content creates a "White underlay" with a tint value of 100% in white areas, but with 0% "White underlay" outside the intended print content. In addition, 3mm of "bleed" are created for the "White underlay" spot color.

13.11 Derive spot color from softmasks

This Fixup creates new images with a specified spot color derived from existing softmasks.

It can be used if the original is a PNG with an alpha channel or a PDF with a softmask. Since the alpha channel/softmask controls transparency, it can be used to derive e.g. a white underlay where the amount of white depends on the amount of visible color.

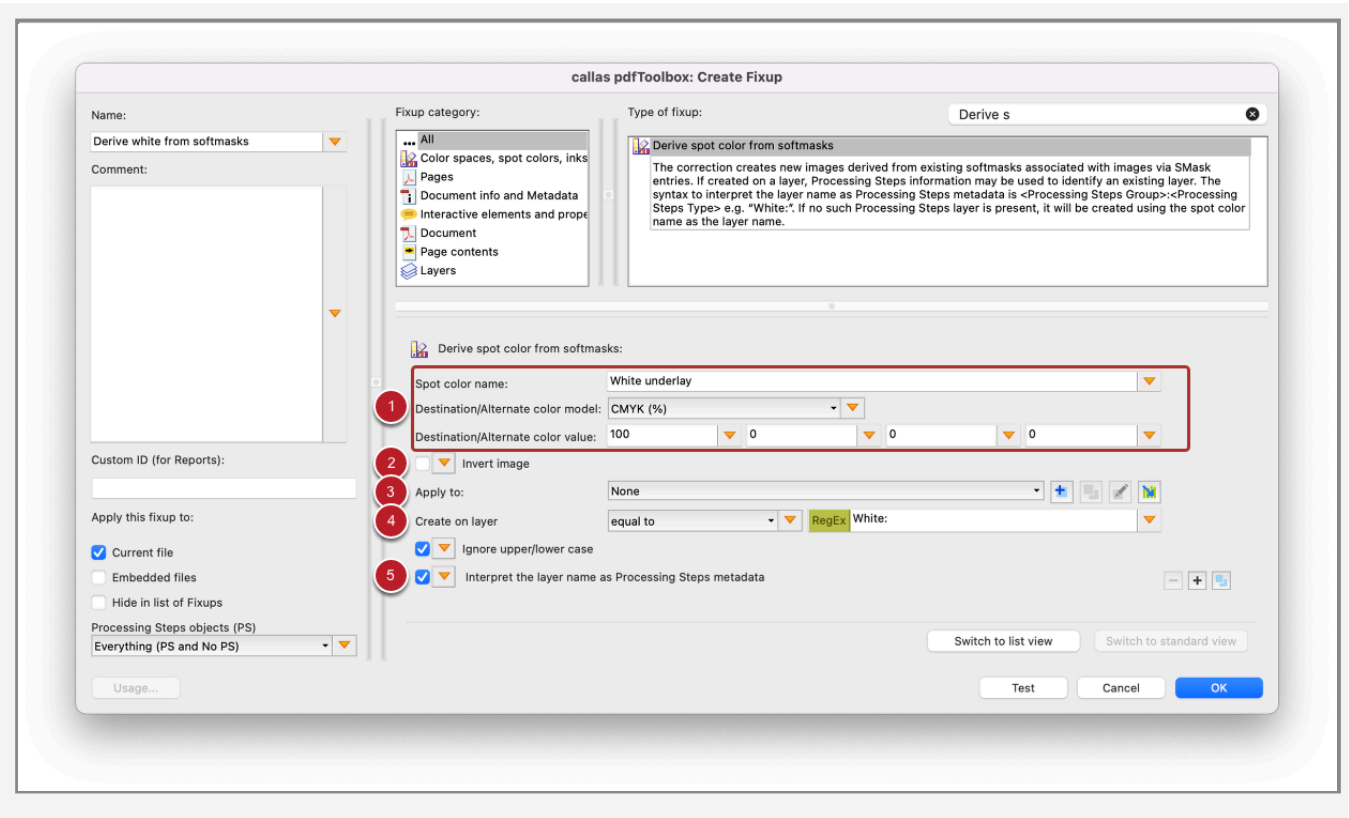
Note: If a PNG file with an alpha channel is used as input, pdfToolbox will convert the PNG file into a PDF with a softmask.

Example: Derive a white underlay from softmasks

The test file is a PNG with an alpha channel. The background is transparent, and some of the objects in the background also have different levels of transparency. With the Fixup "Derive white from softmasks", these transparency levels from the softmask are used to create a white underlay.

Testfile_callas_softmask.png

Derive white from softmasks.kfpx



1. Parameters to define the spot color.
2. An "Apply to" filter can be used to e.g. limit the Fixup to certain pages.
3. To apply the spot color everywhere except the existing softmask, the checkbox must be activated.
4. Layer options: If a layer name is specified, the newly derived images will be placed on a layer. Alternatively, Processing Steps metadata can be defined here. The syntax to interpret the layer name as Processing Steps metadata is <Processing Steps Group>:<Processing Steps Type> e.g. “White:”. If no such Processing Steps layer is present, the layer will be created using the spot color name as the layer name.
5. If the layer name should be interpreted as Processing Steps metadata, the checkbox must be activated.

Result

With the parameters set in the Fixup above, the result looks like this:

- A new image in the spot color "White underlay" has been added to the PDF file (the image always has OPM=0 > Overprint: true).
- The white underlay reflects the transparency levels of the softmask: More transparent print content has less ink coverage of the spot color, more opaque print content has more ink coverage of the spot color.
- The white underlay is placed on a new layer (layer name is the same as the spot color name).
- The layer has the Processing Step OCG "White" to indicate the application of white backing ink



13.12 Ink coverage Check properties

pdfToolbox offers several Check properties that allow you to determine the ink on a page. These properties work in a very similar way and can be found in the Group "Pages". Internally the page is rendered off-screen (converted to a number of pixels) and then the pixels are analyzed by the engine according to the values set in the Check.

In pdfToolbox all such Check properties carry the word "effective" in their name because they determine the effective color after resolving all interdepedendicies between objects.

Overview of the different ink coverage Check properties

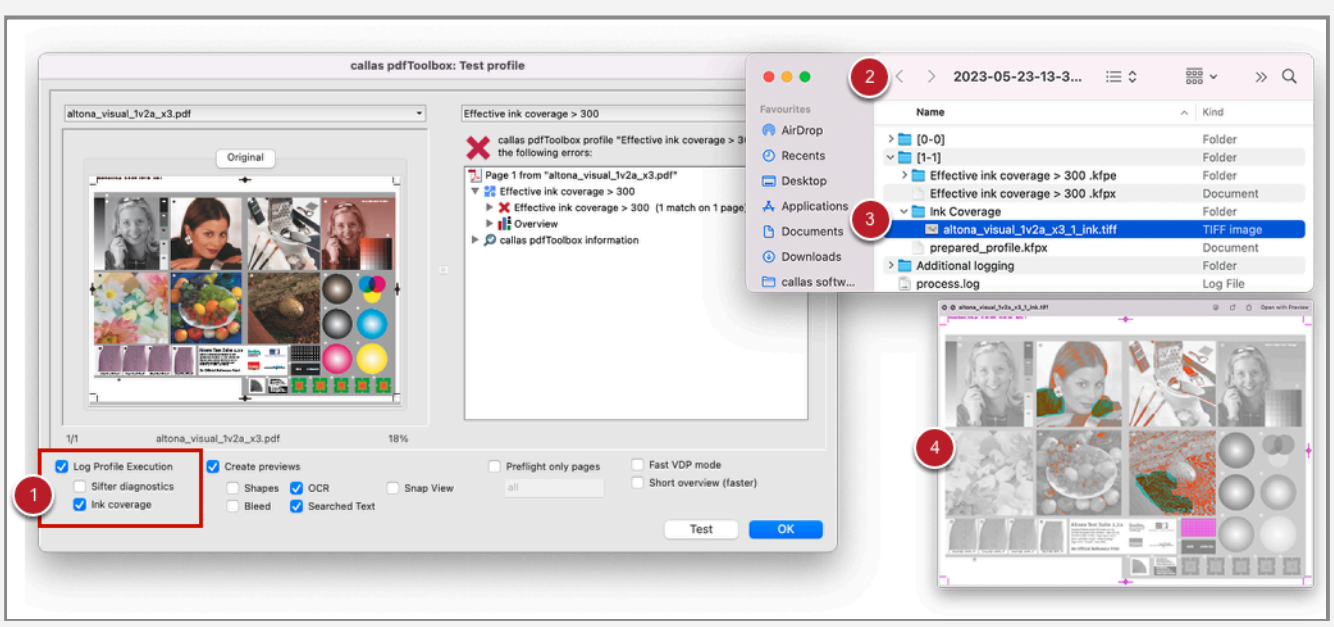
There are properties that help you to determine ink related parameters, these are "Effective ink coverage" and "Effective total area coverage (TAC)".

And there are properties that can be used to get information related to plates and separations: "Effective ink coverage for separated plates" (is there anything that is rendered on a plate), "Effective ink coverage for registration color" (is there registration color) and "Number of effectively non-empty plates". Similar works "Area effectively covered by ink amount above threshold", but here you are more interested in the covered area than in the total ink.

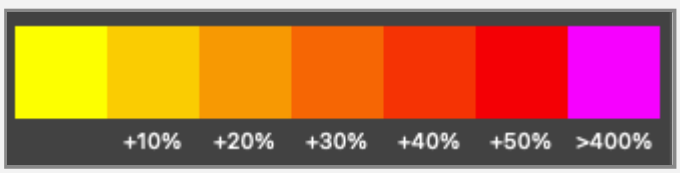
"Effective ink coverage - limit to custom area" is a general helper property that allows you to limit the area. "Effective ink coverage in custom area" should not be used anymore and will be deprecated at some point, it is just a special case of the more general helper property approach.

Log profile execution for ink coverage previews

To get started, there is a helpful tool to visualize the results of the Check:

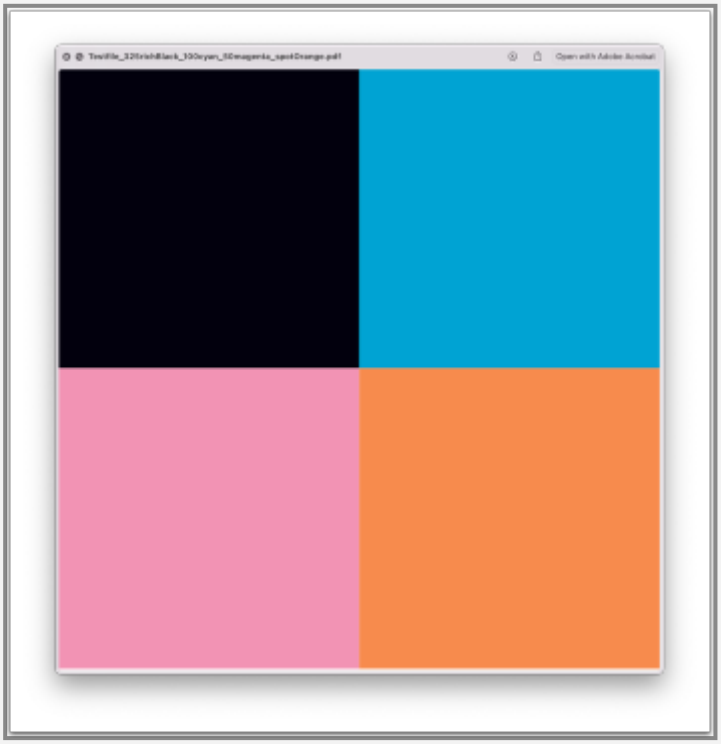


1. In Test Mode, you can check the "Log Profile Execution" and the "Ink coverage" checkboxes.
2. When the test mode is activated and an ink coverage hit is reported, a log profile execution with an ink coverage preview image is generated.
3. The ink coverage previews are stored in a folder named 'Ink coverage'.
4. The affected areas are highlighted in color. This makes it easier to see where in the PDF document the threshold is triggered. The same color coding scheme as the Ink coverage visualiser in 'Highlighting 'mode' is used. In addition, a Cyan overlay is added to the areas where the threshold was triggered:



To better explain the following ink amount Check properties, a simple test file is used to better demonstrate how the properties work. The test file is constructed as follows:

- size of the page: 10 x 10 cm
- one color patch (5 x 5 cm) in rich black (100 % Cyan, 100 % Magenta, 100 % Black, 25 % Yellow)
- one color patch (5 x 5 cm) in 100 % Cyan
- one color patch (5 x 5 cm) in 50 % Magenta
- one color patch (5 x 5 cm) in spot color Orange

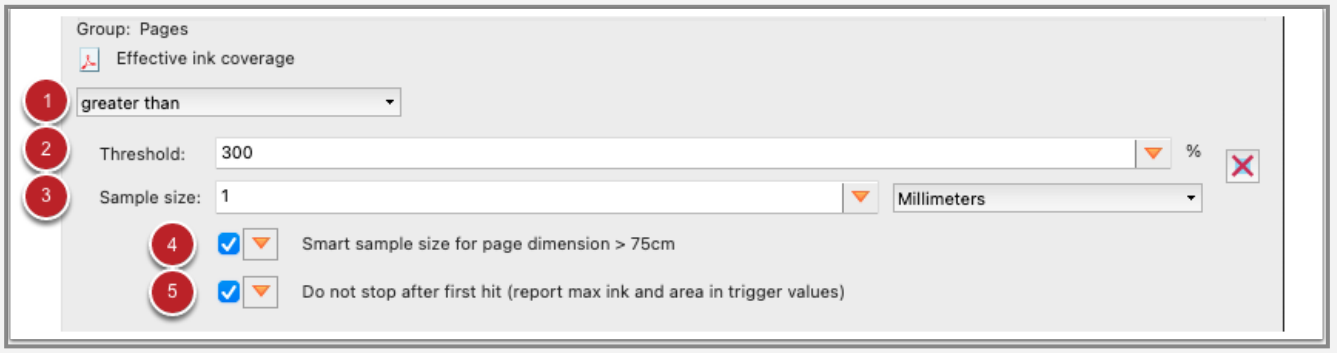


Testfile_325richBlack_100cyan_50magenta_spotOr.pdf

"Effective ink coverage"

This Check property determines whether the total ink coverage is outside of the threshold. It allows you to identify

whether a PDF document has areas on a page where there is too much ink to prevent potential printing problems.



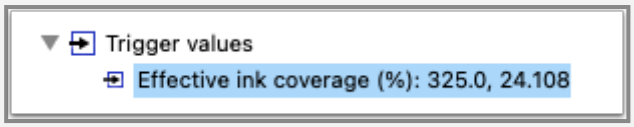
1. Operator for the Threshold.
2. Threshold in % for ink coverage.
3. The size of the sample defines the size of the relevant area i.e. the number of adjacent pixels taken into account for measuring the actual ink amount. The smaller the sample size, the more accurate the trigger values (process time increases).
4. Enables auto adjustment of the sample size for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.
5. If the checkbox is checked the whole page will be evaluated. If the checkbox is unchecked the evaluation is stopped after the first hit that is above the threshold. The trigger values of the Check result depend on whether the checkbox is activated or not. If you are interested in the exact trigger values, the checkbox should be checked.

- With enabled checkbox, two trigger values are provided: 325.0% (max. ink coverage), 24.108% (total area above threshold)

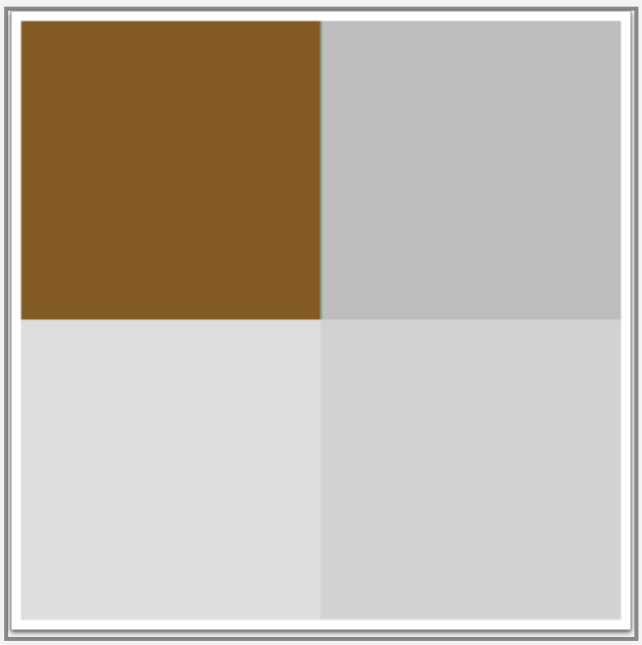
Evaluation of the Check result of the sample file

In this example, an effective ink coverage Check was created to generate a hit when a page has a higher ink coverage than 300 %. Since the checkbox "Do not stop after first hit" is set in the Check property, the first trigger value indicates the maximum ink coverage in % that was found on the page and the second value indicates the total area in % where the ink coverage is above the threshold.

-> The highest ink coverage on the page is 325 % (rich black patch: 100 % C, 100 % M, 100 % K, 25 % Y). This max. ink coverage covers ~25 % of the total area (the slight rounding errors are caused by the rendering of the page).

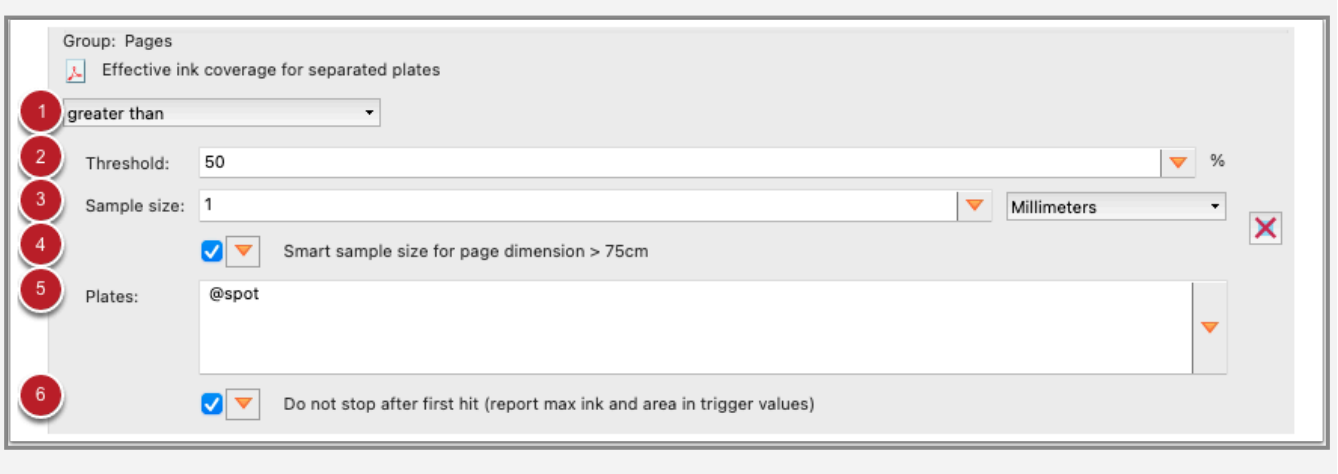


The ink coverage preview from the log profile execution highlights all areas where the effective ink coverage is above 300 % (rich black patch):



"Effective ink coverage for separated plates"

This Check is a variation of the "Effective ink coverage" property. It allows you to limit the Check to one or more specific plates so that only those plates are considered for the ink coverage. It can be used to determine if a particular ink is being used at all, or if a particular amount of ink is being used on a particular plate.



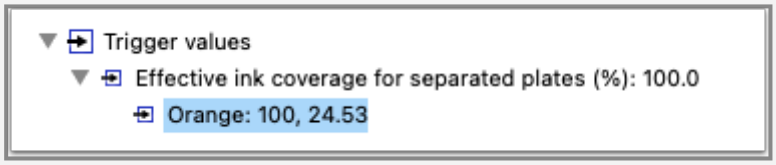
1. Operator for the Threshold.
2. Threshold in % for ink coverage. The threshold defines the minimum ink amount for a plate in order to be taken into account.
3. The size of the sample defines the size of the relevant area i.e. the number of adjacent pixels taken into account for measuring the actual ink amount. The smaller the sample size, the more accurate the trigger values (process time increases).
4. Enables auto adjustment of the sample size for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.
5. In the list of plates @spot or @process may be used in order to render all spot color plates or all process color plates. If more than one plate is to be specified, they must be entered in the list one below the other.
6. If the checkbox is checked the whole page will be evaluated. If the checkbox is unchecked the evaluation is stopped after the first hit that is above the threshold. The trigger values of the Check result depend on whether the checkbox is activated or not. If you are interested in the exact trigger values for each plate, the checkbox should be checked.

- With enabled checkbox, two trigger values are provided: 100% (max. ink coverage), 24.53% (total area above threshold)

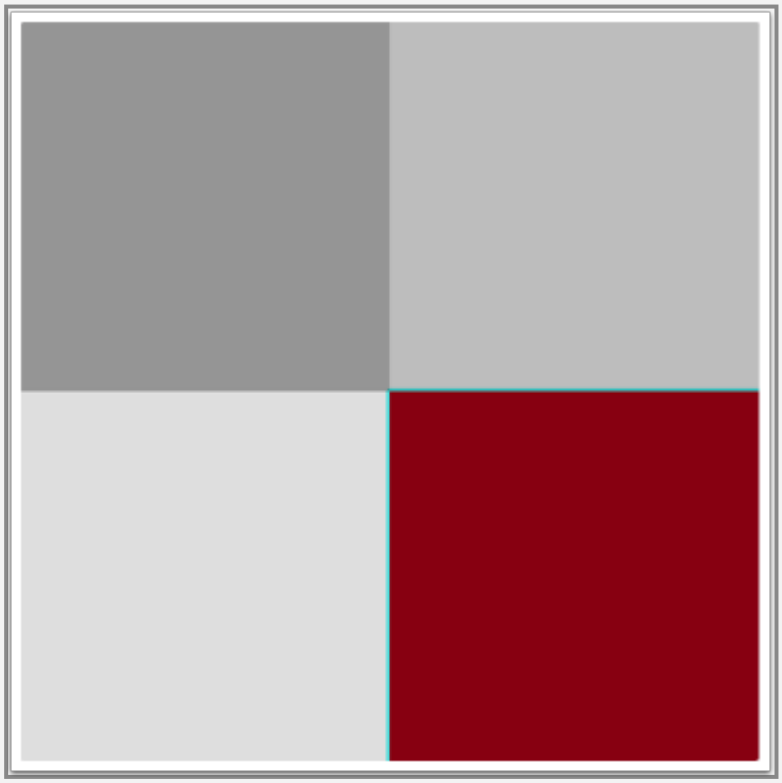
Evaluation of the Check result

In this example, an effective ink coverage Check for all spot colors was created. All process colors are not taken into account. A hit will be created, if the effective ink coverage for a spot color is above 50 %. The trigger values list each spot color above the threshold with its highest ink coverage value (the maximum trigger value per plate cannot exceed 100 %).

-> The highest ink coverage for the spot color Orange is 100 %. This max. ink coverage covers ~25 % of the total area (the slight rounding errors are caused by the rendering of the page).



The ink coverage preview from the log profile execution highlights all areas where the effective ink coverage of a spot color is above the threshold (Orange spot color patch):

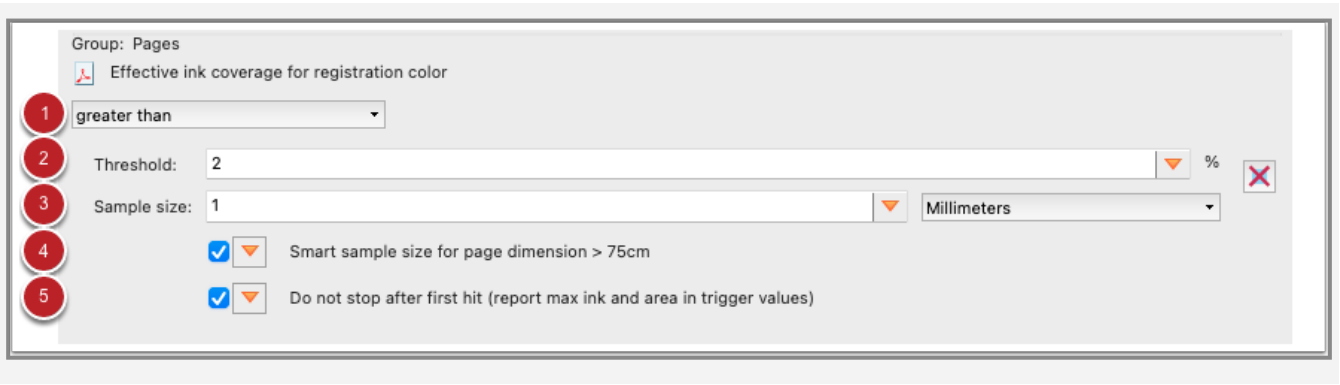


"Effective ink coverage for registration color"

For this Check, Crop Marks, Registration Marks and Page information in Registration "All" were added to the PDF .

Testfile_325richBlack_100cyan_50magenta_spotOr.pdf

It is determined whether any pixels using registration color are present in the document. A hit means that the actual amount of ink for such registration color objects is outside of the thresholds. Typical values are 1-2 % for the threshold.



1. Operator for the Threshold.
2. Threshold in % for ink coverage. The threshold defines the minimum ink amount for registration color in order to be taken into account.
3. The size of the sample defines the size of the relevant area i.e. the number of adjacent pixels taken into account for measuring the actual ink amount.
4. Enables auto adjustment of the sample size for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.
5. If the checkbox is checked the whole page will be evaluated. If the checkbox is unchecked the evaluation is stopped after the first hit that is above the threshold. The trigger values of the Check result depend on whether the checkbox is activated or not. If you are interested in the exact trigger value, the checkbox should be checked.

- With enabled checkbox, two trigger values are provided: 58.505% (max. ink coverage), 1.64% (total area above threshold)

Evaluation of the Check result

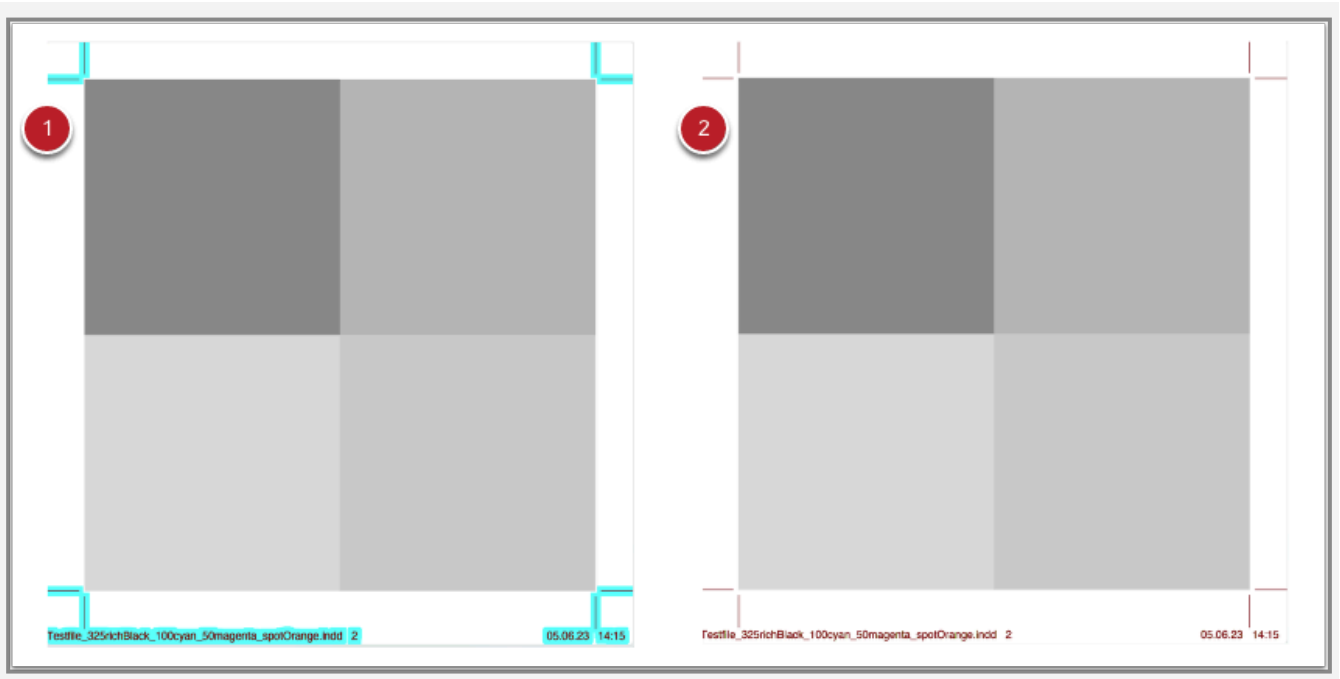
This example illustrates the effect of sample size on trigger values. Especially when dealing with thin page objects such as text and lines, a very small sample size must be set if you want to obtain accurate trigger values:

1. Sample size 1 mm: Because of the page rendering, the ink coverage for separation "All" is reported with 58.505 %.
2. Sample size 0.1 pt: With this sample size, accurate trigger values are obtained. The ink coverage for separation "All" is reported with 100 %.



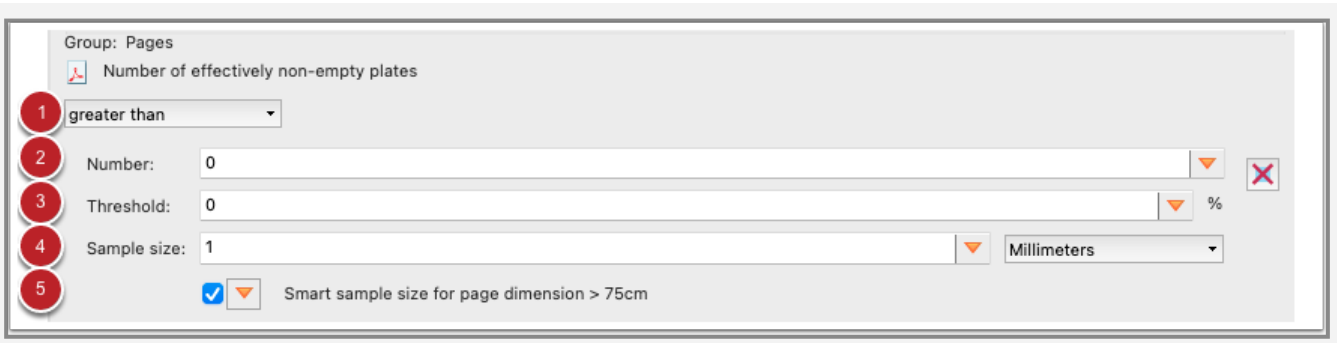
The ink coverage preview from the log profile execution also shows why the sample size affects the trigger values:

1. Sample size 1 mm
2. Sample size 0.1 pt



"Number of effectively non-empty plates"

For this property every plate is rendered internally and it is determined whether there is an ink coverage within any area of the plate that is above the threshold. All plates with an ink coverage above the threshold are added to the number of effectively non-empty plates (that means that the Check will count all effectively used plates).



1. Operator for the Threshold.
2. Total number of non-empty plates.
3. Threshold in % for ink coverage. The threshold defines the minimum ink amount for a plate in order to be taken into account.
4. The size of the sample defines the size of the relevant area i.e. the number of adjacent pixels taken into account for measuring the actual ink amount.
5. Enables auto adjustment of the sample size for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.

- One trigger values is provided: 5 (total number of non-empty plates), additionally all plate names are listed

Evaluation of the Check result

In this example, all effectively used plates will be listed in the trigger value since the "number" and "threshold" parameters were set to 0. The trigger value reflects the total number of effectively non-empty plates (in this example 5). Below that all plates are listed with their names.

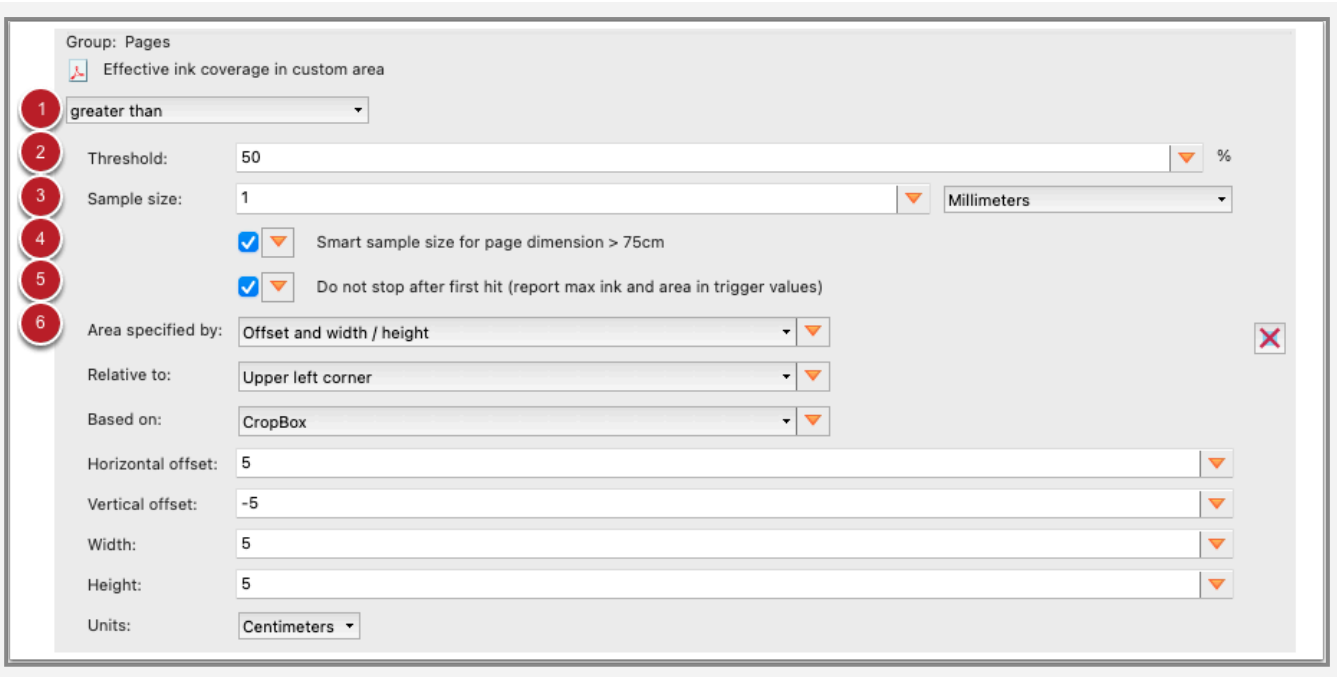


The ink coverage folder of the log profile execution contains a separate ink coverage preview for each non-empty plate that is above the threshold. In this example, you get 5 previews: Black, Cyan, Magenta, Yellow, Orange.

"Effective ink coverage in custom area"

This Check property should not be used anymore and will be deprecated at some point. Instead, the Check property "Effective ink coverage - limit to custom area" should be used in combination with the "Effective ink coverage" Check.

It is determined whether the total ink coverage is outside of the threshold. Additionally, the area which shall be used for analysing can be defined as a rectangle. The rendering for the ink coverage evaluation is then restricted to the custom area. Note: this should not be combined with other properties or settings that limit the area to be analysed.



1. Operator for the Threshold.
2. Threshold in % for ink coverage.
3. The size of the sample defines the size of the relevant area i.e. the number of adjacent pixels taken into account for measuring the actual ink amount. The smaller the sample size in the check property, the more accurate the trigger values will be.

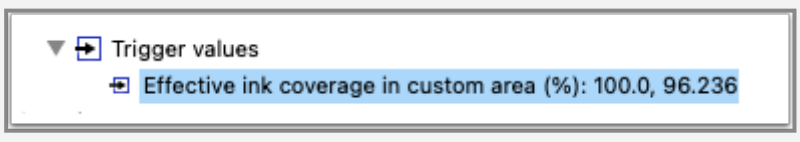
4. If the checkbox is checked the whole page will be evaluated. If the checkbox is unchecked the evaluation is stopped after the first hit that is above the threshold. The trigger values (5a) of the Check result depend on whether the checkbox is activated or not. If you are interested in the exact trigger values, the checkbox should be checked.
5. Enables auto adjustment of the sample size for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.
6. There are two methods to define the custom area:
   - **"Offset and width / height"**: "Relative to" sets the origin point. All subsequent parameters are based on this origin (e.g. if the origin point is set to top left corner, a negative vertical offset will move the origin down, while a positive offset will move the origin up).
   - **"Offsets to box edges"**: Takes the size of a page geometry box as a reference for the custom area. Negative values reduce the size of the page geometry box. Positive values increase the size.

- With enabled checkbox (4), two trigger values are provided: 100% (max. ink coverage in custom area), 96.236% (total area above threshold in custom area)

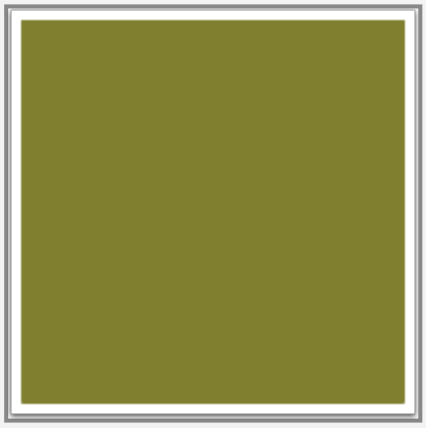
Evaluation of the Check result

In this example, an effective ink coverage Check was created to generate a hit when the ink coverage is greater than 50 %. The origin of the custom area is offset horizontally and vertically from the top left corner of the CropBox. The custom area has a size of 5 x 5 cm. With these values, the ink coverage Check is limited to the Orange spot color patch. Therefore, the trigger values are as follows:

-> The highest ink coverage in the custom area is 100 % (Spot color Orange). This max. ink coverage covers ~100 % of the custom area (the slight rounding errors are caused by the rendering of the page).



The ink coverage preview from the log profile execution shows only the predefined custom area (in this case only the lower right color patch in spot color Orange).

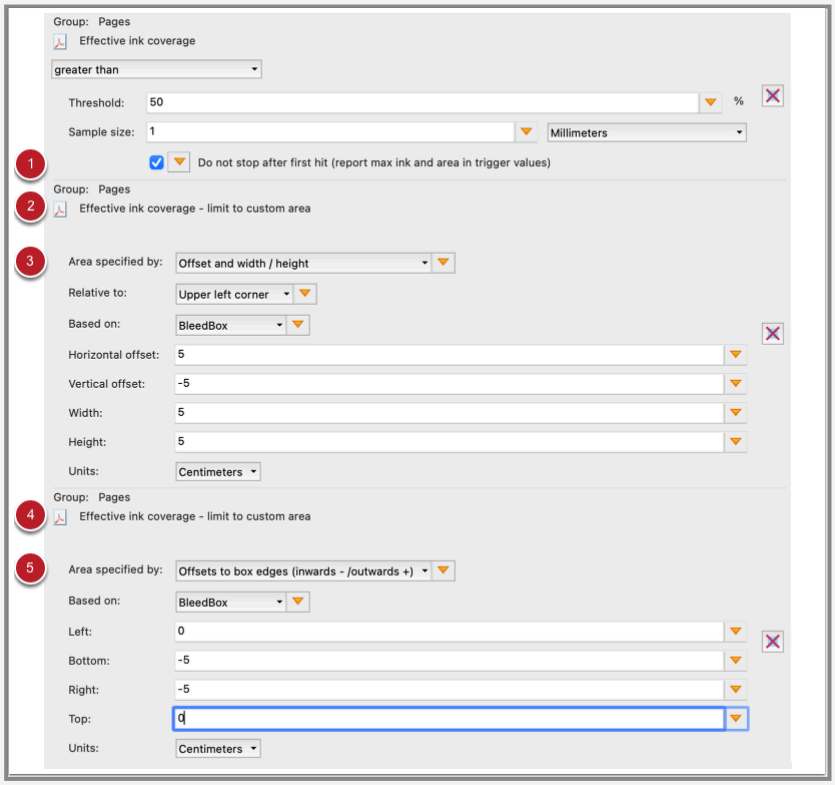


"Effective ink coverage - limit to custom area"

This Check property can be combined with any of the other ink coverage Check properties (except “Effective ink coverage in custom area“). This Check property will never directly create any hit, it has to be added to another ink coverage Check property. If an ink coverage Check is limited to a custom area the rendering for the ink coverage evaluation is restricted to the specified area. The property can also be used multiple times inside a Check in order to specify multiple areas.

Effect of “Fire if any condition is met” setting

- If the checkbox is checked the Check generates a hit if at least one area satisfies the Check condition (OR).
- If the checkbox is unchecked the Check generates a hit if all areas satisfies the Check condition (AND).



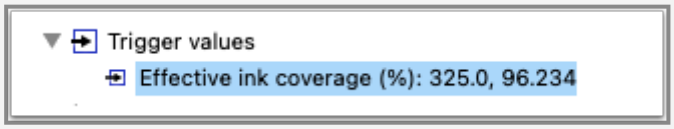
1. If the checkbox is checked all areas are evaluated. If the checkbox is unchecked the evaluation is stopped after the first area that generates a hit.
2. First "Effective ink coverage - limit to custom area" Check property to limit the ink coverage Check to a custom area
3. Two methods to define the custom area: 1. Offset and width / height:
   "Relative to" sets the origin point. All subsequent parameters are based on this origin (e.g. if the origin point is set to top left corner, a negative vertical offset will move the origin down, while a positive offset will move the origin up).
4. Second "Effective ink coverage - limit to custom area" Check property to limit the ink coverage Check to a custom area
5. Second method to define the custom area: Offsets to box edges (inwards - /outwards +):
   Takes the size of a page geometry box as a reference for the custom area. Negative values reduce the size of the page geometry box. Positive values increase the size.

- With enabled checkbox (1), two trigger values are provided: 325.0% (max. ink coverage in custom area), 96.234% (total area above threshold in custom area)

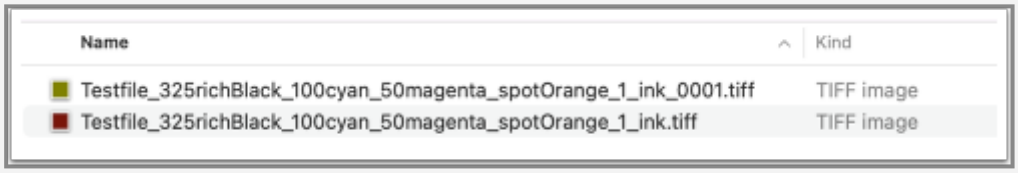
Evaluation of the Check result

The first "Effective ink coverage - limit to custom area" Check property defines the area of the Orange spot color patch. The second property defines the area of the rich black color patch. The trigger value reports the highest ink coverage:

-> The highest ink coverage in one of the custom areas is 325 % (rich black patch: 100 % C, 100 % M, 100 % K, 25 % Y). This max. ink coverage covers 96.236 % of the custom area (the slight rounding errors are caused by the rendering of the page).



The ink coverage previews from the log profile execution shows only the predefined custom areas. Every predefined custom area will create a new ink coverage preview. Each preview highlights all areas where the ink coverage is above the threshold. If the "Do not stop after first hit" checkbox had been checked, the ink coverage evaluation would have stopped after the first hit and only one preview would have been generated.



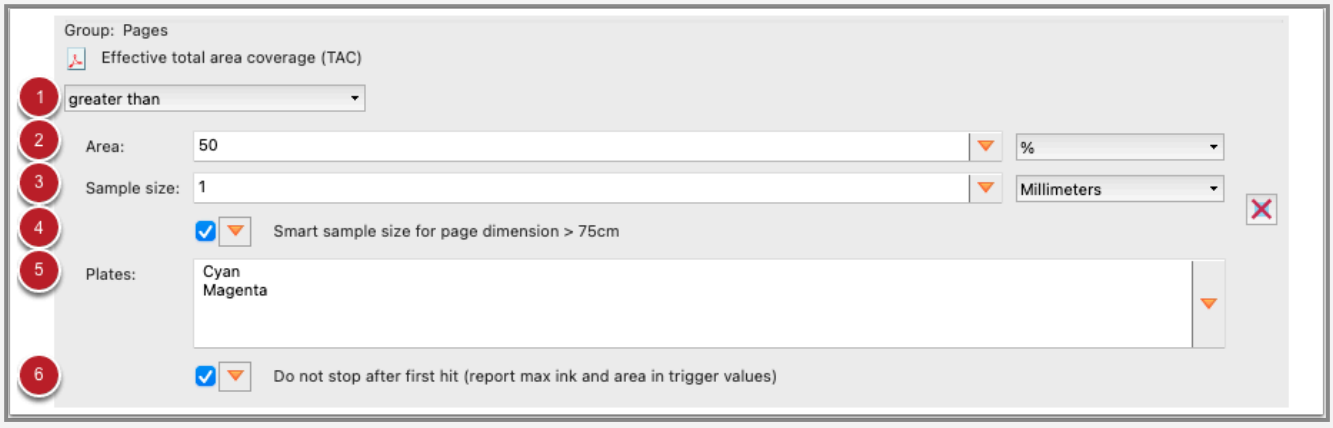
For the following two ink amount Check properties, another test file is used to better demonstrate how the properties work. The test file now only contains

two color swatches: 50 % of the total page is covered with 100 % Cyan and the other half (also 50 %) is covered with 50 % Magenta:

50Cyan_25Magenta.pdf

"Effective total area coverage (TAC)"

Total area coverage is the size of the area that would be filled if all used ink would be printed with 100 %. It shows the ink usage for one or more plates as a percentage of the total page. For example, if the entire page is covered with, 50% Cyan, it will have the same Total Area Coverage value as if the page were only half covered but with 100% Cyan. The ink usage of the defined plates is added up. This means, for example, that the highest trigger value for all process colors could be 400%.



1. Operator for the threshold.
2. Size of the area that should be taken into account.
3. The size of the sample defines the size of the relevant area. i.e. the number of adjacent pixels taken into account for measuring the actual ink amount.
4. In the list of plates @spot or @process may be used in order to render all spot color plates or all process color

plates. If more than one plate is to be specified, they must be entered in the list one below the other.

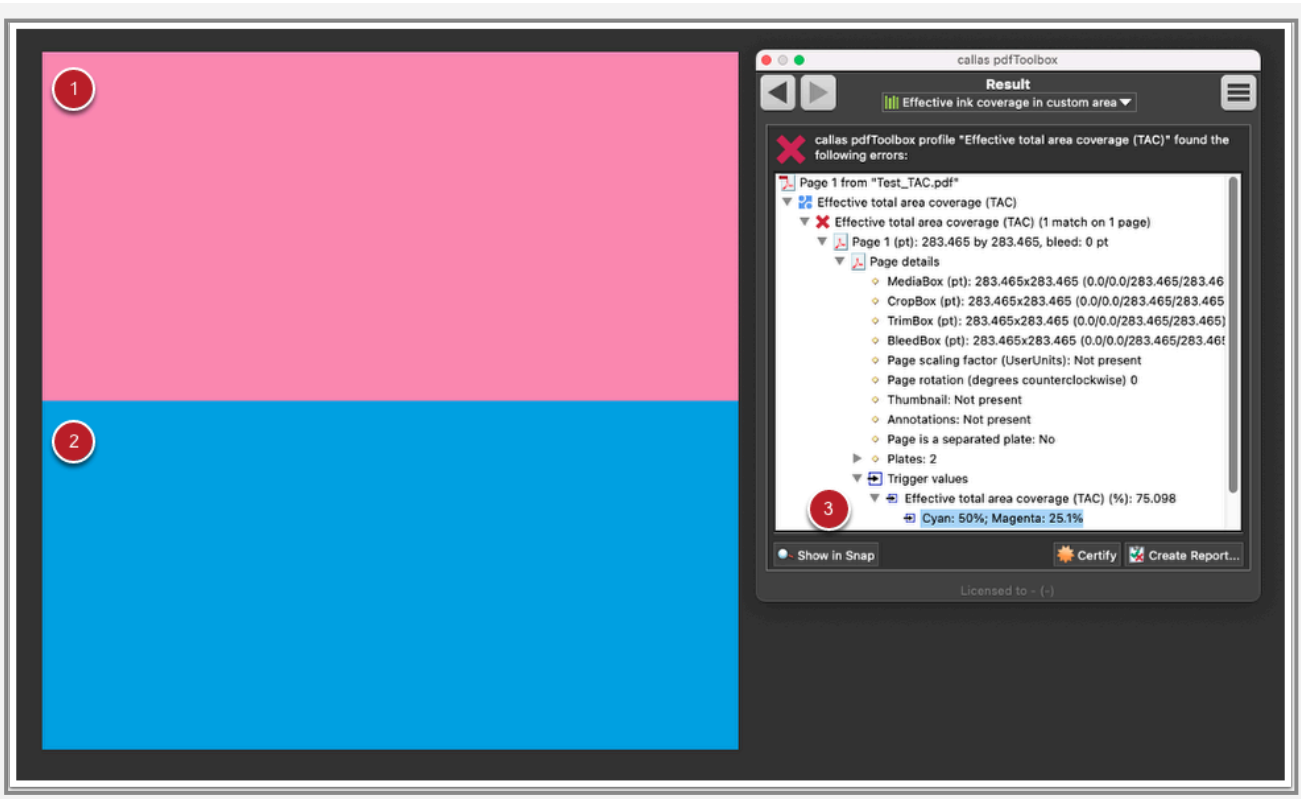
5. Enables auto adjustment of the sample size for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.
6. The trigger values depend on whether the checkbox is checked or unchecked. If the checkbox is checked, in addition to the total area coverage value (sum of all plates), all individual TAC values (for each plate) are also determined and reported in the trigger values. If the checkbox is not checked, only the total area coverage value (sum of all plates) is reported.

- With enabled checkbox, three trigger values are provided: 75.098% (TAC), 50% (TAC Cyan), 25.1% (TAC Magenta)

Evaluation of the Check result

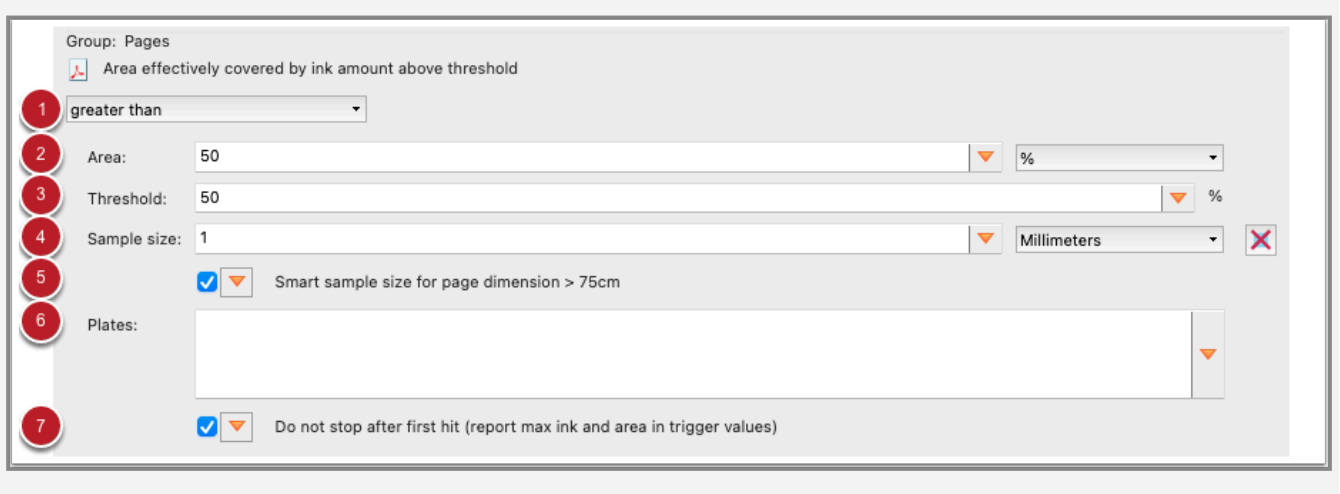
If more than 50 % of the total page of the PDF file is covered with ink, a hit will be created:

1. 50 % of the total page is covered with 50 % Magenta.
2. 50 % of the total page is covered with 100 % Cyan.
3. Total area coverage (TAC) for Magenta: ~25 %, Total area coverage (TAC) for Cyan: 50 %, Total area coverage (TAC for all plates): ~75 %



"Area effectively covered by ink amount above threshold"

Determines the total area with a specified ink amount. The threshold is the minimum amount of ink per plate to be considered. It can be used to detect pages that have only small areas of ink. The ink amount of a single plate (e.g. if 50% of Cyan or 100% of Cyan is part of the page) is not taken into account. The focus is on the area covered by ink, therefore the trigger value will never exceed 100 %.



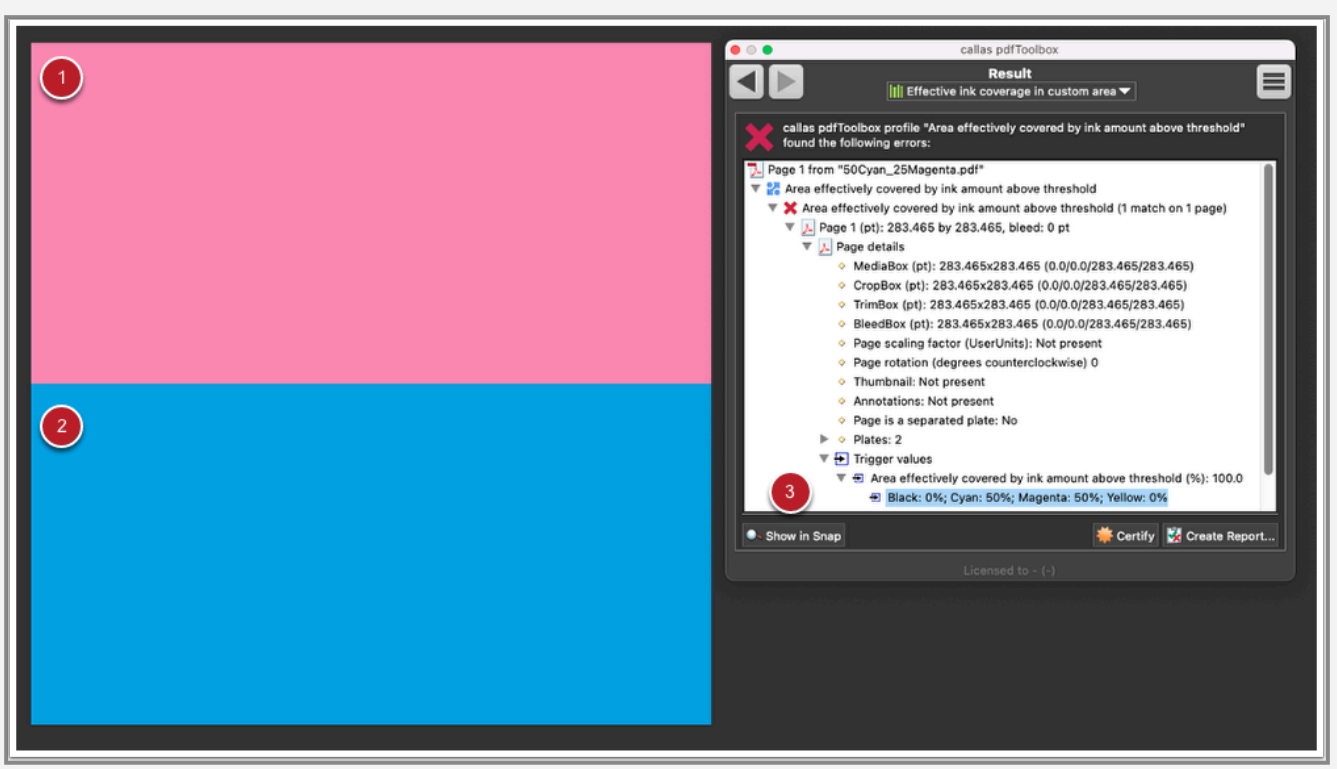
1. Operator for the threshold.
2. Size of the area that should be taken into account.
3. Threshold is the minimum ink amount per plate to be taken into account.
4. The size of the sample defines the size of the relevant area i.e. the number of adjacent pixels taken into account for measuring the actual ink amount. In order to accurately determine the area, rather small sample sizes have to be used. e.g. 1 square millimetre (will increase processing time).
5. Enables auto adjustment of the sample size for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.
6. In the list of plates @spot or @process may be used in order to render all spot color plates or all process color plates. If more than one plate is to be specified, they must be entered in the list one below the other.
7. The trigger values depend on whether the checkbox is checked or unchecked. If the checkbox is checked, in addition to the area that is effectively covered by ink, the areas for each specified plate (in this example for C, M, Y, K) are also determined and reported in the trigger values.

- With enabled checkbox, five trigger values are provided: 100% (total), 0% (Black), 0% (Yellow), 50% (Magenta), 50% (Cyan)

Evaluation of the Check result

If more than 50 % of the total area of the PDF file is covered with at least 50 % ink, a hit will be created. Let's have a look to the trigger values:

1. 50 % of the total area is covered with 50 % Magenta.
2. 50 % of the total area is covered with 100 % Cyan.
3. Area effectively covered by Magenta: 50 %, area effectively covered by Cyan: 50 %, area effectively covered by ink amount: 100 %



14. Large format

14.1 Adding grommets using a Fixup

Besides using the Switchboard action, it is also possible to use a Fixup instead.
To reflect the 2 possible ways of positioning grommets ("by distance" or "by number"), several Variables have to be defined to ensure a proper placement.

Used variables

Key of the Variable	Value	Description
grommet_template	Path to a grommet file (as PDF, JPEG, PNG)	Defines the path to the grommet to be used. Must be an absolute path.
measurement_unit	cm, mm, in, pt Default: mm	Defines the unit used for all defined distances of other Variables.
distance_bottom	Number Default: 20	Distance from TrimBox - bottom
distance_left	Number Default: 20	Distance from TrimBox - left
distance_right	Number Default: 20	Distance from TrimBox - right
distance_top	Number Default: 20	Distance from TrimBox - top
count_horizontal	Number Default: 10	Grommet count for "Grommets by number" (horizontal) Must be set to number of requested grommets or to "-1" if "spacing_..."-Variables are used
count_vertical	Number Default: 10	Grommet count for "Grommets by number" (vertical) Must be set to number of requested grommets or to "-1" if "spacing_..."-Variables are

Key of the Variable	Value	Description
		used
spacing_horizontal	Number Default: -1 (inactive)	Spacing between Grommets for "Grommets by Distance" (horizontal) Must be set to number of requested grommets (or to "-1" if "count_..."-Variables are used)
spacing_vertical	Number Default: -1 (inactive)	Spacing between Grommets for "Grommets by Distance" (vertical) Must be set to number of requested grommets (or to "-1" if "count_..."-Variables are used)

As already mentioned in the table above, the usage of the Variables "count_..." or "spacing_..." will decide if the grommets are placed "by number" or "by distance".
As the default values for the "spacing_..." are set to "-1", they are not active unless Variables are explicitly defined. If they are used, the "count_..."-Variables must be "switched off" by setting their value to "-1".

It is of course possible to combine e.g. the "count_vertical" with the "spacing_horizontal" Variable, but please remember to set the corresponding variable to "-1" if a "spacing_"-Variable is used.

Sample CLI calls

To place e.g. 15 grommets horizontally and 12 vertically, the respective CLI call would look like:

```
pdfToolbox <kfpx> <PDF> --setvariable=grommet_template:<Path to grommet PDF> --setvariable=count_horizontal:15 --setvariable=count_vertical:12
```

For placing grommets by distance (approx. 480mm on the vertical and horizontal edge) , the "spacing_..."-Variables have to be set and (as described above) the "count_..."-Variables must be set to "-1":

```
pdfToolbox <kfpx> <PDF> --setvariable=grommet_template:<Path to grommet PDF> --set-
variable=count_horizontal:-1 --setvariable=count_vertical:-1 --setvariable=spac-
ing_horizontal:480 --setvariable=spacing_vertical:480
```

Please note: As the distance between the grommets is adjusted to the total length of the edge, the effective distance used in the resulting PDF may differ from the defined input values.

Using the Fixup in the Desktop version

The Fixup can be used in the Desktop version as well of course, but using the Switchboard action is much more convenient here.

Sample files

You'll find the Fixup, which is used internally for the "Grommets" Switchboard action as well as a typical LFP PDF and a Grommet PDF below.

LFP_-_Add_grommets.kfpx

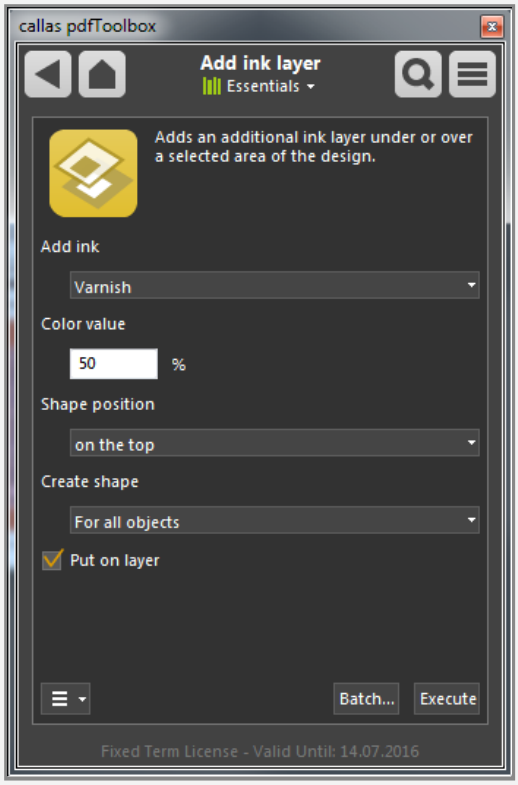
Penguin_Large_Format.pdf

Grommet.pdf

14.2 Add ink layer

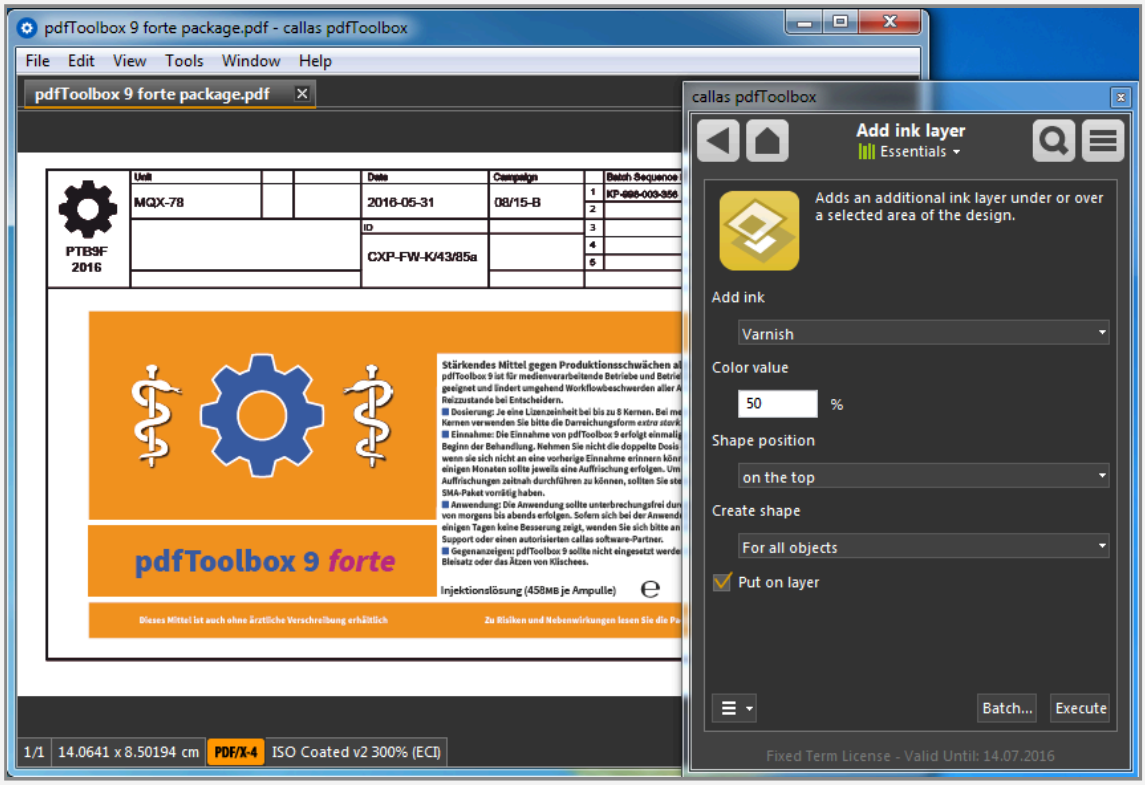
This functionality adds an additional, colored object to the PDF. For determining where existing objects are painting, the page will be internally rendered. A new shape will be created based on this result as a vector object.

Available settings



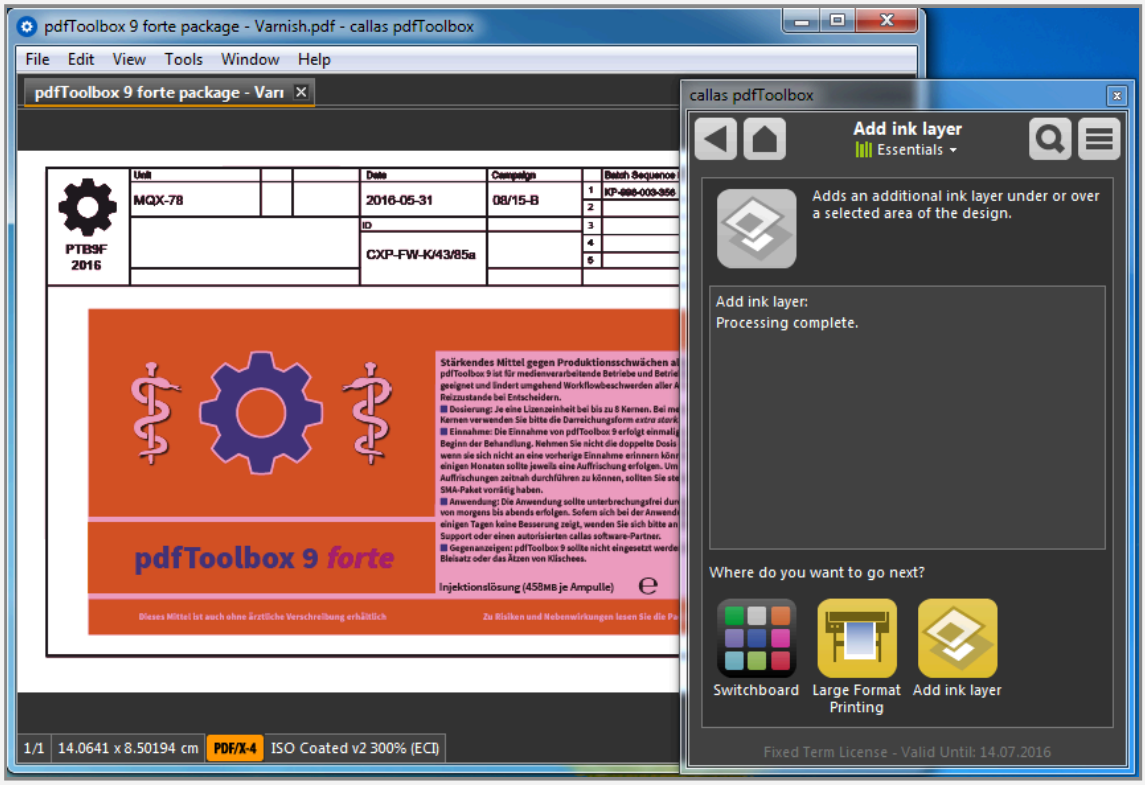
The name of the created spot color can be chosen and the color tint value be defined.
Also the the position of the shape can be set. Optionally, the new object can be created on a layer.

Create "Varnish" object for all painting content



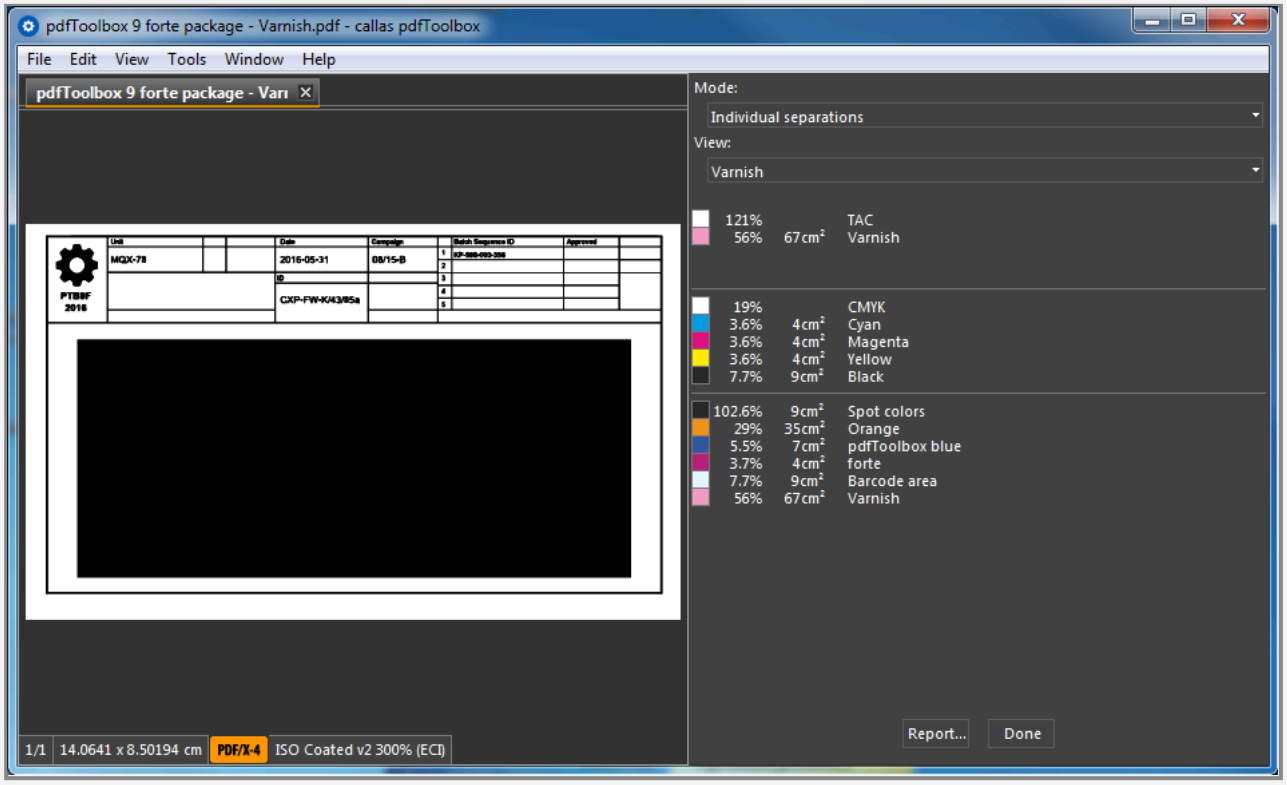
For example, to create a new object, which covers all painting content, just choose "For all objects" and define a color value.

The result



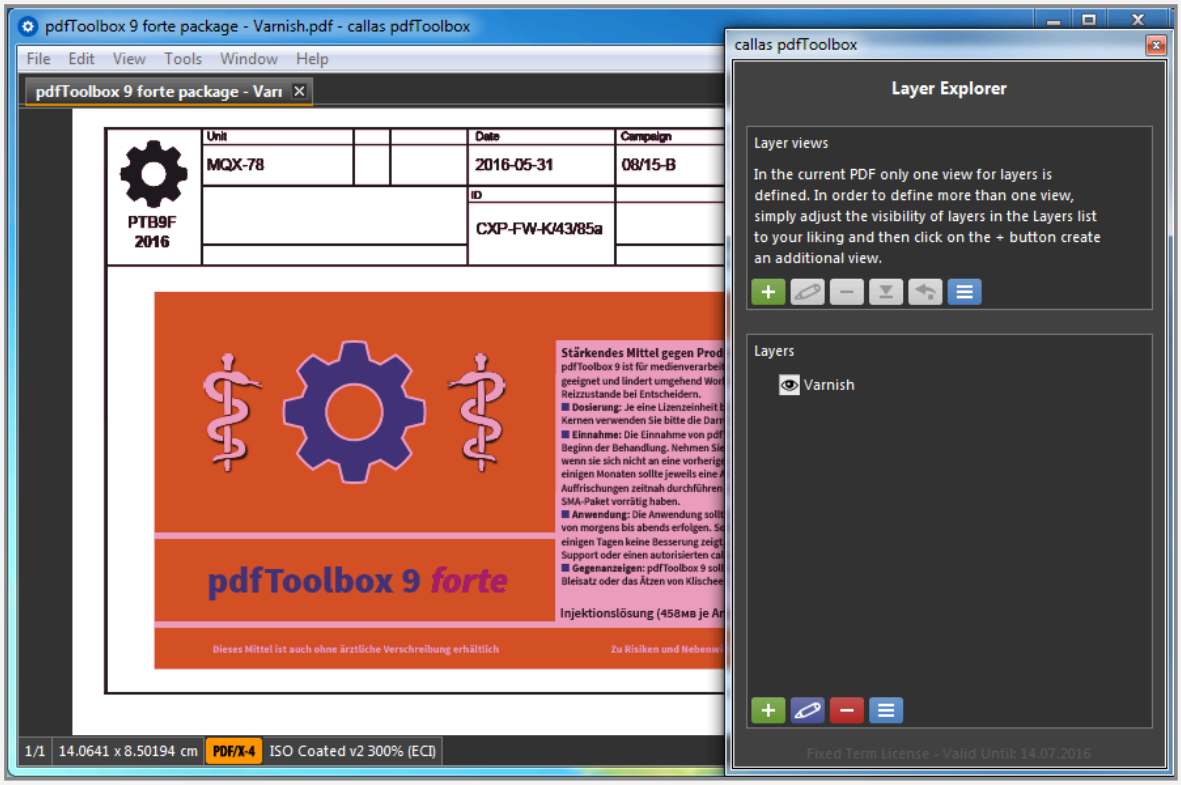
After processing a new vector object, using the spot color "Varnish" will be added. Covering all content, also objects using white.

Inspecting the result - individual separation for spot color



When inspecting the result using "Visualize individual separations", the new created object using "Varnish" can easily be reviewed.

Creation on a layer

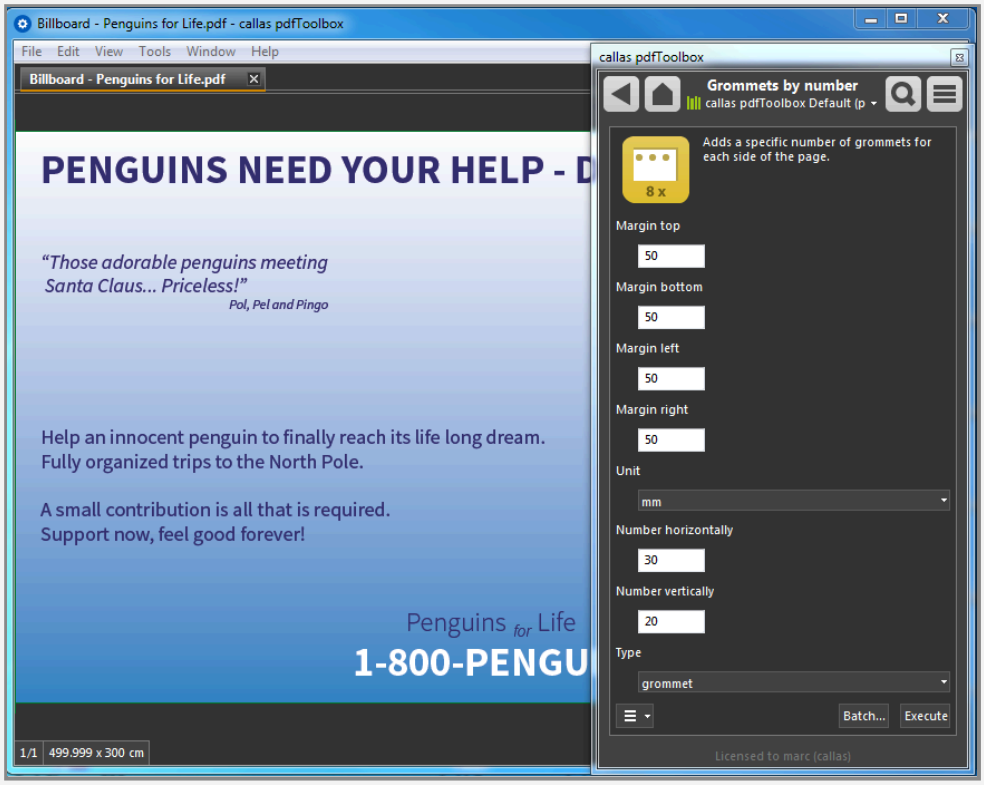


If required, the newly added object will be placed automatically on a layer, which will have the same like the spot color.

14.3 Adding grommets

**During the production of banner or other large format products, sometimes grommets must be added.
To add marks, where these grommets shall be placed after the product is printed, this Switchboard action can be used.**

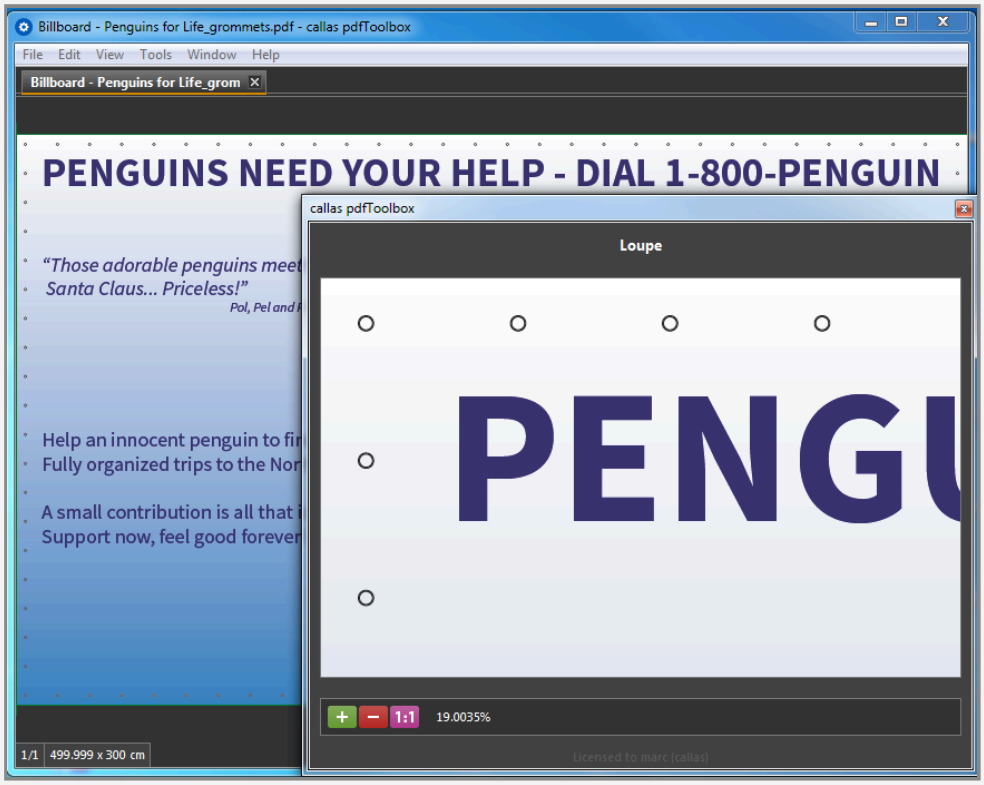
Define the settings



To define the positioning of the grommets, the margin for all 4 edges can be defined.

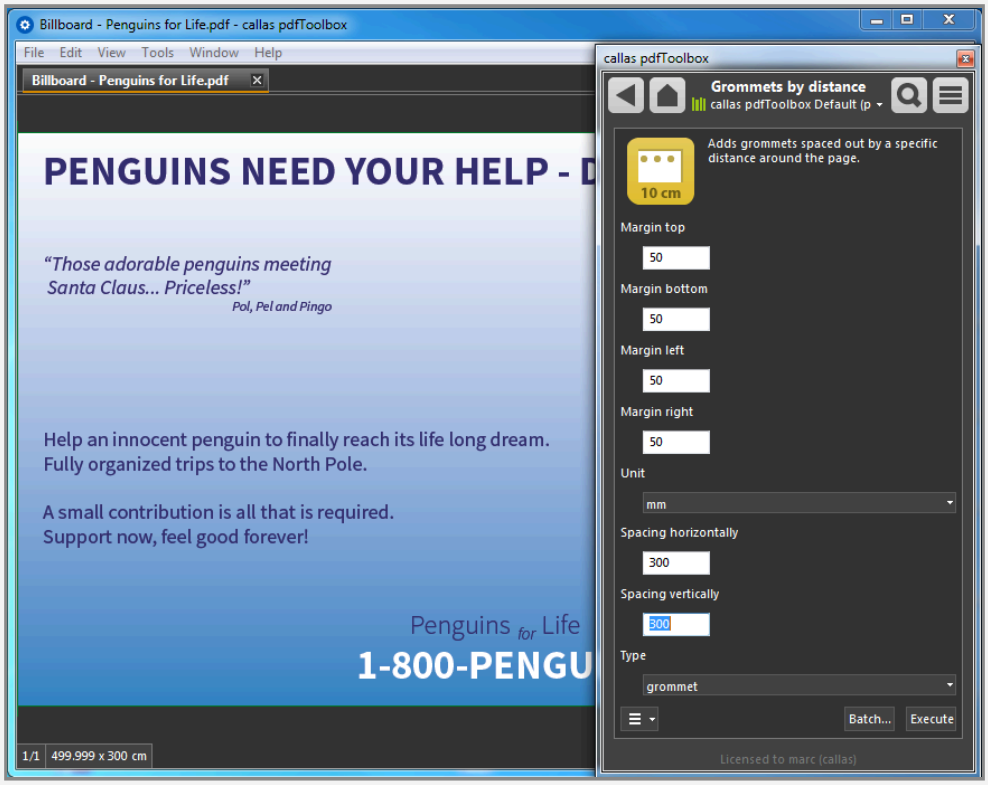
Of course the number of grommets for the horizontal and vertical edges must be defined. The internal calculation will determine the distance between the grommets.

Inspecting the result



Marks for the grommets will be positioned accordingly to the defined settings.

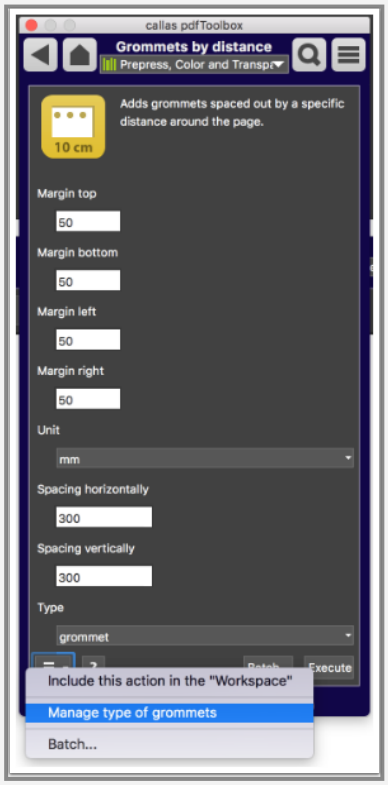
Grommets by distance



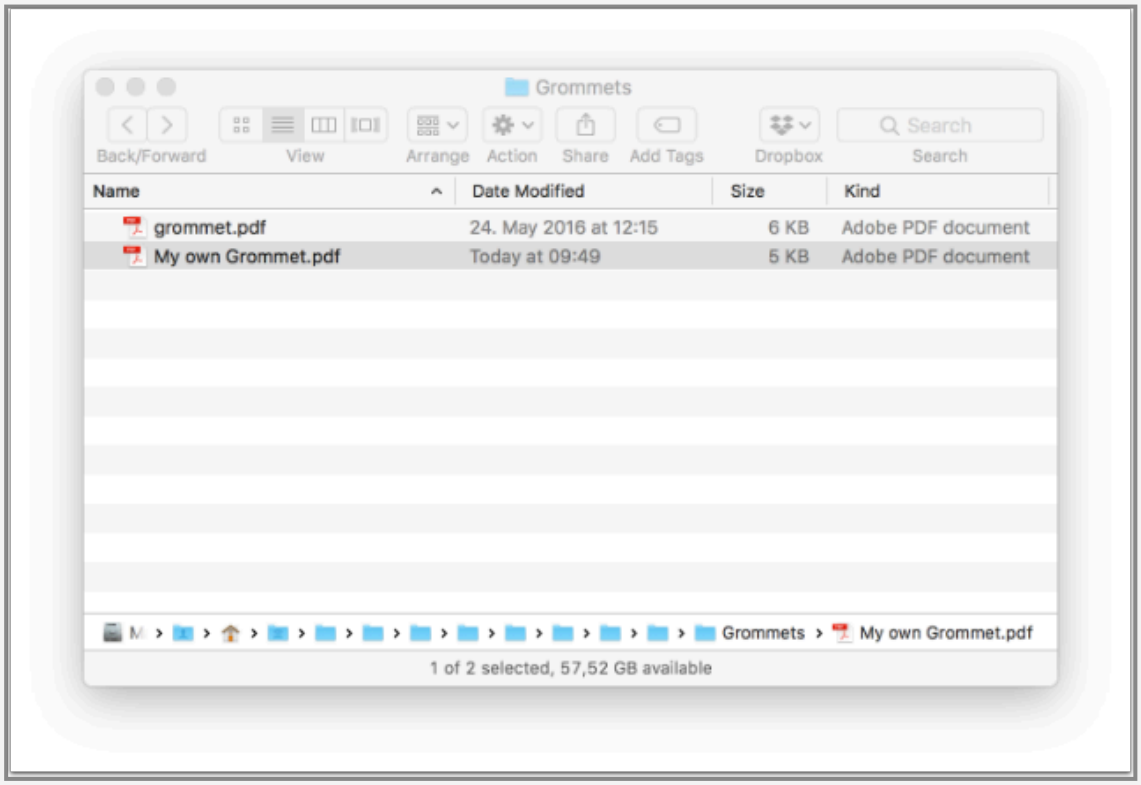
An additional way to add grommets to the document is by defining the distance between the grommets vertically and horizontally.

Adding own types of grommets

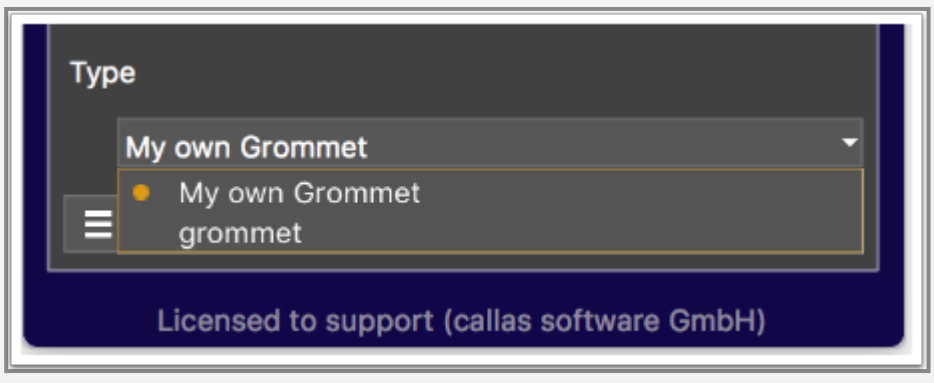
It is possible to add own types of grommets.
Just click on the menu button in the lower left and select "Manage type of grommets"



An Explorer/Finder window will open.
Just copy your own grommet as an PDF into this folder.
Please mention, that the PDF must have the exact size of the grommet, as it will be positioned 1:1 onto the PDF.
Each Library has its own set of grommets. If you want to have your grommets in all your Libraries - just repeat the steps and copy your own grommets into all your Libraries.



When you have copied the own grommets PDF into the folder, go back to the Switchboard, click the "back" arrow in the upper left corner and select one of the grommet Actions again.
The new grommet will show up in the "Type" pop up menu.
Added grommets will be available in both "grommet" Switchboard Actions.



How to place grommets using the CLI

This Switchboard action is based on a Fixup using several Variables to define the placement.

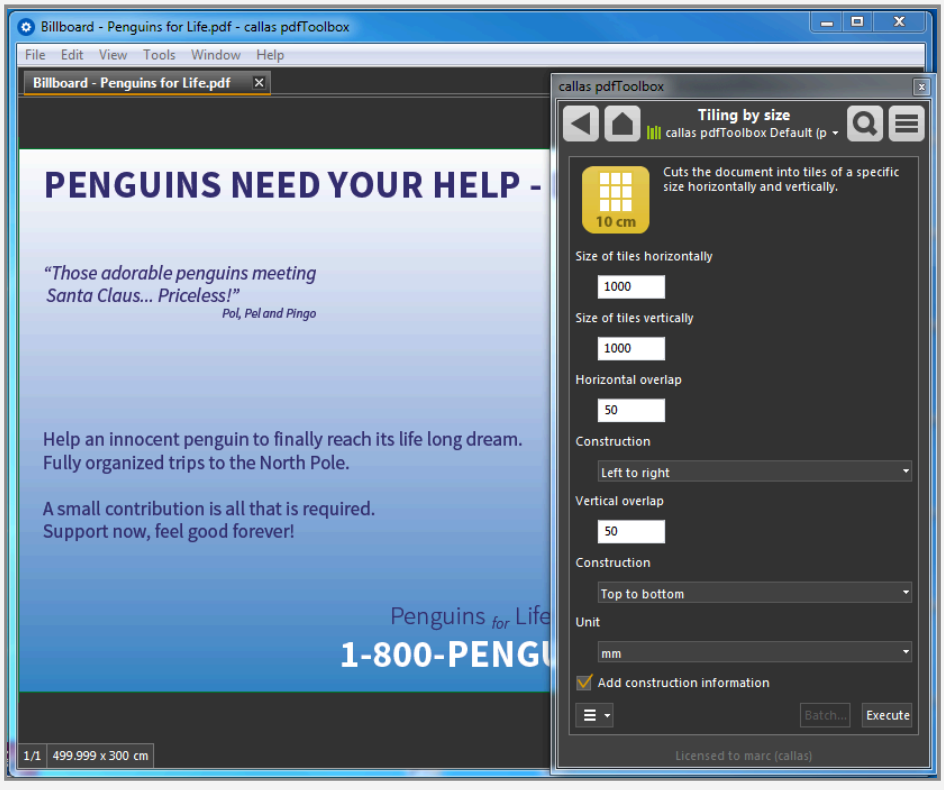
Pleasse see this article how to use this functionality e.g. using the LCI

14.4 Tiling

Tiling will cut the document in a number of parts to prepare the document for various large format printing methods.
The document can either be cut by a defined size for the resulting tiles or by the number of tiles horizontally and vertically.

This function is available starting with pdfToolbox 9.0.

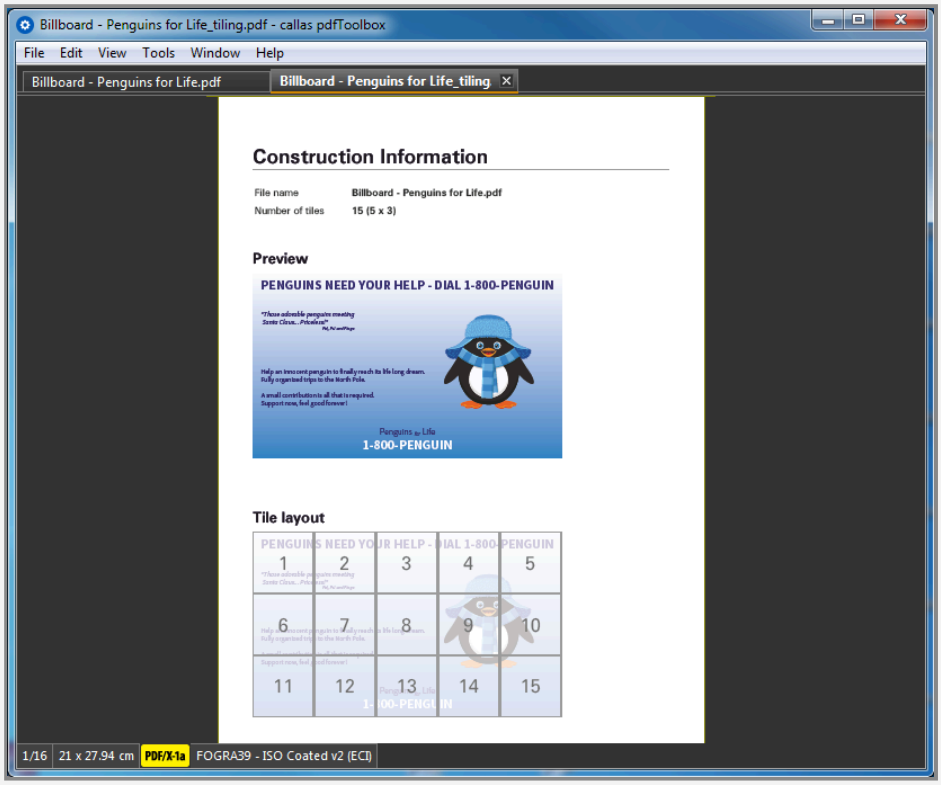
Define the settings



Using the Switchboard action, it is possible to define the size of the resulting tiles and the overlap. Additionally the construction direction vertically and horizontally can be defined.

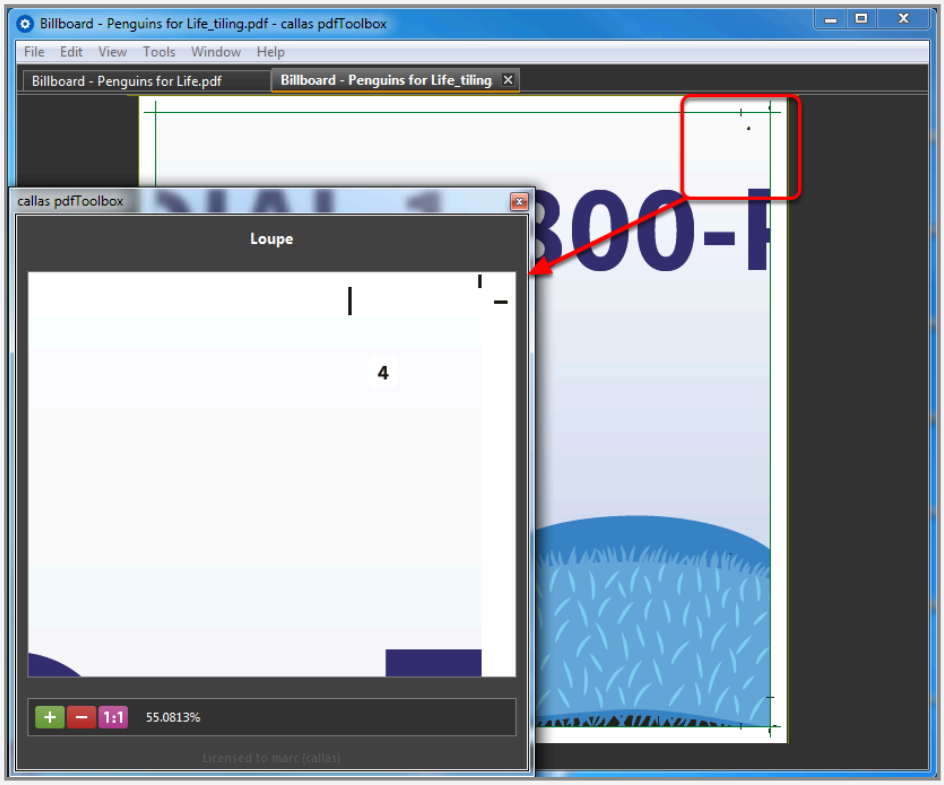
An additional page with construction information can be added as well.

Construction information



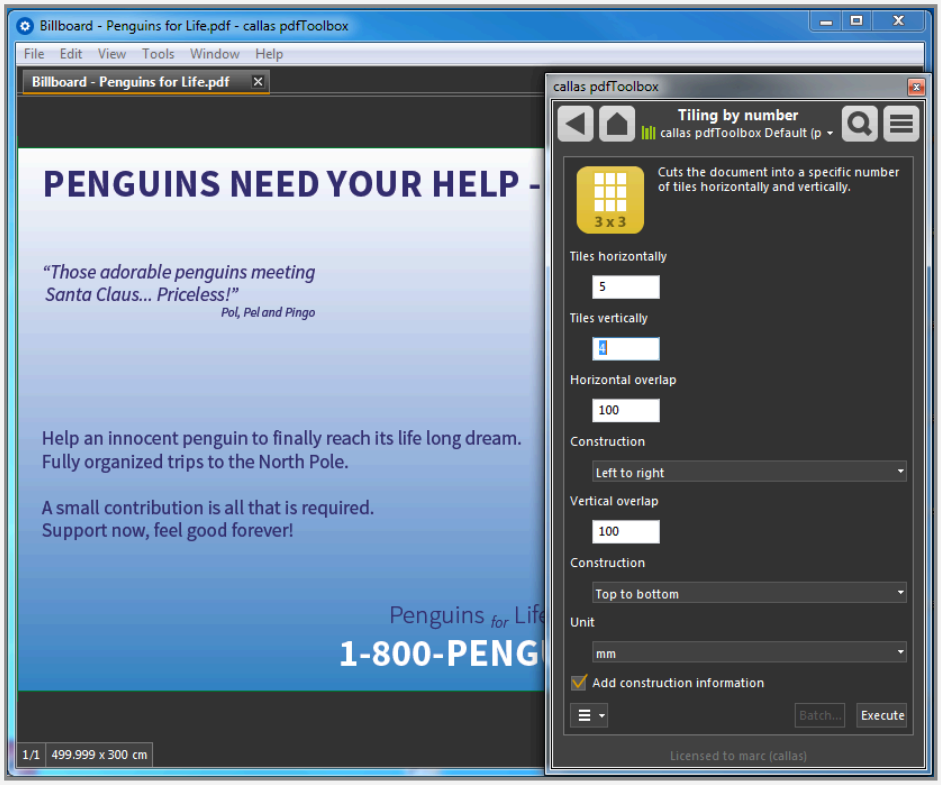
The construction information contains some basic information about the file and the number of tiles, as well as a sketch for the tile layout.

Information for overlap



If an overlap has been defined, a small mark to indicate there the next, overlapping tile has to be positioned will be added. Also the number of the actual tile will be printed into the overlapping area (which will become covered by the next tile).

Tiling by number



An additional way to tile the document is by defining the number of tiles vertically and horizontally.
The overlap can be defined in the same way as described above.

Using this action on the command line (CLI)

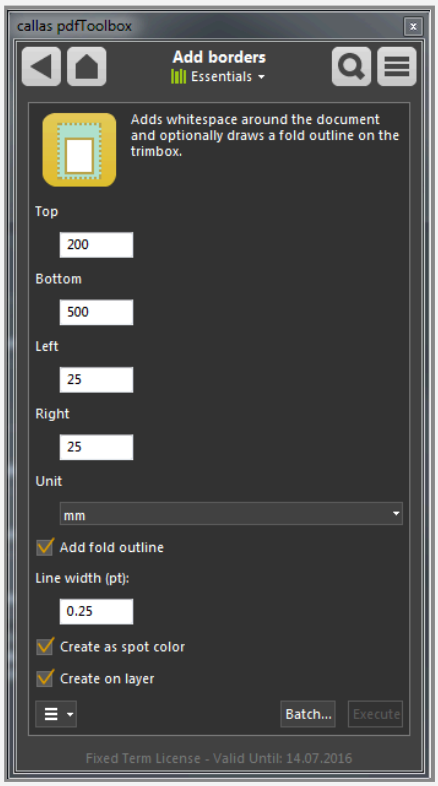
A separate CLI action is available for use on the command line.

The detailed description can be found in the CLI manual.

14.5 Add borders

Adds whitespace around the document by enlarging the existing visible area (defined by the CropBox). This can be useful for producing large format products like Roll Up Banners

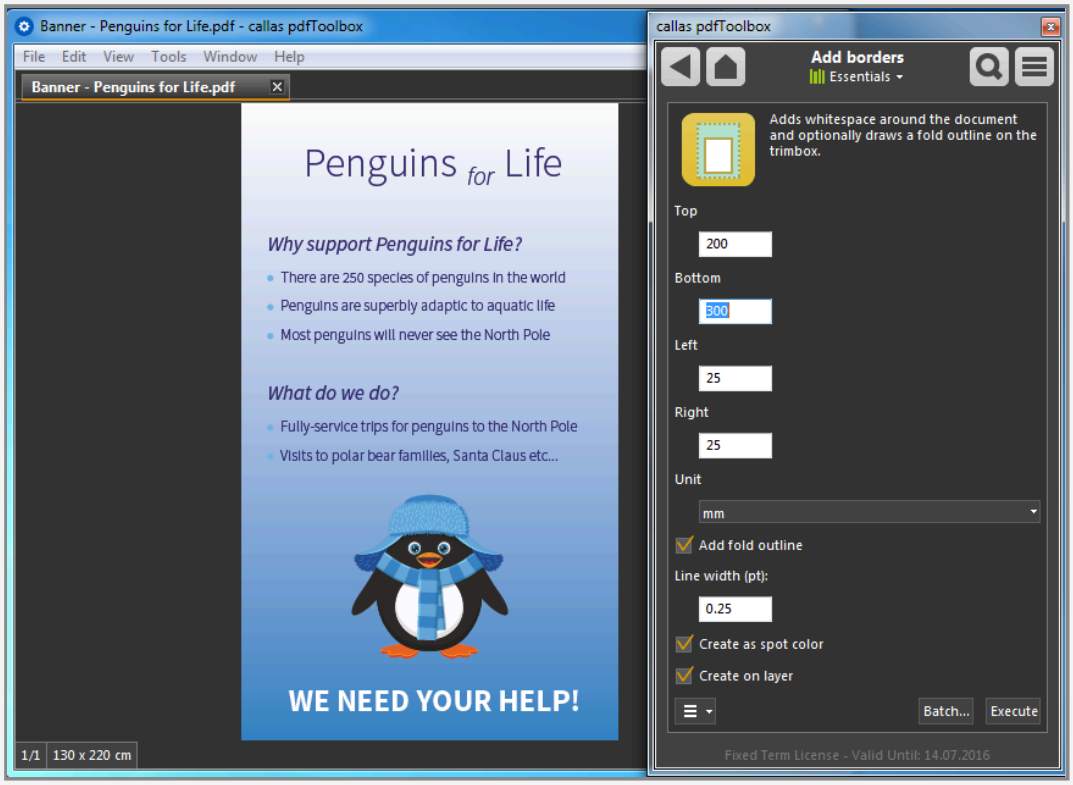
Available settings



The margin to be added can be defined for all 4 edges of the document. The used unit for these values can be chosen as well.

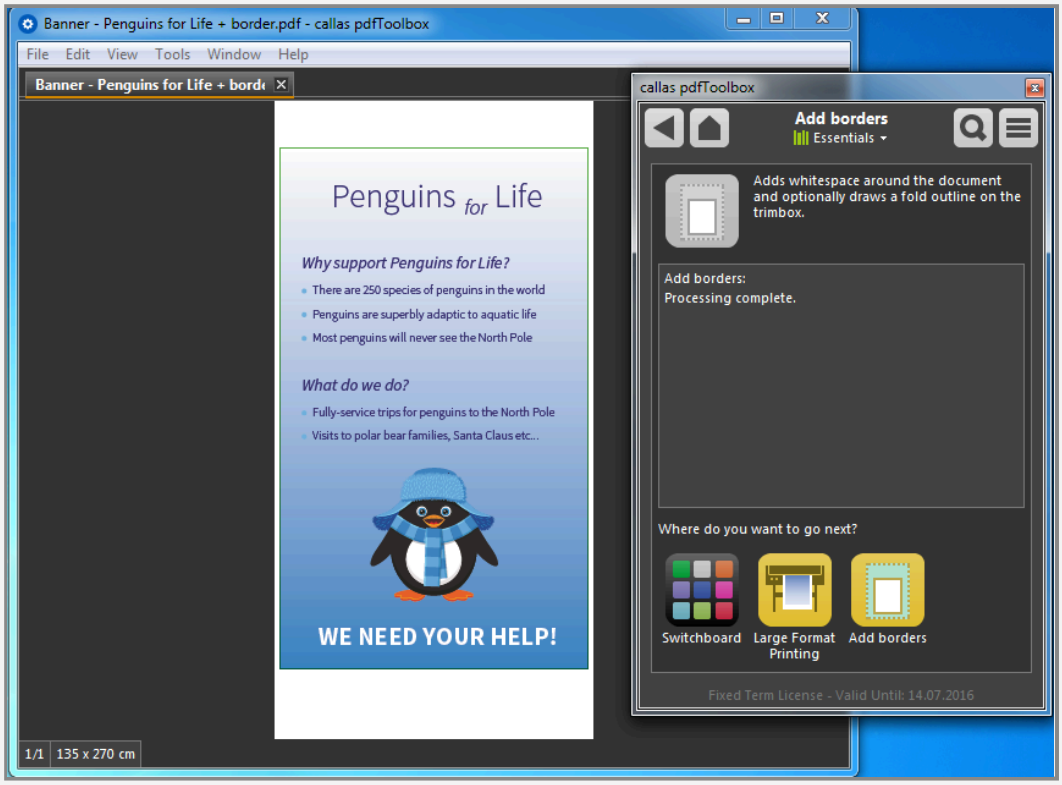
By activating "Add fold outline", the former size of the page will be marked by an outline, which allows cutting or positioning during production afterwards.
The line width, creation as a spot color or separating this outline on a layer can be optionally activated as well.

Extend for Roll Up Banner



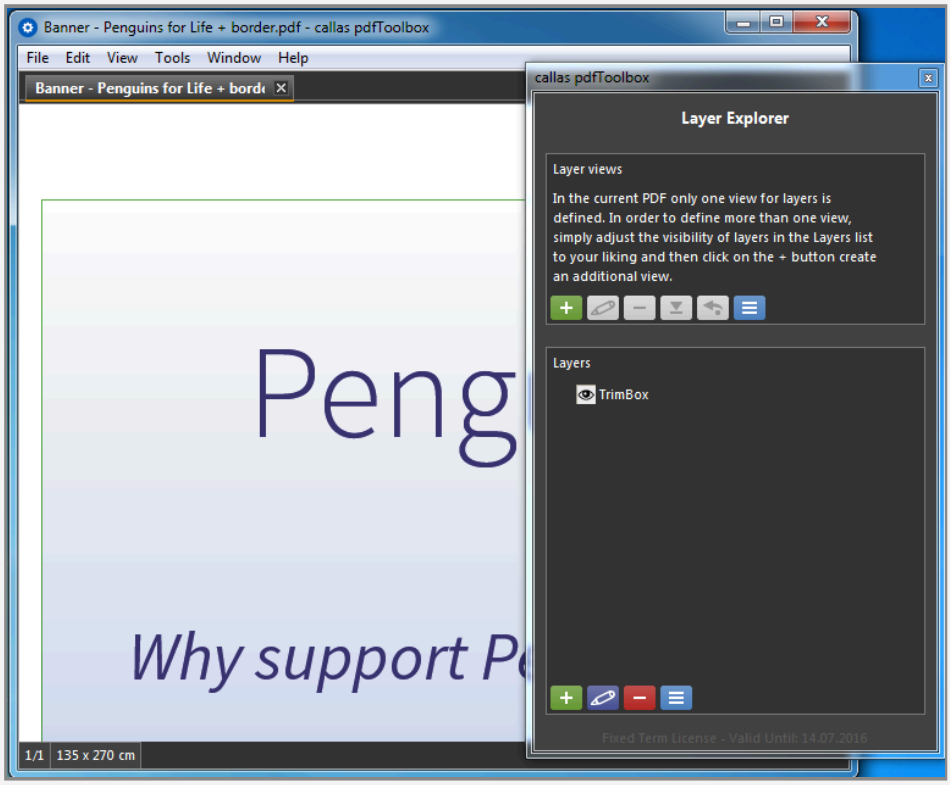
For adding space to prepare the PDF (which has the right dimension) for a Roll Up Banner, just enter the required values. Press "Execute" afterwards.

Extended result PDF



The file gets enlarged by the defined values.

Fold outline on a layer



As the "Add fold outline" was activated, the former page size became outlined. Causes by "Create on layer" this outline is placed on a layer, so it can be switch on or off easily.

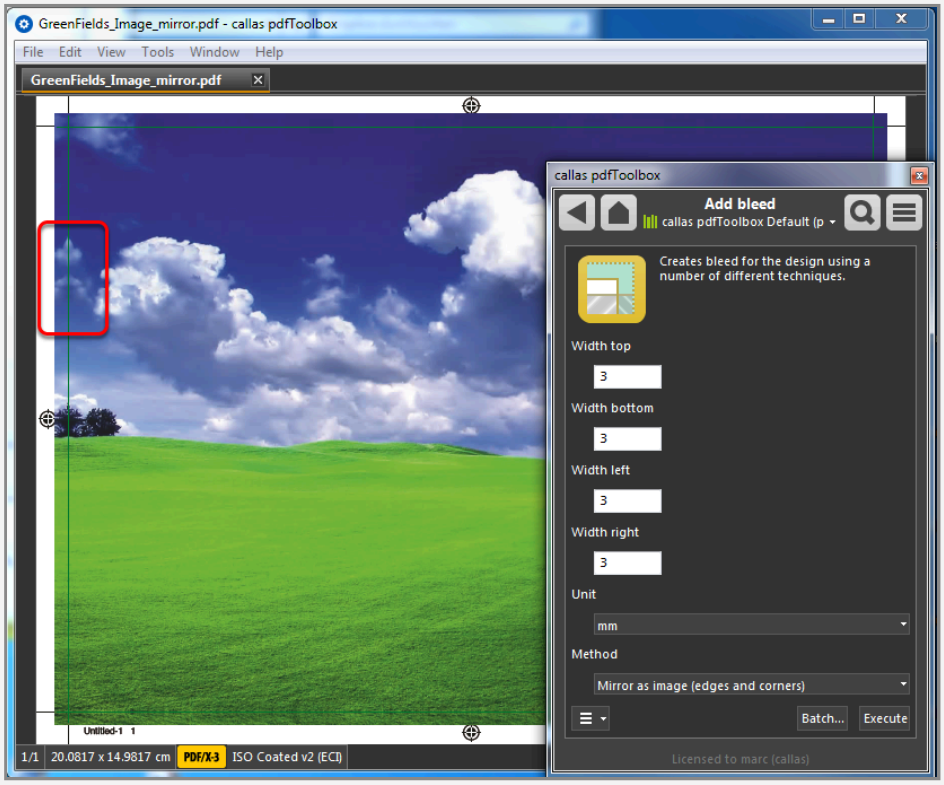
14.6 Add bleed

If a PDF has been created without bleed, or if the bleed is not sufficient, generating bleed out of page content is needed to avoid register problems.
To solve this task, pdfToolbox offers a variety of methods to add bleed from page content:

- Mirror content as image
- Repeat last pixel as image
- Mirror page objects

It is possible to define the width of the bleed per edge. The different methods can even be combined when using this feature as a Fixup.

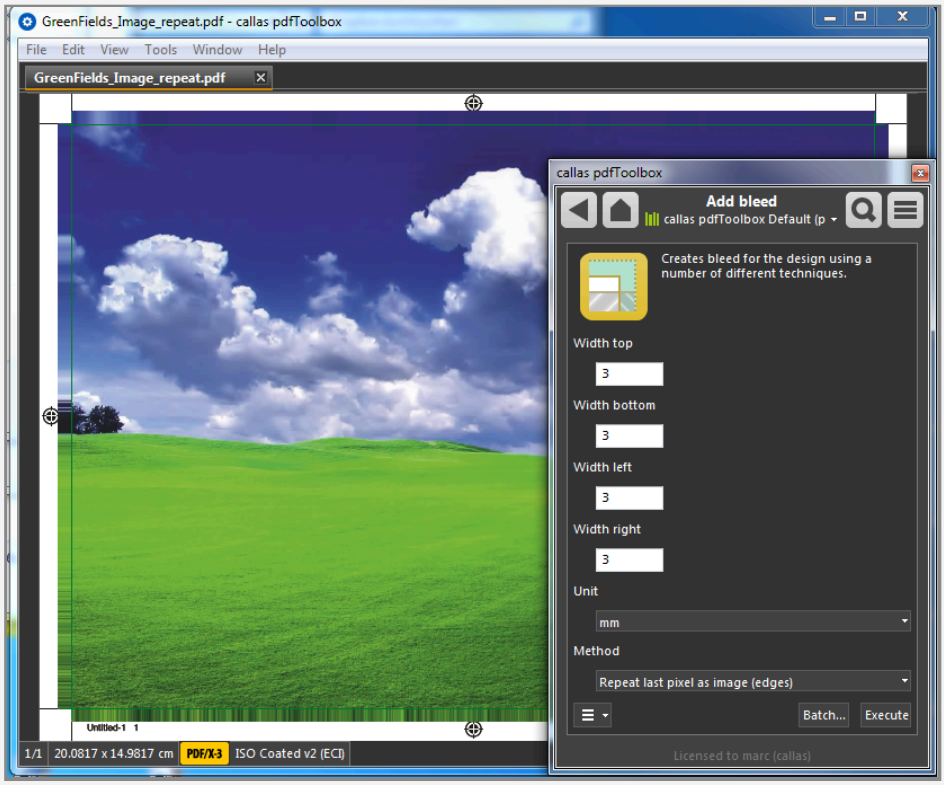
Mirror content as image



Using the "Mirror as image" option will create an image for each edge (and if selected: corner) at the TrimBox from the

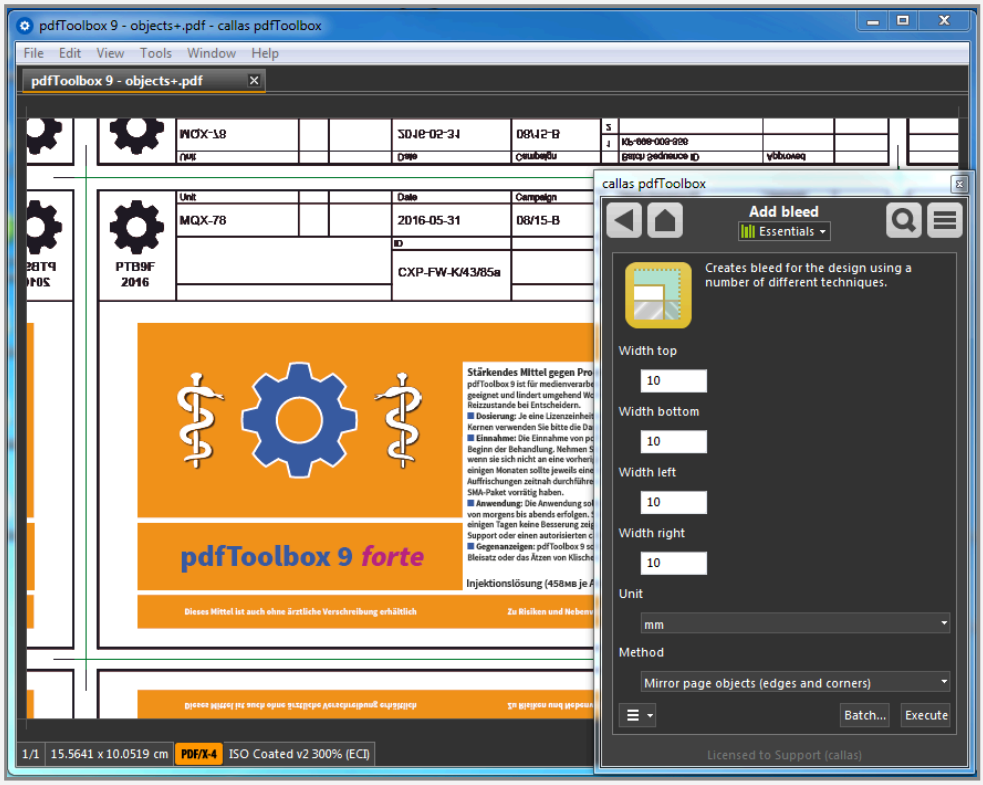
content. This image will be place mirrored outside of the TrimBox.
The red rectangle shows a good example of the effect of this method.

Repeat last pixel as an image



The "Repeat last pixel as an image" method is rendering a thin strip inside the TrimBox and creates a wider image out of this last visible strip. This will result in an image, which "smears" out the last used color inside of the TrimBox outside as the bleed.

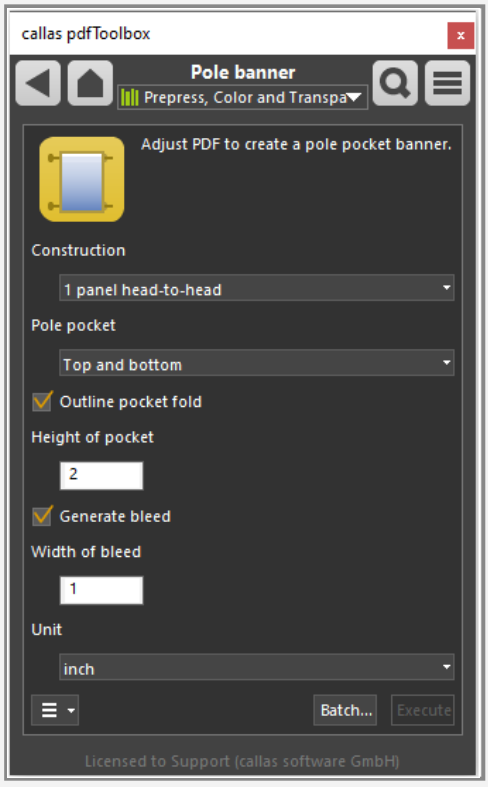
Mirror page objects



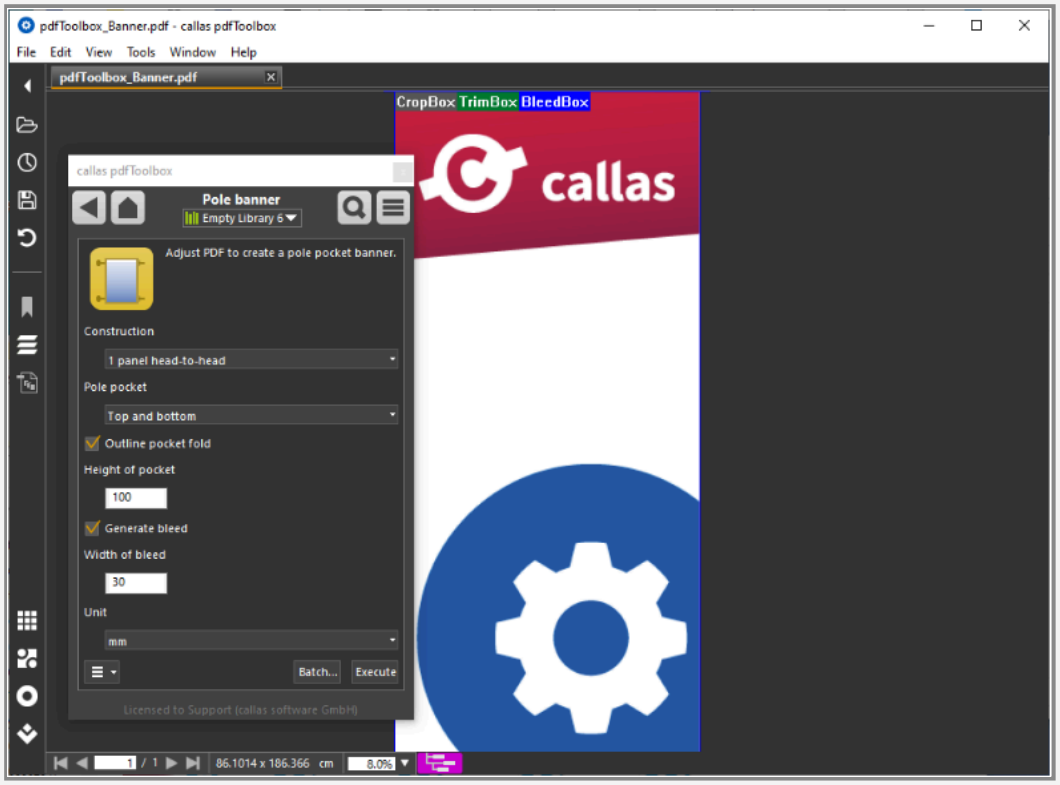
Yet another approach is to use all of the page objects, duplicate them (four times), flip them (two of them horizontally, the other two vertically), and then append them one by one to each of the four sides of the page. This will result in bleed with the same rendering accuracy and rendering quality as the page content itself. Different form what one may expect, the impact on file size is minimal (the page objects are not actually duplicated, but instead four references are created). The only real downside is that both a PDF viewer as well as a PDF output system will have to process – to a certain degree – five times as much page content data. This can lead to longer processing times during output, especially on systems with relatively small amounts of working memory.

14.7 Create pole pocket banner

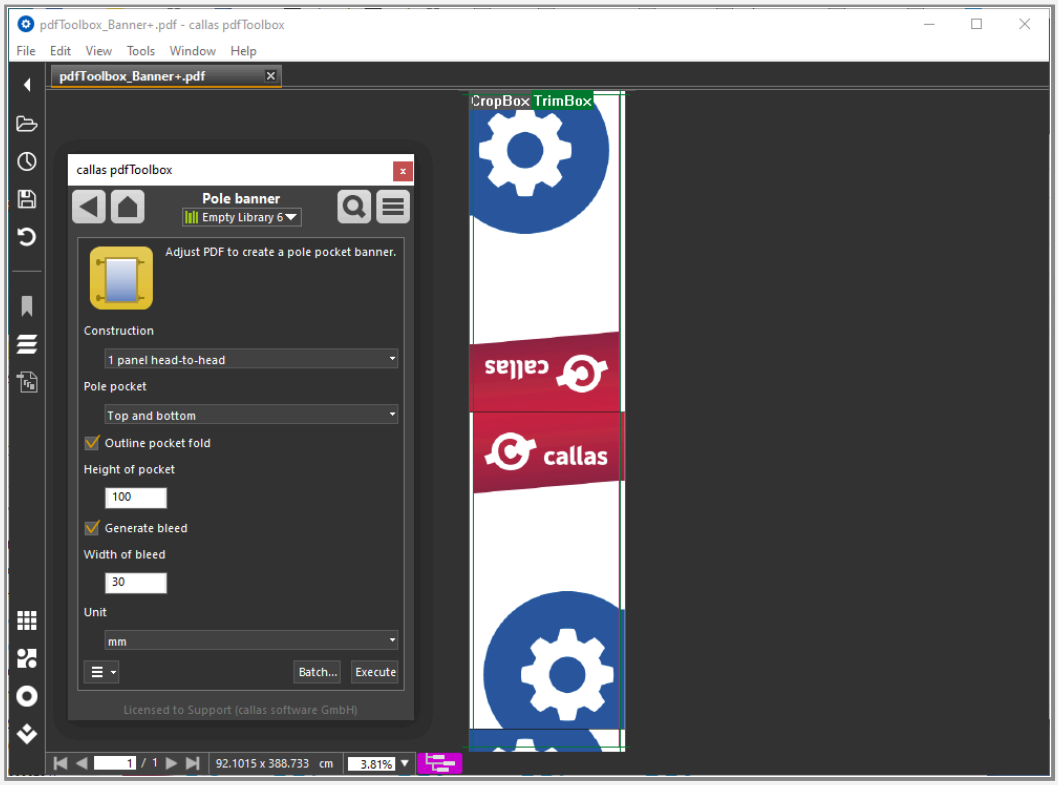
In pdfToolbox 12, the LFP group in Switchboard has a new functionality to prepare PDFs for pole pocket banners. As there are many variants to create pole pocket banners, there might be a need to adjust some parameters to match some specific requirements.
In this article, we will explain the executed steps and provide the used Process Plan, so that it is possible to modify the Process plan when needed.



Sample processing

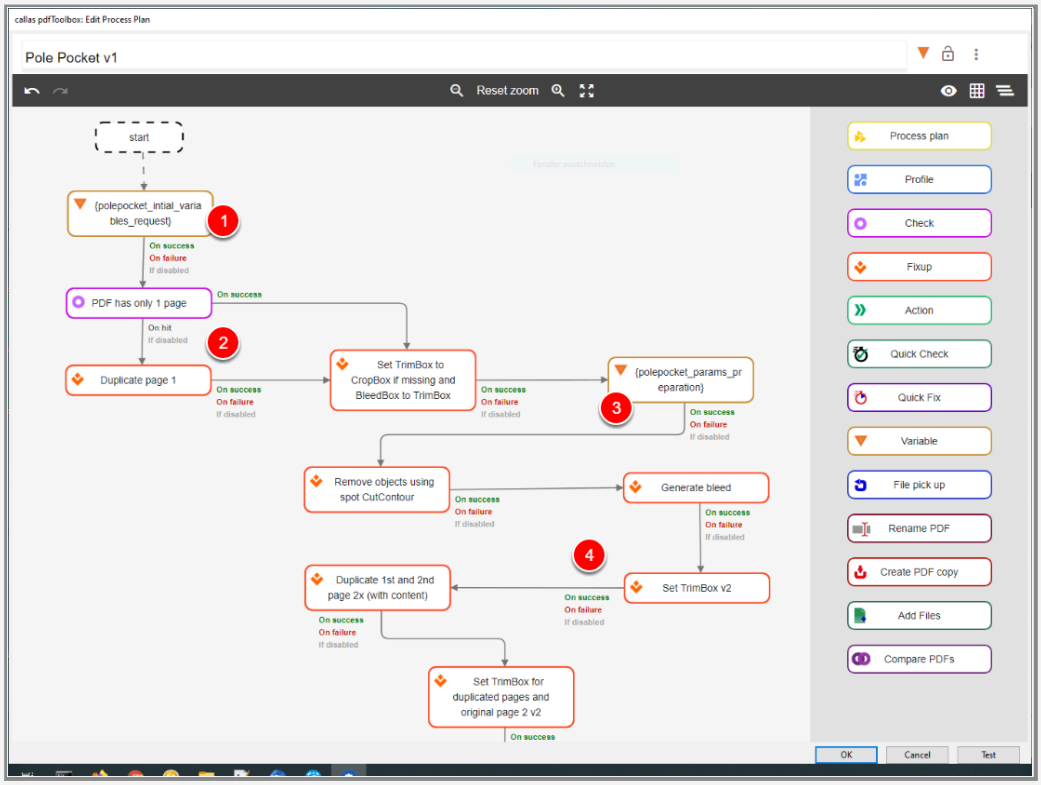


The settings shown above will result in a head-to-head banner, which has 30 mm bleed on all sides:

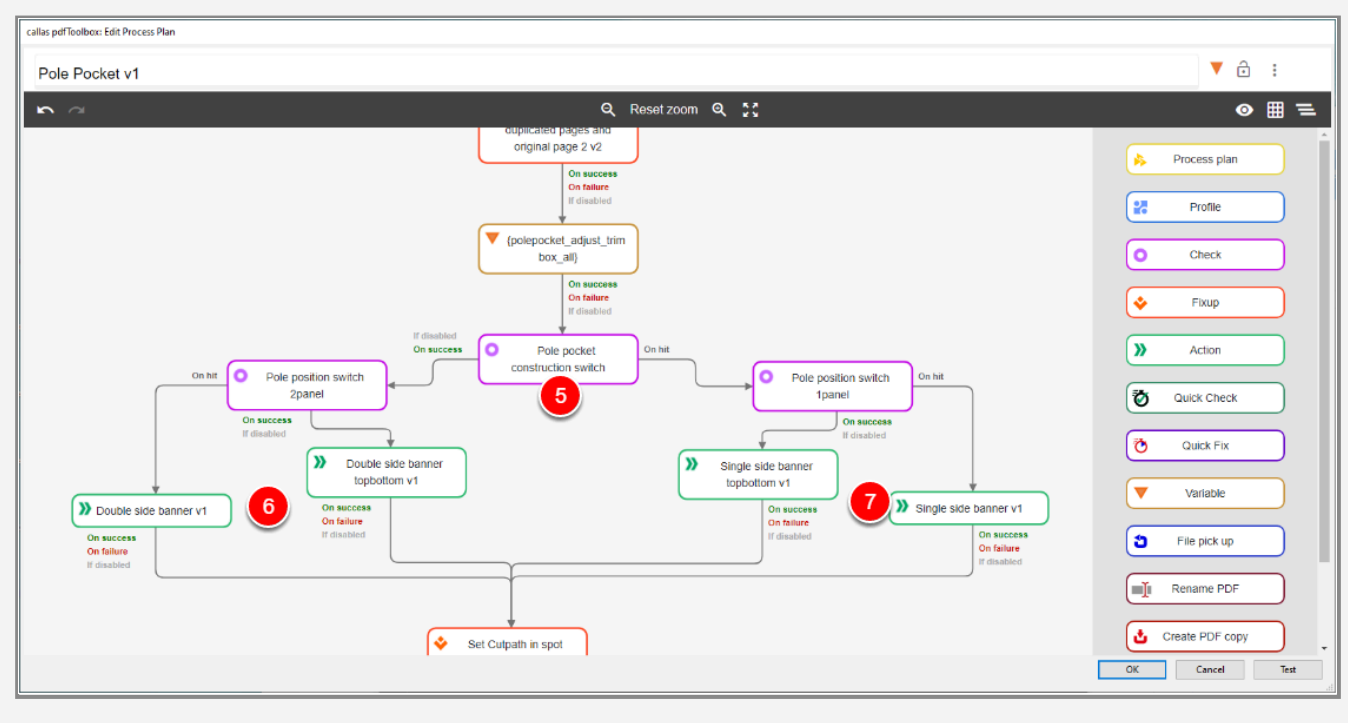


Used Process plan

As already mentioned, this Switchboard function uses a Process Plan where the settings are converted into several variables, which are influencing the way of processing.
Let's have a brief overview about the structure of the Process Plan.



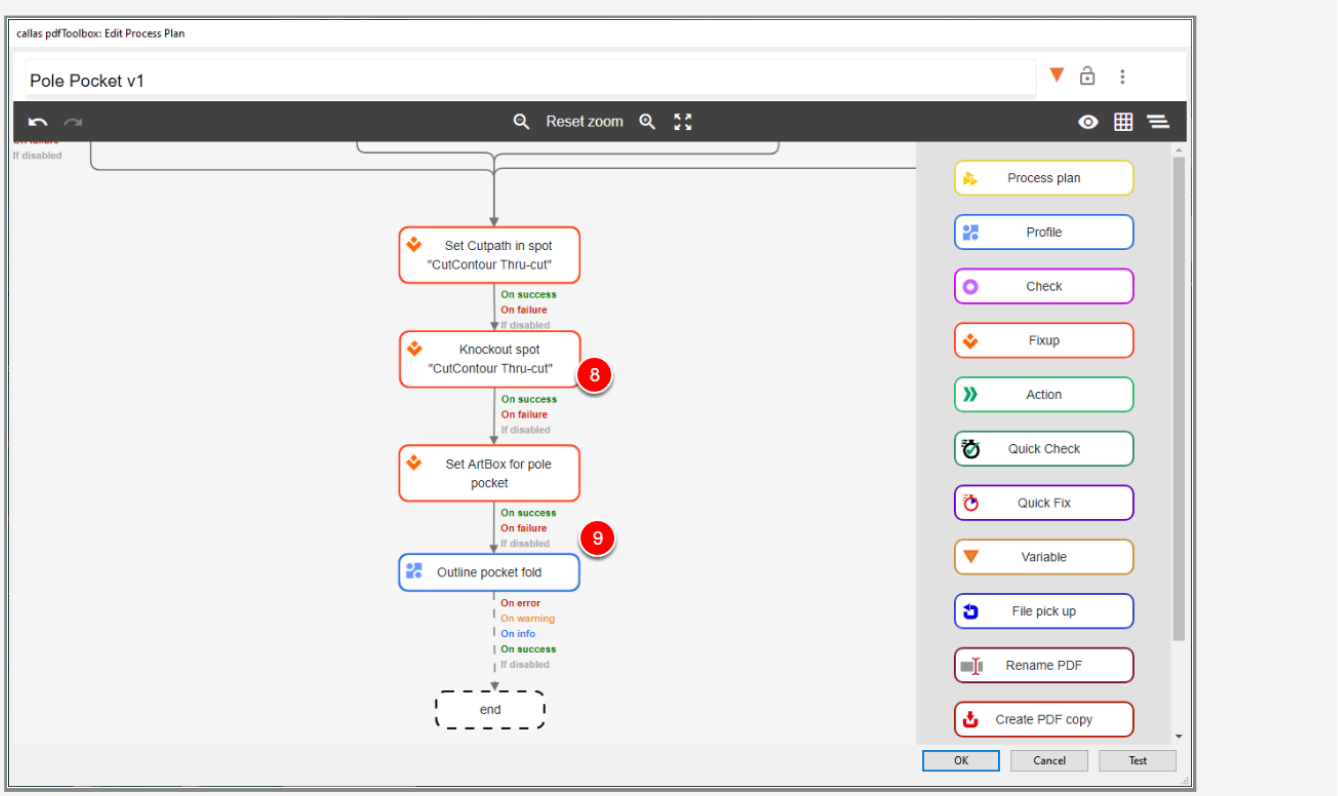
1. A Variable step is requesting all needed Variables from the user
2. If the input PDF is a single-page document, the page is duplicated
3. Another Variable step, which is collecting e.g. the page sizes of the input document.
4. As the positioning of the pages is done by the TrimBox, it is ensured that this box is set properly



5. Depending on the requested output variant, Variables are used to lead the PDF to the correct imposition scheme

6. Imposition is done for double side banner...

7. or for single side banner



8. An additional outlining in color "CutContour" is added to show the final format

9. The folding lines are outlined if requested

Settings and their internally used Variables

Possible settings	Variable used in Process plan
Construction	1 panel head-to-head: 1panel_h2h 2 panels back-to-back: 2panels_b2b *Name of internal variable: polepocket_construction*
Pole pocket	Top: top Top and bottom: topbottom *Name of internal variable: polepocket_position*
Outline pocket fold	Yes|No: true|false or 1|0 *Name of internal variable: polepocket_fold_outline*
Height of pocket	Numeric value

Possible settings	Variable used in Process plan
	Name of internal variable: polepocket_pock-etheight
Generate bleed	Yes|No: true|false or 1|0 *Name of internal variable: polepocket_cre-ate_bleed*
Width of bleed	Numeric value *Name of internal variable: polepocket_bleed-width*
Unit	mm|inch|pt *Name of internal variable: polepocket_unit*
Outline line width (not in UI)	Numeric value *Name of internal variable: polepocket_fold-linewidth*

Used Process plan and sample file

Pole_Pocket_v1.kfpx

pdfToolbox_Banner.pdf

14.8 Smart resolution to limit memory consumption

For many Fixups and Checks, pdfToolbox internally renders a PDF page or all pages in the PDF file.

There are two types:

1. Fixups where the page is rendered and the result is used directly as page content (like bleed generation, convert page into image).
2. Fixups and Checks that are also based on rendering, but only use the result for internal analysis (such as barcode detection, shape creation).

Here is a list of all Fixups and Checks that are based on internal rendering:

Fixups

- Generate bleed at page edges
- Generate bleed for irregular shapes
- Convert page content into image
- Create spot color plate based on ink amount
- Set page geometry boxes (based on page content)
- Create shape (from tracing page content)
- Crop to visible based on rendered page

Check properties

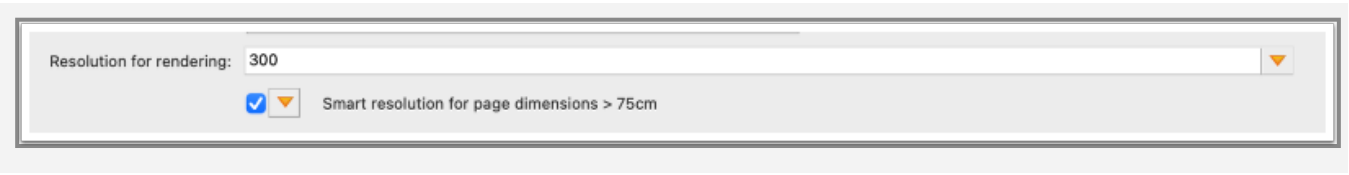
- Effective ink … properties
- Find barcodes
- Shape definition
- Result file different from original (visual comparison)

Problem

The Checks and Fixups listed above require a rendering resolution to be set. Usually reasonable rendering resolutions (e.g. 300 ppi) result in unreasonably large memory consumption when used on large format files, because a lot of pixels have to be created. This leads to long processing times and, if the results are used directly, to large result files.

Solution: Smart resolution

With pdfToolbox 15 all Fixups and Checks listed above have a new checkbox "**Smart resolution for page dimensions > 75 cm**". This checkbox provides an option to automatically adjust the resolution for large format files.



pdfToolbox's internal definition for large format files is that if a page has at least one page dimension that is larger than 75 cm, it is considered as a large format file. When the smart resolution is enabled, a resolution factor will be applied to each PDF page that has at least one page dimension that is greater than 75 cm.

Here is an example for smart resolution if 300 ppi is set as rendering resolution and the "Smart resolution for page dimensions > 75 cm" checkbox is enabled:

Page dimension of PDF page	Resolution factor	Smart resolution
At least one page dimension is greater that 75 cm	0.75	225 ppi
At least one page dimension is greater that 150 cm	0.5	150 ppi
At least one page dimension is greater that 200 cm	0.25	75 ppi
At least one page dimension is greater that 400 cm	0.1	30 ppi

Use Case

If you always process PDF documents with similar page dimensions (e.g. only large-format files or only business cards), you don't need to use the smart resolution feature because you can set a rendering resolution in a Fixup or Check that works for all files. But if you have varying page dimensions, this feature is a really smart way to limit memory consumption.

15. Variable Data Print (VDP)

15.1 Create VDP files from PDF templates

pdfToolbox provides a ‘**search and replace for VDP Quick Fix**’ which can be used for certain types of variable data printing. The Quick Fix is based on a PDF template with placeholder text and a JSON configuration file, which allows to create VDP data with individual text and graphic content (e.g. barcodes).



TIP: if the placeholder text is not really unique and could occur elsewhere on the page, it should be supplemented by one or more ‘special’ character(s). E.g. "car" might occur elsewhere in the text on the page (e.g. in the word "careful"). Therefore, "$car" or "%car%" would be a better choice for the placeholder text.

Two options to run this Quick Fix

On Command line with a JSON configuration file

Use `--quickfix` on the command line and add your JSON configuration file and the input file.

On pdfToolbox Desktop with the "JavaScript based configuration" Quick Fix

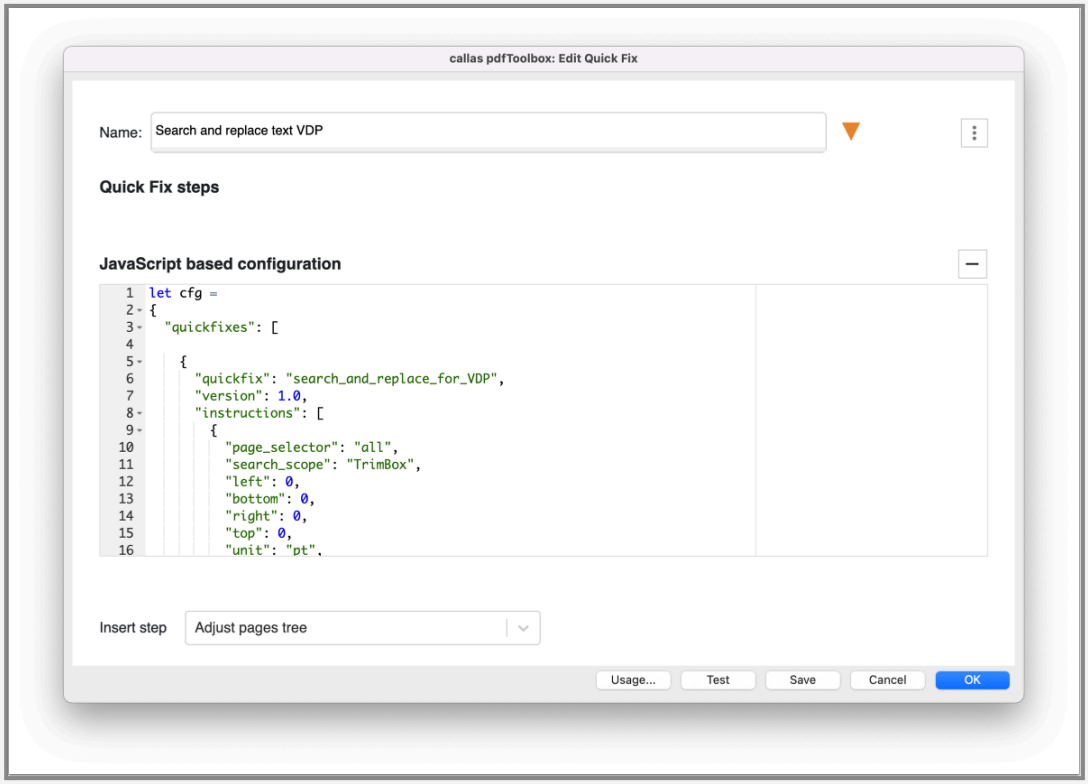
The **"JavaScript based configuration"** Quick Fix provides a text editor where the content of the JSON configuration file can be inserted. With this method, you can use the test mode for debugging.

NOTE: This Quick Fix configuration must always return a JavaScript object that contains the JSON serialisation:

```
let cfg = {

        … Your Quick Fix JSON serialisation
};
cfg; //<- return JavaScript object
```

Click to copy



How to create a JSON file that replaces Text for VDP files

The following shows how to create VDP files with custom text and barcodes with the help of an example. The input file is a PDF with placeholder text that can be customized using a JSON configuration file.

Sample_Search_replace_text_VDP.pdf

search_and_replace_VDP_sample.json

To better understand the individual parameters of the JSON file, they are explained below:

Structure of the JSON

```
{
 "quickfixes": [
 {
   "quickfix": "search_and_replace_for_VDP",
   "version": 1.0,
   "instructions": [
     {
      "page_selector": "all",
      "search_scope": "TrimBox",
      "left": 0,
      "bottom": 0,
      "right": 0,
      "top": 0,
      "unit": "pt",
      "replacements": [],
      "font_locations":  {}
     }
   ]
  }
 ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                "search_scope": "all|MediaBox|CropBox|BleedBox|TrimBox|ArtBox|ab-
solute"
//                "unit": "pt|mm|inch"
```

Click to copy

The parameters "page_selector", "search_scope", "left", "bottom", "right", "top" and "unit" are responsible for the pages

and the page range where the text replacement should take place.

The "replacements" array

In the "replacements" array, the search and replace parameters are defined. The "operator" parameter can be used to specify the text to be searched for. Either you enter the exact text (Operator: equal to) or you use a RegEx expression which allows more complex search patterns (Operator: regex). More information about RegEx expressions can be found here.

Two types of content can be defined: Text and codes (e.g. QR codes, matrix codes). The "records" Array contains the data for the replacement text and/or barcode. Any number of records can be defined in an array. Based on these records, new pages are created during the Quick Fix execution. That means, for each "records" entry, a new page is added. Since there can be multiple "records" arrays, the largest array determines the number of duplications. If one of the "records" is only a string or one array is shorter than the longest array, the last value is repeated as many times as necessary.

Replace placeholder text with text

The replaced text has the same appearance as the placeholder text (font, size, color, etc.). The text alignment can be specified. Left/center/right alignment does not change the width of the text. If "block_aligned" is selected, the text is compressed or stretched within limits.

```
{
   "operator": "equal_to",
   "search_for": "$city",
   "records":  ["Madrid", "Amsterdam", "Berlin"],
   "alignments": "right_aligned"
},


//                "operator": "regex|equal_to"
//                "alignment": "right_aligned|left_aligned|cen-
ter_aligned|block_aligned"
```

Click to copy

Replace placeholder text with barcodes

Placeholder text can also be replaced with a 1D or 2D code. The codes will be bottom aligned at the base line of the placeholder text. Left/center/right alignment will work in the same way as for text replacement.

```
{
   "operator": "equal_to",
   "search_for": "$qr",
   "records": ["8676780-h5s-dh5-7dh-jd7-xpK", "8676792-zxb-vb8-kb4-q78-aLf",
"8676812-84H-1v2-ddG-ivh-uiV"],
   "barcode":
   {
     "type": "QR-Code",
     "width": "20mm",
     "height": "20mm",
     "modulewidth": "0.33mm",
     "textplacement": "below",
     "barwidthreduction": "0%",
     "quietzoneleft": "0",
     "quietzoneright": "0",
     "quietzoneunit": "pt"
   },
   "alignments": "right_aligned"
}


//                "operator": "regex|equal_to"
//                "textplacement": "above|below|none"
//                "quietzoneunit": "pt|mm|in|X|cm|m|ft|pc"
//                "alignment": "right_aligned|left_aligned|center_aligned"
//                "type": see link below (Supported Barcode symbologies)
```

Click to copy

A list of supported Barcode types can be found here: List of supported barcodes and matrix codes.

An explanation of the respective barcode object parameters (such as modulewidth, barwidthreduction or quietzoneleft) are listed in this article: Extended list of parameters for the barcode object.

Example for EAN 13 Code ("records" must contain a thirteen-digit number)

```
{
  "operator": "equal_to",
  "search_for": "$EAN",
  "records": ["1234567890128", "1234567890128", "1234567890128"],
  "barcode":
  {
    "type": "EAN 13",
    "width": "50mm",
    "height": "15mm",
    "modulewidth": "0.33mm",
    "textplacement": "none",
    "barwidthreduction": "0%",
    "quietzoneleft": "0",
    "quietzoneright": "0",
    "quietzoneunit": "pt"
  },
  "alignments": "right_aligned"
}
```

Click to copy

Result:



The "font_locations" parameter

The "font_locations" parameter contains the data that is required for a successful font replacement. Here you can speci-

fy parameters that define where font files are located. If the PDF has all the necessary fonts fully embedded, the text replacement will not cause any missing glyphs problems. In other cases, you must specify which location should be searched by the engine, or you specify the fonts separately using a path.

```
{
   "in_fonts_folder_next_to_input_file": false,
   "in_any_folder_next_to_input_file": false,
   "next_to_input_file": false,
   "in_user_fonts_folder": false,
   "in_system_fonts_folder": true,
   "use_paths": false,
   "paths": [
      "../some_folder"
   ]
}
```

Click to copy

Result

Since three `records` were defined in the JSON configuration file for each placeholder, the output file contains three pages with individual text:


Input PDF template
2022 Tour
Coming to a place near you
$city
$date
Ticket fee: $fee
Ticket holder: $ticket_holder
$qr
World Tour
Coming to a place near you
Berlin
August 24, 2022
Ticket fee: 39,- €
Ticket holder: Rainer Maria Knilch
World Tour
Coming to a place near you
Madrid
August 4, 2022
Ticket fee: 19,- €
Ticket holder: Markus Richter
World Tour
Coming to a place near you
Amsterdam
August 14, 2022
Ticket fee: 29,- €
Ticket holder: Maximilian von Dannen

15.2 Fast VDP mode

In order to understand when this feature can be successfully used we need to understand the nature of VDP files. How exactly are they different from other PDF files?

VDP files often have several thousand pages, but the pages are not fully independent from each other. There typically are repeating page sequences. Each such sequence forms a "record". A record could for instance contain front and back page of a postcard. Another example is a flyer with 8 pages, the record size would then be 8. Usually lots of content is "static", that means it is present on all corresponding pages in the records. In addition there is also some content that is present only in the specific record. That could eg. be the address on the postcard.

In the internal PDF structure, a PDF usually uses form XObjects for the static content. The corresponding pages in different records all reference the same static form XObjects. This technique does not only reduce the file size of the VDP file, it is also important for processing the PDF.

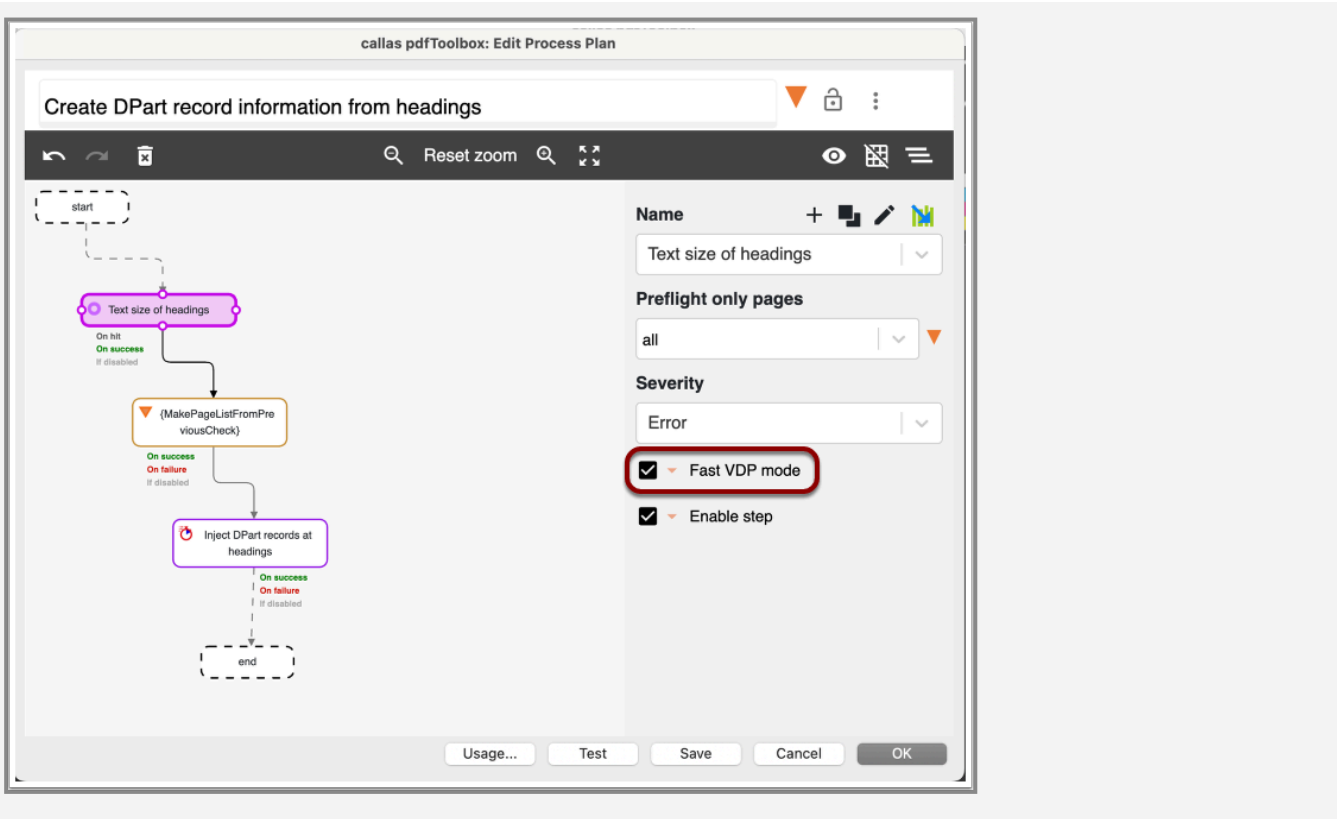
Unfortunately there is one complication: A form XObject may depend on the context in which it is invoked. An extreme example would be a form XObject that draws a rectangle using a color that is not defined in the form XObject itself, but in the parent content from which it is invoked. That means the color of the rectangle when rendered would be different, depending on the parent context. You will probably never see such an extreme case in a real world PDF file, however that a form XObject is fully self contained is definitely not a given in a PDF file.

Since form XObject may depend on the context from which they are invoked, pdfToolbox is not able to reuse any processing results (during analysis or correction). Instead it has to process the same form XObject again and again whenever it is invoked. This is for most PDFs not a big issue, since not many form XObjects are reused.

That is, however, different in VDP files. As we have seen an important characteristic of VDP files is, that lots of the content is contained in reused form XObjects. These form XObjects are almost always invoked from the same context and therefore independent - otherwise the records would not look the same.

For VDP files it is not only safe to reuse the results for the static form XObjects - that in addition has a huge performance impact. For this reason pdfToolbox 14 introduces "Fast VDP mode". In this mode static form XObjects are identified and results (from correction or analysis) are saved (cached) and used again when the same form XObject is invoked. As a result processing a VDP file may only use a fraction of the time that would be needed in normal mode. For regular PDF files on the other hand you will usually not notice a big performance difference. In rare cases, however, you will see incorrect results. So it is important to only use this mode for VDP files or step and repeat results where the same content is used again and again.

"Fast VDP mode" can be enabled in a Process Plan for Checks, Fixups and Profiles.



Fast VDP mode can also be enabled on command line (via --fastvdpmode) or in the SDK. It is also available in Test mode.

15.3 DPart metadata injection

DPart can be added to a PDF file using Quick Fix as described in a chapter of the QuickFix feature overview.

The InjectDPart feature in Quick Fix uses a page list as input. In a Process Plan this list can be dynamically created. pdfToolbox (version 12 or newer) has a predefined Process Plan "Create DPart record information from headings" that does so. It uses the text size as an indication for a heading and creates a DPart structure in which each page with a heading is referenced as a record start.

15.4 Distribute form XObjects on layers

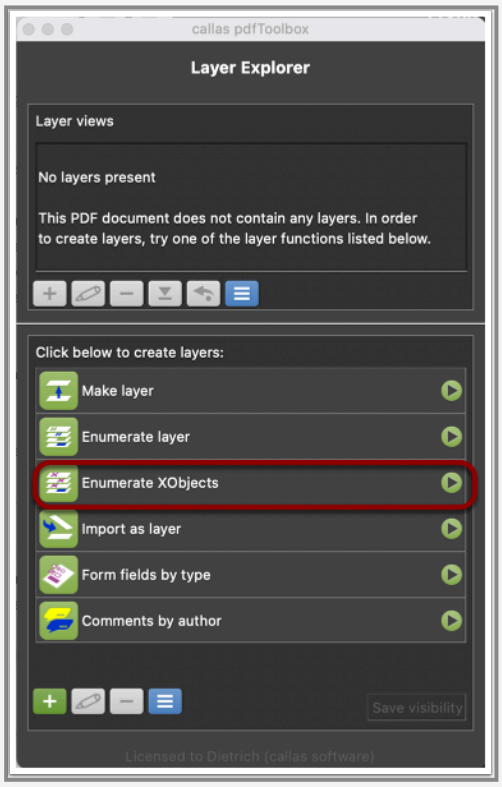
Form XObjects in a PDF are used to encapsulate page content. That is useful e.g. to import new content such as a stamp that is placed on a page, in imposition when pages are placed on a sheet, to group objects together for transparency or to reuse the same content several times. The last use case has a lot to do with variable data print: If static (reused) content is in a form XObject that is referenced whenever used the output engine (RIP, DFE) can process it once and reuse it for rendering (caching) which means that it will run much faster. If that is, however, not the case that can slow down the output significantly. If caching works smoothly the first instance of a page needs as long as it needs to process it, corresponding later pages are processed much faster. However, even if reused content is in a form XObject it may not be possible to cache it, e.g. when it interacts with other content via transparency.

In order to analyze whether a VDP PDF may have caching issues a tool that allows you to visualize the form XObject structure in a PDF would be helpful and it would be even better if it would be possible to modify it.

Distribute form XObjects on layers

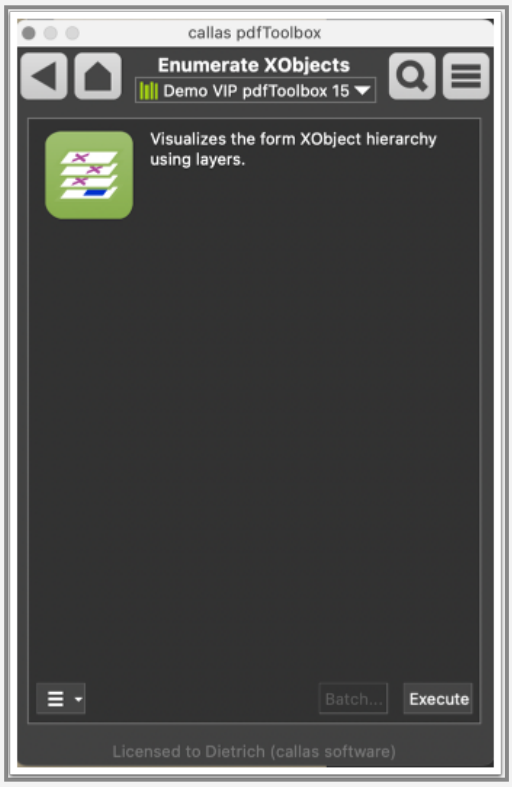
In pdfToolbox you can use layers to visualize the form XObject structure in a PDF. There are three ways to access this feature:

Layer Explorer



The Layer Explorer an be found in the Tools menu. If a PDF is opened that does not already have layers it shows an item "Enumerate XObjects" that then opens the:

Enumerate XObjects action in the Switchboard (group Layers)



If you click on "Execute" the form XObjects of the current PDF are distributed onto new layers and the Layer Explorer is opened.

"Distribute form XObjects on layers" Quick Fix

If you want to distribute form XObjects onto layers with pdfToolbox CLI, SDK or server you can use the Quick Fix with this name.

Investigate a form XObject structure in Layer Explorer

Let us use an example PDF with 10 pages. This PDF has been derived from the PDF/VT sample files.

Travel V1.0.1 - 10 pages badge front.pdf

After you have distributed the form XObjects in this PDF on layers the Layer Explorer will look like this:



The layer structure visualizes the form XObject structure

The PDF has now 8 new layers.

In the PDF red rectangles indicate the position of the form XObjects. These red indicators all belong to the new layer "Form XObject BBoxes". A red rectangle is only visible if this layer is visible and the respective form XObject that it belongs to is visible too.

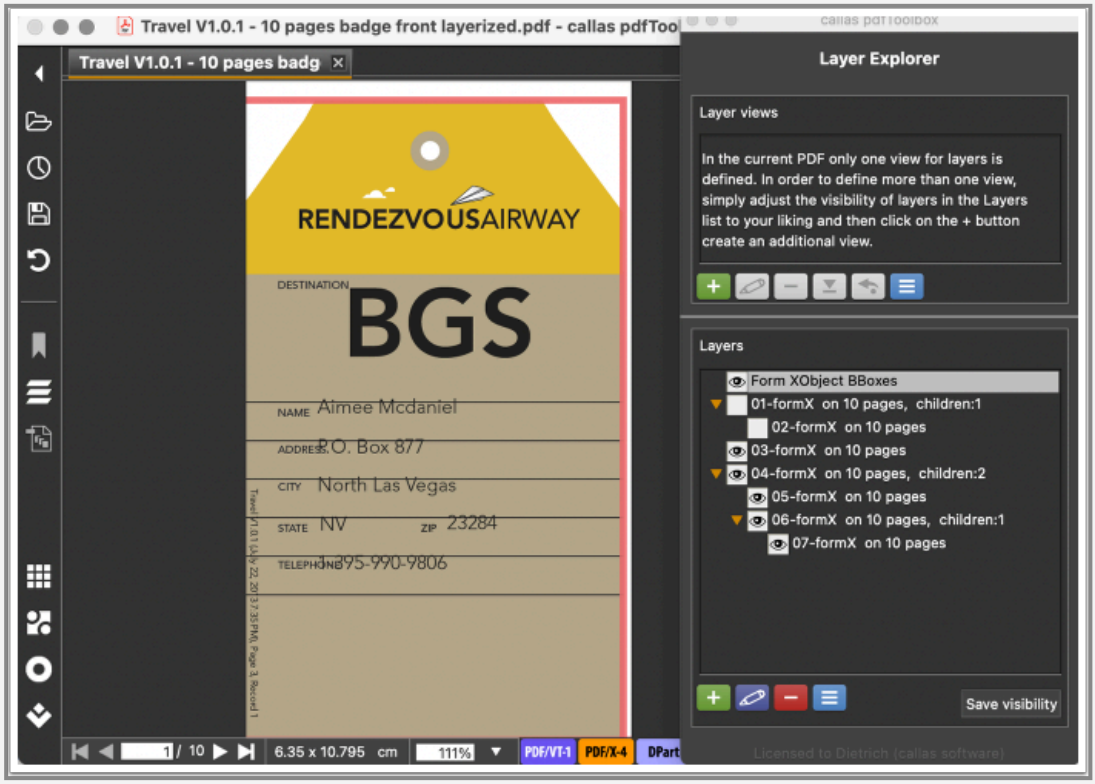
The other new layers tell you something via their names: The name starts with an index number. It also indicates on how many pages it is (re)used and if it has child layers how many that are.

Form XObject can contain other form XObjects thereby creating a hierarchy and the layers are created in a structure that corresponds to that hierarchy.

A child layer is only visible if all of its parent layers are visible.

Investigation

If you switch the layer "01-formX on 10 pages, children:1" off you see that there is an additional, little smaller layout completely covered by the visible objects. Apparently a left over from the designer?

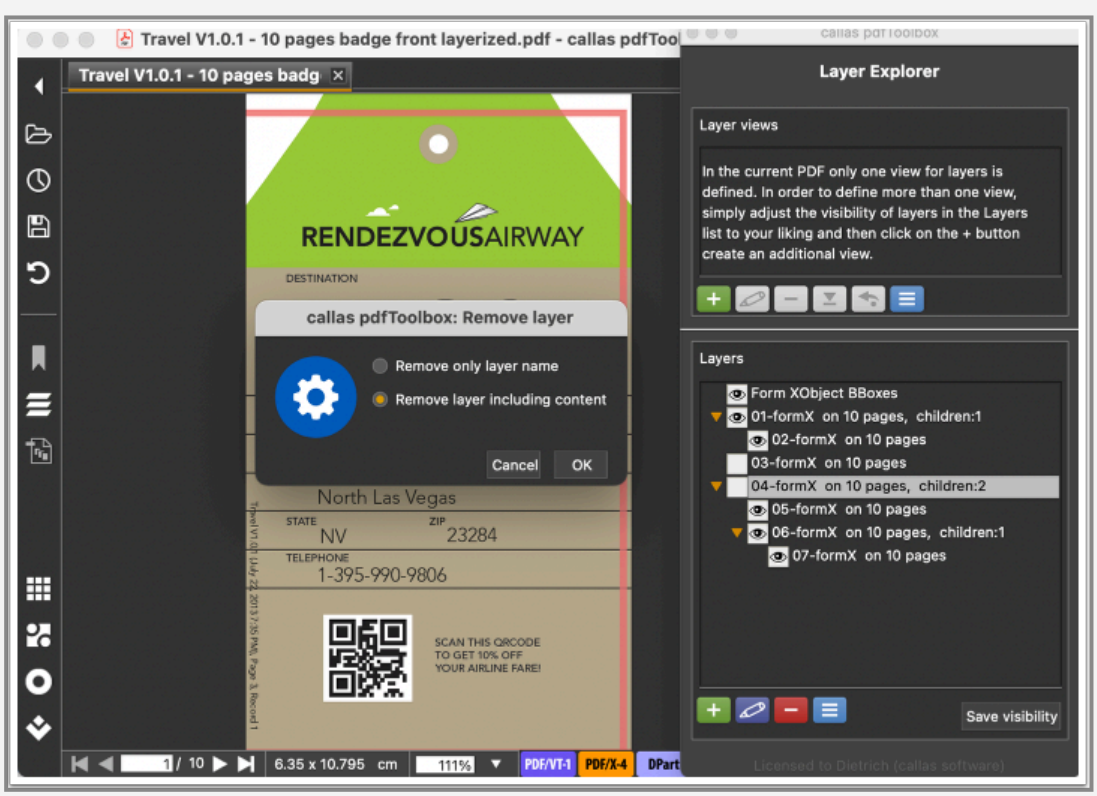


The output RIP will still have to process it which will slow down rendering. In our not very complex 10 page PDF that is not a big deal, but that may be different in a PDF with ten thousand pages and if they e.g. use transparency.

In any case would it be good to remove or disable the hidden content.

Optimizing a PDF for rendering in the RIP

You could now remove the hidden XObject by selecting it in the Layer Explorer and clicking on the red "-" symbol.



You will see a dialogue that allows you to remove it with its content.

This approach works fine, but would in a "real" VDP file with thousands of references to this form XObject have an important downside: All content streams that reference it will have to be modified and that may take very long.

A smarter way is to make this hidden layer initially invisible which means that a RIP will (normally) ignore it. You can modify the initial visibility by switching all layers off that should be invisible and then use the "Save visibility" button at the bottom of the Layer Explorer.



This will just bring up an additional, explanatory dialogue (that you may disable for the future once understood) and you can then save the modified result. Since only parameter has to be modified for the layer that will be a very quick operation.

16. Debugging of Profiles and Process plans

16.1 Why a test mode?

pdfToolbox Desktop allows you to create Profiles, Process Plans, Fixups and Checks. And all of those can evolve to be quite complex, especially in the case of Process Plans. Traditionally, the following was often repeated many times when creating for example a new Process Plan:

- Modify the steps in the Process Plan
- Save the Process Plan
- Open a test file
- Run the Process Plan
- Specify a location for the resulting file
- Ask to overwrite the previous result file, or give the result file a new name
- Notice that things don't work

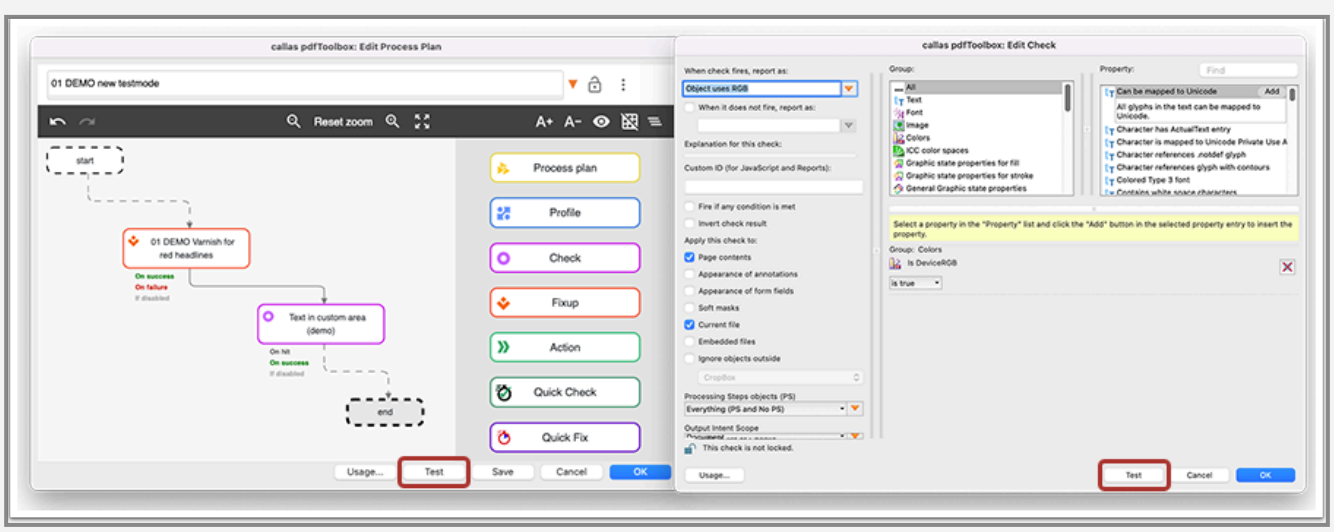
This cycle not only takes a lot of time, it's also often error prone as it's very easy to accidentally overwrite your original test file for example.

Test mode streamlines this process

Starting with pdfToolbox 10, there is a built-in test mode that streamlines this process. It lets you test a Process Plan, Profile, Fixup or Check from within their respective editor. And it lets you test how they do without having to select a place to save the result file nor be afraid you're going to ruin an existing file.

16.2 How to use test mode

Test mode can be started from the Process Plan, Profile, Fix-up, Check and several other editors in pdfToolbox Desktop. Open any of these editors and you can launch test mode directly from within the window by clicking the "Test" button:



This will immediately execute the Profile you're currently editing and open the Test Mode window. In addition to Profile testing, Test Mode offers numerous other helpful tools, all detailed below.

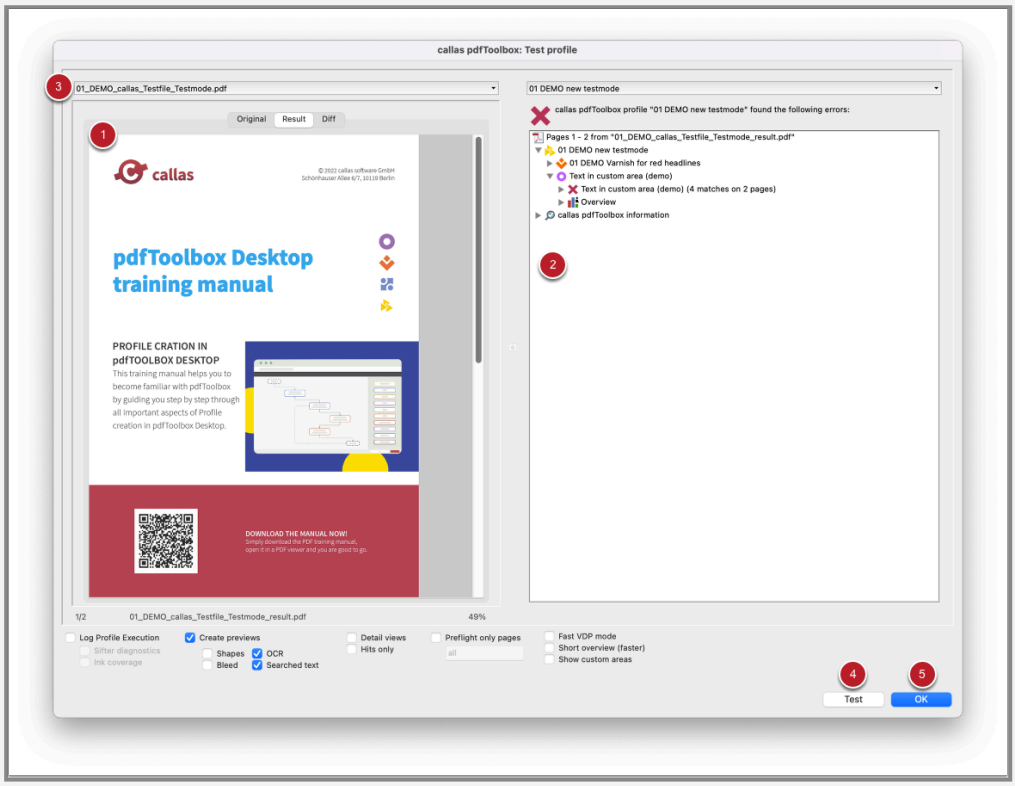
Tip: Starting Test Mode directly from the Profile Window

When the Profile window is open, you can also go directly into Test mode to test a Profile, Process Plan Check, or Fixup by pressing the "Option" key on Mac or the "Alt" key on Windows (see the screenshot below).



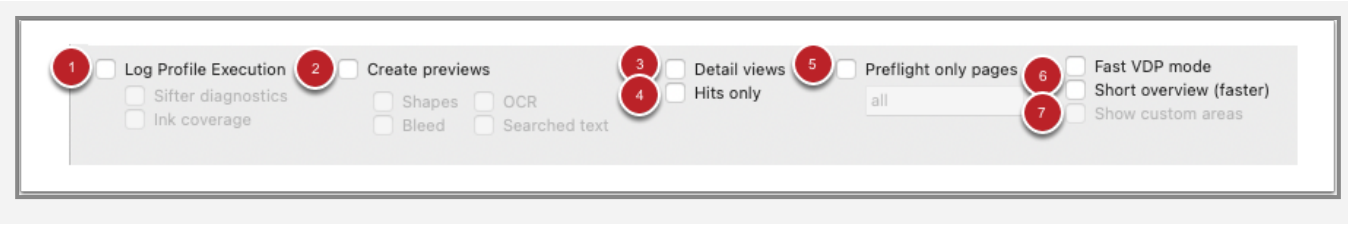
Test mode functionality

1. General functionality



1. Test Mode in pdfToolbox automatically duplicates the document you're working on, execute the Profile you are editing on, and displays the result on the left side of the window. This allows you to immediately see the result of the processing without having to choose a destination for the modified PDF file or worry about overwriting your original document. You can toggle between Original, Result and a Diff of the two.
2. On the right side, the Test Mode window displays the execution log, as you would normally see in the Profiles window when, for example, you run your process Plan. This part allows you to see if the processing was successful and what the result is.
3. To test on different PDF documents, the drop-down menu lists all open and previously used documents in pdfToolbox Desktop and allows to open new PDF documents. Selecting another document will immediately apply the current profile. In addition, you can activate the page geometry boxes so that they are displayed on the PDF document.
4. The "Test" button re-runs the operation. In other words, it creates a fresh copy of the test PDF, runs the Process Plan (or other item you're editing) on it, and displays the results. Note that this is especially useful for things that are based on Place Content, as it allows you to edit the place content template used in an external editor, save your changes in the external editor, return to pdfToolbox's test mode window and press "Test" to run the changes in your template.
5. The "OK" button ends the test session.

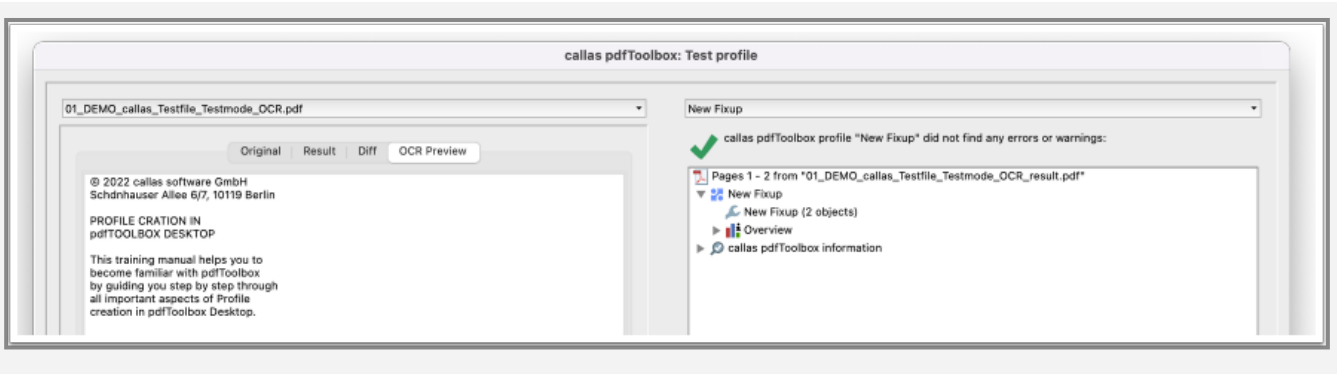
2. Toolbar



1. Log Profile Execution: By checking the "Log Profile Execution" check box, you can save the intermediate processing results. This is especially useful for Process Plans that run through many steps (more infos here). If you are using Log

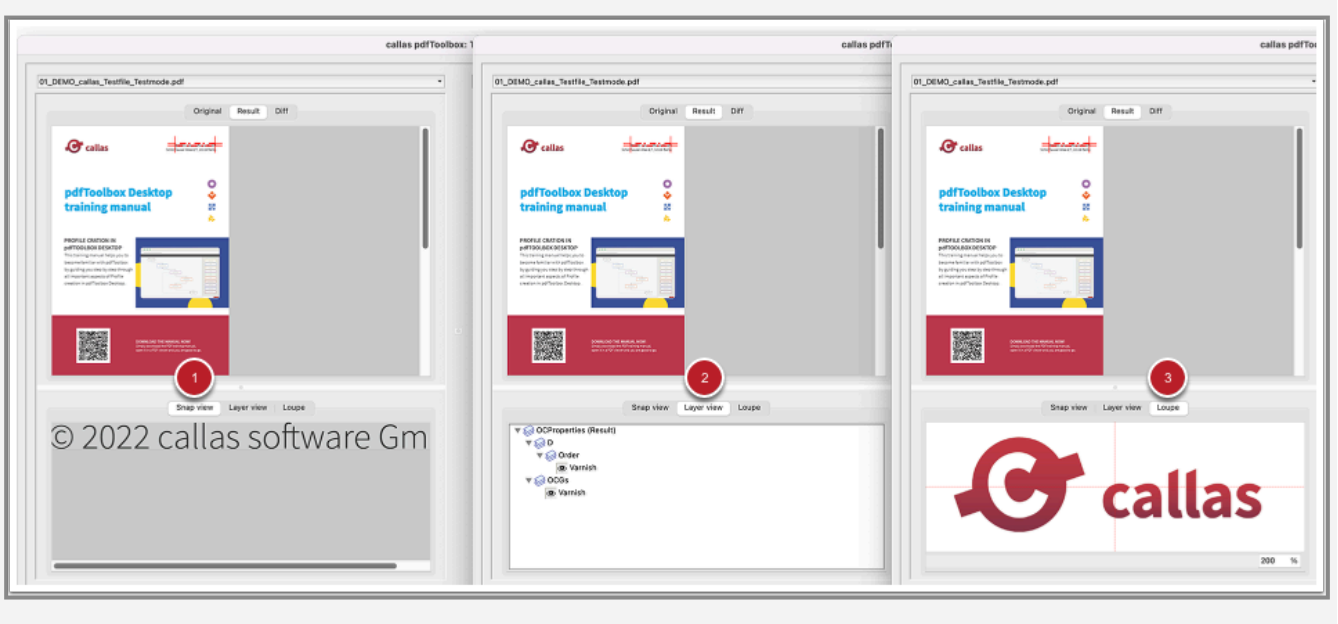
Profile Execution you may create additional images that further demonstrate internal processing by checking "Sifter diagnostics" and/or "Ink coverage" (if you are using one of these).

2. Create previews: You can generate additional previews for Shapes (if you are testing a "Create and Apply Shapes" Fixup), OCR (if you are testing a Fixup that generates OCR text), Bleed (for irregular shapes), and Searched Text (if you are testing a "Search and Replace Text" QuickFix). To generate the previews, select a checkbox and click the Test button again. The new previews will appear as additional panes on the left.

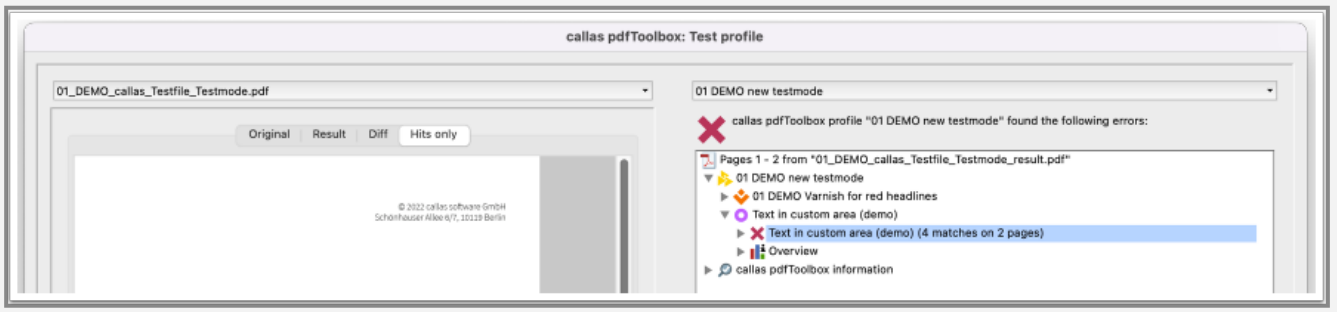


3. Detail views: When you check this box, there are three different views that appear below the PDF preview:

1. Snap view displays selected hits from the execution log on the right-hand side for easy reference.
2. Layer view provides a technical representation of PDF layers, allowing for switching individual layers on and off.
3. Loup view enables faster PDF zooming and analysis, with adjustable scaling factor and real-time cursor position.



4. Hits only: Another way to see all page objects identified by a Check, isolated from the rest of the document (this view only works if the Check is snippet-based. If the Check is page-based or document-based, this view is empty).

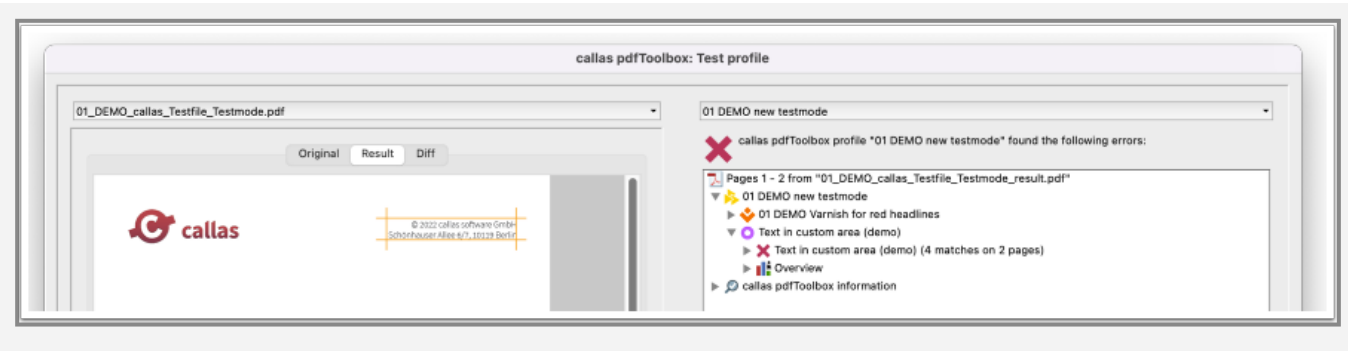


5. Preflight only pages: Ability to limit the number of pages to be preflighted.

6. Fast VDP mode and Short overview (faster): Enable faster processing. "Fast VDP mode" speeds up processing for variable data files as explained here. With "Short overview (faster)" the result window will show only sparse information in the Overview section (because the time was saved to gather it).

8. Show custom areas: If you are configuring a Check or Fixup that is limited to a custom area, you can select this checkbox to place an orange rectangle on the PDF document that highlights the custom area specified in the Check or Fixup. This

makes it really easy to see if the custom area is in the right place.



16.3 Purpose of logging feature

Especially in higher volume environments it can be of interest to have detailed information about the pdfToolbox processes as they happen. Suitable logging data can be a good foundation both for monitoring processing of PDF files as well as for post mortem analyses or analyses aiming to improve overall behavior of the whole processing system.

The logging feature that forms a part of pdfToolbox from version 10 onwards provides logging information in the form of JSON files that get written immediately at the following steps during an invocation of pdfToolbox:

- When pdfToolbox is **launched**; only very basic data about the PDF file to be produced – such as file name – is available at this time, plus some information about the environment or the command line parameters
- When pdfToolbox **initiates processing** of a PDF file; additional data about the PDF to be processed is provided – only aspects though that do not require no 'deeper' analysis in order not to hamper runtime behavior – such as file size or PDF version or size of the first page; in addition all variables – as evaluated at the beginning of processing – are included as well.
- When pdfToolbox **completes** processing of a PDF file; at this stage various aspects of overview information is readily available, such as colors, fonts or ICC profiles used, output intents, conformance with PDF/X or other PDF standards.

Command line and Server versus Desktop

Logging is available in command line, Server and Desktop versions. It is not expected that logging on desktop is normally being used. Instead, on Desktop activating logging will help understand how logging works, and often makes it easier to design and test use of logging data outside of an automated production setup.

How to process logging data

Research has shown that most callas software customers who operate higher volume environments prefer any logging data to be provided rather as many small files using JSON over using other formats or a small number of large files. Beyond the way the logging data is provided, no assumptions are being made how these data are processed.

16.4 Activating logging

Activating logging in command line version

New parameter:

```
--trace[=folderpath]
```

Example:

```
./pdfToolbox --trace='/some-logging-folder'  './some-pdftoolbox-profile.kfpx'  '.
/input.pdf'  -o='./output.pdf'
```

Subfolder structure

By default the logging feature creates a subfolder structure inside the target folder for logging based on values for year, month and day, plus the `app_uuid` value (which can also be found inside each of the logging files):

```
YYYY
  MM
    DD
      app_uuid
        launch.json
        init.json
        finish.json
```

Click to copy

```
2016
  10
    31
      0a0ceba3-7ca9-421f-a973-8caae2950690
        launch.json
        init.json
        finish.json
```

Click to copy

Suppressing creation of subfolder structure

New parameter:

```
--trace_nosubfolders
```

Example:

```
./pdfToolbox --trace='/some-logging-folder'  --trace_nosubfolders './some-pdftool-box-profile.kfpx'   './input.pdf'  -o='./output.pdf'
```

Activating logging in Server version

In the "pdfToolbox Server: Job" configuration window, add the following to the text field under "Additional CLI parameters":

```
--trace[=folderpath]
```

Example:

```
--trace='/some-logging-folder'
```

Suppressing creation of subfolder structure in Server version

In the "pdfToolbox Server: Job" configuration window, add the following to the text field under "Additional CLI parameters":

```
--trace[=folderpath] --trace_nosubfolders
```

Example:

```
--trace='/some-logging-folder'  --trace_nosubfolders
```

Important note: If logging is to be enabled for processing of all files in pdfToolbox Server, this setting must be added to each job configuration.

Activating logging in Desktop version

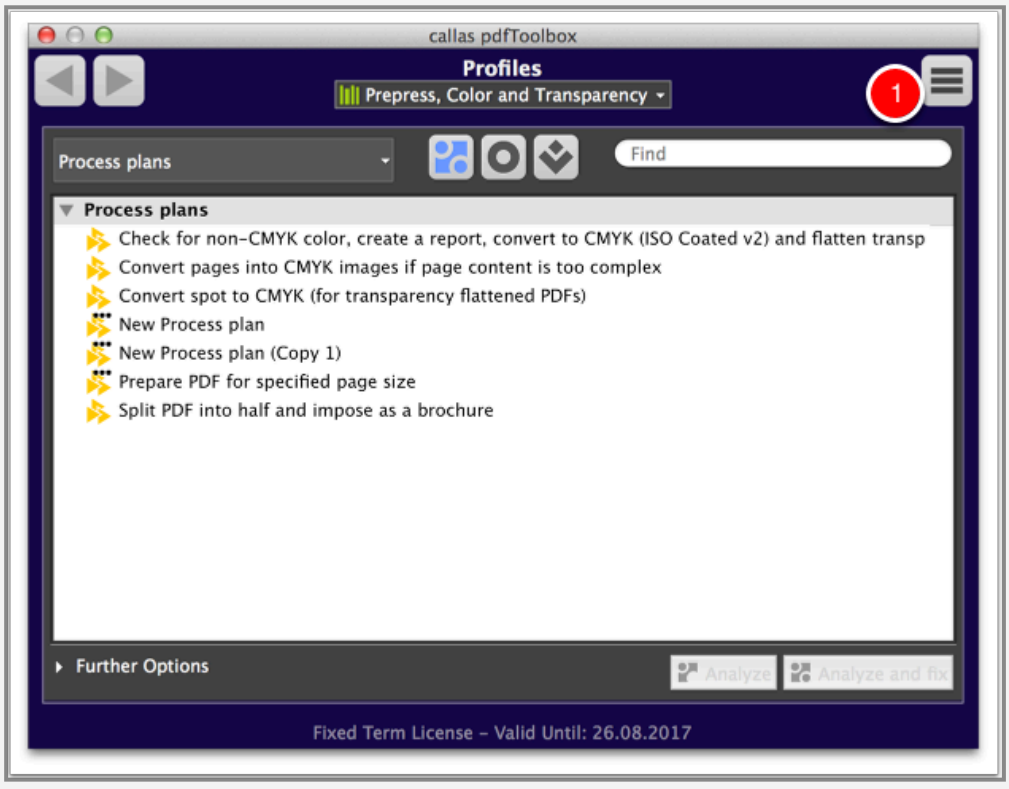
When "Log Profile Execution" (can be found in the menu in upper right of Profiles/Checks/Fixups window) is activated, then a sub folder "Additional logging" will be created if not already present, and inside that folder the logging files will be added in the same fashion / with the same naming conventions and sub-folder structure as for the CLI and Server version.

16.5 How to create a detailed log when executing Process Plans (or Profiles, Checks or Fixups)

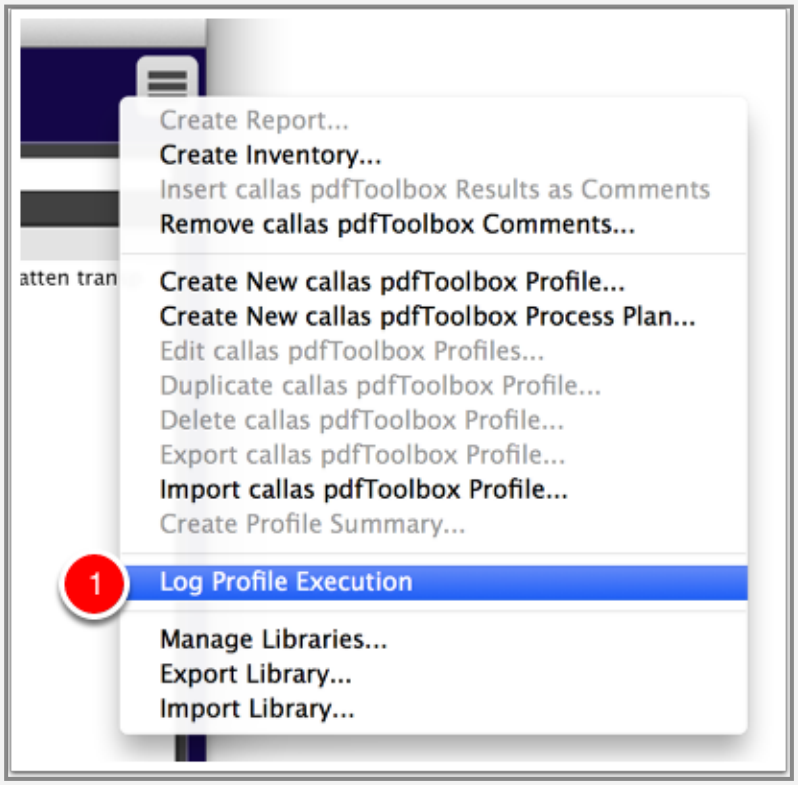
Especially when creating a Process Plan, it can become necessary to inspect intermediate results (such as PDF files in their state in the middle of a Process Plan's processing steps and preferably some details about each step in the form of a log file or similar). This can help to understand and to optimize the configuration of the steps in a Process Plan and their inner workings as a whole.

Note: This logging feature is not only available for Process Plans, but also for Profiles, Checks, and Fixups.

Activate logging



To activate logging, just click within one of the three sections of the Profile window on the flyout menu button in the upper right corner (1).



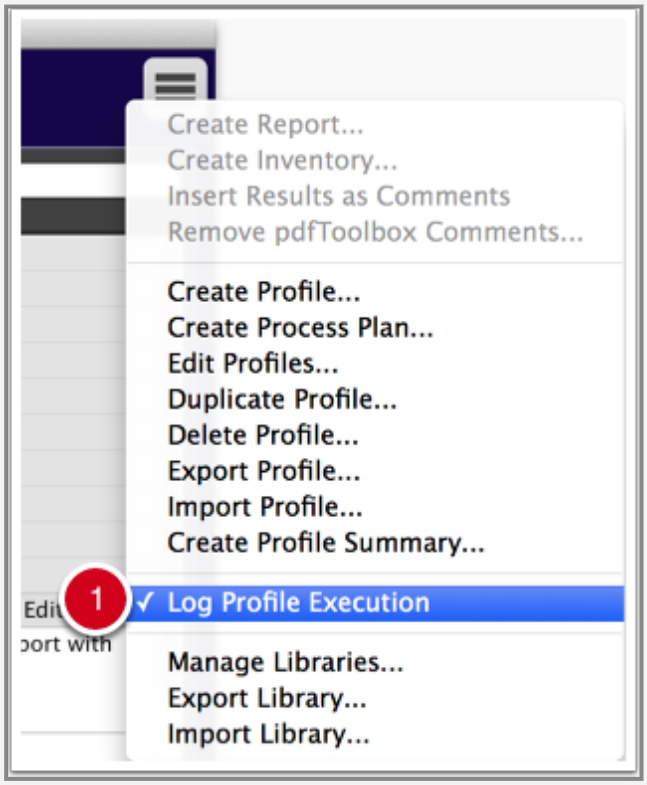
Select the entry "Log Profile Execution" (1).

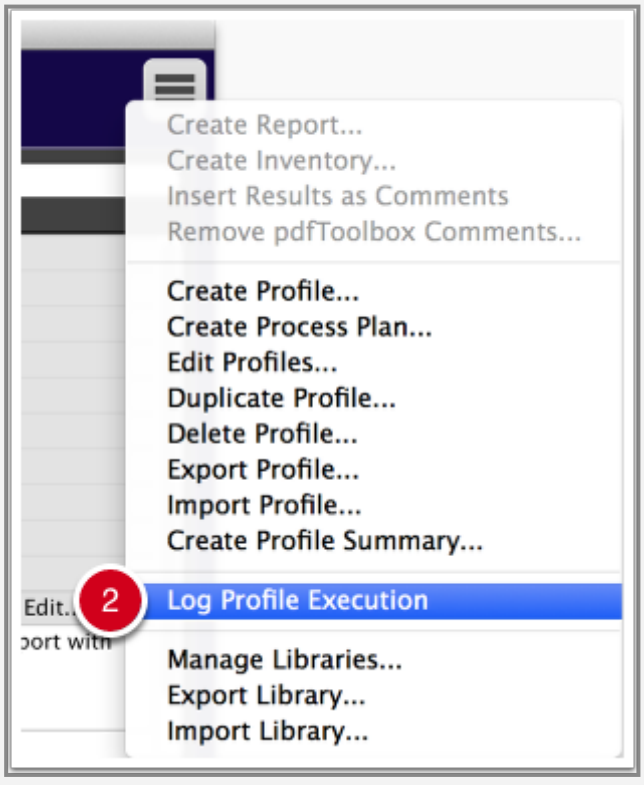
This option activates the logging of the execution of any Process Plans, Profiles, Checks and Fixups.

A check mark ("✓") in front of the menu item indicates that logging is active.

Deactivate logging

In order to deactivate logging, simply execute the menu item again (1). The check mark ("✓") will then disappear from the menu item (2).





Execute a Process Plan (or Profile, Check or

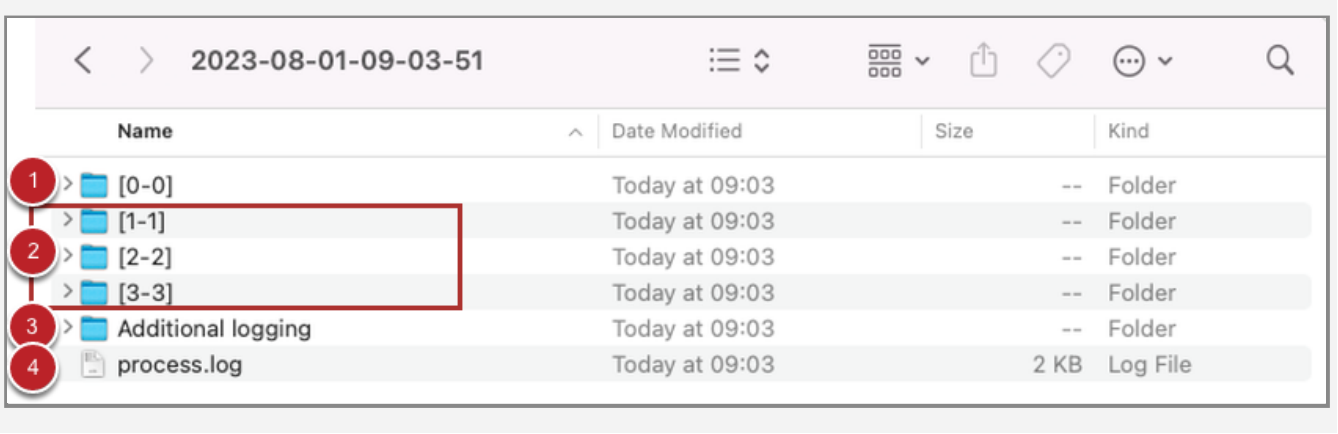
Fixup)

Execute a Process Plan (or a Profile, Check, or Fixup).



Explore folder with logging information

After processing the PDF, a window will open in Finder (on Mac OS X) or in the Explorer (on Windows), revealing a folder (having a time stamp as folder name) with all the logging data and associated files inside it.



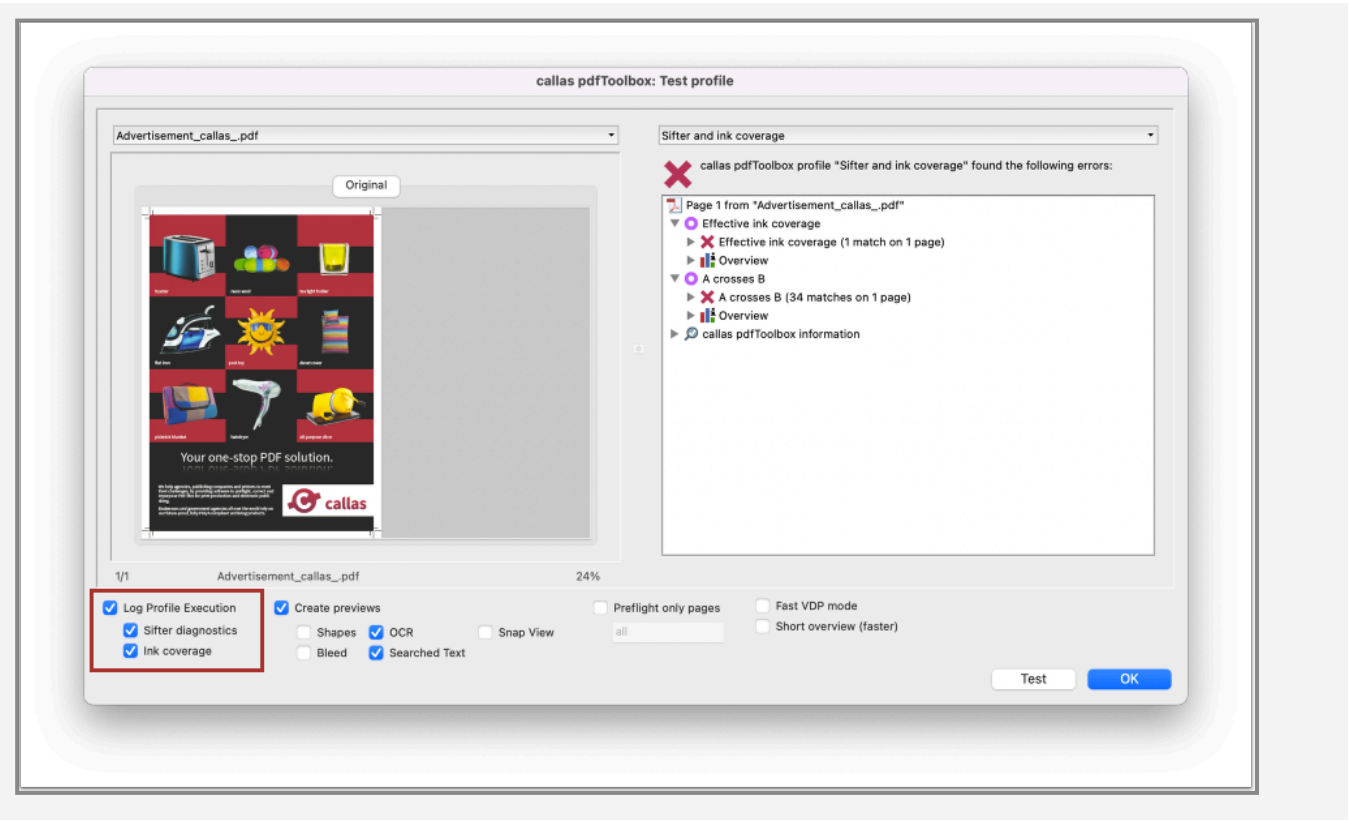
Structure of the files subfolders in the logging folder

This folder will contain the following items:

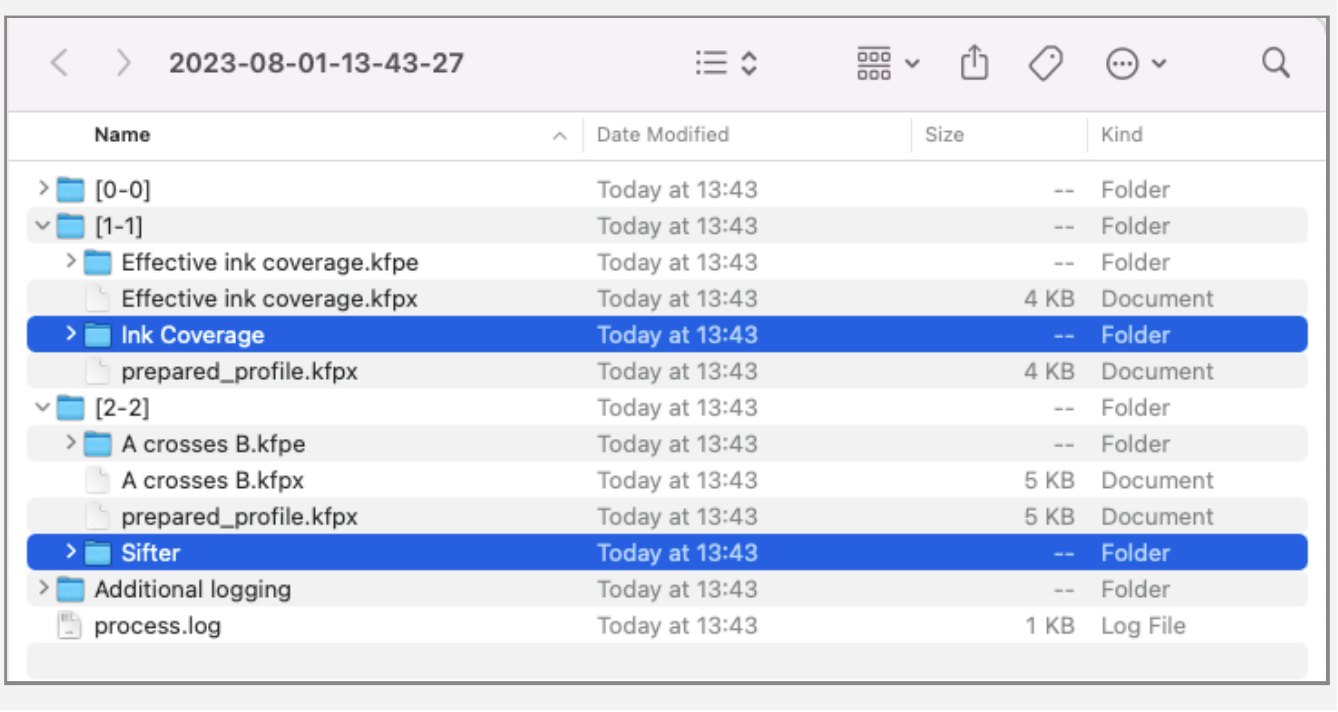
1. The first subfolder ([0-0]) contains the original PDF file and a "kfpx" file with the entire Process Plan (or Profile, Check or Fixup) that was just executed.
2. Subfolders with intermediate results (if applicable) for each step in a Process Plan (Profiles, Checks and Fixups only will have one such subfolder) and the corresponding "kfpx" file
   - The subfolder's name will consist of the sequence number and the step number in square brackets
   - The subfolder will also contain the Profile, Check or Fixup associated with the respective Process Plan step as a "kfpx" file
3. The "Additional logging" subfolder contains three JSON logging files that are created during each successfully completed invocation. See the next article for more information.
4. Logging information in a file named "process.log".

Additional logging information using Test Mode

It is also possible to enable logging in Test Mode. In addition to the "regular" logging information (explained in the section above), you have two optional checkboxes to generate logging information for Sifter and/or Ink Coverage checks.

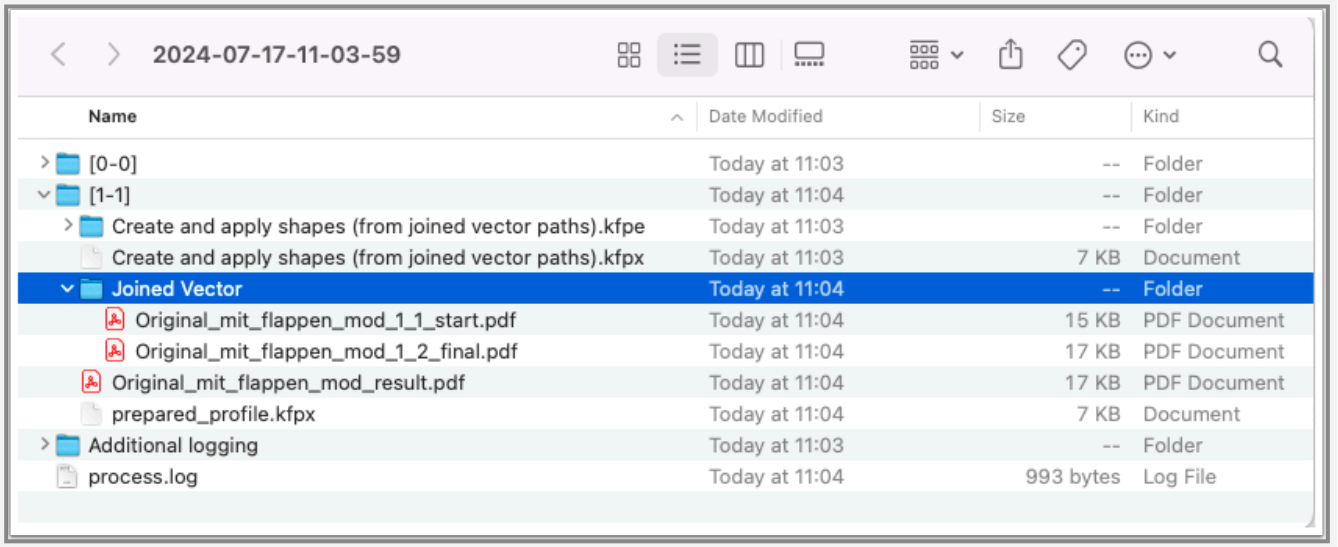


This means that if you are using the Log Profile Execution in Test Mode, you can create additional images that further demonstrate the internal processing by checking Sifter Diagnostics and/or Ink Coverage (if you are using either of these). The images are stored in a subfolder of the corresponding check named "Ink Coverage" or "Sifter" (subfolders shown below as examples).



Additional logging information for "Create and apply shapes - From joined vector paths"

When testing a "Create and apply shapes" Fixup that uses the option "**From joined vector paths**" in Test Mode, additional logging information is available in the Log Profile Execution folder to visualise the path objects found and used :



The "Joined Vector" folder always contains two PDF previews per document page (if you have a multi-page PDF file, you will get a pair of previews for each page):

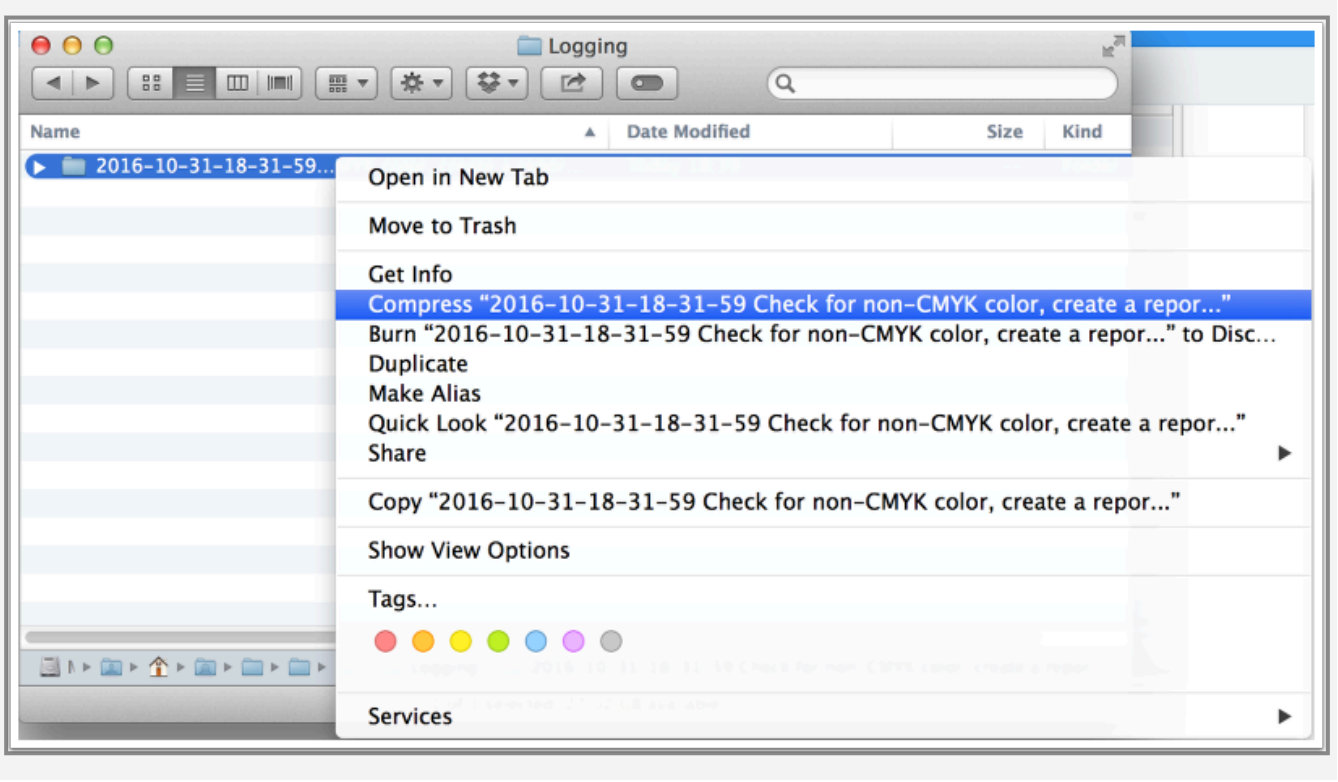
1. **[filename]_[page number]_[continuous counter]_start.pdf:**
   Visualisation of all lines and curves to be considered by the Shape Fixup. Each path/curve is displayed in its own colour. In addition, each path/curve has a number at the start and end (the number at the start is the same color as the path/curve and the number at the end is grey).
2. **[filename]_[page number]_[continuous counter]_final.pdf:**
   Contains only the paths/curves of the newly created shape. They are numbered in the order in which they are connected. Different colours indicate different path segments. Green is used for the start point and red for the end point of the lines/curves.

Sending the logging information for support cases

When requesting support from the callas support team, you might get asked to send the complete logging package.

This will help us to determine how processing was executed on your computer, and why something may not be working as expected. For easy and safe transfer please compress the entire folder into a ZIP archive by clicking the right mouse button after selecting the respective folder:

- Mac OS X: "Compress..."
- Windows: "Send to..." - "ZIP compressed folder"


Logging
Name
Date Modified
Size
Kind
2016-10-31-18-31-59...
Open in New Tab
Move to Trash
Get Info
Compress "2016-10-31-18-31-59 Check for non-CMYK color, create a repor..."
Burn "2016-10-31-18-31-59 Check for non-CMYK color, create a repor..." to Disc...
Duplicate
Make Alias
Quick Look "2016-10-31-18-31-59 Check for non-CMYK color, create a repor..."
Share
Copy "2016-10-31-18-31-59 Check for non-CMYK color, create a repor..."
Show View Options
Tags...
Services

16.6 The JSON log files

There are three JSON logging files that will be created during each successfully completed invocation of pdfToolbox:

1. **launch.json**:
   immediately created upon launching pdfToolbox CLI or on invocating processing of a PDF in pdfToolbox in the desktop version; neither is the kfpx file loaded yet nor is the PDF file to be processed accessed yet; only minimal information about the environment and context is collected, including the content of the command line, plus an automatically generated unique ID (app_uuid) and the job ID (when it was passed as a parameter on the command line)
2. **init.json**:
   gets written once the kfpx file is loaded and parsed, the PDF file to be processed is opened and some basic information extracted from the PDF file
3. **finish.json**:
   gets written once pdfToolbox exits (regardless whether processing was successful or not).

If *init.json* or *finish.json* are seemingly missing from logging, this is an indication that pdfToolbox terminated prematurely without writing those files, which can serve as a good basis for post mortem analysis. If *init.json* is not written, something probably went wrong while reading the kfpx file, accessing the PDF file for some basic analysis, or similar. If only *finish.json* is missing, something went wrong during execution of the kfpx profile, or when creating reports or producing some other output.

Structure of 'launch.json'

name	value
verb	**type of logging file**. Possible values: *launch.* *Note: The other values (init, finish) are only used by the corresponding init.json and finish.json files.*
app_uuid	automatically generated ***unique ID*** **on a per invocation basis**. This unique ID is guaranteed to be the same value across all log files belonging to the

name	value
	same invocation (and also identical to the parent folder name containing the corresponding .json files)
timestamp	**timestamp**. Format: *YYYY/MM/DD hh:mm:ss*
timestamp_hour	the **hour** portion of the timestamp in the format *YY* (e.g. *20*)
timestamp_month	the **month** portion of the timestamp in the format *MM* (e.g. *07*)
timestamp_weekday	the **weekday** portion of the timestamp in the format *W* (e.g. *3* for Wednesday)
job_id	a **job ID** provided via command line argument (not available in desktop version)
process_id	**process ID** of the process in the operating system
filename	**file name** of the file to be processed
filepath	**folder path and file name** of the file to be processed
cli_params	**command line parameters**
program_name	**name of the program** (e.g. *callas pdfToolbox CLI (x64)*)
program_version	**version of the program** (e.g. 10.0.461)
platform	**platform** on which pdfToolbox is executed (e.g. *Mac OS X 10.10.5*)
machine_ips	**list of IP addresses** of the machine on which pdfToolbox is executed, in the form of an array of strings. Example: machine_ips : ["192.168.1.1", "123.45.67.89"] *Note: As machines can have more than one IP address, this entry is structured as a list of entries.* *Known limitation: implemented for Windows/Mac OS X/Linux platforms only*
machine_name	**machine name** of the machine on which pdfToolbox is executed
machine_uuid	**UUID of the machine** on which pdfToolbox is executed Note: this is a UUID derived from hardware parameters of the current machine, with some parts of the information removed such that the UUID is still unique but the hardware parameters as such cannot to be derived from the UUID.
temp_folder	**folder path to the temp folder** used during invocation of pdfToolbox

Example content for launch.json

```
{
    "verb" : "launch",
    "app_uuid" : "0a0ceba3-7ca9-421f-a973-8caae2950690",
    "timestamp" : "2016/10/31 20:08:27",
    "timestamp_hour" : 20,
    "timestamp_month" : 10,
    "timestamp_weekday" : 1,
    "process_id" : 29543,
    "job_id" : "some_job_ID_provided_through_cli_parameter",
    "filename" : "4-Catching Text in PDFs - Michael Fuchs.pdf",
    "filepath" : "/var/folders/80a56def-aa5f-418b-9685-8614ab6d41c2/0x10dd99000/4-
Catching Text in PDFs.pdf",
    "cli_params" : ["--hitsperpage=50", "--report=ERROR,WARNING,TEM-
PLATE=OVERVIEW"],
    "program_name" : "callas pdfToolbox CLI (x64)",
    "program_version" : "9.1.417",
    "platform" : "Mac OS X 10.10.5",
    "machine_ips" : ["192.168.17.63"],
    "machine_name" : "pdftoolbox_satellite_3",
    "machine_uuid" : "---6E851-41E8-5060-B0A7-C0550F43E418",
    "temp_folder" : "/var/folders/yb/16cjr4dn2c5_27r2khx1bvbh0000gn/T/com.callas-
software.pdfToolbox/32e5717f-240a-4a17-86fe-a95551808721",
    "temp_folder_hdd" : "",
}
```

Click to copy

Structure of init.json

Description of each entry in init.json – an addition to those entries described above for "launch.json". The verb entry has a value of "init".

name	value
doc_created	**timestamp** when document was **created**; same format as *timestamp* entry (e.g. "*2015/06/03 12:23:53.000*")
doc_id1	**first** of the two values in the **document ID** entry in the document (e.g.

name	value
	"*DB10E96543FE2B4E93226FA065FE83BC*")
doc_id2	**second** of the two values in the **document ID** entry in the document (e.g. "*00AFB863B1344FDDBE90D24E513AB992*")
doc_modified	**timestamp** when document was **last modified**; same format as *timestamp* entry (e.g. "*2015/06/08 17:03:18.000*")
doc_pages	**number of pages** in the document
doc_size	**size of the PDF file** in bytes (e.g. "*863672*", equivalent to ca. 863 KB)
firstpage_size	structure reflecting the **size of the first page** in the PDF (see further below for a definition of the *firstpage_size* structure)
pdf_creator	value of the **creator** entry in the document metadata (e.g. "*Acrobat PDFMaker 10.1 for PowerPoint*)
pdf_encrypted	**whether the PDF is encrypted**; possible values: 0 (not encrypted) and 1 (encrypted) *Note: for encrypted PDF files an init.json file is only written when a correct password is specified*
pdf_version	**PDF version** of the PDF file (e.g. *1.7*)
pdf_writer	value of the **writer** entry in the document metadata (e.g. *Adobe PDF Library 10.0*)
pdfa_version	if present, the **PDF/A version** (e.g. "*PDF/A-3u"*)
pdfe_version	if present, the **PDF/E** version (e.g. "*PDF/E-2r"*)
pdfua_version	if present, the **PDF/UA** version (e.g. "*PDF/UA-1")*
pdfx_oi_icc_name	if present, the name of the **PDF/X OutputIntent** profile (e.g. "*PSO Coated v3"*)
pdfx_oi_info	if present, the text in the **OutputIntent Info** field
pdfx_oi_output_cond_id	if present, the value of the **PDF/X OutputIntentIdentifier** (e.g. "*FOGRA39"*)
pdfx_version	if present, the **PDF/X version** (e.g. *"PDF/X-1a"*)
profile_filename	file name of the **kfpx profile** (e.g. "*Convert to PDFA-1a.kfpx"*)

name	value
profile_id	**internal ID of the kfpx profile** (e.g. "*P959c755539c8439e62c516c66a4a9097*")
profile_name	**human readable name of the kfpx profile** (e.g. "*Sheetfed offset (CMYK, RGB and spot colors) (GWG 2015)*")
variables	data structure representing the **variables as evaluated** upon initiating processing (equivalent to the JavaScript object *app.variables*; for details see documentation on "*Variables and JavaScript*")

'firstpage_size' entry

The 'firstpage_size' sructure reflects the various page geometry boxes:

- Each of the page geometry boxes (*mediabox, cropbox, bleedbox, trimbox, artbox*) have an array of four entries as their value, in the order left, bottom, right, top.
- Each value in the array represent is expressed in ***pt*** (inch/72)
- The ***cropsize*** represents the effective width (*w*) and height (*h*) of the CropBox, or in its absence that of the MediaBox.
- The ***trimsize*** represents the effective width (*w*) and height (*h*) of the TrimBox, or in its absence that of the CropBox, or in its absence that of the MediaBox.
- The ***bleed*** represents the effective bleed on the four sides (in the order left, bottom, right, top), based on the *trimsize*. If the BleedBox is missing, all four values are *0* (zero).

Structure of page_size entry

```
"firstpage_size" : {
  "mediabox" : [l b r t] ,
  "cropbox"  : [l b r t] ,
  "bleedbox" : [l b r t] ,
  "trimbox"  : [l b r t] ,
  "artbox"   : [l b r t] ,
  "cropsize" : [w h] ,
  "trimsize" : [w h] ,
  "bleed"    : [l b r t]
}
```

Click to copy

Example content for init.json

```
{
    "verb" : "init",
    "app_uuid" : "0a0ceba3-7ca9-421f-a973-8caae2950690",
    "timestamp" : "2016/10/31 20:08:27",
    "timestamp_hour" : 20,
    "timestamp_month" : 10,
    "timestamp_weekday" : 1,
/*  ... and all further entries defined for "launch.json" */



    "profile" : "P959c755539c8439e62c516c66a4a9097",
    "profile_name" : "Sheetfed offset (CMYK, RGB and spot colors) (GWG 2015)",

    "doc_size" : 863672,
    "doc_created" : "2015/06/03 12:23:53.000",
    "doc_modified" : "2016/10/31 20:08:28.000",
    "pdf_version" : "1.5",
    "pdf_creator" : "Acrobat PDFMaker 10.1 für PowerPoint",
    "pdf_writer" : "Adobe PDF Library 10.0",
    "doc_id1" : "DB10E96543FE2B4E93226FA065FE83BC",
    "doc_id2" : "00AFB863B1344FDDBE90D24E513AB992",

    "doc_pages" : 23,
    "firstpage_size" : {
      "mediabox" : [-10 0 581 615] ,
      "cropbox"  : [-10 0 581 615] ,
      "bleedbox" : [5 5 559 570] ,
      "trimbox"  : [10 10 551 565] ,
      "artbox"   : [] ,
      "cropsize" : [591 625] ,
      "trimsize" : [541 555] ,
      "bleed"    : [5 5 8 5]
    }


    "variables" : {
        "Calcs_for_LFP_Preflight_-_viewing_distance" : {
            "eff_min_fontsize":null,
```

```
            "eff_min_imageresolution":null
        },
        "eff_min_fontsize":200,
        "eff_min_imageresolution":40,
        "input_scalingfactor":100,
        "input_viewingdistance":10
    }
}
```

Click to copy

Structure of finish.json

Description of each entry in finish.json (in addition to those entries described above for "launch.json" and for "init.json")

name	value
retcode	the program **exit code** (see pdfTooolbox CLI manual for details). *Note: value is provided as an integer.*
duration	**duration of processing** (essentially the difference between timestamp at finish and timestamp at lauch), formatted as hh:mm:ss:ttt (e.g. "*0:00:11:045*")
doc_corrections	**number of corrections** applied during processing
doc_max_severity	**maximum severity**; defined values are: 3 = error, 2 = warning, 1 = info, 0 = no message
doc_messages	**number of messages**, i.e. the combined total of error, warning and info messages
doc_errors	**number of error messages**
doc_errors_list	an array of **error details**; each array entry contains an ***error-name*** together with its ***counter***
doc_warnings	**number of warning messages**
doc_warnings_list	an array of **warning details**; each array entry contains a ***warning-name*** together with its ***counter***

name	value
doc_infos	**number of info messages**
doc_infos_list	an array of **info details**; each array entry contains an ***info-name*** together with its ***counter***
num_images	**number of images**
num_fonts	**number of fonts**; two different font resources where the font name happens to be the same are counted as two fonts
fonts	data structure representing the **fonts** in the PDF file
num_spotcolors	number of **spot colors**; i.e. all Separation colour spaces whose name is not one of *Cyan*, *Magenta*, *Yellow*, *Black*, *All* or *None*
spotcolor_names	array of **spot colour names**; for example: "spotcolor" : ["Orange", "Purple"])
num_icc_profiles	**number of all ICC profiles**; excluding ICC profiles in output intents
icc_profiles_gray	array of **names of ICC profiles**; excluding ICC profiles in output intents; the CalGray colourspace, which strictly speaking is not an ICC based colourspace, is reported here as "CalGray"; for example: "icc_profiles_gray" : ["Generic Gray Profile", "Gamma 2.2 Gray"])
icc_profiles_rgb	array of **names of 3-component ICC profiles**; excluding ICC profiles in output intents. RGB ICC profiles are reported by their name (content of 'desc' field), the CalRGB colourspace, which strictly speaking is not an ICC based colourspace, is reported here as "CalRGB", the Lab colorspace, which strictly speaking is not an ICC based colourspace either, is reported here as "Lab"; for example: "icc_profiles_rgb" : ["eciRGB v2", "CalRGB", "Lab"]
icc_profiles_cmyk	array of **names of CMYK ICC profiles**; excluding ICC profiles in output intents; for example: "icc_profiles_cmyk" : ["PSO Coated v3", "US Web Coated SWOP"])
icc_profiles_lab	array of names of "Lab" ICC profiles; the "Lab" colour space, which strictly speaking is not an ICC based colur space, is still reported here as "Lab"; for example: "icc_profiles_lab" : ["Lab"]
pdfx_version	if present, the **PDF/X version** (e.g. "PDF/X-1a")
pdfx_oi_output_cond_id	if present, the value of the **PDF/X OutputIntentIdentifier** (e.g. "*FOGRA39*")

name	value
pdfx_oi_info	if present, the text in the **OutputIntent Info** field
pdfx_oi_icc_name	if present, the name of the **PDF/X OutputIntent profile** (e.g. "*PSO Coated v3*")
pdfa_version	if present, the **PDF/A version** (e.g. "*PDF/A-3u*")
pdfua_version	if present, the **PDF/UA version** (e.g. "*PDF/UA-1*")
pdfe_version	if present, the **PDF/E version** (e.g. "*PDF/E-2r*")
pdf_encrypted	whether the PDF is **encrypted**; possible values: *0* (not encrypted) and 1 (encrypted)

Sub-structure for fonts

```
"fonts" : [
  {
    "fontname" : "TimesNewRomanPS-BoldMT",
    "fonttype" : "Type1",
    "embedded" : 1,
    "subset"   : 1
  },
  {
    "fontname" : "MyriadPro-BoldItalic",
    "fonttype" : "Type0",
    "embedded" : 1,
    "subset"   : 1
  }
]
```

Click to copy

Sub-structure for errors_list/warnings_list/ infos_list

Each entry in each of the three lists contains a key value pair, where the key is the name of a check (as configured in the kfpx used), and the value is a number n/m/o/p/q/r/s/t/u that

reflects the number of occurrences the respective error/ warning/info was triggered.

```
"doc_errors_list" : [
   {
     "name of check that has triggered an error" : n
   },
   {
      "another name of a check that has triggered an error" : m
   },
   {
     "yet another name of a check that has triggered an error" : o
   }
]
"doc_warnings_list" : [
   {
     "name of check that has triggered a warning" : p
   },
   {
     "another name of a check that has triggered a warning" : q
   },
   {
     "yet another name of a check that has triggered a warning" : r
   }
]
"doc_infos_list" : [
   {
     "name of check that has triggered an info message" : s
   },
   {
     "another name of a check that has triggered an info message" : t
   },
   {
     "yet another name of a check that has triggered an info message" : u
   }
]
```

Click to copy

Example content for finish.json

```
{
```

```
    "verb" : "finish",
    "app_uuid" : "0a0ceba3-7ca9-421f-a973-8caae2950690",
    "timestamp" : "2016/10/31 20:08:27",
    "timestamp_hour" : 20,
    "timestamp_month" : 10,
    "timestamp_weekday" : 1,
/*  ... and all further entries defined for "launch.json" */


    "profile" : "P959c755539c8439e62c516c66a4a9097",
    "profile_name" : "Sheetfed offset (CMYK, RGB and spot colors) (GWG 2015)",
/*  ... and all further entries defined for "init.json" */


    "retcode" : "8",
    "duration" : "0:00:11:045",


    "doc_corrections" : 870,
    "doc_max_severity" : 3,
    "doc_messages" : 236,
    "doc_errors" : 160,
    "doc_errors_list" : [
       {
         "Font not embeddded" : 17
       },
       {
          "DeviceRGB used" : 28
       },
       {
          "TrimBox entry missing" : 1
       }
    ]
    "doc_warnings" : 76,
    "doc_warnings_list" : [
       {
         "Resolution less than 200ppi for continuous tone image" : 3
       },
       {
         "Page empty" : 2
       }
    ]
    "doc_infos" : 0,
```

```
"doc_infos_list" : [
   {
     "Uses spot color" : 61
   },
   {
     "Uses transpanrency" : 39
   }
]

"pdf_encrypted" : 0,

"num_images" : 70,

"num_fonts" : 2,

"fonts" : [
  {
    "fontname" : "TimesNewRomanPS-BoldMT",
    "fonttype" : "Type1",
    "embedded" : 1,
    "subset"   : 1
  },
  {
    "fontname" : "MyriadPro-BoldItalic",
    "fonttype" : "Type0",
    "embedded" : 1,
    "subset"   : 1
  }
],


"num_spotcolors" : 3,
"spotcolor_names" : [
  "Orange",
  "Purple",
  "Varnish"
],


"num_icc_profiles" : 3,
"icc_profiles_gray" : [
],
"icc_profiles_rgb" : [
```

```
      "sRGB",
      "eciRGB v2"
    ],
    "icc_profiles_cmyk" : [
       "PSO Coated v3"
    ],


    "pdfx_version" : "PDF/X-4",
    "pdfx_oi_output_cond_id" : "FOGRA39",
    "pdfx_oi_info" : "Prepared for ISO 12647-2:2013, coated sheet fed offset",
    "pdfx_oi_icc_name" : "PSO Coated v3",
    "pdfa_version" : "PDF/A-2b",
    "pdfua_version" : "",
    "pdfe_version: ""
}
```

Click to copy

Examples files as downloadable attachments:

launch.json

init.json

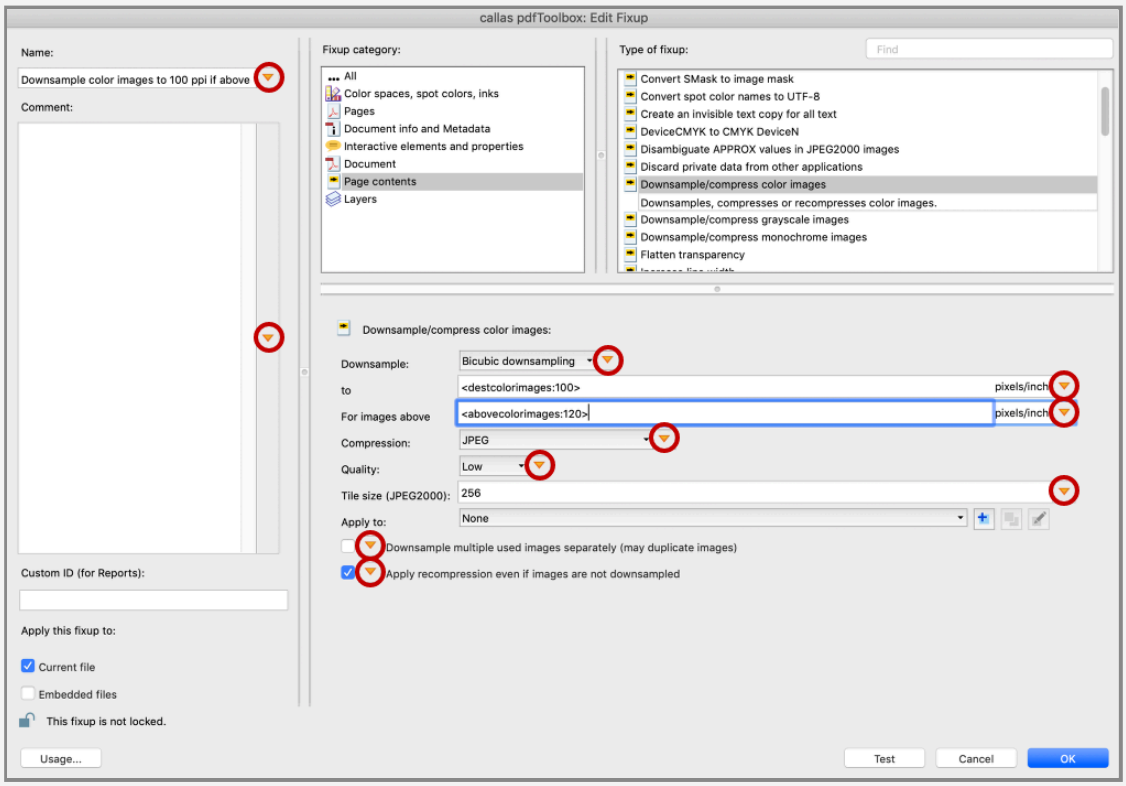
finish.json

17. Variables and JavaScript, Ask-at-run-time dialog

17.1 Simple variables

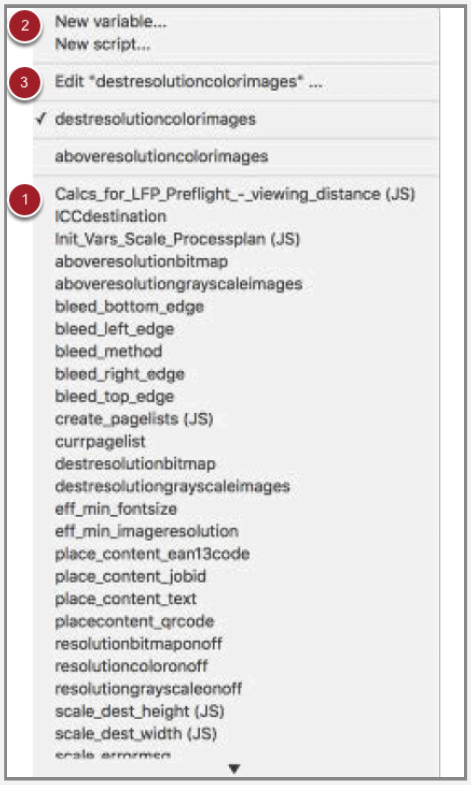
Variables can be defined in several places in pdfToolbox, e. g. in the editor of Profiles, Checks, Fixups or Process Plans. Variables may be assigned to virtually every control including the severity for a Check:

- Text input fields
- Checkboxes
- Pop Up fields
- Severities
- On/Off switch in order to enable/disable for example a Checks or Fixups in a Profile or Process Plan



Watch this short video to learn how you can create a preflight check that uses simple variables so that each PDF is (or can be) checked differently.

Assign a variable



When you click on a variable icon (orange triangle), you will see a list of all variables that are present in the system. Variables that are used in the current context (e.g. the current Profile) appear at the top. You may pick any of the existing variables (1), create a new one (2) or edit one that is already assigned (3).

Two types of variables

As you may have seen, there are two different types of variables:



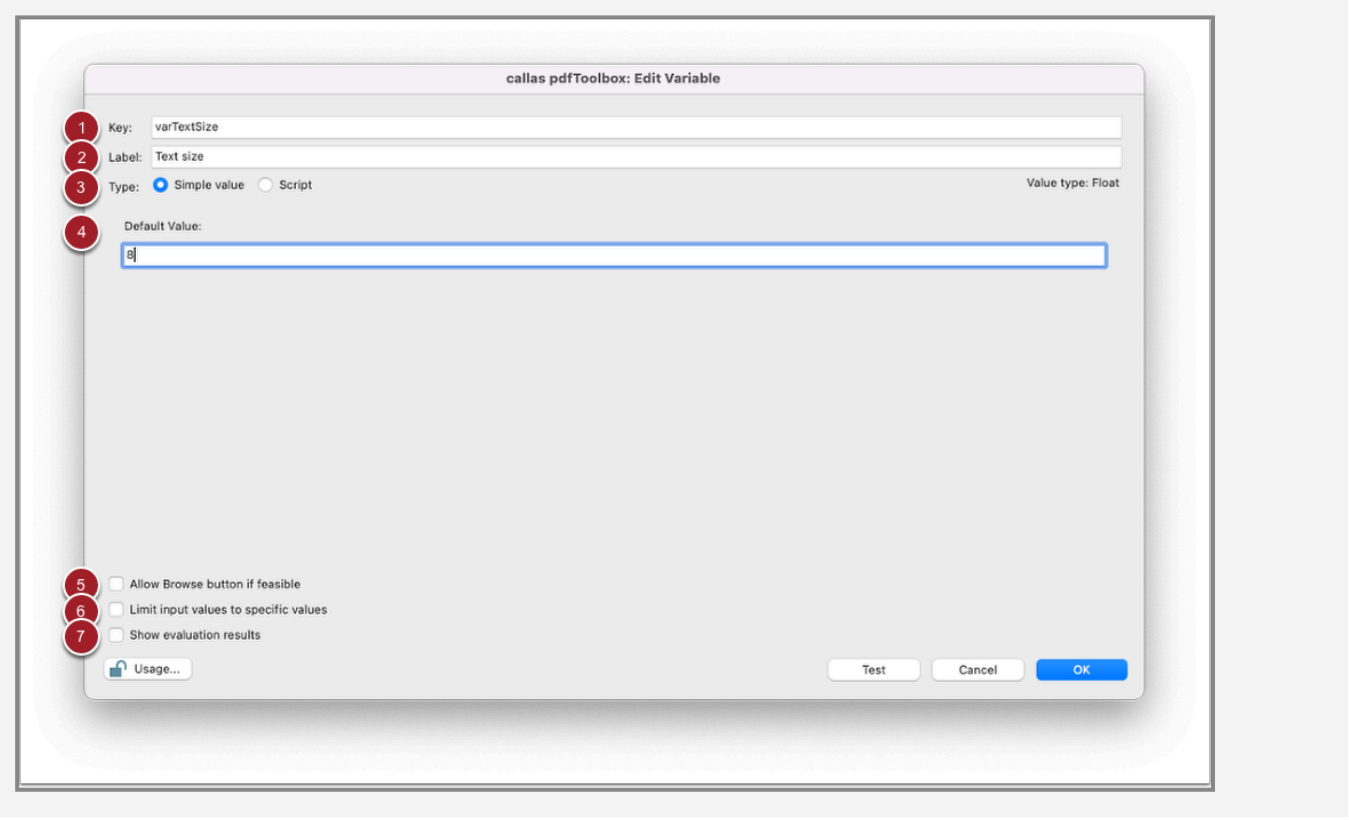
A simple variable can be inserted into a parameter of e. g. a Check to make it dynamic. When the Check is run in pdfTool-

box Desktop, the software asks the user which value to insert for the variable.

Script variables are defined in the form of a JavaScript; this allows deriving the value for a variable from other variables, from the metadata of the current PDF file, or from the results of a previous preflight Check. If you want to learn more about script variables then you can find everything in the next article: Variables using JavaScript: Overview

Variable Editor: creating a new variable

To define a variable pdfToolbox needs at least three pieces of information: key, label and default value.



1. **Key:** for internal use and use on the command line to address the variable (therefore the key should be unique).
2. **Label:** for use in the user interface, for example in the "Ask at runtime" dialog.
3. **Type:** Simple value (simple variable) or Script (JavaScript-based variable). Tip: on the right side the value type is always specified, which must be used for the default value.

The following types are available (Float, Sting, Boolean, Pop-up, Integer).

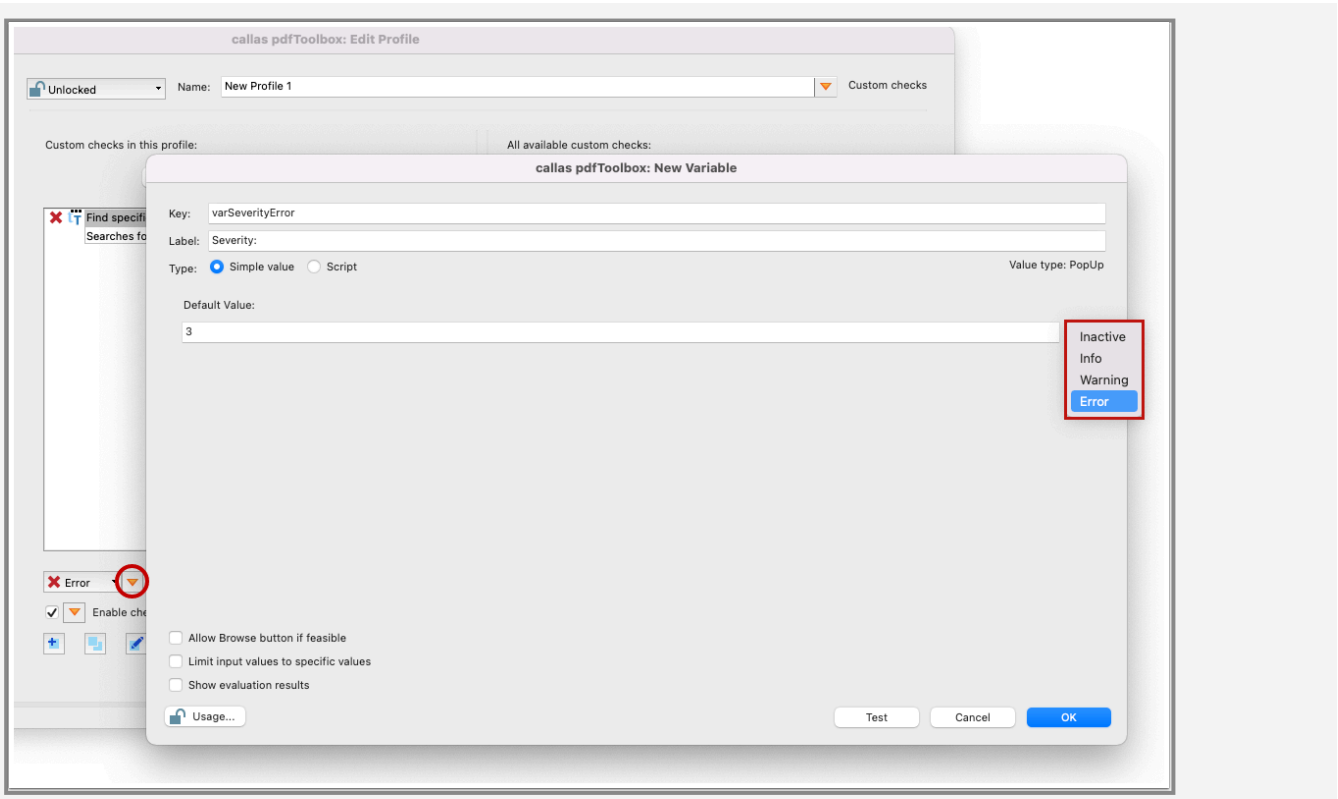
4. **Default value:** to be used unless a different value is provided at runtime.
5. **Allow Browser button in feasible:** determines whether a browse button is displayed in the Ask-at-runtime dialog when executed in pdfToolbox Desktop, allowing the user to select a file from the system (e.g. to load an ICC profile).
6. **Limit input values to specific values:** you may define restrictions (this is explained in a later step in this chapter).
7. **Show evaluation results:** if enabled, a console will be available for debugging.

Default values for Checkboxes and PopUps

If the parameter for which a variable is to be created is a Checkbox or a PopUp, only predefined values are valid for the default value:

A checkbox has the value type **Boolean**, which means that only two default values are valid: 0 (checkbox is disabled) or 1 (checkbox is enabled). Other values are not valid.

The value type **PopUp** has always an info button in the variable editor, which contains all valid values. The example below shows the default values for the severities of a Check. You can choose between 4 severities: Inactive, Info, Warning, Error. Depending on which severity is selected in the info dialog, the appropriate default value is added to the variable. In our example the severity should be "Error" as default, which has the default value 3.



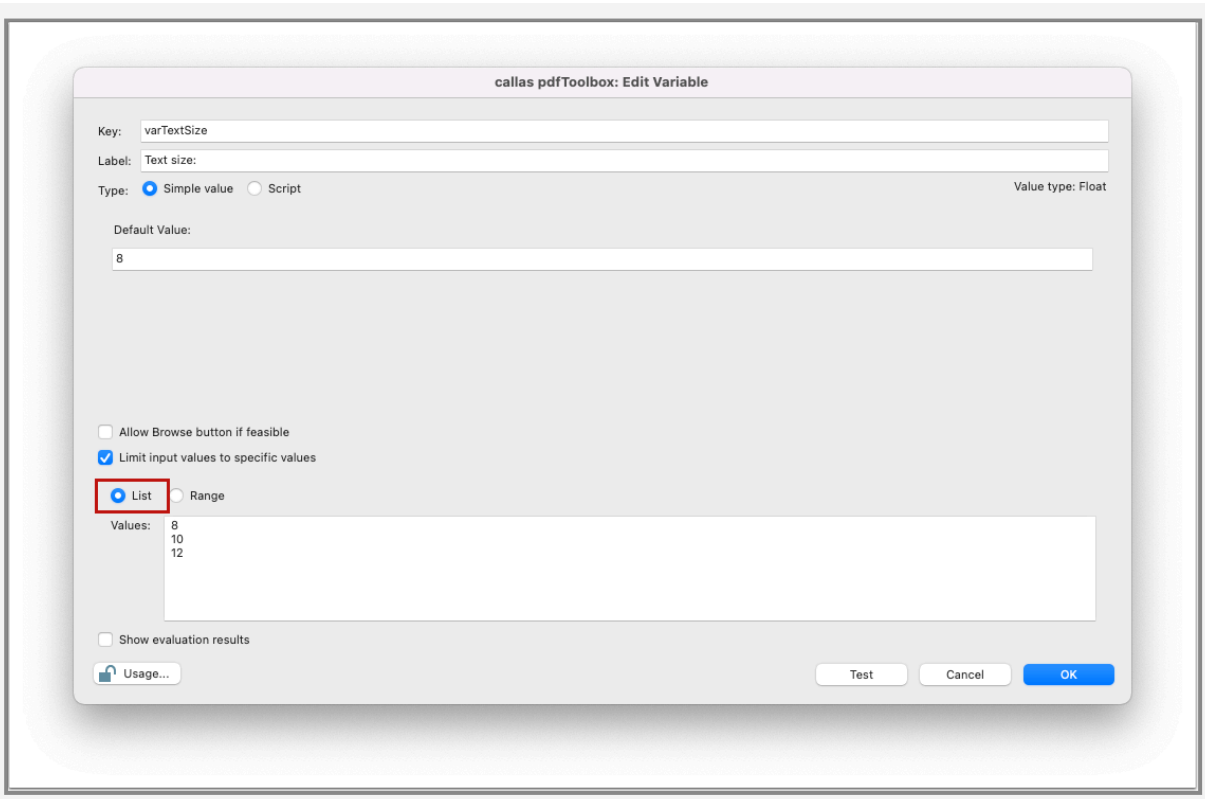
Limit input values to specific values

Constraints can be defined by using the "Limit input values to specific values" checkbox in the variable editor. There are two ways to limit the input values: the value provided for the variable can be limited to a range, or it can be limited to a list of predefined values.

Constraints - List

The entries that are defined under "Values" are used as a list, and will be shown as PopUp in the Ask-at-runtime dialog of pdfToolbox Desktop. That means that only the predefined values can be selected by profile execution.

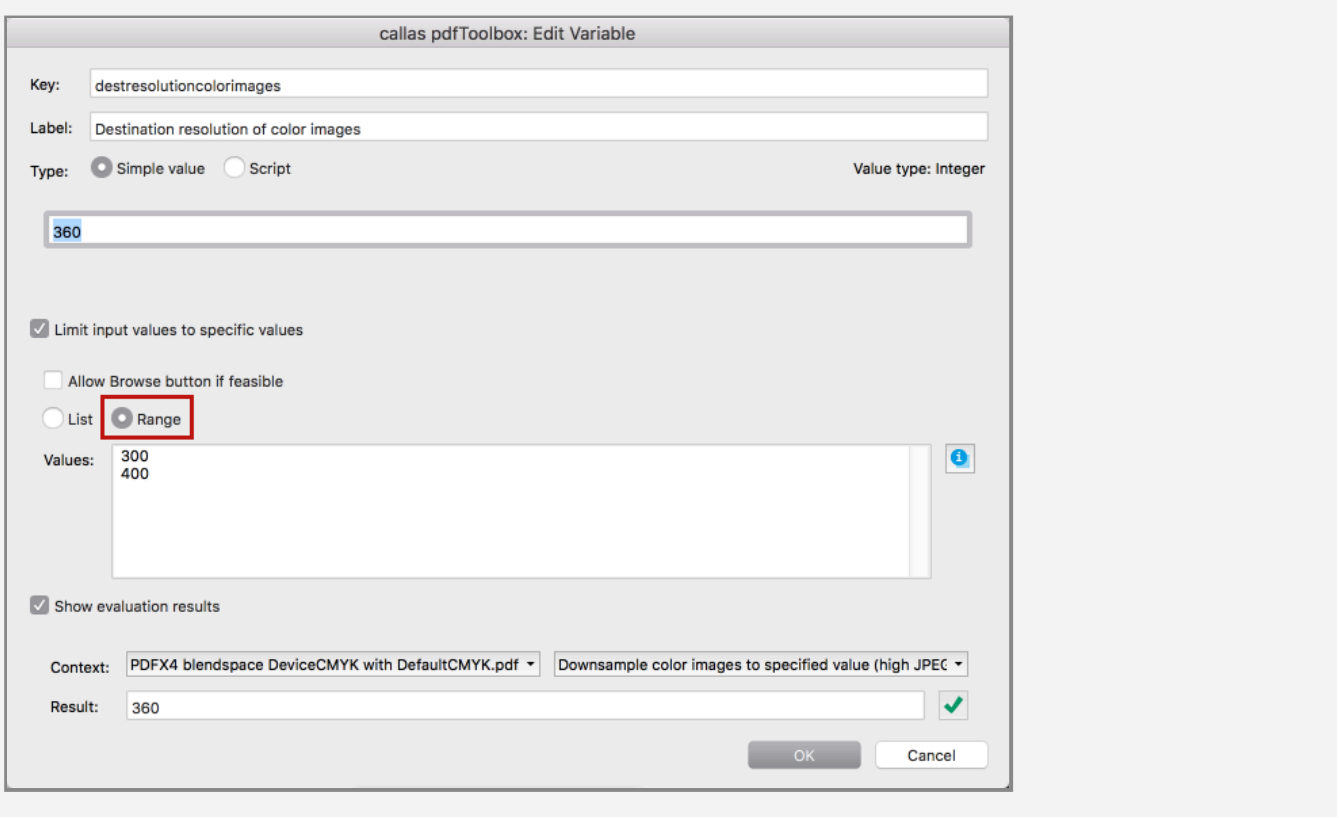
The values must be entered **one below the other** and the default value must be included in the list.



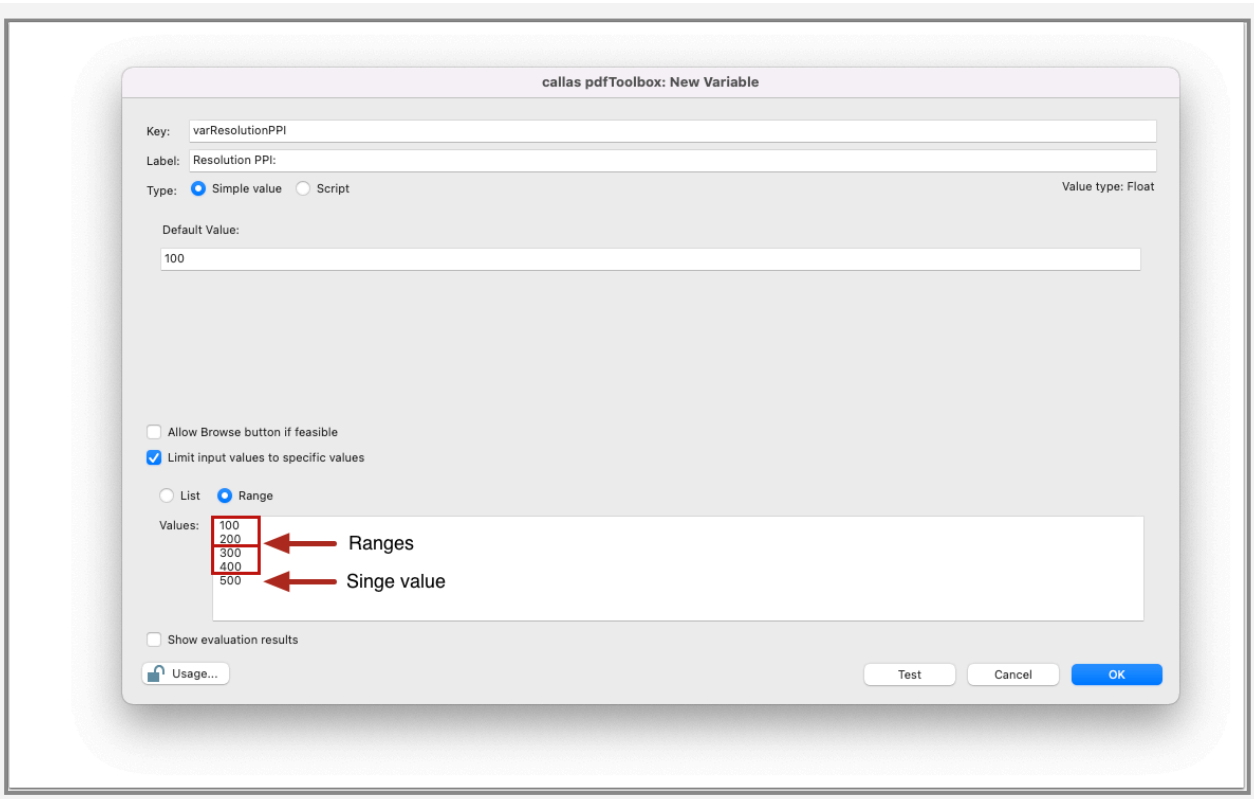
In this example, the values defined in the list are available for selection during profile execution in the "Ask at runtime" dialog, i.e. 8, 10 and 12.

Constraints - Range

When you are using the Range option, two values will define a range, so it only accepts a number between these two values. In this example, all values between 300 and 400 can be specified in the Ask-at-runtime dialog. Values outside this range are not allowed.

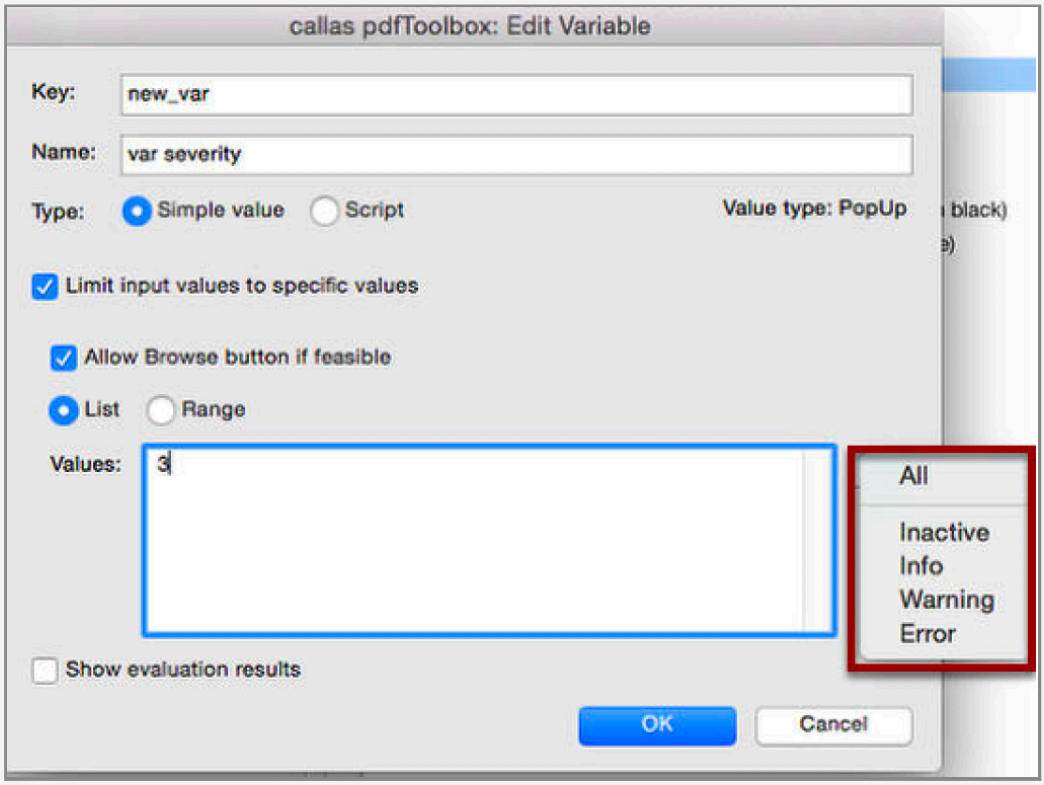


Special feature: multiple ranges can be specified as well as a single value. As soon as two values are added, they form another range. If a single value is added, it is permitted as a single value in addition to the range. In the example below, 5 values100 have been specified: the first two (100, 200) form the first range, the third and fourth values (300, 400) form the second range, and the last value (500) is considered a single value.



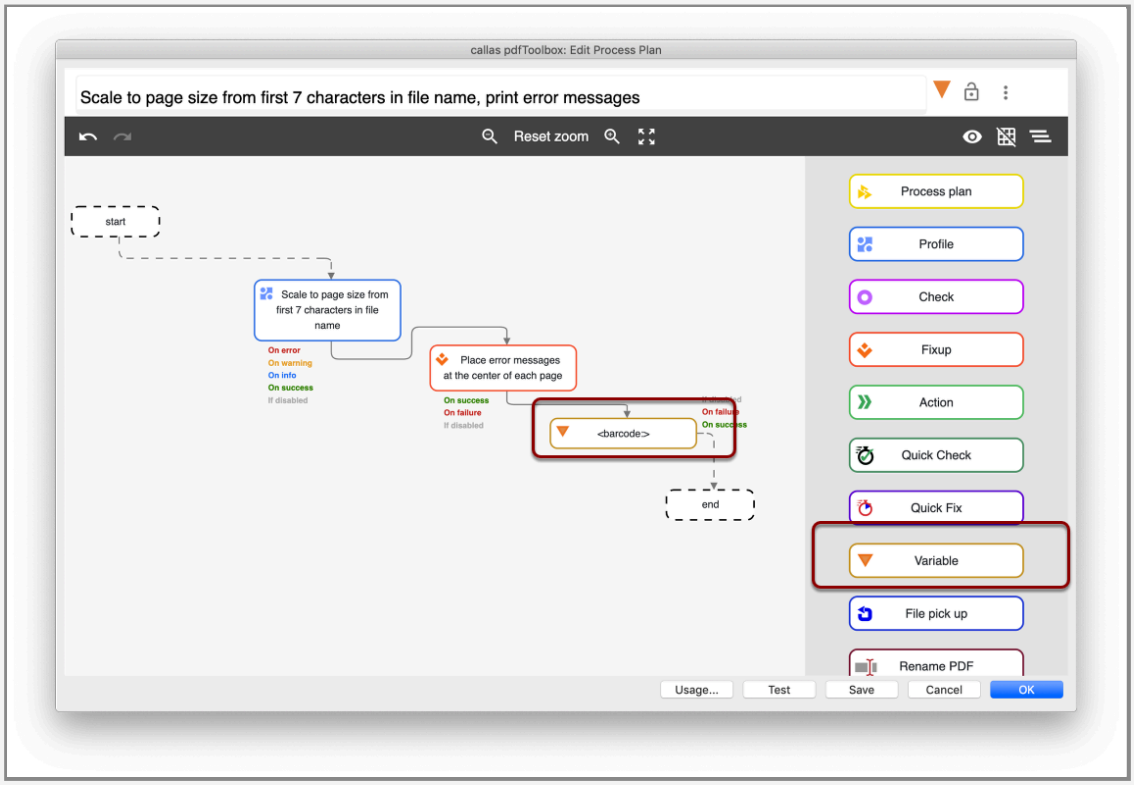
Constraints for PopUp fields

You may again use the info button in order to pick possible values for the PopUp. In this example this is done for a severity, but it works in the same way for other Pop up fields.



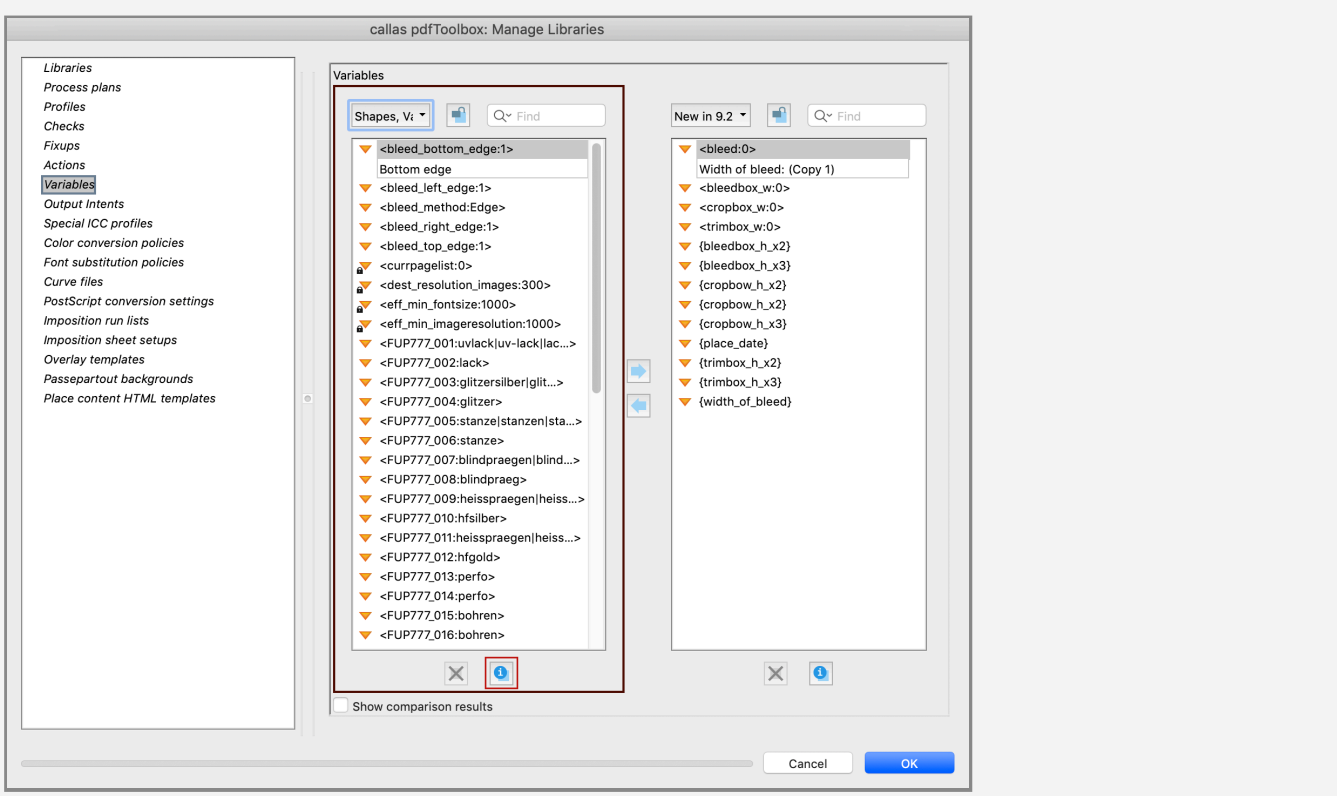
Variables in Processplans

It is possible to define a variable as a step in a Process Plan. This will work similar to a variable on Profile level and is only useful and effective for script variables.



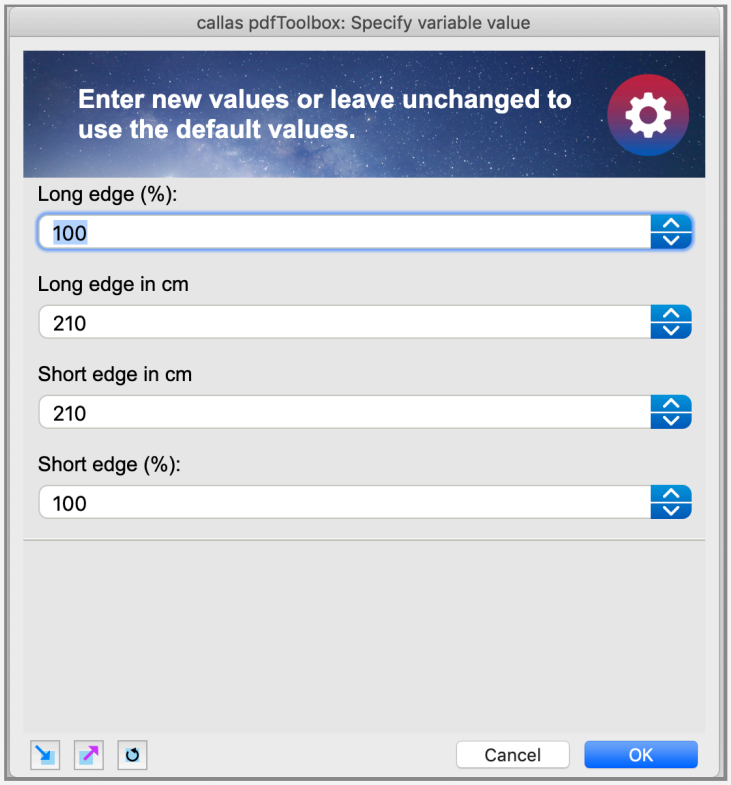
Managing and deleting a variable

Deleting a variable is currently only possible in the Library Manager (Library drop down> Manage Libraries) and only if the variable is not used. Via the info button you can see in which Profiles the variable is used.



The "Ask-at-runtime" dialog in pdfToolbox Desktop

In pdfToolbox Desktop a dialog shows up, when a Profile/ Check/Fixup/Process Plan is executed that has one or more variables – the so called "Ask-at-runtime" dialog. Further information about the Ask-at-runtime dialog can be found here: Ask-at-runtime Dialog: Introduction.

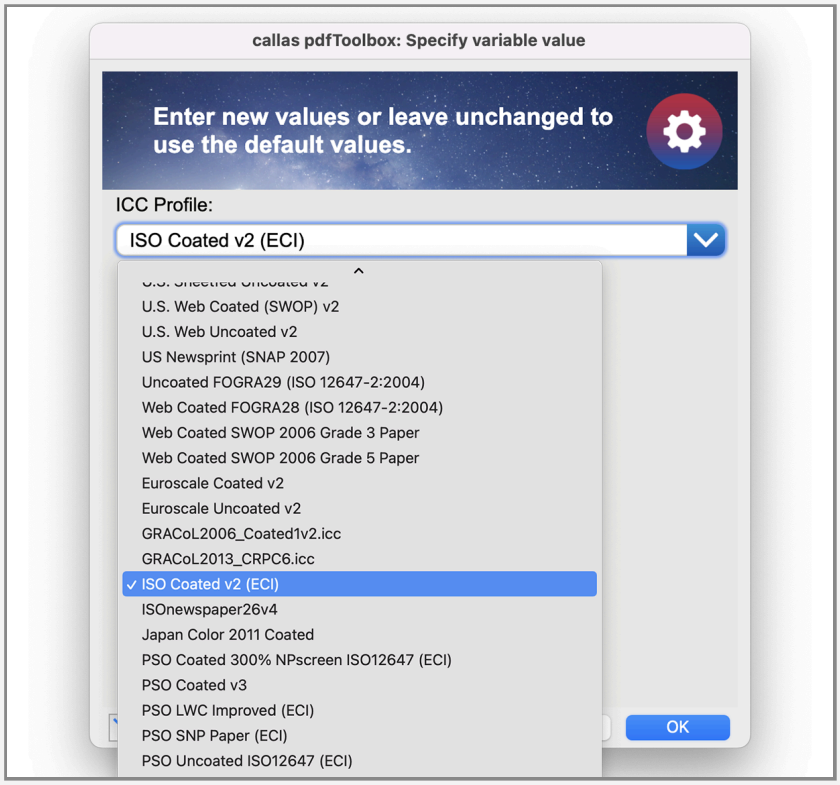

callas pdfToolbox: Specify variable value
Enter new values or leave unchanged to use the default values.
Long edge (%):
100
Long edge in cm
210
Short edge in cm
210
Short edge (%):
100
Cancel
OK

17.2 Using Variables for resources

It is possible to set a Variable for the resources in a Profile, Fixup, Action, etc. (for example for an ICC profile or an Output Intent). The handling is different between pdfToolbox Desktop and the CLI version.

Difference between pdfToolbox Desktop and pdfToolbox CLI

If a Variable is used for resources in **pdfToolbox Desktop** (in this example an ICC Profile), the names of all possible resources are listed in a drop-down menu in the Ask-at-runtime dialog during Profile execution. Simply select the desired ICC Profile and the resource is immediately available.



If you want to execute the Profile on the **CLI**, you have to use the absolute path to the resource because the resource is not part of the Profile.

In order to set a Variable for a resource the following call may be used:

```
./pdfToolbox --setvariable=<myVAR>:<absolut path to resource> sample.kfpx sample.
pdf
```

Click to copy

You can also use `--setvariablepath`. This works the same as `--setvariable` but checks for the existence of the referenced file or folder.

```
./pdfToolbox --setvariablepath=<myVAR>:<absolut path to resource> sample.kfpx sam-
ple.pdf
```

Click to copy

Example

Here is an example of how to assign the ICC profile "ISO Coated v2 (ECI)" to the Variable "varDestination" on the CLI.

```
./pdfToolbox --setvariable=varDestination:/Applications/callas\ pdfToolbox\ Serv-
er\ 14/cli/etc/ICC\ profiles/ISOcoated_v2_eci.icc /Users/Username/Desktop/Convert\
colors.kfpx /Users/Username/Desktop/Sample.pdf
```

Click to copy



If only the name of the resource is specified on the CLI, e.g. `--setvariable=varDestination:"ISO Coated v2 (ECI)"`, an error message would be displayed because pdfToolbox does not have access to the resource.

17.3 Variables using JavaScript: Overview

Where can JavaScript variables be used

Script variables can be used wherever Simple variables can be used:

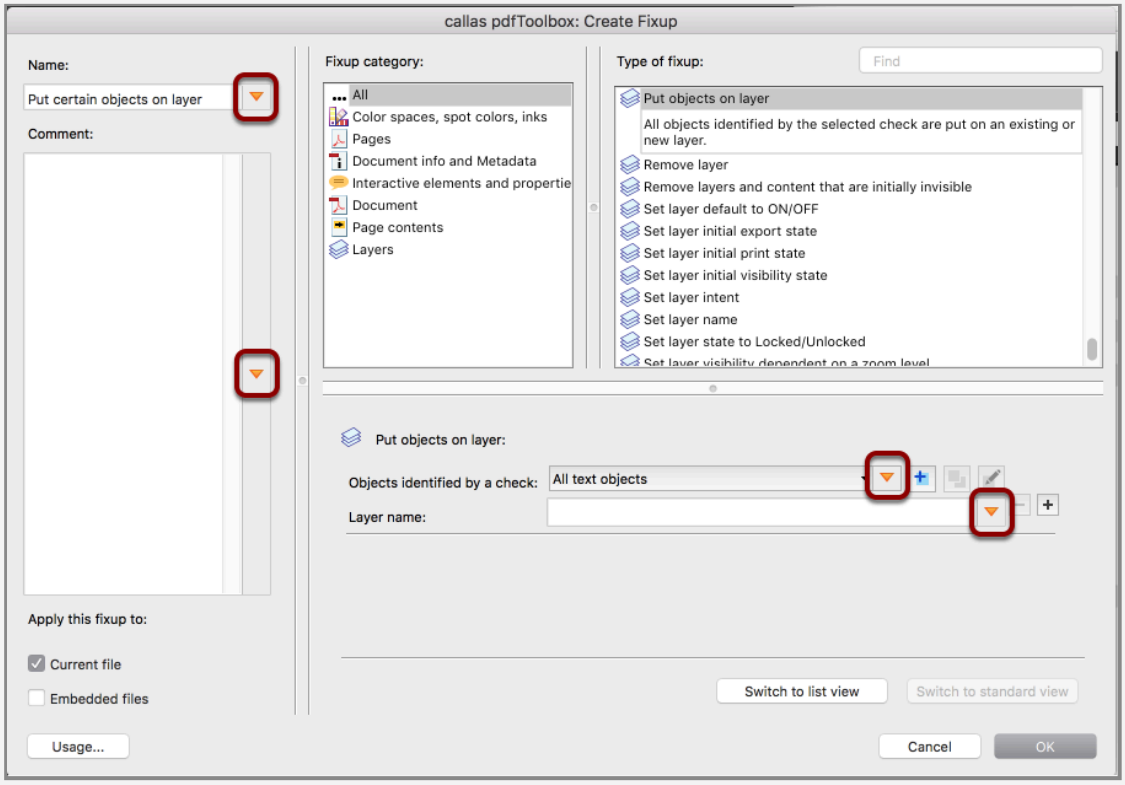
- In Checks or Fixups for text input fields, pop ups, check-boxes
- Severities of checks
- On/Off variables for Checks and Fixups

In addition, it is possible to use Script variables (but not Simple variables):

- in a Profile as Profile script
- as a Process Plan step

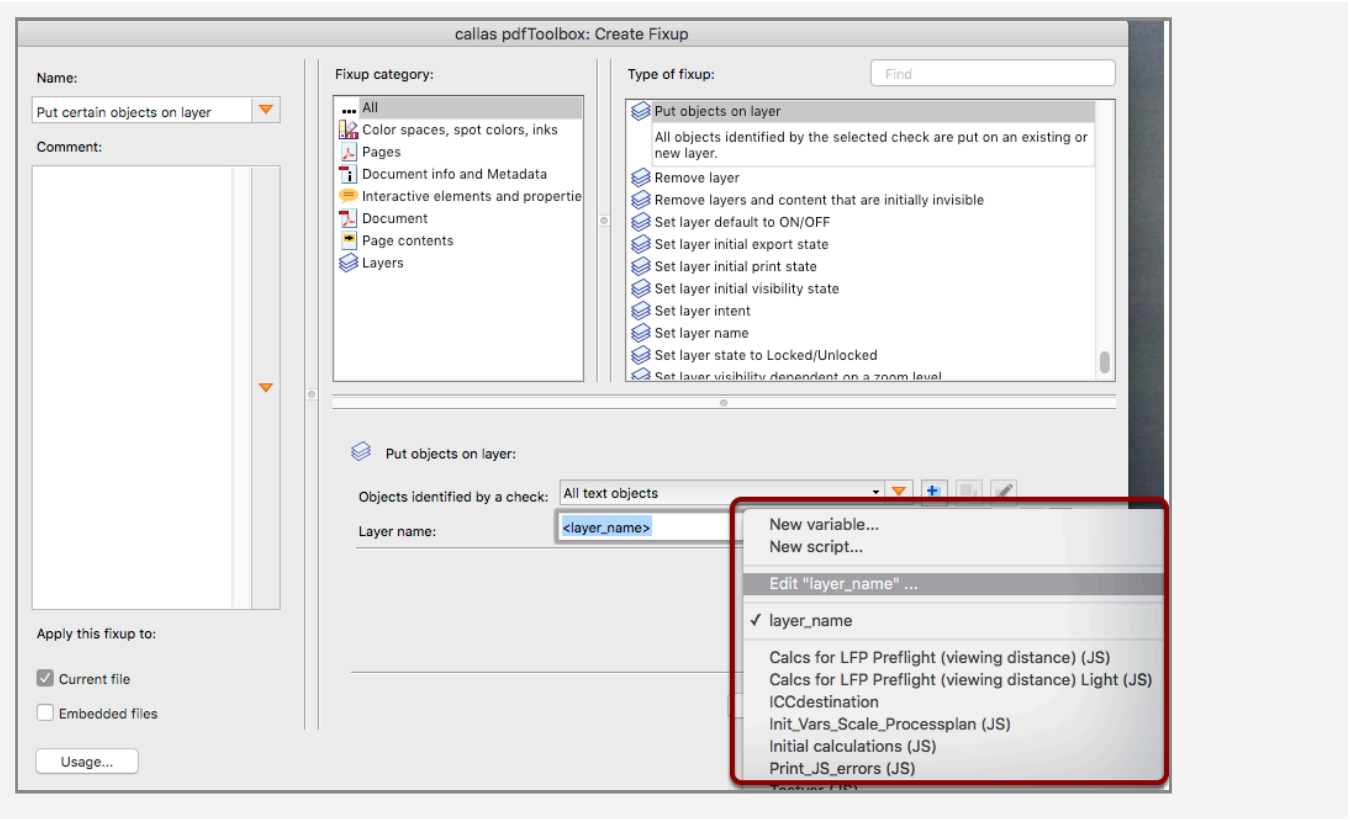
In all places where Simple or Script variables can be used, the variable editor allows you to switch between both by means of a radio button. After a variable has been saved as Simple variable it is possible at any time to convert it into a Script variable here. However, it is not possible to convert a Script variable into a Simple variable. The reason is that this could potentially lead to problems when the same variable would be used in a place where only a Script variable is allowed.

Assigning a variable to a pdfToolbox Desktop control



Wherever you see the variable icon in pdfToolbox Desktop you can click on it in order to assign a variable from a list of all variable keys that are defined in the current Library to the respective control.

Creating or modifying a JavaScript variable



You can create a new variable (either as Simple or as Script variable) and assign it; or you open the variable editor in order to modify a variable that has already been assigned.

The list of variables in the pop up shows those variables that are already used in the current context (Profile, Check, Fixup) first and then all variables in the current Library. Script variable keys are followed by "(JS)" to indicate that these are JavaScript variables. After assigning a variable to a pdfToolbox Desktop control the variable key is displayed in the respective field (for text input fields or pop ups) or next to it (for checkboxes). Simple variables are displayed as <Simple variable>, Script variables as {Script variable}.

In order to un-assign a variable from a control you simply have to remove it from a text input field, to pick any other value in a pop up or to check/uncheck a checkbox.

Creating or modifying a JavaScript variable:

Important differences to pdfToolbox versions earlier than version 9

In pdfToolbox versions earlier than version 9 it was possible to copy a variable out of a pdfToolbox Desktop control and insert it into another control in order to assign it to both controls. This is not possible in pdfToolbox 9. You have to select the variable key from the variable pop up in the second or any further control.

From pdfToolbox 9 on it is no more possible to make two variable occurrences using the same value simply by using the same variable name ("key"). Variables are only then the same if any additional occurrence is selected from the variable pop up in pdfToolbox Desktop. Otherwise two variables using the same key would be present which would at least be confusing when evaluated.

But: It would be difficult to resolve such conflicts when a Profile is imported as kfpx file, if the imported Profile uses the same variable key as a variable that is already present in the current Library. Therefore in such cases internally a variable merge process takes place that merges all variables that are defined in the very same way (key, default value and label) into a single variable.

Defining a variable in a script

If you want to define a variable in a script that is not used in any pdfToolbox Desktop control you may do so by writing at the top of your script:

```
app.requires("myvar")
```

myvar will then be created and show up in the Ask at Runtime dialogue or in --listvariables on command line. If you also want to set a default value and a display name (label) you can write:

```
app.requires("myvar",100,"Input a value for myvar")
```

Setting the value for a Script variable in it's own

script

A Script variable is populated with the value that is the result of the last statement in the script. So, if a Script variable would end with a statement like "pdfToolbox" it would have this string as its value, independent from what code has been executed beforehand. A return statement as in a JavaScript function is neither required nor would it have any effect.

Setting the value for another Script variable with app.vars

It is possible to set a value for another variable in JavaScript code by means of an app.vars.<variable key> statement. The app.vars object is a pdfToolbox object that is available throughout the context (Process Plan, Profile, single Check, single Fixup) in which processing takes place. It allows you to store and retrieve variables within this context:

```
app.vars.myvar = "pdfToolbox";
```

or:

```
localvar = app.vars.myvar;
```

are valid statements. The first statement would create the variable "myvar" if not already present in app.vars. You may e.g. use app.vars to set a value for a variable on Profile level, which is then used in a Fixup in the Profile.

In order to set a value for a Simple variable you can use app.vars.<variable key>. A list of all variables that are present in the current Library can be displayed in the Script editor by using [<command>-2].

Setting a value for a variable via JavaScript code should only take place on Profile level or as a "Variable" step of a Process Plan. The reason is, that it is not defined in which order scripts on "lower levels" (Checks, Fixups, Severities, On/Off) are executed during runtime and therefore the result of e.g. one Fixup modifying a variable in another Fixup is undefined.

In pdfToolbox 10.0.465 it became possible to delete variables in app.vars via

```
delete app.vars.myvar;
```

You also have access to variables in app.vars in the "Place content on page" fixup. This fixup allows you to place content defined in HTML templates that may internally use JavaScript. If you want to read, write or create a variable in app.vars from that JavaScript you have access to all of them in the array

```
cals_doc_info.document.variables
```

Using variables that are defined elsewhere

In order to use the value of a variable in JavaScript it can be accessed using the app.vars object with the key of the variable as already described above: app.vars.<variable key>. If the other variable is defined in a Script variable it has to be defined using app.vars there as well. If the other variable is a Simple variable it is always present in the app.vars object.

When retrieving variable values from app.vars it is important to know that all variables are stored there as strings. With simple value types you will most probably not even notice this, because a string is automatically converted if necessary and possible, e.g. into a number. However, if you are working with more complex variable types like with arrays or objects there will obviously be differences and you might have to work around this limitation.

In complex profiles - actually when a Script variable is used in another Script variable - app.requires("<variable key>") has to be defined at the top of the referencing Script, in addition to the actual reference with app.vars. This is required in order to make sure that the referenced variable is evaluated before the referencing variable is calculated. So, it is good practice to at the top of each Script, list all variables which are not defined in the Script itself in app.requires entries.

Profile level scripts versus Check/Fixup level scripts

When you "design" a profile with JavaScript based calculations you have to decide whether you want to put the "intelligence" (the calculations) into Fixups and Checks that actual-

ly apply things to the PDF or into a Profile level script and set values for the variables that are then used in Fixups and Checks from there.

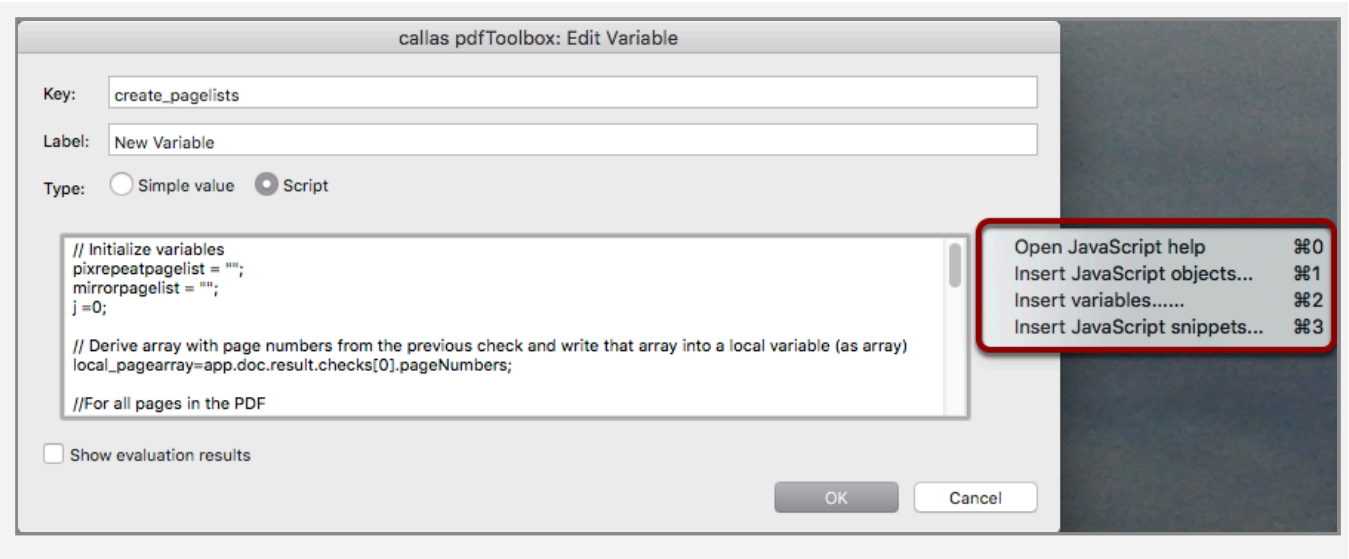
Example: Downsample images in pdfToolbox usually requires to set up three fixups: for color images, grayscale images and for bitmap images. Each of the fixups has two input fields that you may want to make variable: The destination resolution and the minimum resolution for an image to be downsampled. Assume that you want to downsample color and grayscale images to the same resolution. Images should be downsampled if the original image resolution is 1.5 times as high as the destination resolution. Destination resolution for bitmap images should be 3 times as high as for color images, with the same relative minimum resolution (effectively 4.5 times color images' minimum resolution). You may now either make the destination resolution for color images a Simple variable, e.g. "dest_col_res" and make any of the other 5 variables a Script variable that uses dest_col_res and calculates the actual value. Or you set up a Profile level script, do all the calculations there and put the results into a bunch of Simple variables that you assign to each of the 6 variable input fields. (You will have to use app.vars in order to use variables throughout the Profile and in the second case you would use app.requires to define a variable for the destination image resolution in the Profile script.)

Each of the two approaches has advantages:

- If you put the intelligence into Fixups and Checks it is easier to make it possible to use them as Single Fixups or Checks, independent from the Profile.
- If you put the intelligence into the Profile it is usually easier to see what a profile is actually doing and - even more important - to maintain it in the future.

As a result and a rule of thumb it can be said, that it usually makes sense to put as much intelligence into the Profile level script. The more complex a Profile is, the more important is it to follow this approach.

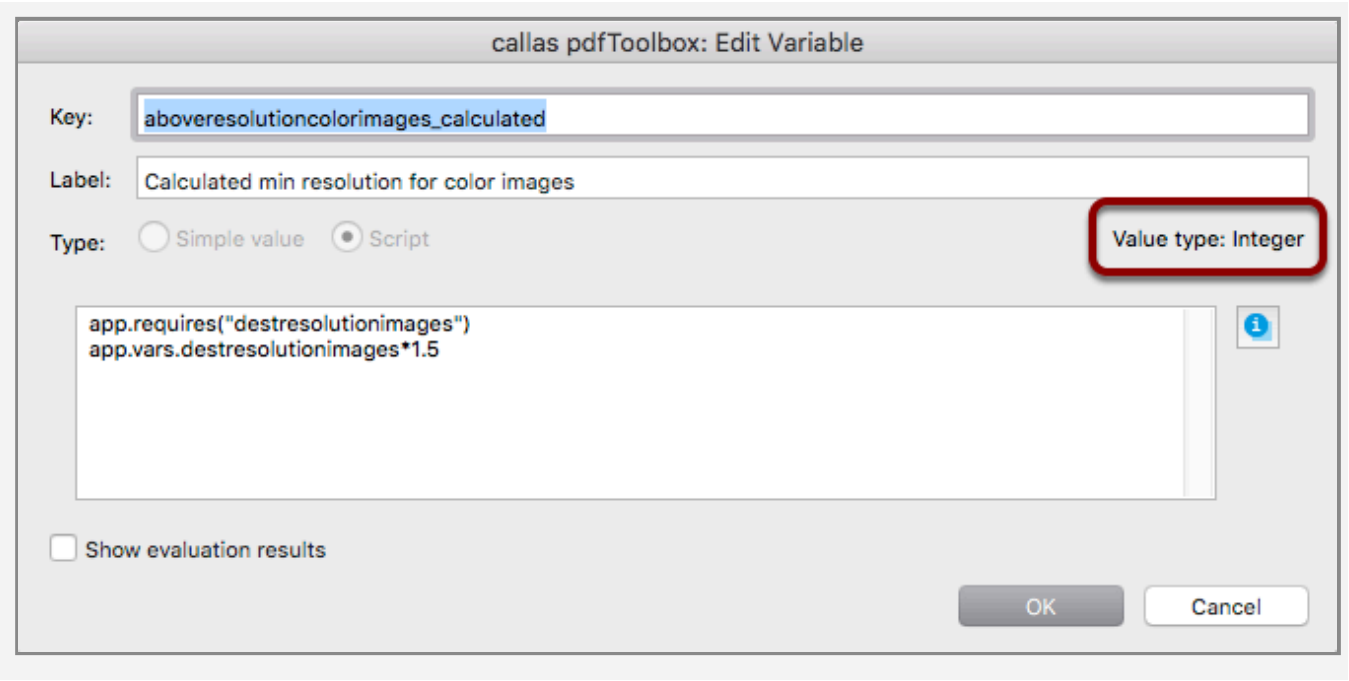
The Script editor: User interface elements: Help



When the variable editor is switched into Script "mode", you can find help with the info button on the upper right side of the Script input field. You will find more information if you click into the Script input field first. This gives you access to

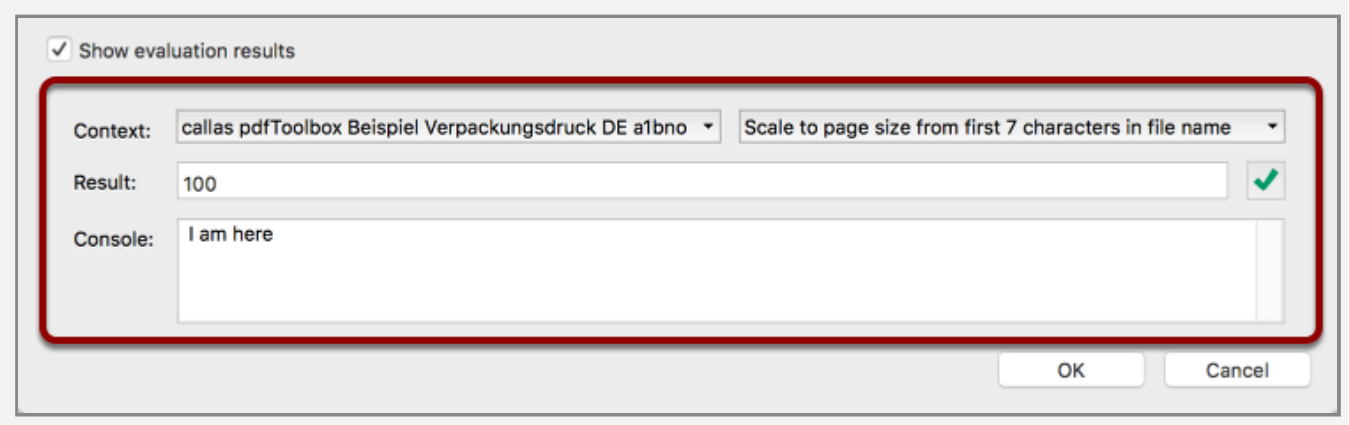
- a general help text (this text) [<command>-0],
- a list of all pdfToolbox specific JavaScript objects and methods [<command>-1],
- a list of all variables that are present in the current Library [<command>-2]
- useful code snippets [<command>-3].

The Script editor: User interface elements: Value type



Above of the info button you see the value type of the pdfToolbox control to which the variable is currently assigned. This information is useful to know what type of result is expected from your script.

The Script editor: User interface elements: Show evaluation results



Below the text input field you can switch "Show evaluation results" on, which will help you to find out what the result of your JavaScript code is.

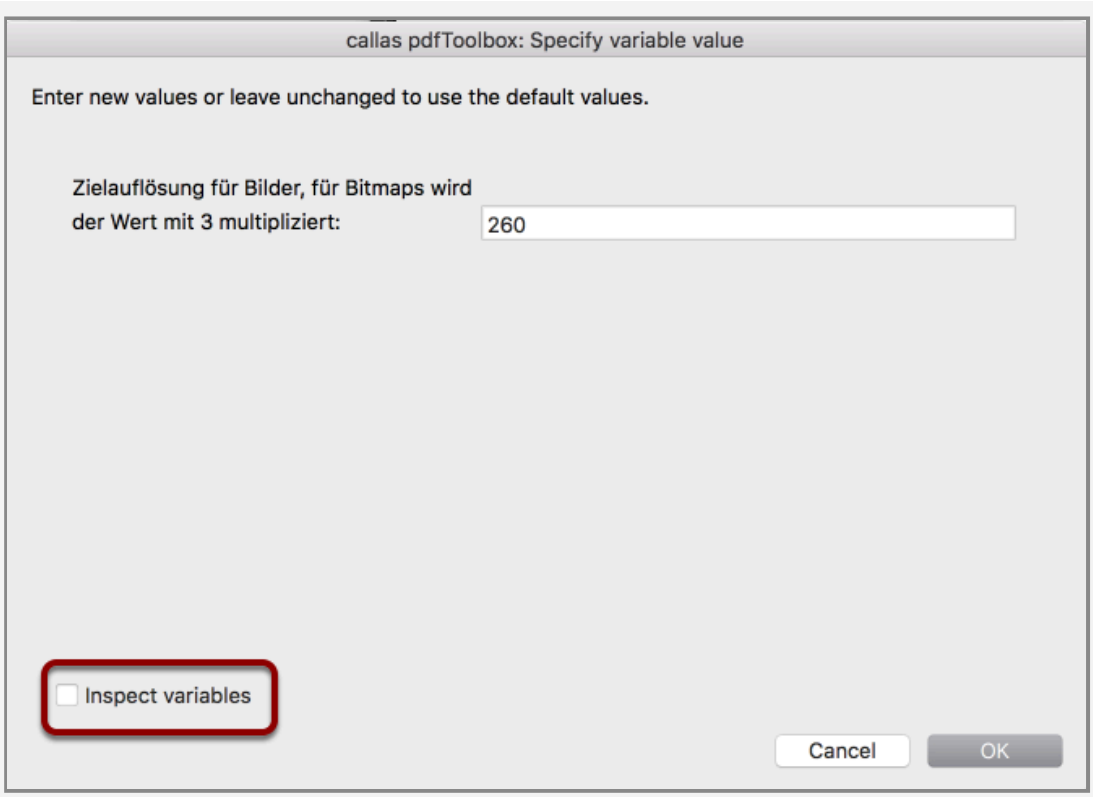
It is important to remember that the result of JavaScript code used in a variable is the result of the last statement. The only exception is if invalid code is used, e.g. if a closing parenthesis is missing, in which case you will see a "Syntax error" with an explanation.

In order to modify evaluation you can simulate how a JavaScript works different when used in a different "Context": You may either load a PDF (it will not open in pdfToolbox) to simulate how your script works on that PDF, e.g. when you are using the PDF path inside of your script. Or you switch between evaluation only for the script (in which case values that are set via other variables in your context are not evaluated) or evaluation within the context. All this information is helpful for debugging your scripts.

A button at the right hand side indicates whether the result of the script works in the current pdfToolbox control. If the result is "pdfToolbox" and you are using the variable for a text input field you will see a green checkmark. However, if the variable is used for an integer number field you will see a red error cross. When you click on it you will read: ""pdfToolbox" cannot be converted to integer".

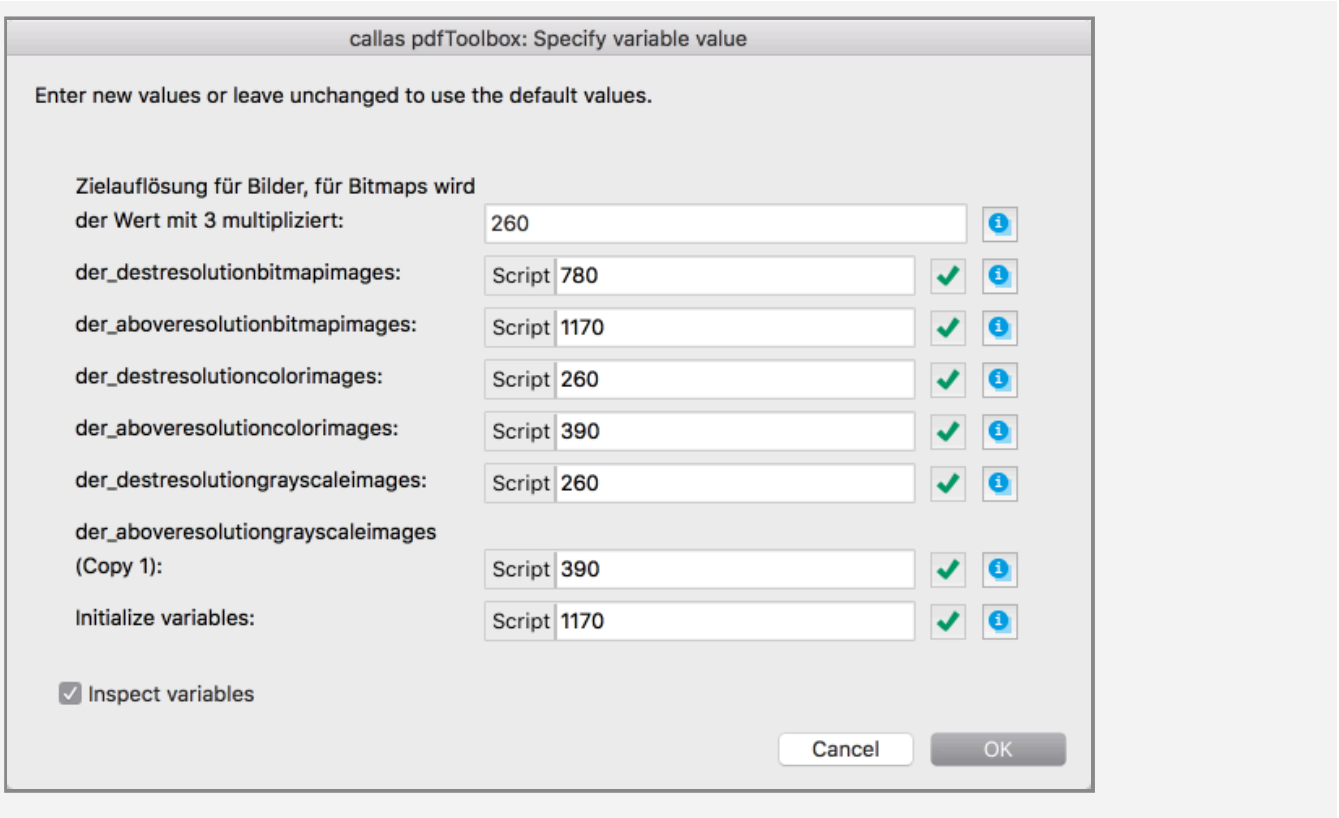
The console window displays information that the JavaScript sends to it. It can be used with console.log.

Inspecting the variable structure in the Ask at Runtime dialogue in pdfToolbox Desktop: Activating the "debug view"



When you run a Process Plan, Profile, Check or Fixup in pdfToolbox Desktop that uses variables you will see the "Ask at Runtime" that allows for updating variable values. A checkbox at the bottom of the dialogue allows the user to "Inspect variables".

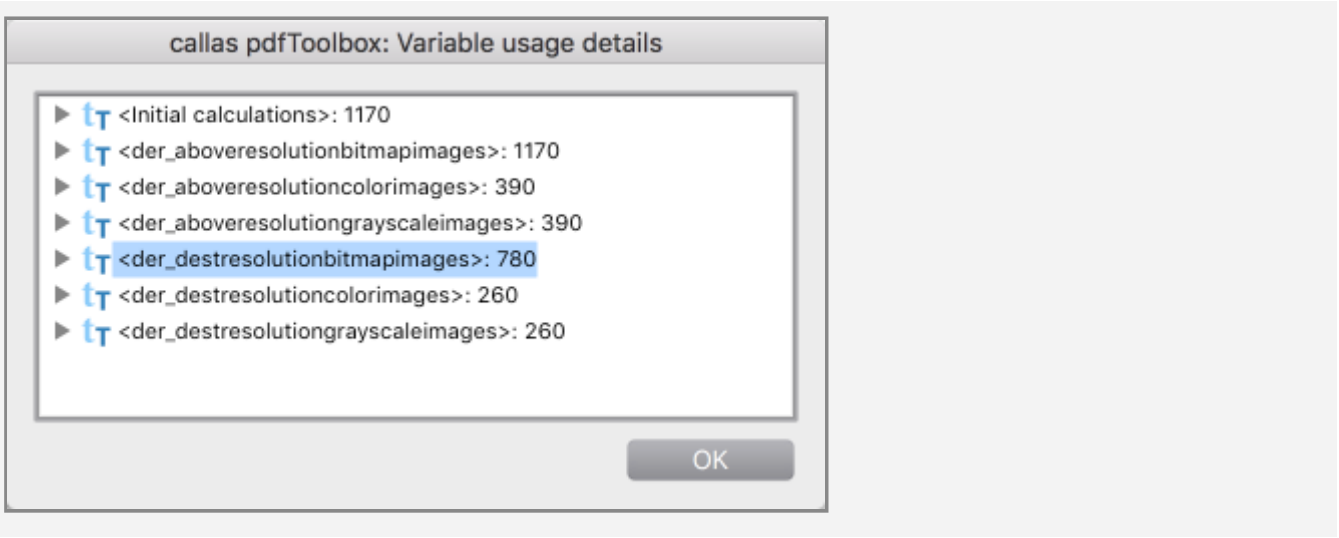
Inspecting the variable structure in the Ask at Runtime dialogue in pdfToolbox Desktop: The "debug view"



If activated all calculated variables are displayed, not only those ones that allow for user input. A "Script" indicator shows values that cannot be modified in this dialogue because they are already calculated in the scripts. A button behind the Script variable fields indicates whether there is a type conflict, e.g. if the variable is used for a number field but the value is a string that cannot be converted to a number. In that case you will see a red cross. You may click on any of those red cross buttons in order to see details for the problem.

The Ask at Runtime dialogue will only appear if a Profile/Check/Fixup has at least one variable that does not already have a value. If you want to enable the debug view in a Profile/Check/Fixup in which all variables are set by means of scripts, you will have to add at least one additional variable, e.g. a Simple variable for that purpose.

Inspecting the variable structure in the Ask at Runtime dialogue in pdfToolbox Desktop: The "debug view" info button



Next to each variable you will see an info button. Clicking on this button allows for accessing information that is useful for debugging purposes. A list of all variables that are defined in the current context is displayed. (The content of the info window is actually the same for each of the info buttons, the only difference is that the control for the respective variable is opened by default.) Each of these variables is followed by the result that has been calculated for the respective variable. If you open the triangle for a variable you see an entry "Variables" that shows details about how that variable has been defined. Below are all contexts listed in which the respective variable is used.

pdfToolbox specific JavaScript objects and methods

pdfToolbox provides a number of objects and methods that can be accessed in JavaScript variables. This includes information about the PDF, like its name or file path, the metadata in the PDF and even results from a previous Check or Profile. You can display a full list of all objects and methods by

using [<command>-1] when in the Script editor, select any of the entries and insert them into your script.

A complete list of these objects and methods can also be found in the chapter "Variables using JavaScript: pdfToolbox objects and methods".

17.4 JavaScript in pdfToolbox: Custom data objects and methods

Inside pdfToolbox, JavaScript functionality for use in JavaScript Variables and in the form of a "Profile JavaScript" is provided as follows:

- The JavaScript engine in pdfToolbox is based on Google's V8 JavaScript engine (see https://developers.google.com/v8/ for more information).
- pdfToolbox (through the underlying V8 engine) supports the complete set of JavaScript features as defined in ECMAScript is specified in ECMA-262, 5th edition (see http://www.ecma-international.org/publications/standards/Ecma-262-arch.htm)
- pdfToolbox provides access to custom pdfToolbox controlled data objects, for example data objects representing metadata, file name, page sizes, and so on for the current PDF
- pdfToolbox provides custom methods, for example for reading a file or parsing XML using XPath
- pdfToolbox also supports storing data in the `app.vars` data object which can be accessed for reading and writing throughout execution of a Process Plan, Profile, Check or Fixup
- The pdfToolbox JavaScript engine comes with a powerful runtime evaluation architecture, that ensures that JavaScripts present in several JavaScript variables possibly relying on each other's execution do work in a predictable manner without the user having to define execution order.
- The pdfToolbox JavaScript engine supports only very limited access to the outside world (i.e. accessing web services, data bases or arbitrary services through whatever protocol are mostly not supported); one exception is a possibility to read data from the local file system.
- pdfToolbox currently does not offer the possibility to reference JavaScript files, as is often used to incorporated JavaScript modules; where such modules are to be incorporated anyway, the JavaScript code for such modules must be included in the JavaScript code for a given script variable (or in the JavaScript for a "Profile JavaScript").

Details about the Google V8 version used in pdfToolbox

- pdfToolbox 9.0 through 10.0: **v8 version 5.2.0**
  ECMA-262, 5th edition
- pdfToolbox 11.0 through 13.0: **v8 version 6.9.7**
  ECMA-262, 9th edition (June 2018)
- pdfToolbox 13.1 onward: **v8 version 9.7.106**
  ECMA-262, 12th edition (June 2021)

Note: While Google V8 claims complete support of JavaScript as defined in ECMAScript 262, this is not 100% correct. For example, the `import` method is not supported in V8, and has to be implemented by the tool or system inside which V8 runs. pdfToolbox will support the `import` method in a future version.

pdfToolbox specific internal data objects

pdfToolbox defines custom objects: 'app', 'console', 'File' and 'XML'.

- The **app object** contains information about
  - **app.doc** the current document (read only)
  - **app.vars** a storage for variables (read/write)
  - **app.http** access to external resources via REST (read only)
  - **app.context** allows customization of the ask at runtime dialog
  - **app.env** information about the environment (read only)
  - **app.name** and **app.version** information about the instance of pdfToolbox (read only)
- The **console object** provides a log function that currently only works in the script editor under "Show evaluation results"
- The **File object** enables reading of files from the file system
- The **XML object** allows parsing of XML data and retrieving values using Path expressions.

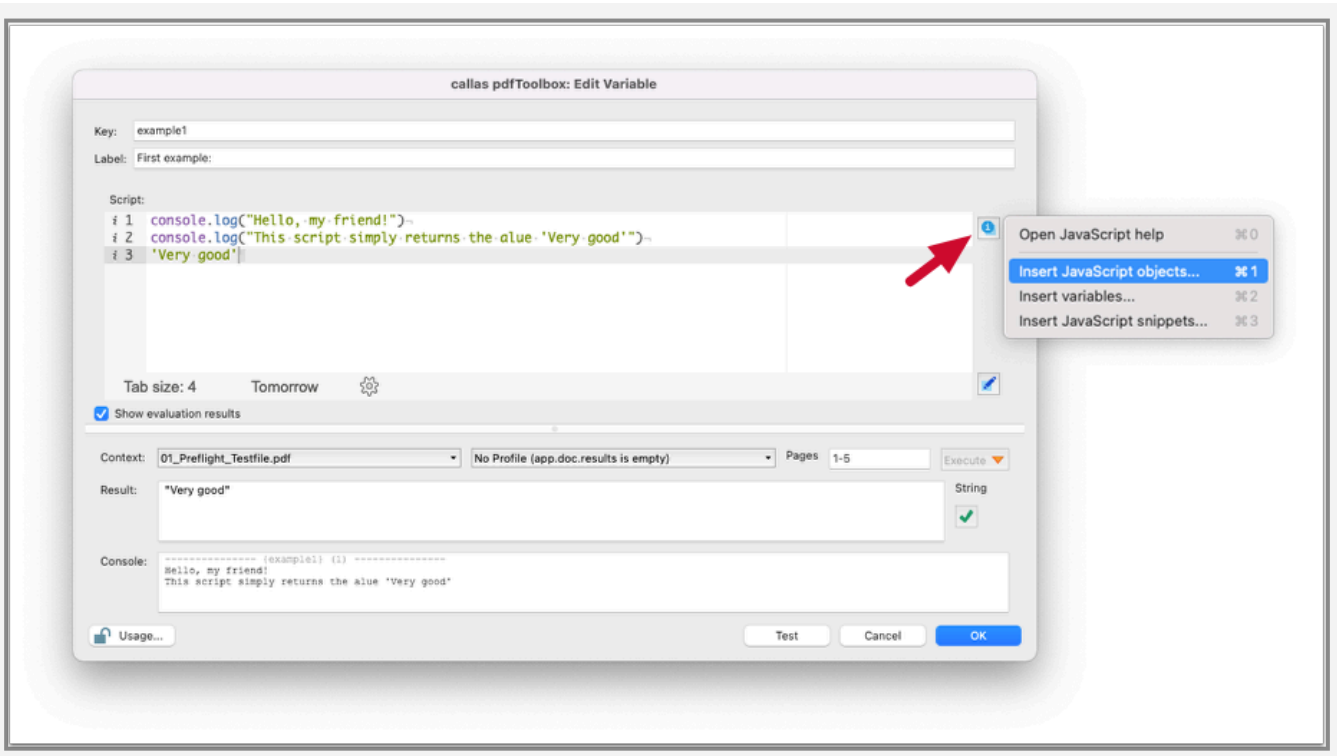
pdfToolbox specific methods

The methods listed below are specific to the pdfToolbox JavaScript engine:

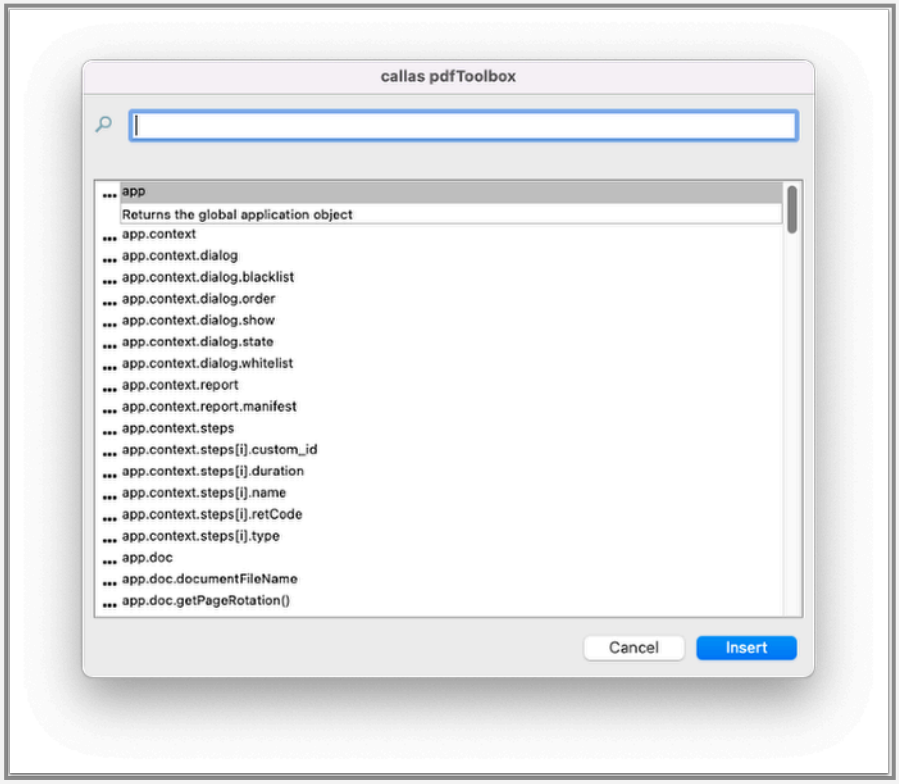
- app.requires
- app.doc.getPageBox()
- app.doc.hasPageBox()
- app.doc.getPageRotation()
- app.doc.pages[i].getPageBox()
- app.doc.pages[i].hasPageBox()
- app.doc.pages[i].getPageRotation()
- app.doc.xmp.getProperty(ns,property)
- app.setTimeout (seconds)
- app.http.get(url)
- console.log()
- console.info()
- console.warn()
- console.error()
- File.read()
- XML.xpath("expression")
- XML.registerNamespace("prefix","uri")

Looking up pdfToolbox specific data objects and methods while creating or editing scripts

While creating or editing a JavaScript variable a list of pdfToolbox specific custom data objects and methods can be accessed (for this to work, make sure the text cursor is inside the "Script:" field):



Clicking on the object gives you the description of what the object or the method can do.



Interacting with the outside world

Reading from a file on the file system

Example for reading a file:

Assuming presence of a file "`name.json`" at the path "`/Users/username/test`" with the following content:

```
{
        "firstname": "Ulrich"
}
```

the following script could determine the value of the key `firstname`:

```
let file = new File("/Users/username/test/name.json");
let obj = JSON.parse(file.read());
obj.firstname;
```

The result of executing the script would then be:

```
“Ulrich"
```

Writing log statements to the Console field in the script editor window

Below are two examples of code that create a log statement in the Console field inside the script editor window:

```
console.log("Hello world!");
```

```
console.log(JSON.stringify(app.env,null,'\t'));
```

Communication with external resources

Here is a short script that returns the Product_ID of the first cartridge on our AWS license server. Important point here is

the JSON.parse() to convert the JSON string into a JavaScript object.

```
let response = app.http.get("http://examaple_amazonaws.com:1234/status/data.json",
[^] {timeout: 100} );
let r = JSON.parse(response);
r.cartridges[0].product_id;
```

Click to copy

17.5 Interpretation of the app.requires values

In version 10.0, the recognition for input types has been improved.
Depending on the kind of default value, the value is recognized as String, Number or Boolean.

Examples:

```
app.requires("type", true)
```

will be handled as Boolean and therefore shows a checkbox in the ask-at-runtime dialog in the user interface.

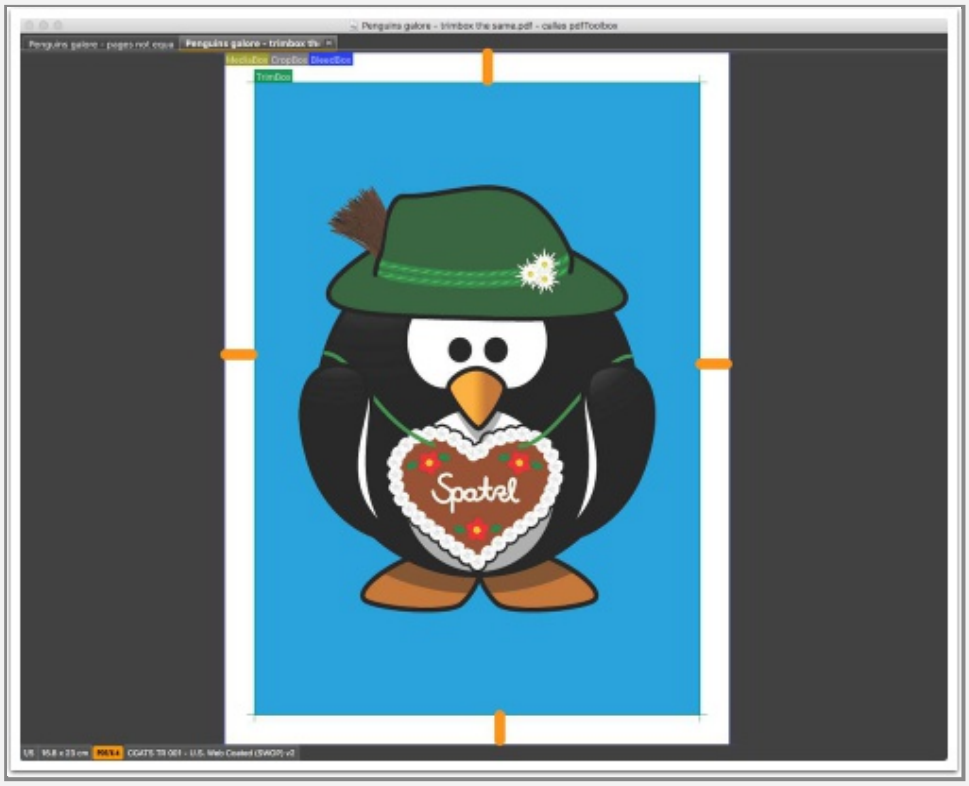
`app.requires("type", "true")` or `app.requires("type", "1")`
will be recognized as String and therefore a text input field is shown in the user interface.

```
app.requires("type", 1)
```

will be handled as Numbers with an input field in the user interface, which is restricted to numerical values.

17.6 Using variables in a property - The TrimBox example

Is the TrimBox centered or not



The file above has a MediaBox and a TrimBox and our goal is to find whether the TrimBox lies in the center of the MediaBox. Well, there is a Property (which could be used in a Check) in pdfToolbox named 'TrimBox is centered inside MediaBox'. This will check the location of the TrimBox inside the MediaBox, but without any tolerance. That means, if the TrimBox is located in the exact center, hence not off by even a point, the Check is a success. Also the distances between MediaBox and TrimBox in horizontal and vertical direction must be identical.

Define your own condition for centering using JavaScript

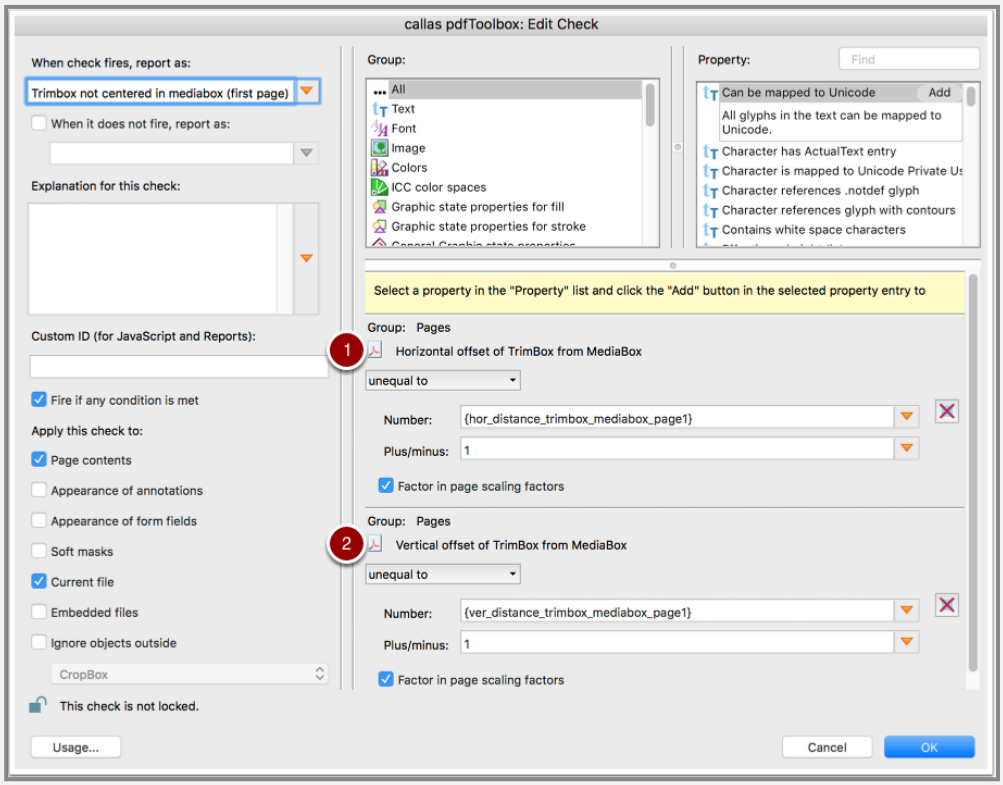
To understand the center, the distance between the TrimBox and the MediaBox on the left and right side of the page (similarly for the top and bottom) has to be equal. Fortunately, there is a condition in pdfToolbox to check the in-between distance on the left hand side named 'Horizontal offset of TrimBox from MediaBox'.

We now have to compare this value to the distance between the two boxes on the right side, for which we will use a JavaScript variable. Hence,

Condition == Distance on the left

JavaScript variable == Distance on the right

And for our requirement, the 'Distance on the left' has to be equal to the 'Distance on the right.'

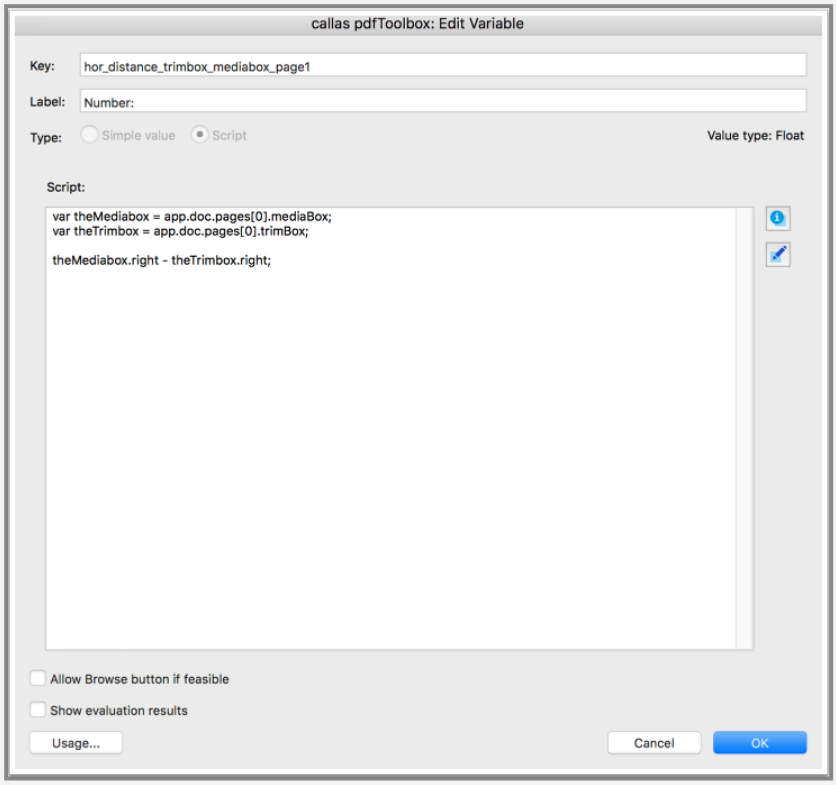


We start by making a new Check, 'TrimBox not centered inside MediaBox' with a custom check including 2 properties:

1. 'Horizontal offset of TrimBox from MediaBox' (to check the distance on the left) and a Javascript variable inside this property to check the distance on the right and comparing them (here, Unequal to)

2. 'Vertical offset of Trimbox from Mediabox' (to check the distance on the top) and a Javascript variable inside this property to check the distance on the bottom and comparing them (here, Unequal to).

What's inside the Script?



Very simple. The script here is calculating the distance between the MediaBox and the TrimBox on the right. (Another script for the vertical distance is not shown here in this example, but it has the same basis)

PLEASE NOTE: This Profile will work brilliantly on a one page file or a file which has the same distance between the two boxes on the right side of the first page and the left side of all the pages.

You can very well notice that the script is picking up the right hand side value only from the first page. Have a look at this:

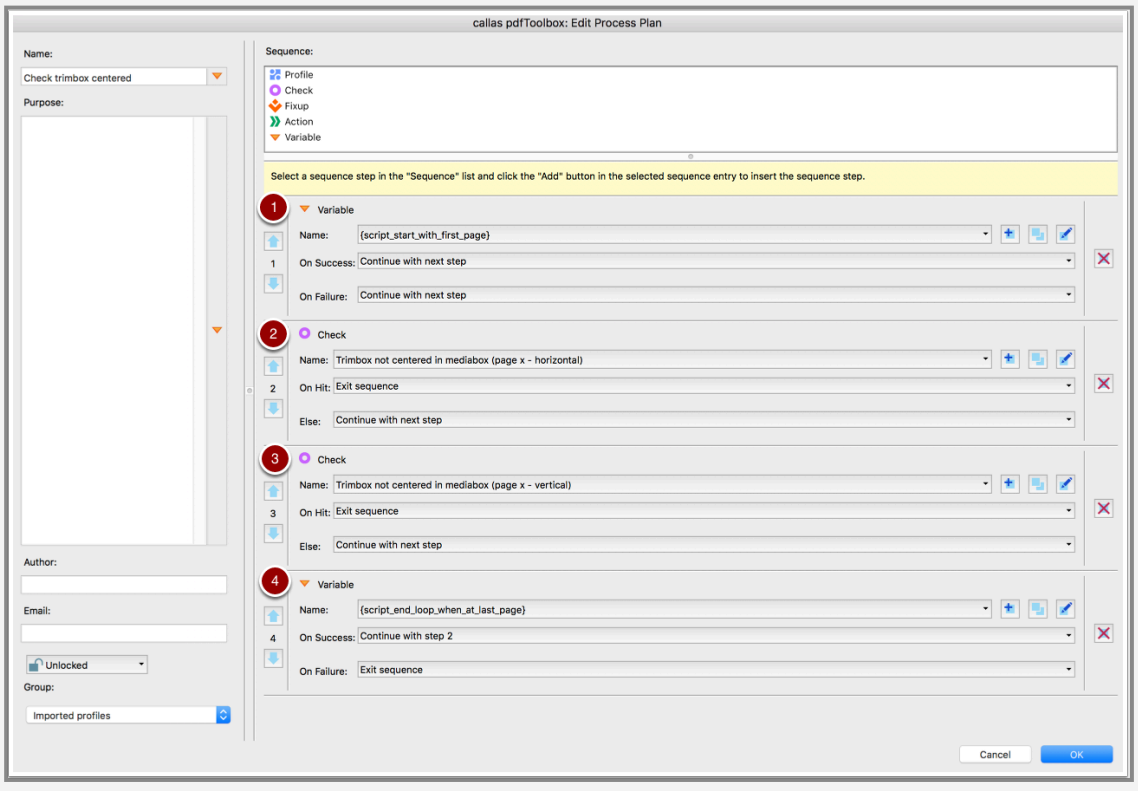
`app.doc.pages[0]` (in JavaScript, the page "0" is the first page)

In case the pages in your file have different MediaBox dimensions on different pages, the script has to be a bit more complicated.

For multi page files or files with different dimensions of boxes on different pages

The problem in the script above was that the JavaScript variable was calculating the distance of the 2 boxes on the right side from the first page of the document and comparing it to the left side of all the pages.

What we want to achieve here is to calculate the distance between the TrimBox and the MediaBox on the right side (hence the JavaScript variable) separately for each page. The solution can be achieved using a Process Plan which will run through all the pages and compare left and right (also top and bottom) for each page individually (Left of page 1 with left of page 2 and so on).



The process plan with loops has 4 parts:

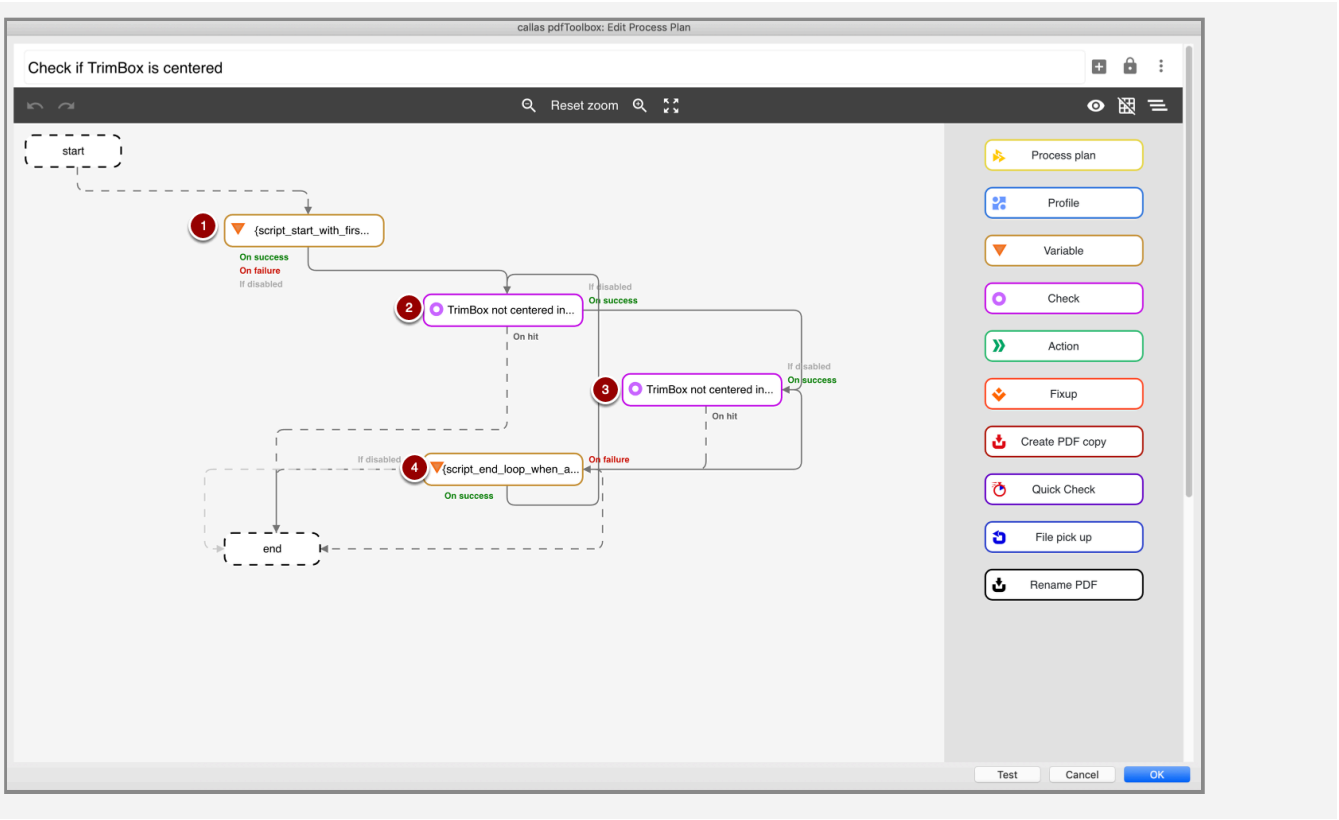
1. A variable to check that the page that we are looking at is page 1.

2. A check for horizontal offset (to calculate the distance between the boxes on the right side for each page individually) for the current page: screenshot below.

3. A check for vertical offset (to calculate the distance between the boxes on the top side for each page individually) for the current page.

4. A variable to increase the page number by 1, continuing with step 2 (if the page number is lower than or equal to the last page) or to end the loop when it is on the last page.

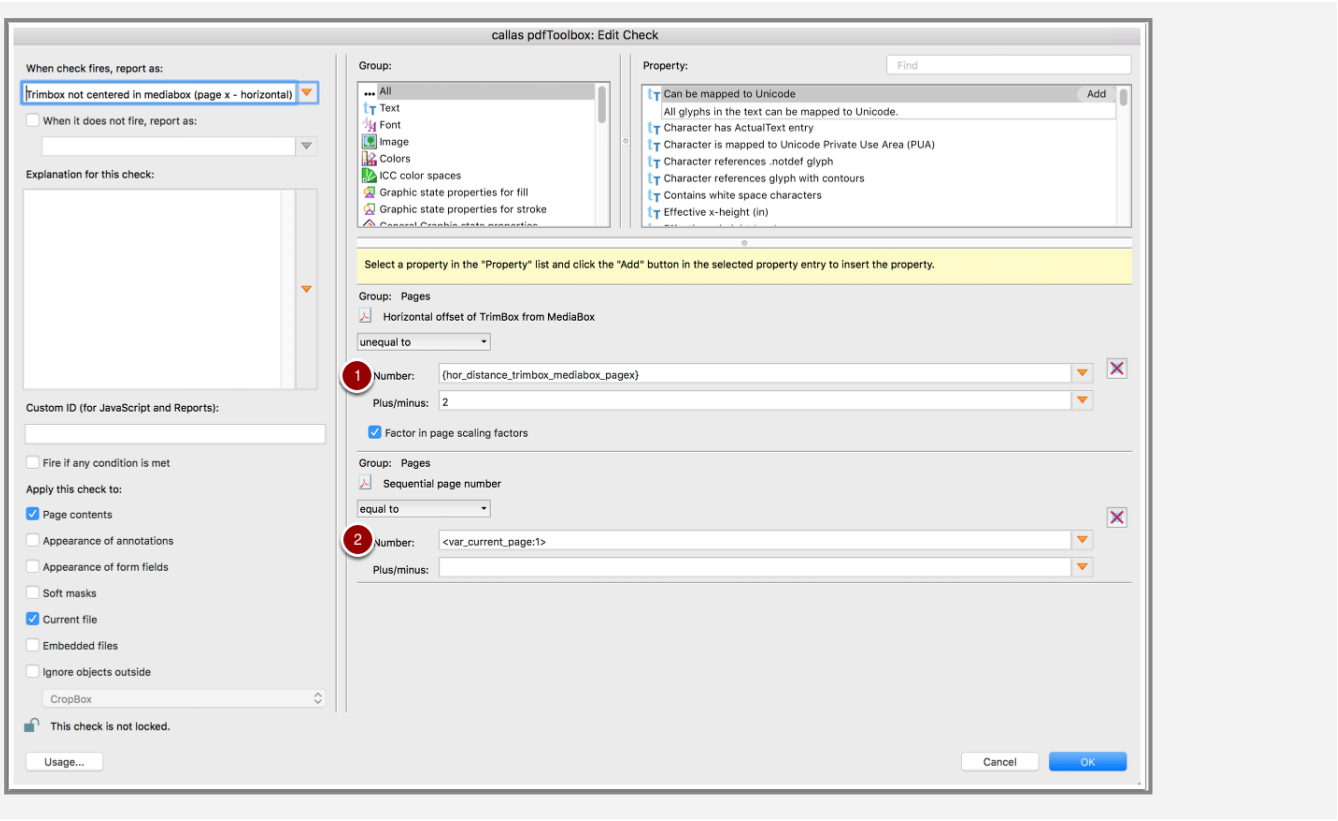
Alternatively starting pdfToolbox 11



The process plan with loops includes 4 sequences:

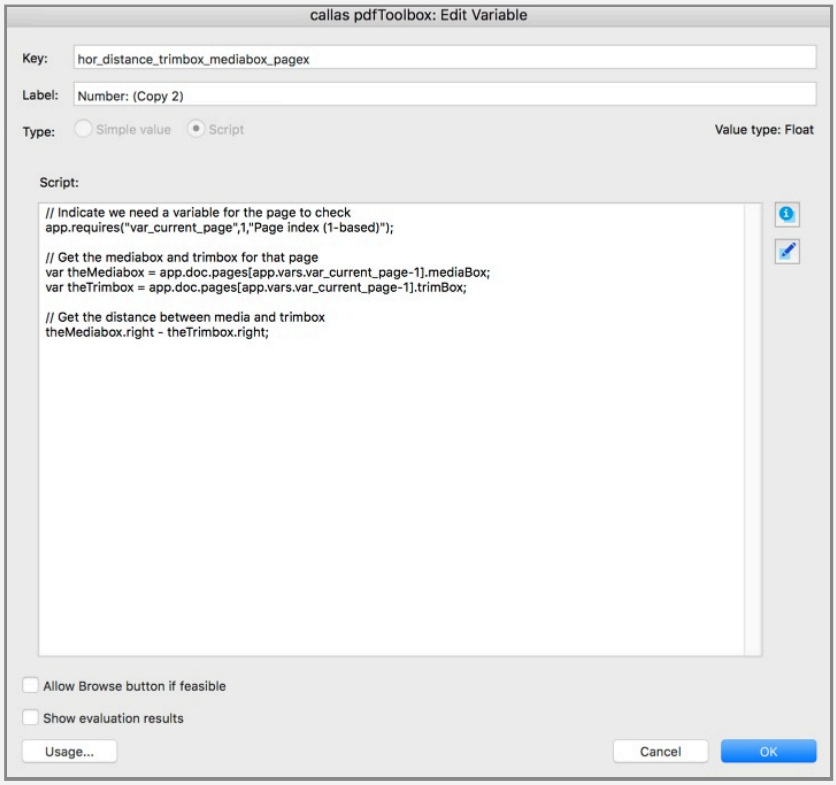
1. A Variable that checks that the page we are looking at is page 1.

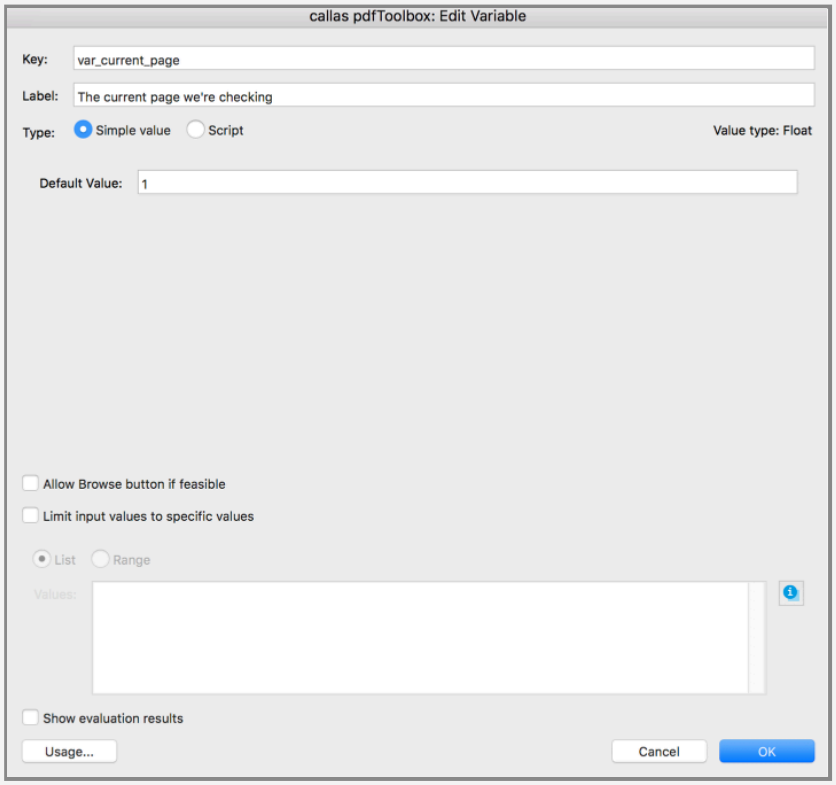
2. A Check on the current page for horizontal offset (to calculate the distance between the frames on the right for each page individually); see figure below.
3. A Check on the current page for vertical offset (to calculate the distance between the frames above for each page individually).
4. A Variable to increase the page number from step 2 by 1 (if the page number is lower than or equal to the last page) or to end the loop with the last page.



The screenshot above shows the Check for the point 2 of the above Process Plan (The Check for point 3 will be similar, hence not shown here).

The above Check contains 2 Scripts (mentioned by numbers and shown below)


callas pdfToolbox: Edit Variable
Key: hor_distance_trimbox_mediabox_pagex
Label: Number: (Copy 2)
Type: Simple value Script
Value type: Float
Script:
// Indicate we need a variable for the page to check
app.requires("var_current_page",1,"Page index (1-based)");
// Get the mediabox and trimbox for that page
var theMediabox = app.doc.pages[app.vars.var_current_page-1].mediaBox;
var theTrimbox = app.doc.pages[app.vars.var_current_page-1].trimBox;
// Get the distance between media and trimbox
theMediabox.right - theTrimbox.right;
Allow Browse button if feasible
Show evaluation results
Usage...
Cancel
OK


callas pdfToolbox: Edit Variable
Key: var_current_page
Label: The current page we're checking
Type: Simple value Script
Value type: Float
Default Value: 1
Allow Browse button if feasible
Limit input values to specific values
List Range
Values:
Show evaluation results
Usage...
Cancel
OK

TrimBox_not_centered_in_MediaBox_(first_page).kfpx

Check_if_TrimBox_is_centered_(ProcessPlan).kfpx

Check_if_TrimBox_is_centered_(ProcessPlan_pdfToolbox_11).kfpx

17.7 Extracting information from an XML Report file via XPath

The pdfToolbox specific "app.doc.result.reports" object returns an array of reports that have been generated in a previous Process Plan step. It can be combined with file.read which would read an XML report into a string and to then convert that string back into an XML object with xml=new XML().

Then xml.registerNamespace allows for associating the XML Report namespace, which is "http://www.callassoftware.com/namespace/pi4" for pdfToolbox XML Reports with a abbreviation.

Finally xml.xpath can be used to read information from the XML object via an XPath expression.

The example below extracts the information about what plates are used by a PDF file from the XML report and writes that information into a variable "text". In the Process Plan example which is attached to this article the value of this variable is then used in a later step to write that information onto all PDF pages.

```
//Get first report, assign it to "file", read it's content into a string and
//convert that string into an XML object
app.vars.report = app.doc.result.reports[0];
var file = new File( app.doc.result.reports[0] );
var string = file.read();
var xml = new XML(string);

//Register the XML namespace with p
xml.registerNamespace("p","http://www.callassoftware.com/namespace/pi4");

//Get the list of platenames
app.vars.plates = xml.xpath( "//p:report/p:document/p:doc_info/p:platenames/
p:platename/text()" );

//Write the list of platenames into a variable that is available throughout the ex-
ecution context
app.vars.text = app.vars.plates;
```

Click to copy

Evaluate_XML_Report_-_Place_Plate_names_extracted_from_XML_report.kfpx

17.8 Using an external JSON jobticket file

This example shows how processing information can be taken from a jobticket file, which has been saved next to the currently processed PDF. The jobticket file uses JSON and does in this example only have one key value pair.

The download contains a Profile with a Process Plan, a sample PDF and a JSON jobticket.



Reading_a_jobticket_from_a_sidecar_file_(JSON).zip

The first step in the Process Plan is a Variable that that tries to read the jobticket and stores one of the values in it into the variable "text". It does the same for the text size which is different for regular content and for an error message that is saved into "text" instead if reading the jobticket fails. This variable is made available via app.vars to the next step which takes it and prints its contents onto the PDF page.

This is the variable from the first step.

```
debug = true;
function buildSidecarFileName( extension )
{
        var path = app.doc.path.split(app.env.pathDelimiter);
                if (debug) console.log( "buildSidecarFileName 1 Path: " + path);
        var name = path[path.length-1].split(".");
                if (debug) console.log( "buildSidecarFileName 2 Name: " + name);
        name.pop();
                if (debug) console.log( "buildSidecarFileName 3 Name: " + name);
        name.push(extension);
                if (debug) console.log( "buildSidecarFileName 4 Name: " + name);
        path.pop();
                if (debug) console.log( "buildSidecarFileName 5 Path: " + path);
        path.push(name.join("."));
                if (debug) console.log( "buildSidecarFileName 6 Path: " + path);
        path = path.join(app.env.pathDelimiter)
                if (debug) console.log( "buildSidecarFileName 7 Path: " + path);
        return new File(path);
```

```
}
try
{
        app.vars.sidecar = JSON.parse( buildSidecarFileName("json").read());
                if (debug) console.log( "main 1 The jobticket file: " + JSON.
stringify(app.vars.sidecar));
        app.vars.text = app.vars.sidecar.msg;
        app.vars.fontsize = 25;
        app.vars.ok = true;
}
catch( x )
{
        app.vars.text = "ERROR: Could not read message from sidecar file: " + x;
        app.vars.fontsize = 20;
        app.vars.ok = false;
}
app.vars.ok;
```

Click to copy

The first call sets the debug variable to true. This allows for reading the current state of processing in the Console window of the JavaScript editor (if "Show evaluation results" is on). This is the output for "testimonial Mercedes.PDF".

```
buildSidecarFileName 1 Path: ,Users,d.seggern,Doku,Reading a jobticket from a side-
car file (JSON),testimonial Mercedes.pdf
buildSidecarFileName 2 Name: testimonial Mercedes,pdf
buildSidecarFileName 3 Name: testimonial Mercedes
buildSidecarFileName 4 Name: testimonial Mercedes,json
buildSidecarFileName 5 Path: ,Users,d.seggern,Doku,Reading a jobticket from a side-
car file (JSON),
buildSidecarFileName 6 Path: ,Users,d.seggern,Doku,Reading a jobticket from a side-
car file (JSON),testimonial Mercedes.json
buildSidecarFileName 7 Path: /Users/d.seggern/Doku/Reading a jobticket from a side-
car file (JSON)/testimonial Mercedes.json
main 1 The jobticket file: {“msg”:”This string came from a sidecar file!”}
```

Click to copy

The Console output shows how the path for the jobticket file is built from the path of the PDF file, in the lines starting with "buildSidecarFileName" and the contents of the jobticket file, which is rather short in this example.

17.9 Defining variables using app.requires with closed choice of allowed values

The app.requires object can be used for two reasons:

To explicitly define dependencies in a Script variable from another variable. That should always be done when variables are used e.g. in Checsk or Fixups where it is important that a certain order is used when evaluation the variables.

The other more important use case is to define a variable within a JavaScript variable. This will be described in this article.

The example is variant of the "Viewing Distance related checks" Profile that you will find in pdfToolbox in the "Shapes, Variables, JavaScript, Place content" library.

Viewing_Distance_related_checks.kfpx

The predefined Profile has a Profile level JavaScript variable that starts with

```
app.requires("input_viewingdistance",10,"Viewing distance [meter]");
app.requires("input_scalingfactor",100,"Scaling factor for PDF [%]");
```

Click to copy

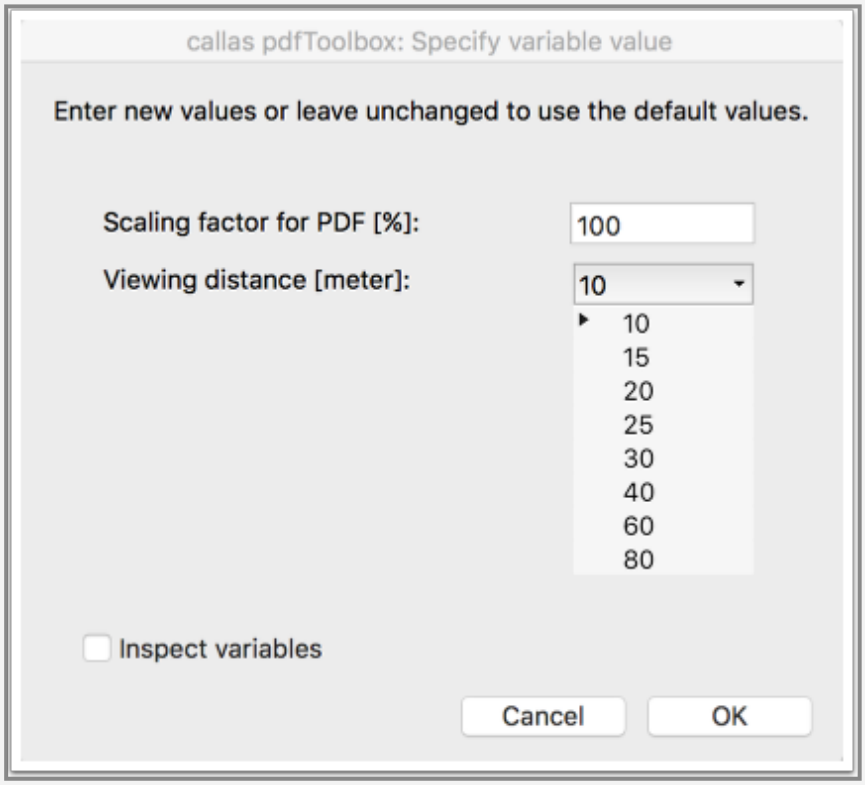
The two app.requires calls define two variables, "input_viewingdistance" and "input_scalingfactor" , with default values (second parameter) and a label text that is used when the Profile is executed in pdfToolbox Desktop (third parameter).

With pdfToolbox 9.1 it is possible to define a list of possible values.

```
app.requires("input_viewingdistance",10,"Viewing distance [meter]",[10,12,14,18,20]);
```

Click to copy

If you want to only allow certain viewing distance values for a variable that is defined using app.requires you can use a fourth parameter that takes an array of values.

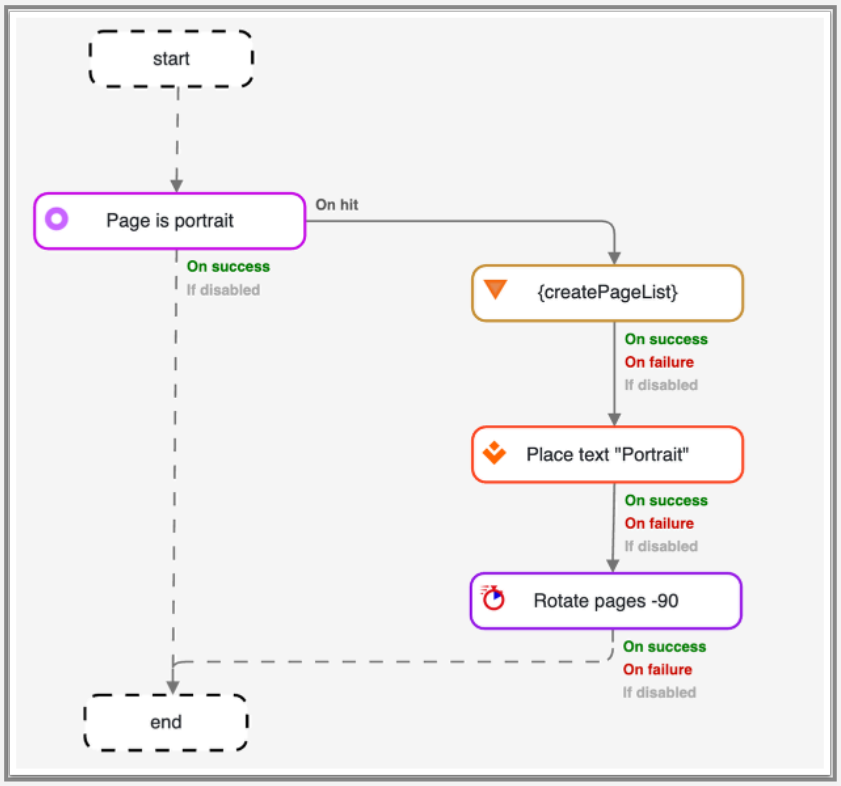


(For variables that are defined via the variable pop up you can do the same by using the "Limit input values to specific values" option.)

17.10 Process pages differently in a Process Plan using a Check and JavaScript

It is possible to process pages differently depending on the result of a Check (or a Profile). In this article we are using a Process Plan that first identifies pages that are portrait (and not landscape). These pages are then stamped and rotated.

The Process Plan uses 4 steps.



Process pages differently with a page list.kfpx

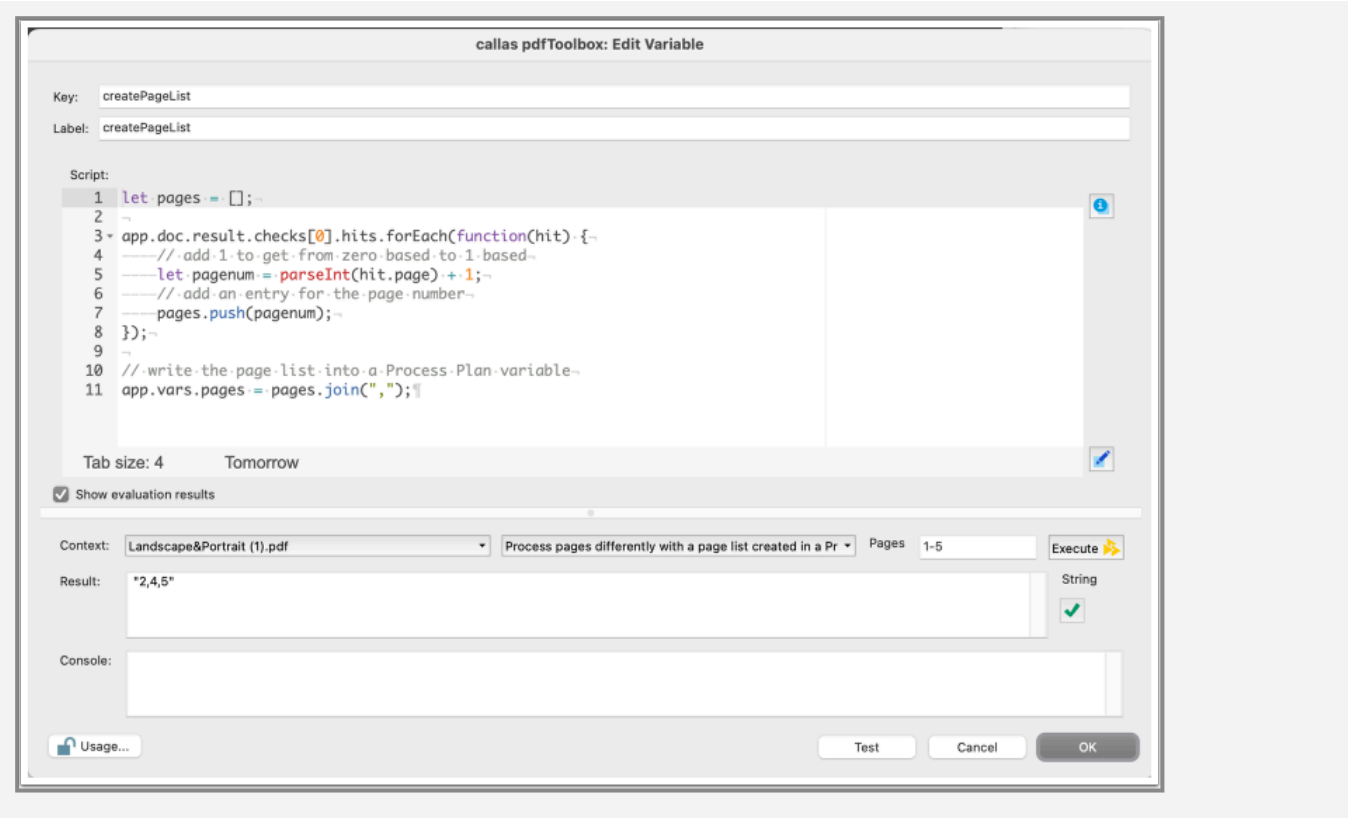
If you need one to try out this is a PDF with portrait and landscape pages.

Landscape&Portrait.pdf

Page is portrait

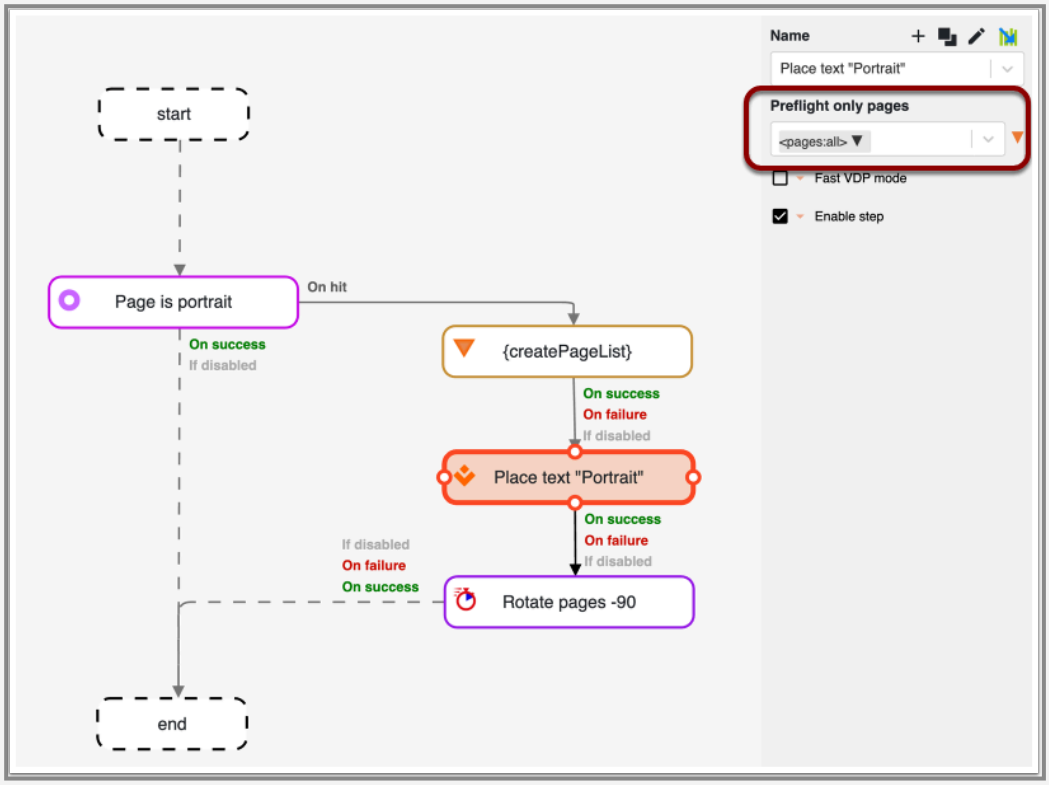
The first step "Page is portrait" checks whether there are any such pages, if not (on success) the Process Plan is immediately terminated. If there are such pages (on hit) a JavaScript step is executed. Since pdfToolbox updates it's JavaScript object after Checks this object now has information which pages are portrait. The JavaScript takes this result object and converts it into a comma separated page list.

Create page list



For each item in the result object the script extracts the page number, adds 1 (because in the result object page numbers are 0 based) and appends it to the page list. Finally the page list that has been created in the script is written into a global variable app.vars.pages.

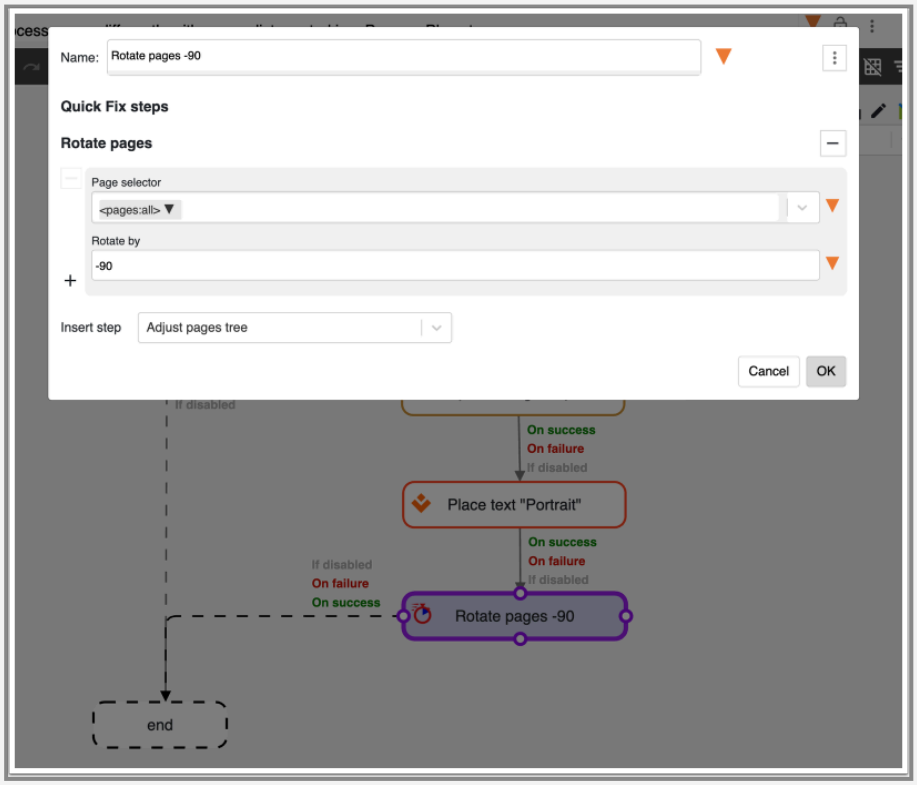
Place text "Portrait"



The next step places a stamp "Portrait" on the respective pages. The page list that had been created in the previous step and saved into app.vars.pages is now used in the page filter "Preflight only pages" for the Fixup.

Rotate pages -90

The next and last step then rotates the pages from the page list. "Rotate pages -90" is based on a Quick Fix since that is faster than the corresponding regular Fixup. A Quick Fix always has it's own page selector, so we do not use the Process Plan sidebar, but a Quick Fix parameter.


Name: Rotate pages -90
Quick Fix steps
Rotate pages
Page selector
<pages:all>
Rotate by
-90
Insert step
Adjust pages tree
Cancel
OK
If disabled
On success
On failure
If disabled
Place text "Portrait"
On success
On failure
If disabled
If disabled
On failure
On success
Rotate pages -90
end

17.11 Using "trigger" values to adjust processing in a Process Plan

Since pdfToolbox 9.1, you can access trigger values via a JavaScript object.

Trigger values are values that pdfToolbox reports back for objects that are identified by a check. The type of value corresponds to the check property that is used. E.g. for a check that finds all image objects with an image resolution below 300 ppi the trigger value reports the actual image resolution for each such image. For a check that finds objects that are close to TrimBox the trigger value has the actual distance between any such object and the TrimBox. The examples below use these two check properties and their trigger values to adjust processing:

- List images with lowest resolution per page (uses trigger values)
  Uses the image resolution trigger values and prints on each page of a PDF file the lowest resolution that is used by an image on that page.
- Use trigger values to calculate the width for page mirroring for bleed creation
  Uses the smallest distance between a text object and the TrimBox to define the width for mirroring page content. That makes sure that text objects do not show up in the bleed. It does this in a loop until there is no text closer than 3 mm to the mirrored page content.

The download contains kfpx Process Plans for the two examples as well as a simple demo file that can be used to try them out.

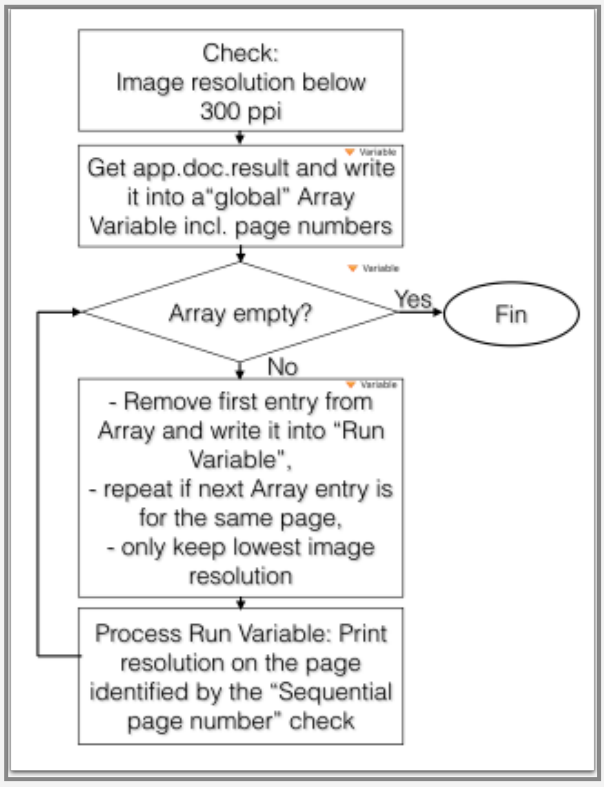


Trigger_value_example.zip

List images with lowest resolution per page (uses trigger values)

This Process Plan runs in a loop. The flow chart below gives an overview about how processing takes place in the 5 steps

of the Process Plan and how that makes sure that each page is processed separately.



Use trigger values to calculate the width for page mirroring for bleed creation

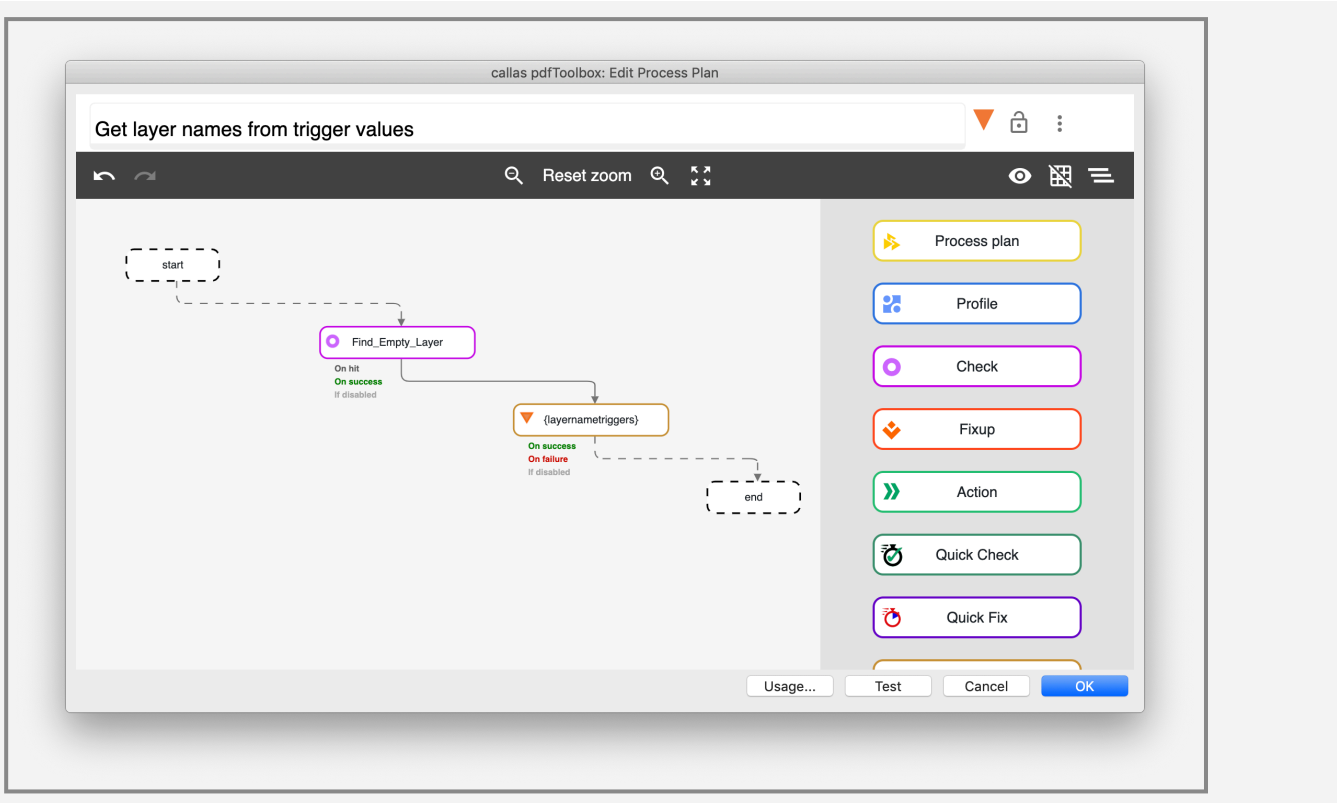
The 6 steps of this Process Plans are:

1. Set ArtBox to TrimBox
   This is done to "remember" the TrimBox since that has to be modified during processing.
2. Check: Text is closer than 3 mm inside to TrimBox
   Finds all text objects that are so close to TrimBox that they would be mirrored by a bleed generation fixup.
3. Mirror page into bleed with a width that text is not mirrored
   The width of bleed is defined by using a Script Variable "width_of_bleed". That first copies app.doc.result.checks[0].hits into a local Variable. It then generates an array variable "loc_triggerarray" into which all trigger values are written. Finally Math.min.apply is used to determine what is the smallest distance.

4. Then the TrimBox is set to the new CropBox because that makes it easier to find out whether we already have generated enough bleed.
   After that processing goes back to the Check in step 2. If by then there is no text that is close to the (new) TrimBox (that at the same time is the CropBox) processing goes forth to step 5, otherwise the procedure above is repeated.
5. Processing get to here only if there is no text closer than 3 mm to TrimBox. It sets the TrimBox back to the ArtBox which was the original TrimBox of the file after step 1.
6. Finally the ArtBox that was only used temporarily has to be removed.

Access "detailed" information from trigger values

Starting pdfToolbox 12.3, it is possible to access "detailed" information from trigger values.



The Process Plan above has

- a Check 'Find_Empty_Layer' that looks for empty layers in a PDF
- a Variable 'layernametriggers' that returns detailed information from the trigger values (see below screenshot)



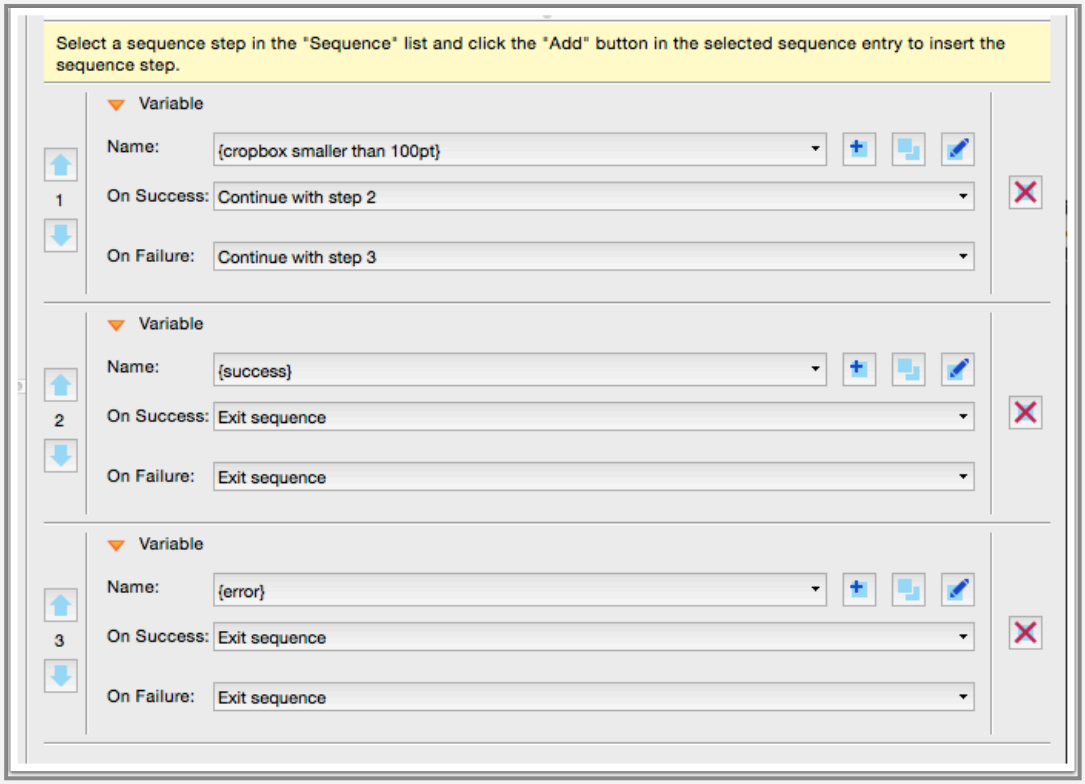

```
app.doc.result.checks[i].hits[i].triggers[i].details;
```

Click to copy

17.12 How to branch processing in a Process Plan using JavaScript

When using a "**Variable**" sequence step within a Process Plan, it is possible to branch depending on the result of the variable.

This sample uses a "Variable" step in step 1, which continues processing with step 2 in the case of "Success" or jumps to step 3 in the case of an "Error".



To branch in a Process Plan using a variable, the variable or the script must return one the two following values:

"false" or "0" (Null): Failure

"true" or "1": Success

For example within the Variable used in step 1 {cropbox smaller than 100pt}:

```
if ( app.doc.pages[0].cropBox.right - app.doc.pages[0].cropBox.left < 100)
    var result = 1;
```

```
else
    var result = 0;
//return a value:
result
```

In order to try this out yourself, please execute the attached Process Plan "Branch_on_script_result.kfpx" with the activated "Log profile execution" option (in the fly out menu in the upper right of the Profiles/Checks/Fixups window) in the desktop version of pdfToolbox.

Executing the Process Plan will create a log as shown below:

```
2016-12-22 17:30:15 Branch on script result
2016-12-22 17:30:15 Input /Volumes/Work/PDFs/blank.pdf
2016-12-22 17:30:15 Starting with step1
2016-12-22 17:30:15 [1-1] cropbox smaller than 100pt
2016-12-22 17:30:15 Var cropbox smaller than 100pt 0
2016-12-22 17:30:15 Result Failure
2016-12-22 17:30:15 Continuing with step 3
2016-12-22 17:30:15 [2-3] error
2016-12-22 17:30:15 Var cropbox smaller than 100pt 0
2016-12-22 17:30:15 Var error Crop box is not smaller than 100pt
2016-12-22 17:30:15 Var message Crop box is not smaller than 100pt
2016-12-22 17:30:15 Result Success
2016-12-22 17:30:15 Terminating
```



Branch_on_script_result.kfpx

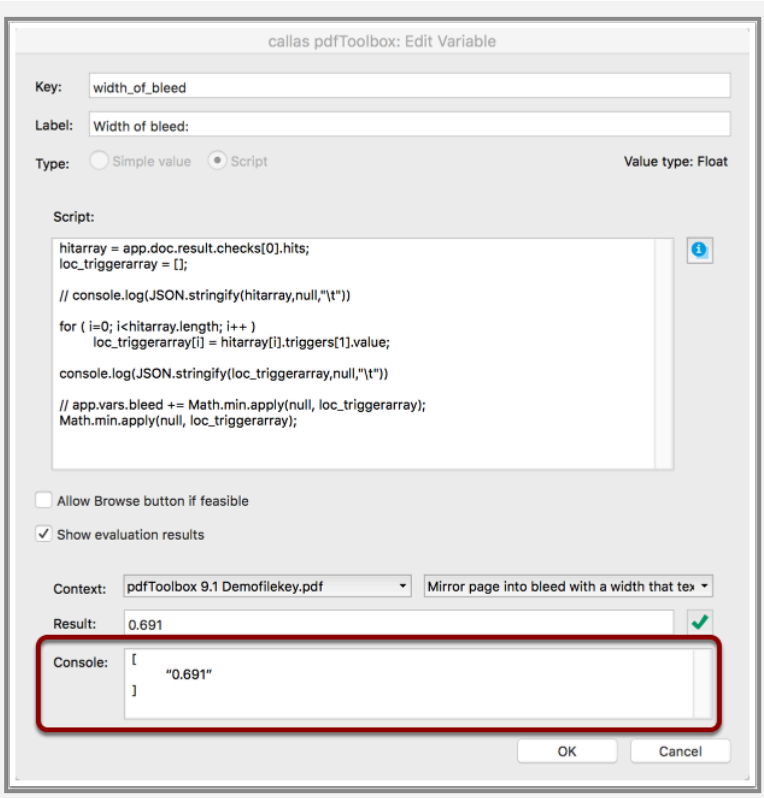
17.13 Debugging JavaScript Variables

pdfToolbox provides features that are designed specifically to make debugging of Script Variables easier:

- Console
- Serialisation of pdfToolbox JavaScript objects
- Variable values are listed when "Log Profile Execution" is active

Console

The Console is available below the Script Editor if "Show evaluation result" in enabled.



It works in the same way as in other JavaScript editors.

Since JavaScript "snippets" may be used at various places in a Profile it is sometimes required to temporarily copy code from somewhere else into the first lines of the Script Variable you are currently working on to make sure that the Variable

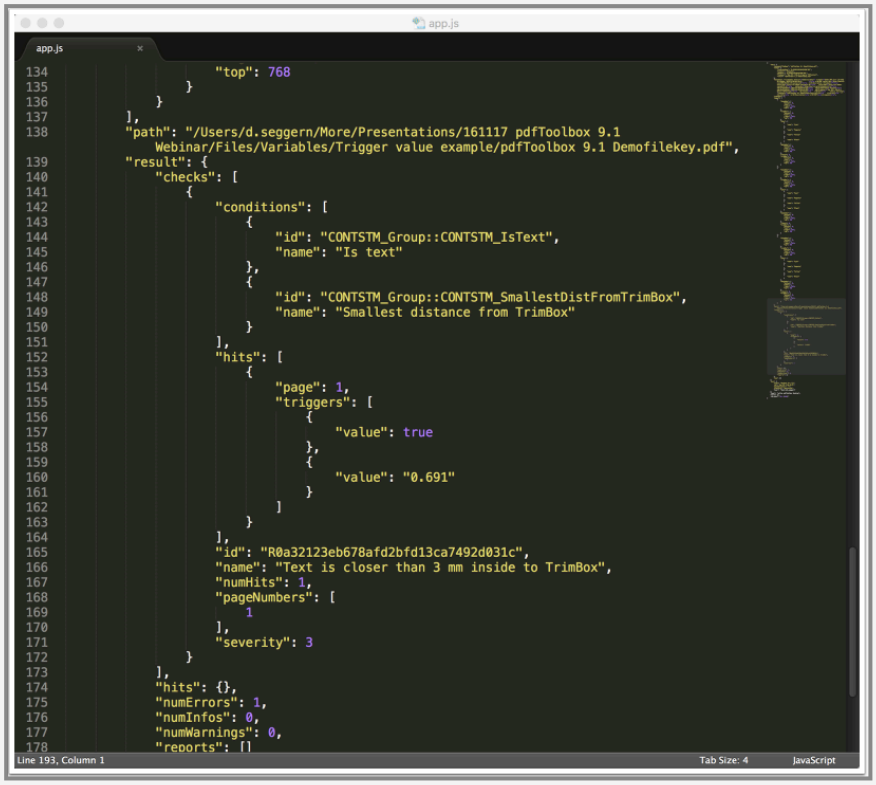
has the same information as when it will be used in the Profile context.

In the example above the check that would fill the app.doc.result.checks[0].hits array has to be manually performed before the code in this Variable can be debugged. Only then the variable "loc_triggerarray" has a meaningful value. It is printed to the Console via console.log using "pretty printing" with the optional parameters null,"\t".

Serialisation of pdfToolbox JavaScript objects

In order to access information in any of the pdfToolbox JavaScript objects you may need to visualise the structure of these sometimes complex objects. The most complex object is the app object since that is the parent object for all other more specific objects.

This and all other pdfToolbox JavaScript objects can be serialised by the JSON.stringify function, if used with the optional parameters null,"\t" they are formatted nicely.

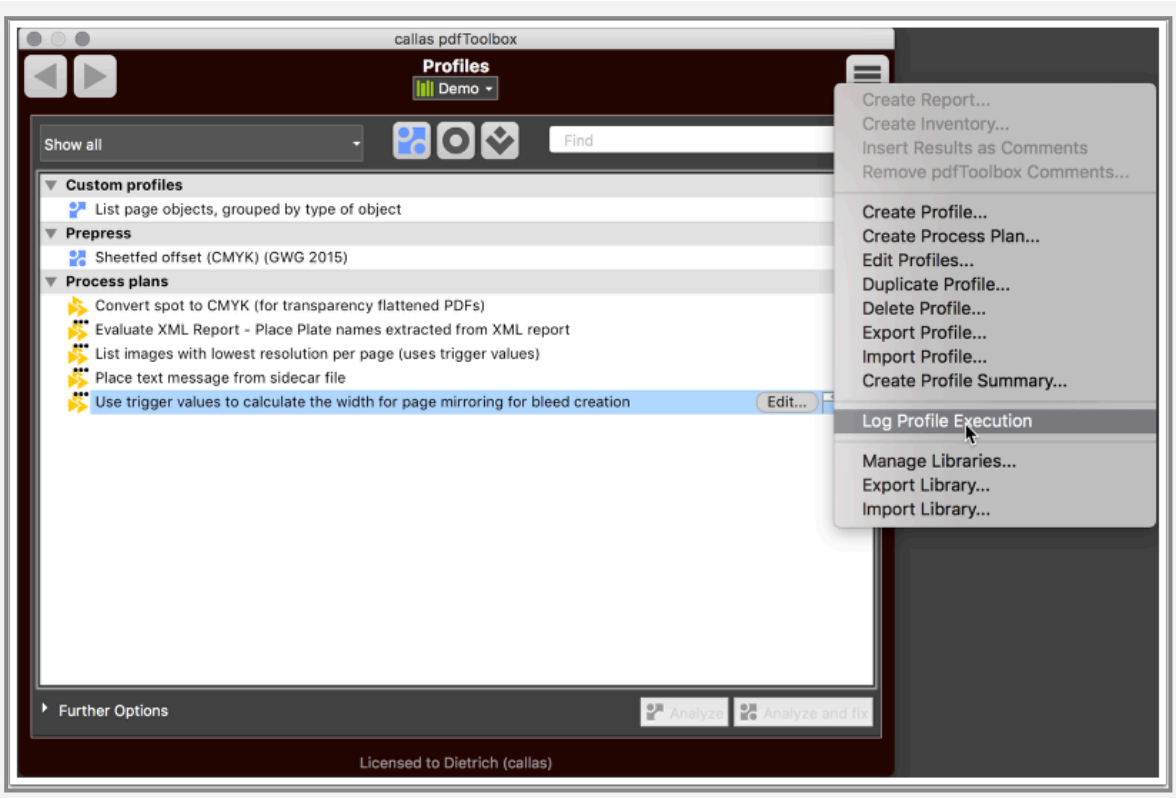


This screenshot shows parts of the app object after that has been output via console.log(JSON.stringify(app,null,"\t"))

and copied from the Console into the Sublime editor. The part above shows the app.doc.result.checks part with hits and their trigger values.

Variable values listed in a logfile created via Log Profile Execution

If you are using Script Variables in a Process Plan you may need to know how Variable values are changed throughout processing. You enable logging via the options menu in the pdfToolbox main window.



If switched on a logfolder will be created that amongst other things includes a log file "process.log". The other contents of this folder are explained in further detail in a different article of this documentation: http://help.callassoftware.com/m/pdftoolbox9-en/l/656888.

All Variables that are stored in app.vars so that they are available throughout a Process Plan are listed for each of the steps of the Process Plan with their current values.



This screenshot shows that the variable "hitarray" had two entries in step 3 and was empty in step 4. (The empty lines before, between and after these two steps have been added to make the example more readable.)

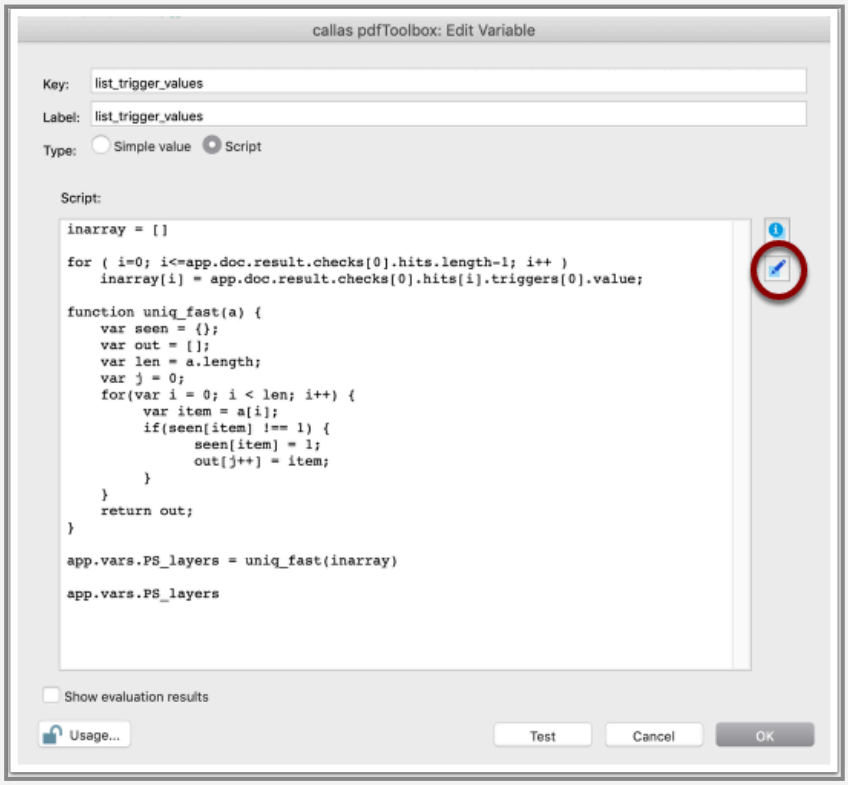
Debugging with MS Visual Studio Code (version 10.1)

You can use MS Visual Studio Code to debug your JavaScript variables. Then the pdfToolbox specific environment is exported to MS Visual Code so that you can truly debug in the pdfToolbox "environment".

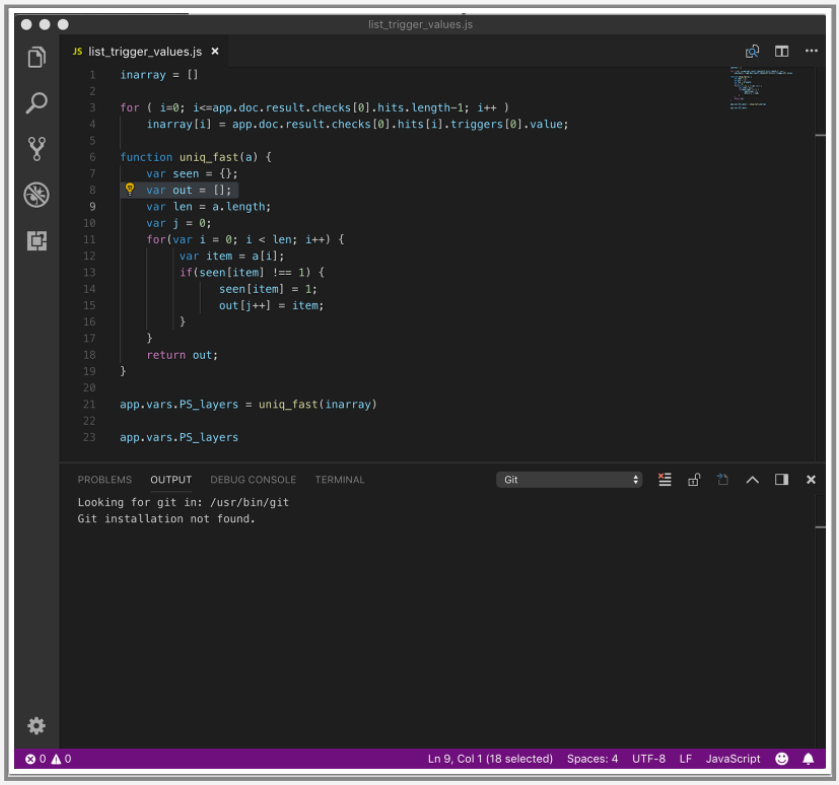
In order to set this up you need to:

1. Install nodeJS (https://nodejs.org/en/download/)
2. Install MS Visual Code (https://code.visualstudio.com/) in the default location
3. Mac Only: Install MS Visual Studio Code command-line tools.
   On Windows and Linux this should be automatically available after installing Visual Studio Code. On macOS, you need to open "View" -> "Command Palette…" and select "Shell Command: Install 'code' command in PATH" (further information on this can be found here).

Then you can use MS Visual Studio Code for debugging.



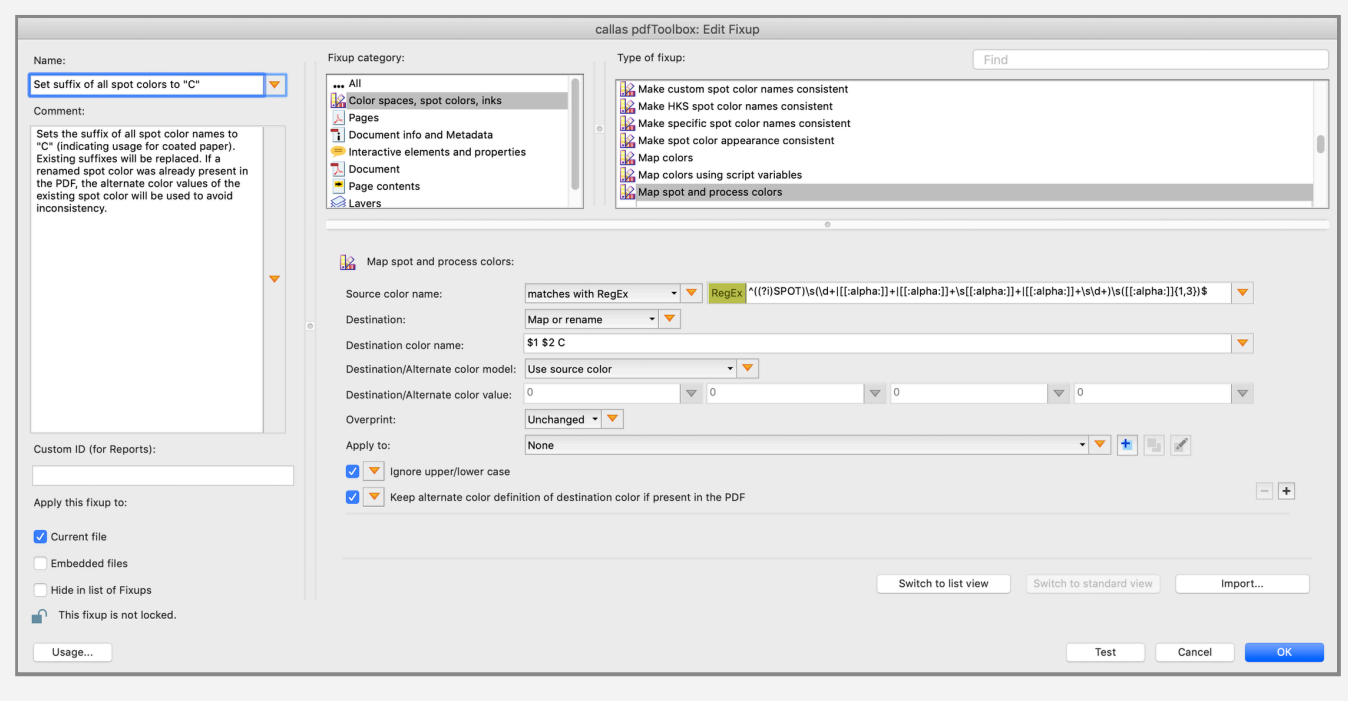
Click on the pen symbol that will open your JavaScript in an external editor. If you have done the above steps, it will then open your JavaScript in MS VS Code and you have all features of that debugger including break points etc.


list_trigger_values.js
list_trigger_values.js
inarray = []
for (i=0; i<=app.doc.result.checks[0].hits.length-1; i++)
inarray[i] = app.doc.result.checks[0].hits[i].triggers[0].value;
function uniq_fast(a) {
var seen = {};
var out = [];
var len = a.length;
var j = 0;
for(var i = 0; i < len; i++) {
var item = a[i];
if(seen[item] !== 1) {
seen[item] = 1;
out[j++] = item;
}
}
return out;
}
app.vars.PS_layers = uniq_fast(inarray)
app.vars.PS_layers
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Git
Looking for git in: /usr/bin/git
Git installation not found.
Ln 9, Col 1 (18 selected) Spaces: 4 UTF-8 LF JavaScript

17.14 Use RegEx in variables

In fields that offer variables, you can use regular expressions too (RegEx). It allows complex search patterns with special strings.

Where can RegEx be used as input values



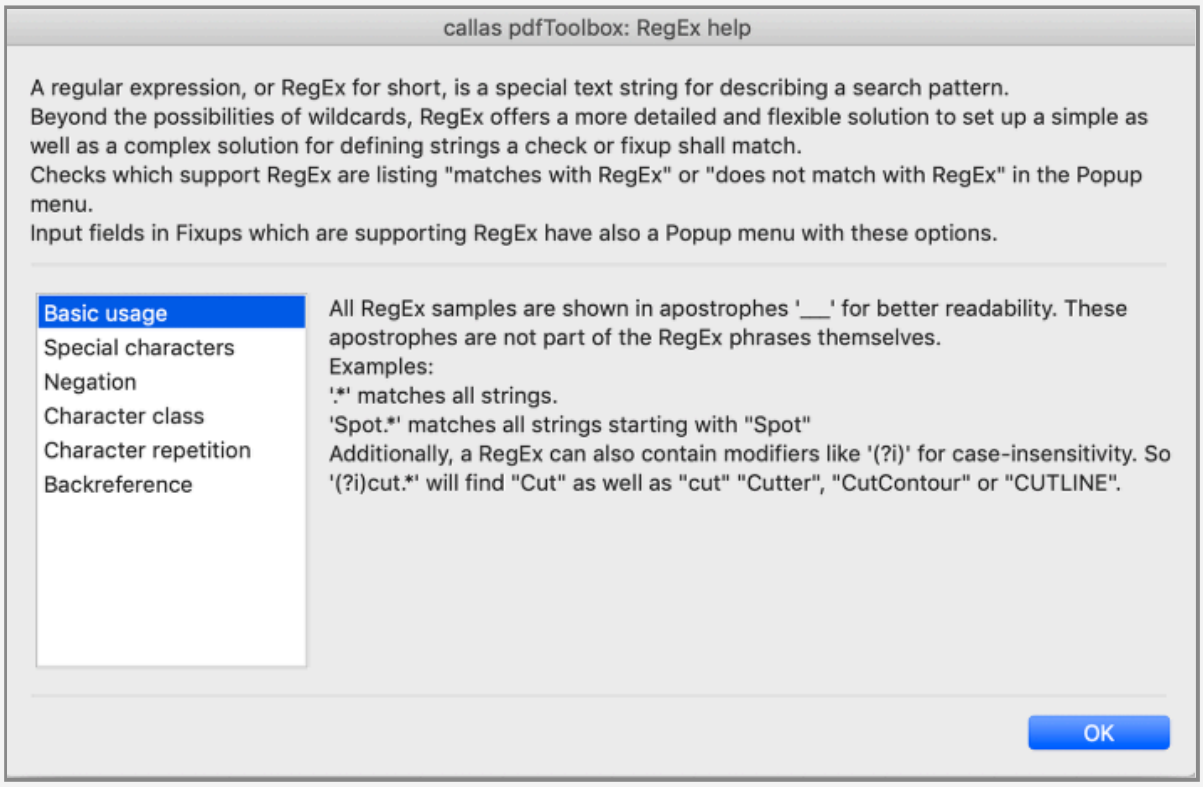
If an input field alles regular expressions can be recognized by the RegEx icon next to the pull down menu. Regular expressions can be used in some Checks or Fixups.

The pull down menu has four states:

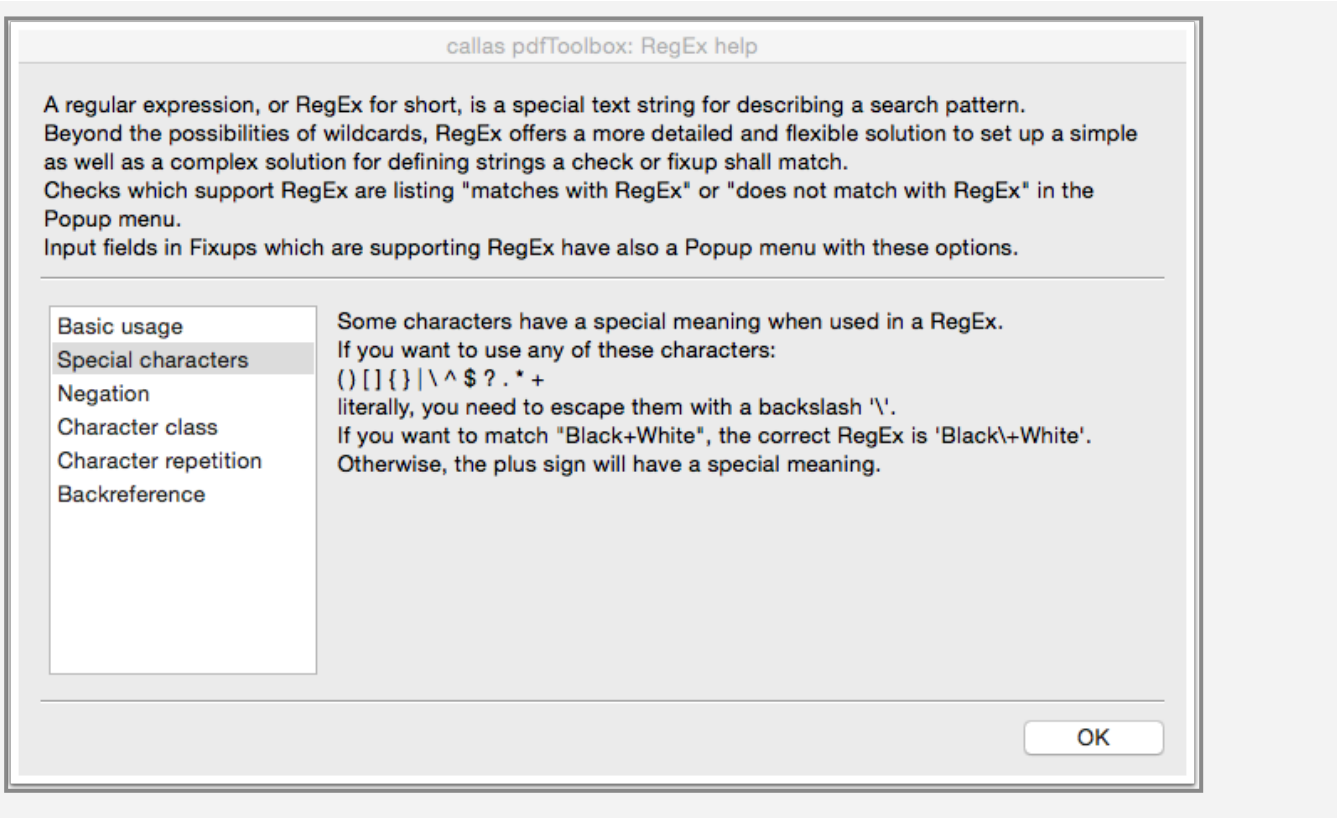
- Equal to
- Unequal to
- Matches with RegEx
- Does not match with RegEx

By clicking on the green "RegEx" icon, an assistance to the practical possibilities of the regular expression is shown.

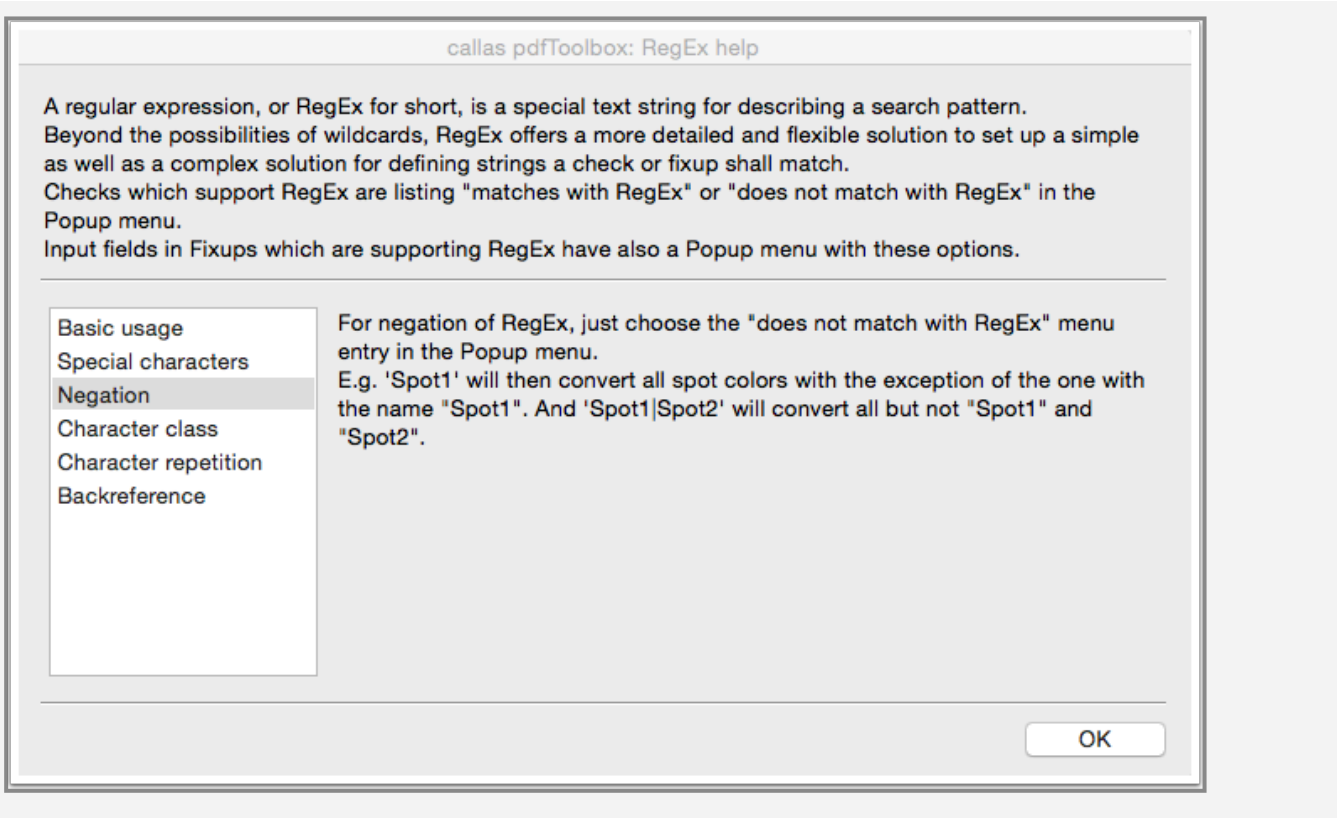
Basic usage


callas pdfToolbox: RegEx help
A regular expression, or RegEx for short, is a special text string for describing a search pattern.
Beyond the possibilities of wildcards, RegEx offers a more detailed and flexible solution to set up a simple as well as a complex solution for defining strings a check or fixup shall match.
Checks which support RegEx are listing "matches with RegEx" or "does not match with RegEx" in the Popup menu.
Input fields in Fixups which are supporting RegEx have also a Popup menu with these options.
Basic usage
Special characters
Negation
Character class
Character repetition
Backreference
All RegEx samples are shown in apostrophes '___' for better readability. These apostrophes are not part of the RegEx phrases themselves.
Examples:
'.*' matches all strings.
'Spot.*' matches all strings starting with "Spot"
Additionally, a RegEx can also contain modifiers like '(?i)' for case-insensitivity. So '(?i)cut.*' will find "Cut" as well as "cut" "Cutter", "CutContour" or "CUTLINE".
OK

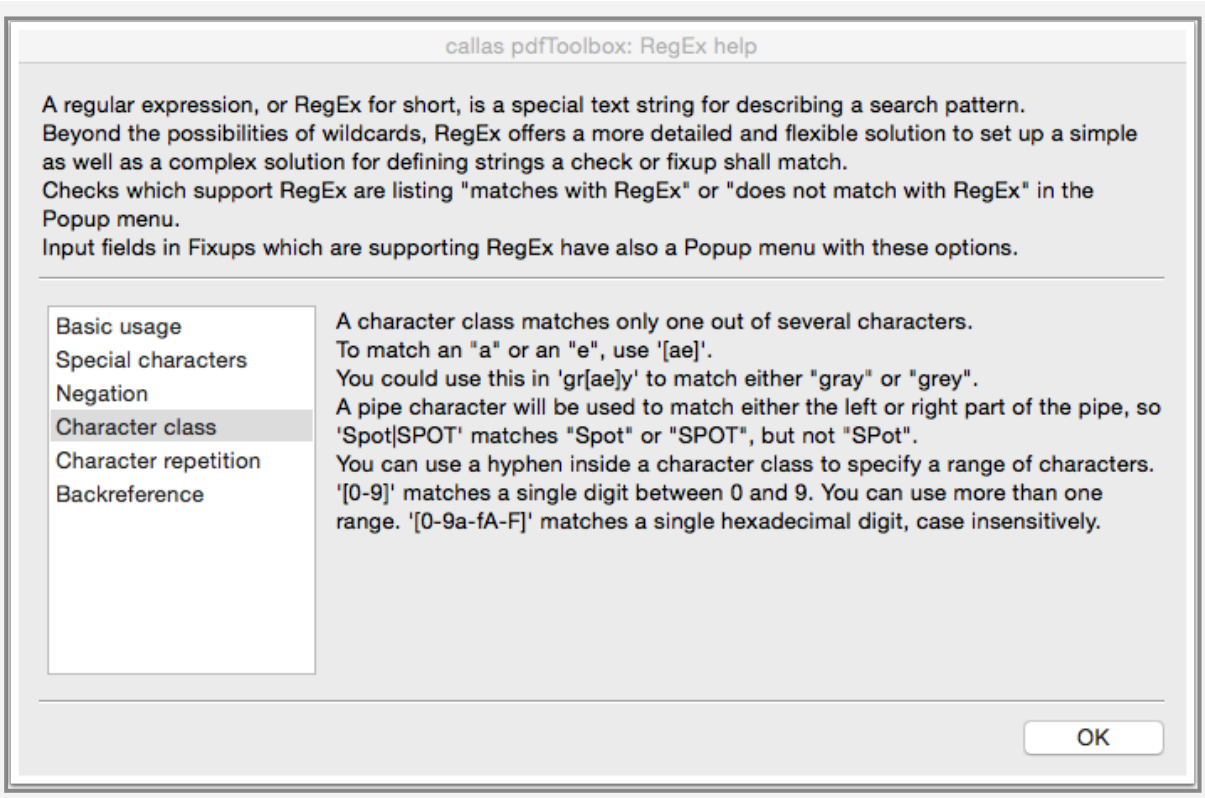
Special characters



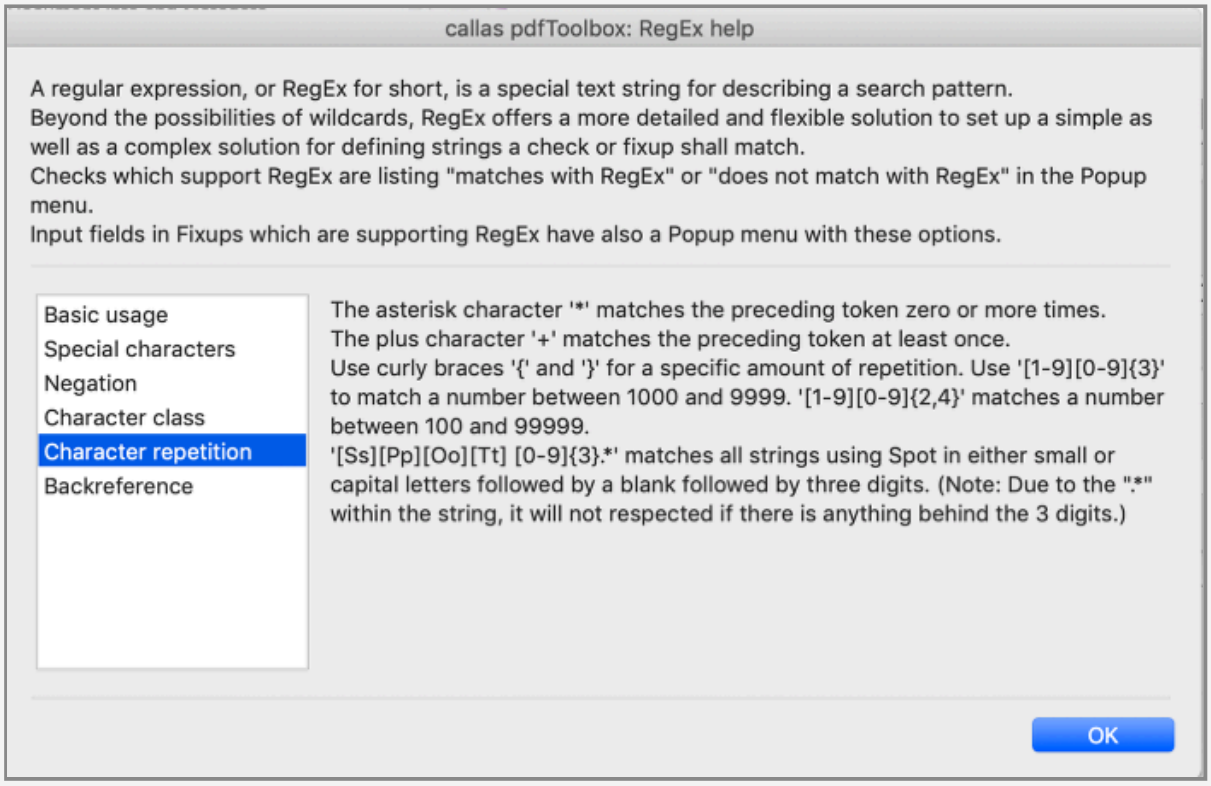
Negation


callas pdfToolbox: RegEx help
A regular expression, or RegEx for short, is a special text string for describing a search pattern.
Beyond the possibilities of wildcards, RegEx offers a more detailed and flexible solution to set up a simple as well as a complex solution for defining strings a check or fixup shall match.
Checks which support RegEx are listing "matches with RegEx" or "does not match with RegEx" in the Popup menu.
Input fields in Fixups which are supporting RegEx have also a Popup menu with these options.
Basic usage
Special characters
Negation
Character class
Character repetition
Backreference
For negation of RegEx, just choose the "does not match with RegEx" menu entry in the Popup menu.
E.g. 'Spot1' will then convert all spot colors with the exception of the one with the name "Spot1". And 'Spot1|Spot2' will convert all but not "Spot1" and "Spot2".
OK

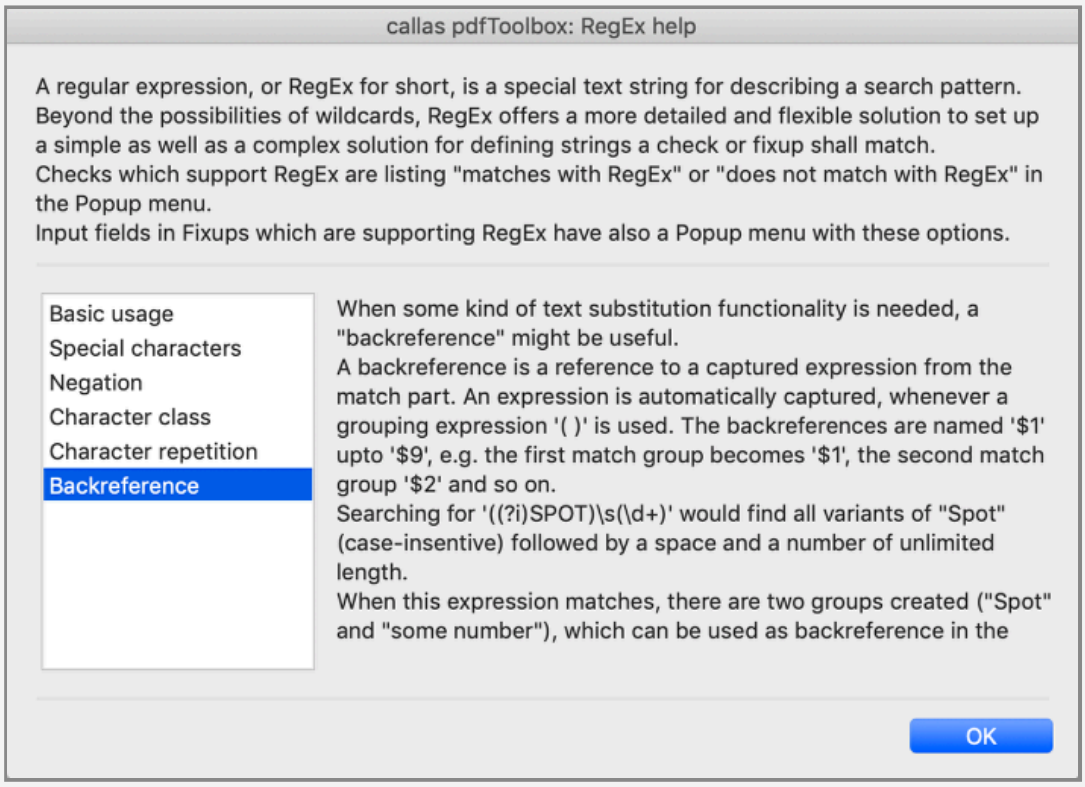
Character class



Character repetition



Backreference



17.15 Using object coordinates from a hit in a Process Plan

In pdfToolbox 11 coordinates for all objects identified by a hit are available in the pdfToolbox JavaScript object app.doc. In this example this information is used to print image resolutions on all images in a PDF file.

Mark_images_with_resolution.kfpx

You can use this Process Plan with any file including this one:

Anzeige_callas_v5.pdf

The Process Plan identifies all images and their image resolutions in step 1. In the second step the results are copied into app.vars.imageHits and in the final step the coordinates are used to place the image resolutions at the top and centered on each image.

An alternative approach is used in this Process Plan.

Mark_images_with_resolution_in_a_loop_via_a_r.kfpx

It prints the resolutions in a loop which is much slower than in the Process Plan above where a combined template is created and then put on top of the page. But since it lacks the complexity of the combined template it may be easier to adjust.

New properties in app.doc.result.checks[i].hits

- "**llx**": lower left x coordinate of snippet bounding box (pt)
- "**lly**": lower left y coordinate of snippet bounding box (pt)
- "**urx**": upper right x coordinate of snippet bounding box (pt)
- "**ury**": upper right y coordinate of snippet bounding box (pt)

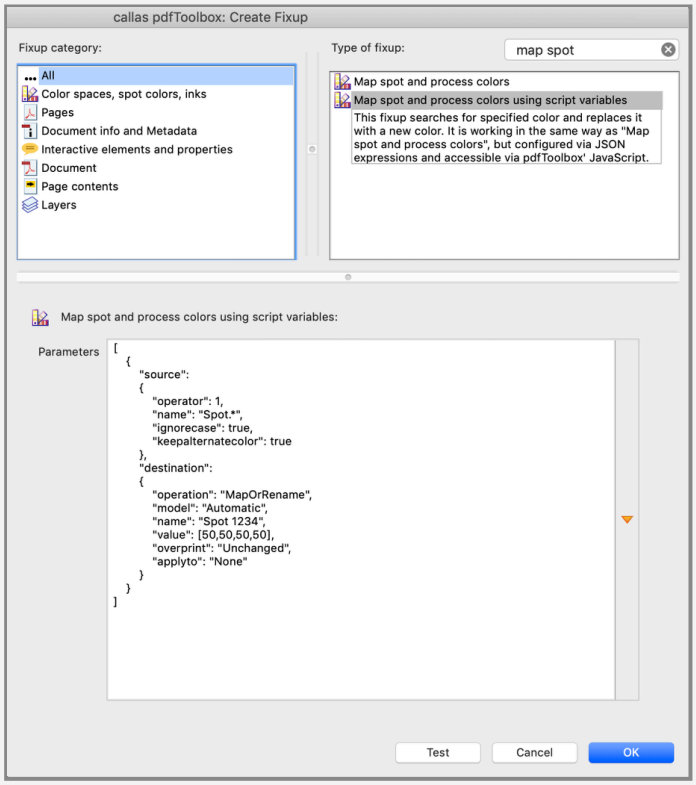
- "**type**": Type of snippets:
  - "Fill"
  - "Stroke"
  - "StrokeFill"
  - "TextFill"
  - "TextOutline"
  - "TextOutlineFill"
  - "TextInvisible"
  - "InlineImage"
  - "XObj"
  - "Image"
  - "FormXObj"
  - "PostScript"
  - "Shade"
  - "Unknown"
  - "InvalidCmd"

17.16 Map (spot and process) colors using script variables

Introduction

When you want to configure the Fixups "Map colors" and "Map spot and process colors" via JavaScript you face some problems, mainly that you can't add additional settings via JavaScript (in the UI you would use the [+] button) and that you have to predefine variable for all dynamic fields.

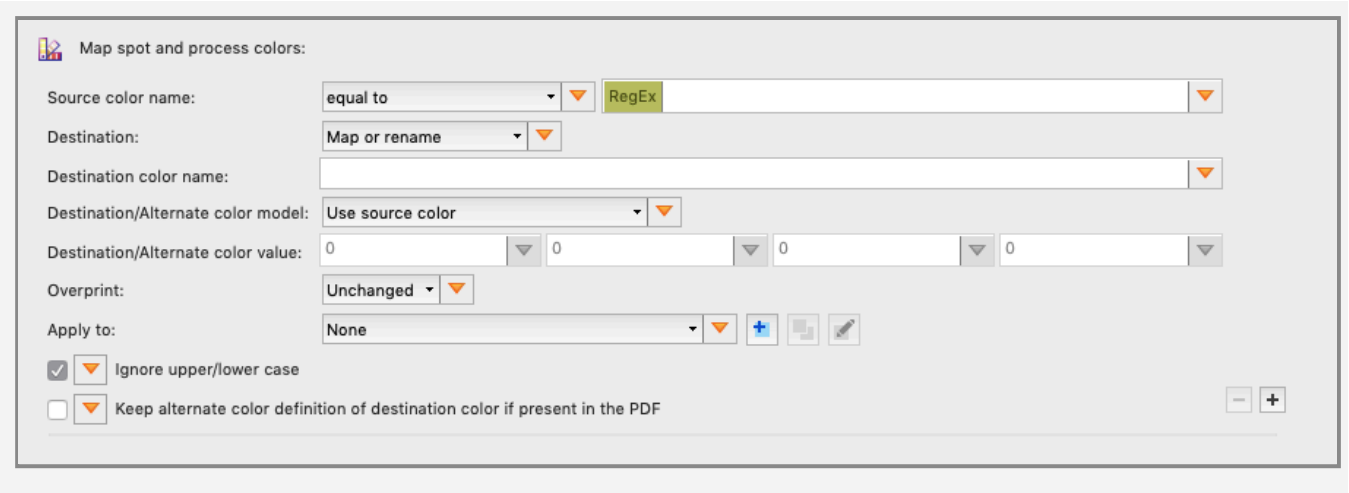
pdfToolbox 11 introduces two new Fixups: "Map colors using script variables" and "Map spot and process colors using script variables" that are configured with a single JSON object.



Usually you will assign a variable to it (via the orange triangle as usual) and create the value of the variable as a JSON structure according to the example in the UI.

Possible values in the JSON structure

Map spot and process colors using script variables



Name	Possible values	Corresponds to ("Map spot and process colors")
source		
operator	1 2 3 4	matches with RegEx does not match with RegEx equal to unequal to
name	<any proper string>	
ignorecase	true false	
keepalternatecolor	true false	
destination		
operation	ConvertToCMYK MapOrRename KeepName ConvertToDestination	Convert to CMYK Map or rename Change alternate color Convert to destination
model	Automatic	Use source color

Name	Possible values	Corresponds to ("Map spot and process colors")
	CMYKPercent CMYKZeroToOne GrayPercent GrayZeroToOne Lab RGBPercent RGBZeroTo255 RGBZeroToOne	CMYK (%) CMYK (0.0...1.0) Gray (%) Gray (0.0...1.0) 0.0 is black Lab (0...100,-128...127,-128...127) RGB (%) RGB (0...255) RGB (0.0...1.0)
name	<any proper string>	
value	array of numbers, length according to color space	
overprint	On Off Unchanged	
applyto	None Images VectorAndText	None All images All vector and text objects

Map colors using script variables



Name	Possible values	Corresponds to ("Map colors")
source		
model	CMYKPercent CMYKZeroToOne GrayPercent GrayZeroToOne RGBPercent RGBZeroTo255 RGBZeroToOne	CMYK (%) CMYK (0.0...1.0) Gray (%) Gray (0.0...1.0) 0.0 is black RGB (%) RGB (0...255) RGB (0.0...1.0)
value	array of numbers, length according to color space	
tolerance	number	
intermediate	true false	
destination		
model	Automatic CMYKPercent CMYKZeroToOne GrayPercent GrayZeroToOne RGBPercent RGBZeroTo255 RGBZeroToOne	Use source color CMYK (%) CMYK (0.0...1.0) Gray (%) Gray (0.0...1.0) 0.0 is black RGB (%) RGB (0...255) RGB (0.0...1.0)
value	array of numbers, length according to color space	
applyto	None Images VectorAndText	None All images All vector and text objects
spotcolor		
create	true false	
name	<any proper string>	
tintvalue	number	

Name	Possible values	Corresponds to ("Map colors")
overprint	On Off Unchanged	
keepalternatecolor	true false	

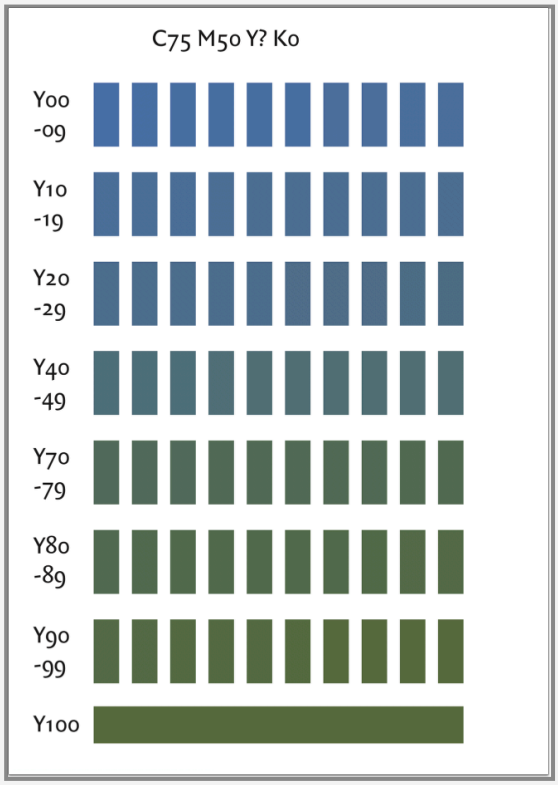
Example: Convert colors using wildcards

To give an example we have created a Fixup to convert colors which allows for using a "wildcard" in one colorant. That means you could convert e.g. all colors using C50 M50 and K50 to something else, keeping all Y values as they are.

Map_colors_with_a_wildcard.kfpx

This PDF has a number of color patches, all using C75, M50, K0 and varying values for Y.

Text_Script_Mapping_original.pdf



If you apply the Fixup and enter C75, M50, K0 for the input values of the respective colorants and a "?" (wildcard) for Y you can convert C,M and K to whatever values you enter (the output value for Y does not matter, the original values will be kept).

In this example we have used C0, M95, K10 for output.



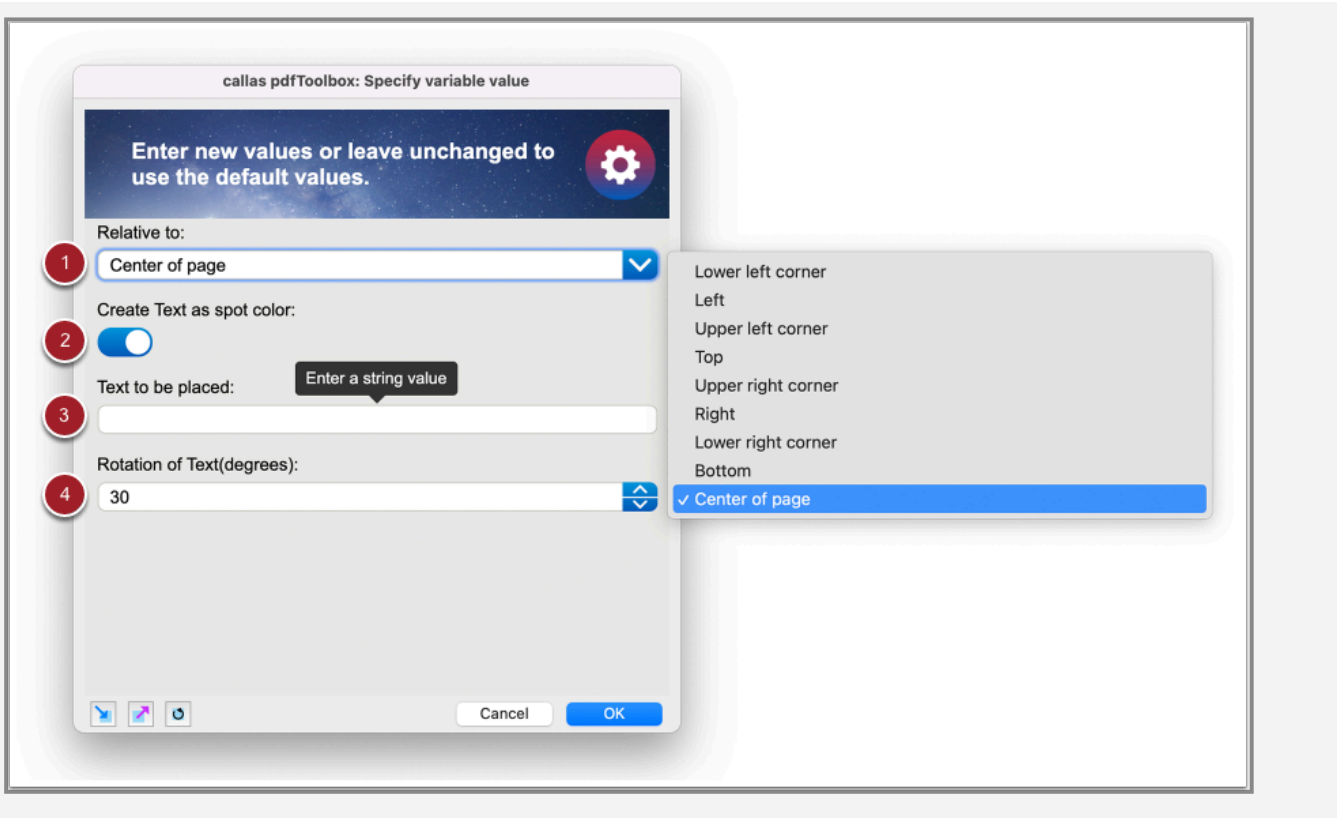
One more thing:

You may as well use more than one wildcard ("?"). The problem is that it will then process much longer, in one test that we made it took roughly 30 minutes. So you should only do so if you have time...

17.17 Ask-at-runtime Dialog: Introduction

When simple variables are used in a Check, Fixup, Profile or Process Plan, pdfToolbox Desktop brings up the Ask-at-runtime dialog before running the Check, Fixup... In this dialog you can modify the values for all variables (initially shown are the default values for each variable) before Profile execution.

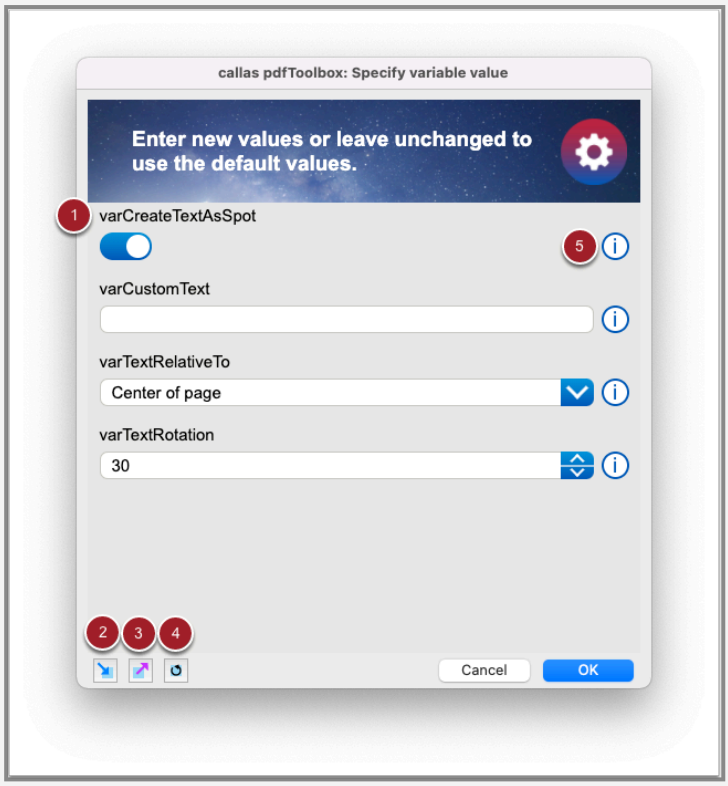
Depending on the value type of the Variable, the Ask-at-runtime dialog will display different input fields and controls. Here is an example of an Ask-at-runtime dialog with four different "Value types":



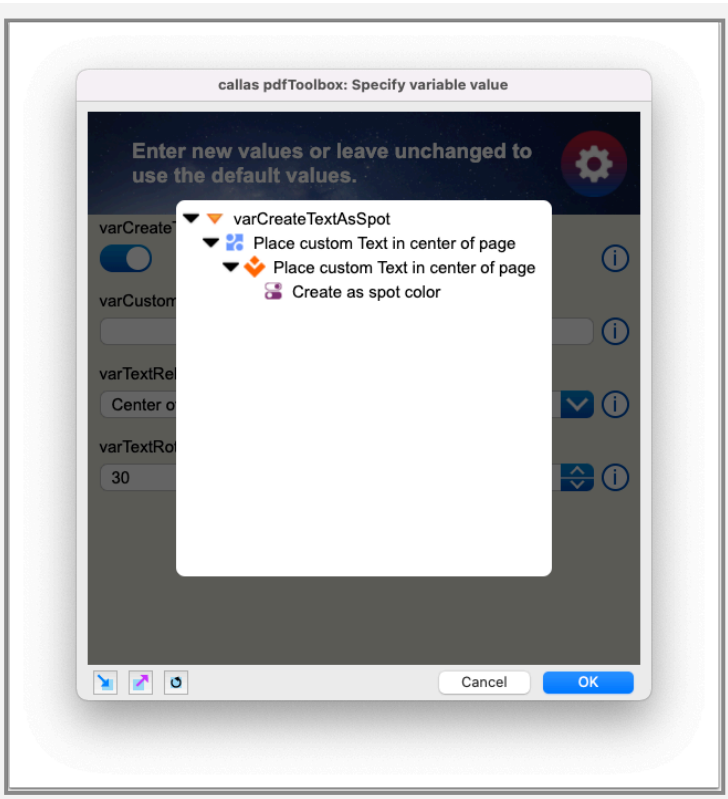
1. Value type **PopUp**: a PopUp is displayed with all the options valid for this parameter.
2. Value type **Boolean**: if a variable is created for a checkbox, a button is displayed in the Ask-at-runtime dialog to enable or disable the checkbox.
3. Value type **String**: a text field is displayed in which any string can be entered.

4. Value type **Float**: any floating point number can be entered (you can either use the controls or enter the number manually).

Tipps and Tricks



1. You can toggle between the label and key of a variable in the variable dialog using Strg/Cmd +K. The lower dialog shows the key view of the variable after pressing Ctrl/Cmd + K
2. Import variables: you can import a JSON file with predefined variable values.
3. Export variables: exports all set values for the variables as a JSON file.
4. Restore default values: previously set values can be restored using this button.
5. If Ctrl/Cmd + i is pressed, an info button appears after each variable, indicating for which parameter the variable was created. The screenshot below shows the info for the first variable in the ask-at-runtime dialog above (Create as spot color button).



Adjust the Ask-at-runtime dialog

As of pdfToolbox 15, it is possible to use Java Script to define the order of the variables in the Ask-at-runtime dialog and to hide or disable variables under certain conditions. All infora-tion about adjusting the Ask-at-runtime dialog can be found in this article: Adjust the Ask-at-runtime dialog.

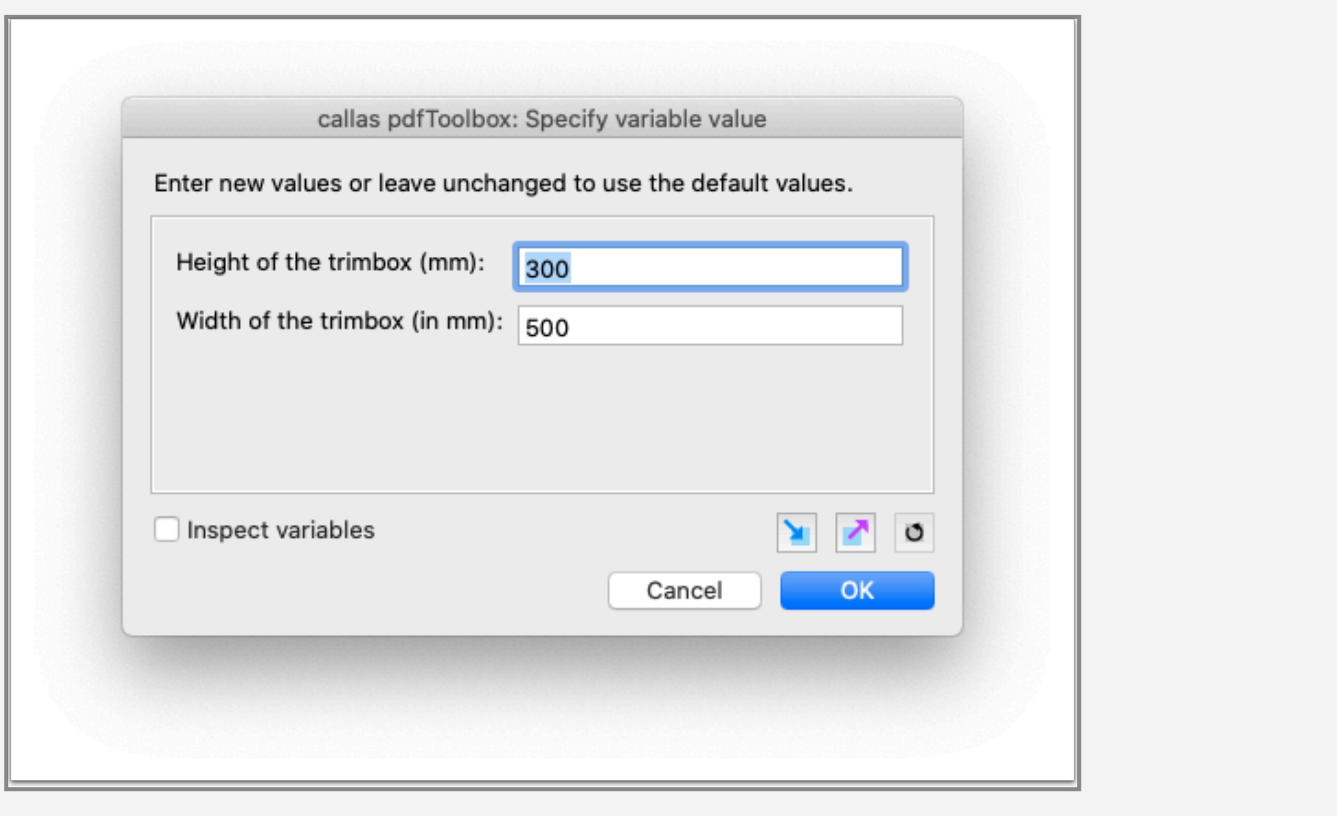
Customizing the Ask-at-runtime dialog

The Ask-at-runtime dialog uses a HTML template that makes changes much easier. All information about customizing the HTML template can be found in this article: Working with ask-at-runtime templates.

Ask-at-runtime dialog before pdfToolbox 11

Before the release of pdfToolbox 11 the Ask-at-runtime dialog looked different and had the following limitations :

- The dialog always looked the same and could not be customised in any way.
- The order of the variables in the dialog window could not easily be modified.
- Any kind of additional grouping or structure, or any modification in the type of input fields used was impossible.
- Only the most basic validation was available.



17.18 Adjust ask-at-runtime dialog

If a Profile, Check, Fixup or Process Plan contains variables, pdfToolbox Desktop displays the ask-at-runtime dialog before executing the profile to specify the variable values. As of pdfToolbox 15 it is possible to adjust the ask-at-runtime dialog using Java Script:

- Adjust order of variables in ask-at-runtime dialog
- Hide / disable variables in ask-at-runtime dialog

Adjust order of variables in ask-at-runtime dialog

By default, the ask-at-runtime dialog displays the variables in alphabetical order according to their variable keys. For example, if a preflight check has three variables specified with the following variable keys "width", "height", "unit", the ask-at-runtime dialog will display them in the following order:



To set the order of the variables in tha ask-at-runtime dialog the JavaScript object `app.context.dialog.order` can be used. You can insert the Java Script object by clicking

on the blue info icon next to the script editor (Insert JavaScript objects…).

`app.context.dialog.order` is an Array containing the variable keys. The first key in the array is asked first in the dialog. If a key is not named in the array, it is automatically placed at the end in alphabetical order:

```
app.context.dialog.order = ["width", "height", "unit"]

OR

app.context.dialog.order = ["width", "height"]
```

Click to copy

After setting the order with `app.context.dialog.order`, the ask-at-runtime dialog will display the variables in the specified order:



Example to download

Check page size (var).kfpx

(earliest version with full support is **pdfToolbox 15**)

Hide / disable variables in ask-at-runtime dialog

By default, the ask-at-runtime dialog always shows all variables. In some cases, it is useful to hide or disable variables under certain conditions. For example, if you have a process plan where you can choose whether or not to create a bleed, the bleed-specific variables should only be adjustable/displayed if the "Create Bleed" Fixup is enabled.



To disable or hide Variables in an ask-at-runtime dialog the JavaScript object `app.context.dialog.state` can be used. You can insert the Java Script object by clicking on

the blue info icon next to the script editor (Insert JavaScript objects…):

```
//to disable variables

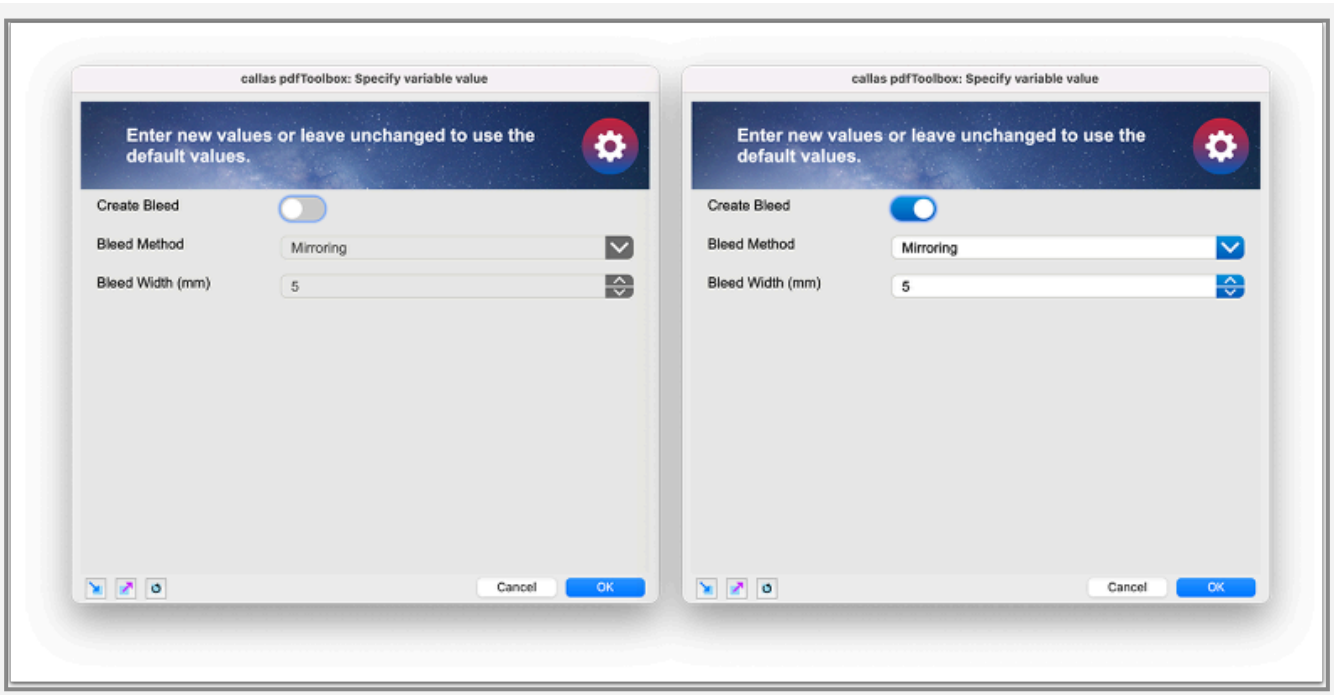
app.context.dialog.state = {
        "method": {
                "disable_if": {
                        "create_bleed": false,
                }
        },
        "bleed_width": {
                "disable_if": {
                        "create_bleed": false,
                }
        }
};


//to hide variables

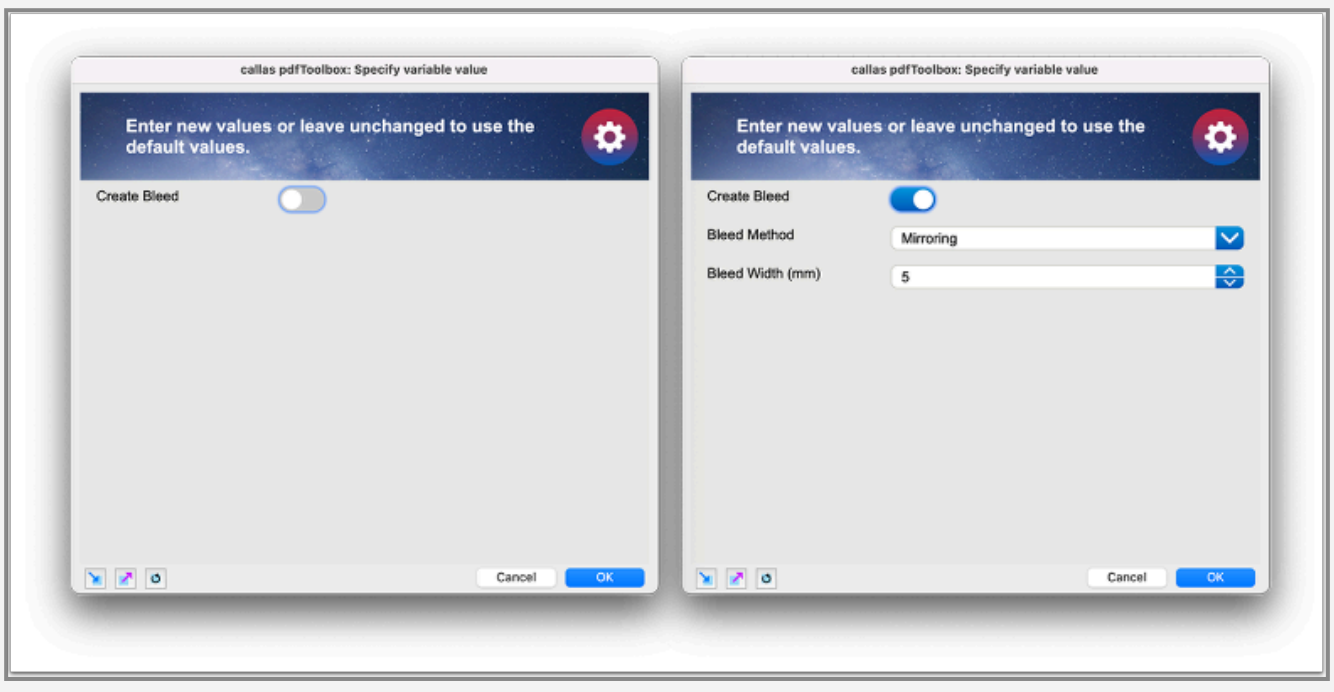
app.context.dialog.state = {
        "method": {
                "hide_if": {
                        "create_bleed": false,
                }
        },
        "bleed_width": {
                "hide_if": {
                        "create_bleed": false,
                }
        }
};
```

Click to copy

If `"disable_if"` is used in `app.context.dialog.state`, the specified variables are grayed out and cannot be modified in the ask-at-runtime dialog unless the "Create Bleed" fixup is enabled:



If `"hide_if"` is used in `app.context.dialog.state`, the specified variables are not displayed in the ask-at-runtime dialog unless the "Create Bleed" fixup is enabled:



Example to download

Generate bleed (var).kfpx

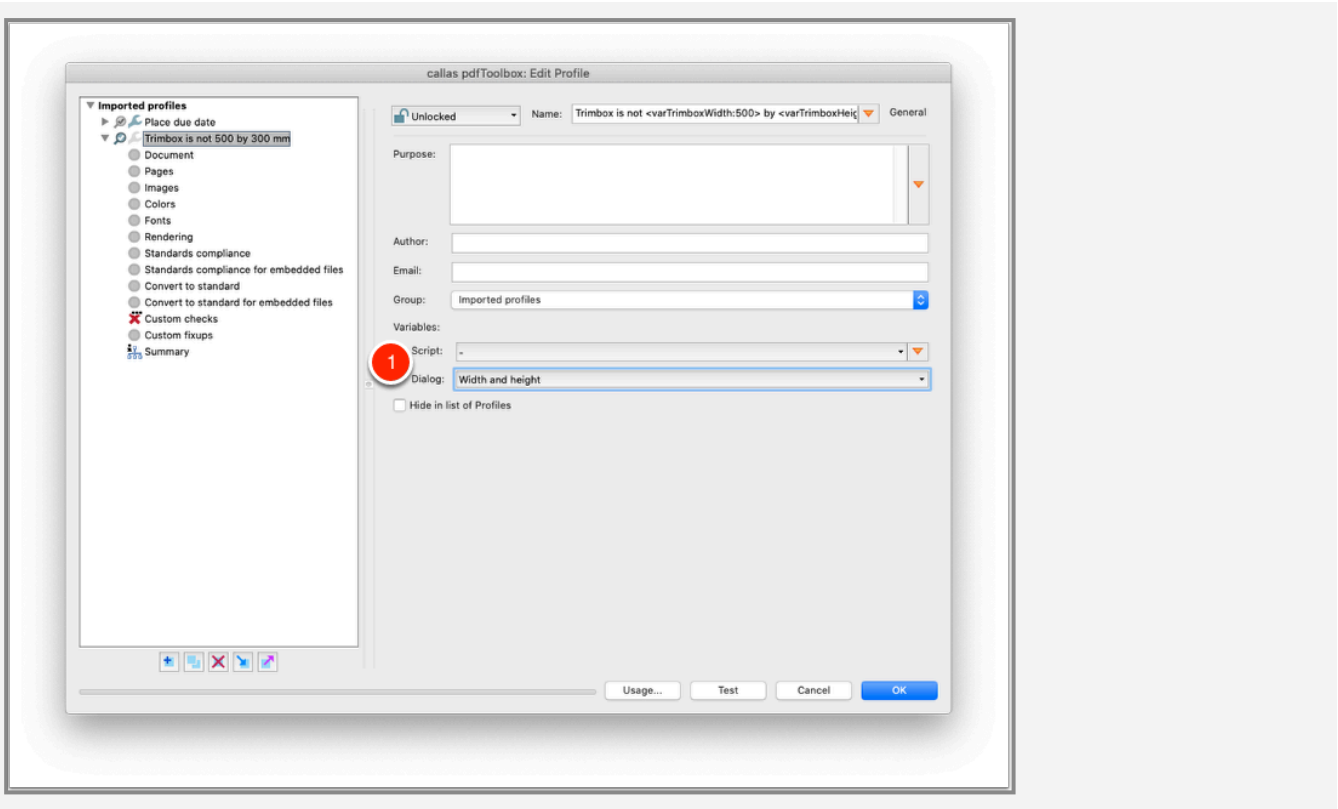
(earliest version with full support is **pdfToolbox 15**)

17.19 Working with ask-at-runtime templates

The default template for the ask-at-runtime dialog window is embedded in the pdfToolbox Desktop executable and shouldn't be modified. While this article doesn't cover editing templates, it does describe how to select them, create a new one to start modifications from or find the templates to edit them.

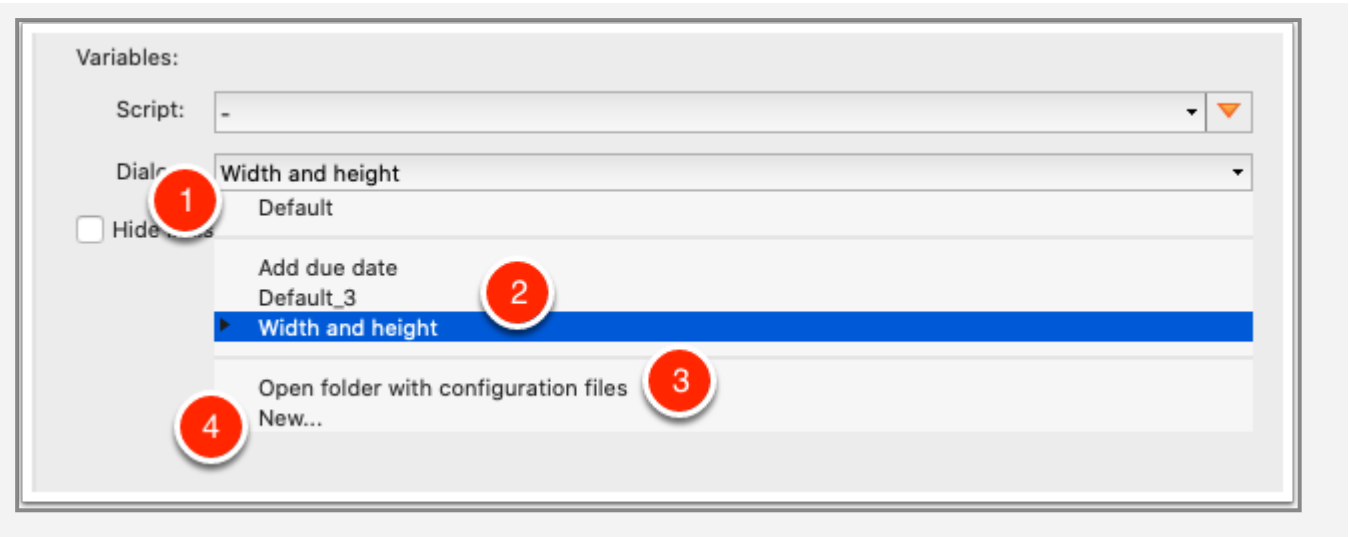
Selecting a template to use

Checks or fixups always use the default template. When creating a profile, it is possible to select the profile that will be used. Look for the Dialog property under the general profile properties (Labeled (1) in the window shown below).



Available templates

Opening the "Dialog" pull-down menu reveals the following choices:



1. Select "Default" to use the default pdfToolbox Desktop profile for this profile.
2. This section displays the currently available templates and lets you choose any of these templates for this profile.
3. "Open folder with configuration files" opens the folder where the current templates (see (2)) are stored. Use this to edit a template or to create a copy.
4. "New..." creates a new template. The template will be a copy of the default pdfToolbox Desktop template. You will be able to find it in the folder with configuration files (see option (3)).

Debugging a template

It is possible to connect the Chrome debugger to a running instance of the ask-at-runtime dialog window. In order to do so, you need to install an additional JSON file in the pdfToolbox Desktop preferences folder first. Locate the "Settings" folder in the pdfToolbox preferences folder.

The preferences folder can be found here:

- On Mac: */Users/<USERNAME>/Library/Preferences/callas software/callas pdfToolbox <VERSION>*
- *On Windows: C:\Users\<USERNAME>\AppData\Roaming\callas software\callas pdfToolbox <VERSION>*

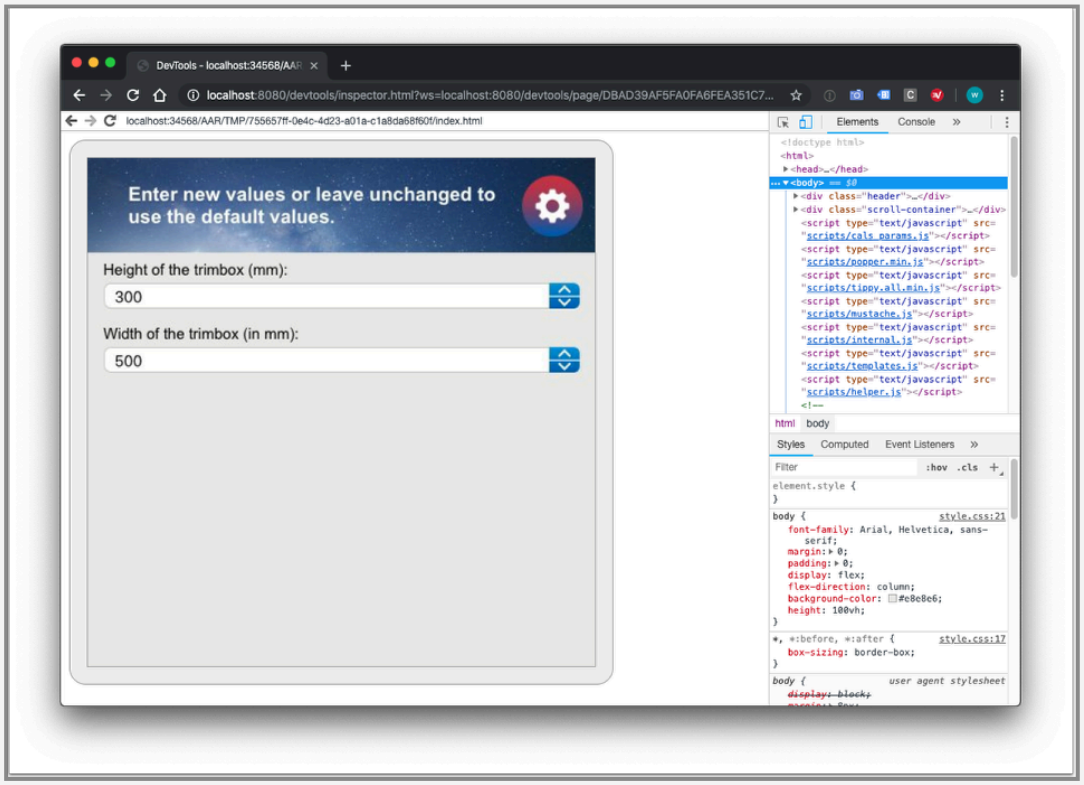
In this settings folder, copy the attached JSON file:

extras.json

Make sure the file is called "extras.json" and is placed inside the "Settings" folder. After installing it, restart pdfToolbox Desktop if it was running. Then open an ask-at-runtime dialog window and with it open, switch to Google Chrome. In Chrome, surf to:

http://localhost:8080

The port number can be modified in the "extras.json" file if desired.



Doing this connects the Chrome debugger to the running ask-at-runtime dialog window. Making changes to the CSS will now be reflected in the ask-at-runtime dialog window inside of pdfToolbox Desktop in real time!

The following tutorial will show you how to customize the ask-at-runtime dialog using Html, CSS and Javascript:

17.20 Arbitrary JavaScript controlled Fixups

Why this new type of Fixup?

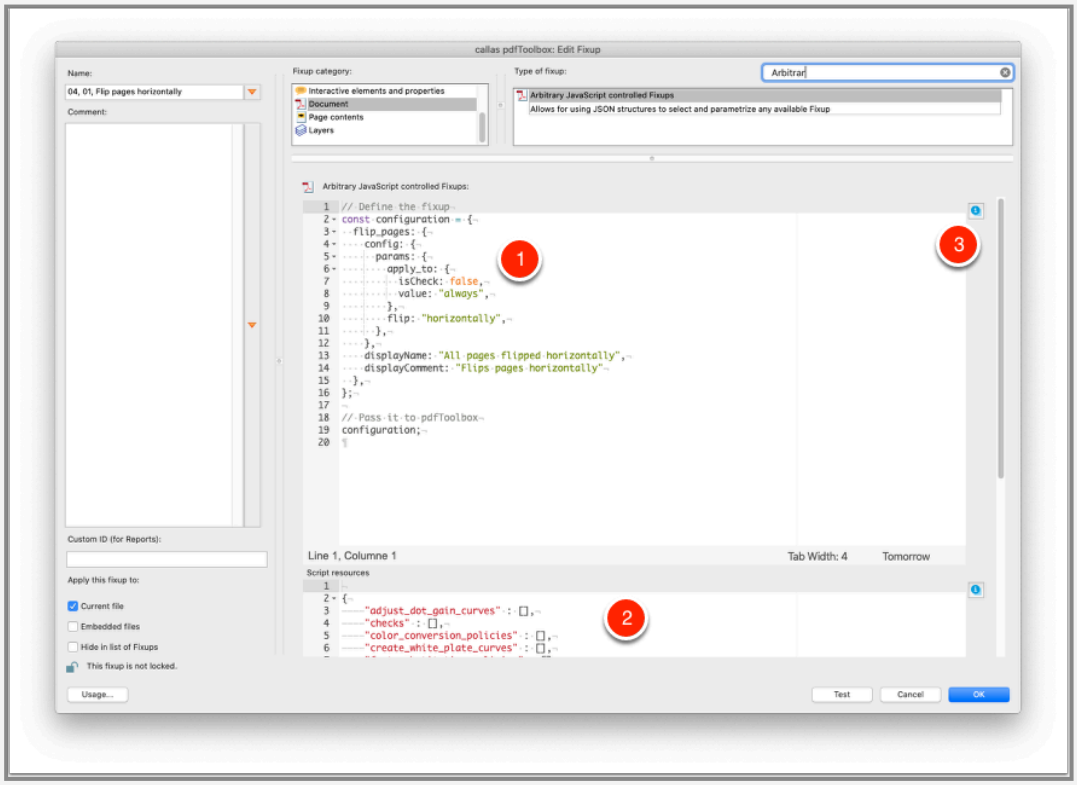
Sometimes you need to do calculations or you want to use lots of variables while configuring one or more Fixups in pdfToolbox. The "Arbitrary JavaScript controlled Fixups" provides a powerful alternative to using regular Fixups in this case.

The Fixup works by providing you with two JavaScript code areas where you can build a configuration for one or more regular Fixups. pdfToolbox then builds the Fixups from the JavaScript objects you provide, and executes them as part of a Profile or Process plan. You shouldn't see this as an easy replacement of standard Fixups, but as an alternative when using regular Fixups would provide very difficult; a good example would be doing color replacement without knowing ahead of time how many colors you want to replace.

Setting up the Fixup

The Fixup contains two JavaScript areas in order to setup it up:

1. The main JavaScript area. Here, you should create a configuration object and pass it back to pdfToolbox.
2. The resource area. Here you must list any resources your JavaScript created Fixup requires so they can be properly managed by pdfToolbox.
3. Info button which provides help on what to enter in the configuration area. The different possibilities are explained below.



Some property names have been changed in pdfToolbox 13. The old values are still accepted during execution, but if inserted by one of the menu items (“Insert configured fixup…” ec.pp.) the new property names are used. If both properties are present the new property name takes precedence.

pdfToolbox 12	pdfToolbox 13
displayName	name
displayComment	comment
customID	custom_id

In pdfToolbox 13, additional Fixup settings have been added to the JSON serialisation:

```
  "scope" : {
     "current_file" : true,
     "embedded_files" : false
  }
```

Click to copy

The main JavaScript area

pdfToolbox expects you to return a JavaScript object containing one or more Fixups. The standard form could look something like this:

```
// The object containing all of the Fixup definitions
const configuration = {

        // A fixup
        flip_pages: {
                config: {
                        // fixup specific setup
                }
                displayName: "The name to be used in the report",
                displayComment: "The comment sometimes also shown in reports"
        },

        // Optionally a second, third... fixup

};

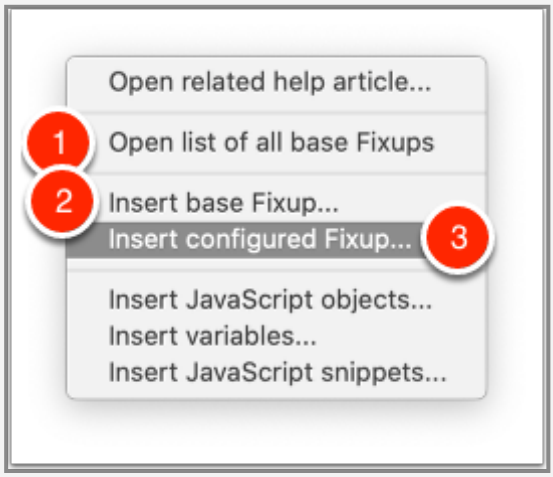
// This line makes sure it is passed back to pdfToolbox. Do not use "return" here.
configuration;
```

Click to copy

Some remarks about the code above:

- "flip_pages" is the identifier for the Fixup we want to use.
- "config" is where you add the properties for the specific Fixup you are setting up. As each Fixup is different, this section typically will be different for each Fixup.
- "displayName" and "displayComment" are optional, but allow you to build the name and comment used in pre-flight reports when this Fixup is run.

So how do you find out what to put in the config area? Click on the blue "i" button and you have three possibilities:



1. **Open list of all base Fixups**
   This opens documentation on the setup required for all Fixups. Read more in this article.
2. **Insert base Fixup**
   This opens the "create Fixup" window. Now you can create a new Fixup with all specific properties. After you click the "Ok" button, the Fixup is inserted into the JavaScript editor.
3. **Insert configured Fixup**
   This shows you a list of all Fixups of the current library and lets you choose one. pdfToolbox then adds the Fixup with all defined properties into the JavaScript editor.

Things to look out for

While configuring this Fixup is relatively straightforward, there are still some things to be cautious about...

- pdfToolbox expects JavaScript code, not a JSON object. You'll have to write some JavaScript statements that produce a "result". This result should then contain the setup for the Fixup. You can accomplish this in different ways, but pdfToolbox has no preference - your code simply has to produce a result.
- You can create multiple Fixups in this JavaScript structure. You can't however have the same type of Fixup twice. If I inserted a "flip_pages" Fixup, I can't add another of those to my JavaScript structure. While there are

way to cheat pdfToolbox by inserting something like this into your JavaScript, it's not going to work when you execute the Fixup.

- Some Fixups internally allow multiple configurations to be setup (using the little "+" and "-" buttons in the user interface when setting up the Fixup). Such Fixups can be configured as well, and in this case the "config" property in your JavaScript structure will be an array. If you're unclear how to do this, create an example Fixup and then use "Insert configured Fixup" to have pdfToolbox show you the way.

Example

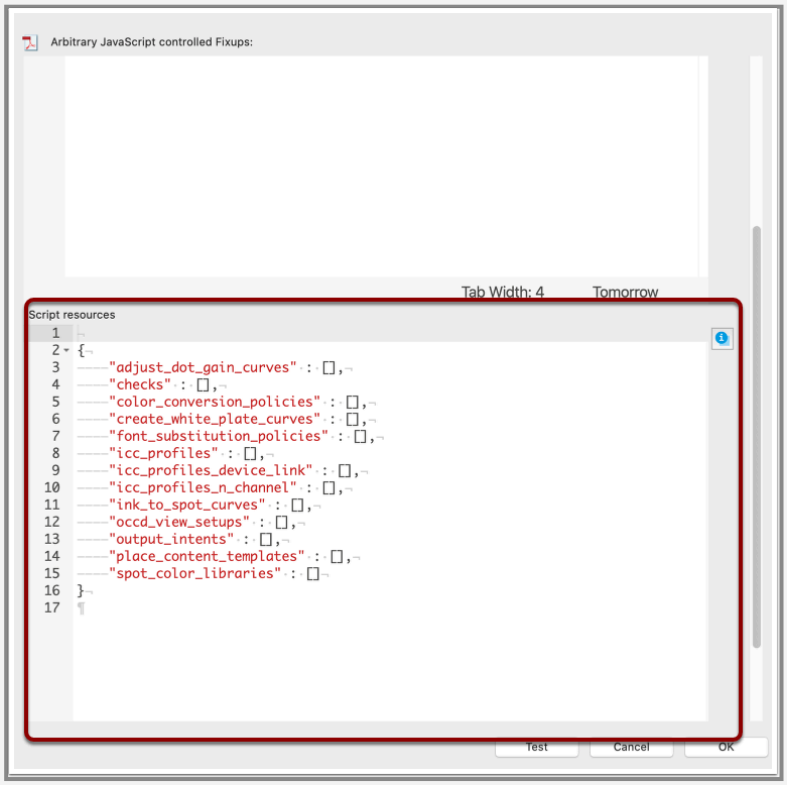
A simple example to mirror pages horizontally is included here and can be imported into your library. However, with the "Insert configured Fixup" option it's easy to create your own examples. All that's left afterwards is to adjust them to your taste!



04__01__Flip_pages_horizontally.kfpx

17.21 Referencing resources in arbitrary JavaScript controlled Fixups

Fixups may use resources, such as ICC profiles or - most importantly - Checks that are used as filters. In order to use such resources in the JavaScript structure of a Fixup they have to be listed in the second input field of the Fixup.

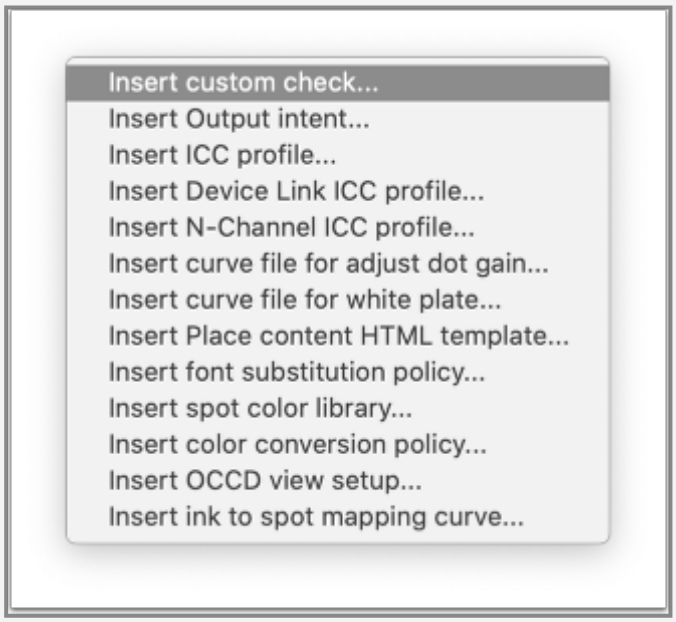


If you are using the method 3. "Insert configured Fixup" described in Arbitrary JavaScript controlled Fixups all configured resources are automatically added. However, even then you may want to add more resources. That is when you want to change what resources are used during runtime. You may for instance want to change the destination ICC profile of a color conversion during runtime via JavaScript. You could then put all ICC profiles that may be used into the "icc_profiles" section of the list and can then reference any of them in the JavaScript.

Adding resources

In order to add any item to the list of resources you would again use the blue info button.





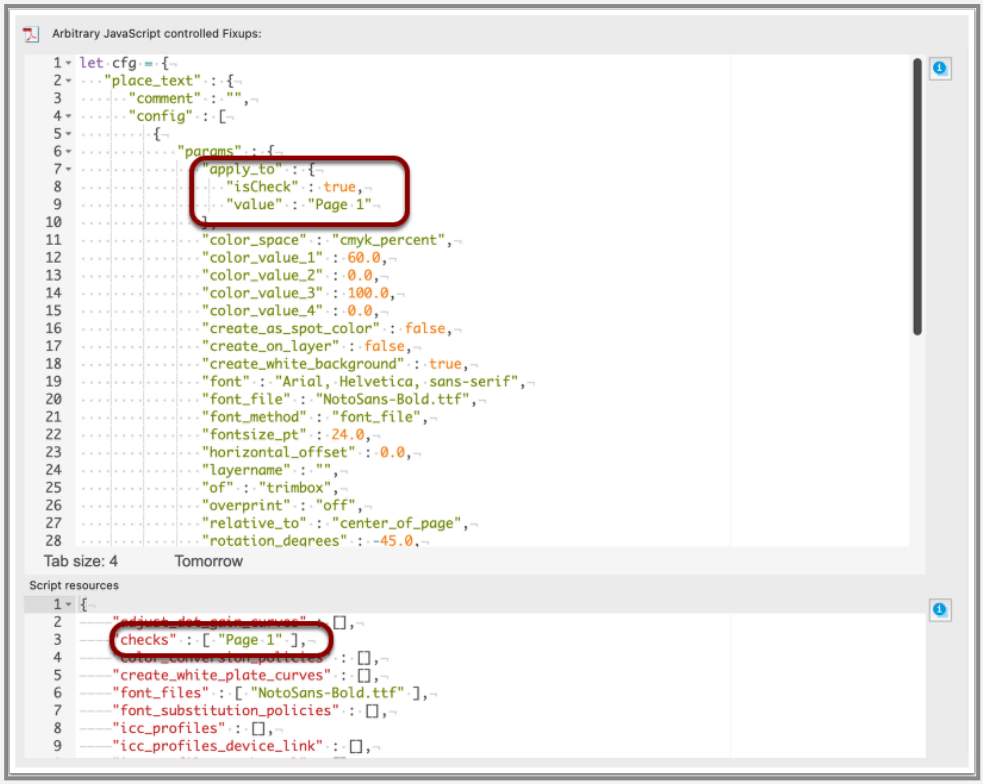
Using Checks for filters

A special case are Checks that can be used in an "Apply to" filter. We use an example to demonstrate how that can be done.

This Fixup places some text on the first page of a PDF file.

Place text "Page 1" on page 1.kfpx

When you create a new Fixup based on "Arbitrary JavaScript controlled Fixups" and import this configured Fixup via the blue info button you see this structure:



The Check for "Page 1" is automatically added to the Script resources and in the params section there is an entry "apply_to" that references that resource. If you would have more than one Check in the resources you could select the one you want to use in your JS code.

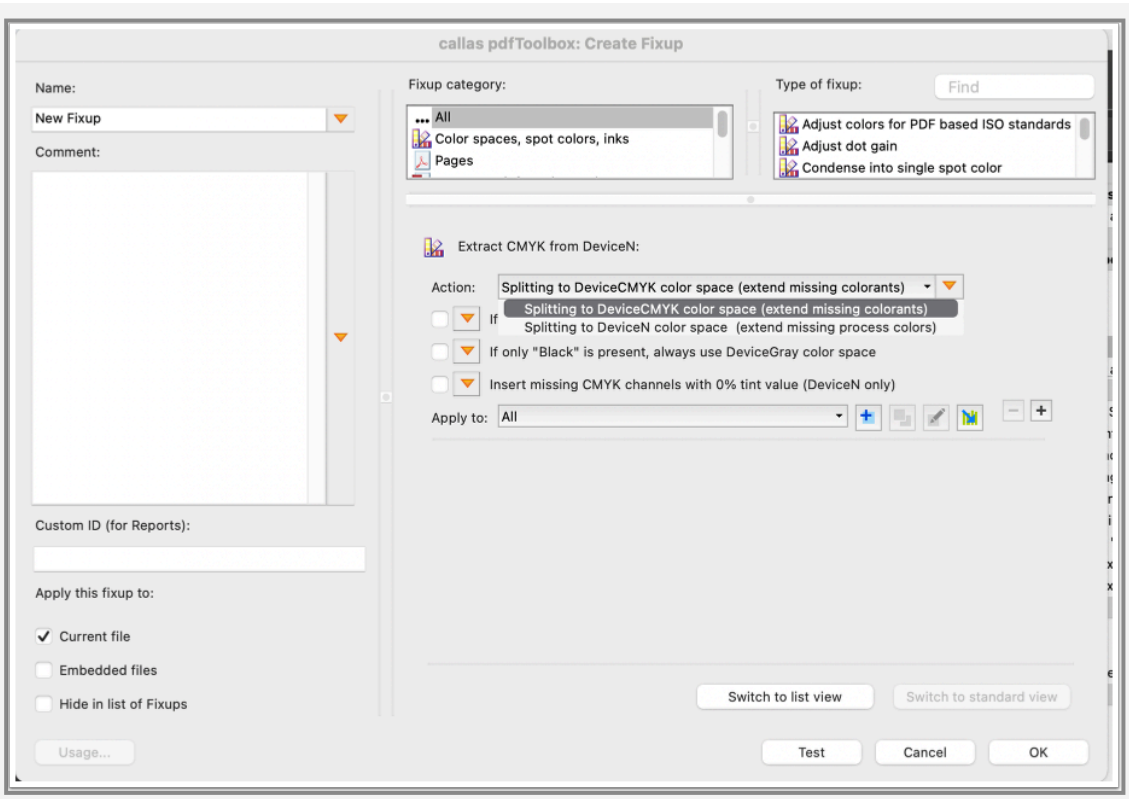
The "isCheck" parameter serves a very special use case: It should almost always be "true". Only if the filter of the Fixup has predefined values (e.g. "All" or "Images") it may be set to "false".

Explanation: If there is a string in "value" that means it is not a JavaScript structure (as explained here). It could then either be a Check in the resources of the JS based Fixup or it could be such a predefined value. In the latter case you would tell the engine by setting "isCheck" to "false" that the value should be treated as a value rather than a Check name.

But you can be even more dynamic, you could as well define the whole "Apply to" Check using JavaScript. Read Using JavaScript in an (apply to) filter in order to find out how.

17.22 Using JavaScript for pop up values in "Arbitrary JavaScript controlled Fixups"

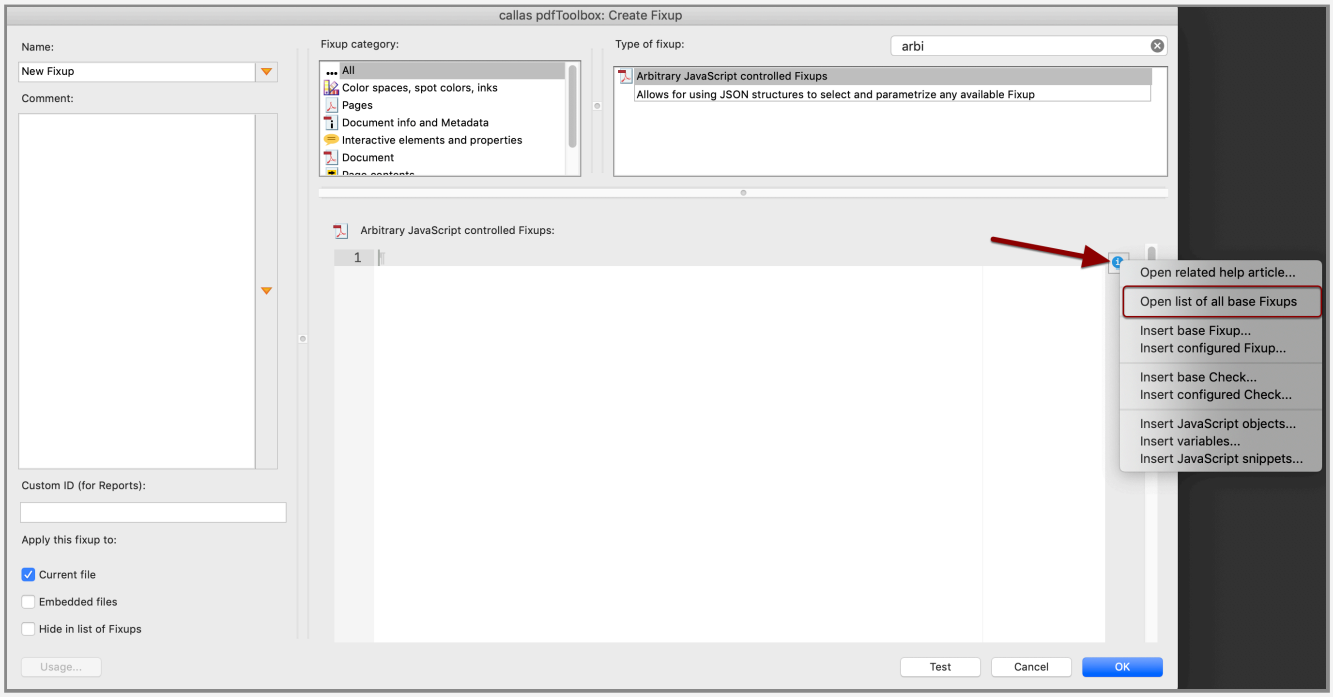
When creating your Fixups using "Arbitrary JavaScript controlled Fixups", you may also want to know what internal values may be used for a pop up control in the "regular" Fixup.
In this article, we use "Extract CMYK from DeviceN".



There is a pop up "Action" with two possible values.

- Splitting to DeviceCMYK color space (extend missing colorants)
- Splitting to DeviceN color space (extend missing process colors)

In order to find out what internal strings you may use in "Arbitrary JavaScript controlled Fixups", you use the blue info icon in that Fixup and select "Open list of all base Fixups".



It will open a list of all existing Fixup properties in your web browser. There you would (as in our example above) search for "extract_cmyk_from_devicen" which would then show you something like:



```
"extract_cmyk_from_devicen" :
{
        "comment" : "",
        "config" :
        [
                {
                        "params" :
                        {
                                "action" :
                                {
                                        "hasApplyTo" : false,
                                        "label" : "Action:",
                                        "value" : "spit_to_devicecmyk",
                                        "value_type" : "value_list",
                                        "values" :
                                        [

                                                {
                                                        "label" : "Splitting to DeviceCMYK color space (extend missing colorants)",
                                                        "value" : "spit_to_devicecmyk"
                                                },

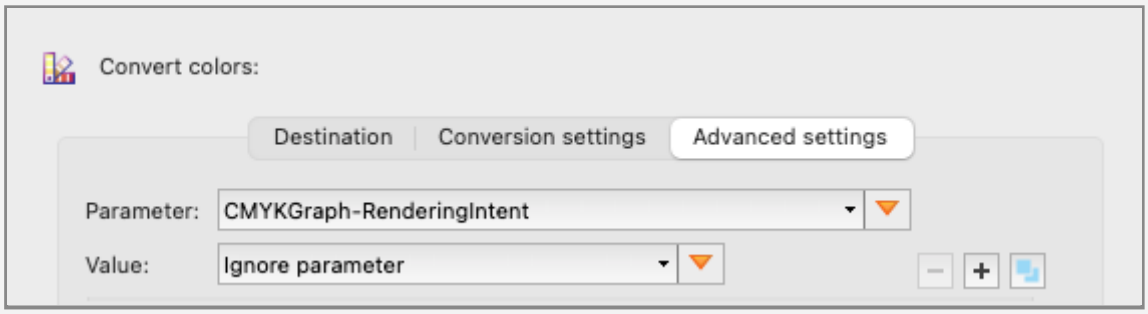
                                                {
                                                        "label" : "Splitting to DeviceN color space  (extend missing process colors)",
                                                        "value" : "spit_to_devicen"
                                                }
                                        ]
                                },
                                "apply_to" :
                                {
                                        "hasApplyTo" : true,
```

This lists the possible values for "action" which are "split_to_devicecmyk" and "split_to_devicen".

Additional information regarding Convert colors

As already mentioned, the list contains all internal values that are possible for a particular Fixup.

Unfortunately, if you want to use the Convert Colors Fixup, the list is not clear about which values are valid for a particular parameter in the **"Advanced setting"** tab.



Therefore, a list of all valid values for each parameter is provided here.

Valid values for "advanced_settings"

Parameter	Value
cmykgraph_renderingintent	"ignore_parameter" "use_rendering_intent_of_docur "relativecolorimetric" "absolutecolorimetric" "perceptual" "saturation" "convert_as_defined_for_color_s
"cmykimage_renderingintent"	"ignore_parameter" "use_rendering_intent_of_docur "relativecolorimetric" "absolutecolorimetric" "perceptual" "saturation" "convert_as_defined_for_color_s
"rgbgraph_renderingintent"	"ignore_parameter"

Parameter	Value
	"use_rendering_intent_of_docur "relativecolorimetric" "absolutecolorimetric" "perceptual" "saturation" "convert_as_defined_for_color_s
"rgbimage_renderingintent"	"ignore_parameter" "use_rendering_intent_of_docur "relativecolorimetric" "absolutecolorimetric" "perceptual" "saturation" "convert_as_defined_for_color_s
"graygraph_renderingintent"	"ignore_parameter" "use_rendering_intent_of_docur "relativecolorimetric" "absolutecolorimetric" "perceptual" "saturation" "convert_as_defined_for_color_s
"grayimage_renderingintent"	"ignore_parameter" "use_rendering_intent_of_docur "relativecolorimetric" "absolutecolorimetric" "perceptual" "saturation" "convert_as_defined_for_color_s
"labgraph_renderingintent"	"ignore_parameter" "use_rendering_intent_of_docur "relativecolorimetric" "absolutecolorimetric" "perceptual" "saturation" "convert_as_defined_for_color_s
"labimage_renderingintent"	"ignore_parameter" "use_rendering_intent_of_docur "relativecolorimetric" "absolutecolorimetric" "perceptual"

Parameter	Value
	"saturation" "convert_as_defined_for_color_s
"cmykgraph_c_eq_m_eq_y_is_black"	"ignore_parameter" "devicegray" "devicecmyk_black" "separation_black"
"cmykimage_c_eq_m_eq_y_is_black"	"ignore_parameter" "devicegray" "devicecmyk_black" "separation_black"
"rgbgraph_r_eq_g_eq_b_is_black"	"ignore_parameter" "devicegray" "devicecmyk_black" "separation_black"
"rgbimage_r_eq_g_eq_b_is_black"	"ignore_parameter" "devicegray" "devicecmyk_black" "separation_black"
"graygraph_setgraycolorspaceto"	"ignore_parameter" "devicegray" "devicecmyk_black" "separation_black" "convert_as_defined_for_color_s
"grayimage_setgraycolorspaceto"	"ignore_parameter" "devicegray" "devicecmyk_black" "separation_black" "convert_as_defined_for_color_s
"labgraph_a_eq_b_eq_0_is_black"	"ignore_parameter" "devicegray" "devicecmyk_black" "separation_black"
"labimage_a_eq_b_eq_0_is_black"	"ignore_parameter" "devicegray" "devicecmyk_black" "separation_black"

Parameter	Value
"cmykgraph_c_eq_m_eq_y_is_black_excludeblendmodes"	"ignore_parameter" "normal" "multiply" "screen" "overlay" "darken" "lighten" "colordodge" "colorburn" "hardlight" "softlight" "difference" "exclusion" "hue" "saturation" "color" "luminosity"
"cmykimage_c_eq_m_eq_y_is_black_excludeblendmodes"	"ignore_parameter" "normal" "multiply" "screen" "overlay" "darken" "lighten" "colordodge" "colorburn" "hardlight" "softlight" "difference" "exclusion" "hue" "saturation" "color" "luminosity"
"rgbimage_r_eq_g_eq_b_is_black_excludeblendmodes"	"ignore_parameter" "normal" "multiply" "screen" "overlay" "darken" "lighten" "colordodge"

Parameter	Value
	"colorburn" "hardlight" "softlight" "difference" "exclusion" "hue" "saturation" "color" "luminosity"
"rgbgraph_r_eq_g_eq_b_is_black_excludeblendmodes"	"ignore_parameter" "normal" "multiply" "screen" "overlay" "darken" "lighten" "colordodge" "colorburn" "hardlight" "softlight" "difference" "exclusion" "hue" "saturation" "color" "luminosity"
"labgraph_a_eq_b_eq_0_is_black_excludeblendmodes"	"ignore_parameter" "normal" "multiply" "screen" "overlay" "darken" "lighten" "colordodge" "colorburn" "hardlight" "softlight" "difference" "exclusion" "hue" "saturation" "color"

Parameter	Value
	"luminosity"
"labimage_a_eq_b_eq_0_is_black_excludeblendmodes"	"ignore_parameter" "normal" "multiply" "screen" "overlay" "darken" "lighten" "colordodge" "colorburn" "hardlight" "softlight" "difference" "exclusion" "hue" "saturation" "color" "luminosity"
"cmykgraph_handleprocesscolorindevicen_separationasdevicecmyk"	"ignore_parameter" "treat_process_colors_in_device "treat_process_colors_in_device
"cmykimage_handleprocesscolorindevicen_separationasdevicecmyk"	"ignore_parameter" "treat_process_colors_in_device "treat_process_colors_in_device
"cmykgraph_advancedcolorconversion"	"ignore_parameter" "no_special_treatment" "apply_dot_gain_difference"
"cmykimage_advancedcolorconversion"	"ignore_parameter" "no_special_treatment" "apply_dot_gain_difference"
"graygraph_advancedcolorconversion"	"ignore_parameter" "no_special_treatment" "apply_dot_gain_difference"
"grayimage_advancedcolorconversion"	"ignore_parameter" "no_special_treatment" "apply_dot_gain_difference"
"compressionmethod"	"ignore_parameter"

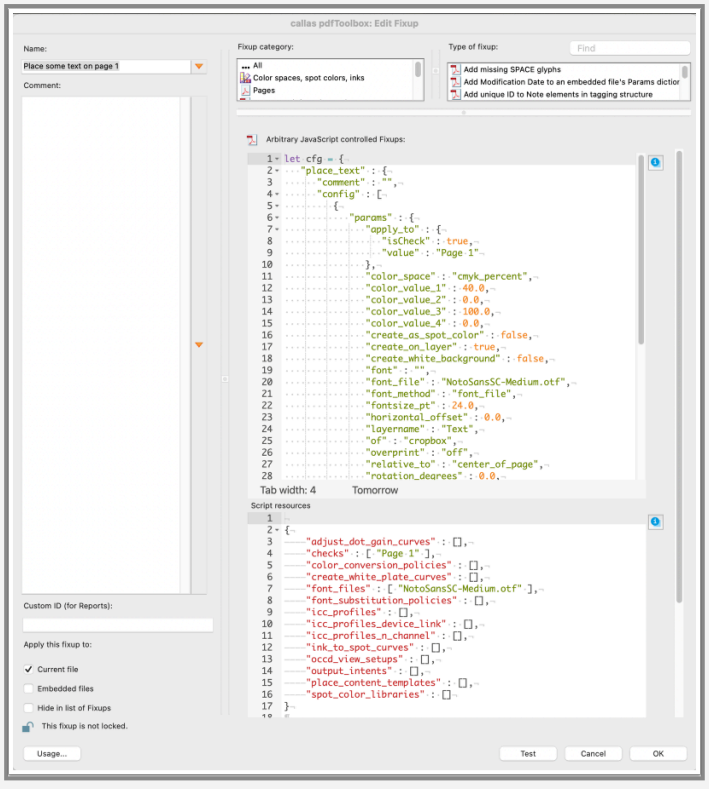
Parameter	Value
	"keep_compression_method" "compress_all_to_zip" "compress_all_to_jpeg"
"destination_jpegquality"	"ignore_parameter" "minimum" "low" "medium" "high" "maximum"
"destination_settransparencyblendspacetodest"	"ignore_parameter" "leave_unchanged" "set_to_destination_icc_profile" "set_to_destination_as_device_c "set_to_destination_if_equal_to_ "ignore_parameter"

17.23 Using JavaScript in an (apply to) filter

Beginning with pdfToolbox 13, you may use JavaScript code for a filter, before it was only possible to use Checks as a (static) resource. This makes creating Fixups even more dynamic.

We are using an example 'Arbitrary JavaScript controlled Fixup' (also attached below) where some text is put at the center of page 1. We want to create a JavaScript structure that replaces the Check that is in the resource "Page 1".

Place_some_text_on_page_1.kfpx

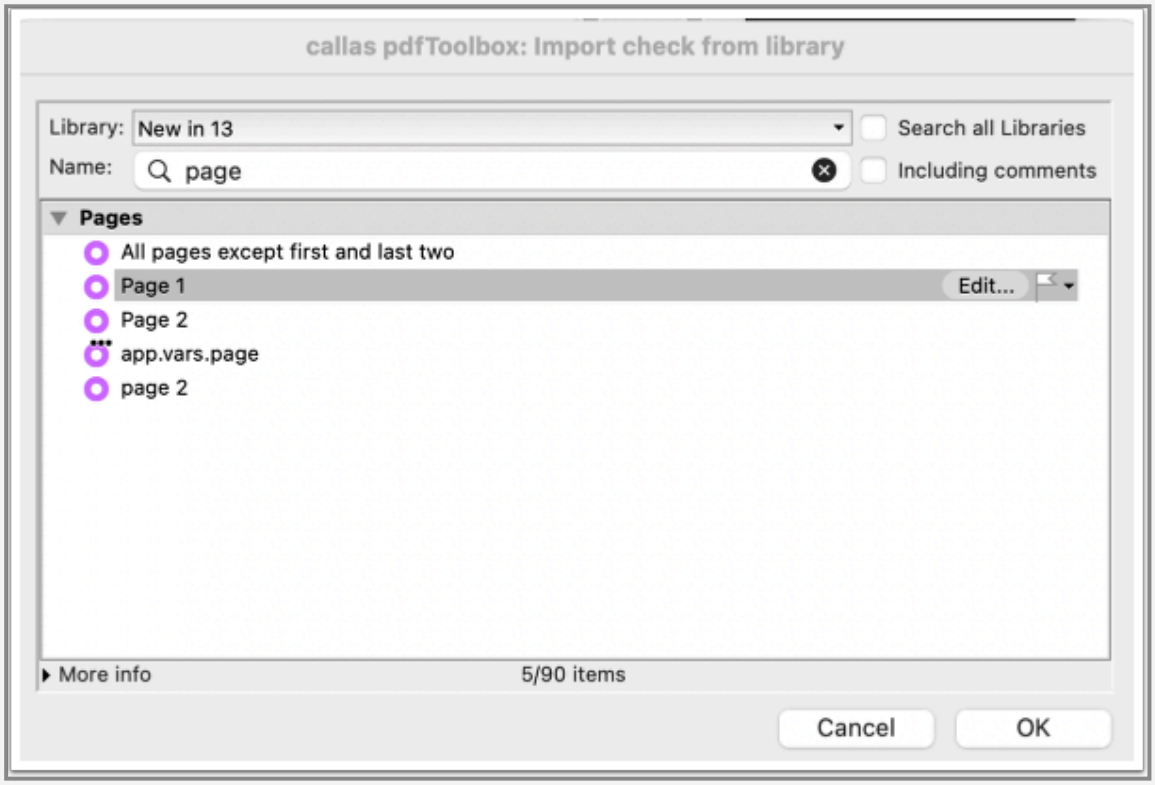


In order to do so:

1. We create some delimiter (comment) at the top of the JavaScript code
2. Place the cursor above it
3. Use the blue 'i' icon and select "Insert configured Check".



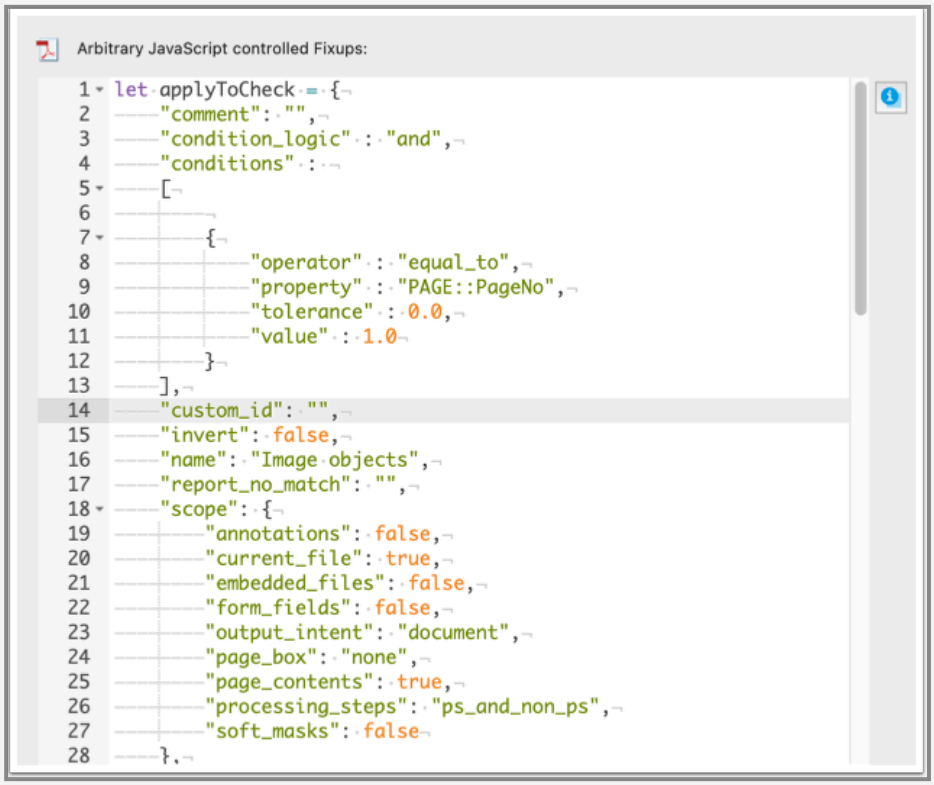
Then we pick the preconfigured Check (or we could use "Insert base Check..." to create one).



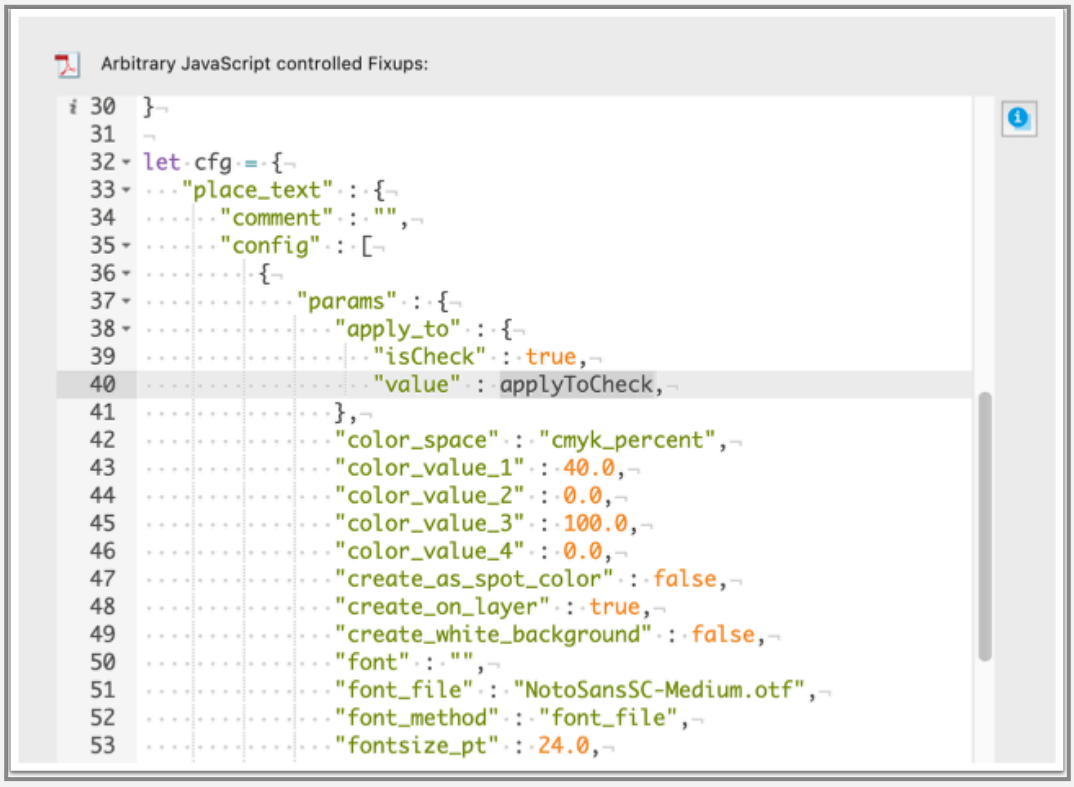
This inserts a JavaScript structure that represents the Check.



We then assign this new structure to a variable, e.g. "apply-ToCheck":



Then we scroll down to the Fixup structure and replace "Page 1" with the name of this variable.



And then the Fixup can be used.

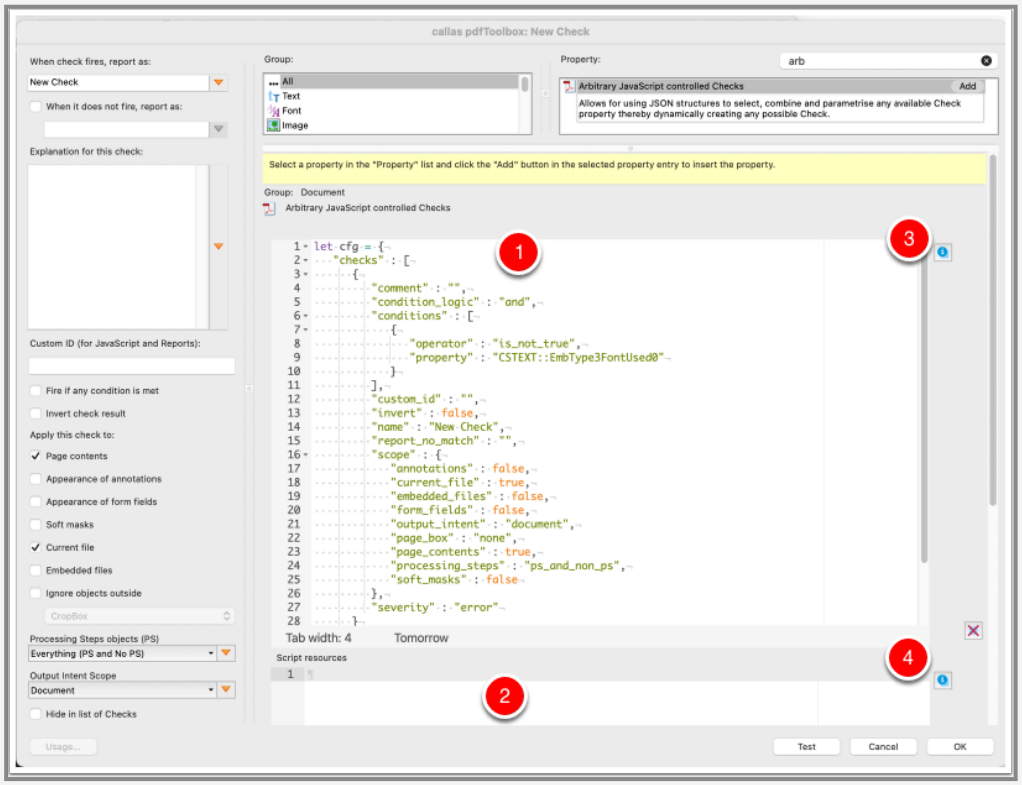
Of course the above makes more sense if the structure is created dynamically.

Place_text_on_page_1_using_JavaScript.kfpx

17.24 Arbitrary JavaScript controlled Checks

pdfToolbox 13 introduces "Arbitrary JavaScript controlled Checks" which works similar to the Fixup "Arbitrary JavaScript controlled Fixups".

Setting up the Check



It has two areas, one for the actual JavaScript configuration (1) and one for the resources (2).

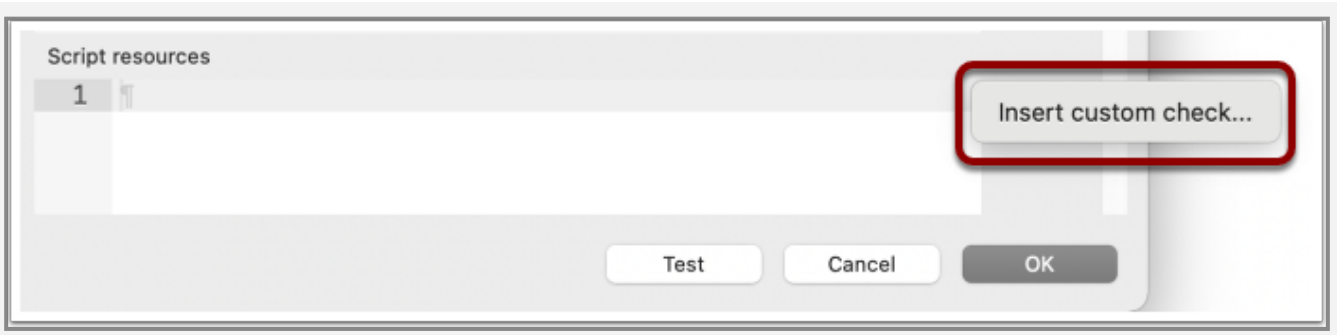
Two info buttons, (3) and (4), at the right side allows to insert preconfigured information.

Resources

While a Fixup may use various types of resources, the only resource that can be used in a Check is another Check.

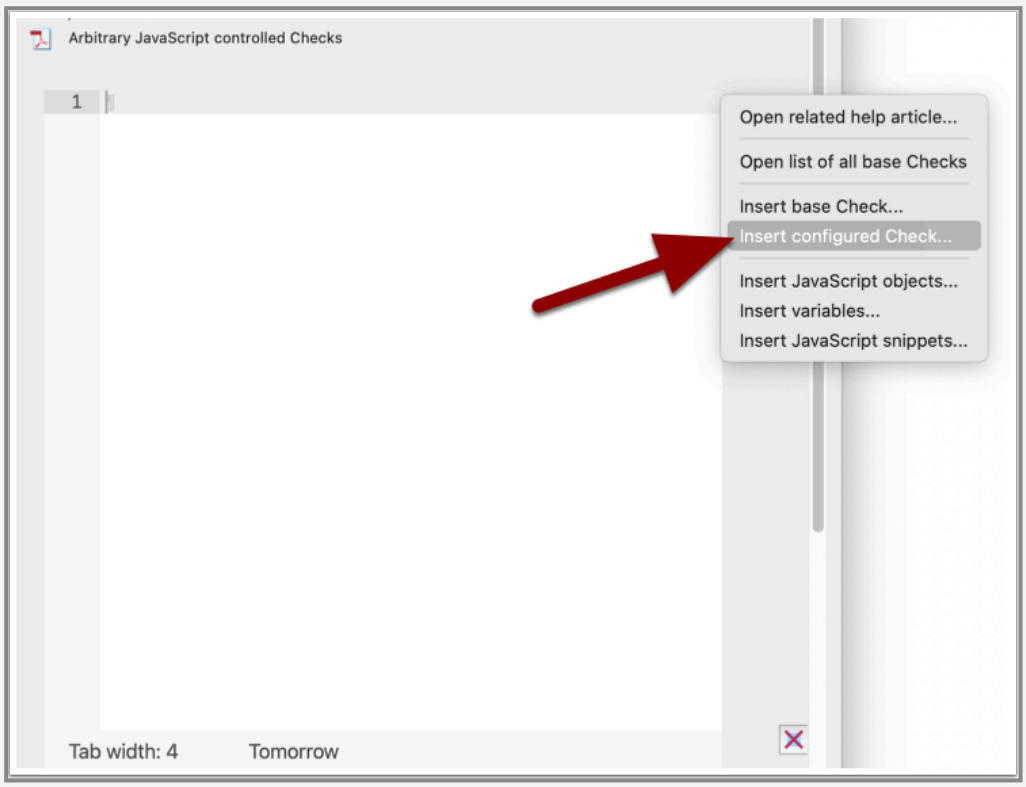
Such resources can be used in "Include other check" and most of the "Context aware object detection" (Sifter) Check properties.

Please note that JavaScript may also be used instead of such a resource.

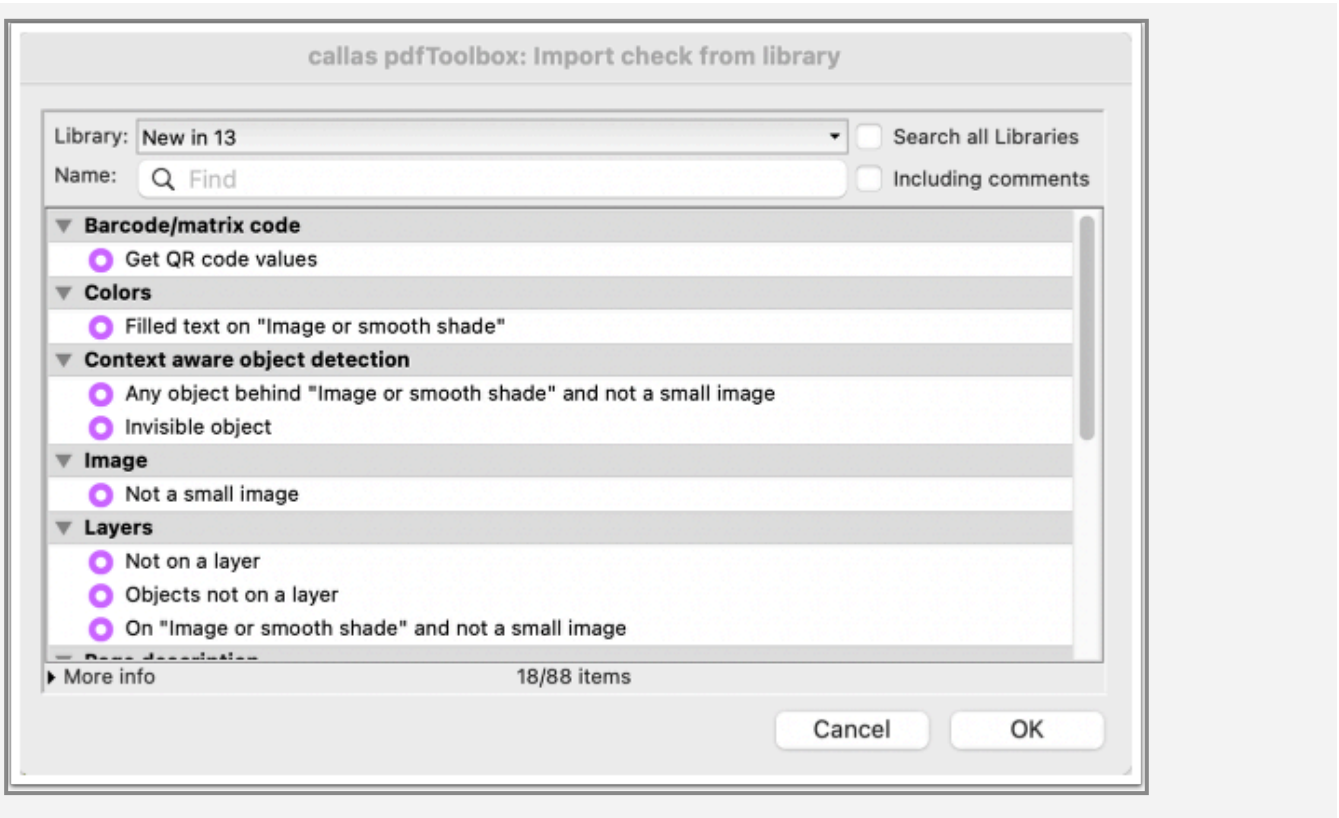


The main JavaScript area

The easiest way to create a JavaScript based Check is to first create it as a regular Check and then insert it as configured Check.

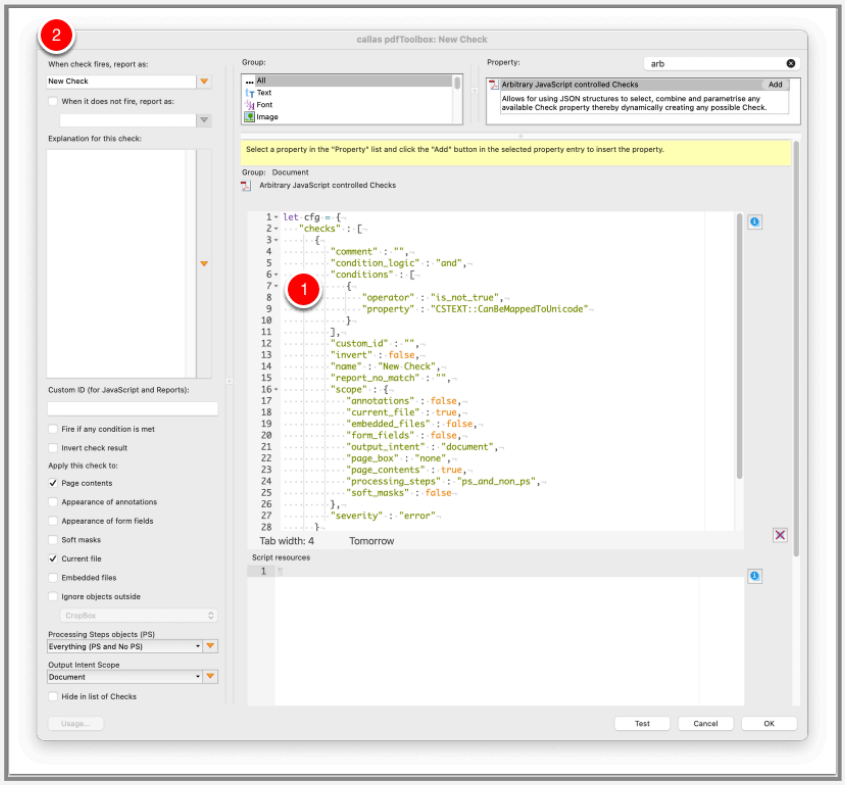


This opens a new dialogue that lets users pick a previously configured Check.



As an alternative, "Insert base Check..." might also be used which lets you configure one Check on the fly.

In both cases, a structure is created that the users can then modify which can be created in a similar way using JavaScript code.



Everything that is specific to this particular Check property is in the conditions array (1).

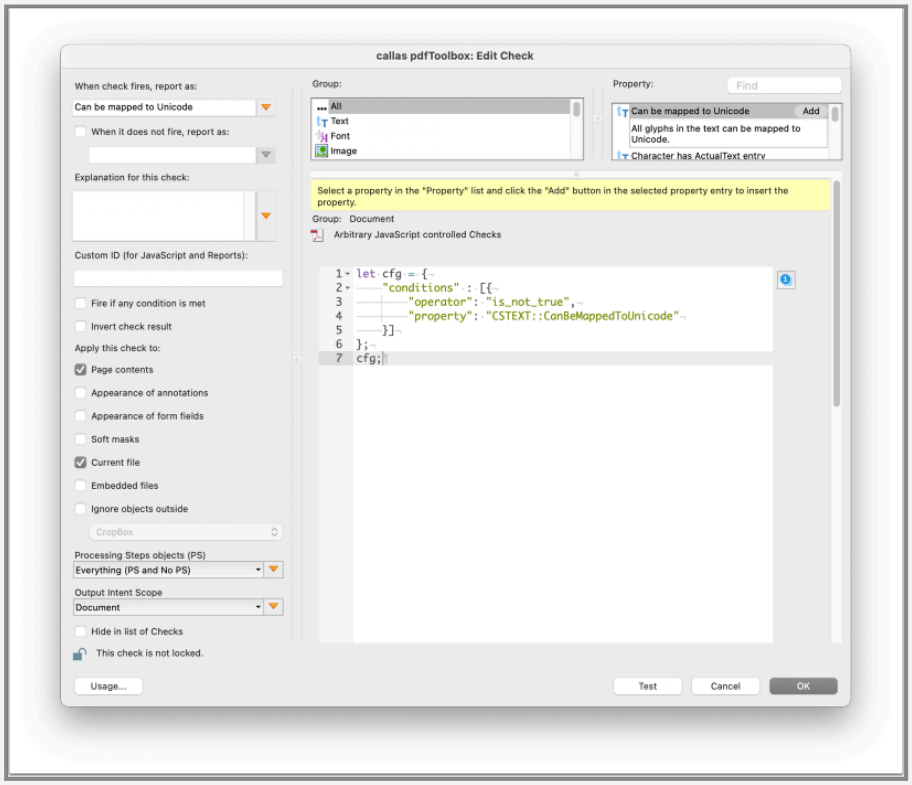
All other parameters represent something that is common to all Checks, basically what is on the left hand side of the Check editor (2). This is the name, the comment etc. E.g. the parameter "Fire if any condition is met" would be reflected by

```
"condition_logic" : "or"
```

Check or condition

In the example above, would everything that is entered on the left side be overwritten by the values in the JavaScript code? Yes, it would.

However "Arbitrary JavaScript controlled Checks" also allows to use the conditions array alone.

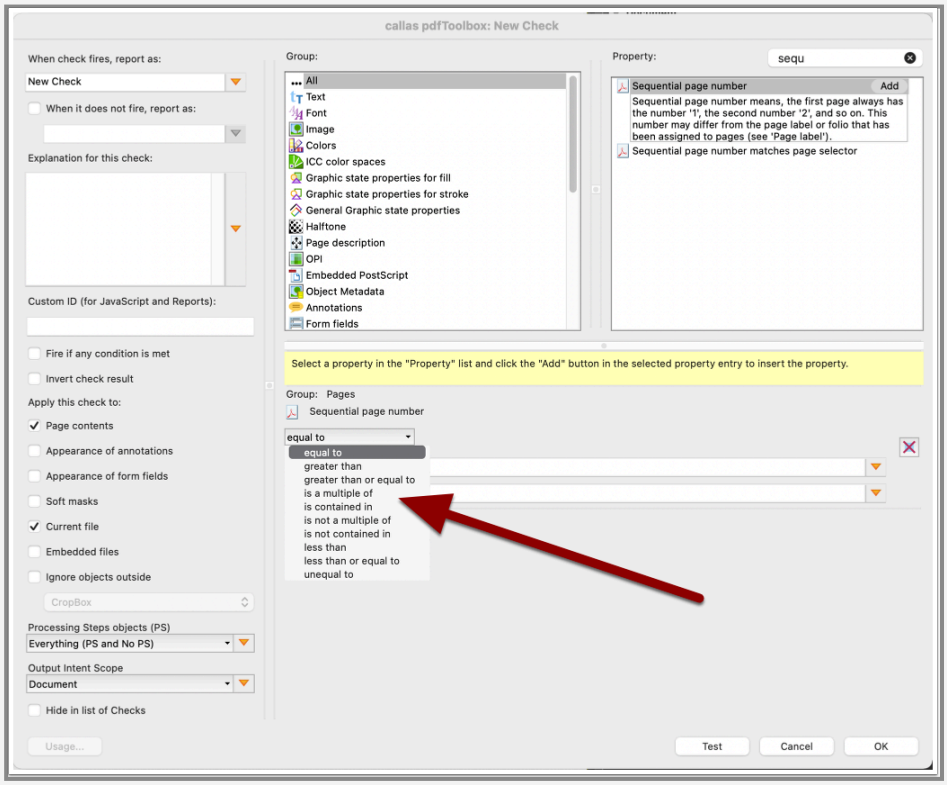


In this example, there only is the conditions array. In that case, the name, comment etc. are taken from the left hand settings of the Check.

This Check now may also be combined with other conditions, by "loading" them from the Property list.

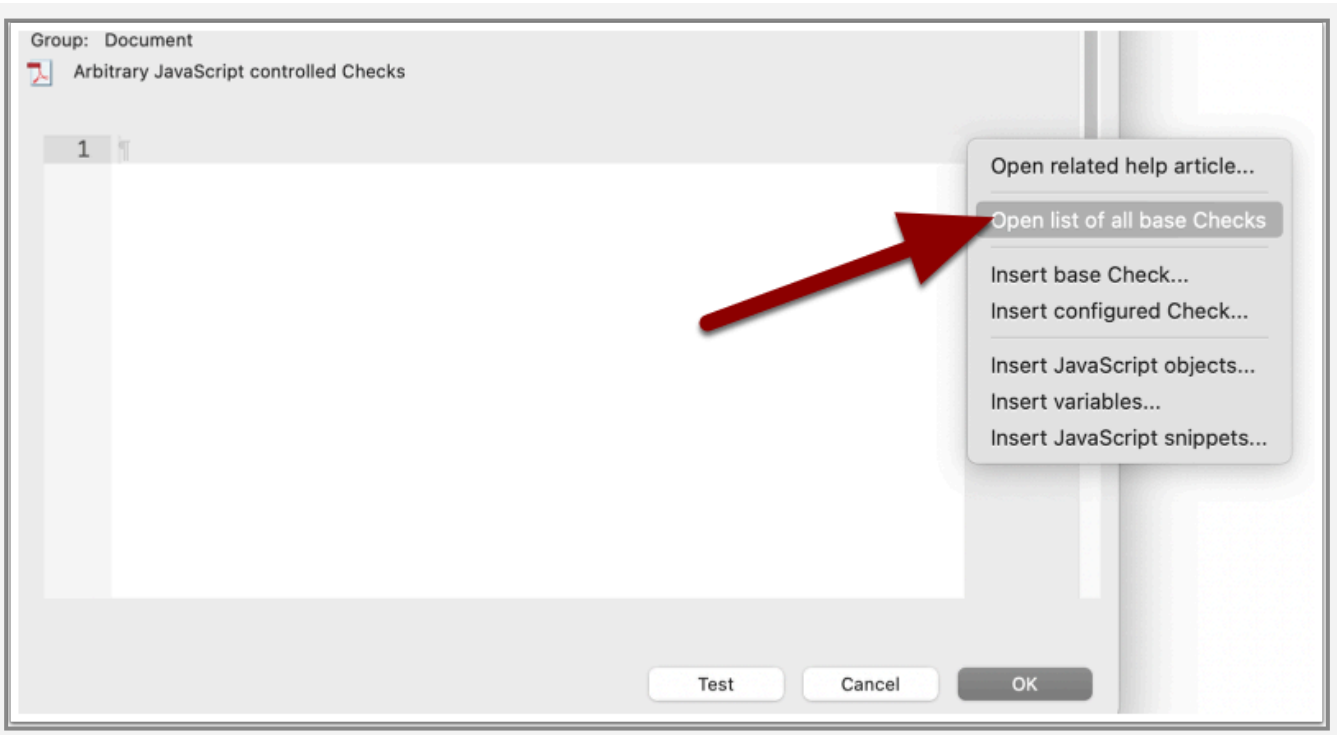
Values for pop up/drop down lists

Users may want to find out about the proper values for a Check that uses a Pop up, e.g. for "Sequential page number" that may have various operator values, like shown below.

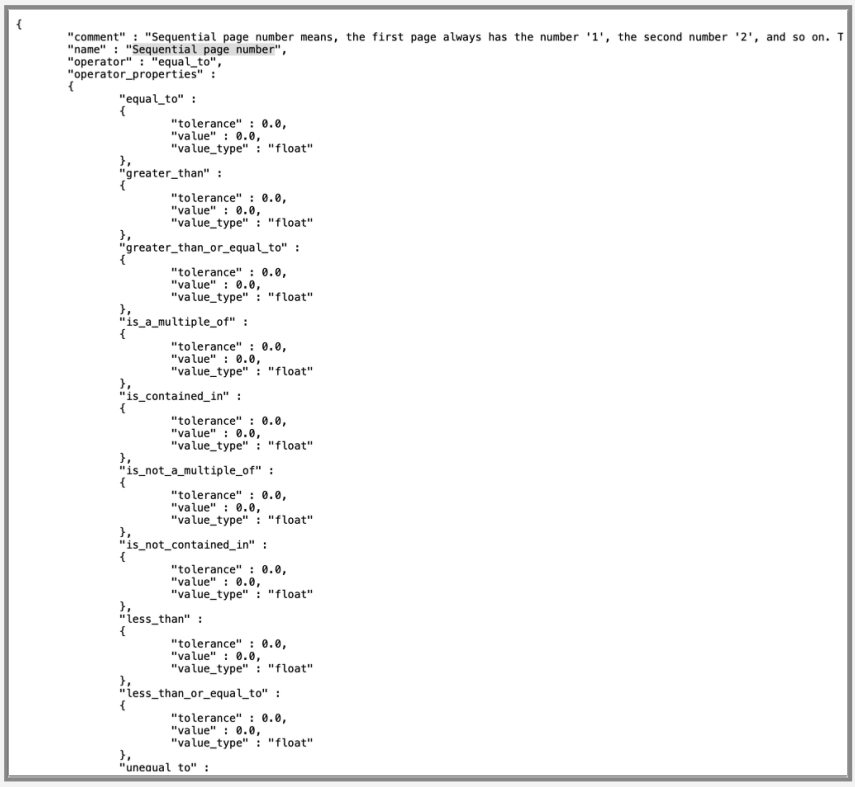


Here the "Open list of all base Checks" option comes into play.

(Go to 'Arbitrary JavaScript controlled Checks', click on the 'i' info button)



It opens a list of all existing Check properties in your web browser. Considering our example, search for "Sequential page number":



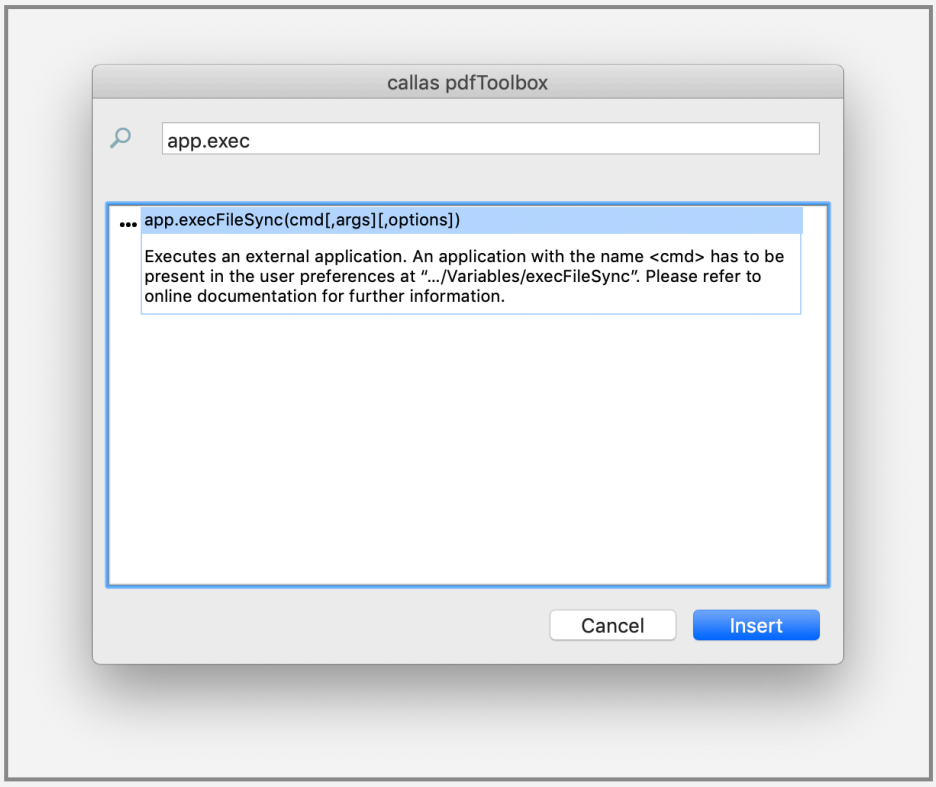
Any of the listed values for "operator_properties" can be used to configure the Check in your JavaScript code.

17.25 Execute external application via JavaScript function

callas pdfToolbox allows you to execute an external application via JavaScript function.

The application has to be present in the user preferences at "…/Variables/execFileSync".

Where to find User Preferences in Windows and Mac? Here exactly




```
app.execFileSync(cmd[,args][,options])
```

Click to copy

- `cmd` : name of a tool that must be present in <PREFS>/Variables/execFileSync with execute rights

- `args`: Optional array of arguments that are passed to cmd. It can either be a string "param" or an array of strings [""param1, "param2"] (if more than one parameter is needed)
- options: Optional. Currently supported properties are:
  - `cwd`: The `cwd` command is issued to change the client's current working directory to the path specified with the command. Default is the parent folder of the current document
  - `timeout`: <number> In milliseconds the maximum amount of time the process is allowed to run

Returns

The command will return an object containing the status stdout and stderr of the command execution:

```
{
  status: 0,
  stderr:"",
  stdout:""
}
```

Click to copy

Exception thrown when:

- cmd cannot be found in folder <PREFS>/Variables/execFileSync
- cmd is not an executable: The cmd file is not set to 'executable' in the file system. In that case the command line call "chmod +x<executable>" will help.

- cmd is a shell script and has no shebang entry (e.g. #! /bin/bash)

Please note that this feature is not portable since the commands are not included in the kfpx packages and the commands may differ between platforms.

Please note that executing an external application via JavaScript function creates an additional security risk.

Sample:

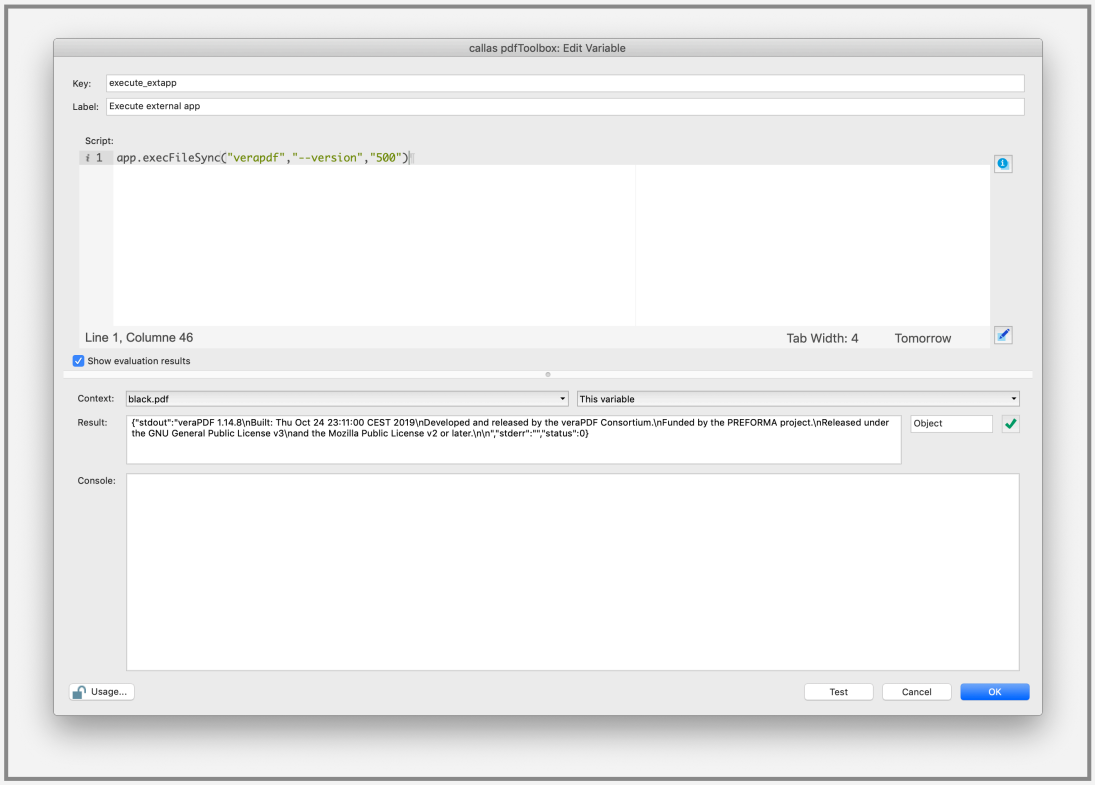
Please note that 'verapdf' application is placed under:

MacOS:
/Users/<USERNAME>/Library/Preferences/callas software/callas pdfToolbox CLI <VERSION>/Variables/execFileSync
Windows:
C:\Users\<USERNAME>\AppData\Roaming\callas software\callas pdfToolbox CLI <VERSION>\Variables\execFileSync

```
app.execFileSync("verapdf",["--version]",{timeout:500})
```

Click to copy

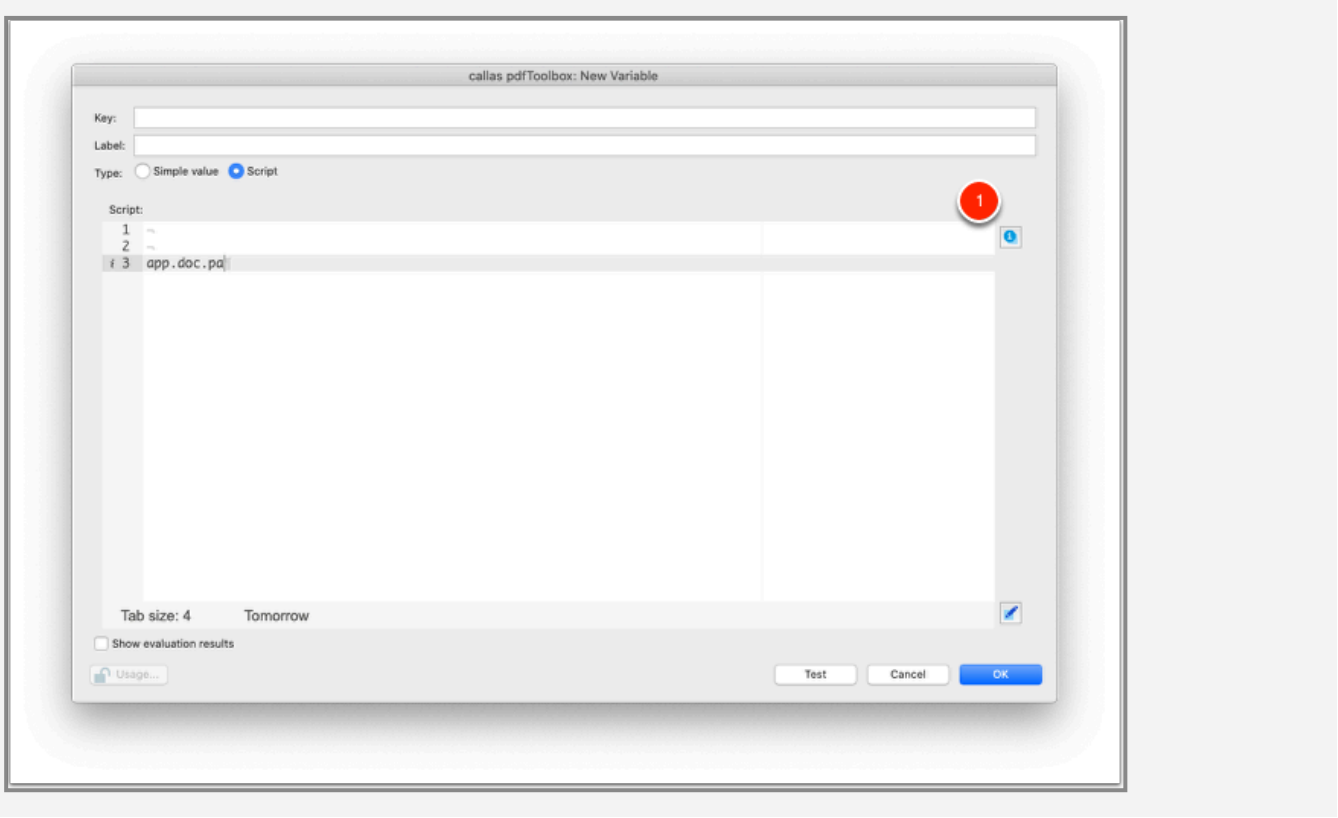
- cmd: verapdf
- args: --version
- options:
  - timeout: 500


callas pdfToolbox: Edit Variable
Key: execute_extapp
Label: Execute external app
Script:
1 app.execFileSync("verapdf","--version","500")
Line 1, Columne 46
Tab Width: 4
Tomorrow
Show evaluation results
Context: black.pdf
This variable
Result: {"stdout":"veraPDF 1.14.8\nBuilt: Thu Oct 24 23:11:00 CEST 2019\nDeveloped and released by the veraPDF Consortium.\nFunded by the PREFORMA project.\nReleased under the GNU General Public License v3\nand the Mozilla Public License v2 or later.\n\n","stderr":"","status":0}
Object
Console:
Usage...
Test
Cancel
OK

17.26 Autocomplete while writing JavaScript

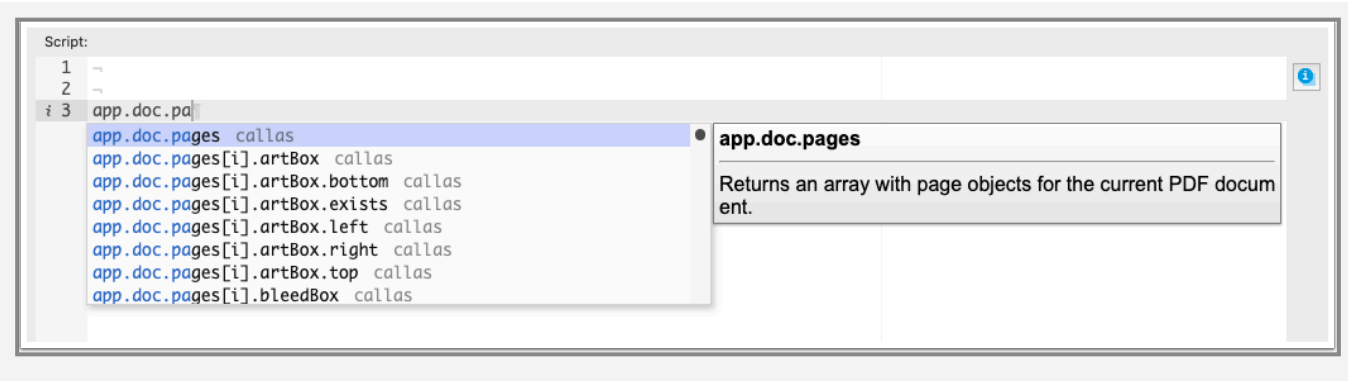
callas pdfToolbox includes a full JavaScript editor to allow writing JavaScript variables, configurations for QuickChecks, imposition run lists and so on. This editor of course understands modern JavaScript syntax, but the different places where it is used also support custom objects and syntax added by callas.

Typically this is done to make the bridge between the JavaScript engine and the callas processing engines, much like JavaScript in a browser has access to a Window object to make the connection to the browser window in which HTML is shown. In order to write effective JavaScript, it's important to know which objects are available. This can be done by using the information button and then selecting "Insert JavaScript objects".



This brings up a dialog box where the different JavaScript connection objects can be seen. Starting with pdfToolbox 14 however, there is a much faster way to find the correct con-

nection object, and this is simply to start typing the first letters of one of these connection objects. The editor in pdfToolbox immediately shows a list of all objects that start with the letters you already typed. You can continue typing, or scroll the list to pick one of the options displayed.



Where does this work?

This type of autocomplete works in many of the editors where JavaScript is used, such as:

- The script variable editor
- The QuickCheck configuration editor

17.27 Editing JavaScript in Visual Studio Code

callas pdfToolbox contains a full JavaScript editor that supports syntax coloring, code completion, theme support and more. Sometimes it is preferable to:

- Edit JavaScript in an editor with more possibilities (such as the inclusion of syntax-checking, linting or prettify-ing tools.
- Be able to edit all JavaScripts used in a Process Plan at the same time, with easy switching between the different scripts.

This is not possible in pdfToolbox Desktop itself, but is possible through a link with Microsoft Visual Studio Code. Because of the support necessary to link with an external editor, the fact that Microsoft Visual Studio Code can be used freely and is available on both Windows and macOS, this feature only works with this editor.

Configuration

In order for this type of editing to work, your system needs to be correctly configured. Specifically you need to install:

- Microsoft Visual Studio Code.
  You can install the software from the Microsoft web site.
- Microsoft Visual Studio Code command-line tools.
  On Windows and Linux this should be automatically available after installing Visual Studio Code. On macOS, you need to follow some extra steps: open "View" -> "Command Palette…" and select "Shell Command: Install 'code' command in PATH" (further information on this can be found here).

After installing Visual Studio Code, it is highly advised to read the documentation on the Microsoft web site and install some of the optional plug-ins that can help with JavaScript syntax coloring, syntax checking and prettify-ing your code.



To test whether installing the command-line tools was succesful, open a terminal window and run the "code" command.

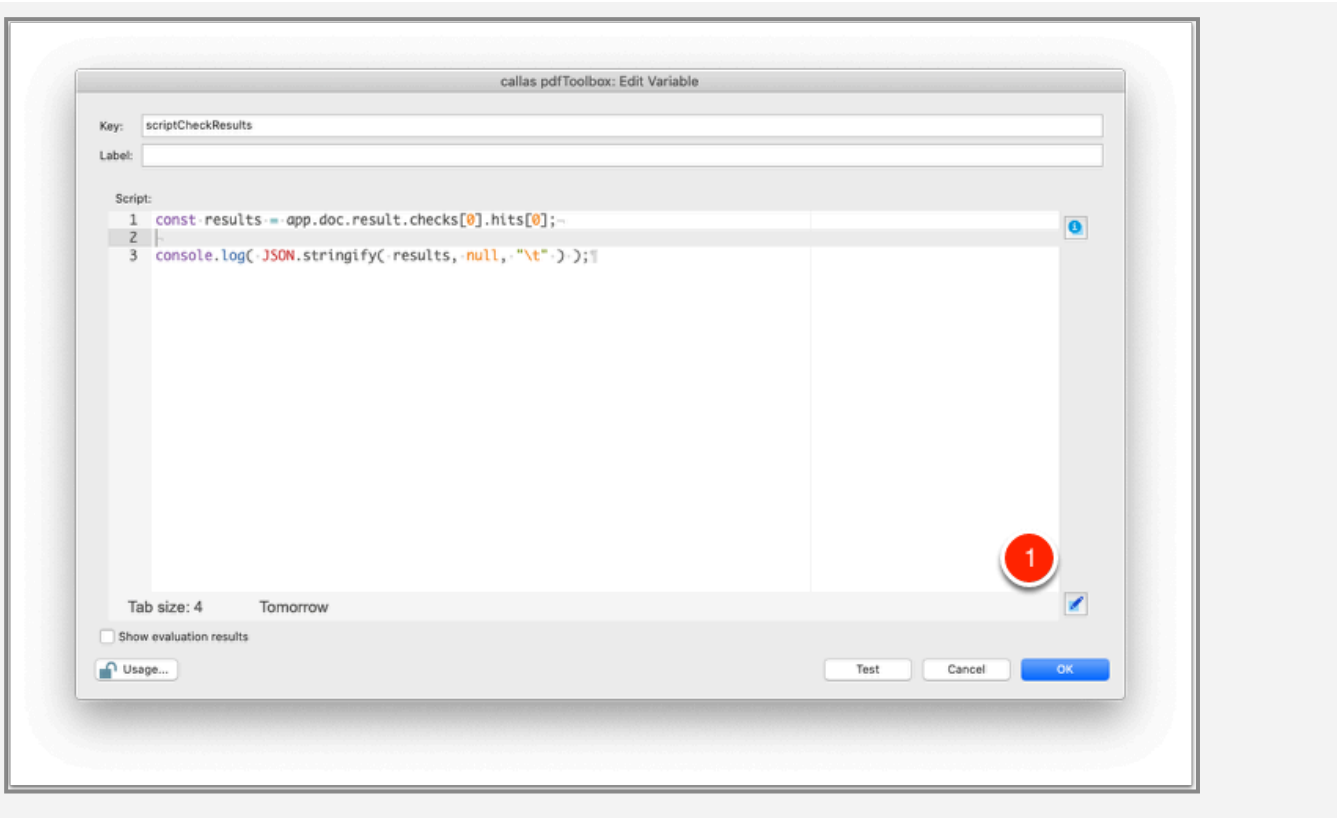
If everything is installed as required, this opens the Visual Studio Code application.

Starting the editor

There are two possible ways to open a script inside Visual Studio Code for editing.

Editing one script

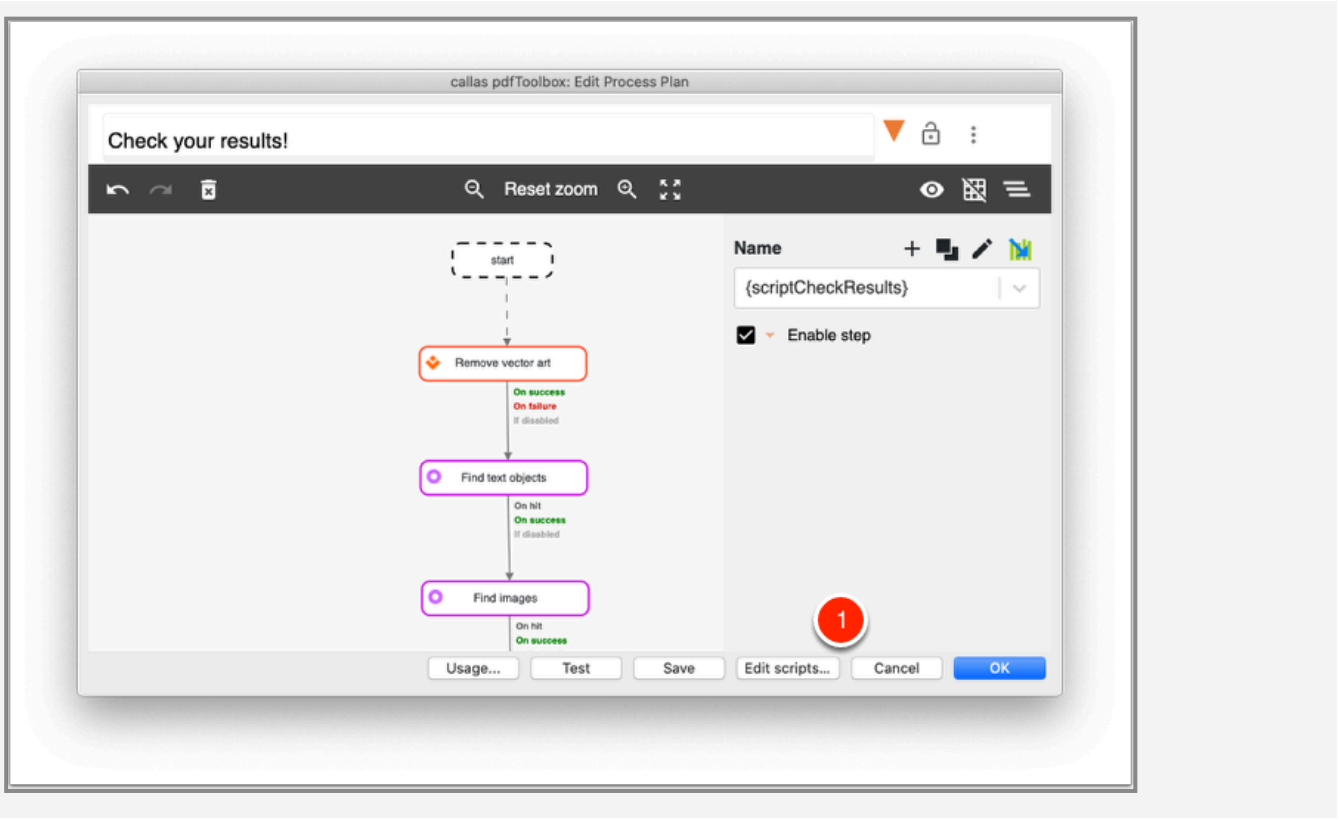
Open the script in the JavaScript editor, and click the "Edit in external editor" button.



If you want to know how to **debug your Java Script Variables** in MS Visual Studio Code, read the following article: [Debugging JavaScript Variables with Visual Studio Code]

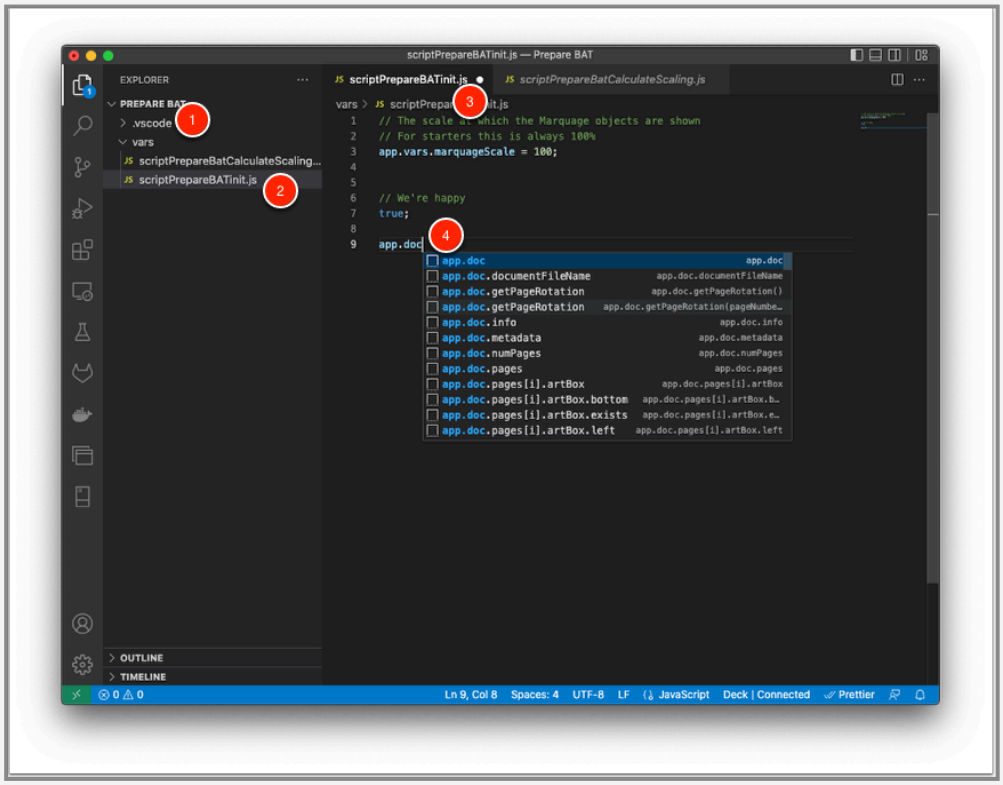
Editing all scripts

Open a Process Plan and click the "Edit scripts..." button at the bottom of the Process Plan editor.



Editing your scripts

When you opened a single script or configuration file, Visual Studio Code opens just that file. If you opened multiple scripts, you'll find all of them in the Visual Studio Code editor that opens.



Remark the following features of the Visual Studio Code editor:

1. The ".vscode" folder is inserted automatically by callas pdfToolbox. You should not modify or delete it.
2. All of your scripts and configuration files appear in the "vars" folder. Click any of them to open it in the editor, ready for editing.
3. At the top of the Visual Studio Code editor, you have a tab for each script file you have opened. If the file was changed, there is a filled circle marker at the end of the name of the script.
4. Even though you are not editing in pdfToolbox, you still get auto-completion. In this example typing "app.doc" brought up all properties of the document pdfToolbox gives access to through JavaScript.

Saving your changes

You do not have to copy and paste your changes in the Visual Studio Code editor to your Process Plan, simply use the

"Save" menu command or use "Ctrl+S" on Windows, or "Command+S" on macOS to save your changes.

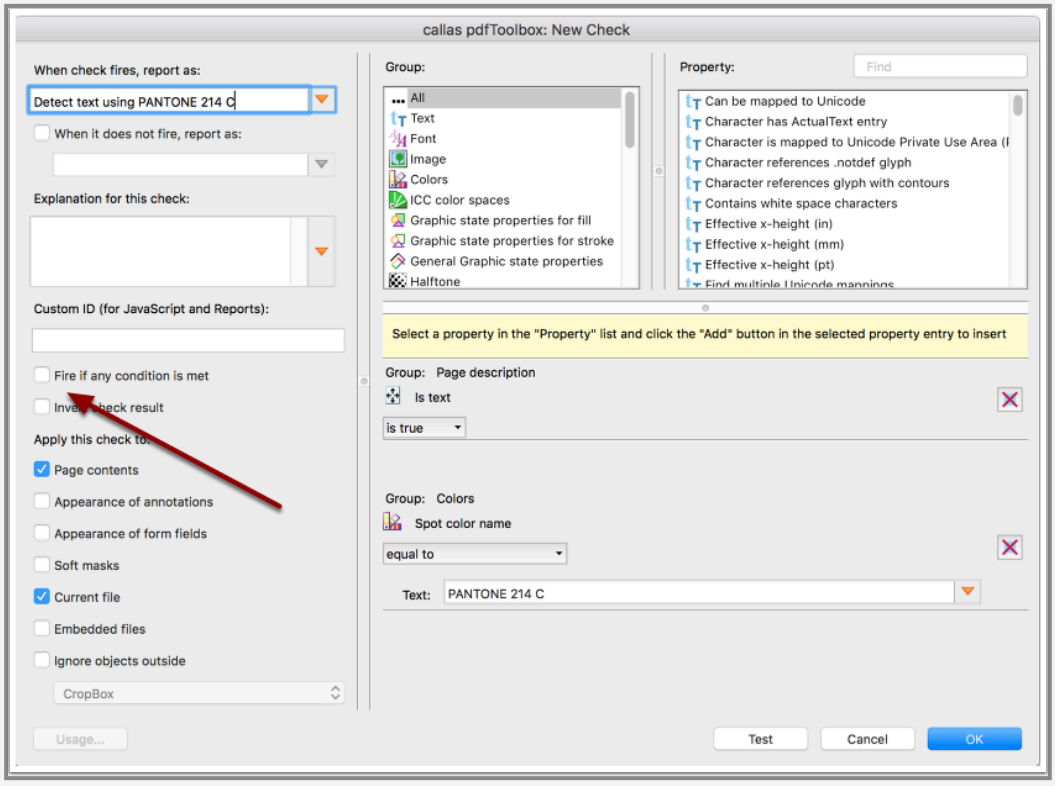
callas pdfToolbox detects you made a change and automatically updates your Process Plan scripts as necessary.

18. Advanced configuration of Profiles

18.1 Boolean logic and conditions in preflight checks

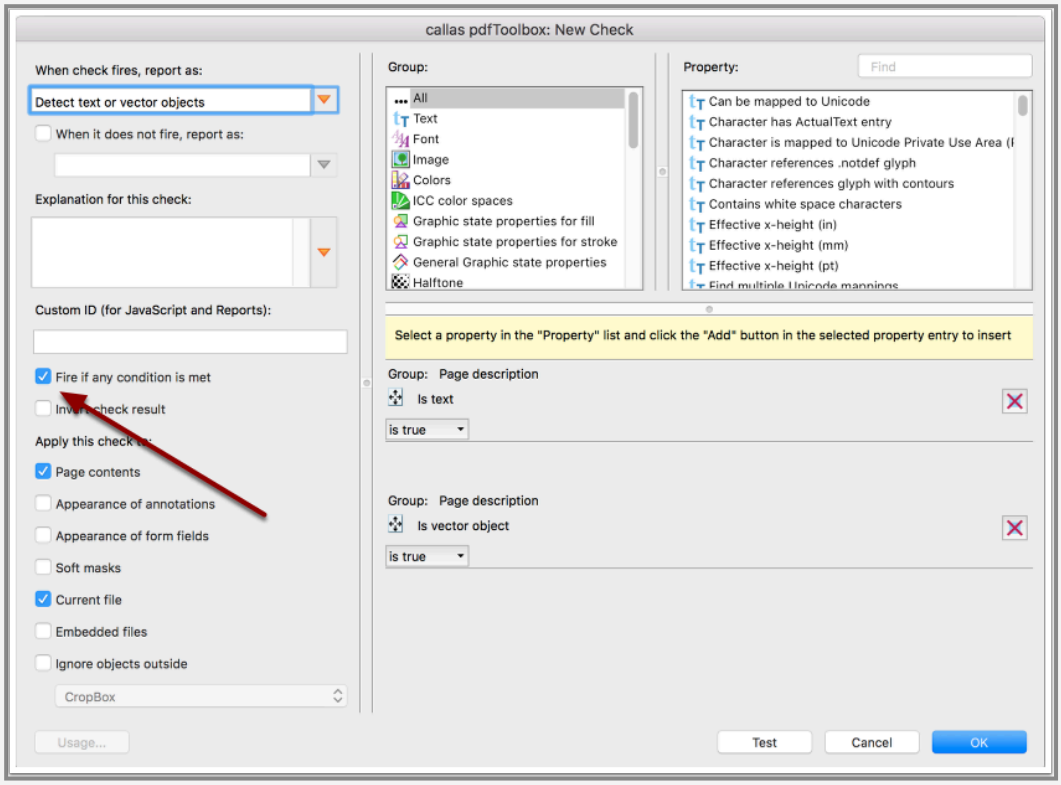
In pdfToolbox, preflight checks are built from individual conditions. Combining these conditions is what makes pdfToolbox such a flexible tool. This article shows how conditions work and how they can be combined with Boolean conditions. The next article shows how this can be made even more flexible in pdfToolbox 10. If you have a good working knowledge of how checks work in pdfToolbox 9 and before, feel free to skip to this next article straight away.

Using a logical AND



This preflight check contains two conditions: one to select objects that are text objects, and a second to select objects using a specific spot color. Because the "Fire if any condition is met" checkbox is *off*, those two conditions are combined using a logical AND. In other words, this check will only fire if objects are found that are text *AND* use that particular spot color.

Using a logical OR



This preflight check contains two conditions: one to select objects that are text objects, and a second to select objects that are vector objects. Because the "Fire if any condition is met" checkbox is *on*, those two conditions are combined with a logical *OR*. In other words, this check will fire if objects are found that are text *OR* vector.

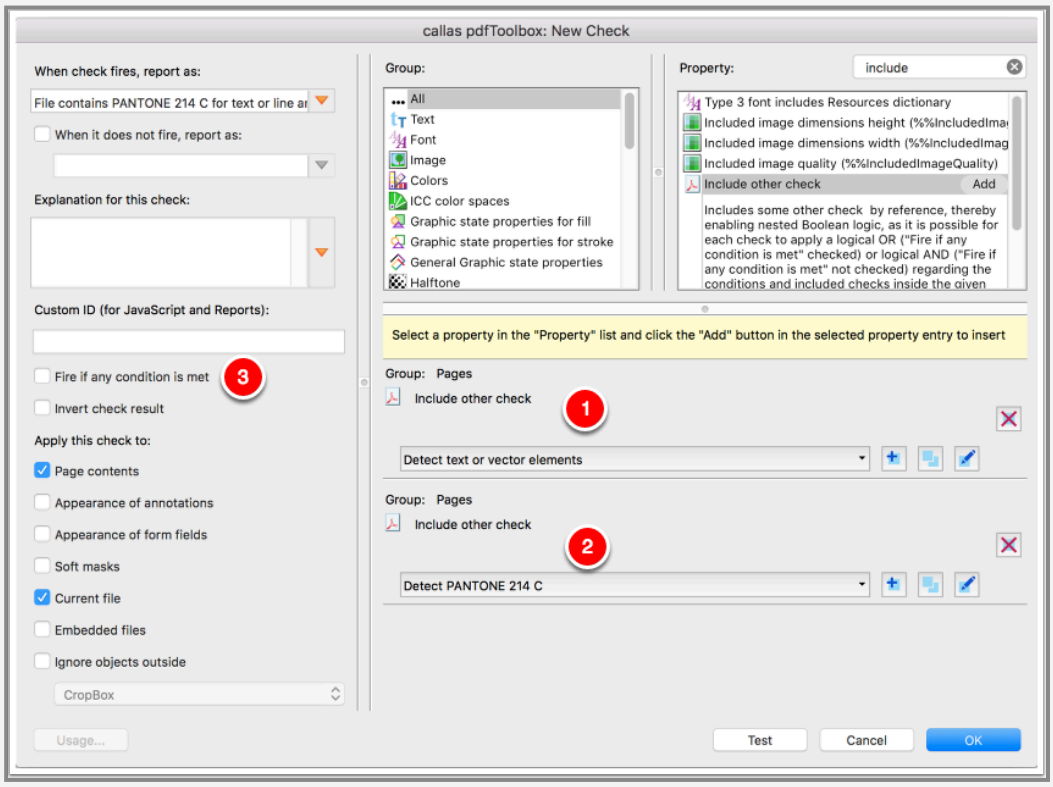
Combining AND and OR in one check

In pdfToolbox 9 and before, it was not possible to make more complex checks, checks would always use logical *AND* or logical *OR*. The next article explains how this changes in pdfToolbox 10 and how you can make more powerful checks.

18.2 Combining different checks to solve complex problems

If you need a refresher on how conditions in preflight checks worked in versions of pdfToolbox before version 10, please first read the previous article to set the scene.

In pdfToolbox 10, a new condition is added. The condition is called "Include other check" and it allows combining the results of other checks in a new check. An example is shown below:



The check consists of two conditions. In both cases the condition used is "Include other check" which simply returns the result of another previously made preflight check. In the example shown, the first check will fire if either a text object *OR* if a vector object has been found and the second check will fire is a particular spot color is found.

The results of these two preflight checks are combined with a logical *AND* as the "Fire if any condition is met" checkbox is switched *off*. The final result of the pictured preflight check is

that it will fire if an object is found that is using that particular spot color *AND* it is a text object *OR* a vector object.

Remarks

- This example check shows two "Include other check" conditions being used. Of course you can combine such a condition with other, regular, conditions. You can also have more than two "Include other check" conditions being used in a single preflight check.
- With this addition, you can create arbitrarily complex preflight checks. The result of the pictured preflight check before could be used in yet another preflight check to build an even more powerful boolean expression.



File_contains_PANTONE_214_C_for_text_or_line_.kfpx

18.3 Negating Check results

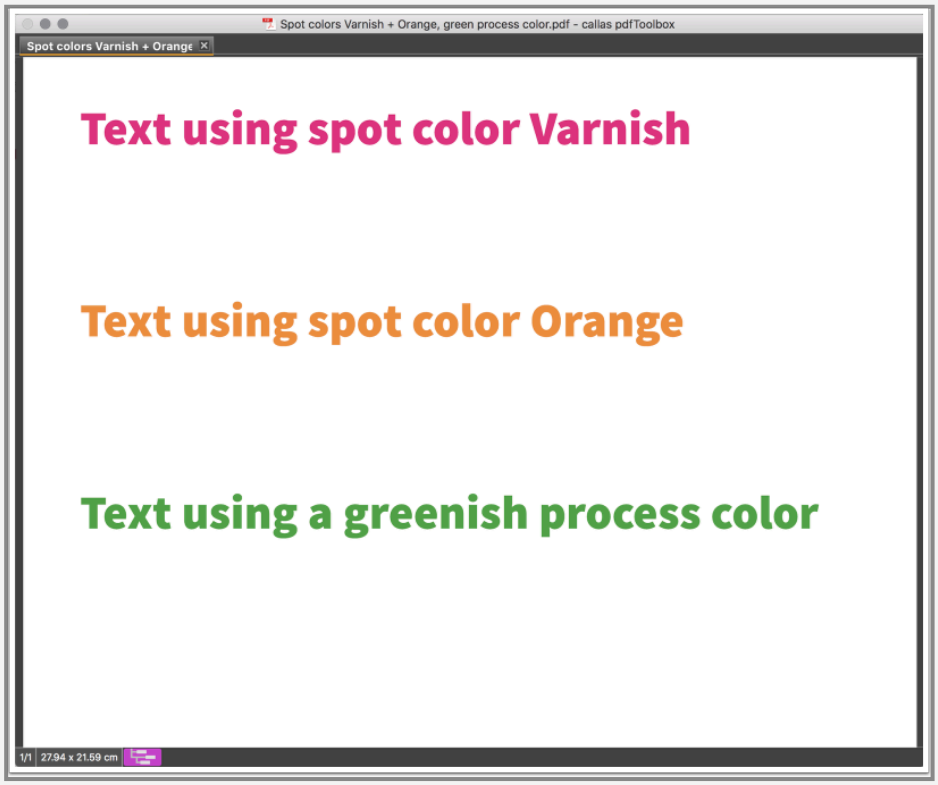
pdfToolbox 10 introduces two new methods for negating Check results:

- by inverting a Check (in a boolean way)
- by adding a condition that fires if the number of hits in the other conditions is zero

In older versions it was possible to use operators like "does not contain", "unequal to" or "is not true". All three variants of negation have different results.

The easiest way to understand the differences and use cases is to use an example. We will use a PDF that uses two spot colors and a process color.

Spot_colors_Varnish___Orange__green_process_color.pdf



The attached Profile uses Checks for each of the three possible ways to configure negation.

Negation.kfpx

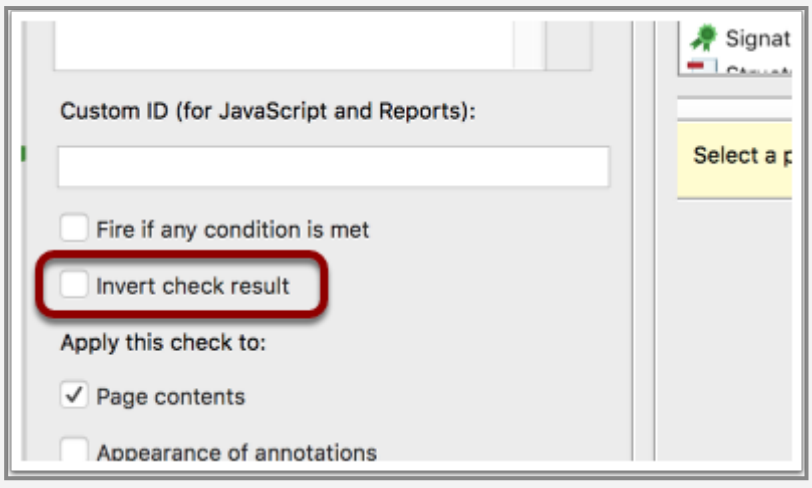
Negation by operators "does not contain", "unequal to" or "is not true"

The Profile has a Check "Spot color name equal to Orange" and a Check "Spot color name unequal to Orange". The first finds one item in the sample PDF, the second line of text, the second Check finds the first line.

The background to that is that the Property "Spot color name" that has been used only checks objects that are using spot color. Most of the Properties in pdfToolbox internally have such limitations, e.g. the Property "Image resolution" will only find images. Usually these limits work intuitively right, but sometimes that is not intended - and that is when the next option comes into play.

Negation by "Invert check result"

In the left column of the Check editor is a checkbox "Invert check result".



In the sample Profile it is used in "Inverted: Spot color name unequal to Orange". This Check finds now two lines of text:

The one that uses spot color Varnish and in addition the one that uses process color.

For the properties:

- “**Include other check**” and
- any **Sifter properties** (Context aware object detection)

the option “invert check result” is not possible and therefore grayed out.If you want to add one of these properties to a Check where the “Invert check result” checkbox is already activated, the checkbox will be automatically disabled and grayed out. The reason for this is that the result would otherwise be undefined.

Negation by adding a condition that fires if the number of hits in the other conditions is zero

The Profile also contains Checks "Number of hits equal 0 for Spot color name equal to Orange" and "Number of hits equal 0 for Spot color name equal to Violet". The additional Property "Number of hits in this check" that is used in both of them is not new in pdfToolbox 10, new is that is can now also be combined with "equal to" 0 (zero).



This property always works on a page basis, this means when used, the scope of the Check are no more objects on the page

but the page itself. So it converts the other Property in the Check, e.g. "Spot color name" into a page based Check that fires if the number of hits on the page matches what has been defined in "Number of hits in this check".

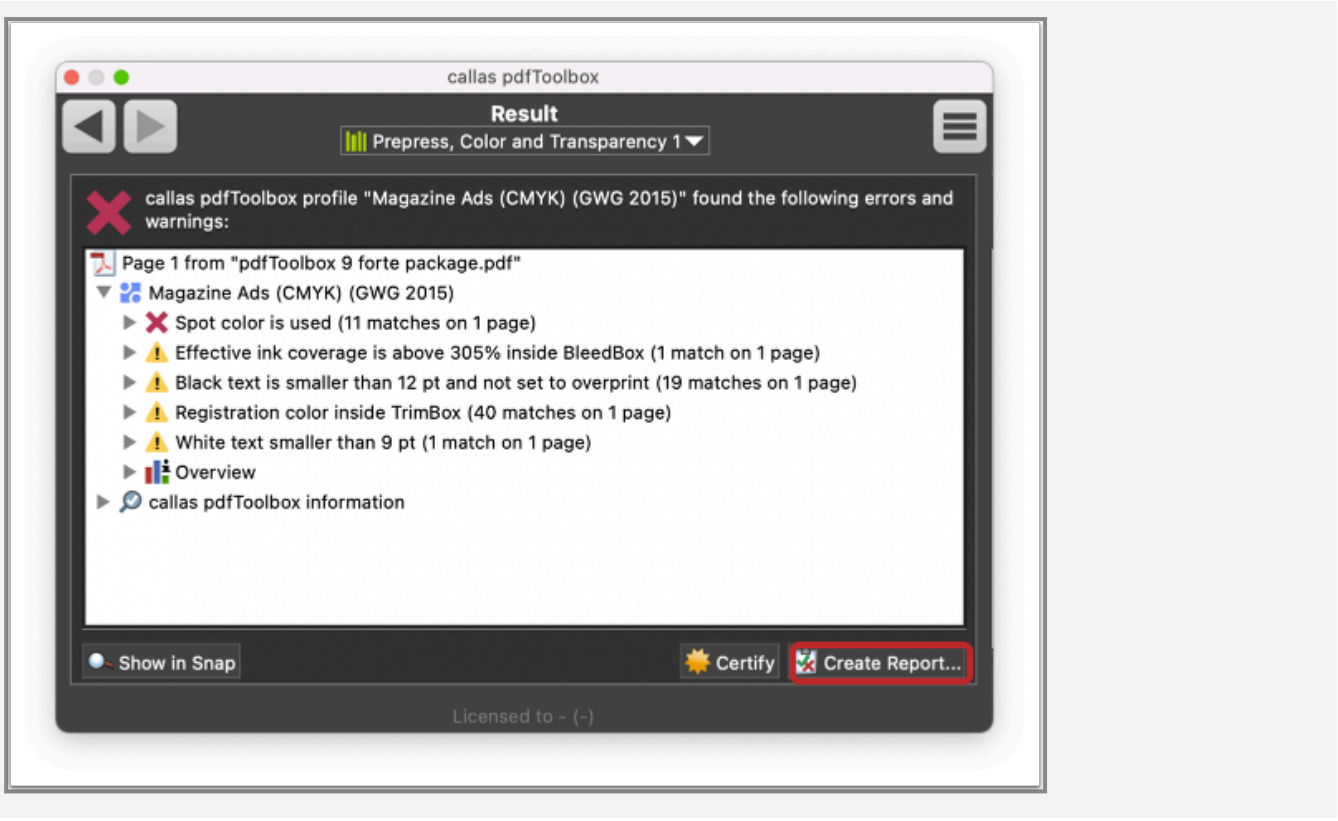
In the sample only the Check "Number of hits equal 0 for Spot color name equal to Violet" will generate a hit since that spot color is not used on the page.

pdfToolbox 10 also introduces a new Property "Number of hits in this check in document" that counts hits not on page level but on document level. So it would convert the other conditions in the Check (whether they are designed to work for objects or pages) into document level Properties.

19. Reports for Profiles

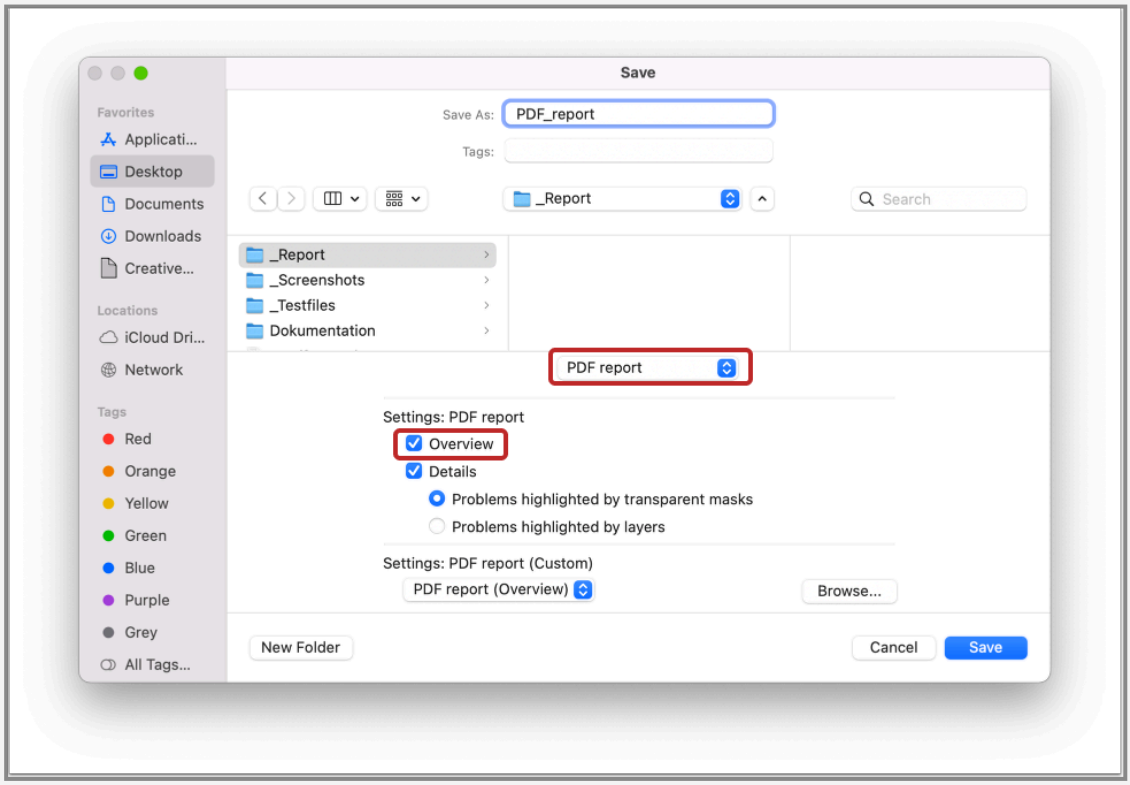
19.1 PDF reports (using masks or layers)

This article explains the different ways to create a PDF report. You have two different options to create a PDF report. Lets have a look at these.

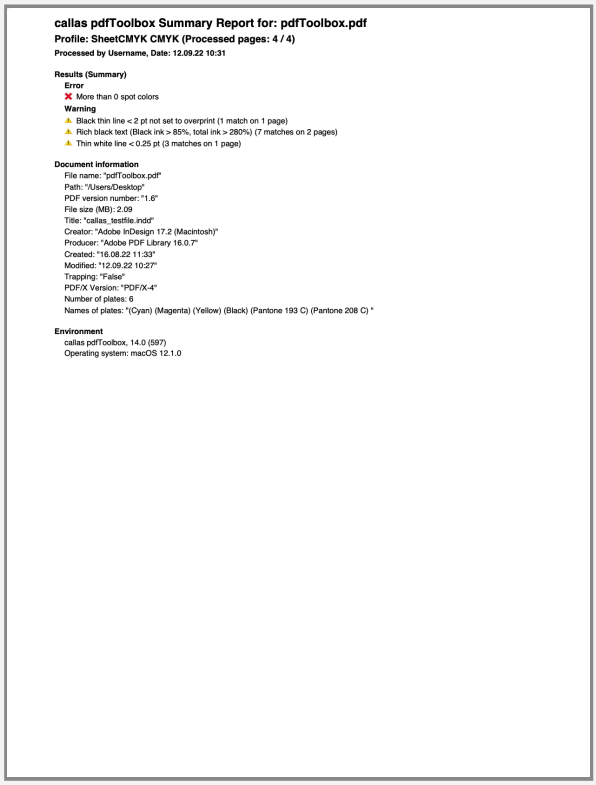


PDF report – Overview page

If "**PDF report**" is selected, a PDF document is created that can be opened in any PDF viewer. This PDF report optionally begins with an overview page; this page summarizes information about the preflighted document, the environment in which it was preflighted and provides a list of errors, infos and warnings. This PDF report ist static, in the sense that you cannot change the layout, colors or text of the preflight report.



Example of the overview page of a PDF report



callas pdfToolbox Summary Report for: pdfToolbox.pdf
Profile: SheetCMYK CMYK (Processed pages: 4 / 4)
Processed by Username, Date: 12.09.22 10:31

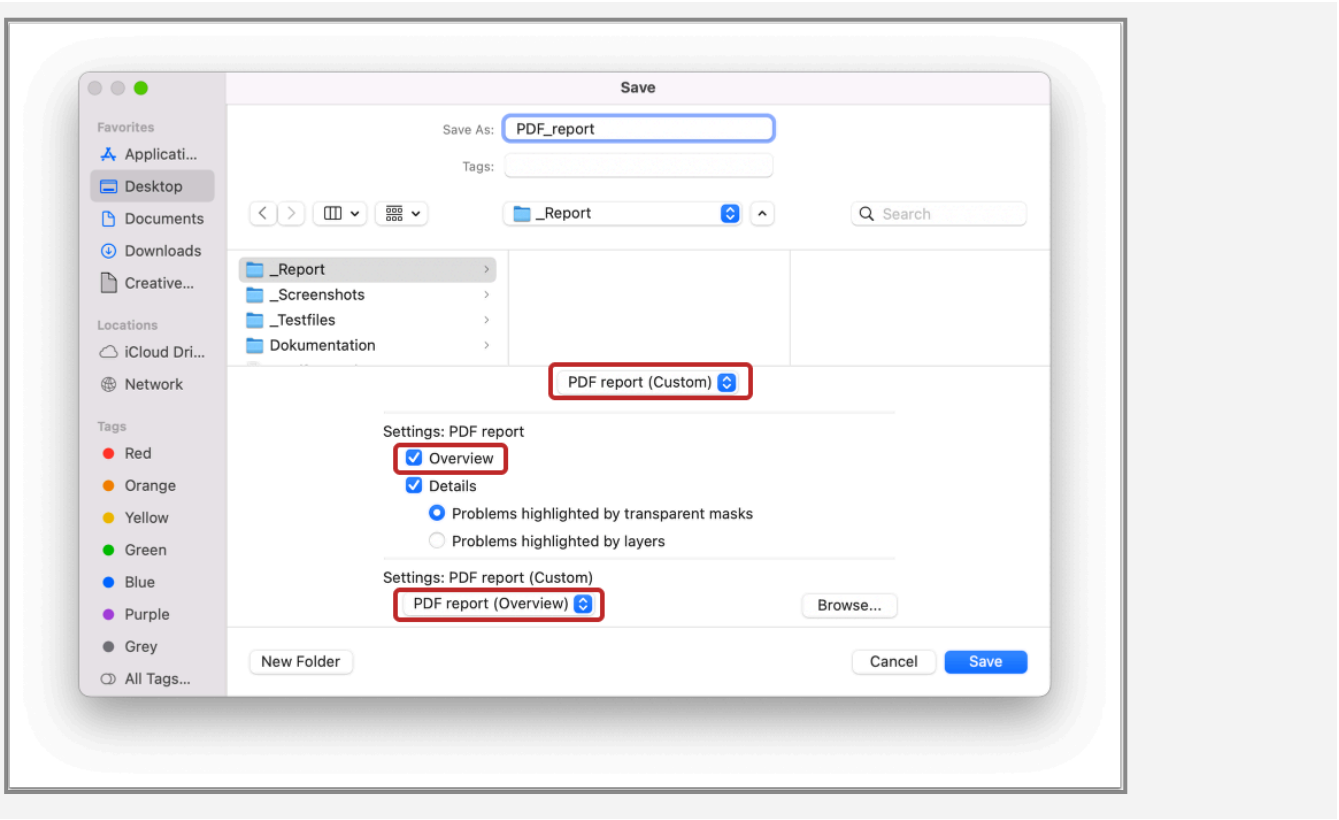
Results (Summary)
Error
✖ More than 0 spot colors
Warning
▲ Black thin line < 2 pt not set to overprint (1 match on 1 page)
▲ Rich black text (Black ink > 85%, total ink > 280%) (7 matches on 2 pages)
▲ Thin white line < 0.25 pt (3 matches on 1 page)

Document information
File name: "pdfToolbox.pdf"
Path: "/Users/Desktop"
PDF version number: "1.6"
File size (MB): 2.09
Title: "callas_testfile.indd"
Creator: "Adobe InDesign 17.2 (Macintosh)"
Producer: "Adobe PDF Library 16.0.7"
Created: "16.08.22 11:33"
Modified: "12.09.22 10:27"
Trapping: "False"
PDF/X Version: "PDF/X-4"
Number of plates: 6
Names of plates: "(Cyan) (Magenta) (Yellow) (Black) (Pantone 193 C) (Pantone 208 C) "

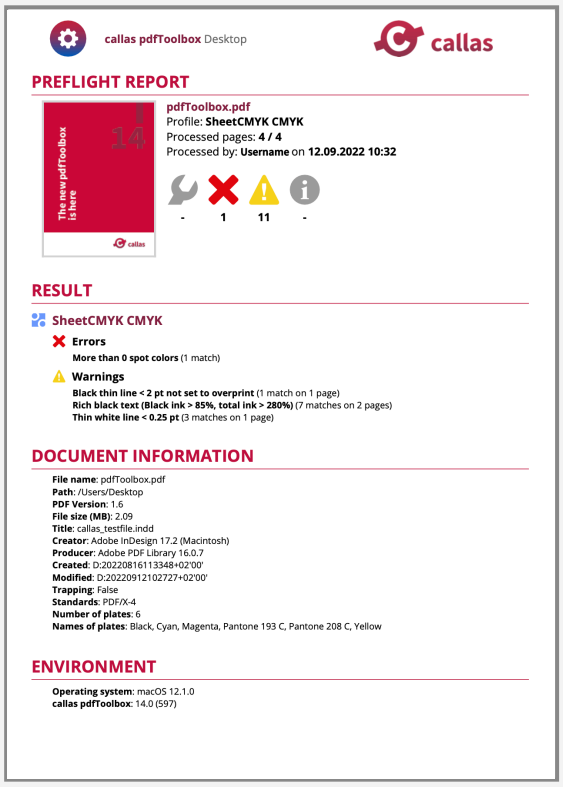
Environment
callas pdfToolbox, 14.0 (597)
Operating system: macOS 12.1.0

PDF report (Custom) – Overview page

If "**PDF report (Custom)**" is selected, a HTML template is used for the overview page layout. The product always comes with a predefined template called "**PDF Report (Overview)**", which can be selected at the bottom of the dialog.



Example of the overview page of a PDF report (Custom)


callas pdfToolbox Desktop

callas

PREFLIGHT REPORT

pdfToolbox.pdf
Profile: **SheetCMYK CMYK**
Processed pages: **4 / 4**
Processed by: **Username** on **12.09.2022 10:32**

- 1 11 -

RESULT

SheetCMYK CMYK

Errors
More than 0 spot colors (1 match)

Warnings
Black thin line < 2 pt not set to overprint (1 match on 1 page)
Rich black text (Black ink > 85%, total ink > 280%) (7 matches on 2 pages)
Thin white line < 0.25 pt (3 matches on 1 page)

DOCUMENT INFORMATION

File name: pdfToolbox.pdf
Path: /Users/Desktop
PDF Version: 1.6
File size (MB): 2.09
Title: callas_testfile.indd
Creator: Adobe InDesign 17.2 (Macintosh)
Producer: Adobe PDF Library 16.0.7
Created: D:20220816113348+02'00'
Modified: D:20220912102727+02'00'
Trapping: False
Standards: PDF/X-4
Number of plates: 6
Names of plates: Black, Cyan, Magenta, Pantone 193 C, Pantone 208 C, Yellow

ENVIRONMENT

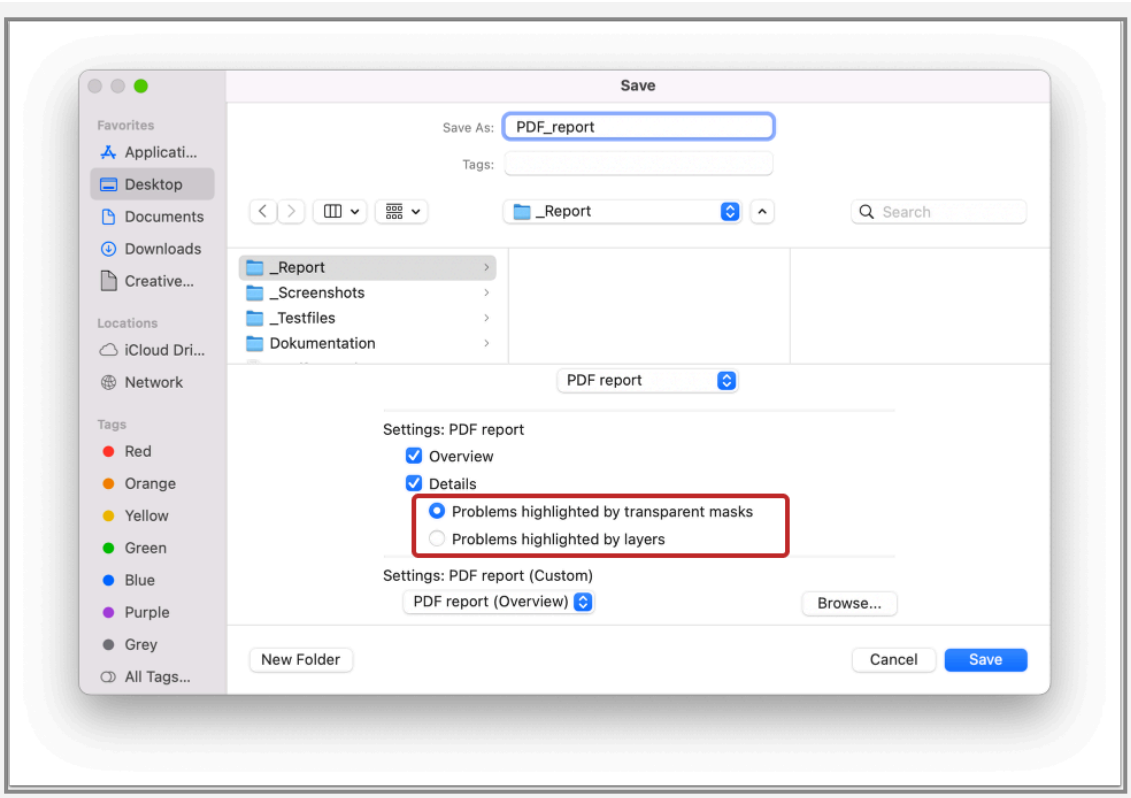
Operating system: macOS 12.1.0
callas pdfToolbox: 14.0 (597)

Unlike the static PDF report, the layout and content of the custom PDF report can be extended and customized:

- **Optional pages**: It is possible to add five optional pages to your report. These pages provide information about the spot colors, page information, ink coverage, ink amount and separation previews. How to add these pages to your report is explained in this article.
- **Modify the template**: You can create your own templates, by modifying the predefined template. For example you can change colors, add your own logos or artwork and even modify what text is shown. The next article will give you an overview of all components of the report template.

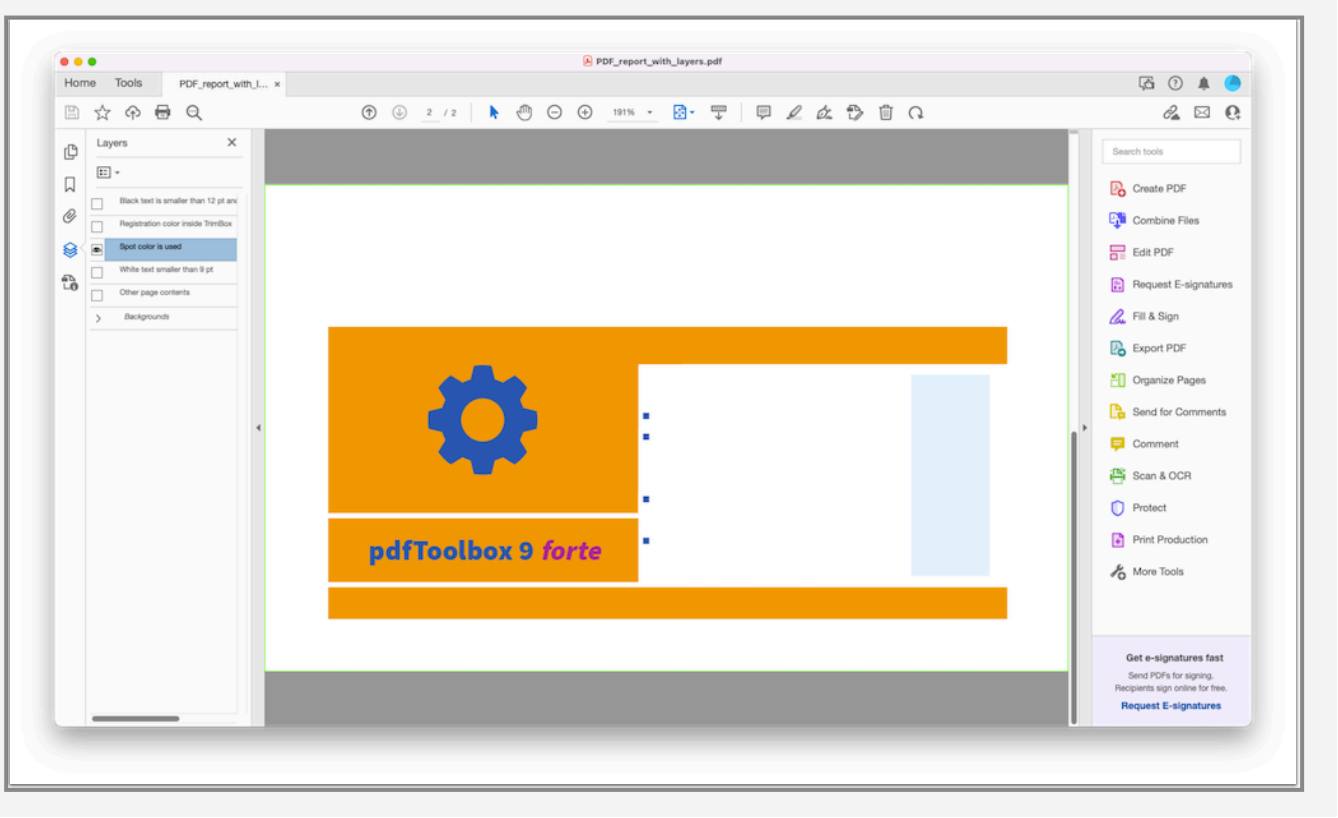
Highlighting problematic objects using masks or layers

In addition to the overview page you can generate a copy of the PDF file that was preflighted where all hits are identified and highlighted either by layers or masks. This copy will be added after the overview page.



Example of a layer report

The layer report puts objects with a particular problem on a layer; this allows toggling layers on and off to see all objects causing a Warning or Error. Because the report also adds background layers, you can easily see the problem objects on differently colored backgrounds (which is great to see white objects for example).



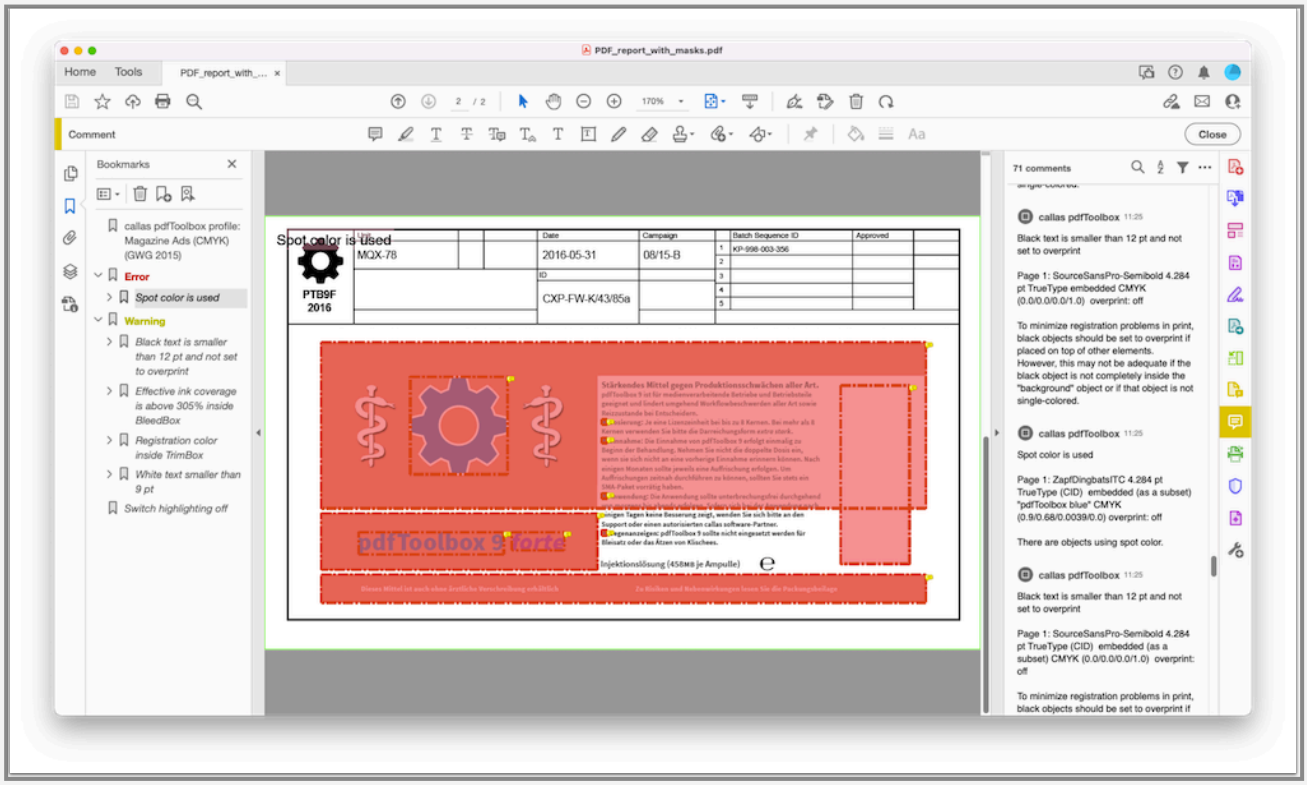
Example of a mask report

The transparency mask report uses a combination of transparent objects and comments to highlight all objects causing a Warning or Error. Additionally, bookmarks are added to make it easier to navigate through the highlighted areas.

The first level of the bookmark hierarchy has:

- The name of the Profile, Check or Fixup that was used. The destination is the first page of the report.
- The severities (Error, Warning or Info) of the hits encountered. The destinations are the first pages with a hit of the respective severity.
- "Switch highlighting off" stays on the current page but switches all masks off.

On the second level within Severities, there are the names of the respective Checks. Within each Check on the third level are all pages where the respective Check found a hit.

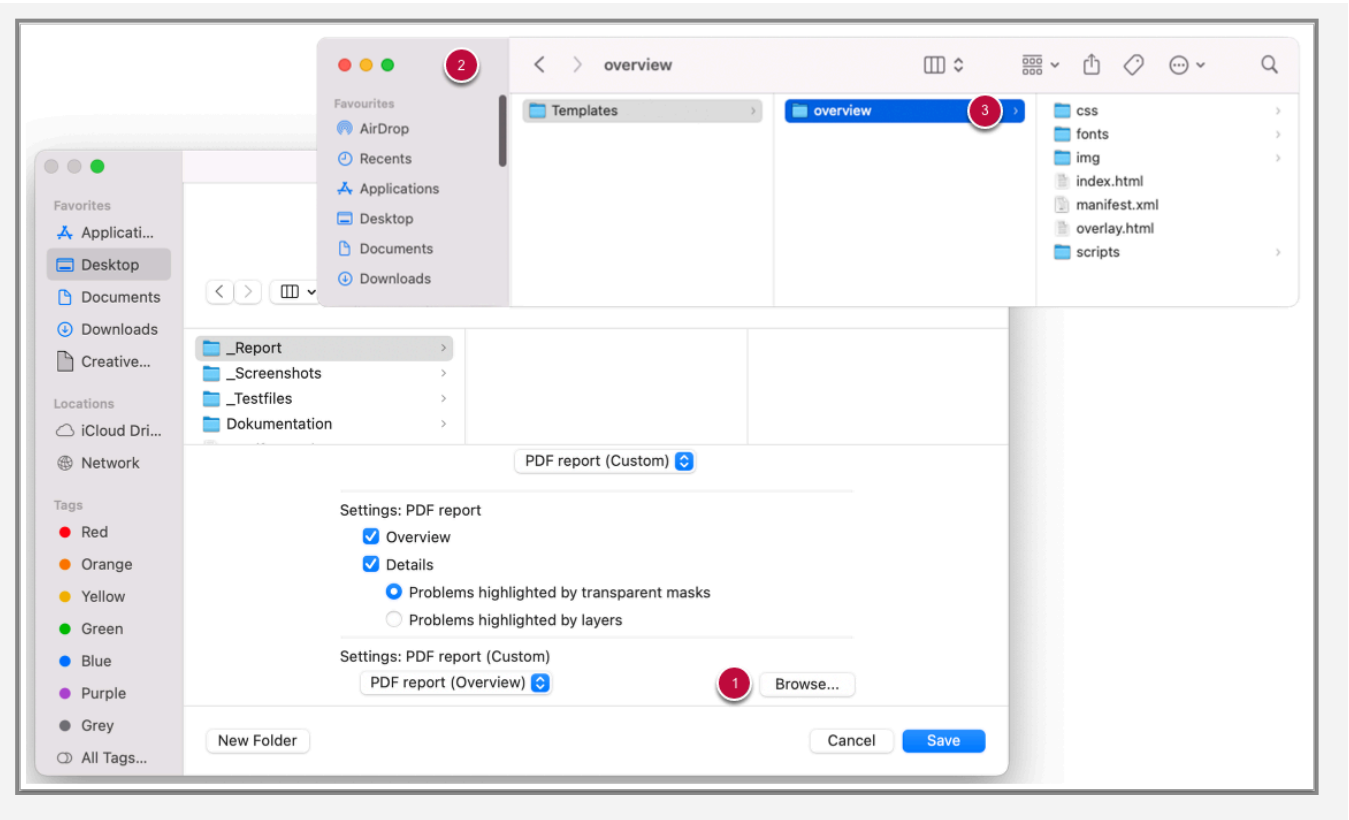

PDF_report_with_masks.pdf
Home
Tools
Comment
Close
Bookmarks
callas pdfToolbox profile: Magazine Ads (CMYK) (GWG 2015)
Error
Spot color is used
Warning
Black text is smaller than 12 pt and not set to overprint
Effective ink coverage is above 305% inside BleedBox
Registration color inside TrimBox
White text smaller than 9 pt
Switch highlighting off
Spot color is used
71 comments
callas pdfToolbox
Black text is smaller than 12 pt and not set to overprint
Page 1: SourceSansPro-Semibold 4.284 pt TrueType embedded CMYK (0.0/0.0/0.0/1.0) overprint: off
To minimize registration problems in print, black objects should be set to overprint if placed on top of other elements. However, this may not be adequate if the black object is not completely inside the "background" object or if that object is not single-colored.
Spot color is used
Page 1: ZapfDingbatsITC 4.284 pt TrueType (CID) embedded (as a subset) "pdfToolbox blue" CMYK (0.9/0.68/0.0039/0.0) overprint: off
There are objects using spot color.
Page 1: SourceSansPro-Semibold 4.284 pt TrueType (CID) embedded (as a subset) CMYK (0.0/0.0/0.0/1.0) overprint: off

19.2 HTML based custom reports

As mentioned in the previous article, it is possible to generate a custom PDF report using a HTML template. The visual appearance is controlled by a HTML-Template and Custom Style Sheets (CSS), while the reported details are directly requested from the software itself or (optionally) parsed from an internally created JSON or XML report.

Structure of the custom HTML report

A predefined HTML template is always contained in all installer packages for Desktop and Server/CLI. The following screenshot shows how to get to the template folder:



1. When generating a report, click on the "**Browse…**" button in the settings section.
2. A new window into the file system opens where the Template folder is stored.
3. The **overview** folder is the predefined Template folder.

The predefined template contains several folders and files

- **index.html:** the template in HTML format that references a number of stylesheets, JavaScripts as well as a number of images
- **overlay.html:** contains the company and product logo that is placed on top of every report page
- **manifest.xml:** a XML file which defines information needed as content for the report, to be delivered by the engine
- **/css:** contains a style sheet
- **/fonts:** contains used fonts
- **/img:** contains used images
- **/scripts:** contains used JavaScripts



It is highly recommended to create a copy of the original template in a separate folder when starting to adjust a HTML-template based report.

The manifest.xml

The manifest.xml defines the set of information to be provided by the engine. This information will be used to fill up the details in the report based on the HTML-Template. Basic document information as well as all results of the processed profiles and a preview image are provided by default. Other parts like comparision images, a JSON or XML report, or parameters for optional document information can be also requested to enable picking up additional information about the PDF. These parts are commented out in the manifest by default. Here an example of an optional part of the manifest.xml:

```
<!--
    Create a xml report

        The path to the xml report is stored in cals_res_info.xml_report'

        If either inkcovres or inkcovbox is specified, ink coverage information is added to the report

        path: Destination path for XML report, Default: xml/report.xml
        inkcovres: Resolution for ink coverage. Default: "10"
        inkcovbox: Page box for ink coverage. Default: "CropBox"

<x:xmlreport path="xml/report.xml" inkcovres="10" inkcovbox="CropBox"/>
-->
```

As you can see, the hole XML report section is commented out by surrounding the code with `<!--` and `-->`. Down at the bottom you can find the XML report parameter. If you want to request a XML report, all you have to do is to uncomment the XML report parameter:

```
<!--
    Create a xml report

        The path to the xml report is stored in cals_res_info.xml_report'

        If either inkcovres or inkcovbox is specified, ink coverage information is added to the report

        path: Destination path for XML report, Default: xml/report.xml
        inkcovres: Resolution for ink coverage. Default: "10"
        inkcovbox: Page box for ink coverage. Default: "CropBox"

-->
<x:xmlreport path="xml/report.xml" inkcovres="10" inkcovbox="CropBox"/>
```

Parameters of the manifest.xml

Keep the temporarily generated files

```
<x:keeptemp>false/x:keeptemp>
```

If `true` the temporarily generated files like the filled index.html, CSS-files, images, JSON or XML reports and used JavaScripts will not be deleted after finishing the PDF report. They will be saved in a new folder with the extension .html instead. The folder name is identical to the PDF file. The folder is saved in the same location where the PDF file is located.

Options

- `false`: temporarily generated files will be deleted (default)
- `true`: temporarily generated files will be saved

Resolution for preview images

```
<x:defaults resolution="20"/>
```

Click to copy

This parameter allows you to set the default value for the resolution of all preview images. This value may be overwritten by the individual settings.

Options

- `resolution`: resolution used in ppi (default: "20")

Request basic information about PDF

```
<x:dict>
   <x:overview/>
</x:dict>
```

If contained, document information and results of the performed profile will be available for using them in the HTML template.

Visual comparision of original and processed file

```
<x:compare>
   <x:document_a resolution=20/>
   <x:document_b resolution=20/>
   <x:diffresult resolution=20/>
</x:compare>
```

Include compare tree if comparison resources are used inside index.html.

Attribut

- `resolution` : resolution used in ppi for rendering the comparision (dafault: "20")

Create a XML report

```
<x:xmlreport path="xml/report.xml" inkcovres="10" inkcovbox="CropBox"/>
```

Requests a XML report of the performed profile to extract additional information using JavaScript which can be used in the report. The ink coverage is determined only if one of the corresponding parameters is present.

Options

- `path` : Destination path for XML report (Default: "xml/report.xml")
- `inkcovres` : Resolution for ink coverage (Default: "10")
- `inkcovbox` : Page box for ink coverage (Default: "CropBox")

Create a JSON report

```
<x:jsonreport path="json/report.json" quickcheck="default" />
```

Click to copy

Requests a JSON report to extract additional information.

Options

- `path` : Destination path for JSON report (Default:" json/report.json")
- `quickcheck` : Optional relative path to a custom quickcheck config in the template folder. This quickcheck config will replace the internal default config (quickcheck="none" will disable the default config). More about the default Quick Check configuration file can be found here.

Page previews

```
<x:preview resolution="20" pageselector="1"/>
```

If this parameter is present in the manifest.xml, page previews are generated for the selected pages. The preview of the first selected page is added to the overview page of the preflight report for visual representation of the PDF file.
The preview images are exported into the file system as png files. The folder in the intermediate HTML is:

`./img/cals_pages/../../cals_src_a.png`

Options

- `resolution` : Resolution in ppi for ink amount previews (Default: "20")
- `pageselector` : Selected pages for previews (Default: "1")

Optional information

```
<x:inkamountheatmaps … />
<x:inkcoverage … />
<x:spotcolors … />
<x:separations … />
<x:pageinfo … />
```

Click to copy

There are five new parameters in the results section, that generate optional information about the document. Read

more about that in the next article: HTML based custom reports – generate optional information.

19.3 HTML based custom reports – generate optional information

The previous article (HTML based custom reports) explained the different components of the HTML based custom report. In this article you will learn how to add additional pages to your custom HTML report.

Note: The optional pages are available since **pdfToolbox 14 (and pdfaPilot 12)** – The optional pages "Image resolution" and "Small objects" are available since **pdfToolbox 15 (and pdfaPilot 13)**

The optional information

The custom PDF report contains an overview page that summarizes information about the preflighted document, the environment and information about the executed profile with a list of Errors, Infos, Warnings and Fixups. Until pdfToolbox 14 the report did not provide any information about spot colors, page size, ink amount, ink coverage or seperation previews. Now you have the possibility to order seven additional pages in the **manifest.xml** that are then added to your custom PDF report:

- Page information
- Ink coverage
- Ink amount
- Spot color information
- Separation preview
- Small objects
- Image resolution

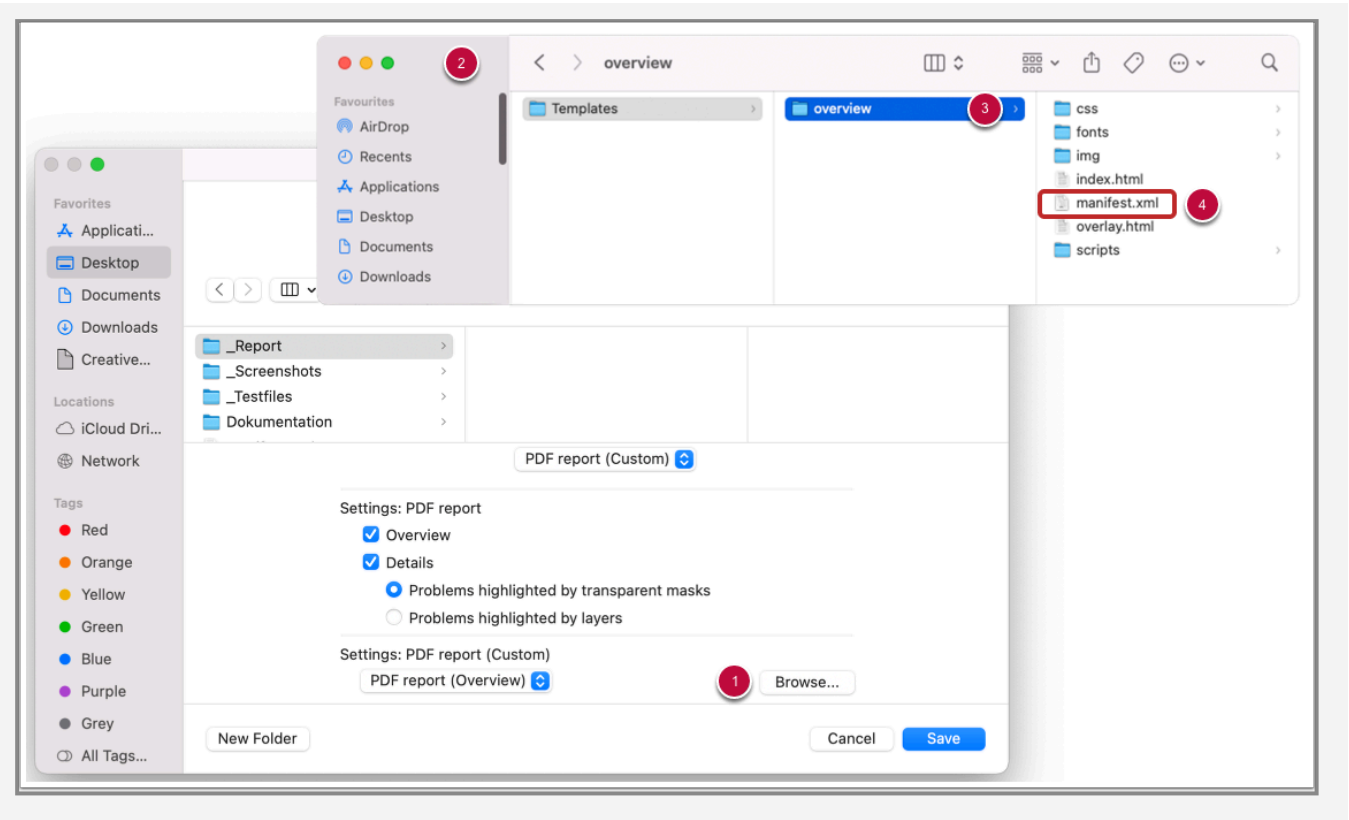
All the parameters are part of the `<x:results>` section of the manifest.xml.

Note: If you do not find the additional information listed below in your manifest.xml, or if you still get

the old layout for the overview page, this may be because you imported the old libraries and resources when you upgraded to pdfToolbox 14/ pdfToolbox 15. If this is the case, you can simply create a new library (including resources) to get the new template, or you can replace the old templates folder with the folder attached below.

How to open the manifest.xml?

The additional information can be ordered manually via the manifest.xml. The following screenshot documents how to get to the manifest.xml:



1. When generating a report, click on the "**Browse…**" button in the settings section.
2. A new window into the file system opens where the Template folder is stored.
3. The **overview** folder is the predefined Template folder.
4. In this folder the **manifest.xml** ist stored.

How to order the optional information in the manifest.xml?

Since these pages are optional, you decide if you want to add one or all pages to your report or not. The default is that none of these information will be generated and added to the report. All five parameters are commented out in the manifest.xml with `<!--` and `-->`:

```
<!-- - Create ink amount heatmap previews and show ink amount page in report
                If present page ink amount previews are generated for the selected pages.

                Paths to the preview files are stored in cals_doc_info.docs[0].pages[i].ink_amount

                resolution  : Resolution in ppi for ink amount previews. Default: "20"
                threshold   : Threshold in % for ink amount previews. Default: "300"
                pageselector: Selected pages for previews, default: "all"
                    (see online documentation for full pageselector syntax)
                show        : Add ink amount section to custom report. Default: "1"

            <x:inkamountheatmaps resolution="20" threshold="300" pageselector="all" show="1"/>
            -->
```

To add one page to your report all you have to do is uncomment the parameter:

```
<!-- - Create ink amount heatmap previews and show ink amount page in report
                If present page ink amount previews are generated for the selected pages.

                Paths to the preview files are stored in cals_doc_info.docs[0].pages[i].ink_amount

                resolution  : Resolution in ppi for ink amount previews. Default: "20"
                threshold   : Threshold in % for ink amount previews. Default: "300"
                pageselector: Selected pages for previews, default: "all"
                    (see online documentation for full pageselector syntax)
                show        : Add ink amount section to custom report. Default: "1"

            -->
            <x:inkamountheatmaps resolution="20" threshold="300" pageselector="all" show="1"/>
```

Now the parameter is uncomment and present in the manifest.xml.

You want to add all optional pages to your report?

If you want to add all pages to your PDF report, we provide a template for download, where all optional information in manifest.xml is already uncommented. Just follow the instructions:

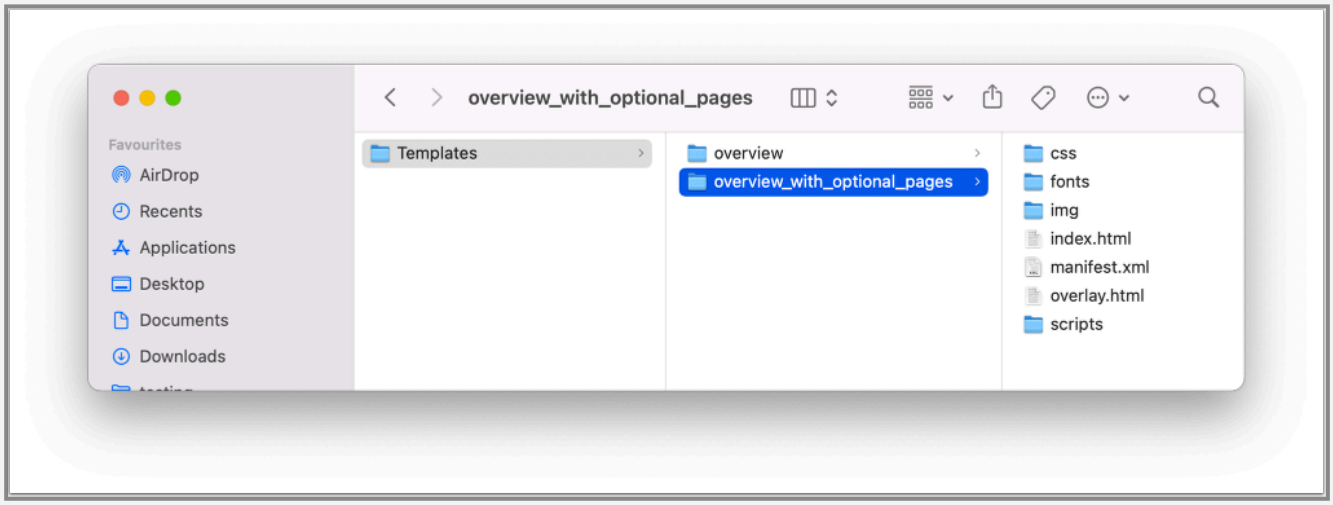
1. Download one of the templates below:

- overview_with_optional_pages.zip: earliest support version = pdfToolbox 14
- overview_with_optional_pages_pdfToolbox15.zip: earliest support version = pdfToolbox 15

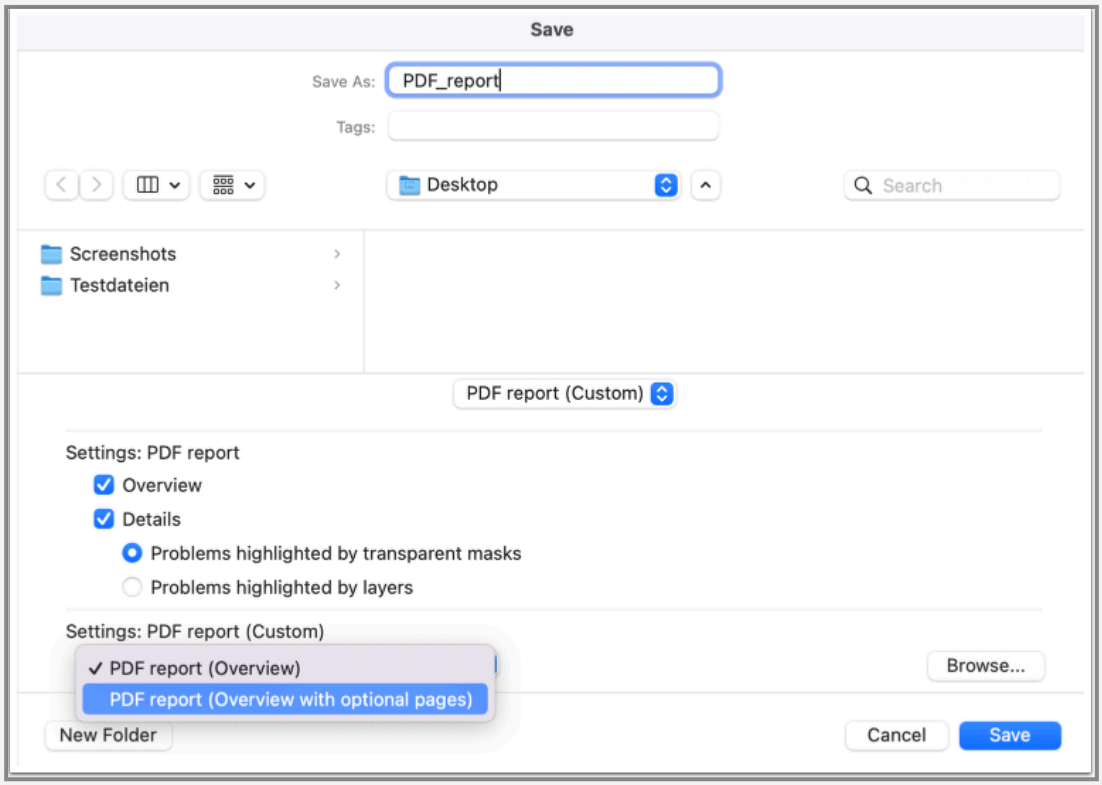
overview_with_optional_pages.zip

overview_with_optional_pages_pdfToolbox15.zip

2. Add the downloaded template to the "Templates" folder.
The screenshot at the beginning of the article shows you how to get to the template folder.



3. In order to display the new template in pdfToolbox, the "Save" dialog must be opened again. Then the template "PDF report (Overview with optional pages)" can be selected in the settings options.



Parameters for the optional report information

The show option

All optional information parameters have a show option.

The default for the show option is **show="1"**. This means that the data will be generated and the optional page will be added to the report.

If you want to use your own template you can set **show="0"**. This will generate the data for the specific parameter, but not the optional report page.

Ink amount heatmaps

```
<x:inkamountheatmaps resolution="20" threshold="300" onlyproblems="1" pageselector="all" show="1"/>
```

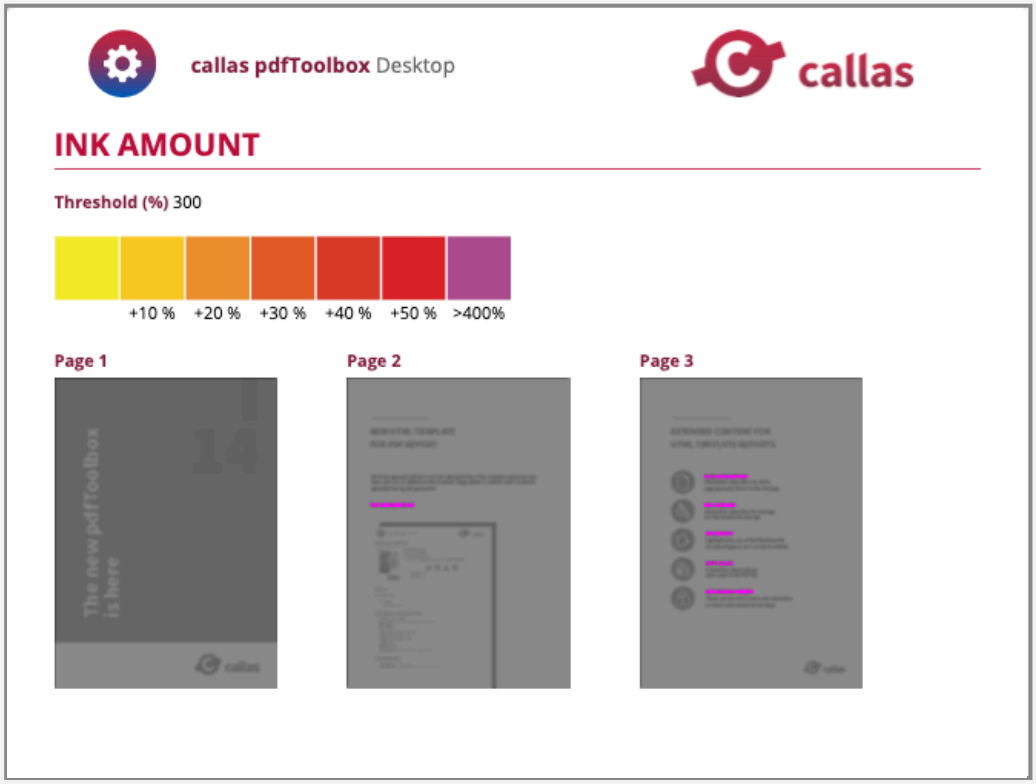
Click to copy

Creates ink amount heatmap previews and show ink amount for all pages or selected pages by highlighting any area of the file that goes over a certain threshold. The visuals that are created are similar to the visualizer mode "Ink coverage".

Parameters

- `resolution` : Resolution in ppi for ink amount previews (Default: "20")
- `threshold` : Threshold in % for ink amount previews (Default: "300")
- `pageselector` : Selected pages for previews (Default: "all") (click here for full pageselector syntax)
- `onlyproblems` : Only emit pages that have problems (Default: "1") (earliest support version = pdfToolbox 15)
- `show` : Add ink amount section to custom report (Default: "1"), show="0" will only generate the data

Example ink amount report page



Ink coverage

```
<x:inkcoverage resolution="10" pagebox="CropBox" pageselector="all" listpages="1"
show="1"/>
```

Click to copy

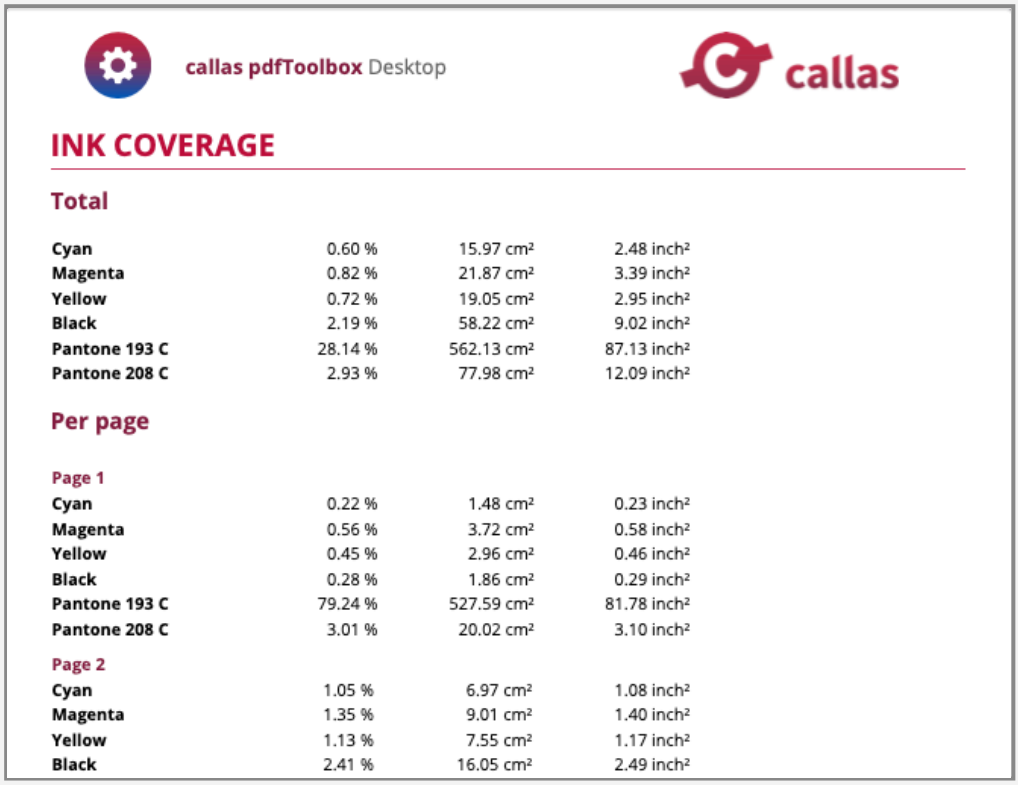
If present the page "Ink coverage" is added to the PDF report. The ink coverage information can refer to all or selected pages. For large files, the creation of this page may take some time, since all pages must be rendered.

Parameters

- `resolution`: Resolution in ppi for ink coverage determination (Default: "10")
- `pagebox`: Page box for ink coverage (Default: "CropBox")

- `pageselector`: Selected pages for ink coverage information in custom report (Default: "all") (click here for full pageselector syntax)
- `listpages`: Add ink coverage information per Page (Default: "1"), listpages="0" will only list ink coverage for the hole document (total)
- `show`: Add ink coverage section to custom report (Default: "1"), show="0" will only generate the data

Example ink coverage report page



callas pdfToolbox Desktop

callas

INK COVERAGE

Total

Cyan	0.60 %	15.97 cm²	2.48 inch²
Magenta	0.82 %	21.87 cm²	3.39 inch²
Yellow	0.72 %	19.05 cm²	2.95 inch²
Black	2.19 %	58.22 cm²	9.02 inch²
Pantone 193 C	28.14 %	562.13 cm²	87.13 inch²
Pantone 208 C	2.93 %	77.98 cm²	12.09 inch²

Per page

Page 1			
Cyan	0.22 %	1.48 cm²	0.23 inch²
Magenta	0.56 %	3.72 cm²	0.58 inch²
Yellow	0.45 %	2.96 cm²	0.46 inch²
Black	0.28 %	1.86 cm²	0.29 inch²
Pantone 193 C	79.24 %	527.59 cm²	81.78 inch²
Pantone 208 C	3.01 %	20.02 cm²	3.10 inch²
Page 2			
Cyan	1.05 %	6.97 cm²	1.08 inch²
Magenta	1.35 %	9.01 cm²	1.40 inch²
Yellow	1.13 %	7.55 cm²	1.17 inch²
Black	2.41 %	16.05 cm²	2.49 inch²

Spot color information

```
<x:spotcolors pageselector="all" show="1"/>
```

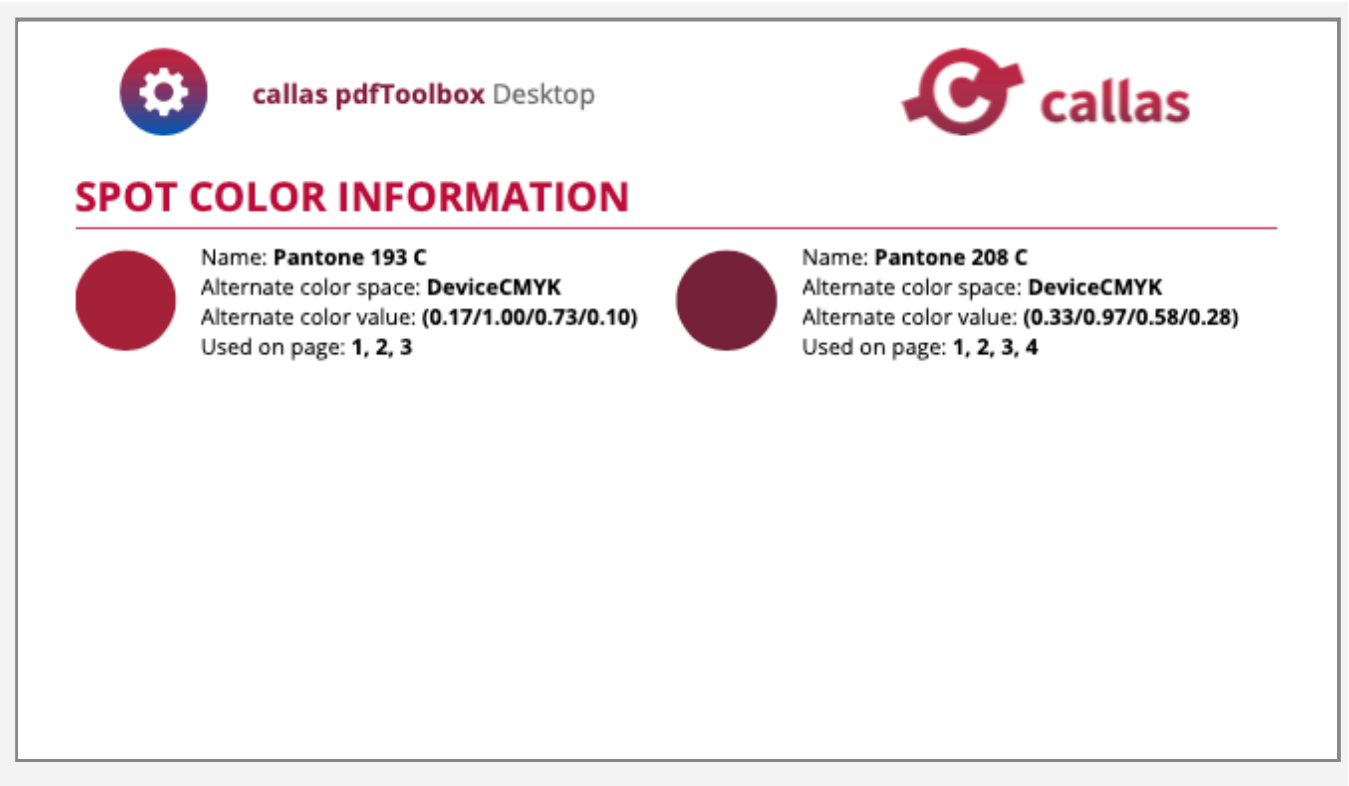
Click to copy

If present the spot color information is generated. The spot color information page will contain all used spot colors including the spot color name, the alternate color space and value and on which pages the color has been used.

Parameters

- `pageselector`: Selected pages for spot color information in custom report, default: "all" (click here for full pageselector syntax)
- `show`: Add spot color section to custom report (Default: "1"), show="0" will only generate the data

Example Spot color report page



Seperation preview for Process and Spot colors

```
<x:separations type="spotifpresent" resolution="20" pageselector="all" show="1"/>
```

Click to copy
Creates separation previews for process separation (Cyan, Magenta, Yellow and Black) and spot color separation.

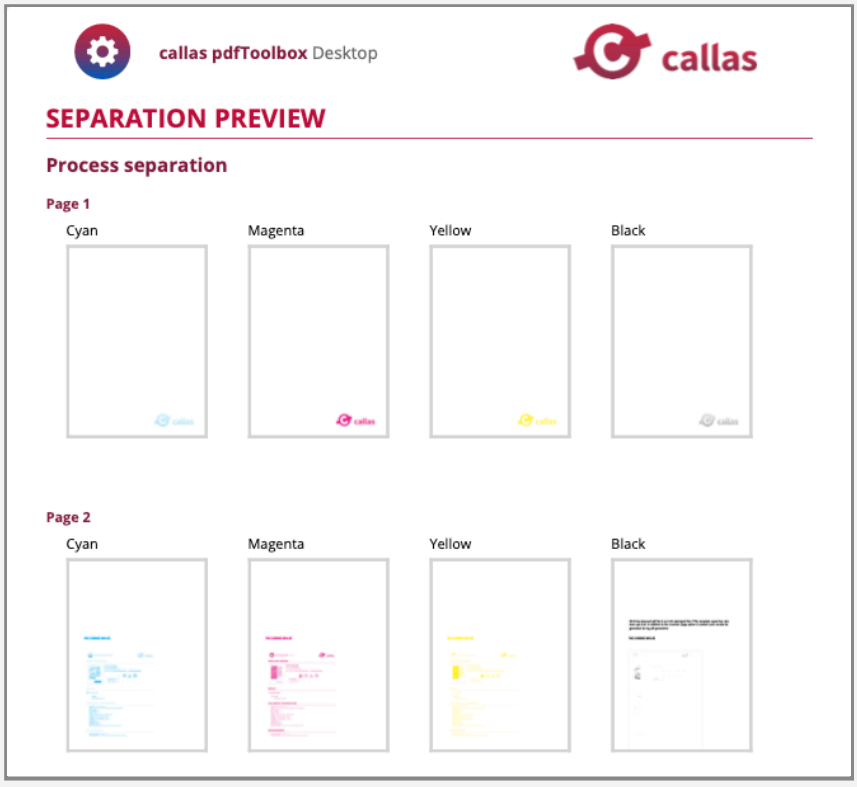
Parameters

- `resolution`: Resolution in ppi for separation previews (Default: "20")
- `type`: Type for separation views, any of (earliest support version = pdfToolbox 15):

- "all": all separation views (Process separation and spot color separation pages are always created)
- "process": Only process color separation views
- "spot": Only spot color separation views
- "spotifpresent": Process color separation views and only spot color separation views if document has spot color (Default)

- `pageselector`: Selected pages for previews (Default: "all") (click here for full pageselector syntax)
- `show`: Add separation preview section to custom report (Default: "1"), show="0" will only generate the data

Example separation preview report page



Page information

```
<x:pageinfo pageselector="1" resolution="20" safetyzoneinside="3mm" safetyzoneout-
side="3mm" usebleedbox="0" unit="mm" show="1"/>
```

Click to copy

Generates a page information for the first page by default. This provides the sizes of all page geometry boxes, as well as an information about the page rotation and the page scaling factor. If a page scaling factor is set, you also get the effective sizes of the geometry boxes. A preview images with a predefined safety zone and highlighted TrimBox and BleedBox is also part of the page. Additionally you get the information if all pages of the document have the same size or not.



If multiple pages are specified in the pageselector, only the first selected page will be considered for the template report.

Parameters

- `resolution`: Resolution in ppi for safety zone preview (Default: "20")
- `pageselector`: Selected pages for previews (Default: "1") (click here for full pageselector syntax)
- `safetyzoneinside`: Safety zone inside (Default: "3 mm")
- `safetyzoneoutside`: Safety zone outside (Default: "3 mm")
- `usebleedbox`: Use bleed box if existing (Default: "0")
- `unit`: Unit for page geometry boxes. Options: mm, pt, cm, inch (Default: "mm")
- `show`: Add page information section to custom report (Default: "1"), show="0" will only generate the data

Example page information report page

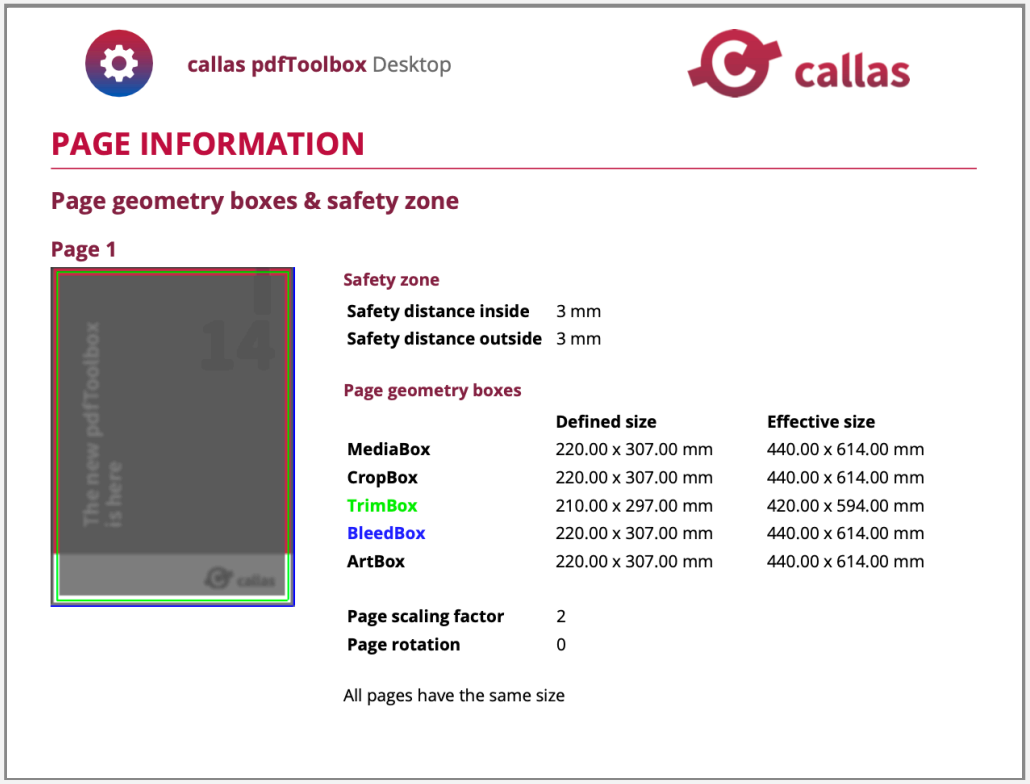


Image resolution

```
<x:imageresolution type="img" resolution="20" imgthreshold="150" bmpthresh-
old="550" onlyproblems="1" pageselector="all" show="1"/>
```

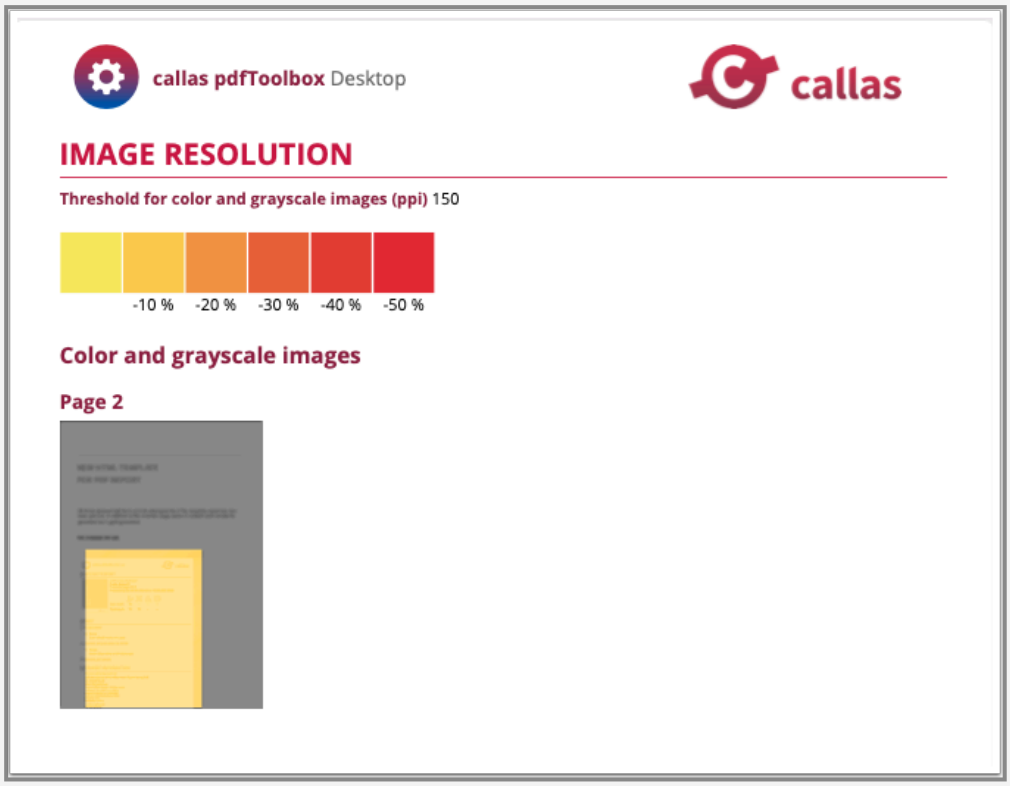
Click to copy

Creates the Image Resolution report page, which shows all pages where an image resolution is below the specified threshold by highlighting those images. The visuals that are created are similar to the visualizer mode "Image resolution".

Parameters

- `type`: Image type for image resolution views, any of:
  - "all": all image resolution views
  - "img": image resolution views (Default)
  - "bmp": bitmap resolution views

- `resolution`: Resolution in ppi for image resolution previews (Default: "20")
- `imgthreshold`: Threshold in ppi for image resolution previews (Default: "150")
- `bmpthreshold`: Threshold in ppi for bitmap resolution previews (Default: "550")
- `onlyproblems`: Only emit pages that have problems (Default: "1")
- `pageselector`: Selected pages for previews (Default: "all") (click here for full pageselector syntax)
- `show`: Add page information section to custom report (Default: "1"), show="0" will only generate the data



Small objects

```
<x:smallobjects type="all" threshold="medium" resolution="20" onlyproblems="1"
pageselector="all" show="1"/>
```

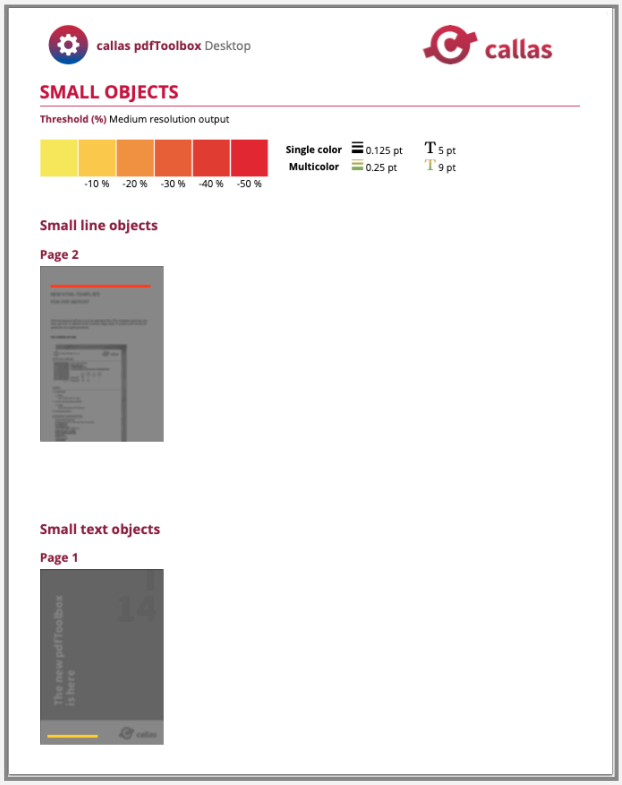
Click to copy

Creates the Small Objects report page and displays thin line objects and/or small text objects for all pages or selected pages by highlighting any object that is below a specified

threshold. The visuals that are created are similar to the visualizer mode "Small objects".

Parameters

- `type`: Type of small objects, any of:
  - "all": all small object views (Default)
  - "text": small text objects below threshold
  - "lines": small vector objects below threshold
- `resolution`: Resolution in ppi for image resolution previews (Default: "20")
- `threshold`: Threshold for small object coverage highlighting, any of:
  - "low": low resolution output (0.2 pt single color lines & 0.5 pt multicolour lines, 8 pt single color text & 10 pt multicolour text)
  - "medium": medium resolution output (Default) (0.125 pt single color lines & 0.25 pt multicolour lines, 5 pt single color text & 9 pt multicolour text)
  - "high": high resolution output (0.125 pt single color lines & 0.25 pt multicolour lines, 5 pt single color text & 8 pt multicolour text)
- `onlyproblems`: Only emit pages that have problems (Default: "1")
- `pageselector`: Selected pages for previews (Default: "all") (click here for full pageselector syntax)
- `show`: Add page information section to custom report (Default: "1"), show="0" will only generate the data


callas pdfToolbox Desktop
callas
SMALL OBJECTS
Threshold (%) Medium resolution output
-10 %
-20 %
-30 %
-40 %
-50 %
Single color
Multicolor
0.125 pt
0.25 pt
5 pt
9 pt
Small line objects
Page 2
Small text objects
Page 1

19.4 HTML based custom reports – control custom parameters via JavaScript

The `app.context.report.manifest` JS object allows you to customize the manifest.xml for a custom PDF report from within a process plan or profile. You can use this JS object within a variable in pdfToolbox to enable and disable specific sections of the manifest.xml and to set new values for parameters using JavaScript.

Once an `app.context.report.manifest` JS object is specified, the parameters set in that object will overwrite the values in manifest.xml. For any parameter that does not have a custom value set in `app.context.report.manifest`, the values in manifest.xml are used.

Deactivate a specific section in manifest.xml via the app.context.report.manifest object

To deactivate a specific section in the manifest.xml there are two different options:

```
//Option 1 (null)

app.context.report.manifest = {
        "resources" : {
                "results": {
                        "pageinfo": null
                }
        }
};


//Option 2 (false)


app.context.report.manifest = {
        "resources" : {
                "results": {
```

```
                        "pageinfo": false
                }
        }
};
```

Click to copy

Activate a specific section in manifest.xml via the app.context.report.manifest object

To enable a specific section in manifest.xml there are 3 options:

```
//Option 1 (empty object)


app.context.report.manifest = {
        "resources" : {
                "results": {
                        "pageinfo": {}
                }
        }
};


//Option 2 (true)


app.context.report.manifest = {
        "resources" : {
                "results": {
                        "pageinfo": true
                }
        }
};


//Option 3 (at least one parameter is specified for that section)

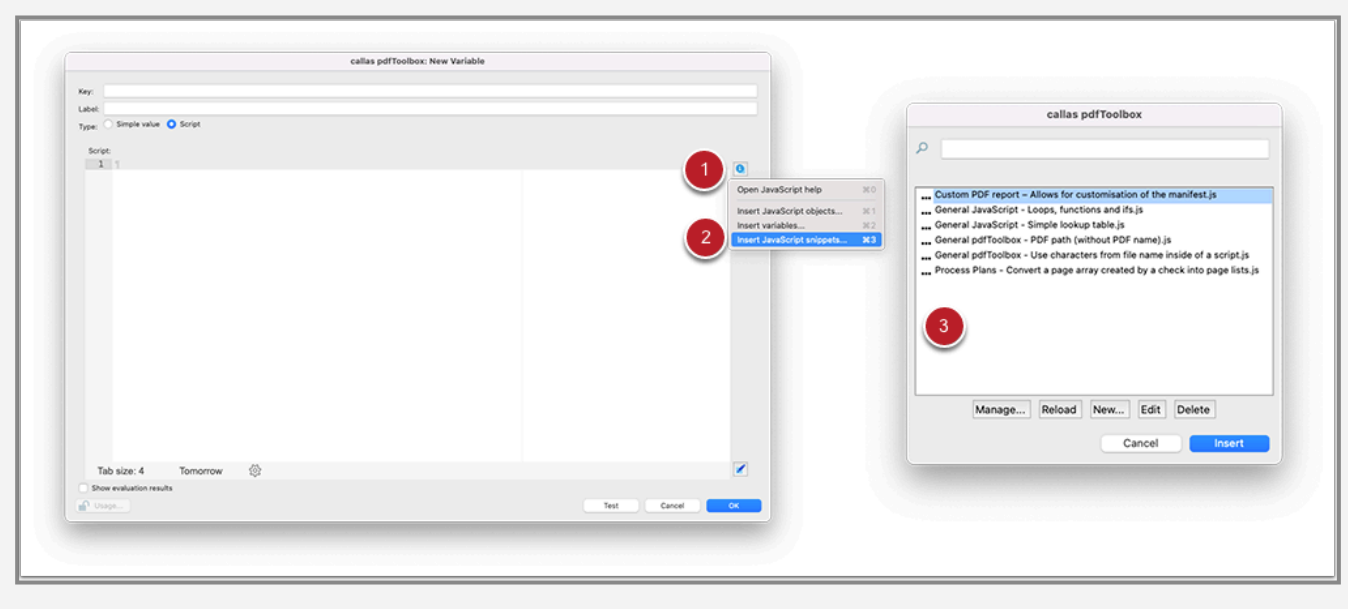

app.context.report.manifest = {
        "resources" : {
```

```
                "results": {
                        "pageinfo": {
                                "resolution": 72
                        }
        }
};
```

Click to copy

app.context.report.manifest is available as JavaScript snippet

To simplify use, the `app.context.report.manifest` JS object is available as a code snippet in pdfToolbox. After inserting the JavaScript snippet, you receive the JS object with all manifest.xml sections and default values. This can then be easily adapted to your own requirements.



1. Click on the blue info icon next to the script editor in pdfToolbox
2. Click on "Insert JavaScript snippets…"
3. A new window opens where you can insert the `app.context.report.manifest` JS object

Complete app.context.report.manifest

JavaScript object with all report parameters

```
app.context.report.manifest = {
"settings": {
  "keeptemp": false,
  "defaults": {
    "resolution": 20
  }
},
"resources": {
  "dict": {
    "overview": true
  },
  "results": {
    "xmlreport": {
      "path": "xml/report.xml",
      "inkcovres": 10,
      "inkcovbox": "CropBox"
    },
    "jsonreport": {
      "path": "json/report.json",
      "quickcheck": "default"
    },
    "jsonv2report": {
      "path": "jsonv2/report.json",
      "inkcovres": 10,
      "inkcovbox": "CropBox",
      "quickcheck": "none"
    },
    "preview": {
      "resolution": 20,
      "pageselector": "1"
    },
    "inkamountheatmaps": {
      "resolution": 20,
      "threshold": 300,
      "onlyproblems": true,
      "pageselector": "all",
      "show": true
    },
    "inkcoverage": {
```

```
        "resolution": 10,
        "pagebox": "CropBox",
        "pageselector": "all",
        "show": true
      },
      "spotcolors": {
        "pageselector": "all",
        "show": true
      },
      "separations": {
        "resolution": 20,
        "type": "spotifpresent",
        "pageselector": "all",
        "show": true
      },
      "pageinfo": {
        "pageselector": "1",
        "resolution": 20,
        "safetyzoneinside": "3mm",
        "safetyzoneoutside": "3mm",
        "usebleedbox": false,
        "unit": "mm",
        "show": true
      },
      "imageresolution": {
        "resolution": 20,
        "type": "img",
        "imgthreshold": 150,
        "bmpthreshold": 550,
        "onlyproblems": true,
        "pageselector": "all",
        "show": true
      },
      "smallobjects": {
        "resolution": 20,
        "type": "all",
        "threshold": "medium",
        "onlyproblems": true,
        "pageselector": "all",
        "show": true
      }
    }
  }
```

```
};
```

Click to copy

19.5 Dump variables report (using JSON)

Beginning with pdfToolbox 10, a new report type is available on command line (and in Switch):

```
VARDUMP:   Dumps the app.vars object into JSON
```

Click to copy

This report dumps all Variables and their values in a JSON file. The other report options can be used e.g. in order to specify the location and the name with which the JSON report is created.

The JSON file contains a "vars" object and key value pairs for all Variables.
This report type can be helpful for use cases, where JavaScript Variables are e.g. calculating multiple thresholds based on a single Variable defined during runtime or when a JavaScript Variable is deriving information about the PDF itself.

The report can be used in Switch as a dataset or in other workflow applications, MIS-systems or any other applications that integrate pdfToolbox in order to export any kind of information that is available in the pdfToolbox internal JavaScript object.

```
pdfToolbox Viewing Distance related checks.kfpx <any PDF> --report=vardump
```

Click to copy

The content of the vardump JSON for this CLI call will be:

```
{
   "vars" :
   {
      "Calcs_for_LFP_Preflight_-_viewing_distance" : 200,
      "eff_min_fontsize" : 200,
      "eff_min_imageresolution" : 40,
      "input_scalingfactor" : 100,
      "input_viewingdistance" : 10
   }
}
```

And will vary if the values of the Variables "input_viewingdistance" and "input_scalingfactor" are changed using --setvariables=...

Note: The used Profile is part of the "Shapes, Variables, JavaScript, Place content" Library and contained in the Desktop version.

Viewing_Distance_related_checks.kfpx

Billboard_-_Penguins_for_Life.pdf

19.6 Command line options for multi-language support

Until pdfToolbox 9, all reports in a call needed to have the same language, for example --language=en.
From the pdfToolbox 10 onwards, the use of different languages in reports created on the same pdfToolbox run is possible. For example

```
--report=XML,language=en,path=<output path> --report=MASK,language=fr,path=<output path>
```

Seven languages are completely localized in pdfToolbox:

- English, German, French,Spanish, Italian, Japanese and Chinese

It could led to a problem of appearance of DictKeys for unknown languages or not fully localized languages in pdfToolbox reports.
Since pdfToolbox 10, the DictKey mechanism has been extended, which allows customers to add translations to inactive languages and allows creation, maintenance and use of custom dicts with additional strings for any Profiles (or other objects) for active, inactive or any other language.
More about custom dicts below:

Custom Dict files

Custom Dict files can be used to add or modify strings that are used in reports. Since pdfToolbox 10.0 it is allowed for custom dicts to cover strings for custom profiles or unknown Dictkeys in a particular language.

Creation of custom dicts

In the following example, a custom dict is created for strings of a non localised language to be added/replaced in a predefined Profile (kfpx). This can be achieved using the command line argument:

`--createcustomdict <Path to Profile file>`

```
./pdfToolbox --createcustomdict /var/Profiles/PDF analysis/List potential font
problems.kfpx
```

Click to copy

This creates an XML for the profile (kfpx).

```
<?xml version="1.0" encoding="UTF-8" ?>
<callas>
        <dict version="3.0" lang="en">
                <keys>
                        <group key="P">
                                <entry key="8_Listpotentialfontproblems">
                                        <value var="long">List potential font prob-
lems</value>
                                        <value var="short">Lists possible font is-
sues.</value>
                                </entry>
                        </group>
```

Click to copy

A user can add his own custom language and values (long for Name and short for Comment as in this example).

```
<?xml version="1.0" encoding="UTF-8" ?>
<callas>
        <dict version="3.0" lang="in">
                <keys>
            <group key="CUSTOMDICT">
                <entry key="भारत">
                    <value var="long">भारत</value>
                    <value var="short">भारतभारतभारत</value>
                </entry>
            </group>
```

Click to copy

The custom dictionary can now be used while generating reports using --customdict

```
./pdfToolbox --report=xml --customdict=./List potential font problems.xml --lan-
guage=in ./List potential font problems.kfpx ./123.pdf
```

Click to copy

Profile name and comment without the appearance of DictKeys in the XML report (other report types will work as well of course):

```
<profile_name>भारत</profile_name>
                <profile_comment>भारतभारतभारत</profile_comment>
```

Click to copy

19.7 Modify strings for internal syntax checks using Custom Dicts

If you are coming from the previous article, you might already know that you can use Custom Dict files to add or modify strings that are used in reports. In case you want to modify the strings for our internal syntax checks, this is also possible using custom dictionaries.

In order to modify the strings for the internal syntax checks, a custom dict (resulting in an xml file) is first created using --createcustomdict command, like below:

```
./pdfToolbox --createcustomdict ./Downloads/Only\ syntax\ checks.kfpx
```

Click to copy

To modify the strings for our internal syntax checks, you will have to know our internal DictKeys, **which you can find in the attached sample file.** An example where the DictKey 'SyntaxCheckMissingFont' is added to the custom dict file is shown below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<callas>
        <dict version="3.0" lang="en">
                <keys>
                        <group key="PRCWzDocu">
                                <entry key="SyntaxCheckMissingFont">
                                        <value var="long">Looks like there is a
missing font</value>
                                        <value var="short">Missing font</value>
                                </entry>
                .
     .
   .
```

Click to copy

On executing a (faulty) PDF using --customdict like this:

```
./pdfToolbox --report=XML --customdict=./Downloads/Only\ syntax\ checks.xml ./Down-
loads/Only\ syntax\ checks.kfpx ./Downloads/defect.pdf
```

Click to copy

The result now shows the modified string:

```
Profile        ./Downloads//Only syntax checks.kfpx
Input         ./Downloads//defect.pdf
Pages         10
Progress         1         %
Progress         19         %
Hit         Error         Looks like there is a missing font
Progress         22         %
.
.
```

Click to copy

Attached you will find a sample custom dict (Only_syntax_checks.xml) derived from a simple Profile that contains all possible Dict Keys

Only_syntax_checks.xml

19.8 XML report: Convert pt to mm

Users can convert pt into mm "on the fly"in an XML report's page box dimensions.

Understanding the basics

A sample Profile when run against a sample PDF like below:

```
/pdfToolbox /Image_resolutions_10_ppi.kfpx /test_X-1a.pdf --report=XML
```

Click to copy
generates an XML report with page dimensions in pt:

```
..omitted..
        <pages>
                <page id="PAG1" nr="1"
                mediabox="0.0/0.0/595.280029/841.890015"
                effective_mediabox="0.0/0.0/595.280029/841.890015"
                cropbox="0.0/0.0/595.280029/841.890015"
                effective_cropbox="0.0/0.0/595.280029/841.890015"
                trimbox="0.0/0.0/595.280029/841.890015"
                effective_trimbox="0.0/0.0/595.280029/841.890015">
..omitted..
```

Click to copy

Transform the data with XSLT report

In the XML reports, most measurements are in pt. Most users of the report do some code for extraction and convert the values easily. The XSLT report is exactly that: It uses the internally geneated XML report and is converting the contained measurements.

Usage:

```
/pdfToolbox /Image_resolutions_10_ppi.kfpx /test_X-1a.pdf --report=XSLT=trans-
form_pageboxes_into_mm.xslt
```

Click to copy

The XML data can be changed or extended with XSLT (attached below) as needed.

Report:

```
..omitted..
        <pages>
                <page id="PAG1" nr="1"
                mediabox="0/0/210/297"
                effective_mediabox="0/0/210/297"
                cropbox="0/0/210/297"
                effective_cropbox="0/0/210/297"
                trimbox="0/0/210/297"
                effective_trimbox="0/0/210/297">
..omitted..
```

Click to copy

transform_pageboxes_into_mm.xslt

19.9 JSON report – based on JS object

With pdfToolbox 13, beside the existing human-readable PDF reports, the JSON report has been added. This report type allows easier access to details about the PDF and its processing. Parsing the structure might be simpler than with the existing XML report for several automation environments.

You have the option to choose between two different JSON reports:

1. JSON report – based on JS object: Structure is similar to the `app.doc.results` array available in JavaScript Variables within callas products.
2. JSON report – similar to XML report: Structure is similar to the XML report and lists resources that are referenced from the "result" section.

The JSON report is available:

- After the normal Profile execution in Desktop in the "Create Report" functionality of the results dialog
- In Process plans as a report type in transitions after suitable Steps (Profile, Fixup, Check)
- On the CLI for Profile execution using `--report=JSON` (all possible parameters for the JSON report on the CLI are are documented here).

Components of the JSON report



The JSON report contains five sections:

1. `doc` : information regarding to the document . These informations are independent of the executed profile.
2. `env` : information about various aspects of the environment such as platform and version.
3. `result` : shows the executed steps (Checks/Fixups) and the number of corrections.
4. `settings` : reflects the options selected for the report generation.
5. `steps` : lists name, comment and type from all steps.

The doc section

```
{
   "doc" : {
      "documentFileName" : "Testfile_JSON.pdf",
   (1) "info" : {},
   (2) "metadata" : "",
      "numPages" : 1,
   (3) "pages" : [],
      "path" : "/Users/sarahnitschmann/Desktop/Testfile_JSON.pdf",
   (4) "quickcheck" : {}
   },
```

1. `info` : information about the PDF file e. g. creation date, PDFX version, Titel
2. `metadata` : contains the XMP metadata of the PDF
3. `pages` : contains the dimension (relative and absolute) and the inks of all pages
4. `quickcheck` : since pdfToolbox 14 a predefined Quick Check configurations file is included in the report creation. This file provides the JSON report with additional information about the document, such as used ressources (images, fonts, spot colors), annotations, embedded files, layers or the output intent. If a JSON report is generated within a Process Plan or on CLI, various settings can be made regarding the Quickcheck configurations file.

The result section



```
"result" : {
   "checks" : [
      {
         "conditions" : [
            {
               "id" : "CSIMAGE_Group::CSIMAGE_BitsPerColourComponent",
               "name" : "Bits per color component"
            },
            {
               "id" : "CSIMAGE_Group::CSIMAGE_Resolution",
               "name" : "Image resolution"
            }
         ],
         "customId" : "",
         "hits" : [
            {
               "llx" : 36,
               "lly" : 204.09500122070312,
               "page" : 0,
               "triggers" : [
                  {
                     "id" : "CSIMAGE_Group::CSIMAGE_Resolution",
                     "value" : 300
                  },
                  {
                     "id" : "CSIMAGE_Group::CSIMAGE_BitsPerColourComponent",
                     "value" : 8
                  }
               ],
               "type" : "Image",
               "urx" : 313.22799682617188,
               "ury" : 805.88995361328125
            }
         ],
         "id" : "R69e4832a5f9e805a1534f34e1a7f6739",
         "name" : "Resolution of color and grayscale images is greater than 250 ppi",
         "numHits" : 1,
         "pageNumbers" : [ 0 ],
         "severity" : 3
      }
   ],
   "fixups" : [
      {
         "customId" : "",
         "id" : "Fcb4d36aa6fefdfc771e6d978c201c07c",
         "name" : "Convert all spot colors to CMYK",
         "numFailed" : 0,
         "numSucceeded" : 1
      }
   ],
   "numErrors" : 1,
   "numInfos" : 0,
   "numWarnings" : 0,
   "profile" : "Convert Spot color",
   "reports" : [],
   "resultFiles" : []
},
```


1. `checks` : lists all checks which created a hit, whereas Checks without hits will not be listed. If a Property used in the respective Check provides a trigger value, it will also be listed.
2. `fixups` : lists all executed Fixups, the number of corrections is included. For process plan execution only the last sequence step will be listed in the result section.

The steps section

```
"steps" : [
    {
        "comment" : "Converts all Spot Color to CMYK",
        "name" : "Convert Spot color",
        "type" : "profile"
    }
]
```

Under the `steps` section all executed steps will be listed with name, comment and type. Regarding Process Plans: unlike the `result` section, the `steps` section lists all sequence steps contained in a Process Plan (not just the last one).

19.10 JSON report – similar to XML report

The JSON report type allows easier access to details about the PDF and its processing. Parsing the structure might be simpler than with the existing XML report for several automation environments.

With pdfToolbox 15, an additional JSON report has been added. This gives you the option to choose between two different JSON reports:

1. JSON report – based on JS object: Structure is similar to the `app.doc.results` array available in JavaScript Variables within callas products.
2. JSON report – similar to XML report: Structure is similar to the XML report and lists resources that are referenced from the "result" section.

Advantages of the JSON v2 report

- Access to additional result files such as SaveAsImage or reports
- Improved overview of the process steps
- Listing of only existing plate names (done by analysis); example: only Black for CMYK

The JSON v2 report is available:

- After the normal Profile execution in Desktop in the "Create Report" functionality of the results dialog: **JSON report – similar to XML report**
- In Process Plans as a report type in transitions after suitable Steps (Profile, Fixup, Check): **JSONv2**
- On the CLI for Profile execution using `--report=JSONV2` (all possible parameters for the JSON v2 report on the CLI are are documented here).

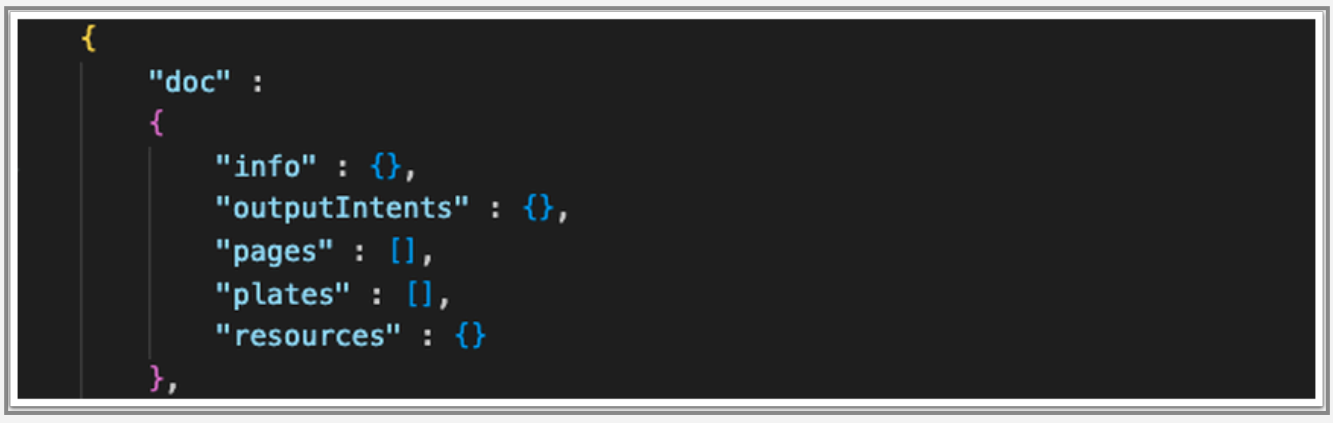
Components of the JSON v2 report



The JSON report contains four sections:

1. `doc` : Information about the document. This information is independent of the executed profile.
2. `env` : Information on various aspects of the environment such as platform, pdfToolbox version and license details.
3. `result` : Lists the executed process steps.
4. `settings` : Reflects the options selected for the report generation.

The doc section



1. `info` : Information about the PDF file e.g. creation date, PDFX version, producer.

2. `outputIntents` : Contains a reference to the output intent information stored in "resources".
3. `pages` : Contains the dimensions (relative and absolute) and the plates of all pages.
4. `plates` : Lists all plates in the PDF document.
5. `resources` : Contains all resource information for annotations, bookmarks, colorspaces, embedded files, fonts, form XObjects, images, layers, output intents, patterns and smooth shades.

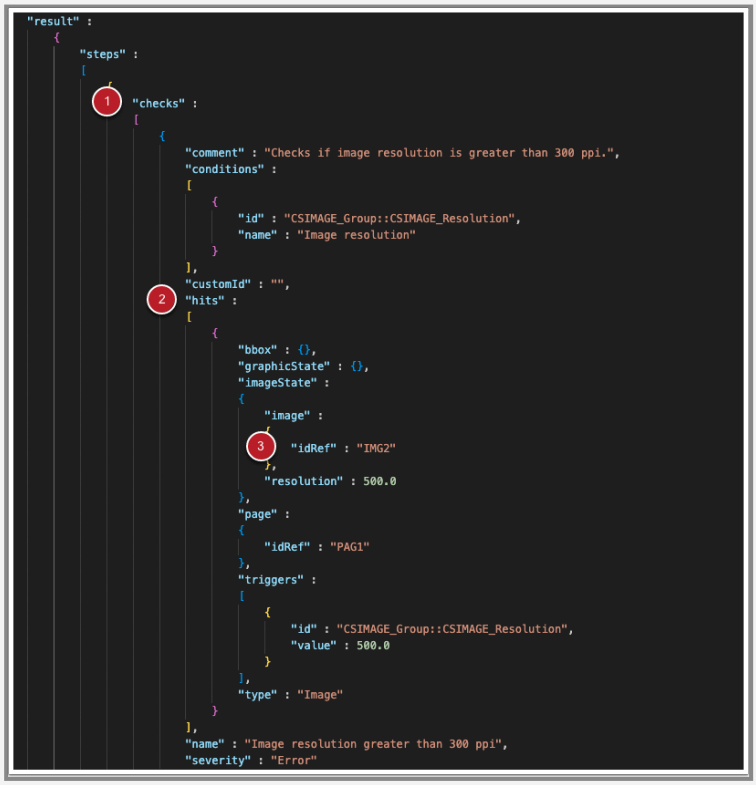
Syntax of "resources"

1. `id` : Each document resource has a unique ID name for identification.
2. `idRef` : Referenced ID to a resource ID.



The result section

The `result` section lists all executed steps in the `steps` section. The following example shows the result of a preflight check that detects images with a resolution greater than 300 ppi:



1. `checks` : Lists all Checks which created a hit, whereas Checks without hits will not be listed.

    2. `hits` : If a Property used in the respective Check provides a trigger value, it will be listed.

        3. `idRef` : The hit refers to a specific resource ID that is listed in the "doc" section under "resources". This makes it possible to identify which object on which page triggered the hit.

The settings section

This section shows the options selected for the report generation:

`allchecks` : Include Checks without hits. By default Checks are only listed when they generate at least one hit.
`allfixups` : Include Fixups without changes. By default Fixups are only listed when they modify any objects.
`compact` : The JSON report will be written in a single line.
`variables` : All used Variables will be listed as well.

```
"settings" :
{
    "allchecks" : false,
    "allfixups" : false,
    "compact" : false,
    "variables" : false
}
```

20. Place content

20.1 Place dynamic page numbers

Since version 8.0, pdfToolbox supports a variety of information using static as well dynamic text in a PDF file. Different fonts, sizes, colors, placement options and other design specifications can be added.

In this tutorial the dynamic text functionality is used. Page numbers are added in the PDF file.

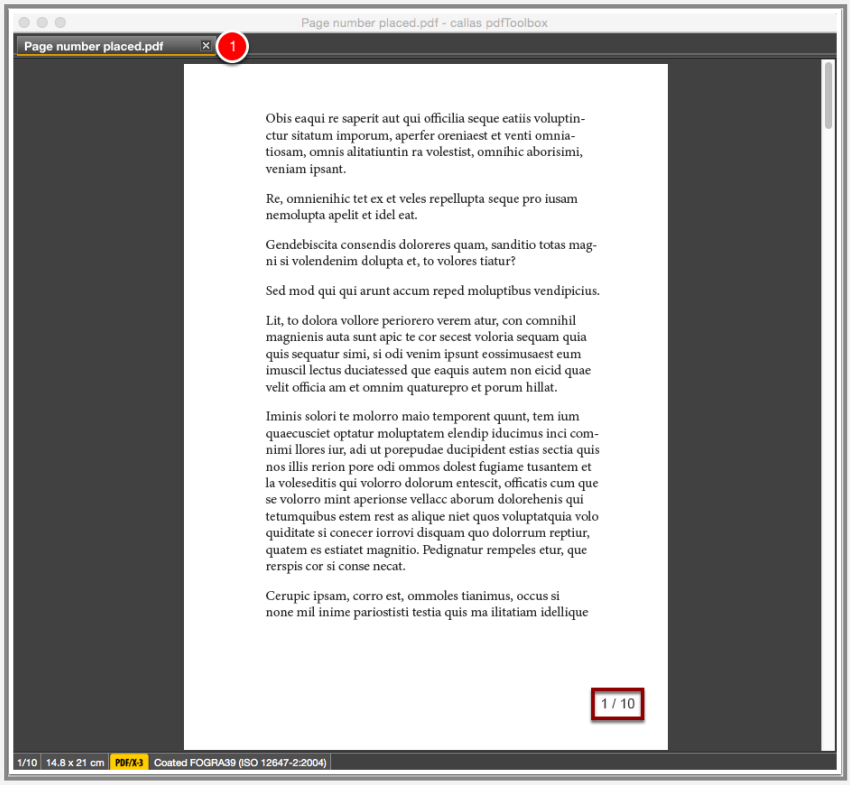
Apply the predefined Fixup

Apply the Fixup "Place page number in left or right corner" on "Page numbers_Demo file.pdf"



1. In the search field, look for "Place page number in left or right corner".
2. Select the predefined Fixup "Place page number in left or right corner".
3. Click "Fix".

The processed PDF file

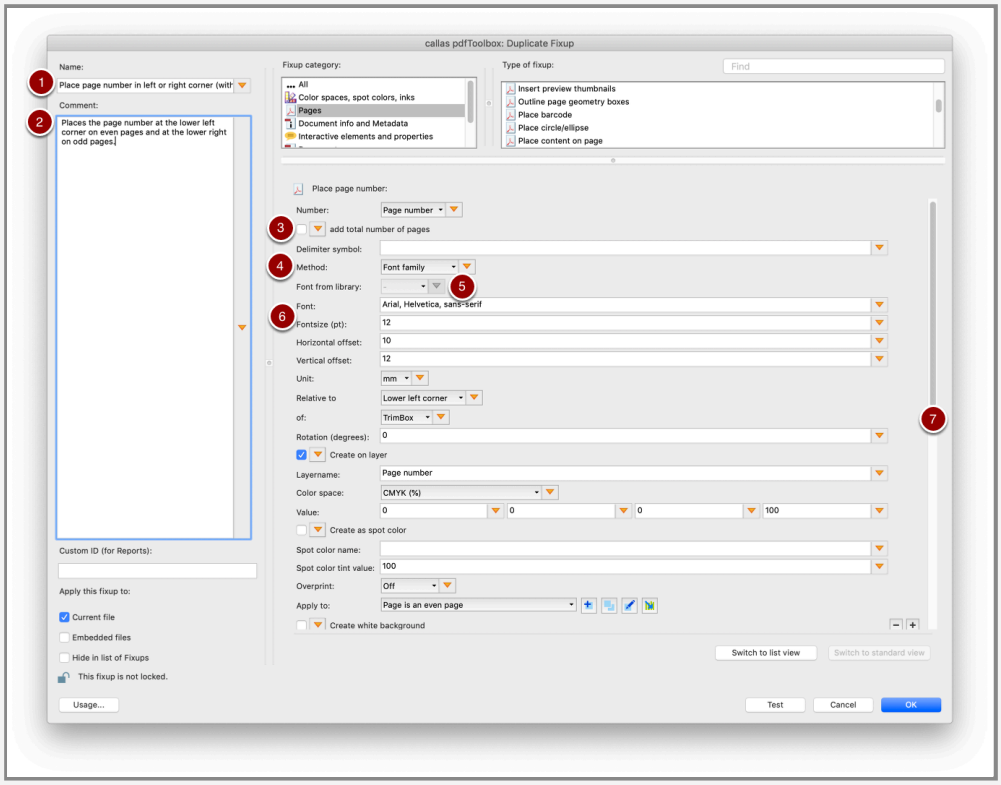


On each page, the page number is added. The page number is placed in the lower outside corner (right odd pages; left even pages). After "/", the total page count is also mention. The font is "Arial" in a 12 pt size.

In the next steps, you will learn how to determine the page numbers with other design specifications.

1. Close the processed PDF file.

Modify the Fixup



1. Duplicate the original Fixup and save the duplicated Fix-up as "Place page number in left or right corner (without total number of pages)"
2. Change the description of the Fixup in "Places the page number at the lower left corner on even pages and at the lower right on odd pages."
3. Toggle off "Add total number of pages" (even pages)
4. Method allows to add:
    1. Font families
    2. Web fonts
    3. Fonts from library
5. Upon selecting 'Fonts from library' method, fonts can be imported from the system
6. Change the font properties (even pages)
7. Scroll down and edit the same for odd pages

Apply the Fixup "Place page number in left or

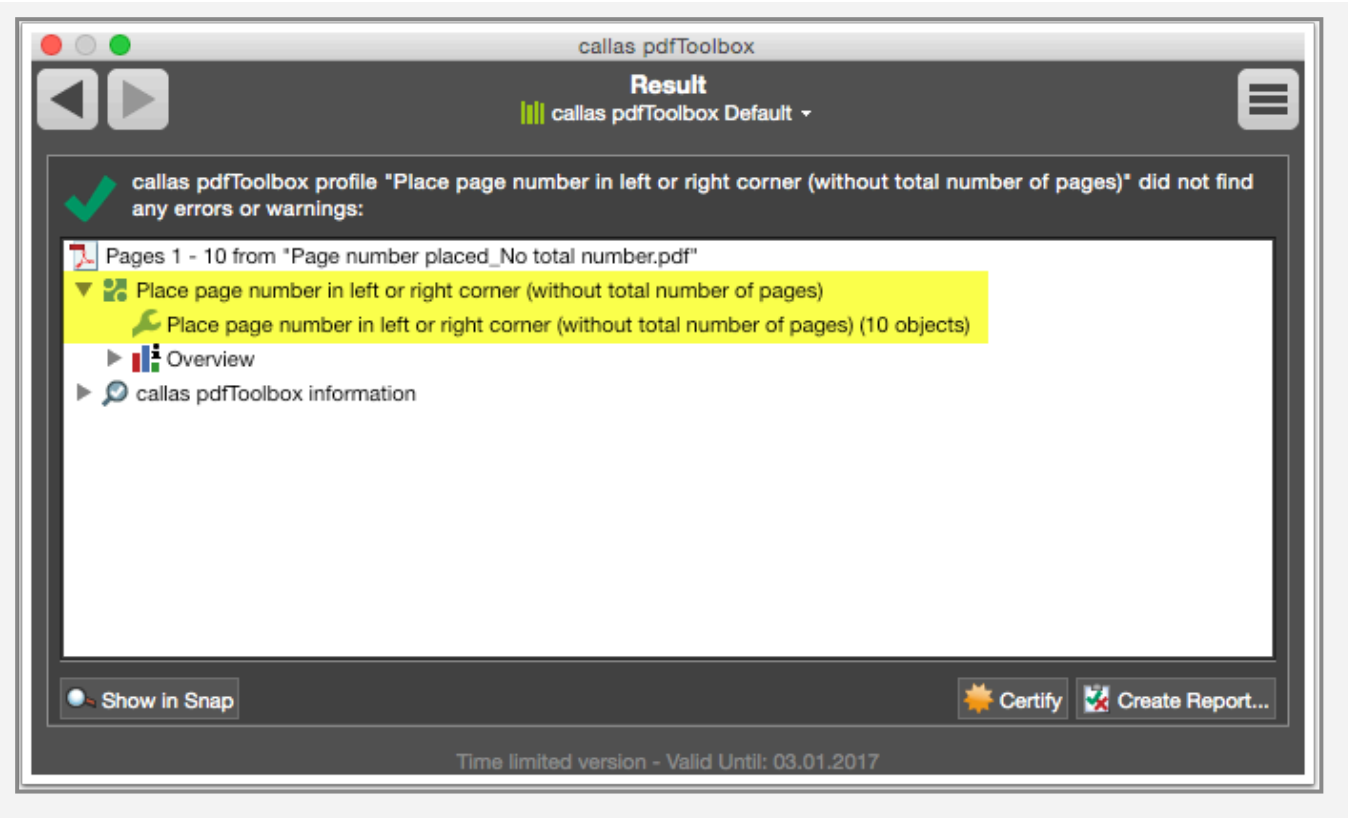
right corner (without total number of pages)"



1. Select the Fixup "Place page number in left or right corner (without total number of pages)".
2. Click "Fix".

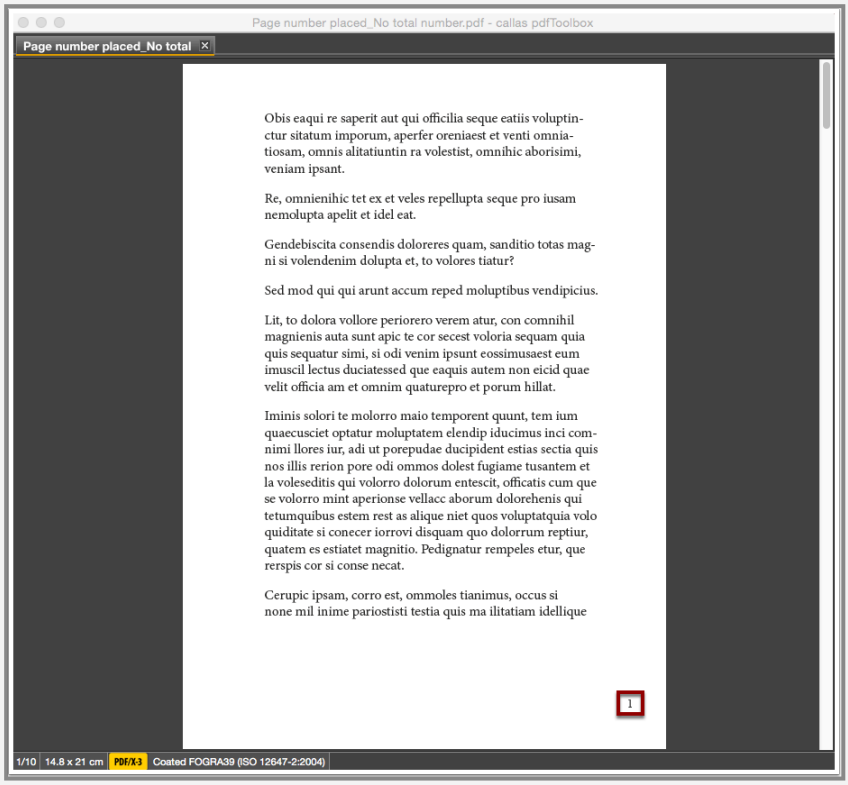
Save the output file.

Review the preflight report



A green checkmark shows, no errors occurred.

The processed PDF file



On each page, the page number is added. The page nummer is placed in the lower outside corner (right odd pages; left even pages). The font is changed to "Times" in a 10 pt size. The total page count is removed.

20.2 Place dynamic text

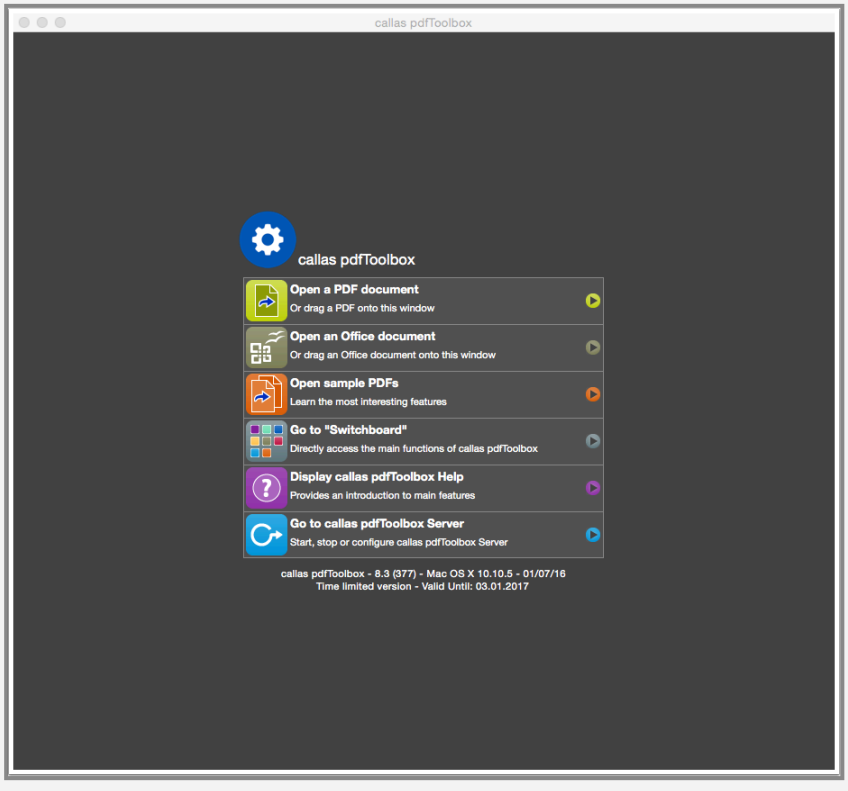
Since version 8, pdfToolbox supports a variety of information using static as well dynamic text in a PDF file. Different fonts, colors and placement options can be added.

Using the Fixup "Place specified text at the center of each page" you can add text in a PDF file. In this tutorial we add the watermark "Draft".

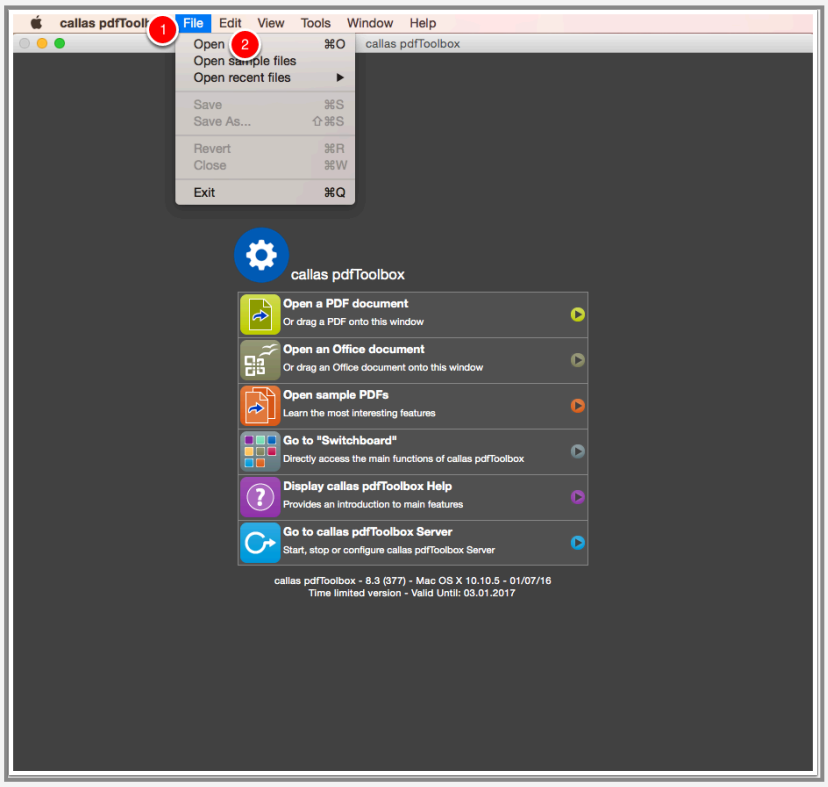
Draft_Demo_file.pdf

First steps (beginner)

Launch pdfToolbox Desktop

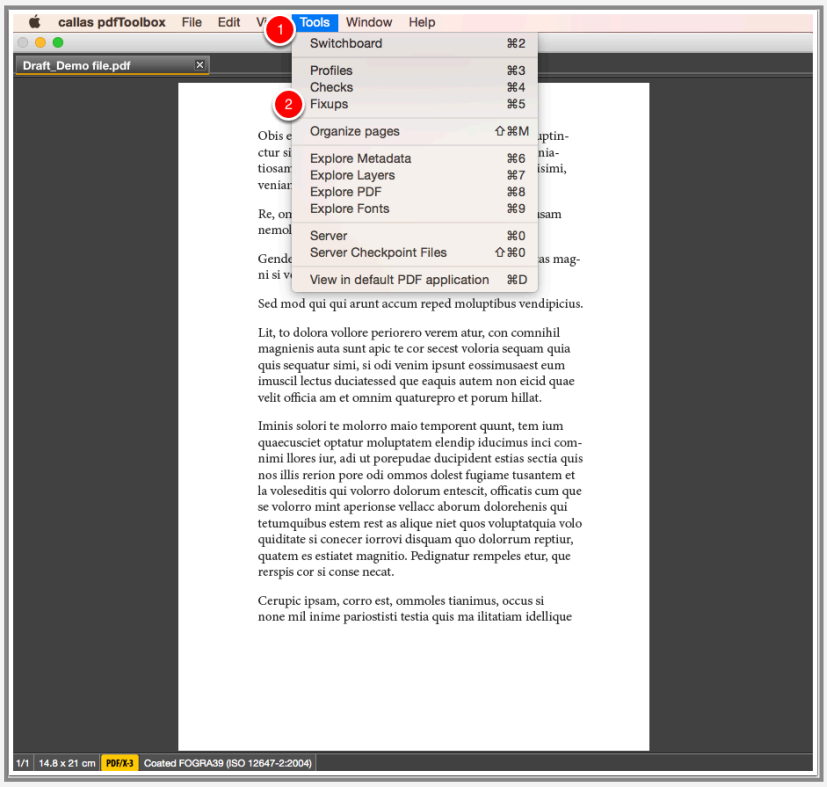


Open the PDF file "Draft_Demo file.pdf"



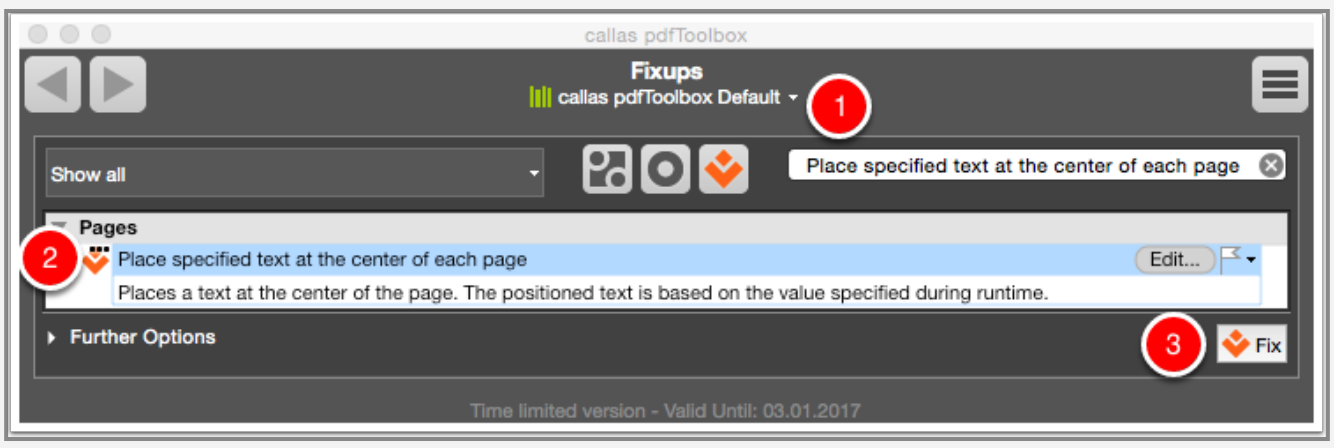
1. Go to "File".
2. Click "Open" to launch the file load dialog box and navigate to the folder where the input PDF file "Draft_Demo file" is located.

Open the Fixups dialog



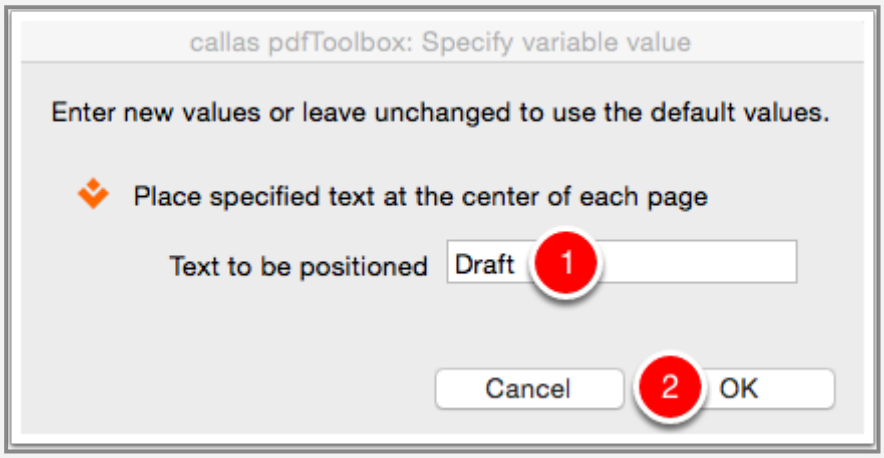
1. Go to "Tools".
2. Click "Fixups".

Apply the Fixup "Place specified text at the center of each page"



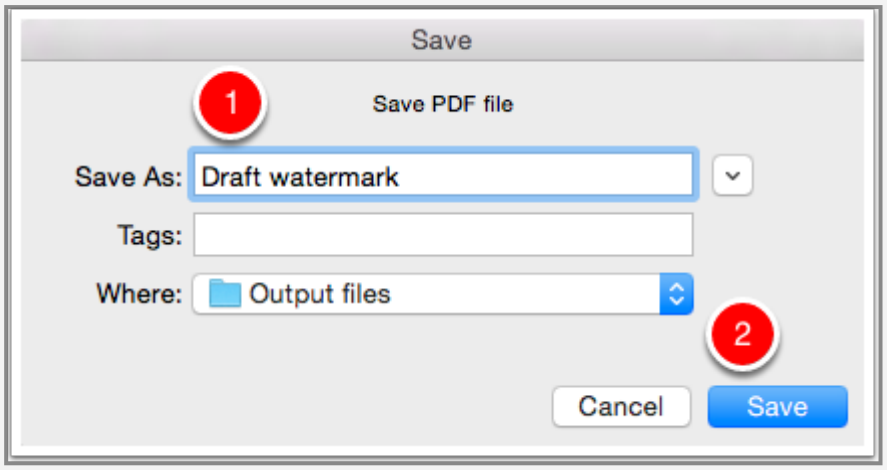
1. In the search field, look for "Place specified text at the center of each page".
2. Select the Fixup "Place specified text at the center of each page".
3. Click "Fix".

Enter new value



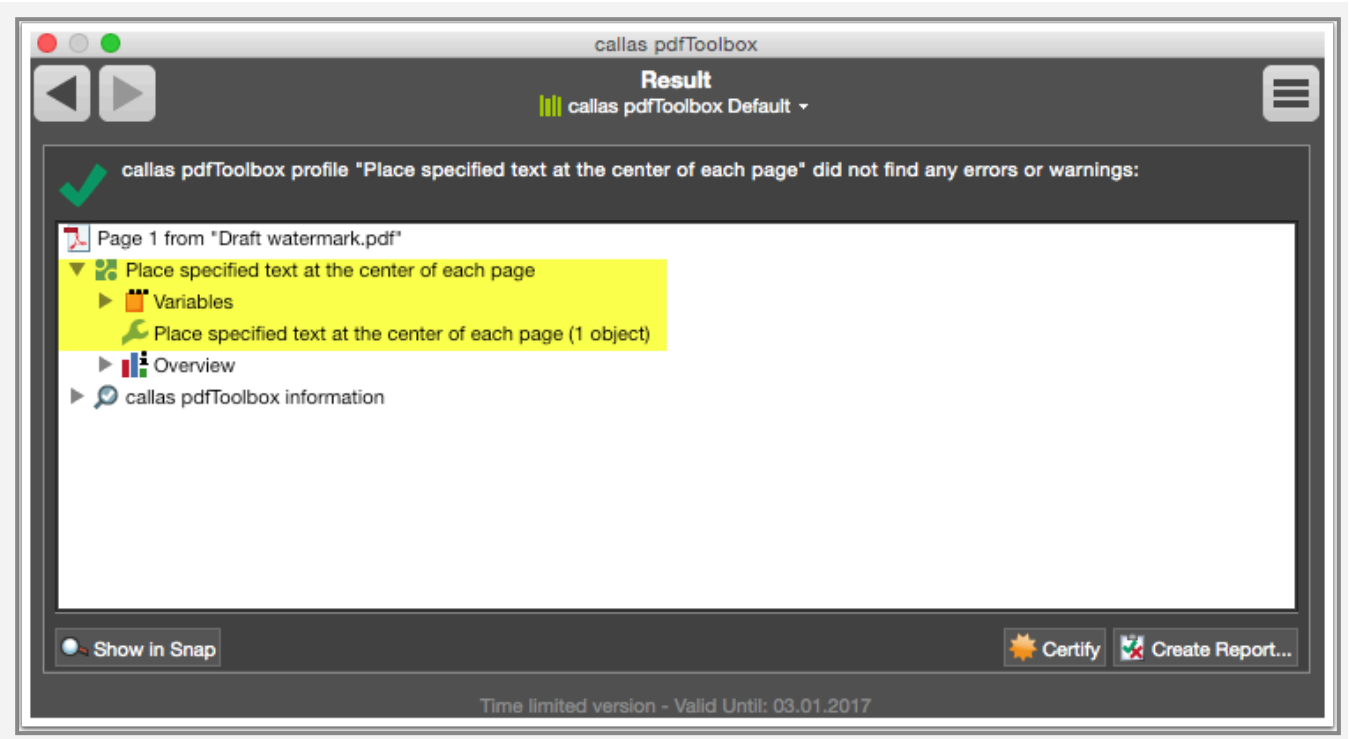
1. Use the default text value "Draft".
2. Click "OK".

Save the output PDF file



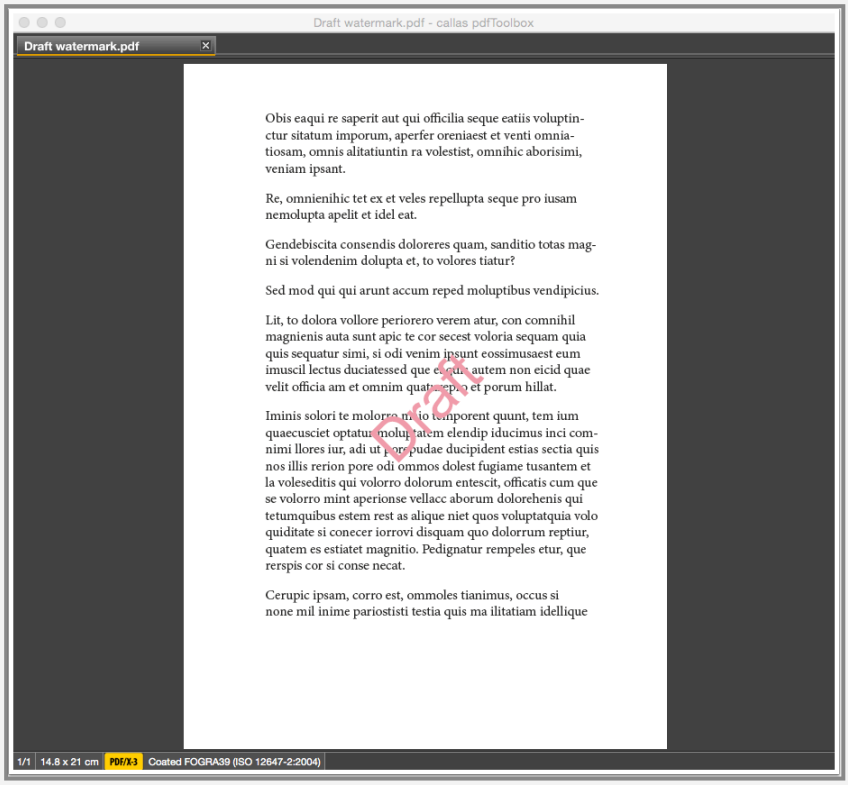
1. Save the output PDF file as "Draft watermark".
2. Click "Save".

Review the preflight report



A green checkmark shows, no errors occurred.

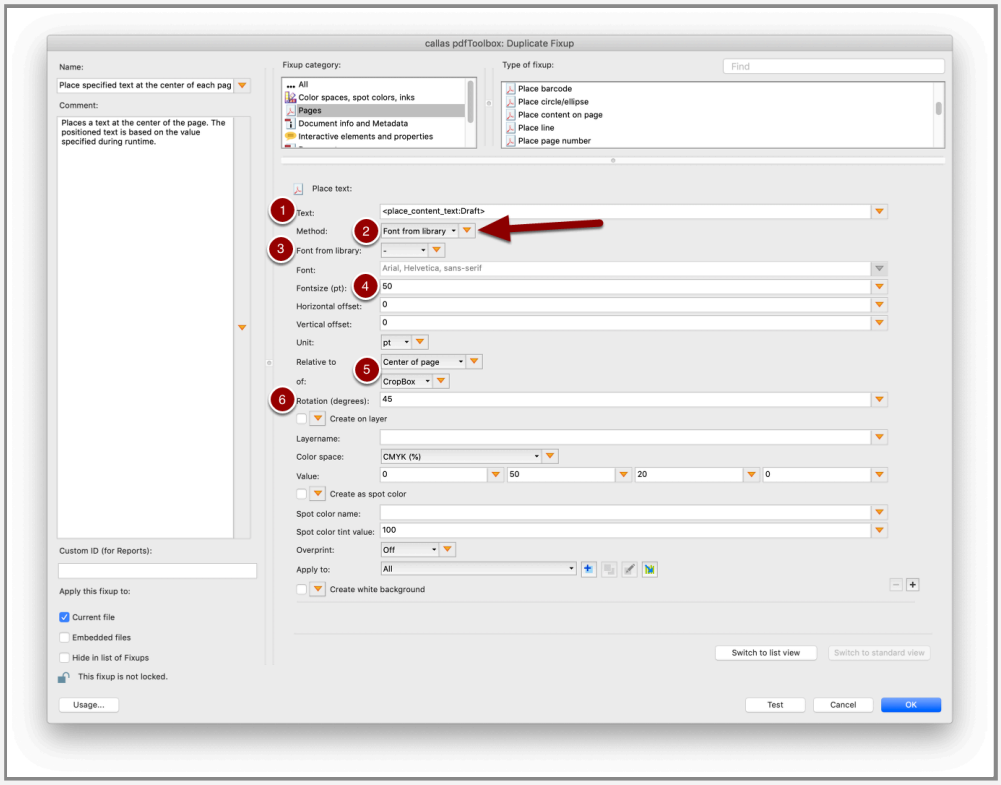
The output PDF file



The text "Draft' is placed in the center of the page.

NOTE: By duplicating the Fixup you can modify the font, size, color or other design specifications of the text.

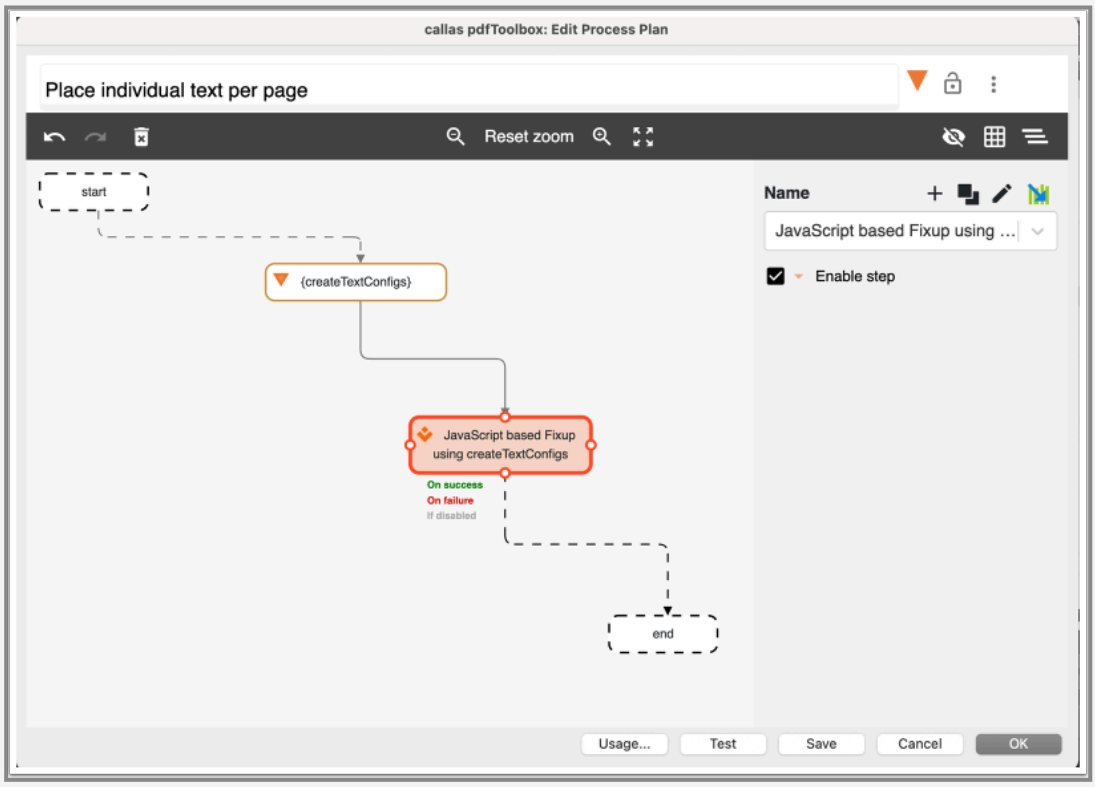
Modify the Fixup



1. The sample text that needs to be placed on the PDF
2. 'Method' allows to add:
   - Font families
   - Web fonts
   - Fonts from library
3. Upon selecting 'Fonts from library' method, fonts can be imported from the system
4. Font size (50 in the example above)
5. Position of the text to be placed
6. Rotation in degrees

20.3 Place individual text per page

If you want to place individual text on the pages in a PDF file, you may build your solution based on an adjustable Process Plan "Place individual text per page" that is introduced with pdfToolbox 13. You will find it in the Library "Shapes, Variables, JavaScript, Place content".



The Process Plan uses in the second step the Fixup "Arbitrary JavaScript controlled Fixups". This Fixup is configured based on JavaScript structures representing any other ("real") Fixup. The structure that is used here is the one of "Place text".

You may adjust the structure in the first step so that e.g. the text strings, their position and their color meet your expectations.

Step 1: createTextConfigs

There are two adjustable sections: The textArray with the text strings and the layout parameters for the texts.

textArray

```
const textArray = [
    "text on page one",
    "text on page two",
    "text on page three",
    "text on page four"
];

```

The first element in this array will be placed on the first page and so on. With a 4 element textArray the fifth page will again use the same text that was used on the first page. It should be easy to adjust the textArray according to your requirements.

Layout parameters

```
function buildConfigs() {
    let retArr = [];

    let textArrayIndex = 0;
    for (let i = 0; i < app.doc.numPages; i++, textArrayIndex++) {
        if (textArray.length === textArrayIndex) {
            textArrayIndex = 0;
        }

        retArr.push({
            "params": {
                "apply_to": applyToPage(i + 1),
                "color_space": "cmyk_percent",
                "color_value_1": 40.0,
                "color_value_2": 0.0,
                "color_value_3": 100.0,
                "color_value_4": 0.0,
                "create_as_spot_color": false,
                "create_on_layer": true,
                "create_white_background": false,
                "font": "",
                "font_file": "NotoSansSC-Medium.otf",
                // use "font_family" in "font_method" to specify system fonts in "font"
                "font_method": "font_file",
                "fontsize_pt": 24.0,
                "horizontal_offset": 0.0,
                "layername": "Text",
                "of": "cropbox",
                "overprint": "off",
                // other possible values for relative to are: "lower_left_corner", "left",
                // "upper_left_corner", "top", "upper_right_corner", "right"
                // "lower_right_corner", "bottom", "center_of_page"
                "relative_to": "center_of_page",
                "rotation_degrees": 0.0,
                "spot_color_name": "",
                "spot_color_tint_value": 100,
                "text": textArray[textArrayIndex],
                "unit": "pt",
                "vertical_offset": 0.0
            }
        });
    }
    return retArr;
}
```

Right below the textArray there is the function "buildConfigs" which specifies configuration sets for each page of the PDF file. In the core of this function is the object params. For each page one instance of this object is put onto the list of configurations for Place text so that there will be as many such configurations as there are pages in the PDF file.

You may edit all values in params except:

- "apply_to" which specifies the page for the current configuration
- "text" which takes one element out of the textArray

All other parameters should be self explanatory, you will find additional information in the comments of the step.

Step 2: JavaScript based Fixup using createTextConfigs

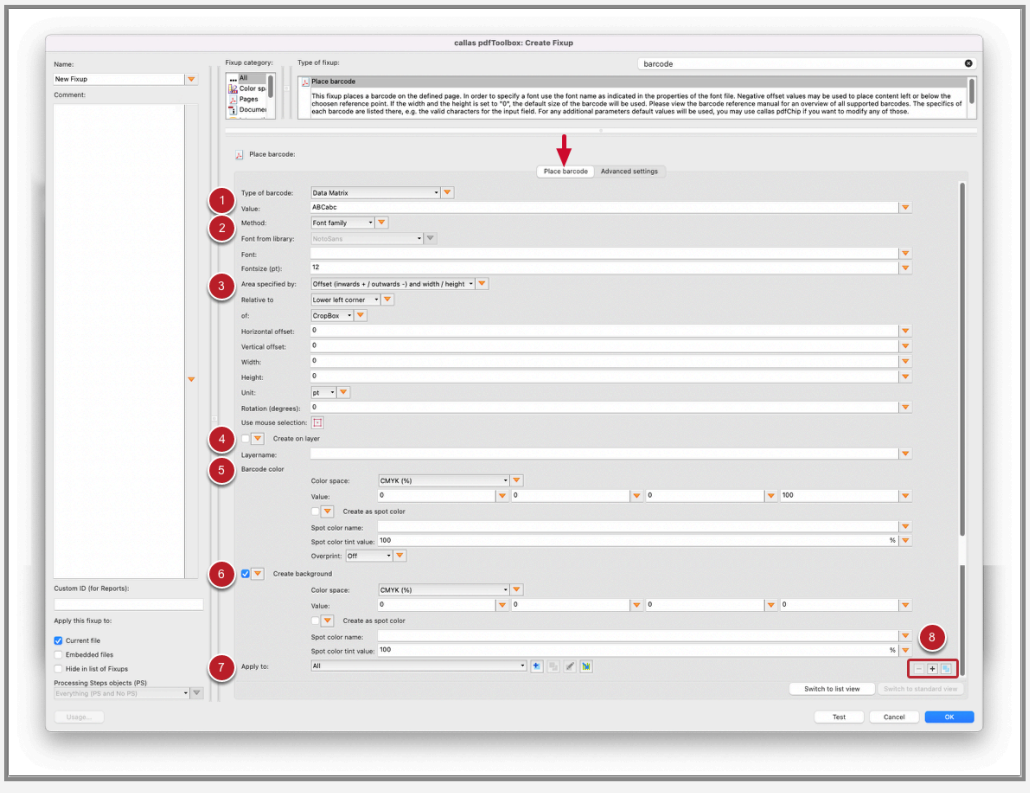
This step only applies what has been set up before.

20.4 Place barcodes and matrix codes

pdfToolbox supports the placement of both 1D barcodes (e.g., EAN-13) and 2D codes (e.g., QR codes, Data Matrix codes) on PDF pages. With support for over 100 barcode types, it covers all commonly used codes in practice.

A complete list of supported barcodes and matrix codes can be found in this article: List of supported barcodes and matrix codes.

The "Place barcode" Fixup



1. Type and expected value: A list of supported barcodes and matrix codes, along with examples of the expected value for each code, can be found here)

2. Font settings: The "Method" option allows you to define fonts using: Font family, Web font or Font from library. When selecting "Fonts from Library", fonts can be imported from the system.
3. Position settings: Defines the barcodes position and dimensions. The mouse selection tool can be used to visually select an area on the PDF page.
4. Option to place the barcode on a new layer.
5. Color settings for the barcode.
6. Color settings for the barcode background.
7. A preflight check can be set as a filter to specify which pages the barcode should be applied to.
8. To create multiple barcodes within the Fixup, use the "+" button. Alternatively, the "Duplicate Setting" button can be used to copy an existing set of configured parameters.

Advanced settings for barcode and matrix codes

Earliest version with full support for specifying advanced settings for barcode and matrix codes is **pdfToolbox 16**

The "Place barcode" Fixup provides various options for barcode creation, such as setting the barcode and background colors, adjusting font settings, and more. However, additional advanced parameters allow further customization of the barcode appearance. These parameters can be used, for example, to add a rectangular matrix code or to remove the text from an EAN-13 code.

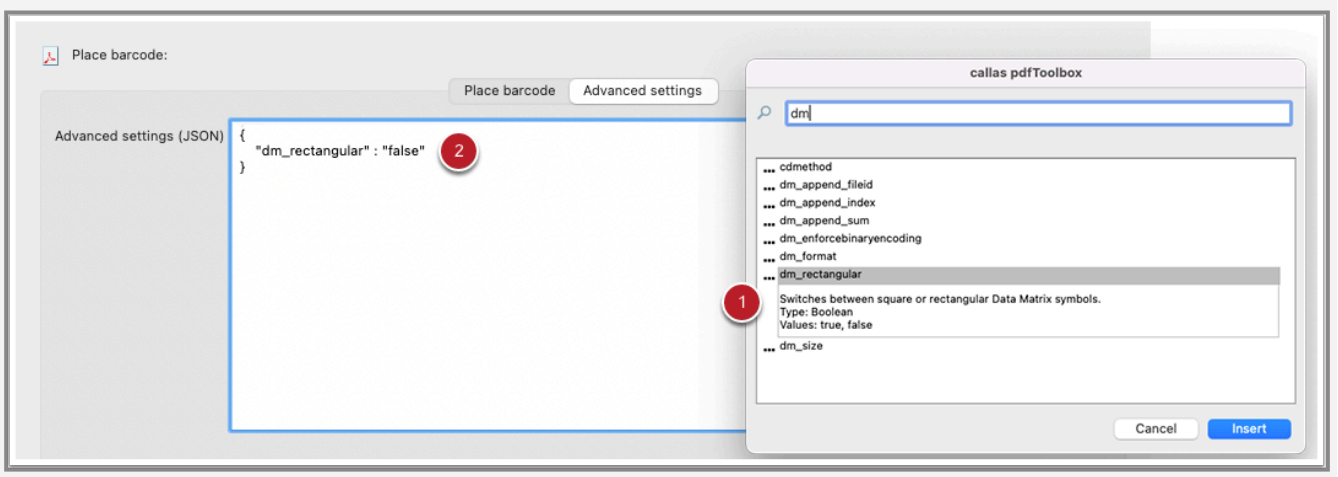
The "Advanced settings" tab provides a multiline input field for adding JSON-based key-value pairs to specify additional advanced parameters.

Note: Many of these parameters are symbology-specific, and a good understanding of how a given barcode or matrix code works is required in order to successfully use the parameters.

1. Clicking the info button opens a dialog showing all supported parameters for the barcode object that can be added (a list of these parameters can also be found in this article: Extended list of parameters for the barcode object)



1. Each parameter includes a short description, the expected data type, and a list of possible values.
2. After inserting a parameter from the list, its key and default value are automatically inserted, allowing for easy customization.



Multiple advanced parameters can be defined for a barcode.

Note: Advanced settings are always applied after the barcode is generated. This means e.g. that if a 90° rotation is set in the "*Place Barcode*" tab and a "rotation" key ("rotation" : "90") is also specified in the "*Advanced settings*", the barcode will be placed on the PDF with a total rotation of 180°.

Note: For each barcode defined in the *"Place Barcode"* section of the Fixup, a separate advanced setting must be created.



Predefined Fixup to place EAN 13 barcode

pdfToolbox already comes with a predefined Fixup that allows you to place an EAN-13 barcode with a custom value on a PDF page.



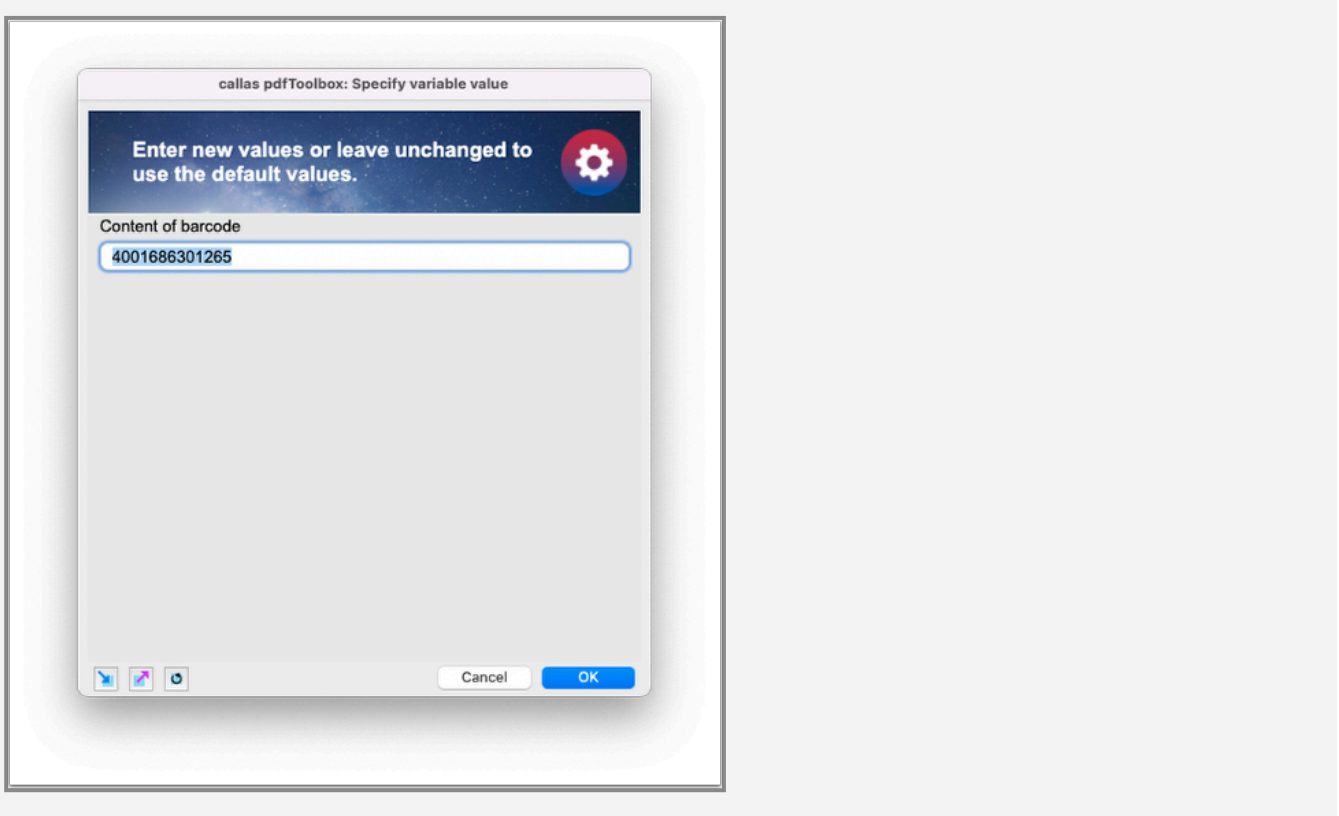
1. Go to the predefined pdfToolbox Library "Shapes, Variables, JavaScript, Place content".
2. Go to the "Fixups" section.

3. Enter "EAN" in the search box to restrict the list of Fixups shown to those containing the term "EAN".
4. Select the Fixup named "Place EAN 13 barcode with specified value"
5. Click "Fix" to place the EAN 13 barcode.

Optional: specify the value of the EAN 13 barcode to be placed

A dialog window will appear where you can enter the value for the EAN 13 barcode that will be placed in the file. If no entry is given, the default value will be used.

Important: The value entered must consist of 12 or 13 digits. If 12 digits are entered, pdfToolbox will automatically calculate the required check digit. If 13 digits are entered, the 13th must be a valid check digit, otherwise the attempt to place the barcode will return an error.





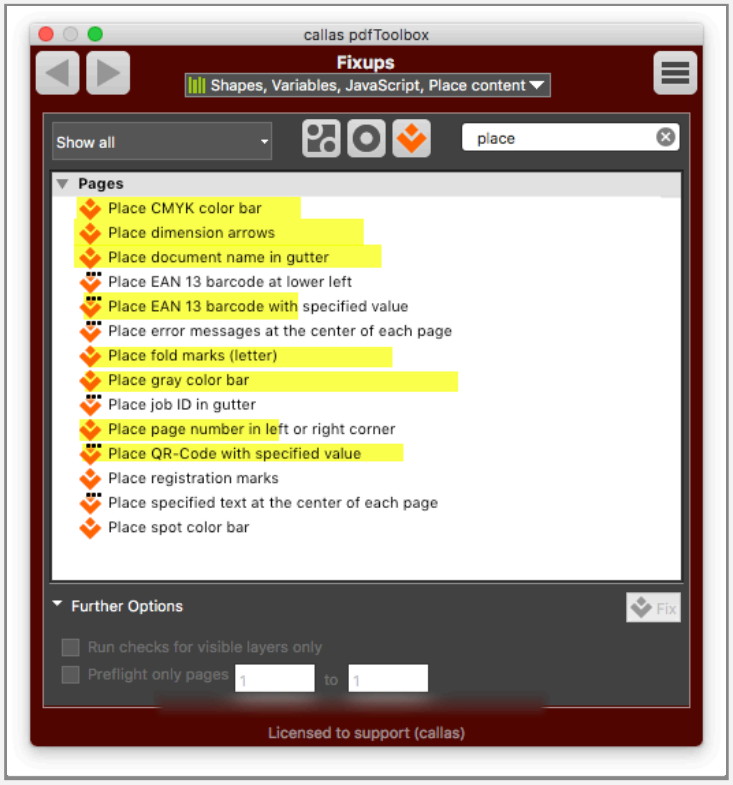
In order to adapt the Fixup to your own requirements, it is advisable to create a copy of the existing Fixup and make the required changes to this copy.

20.5 Place any content: Basics (“Place Content” via HTML templates

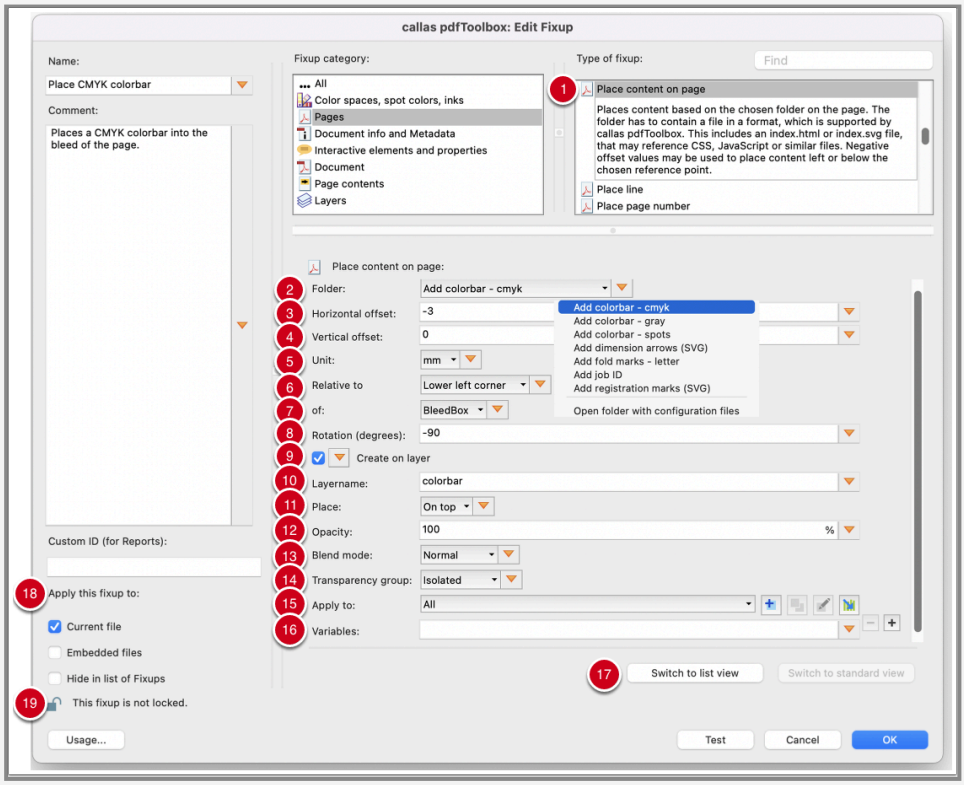
The Place Content Fixup (Fixup type: “Place content on page”) allows you to place PDF content. It can make use of standard or user-defined HTML templates. To achieve this, the pdfToolbox uses callas pdfChip technology. This latter software is designed to convert HTML files into high-quality PDF documents. In short, the following formula applies: HTML + CSS + JavaScript = PDF.

An overview of the release versions of pdfChip in pdfToolbox can be found here.

Settings options for Fixups of type “Place content on page”



The Fixups supplied with the pdfToolbox based on the “Place content on page” template can be found in the **“Shapes, Variables, JavaScript, Place content” library**. To see and/or edit settings options for this function, select one of the highlighted “Place” fixups (you can restrict the number displayed using a search term), then click the **Edit...** button.

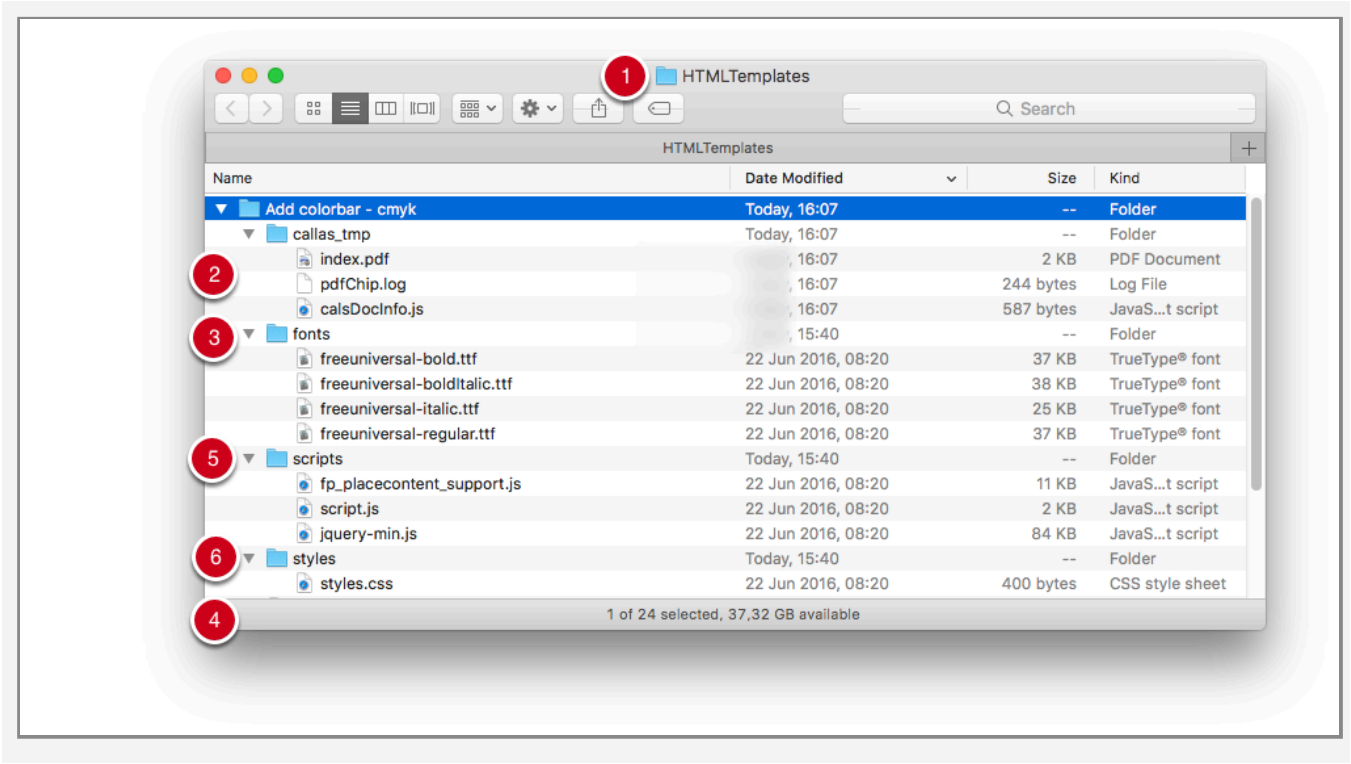


In the “Edit Fixup” window, you can see the following options.

1. The **Fixup type** in this case is “Place content on page”
2. Under **Folders**, you can select the folder containing all **HTML** files, **styles** and **scripts** required to use HTML templates. The folders shown in the image below are supplied as standard. Click on **Open folder with configuration files** to view the corresponding location within the file system. The **orange triangle** allows you to use **variables** here and for all other relevant settings.
3. In this field you can specify a **horizontal offset** (relative to the point specified under **6.**) Negative values move the element to be placed downward.
4. This field is used to specify a **vertical offset** (relative to the point specified under **6.**) Negative values move the element to be placed to the left.
5. Under **Unit**, you can choose between **pt, mm** and **inches.**
6. The **Relative to** option lets you set the **reference point** or **reference area**: Bottom left corner, left side of page, upper left corner, top of page, right side of page, bottom right corner, bottom of page and center of page.

7. **Of** relates to page geometry spaces to be used as the starting point (CropBox, TrimBox, BleedBox, MediaBox, ArtBox.)
8. If you want to rotate the placed content, you can enter a figure under **Rotation (degrees)**.
9. Activate the **“Create on layer”** checkbox to place the content on a new, separate layer.
10. You can also enter the **layer name** here.
11. The **Place** option lets you specify whether the content should be placed in the foreground or the background.
12. In this field, you can set the **Opacity** in percentage so that the content is placed transparent on the page.
13. You can select a **Blend mode** for the transparency.
14. If the content to be placed should be transparent on the PDF page, an isolated or non-isolated **Transparency group** must be defined. Read more about this in the article: Place content transparently or opaquely.
15. **Apply to** relates to page areas such as **“All”, “Page is an even page,” “Page is an odd page,”** and many others.
16. You can specify scripts and variables under **Variables**.
17. The **List view** provides an overview, even when the available settings are very extensive.
18. **Apply this fixup to:** This can be limited to **“Current file”** and/or **“Embedded files”**.
19. This setting lets you **lock** the Fixup to protect it against changes.
20. The **Name** of the fixup should reflect its intended purpose.
21. The **Comment** field allows you to describe the Fixup in greater detail.

Folder and file structure for HTML, styles and scripts



For the use of HTML templates to place content, in this example we see **folders and files** for the **fonts, scripts** and **styles** used *(each of which is optional)*, as well as the obligatory **index.html**. The content in the **Temp files** folder is updated every run:

1. The **HTMLTemplates** folder will be created by the pdfToolbox in the folder for the callas pdfToolbox version currently in use, under Repositories/Custom/Folder with a time stamp.
2. This will include - depending on the library - a number of predefined examples such as *“Add colorbar - cmyk”*. It will include the following folders and files:
3. The **fonts** folder, containing all fonts required for the HTML template,
4. The **index.html** file which is required in all cases,
5. The **scripts** folder containing JavaScript files,
6. The **styles** folder for CSS files.

Naturally, the folder names for “fonts”, “scripts” and “styles” can be changed depending on the template.

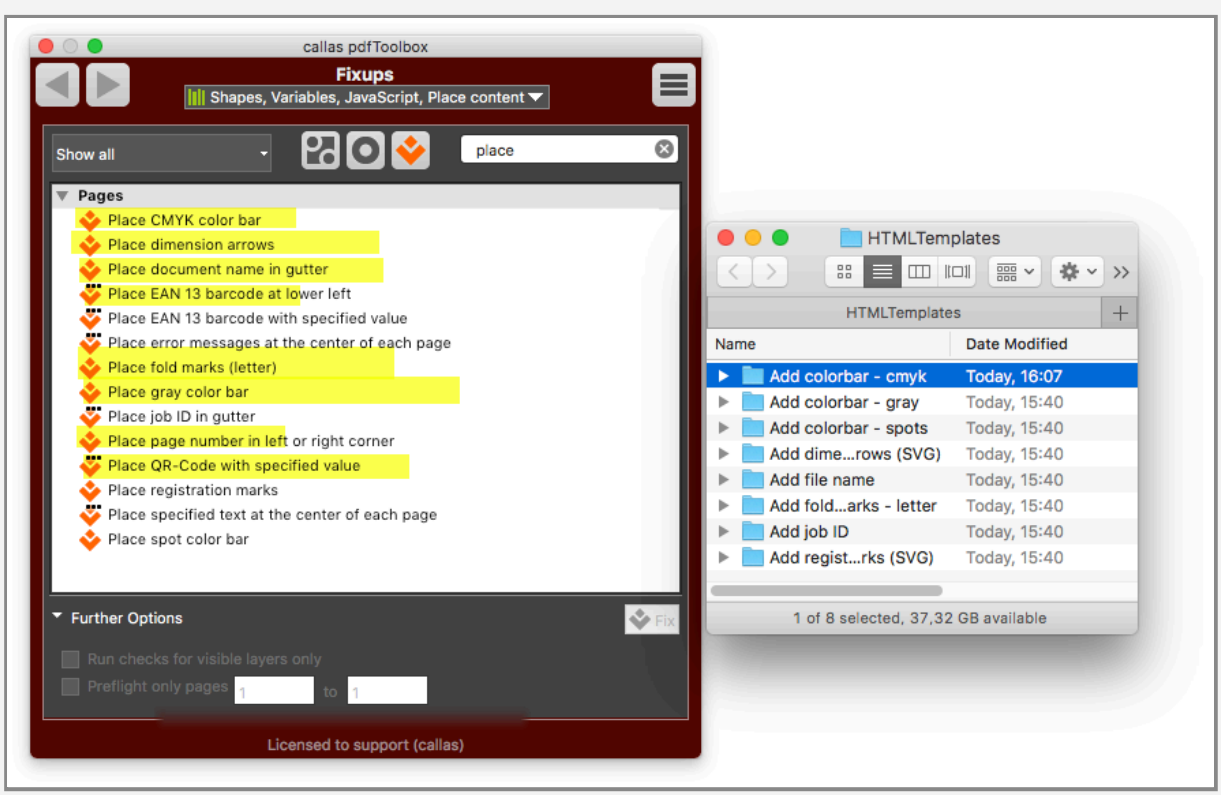
After the first run using a given HTML template, pdfToolbox will create the **callas_temp** folder.

This folder contains temporary files and will be recreated each time the Fixup is run.

The following files will be included here: **“calsDocInfo.js”** (contains information regarding the running of the “Place content” Fixup; see Use information about the PDF document); the **index.pdf** file (the PDF content generated using the HTML template); and **pdfChip.log** (a log file).

Important: *The user can create folders for their own Place Content Fixups using any name, such as an “images” folder for images. These folders must be referenced within index.html to allow pdfToolbox to access them.*

Supplied examples for “Place content on page” using HTML templates



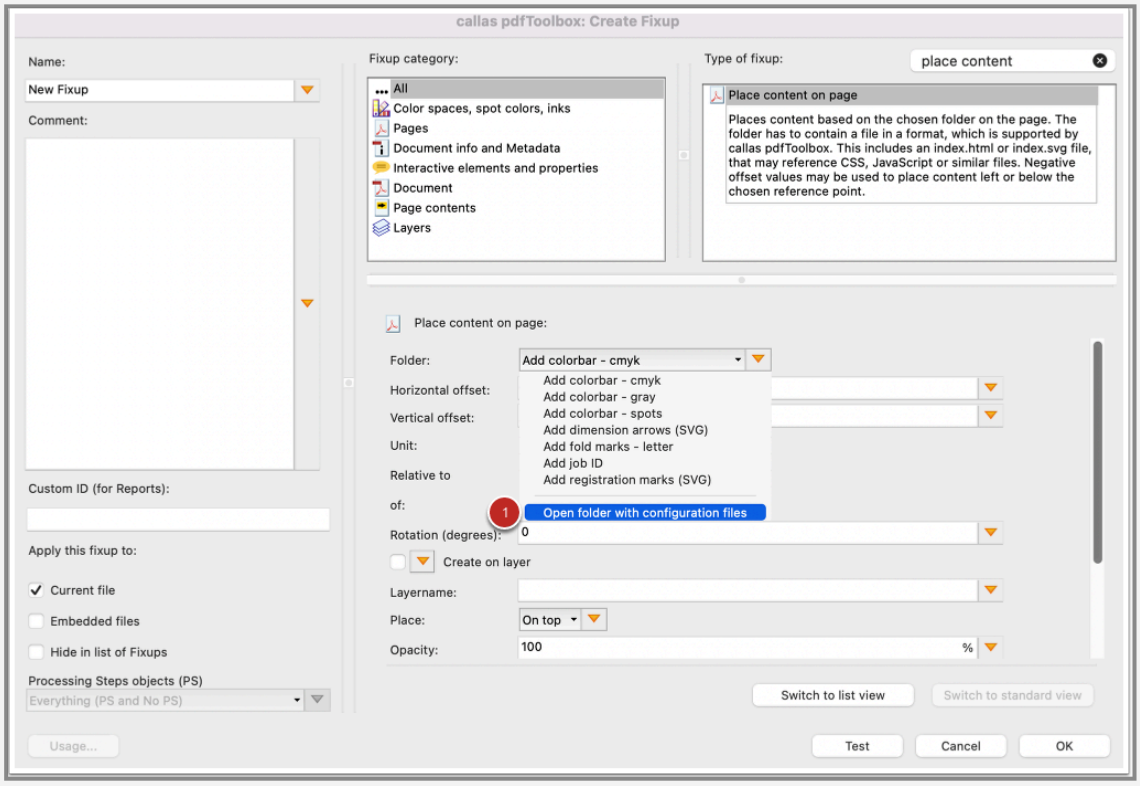
callas software supplies a number of “Place content on page”-type Fixups with the standard version of pdfToolbox which use HTML templates.

This includes three Fixups for adding **color bars** in CMYK, grayscale and spot colors, as well as Fixups for **dimension arrows** (GitHub link), **document names, fold marks (letter)** (GitHub link), **job ID** (GitHub link) and **registration marks** (GitHub link).

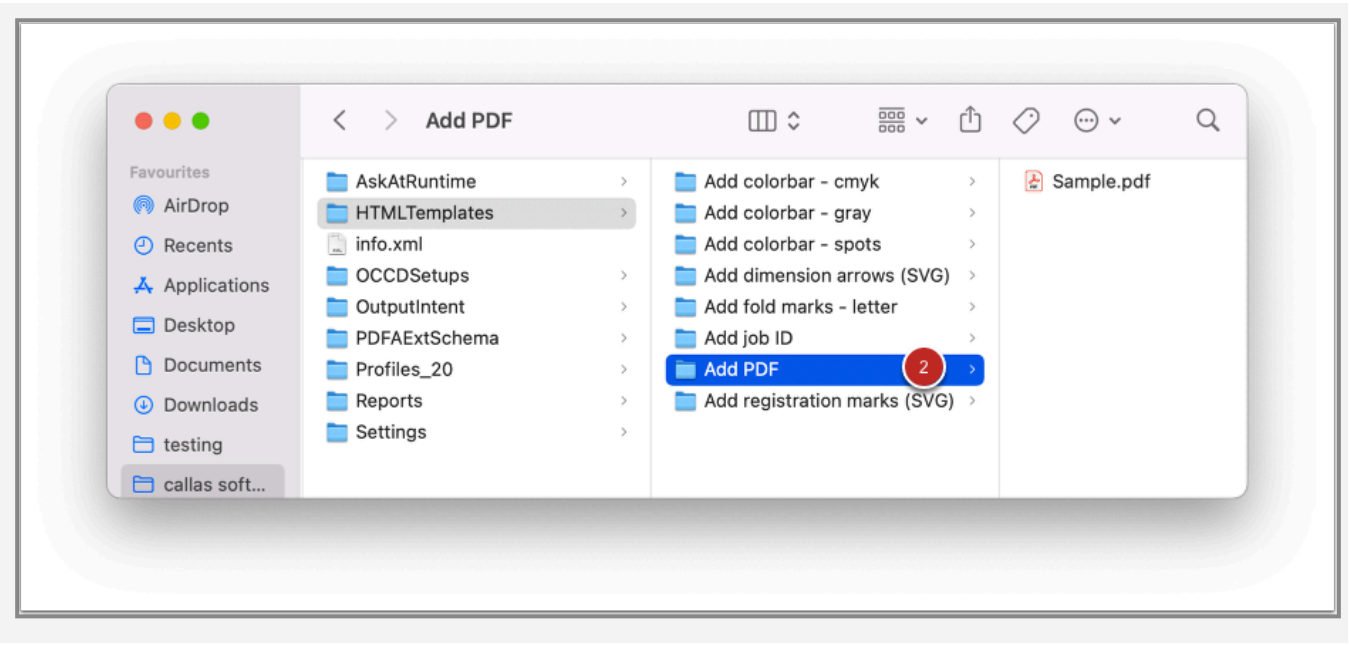
Important: *As you by now know that the "Place Content" fixup in callas pdfToolbox uses an HTML file and converts it to PDF. This gives you a lot of liberty in how to structure your HTML, in how you link to other files such as scripts, CSS and fonts and in how to structure the folder you use to hold that template. We have a template folder on GitHub where you find one possible way to do this. You'll find a liberal use of sub folders and CSS and Javascripts that are referred to, rather than included in the HTML file. You're free to follow this standard or implement your own.*

Add additional content: PDFs or Images on PDF

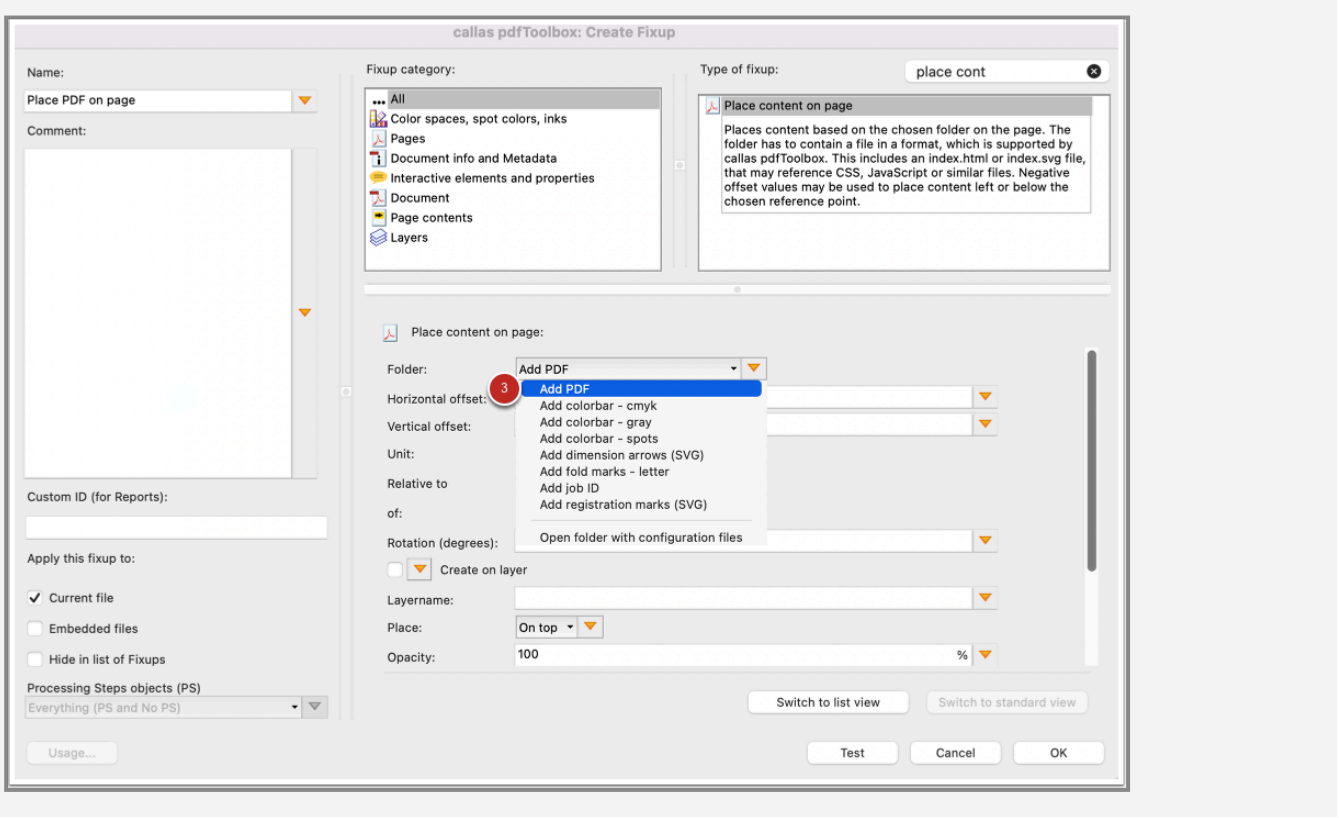
The "Place content on page" Fixup also gives you the possibility to place an image or a PDF file on the page - all without a full HTML template.



1. Under **Folders**, you can select the folder containing all **HTML** files, **styles** and **scripts** required to use HTML templates. The folders shown in the image below are supplied as standard. Click on **Open folder with configuration files** to view the corresponding location within the file system.



2. Add a new folder, for example "Add PDF". Then you can copy any PDF or image file to this folder.



3. After the Fixup is closed and reopened, the new "Add PDF" folder appears in the dropdown menu. Now the PDF or image can be placed on the page.

20.6 Place any content: Preparation

As a particularly widespread and powerful "page description language", HTML is ideal for flexibly specifying additional content for pages in a PDF file. The specific options provided by pdfToolbox (e.g. when defining CMYK colors) help to compensate for the language's limitations. When placing objects on PDF pages, however, you can free yourself of the basic principles which apply on the Web. For example, unlike HTML in a Web environment, tables and absolute positioning are not a problem for HTML for PDF.

There are also some default settings for HTML and especially for CSS which are not always intuitive or desirable for PDFs. This article therefore describes a number of preparatory tricks for setting up user-defined content-placing Fixups. These tricks generally also use ready-made Fixups (such as placing color bars).

The tips given here are based on the HTML templates, or more precisely on specific CSS conditions. You can use any text editor to create or edit the **styles.css** file.

"Sublime Text" (available for Mac OS, Windows and Linux) is particularly well suited to the pdfToolbox environment, as it works well when displaying folders and PDF files.

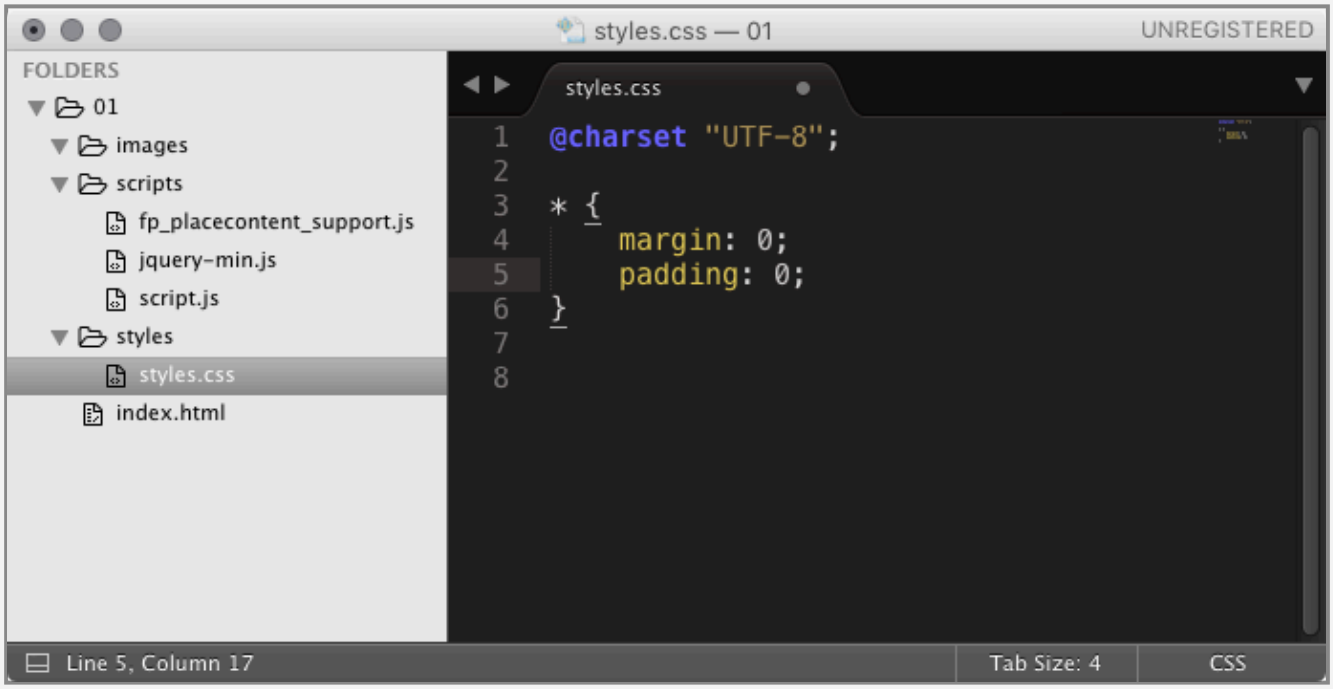


Adding helpful code-based instructions to

styles.css with a text editor

Customizing default margins and padding

All of the following optimizations relate to element display as adapted to place content and can be found in one of the referenced CSS files such as **styles.css**.



Many HTML elements use pre-specified **margins** and **padding**. This results in e.g. element positioning always maintaining a given distance between elements. In order to avoid this outcome and to ensure predictable results, change any margin or padding values in the CSS from **"greater than 0"** to "0".

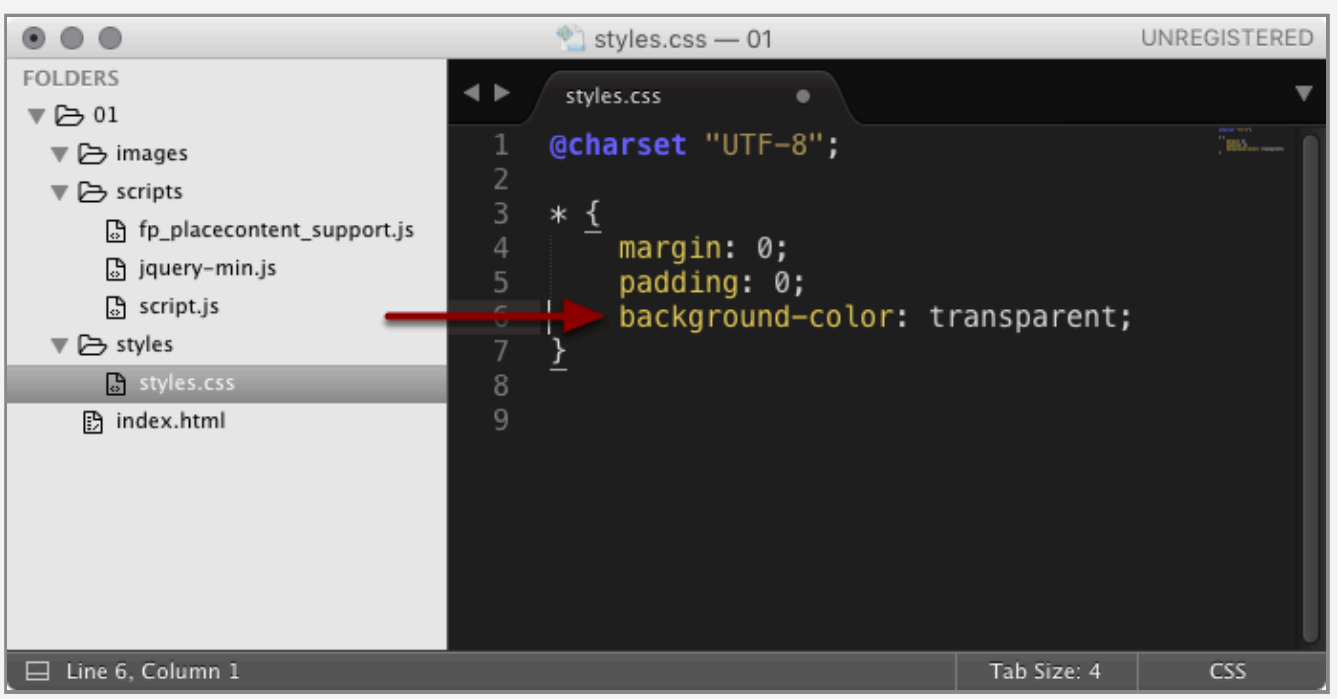
The following CSS code will set all margin and padding values to 0:

```
* {
margin: 0;
padding: 0;
}
```

(is the universal CSS selector)*

Set background color to transparent

The default background color in HTML is white, specifically **RGB-white**; this is not desirable when printing PDFs.



Such a background is generally undesirable when placing HTML objects. The code for this is as follows:

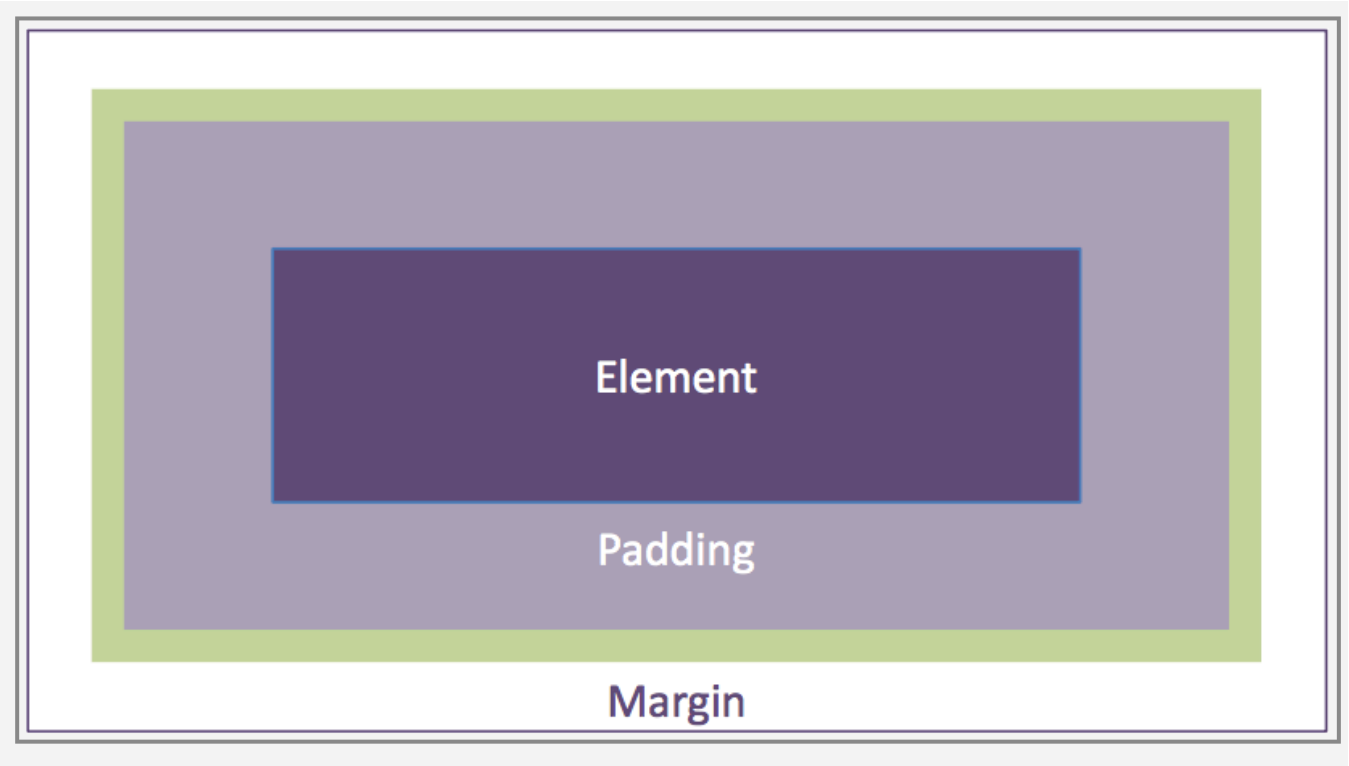
```
* {
background-color: transparent;
}
```

A theoretically identical but somewhat more precise option is to only set the **body** background to transparent:

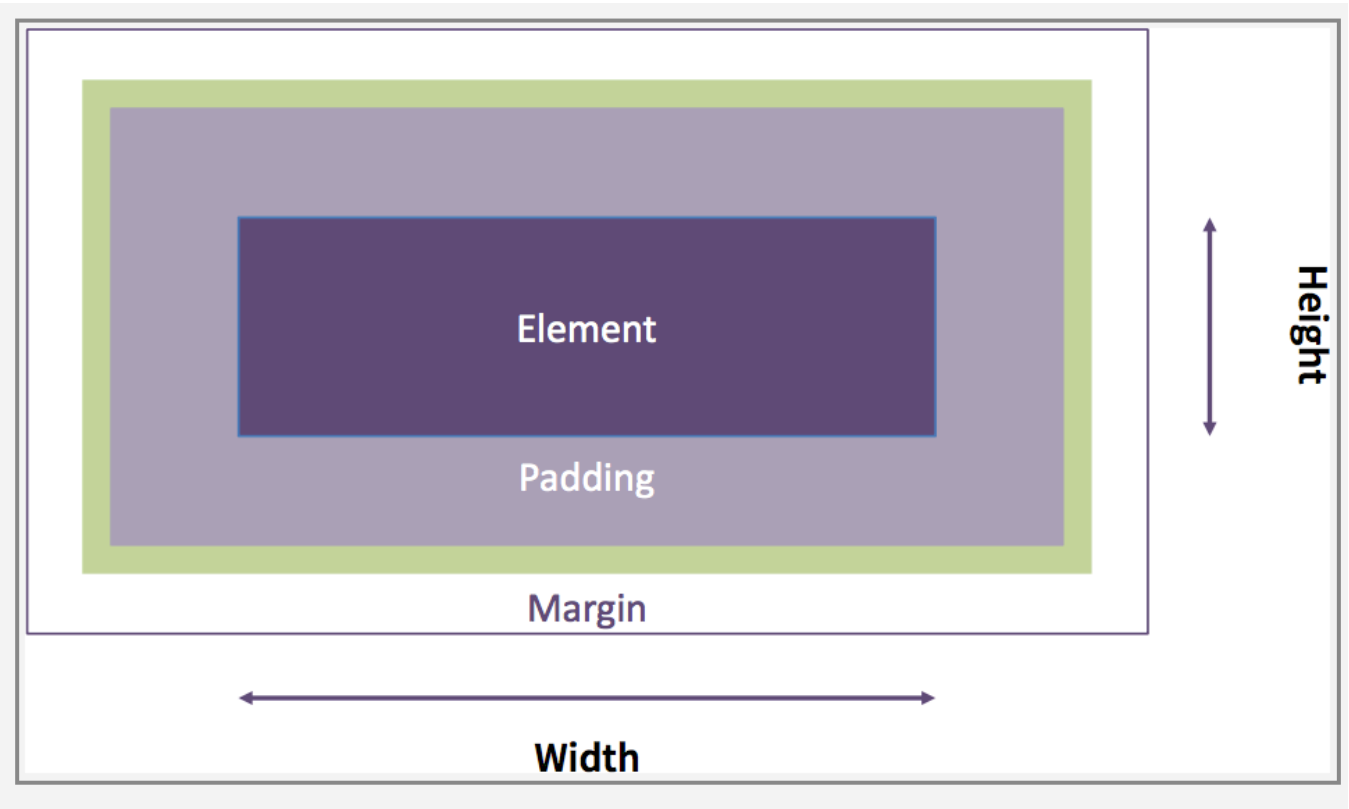
```
body {
     background-color: transparent;
}
```

As a rule, both options will lead to the same results.

The size of the box

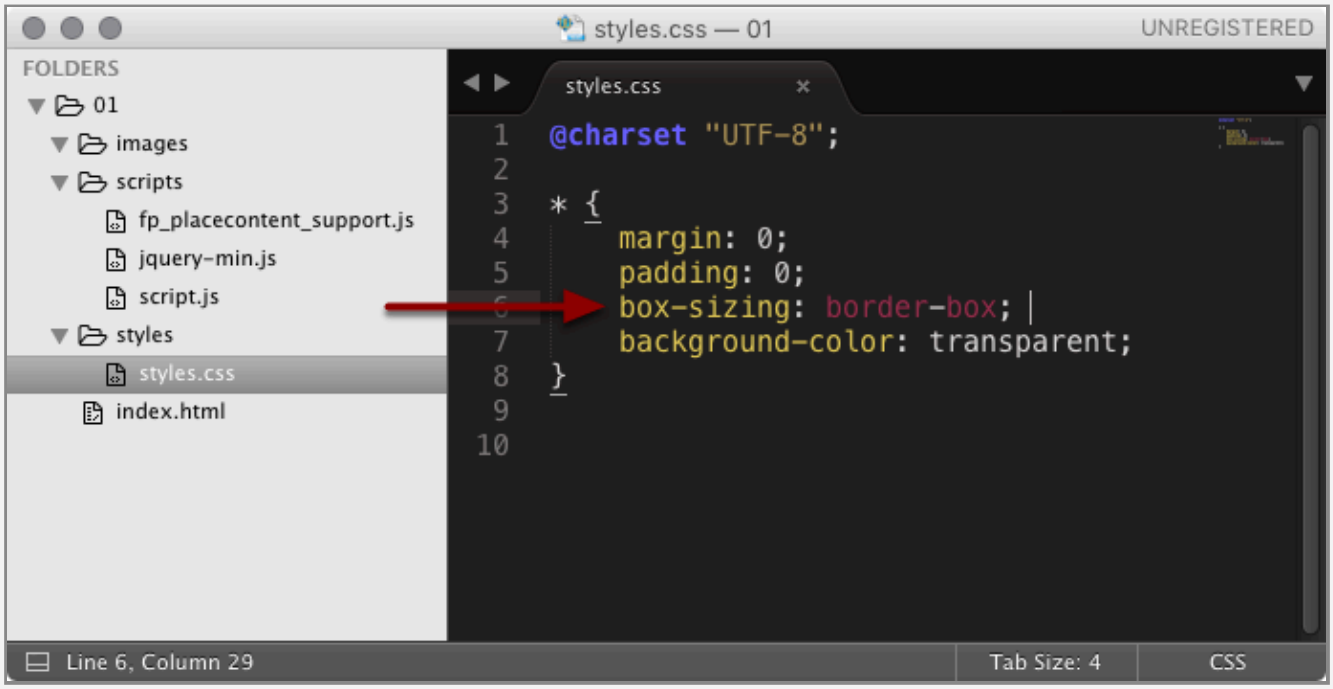


Each HTML element is a box which - as described above - is somewhat larger than the actual element due to margins and padding.



All **width** and **height** values given in the CSS relate only to the element itself; additional **margins** and **padding** are not taken into account. When working with print content, however, it is often desirable to specify these details for the entire element including padding, line strength and margins. This is naturally only necessary if these details for this element have not been set to 0 as described above.

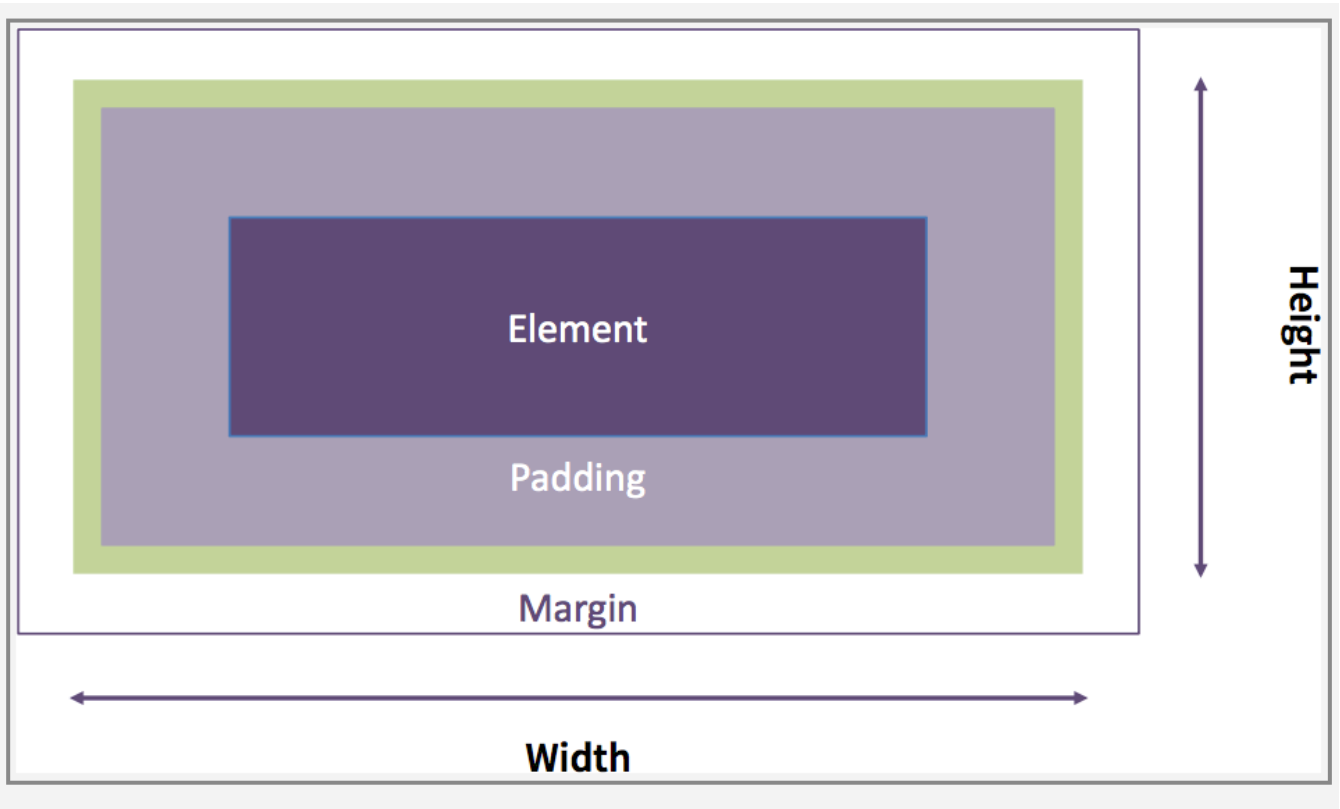
To avoid having to calculate the desired width and height values, you can add a suitable instruction to the CSS.



The CSS code is as follows:

```
* {
      box-sizing: border-box;
  }
```

This application will again apply to all elements on an HTML page.



As a result, the height and width values will be measured for **the entire border box** including margins and padding.

Example with ready-made HTML templates

The following zip file contains a number of relevant HTML templates that can be downloaded and used as reference/examples:

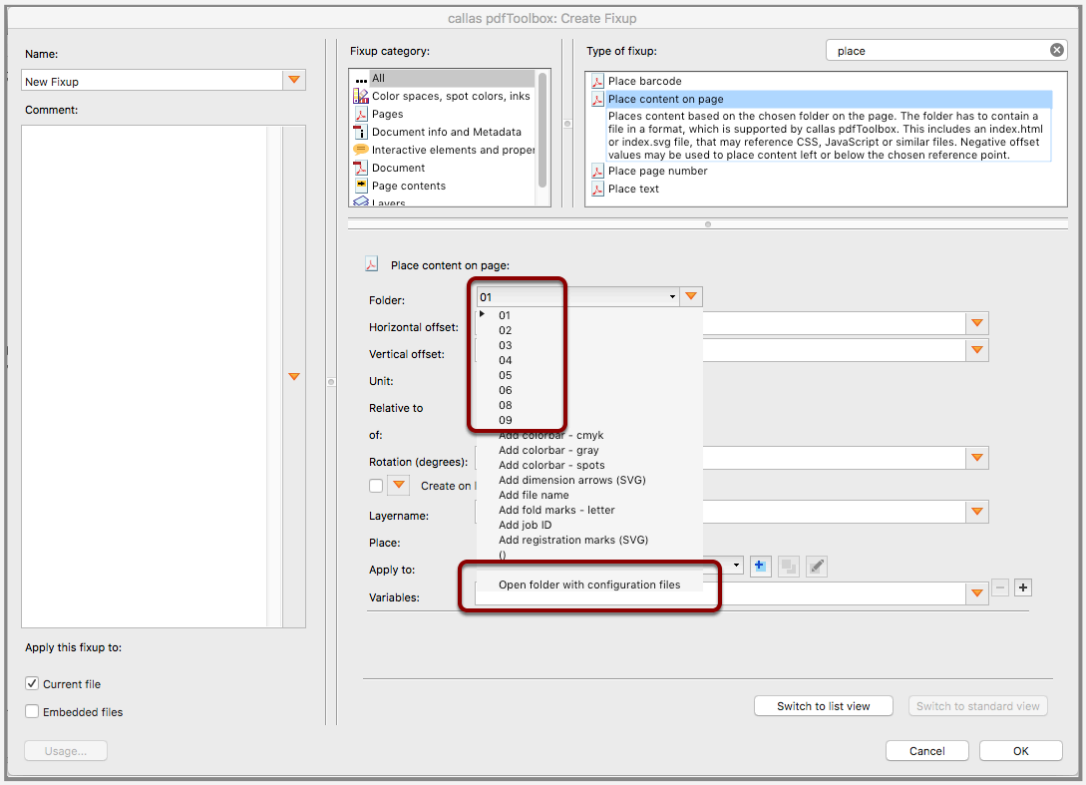
HTMLTemplates.zip

Allowing pdfToolbox to access the “HTMLTemplates” folder

Extract the downloaded ZIP file and **open pdfToolbox**.

Under **Fixups**, you can **create a new Fixup** from the flyout menu. Select the **“Place content on page”** Fixup. From the

"Folder" drop-down menu, you can select **"Open folder with configuration files"**.

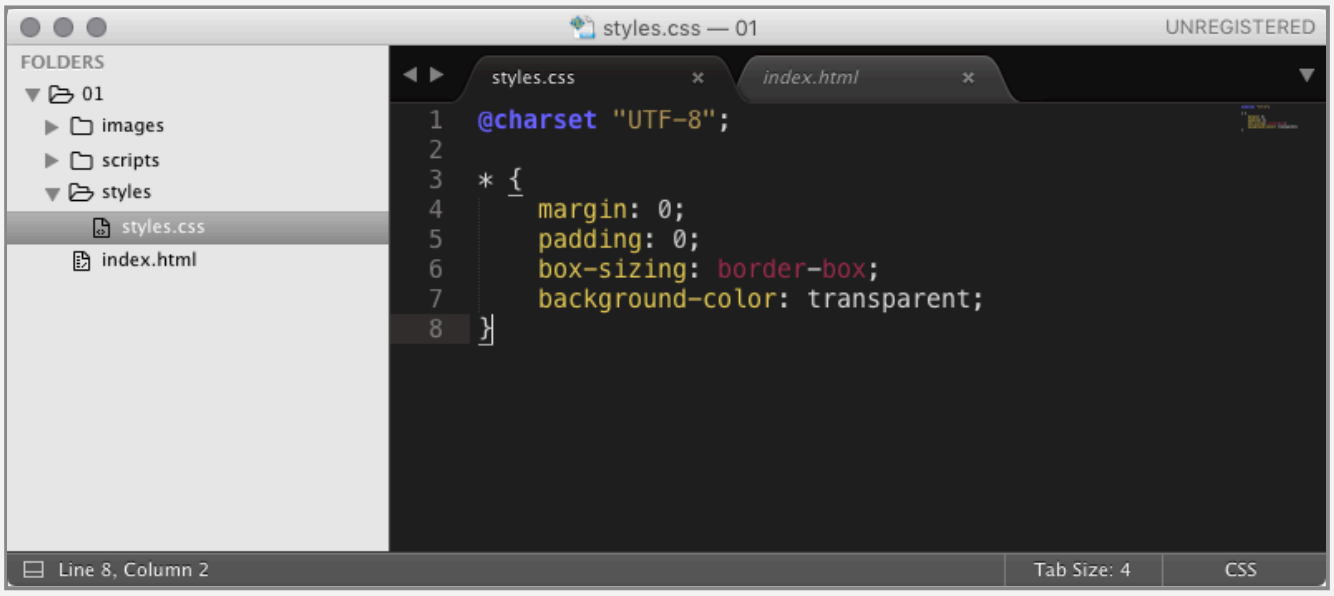


Move or copy folders **01 to 09** from the previously unpacked **"HTMLTemplates"** folder to this location. These folders contain ready-made CSS styles, scripts and much more.

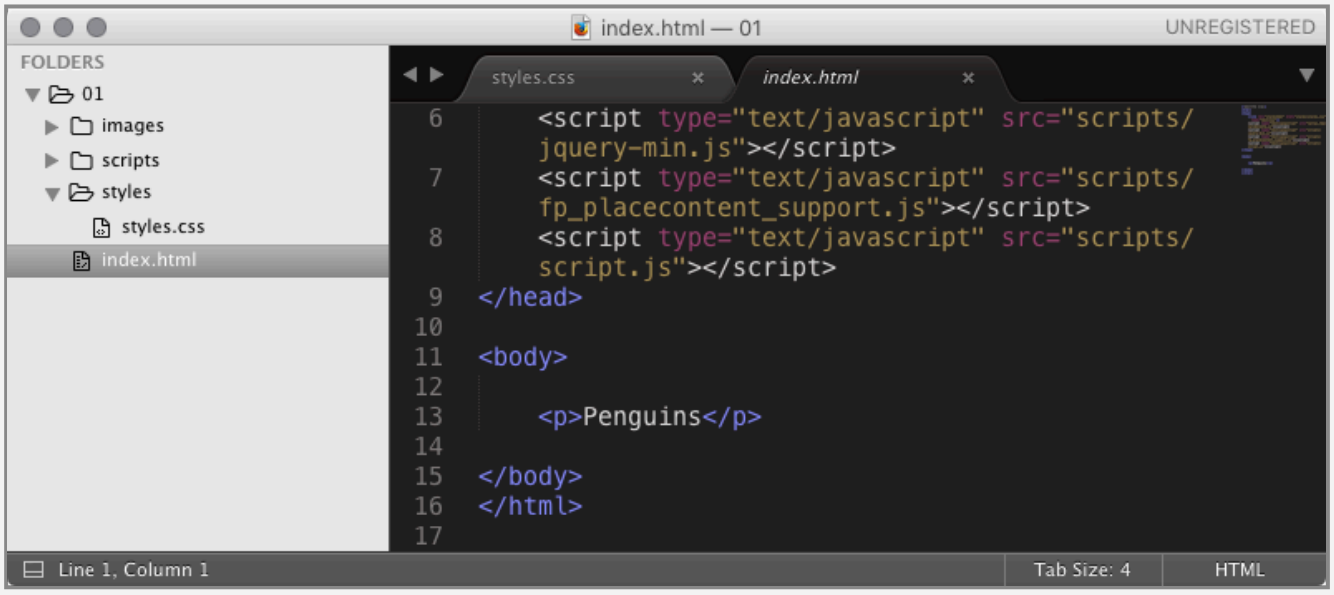
Files in "HTMLTemplates"

As mentioned in the introduction, Sublime Text is an ideal text editing application. To see an overview of the files and the folder structure, **drag the desired folder into the Sublime Text window**.

Selected files in Folder 01



styles.css contains all lines of code which may be helpful with preparation (padding and margins set to “0”; Box-Sizing = Border-Box to more easily calculate placements and a transparent background).

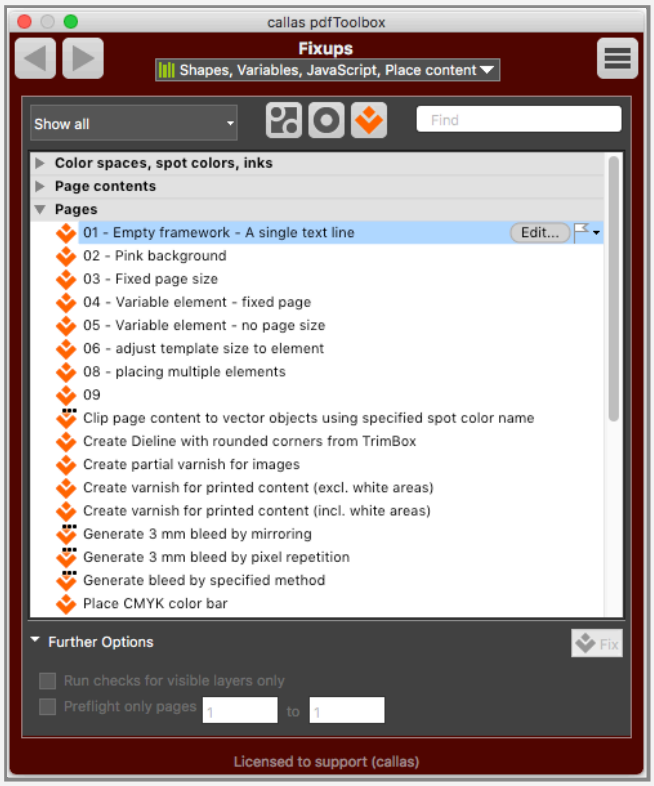


The most interesting part of **index.html** is the **body** section: The text “Penguins” is located here and will be placed within the current open PDF when the Fixup is run.

Create and run Fixup

- Create a **new Fixup** as usual.
- Select the **“Place content on page”** Fixup type.
- Under Folder, select **Folder 01**.
- **Save** the Fixup under a suitable name and click OK

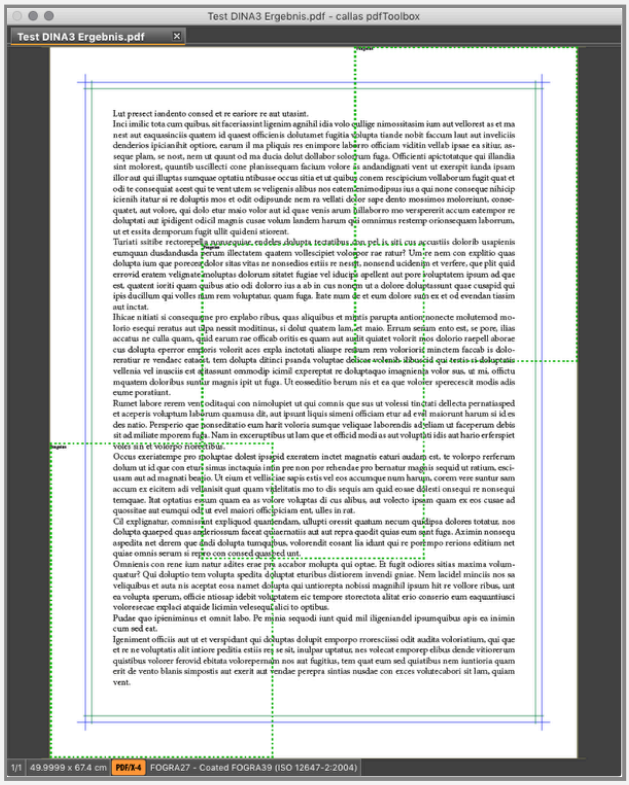
Tip: In order to clearly see that pdfToolbox places a PDF in DIN A4 format within the open PDF document by default, choose a PDF in a format larger than A4 for editing.



Select the new Fixup and **start** it.

Depending on the **placing parameters** you selected in the pdfToolbox dialog (bottom left, top right, middle, etc. as well as frames of reference like CropBox, TrimBox, etc.), the text **“Penguins”** (upper left within the internally provided DIN A4 page) will be added to the open PDF.





Placed in three locations: In this image, **“Penguins” has been added to an A3-size PDF document** (plus first cut and info

area) at the bottom left, in the middle and at the top right relative to the CropBox.

The **green area** showing the internally used A4 page has been added later to **clarify** the image.



Important: *As you by now know that the "Place Content" fixup in callas pdfToolbox uses an HTML file and converts it to PDF. This gives you a lot of liberty in how to structure your HTML, in how you link to other files such as scripts, CSS and fonts and in how to structure the folder you use to hold that template. We have a template folder on GitHub where you find one possible way to do this. You'll find a liberal use of sub folders and CSS and Javascripts that are referred to, rather than included in the HTML file. You're free to follow this standard or implement your own.*

20.7 Place any content: Positioning content

Using Fixup parameters or HTML size relative to PDF size to place objects

The positioning of HTML elements placed within a document can be influenced in one of two ways:

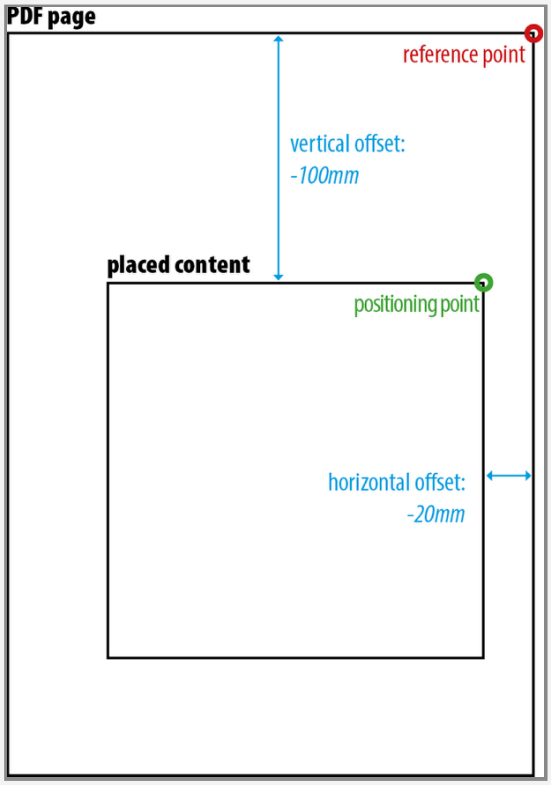
- By entering details in the “Edit Fixup” dialog
- Directly in the HTML template or in referenced CSS or JavaScript files

Positioning content using the reference point

PDF and HTML files actually use **coordinate systems** which differ based on the origin points of the X and Y axes.
Whereas PDF uses a coordinate system with the origin on the left, HTML uses a point at the upper left as the origin.
pdfToolbox compensates for this difference.

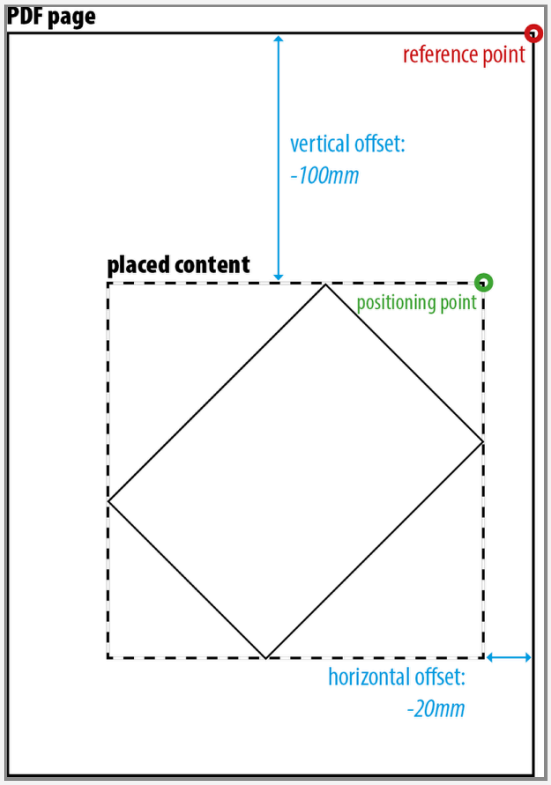
Important: The effective coordinate system for the PDF page as used in the Fixup is defined by setting the **reference point** to “relative to”. If you also select “Upper left corner” here, both coordinate systems will use the same origin. Positioning will generally only be “intuitive” if this is done.

In all other cases, it is important to be clear about the **basic positioning**: pdfToolbox first identifies the **positioning point** on the page using the details within the Fixup.
The **reference point** for the corresponding corner of the HTML object will be placed on this point.



Example: If the **reference point** is set to "relative to" the "upper right corner", the *horizontal offset to -20 mm* and the *vertical offset to -100 mm*, the result will be that the **positioning point** is 20 mm to the left of and 100 mm below the upper right corner of the PDF page. The upper right corner of the HTML object will then be placed on this point.

The issue becomes more complicated if we rotate the HTML object as part of the fixup by specifying **"Rotation (degrees)"**. In this case, the HTML object is rotated before placing it. A virtual rectangle with edges parallel to the coordinate system will then be created. For all rotations other than 90°, 180° or 270° (or their negative counterparts), this virtual rectangle will then be larger than the unrotated object itself. This virtual rectangle will be placed based on the rules described above.



In other words, when the HTML object is placed, the source in the HTML coordinate system is not used. Rather, the corner or edge corresponding to the reference point described in the Fixup is used.
The size of the HTML object therefore plays a particularly decisive role when positioning content.

Internal page size: Default DIN A4

It is possible to assign a fixed size to the HTML object within the CSS (in the *@page rule*). This is also advisable if the reference point is not the upper left corner. If this space is left blank, the default size of DIN A4 will be used.

This means that unless otherwise specified, pdfToolbox will place the generated PDF at DIN A4 size within the PDF file to be fixed.

(This can sometimes lead to irritating but logical results if placing a line of text from index.html within a DIN A4 PDF. Certain placing options (bottom left, middle, etc.) which you may want to access using the pdfToolbox dialog will then appear not to work. This is because it is not just the line of text

which is being placed, but rather the text within an automatically generated A4 page format.)

Custom page formats for content to be placed must be specified using CSS.

This also means that unless specific provisions are made, an HTML object's size will never be defined within its own content. A square placed relative to the bottom-left corner can therefore only be placed exactly on this bottom edge if the position of the edge has been calculated in advance and accounted for either in the HTML or in the Fixup's positioning settings.

Placing multiple HTML "pages"

Multiple pages can be created when processing the HTML template if the dimensions of the objects exceed the defined or default (A4) format. In this case, pdfToolbox will place the first page of the HTML template on the first page, then the second and so on until all pages of the HTML template have been processed. Processing will then begin again on the first page of the HTML template until all pages of the PDF have been filled.

It is therefore entirely possible to use a single template to attach different content to each page of a PDF.

Alternate approach to positioning

As shown above, specifying the desired position for an object defined in HTML can easily get complicated. In many cases it therefore makes sense **to create a page in HTML which matches the dimensions of the PDF page**. This example, when placed in the header of an HTML file, will produce a DIN A3 "page" in landscape format:

```
<style>
  @page {
    size:420mm 297mm;
  }
</style>
```

Click to copy

Objects can then be positioned within this HTML page using HTML and especially CSS. This page will then be placed relative to the upper left corner without any offset.

20.8 Place any content: Simple example placing content in the upper left

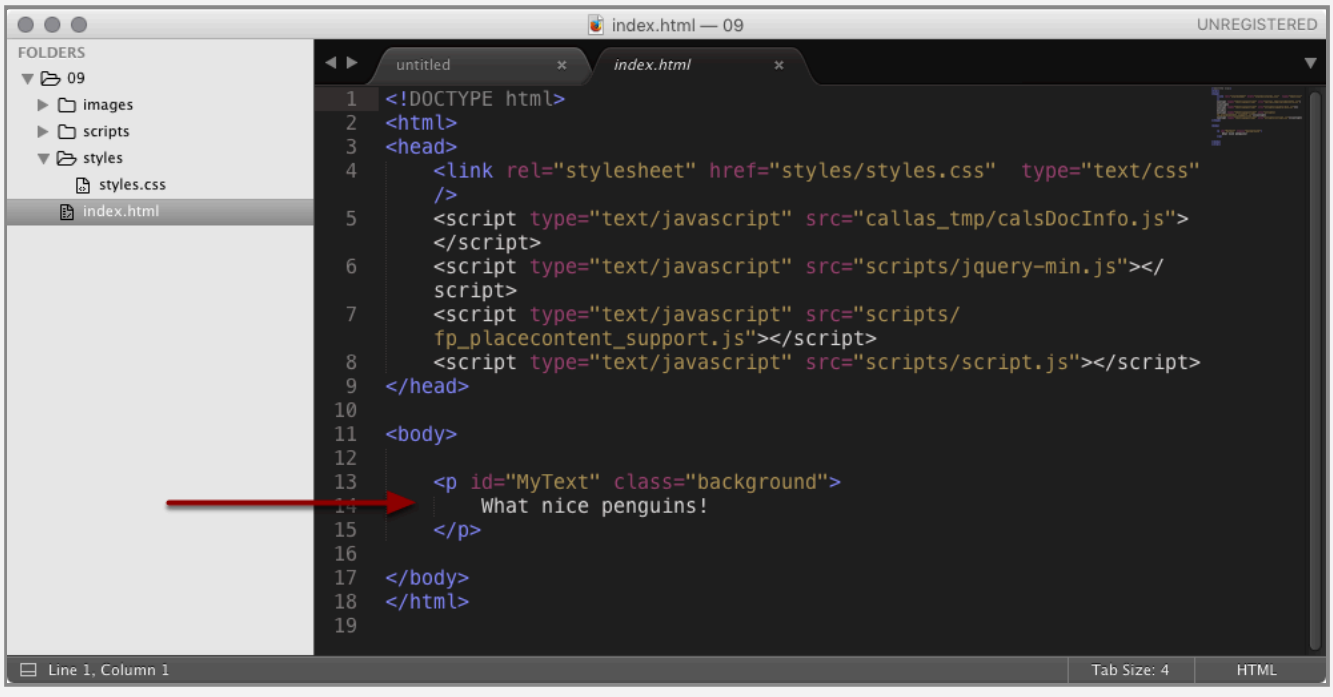
HTMLTemplates.zip

This example uses the HTMLTemplate in folder 09. So-called “absolute positioning” is used in this case. The PDF document in which the content will be placed is in DIN A3 format.

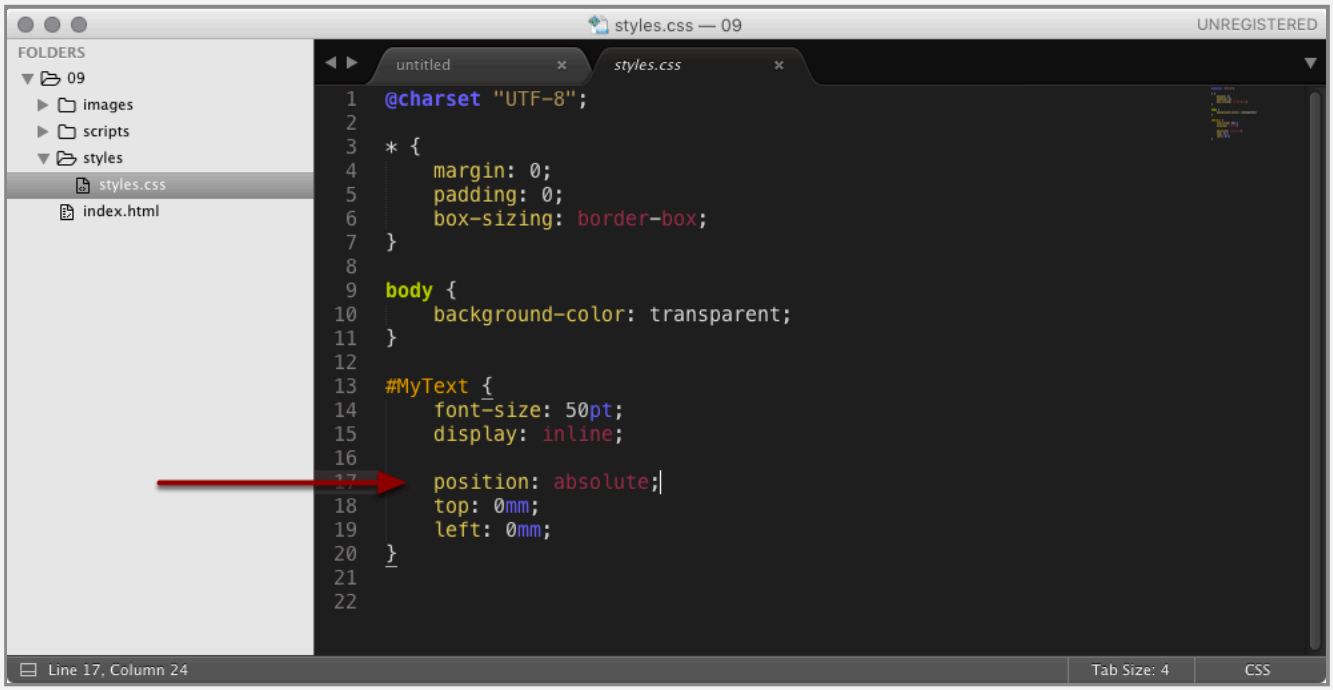
A look at index.html and styles.css

First, we open the folder in the “Sublime Text” text editor. (Other text editors can also be used.)

The file **index.html** contains, among other things, an instruction to place the text *“What nice penguins!”*.



The file styles.css specifies the **text size of 50 pt** and the **position**.



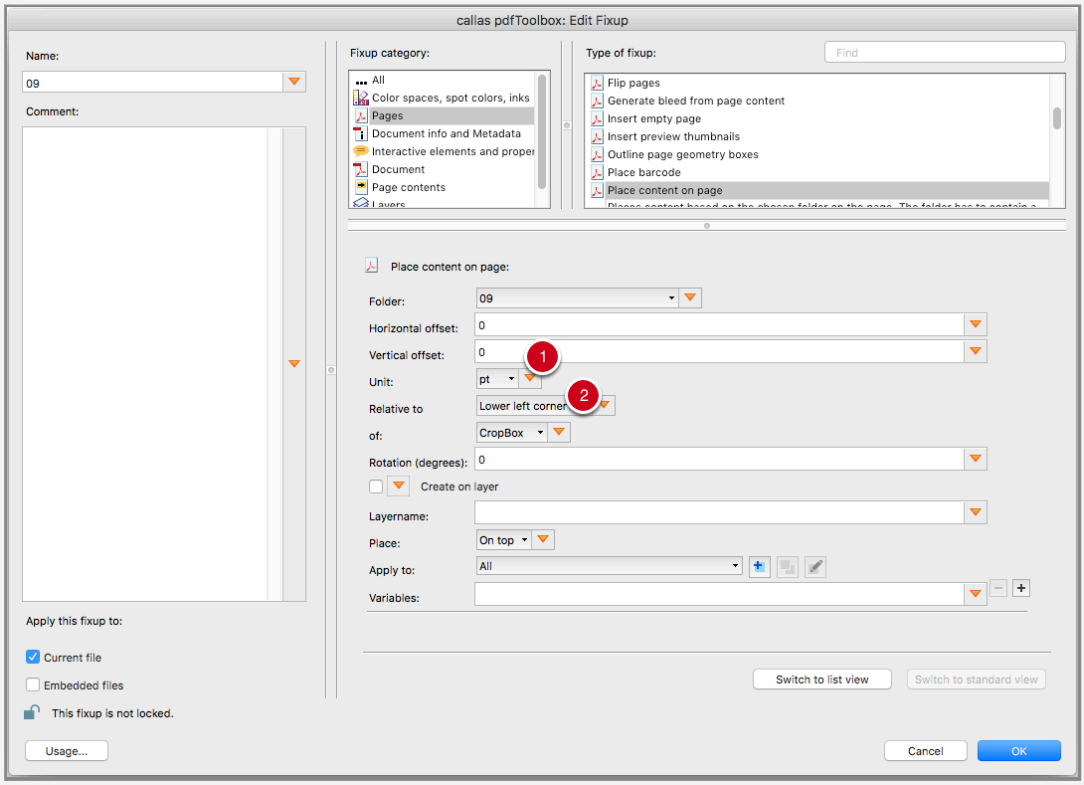
This is a case of **absolute positioning** with a **0 mm offset** from the **top left**.

The code in detail:

```
#MyText {
font-size: 50pt;
display: inline;
position: absolute;
top: 0mm;
left: 0mm;
}
```

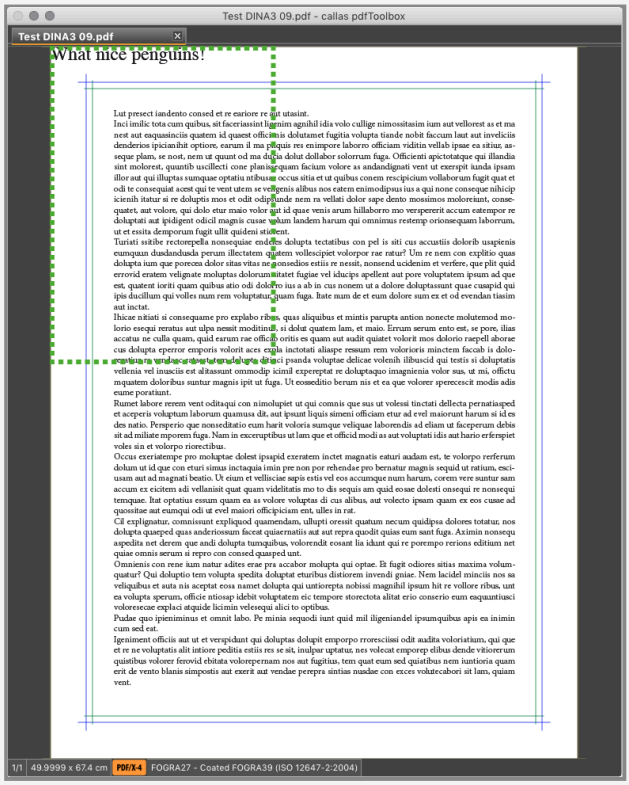
Setting 1 in pdfToolbox: Relative to “Top-left corner” of “CropBox”

The placement of the HTMLTemplate is specified as follows, within pdfToolbox, in the “Place content on page” Fixup:



1. Relative to: “Top-left corner”
2. of: “CropBox”

The result when applied to a PDF document (here in DIN A3 format plus first cut and info area) is as follows:

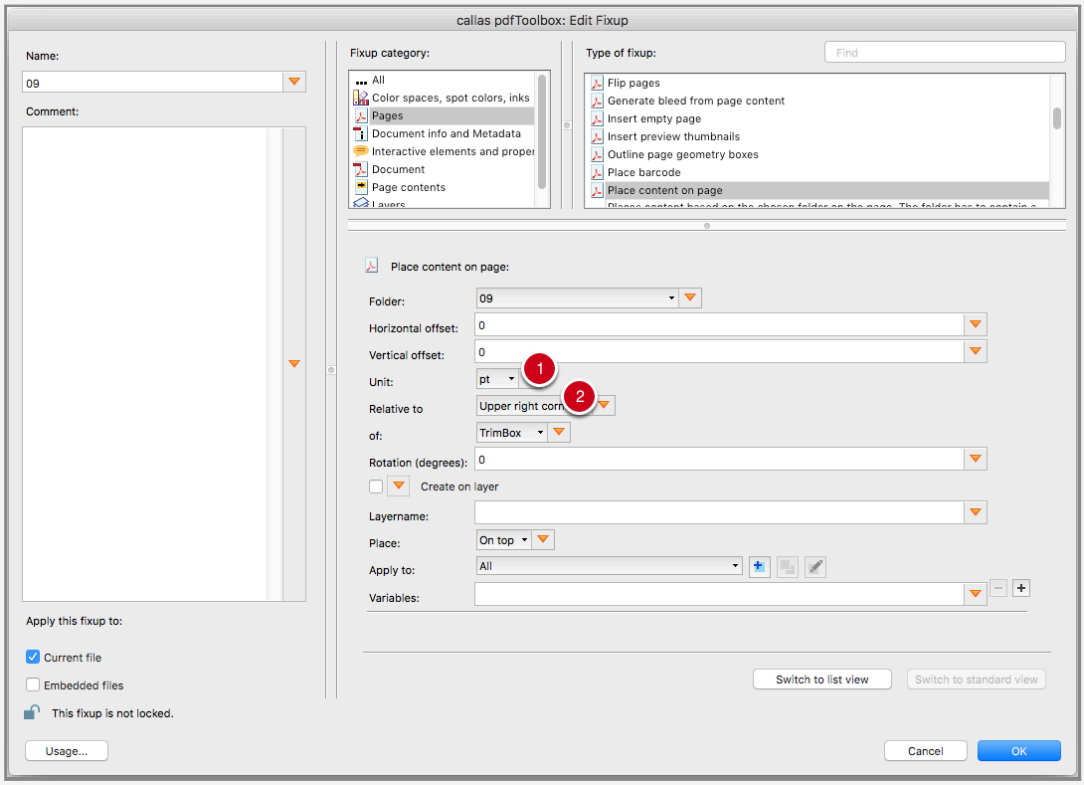


The **50 pt text** is located at the top left **within a PDF document created at the default size of DIN A4** (green dotted line added here for clarity.)

This entire construct is placed within the page on the open PDF document (DIN A3, as already mentioned) **in the top left relative to the CropBox** (the mask space.)

Setting 2 in pdfToolbox: Relative to “Top-right corner” of “TrimBox”

In this example, the placement is configured within pdfToolbox as follows:



1. Relative to: “Top-right corner”
2. of: “TrimBox”

The result with these settings on a page in DIN A3 format:



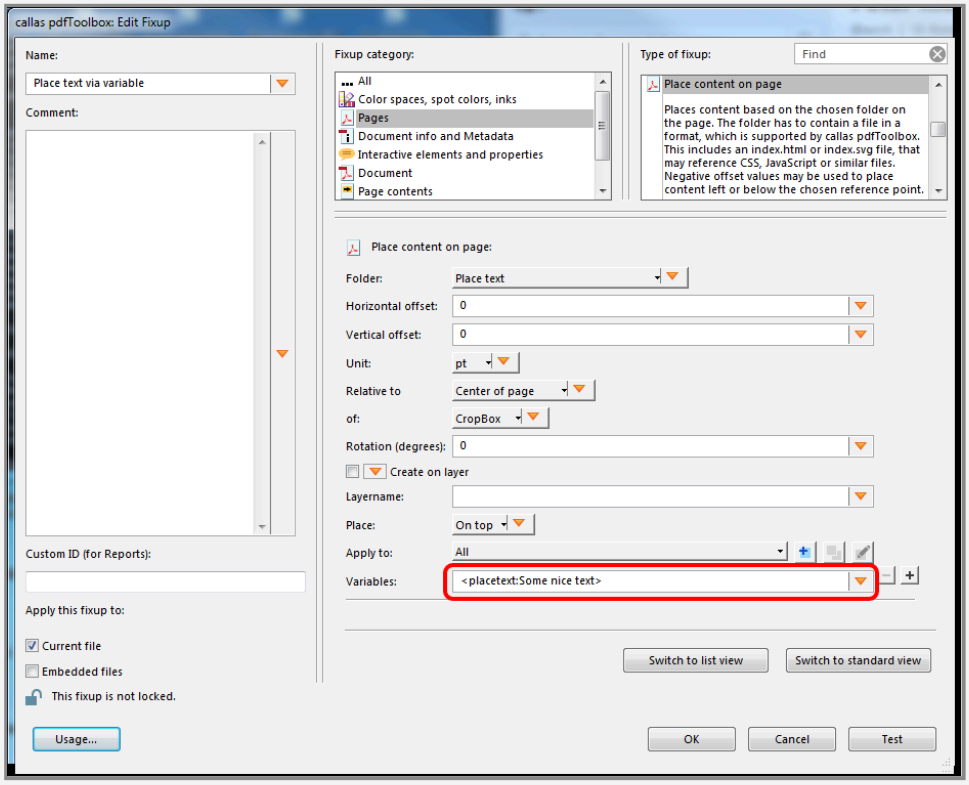
The **50 pt text** is again located at the top left **within a PDF document created at the default size of DIN A4** (green dotted line added here for clarity.)

This entire construct is placed within the page on the open PDF document **at the top right relative to the TrimBox** (the trimmed size space.)

20.9 Using Variables in a Place Content HTML template to place text

As the normal "Place text" Fixup does not offer many options to define e.g. the font to be used for text, using a small template incl. CSS is an easy way to get more control about the layout of the placed text.
To use a template, the type of Fixup "Place content on page" has to be used.

As the text is variable in most cases, a normal variable (with keyword of the Variable: "placetext") is used within the attached Fixup.



In the HTML file, the place where the text shall be positioned is defined using a `<span>` within the body of the HTML. To identify this span, the ID "`textus`" is used

index.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset='utf-8'>
    <link rel="stylesheet" href="styles/styles.css" type="text/css"/>
    <script type="text/javascript" src="callas_tmp/calsDocInfo.js"></script>
    <script type="text/javascript" src="scripts/jquery-min.js"></script>
    <script type="text/javascript" src="scripts/script.js"></script>
</head>

<body>
    <span id="textus"></span>
</body>
</html>

```

It is important to know that the values of Variables are available from the "callas_tmp/calsDocInfo.js" file, which is created during runtime.

With some JavaScript in the "script.js", the current values of Variables of the process run are derived from the "calsDocInfo.js":

```
//
function getVariableValue( inName ) {

    // Get the array with variables
    var theVariables = cals_doc_info.document.variables;

    // Loop over them and try to find the correct one
    for (var theIndex = 0; theIndex < theVariables.length; theIndex++) {
        if (theVariables[theIndex].name === inName) return theVariables[theIndex].value;
    }

    // Found nothing
    return null;
}
```

The respective value for the "placetext" Variable is evaluated and set for the "`textus`" span.

```
for( var thePageIndex = theLowerLimit; thePageIndex < theUpperLimit; thePageIndex++ ) {

    // Set the reference text
    var theVariableReference = getVariableValue( 'placetext' );
    $( '#textus' ).text( theVariableReference );

    // Our generated PDF file will be just as big as the text
    adjustDocumentSizeToHtmlElement( '#textus' );
```

To define formatting of the text, the "style.css" has to be adjusted to your individual needs. In our sample, this css is located in the subfolder CSS of the Template.

To open the Template folder, just select the entry "Open folder with configuration files" in the Folder selection pop up menu in the "Edit Fixup" dialog of "Place content on page".

It is of course also possible to reference a specific font file within the Template folder. This ensures, that the selected font will be used also on other machines, where e.g. the font is not installed.

You can also define now the respective Font-Face (e.g. "Bold" or "Italic") within the CSS.

Please note:
The positioning of text from created by this Template is controlled via the settings in the Fixup.

Place_text_via_variable.kfpx

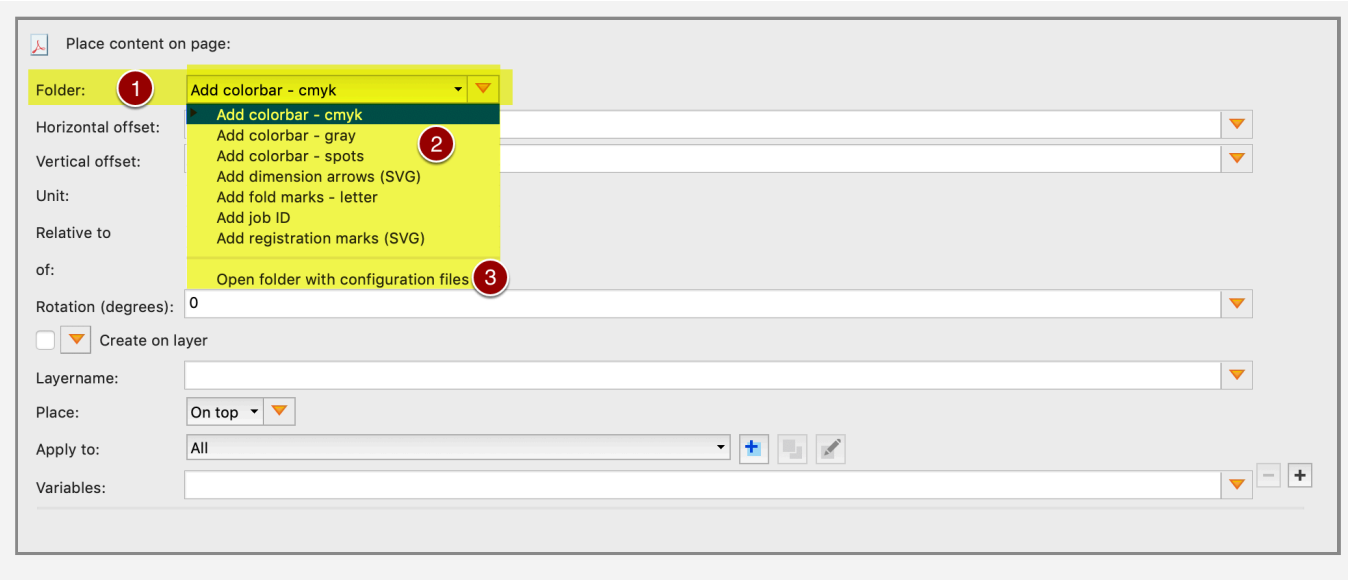
20.10 Place any content: Use information about the PDF document

When placing content, in order to make use of information about the PDF document or the variables used and their values, a file named “calsDocInfo.js” is provided in the “callas_tmp” sub-folder at the time the function is called. This file is updated every time a “Place content” Fixup is run and contains several useful pieces of information:

- File name
- File path
- Variables used for the “Place content” Fixup (or the Profile containing this Fixup), including the values provided for these variables
- Document information metadata fields: Title, Author, Subject, Keywords
- Page geometry borders for each page (MediaBox, CropBox, BleedBox, TrimBox, ArtBox) as well as the “Page label”

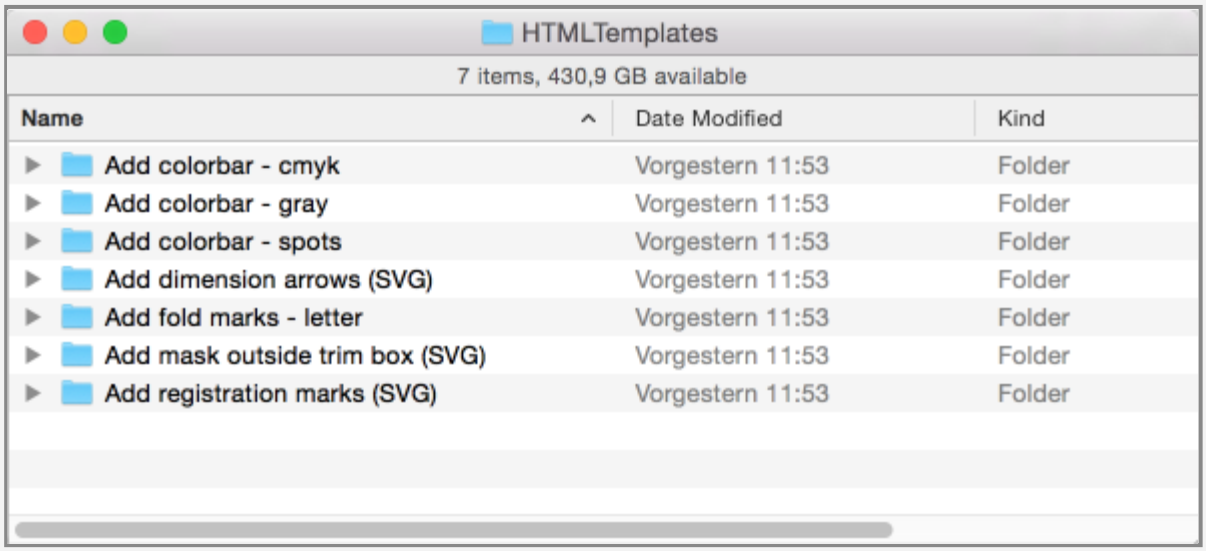
This information is represented as values in the “cals_doc_info” variables and can be read using JavaScript.

Selecting an HTML file folder for content to be placed



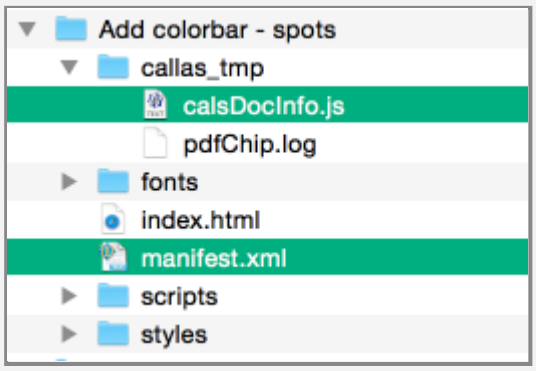
1. Configuring the “HTML file folder”
2. Selected “HTML file folder”
3. Selecting this menu item will take you to the folder containing all existing “HTML file folders”

“Open folder with configuration files” option (#3. from above) opens the corresponding “HTMLTemplates” folder



Selecting “Open folder with configuration files” will take you to the corresponding “HTMLTemplates” folder. Here you can find all folders which contain at least an “index.html” file and which usually also contain other files such as CSS or JavaScript files as well as other required resources such as images or fonts.

Example structure for an HTML file folder containing “calsDocInfo.js” and “manifest.xml”



Important:

- If it does not exist yet, the “callas_tmp” folder will be created by pdfToolbox when the corresponding configured “Place content” Fixup is run for the first time
- The “callas_tmp” folder will contain the corresponding document information for the last processed PDF in the file “calsDocInfo.js”
- If you need to request additional information for the next processing run, the “manifest.xml” file must be placed in the same folder which contains “index.html”
- This manifest.xml file must contain the relevant requests

Example content for a “manifest.xml” file requesting all possible information

```
<?xml version="1.0" encoding="UTF-8" ?>
<manifest xmlns:x="http://www.callassoftware.com/cchip/template/manifest/">
        <x:resources>

                <!--- include ink info -->
                <x:inkinfo/>

                <!--- include ink coverage info-->
                <x:inkcov aperture="1mm" bbox="trimbox"/>

                <!--- include info for items found by "Apply to" check -->
                <x:hitinfo/>

        </x:resources>
</manifest>
```

Content for a “calsDocInfo.js” file containing special color, coverage and hit positioning information for “Apply to” check


```
var cals_doc_info = {
   'document':{
      'name':'zwei Sonderfarben.pdf',
      'path':'\/Users\/odruemmer\/Desktop',
      'info':{   'author':'OD',
                 'title':'zwei Sonderfarben',
                 'subject':'Testdatei für Sonderfarben',
                 'keywords':null       },
      'numberofpages':1,
      'completeinkinformation':true,
      'variables':[ ]                  },
   'pages':[
      {  'mediabox': [ 0, 0, 792, 612 ],
         'trimbox':  [ 0, 0, 792, 612 ],
         'bleedbox': [ 0, 0, 792, 612 ],
         'cropbox':  [ 0, 0, 792, 612 ],
         'artbox':   [ 0, 0, 792, 612 ],
         'pagenumber':0,
         'pagelabel':null,
         'inks':[
            {  'name':"Cyan",
               'usagecm':0,
               'usagepercent':0        },
            {  'name':"Magenta",
               'usagecm':0,
               'usagepercent':0        },
            {  'name':"Yellow",
               'usagecm':0,
               'usagepercent':0        },
            {  'name':"Black",
               'usagecm':1.09066,
               'usagepercent':0.180804  },
            {  'name':"Himmelblau",
               'alternatename':'cmyk',
               'alternatecomps':[  1, 0, 0, 0 ],
               'alternateicc':null,
               'usagecm':37.6872,
               'usagepercent':6.24762   },
            {  'name':"Tomatenrot",
               'alternatename':'cmyk',
               'alternatecomps':[  0.15, 1, 1, 0 ],
               'alternateicc':null,
               'usagecm':37.6685,
               'usagepercent':6.24453   }
         ],
         'hits':[
            {  'bbox':[ 82.2098, 129.289, 253.823, 229.147 ]   },
            {  'bbox':[ 473.391, 209.367, 253.823, 229.147 ]   }
         ]
      }
   ]
};
```


calsDocInfo.js

In an HTML file, in order to access information in the `cals_doc_info` data object, it is necessary to include the JavaScript file calsDocInfo.js from the "callas_tmp" sub-folder, using `<script type="text/javascript" src="callas_tmp/calsDocInfo.js"></script>`:

```
<!DOCTYPE html>
<html>
<head>
        <!-- Define our text encoding as unicode to avoid problems with accents
etc... -->
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
        <!-- Style sheets -->
```

```
        <link rel="stylesheet" href="your/styles.css"  type="text/css"/>
        <!-- Javascripts -->
        <script type="text/javascript" src="callas_tmp/calsDocInfo.js"></script>
<!-- ⬅⬅⬅ -->
        <script type="text/javascript" src="your/javascripts.js"></script>
</head>
<body>
    <!-- your HTML content -->
</body>
```

Click to copy

Thereafter you can access any data inside the `cals_doc_info` data object using common JavaScript conventions, as illustrated by the following example:

```
function getTrimBox( inPageNumber ) {
        var thePage = cals_doc_info.pages[inPageNumber];
        if thePage.trimbox != null) {
                        return thePage.trimbox;
        } else {
                        return thePage.mediabox;
        }
}
```

Click to copy

Example information contained within a “calsDocInfo.js” file:

```
var cals_doc_info = {
        'document':{
                'name':'Example.pdf',
                'path':'\/Users\/odruemmer\/Desktop',
                'info':{
                        'author':'Max Meyer'
                        'title':'On cals_doc_info and other useful things',
                        'subject':null,
                        'keywords':'pdfChip; useful; document information; vari-
able; special colors; coverage; hit positions'
                },
                'numberofpages':2,
                'completeinkinformation':false,
                'variables':[
                        {
                                'name':'minimum',
```

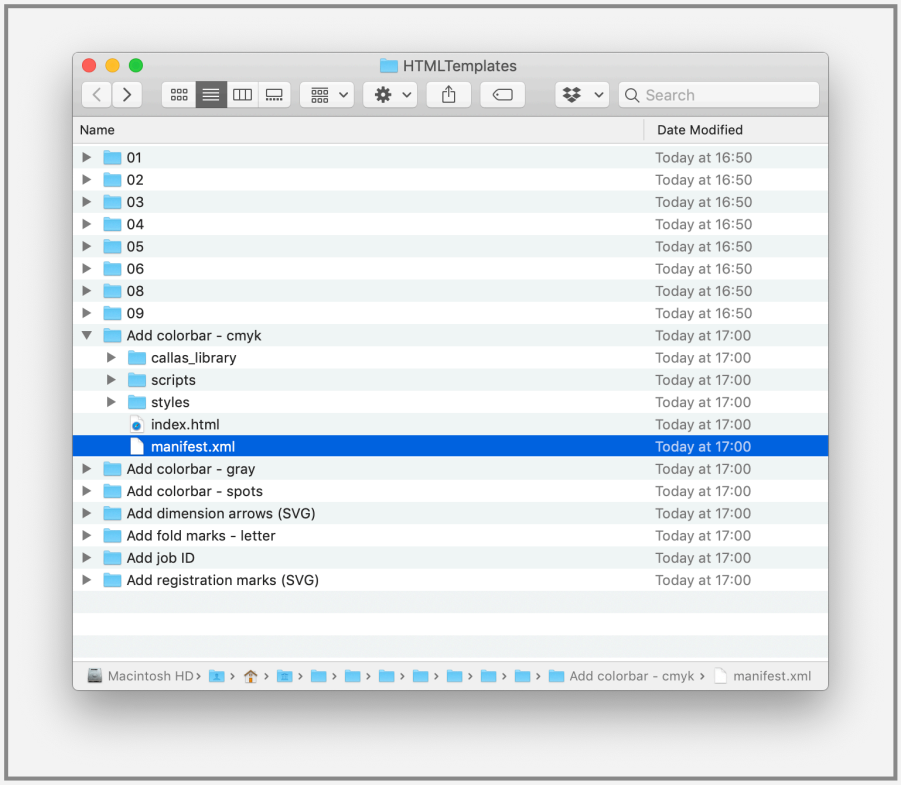
```
                                'value':'0'
                        },
                        {
                                'name':'maximum',
                                'value':'100'
                        }
                ]
        },
        'pages':[
                {
                        'mediabox':[ 0, 0, 4251.97, 2834.65 ],
                        'trimbox':[ 0, 0, 4251.97, 2834.65 ],
                        'bleedbox':[ 0, 0, 4251.97, 2834.65 ],
                        'cropbox':[ 0, 0, 4251.97, 2834.65 ],
                        'artbox':[ 0, 0, 4251.97, 2834.65 ],
                        'pagenumber':0,
                        'pagelabel':'front side'
                },
                {
                        'mediabox':[ 0, 0, 4251.97, 2834.65 ],
                        'trimbox':[ 0, 0, 4251.97, 2834.65 ],
                        'bleedbox':[ 0, 0, 4251.97, 2834.65 ],
                        'cropbox':[ 0, 0, 4251.97, 2834.65 ],
                        'artbox':[ 0, 0, 4251.97, 2834.65 ],
                        'pagenumber':1,
                        'pagelabel':'back side'
                }
        ]
};
```

You can also keep the temporary calsDocInfo.js via:

```
<x:settings>
        <!--- if 'true' temporary files are moved next to the resulting PDF file
(useful for developing templates) -->
        <x:keeptemp>true</x:keeptemp>
</x:settings>
```

Click to copy

Manifest.xml: Request additional information



Additional information about the PDF document can be requested with the help of a so-called manifest file ("manifest.xml"). This manifest file can additionally request the following types of information:

- Name and replacement appearance for special colors contained within the PDF: **<x:inkinfo/>**
- Coverage per color channel per page: **<x:inkcov aperture="1mm" bbox="trimbox"/>**; Note: This information requires each page to be rendered using a configurable resolution (specified using the blend value *aperture*), which can result in the Fixup taking longer to run.
- The *bounding box* per "hit" for an "apply to" check set up within the "Place content" Fixup: **<x:hitinfo/>**

These details for requesting information must be contained in an <**x:resources**> note within the <**manifest**> root node:

```
?xml version=1.0 encoding=UTF-8 ?
manifest xmlns:x=http://www.callassoftware.com/cchip/template/manifest/
```

```
x:resources
 ... one or more entries requesting information ... /
/x:resources
/manifest
```

If all available information is requested, the manifest file will look like this:

```
?xml version=1.0 encoding=UTF-8 ?
manifest xmlns:x=http://www.callassoftware.com/cchip/template/manifest/
x:resources

!--- include ink info (ink names and alternate appearance only): --
x:inkinfo/

!--- include ink coverage info: ink names, usagecm, usage percent
   Ink coverage is determined by rendering the bbox of each
   page with the specified aperture; units for aperture can be pt or mm
   bbox can be mediabox, cropbox, bleedbox, trimbox
   (Default: bbox: trimbox, aperture: 15mm) --
x:inkcov aperture=1mm bbox=trimbox/

!--- include bbox info for items found by Apply to check --
x:hitinfo/

/x:resources
/manifest
```

An example of a “calsDocInfo.js” file containing the maximum amount of information follows. The following items are of particular interest:

- **“completeinkinformation”** is set to *true*; this means that both the existing special colors are listed (including their alternative appearance in CMYK values) as well as all color channels including each one's coverage (in percent based on the area specified (e.g. *trimbox*) and in absolute terms as an area in square centimeters)
- Each document page has an entry named **“inks”** which includes the aforementioned information about special colors and coverage for all color channels.
- Tonal values are given for the two special colors, **“sky blue”** and **“tomato red”** for their alternative appearance in CMYK.

- The *bounding boxes* are listed under **“hits”** (**“bbox”**) for each hit in the “Place content” Fixup containing the “Apply to” check - in this example, we are testing for special color objects.

```
var cals_doc_info = {
  'document':{
   'name':'zwei Sonderfarben.pdf',
   'path':'\/Users\/odruemmer\/Desktop',
   'info':{
     'author':'OD',
     'title':'two special colors',
     'subject':'Test file for special colors',
     'keywords':null
   },
   'numberofpages':1,
   'completeinkinformation':true,   'variables':[
   ]
  },
  'pages':[
   {
     'mediabox':[  0, 0, 792, 612 ],
     'trimbox':[  0, 0, 792, 612 ],
     'bleedbox':[  0, 0, 792, 612 ],
     'cropbox':[  0, 0, 792, 612 ],
     'artbox':[ 0, 0, 792, 612 ],
     'pagenumber':0,
     'pagelabel':null,
     'inks':[
      {
        'name':Cyan,
        'usagecm':0,
        'usagepercent':0
      },
      {
        'name':Magenta,
        'usagecm':0,
        'usagepercent':0
      },
      {
        'name':Yellow,
        'usagecm':0,
        'usagepercent':0
      },
```

```
      {
        'name':Black,
        'usagecm':1.09066,
        'usagepercent':0.180804
      },
      {
        'name':"sky blue",
        'alternatename':'cmyk',
        'alternatecomps':[
         1, 0, 0, 0
        ],
        'alternateicc':null,
        'usagecm':37.6872,
        'usagepercent':6.24762
      },
      {
        'name':"tomato red",
        'alternatename':'cmyk',
        'alternatecomps':[
         0.15, 1, 1, 0
        ],
        'alternateicc':null,
        'usagecm':37.6685,
        'usagepercent':6.24453
      }
     ],
     'hits':[
      {
        'bbox':[ 82.2098, 129.289, 253.823, 229.147 ]
      },
      {
        'bbox':[  473.391, 209.367, 253.823, 229.147 ]
      }
     ]
   }
  ]
};
```

For processing to succeed, it is important for the file to be named manifest.xml and for it to be located in the processing folder (alongside the index.html file.)

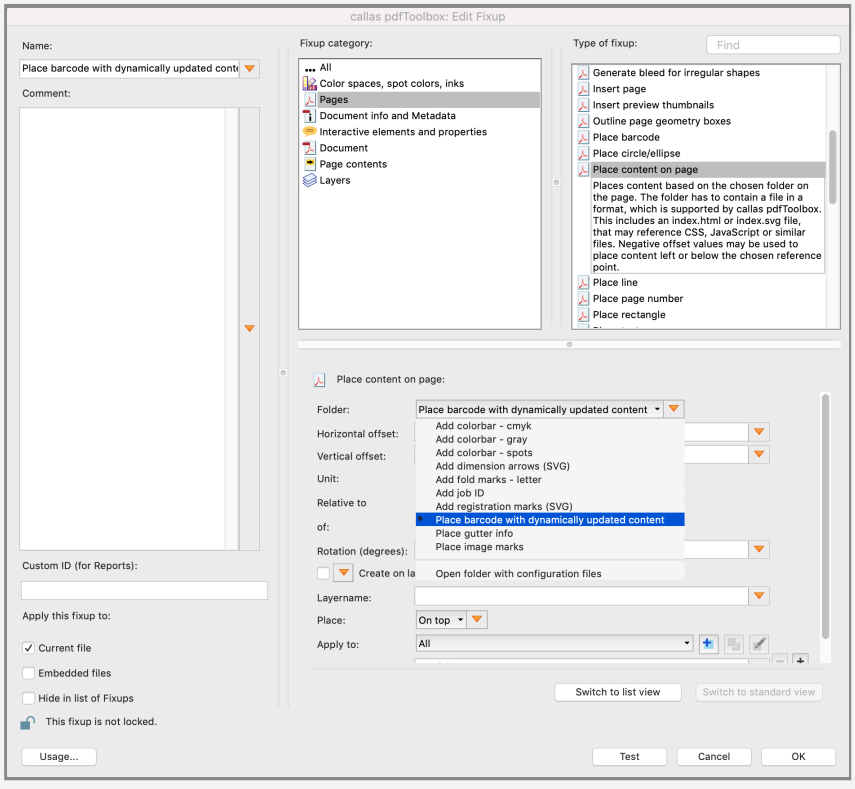
20.11 Place barcode: Using an HTML-template for an extended configuration

It is possible to use many more parameters when placing barcodes with pdfToolbox (and not just the ones in the Fixup).

To keep the user interface straight forward, the number of possible settings for the more than 100 different barcodes we support are available when using HTML-based template positioning of content.

Placing a "Code 39" barcode: simplified example

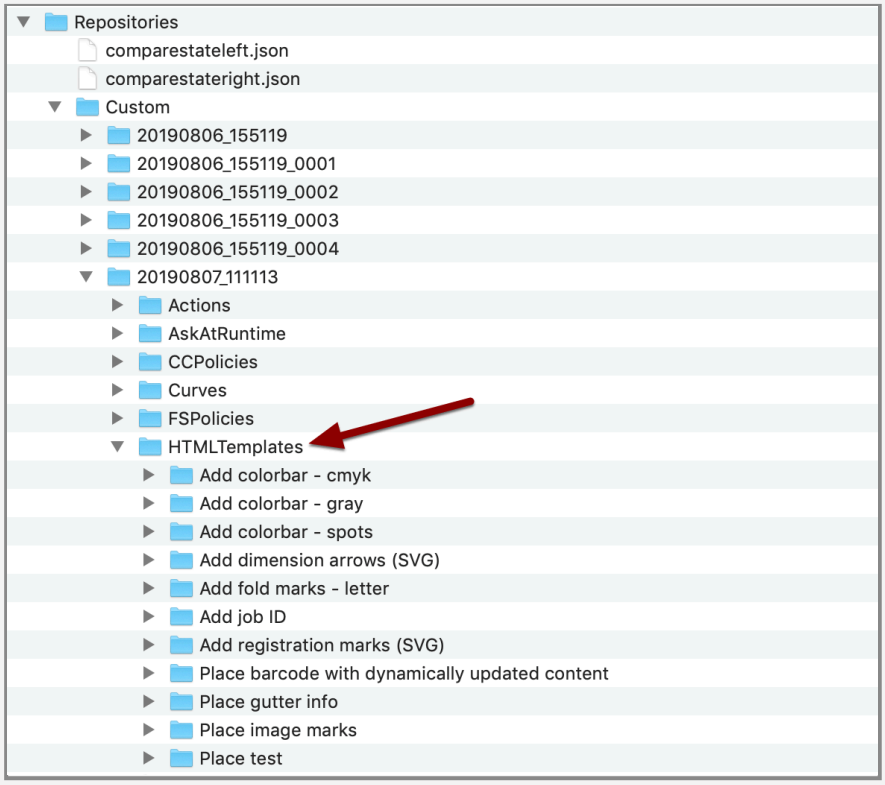
You first need to setup a new Fixup based on “Place content on page” Fixup type.
How to set up a Fixup like this is explained very well in [this article](#).



Using an HTML template in order to place a barcode, you will have to set up the HTML templates folder.

Click on 'Open folder with configuration files' in the Folder dropdown of the Fixup and once you have the defined the folder and file structure for HTML, styles and scripts, you can use this template in your 'Place content' Fixup.

In order to place a barcode as in this example, the configuration files are set up under 'Place barcode with dynamically updated content' folder under HTML templates.



The folder 'Place barcode with dynamically updated content' contains the obligatory **index.html**, the **scripts** folder containing JavaScript files, the **styles** folder for CSS files.

The index.html as in the code below has the body section as the most interesting and important part. Alongwith the width, height and the type of barcode, it has some additional parameters.

```
<html>
<head>
        <meta charset='utf-8'>
        <link rel="stylesheet" href="styles/styles.css"  type="text/css"/>
        <script type="text/javascript" src="callas_tmp/calsDocInfo.js"></script>
        <script type="text/javascript" src="scripts/jquery-min.js"></script>
```

```
        <script type="text/javascript" src="scripts/script.js"></script>
</head>

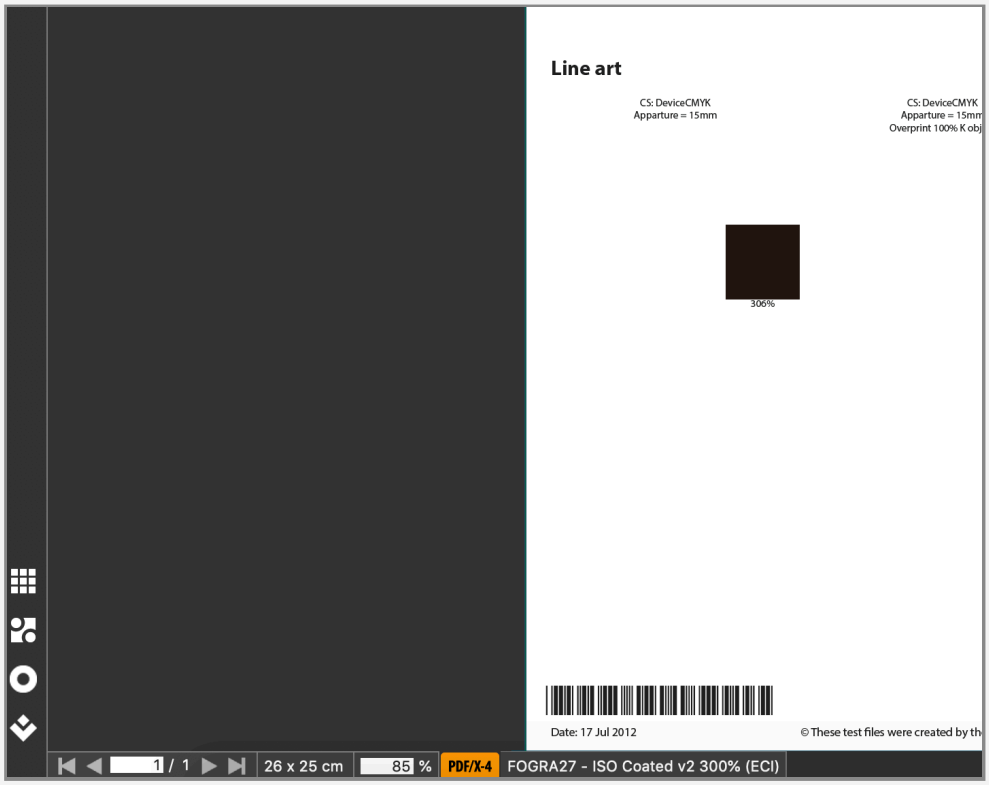
<body>
        <object id="id_barcode" class="barcode" type="application/barcode"
style="width:50mm; height:10mm;">
                <param name="type" value="Code 39 Full ASCII">
                <param id="id_barcodedata" name="data" value="default">
                <param name="quietzonebottom" value="2">
                <param name="quietzonetop" value="2">
                <param name="quietzoneleft" value="2">
                <param name="quietzoneright" value="2">
                <param name="quietzoneunit" value="mm">
                <param name="textplacement" value="none">
        </object>
</body>
</html>
```

Click to copy

To define the type of barcode that you want to place, you can select one from over 100 types as defined here. Our example above says '<param name="type" value="Code 39 Full ASCII">'

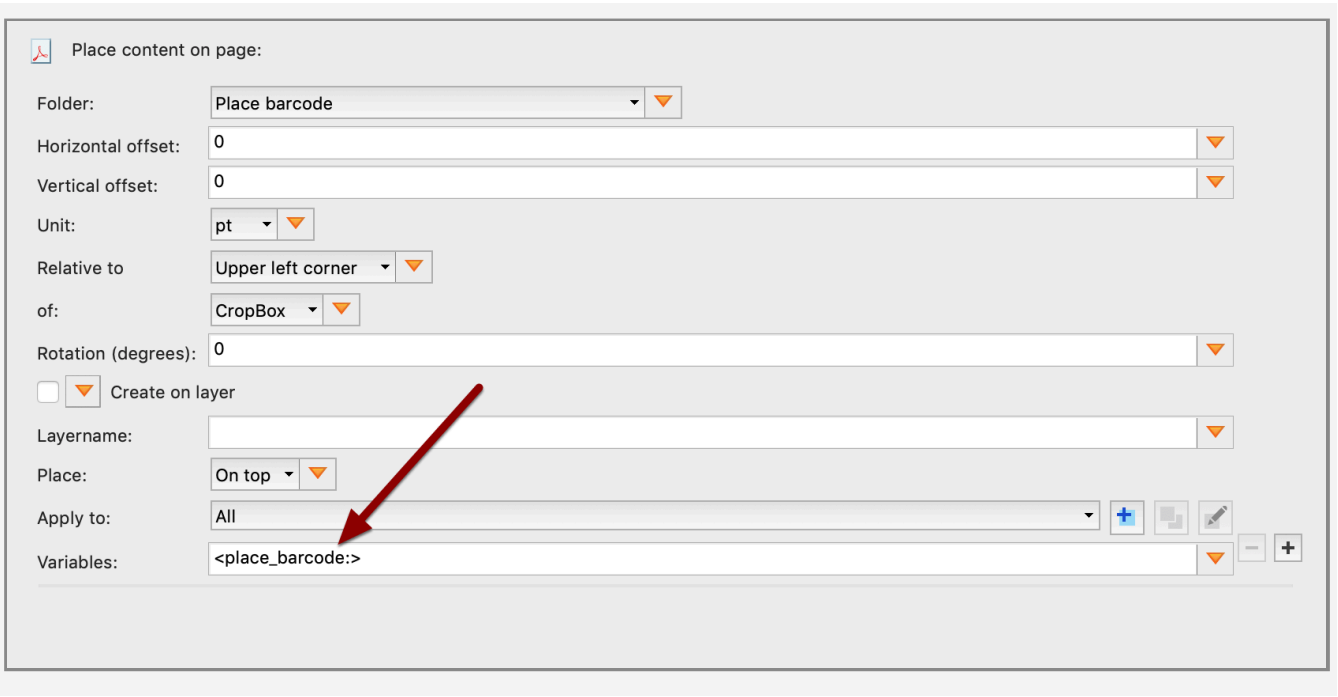
Other parameters like 'quietzonebottom', 'quietzonetop' and tens of more settings can be defined in the index.html and are explained in full detail here.

Depending on the **placing parameters** you selected in the pdfToolbox dialog (bottom left, top right, middle, etc. as well as frames of reference like CropBox, TrimBox, etc.), the "Code 39" barcode will be added to the PDF on running the Fixup.



Get the barcode value from the defined variable into the HTML-template (using Javascript)

Attached below is a 'Place barcode' Profile that uses Javascript to get the barcode value in the HTML template from the variable as defined in the Fixup configuration. The variable named 'place_barcode' for barcode value is defined in the Fixup configuration (marked below with an arrow):



script.js uses this variable value and sets it in the HTML template (only a part of the script is shown below):

```
function cchipPrintLoop() {

        // Calculate the page range we need to execute this on
        var theLowerLimit = 0;
        var theUpperLimit = cals_doc_info.pages.length;

        // Loop over all pages
        for( var thePageIndex = theLowerLimit; thePageIndex < theUpperLimit; thePa-
geIndex++ ) {

                // Set the variable value
                const theVariablevalue = getVariableValue( 'place_barcode' );
                setBarcode( theVariablevalue );

                // Our generated PDF file will be just as big as the barcode
                adjustDocumentSizeToHtmlElement( '#id_barcode' );

                // Output to the current page
                cchip.printPages(1);
        }
}
```

```
//-------------------------------------------------------------------------------
---------------
// Utility routines
//-------------------------------------------------------------------------------
---------------

function setBarcode(d) {
.
.
.
```

Click to copy

In case you are writing your own script or making changes to the one in the attached Profile, it is very important to force an update for parameters of the barcode object, like below:

```
//force update for parameters of the barcode object:
                var b = document.getElementById('id_barcode');
                b.innerHTML = b.innerHTML;
```

Click to copy

Place_barcode.kfpx

As you by now know that the "Place Content" fixup in callas pdfToolbox uses an HTML file and converts it to PDF. This gives you a lot of liberty in how to structure your HTML, in how you link to other files such as scripts, CSS and fonts and in how to structure the folder you use to hold that template. We have a template folder on GitHub where you find one possible way to do this. You'll find a liberal use of sub folders and CSS and Javascripts that are referred to, rather than included in the HTML file. You're free to follow this standard or implement your own.

20.12 Advanced 2D code use cases

Deutsche Post DP Matrix, Data Matrix Industry, rainbow colored QR Code (9.1)

pdToolbox 9.1 comes with a number of extended capabilities that makes it possible to create barcodes and matrix codes for all kinds of industries and use cases. This article illustrates the possible use of the barcode and matrix code creation in pdfToolbox in the form of several interesting examples.

Note: These examples make use of a couple of advanced features that are not feasible with the "Place barcode" fixup, but require use of the more advanced "Place dynamic content" fixup and custom written HTML and JavaScript.

pdfToolbox Library with pre-configured example fixups

Please feel free to download and import the pdfToolbox Library provided below:



Barcode_and_matrix_code_examples.kfpl

After import you will find the examples below in the Fixups area of pdfToolbox.

Example: Deutsche Post DP Matrix code examples

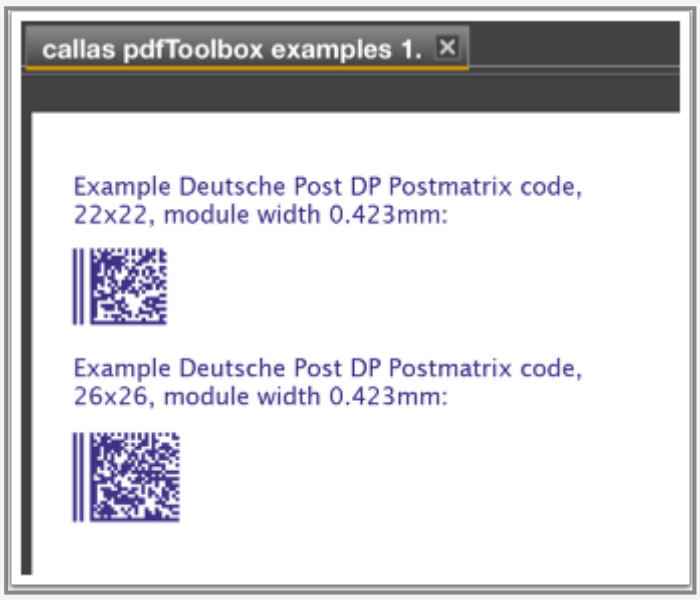
Deutsche Post DP Matrix 2D codes have to follow very strict specification. It is based upon DataMatrix codes, combined with a number of Deutsche Post specific rules.

The example fixup provided creates two variants of such a Deutsche Post DP Matrix code in the upper left of every page of the currently open document.

At the core of custom HTML template is the following code:

```
                <object class="barcode_object"  type="application/barcode" >
                        <param name="type" value="Data Matrix">
                        <param name="data" value="***insert data!***">
                        <param name="modulewidth" value="0.423mm">
                        <param name="dm_format" value="PostMatrix">
                                <!-- **empty string**, UCCEAN, Industry, _Macro05,
Reader, PostMatrix -->
                        <param name="dm_size" value="22x22">
                                <!-- **empty string**, 10x10, 12x12, 14x14, 16x16,
18x18,
                                                20x20, 22x22, 24x24, 26x26, 32x32,
36x36, 40x40,
                                                44x44, 48x48, 52x52, 64x64, 72x72,
80x80, 88x88,
                                                96x96, 104x104, 120x120, 132x132,
144x144, 8x18,
                                                8x32, 12x26, 12x36, 16x36,
16x48                                        -->
                        <param name="dm_enforcebinaryencoding" value="false">
                                <!-- **false**, true-->
                        <param name="dm_rectangular" value="false">
                                <!-- **false**, true-->
                </object>
```

Applying this code with proper CSS styling will put a Deutsche Post DP Matrix code in the upper left corner on the pages of the currently open PDF :



Example: DataMatrix Industry 2D code 16x48

DataMatrix codes come in various flavors specific to the sector where they are used. For certain industry uses a sub-type "Industry" exists, that uses a rectangular form not a square form of DataMatrix codes

The example fixup provided creates one variants of 16x48 cells, and places it in the lower left of every page of the currently open document.

At the core of custom HTML template is the following code:

```
<object class="barcode_object"  type="application/barcode" >
        <param name="type" value="Data Matrix">
        <param name="data" value="Actual data"> <!-- <<== Actual data must go here
-->
        <param name="modulewidth" value="0.25577mm">
        <param name="dm_format" value="Industry">
        <param name="dm_rectangular" value="true">
        <param name="dm_size" value="16x48">
<!--
        <param name="dm_size" value="16x48">
        supported values:
        **empty string**, 10x10, 12x12, 14x14, 16x16, 18x18, 20x20, 22x22, 24x24,
        26x26, 32x32, 36x36, 40x40, 44x44, 48x48, 52x52, 64x64, 72x72, 80x80,
88x88,
        96x96, 104x104, 120x120, 132x132, 144x144, 8x18, 8x32, 12x26, 12x36,
16x36, 16x48
-->
</object>
```

Applying this code with proper CSS styling will put a DataMatrix Industry 2D code 16x48 in the lower left corner on the pages of the currently open PDF:



Example: Rainbow colored QR Code with your name

QR codes have many uses. This example illustrates a concept that won't be acceptable when it comes to maximizing readability of codes in an industrial environment, but can still put to good uses in some creative scenarios. Be prepared though to accept that such codes will not conform to any of the applicable ISO standards. Still they can be scanned surprisingly well with your average smartphone barcode app or any up-to-date 2D code reader.

Note: The technique shown here can also be applied to any other type of barcode and matrix code.

The example fixup provided creates a rainbow color QR code with the name being entered upon executing the fixup.

At the core of custom HTML template is the following code:

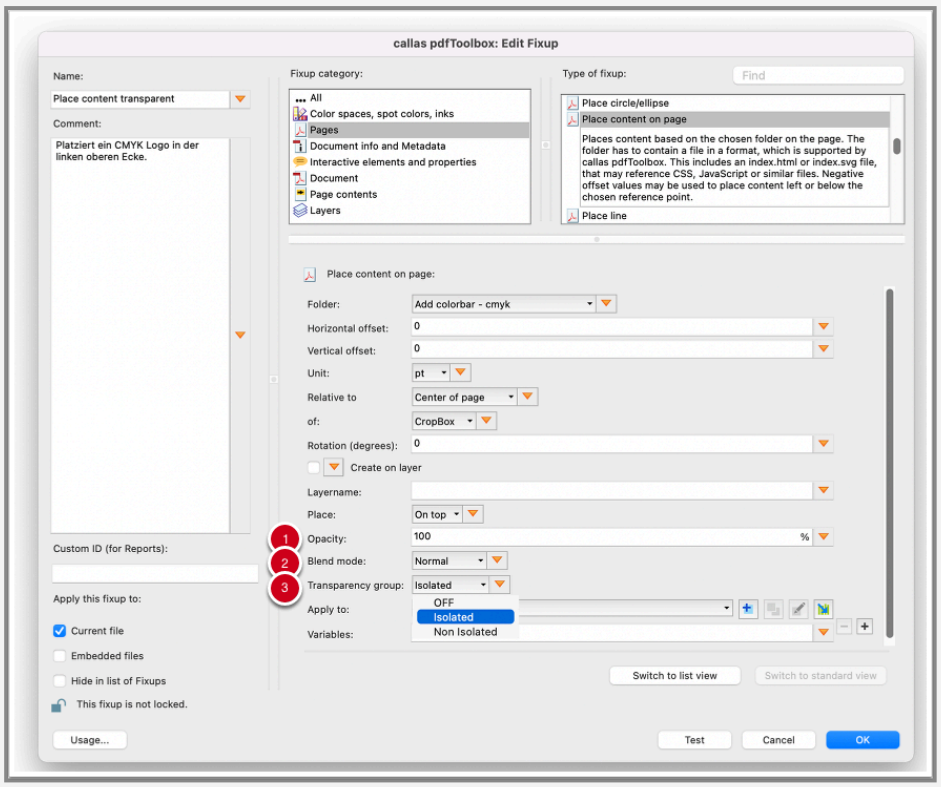
```
<object id="barcode_object"                type="application/barcode"
                style = "color: #eee;        color: -cchip-cmyk(0,0,0,0.1);
                background-color: pink;
                background: linear-gradient(135deg, firebrick, red, orange, or-
ange, green, blue, indigo, violet); "
>
        <param name="type" value="QR-Code">
```

```
        <param name="modulewidth" value="1mm">
        <param name="data" id="id_barcodevalue" value="fill in actual value<param
name="swap_foreground_background" value="true">
        <param name="quietzoneleft" value="1">
        <param name="quietzoneright" value="1">
        <param name="quietzonetop" value="1">
        <param name="quietzonebottom" value="1">
        <param name="quietzoneunit" value="X">
</object>
```



20.13 Place content transparently or opaquely

Since pdfToolbox 13.1, extended parameters for the existing Fixup "**Place content on page**" are available, that allow you to deal with transparencies as explained below.



1. In this field, you can set the **Opacity** in percentage. Instead of placing the content opaquely over the existing page content, it can appear transparent on the page using this parameter, so that the original PDF still shines through.
2. You can select a **Blend mode** for the transparency.
3. If the content to be placed should be transparent, an isolated or non-isolated **Transparency group** must be defined.

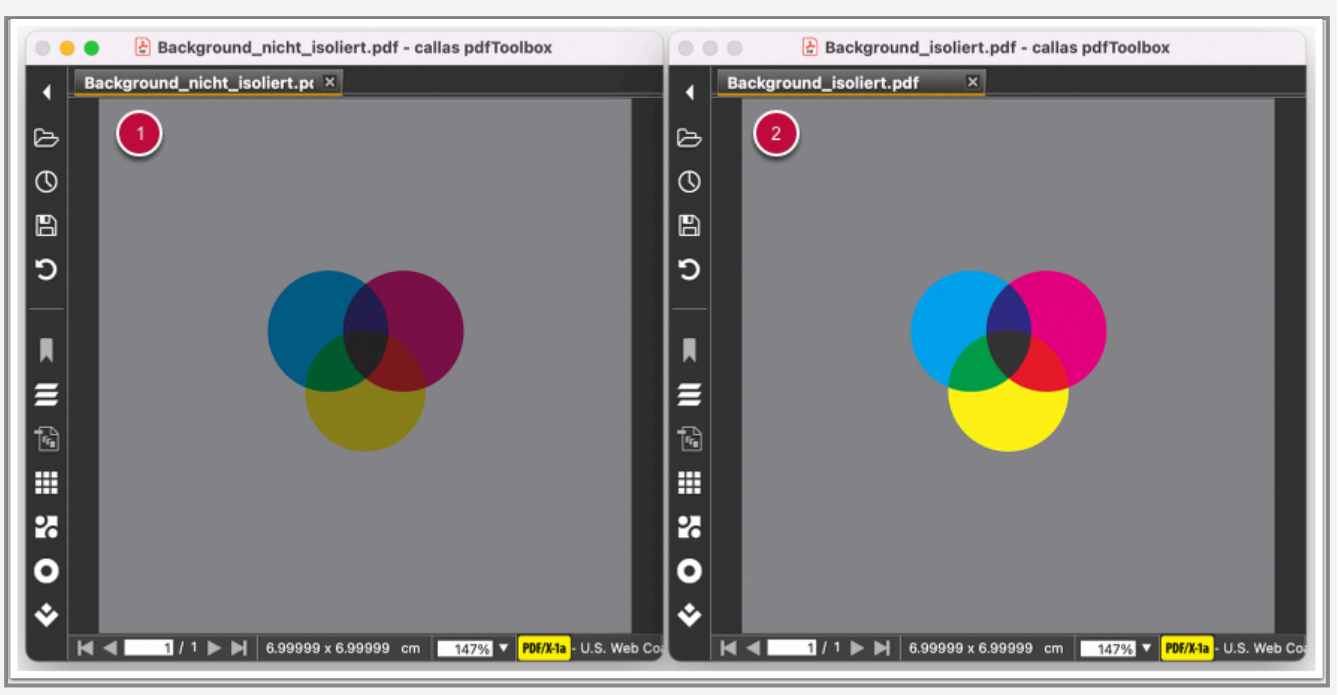
Two use cases:

Place content transparently on page

If a content that does not contain transparency should be placed transparently on a page, it can be realised with the **Opacity** parameter. In addition to a percentage for the transparency, a **Transparency group** must be defined to activate the transparency. Whether the group is set to isolated or non-isolated is irrelevant for non-transparent content. Both settings generate the same result.

Place transparent content on page

If you want to place content, that is already transparent and interacts transparently with each other, over the existing PDF file, in many cases the appearance of this content should not be influenced by the PDF page. To ensure this, pdfToolbox offers the possibility to create an **isolated Transparency group**. That means that all the transparencies that are specified within that group are still there but do not interact with the existing background of the PDF file.



1. The logo has been placed as a **non-isolated transparency group**.
2. The logo has been placed as an **isolated transparency group**.

20.14 Overview of pdfChip versions in pdfToolbox

pdfToolbox	pdfChip
16.1.661	2.6.098
16.0.656	2.6.095
15.1.639	2.5.084.1
14.4.621	2.5.084
14.2.612	2.5.083
14.1.606	2.5.080
13.0.576	2.4.072
12.1.556	2.3.070
12.0.552	2.2.068
11.1.542	2.2.066
10.2.503	2.1.059
10.1.490	2.0.056
10.1.484	2.0.053
10.1.482	2.0.052

21. Shapes

21.1 Shapes: An overview

Shapes are a way to use existing content or page information and either derive new content or clip the existing content on a page.

This can come in handy in number of scenarios, e.g. when:

- adding white background to be printed behind the page content, but not in areas where nothing is printed
- adding partial varnish on certain objects
- creating a dieline based on page content or page geometry
- clipping page content where for example irregularly shapes have to be imposed without wasting space on the imposed sheet as would be the case if the imposition would be based on the bounding rectangle of the label
- create versions of complex production files that clip or overlay distracting technical content and allow an unobstructed view of the main page content, e.g the label as it will actually appear out of the printing process

In order to enable these and many more uses, the Shapes feature are configured in two steps

- the actual shape(s) have to be defined; at this stage "shapes" are considered to just be abstract definitions of some area(s) on a page
- next, an action is defined that is executed using the defined shape(s); for example, a shape can be filled, outlined or used for clipping

Both steps come with a number of parameters that determine exactly how shapes are created, or how actions are to be executed. The necessary details are discussed in the next two articles.

Designing shapes

In many cases, the use of shapes will be obvious. For example, when creating a die line based on the trim box, optionally with rounded corners, a user would simply define shape based on the page's trim box, set rounded corner radius to 3mm, and would then define the action to be applied to the sahpe as a spot color outline with a spot color named "Dieline".

In other cases more complex requirements may have to be accommodated. For example, when a partial has to be created over any part of the page where something is actually printed, except one area where the barcode goes, as this area shall not become glossy (it is difficult to read barcodes with a glossy surface). While this can be achieved relatively easily using the Shapes feature, it can quickly become confusing, especially at the beginning, as Shapes are defined in a very abstract fashion.

In all cases where shapes are to be defined and used in non-trivial ways, it is highly recommended to make a simple drawing, using old fashioned pencil and paper, reflecting the expected page content page geometry and so forth, and the draw the shape information to be derived, and how it is to be derived based on existing information. When doing this, please also note, that in some cases two or more separate steps may be necessary. In such a case a Process Plan may be used, that runs a sequence of Shape steps.

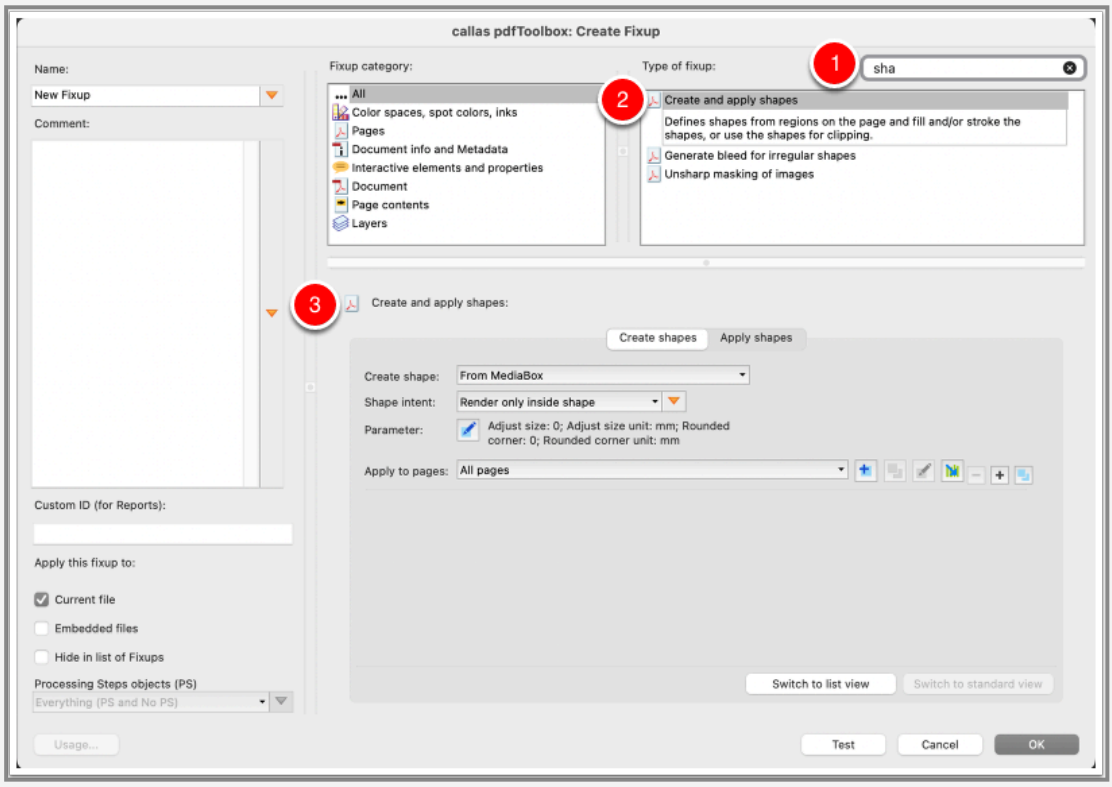
21.2 Defining shapes

Working with shapes means using the "Create and apply shapes" Fixup. pdfToolbox comes with a small set of predefined "Create and apply shapes" Fixups, which can be found in the "Shapes, Variables, JavaScript, Place content" Library. The content below will explain how to create custom "Create and apply shapes" Fixups - specifically, how to configure the part that creates shapes. Applying actions to shapes will be explained in the next article.

Creating a Shapes Fixup

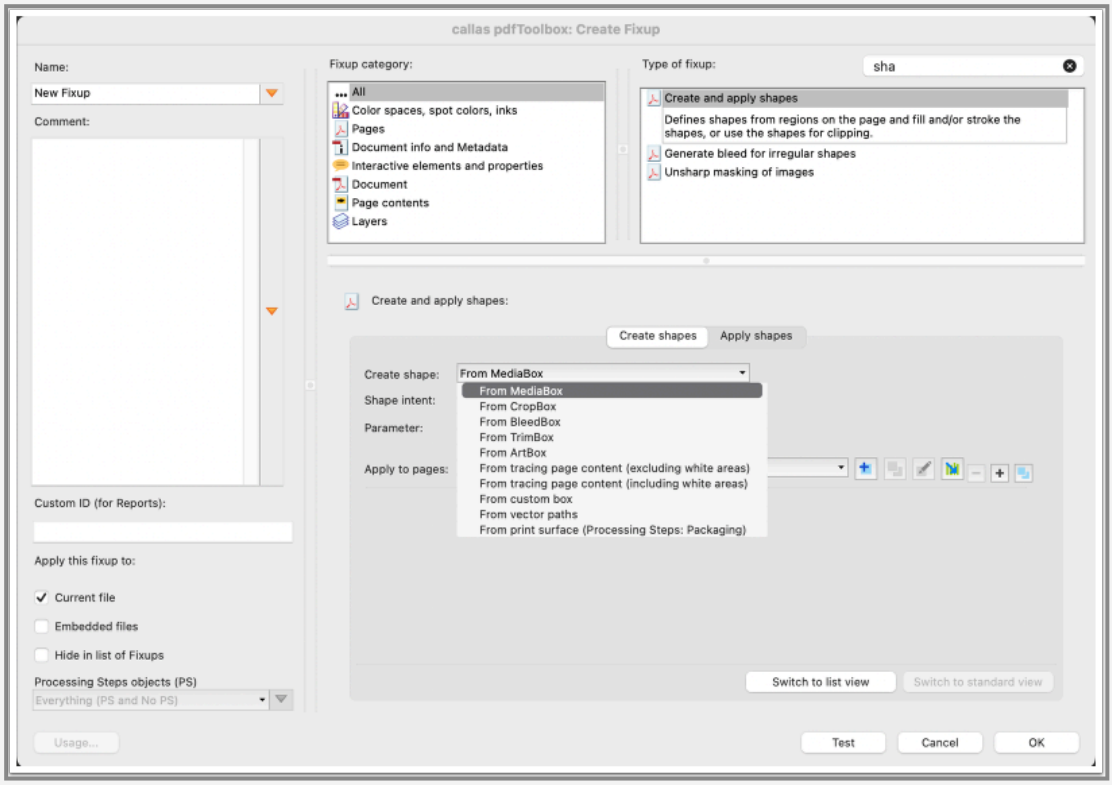
In order to create a new Shapes Fixup, simply create a new Fixup, search for "Create and apply shapes" under "Type of Fixup" and select "Create and apply shapes"

Setting up new Fixup as "Create and apply shapes" Fixup



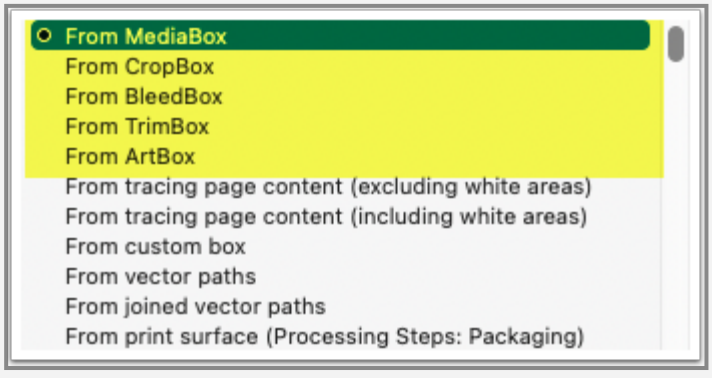
1. Enter "shape" (or just "sha") in the search field
2. Select the "Create and apply shapes" entry in the "Type of Fixup" list
3. Use the options under "Create shapes" the configure how shapes are to be created

"Create shape" parameter



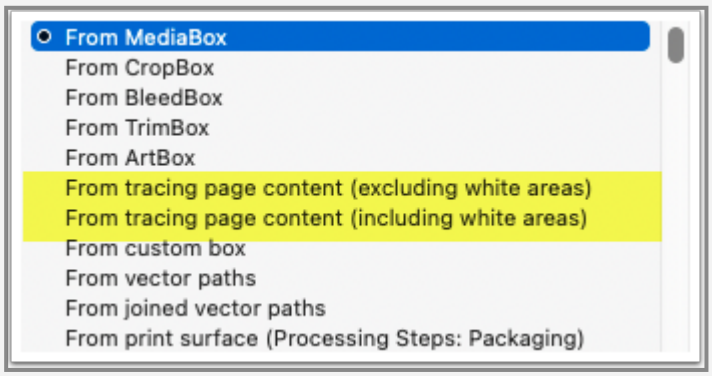
There are a number of approaches how one or several shapes can be created. The steps below will discuss each (group of) such approaches.

Discussion of the "Create shape" parameters: MediaBox, CropBox, B leedBox, TrimBox, ArtBox



The simplest approach is to pick up a page geometry box (i.e. one of MediaBox, CropBox, BleedBox, TrimBox or ArtBox). Further below you will find a discussion how to fine tune the use of such a page geometry box.

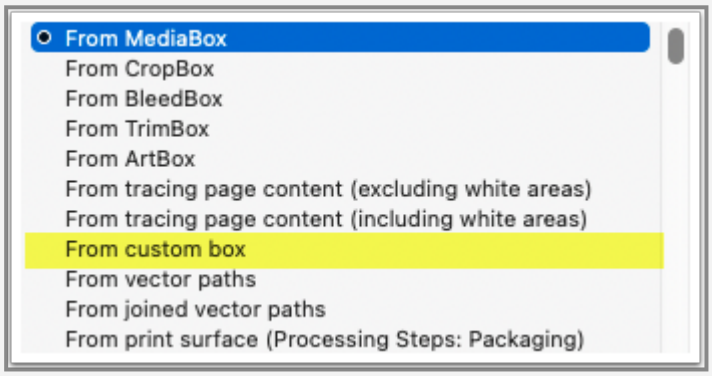
Discussion of the "Create shape: parameters: From tracing page content



A more advanced approach is to use the rendered appearance of page content and create an outline around it (and for any 'holes' inside it). Depending on whether white areas (as opposed to areas that just look white because they are actually transparent and let the white background shine through) are considered part of the rendered page content or not, it is necessary to choose between "From tracing page content (*including* white areas)" and "From tracing page content (*excluding* white areas)".

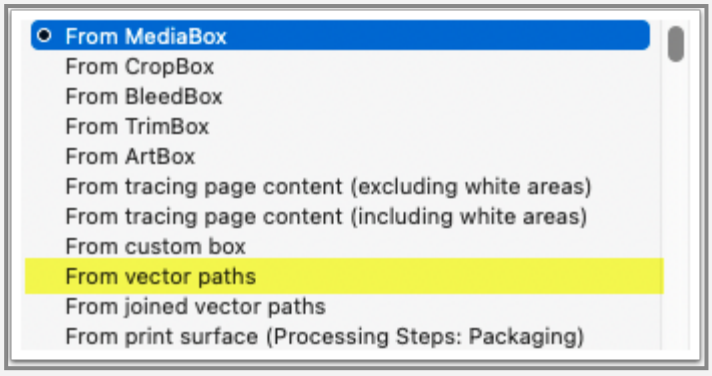
Please see below for a discussion of how to adjust parameters for tracing page content.

Discussion of the "Create shape: parameters: From custom box



This approach is similar to the "From MediaBox" etc. approaches but instead of picking up the existing page geometry box, it is necessary to provide the coordinates of the desired box and its position on the page. Please see below for a discussion of the necessary parameters.

Discussion of the "Create shape: parameters: From vector paths



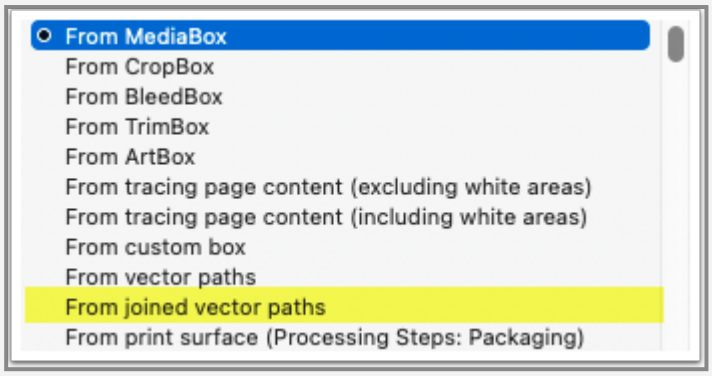
Another way to derive shapes from page content is to use "From vector paths". This will simply pick up existing path objects (and by implication it does not pick up images, image masks, font based text objects, soft masks, or smooth shades) and turn them into shapes.

Important:

Usually it does not make sense to pick up all path objects on the page, but rather only a small set of path objects, or even just exactly one path object – e.g. a dieline colored in a specific spot color.

Please see below for discussion of how to use parameters for this approach.

Discussion of the "Create shape: parameters: From joined vector paths



This option allows the creation of a closed vector path object from vector path objects with endpoints at the same position but not connected.

In certain cases, stroked vector objects appear as closed paths, but internally in the PDF structure, they consist of multiple open paths. This option works by:

1. Duplicating all relevant path objects.
2. Identifying endpoints in stroked path objects that are identical to other such endpoints.
3. Creating an internal copy and connecting the endpoints of open paths that are close to each other.
4. Merging the identified path objects from the internal copy to create one closed vector path.

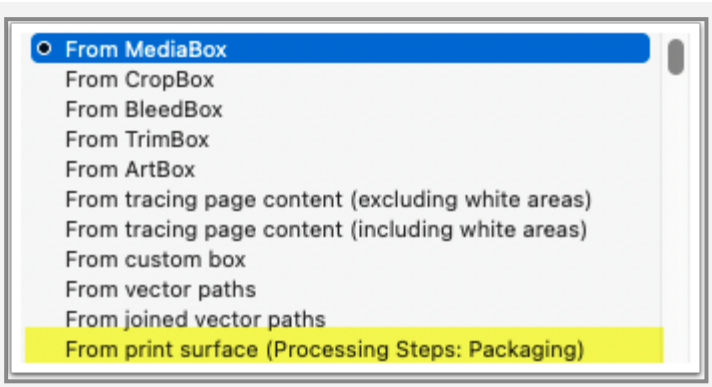
Known Limitations

The "From joined vector paths" option only takes open path objects into account and can only create proper results for stroked vector paths (not for lines that are actually filled vector paths).

You have to specify the exact kind of path object that should be joined for example, a dieline colored in a specific spot color. Please see below for a discussion of how to use parameters for this approach.

Logging information for "From joined vector paths" using Test Mode:
When testing a "Create and apply shapes" Fixup that uses the option "From joined vector paths" additional logging information is available in the Log Profile Execution folder: click here for more information.

Discussion of the "Create shape: parameters: From print surface (Processing Steps: Packaging)



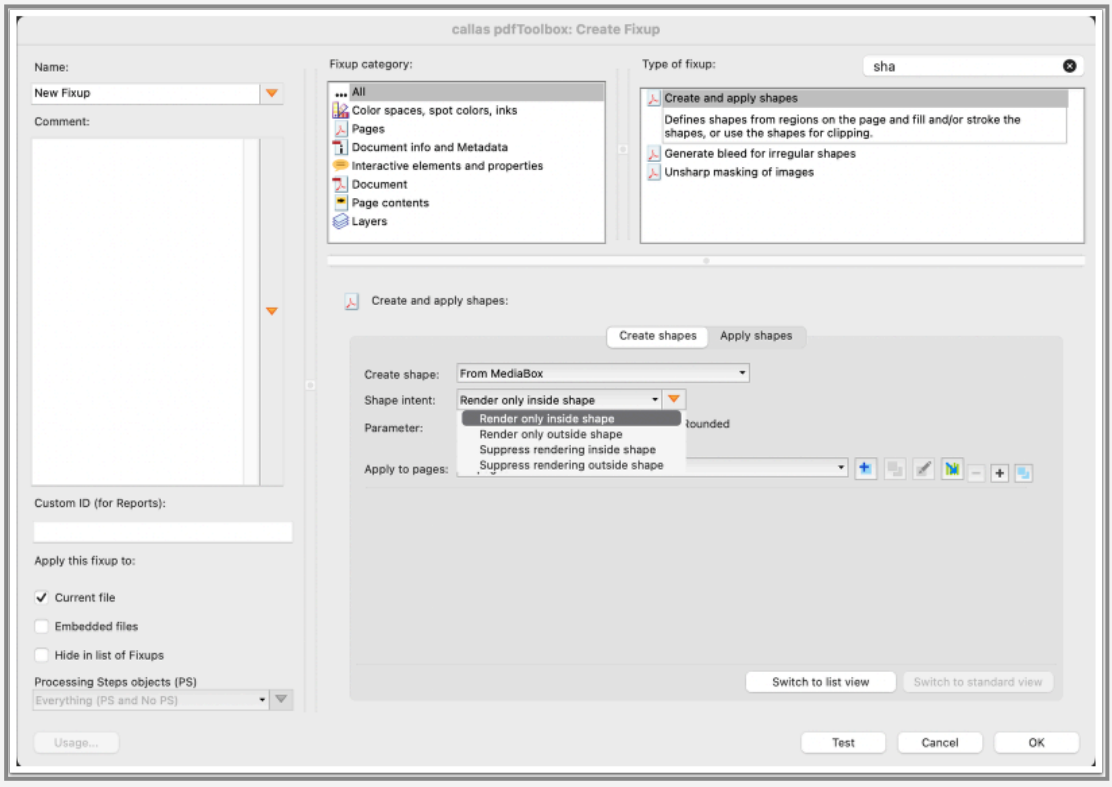
The last way to derive shapes from page content is to use "From print surface" using processing steps information.

"Print surface" is defined in the Processing Steps ISO standard as being the first possible of:

1. The area contained by all contours specifying the bleed area (i.e. within all Structural/Bleed processing step OCGs) on the PDF page) if present.
2. The area contained by all contours specifying the die line area (i.e. within all Structural/Cutting processing step OCGs on the PDF page) if present.
3. The BleedBox of the PDF Page, if present.
4. The TrimBox of the PDF Page, if present
5. The ArtBox of the PDF page, if present.

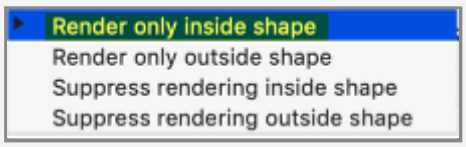
6. The CropBox of the PDF page, if present
7. The MediaBox of the PDF Page

"Shape intent" parameter



Shapes can be put to use in a number of ways. While aspects such as how to color the area inside a shape or how to stroke its contour are discussed in the next article about applying shapes, some control over the use of each of the individual shapes is already provided in this "Create shape" article.

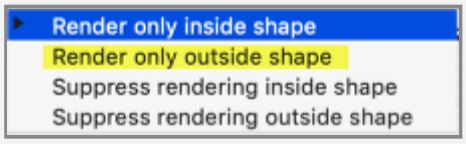
Discussion of the "Shape intent" parameter: Render only inside shape



Where shapes are to be used for filling the area inside them with a certain color, this would be the option to choose. It is

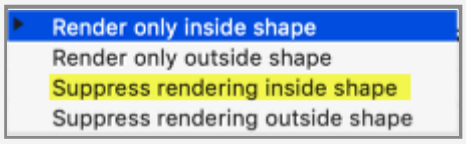
important though to understand that where shapes are inside one another, the even odd rule for painting applies – i.e. if a shape consists of two circles where one circle is completely inside the other circle, only the area between the contours of the two circles will be colored, the areas outside the outer circle and the area inside the inner circle would remain uncolored.

Discussion of the "Shape intent" parameter: Render only outside shape



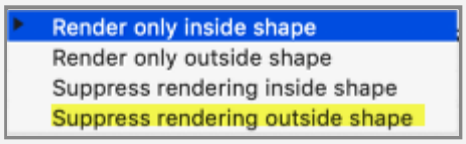
Where shapes are to be used for filling the outside of an area, but not the area inside it, with a certain color, this would be the option to choose. It is important though to understand that where shapes are inside one another, the even odd rule for painting applies. I.e. in this case if a shape consists of two circles where one circle is completely inside the other circle, the area between the contours of the two circles will not be colored, but the areas outside the outer circle and the area inside the inner circle would be colored.

Discussion of the "Shape intent" parameter: Suppress rendering inside shape



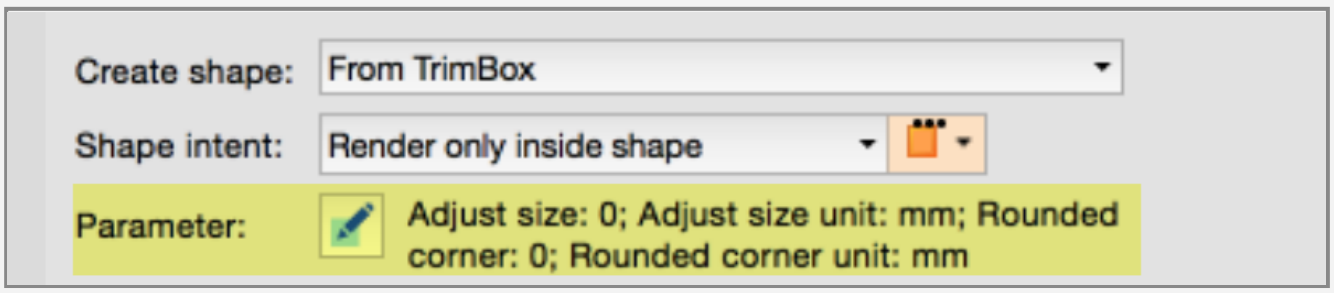
This shape intent ensures that regardless of any of the other shapes and their shape intent the inner area defined by this shape will remain uncolored. This will normally only make sense if this shape is combined with at least one other shape, that creates a colored (or stroked) area.

Discussion of the "Shape intent" parameter: Suppress rendering outside shape



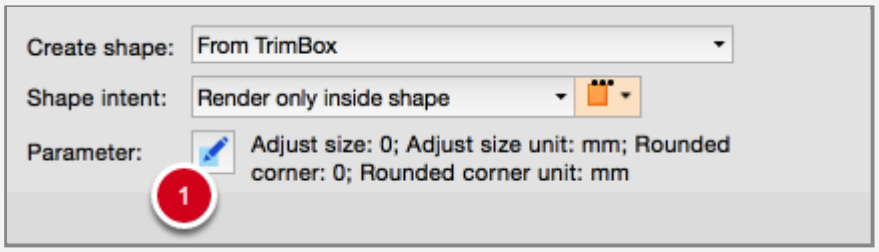
This shape intent ensures that regardless of any of the other shapes and their shape intent the outside area defined by this shape will remain uncolored. This will normally only make sense if this shape is combined with at least one other shape, that creates a colored (or stroked) area.

Additional parameters for defining shapes



The third setting "Parameter" in a shape configuration depends on the actual "Create shape" setting, and will differ substantially between them. The steps below explain the parameters for each of the "Create shape" settings.

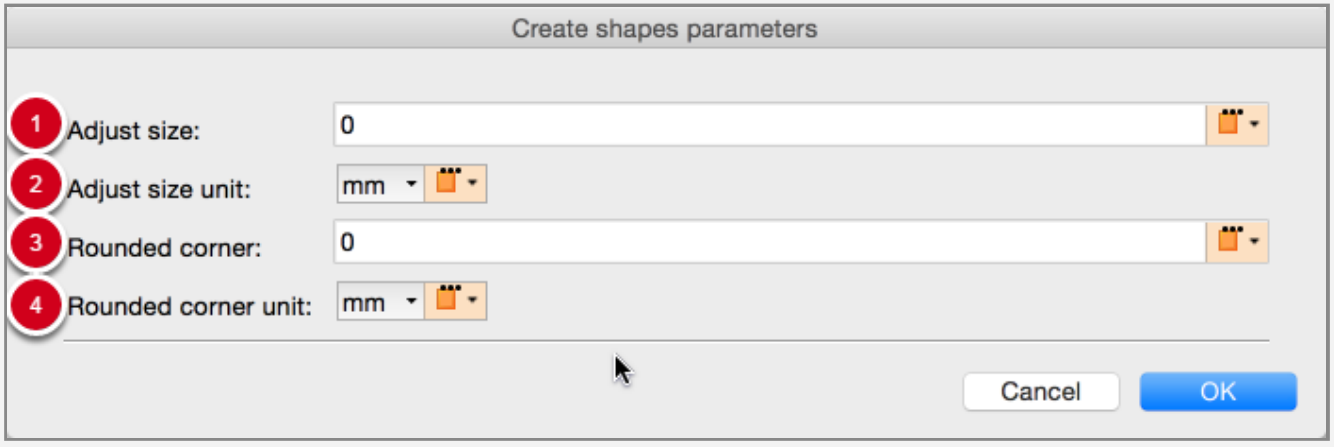
Shapes based on MediaBox, CropBox, BleedBox, TrimBox, ArtBox



When the "Create shape" setting is set to any of the page geometry boxes (in this example it is set to "From TrimBox"), the current "Parameter" values are reported.

Clicking on the "Edit" button, the "Create shape parameters" dialog will open, offering the applicable parameters for editing.

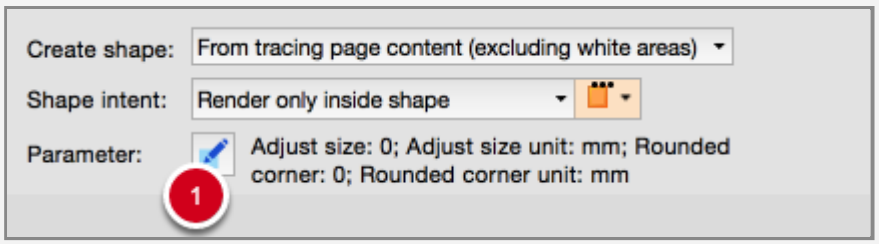
Parameters for shapes based on MediaBox, CropBox, BleedBox, TrimBox, ArtBox



1. Adjust size makes it possible to subtract from (negative numbers) or to add (positive numbers) to the page geometry box. For example, entering "-5" will make the resulting rectangular shape smaller by 5 units on each side of the rectangle.
2. "Adjust size unit" makes it possible to pick between point, millimeter and inch.

3. Where it is desired to create a rectangular shape with rounded corners, a number greater than zero needs to be entered in this field. The number defines the radius of the rounded corners.
4. "Rounded corner unit" makes it possible to pick between point, millimeter and inch.

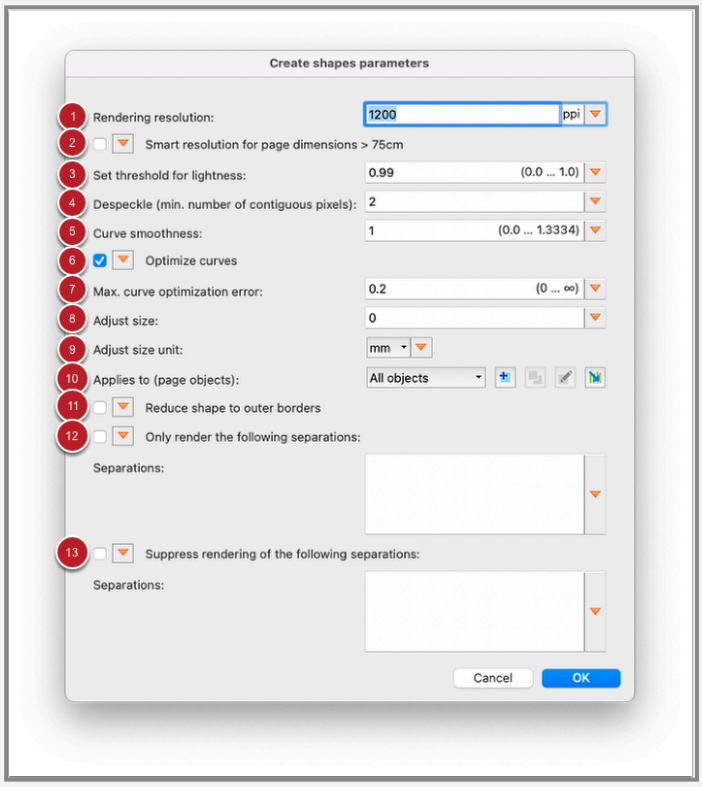
Shapes based on tracing page content (including or excluding white areas)



When the "Create shape" setting is set to one of the two "From tracing page content" options (in this example it is set to "From tracing page content (excluding white areas)"), the current "Parameter" values are reported.

Clicking on the "Edit" button (1), the "Create shape parameters" dialog will open, offering the applicable parameters for editing.

Parameters for shapes based on tracing page content (including or excluding white areas)



The parameters in this dialog provide some control over the rendering of the page for tracing purposes, and for the tracing as such.

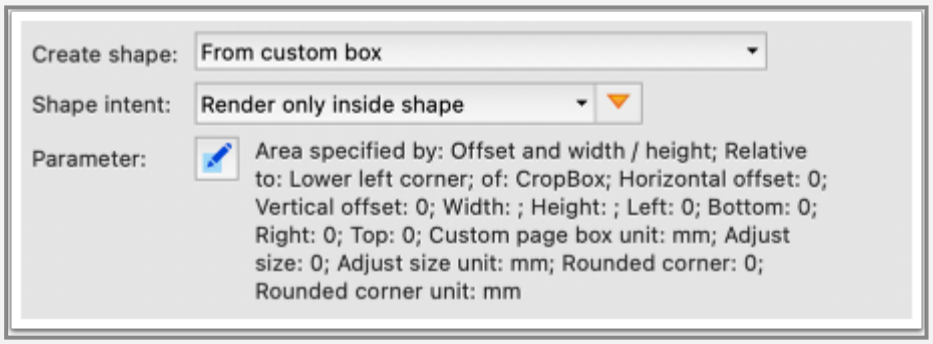
Important: The default settings for "Despeckle", "Curve smoothness" and "Max. curve optimization error" are the result of extensive research, and are considered to be highly suitable for most use cases. Only in the rare case, where there is reason to assume modified values for these parameters will provide more suitable results, should the values be modified.

1. Resolution of the (internally rendered) page image
   Value between 36 and 3200, default: 1200; Recommended values: 300 - 1200 ppi
2. Enables auto adjustment of the resolution for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.
3. Set threshold for lightness: Value between 0.0 to 1
   0.0: fully opaque, partially transparent objects are ig-

nored;
1.0: fully transparent, amounts to the whole page area;
Recommended max. value: 0.99, capture all objects, but not areas where no object is rendered.

4. Apply despeckling no more than for this number of pixels that are of the same color and are surrounded on all sides by pixels of another color.
5. Determines the "curve smoothness" to achieved.
   A value of zero would lead to a tracing result where the sides of each pixel of the rendered image are followed precisely, leading to a jaggy tracing result (though at 1200 ppi) the jagginess might not be readily noticeable – nonetheless, this lack of smoothness will increase time to process, and will not add any actual quality to the tracing result.
   0.0: polygons only, no curves;
   1.3334: curves only, no corners;
   Recommended value: 1.0
6. Makes it possible to enable or disable curve optimization (see 6.)
7. Maximum delta allowed between a pixel perfect tracing result and the result of curve optimization. A value 0.2 will keep the optimization error at a minimum that will hardly ever be noticeable while still allowing for creation of efficient curves. Values greater than 1 are typically not useful.
8. Adjust size
9. Adjust size unit
10. Makes it possible to create a rendered page only based on the objects specified in this filter. For example, if the intention is to create a shape over all text objects of spot color "Orange", but not over the rest of the page content, a filter "Is text object using spot color 'Orange'" could be configured and selected, and tracing would happen based on a rendering of the page where the page (temporarily) only shows orange text.
11. By default the shape is created at the inner and outer boarders of the page content. If this checkbox is checked the shape will only be created at the outer boarders of the page content.
12. Makes it possible to create a rendered page based on specific separations.
13. Makes it possible to suppress rendering for specific separations.

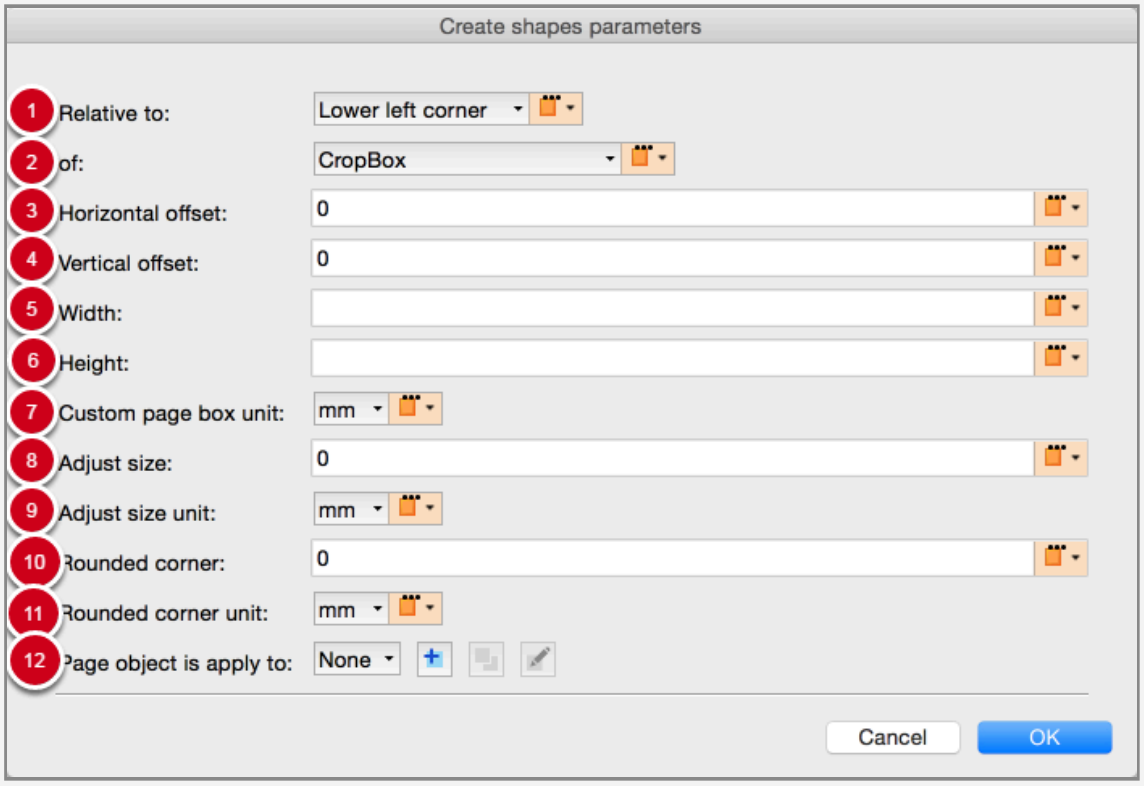
Shapes based on a custom defined box



When the "Create shape" setting is set to "From custom box", the current "Parameter" values are reported.

Clicking on the "Edit" button, the "Create shape parameters" dialog will open, offering the applicable parameters for editing.

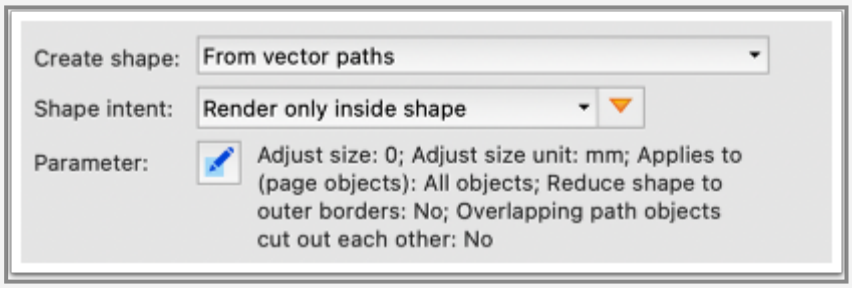
Parameters for shapes based on a custom defined box



The parameters for "From custom box" are a combination of the parameters for page geometry boxes, and the parameters needed to describe the size and position of the custom page on the page area:

1. Determines the reference point (relative to the reference rectangle under 2.) starting from which the custom box shall be positioned
2. Determines the reference rectangle from which the custom box shall be positioned
3. Horizontal offset from the reference point where the respective corner of the custom box shall be (e.g. for "Lower right corner" of "CropBox" a positive value would move the lower right corner of the custom box to the right)
4. Vertical offset from the reference point where the respective corner of the custom box shall be (e.g. for "Lower right corner" of "CropBox" a positive value would move the lower right corner of the custom box downwards)
5. Width of the custom box
6. Height of the custom box
7. Measurement unit for the values entered in fields 3. through 6.
8. Adjust size makes it possible to subtract from (negative numbers) or to add (positive numbers) to the page geometry box. For example, entering "-5" will make the resulting rectangular shape smaller by 5 units on each side of the rectangle.
9. "Adjust size unit" makes it possible to pick between point, millimeter and inch.
10. Where it is desired to create a rectangular shape with rounded corners, a number greater than zero needs to be entered in this field. The number defines the radius of the rounded corners.
11. "Rounded corner unit" makes it possible to pick between point, millimeter and inch.
12. Makes it possible to limit the creation (and application) of this shape only t pages where the applicable filter applies. For example, if the filter is set to find objects using a spot color "Lime Green", only for pages that contain at least one object using that spot color will trigger the creation of this custom box.

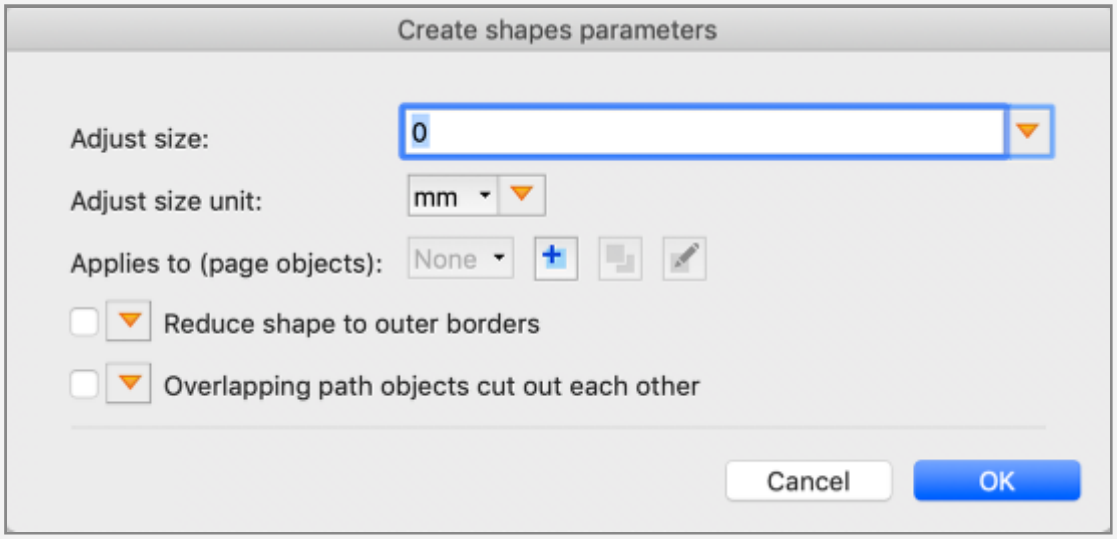
Shapes based on existing vector paths



When the "Create shape" setting is set to "From vector paths", the current "Parameter" values are reported.

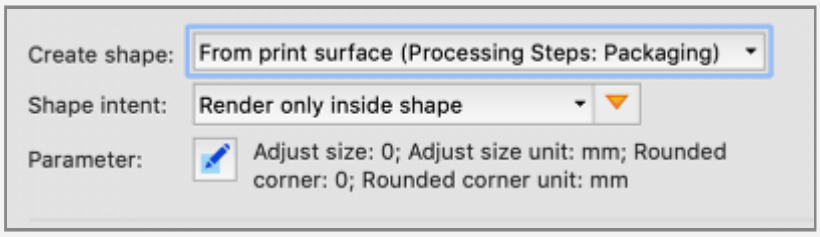
Clicking on the "Edit" button, the "Create shape parameters" dialog will open, offering the applicable parameters for editing.

Parameters for shapes based on existing vector paths



For creation of shapes "From vector paths", a couple of parameters can be configured: a filter determining which of the vector objects on the respective page to use and adjusting the size.

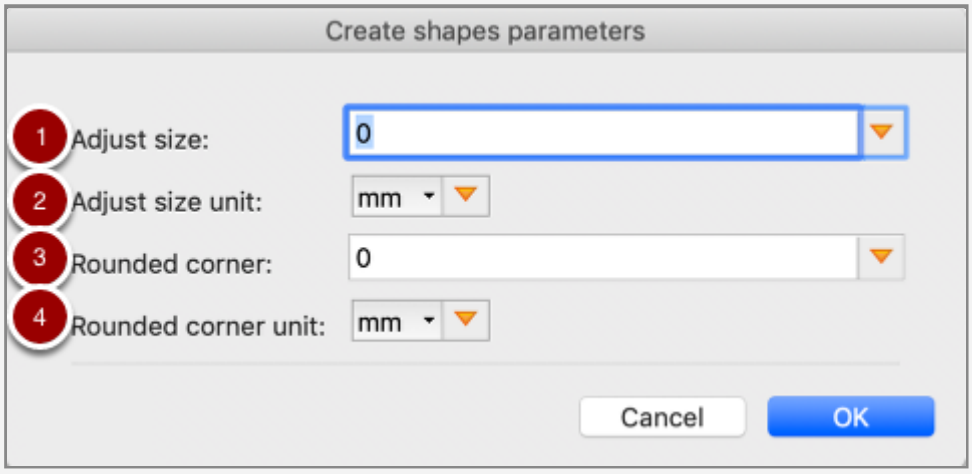
Shapes based on print surface



When the "Create shape" setting is set to "From print surface", the current "Parameter" values are reported.

Clicking on the "Edit" button, the "Create shape parameters" dialog will open, offering the applicable parameters for editing.

Parameters for shapes based on print surface



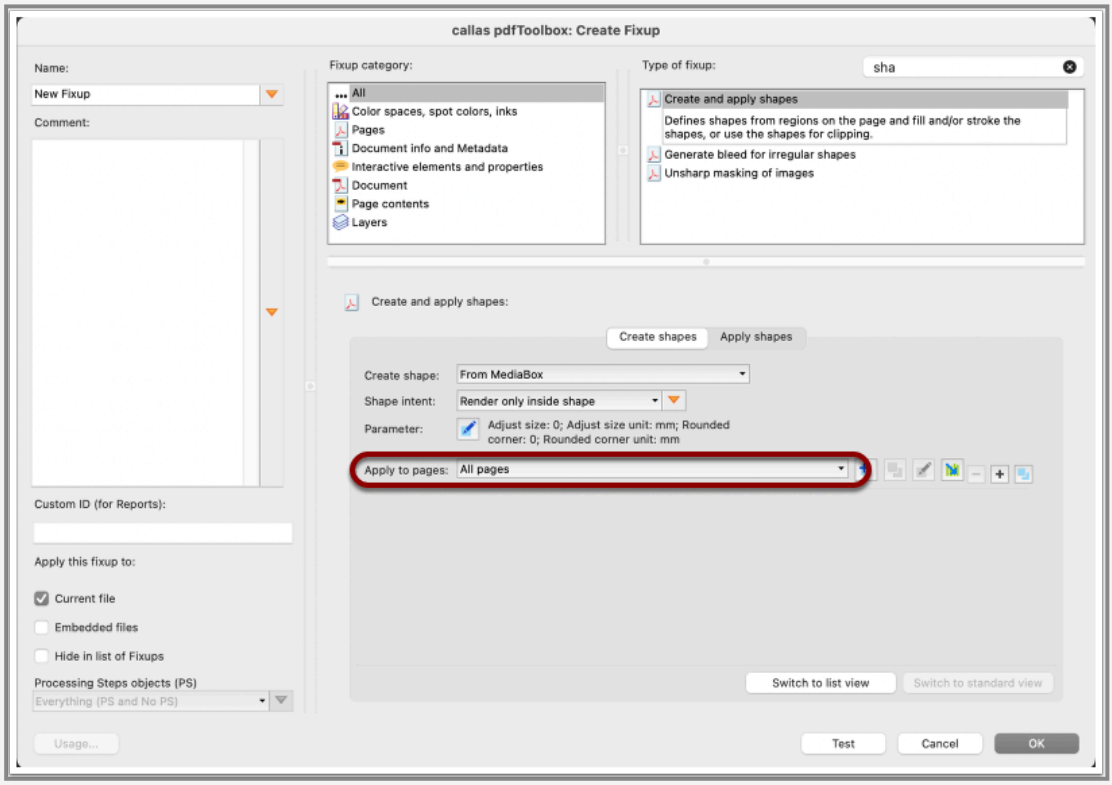
For creation of shapes "From print surface", a couple of parameters can be configured:

1. Adjust size makes it possible to subtract from (negative numbers) or to add (positive numbers) to the page geometry box. For example, entering "-5" will make the resulting rectangular shape smaller by 5 units on each side of the rectangle.
2. "Adjust size unit" makes it possible to pick between point, millimeter and inch.

3. Where it is desired to create a rectangular shape with rounded corners, a number greater than zero needs to be entered in this field. The number defines the radius of the rounded corners.
4. "Rounded corner unit" makes it possible to pick between point, millimeter and inch.

Select pages for which shapes are to be created

Beginning with pdfToolbox 14 it is possible to specify a page filter (based on a Check), so that shapes will only be created on selected pages.

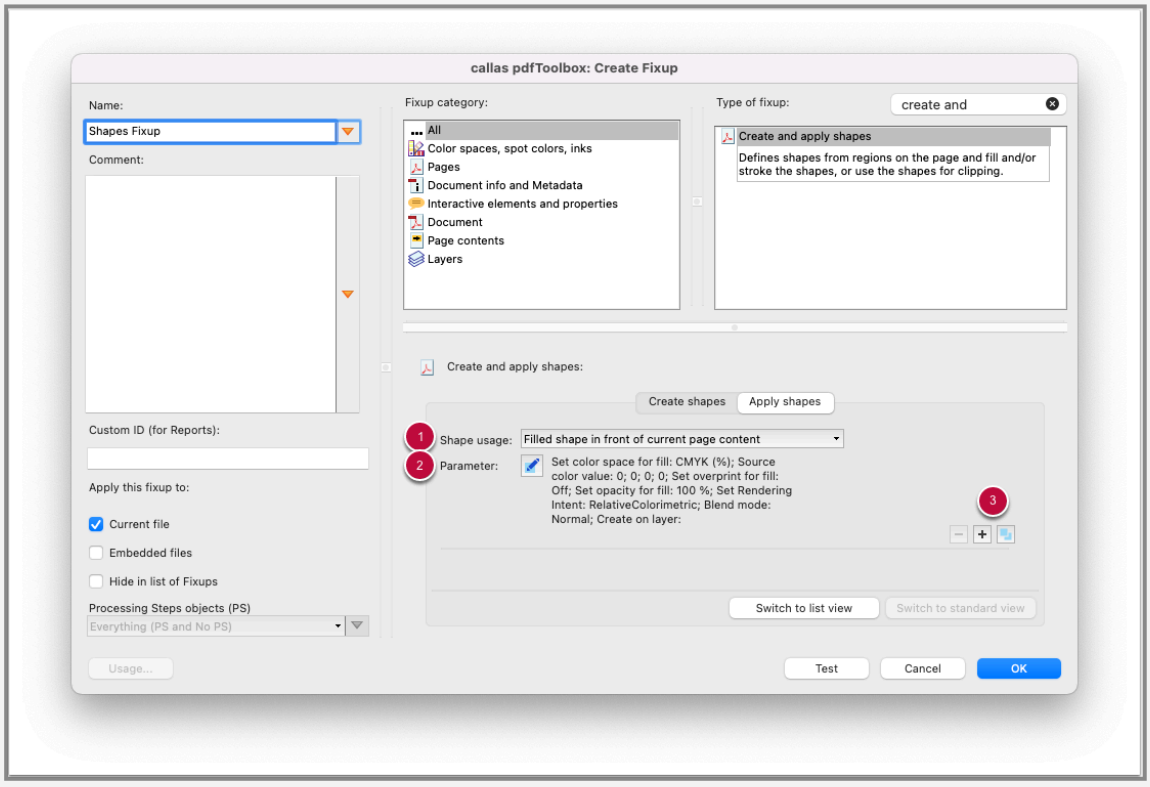


21.3 Applying shapes

The "Create and apply shapes" Fixup consists of two configuration sections. First, one or more shapes need to be defined under "Create shapes", next an action has to be defined under "Apply shapes" that determines how to make use of these shapes.

This article describes the configuration of the "Apply shapes" section.

"Apply shapes" parameters

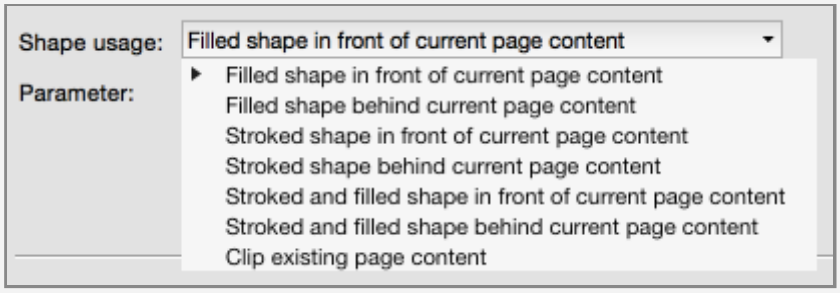


1. The popup menu under "Shape usage" determines what action to apply to to the shapes created in the "Create shapes" section.
2. Depending on the selected "Shape usage", different sets of parameters become available under "Parameter"

3. The "+" (plus) button to the right makes it possible to define more than one action on the defined shapes. The "-" (minus) button makes it possible to remove such an action. It is only enabled if there are at least two actions in the list of "Apply shapes" actions. Since pdfToolbox 14 it is also possible to duplicate the respective set of configured parameters with the “Duplicate setting” button.

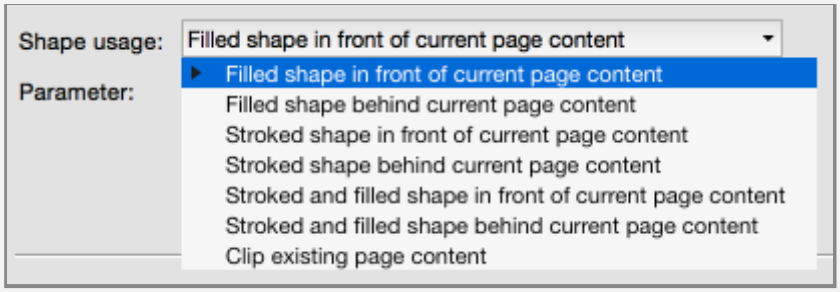
The following steps describe both the "Shape usage" options and the parameters that come with each these options.

"Apply shape": List of settings



This screenshot lists the currently available"Shape usage" options.

Filling shapes in front or behind existing page content

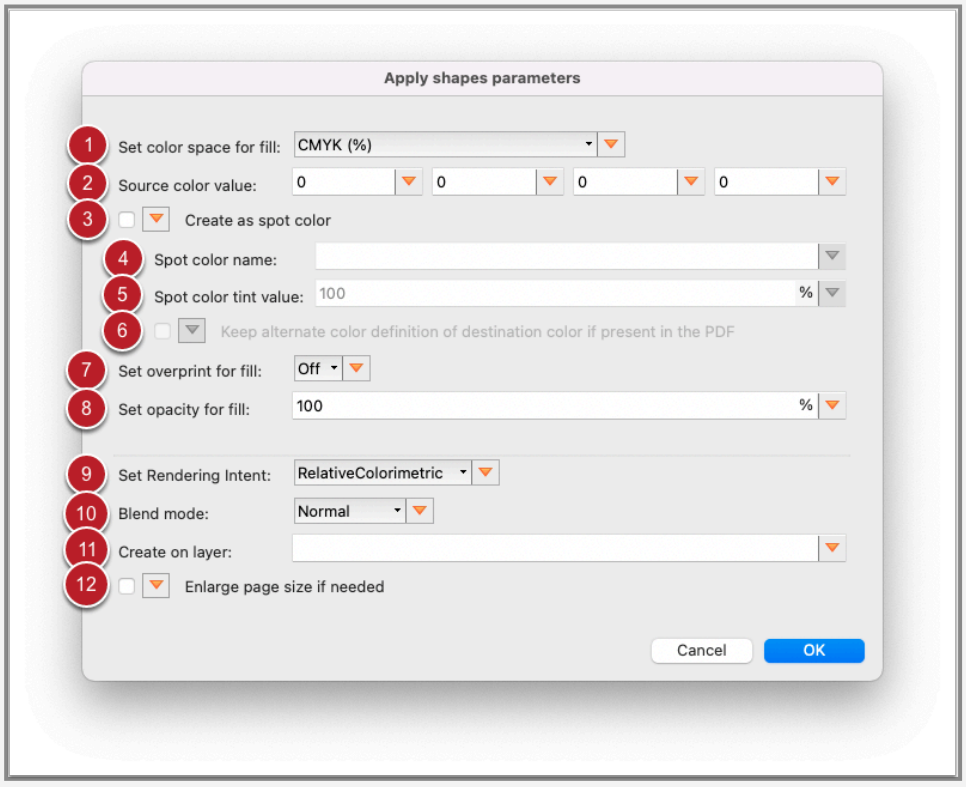


Important note:
The description below applies in the exact same fashion to filled shapes created **behind** current page content. The option to create a filled shape behind current page content **will only work as expected in very few cases**, as any current page content that is opaque will hide the filled shape created behind it. If for example the whole page has an opaque white

background object, or any other opaque object that fills the whole page or most of it, the created filled shape will most probably not be visible at all.

The creation of a "Filled shape in front of current page content" means that the defined shape(s) will be inserted into each page as a path object. If the shape consists of severals part, it will be created as a path object with as many closed sub-paths. For this path object a fill will be applied as configured under "Parameters".

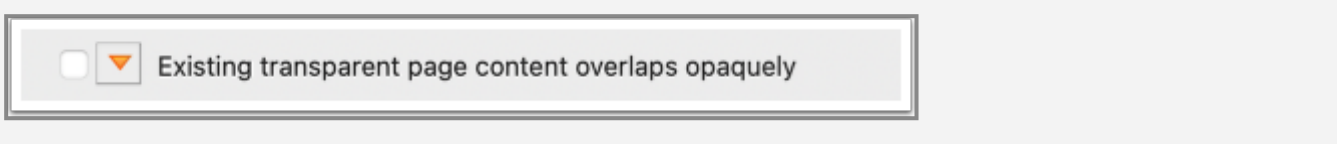
Filling shapes in front or behind existing page content – Fill parameters



1. Color model to be used for filling the path object; available models are CMYK, RGB, gray, and Lab. For CMYK, RGB and gray, values can be set to be provided in percent (0...100%) or as a number (0.0...1.0). Lab values must always be provided as 0...100 for the L value, and -127...128 for the a and b values.

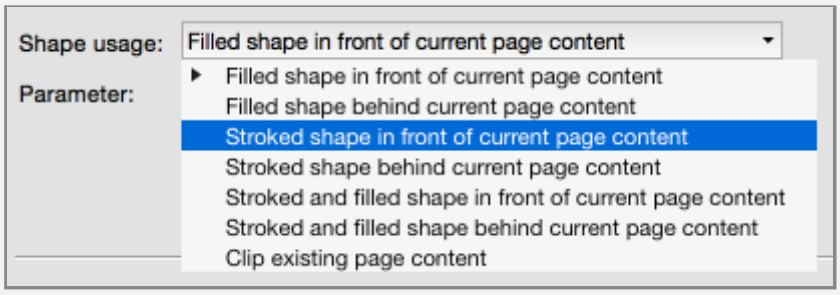
2. Depending on the chosen color space, one, three or four values have to be provided
3. If this check box is activated, the color used for the fill be create as a spot color, the color model will then be used as the alternate color space for the spot color, and the color values will determine the appearance of a 100% tint value of the spot color
4. Name of the spot color (only enabled if checkbox "Create as spot color" is activated)
5. Tint value to use for filling the path with the spot color (only enabled if checkbox "Create as spot color" is activated)
6. If a spot color already exists in the document, the alternate color values of this existing spot color are used instead of the color values specified above when this check box is activated.
7. If this checkbox is activated, the fill color will be set to overprint.
8. Sets the opacity for the filled path object, i.e. the degree to which the path object will be transparent or not. A value of 100% means that the object is opaque (not transparent at all), and a value of 0% means the object is fully transparent (which implies that it will not be visible at all).
9. Determines the rendering intent. This will only become relevant when a color conversion is applied at a later stage.
10. Set the transparency blend mode. All 16 blend modes defined in the PDF imaging model are available.
11. If not empty, determines that path object is created on a separate layer, named according to the value in this entry.
12. If this check box is activated, the page is automatically enlarged if the shape is larger than the actual page size.

For shapes behind existing page content there is an additional option:



This allows for avoiding transparency interactions between the existing page content and the new shape behind.

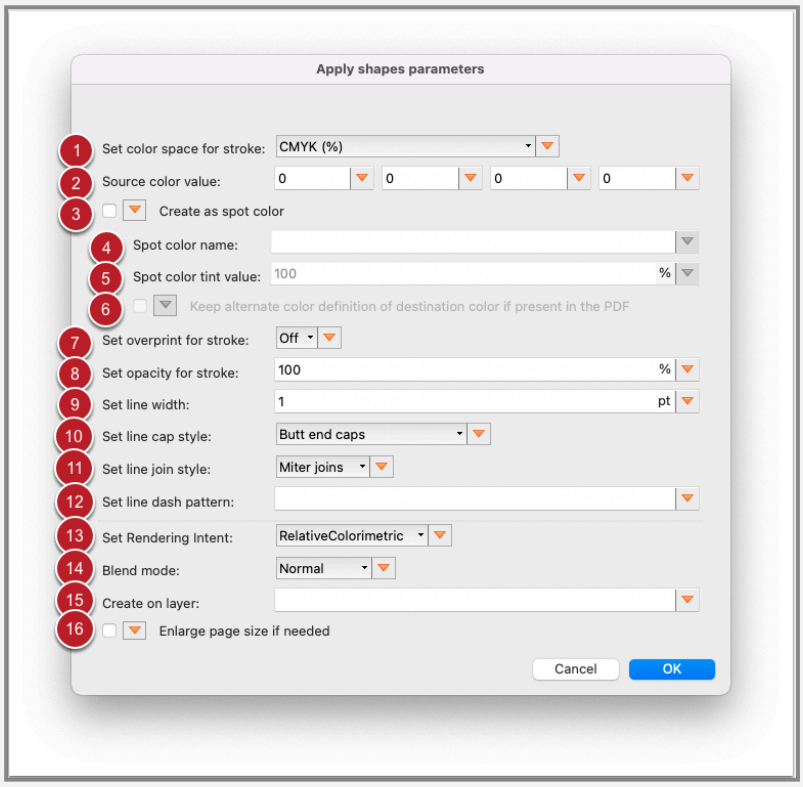
Stroking shapes in front or behind existing page content



Important note:
The description below applies in the exact same fashion to stroked shapes created **behind** current page content. The option to create a stroked shape behind current page content **will only work as expected in very few cases**, as any current page content that is opaque will hide the stroked shape created behind it. If for example the whole page has an opaque white background object, or any other opaque object that fills the whole page or most of it, the created stroked shape will most probably not be visible at all.

The creation of a "Stroked shape in front of current page content" means that the defined shape(s) will be inserted into each page as a path object. If the shape consists of severals part, it will be created as a path object with as many closed sub-paths. For this path object a stroke (also called contour or outline) will be applied as configured under "Parameters".

Stroking shapes in front or behind existing page content - Stroke parameters



1. Color model to be used for stroking the path object; available models are CMYK, RGB, gray, and Lab. For CMYK, RGB and gray, values can be set to be provided in percent (0...100%) or as a number (0.0...1.0). Lab values must always be provided as 0...100 for the L value, and -127...128 for the a and b values.
2. Depending on the chosen color space, one, three or four values have to be provided
3. If this check box is activated, the color used for the stroke be create as a spot color, the color model will then be used as the alternate color space for the spot color, and the color values will determine the appearance of a 100% tint value of the spot color
4. Name of the spot color (only enabled if checkbox "Create as spot color" is activated)
5. Tint value to use for stroking the path with the spot color (only enabled if checkbox "Create as spot color" is activated)

6. If a spot color already exists in the document, the alternate color values of this existing spot color are used instead of the color values specified above when this check box is activated.
7. If this checkbox is activated, the stroke color will be set to overprint.
8. Sets the opacity for the stroked path object, i.e. the degree to which the path object will be transparent or not. A value of 100% means that the object is opaque (not transparent at all), and a value of 0% means the object is fully transparent (which implies that it will not be visible at all).
9. Sets the line width of the stroke in pt.
10. Sets the line cap style for the stroke. This will only have an effect if the stroke is created as a dashed line (see parameter 11). In order to create a dotted line, a suitable line dash pattern needs to be defined, where the part of the dash being painted must have the same length as the line is wide, and the line style must be defined as "Round cap".
11. Sets the line join style This will determine the shape of the line in the corners. Miter joins are ideal for orthogonal corners (e.g. in a rectangle), but can lead to very long pointed corners for corners that are of a sharp angle (this will often be perceived as strange artifacts). For sharp angles it is better to use Bevel join or Round join.
12. Using PDF syntax for dashed lines, this makes it possible to create dashed or dotted lines in many variations. The syntax consists of a sequence of numbers inside one pair of square brackets. Each number determines the length of a line segment that is painted or that is a gap. The first number inside the pair of brackets always defines the length of a painted segment, the second number defines the length of a gap. The third number, defines the length of a painted segment, and so on. Once the sequence of numbers inside the square brackets have been used up the sequence will start again at the beginning. A simple example would be [3 2] which would lead to a line of painted segments three units long, and gaps between painetd segments of a length of two units. In order to create an evenly dotted line of 2pt width, use 2 pt for the width of the line, a painted segment length of 0 [sic!] and a gap of 4 units (i.e. a line dash parameter of "[0 4]"), and line cap style of "Round caps", The he line join style is irrelevant in this scenario

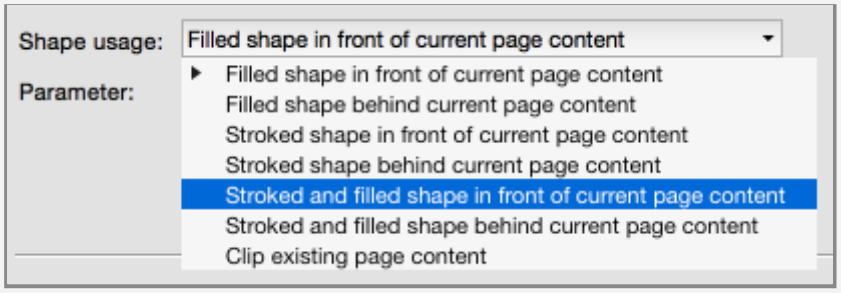
13. Determines the rendering intent. This will only become relevant when a color conversion is applied at a later stage.
14. Set the transparency blend mode. All 16 blend modes defined in the PDF imaging model are available.
15. If not empty, determines that path object is created on a separate layer, named according to the value in this entry.
16. If this check box is activated, the page is automatically enlarged if the shape is larger than the actual page size.

For shapes behind existing page content there is an additional option:



This allows for avoiding transparency interactions between the existing page content and the new shape behind.

Stroking and filling shapes at the same time in front or behind existing page content



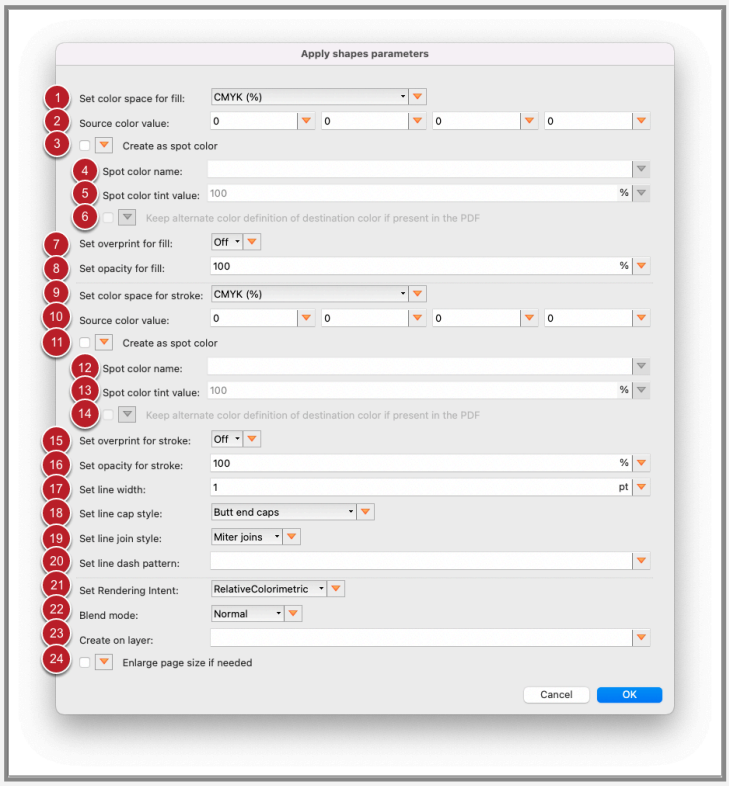
Important note:
The description below applies in the exact same fashion to stroked and filled shapes created **behind** current page content. The option to create a stroked and filled shape behind current page content **will only work as expected in very few cases**, as any current page content that is opaque will hide the stroked and filled shape created behind it. If for example the whole page has an opaque white background object, or any other opaque object that fills the whole page or most of it, the created stroked and filled shape will most probably not be visible at all.

The creation of a "Stroked and filled shape in front of current page content" means that the defined shape(s) will be inserted into each page as a path object. If the shape consists of severals part, it will be created as a path object with as many closed sub-paths. For this path object a stroke (also called contour or outline) and a fill will be applied as configured under "Parameters". The color for the stroke and the fill can be configured to be different.

According to the PDF imaging model, the fill will always be painted first, followed by the stroke. This implies that the line is guaranteed to always be shown with its indicated line width, half of which will be rendered towards the inside of the path segments, overlaying the fill, with the other half of it being rendered outside of the path segments.

A specific use of this is to define both fill and stroke using the same color definition (preferably a spot color for this to work well), and to use a tint value of 100% for the fill and of 0% for the stroke. If both are set to overprint, hey will not affect any content underneath, and the resulting rendered path object will appear to be half of the line width smaller. This can be very useful for example for creating a white background that is slightly smaller than the shape from which it is generated.

Stroking and filling shapes at the same time in front or behind existing page content - Stroke and fill parameters



1. Color model to be used for filling the path object; available models are CMYK, RGB, gray, and Lab. For CMYK, RGB and gray, values can be set to be provided in percent (0...100%) or as a number (0.0...1.0). Lab values must always be provided as 0...100 for the L value, and -127...128 for the a and b values.
2. Depending on the chosen color space, one, three or four values have to be provided
3. If this check box is activated, the color used for the fill be create as a spot color, the color model will then be used as the alternate color space for the spot color, and the color values will determine the appearance of a 100% tint value of the spot color
4. Name of the spot color (only enabled if checkbox "Create as spot color" is activated)

5. Tint value to use for filling the path with the spot color (only enabled if checkbox "Create as spot color" is activated)
6. If a spot color already exists in the document, the alternate color values of this existing spot color are used instead of the color values specified above when this check box is activated.
7. If this checkbox is activated, the fill color will be set to overprint.
8. Sets the opacity for the filled path object, i.e. the degree to which the path object's fill will be transparent or not. A value of 100% means that the object is opaque (not transparent at all), and a value of 0% means the object is fully transparent (which implies that it will not be visible at all).
9. Color model to be used for stroking the path object; available models are CMYK, RGB, gray, and Lab. For CMYK, RGB and gray, values can be set to be provided in percent (0...100%) or as a number (0.0...1.0). Lab values must always be provided as 0...100 for the L value, and -127...128 for the a and b values.
10. Depending on the chosen color space, one, three or four values have to be provided
11. If this check box is activated, the color used for the stroke be created as a spot color, the color model will then be used as the alternate color space for the spot color, and the color values will determine the appearance of a 100% tint value of the spot color
12. Name of the spot color (only enabled if checkbox "Create as spot color" is activated)
13. Tint value to use for stroking the path with the spot color (only enabled if checkbox "Create as spot color" is activated)
14. TBD
15. If this checkbox is activated, the stroke color will be set to overprint.
16. Sets the opacity for the stroked path object, i.e. the degree to which the path object's stroke will be transparent or not. A value of 100% means that the object's stroke is opaque (not transparent at all), and a value of 0% means the object's stroke is fully transparent (which implies that it will not be visible at all).
17. Sets the line width of the stroke in pt.
18. Sets the line cap style for the stroke. This will only have an effect if the stroke is created as a dashed line (see pa-

rameter 11). In order to create a dotted line, a suitable line dash pattern needs to be defined, where the part of the dash being painted must have the same length as the line is wide, and the line style must be defined as "Round cap".

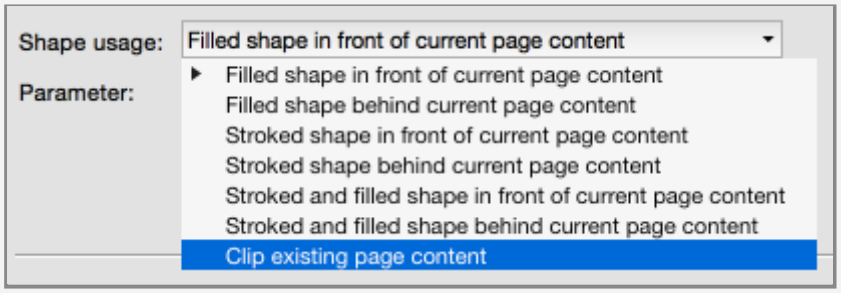
19. Sets the line join style This will determine the shape of the line in the corners. Miter joins are ideal for orthogonal corners (e.g. in a rectangle), but can lead to very long pointed corners for corners that are of a sharp angle (this will often be perceived as strange artifacts). For sharp angles it is better to use Bevel join or Round join.
20. Using PDF syntax for dashed lines, this makes it possible to create dashed or dotted lines in many variations. The syntax consists of a sequence of numbers inside one pair of square brackets. Each number determines the length of a line segment that is painted or that is a gap. The first number inside the pair of brackets always defines the length of a painted segment, the second number defines the length of a gap. The third number, defines the length of a painted segment, and so on. Once the sequence of numbers inside the square brackets have been used up the sequence will start again at the beginning. A simple example would be [3 2] which would lead to a line of painted segments two units long, and gaps between painetd segments of a length of 2 units. In order to create an evenly dotted line of 2pt width, use 2 pt for the width of the line, a painted segment length of 0 [sic!] and a gap of 4 units (i.e. a line dash parameter of "[0 4]"), and line cap style of "Round caps", The he line join style is irrelevant in this scenario
21. Determines the rendering intent. This will only become relevant when a color conversion is applied at a later stage.
22. Set the transparency blend mode. All 16 blend modes defined in the PDF imaging model are available.
23. If not empty, determines that path object is created on a separate layer, named according to the value in this entry.
24. If this check box is activated, the page is automatically enlarged if the shape is larger than the actual page size.

For shapes behind existing page content there is an additional option:



This allows for avoiding transparency interactions between the existing page content and the new shape behind.

Use shape as clipping path for existing page content



This "Shape usage' is a special one that simply uses the defined shape as a clipping path. Depending on whether the shape intent has defined as being "Render only **inside** shape" or "Render only **outside** shape" the clipping path will clip the page content **inside** the path object or the page content **outside** the path object.

There are no configurable parameters for this shape usage.

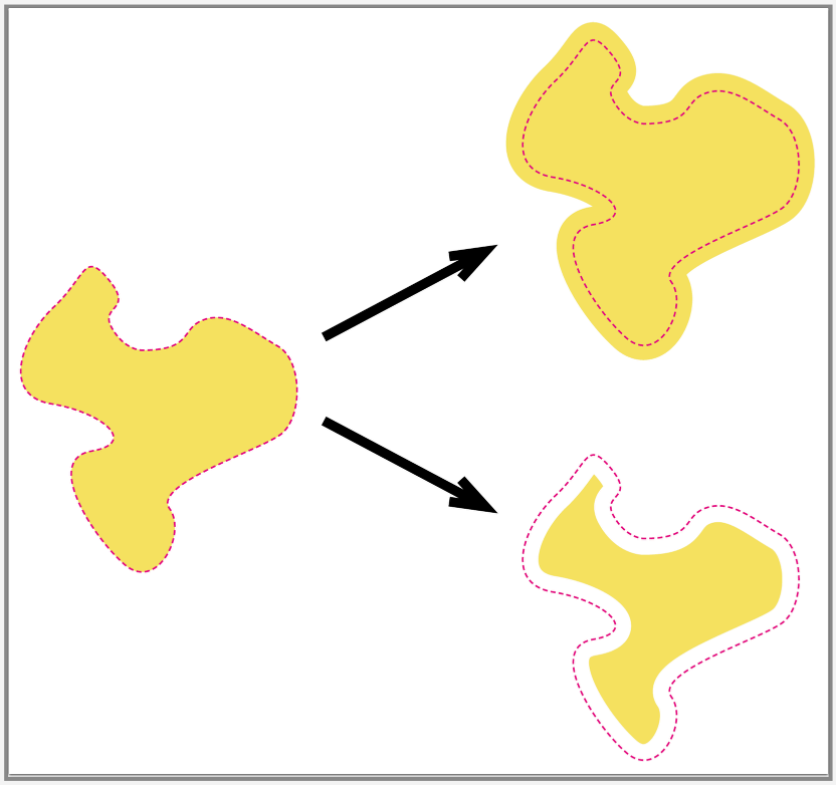
21.4 Extended "Shapes" features

Based on numerous feature requests, the "Shapes" feature, introduced in pdfToolbox 9.0, has already been extended in pdfToolbox 9.1. There are two areas that have been enhanced or added:

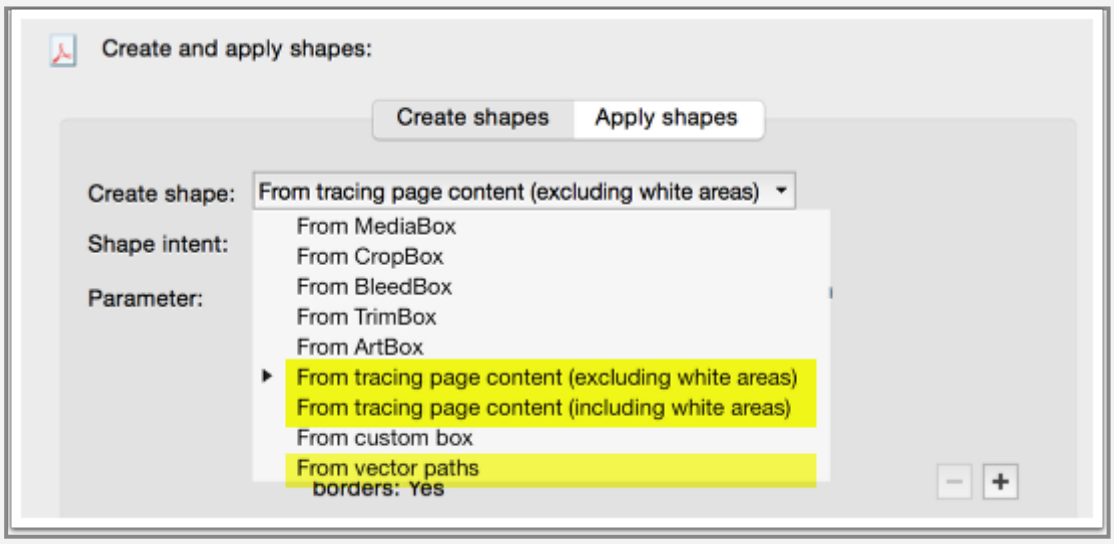
- **enlarging or reducing a shape** by a certain amount now also is available for shapes derived from tracing page content or using existing vector paths (until now, enlarging/reducing was only supported for rectangular shapes based on page geometry boxes or a custom box)
- **"all-inclusive" shape**: it is now possible to merge several nested shapes such that only the outer border of such shapes will be used. For example, for a donut shape, it is sometimes necessary to use the donut shape with the hole in it, and in other cases it is necessary to just use its outer border, i.e. the outer circle, and not take the hole in it into account

Enlarging or reducing non-rectangular shapes

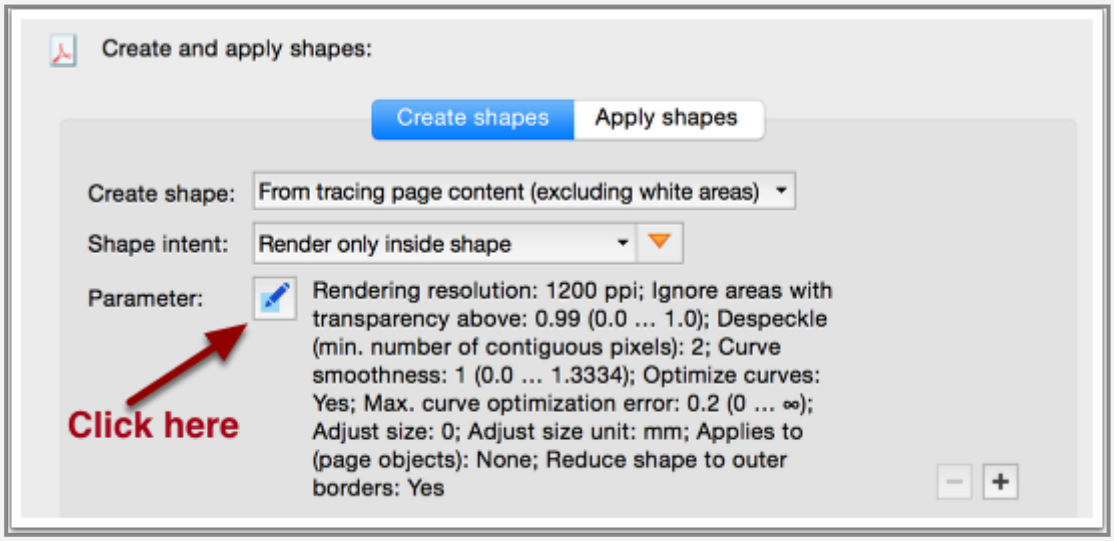
In many cases a shape cannot be directly used as derived from page content. Instead, it needs to be reduced in size by a small amount, or enlarged.
For example, for a white background in label printing on transparent substrate, it could be necessary to print white in all areas of the printed content, but at the same time the white should never become visible itself, e.g. in the case of mis-registration of the colorants during the printing process. Thus it can make sense to reduce the area where white is to be a printed by a millimeter or so.
Along the same lines, it can be desirable for varnish to make sure it gets printed in top of all printed page content – and a tiny bit beyond it, to again compensate for possible mis-registration between colorants during the printing process. In this case, the shape would be enlarged by a millimeter or so.



The option to enlarge or reduce a shape's size is now also available for shapes derived from tracing page content or from existing vector paths:



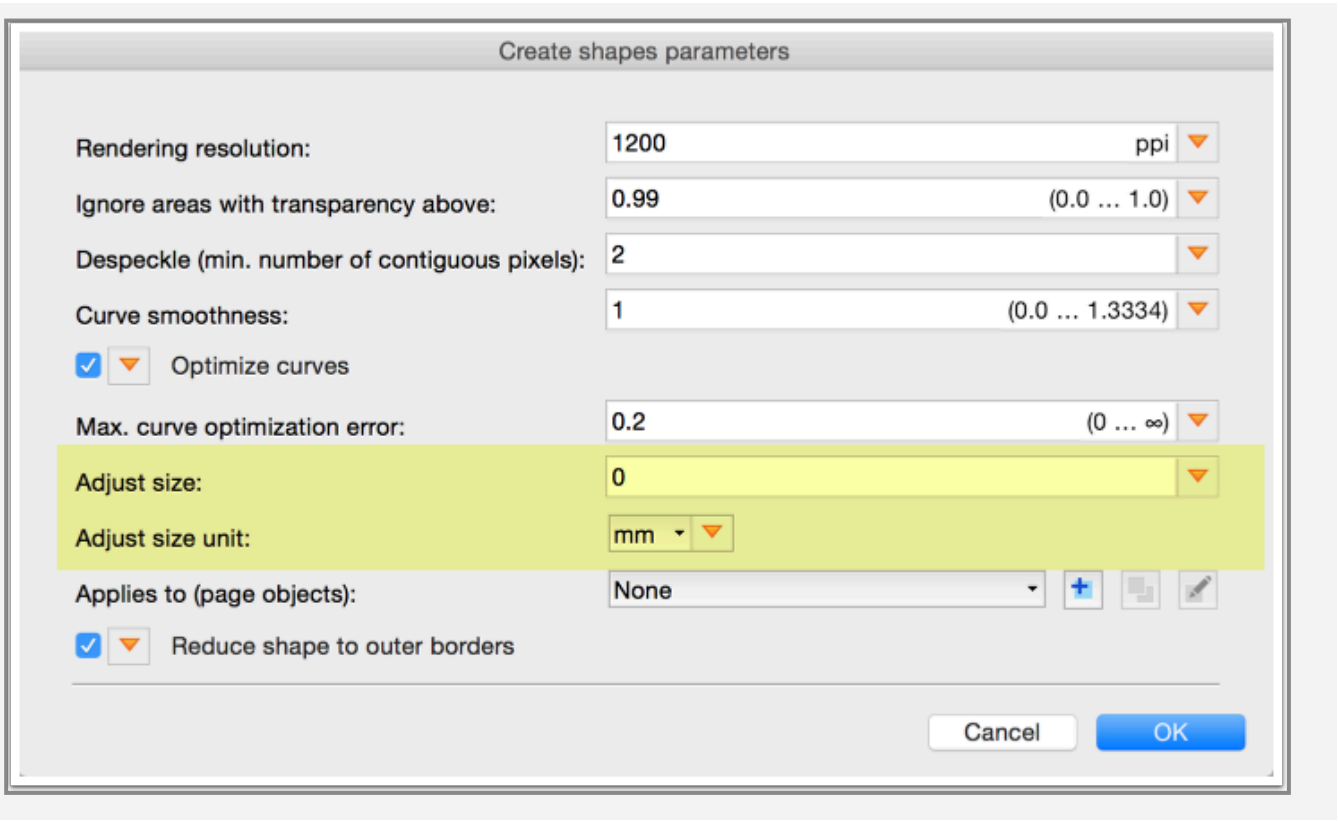
The settings for enlarging or reducing a shape's size can be found inside the "Parameter:" dialog.



The settings for enlarging or reducing a shape's size can be found in the lower half of the "Parameter:" dialog in the form of

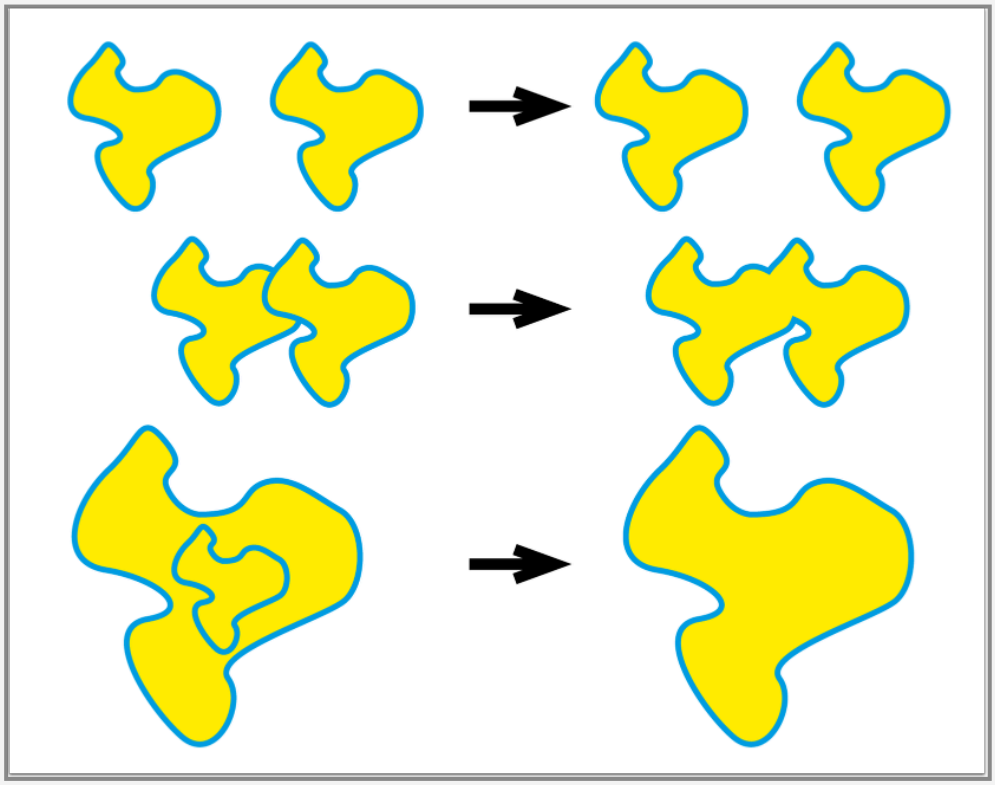
- the "Adjust size" field, and
- the "Adjust size unit" option, supporting "mm" (millimeter), "pt" (point) or "in" (inch) as units

A negative value for "Adjust size" will reduce the shape, whereas a positive value for "Adjust size" will enlarge it.



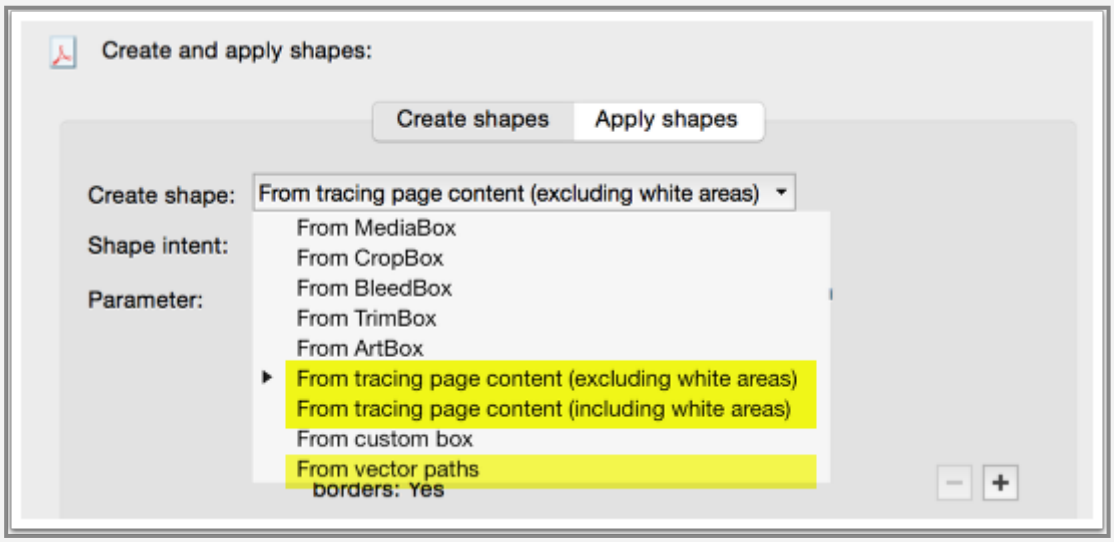
"All-inclusive" shape: Reduce shape to outer borders

The drawing below illustrates how the new "all-incluse" option impacts the shape that results from tracing page content or deriving a shape from existing vector paths. The main effect is that any shapes that exist inside other shapes are discarded. Where shapes overlap, the combined area of such overlapping shapes will be used as the actual shape. For shapes that are neither nested nor overlapping, the option has no effect.

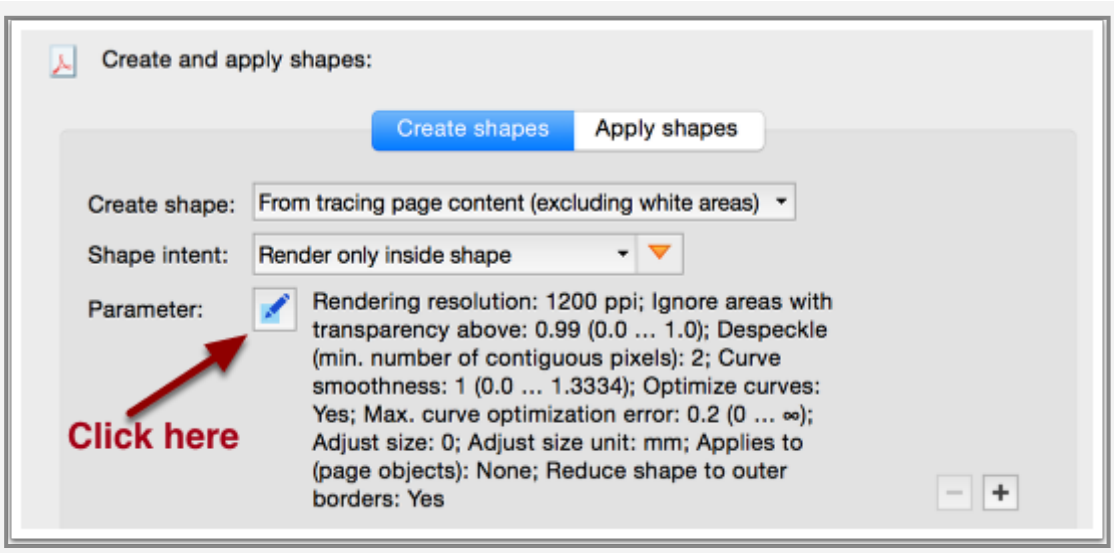


The "Reduce shape to outer borders" is only available for the three "Create shape" variants

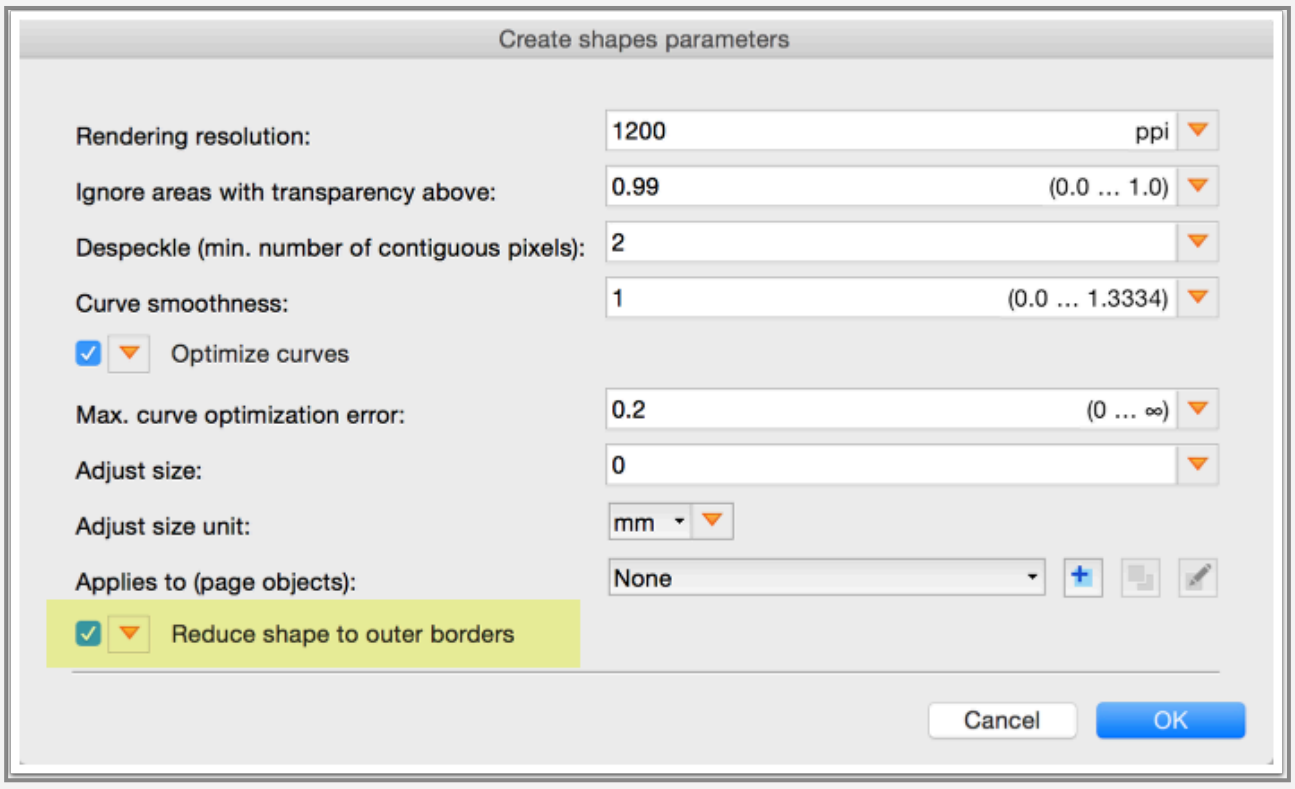
- From tracing page content (excluding white area)
- From tracing page content (including white area)
- From vector paths



The "Reduce shape to outer borders" can be found inside the "Parameter:" dialog.



The "Reduce shape to outer borders" can be found at the bottom of the "Parameter:" dialog.



21.5 Efficiently creating varnish or white background (requires at least v9.1)

In pdfToolbox 9.1, the "Shape" feature has been enhanced. This article shows how to take advantage of the enhancements when creating a varnish or a white background. The two attachments below provide the sample PDF and the pdfToolbox Library containing the fixups used in this article.

Sample file and pdfTo pdfolbox Library with pre-configured fixups

donuts.pdf

Shapes-_creating_varnish_or_white_background.kfpl

Growing or shrinking a shape

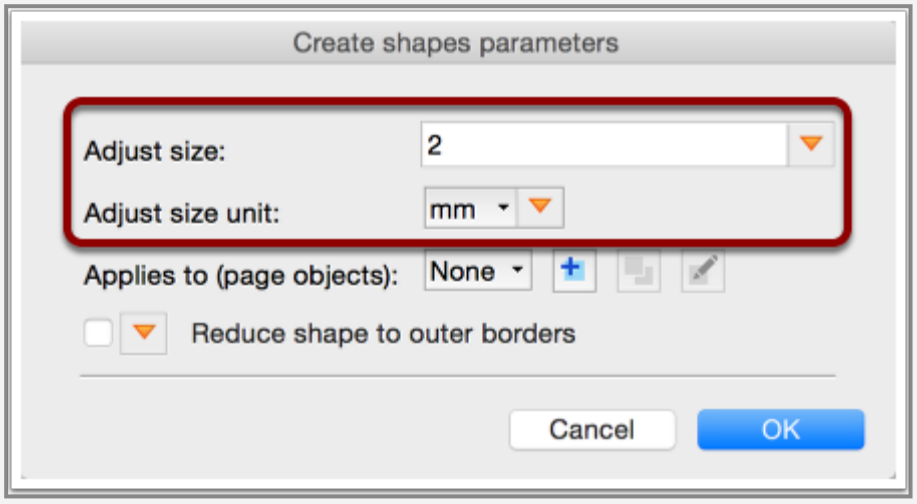
When creating a **partial varnish** it is often desirable to derive the area where varnish shall be applied from actually printed content. At the same time, mostly in order to compensate for less than perfect registration of plates or print heads, it is usually necessary **that the varnish extends slightly, maybe by a millimeter or two, beyond the printed content area**, to ensure that printed content is always varnished. The fact, that some small area where nothing is printed also receives varnish, is typically not considered a problem.

Along the same lines, but usually in the other direction, a white background may have to be created for printing on transparent substrate – but as the background shall not become visible as such, it needs to be shrunk by a little bit to pull back from the border of the print content area.

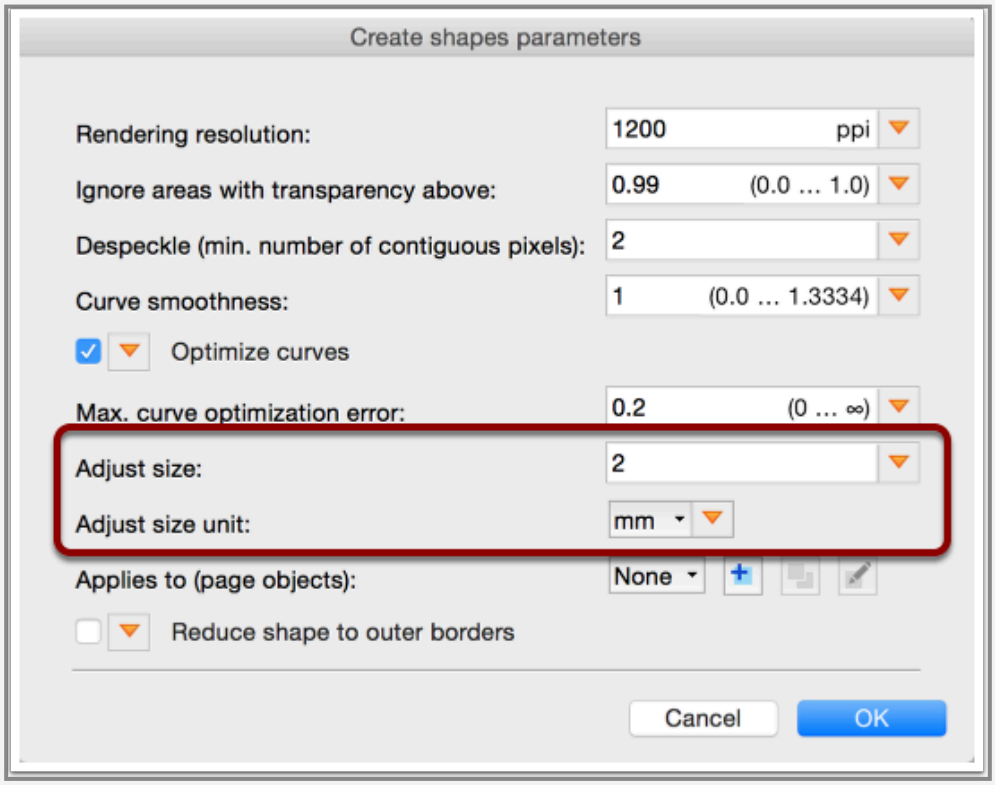
Both requirements can now be met very easily by the extended "Shape" feature, using the "adjust size" setting. This setting was already available for shapes based on page geometry boxes or a custom box but now can also be used for shapes derived from tracing page content or from existing vector paths.

In addition, a new setting "Reduce shape to outer borders" makes it possible to include 'holes' inside print content areas, which can be very handy for the generation of a white background (see example below). Extending/shrinking shapes and 'reduce to outer border' can be freely combined with each other and any of the other options.

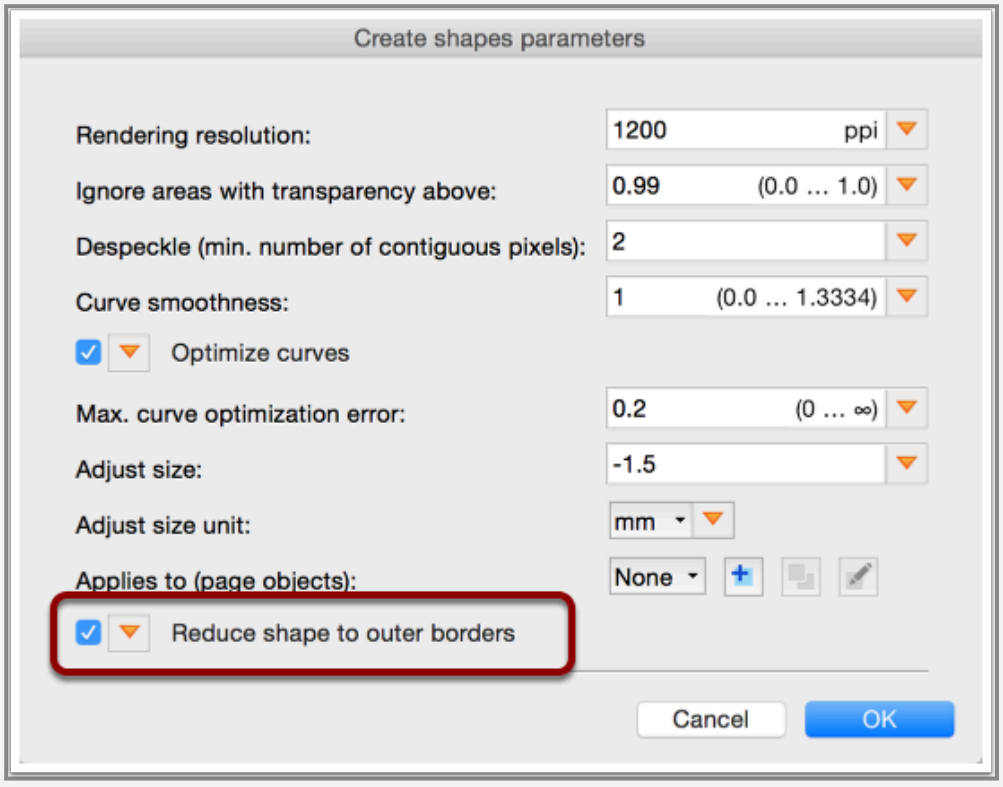
Create shape parameters for "From vector paths" option, with highlighted "Adjust size" settings:



Create shape parameters for "From tracing page content" option, with highlighted "Adjust size' settings:

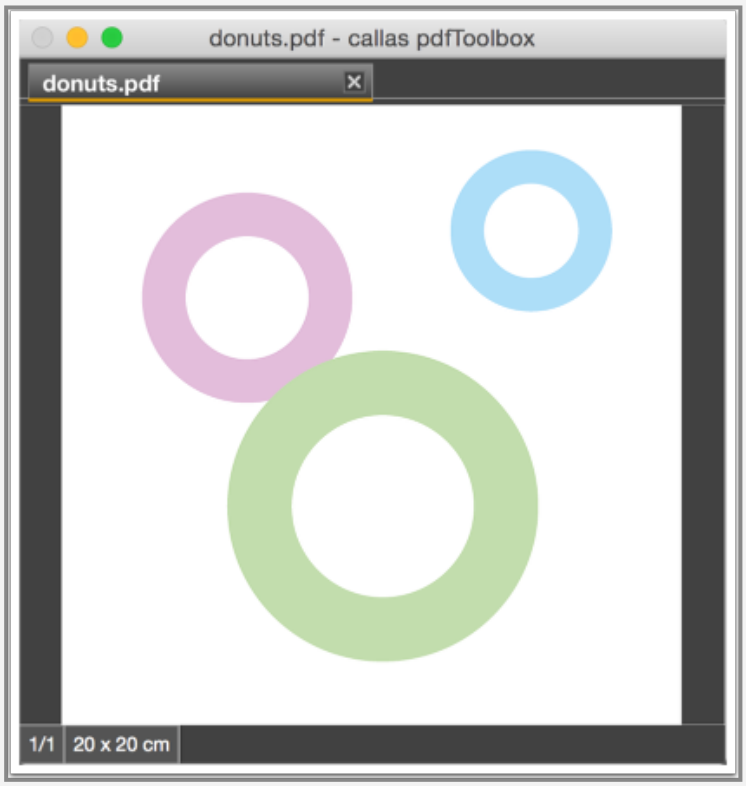


"Reduce shape to outer borders" setting:

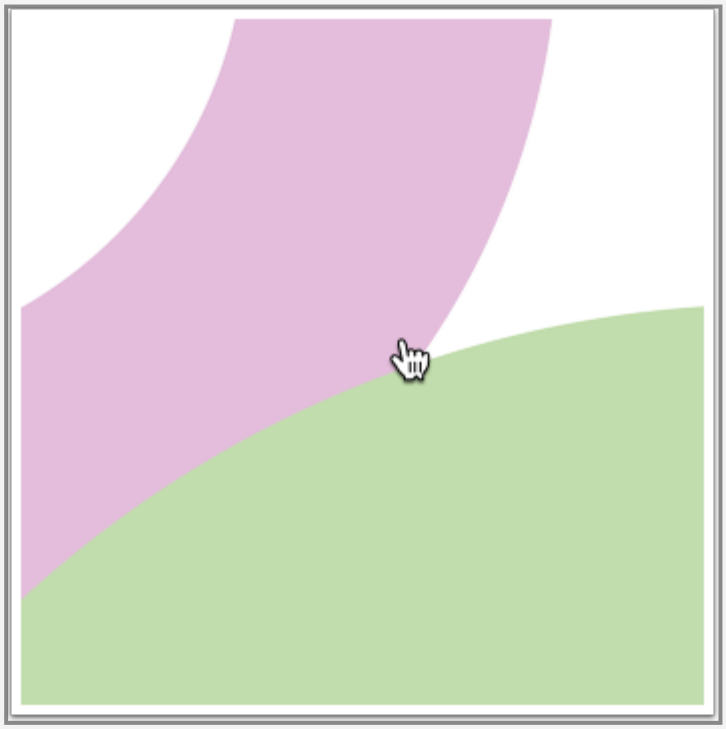


Example: Extending varnish

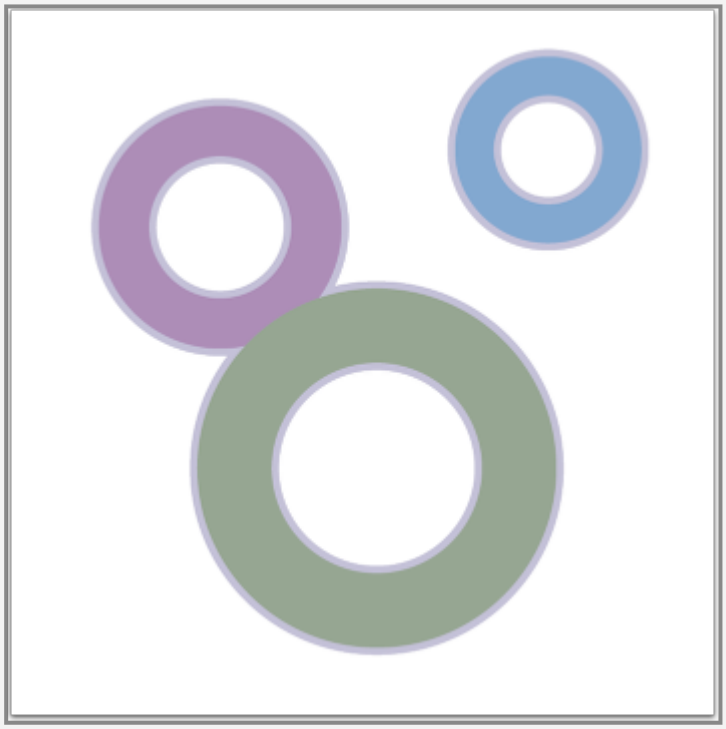
Sample file (see above for download):



Sample file (see above for download, enlarged detail):



Varnish applied to all page objects based on their vector paths, after applying "Varnish over print objects +2mm (vector)":

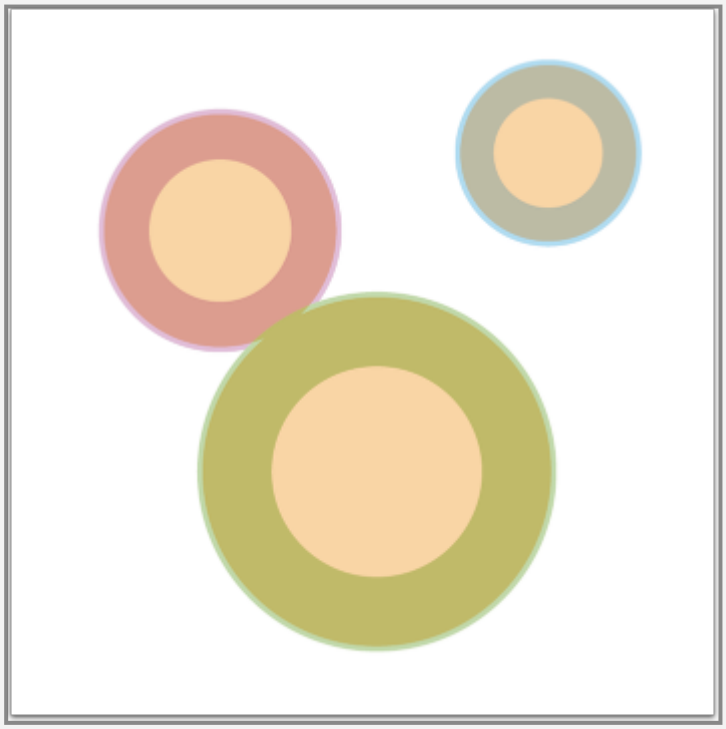


Varnish applied to all page objects based on their vector paths (enlarged detail):



Example: Shrinking white background, after reducing shape to outer border

White background derived from all page objects based on their vector paths, using only the outer border of all paths, using "White backing under print objects -1.5mm, based on outer border (vector)":



White background derived from all page objects based on their vector paths, using only the outer border of all paths (enlarged detail):



21.6 Use shapes to visualize small distances between objects or inside of outlined objects

The shapes technology can be combined with a method used in image analysis to visualize areas where minimum distance requirements are not met.

Example of practical application

Determining the minimum distance between two objects (or contour lines of an object) is important for films that are output on cutting plotters.

How does this image analysis work?

The algorithm first reduces the shape of the object by half of the minimum distance. Where the minimum distance is not kept that means that the shape will be completely removed. Then the shape is enlarged by the same value so that everything is restored - except those areas that have been removed in the first step.

This can best be demonstrated with an example. We have a PDF file in which the word "Test" has been placed twice with only a cut contour.

To check the critical distances, three Fixups must be performed in each of the two cases: Visualize the critical distances between objects and visualize the critical distances within objects.



Testfile.pdf

Check minimum distance between contour lines within an object:

1. Fixup: Fills all vector objects with Magenta.
2. Fixup: Reduces the area by a specified value and fills it with Magenta as well.
3. Fixup: Enlarges the area defined in Fixup 2 again by the previously specified value and fills it with gray.

As a result, all areas where the filled objects become narrower than 2 mm are marked Magenta. Unproblematic areas are covered with gray color.



Check minimum distance between two or more objects:

1. Fixup: Fills the page outside the vector objects with Cyan.
2. Fixup: Reduces the area by a specified value and also fills it with Cyan.
3. Fixup: Enlarges the area (defined in Fixup 2) again by the previously specified value and fills it with gray.

As a result, all areas where the minimum distance between the objects is less then 4 mm are marked Cyan. Unproblematic areas are covered with gray color.



Of course, these two analyses can be combined in one Process Plan. The Process Plan attached below visualizes both cases so that you can easily check all critical areas in your PDF. The Process Plan contains variables so that you can individually define the spot color name of your cut contour and the minimum distance (in mm).

Visualize minimum distances between objects.kfpx

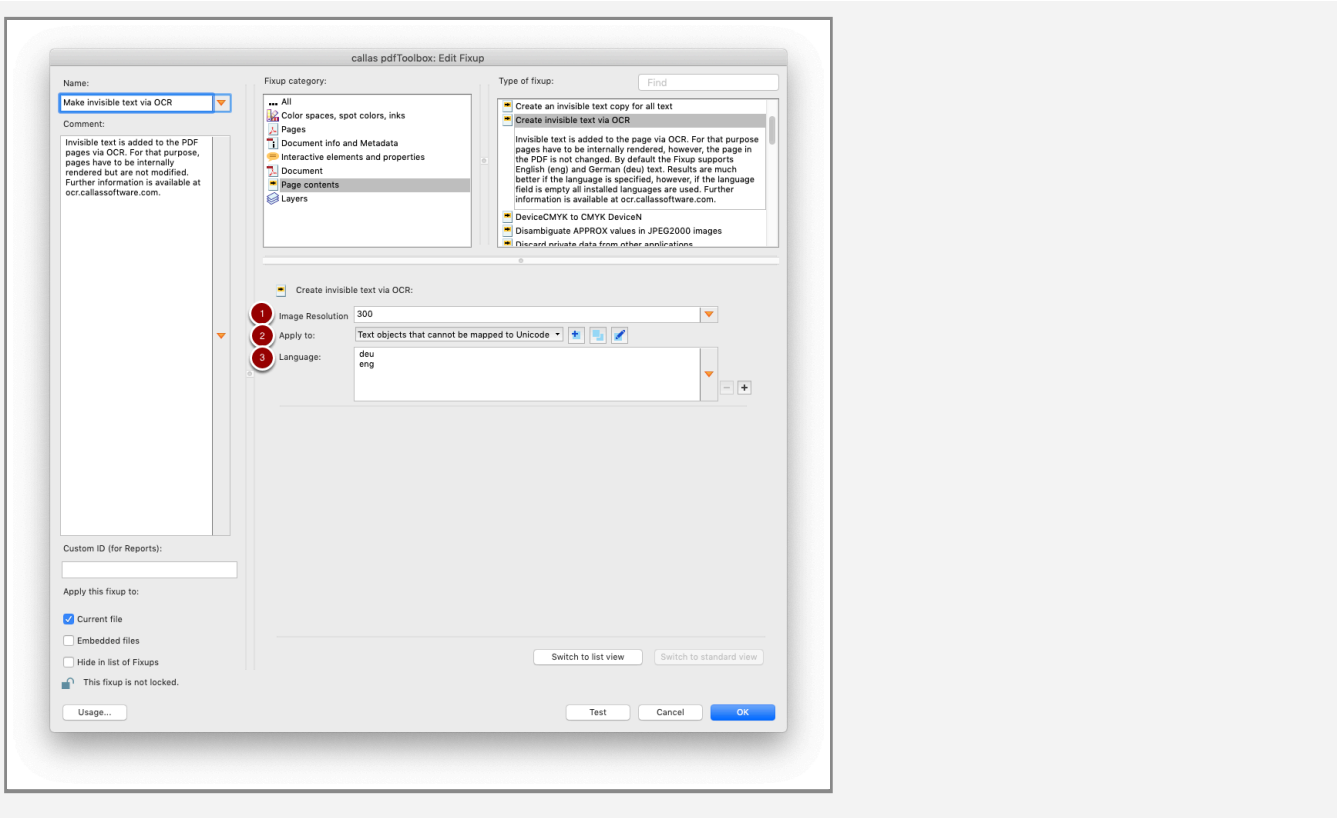

TEST
TEST

22. Optical character recognition (OCR)

22.1 Create invisible text via OCR

A customer scans a document from paper format and now the document is available in a "primitive" electronic form (as an image, with e.g. TIFF format). However, the text is not searchable.
pdfToolbox 12, using Tesseract technology (open source), allows to create an output PDF with searchable text from an input document which can be an image or a PDF. The visual representation of the input document is preserved where the produced PDF has an overlay containing the searchable text without a visual representation.

New Fixup: Create invisible text via OCR (in English and German)

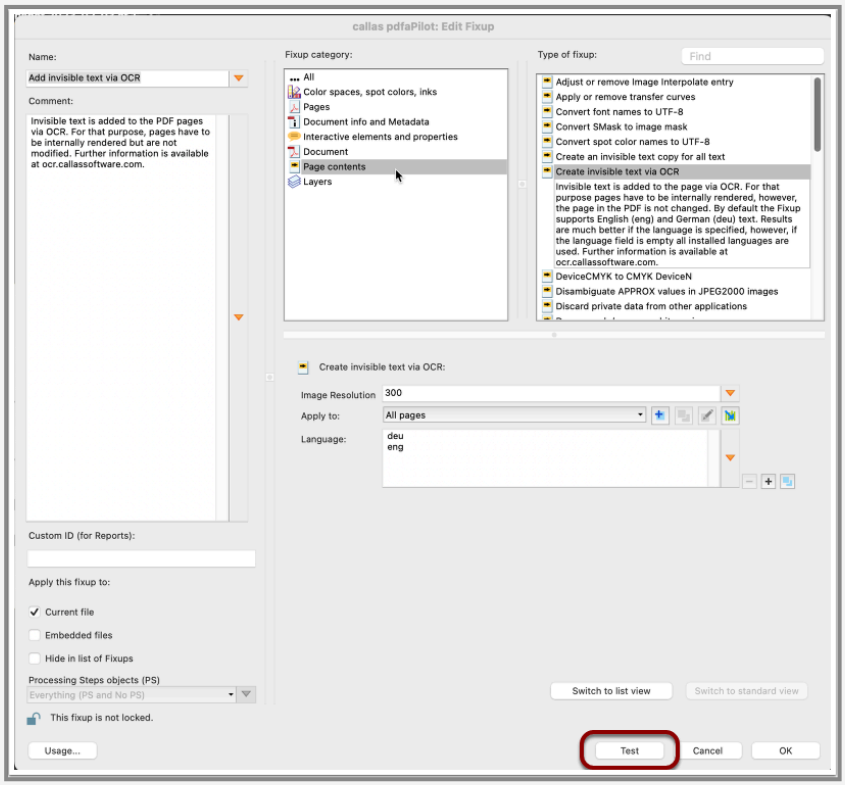


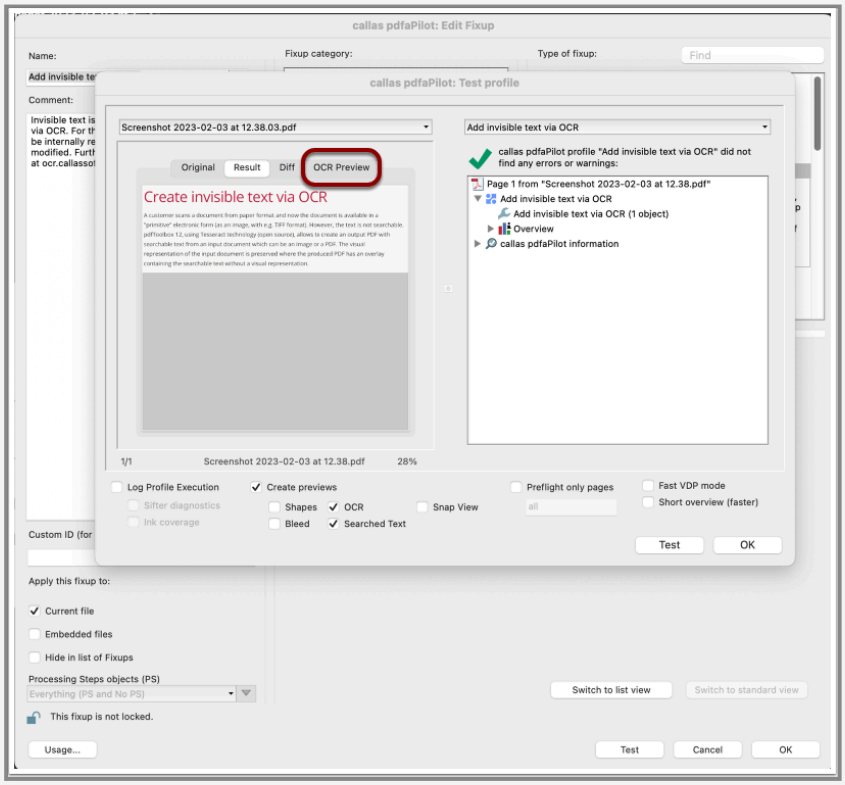
1. Image Resolution:
2. Apply to: Apply the Fixup to, for example, all pages OR
   - Only to text that cannot be mapped to Unicode (as in the screenshot above)

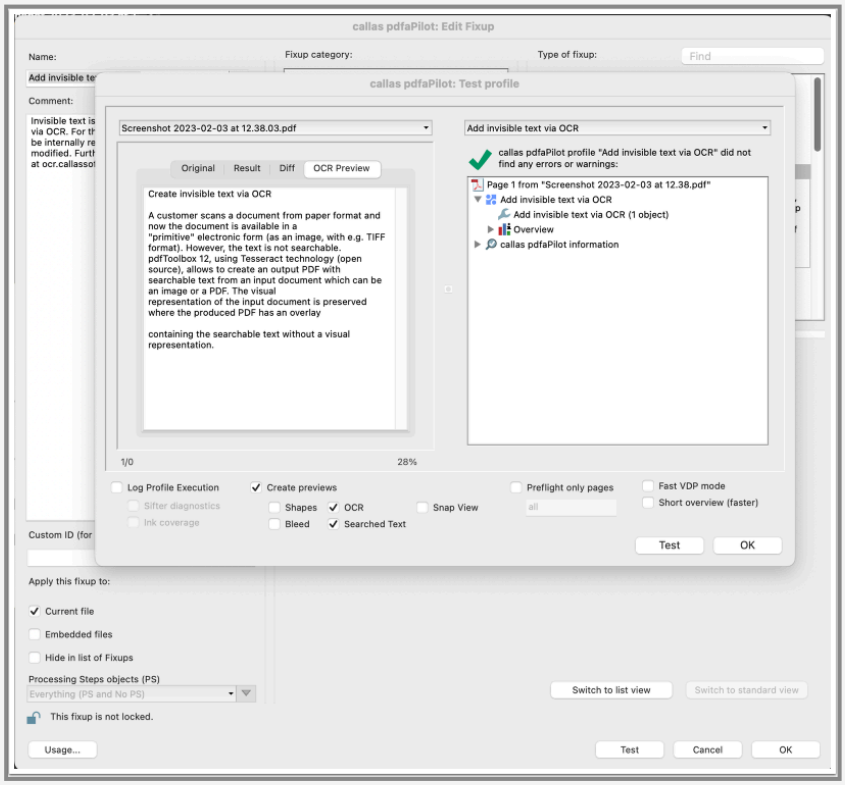
3. Language: Results are much better if the language is specified, however, if the language field is empty, all the installed languages (how to install languages) are used. This input field has to be used with 3 character ISO language codes.

Double check OCR results

You can see results of the OCR in "Test" mode.



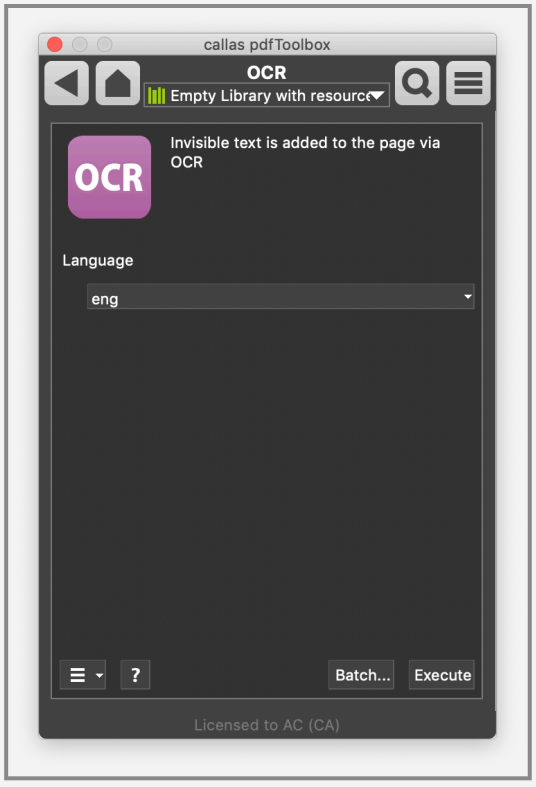

callas pdfaPilot: Edit Fixup
callas pdfaPilot: Test profile
Screenshot 2023-02-03 at 12.38.03.pdf
Add invisible text via OCR
Original
Result
Diff
OCR Preview
Create invisible text via OCR
callas pdfaPilot profile "Add invisible text via OCR" did not find any errors or warnings:
Page 1 from "Screenshot 2023-02-03 at 12.38.pdf"
Add invisible text via OCR
Add invisible text via OCR (1 object)
Overview
callas pdfaPilot information
Log Profile Execution
Create previews
Preflight only pages
Fast VDP mode
Short overview (faster)
Shapes
OCR
Snap View
Bleed
Searched Text
Test
OK
Apply this fixup to:
Current file
Embedded files
Hide in list of Fixups
Processing Steps objects (PS)
This fixup is not locked.
Switch to list view
Switch to standard view
Usage...
Cancel


callas pdfaPilot: Edit Fixup
callas pdfaPilot: Test profile
Screenshot 2023-02-03 at 12.38.03.pdf
Add invisible text via OCR
Original
Result
Diff
OCR Preview
Create invisible text via OCR
A customer scans a document from paper format and now the document is available in a "primitive" electronic form (as an image, with e.g. TIFF format). However, the text is not searchable. pdfToolbox 12, using Tesseract technology (open source), allows to create an output PDF with searchable text from an input document which can be an image or a PDF. The visual representation of the input document is preserved where the produced PDF has an overlay
containing the searchable text without a visual representation.
callas pdfaPilot profile "Add invisible text via OCR" did not find any errors or warnings:
Page 1 from "Screenshot 2023-02-03 at 12.38.pdf"
Add invisible text via OCR
Add invisible text via OCR (1 object)
Overview
callas pdfaPilot information
Log Profile Execution
Create previews
Preflight only pages
Fast VDP mode
Short overview (faster)
Shapes
OCR
Snap View
Bleed
Searched Text
Test
OK
Apply this fixup to:
Current file
Embedded files
Hide in list of Fixups
Processing Steps objects (PS)
This fixup is not locked.
Switch to list view
Switch to standard view
Usage...
Cancel

As stated in the Fixup's comment, the Fixup supports English (eng) and German (deu) text by default. You can install further languages as explained here.

Action: OCR

The same can be achieved using a Switchboard Action. Simply look for the Action 'OCR':



22.2 OCR support for additional languages

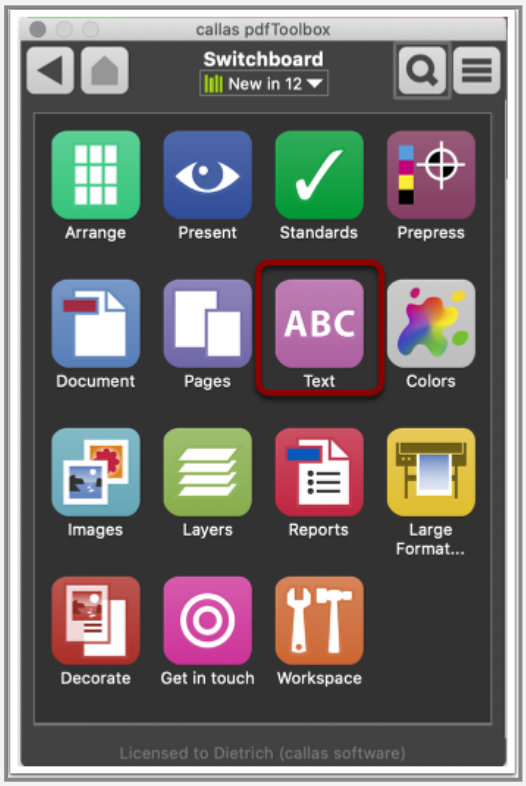
The Fixup 'Create invisible text via OCR' by default supports English and German. This article describes how to add support for additional languages.

The "Create invisible text via OCR" Fixup internally uses the Tesseract engine. Language files (trainings) for various languages can be obtained from a github repository that is maintained by the Tesseract community:

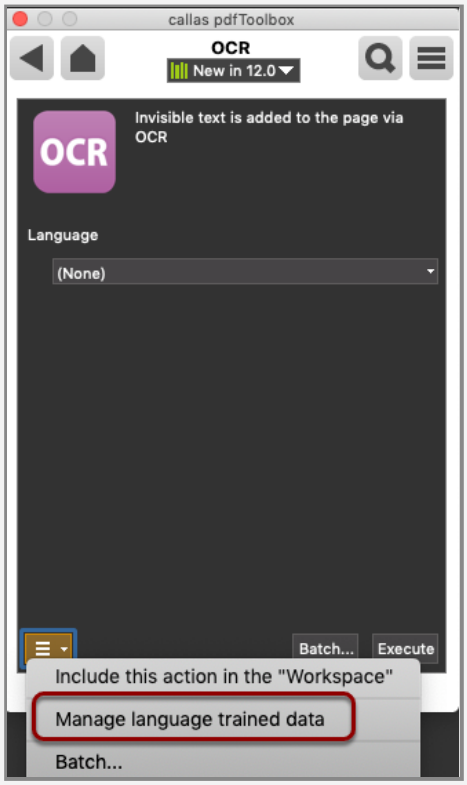
https://github.com/tesseract-ocr/tessdata_fast

Making language trainings available to pdfToolbox Desktop

In order to use these files in pdfToolbox Desktop, they have to be put into the folder "OCR" in the preferences folder. The easiest way to get to this folder is to open the Switchboard.

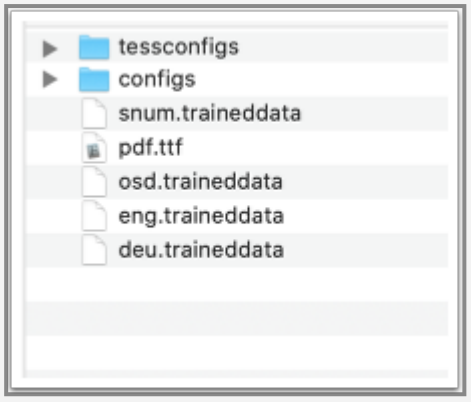


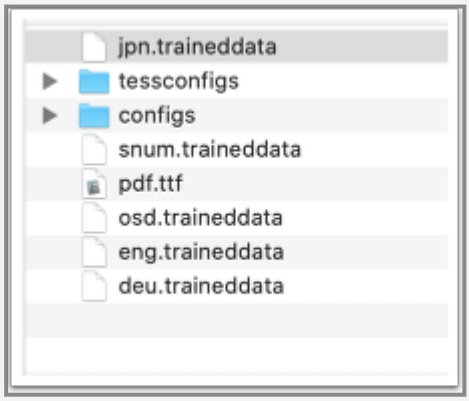
Go to Text, OCR



Then click on the options item at the bottom and select: Manage language trained data. That will open the folder or - if no language trainings were used beforehand - create it.

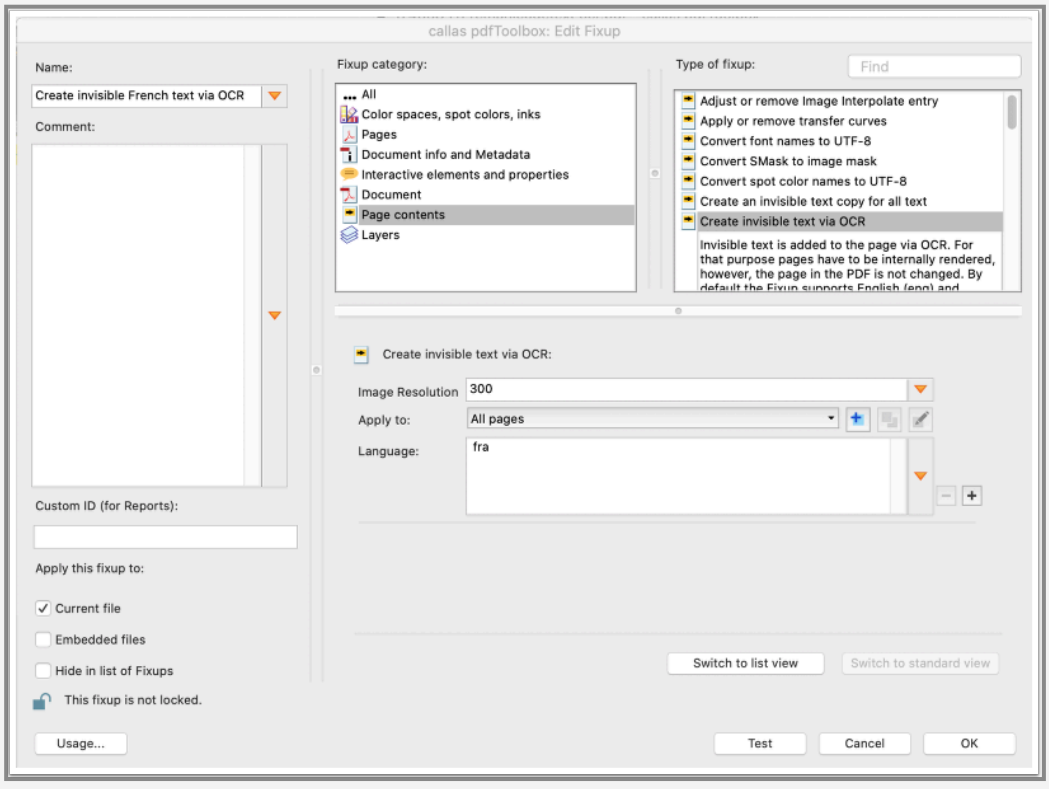
You may now put language trainings into this folder.





Referencing language trainings in a Fixup

After a language training is installed, it can be used in a Fixup.

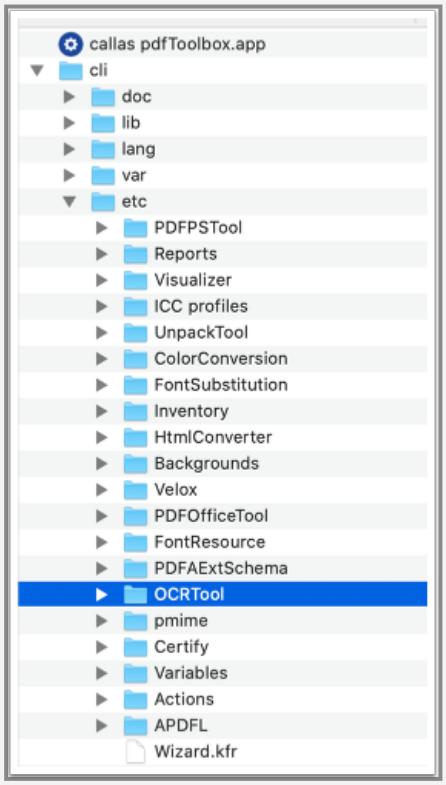


Performance and quality are better if only those languages are specified that are absolutely needed.

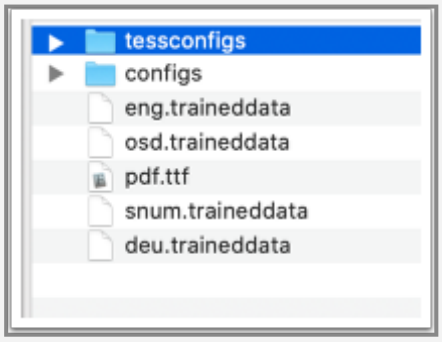
If more than one language training is used the most accurate one should be placed at the top of the list, since it will be used with priority by the engine.

Using language trainings in pdfToolbox Server/ CLI or SDK

When you export a Profile using the "Create invisible text via OCR" Fixup, the language trainings will not be exported. Instead they will have to be installed in the instance of pdfToolbox Server/CLI or pdfToolbox SDK that you are using. All these applications have a subfolder named "etc" in their program folders.



There you will find the folder "OCRTool" and there "tessdata". Any language trainings that you want to use in pdfToolbox Server/CLI or pdfToolbox SDK have to be put into this folder.

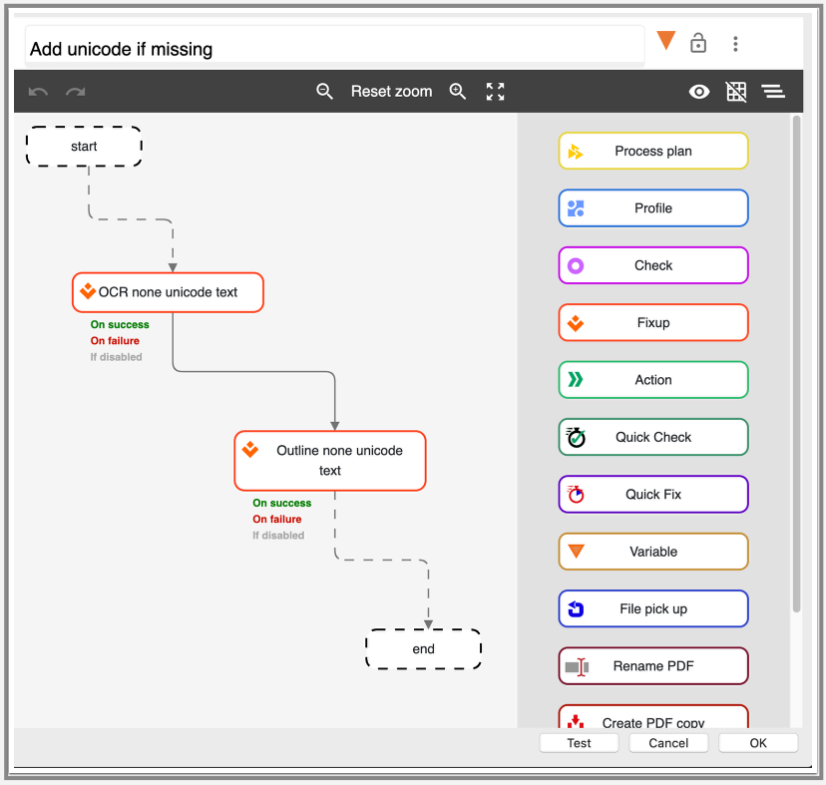


Note that English and German are installed beforehand. They will be used even if no language setting is specified in the Profile. However, if you process more German text than English you should still specify the language to improve results.

22.3 Partial OCR (filtering page content)

You can understand OCR as adding semantics to characters and words which are otherwise only shapes (glyphs). Whether or not a character has such semantics is often described as Unicode representations: if there is such semantics that means that there is a known Unicode code point for the respective character (glyph).

In most PDFs at least most characters have Unicode representations, unless they are created by a scanner from paper. However, some PDF creators are not as thorough as they should be and skip certain characters. In such cases, a full OCR would add additional Unicode codes to characters that already have one. This is not a big problem for text search or copying text out of the PDF, however, if you create a full text extract from the PDF you will end up with double paragraphs with the same content. It would be better to OCR only those text portions that do not already have Unicode. The Process Plan introduced in this article does exactly that.

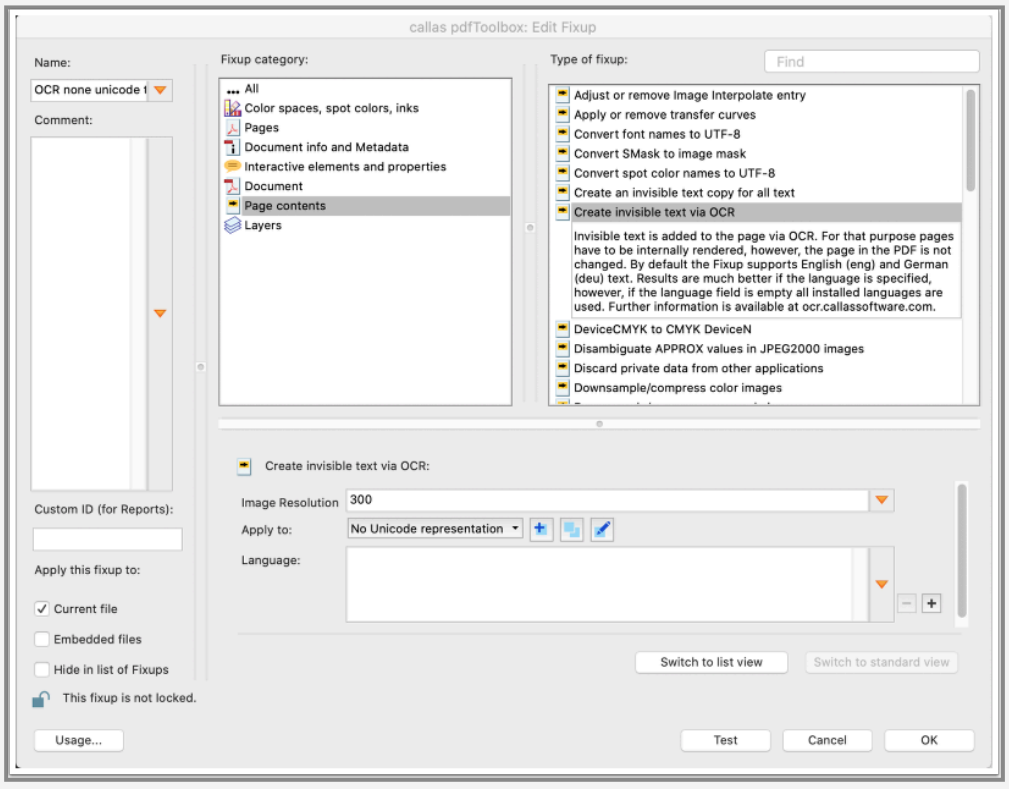


Add_unicode_if_missing.kfpx

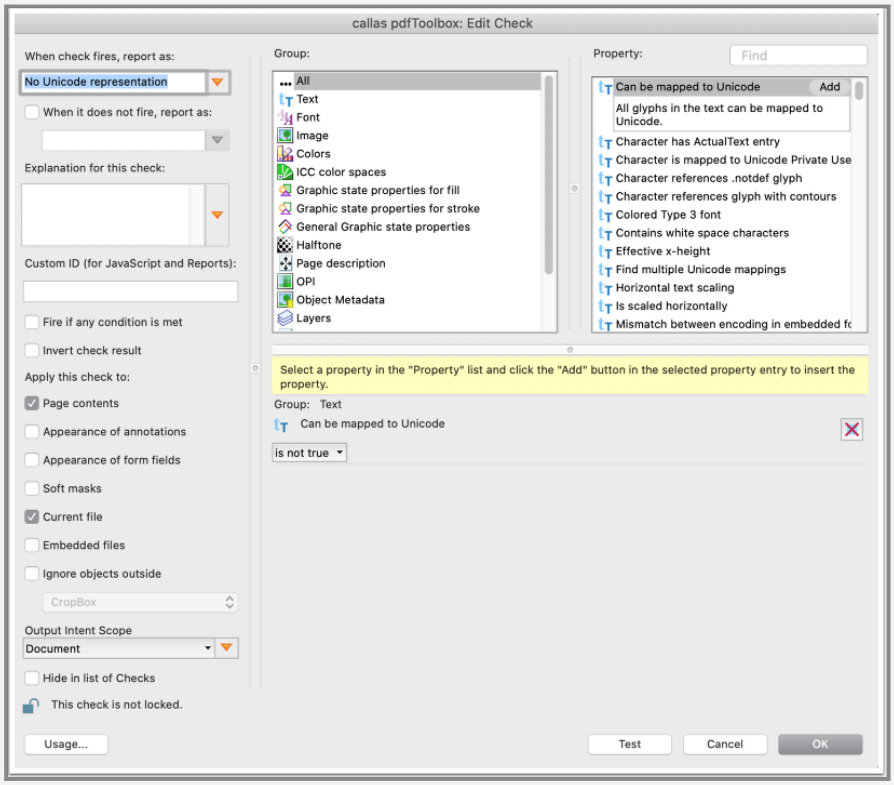
Step 1: OCR text that does not have Unicode

The "Create invisible text via OCR" Fixup has an "Apply to" filter. If used only those page objects are rendered into the intermediate image that is the basis for the OCR that are found by the filter.

Since we want to OCR all text that does not already have Unicode we are using this filter.



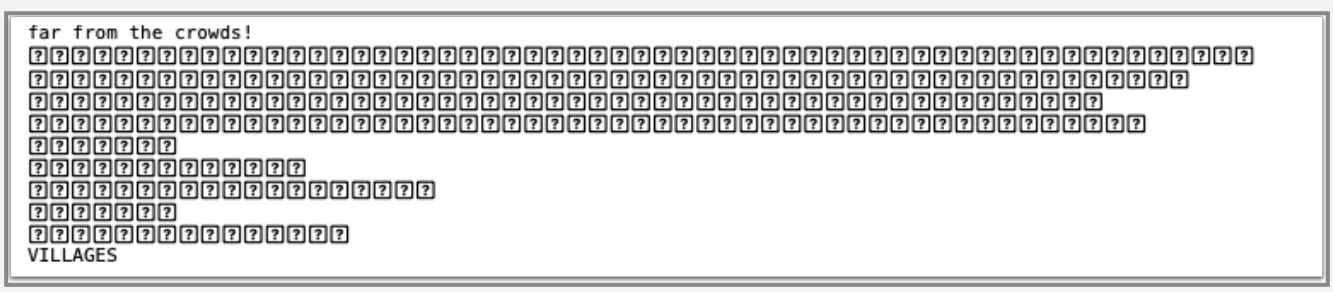
And the "No Unicode representation" filter uses



This adds additional, invisible "characters" to all text that has no Unicode representation and these characters have Unicode.

Step 2: Outline none Unicode text

When you want to prepare for text extract (and that is the main purpose of this approach as explained above) you should in addition remove all text without unicode, otherwise you will see some "invalid unicode" indicators in your text extract:



Thanks to the OCR that we have done there in addition is the unicode text, but better to get rid of this as well.

Therefore the second step converts the glyphs of the none Unicode text into outlines. Then it will not be text for the extraction engine.

23. Barcode recognition

23.1 Find barcodes

This callas pdfToolbox property searches for any number of barcodes or matrix codes in the specified area for a rendering with given resolution.

Available since version 12

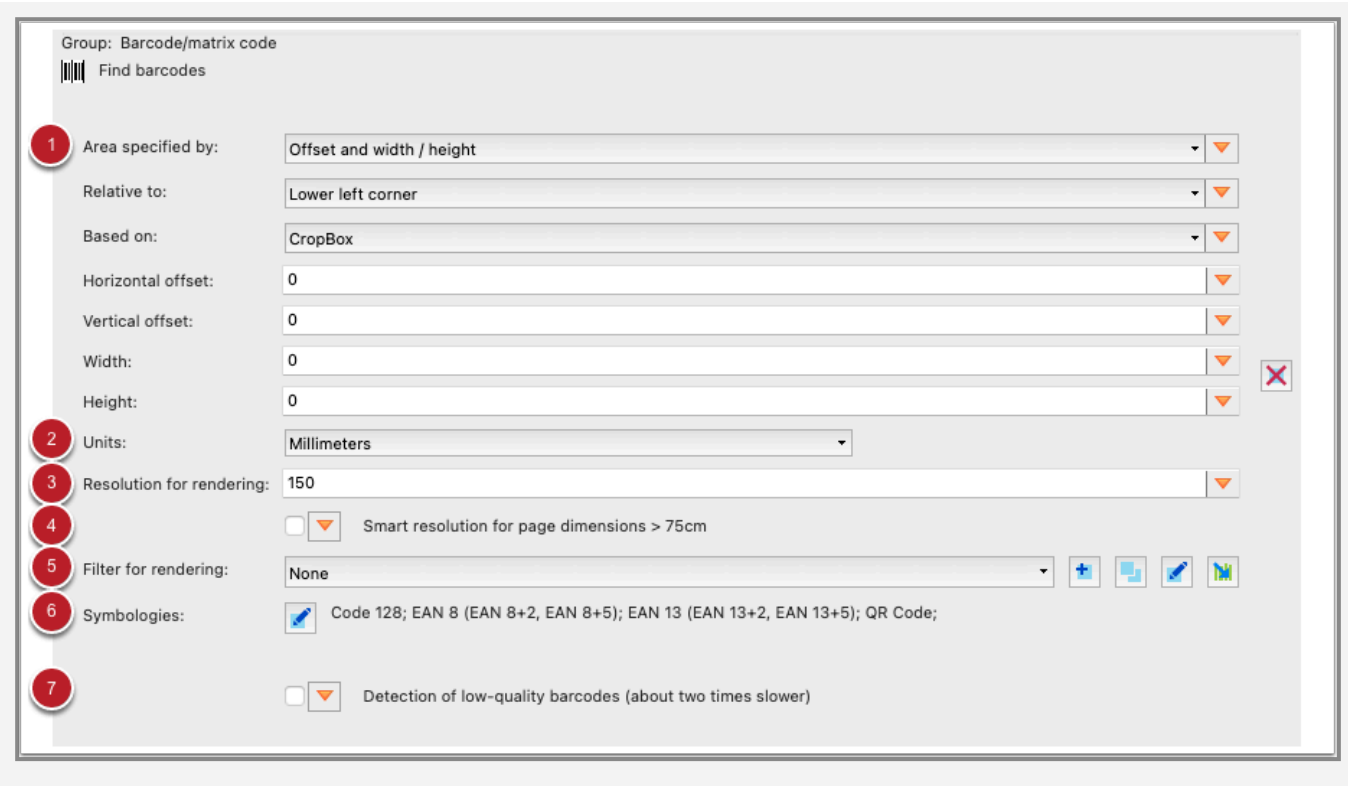
How is 'Find barcodes' different from 'Barcode is in area'?

'Find barcode' property can return any number of barcodes which shall be combined with other barcodes properties like 'Barcode value' or 'Symbology' (type of barcode/matrix code) and therefore can create multiple hits in contrast to the 'Barcode is in area' property that always only creates one hit per check.

Find barcodes property, in addition, enables:

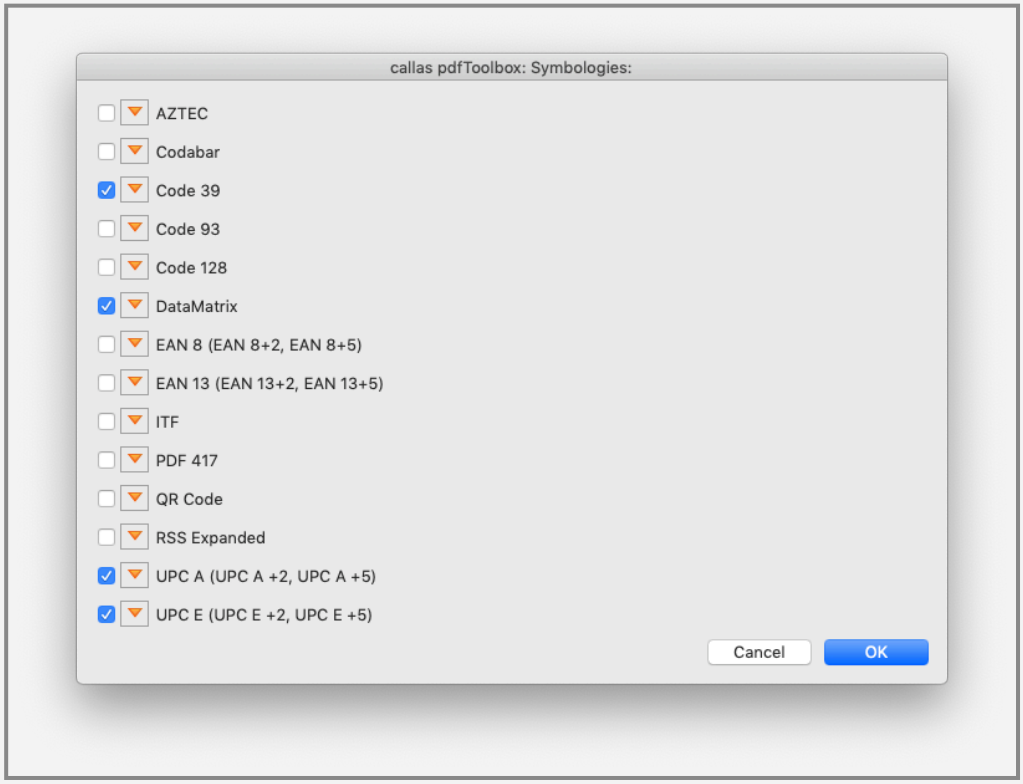
- finding barcodes/matrix codes for low-resolution scans or similar low-level quality originals
- finding barcodes/matrix codes for pages with many barcodes very close to each other

'Find barcodes' property



1. Area specified by: Two methods to define the search area:
   - Offset and width / height: "Relative to" sets the origin point. All subsequent parameters are based on this origin (e.g. if the origin point is set to top left corner, a negative vertical offset will move the origin down, while a positive offset will move the origin up).
   - Offsets to box edges (inwards - /outwards +): Takes the size of a page geometry box as a reference for the custom area. Negative values reduce the size of the page geometry box. Positive values increase the size.
2. Units: Unit of measurement in cm, mm, inches, points or picas
3. Smart resolution: Enables auto adjustment of the sample size for PDF files with at least one page dimension greater than 75 cm. Read more about Smart Resolution here.
4. Resolution for rendering: Resolution of the (internally rendered) page image
5. Filter for rendering: Makes it possible to create a rendered page only based on the objects found by this filter.

6. Symbologies: Supported barcode types (also shown below)
7. Detection of low-quality barcodes: Checkbox to detect low-quality barcodes which makes the search about two times slower

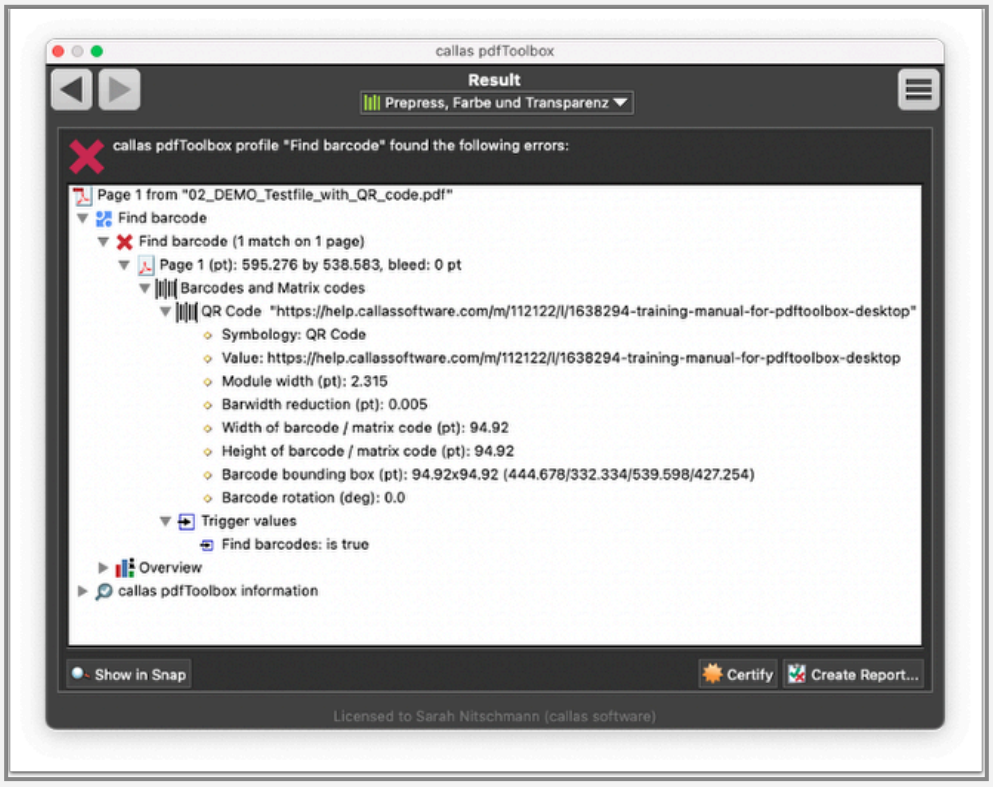


The Barcode detection is performed on an internally rendered image of each page. Therefore only the coordinates are available (besides the determined details of the barcode). For this reason, found barcodes can not be handled by other Fixups (e.g. colors of a barcode can not be explicitly converted).

Trigger values

The trigger value of the "Find barcodes" Check is alway "is true". If you want to receive the code's parameters in the trigger values the Check needs to be combined with all other barcode Check properties in the group "Barcode/matrix

code". How to do this is explained in this article: Read Barcode or Matrix code and determine properties.



Since pdfToolbox 15 you can get all properties of a barcode or matrix code in the result object (`app.doc.result.checks[i].hits[h]`) under `"parameters"` in JavaScript.

```
{
  "llx": 444.6780090332031,
  "lly": 332.3340148925781,
  "page": 0,
  "parameters": {
    "symbology": "QR Code",
    "value": "https://help.callassoftware.com/m/112122/l/1638294-training-manual-
for-pdftoolbox-desktop",
    "barWidthReduction": 0.0050000000000022735,
    "moduleWidth": 2.315,
    "barRotation": 0
  },
  "triggers": [
    {
      "id": "BARCODE_Group::BARCODE_FindBarcodes",
```

```
      "value": true
    }
  ],
  "type": "Barcode",
  "urx": 539.5980224609375,
  "ury": 427.2539978027344
}
```


Click to copy

Note: If you want to have the **quiet zone** or **color distance (includes color distance, barcode color and background color)** properties in the result object, you must add the appropriate Check properties to the "Find Barcodes" Check. They are not provided by default because it takes some time to calculate the results.

23.2 Supported Barcode symbologies

Focussing on quality over quantity, this page defines those symbologies that will be supported for best results in pdfToolbox. More symbologies may follow at a later stage, also driven by your feedback/requests. If you think there is something missing from the below list and it might be of interest to you, please let us know by contacting our support.

Supported Barcodes

Code type	Notes
EAN 13	EAN 13 is an extension of the UPC-A barcode symbology that usually carries a GTIN-13. Also known as: EAN, UCC-13, European Article Number, International Article Number, JAN, JAN-13, IAN, WPC, SAAN, UCCET, ABAC, BCCI, ICA, MANA, KANC, ANA, ANC.
ISBN	An ISBN barcode is a variant of EAN-13 that is used to identify books. Also known as: ISBN-13, International Standard Book Number, Bookland EAN-13. Standards: ISO 2108, ISO/IEC 15420, BS EN 797, GS1 General Specifications.
ISMN	Variant of EAN-13 with a prefix 979 that is used to identify printed music. Also known as: International Standard Music Number, ISMN-13. Standards: ISO 10957, ISO/IEC 15420, BS EN 797, GS1 General Specifications
ISSN	An EAN-13 with prefix 977 used to identify periodicals. Also known as: International Standard Serial Number. Standards: ISO 3297, ISO/IEC 15420, BS EN 797, GS1 General Specifications
EAN 8	EAN 8 is derived from the EAN 13 barcode symbology and is designed for small packaging. It usually carries a GTIN-8. Also known as: UCC-8, JAN-8.
UPC A	The UPC A barcode symbology is used for identification of retail goods at point of sale inside of the

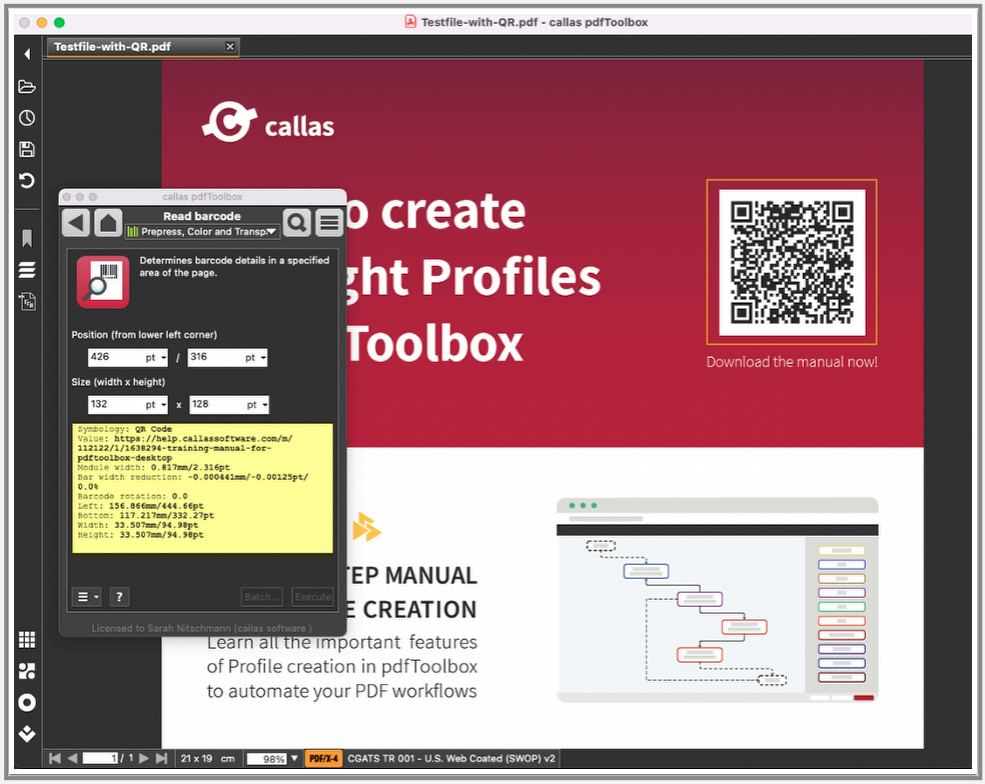
Code type	Notes
	US. It usually carries a GTIN-12. Also known as: UPC, UCC-12, Universal Product Code.
UPC E	UPC E is a compacted form of the UPC-A barcode symbology that usually carries a GTIN-12. tandards: ISO/IEC 15420, BS EN 797, GS1 General Specifications.
CODE 39	Also known as: Code 3 of 9, LOGMARS, Alpha39, USD-3, USS-39. Standards: ISO/IEC 16388, ANSI/AIM BC1 - USS Code 39, BS EN 800, MIL STD 1189
PZN7/PZN8	Used for pharmaceutical products in Germany, variations of CODE 39
CODE 93	Code 93 is a barcode symbology designed in 1982 by Intermec to provide a higher density and data security enhancement to Code 39
Code 128	Alpha-numeric codes supported. Used extensively worldwide in shipping and packaging industries as a product identification code for the container and pallet levels in the supply chain
ITF	Also known as: UPC Shipping Container Symbol, SCS, UPC Case Code. Standards: ISO/IEC 16390, ANSI/AIM BC2-1995 USS, BS EN 801, GS1 General Specifications.
CODABAR	Widely used for applications that require serial numbers, such as management of blood banks, slips for door-todoor delivery services and member cards
RSS 14	GS1 DataBar, a variant of RSS Expanded
RSS EXPANDED	
QR-CODE	Quick Response Code is the trademark for a type of matrix barcode or two-dimensional barcode
Data Matrix	A two-dimensional code consisting of black and white "cells" or dots arranged in either a square or rectangular pattern, also known as a matrix
Aztec Code	2D barcode. Standards: ISO/IEC 24778:2008

Code type	Notes
PDF417	PDF417 is used by both commercial and government organizations, e.g. airline boarding passes, car licenses or lottery tickets

23.3 Read Barcode or Matrix code and determine properties

Determine barcode properties using pdfToolbox Switchboard

You can read barcodes or matrix codes using pdfToolbox Switchboard Action 'Read Barcode' under the Group 'Report' in Switchboard. You can either define the position and size of the custom rectangle where you want to find the Barcode or work it automatically via "Mouse selection".
This will render the selected region to a grayscale buffer at a high resolution.



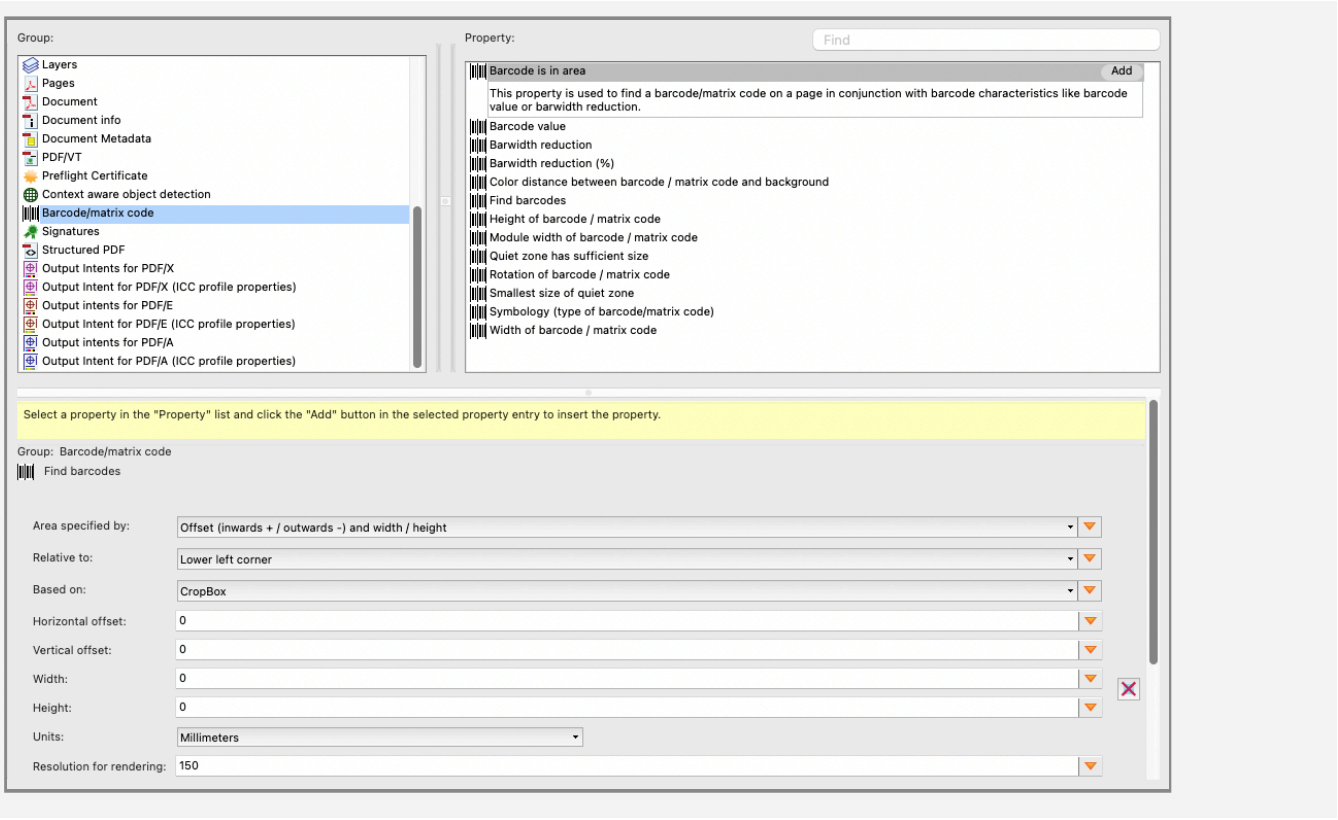
The results (value, type of barcode, bar width reduction, exact position and dimension of the barcode etc.), if any barcode or matrix code is found, will be shown in a new window from where the barcode information can be selected and copied, like the one below:

```
Symbology: QR Code
Value: https://help.callassoftware.com/m/112122/l/1638294-training-manual-for-pdftoolbox-desktop
Module width: 0.817mm/2.316pt
Bar width reduction: -0.000441mm/-0.00125pt/0.0%
Barcode rotation: 0.0
Left: 156.866mm/444.66pt
Bottom: 117.217mm/332.27pt
Width: 33.507mm/94.98pt
Height: 33.507mm/94.98pt
```

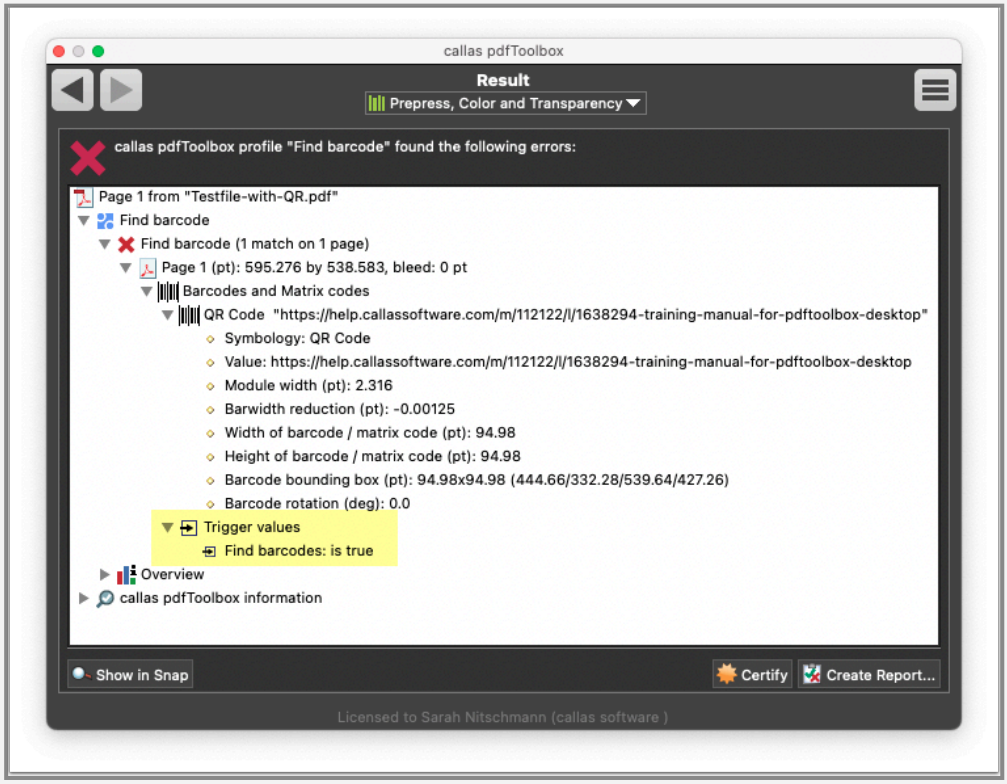
Click to copy

Determine barcode properties using pdfToolbox Checks

You can determine barcode properties like coordinates, height, module width, symbology (type), bar width reduction or width of barcode, using Checks in pdfToolbox (screenshot below).



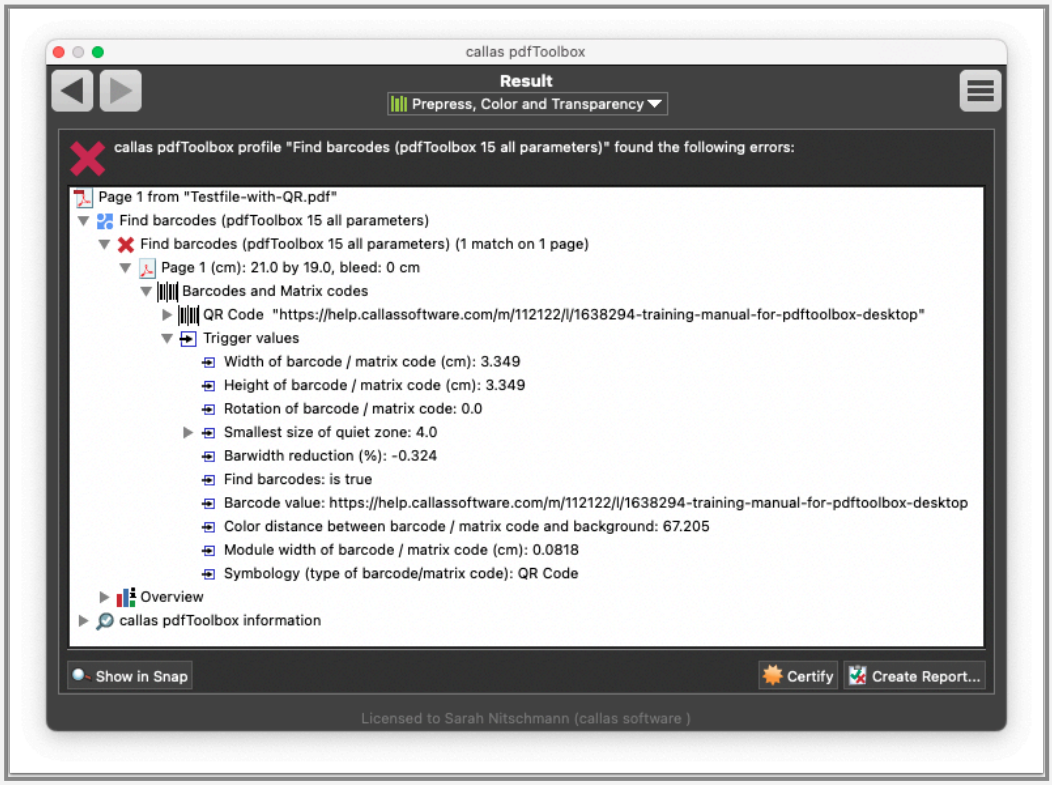
The identified barcodes and their parameters are displayed in the result view. However only the trigger for the selected Check property is available in the Trigger values. That is unfortunate if you want to e.g. use the barcode value in JavaScript.



That can be resolved by adding additional properties to the Check, e.g. "Barcode value", "Symbology (type of barcode/matrix code) and "Barwidth reduction (%)". You may add any of the barcode/matrix code related Check properties. You have to make sure that the Condition is set up to always generate a hit as in this example.



In this example we have added all Check properties.



You may download the Check from here.

Compatible with pdfToolbox 15 or newer:

Find barcodes (pdfToolbox 15 all parameters).kfpx

Compatible with pdfToolbox 12 or newer (without color distance, quiet zone and rotation Check properties):

Find barcodes (all parameters).kfpx

Barcode Reference Manual

If you want more information about barcodes and matrix codes in general, please download the "Barcode Reference Manual":

Barcode_Reference_EN_2015-10-30.pdf

24. Context aware object detection: Sifter

24.1 Beyond classic preflighting: Context aware object detection (Sifter)

Classic preflighting focuses on properties of objects as they happen to exist on a given page. Sometimes, simple geometric constraints – e.g. whether an object is located inside the TrimBox or BleedBox – are also taken into account. Using classic preflighting many potential problems can be detected before they begin to cause delays or even actual damage in production in the form of additional waste or lost revenue.

Especially in a world of more and more digital printing, ever tighter deadlines, increased levels of automation, more demanding print buyers and fast growing challenges in more and more complex production setups, additional preflighting (and processing) needs arise.

One of the weaknesses of classic preflighting is the sometimes significant number of false positives: flagging something that ***could be*** a problem ***but actually is not a problem*** in the given constellation of content on a page. For example, it can be essential to ensure that small black text is set to overprint, to compensate for registration issues on the printing press. Nonetheless – such a need to overprint small black text is only a relevant requirement if there is any contet below the small black text.

In other cases it is not sufficient to limit a requirement to objects inside the TrimBox or BleedBox – in packaging and label printing the final printed objects are typically not rectangular anymore, rather, they can have any shape (also including holes in such shapes). Here, it is critical to limit certain checks to objects inside the cut line, or to ensure that text mandated by food related or other regulations is not too close to the cut line as it could be cut off if the cutting process happens to be slightly off.

Some usage examples

The examples below are meant to give some ideas how Context aware object detection could be put to use. In some cas-

es, specialized properties make it very easy to set up a check accordingly. In other, more demanding cases, it may be necessary to make use of the general purpose "Context aware object detection (Sifter)" property, with full control over each and every aspect of the property, but also more difficult to set up.

An aspect not be missed is the fact that checks making use of Context aware object detection can be used as filters in a number of fixups.

Only enforce overprint of small black text if there are actually objects behind it

Small black text will often be set to overprint to accommodate effects of slight mis-registration in the printing process: if such small text where to knock out content behind it, small but very noticeable white lines might show up around the text.

Many preflight configurations will flag small text that is not set to overprint. In many cases though such text does not have any content underneath it – it then simply does not matter whether that small text is set to overprint or to knock-out.

By using Context aware object detection it is possible to only flag small text not set to overprint in those cases where it does have visible content underneath it.

Detect whether a dieline is fully or partially covered by other objects

Cutlines on a PDF page are usually constructed in the form of a simple line using a spot color with a certain name (e.g. "Cutline"). Cutting devices can extract such cutlines from a PDF file and use them for carrying out the cutting task.

In order to avoid a mismatch between what looks like the cutline when looking at a PDF page and the actual cutline, it is essential that no part of the cutline is covered by other objects.

A Context aware object detection based check can be used to look for objects that intersect with a line object using a spot color "Cutline", and are on top of that line object (if they are below, the intersection does not have any negative impact on the cutline).

Find and reveal invisible objects that are covered by other objects

Graphics objects on a page can be invisible for a number of reasons:

- they are outside the page area
- they are clipped by means of a clipping path
- they are covered by some other opaque object
- or any combination of the above, possibly affecting different parts of a graphics object (i.e. a graphics object can be come effectively invisible because different parts of it are subject to one of the above listed effects)

In some cases it can be especially interesting to check for objects that are covered by other objects, as someone may have inadvertently moved them to the background.

Using Context aware object detection it is possible to

- check for grapbics objects that are invisible because they are covered by other opaque objects
- after carrying out the check, use the "Snap" feature to revew each of these invisible graphics objects
- for diagnostic purposes: remove all visible objects such that the invisible objects (covered by other opaque objects) become visible

Make sure all text objects are above all other objects

In workflows that wish to repurpose PDFs by keeping the graphical content but replacing text content with updated or translated versions, it is essential that removing existing or inserting modified text does not have a negative impact on the appearance of the graphical content.

An important aspect in this regard is to make sure all text objects are on top of all other objects. Context aware object de-

tection can be used to ascertain that no visible objects are on top of any text objects.

Where repurposing workflows only exchange black text – so they only have to remake the black printing form – it is of course possible to limit checking to black text.

In cases where text is not already in front of all other content on a page, the following option may be useful.

Determine whether an object can be brought to the front without impacting the visual appearance of the page

There are two methods to determine whether bringing text objects to the front has any downside effect:

- use Context aware object detection to find all text objects that have visible content above them
- execute a profile containing the "Bring to front" fixup for text objects, and also the "Result file different from original (visual comparison)" check; whenever bringing text to the front causes a change in the visual appearance of a page, a warning will be issued

Remove invisible objects (regardless whether outside of page area, covered or clipped)

As described above, objects can be invisible for several reasons (and even combinations of reasons). A Context aware object detection based check can be used in the "Remove objects" fixup to remove invisible objects.

Find objects too close to the cutline

Cutlines in many cases are non-rectangular, making it difficult to use classic preflighting to find objects that are a bit too close to the cutline. As a cutline can be identified by the spot color name it is using, or by the fact that it is on a certain layer (for example, according to the Processing Steps specification as defined in ISO 19593-1), it possible to detect its presence on a PDF page.

Using Context aware object detection it is possible to check for all visible objects on a page whether they are closer to the cutline than allowed by a user configurable threshold value (e.g. 3mm or 10pt). This can be done with different threshold values from the inside and from the outside of a cutline's area.

Remove objects outside the bleed area of an arbitrarily shaped label

During creation and review of print files for label printing, extensive amounts of content can be present that will not ultimately be printed. It will usually exist outside the bleed area of the label, where the bleed area is the area defined by the cutline for the label plus an addditional 3mm around the area defined by the cutline.

Being able to completely remove graphics objects outside the bleed area of a label can help clean up the production file. Where certain content – such as instructions or a company logo – is to be kept regardless, a Context aware object detection check can be set up accordingly. Using the check as a filter in the "Remove objects" fixup will carry out the respective removal of extraneous objects.

Conformance with Processing Steps specification (ISO 19593-1)

The "Processing Steps" specification is an ISO standard that mandates that graphics objects not actually intended for printing in the production printing process are put on certain layers, where such layers have certain metadata information, and also adhere to certain rules. Typical examples of 'non-print' objects are cutlines, folding lines, indication of punch or drilling holes, glue areas, indication of dimensions, instructional text, and so forth.

The main goal of the The "Processing Steps" specification is to reduce production errors and facilitate exchange and use of non-print objects, for example for automated cutting processes.

Numerous provisions in the "Processing Steps" specification take relationships between graphics objects into account. For

example, in a PDF file conforming to the "Processing Steps" specification, it is prohibited to have any graphics object above a cutline intersect with that cutline.

Based on Context aware object detection, a full conformance check against the "Processing Steps" specification can be carrried out using pdfToolbox.

Context awareness

Linking classic preflighting to the context in which graphics objects exist on a page adds a new and essential dimension to preflighting. Context does exist in a number of ways:

- *stacking order*: whether a graphics objects is drawn before or after other graphics objects; a graphics object can only be obstructed or covered up by other graphics objects if those are painted after the graphics object at hand
- *intersection*: whether a graphics object shares a certain area of the page with other graphics objects, or an arbitrary area on the page, where such an area could defined by means of a cutline, the extent of a white background necessary for printing on transparent foils or a "glue" area to be kept free of printing ink
- *insideness*: whether a graphics object is inside or outside other graphics objects, or an arbitrarily shaped area on the page
- *proximity*: how close a graphics object is to other graphics objects or the border of an arbitrarily shaped area on the page
- *clipping*: whether a graphics object is partially or fully clipped
- *obliteration*: whether a grapics object is partially or fully covered by other opaque graphics objects

Depending on the type of content to be preflighted, different aspects of context can play a role, in varying combinations. In addition, detecting graphics objects based on their context can also be a powerful filtering mechanism: whether this is about removing clipped or obliterated content, limiting modification of small black text to occurrences where it actually exists on top of other graphics objects, or only applying color conversion to graphics objects inside cut lines, and so forth.

Sifter: Context aware preflighting at your finger tips

Sifter – a shorthand for referring to the Context aware object detection technology – has been developed by callas software in an extensive research and development process. Already in 2014 a research project was carried out in cooperation with the Beuth University of Applied Sciences in Berlin. Among other activities, the Master Thesis "Context Aware Preflighting in Packaging Printing" by Nils Niggemann contributed important insights to the then *status quo* of advanced preflighting needs in the packaging industry in Europe and North America.

While experienced partitioners participating in Niggemann's research struggled to think outside of the box and rather preferred to stick to the tools they had, and despite the glaring limitations of those tools, it became apparent, that especially in packaging and label printing, and more so as this industry is moving towards much faster paced digital printing processes, flexible and fine grained context aware preflighting would soon become a critical requirement.

A major concern in early implementation stages was performance: complex packaging designs – and more so imposed sheets of such designs – can easily contain tens or even hundreds of thousands of individual grapics objects. Still, the execution of a complex context aware check on a typical page should typically take seconds, not minutes.

From continued research and implementation cycles, callas software's Sifter technology emerged as the engine, that provides extensive context aware checking capabilities at the speed mandated by today's workflows. Sifter is an integral part of callas pdfToolbox starting with version 10, and is equally available in the desktop, server, command line and SDK variants of callas pdfToolbox on the Apple MacOS, Microsoft Windows and Linux operating systems.

24.2 Finding the right "Context aware object detection" property

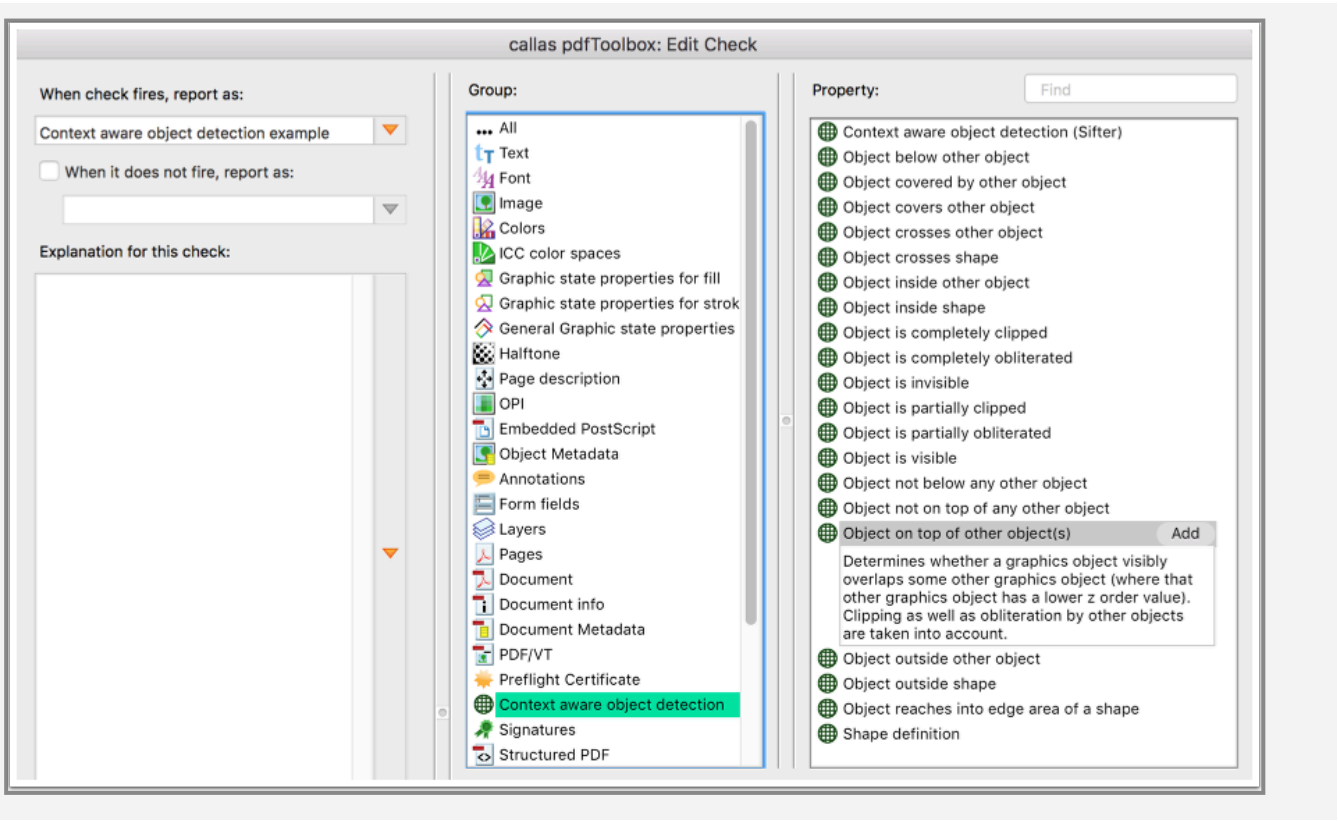
Checks based on "Context aware object detection" properties can be configured in a various ways.

1. There around 20 relatively simple variants that address specific use cases – if they match a certain requirement, it's best to start from such an easy to use variant.
2. There is master version of "Context aware object detection", which is called like the Group itself: "Context aware object detection (Sifter)". It offers a highly flexible but also possibly challenging approach to setting up a context aware object detection check.
3. Last but not least there is the "Shapes" property, which can only be used in the context of "Context aware object detection" checks: such a Shape property make it possible to flexibly define areas – for example the area defined by the outer border of all cut lines on a page – against which to detect certain objects.

The remainder of this article gives an overview of the available "Context aware object detection" properties, followed by one example that demonstrates how to set up a check based on the "Object reaches into edge area of a shape" property.

Where to find the "Context aware object detection" properties

When creating a new check, under "Group" look for the "Context aware object detection" entry. Upon selecting that entry, under "Property" all the properties belonging to the "context aware object detection" group will be listed.



List of special purpose context aware object detection options

- Proximity:
  - Object reaches into edge area of a shape:
    Determines whether the extent of a graphics object ends inside the edge area of a shape.
  - Object crosses shape:
    Determines whether the edge of a graphics object visibly crosses the edge of the area defined by a shape. Clipping as well as obliteration by other objects are taken into account, whereas z order is not taken into account.
- Above versus below:
  - Object on top of other object(s):
    Determines whether a graphics object visibly overlaps some other graphics object (where that other graphics object has a lower z order value). Clipping as well as obliteration by other objects are taken into account.

- **Object not on top of any other object:**
  Finds graphics objects that do not visibly overlaps any other graphics object (where that other graphics object has a lower z order value) on the page. Clipping as well as obliteration by other objects are taken into account.
- **Object covers other object:**
  Determines whether a graphics object visibly covers some other graphics object (where that other graphics object has a lower z order value). Clipping as well as obliteration by other objects are taken into account. Depending on applicable overprint and transparency properties, the object below may be visible or not.
- **Object below other object: Object below other object:**
  Determines whether a graphics object is visibly overlapped by some other graphics object (where that other graphics object has a higher z order value). Clipping as well as obliteration by other objects are taken into account.
- **Object not below any other object:**
  Finds graphics objects that are not visibly overlapped by any other graphics object (where that other graphics object has a higher z order value). Clipping as well as obliteration by other objects are taken into account.
- **Object covered by other object:**
  Determines whether a graphics object is visibly covered by some other graphics object (where that other graphics object has a higher z order value). Clipping as well as obliteration by other objects are taken into account. Depending on applicable overprint and transparency properties, the covered object may be visible or not.

- **Inside versus outside:**
  - **Object inside other object:**
    Determines whether the extent of a graphics object visibly lies completely inside the area occupied by some other graphics object. Clipping as well as obliteration by other objects are taken into account, whereas z order is not taken into account.
  - **Object inside shape:**
    Determines whether the extent of a graphics object visibly lies completely inside the area defined by a

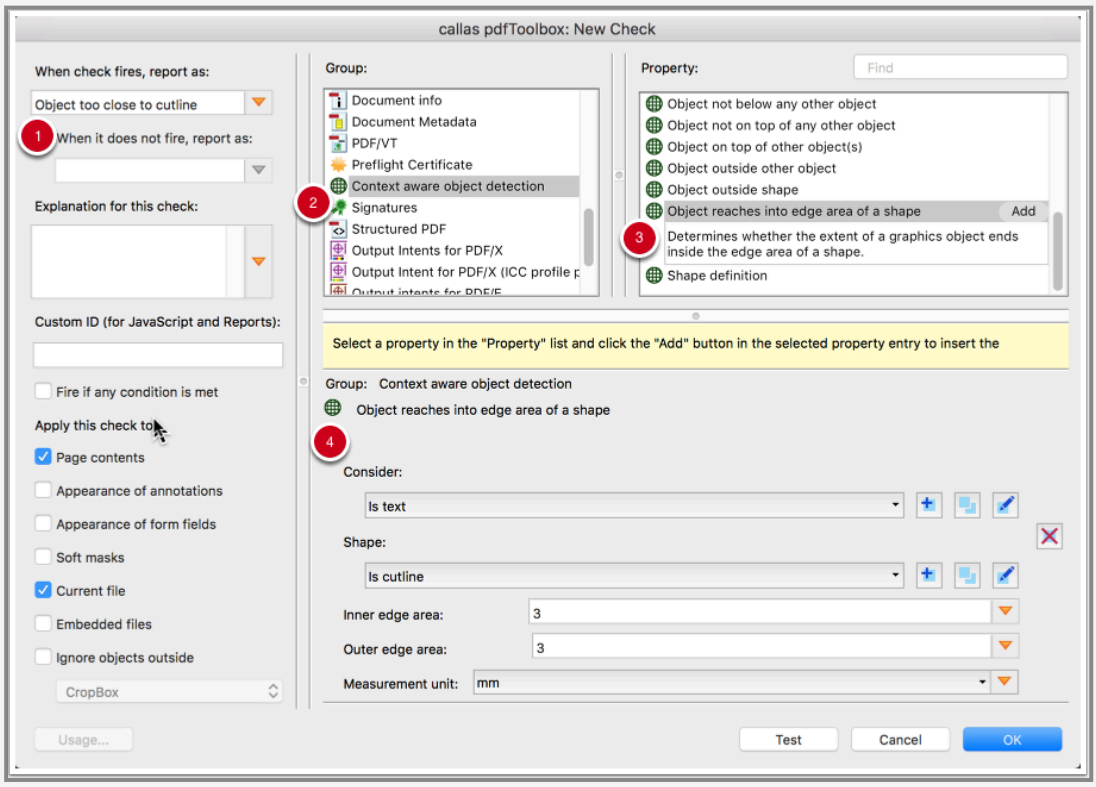
shape. Clipping as well as obliteration by other objects are taken into account, whereas z order is not taken into account.

- **Object outside other object:**
  Determines whether the extent of a graphics object visibly lies completely outside the area occupied by some other graphics object. Clipping as well as obliteration by other objects are taken into account, whereas z order is not taken into account.
- **Object outside shape:**
  Determines whether the extent of a graphics object visibly lies completely outside the area defined by a shape. Clipping as well as obliteration by other objects are taken into account, whereas z order is not taken into account.
- **Object crosses other object:**
  Determines whether the edge of a graphics object visibly crosses the edge of some other graphics object. Clipping as well as obliteration by other objects are taken into account, whereas z order is not taken into account.

- **Visible, partially visible or invisible**
  - **Object is invisible:**
    Determines the visibility of a graphics object. An object may be invisible due to clipping – through explicit clipping paths, or due to CropBox and MediaBox – or obliteration by other objects that are opaque.
  - **Object is visible:**
    Determines the visibility of a graphics object. An object may be invisible due to clipping – through explicit clipping paths, or due to CropBox and MediaBox – or obliteration by other objects that are opaque.
  - **Object is partially obliterated:**
    Determines whether a part of a graphics object is obliterated by other opaque graphics objects. Completely obliterated graphics objects are not found by this property.
  - **Object is completely obliterated:**
    Determines the visibility, or lack thereof, of a graphics object caused by obliteration. An object may be invisible due to clipping – through explicit clipping paths, or due to CropBox and MediaBox – or obliteration by other objects that are opaque.

- **Object is partially clipped:**
  Determines whether a part of a graphics object is clipped (through explicit clipping paths, or due to CropBox and MediaBox). Completely clipped graphics objects are not found by this property.
- **Object is completely clipped:**
  Determines the visibility, or lack thereof, of a graphics object caused by clipping. An object may be invisible due to clipping – through explicit clipping paths, or due to CropBox and MediaBox – or obliteration by other objects that are opaque.

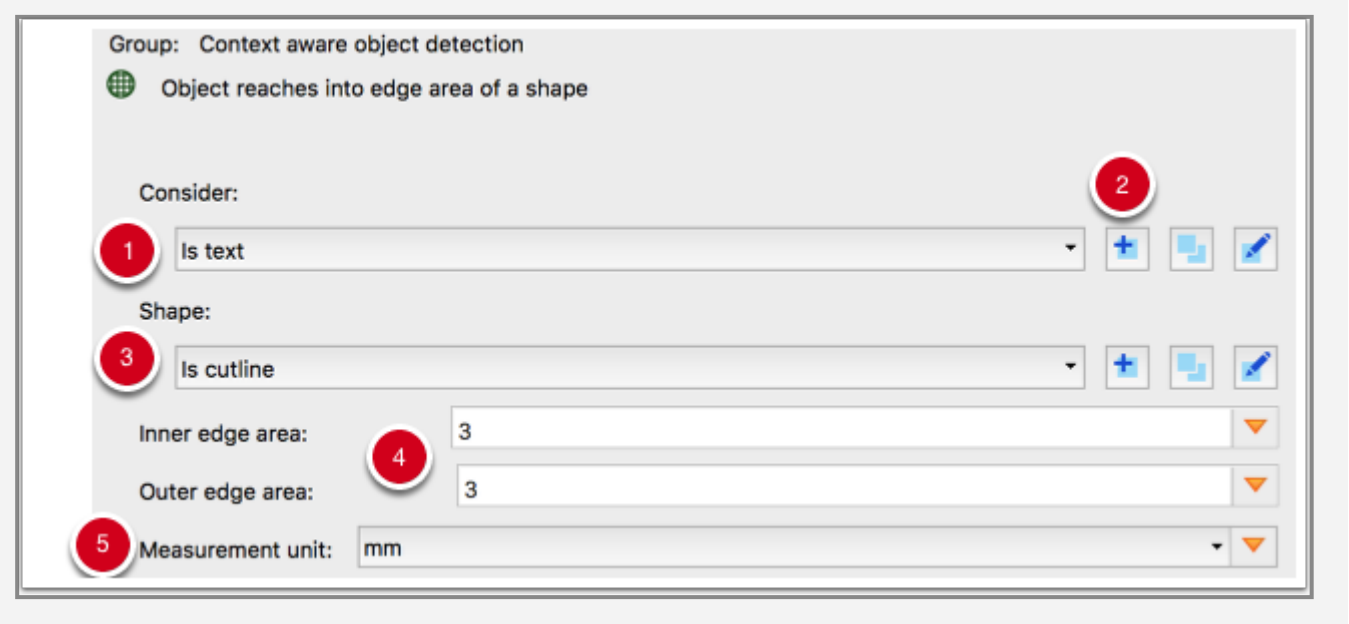
Example: Configuring the «Object reaches into edge area of a shape» property



Create a new check using the fly out menu in the upper right of the "Checks" window (open from "Tools" menu via "Checks" menu entry). In the "New Check" dialog, start configuring the context aware object detection check "Object too close to cutline":

1. Enter a suitable name for the check

2. Select the "Context aware object detection" group in the "Group" list.
3. Next select the "Object reaches into edge area of a shape" property in "Property" list, then click the "Add" button in that list entry.
4. A list of configuration options will show up below the "Group" and "Property" lists.



1. Select a check that finds the objects you are interested in. In this example, we want to check for text objects that may be too close to a cutline.
2. If there is no "Is text" check yet, simply create a new one using the "+" button to the right of the popup menu with the list of already existing checks.
3. Choose a suitable check in the second popup menu that will find lines using a spot color "Cutline". If there is not such a check yet, simply create a new one using the "+" button to the right of the popup menu with the list of already existing checks.
4. Next, enter suitable values to use when checking the distance of above configured text objects against any line colored with the spot color "Cutline". Due to the fact that for any object only the area that it actually covers is taken into account, there is not a lot of area inside the line (i.e. the inside area essentially just has an extent equal to the line width) so more important is the outside edge of the line: fill in a number that matches production requirements.

5. Set the measurement unit for the "Inner edge area" and the "Outer edge area" to whatever may be most convenient. Supported measurement units are millimeters (mm), point (pt) and inch (in).

24.3 Proximity: Object reaches into edge area of a shape

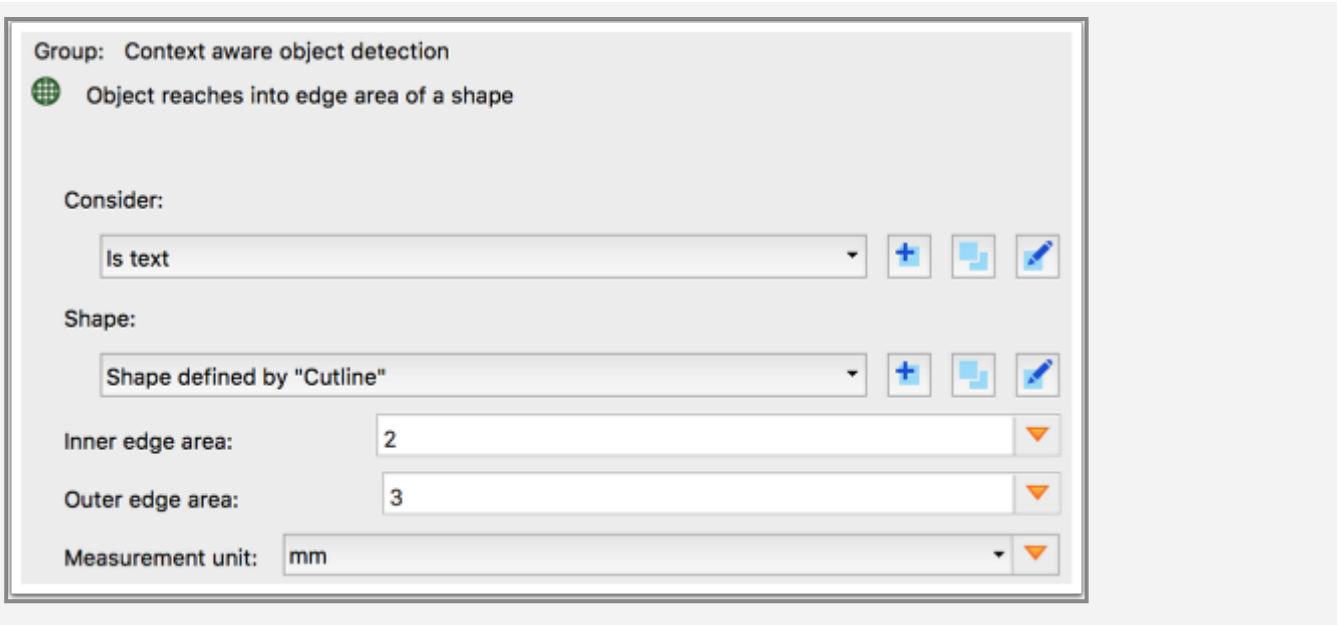
The property "Object reaches into edge area of a shape" is based on the concept that graphics objects on a PDF page can be checked whether they reach into the 'safety zone' of a shape. The shape is defined by means of a Shape property (for more details see the article "Shapes" property for use in "Context aware object detection" checks).

In the example shown further below, this shape is derived from a cutline that is expected to be present in the PDF file to be examined. Using the outer border of such a cutline it is possible to represent even relatively complex setups.

The 'safety zone' for the cutline is then established by two threshold values, one for objects inside the cutline, and the other one for objects outside of the cutline.

Last but not least it needs to be determined which objects to check against the safety zone of the cutline. In the example shown below, text objects are checked.

The configuration options for a check based on the "Object reaches into edge area of a shape" property can be seen below:



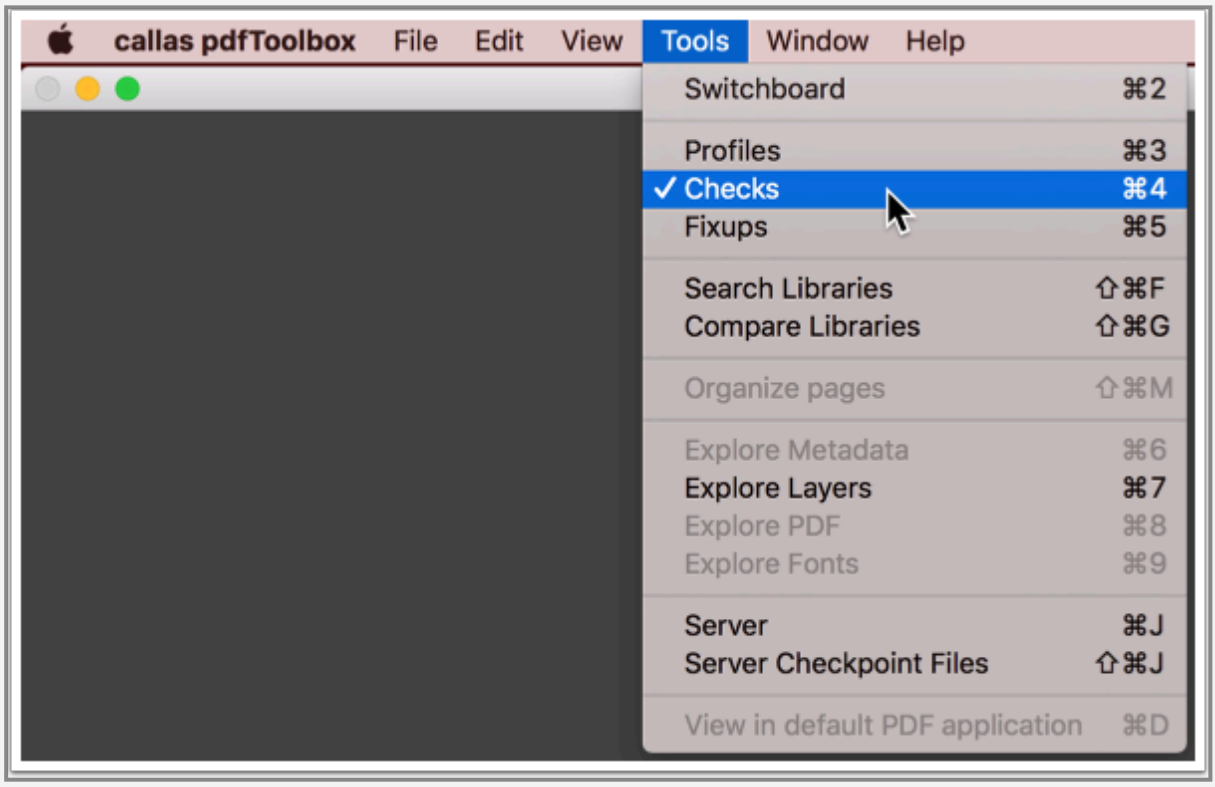
Sample files

The example below makes use of the check *Text too close to "Cutline".kfpx* and the PDF file *Yummy Pumpkin Yoghurt.pdf* which are available for download:

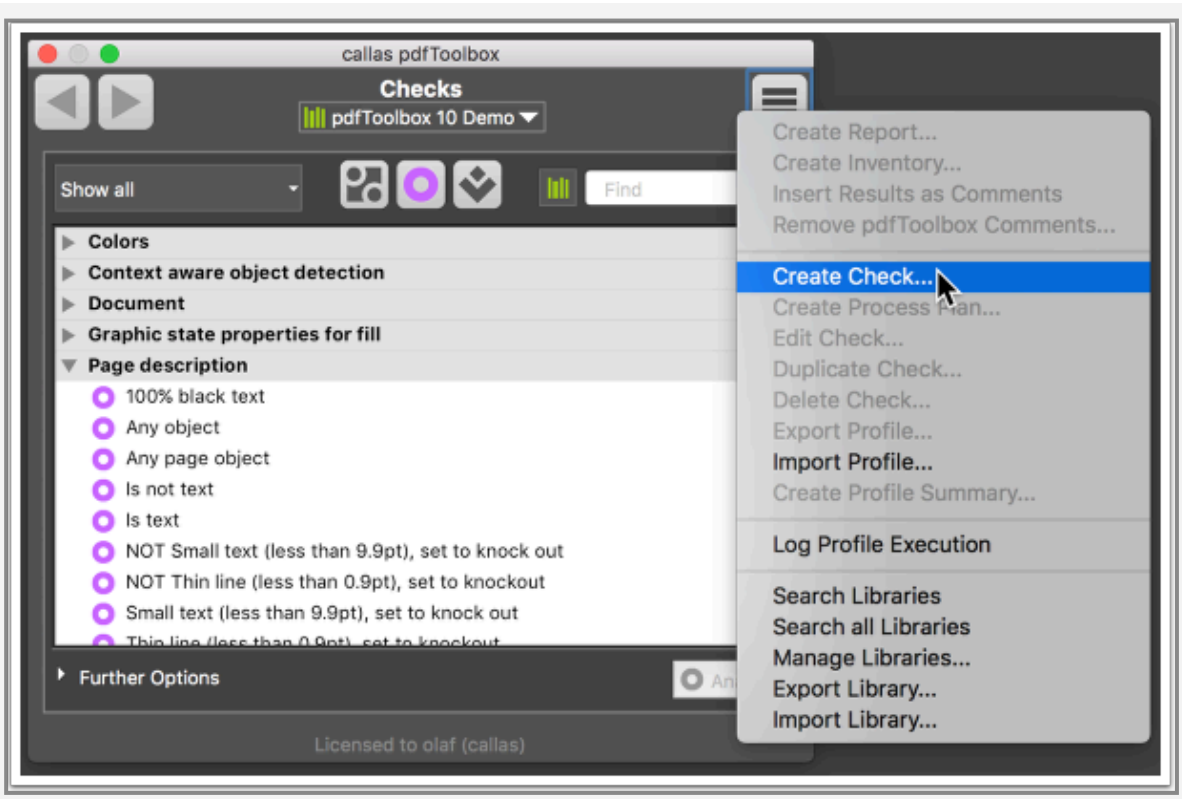
Text_too_close_to__Cutline_.kfpx

Yummy_Pumpkin_Yoghurt.pdf

Create a check based on the "Object reaches into edge area of a shape" property



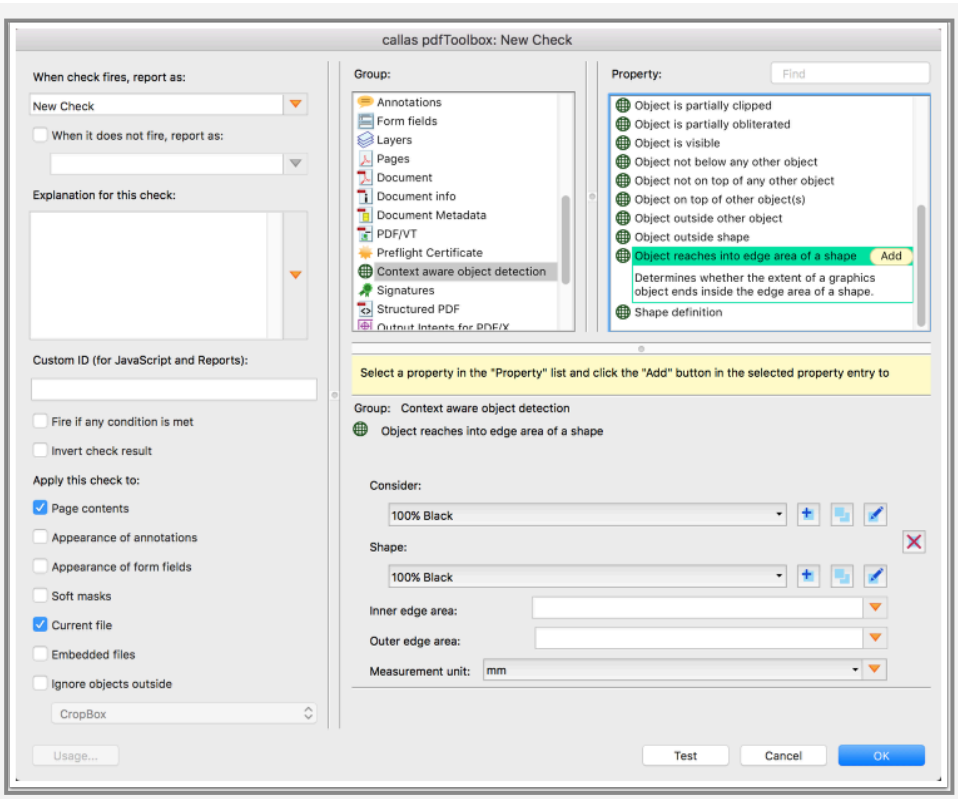
Inside the "Checks" window, select the "Create Check..." entry from the fly out menu.



The "Edit Check" window will open. In the "Group" list select the "Context aware object detection" entry, then in the "Property" list select the "Object reaches into edge area of a shape".

Click on the small "Add" button in the "Object reaches into edge area of a shape".

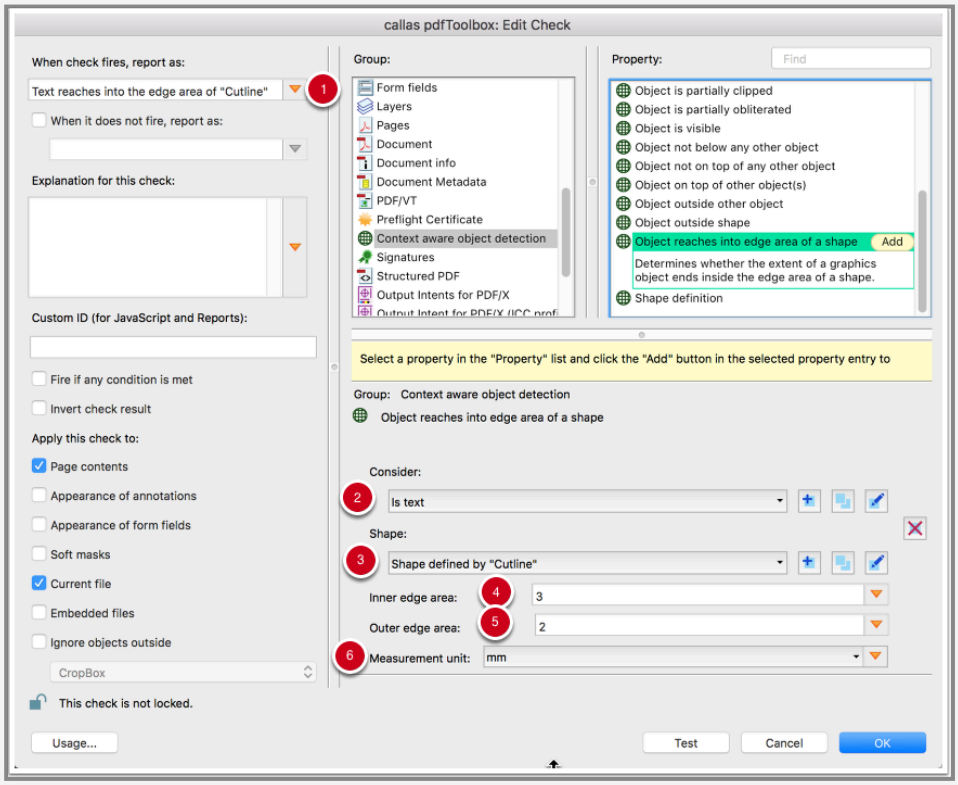
Below the "Group" and "Property" lists, a configuration area for the "Object reaches into edge area of a shape" property gets inserted, ready to be configured.



Make sure to fill out all applicable fields:

1. choose a suitable name for the new check
2. select an already existing check (or create a new one) that defines for which objects to check whether they are reaching into the edge area of a shape
3. select an already existing shape (or create a new one) that defines a shape based on the page content or objects that represent the cutline (of course any other type of content or objects could be used as well)
4. set the value for the minimal required distance for content reaching towards the "Cutline" shape's edge from the ***inside*** of the shape
5. set the value for the minimal required distance for content reaching towards the "Cutline" shape's edge from the ***outside*** of the shape
6. adjust the measurement unit as necessary; possible values are mm, pt and inch.

Make sure to save your new check by clicking the OK button. You can modify the check any time.



Testing whether the check works as expected

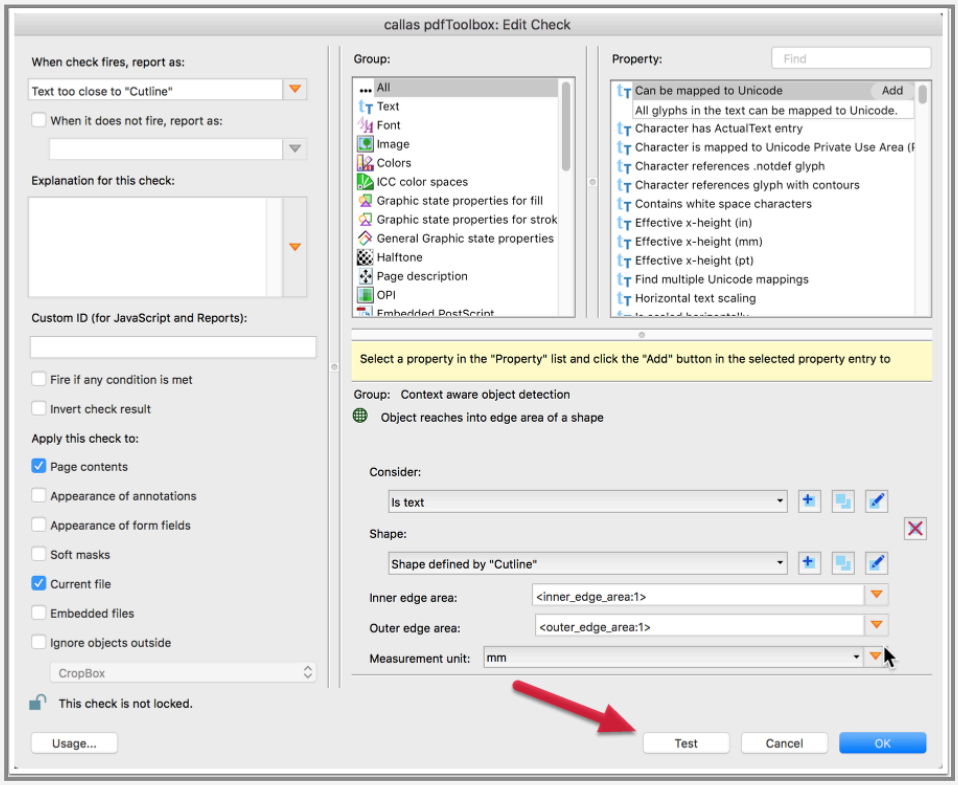
While editing a new or existing check there is an easy way to test whether the check actually does what it is intended to do.

Please install the attached check *Text too close to "Cutline".kfpx* and open the attached example file *Yummy Pumpkin Yoghurt.pdf* in order to follow the exercise below.

Open "Edit Check" window for the *Text too close to "Cutline"* check and click on the "**Test**" button.

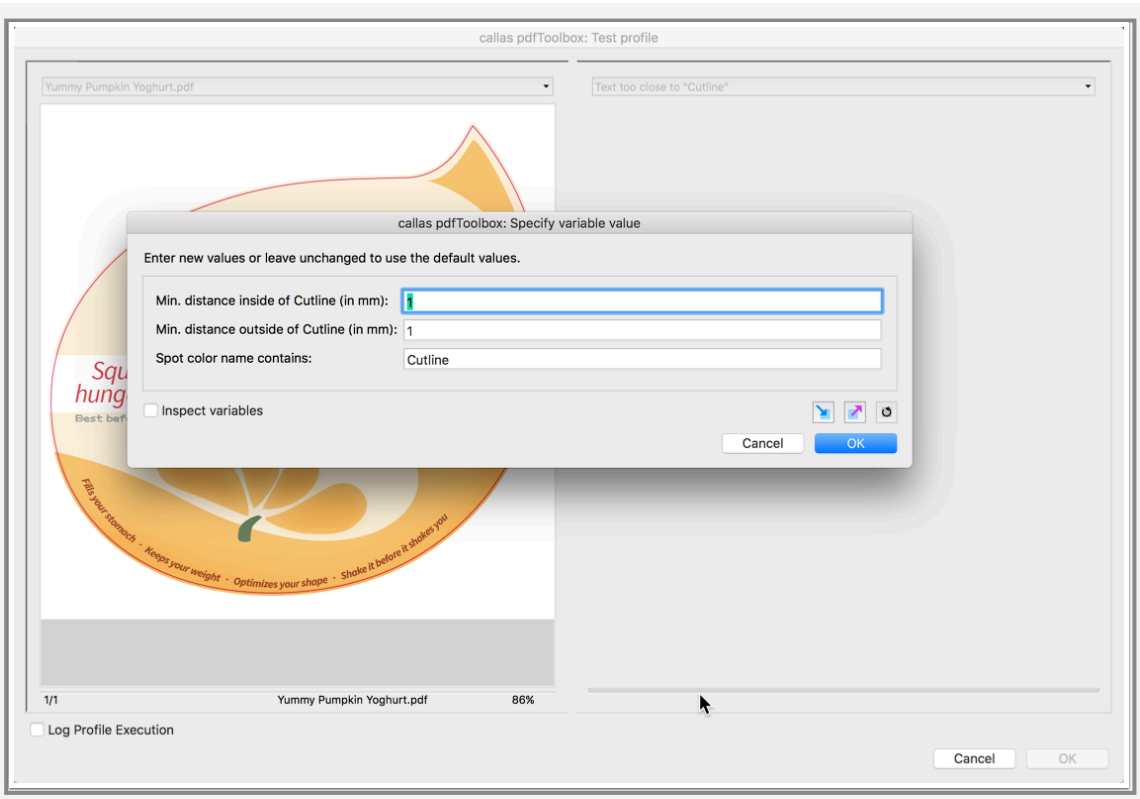


In order to learn more about how the *Test mode* can speed up construction and adjustments of checks, but also profiles, process plans or fixups, please check out the chapter Test mode.



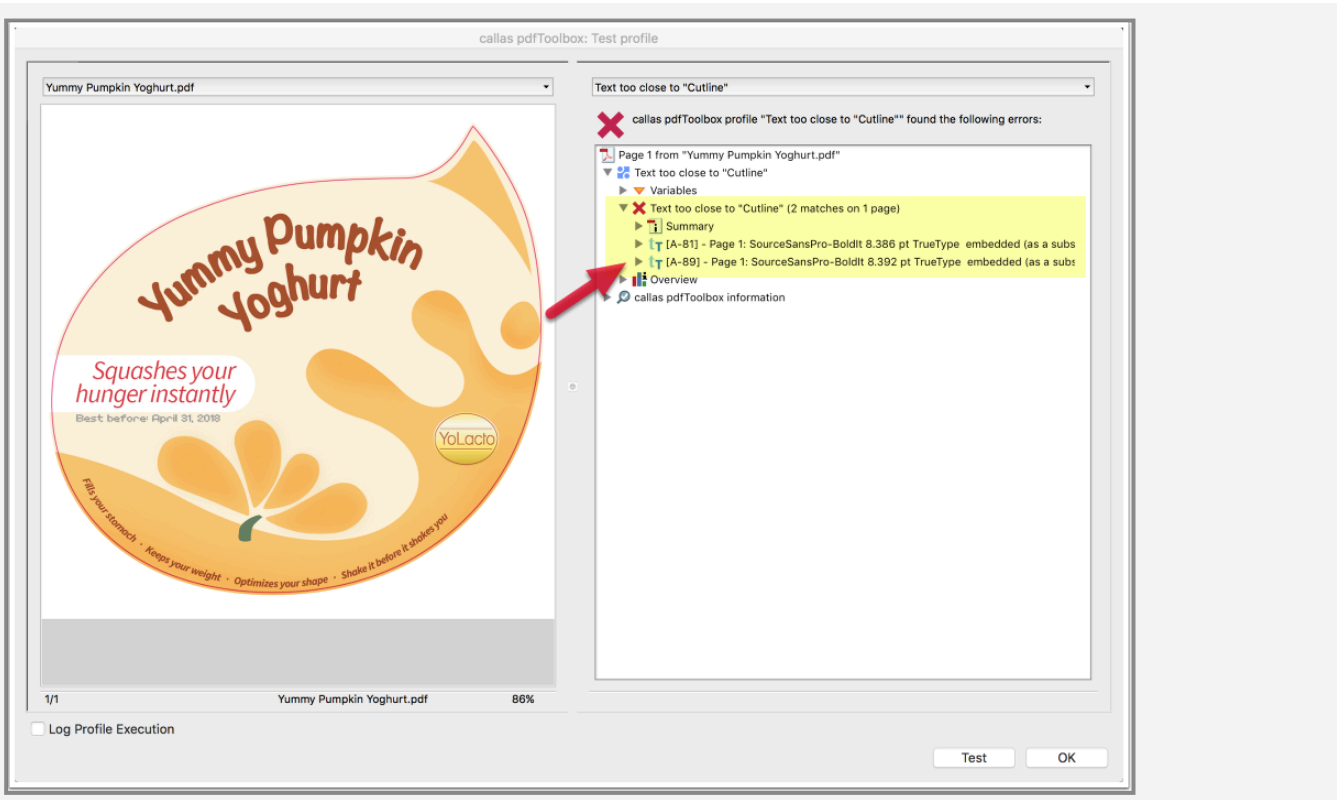
A *Test profile* window will open, with a dialog window on top of it with three fields where information can be entered.

These three field are shown, because the check in this example makes use of variables – variables can represent values that can be modified before executing a check. It is also acceptable to leave the values unmodified. In any case, click **OK** to trigger execution of the check.

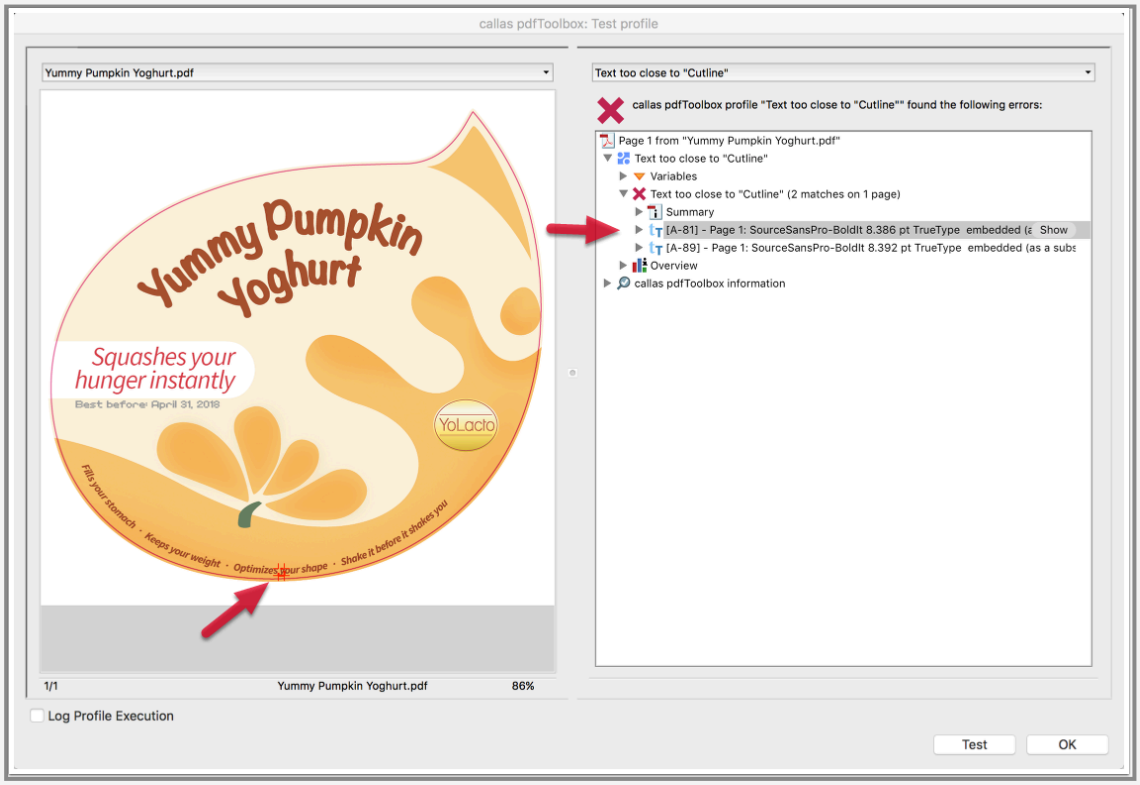


The Test profile window will now display the file ***Yummy Pumpkin Yoghurt.pdf*** in the left half of its window, and the results from running the check on the right side.

By double clicking on the entries below the line saying *Text too close to "Cutline" (2 matches)* the respective objects can be highlighted on the page.

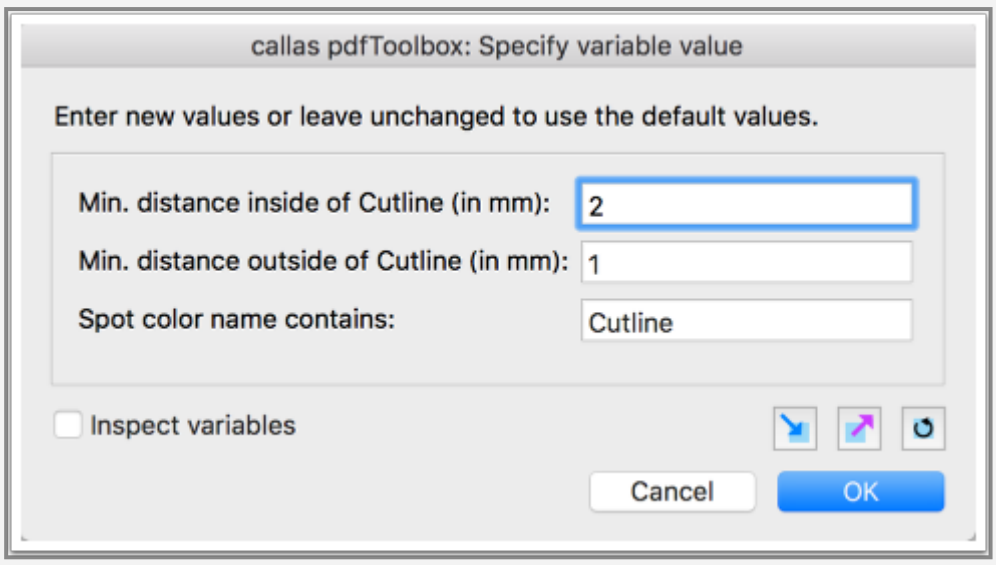


Double clicking on the first entry will highlight the character 'y' in the word 'your' towards the bottom of the label in the example PDF file.

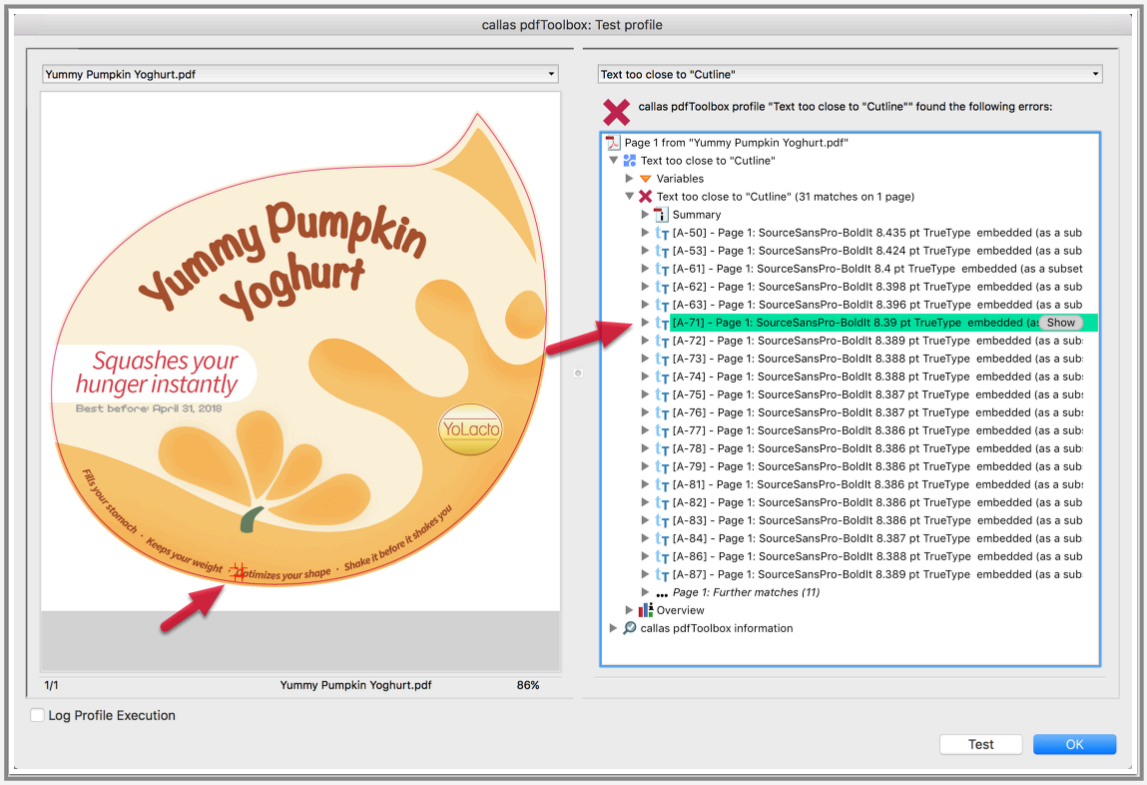


In order to test the check with different values, simply click on the "Test" button in the "Test profile" window.

By entering a different value for the inside distance – 2mm instead of 1mm – more objects will be found.



As there are objects that are less than 2 mm, but more than 1 mm, away from the cutline, the check now finds 31 objects instead of just two.



24.4 Proximity: Object crosses shape

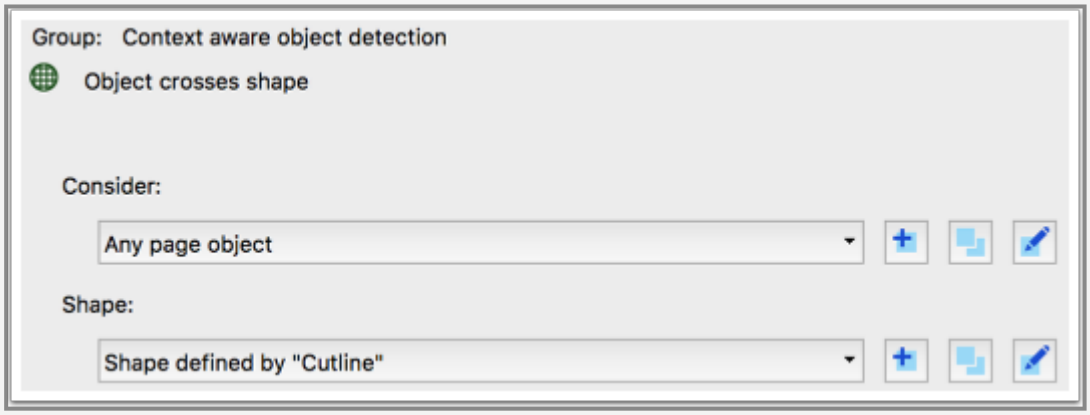
The property "Object crosses shape" detects graphics objects whose border crosses the border of a shape.

The shape is defined by means of a Shape property (for more details see the article "Shapes" property for use in "Context aware object detection" checks).

In the example shown further below, this shape is derived from a cutline that is expected to be present in the PDF file to be examined. Using the outer border of such a cutline it is possible to represent even relatively complex setups.

For each object it is checked whether its border crosses the border defined by the shape's area.

The configuration options for a check based on the "Object crosses shape" property can be seen below:



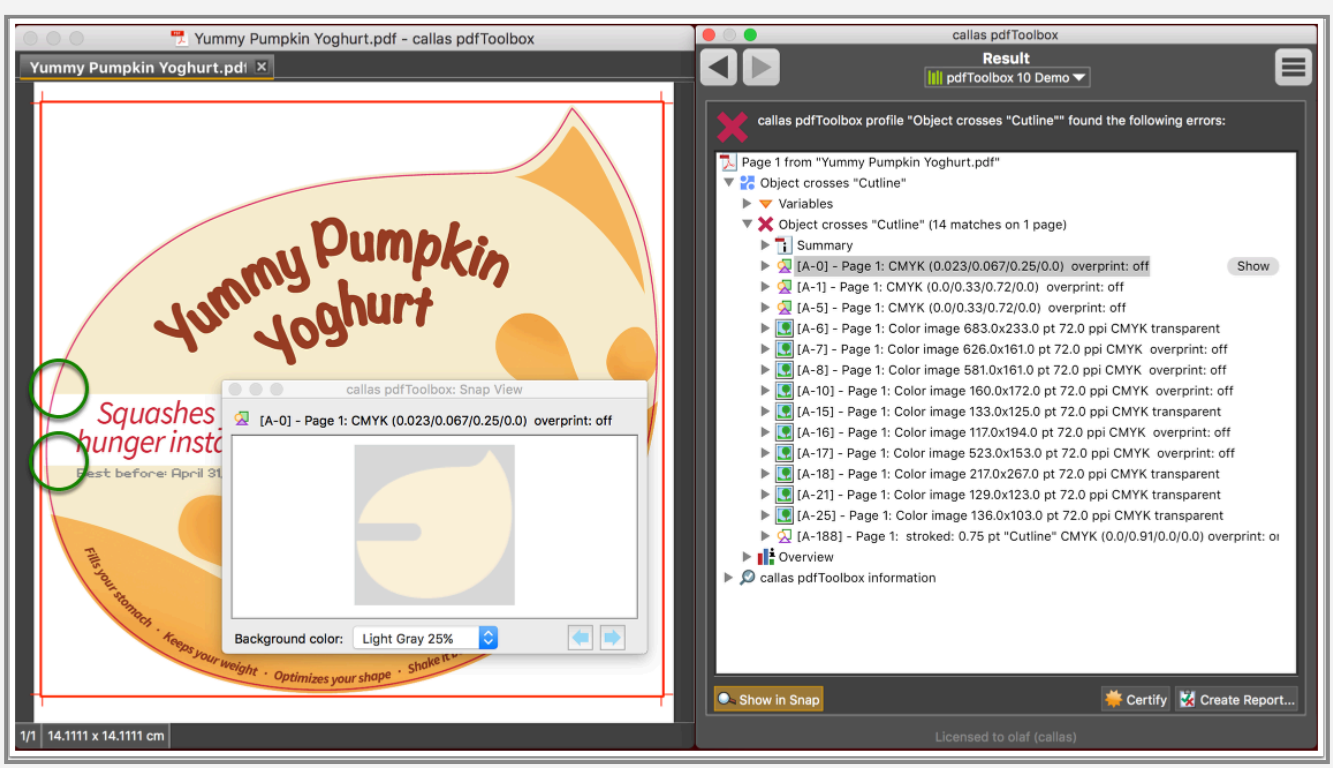
Sample files

The example below makes use of the check *Object crosses "Cutline".kfpx* and the PDF file *Yummy Pumpkin Yoghurt.pdf* which are available for download:

Object_crosses__Cutline_.kfpx

Yummy_Pumpkin_Yoghurt.pdf

Running the check ***Object crosses "Cutline"*** detects all objects whose border crosses the border of the area defined by the cutline. For example, the border of the light-yellow background is mostly just outside the cutline, but on the left its borders is retracted towards the inside of the label (possibly to reveal the aluminium foil surface below it to make the area look silver), thus crossing the cutline.

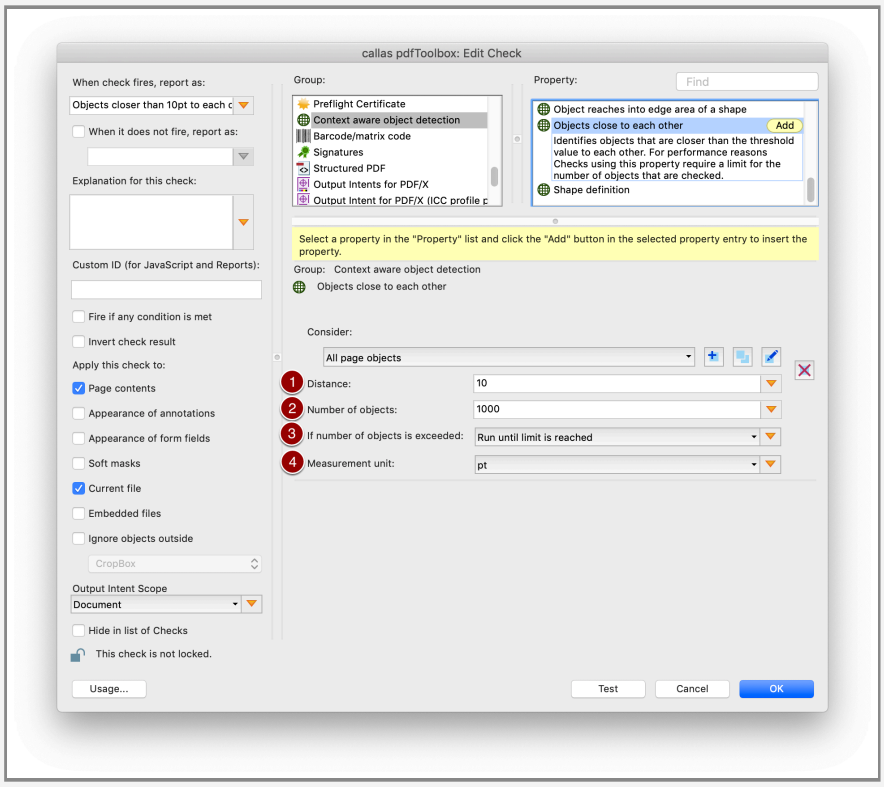



Squ
hung
Best bef

24.5 Proximity: Objects close to each other

The property "Objects close to each other" is based on the concept that graphics objects on a PDF page can be checked whether they are too close to one another.

The configuration options for a check based on the "Objects close to each other" property can be seen below:



1. Distance: Distance to be checked between objects
2. Number of objects: Limit to the number of objects to be checked (1000 objects to be checked in the example above)
3. If number of objects is exceeded: For performance reasons, test conditions can be applied:
    1. Run until the limit is reached
    2. Do not run if the test runs for the specified number of objects
4. Measurement unit: mm, pt, inch

Sample files

The example below makes use of the check ***Objects closer than 10pt to each other.kfpx*** and the PDF file ***Objects_close_to_each_other.pdf*** which are available for download:



Objects_closer_than_10pt_to_each_other.kfpx



Objects_close_to_each_other.pdf

Testing whether the check works as expected

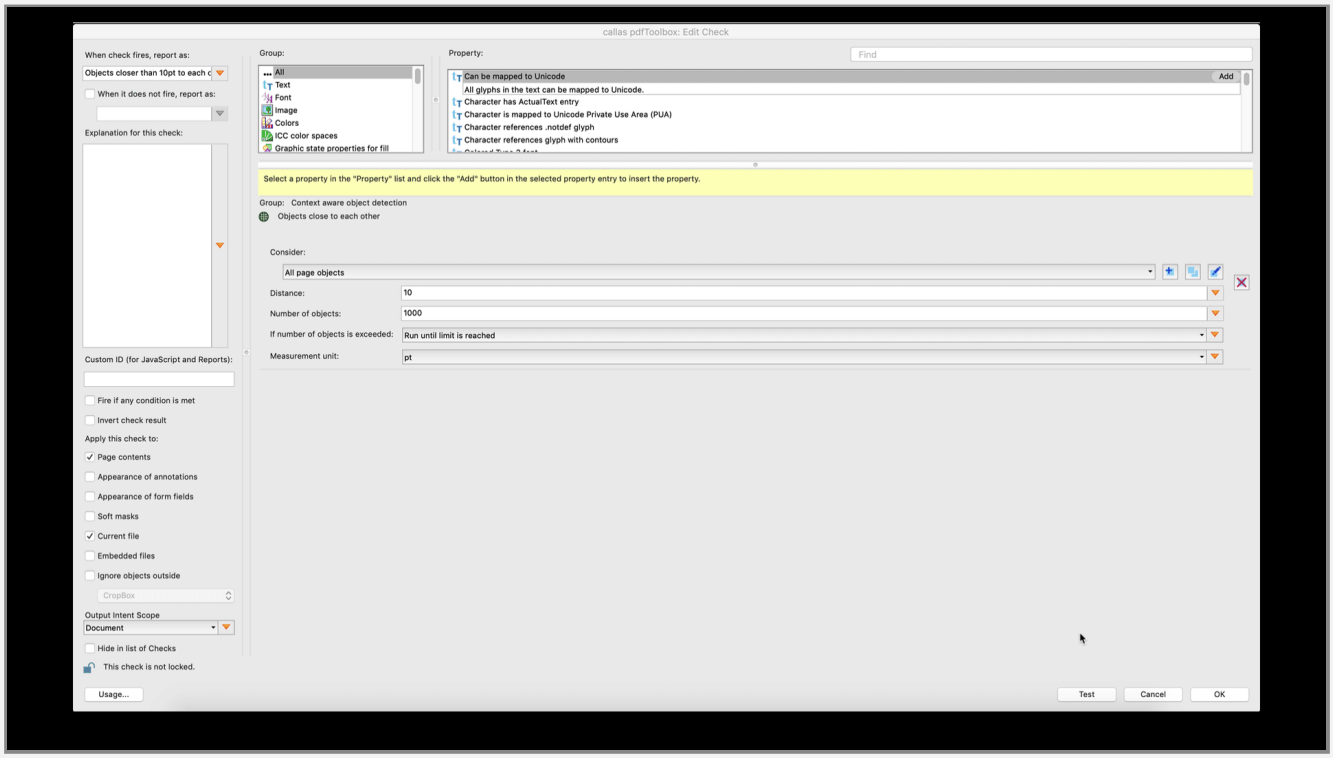
While editing a new or existing check there is an easy way to test whether the check actually does what it is intended to do.

Please install the attached check ***Objects closer than 10pt to each other.kfpx*** and open the attached example file ***Objects_close_to_each_other.pdf*** in order to follow the test.

Open "Edit Check" window for the ***Objects closer than 10pt to each other.kfpx*** check and click on the "**Test**" button.



In order to learn more about how the ***Test mode*** can speed up construction and adjustments of Checks, but also Profiles, Process Plans or Fixups, please check out the chapter Test mode.



The Test profile window will now display the file *Objects_close_to_each_other.pdf* in the left half of its window, and the results from running the Check on the right side.

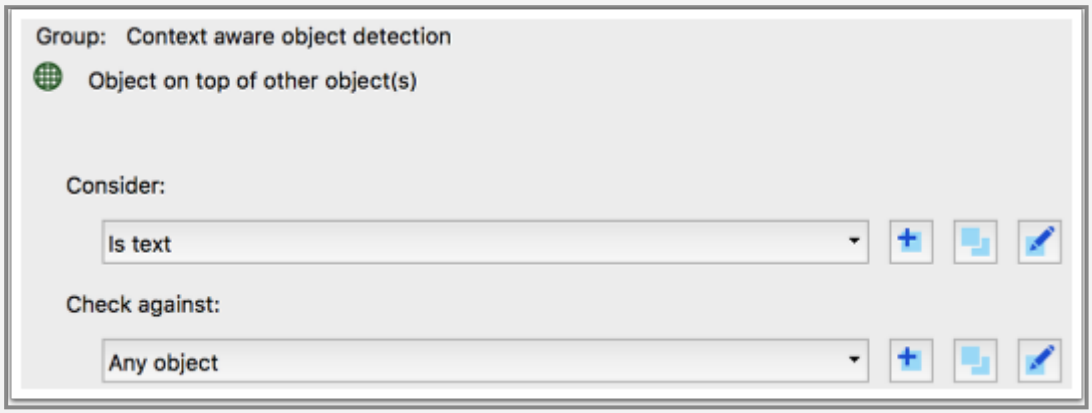
By double clicking on the entries below the line saying ***Objects closer than 10pt to each other (11 matches),*** the respective objects can be highlighted on the page.

24.6 Above versus below: Object on top of other object(s)

The property "Object on top of other object(s)" detects whether an object visibly overlaps at least one other object.

The configuration options for a check based on the "Object on top of other object(s)" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined which objects from this set overlap at least one other object that belongs to those objects defined under "Check against".



Sample files

The example and exercise below make use of the check *Text object on top of other objects.kfpx* and the PDF file *Yummy Pumpkin Yoghurt.pdf* which are available for download:

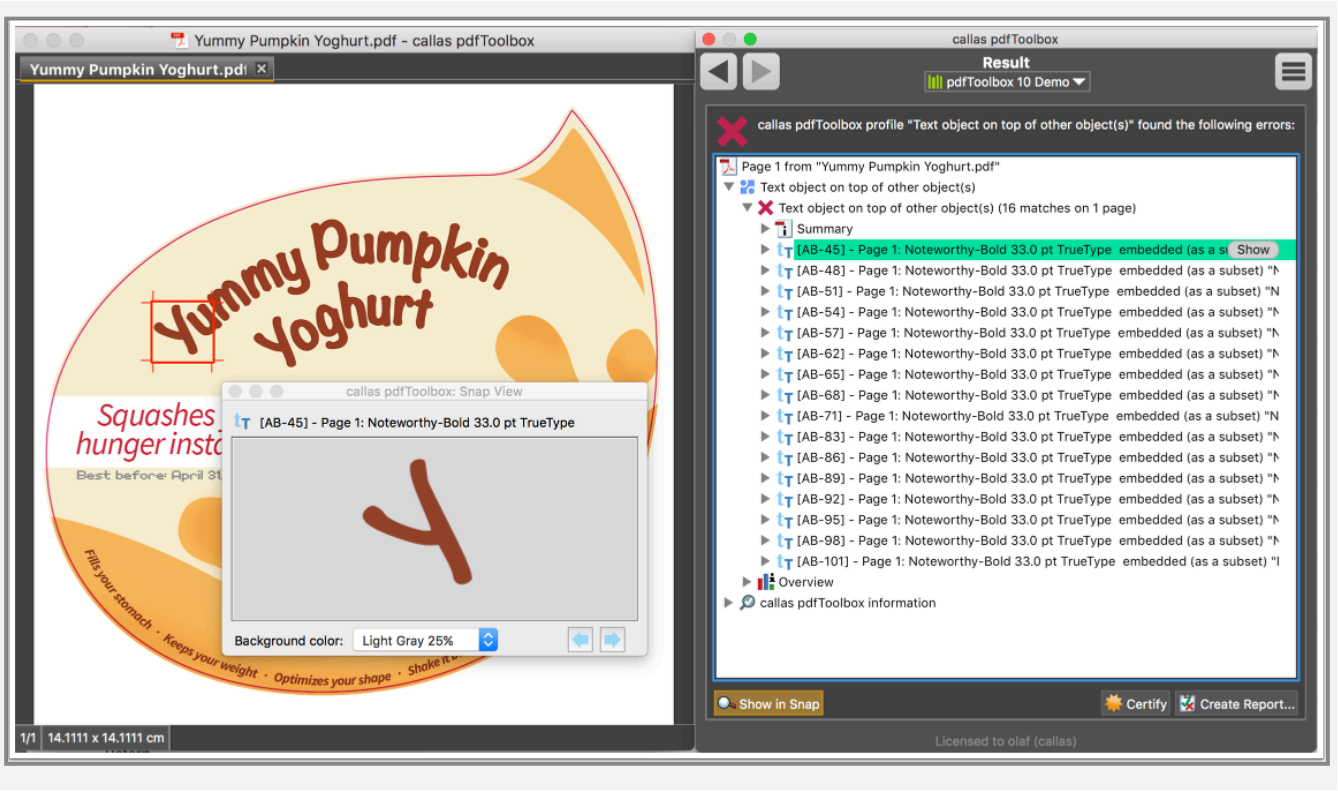
Text_object_on_top_of_other_object(s).kfpx

Yummy_Pumpkin_Yoghurt.pdf

Running the check *Text object on top of other object(s)* detects all text objects that visibly overlap at least one other object below them. As a consequence, this check will find all

text objects that sit on top of the label background, as can be seen in the screenshot below.

In order to find text objects that do not visibly overlap other objects, see the article [Above versus below: Object not on top of any other object](#).

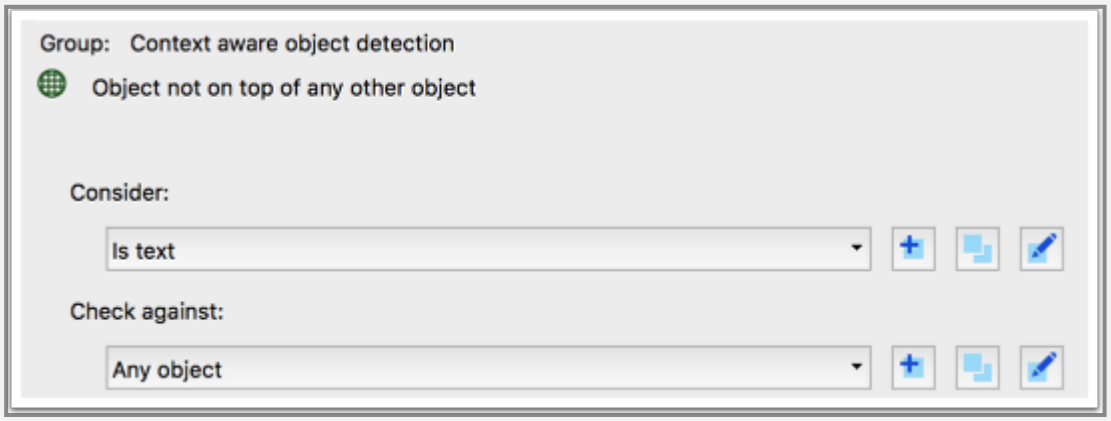


24.7 Above versus below: Object not on top of any other object

The property "Object not on top of other object(s)" detects whether an object does not have other object below it that it visibly overlaps.

The configuration options for a check based on the "Object not on top of other object(s)" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined which of these objects do not visibly overlap any other object from the set of objects defined by "Check against".



Sample files

The example and exercise below make use of the check ***Text object not on top of other objects.kfpx*** and the PDF file ***Yummy Pumpkin Yoghurt.pdf*** which are available for download:

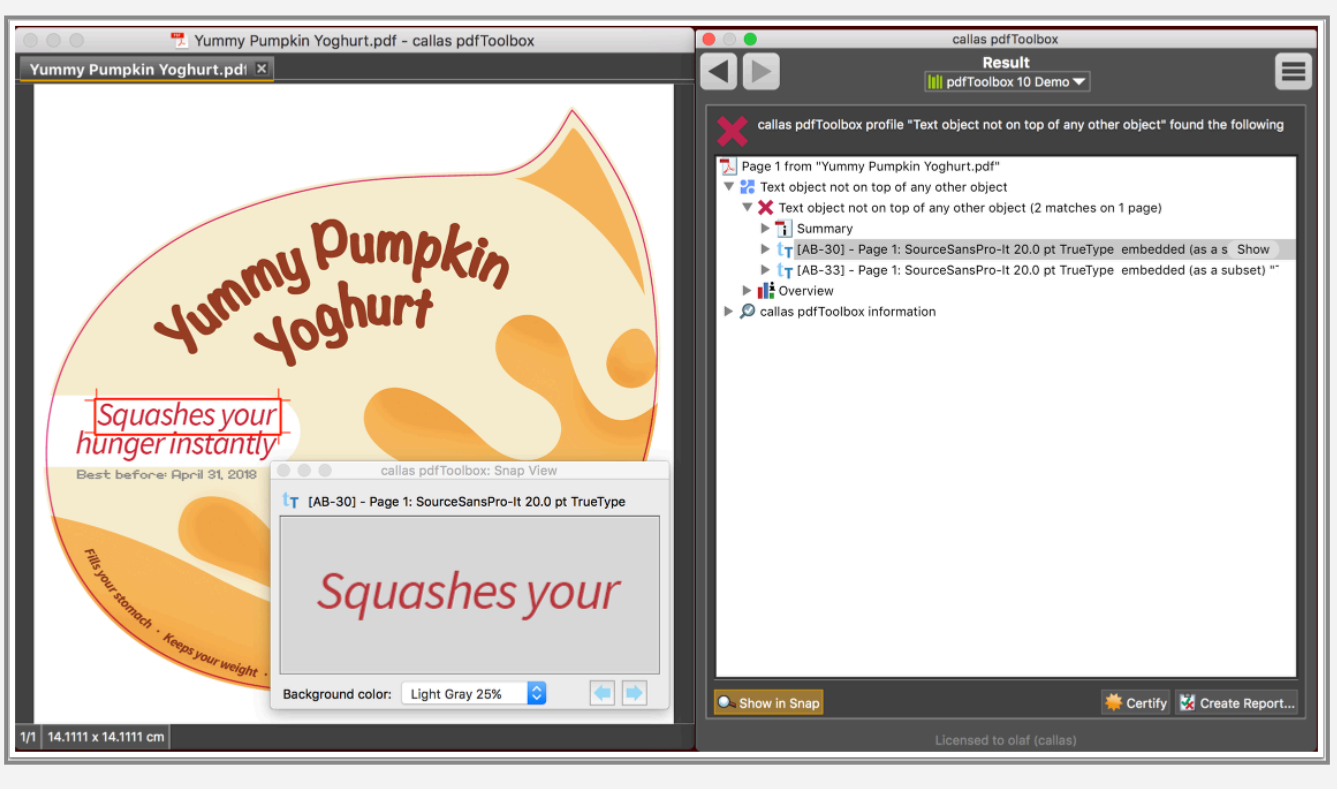
Text_object_not_on_top_of_any_other_object.kfpx

Yummy_Pumpkin_Yoghurt.pdf

Running the check ***Text object not on top of other object(s)*** detects all text objects that do not visibly overlap any other

object below them. As a consequence, this check will find all text objects "have nothing below them", as can be seen in the screenshot below: in the example, when the text "Squashes your hunger instantly" will be painted, nothing will have been painted in that text area before.

In order to find text objects that do visibly overlap other objects, see the article Above versus below: Object on top of other object(s).

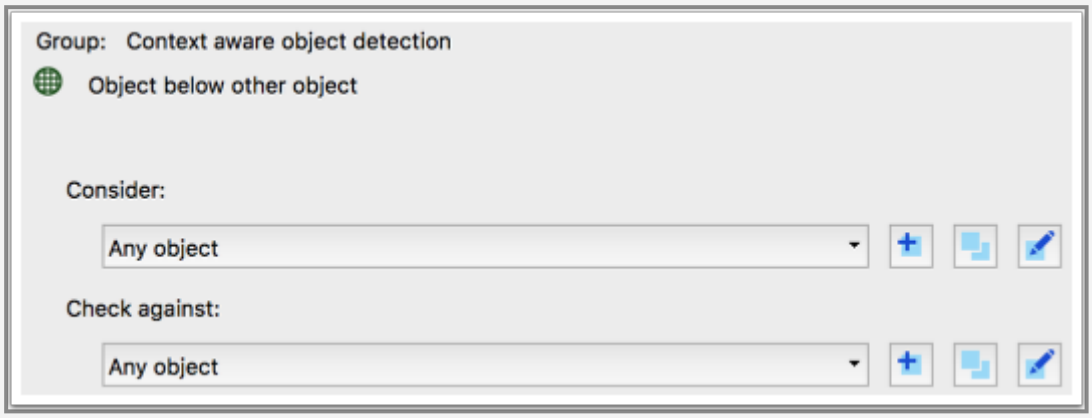


24.8 Above versus below: Object below other object

The property "Object below other object" detects whether an object is overlapped by at least one other object above it.

The configuration options for a check based on the "Object below other object" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined which of these objects is overlapped by at least one other object from the set of objects defined by "Check against".



Sample files

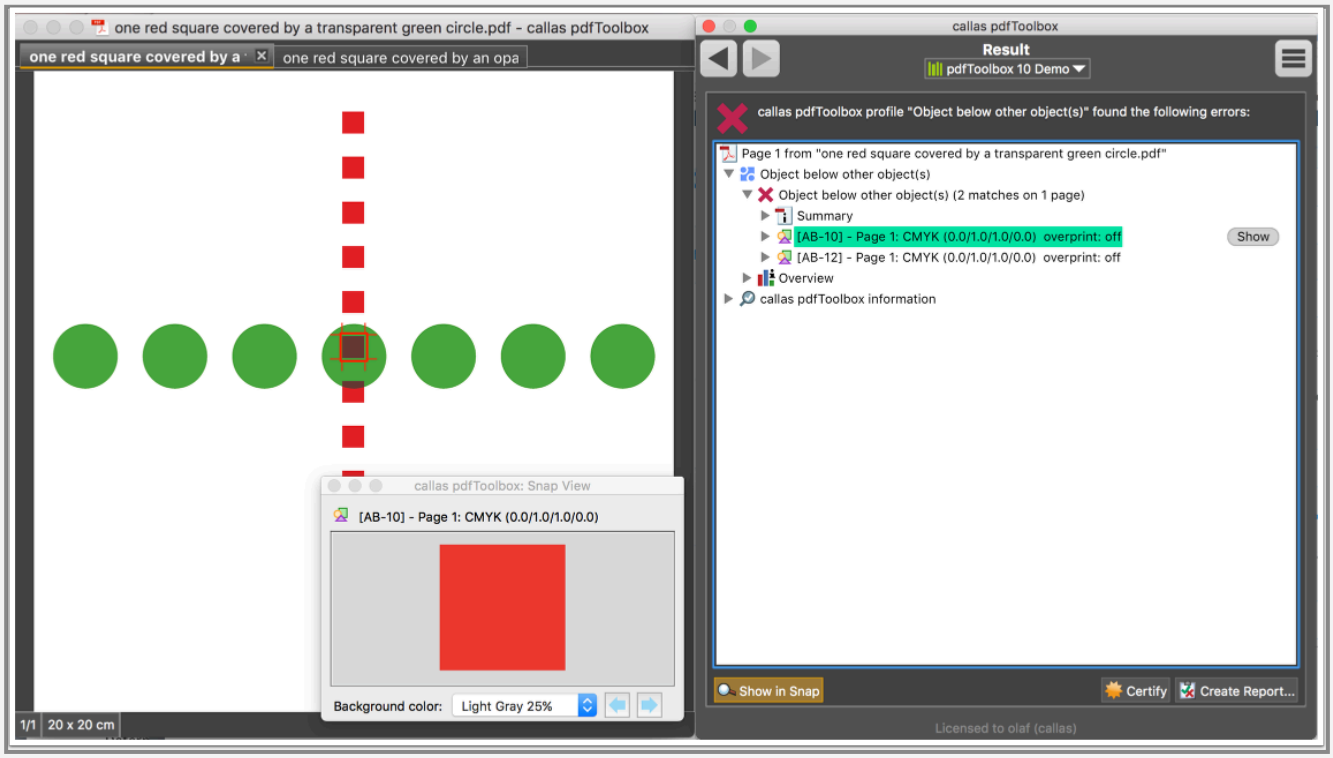
The example and exercise below make use of the check *Object below other object(s).kfpx* and the PDF files *one red square covered by a transparent green circle.pdf* and *one red square covered by an opaque green circle.pdf* which are available for download:

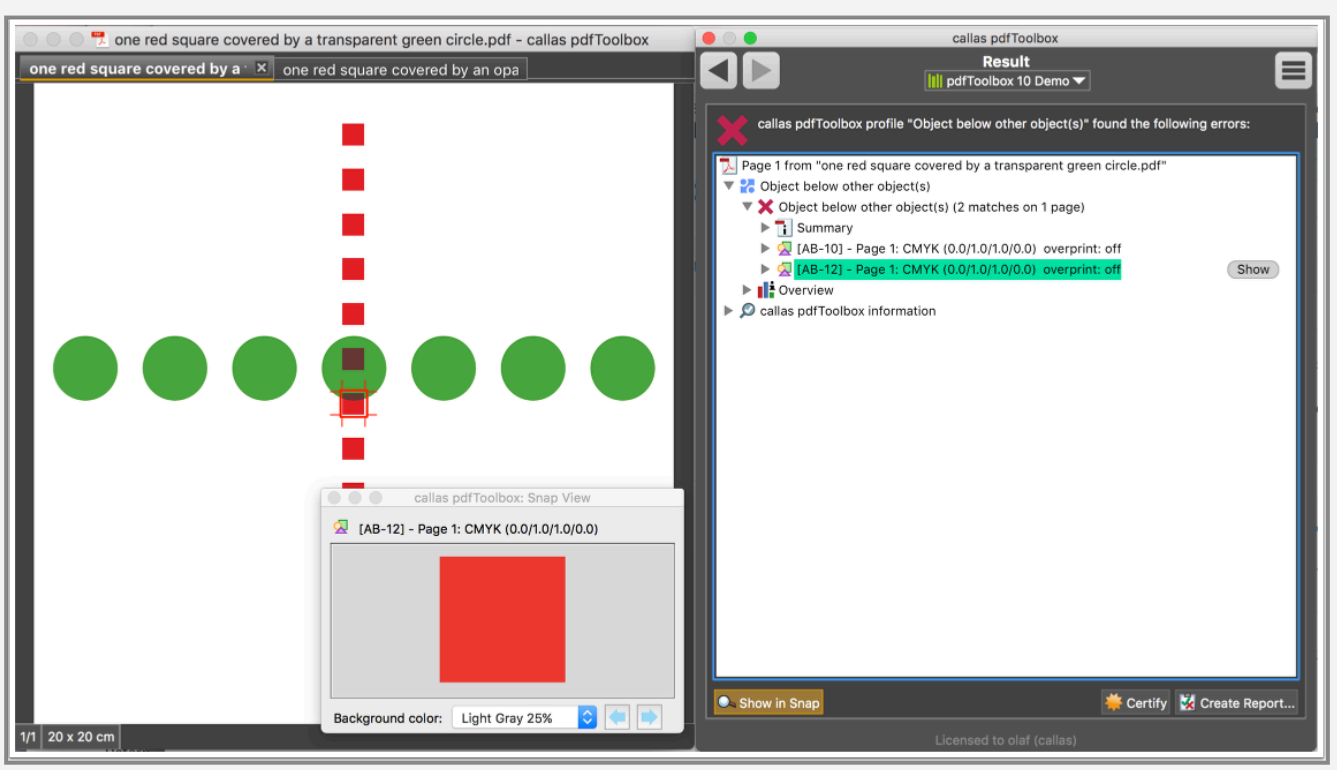
Object_below_other_object(s).kfpx

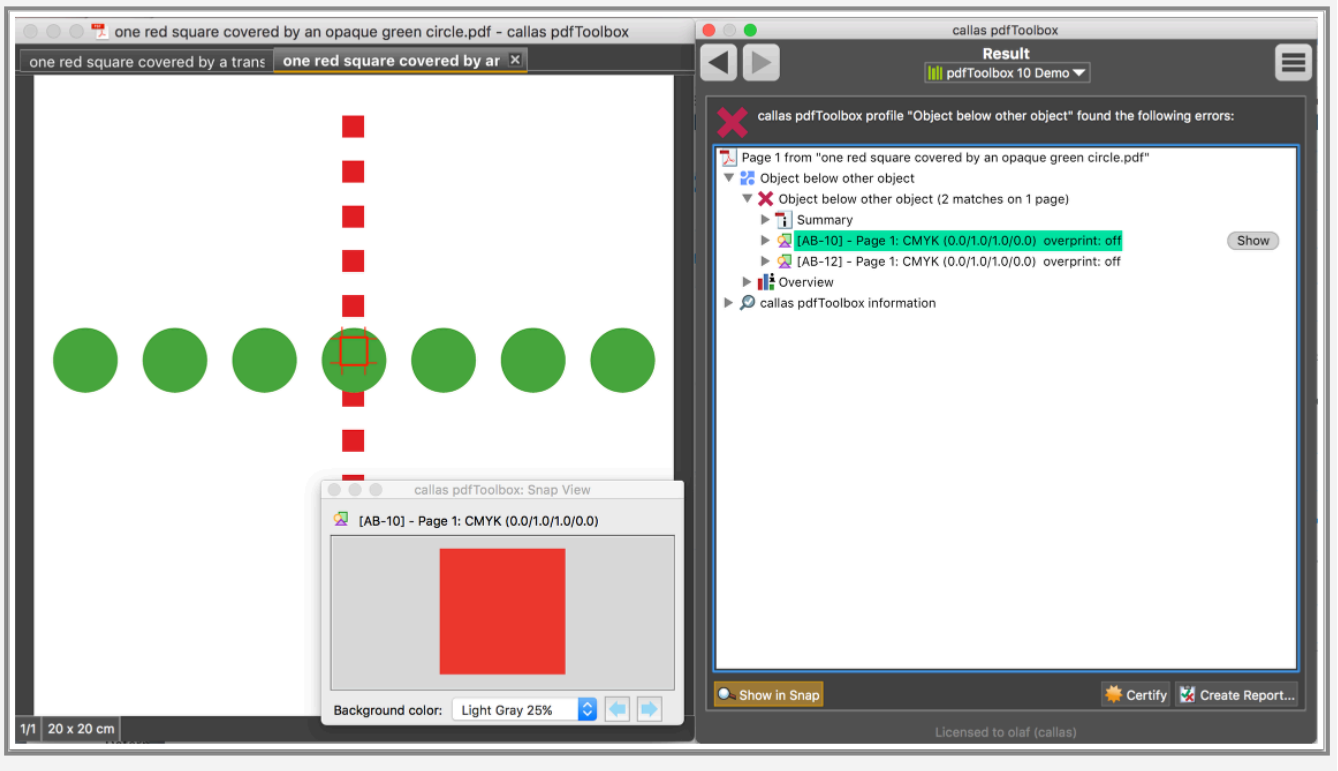
one_red_square_covered_by_a_transparent_green_.pdf

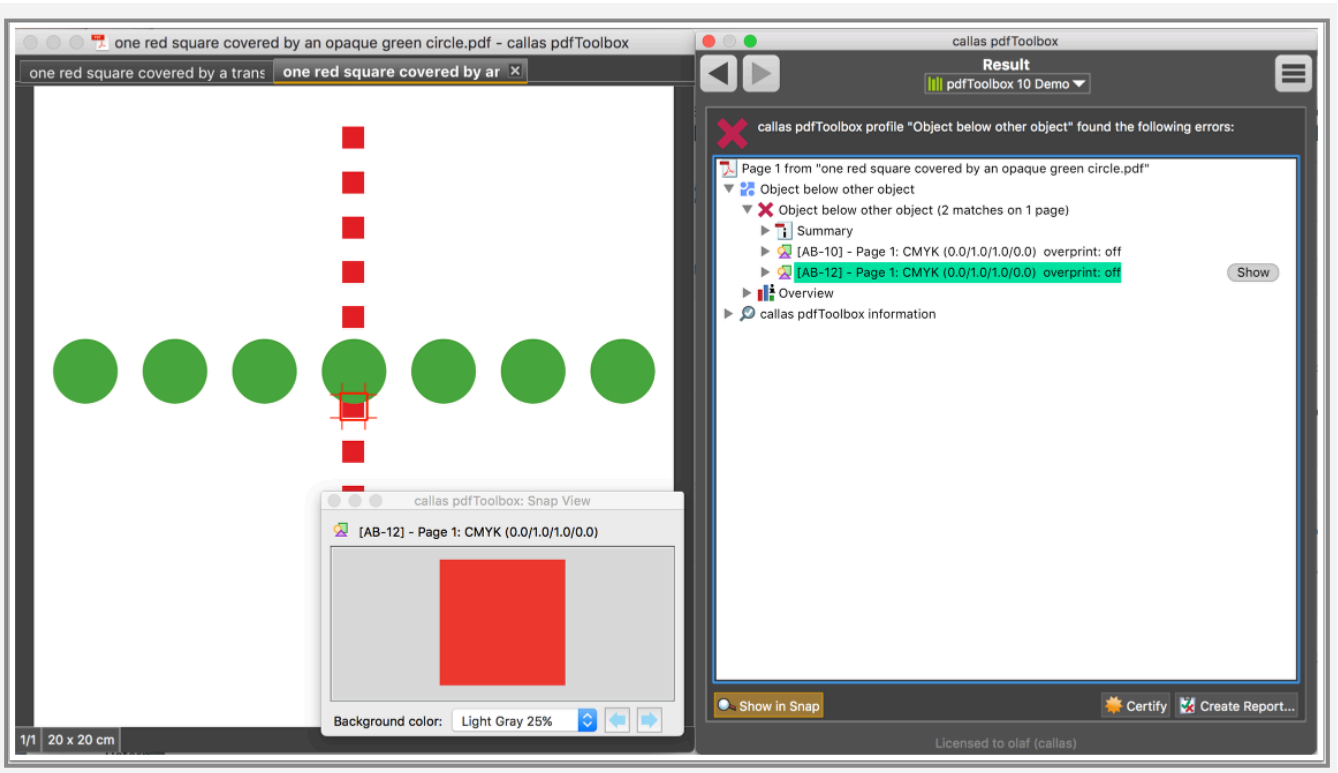
one_red_square_covered_by_an_opaque_green_circ.pdf

In the four screenshots shown below, it can be seen how two red squares – one fully covered by the green circle above it, the other one partiallly overlapped – are detected. The result is the same regardless whether the green circle is opaque (and thus makes the fully overlapped red square invisible) or transparent.




one red square covered by a transparent green circle.pdf - callas pdfToolbox
one red square covered by a
one red square covered by an opa
callas pdfToolbox: Snap View
[AB-12] - Page 1: CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
1/1
20 x 20 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "Object below other object(s)" found the following errors:
Page 1 from "one red square covered by a transparent green circle.pdf"
Object below other object(s)
Object below other object(s) (2 matches on 1 page)
Summary
[AB-10] - Page 1: CMYK (0.0/1.0/1.0/0.0) overprint: off
[AB-12] - Page 1: CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)


one red square covered by an opaque green circle.pdf - callas pdfToolbox
one red square covered by a trans
one red square covered by ar
callas pdfToolbox: Snap View
[AB-10] - Page 1: CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
1/1
20 x 20 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "Object below other object" found the following errors:
Page 1 from "one red square covered by an opaque green circle.pdf"
Object below other object
Object below other object (2 matches on 1 page)
Summary
[AB-10] - Page 1: CMYK (0.0/1.0/1.0/0.0) overprint: off
[AB-12] - Page 1: CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)

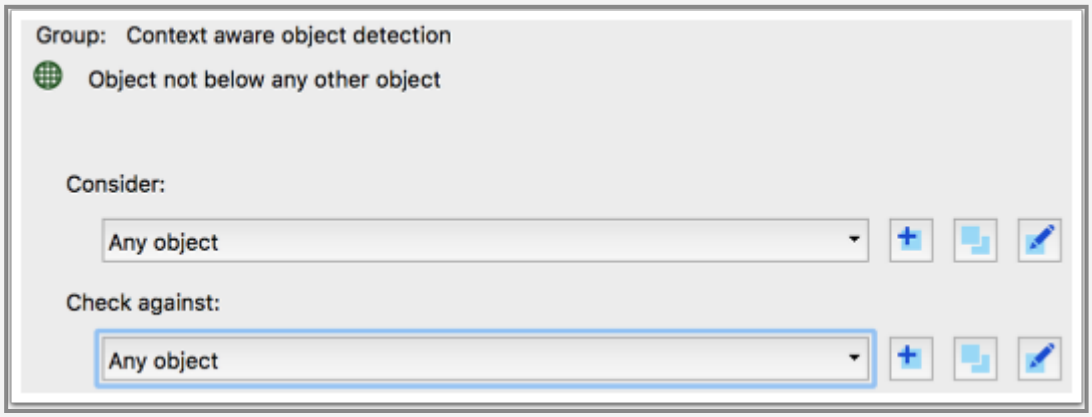

one red square covered by an opaque green circle.pdf - callas pdfToolbox
one red square covered by a trans
one red square covered by ar
callas pdfToolbox: Snap View
[AB-12] - Page 1: CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
1/1
20 x 20 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "Object below other object" found the following errors:
Page 1 from "one red square covered by an opaque green circle.pdf"
Object below other object
Object below other object (2 matches on 1 page)
Summary
[AB-10] - Page 1: CMYK (0.0/1.0/1.0/0.0) overprint: off
[AB-12] - Page 1: CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)

24.9 Above versus below: Object not below any other object

The property "Object not below any other object" detects whether an object is not overlapped by any other object above it.

The configuration options for a check based on the "Object not below other object" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined which of these objects is not overlapped by any other object from the set of objects defined by "Check against".



Sample files

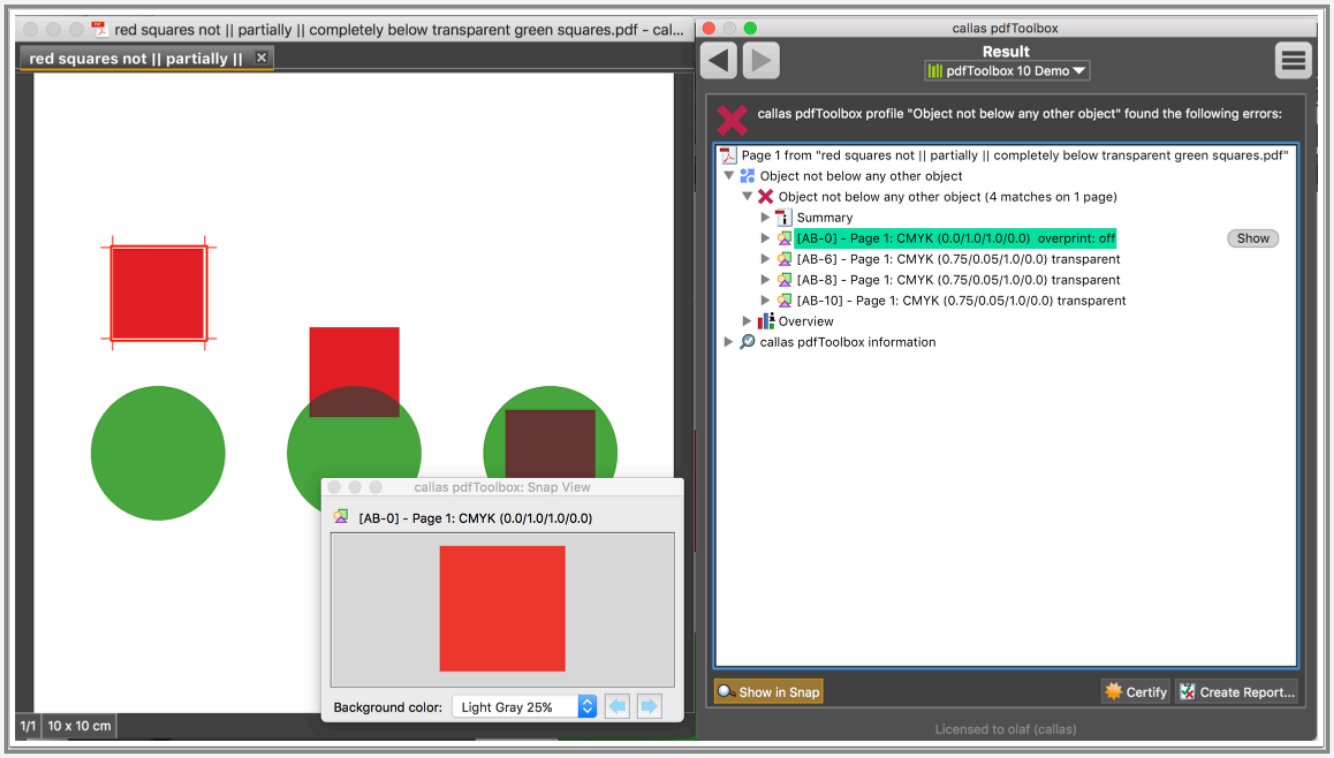
The example below makes use of the check ***Object not below other object.kfpx*** and the PDF file ***red squares not || partially || completely below opaque green squares*** which are available for download:

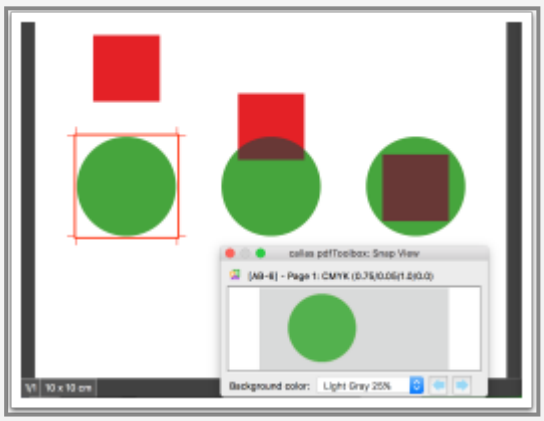
Object_not_below_any_other_object.kfpx

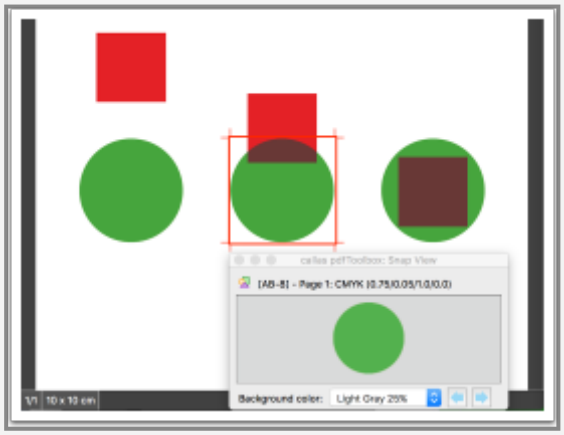
red_squares_not____partially____completely_bel.pdf

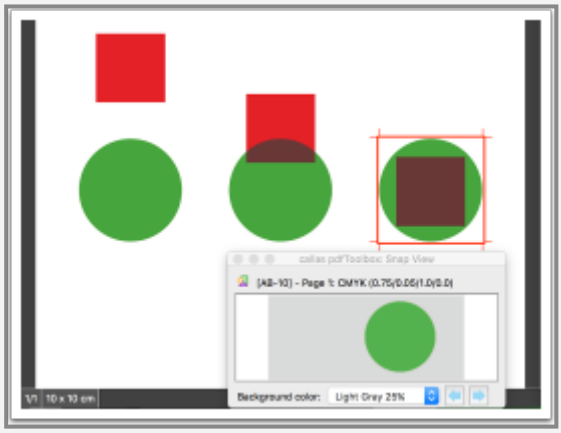
In the four screenshots shown below, it can be seen how one red square as well as the three green circles are detected as those objects do not have any object above them that overlap or cover them.

Note: In the sample file, first the red squares are painted, followed b the green circles. Thus, by definition, none of the green circles will have objects above them that overlap or cover them.







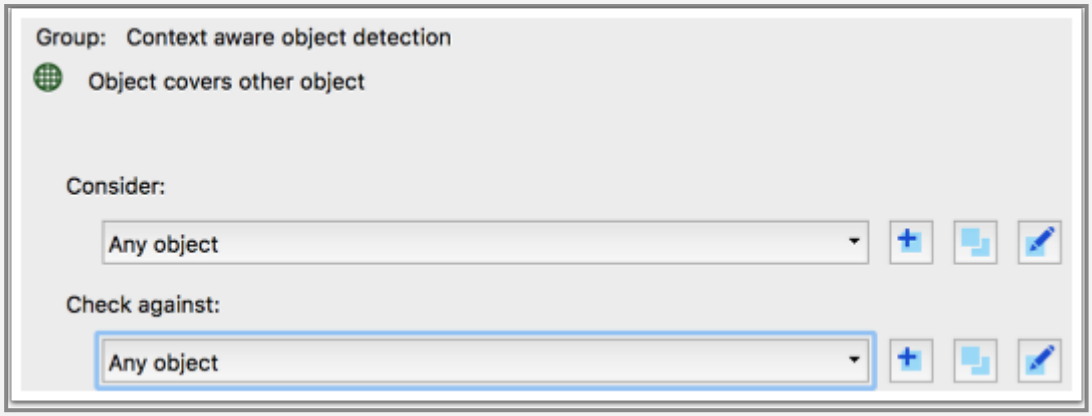


24.10 Above versus below: Object covers other object

The property "Object covers other object" detects whether an object covers at least the same area taken by some other object underneath it, regardless whether the upper object is opaque and thus makes the object underneath it invisible, or not.

The configuration options for a check based on the "Object covers other object" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined which of these objects covers at least one other object from the set of objects defined by "Check against".



Sample files

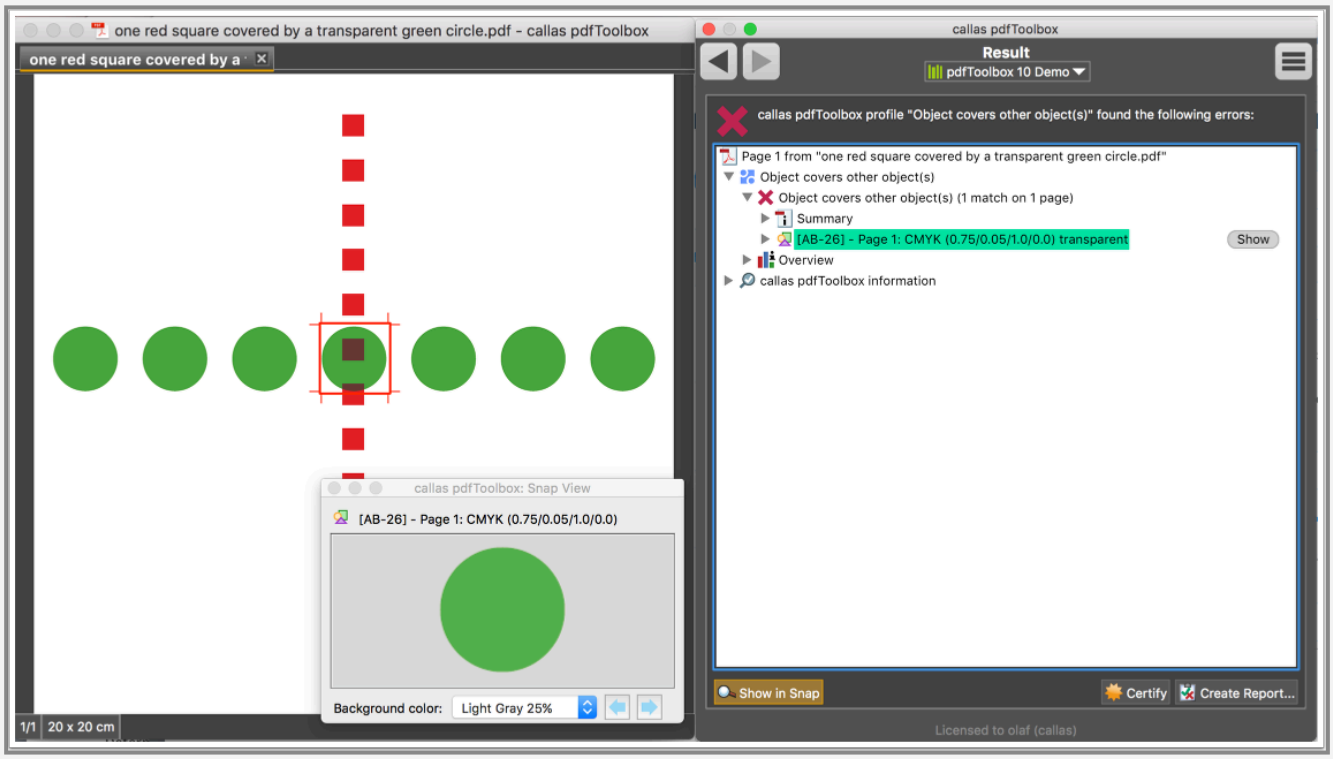
The example and exercise below make use of the check ***Object covers other object(s).kfpx*** and the PDF files ***one red square covered by a transparent green circle.pdf*** and ***one red square covered by an opaque green circle.pdf*** which are available for download:

Object_covers_other_object(s).kfpx

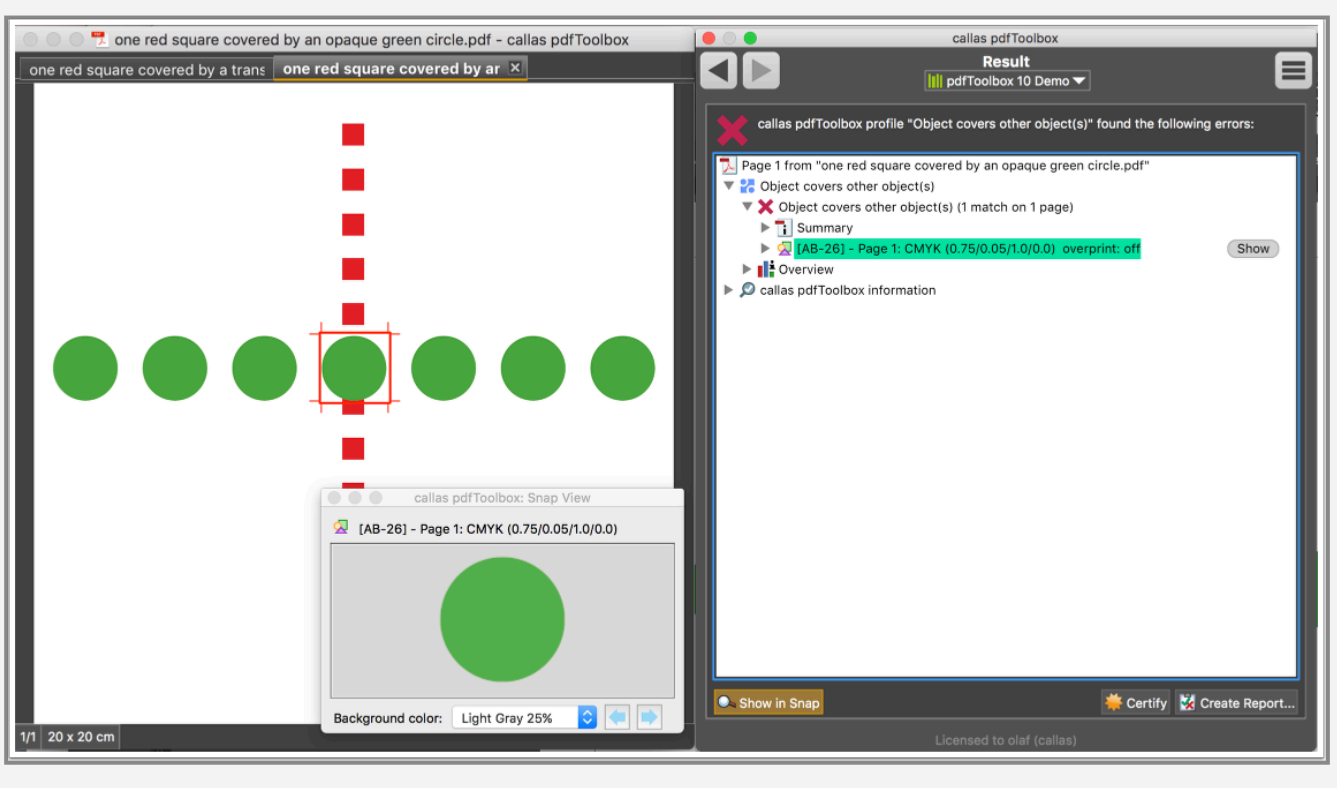
one_red_square_covered_by_a_transparent_green_.pdf

one_red_square_covered_by_an_opaque_green_circ.pdf

In the example shown below, the green circle (set to be transparent) in the center is detected, as it fully "covers" the red rectangle below it, and is on top of that red recangle.



In the example shown below, the green circle (set to be opaque) in the center is detected, as it fully "covers" the red rectangle below it, and is on top of that red recangle.

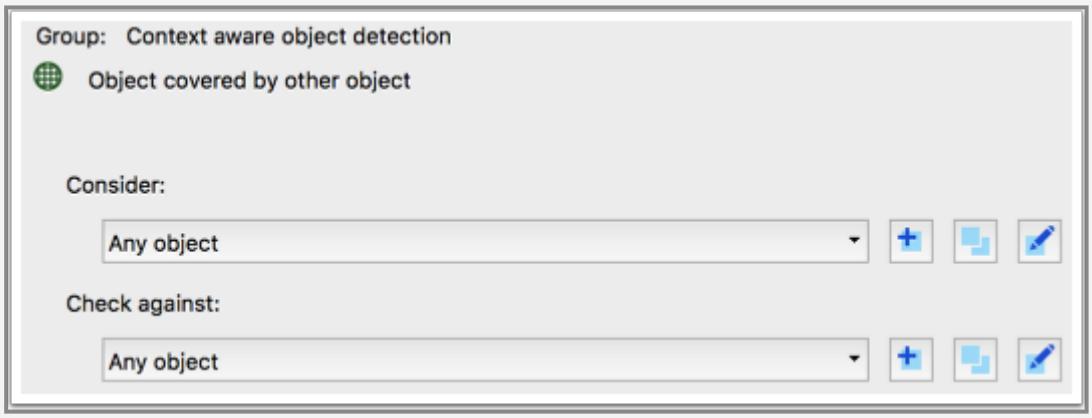

one red square covered by an opaque green circle.pdf - callas pdfToolbox
one red square covered by a trans
one red square covered by ar
callas pdfToolbox: Snap View
[AB-26] - Page 1: CMYK (0.75/0.05/1.0/0.0)
Background color:
Light Gray 25%
1/1
20 x 20 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "Object covers other object(s)" found the following errors:
Page 1 from "one red square covered by an opaque green circle.pdf"
Object covers other object(s)
Object covers other object(s) (1 match on 1 page)
Summary
[AB-26] - Page 1: CMYK (0.75/0.05/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)

24.11 Above versus below: Object covered by other object

The property "Object covered by other object" detects whether an object is covered by at least one other object above it. In this context it is irrelevant, whether an object above is opaque or transparent.

The configuration options for a check based on the "Object covered by other object" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined whether any of these objects is covered by at least one other object from the set of objects defined by "Check against".



Sample files

The example below makes use of the check ***Object covered by other object.kfpx*** and the PDF files ***red squares not || partially || completely below opaque green squares.pdf*** and ***red squares not || partially || completely below transparent green squares.pdf*** which are available for download:

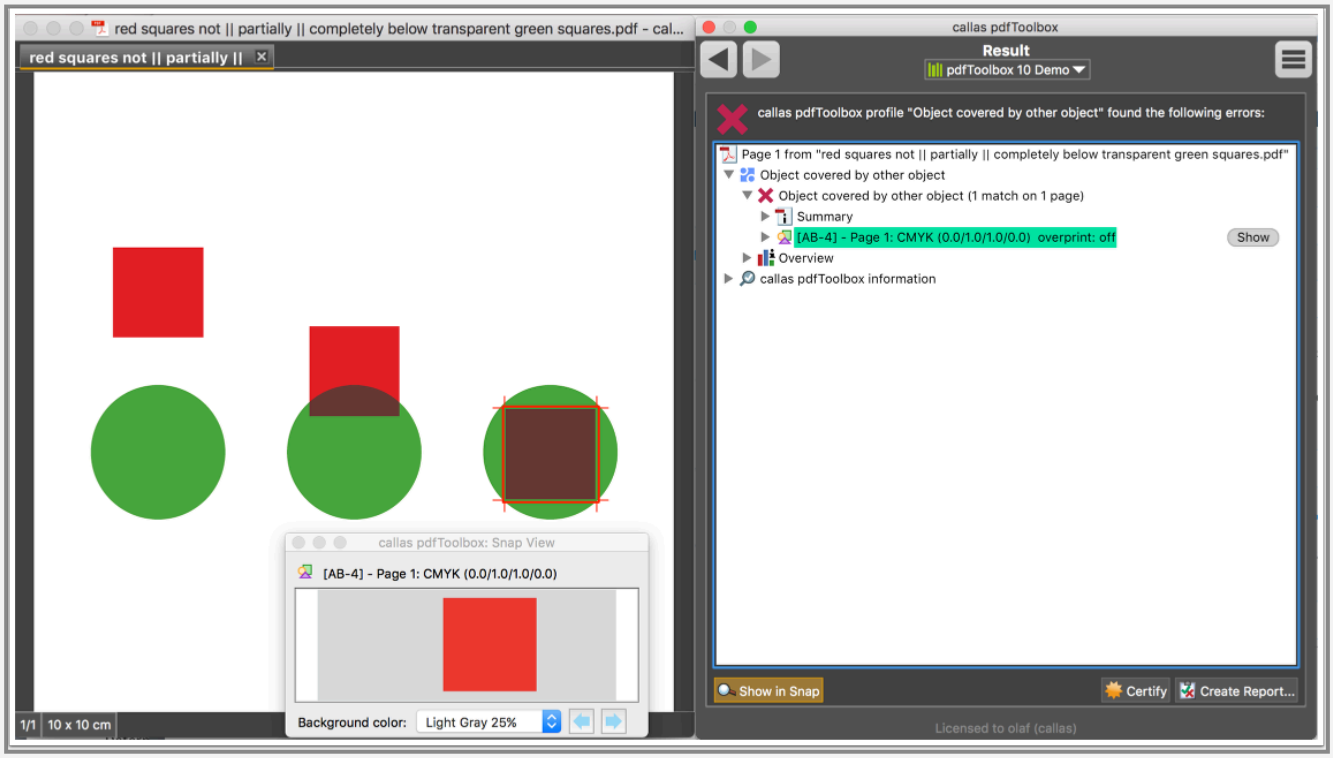
Object_covered_by_other_object.kfpx

red_squares_not____partially____completely_bel.pdf

red_squares_not____partially____completely_b-1.pdf

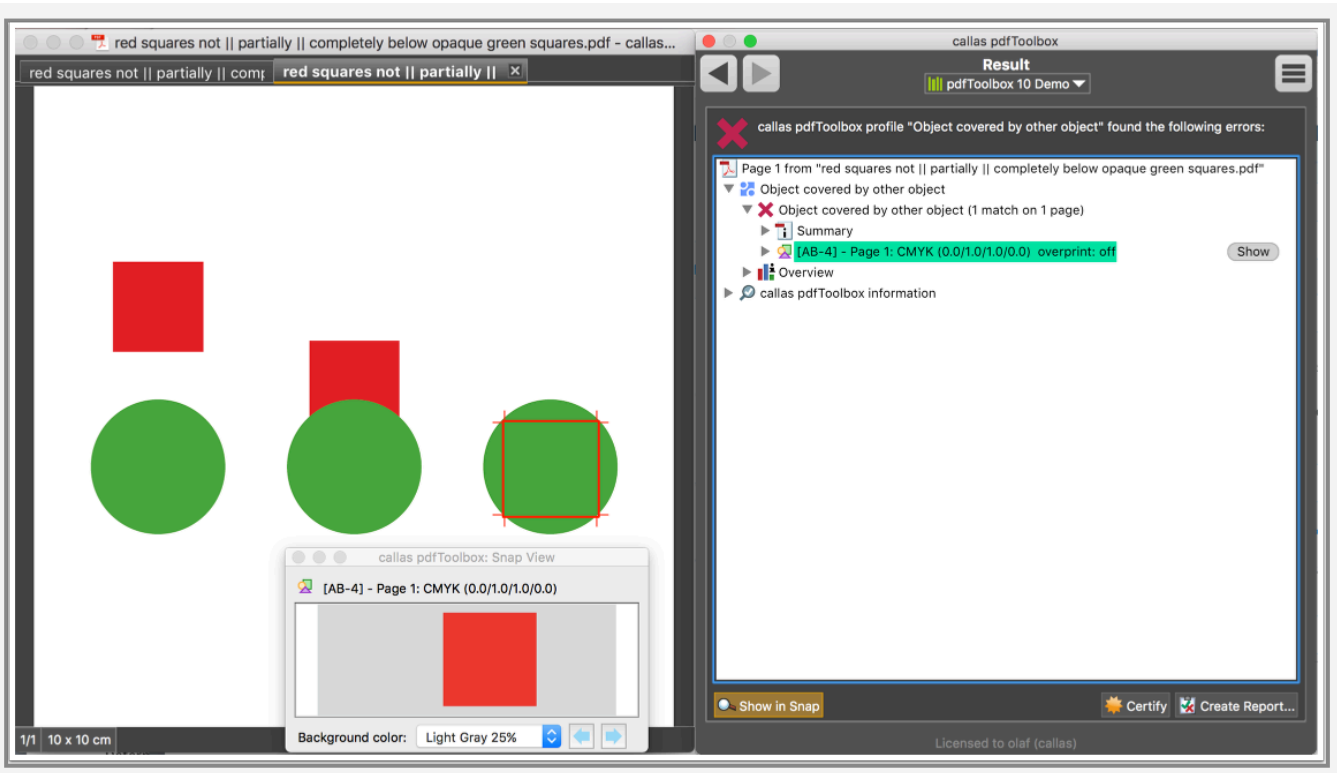
In the screenshot shown below, it can be seen how the red square completely under a green circle is detected.

Note: In the sample file, first three red squares are painted, followed by three green circles.



In the screenshot shown below, the red square completely under a green circle is detected, despite being invisible due to the green circle above it being opaque and thus hiding it.

Note: In the sample file, first three red squares are painted, followed by three green circles.


red squares not || partially || completely below opaque green squares.pdf - callas...
red squares not || partially || comp
red squares not || partially ||
callas pdfToolbox: Snap View
[AB-4] - Page 1: CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
1/1
10 x 10 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "Object covered by other object" found the following errors:
Page 1 from "red squares not || partially || completely below opaque green squares.pdf"
Object covered by other object
Object covered by other object (1 match on 1 page)
Summary
[AB-4] - Page 1: CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)

24.12 Inside versus outside: Object inside other object

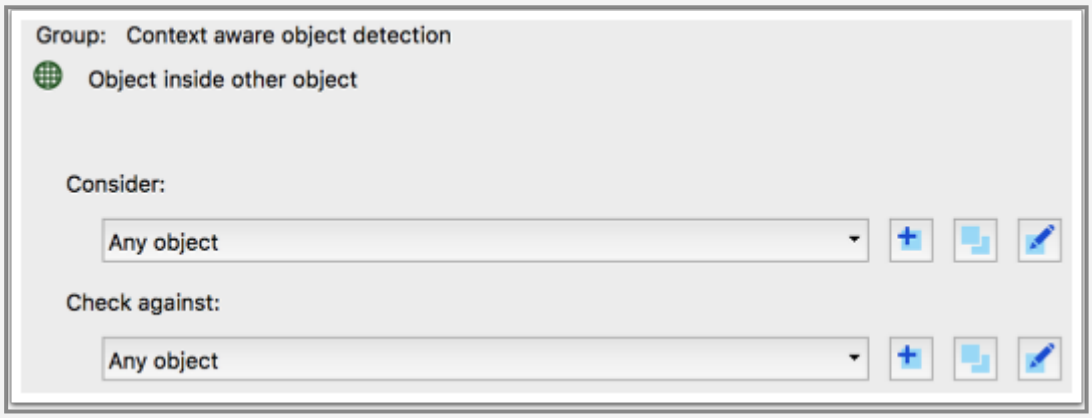
The property "Object inside other object" detects whether an object is inside the *area covered* by some other object above it. In this context it is irrelevant, which object is on top and which is below, and also whether one or the other object or both are opaque or transparent.

The configuration options for a check based on the "Object inside other object" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined whether any of these objects is inside the *area covered* by some other object from the set of objects defined by "Check against".

In this context, it is very important to understand the meaning of *area covered by an object*: the area covered by an object is the area where the object draws something.

- For a **circle filled with red color**, that area is a circle (including everything inside the border of that circle).
- For **circle shaped line drawn in red**, the area of that circular line object is just the area where the line is drawn: a thin red stripe going around in a circular fashion, but not the area inside the circular line.



Sample files

The example below makes use of the check ***Object inside other object.kfpx*** and the PDF file ***six red squares below and six green circles on top.pdf*** which are available for download:

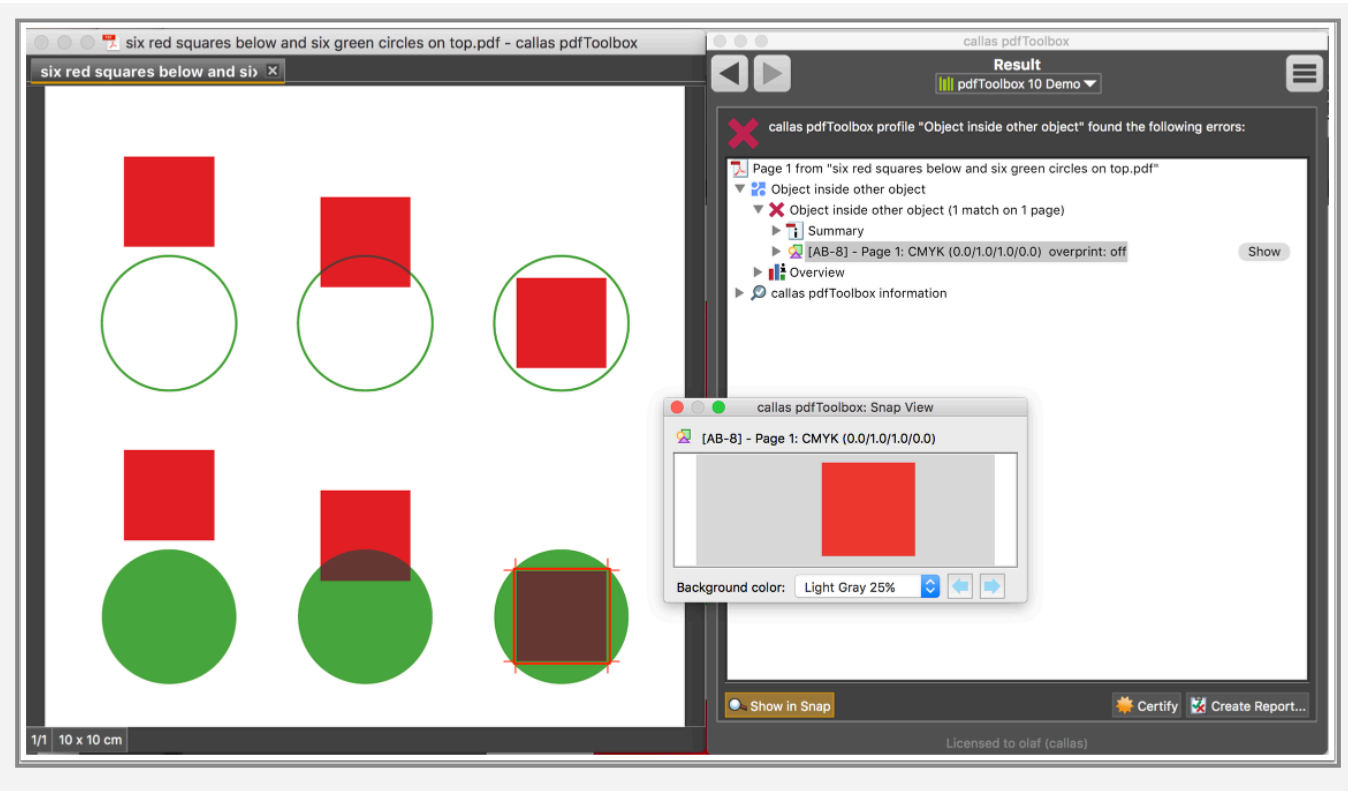
Object_inside_other_object.kfpx

six_red_squares_below_and_six_green_circles_on.pdf

In the screenshot shown below, it can be seen how only the one red square completely inside the area taken by a green circle is detected.

Counter to what some may expect, the red square "inside" the green circular line is not detected. This is by design: the area of the green circular line is just the area where the green line is drawn.

Note: In the sample file, first six red squares are painted, followed by six green circles.


six red squares below and six green circles on top.pdf - callas pdfToolbox
six red squares below and si
1/1
10 x 10 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "Object inside other object" found the following errors:
Page 1 from "six red squares below and six green circles on top.pdf"
Object inside other object
Object inside other object (1 match on 1 page)
Summary
[AB-8] - Page 1: CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
callas pdfToolbox: Snap View
[AB-8] - Page 1: CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)

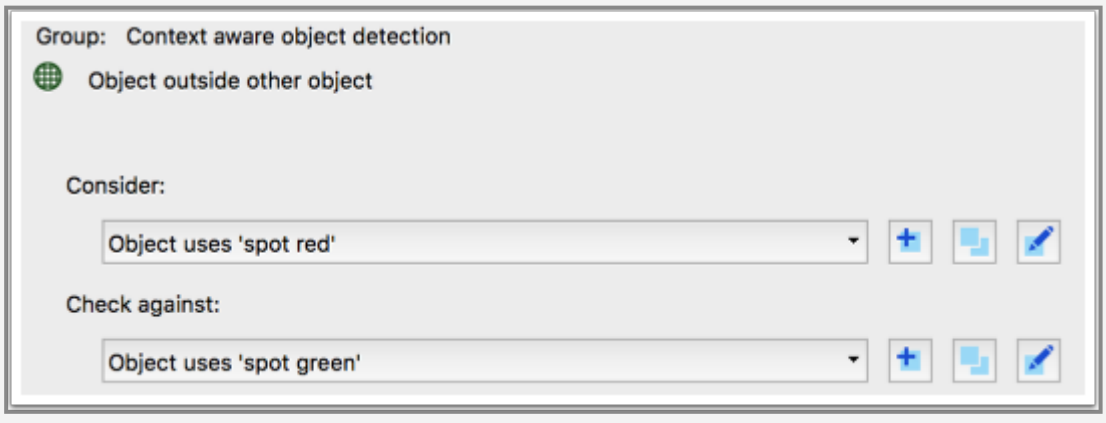
24.13 Inside versus outside: Object outside other object

The property "Object outside other object" detects whether an object is outside the *area covered* by some other object above it. In this context it is irrelevant, which object is on top and which is below, and also whether one or the other object or both are opaque or transparent.

The configuration options for a check based on the "Object outside other object" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined whether any of these objects is outside the *area covered* by some other object from the set of objects defined by "Check against".

In this context, it is very important to understand the meaning of *area covered by an object*: the area covered by an object is the area where the object draws something. For a circle filed with green color, that area is a circle (including everything inside the border of that circle). For a circle shaped line drawn in green, the area of that circular line object is just the area where the line is drawn: a thin green stripe going around in a circular fashion, but not the area inside that circular line.



Sample files

The example below makes use of the check *'red spot' object outside 'spot green' objects.kfpx* and the PDF file *six 'spot*

red' squares below and six 'spot green' circles on top.pdf
which are available for download:

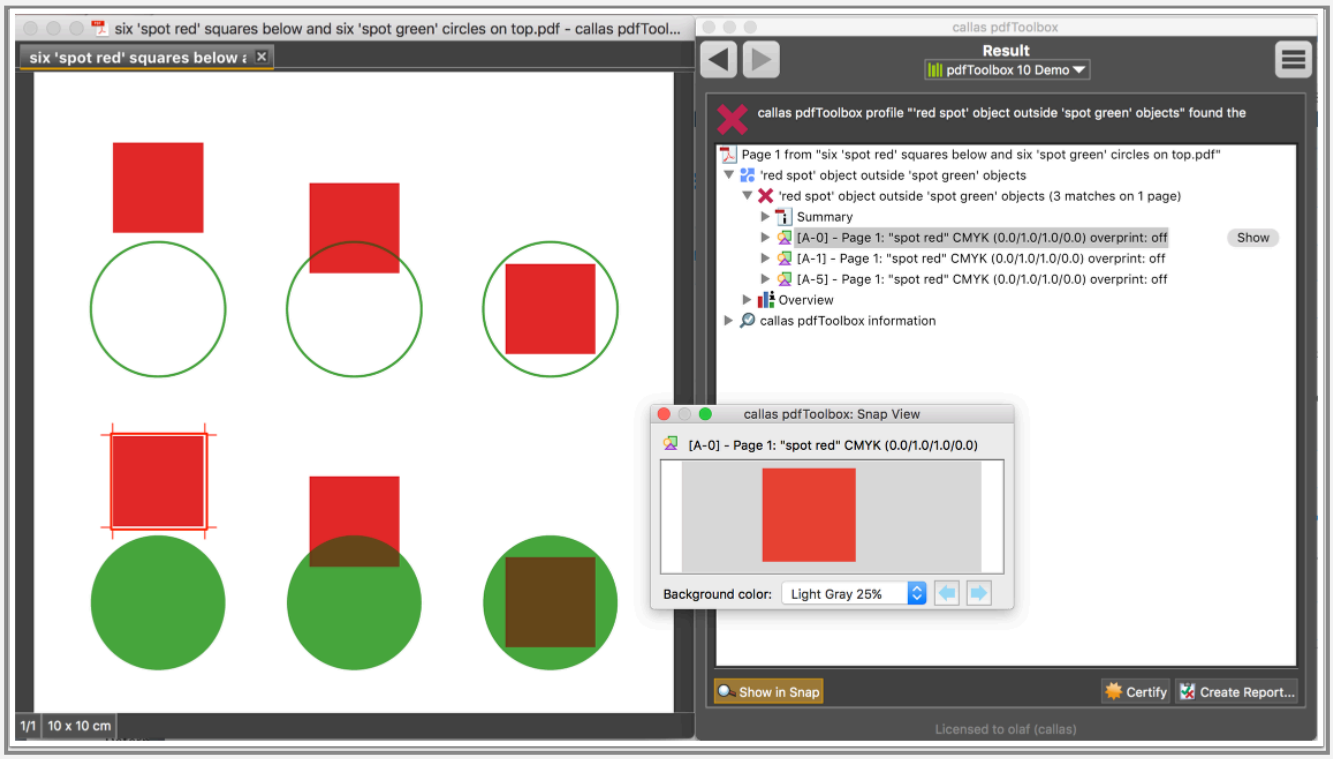
'red_spot'_object_outside_'spot_green'_objects.kfpx

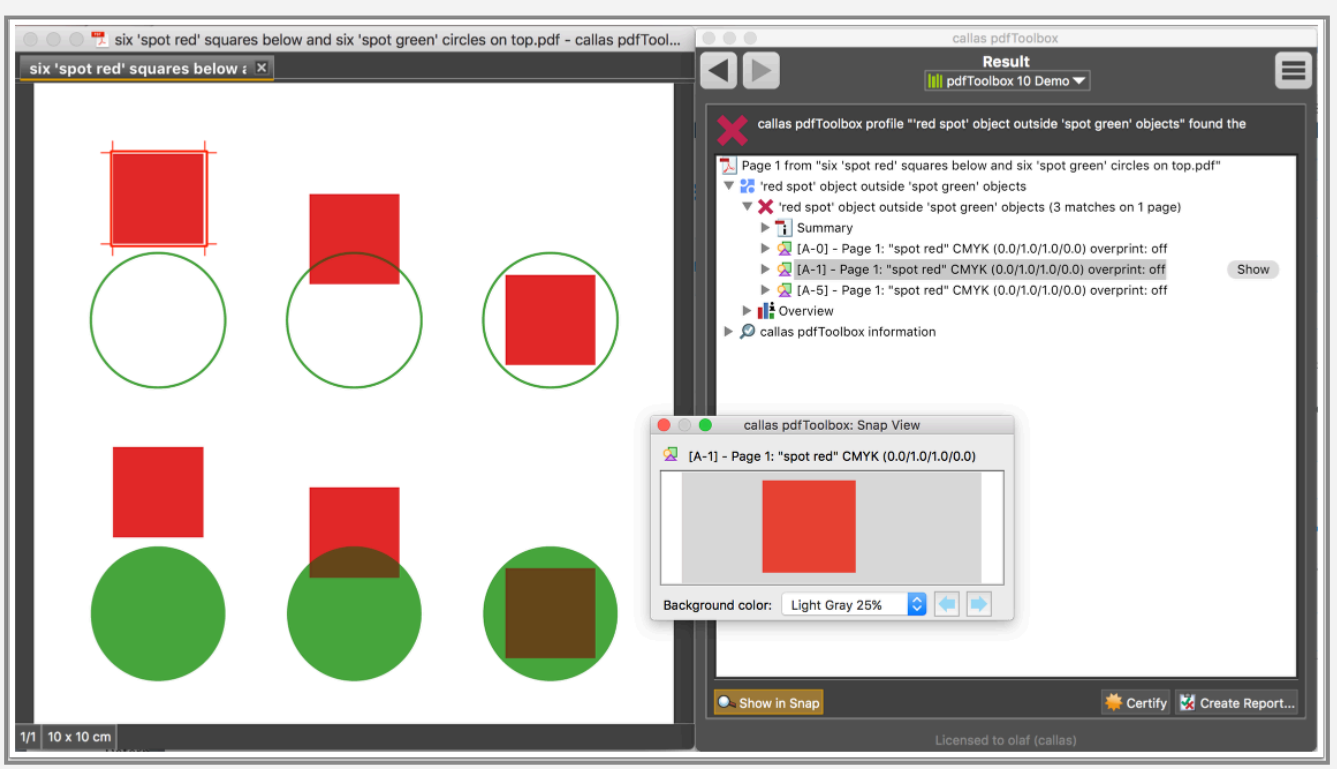
six_red_squares_below_and_six_green_circles_on.pdf

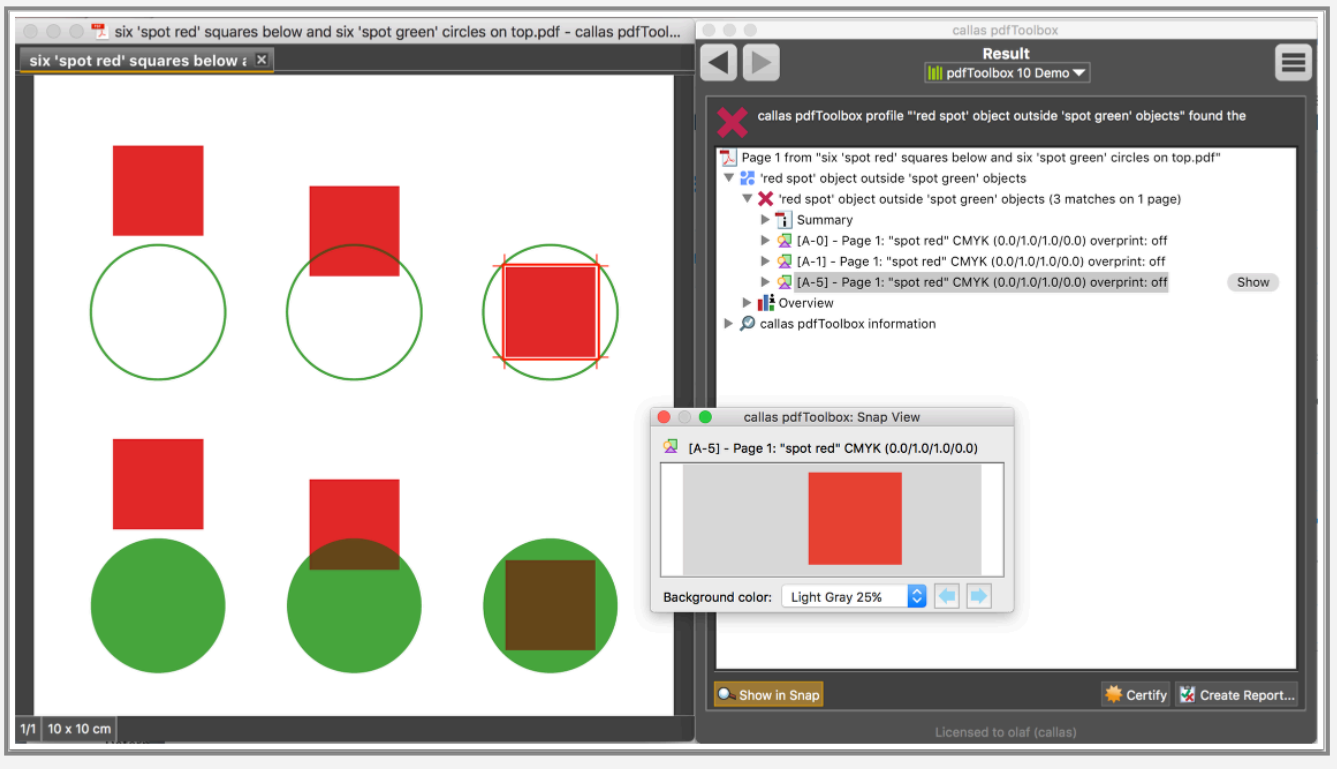
In the screenshot shown below, it can be seen how three red squares are detected as being completely outside the area taken by the green squares.

Counter to what some may expect, the red square "inside" the green circular line is also detected (see third screenshot below). This is by design: the area of the green circular line is just the area where the green line is drawn – relative to that circular line, the red square is outside of the area where that line is drawn.

Note: In the sample file, first six red squares are painted, followed by six green circles.




six 'spot red' squares below and six 'spot green' circles on top.pdf - callas pdfTool...
six 'spot red' squares below a
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "'red spot' object outside 'spot green' objects" found the
Page 1 from "six 'spot red' squares below and six 'spot green' circles on top.pdf"
'red spot' object outside 'spot green' objects
'red spot' object outside 'spot green' objects (3 matches on 1 page)
Summary
[A-0] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
[A-1] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
[A-5] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
callas pdfToolbox: Snap View
[A-1] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)
1/1
10 x 10 cm


six 'spot red' squares below and six 'spot green' circles on top.pdf - callas pdfTool...
six 'spot red' squares below a
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "'red spot' object outside 'spot green' objects" found the
Page 1 from "six 'spot red' squares below and six 'spot green' circles on top.pdf"
'red spot' object outside 'spot green' objects
'red spot' object outside 'spot green' objects (3 matches on 1 page)
Summary
[A-0] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
[A-1] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
[A-5] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
callas pdfToolbox: Snap View
[A-5] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)
1/1
10 x 10 cm

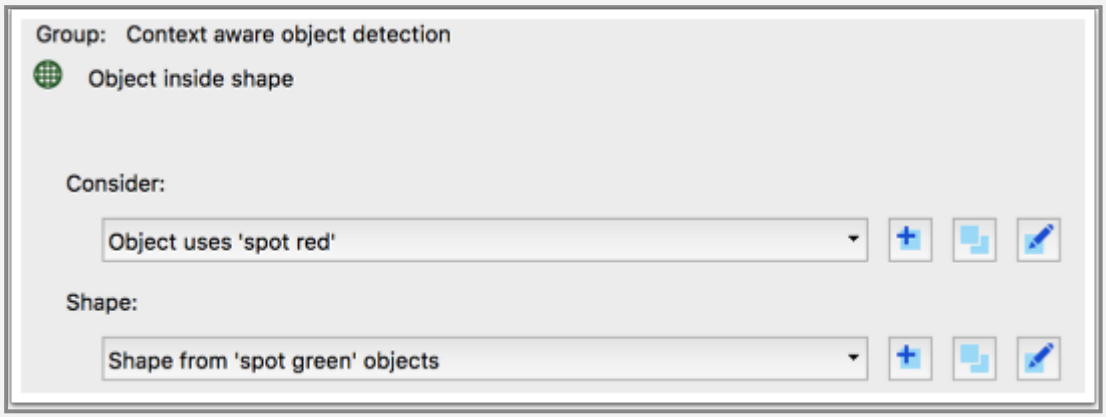
24.14 Inside versus outside: Object inside shape

The property "Object inside shape" detects whether an object is inside the area defined by a shape. In this context it is irrelevant, whether objects are on top or below, and also whether objects are opaque or transparent.

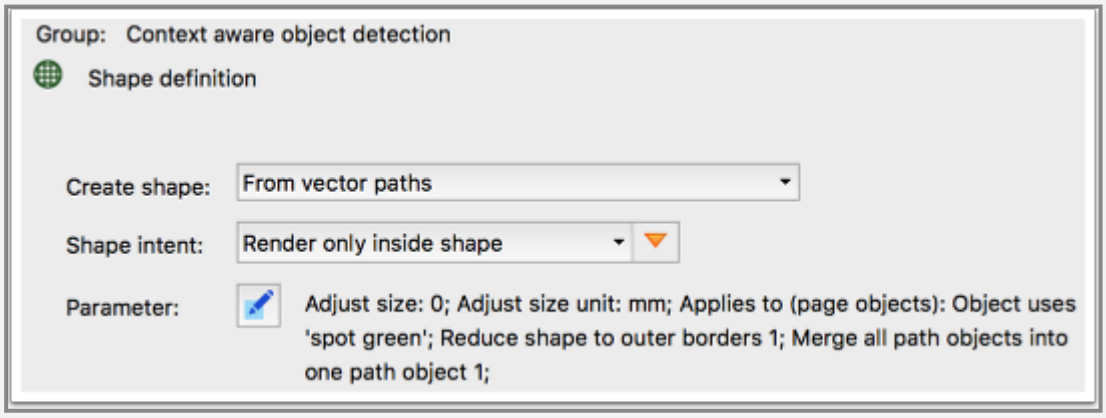
Shapes provide a means to establish a definition for an area in a very flexible manner, starting from simple shapes such as rectangles, areas defined by vector objects, or areas derived from rendering some or all of the content of a page and then turning the border of the rendered content into a definition of an area. For details see the article "Shapes" property for use in "Context aware object detection" checks.

The configuration options for a check based on the "Object inside shape" property can be seen below.

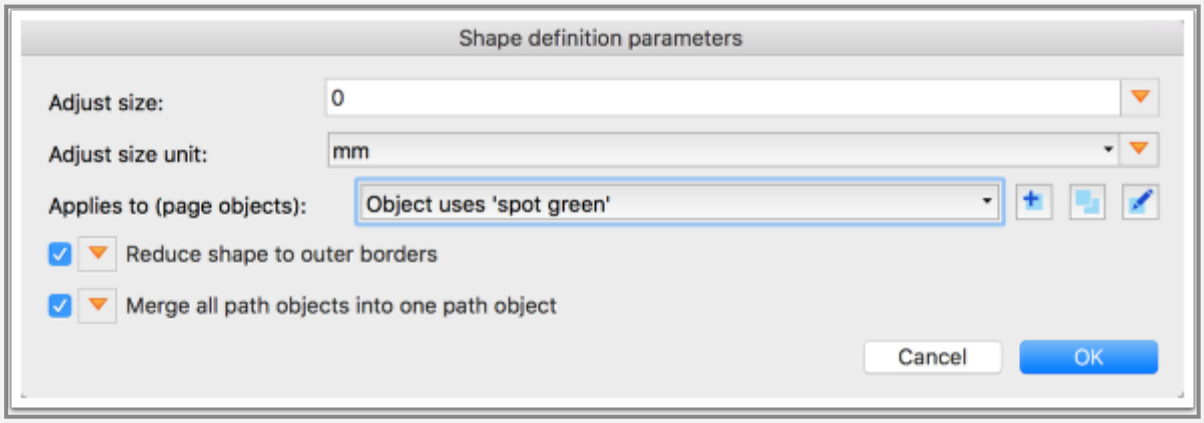
The "Consider" option defines the set of objects for which it shall be determined whether any of these objects is inside the shape defined by the "Check against" option.



In the example used below, the shape is derived from a subset of the vector objects on the page.



The subset of vector objects on the page to be taken into account for establishing the shape are 'spot green' vector objects. In the example used below, in order to achieve the desired effect, the option "Reduce shape to outer border" has been checked.



Sample files

The example below makes use of the check *'spot red' object(s) inside outer border shape of 'spot green' objects.kfpx.kfpx* and the PDF file *six 'spot red' squares below and six 'spot green' circles on top.pdf* which are available for download:

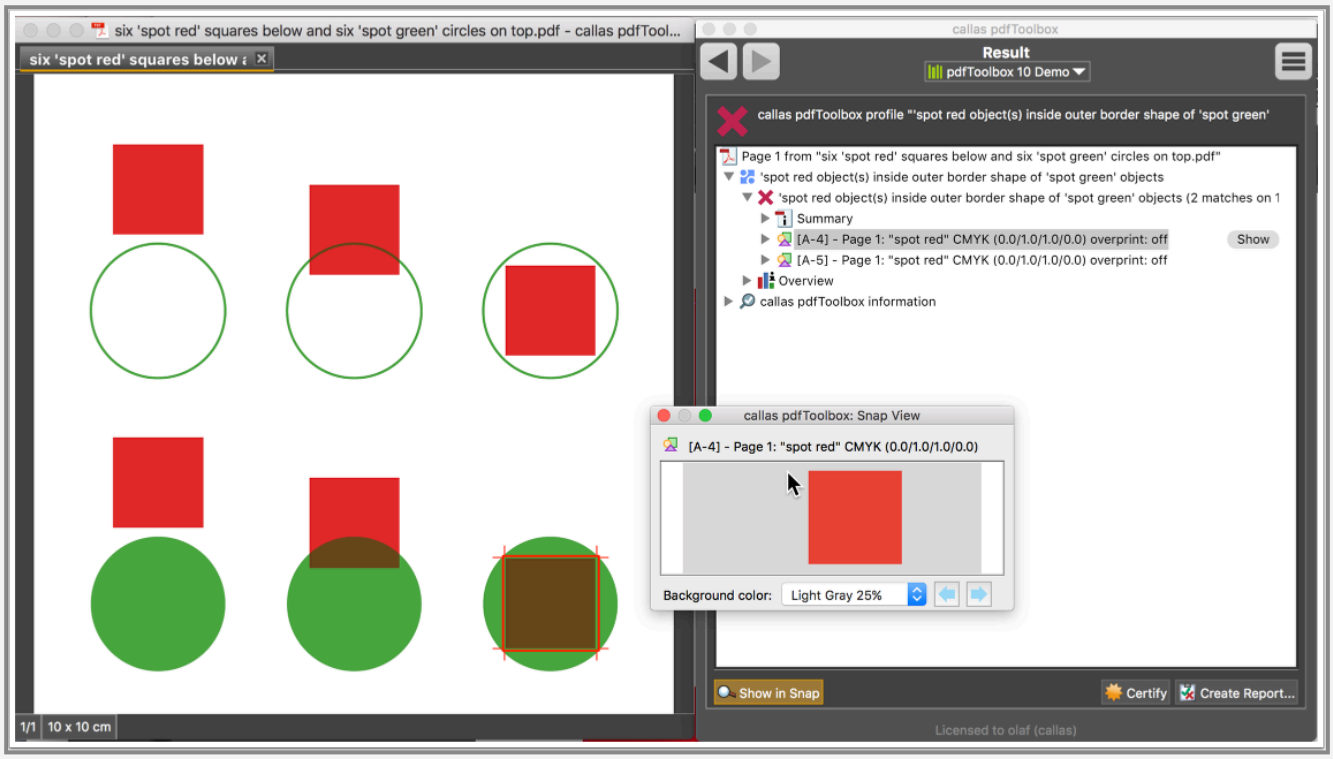
'spot_red'_object(s)_inside_outer_border_shap.kfpx

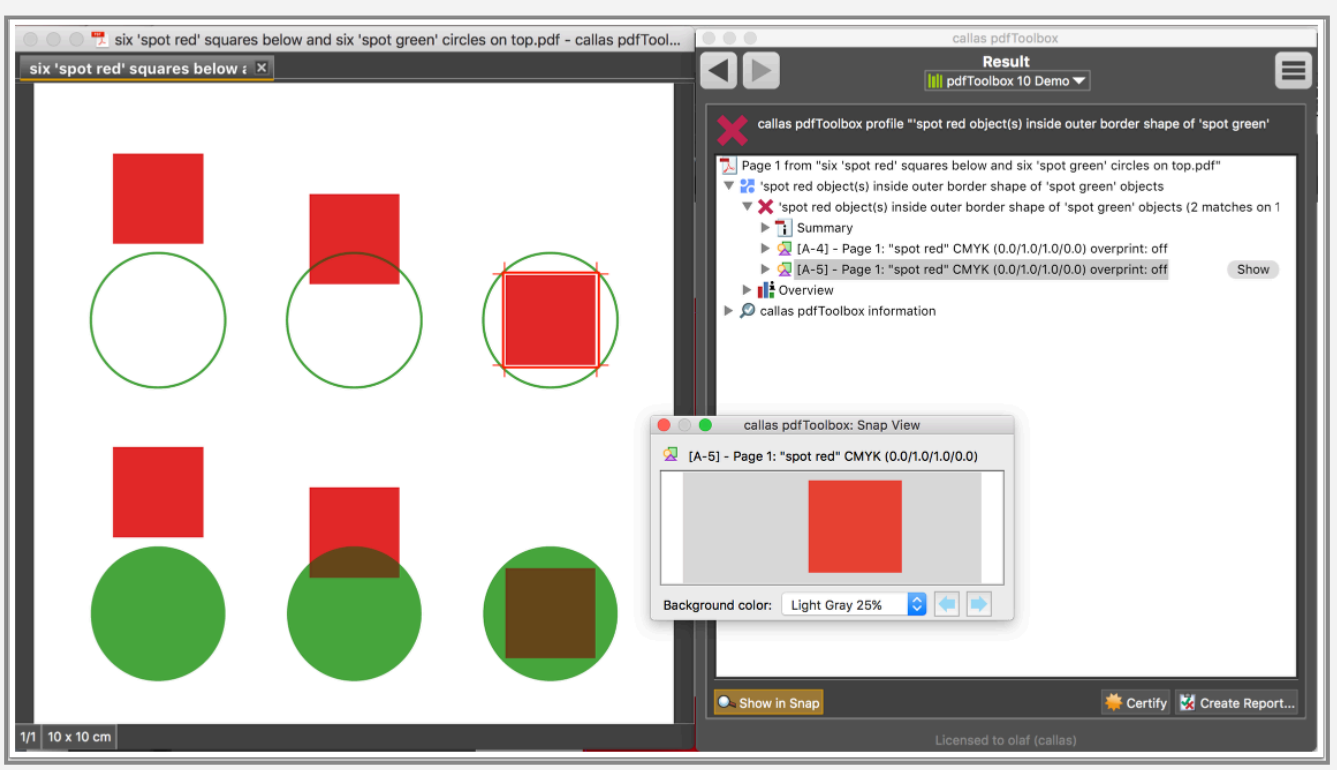
six_'spot_red'_squares_below_and_six_'spot_gre.pdf

In the two screenshots shown below, it can be seen how two spot red squares completely inside the area defined by the shape defined by the outer border of the spot green circles (regardless whether they are filled or just stroked) are detected.

This is different from what the property "Object inside other object" would detect – for details see the article Inside versus outside: Object inside other object.

Note: In the sample file, first six 'spot red' squares are painted, followed by six ' spot green' circles.




six 'spot red' squares below and six 'spot green' circles on top.pdf - callas pdfTool...
six 'spot red' squares below a
1/1
10 x 10 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "'spot red object(s) inside outer border shape of 'spot green'
Page 1 from "six 'spot red' squares below and six 'spot green' circles on top.pdf"
'spot red object(s) inside outer border shape of 'spot green' objects
'spot red object(s) inside outer border shape of 'spot green' objects (2 matches on 1
Summary
[A-4] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
[A-5] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
callas pdfToolbox: Snap View
[A-5] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)

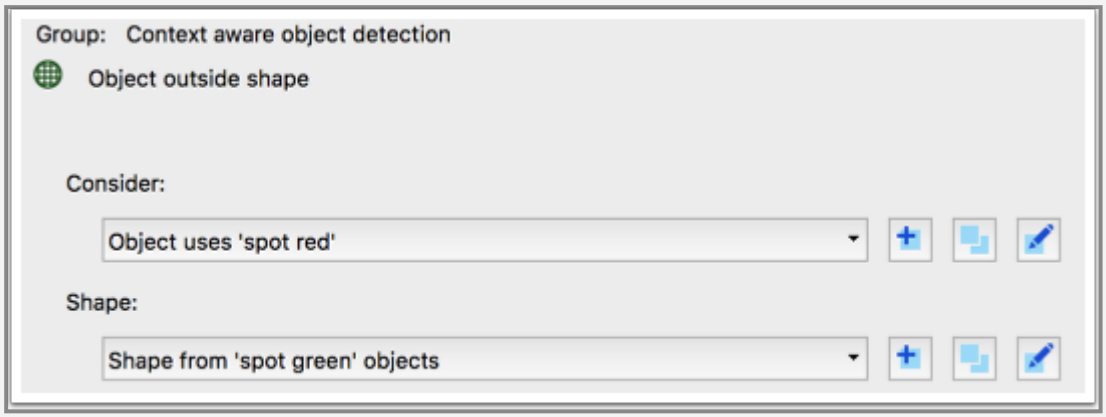
24.15 Inside versus outside: Object outside shape

The property "Object outside shape" detects whether an object is outside the area defined by a shape. In this context it is irrelevant, whether objects are on top or below, and also whether objects are opaque or transparent.

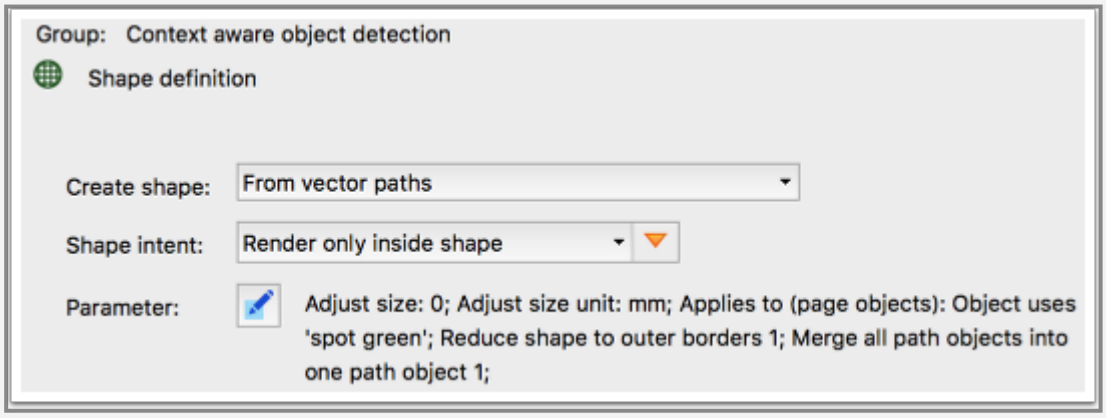
Shapes provide a means to establish a definition for an area in a very flexible manner, starting from simple shapes such as rectangles, areas defined by vector objects, or areas derived from rendering some or all of the content of a page and then turning the border of the rendered content into a definition of an area. For details see the article "Shapes" property for use in "Context aware object detection" checks.

The configuration options for a check based on the "Object outside shape" property can be seen below.

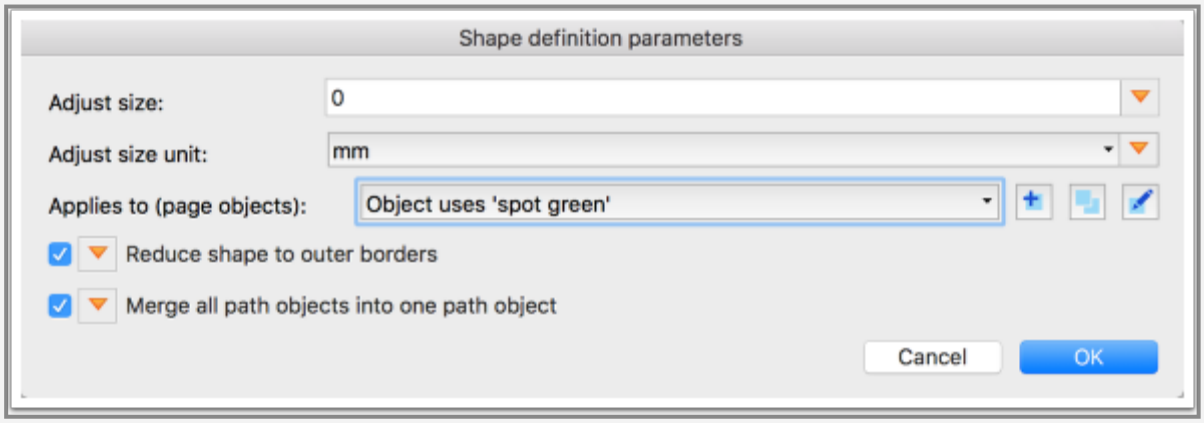
The "Consider" option defines the set of objects for which it shall be determined whether any of these objects is outside the shape defined by the "Check against" option.



In the example used below, the shape is derived from a subset of the vector objects on the page.



The subset of vector objects on the page to be taken into account for establishing the shape are 'spot green' vector objects. In the example used below, in order to achieve the desired effect, the option "Reduce shape to outer border" has been checked.



Sample files

The example below makes use of the check *'spot red' object(s) outside outer border shape of 'spot green' objects.kfpx.kfpx* and the PDF file *six 'spot red' squares below and six 'spot green' circles on top.pdf* which are available for download:

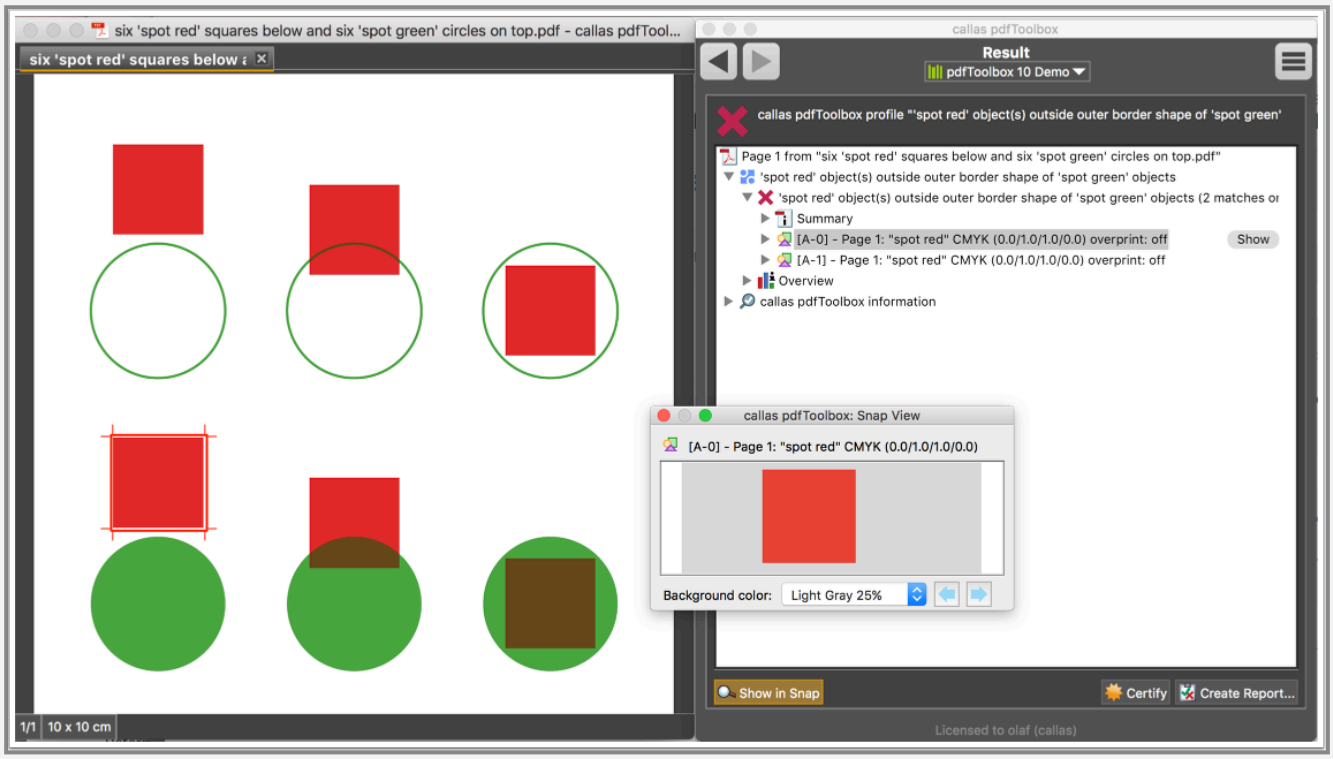
'spot_red'_object(s)_outside_outer_border_sha.kfpx

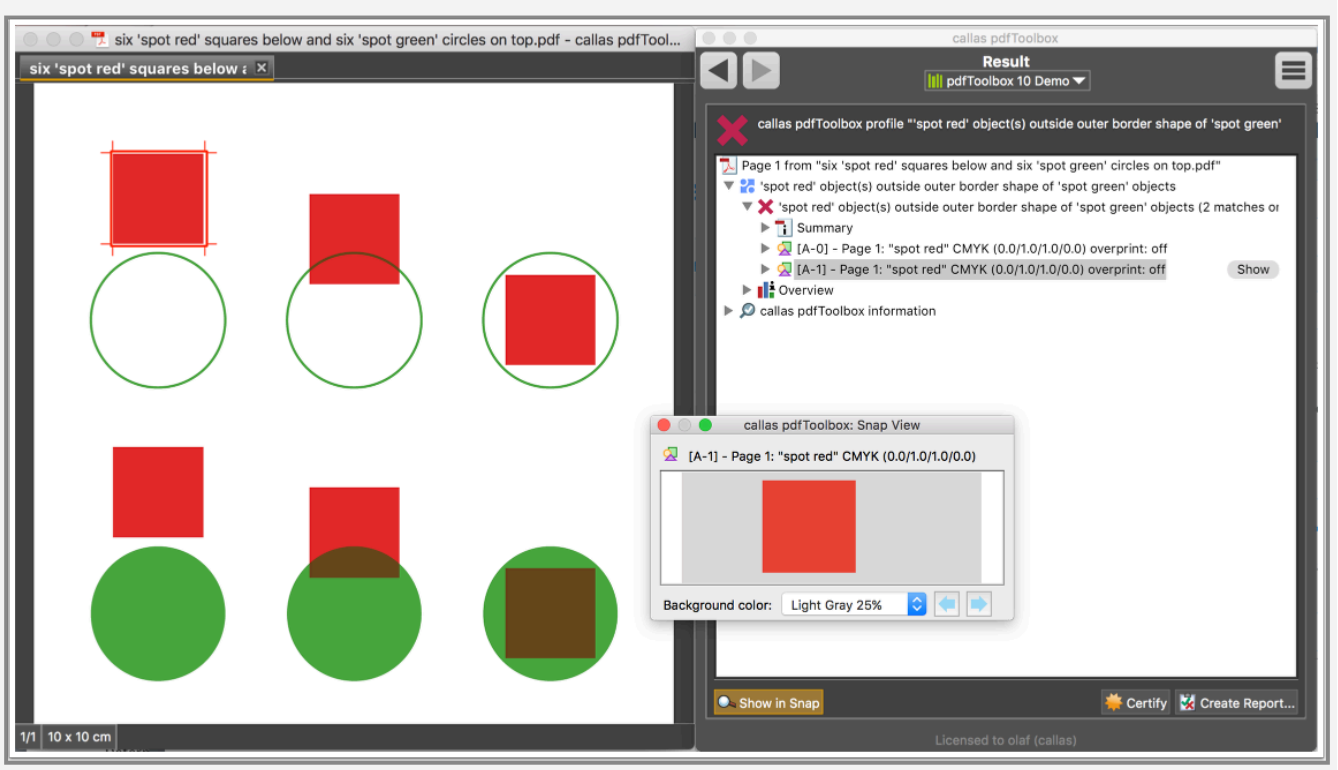
six_'spot_red'_squares_below_and_six_'spot_gre.pdf

In the two screenshots shown below, it can be seen how two spot red squares completely outside the area defined by the shape defined by the outer border of the spot green circles (regardless whether they are filled or just stroked) are detected.

This is different from what the property "Object outside other object" would detect – for details see the article Inside versus outside: Object outside other object.

Note: In the sample file, first six 'spot red' squares are painted, followed by six ' spot green' circles.



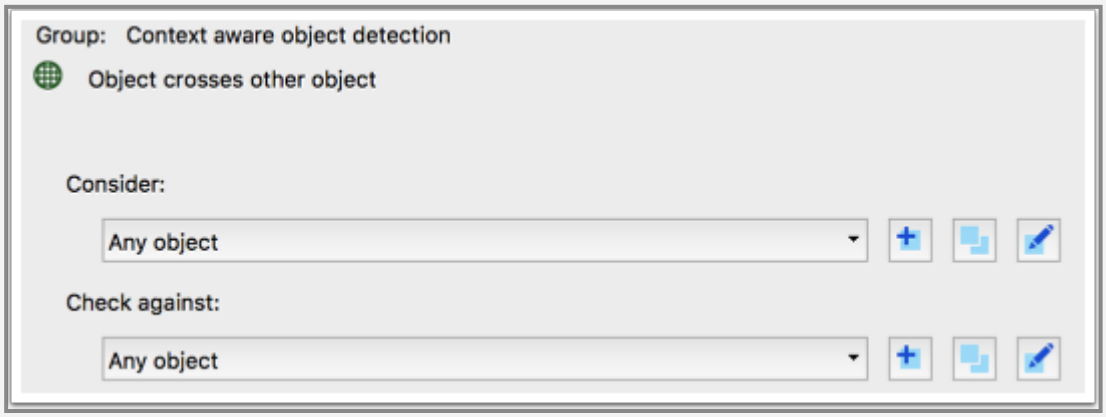

six 'spot red' squares below and six 'spot green' circles on top.pdf - callas pdfTool...
six 'spot red' squares below a
1/1
10 x 10 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "'spot red' object(s) outside outer border shape of 'spot green'
Page 1 from "six 'spot red' squares below and six 'spot green' circles on top.pdf"
'spot red' object(s) outside outer border shape of 'spot green' objects
'spot red' object(s) outside outer border shape of 'spot green' objects (2 matches or
Summary
[A-0] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
[A-1] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
callas pdfToolbox: Snap View
[A-1] - Page 1: "spot red" CMYK (0.0/1.0/1.0/0.0)
Background color:
Light Gray 25%
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)

24.16 Inside versus outside: Object crosses other object

The property "Object crosses other object" detects graphics objects whose border crosses the border of other graphics objects.

The configuration options for a check based on the "Object crosses shape" property can be seen below.

The "Consider" option defines the set of objects for which it shall be determined whether any of these objects' border crosses the border of some other object from the set of objects defined by "Check against".



Sample files

The two examples below makes use of the checks ***Object crosses other object(s).kfpx*** and ***'spot red' object crosses 'spot green' object(s).kfpx***, as well as the PDF file ***'spot red' squares with 'spot green' squares (set to overprint).pdf*** which are available for download:

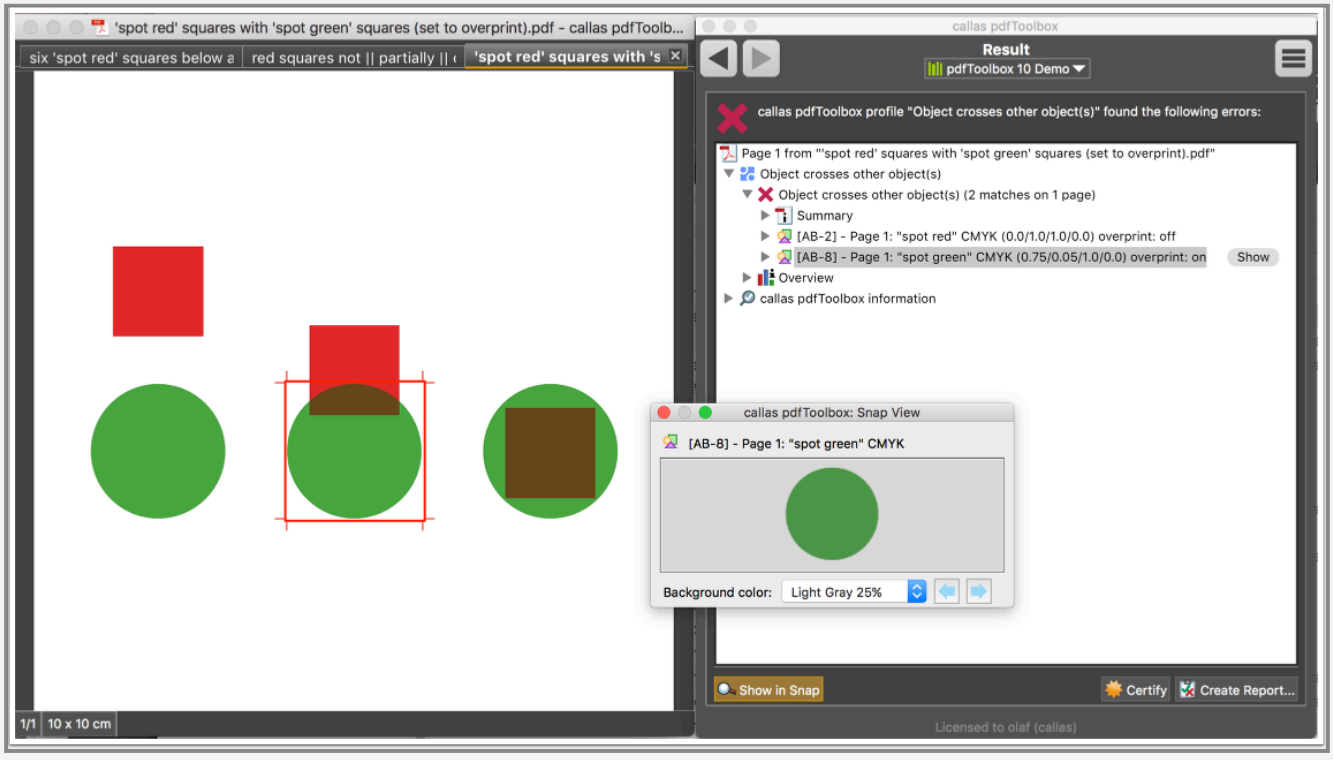
'spot_red'_object_crosses_'spot_green'_object.kfpx

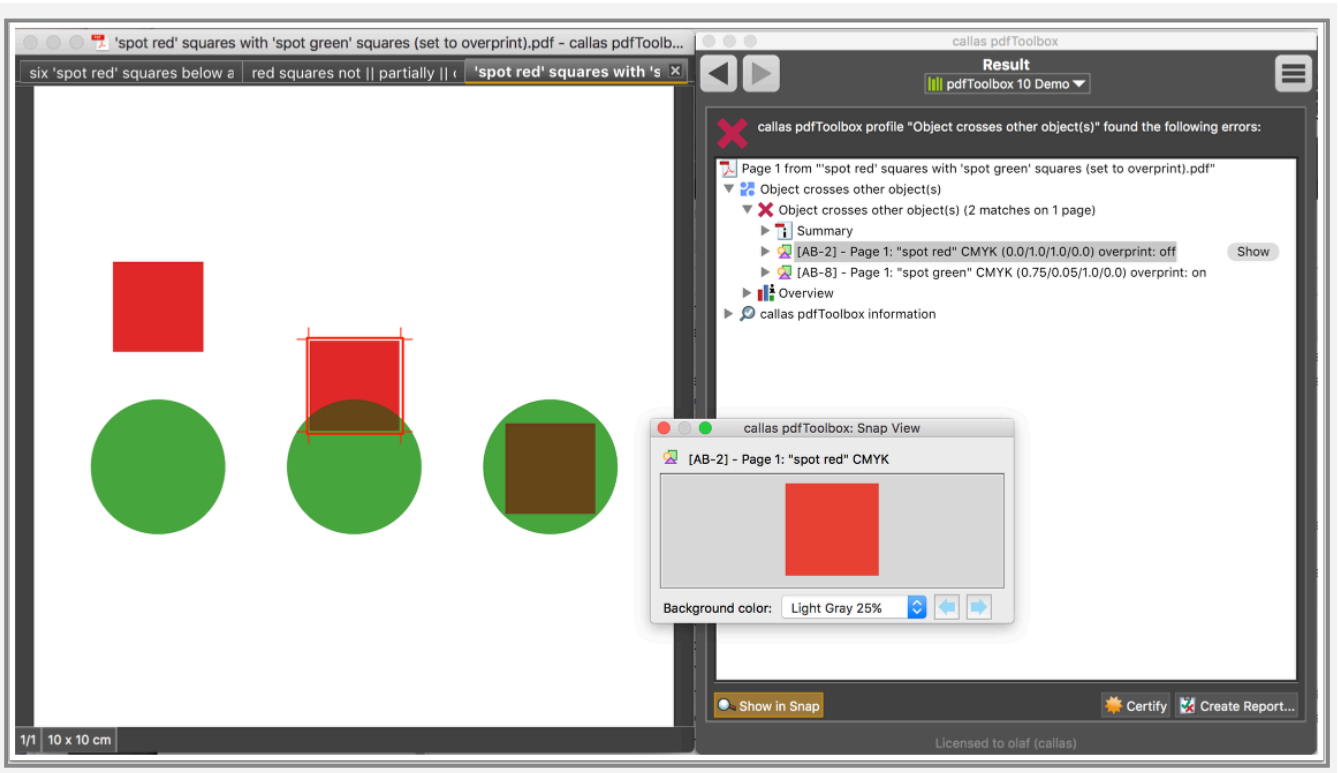
Object_crosses_other_object(s).kfpx

'spot_red'_squares_with_'spot_green'_squares_(.pdf

Running the check *Object crosses other object(s)* detects two objects:

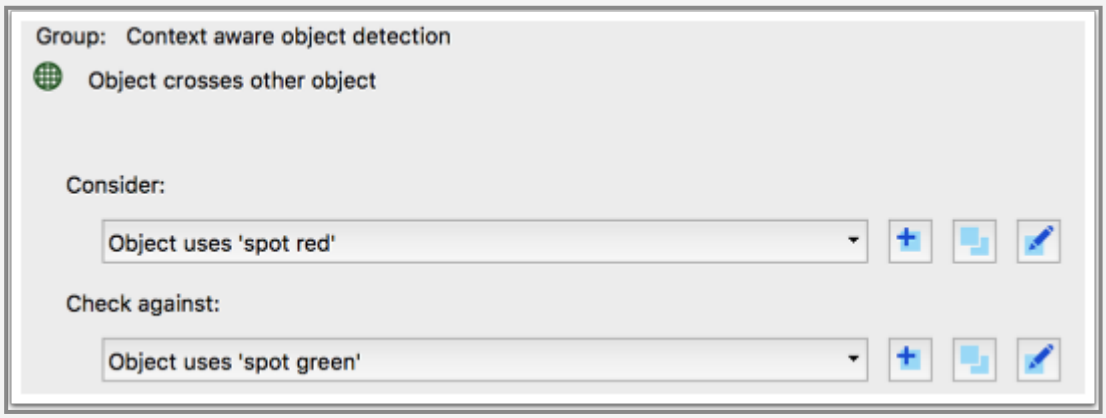
- the red square in the center: because its border crosses the border of the green square that overlaps it.
- the green square in the center: because its border crosses the border of the red square that it overlaps.





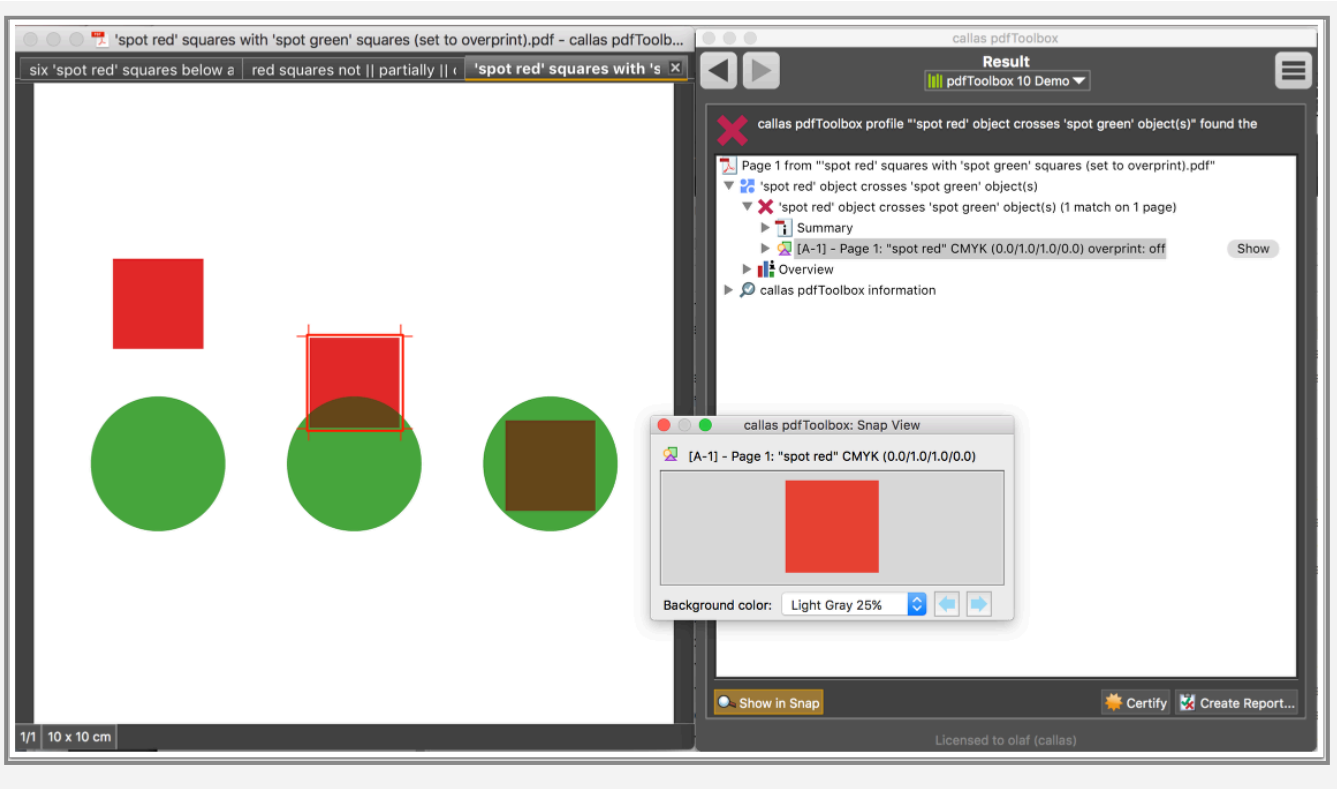
In order to not find both halves of each and every pair of objects that cross each other, one can be more specific and only check one set of objects against a second set of different objects – in this example 'spot red' objects against 'spot green' objects.

A suitable setup for a check such as ***'spot red' object crosses 'spot green' object(s).kfpx*** is shown below:



Running this check on the sample PDF ***'spot red' squares with 'spot green' squares (set to overprint).pdf*** leads to the

following result – with just the red square in the center being detected:



24.17 Visibility: Object is invisible

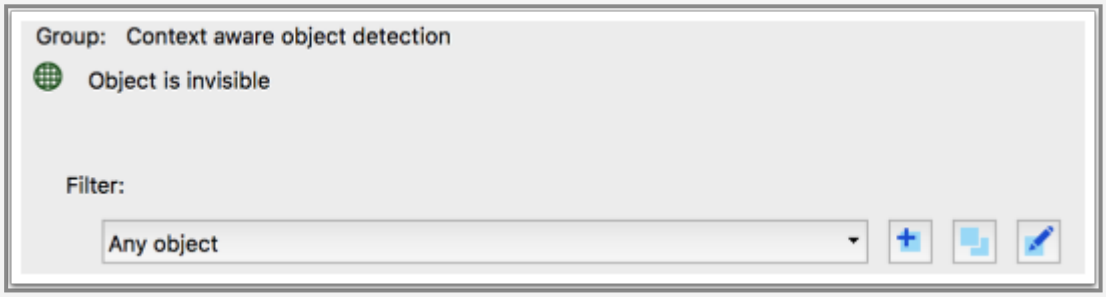
The property "Object is invisible" detects whether an object is invisible.

An object can be invisible because of the following reasons:

- it is completely clipped (whether by a clipping path, or because it is outside of the MediaBox/CropBox, or a combination thereof)
- it is completely obliterated, i.e. covered by one or more opaque objects that hide all of the object and thus suppress its rendering on the page
- any combination of the above; for example, one part of an object could be obliterated, and the rest could be clipped

The configuration options for a check based on the "Object is invisible" property can be seen below.

The "Filter" option defines the set of objects for which it shall be determined whether they are invisible.



Sample files

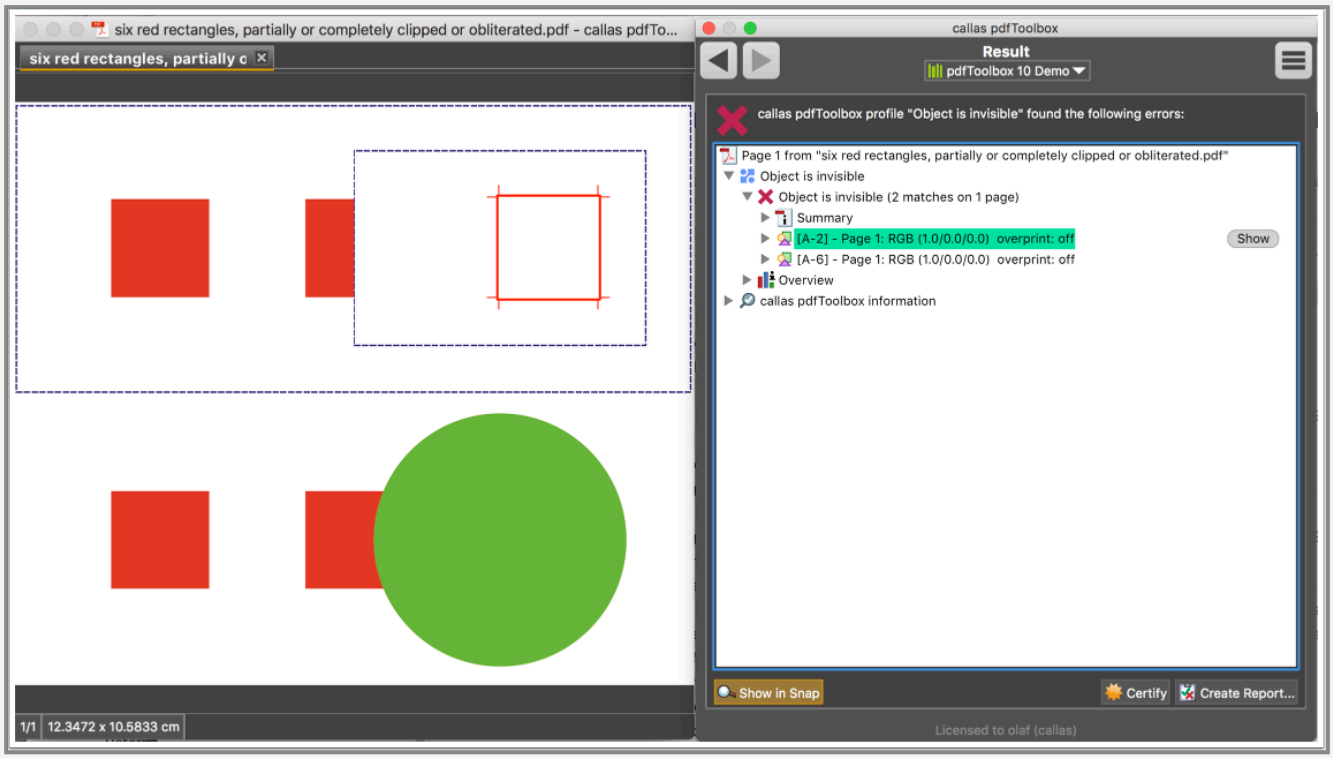
The example below makes use of the check ***Object is invisible.kfpx*** and the PDF file ***six red rectangles, partially or completely clipped or obliterated.pdf*** which are available for download:

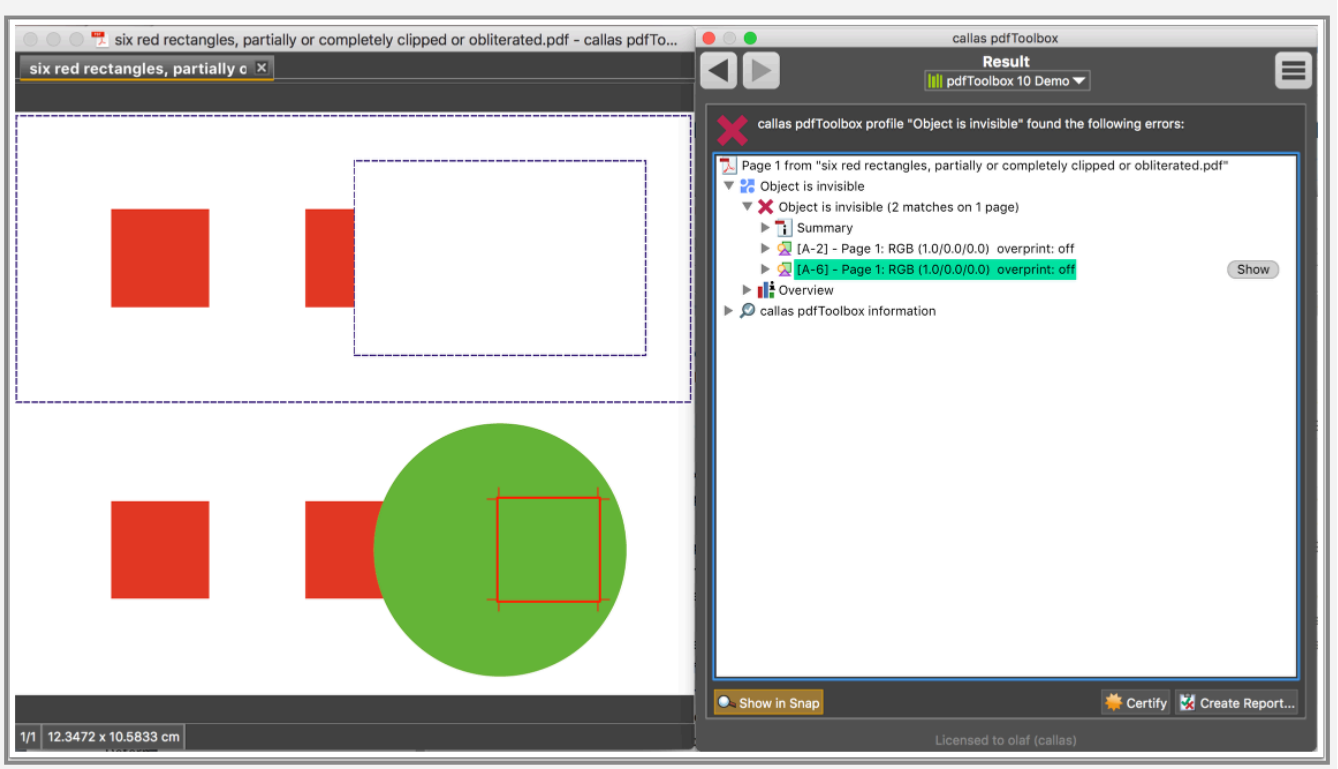
Object_is_invisible.kfpx

six_red_rectangles__partially_or_completely_cl.pdf

In the screenshot shown below, it can be seen how two red squares are detected as being invisible.

Note: In the sample file, first six red squares are painted, followed by one green circle. The blue dashed lines indicate where clipping is in effect.




six red rectangles, partially or completely clipped or obliterated.pdf - callas pdfTo...
six red rectangles, partially c
1/1
12.3472 x 10.5833 cm
callas pdfToolbox
Result
pdfToolbox 10 Demo
callas pdfToolbox profile "Object is invisible" found the following errors:
Page 1 from "six red rectangles, partially or completely clipped or obliterated.pdf"
Object is invisible
Object is invisible (2 matches on 1 page)
Summary
[A-2] - Page 1: RGB (1.0/0.0/0.0) overprint: off
[A-6] - Page 1: RGB (1.0/0.0/0.0) overprint: off
Show
Overview
callas pdfToolbox information
Show in Snap
Certify
Create Report...
Licensed to olaf (callas)

24.18 Visibility: Object is visible

The property "Object is visible" detects whether an object is visible.

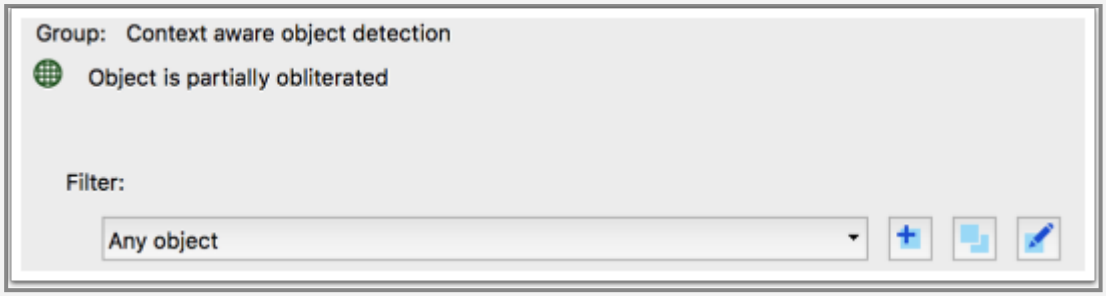
Any object that is not invisible is considered visible. For a discussion on how invisible objects are determined, please see the article Visibility: Object is invisible.

24.19 Visibility: Object is partially obliterated

The property "Object is partially obliterated" detects objects that are partially obliterated by other objects (and are thus only partially visible on the page). An object is considered partially obliterated in this context if at least one other opaque object (i.e. neither overprinting nor transparent) is above it and overlaps it, thus partially suppressing rendering of that object in the overlap area.

For detecting complete obliteration see the article Visibility: Object is completely obliterated.

The "Filter" option defines the set of objects for which it shall be determined whether they are partially obliterated.



Sample files

The example below makes use of the check ***Object is partially obliterated.kfpx*** and the PDF file ***three red rectangles, one half obliterated, one fully obliterated.pdf*** which are available for download:

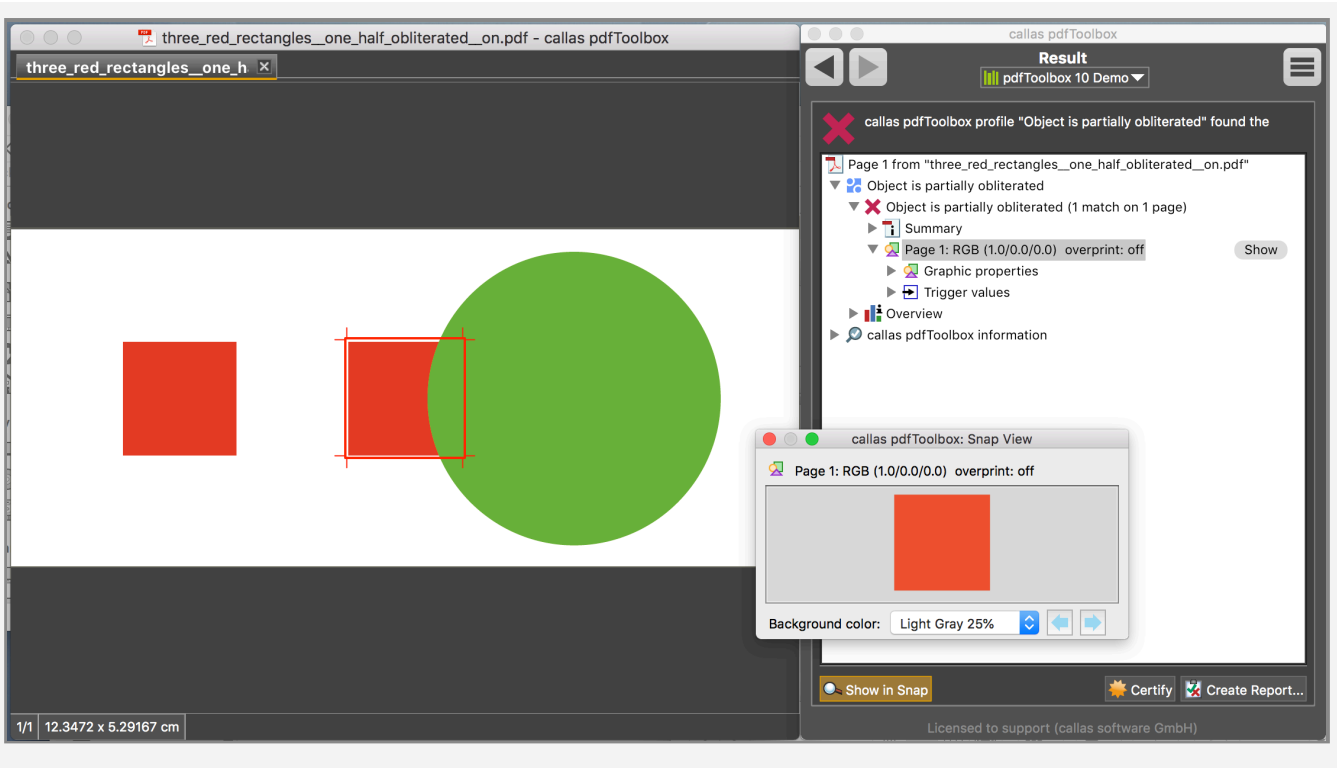
Object_is_partially_obliterated.kfpx

three_red_rectangles__one_half_obliterated__on.pdf

The page shown in the screenshot below contains three evenly spaced red squares, where the middle one is half

obliterated, and the rightmost one is fully obliterated (and thus not visible).

It can be seen below how the partially obliterated (and thus only partially visible) red square in the middle is detected.

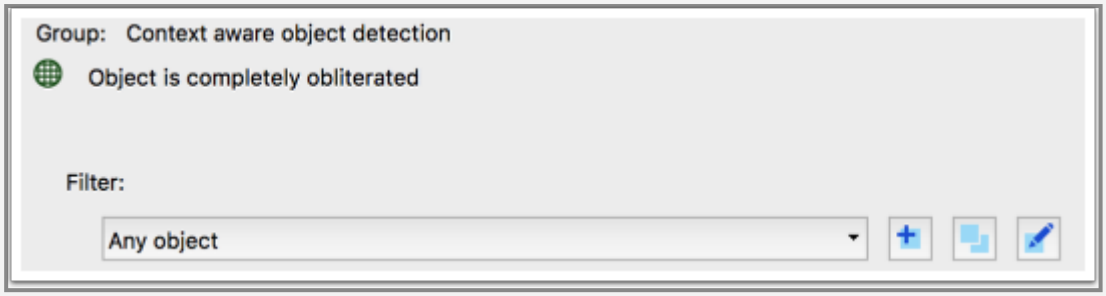


24.20 Visibility: Object is completely obliterated

The property "Object is completely obliterated" detects objects that are completely obliterated by other objects (and are thus not visible on the page). An object is considered completely obliterated in this context if at least one other opaque object (i.e. neither overprinting nor transparent) above it fully overlaps it, thus completely suppressing rendering of that object.

For detecting partial obliteration see the article Visibility: Object is partially obliterated.

The "Filter" option defines the set of objects for which it shall be determined whether they are completely obliterated.



Sample files

The example below makes use of the check ***Object is completely obliterated.kfpx*** and the PDF file ***three red rectangles, one half obliterated, one fully obliterated.pdf*** which are available for download:

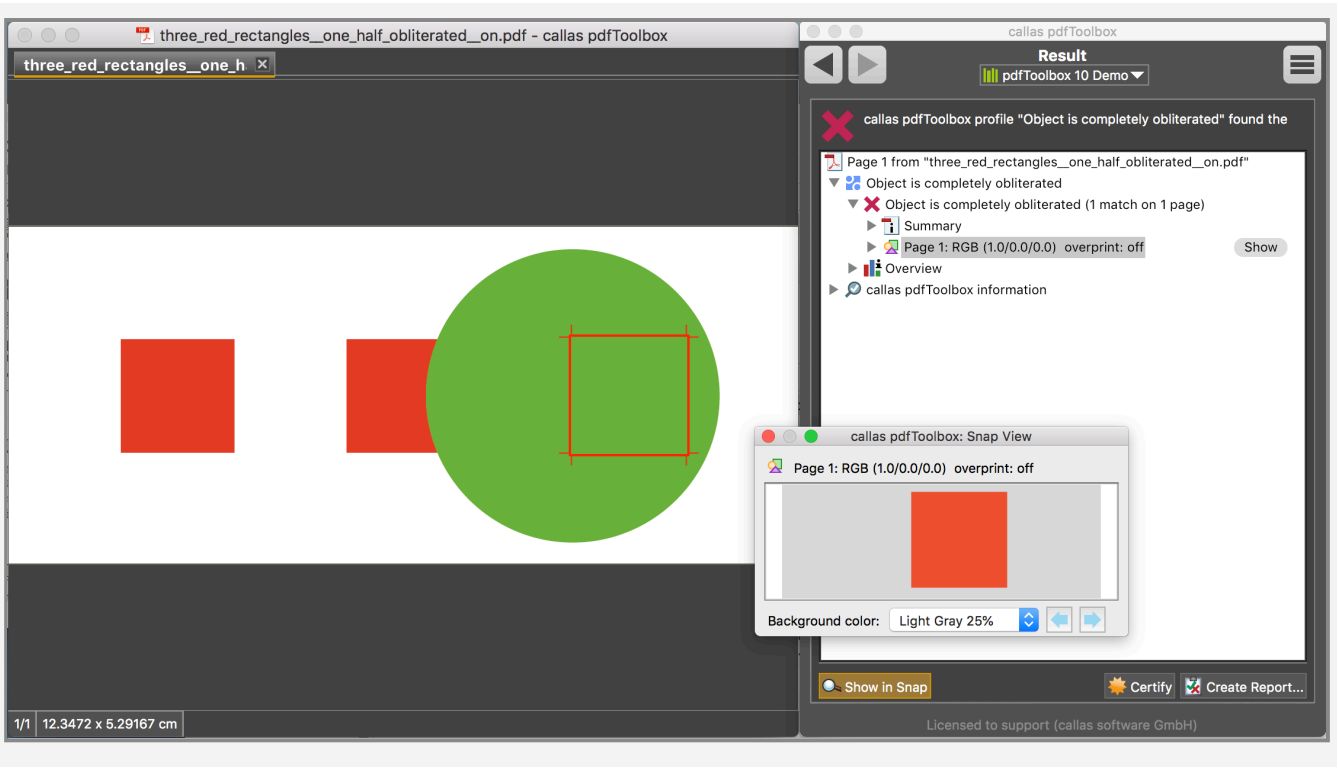
Object_is_completely_obliterated.kfpx

three_red_rectangles__one_half_obliterated__on.pdf

The page shown in the screenshot below contains three evenly spaced red squares, where the middle one is half

obliterated, and the rightmost one is fully obliterated (and thus not visible).

It can be seen below how the completely obliterated (and thus invisible) red square on the right is detected.



24.21 Visibility: Object is partially clipped

The property "Object is partially clipped" detects partially clipped objects (which are thus only partially visible on the page).

Clipping occurs in the following ways:

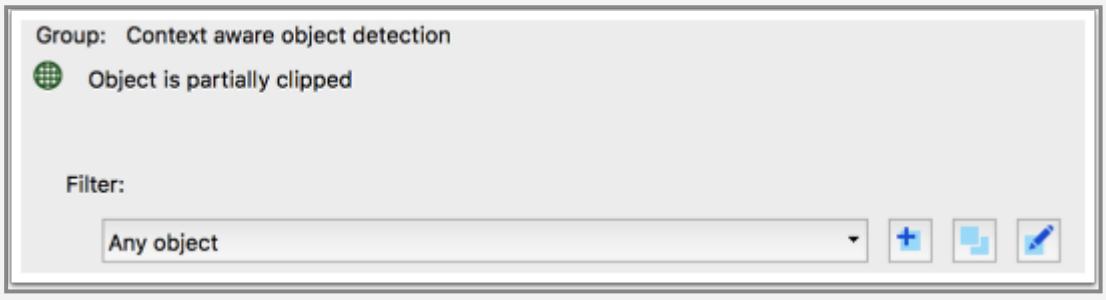
- through an explicit clipping path, or a combination of several clipping plaths
- through the implicit clipping path defined by the Crop-Box, or in its absence the MediaBox, of a page; an object is always completely clipped if it is outside the CropBox, or in its absence the MediaBox, of a page
- through the combination of explicit clipping paths and the implicit clipping path defined by the CropBox, or in its absence the MediaBox, of a page

The following objects will not be found by this property:

- objects that are completely clipped
- objects that are only partially (in)visible because they are partially obliterated

For detecting complete clipping see article Visibility: Object is completely clipped.

The "Filter" option defines the set of objects for which it shall be determined whether they are partially clipped.



Sample files

The example below makes use of the check ***Object is partially clipped.kfpx*** and the PDF file ***three red rectangles, one half***

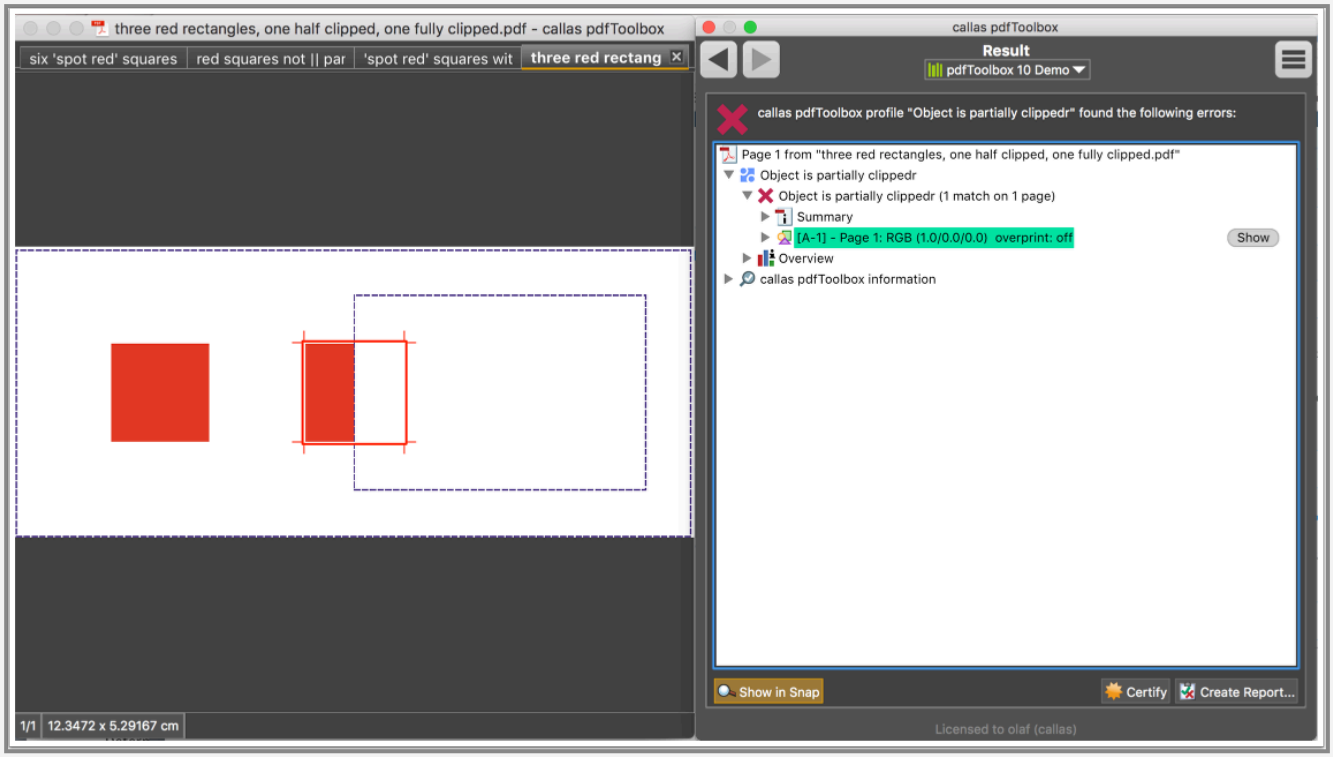
clipped, one fully clipped.pdf which are available for download:

Object_is_partially_clipped.kfpx

three_red_rectangles__one_half_clipped__one_fu.pdf

The page show in the screenshot below contains three evenly spaced red squares, where the middle one is half clipped, and the rightmost one is fully clipped (and thus not visible). The clipping paths in effect are indicated by the blue dashed lines.

It can be seen below how the partially clipped red square in the middle is detected.



24.22 Visibility: Object is completely clipped

The property "Object is completely clipped" detects completely clipped objects (which are thus not visible on the page).

Clipping occurs in the following ways:

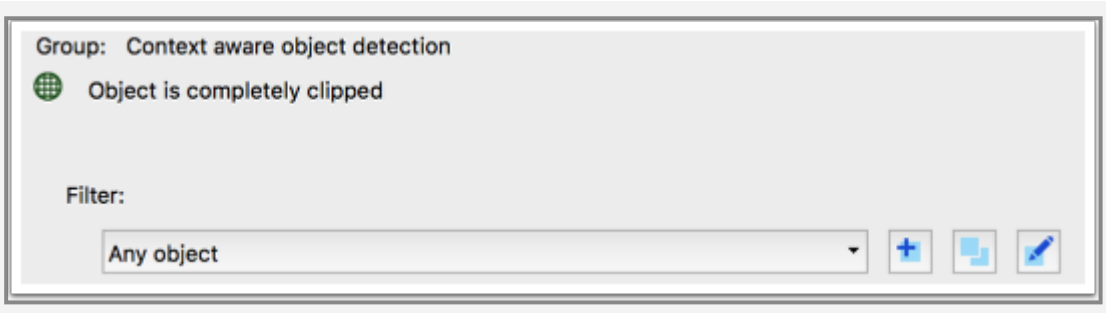
- through an explicit clipping path, or a combination of several clipping plaths
- through the implicit clipping path defined by the CropBox, or in its absence the MediaBox, of a page; an object is always completely clipped if it is outside the CropBox, or in its absence the MediaBox, of a page
- through the combination of explicit clipping paths and the implicit clipping path defined by the CropBox, or in its absence the MediaBox, of a page

The following objects will not be found by this property:

- objects that are only partially clipped
- objects that are invisible because they are partially clipped and also partially or fully obliterated

For detecting partial clipping see article Visibility: Object is partially clipped.

The "Filter" option defines the set of objects for which it shall be determined whether they are completely clipped.



Sample files

The example below makes use of the check ***Object is completely clipped.kfpx*** and the PDF file ***three red rectangles, one***

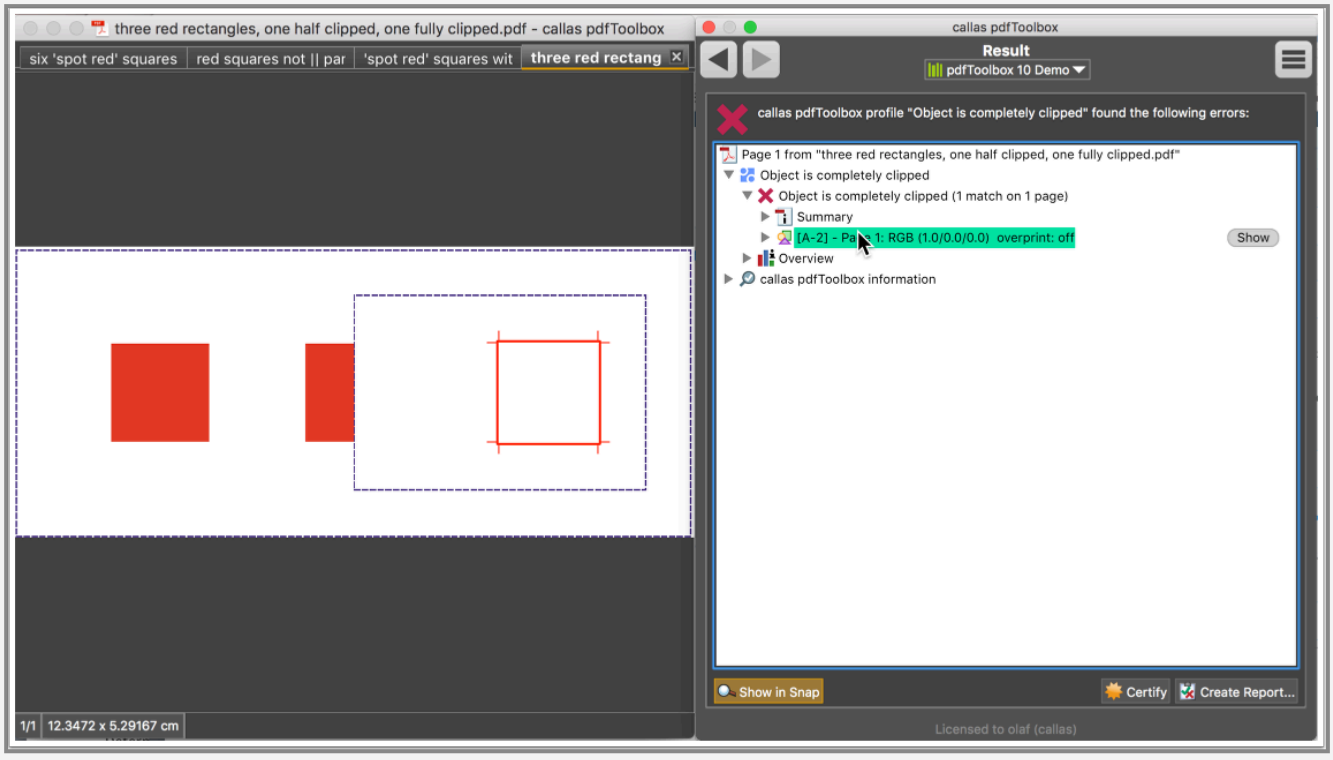
half clipped, one fully clipped.pdf which are available for download:

Object_is_completely_clipped.kfpx

three_red_rectangles__one_half_clipped__one_fu.pdf

The page shown in the screenshot below contains three evenly spaced red squares, where the middle one is half clipped, and the rightmost one is fully clipped (and thus not visible). The clipping paths in effect are indicated by the blue dashed lines.

It can be seen below how the completely clipped (and thus invisible) red square on the right is detected.



24.23 Advanced "Context aware object detection" property

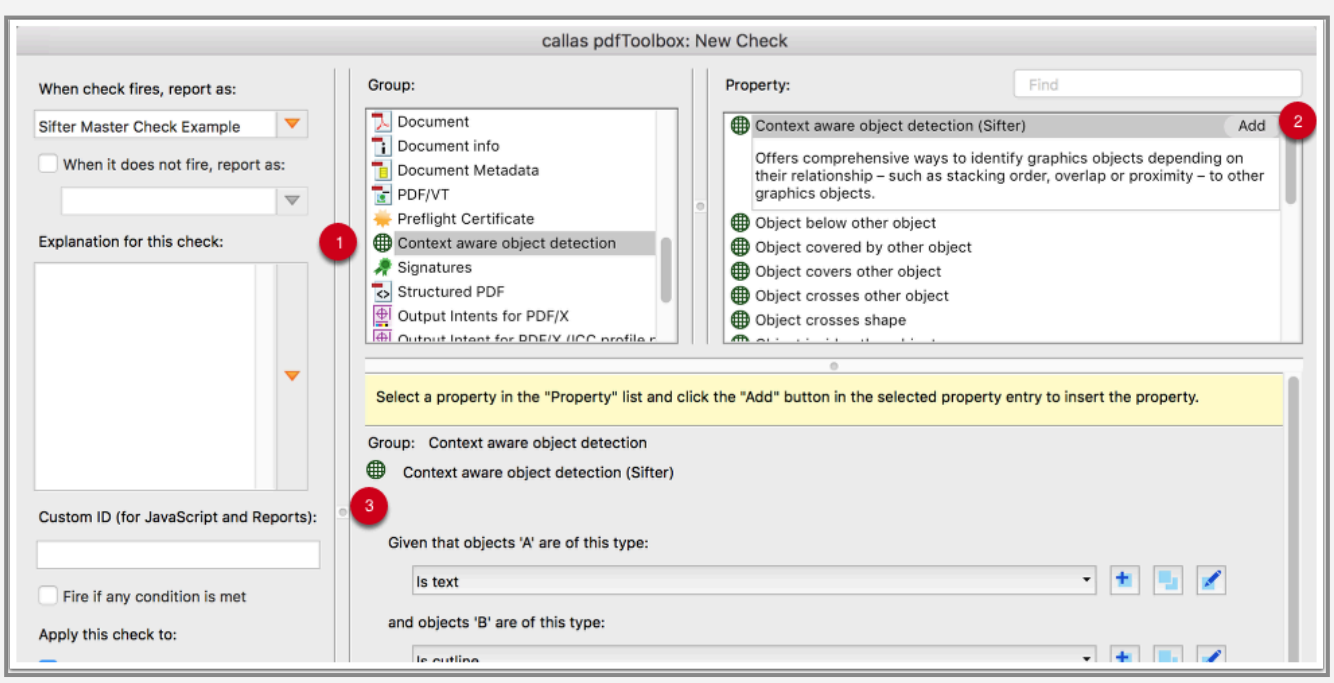
The property *Context aware object detection (Sifter)* provides extensive configurability for context aware object detection, going far beyond what can be achieved with the various more specialized properties based on Context aware object detection.

In most cases it will not be necessary to use the advanced Context aware object detection property, but whenever a need arises to set up more specific checks, the instructions below offer the necessary guidance.

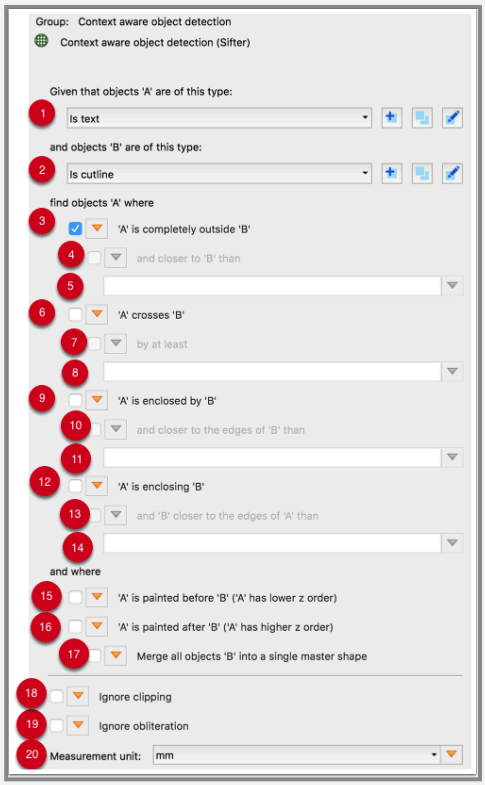
Note: Most of the other Context aware object detection properties are essentially pre-configured variants of the advanced *Context aware object detection (Sifter)* property, with the main effect that they are much more straightforward to use.

Configuring the «Context aware object detection (Sifter)» master property

The *Context aware object detection (Sifter)* property can easily be located in the *New Check* or *Edit Check* dialog, just select the "Context aware object detection" entry in the "Group" list and click the small yellow "Add" button to the left of the entry *Context aware object detection (Sifter)* in the "Property" list, to add it to the configuration area of the dialog.



Configuration options for condition based on a ***Context aware object detection (Sifter)*** property:



1. check that defines the objects to test (*objects 'A'*); shapes as possibly used in objects 'B' cannot be used here
2. check that defines the objects (*objects 'B'*) or shape (*master shape*) against which to test objects 'A'
3. detect objects 'A' that are completely outside objects B resp. outside the master shape
4. whether to only detect those outside objects 'A' that are closer to objects 'B' or the master shape than a chosen threshold value (note: this can only be set if the checkbox under 17. is enabled)
5. the threshold value to use (note: the measurement unit is the same for for all threshold values and is defined under 20. "Measurement unit")
6. detect objects 'A' whose borders cross the border of objects 'B' resp. the the border of the master shape
7. whether to only detect those objects 'A' whose border crosses the border of objects 'B' or the border of the master shape by more than a chosen threshold value. This can be used to find out how far an object 'A' reaches into the inner area of an object 'B' or a master shape (note: this can only be set if the checkbox under 17. is enabled)
8. the threshold value to use (note: the measurement unit is the same for for all threshold values and is defined under 20. "Measurement unit")
9. detect objects 'A' that are completely enclosed by objects 'B' resp. enclosed by the master shape
10. whether to only detect those objects 'A' enclosed by objects 'B' or enclosed by the master shape that are closer to the (inner side of the) border of objects 'B' or the master shape than a chosen threshold value (note: this can only be set if the checkbox under 17. is enabled)
11. the threshold value to use (note: the measurement unit is the same for for all threshold values and is defined under 20. "Measurement unit")
12. detect objects 'A' that completely enclose objects 'B' resp. the master shape; "enclose" means in this context that object 'A' need to paint something in that area it e.g. does not cover a case where an object 'B' is inside of an outlined rectangle object 'A'
13. whether to only detect those objects 'A' enclosing objects 'B' or enclosing the master shape where objects 'B' or the master shape are closer to the (inner side of the) border of objects 'A' than a chosen threshold value (note: this can only be set if the checkbox under 17. is enabled)

14. the threshold value to use (note: the measurement unit is the same for for all threshold values and is defined under 20. "Measurement unit")
15. restrict object detection to those objects 'A' that are painted before the resp. objects 'B' (not meaningful for master shape); in other words: for any object 'A' only those objects 'B' will be considered that have not yet been drawn
16. restrict object detection to those objects 'A' that are painted after the resp. objects 'B' (not meaningful for master shape); in other words: for any object 'A' only those objects 'B' will be considered that have already been drawn
17. *[requires that 15. and 16. are checked]* instead of processing objects 'B' one by one, combine them into a single virtual object and use them as if they were a single path object (*master shape*)
18. optionally disregard all clipping effects; if checked, analysis is carried out as if not a single clipping path were present on the page (and no object would be considered invisible due to clipping effects)
19. optionally disregard all obliteration effects; if checked, analysis is carried out as if all objects were partially transparent (and no object would be considered invisible due to obliteration effects)
20. Measurement unit to use for the threshold values (see entries 5., 8., 11. and 14.). Available options are *mm*, *pt* and *inch*.

When more than one checkboxe under "find objects 'A' where" is active a hit will be generated if any of the active conditions is true (logical OR).

24.24 "Shapes" Property for use in "Context aware object detection" checks

The **Shapes** Property – which only works inside certain *Context aware object detection* Checks – uses the same concept on which the Shapes Fixup is based.

Where the Shapes Fixup establishes a shape to be used for creating filled or stroked graphics objects or for clipping, the Shapes Property for use in "Context aware object detection" Checks establishes a shape that serves as a context against which graphics objects can be tested.

The options for defining a shape are the same in the Property and in the Fixup. For a discussion of the Shapes Fixup see the chapter on Shapes Fixups.

The following Properties require a reference to a check based on the Shapes Property:

- Object crosses shape
- Object inside shape
- Object outside shape
- Object reaches into edge area of a shape

Example: Setting up an 'Object inside "Cutline"' check using a shape Property

Sample files

This example makes use of the check ***Object inside "Dieline" (uses Shape property).kfpx*** and the PDF file ***spot green circles inside and outside red "Dieline".pdf***. In addition the fixup ***Create 'pink' overlay from "Cutline".kfpx*** and the result from applying it to the sample file, ***spot green circles inside and outside red "Dieline" with pink overlay.pdf***, are also provided.

All these files are available for download:

Object_inside__Dieline__(uses_Shape_property).kfpx

spot_green_circles_inside_and_outside_red_"Die.pdf

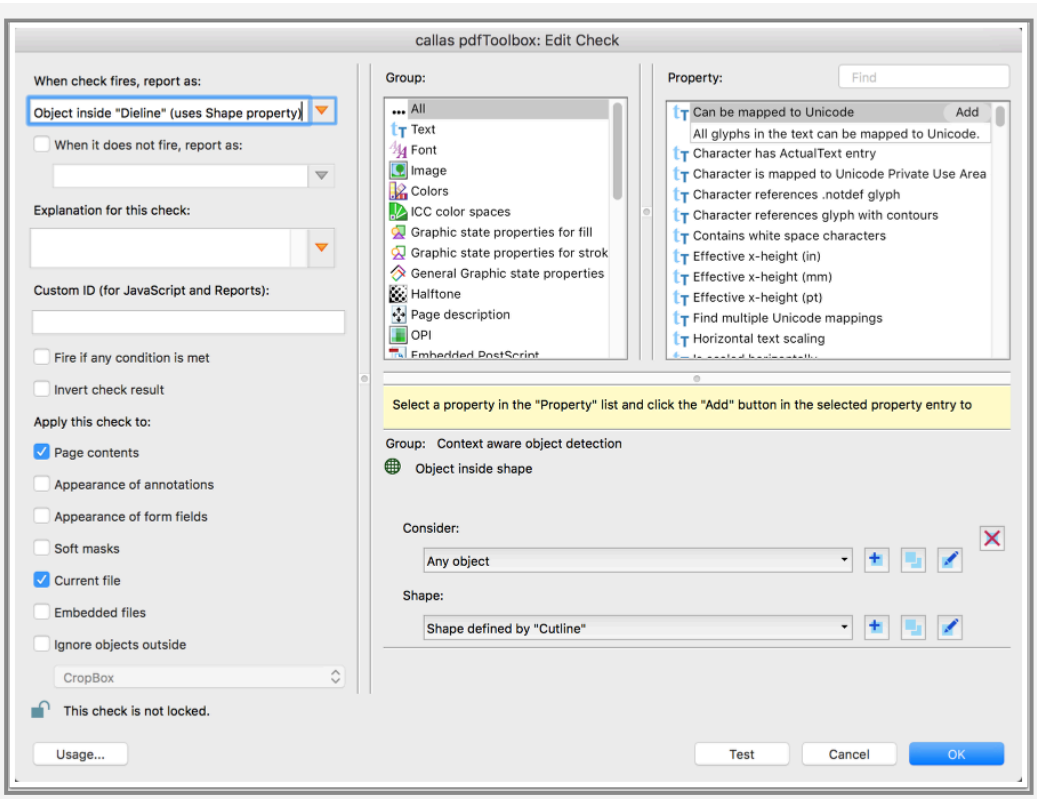
Create_'pink'_overlay_from__Cutline_.kfpx

spot_green_circles_inside_and_outside_red_"D-1.pdf

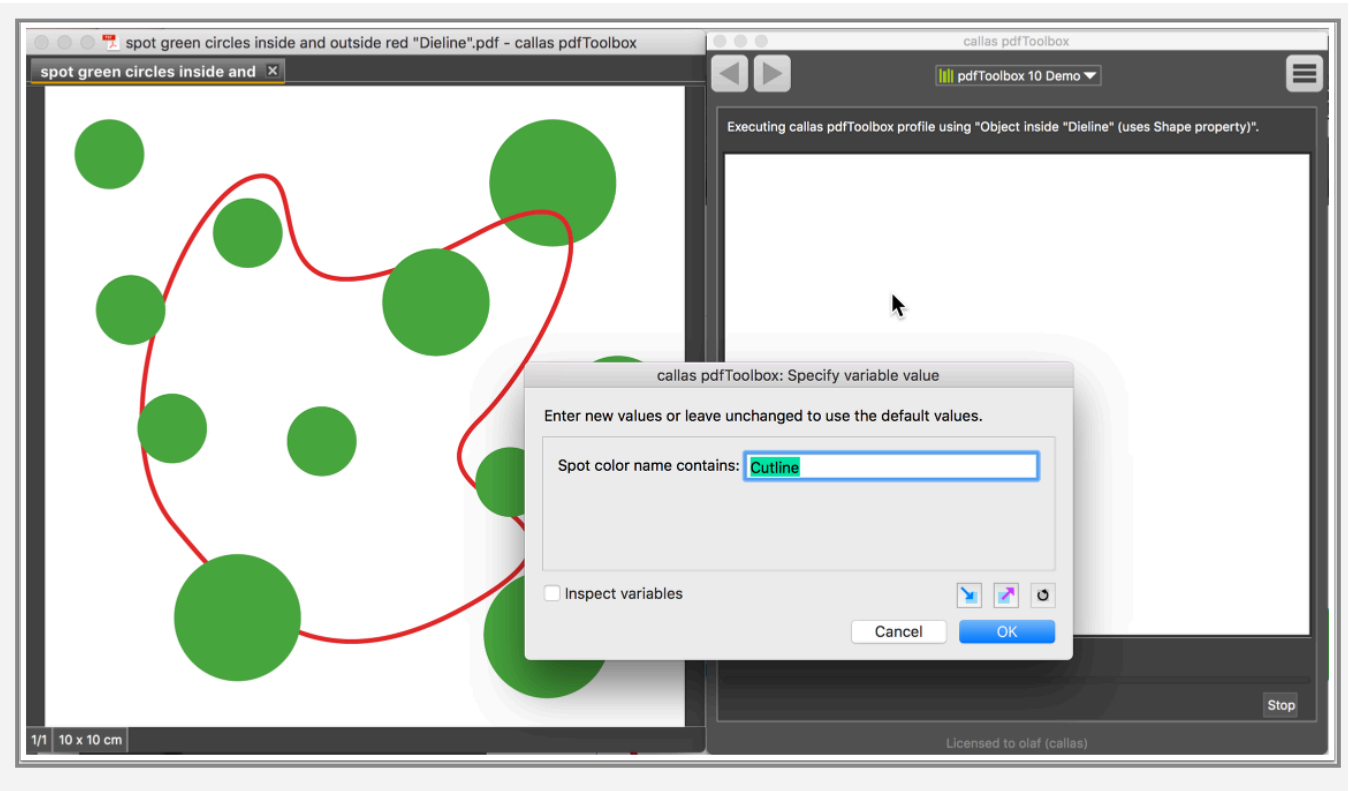
Setting up the shape based check

See below how to set up the intended check based on the "Object inside shape" Property.

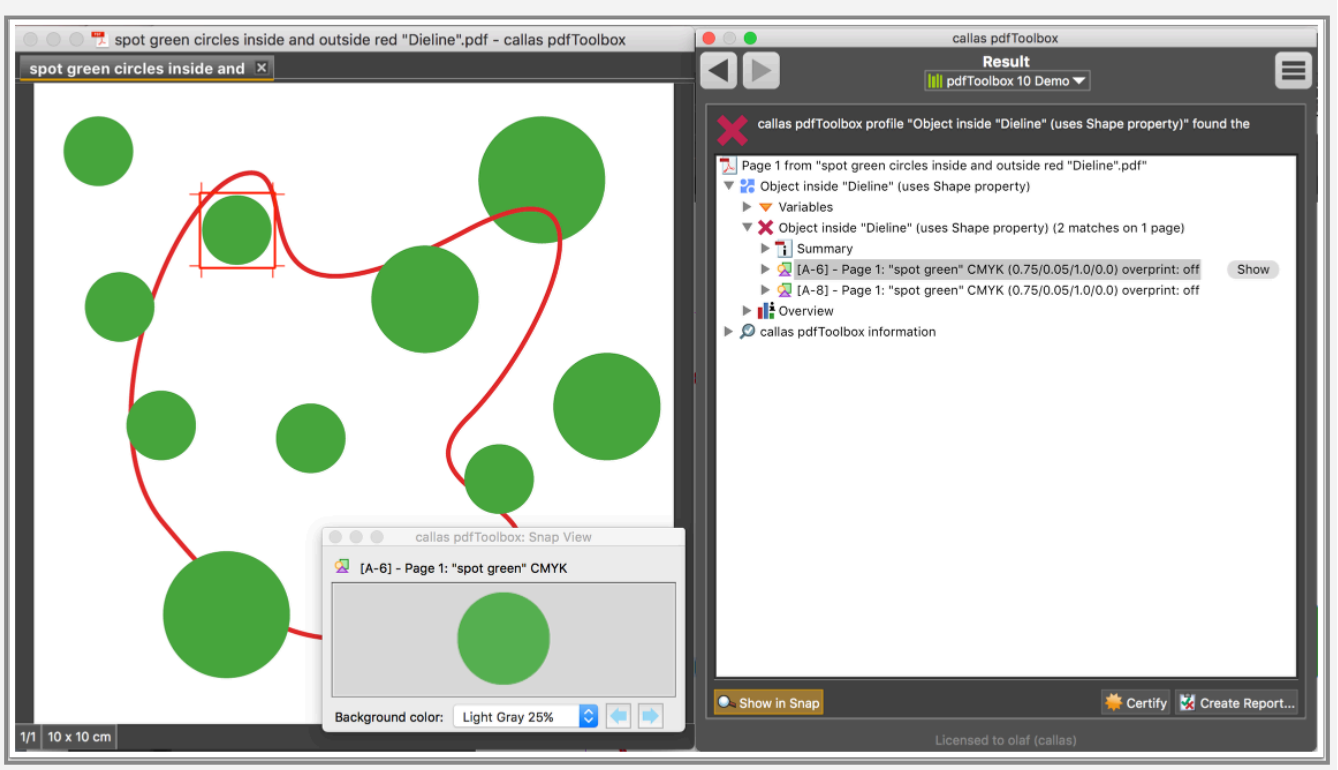
The goal is to detect objects on the page that are inside the area enclosed by the "Cutline".

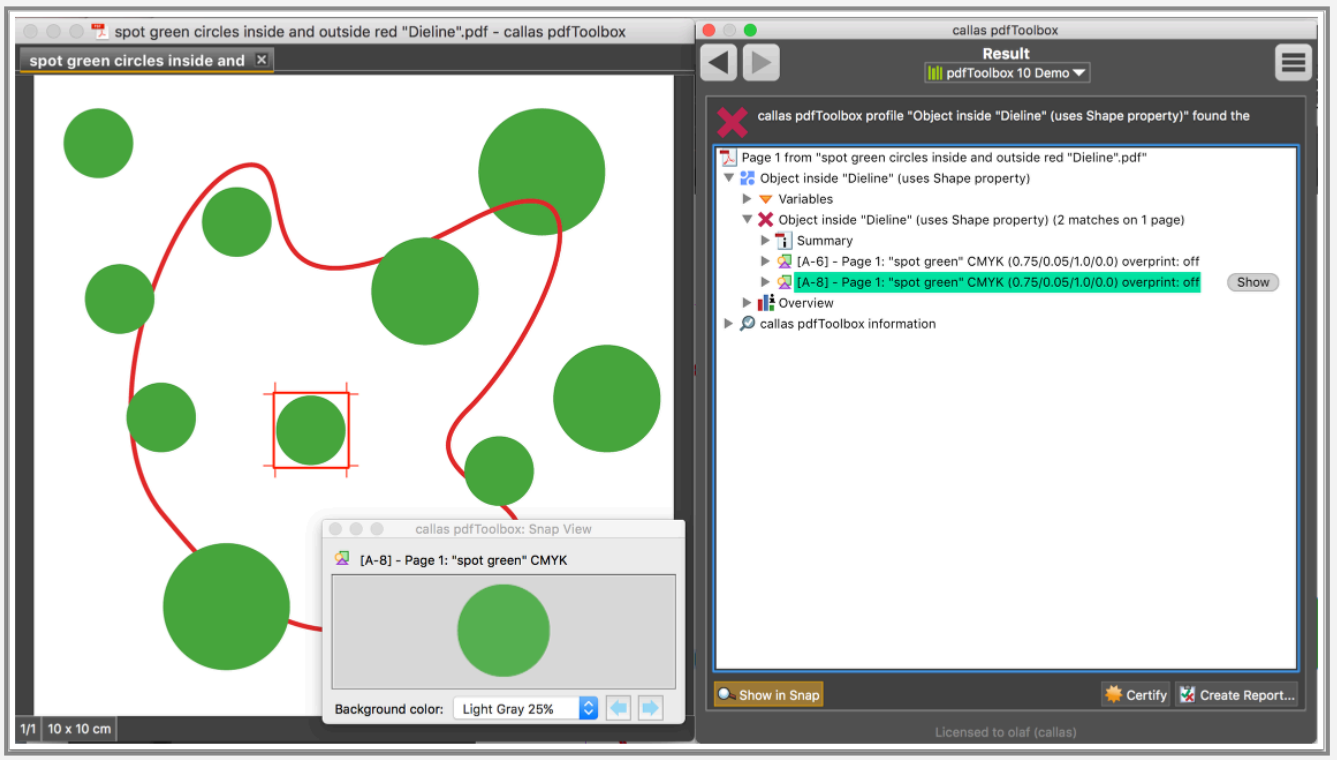


As the ***Shape defined by "Cutline"*** check is using a variable, it is possible to decide at runtime whether to look for a "Cut-line" colored line or a line using a different color.



After execution of the ***Object inside "Dieline" (uses Shape property).kfpx*** it turns out there are two objects that are considered to be inside the shape defined by the "Cutline", as can be seen in the two screenshots below:

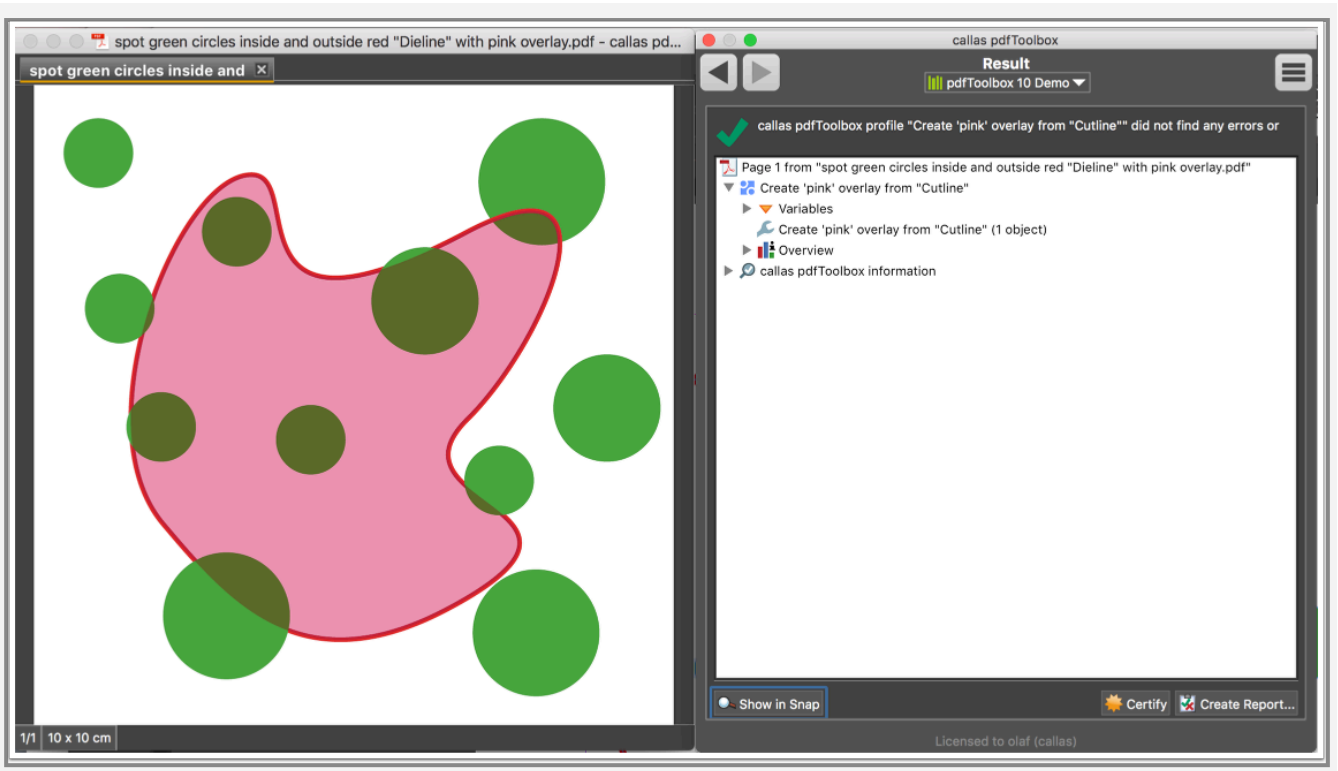




In order to find out whether the shape defined in the ***Shape defined by "Cutline"*** check is actually what one expects it to be, one can use the same shape definition (unfortunately it

has to be re-entered in the Shape fixup configuration) and create a filled area based on the shape definition.

As can be seen below the filled 'pink' area covers the "Cut-line" shape as expected:



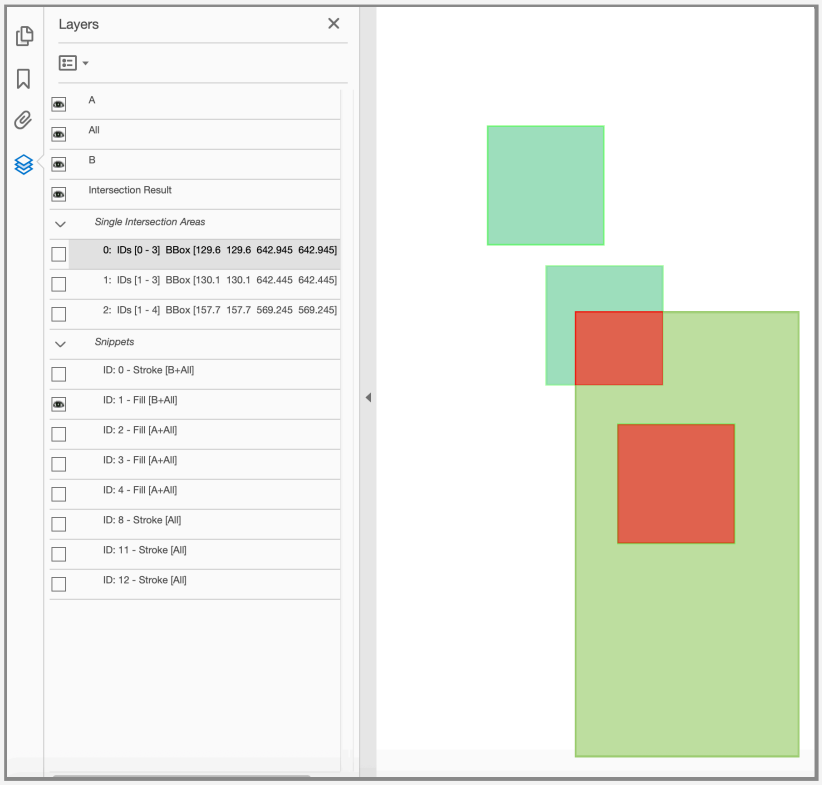
24.25 How to debug Sifter Checks and see their results

Sifter shapes can be evaluated using Profile Execution and Test mode.

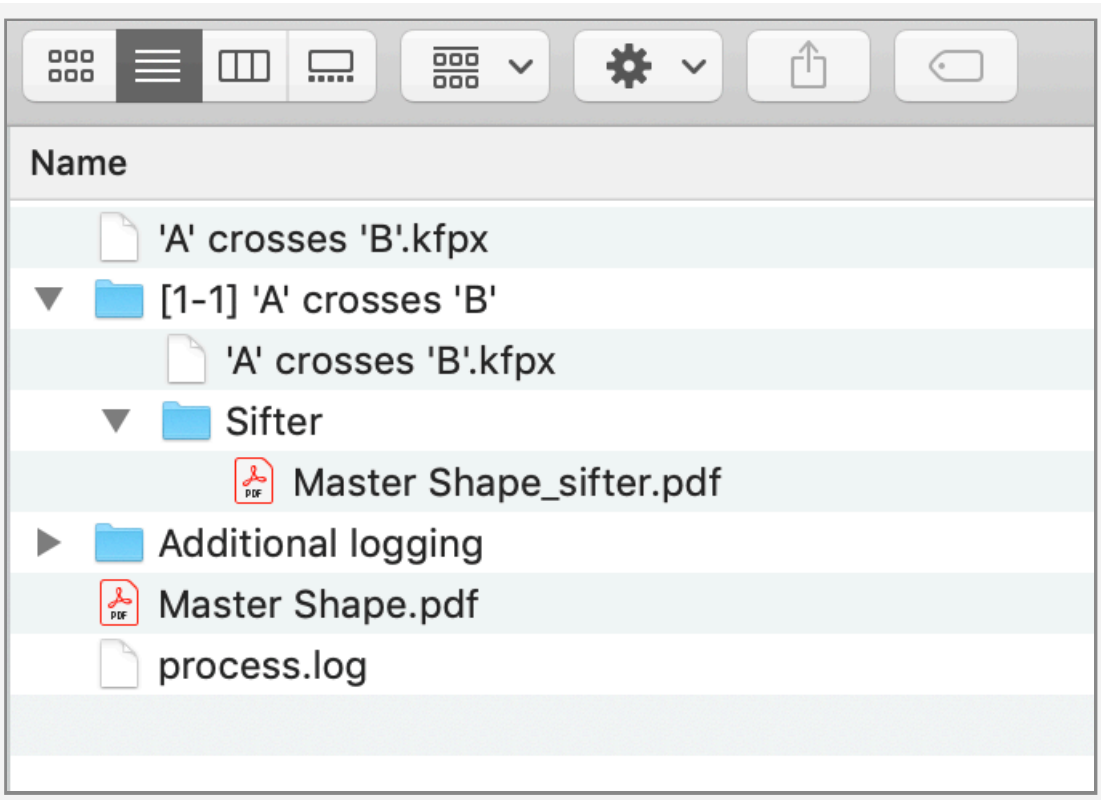
Sifter shapes with Log Profile Execution

Log Profile Execution creates a sifter input or snippet report that contains layers for all snippet categories that are playing a role during a sifter check. The report is a PDF file containing layers for the following categories:

- **A:** All snippets from collection A
- **B:** All snippets from collection B
- **All:** All snippets from collection A & B
- **Intersection Results:** All intersections between A and B
- **Single Intersection Areas:** One layer for each individual intersection
  - Layer name encodes snippet IDs and bounding box of intersection
    - **0: IDs [-1 - 2] BBox [157.7 157.7 569.245 569.245]**
- **Snippets:** One layer for each snippet from collection A or B
  - Layer name encodes snippet ID, fill or stroke and collections, e.g.:
    - **ID: 0 - Fill [A+All]**



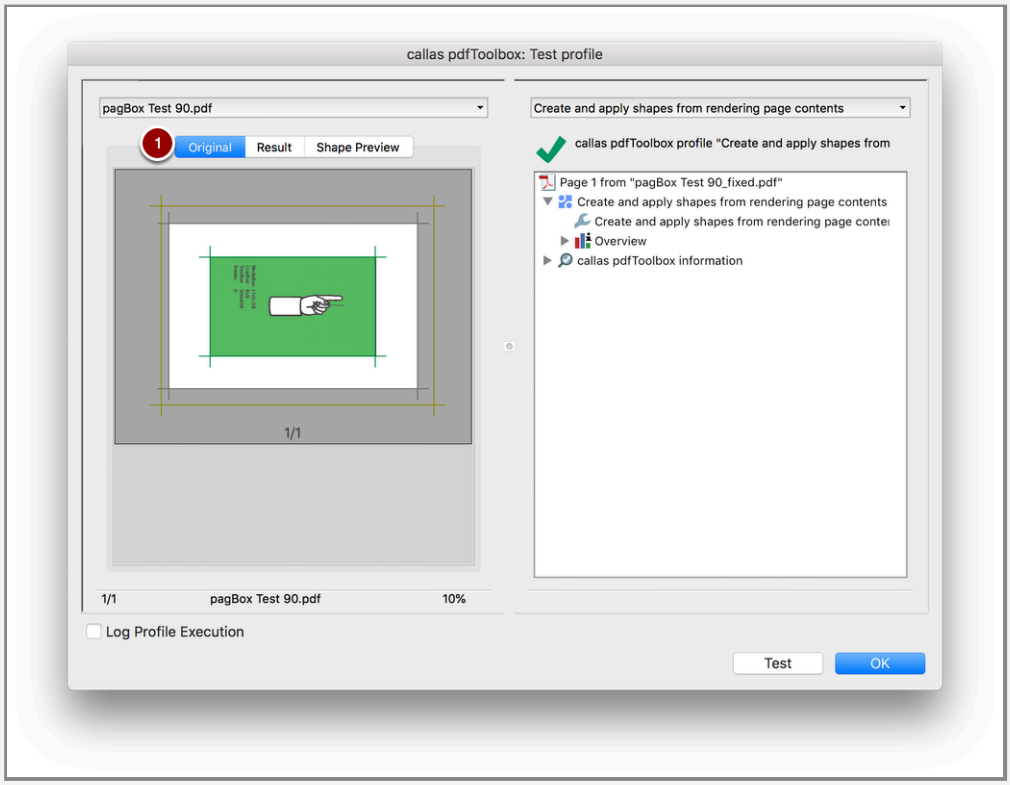
When "Log profile execution" is active, such a report is written for each sifter check to the step folder in a sub folder (as shown in the sample screenshot below) with the name "Sifter" during profile execution.

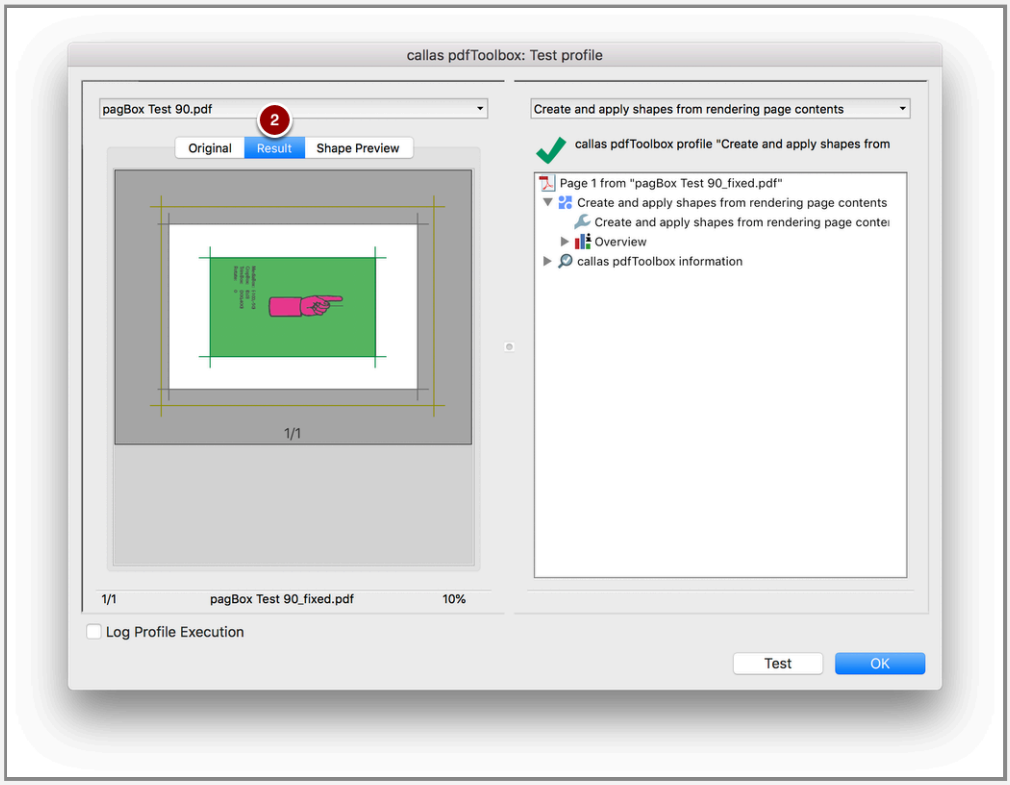


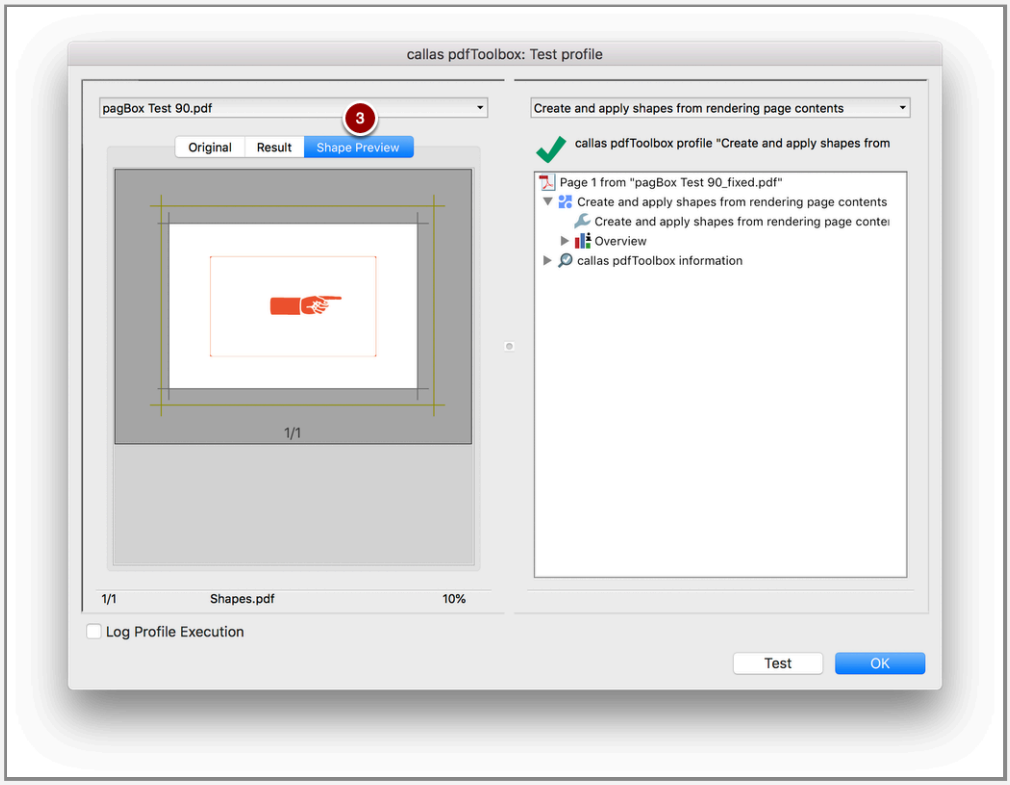
Sifter shapes with Test mode

In Test mode, if a profile creates at least one shape (e.g. by running a "Create and apply shape" Fixup or a "Shape definition" check), an extra tab "Shape preview" is created on the left hand panel displaying a shape preview created by filling the shape with red color. The screenshots below show 3 different tabs of the preview of the Test Mode:

- Original (1): First tab in the left hand panel during Test Mode showing the original PDF
- Result (2): Second tab in the left hand panel during Test Mode showing the resultant PDF after applying "Create and apply shape" Profile
- Shape preview (3): Third tab in the left hand panel during Test Mode showing the generated shape filled by red color


callas pdfToolbox: Test profile
pagBox Test 90.pdf
Create and apply shapes from rendering page contents
1
Original
Result
Shape Preview
callas pdfToolbox profile "Create and apply shapes from
Page 1 from "pagBox Test 90_fixed.pdf"
Create and apply shapes from rendering page contents
Create and apply shapes from rendering page conte
Overview
callas pdfToolbox information
1/1
1/1
pagBox Test 90.pdf
10%
Log Profile Execution
Test
OK


callas pdfToolbox: Test profile
pagBox Test 90.pdf
Create and apply shapes from rendering page contents
2
Original
Result
Shape Preview
callas pdfToolbox profile "Create and apply shapes from
Page 1 from "pagBox Test 90_fixed.pdf"
Create and apply shapes from rendering page contents
Create and apply shapes from rendering page conte
Overview
callas pdfToolbox information
1/1
1/1
pagBox Test 90_fixed.pdf
10%
Log Profile Execution
Test
OK


callas pdfToolbox: Test profile
pagBox Test 90.pdf
3
Original
Result
Shape Preview
1/1
1/1
Shapes.pdf
10%
Create and apply shapes from rendering page contents
callas pdfToolbox profile "Create and apply shapes from
Page 1 from "pagBox Test 90_fixed.pdf"
Create and apply shapes from rendering page contents
Create and apply shapes from rendering page conte
Overview
callas pdfToolbox information
Log Profile Execution
Test
OK

25. Processing Steps

25.1 Design and more

PDF documents typically are WYSIWYG: what you see is what you get. The PDF document contains those elements that need to be printed and it thus contains "design" elements in a complete and accurate way. However, in many cases, additional information needs to be transmitted from the designer to the printer as well, and that additional information often is added to the PDF document.



Typical non-design content

It is a bit dangerous to call this additional content "non-design". It is strictly true, because such content will not be reproduced faithfully as is the rest of the design, but it can still influence the final appearance of the PDF document after print. Some examples:

- When designing non-rectangular jobs (such as labels and packaging), a cut line needs to be defined. This cut line indicates how the design is going to be cut to get the final printed piece.
- When designing something that that will be printed on a transparent material, it is often necessary to add an additional white ink layer underneath the rest of the design.

This additional white layer is usually included in the PDF document, but it's usually included with a 'fake' (non-white) color appearance so it's visible in the PDF document.
- When creating complex jobs, it is often necessary to add all kinds of additional information to the job, such as job identity, printing and cutting marks, color patches, dimensions and so on. While this information is often critical to the job, it is of course not to be printed.
- Some jobs require special coatings or finishing processes; parts of a job might need to be varnished, may have silver or gold foils applied to them, or require embossing. These special processes are included in the PDF document, again with 'fake' colors to show where they will affect the design.

Current practices

In workflows where such information is required, these special elements are typically indicated by using spot colors. Elements using a spot color with the name "White" refer to the additional white ink layer. Elements using the name "Varnish" indicate areas of the file to be varnished. Spot colors such as "Legend" or "Registration" or "Marks" may be used for elements that are not print content, but job identification or assisitive printing or cutting marks.

Problems with this approach

This approach with using spot colors raises a number of problems:

- The kind of jobs we are talking about here also typically use a list of spot colors for design elements (elements that do need to be printed in specific brand colors for example). Using spot colors for both design and non-design elements can lead to confusion.
- There is no standardisation on the spot color names used. The spot color used to indicate a die-cut line, may be called "Cut" or "Cutter" or "Die" or "Die-cut" or a variety of other names. On top of that, the design community today is global; a French designer will be likely to use a French name while a Finnish designer will use his own

language. This makes it very hard to build any kind of automation for these files as key information can be encoded in a variety of different ways.

25.2 Using metadata for standardisation

Because of the challenges described in the previous article, work of the Ghent Workgroup lead to the creation of a new ISO standard (ISO 19593) called Processing Steps. In full, the standard is called: "Use of PDF to associate processing steps and content data". Content data here obviously refers to the design elements itself, what will be printed. Processing steps refers to this additional "non-design" information stored in the PDF document.

So how does this standard work?

Use of layers

The PDF standard has a built-in feature called "Optional Content Groups" (OCDs). This is commonly referred to as "layers" though it's important to realise that there are important differences between layers such as you might know them from Adobe Photoshop or Adobe Illustrator and optional content groups. Layers in design application typically reflect stacking order: the objects in the front layer are "on top" of objects in all other layers. This is not the case for optional content groups; PDF documents can contain an optional content group that contains all images in the document, regardless of their stacking order. And moving an element from one optional content group to another, doesn't change it's stacking nor the visual appearance of the document.

These optional content groups are used to gather all elements belonging to a processing step. All vector elements that form the die cut line for example, are placed in an optional content group. Such optional content groups have a name that can be used to easily identify them.

Attaching metadata to layers

Of course using optional content group names to identify them, would bring us right back to the problem of standardisation; everyone would use their own version of a name... To solve this, the processing steps standard uses metadata at-

tached to the layer for the actual idenfication. Each layer has two pieces of identifying metadata attached to it:

- Group
  Identifies what kind of processing step this is. Possible groups are "Structural", "Dimensions", "Braille", "Legend", "White", "Varnish" and "Positions".
- Type
  Identifies the type of processing step in that particular group. In the group "Structural", possible types include "Cutting", "Creasing", "Gluing" and so on.

Using spot colors in layers

Using layers and metadata associated to layers, solves the standardisation problem for processing steps information. However, the elements that are in such a layer still need to have a color, and it makes the most sense to continue to use spot colors for this.

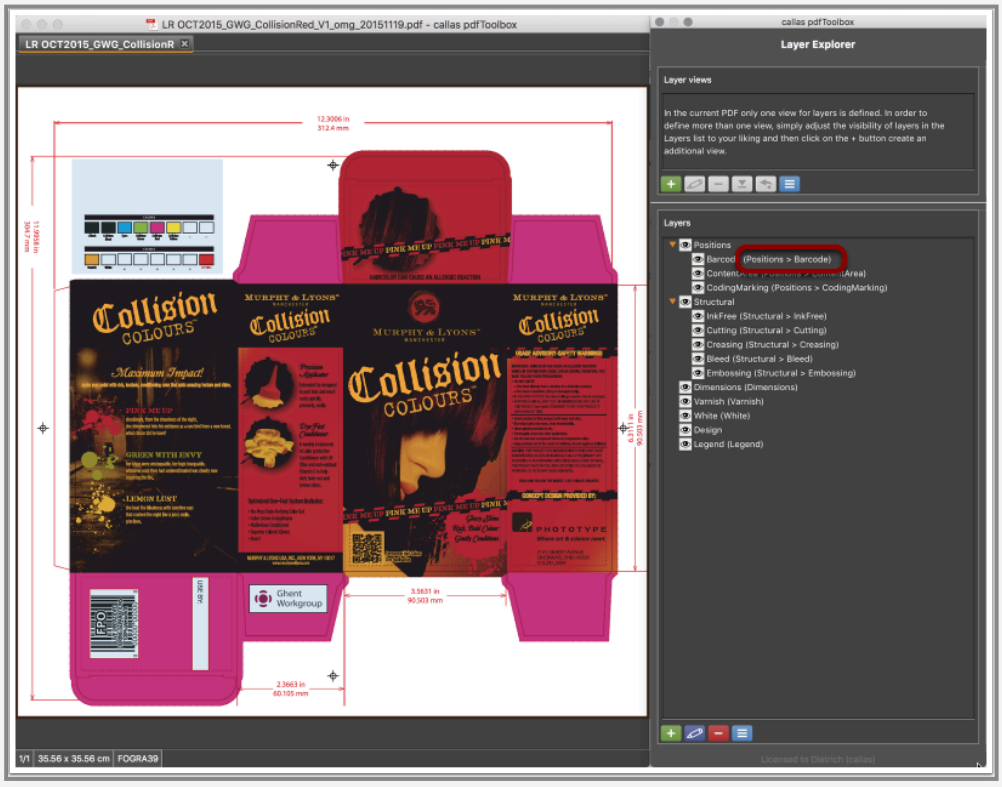
Because of the layers though, these spot colors can be named whatever the designer wants them to be named. As long as the proper processing steps metadata is used, they can be identified regardless.

25.3 Working with processing steps metadata for a layer

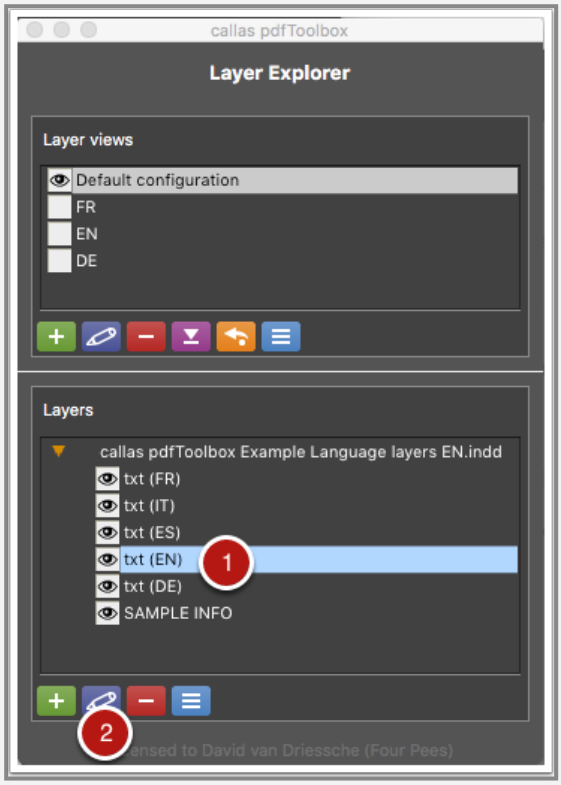
Layers can have regular metadata and processing steps metadata attached to them.

Processing Steps metadata can be added, viewed, edited or deleted using pdfToolbox's Layer Explorer. The Layer Explorer opens via Tools -> Layer Explorer.

When opened for a PDF with Processing Steps metadata it might look like this:

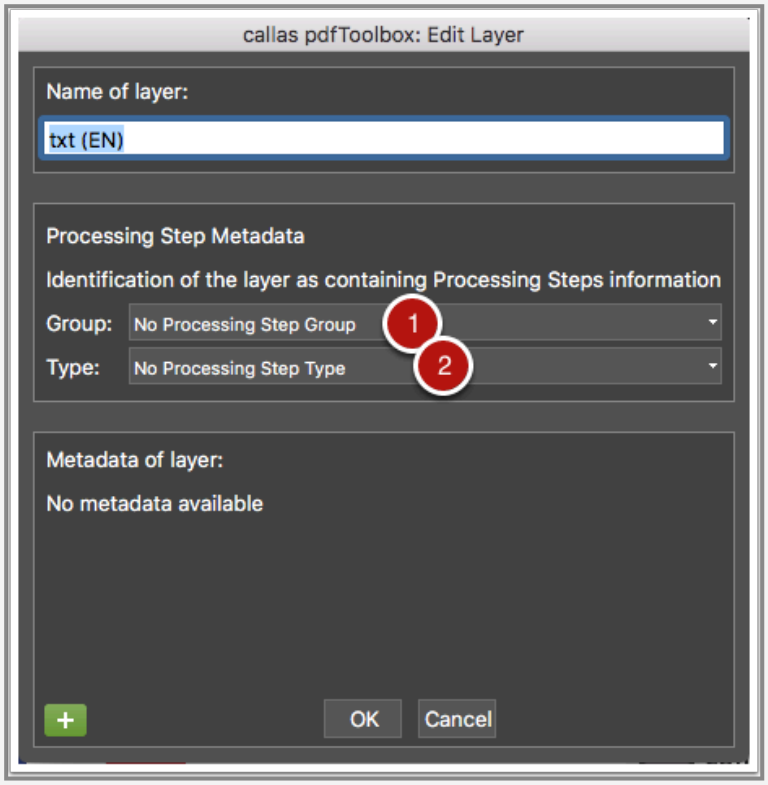


Accessing layer metadata



1. Click the layer you want to see the metadata of.
2. Click the "edit" button to "Edit layer" dialog window.

Viewing processing steps metadata



The "Edit layer" dialog window contains three sections with information about the layer:

1. The name of the layer
2. The processing steps information for the layer
3. Additional metadata associated with the layer

Look at the middle section to work with the processing steps information. This section lists:

1. The processing steps group associated with this layer, or "No Processing Steps Group" if no information is available for this layer.
2. The processing steps type associated with this layer, or "No Processing Steps Type" if no information is available for this layer.

Changing processing steps information

You can change the processing steps group or type by using the pull down menus in the middle section of the "Edit layer" dialog window.

Deleting processing steps information

Processing steps information can be removed by using the pull down menus in the middle section of the "Edit layer" dialog window. Simply select the top value in both menus ("No Processing Steps Group / Type").

25.4 Predefined "Processing Steps" library

With **pdfToolbox 16**, the predefined **"Processing Steps"** library is available. This library includes predefined Profiles, Process Plans, Checks, and Fixups - all related to Processing Steps. This article provides a brief overview of the components in this library:



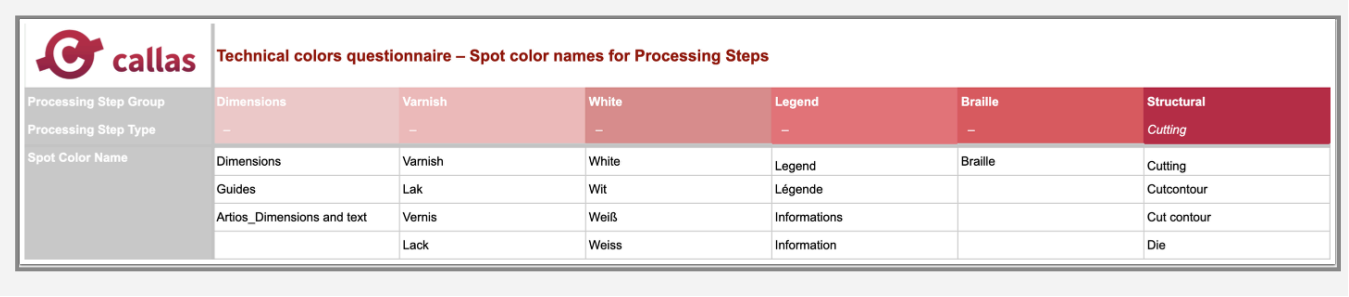
1. *"Add Processing Steps layers"*
   Puts objects using technical colors on Processing Steps layers. Is part of the predefined Process Plan "Prepare PDF using technical colors with Processing Steps layers" (3).
2. *"List Processing Steps metadata information"*
   Provides information about what Processing Steps metadata is present in a PDF file, whether any layers are empty etc.
3. *"Prepare PDF using technical colors with Processing Steps layers"*
   Remove existing layers and put objects using technical

colors on Processing Steps layers and print content on a layer "Design". See below for more details.

4. *"Verify compliance with Processing Steps (ISO 19593-1) for packaging and label"*
   An implementation of all requirements of the ISO standard. It allows to check whether the PDF complies with this standard.

Prepare PDF using technical colors with Processing Steps layers

This Process Plan automates the process of adding Processing Step layers to a PDF file. To automate this process, a collection of spot color names commonly used in practice for all Processing Step types was required. To achieve this, we created the "Technical colors questionnaire" where users can enter their commonly used spot color names for these objects:

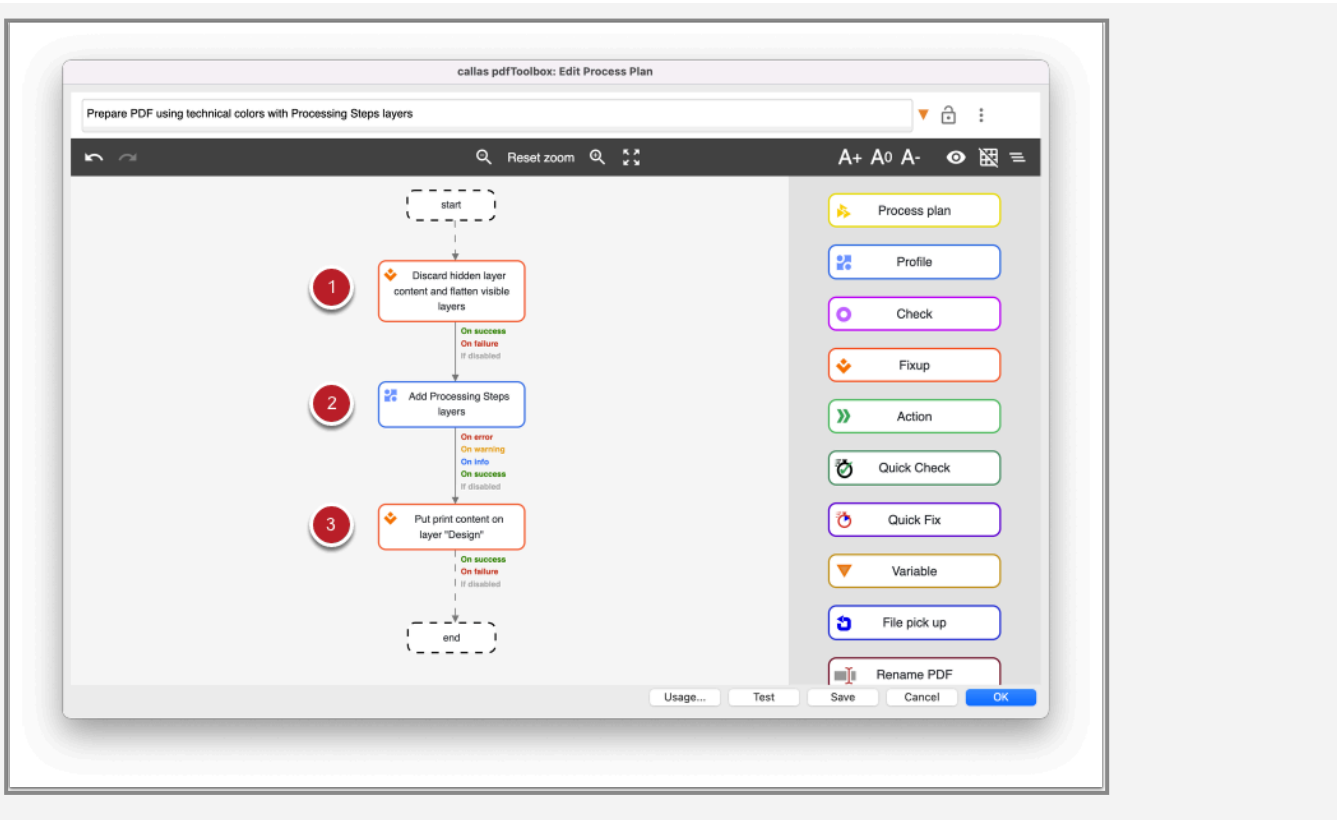


callas

Technical colors questionnaire – Spot color names for Processing Steps

Processing Step Group / Processing Step Type	Dimensions	Varnish	White	Legend	Braille	Structural *Cutting*
Spot Color Name	Dimensions	Varnish	White	Legend	Braille	Cutting
	Guides	Lak	Wit	Légende		Cutcontour
	Artios_Dimensions and text	Vernis	Weiß	Informations		Cut contour
		Lack	Weiss	Information		Die

The Process Plan is based on this collection and will help you automatically add Processing Step layers to your PDF files:



1. First, all existing layers in the PDF will be removed to create the proper Processing Step layers.
2. The Profile places all objects identified by a list of spot color names (based on the technical colors spreadsheet) onto Processing Steps layers. For example:
   - Objects using the spot colors "Cutting," "Dieline," "Stanze", etc., are placed on a layer called "Cutting" with the Processing Steps metadata "Structural:Cutting".
   - Objects using the spot colors "White," "Weiss," "Wit", etc., are placed on a layer called "White" with the Processing Steps metadata "White".
3. All print content (objects not assigned to a layer) is placed on a new layer called "Design".

Adjusting the Process Plan to your needs

As mentioned, the spot color names defined in the Process Plan are based on our questionnaire and cover the most commonly used spot color names. However, if you use specific names for Processing Step objects (such as varnish layers,

cutlines, barcode areas, etc.), you can easily adjust the predefined spot color name list in the "Add Processing Steps Layers" Profile. To do so:

1. Unlock both the Process Plan and the Profile to enable changes.
2. Search for a specific Processing Step in the list of Fixups where you want to add one or more spot color names, and click the Pen icon to edit it.
3. Open the Check listed under "Objects identified by a check:" and add the spot color name to the list.

The technical color spreadsheet is still accessible, and you can continue adding spot color names. To access the spreadsheet, visit the "**technicalcolors**" subdomain on the callas software website.

For pdfToolbox 15 and earlier versions

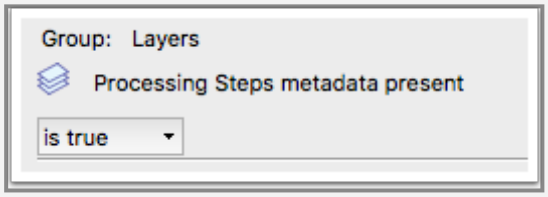
Until pdfToolbox 15, the two profiles "List Processing Steps metadata information" and "Verify compliance with Processing Steps (ISO 1959-3-1) for packaging and label" can be found in the **"PDF Standards"** library.


callas pdfToolbox
Profiles
PDF Standards
Default
Find
Imported profiles
PDF
PDF/A
PDF/E
PDF/UA
PDF/VT
PDF/X
PDF/X and PDF/A
Process plans
Processing Steps
List Processing Steps metadata information
Verify compliance with Processing Steps (ISO 19593-1) for packaging and label
Further Options
Analyze
Analyze and fix
Licensed to Sarah Nitschmann (callas software)

25.5 Checking processing steps information

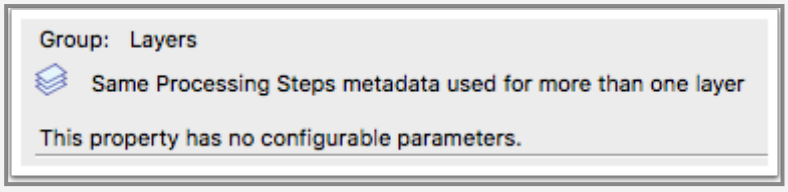
Having correct processing steps information can be important for the functioning of automatic workflows. As such, pdfToolbox implements a number of specific processing steps Checks.

Checking for presence



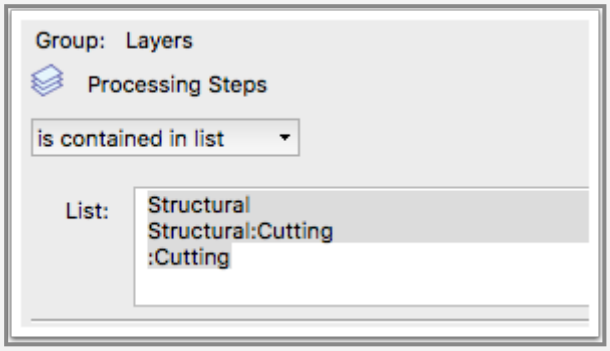
This condition returns true if processing information is present in the document, false if it is not.

Checking for conflicts



This condition can be used to find out whether the same processing steps information is used for more than one layer. If two layers are marked "Structural" > "Cutting" for example, it makes it harder to figure out which of those two is the actual die-line, and it might indicate other problems with the file or the workflow.

Identifying layers with specific processing types



This condition is useful to identify specific processing steps layers in a document. Multiple items can be searched for by listing each item on a new line (as in the example above). Each line must have one of three possible formats:

- **<group name>**
  The line contains just the name of a processing steps group, no type is mentioned. This will create a hit for any processing steps layer that has this specific group (regardless of type).
- **<group name>:<type name>**
  The line contains the name of a processing steps group, followed by a colon (':'), followed by the name of a processing steps type. This creates a hit for any layer that has the specified group *and* type.
- **:<type name>**
  The line contains a colon (':'), followed by the name of a processing steps type. This creates a hit for any layer that has the specified type (regardless of group).

Identifying custom processing steps information



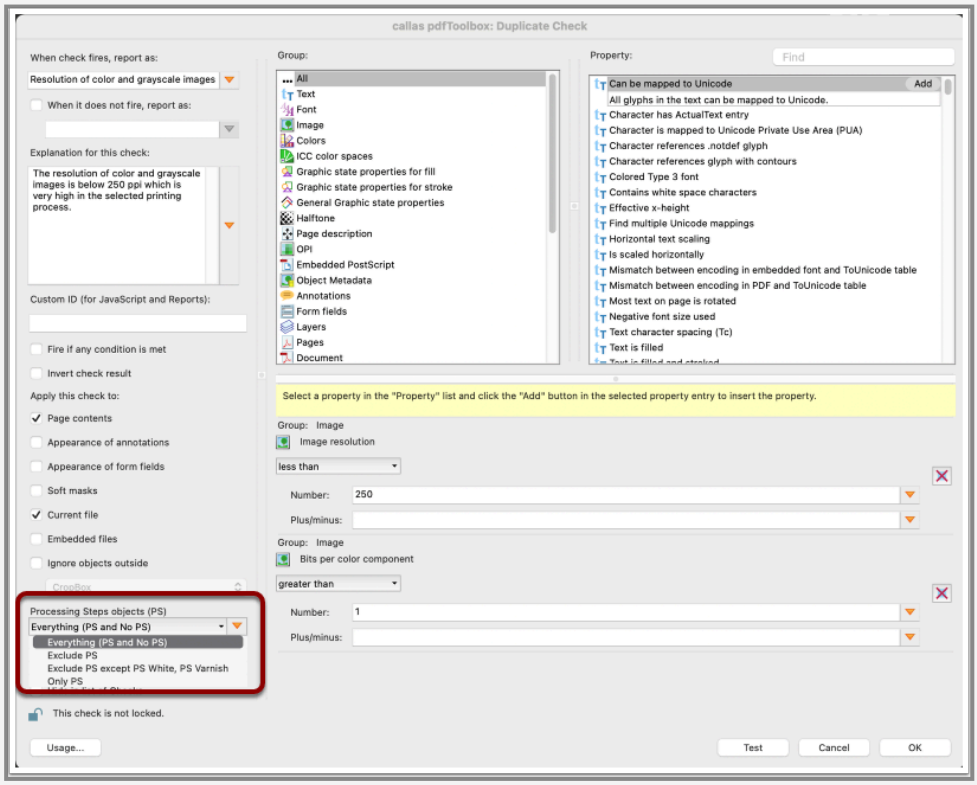
The processing steps standard defines a list of predefined groups and types, but it also allows custom values to be used when none of the predefined values can be used. This condition finds layers where such custom values are used.

Specifying whether Processing Steps information is checked or not

When PDF files with Processing Steps information are processed it is not only necessary to check for the Processing Steps information itself, it should also be possible to specify for each Check whether it is applied to information on a Processing Steps layer as well or not. E.g. a Check that creates a hit for images where the resolution is too low may not be as useful for objects on Processing Steps "Legend" as for regular page content.

It is possible to add a condition to every Check that rules out objects on a Processing Steps layer.

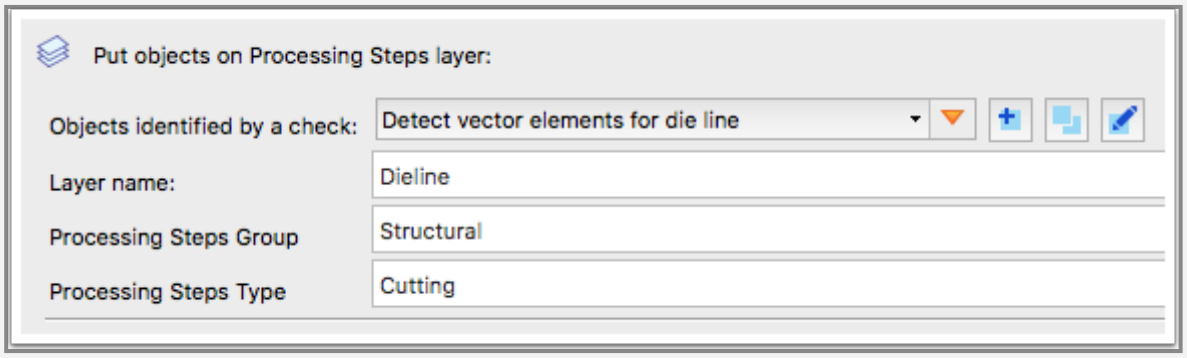
In pdfToolbox 13 that becomes more convenient, there is new pop up in the sidebar that allows you to specify whether or not Processing Steps information should be taken into account or not. Since the Processing Steps groups White and Varnish are "almost" regular print content it is possible to include only this information.


callas pdfToolbox: Duplicate Check
When check fires, report as:
Resolution of color and grayscale images
When it does not fire, report as:
Explanation for this check:
The resolution of color and grayscale images is below 250 ppi which is very high in the selected printing process.
Custom ID (for JavaScript and Reports):
Fire if any condition is met
Invert check result
Apply this check to:
Page contents
Appearance of annotations
Appearance of form fields
Soft masks
Current file
Embedded files
Ignore objects outside
Processing Steps objects (PS)
Everything (PS and No PS)
Exclude PS
Exclude PS except PS White, PS Varnish
Only PS
This check is not locked.
Usage...
Group:
All
Text
Font
Image
Colors
ICC color spaces
Graphic state properties for fill
Graphic state properties for stroke
General Graphic state properties
Halftone
Page description
OPI
Embedded PostScript
Object Metadata
Annotations
Form fields
Layers
Pages
Document
Property:
Find
Can be mapped to Unicode
Add
All glyphs in the text can be mapped to Unicode.
Character has ActualText entry
Character is mapped to Unicode Private Use Area (PUA)
Character references .notdef glyph
Character references glyph with contours
Colored Type 3 font
Contains white space characters
Effective x-height
Find multiple Unicode mappings
Horizontal text scaling
Is scaled horizontally
Mismatch between encoding in embedded font and ToUnicode table
Mismatch between encoding in PDF and ToUnicode table
Most text on page is rotated
Negative font size used
Text character spacing (Tc)
Text is filled
Select a property in the "Property" list and click the "Add" button in the selected property entry to insert the property.
Group: Image
Image resolution
less than
Number: 250
Plus/minus:
Group: Image
Bits per color component
greater than
Number: 1
Plus/minus:
Test
Cancel
OK

25.6 Fixing processing steps data

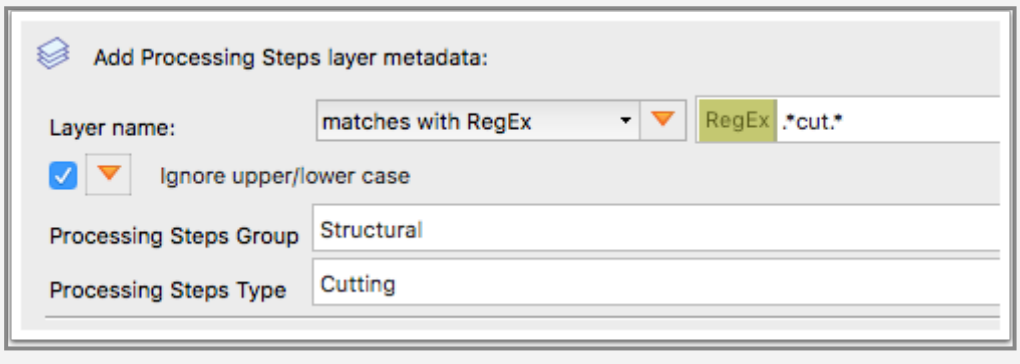
pdfToolbox can fix a number of common problems with processing steps information and can be used to convert legacy files using spot color identification to processing steps.

Putting objects on a specific layer



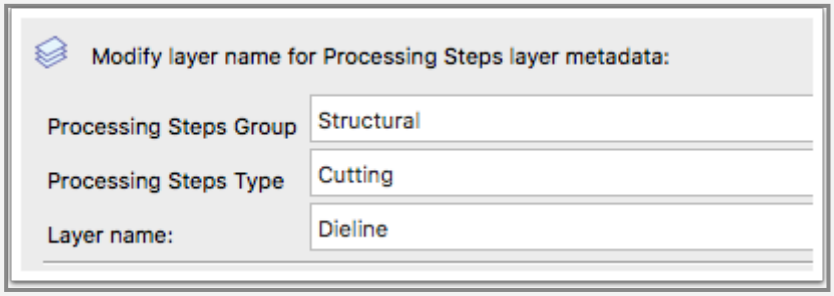
This fixup identifies objects with a preflight check; those objects are then put on layer identified by its processing steps group and type.

Adding processing steps information to a layer



This fixup identifies a layer by name, and then adds specific processing steps information to it.

Rename layer identified by processing steps information



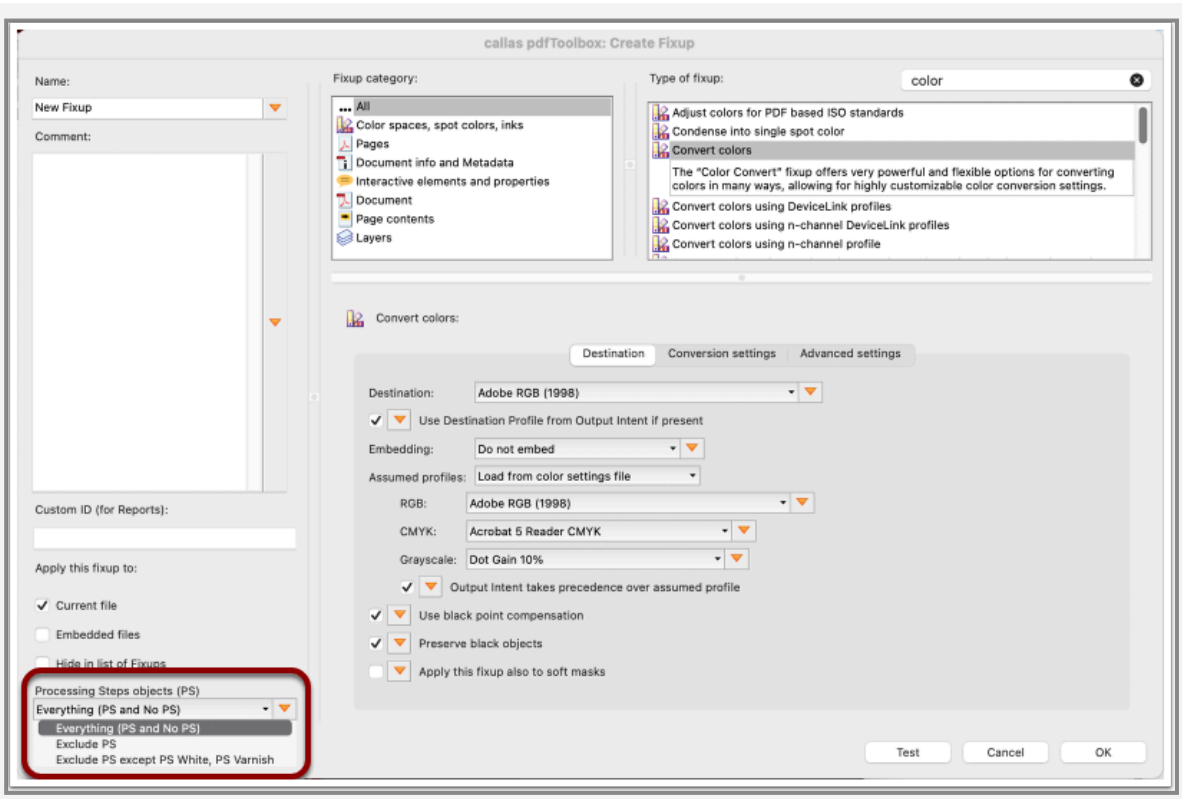
This fixup identifies a layer using the specified processing steps information and changes its name.

Specifying whether Processing Steps information is converted or not

When PDF files with Processing Steps information are processed it is not only necessary or even desirable to modify the Processing Steps information. E.g. if you convert all spot color to CMYK you will want to exclude Processing Steps objects.

It is possible to add a filter in the "Apply to" pop up or a similar means to Fixups that would rule out objects on a Processing Steps layer.

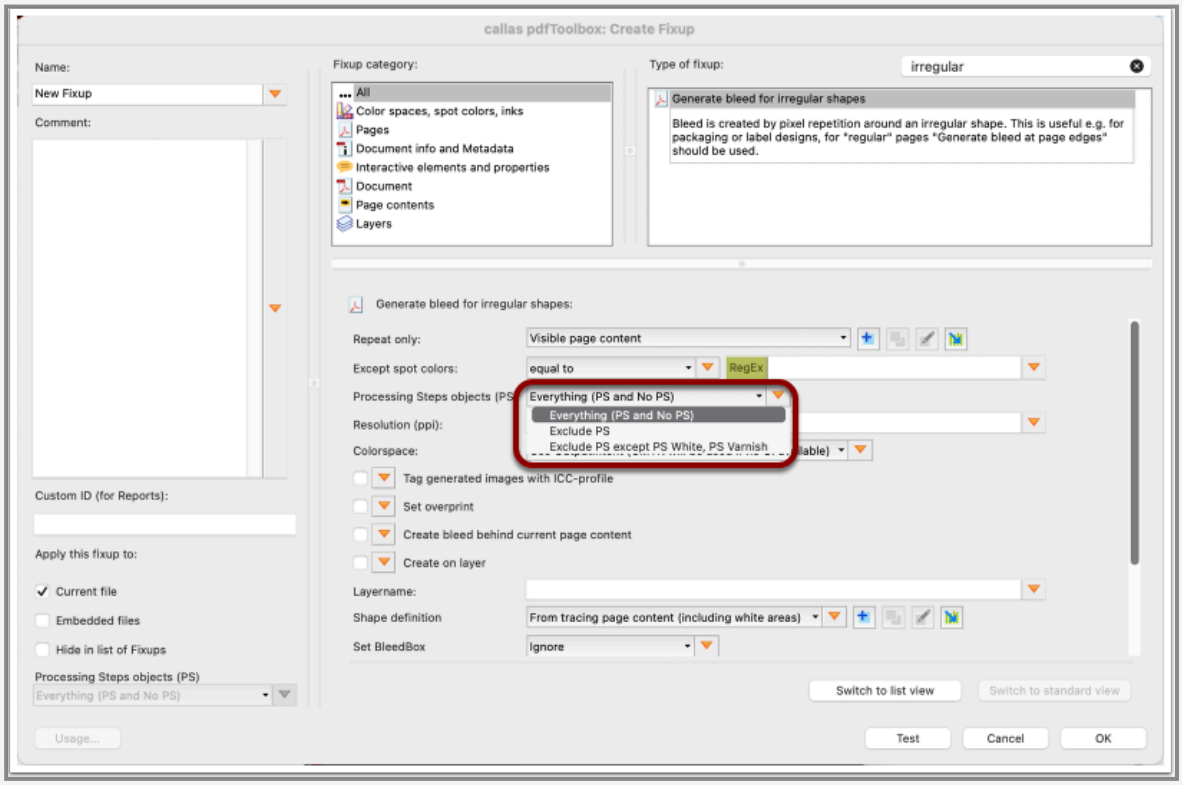
Since pdfToolbox 14 that becomes more convenient for the most relevant Fixups, because there is a new pop up in the sidebar that allows you to specify whether or not Processing Steps information should be taken into account or not. Since the Processing Steps groups White and Varnish are "almost" regular print content it is possible to still include only these objects.



This pop up is not active in all Fixups, but only in those where excluding Processing Steps objects actually makes sense.
These Fixups are:

- Generate bleed at page edges
- Convert colors
- Map spot and process colors
- Set Overprint and Knockout
- Crop to visible based on rendered page
- Remove objects

In "Generate bleed for irregular shapes" there is a similar mechanism in the Fixup settings:


callas pdfToolbox: Create Fixup
Name:
New Fixup
Comment:
Custom ID (for Reports):
Apply this fixup to:
Current file
Embedded files
Hide in list of Fixups
Processing Steps objects (PS)
Everything (PS and No PS)
Usage...
Fixup category:
All
Color spaces, spot colors, inks
Pages
Document info and Metadata
Interactive elements and properties
Document
Page contents
Layers
Type of fixup:
irregular
Generate bleed for irregular shapes
Bleed is created by pixel repetition around an irregular shape. This is useful e.g. for packaging or label designs, for "regular" pages "Generate bleed at page edges" should be used.
Generate bleed for irregular shapes:
Repeat only:
Visible page content
Except spot colors:
equal to
RegEx
Processing Steps objects (PS
Everything (PS and No PS)
Exclude PS
Exclude PS except PS White, PS Varnish
Resolution (ppi):
Colorspace:
Tag generated images with ICC-profile
Set overprint
Create bleed behind current page content
Create on layer
Layername:
Shape definition
From tracing page content (including white areas)
Set BleedBox
Ignore
Switch to list view
Switch to standard view
Test
Cancel
OK

25.7 Overview of predefined groups and types for processing steps

Below you will find a list of layer metadata as defined in the "Processing Steps Specification" as published by the Ghent Workgroup (see details on gwg.org). The list below may be subject to further change in these discussions, so please be aware of its preliminary state.

Groups

Structural	Group containing a set of processing step objects that describe how the printed product will be processed in finishing.
Dimensions	Group containing a set of processing step objects (arrows, numbers and units) indicating physical sizes of items in the design.
Braille	Group containing a set of processing step objects representing braille text.
Legend	Group containing a set of processing step objects representing product related administrative and technical information.
Positions	Group containing a set of processing step objects for positioning information for various graphical and non graphical elements.
White	Group containing a set of processing step objects describing a white backing applied on transparent foils, metallic surfaces, etc..
Varnish	Group containing processing step objects describing printed varnish.

Types in Group "Structural"

Cutting	Processing step objects indicating where the printed artwork will be cut from the printed

	sheet e.g. with a guillotine cutter or die cutting device.
PartialCutting	Processing step objects indicating where the substrate will be cut partially i.e. not entirely through the material.
ReversePartialCutting	Processing step objects indicating where the substrate will be cut partially i.e. not entirely through the material on the back side of the substrate.
Creasing	Processing step objects indicating where the substrate will be creased to guide subsequent folding.
ReverseCreasing	Processing step objects indicating where the substrate will be creased on the back side of the substrate.
CuttingCreasing	Processing step objects indicating where the substrate will undergo alternating cutting and creasing.
ReverseCuttingCreasing	Processing step objects indicating where the substrate will undergo alternating cutting and creasing on the back side of the substrate.
PartialCuttingCreasing	Processing step objects indicating where the substrate will undergo alternating partial cutting and creasing.
ReversePartialCuttingCreasing	Processing step objects indicating where the substrate will undergo alternating partial cutting and creasing on the back side of the substrate.
Drilling	Processing step objects indicating locations where the substrate will be drilled and the intended size of the resulting hole.
Gluing	Processing step objects enclosing an area where glue will be applied.
FoilStamping	Processing step objects enclosing an area where foil will be applied through hot foil stamping.

ColdFoilStamping	Processing step objects enclosing an area where foil will be applied through cold foil stamping (i.e. using glue).
Embossing	Processing step objects enclosing an area where embossing will be applied.
Debossing	Processing step objects enclosing an area where debossing will be applied.
Perforating	Processing step objects indicating where the substrate will be perforated.
Bleed	Processing step objects indicating the intended bleed for print.
VarnishFree	Processing step objects enclosing an area where it is not allowed to have varnish.
InkFree	Processing step objects enclosing an area where it is not allowed to have printing ink.
InkVarnishFree	Processing step objects enclosing an area where it is not allowed to have printing ink and where it is not allowed to have varnish.
Folding	Processing step objects indicating where the substrate will be folded without prior creasing.
Punching	Processing steps objects indicating the locations at which the substrate will be punched and the size and shape of the resulting holes.
Stapling	Processing steps objects indicating the locations at which the substrate will be stapled or stitched, and the size of the staples or stitches to be used.

Types in Group "Positions"

Hologram	Processing step objects that indicate the intended position of holograms.
Barcode	Processing step objects that indicate the intended position of barcodes.

ContentArea	Processing step objects that indicate areas where it is allowed to place text and other graphical elements.
CodingMarking	Processing step objects specific to packaging that indicate areas where additional information not contained in the PDF will be printed on the packaging at a post-press stage, such as on filling lines where final products are being packed. Some examples of information that might be marked as CodingMarking include but are not limited to best before dates, lot numbers, production dates, and tracking codes.
Imprinting	Processing step objects that indicate areas where additional information (e.g. variable data) will be printed on pre-printed shells which contain the bulk of the graphical content, as a secondary print stage prior to finishing.

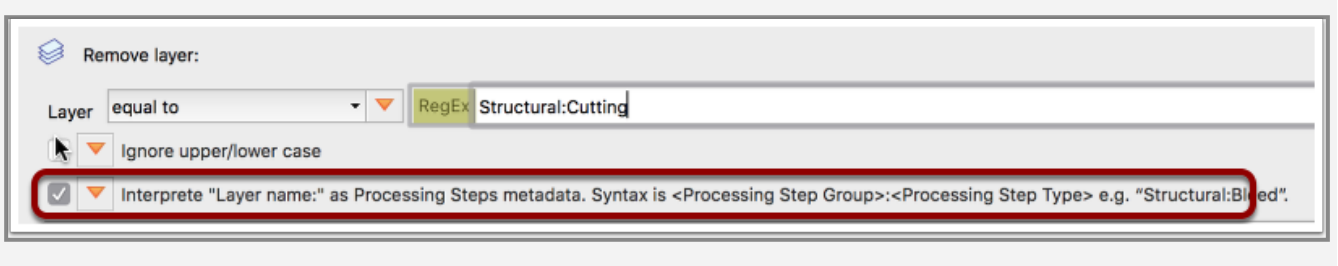
25.8 Configurable Checks and Fixups for Processing Steps

The most important configurable Properties (Checks) related to Processing Steps

- *Document has Processing Steps metadata*
- *Page has Processing Steps metadata*
- *Processing Step layers missing on page*
  This Property can be used with a list of Processing Steps metadata values. It will report a problem when any of the Processing Steps layers is not present on any of the pages in a PDF file
- *Processing Steps metadata uses custom values*
- *Same Processing Steps metadata used for more than one layer*

Configurable Fixups for Processing Steps

Many of the Fixups for layers have a checkbox that allows for interpreting the entry for layer name as related to Processing Steps metadata.



So that you may use all such fixups to adjust or remove layers can be used for layers identified by Processing Steps metadata as well.

Other fixups for Processing Steps are:

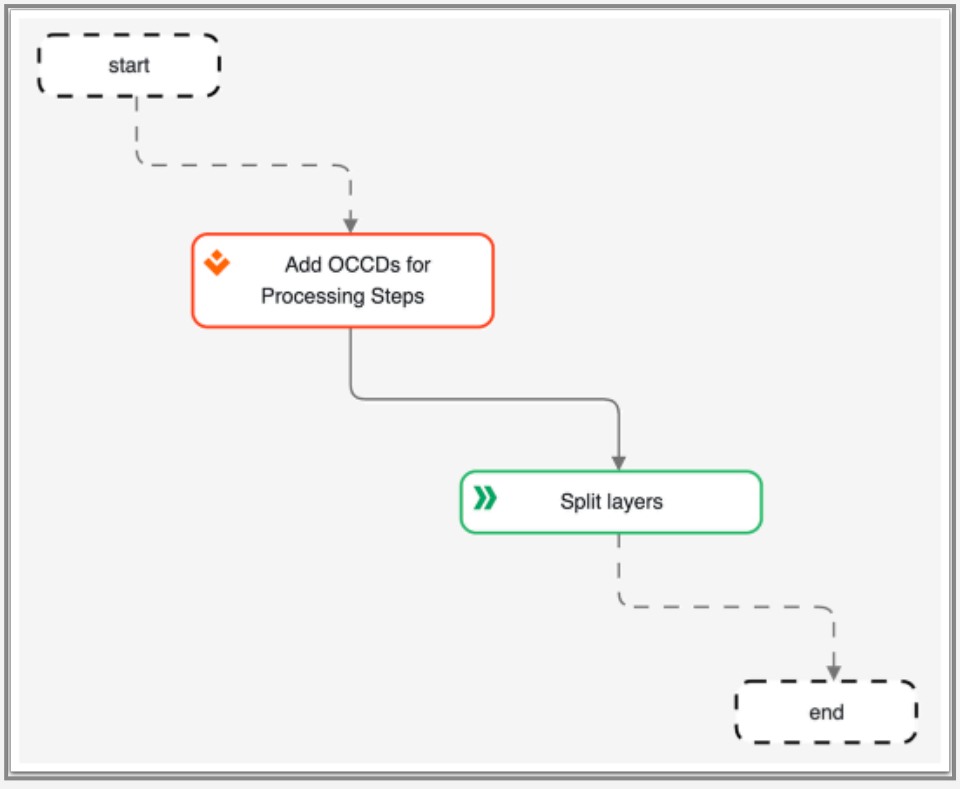
- Add Processing Steps layer metadata
- Put objects on Processing Steps layer
- Modify layer name for Processing Steps layer metadata

25.9 Split PDFs based on Processing Steps

Beginning with pdfToolbox v13, it is possible to split a PDF into single files based on Processing Steps layers. In order to support grouping of layers in the result files, the OCCD mechanism that is available in PDF to group layers is used. The process to split a PDF runs in two steps:

1. Add OCCD information
2. Split the PDF based on the OCCD information

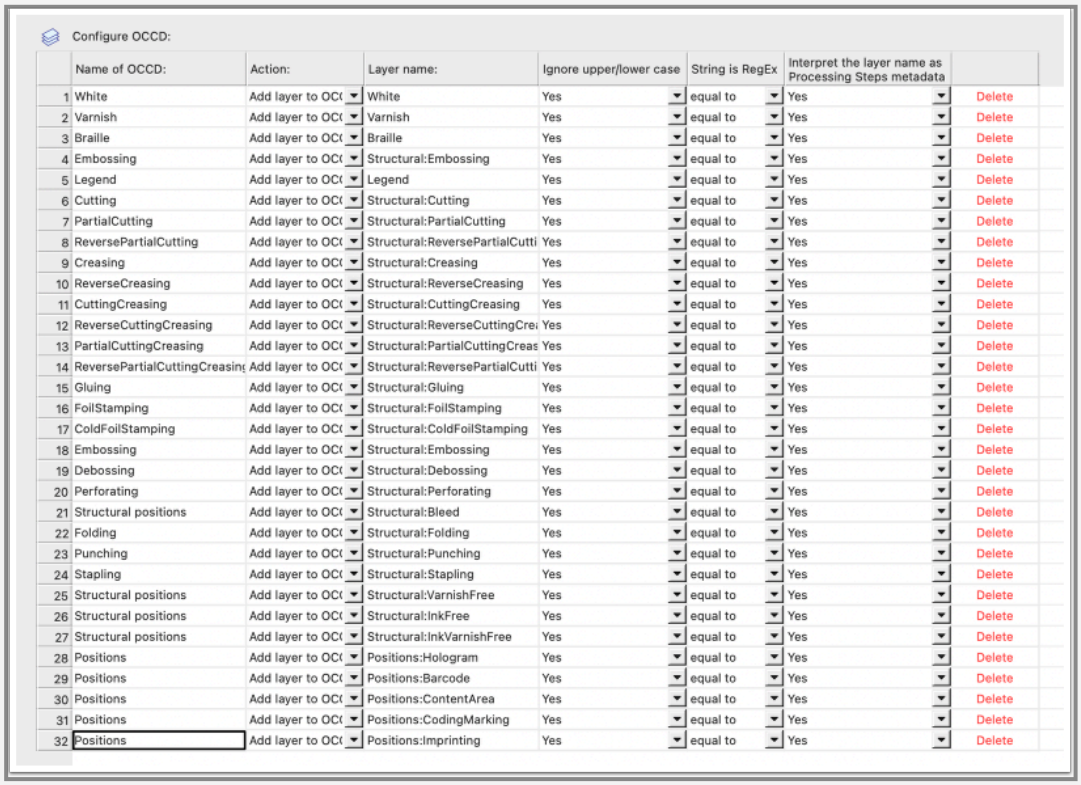
Split_Processing_Steps_layers.kfpx



Add OCCD information

In this step each of the possible standard Processing Steps is assigned to an OCCD set. Most of the Processing Steps types are assigned to a dedicated OCCD set, but some, e.g. the

ones in the group Positions are assigned to a combined OCCD set.



Configure OCCD:

	Name of OCCD:	Action:	Layer name:	Ignore upper/lower case	String is RegEx	Interpret the layer name as Processing Steps metadata	
1	White	Add layer to OCC	White	Yes	equal to	Yes	Delete
2	Varnish	Add layer to OCC	Varnish	Yes	equal to	Yes	Delete
3	Braille	Add layer to OCC	Braille	Yes	equal to	Yes	Delete
4	Embossing	Add layer to OCC	Structural:Embossing	Yes	equal to	Yes	Delete
5	Legend	Add layer to OCC	Legend	Yes	equal to	Yes	Delete
6	Cutting	Add layer to OCC	Structural:Cutting	Yes	equal to	Yes	Delete
7	PartialCutting	Add layer to OCC	Structural:PartialCutting	Yes	equal to	Yes	Delete
8	ReversePartialCutting	Add layer to OCC	Structural:ReversePartialCutti	Yes	equal to	Yes	Delete
9	Creasing	Add layer to OCC	Structural:Creasing	Yes	equal to	Yes	Delete
10	ReverseCreasing	Add layer to OCC	Structural:ReverseCreasing	Yes	equal to	Yes	Delete
11	CuttingCreasing	Add layer to OCC	Structural:CuttingCreasing	Yes	equal to	Yes	Delete
12	ReverseCuttingCreasing	Add layer to OCC	Structural:ReverseCuttingCrea	Yes	equal to	Yes	Delete
13	PartialCuttingCreasing	Add layer to OCC	Structural:PartialCuttingCreas	Yes	equal to	Yes	Delete
14	ReversePartialCuttingCreasing	Add layer to OCC	Structural:ReversePartialCutti	Yes	equal to	Yes	Delete
15	Gluing	Add layer to OCC	Structural:Gluing	Yes	equal to	Yes	Delete
16	FoilStamping	Add layer to OCC	Structural:FoilStamping	Yes	equal to	Yes	Delete
17	ColdFoilStamping	Add layer to OCC	Structural:ColdFoilStamping	Yes	equal to	Yes	Delete
18	Embossing	Add layer to OCC	Structural:Embossing	Yes	equal to	Yes	Delete
19	Debossing	Add layer to OCC	Structural:Debossing	Yes	equal to	Yes	Delete
20	Perforating	Add layer to OCC	Structural:Perforating	Yes	equal to	Yes	Delete
21	Structural positions	Add layer to OCC	Structural:Bleed	Yes	equal to	Yes	Delete
22	Folding	Add layer to OCC	Structural:Folding	Yes	equal to	Yes	Delete
23	Punching	Add layer to OCC	Structural:Punching	Yes	equal to	Yes	Delete
24	Stapling	Add layer to OCC	Structural:Stapling	Yes	equal to	Yes	Delete
25	Structural positions	Add layer to OCC	Structural:VarnishFree	Yes	equal to	Yes	Delete
26	Structural positions	Add layer to OCC	Structural:InkFree	Yes	equal to	Yes	Delete
27	Structural positions	Add layer to OCC	Structural:InkVarnishFree	Yes	equal to	Yes	Delete
28	Positions	Add layer to OCC	Positions:Hologram	Yes	equal to	Yes	Delete
29	Positions	Add layer to OCC	Positions:Barcode	Yes	equal to	Yes	Delete
30	Positions	Add layer to OCC	Positions:ContentArea	Yes	equal to	Yes	Delete
31	Positions	Add layer to OCC	Positions:CodingMarking	Yes	equal to	Yes	Delete
32	Positions	Add layer to OCC	Positions:Imprinting	Yes	equal to	Yes	Delete

You may adjust the OCCD configuration to your own needs.
Each of the OCCD sets will end up in a single PDF file.

Split the PDF based on the OCCD information

In this step the "Split PDF" Action is used to split the PDF layer structure into single PDF files.

25.10 Index layer names if initial visibility or Processing Steps metadata is different

Layers can have regular and/or processing steps metadata attached to them.

When PDF files with identical layer names are merged together (for example, by the means of overlay), the layers are also merged.

This is problematic if the layers have different parameters, most importantly, different initial visibility or Process Steps metadata.

Indexed layer names

The PDF in the screenshot below has various 'layers' like 'Varnish', 'White' etc, as shown in the screenshot.



A modified version of the original PDF with reduced number of layers is now placed on the original PDF using 'Place content on page' Fixup.

This should merge the layer names as well.

Starting pdfToolbox 13, the layer names are indexed when PDF files with identical layer names are:

- **Merged (Merge PDF)**
- **Imposed (Imposition or imposition based Actions like Overlay)**
- **Place content (with a PDF)**



Please note that layer names are indexed and not merged when:

- initial visibility is different
- Processing Steps metadata is different

26. PDF 2.0 in prepress

26.1 Check for print and prepress related PDF 2.0 features

PDF 2.0 was published by ISO in July 2017. In it's introduction chapter is stated that a lot has changed:

- 14 completely new features
- 26 extensions to existing features
- 22 deprecated features

All PDF 2.0 features will usually be treated as "unknown" data for older equipment. Since the PDF concept always allowed for extensibility such processors will ignore them. That means that in most cases current workflows will not crash with a PDF 2.0 file. Problems will, however, occur once PDF 2.0 features are expected to be taken into account for print production.

The Profile below checks for new PDF 2.0 features that may have relevance for such workflows. It requires pdfToolbox 9.4 or newer.



Uses_PDF_2.0_features.kfpx

Which features are new and could require changes in present PDF based prepress workflows?

PDF 2.0 entry in the PDF header

Each PDF has in it's header the PDF version according to which it has been produced. A newer PDF creator may very well write "%PDF 2.0" without using any of the new features.

A PDF processor who reads this entry may refuse to process the file, when it expects this value to start with a '1'.

The PDF Association has published some very simple PDF 2.0 files that may be used in order to test workflows whether they can in principle process those files.

DPart metadata for PDF pages

DPart metadata is associated with PDF pages or parts. Other than XMP metadata it is organized in a tree so that it is easier for any PDF processors to identify pages or page ranges that have certain properties. It was initially invented in PDF/VT and is taken over from there into PDF 2.0. It may be used e.g. in order to mark certain pages for certain processes, e.g. for the first and last pages of a PDF to be output in 4c (to form a cover) and for the remaining page to be output in b/w (to form the inner part).

Page based Output Intents

Output Intent objects were until PDF 2.0 always associated with a PDF as a whole. PDF 2.0 makes it possible to have page based Output Intents, e.g. for the same example as before: first and last pages have a 4c Output Intent while the remaining pages have one for gray.

Black Point Compensation entry

BPC is a parameter for color conversion that takes the maximum black of source and destination color space into account. Usually results are better when applied, so most color conversion engines just use it. However, in some cases this is not desired and up until PDF 2.0 it was not possible to treat objects different in a single PDF. It was not even possible to switch BPC on or off for the whole document. Both becomes possible with PDF 2.0 and PDF creators may now mark certain objects for color conversion using BPC and other for not using it.

CxF and Mixing Hints in Output Intent dictionaries

Both entries provide data for processing of spot colors in a PDF.

Mixing Hints contain information about expected results when spot colors interact with each other in print.
This information might be used e.g. by a proofer. Until PDF 2.0 Mixing Hints could be present in an NChannel color space, in PDF 2.0 also in an Output Intent dictionary for the whole PDF. (That actually makes more sense, since such interacting spot colors are not necessarily part of an NChannel object).

SpectralData has spectral measurements for spot colors in *CxF/X-4* form (ISO 17972-4).
Measurements are required for blank substrate (0% and 100%) and optional for intermediate values. In addition measurements on black preprinted substrate are possible to allow for calculating results from spot colors that interact with each other in the same object or by means of overprint or transparency.

Halftones

The new HTO entry allows for specifying the halftone origin.

Requirements array in Catalog

The Requirements entry in the Catalog indicates which requirements a PDF file has.
Up until PDF 2.0 the only possible entry in Requirements was “EnableJavaScripts”. In 2.0 there are many new types e.g. PRC, Attachment or DPartInteract.

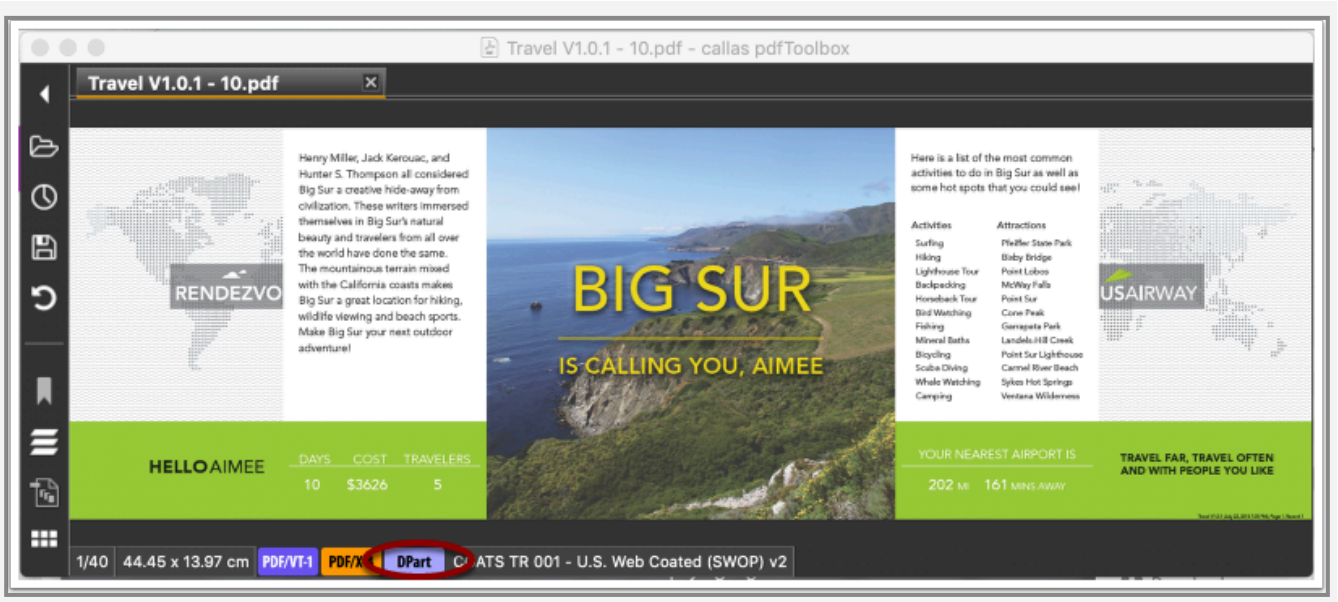
26.2 Display DPart metadata

DPart metadata is associated with pages or page ranges. It was specified in PDF/VT first and is defined in PDF 2.0 as well. it is intended to be used in automation to allow for processing pages in the same PDF in different ways. This can be done in pdfToolbox using QuickCheck which is explained here...

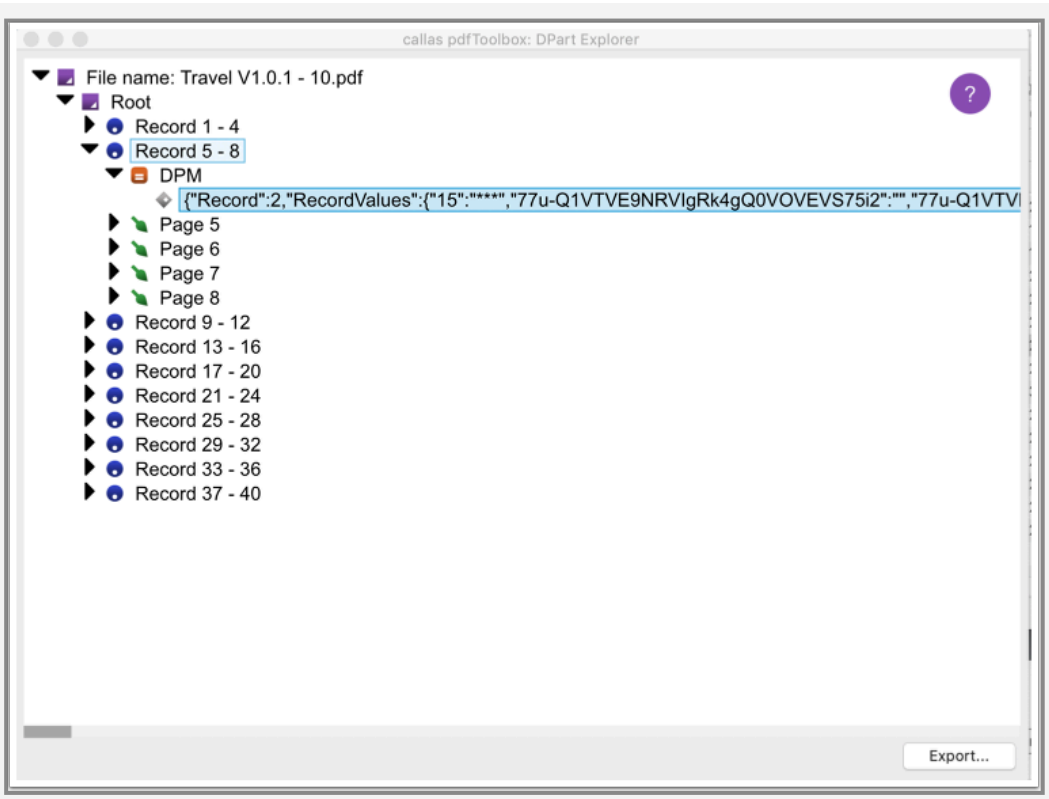
But pdfToolbox also is as of today the only tool that can easily display such metadata.

When a PDF has DPart metadata a button is indicating this at the bottom of the pdfToolbox window.

A suite of sample files including the one below is available from the PDF Association:
https://www.pdfa.org/resource/cal-poly-pdfvt-test-suite/



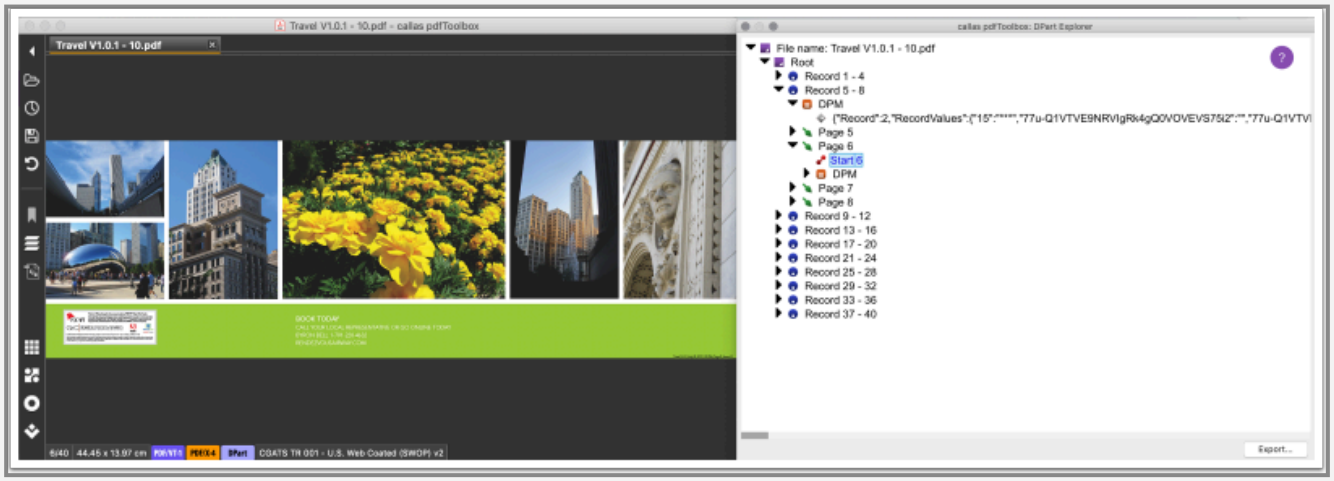
Clicking on this button opens the DPart viewer.



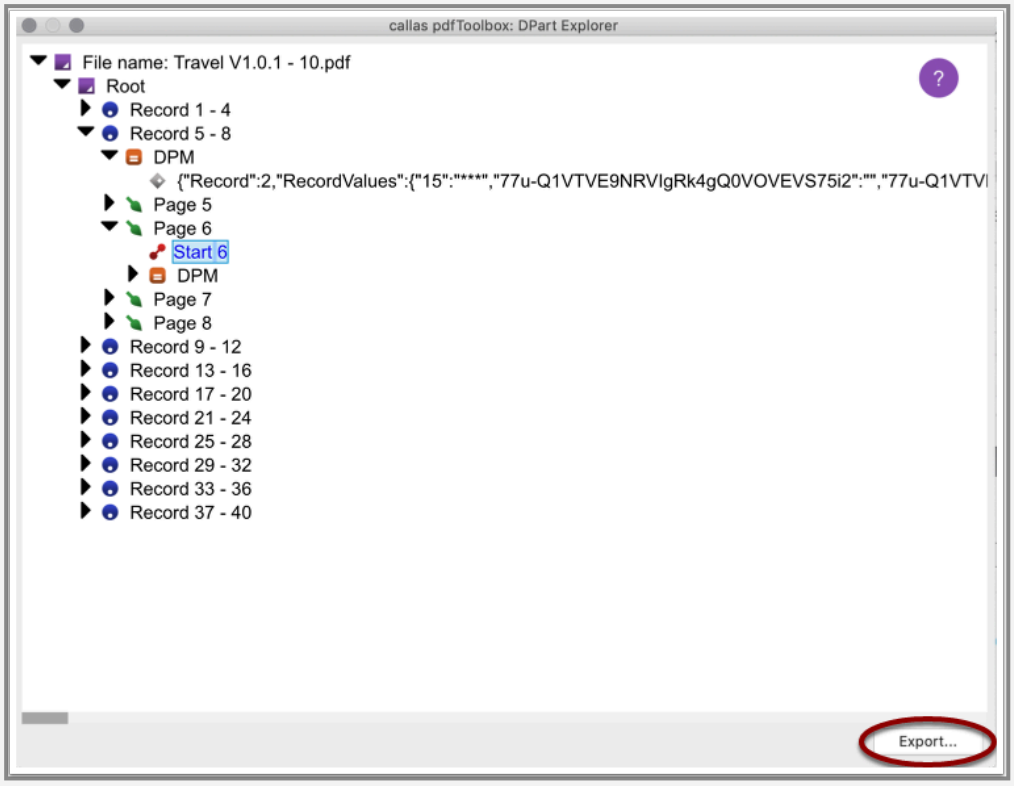
It shows Records, leafs and DPM nodes. Records are expected to be present in each DPart hierarchy on at least one level to define where records are differentiated. pdfToolbox shows right after the name (in this example "Record" the pages with which the respective record is associated.

DPM nodes contain the actual metadata which may in the PDF use any format (key - value pairs, XML, PDF syntax) but is in pdfToolbox converted into JSON.

Leafs hold the page associations which may be page ranges, but are in this example only single pages. The leaf entries contain links to the respective pages in the DPart viewer.



An Export button allows for saving the structure as a JSON file to the disk.



Here you can get to know everything about DPartner and DPart metadata:

26.3 Use DPart metadata in a Process Plan via QuickCheck

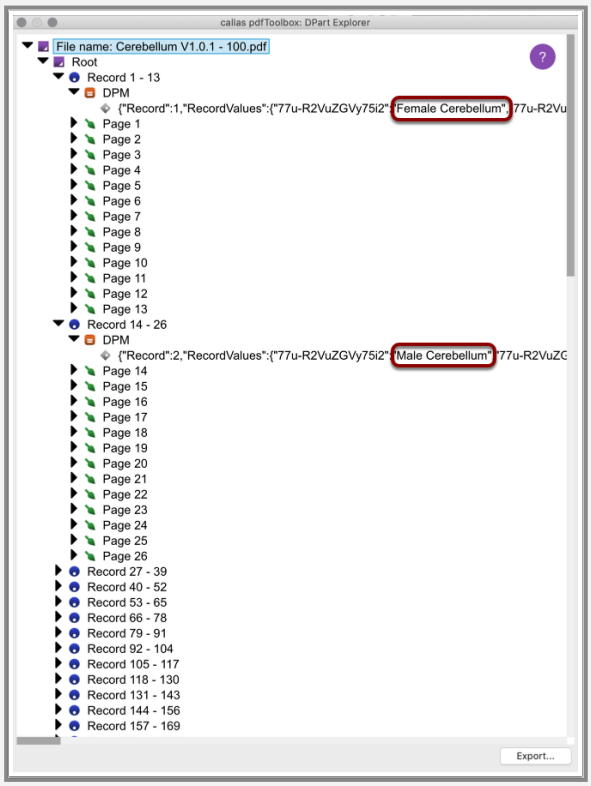
You can extract DPart metadata from a PDF file into a JSON structure via QuickCheck, e.g. as the first step in a Process Plan and then use this structure in the following steps.

The attached Process Plan demonstrates how to do that.

Remove_pages_with_DPart_"Male_Cerebellum".kfpx

It is very specific to the DPart structure and can only be used with one of the "Cerebellum ..." sample files from the "Cal Poly Graphic Communications PDF/VT Test File Suite" that is available from the PDF Association:
https://www.pdfa.org/resource/cal-poly-pdfvt-test-suite/

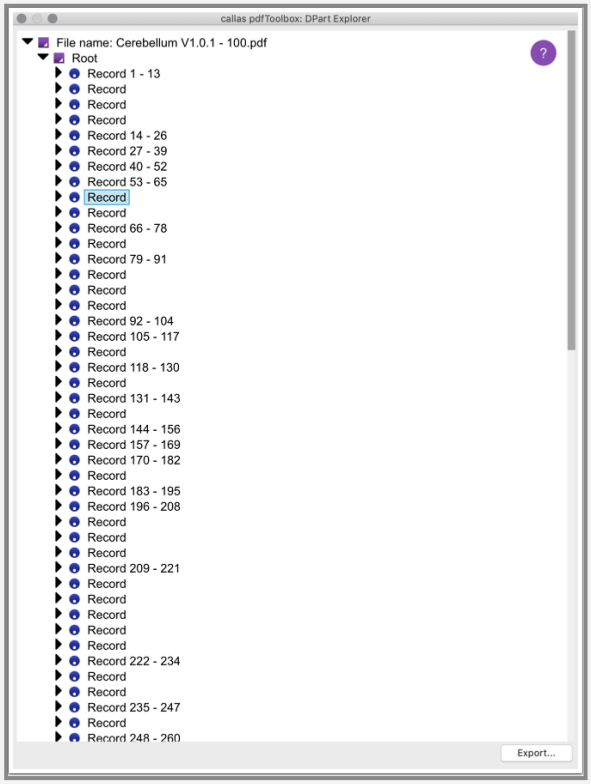
When you open the PDF in pdfToolbox you may first analyze the DPart structure (as described in Display DPart metadata).



When you open some DPM entries you will see that some have a value with "Male Cerebellum" while others have "Female Cerebellum".

In this example we use this metadata to decide what to do and in this example that is to remove all pages that are associated with "Male Cerebellum".

When you apply the Process Plan above to one of the "Cerebellum ..." files it will reduce the number of pages and after that operation many of the records will not have any page associations anymore, since the pages have been removed.



You can immediately identify records with pages and without pages since the DPart viewer displays page numbers after each record. If you open those entries and look into the DPM entries you will see that only Female Cerebellums are left.

26.4 DPart metadata injection

DPart can be added to a PDF file using Quick Fix as described in a chapter of the QuickFix feature overview.

The InjectDPart feature in Quick Fix uses a page list as input. In a Process Plan this list can be dynamically created. pdfToolbox (version 12 or newer) has a predefined Process Plan "Create DPart record information from headings" that does so. It uses the text size as an indication for a heading and creates a DPart structure in which each page with a heading is referenced as a record start.

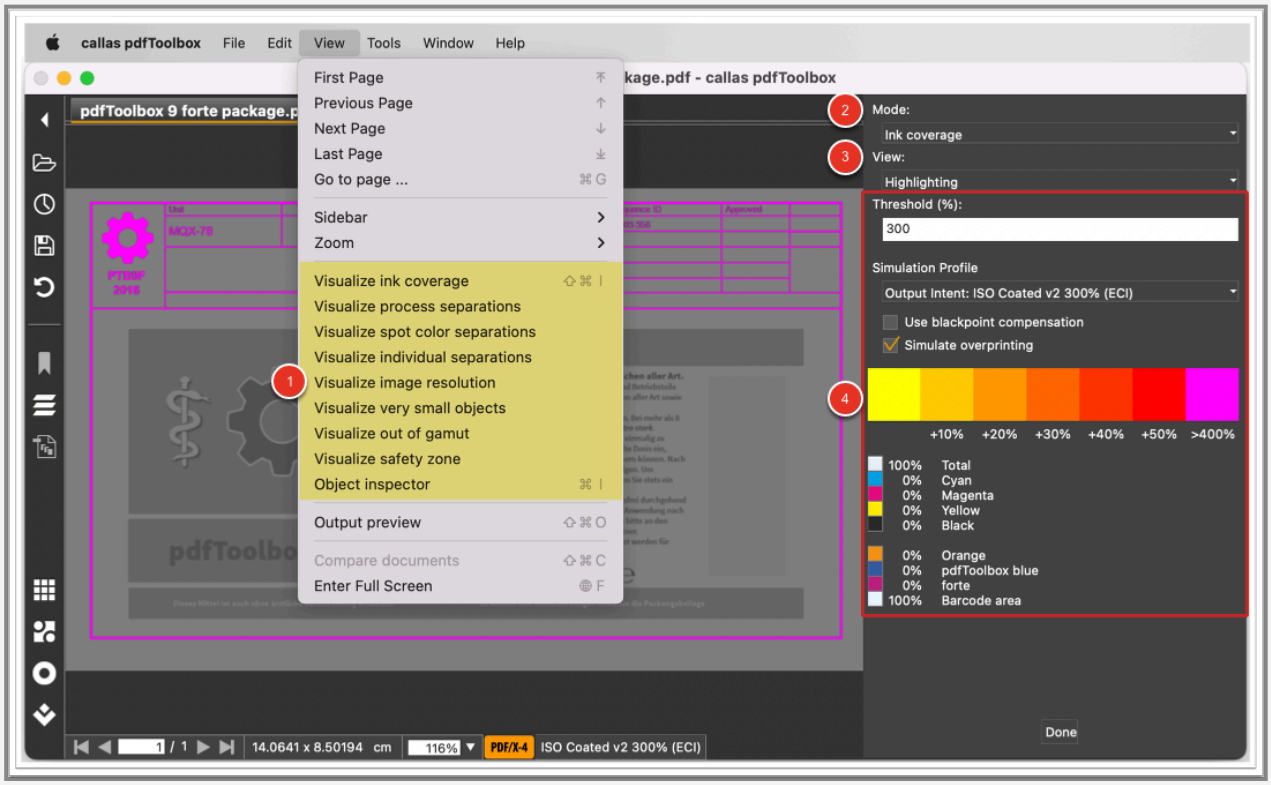
27. Interactively analyse and explore PDF documents

27.1 Visually inspect PDF files

pdfToolbox provides a visualizer technology to visually inspect a PDF file. With this you get a number of different tools to help you with the analysis of problematic PDF files. This article gives you an overview of the different possibilities.

Using the visualizer to inspect the PDF document

Sometimes knowing which objects are in the PDF document isn't sufficient, and in those cases the visualizer technology in pdfToolbox comes to the rescue. The Visualizer allows you to explore aspects of a page that may be relevant for printing purposes.



1. To access the Visualizer, use one of the "Visualize..." menu items under the "View" menu in pdfToolbox Desktop.
2. Depending on the menu item you use, a different visualizer "Mode" will be shown. You can use the pull-down menu here to select different modes without having to go through the menus again.

3. Each visualizer mode has different views. Use this menu item to select a different viewing mode.
4. Each visualizer mode has different additional information and options.

Visualize ink coverage

The ink coverage mode of the visualizer overlays the PDF document with a gray mask and highlights any area of the file that goes over a certain threshold with bright colors. Read more about it in this article: View ink coverage per separation.

Visualize seperations

With the different separation preview modes you can view single separations, just the process separations, just the spot color separations and more. Read more about it in this article: Display ink coverage information for all separations.

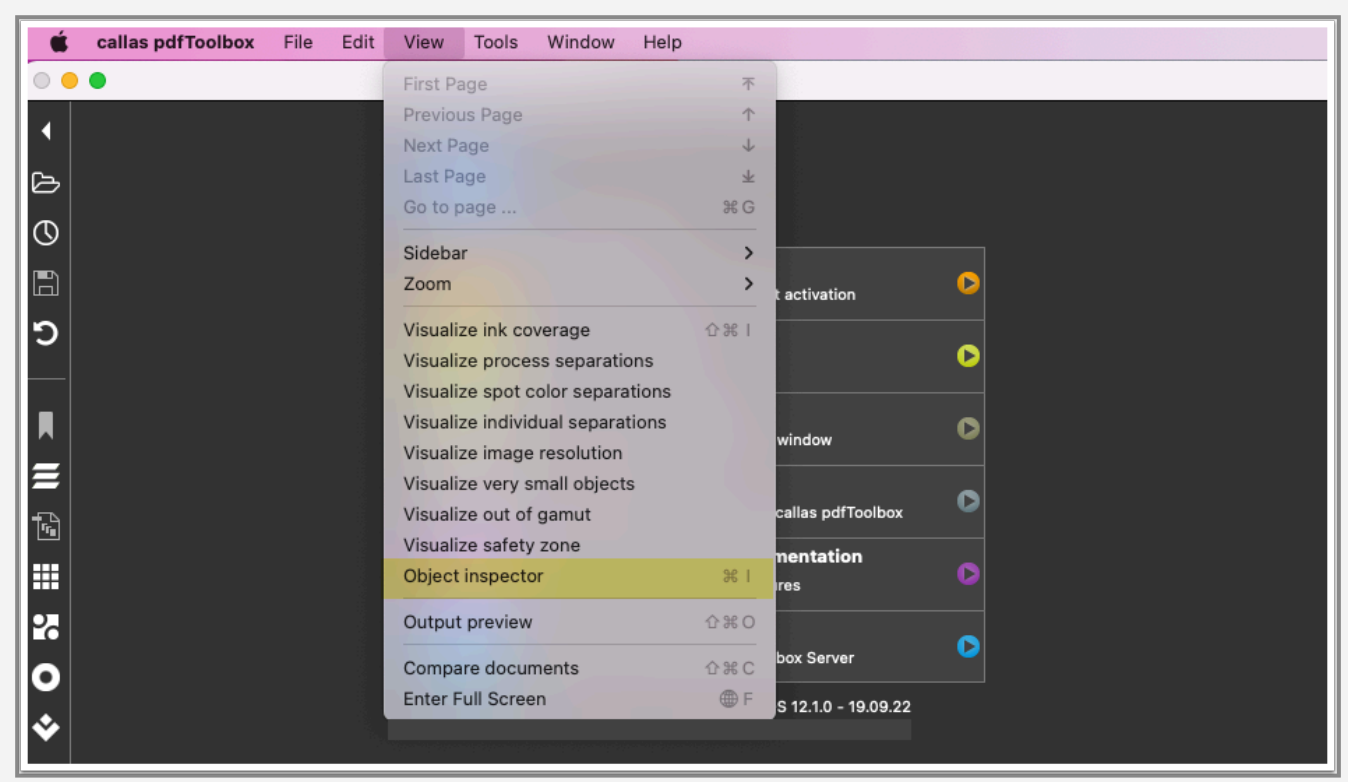
Visualize safety zone

In this mode, a safety zone can be defined and visualized. Read more about it in this article: View safety zone in PDF.

Visualize out of gamut

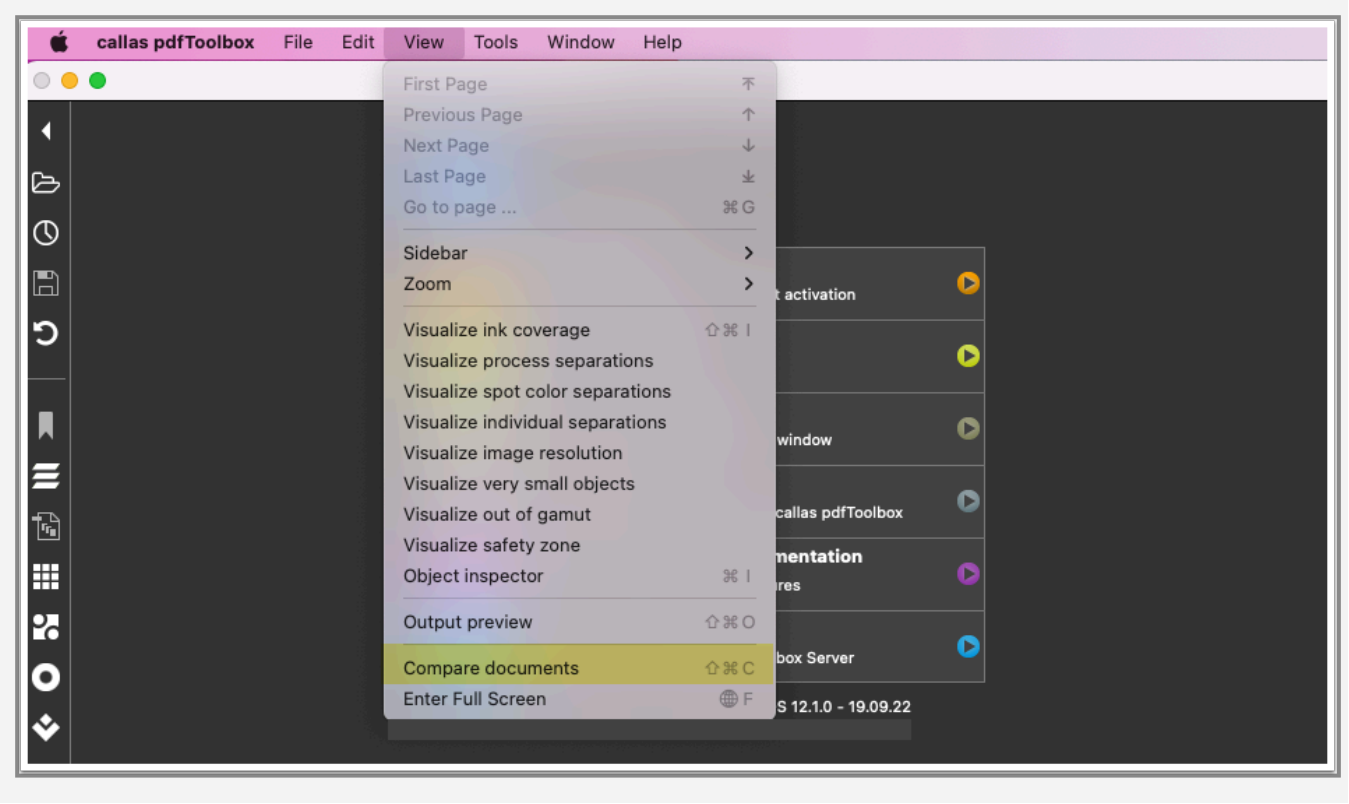
This feature compares the currently opened PDF page based on its actual or assigned source color space against a color converted version of that page in an out of gamut heat map view. Read more about it in this article: Heat map for «Out of gamut» visualization.

Using the object inspector



The Object Inspector allows identifying the page content by showing which objects are on the PDF page (e.g. text, images, vectors, shadings) and displaying the respective attributes for them. Read more about it in this article: [Examining page content: The Object Inspector].

Compare PDFs



The 'Compare documents' feature compares two PDF files. As result, all differences are displayed in the document using masks. Read more about it in this article: Compare documents.

Visualizer on CLI

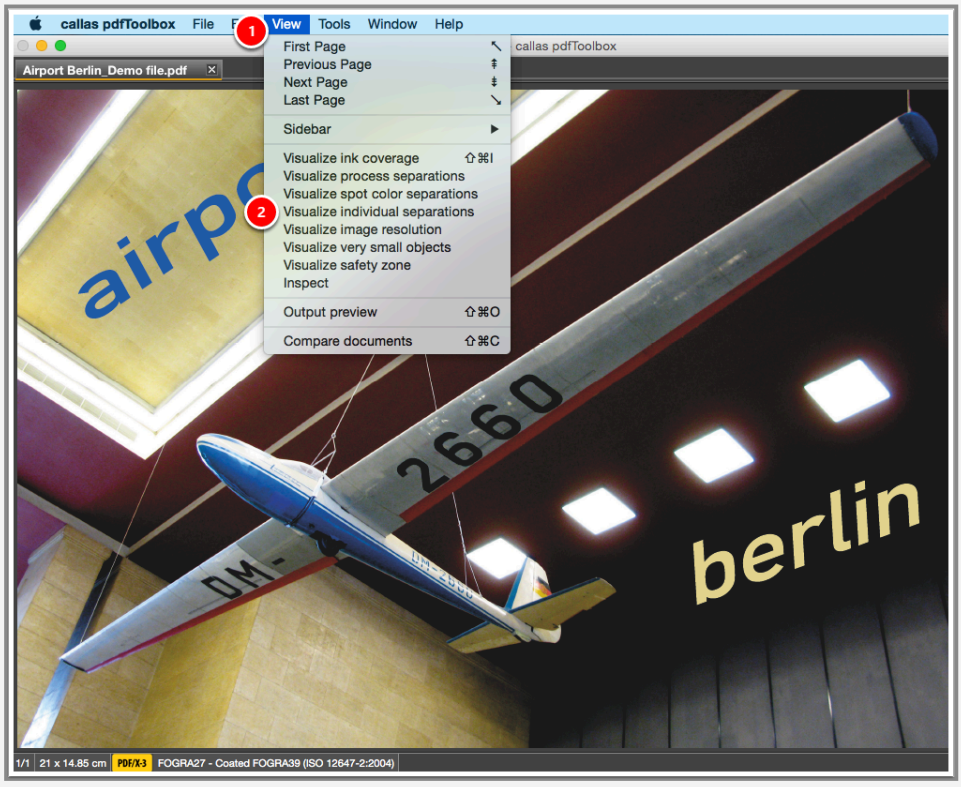
The visualizer is also available on CLI. In this article you can find all command line options.

27.2 Display ink coverage information for all separations

Since pdfToolbox 8.0 you have the opportunity to select individual color separations in the PDF file using the Visualize panel to view the values. In version 8.1 the tool is extended with ink coverage information for all separations.

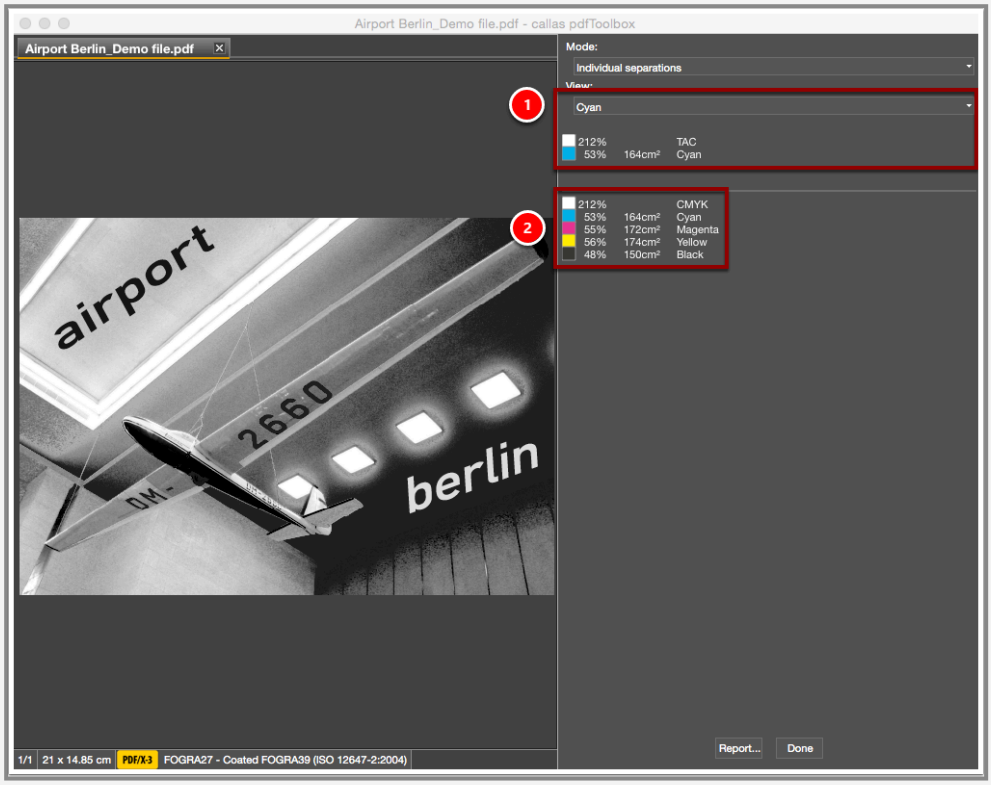
Airport_Berlin_Demo_file.pdf

Open "Visualize individual separations" panel



1. Go to "View".
2. Click "Visualize individual separations".

Inspect the PDF file



1. Select in the list the color separation that you want to view. The values of the selected color separation showes below. The image will change.
2. The user can see not only the values of the selected color separation, but also the values of all other color separations in the PDF file.

Detect and list separations using Preflight Checks or Reports

It is of course possible to detect colorants using a Preflight Check, e.g. by using the predefined "Analyze pages for effectively used plates" or the attached Profile, which has been extended by separate Checks for individual CMYK colorants. The percentage ink coverage is listed in the trigger values of the respective Check.



Analyze_pages_for_effectively_used_plates_(extended).kfpx

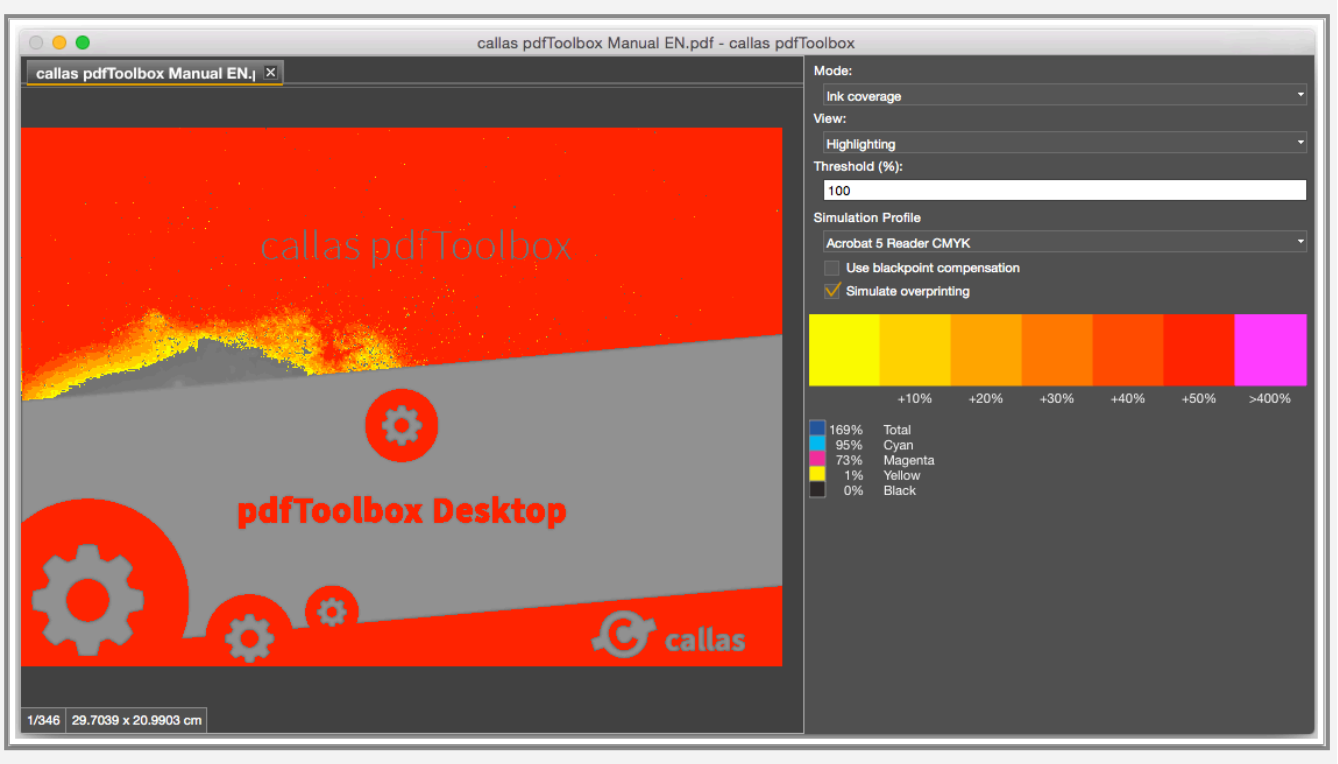
27.3 View ink coverage per separation

The Visualizer allows you to explore aspects of a page that may be relevant for printing purposes, such as color applications, color spaces or page object types. You will find the option in the "Reports" category and in the "View" menu of the standalone edition. Below the viewing options for Ink coverage in pdfToolbox.

Highlighting

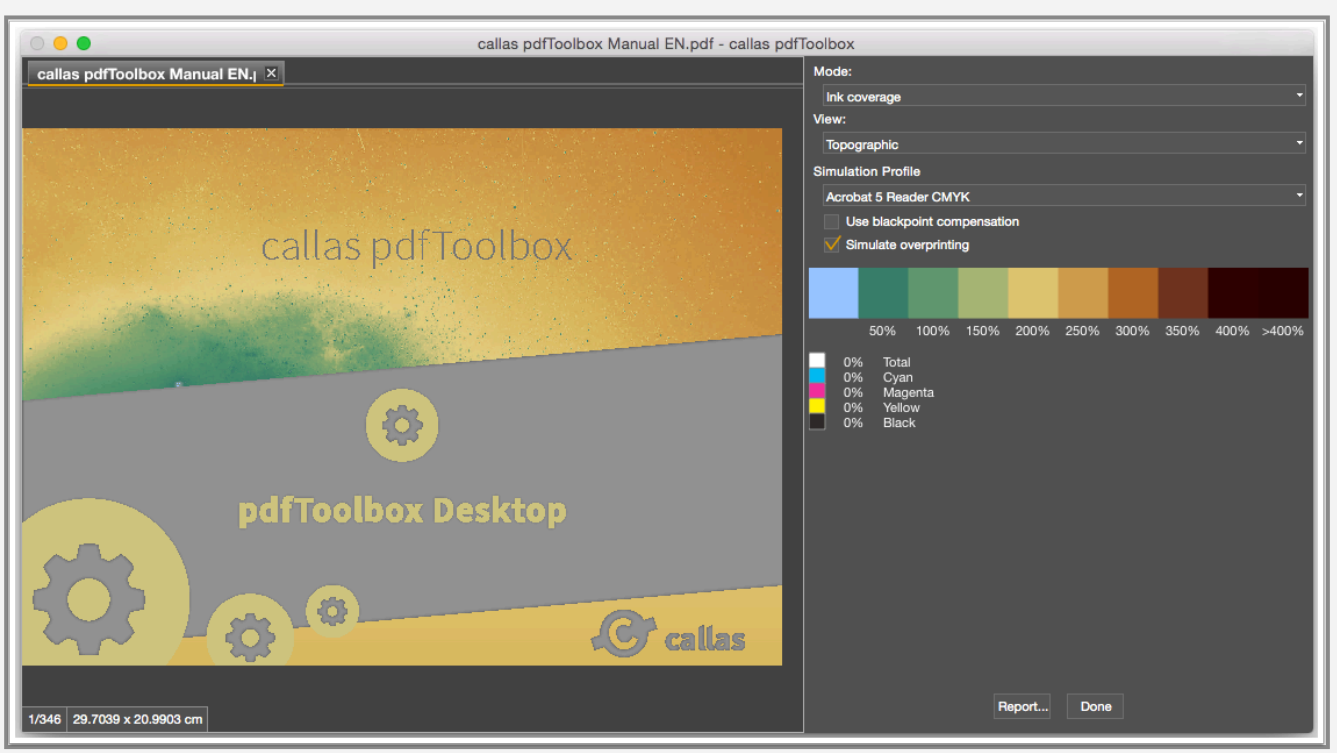
You can use the color application to show problematic regions, as well as for other purposes.

This example shows where the color application exceeds the threshold of 100%.



Topographic

It is also possible to display a page in a similar fashion to a map.

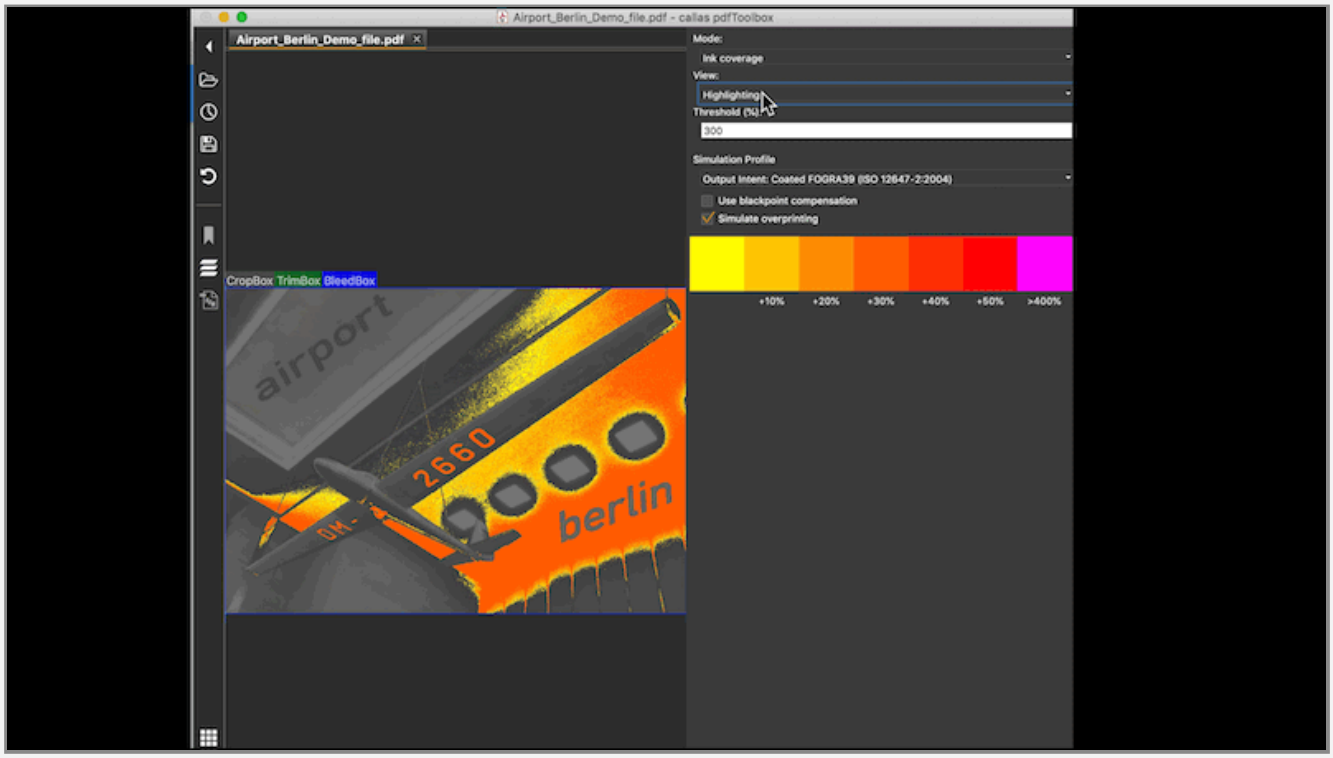


Ink coverage per separation

In packaging, it is sometimes helpful to find out whether there is any separation (ink) with less than 6% ink or more than 96% ink in certain areas. This heat map view helps to find out exactly that:

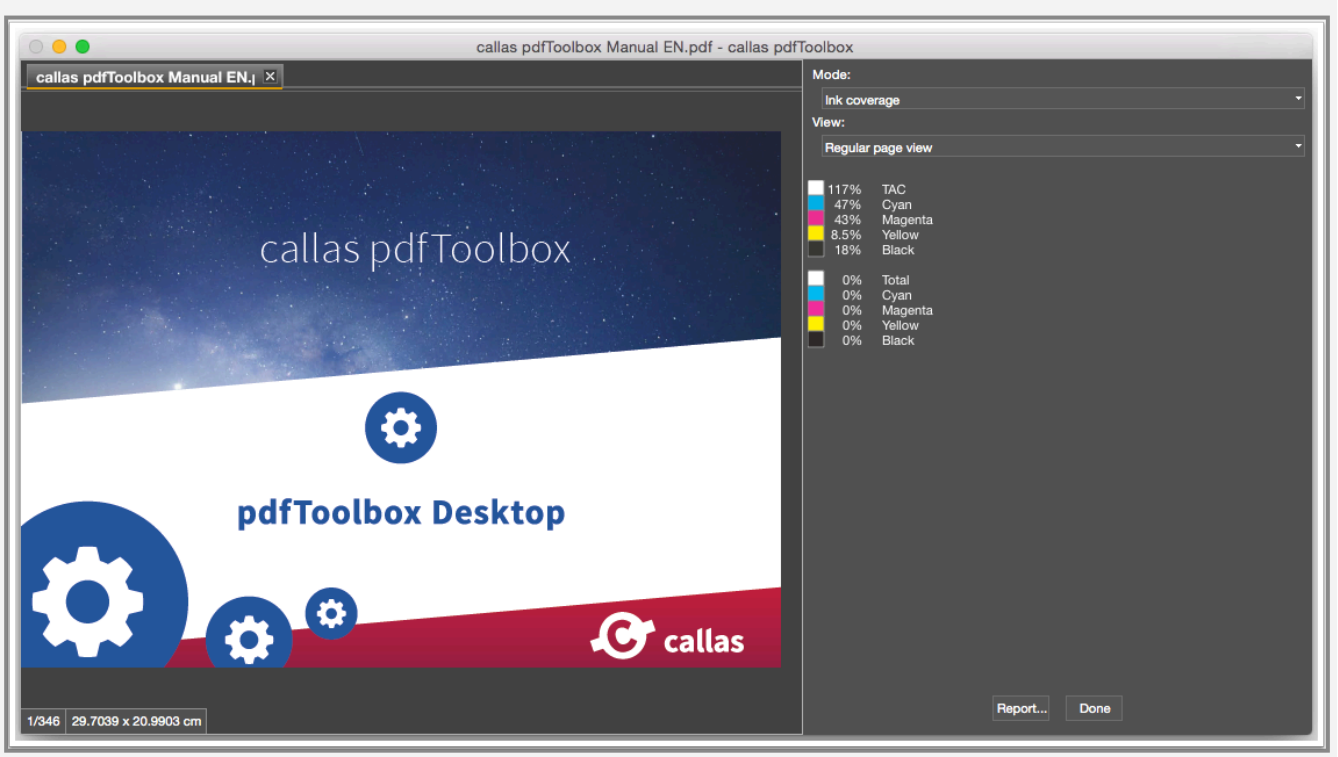
- Select 'Ink coverage per separation' view from the drop down
- Activate Low and/or High ink coverage, as per requirement
- Indicate the minimum and/or maximum threshold percentage
- Activate/deactivate a separation using the checkbox
- Display ink coverage with mouse-over functionality for ink coverage of current mouse position

As shown here:



CMY channels

You can also view different color separation options or make small objects easier to spot.



You can find CLI parameters for these views under
'Reports'
or simply:

```
./cli/pdfToolbox --help visualizer
```

Click to copy

Usability features:

You can use keyboard shortcuts to navigate easily through the Visualizer dialog box:

- *Ctrl + Left arrow:* Next mode

- *Ctrl + Right arrow:* Preview mode

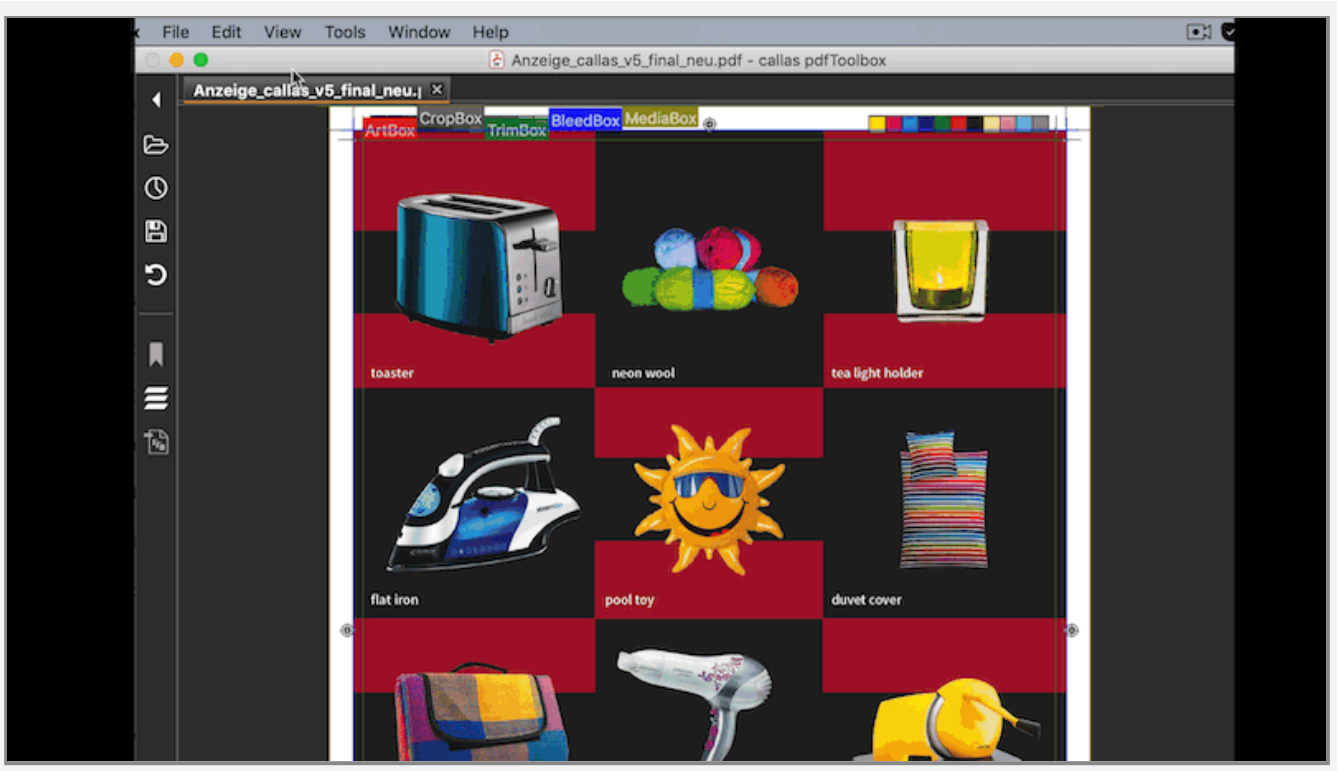
- *Left arrow:* Next view type

- *Right arrow:* Previous view type

27.4 View safety zone in PDF

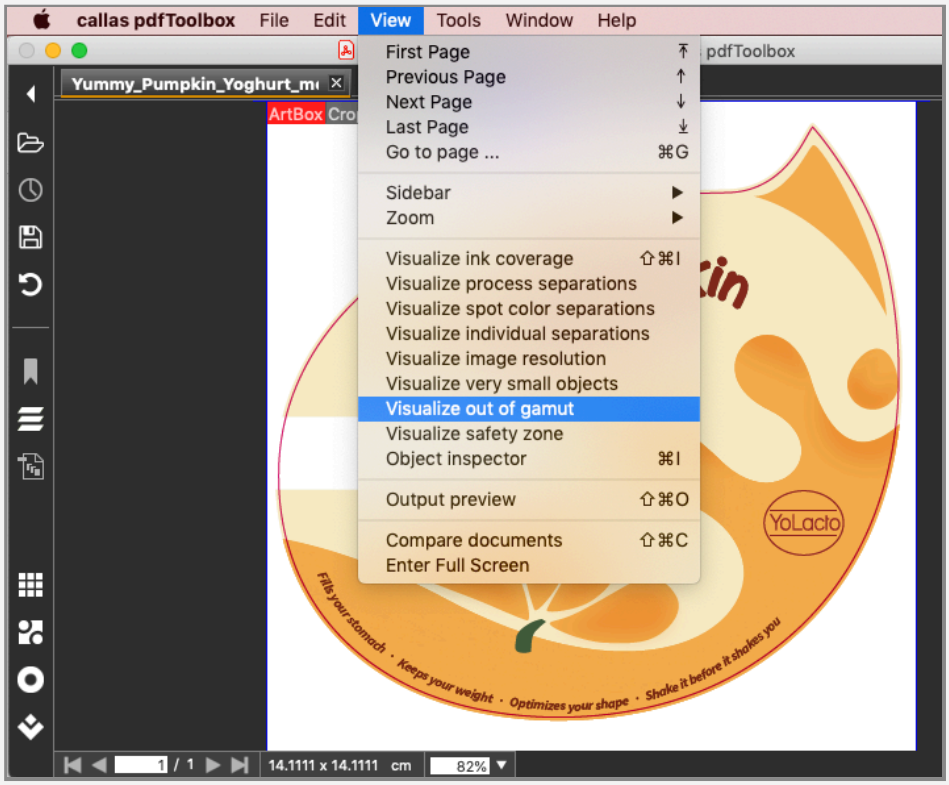
There is a view in pdfToolbox 12 which is a merge of views known as “Bleed area” and “Page border” in previous versions. This view offers the possibility to define the width of the security zone, which wasn't available yet.

Depending on the entered values, the security zone is updated and the result can have either a highlighting inside (like the current “page border” view), inside and outside (like “bleed area) or just outside (new possibility), as shown below.

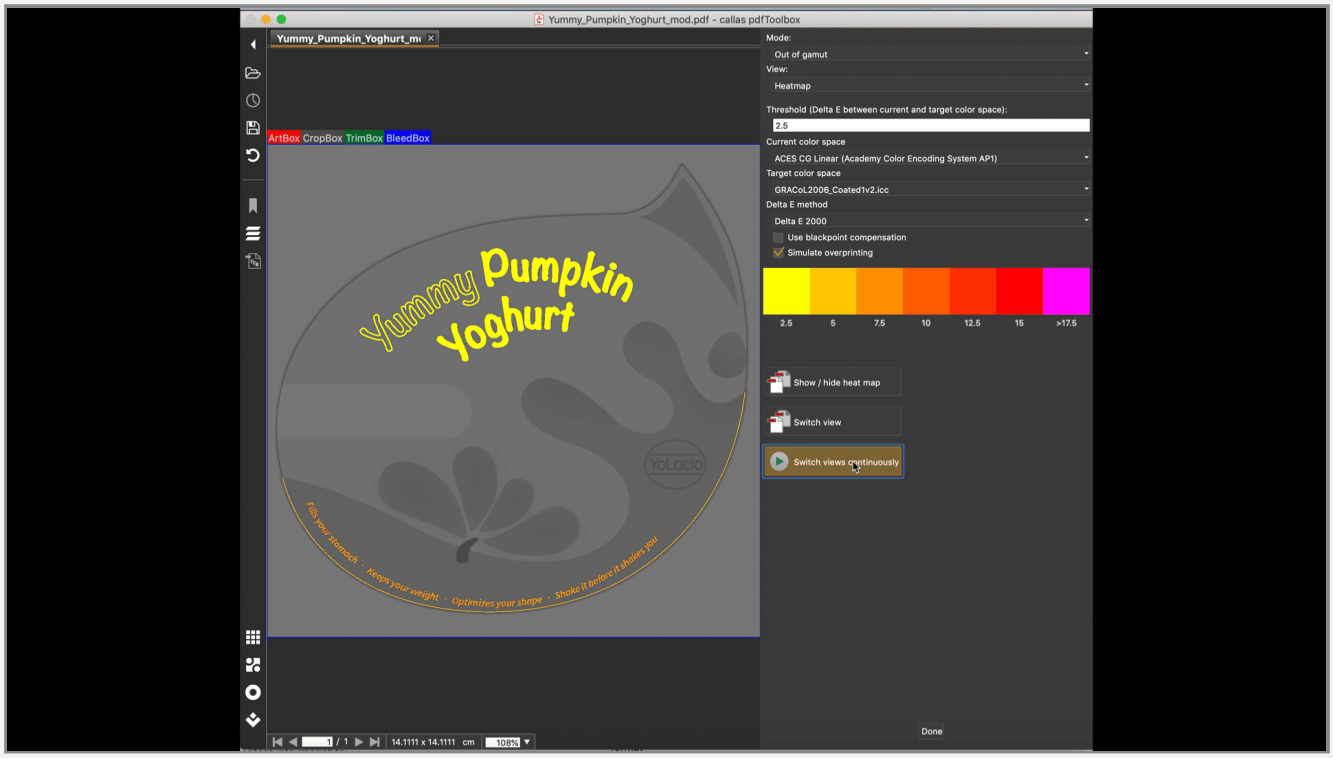


27.5 Heat map for «Out of gamut» visualization

This feature compares the currently opened PDF page based on its actual or assigned source color space/output intent profile against a color converted version of that page in an out of gamut heat map view.



Understand the UI:

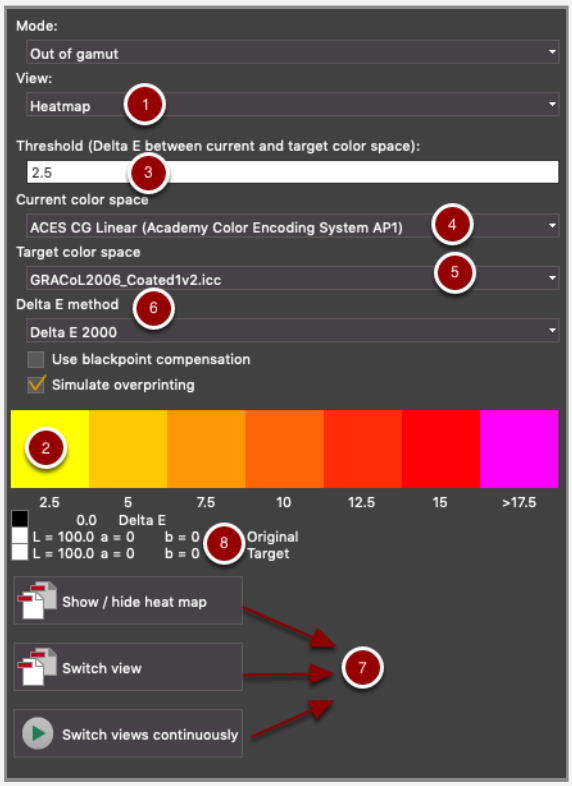


1. The feature offers views that can be toggled between different specialised views like the original color space image, target color space image, overlay with heat map and regular page view.
2. For each pixel, the color distance (in delta E) is computed and the result is represented as a pixel image where each pixel represents that distance by using a certain color (the ‘heat map image’); the colors range from yellow over red to pink
3. Threshold distance: Pixels below a threshold distance of e.g. 2.5 delta E shall be transparent
4. The current “color state” is the ‘current document color space’
5. The “color state” that results from color converting the PDF page is the ‘target color space’
6. Delta E methods: Delta E 2000 OR Delta E 1976
7. Views:
   - Toggle between current and target view manually by clicking the switch view button

- Turn heat map view on and off manually by clicking the show/hide heat map button
- Continuously toggle between the two views

8. To find out what the color difference is in a given position, delta E value and matching color patch are shown for current mouse position
   - If delta E at mouse position is below threshold, the color patch is shown using black
   - If mouse position is outside of page area, delta E is shown as "–", and the color patch is shown using black

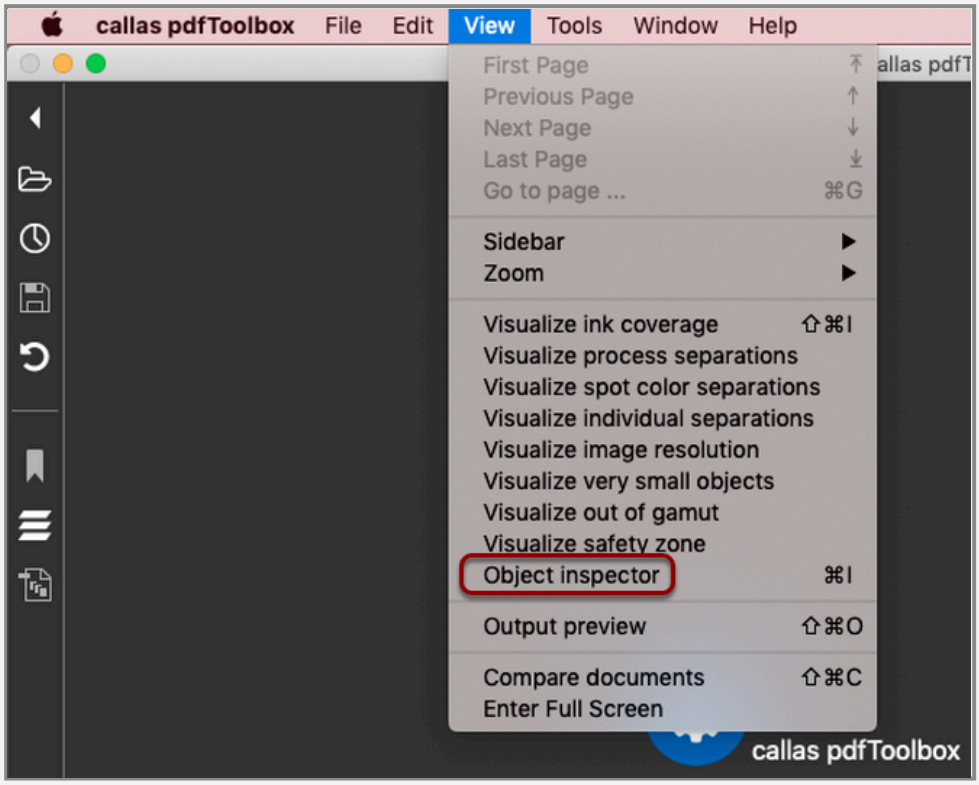
- The heat map image is placed on top of the current page view



27.6 Examining page content: The Object Inspector

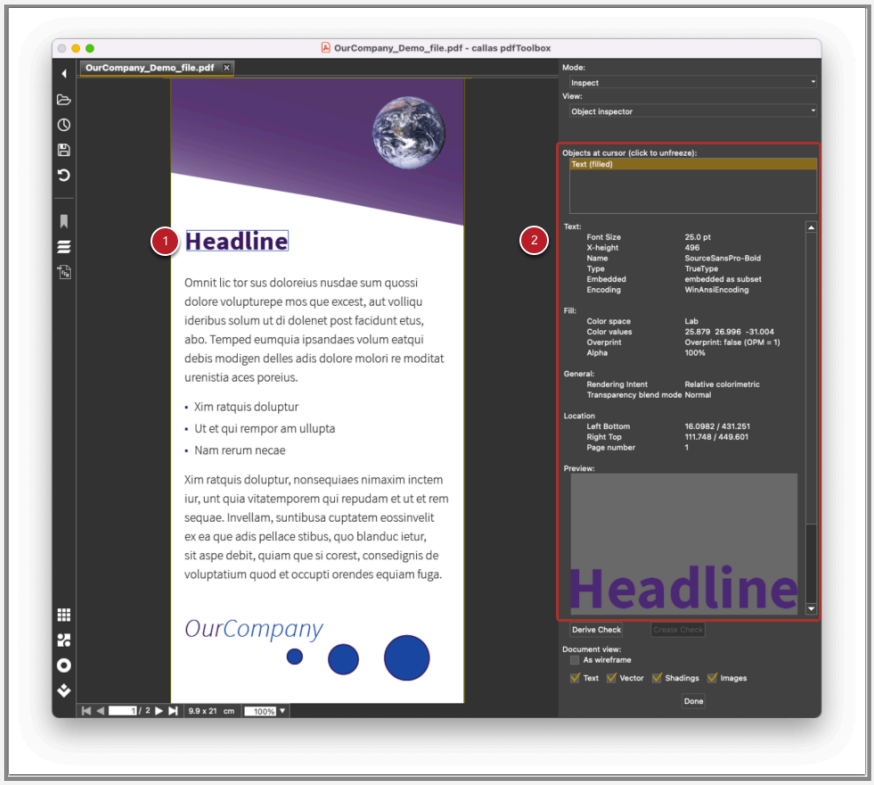
The Object Inspector is a view mode in pdfToolbox Desktop. It provides an interactive expandable list of properties for objects in PDF files. The Object Inspector window consists of two parts: A preview of the selected page on the left side, while the right side shows object-specific information (such as font, color, transparency, resolution...).

To open the Object Inspector in pdfToolbox, go to View > Object Inspector:



Object inspector: Example

OurCompany_Demo_file.pdf



The Object Inspector is context sensitive. When you move the cursor over any of the page objects on the PDF page, you will see a detailed preview and object-specific information for that object type on the right side (2). The object-specific information varies depending on the selected object type (see screenshot below).

NOTE: The view is fixed when you click directly on the desired object. Clicking again will trigger the connection of the selected object.

Object inspector: Example image



If you select for example the image in the top left corner (1), you will see image-related information on the right side (2), such as image dimensions, resolution, or compression.

Derive preflight checks from object inspector

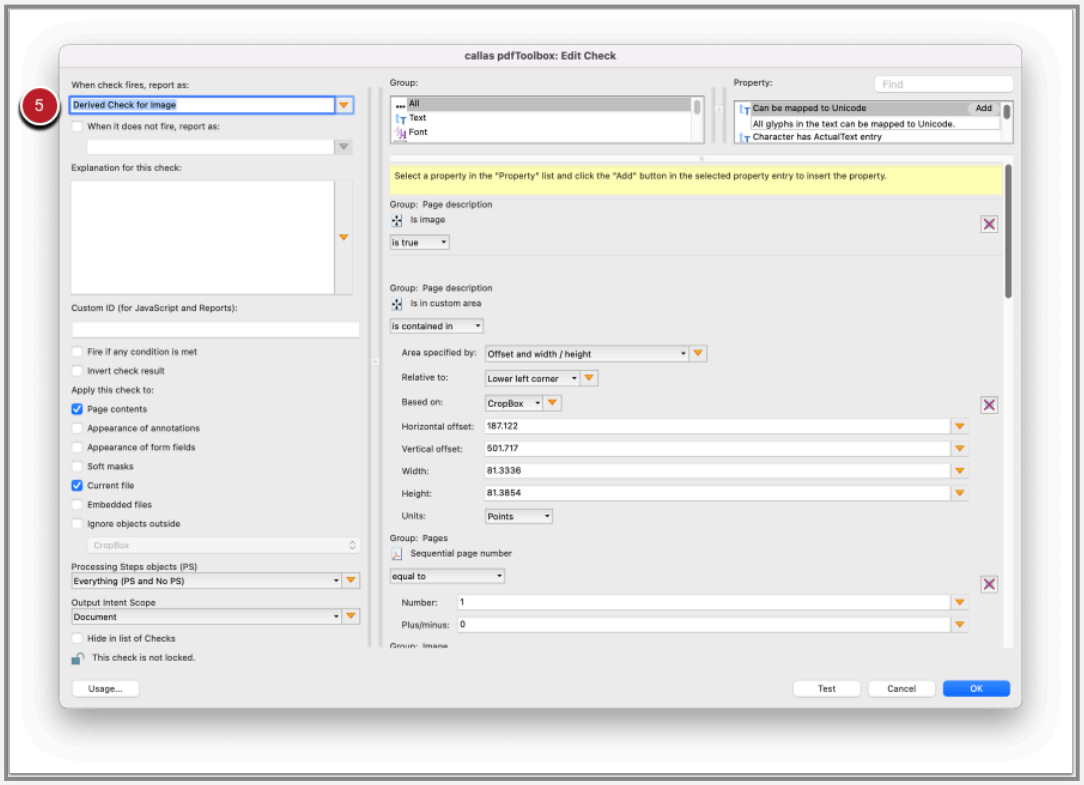
Supported since **pdfToolbox 15**

You can use the object-related properties determined by the Object Inspector to create a preflight check that consists of all or some of the properties of the selected object. The Object Inspector creates preflight checks that can be further customized. This can dramatically reduce the time required to create complex preflight checks.



1. Select a page object so that its attributes appear on the right.
2. Click the "Derive Check" button.
3. A checkbox appears next to each attribute (all checkboxes are checked by default). You can uncheck attributes you are not interested in. (Clicking the "Derive Check" button again will disable the mode).
4. The "Create Check" button is now enabled.
5. When you click the "Create Check" button, the "Edit Check" window opens. Depending on the selected checkboxes (attributes), the new derived check will have the corresponding check properties already set, including their specific values displayed in the Object Inspector.


OurCompany_Demo_file.pdf - callas pdfToolbox
Headline
OurCompany
3
4
Create Check


callas pdfToolbox: Edit Check
5
When check fires, report as:
Derived Check for Image
Explanation for this check:
Custom ID (for JavaScript and Reports):
Apply this check to:
Page contents
Group:
Property:
Select a property in the "Property" list and click the "Add" button in the selected property entry to insert the property.
Test
Cancel
OK

Shortcuts for enabling and disabling checkboxes:

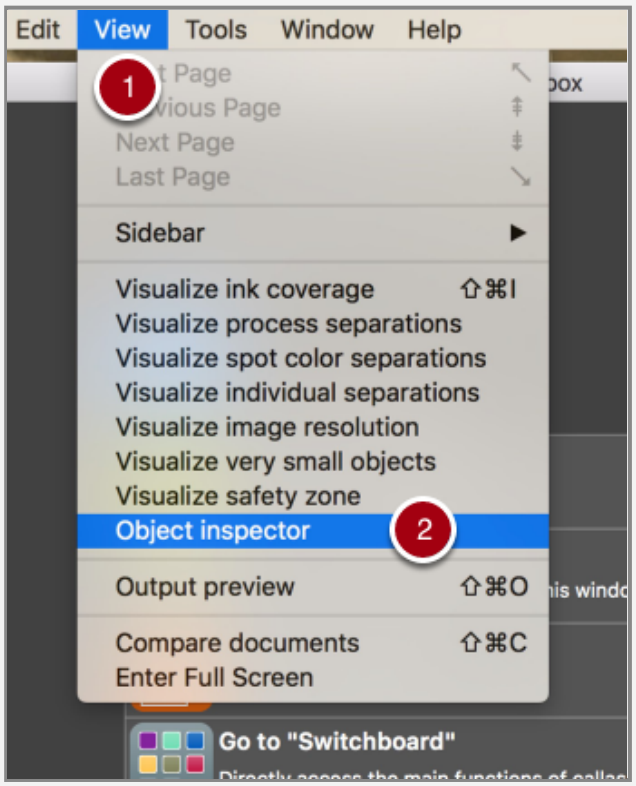
Mac: ⌘ (Cmd) -> Remove all selections except the current , ⌘ (Cmd) + shift -> Select all checkboxes

Win: Ctrl -> Remove all selections except the current , Ctrl + shift -> Select all checkboxes

27.7 Examining page content: Filter in the Object Inspector

In the visualizer section of pdfToolbox Desktop, the Object Inspector allows identifying page content, both viewing which objects form a PDF page, and what their attributes are. Two additional concepts: "Wireframe viewing" and "Object type filtering" have been implemented here.

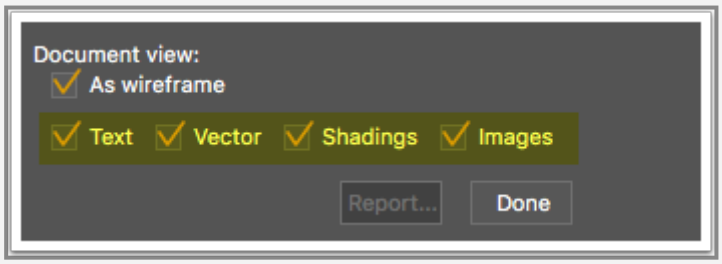
Showing the object inspector



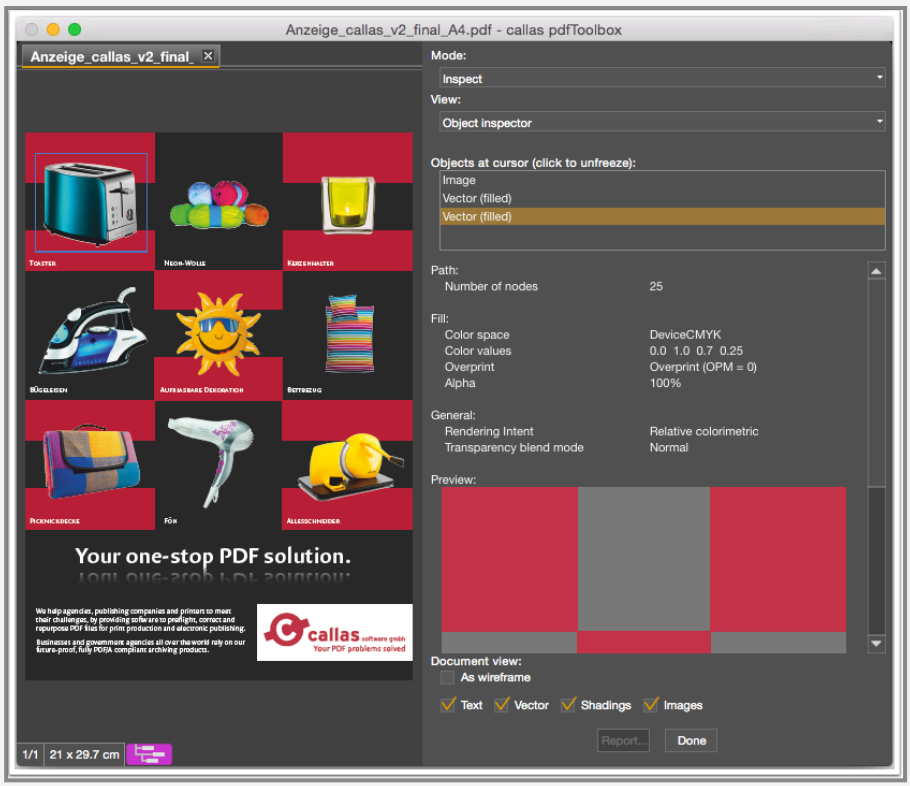
1. Click on the "View" menu
2. Select the "Object inspector" menu item

Object type filtering

By default, the object inspector shows all PDF objects on the page. This can be changed by disabling or enabling the checkboxes at the bottom of the Object inspector area. Deselecting "Text" for example, will cause all text objects on the page to be hidden. This allows examining whether objects are actually text for example, or allows viewing what is behind other objects (and normally hidden from view).



Object properties



Every single object can be selected by clicking on it inside the view on the left hand side. A click will fix the selection on the right hand side. Another click will release it.

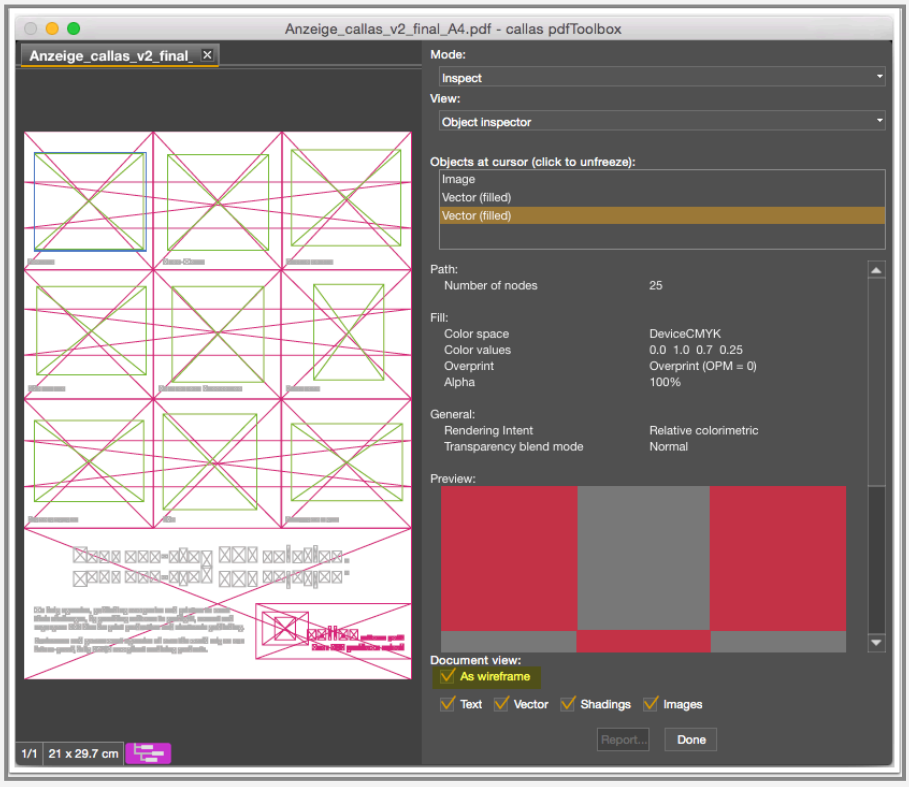
Selected objects are also visible in the preview on the right hand side.

Under "Objects at cursor", you will see a stack of all the objects in the same order as they appear in the PDF.

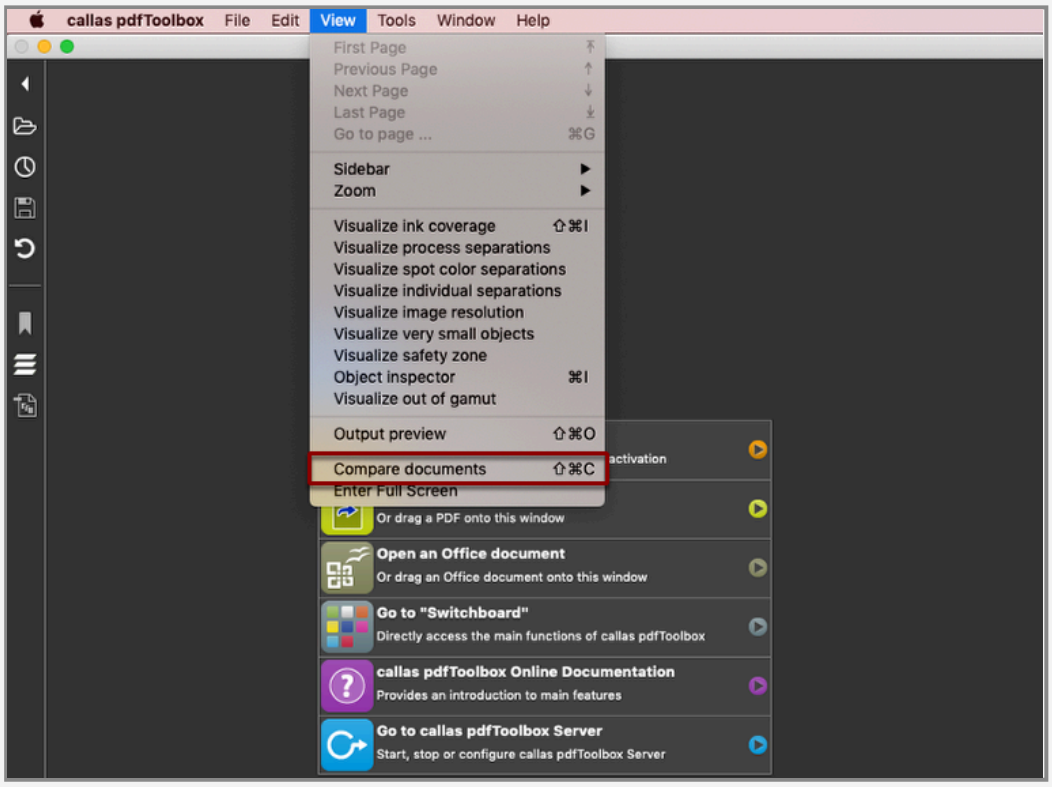
The topmost object is the topmost in the PDF. You can click at objects below the topmost object to see it's properties.

Wireframe viewing

At the bottom of the Object inspector area, check the "As Wireframe" checkbox, to show the displayed PDF page as wireframe. In this mode, all objects are shown with differently color rectangle outlines. This allows seeing the structure of the page (which objects are on the page, how they are layers etc...).



27.8 Compare documents

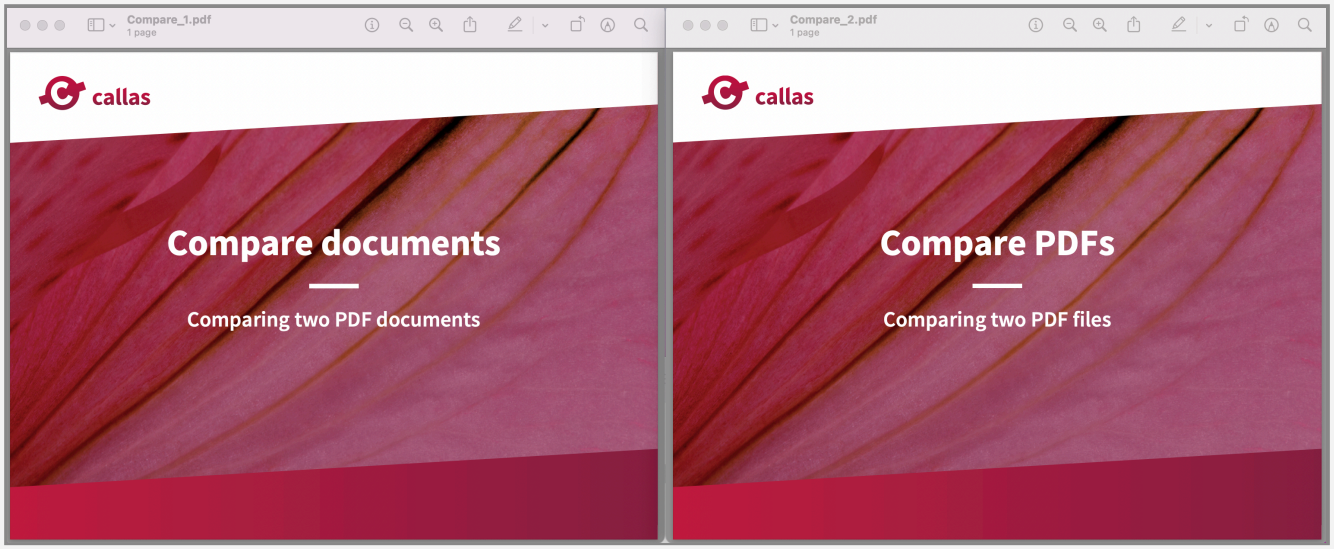


The 'Compare documents' feature compares two PDF files. The engine renders these two PDFs and then compares the rendered pixels of these two files. You can set different parameters to affect the comparison result. As result, all differences are displayed in the document using masks.

'Compare documents' is also available on CLI. In this article you can find all command line options.

Let's have a look at the different parameters:

Example files to be compared with each other



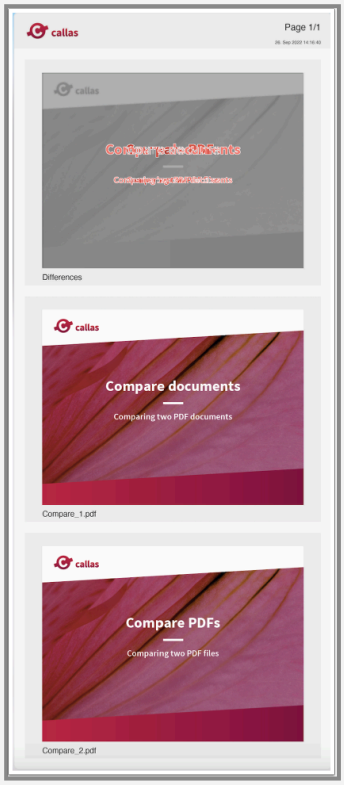


1. **Compare:** Defines which PDF files should be compared with each other.
2. **Sensitivity:** Here you can set, which differences should be highlighted (e. g. minimum sensitivity, maximum sensitiv-

ity, no highlighting). These are shortcuts to the more detailed parameters explained in 9 - 11.

3. **Switch documents:** Here you can switch between the two PDF files.
4. **Start quick change:** This option displays the two document pages alternately, so that the differences are easy to identify.
5. **Resolution (ppi):** You can specify the resolution for the rendering (in ppi). The higher the resolution, the more accurate the result (but, of course, the processing time will be longer).
6. **Channels:** Here you can select the channel to be compared (e. g. RGB, CMYK, K channel only…). If you want to do a color comparison, you can select "Delta C" or "Delta E" variants.
7. **Highlight:** Highlight appearance for the differences.With the default setting "Red mask" the differences are highlighted in red.
8. **Relative to:** Here you can specify the area that will be used for rendering.
9. **Per pixel:** Sets the threshold for the difference of the compared pixels. The unit depends on the option selected under "Channels" (Possible values: %, dE, dC).
10. **Total area:** Threshold in %, that defines how much percent of the area on the page can be different without generating a hit.
11. **Contiguous area (in mm):** The engine will only detect a difference if it is larger than the specified value. If you want to find differences only in larger areas, this parameter can help filter out small areas on the page.
12. **Measurements:** Values calculated by the engine, which refer to the differences on the page (cannot be adjusted)
13. **Go to next page with differences:** When you click on one of the arrows, the engine renders each page, stopping only when a difference is found on a page.
14. A **report** can be created from the result, showing the "Compare documents" view and the two PDFS.

Example of a Compare PDF report



Log comparison based information

You can log all comparison-related information. How to do that will be explained in the next article.

27.9 Log comparison based information

The feature "Compare documents" in pdfToolbox comes with a resolution setting, among other parameters, to compare 2 PDFs.

This article explains how to log all comparison related information.

How to log information

For comparison related information to be logged, a '**Compare.log**' file has to be present in the user preferences. Simply executing the compare functionality would log all comparison parameters in the log file.

What's inside 'Compare.log'

Compare.log is a tab-delimited file containing the following information (a sample is attached at the bottom of this article)

```
timestamp        doc1        page1        resolution1        rect1 [left,bot-
tom,right,top]        doc2        page        resolution1        rect2 [left,bot-
tom,right,top]        algorithm        anchor        anchorBox        thresh-
old        areaThreshold        diff.min        diff.max        area
```

Click to copy

Please note that the headers should be tab-separated.

Where to add Compare.log

The log file has to be stored in the User Preferences like shown below:

Server/CLI

For logging compare on CLI, place the 'Compare.log' here.

Desktop

For logging compare on Desktop, place the 'Compare.log' here.

Example

A simple compare command like below

```
pdfToolbox --compare <PDF file 1> <PDF file 2>
```

Click to copy

will log the information to 'Compare.log', if 'Compare.log' is present under User Preferences, like shown here:

```
timestamp        doc1        page1        resolution1        rect1 [left,bot-
tom,right,top]        doc2        page        resolution1        rect2 [left,bot-
tom,right,top]        algorithm        anchor        anchorBox        thresh-
old        areaThreshold        diff.min        diff.max
area
20211102_134920        PDF file 1.pdf        0        72
[0,420,578,0]        PDF file 2.pdf        0        72        [0,420,578,0]
RGB        TopLeft        CropBox        0        0        0        0        1.
96078        10.6706
```

Click to copy

Sample

Simple download the attached 'Compare.log' and copy-paste it in your Preferences.

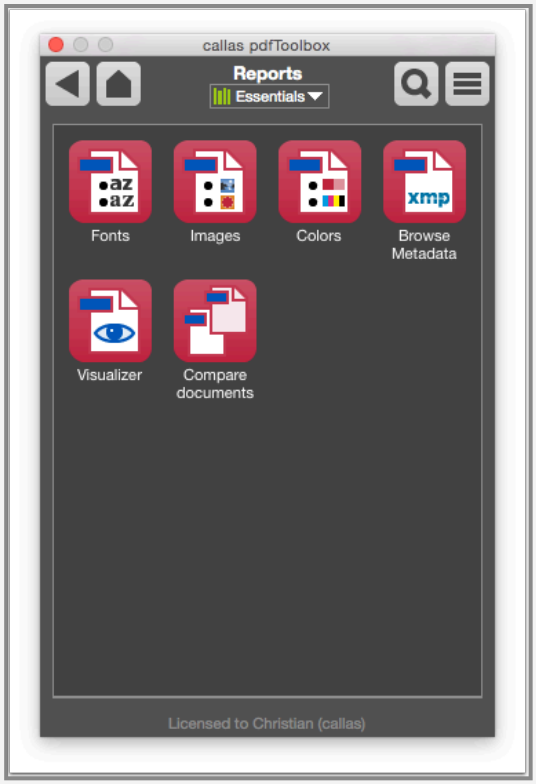
Compare.log

27.10 Explore Metadata

Select “Browse metadata”

In the “Reports” category, you will find a button marked “Browse Metadata” which opens a new dialog box providing a complete overview of all metadata contained within the current document.

This can be opened either using the “Metadata...” menu item in the Acrobat “File” menu, or using the “Explore Metadata” menu item in the plugin or the standalone version of the software.



Metadata Explorer

As well as the document's XMP metadata, the Metadata Explorer also shows metadata for individual page objects. It lists all objects, shows a small preview and indicates their position on the page.



Export Metadata

Metadata can be exported in the form of a configurable XML report. This function is available in the Options menu to the upper right.

27.11 Explore Layers

As Adobe Acrobat, pdfToolbox uses the term "Layers" to refer to what is technically called "Optional Content Groups" in the PDF specification. The rest of this article will use the term layers. The Layer Explorer lets you create and manage layers as well as explore layers already present in the PDF document.

Where to find the “Layer Explorer” in pdfToolbox Desktop

The Layer Explorer provides a complete overview of all layers contained within the current document. To open the Layer Explorer you have different options:

1. Click on the "Explore layers" icon in the right menu bar.
2. Use the Tools menu and click on "Explore layers".
3. Or open the "Layers" group in the Switchboard and click on the "Browse Layer" Action, which will also open the Layer Explorer.



The Layer Explorer



1. The "Layer views" section allows you to define and manage different layer views. A layer view determines the visibility of layers, allowing you to create predefined sets of layer visibilities.
2. The "Settings menu" provides options to create, edit, delete, and save layer views. To add a new layer view:
    1. Adjust the visibility of layers in the Layers List (3).
    2. Click the + button to save the current visibility settings as a new predefined view.

3. The lower section of the Layer Explorer displays all layers present in the current document. These layers correspond to Optional Content Groups (OCG) in PDF syntax. You can toggle their visibility by clicking the eye icon next to each layer's name.
    - Use the following shortcut to toggle layer visibility: Alt (Win) / Option (macOS) + click on the eye icon → Hides all layers except the selected one.
    - If Processing Steps information is available for a layer, it will be listed after the name of the layer. More de-

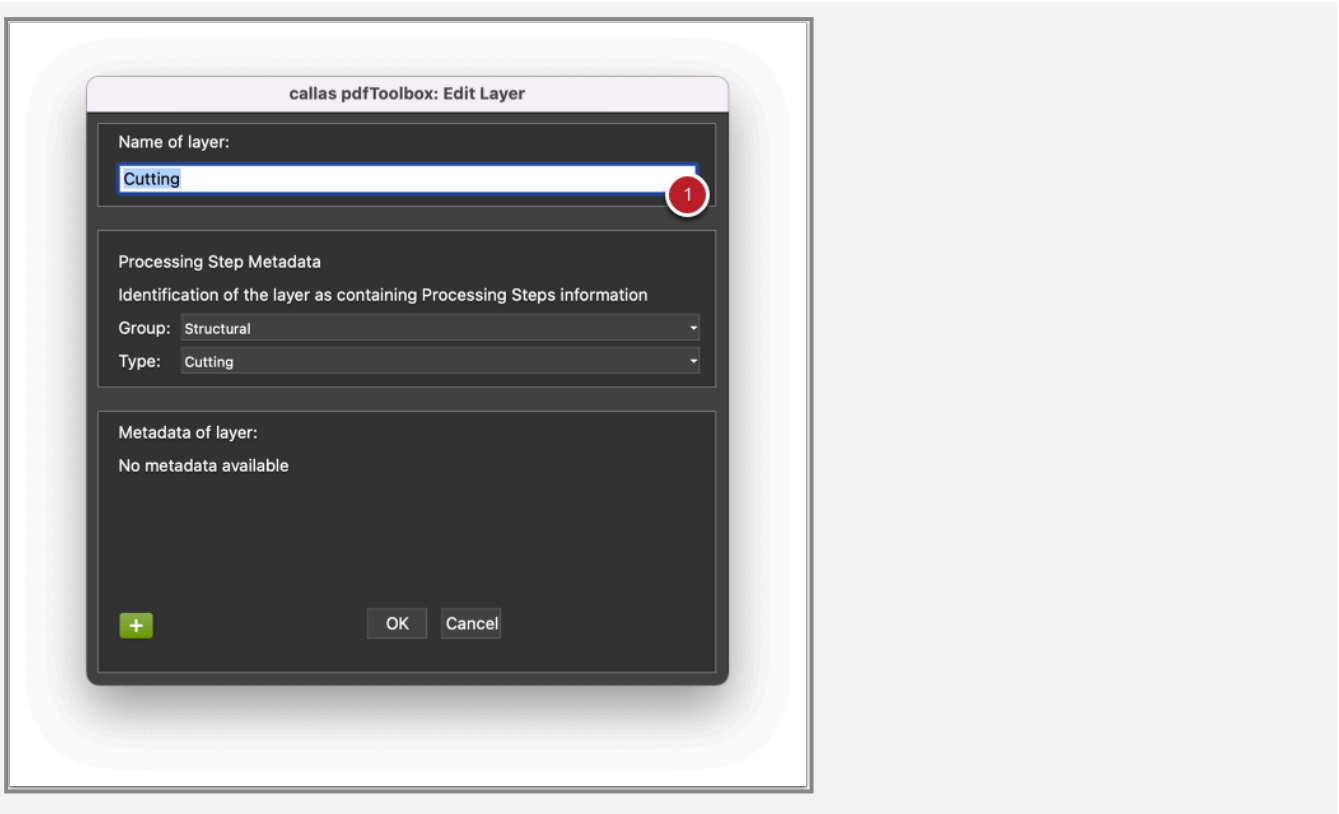
tails can be found in the article: Working with processing steps metadata for a layer.

4. The "Actions menu" allows you to:
   - Create, rename, and delete layers.
   - Use the "Add missing layer" Action (introduced in **pdfToolbox 16**) to assign objects without a layer to a new or existing layer.
5. The "Save Visibility" option allows you to set specific layers as initially invisible. To modify initial visibility:
   1. Turn off the layers that should be invisible.
   2. Click the "Save Visibility" button at the bottom of the Layer Explorer.

If the current PDF file has no layers, the Layer Explorer will display options for creating new layers using pdfToolbox.

Edit layers

You can easily edit individual layers or Processing Step metadata using the pencil icon.



27.12 Explore PDF

The "Explore PDF" functionality gives you an insight into the internal structure of PDF files.
Usually, this is not needed for common use, but might be helpful if you encounter a damaged file or if you are simply interested in learning more about the internal structure of a PDF file. The entry "Explore PDF..." can be found in the "Plug-Ins" menu of Acrobat ("Miscellaneous") or the "Tools" menu in the Standalone version.

With the "Explore PDF" tool you can have a look into the data structure of a PDF file, exposing the several commands and details that are forming the page objects.

Document Structure view

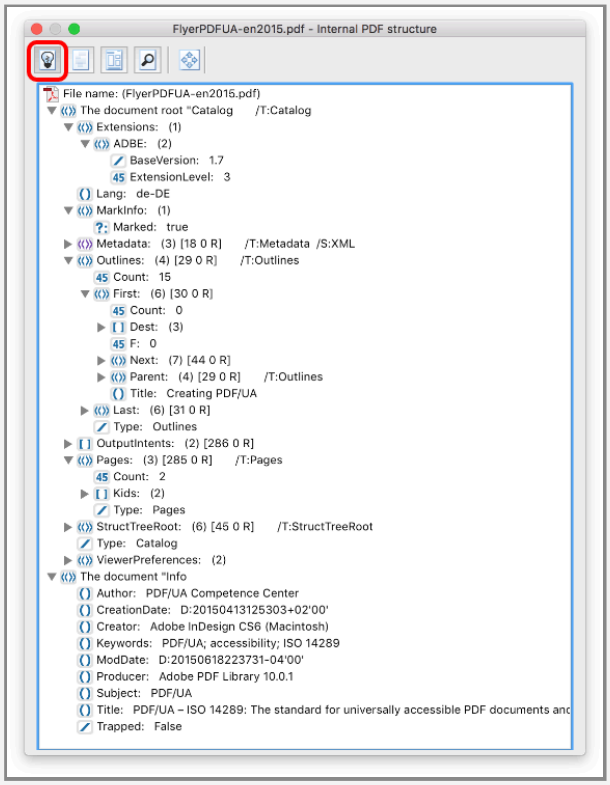
The first button opens the "Document Structure" view, which lists 2 items:

- The document root "Catalog"

The root of a document's object hierarchy is the "Catalog" dictionary. The catalog contains references to other objects, defining the document's contents, outlines, article threads, named destinations and other attributes. In addition, it contains information about how the document shall be displayed on screen, such as its outline and thumbnail page images shall be displayed automatically and whether some location other than the first page shall be shown when the document is opened.

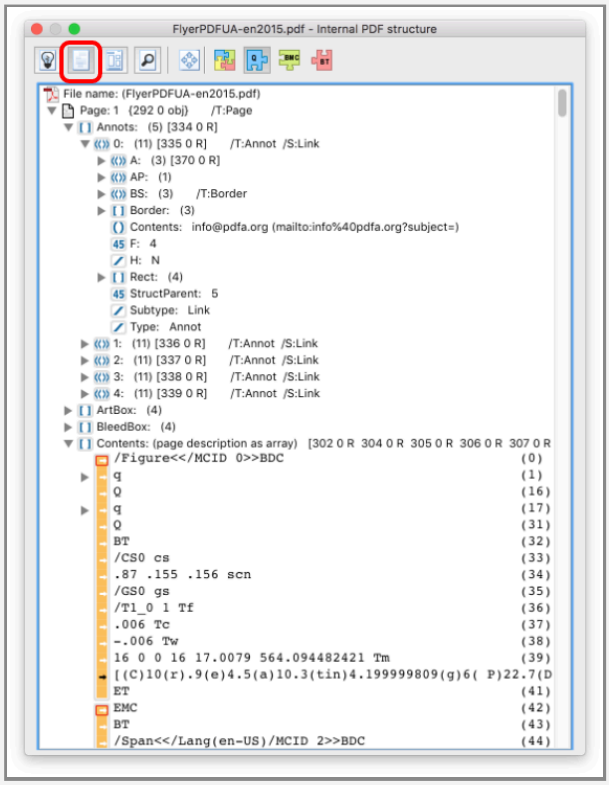
- The document info

The document info area lists some basic information about the file, like title, author, creation date, producer, creator, keywords and so on



Logical Structure view

While the "Document Structure" view contains the complete view of the documents content, the "Logical Structure" view offers a page-by-page view of the different properties of a page like page geometry boxes, used resources, content stream and more as well as other optional attributes like annotations or thumbnails.

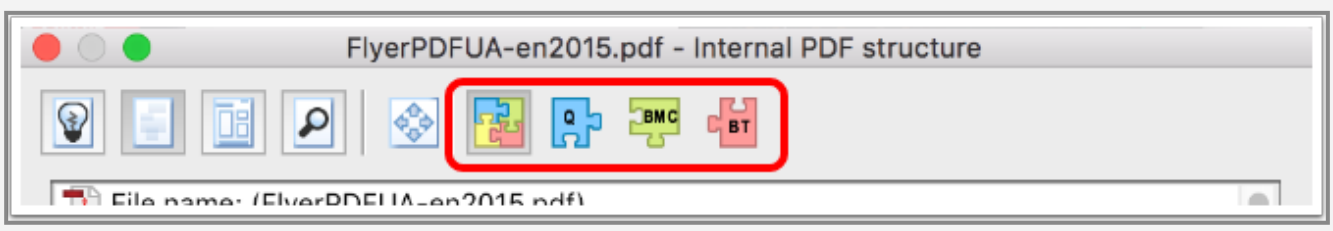


Different views of the content stream

The "Logical Structure" view offers 4 different representations of the content stream:

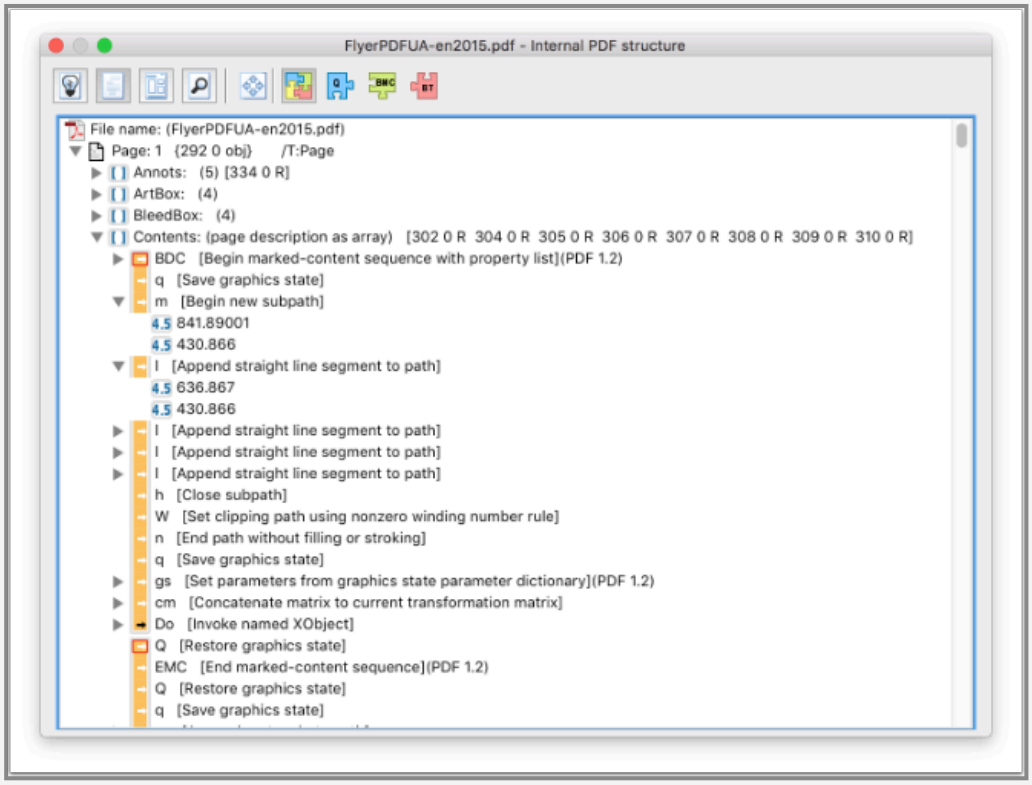
- Content Stream snippets: explained
- Content Stream snippets: q/Q pairs
- Content Stream snippets: Marked content
- Content Stream snippets: text

These views can be selected using the 4 colored buttons on the right of the selection bar:



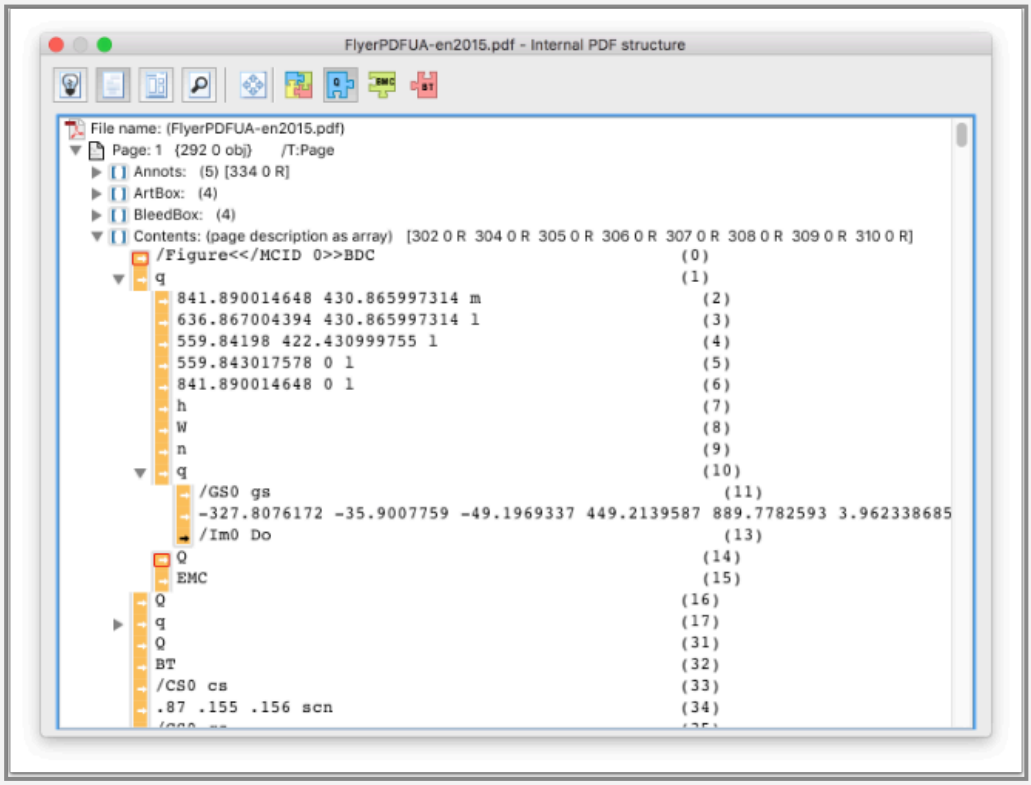
Content Stream snippets: explained

This mode shows minute explanations for all painting sequences of the content stream and makes it easy to understand the way the content of a page is composed step by step.



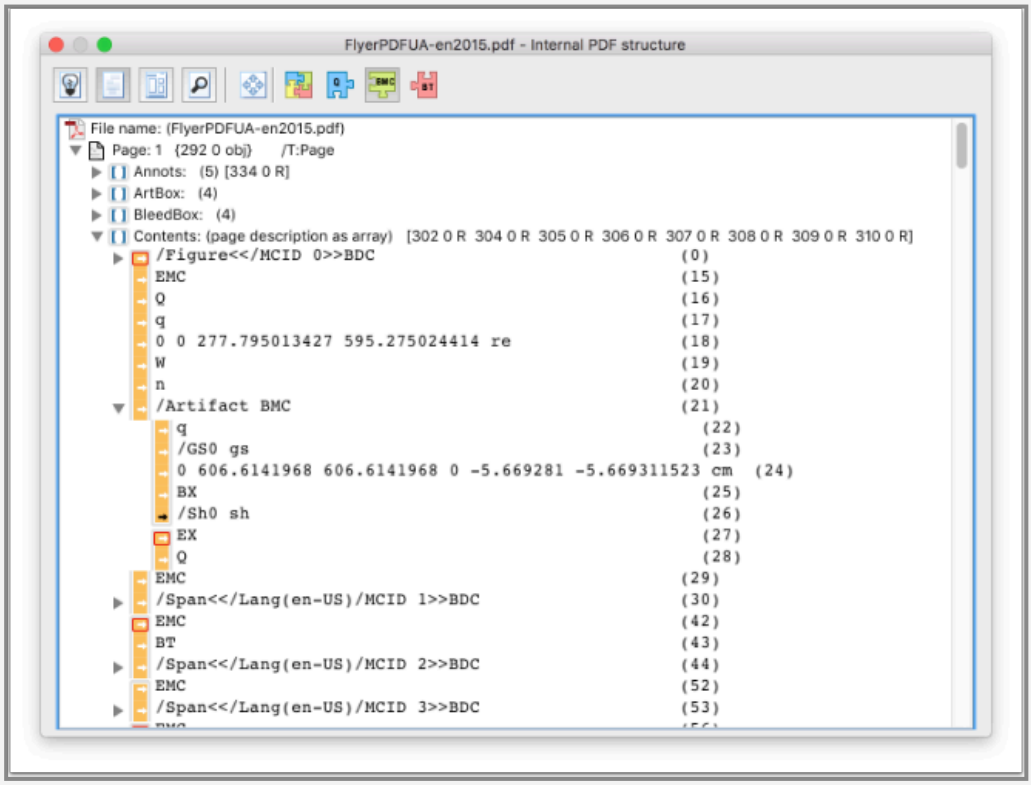
Content Stream snippets: q/Q pairs

The "q/Q pairs" view shows the content stream in a more condensed way, as the painting sequences are sorted in their respective graphic state nesting, also known as q/Q pairs.



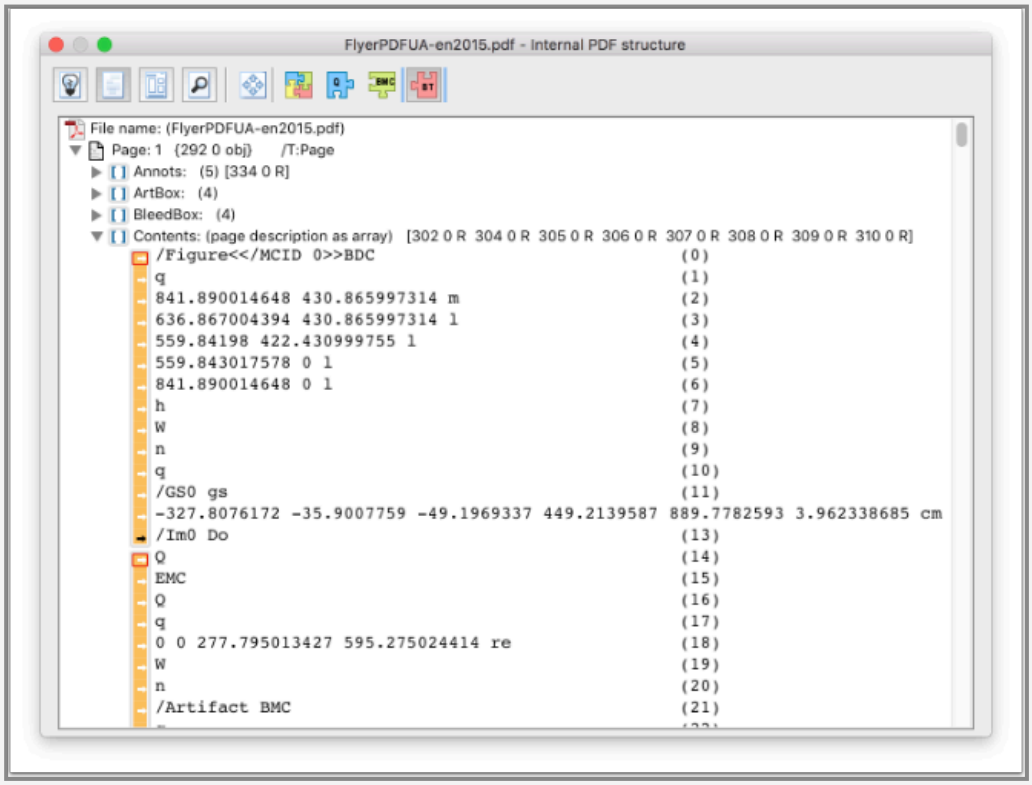
Content Stream snippets: Marked content

The "Marked content" view shows the content stream from the tagging or marked content perspective, as the content stream is grouped by the respective BMC or BDC properties.



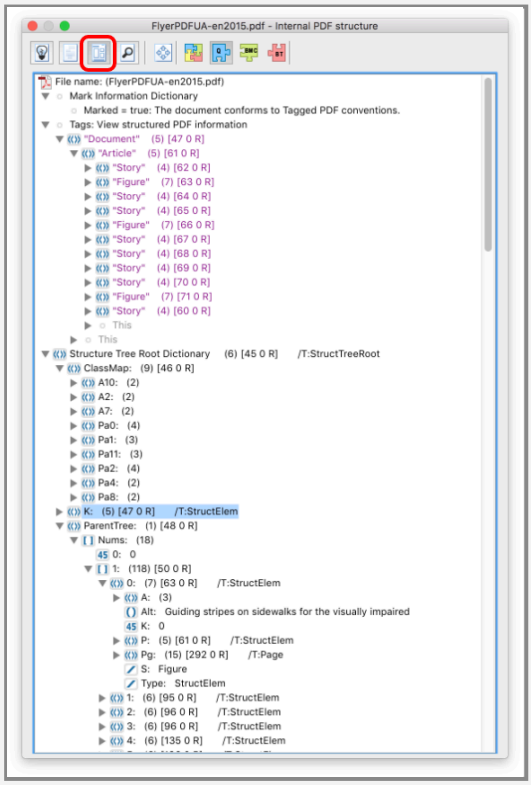
Content Stream snippets: text

The fourth view offers a plain text view of the content stream in a readable fashion.



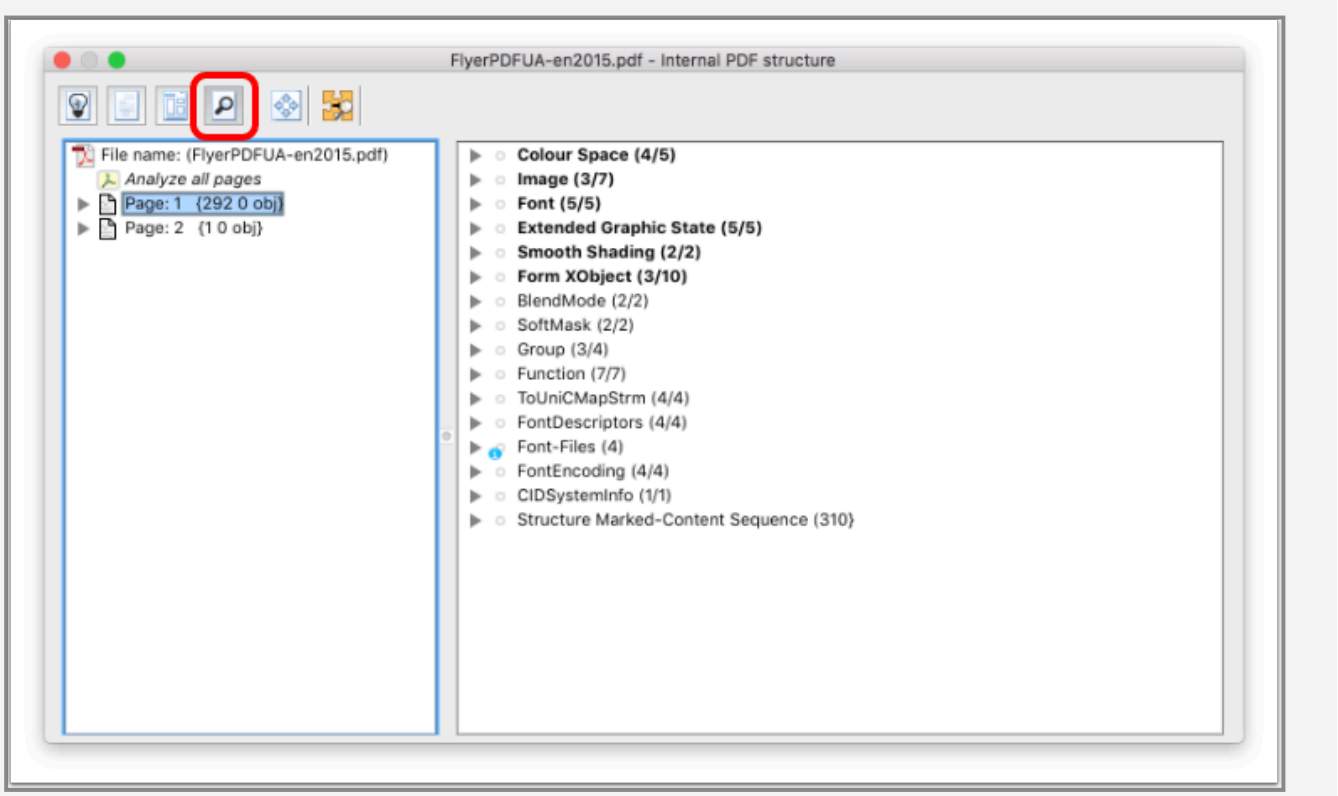
Tagging Stucture view

When a PDF file contains tagging information, this view gives a detailed overview about all the details like the ClassMap, a structural view of the various tags as well as other interesting details.



Resource view

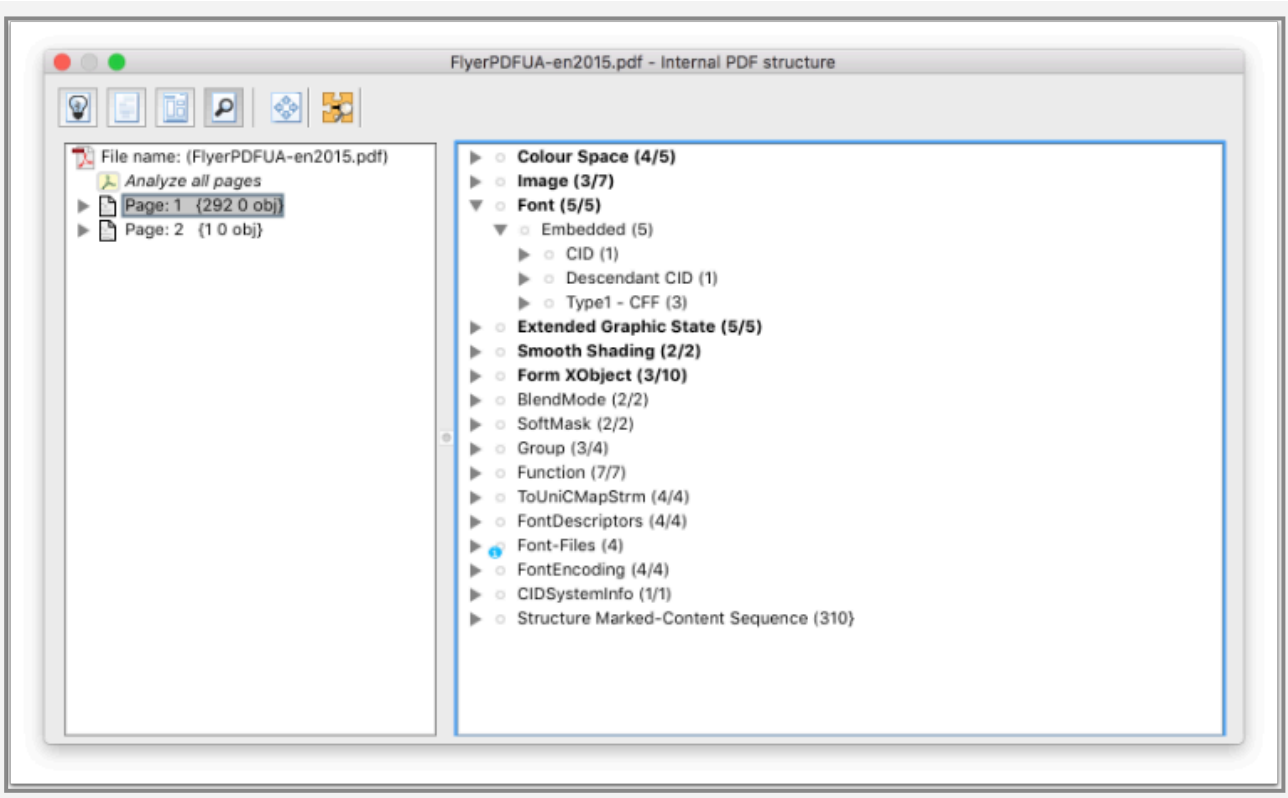
While the Logical view gives you (amongst others) a view into the content stream, the "Resource" view offers a page-by-page listing of all painting objects (like images, shadings, text, vector objects, ...).



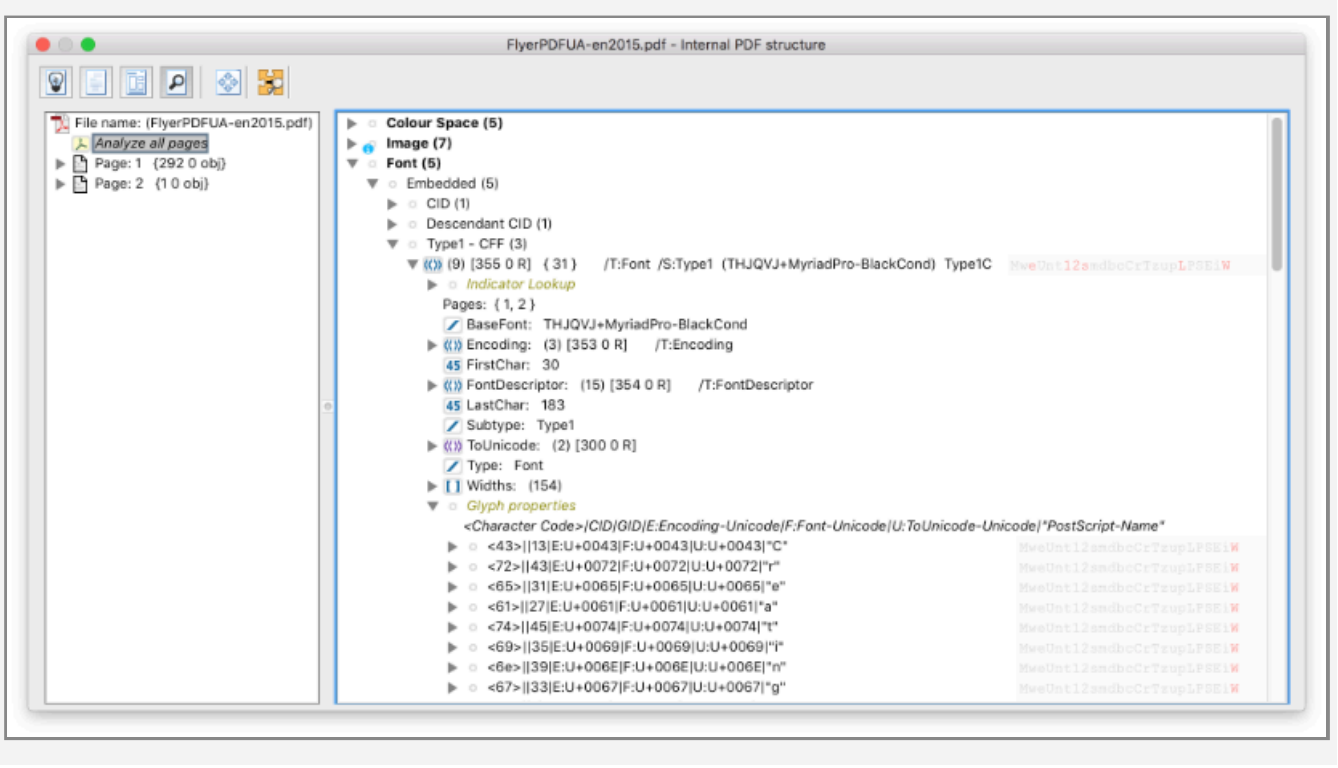
Each of the resources has a specific substructure with further information. Also all resources are listed that are used on the selected page, independent from whether they are specified in the page resource or in Form XObject resources.

The "Font" section is specifically rich with many results from pdfToolbox' font engine.

Detailed glyph information for embedded fonts



The font section shows embedded and not embedded fonts and then font types. The results of the font engine are available for the whole font and for each glyph by a list of indicators behind the respective entries.



When selecting a font, a lot of detailed information about the font file itself and the contained glyphs is available. Depending on if the selection on the left pane is on a specific page or on "Analyze all pages", the fonts used on that page or in the whole document are listed.

For all glyphs of an embedded font, there are several indicators behind each glyph. If such an indicator is red, this means that the corresponding property of the indicator applies to the glyph. This does not have to be a problem right away, it can help to make the different properties of the glyphs quickly accessible.

In this example for almost all glyphs a capital "W" and for some glyphs, also "e" and "s" are active - as indicated by the indicator being red. The section "Indicator lookup" explains the indicators.

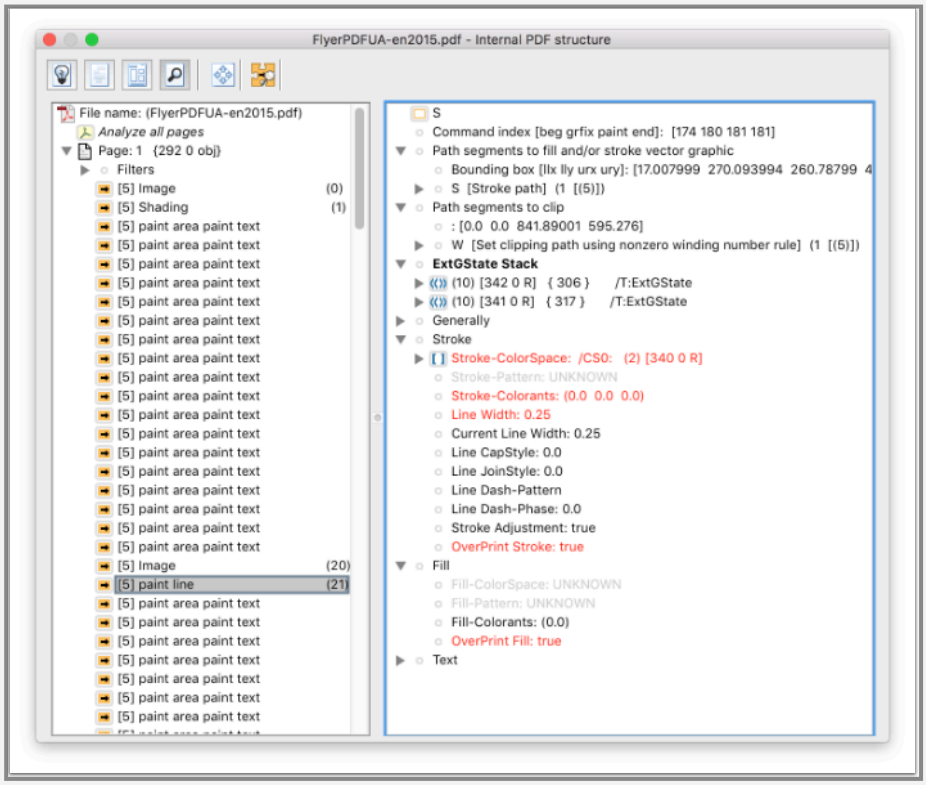


A list of the available indicators can be found at "Indicator lookup" entry. Please note, that the order of the indicators has to be considered.

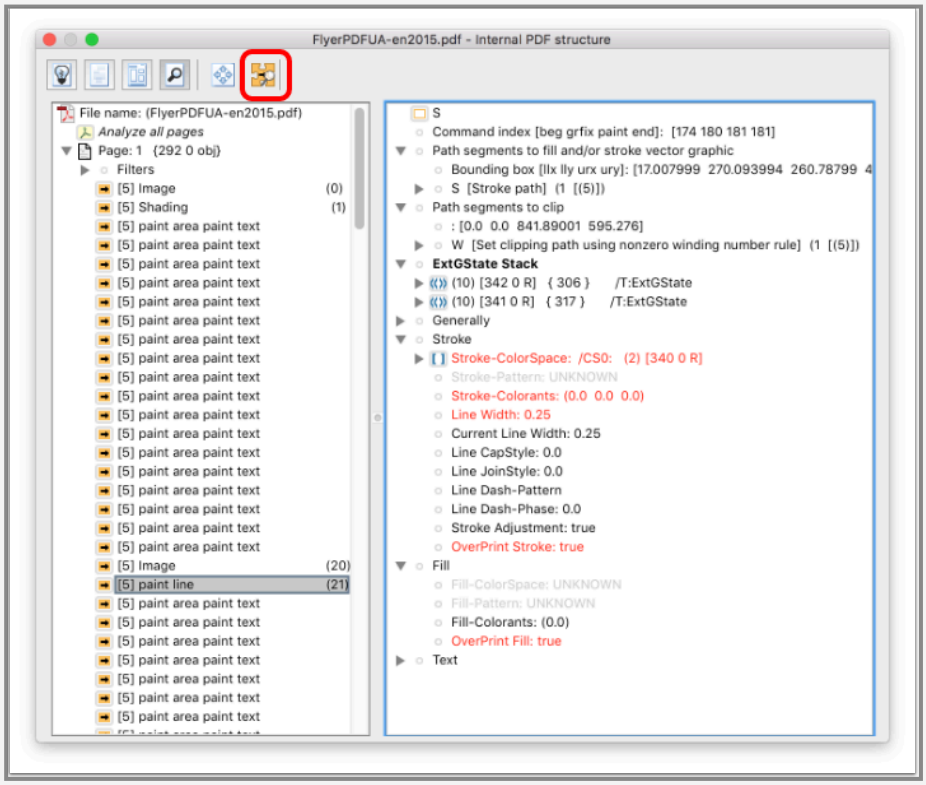
The indicator lookup informs us that the capital "W" means that the glyph width is used for positioning (the glyph is not positioned using coordinates but the width of a previous glyph). "e" stands for glyphs without contour and "s" is for such empty glyphs with a width, so in fact the respective glyph is a whitespace.

Analyze snippets and export selected snippets to a new PDF

For each object, a detailed view shows the painting and clipping area, the used color space as well as a lot of other information of the current Extend GraphicState, a used transformation matrix, for text the used font and font size, blend spaces, overprint modes and much much more.



The "open snippet as PDF" icon at the top of the Resource view allows for creating a PDF from the current selection in the left pane. The selection might also include several objects and Filters are provided to select all object of a certain type. Such PDF parts can be used for analysis to simplify a PDF.



The new PDF will be opened separately and can be used for further investigation of the PDF.
You'll see the functionality indicated by the red rectangle in the screen shot above.

PDF sample file used

The attached file has been used to show the various views of "Explore PDF".
This file has been created by the PDF Association.

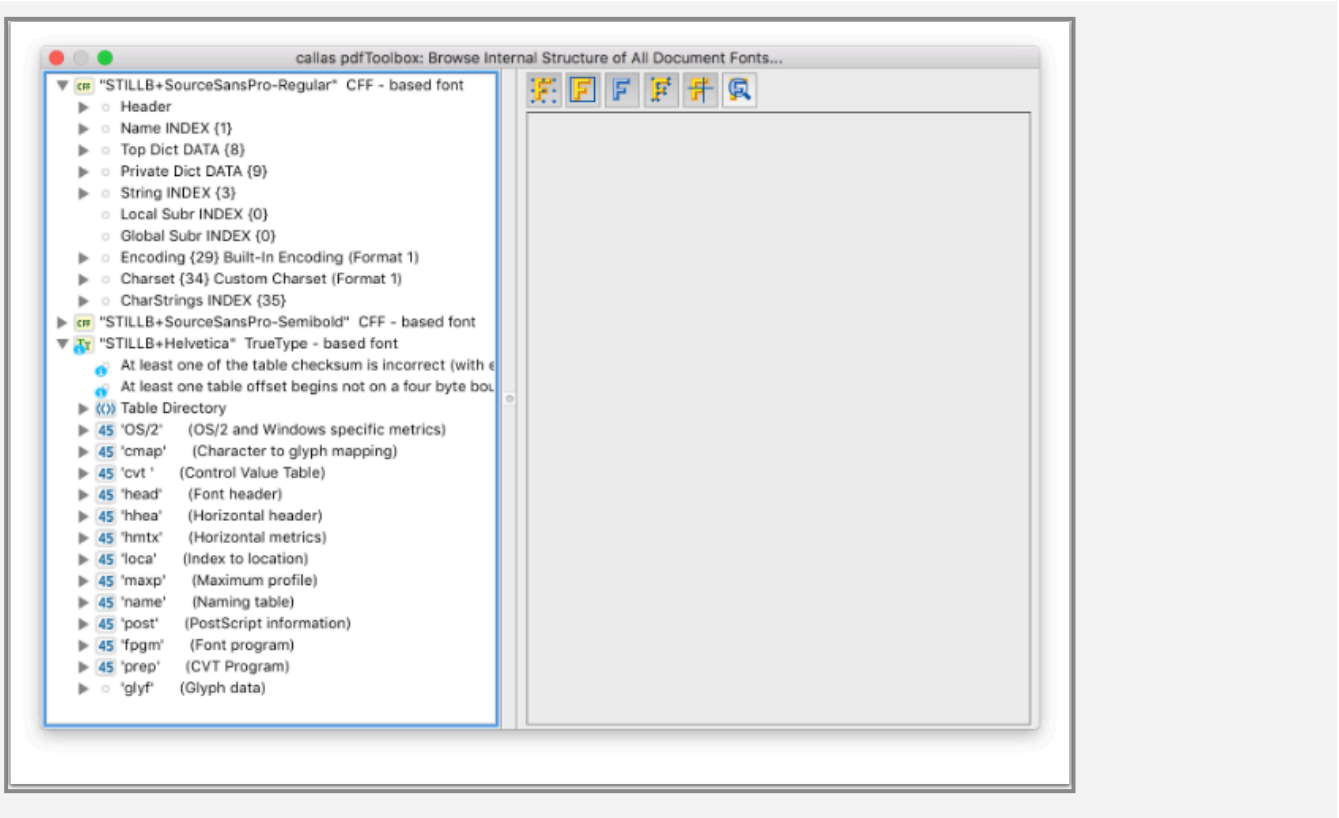
FlyerPDFUA-en2015.pdf

27.13 Explore Fonts

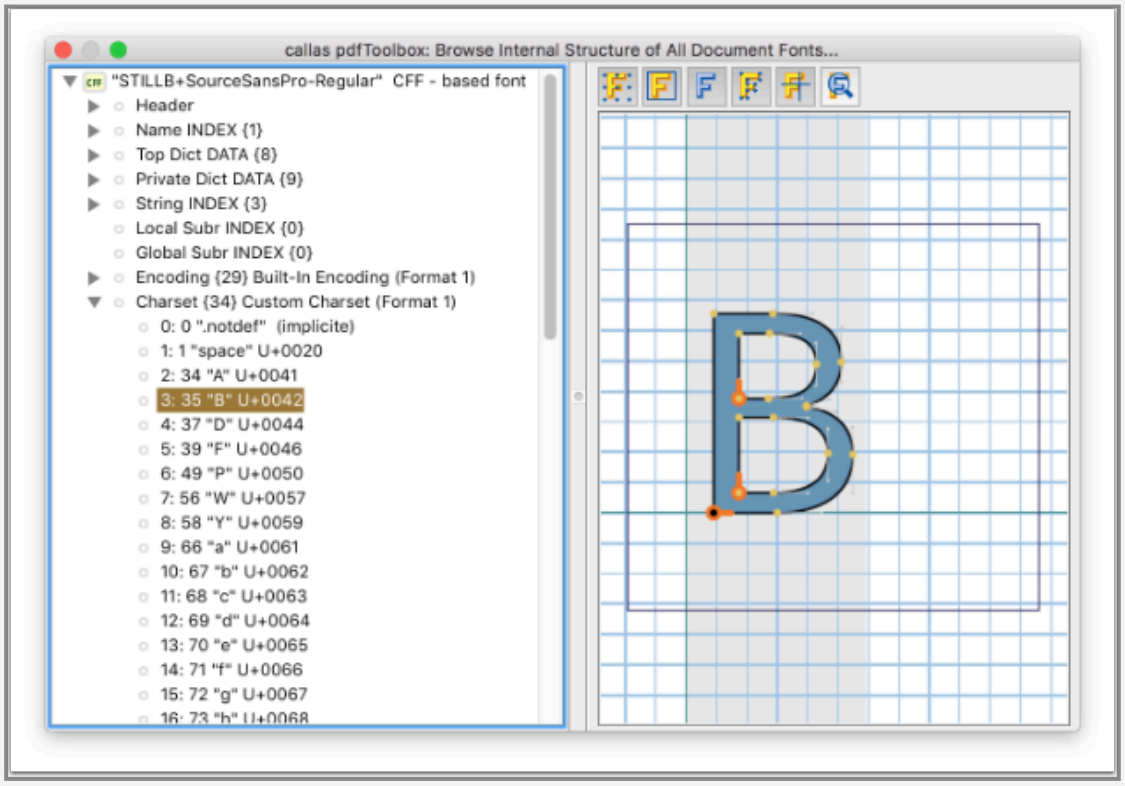
The "Explore Fonts" function gives you an insight into the internal font structure of PDF files. This is usually not needed for common use, but might be helpful if you encounter a damaged file or if you simply are interested in learning more about the internal font structure. The entry "Explore Fonts..." can be found inside the "Plug-Ins" menu of Acrobat ("Miscellaneous") or the "Tools" menu in the Standalone version.

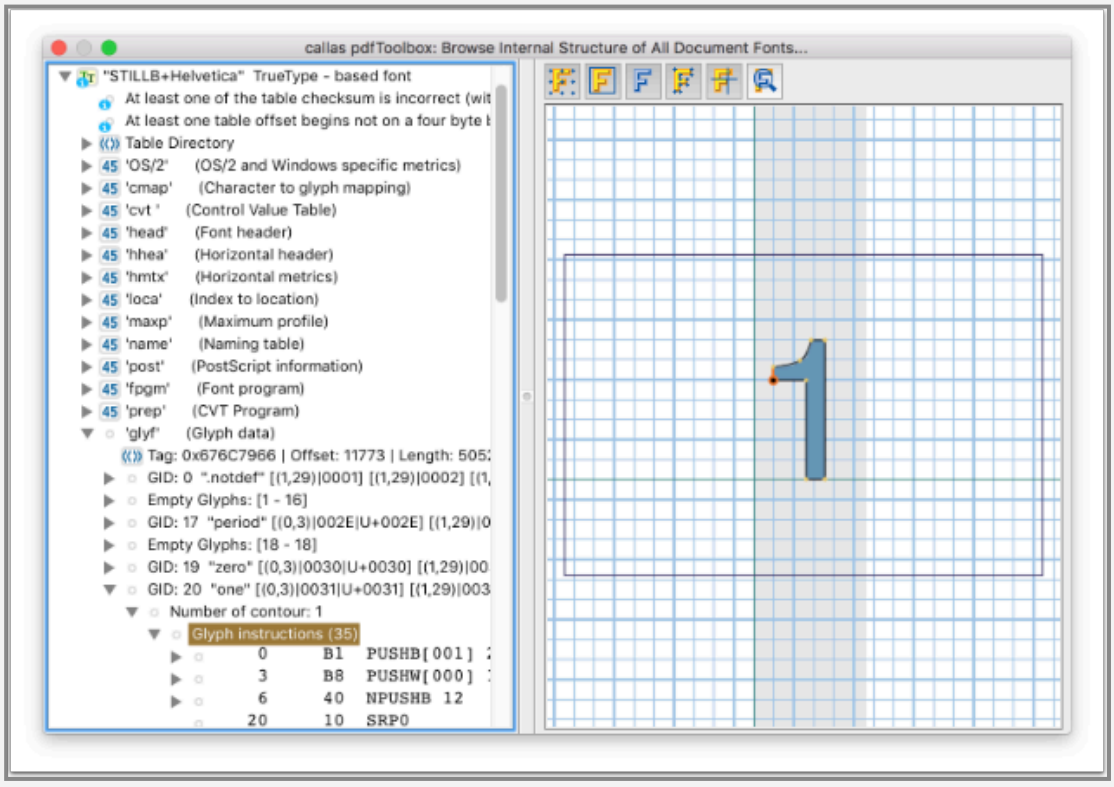
In the "Explore Fonts" dialog you will find information about e.g. font type and embedding state of all fonts present in the current document.

Depending on the type of font (e.g. TrueType, Type1, ...), the way how information of the font is shown is different:



Also, painting information about the contained glyphs of embedded fonts will be displayed:


callas pdfToolbox: Browse Internal Structure of All Document Fonts...
"STILLB+SourceSansPro-Regular" CFF - based font
Header
Name INDEX {1}
Top Dict DATA {8}
Private Dict DATA {9}
String INDEX {3}
Local Subr INDEX {0}
Global Subr INDEX {0}
Encoding {29} Built-In Encoding (Format 1)
Charset {34} Custom Charset (Format 1)
0: 0 ".notdef" (implicite)
1: 1 "space" U+0020
2: 34 "A" U+0041
3: 35 "B" U+0042
4: 37 "D" U+0044
5: 39 "F" U+0046
6: 49 "P" U+0050
7: 56 "W" U+0057
8: 58 "Y" U+0059
9: 66 "a" U+0061
10: 67 "b" U+0062
11: 68 "c" U+0063
12: 69 "d" U+0064
13: 70 "e" U+0065
14: 71 "f" U+0066
15: 72 "g" U+0067
16: 73 "h" U+0068
B


callas pdfToolbox: Browse Internal Structure of All Document Fonts...
"STILLB+Helvetica" TrueType - based font
At least one of the table checksum is incorrect (wit
At least one table offset begins not on a four byte b
Table Directory
'OS/2' (OS/2 and Windows specific metrics)
'cmap' (Character to glyph mapping)
'cvt ' (Control Value Table)
'head' (Font header)
'hhea' (Horizontal header)
'hmtx' (Horizontal metrics)
'loca' (Index to location)
'maxp' (Maximum profile)
'name' (Naming table)
'post' (PostScript information)
'fpgm' (Font program)
'prep' (CVT Program)
'glyf' (Glyph data)
Tag: 0x676C7966 | Offset: 11773 | Length: 505
GID: 0 ".notdef" [(1,29)|0001] [(1,29)|0002] [(1,
Empty Glyphs: [1 - 16]
GID: 17 "period" [(0,3)|002E|U+002E] [(1,29)|0
Empty Glyphs: [18 - 18]
GID: 19 "zero" [(0,3)|0030|U+0030] [(1,29)|00
GID: 20 "one" [(0,3)|0031|U+0031] [(1,29)|003
Number of contour: 1
Glyph instructions (35)
0 B1 PUSHB[001]
3 B8 PUSHW[000]
6 40 NPUSHB 12
20 10 SRP0
1

27.14 Explore tagging

For the files that are structured on tags (PDF/UA files), the user is able to receive a comprehensive overview of the structure of the elements in a web browser.

Tagged PDF file

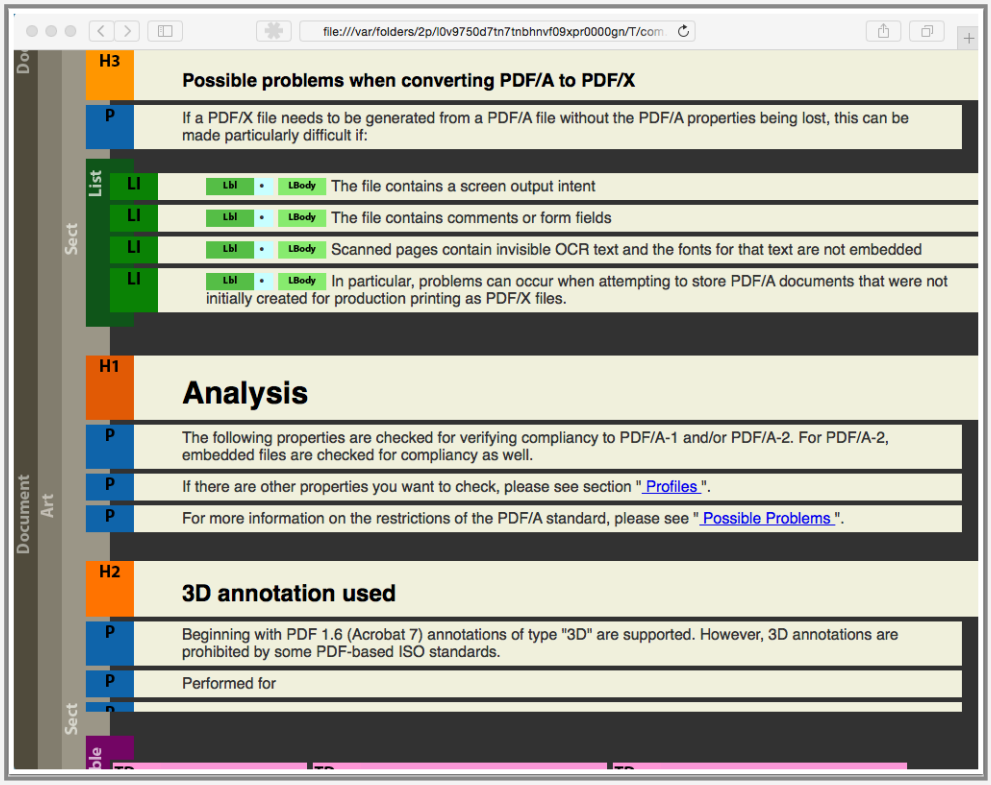


pdfToolbox shows immediately when a PDF file has labels indicated by a purple "tree" icon at the bottom. You can:

1. Click on the purple "tree" icon.
2. Go to Tools --- Explore Tagging
3. pdfToolbox Standalone Home window --- Go to Free tools --- Explore Tagging

- A comprehensive structural overview opens in the operating system default browser.

Inspect the PDF structural overview



From each PDF file the user receives a detailed overview of the PDF structure.

In the overview you have areas like "Document", "Article" and "Section". In the Section you have categories indicated by color codes such as Heading 1, 2, ..., paragraph lists and tables.

27.15 XMP Metadata reports

Configuration

The configuration methods described here apply to the XMP metadata report created with the "Browse Metadata"/"Extract XMP metadata" action.

The configuration file enables the user to set up named filters for XMP metadata and other meta information (e.g. DocInfo Dictionary entries, or information retrieved from the PDF document directly such as image resolution).

The XMP metadata report will be created as an XML report according to the chosen config file.

File format

A report configuration file must be stored as a tab delimited UTF-8 encoded cfg file.

Constraints

For each prefix used for **Include**, **Exclude** or **GroupingKey** in the config file a matching namespace entry defining the namespace URI for this prefix must be contained in the file.

Example Configuration

You will find example configuration files in your CLI installation directory in `"/var/Actions/Metadata/Filters/Export"` for the following sections:

- DublinCore
- EXIF
- General
- IPTC
- Photo
- PLUS
- Workflow

Configuration Keys

The configuration file can consist of the following elements (Keys):

DisplayName

A single entry that specifies the display name for this configuration. This key must exactly be defined once.

DictFlag	0 = Name shall be interpreted as string, 1 = Name shall be interpreted as dict key
Title	if DictFlag = 1, Dict key for display name lookup (does only apply for implemented properties) if DictFlag = 0, the display name

Example

DisplayName	0	Genera
DisplayName	1	BOAGU

Namespace

Defines a namespace URI for usage in the XMP metadata report and associates it with a namespace prefix and a schema name. This key is optional and can be used multiple times.

Prefix	The namespace prefix that is pre-ferred for the namespace URI, e.g. "dc"
Namespace URI	The namespace URI, e.g. "http://purl.org/dc/elements/1.1/"
DictFlag	0 = Schema shall be interpreted as string, 1 = Schema shall be interpreted as dict key
Schema	The display name for the sche-ma, that is associated with the namespace, e.g. "Dublin Core"

Example

```
Namespace dc http://purl.org/dc/elements/1.1/ 0 Dublin Core
```

Property

Defines a namespace property for usage in the XMP metadata report and associates it with a namespace prefix and a prop-

erty name. This key is optional and can be used multiple times.

Prefix	The name-space prefix, e.g. "ptb_image"
Name	The proper-ties name, e.g. "px_width"
DictFlag	0 = Label shall be in-terpreted as string, 1 = Label shall be in-terpreted as dict key
Label	Display name for the property, e.g. "Image width in pix-els"

Example

```
Property ptb_document file 0 File Name
```

GroupingKey

Defines the grouping of XML reports. For each value type of the property in the namespace that is associated with the prefix, a distinct XMP metadata report will be generated con-taining only elements defined by the "Include" clause, that have the same value for prefix/property. This key is optional but must not be used more than once.

Type	Specifies for which objects in the PDF document the grouping key shall be searched/ applied Possible values are Document, Page, Image
Prefix	The namespace prefix Must be included in the namespace definition
Property	The XMP property in the namespace identified by prefix to be used for report grouping

Example

```
GroupingKey Image xmpRights Owner
```

Include

Whitelist. All metadata that matches any entry in this list and is not excluded by the Exclude statement will be exported.

This key should at least be used once (in order to create a report at all) but can be used multiple times.

Type	Specifies for which objects in the PDF document this include state-ment will be applied Possible values are Document, Page, Image, * Note: * is used as a wildcard and matches all types
Prefix	The namespace prefix Must be included in the namespace definition Note: * is used as a wildcard and matches all prefixes
Property	The XMP property in the namespace identified by prefix to be used for matching Note: * is used as a wildcard and matches all properties

Example

```
Include Image ptb_image thumbnail
```

Exclude

Blacklist. All metadata that matches any entry in this list will not be exported. This key is optional and can be used multiple times.

Type	Specifies for which objects in the PDF document this include statement will be applied Possible values are Document, Page, Image, * Note: * is used as a wildcard and matches all types
Prefix	The namespace prefix Must be included in the namespace definition Note: * is used as a wildcard and matches all prefixes
Property	The XMP property in the namespace identified by prefix to be used for matching Note: * is used as a wildcard and matches all properties

Example

```
Exclude Document dc title
```

Order of filtering

Filtering will be executed in the following order:

GroupingKey not present

Include all items that

- match at least one entry in the white list (`Include` key)
- and match no entry in the black list (`Exclude` key)

GroupingKey present

For each value of the XMP metadata property as defined in the GroupingKey a separate report will be created which includes all items that

- have an XMP metadata property as defined in the GroupingKey (e.g. "Image") which has a value as defined in the GroupingKey (e.g. "xmpRights")
- and match at least one entry in the white list (`Include` key)
- and match no entry in the black list (`Exclude` key)

Handling of non-XMP metadata

There are some namespaces and properties additionally defined by pdfaPilot and pdfToolbox.

DocInfo Dictionary

Namespace URI	http://www.callassoftware.com/ns/pdfaPilot2/1.0/metadatareport/docu-ment
Namespace URI	http://www.callassoftware.com/ns/pdfToolbox4/1.0/metadatareport/docu-ment
Preferred prefix	ptb_document

Available properties

DocInfo_<key>	Document info entry <key>

<key> has to be one of the following PDF document info dictionary keys:

CreationDate	The date when the PDF document was created
ModDate	The date when the PDF document was last modified
Creator	The application the original docu-ment was created with
Producer	The application the PDF was pro-duced with
Title	The document title
Subject	The subject of the document
Keywords	The keywords for the document
Trapped	The trapped key
PageMode	The mode in which the doc-

	ument shall be displayed when opened (e.g. "UseOutlines")
PageLayout	The way the pages are displayed when opening (e.g. "SinglePage")
PdfXVersion	The PDF/X version (e.g. PDF/X-1a)
PdfXConformance	The PDF/X conformance level (e.g. 1a)
PdfE1Version	The PDF/E version

Example

```
Property ptb_document DocInfo_Creator 0 Creator
```

Image properties

Namespace URI	http://www.callassoftware.com/ns/pdfaPilot2/1.0/metadatareport/image
Namespace URI	http://www.callassoftware.com/ns/pdfToolbox4/1.0/metadatareport/image
Preferred prefix	ptb_image

Available properties

px_width	Image width in pixel
px_height	Image height in pixel
ppi_horizontal	Horizontal resolution in ppi
ppi_vertical	Vertical resolution in ppi
left	Offset of left image border in pt (relative to crop box)
right	Offset of right image border in pt (relative to crop box)
top	Offset of top image border in pt (relative to crop box)
bottom	Offset of bottom image border in pt (rela-

	tive to crop box)
pt_llx	Quad-Point Lower Left x in Pt
pt_lly	Quad-Point Lower Left y in Pt
pt_ulx	Quad-Point Upper Left x in Pt
pt_uly	Quad-Point Upper Left y in Pt
pt_urx	Quad-Point Upper Right x in Pt
pt_ury	Quad-Point Upper Right y in Pt
pt_lrx	Quad-Point Lower Right x in Pt
pt_lry	Quad-Point Lower Right y in Pt

pt_width	Image width in pt
pt_height	Image height in pt
thumbnail	Image thumbnail

Example

```
Property ptb_image px_height 0 Image height in pixel
```

Page properties

Namespace URI	http://www.callassoftware.com/ns/pdfaPilot2/1.0/metadatareport/page
Namespace URI	http://www.callassoftware.com/ns/pdfToolbox4/1.0/metadatareport/page
Preferred prefix	ptb_page

Available properties

nr	Page sequence number (’1’ basecd)
cropbox_width	Cropbox width
cropbox_height	Cropbox height

cropbox_left	Cropbox left
cropbox_right	Cropbox right
cropbox_top	Cropbox top
cropbox_bottom	Cropbox bottom
thumbnail	Page thumbnail

Example

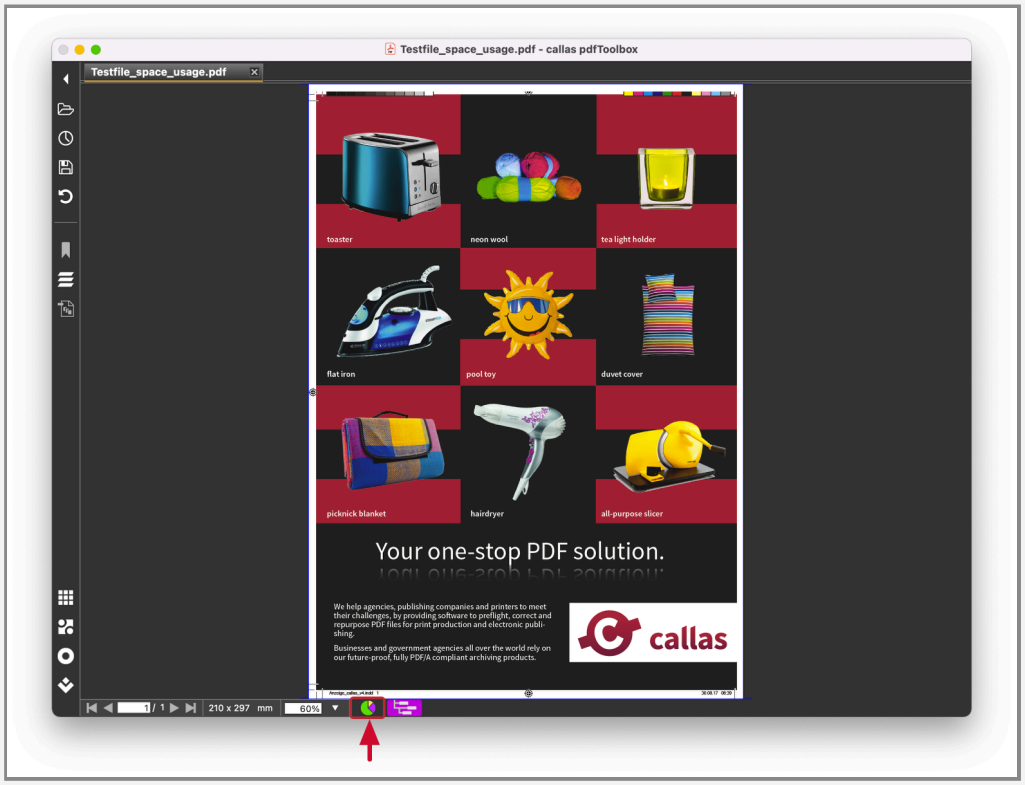
```
Property ptb_page cropbox_top 0 Cropbox top
```

27.16 Analyze the space usage of a PDF

The space usage is available starting with **pdfToolbox 16**

The **space usage** tool in pdfToolbox analyzes a PDF file in detail and shows exactly how much space each component of the PDF file takes up. By providing a precise byte-by-byte breakdown, it helps to identify which elements are consuming the most space.

To access the space usage in pdfToolbox, simply click on the pie chart icon in the lower menu bar in pdfToolbox Desktop. This will open the space usage dialog:



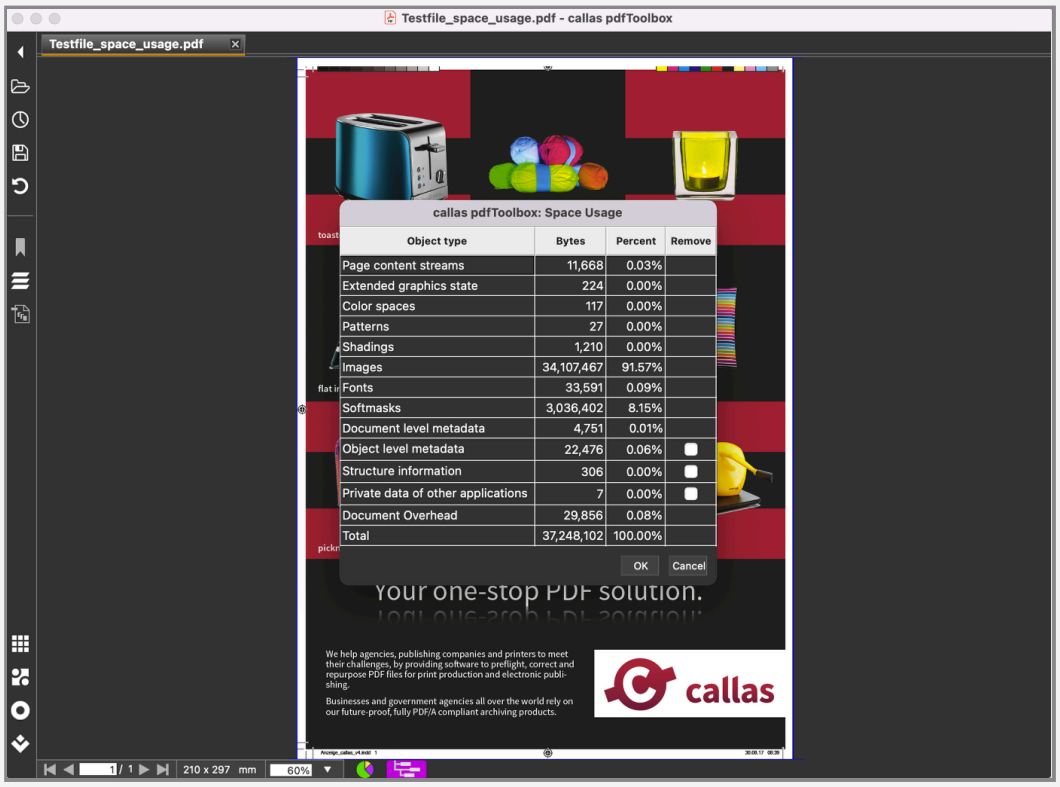
The space usage provides a detailed breakdown of the total number of bytes used for specific document elements. These elements are grouped into different categories such as:

- Page content streams

- Extended graphics state
- Color spaces
- Fonts
- Images
- Annotations
- Shadings
- And more...

The total space usage for each of the categories is provided based on the size of the objects. The sizes are reported in bytes and as a percentage, so you can see the proportional contribution of each category to the total file size.

The total file size is displayed at the bottom. This total is the sum of all categories.



In the example shown in the screenshot, the PDF file has a size of **37.2 MB (37,248,102 bytes)**. The space usage shows that 91.57% of the space is used by images. This indicates that the file contains high-resolution images that contribute significantly to its large file size.

Reducing the file size

In addition to analyzing space usage, it also allows you to reduce file size by removing data from the PDF. This can include

- Embedded files
- Annotations
- Form data
- Private data from other applications
- XFA forms
- Multimedia
- Object level metadata
- Structure information

To remove the data, select one or more checkboxes in the fourth column of the table and click OK to proceed. This will bring up a "Save as" dialog. When the PDF is saved, the selected data will be removed.



28. Quick Check

28.1 Quick Check – Introduction

Quick Check functionality in pdfToolbox makes it possible to retrieve certain information from a PDF very quickly.

Internal tests at callas software have shown that it typically takes 5 seconds to retrieve and return complete page size information for all pages in a 40.000 page PDF document.

Scope and limitations of Quick Check

For deep and comprehensive analysis of a PDF file, using a preflight profile is still the recommended approach.

Quick Check offers a much faster and more lightweight alternative for retrieving PDF-related information - but with certain limitations. In particular, Quick Check does not analyze the content stream (the actual page description), because doing so would compromise its speed. This means that any information defined within the content stream - such as color spaces - is not available through Quick Check.

If reliable and complete details about content-stream-based information (such as color spaces) are required, running the full preflight engine is mandatory.

For more basic data, or for information that is not defined in the content stream, Quick Check provides an extremely fast and lightweight mechanism for retrieving such information. It is highly configurable, making it possible to retrieve and return only the information that is actually needed.

Typical examples for Quick Check based information retrieval:

- number of pages
- page sizes
- color spaces used (e.g. whether RGB is used)
- spot colors used
- fonts used
- layers used

- whether the PDF claims conformance with PDF/X, PDF/A, PDF/UA, PDF/VT or PDF/E
- output intents
- information about bookmarks
- information about embedded files
- use of transparency (opacity, blend mode, soft masks)
- image resolution (or any other information pertaining to images)
- XMP metadata
- information about markup annotations or other annotations



If you feel a need for any aspects currently not available in Quick Check, please use the commenting feature at the end of this article for requesting their implementation in a future version.

How to use Quick Check

Quick Check is available in two ways:

- As a step in a Process Plan:
  This makes it possible to first quickly retrieve some core data for a PDF about to be processed, and then use that data to make decisions – via a JavaScript step or JavaScript inside other Process Plan steps – how to process the PDF; e.g. if there is no non-CMYK color data present, there is no need to go through a "Convert to CMYK" Fixup
- As a call to on the command line (requires the Server or command line version of pdfToolbox):
  In a special Quick Check execution mode, only a small part of the pdfToolbox command line executable is launched, cutting down on launch time and use of resources; core data retrieved can be used by the system that drives pdfToolbox, for example to deliver immediate information on incoming PDF files or to decide about further processing.

Configuring Quick Check

Quick Check is configured by a list of filter expressions that white lists or black lists different sets and subsets of available data. For a Quick Check step in a Process Plan, the filter expressions have to be provided as an array through a JavaScript variable. On the command line, a simple text based configuration file with one filter expression per line is used.
A more detailed article about Quick Check configuration is found here.

Quick Check output

For a Quick Check step in a Process Plan, the result from running a Quick Check is added to the `app.vars` data object (which is data structure in the form of a JavaScript variable which is maintained and enriched throughout the execution of a Process Plan). Subsequent steps in a Process Plan can retrieve data from the `app.vars` data object and use it for processing.

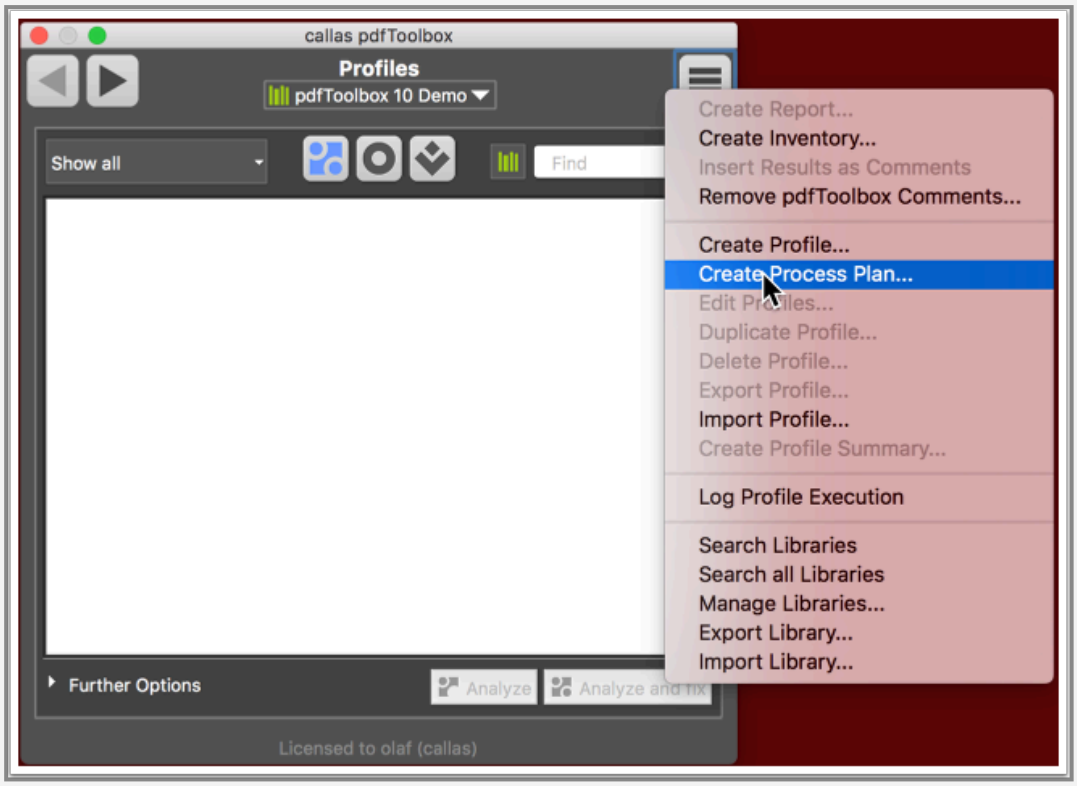
For Quick Check execution in the command line, the output is created in the form of a JSON file.

The content of this JSON file is completely equivalent to the data substructure added to the `app.vars` object when using Quick Check in a Process Plan.

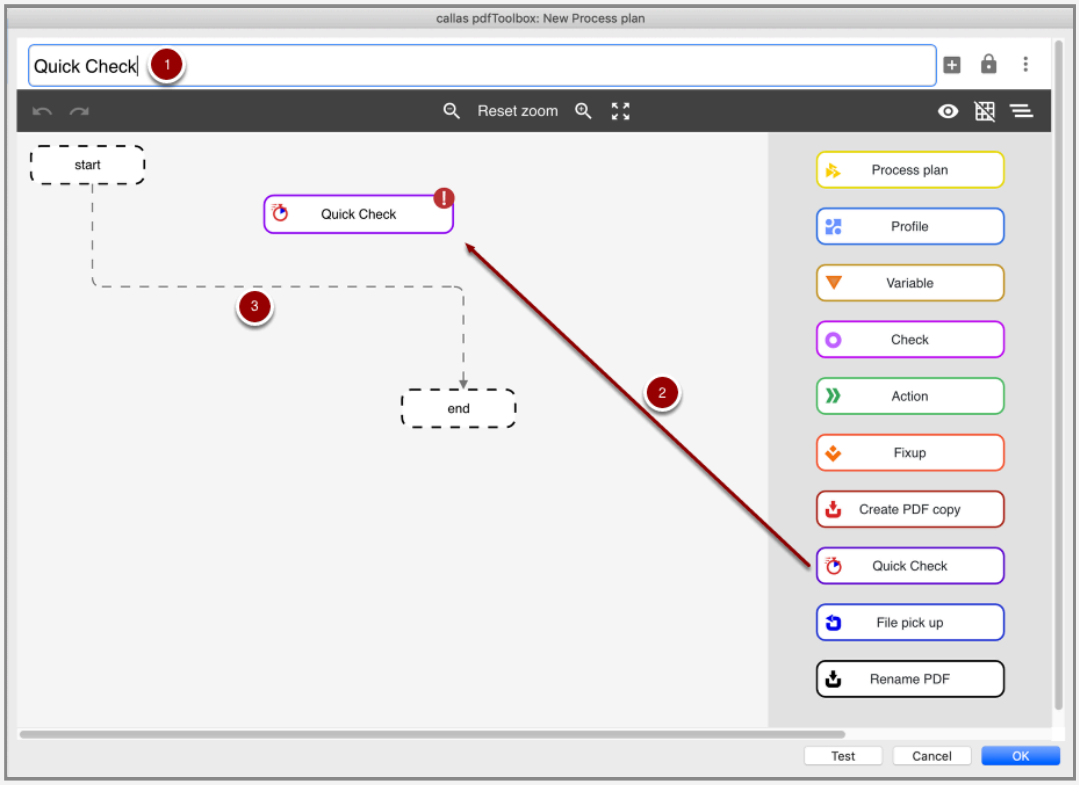
28.2 Using Quick Check as a step in a Process Plan

When creating or editing a Process Plan, it is possible to insert a Quick Check step in the Process Plan.

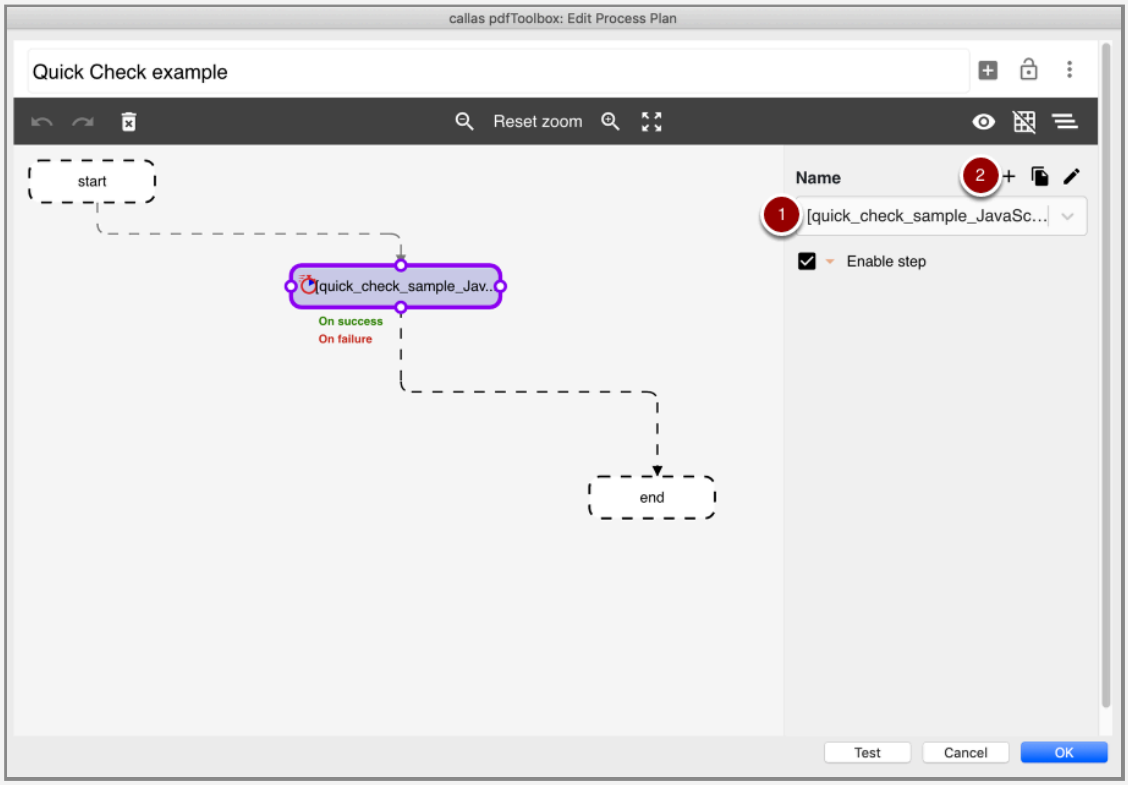
For the purpose of this article, a new Process Plan is created:



Add Quick Check:



1. In the top left field you can enter a suitable name for the new Process plan
2. Select "Quick Check" from the "Sequence" list on the right and drag this step onto the area of the Process plan editor
3. Delete the default connection type between "Start" and "End" (dashed line) and connect "Start" with "Quick Check" and "Quick Check" again with "End" (as shown below)

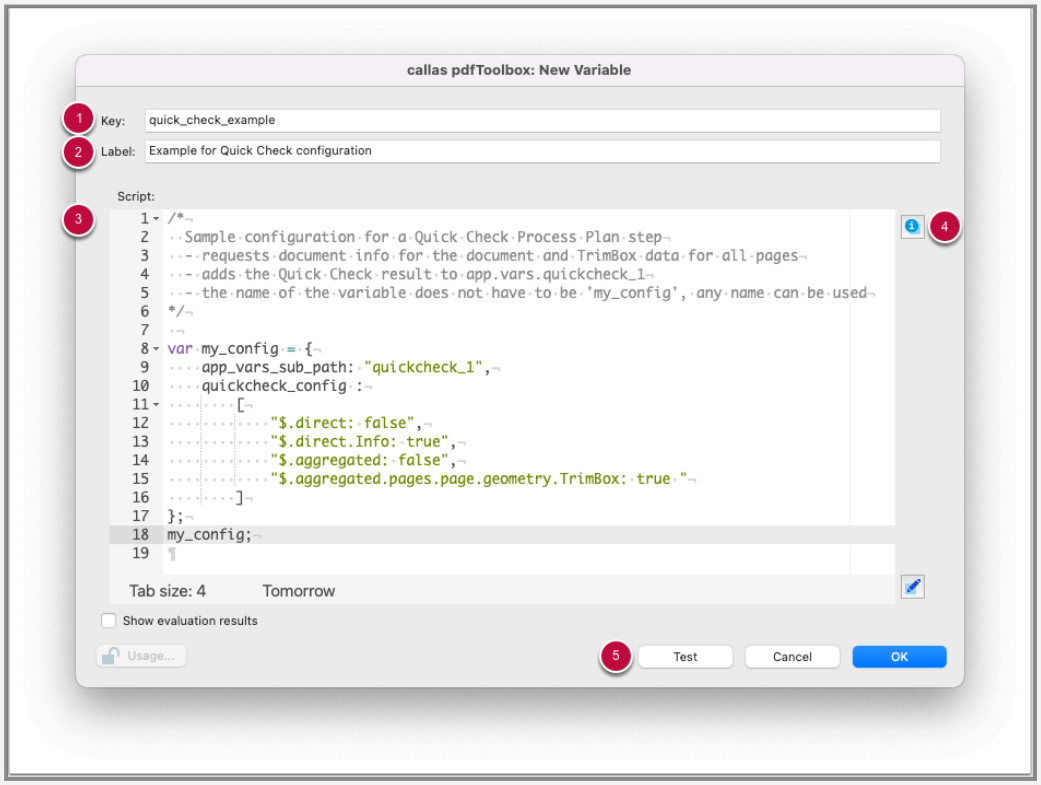


The Quick Check step has now been inserted into the process plan as an element.

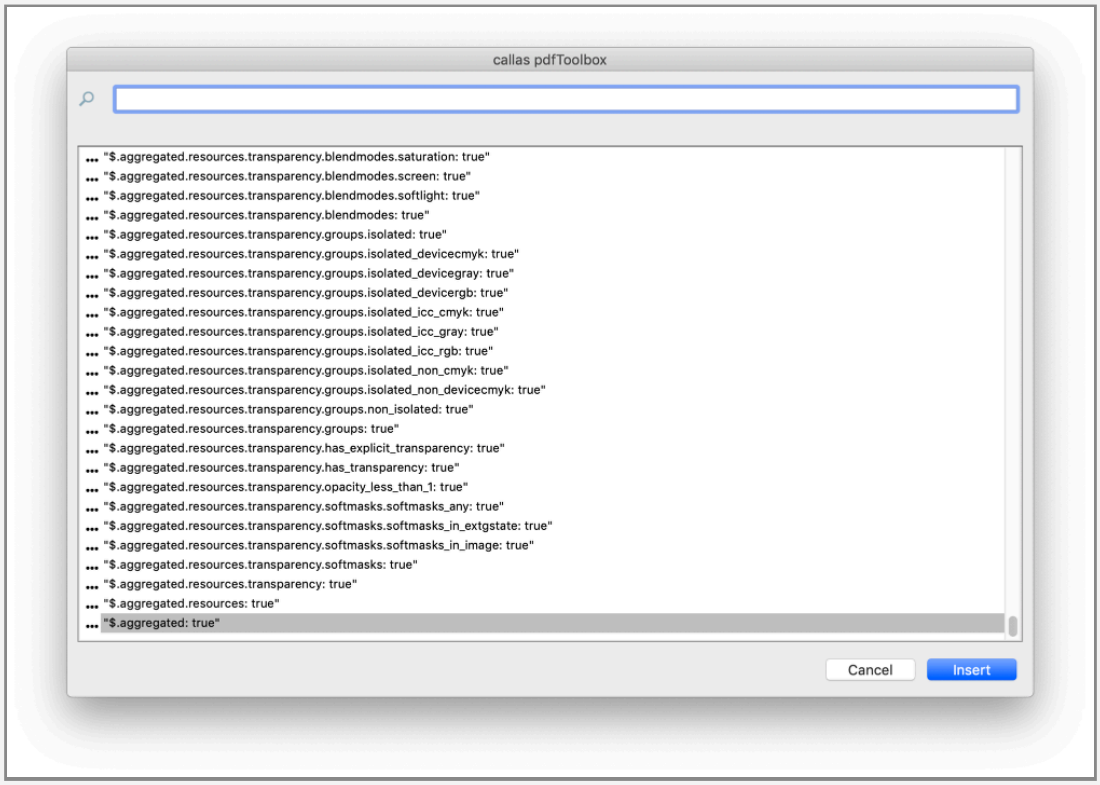
1. In order to set up what the Quick Check step should contribute as a result, either a suitable Quick Check JavaScript variable is selected from the "Name" drop-down menu,
2. or a new Quick Check JavaScript variable is created using the "+" symbol (explained below).

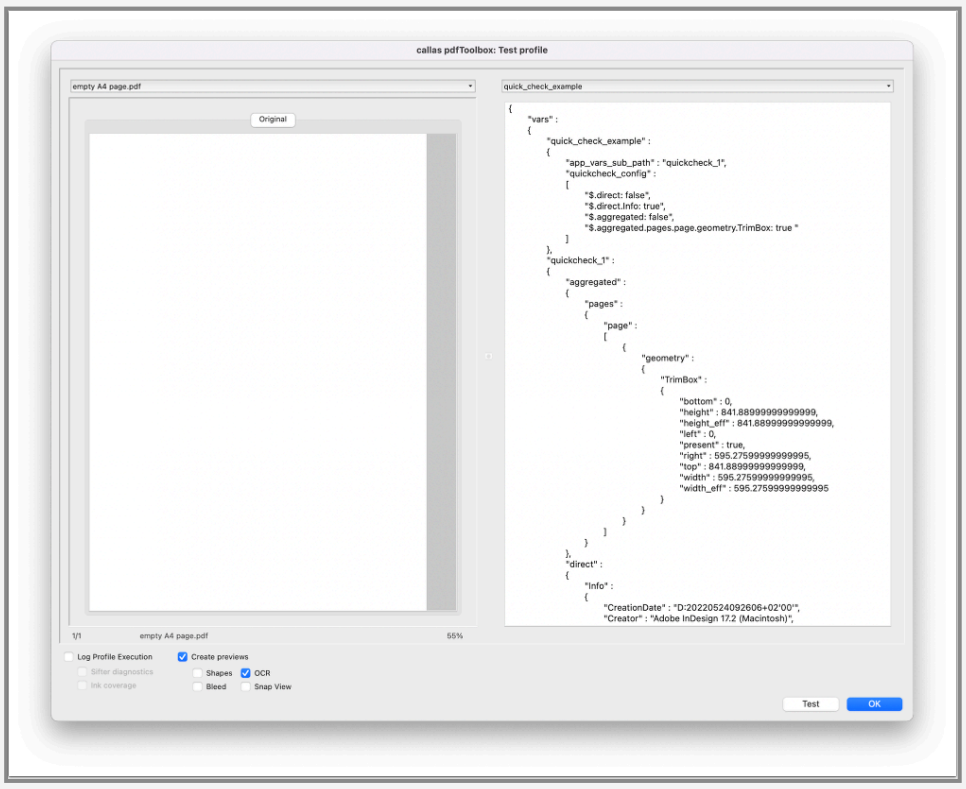
New Quick Check JavaScript variable

When creating a new Quick Check-JavaScript variable, the Edit Quick Check window is opened, as shown below.



1. Enter suitable value for the "Key" (by which the variable can be referenced in the `app.vars` object)
2. Enter suitable value for the "Label" (shown in the user interface when displaying a reference to the variable)
3. Edit the "Script" (what is shown above is the default script that requests the TrimBox for all pages)
4. To view the list of all Quick Check objects, click 'i' and 'Insert Quick Check objects' (a new window opens that is shown below)
5. The "Test" button will open the "Test profile" window which shows the output of the Quick Check Variable (here the size of the TrimBox)


callas pdfToolbox
"$.aggregated.resources.transparency.blendmodes.saturation: true"
"$.aggregated.resources.transparency.blendmodes.screen: true"
"$.aggregated.resources.transparency.blendmodes.softlight: true"
"$.aggregated.resources.transparency.blendmodes: true"
"$.aggregated.resources.transparency.groups.isolated: true"
"$.aggregated.resources.transparency.groups.isolated_devicecmyk: true"
"$.aggregated.resources.transparency.groups.isolated_devicegray: true"
"$.aggregated.resources.transparency.groups.isolated_devicergb: true"
"$.aggregated.resources.transparency.groups.isolated_icc_cmyk: true"
"$.aggregated.resources.transparency.groups.isolated_icc_gray: true"
"$.aggregated.resources.transparency.groups.isolated_icc_rgb: true"
"$.aggregated.resources.transparency.groups.isolated_non_cmyk: true"
"$.aggregated.resources.transparency.groups.isolated_non_devicecmyk: true"
"$.aggregated.resources.transparency.groups.non_isolated: true"
"$.aggregated.resources.transparency.groups: true"
"$.aggregated.resources.transparency.has_explicit_transparency: true"
"$.aggregated.resources.transparency.has_transparency: true"
"$.aggregated.resources.transparency.opacity_less_than_1: true"
"$.aggregated.resources.transparency.softmasks.softmasks_any: true"
"$.aggregated.resources.transparency.softmasks.softmasks_in_extgstate: true"
"$.aggregated.resources.transparency.softmasks.softmasks_in_image: true"
"$.aggregated.resources.transparency.softmasks: true"
"$.aggregated.resources.transparency: true"
"$.aggregated.resources: true"
"$.aggregated: true"
Cancel
Insert


callas pdfToolbox: Test profile
empty A4 page.pdf
quick_check_example
Original
{
"vars" :
{
"quick_check_example" :
{
"app_vars_sub_path" : "quickcheck_1",
"quickcheck_config" :
[
"$.direct: false",
"$.direct.Info: true",
"$.aggregated: false",
"$.aggregated.pages.page.geometry.TrimBox: true "
]
},
"quickcheck_1" :
{
"aggregated" :
{
"pages" :
{
"page" :
[
{
"geometry" :
{
"TrimBox" :
{
"bottom" : 0,
"height" : 841.88999999999999,
"height_eff" : 841.88999999999999,
"left" : 0,
"present" : true,
"right" : 595.27599999999995,
"top" : 841.88999999999999,
"width" : 595.27599999999996,
"width_eff" : 595.27599999999995
}
}
}
]
}
},
"direct" :
{
"Info" :
{
"CreationDate" : "D:20220524092606+02'00'",
"Creator" : "Adobe InDesign 17.2 (Macintosh)",
1/1
empty A4 page.pdf
55%
Log Profile Execution
Create previews
Sifter diagnostics
Ink coverage
Shapes
OCR
Bleed
Snap View
Test
OK

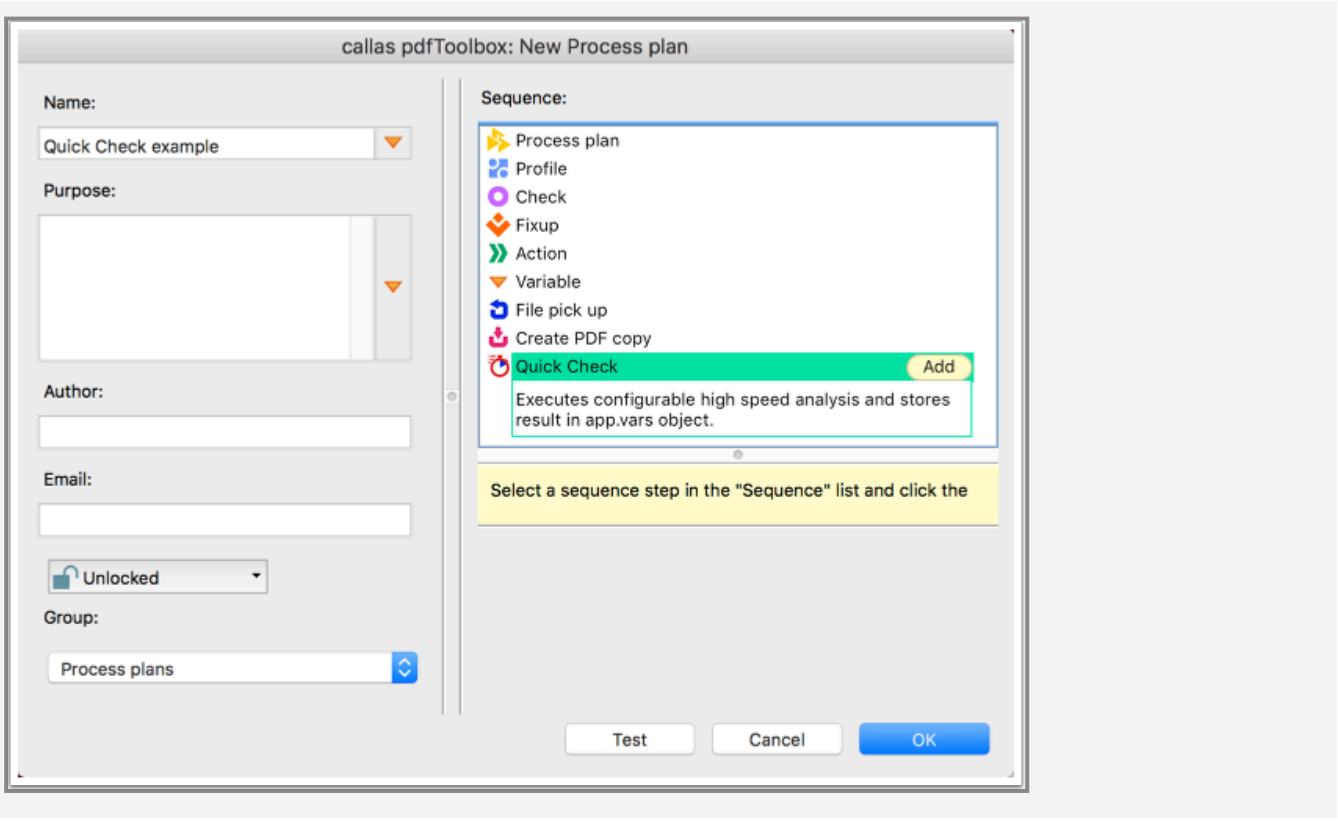
After inserting the required Quick Check object, click OK to save this Quick Check-JavaScript variable, and click OK again to save the new Process Plan.

Whenever this Process Plan is executed, the TrimBox of all pages gets stored in the `app.vars` object under `quickcheck_1`.

Obviously, in this simple form the Process Plan is not very useful. The next part of the article shows a slightly extended variant of this Process Plan, that actually makes some interesting use of the information retrieved by the Quick Check step.

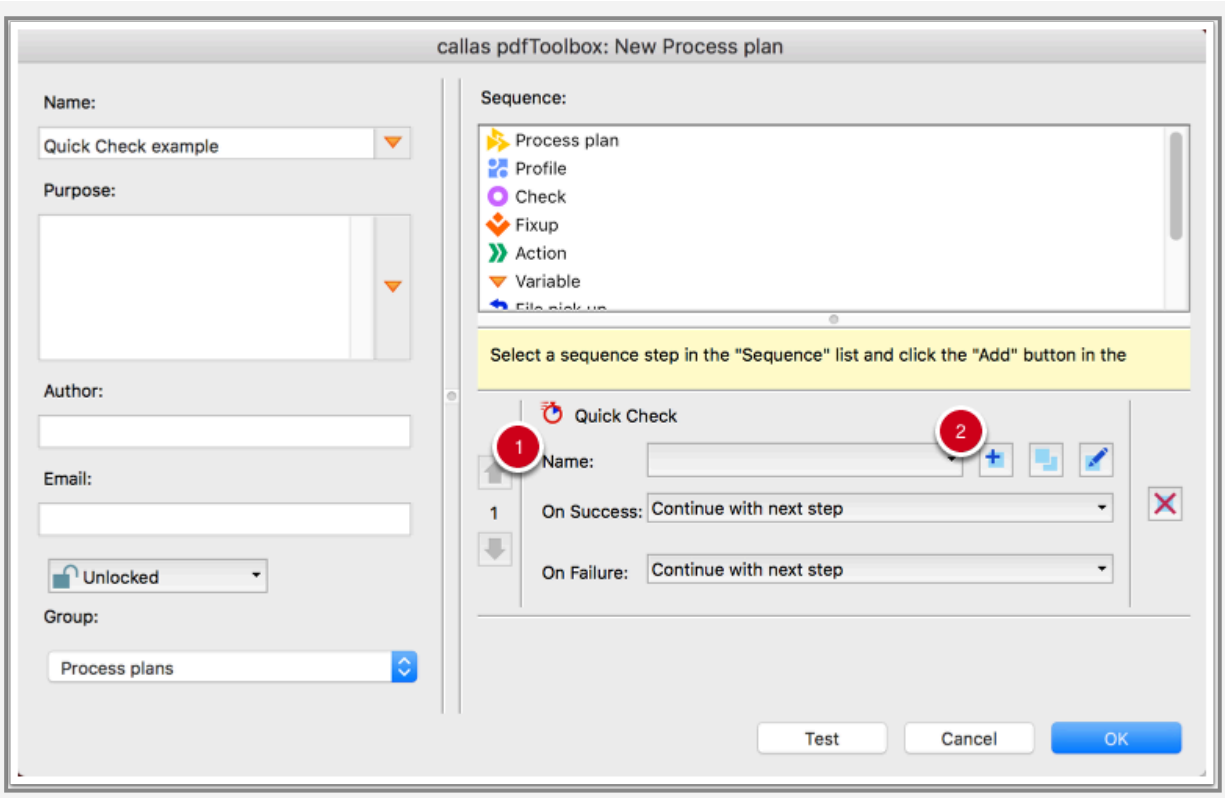
Old Process Plan editor

After entering a suitable name for the new Process Plan, select "Quick Check" from the "Sequence" list and click on the "Add" button in that list entry:



A new "Quick Check" step gets inserted as the first item in the list of Process Plan steps.

1. In order to configure what the "Quick Check" step is to provide as a result, either a suitable Quick Check-JavaScript variable has to be chosen in the "Name" pop-up list,
2. or a new Quick Check-JavaScript variable has to be created using the "+" icon.



How to use information retrieved by a Quick Check step in a Process Plan

In order to try out the Process Plan presented below, please download it here:

Quick_Check_example.kfpx

After importing the downloaded "Quick_Check_example.kfpx" file and importing it into pdfToolbox Desktop, it should show up as can be seen below:



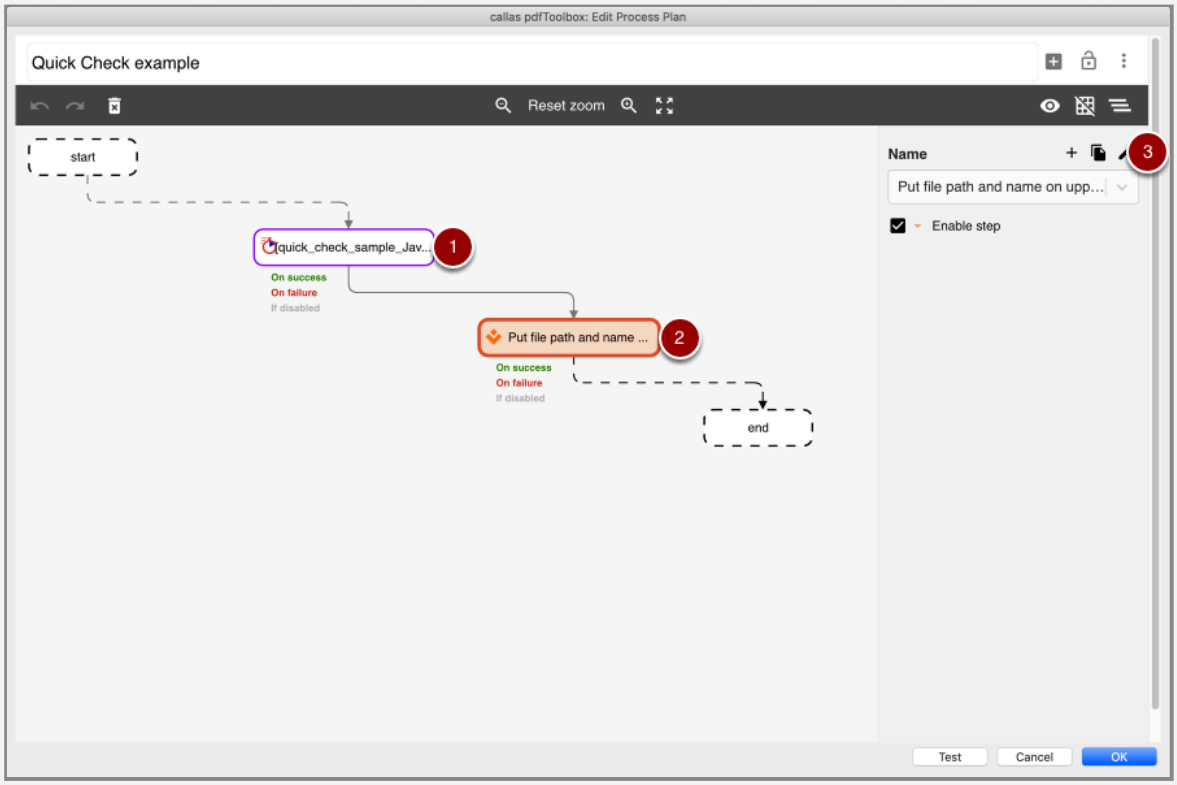
In order to explore how this Process Plan has been constructed, click on the "Edit..." button next to the "Quick Check example" entry.

The "Edit Process Plan" dialog will open. It contains two steps:

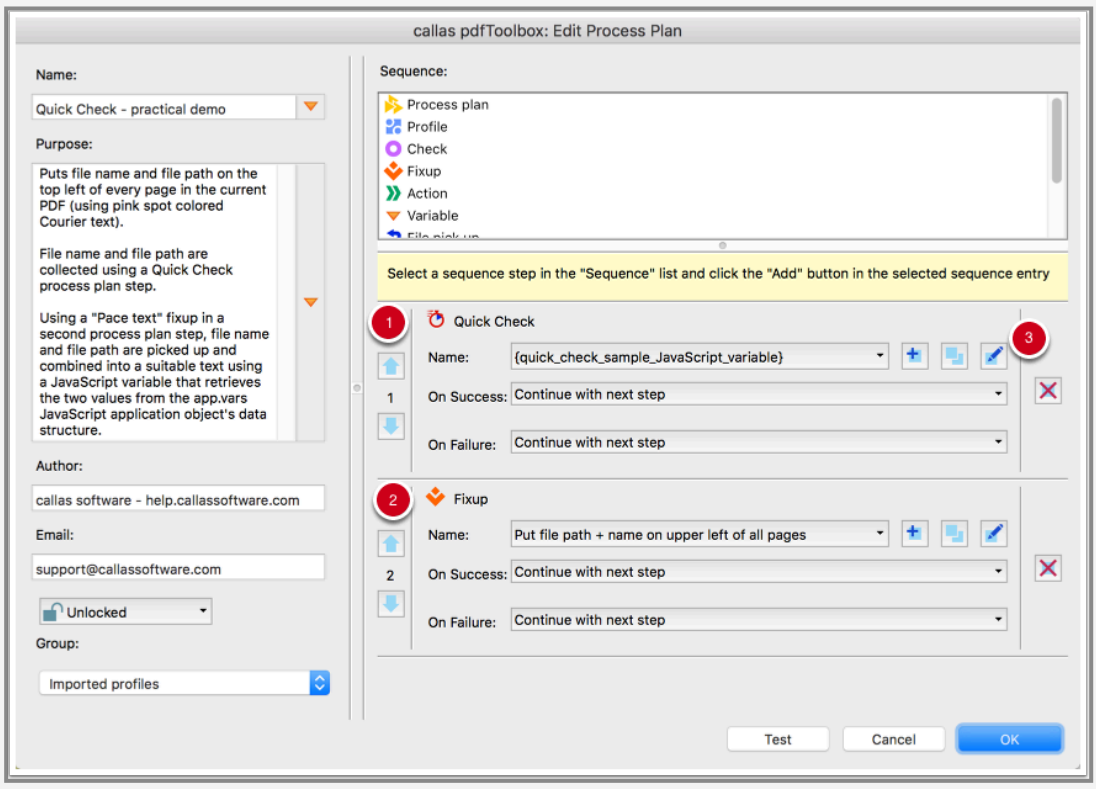
1. Quick Check step using a variable named "quick_check_sample_JavaScript_variable" that collects information from the current PDF
2. Fixup step "Put file name and file path on upper left of all pages" that picks up the collected information and uses it to place text on each page of the current PDF

In order to have a look at the way the JavaScript variable has been set up,

3. click on the Edit button to the right of the "Name" popup.



Alternate image for the older Process Plan UI:

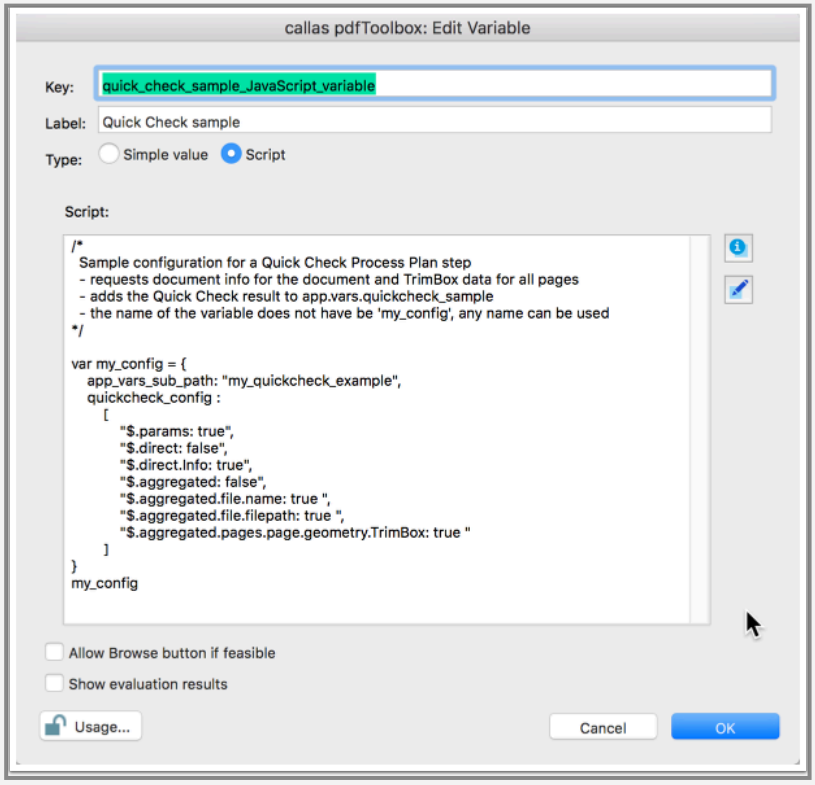


The "Edit Variable" window will open, revealing the setup of the variable "quick_check_sample_JavaScript_variable".

The relevant part of the configuration are these two lines:

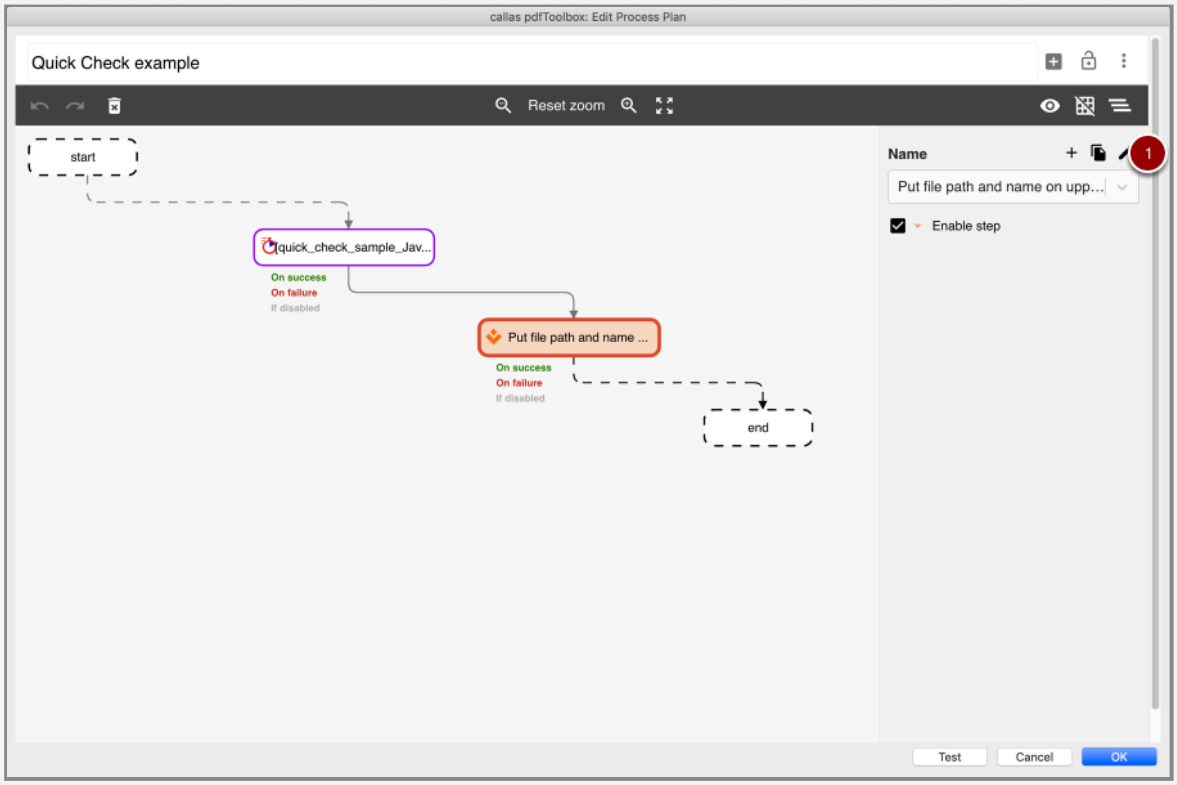
```
            "$.aggregated.file.name: true ",
            "$.aggregated.file.filepath: true ",
```

which request that Quick Check retrieves information about the file name and the file path of the current PDF. For a detailed description of the configuration set up please check out the article Quick Check configuration syntax.

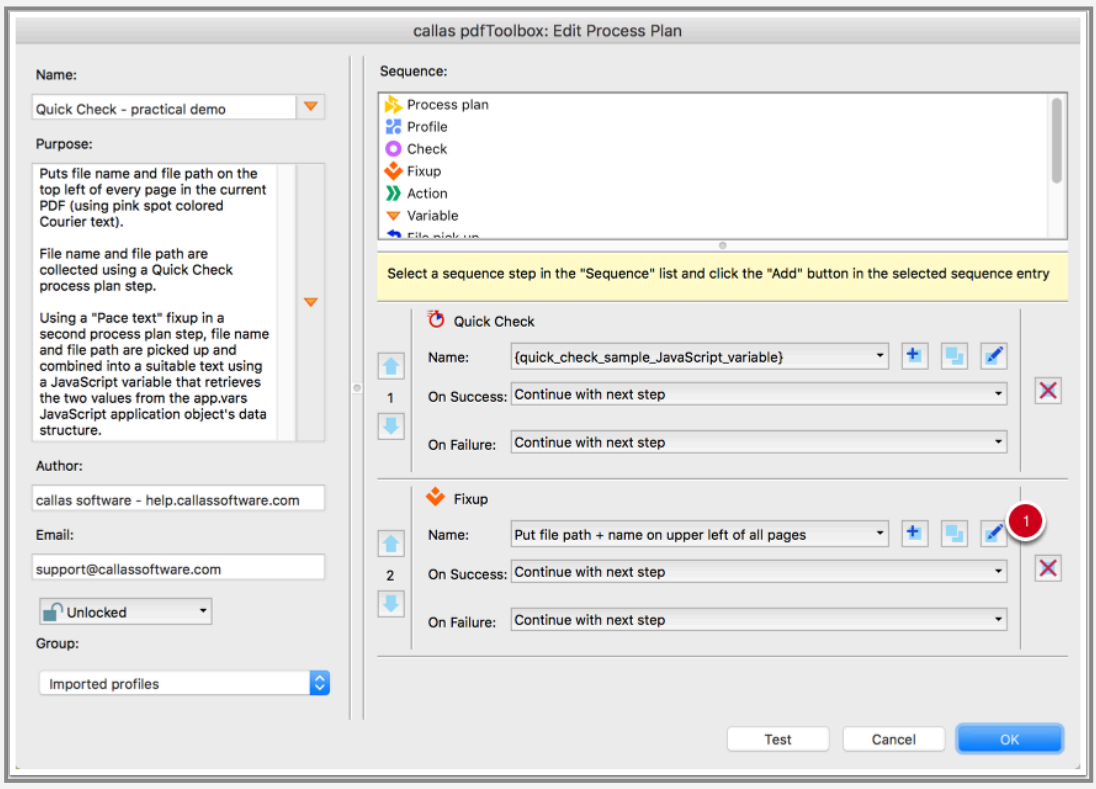


Close the "Edit Variable" window, and thus go back to the Edit window for the Process Plan.

1. Click on the Edit icon to the right of the Name entry for the Fixup step.



Alternate image for the older Process Plan UI:

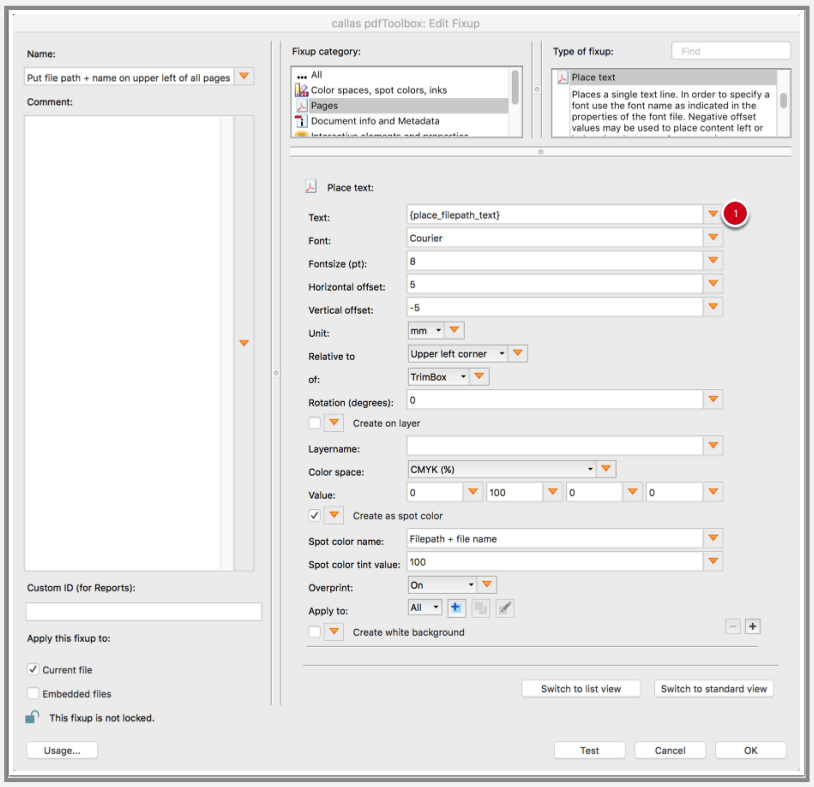


The "Edit Fixup" window for the "Put file path and name on upper left of all pages" Fixup will open.

The numerous fields in this Fixup are filled with values that will make the contents of the "Text" field show up in the upper left of all pages of the current page using Courier font in pink spot color.

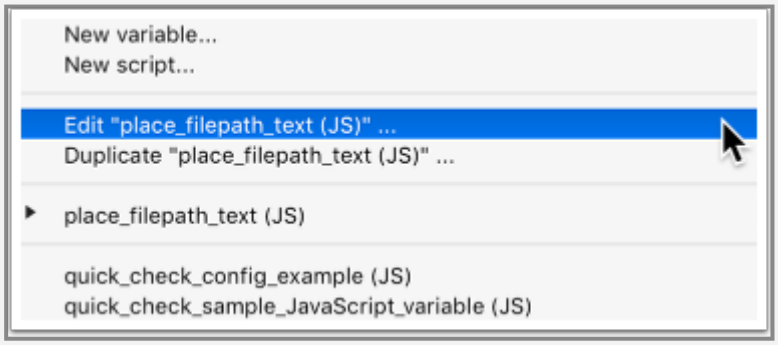
The more interesting part of this Fixup at least in this context is how the contents of the "Text" is defined.

1. Click on the orange triangle popup to the right of the "Text" field.



A list of already defined variables is now shown, plus various options regarding existing or new variables.

1. Click on 'Edit "place_filepath_text"...'



The "Edit Variable" window for the variable "place_filepath_text" will open.

Important: this not the same variable as the JavaScript variable defined in the QuickCheck step (which configures that Quick Check step), rather, it is a separate JavaScript variable that makes use of the information retrieved by the Quick Check step.

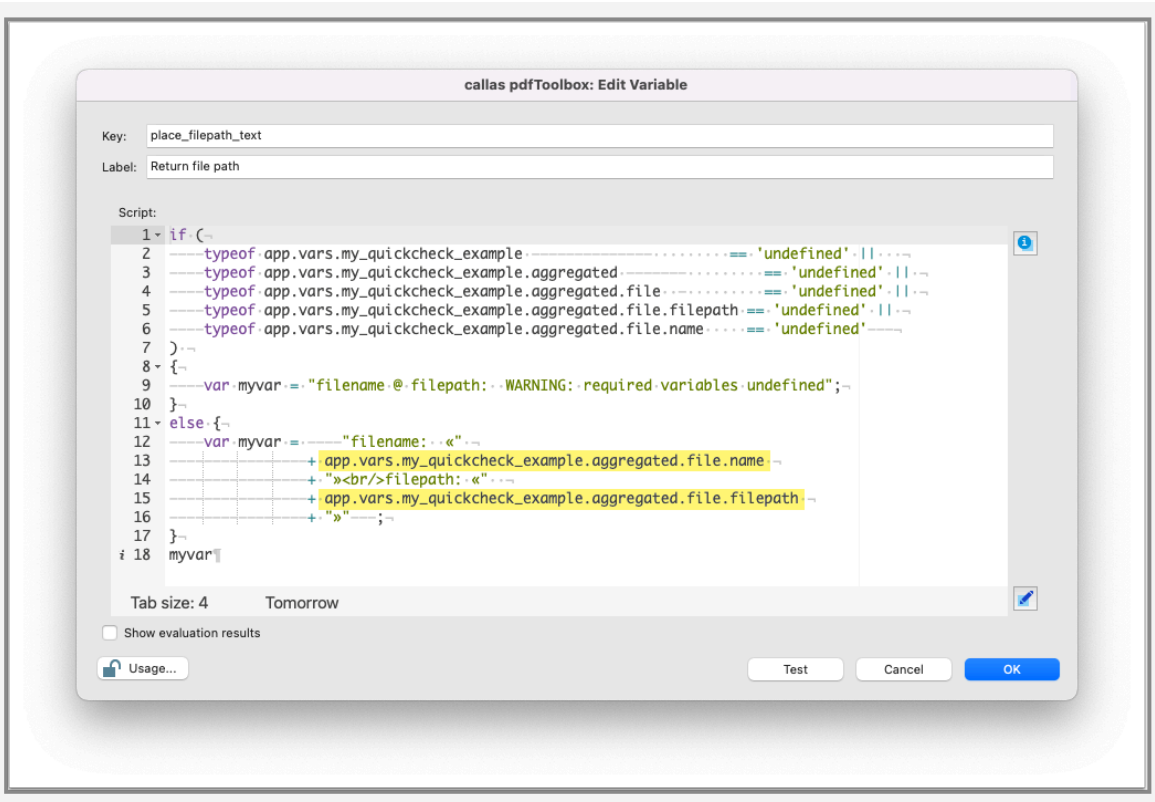
As that information will only be available if the Fixup gets executed as part of a Process Plan, where in a previous step a suitable Quick Check step has been executed. Some precautions are taken in the JavaScript code that give a kind of error message "filename @ filepath: WARNING: required variables undefined" if the Fixup is executed on its own or in a different Process Plan or Profile. This is managed by the `if` branch of the JavaScript code:

```
if (
        typeof app.vars.my_quickcheck_example
           == 'undefined' ||
        typeof app.vars.my_quickcheck_example.aggregated
 == 'undefined' ||
        typeof app.vars.my_quickcheck_example.aggregated.file                == 
'undefined' ||
        typeof app.vars.my_quickcheck_example.aggregated.file.filepath  == 'unde-
fined' ||
        typeof app.vars.my_quickcheck_example.aggregated.file.name      == 'unde-
fined'
)
{
        var myvar = "filename @ filepath:  WARNING: required variables undefined";
}
```

The intended main portion of the JavaScript variable is contained in the `else` branch:

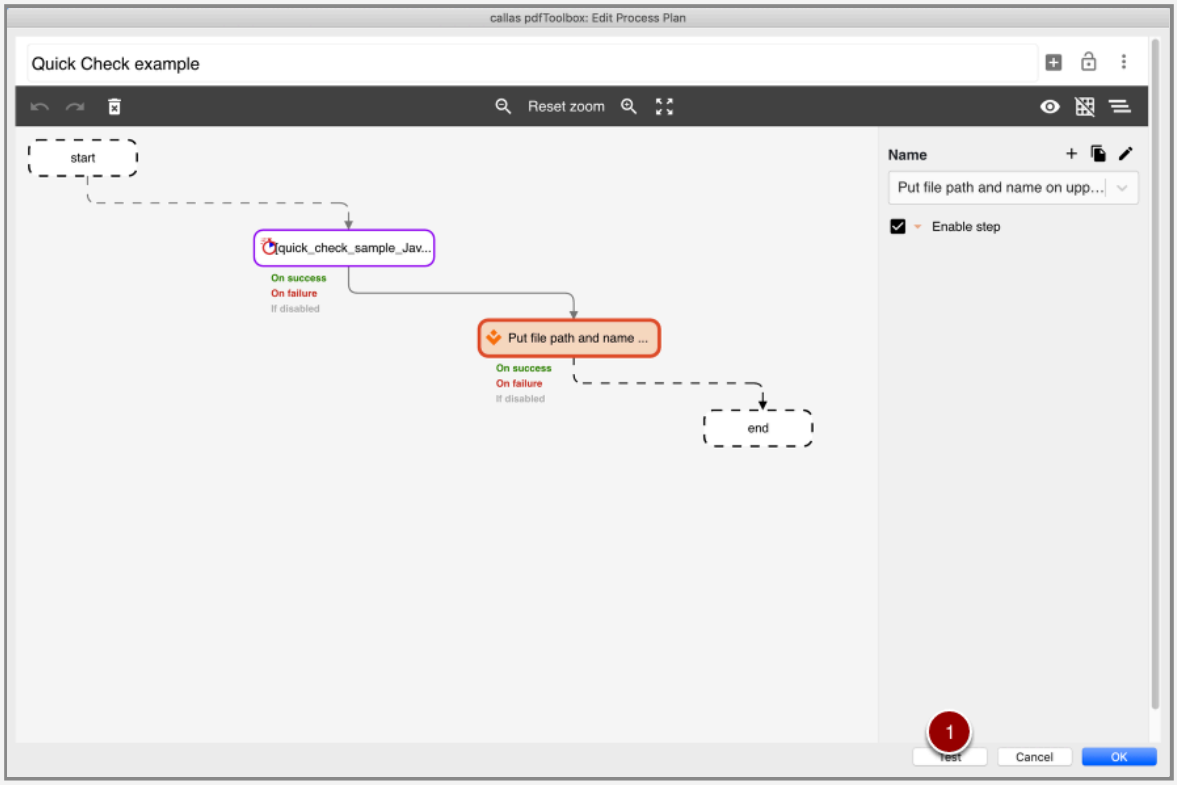
```
else {
        var myvar =         "filename:  «"
                                + app.vars.my_quickcheck_example.aggregated.file.
name
                                + "»<br/>filepath: «"
                                + app.vars.my_quickcheck_example.aggregated.file.
filepath
                                + "»"        ;
}
```

This else branch creates a string built from two variables, `app.vars.my_quickcheck_example.aggregated.file.name` and `app.vars.my_quickcheck_example.aggregated.file.filepath`, concatenated with other bits of text into a nicely formatted string.

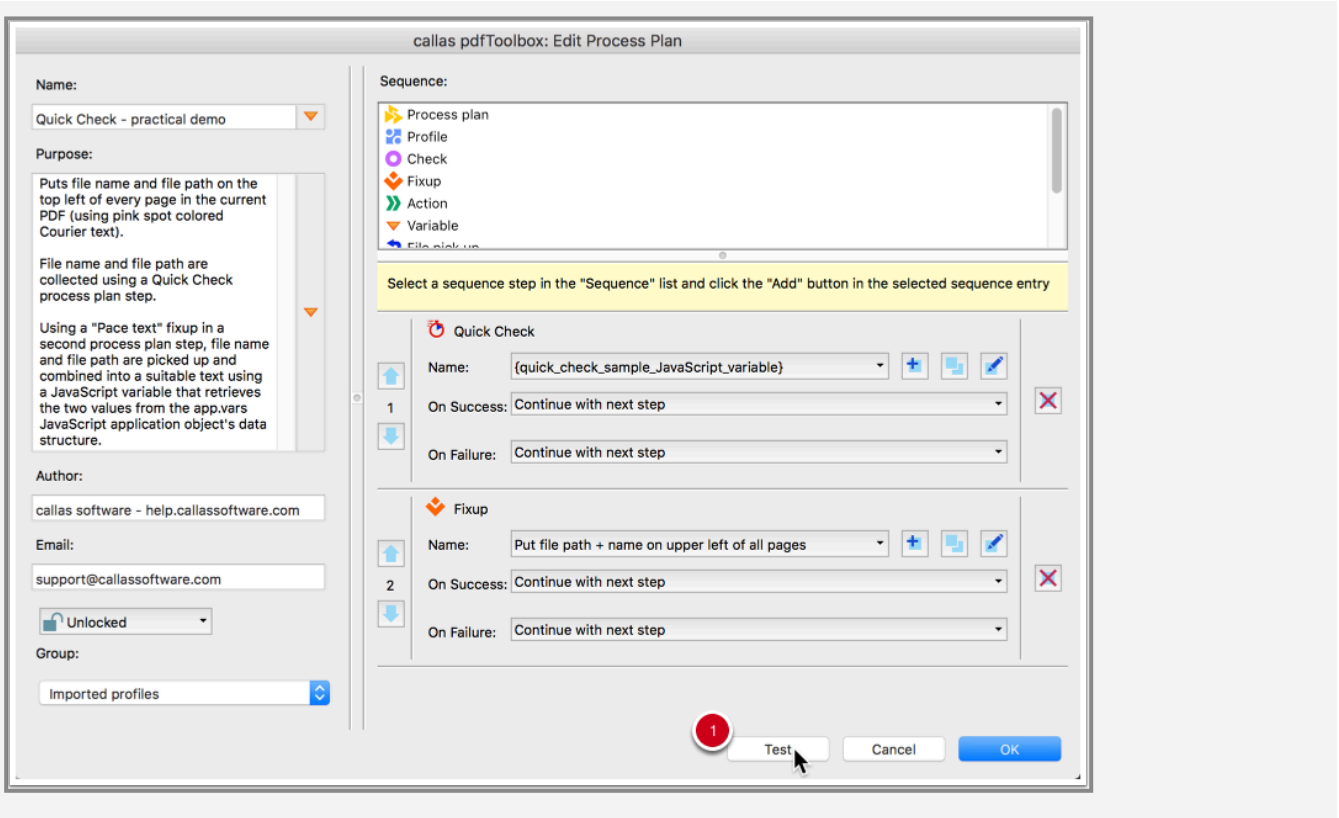


After closing this "Edit Variable" window, and then also the "Edit Fixup" window, the "Edit Process Plan" window will be shown again. Instead of also closing this window and trying the Process Plan on one of your files,

1. use the "Test" feature by clicking on the "Test" button

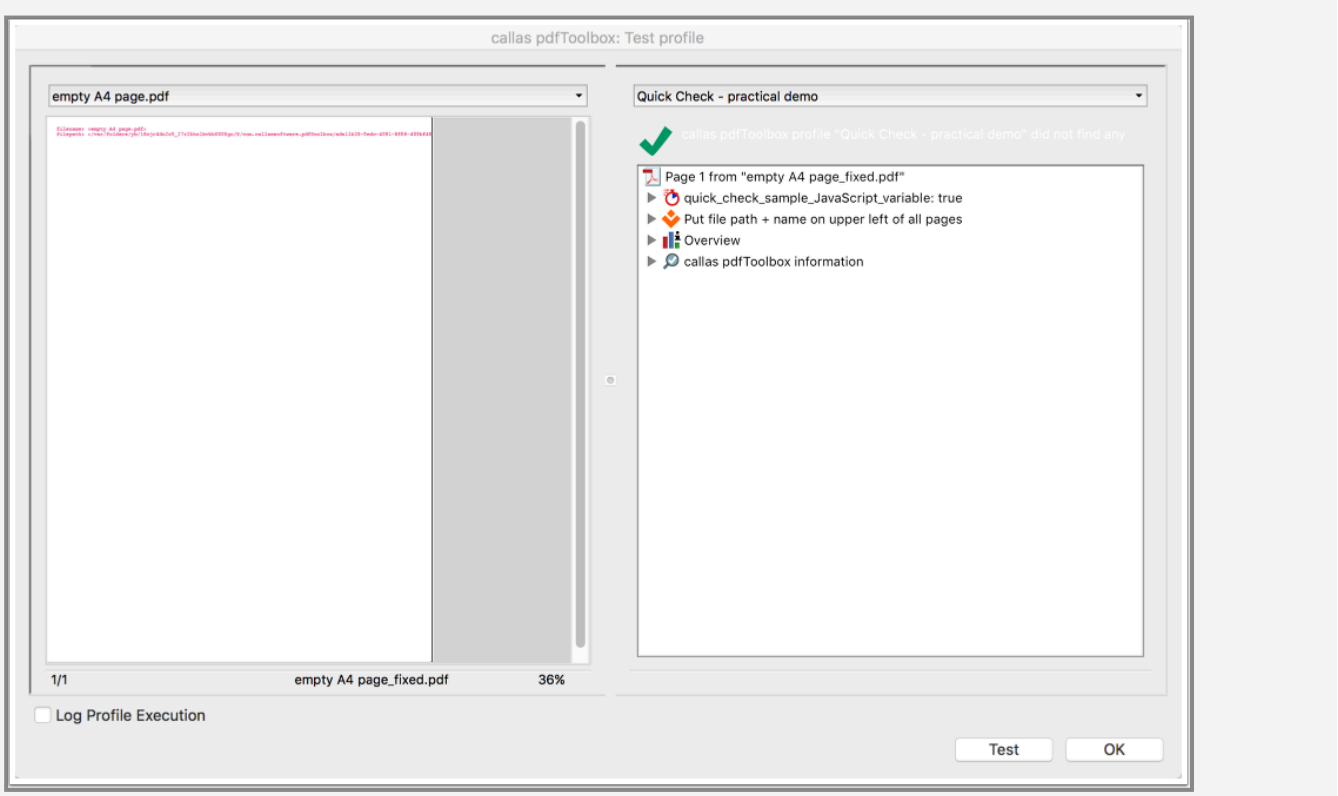


Alternate image for the older Process Plan UI:



This will open the "Test profile" window – a very convenient environment to try Process Plans (or Profiles, Fixups and Checks) on a copy of a currently open PDF file without the risk of inadvertently breaking your files ("Test" processing is always carried out on temporary copies of your PDF files), and without tedious open/edit/close/apply cycles, and without any need to clean up processed copies of your PDF files.

As you can see in the screen shot some text is showing up in the upper left of the sample file used here (a PDF consisting of an empty A 4 sized page, and named accordingly).


callas pdfToolbox: Test profile
empty A4 page.pdf
Quick Check - practical demo
callas pdfToolbox profile "Quick Check - practical demo" did not find any
Page 1 from "empty A4 page_fixed.pdf"
quick_check_sample_JavaScript_variable: true
Put file path + name on upper left of all pages
Overview
callas pdfToolbox information
1/1
empty A4 page_fixed.pdf
36%
Log Profile Execution
Test
OK

28.3 Quick Check configuration syntax

Quick Check configuration is done by using one or several filter expressions. Any filter expression has a certain specificity and either includes or excludes the respective sub data structure or data field. Filter expressions are interpreted against the structure of possible Quick Check output (for more details see the next articles for the syntax of Quick Check Output). Filtering works for the following areas:

- "aggregated"
- "direct"
- "status" (this data structure will always be written and cannot be turned off)

The next 3 articles explain each of these 3 data structures.

Example: Configuration syntax

The filter expressions are based on the 'path' to a given data (sub)structure or data element in the Quick Check output structure. Assuming the following output excerpt:

```
"aggregated": {
  "pages": {
                        "page" : [
                                {
                                        "geometry" : {
                                                "TrimBox" : {
                                                        "left" : 0,
                                                        "bottom" : 0,
                                                        "right" : 612,
                                                        "top" : 792,
                                                        "width" : 612,
                                                        "height" : 792,
                                                        "width_eff" : 612,
                                                        "height_eff" : 792
                                                }
                                        }
                                                                },
                                {
```

```
                                        "geometry" : {
                                                "TrimBox" : {
                                                        "left" : 0,
                                                        "bottom" : 0,
                                                        "right" : 612,
                                                        "top" : 792,
                                                        "width" : 612,
                                                        "height" : 792,
                                                        "width_eff" : 612,
                                                        "height_eff" : 792
                                                }
                                        }
                                }
                        ]
                }
}
```

Click to copy

and assuming that only the effective width `width_eff` and effective height `height_eff` data elements of the TrimBox of each page are of interest, the following filtering expression would be used:

```
$.direct: false
$.aggregated: false
$.aggregated.pages.page.geometry.TrimBox.width_eff : true
$.aggregated.pages.page.geometry.TrimBox.height_eff : true
```

Click to copy

1. **The $ element serves as a kind of virtual root element, to which the path to the data structure or element of interest is appended.**
2. **In order to not trigger creation output of massive amounts of data it is recommended to completely turn off the top most level of the two main areas, using `$.direct: false` and `$.aggregated: false`.**
3. **Next, filter expressions that target the exact data substructure or element that are to be included in the output are added – in this example just the two**

entries inside the TrimBox data structures for each page.

Filtering for all pages, a specific page or consolidated data from all pages

The Quick Check filter expressions make it possible to request page related data either for a specific page (or range of pages), all pages, or in a consolidated form aggregating information from all pages. The page index is one based. This means that a particular page can be addressed using the usual array notation (the first element starts at 1). The following expressions illustrate this:

```
$.aggregated.resources.color.spotcolors: true
```

Output: summary of all spotcolors used in the document.

```
$.aggregated.pages.page.resources.color.spotcolors : true
```

Output: all spotcolors per page for the hole document.

```
$.aggregated.pages.page[2].resources.color.spotcolors: true
```

Output: only the spotcolors used on page 2.

```
$.aggregated.pages.page[-1].resources.color.spotcolors: true
```

Output: only the spotcolors used on the last page.

```
$.aggregated.pages.page[2 4].resources.color.spotcolors: true
```

Output: all spotcolors per page for page 2, 3 and 4.

For all filter expressions that start with `$.aggregated.pages.page.` a page index can be used. Otherwise

they will be summarised. In the sample above the page index is: `[2]`, `[-1]` & `[2 4]`.

For `$.aggregated.pages.page` the following sub-structures are available:

- `geometry`: various data elements based on page geometry boxes
- `info`: data elements for sequential page number (starting at 1 for first page) and page label
- `resources`: data structures and elements for color and font resources
- `annotations`: data elements regarding annotations and their properties
- `contentstream`: data elements based on the size of the content stream (bytes)
- `pieceinfo`: data elements regarding metadata for application-specific data

Set a timeout and reflect it in the Quick Check(JSON) output

A document with millions of pages is not unusual in production print which is a typical environment for QuickCheck. A timeout can be set within the QuickCheck configuration with the below parameters:

```
$.settings.timeout_is_error: true
$.settings.timeout: *number of seconds*
```

Click to copy

- `$.settings.timeout_is_error`: A timeout will appear as an error in the 'status'; default = false
- `$.settings.timeout`: Initiates a timeout after ... seconds; default = -1 (off)

Example:

For the below QuickCheck call:

```
/pdfToolbox --quickcheck /Downloads/Quickcheck-timeout.cfg /Downloads/Large.PDF
```

Click to copy
and the following configuration 'Quickcheck-timeout.cfg':

```
$.direct: false
$.direct.Info: true
$.aggregated: true
$.settings.timeout_is_error: true
$.settings.timeout: 1
$.settings.status: true
```

Click to copy
The QuickCheck status looks like this:

```
..omitted
  },
  "status": {
    "time_needed_sec" : 1.011101,
    "result" : "incomplete",
    "number_of_pages" : 984,
    "abort_at_page" : 515,
    "level" : "errors",
    "returncode" : 113,
    "errors" : [ { "code" : 113, "msg" : "Conversion did not finish during user
specified timeout." }
]


  }
```

Click to copy

Hints and tricks

If a dot (.) or colon (:) occurs in a filter path identifier, then this glyph must be escaped with a preceding backslash (\), e.g..:

- $.direct.Root.Private.Test\:2\:Colon : true
- $.direct.Root.Private.Test\.2\.Points : true

28.4 All "aggregated" Quick Check objects and output

If you are coming here from the previous articles, you might know that Quick Check configuration is done by using one or several filter expressions. Any filter expression has a certain specificity and either includes or excludes the respective sub data structure or data field.

The "aggregated" block

The "aggregated" block contains several sub-divisions that reflect aggregated information from various areas, such as color or font resources, transparency and overprint, etc.

In principle, most of the information provided here could also be retrieved by accessing data structures using the "direct" block mechanism, but that would require solid understanding of the underlying data structures and also sometimes quite complicated processing. The "aggregated" block offers such information in a ready to use fashion. Still, if some information is needed beyond what "aggregated" offers, it might be feasible to retrieve such information from processing "direct" data structures.



The best approach to find out how "aggregated" can be used is to request all data under "aggregated" ("`$.aggregated: true`"), find the area needed, and then build the configuration filter expressions by following the 'path' to that area.

Areas inside "aggregated" block

The below areas under "aggregated" block contain

- A basic explanation
- Filter expressions for use in the config file (for use in pdfToolbox CLI)
- An example of a filter and the respective result

NOTE: a list of all filter expressions for "aggregated" data substructures required for JavaScript variables in a Quick Check step in a Process Plan are at the bottom of the page (you can also click here to get there).

1. "bookmarks" area

Creates output reflecting bookmarks in a PDF file (called Outline in the PDF syntax). A flat array contains a list of all bookmarks found in the PDF file. The nesting level of each bookmark is indicated by the `level` data element, reflecting the nesting of bookmarks as typically displayed in a PDF viewing program. The main piece of information actually conveyed is the text of the bookmark. The `bookmarks` arrays does not reflect the actual PDF data structures in any way.

Filter expressions (for use in the config file)

```
$.aggregated.bookmarks: true
$.aggregated.bookmarks.bookmark: true
$.aggregated.bookmarks.bookmark.level: true
$.aggregated.bookmarks.bookmark.name: true
$.aggregated.bookmarks.bookmark.page: true
$.aggregated.bookmarks.length: true
```

Click to copy

"bookmarks" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.bookmarks: true
```

Click to copy

when used for the PDF file for the ISO 32000-1 standard results in the following Quick Check output:


```
{
  "aggregated": {
    "bookmarks": {
      "length" : 822,
      "bookmark": [
        {
          "name" : "Contents Page",
          "level" : 1,
          "page" : 3
        },
        {
          "name" : "Foreword",
          "level" : 1,
          "page" : 6
        },
        {
          "name" : "Introduction",
          "level" : 1,
          "page" : 7
        },
        {
          "name" : "1 Scope",
          "level" : 1,
          "page" : 9
        },
        {
          "name" : "2 Conformance",
          "level" : 1,
          "page" : 9
        },
        {
          "name" : "2.1 General",
          "level" : 2,
          "page" : 9
        },
        {
          "name" : "2.2 Conforming readers",
          "level" : 2,
          "page" : 9
        },
        {
```

```
          "name" : "2.3 Conforming writers",
          "level" : 2,
          "page" : 10
        },
        {
          "name" : "2.4 Conforming products",
          "level" : 2,
          "page" : 10
        },
[... remaining entries omitted ...]
```

Click to copy

2. "doc" area

Creates output reflecting a handful of document properties:

- `annotations` : lists all annotations in the PDF file with their properties:
- `annotation_flags` : define the behavior, editing options and the representation (screen and paper) of an annotations e. g. hidden, invisible
- `annotation_state` : author-specific state of an annotation e.g. marked, accepted
- `bbox` : defines the location of the annotation on the page
- `fields` : properties from a form field (titel, value)
- `created` : date and time when the annotation was created
- `content` : either text to be displayed for the annotation or alternate description of the annotations content (depending on annotation type)
- `filename` : for file attachment annotations
- `in_reply_to` : reference to the annotation that this annotation is “in reply to”
- `intent` : name that discribes the intent of an annotation
- `last_modified` : date and time when the annotation was last modified
- `layer` : An annotation can be placed on a layer. This filter expression provides infos about the layer (e.g. intent, name, type)

- `name` : The name of an icon to be used in displaying the annotation e. g. comment
- `pagenum` : number of the page where the annotation is placed
- `quadpoints` : additional entry to a link annotation. The defined array specifies the area (coordinates) where the link should be activated.
- `subject` : short description of the annotation subject e. g. Sticky Note, Typewritten Text
- `text_lable` : represents the name of the user authoring the annotation
- `type` : annotation type such as Link, Text, Square, Free Text
- `unique_id` : returns the ID of the indirect object (object number as used in the PDF)
- `url` : destination URL of a link
- `num` : number of annotations
- `create_id` : first of the two entries in the **ID** array in the PDF document trailer
- `created` : creation date of the PDF as encoded inside the PDF in the **Info** dictionary
- `modified` : last modification date of the PDF as encoded inside the PDF in the **Info** dictionary
- `modified_id` : second of the two entries in the **ID** array in the PDF document trailer
- `pdf_version` : PDF version as encoded inside the PDF, either by the header of the PDF file or via the **Version**entry in the **Catalog** dictionary of the PDF
- `encryption` : encryption properties like the encryption type/algorithm used from the document's **encryption**dictionary
- `xmpmetadata` : XMP Metadata as in the **Catalog** dictionary of the PDF
- `potential_syntax_issues` : issues about the syntax of PDF at the object, file, and document level
- `dpartroot` : properties of DPartRoot dictionary to describe the document parts hierarchy for a PDF document
- `outputintents` : used output intents in a PDF file
- `pdf/a/e/vt/x/ua` : true, if the PDF file conforms to a specific PDF standard
- `rolemap` : true, if the rolemap dictionary is present
- `is_tagged` : true, if the PDF is tagged (accessible PDF)

- `digsig`: properties from a Digital Signature (filter, name or SubFilter)
- `lang`: language identifier specifying the natural language for all text in the document

Filter expressions (for use in the config file)

```
$.aggregated.doc: true

$.aggregated.doc.annotations: true
$.aggregated.doc.annotations.annot: true
$.aggregated.doc.annotations.annot.annotation_flags: true
$.aggregated.doc.annotations.annot.annotation_state: true
$.aggregated.doc.annotations.annot.bbox: true
$.aggregated.doc.annotations.annot.created: true
$.aggregated.doc.annotations.annot.content: true
$.aggregated.doc.annotations.annot.filename: true
$.aggregated.doc.annotations.annot.in_reply_to: true
$.aggregated.doc.annotations.annot.intent: true
$.aggregated.doc.annotations.annot.last_modified: true
$.aggregated.doc.annotations.annot.layer: true
$.aggregated.doc.annotations.annot.name: true
$.aggregated.doc.annotations.annot.pagenum: true
$.aggregated.doc.annotations.annot.quadpoints: true
$.aggregated.doc.annotations.annot.subject: true
$.aggregated.doc.annotations.annot.text_label: true
$.aggregated.doc.annotations.annot.type: true
$.aggregated.doc.annotations.annot.unique_id: true
$.aggregated.doc.annotations.annot.url: true
$.aggregated.doc.annotations.num: true

$.aggregated.doc.fields: true
$.aggregated.doc.fields.field: true
$.aggregated.doc.fields.field.title: true
$.aggregated.doc.fields.field.value: true

$.aggregated.doc.create_id: true
$.aggregated.doc.created: true

$.aggregated.doc.dpartroot: true
$.aggregated.doc.dpartroot.dparts.dpm: true
$.aggregated.doc.dpartroot.dparts.dparts: true
$.aggregated.doc.dpartroot.dparts.end: true
```

```
$.aggregated.doc.dpartroot.dparts.start: true
$.aggregated.doc.dpartroot.dparts: true
$.aggregated.doc.dpartroot.nodenamelist: true
$.aggregated.doc.dpartroot.recordlevel: true

$.aggregated.doc.modified: true
$.aggregated.doc.modified_id: true

$.aggregated.doc.is_tagged: true

$.aggregated.doc.lang: true

$.aggregated.doc.outputintents: true
$.aggregated.doc.outputintents.length: true
$.aggregated.doc.outputintents.outputintent: true
$.aggregated.doc.outputintents.outputintent.destoutprofile: true
$.aggregated.doc.outputintents.outputintent.destoutprofile.colorspace: true
$.aggregated.doc.outputintents.outputintent.destoutprofile.name: true
$.aggregated.doc.outputintents.outputintent.destoutprofile.profiletype: true
$.aggregated.doc.outputintents.outputintent.outputintentid: true
$.aggregated.doc.outputintents.outputintent.type: true

$.aggregated.doc.pdf_version: true

$.aggregated.doc.pdfa: true
$.aggregated.doc.pdfa.destoutprofile: true
$.aggregated.doc.pdfa.destoutprofile.colorspace: true
$.aggregated.doc.pdfa.destoutprofile.name: true
$.aggregated.doc.pdfa.destoutprofile.profiletype: true
$.aggregated.doc.pdfa.outputintentid: true
$.aggregated.doc.pdfa.type: true

$.aggregated.doc.pdfe: true
$.aggregated.doc.pdfe.destoutprofile: true
$.aggregated.doc.pdfe.destoutprofile.colorspace: true
$.aggregated.doc.pdfe.destoutprofile.name: true
$.aggregated.doc.pdfe.destoutprofile.profiletype: true
$.aggregated.doc.pdfe.outputintentid: true
$.aggregated.doc.pdfe.type: true

$.aggregated.doc.pdfua: true
$.aggregated.doc.pdfua.type: true
```

```
$.aggregated.doc.pdfvt: true
$.aggregated.doc.pdfvt.type: true

$.aggregated.doc.pdfx: true
$.aggregated.doc.pdfx.destoutprofile: true
$.aggregated.doc.pdfx.destoutprofile.colorspace: true
$.aggregated.doc.pdfx.destoutprofile.name: true
$.aggregated.doc.pdfx.destoutprofile.profiletype: true
$.aggregated.doc.pdfx.outputintentid: true
$.aggregated.doc.pdfx.type: true

$.aggregated.doc.digsig: true
$.aggregated.doc.digsig.Filter: true
$.aggregated.doc.digsig.Name: true
$.aggregated.doc.digsig.SubFilter: true

$.aggregated.doc.potential_syntax_issues: true

$.aggregated.doc.rolemap: true

$.aggregated.doc.xmpmetadata: true
```

Click to copy

"doc" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.doc: true
```

Click to copy

when used for the PDF file for the ISO 32000-1 standard results in the following Quick Check output:

```
{
  "aggregated": {
    "doc": {
      "created" : "2008/04/11 10:52:58",
      "modified" : "2014/01/14 14:44:26",
      "create_id": "<46f0099e3a8f6898a663db430bf77fa7>",
      "modified_id": "<c881b9ff7675107b9882a7f79d2f8c5e>",
      "pdf_version" : "%PDF-1.6"
```

```
    "encryption": {
      "type" : "40-bit RC4",
      "openpassword" : false,
      "masterpassword" : true

    "annotations" : {
      "num" : 3909,
      "annot" : [
        {
          "unique_id" : 8,
          "pagenum" : 3,
          "bbox" : [70.679, 715.241, 558.239, 726.761],
          "type" : "Link"
        },
        [... remaining entries omitted ...]

      "xmpmetadata": {
      "http://ns.adobe.com/pdf/1.3/": {
        "pdf:Producer": "Acrobat Distiller 6.0.1 (Windows)"

      "potential_syntax_issues": [
      "No_Header"
  }
}
```


Click to copy

3. "embeddedfiles" area

Creates output reflecting the files embedded in the PDF (as represented in the **EmbeddedFiles** name tree of the PDF file). The following properties are reported for each embedded file:

- `af_relationship`: PDF/A-3 requires the AFRelationship entry in each case. It specifies the nature of the relationship between the PDF and the related content. The Predefined values for relationships for associated files are: *Source, Data, Alternative, Supplement, Unspecified*
- `description`: if a relationship type is selectable, a description can be added
- `bytes`: file size of the embedded file in Bytes. Note: embedded files are typically compressed inside the PDF file,

and use less space inside the PDF file than they would once extracted again to a file system.
- `created`: creation date of the embedded file (typically based on the creation of that file in the file system at the time the file was embedded into the PDF)
- `last_modified`: last modification date of the embedded file (typically based on the last modification of that file in the file system at the time the file was embedded into the PDF)
- `name`: file name of the embedded file
- `length`: number of the embedded files

Filter expressions (for use in the config file)

```
$.aggregated.embeddedfiles: true
$.aggregated.embeddedfiles.embeddedfile: true
$.aggregated.embeddedfiles.embeddedfile.af_relationship: true
$.aggregated.embeddedfiles.embeddedfile.bytes: true
$.aggregated.embeddedfiles.embeddedfile.created: true
$.aggregated.embeddedfiles.embeddedfile.description: true
$.aggregated.embeddedfiles.embeddedfile.last_modified: true
$.aggregated.embeddedfiles.embeddedfile.name: true
$.aggregated.embeddedfiles.length: true
```

Click to copy

"embeddedfiles" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.embeddedfiles: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
        "aggregated" :
                        {
                                "embeddedfiles" :
```

```
                                {
                                        "embeddedfile" :
                                        [
                                                {
                                                        "af_relationship" :
"Source",
                                                        "bytes" : 157742,
                                                        "created" :
"D:20220525075610+02'00'",
                                                        "description" : "Here can
be a description",
                                                        "last_modified" :
"D:20220525075610+02'00'",
                                                        "name" : "demo_file.json"
                                                },
                                                {
                                                        "af_relationship" : "Data",
                                                        "bytes" : 595986,
                                                        "created" :
"D:20220511123438+02'00'",
                                                        "description" : "Here can
be a description",
                                                        "last_modified" :
"D:20220511123438+02'00'",
                                                        "name" : "demo_file.png"
                                                },
                                                {
                                                        "af_relationship" : "Alter-
native",
                                                        "bytes" : 396,
                                                        "created" :
"D:20220602095353+02'00'",
                                                        "description" : "Here can
be a description",
                                                        "last_modified" :
"D:20220602095353+02'00'",
                                                        "name" : "demo_file.txt"
                                                },
                                                {
                                                        "af_relationship" : "Un-
specified",
                                                        "bytes" : 20511,
                                                        "created" :
```

```
"D:20220602095552+02'00'",
                                                    "last_modified" :
"D:20220602095552+02'00'",
                                                    "name" : "demo_file.xlsx"
                                                }
                                            ],
                                            "length" : 4
                                        }
                                }
}
```

Click to copy

4. "env" area

Creates output reflecting various aspects of the environment in which Quick Check has been executed.

- `job_id` : job ID (empty unless explicitly if provided for the execution of Quick Check)
- `machine_name` : name of the machine on which Quick Check was executed
- `pdft_uuid` : a universally unique ID for the Quick Check execution instance
- `platform` : platform on which Quick Check was executed
- `process_id` : platform specific process ID of Quick Check instance
- `program_name` : name of the program executing Quick Check
- `program_version` : version number of the program executing Quick Check
- `timestamp` : date and time at which Quick Check was executed
- `timestamp_day` : day portion of the date at which Quick Check was executed
- `timestamp_hour` : hour portion of the time at which Quick Check was executed
- `timestamp_month` : month portion of the date at which Quick Check was executed
- `timestamp_weekday` : weekday portion of the date at which Quick Check was executed

- `verb`: output is always "quickcheck", serves as ID for the JSON report

Filter expressions (for use in the config file)

```
$.aggregated.env: true
$.aggregated.env.job_id: true
$.aggregated.env.machine_name: true
$.aggregated.env.pdft_uuid: true
$.aggregated.env.platform: true
$.aggregated.env.process_id: true
$.aggregated.env.program_name: true
$.aggregated.env.program_version: true
$.aggregated.env.timestamp: true
$.aggregated.env.timestamp_day: true
$.aggregated.env.timestamp_hour: true
$.aggregated.env.timestamp_month: true
$.aggregated.env.timestamp_weekday: true
$.aggregated.env.verb: true
```

Click to copy

"env" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.env: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
  "aggregated": {
    "env": {
      "verb" : "quickcheck",
      "pdft_uuid" : "494aa478-a93e-4c5b-882e-b0f0238713fa",
      "timestamp" : "2018/03/29 00:46:22",
      "timestamp_hour" : 0,
      "timestamp_month" : 3,
      "timestamp_day" : 29,
```

```
      "timestamp_weekday" : "Thursday",
      "process_id" : 7127,
      "program_name" : "",
      "program_version" : "",
      "platform" : "Mac OS X 10.12.6",
      "machine_name" : "pdfToolbox 10 demo machine",
      "job_id" : ""
    }
  }
```

Click to copy

5. "file" area

Creates output reflecting various aspects of the PDF file on the file system level.

- `bytes` : file size in Bytes
- `created` : creation time/date of the file
- `modified` : last modification time/date of the file
- `name` : file name
- `path` : path to the folder where the PDF file is stored
- `filepath` : path and file name

Filter expressions (for use in the config file)

```
$.aggregated.file: true
$.aggregated.file.bytes: true
$.aggregated.file.created: true
$.aggregated.file.modified: true
$.aggregated.file.name: true
$.aggregated.file.path: true
$.aggregated.file.filepath: true
```

Click to copy

"file" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.file: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
  "aggregated": {
    "file": {
      "bytes" : 1252256,
      "created" : "2018/03/29 00:27:19",
      "modified" : "2018/03/29 00:27:19",
      "name" : "demo.pdf",
      "path" : "/pdfToolbox 10 Demo",
      "filepath" : "/pdfToolbox 10 Demo/demo.pdf"
    }
  }
```

Click to copy

6. "ocgs" area

Creates output reflecting the layers in the PDF file (if any are present). In PDF syntax layers are represented using data structures named OCG (optional content group).

The information about each layer consists of three entries:

- `length` : number of layers
- `gui` : whether the layer would be listed in the user interface of a PDF viewing application (i.e. it is included in the **Orders** array of the default optional content configuration dictionary)
- `name` : name of the layer
- `visible` : whether the content belonging to this layer is set to be visible or not
- `processing_steps` : information about processing steps (ISO 19593)

Filter expressions (for use in the config file)

```
$.aggregated.ocgs: true
$.aggregated.ocgs.length: true
$.aggregated.ocgs.ocg: true
$.aggregated.ocgs.ocg.gui: true
```

```
$.aggregated.ocgs.ocg.name: true
$.aggregated.ocgs.ocg.visible: true
$.aggregated.ocgs.ocg.processing_steps: true
$.aggregated.ocgs.ocg.processing_steps.group: true
$.aggregated.ocgs.ocg.processing_steps.type: true
```

Click to copy

"ocgs" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.ocgs: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
                        "aggregated" :
                        {
                                "ocgs" :
                                {
                                        "length" : 3,
                                        "ocg" :
                                        [
                                                {
                                                        "gui" : true,
                                                        "name" : "Content",
                                                        "visible" : true
                                                },
                                                {
                                                        "gui" : true,
                                                        "name" : "Dieline",
                                                        "processing_steps" :
                                                        {
                                                                "group" : "Struc-
tural",
                                                                "type" : "Cutting"
                                                        },
                                                        "visible" : true
                                                },
```

```
                                        {
                                                "gui" : true,
                                                "name" : "Varnish",
                                                "processing_steps" :
                                                {
                                                        "group" : "Varnish"
                                                },
                                                "visible" : true
                                        }
                                ]
                        }
                }
```


Click to copy

7. "pages" area

The `pages` area can collect information about page geometry boxes, transparency groups, userunit and content stream. These sub-areas are explained individually below.

To request all information for this area, the following filter expression can be used:

```
$.aggregated.pages: true
```

Click to copy

7.1 "pages" area – aggregated information about page geometry boxes

Under the `pages` area, not only the list of pages in the form of the `page` array is available, but also several other sub-areas that cover various aspects of aggregated information about page geometry for the pages in the PDF document.

For each type of page geometry box (e.g. CropBox or TrimBox), a substructure is used to convey information about all page geometry boxes of that type in the given PDF document.

The meaning of each of the entries is as follows:

- `num` : number of times this page geometry is explicitly specified
- `num_portrait` : number of occurrences the page geometry box reflect a portrait orientation
- `num_square` : number of occurrences where the width and the height of the page geometry box are the same
- `num_landscape` : number of occurrences the page geometry box reflect a landscape orientation
- `width_min` : smallest width for this page geometry box in the current PDF document
- `width_max` : largest width for this page geometry box in the current PDF document
- `height_min` : smallest height for this page geometry box in the current PDF document
- `height_max` : largest height for this page geometry box in the current PDF document

For all page geometry types there are additional sub-areas, which refer to the effective size. These provide more or less the same information as the other variants, except that the entries for the smallest and largest dimensions – `width_min`, `width_max`, `height_min` and `height_max` – take the *page scaling factor* (UserUnit) into account.

For the BleedBox, there are additional sub-areas availible.

Filter expressions (for use in the config file)

```
$.aggregated.pages.length: true

$.aggregated.pages.ArtBox: true
$.aggregated.pages.ArtBox.height_max: true
$.aggregated.pages.ArtBox.height_min: true
$.aggregated.pages.ArtBox.num: true
$.aggregated.pages.ArtBox.num_landscape: true
$.aggregated.pages.ArtBox.num_portrait: true
$.aggregated.pages.ArtBox.num_square: true
$.aggregated.pages.ArtBox.width_max: true
$.aggregated.pages.ArtBox.width_min: true

$.aggregated.pages.effective_ArtBox: true
$.aggregated.pages.effective_ArtBox.height_max: true
```

```
$.aggregated.pages.effective_ArtBox.height_min: true
$.aggregated.pages.effective_ArtBox.num: true
$.aggregated.pages.effective_ArtBox.num_portrait: true
$.aggregated.pages.effective_ArtBox.num_landscape: true
$.aggregated.pages.effective_ArtBox.num_square: true
$.aggregated.pages.effective_ArtBox.width_max: true
$.aggregated.pages.effective_ArtBox.width_min: true

$.aggregated.pages.BleedBox: true
$.aggregated.pages.BleedBox.bottom_max: true
$.aggregated.pages.BleedBox.bottom_min: true
$.aggregated.pages.BleedBox.height_max: true
$.aggregated.pages.BleedBox.height_min: true
$.aggregated.pages.BleedBox.inner_max: true
$.aggregated.pages.BleedBox.inner_min: true
$.aggregated.pages.BleedBox.left_max: true
$.aggregated.pages.BleedBox.left_min: true
$.aggregated.pages.BleedBox.num: true
$.aggregated.pages.BleedBox.num_landscape: true
$.aggregated.pages.BleedBox.num_portrait: true
$.aggregated.pages.BleedBox.num_square: true
$.aggregated.pages.BleedBox.outer_max: true
$.aggregated.pages.BleedBox.outer_min: true
$.aggregated.pages.BleedBox.right_max: true
$.aggregated.pages.BleedBox.right_min: true
$.aggregated.pages.BleedBox.top_max: true
$.aggregated.pages.BleedBox.top_min: true
$.aggregated.pages.BleedBox.width_max: true
$.aggregated.pages.BleedBox.width_min: true

$.aggregated.pages.effective_BleedBox: true
$.aggregated.pages.effective_BleedBox.height_max: true
$.aggregated.pages.effective_BleedBox.height_min: true
$.aggregated.pages.effective_BleedBox.num: true
$.aggregated.pages.effective_BleedBox.num_portrait: true
$.aggregated.pages.effective_BleedBox.num_landscape: true
$.aggregated.pages.effective_BleedBox.num_square: true
$.aggregated.pages.effective_BleedBox.width_max: true
$.aggregated.pages.effective_BleedBox.width_min: true

$.aggregated.pages.CropBox: true
$.aggregated.pages.CropBox.height_max: true
$.aggregated.pages.CropBox.height_min: true
```

```
$.aggregated.pages.CropBox.num: true
$.aggregated.pages.CropBox.num_landscape: true
$.aggregated.pages.CropBox.num_portrait: true
$.aggregated.pages.CropBox.num_square: true
$.aggregated.pages.CropBox.width_max: true
$.aggregated.pages.CropBox.width_min: true

$.aggregated.pages.effective_CropBox: true
$.aggregated.pages.effective_CropBox.height_max: true
$.aggregated.pages.effective_CropBox.height_min: true
$.aggregated.pages.effective_CropBox.num: true
$.aggregated.pages.effective_CropBox.num_landscape: true
$.aggregated.pages.effective_CropBox.num_portrait: true
$.aggregated.pages.effective_CropBox.num_square: true
$.aggregated.pages.effective_CropBox.width_max: true
$.aggregated.pages.effective_CropBox.width_min: true

$.aggregated.pages.TrimBox: true
$.aggregated.pages.TrimBox.height_max: true
$.aggregated.pages.TrimBox.height_min: true
$.aggregated.pages.TrimBox.num: true
$.aggregated.pages.TrimBox.num_landscape: true
$.aggregated.pages.TrimBox.num_portrait: true
$.aggregated.pages.TrimBox.num_square: true
$.aggregated.pages.TrimBox.width_max: true
$.aggregated.pages.TrimBox.width_min: true

$.aggregated.pages.effective_TrimBox: true
$.aggregated.pages.effective_TrimBox.height_max: true
$.aggregated.pages.effective_TrimBox.height_min: true
$.aggregated.pages.effective_TrimBox.num: true
$.aggregated.pages.effective_TrimBox.num_landscape: true
$.aggregated.pages.effective_TrimBox.num_portrait: true
$.aggregated.pages.effective_TrimBox.num_square: true
$.aggregated.pages.effective_TrimBox.width_max: true
$.aggregated.pages.effective_TrimBox.width_min: true

$.aggregated.pages.MediaBox: true
$.aggregated.pages.MediaBox.height_max: true
$.aggregated.pages.MediaBox.height_min: true
$.aggregated.pages.MediaBox.num: true
$.aggregated.pages.MediaBox.num_landscape: true
$.aggregated.pages.MediaBox.num_portrait: true
```

```
$.aggregated.pages.MediaBox.num_square: true
$.aggregated.pages.MediaBox.width_max: true
$.aggregated.pages.MediaBox.width_min: true

$.aggregated.pages.effective_MediaBox: true
$.aggregated.pages.effective_MediaBox.height_max: true
$.aggregated.pages.effective_MediaBox.height_min: true
$.aggregated.pages.effective_MediaBox.num: true
$.aggregated.pages.effective_MediaBox.num_landscape: true
$.aggregated.pages.effective_MediaBox.num_portrait: true
$.aggregated.pages.effective_MediaBox.num_square: true
$.aggregated.pages.effective_MediaBox.width_max: true
$.aggregated.pages.effective_MediaBox.width_min: true
```

Click to copy

"pages" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.pages: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
  "aggregated": {
    "pages": {
      "length" : 4,
      "artbox" : {
        "num" : 0
      },
      "trimbox" : {
        "num" : 4,
        "num_portrait" : 4,
        "num_square" : 0,
        "num_landscape" : 0,
        "width_min" : 425.197,
        "width_max" : 425.197,
        "height_min" : 651.968,
        "height_max" : 651.968
```

```
    },
    "effective_trimbox" : {
      "num" : 4,
      "num_portrait" : 4,
      "num_square" : 0,
      "num_landscape" : 0,
      "width_min" : 425.197,
      "width_max" : 425.197,
      "height_min" : 651.968,
      "height_max" : 651.968
    },
    "bleedbox" : {
      "num" : 4,
      "num_portrait" : 4,
      "num_square" : 0,
      "num_landscape" : 0,
      "width_min" : 442.205,
      "width_max" : 442.205,
      "height_min" : 668.976,
      "height_max" : 668.976,
      "left_min" : -19.8425,
      "left_max" : 8.50398,
      "top_min" : 8.50396,
      "top_max" : 688.819,
      "right_min" : 8.50399,
      "right_max" : 462.047,
      "bottom_min" : -19.8425,
      "bottom_max" : 8.50398,
      "outer_min" : -19.8425,
      "outer_max" : 8.50398,
      "inner_min" : -19.8425,
      "inner_max" : 8.50398
    },
    "cropbox" : {
      "num" : 4,
      "num_portrait" : 4,
      "num_square" : 0,
      "num_landscape" : 0,
      "width_min" : 425.197,
      "width_max" : 425.197,
      "height_min" : 651.968,
      "height_max" : 651.968
    },
```

```
      "effective_cropbox" : {
        "num" : 4,
        "num_portrait" : 4,
        "num_square" : 0,
        "num_landscape" : 0,
        "width_min" : 425.197,
        "width_max" : 425.197,
        "height_min" : 651.968,
        "height_max" : 651.968
      },
      "mediabox" : {
        "num" : 4,
        "num_portrait" : 4,
        "num_square" : 0,
        "num_landscape" : 0,
        "width_min" : 481.89,
        "width_max" : 481.89,
        "height_min" : 708.661,
        "height_max" : 708.661
      }
    }
  }
```


Click to copy

7.2 "pages.contentstream" area – aggregated information about content stream size

Creates output reflecting the size of the content stream (in bytes). "contentstream.size" is the size of the content stream as encountered in the PDF before any decompression or de-coding. The content stream size in the output is of the raw data, or in other words the not compressed stream is used. The meaning of the entries is as follows:

- `size_max` : reflects the largest content stream size in the PDF
- `size_min` : reflects the smallest content stream size in the PDF
- `size_total` : reflects the combined size of the content stream size of all pages in the PDF
- `size_wfx_max` : reflects the size of the largest content stream plus the size of the largest existing Form XObject

content streams (where wfx stands for 'with Form XObject') on a page in a PDF
- `size_wfx_min`: reflects the size of the smallest content stream plus the size of the smallest existing Form XObject content streams on a page in a PDF
- `size_wfx_total`: reflects the combined value of the content streams length plus the length of any existing Form XObject content streams of all pages in the PDF

Filter expressions (for use in the config file)

```
$.aggregated.pages.contentstream: true
$.aggregated.pages.contentstream.size_max: true
$.aggregated.pages.contentstream.size_min: true
$.aggregated.pages.contentstream.size_total: true
$.aggregated.pages.contentstream.size_wfx_max: true
$.aggregated.pages.contentstream.size_wfx_min: true
$.aggregated.pages.contentstream.size_wfx_total: true
```

Click to copy

"pages" example for content stream size: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.pages.contentstream: true
```

Click to copy
on a sample PDF lists the output as below:

```
{
        "vars" :
        {
                "Quick_check" :
                {
                        "app_vars_sub_path" : "quickcheck_1",
                        "quickcheck_config" :
                        [
                                "$.direct: false",
                                "$.aggregated: false",
```

```
                                "$.aggregated.pages.contentstream: true"
                        ]
                },
                "quickcheck_1" :
                {
                        "aggregated" :
                        {
                                "pages" :
                                {
                                        "contentstream" :
                                        {
                                                "size_max" : 10990,
                                                "size_min" : 9712,
                                                "size_total" : 156708,
                                                "size_wfx_max" : 11341,
                                                "size_wfx_min" : 10110,
                                                "size_wfx_total" : 166743
                                        }
                                }
                        }
}
```

Click to copy

7.3 "pages.userunit"

The `userunit` is a number reflecting the page scaling factor to be applied to the page; smallest allowed and also the default value is 1.0. `pages.userunit` includes all pages contained in the PDF document. There are three entries for this area:

- `max`: highest userunit value fond across all pages
- `min`: lowest userunit value fond across all pages
- `num`: number of page for which a userunit entries present

Filter expressions (for use in the config file)

```
$.aggregated.pages.userunit: true
$.aggregated.pages.userunit.max: true
$.aggregated.pages.userunit.min: true
```

```
$.aggregated.pages.userunit.num: true
```

Click to copy

"pages.userunit" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.pages.userunit: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
        "aggregated" :
                        {
                        "pages" :
                                {
                                        "userunit" :
                                        {
                                                "max" : 10,
                                                "min" : 2,
                                                "num" : 2
                                        }
                                }
                        }
}
```

Click to copy

7.4 "pages.transparency_group"

There are two entries for this area:

- `first_page`: specifies the page number of the page on which the first transparency group is located
- `num`: number of transparency groups

Filter expressions (for use in the config file)

```
$.aggregated.pages.transparency_group: true
$.aggregated.pages.transparency_group.first_page: true
$.aggregated.pages.transparency_group.num: true
```

Click to copy

"pages.transparency_group" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.pages.transparency_group: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
        "aggregated" :
                {
                        "pages" :
                                {
                                        "transparency_group" :
                                        {
                                                "first_page" : 2,
                                                "num" : 3
                                        }
                                }
                }
}
```

Click to copy

8. "resources" area: "color", "fonts", "images", "transparency" and "overprint"

Under the `resources` area four sub-areas can be requested:

- `transparency`
- `fonts`
- `overprint`
- `images`
- `color`

To request all information for this area, the following filter expression can be used:

```
$.aggregated.resources: true
```

Click to copy

8.1 "resources.transparency"

The resources.transparency sub-area can be used to convey information about the transparencies contained in the PDF:

- `blendmodes` : indicates which blend mode is used
- `groups` : information whether an isolated or non-isolated transparency group is included
- `has_explicit_transparency` : information whether transparencies are used via blendmodes, softmasks or softmasks in images.
- `has_transparency` : information whether transparencies are used via blendmodes, softmasks or softmasks in images or are included in a transparency group
- `opacity_less_than_1` : Information whether the opacity is less than 1
- `softmasks` : information whether soft masks are included

Filter expressions (for use in the config file)

```
$.aggregated.resources.transparency: true

$.aggregated.resources.transparency.blendmodes: true
$.aggregated.resources.transparency.blendmodes.color: true
$.aggregated.resources.transparency.blendmodes.colorburn: true
$.aggregated.resources.transparency.blendmodes.coloredge: true
$.aggregated.resources.transparency.blendmodes.darken: true
$.aggregated.resources.transparency.blendmodes.difference: true
$.aggregated.resources.transparency.blendmodes.exclusion: true
```

```
$.aggregated.resources.transparency.blendmodes.hardlight: true
$.aggregated.resources.transparency.blendmodes.hue: true
$.aggregated.resources.transparency.blendmodes.lighten: true
$.aggregated.resources.transparency.blendmodes.luminosity: true
$.aggregated.resources.transparency.blendmodes.multiply: true
$.aggregated.resources.transparency.blendmodes.non_normal: true
$.aggregated.resources.transparency.blendmodes.overlay: true
$.aggregated.resources.transparency.blendmodes.saturation: true
$.aggregated.resources.transparency.blendmodes.screen: true
$.aggregated.resources.transparency.blendmodes.softlight: true

$.aggregated.resources.transparency.groups: true
$.aggregated.resources.transparency.groups.isolated: true
$.aggregated.resources.transparency.groups.isolated_devicecmyk: true
$.aggregated.resources.transparency.groups.isolated_devicegray: true
$.aggregated.resources.transparency.groups.isolated_devicergb: true
$.aggregated.resources.transparency.groups.isolated_icc_cmyk: true
$.aggregated.resources.transparency.groups.isolated_icc_gray: true
$.aggregated.resources.transparency.groups.isolated_icc_rgb: true
$.aggregated.resources.transparency.groups.non_isolated: true
$.aggregated.resources.transparency.groups.isolated_non_cmyk: true
$.aggregated.resources.transparency.groups.isolated_non_devicecmyk: true

$.aggregated.resources.transparency.has_explicit_transparency: true
$.aggregated.resources.transparency.has_transparency: true
$.aggregated.resources.transparency.opacity_less_than_1: true

$.aggregated.resources.transparency.softmasks.softmasks_any: true
$.aggregated.resources.transparency.softmasks.softmasks_in_extgstate: true
$.aggregated.resources.transparency.softmasks.softmasks_in_image: true
$.aggregated.resources.transparency.softmasks: true
```

Click to copy

"resources.transparency" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.resources.transparency: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output:


```
{
                        "aggregated" :
                        {
                                "resources" :
                                {
                                        "transparency" :
                                        {
                                                "blendmodes" :
                                                {
                                                        "color" : false,
                                                        "colorburn" : false,
                                                        "colordodge" : false,
                                                        "darken" : false,
                                                        "difference" : false,
                                                        "exclusion" : false,
                                                        "hardlight" : false,
                                                        "hue" : false,
                                                        "lighten" : false,
                                                        "luminosity" : false,
                                                        "multiply" : true,
                                                        "non_normal" : true,
                                                        "overlay" : false,
                                                        "saturation" : false,
                                                        "screen" : false,
                                                        "softlight" : false
                                                },
                                                "groups" :
                                                {
                                                        "isolated" : true,
                                                        "isolated_devicecmyk" :
true,
                                                        "isolated_devicegray" :
false,
                                                        "isolated_devicergb" :
false,
                                                        "isolated_icc_cmyk" :
false,
                                                        "isolated_icc_gray" :
false,
                                                        "isolated_icc_rgb" : false,
                                                        "isolated_non_cmyk" :
```

```
false,
                                                        "isolated_non_devicecmyk"
: false,
                                                        "non_isolated" : true
                                                    },
                                                    "has_explicit_transparency" : true,
                                                    "has_transparency" : true,
                                                    "opacity_less_than_1" : false,
                                                    "softmasks" :
                                                    {
                                                        "softmasks_any" : false,
                                                        "softmasks_in_extgstate" :
false,
                                                        "softmasks_in_image" :
false
                                                    }
                                                }
                                            }
                                        },
```


Click to copy

8.2 "resources.fonts"

Creates output reflecting font usage and contains two entries:

- `font` : an array of entries for each font
- `length` : number of fonts

Each `font` array contains the following entries:

- `name` : the name of the font
- `fonttype` : the font type, possible values are Type0, Type1, TrueType, Type3
- `embedded` : whether the font is embedded or not
- `subset` : whether the font is embedded as a subset or not; only meaningful if the font is actually embedded

Filter expressions (for use in the config file):

```
$.aggregated.resources.fonts: true,
$.aggregated.resources.fonts.font: true
```

```
$.aggregated.resources.fonts.font.embedded: true
$.aggregated.resources.fonts.font.fonttype: true
$.aggregated.resources.fonts.font.name: true
$.aggregated.resources.fonts.font.subset: true
$.aggregated.resources.fonts.length : true
```

Click to copy

"resources.fonts" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.resources.fonts: true
```

Click to copy

```
{
        "aggregated" :
                {
                        "resources" :
                        {
                                "fonts" :
                                        {
                                                "length" : 3,
                                                "font" :
                                                [
                                                        {
                                  "name" : "Helvetica",
                                  "subset" : true,
                                  "fonttype" : "TrueType",
                                  "embedded" : true
                                },
                                {
                                  "name" : "SourceSansPro-Regular",
                                  "subset" : true,
                                  "fonttype" : "Type1",
                                  "embedded" : true
                                },
                                {
                                  "name" : "SourceSansPro-Semibold",
                                  "subset" : true,
```

```
                                        "fonttype" : "Type1",
                                        "embedded" : true
                                    }
                                ]
                            }
                        }
                    }
}
```

Click to copy

8. 3 "resources.overprint"

Creates output reflecting the overprint parameters in a PDF file (if any are present). This sub-area has the following two entries:

- `uses_opm` : overprint mode parameter (OPM)
- `uses_overprint` : overprint parameter (OV)

Filter expressions (for use in the config file):

```
$.aggregated.resources.overprint: true
$.aggregated.resources.overprint.uses_opm: true
$.aggregated.resources.overprint.uses_overprint: true
```

Click to copy

"resources.overprint" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.resources.overprint: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
```

```
                "aggregated" :
                        {
                                "resources" :
                                {
                                        "overprint" :
                                        {
                                                "uses_opm" : true,
                                                "uses_overprint" : true
                                        }
                                }
                        }
}
```

Click to copy

8.4 "resources.images"

Under `images` you can find six different types of images which can appear in a PDF file:

- `bitmap_images`
- `bitmap_images_in_softmasks`
- `ct_images` (comtinuous tone)
- `ct_images_in_softmasks`
- `imagemasks`
- `imagemasks_in_softmasks`

All `images` types use the following entries:

- `highest_pixel_h` : indicates the pixel count of the image with the highest (horizontal) number of pixels
- `highest_pixel_v` : indicates the pixel count of the image with the highest (vertical) number of pixels
- `highest_ppi` : specifies the image resolution of the image with the highest image resolution
- `lowest_ppi` : specifies the image resolution of the image with the lowest image resolution
- `eff_highest_ppi` : specifies the image resolution of the image with the highest image resolution (takes the *page scaling factor* (**UserUnit**) into account)
- `eff_lowest_ppi` : specifies the image resolution of the image with the lowest image resolution (takes the *page scaling factor* (**UserUnit**) into account)

- `lowest_pixel_h`: indicates the pixel count of the image with the lowest (horizontal) number of pixels
- `lowest_pixel_v`: indicates the pixel count of the image with the lowest (vertical) number of pixels
- `number`: number of images for this image type
- `colorspaces`: an array listing the color spaces that are used in the images

Filter expressions (for use in the config file)

```
$.aggregated.resources.images: true
$.aggregated.resources.images.summary: true
$.aggregated.resources.images.summary.number: true
$.aggregated.resources.images.summary.bitmap_images: true
$.aggregated.resources.images.summary.bitmap_images.eff_highest_ppi: true
$.aggregated.resources.images.summary.bitmap_images.eff_lowest_ppi: true
$.aggregated.resources.images.summary.bitmap_images.highest_ppi: true
$.aggregated.resources.images.summary.bitmap_images.lowest_ppi: true
$.aggregated.resources.images.summary.bitmap_images.highest_pixel_h: true
$.aggregated.resources.images.summary.bitmap_images.highest_pixel_v: true
$.aggregated.resources.images.summary.bitmap_images.lowest_pixel_h: true
$.aggregated.resources.images.summary.bitmap_images.lowest_pixel_v: true
$.aggregated.resources.images.summary.bitmap_images.number: true

$.aggregated.resources.images.summary.bitmap_images_in_softmasks: true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.eff_highest_ppi:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.eff_lowest_ppi:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.highest_ppi: true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.lowest_ppi: true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.highest_pixel_h:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.lowest_pixel_h:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.highest_pixel_v:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.lowest_pixel_v:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.number: true

$.aggregated.resources.images.summary.ct_images: true
$.aggregated.resources.images.summary.ct_images.eff_highest_ppi: true
```

```
$.aggregated.resources.images.summary.ct_images.eff_lowest_ppi: true
$.aggregated.resources.images.summary.ct_images.highest_ppi: true
$.aggregated.resources.images.summary.ct_images.lowest_ppi: true
$.aggregated.resources.images.summary.ct_images.highest_pixel_h: true
$.aggregated.resources.images.summary.ct_images.lowest_pixel_h: true
$.aggregated.resources.images.summary.ct_images.highest_pixel_v: true
$.aggregated.resources.images.summary.ct_images.lowest_pixel_v: true
$.aggregated.resources.images.summary.ct_images.number: true

$.aggregated.resources.images.summary.ct_images_in_softmasks: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.eff_highest_ppi: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.eff_lowest_ppi: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.highest_ppi: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.lowest_ppi: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.highest_pixel_h: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.lowest_pixel_h: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.highest_pixel_v: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.lowest_pixel_v: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.number: true

$.aggregated.resources.images.summary.imagemasks: true
$.aggregated.resources.images.summary.imagemasks.eff_highest_ppi: true
$.aggregated.resources.images.summary.imagemasks.eff_lowest_ppi: true
$.aggregated.resources.images.summary.imagemasks.highest_ppi: true
$.aggregated.resources.images.summary.imagemasks.lowest_ppi: true
$.aggregated.resources.images.summary.imagemasks.highest_pixel_h: true
$.aggregated.resources.images.summary.imagemasks.lowest_pixel_h: true
$.aggregated.resources.images.summary.imagemasks.highest_pixel_v: true
$.aggregated.resources.images.summary.imagemasks.lowest_pixel_v: true
$.aggregated.resources.images.summary.imagemasks.number: true

$.aggregated.resources.images.summary.imagemasks_in_softmasks: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.eff_highest_ppi: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.eff_lowest_ppi: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.highest_ppi: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.lowest_ppi: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.highest_pixel_h: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.lowest_pixel_h: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.highest_pixel_v: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.lowest_pixel_v: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.number: true

$.aggregated.resources.images.summary.colorspaces: true
```

```
$.aggregated.resources.images.summary.colorspaces.length: true
$.aggregated.resources.images.summary.number: true
```

Click to copy

"resources.images" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.resources.images: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output:

```
{
                        "aggregated" :
                        {
                                "resources" :
                                {
                                        "images" :
                                        {
                                                "summary" :
                                                {
                                                        "colorspaces" :
                                                        {
                                                                "colorspace" : [
"DeviceRGB" ],
                                                                "length" : 1
                                                        },
                                                        "bitmap_images" :
                                                        {
                                                                "number" : 0
                                                        },
                                                        "bitmap_images_in_soft-
masks" :
                                                        {
                                                                "number" : 0
                                                        },
                                                        "ct_images" :
```

```
                                                        {
                                                                "eff_highest_ppi"
: 72,
                                                                "eff_lowest_ppi" :
33.982900000000001,
                                                                "highest_pixel_h"
: 2742,
                                                                "highest_pixel_v"
: 3265,
                                                                "highest_ppi" :
691.09199999999998,
                                                                "lowest_pixel_h" :
286,
                                                                "lowest_pixel_v" :
343,
                                                                "lowest_ppi" : 72,
                                                                "number" : 3
                                                        },
                                                        "ct_images_in_softmasks" :
                                                        {
                                                                "number" : 0
                                                        },
                                                        "imagemasks" :
                                                        {
                                                                "number" : 0
                                                        },
                                                        "imagemasks_in_softmasks"
:
                                                        {
                                                                "number" : 0
                                                        }
                                                }
                                        }
                                }
                        }
}
```


Click to copy

8.5 "resources.color"

The resources.color sub-area can be used to convey information about the colorspaces, spot colors and color plates con-

tained in the PDF file. The "color" area contains four sub-areas:

- `summary`: a list of entries where each represents a certain aggregated aspect of color usage; for example, if `DeviceCMYK` has a value of 0, there is no graphics object on the page that uses DeviceCMYK (but there might be a graphics object that uses DeviceN with the colorants Cyan, Magenta, Yellow, Black, or ICC based CMYK). In comparison, `Any_CMYK` reports any use of CMYK, whether DeviceCMYK, ICC based CMYK, DeviceN with one, several or all of the Cyan, Magenta, Yellow, Black colorants, or Separation color space Cyan, Magenta, Yellow, Black.
- `colorspaces`: a list of entries (only those are shown that are applicable) reflecting the presence of certain color spaces.
- `spotcolors`: a list of entries for spot colors used, including their name and the alternate color space used. together with associated color values for 100% tint value of the spot color
- `color_plates`: a list of color plates on each page and for the whole document. This is not the exact but the maximum possible number of color separations.

To request all information for these areas, the following filter expression can be used:

```
$.aggregated.resources.color: true
```

Click to copy

"resources.color.summary"

All entries under `summary` have an integer as value, which indicates how often the respective color type is used. In the following, each entry is classified into a color space group and briefly described:

Device color spaces (specify colors or shades of gray that the output device is to produce):

- `Any_Device`: use of any device color space, whether **DeviceGray**, **DeviceRGB**, **DeviceCMYK**, **Separation** or **DeviceN**

- `DeviceCMYK` : use of **DeviceCMYK**
- `DeviceGray` : use of **DeviceGray**
- `DeviceRGB` : use of **DeviceRGB**

CIE-based color spaces (based on an international standard for color specification):

- `Any_Calibrated` : use of any calibrated color space whether any **ICC based** color space, **Lab**, **CalGray** or **CalRGB**
- `CalGray` : use of **CalGray**
- `CalRGB` : use of **CalRGB**
- `Calibrated_RGB` : use of **CalRGB** or **ICC based RGB**
- `ICCBased_CMYK` : use of **ICC based CMYK**
- `ICCBased_Gray` : use of **ICC based gray**
- `ICCBased_Lab` : use of **ICC based Lab**
- `ICCBased_RGB` : use of **ICC based RGB**
- `Lab` : use of **Lab** colorspace

Default color space

Device-dependent colors can be remapped into a device-independent CIE-based color space by setting a `Default` color space in the resources. Any color space (except Lab, Indexed, or Pattern) can be used as a default color space provided that it is compatible with the original device color space:

- `Any_Default` : use of Default color space
- `Default_CMYK` : lists **DeviceCMYK** which are overwritten by a DefaultCMYK
- `Default_RGB` : lists **DeviceRGB** which are overwritten by a DefaultRGB
- `Default_Gray` : lists **DeviceGray** which are overwritten by a DefaultGray
- `Default_CalGray` : use of a Default color space that is **CalGray**
- `Default_CalRGB` : use of a Default color space that is **CalRGB**
- `Default_DeviceN_All_Of_CMYK_No_Spot` : use of Default color space that is **DeviceN** with all four colorants Cyan, Magenta Yellow or Black, but no spot color (one or more colorants None might also be present)
- `Default_ICCBased_CMYK` : use of a Default color space that is **ICCBased CMYK**

- `Default_ICCBased_Gray` : use of a Default color space that is **ICCBased Gray**
- `Default_ICCBased_RGB` : use of a Default color space that is **ICCBased RGB**

Special color spaces (add features or properties to an underlying color space):

When printing a page, some devices produce a separate rendition of the page, called a `separation` for each colorant (process colorants and spot colorants):

- `Any_Separation` : use of any **Separation** color space, whether a spot color, or a colorant name that is Cyan, Magenta, Yellow, Black, None or All
- `Separation_All` : use of **Separation** All (also often referred to as registration color)
- `Separation_Any_Of_CMYK` : use of **Separation** color space with any of the colorants **Cyan, Magenta Yellow or Black**
- `Separation_Any_Spot` : use of a **Separation** color space for a **spot color**
- `Separation_Black` : use of **Separation Black**
- `Separation_Cyan` : use of **Separation Cyan**
- `Separation_Magenta` : use of **Separation Magenta**
- `Separation_None` : use of **Separation None** (any object using Separation None will not be rendered)
- `Separation_Yellow` : use of **Separation Yellow**

A `DeviceN` color space can contain an arbitrary number of different color components into one color space.

- `Any_DeviceN` : use of **DeviceN**
- `DeviceN_All_Of_CMYK` : use of **DeviceN** where all four colorants Cyan, Magenta Yellow or Black, either with or without additional colorants (spot colorants or None)
- `DeviceN_All_Of_CMYK_And_Spot` : use of **DeviceN** where all four colorants Cyan, Magenta Yellow or Black, but also at least one spot color (one or more colorants None might also be present)
- `DeviceN_All_Of_CMYK_No_Spot` : use of **DeviceN** where all four colorants Cyan, Magenta Yellow or Black, but no spot color (one or more colorants None might also be present)

- `DeviceN_All_Of_Spot` : use of **DeviceN** where all colorants are spot colors (one or more colorants None might also be present)
- `DeviceN_Any_Of_CMYK` : use of **DeviceN** where at least one colorants has a name that is Cyan, Magenta Yellow or Black
- `Pattern` : use of **patterns**
- `Smooth_Shades` : use of **smooth shades**

Color space summary:

- `Any_CMYK` : any use of the colorants Cyan, Magenta, Yellow or Black, whether by means of **DeviceCMYK**, **ICC based CMYK**, **DeviceN** with one or several of the four colorants or **Separation** color spaces using one of the four colorants
- `Any_Gray` : use of any gray colorspace, whether **DeviceGray**, **ICC based** gray or **CalGray**
- `Any_RGB` : any use of RGB, whether by means of **DeviceRGB**, **ICC based RGB** or **CalRGB**
- `Any_Spot` : use of any spot color, whether **Separation** color space with a colorant name other than Cyan, Magenta, Yellow, Black, None or All, or **DeviceN** with at least one colorant with a colorant name other than Cyan, Magenta, Yellow, Black or None
- `Not_DeviceCMYK` : any use of a color space that is **not DeviceCMYK**
- `Not_DeviceCMYK_Or_Spot` : any use of a color space that is **not DeviceCMYK** or a **spot color**

Filter expressions (for use in the config file):

```
$.aggregated.resources.color.summary: true
```

Click to copy

"resources.color.summary" example: Filter expression and output

```
$.direct: false
```

```
$.aggregated: false
$.aggregated.resources.color.summary: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output:


```
{
                        "aggregated" :
                        {
                                "resources" :
                                {
                                        "color" :
                                        {
                                                "summary" :
                                                {
                                                        "Any_CMYK" : 5,
                                                        "Any_Calibrated" : 6,
                                                        "Any_Default" : 0,
                                                        "Any_Device" : 7,
                                                        "Any_DeviceN" : 7,
                                                        "Any_Gray" : 4,
                                                        "Any_RGB" : 3,
                                                        "Any_Separation" : 6,
                                                        "Any_Spot" : 0,
                                                        "CalGray" : 0,
                                                        "CalRGB" : 0,
                                                        "Calibrated_RGB" : 1,
                                                        "Default_CMYK" : 0,
                                                        "Default_CalGray" : 0,
                                                        "Default_CalRGB" : 0,
                                                        "Default_Devi-
ceN_All_Of_CMYK_No_Spot" : 0,
                                                        "Default_Gray" : 0,
                                                        "Default_ICCBased_CMYK" :
0,
                                                        "Default_ICCBased_Gray" :
0,
                                                        "Default_ICCBased_RGB" : 0,
                                                        "Default_RGB" : 0,
                                                        "DeviceCMYK" : 3,
                                                        "DeviceGray" : 2,
                                                        "DeviceN_All_Of_CMYK" : 0,
```

```
                                                                "Devi-
ceN_All_Of_CMYK_And_Spot" : 0,
                                                                "Devi-
ceN_All_Of_CMYK_No_Spot" : 0,
                                                                "DeviceN_All_Of_Spot" : 7,
                                                                "DeviceN_Any_Of_CMYK" : 6,
                                                                "DeviceRGB" : 2,
                                                                "ICCBased_CMYK" : 2,
                                                                "ICCBased_Gray" : 2,
                                                                "ICCBased_Lab" : 0,
                                                                "ICCBased_RGB" : 1,
                                                                "Lab" : 1,
                                                                "Not_DeviceCMYK" : 16,
                                                                "Not_DeviceCMYK_Or_Spot" :
4,
                                                                "Pattern" : 0,
                                                                "Separation_All" : 2,
                                                                "Separation_Any_Of_CMYK" :
1,
                                                                "Separation_Any_Spot" : 5,
                                                                "Separation_Black" : 0,
                                                                "Separation_Cyan" : 0,
                                                                "Separation_Magenta" : 1,
                                                                "Separation_None" : 1,
                                                                "Separation_Yellow" : 0,
                                                                "Smooth_Shades" : 0
                                                        }
                                                }
                                        }
                                }
```


Click to copy

"resources.color.colorspaces"

Under `colorspaces` four entries can be found:

- `colorspaces`: an array listing the color spaces used
- `length`: the number of color spaces listed in the color-spaces array
- `icc_source_profiles`: if the PDF document contains an output condition, it is listed here

- `default_colorspaces`: like `colorspaces`, `default colorspaces` have the same entries in its array as mentioned below

The entries in the `colorspaces` arrays can be any of the following:

- `DeviceCMYK`: DeviceCMYK was used at least once
- `ICCBased`: an ICC based color space was used at least once
- `ICCBased_CMYK`: an ICC based CMYK color space was used at least once
- `Separation`: a Separation color space was used at least once
- `Separation_Spot`: a Separation color space with a spot colorant was used at least once
- `Separation_CMYK`: a Separation color space with a colorant whose name is Cyan, Magenta, Yellow or Black was used at least once
- `Separation_All`: a Separation All (registration) color space was used at least once
- `Separation_None`: a Separation None color space (graphics object using this color space will not be rendered) was used at least once
- `DeviceN`: DeviceN was used at least once
- `DeviceN_SpotOnly`: DeviceN using spot color but none of the CMYK colorants was used at least once
- `DeviceN_Spot_CMYK`: DeviceN using a combination of spot and CMYK colorants was used at least once
- `DeviceN_CMYK`: DeviceN using only CMYK, but not spot colorants was used at least once

Filter expressions (for use in the config file):

```
$.aggregated.resources.color: true
$.aggregated.resources.color.colorspaces: true
$.aggregated.resources.color.colorspaces.colorspace: true
$.aggregated.resources.color.colorspaces.icc_source_profiles: true
$.aggregated.resources.color.colorspaces.length: true
$.aggregated.resources.color.default_colorspaces: true
$.aggregated.resources.color.default_colorspaces.colorspace: true
$.aggregated.resources.color.default_colorspaces.icc_source_profiles: true
$.aggregated.resources.color.default_colorspaces.length: true
```

Click to copy

"resources.color.colorspaces" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.resources.color: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
                        "aggregated" :
                        {
                                "resources" :
                                {
                                        "color" :
                                        {
                                                "colorspaces" :
                                                {
                                                        "colorspace" :
                                                        [
                                                                "DeviceCMYK",
                                                                "ICCBased",
                                                                "ICCBased_CMYK",
                                                                "Separation",
                                                                "Separation_Spot"
                                                        ],
                                                        "icc_source_profiles" : [
"Europe ISO Coated FOGRA27" ],
                                                        "length" : 5
                                                }
                                        }
                                }
                        }
}
```

Click to copy

"resources.color.spotcolors"

Under `spotcolors` two entries can be found:

- `spotcolor`: an array listing the spot colors used
- `length`: the number of spot colors listed in the spotcolor array

The entries in the `spotcolor` array list each spot color using the following entries:

- `name`: name of the spot color
- `alternatespace`: alternate space for the spot color; can be *DeviceCMYK, ICCBased_CMYK, DeviceRGB, CalRGB, ICCBased_RGB, Lab, DeviceGray, CalGray, ICCBased_Gray* or *undefined*; undefined occurs in cases where a DeviceN color space includes a spot color, but no alternate space for that spot color is provided (and only for the DeviceN color space as a whole an alternate space is provided)
- `alternatevalues`: an array of values that when used on the background of the alternate space emulate the appearance of a 100% tint value of the spot color

Filter expressions (for use in the config file):

```
$.aggregated.resources.color.spotcolors: true
$.aggregated.resources.color.spotcolors.length: true
$.aggregated.resources.color.spotcolors.spotcolor: true
$.aggregated.resources.color.spotcolors.spotcolor.alternatespace: true
$.aggregated.resources.color.spotcolors.spotcolor.alternatevalues: true
$.aggregated.resources.color.spotcolors.spotcolor.name: true
```

Click to copy

"resources.color.spotcolors" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.resources.color.spotcolors: true
```

Click to copy

when used for a single page demo PDF file results in the following Quick Check output:


```
{
        "aggregated" :
        {
                "resources" :
                        {
                                "color" :
                                {
                                        "spotcolors" :
                                        {
                                  "length" : 3,
                                  "spotcolor" :
                                                [
                                                  {
                                                    "name" : "ImgMaskColor_1",
                                                    "alternatespace" : "DeviceRGB",
                                                    "alternatevalues" : [0.000000, 0.
000000, 1.000000]
                                                  },
                                                  {
                                                    "name" : "ImgMaskColor_2",
                                                    "alternatespace" : "DeviceRGB",
                                                    "alternatevalues" : [1.000000, 0.
882353, 0.000000]
                                                  },
                                                  {
                                                    "name" : "ImgMaskColor_3",
                                                    "alternatespace" : "DeviceRGB",
                                                    "alternatevalues" : [1.000000, 0.
000000, 0.000000]
                                                  }
                                  ]
                                }
                        }
                }
        }
},
```


Click to copy

"resources.color.color_plates"

Creates a list of color plates for the whole document. "color_plates" contains two entires:

- `length` : number of color plates
- `plates` : an array of entries for each color plate

Filter expressions (for use in the config file):

```
$.aggregated.resources.color.color_plates: true
$.aggregated.resources.color.color_plates.length: true
$.aggregated.resources.color.color_plates.plates: true
```

Click to copy

"resources.color.color_plates" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.resources.color.color_plates: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
                        "aggregated" :
                        {
                                "resources" :
                                {
                                        "color" :
                                        {
                                                "color_plates" :
                                                {
                                                        "length" : 6,
                                                        "plates" : [ "Black",
"Cyan", "Dieline", "Magenta", "Varnish", "Yellow" ]
                                                }
```

```
                                        }
                                    }
                                }
}
```

Click to copy

8.6 "resources.form_xobjects"

This filter expression lists PDF resources which are not redefined within a Form X Object, but are used when drawing the Form X Object content stream. This resources may have a direct effect on the rendering of the Form X Object. Examples of affected PDF resources:

- `fill_colorspace`
- `text_font`
- `line_width`
- `overprint`
- `rendering_intent`
- `blend_mode`
- etc.

Filter expressions (for use in the config file):

```
$.aggregated.resources.form_xobjects: true
$.aggregated.resources.form_xobjects.inherited_resources: true
```

Click to copy

9. multiple_resource_refs_across_pages

A summary across all pages related to each resource type used in the document: colorspace, font, pattern, objects, shade etc.

Filter expressions (for use in the config file):

```
$.aggregated.multiple_resource_refs_across_pages: true
$.aggregated.multiple_resource_refs_across_pages.colorspace: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_alt: true
```

```
$.aggregated.multiple_resource_refs_across_pages.colorspace_cie: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_device: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_icc: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_indexed: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_inside_cs: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_inside_image: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_inside_shade: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_separation_devicen:
true
$.aggregated.multiple_resource_refs_across_pages.colorspace_special: true
$.aggregated.multiple_resource_refs_across_pages.extgstate: true
$.aggregated.multiple_resource_refs_across_pages.font: true
$.aggregated.multiple_resource_refs_across_pages.font_cid: true
$.aggregated.multiple_resource_refs_across_pages.font_descriptor: true
$.aggregated.multiple_resource_refs_across_pages.font_encoding: true
$.aggregated.multiple_resource_refs_across_pages.font_simple: true
$.aggregated.multiple_resource_refs_across_pages.pattern: true
$.aggregated.multiple_resource_refs_across_pages.pattern_shading: true
$.aggregated.multiple_resource_refs_across_pages.pattern_tiling: true
$.aggregated.multiple_resource_refs_across_pages.procset: true
$.aggregated.multiple_resource_refs_across_pages.properties: true
$.aggregated.multiple_resource_refs_across_pages.resource_dictionary: true
$.aggregated.multiple_resource_refs_across_pages.shade: true
$.aggregated.multiple_resource_refs_across_pages.xobject: true
$.aggregated.multiple_resource_refs_across_pages.xobject_forms: true
$.aggregated.multiple_resource_refs_across_pages.xobject_image: true
```

Click to copy

"multiple_resource_refs_across_pages" example: Filter expression and output

```
$.direct: false
$.aggregated: false
$.aggregated.multiple_resource_refs_across_pages: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output:

```
{
```

```
        "aggregated" :
                {
                        "multiple_resource_refs_across_pages" :
                                {
                                        "summary" :
                                        {
                                                "any" : 4,
                                                "colorspace" : 1,
                                                "colorspace_alt" : 0,
                                                "colorspace_cie" : 1,
                                                "colorspace_device" : 1,
                                                "colorspace_icc" : 1,
                                                "colorspace_indexed" : 0,
                                                "colorspace_inside_cs" : 1,
                                                "colorspace_inside_image" : 0,
                                                "colorspace_inside_shade" : 0,
                                                "colorspace_separation_devicen" :
0,
                                                "colorspace_special" : 0,
                                                "extgstate" : 2,
                                                "font" : 0,
                                                "font_cid" : 0,
                                                "font_descriptor" : 0,
                                                "font_encoding" : 0,
                                                "font_simple" : 0,
                                                "pattern" : 0,
                                                "pattern_shading" : 0,
                                                "pattern_tiling" : 0,
                                                "procset" : 0,
                                                "properties" : 0,
                                                "resource_dictionary" : 0,
                                                "shade" : 0,
                                                "xobject" : 0,
                                                "xobject_forms" : 0,
                                                "xobject_image" : 0
                                        }
                                }
                }
}
```


Click to copy

10. "pages.page" area

As mentioned, the Quick Check filter expressions also make it possible to request page related data for either a specific page or a specific page range. The page index is one based. This means that a particular page can be addressed using the usual array notation (the first element starts at 1). If no page index is added to a `$.aggregated.pages.page` filter expression, it will be applied to every page of the document.

```
$.aggregated.pages.page: true      (generates output for all pages of the document)
$.aggregated.pages.page[1]: true   (generates output only for the first page)
```

Click to copy

An example that shows all possible page index variants can be found here.

The pages.page area has several sub-areas:

1. "pages.page.annotations"

Creates output reflecting a handful of document properties regarding annotations and their properties like page number, type, unique id, URL etc. for a given page. A description of all annotation properties can be found here.

Filter expressions (for use in the config file)

```
$.aggregated.pages.page.annotations: true
$.aggregated.pages.page.annotations.annot.annotation_flags: true
$.aggregated.pages.page.annotations.annot.annotation_state: true
$.aggregated.pages.page.annotations.annot.bbox: true
$.aggregated.pages.page.annotations.annot.content: true
$.aggregated.pages.page.annotations.annot.created: true
$.aggregated.pages.page.annotations.annot.filename: true
$.aggregated.pages.page.annotations.annot.in_reply_to: true
$.aggregated.pages.page.annotations.annot.intent: true
$.aggregated.pages.page.annotations.annot.last_modified: true
```

```
$.aggregated.pages.page.annotations.annot.layer: true
$.aggregated.pages.page.annotations.annot.name: true
$.aggregated.pages.page.annotations.annot.quadpoints: true
$.aggregated.pages.page.annotations.annot.subject: true
$.aggregated.pages.page.annotations.annot.text_label: true
$.aggregated.pages.page.annotations.annot.type: true
$.aggregated.pages.page.annotations.annot.unique_id: true
$.aggregated.pages.page.annotations.annot.url: true
$.aggregated.pages.page.annotations.annot: true
$.aggregated.pages.page.annotations.num: true
```

Click to copy

2. "pages.page.info"

Creates output reflecting two pieces of information about a given page:

- `pagenum`: the page's sequential page number, the first page in a PDF document having a `pagenum` value of 1
- `pagelabel`: the page's label

Filter expressions (for use in the config file)

```
$.aggregated.pages.page.info: true
$.aggregated.pages.page.info.pagelabel: true
$.aggregated.pages.page.info.pagenum: true
```

Click to copy
As example, the following configuration:

```
$.direct: false
$.aggregated: false
$.aggregated.pages.page.info: true
```

Click to copy
when used for a demo PDF file results in the following Quick Check output:

```
{
  "aggregated": {
    "pages": {
```

```
    "page" : [
      {
        "info" : {
          "pagenum" : 1,
          "pagelabel" : "Introduction - page no. 17"
        }
      },
      {
        "info" : {
          "pagenum" : 2,
          "pagelabel" : "Introduction - page no. 18"
        }
      },
      {
        "info" : {
          "pagenum" : 3,
          "pagelabel" : "Introduction - page no. 19"
        }
      },
      {
        "info" : {
          "pagenum" : 4,
          "pagelabel" : "Introduction - page no. 20"
        }
      }
    ]
  }
}
```


Click to copy

3. "pages.page.geometry"

Creates output reflecting page rotation, page scaling, and various aspects of page geometry boxes for a given page:

- `Rotate`: an integer reflecting the rotation to be applied to the page when rendering the page; possible values are 0 (not rotated) or a multiple of 90.
- `UserUnit`: a number reflecting the page scaling factor to be applied to the page; smallest allowed and also the default value is 1.0

- for each of the possible page geometry boxes – `MediaBox`, `CropBox`, `BleedBox`, `TrimBox`, `ArtBox` - the following data elements are provided:
  - `left`: left side of the page geometry box (*without applying* the `UserUnit` page scaling factor or the `Rotate` key for page rotation)
  - `top`: top side of the page geometry box (*without applying* the `UserUnit` page scaling factor or the `Rotate` key for page rotation)
  - `right`: right side of the page geometry box (*without applying* the `UserUnit` page scaling factor or the `Rotate` key for page rotation)
  - `bottom`: bottom side of the page geometry box (*without applying* the `UserUnit` page scaling factor or the `Rotate` key for page rotation)
  - `height`: difference between `top` and `bottom` (*without applying* the `UserUnit` page scaling factor or the `Rotate` key for page rotation)
  - `width`: difference between `right` and `left` (*without applying* the `UserUnit` page scaling factor or the `Rotate` key for page rotation)
  - `height_eff`: difference between `top` and `bottom` (*after applying* the page scaling factor, but without *applying* the `Rotate` key for page rotation)
  - `width_eff`: difference between `right` and `left` (*after applying* the page scaling factor, but without *applying* the `Rotate` key for page rotation)
  - `present`: info if this page geometry box is included in the given page

Filter expressions (for use in the config file)

```
$.aggregated.pages.page.geometry: true
$.aggregated.pages.page.geometry.ArtBox: true
$.aggregated.pages.page.geometry.ArtBox.bottom: true
$.aggregated.pages.page.geometry.ArtBox.height: true
$.aggregated.pages.page.geometry.ArtBox.height_eff: true
$.aggregated.pages.page.geometry.ArtBox.present: true
$.aggregated.pages.page.geometry.ArtBox.left: true
$.aggregated.pages.page.geometry.ArtBox.right: true
$.aggregated.pages.page.geometry.ArtBox.top: true
```

```
$.aggregated.pages.page.geometry.ArtBox.width: true
$.aggregated.pages.page.geometry.ArtBox.width_eff: true

$.aggregated.pages.page.geometry.BleedBox: true
$.aggregated.pages.page.geometry.BleedBox.bottom: true
$.aggregated.pages.page.geometry.BleedBox.height: true
$.aggregated.pages.page.geometry.BleedBox.height_eff: true
$.aggregated.pages.page.geometry.BleedBox.present: true
$.aggregated.pages.page.geometry.BleedBox.left: true
$.aggregated.pages.page.geometry.BleedBox.right: true
$.aggregated.pages.page.geometry.BleedBox.top: true
$.aggregated.pages.page.geometry.BleedBox.width: true
$.aggregated.pages.page.geometry.BleedBox.width_eff: true

$.aggregated.pages.page.geometry.CropBox: true
$.aggregated.pages.page.geometry.CropBox.bottom: true
$.aggregated.pages.page.geometry.CropBox.height: true
$.aggregated.pages.page.geometry.CropBox.height_eff: true
$.aggregated.pages.page.geometry.CropBox.present: true
$.aggregated.pages.page.geometry.CropBox.left: true
$.aggregated.pages.page.geometry.CropBox.right: true
$.aggregated.pages.page.geometry.CropBox.top: true
$.aggregated.pages.page.geometry.CropBox.width: true
$.aggregated.pages.page.geometry.CropBox.width_eff: true

$.aggregated.pages.page.geometry.MediaBox: true
$.aggregated.pages.page.geometry.MediaBox.bottom: true
$.aggregated.pages.page.geometry.MediaBox.height: true
$.aggregated.pages.page.geometry.MediaBox.height_eff: true
$.aggregated.pages.page.geometry.MediaBox.present: true
$.aggregated.pages.page.geometry.MediaBox.left: true
$.aggregated.pages.page.geometry.MediaBox.right: true
$.aggregated.pages.page.geometry.MediaBox.top: true
$.aggregated.pages.page.geometry.MediaBox.width: true
$.aggregated.pages.page.geometry.MediaBox.width_eff: true

$.aggregated.pages.page.geometry.TrimBox: true
$.aggregated.pages.page.geometry.TrimBox.bottom: true
$.aggregated.pages.page.geometry.TrimBox.height: true
$.aggregated.pages.page.geometry.TrimBox.height_eff: true
$.aggregated.pages.page.geometry.TrimBox.present: true
$.aggregated.pages.page.geometry.TrimBox.left: true
$.aggregated.pages.page.geometry.TrimBox.right: true
```

```
$.aggregated.pages.page.geometry.TrimBox.top: true
$.aggregated.pages.page.geometry.TrimBox.width: true
$.aggregated.pages.page.geometry.TrimBox.width_eff: true

$.aggregated.pages.page.geometry.UserUnit: true
$.aggregated.pages.page.geometry.Rotate: true
```

Click to copy

As example, the following configuration:

```
$.direct: false
$.aggregated: false
$.aggregated.pages.page.geometry: true
```

Click to copy

when used for a 1 page demo PDF file results in the following Quick Check output:

```
{
                        "aggregated" :
                        {
                                "pages" :
                                {
                                        "page" :
                                        [
                                                {
                                                        "geometry" :
                                                        {
                                                                "ArtBox" :
                                                                {
                                                                        "bottom" :
21,
                                                                        "height" :
841.88999999999999,

"height_eff" : 1683.78,
                                                                        "left" :
21,
                                                                        "present"
: true,
                                                                        "right" :
616.27599999999995,
                                                                        "top" :
```

```
862.88999999999999,
                                                                        "width" :
595.27599999999995,

"width_eff" : 1190.5519999999999
                                                                    },
                                                                    "BleedBox" :
                                                                    {
                                                                        "bottom" :
12.4961,
                                                                        "height" :
858.89800000000002,

"height_eff" : 1717.7958000000001,
                                                                        "left" :
12.4961,
                                                                        "present"
: true,
                                                                        "right" :
624.77999999999997,
                                                                        "top" :
871.39400000000001,
                                                                        "width" :
612.28399999999999,

"width_eff" : 1224.5678
                                                                    },
                                                                    "CropBox" :
                                                                    {
                                                                        "bottom" :
0,
                                                                        "height" :
883.88999999999999,

"height_eff" : 1767.78,
                                                                        "left" : 0,
                                                                        "present"
: true,
                                                                        "right" :
637.27599999999995,
                                                                        "top" :
883.88999999999999,
                                                                        "width" :
```

```
637.27599999999995,

"width_eff" : 1274.5519999999999
                                                                    },
                                                                    "MediaBox" :
                                                                    {
                                                                            "bottom" :
0,
                                                                            "height" :
883.88999999999999,

"height_eff" : 1767.78,
                                                                            "left" : 0,
                                                                            "present"
: true,
                                                                            "right" :
637.27599999999995,
                                                                            "top" :
883.88999999999999,
                                                                            "width" :
637.27599999999995,

"width_eff" : 1274.5519999999999
                                                                    },

                                                                    "TrimBox" :
                                                                    {
                                                                            "bottom" :
21,
                                                                            "height" :
841.88999999999999,

"height_eff" : 1683.78,
                                                                            "left" :
21,
                                                                            "present"
: true,
                                                                            "right" :
616.27599999999995,
                                                                            "top" :
862.88999999999999,
                                                                            "width" :
595.27599999999995,
```

```
"width_eff" : 1190.5519999999999
                                                                    },
                                                                    "Rotate" :
0,
                                                                    "UserUnit" : 2
                                                                }
                                                            }
                                                        ]
                                                    }
                                                }
}
```


Click to copy

4. "pages.page.contentstream"

Creates output reflecting the size of the content stream (in bytes) per page. The following configuration applies:

```
$.aggregated.pages.page.contentstream: true
$.aggregated.pages.page.contentstream.size: true
$.aggregated.pages.page.contentstream.size_wfx: true
```

Click to copy

For more information about content stream size at the aggregate level, see the section 8. "page.contentstream" above.

5. "pages.page.resources"

For each page, it is also possible to request summary information about color usage, and information about color, font resources, images referenced by that page's Resources dictionary and any Resources dictionaries of Form XObjects referenced on that page.

pages.page.resources has the following sub-areas:

- pages.page.resources.color
- pages.page.resources.fonts
- pages.page.resources.images
- pages.page.resources.overprint
- pages.page.resources.transparency

- pages.page.resources.form_xobjects

> Note: resources referenced by a page's **Resources** dictionary – or those referenced by form XObjects on that page – do not actually have to be used by that page or its form XObjects.

"pages.page.resources.color"

```
$.aggregated.pages.page.resources.color: true
```

Click to copy

Creates information about color usage for a page, and information about color resources referenced by that page (or formXObjects on that page).

The "color" area contains four sub-areas:

- `summary`: a list of entries where each represents a certain aggregated aspect of color usage; for example, if `DeviceCMYK` has a value of 0, there is no graphics object on the page that uses DeviceCMYK (but there might be a graphics object that uses DeviceN with the colorants Cyan, Magenta, Yellow, Black, or ICC based CMYK). In comparison, `Any_CMYK` reports any use of CMYK, whether DeviceCMYK, ICC based CMYK, DeviceN with one, several or all of the Cyan, Magenta, Yellow, Black colorants, or Separation color space Cyan, Magenta, Yellow, Black.
- `colorspaces`: a list of entries (only those are shown that are applicable) reflecting the presence of certain color spaces.
- `spotcolors`: a list of entries for spot colors used, including their name and the alternate color space used. together with associated color values for 100% tint value of the spot color
- `color_plates`: a list of color plates on each page and for the whole document. This is not the exact but the maximum possible number of color separations.

"summary"

All entries under `summary` have as its value an integer reflecting how often the respective type of color is used. A full description of each `summary` entry can be found in the section 8.5 "resources.color".

Filter expressions (for use in the config file):

```
$.aggregated.pages.page.resources.color.summary: true
```

Click to copy

"colorspaces"

Creates output reflecting `colorspaces` and `default colorspaces` for a given page. For more information about colorspaces at the aggregate level, see the section "resources.color.colorspaces" above.

Filter expressions (for use in the config file):

```
$.aggregated.pages.page.resources.color.colorspaces: true
$.aggregated.pages.page.resources.color.colorspaces.colorspace: true
$.aggregated.pages.page.resources.color.colorspaces.icc_source_profiles: true
$.aggregated.pages.page.resources.color.colorspaces.length: true

$.aggregated.pages.page.resources.color.default_colorspaces: true
$.aggregated.pages.page.resources.color.default_colorspaces.colorspace: true
$.aggregated.pages.page.resources.color.default_colorspaces.icc_source_profiles:
true
$.aggregated.pages.page.resources.color.default_colorspaces.length: true
```

Click to copy

"spotcolors"

Creates output reflecting spotcolor usage for a given page. For more information about `spotcolors` at the aggregate level, see the section "resources.color.spotcolors" above.

Filter expressions (for use in the config file):

```
$.aggregated.pages.page.resources.color.spotcolors: true
$.aggregated.pages.page.resources.color.spotcolors.length: true
$.aggregated.pages.page.resources.color.spotcolors.spotcolor: true
$.aggregated.pages.page.resources.color.spotcolors.spotcolor.alternatespace: true
$.aggregated.pages.page.resources.color.spotcolors.spotcolor.alternatevalues: true
$.aggregated.pages.page.resources.color.spotcolors.spotcolor.name: true
```

Click to copy

As example, the following configuration:

```
$.direct: false
$.aggregated: false
$.aggregated.pages.page.resources.color: true
```

Click to copy

when used for a single page demo PDF file results in the following Quick Check output:

```
{
  "aggregated": {
    "pages": {
      "page" : [
        {
          "resources" : {
            "color" : {
              "summary" : {
                "Any_CMYK" : 45,
                "DeviceCMYK" : 45,
                "ICCBased_CMYK" : 0,
                "Any_RGB" : 0,
                "DeviceRGB" : 0,
                "CalRGB" : 0,
                "ICCBased_RGB" : 0,
                "Calibrated_RGB" : 0,
                "Lab" : 0,
                "ICCBased_Lab" : 0,
                "Any_Gray" : 0,
                "DeviceGray" : 0,
                "CalGray" : 0,
                "ICCBased_Gray" : 0,
                "Any_Device" : 45,
                "Any_Calibrated" : 0,
                "Any_Spot" : 5,
```

```
      "Not_DeviceCMYK" : 4,
      "Not_DeviceCMYK_Or_Spot" : 0,
      "Smooth_Shades" : 7,
      "Pattern" : 0,
      "Any_Separation" : 3,
      "Separation_All" : 0,
      "Separation_None" : 0,
      "Separation_Cyan" : 0,
      "Separation_Magenta" : 0,
      "Separation_Yellow" : 0,
      "Separation_Black" : 0,
      "Separation_Any_Of_CMYK" : 0,
      "Separation_Any_Spot" : 3,
      "Any_DeviceN" : 1,
      "DeviceN_Any_Of_CMYK" : 0,
      "DeviceN_All_Of_CMYK" : 0,
      "DeviceN_All_Of_CMYK_And_Spot" : 0,
      "DeviceN_All_Of_CMYK_No_Spot" : 0,
      "DeviceN_All_Of_Spot" : 1
    },
    "colorspaces" : {
      "length" : 5,
      "colorspace" : [
        "DeviceCMYK",
        "DeviceN",
        "DeviceN_SpotOnly",
        "Separation",
        "Separation_Spot"
      ]
    },
    "spotcolors" : {
      "length" : 3,
      "spotcolor" : [
      {
        "name" : "Silver",
        "alternatespace" : "ICCBased_Lab",
        "alternatevalues" : [
          63.603600,
          -0.977516,
          -2.108350
        ]
      },
      {
```

```
                "name" : "Violet",
                "alternatespace" : "ICCBased_Lab",
                "alternatevalues" : [
                  35.065900,
                  41.242200,
                  -46.451700
                ]
              },
              {
                "name" : "Red",
                "alternatespace" : "ICCBased_Lab",
                "alternatevalues" : [
                  44.297300,
                  68.503100,
                  44.137000
                ]
              }
              ]
            }
          }
        }
      }
    ]
  }
}
```


Click to copy

"color_plates"

Creates a list of color plates on each page and for the whole document. This is not the exact but the maximum possible number of color separations. "color_plates" contains two entires:

- `length` : number of color plates
- `plates` : an array of entries for each color plate

Filter expressions (for use in the config file):

```
$.aggregated.pages.page.resources.color.color_plates: true
$.aggregated.pages.page.resources.color.color_plates.length: true
$.aggregated.pages.page.resources.color.color_plates.plates: true
```

Click to copy

"pages.page.resources.fonts"

Creates output reflecting `font` usage for a given page. For more information about `fonts` at the aggregate level, see the section "resources.fonts" above.

Filter expressions (for use in the config file):

```
$.aggregated.pages.page.resources.fonts: true
$.aggregated.pages.page.resources.fonts.font: true
$.aggregated.pages.page.resources.fonts.font.embedded: true
$.aggregated.pages.page.resources.fonts.font.fonttype: true
$.aggregated.pages.page.resources.fonts.font.name: true
$.aggregated.pages.page.resources.fonts.font.subset: true
$.aggregated.pages.page.resources.fonts.length: true
```

Click to copy

As an example, the following configuration:

```
$.direct: false
$.aggregated: false
$.aggregated.pages.page.resources.fonts: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output:

```
{
  "aggregated": {
    "pages": {
      "page" : [
        {
          "resources" : {
            "fonts" : {
              "length" : 4,
              "font" : [
                {
                  "name" : "FrutigerLTStd-BoldItalic",
                  "subset" : true,
                  "fonttype" : "Type1",
                  "embedded" : true
                },
```

```
                {
                  "name" : "FrutigerLTStd-Italic",
                  "subset" : true,
                  "fonttype" : "Type1",
                  "embedded" : true
                },
                {
                  "name" : "FrutigerLTStd-Cn",
                  "subset" : true,
                  "fonttype" : "Type1",
                  "embedded" : true
                },
                {
                  "name" : "ArialMT",
                  "subset" : true,
                  "fonttype" : "TrueType",
                  "embedded" : true
                }
              ]
            }
          }
        }
      ]
    }
  }
```


Click to copy

"pages.page.resources.images"

Creates an output that reflects the use of `images` for a given page. For more information about `images` at the aggregate level, see the section "resources.images" above.

Filter expressions (for use in the config file):

```
$.aggregated.pages.page.resources.images: true

$.aggregated.pages.page.resources.images.summary.bitmap_images.eff_highest_ppi:
true
$.aggregated.pages.page.resources.images.summary.bitmap_images.eff_lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.highest_pixel_h:
true
```

```
$.aggregated.pages.page.resources.images.summary.bitmap_images.highest_pixel_v:
true
$.aggregated.pages.page.resources.images.summary.bitmap_images.highest_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.lowest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.lowest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.number: true
$.aggregated.pages.page.resources.images.summary.bitmap_images: true

$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.
eff_highest_ppi: true"
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.
eff_lowest_ppi: true"
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.high-
est_pixel_h: true"
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.high-
est_pixel_v: true"
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.high-
est_ppi: true"
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.low-
est_pixel_h: true"
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.low-
est_pixel_v: true"
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.num-
ber: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks: true

$.aggregated.pages.page.resources.images.summary.colorspaces: true
$.aggregated.pages.page.resources.images.summary.colorspaces.length: true

$.aggregated.pages.page.resources.images.summary.ct_images.eff_highest_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images.eff_lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images.highest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.ct_images.highest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.ct_images.highest_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images.lowest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.ct_images.lowest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.ct_images.lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images.number: true
$.aggregated.pages.page.resources.images.summary.ct_images: true
```

```
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.eff_high-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.eff_low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.high-
est_pixel_h: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.high-
est_pixel_v: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.high-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.lowest_pix-
el_h: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.lowest_pix-
el_v: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.number:
true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks: true

$.aggregated.pages.page.resources.images.summary.imagemasks.eff_highest_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks.eff_lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks.highest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.imagemasks.highest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.imagemasks.highest_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks.lowest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.imagemasks.lowest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.imagemasks.lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks.number: true
$.aggregated.pages.page.resources.images.summary.imagemasks: true

$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.eff_high-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.eff_low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.high-
est_pixel_h: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.high-
est_pixel_v: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.high-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.low-
est_pixel_h: true
```

```
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.low-
est_pixel_v: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.number:
true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks: true

$.aggregated.pages.page.resources.images.summary.number: true
$.aggregated.pages.page.resources.images.summary: true
```

Click to copy

The following configuration:

```
$.direct: false
$.aggregated: false
$.aggregated.pages.page.resources.images: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output (click on the drop down):

```
{
  "aggregated": {
    "pages": {
      "page" : [
        {
          "resources" : {
            "images" : {
              "summary" : {
                "number" : 1,
                "colorspace" : [
                  "DeviceRGB"
                ],
                "ct_images" : {
                  "number" : 1,
                  "lowest_ppi" : 100,
                  "highest_ppi" : 100,
                  "eff_lowest_ppi" : 100,
                  "eff_highest_ppi" : 100,
                  "lowest_pixel_h" : 850,
                  "highest_pixel_h" : 850,
                  "lowest_pixel_v" : 1100,
```

```
                "highest_pixel_v" : 1100
              },
              "bitmap_images" : {
                "number" : 0
              },
              "imagemasks" : {
                "number" : 0
              },
              "ct_images_in_softmasks" : {
                "number" : 0
              },
              "bitmap_images_in_softmasks" : {
                "number" : 0
              },
              "imagemasks_in_softmasks" : {
                "number" : 0
              }
            }
          }
        }
      }
```


Click to copy

"pages.page.resources.overprint"

Creates information of used overprinting parameters for a given page. This sub-area has the following two entries:

- `uses_opm`: overprint mode parameter (OPM)
- `uses_overprint`: overprint parameter (OV)

Filter expressions (for use in the config file):

```
$.aggregated.pages.page.resources.overprint: true
$.aggregated.pages.page.resources.overprint.uses_opm: true
$.aggregated.pages.page.resources.overprint.uses_overprint: true
```

Click to copy
The following configuration:

```
$.direct: false
$.aggregated: false
```

```
$.aggregated.pages.page.resources.overprint: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output (click on the drop down):


```
{
  "aggregated": {
    "pages": {
      "page" : [
        {
          "resources" : {
            "overprint" : {
              "uses_overprint" : false,
              "uses_opm" : false
            }
          }
        },
        {
          "resources" : {
            "overprint" : {
              "uses_overprint" : true,
              "uses_opm" : true
            }
          }
        }
      ]
    }
```


Click to copy

"pages.page.resources.form_xobjects"

This filter expression lists PDF resources which are not redefined within a Form X Object, but are used when drawing the Form X Object content stream. This resources may have a direct effect on the rendering of the Form X Object. Examples of affected PDF resources:

- `fill_colorspace`
- `text_font`
- `line_width`
- `overprint`

- `rendering_intent`
- `blend_mode`
- etc.

```
$.aggregated.pages.page.resources.form_xobjects: true
$.aggregated.pages.page.resources.form_xobjects.inherited_resources: true
```

Click to copy

"pages.page.resources.transparency"

Creates an output that reflects the use of `transparency` for a given page. For more information about `transparency` at the aggregate level, see the section "8.1 resources.transparency" above.

Filter expressions (for use in the config file)

```
$.aggregated.pages.page.resources.transparency: true

$.aggregated.pages.page.resources.transparency.blendmodes: true
$.aggregated.pages.page.resources.transparency.blendmodes.color: true
$.aggregated.pages.page.resources.transparency.blendmodes.colorburn: true
$.aggregated.pages.page.resources.transparency.blendmodes.colordodge: true
$.aggregated.pages.page.resources.transparency.blendmodes.darken: true
$.aggregated.pages.page.resources.transparency.blendmodes.difference: true
$.aggregated.pages.page.resources.transparency.blendmodes.exclusion: true
$.aggregated.pages.page.resources.transparency.blendmodes.hardlight: true
$.aggregated.pages.page.resources.transparency.blendmodes.hue: true
$.aggregated.pages.page.resources.transparency.blendmodes.lighten: true
$.aggregated.pages.page.resources.transparency.blendmodes.luminosity: true
$.aggregated.pages.page.resources.transparency.blendmodes.multiply: true
$.aggregated.pages.page.resources.transparency.blendmodes.non_normal: true
$.aggregated.pages.page.resources.transparency.blendmodes.overlay: true
$.aggregated.pages.page.resources.transparency.blendmodes.saturation: true
$.aggregated.pages.page.resources.transparency.blendmodes.screen: true
$.aggregated.pages.page.resources.transparency.blendmodes.softlight: true

$.aggregated.pages.page.resources.transparency.groups: true
$.aggregated.pages.page.resources.transparency.groups.isolated: true
$.aggregated.pages.page.resources.transparency.groups.isolated_devicecmyk: true
$.aggregated.pages.page.resources.transparency.groups.isolated_devicegray: true
$.aggregated.pages.page.resources.transparency.groups.isolated_devicergb: true
```

```
$.aggregated.pages.page.resources.transparency.groups.isolated_icc_cmyk: true
$.aggregated.pages.page.resources.transparency.groups.isolated_icc_gray: true
$.aggregated.pages.page.resources.transparency.groups.isolated_icc_rgb: true
$.aggregated.pages.page.resources.transparency.groups.isolated_non_cmyk: true
$.aggregated.pages.page.resources.transparency.groups.isolated_non_devicecmyk: true
$.aggregated.pages.page.resources.transparency.groups.non_isolated: true

$.aggregated.pages.page.resources.transparency.has_explicit_transparency: true
$.aggregated.pages.page.resources.transparency.has_transparency: true
$.aggregated.pages.page.resources.transparency.opacity_less_than_1: true

$.aggregated.pages.page.resources.transparency.softmasks: true
$.aggregated.pages.page.resources.transparency.softmasks.softmasks_any: true
$.aggregated.pages.page.resources.transparency.softmasks.softmasks_in_extgstate:
true
$.aggregated.pages.page.resources.transparency.softmasks.softmasks_in_image: true
```

Click to copy

The following configuration:

```
$.direct: false
$.aggregated: false
$.aggregated.pages.page.resources.transparency: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output:

```
{
  "aggregated": {
    "pages": {
      "page" : [
        {
          "resources" : {
            "transparency" : {
              "has_transparency" : false,
              "has_explicit_transparency" : false,
              "opacity_less_than_1" : false,
              "softmasks" : {
                "softmasks_any" : false,
                "softmasks_in_extgstate" : false,
                "softmasks_in_image" : false
              },
```

```
            "groups" : {
              "isolated" : false,
              "isolated_devicecmyk" : false,
              "isolated_icc_cmyk" : false,
              "isolated_devicergb" : false,
              "isolated_icc_rgb" : false,
              "isolated_devicegray" : false,
              "isolated_icc_gray" : false,
              "isolated_non_devicecmyk" : false,
              "isolated_non_cmyk" : false,
              "non_isolated" : false
            },
            "blendmodes" : {
              "non_normal" : false,
              "multiply" : false,
              "screen" : false,
              "overlay" : false,
              "darken" : false,
              "lighten" : false,
              "colordodge" : false,
              "colorburn" : false,
              "hardlight" : false,
              "softlight" : false,
              "difference" : false,
              "exclusion" : false,
              "hue" : false,
              "saturation" : false,
              "color" : false,
              "luminosity" : false
            }
          }
        }
      },
      {
        "resources" : {
          "transparency" : {
            "has_transparency" : true,
            "has_explicit_transparency" : true,
            "opacity_less_than_1" : true,
            "softmasks" : {
              "softmasks_any" : true,
              "softmasks_in_extgstate" : false,
              "softmasks_in_image" : true
```

```
            },
            "groups" : {
              "isolated" : false,
              "isolated_devicecmyk" : false,
              "isolated_icc_cmyk" : false,
              "isolated_devicergb" : false,
              "isolated_icc_rgb" : false,
              "isolated_devicegray" : false,
              "isolated_icc_gray" : false,
              "isolated_non_devicecmyk" : false,
              "isolated_non_cmyk" : false,
              "non_isolated" : true
            },
            "blendmodes" : {
              "non_normal" : false,
              "multiply" : false,
              "screen" : false,
              "overlay" : false,
              "darken" : false,
              "lighten" : false,
              "colordodge" : false,
              "colorburn" : false,
              "hardlight" : false,
              "softlight" : false,
              "difference" : false,
              "exclusion" : false,
              "hue" : false,
              "saturation" : false,
              "color" : false,
              "luminosity" : false
            }
          }
        }
      }
    ]
  }
```


Click to copy

6. "pages.page.pieceinfo"

A way of embedding metadata is the PieceInfo Dictionary, used by Illustrator and Photoshop for application-specific data when you save a file as a PDF. The following configuration:

```
$.aggregated.pages.page.pieceinfo: true
```

Click to copy

when used for a demo PDF file results in the following Quick Check output:

```
{
        "vars" :
        {
                "pieceinfo" :
                {
                        "app_vars_sub_path" : "quickcheck_1",
                        "quickcheck_config" :
                        [
                                "$.direct: false",
                                "$.direct.Info: false",
                                "$.aggregated: false",
                                "$.aggregated.pages.page.geometry.TrimBox: false ",
                                "$.aggregated.pages.page.pieceinfo: true "
                        ]
                },
                "quickcheck_1" :
                {
                        "aggregated" :
                        {
                                "pages" :
                                {
                                        "page" :
                                        [

                                                {
                                                        "pieceinfo" :
                                                        {
                                                                "Illustrator" :
                                                                {
                                                                        "LastModi-
```

```
fied" : "D:20191105172949+02'00'",
                                                                    "Private"
:
                                                                    {

"AIMetaData" :
                                                                            {

     "Length" : 1116
                                                                            },

"AIPDFPrivateData1" :
                                                                            {

     "Length" : 23074
                                                                            },

"ContainerVersion" : 11,

"CreatorVersion" : 24,

"RoundtripStreamType" : 2,

"RoundtripVersion" : 24
                                                                    }
                                                                }
                                                            }
                                                        }
                                                    ]
                                                }
                                            },
                                            "status" :
                                            {
                                                    "result" : "complete",
                                                    "time_needed_sec" : 0.0010370
                                            }
                                    }
                            }
}
```


Click to copy

List of all "aggregated" filter expressions

List of all filter expressions for data substructures required for JavaScript variables in a Quick Check step in a Process Plan

```
$.aggregated.bookmarks.bookmark.level: true
$.aggregated.bookmarks.bookmark.name: true
$.aggregated.bookmarks.bookmark.page: true
$.aggregated.bookmarks.bookmark: true
$.aggregated.bookmarks.length: true
$.aggregated.bookmarks: true

$.aggregated.doc.annotations.annot.annotation_flags: true
$.aggregated.doc.annotations.annot.annotation_state: true
$.aggregated.doc.annotations.annot.bbox: true
$.aggregated.doc.annotations.annot.content: true
$.aggregated.doc.annotations.annot.created: true
$.aggregated.doc.annotations.annot.filename: true
$.aggregated.doc.annotations.annot.in_reply_to: true
$.aggregated.doc.annotations.annot.intent: true
$.aggregated.doc.annotations.annot.last_modified: true
$.aggregated.doc.annotations.annot.layer: true
$.aggregated.doc.annotations.annot.name: true
$.aggregated.doc.annotations.annot.pagenum: true
$.aggregated.doc.annotations.annot.quadpoints: true
$.aggregated.doc.annotations.annot.subject: true
$.aggregated.doc.annotations.annot.text_label: true
$.aggregated.doc.annotations.annot.type: true
$.aggregated.doc.annotations.annot.unique_id: true
$.aggregated.doc.annotations.annot.url: true
$.aggregated.doc.annotations.annot: true
$.aggregated.doc.annotations.num: true
$.aggregated.doc.annotations: true

$.aggregated.doc.digsig.Filter: true
$.aggregated.doc.digsig.Name: true
$.aggregated.doc.digsig.SubFilter: true
$.aggregated.doc.digsig: true
```

```
$.aggregated.doc.fields: true
$.aggregated.doc.fields.field: true
$.aggregated.doc.fields.field.title: true
$.aggregated.doc.fields.field.value: true

$.aggregated.doc.create_id: true
$.aggregated.doc.created: true

$.aggregated.doc.dpartroot.dparts.dparts: true
$.aggregated.doc.dpartroot.dparts.dpm: true
$.aggregated.doc.dpartroot.dparts.end: true
$.aggregated.doc.dpartroot.dparts.start: true
$.aggregated.doc.dpartroot.dparts: true
$.aggregated.doc.dpartroot.nodenamelist: true
$.aggregated.doc.dpartroot.recordlevel: true
$.aggregated.doc.dpartroot: true

$.aggregated.doc.is_tagged: true
$.aggregated.doc.lang: true
$.aggregated.doc.modified: true
$.aggregated.doc.modified_id: true

$.aggregated.doc.outputintents.length: true
$.aggregated.doc.outputintents.outputintent.destoutprofile.colorspace: true
$.aggregated.doc.outputintents.outputintent.destoutprofile.name: true
$.aggregated.doc.outputintents.outputintent.destoutprofile.profiletype: true
$.aggregated.doc.outputintents.outputintent.destoutprofile: true
$.aggregated.doc.outputintents.outputintent.outputintentid: true
$.aggregated.doc.outputintents.outputintent.type: true
$.aggregated.doc.outputintents.outputintent: true
$.aggregated.doc.outputintents: true

$.aggregated.doc.pdf_version: true

$.aggregated.doc.pdfa.destoutprofile.colorspace: true
$.aggregated.doc.pdfa.destoutprofile.name: true
$.aggregated.doc.pdfa.destoutprofile.profiletype: true
$.aggregated.doc.pdfa.destoutprofile: true
$.aggregated.doc.pdfa.outputintentid: true
$.aggregated.doc.pdfa.type: true
$.aggregated.doc.pdfa: true
```

```
$.aggregated.doc.pdfe.destoutprofile.colorspace: true
$.aggregated.doc.pdfe.destoutprofile.name: true
$.aggregated.doc.pdfe.destoutprofile.profiletype: true
$.aggregated.doc.pdfe.destoutprofile: true
$.aggregated.doc.pdfe.outputintentid: true
$.aggregated.doc.pdfe.type: true
$.aggregated.doc.pdfe: true

$.aggregated.doc.pdfua.type: true
$.aggregated.doc.pdfua: true

$.aggregated.doc.pdfvt.type: true
$.aggregated.doc.pdfvt: true

$.aggregated.doc.pdfx.destoutprofile.colorspace: true
$.aggregated.doc.pdfx.destoutprofile.name: true
$.aggregated.doc.pdfx.destoutprofile.profiletype: true
$.aggregated.doc.pdfx.destoutprofile: true
$.aggregated.doc.pdfx.outputintentid: true
$.aggregated.doc.pdfx.type: true
$.aggregated.doc.pdfx: true

$.aggregated.doc.potential_syntax_issues: true
$.aggregated.doc.rolemap: true
$.aggregated.doc.xmpmetadata: true
$.aggregated.doc: true

$.aggregated.embeddedfiles.embeddedfile.af_relationship: true
$.aggregated.embeddedfiles.embeddedfile.bytes: true
$.aggregated.embeddedfiles.embeddedfile.created: true
$.aggregated.embeddedfiles.embeddedfile.description: true
$.aggregated.embeddedfiles.embeddedfile.last_modified: true
$.aggregated.embeddedfiles.embeddedfile.name: true
$.aggregated.embeddedfiles.embeddedfile: true
$.aggregated.embeddedfiles.length: true
$.aggregated.embeddedfiles: true

$.aggregated.env.job_id: true
$.aggregated.env.machine_name: true
$.aggregated.env.pdft_uuid: true
$.aggregated.env.platform: true
$.aggregated.env.process_id: true
$.aggregated.env.program_name: true
```

```
$.aggregated.env.program_version: true
$.aggregated.env.timestamp: true
$.aggregated.env.timestamp_day: true
$.aggregated.env.timestamp_hour: true
$.aggregated.env.timestamp_month: true
$.aggregated.env.timestamp_weekday: true
$.aggregated.env.verb: true
$.aggregated.env: true

$.aggregated.file.bytes: true
$.aggregated.file.created: true
$.aggregated.file.filepath: true
$.aggregated.file.modified: true
$.aggregated.file.name: true
$.aggregated.file.path: true
$.aggregated.file: true

$.aggregated.ocgs.length: true
$.aggregated.ocgs.ocg.gui: true
$.aggregated.ocgs.ocg.name: true
$.aggregated.ocgs.ocg.visible: true
$.aggregated.ocgs.ocg: true
$.aggregated.ocgs: true

$.aggregated.pages.ArtBox.height_max: true
$.aggregated.pages.ArtBox.height_min: true
$.aggregated.pages.ArtBox.num: true
$.aggregated.pages.ArtBox.num_landscape: true
$.aggregated.pages.ArtBox.num_portrait: true
$.aggregated.pages.ArtBox.num_square: true
$.aggregated.pages.ArtBox.width_max: true
$.aggregated.pages.ArtBox.width_min: true
$.aggregated.pages.ArtBox: true

$.aggregated.pages.BleedBox.bottom_max: true
$.aggregated.pages.BleedBox.bottom_min: true
$.aggregated.pages.BleedBox.height_max: true
$.aggregated.pages.BleedBox.height_min: true
$.aggregated.pages.BleedBox.inner_max: true
$.aggregated.pages.BleedBox.inner_min: true
$.aggregated.pages.BleedBox.left_max: true
$.aggregated.pages.BleedBox.left_min: true
$.aggregated.pages.BleedBox.num: true
```

```
$.aggregated.pages.BleedBox.num_landscape: true
$.aggregated.pages.BleedBox.num_portrait: true
$.aggregated.pages.BleedBox.num_square: true
$.aggregated.pages.BleedBox.outer_max: true
$.aggregated.pages.BleedBox.outer_min: true
$.aggregated.pages.BleedBox.right_max: true
$.aggregated.pages.BleedBox.right_min: true
$.aggregated.pages.BleedBox.top_max: true
$.aggregated.pages.BleedBox.top_min: true
$.aggregated.pages.BleedBox.width_max: true
$.aggregated.pages.BleedBox.width_min: true
$.aggregated.pages.BleedBox: true

$.aggregated.pages.contentstream.size_max: true
$.aggregated.pages.contentstream.size_min: true
$.aggregated.pages.contentstream.size_total: true
$.aggregated.pages.contentstream: true
$.aggregated.pages.contentstream.size_wfx_max: true
$.aggregated.pages.contentstream.size_wfx_min: true
$.aggregated.pages.contentstream.size_wfx_total: true

$.aggregated.pages.CropBox.height_max: true
$.aggregated.pages.CropBox.height_min: true
$.aggregated.pages.CropBox.num: true
$.aggregated.pages.CropBox.num_landscape: true
$.aggregated.pages.CropBox.num_portrait: true
$.aggregated.pages.CropBox.num_square: true
$.aggregated.pages.CropBox.width_max: true
$.aggregated.pages.CropBox.width_min: true
$.aggregated.pages.CropBox: true

$.aggregated.pages.effective_ArtBox.height_max: true
$.aggregated.pages.effective_ArtBox.height_min: true
$.aggregated.pages.effective_ArtBox.num: true
$.aggregated.pages.effective_ArtBox.num_landscape: true
$.aggregated.pages.effective_ArtBox.num_portrait: true
$.aggregated.pages.effective_ArtBox.num_square: true
$.aggregated.pages.effective_ArtBox.width_max: true
$.aggregated.pages.effective_ArtBox.width_min: true
$.aggregated.pages.effective_ArtBox: true

$.aggregated.pages.effective_BleedBox.height_max: true
$.aggregated.pages.effective_BleedBox.height_min: true
```

```
$.aggregated.pages.effective_BleedBox.num: true
$.aggregated.pages.effective_BleedBox.num_landscape: true
$.aggregated.pages.effective_BleedBox.num_portrait: true
$.aggregated.pages.effective_BleedBox.num_square: true
$.aggregated.pages.effective_BleedBox.width_max: true
$.aggregated.pages.effective_BleedBox.width_min: true
$.aggregated.pages.effective_BleedBox: true

$.aggregated.pages.effective_CropBox.height_max: true
$.aggregated.pages.effective_CropBox.height_min: true
$.aggregated.pages.effective_CropBox.num: true
$.aggregated.pages.effective_CropBox.num_landscape: true
$.aggregated.pages.effective_CropBox.num_portrait: true
$.aggregated.pages.effective_CropBox.num_square: true
$.aggregated.pages.effective_CropBox.width_max: true
$.aggregated.pages.effective_CropBox.width_min: true
$.aggregated.pages.effective_CropBox: true

$.aggregated.pages.effective_MediaBox.height_max: true
$.aggregated.pages.effective_MediaBox.height_min: true
$.aggregated.pages.effective_MediaBox.num: true
$.aggregated.pages.effective_MediaBox.num_landscape: true
$.aggregated.pages.effective_MediaBox.num_portrait: true
$.aggregated.pages.effective_MediaBox.num_square: true
$.aggregated.pages.effective_MediaBox.width_max: true
$.aggregated.pages.effective_MediaBox.width_min: true
$.aggregated.pages.effective_MediaBox: true

$.aggregated.pages.effective_TrimBox.height_max: true
$.aggregated.pages.effective_TrimBox.height_min: true
$.aggregated.pages.effective_TrimBox.num: true
$.aggregated.pages.effective_TrimBox.num_landscape: true
$.aggregated.pages.effective_TrimBox.num_portrait: true
$.aggregated.pages.effective_TrimBox.num_square: true
$.aggregated.pages.effective_TrimBox.width_max: true
$.aggregated.pages.effective_TrimBox.width_min: true
$.aggregated.pages.effective_TrimBox: true

$.aggregated.pages.length: true

$.aggregated.pages.MediaBox.height_max: true
$.aggregated.pages.MediaBox.height_min: true
$.aggregated.pages.MediaBox.num: true
```

```
$.aggregated.pages.MediaBox.num_landscape: true
$.aggregated.pages.MediaBox.num_portrait: true
$.aggregated.pages.MediaBox.num_square: true
$.aggregated.pages.MediaBox.width_max: true
$.aggregated.pages.MediaBox.width_min: true
$.aggregated.pages.MediaBox: true

$.aggregated.multiple_resource_refs_across_pages: true
$.aggregated.multiple_resource_refs_across_pages.colorspace: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_alt: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_cie: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_device: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_icc: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_indexed: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_inside_cs: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_inside_image: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_inside_shade: true
$.aggregated.multiple_resource_refs_across_pages.colorspace_separation_devicen:
true
$.aggregated.multiple_resource_refs_across_pages.colorspace_special: true
$.aggregated.multiple_resource_refs_across_pages.extgstate: true
$.aggregated.multiple_resource_refs_across_pages.font: true
$.aggregated.multiple_resource_refs_across_pages.font_cid: true
$.aggregated.multiple_resource_refs_across_pages.font_descriptor: true
$.aggregated.multiple_resource_refs_across_pages.font_encoding: true
$.aggregated.multiple_resource_refs_across_pages.font_simple: true
$.aggregated.multiple_resource_refs_across_pages.pattern: true
$.aggregated.multiple_resource_refs_across_pages.pattern_shading: true
$.aggregated.multiple_resource_refs_across_pages.pattern_tiling: true
$.aggregated.multiple_resource_refs_across_pages.procset: true
$.aggregated.multiple_resource_refs_across_pages.properties: true
$.aggregated.multiple_resource_refs_across_pages.resource_dictionary: true
$.aggregated.multiple_resource_refs_across_pages.shade: true
$.aggregated.multiple_resource_refs_across_pages.xobject: true
$.aggregated.multiple_resource_refs_across_pages.xobject_forms: true
$.aggregated.multiple_resource_refs_across_pages.xobject_image: true

$.aggregated.pages.page.annotations.annot.annotation_flags: true
$.aggregated.pages.page.annotations.annot.annotation_state: true
$.aggregated.pages.page.annotations.annot.bbox: true
$.aggregated.pages.page.annotations.annot.content: true
$.aggregated.pages.page.annotations.annot.created: true
$.aggregated.pages.page.annotations.annot.filename: true
```

```
$.aggregated.pages.page.annotations.annot.in_reply_to: true
$.aggregated.pages.page.annotations.annot.intent: true
$.aggregated.pages.page.annotations.annot.last_modified: true
$.aggregated.pages.page.annotations.annot.layer: true
$.aggregated.pages.page.annotations.annot.name: true
$.aggregated.pages.page.annotations.annot.quadpoints: true
$.aggregated.pages.page.annotations.annot.subject: true
$.aggregated.pages.page.annotations.annot.text_label: true
$.aggregated.pages.page.annotations.annot.type: true
$.aggregated.pages.page.annotations.annot.unique_id: true
$.aggregated.pages.page.annotations.annot.url: true
$.aggregated.pages.page.annotations.annot: true
$.aggregated.pages.page.annotations.num: true
$.aggregated.pages.page.annotations: true

$.aggregated.pages.page.contentstream.size: true
$.aggregated.pages.page.contentstream.size_wfx: true
$.aggregated.pages.page.contentstream: true

$.aggregated.pages.page.geometry.ArtBox.bottom: true
$.aggregated.pages.page.geometry.ArtBox.height: true
$.aggregated.pages.page.geometry.ArtBox.height_eff: true
$.aggregated.pages.page.geometry.ArtBox.left: true
$.aggregated.pages.page.geometry.ArtBox.present: true
$.aggregated.pages.page.geometry.ArtBox.right: true
$.aggregated.pages.page.geometry.ArtBox.top: true
$.aggregated.pages.page.geometry.ArtBox.width: true
$.aggregated.pages.page.geometry.ArtBox.width_eff: true
$.aggregated.pages.page.geometry.ArtBox: true

$.aggregated.pages.page.geometry.BleedBox.bottom: true
$.aggregated.pages.page.geometry.BleedBox.height: true
$.aggregated.pages.page.geometry.BleedBox.height_eff: true
$.aggregated.pages.page.geometry.BleedBox.left: true
$.aggregated.pages.page.geometry.BleedBox.present: true
$.aggregated.pages.page.geometry.BleedBox.right: true
$.aggregated.pages.page.geometry.BleedBox.top: true
$.aggregated.pages.page.geometry.BleedBox.width: true
$.aggregated.pages.page.geometry.BleedBox.width_eff: true
$.aggregated.pages.page.geometry.BleedBox: true

$.aggregated.pages.page.geometry.CropBox.bottom: true
$.aggregated.pages.page.geometry.CropBox.height: true
```

```
$.aggregated.pages.page.geometry.CropBox.height_eff: true
$.aggregated.pages.page.geometry.CropBox.left: true
$.aggregated.pages.page.geometry.CropBox.present: true
$.aggregated.pages.page.geometry.CropBox.right: true
$.aggregated.pages.page.geometry.CropBox.top: true
$.aggregated.pages.page.geometry.CropBox.width: true
$.aggregated.pages.page.geometry.CropBox.width_eff: true
$.aggregated.pages.page.geometry.CropBox: true

$.aggregated.pages.page.geometry.MediaBox.bottom: true
$.aggregated.pages.page.geometry.MediaBox.height: true
$.aggregated.pages.page.geometry.MediaBox.height_eff: true
$.aggregated.pages.page.geometry.MediaBox.left: true
$.aggregated.pages.page.geometry.MediaBox.present: true
$.aggregated.pages.page.geometry.MediaBox.right: true
$.aggregated.pages.page.geometry.MediaBox.top: true
$.aggregated.pages.page.geometry.MediaBox.width: true
$.aggregated.pages.page.geometry.MediaBox.width_eff: true
$.aggregated.pages.page.geometry.MediaBox: true

$.aggregated.pages.page.geometry.TrimBox.bottom: true
$.aggregated.pages.page.geometry.TrimBox.height: true
$.aggregated.pages.page.geometry.TrimBox.height_eff: true
$.aggregated.pages.page.geometry.TrimBox.left: true
$.aggregated.pages.page.geometry.TrimBox.present: true
$.aggregated.pages.page.geometry.TrimBox.right: true
$.aggregated.pages.page.geometry.TrimBox.top: true
$.aggregated.pages.page.geometry.TrimBox.width: true
$.aggregated.pages.page.geometry.TrimBox.width_eff: true
$.aggregated.pages.page.geometry.TrimBox: true

$.aggregated.pages.page.geometry: true

$.aggregated.pages.page.info.pagelabel: true
$.aggregated.pages.page.info.pagenum: true
$.aggregated.pages.page.info: true

$.aggregated.pages.page.pieceinfo: true

$.aggregated.pages.page.resources.color.color_plates.length: true
$.aggregated.pages.page.resources.color.color_plates.plates: true
$.aggregated.pages.page.resources.color.color_plates: true
```

```
$.aggregated.pages.page.resources.color.colorspaces.colorspace: true
$.aggregated.pages.page.resources.color.colorspaces.icc_source_profiles: true
$.aggregated.pages.page.resources.color.colorspaces.length: true
$.aggregated.pages.page.resources.color.colorspaces: true

$.aggregated.pages.page.resources.color.default_colorspaces: true
$.aggregated.pages.page.resources.color.default_colorspaces.colorspace: true
$.aggregated.pages.page.resources.color.default_colorspaces.icc_source_profiles:
true
$.aggregated.pages.page.resources.color.default_colorspaces.length: true

$.aggregated.pages.page.resources.color.spotcolors.length: true"
$.aggregated.pages.page.resources.color.spotcolors.spotcolor.alternatespace: true
$.aggregated.pages.page.resources.color.spotcolors.spotcolor.alternatevalues: true
$.aggregated.pages.page.resources.color.spotcolors.spotcolor.name: true
$.aggregated.pages.page.resources.color.spotcolors.spotcolor: true
$.aggregated.pages.page.resources.color.spotcolors: true

$.aggregated.pages.page.resources.color.summary: true

$.aggregated.pages.page.resources.color: true

$.aggregated.pages.page.resources.fonts.font.embedded: true
$.aggregated.pages.page.resources.fonts.font.fonttype: true
$.aggregated.pages.page.resources.fonts.font.name: true
$.aggregated.pages.page.resources.fonts.font.subset: true
$.aggregated.pages.page.resources.fonts.font: true
$.aggregated.pages.page.resources.fonts.length: true
$.aggregated.pages.page.resources.fonts: true

$.aggregated.pages.page.resources.images.summary.bitmap_images.eff_highest_ppi:
true
$.aggregated.pages.page.resources.images.summary.bitmap_images.eff_lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.highest_pixel_h:
true
$.aggregated.pages.page.resources.images.summary.bitmap_images.highest_pixel_v:
true
$.aggregated.pages.page.resources.images.summary.bitmap_images.highest_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.lowest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.lowest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images.number: true
$.aggregated.pages.page.resources.images.summary.bitmap_images: true
```

```
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.
eff_highest_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.
eff_lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.high-
est_pixel_h: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.high-
est_pixel_v: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.high-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.low-
est_pixel_h: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.low-
est_pixel_v: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks.num-
ber: true
$.aggregated.pages.page.resources.images.summary.bitmap_images_in_softmasks: true

$.aggregated.pages.page.resources.images.summary.colorspaces: true

$.aggregated.pages.page.resources.images.summary.ct_images.eff_highest_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images.eff_lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images.highest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.ct_images.highest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.ct_images.highest_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images.lowest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.ct_images.lowest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.ct_images.lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images.number: true
$.aggregated.pages.page.resources.images.summary.ct_images: true

$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.eff_high-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.eff_low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.high-
est_pixel_h: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.high-
est_pixel_v: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.high-
```

```
est_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.lowest_pix-
el_h: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.lowest_pix-
el_v: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks.number:
true
$.aggregated.pages.page.resources.images.summary.ct_images_in_softmasks: true

$.aggregated.pages.page.resources.images.summary.imagemasks.eff_highest_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks.eff_lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks.highest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.imagemasks.highest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.imagemasks.highest_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks.lowest_pixel_h: true
$.aggregated.pages.page.resources.images.summary.imagemasks.lowest_pixel_v: true
$.aggregated.pages.page.resources.images.summary.imagemasks.lowest_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks.number: true
$.aggregated.pages.page.resources.images.summary.imagemasks: true

$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.eff_high-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.eff_low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.high-
est_pixel_h: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.high-
est_pixel_v: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.high-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.low-
est_pixel_h: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.low-
est_pixel_v: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.low-
est_ppi: true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks.number:
true
$.aggregated.pages.page.resources.images.summary.imagemasks_in_softmasks: true

$.aggregated.pages.page.resources.images.summary.number: true
```

```
$.aggregated.pages.page.resources.images.summary: true

$.aggregated.pages.page.resources.images: true

$.aggregated.pages.page.resources.form_xobjects: true
$.aggregated.pages.page.resources.form_xobjects.inherited_resources: true

$.aggregated.pages.page.resources.overprint.uses_opm: true
$.aggregated.pages.page.resources.overprint.uses_overprint: true
$.aggregated.pages.page.resources.overprint: true

$.aggregated.pages.page.resources.transparency.blendmodes.color: true
$.aggregated.pages.page.resources.transparency.blendmodes.colorburn: true
$.aggregated.pages.page.resources.transparency.blendmodes.colordodge: true
$.aggregated.pages.page.resources.transparency.blendmodes.darken: true
$.aggregated.pages.page.resources.transparency.blendmodes.difference: true
$.aggregated.pages.page.resources.transparency.blendmodes.exclusion: true
$.aggregated.pages.page.resources.transparency.blendmodes.hardlight: true
$.aggregated.pages.page.resources.transparency.blendmodes.hue: true
$.aggregated.pages.page.resources.transparency.blendmodes.lighten: true
$.aggregated.pages.page.resources.transparency.blendmodes.luminosity: true
$.aggregated.pages.page.resources.transparency.blendmodes.multiply: true
$.aggregated.pages.page.resources.transparency.blendmodes.non_normal: true
$.aggregated.pages.page.resources.transparency.blendmodes.overlay: true
$.aggregated.pages.page.resources.transparency.blendmodes.saturation: true
$.aggregated.pages.page.resources.transparency.blendmodes.screen: true
$.aggregated.pages.page.resources.transparency.blendmodes.softlight: true
$.aggregated.pages.page.resources.transparency.blendmodes: true

$.aggregated.pages.page.resources.transparency.groups.isolated: true
$.aggregated.pages.page.resources.transparency.groups.isolated_devicecmyk: true
$.aggregated.pages.page.resources.transparency.groups.isolated_devicegray: true
$.aggregated.pages.page.resources.transparency.groups.isolated_devicergb: true
$.aggregated.pages.page.resources.transparency.groups.isolated_icc_cmyk: true
$.aggregated.pages.page.resources.transparency.groups.isolated_icc_gray: true
$.aggregated.pages.page.resources.transparency.groups.isolated_icc_rgb: true
$.aggregated.pages.page.resources.transparency.groups.isolated_non_cmyk: true
$.aggregated.pages.page.resources.transparency.groups.isolated_non_devicecmyk: true
$.aggregated.pages.page.resources.transparency.groups.non_isolated: true
$.aggregated.pages.page.resources.transparency.groups: true

$.aggregated.pages.page.resources.transparency.has_explicit_transparency: true
$.aggregated.pages.page.resources.transparency.has_transparency: true
```

```
$.aggregated.pages.page.resources.transparency.opacity_less_than_1: true
$.aggregated.pages.page.resources.transparency.softmasks.softmasks_any: true
$.aggregated.pages.page.resources.transparency.softmasks.softmasks_in_extgstate:
true
$.aggregated.pages.page.resources.transparency.softmasks.softmasks_in_image: true
$.aggregated.pages.page.resources.transparency.softmasks: true
$.aggregated.pages.page.resources.transparency: true

$.aggregated.pages.page.resources: true

$.aggregated.pages.page: true
$.aggregated.pages.page[1]: true

$.aggregated.pages.transparency_group.first_page: true
$.aggregated.pages.transparency_group.num: true
$.aggregated.pages.transparency_group: true

$.aggregated.pages.TrimBox.height_max: true
$.aggregated.pages.TrimBox.height_min: true
$.aggregated.pages.TrimBox.num: true
$.aggregated.pages.TrimBox.num_landscape: true
$.aggregated.pages.TrimBox.num_portrait: true
$.aggregated.pages.TrimBox.num_square: true
$.aggregated.pages.TrimBox.width_max: true
$.aggregated.pages.TrimBox.width_min: true
$.aggregated.pages.TrimBox: true

$.aggregated.pages.userunit.max: true
$.aggregated.pages.userunit.min: true
$.aggregated.pages.userunit.num: true
$.aggregated.pages.userunit: true

$.aggregated.pages: true

$.aggregated.resources.color.color_plates.length: true
$.aggregated.resources.color.color_plates.plates: true
$.aggregated.resources.color.color_plates: true

$.aggregated.resources.color.colorspaces.colorspace: true
$.aggregated.resources.color.colorspaces.icc_source_profiles: true
$.aggregated.resources.color.colorspaces.length: true
$.aggregated.resources.color.colorspaces: true
```

```
$.aggregated.resources.color.default_colorspaces: true
$.aggregated.resources.color.default_colorspaces.colorspace: true
$.aggregated.resources.color.default_colorspaces.icc_source_profiles: true
$.aggregated.resources.color.default_colorspaces.length: true


$.aggregated.resources.color.spotcolors.length: true

$.aggregated.resources.color.spotcolors.spotcolor.alternatespace: true
$.aggregated.resources.color.spotcolors.spotcolor.alternatevalues: true
$.aggregated.resources.color.spotcolors.spotcolor.name: true
$.aggregated.resources.color.spotcolors.spotcolor: true
$.aggregated.resources.color.spotcolors: true

$.aggregated.resources.color.summary: true

$.aggregated.resources.color: true

$.aggregated.resources.fonts.font.embedded: true
$.aggregated.resources.fonts.font.fonttype: true
$.aggregated.resources.fonts.font.name: true
$.aggregated.resources.fonts.font.subset: true
$.aggregated.resources.fonts.font: true
$.aggregated.resources.fonts.length: true
$.aggregated.resources.fonts: true

$.aggregated.resources.form_xobjects: true
$.aggregated.resources.form_xobjects.inherited_resources: true

$.aggregated.resources.images.summary.bitmap_images.eff_highest_ppi: true
$.aggregated.resources.images.summary.bitmap_images.eff_lowest_ppi: true
$.aggregated.resources.images.summary.bitmap_images.highest_pixel_h: true
$.aggregated.resources.images.summary.bitmap_images.highest_pixel_v: true
$.aggregated.resources.images.summary.bitmap_images.highest_ppi: true
$.aggregated.resources.images.summary.bitmap_images.lowest_pixel_h: true
$.aggregated.resources.images.summary.bitmap_images.lowest_pixel_v: true
$.aggregated.resources.images.summary.bitmap_images.lowest_ppi: true
$.aggregated.resources.images.summary.bitmap_images.number: true
$.aggregated.resources.images.summary.bitmap_images: true

$.aggregated.resources.images.summary.bitmap_images_in_softmasks.eff_highest_ppi:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.eff_lowest_ppi:
```

```
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.highest_pixel_h:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.highest_pixel_v:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.highest_ppi: true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.lowest_pixel_h:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.lowest_pixel_v:
true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.lowest_ppi: true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks.number: true
$.aggregated.resources.images.summary.bitmap_images_in_softmasks: true

$.aggregated.resources.images.summary.colorspaces: true
$.aggregated.resources.images.summary.colorspaces.length: true

$.aggregated.resources.images.summary.ct_images.eff_highest_ppi: true
$.aggregated.resources.images.summary.ct_images.eff_lowest_ppi: true
$.aggregated.resources.images.summary.ct_images.highest_pixel_h: true
$.aggregated.resources.images.summary.ct_images.highest_pixel_v: true
$.aggregated.resources.images.summary.ct_images.highest_ppi: true
$.aggregated.resources.images.summary.ct_images.lowest_pixel_h: true
$.aggregated.resources.images.summary.ct_images.lowest_pixel_v: true
$.aggregated.resources.images.summary.ct_images.lowest_ppi: true
$.aggregated.resources.images.summary.ct_images.number: true
$.aggregated.resources.images.summary.ct_images: true

$.aggregated.resources.images.summary.ct_images_in_softmasks.eff_highest_ppi: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.eff_lowest_ppi: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.highest_pixel_h: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.highest_pixel_v: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.highest_ppi: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.lowest_pixel_h: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.lowest_pixel_v: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.lowest_ppi: true
$.aggregated.resources.images.summary.ct_images_in_softmasks.number: true
$.aggregated.resources.images.summary.ct_images_in_softmasks: true

$.aggregated.resources.images.summary.imagemasks.eff_highest_ppi: true
$.aggregated.resources.images.summary.imagemasks.eff_lowest_ppi: true
$.aggregated.resources.images.summary.imagemasks.highest_pixel_h: true
$.aggregated.resources.images.summary.imagemasks.highest_pixel_v: true
```

```
$.aggregated.resources.images.summary.imagemasks.highest_ppi: true
$.aggregated.resources.images.summary.imagemasks.lowest_pixel_h: true
$.aggregated.resources.images.summary.imagemasks.lowest_pixel_v: true
$.aggregated.resources.images.summary.imagemasks.lowest_ppi: true
$.aggregated.resources.images.summary.imagemasks.number: true
$.aggregated.resources.images.summary.imagemasks: true

$.aggregated.resources.images.summary.imagemasks_in_softmasks.eff_highest_ppi: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.eff_lowest_ppi: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.highest_pixel_h: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.highest_pixel_v: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.highest_ppi: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.lowest_pixel_h: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.lowest_pixel_v: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.lowest_ppi: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks.number: true
$.aggregated.resources.images.summary.imagemasks_in_softmasks: true

$.aggregated.resources.images.summary.number: true

$.aggregated.resources.images.summary: true

$.aggregated.resources.images: true

$.aggregated.resources.overprint.uses_opm: true
$.aggregated.resources.overprint.uses_overprint: true
$.aggregated.resources.overprint: true

$.aggregated.resources.transparency.blendmodes.color: true
$.aggregated.resources.transparency.blendmodes.colorburn: true
$.aggregated.resources.transparency.blendmodes.colordodge: true
$.aggregated.resources.transparency.blendmodes.darken: true
$.aggregated.resources.transparency.blendmodes.difference: true
$.aggregated.resources.transparency.blendmodes.exclusion: true
$.aggregated.resources.transparency.blendmodes.hardlight: true
$.aggregated.resources.transparency.blendmodes.hue: true
$.aggregated.resources.transparency.blendmodes.lighten: true
$.aggregated.resources.transparency.blendmodes.luminosity: true
$.aggregated.resources.transparency.blendmodes.multiply: true
$.aggregated.resources.transparency.blendmodes.non_normal: true
$.aggregated.resources.transparency.blendmodes.overlay: true
$.aggregated.resources.transparency.blendmodes.saturation: true
$.aggregated.resources.transparency.blendmodes.screen: true
```

```
$.aggregated.resources.transparency.blendmodes.softlight: true
$.aggregated.resources.transparency.blendmodes: true

$.aggregated.resources.transparency.groups.isolated: true
$.aggregated.resources.transparency.groups.isolated_devicecmyk: true
$.aggregated.resources.transparency.groups.isolated_devicegray: true
$.aggregated.resources.transparency.groups.isolated_devicergb: true
$.aggregated.resources.transparency.groups.isolated_icc_cmyk: true
$.aggregated.resources.transparency.groups.isolated_icc_gray: true
$.aggregated.resources.transparency.groups.isolated_icc_rgb: true
$.aggregated.resources.transparency.groups.isolated_non_cmyk: true
$.aggregated.resources.transparency.groups.isolated_non_devicecmyk: true
$.aggregated.resources.transparency.groups.non_isolated: true
$.aggregated.resources.transparency.groups: true

$.aggregated.resources.transparency.has_explicit_transparency: true
$.aggregated.resources.transparency.has_transparency: true
$.aggregated.resources.transparency.opacity_less_than_1: true

$.aggregated.resources.transparency.softmasks.softmasks_any: true
$.aggregated.resources.transparency.softmasks.softmasks_in_extgstate: true
$.aggregated.resources.transparency.softmasks.softmasks_in_image: true
$.aggregated.resources.transparency.softmasks: true

$.aggregated.resources.transparency: true

$.aggregated.resources: true
```

Click to copy

Hints and tricks

If a dot (.) or colon (:) occurs in a filter path identifier, then this glyph must be escaped with a preceding backslash (\), e.g..:

- $.direct.Root.Private.Test\:2\:Colon : true
- $.direct.Root.Private.Test\.2\.Points : true

28.5 "direct" data structures and output

Filtering for data substructures or elements inside the "direct" data structure usually requires intimate knowledge of PDF syntax. A few examples are given below:

```
$.direct.Root: false
$.direct.Info: true
$.direct.ID: true
$.direct.Encrypt: true
```

Click to copy

> ⚠ Using a filter expression like "`$.direct:true`" or "`$.direct.Root:true`" **risks creating massive amounts of output data**, especially for non-trivial PDF files. Typically, using any of these two filter expressions will lead to output files with possibly several times the size of the original PDF file, despite the fact that actual content data – such as page descriptions or image data – are not even ever included.
>
> If you actually intend to use these two filter expressions anyway, **try them first on small and simple PDFs.**

The "direct" block output

The "direct" block is a more or less direct translation of PDF syntax into JSON syntax.

1. For requesting the *Catalog* root object, "`$.direct.Root:true`" must be used
2. For requesting entries in the trailer dictionary, such as *Info* or *ID*, use "`$.direct.Info:true`" and "`$.direct.ID:true`"

For stream dictionaries, the stream portion will be omitted.

In the Quick Check configuration, specific parts of the PDF data structure can be requested by using the respective entry names in a concatenated path expression. For example, in order to request the ExtGState dictionary for pages in a PDF, the following filter expression could be used (which only works if the `Page` objects are direct children of the `Kids` element):

```
$.direct.Root.Pages.Kids.Resources.ExtGState:: true
```

Click to copy



PDFs can include pages in very different ways – either as `Kids` entries directly under the `Pages` key. But like in real life, `Kids` can have `Kids`, and these again can also have `Kids`. This makes it very unpredictable to actually locate where pages of interest can be found in the PDF data structure. Of course one could simply retrieve any data below the top most `Pages` entry – but this create massive output for any not so small multi-page PDF files, and would also require undue burden on JavaScript code that would have to parse and interpret the collected data.

Future versions of pdfToolbox will offer more elegant ways to walk nested trees of arrays, but for now the current approach has to be accepted as a known limitation.



Currently there is no mechanism to retrieve data inside stream objects. Usually this is not much of a problem – Quick Check is not the right approach to, for example, retrieve raw image data. There is at least one type of data that exists in stream objects: XMP metadata. In some scenarios it might be useful to be able to retrieve raw XMP metadata in the context of using Quick Check. For now this is not supported. Depending on user demand, we may add extended capabilities in future versions of pdfTool-

box. If this is of interest to you, please get in touch via our support email address, support@callassoftware.com, and please make us understand why this would matter to you.

Example of complete "direct" output from a simple 1 page PDF



simple_pdfToolbox_10_sample_file.pdf


```
{
  "direct": {
    "Root": {
      "Metadata" : {
        "Type" : "Metadata",
        "Length" : 51198,
        "Subtype" : "XML"},
      "OCProperties" : {
        "D" : {
          "Name" : "D",
          "ON" : [
             {
              "Name" : "Image layer",
              "Type" : "OCG",
              "Intent" : [ "View", "Design"],
              "Usage" : {
                "CreatorInfo" : {
                  "Creator" : "Adobe Illustrator 22.1",
                  "Subtype" : "Artwork"}}},
             {
              "Name" : "Text layer",
              "Type" : "OCG",
              "Intent" : [ "View", "Design"],
              "Usage" : {
                "CreatorInfo" : {
                  "Creator" : "Adobe Illustrator 22.1",
                  "Subtype" : "Artwork"}}}],
```

```
      "Order" : [],
      "RBGroups" : []},
    "OCGs" : []},
  "OutputIntents" : [
     {
      "Info" : "U.S. Web Coated (SWOP) v2",
      "DestOutputProfile" : {
        "Length" : 557168,
        "N" : 4
      },
      "OutputCondition" : "",
      "OutputConditionIdentifier" : "CGATS TR 001",
      "RegistryName" : "http://www.color.org",
      "S" : "GTS_PDFX",
      "Type" : "OutputIntent"
    }
  ],
  "Type" : "Catalog",
  "Pages" : {
    "Type" : "Pages",
    "Count" : 1,
    "Kids" : [
       {
        "Type" : "Page",
        "BleedBox" : [ 0.000000, 0.000000, 400.000000, 300.000000],
        "Contents" : {
          "Length" : 931,
          "Filter" : "FlateDecode"},
        "CropBox" : [ 0.000000, 0.000000, 400.000000, 300.000000],
        "Group" : {
          "S" : "Transparency",
          "CS" : "DeviceCMYK",
          "I" : "false",
          "K" : "false"},
        "MediaBox" : [ 0.000000, 0.000000, 400.000000, 300.000000],
        "Resources" : {
          "ColorSpace" : {
            "CS0" : [
               "ICCBased",
               {
                "Length" : 2574,
                "Filter" : "FlateDecode",
                "N" : 3
```

```
        }
      ]
    },
    "ExtGState" : {
      "GS0" : {
        "Type" : "ExtGState",
        "AIS" : "false",
        "BM" : "Normal",
        "CA" : 1.000000,
        "OP" : "false",
        "OPM" : 1,
        "SA" : "true",
        "SMask" : "None",
        "ca" : 1.000000,
        "op" : "false"},
      "GS1" : {
        "Type" : "ExtGState",
        "AIS" : "false",
        "BM" : "Normal",
        "CA" : 0.600006,
        "OP" : "false",
        "OPM" : 1,
        "SA" : "true",
        "SMask" : "None",
        "ca" : 0.600006,
        "op" : "false"}},
    "Properties" : {},
    "Shading" : {
      "Sh0" : {
        "AntiAlias" : "false",
        "Coords" : [ 0.000000, 0.000000, 1.000000, 0.000000],
        "Domain" : [ 0.000000, 1.000000],
        "Extend" : [ "true", "true"],
        "Function" : {
          "Domain" : [  0.000000, 1.000000],
          "Bounds" : [],
          "Encode" : [ 0.000000, 1.000000],
          "FunctionType" : 3,
          "Functions" : [
             {
              "N" : 1.651740,
              "Domain" : [ 0.000000, 1.000000],
              "FunctionType" : 2,
```

```
                    "C0" : [ 0.749020, 0.145098, 0.250980],
                    "C1" : [ 0.000000, 0.352941, 0.725490]}]},
                "ShadingType" : 2}},
            "XObject" : {
              "Fm0" : {
                "Length" : 9272,
                "Subtype" : "Form",
                "Group" : {
                  "S" : "Transparency",
                  "Type" : "Group",
                  "I" : "false",
                  "K" : "false"},
                "Resources" : {
                  "ExtGState" : {},
                  "Font" : {
                    "TT0" : {
                      "Type" : "Font",
                      "Subtype" : "TrueType",
                      "BaseFont" : "LQUSKK+SourceSansPro-Bold",
                      "Encoding" : "WinAnsiEncoding",
                      "FirstChar" : 32,
                      "FontDescriptor" : {
                        "Type" : "FontDescriptor",
                        "Ascent" : 974,
                        "CapHeight" : 660,
                        "Descent" : -383,
                        "Flags" : 32,
                        "FontBBox" : [ -231, -383, 1223, 974],
                        "FontFamily" : "Source Sans Pro",
                        "FontFile2" : {
                          "Length" : 5645,
                          "Filter" : "FlateDecode",
                          "Length1" : 13038
                        },
                        "FontName" : "LQUSKK+SourceSansPro-Bold",
                        "FontStretch" : "Normal",
                        "FontWeight" : 700,
                        "ItalicAngle" : 0,
                        "StemV" : 148,
                        "XHeight" : 496
                      },
                      "LastChar" : 120,
                      "Widths" : [ 208, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
0, 0, 528, 528, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 556, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
573, 0, 573, 0, 341, 0, 0, 0, 0, 0, 286, 0, 0, 555, 573, 0, 0, 0, 0, 0, 0, 0,
514]}},
                    "ProcSet" : [
                       "PDF",
                       "Text"
                      ]
                    },
                  "BBox" : [ 0.000000, 78.644500, 400.000000, 17.500000],
                  "Matrix" : [ 1.000000, 0.000000, 0.000000, 1.000000, 0.000000, 0.
000000]
                  }
              }
            },
            "TrimBox" : [ 0.000000, 0.000000, 400.000000, 300.000000]}]}
    },
    "Info": {
      "CreationDate" : "D:20180329161202+02'00'",
      "Creator" : "Adobe Illustrator CC 22.1 (Macintosh)",
      "GTS_PDFXVersion" : "PDF/X-4",
      "ModDate" : "D:20180329161202+02'00'",
      "Producer" : "Adobe PDF library 15.00",
      "Title" : "simple pdfToolbox 10 sample file",
      "Trapped" : "False"
    },
    "ID": [ "97ef49f0c10247839eb4d4e368588648", "c330dc533cf143f096cd2e246bfb39dc"
]
  },
  "status": {
    "time_needed_sec" : 0.016667,
    "result" : "com-
plete"




  }
}
```


Click to copy

Hints and tricks

If a dot (.) or colon (:) occurs in a filter path identifier, then this glyph must be escaped with a preceding backslash (\), e.g..:

- $.direct.Root.Private.Test\:2\:Colon : true
- $.direct.Root.Private.Test\.2\.Points : true

28.6 "status" data structure and output

The status block is the part of the JSON output that is always created/written, even if other blocks are turned off or everything else during Quick Check execution goes wrong.

Output of the "status" block:

```
"status": {
      "time_needed_sec" : 0.05233,
      "result" : "incomplete",
      "level" : "error",
      "error" : [
            { "code": 24, "msg": "Invalid operator in content stream" }
      ],
}
```

Click to copy

Description of "status" entries:

- `time_needed_sec`: time needed (in seconds) from launching Quick Check to its completion; "status" is created/written as the last Quick Check execution step
- `result`: an indication of the quality of the result:
  - *complete*: the usually expected result – everything was analyzed and delivered according to the configuration
  - *incomplete*: only some of the requested output was created/written; for example, creating output stopped in the middle of a content stream analysis, but document info/metadata was already collected
  - *none*: except for the "status" block, no output was created
- `level`: the 'worst' level that was encountered (*none* → *info* → *warning* → *error*)
- `info`: an array of info messages, each consisting of a return code and a (non-localized) text briefly describing the info

- `warning`: an array of warning messages, each consisting of a return code and a (non-localized) text briefly describing the warning
- `error`: an array of error messages, each consisting of a return code and a (non-localized) text briefly describing the error

28.7 Using Quick Check directly on the command line



Note: As with any other use of the command line version of pdfToolbox (or actually any command line tool), at least some familiarity with the use of applications in a command line environment is required.

Looking up calling conventions for Quick Check

Using `./pdfToolbox --help quickcheck` an overview of the usage of the Quick Check feature on the command line is shown:

```
./pdfToolbox --help quickcheck
Usage:
  pdfToolbox --quickcheck [--satellite_type=satellite_type] [--timeout_satel-
lite=timeout_satellite] [--timeout_dispatcher=timeout_dispatcher] [--nolocal] [--
endpoint=endpoint] [--dist] [--timeout=timeout] [-l=l] [--cachefolder=cachefolder]
[--noprogress] [-t] [--nooptimization] [-o=o] [-f=f] [-w] [-s=s] <config file> <in-
put file>
Purpose:
  Performs a QuickCheck
Options:
      --satellite_type      Distribute to satellite with specific type
      --timeout_satellite   Time interval before processing of job is cancelled on
satellite
      --timeout_dispatcher  Time interval before search for available satellite is
cancelled (s.a. --nolocal)
      --nolocal             do not process locally, return error in case of timeout
      --endpoint            dispatcher url
      --dist                distribute execution to satellites
      --timeout             Time interval before local processing of job is can-
celled
  -l  --language            Reporting language (e.g. en (English, default), de or
fr)
      --cachefolder         Sets the cache folder path
```

```
      --noprogress          Switches off progress information
  -t  --timestamp           show time stamp in output
      --nooptimization      The internal PDF structure is not optimized
                            when saving the PDF.
  -o  --outputfile          Destination for modified input file(s)
  -f  --outputfolder        Puts modified input file(s) into folder
  -w  --overwrite           Overwrites existing files (default: index file name)
  -s  --suffix              Adds suffix to modified file(s)
Arguments:
  config file   Configuration for the QuickCheck
  input file    File to be processed
```

Click to copy

In most cases it will be sufficient to

1. point to a suitable configuration file
2. indicate the PDF file to be analyzed by Quick Check
3. where to write the resulting JSON file

Thus an example call might look like this:

```
./pdfToolbox --quickcheck -o=demo.json sample.cfg demo.pdf
```

Click to copy



Note: In the article "All aggregated Quick Check objects and output" you can find a list of all "aggregated" filter expressions which you can copy into a configuration file.

The above syntax will not overwrite existing JSON files, but instead will append a 4 digit number to a newly created JSON file if a file with the same name as the requested name already exists. In order to enforce that the JSON file is always written with the requested name (and a pre-existing file with the same name is overwritten) the following call may be used (via the `-w` option):

```
./pdfToolbox --quickcheck -o=demo.json -w sample.cfg demo.pdf
```

Click to copy

Example

Copy these contents to notepad and save the configuration file as 'sample.cfg':

```
$.direct: false
$.aggregated: false
$.aggregated.pages.page.info.pagenum: true
$.aggregated.pages.page.geometry.TrimBox.width_eff: true
```

Click to copy

Executing the command for a 4 page test PDF would result in something like:

```
{
  "aggregated": {
    "pages": {
      "page" : [
        {
          "info" : {
            "pagenum" : 1
          },
          "geometry" : {
            "TrimBox" : {
              "width_eff" : 425.197,
              "height_eff" : 651.968
            }
          }
        },
        {
          "info" : {
            "pagenum" : 2
          },
          "geometry" : {
            "TrimBox" : {
              "width_eff" : 425.197,
              "height_eff" : 651.968
            }
          }
        },
        {
          "info" : {
```

```
            "pagenum" : 3
          },
          "geometry" : {
            "TrimBox" : {
              "width_eff" : 425.197,
              "height_eff" : 651.968
            }
          }
        },
        {
          "info" : {
            "pagenum" : 4
          },
          "geometry" : {
            "TrimBox" : {
              "width_eff" : 425.197,
              "height_eff" : 651.968
            }
          }
        }
      ]
    }
  },
  "status": {
    "time_needed_sec" : 0.000002,
    "result" : "com-
plete"




  }
}
```


Click to copy

28.8 Error codes and Return codes for Quick Check Results

As described in the previous article "Structure and syntax of Quick Check output", in the "status" block some entries give more details about the result of the performed QuickCheck.

Beside the "*level*" entry, a "*returncode*" is included, which gives helpful information especially if the processing was not successful.

List of possible Error codes and Return codes

Error or Return code	Description
0	No error (successful analysis)
10	Unable to open config file for reading
11	Unable to open pdf file for reading
12	Unable to open json file for writing
20	An unknown error has occurred
21	Unable to write pdf file, file access error
22	Undefined Mediabox, minimum page size should be 3 by 3 units
23	An error has occurred in flate encoder
24	An error has occurred in flate decoder
25	An error has occurred in lzw decoder
26	An error has occurred in ASCII-85 decoder
27	An error has occurred in ASCII-Hex decoder
28	An error has occurred in RunLength decoder
29	An error has occurred in prediction decoder
30	An error has occurred in dct (jpeg-image) decoder
31	File offset is greater than 9999999999 bytes. PDF only allows offsets up to 10 decimal digits
32	The required resource name is too large, too much resources are required
33	A spot color space has no name or a wrong alternate color space type
34	A path length entry is out of range
35	There are too many PDF files open
36	Unable to open the PDF file for import
37	PDF file is not supported, it might be encrypted
38	Unable to parse pdf file, syntax error found
39	The page could not be found
40	A given parameter is invalid or wrong

Error or Return code	Description
41	The given object number is invalid
42	A inline-image contains an ID without a BI
43	A content stream contains illegal commands
44	A resource could not be found
45	A given resource is wrong
46	A given function object is wrong
47	Unable to read font resource
48	Unable to find glyphs in current font
49	An error has occured in XMP Metadata function
50	Invalid context (e.g. a context is popped wich never was pushed)
51	Page without underlay or overlay reservation
52	The keycode you have entered is invalid
53	The keycode you have entered has expired
54	Unable to draw barcode
55	Unable to open icc profile from file path
56	Unable to read all required encryption parameter
57	The page limit is exhausted
58	Too many parallel processes
59	The requested barcode is not supported
60	Unable to read all linearized hint data
61	There are too many colorants in a DeviceN color space, the limit is 32
62	Unable to execute shared resource objects process
63	Unable to open, read or write temporary file
64	Unable to process a given png image file
65	Unable to process a given gif image file
66	The given file is empty
67	File could not be identified as a PDF file
68	The PDF file is encrypted and need a password to open
100	--Exception caught-- "<CCW Exceptions Message>"
101	Unable to remove corrupt json file after fatal error occured
110	--Access violation caught-- "Fatal error - abort"
113	Conversion did not finish during user specified timeout

29. Quick Fix

29.1 Quick Fix overview

QuickFixes versus Fixups

QuickFix is a new type of PDF manipulation feature. Whereas Fixups are based on a powerful PDF analysis and modification engine with extensive customization capabilities, the QuickFix architecture is based on a comparatively lean analysis engine combined with highly specialized modification modules. The main implications are

- QuickFix is much faster than comparable Fixups
- QuickFixes are less customizable than Fixups
- QuickFixes exist only for certain PDF modification functions, there are many types of modifications that can only be achieved through Fixups
- in some cases both a QuickFix and a Fixup are available for more or less the same functionality; one would choose QuickFix if speed of the essence, and a Fixup, if customizability is more important.
- in some cases the limited degree of customizability of a QuickFix as such can be overcome by using a QuickCheck based analysis and some JavaScript to set parameters for a QuickFix on the fly

Functional areas

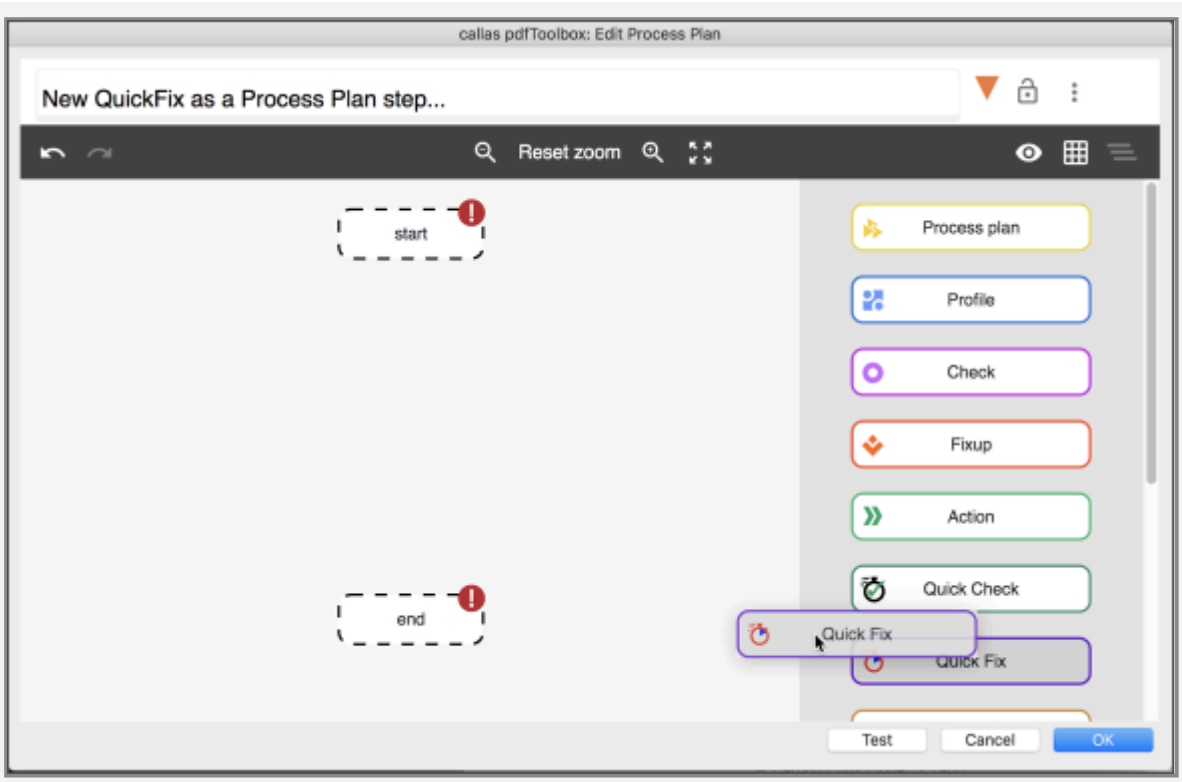
QuickFixes exists for the following functional areas:

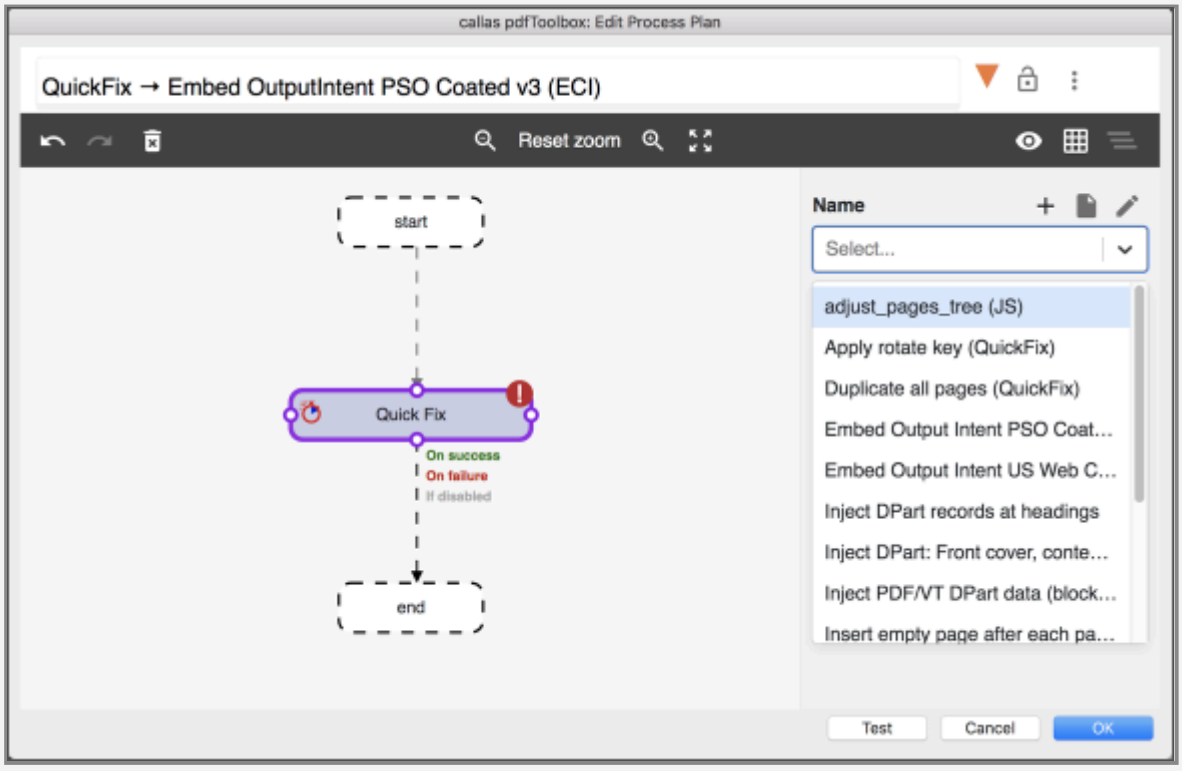
- Spot color
- Page geometry
- Scale/rotate/flip pages
- Create/duplicate/reorder/delete pages
- Layers (including Processing Steps metadata)
- Output intents
- PDF/VT DPart

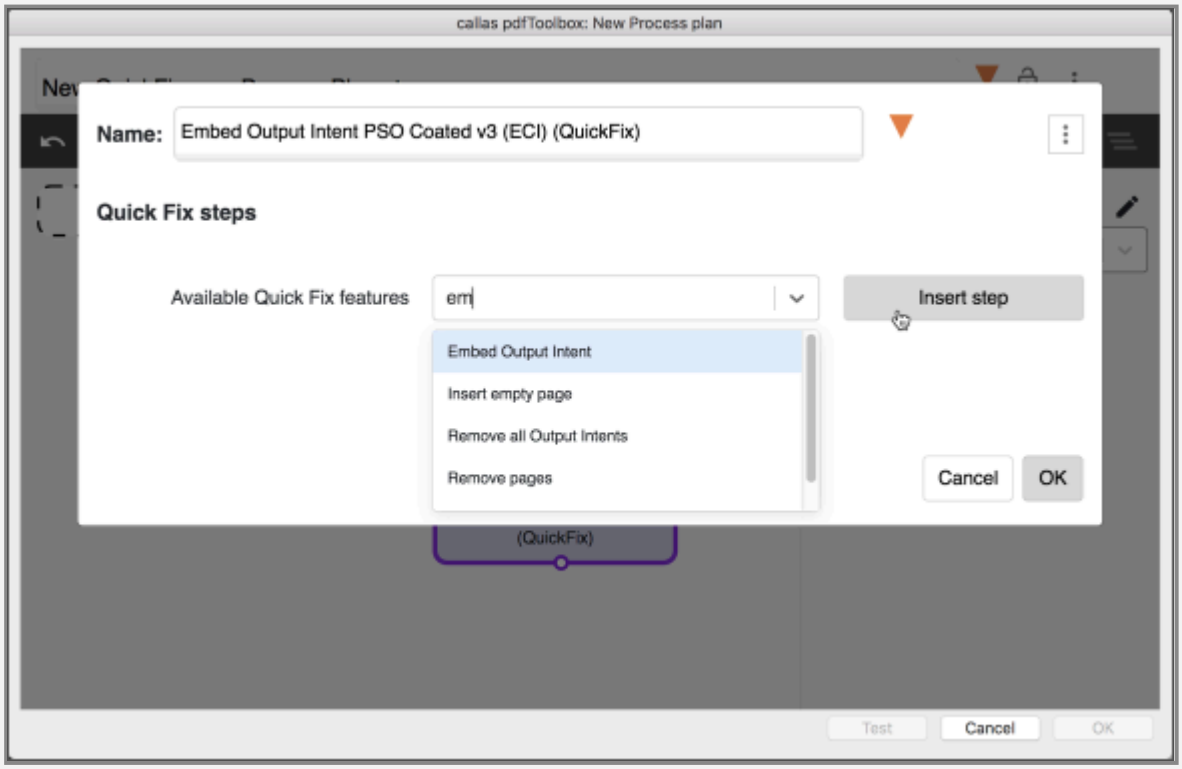
Where to find and use QuickFixes

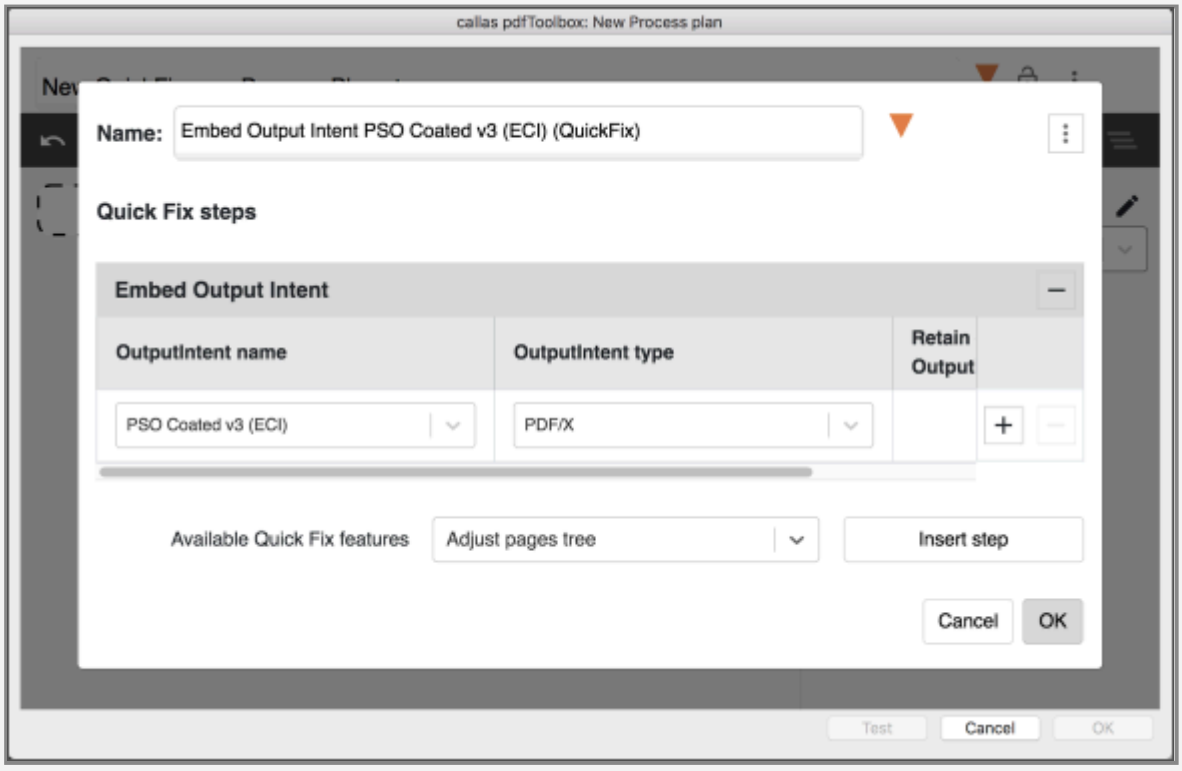
QuickFixes can be used either as a step in Process Plan or on their own (they are listed in the Fixup view). As QuickFixes are based on a completely different architecture than Fixups, they cannot be included in a Profile. In addition, QuickFix can be executed directly on the command line, using JSON files.

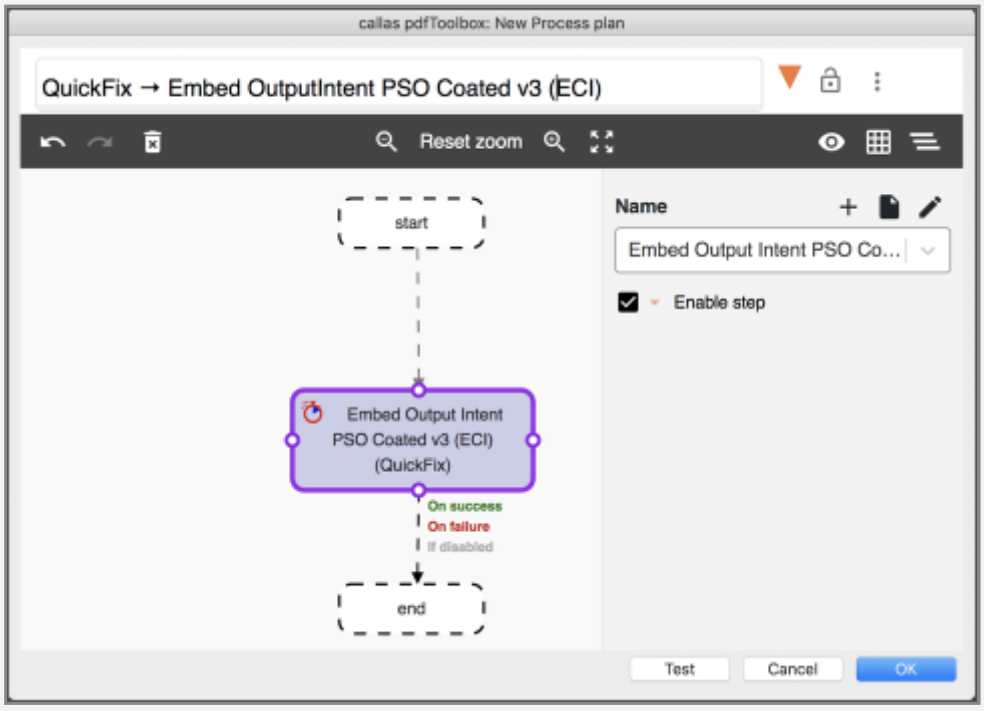
QuickFixes as Process Plan steps



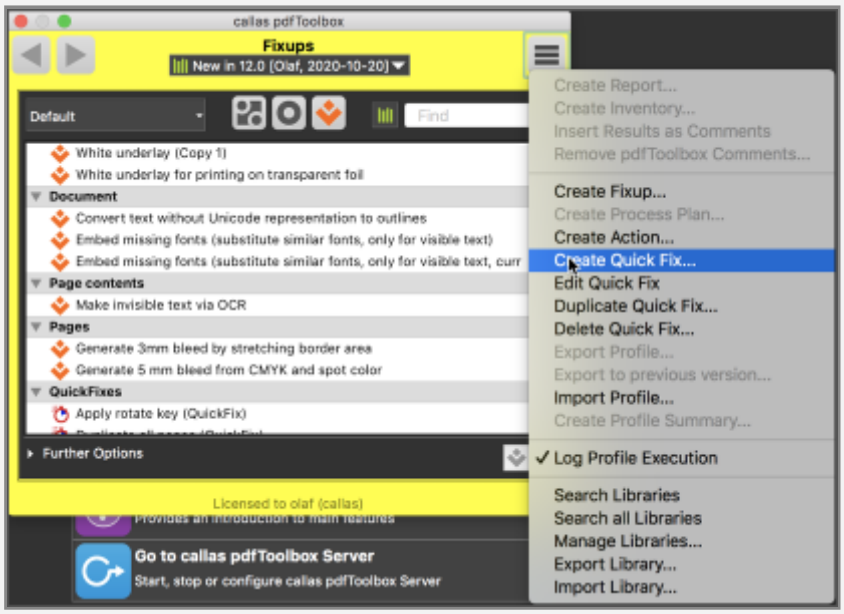

callas pdfToolbox: Edit Process Plan
QuickFix → Embed OutputIntent PSO Coated v3 (ECI)
Reset zoom
start
Quick Fix
On success
On failure
If disabled
end
Name
Select...
adjust_pages_tree (JS)
Apply rotate key (QuickFix)
Duplicate all pages (QuickFix)
Embed Output Intent PSO Coat...
Embed Output Intent US Web C...
Inject DPart records at headings
Inject DPart: Front cover, conte...
Inject PDF/VT DPart data (block...
Insert empty page after each pa...
Test
Cancel
OK

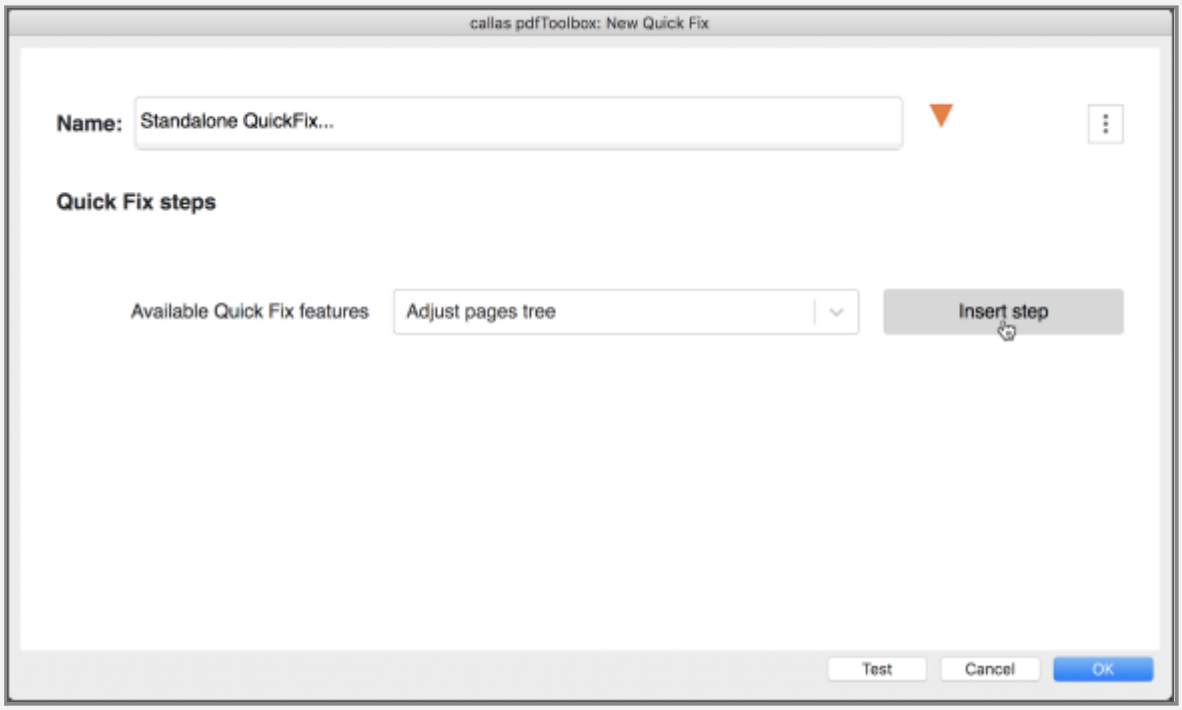

callas pdfToolbox: New Process plan
Name:
Embed Output Intent PSO Coated v3 (ECI) (QuickFix)
Quick Fix steps
Available Quick Fix features
em
Insert step
Embed Output Intent
Insert empty page
Remove all Output Intents
Remove pages
Cancel
OK
(QuickFix)
Test
Cancel
OK


callas pdfToolbox: New Process plan
Name:
Embed Output Intent PSO Coated v3 (ECI) (QuickFix)
Quick Fix steps
Embed Output Intent
OutputIntent name
OutputIntent type
Retain Output
PSO Coated v3 (ECI)
PDF/X
Available Quick Fix features
Adjust pages tree
Insert step
Cancel
OK
Test
Cancel
OK


callas pdfToolbox: New Process plan
QuickFix → Embed OutputIntent PSO Coated v3 (ECI)
Reset zoom
start
Embed Output Intent PSO Coated v3 (ECI) (QuickFix)
On success
On failure
If disabled
end
Name
Embed Output Intent PSO Co...
Enable step
Test
Cancel
OK

Standalone QuickFixes (in Fixups list)





JSON file based QuickFix mode for pdfToolbox CLI

On the command line, it is possible to execute a QuickFix directly, using a JSON file based QuickFix configuration. For details see the article Using QuickFix on the command line.

Example:

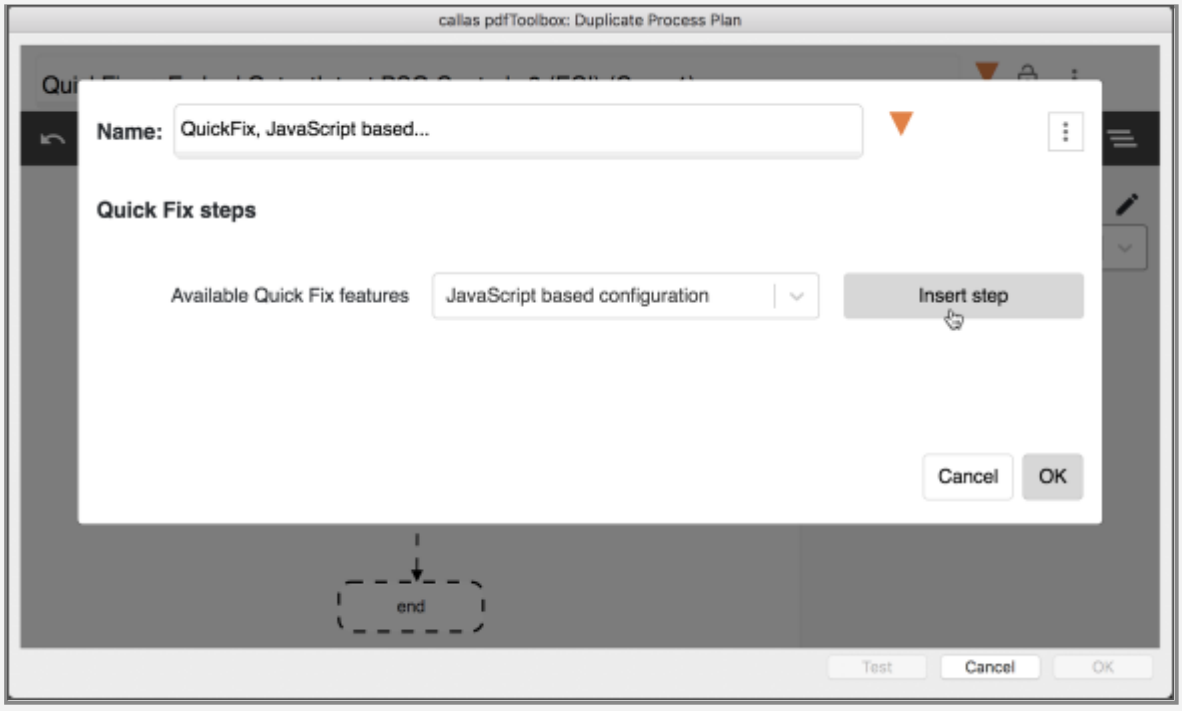
```
./pdfToolbox my_quickfix_config.json my_pdf_file.pdf
```

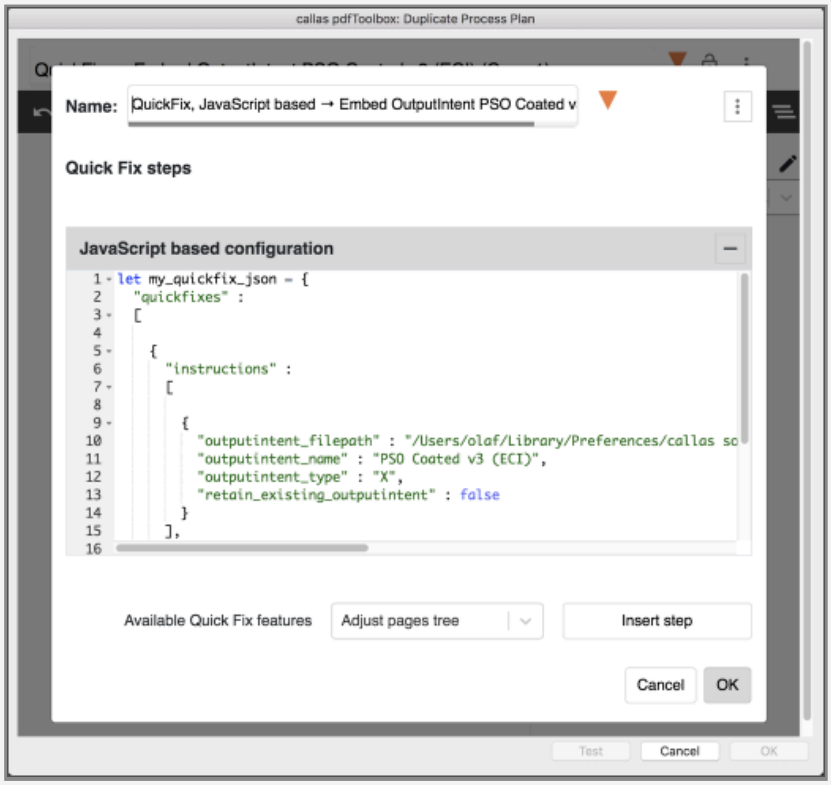
Click to copy

JavaScript based QuickFix as Process Plan steps

The below shown JavaScript based configuration must return a JavaScript object that has a JSON serialisation according to the specification for the Quick Fix features in the form as documented here.

For more information and an example of a **JavaScript based configuration** Quick Fix can be found here.




callas pdfToolbox: Duplicate Process Plan
Name:
QuickFix, JavaScript based → Embed OutputIntent PSO Coated v
Quick Fix steps
JavaScript based configuration
let my_quickfix_json = {
"quickfixes" :
[
{
"instructions" :
[
{
"outputintent_filepath" : "/Users/olaf/Library/Preferences/callas sc
"outputintent_name" : "PSO Coated v3 (ECI)",
"outputintent_type" : "X",
"retain_existing_outputintent" : false
}
],
Available Quick Fix features
Adjust pages tree
Insert step
Cancel
OK
Test
Cancel
OK

29.2 Overview Quick Fixes with corresponding Fixups

The table below provides an overview of all Quick Fixes available in pdfToolbox, along with their corresponding Fixup (if available).

While Quick Fixes offer a faster solution for certain PDF modifications, Fixups offer more flexibility and customization. So when both options exist for the same functionality, the choice depends on whether speed or customization is the priority.

Quick Fix	Fixup
Adjust pages tree	-
Adjust spot colors	Map spot and process colors
Apply 2D transformation matrix to page content	
Apply Rotate key	Rotate pages
Auto-correct page geometry boxes	Auto-correct page geometry boxes
Convert PDF internal names to UTF-8	Convert PDF internal names to UTF-8
Distribute from XObjects on layers	-
Duplicate pages	-
Embed Output Intent	Embed Output Intent
Embed Output Intent on page level	Embed Output Intent on page level
Embed Output Intent with specified parameters	Embed Output Intent with specified parameters
Embed Output Intent with specified parameters on page level	Embed Output Intent with specified parameters on page level
Enlarge pages	Enlarge pages
Flip pages	Flip pages
Inject DPart	-

Quick Fix	Fixup
Insert empty page	Insert page
Invert page order	-
JavaScript based configuration	Arbitrary JavaScript controlled Fixups
Merge spot colors	Map spot and process colors
Remove all Output Intents	Remove all Output Intents
Remove page geometry boxes	Remove page geometry boxes
Remove page level Output Intents	Remove page level Output Intents
Remove pages	Remove pages
Rename spot colors	Map spot and process colors
Reorder pages	-
Resave PDF	-
Rotate pages	Rotate pages
Scale page content only	Scale page content only
Scale pages	Scale pages
Search and replace text	-
Set MediaBox to origin	Set MediaBox to origin
Set Processing Steps metadata for layer	Add Processing Steps layer metadata
Set layer visibility	Set layer initial visibility state
Set page geometry box	Set page geometry boxes
Set page geometry box by dimensions	Set page geometry boxes

29.3 Quick Fix configuration essentials: String comparison operators and page selection expressions

General considerations

QuickFixes are optimized for speed. for example, all changes are applied to a PDF file in the form of an "incremental update operation" – changes are written to the end of PDF file in a manner that relevant existing data is logically (but not 'physically') overwritten or removed. While the PDF may become slightly larger, it does not need to be completely rewritten – a significant time saver especially for larger PDFs.

Furthermore, the QuickFix engine executes one incremental save operation per QuickFix step in a Process Plan, and combines QuickFixes of the same type – e.g renaming several spot colors or manipulating different page geometry boxes on different page ranges – into a single operation. As a consequence – whenever more than QuickFix feature is to be applied to a PDF file – it is advised to combine all needed QuickFix features into one QuickFix step inside a Process Plan, instead of having several QuickFix based Process Plan steps.

The order of execution always follows the order in which QuickFix processing instructions are defined in a QuickFix. This can be important for example for renaming spot colors – specific instructions could be defined first, and following instructions could then rename remaining spot colors in a more generic fashion.

Important configuration options

Many QuickFixes use two types of configuration options that in many cases are highly relevant for efficient execution.

String comparison operators

For some QuickFix features – for example, spot color renaming – flexible string handling is very important. For this pur-

pose, the following string operators are offered in a number of QuickFix configurations:

- “noop”
- "regex"
- "begins_with"
- "contains"
- "does_not_begin_with"
- "does_not_contain"
- "does_not_end_with"
- "ends_with"
- "equal_to"
- "is_contained_in"
- "is_not_contained_in"
- "unequal_to"

These string comparison operators work in the same way as the (similarly named) comparison operators in checks and Fixups.

Page selection

For some QuickFix features – for example, adjusting page geometry boxes – it is very important to have a flexible option to determine which pages to process and which pages not to process (e.g. only the first two and the last pages, or only even pages, or every 4th page, etc.). For this purpose, a flexible syntax is used to express the desired groups of pages or page ranges. This is the same syntax as used for split schemes (`--splitscheme=<expression>`) in the "Split and merge" feature – see Split and merge for details.

Split Scheme

A split scheme expression may be a number with an asterisk "*" or a more complex string. If it is a number with an asterisk "*" it creates groups of pages where each group has the defined number of pages. E.g. if the expression is "3*" it would group the PDF into groups of 3 pages.

Expressions

Type	Syntax	Example	
Simple expression	number[-number]	1-5	Page 1 to 5: [1,2,3,4,5]
		5-1	Page 5 to 1: [5,4,3,2,1]
		8	Only page 8
		-1	Last page
		-3--1	Last 3 pages: [n, n-1, n-2]
		-1--3	Last 3 pages in reverse order: [n-2, n-1, n]
		-1-3	Last n - 2 pages in reverse order: [n, n-1, ... ,3]
		*2(2)	[2][4][6]...
Simple expression with Simple Range	number[-number]_ number[-number]	1-2_-2--1	First and last 2 pages: [1,2,n-1,n]
		1-2_-2--1,$	First and last 2 pages [1,2,n-1,n] and remaining inner pages [3, ... ,n-2]
		1_1_1_1	4 times page 1: [1,1,1,1]

Type	Syntax	Example	
Multipage expression	even_pages even	even	All even pages (same as *2(2))
	uneven_pages uneven odd	uneven	All uneven pages (same as *2(1))
	Package number* [(start_page)]	5*	Packages of 5 pages
		5*(2)	Packages of 5 pages, starting with page 2
	Intervall *number [(start_page)]	*5	Every 5th page
		*5(2)	Every 5th page, starting with page 2
		*5(-20)	Every 5th page of the last 20 pages of a document (totally 4 pages)
Simple expression list	simple_expression {"," simple_expression} ["," joker]	1-5,8,-3--1	1 PDFs with page 1-5, page 8 and the last 3 pages of the input PDF
		1-5,8,-1--3	1 PDFs with page 1-5, page 8 and the last 3 pages of the input PDF, but the last 3 pages in reverse order

Type	Syntax	Example	
		5*(2)	Packages of 5 pages, starting with page 2
		*5(2)	Every 5th page, starting with page 2
		*5(-20)	Every 5th page of the last 20 pages of a document (totally 4 pages)

Joker

```
<expression>,$
```

Can be combined with other expressions (has to be the last item in a list) in order to group all pages that are not part of any other expression into their own group.

Example

```
1-5,8,-3--1,$
```

would create groups of pages with pages 1-5, page 8, the last 3 pages and the rest of the pages of the PDF.

29.4 Quick Fix features

This article gives an overview of all Quick Fixes included in pdfToolbox.

Since a Quick Fix can also be executed on the command line, by using a JSON file based Quick Fix configuration, this article also includes JSON serializations for each Quick Fix. Detailed information about using Quick Fix on the command line can be found here.

In addition the JSON serialisations can also be used in the "JavaScript based configuration" Quick Fix. How to do is documented here.

List of Quick Fix functional areas:

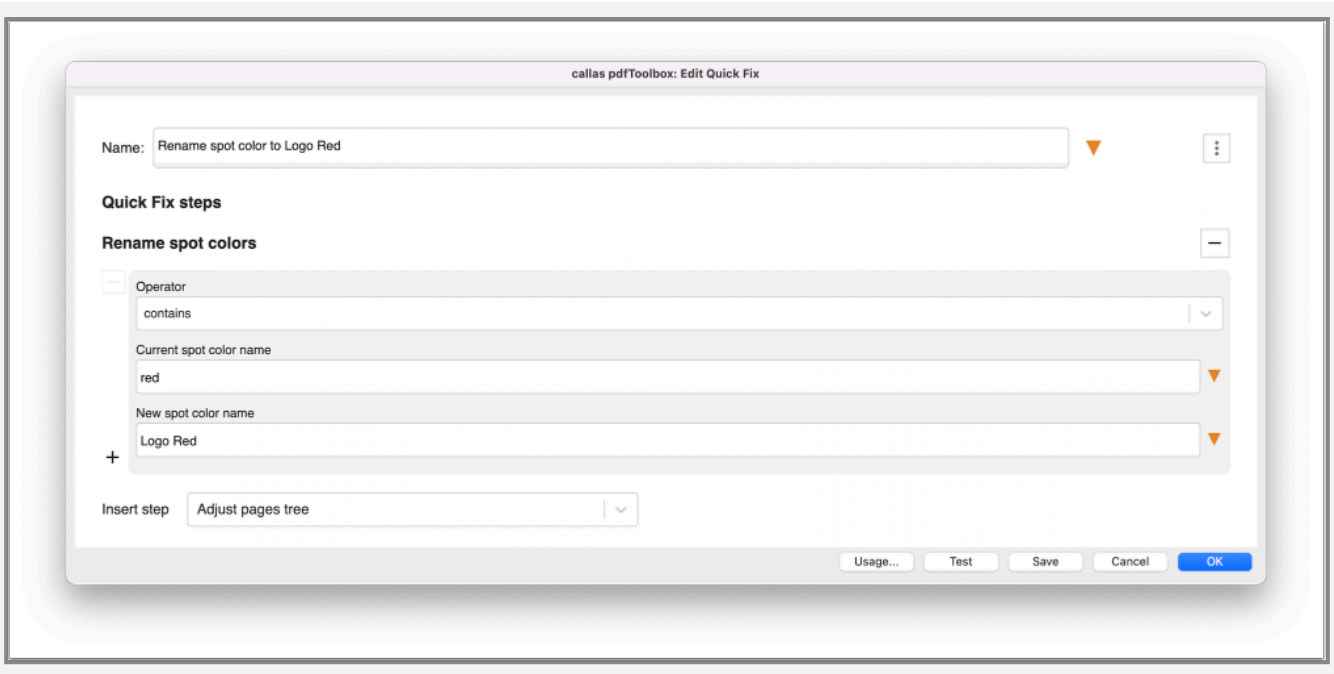
- Spot color
- Page geometry
- Scale/rotate/flip pages
- Insert/duplicate/remove/invert/reorder pages
- Layers (including Processing Steps metadata)
- Output Intents
- VDP, PDF/VT DPart
- Resave PDF
- Replace Text
- Convert PDF internal names to UTF-8
- JavaScript based configuration

Spot color Quick Fixes

Rename spot color

This Quick Fix feature simply renames spot colors. Using the operator in conjunction with the "Current spot color name" allows for flexible renaming – whether for just one specific spot color name or a list of names or mappings that could be expressed using RegEx. The table below illustrates this with a couple of examples.

Operator	Current spot color name	New spot color name	Effect
equal to	red	Logo Red	If there is a spot color whose name is exactly "red" it will be re-named to "Logo Red"
contains	red	Logo Red	All spot color names that contain "red" ("Deep red", "Fred", "ingredient", ...) will get renamed to "Logo Red". By implication all these spot colors containing "red" in their name will be merged into a single new spot color "Logo Red"
regex	.*	Logo $0	All spot colors will be renamed by prefixing their current name with "Logo ".




```
{
```

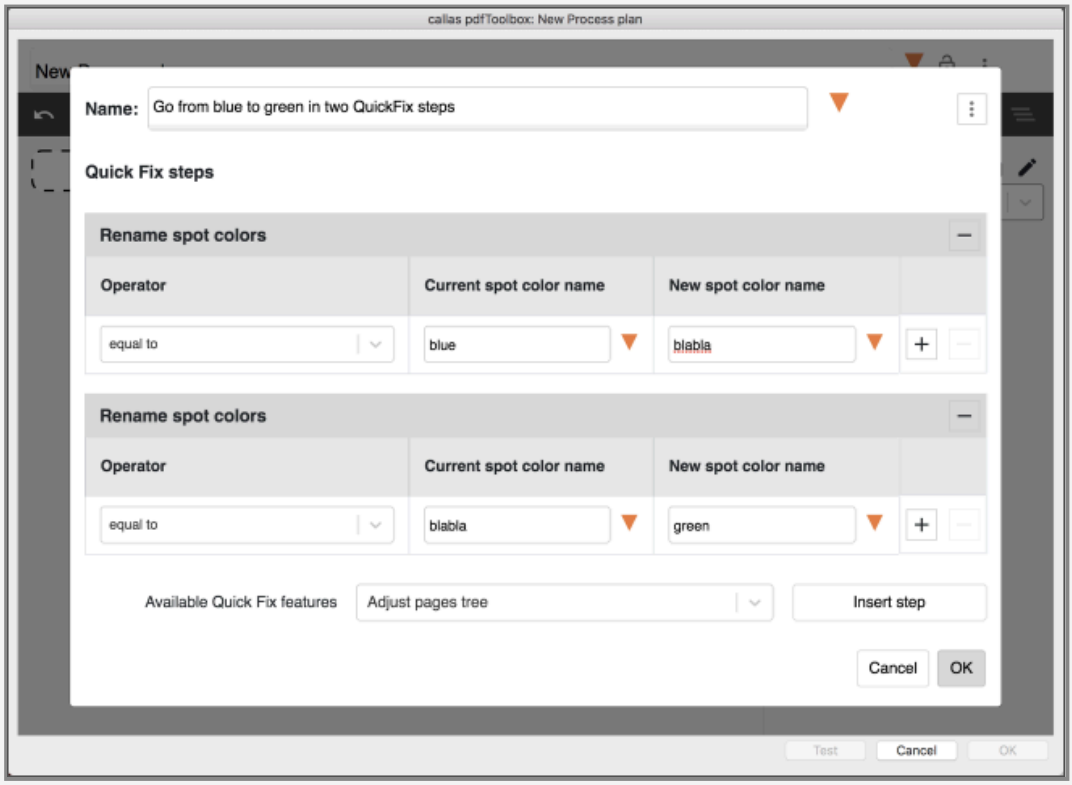
```
        "quickfixes" : [
                {
                                "quickfix" : "rename_spot",
                                "version" : "1.0"
                                "instructions" : [
                                {
                                        "new_spot" : "Logo Red",
                                        "old_spot" : "red",
                                        "operator" : "contains"
                                }
                        ]

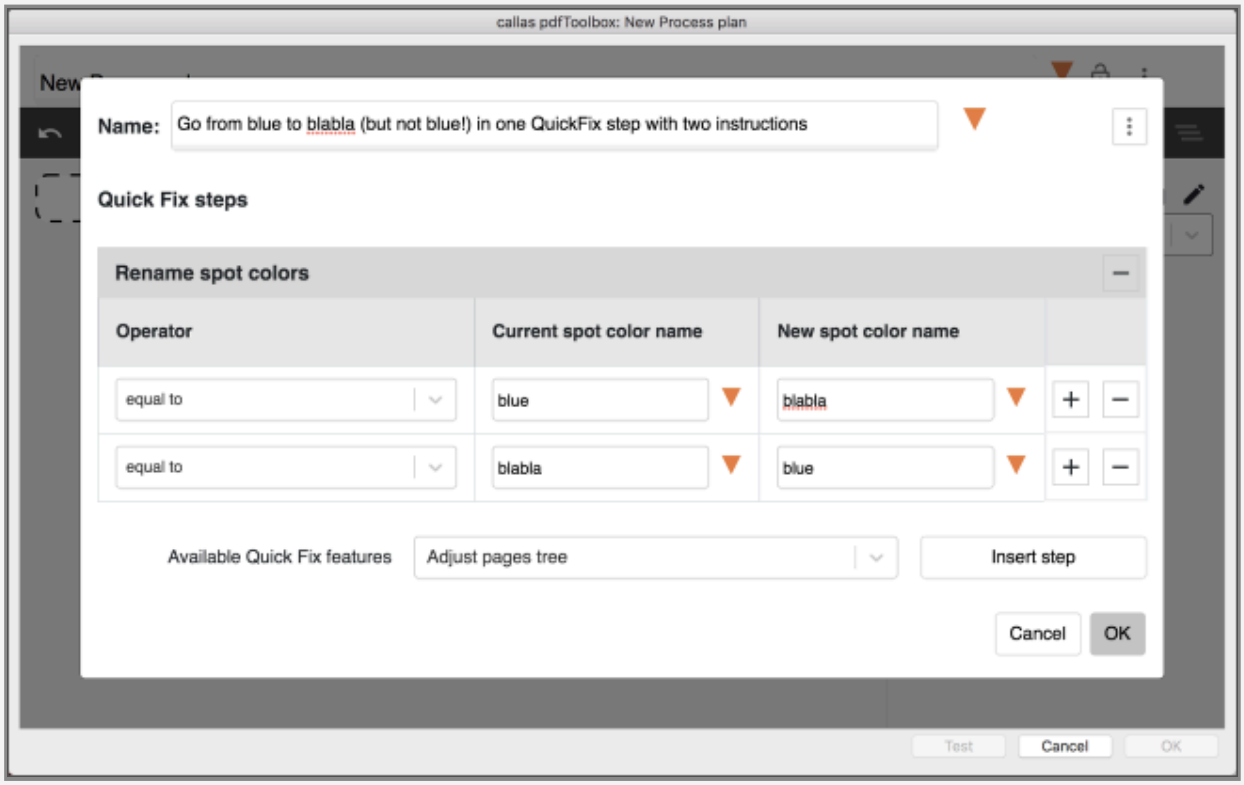
                }
        ]
}

// "operator" : "regex|begins_with|contains|does_not_begin_with|does_not_con-
tain|does_not_end_with|ends_with|equal_to|is_contained_in|is_not_contained_in|un-
equal_to"
```

Click to copy

If there are several instructions in a Quick Fix step, each spot color will only be "touched" once (if at all). If repeated modifications are desired (change a spot color name, then change that already modified spot color name again), the respective instruction have to be put inside separate Quick Fix steps.

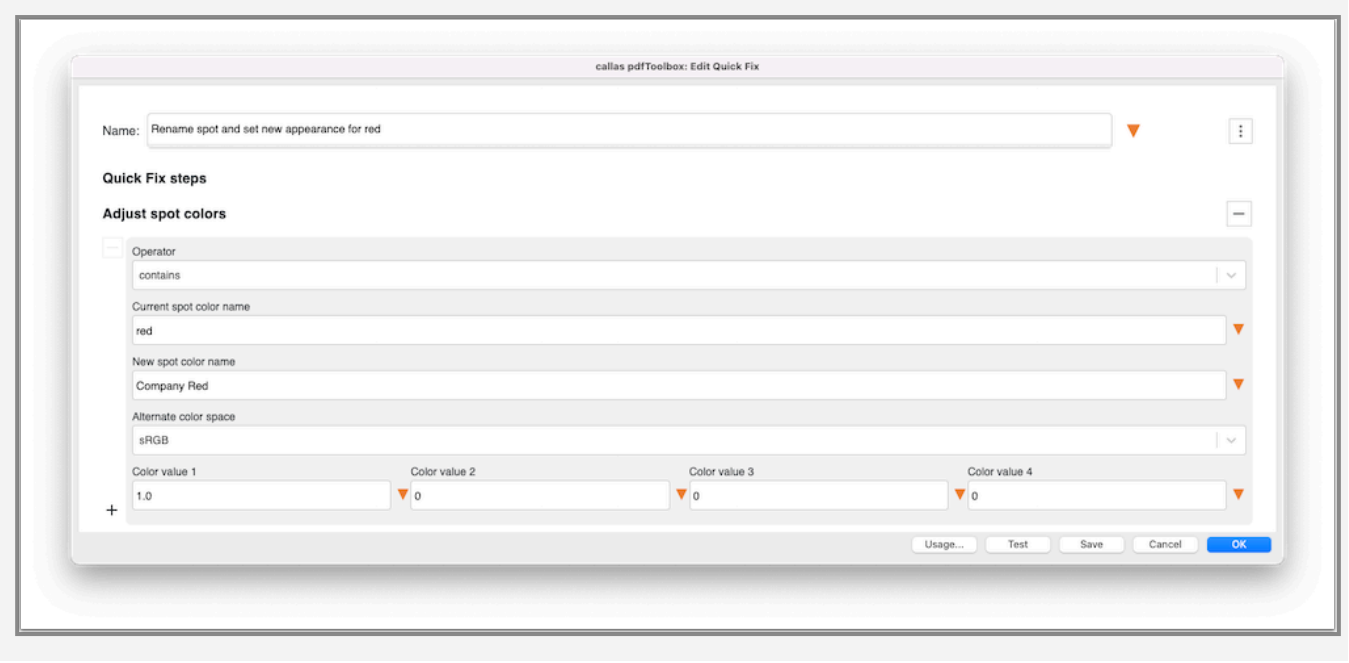

callas pdfToolbox: New Process plan
Name: Go from blue to green in two QuickFix steps
Quick Fix steps
Rename spot colors
Operator
Current spot color name
New spot color name
equal to
blue
blabla
Rename spot colors
Operator
Current spot color name
New spot color name
equal to
blabla
green
Available Quick Fix features
Adjust pages tree
Insert step
Cancel
OK
Test
Cancel
OK


callas pdfToolbox: New Process plan
Name: Go from blue to blabla (but not blue!) in one QuickFix step with two instructions
Quick Fix steps
Rename spot colors
Operator
Current spot color name
New spot color name
equal to
blue
blabla
equal to
blabla
blue
Available Quick Fix features
Adjust pages tree
Insert step
Cancel
OK
Test
Cancel
OK

Adjust spot colors

This Quick Fix is an extended version of the "Rename spot color" Quick Fix. In addition to the new spot color name it is also possible to specify the appearance of that new spot color, using DevciceCMYK, DeviceRGB, sRGB, Lab D50, Lab D65, sRGB, DeviceGray (or an ICC profile by means of an <icc-profil-file-path>) with the appropriate number of color values.

If the new spot color name is left empty, only the appearance will be changed. If the alternate color space and color values are left empty, only renaming will be applied (same effect as with the "Rename spot color" Quick Fix).




```
{
    "quickfixes": [
        {
            "quickfix": "adjust_spot",
            "version": "1.0",
            "instructions": [
                {
                    "operator": "contains",
                    "old_spot": "red",
                    "new_spot": "Company Red",
                    "alt" : "sRGB",
```

```
                    "c0" : 1.0,
                    "c1" : 0.0,
                    "c2" : 0.0,
                    "c3" : 0.0
                }
            ]
        }
    ]
}

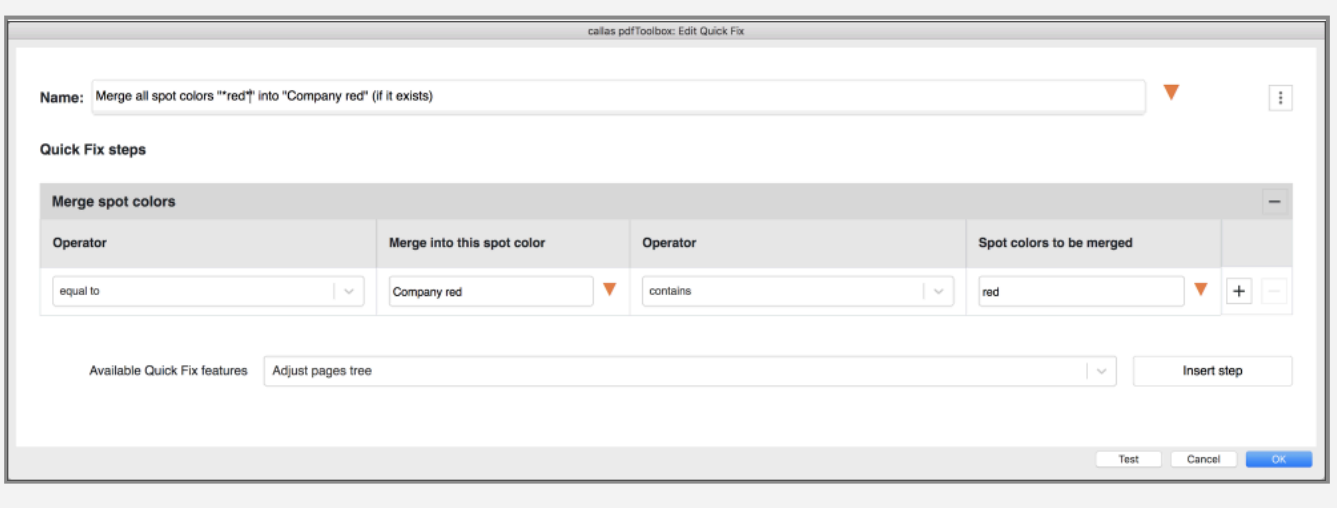
//                "operator": "regex|begins_with|contains|does_not_be-
gin_with|does_not_contain|does_not_end_with|ends_with|equal_to|is_con-
tained_in|is_not_contained_in|unequal_to"
//                "alt": "DeviceGray|DeviceRGB|De-
viceCMYK|Lab_D50|Lab_D65|sRGB|<icc-profil-file-path>"
```

Click to copy

Merge spot colors

The "Merge Spot color" Quick Fix differs from the "Rename spot color" Quick Fix insofar, as on both sides – "Spot colors to be merged" and spot color to be merged into ("Merge into this spot color") – an operator can be used.

In general, it is recommended to make the combination of operator and "Merge into this spot color" specific enough to identify that one spot color in the PDF into which the other spot colors are to be merged. The reason for this is that if more than one spot color matches the "Merge into this spot color" condition, the Quick Fix analysis will select the spot color that is identifies first. Therefore, there is a risk that a wrong spot color is selected by pdfToolbox, into which the other spot colors are then merged.




```
{
        "quickfixes" :
        [
                {
                        "instructions" :
                        [
                                {
                                        "master_spot" : "Company red",
                                        "operator1" : "equal_to",
                                        "operator2" : "contains",
                                        "slave_spot" : "red"
                                }
                        ],
                        "quickfix" : "merge_spot",
                        "version" : "1.0"
                }
        ]
}

//                "operator1": "regex|begins_with|contains|does_not_be-
gin_with|does_not_contain|does_not_end_with|ends_with|equal_to|is_con-
tained_in|is_not_contained_in|unequal_to"
//                "operator2": "regex|begins_with|contains|does_not_be-
gin_with|does_not_contain|does_not_end_with|ends_with|equal_to|is_con-
tained_in|is_not_contained_in|unequal_to"
```

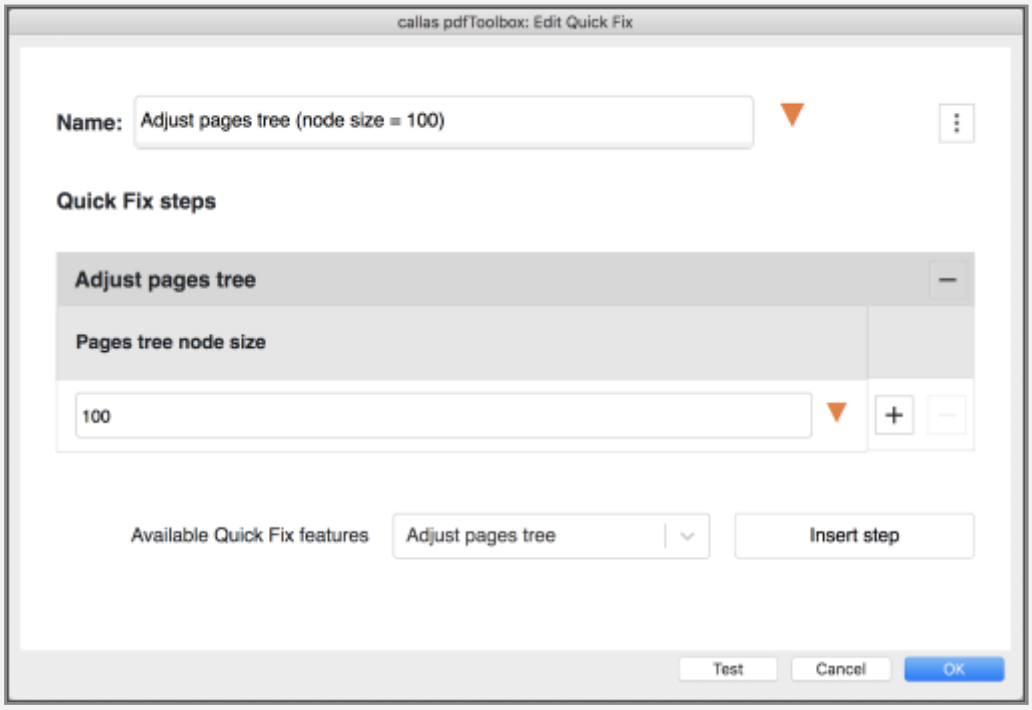
Click to copy

Page geometry (and Pages tree) Quick Fixes

Adjust pages tree

This Quick Fixes will hardly ever be of relevant use of its own. It is always executed implicitly if any of the QuickFixes dealing with page geometry are executed.

Due to the flexibility of the PDF syntax, there are numerous ways to put pages and their accompanying page geometry boxes (at least a MediaBox must always be specified) inside a PDF. The "Adjust document Pages Tree" Quick Fix "normalizes" page organisation in PDFs. For multi-page PDFs it makes sure, no more than the number of pages specified in "Page tree node size" are present in a page tree node (this only has practical implications for retrieving pages in PDFs with a large number of pages). In addition, and entries for page geometry boxes – which could be present in intermediate page tree nodes and would then apply to all pages under that node – are moved to the page to which they apply, such that each page stands on its own feet regarding page geometry boxes (and makes them independent of all other pages). Again, this is relevant for fast processing of page related information.



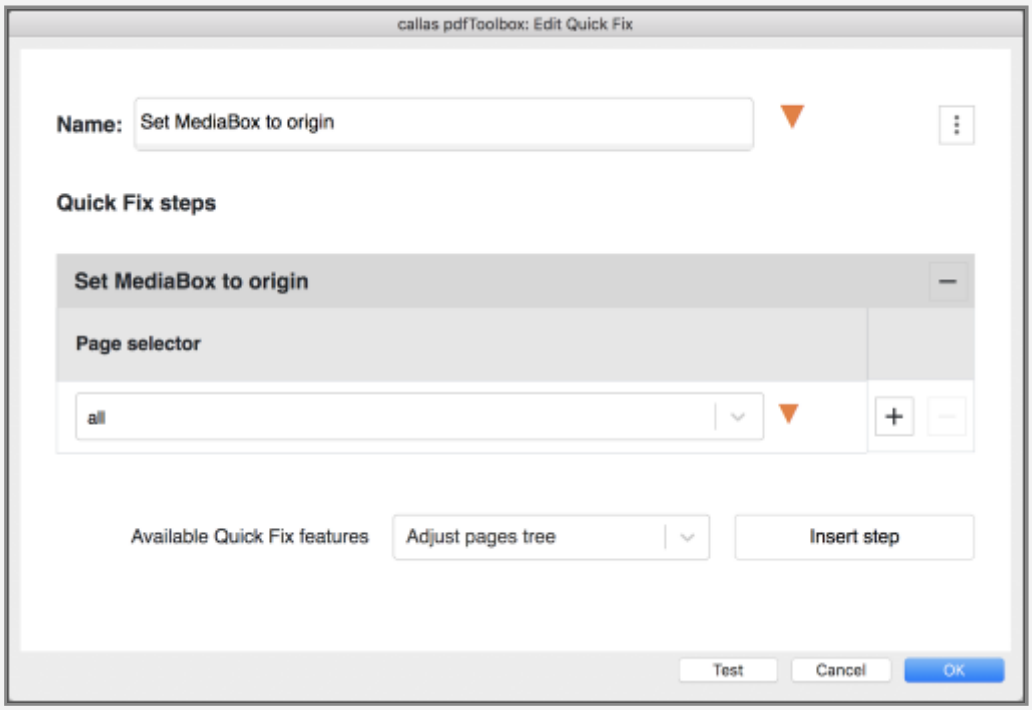

```
{
    "quickfixes": [
        {
            "quickfix": "adjust_pages_tree",
            "version": "1.0",
            "instructions": [
                {
                    "pages_size": 100
                }
             ]
        }
    ]
}
```

Click to copy

Set MediaBox to origin

This Quick Fix does a very specific thing: it makes sure, that the lower left of the MediaBox sits at 0:0. Page geometry manipulations can easily lead to situations where this is not the case. Strictly speaking this does not matter anyway – as long as width and height of page geometry boxes – and their rela-

tive position towards each other – are correct, everything else is irrelevant. The lower left of the MediaBox could be at 2000:-900 and there would not be a problem. Except with some output or other PDF processing systems that got used to the fact that especially in the earlier days of PDF all MediaBoxes would sit at 0:0. These systems struggle with PDFs whose MediaBox does not sit at 0:0 and might create 'unexpected' output.



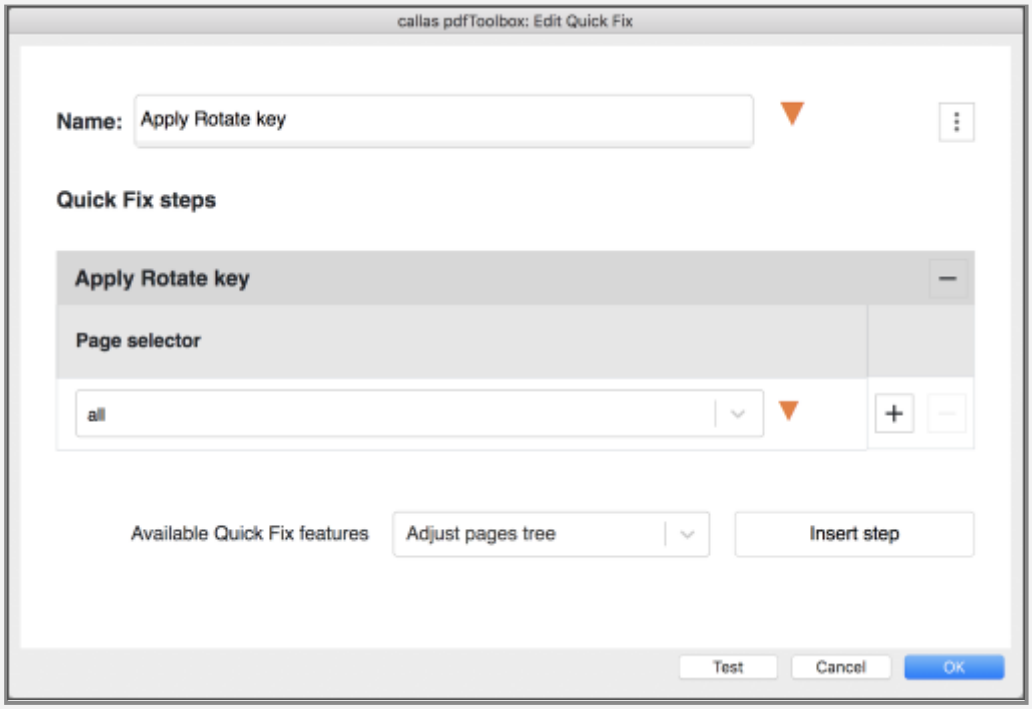

```
{
    "quickfixes": [
        {
            "quickfix": "set_mediabox_to_origin",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "all"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

Click to copy

Apply Rotate key

This will have an effect only if a given page has a Rotate entry, and if that Rotate entry is not equal zero. The fact of the Rotate key is then incorporated into the page description, and the Rotate key is removed. In addition, the page geometry boxes and the positions of annotations and form fields are adjusted accordingly.




```
{
    "quickfixes": [
        {
            "quickfix": "apply_rotate_key",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "all"
                }
            ]
        }
    ]
}
```

```
//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

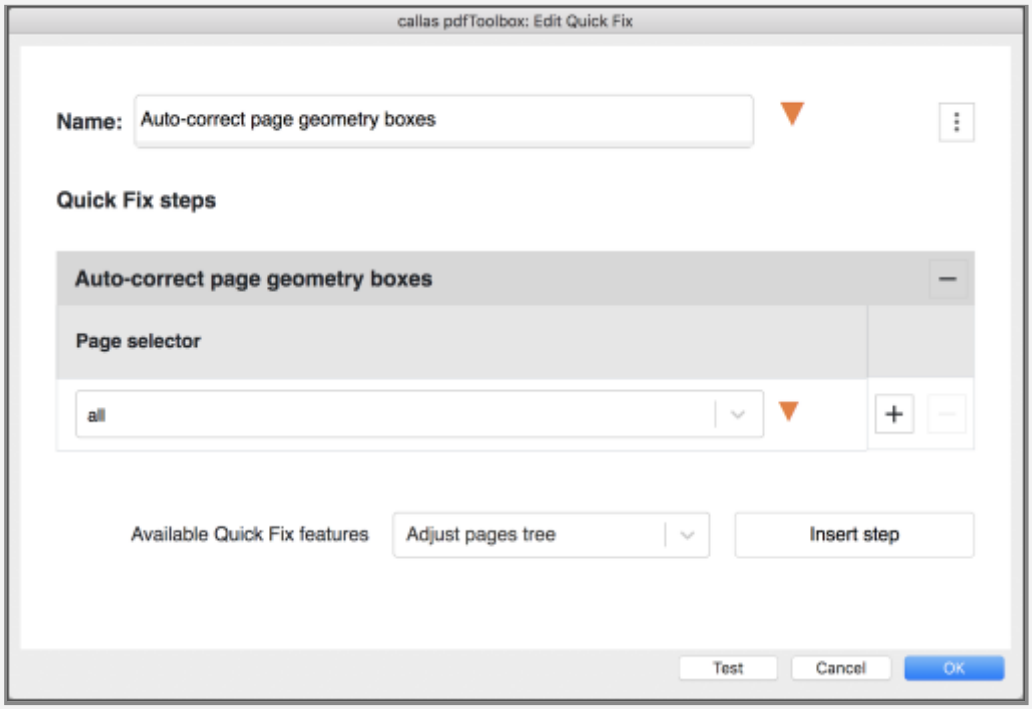
Click to copy

Auto-correct page geometry boxes

Works in the same way as the respective fixup, except that in the Quick Fix it is possible to define the pages whose page geometry boxes are to be auto-corrected:

Automatically corrects nesting of page geometry boxes (TrimBox or ArtBox, BleedBox, CropBox, MediaBox – in that order), if necessary, on all pages defined by the Page selector.

In addition, for these pages the lower left of the MediaBox is set to the origin.




```
{
    "quickfixes": [
        {
            "quickfix": "auto_correct_page_boxes",
            "version": "1.0",
            "instructions": [
                {
```

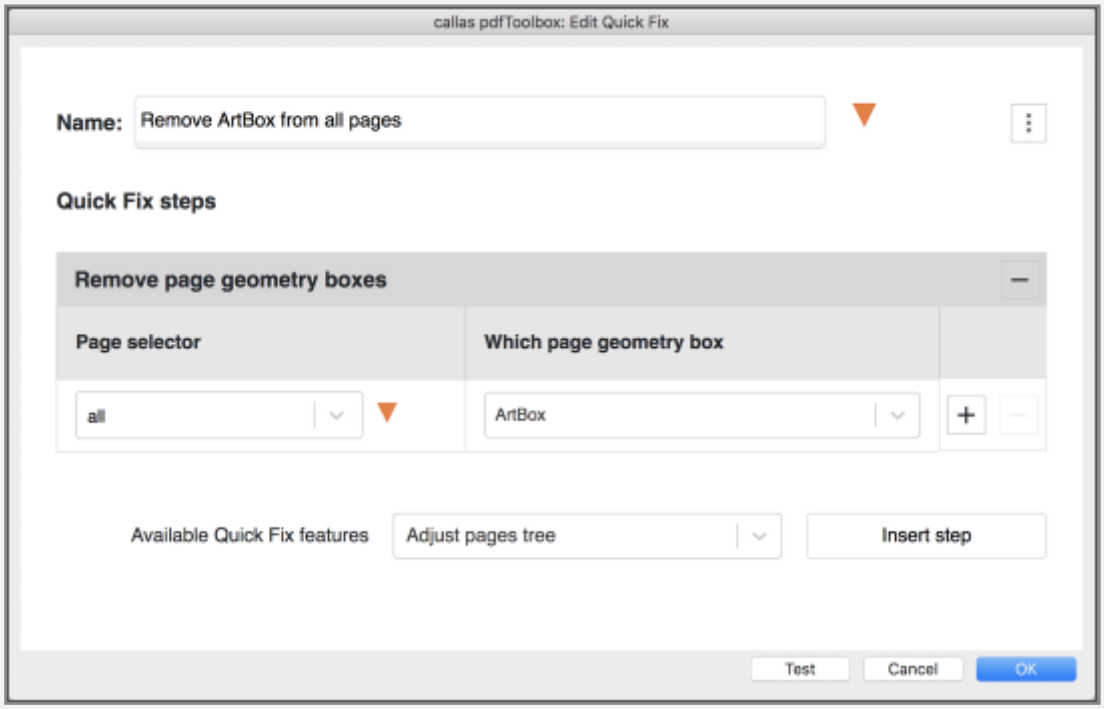
```
                    "page_selector": "all"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

Click to copy

Remove page geometry boxes

Removes the selected page geometry box(es) from the pages defined by the Page selector.




```
{
    "quickfixes": [
        {
            "quickfix": "remove_page_box",
            "version": "1.0",
            "instructions": [
                {
```

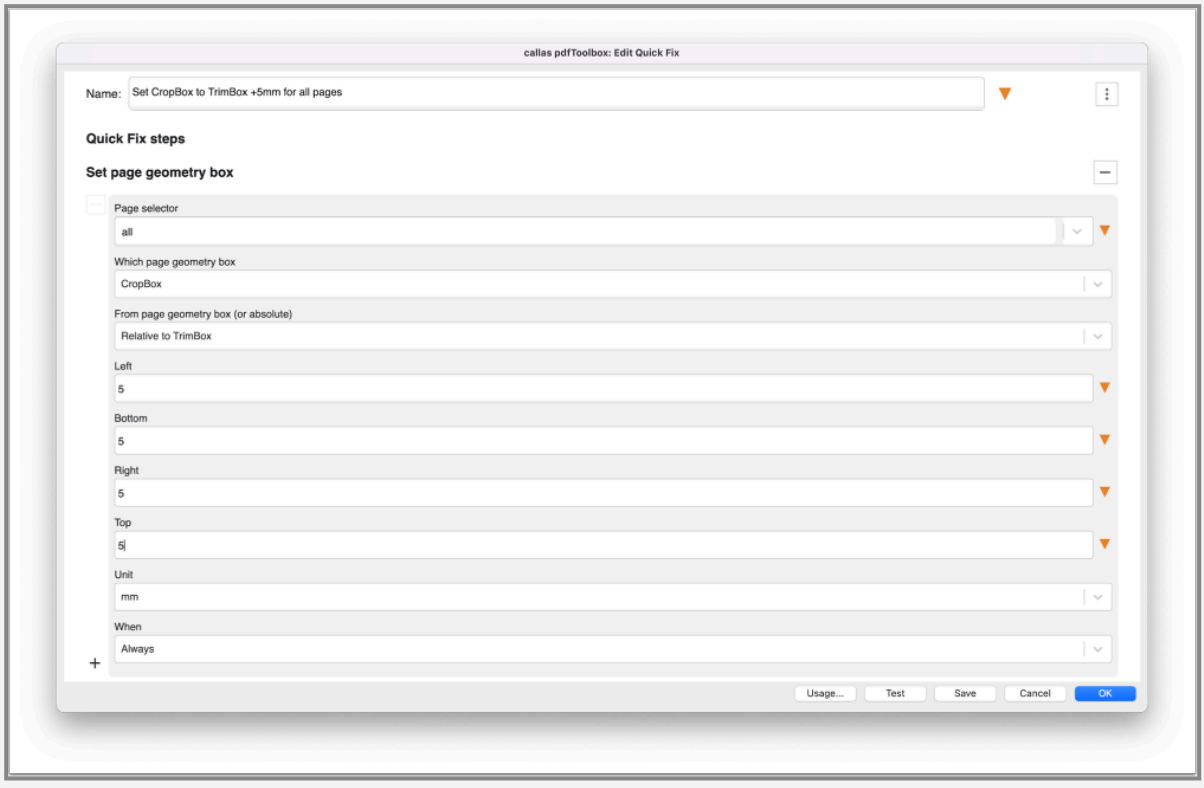
```
                    "page_selector": "all",
                    "which_box": "ArtBox"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                "which_box": "CropBox|BleedBox|TrimBox|ArtBox"
```

Click to copy

Set page geometry box

Sets page geometry boxes either relative to an existing page geometry box or to absolute coordinates.




```
{
    "quickfixes": [
        {
```

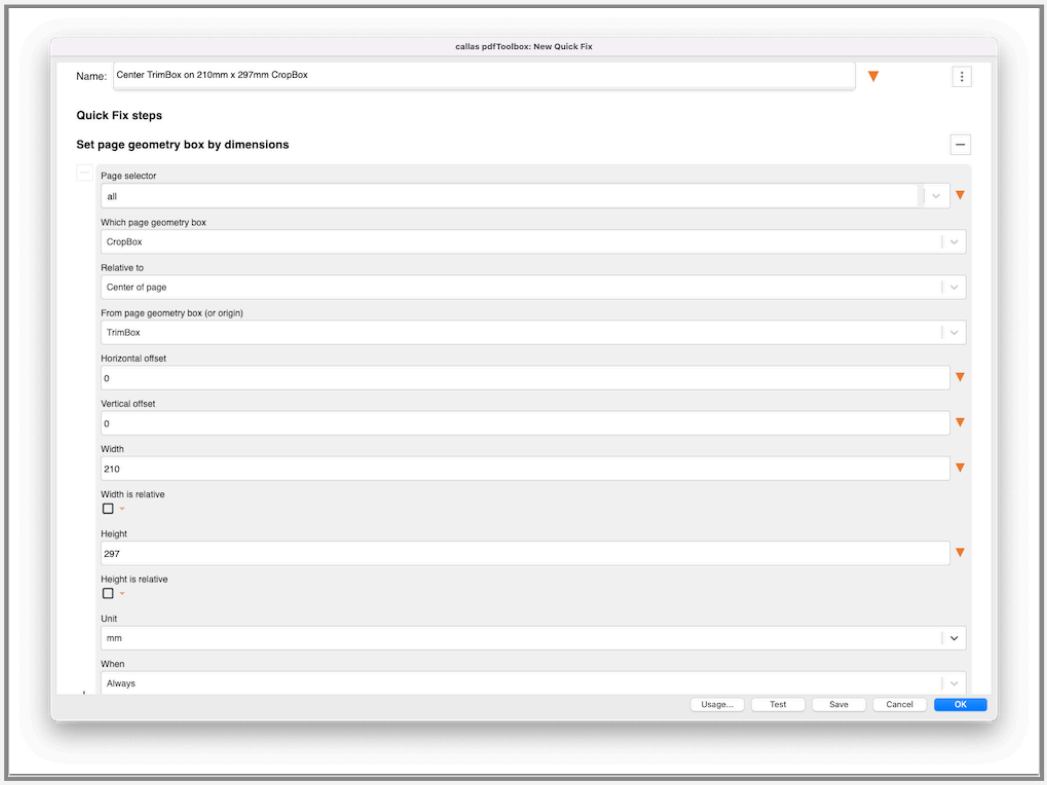
```
            "quickfix": "set_page_box",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "all",
                    "which_box": "CropBox",
                    "from_box_or_absolute":  "BleedBox",
                    "left": 5.0,
                    "bottom": 5.0,
                    "right": 5.0,
                    "top": 5.0,
                    "unit": "mm",
                    "when": "always"
                }
             ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                "which_box": "CropBox|BleedBox|TrimBox|ArtBox"
//                "from_box_or_absolute": "CropBox|BleedBox|TrimBox|ArtBox|ab-
solute"
```

Click to copy

Set page geometry box (by dimensions)

Sets page geometry boxes by specifying page box dimensions.




```
{
    "quickfixes": [
        {
            "quickfix": "set_page_box_by_dimensions",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "all",
                    "which_box": "CropBox",
                    "relative_to": "center",
                    "from_box_or_origin":  "TrimBox",
                    "hor_offset": 0.0,
                    "vert_offset": 0.0,
                    "width": 210.0,
                    "width_is_relative": false,
                    "height": 297.0,
                    "height_is_relative": false,
                    "unit": "mm",
                    "when": "always"
                }
             ]
        }
```

```
    ]
}

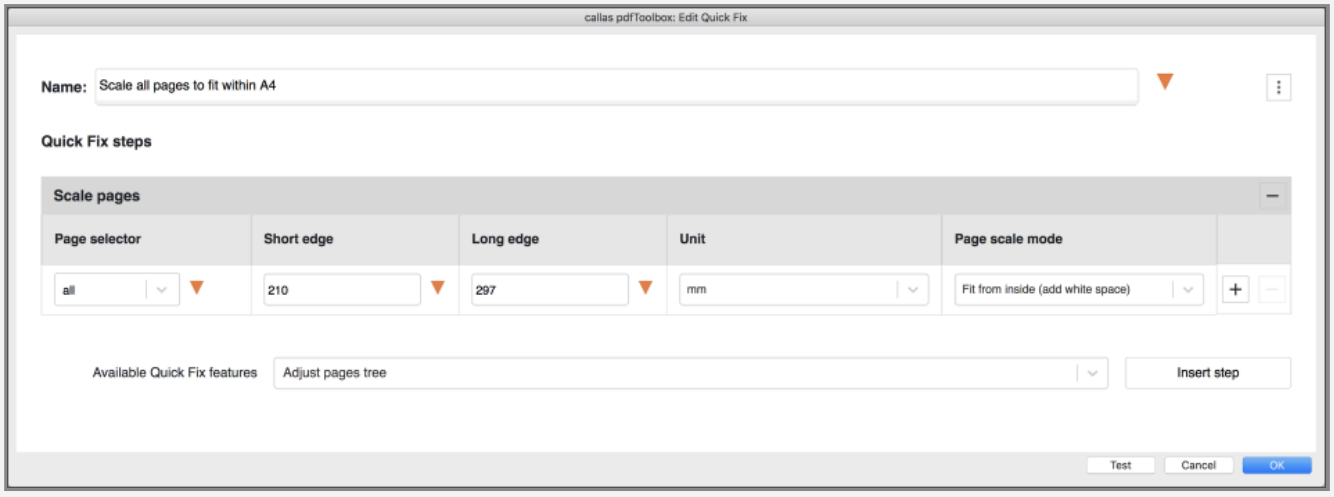
//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                "which_box": "CropBox|BleedBox|TrimBox|ArtBox"
//                "relative_to": "lower_left_corner|left_center|upper_left_cor-
ner|top_center|lower_right_corner|right_center|upper_right_corner|upper_right_cor-
ner|bottom_center|center"
//                "from_box_or_origin": "CropBox|BleedBox|TrimBox|ArtBox|origin"
```

Click to copy

Scale/rotate/flip pages Quick Fixes

Scale pages

Scales all pages in the PDF defined by the Page selector. In addition, for these pages the lower left of the MediaBox is set to the origin (0:0).




```
{
    "quickfixes": [
        {
            "quickfix": "scale_pages",
            "version": "1.0",
            "instructions": [
                {
```

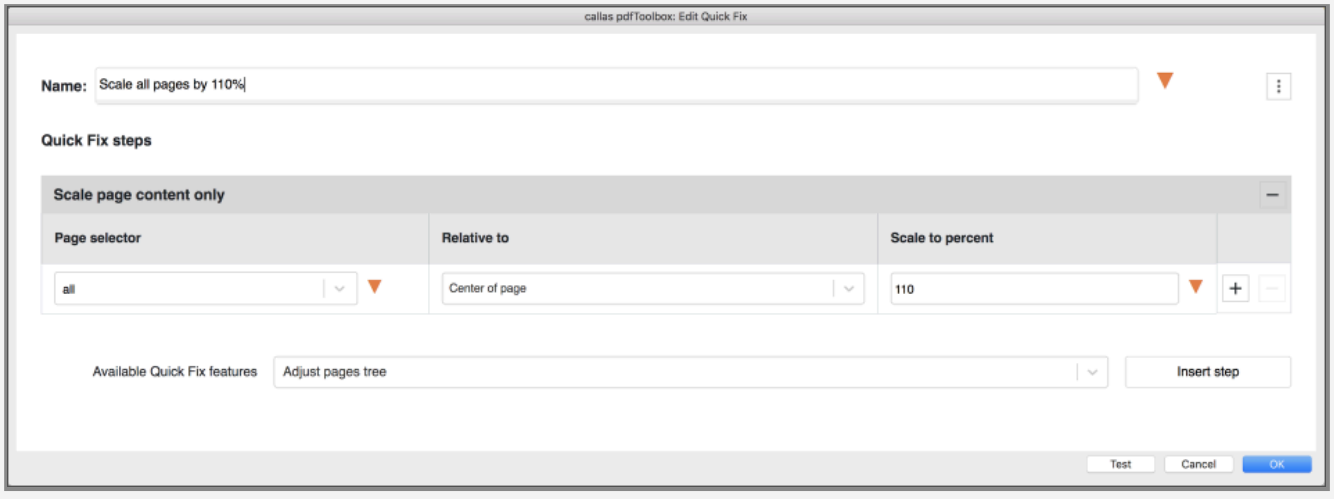
```
                    "page_selector": "all",
                    "short_edge": 210,
                    "long_edge": 297,
                    "unit": "mm",
                    "page_scale_mode": "fit_from_inside_add_white_space"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                "page_scale_mode": "fit_from_inside_add_white_space|fit_from_in-
side_scale_page_edge_proportionally|fit_from_outside_cut_page|fit_from_out-
side_scale_page_edge_proportionally|stretch_to_fill"
```

Click to copy

Scale page content only

Scales the page content to a given percentage without changing page dimensions for all pages defined by th Page selector. In addition, for these pages the lower left of the MediaBox is set to the origin.




```
{
    "quickfixes": [
        {
```

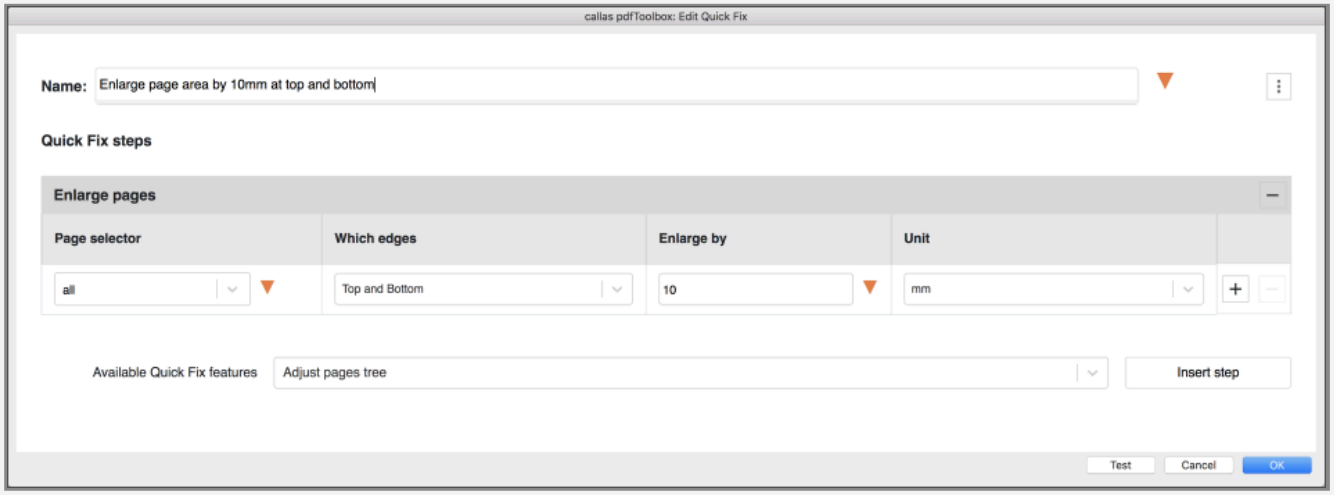
```
            "quickfix": "scale_page_content_only",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "all",
                    "relative_to": "center",
                    "scale_to_percent": 110
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                "relative_to": "lower_left_corner|left_center|upper_left_cor-
ner|top_center|lower_right_corner|right_center|upper_right_corner|upper_right_cor-
ner|bottom_center|center"
```

Click to copy

Enlarge page

Enlarges the page area without modifying the page content for all pages defined by the Page selector. In addition, for these pages the lower left of the MediaBox is set to the origin.




```
{
```

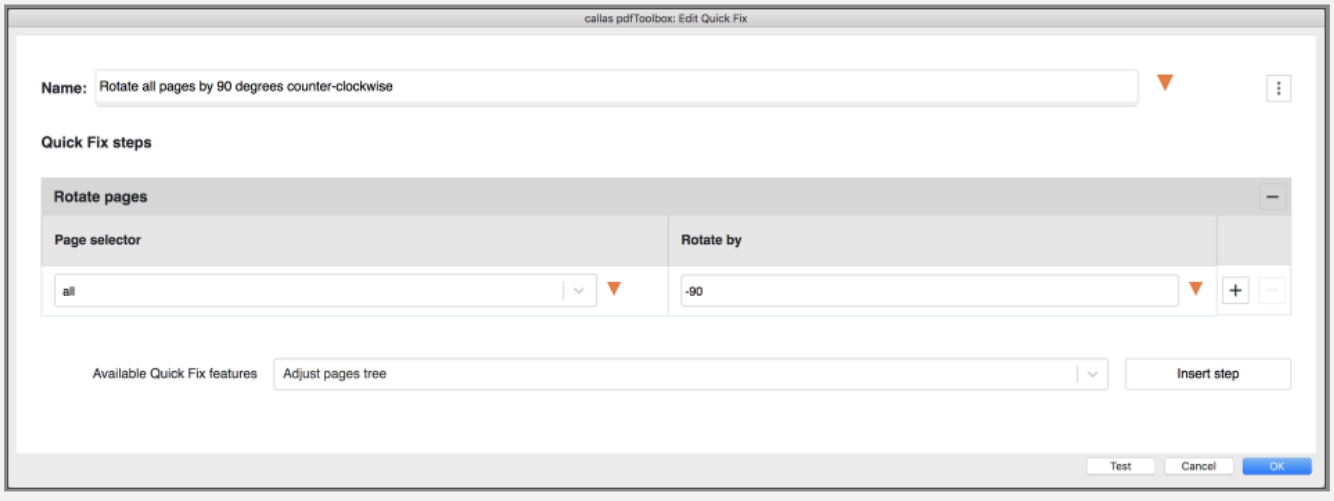
```
    "quickfixes": [
        {
            "quickfix": "enlarge_pages",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "all",
                    "which_edges": "top_and_bottom",
                    "enlarge_by": 10,
                    "unit": "mm"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                "which_edges": "left|bottom|right|top|left_and_right|top_and_bot-
tom|all"
```

Click to copy

Rotate pages

Rotates pages defined by the Page selector by the defined angle. If there is a page rotation parameter it is applied. In addition, for these pages the lower left of the MediaBox is set to the origin.



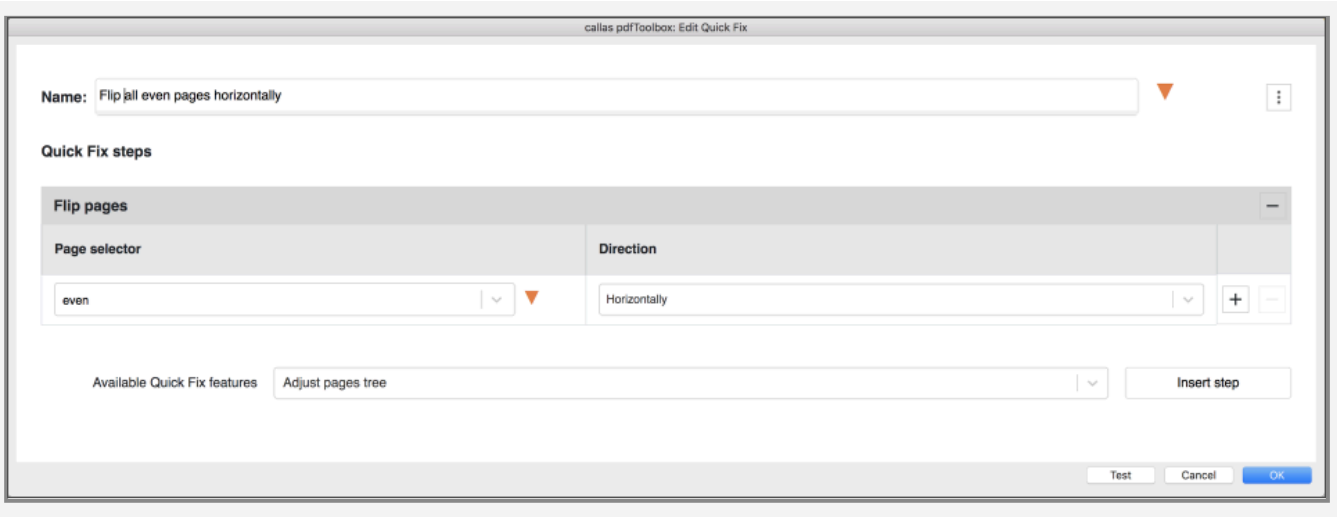
```
{
    "quickfixes": [
        {
            "quickfix": "rotate_pages",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "all",
                    "rotate_by": -90
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

Click to copy

Flip pages

Flips the pages horizontally or vertically for all pages defined by the Page selector. In addition, for these pages the lower left of the MediaBox is set to the origin.




```
let cfg =
{
    "quickfixes": [
```

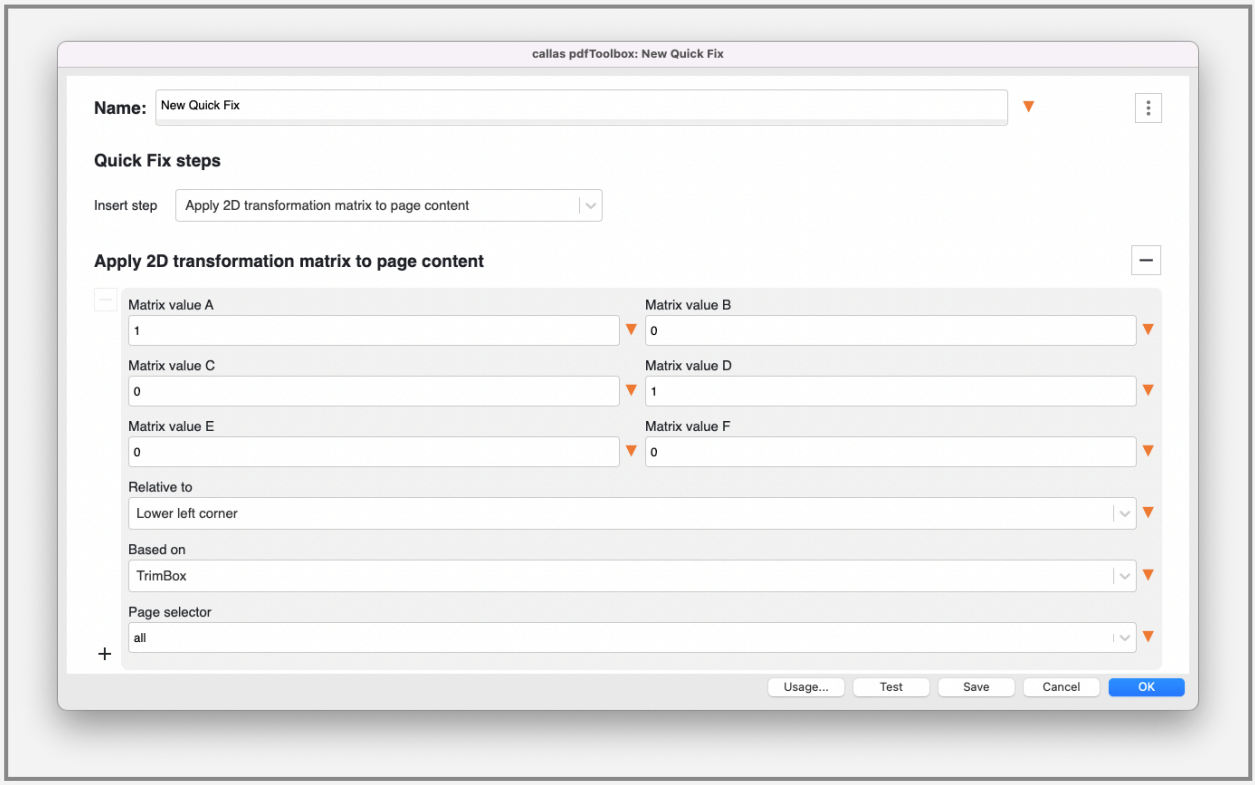
```
        {
            "quickfix": "flip_pages",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "even",
                    "flip_direction": "horizontally"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>",
//                "flip_direction": "horizontally|vertically"
```

Click to copy

Apply 2D transformation matrix to page content

Applies a generic 6-element transformation matrix on the content of a page. It only transforms the page content; it does not modify any of the page geometry boxes in the PDF document. More information about this QuickFix can be found here.




```
{
	"quickfixes" :
	[
		{
			"instructions" :
			[
				{
					"based_on" : "TrimBox",
					"m1" : 1.0,
					"m2" : 0.0,
					"m3" : 0.0,
					"m4" : 1.0,
					"m5" : 0.0,
					"m6" : 0.0,
					"page_selector" : "all",
					"relative_to" : "lower_left_corner"
				}
			],
			"quickfix" : "apply_transformation_matrix",
			"version" : "1.0"
		}
	]
```

```
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                “based_on: "CropBox|MediaBox|BleedBox|TrimBox|ArtBox"
//                "relative_to": "lower_left_corner|left_center|upper_left_cor-
ner|top_center|lower_right_corner|right_center|upper_right_corner|upper_right_cor-
ner|bottom_center|center"
```

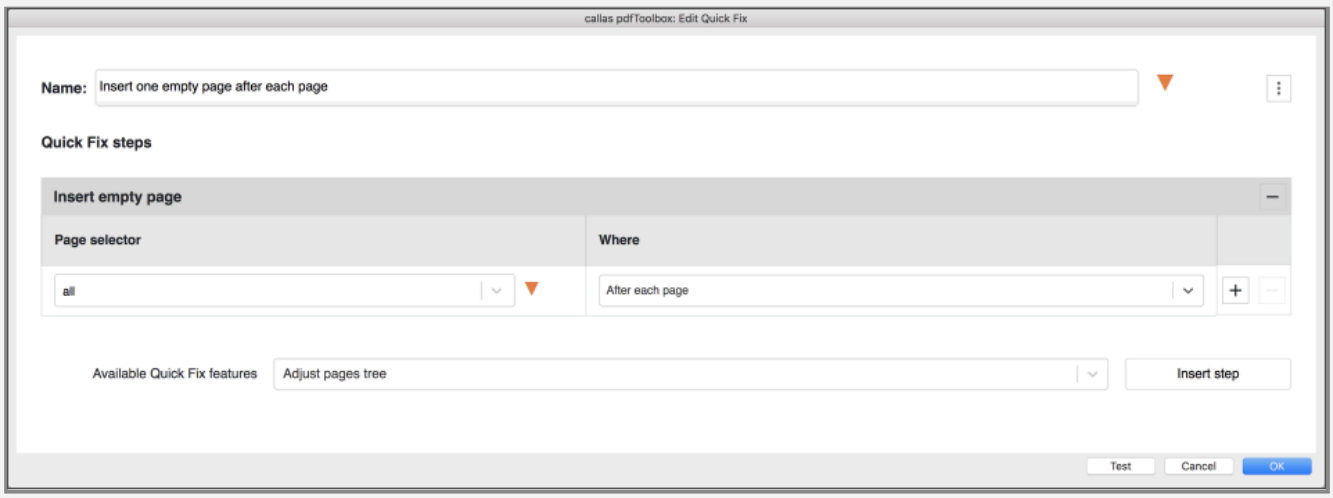
Click to copy

Insert/Duplicate/Remove/Invert/Reorder pages Quick Fixes

Insert empty page

Works in the same way as the Fixup "Insert page":

Inserts an empty page before or after the pages defined by the Page selector. The page geometry boxes will be a copy of the page before or after where the empty page is inserted.




```
{
    "quickfixes": [
        {
            "quickfix": "insert_empty_page",
            "version": "1.0",
            "instructions": [
```

```
                {
                    "page_selector": "all",
                    "where": "after"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>",
//                "where": "before|after"
```

Click to copy

Duplicate pages

Duplicates the page(s) defined by the Page selector.




```
{
    "quickfixes": [
        {
            "quickfix": "duplicate_page",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "1,-1"
                }
            ]
```

```
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

Click to copy

Remove pages

Removes the page(s) defined by the Page selector.




```
{
    "quickfixes": [
        {
            "quickfix": "remove_page",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "2--1"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

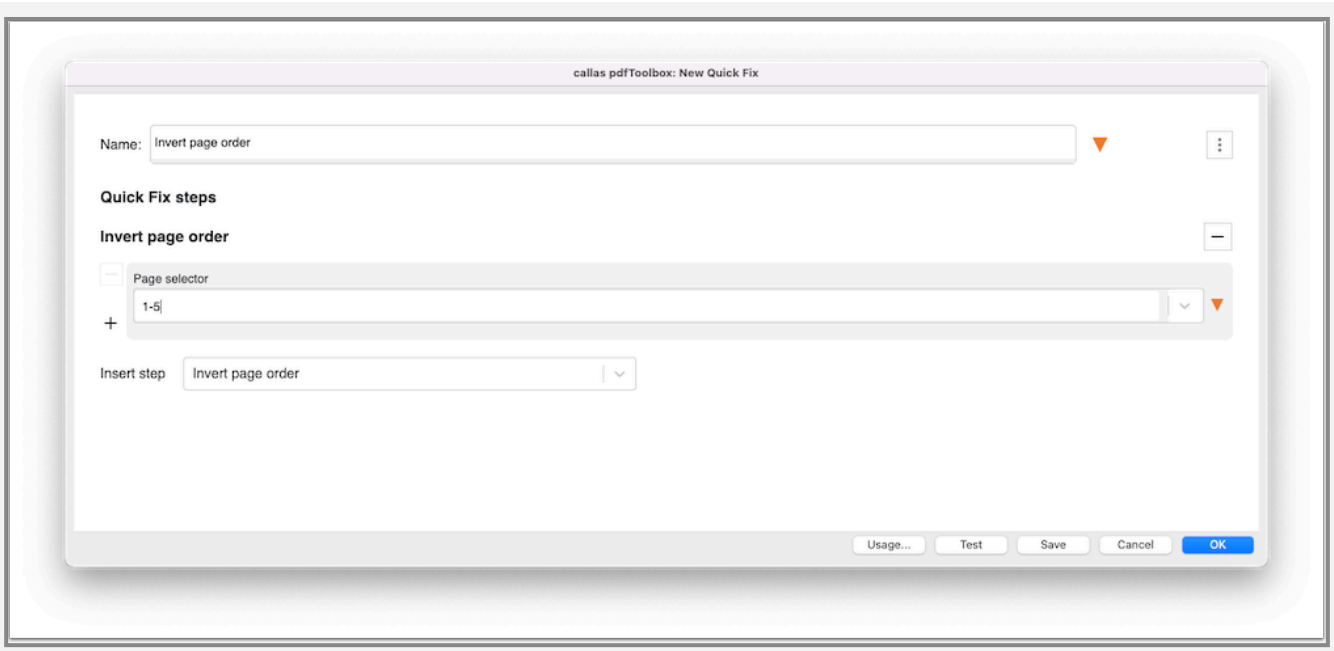
Click to copy

Invert page order

This Quick Fix inverts the page order in accordance with the defined 'page selector', like all pages or only even pages etc.

Example: A simple 10 page PDF with a page selector “1-5” will thereafter have:5,4,3,2,1,6,7,8,9,10.

More infos here.




```
{
    "quickfixes": [
        {
            "quickfix": "invert_pages",
            "version": "1.0",
            "instructions": [
                {
                    "page_selector": "1-5"
                }
            ]
        }
    ]
}

//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

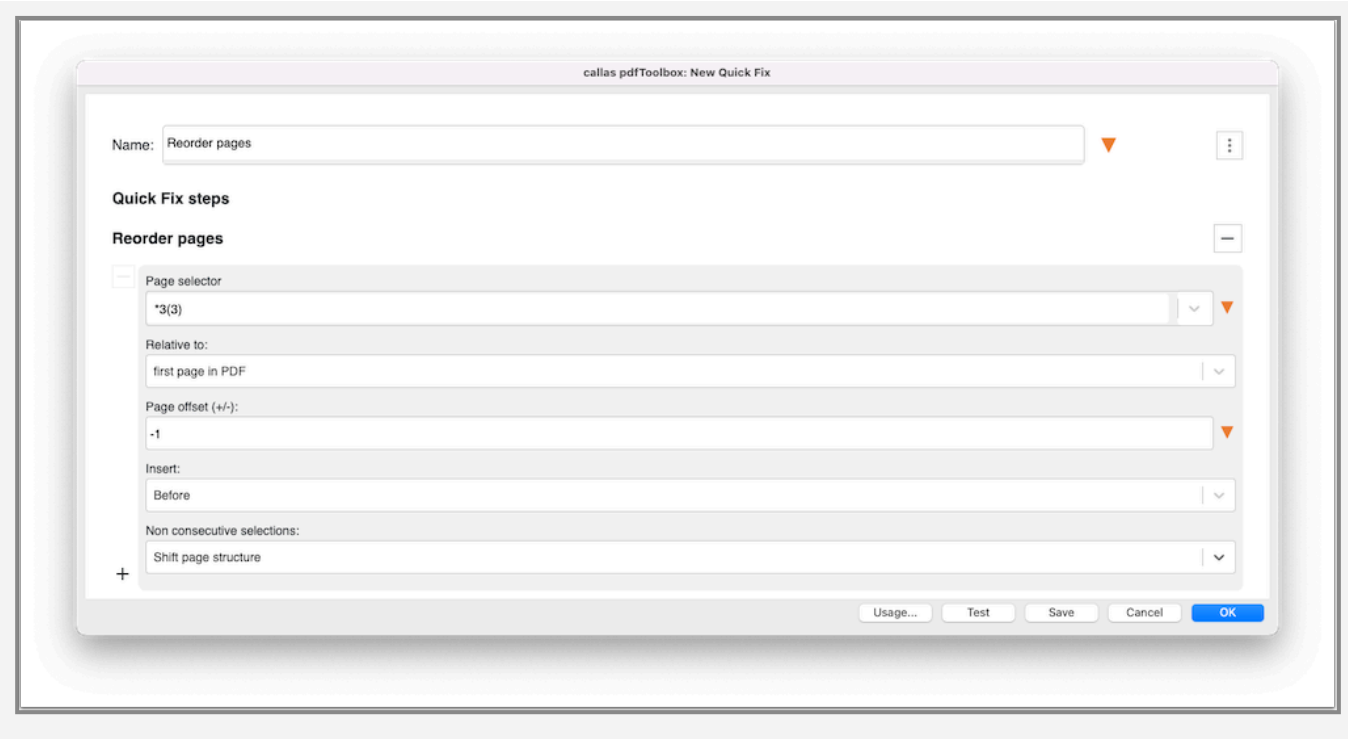
Click to copy

Reorder pages

This Quick Fix lets you reorder pages in a document based on the given attributes:

- Page selector, which can be a list eg, 1,2,3 or a simple expression etc
- Relative to (first/last page in PDF/selection)
- Page offset, which is the anchor point by which the selected pages are to be reordered
- Insert (before or after)
- Non consecutive selections, either
  - Insert as consecutive pages or
  - Shift page structure

More info here.




```
{
        "quickfixes" :
        [
                {
                        "instructions" :
                        [
```

```
                                    {
                                            "insert_mode" : "shift_pages",
                                            "insert_pages" : "insert_before",
                                            "page_offset" : -1,
                                            "page_selector" : "*3(3)",
                                            "relative_to_page" : "first_page"
                                    }
                            ],
                            "quickfix" : "reorder_pages",
                            "version" : "1.0"
                    }
            ]
}

//                "insert_mode": "insert_consecutive|shift_pages"
//                "insert_pages": "insert_before|insert_after"
//                "page_selector": "all|even|odd|<splitscheme_expression>",
//                "relative_to_page":
"first_page|first_sel_page|last_sel_page|last_page"
```

Click to copy

Layers (including Processing Steps metadata) Quick Fixes

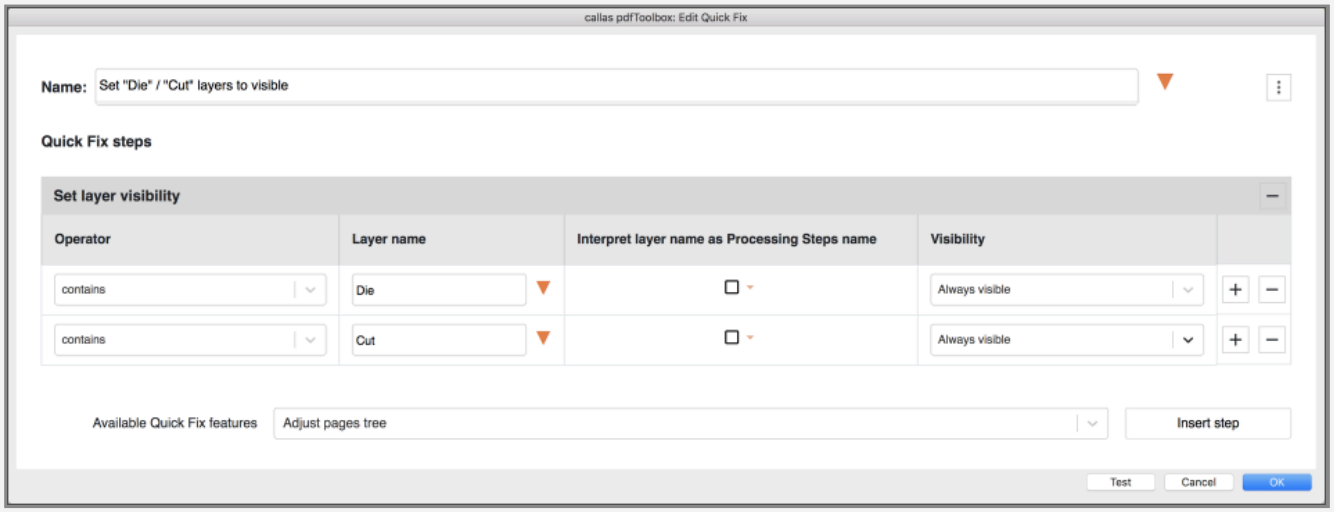
Set layer visibility

This Quick Fix is similar to the "Set layer default to ON/OFF" Fixup.

It sets the default visibility of a layer to either visible or not visible.

If "Interpret layer name as Processing Steps name" is checked, the names of layers are ignored, and instead their Processing Steps metadata is used to determine whether to turn their visibility on or off. Processing Steps metadata are to be written by combining group and type:

- the syntax for writing the Processing Steps metadata value in the Quick Fix step configuration is "<Processing Steps Group>:<Processing Steps Type>"
- for example: "Structural:Bleed"




```
{
    "quickfixes": [
        {
            "quickfix": "set_ocg_visibility",
            "version": "1.0",
            "instructions": [
                {
                    "operator": "contains",
                    "ocg_name": "Die",
                    "interpret_ocg_name_as_processsing_steps_name: false,
                    "visibility" : "on"
                },
                {
                    "operator": "contains",
                    "ocg_name": "Cut",
                    "interpret_ocg_name_as_processsing_steps_name: false,
                    "visibility" : "on"
                }
            ]
        }
    ]
}

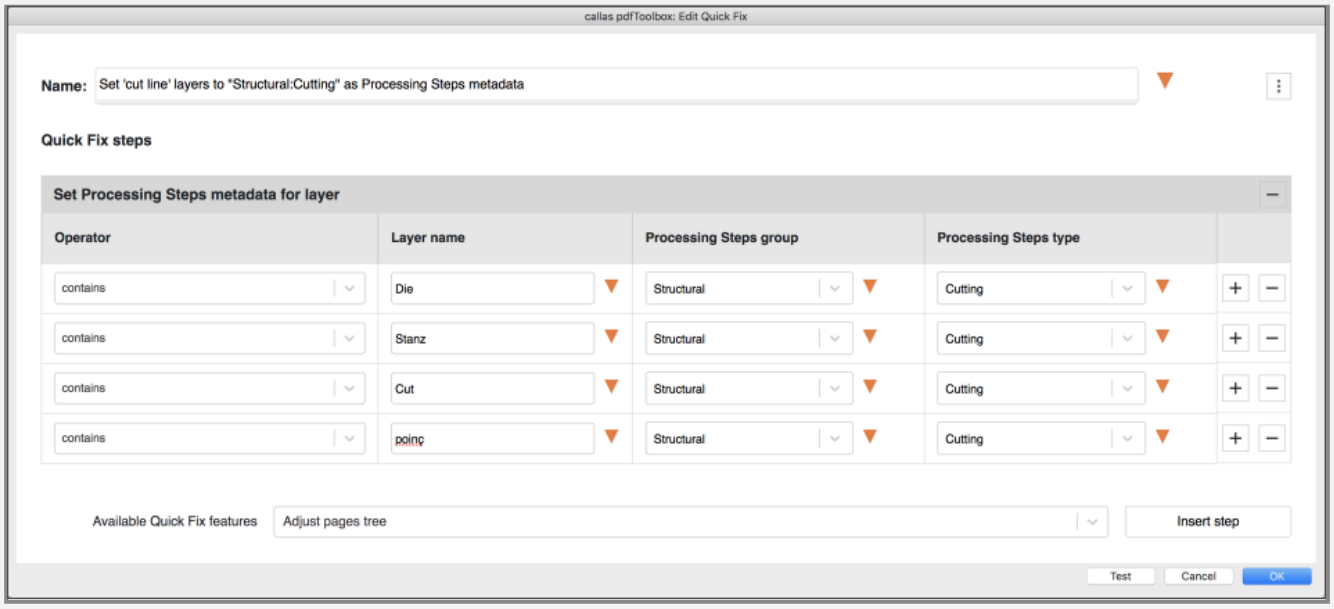
//                "operator": "regex|begins_with|contains|does_not_be-
gin_with|does_not_contain|does_not_end_with|ends_with|equal_to|is_con-
tained_in|is_not_contained_in|unequal_to"
//                "interpret_ocg_name_as_processsing_steps_name": true|false <op-
tional: default false>
```

```
//                "visibility": "on|off"
```

Click to copy

Set Processing Steps metadata for layer

This Quick Fix adds Processing Steps metadata to a layer / to layers. Already existing Processing Steps metadata entries will be overwritten.




```
{
        "quickfixes" :
        [
                {
                        "instructions" :
                        [
                                {
                                        "ocg_name" : "Die",
                                        "operator" : "contains",
                                        "processing_steps_group" : "Structural",
                                        "processing_steps_type" : "Cutting"
                                },
                                {
                                        "ocg_name" : "Cut",
                                        "operator" : "contains",
```

```
                                        "processing_steps_group" : "Structural",
                                        "processing_steps_type" : "Cutting"
                                },
                                {
                                        "ocg_name" : "Stanz",
                                        "operator" : "contains",
                                        "processing_steps_group" : "Structural",
                                        "processing_steps_type" : "Cutting"
                                },
                                {
                                        "ocg_name" : "poinc",
                                        "operator" : "contains",
                                        "processing_steps_group" : "Structural",
                                        "processing_steps_type" : "Cutting"
                                }
                        ],
                        "quickfix" : "set_processing_steps_metadata_for_ocg",
                        "version" : "1.0"
                }
        ]
}

//                "operator": "regex|begins_with|contains|does_not_be-
gin_with|does_not_contain|does_not_end_with|ends_with|equal_to|is_con-
tained_in|is_not_contained_in|unequal_to"
//                "processing_steps_group": "<custom>|Structural|Dimen-
sions|Braille|Legend|Positions|White|Varnish"
//                "processing_steps_type": "<custom>|Cutting|PartialCutting|Re-
versePartialCutting|Creasing|ReverseCreasing|CuttingCreasing|ReverseCuttingCreas-
ing|PartialCuttingCreasing|ReversePartialCuttingCreasing|Drilling|Gluing|FoilStamp-
ing|ColdFoilStamping|Embossing|Debossing|Perforating|Bleed|VarnishFree|Ink-
Free|InkVarnishFree|Folding|Punching|Stapling|Hologram|Barcode|ContentArea|Coding-
Marking|Imprinting
```

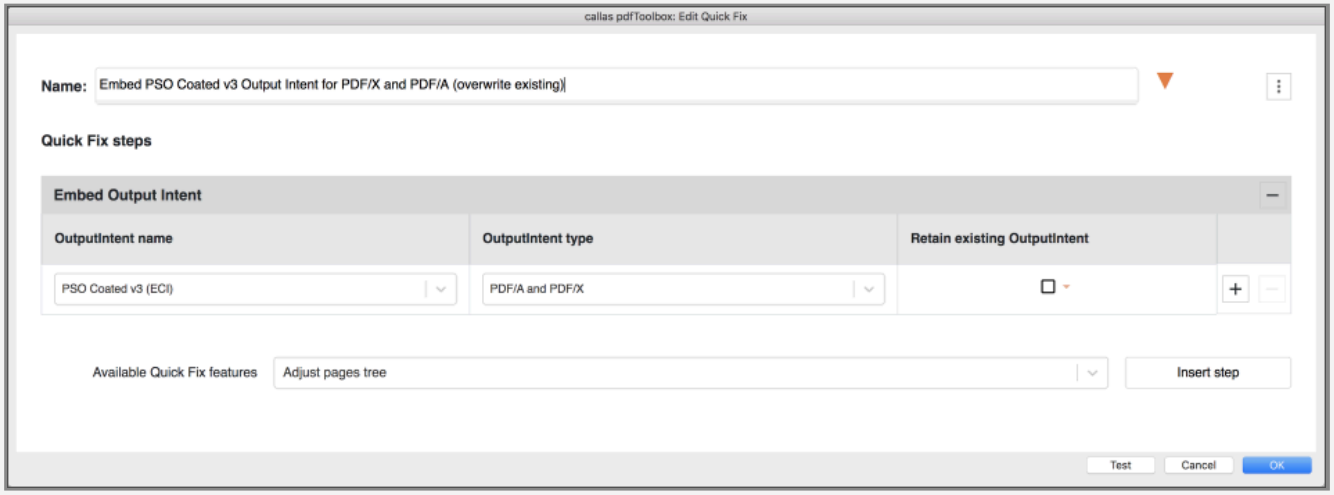
Click to copy

Output Intents Quick Fixes

Embed Output Intent

This Quick Fix embeds the specified Output Intent. It is possible to choose between different PDF standards – such as

PDF/X, PDF/A etc. – for the output intent to be embedded (these standards mandate that the output intent be the same for all such standards to which a PDF claims conformance, so a single embedded output intent can serve more than one PDF standard in the same PDF file).



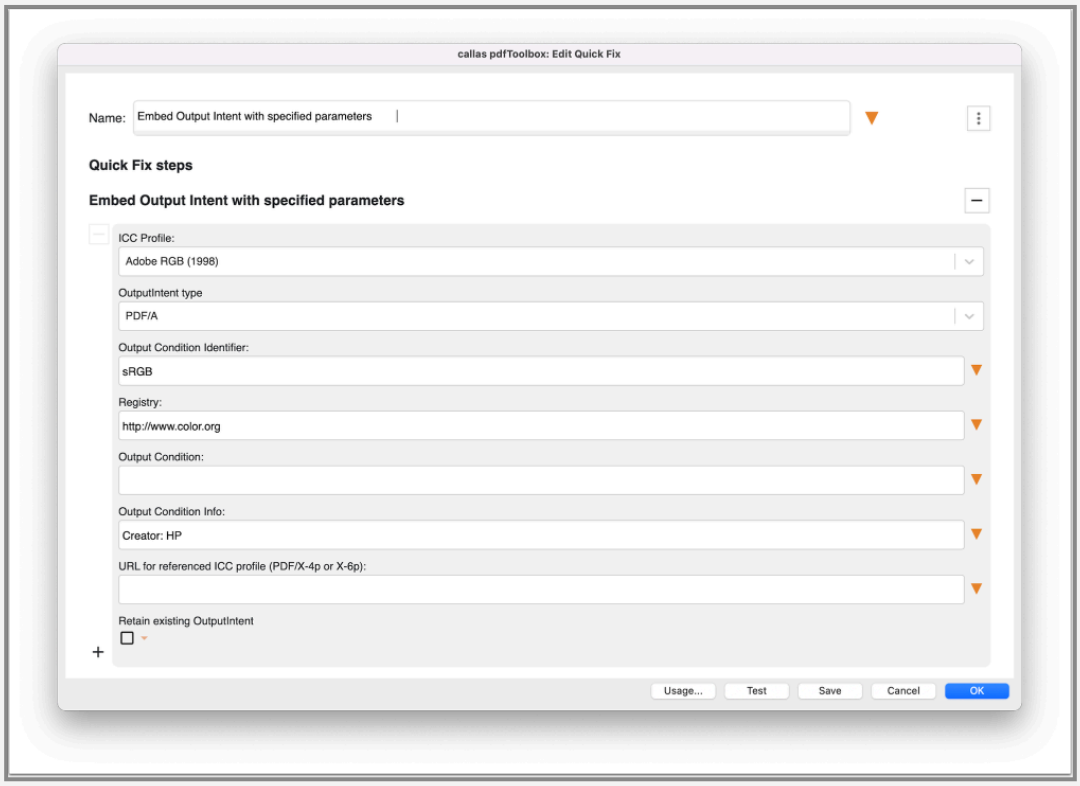

```
{
    "quickfixes": [
        {
            "quickfix": "embed_outputintent",
            "version": "1.0",
            "instructions": [
                {
                    "outputintent_name": "PSO Coated v3 (ECI)",
                    "outputintent_type" : "AX",
                    "retain_existing_outputintent" : false
                }
            ]
        }
    ]
}

//                "outputintent_type": "A|E|X|AE|AX|EX|AEX|X5n"
//                "retain_existing_outputintent": true|false
```

Click to copy

Embed Output Intent with specified parameters

This Quick Fix embeds a PDF/X, PDF/A and/or PDF/E Output Intent with all the parameters like the ICC color Profile (e.g. CoatedFOGRA27.icc) or an output condition identifier, which is simply a text description of the intended print specifications (e.g. FOGRA27).




```
{
        "quickfixes" :
        [
                {
                        "instructions" :
                        [
                                {
                                        "outputintent_condition" : "",
                                        "outputintent_condition_identifier" :
"sRGB",
                                        "outputintent_filepath_icc" : "/var/fold-
ers/lr/0vnqj68514j5w4t1pbd9jz840000gn/T/com.callassoftware.pdfToolboxDT/10490/
QuickFix-E0mtUs/Adobe RGB (1998).icc",
```

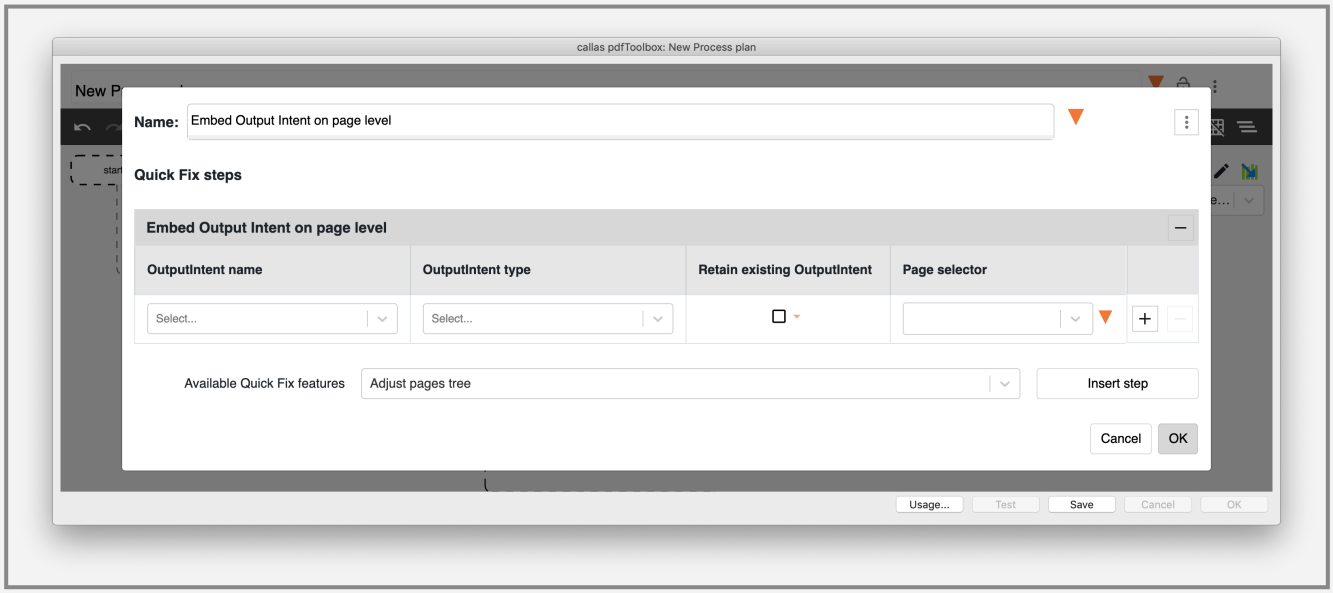
```
                                        "outputintent_info" : "Creator: HP",
                                        "outputintent_name_icc" : "Adobe RGB
(1998)",
                                        "outputintent_registry" : "http://www.col-
or.org",
                                        "outputintent_type" : "A",
                                        "outputintent_url_referenced_icc" : "",
                                        "retain_existing_outputintent" : false
                                }
                        ],
                        "quickfix" : "embed_outputintent_with_params",
                        "version" : "1.0"
                }
        ]
}


//                "outputintent_type": "A|E|X|AE|AX|EX|AEX|X5n"
//                "retain_existing_outputintent": true|false
```

Click to copy

Embed Output Intent on page level

This Quick Fix embeds the specified Output Intent on a page level (using page selector). It is possible to choose between different PDF standards – such as PDF/X, PDF/A etc. – for the Output Intent to be embedded.



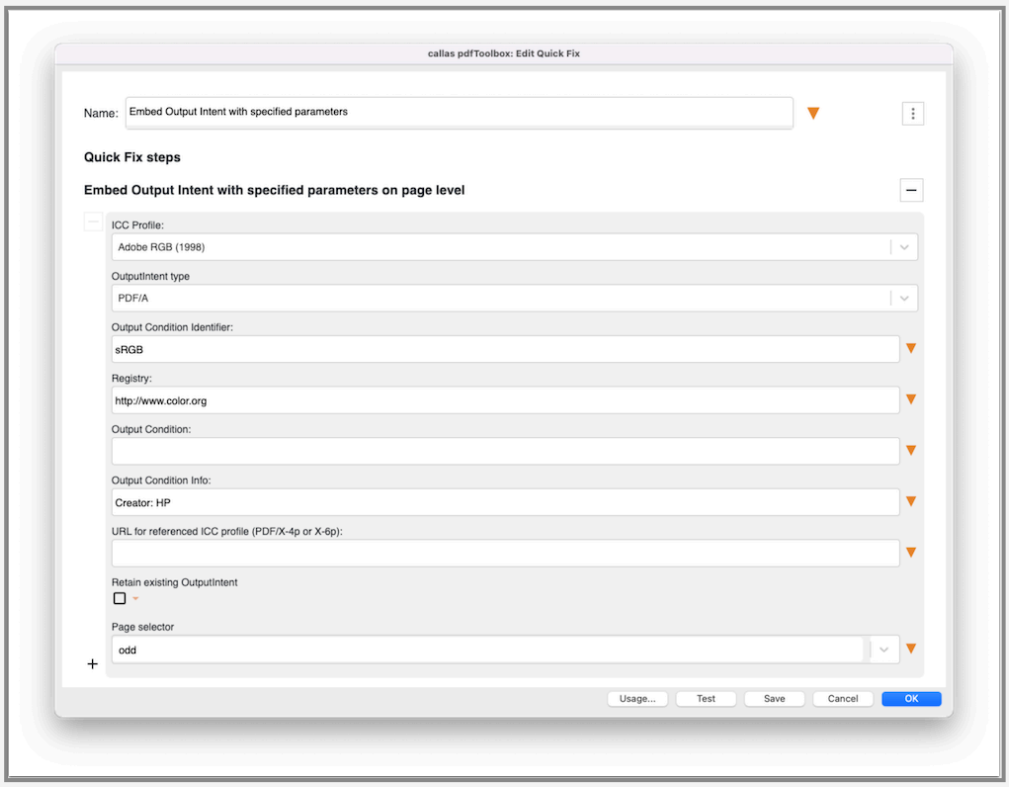

```
{
    "quickfixes": [
        {
            "quickfix": "embed_page_outputintent",
            "version": "1.0",
            "instructions": [
                {
                    "outputintent_name": "PSO Coated v3 (ECI)",
                    "outputintent_type" : "AX",
                    "retain_existing_outputintent" : false,
                    "page_selector": "odd"
                }
            ]
                        }
    ]
}

//                "outputintent_type": "A|E|X|AE|AX|EX|AEX|X5n"
//                "retain_existing_outputintent": true|false
//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

Click to copy

Embed Output Intent with specified parameters on page level

This Quick Fix embeds a PDF/X, PDF/A and/or PDF/E Output Intent with all the parameters like the ICC color Profile (e.g. CoatedFOGRA27.icc) and an output condition identifier, which is simply a text description of the intended print specifications (e.g. FOGRA27) on a page level (using page selector).




```
{
        "quickfixes" :
        [
                {
                        "instructions" :
                        [
                                {
                                        "outputintent_condition" : "",
                                        "outputintent_condition_identifier" :
"sRGB",
                                        "outputintent_filepath_icc" : "/var/fold-
ers/lr/0vnqj68514j5w4t1pbd9jz840000gn/T/com.callassoftware.pdfToolboxDT/10490/
```

```
QuickFix-gTXfRn/Adobe RGB (1998).icc",
                                        "outputintent_info" : "Creator: HP",
                                        "outputintent_name_icc" : "Adobe RGB
(1998)",
                                        "outputintent_registry" : "http://www.col-
or.org",
                                        "outputintent_type" : "A",
                                        "outputintent_url_referenced_icc" : "",
                                        "page_selector" : "odd",
                                        "retain_existing_outputintent" : false
                                }
                        ],
                        "quickfix" : "embed_page_outputintent_with_params",
                        "version" : "1.0"
                }
        ]
}


//                "outputintent_type": "A|E|X|AE|AX|EX|AEX|X5n"
//                "retain_existing_outputintent": true|false
//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

Click to copy

Remove all Output Intents

This QuickFix removes all output intents in a PDF file.

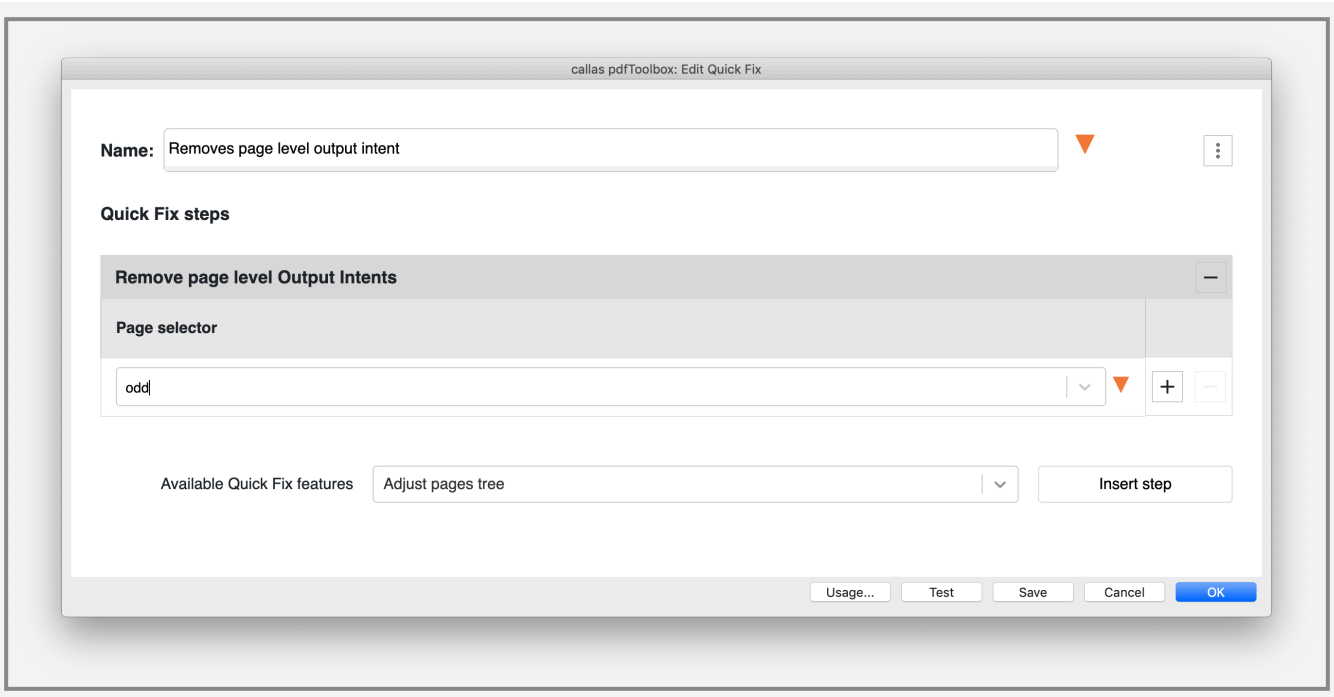


```
{
        "quickfixes" :
        [
                {
                        "instructions" : [ {} ],
                        "quickfix" : "remove_outputintents",
                        "version" : "1.0"
                }
        ]
}
```

Click to copy

Remove page level Output Intents

This Quick Fix removes Output Intents on a page level in a PDF file.




```
{
        "quickfixes" :
        [
                {
        "quickfix": "remove_page_outputintents",
        "version": "1.0",
```

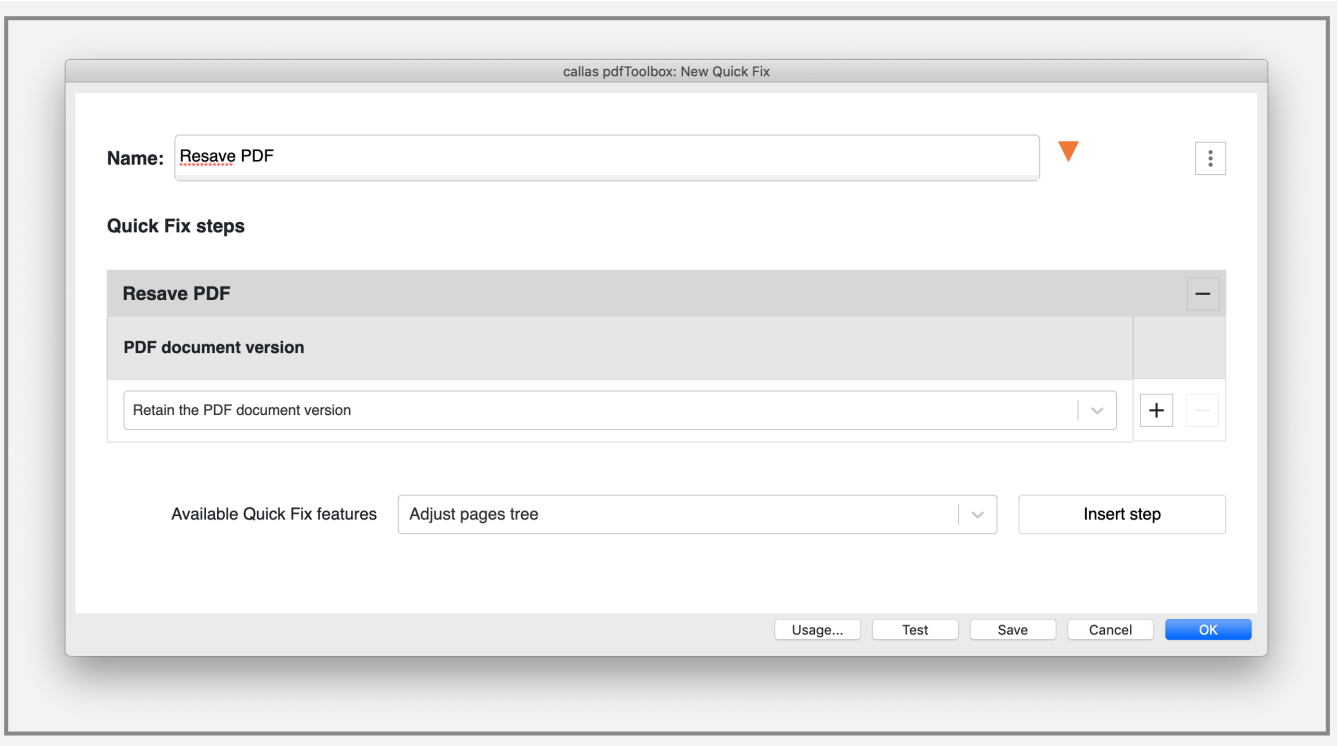
```
        "instructions":
        [
            {
                "page_selector": "odd"
            }
        ]
                }
  ]
}


//                "page_selector": "all|even|odd|<splitscheme_expression>"
```

Click to copy

Resave PDF Quick Fix

Resaves the PDF document by keeping the current version of the PDF document or saving up to another version.




```
{
        "quickfixes" :
        [
```

```
                {
        "quickfix": "fullsave",
        "version": "1.0",
        "instructions":
        [
            {
                "pdf_version": "retain"
            }
        ]
                }
  ]
}

//                "pdf_version": "retain|1.4|1.5|1.6|1.7|2.0"
```

Click to copy

VDP PDF/VT DPart Quick Fix

Inject DPart

This Quick Fix injects very simple DPart data into a PDF file. In essence, it creates logical groups in the form of page ranges. Using the Page selector syntax, a number of approaches exist to define such page ranges:

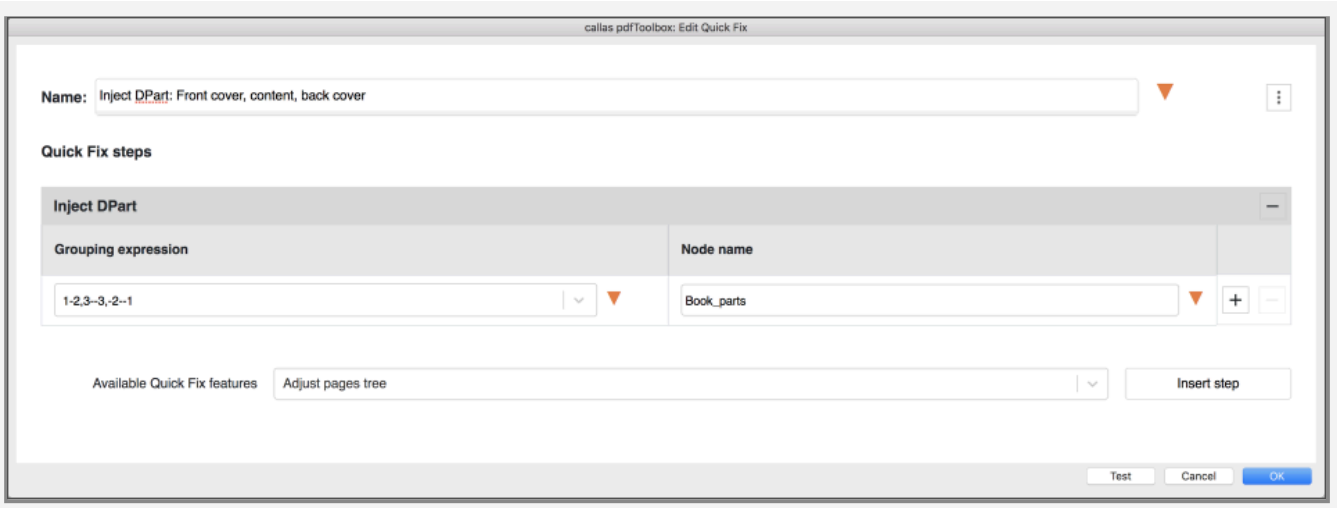
- explicitly, e.g. 1-4, 5-8, 9-12 – creates three ranges accordingly, only works in a predictable fashion for PDFs that contain exactly 12 pages; any exceeding pages will be put into their own page range; for PDFs with less than 12 pages, one or several page ranges will be 'incomplete' or missing altogether.
- rule based, e.g. 4* – creates a page range for each block of four pages
- all kinds of more complex expressions...

For details see the article Page selection.

The entry for Node name expressed the meaning of the grouped page ranges. This could be arbitrary parts of a book, account statements for a bank's customers, photo books, ...

Where DPPart injection is to be dynamically based upon a PDF's characteristics or content, execution of a Quick Check

or a regular check , in combination with some JavaScript, inside a Process Plan, and before execution of a QuickFix with DPart injection, makes it possible to address more advanced scenarios in a very powerful manner. An example is provided in the form of the Process Plan "Create DPart record information from headings" which ships with pdfToolbox 12.




```
{
        "quickfixes" : [
                {
                        "quickfix" : "inject_dpart",
                        "version" : "1.0",
                        "instructions" : [
                                {
                                        "grouping_expression" : "1-2,3--3,-2--1",
                                        "node_name" : "Book_parts"
                                }
                        ]

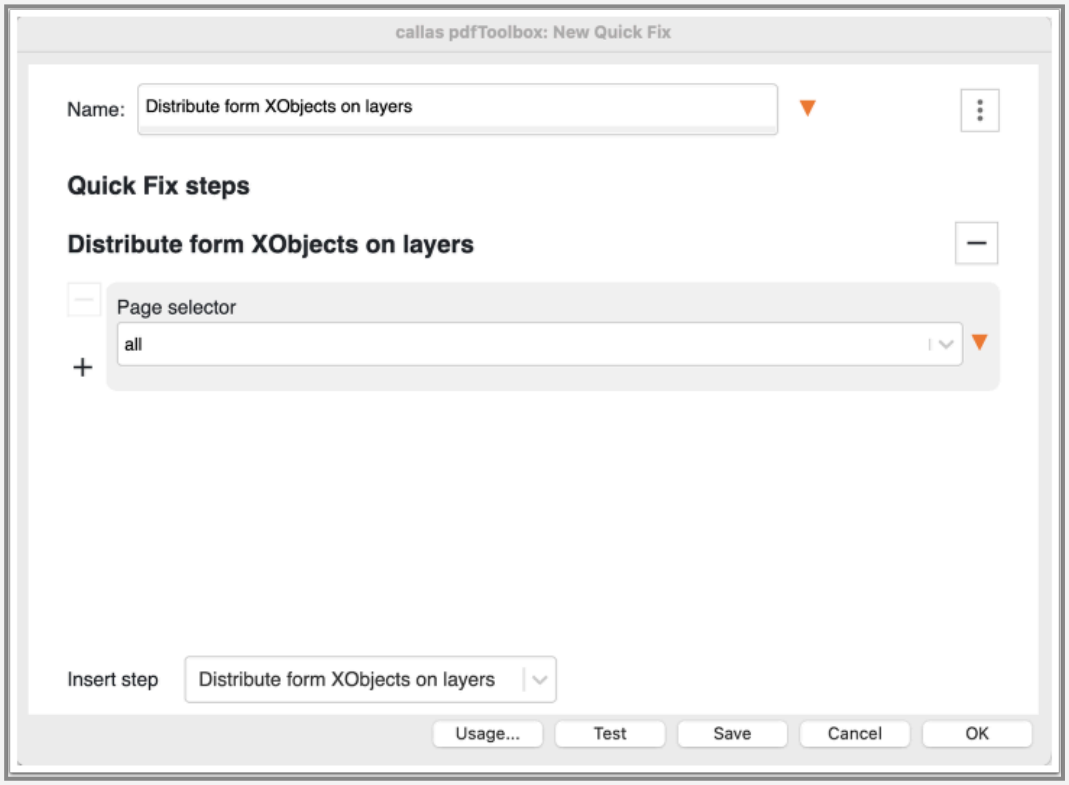
                }
        ]
}
```

Click to copy

Distribute form XObjects on layers

For variable data print (VDP) is is useful to visualize a form XObject structure as explained in this article, that describes

how that can be done interactively. If you want to create a layer structure that is derived from the form XObject structure in a PDF you should use this Quick Fix.




```
{
        "quickfixes": [
                {
                        "quickfix": "distribute_xobj_on_layers",
                        "version": "1.0",
                        "instructions": [
                                {
                                        "page_selector": "all"
                                }
                        ]
                }
        ]
}
```

Click to copy

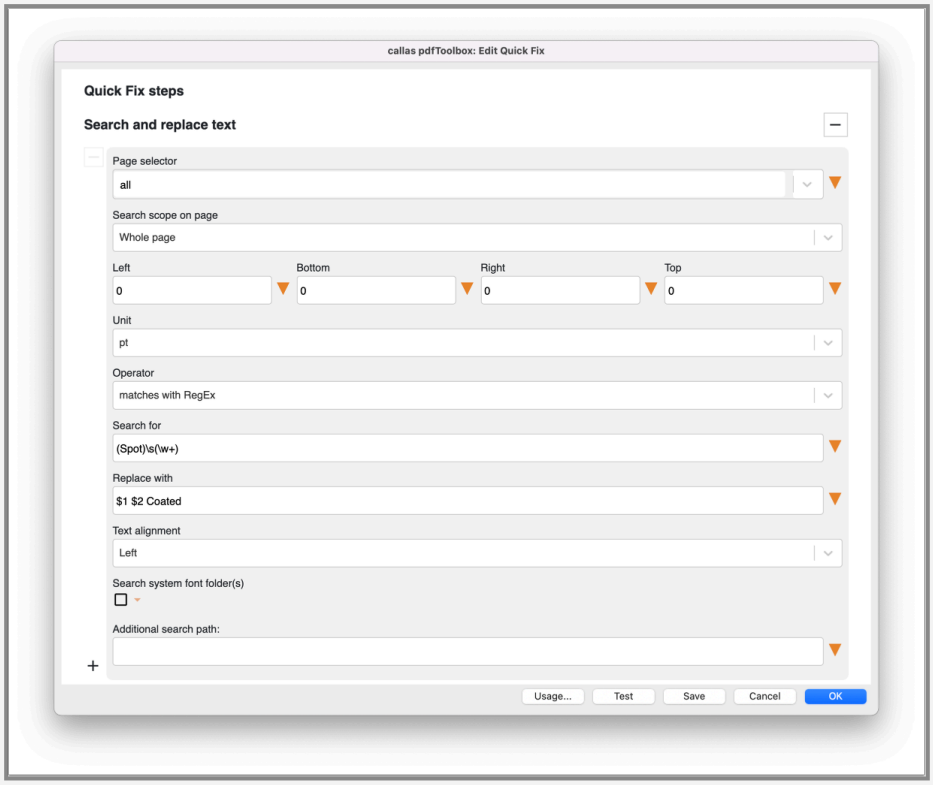
Search and replace text

This Quick Fix feature is available since **pdfToolbox 14** and searches and replaces text in a document.

The operator parameter can be used to specify the text to be searched for. Either you enter the exact text to be searched (Operator: equal to) or you use a RegEx expression, which allows more complex search patterns (Operator: matches with RegEx). More information about RegEx expressions can be found here.

For the text to be replaced, the alignment (left-aligned, right-aligned, centered, justified) can be specified. "Left-aligned", "right-aligned" or "centered" does not change the width of the text. If "justified" is selected, the text is compressed or stretched within limits.

If the checkbox (Search system font folder(s)) is not activated and no path to a explicit font is specified, only the characters embedded in the PDF file can be used for replacement.



```
{
        "quickfixes" :
        [
                {
                        "instructions" :
                        [
                                {
                                        "alignment" : "left_aligned",
                                        "bottom" : 0.0,
                                        "font_locations" :
                                        {
                                                "in_system_fonts_folder" : false,
                                                "in_user_fonts_folder" : false,
                                                "use_paths" : false
                                        },
                                        "left" : 0.0,
                                        "operator" : "regex",
                                        "page_selector" : "all",
                                        "replace_with" : "$1 $2 Coated",
                                        "right" : 0.0,
                                        "search_for" : "(Spot)\\s(\\w+)",
                                        "search_scope" : "all",
                                        "top" : 0.0,
                                        "unit" : "pt"
                                }
                        ],
                        "quickfix" : "replace_text",
                        "version" : "1.0"
                }
        ]
}

//                "alignment": "right_aligned|left_aligned|cen-
ter_aligned|block_aligned"
//                "in_system_fonts_folder" : true|false
//                "in_user_fonts_folder" : true|false
//                "use_paths" : true|false -> if true -> "paths" : [""]
//                "page_selector": "all|even|odd|<splitscheme_expression>"
//                "operator": "regex|equal_to"
//                "unit": "pt|mm|inch"
//                "search_scope": "all|MediaBox|CropBox|BleedBox|TrimBox|ArtBox|ab-
solute"
```

Click to copy

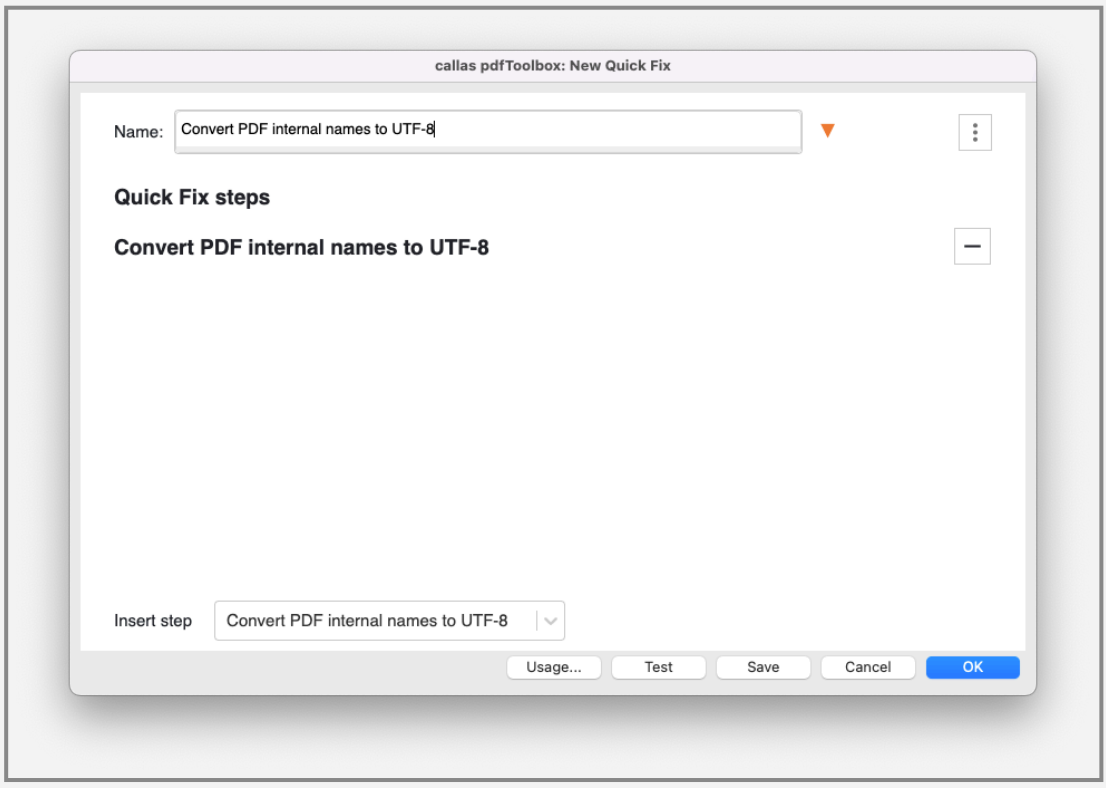
The example shown above uses a RegEx expression ((Spot)\s(\w+)). This has the effect of inserting the word "Coated" after each spot color name (e.g. Spot Green, Spot Yellow, Spot Red will be replaced in Spot Green Coated, Spot Yellow Coated, Spot Red Coated).



There is also a "Search and replace" function that can be applied to certain types of **variable data printing**. Learn more about this in the following article: Create VDP files from PDF templates.

Convert PDF internal names to UTF-8

Some PDF based ISO standards require that all internal names are encoded in UTF-8. All PDF internal names that do not comply with UTF-8 will be renamed.



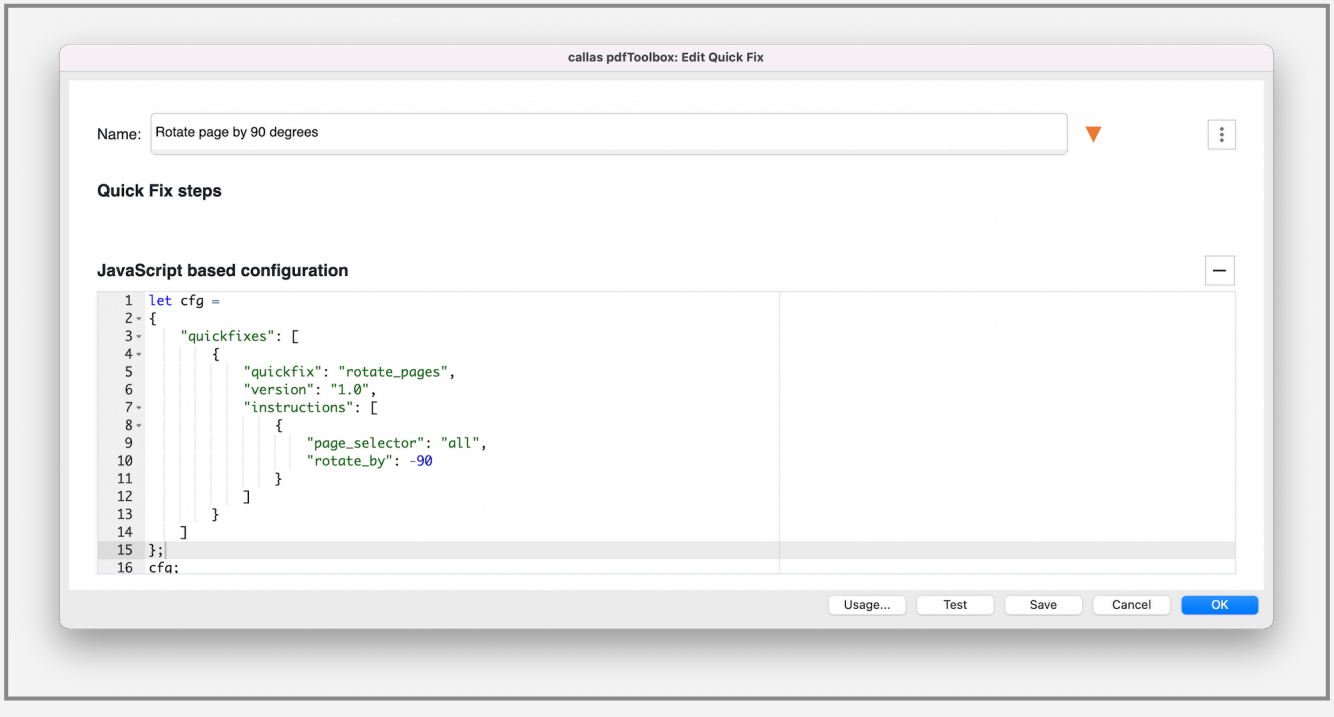

```
{
        "quickfixes" :
        [
```

```
                {
                        "instructions" :
                        [
                                {}
                        ],
                        "quickfix" : "repair_names_to_utf8",
                        "version" : "1.0"
                }
        ]
}
```

Click to copy

JavaScript based configuration

This configuration must return a JavaScript object that has a JSON serialisation according to the specification for the Quick Fix features in the form as documented above. Read more about it here.



29.5 Using Quick Fix on the command line

On the command line (CLI), pdfToolbox provides the possibility to run a QuickFix without loading the whole pdfToolbox executable, instead, only the part of pdfToolbox is loaded and executed that carries out a QuickFix. This results in significantly shorter launch times. In addition, at least when used/configured accordingly, no files are copied or completely rewritten, again reducing processing time.

In the simplest case, a call on the command line would look like this:

```
./pdfToolbox --quickfix my_quickfix_config.json my_pdf_file.pdf
```

Click to copy



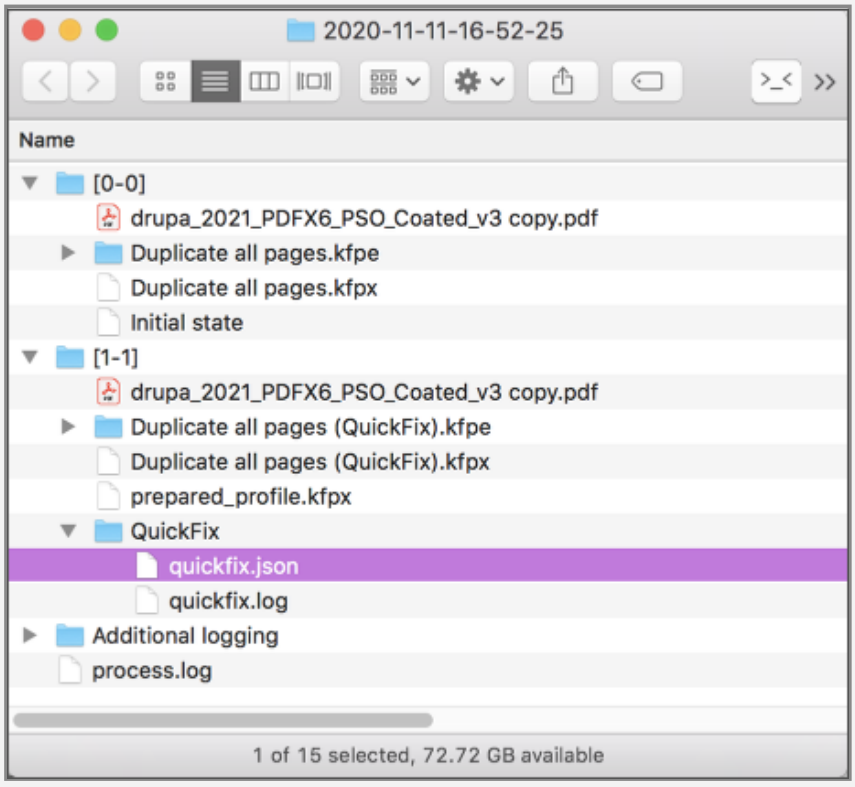
Use `./pdfToolbox --help quickfix` on the command line for more information (though all other parameters for `quickfix` are generic pdfToolbox parameters and not specific to QuickFixes).

The second parameter – called `my_pdf_file.pdf` in the example – specifies the PDF file to process. It is intentional in the example, that no output file is specified. Instead, the original file will be modified (in thee form of an incremental update). This guarantees to fastest possible execution. If an output PDF file is specified, the original PDF will first be copied to the destination, and than that copy of the PDF file be processed 'in situ'.

The first parameter – called `my_quickfix_config.json` in the example, contains a JSON expression defining the QuickFix instructions to be carried out. The syntax for each QuickFix feature is shown in the article QuickFix features.

Use "Log profile execution" to capture a JSON representation of a QuickFix in pdfToolbox Desktop

The easiest way to construct a QuickFix configuration file as a JSON expression is to create a ProcessPlan with a QuickFix step in pdfToolbox Desktop. Running that Process Plan with "Log profile execution" turned on, the log output will also contain th QuickFix definition as a JSON file:



29.6 Reorder pages using Quick Fix

There are two new QuickFixes in pdfToolbox 13 to reorder pages:

- Reorder pages
- Invert page order

QuickFix: Reorder pages



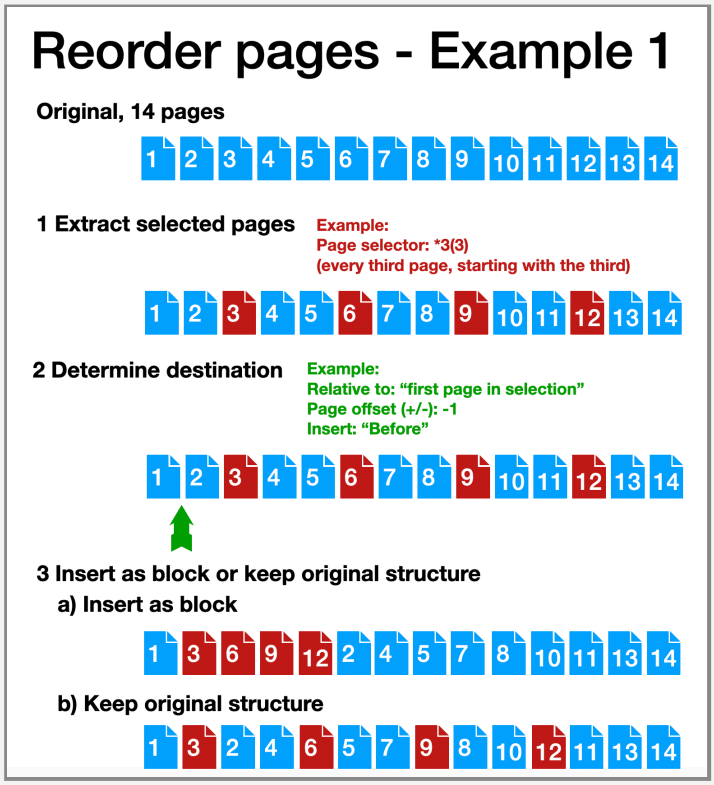
1. Page selector: A page selector expression can be a list e.g. 1,2,3 or even a simple expression, e.g. 3--3 which will find all pages except the first and last two (last page being -1). It can even use more advanced syntax: e.g. *3 will select every third page. More info here.
2. Relative to:
   - first page in PDF OR
   - first page in selection OR
   - last page in selection OR
   - last page in PDF
3. Page offset: The anchor point by which the selected pages are to be reordered
4. Insert:
   - Before OR
   - After
5. Non consecutive selections:

- Insert as consecutive pages: insert the selected pages as a block
- Shift page structure: insert the selected pages keeping their original structure

Usage examples

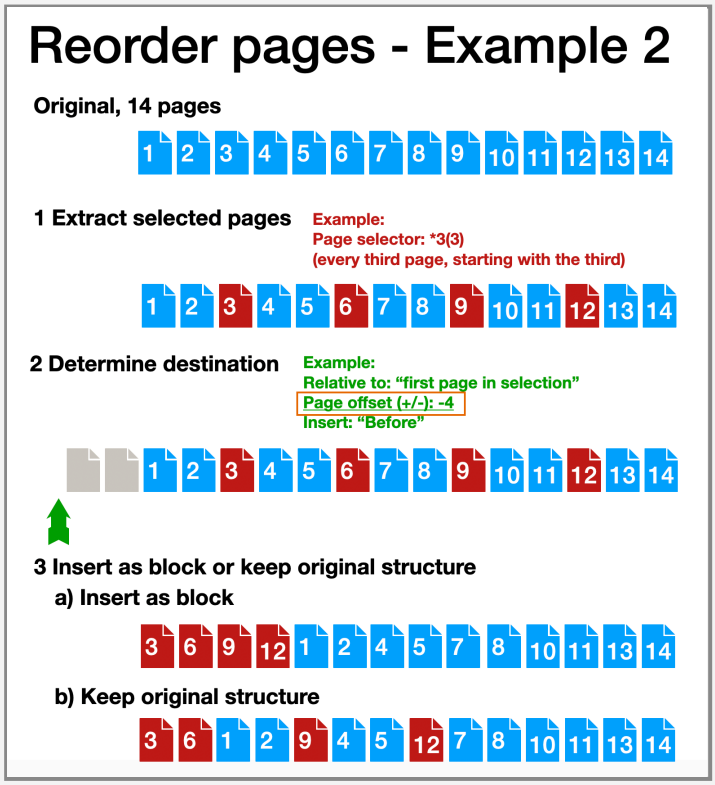
Example 1: The below example shows the original PDF with 14 pages

1. Page selector: *3(3) selects every third page in the document starting with the third page
2. Relative to: 'first page in selection' which is page 3
3. Page offset: set to -1 which brings the anchor point to page 2
4. Insert: before, which means before page 2
5. Non consecutive selections:
   - Insert as consecutive pages: inserts the selected pages (3,6,9,12) consecutively before the anchor point
   - Shift page structure: inserts the selected pages (3,6,9,12) keeping their original structure which means, for example: for the selected page 6, 'page offset' of -1 will bring the anchor to page 5 and 'insert before' would insert page 6 before page 5

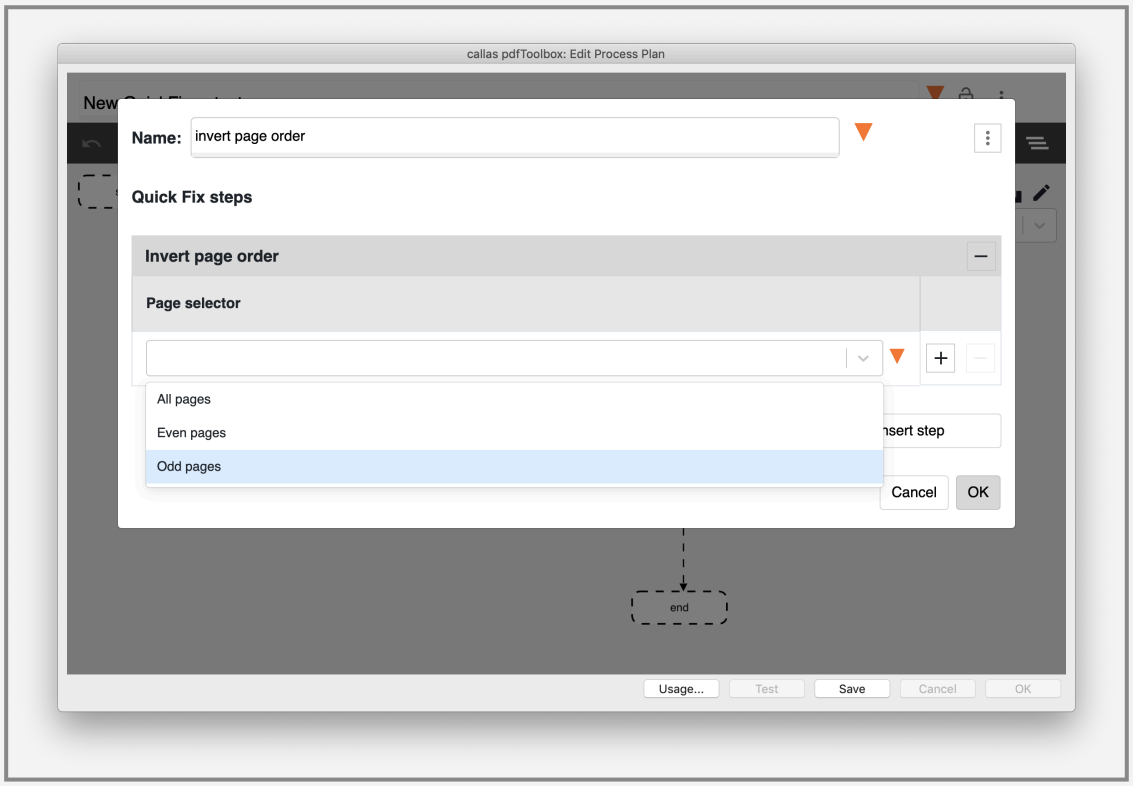


Example 2: The below example shows the original PDF with 14 pages

1. Page selector: *3(3) selects every third page in the document starting with the third page
2. Relative to: 'first page in selection' which is page 3
3. Page offset: set to -4 which brings the anchor point to 2 pages before the first page
4. Insert: before, which means before the anchor point
5. Non consecutive selections:
   - Insert as consecutive pages: inserts the selected pages (3,6,9,12) consecutively before the anchor point
   - Shift page structure: inserts the selected pages (3,6,9,12) keeping their original structure which means, for example: for the selected page 6, 'page offset' of -4 will bring the anchor at page 1 anchor point and 'insert before' would insert page 6 before page 1



QuickFix: Invert page order



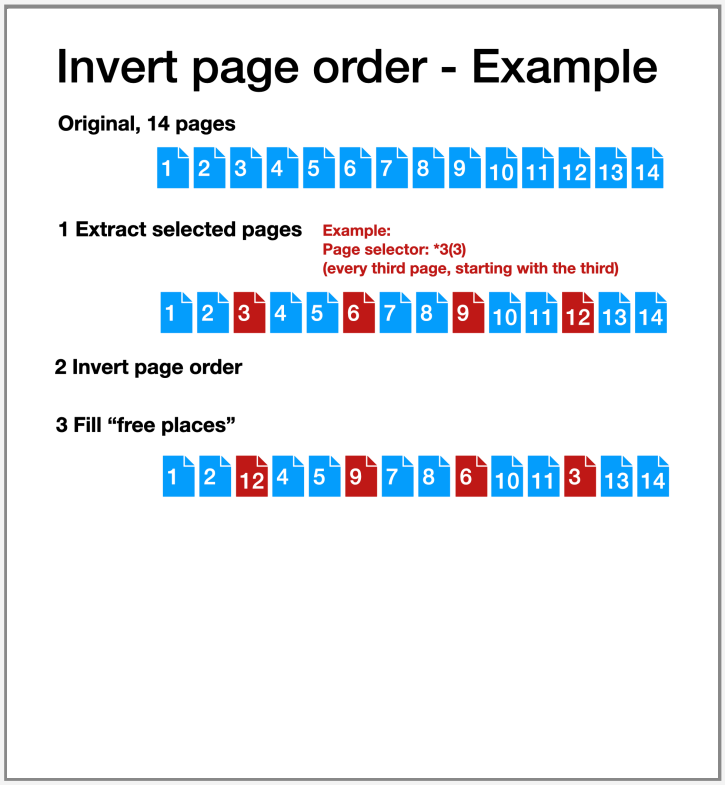
1. Page selector: The page selector can be selected from the dropdown:
   - All pages
   - Even pages
   - Odd pages
   - It can also be an expression which can be a list e.g. 1,2,3 or even a simple expression, e.g. 3--3 which will find all pages except the first and last two (last page being -1). It can even use more advanced syntax: e.g. *3 will select every third page. More info here.

Usage example

The below example shows the original PDF with 14 pages

1. Page selector: *3(3) selects every third page in the document starting with the third page

Upon execution, the selected pages will be inverted.



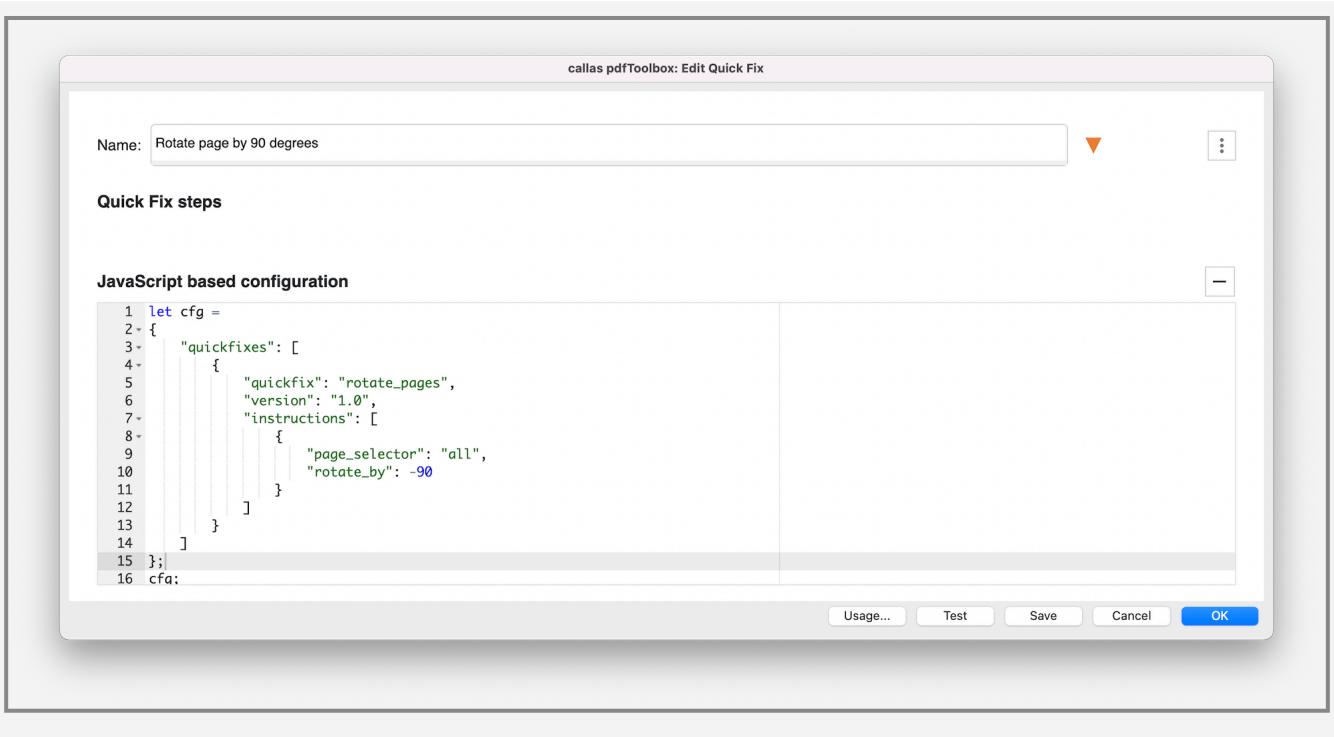
29.7 JavaScript based configuration Quick Fix

Things that can be done with Quick Fix can be done very fast. Sometimes you can only take full advantage of that when you are using JavaScript. The "**JavaScript based configuration**" Quick Fix can be used as follows:

```
let cfg = {
        … Your Quick Fix JSON serialisation
};
cfg; //<- return your Quick Fix object
```

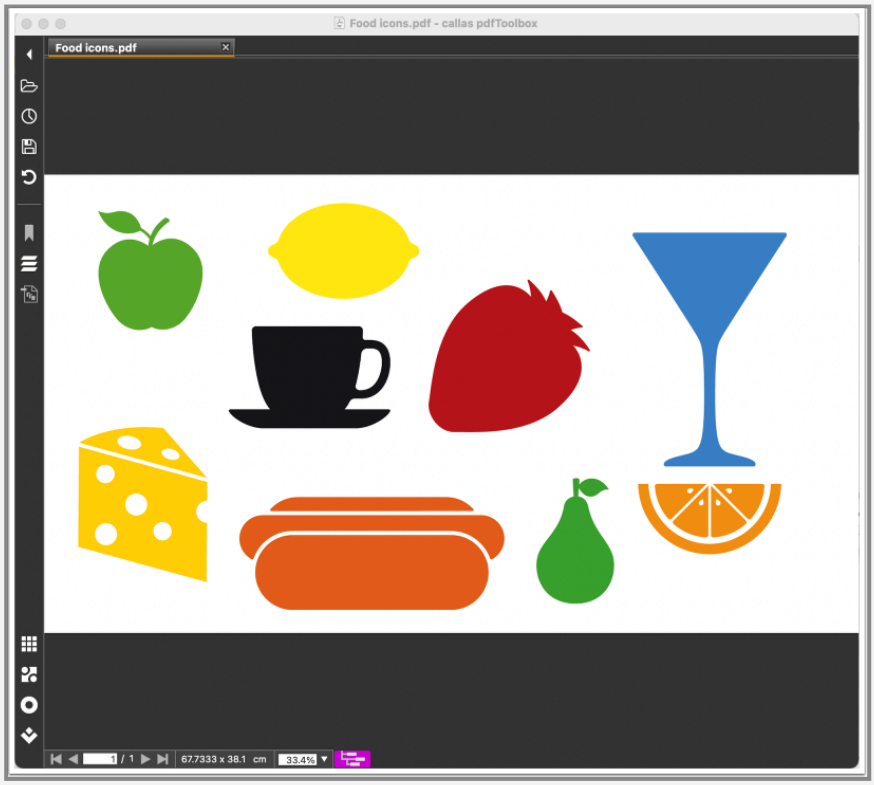
Click to copy

The JSON serialisation for the existing Quick Fixes are documented here.



Example of a JavaScript based QuickFix as Process Plan steps

In this example we have a single page PDF with 10 objects. What we want to do is to split this page so that each page in the result PDF has only one object.



Food_icons.pdf

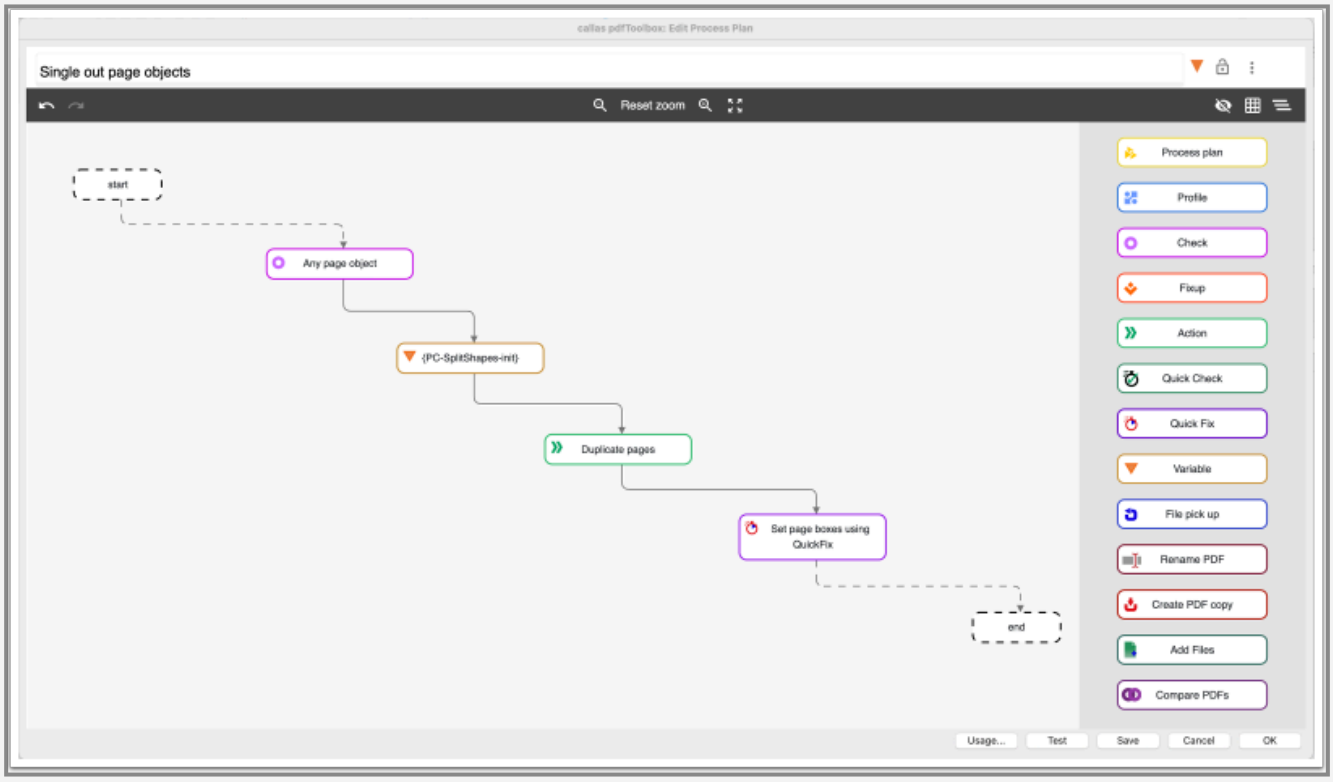
We know that we can set the Crop Box to the dimensions of an object, so what we want to do is to duplicate the first page, set the Crop Box to the dimensions of the first object, then duplicate the first page again for the second object and so forth. We also know that we can determine the dimensions of an object using a Check.

So what we could now do is to create a Check that finds any objects and then loop over the result list in a Process Plan duplicating the pages and setting the Crop Box. We could use a Quick Fix for setting the Crop Box because that is much faster than a Fixup. The problem is that the solution is not really quick because all of the overhead for our Process Plan based loop.

A much smarter way is to first create as many duplicates from the first page as we need (as many objects as we have minus one because we can use the existing page). Then we create a list of Quick Fix instructions, one for each page and for the re-

spective object that should be visible on that page and then we run Quick Fix.

The Process Plan "Single out page objects"



Single_out_page_objects.kfpx

Step 1 to 3

1. Check "Any page object" is a single Check using the property "Is any page object". It creates a result object.

2. Variable / JavaScript "PageObjects-init" initializes app.vars.objects with all hits (all objects) and creates a variable "app.vars.nbhit" with the number of objects minus one, that is the number of pages that we need to create.

```
app.vars.objects = app.doc.result.checks[0].hits;
```

```
app.vars.nbhit = app.vars.objects.length -
1;
```

3. Action "Duplicate pages" duplicates the first page according to what has been defined in app.vars.nbhit.

Step 4 "Set page boxes using Quick Fix"

This is a Quick Fix that uses "JavaScript based configuration". It first creates its own list of instructions and then applies them on the PDF file. Let us see what it does:

```
const margin=5;
```

Specifies a margin around each object. The value is 5 [pt].

```
let instructions= [];
```

Defines an empty array that will contain our instructions (one per object/page).

Then comes a for loop that runs over our app.vars.objects array and sets the Crop Box to the dimensions (plus margin) of the first object on the first page and so forth.

```
for (let i = 0; i < app.vars.ob-
jects.length; i++) {
```

```
  let llx = (app.vars.objects[i].llx -
margin);
```

```
  let lly = (app.vars.objects[i].lly -
margin);
```

```
  let urx = (app.vars.objects[i].urx +
margin);
```

```
  let ury = (app.vars.objects[i].ury +
margin);
```

The variable "page" is used to specify the page for the current set of coordinates. It is used in the "page_selector" object for the instructions below.

```
  let page = (i + 1).toString();
```

```
  instructions.push({
```

```
                    "bottom" : lly,
```

```
                "from_box_or_absolute" : "absolute",
                "left" : llx,
                "page_selector" : page,
                "right" : urx,
                "top" : ury,
                "unit" : "pt",
                "when" : "always",
                "which_box" : "CropBox"
            });
}
```

Now we have created an array "instructions" in the syntax that is expected by the Quick Fix to set page geometry boxes and that has a set of coordinates for the CropBox and a page_selector for the page it should be used on.

We now put this into a full Quick Fix configuration:

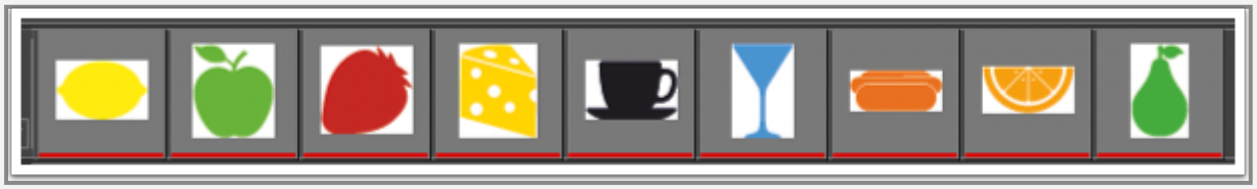
```
let cfg =
{
    "quickfixes" :
    [
        {
            "instructions" : instructions,
            "quickfix" : "set_page_box",
            "version" : "1.0"
        }
    ]
```

```
};
```

And finally we establish that configuration by calling it:

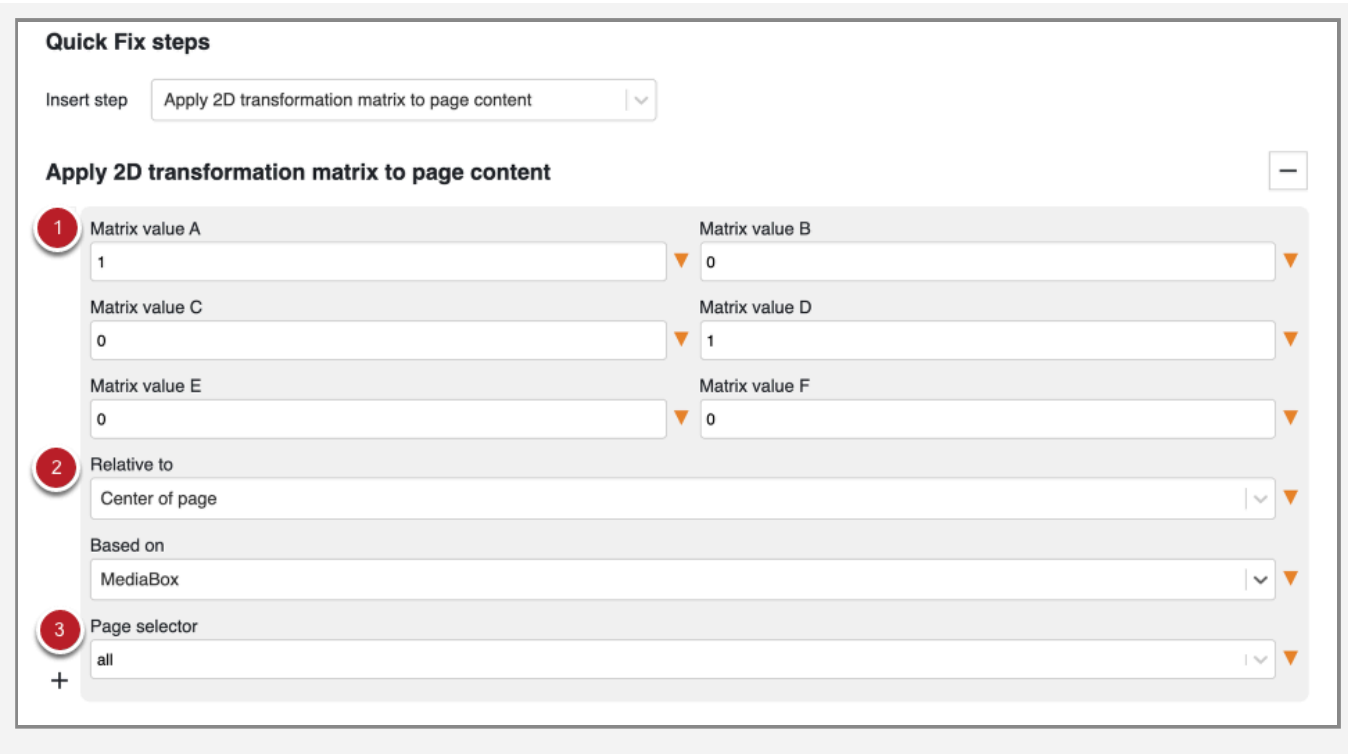
```
cfg;
```

The result



29.8 Apply transformation matrix

This quick fix allows applying a generic 6-element transformation matrix on the content of a page. The quick fix step for it is called "Apply 2D transformation matrix to page content". The quick fix transforms *only* the page content; it does not modify any of the page geometry boxes in the PDF document.



The quick fix needs three important pieces of information:

1. What is the matrix that is going to be applied?
2. What is the origin of the transformation?
3. Which pages are affected?

The matrix

There are different ways in which a transformation matrix can be specified. This quick fix uses the method identified and explained by the PDF specification. Some examples of useful matrices:

The identity matrix

```
1                0
0                1
0                0
```

Click to copy
This matrix is a "null operation"; it doesn't modify the pages it is applied to.

Translation (move) by 1 inch up and to the right

```
1                0
0                1
72          72
```

Click to copy
This matrix only performs a translation of all page content. The translation is done by 72 points (or 1 inch).

Scale down to 50%

```
0.5          0
0                0.5
0                0
```

Click to copy
This matrix performs scaling of all page content to 50% of the current size.

The transformation origin

Transformations are always done around a transformation origin point. For a rotation for example, this is the point the page is rotated around. The "Relative to" and "Based on" properties allow selecting which point of which page box is going to be used as the transformation origin point.

The pages

The "Page selector" property determines which pages are affected by the transformation. The default is "all" so that the transformation is executed on all pages, but all other page selectors are supported.

30. Imposition

30.1 Overview of imposition in pdfToolbox

pdfToolbox contains a somewhat unusual but quite complete imposition engine. It has significantly grown over the years, and is used in the product in a few different ways. All that together makes this chapter about imposition a long one. This article explains some of the basic principles and history behind the imposition engine, provides some insight into capabilities and limitations, and introduces the three big ways the imposition engine can be used today. The rest of the chapter fills in the details on all of this.

The callas YouTube channel contains quite some information about imposition as well; the different articles in this chapter are linked appropriately with individual videos.

Architecture of the engine

Where many imposition applications use visual template editors, and are optimized to handle imposition tasks manually, this has never been the goal of the pdfToolbox imposition engine. pdfToolbox focuses (as in most other places) on automation. As a result, the functionality is oriented much more toward reaching the highest possible throughput, rather than a comfortable user interface.

Because of this, the engine has a quite steep learning curve sometimes. But at the same time, it is ideally suited for tasks that are perhaps not 'classic' imposition. The imposition engine is used in many different projects where other imposition applications wouldn't be suitable. And because the imposition functionality can be combined with all of the other functionality in pdfToolbox (for example in process plans), it can be incredibly versatile.

Capabilities and limitations

In its most elementary form, the imposition engine takes a single input file, and produces a single output file. Because pdfToolbox also provides the capabilities to split and merge PDF files, pull in additional PDF files where needed, and do

advanced calculations using JavaScript, the engine still covers a lot of ground.

The engine does have a number of limitations:

- There is no visual template editor. While there are a few simple imposition capabilities, the majority of the imposition functionality requires you to write imposition configurations. This makes the imposition engine impractical for situations where there is a high variability of input jobs (one-offs).
- There is no built-in support for ganging multiple files together (Ganging as in, combining multiple PDF files or PDF file pages on a single imposed sheet while maximizing sheet coverage. As JavaScript is supported, certain scenarios can be handled quite well, but the engine has no optimization capabilities.
- Imposition is always based on rectangular pages. There is no built-in support for nesting pages together based on a cut contour shape. Again, if you know what jobs need to be imposed and what their geometry is, the imposition engine can be used to impose them, but it cannot calculate optimal coverage by itself.

Different imposition modes

This chapter explores three big ways the imposition engine can be used. Summarized, they are:

1. **Simple Imposition**
   These are imposition features for which no imposition configurations need to be made. Examples are "Step and repeat", "Fill page", "Booklet imposition"... These are typically available in the Switchboard, and as Actions, both for use in process plans, and through the CLI or SDK. All help articles regarding this functionality start with "Simple imposition".
2. **JavaScript imposition**
   The modern way to create imposition configurations in pdfToolbox is using JavaScript. All articles beginning with "JavaScript imposition" speak specifically about this type of imposition configuration. The imposition configurations created this way can be used in the Switchboard, but also using an "Impose" action, both in process plans, and through the CLI or SDK.

3. **Old-style imposition**
   Before JavaScript imposition configurations existed, pdfToolbox featured its own tabbed-text imposition configuration language. While this is quite powerful, it can also be a bit of a pain to get correct. Because of this, and because the JavaScript imposition now supports more features than the old-style language, it is advised to use JavaScript imposition instead. The help articles for the old-style imposition are still available in this chapter: they all start with "Old-style imposition".



Besides these three imposition modes, this chapter contains a number of more generic articles. You're reading one now, but there are also articles on terminology, editing and debugging configurations, and so on. They are typically useful for more than one imposition mode.

30.2 Simple imposition - how to use

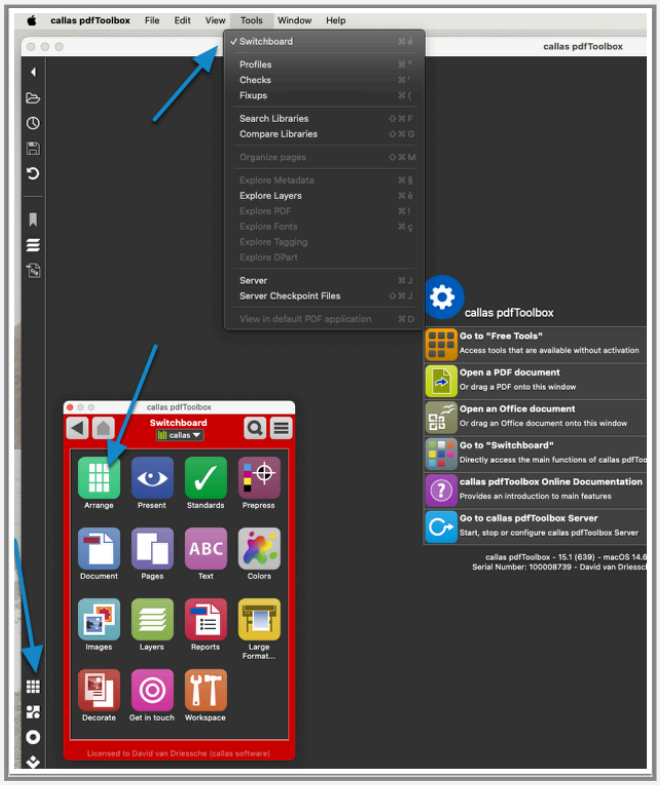
What is called simple imposition in pdfToolbox, are those imposition capabilities for which no imposition configuration needs to be made. These types of imposition can be used by providing some relatively simple pieces of information. Examples are "Step and repeat", "Fill page", "Booklet imposition"...

The actual imposition methods (the details for what is available for "Step and repeat" and so on) are described in the next article; this article details where you find this functionality and what the different ways are you can use it manually and automatically.

Using the Switchboard

All simple imposition functionality is available on the Switchboard. To get to it:

1. Click the "Tools" menu, and then "Switchboard",
2. Click the "Switchboard" quick access button in the left-hand side bar.
3. If the Switchboard doesn't look like the screengrab below, click on the "Home" button (the second one from the left).
4. Click on the "Arrange" button.

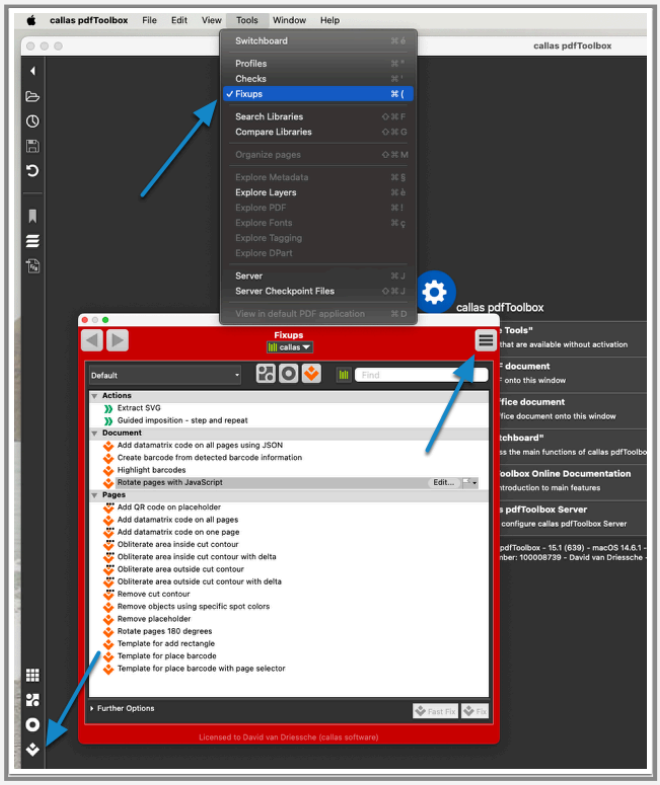


On the "Arrange" pane are all quick imposition capabilities pdfToolbox has. Each of them is described in more detail in the next article.

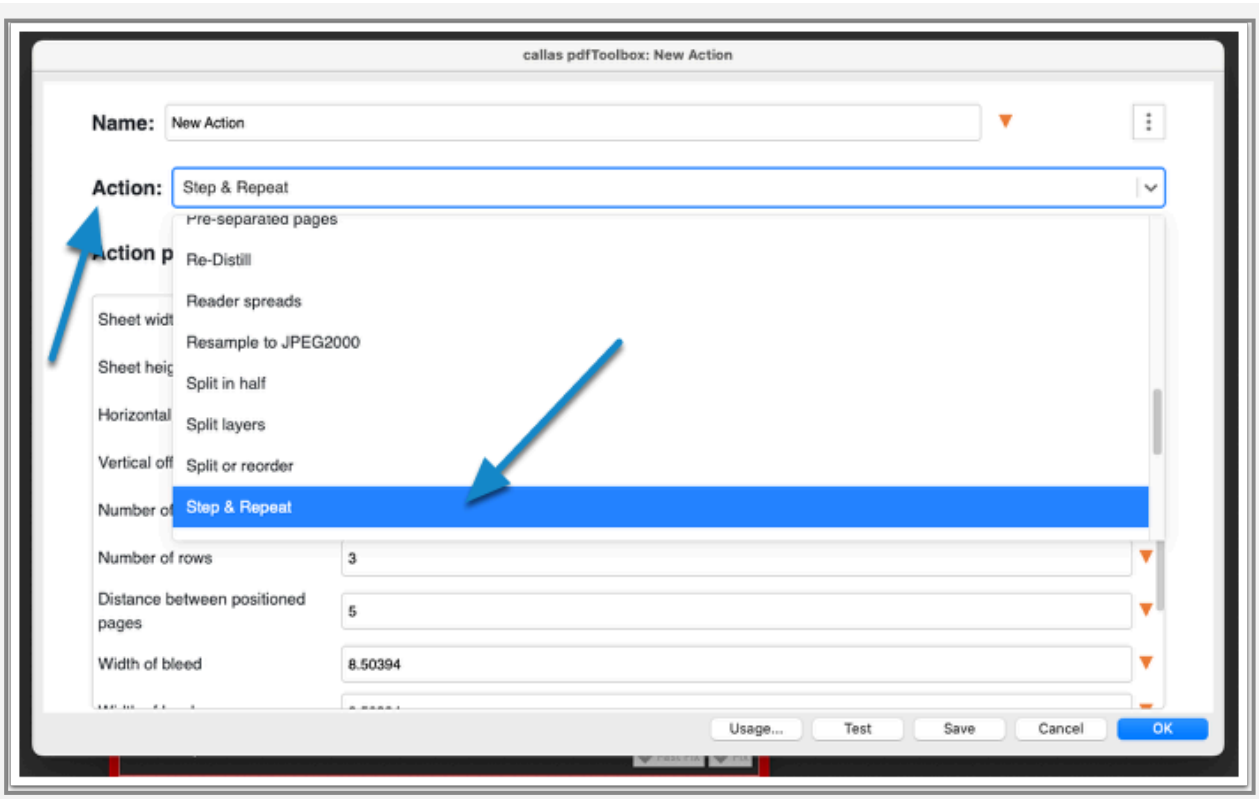
Using actions

To automate the simple imposition functionality, create a new action:

1. Click on the "Tools", and then "Fixups" menu item, or click directly on the "Fixups" quick access button in the left-hand side bar.
2. In the fixups window that opens, click on the "Hamburger" button in the top right corner.
3. This opens up a menu, in which you can choose "Create Action...".



After that, the "New Action" dialog window opens up. There is a list of all available action types. Click that, and in this list, you'll find all simple imposition functionality.



After configuring the action exactly how you want it, click on "OK" and it will appear in your fixups list. You can now either export it directly to a kfpx file, or you can add it into a process plan.

Using the CLI

All simple imposition functionality is available as CLI commands directly. Of course you can embed the functionality in a process plan, and then run the process plan from the command-line, but with these CLI commands, you can also do it directly. In order to know what is available, check the chapter on pdfToolbox CLI.

Using the SDK

All simple imposition functionality is available as direct calls in the SDK. Of course you can embed the functionality in a process plan, and then run the process plan from the command-line, but with the SDK, you can also do it directly. In or-

der to know what is available, check the chapter on pdfToolbox SDK.

30.3 Simple imposition - available methods

This article describes the different simple imposition features available in pdfToolbox, and gives an overview of what you can do with them. How you can use them manually or automatically is described in the previous article. Keep in mind that not all parameters described in this article are necessarily available everywhere you can use this functionality. The Switchboard, for example, provides a simplified version of the functionality.

Booklet

This imposition method prepares an incoming PDF file for double-sided printing. Essentially, this method expects a PDF file with single pages (not spreads). It then creates imposition sheets that have two pages on each sheet. The pages are ordered on the imposed sheets in such a way, that all imposed sheets can be printed, and then folded to form a booklet. If your incoming PDF file doesn't have single pages, but spreads, you can use the "Split in half" imposition method to separate them into single pages first.

The imposition can be done either on a specific sheet size, or on the minimal sheet size possible for your input document. Additional parameters allow specifying an offset (this determines the distance from the left bottom corner and the imposed pages), whether you want trim marks, and the width of the bleed and border area. Shingling settings allow adjusting for the paper thickness while folding (there is a dedicated article about shingling, also called creep).

Fill page

Fill page expects a multi-page PDF file as input document. It then creates imposed sheets of the size specified, and fills each imposed sheet with as many pages from the input document as possible. The way this is done is not full sheet optimization; essentially it creates a grid (with the size of each element of the grid determined by the biggest page in the in-

coming PDF file) and flows PDF pages into this grid until all pages have been positioned. Pages are not rotated nor scaled to place them on the imposed sheets.

Additional parameters allow specifying how much bleed each placed PDF file should show (bleed is not created, but clipped at the size specified, and how much space there should be between placed pages. Placing cut marks is also possible.

Fill page (repeat)

Even though it has "Fill page" in its name, this imposition method is closer to "Step and repeat". Where in the step and repeat method, you have to specify how many rows and columns you want, this imposition method calculates the maximum number of rows and columns that will fit the imposed sheet.

The parameters are the same as for "Fill page": the size of the imposed sheet, how much bleed each placed PDF file should show (bleed is not created, but clipped at the size specified, and how much space there should be between placed pages. Placing cut marks is also possible.

N-up

The N-up imposition method expects a multi-page PDF input file. It then places different pages from the input file sequentially in a grid pattern, where you specify the number of rows and columns.

The imposition can be done either on a specific sheet size, or on the minimal sheet size possible for your input document. Additional parameters allow specifying an offset (this determines the distance from the left bottom corner and the imposed pages), whether you want trim marks, and the width of the bleed and border area.

Reader spreads

Reader spreads expects a PDF file with single pages (as opposed to spreads consisting of two pages on one sheet). The

imposition method combines two single pages into a spread without reordering the pages. Optionally, the first page can be handled as a cover page and left as a single page. This means:

- Without the cover page option, page 1 and 2 are combined, then page 3 and 4, and so on.
- With the cover page option, page 1 remains a single page, page 2 and 3 are combined, page 4 and 5, and so on.

The imposition can be done either on a specific sheet size, or on the minimal sheet size possible for your input document. left bottom corner and the imposed pages).

Split in half

Sometimes, when you have a document that was exported with reader spreads (meaning, two pages are placed on each imposed sheet), you need single pages instead. Examples would be if you wanted to format the document as a booklet, or use it in an online PDF viewer, or... This imposition method does exactly that. It assumes your document was exported as reader spreads, and has 2 pages on each sheet.

The imposition method cuts each page in two to give you single pages again. If your document also contains single pages (perhaps the first page is a cover page), the single pages are not cut in half.

Step and Repeat

This imposition method creates one imposed sheet for each page of the incoming PDF file. On this imposed sheet, it places copies of that page in a grid-like fashion depending on the amount of rows and columns you specify. So an incoming 3-page PDF file gives rise to a 3-page imposed PDF file, where page 1 contains multiple copies of page 1 of the incoming file, and so on.

The imposition can be done either on a specific sheet size, or on the minimal sheet size possible for your input document. Additional parameters allow specifying an offset (this determines the distance from the left bottom corner and the imposed pages), whether you want trim marks, and the width of the bleed and border area.

30.4 Simple imposition - defining sheet sizes

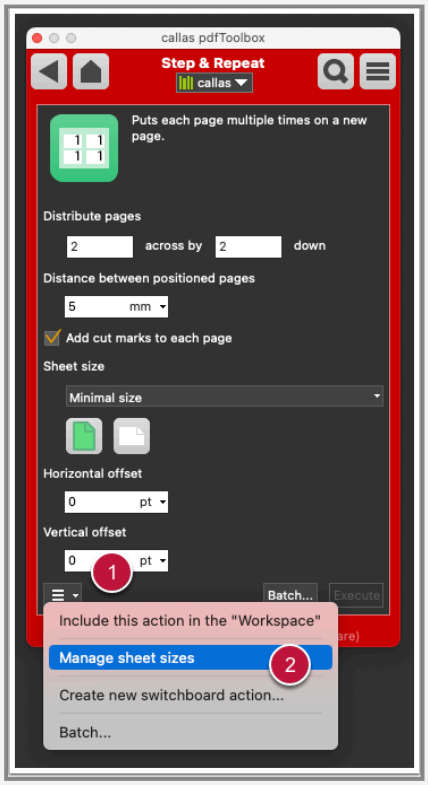
Many simple imposition features let you choose between a number of standard sheet sizes. By default, the list contains a few European, North American, and Asian standard formats. It is possible to edit this list, either to remove formats, or add custom formats that you frequently use.

The sheet sizes are stored in the library. This means you can have different custom sheet sizes in each library you create in pdfToolbox Desktop.

Opening the sheet size editor

To do this, follow these steps:

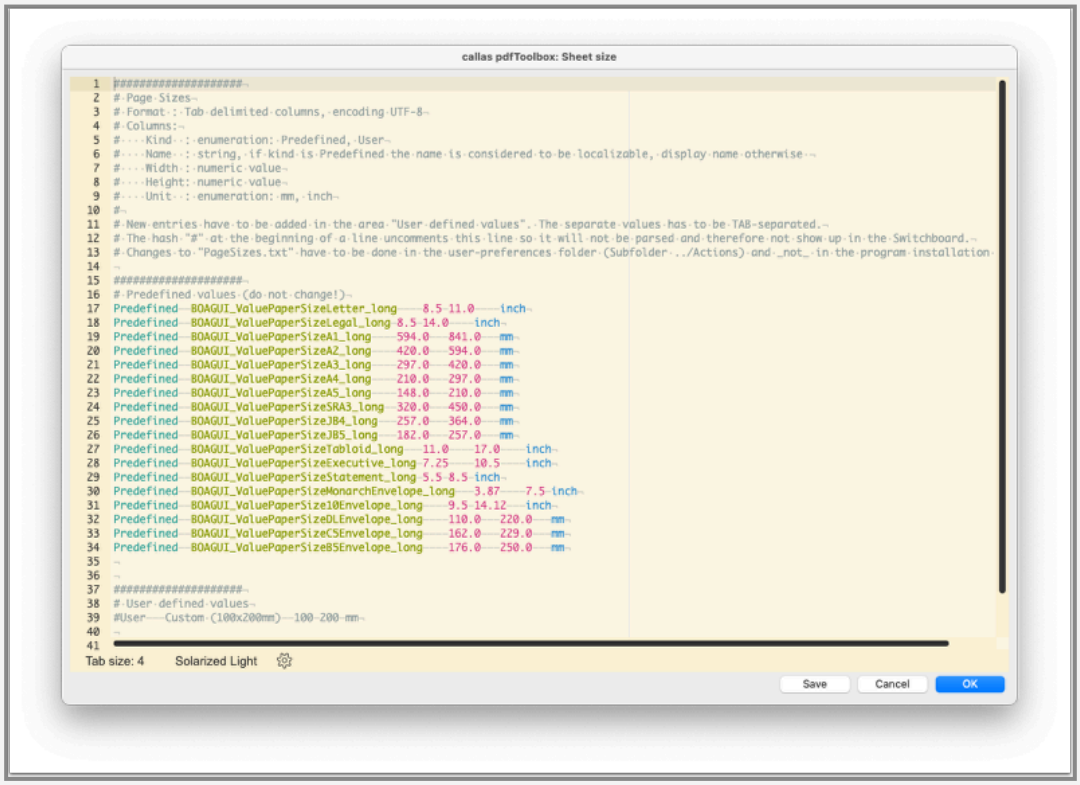
- Open the Switchboard.
- Click on the "Home" button if necessary, and then on "Arrange".
- Click on any of the simple imposition methods. Shown below is "Step & Repeat".



1. Click on the "hamburger" button in the bottom left corner.
2. In the menu that appears, click on the "Manage sheet sizes" menu item.

Understanding the editor

The sheet size editor window opens, and shows all current sheet sizes. The editor supports syntax coloring and code completion and shows error indications in the left margin if relevant.



Keep in mind the following rules:

- Each line that starts with the pound sign ("#") is interpreted as a comment, and ignored.
- Empty lines are ignored.
- The different items on each line are separated with tab characters.
- There are two sections: a predefined section you should not change (marked with "Predefined values (do not change!)", and a user-defined section (marked with "user defined values").

Adding a custom sheet size

The easiest way to add a custom sheet size is to duplicate the template line.

```
#User        Custom (100x200mm)        100        200        mm
```

Click to copy

Remove the pound sign, and modify the different items on the line to your liking. The comment at the top of the editor explains all fields.

```
User        Custom 50x30 Business card        50        30        mm
```

Click to copy

Make sure to preserve the tabs in between the fields, if you don't your format isn't going to work (and the editor shows a red error mark in the left margin). You can add as many formats as you like. When you're done, click "OK" to save the custom formats.

30.5 Getting started with imposition configurations

In contrast to the simple imposition methods, pdfToolbox also supports impositions based on imposition configurations. These consist of two types of files: referred to as sheet definitions and runlists, that need to be manually edited. Each imposition type needs its own configuration files, but imposition configurations can be quite dynamic (so you can create configurations that cover different page sizes, options, and even imposition methods, if you so desire.

Types of imposition configurations

Two types of imposition configurations are supported: JavaScript (which is the advised way of working with imposition in pdfToolbox), and old-style imposition (which has been in the product for a long time, isn't going away, but in general is more difficult to use, less flexible, and more error-prone).

In both cases, two files are used: a runlist file and a sheet definition file (but in the case of JavaScript imposition configurations, the sheet definition can be empty).

Using imposition configurations

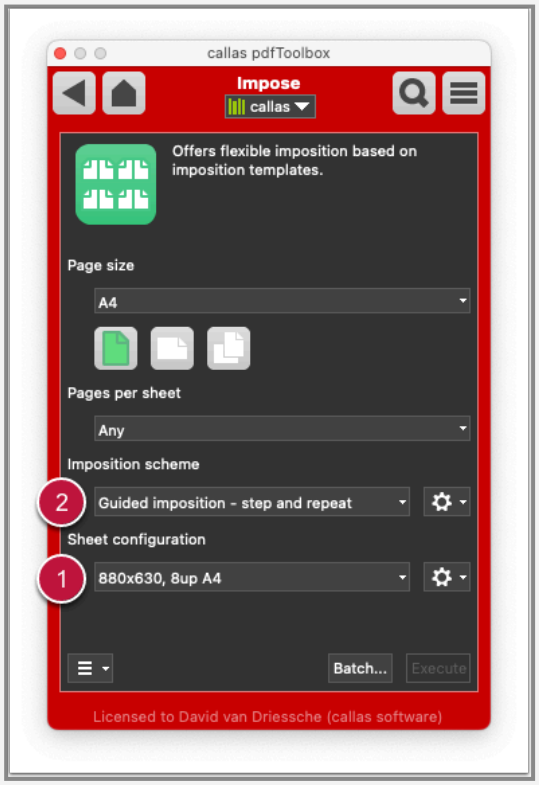
The rest of this chapter goes into detail on how to create imposition configurations, both for old-style, and for JavaScript configurations. The rest of this article points to how you can use these configurations manually or automatically.

Using the Switchboard

To use configuration-based imposition functionality in pdfToolbox Desktop, open the Switchboard. Click on the "Home" button, then on "Arrange".



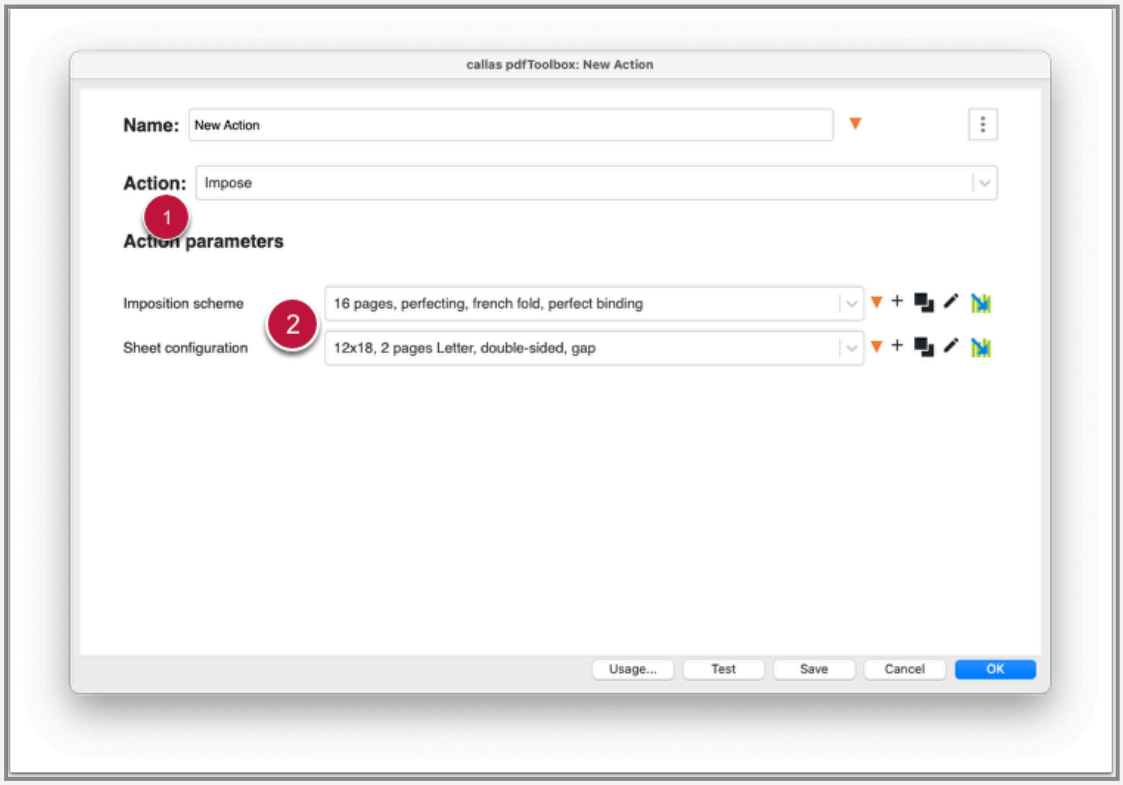
The "Arrange" area shows all simple imposition methods, and the "Impose" button. Click on "Impose" to get to the configuration-based imposition functionality.



The "Sheet configuration" menu shows all sheet definition files found in the current library. The "Imposition scheme" menu shows all runlists in the current library. Both the old-style configuration files and the JavaScript configuration files are shown in the same list. Select the configuration files you want to use, and click "Execute" to run the imposition on the currently open document in pdfToolbox Desktop.

Using an action

To use configuration-based imposition automatically, create a new action.



In the "Action" menu, select "Impose". In the action parameters, you can now select the runlist and sheet definition you want to use; both menus shows all files (old-style and JavaScript-based) in the current library.

After configuring the action exactly how you want it, click on "OK" and it will appear in your fixups list. You can now either export it directly to a kfpx file, or you can add it into a process plan.

Using CLI or SDK

Configuration-based imposition functionality is available as CLI commands or SDK calls directly. Of course you can embed the functionality in a process plan, and then use the process plan from the command-line or using the SDK as well. Check the chapter on pdfToolbox CLI, or the chapter on pdfToolbox SDK for more information.

30.6 Imposition configuration terminology - general

The JavaScript-based and old-style imposition configurations in pdfToolbox use a lot of common terminology. This article goes over the most commonly used terms, so those don't have to be repeated in the rest of the documentation.

Sheet configuration

The sheet configuration is a (mainly old-style) way to define how imposed sheets should look. Sheet configurations are saved on disk (as part of the pdfToolbox library system) as files with the extension ".sheetconfig". In JavaScript imposition configurations, the sheet configuration can be (and in most cases will be) empty, as sheet definitions and slots are typically defined as part of the JavaScript runlist.

Sheet configuration template

Normally, when a sheet is created in the imposition process, it is a blank (empty) sheet. However, the imposition engine also supports the use of sheet templates. These are PDF files that serve as a background. They allow creating PDF files with all relevant color bars and other marks, and use those as backdrop for imposed files. To select such a template, use the sheet configuration editor.

Runlist

The runlist defines the logic of an imposition. In an old-style configuration, the runlist defines which sheets are created, and which content is placed in what slots, based on the definitions in the sheet configuration. In JavaScript configurations, the runlist typically defines the sheet definitions, slots, sheets and content placement (while the sheet configuration is empty). Runlists are saved on disk (as part of the pdfToolbox library system) as files with the extension ".runlist".

Sheet definition

The imposition process ingests an input PDF file, and outputs an imposed PDF file. This imposed PDF file consists of pages that are also referred to as imposed sheets. How these sheets look is determined by a sheet definition. The sheet definition determines what the width and height of the imposed sheet is going to be, and which slots are placed on it.

This implies that, if you want to have an imposition configuration that creates at least two different imposed sheet sizes, there have to be at least two sheet definitions. Each sheet definition has a unique ID that can be used to reference it.

Sheet

The imposition process ingests an input PDF file, and outputs an imposed PDF file. This imposed PDF file consists of pages that are also referred to as imposed sheets. A sheet can be empty, or it can have content (such as pages from the input PDF file, or decorations) on it.

An imposed PDF can have as many pages as necessary. Each sheet is created by referring to a sheet definition.

Slot

A slot is a rectangular area on a sheet. It is defined by the offset of its lower left corner to the lower left corner of the mediabox of the sheet it is placed on, and its width and height. The slot also defines bleed properties, scaling, rotation, placement, cropmarks... In a slot, it is possible to place either a page from the input PDF file, or text. The content that is placed into the slot always picks up the relevant properties of the slot (an example of an irrelevant property would be the bleed settings for text placed in a slot). Each slot has a unique ID that can be used to reference it.

A sheet definition can contain as many slots as necessary. Slots do not have to be used (filled) for every sheet. It is perfectly okay to have a sheet definition with 10 slots defined and only use two or three to create a particular sheet. Each

sheet created from that sheet definition can also use *different* slots.

Slots are typically clipped to their size + any specified bleed, but can also be configured to be clipped at the slot boundary (without bleed). Slots can overlap on the output sheet if necessary.

Decorations

While slots can contain only pages from the input PDF file, and text, the imposition configuration also allows specifying decorations. These can appear anywhere on the imposed sheet (they are defined with their own parameters, independent from the slot system. Decorations include things such as: multi-line text, barcodes, lines, rectangles, circles...



Decorations are available only for JavaScript imposition configurations, starting from pdfToolbox 16.

Bleed

Slots include a bleed definition for all four of their sides. If a slot defines bleed, that amount of bleed from the input PDF file is used when a page is placed in the slot. Specifying bleed for a slot does not *create* bleed, it allows that specific amount of bleed to be visible.

Shingling (creep)

As shingling is a bigger topic, it is described in full in a separate article.

Scaling pages

Slots include two scaling factors: one for horizontal scaling (referred to as "scale x" or "horizontal"), and one for vertical scaling (referred to as "scale y" or "vertical"). By default, slots scale to 100% of true size. Other percentage values can

be used: examples would be 25, to scale down 4 times, or 200 to scale up 2 times.

Special values are 0, which signifies "scale to fit in the slot, if necessary unproportionally", and -1 which signifies "scale to fit in the slot, but proportionally.

Rotating pages

Before being placed in a slot, pages can be rotated. Rotation is always counter-clockwise, and allowed values are 0 (for no rotation, 90, 180 and 270 (careful, -90 is not allowed, use 270 instead).

Placement of pages in a slot

Sometimes, the content placed in a slot is not exactly the same size as the size of the slot itself. in that case, the placement property of the slot determines how content is placed. In all cases, a reference point of the content is placed exactly on the corresponding reference point of the slot. The following values are allowed:

Value	Meaning
LB	Align left bottom
LC	Align left top
LT	Align left top
CT	Align center top
RT	Align right top
RC	Align right center
RB	Align right bottom
CB	Align center bottom
CC	Align center center (the center of the content is aligned with the center of the slot)

Cropmark styles

Slots contain cropmark properties for the four corners of the slot. For each of these corners, it is possible to get all of the cropmark styles defined below.

N	none (default)
L	left
R	right
T	top
B	bottom
LT	left top
RT	right top
LB	left bottom
RB	right bottom

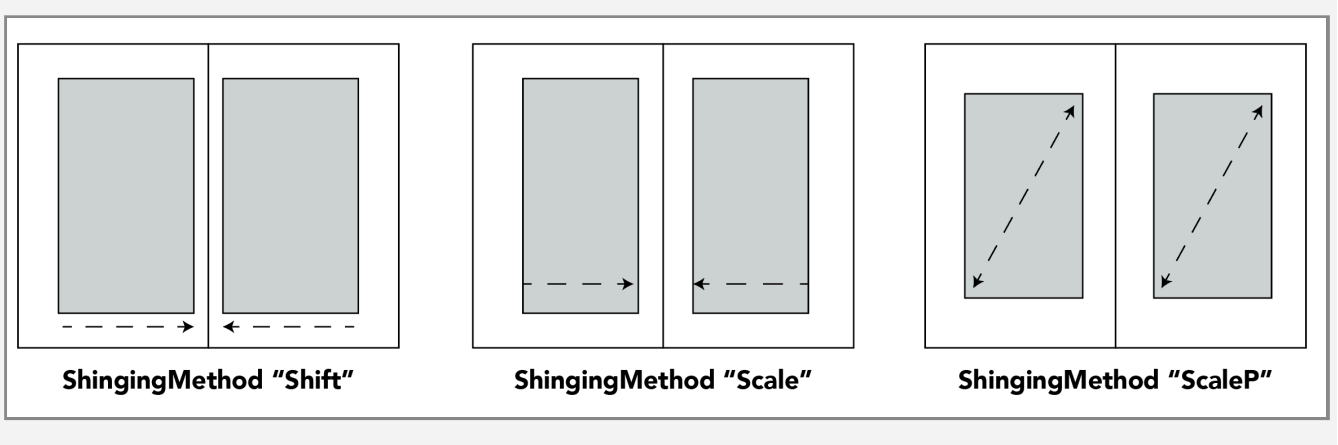
30.7 Imposition configuration terminology - shingling

ShinglingMethod

Shingling can be corrected in two ways: either by moving the content on a page or by scaling the page to receive a consistent outer margin of the pages.

The following methods are supported:

- **"Legacy" (not for JavaScript runlist):** Shingling by shifting the page contents inwards (ShinglingOffset < 0) or outwards (ShinglingOffset > 0)
- **"Shift":** Shingling by shifting the page contents inwards
- **"Scale":** Shingling by scaling the page contents anamorphic
- **"ScaleP":** Shingling by scaling the page contents proportional

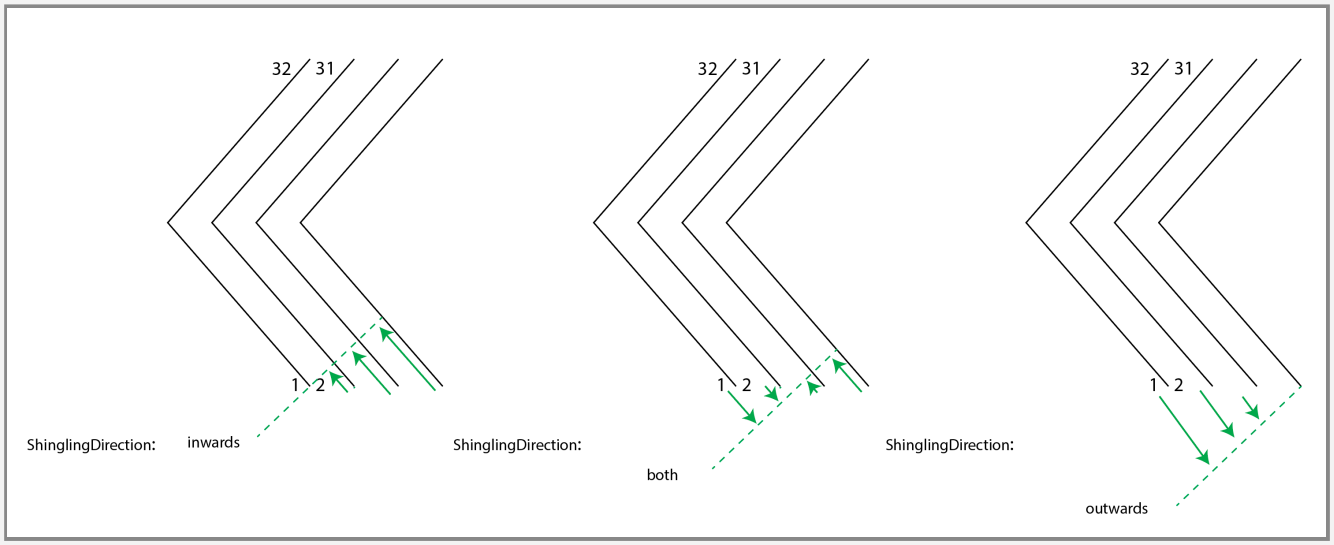


ShinglingDirection

The following directions are supported (no effect for the ShinglingMethod "Legacy"):

- **"inwards":** Shifts/scales all pages, except the outer page, towards the spine. The goal of this scaling is the leading edge of the outer pages.
- **"outwards":** Shifts/scales all pages, except the inner page. The goal of this scaling is the leading edge of the inner pages

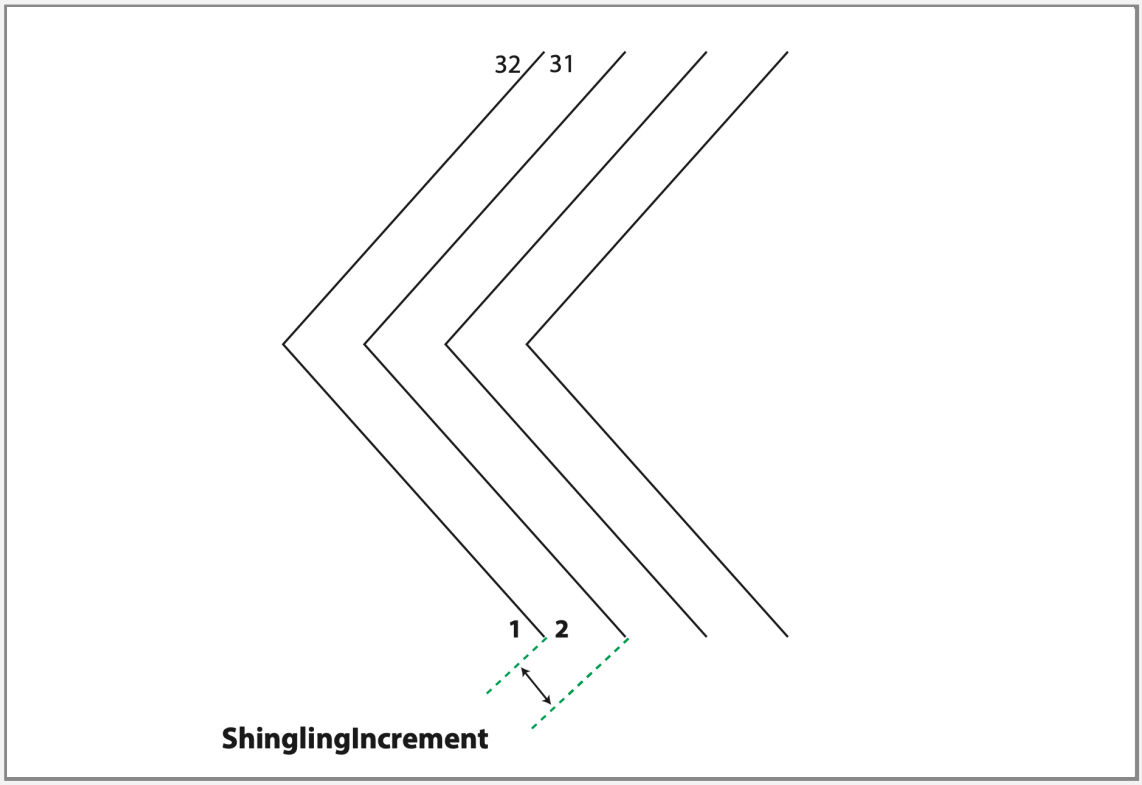
- **"both":** Shifts/scales inside pages toward the spine and outside pages toward the leading edge. The goal of this shingling is the final cut width of the product.



ShinglingIncrement

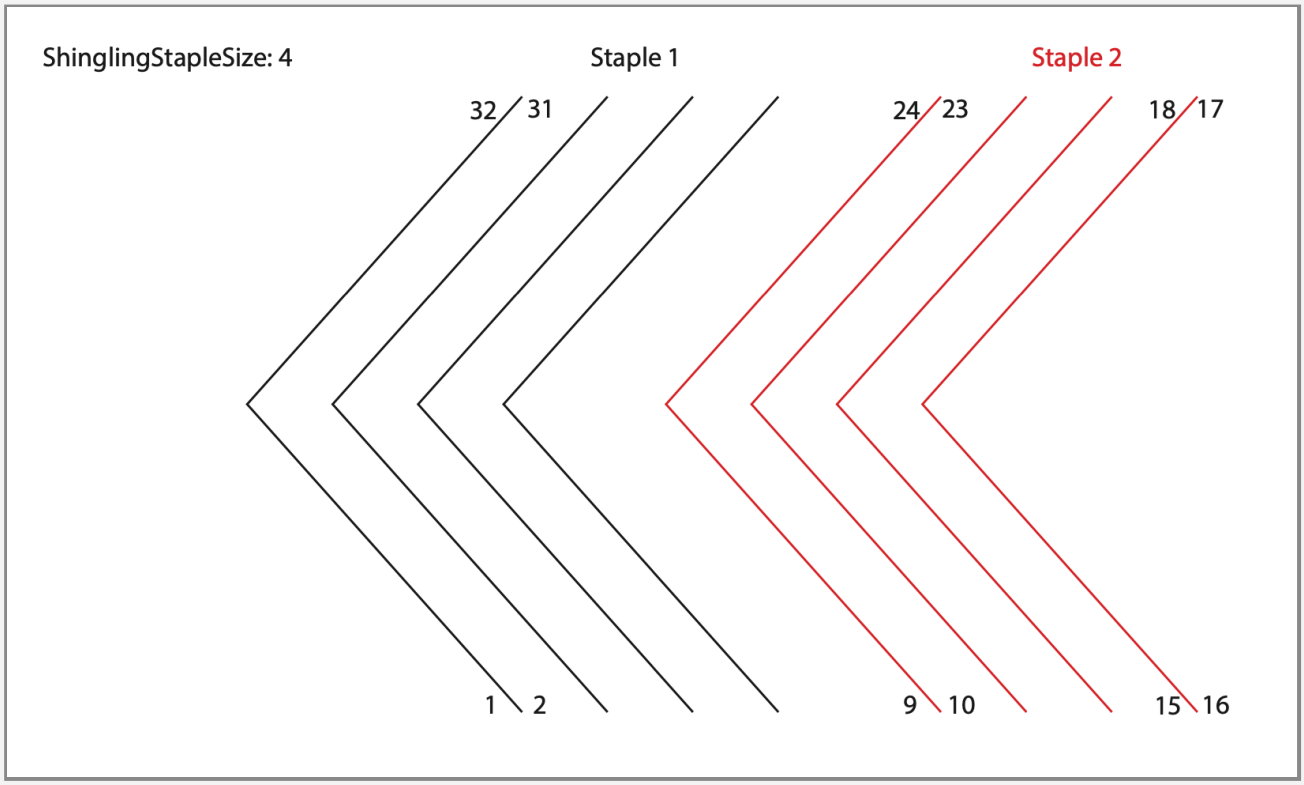
The shingling value is incremented by this value for each signature in a staple

(no effect for the ShinglingMethod "Legacy").



ShinglingStapleSize

Defines the number of signatures in a staple. In the example below, the ShinglingStapleSize would be 4.

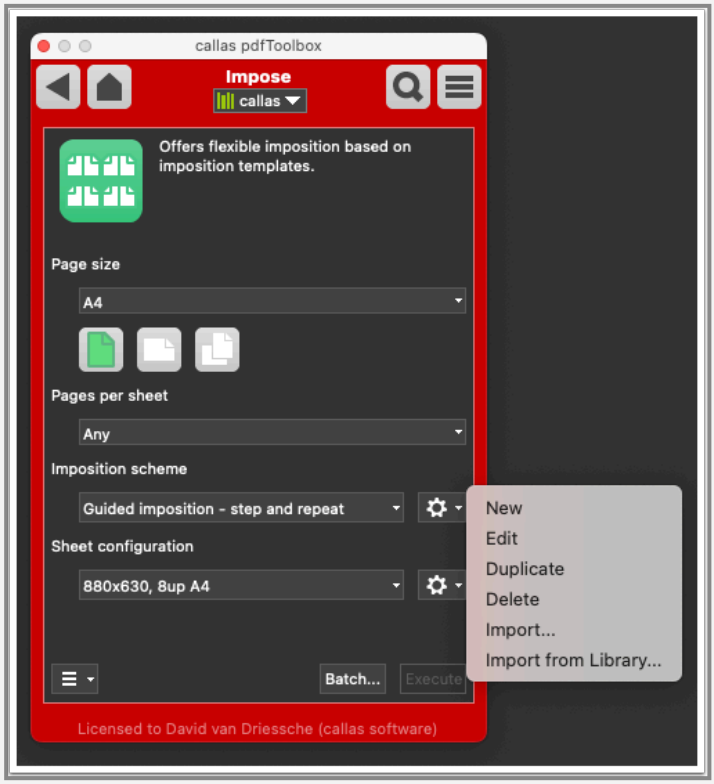

ShinglingStapleSize: 4
Staple 1
Staple 2
32
31
24
23
18
17
1
2
9
10
15
16

30.8 Imposition configuration editing and debugging

To edit imposition configurations, go to the "Impose" section of the Switchboard (as explained in this article). The sheet configurations and runlist can be found in the "Sheet configuration" and "Imposition scheme" menus.

Managing imposition configurations

The lists mentioned above contain all old-style and JavaScript configuration files. The configuration menu at the right of each menu allows managing these configuration files.

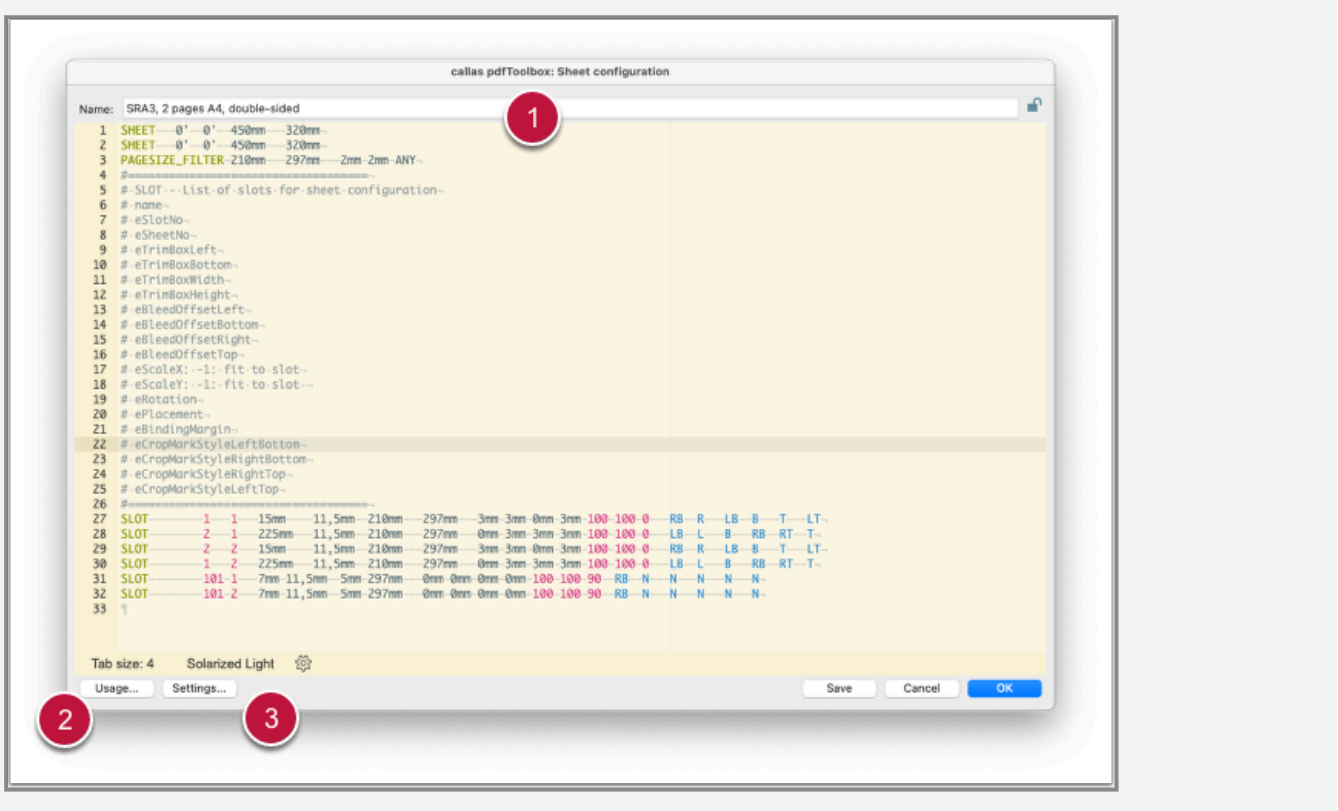


- "New", allows creation of a new runlist or sheet configuration. Both old-style and JavaScript configuration files can be created here; the choice is made in the editor that appears after "New" is clicked.
- "Edit", opens the selected configuration file in the relevant editor.

- "Duplicate", creates a copy of the selected configuration file and opens it in the relevant editor.
- "Delete", confirms and then deletes the selected configuration file.
- "Import...", imports a configuration file from disk.
- "Import from Library...", imports a configuration file from a different library. A dialog window appears where the library and then the configuration file from that library can be selected.

The sheet configuration editor

Select an existing sheet configuration, and use the "Edit" menu item to open it in the Sheet Configuration editor. If your library doesn't contain sheet configurations, you may need to switch to one of the standard libraries.



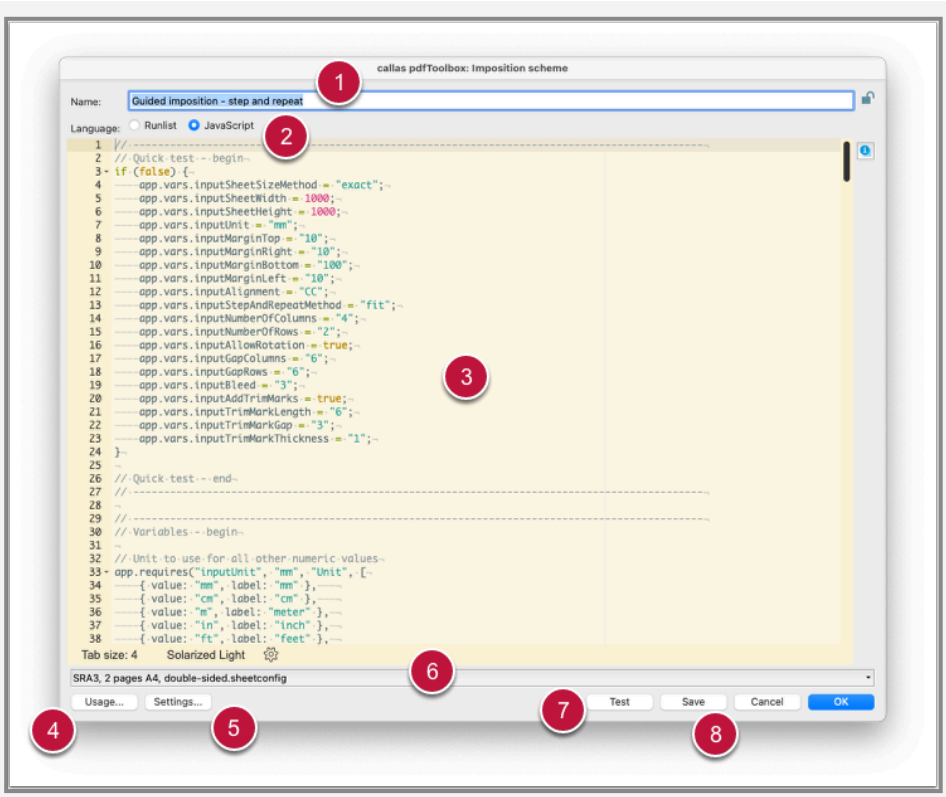
Sheet configurations always use old-style syntax, which is a tab-based textual format (not JavaScript). The details of the sheet configuration format are described here.

1. You can set or change the name of the sheet configuration.

2. The "Usage..." button brings up a dialog window that displays all actions that use this sheet configurations.
3. The "Settings..." button brings up a dialog that shows all sheet templates available in the current library and allows selecting a template.

The runlist editor

Select an existing runlist (from the menu under "Imposition scheme", and use the "Edit" menu item to open it in the runlist editor. If your library doesn't contain runlists, you may need to switch to one of the standard libraries.



The runlist editor uses either old-style syntax (further described here), or JavaScript (further described starting here). Additional features of the editor are explained in those chapters where relevant, but for both formats a few things can already be noted:

1. Give your runlist a clear name. This is used in the Switchboard menu, but also if you later use the runlist in an "Impose" action. And it is the name used in the file system for the ".runlist" file.

2. At the top of the editor, you can switch between the old-style and JavaScript format.
3. Regardless of the type of runlist, the editor supports syntax coloring, code completion, and error indications in case of problems with the runlist.
4. The "Usage..." button brings up a dialog window that displays all actions that use this runlist.
5. The "Settings..." button brings up a dialog window that shows additional filter settings. These determine when your runlist is shown in the Switchboard "Impose" panel (you can indicate that your runlist is only valid for certain sheet sizes or for a certain number of pages per imposed sheet).
6. The sheet configuration menu shows all sheet configurations in the current library. Selecting the proper sheet configuration is here only relevant if you are going to use test mode.
7. The "Test" button brings up the test mode dialog window, and runs your imposition. Test mode allows seeing the variables created during imposition, the imposed result etc...
8. The "Save" button saves your runlist to the current library without closing the editor.

Debugging JavaScript imposition scripts

Because JavaScript imposition configurations can become quite complex, there are tools available to make your life easier. To use them, enable the "Show evaluation results" checkbox.



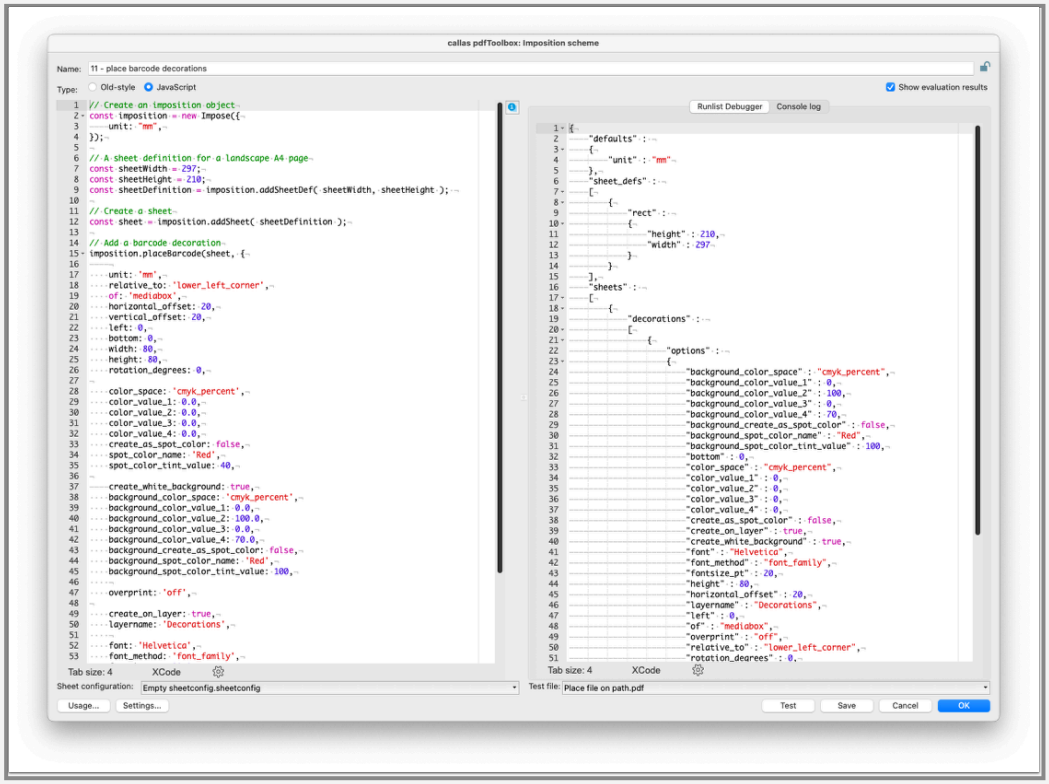
This functionality is only available for JavaScript imposition runlists, and only from pdfToolbox 16 onwards.

On the right hand side of the dialog window appear two new views:

- Runlist debugger
  The runlist debugger uses the sheet configuration and test file specified at the bottom of the dialog window, runs the imposition script, and then shows the created

sheet definitions, sheets, slots etc as JavaScript objects on the right hand side.

- Console log
  The console log tab shows the JavaScript console. To use it, in your script use one of the JavaScript console functions such as console.log or console.info (look here for console documentation).

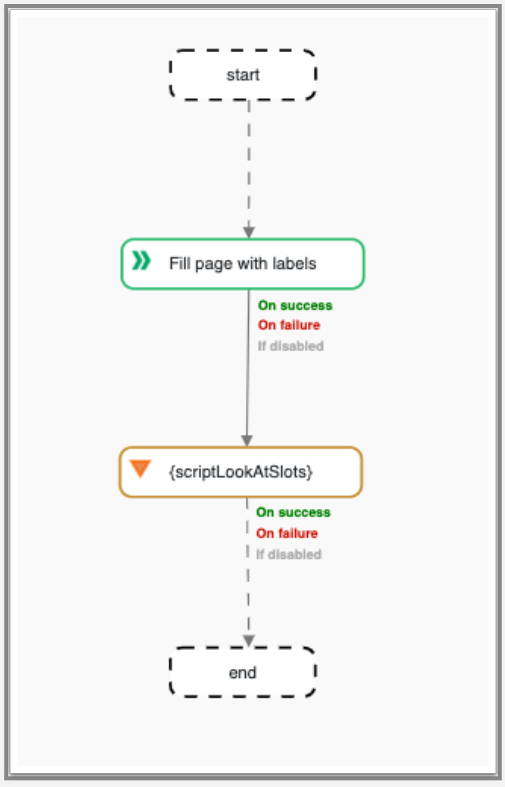


30.9 Getting imposed sheet information after imposition



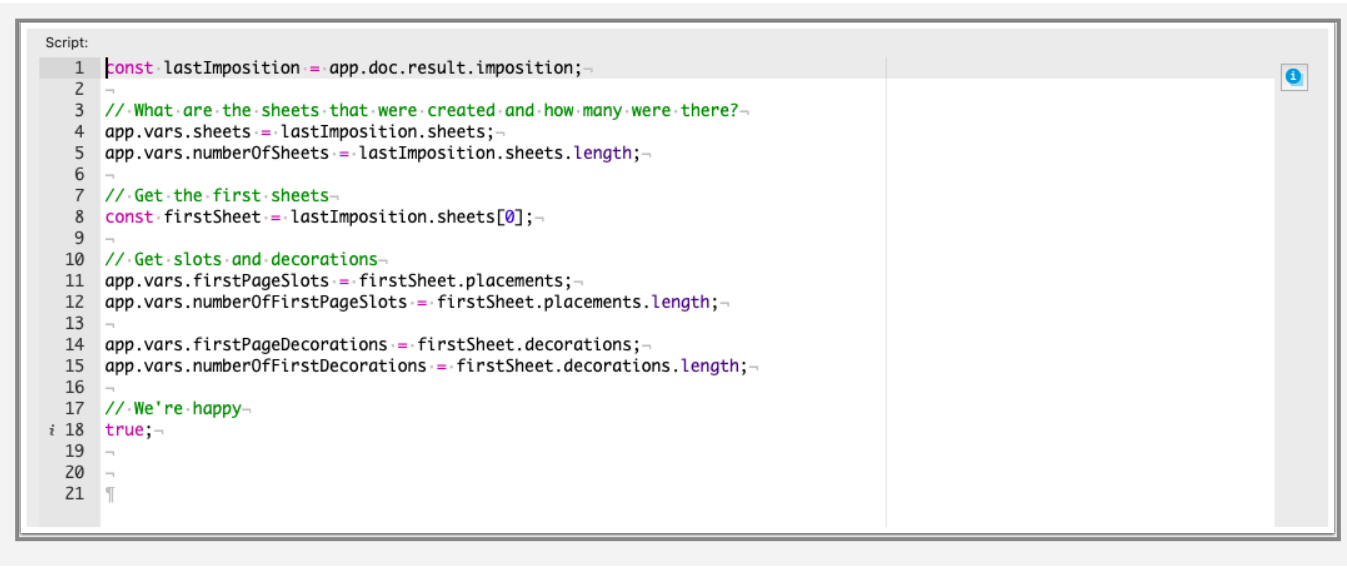
This functionality is available from pdfToolbox 16 onwards.

In a process plan, it is possible to retrieve the results of a previous imposition step using JavaScript. In order to do so, the imposition step needs to be followed by a JavaScript variable step, like so:



Where imposition results live in JavaScript

Once the imposition step has run, pdfToolbox stores the imposition results in the app.doc.result.imposition variable. Using JavaScript, you can use it as illustrated below.



This simple JavaScript access the results stored in:

```
app.doc.result.imposition
```

Click to copy

Write this object to the console to get a deeper understanding of what you can find in it. The structure is identical to what you can retrieve using the "Check evaluation results" checkbox in the JavaScript imposition editor for a JavaScript runlist configuration.

30.10 JavaScript imposition - getting started

To get started with the JavaScript imposition, the first step is to set everything up. This article documents how to create a JavaScript configuration setup, start editing the JavaScript runlist and put the basic JavaScript framework in place. The next article then shows how to build a simple but complete JavaScript configuration.

Setting up the environment

First of all, bring up the Switchboard, and locate the "Impose" action in there. You can do this either in an existing library, or in a new library.



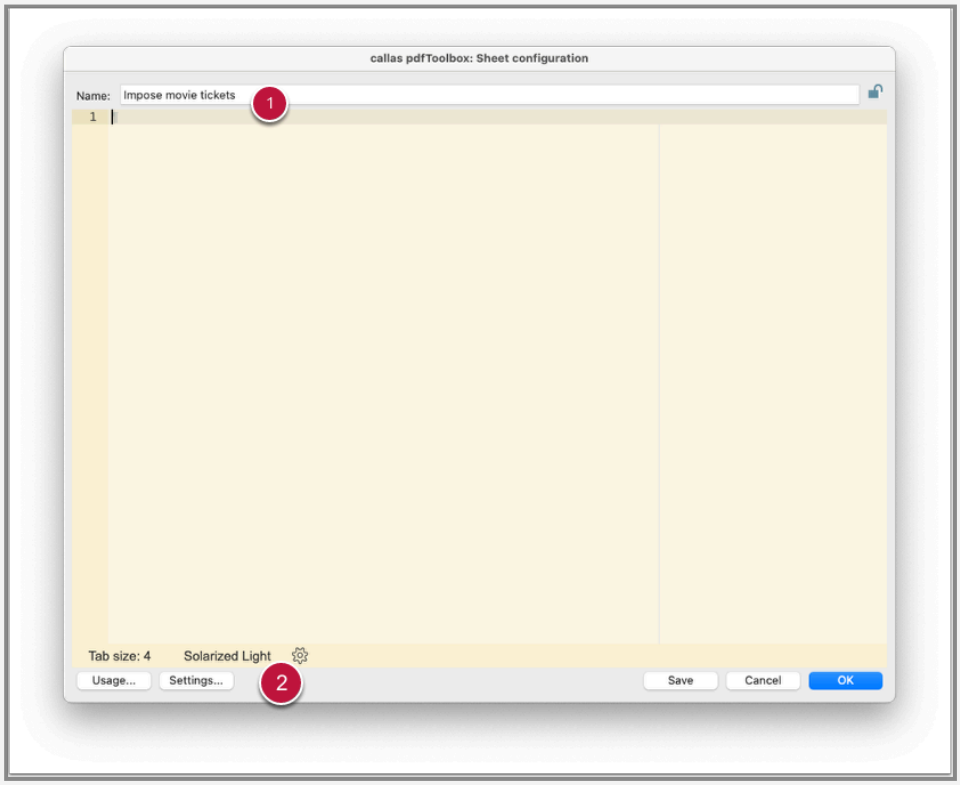
The "Impose" action can be found on the "Arrange" pane of the Switchboard.

Creating a sheet configuration file

Every JavaScript imposition configuration must have a sheet configuration file. In many cases this file will be empty, but it must still exist. If you want to, you can create one empty sheet configuration file and use it for many different runlist files.



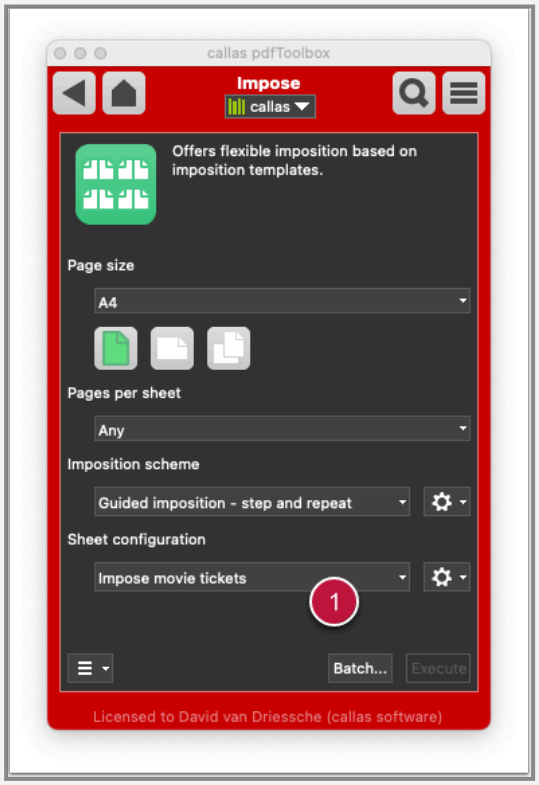
To create a sheet configuration file, click on the gear icon next to the "Sheet configuration" menu. In the menu that appears, select "New". This opens the sheet configuration editor.



The editor has no name yet and is empty. For the name, fill in: "Impose movie tickets". Leave the editor part empty as we're not going to use the sheet configuration file in this example.

The editor may look different in your case as it supports different color schemes. The name of the theme shown is "Solarized Light", but you can click on it to change it to other light or dark themes.

After you have given the sheet configuration its proper name, click on "OK".



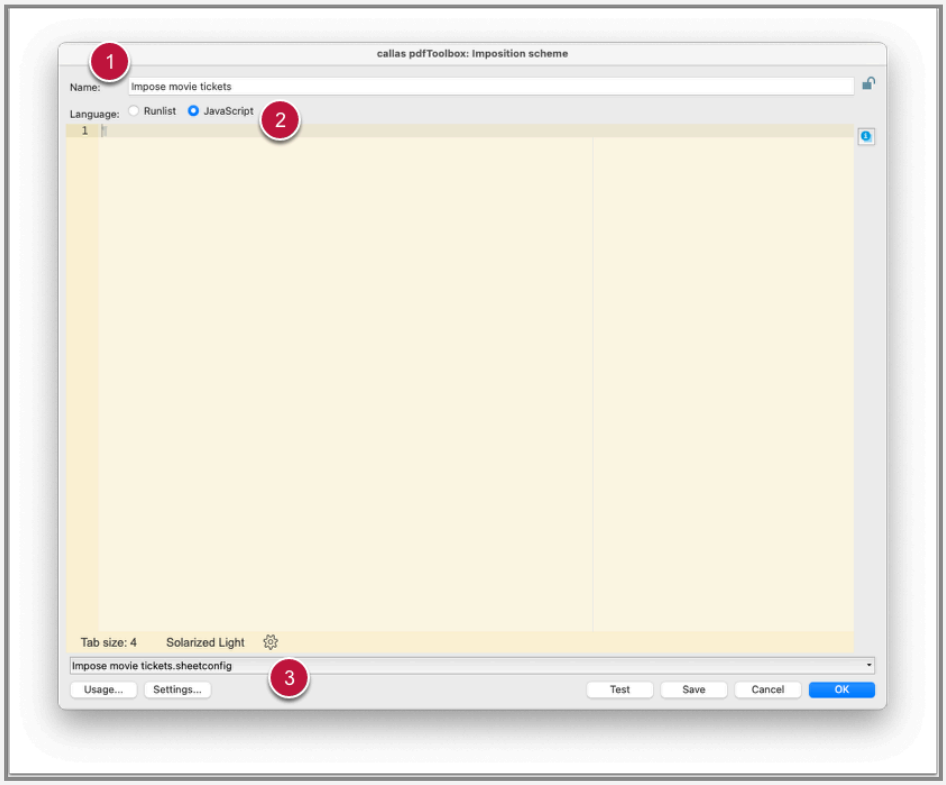
The "Impose movie tickets" sheet configuration is added to the library and is automatically selected in the sheet configuration menu in the Switchboard.

Creating a runlist file

Every JavaScript imposition configuration must have a runlist file. This is where the imposition logic is defined, but for JavaScript imposition configurations also often where sheets and slots are defined. In other words, for most JavaScript imposition configurations, everything can be found in this one file.



To create a runlist file, click on the gear icon next to the "Imposition scheme" menu. In the menu that appears, select "New". This opens the runlist editor.

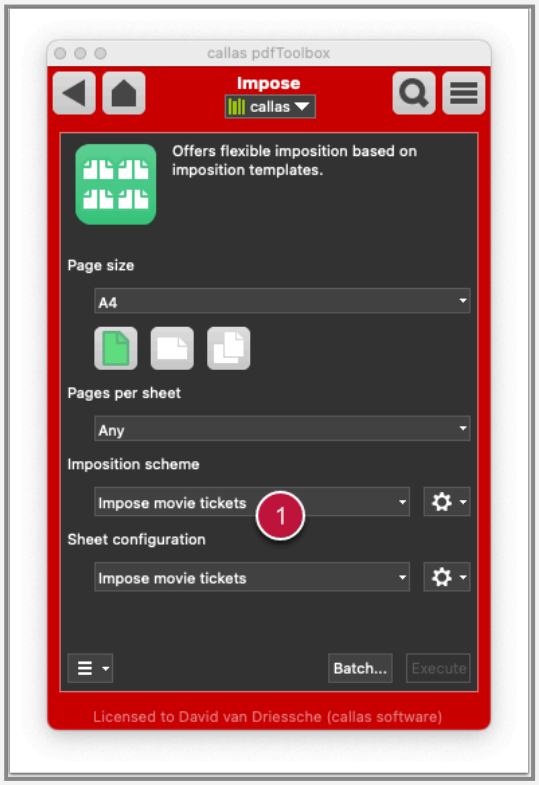


The editor has no name yet and is empty. For the name, fill in: "Impose movie tickets". Make sure that the "Language" radiobutton is set to "JavaScript" (it should, as this is the default in pdfToolbox).

From pdfToolbox 16, the editor looks slightly different and has more capabilities. The "Language" text referred to above is now called "Type".

Leave the editor part empty for now; we'll come back to editing it soon. The editor may look different in your case as it supports different color schemes. The name of the theme shown is "Solarized Light", but you can click on it to change it to other light or dark themes.

After you have given the runlist its proper name, click on "OK".



The "Impose movie tickets" runlist is added to the library and is automatically selected in the imposition scheme menu in the Switchboard.

The JavaScript runlist concept

Now that we have both configuration files needed for our JavaScript configuration, let's add some boilerplate code to our JavaScript runlist. This code will typically be the same (or very similar) for every JavaScript runlist you write. Click on the menu next to our runlist, and this time choose "Edit" to open up the runlist editor again.

Add the following code to the editor:

```
// Create an imposition object
const imposition = new Impose({
        unit: "mm",
});

// Imposition logic missing...
```

```
// return the imposition object to pdfToolbox
imposition;
```

Click to copy
This code:

- Creates a new JavaScript object of type "`Impose`" and assign it to a local variable ('`imposition`').
- Creates this object with a default value for the units used by the imposition, in this case millimeter.
- As the last line, 'returns' this local "`imposition`" variable to pdfToolbox so it has all information to perform the imposition.

Once you have entered this code, click "OK" to save it to the imposition runlist.



Do not attempt to run the imposition configuration on a PDF file at this time. Our code is not complete. If you would do so, pdfToolbox will return an error that says it can't find any sheets as a result of your imposition code.

30.11 JavaScript imposition - building an imposition configuration

The last article explored how to get started with a JavaScript runlist. We ended it with just boilerplate code to create a new Impose object and return it to pdfToolbox for processing. If you're following along, your code should now look like this:

```
// Create an imposition object
const imposition = new Impose({
        unit: "mm",
});

// Imposition logic missing...

// return the imposition object to pdfToolbox
imposition;
```

Click to copy

This article completes this into a working, albeit simple, imposition configuration. Several imposition concepts are explained along the way. In order to do this, we're using an example PDF file for a movie ticket, that needs to be printed on a larger sheet, essentially a 2-up configuration. To follow along, download this PDF file:



01 - Add QR code in film ticket.pdf

You can of course use other PDF files as well, but this article is going to assume the dimensions of the example file. It is 90 by 45 mm, with a bleed all around of 3mm.

Creating a sheet definition

In an imposition runlist, imposed sheets can only be created based on a sheet definition. So the first thing we need to create in our JavaScript is a proper sheet definition. The width and height of the sheets we want to create are:

```
const sheetWidth = 10 + 3 + 90 + 3 + 10;
```

```
const sheetHeight = 10 + 3 + 45 + 3 + 10 + 3 + 45 + 3 + 10;
```

Click to copy

This gives us enough space for two tickets (vertically), with 6 mm of bleed around each ticket, 10 mm of space outside of the tickets and 10 mm in between the two tickets. Creating the sheet definition is then easy:

```
const sheetDefinition = imposition.addSheetDef( sheetWidth, sheetHeight );
```

Click to copy

The "`Impose`" object in pdfToolbox has convenience methods to create sheet definitions, slots, sheets and more. As we already set the units we want to use when the Impose object was created, we can use the simple version of the "`addSheetDef`" function which only specifies the width and height of the sheet in the default unit.

Creating the slots

On this sheet, we need two slots as we want to place two tickets on each imposed sheet. To create these slots, we also have convenience functions. They take the ID of the sheet definition we just created as a parameter. The second parameter is the slot definition, which is a JavaScript object with properties for the slot. The definition of the first slot is shown in the code below, the second slot is identical, except for the bottom and left parameters.

```
const slot1 = imposition.addSlot( sheetDefinition, {
  "rect": { "bottom" : 10 + 3, "left": 10 + 3, "width": 90, "height": 45 },
  "bleed": { "bottom": 3, "left": 3, "top": 3, "right": 3 },
  "crop_marks" : { "left_bottom": "LB",
     "left_top": "LT",
     "right_bottom": "RB",
     "right_top": "RT",
     "length" : 5,
     "gap": 3 },
});
```

Click to copy

While slots can have many more properties, the ones in the code above are the important ones for this use case. Rota-

tion, scale, positioning, and other properties can be left on their default values.

Creating imposed sheets

Now that we have a sheet definition with proper slots defined, we can use it to create an imposed sheet:

```
const sheet = imposition.addSheet( sheetDefinition );
```

Click to copy

The only parameter is the sheet definition. On this imposed sheet, we now place the first page of the input document into both slots (in this example, we handle only the one-page input document case, of course a loop can be created to repeat this for all other pages as well).

```
imposition.addPage( sheet, slot1, 0 );
imposition.addPage( sheet, slot2, 0 );
```

Click to copy

The "`addPage`" convenience function, uses the ID of the sheet and of the slot. The last parameter is the index of the page that must be placed into the slot. In the example above, that's page "0" as page indexes here are zero-based.

After doing this, our code is complete and we can test it by clicking the "Test" button. The full JavaScript code can also be downloaded below for your reference.

runlist.js

30.12 JavaScript imposition - step by step examples

The rest of this chapter contains detailed documentation on the JavaScript objects and convenience functions available to create JavaScript configurations. Sometimes though, it is easier to learn from code examples, so this article lists specific examples to help you grasp different concepts.

The article comes with a downloadable pdfToolbox library. Import it into pdfToolbox Desktop by dragging it on top of the main pdfToolbox Desktop window. Then go to the Switchboard > Arrange > Impose, to see the JavaScript configurations in it. The library also contains an example Impose action so you can see how to include the imposition configurations into a process plan when necessary.



JavaScript imposition examples.kfpl

The rest of this article provides a summary of what each example shows, and any special remarks you should keep in mind about it.

01 - creating a single empty A4 sheet

This example shows how to setup a basic imposition configuration that uses millimeter as it's default measurement. It also creates a single A4 imposed sheet. As no slots are created for the sheet, it is a blank sheet.

02 - creating one blank A4 for each input page

The JavaScript imposition engine also gives you access to the JavaScript "`app`" object that pdfToolbox provides. This gives you access to simple variables, information about the current document and more. This example shows how to use `app.doc.numPages` to know how many pages the input PDF contains. The output of this imposition configuration consists of a PDF with blank A4 pages; as many of them as the number of pages in the input PDF file.

03 - placing each input page on a separate A4

In this example a "slot" is introduced. It is setup with a size that is the A4 output sheet minus a 10 mm border. It also uses some other properties to get specific behavior:

- crop marks all around the slot,
- scaling so that the placed PDF pages are scaled (up or down) to fit in the dimensions of the slot,
- placement in the center of the slot.

04 - asking for imposed sheet dimensions

As we have access to the `app` object, this example asks for two pieces of information from the user: the width and height of the imposed sheets that are going to be created.

To keep things neat, it's using `app.context.dialog.order` to make sure the information is asked in the correct order (width than height).

05 - demonstrating place text

This example shows how to add text to the imposed sheets. Different fonts, text sizes, and colors are illustrated.

06 - rotating slots

Content can be rotated in slots as it is placed; as demonstrated in this example. It shows the four different rotations supported. Because the slot used always has the same size, and the scaling is set to "fit slot proportionally", this configuration affects the size of the PDF pages placed in the slot if the placed pages are rectangular.

Placing decorations

Starting with pdfToolbox 16, the place content technology from pdfToolbox is available during the imposition process.

There are a number of examples showing how to use each individual decoration you can add to an imposed sheet:

- 07, place text text decorations
- 08, place line decorations
- 09, place ellipse decorations
- 10, place rectangle decorations
- 11, place barcode decorations

Of course multiple decorations can be added in a single imposition configuration. It suffices to combine the code from the different examples to do so.

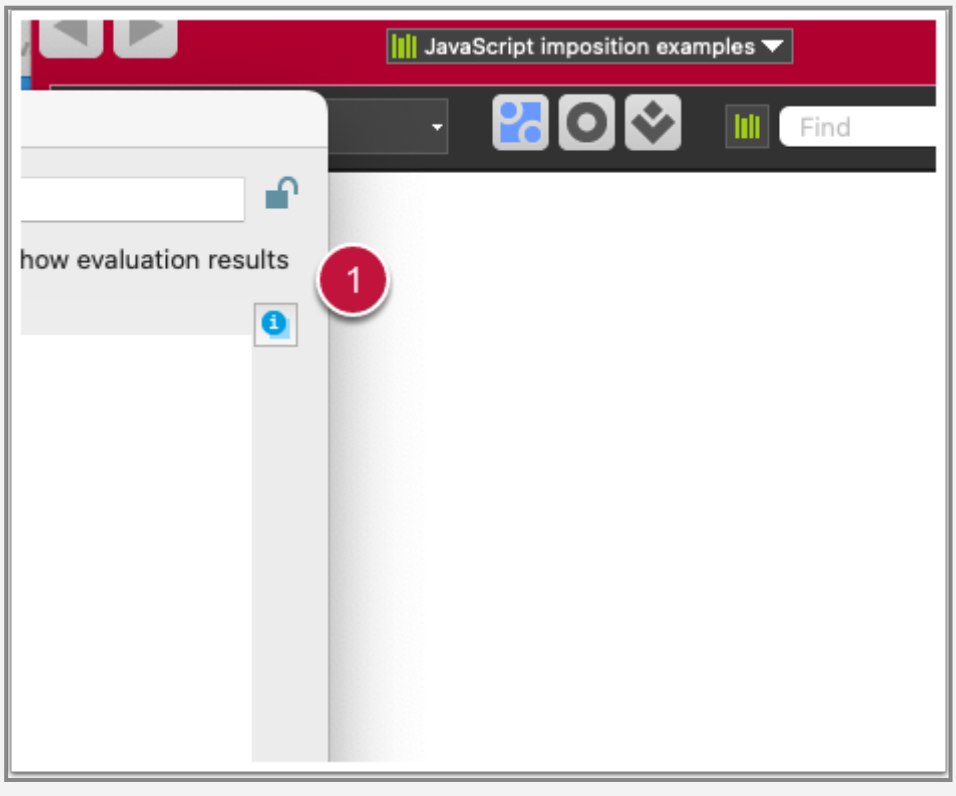
30.13 JavaScript imposition - inserting imposition snippets

While typing JavaScript after reading all the documentation can certainly be fun, often it's easier to start with snippets of code and modify them as required.

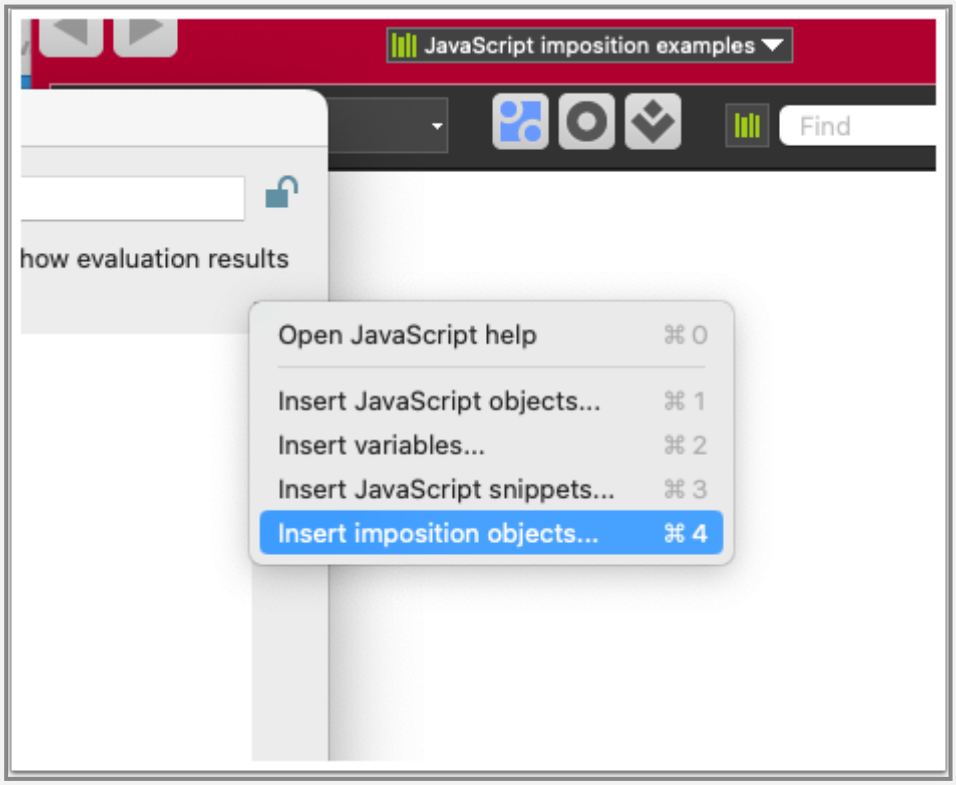
⚠ This functionality is available in pdfToolbox 16 and later.

Inserting a JavaScript imposition snippet

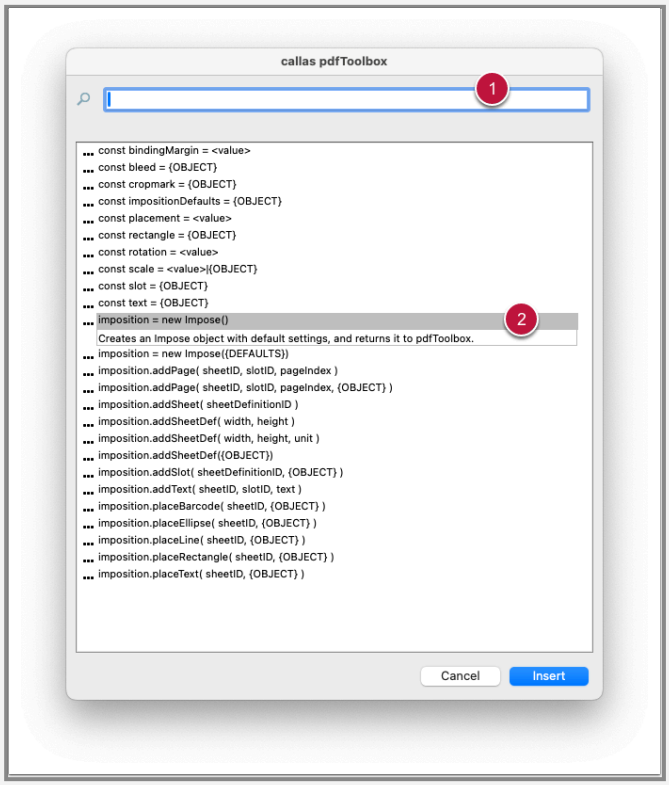
To get the list of snippets, click on the "i" button.



The "i" button located at the top right of the JavaScript imposition editor.



In the menu that appears, select the "Insert imposition objects..." option. The dialog with snippets appears.



In this dialog, you can select the snippet you want from the list (2), or type into the search field (1) to find it.

Tips & tricks

The snippets that are inserted can be quite verbose, and they often are not valid JavaScript code without some modifications. For example, this is the code inserted for the addPage snippet:

```
// Add a page from the input document into a specific slot on an imposed sheet.
// 'sheetID' should be the result of a call to imposition.addSheet().
// 'slotID' should be the result of a call to imposition.addSlot().
// The page index is 0 - based.
imposition.addPage(sheetID, slotID, pageIndex);
```

Click to copy

To make this code work, you need to have variables called "sheetID", "slotID", and "pageIndex" or you have to edit the invocation to "addPage" to use your variables. That being said, the snippet does show you the correct call signature and explains where these arguments should come from.

30.14 JavaScript imposition - convenience functions

In order to use the convenience functions, you must first create an instance of the "Impose" class. This article defines this "Impose" class.

Constructor

```
const imposition = new Impose();
const imposition = new Impose( defaults );
```

Click to copy

Without defaults argument, the constructor creates a new Impose object with completely default settings. Specifying the defaults parameter allows overriding specific defaults for the imposition process.

If present, "defaults" must be an object with the following properties (which are all optional):

- "unit": the default unit for the imposition configuration. Defaults to "pt" for point.
- "slots": an array with slot definitions.
- "shingling": default settings for automatic shingling.

addSheetDef

```
const sheetDefinitionID = imposition.addSheetDef( sheetDefinition );
const sheetDefinitionID = imposition.addSheetDef( width, height );
const sheetDefinitionID = imposition.addSheetDef( width, height, unit );
```

Click to copy

Creates a new sheet definition (not an imposed sheet, the definition from which you can later create an imposed sheet). The sheet definition can be created by passing a width and height (with an optional unit, if not passed it reverts to the default unit for this imposition). It can also be created from a sheet definition object.

The return value is the ID of the newly created sheet definition (which can then be used to create slots or actual imposed sheets).

addSheet

```
const sheetID = imposition.addSheet( sheetDefinitionID );
```

Click to copy
Creates an imposed sheet from the given sheet definition ID.

The return value is the ID of the newly created sheet. After creating the sheet, content can be added to it by using addPage or addText.

addSlot

```
const slotID = imposition.addSlot( sheetDefinitionID, slotDefinition );
```

Click to copy
Creates a new slot for the sheet definition with the given ID. The slot is configured using the properties in the slot definition object.

The return value is the ID of the newly created slot, which can be used to add content to slots on specific sheets.

addPage

```
imposition.addPage( sheetID, slotID, pageIndex );
imposition.addPage( sheetID, slotID, pageIndex, options );
```

Click to copy
Places the page with (zero-based) index from the document we're imposing into the slot with ID "slotID" on the sheet with ID "sheetID". Optionally an options object is provided with the following properties:

- shingling: a value that overrides the automatically generated shingling value for this page.

There is no return value.

addText

```
imposition.addText( sheetID, slotID, text );
```

Click to copy
Places the given "text" into the slot with ID "slotID" on the sheet with ID "sheetID".

There is no return value.

30.15 JavaScript imposition - object definitions

In the convenience functions, or when creating an imposition runlist using nothing but JavaScript objects directly, the following objects can be used.

The "sheet_defs" array

A collection of all defined sheets and its slots, either from the sheet config or dynamic sheets. The `"sheet_defs"` array can contain sheet definition objects. A sheet definition object can contain the following properties:

"id"

- `"id"`: String ID, can be used to reference a sheet definition by name. This property is required.

"rect"

- `"rect"`: A rectangle that defines the size of the sheet. See "rect" object.

trim_box

- `"trim_box"`: A trimbox object that defines the trimbox to be defined on the imposed sheets.

"slots"

- `"slots"` is an array of slot objects.

```
"sheet_defs":
  [
    {
      "id": "",
```

```
      "rect": {  },
      "slots":
            [
              {  }
            ]
    }
  ],
```

Click to copy

The "sheets" array

A collection of all generated sheets including their placements. The `"sheets"` array contains sheet objects. A sheet object can contain the following properties:

"sheet_def"

- `"sheet_def"`: Index or string id of reference sheet definition.

"placements"

- `"placements"`: An array of "placements" objects.

```
Example

"sheets":
  [
    {
      "sheet_def": 0,
      "placements":
      [
        {  }

      ]
    }
  ]
```

Click to copy

The "defaults" object

The "defaults" object is optional. If certain properties are defined here, they will be used if no specifications have been made in the "sheet_defs" array. The `"defaults"` object can contain the following objects:

"unit"

- The default unit for the imposition configuration. All possible `"unit"` values are listed here. Default: "pt".

"slots"

- An array with slot definitions. All `"slots"` properties are listed here.

"shingling"

- Default settings for **automatic** shingling. Depending on the values specified in the shingling object the engine calculates the value for the shingling automatically. The `"shingling"` object can contain the following properties:
- `"unit"`: see Unit. Default: inherited, "pt"
- `"method"`: String, one of `"shift"`, `"scale"`, `"scale_proportional"`. Default: "scale".
- `"direction"`: String, one of `"inwards"`, `"outwards"`, `"both"`. Default: "inwards"
- `"increment"`: Number. Default: 0
- `"book_size"`: Integer. Default: 0
- `"staple_size"`: Integer. Default: 0

Note: A more detailed explanation of the shingling parameters can be found in this article.

```
"defaults":
  {
    "unit": "mm",
```

```
    "slots":
        { },
    "shingling": {
        "unit" : "mm",
        "method" : "scale",
        "direction" : "inwards",
        "increment" : 0.12,
        "book_size" : 16,
        "staple_size" : 4
        }
  },
```

Click to copy

Common objects and strings

"rect" object

Definition of a rectangle. It can be part of the "sheet_defs" array and of a slot object. The `"rect"` object contains the following properties:

- `"unit"` : see Unit. The unit for all other measurements. Optional. Default: inherited, "pt".
- `"left"` : Number. The left hand side of the rectangle. Default: 0.
- `"bottom"` : Number. The bottom side of the rectangle. Default: 0.
- `"right"` : Number. The right hand side of the rectangle. Default: 0.
- `"top"` : Number. The top side of the rectangle. Default: 0.
- `"width"` : Number. Optional. Overwrites `"right"` with `"left + width"` if present.
- `"height"` : Number. Optional. Overwrites `"top"` with `"bottom + height"` if present.

```
Example

"rect" :
  {
```

```
    "bottom" : 28.5,
    "height" : 114,
    "left" : 552.5,
    "width" : 171
  },
```

Click to copy

"trim_box" object

Definition of an optional rectangle. It can be part of the "sheet_defs" array only. The `"trim_box"` object contains the following properties:

- `"unit"` : see Unit. The unit for all other measurements. Optional. Default: inherited, "pt".
- `"left"` : Number. The left hand side of the rectangle. Default: 0.
- `"bottom"` : Number. The bottom side of the rectangle. Default: 0.
- `"right"` : Number. The right hand side of the rectangle. Default: 0.
- `"top"` : Number. The top side of the rectangle. Default: 0.
- `"width"` : Number. Optional. Overwrites `"right"` with `"left + width"` if present.
- `"height"` : Number. Optional. Overwrites `"top"` with `"bottom + height"` if present.

```
Example

"trim_box" :
  {
    "left":3,
    "bottom":3,
    "width": 297,
    "height": 210
  },
```

Click to copy

"crop_marks" object

The `"crop_marks"` object can be part of a slot object. It can contain the following properties:

- `"unit"` : see Unit. Default: inherited, "pt"
- `"left_bottom"` : see Crop Mark Direction. Default: "N".
- `"left_top"` : see Crop Mark Direction. Default: "N".
- `"right_bottom"` : see Crop Mark Direction. Default: "N".
- `"right_top"` : see Crop Mark Direction. Default: "N".
- `"length"` : Number. Default: 10
- `"gap"` : Number. Default: 3
- `"width"` : Number. Default: 1
- `"color_space"` : see Color Space. Default: "ALL"
- `"spot_name"` : Name of the spot color
- `"color_components"` : Array of doubles (in percent). Default: [0,0,0,100]
- `"tint_value"` : double [0,100]. Default: 100

Example

```
"crop_marks":
  {
    "unit": "pt"
    "left_bottom": "N",
    "right_bottom": "N",
    "right_top": "N",
    "left_top": "N",
    "length": 10,
    "width": 0.04,
    "gap": 2,
    "color_space": "SPOT",
    "spot_name": "Green",
    "color_components": [100,0,100,0],
    "tint_value": 100
  },
```

Click to copy

"text" object

The `"Text"` object can be part of a slot object. It can contain the following properties:

- `"unit"` : see Unit. Default: inherited, “pt”
- `"font"` : String. Default: "Helvetica"
- `"size"` : Number. Default: "12"
- `"color_space"` : see Color Space. Default: "ALL"
- `"color_components"` : Array of doubles (in percent). Default: [0,0,0,100]
- `"spot_name"` : Name of the spot color (if "color_space": "SPOT")
- `"tint_value"` : double [0,100]. Default: 100

```
Example

"text" :
  {
    "color_components" : [0,0,0,100],
    "color_space" : "ALL",
    "font" : "Helvetica",
    "size" : 8,
    "unit" : "pt"
  },
```

Click to copy

"bleed" object

The definition of where (and how much) bleed is required for a Rectangle. The `"bleed"` object can be part of a slot object. It can contain the following properties:

- `"unit"` : see Unit. The unit for all other measurements. Optional. Default: inherited, "pt".
- `"left"` : Number. Bleed required on the left. Default: 0.
- `"bottom"` : Number. Bleed required at the bottom. Default: 0.
- `"right"` : Number. Bleed required on the right. Default: 0.
- `"top"` : Number. Bleed required at the top. Default: 0.

```
Example

"bleed" :
  {
    "bottom" : 2.5,
    "left" : 2.5,
    "right" : 2.5,
    "top" : 2.5
  },
```

Click to copy

"clip_mode" string

Defines how imposed pages are clipped. The `"clip_mode"` string can be part of a slot object. It can contain the following values:

- `"S"`: Slot. Imposed pages are clipped at the slot boundaries
- `"P"`: Positioned page. Imposed pages are clipped according to their TrimBox plus bleed as set up in the slot. Default: "P"

```
Example

"clip_mode": "P",
```

Click to copy

"rotation" string

Page rotates anti-clockwise (value, basic unit of measurement: degrees). The `"rotation"` string can be part of a slot object. It can have the following values: `0`, `90`, `180`, `270`

```
Example

"rotation" : 270,
```

Click to copy

"scale" object

The `"scale"` object can be part of a slot object. It can be a number or it can contain the following properties:

- `"horizontal"`: Double
- `"vertical"`: Double

```
Example 1

"scale": {"horizontal": -1, "vertical":-1 },
// the page is scaled proportionally to fit the slot
// default if no "scale" object is set

Example 2

"scale": 25,
// affects the corresponding scaling

Example 3

"scale": 100,
// no scaling

Example 4

"scale" : 0,
// the page is scaled unproportionally to fill the slot
```

Click to copy

"binding_margin" string

Binding margin defines the edge on which the creep should be equalized. It is required if shingling shall be applied to a slot and must be set to a value other than "N" in this case. The `"binding_margin"` string can be part of a slot object. It can have the following values:

- `"L"`: Left
- `"R"`: Right
- `"T"`: Top

- `"B"` : Bottom
- `"N"` : None

```
Example

"binding_margin": "N",
```

Click to copy

"placement" string

Direction of the margin. The `"placement"` string can be part of a slot object. It can have the following values:

- `"LB"` : align left bottom
- `"LC"` : align left center
- `"LT"` : align left top
- `"CT"` : align center top
- `"RT"` : align right top
- `"RC"` : align right center
- `"RB"` : align right bottom
- `"CB"` : align center bottom
- `"CC"` : align center, i.e. both vertically and horizontally

```
Example

"placement": "CC",
```

Click to copy

Slot object

A slot object is used in the "sheet_defs" array and in the "defaults" object (optional). It can contain the following properties:

- `"id"` : String.
- `"isolated"` : Boolean
- `"rect"` : see "rect"
- `"crop_marks"` : see "crop_marks"
- `"text"` : see "text"
- `"bleed"` : see "bleed"

- `"clip_mode"` : see "clip_mode"
- `"rotation"` : see "rotation"
- `"scale"` : see "scale"
- `"binding_margin"` : see "binding_margin"
- `"bleed_shingling"` : Boolean
- `"placement"` : see "placement"

```
Example

{
  "id": "left",
  "isolated": true,
  "rect" : {
     "bottom" : 274,
     "height" : 226,
     "left" : 5.2999999999999998,
     "width" : 5
     },
  "bleed": {
     "left": 0,
     "bottom": 0,
     "right": 0,
     "top": 0
     },
  "crop_marks" : {
     "color_components" : [ 100, 0, 100, 0 ],
     "color_space" : "SPOT",
     "gap" : 2,
     "left_bottom" : "B",
     "left_top" : "N",
     "length" : 5,
     "right_bottom" : "RB",
     "right_top" : "R",
     "tint_value" : 100,
     "unit" : "mm"
     },
  "text" :
     {
     "font" : "Helvetica",
     "size" : 8,
     "unit" : "pt"
     },
  "scale" : 100,
```

```
  "rotation": 0,
  "clip_mode": "P",
  "placement": "CC",
  "bleed_shingling": false,
  "binding_margin": "N"
},
```

Click to copy

Placements object

`"placements"` is an array of placements objects. It is used in the "sheets" array. A placements object can contain the following properties:

- `"slot"` : String or number. Id or index of slot to be used for placement.
- `"page"` : Number, zero based. Page index to be placed in slot.
- `"text"` : A text to be placed in the slot. Sring.
- `"shingling"` : see Shingling. If a `"value"` property is defined automatic shingling is disabled and value is used directly.

```
Example

{
  "slot": "middle",
  "page": 0,
  "text": "This is text",
  "shingling":
    {
      "unit": "mm",
      "value": 5
    }
},
```

Click to copy

Shingling object

The `"shingling"` object can be part of the "placements" object. It can contain the following properties:

- `"unit"`: see Unit. Default: inherited, "pt"
- `"value"`: Number. If a value is defined, the automatic shingling is disabled. Default: 0

```
Example

"shingling":
  {
    "unit": "mm",
    "value": 5
  }
```

Click to copy

Possible values

Unit

A string used to identify the unit of a number:

- one of: `"mm"`, `"cm"`, `"m"`, `"in"`, `"ft"`, `"pt"` or `"pc"`

Crop Mark Directions

- String. One of:
  - `"N"`: None
  - `"L"`: Left
  - `"R"`: Right
  - `"T"`: Top
  - `"B"`: Bottom
  - `"LT"`: Left and Top
  - `"LB"`: Left and Bottom
  - `"RT"`: Right and Top

- `"RB"`: Right and Bottom

Color Space

- String. One of: `"ALL"`, `"CMYK"`, `"RGB"`, `"GRAY"`, `"SPOT"`

30.16 JavaScript imposition - using Place Content during imposition

⚠ Using Place Content functionality during imposition is only available for pdfToolbox 16 or later.

Next to using the core imposition functions that place pages or text into slots on an imposed sheet, pdfToolbox also supports using Place Content functionality during the imposition process. This functionality does **not** use the slot system (in other words, you don't have to create slots to add content to the imposed sheet), but it does come with convenience functions. The functionality available during imposition is very similar to the respective place content fixups. To play with the possibilities of the functionality, using those fixups is a good idea. This article describes all place content functionality available.

Common aspects

All of the place content functions follow the same call signature:

```
const imposition = new Impose();
imposition.placeText( sheetID, {...});
```

Click to copy

The convenience functions have to be called on an "Impose" object, just as the other convenience functions. The all take a "sheet ID" as their first parameter. This typically is the result of a call to the addSheet convenience function. The second parameter is a JavaScript object containing all properties for the object that is to be created.

Because these functions don't use the slot system, the properties for the object include where the object should be positioned on the imposed sheet.

Place text

This function places text in a specific area of a sheet. While there is some similarity to the "addText" convenience function that uses the slot system, this function is based on place content and thus supports more capabilities (such as HTML formatting).

```
imposition.placeText( sheetID, {
        color_space: 'cmyk_percent',
        color_value_1: 0.0,
        color_value_2: 0.0,
        color_value_3: 0.0,
        color_value_4: 100.0,
        create_as_spot_color: false,
        create_on_layer: false,
        create_white_background: false,
        font: 'Herculanum, Arial',
        font_method: 'font_family',
        fontsize_pt: 24.0,
        horizontal_offset: 10.0,
        layername: '',
        of: 'cropbox',
        overprint: 'off',
        relative_to: 'lower_left_corner',
        rotation_degrees: 0.0,
        spot_color_name: '',
        spot_color_tint_value: 100,
        text: '<i>hello</i> <b>world</b>',
        unit: 'mm',    vertical_offset: 10.0,
});
```

Click to copy

Place line

This function places a line in a specific area of a sheet. Check out the "Place line" fixup to understand the properties.

```
imposition.placeLine( sheetID, {
        blend_mode: 'normal',
```

```
        create_as_spot_color: false,
        create_objects_behind_current_page_content: false,
        create_on_layer: '',
        endpoint_x: 100.0,
        endpoint_y: 100.0,
        of: 'cropbox',
        relative_to: 'lower_left_corner',
        set_color_space_for_stroke: 'cmyk_percent',
        set_line_cap_style: 'butt_end_caps',
        set_line_dash_pattern: '',
        set_line_join_style: 'miter_joins',
        set_line_width: 5.0,
        set_opacity_for_stroke: 100,
        set_overprint_for_stroke: 'off',
        set_rendering_intent: 'relativecolorimetric',
        source_color_value_1: 0.0,
        source_color_value_2: 0.0,
        source_color_value_3: 0.0,
        source_color_value_4: 100.0,
        spot_color_name: '',
        spot_color_tint_value: 100,
        startpoint_x: 10.0,
        startpoint_y: 10.0,
        unit: 'mm',
});
```

Click to copy

Place rectangle

This function places a rectangle in a specific area of a sheet.
Check out the "Place rectangle" fixup to understand the properties.

```
imposition.placeRectangle( sheetID, {
        blend_mode: 'normal',
        create_objects_behind_current_page_content: false,
        create_on_layer: '',
        file_spot_color_name: '',
        file_spot_color_tint_value: 100,
        fill_create_as_spot_color: false,
        fill_source_color_value_1: 0.0,
```

```
        fill_source_color_value_2: 0.0,
        fill_source_color_value_3: 0.0,
        fill_source_color_value_4: 0.0,
        height: 50,
        of: 'cropbox',
        position_of_center_x: 60,
        position_of_center_y: 60,
        relative_to: 'lower_left_corner',
        rotation_degrees: 0.0,
        rounded_corner: 0.0,
        set_color_space_for_fill: 'none',
        set_color_space_for_stroke: 'cmyk_percent',
        set_line_dash_pattern: '',
        set_line_width: 1.0,
        set_opacity_for_fill: 100,
        set_opacity_for_stroke: 100,
        set_overprint_for_fill: 'off',
        set_overprint_for_stroke: 'off',
        set_rendering_intent: 'relativecolorimetric',
        stroke_create_as_spot_color: false,
        stroke_source_color_value_1: 0.0,
        stroke_source_color_value_2: 0.0,
        stroke_source_color_value_3: 0.0,
        stroke_source_color_value_4: 100.0,
        stroke_spot_color_name: '',
        stroke_spot_color_tint_value: 100,
        unit: 'mm',
        width: 50,
});
```

Click to copy

Place ellipse

This function places a circle / ellipse in a specific area of a sheet. Check out the "Place circle/ellipse" fixup to understand the properties.

```
imposition.placeEllipse( sheetID, {
        blend_mode: 'normal',
        create_objects_behind_current_page_content: false,
        create_on_layer: '',
```

```
        fill_create_as_spot_color: false,
        fill_source_color_value_1: 0.0,
        fill_source_color_value_2: 0.0,
        fill_source_color_value_3: 0.0,
        fill_source_color_value_4: 0.0,
        fill_spot_color_name: '',
        fill_spot_color_tint_value: 100,
        horizontal_diameter: 50,
        of: 'cropbox',
        position_of_center_x: 60,
        position_of_center_y: 60,
        relative_to: 'lower_left_corner',
        rotation_degrees: 0,
        set_color_space_for_fill: 'cmyk_percent',
        set_color_space_for_stroke: 'none',
        set_line_dash_pattern: '',
        set_line_width: 5.0,
        set_opacity_for_fill: 50,
        set_opacity_for_stroke: 100,
        set_overprint_for_fill: 'off',
        set_overprint_for_stroke: 'off',
        set_rendering_intent: 'relativecolorimetric',
        stroke_create_as_spot_color: false,
        stroke_source_color_value_1: 0.0,
        stroke_source_color_value_2: 0.0,
        stroke_source_color_value_3: 0.0,
        stroke_source_color_value_4: 100.0,
        stroke_spot_color_name: '',
        stroke_spot_color_tint_value: 100,
        unit: 'mm',
        vertical_diameter: 40,
});
```

Click to copy

Place barcode

This function places a barcode in a specific area of a sheet. Check out the "Place barcode" fixup to understand the properties.

```
imposition.placeBarcode( sheetID, {
```

```
        background_color_space: 'cmyk_percent',
        background_color_value_1: 0.0,
        background_color_value_2: 0.0,
        background_color_value_3: 0.0,
        background_color_value_4: 0.0,
        background_create_as_spot_color: false,
        background_spot_color_name: '',
        background_spot_color_tint_value: 100,
        bottom: 0.0,
        color_space: 'cmyk_percent',
        color_value_1: 0.0,
        color_value_2: 0.0,
        color_value_3: 0.0,
        color_value_4: 100.0,
        create_as_spot_color: false,
        create_on_layer: false,
        create_white_background: true,
        font: 'Arial',
        font_method: 'font_family',
        fontsize_pt: 12.0,
        height: 40.0,
        horizontal_offset: 10.0,
        layername: '',
        left: 0.0,
        of: 'cropbox',
        overprint: 'off',
        place_barcode_parameter: {
                config: [
                        {
                          params: {
                                        barcode_parameters_json: {},
          },
        },
                ],
        },
        relative_to: 'lower_right_corner',
        right: 0.0,
        rotation_degrees: 0.0,
        spot_color_name: '',
        spot_color_tint_value: 100,
        top: 0.0,
        type_of_barcode: 'QR-Code',
        unit: 'mm',
```

```
        value: 'www.callassoftware.com',
        vertical_offset: 10.0,
        width: 40.0,
});
```

Click to copy

30.17 Old-style imposition - building blocks

In the following sections we will explain how the individual configuration ranges and all the parameters work. After that we will show how to proceed with setting up a new imposition using one of the examples supplied.

Sheet definition (Sheet Setup)

A sheet definition (Sheet Setup) comprises a PDF file with optional templates (Sheet Template) and a template definition (Sheet Template Con-figuration), which determines all the slots available on each individual template.

Name of a template layout

The configuration files file extension has to be ".sheetconfig" and the content has to start with the reserved word "NAME" followed by a <Tab> and a Name. In this file the sheet definition name is in UTF-8 code. (As long as only standard letters from A to Z and a to z without umlauts or accents, and numbers and simple signs such as period, comma, hyphen or under-score are used in names, ASCII corresponds to UTF-8 code: if you wish to use names with umlauts or signs from non-Western languages – such as Russian or Japanese – these will have to be saved in UTF-8. Some text editors – e.g. BBEdit 7 in Mac – have special save options for this).

- Note: UTF-8 encoding ensures that even names containing special characters (ä, ö, u, ß, …) or non-Western languages can be processed.

Under this name the sheet definition can be selected in the Desktop version of pdfToolbox in the "Impose" action dialog.



The configuration files file extension has to be ".sheetconfig" and the content has to start with the reserved word "NAME" followed by a <TAB> and a

Name. Many customers miss this TAB and use SPACE instead which will never work.

Template sheet

The SheetTemplate PDF file is an optional PDF file whose filename has to be the same as the filename of the sheetconfig (e.g. "880x630, 8up A4.sheetconfig.pdf"). During the imposition process the PDF pages from this file are inserted as a background in the PDF document to be imposed. The pages to be imposed are then positioned against this background. In the process the complete page contents of the template are transferred. In this way all static elements like crop marks and print control strips are perfectly positioned on the final imposition sheet.

At the same time, care should be taken with this file that the page size – as MediaBox (and if explicitly defined: CropBox) – corresponds to the sheet format.

The geometry of the individual slots on the template is determined in the sheetconfig file independently from this PDF file.

The template file can contain one or more templates (each PDF page in a template file constitutes a separate template). During the imposition pro-cess, the sheets are inserted sequentially into the target document accord-ing to the instructions in the imposition runlist.

If there are no more template pages in the template file, each page will be used in sequence during the imposition process, starting again from the first page if no pages are left but a new template is needed. Optionally, each page of the template can be used directly. If the template file only contains one template (i.e. a PDF page), the same template will be used over and over again.

If a SheetTemplate PDF exists, no SHEET entries are allowed in the ".sheetconfig"-file.

If no template PDF is available, the sheet size has to be defined using the following syntax:

```
SHEET        <left_bottom_x>        <left_bottom_y>        <right_top_x>
<right_top_y>
```

If more than one Sheet number is used in the slot configuration, there need to be the respective number of "SHEET" entries at the beginning of the .sheetconfig file. If a defined sheet number is used in the runlist but does not exist, the entries are iterated through using modulo and then the respective entry is used (e.g. if Sheet 3 is requested, but only 2 SHEET are defined, the first one will be used).

TotalPagesMultipleOf is optional and used for rounding up the number of pages to the next value which can be devided by TotalPagesMultipleOf. See the .runlist-section of this manual for a detailed example.

Further information about using template pages is explained during the definition of a runlist.

Template configuration (*.sheetconfig)

The details regarding the placing of the pages to be imposed on the tem-plate are in the SheetTemplateConfig.dat file. Here the individual slots are defined together with further information about their desired positioning. Every slot definition has to start with the reserved word "SLOT" followed by the following parameter:

Slot Name

Allows to define a name for this slot for easier understanding and readability.

Slot ID

Clear and unmistakable ID of a slot on a sheetpage (integer); valid values: 1, 2, 3, 4, etc. It is not necessary to number all of the slots although we recommend doing so for reasons of clarity.

Sheet ID

Page number of the templates in the template file, for which the slot is intended (integer): valid values: 1, 2, 3, 4, etc., whereby only the number 1 is valid in a one page template, and only 1 and 2, etc. in a two-page template file.

Trim Box left

Margin (Offset) of the left edge of the slot relative to the left edge of the MediaBox of the template (value and unit of measurement; acceptable units of measurement: mm, pt, cm, '); valid values e.g.: 12cm, 20.5pt, 3cm, 7.5'.

Trim Box bottom

Margin (Offset) of the bottom edge of the slot relative to the bottom edge (of the MediaBox) of the sheet (value and unit of measurement; acceptable units of measurement: mm, pt, cm, '); valid values: 12cm, 20.5pt, 3cm, 7.5'.

Trim Box width

Slot width (value and unit of measurement; acceptable units of measurement: mm, pt, cm, '); valid values: 12cm, 20.5pt, 3cm, 7.5' etc.

Trim Box height

Slot height (value and unit of measurement; acceptable units of measurement: mm, pt, cm, '); valid values: 12cm, 20.5pt, 3cm, 7.5' etc.

Bleed Offset left

Allowance (Offset) to the left edge of the page position for additional bleed (value and unit of measurement; acceptable units of measurement: mm, pt, cm, '); valid values: 12cm, 20.5pt, 3cm, 7.5', etc.

Bleed Offset bottom

Allowance (Offset) to the lower edge of the page position for additional bleed (value and unit of measurement; acceptable units of measurement: mm, pt, cm, '); valid values: 12cm, 20.5pt, 3cm, 7.5', etc.

Bleed Offset right

Allowance (Offset) to the right edge of the page position for additional bleed (value and unit of measurement; acceptable units of measurement: mm, pt, cm, '); valid values: 12cm, 20.5pt, 3cm, 7.5', etc.

Bleed Offset top

Allowance (Offset) to the upper edge of the page position for additional bleed (value and unit of measurement; acceptable units of measurement: mm, pt, cm, '); valid values: 12cm, 20.5pt, 3cm, 7.5', etc.

Scale X

Horizontal scaling factor; values greater than zero affect the corresponding scaling of the page to be positioned on the sheet in a horizontal direction; a zero value ensures that the page to be positioned is fitted onto the sheet horizontally; a value of minus one ensures that the same scaling factor is used as for the vertical scaling.

Valid values (each applicable to horizontal scaling):

100	equals 100%, or unscaled positioning
25	equals 25%, or reduce to one quarter of size
270	equals 270%, or enlarge by 2.7
etc.	

0	scale so that the page fits horizontally into the slot's Trim Box, i.e. scale to the Trim Box width

-1	use the same scaling factor as for the vertical scaling if -1 has been entered as a value for Scale Y, the page will be scaled proportionally so that it fits horizontally and vertically into the slot's TrimBox

Scale Y

Vertical scaling factor; values greater than zero affect the corresponding scaling of the page to be positioned on the sheet in a vertical direction; a zero value ensures that the page to be positioned is fitted onto the sheet vertically; a value of minus one ensures that the same scaling factor is used as for the horizontal scaling.

Valid values (each applicable to vertical scaling):

100	equals 100%, or unscaled positioning
25	equals 25%, or reduce to one quarter of size
270	equals 270%, or enlarge by 2.7
etc.	

0	scale so that the page fits vertically into the slot's TrimBox, i.e. scale to Trim Box height
-1	use the same scaling factor as for the horizontal scaling if the value -1 has also been entered for Scale X, the page is scaled pro-portionally in a horizontal and vertical direction so that it fits the slot's TrimBox

Rotation

Page rotates anti-clockwise (value, basic unit of measurement: degrees). The only approved values are: 0, 90, 180, 270.

Placement

The only approved values are:

LB	align left bottom
LC	align left center

LT	align left top
CT	align center top
RT	align right top
RC	align right center
RB	align right bottom
CB	align center bottom
CC	align center, i.e. both vertically and horizontally

- Note: The entry for the placing of the page to be positioned is relative to the slot, which in turn is defined by the TrimBox specifications.
- Note: If a slot is used for PlaceText, this text cannot be set to be right aligned by using Placement due to technical restrictions. Text will always start at the point defined by Placement, respecting orientation given by Rotation.

Binding Margin

Binding Margin defines the edge on which the creep should be equalized. Possible values are:

N	none
L	left
R	right
T	top
B	bottom

L0	left, without increase of the BleedBox
R0	right, without increase of the BleedBox
T0	top, without increase of the BleedBox
B0	bottom, without increase of the BleedBox

- Note: Additionally you need to set ShinglingOffset in the runlist for defining the paper thickness ("0mm" by default, which means no binding margin).

Cropmark Style Left Bottom

Define the style of the cropmark placed in the lower left corner.

Possible values:

N	none (default)
L	left
R	right
T	top
B	bottom
LT	left top
RT	right top
LB	left bottom

RB	right bottom

Cropmark Style Right Bottom

Define the style of the cropmark placed in the lower right corner.

Possible values:

see " Cropmark Style Left Bottom ".

Cropmark Style Right Top

Define the style of the cropmark placed in the upper right corner.

Possible values:

see " Cropmark Style Left Bottom ".

Cropmark Style Left Top

Define the style of the cropmark placed in the upper left corner.

Possible values:

see " Cropmark Style Left Bottom ".

Clip mode

Usually imposed pages are cropped according to their TrimBox plus bleed as set up in the slot.
Sometimes it is needed that the imposed page is clipped at the slot boundaries instead.

P	Positioned page (default)
S	Slot

Slot as isolated transparency group

This option allow to create a slot as a non-isolated transparency group.

I	Isolated (Default)
N	Not isolated

Comment (# sign)

After the # sign, it is possible to enter a comment.

Pagesize Filter

The `PAGESSIZE_FILTER` entry is used for filtering the display of sheet configurations in the callas pdfToolbox GUI. It has no effect on the imposition process. A sheet configuration is only displayed in the "Sheet configuration" pop-up menu if the `PAGESIZE_FILTER` matches the selected "Page size" pop-up in the 'Impose' single action window. If `IGNORE_ORIENTATION` is either one of "`YES`", "`ANY`", "`TRUE`", "`*`" the page orientation is ignored, otherwise the page size orientation is regarded.

```
PAGESIZE_FILTER        <width of n-ups>        <height of n-ups>        <allowed
widht-tolerance of n-up>        <allowed height-tolerance of n-up>        <ig-
nore_orientation>
```

The file format of a template configuration file

The file is saved and read in ASCII format. Columns are separated from each other using the <TAB> (Tabulator) key.

Empty lines are ignored. Lines which start with a # sign are interpreted as comments and as such are similarly disregarded.

30.18 Old-style imposition - controlling the process

In the imposition process the pages to be placed in the predefined slots are positioned within the respective templates.

There are three steps of placing pages in the imposition process. They are carried out in this strict order:

- rotation
- scaling
- relative placing

Rotating

If the configuration template SheetTemplateConfig.dat specifies a rota-tion (anti-clockwise by 90, 180 or 270 degrees) for the slot concerned, the page from the source document which is about to be imposed is rotated accordingly before further processing (the source document will remain unaltered by this).

Scaling

Next the page to be imposed from the source document is scaled by the given factor (if a scaling factor of 0.5 has been entered the page will be scaled down to half its original size).

If the special scaling values 0.0 or -1.0 have been entered, the size of the page to be imposed from the source document is compared with the TrimBox measurements for the relevant slot. The necessary scaling factor is then calculated and the page is automatically scaled using this factor so that the TrimBox fits into the slot.

Relative Placing

Finally, according to the information for relative positioning, the following action will be carried out: the respective reference point – e.g. left edge vertical centered (LC) – is calculated for the page to be imposed as well as for the slot con-

cerned and the page to be imposed is placed in the slot so that both reference points are exactly on top of each other. If a page needs to be rotated, it is important here to remember to calculate the reference point of the page to be imposed after this has happened.

30.19 Old-style imposition - runlists

The Runlist is based on a very simple scripting language that uses a few statements and variables to create sheets based on templates, place pages on those sheets, and manipulate the values of the variables used.

- Note: The Runlist contains information and instructions for the imposition process in a simple script language.

Name of the runlist

The configuration files file extension has to be ".runlist" and the content has to start with the reserved word "NAME" followed by a <Tab> and a Name. This file is just assigned the name Runlist in UTF-8 code. As long as only standard letters from A to Z and a to z without umlauts or accents, and numbers and simple signs such as period, comma, hyphen or under-score are used in names, ASCII corresponds to UTF-8 code.
If you wish to use names with umlauts or signs from non-Western languages – such as Russian or Japanese – these will have to be saved in UTF-8. Some text editors – e.g. BBEdit 7 in Mac – offer special save options for this. Under this name the sheet definition can be activated in the Desktop version of pdfToolbox in the "Impose" action dialog.

.runlist configuration

The .runlist file contains the Runlist with the actual imposition script that arranges the pages on sheets that have been taken from the sheet supply. With the help of commands and special counters the actual imposition process can be controlled within such an imposition script. The commands and counters available are explained below. The file hat to start with the reserved word "NAME" followed by a <Tab> and a Name. This is the name of the configuration as it will show up using the runlist pulldown menu of the imposition action in pdfToolbox.

Counters available in the run list

Page counters are needed when placing pages of the original PDF file onto the imposition sheet.

FirstPage

At the beginning of the imposition process this is set to 1 (as the first page in the source document is always specified as 1).

LastPage

At the start of the imposition process this is set to the number of pages in the source document. However, at the same time the parameter TotalPagesMultipleOf in the settings file SheetConfig.dat is taken into consideration and the value for the number of pages in the source document is rounded up to the next value that can be evenly divided by TotalPagesMultipleOf.
Without considering TotalPagesMultipleOf, in a 29-page document LastPage would have a value of 29. If the entry TotalPagesMultipleOf is set to 4, 29 will be rounded up to the next value dividable by 4 so that LastPage would have the value 32.

MidPage

At the beginning of the imposition process this is set to (LastPage / 2) +1

In the case of a 16-page document MidPage would have a value of 9 at the outset of the imposition process (LastPage would then be 16, as long as TotalPagesMultipleOf were not applied; 16 divided by 2 equals 8, plus one equals 9).

LastPhysicalPage

Return the number of the last physical page in the PDF. Might be useful as the value of "LastPage" usually changes during imposition.

Runlist commands

With the following commands you can insert a new sheet, place a page on a sheet or manipulate one of the counter variables.



There is a TAB (and not SPACE) between the command and the argument.

New Sheet

```
NewSheet        <Sheet number>
```

The parameter NewSheet inserts the next template page from the tem-plate file as a new sheet. If a template consists of multiple sheets, the sheets will be used in sequence. If there are no more sheets, the first sheet will be used again. The optional parameter <Sheet number> allows to use a specific sheet from the template.

Position Page

```
PositionPage        <COUNTER>        Slot_##:
```

```
PositionPage        FirstPage        Slot_1
```

This places the `<COUNTER>` page from the source document in the slot with the ID Slot_## and the current template.

<COUNTER>	must be one of the variables FirstPage, MidPage or LastPage

Slot_##	## is the number of the Slot, as defined in the template configuration

- Note: Before calling up PositionPage for the first time, the command NewSheet must have been called up at least once.

Position Page

```
PositionPage        <PAGE_COUNTER>        <SLOT>        <SLOT_COUNTER>
```

Click to copy
PositionPage with three arguments.
Places page <PAGE_COUNTER> in Slot <SLOT_COUNTER>, the <SLOT> param is ignored.
This allows to address the slot using a counter instead of a slot identifier. <SLOT_COUNTER> is optional.

Increment

```
Increment        <COUNTER>
```

```
Increment        FirstPage
```

The counter `<COUNTER>` counts upwards by one; the counter then points to the next page in the source document.

<COUNTER>	must be one of the values FirstPage, MidPage or LastPage

Decrement

```
Decrement        <COUNTER>
```

```
Decrement        LastPage
```

The counter `<COUNTER>` counts downwards by one; the counter then points to the previous page in the source document.

<COUNTER>	muss eine der Variablen FirstPage, MidPage oder LastPage sein

Append Pages

```
AppendPages
```

The command AppendPages does not take any parameters. Instead it appends empty pages according to the current value of variable <LastPage>. This fills the file to be imposed with pages up to the page count defined by LastPage.

- Note: This command should be executed early, best before calling the positioning of pages.

Destination Sheets

```
Set    DestSheetsMultipleOf    <Number (default: 1)>
```

```
Set    DestFillupSheet    <Number (default: 1)>
```

The imposed file will be filled up with empty pages from the sheet template (page DestFillupSheet) to the next page count divisible by the number defined in DestSheetsMultipleOf.

Place text

TextFont

```
Set        TextFont        <FONT>
```

```
Set        TextFont        "AmericanTypewriter"
```

Sets the font to be used for PlaceText. The default value is "Arial".

You can use each value from the list of known fonts, which will be created automatically when going to the action "Impose" in the Desktop version of callas pdfToolbox (section "Arrange"). You will then find the file "FontNames.txt" in the folder "Actions/Impose/Runlists" inside your pdfToolbox user preferences.

Using the CLI, the command `--listfonts` will list all font names available for impositioning.

TextSize

```
Set        TextSize        <SIZE>
```

```
Set        TextSize        "12pt"
```

Sets the text size to be used for PlaceText. The default value is "10pt".

TextColorSpace

```
Set        TextColorSpace        <Color space>
```

- Note: Color space can be any of: DeviceGray, DeviceRGB, DeviceCMYK or a Spot color name.

By default the text is using Separation color "All", also known as Regis-tration color. This parameter allows to define a specific color space to be used for the text element to be placed.

Defining this parameters requires to also define the parameter TextColorValues.

TextColorValues

```
Set        TextColorValues        <Color value>
```

```
Set        TextColorValues        "100/100/0/0"
```

This parameter defines the color values to be used based on the color space defined by TextColorSpace. The values have to be formatted in such a way that they represent the color tints of the color channels avail-able in the defined color space. The values have to be defined in a range from 0 to 100.

- For DeviceGray the values has to be the <Gray> tint in %
- For DeviceRGB the values have to be <Red>/<Green>/<Blue> in %
- For DeviceCMYK the values have to be <Cyan>/<Magenta>/<Yellow>/<Black> in %
- For <Spot Color> the values have to be either <Red>/<Green>/<Blue> or <Cyan>/<Magenta>/<Yellow>/<Black> in % and represent the alternate color space definition

TextTintValue

```
Set        TextTintValue        <Spot color tint value>
```

```
Set        TextTintValue        "50"
```

If using a spot color, this parameters allow to define the used tint value of the spot color.

PlaceText

```
PlaceText        <SLOT>        <TEXT>
```

```
PlaceText        Slot_3        "callas"
```

PlaceText uses the currently set values for TextFont and TextSize – if these are not defined, "Arial" and "10pt" will be used by default.

- Note: Only characters contained in WinAnsi or MacRoman encoding (by usage of the system's fonts) are supported at the moment. Therefore the text may only consist of characters that are present in the by font set by TextFont.

PageBox

```
BoxOrder        <BOX NAME>
```

```
BoxOrder        CropBox<Tab>TrimBox
```

This parameter defines which PageBox is used for positioning pages during imposition. A list of box names, tabulator-key separated is possible.

Allowed values are: ArtBox, BleedBox, TrimBox, CropBox, MediaBox

Shingling

Below are all the possible runlist commands related to shingling. A more detailed explanation of each parameter can be found in this article.

ShinglingMethod

```
Set        ShinglingMethod        <method>
```

Click to copy

```
Set        ShinglingMethod        "Scale"
```

Click to copy

This parameter is optional. If no ShinglingMethod is set, shifting is applied by default.

Supported shingling methods:

- `"Legacy"` **(Default):** Shingling by shifting the page contents inwards (ShinglingOffset < 0) or outwards (ShinglingOffset > 0)
- `"Shift"`
- `"Scale"`
- `"ScaleP"`

Note: The `"Legacy"` method uses the legacy ShinglingOffset. All other methods use the ShinglingIncrement instead.

ShinglingStapleSize

Note: This parameter has no effect for the `"Legacy"` shingling method.

```
Set        ShinglingStapleSize        <value>
```

Click to copy

```
Set        ShinglingStapleSize        "16"
```

Click to copy
Defines the number of signatures in a staple.

ShinglingDirection

Note: This parameter has no effect for the `"Legacy"` shingling method.

```
Set        ShinglingDirection        <direction>
```

Click to copy

```
Set        ShinglingDirection        "both"
```

Click to copy
Defines the direction of the shingling:

Supported singling directions:

- `"inwards"`
- `"outwards"`
- `"both"`

ShinglingIncrement

Note: This parameter has no effect for the `"Legacy"` shingling method.

```
Set         ShinglingIncrement         <unit value>
```

Click to copy

```
Set         ShinglingIncrement         "0,563mm"
```

Click to copy
The shingling value is incremented by this value for each signature in a staple.

This parameter is a unit value, e.g. "1mm" or "10pt".

ShinglingOffset

Note: Only if the `"Legacy"` shingling method is used the **ShinglingOffset** parameter can be used.

```
Set         ShinglingOffset         <offset>
```

```
Set         ShinglingOffset         "0,02mm"
```

ShinglingOffset or increment is applied **automatically** based on the page currently placed in relation to the middle page in the document. The default value is 0,00 mm for no binding moves.
A positive value moves placed pages away from the binding margin and negative value moves placed pages towards the binding margin. Normally negative values are more useful since inner pages of a booklet normally require moving page contents inside.

Starting from the first and last page the shingling value is incremented every two pages by ShinglingOffset starting with the value 0.
Example:

For a 16 page document:

Pages 1,2,15,16: Shingling = 0 * ShinglingOffset
Pages 3,4,13,14: Shingling = 1 * ShinglingOffset
Pages 5,6,11,12: Shingling = 2 * ShinglingOffset
Pages 7,8,9,10: Shingling = 3 * ShinglingOffset

ShinglingValue

```
Set        ShinglingValue        <value>
```

```
Set        ShinglingValue        "-0,05mm"
```

When a value is defined, "ShinglingValue" overwrites the current shingling value calculated on the basis of "ShinglingOffset". The default value is 'empty'.

Shingling

Shingling is read only that denotes the "ShinglingValue" that was effective during the last placement action.

If-else condition

This allows to process runlist functions and commands depending on specific conditions.

```
If        <boolean condition>
<processing steps>
Else
<processing steps>
EndIf
```

Additional available functions are:

```
ElseIf
ErrorIf
```

Example:

```
If        var("FirstPage")>1
PlaceText    Slot_102    var("FirstPage")
```

```
Else
PlaceText    Slot_102    "Not first page"
EndIf
```

Working through the script to be imposed

The Runlist is gone through from start to finish, step by step, repeating as often as necessary, until the condition

```
FirstPage > LastPage
```

is met. This means that the Runlist must be set up in such a way that this condition is only met once all the pages have been imposed.

If the Runlist has been worked through to the end but the break condition has not yet been met, the process will start all over again.

If the break condition is met before the Runlist reaches the end, the way the Runlist process terminates is defined by the `RunListTerminationMode` parameter.

`RunListTerminationMode` defines the behaviour when FirstPage gets larger than LastPage. Allowed values are 1 and 2, the default is 1.

1. The loop stops. If the sheet is not completely imposed, it is discarded.
2. If the loop stops (FirstPage > LastPage), the currently imposed sheet is stored even if not all slots are filled.

30.20 Old-style imposition - token engine

The TokenEngine can process literal and numeric values, the results of call up functions, and delivers results in the form of dynamically generated strings which can be used in various places within the program.

For example, during the imposition process it is possible with the Token Engine to place dynamically generated texts on an imposition sheet.

Syntax

Literal

A literal character token is a text enclosed by quotation marks.

If quotation marks themselves are used in the text, these must be marked by a preceding '\' backslash: \ "

If the backslash '\' is used in the text, this must also be marked by a back-slash: '\\'.

Examples of literal tokens

```
"This is a text"
```

```
"Date:"
```

```
"He said \"Good      morning!\""
```

```
"C:\\\\Programs\\Test.txt"
```

Numbers

A number is made up of the numeric characters 01234567892 and one or no decimal point .

Examples of numbers

```
1
```

```
1.982
```

```
5000.0
```

```
75
```

Operators

Calculation operators for numeric tokens: + - * / %

Combination operators for literal tokens: &

Parameter lists

A parameter list is a list of expressions separated by commas.

Exceptions to parameter lists are the empty list and the list that consists of one parameter only.

Example for parameter lists

```
"benjamin","britten"
```

```
1,2,3
```

```
"text and numbers",5
```

```
"today is the "&      date() , 8 , 9
```

Functions

There are various functions depending on the module. But the syntax for calling up functions is always the same:

```
functionname(parameterlist)
```

Each function requires a particular number of parameters. Please see below in the documentation for each function for details.

Examples of call up functions

```
date()
```

```
date("DD.MM.YY")
```

```
length("text")
```

```
left("this is a text",4)
```

Numerical expressions

A numerical expression consists of numbers, operators and functions.

The Token Engine recognizes the four basic calculation methods +, -, * and / . Priority conventions are respected ,e.g. multiplication and division comes before addition and subtraction. All function tokens can be used in numerical expressions as long as they result in numerical values.

Expression

An expression is a string of literal tokens, numeric expressions and call up functions joined by the combination operator '&'.

Examples for expressions

```
"This is a simple text!      Believe me!"
```

```
"today is "&      date("DD.MM.YYYY") &"."
```

```
"5 + 5 = "& 5 + 5
```

```
fileName(docpath())
```

```
left("this is a      test",length("this"))
```

Function groups

Text functions

left(text,num)

gives us num characters from the beginning of text.

```
left("This is an example",4)
```

gives us "This"

right(text,num)

gives us num characters from the end of text.

```
right("This is an example",3)
```

gives us "ple"

middle(text,pos,num)

gives us num characters beginning with pos from the text.

```
middle("This is an example",6,2)
```

gives us "is"

replace(text,pos,count,replacementText)

replaces count characters from Position pos in the text with replacementText and gives us the result

```
replace("This is an example", 10, 9 , " test")
```

gives us "This is a test"

substitute(text,pattern,replacementText)

replaces all occurrences of pattern in the text with replacementText.

```
substitute("This is an example","s","***")
```

gives us "Thi*** i*** an example"

length(text)

gives us the length of a text.

```
length("This is an example")
```

gives us 18

position(text, searchText,pos,n)

gives us the position of the n-th occurrence of searchText from Position pos.

```
position("This is an example","s",3,2)
```

gives us 7 .

regex(text,pattern)

gives us 1 if the regular expression pattern matches the input text text.

```
regex("This is an example","^(This)(.*)(example)$")
```

gives us 1

regex(text,pattern,format,noMatchText)

gives us noMatchText if the regular expression pattern does not match the input text text, otherwise the text will be re-turned having been for-matted with format

```
regex("This is an example","^(Dies)(.*)(example)$","$1","ERROR")
```

gives us "This"

```
regex("This is an example","^(This)(.*)(example)$","$2","ERROR")
```

gives us " is an"

```
regex("This is an example","^(This)(.*)(example)$","$3$2$1", "ERROR")
```

gives us "example is an This"

```
regex("This is a text","^(This)(.*)(example)$","$3$2$1","ERROR")
```

gives us "ERROR"

Date and time functions

date()

gives us today’s date in the format "DD.MM.YYYY" (e.g. "03.11.2008")

date(format)

gives us today's date in any number of possible formats.

Placeholders that can be used are: D: day, M: month, Y: year.

Examples:

DD.MM.YYYY	03.11.2008
YYYY-MM-DD	2008-11-03
DD	03
D	3

time()

gives us the current time in the format "hh:mm:ss" (e.g. "10:05:49")

time(format)

gives us the current time in any number of formats.

Placeholders that can be used are: h: hour, m: minute, s: second.

Examples:

hh.mm.ss	10:05:49
mm	05
m	5

datetime()

gives us the current date and time in the format "DD.MM.YYYY hh:mm:ss" (e.g. "08.09.2003 10:05:49")

datetime(format)

gives us the current date and time in any number of formats.

Placeholders that can be used are:

D	day
M	month
Y	year
h	hour
m	minute
s	second

Examples:

YYYY-MM-DD-hh-mm-ss	2008-11-03-10-05-49
YYYYMMDDhhmmss	20081103100549

Logic functions

if(a,b,c)

If parameter a = "0" expression c will be returned, otherwise expression b:

```
if(var("LastPositionedPage") , "Page: " & var ("LastPositionedPage"), "Pagenumber not valid!")
```

(Below you will find further explanation about the "var"-function.)

choose(a,b,c,...)

gives us independently of parameter a the a-th entry from the list (count start with 0).

```
choose(0,"Null","One","Two","Three")
```

gives us "Null"

```
choose(3,"Null","One","Two","Three")
```

gives us "Three"

```
choose(date("M"),"","Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec")
```

in September gives us "Sep"

Numeric functions

a+b

gives us the sum of a and b.

```
1+2
```

gives us 3

a-b

gives us the difference between a and b.

```
4-2
```

gives us 2

a*b

gives us the product of a and b.

```
4*2
```

gives us 8

a/b

gives us the division of a with b.

```
4/2
```

gives us 2

a%b

gives us the rest of the division of a with b.

```
33%16
```

gives us 1

abs(a)

gives us the absolute value of a

```
abs(-5)
```

gives us 5

The basic calculation methods can, however, also be written as direct numerical expressions. Here arithmetic rules and conventions such as the use of brackets must be respected.

```
3 + 3
```

gives us 6

```
4 - 2
```

gives us 2

```
8 / 2
```

gives us 4

```
4 * 4
```

gives us 16

```
(6 + 2 * (3 - 1)) / 2
```

gives us 5

max(a,b,c,...)

gives us the highest value from all listed values.

```
max(1,8,2,7,3,6,4,5)
```

gives us 8

min(a,b,c,...)

gives us the lowest value from all listed values.

```
min(1,8,2,7,3,6,4,5)
```

gives us 1

ROUND(<FLOAT>[,<fractional digits>])

rounds a floating number to an integer. The number of fractional digits can be optionally defined as well.

```
ROUND(10.4)
```

gives us 10

```
ROUND(10.5)
```

gives us 11

```
ROUND(10.0015,3)
```

gives us 10.002

File functions

docpath()

gives us the complete path to the current PDF file including the file name.

```
docpath()
```

gives us e.g. "/Users/callas/Documents/test.pdf" on Mac and "C:\\temp\test.pdf" on Windows.

filename(path)

gives us the file name of the currently processed PDF document.

```
filename("C:\PDF files\impose.pdf")
```

gives us "impose.pdf"

parentfolder(path)

gives us the parent folder name of the currently processed PDF document.

```
parentfolder("C:\PDF files\impose.pdf")
```

gives us "C:\ PDF files"

appendfileorfolder(path, file or folder name)

allows to merge a file path

```
appendfileorfolder("C:\PDF files\","impose.pdf")
```

gives us "C:\ PDF files\impose.pdf"

readfile(path)

gives us the content of a specified file.

```
readfile("C:\TXT files\impose.txt")
```

gives us the content of the text file "impose.txt"

lookup(path, key, selector)

allows to read values from a tab de-limited file which could be a text file containing entries separated by tabulators. Key specifies the entry in the first row. selector specifies an entry in the first line.

```
lookup("C:\TXT files\chart.txt", LINE2, ROW3)
```

gives us the value that is listet at position LINE2 and ROW3 of the tab de-limited file chart.txt.

Systeminfo functions

username()

Gives us the user name of the user currently registered on the workstation

machinename()

gives us the name of the workstation

osversiontext()

gives us the system version of the operating system (e.g. "Mac OS X 10.5.8")

CLI functions

var(name,format)

gives us the variable name from the RunList (e.g. in the sheet configuration) in its format format. Any number of variables can be defined with the command set from the runlist.

Pre-defined variables of particular importance are:

ShinglingOffset	current value of the creep correction
CropMarkGap	current distance between TrimBox and crop-mark
CropMarkLength	current length of the crop-marks
CropMarkWidth	current line width of the cropmarks
TextSize	current text size
TextFont	current font name
LastPositionedPage	last posi-tioned page
CurrentSheet	current sheet number

RunListName	name of the current run-list
SheetConfigName	name of the current sheet configu--ration

All formats are available:

Text as entered	"string" (default)
Point	"pt"
Millimeter	"mm"
Centimeter	"cm"
Inch	""

RunListTerminationMode(num)

defines the behaviour when `FirstPage` gets larger than `LastPage`.

Allowed values are 1 and 2, the default is 1.

- 1 The loop stops. If the sheet is not completely imposed, it is discarded.
- 2 If the loop stops (FirstPage > LastPage), the currently imposed sheet is stored even if not all slots are filled.

CropMarkColorSpace

```
Set CropMarkColorSpace {"DeviceGray" | "DeviceRGB" | "DeviceCMYK" | <Spot Color>}
```

By default crop marks are using Separation color "All", also known as Registration color. This parameter allows to define a specific color space to be used for the crop marks to be cre-

ated. Defining this parameters requires to also define the parameter CropMarkColorValues.

CropMarkColorValues

```
Set CropMarkColorValues <Color value>
```

```
Set CropMarkColorValues "100/100/0/0"
```

This parameter defines the color values to be used based on the color space defined by CropMarkColorSpace. The values have to be format-ed in such a way that they represent the color tints of the color channels available in the defined color space. The values have to be defined in a range from 0 to 100.

- For DeviceGray the values has to be the <Gray> tint in %
- For DeviceRGB the values have to be <Red>/<Green>/<Blue> in %
- For DeviceCMYK the values have to be <Cyan>/<Magenta>/<Yellow>/<Black> in %
- For <Spot Color> the values have to be either <Gray> or <Red>/<Green>/<Blue> or <Cyan>/<Magenta>/<Yellow>/<Black> in % and represent the alternate color space definiton

CropMarkTintValue

Set CropMarkTintValue <Spot color tint value>

Set CropMarkTintValue "50"

docinfo(key)

gives us the document information of a PDF file for a particular key.

Available keys:

- "Title"
- "Subject"
- "Author"

30.21 Old-style imposition - an example imposition configuration

Creation of a Sheet Setup Configuration

Requirement

Impose 4 pages A5 on a press sheet with a dimension of 468mm by 315 using a "cut and stack" imposition method. The imposed PDFs shall be centered on the press sheet with no cutting gap (extra distance between the imposed pages)

1) .sheetconfig

Create a text file called "468x315_4up_upright_A5.sheetconfig".

The first line has to define the name. In our example it looks like this:

```
NAME        468x315,        4up A5,        upright
```

This file contains the positions (slots) on the press sheet where the PDFs get placed. Each line in this file represents a slot. All entered values have to be Tab-delimited. The sequence of the needed values are described in "Sheet definition (Sheet Setup)" in this section.

2) Template size

We are imposing on a blank sheet. this means we do not need an extra template PDF but we need to define the sheet size. In our example the line looks like this:

```
SHEET        0mm        0mm        468mm        315mm
```

3) Mathematics

Next calculate the positions of the lower left corner of the first trimmed pages on the press sheet.

Since the total width of the 2 A5 landscape pages is 420 mm (2 x 210mm), the first page starts at 24mm from the left page boarder. This is the x-axis (468 - 420 = 48 divided by 2 = 24).

The start on the y-axis is then at 9.5mm = (315-(148+148))/2

If additional cutting gap is needed, it has to add it when calculating the total height or width of the imposed pages. It is recommended to prepare a little drawing with the values on it.

Our first slot on the (first) template sheet starts a 24mm/9.5mm and has a dimension of 210mm/148mm.

If we need bleed around the imposed pages and the current slot is the lower/left slot on our imposed sheet, bleed is only possible on the left and the bottom border. We must not define any bleed on the right and upper border, since it would reach into the trimmed are of the adjacent slot as there is no cutting gap in between.

Example: 3mm bleed

The next entries in the SheetTemplateConfig allow for scaling and rota-tion of the content of the slot. You would enter rotation for 2 of the 4 slots, that we are going to define, if you would create a regular 4up imposition schema where you need the pages to be placed head-to-head.

Example: We need the pages to be placed at 100% with no rotation.

The next entry defines which "hot-spot" of the placed PDF page shall be positioned at which "hot-spot" of the slot. This has no relevance if the slot dimension and the trimmed PDF page dimension is the same. But if the trimmed size was smaller for example, you had to define where the page shall be positioned within the larger slot area.

Example: RT (right top corner)

The next entry defines where at which edge creep shall be applied. This is currently not fully supported and I therefore recommend to enter the value "N", which stands for "None".

The last four entries are to define cut marks positions. Every corner of a slot can have cut marks defined. If you image the imposed page and the slot 1 (lower left on the sheet), we would need cut marks at the following positions:

Bottom left corner	LB	Cut mark going left and down
Bottom right corner	B	Cut mark going down (left is an adjacent page)
Top right corner	N	No cut mark (this is the area where all 4 imposed pages meet)
Top left corner	L	Cut mark going left (up is an adjacent page)

- Note: Please note that the following examples have to be pasted into one line without line breaks. A new line is always marked with the keyword "Slot".

The resulting line in the config file now looks like that:

```
Slot Slot_LB 1 1 24mm 9.5mm 210mm
148mm 3mm 3mm 0mm 0mm 100 100
0 RT N LB B N L
```

For the next slot we have to calculate the next start coordinate and adjust the bleed and the cut marks accordingly. If we decide to place slot 2 right of slot 1, the line entered looks like that:

```
Slot Slot_RB 2 1 234mm 9.5mm
210mm 148mm 0mm 3mm 3mm 0mm 100
100 0 RT N B RB R N
```

For the next 2 slots (slot 3 being placed above slot 1) the entries look like that:

```
Slot Slot_LT 3 1 24mm 157.5mm
210mm 148mm 3mm 0mm 0mm 3mm 100
100 0 RT N L N T LT
```

```
Slot Slot_RT 4 1 234mm 157.5mm
210mm 148mm 0mm 0mm 3mm 3mm 100
100 0 RT N N R RT T
```

Since front page and back page of the imposed sheet use the same geom-etry (imposed pages are centered on the press sheet), we do not need to add extra positions for sheet 2.

Our sheet template is now ready.

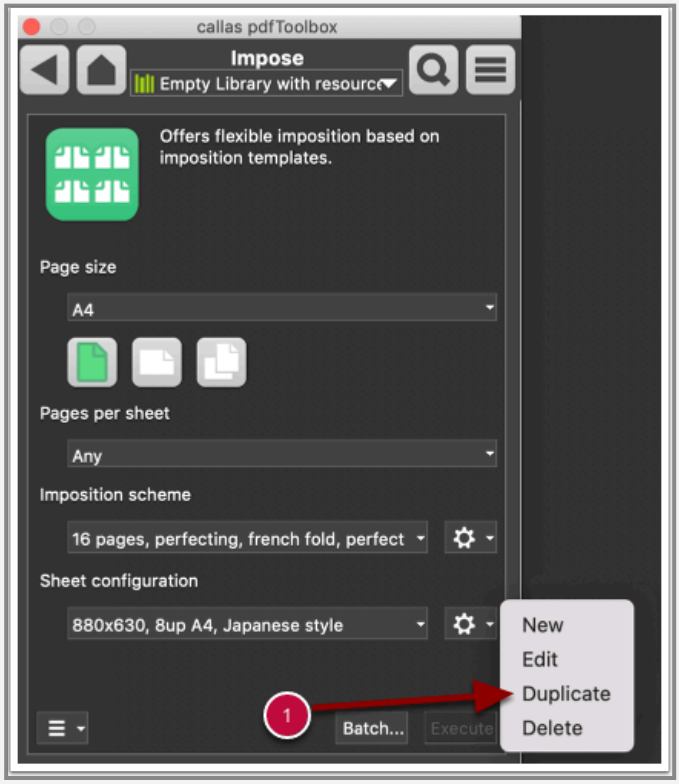
6) Usage

For the Server/CLI version the folder is

```
<CLI folder>/var/Actions/Impose
```

For the Desktop version and starting pdfToolbox 13, you can edit the sheet configuration directly in the Switchboard Action:

1. The best way is to duplicate an existing configuration and work on that



Up until pdfToolbox 12:

To use this template in the Desktop version of pdfToolbox, copy the just created settings folder into the following folder inside your user preferences:

```
callas software/callas pdfToolbox <version>/etc/Actions/Impose
```

Creation of a Run List Configuration

The runlist defines, which page of a PDF has to be placed on which sheet at what position (slot).

1) RunList.runlist

Now create a 2nd text file with the name "4pages_cut-stack.runlist": The first line has to define the name. In our example it looks like this:

Name 4 pages, cut & stack

followed by:

2) Global Settings

First we define some global settings. This is:

```
Set         CropMarkLength          "2,0mm"
Set         CropMarkWidth         "0,04mm"
Set         CropMarkGap         "2,5mm"
Set         ShinglingOffset          "0,00mm"
Set         RunListTerminationMode          2
Set         CropMarkColorSpace        "DeviceCMYK"
Set         CropMarkColorValues         "0/0/0/100"
```

- Note: Do not forget to separate the parameters, the key words and the values using the `<Tab>` -Key.

RunListTerminationMode is used to make sure the the engine fills each printed sheet, if necessary with blank pages, if there are too little pages in the original PDF file as are needed for imposition.

3) Mathematics

Since we need a multiple of 4 pages in order to impose the sheet, we have to do some math. This math is also needed, since we do a cut & stack imposition where we need to know the page count difference between the first page and the adjacent imposed page. The following formula is for sin-gle sided printing. At the end I give an example for work and turn printing.

```
Set         c         (var('LastPage')%4)
```

I check to see if the pagecount can be divided by 4.

```
Set         y         if(var('c'),4-var('c'),0)
```

Now I fill up to have a "virtual" pagecount that can be divided by 4.

```
Set         c1         (var('LastPage')+var('y'))/4
```

Now I calculate the value we need to increment between each facing page on an imposed sheet.

```
Set         LastPage         var('c1')
```

This use of LastPage is defined later.

Another method do have the page count as a multiple of 4 is to use the former defined "SheetConfig.runlist" file of the Sheet setup. You get the same result if you enter there the number 4 instead of 1. The downside of that is, that this sheet config is now limited to be used with a minimum of 4 pages and can not be used for example for step & repeat, where 1 page in a PDF file is already enough to fill the press sheet.

4) Sheet content

Now we have to fill the sheets with content. This is done starting with a loop command using the script language of the runlist.

So the next command is:

Loop	This marks the position in the script to jump back to when the engine hits the end of the runlist file, put still has pages

	left to im- pose

Next we need a new press sheet using

```
NewSheet
```

Now we are ready to position a page on the sheet. This is done by:

```
PositionPage        FirstPage        Slot_3
```

FirstPage currently refers to the first page of the PDF being processed. This page is placed at Slot 3 of the sheet template as we defined it above (which is the upper left slot)

Since PositionPage only works with any of the three page pointers, which are FirstPage, MidPage and LastPage, we have to set them to the proper values.

In our example I go ahead with MidPage. For that, I define which page should get placed next to the already positioned page. The variable "c1" holds the value that is needed to increment the page pointer.

```
Set        MidPage        var('FirstPage')+var('c1')
```

Now position the page in a slot (upper right slot):

```
PositionPage        MidPage        Slot_4
```

We have now filled the first row of the press sheet. Now we go ahead with the 2 remaining slots:

```
Set        MidPage        var('MidPage')+var('c1')        PositionPage        Mid-
Page        Slot_1
Set        MidPage        var('MidPage')+var('c1')        PositionPage        Mid-
Page        Slot_2
```

At the end we increment FirstPage. This has 2 reasons:

- At every new press sheet, we start by positioning First-Page. FirstPage therefore has to have the correct page number.

- If FirstPage is higher or equal to LastPage, the process is finished and the imposed sheets are ready. LastPage is calculated in such a way, that it works for the needed cut & stack imposition schema. This is also the reason why working with MidPage while filling the press sheet. If using FirstPage (LastPage being set to the original LastPage div 4) the engine would stop after the first press sheet since FirstPage had a value higher than LastPage.

Finally, the content of the "RunList.runlist" file is:

```
Name         4pages, cut and stack
Set         CropMarkLength         "2,0mm"
Set         CropMarkWith         "0,04mm"
Set         ShinglingOffset         "0,025mm"
Set         RunListTerminationMode         2
Set         CropMarkColorSpace         "DeviceCMYK"
Set         CropMarkColorValues         "0/0/0/100"
Set         c         (var('LastPage')%4)
Set         y         if(var('c'),4-var('c'),0)
Set         c1         (var('LastPage')+var('y'))/4
Set         LastPage         var('c1')
Loop
NewSheet
PositionPage         FirstPage         Slot_3
Set         MidPage         var('FirstPage')+var('c1')
PositionPage         MidPage         Slot_4
Set         MidPage         var('MidPage')+var('c1')
PositionPage         MidPage         Slot_1
Set         MidPage         var('MidPage')+var('c1')
PositionPage         MidPage         Slot_2
Set         FirstPage         var('FirstPage')+1
```

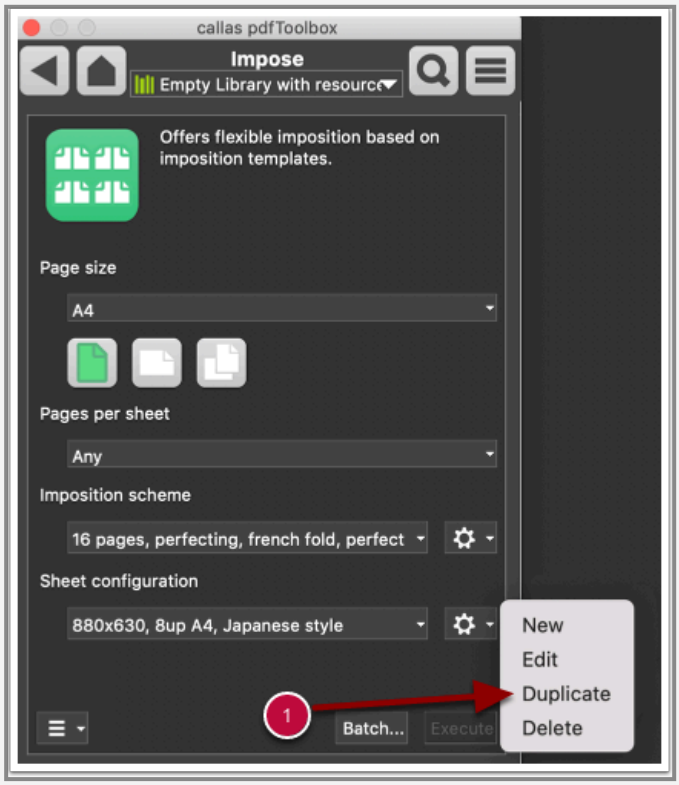
5) Usage

For the Server/CLI version, the folder is

```
<CLI folder>/var/Actions/Impose
```

For the Desktop version and starting pdfToolbox 13, you can edit the runlist configuration directly in the Switchboard Action:

1. The best way is to duplicate an existing configuration and work on that



Up until pdfToolbox 12: To use this template in the Desktop version of pdfToolbox, copy the just created settings folder into the following folder inside your user prefer-ences:

```
callas software/callas pdfToolbox <version>/etc/Actions/Impose
```

6) Process PDF

If you now open a PDF, choose the "Impose" Action from pdfToolbox, and you should see the 2 configurations just cre-ated.

Runlist for Work & Turn Printing (2 sided imposition)

If we would do a work & turn printing, we need a multiple of 8 pages to impose a sheet (4 pages per side).

Here is the formula for that:

```
Set         c          (var('LastPage')%8)
Set         y          if(var('c'),8-var('c'),0)
Set         c1          (var('LastPage')+var('y'))/4
Set         LastPage          var('c1')
```

And we have to place the page on the back page of the imposed sheet in a slightly different order to have them in the right position for a work & turn printing:

```
Loop
NewSheet
PositionPage          FirstPage          Slot_3
Set         MidPage          var('FirstPage')+var('c1')
PositionPage          MidPage          Slot_4
Set         MidPage          var('MidPage')+var('c1')
PositionPage          MidPage          Slot_1
Set         MidPage          var('MidPage')+var('c1')
PositionPage          MidPage          Slot_2
Set         FirstPage          var('FirstPage')+1
```

Up to here it is the same, but now it changes a little bit:

Now we need a new sheet:

```
NewSheet
```

and fill it in a slightly different order:

```
PositionPage          FirstPage          Slot_4
Set         MidPage          var('FirstPage')+var('c1')
PositionPage          MidPage          Slot_3
Set         MidPage          var('MidPage')+var('c1')
PositionPage          MidPage          Slot_2
Set         MidPage          var('MidPage')+var('c1')
PositionPage          MidPage          Slot_1
Set         FirstPage          var('FirstPage')+1
```

An alternate method for the work & turn would have been to create a dif-ferent sheet setup which defines 2 sheets, where the names of the slots on the 2nd sheet are changed in such a way, that it can get filled using the first runlist.

- Note: It depends on your personal liking if you would prefere to have less sheet configs and more Runlists or the other way round.

More versatile Runlist

Here is an extended runlist where you just need to change the content of the 2 variables. This way it is more easy to adapt the runlist.

General variables

```
Name         4pages, cut and stack, duplex
Set         CropMarkLength         "2,0mm"
Set         CropMarkWith         "0,04mm"
Set         CropMarkGap         "2,5mm"
Set         ShinglingOffset         "0,0mm"
Set         RunListTerminationMode         2
Set         CropMarkColorSpace         "DeviceCMYK"
Set         CropMarkColorValues         "0/0/0/100"
```

Please set accordingly

```
Set         PagesPerSide         4         Set         NumberOfSides         2
```

And here begins the imposition myth...

```
Set         c         (var('LastPage')%(var('PagesPerSide')* var('NumberOfSides')))
Set         y         if(var('c'),(var('PagesPerSide')* var('NumberOfSides'))-
var('c'),0)
Set         IncrCount         (var('LastPage')+var('y'))/ var('PagesPerSide')
Set         LastPage         var('IncrCount')
Loop
NewSheet
PositionPage         FirstPage Slot_3
Set         MidPage         var('FirstPage')+var('IncrCount')
PositionPage         MidPage         Slot_4
Set         MidPage         var('MidPage')+var('IncrCount')
PositionPage         MidPage         Slot_1
Set         MidPage         var('MidPage')+var('IncrCount')
PositionPage         MidPage         Slot_2
Set         FirstPage         var('FirstPage')+1
```

```
Set         Condition         var('NumberOfSides')    -1
```

```
If        Var("Condition")
NewSheet
PositionPage        FirstPage        Slot_4
Set        MidPage        var('FirstPage')+var('IncrCount')
PositionPage        MidPage        Slot_3
Set        MidPage        var('MidPage')+var('IncrCount')
PositionPage        MidPage        Slot_2
Set        MidPage        var('MidPage')+var('IncrCount')
PositionPage        MidPage        Slot_1
Set        FirstPage        var('FirstPage')+1 EndIf
```

30.22 Old-style imposition - using variables during imposition

The CLI version allows to define variables using the `--set-variable` parameter within the command line call. These CLI Environment variables can be retrieved using the following syntax :

ENVIRONMENT(<name of variable>[,<type of variable>,<default>])

- Note: The type must always be "string".

Example:

```
Set        Tint        ENVIRONMENT("Tint")
```

```
Set        Tint        ENVIRONMENT("Tint","string","50")
```

Command line parameter to define the Environment variable:

```
--setvariable=Tint:80
```

30.23 Old-style imposition - creating variable-size impositions

To make imposition more dynamic, a number of Token about page sizes and Runlist Variables have been made available.
They can be used to create new or modify existing SheetSizes as well as slots.
Also the number of Sheets, the current Sheet and the current Slot can be determined.

Samples how to use them are available in this manual as well.

Token for page sizes

All page geometry boxes of each page can be derived using the following Token.

The resulting values will be given in pt.
<PAGE> must be replaced by the requested page number of the PDF.

MEDIABOXLEFT(<PAGE>) MEDIABOXRIGHT(<PAGE>) MEDIABOXTOP(<PAGE>) MEDIABOXBOTTOM(<PAGE>) MEDIABOXWIDTH(<PAGE>) MEDIABOXHEIGHT(<PAGE>)	Left edge on x axis Right edge on xaxis Top edge on y axis Bottom edge on y axis Width Height
CROPBOXLEFT(<PAGE>) CROPBOXRIGHT(<PAGE>) CROPBOXTOP(<PAGE>) CROPBOXBOTTOM(<PAGE>) CROPBOXWIDTH(<PAGE>) CROPBOXHEIGHT(<PAGE>)	Left edge on x axis Right edge on xaxis Top edge on y axis Bottom edge on y axis Width Height
BLEEDBOXLEFT(<PAGE>) BLEEDBOXRIGHT(<PAGE>) BLEEDBOXTOP(<PAGE>) BLEEDBOXBOTTOM(<PAGE>) BLEEDBOXWIDTH(<PAGE>) BLEEDBOXHEIGHT(<PAGE>)	Left edge on x axis Right edge on x axis Top edge on y axis Bottom edge on y axis Width Height
TRIMBOXLEFT(<PAGE>)	Left edge on x axis

TRIMBOXRIGHT(<PAGE>) TRIMBOXTOP(<PAGE>) TRIMBOXBOTTOM(<PAGE>) TRIMBOXWIDTH(<PAGE>) TRIMBOXHEIGHT(<PAGE>)	Right edge on x axis Top edge on y axis Bottom edge on y axis Width Height
ARTBOXLEFT(<PAGE>) ARTBOXRIGHT(<PAGE>) ARTBOXTOP(<PAGE>) ARTBOXBOTTOM(<PAGE>) ARTBOXWIDTH(<PAGE>) ARTBOXHEIGHT(<PAGE>)	Left edge on x axis Right edge on x axis Top edge on y axis Bottom edge on y axis Width Height

Create new or modify existing Sheets

```
SetSheet        <Sheet number>
```

The command SetSheet changes or adds a Sheet definition. If a Sheet definition for <Sheet number> already exists the command overrides the current definition. Otherwise it adds a new Sheet definition.

Added Sheet definitions will not be part of the Sheet sequence evaluation if NewSheet is used without <Sheet number> parameter. Sheets are actually inserted into the imposed document only after the complete runlist has executed. Therefore it is not recommended to set Sheet sizes to different values for the same <Sheet number>, the values from the last call to NewSheet will always be used for all Sheets created with NewSheet regardless of the Sheet definition that was in place at the time where NewSheet was called!

The MediaBox and optional TrimBox values for the new or modified Sheet definition are taken from the following variables, which must be set before calling SetSheet.

SHEET_DEF_MEDIABOX_L SHEET_DEF_MEDIABOX_B SHEET_DEF_MEDIABOX_W SHEET_DEF_MEDIABOX_H	Lower left starting point on y-axis Lower left starting point on x-axis Width Height
SHEET_DEF_TRIMBOX SHEET_DEF_TRIMBOX_L	Activate TrimBox for a Sheet, value must be "1" Lower left starting point on y-axis

SHEET_DEF_TRIMBOX_B SHEET_DEF_TRIMBOX_W SHEET_DEF_TRIMBOX_H	Lower left starting point on x-axis Width Height

Create new or modify existing Slots

```
MakeSlot        <Sheet number>        <Slot number>
```

Creates a dynamic Slot with id <Slot number> for Sheet <Sheet number>.
If a Slot with the same Id already exists in the Sheet config the old Slot is shadowed by the dynamic Slot.
The Slot parameters for the dynamic Slot are taken from the following variables (see also the documentation for the Slot parameters in the documentation of the sheet config file format).

SLOT_DEF_TRIMBOX_L SLOT_DEF_TRIMBOX_B SLOT_DEF_TRIMBOX_W SLOT_DEF_TRIMBOX_H SLOT_DEF_BLEED_L SLOT_DEF_BLEED_B SLOT_DEF_BLEED_R SLOT_DEF_BLEED_T SLOT_DEF_SCALE_X SLOT_DEF_SCALE_Y SLOT_DEF_ROTATION SLOT_DEF_PLACEMENT SLOT_DEF_BindingMargin SLOT_DEF_CropMarkStyleLB SLOT_DEF_CropMarkStyleLT SLOT_DEF_CropMarkStyleRT SLOT_DEF_CropMarkStyleRB SLOT_DEF_ClipMode SLOT_DEF_TGSetting	Lower left starting point on y-axis (Default: 0) Lower left starting point on x-axis (0) Width (100) Height (100) Bleed offset left (0) Bleed offset bottom (0) Bleed offset right (0) Bleed offset top (0) Scaling x-axis (100 percent) Scaling y-axis (100 percent) Rotation (0) Placement in slot (LB) Binding margin (N) Crop mark style left bottom (N) Crop mark style left top (N) Crop mark style right top (N) Crop mark style right bottom (N) Crop to page box (P) or to slot (S) (default: page box) Position as isolated (I) or non-isolated transpancy group (N) (default: isolated)

Delete Slot

```
DeleteSlot        <Sheet number>        <Slot number>
```

Deletes a dynamic Slot with id <Slot number> for sheet <Sheet number>.
If a Slot with the same Id exists in the sheet config the old Slot is reactivated.

Capture Slot

```
CaptureSlot        <Sheet number>        <Slot number>
```

Captures the values from a Slot from the sheet config.
Populates all Variables used by MakeSlot with the values defined in the sheet config.

Create and use a Function definition

```
Function        <TAB>        <FUNC_NAME>
```

Start the definition of a Function with name <FUNC_NAME> (literal)

```
EndFunc
```

Terminates a Function definition

```
Return
```

Terminates the execution of a Function before its completion

```
Call        <CALL_FUNC_NAME>
```

Calls the Funtion with the name <CALL_FUNC_NAME> (Token expression)

Retrieving information about Sheets and Slots

```
NUMSHEETS()
```

gives us the number of available/defined sheets.

```
NUMSLOTS(<SHEET_INDEX>)
```

gives us the number of the Slots on the Sheet with the defined index.

```
GETSLOTID(<SHEET>,<SLOT_INDEX>)
```

gives us the Slot ID (needed for defining the Slot during imposition) of the specified Sheet and the specified Slot index.

31. Downloadable and adjustable solutions

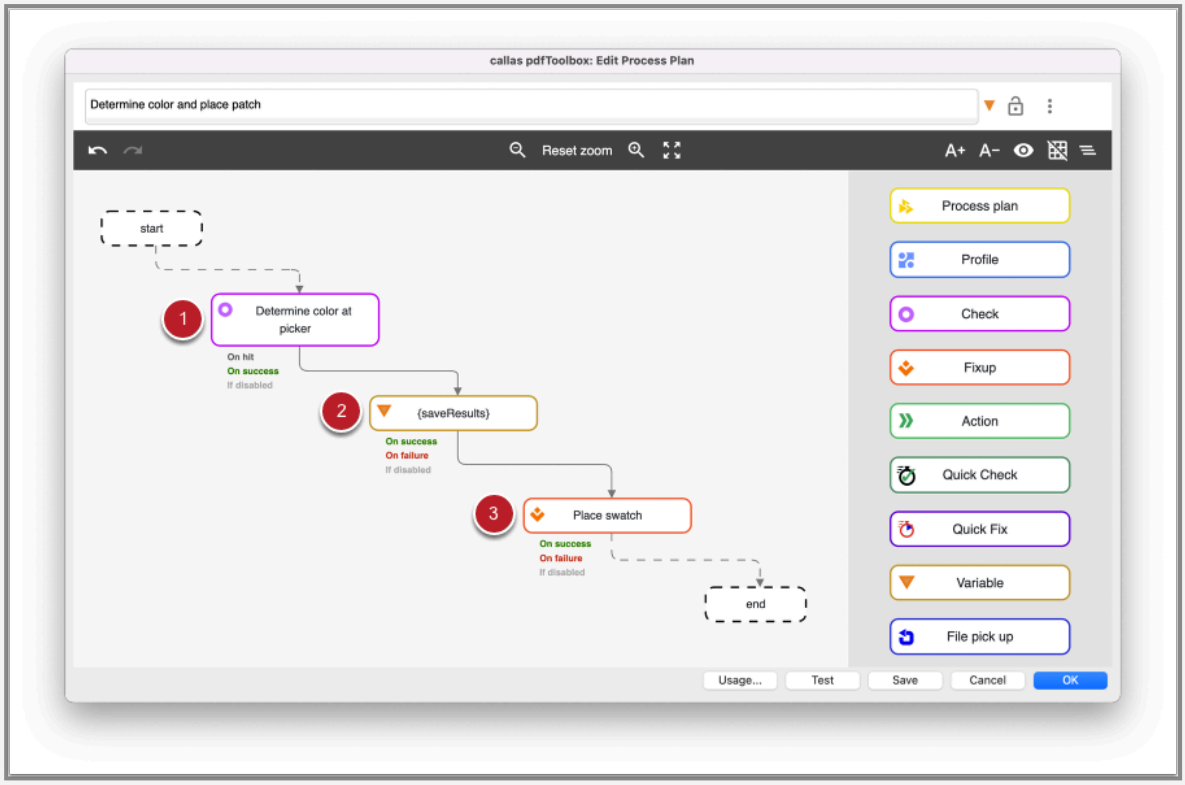
31.1 Process Plan: Color picker to derive a specific color

This Process Plan can be used to derive a color at the specified area and place a color swatch:

Determine color and place patch.kfpx

(earliest version with full support is **pdfToolbox 15**)

callas_testfile.pdf



1. Determine color at picker

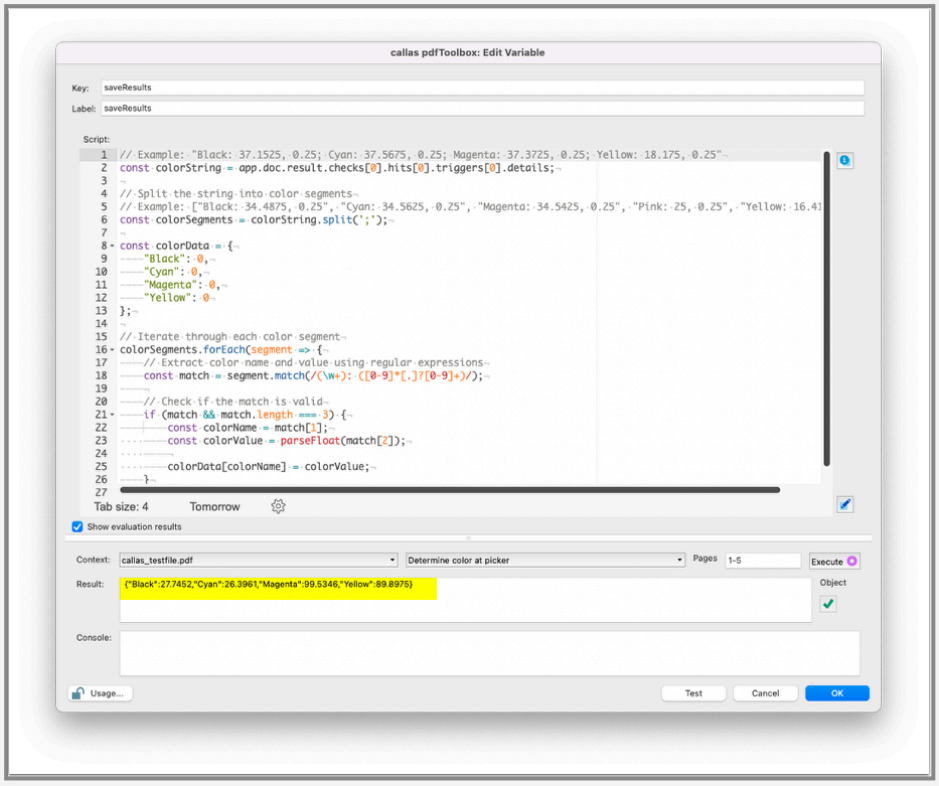
To determine the average color of a custom area, the "Effective ink coverage for separated plate" Check property is used in combination with the "Effective ink coverage – limit to custom area" Check property to limit the color detection to a

custom area. The Check uses variables, so that the position and size of the color swatch can be defined in the Ask-at-runtime dialog that appears when the Process Plan is executed.

In the trigger value the Check returns the color values for all color separations (process and spot colors) in percent (example: "Black: 31.7922, 0.28; Cyan: 30.5014, 0.28; Magenta: 94.5429, 0.28; Yellow: 79.59, 0.28").

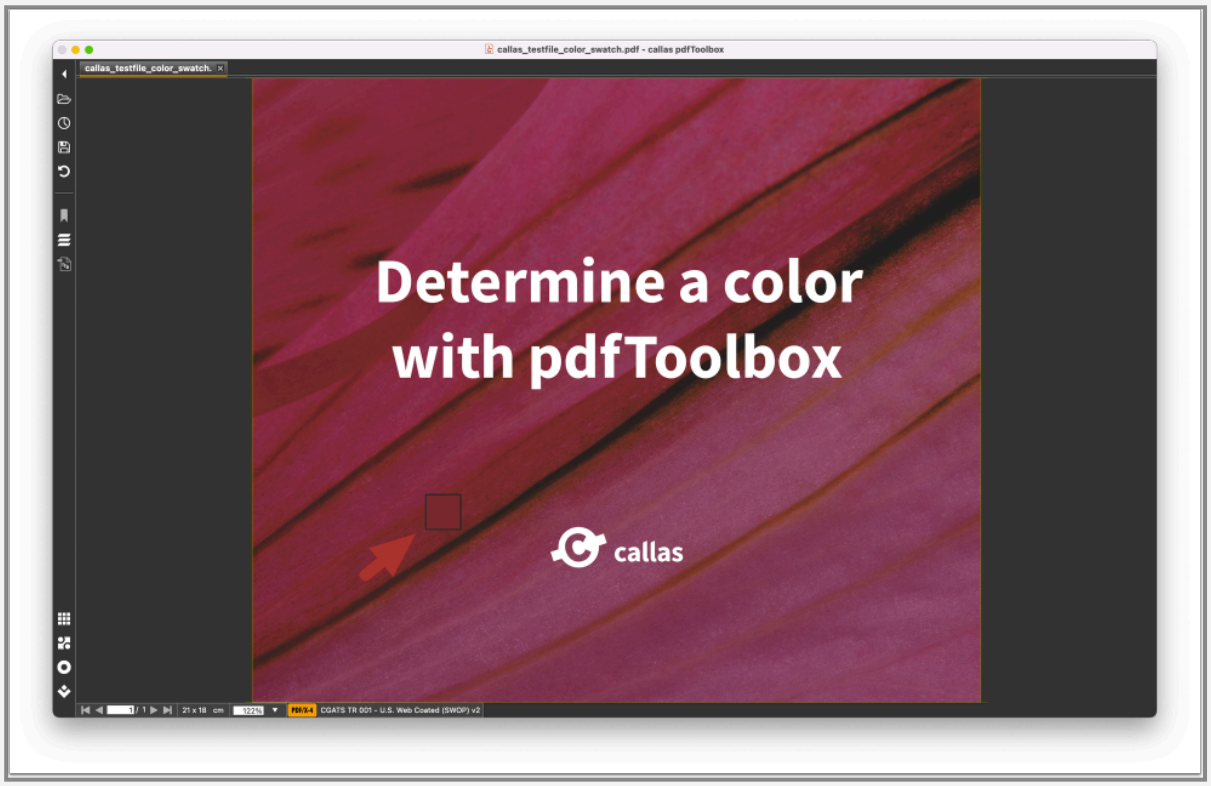
2. Variable step

To use the color values for further processing, they have to be extracted from the string. This is done in a variable step using JavaScript: The string is split into segments, each representing a separation name with its corresponding ink coverage percentage. As a result, the extracted color data is stored in a JavaScript object, where each separation name is associated with its respective ink coverage percentage. The area highlighted in yellow in the screenshot below shows the evaluation result.



3. Place swatch

This Fixup step places a square color patch with the determined color values of the JavaScript object on top of the PDF page. This step is optional and is only used for demonstration purposes to visualize where the color was determined and what the color looks like.



Limitations

The color swatch does not take spot colors into account. However, the spot color values in the respective area are available in the JavaScript object.

This Process Plan can only determine a color on one page. It is not designed to determine colors on multiple pages.

31.2 Process Plan: Determine text in custom area

This Process Plan can be used to derive text in a specified area. There are two variations of this Process Plan:

1. Determine text in area: Can be used for PDF files containing "regular" text

Determine text in area.kfpx

(earliest version with full support is **pdfToolbox 14**)

Testfile_extract_text.pdf

2. Determine text in area with OCR: Can be used for PDF files that do not contain "regular" text objects (e.g. scanned page)

Determine text in area with OCR.kfpx

(earliest version with full support is **pdfToolbox 15**)

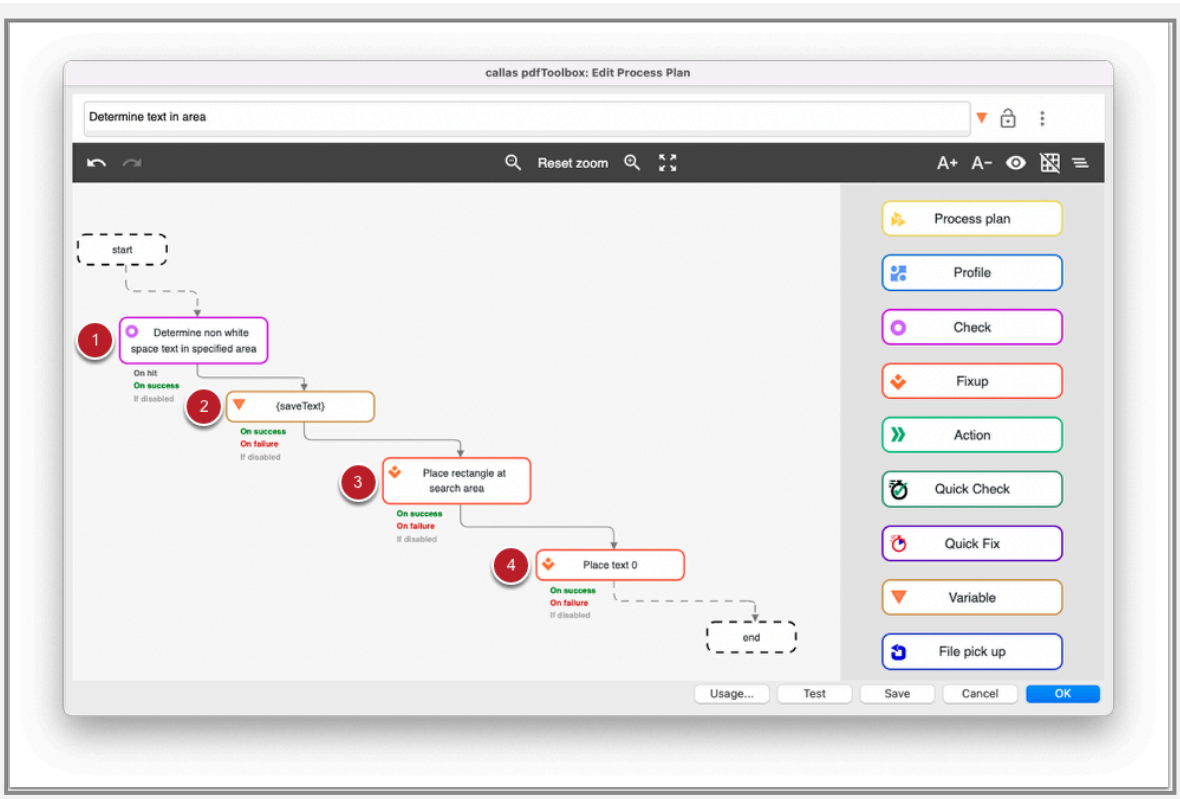
Testfile_extract_text_OCR.pdf

Both Process Plans are essentially the same. "Determine text in area with OCR" has three additional steps at the beginning:

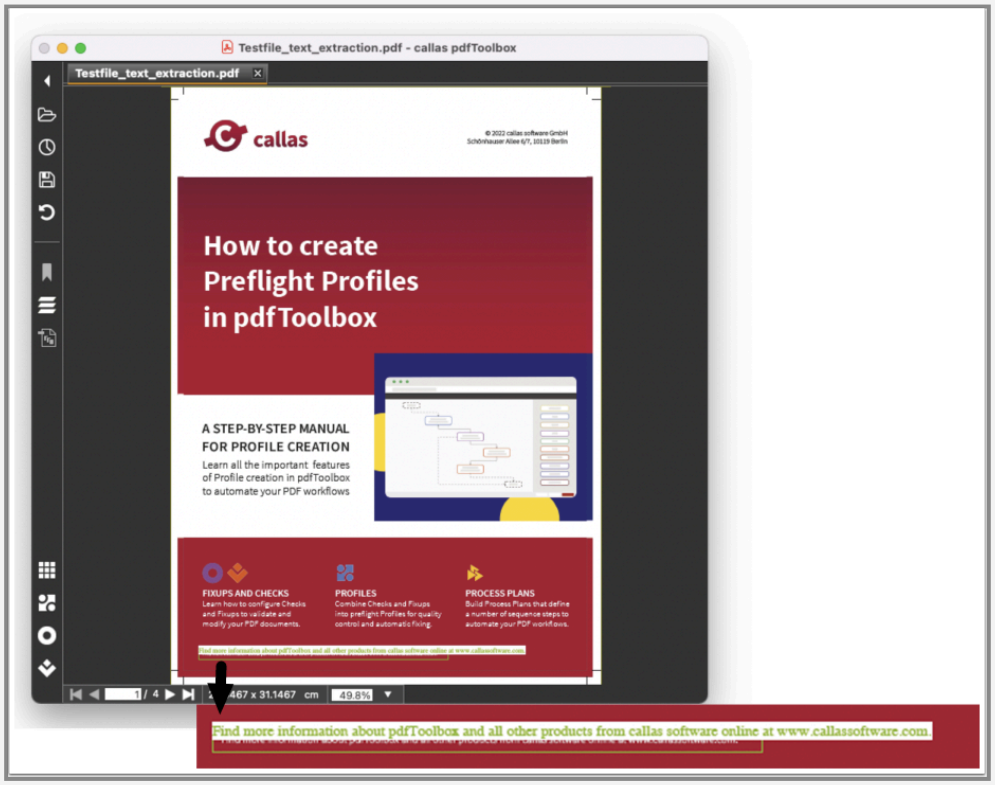
1. Remove existing OCR text
2. Convert page into an image
3. Create a new OCR text

These steps are only necessary to create a proper OCR text that allows good text extraction. After the engine has analyzed the text in the defined area, the original file will be picked up.

The next steps are the same for both variants:



1. This Check uses the property "Text on page" with a RegEx to determine non white space text in a specified area on the page. The regular expression matches any string that contains at least one non-whitespace character.
2. If text is found in the specified area, it is returned as a string in the JavaScrip object. If no text is found, the string "no text found" is returned.
3. For demonstration purposes: Places a green rectangle around the search area to visualize where the text was extracted (see result PDF below).
4. For demonstration purposes: Places the extracted text in green color on the page at the same location where it was extracted (see result PDF below).



Limitations

These Process Plans can only extract text on one page. They are not designed to extract text on multiple pages.

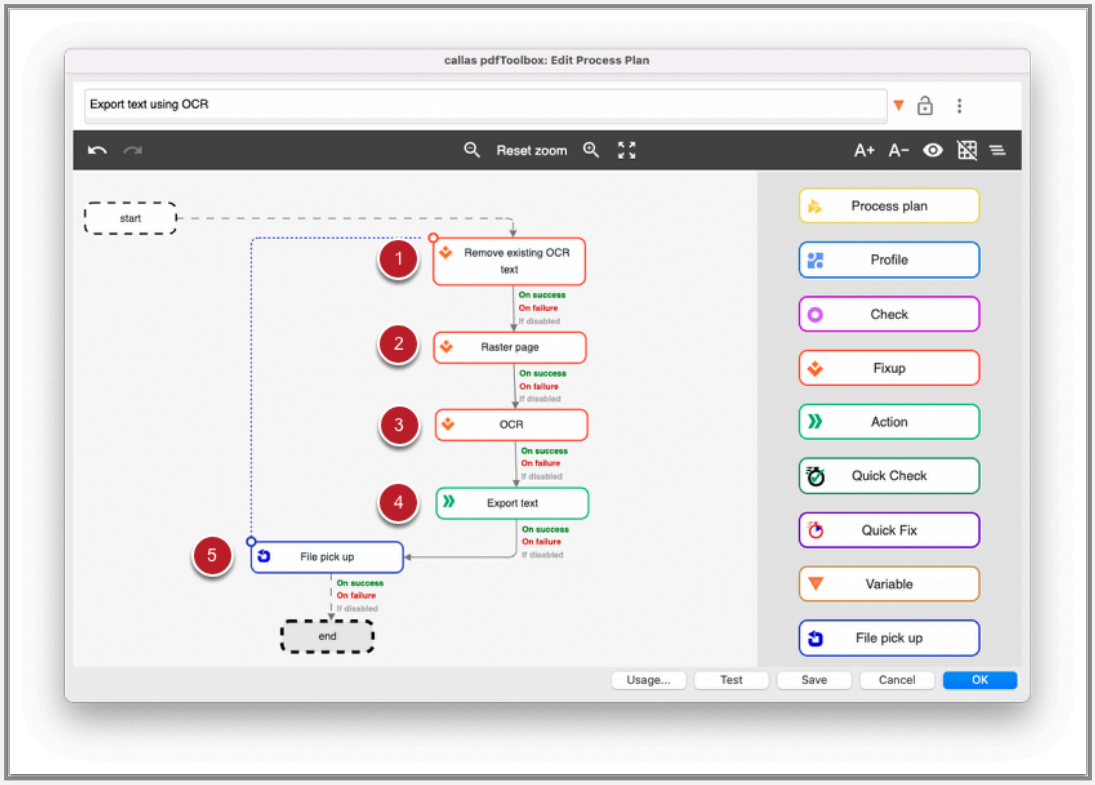
31.3 Process Plan: Extract text using OCR

This Process Plan creates ORC text on all pages and extracts the text into a .txt file next to the input file

Export text using OCR.kfpx

(earliest version with full support is **pdfToolbox 15**)

callas_testfile_screenshot.pdf

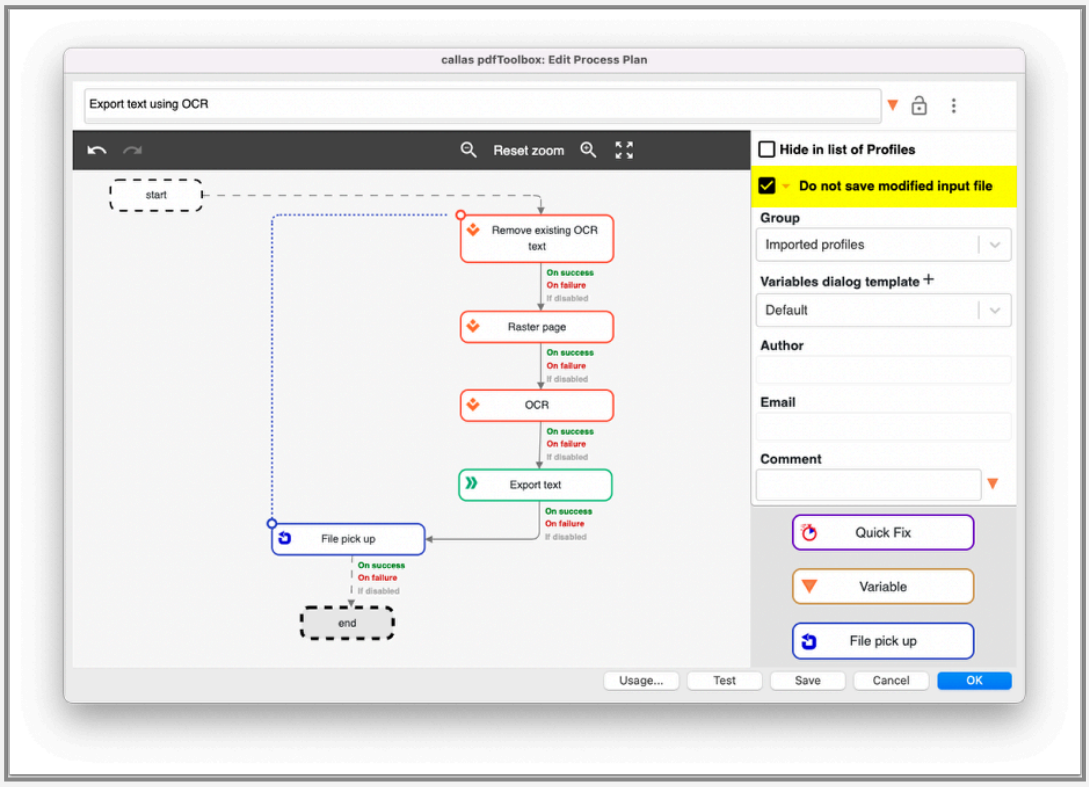


1. To make sure that the OCR text is good, the existing OCR text will be deleted.
2. The page will be convert into an image.
3. Create a new OCR text for all pages.
4. Extracts the text into a .txt file that will be saved next to the input file.

5. After the engine has extracted the OCR text into a text file, the original file will be picked up.

Why does pdfToolbox not show a "save as" dialog when executing the Process Plan?

Since the last step in the Process Plan is a "File pick up", this means that the original file is restored. It is therefore not necessary to save the result. In these cases, the "Do not save modified input file" checkbox can be activated in the Process Plan parameters to suppress the "Save as" dialog.



31.4 Process Plan: Scale page excluding Processing Steps

For some Fixups, it is not possible to limit the Fixup to specific page objects, but only to select a page number in the Apply to box.

For example, the page content needs to be scaled, but the line needs to be excluded from the scaling. To make this possible, this Process Plan provides a solution to exclude specific page objects from scaling:

Scale page excluding Processing Steps.kfpx

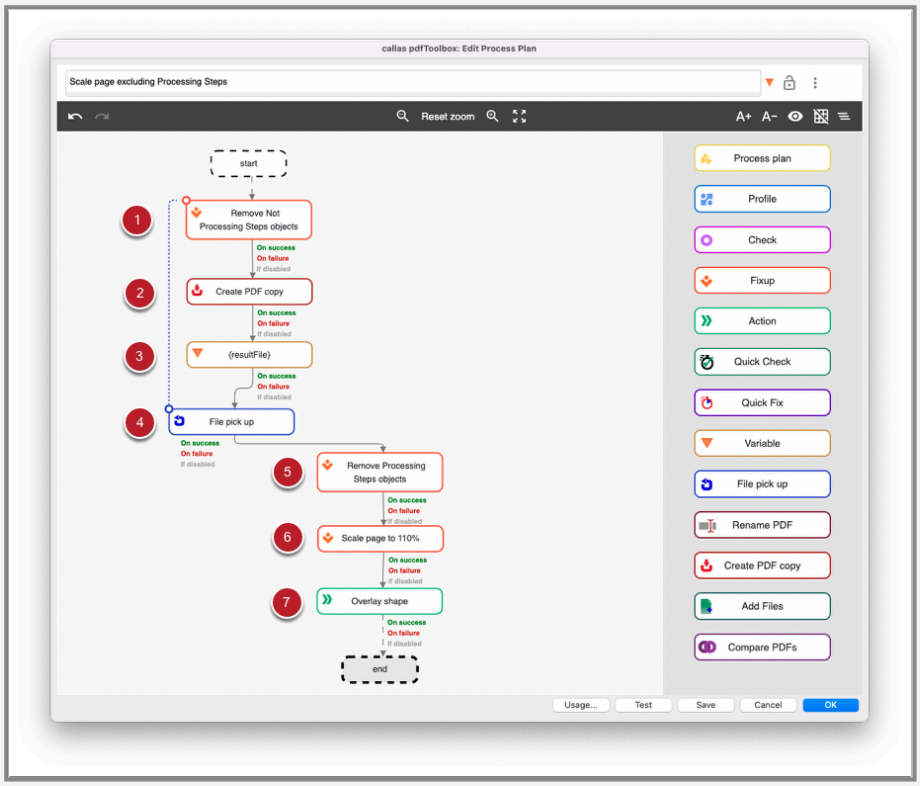
(earliest version with full support is **pdfToolbox 15**)

Lable_Processing_Steps.pdf

The Testfile "Lable_Processing_Steps.pdf" has a Die line with Processing Step Metadata Structual:Cutting.

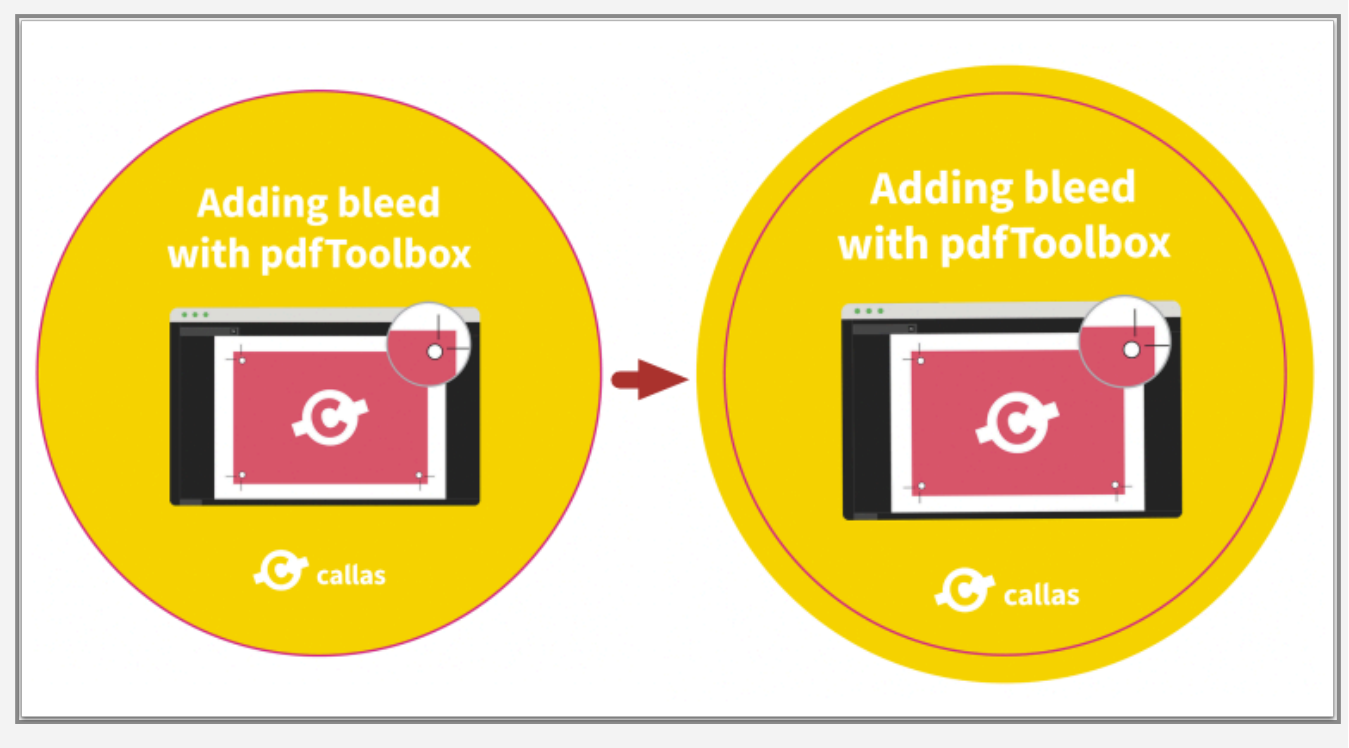


Steps of the Process Plan



1. This Fixup removes all page objects that do not have Processing Steps Metadata. In this example, only the Die line is not deleted.
2. A temporary copy of this result (only objects with processing step metadata) is created and will be deleted again after processing.
3. In this variable step, the temporary copy of step 2 is assigned the variable "overlay" using JavaScript.
4. The original file is then picked up again.
5. This Fixup removes any page objects with Processing Steps Metadata. In this example the Die line will be deleted.
6. The page is scaled to 110% (since the Die line was deleted in the step before it was excluded from the scaling).
7. An Action overlays the created copy from step 2 (includes only the Die line) on top of the scaled page.

Result (original vs. output)



31.5 Process Plan: Create bookmarks from headings

This article provides a Process Plan that can be used to automatically generate bookmarks from the headings in a PDF.

The test file (taken from the PDF/VT sample files at https://pdfa.org/resource/cal-poly-pdfvt-test-suite/) contains several brochures for different cities. The Process Plan determines the headings (city name on the first page of each brochure) and uses them to create a bookmark structure.

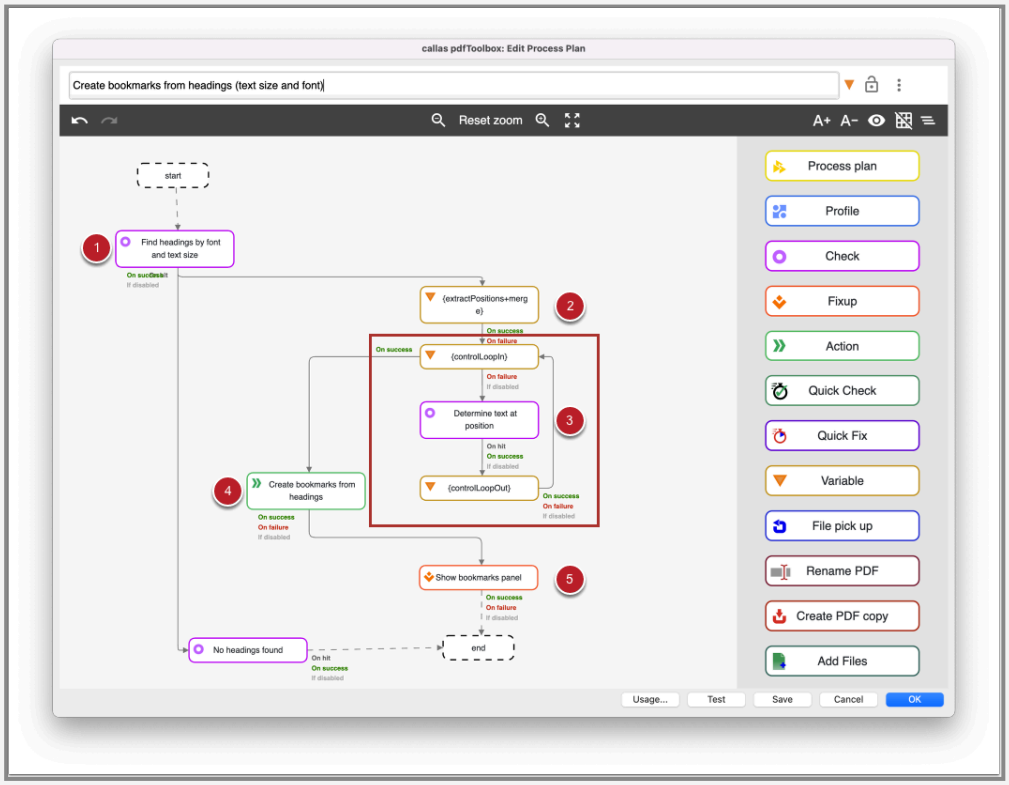


1 Travel V1.0.1 - 10 small.pdf

Create bookmarks from headings.kfpx

(earliest version with full support is **pdfToolbox 14**)

Let's have a look at the Process Plan:

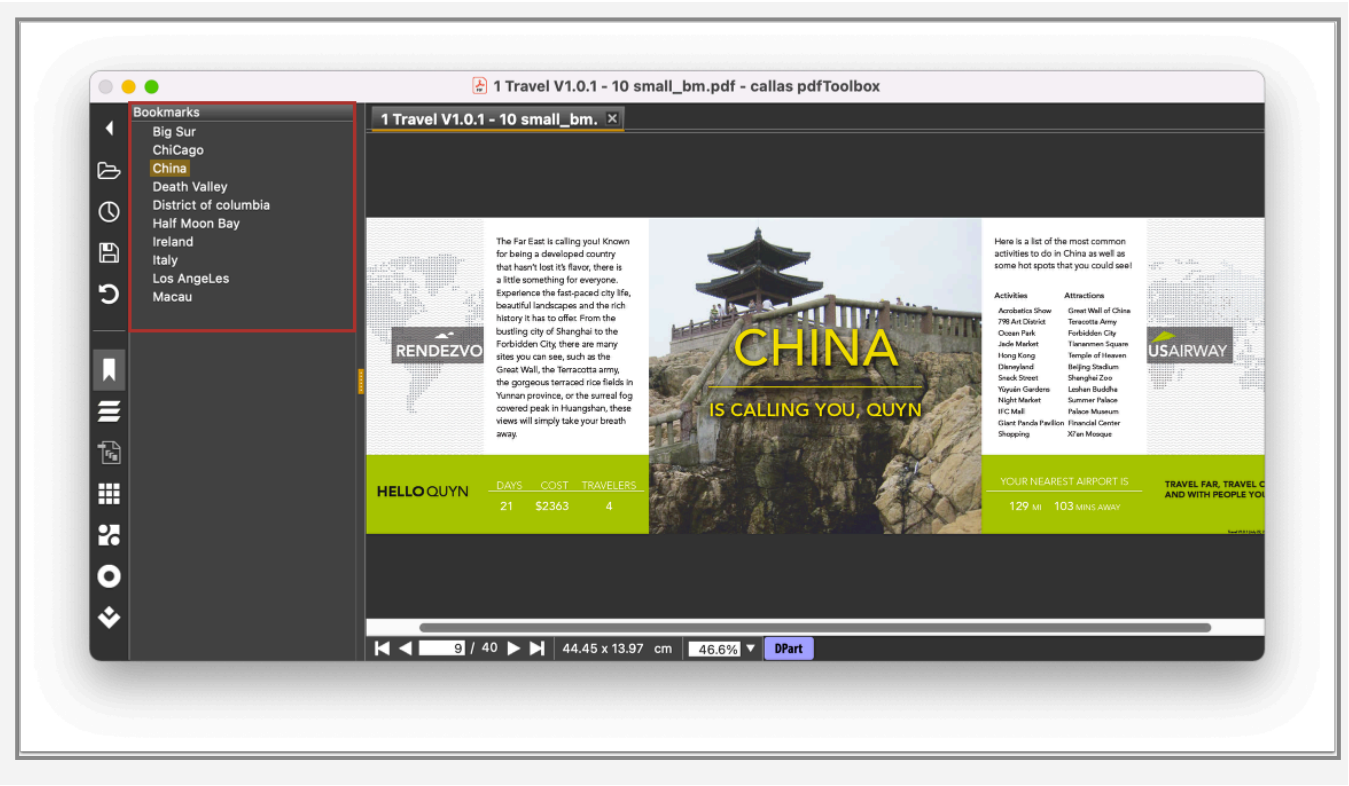


1. First, the positions of the headings are determined by a matching Check. This Check is individual and works only for the sample PDF file. It is important to properly detect all headings in the PDF using a combination of Check properties (font size, font type, font color, etc.).
2. The next step is the extraction of the text at the positions that have been detected in step 1. If the position of a text snippet touches another text snippet, they will be merged. This is important if you have headings that extend over two lines (like "District of Columbia" - see screenshot above) to receive only one bookmark for that heading.
3. A loop goes over an array to determine all headings. When all headings are determined, the next step starts.
4. To inject the bookmarks into the PDF the Process Plan uses the "Apply strutures" Action. The information that has been gathered so far is converted into a proper JSON structure that can be used in this Action for bookmarks.
5. Finally, a Fixup defines that the bookmarks panel should be visible by default in the PDF viewer.

When executing the Process Plan, a Ask-at-runtime dialog will appear, to increase the search area for the text extraction

slightly in order to find all headings properly (the default value of 20 pt is suitable in most cases).

Result PDF with new bookmark structure



31.6 Process Plan: Use QR Codes to place icons and link annotations

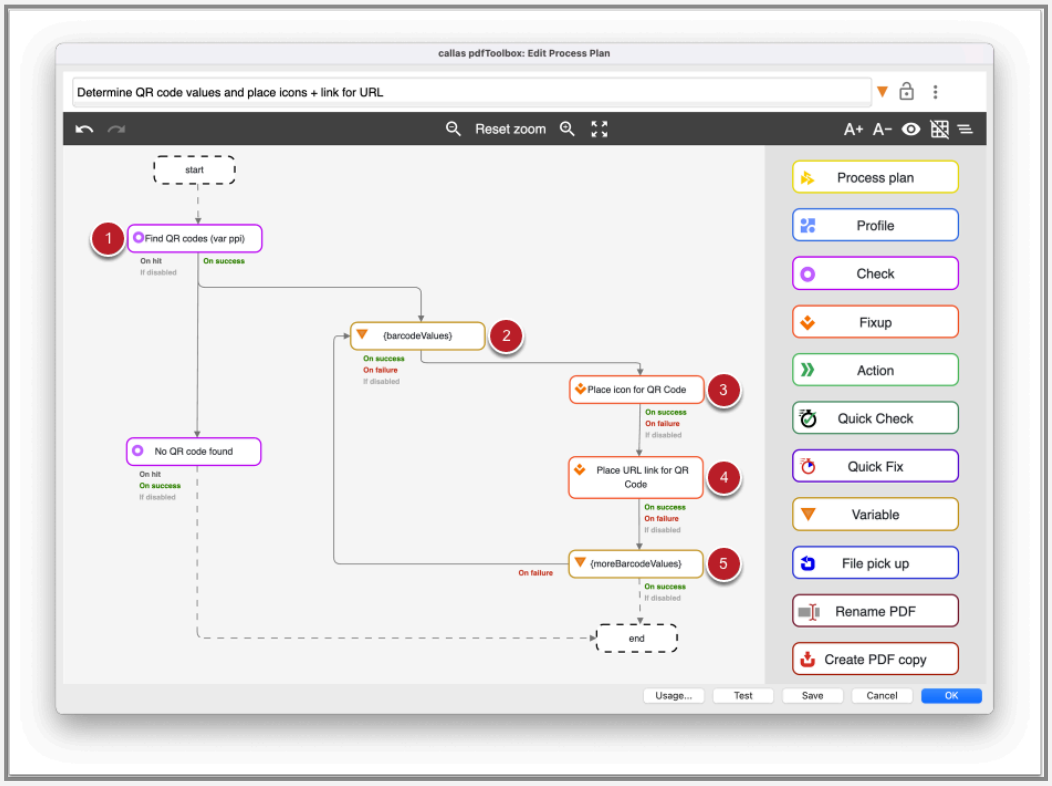
As QR Codes are rather inconvenient in digital publications, the Process Plan "Determine QR Code values and place icons + link for URL" can be used to optimise QR Codes for digital use. The Process Plan first identifies all QR Codes, decodes the destination URLs and then adds a link annotation (hyper-link) and an icon over each QR Code to indicate that the QR Codes are clickable.

You can download a zip package containing the Process Plan, a test file and a "click here" icon.

Optimise QR Codes for digital use.zip

(earliest version with full support is **pdfToolbox 14**)

Structure of the Process Plan

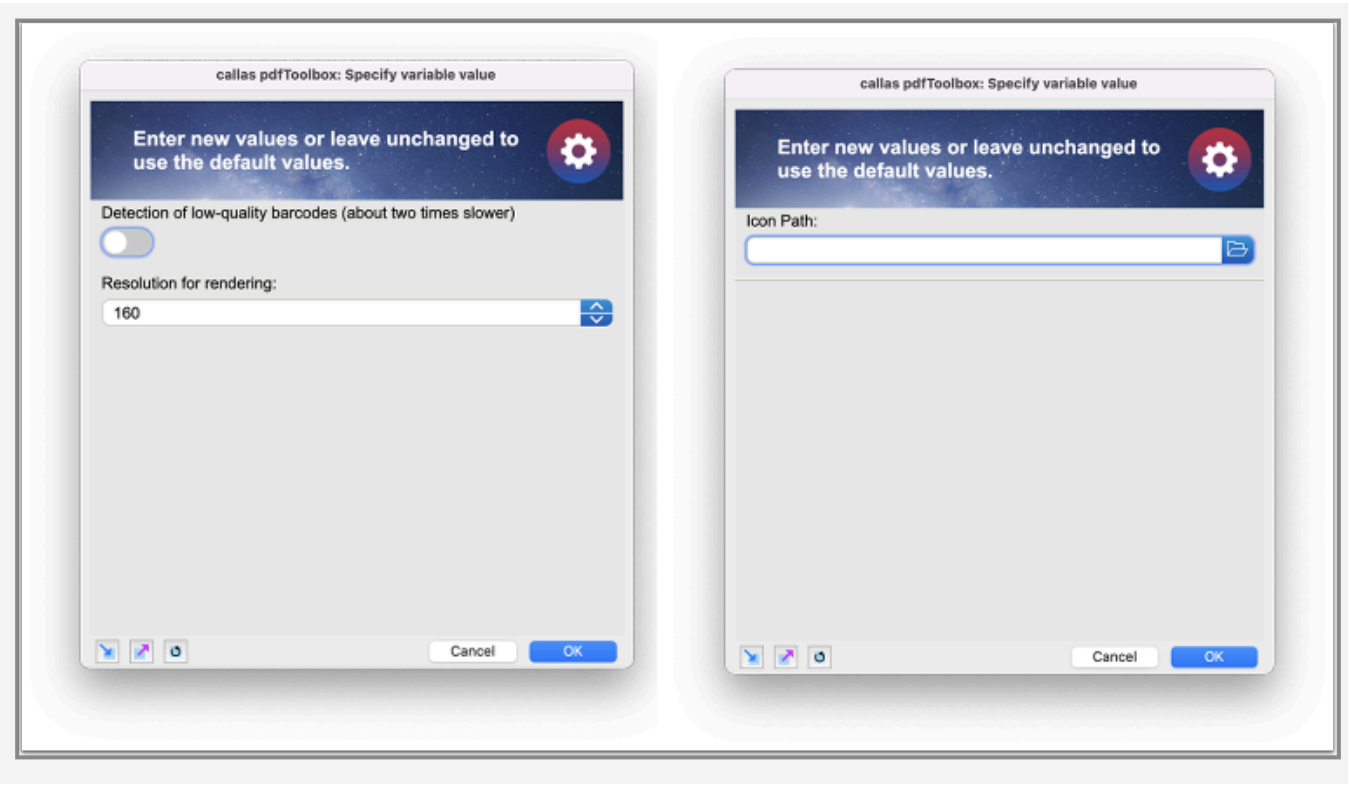


1. **Check "Find Barcodes":** This Check identifies all QR Codes in a PDF document. If no QR Code is found, the Process Plan stops and reports "no QR code found".
2. **Variable step:** Calculates the values for the QR Codes (destination URLs of the QR Codes and position data) so that they can be used in the next steps.
3. **Fixup "Place icon for QR Code":** Uses the "place content" technology to place a custom icon over the QR Codes. All icons will be placed on a new Layer called "Link".
4. **Fixup "Place URL link for QR Code":** Places the destination URLs on top of the QR Codes as a link annotation.
5. **Variable step:** If there are more QR Codes to process, this step will go back to Setp 2.

Ask-at-runtime dialogues

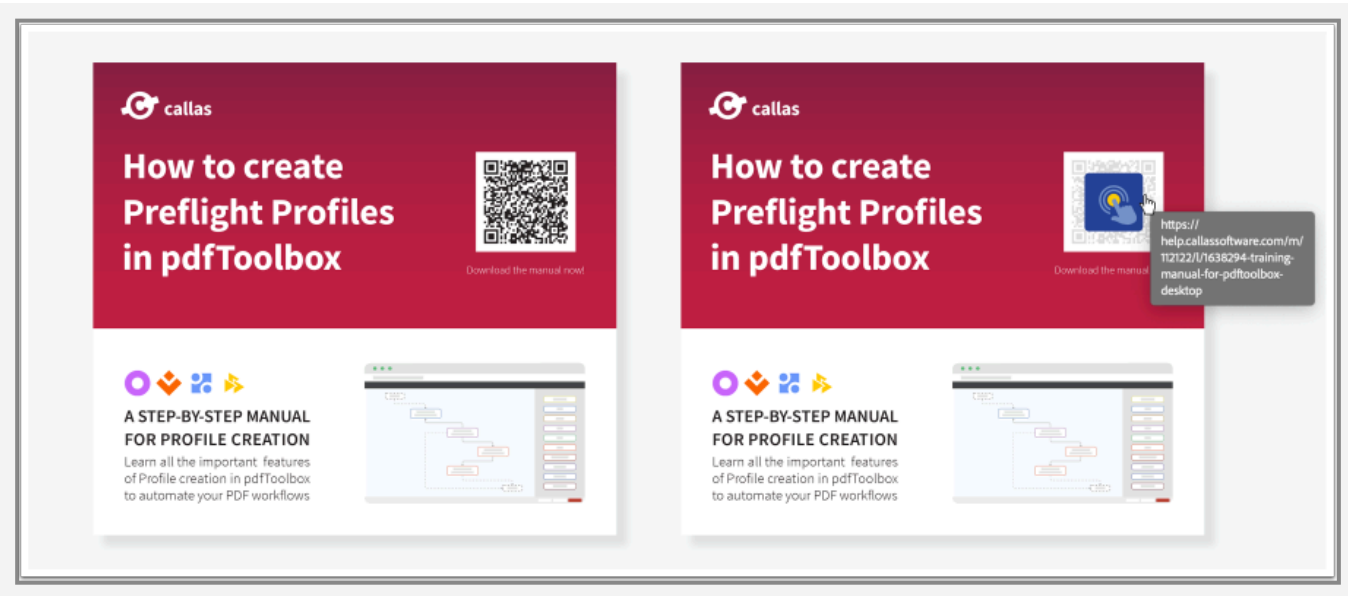
When the Process Plan is executed, two Ask-at-runtime dialogues will appear:

1. The first dialogue allows you to set the rendering resolution (the default of 160 ppi is suitable in most cases) for barcode detection. You can also enable the "Detect low quality barcodes" option (performance will be slower).
2. In the next dialogue you have to specify an absolute path to the icon that will be placed on top of the QR Codes.



Result

The QR Code is now overlayed with the "click-here" icon and a link annotation with the destination URL of the QR Code. When you tap the icon, the linked website will be opend.



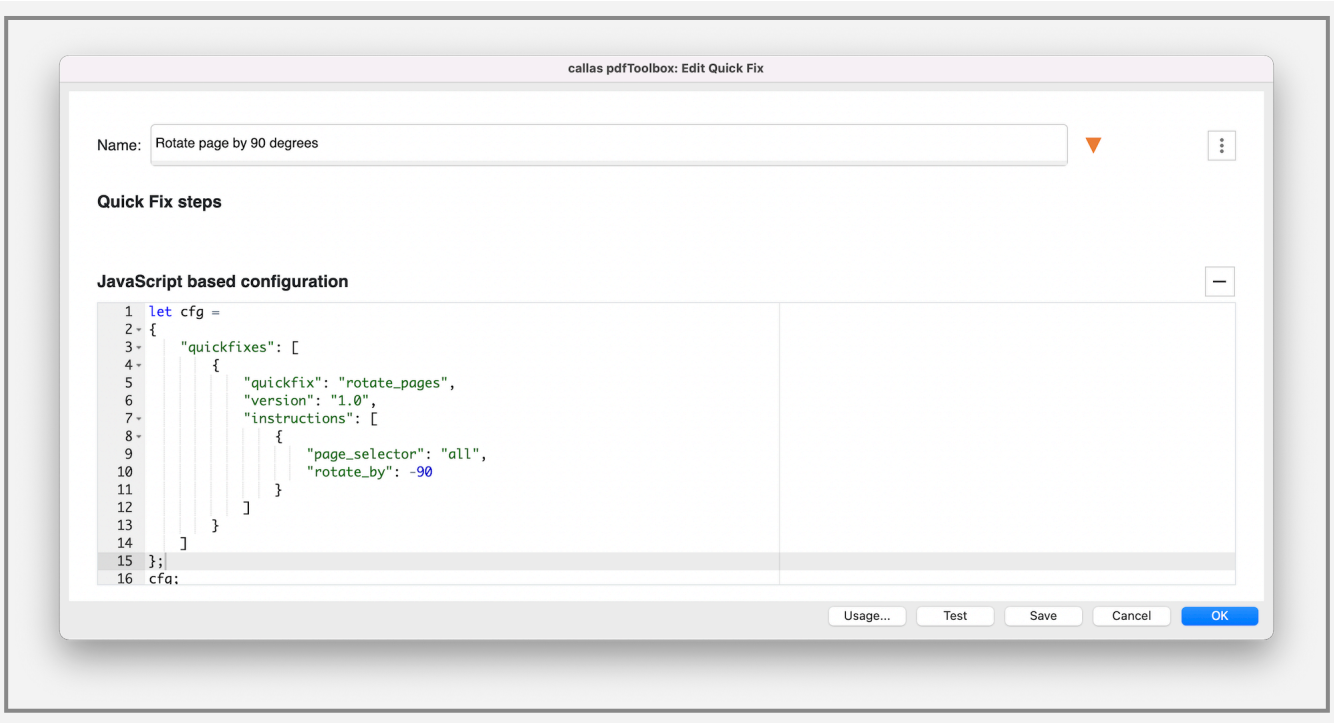
31.7 Process Plan: Single out page objects

Things that can be done with Quick Fix can be done very fast. Sometimes you can only take full advantage of that when you are using JavaScript. The "**JavaScript based configuration**" Quick Fix can be used as follows:

```
let cfg = {
        … Your Quick Fix JSON serialisation
};
cfg; //&lt;- return your Quick Fix object
```

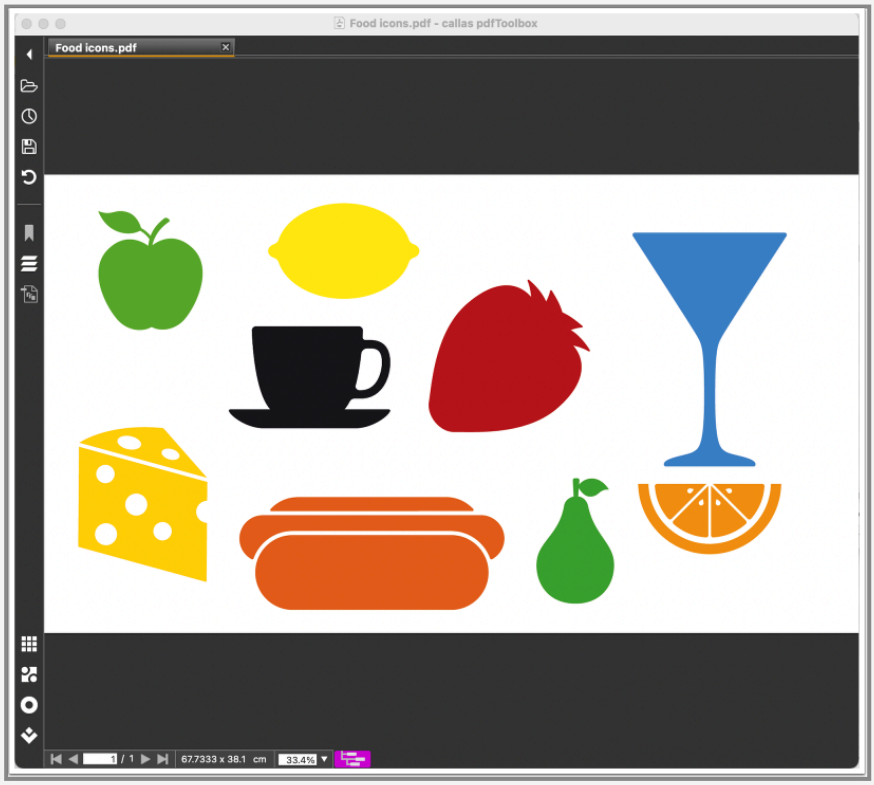
Click to copy

The JSON serialisation for the existing Quick Fixes are documented here.



Example of a JavaScript based QuickFix as Process Plan steps

In this example we have a single page PDF with 10 objects. What we want to do is to split this page so that each page in the result PDF has only one object.



Food_icons.pdf

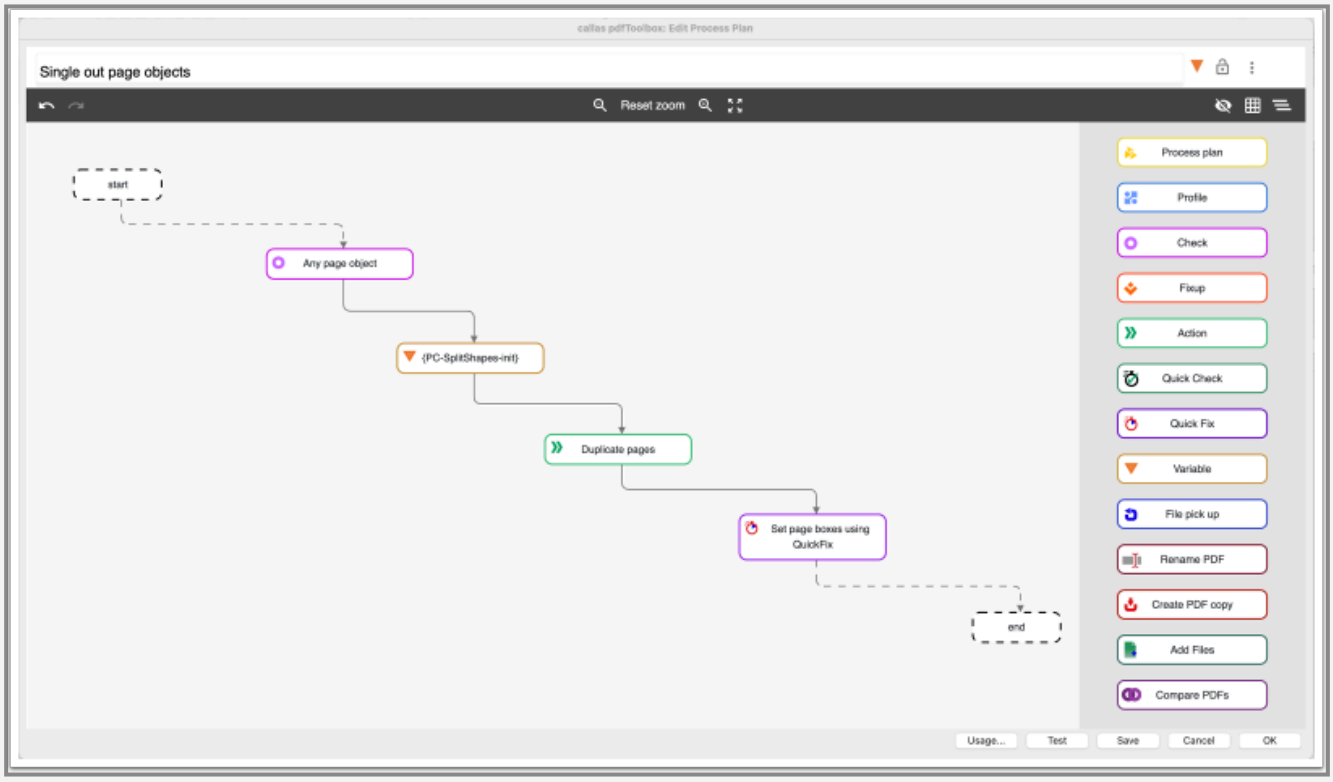
We know that we can set the Crop Box to the dimensions of an object, so what we want to do is to duplicate the first page, set the Crop Box to the dimensions of the first object, then duplicate the first page again for the second object and so forth. We also know that we can determine the dimensions of an object using a Check.

So what we could now do is to create a Check that finds any objects and then loop over the result list in a Process Plan duplicating the pages and setting the Crop Box. We could use a Quick Fix for setting the Crop Box because that is much faster than a Fixup. The problem is that the solution is not really quick because all of the overhead for our Process Plan based loop.

A much smarter way is to first create as many duplicates from the first page as we need (as many objects as we have minus one because we can use the existing page). Then we create a list of Quick Fix instructions, one for each page and for the re-

spective object that should be visible on that page and then we run Quick Fix.

The Process Plan "Single out page objects"



Single_out_page_objects.kfpx

Step 1 to 3

1. Check "Any page object" is a single Check using the property "Is any page object". It creates a result object.

2. Variable / JavaScript "PageObjects-init" initializes app.vars.objects with all hits (all objects) and creates a variable "app.vars.nbhit" with the number of objects minus one, that is the number of pages that we need to create.

```
app.vars.objects = app.doc.result.checks[0].hits;
```

```
app.vars.nbhit = app.vars.objects.length -
1;
```

3. Action "Duplicate pages" duplicates the first page according to what has been defined in app.vars.nbhit.

Step 4 "Set page boxes using Quick Fix"

This is a Quick Fix that uses "JavaScript based configuration". It first creates its own list of instructions and then applies them on the PDF file. Let us see what it does:

```
const margin=5;
```

Specifies a margin around each object. The value is 5 [pt].

```
let instructions= [];
```

Defines an empty array that will contain our instructions (one per object/page).

Then comes a for loop that runs over our app.vars.objects array and sets the Crop Box to the dimensions (plus margin) of the first object on the first page and so forth.

```
for (let i = 0; i < app.vars.ob-
jects.length; i++) {
```

```
  let llx = (app.vars.objects[i].llx -
margin);
```

```
  let lly = (app.vars.objects[i].lly -
margin);
```

```
  let urx = (app.vars.objects[i].urx +
margin);
```

```
  let ury = (app.vars.objects[i].ury +
margin);
```

The variable "page" is used to specify the page for the current set of coordinates. It is used in the "page_selector" object for the instructions below.

```
  let page = (i + 1).toString();
```

```
  instructions.push({
```

```
                    "bottom" : lly,
```

```
                    "from_box_or_ab-
solute" : "absolute",
                    "left" : llx,
                    "page_selector" :
page,
                    "right" : urx,
                    "top" : ury,
                    "unit" : "pt",
                    "when" : "always",
                    "which_box" : "Crop-
Box"
                });
}
```

Now we have created an array "instructions" in the syntax that is expected by the Quick Fix to set page geometry boxes and that has a set of coordinates for the CropBox and a page_selector for the page it should be used on.

We now put this into a full Quick Fix configuration:

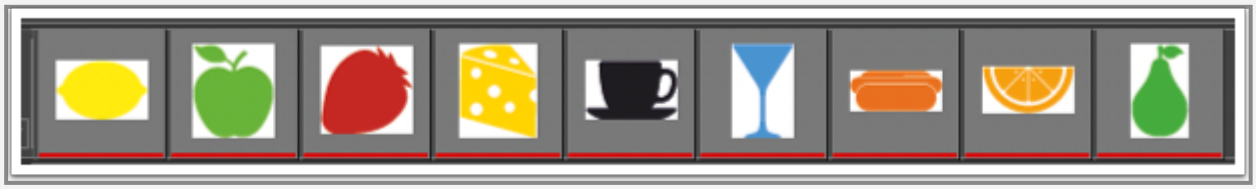
```
let cfg =
{
    "quickfixes" :
    [
        {
            "instructions" : in-
structions,
            "quickfix" :
"set_page_box",
            "version" : "1.0"
        }
    ]
```

```
};
```

And finally we establish that configuration by calling it:

```
cfg;
```

The result



32. callas pdfToolbox CLI (command line interface)

32.1 Introduction to pdfToolbox CLI

Install and activate

To get going with pdfToolbox CLI, the first thing needed is installation and activation. This is explained step by step in the next article.

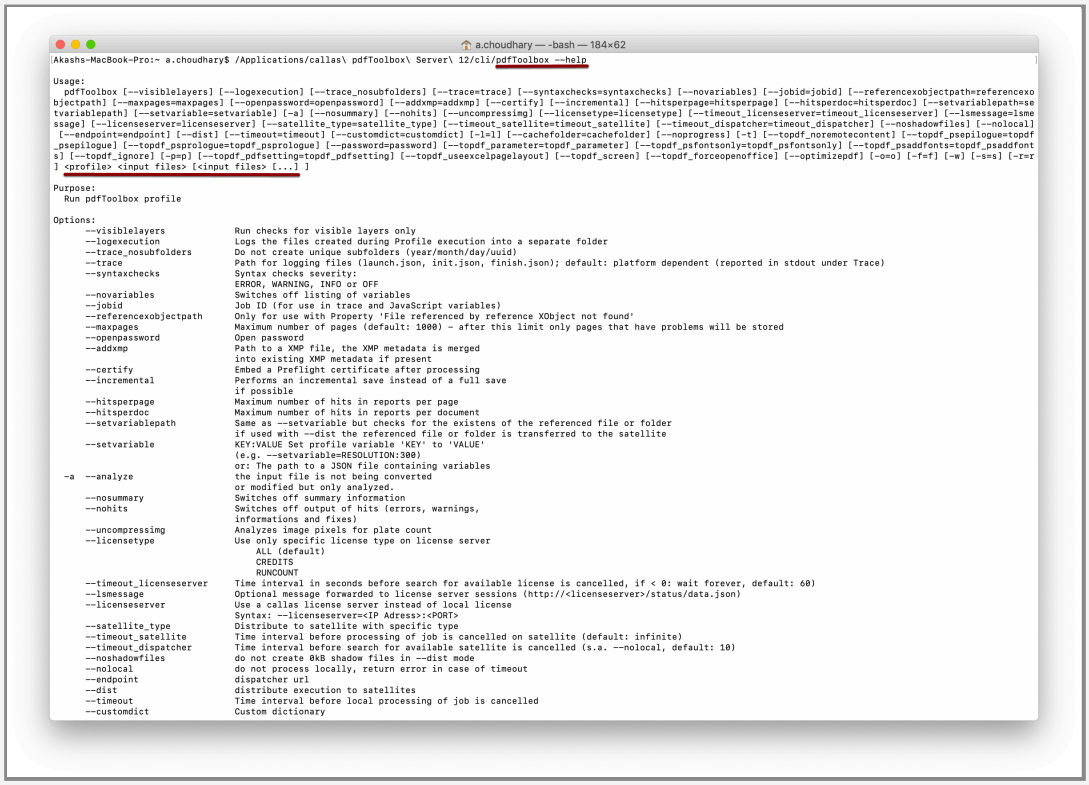
Usage



Watch this short video to discover how to find you way around the pdfToolbox CLI and how the built-in help capabilities can take you a long way toward becoming proficient.

CLI help

The easiest way to find out what you can do with the command-line is to run the --help command.



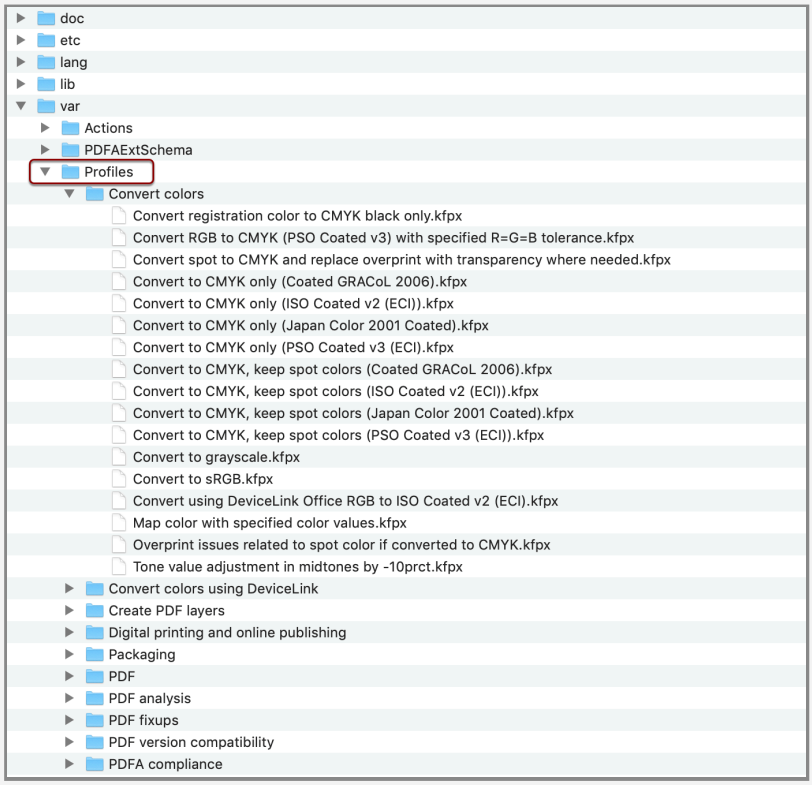
This will provide a long help text with

1. Parameters available when working with Profiles (as shown in the screenshot above)
2. Various "commands" usable with pdfToolbox CLI (not visible in the screenshot above but explains below)

Executing Profiles with pdfToolbox CLI

pdfToolbox uses **Profiles** to perform quality control and/or to fix PDF documents. An example of a simple Profile is: 'Flatten transparency (high resolution).kfpx'
pdfToolbox CLI gets shipped with a set of predefined Profiles stored in logical groups within:

<Application folder>/var/Profiles (shown below).

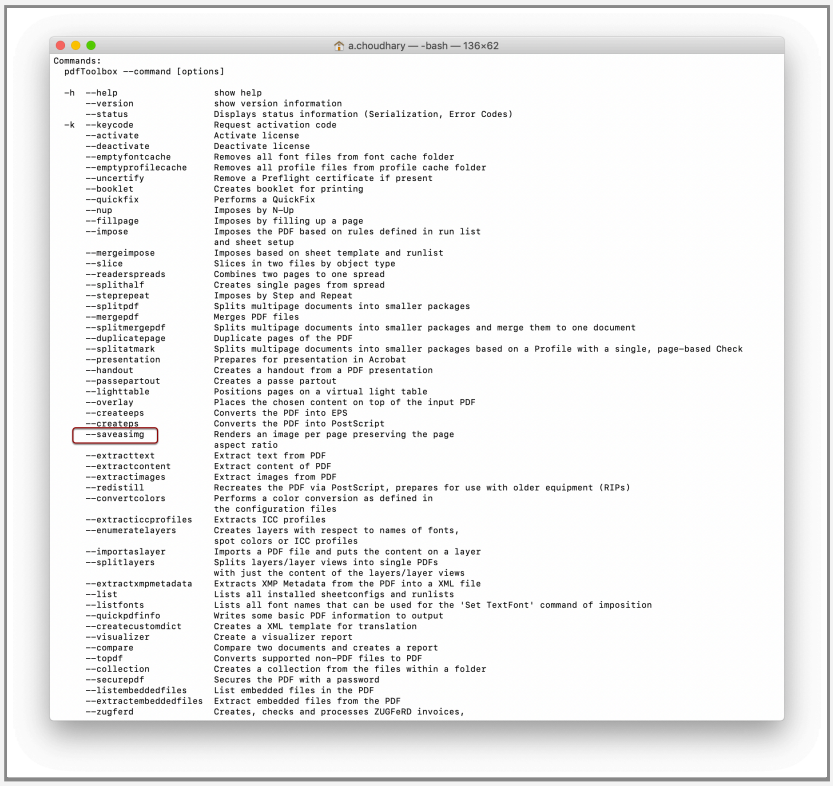


Most of the functionality is only available in Profiles. Given that there are more than 1000 different Check properties (that can even be combined) and several hundred Fixups, there is no way to put this into a command line interface. Profiles will usually convert a source file into a result PDF but when file conversions or extraction of content are performed, then Actions can be used.

The Server/CLI package includes the Desktop application for configuration of Profiles on MacOS and Windows.

Run commands directly (without Profiles)

Another way to process PDF files using pdfToolbox CLI is **running commands**. pdfToolbox --help will also furnish the list of commands that you can execute (as shown below):



Once you find the command you want to work with, drill-down; for example, to know how to save images from pdfToolbox, issue the command:

```
./pdfToolbox --help saveasimg
```

Click to copy

Watch this 6-minute video 'How to use pdfToolbox CLI':

Get in touch

If some necessary information is not provided by this manual or if there are any questions or feedback please contact our support line by using the "Contact Support" form.

Alternatively, you can also send an e-mail to support@callas-software.com.

32.2 Installation and activation of pdfToolbox Server/CLI

pdfToolbox CLI (command line interface) offers a wide range of options to analyze, correct and enhance PDF files as well as impositioning features and color conversion.
For activating the SDK (software developer kit), please visit this page.

System requirements & hardware recommendations

The command line version of pdfToolbox is available for Windows, Mac and Linux.
You can check the following website for the hardware recommendations and the currently supported version of these operating systems:

- System requirements

You can easily test if pdfToolbox CLI is working on your system: Just type pdfToolbox --help in the terminal.

There are 64 bit versions of pdfToolbox CLI available for MacOS, Windows and Linux. pdfToolbox CLI does also run on 64 bit systems if the required 32 bit compatibility packages are available.

Installing the software

Installer

You can download the latest version of pdfToolbox Server/CLI from our website by requesting a trial:

1. pdfToolbox Server/CLI: pdftoolboxserver
2. pdfToolbox CLI: pdftoolboxcli

You can also register on our website for easy access to all pdfToolbox download links. Once registered and logged in, you can access the list of download links:

1. pdfToolbox Server/CLI: pdftoolboxserver
2. pdfToolbox CLI: pdftoolboxcli

MacOS/Windows

To install the software start the pdfToolbox Server installer. The installation program will then take you through the necessary steps.

Linux

Extract all files from the archive to a destination folder of your choice.

Additional information is provided in <pdfToolbox CLI directory>/ReadMe.txt

NOTE: A new major version of the software should always be installed in a new directory and not in the directory of the previous version.

Request activation

Before callas pdfToolbox CLI can be used, the software has to be activated. Open a terminal window and change to your pdfToolbox CLI installation directory.

Request an activation code for full version

If you have bought a license for callas pdfToolbox Server/CLI you can active the full version with your license key that can be found on your License.pdf:

```
pdfToolbox --keycode [--aws] <name> <company> <licenseCode>
```

Click to copy

Request an activation code for trial version

If you just want to try the product, you can request an activation code for a free 14-day trial:

```
pdfToolbox --keycode  [--aws] <name> <company> trial

Example:

pdfToolbox --keycode "Demo Name" "callas software" trial
```

Click to copy

Parameters

name	Name of licensee (e.g. "Registered User")
company	Name of company (e.g. "User's company")
licencecode	Licence key obtained from the registration card or the License.pdf provided by callas or the reseller.
trial	To make a request for a trial version, please use the keyword "*trial*" (for a pdfToolbox Server/CLI trial version)
aws	For installation on Amazon Web Services (using Windows, Linux 32bit and 64bit)

The textual output of --keycode has to be send via e-mail to the e-mail address named in the text in order to receive an activation code from the registration server.

To make a request for a trial version, please use the keyword "trial" (for a pdfToolbox trial version) for this parameter.

Activating pdfToolbox CLI

After receiving the automatic reply e-mail from the activation server, save the attached 'activation file' to the file system. Then use the following command:

```
pdfToolbox --activate <activation file>
```

If no response is received or in the event of an error, please contact support@callassoftware.com to determine the exact cause.

Parameters

activation file	Full path to Activation PDF



It is necessary to activate the received license file to get a permantent valid license file.

The Activation.pdf (or the content of the e-mail) can only be used for activation for 48 hours. After this time frame, a new Activation.pdf has to be requested from the activation server.

The activated license file will be stored in the user-preferences when the normal activation (command above) is used.

To create an activated license file at a custom location, just use the following command:

```
pdfToolbox --activate <licence file> -o=<path to result folder/License.txt>
```

pdfToolbox CLI is searching for the license file at various folders:

- user-preferences-folder of actual user
- next to the pdfToolbox CLI binary
- cachefolder (if set)
- user-preferences-folder for all users (shared)

When using UNIX-based-systems the environment variable CALLAS_SYSTEM_PREFERENCES the path of the standard */usr/share/callas software/callas pdfToolbox CLI* can be changed:

CALLAS_SYSTEM_PREFERENCES=tmp

would result in the searchpath: */tmp/callas software/callas Toolbox CLI*

It is highly recommended to use the option --cachefolder instead.

Time-limited trial version

After requesting and entering a trial activation code, pdfToolbox CLI can be tested without any restrictions. When the evaluation period has expired, processing PDF files will no longer be possible until you request and enter a new activation code.

Activation using the Standalone application

Using Windows and MacOS, also the activation dialog of the Standalone Application can be used for requesting an activation as well as using a Keycode or just for a trial version. Also the activation itself can be done using that Interface.

All activations (for Desktop, Server as well as for the DeviceLink Addons) can be done using this dialog.

Deactivate pdfToolbox using the CLI

As the activation (and the resulting license file) is bound to the hardware. It is necessary to deactivate a license on one machine before an activation takes place on the new machine.

```
pdfToolbox --deactivate <activation code>
```

activation code	Unique identifier for each license

The respective license will be removed from the system

To complete the deactivation, the output of the command has to be sent manually to the activation server by e-mail.

The activation code for all license are listed using the status command:

```
pdfToolbox --status
```

Deactivation using the Standalone application

Similar to the deactivation using the CLI, the Desktop on Windows and MacOS can be used for deactivation.

The selected license will be removed from the system as well and the necessary e-mail to the activation server will be sent automatically.

32.3 Hints and troubleshooting & Displaying program information

Displaying program information

Display program version

```
pdfToolbox --version
```

will display the currently used version of pdfToolbox CLI.

Display usage information

```
pdfToolbox --help
```

will give you a complete overview about all available commands for processing.

```
pdfToolbox --help <command>
```

will give you an overview about all available options for the command.

Display status

```
pdfToolbox --status
```

will inform you about the current license state as well as the possible return and reason codes (see *"Results"*).

Hints and troubleshooting

Ensure sufficient free disk space

To ensure stable processing, it is recommended to have at least 4 times of the input file size of processed files available for intermediate file system storage (e.g. /tmp on Unix and similar on other systems).

Avoid stopping workflows on Windows

On Windows, you can prevent your workflow from stopping in case of a pdfToolbox CLI crash by setting the following registry entry:

```
HKEY_LOCAL_MACHINE\
SYSTEM\
CurrentControlSet\
Control\
Windows\
ErrorMode
```

If ErrorMode is set to "2", crash dialogs will be suppressed. For further details, see: http://support.microsoft.com/kb/128642/en-us?fr=1

Limiting the maximum memory used by pdfToolbox

Using Linux, you can limit the amount of memory used by a single process by an additional parameter:

```
--maxmemory=<max. memory in MB>
```

Processing will stop and result in an error if memory is exceeded.

Limitation of concurrent processes

If the StdOut of the command line indicates "Maximum number of parallel processes allowed by license already running; waiting until one of the current processes terminates" (before pdfToolbox 10: "Waiting for CPU"), the maximum number of parallel processes is reached. A default Server/CLI license allows up to 8 parallel processes. To run more than 8 processes simultaneously, you should look at callas License Server as the better option. You can find more info here.

Performance enhancement

If you want to enhance the performance of your pdfToolbox CLI processes, please keep in mind the following rules:

• For analysis, you can limit processing to a certain page range (see "Only process certain pages").

• Rather remove fixups from a profile only intended for analysis than using --analyze (e.g. when using --analyse, initialization of ICC profiles for color conversion fixups still takes place).

• Fixups containing an "Apply to" option need more processing time if this option is set to something else but "None", since an analysis of the file contents is required before the fixup can be executed.

• If you are using any font embedding fixups, your system font folder will be scanned unless defined otherwise in the fixup configuration. A font cache will be created to improve the performance time, but still it might be useful to remove fonts that are not needed from this directory.

• Keep in mind that the option --uncompressimg (see "Analyze image data") will uncompress images and analyze every single pixel, which may take a long time for some files.

• Creation of XML or PDF reports takes less time than the XSLT option

(see "Report types").

• Creation of reports takes additional time – even if a profile contains only fixups, an analysis will be executed for gathering report information.

Optimization of needed installation space

To reduce the space needed by the installation of pdfToolbox, it is possible to delete some subfolders of the CLI component (in subfolder /cli) if their respective functions are not needed in the individual use case.

To avoid processing errors or unexpected behaviour of pdfToolbox any modification should be done well-considered.

etc/Actions	If no Arrange action is used
etc/APDFL	If no font embedding or PDF/A conversion is used (or if font situation is clear)
etc/Backgrounds	If no layer/image mask report is used
etc/Certify	If no preflight certification is used
etc/ColorConversion	If no color conversion is used
etc/HtmlConverter	If no PDF report based on HTML template is used
etc/Inventory	If no inventory report is used
etc/PDFOfficeTool	If no Office-files are processed
etc/PDFPSTool	If no PostScript-files are processed
etc/Reports	If no PDF/A-HTML Report or ZUGFeRD is used
etc/UnpackTool	If no archive files are processed
etc/Visualizer	If no Comparison is used

Install basic fonts on Linux

As most Linux distributions are delivered without or with very few fonts, it is advisable to install at least a basic set of fonts.

We recommend installing at least the the fontconfig package ant the MS core fonts, which is e.g. done on Debian-based distributions by the following commands:

```
sudo apt -y install fontconfig
```

```
sudo apt install ttf-mscorefonts-in-staller
```

For other distributions, please consult the respective documentation.

Possible errors within the Server-UI communication

The communication between the Server-UI and the underlying Server process (which observes the folders) takes place by network communication ports using SOAP.

When the Server-UI shows a warning for a connected Server like "Error 28", it indicates that there were errors either during the connection to the Server process itself or that the Server process has stopped working.

You can try to reconnect to the Server in the UI or try to start the local Server process again by the "Start Server" option.



How to use pdfToolbox CLI:

32.4 Processing

Profiles

Optional parameters are marked with [ ].

Run a profile:

```
pdfToolbox [-r=r] [-l=l] [-p=p] [--hitsperpage=hitsperpage]
[--hitsperdoc=hitsperdoc] [--setvariable=setvariable] [-t]
[--cachefolder=cachefolder] [-o=o] [-f=f] [--analyze] [-w]
[--incremental] [--noprogress] [--nosummary] [--nohits]
[--uncompressimg] [--password] [--timeout=timeout] [-s=s]
<profile> <input file> [<input file> [...] ]
```

Run an action:

```
pdfToolbox <action> [--cachefolder=cachefolder] [-o=o] [-f=f ] [-s=s]
[--incremental] [-w] [-t] {action specific parameters} <input file> [<input file>
[...] ]
```

On Unix systems, if the environment variable TMPDIR is defined, its value is used instead of the default /tmp directory for storing
temporary files.

Processing files to PDF

pdfToolbox CLI is able to convert common file formats directly to PDF. To simply convert a file format to PDF, the command `--topdf` can be used. If a Profile or an Action is used, the file format is automatically converted to PDF before processing.

For more information have a look at the following article: Requirements for conversions to PDF.

- Using Linux, Office file conversion requires an OpenOffice or LibreOffice installation on the respective system.

Conversion options for Office files

The following switches do only apply for Office files and must be set in combination with a Profile, Action or the `--topdf` command.

OpenOffice

```
--topdf_forceopenoffice
```

When defined, Microsoft Office files are processed with OpenOffice.

Start page

```
--topdf_startpage
```

Defines the first page in the given Office document which should be converted to PDF. This is set to 1 by default.

End page

```
--topdf_endpage
```

Defines the last page in the given Office document which should be converted to PDF. This is set to the last page by default.

Create PDF for screen

```
--topdf_screen
```

The images of the created PDF file have a lower quality, the resulting file size is smaller. Comments/Annotations (e.g. for tracking of changes) will be included.

Excel-Sheets without removing white space

```
--topdf_useexcelpagelayout
```

Uses the Excel page layout for creating the PDF, white space will not be removed.

Special handling for MS Office files

```
--topdf_parameter=[ShowHiddenColumns|ShrinkToFit|PrintQualityAndComments|Update-
ChangedFields|DoNotHideOffice|NoMemoryOptimization]
```

Parameters

ShowHiddenColumns	Show columns which are not visible due to small width or other settings. (for Office files processed with MS Excel only). Note: When --topdf_useexcelpagelayout is used, this parameter will not be respected.
ShrinkToFit	Shrinks the content of a cell so that the content fits inside. (for Office files processed with MS Excel only). Note: When --topdf_useexcelpagelayout is used, this parameter will not be respected.
PrintQualityAndComments	Images will have bigger resolutions and comments/annotations (e.g. from tracking changes) will be included as well (for Office files processed with MS Word only; can not be combined with --topdf_print).
UpdateChangedFields	Updates the changed fields like automatically generated Table of contents. By default, such fields are not updated (for Office files processed with MS Word only)
DoNotHideOffice	Parameter to avoid hiding Office application for a successful conversion to PDF
NoMemoryOptimization	Parameter to to disable internal memory optimization during processing for improved performance

Logging of dialogs in defined log file

```
--topdf_guiactionslog=<path>
```

Parameters

path	Path to folder or logfile.

All dialogs occuring during processing the office file will be logged within this file. See internet page (listed above) for further information about handling of dialogs from office applications.

Conversion options for Postscript files

The following switches do only apply for Postscript files.

Define folders used for font search

```
--topdf_psaddfonts=<path>
```

Parameters

path	Path to addtional font folder for PS to PDF conversion which will be additionally used for font search.

System font folders will also be used.

```
--topdf_psfontsonly=<path>
```

Parameters

path	Path to font folder for PS to PDF conversion, no usage of system fonts will take place.

```
--topdf_pdfsetting=<path>
```

Parameters

path	Path to PDF settings file to be used for conversion of PS and EPS files only, must be a Distiller .joboptions file

Define a prolgoue and/or an epilogue file

```
--topdf_psprologue=<path>
```

Parameters

path	Path to an addtional prologue file, which is taken into account when converting files to PDF, must be a .ps file with prologue content

```
--topdf_psepilogue=<path>
```

Parameters

path	Path to an addtional epilogue file, which is taken into account when converting files to PDF, must be a .ps file with epilogue content

32.5 Using Profiles

Provided profiles

pdfToolbox CLI gets shipped with a set of predefined profiles stored in logical groups within <Application folder>/var/Profiles (All Profiles are mentioned here):

Acrobat PDF version compatibility
Profiles for checking the compatibility of a file to a specified Acrobat version

Convert colors
Profiles to perform color conversions
To perform a color conversion using the DeviceLink profiles available as payable option of pdfToolbox, you have to have a valid license for the callas DeviceLink Add-on. The list of provided profiles can be found in section "DeviceLink profiles for Desktop" and "DeviceLink profiles for Server/CLI" on the callas website.
You can use own DeviceLink profiles without the Add-on license of course.

Create PDF layers
Profiles to put specified objects to different layers

Digital printing and online publishing
Profiles to optimize PDF files for digital printing or online publishing

Packaging
Profiles based on the recommendations of the Ghent PDF Workgroup for Packaging

PDF
Profile for checking PDF 2.0 related issues

PDF analysis
Profiles for general analysis of the PDF and its objects (e.g. number of
plates, image resolution etc.)

PDF fixups
Profiles for modifying the contents of a PDF (e.g. downsampling of
images, embedding of fonts etc.)

PDFA compliance
Profiles for verifying compliancy with and converting to PDF/A

PDFE compliance
Profiles for verifying compliancy with and converting to PDF/E

PDFUA compliance
Profiles for verifying compliancy with and converting to PDF/UA

PDFVT compliance
Profiles for verifying compliancy with and converting to PDF/VT

PDFX compliance
Profiles for verifying compliancy with and converting to PDF/X

Prepress
Profiles based on the recommendations of the Ghent PDF Workgroup that are based on PDF/X and specify further requirements for various printing conditions. For more information see: www.gwg.org

Place content
Sample Profiles for placing various content (like text, barcodes or content based on a HTML template) on a page. Please use the Desktop version for configuration of custom Profiles.

Preflight Certificate
Profile to validate the existence and validity of a Preflight Certificate in the PDF document.

ProcessPlans
Processplans as examples how to use the dynamically controlled combination
of Profiles, Fixups, Checks and Actions.

Processing steps
Profiles for checking compliancy with the Processing Steps standard

In order to generate, modify or view pdToolbox profiles you need the Desktop version of pdfToolbox.
pdfToolbox CLI is able to work with all profiles set up with pdfToolbox Plug-In or Standalone. Profiles delivered with pdfToolbox CLI may be edited as well. In order to use a certain profile you will have to export it as a profile package (*.kfpx -file). For further details on setting up a profile see the chapter "callas pdfToolbox Basics" in this manual.

General profile options

```
pdfToolbox [-r=r] [-l=l] [-p=p] [-o=o] [-f=f] [-w] [-t] [--hitsperpage=hitsper-
page] [--hitsperdoc=hitsperdoc] [--setvariable=setvariable] [--cachefold-
er=cachefolder] [-s=s] [--incremental] [--analyze] [--noprogress] [--nosummary] [--
nohits] [--uncompressimg] [--certify] [--timeout=timeout]
<profile> <input file> [<input file> [...] ]
```

Disable fixups

```
--analyze
```

Disable execution of fixups defined in the used profile. Only defined
checks are carried out.

Display

```
--noprogress
```

Do not show progress information in Standard output (stdout) during processing.

```
--nohits
```

Do not show detailed hit information in Standard output (stdout) during processing.

```
--nosummary
```

Do not show summary of hits and fixups in Standard output (stdout) at the end of processing.

Analyze image data

```
--uncompressimg
```

Images get uncompressed during the checking to allow a proper calculation of used colorants. This option may increase the processing time depending on the amount of images contained in the processed PDFs.

Certify

```
--certify
```

Embed a Preflight certificate (also known as "Audit Trail") after processing.

Fast VDP mode

```
--fastvdpmode
```

Click to copy
The "Fast VDP mode" only works for files representing variable data. Is it activated, XObjects are processed only once ignoring context. Read this article for further information.

Using dynamic profiles

A pdfToolbox profile may contain variable values which can be exchanged during runtime. For more details on setting up those dynamic profiles please see section "Use of kfpx Profiles" in the callas pdfEngine Reference.

```
--listvariables
```

Lists all variables defined in a kfpx profile.

```
--setvariable=<Key>:<Value>
```

Set variable 'KEY' to 'VALUE' in the provided profile.

```
--setvariable=<JSON>
```

Sets all variables contained in the JSON file in the provided profile. JSON file can be exported when processing files in the Desktop version in the variables ask-at-runtime dialog. (pdfToolbox 9.3 and later)

```
--novariables
```

Switches off listing of variables.

To export variables used in a Profile to a JSON file (together with more details about the profile), please use Enumerate Profiles.

Parameters

Key	identifier used in dynamic profile
Value	value to be set for this key
JSON	path to a JSON file containing variables

If you want to use values containing spaces, you either have to put the string into quotes or escape the space character (e.g. "Spot 300 U" or Spot\ 300\ U).

Example

```
pdfToolbox --listvariables <profile>
```

```
pdfToolbox --setvariable=RESOLUTION:300
```

The following characters need to be escaped with \:

'!?*$[]\()|/]

```
--setvariablepath=<path>
```

Same as --setvariable but checks for the existence of the referenced file or folder.

If used with --dist (Distributed processing) the referenced file or folder is transferred to the Satellite

32.6 General command line options

Run a profile:

```
pdfToolbox [-w] [-t] [-o=o] [-f=f] [-s=s] [--incremental] [-p=p] [--hitsper-
page=hitsperpage] [--hitsperdoc=hitsperdoc] [--setvariable=setvariable] [-r=r] [-
l=l] [--analyze] [--cachefolder=cachefolder] [--noprogress] [--nosummary] [--no-
hits] [--uncompressimg] [--timeout=timeout] <profile> <input file> [<input file>
[...] ]
```

Run an action:

```
pdfToolbox <action> [--cachefolder=cachefolder] [-o=o] [-f=f] [-s=s] [--incremen-
tal] [-w] [-t] {action specific parameters} <input file> [<input file> [...] ]
```

Only process certain pages

```
-p --pagerange=<firstpage>[-<lastpage>]
```

Allows to define a pagerange to process when perfoming the following tasks:

- Running a profile that contains only checks
- Running the action --createeps
- Running the action --saveasimg

Running a profile with the option --analyze also honors this option.

Parameters

first page	first page to be processed
last page	last page to be processed

Setting the cache folder

```
--cachefolder=<path>
```

Sets the cache folder path. This is set by default to:

System	path
Windows:	%AppData%\callas software\callas pdfToolbox CLI <version>
MacOS:	/Users/<USERNAME>/Library/Preferences/ callas software/callas pdfToolbox CLI <version>
Linux:	<home directory as defined in /etc/passwd>/.callas software/callas pdfToolbox CLI <version>

This option is mandatory when running the CLI as a user without a home directory.

The cachefolder should have sufficient read/write permissions for the executing user. Especially when the license file is only stored in the cachefolder, this file should be readable by other users.

Parameters

path	absolute path to custom cache folder

Empty the Profile cache

```
--emptyprofilecache [--cachefolder=<path>]
```

For performance reasons, bigger profiles are unpacked during first usage and containing ICC-profiles and config files are stored in a local profile cache. This command deletes this profile cache.

Empty the font cache

```
--emptyfontcache [--cachefolder=<path>]
```

For performance reasons, fonts found on the respective system are catalogized in an internal font cache. This command deletes this font cache.

Incremental saving

```
--incremental
```

Allows to modify the input file, only writing the changes to the original PDF. This can increase the speed significantly since pdfToolbox CLI does not need to create a new copy of the file.

When using the action --impose together with option --pre-processingprofile

or the action --mergeimpose, the incremental saving option can be used in conjunction with --outputfile or --outputfolder to speed up the overall processing time, because all file modifications during these multi-step processes are then performed on a single temporary PDF file.

PDF structure and font optimization

```
--nooptimization
```

The internal PDF structure and fonts are not optimized when saving the PDF file.

Enable processing PDF with password protection for editing and printing

```
--password=<password>
```

To enable Profile-processing of a password-protected PDF. This option can be used for PDFs with restrictions for editing and printing, which will become unsecured. The resulting PDF will have no security setting.

The entered password will be visible and may be grabbed or logged by other processes on the machine.

Parameters

password	Password of the PDF (avoiding editing or printing of the PDF)

Enable processing PDF with password protection for opening

```
--openpassword=<password to open PDF>
```

To enable Profile-processing of an "open"-password-protected PDF. With this option only PDFs with restrictions for opening can be unsecured. The resulting PDF will have no security setting.
Available since pdfToolbox 10.2.

The entered password will be visible and may be grabbed or logged by other processes on the machine.

Parameters

openpassword	"Open" password of the PDF

Define the suffix

```
-s --suffix=<suffix>
```

Defines the suffix that will be appended to the resulting file(s) filename.

The defined suffix is added before the files type suffix (e.g. Output.pdf will become Output_PDFA.pdf when using --suffix=_PDFA).

Parameters

suffix	string to append to filename

Font folders

If a font is not embedded and an embedding is required by a profile, pdfToolbox CLI will search the system font directories in order to find the needed font file, which are:

System	Folder
Windows	• C:\Windows\Fonts
MacOS	• /Users/<user>/Library/Fonts • /Library/Fonts • /System/Library/Fonts
Linux	• /usr/share/fonts • /usr/lib/X11/fonts • /usr/local/X11R6/lib/X11/fonts • /<user home>/.fonts

Additionally the font folder installed together with pdfToolbox CLI will be searched.
This folder lies next to the executable in "<callas pdfToolbox CLI directory>\etc\APDFL\Resource\Font".

ICC profiles folders

The following folders are searched for required ICC-profiles, unless they are already contained in the .kfpx-profile already.

These folders lies next to the executable in:

- "<callas pdfToolbox CLI directory>\etc\ICC profiles"
- "<callas pdfToolbox CLI directory>\etc\APDFL\Resource\Color\Profiles"

Some system folders for colors are searched addtionally:

MacOS:	\Library\Application Support\Adobe\Color
Windows:	\Windows\system32\spool\drivers\color

Set a processing timeout

```
--timeout=<seconds>
```

Sets the maximum processing time in seconds. If the process exceeds this duration, the execution process will be killed and the processing will result in an error.

Set path to referenced XObjects for PDF/X-5g, PDF/X-5pg and PDF/VT-2

```
--referencexobjectpath=<path to folder with resources>
```

Referenced, external resources of the PDF are searched in the same folder as the input PDF by default. To define another folder, this parameter can be used.

32.7 Converting office documents to PDF or PDF/A

Input files from Office applications

The CLI component is able to convert common file formats from Office applications directly to PDF/A. For more information and a list of supported applications and files have a look at: http://www.callassoftware.com/goto/apl_ENU_topdf (for pdfaPilot)
http://www.callassoftware.com/goto/tbx_ENU_topdf (for pdfToolbox)

- Note: *Since pdfToolbox11/pdfaPilot 9/pdfEngine 11 onwards in the 64bit variant on Windows, the Libre/OpenOffice has to be 64bit as well (and 32bit when the 32bit variant is installed).*

Forced use of OpenOffice

```
--topdf_forceopenoffice
```

When defined, Microsoft Office files are processed with OpenOffice (or LibreOffice if installed).

Create PDF for screen display

```
--topdf_screen
```

Images from Office documents will have lower quality, result file will be smaller.
Default in callas pdfaPilot.

Create PDF for print

```
--topdf_print
```

The PDF will be created with image resolution sufficient for printing, thus leading to larger files.
Default in callas pdfToolbox.

Excel-Sheets without removing white space

```
--topdf_useexcelpagelayout
```

Use Excel page layout, white space will not be removed.

Special handling for MS Office files

```
--topdf_parameter=[ShowHiddenColumns|ShrinkToFit|PrintQualityAndComments|Update-
ChangedFields|DoNotHideOffice|NoMemoryOptimization]
```

Special parameters to achieve some special layouts for MS Office files.

Parameters

ShowHiddenColumns	Show columns which are not visible due to small width or other settings. (for Office files processed with MS Excel only). Note: When --topdf_useexcelpagelayout is used, this parameter will not be respected.
ShrinkToFit	Shrinks the content of a cell so that the content fits inside. (for Office files processed with MS Excel only). Note: When --topdf_useexcel-pagelayout is used, this parameter will not be respected.
PrintQualityAndComments	Images will have bigger resolu-tions and comments/annotations (e.g. from tracking changes) will be included as well (for Office files processed with MS Word only; can not be combined with --topdf_print).
UpdateChangedFields	Updates the changed fields like automatically generated Table of contents. By default, such fields are not updated (for Office files processed with MS Word only)

DoNotHideOffice	Parameter to avoid hiding Office application for a successful conversion to PDF
NoMemoryOptimization	Parameter to to disable internal memory optimization during processing for improved performance

Logging of dialogs in defined log file

```
--topdf_guiactionslog=<path>
```

Parameters

path Path to folder or logfile.

All dialogs occuring during processing the office file will be logged within this file.

- Note: See the "to PDF" internet page (listed above) for further information about handling of dialogs from MS Office applications.

32.8 Additional command line options and response files

Define the overwrite mode

```
--overwrite
```

New files override existing files with the same name (applies to report files and to created PDF files).

Set the result path

- Note: If neither an output path nor an output folder is defined, any result will be created next to the input file (default: input file name with suffix _PDFA or _NOPDFA, will be indexed if necessary).
- Note: The use of --outputfile together with --outputfolder is not sup-ported within one CLI call.

Path to output file

```
-o --outputfile=<path>
```

Defines the absolute path of the destination file. The parent folder must exist.

- Consult section "Results" to see if a new file was created. When running a profile containing checks only, no new output file is created.

Parameters

path	absolute path to output file

Set an output folder

```
-f --outputfolder=<path>
```

Defines an absolute path to a folder where the resulting files of an execution are stored.

If neither an output path nor an output folder is defined any result will be created next to the input file (filename will be indexed if necessary).

The use of --outputfile together with --outputfolder is not supported within one CLI call.

Parameters

path	absolute path to output folder Out--put file

Timestamp

```
--timestamp
```

Every line in the Standard output (stdout) is prefixed using a time stamp.

Using response files

To keep the command line call structured and straightforward, pdfToolbox CLI supports the usage of response files. These offer the possibility to define each command line switch line by line and also add some comments.
Please make sure the response file is saved as UTF-8 (without BOM).

Example
Response file variables.rsp:

```
###################
# Check for specified resolution and change name of Check
#
--setvariable=RESOLUTION:300
--setvariable=CHECKNAME:Image resolution for images
#
```

```
###################
# EOF
```

Using different responsefiles enables the easy definition of own, localized sets of strings for names of Profiles, Fixups and Checks as well as different settings for processing PDFs for different output environments.

Command line call:

```
pdfToolbox @<absolute path to "variables.rsp"> <profile> <PDF file>
```

If command line arguments or options with space characters are used in the response file, they shall not be escaped or set in quotes as normally used in CLI commands, where it would e.g. look like: `--setvariable=CHECKNAME:"Image resolution for images"`

`"var/Profiles/PDF analysis/List page objects, grouped by type of object.kfpx"`

or

`var/Profiles/PDF\ analysis/List\ page\ objects\,\ grouped\ by\ type\ of\ object.kfpx`

You can also reference files, e.g. Profiles directly from the response file (you have to use the correct path to the correct location or course).
Like explained above, the correct syntax in a response file has to look like:

```
###################
# Check for specified resolution and change name of Check
#
Resolution of color and grayscale images is less than specified value.kfpx
--setvariable=RESOLUTION:300
--setvariable=CHECKNAME:Image resolution for images
#
###################
# EOF
```

Command line call:

```
pdfToolbox @<absolute path to "variables.rsp"> <PDF file>
```

You'll find a sample Profile (with the Variables used above) here:

Resolution_of_color_and_grayscale_images_is_less_than_specified_value.kfpx

Additional notes

- It is possible to use multiple response files within one CLI call.
- You can also add other command line parameters to the response file.
- The order of the content and the order of the response files will define the final order of all options and commands for the CLI call.
- Limitations with regard to the maximal length of a command line call can thus be avoided.

32.9 Creating a report using Profiles

You have a wide variety of options for creating a report. Only adding -r to your call would create an XML report next to the input PDF file, no matter if any hits occurred. You can modify this behavior by the following parameters:

```
--report=<type>,<trigger>,[options,]<PATH=path>
```

You can use --report as often as you like in one run to create different type of reports.

Parameters

type	see "Report types"
trigger	see "Report triggers"
options	see "Further options"
path	see "Report path"

Report types

XML	XML report
XSLT=<type>	XSLT report, type can be a custom type or one of the types delivered with pdfToolbox CLI ("compacttext" or "compacthtml")
MASK	PDF report, problems highlighted by transparent masks
COMMENT	PDF report, problems highlighted by annotations
LAYER	PDF report, problems separated on layers
INVENTORY	PDF report which lists all resources used in the PDF file
COMPARE	PDF compare report

SUMMARY	PDF overview report
TEMPLATE	PDF report based on HTML templates
VARDUMP	Dumps the app.vars object of JavaScript Variables in a JSON file (starting with pdfToolbox 10.0)
JSON	JSON report – based on JS object (starting with pdfToolbox 13.0)
JSONV2	JSON report – similar to XML report (starting with pdfToolbox 15.0)

Report triggers

ALWAYS	Always create report (default)
ERROR	Create if at least one problem with severity "Error" was found
WARNING	Create if at least one problem with severity "Warning" was found
INFO	Create if at least one problem with severity "Info" was found
HIT	Same as INFO,WARNING,ERROR

Further options

OVERVIEW	Include overview for PDF reports
OVERVIEW_TEMPLATE	Include overview based on HTML templates for PDF reports (starting with pdfToolbox 10.0). Click here for further information.
DOWNSAMPLING [#<PPI>]	Downsample images in PDF reports of type Mask, Layer and Comment to the given resolution (Default: 150 ppi)
HIDEERROR	Hide all checks with severity "Error" (not for type "Layer")

HIDEWARNING	Hide all checks with severity "Warning" (not for type "Layer")
HIDEINFO	Hide all checks with severity "Info" (not for type "Layer")
LANGUAGE	Language for the respective report, will over-write general --language option of the CLI (starting with pdfToolbox 10.0)

PDF layer report options

ICCNAMES	Create layer for all ICC color space names
SPOTNAMES	Create layer for all spot color space names
FONTNAMES	Create layer for all font names

PDF inventory report options

FONTS	Include fonts
COLORS	Include colors
SHADES	Include smooth shades
PATTERNS	Include patterns
IMAGES	Include images, optional number of pixels like IMAGES_100
FORMXOB	Include Form XObjects
XMP	Include XMP
XMPADV	Include XMP advanced

XML report options

ALL	Include all ressources (Default)

NONE	Include no ressources
IMAGES[#<NUM>]	Include first <NUM> images
FONTS[#<NUM>]	Include first <NUM> fonts
PAGES[#<NUM>]	Include first <NUM> pages
COLORS[#<NUM>]	Include first <NUM> color spaces
SHADES[#<NUM>]	Include first <NUM> smooth shades
PATTERNS[#<NUM>]	Include first <NUM> patterns
FORMXOB[#<NUM>]	Include first <NUM> Form XObjects
INKCOV	Include ink coverage for every page
INKCOVRES[#<PPI>]	Rendering resolution used for ink coverage calculation (Default: 10 ppi)
INKCOVBOX[#<Box>]	Defines the PageBox (as ArtBox, TrimBox, BleedBox, CropBox or MediaBox) which will be used for rendering (Default: CropBox)

JSON report options (JSONv1 – based on JS object)

ALLCHECKS	Include Checks without hits
COMPACT	The JSON document is written in a single line
QUICKCHECK=<cfg>	Custom QuickCheck configuration file, use NONE to disable
INKCOV	Include ink coverage for every page
INKCOVRES[#<PPI>]	Rendering resolution used for ink coverage calculation (Default: 10 ppi)
INKCOVBOX[#<Box>]	Defines the PageBox (as ArtBox, TrimBox, BleedBox, CropBox or MediaBox) which will be used for rendering (Default: CropBox)

Since pdfToolbox 14 a predefined Quick Check configurations file is included in the report creation. This file provides the

JSON report with additional information about the document (e. g. resources).

It is possible to reference your own Quick Check configuration file, which will disable the default one.

Here you can download the default Quick Check configuration and customize it according to your needs (all Quick Check objects can be found here).

defaultQuickCheck.cfg

With `--report=JSON,QUICKCHECK=NONE` you can deactivate the default Quick Check configuration file, so that no additional information is generated (this saves a little bit of time during report generation).

JSONV2 report options (JSONv2 – similar to XML report)

The JSON v2 report, lists contents similar to the XML report (starting with pdfToolbox 15.0).

ALLCHECKS	Include Checks without hits
COMPACT	The JSON document is written in a single line
QUICKCHECK=<cfg>	Custom QuickCheck configuration file, use NONE to disable
INKCOV	Include ink coverage for every page
INKCOVRES[#<PPI>]	Rendering resolution used for ink coverage calculation (Default: 10 ppi)
INKCOVBOX[#<Box>]	Defines the PageBox (as ArtBox, TrimBox, BleedBox, CropBox or MediaBox) which will be used for rendering (Default: CropBox)
ALLFIXUPS	Include Fixups with no changes
VARIABLES	Include Variables
Resource limit options	

ALL	Include all resources (default)
NONE	Include no resources
IMAGES [#<NUM>]	Include first <NUM> images (e.g. IMAGES [#100] -> JSON report will include the first 100 images.
FONTS [#<NUM>]	Include first <NUM> fonts
PAGES [#<NUM>]	Include first <NUM> pages
COLORS [#<NUM>]	Include first <NUM> colors
SHADES [#<NUM>]	Include first <NUM> shades
PATTERNS [#<NUM>]	Include first <NUM> patterns
FORMXOB[#<NUM>]	Include first <NUM> Form XObjects

Report path

PATH=<path>

path	Path to report file (if not defined, report is created next to input file) When defined, this must always be the last element of the --report parameter.

PDF report based on HTML template

```
TEMPLATE=<path>
```

path	Path to template folder (or respective index.html directly) PDF overview reports are created based on a style defined in a HTML/CSS template. Click here for further information. A predefined template can be found in the Server/CLI path: ../var/Reports/Templates.

Maximum number of pages

```
--maxpages
```

Maximum number of pages (default: 1000) - after this limit only pages that have problems will be stored.

Hits per page

```
--hitsperpage=<number>
```

Maximum number of hits per page reported. Every check will be considered individually.

Parameters

number	maximum number of hits per check per page that will be reported

Hits per document

```
--hitsperdoc=<number>
```

Maximum number of hits per document reported. Every check will be considered individually.

Parameters

number	maximum number of hits per check per document that will be reported

Setting the report language

```
-l --language=<language>
```

Sets the desired language for report files.
See article "Command line options for multi-language support" for a list of fully localized languages and how to extend incomplete translations or how to modify strings.

Parameters

language	language of report files
en	English
de	German

language	language of report files
fr	French
es	Spanish
it	Italian
pt	Brazilian Portoguese
cz	Czech
da	Danish
nl	Dutch
fi	Finnish
ja	Japanese
ko	Korean
no	Norwegian
pl	Polish
sv	Swedish

Example

```
pdfToolbox --language=fr
--report=ERROR,WARNING,LAYER,OVERVIEW,PATH=<path to report
file> <profile> <PDF file>
```

```
pdfToolbox --report=ERROR,WARNING,TEMPLATE=OVERVIEW,PATH=<path
to report file> <profile> <PDF file>
```

Custom Dict files

Custom Dict files can be used to add or modify strings that are used in reports. Since pdfToolbox 10.0 it is allowed for custom dicts to cover strings for custom profiles or unknown Dictkeys in a particular language.

Creation of custom dicts

In the following example, a custom dict is created for strings of a non localised language to be added/replaced in a predefined Profile (kfpx). This can be achieved using the command line argument:

```
--createcustomdict <Path to Profile file>
```

This creates an XML for the profile (kfpx).

```
<?xml version="1.0" encoding="UTF-8" ?>
<callas>
        <dict version="3.0" lang="en">
                <keys>
                        <group key="P">
                                <entry key="8_Listpotentialfontproblems">
                                        <value var="long">List potential font prob-
lems</value>
                                        <value var="short">Lists possible font is-
sues.</value>
                                </entry>
                        </group>
```

Click to copy

A user can add his own custom language and values (long for Name and short for Comment as in this example).

```
<?xml version="1.0" encoding="UTF-8" ?>
<callas>
        <dict version="3.0" lang="in">
                <keys>
            <group key="CUSTOMDICT">
                <entry key="भारत">
                    <value var="long">भारत</value>
                    <value var="short">भारतभारतभारत</value>
                </entry>
            </group>
```

Click to copy

On running the profile along with the custom dict set to the XML with overwritten strings and custom language, there is

no problem of appearance of DictKeys for unknown languages in pdfToolbox reports:

```
--customdict=<Path to custom XML file>
```

```
./pdfToolbox /Users/List potential font problems.kfpx /Users/pdfToolbox CLI 10/123.pdf --report --customdict=/Users/Downloads/List potential font problems.xml --language=in
```

Click to copy

Profile name and comment without the appearance of DictKeys in the XML report (other report types will work as well of course):

```
<profile_name>भारत</profile_name>
                <profile_comment>भारतभारतभारत</profile_comment>
```

Click to copy

32.10 Enumerate Profiles

Generate XML report of all Profiles and optional generates a Profile summary report:

```
--enumprofiles [--format=format] [--cachefolder=cachefolder] [--relativepath] [--profilesummary=profilesummary] <profile (folder)> <report file>
```

Click to copy

Options

format	XML, JSON, JSON_COMPACT (Default: XML)
cachefolder	sets the cache folder path
relativepath	writes the path in the XML as relative POSIX path
profilesummary	create an additional PDF summary report, any of: PDF (PDF summary - default) CSV (comma-separated file with all Fixups and Checks) CSVVAR (comma-separated file with all Variables)

Arguments

profile (folder)	pdfToolbox Profile or folder containing pdfToolbox Profiles
report file	Destination path for profile summary report

JSON format example

For the attached Profile, the CLI execution call:

```
./pdfToolbox --enumprofiles --format=JSON ./"Enlarge page at edges.kfpx" ./"Enlarge page at edges.kfpx.json"
```

Click to copy

Enlarge_page_at_edges.kfpx

creates a JSON file like below:


```
{
        "information" :
        {
                "computername" : "ws-osx-cso",
                "date_time" : "2021-06-01T16:33:22+02:00",
                "operating_system" : "macOS 10.15.7",
                "product_name" : "pdfToolbox",
                "product_version" : "12.3 (565)",
                "username" : "callas software"
        },
        "profiles" :
        [

                {
                        "comment" : "",
                        "creation_date" : "2021-06-01T14:12:22+02:00",
                        "modification_date" : "2021-06-01T14:12:22+02:00",
                        "name" : "Enlarge page at edges",
                        "path" : "/Users/callassoftware/Enlarge page at edges.kf-
px",
                        "size" : "6666",
                        "variables" :
                        [

                                {
                                        "key" : "Add_mm_bottom",
                                        "label" : "Add [mm] (bottom)",
                                        "type" : "Float",
                                        "value" : 0.0
                                },

                                {
                                        "key" : "Add_mm_left",
                                        "label" : "Add [mm] (left)",
                                        "type" : "Float",
                                        "value" : 0.0
                                },
```

```
                                {
                                        "key" : "Add_mm_right",
                                        "label" : "Add [mm] (right)",
                                        "type" : "Float",
                                        "value" : 0.0
                                },

                                {
                                        "key" : "Add_mm_top",
                                        "label" : "Add [mm] (top)",
                                        "type" : "Float",
                                        "value" : 0.0
                                }
                        ],
                        "vars" :
                        {
                                "Add_mm_bottom" : 0.0,
                                "Add_mm_left" : 0.0,
                                "Add_mm_right" : 0.0,
                                "Add_mm_top" : 0.0
                        }
                }
        ]
}
```

Click to copy

- The "**variables**" object provides all information about a variable incl. a label name. This can be used for example, to build a generic UI.
- The "**vars**" object provides only key and value pairs and can directly be used in APIs.

32.11 Results (Return codes, Error codes and Reason codes)

Reason codes

The reason codes will be printed in the command line output if a general error occurs.

1000	Unknown reason
1001	A parameter is wrong
1002	A requested file could not be found
1003	A requested folder could not be found
1004	A requested folder is a file
1005	A requested file is a folder
1006	30 days trial period expired
1007	Time limited keycode expired
1008	Not activated (no keycode)
1009	PDF does not contain ICC profiles
1010	File could not be opened
1011	File is encrypted and could not be opened for writing
1012	File could not be saved
1013	File is damaged and needs repair

Return codes

All return codes below 100 indicate a successful operation.

0	Successful operation

Running a profile

0	No hit, no Fixups executed
1	At least one hit with severity ‘info’, no Fixups executed
2	At least one hit with severity ‘warning’, no Fixups executed
3	At least one hit with severity ‘error’, no Fixups executed
5	No hit, Fixups have been executed
6	At least one hit with severity ‘info’, Fixups have been executed
7	At least one hit with severity ‘warning’, Fixups have been executed
8	At least one hit with severity "error", Fixups have been executed; Fixups failed

When executing the compare action

0	Compared PDFs are equal
1	Compared PDFs have differences

Errors

100	Not serialized (no valid serialization found or keycode expired)
101	Command line parameter error
102	Command line syntax error (illegal command)
103	Unknown error (internal error)

104	File could not be opened
105	File is encrypted and could not be opened for writing
106	File could not be saved
107	File is damaged and needs repair
Error codes above 128 are **NOT** Windows (only Linux or Mac)	
130	Processing is aborted due to user cancel
132	Illegal instruction
134	Abnormal termination (e.g. out-of-memory)
136	Floating point exception (e.g. div-by-zero)
137	Kill (e.g. explicitly sent by user or by the system)
138	BUS error (e.g. bad memory access)
139	Segmentation violation (e.g. invalid memory reference)
Cells from 134-139 on Mac/Linux indicate an error (crash) of the environmental system. Please report such cases together with the details and files to support@callassoftware.com	
141	Broken pipe (e.g. used for interprocess communication)
142	Execution is cancelled after timeout
143	Termination request (e.g. explictely sent by user or by the system)
145	Child process died (e.g. used for interprocess communication)
159	Bad system call (e.g. a system call that is not supported by the kernel)

Errors for distributed processing

110	Action is not distributable

111	No Dispatcher is found
112	No Satellite was found or is ready for execution

Listing of all possible codes on the command line

All Return codes and Reason codes can also be listed on the CLI, using the following command:

```
pdfToolbox --status
```

This command will also list the current licensing status of the software.

32.12 Commands related to Arrange

Booklet

--booklet [--voffset=0mm] [--hoffset=0mm] [--page-height=pageheight] [--pagewidth=pagewidth] [--cutmarks] [--border=0mm] [--bleed=0mm]

Purpose

Prepares a PDF document for double sided printing, such that the printout can be folded and saddle-stitched.

Parameters

voffset	optional, vertical offset from placement (pt, in, mm, cm)
hoffset	optional, horizontal offset from placement (pt, in, mm, cm)
pageheight	optional, page height of the new page (pt, in, mm, cm)
pagewidth	optional, page width of the new page (pt, in, mm, cm)
cutmarks	optional, place cutmarks around every imposed page
bleed	optional, width of bleed added at outward edges of slots (pt, in, mm, cm)
border	optional, width of border added at outward edges of slots (pt, in, mm, cm)

Example

```
pdfToolbox --booklet --cutmarks <PDF file>
```

N-Up

--nup [--cutmarks] [--voffset=0mm] [--hoffset=0mm] [--page-height=pageheight] [--pagewidth=pagewidth] [--bor-

der=0mm] [--bleed=0mm] [--distance=distance] --htimes=<>
--vtimes=<>

Purpose

Puts several pages onto a new page. You have to define how many pages should be placed next to each other horizontally and vertically as well as the distance between the placed pages.

Parameters

cutmarks	optional, place cutmarks around every imposed page
voffset	optional, vertical offset from center (pt, in, mm, cm)
hoffset	optional, horizontal offset from center (pt, in, mm, cm)
pageheight	optional, height of the page where the single pages are placed on (pt, in, mm, cm)
pagewidth	optional, width of the page where the single pages are placed on (pt, in, mm, cm)
distance	optional, distance between placed pages (pt, in, mm, cm)
htimes	number of pages to be placed next to each other horizontally
vtimes	number of pages to be placed next to each other vertically
bleed	optional, width of bleed added at outward edges of slots (pt, in, mm, cm)
border	optional, width of border added at outward edges of slots (pt, in, mm, cm)

Example

```
pdfToolbox --nup --htimes=3 --vtimes=2 --distance=10mm <PDF file>
```

Fill page

--fillpage [--cutmarks] [--border=0mm] [--bleed=0mm] --distance=distance --pageheight=pageheight --pagewidth=pagewidth

Purpose

Puts several pages onto a new sheet with a defined page size.
Distributes pages across/down as space permits.

Parameters

cutmarks	optional, place cutmarks around every imposed page
distance	distance between placed pages (pt, in, mm, cm)
pageheight	height of the page where the single pages are placed on (pt, in, mm, cm)
pagewidth	width of the page where the single pages are placed on (pt, in, mm, cm)
bleed	optional, width of bleed added at outward edges of slots (pt, in, mm, cm)
border	

Example

```
pdfToolbox --fillpage --distance=10mm --pageheight=420mm --pagewidth=594mm <PDF file>
```

Merge & Impose

```
--mergeimpose <runlist> <sheet config>
```

Purpose

Merges PDF files and imposes merged PDF based on rules defined in run list and sheet setup. For more information see "Use of Imposition cfgs" in the callas pdfToolbox manual.

This is the same as running the actions --mergepdf and --impose in sequence.

To speed up this process, consider using the --incremental parameter.

Parameters

runlist	imposition run list folder or file; file extension has to be ".runlist"
sheet config	sheet configuration files; file extension has to be ".sheetconfig"

Pre-installed imposition configurations can be found in

<Application folder>/var/Actions/Impose

Example

```
pdfToolbox --mergeimpose <runlist> <sheet config> <PDF file> <PDF file>
```

Impose

--impose [--preprocessingprofile=preprocessingprofile] [--setvariable=setvariable] [--sheettemplate=<path to PDF>] <runlist folder> <sheet config folder>

Purpose

Imposes the PDF based on rules defined in run list and sheet setup. For more information see "Use of Imposition cfgs" in the callas pdfToolbox manual.

To speed up this process, consider using the --incremental parameter.

Parameters

preprocessingprofile	optional, path to a kfpx profile that is executed as a preprocessing step; the used profile can not contain variables
setvariable	optional, set imposition runlist environment variable (for examples see "Use of dynamic profiles")

runlist	imposition run list configuration files; file extension has to be ".runlist"
sheet config	sheet configuration files; file extension has to be ".sheetconfig"
sheettemplate	Places pages from the chosen PDF underneath of the imposed pages as a background.

Pre-installed imposition configurations can be found in

<Application folder>/var/Actions/Impose

Example

```
pdfToolbox --impose --preprocessingprofile=<profile> <runlist> <sheet config> <PDF file>
```

Listing all available imposition configurations

```
--list [--language=en] [--runlists] [--sheetconfigs]
```

Lists all imposition configuration files which are stored in <Application folder>/var/Actions/Impose.

Parameters

language	optional, language used for listing, supported values are: en English de German fr French
runlists	optional, this will list all available runlist configuration files
sheetconfigs	optional, this will list all available sheet configuration files

The usage of both --runlists and --sheetconfigs equals the usage of --list without any additional parameters.

Example

```
pdfToolbox --list --runlists --language=fr
```

Listing all fonts available for usage in an imposition runlist

```
--listfonts
```

Lists all font names that can be used for the "Set TextFont" command in an imposition runlist.

Slice

--slice <profile>

Purpose

Extracts objects from the current document defined by a pre-flight check and saves two files as a result (one containing the chosen objects, one containing the remaining objects).

Parameters

profile	pdfToolbox profile containing a single check that defines the objects to be sliced from the current document (e.g. color images with a resolution below 150dpi), pre-configured profiles can be found in <Application folder>/var/Actions/Slice

Example

```
pdfToolbox --slice <profile> <PDF file>
```

Reader spreads

--readerspreads [--nocoverpage] [--pageheight=pageheight] [--pagewidth=pagewidth] [--voffset=0mm] [--hoffset=0mm]

Purpose

Imposes a PDF document by placing two contiguous pages next to each other.

Parameters

voffset	optional, vertical offset from center (pt, in, mm, cm)
hoffset	optional, horizontal offset from center (pt, in,

	mm, cm)
pageheight	optional, page height of the new page (pt, in, mm, cm)
pagewidth	optional, page width of the new page (pt, in, mm, cm)
nocoverpage	optional, diables front cover handling

Example

```
pdfToolbox --readerspreads <PDF file>
```

Split in half

--splithalf [--setclipping]

Purpose

Splits double pages into single pages. Recognizes if a document contains single pages as well as double pages are, and then only splits the double pages, leaving the single pages as they are.

Parameters

setclipping	optional, sets clipping path to page dimension

Example

pdfToolbox --splithalf <PDF file>

Step & Repeat

--steprepeat [--cutmarks] [--voffset=0mm] [--hoffset=0mm] [--pageheight=pageheight] [--pagewidth=pagewidth] [--border=0mm] [--bleed=0mm] [--distance=distance] --htimes=<> --vtimes=<>

Purpose

Imposes a PDF by placing a page multiple times onto a newly created page.

Parameters

cutmarks	optional, place cutmarks around every imposed page
voffset	optional, vertical offset from center (pt, in, mm, cm)
hoffset	optional, horizontal offset from center (pt, in, mm, cm)
pageheight	optional, page height of the new page (pt, in, mm, cm)
pagewidth	optional, page width of the new page (pt, in, mm, cm)
distance	distance between placed pages (pt, in, mm, cm)
htimes	number of pages to be placed next to each other horizontally
vtimes	number of pages to be placed next to each other vertically
bleed	optional, width of bleed added at outward edges of slots (pt, in, mm, cm)
border	optional, width of border added at outward edges of slots (pt, in, mm, cm)

Example

```
pdfToolbox --steprepeat --htimes=3 --vtimes=2 --distance=10mm <PDF file>
```

Split PDF

--splitpdf [--digits=4] [--splitscheme=splitscheme]

Purpose

Splits multipage documents into smaller packages.

Parameters

digits	optional, defines the number of digits for page

	number (Default = 4)
splitscheme	optional, custom split scheme (see "Split scheme")

Naming of output files

If output option defines a folder, packages are always named as <document_name>_<suffix with 4 digits that has the number of the first page in file>

If output option defines a file name, the first package is named according to this file name. Further packages are created at the same place as the first package using a name as described above for folders.

Simple tokens may also be used in order to define the output:

<docname>	Defines the name of the original document
<firstpage>	Defines the first page number
<lastpage>	Defines the first page number
<firstpage><lastpage>	Use 4 digits if not changed using --digits
<firstpagelabel>	Evaluates page label of first page of splitted file
<lastpagelabel>	Evaluates page label of last page of splitted file

For more possible tokens please see "Token Engine" in the "callas pdfToolbox manual".

Split Scheme

```
--splitscheme=<expression>
```

Expression may be a number with an asterisk "*" or a more complex string. If it is a number with an asterisk "*" it creates PDF files with the defined number of pages. E.g. if the number is 3* it would create 3 packages with 3 pages and one package with one page from a 10 page file.

For possible expressions, please see table "Expressions".

Parameters

expression	can be a single value or a combination of values like the examples below for "Simple expressions" "Multipage expressions", "Simple expressions list" and the "Joker"

General Syntax

Start Page	(number)
Digit	0|1|2|3|4|5|6|7|8|9
Unsigned	digit {digit}.2
Number	[+|-] unsigned

Joker

<expression>,$

Can be combined with other expressions (has to be the last item in a list) in order to save all pages that are not part of any other expression into a separate PDF.

Example

```
1-5,8,-3--1,$
```

would create 4 PDFs with page 1-5, page 8, the last 3 pages and the rest of the pages of an input PDF.

Expressions

Type	Syntax	Example	
Simple expression	number[-number]	1-5	Page 1 to 5: [1,2,3,4,5]
		5-1	Page 5 to 1: [5,4,3,2,1]
		8	Only page 8
		-1	Last page
		-1--3	Last 3 pages in reverse order: [n-2,

Type	Syntax	Example	
			n-1, n]
		-1-3	Last n - 2 pages in reverse order: [n, n-1, ... ,3]
		*2(2)	[2][4][6]...

Type	Syntax	Example	Column 2
Simple expression with Simple Range	number[-number]_ number[-number]	1-2_-2--1	First and last 2 pages: [1,2,n-1,n]
		1-2_-2--1,$	Split PDF into 2 documents: First and last 2 pages [1,2,n-1,n] and remaining inner pages [3, ... ,n-2]
		1_1_1_1	4 times page 1: [1,1,1,1]
Multipage expression	even_pages even	even	All even pages (same as *2(2))
	uneven_pages uneven odd	uneven	All uneven pages (same as *2(1))
	Package number* [(start_page)]	5*	Packages of 5 pages
		5*(2)	Packages of 5 pages, starting with page 2

Type	Syntax	Example	Column 2
	Intervall *number [(start_page)]	*5	Every 5th page
		*5(2)	Every 5th page, starting with page 2
		*5(-20)	Single page PDFs for every 5th page of the last 20 pages of a document (totally 4 PDFs)

Type	Syntax	Example	
Simple expression list	simple_expression {"," simple_expression} ["," joker]	1-5,8,-3--1	3 PDFs with page 1-5, page 8 and the last 3 pages of an input PDF
		5*(2)	Packages of 5 pages, starting with page 2
		*5(2)	Every 5th page, starting with page 2
		*5(-20)	Single page PDFs for every 5th page of the last 20 pages of a document (totally 4 PDFs)

Merge PDF

--mergepdf {<List of PDF files> | <Folder>}

Merges PDF files.

PDFs are merged in the order as defined in the call. PDFs in folders are merged in alphabetical order.

If a folder is defined as input path, only PDF files inside this folder will be merged. Any other file formats will be ignored.

If no name is defined via -o:
- the name of the first original PDF is used
- if a folder is defined as input, the name of the folder is used

Example

```
pdfToolbox --mergepdf <input folder with PDF files to merge>
```

or for separate PDF files (the order of the files will be regarded in this case):

```
pdfToolbox --mergepdf <PDF 1> <PDF 2> ...
```

Split and merge PDF

```
--splitmergepdf [--splitscheme=splitscheme]
```

Purpose

Splits multipage documents into smaller packages and merges them to one document.

Parameters

splitscheme	optional, custom split scheme (see "Split scheme")

Split Scheme

```
--splitscheme=<expression>
```

Expression may be a number with an asterisk "*" or a more complex string. If it is a number with an asterisk "*" it creates PDF files with the defined number of pages. E.g. if the number is 3* it would create 3 packages with 3 pages and one package with one page from a 10 page file.

For possible expressions, please see table "Expressions".

Parameters

expression	can be a single value or a combination of values like the examples below for "Simple expressions" "Multipage expressions", "Simple expressions list" and the "Joker"

General Syntax

Start Page	(number)
Digit	0|1|2|3|4|5|6|7|8|9
Unsigned	digit {digit}.2
Number	[+|-] unsigned

Joker

```
<expression>,$
```

Can be combined with other expressions (has to be the last item in a list) in order to save all pages that are not part of any other expression into a separate place in the PDF.

Example

```
1-5,8,-3--1,$
```

would create 1 PDFs with page 1-5, page 8, the last 3 pages and the rest of the pages of an input PDF.

Expressions

Type	Syntax	Example	
Simple expression	number[-number]	1-5	Page 1 to 5: [1,2,3,4,5]

Type	Syntax	Example	
		5-1	Page 5 to 1: [5,4,3,2,1]
		8	Only page 8
		-1	Last page
		-3--1	Last 3 pages: [n, n-1, n-2]
		-1--3	Last 3 pages in reverse order: [n-2, n-1, n]
		-1-3	Last n - 2 pages in reverse order: [n, n-1, ... ,3]
		*2(2)	[2][4][6]...
Simple expression with Simple Range	number[-number]_ number[-number]	1-2_-2--1	First and last 2 pages: [1,2,n-1,n]
		1-2_-2--1,$	First and last 2 pages [1,2,n-1,n] and remaining inner pages [3, ... ,n-2]
		1_1_1_1	4 times page 1: [1,1,1,1]

Type	Syntax	Example	
Multipage expression	even_pages even	even	All even pages (same as *2(2))
	uneven_pages uneven	uneven	All uneven pages (same

Type	Syntax	Example	
	odd		as *2(1))
	Package number* [(start_page)]	5*	Packages of 5 pages
		5*(2)	Packages of 5 pages, starting with page 2
	Intervall *number [(start_page)]	*5	Every 5th page
		*5(2)	Every 5th page, starting with page 2
		*5(-20)	Every 5th page of the last 20 pages of a document (totally 4 pages)
Simple expression list	simple_expression {"," simple_expression} ["," joker]	1-5,8,-3--1	1 PDFs with page 1-5, page 8 and the last 3 pages of the input PDF
		1-5,8,-1--3	1 PDFs with page 1-5, page 8 and the last 3 pages of the input PDF, but the last 3 pages in reverse order
		5*(2)	Packages of 5 pages, starting with page 2

Type	Syntax	Example	
		*5(2)	Every 5th page, starting with page 2
		*5(-20)	Every 5th page of the last 20 pages of a document (totally 4 pages)

Example

```
pdfToolbox --splitmergepdf --splitscheme=-3--1 <PDF file>
```

Duplicate page

pdfToolbox --duplicatepage [--times=1] [--pageorder] [--pagerange]

Duplicate pages of the PDF.

Parameters

times	optional, defines the number of copies created (default = 1)
pageorder	optional, respects the order of pages and add dupli cates as a complete set (only available if times=1)
pagerange	

Example

pdfToolbox --duplicatepage --times=1 --pageorder <PDF file>

Split PDF at mark

pdfToolbox --splitatmark [--split]

Splits multipage documents into smaller packages based on a Profile with a single, page-based Check

Parameters

split	BEFORE (Default), AFTER, BERFOREAFTER (splits the PDF before and after any page with a hit), REMOVE (splits the PDF at any page with a hit and removes the page

Example

```
pdfToolbox --splitatmark --split=REMOVE <Profile> <PDF file>
```

32.13 Commands related to Large format printing

Large format printing

Tiling

```
--tiling [--tileshorizontal=1] [--tilesvertical=1]
[--sizehorizontal=0.00] [--sizevertical=0.00]
[--overlaphorizontal=0.00] [--overlapvertical=0.00]
[--consthorizontal=ltr] [--constvertical=ttb] [--addconstinfo] [--template=<path
to template>]
```

Purpose

Creates tiles from the input PDF either by number of tiles or by dimension of created tiles.

Parameters

tileshorizontal	Defines the number of tiles horizontally (can not be combined with --sizehorizontal or --sizevertical)
tilesvertical	Defines the number of tiles vertically (can not be combined with --sizehorizontal or --sizevertical)
sizehorizontal	Defines the horizontal size of the tiles (can not be combined with --tileshorizontal or --tilesvertical) Possible units are: mm|inch|pt
sizevertical	Defines the vertical size of the tiles (can not be combined with --tileshorizontal or --tilesvertical) Possible units are: mm|inch|pt
overlaphorizontal	Horizontal overlap of the tiles Possible units are: mm|inch|pt
overlapvertical	Vertical overlap of the tiles Possible units are: mm|inch|pt

consthorizontal	Type of horizontal construction direction. Possible: ltr (= Left to right)|rtl (=Right to left)]
constvertical	Type of vertical construction direction. Possible: ttb (=Top to bottom)|btt (=Bottom to top)
addconstinfo	Add construction information as a separate page.
template	Use custom template for creation (available with pdfToolbox 10.0)

Example

```
pdfToolbox --tiling --tileshorizontal=5 --tilesvertical=5
--overlaphorizontal=20mm --overlapvertical=20mm --addconstinfo
--consthorizontal=rtl --constvertical=btt <PDF file>
```

32.14 Commands related to Present

Actions

Present

Presentation

```
--presentation [--progressthermometer] [--blackslideatend] [--fullscreen] [--self-
running=0] [--transition=transition]
```

Purpose

Prepares a PDF for use as a slide presentation.

Parameters

progressthermometer	optional, add a progress thermometer at the bottom of the pages of the document
blackslideatend	optional, add black slide at the end of the document
fullscreen	optional, display presentation in full screen mode
selfrunning	optional, number of seconds for each page in a selfrunning presentation
transition	optional, transition between pages (any of: blinds, box, comb, cover, dissolve, fade, glitter, push, random, replace, split, uncover, wipe, zoomin, zoomout)

Example

```
pdfToolbox --presentation --selfrunning=5 --transition=split
--progressthermometer --fullscreen <PDF file>
```

Handout

```
--handout [--pagesize=DINA4] [--firstpageonlyoneslide]
[--slidesperpage=3]
```

Purpose

Creates a handout containing several slides on one page and optionally some lines for notes.

Parameters

pagesize	optional, pagesize of resulting document (any of: Letter, DINA4)
firstpageonlyoneslide	optional, place only one slide on the first page
slidesperpage	optional, number and layout of slides on page (any of: 2, 2nonotes, 3, 3nonotes, 2x2, 2x2nonotes, 2x3nonotes, 3x2)

Example

```
pdfToolbox --handout --pagesize=Letter --firstpageonlyoneslide
--slidesperpage=2x2 <PDF file>
```

Passe partout

```
--passepartout [--backgroundborderwidth=5mm] --background=<Path>
```

Purpose

Adds a background border around the current page content.

Parameters

backgroundborderwidth	optional, width of background border around each page (pt, in, mm, cm)
background	path to a PDF file used as the background pattern, pre-installed background pattern files can be found in <Application folder>/var/Actions/

	PassePartout

Example

pdfToolbox --passepartout --backgroundborderwidth=1cm

```
--background=<Background PDF> <PDF file>
```

Light table

```
--lighttable --background=<Path> [--numberofcolumns=5]
--pageheight=<value> --pagewidth=<value>
```

Purpose

Puts several pages on one new page to give the impression of a light table.

Parameters

background	path to a pdf file with the background pattern, pre-installed background pattern files can be found in <Application folder>/var/Actions/LightTable
numberofcolumns	optional, number of columns
pageheight	page height of the new page (pt, in, mm, cm)
pagewidth	page width of the new page (pt, in, mm, cm)

Example

```
pdfToolbox --lighttable --numberofcolumns=3 --background=<Background PDF>
--pageheight=420mm --pagewidth=594mm <PDF file>
```

32.15 Commands related to Document

Actions

Overlay

```
pdfToolbox --overlay [--voffset=0] [--hoffset=0] [--placement=TopRight]
[--placebelow[=1|2]]
```

Click to copy

Purpose

Places the selected overlay content on top of (or underneath) the processed PDF.

Parameters

hoffset	Optional, horizontal offset from placement (pt, in, mm, cm)
voffset	Optional, vertical offset from placement (pt, in, mm, cm)
placement	Optional, placement of the pages (any of TopLeft, TopCenter, TopRight, LeftCenter, Center, RightCenter, BottomLeft, BottomCenter, BottomRight)
placebelow	Optional, places the selected overlay content underneath the input PDF. Number of pages in resulting PDF is determined from the number of pages of:

	1: first argument (= overlay file) 2: input file (Default behavior of --placebelow) If no output name is defined, the name of output file will be derived from the input file name.
overlay file	Full path to PDF to put on top of the input PDF, preinstalled overlay files can be found in <Application folder>/var/Actions/Overlay

Example

```
pdfToolbox --overlay --voffset=10mm --hoffset=50mm <overlay file> <input PDF file>
```

Create EPS

```
--createeps [--rect=rect] [--pagebox=pagebox] [--transparencyquality=100]
[--gradientresolution=360] [--bitmapresolution=1200]
[--applyoutputpreviewsettings]
[--simulationprofile='ISO Coated v2 (ECI)'] [--colormanagement]
[--marksweight=0.125] [--pageinformation] [--colorbars]
[--registrationmarks] [--cutmarks] [--simulateoverprint]
[--postscript=3] [--ascii] [--workingspacecmyk=<ICC-profile>]
[--workingspacergb=<ICC-profile>]
[--workingspacegray=<ICC-profile>]
```

Purpose

Converts all pages of the PDF into EPS. The EPS files are saved next to the input PDF file unless you use -f to define an output path.

Parameters

rect	optional, define and use only part of page relative to --pagebox <lower left x>,<lower left y>,<upper right x>,<upper right y>[<unit=pt (default), mm>]
pagebox	optional, crop at page geometry box (options: MEDIABOX, CROPBOX (default), TRIMBOX, BLEEDBOX, ARTBOX)
transparencyquality	optional, transparency quality in % (default: 100)
gradientresolution	optional, gradient resolution in ppi (default: 360)
bitmapresolution	optional, bitmap resolution in ppi (default: 1200)
applyoutputpreviewsettings	optional, apply output preview settings
colormanagement	optional, apply host based color management (required for the e.g. simulationprofile)
simulationprofile	optional, simulation profile (default: 'ISO Coated v2 (ECI)')
marksweight	optional, line weight of cut marks in pt (default: 0.125)
pageinformation	optional, add page information
colorbars	optional, add color bars
registrationmarks	optional, add registration marks
cutmarks	optional, add cutmarks
simulateoverprint	optional, simulate overprint
postscript	optional, Postscript level [2|3] (default: 3)
ascii	optional, Postscript is written 'Clean 7 Bit'
workingspacecmyk	optional, working space profile CMYK (default: ISO Coated v2 (ECI))
workingspacergb	optional, working space profile RGB (default: sRGB IEC61966-2.1)

workingspacegray	optional, working space profile Gray (default: Dot Gain 15%)

Example

```
pdfToolbox --createeps --postscript=2 --pageinformation
--colorbars --registrationmarks --cutmarks <PDF file>
```

Create PostScript

```
--createps [--transparencyquality=100]
[--gradientresolution=360] [--bitmapresolution=1200]
[--applyoutputpreviewsettings]
[--simulationprofile='ISO Coated v2 (ECI)'] [--colormanagement]
[--marksweight=0.125] [--pageinformation] [--colorbars]
[--registrationmarks] [--cutmarks] [--simulateoverprint]
[--postscript=3] [--ascii] [--workingspacecmyk=<ICC-profile>]
[--workingspacergb=<ICC-profile>]
[--workingspacegray=<ICC-profile>]
```

Purpose

Converts all pages of the PDF into PostScript. The PostScript files are saved next to the input PDF file unless you use -f to define an output path.

Parameters

transparencyquality	optional, transparency quality in % (default: 100)
gradientresolution	optional, gradient resolution in ppi (default: 360)
bitmapresolution	optional, bitmap resolution in ppi (default: 1200)
applyoutputpreviewsettings	optional, apply output preview settings
simulationprofile	optional, simulation profile (default: 'ISO Coated v2 (ECI)') Not available on Unix
colormanagement	optional, apply host based color

	management
marksweight	optional, line weight of cut marks in pt (default: 0.125)
pageinformation	optional, add page information
colorbars	optional, add color bars
registrationmarks	optional, add registration marks
cutmarks	optional, add cutmarks
simulateoverprint	optional, simulate overprint
postscript	optional, Postscript level [2|3] (default: 3)
ascii	optional, Postscript is written 'Clean 7 Bit
workingspacecmyk	optional, working space profile CMYK (default: ISO Coated v2 (ECI))
workingspacergb	optional, working space profile RGB (default: sRGB IEC61966-2.1)
workingspacegray	optional, working space profile Gray (default: Dot Gain 15%)'

Example

```
pdfToolbox --createps --postscript=2 --pageinformation
--colorbars --registrationmarks --cutmarks <PDF file>
```

Save as image

Watch this short video to learn more about the "Save as image" action on the CLI.

```
--saveasimg [--parallel=parallel][--nosimulateoverprint] [--simulationprofile=<ICC
profile>]
[--smoothing=lines] [--resolution=72] [--colorspace=colorspace]
[--jpegformat=Baseline_Standard] [--compression=JPEG_medium]
```

```
[--imgformat=JPEG] [--pagebox=cropbox] [--rect=<left>,<bottom>,<right>,<top>[unit]
[--digits=4]]
```

Purpose

Renders an image per page preserving the page's aspect ratio. RGB images always use sRGB as Destination ICC profile which gets embedded into the resulting image. CMYK and gray TIFF images are saved without an ICC profile, while JPEG images will contain the ICC profile.

For rendering purposes, the order in which profiles used as a working space (and in which is rendered) are determined:

- if a simulationprofile is defined, it will be used
- if a simulationprofile is not defined, the Output Intent is used
- if no simulationprofile or Output Intent exists, the following profiles will be used:
  - RGB: sRGB IEC61966-2.1;
  - CMYK: ISO Coated v2 (ECI);
  - Gray: Dot Gain 15%.

The defined simulation profile will only replace the default profile for the respective colorspace.

If the destination colorspace is same as used for rendering, this ICC profile will be used. Otherwise one of the following is used:

- RGB: sRGB IEC61966-2.1;
- CMYK: ISO Coated v2 (ECI);
- Gray: Dot Gain 15%.

As Rendering Intent "AC_RelColorimetric" is used by default.

Starting pdfToolbox 12.3, Gray with Alpha channel(GrayA) is now supported for TIFF and PNG. "RGBA" is now also added for TIFF (only PNG up until now)

Parameters:

parallel	optional; Parallel processing for page rendering. Usage e.g.

	--parallel=4 (Rendering with 4 cores)
nosimulateoverprint	optional; avoids the overprint-simulation
simulationprofile	optional; using a user-defined ICC-profile for rendering (RGB, CMYK or Gray)
smoothing	optional; None, All, Lines, Images, Text, NTLH (default: All; NTLH includes "All")
resolution	optional; resolution in ppi or width x height in pixel, e.g. 1024x800 (default: 72)
colorspace	optional; one of RGB, RGBA, CMYK, Gray, Lab, Multichannel (default: RGB) availabilty depends on imageformat: **JPEG**: RGB, Gray, CMYK (default: RGB) **PDF**: RGB, Gray, CMYK, Lab (default: RGB) **TIFF**: RGB, RGBA, Gray, GrayA, CMYK, Lab, Multichannel (default: RGB) **PNG**: RGB, RGBA, Gray, GrayA (default: RGB)
jpegformat	optional; Baseline_Standard, Progressive_3_Scan (default: Baseline_Standard)
compression	optional; for JPEG: JPEG_minimum, JPEG_low, JPEG_medium, JPEG_high, JPEG_maximum (default: JPEG_medium) for TIFF: TIFF_None, TIFF_LZW, TIFF_Flate (default: TIFF_LZW)
imgformat	optional; JPEG, PNG, TIFF, PDF (default: JPEG)
pagebox	optional; using a geometry box as size for image: CROPBOX, TRIMBOX, BLEEDBOX, MEDIABOX (default: CROPBOX)

rect	optional; render only the part defined by lower left and upper right from origin geometry box (default: CROPBOX); in pt or mm (default:pt)
simulatepaper	optional; simulates paper color (by using absolute colorimetric color conversion) not available if --nosimulateoverprint is set; needs a defined --simulationprofile; only available for colorspace = RGB
blackpointcompensation	optional; using blackpoint compensation (not available if --nosimulateoverprint is set)
digits	Defines the number of digits for page number in file name of created image

Example

```
pdfToolbox --saveasimg --imgformat=PNG --resolution=800x600 <PDF file>
```

Extract text

```
--extracttext
```

Purpose

Extracts the text of PDF documents to the command line or to a specified file.

Example

```
pdfToolbox --extracttext <PDF file>
```

Extract content

```
--extractcontent [--words] [--wordbbox] [--wordquads]
[--chars] [--docxmp] [--docinfo] [--annots]
```

Purpose

Extracts the text in the form of words or characters to an XML file.

Parameters

words	Include words
wordbbox	
wordquads	Include quad point information for word parts
chars	Include quad point information for individual characters
docxmp	Include document XMP metadata
docinfo	Include document info
annots	Include link annotations

Example

```
pdfToolbox --extractcontent [--words] [--docinfo] <PDF file>
```

Extract images

```
--extractimages [--threshold=0] [--report=<path>]
```

Purpose

Extracts images from the file and creates a special XML report, which lists all extracted images with their relevant details.

Parameters

threshold	Extracts only images with width and height larger than threshold (default: 0)
report	Creates a report with details about the extracted images and their former position in the PDF.

Example

```
pdfToolbox --extractimages --report --threshold=250 <PDF file>
```

This action can not be used with distributed processing.

Redistill

```
--redistill [--topdf_pdfsetting=<joboptions>] <PDF file>
```

Purpose

Recreates the PDF via PostScript, prepares for use with older equipment (RIPs).

Parameters

topdf_pdfsetting	Path to PDF settings file to be used for conversion of PS and EPS files only, must be a Distiller .joboptions file

Example

```
pdfToolbox --redistill <PDF file>
```

Optimize PDF

```
--optimizepdf <PDF file>
```

Purpose

Optimizes the internal structure of the PDF and saves for Fast Web View.

Example

```
pdfToolbox --optimizepdf <PDF file>
```

This action is deprecated with pdfToolbox 12 as a single action and has become a option for normal processing.

To perform a complete file optimization, just use "--topdf" with the additional option "--optimizepdf".

To PDF

```
--topdf [--topdf_pdfsetting=<path>] [--topdf_psprologue=<path>][--topdf_psepilogue=<path>] [--ignorefontembeddingproblems=<on|off>]
```

Purpose

Converts supported non-PDF files to PDF. Information about supported file types can be found here:

http://www.callassoftware.com/goto/tbx_ENU_topdf

Parameters

topdf_pdfsetting	Full path to PDF *settings* file to be used for conversion of PS and EPS files only, must be a Distiller .joboptions file
topdf_psprologue	Full path to a *prologue* file which will be prepended to the PostScript/EPS file to be converted. To be used for conversion of PS and EPS files only. Must be a valid PostScript file.
topdf_psepilogue	Full path to a *epilogue* file which will be appended to the PostScript/EPS file to be converted. To be used for conversion of PS and EPS files only. Must be a valid PostScript file.

Note:

- ICC profiles referenced in a PDF settings file (.joboptions) need to be copied into the operating system folder for ICC profiles, e.g.:
  - Windows: C:\Windows\system32\spool\drivers\color
  - MacOS: /MacOS HD/Library/ColorSync/Profiles

- The application will also look into the following folders for ICC profiles:
  - /Library/Application Support/Adobe/Color/Profiles/Recommended
  - /Library/Application Support/Adobe/Color/Profiles
  - /System/Library/ColorSync/Profiles
- Alternatively, you can put ICC-files for PostScript to PDF in the subfolder of the application:
  .../etc/PDFPSTool/ICCProfiles
- A Color settings file (.csf) that is referenced in the PDF settings file is not necessary for the processing.

Examples

```
pdfToolbox --topdf <non-PDF file>
```

```
pdfToolbox --topdf /path/to/file/mypostscript.ps
           --topdf_pdfsetting=/path/to/file/mysettings.joboptions
           --topdf_psprologue=/path/to/file/myprologue.ps
           --topdf_psepilogue=/path/to/file/myepilogue.ps
```

Uncertify

```
--uncertify <PDF file>
```

Purpose

Removes a Preflight certificate if present.

Example

```
pdfToolbox --uncertify <PDF file>
```

Secure PDF

```
--securepdf --password=<password>
```

Restrict editing and printing of the PDF. A password is needed in order to change these permission settings or to perform changes. The PDF can only be read afterwards

The entered password will be visible and may be grabbed or logged by other processes on the machine.

Parameters

password	password to avoid editing or printing

Creating file packages

Some PDF standards allows the embedding of PDF- and also non-PDF-files into another PDF file. Sometime these file packages are also called collections.

Using pdfToolbox CLI it is possible to create such file packages from a complete folder or to define different ways how a file which shall be embedded is handled.

In general a file package is created with --collection This will create an index document, which lists all embedded files from the given folder. Also an existing folder structure will be respected

```
--collection <folder>
```

In general a file package is created with --collection This will create an
index document, which lists all embedded files.

```
--collection <file> [<file>]
```

Settings for file embedding

```
--collection [--embedinto=[target],<file>] [--embedfile= [target,[relation-
ship],<file>] [--embedwithlink= [area,<file>]
```

--embedinto

It is possible to use own templates or normal PDF for embedding files. The standard for the file where other files will be embedded can be defined using the conversion target (see below). If no file is defined, an index file is created.

--embedfile

Also for files to embed a conversion target can be defined using the conversion target. For PDF/A-3 standards also a relationship entry for each embedded file can be set.

Parameters

target	A3b, A3u, A3a, A2b, A2u, A2a, A1b, A1a or No (Default)

Using the target "No", no conversion to PDF is done. (Only available for embedded files.)

relationship	Source, Data, Alternative, Supplement, Unspecified (Default)

--embedwithlink

Alternatively, files can be embedded with defining an area in the containing document, where a link to the contained file is created. No conversion will take place with the file to embed.

Parameters

Defines a rectangular area, based on the lower left corner of the page, where a link to the embedded file is inserted. Default unit is pt.

area	X1,X2,Y1,Y2[pt, in, cm, mm]

Example:

```
--collection --embedinto=A3b,<PDF file> --embedfile=A3b,Alternative,<file> --embed-
file=A2b,Source,<Office file>
--embedfile=No,Data,<file>
```

```
--collection --embedwithlink=10,10,100,100,<file> --embedwith-
link=10mm,100mm,100mm,200mm,<file>
```

Extracting files from file packages

```
--extractembeddedfiles [--plain] [--filter=filter] <PDF file>
```

Purpose

Extracts embedded files from a PDF.

Parameters

plain	Files are extracted directly into the destination folder without restoring an existing folder structure of the embedded files.
filter	RegEx based file name filter, e.g. =.*.doc

Example:

```
--extractembeddedfiles --plain <PDF file>
```

Extract dieline

```
--extractdieline [--format] [--includeclippingpaths] [--pagebox=TRIMBOX] <Profile>
<PDF file>
```

Purpose

Extract dieline(s) from PDF

Parameters

format	svg, dxf, cf2 svg:fill, svg:stroke, svg:fillstroke Default: svg (which is the same as svg:fill) The fill, stroke, fillstroke variations specify whether from stroked PDF paths filled, stroked or filled and stroked SVG objects are created. What format should be used depends on the machinery used.
includeclippingpaths	Export clipping paths as well (in addition to stroking paths)
pagebox	Defines the box used for the dimension of the created dieline format Available: MEDIABOX, CROPBOX (default), TRIMBOX, BLEEDBOX, ARTBOX

Example:

```
--extractdieline --format=svg:fillstroke ./dieline.kfpx <PDF file>
```

32.16 Command to add bookmark structure

Add bookmarks

```
--addbookmark <XML file> <PDF file>
```

Purpose

The structure for bookmarks has to be defined in a XML file. It is also possible to use various formatting styles for the bookmarks ("PLAIN", "BOLD", "ITALIC", "BOLD_ITALIC" and colors "R,G,B"). This formatting will only be shown if the respective PDF viewer supports such styling information.

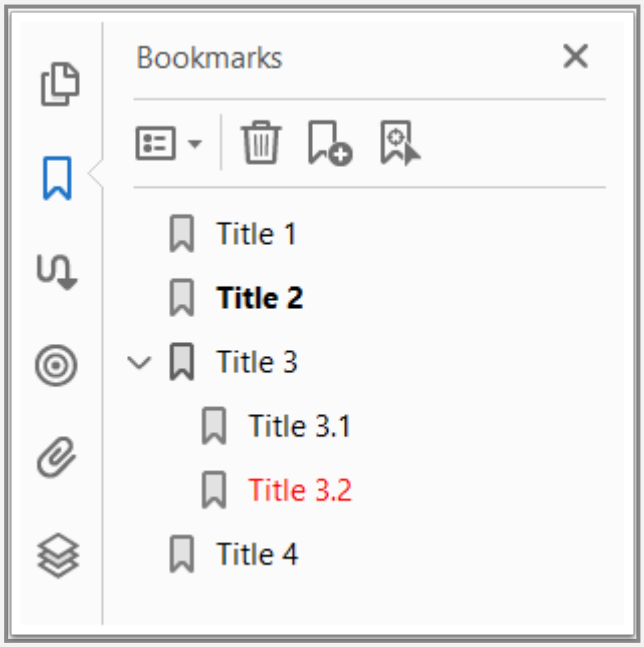
Also a subcategorization for the bookmark tree is possible.

Only page numbers as destinations can be defined, but e.g. no coordinates on a specific page.

Sample XML file

```
<?xml version="1.0" encoding="utf-8"?>
<BOOKMARKS xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noName-
spaceSchemaLocation="bookmarks.xsd">
  <BOOKMARK ORDER="100" TITLE="Title 1" PAGE_NUMBER="2" />
  <BOOKMARK ORDER="200" TITLE="Title 2" PAGE_NUMBER="3"  STYLE="BOLD" />
  <BOOKMARK ORDER="300" TITLE="Title 3" PAGE_NUMBER="1" />
  <BOOKMARK ORDER="301" TITLE="Title 3.1" PAGE_NUMBER="4" />
  <BOOKMARK ORDER="302" TITLE="Title 3.2" PAGE_NUMBER="5" COLOR="255,0,0"/>
  <BOOKMARK ORDER="400" TITLE="Title 4" PAGE_NUMBER="7" />
</BOOKMARKS>
```

Click to copy
Resulting bookmark structure:


Bookmarks
Title 1
Title 2
Title 3
Title 3.1
Title 3.2
Title 4

32.17 Commands related to Colors

Actions

Process conversion

This action is only included for legacy reasons.
For new implementations, please use the "Convert colors"
fixup, which you can configure via the desktop UI.

```
pdfToolbox --convertcolors [--spotcolor=spotcolor] --destination=destination] [--
source=source]
```

Purpose

Prepares the current PDF for the chosen printing condition and carries out the necessary color conversion. For further information on setting up configuration files (*.cfg) see "Policies" in the callas pdfToolbox manual. You can either define a separate config file for each of source, spot color and destination or only use one of the switches with a single config file containing all necessary conversion parameters.

Parameters

spotcolor	full path to a cfg file defining the spot color options, pre-installed config files can be found in <Application folder>/var/Actions/ConvertColors/SpotColors
destination	full path to a cfg file defining the destination options, pre-installed config files can be found in <Application folder>/var/Actions/ConvertColors/Destination
source	

Example

```
pdfToolbox --convertcolors --spotcolor=<spot config file>
--destination=<destination config file>
--source=<source config file> <PDF file>
```

Extract ICC profiles

```
pdfToolbox --extracticcprofiles
```

Purpose

Saves all embedded ICC profiles from the document. Profiles of ICC based color spaces as well as ICC profiles used in Output Intents are extracted.

Example

```
pdfToolbox --extracticcprofiles <PDF file>
```

Extract or embed CxF data

```
pdfToolbox --cxf [--extract | --embed=<path to folder with CxF file>]
```

Purpose

Extracts or Embeds CxF data from or into the current PDF file.
See this chapter for more details regarding CxF.

Parameters

--extract	Extracts all CxF data from a PDF into XML files
--embed>	Embeds CxF files into the Output Intent of a PDF and adds corresponding XMP metadata (e.g. --embed=<Path to folder with CxF file>

32.18 Commands related to Layers

Actions

Enumerate layers

```
--enumeratelayers [--iccnames] [--fontnames] [--spotnames]
[--language=<language>]
```

Purpose

Enumerate the chosen objects on separate layers.

Parameters

iccnames	optional, creates a layer for each ICC profile in the PDF
fontnames	optional, creates a layer for each font in the PDF
spotnames	optional, creates a layer for each spot color in the PDF
language	language for naming of layers Supported values are: en English de German fr French es Spanish it Italian pt Brazilian Portoguese cz Czech da Danish nl Dutch fi Finnish ja Japanese ko Korean no Norwegian pl Polish sv Swedish

Example

```
pdfToolbox --enumeratelayers --fontnames <PDF file>
```

Import as layer

```
--importaslayer [--name="Layer 1"] <import file>
```

Purpose

Imports the chosen PDF document as a layer in the processed PDF.

- If e.g. the imported PDF contains 2 page and the document it is imported to contains 5, only page 1 and 2 of the resulting document would contain a layer.
- If e.g. the imported PDF contains 2 pages and the document it is imported to 1 page, then only the first page would be imported.

Parameters

name	optional, name of the new layer
import file	full path to a PDF to be imported

Example

```
pdfToolbox --name="My Layer" <PDF file> <PDF file>
```

Split layers

```
--splitlayers [--singlepages]
```

Creates a separate PDF file for every layer view or layer with the content visible when viewing this layer view or single layer.

The name of the output files (if not defined via -o) will have the following syntax:
originalfilename_layer(view)

Parameters

singlepages	optional, creates a separate PDF per page of the original PDF File names are suffixed using the page number

Example

```
pdfToolbox --splitlayers --singlepages <PDF file>
```

32.19 Commands related to Reports

Reports

Extract XMP metadata

```
--extractxmpmetadata <report config file>
```

Purpose

Extract XMP Metadata of the processed PDF into a configurable XML file. For more information see "Use of XMP Metadata reports".

Parameters

report config file	full path to a meta data report configuration file, pre-installed config files can be found in <Application folder>/var/Actions/Metadata/Filters/Export

Example

```
pdfToolbox <report config file> <PDF file>
```

List basic PDF info

```
--quickpdfinfo
```

Purpose

Lists some basic PDF information to output.

```
Example pdfToolbox --quickpdfinfo <PDF file>
```

Visualizer

```
--visualizer [--usebleed] [--safetyoutside=safetyoutside] [--safetyinside=safetyinside] [--gamut_method=gamut_method] [--gamut_targetprofile=gamut_targetprofile] [--gamut_currentprofile=gamut_currentprofile] [--gamut=gamut] [--sep_colors] [--on-
```

```
lyproblems] [--smlobj=smlobj] [--inkcov=inkcov] [--bmpres=bmpres] [--imgres=im-
gres] [--part=part] [--resolution=resolution] [--colorspace=colorspace] [--jpegfor-
mat=jpegformat] [--compression=compression] [--imgformat=imgformat] ... <input
file>
```

Purpose

Create a report listing print relevant aspects of a PDF document.

Parameters

usebleed	Width of BleedBox is used if existing, otherwise the default value or the value defined by “--safetyoutside” (default:off)
safetyoutside	Used as safety distance outside of the TrimBox, default: 3mm
safetyinside	Used as safety distance inside of the TrimBox, default: 3mm
gamut_method	Method used for Delta E (Default: Delta E 2000)
gamut_targetprofile	View in target color space
gamut_currentprofile	View in current color space
gamut	View in current and target color space and as heat map
smlobj	Threshold for small object coverage highlighting, see "Resolution output" (default: low)
inkcov	Threshold for ink coverage highlighting (default: 250)
bmpres	Threshold for bitmap resolution highlighting (default: 550)
imgres	Threshold for image resolution highlighting (default: 150)
part	See "Report parts"
format	See "Report type"
resolution	Resolution in ppi or width x height in pixel, e.g.

	1024x800 (default: 72)
language	report language (e.g. en (English, default), de, es, fr or it)
sep_colors	Renders all individual separations in their respective color

Resolution output

Following you will find the values taken as tresholds for the chosen output resolution.

	Text	Multicolored text	Line	Multicolored line
low	8 pt	10 pt	0.5 pt	2 pt
medium	5 pt	9 pt	0.125 pt	0.25 pt
high	5 pt	8 pt	0.125 pt	0.25 pt

Report parts

Image report:

all	all visualizer parts
full	regular page view
ink	all ink coverage views
ink_temp	ink coverage above threshold
ink_topo	ink coverage topographic view
process	all process color views
process_CMYK	CMYK channels (without spots)
process_CMY	CMY
process_K	K channel only
spot	all spot color views
spot_spots	spot color channels
spot_spots_K	spot color + K channels

spot_CMYK	CMYK channels (without spots)
sep	all individual separations
sep_process	all individual process separations
sep_spot	all individual spot color separations
sep:<NAME>	separation of colorant with name <NAME>
imgres	all image resolution views
imgres_img	image resolution below threshold
imgres_bmp	bitmap resolution below threshold
smallobj	all small object views
smallobj_text	very small text objects below threshold
smallobj_lines	very small vector objects below threshold
safety	all safety zone views
safety_bleed	bleed area safety zone
safety_trim	page border safety zone
safety_full	safety zone regular page view

PDF report (supported until pdfToolbox 11):

all	all visualizer parts
ink	all ink coverage views
sep	all individual separations
imgres	all image resolution views
smallobj	all small object views
safety	all safety zone views

Example

```
pdfToolbox --visualizer --smlobj=medium --inkcov=300
--part=ink --format=pdfreport -p=1-5 <PDF file>
```

Please note that 'pdfreport' as format is only functional until and including pdfToolbox 11.

Compare

```
--compare [--channels=rgb] [--onlydifferences][--highlighting=mask] [--
diffres=150] [--threshold=20] [--areathreshold=5] [--format=images] [--anchor=an-
chor] [--anchorbox=anchorbox] [--offset=offset] [--pagebox=pagebox] [--nosimula-
teoverprint] [--colorspace=colorspace] [--jpegformat=Baseline_Standard] [--compres-
sion=JPEG_medium] [--imgformat=JPEG] [--anchor=upperleft] [--anchorbox=CROPBOX] [--
offset=5mm,5mm] --pagebox=CROPBOX] <PDF file 1> <PDF file 2>
```

Purpose

Compares two PDF documents and creates a report.

Parameters

channels	optional, channels to be compared, any of: lab, lab76, ab, ab76, RGB (default, including spot colors), CMYK, CMY (both including spot colors), or as separation (without spot colors): PROCESS_CMYK, PROCESS_CMY, PROCESS_C, PROCESS_M, PROCESS_Y, PROCESS_K
onlydifferences	Only emit pages that have differences
highlighting	optional, highlighting format for differences, redmask (default), mask
diffres	Resolution used for comparison in ppi (Default = 72 ppi)
threshold	optional, defines the difference highlighting in % of the compared pixel, to be used instead of --sensitivity Default = 0% (highest sensibility)
sensitivity	deprecated parameter - use threshold controls difference highlighting, any of: maximum: Maximum sensitivity (default)

	medium: Medium sensitivity minimum: Minimum sensitivity
areathreshold	optional, defines the threshold for the area with pixel differences above value defined with parameter "threshold"
format	Compare report format, any of: pdfreport (default): creates a PDF compare report (based on default template) imgreport: creates an images compare report (JPEG as default) images: creates 3 individual images for PDF file 1, PDF file 2 and difference (JPEG as default) template:<template>: creates a PDF compare report based on a template (example can be found below)
nosimulateoverprint	optional, deactivates simulate overprinting
colorspace	optional, one of: RGB, CMYK, Gray (default: RGB) (only applicable if report format is images)
jpegformat	optional, Baseline_Standard (default), Progressive_3_Scan (only applicable if report format is images)
compression	optional, JPEG_low, JPEG_medium (default), JPEG_high (only applicable if report format is images)
imgformat	optional, JPEG, PNG, TIFF (default: JPEG) (only applicable if report format is images)
anchor	optional, page box anchor point used to align the two page images for comparison (default: upperleft) - lowerleft: Use lower left corner of anchor box of page - bottom: Use center of lower edge of anchor box of page - lowerright: Use lower right corner of anchor box of page - right: Use center of right edge of anchor box of page - upperright: Use upper right corner of anchor box of page - top: Use center of top edge of anchor box of page - upperleft: Use upper left corner of anchor box of page - left: Use center of left edge of anchor box of page - center: Use center of anchor box of page

anchorbox	optional, page box used to align the two page images for comparison (default: CROPBOX) - MEDIABOX: Use MediaBox of page - CROPBOX: Use CropBox of page - TRIMBOX: Use TrimBox of page - BLEEDBOX: Use BleedBox of page - ARTBOX: Use ArtBox of page
offset	optional, offset to anchor point used to align the two page images for comparison. (default: 0,0): <x-offset>[<unit>],<x-offset>[<unit>] - x-offset: Move second document to the right - y-offset: Move second document upwards - unit: Optional unit for offsets: mm, inch, pt (default: mm)
pagebox	optional, render pages using a page geometry box (default: CROPBOX) - MEDIABOX: Render MediaBox of page - CROPBOX: Render CropBox of page - TRIMBOX: Render TrimBox of page - BLEEDBOX: Render BleedBox of page - ARTBOX: Render ArtBox of page

Regular example

```
pdfToolbox --compare --threshold=2 --areathreshold=5 <PDF file 1> <PDF file 2>
```

Template example

```
pdfToolbox --compare --format=template:<path to template folder> <PDF file 1> <PDF file 2>
```

The resolutions of the preview images can be set in the Manifest.xml.Sample folders for a customised template can be downloaded here:



compare_templat_custom.zip

32.20 DeviceLink Conversion

DeviceLink profiles complement the usage of regular ICC profiles to avoid weaknesses mainly are the CMYK to CMYK transformation and e.g. the preservation of pure black text in a picture. A CMYK to CMYK transformation with ICC profiles is always performed via the device independent Lab color space, which leads to a complete reseparation of the data with partly unpredictable and unwanted results. This does not happen with DeviceLink profiles, which offer a direct control of color composition.

Using DeviceLink profiles

Performing a DeviceLink conversion with pdfToolbox CLI is rather simple.

Just set up a new profile with pdfToolbox Desktop and set up the fixup "Convert colors using DeviceLink profiles". Choose the desired profile from the drop down list and define if the conversion should only take place for a certain type of objects or color spaces. Then add the fixup "Embed Output Intent" with the desired settings.

No extra software installation is needed after entering the license string when purchasing the DeviceLink Add-on.

You will find more information on the provided DeviceLink profiles and their settings in the “callas pdfEngine Reference”.

Using your own profiles

You can also use your own DeviceLink profiles – no DeviceLink Add-on license is required in that case. For installation use the "Import" button at the bottom of the Fixup dialog "Convert colors using DeviceLink profile" and choose the profile location from your hard disc.

Additional display properties for the user interface of the Plug-In/Standalone can be defined in a XML-file.

32.21 Run as a Server

Start as a server

callas pdfToolbox Server/CLI can also be used for processing hotfolders on platforms, where no user interface for the configuration of the require settings is available (like Linux, SunSparc, SunIntel).

This possiblity to start a server without a user interface is available on MacOS and Windows also of course.

First, the pdfToolbox CLI has to be started in server mode:

```
--server [--quiet] [--accesskey=accesskey]
[--port=port] [--cachefolder=cachefolder]
```

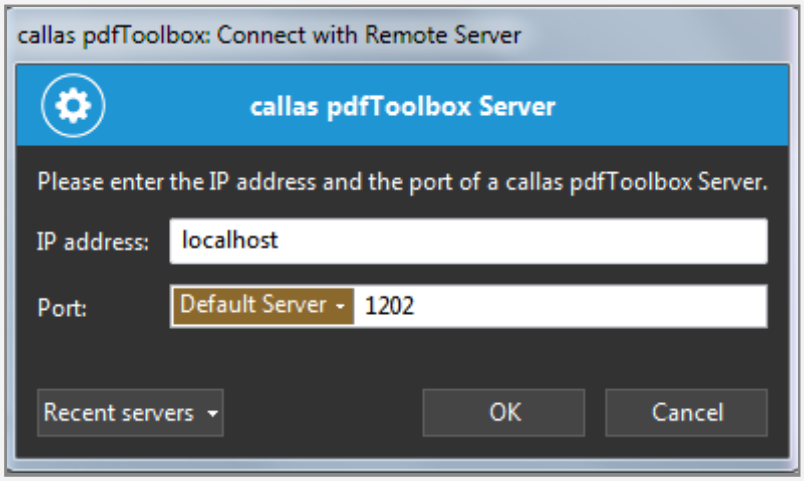
Options

--quiet	optional, suppresses output
--accesskey	optional, sets a accesskey for restricting the possibilty to change configuration using the server user interface
--port	optional, defines the port for communication between CLI and server user interface via the network (Default: 1202)
--cachefolder	optional, defines the path to cachefolder

Example:

```
--server --port=1202 --accesskey=123456
```

For setting up a job for hotfolder processing, connect to the remote server using any pdfToolbox desktop installation in the same network:



After entering the accesskey, new jobs can be configured, started or stopped. Profiles and Settings will be transfered to the remote server,

where the are stored at /usr/share/callas software (path must be writable)

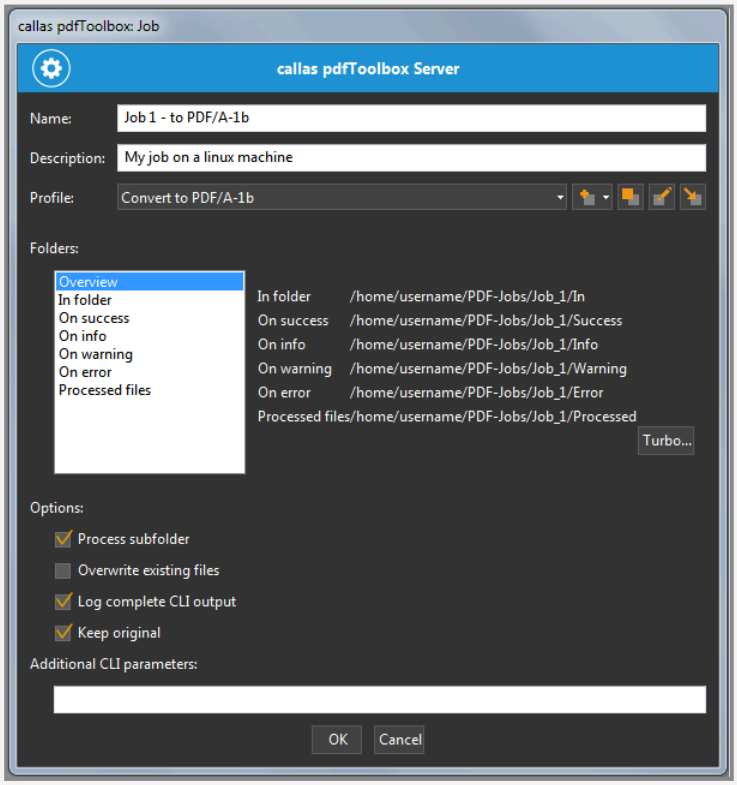
If this location can not be used caused by limitations on the respective environment, the additional option --cachefolder can be used for defining a custom path when starting the server:

Example:

```
--server --port=1202 --cachefolder=<PATH>
```

All paths defined for hotfolder processing need to be entered manually and have to be valid path specifications. (Hotfolder paths of any remote server jobs (IN, OUT, etc.) have to be configured so that they are valid from the service's perspective (the system where the service is running) - and not from the perspective of the controlling standalone application.)

The server can also be stopped by remote, but not started.



Preferences files of the Server

In general, there is no need to touch the preferences files of the Server.
Using MacOS and Windows, the Server settings from a previous version can be imported when starting a new major version the first time. Within updates in a generation of a major version (e.g. 10.0 to 10.1), there is no change regarding the configuration files in the preferences.

Nonetheless, if there is a need for a manual change to these files, these are the common locations for the preferences folder.

Please contact our support team if you have any questions about the files and there usage before changing them.

32.22 Distributed Processing

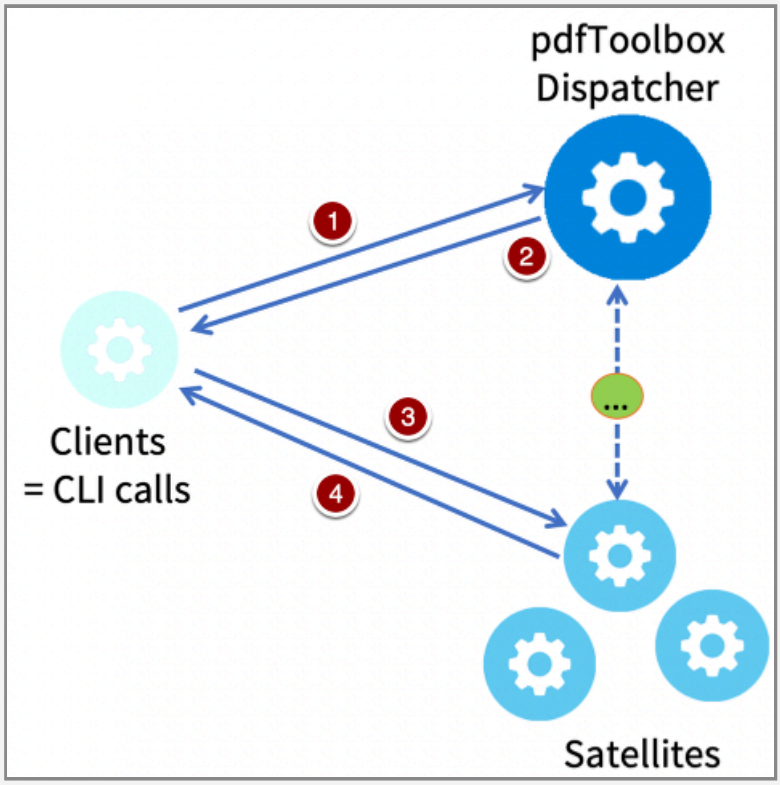
Distributed Processing mode

callas pdfToolbox Server/CLI offers a distributed processing mode, allowing tasks to be efficiently distributed across the network to multiple "satellite" instances. The outcomes are then returned back to the "origin of processing". As a result, pdfToolbox Server/CLI can be initiated in various operational modes:

- **"Dispatcher"** controls which tasks are to be processed by which machines: the "Satellites". There must always be at least one Dispatcher in the network.
- **"Satellite"** receives tasks from the "Clients" or directly from the Dispatcher (if the Dispatcher is running with hot-folders), processes them and sends them back to the "Clients".
- **"Client"** asks the Dispatcher for Satellites. After receiving an available Satellite, it sends the tasks to the Satellites and receives the results after processing.
- **"Monitor"** monitors the Dispatcher and displays the current situation.

All these components can run on the same machine or on different machines. There must be at least one Dispatcher and at least one Satellite in the network. At least one client is required to submit tasks.

Communication



1. Clients send a request for Satellite to Dispatcher
2. Dispatcher assigns a Satellite and sends the address to the Client
3. Client sends the task to the Satellite
4. Satellite returns result to client

Starting a Dispatcher

```
pdfToolbox --dispatcher [--port=<port number>]
```

Example:

```
pdfToolbox --dispatcher [--port=1234]
```

`port` is the port number on which the dispatcher can be called over the network. The dispatcher port is 1200 by default and can be optionally set to a different port (see example).

Starting a Dispatcher using the Server UI

It is also possible to start the server as a dispatcher on Windows and MacOS using the user interface (Desktop). Hotfolder processing can be set up here as well. In this mode, the dispatcher also distributes tasks sent by other clients.

Starting a Satellite

```
pdfToolbox --satellite --endpoint=<Dispatcher IP number>[:<dispatcher port]> [--
port=<port number>] [--connections=<number of concurrent connections]
```

Example with default Dispatcher:

```
pdfToolbox --satellite --endpoint=10.0.0.100
```

Example with different Dispatcher port:

```
pdfToolbox --satellite --endpoint=10.0.0.100:1234 --port=4321
```

At least one Satellite is required to process tasks.

`endpoint` is the IP number and port of the Dispatcher. The dispatcher port is 1200 by default, but can be changed when the dispatcher is started (see "Starting a Dispatcher"). If the default dispatcher port is used, the dispatcher port can be omitted.

`port` is the one used by the Satellite to communicate with the Clients. The Satellite port is 1201 by default and can be set to a different port at startup.

Starting a Satellite using the Server UI

It is also possible to start the server as a Satellite on Windows and MacOS using the user interface (Desktop). In this mode, the Satellite will not process any hotfolder jobs on the computer.A Satellite will always use the number of CPUs on the machine as the number of concurrent connections/processes. To limit this number, the Satellite must be started on the CLI with the `--connections` parameter.

The number of connections should not exceed the number of CPUs, as this can reduce performance per process and lead to system instability.

Assigning more than one Dispatcher to a Satellite

To connect a Satellite to more than one Dispatcher, it is possible to define more than one (--) endpoint. Please refer to the "Fallback for Dispatcher" section at the end of this chapter.

Distributing a process using a Client

The client is called using any regular pdfToolbox command line call.To distribute the call over the network, the command line parameters `--dist` and `--endpoint` are added. The client will first ask the dispatcher for a satellite connection, then send the command to the satellite and wait for the result to be sent back from the satellite.

```
pdfToolbox --dist --endpoint=<dispatcher IP number>[:<dispatcher port>] <any regular pdfToolbox call>
```

Examples:

```
pdfToolbox --dist --endpoint=10.0.0.100 <anyProfile.kfpx> <myPDF.pdf>
pdfToolbox --dist --endpoint=10.0.0.100 --redistill <myPDF.pdf>
```

Distributed processing: Variables and resources

When using normal Profiles, there is nothing to worry about when processing a file. All needed resources (like ICC profiles or "Place content"-Templates) are included in the Profile.

But sometimes some advanced scripting of e.g. a template requires external resources that are defined/referenced by a variable.To ensure that these resources are passed to the satellite during distributed processing, a variant of the `--setvariable=<variable>` option can be used:

```
--setvariable=RESOLUTION:300
```

```
--setvariablepath=<path to ressources file or folder>
```

Set the type of satellite (Optional)

Since some types of tasks should only be processed on a specific type of Satellite, it is possible to start a Satellite with one or more types set.

Each CLI call to process a task can be customized to one or more types of allowed Satellites.

Set typification for Satellite:

```
pdfToolbox --satellite --endpoint=<dispatcher IP number> --satellite_type=<type> [--satellite_type=<type>]
```

for example:

```
pdfToolbox --satellite --endpoint=10.0.0.100 --satellite_type=A
pdfToolbox --satellite --endpoint=10.0.0.100 --satellite_type=A --satellite_type=B
```

Set typification for Client:

```
pdfToolbox --dist --endpoint=<dispatcher IP number> --satellite_type=<type> [--satellite_type=<type>] <any regular pdfToolbox call>
```

for example:

```
pdfToolbox --dist --endpoint=10.0.0.100 --satellite_type=A <any regular pdfToolbox call>
```

Implementation details:

• If a Satellite has been started with a typification, only Client calls with the same type will be send to this Satellite.

• If a Client call contains multiple typifications, all typifications must match with those set for a satellite.

• If a Client call has no typification set, it can be processed on all satellites, even if they were started with a typification.

• The <type> string must be alpha-numeric and is case sensitive.

Disallow local processing

As a fallback, processing might happen locally (on the Client or on Dispatcher if run in hotfolder mode) if an action cannot be distributed, a Satellite cannot be assigned within a timeframe or if no Dispatcher is available.
Local processing might not be desired for several reasons.

To prevent such local processing, both the client call and the start of a Dispatcher (when used as a server for hotfolders) can be modified with the option:

```
--nolocal
```

Example for Client:

```
pdfToolbox --dist --endpoint=<dispatcher IP number> --nolocal <any regular pdfToolbox call>
```

Local processing will be disabled and tasks will fail if no Satellite is ready for processing.

Example for Dispatcher:

```
pdfToolbox --dispatcher --nolocal
```

Here `--nolocal` is forwarded to child processes for hotfolder jobs. It has no effect on the processing of non-hotfolder files from a Client distributed by the Dispatcher.
If a Client wants to disable local processing, the `--nolocal` setting has to be set in each CLI call of the Client.

Fallback for Dispatcher

Some workflow systems may require a fallback for a dispatcher to ensure production stability.To cover this, a number of Dispatchers can be set up to run individually. One or more Dispatchers can be assigned to a Satellite.

Define multiple Dispatcher for a Satellite

Connects a satellite to two (or more) Dispatchers.

```
pdfToolbox --satellite --endpoint=<dispatcher 1 IP> [--endpoint=<dispatcher 2 IP>
[--endpoint=<dispatcher IP>]
```

Set multiple Dispatcher in a Client call

Distributes a Client call via two (or more) Dispatcher. The first reachable Dispatcher with a free satellite will handle the task.

```
pdfToolbox --dist --endpoint=<dispatcher 1 IP> --endpoint=<dispatcher 2 IP> [--end-
point=<dispatcher IP>] <any regular pdfToolbox call>
```

Define a timeout for processing

In some workflow systems, long-running processes may not be allowed and must be terminated when a certain time frame is reached.

Due to the flexibility of distributed processing, a variety of timeouts can be set for each part:

- for the Client call
- for the Satellite
- for the Dispatcher

Timeout for processing on a Satellite

- If a timeout is defined for the client call, execution will be canceled after the specified time.
- If a timeout is defined when a Satellite is started, all tasks processed by that Satellite will be canceled after the specified time.
- If both are defined, the shorter timeout is used.

Example for Client:

```
pdfToolbox --dist --endpoint=<dispatcher IP> --timeout_satellite=<seconds> <any
regular pdfToolbox call>
```

Example for Satellite:

```
pdfToolbox --satellite --endpoint=<dispatcher IP> --timeout=<seconds>
```

Timeout for local processing of Dispatcher or Client

A processing timeout (if no satellite is available or if the type of task cannot be distributed) for the fallback to local processing on the Client or the Dispacher (when used as a server for hotfolders) can also be defined.
If both are defined, the shorter defined timeframe will be used.

Example for Client:

```
pdfToolbox --dist --endpoint=<dispatcher IP> --timeout=<seconds> <any regular
pdfToolbox call>
```

Example for Dispatcher:

```
pdfToolbox --dispatcher --timeout=<seconds>
```

Timeout for Dispatcher to search for Satellites

Additionally, a dispatcher timeout can be set, which defines the timeframe in which a satellite is searched. This can be set individually for each client call.

Example for Client:

```
pdfToolbox --dist --endpoint=<dispatcher IP> --timeout_dispatcher=<seconds> <any
regular pdfToolbox call>
```

When running the dispatcher in hotfolder mode, the setting can be specified when starting the dispatcher (will then affect all distributed files from hotfolders):

Example for Dispatcher:

```
pdfToolbox --dispatcher --timeout_dispatcher=<seconds>
```

If a satellite or dispatcher timeout is set and the --nolocal option is defined, the task will not be processed locally. Processing will result in an error.

Setting --timeout_... or --nolocal parameters in the Additional CLI Parameters section of the Server UI when defining hotfolder jobs is not supported.

Using the CLI-Monitor

```
pdfToolbox --monitor --endpoint=<dispatcher IP>:<dispatcher port> [--endpoint=<dis-
patcher IP>:<dispatcher port>]
```

Example:

```
pdfToolbox --monitor --endpoint=10.0.0.100
```

Monitor is optional and mirrors the dispatcher's command line output to another computer. Endpoint is the IP number and port of the dispatcher.
If more than one dispatcher is used, multiple dispatcher IPs can be entered and monitored.

Licensing

- Server: Regular pdfToolbox Server/CLI license required
- Dispatcher: Dispatcher pdfToolbox Server/CLI license required
- Satellite: Regular pdfToolbox Server/CLI license required
- Monitor: No license required
- Client: No license required

Distributed processing can be combined with the License Server in order to activate some or all components of these modules via this License Server. The setup is described here.

Distributed Processing in Enfocus Switch

Simply configure the appropriate settings in the Configurator for the steps to be distributed. If all tasks are to be processed on other machines (satellites), no local server license is required.
Some installations made better experiences, when the setting "Concurrent transfers to the same site" in Switch was set to "Automatic". Also the "Default number of slots for concurrent elements" should not be 0.

Known limitations until version 11.1

Due to technical limitatations of the used communication method (SOAP), it is not possible to process files greater than 2 GB using distributed processing until pdfToolbox v.11.1.

Since version 11.1, callas has removed this technical limitation. Until this version, files are processed locally on the client.

32.23 Running pdfToolbox via Web-services (SOAP)

It is possible to submit jobs to a pdfToolbox instance and retrieve the results via SOAP requests.

In order to do so you first have to start the pdfToolbox in a "listening mode":

```
./pdfToolbox --satellite --port=<any free port number, e.g. 1201>
```

Click to copy
Or on Windows:

```
.\pdfToolbox.exe --satellite --port=<any free port number, e.g. 1201>
```

Click to copy
It is currently not possible to submit PDF files or kfpx Profiles via such SOAP requests, so both have to be available to pdfToolbox on mounted volumes.

The SOAP requests wrap any command line parameters in <ns:args> tags. The simple requests below asks pdfToolbox to send usage information back by submitting the --help parameter.

Beginning from pdfToolbox 9:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ns="http://callassoftware.com/cws.xsd">
 <SOAP-ENV:Body>
  <ns:extExecute>
   <ns:args>
    <ns:userID></ns:userID>
    <ns:args>--noprogress</ns:args>
    <ns:args>--nosummary</ns:args>
       <ns:args>--nohits</ns:args>
    <ns:args>-o=/AppData/pdfToolbox/WebService/sample_result.pdf</ns:args>
```

```
    <ns:args>-r=XML,ALWAYS,ALL,PATH=/AppData/pdfToolbox/WebService/sample_result.
xml</ns:args>
    <ns:args>/Applications/callas pdfToolbox Server 10/cli/var/Profiles/PDFX com-
pliance/Convert to PDFX-1a (ISO Coated v2 (ECI)).kfpx</ns:args>
    <ns:args>/AppData/pdfToolbox/WebService/sample.pdf</ns:args>
   </ns:args>
  </ns:extExecute>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Click to copy

Beginning from pdfToolbox 10:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ns="http://callassoftware.com/cws.xsd">
 <SOAP-ENV:Body>
  <ns:extExecute>
   <args>
    <userID></userID>
    <args>--noprogress</args>
    <args>--nosummary</args>
       <args>--nohits</args>
    <args>-o=/AppData/pdfToolbox/WebService/sample_result.pdf</args>
    <args>-r=XML,ALWAYS,ALL,PATH=/AppData/pdfToolbox/WebService/sample_result.
xml</args>
    <args>/Applications/callas pdfToolbox Server 10/cli/var/Profiles/PDFX compli-
ance/Convert to PDFX-1a (ISO Coated v2 (ECI)).kfpx</args>
    <args>/AppData/pdfToolbox/WebService/sample.pdf</args>
   </args>
  </ns:extExecute>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Click to copy

You may include as many parameters as required, there is no limitation.

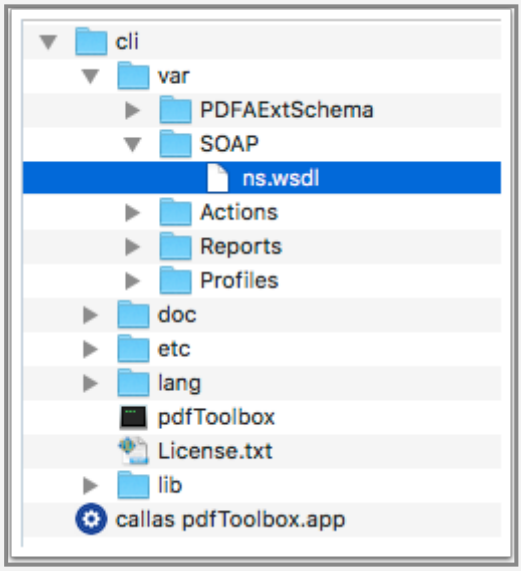
Any results will be sent back in XML back to the program that you have used to send the request.

You may use the curl tool on command line to send requests and retrieve results:

```
curl -H "Content-Type: text/xml; charset=utf8" -d@.\ns.extExecute_help.req.xml
http://<pdfToolbox' IP>:<Port>
```

Click to copy

A Web Service Defintion Language file can be found in the pdfToolbox program folder:



The example below is a simple SOAP request that asks pdfToolbox to create single pages from the original PDF file. Path and name of the PDF need of course to be adapted.

ns.extExecute_makesinglepages.req.xml

Getting progress

If you want to get progress for a longer running action, this can be done as well. When making the original request, make sure the userID element is filled with a unique identifier for the call you're making.

```
...
  <ns:extExecute>
```

```
   <args>
    <userID>Job777</userID>
    <args>--noprogress</args>
    <args>--nosummary</args>
       <args>--nohits</args>
    <args>-o=/AppData/pdfToolbox/WebService/sample_result.pdf</args>
    <args>-r=XML,ALWAYS,ALL,PATH=/AppData/pdfToolbox/WebService/sample_result.
xml</args>
    <args>/Applications/callas pdfToolbox Server 10/cli/var/Profiles/PDFX compli-
ance/Convert to PDFX-1a (ISO Coated v2 (ECI)).kfpx</args>
    <args>/AppData/pdfToolbox/WebService/sample.pdf</args>
   </args>
  </ns:extExecute>
...
```

Click to copy

The unique identifier ("Job777" in the above example), can then be used to get progress information through a REST call:

```
http://<HOST:PORT>/exExecute/<USERID>/progress
```

Click to copy

32.24 Activating logging

Purpose of logging feature

Especially in higher volume environments it can be of interest to have detailed information about the pdfToolbox processes as they happen. Suitable logging data can be a good foundation both for monitoring processing of PDF files as well as for post mortem analyses or analyses aiming to improve overall behavior of the whole processing system.

The logging feature that forms a part of pdfToolbox provides logging information in the form of JSON files that get written immediately at the following steps during an invocation of pdfToolbox:

- When pdfToolbox is **launched**; only very basic data about the PDF file to be produced – such as file name – is available at this time, plus some information about the environment or the command line parameters
- When pdfToolbox **initiates processing** of a PDF file; additional data about the PDF to be processed is provided – only aspects though that do not require no 'deeper' analysis in order not to hamper runtime behavior – such as file size or PDF version or size of the first page; in addition all variables – as evaluated at the beginning of processing – are included as well.
- When pdfToolbox **completes** processing of a PDF file; at this stage various aspects of overview information is readily available, such as colors, fonts or ICC profiles used, output intents, conformance with PDF/X or other PDF standards.

Command line and Server versus Desktop

Logging is available in command line, Server and Desktop versions. It is not expected that logging on desktop is normally being used. Instead, on Desktop activating logging will help understand how logging works, and often makes it easier to design and test use of logging data outside of an automated production setup.

How to process logging data

Research has shown that most callas software customers who operate higher volume environments prefer any logging data to be provided rather as many small files using JSON over using other formats or a small number of large files. Beyond the way the logging data is provided, no assumptions are being made how these data are processed.

Activating logging in command line version

There are 2 parameters that can be used to activate logging in the CLI version of callas products:

- --trace
- --logexecution

--trace

Parameter:

```
--trace[=folderpath]
```

Example:

```
./pdfToolbox --trace='/some-logging-folder' './some-pdftoolbox-profile.kfpx' './input.pdf' -o='./output.pdf'
```

Subfolder structure

By default the logging feature creates a subfolder structure inside the target folder for logging based on values for year, month and day, plus the `app_uuid` value (which can also be found inside each of the logging files):

```
YYYY
  MM
```

```
    DD
      app_uuid
        launch.json
        init.json
        finish.json
```

Click to copy

```
2016
  10
    31
      0a0ceba3-7ca9-421f-a973-8caae2950690
        launch.json
        init.json
        finish.json
```

Click to copy

The JSON log files

There is an extensive article about the 3 JSON logging files, which will be created:

The JSON log files

Suppressing creation of subfolder structure

Parameter:

```
--trace_nosubfolders
```

Example:

```
./pdfToolbox --trace='/some-logging-folder'  --trace_nosubfolders './some-pdftool-box-profile.kfpx'   './input.pdf'  -o='./output.pdf'
```

Activating --trace logging in Server version

In the "pdfToolbox Server: Job" configuration window, add the following to the text field under "Additional CLI parameters":

```
--trace[=folderpath]
```

Example:

```
--trace='/some-logging-folder'
```

Suppressing creation of subfolder structure in Server version

In the "pdfToolbox Server: Job" configuration window, add the following to the text field under "Additional CLI parameters":

```
--trace[=folderpath] --trace_nosubfolders
```

Example:

```
--trace='/some-logging-folder'  --trace_nosubfolders
```

Important note: If logging is to be enabled for processing of all files in pdfToolbox Server, this setting must be added to each job configuration.

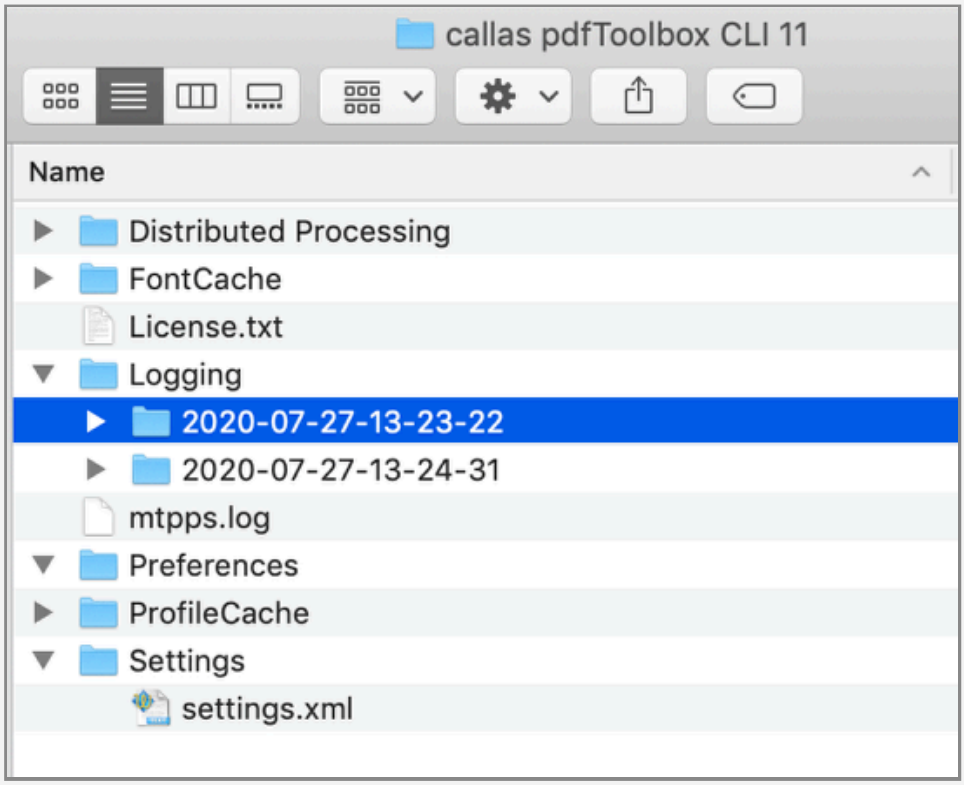
--logexecution

Works similar to the "Log Profile Exceution" function on the desktop application, where logs are created during Profile execution into a separate folder.

Parameter usage:

```
./pdfToolbox "Place content.kfpx" "simple sample file.pdf" --logexecution
```

Please note that for --logexecution, the resultant logging folder is written into the User Preferences folder.


callas pdfToolbox CLI 11
Name
Distributed Processing
FontCache
License.txt
Logging
2020-07-27-13-23-22
2020-07-27-13-24-31
mtpps.log
Preferences
ProfileCache
Settings
settings.xml

32.25 Predefined Profiles

Below is a list of all predefined Profiles that are shipped with callas SDK/CLI. You can find these Profiles in the package folder under:

var/Profiles

For the use of Profiles in web applications, all Profiles are also available in a cloud space. The path for this can also be found in the list below.

List of Profiles based on their categories are listed below:

- Convert colors
- Convert colors using DeviceLink
- Create PDF layers
- Digital printing and online publishing
- GWG 2022
- PDF
- PDF analysis
- PDF Fixups
- PDF version compatibility
- PDFA compliance
- PDFE compliance
- PDFUA compliance
- PDFVT compliance
- PDFX compliance
- PDFX and PDFA
- Packaging
- Place content
- Preflight Certificate
- Prepress
- Process Plans
- Processing Steps

Group: Convert colors

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
Comment	Converts RGB objects to CMYK (PSO Coated v3) with a specified tolerance for R=G=B, i.e. objects that are converted to CMYK gray. Spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Con-

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
	vert_RGB_to_CMYK_(PSO_Coated_v3)_with_specified_R%3DG%3DB_tolerance.kfpx
Name	Convert registration color to CMYK black only
Comment	Converts all registration color objects to DeviceCMYK black only.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_registration_color_to_CMYK_black_only.kfpx
Name	**Convert spot to CMYK and replace overprint with transparency where needed**
Comment	When spot color objects are converted to CMYK the overprint rules change which might lead to a dramatically changed appearance.This Process Plan analyzes objects in context and replaces overprint with transparency where needed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_spot_to_CMYK_and_replace_overprint_with_transparency_where_needed.kfpx
Name	**Convert to CMYK only (Coated GRACoL 2006)**
Comment	Converts the current PDF document to CMYK only, using Coated GRACoL 2006 as destination profile. All spot colors are converted to CMYK as well.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK_only_(Coated_GRACoL_2006).kfpx
Name	**Convert to CMYK only (ISO Coated v2 (ECI))**
Comment	Converts the current PDF document to CMYK only, using ISO Coated v2 (ECI) as destination profile. All spot colors are converted to CMYK as well.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK_only_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to CMYK only (Japan Color 2001 Coated)**
Comment	Converts the current PDF document to CMYK only, using Japan Color 2001 Coated as destination profile. All spot colors are converted to CMYK as well.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK_only_(Japan_Color_2001_Coated).kfpx
Name	Convert color to PSO Coated v3 (ECI) (convert spot colors to CMYK)

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
Comment	Converts the current PDF document to CMYK only, using PSO Coated v3 (ECI) as destination profile. All spot colors are converted to CMYK as well.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK_only_(PSO_Coated_v3_(ECI).kfpx
Name	**Convert to CMYK, keep spot colors (Coated GRACoL 2006)**
Comment	Converts the current PDF document to CMYK, using Coated GRACoL 2006 as destination profile. All spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK%2C_keep_spot_colors_(Coated_GRACoL_2006).kfpx
Name	**Convert to CMYK, keep spot colors (ISO Coated v2 (ECI))**
Comment	Converts the current PDF document to CMYK, using ISO Coated v2 (ECI) as destination profile. All spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK%2C_keep_spot_colors_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to CMYK, keep spot colors (Japan Color 2001 Coated)**
Comment	Converts the current PDF document to CMYK, using Japan Color 2001 Coated as destination profile. All spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK%2C_keep_spot_colors_(Japan_Color_2001_Coated).kfpx
Name	**Convert color to PSO Coated v3 (ECI) (keep s**pot colors)
Comment	Converts the current PDF document to CMYK, using PSO Coated v3 (ECI) as destination profile. All spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK%2C_keep_spot_colors_(PSO_Coated_v3_(ECI)).kfpx
Name	**Convert to grayscale**
Comment	Converts all colors (including spot colors) in the current PDF document to grayscale.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_grayscale.kfpx

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
Name	**Convert to sRGB**
Comment	Converts all colors (including spot colors) in the current PDF document to sRGB.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_sRGB.kfpx
Name	**Convert using DeviceLink Office RGB to ISO Coated v2 (ECI)**
Comment	Converts the current PDF document to CMYK (ISO Coated v2 (ECI)) as destination profile according to the Fogra PSD (Process Standard Digital) guidelines. A special DeviceLink profile makes sure that line art elements defined in RGB keep their vivid color and that colors which are different in the original PDF have similar distances in the result. RGB Gray gets converted to pure Black, Black text is set to overprint. The DeviceLink profile has been developed for preparation files created with office applications for the print process.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_using_DeviceLink_Office_RGB_to_ISO_Coated_v2_(ECI).kfpx
Name	**Map color with specified color values**
Comment	Maps colors in the current PDF document based on parameters specified during runtime.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Map_color_with_specified_color_values.kfpx
Name	**Overprint issues related to spot color if converted to CMYK**
Comment	The Profile identifies overprinting or overprinted spot color objects that are problematic when converted to CMYK. The underlying reason is that the overprint rules are much different for spot and CMYK objects.Two problematic areas are analyzed: 1.) OPM is off and 2.) OPM is on and at least one colorant (C, M, Y or K) is used by foreground and background object.These two areas are analyzed for spot color objects in front of or behind CMYK objects or other spot color objects.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Overprint_issues_related_to_spot_color_if_converted_to_CMYK.kfpx
Name	**Tone value adjustment in midtones by -10%**
Comment	Reduces the tone values in midtones by 10% for device dependent CMYK and spot colors.

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Tone_value_adjustment_in_midtones_by_-10prct.kfpx

Group: Convert colors using DeviceLink

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Comment	Converts objects from CMYK exchange color space (eciCMYK) to Sheetfed offset (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/CMYK_exchange_CS_to_Sheetfed_(DL_eciCMYK_to_ISOcoatedv2).kfpx
Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to PSO Coated v3)**
Comment	Converts objects from CMYK exchange color space (eciCMYK) to Sheetfed offset (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/CMYK_exchange_CS_to_Sheetfed_(DL_eciCMYK_to_PSOcoatedv3).kfpx
Name	**RGB to CMYK (DeviceLink: Adobe RGB to ISO Coated v2)**
Comment	Converts objects from RGB (Adobe RGB) to CMYK (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/RGB_to_CMYK_(DL_AdobeRGB_to_ISOcoatedv2).kfpx
Name	**RGB to CMYK (DeviceLink: AdobeRGB to PSO Coated v3)**
Comment	Converts objects from RGB (Adobe RGB) to CMYK (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/RGB_to_CMYK_(DL_AdobeRGB_to_PSOcoatedv3).kfpx
Name	**RGB to CMYK (DeviceLink: sRGB to ISO Coated v2)**
Comment	Converts objects from RGB (sRGB) to CMYK (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/RGB_to_CMYK_(DL_sRGB_to_ISOcoatedv2).kfpx
Name	**RGB to CMYK (DeviceLink: sRGB to PSO Coated v3)**
Comment	Converts objects from RGB (sRGB) to CMYK (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/RGB_to_CMYK_(DL_sRGB_to_PSOcoatedv3).kfpx
Name	**Sheetfed offset to CMYK exchange color space (DeviceLink: ISO Coated v2 to eciCMYK)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to CMYK exchange color space (eciCMYK) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_CMYK_exchange_CS_(DL_ISOcoatedv2_to_eciCMYK).kfpx
Name	**Sheetfed offset to CMYK exchange color space (DeviceLink: PSO Coated v3 to eciCMYK)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to CMYK exchange color space (eciCMYK) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_CMYK_exchange_CS_(DL_PSOcoatedv3_to_eciCMYK).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: GRACoL2006 to Web Coated SWOP grade 3)**
Comment	Converts objects from Sheetfed offset (GRACoL 2006 Coated 1v2) to Web offset (Web Coated SWOP grade 3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_GRACoL2006_to_WebcoatedSWOPgrade3).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: GRACoL2006 to Web coated SWOP grade 5)**
Comment	Converts objects from Sheetfed offset (GRACoL 2006 Coated 1v2) to Web offset (Web Coated SWOP grade 5) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_GRACoL2006_to_WebcoatedSWOPgrade5).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to ISOnewspaper26v4)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (ISOnewspaper26v4) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_ISOnewspaper26v4).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to ISO Web Coated)**

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (ISO Web Coated) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_ISOwebcoated).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to PSO LWC Improved)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (PSO LWC Improved) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_PSOLWCimproved).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to PSO LWC Standard)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (PSO LWC Standard) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_PSOLWCstandard).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to PSO SC-B paper v3 (FOGRA54))**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (PSO SC-B paper v3 (FOGRA54)) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_PSOSC-Bpaperv3).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to SC Paper)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (SC Pa-

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	per) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_SCpaper).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to ISOnewspaper26v4)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (ISOnewspaper26v4) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_ISOnewspaper26v4).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to ISO Web Coated)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (ISO Web Coated) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_ISOwebcoated).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to PSO LWC Improved)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (PSO LWC Improved) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_PSOLWCimproved).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to PSO LWC Standard)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (PSO

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	LWC Standard) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_PSOLWCstandard).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to PSO SC-B Paper v3)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (PSO SC-B paper v3 (FOGRA54)) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_PSOSC-Bpaperv3).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to SC Paper)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (SC Paper) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_SCpaper).kfpx
Name	**Sheetfed offset, Europe to Japan (DeviceLink: ISO Coated v2 to Japan Color 2011 Coated)**
Comment	Converts objects from Europe (ISO Coated v2) to Japan (Japan Color 2011 Coated) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Europe_to_Japan_(DL_ISOcoatedv2_to_Japan2011Coated).kfpx
Name	**Sheetfed offset, Europe to Japan (DeviceLink: PSO Coated v3 to Japan Color 2011 Coated)**
Comment	Converts objects from Europe (PSO Coated v3) to Japan (Japan Color 2011

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	Coated) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Europe_to_Japan_(DL_PSOcoatedv3_to_Japan2011Coated).kfpx
Name	**Sheetfed offset, Europe to US (DeviceLink: ISO Coated v2 to GRACoL2006)**
Comment	Converts objects from Europe (ISO Coated v2) to US (GRACol 2006 Coated 1v2) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Europe_to_US_(DL_ISOcoatedv2_to_GRACoL2006).kfpx
Name	**Sheetfed offset, Europe to US (DeviceLink: PSO Coated v3 to GRACoL2006)**
Comment	Converts objects from Europe (PSO Coated v3) to US (GRACol 2006 Coated 1v2) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Europe_to_US_(DL_PSOcoatedv3_to_GRACoL2006).kfpx
Name	**Sheetfed offset, Japan to Europe (DeviceLink: Japan Color 2011 Coated to ISO Coated v2)**
Comment	Converts objects from Japan (Japan Color 2011 Coated) to Europe (ISO Coated v2) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Japan_to_Europe_(DL_Japan2011Coated_to_ISOcoatedv2).kfpx

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Name	**Sheetfed offset, Japan to Europe (DeviceLink: Japan Color 2011 Coated to PSO Coated v3)**
Comment	Converts objects from Japan (Japan Color 2011 Coated) to Europe (PSO Coated v3) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Japan_to_Europe_(DL_Japan2011Coated_to_PSOcoatedv3).kfpx
Name	**Sheetfed offset, US to Europe (DeviceLink: GRACoL2006 to ISO Coated v2)**
Comment	Converts objects from US (GRACol 2006 Coated 1v2) to Europe (ISO Coated v2) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_US_to_Europe_(DL_GRACoL2006_to_ISOcoatedv2).kfpx
Name	**Sheetfed offset, US to Europe (DeviceLink: GRACoL2006 to PSO Coated v3)**
Comment	Converts objects from US (GRACol 2006 Coated 1v2) to Europe (PSO Coated v3) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_US_to_Europe_(DL_GRACoL2006_to_PSOcoatedv3).kfpx
Name	**Sheetfed offset, coated paper (DeviceLink: ISO Coated v2 to PSO Coated v3)**
Comment	Converts objects from Sheetfed offset according to ISO Coated v2 to PSO Coated v3 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_coated_paper_(DL_ISOcoatedv2_to_PSOcoatedv3).kfpx
Name	**Sheetfed offset, coated paper (DeviceLink: PSO Coated v3 to ISO Coated v2)**
Comment	Converts objects from Sheetfed offset according to PSO Coated v3 to ISO Coated v2 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed,_coated_paper_(DL_PSOcoatedv3_to_ISOcoatedv2).kfpx
Name	**Sheetfed offset, coated to uncoated (DeviceLink: PSO Coated v3 to PSO Uncoated v3)**
Comment	Converts objects from Sheetfed offset, coated paper (PSO Coated v3) to Sheetfed offset, uncoated paper (PSO Uncoated v3 (FOGRA52)) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_coated_to_uncoated_(DL_PSOcoatedv3_to_PSOuncoatedv3).kfpx
Name	**Sheetfed offset, reduce to 280% ink limit (DeviceLink: ISO Uncoated)**
Comment	Reduces the ink coverage to 280% for Sheetfed offset (ISO Uncoated) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_280%25_ink_limit_(DL_ISOuncoated).kfpx
Name	**Sheetfed offset, reduce to 280% ink limit (DeviceLink: PSO Uncoated v3)**
Comment	Reduces the ink coverage to 280% for Sheetfed offset (PSO Uncoated v3 (FOGRA52)) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_280%25_ink_limit_(DL_PSOuncoatedv3).kfpx
Name	**Sheetfed offset, reduce to 300% ink limit (DeviceLink: ISO Coated v2)**
Comment	Reduces the ink coverage to 300% for Sheetfed offset (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_300%25_ink_limit_(DL_ISOcoatedv2).kfpx
Name	**Sheetfed offset, reduce to 300% ink limit (DeviceLink: PSO Coated v3)**
Comment	Reduces the ink coverage to 300% for Sheetfed offset (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_300%25_ink_limit_(DL_PSOcoatedv3).kfpx
Name	**Sheetfed offset, reduce to 320% ink limit (DeviceLink: GRACoL2006)**
Comment	Reduces the ink coverage to 320% for Sheetfed offset (GRACoL 2006 Coated 1v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_320%25_ink_limit_(DL_GRACoL2006).kfpx
Name	**Sheetfed offset, reduce to 330% ink limit (DeviceLink: ISO Coated v2)**
Comment	Reduces the ink coverage to 330% for Sheetfed offset (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_330%25_ink_limit_(DL_ISOcoatedv2).kfpx

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Name	**Sheetfed offset, reduce to 340% ink limit (DeviceLink: Japan Color 2011)**
Comment	Reduces the ink coverage to 340% for Sheetfed offset (Japan Color 2011) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_340%25_ink_limit_(DL_Japan2011Coated).kfpx
Name	**Sheetfed offset, uncoated paper (DeviceLink: PSO Uncoated v2 to PSO Uncoated v3)**
Comment	Converts objects from Sheetfed offset, uncoated paper according to PSO Uncoated v2 to PSO Uncoated v3 (FOGRA52) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_uncoated_paper_(DL_PSOuncoatedv2_to_PSOuncoatedv3).kfpx
Name	**Sheetfed offset, uncoated paper (DeviceLink: PSO Uncoated v3 to PSO Uncoated v2)**
Comment	Converts objects from Sheetfed offset, uncoated paper according to PSO Uncoated v3 (FOGRA52) to PSO Uncoated v2 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_uncoated_paper_(DL_PSOuncoatedv3_to_PSOuncoatedv2).kfpx
Name	**Web offset (DeviceLink: ISO Web Coated to ISOnewspaper26v4)**
Comment	Converts objects from Web offset according to ISO Web Coated to ISOnewspaper26v4 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web_(DL_ISOwebcoated_to_ISOnewspaper26v4).kfpx
Name	**Web (DeviceLink: PSOLWCimproved to ISOnewspaper26v4)**
Comment	Converts objects from Web offset according to PSO LWC Improved to ISOnewspaper26v4 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web_(DL_PSOLWCimproved_to_ISOnewspaper26v4).kfpx
Name	**Web offset (DeviceLink: Web Coated SWOP grade 3 to Web Coated SWOP grade 5)**
Comment	Converts objects from Web offset according to Web Coated SWOP grade 3 to Web Coated SWOP grade 5 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web_(DL_WebcoatedSWOPgrade3_to_WebcoatedSWOPgrade5).kfpx
Name	**Web offset to Sheetfed offset (DeviceLink: PSO SC-B paper v3 (FOGRA54) to ISO Coated v2)**
Comment	Converts objects from Web offset (PSO SC-B paper v3 (FOGRA54)) to Sheetfed offset (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web_to_Sheetfed_(DL_PSOSC-Bpaperv3_to_ISOcoatedv2).kfpx
Name	**Web offset to Sheetfed offset (DeviceLink: PSO SC-B paper v3 (FOGRA54) to PSO Coated v3)**
Comment	Converts objects from Web offset (PSO SC-B paper v3 (FOGRA54)) to Sheetfed offset (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_us-

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	ing_DeviceLink/Web_to_Sheetfed_(DL_PSOSC-Bpaperv3_to_PSOcoatedv3).kfpx
Name	**Web offset, reduce to 240% ink limit (DeviceLink: ISOnewspaper26v4)**
Comment	Reduces the ink coverage to 240% for Web offset (ISOnewspaper26v4) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_240%25_ink_limit_(DL_ISOnewspaper26v4).kfpx
Name	**Web offset, reduce to 280% ink limit (DeviceLink: Web Coated SWOP grade 5)**
Comment	Reduces the ink coverage to 280% for Web offset (Web Coated SWOP grade 5) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_280%25_ink_limit_(DL_WebcoatedSWOP-grade5).kfpx
Name	**Web offset, reduce to 300% ink limit (DeviceLink: ISO Web Coated)**
Comment	Reduces the ink coverage to 300% for Web offset (ISO Web Coated) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_300%25_ink_limit_(DL_ISOwebcoated).kfpx
Name	**Web offset, reduce to 300% ink limit (DeviceLink: PSO LWC Improved)**
Comment	Reduces the ink coverage to 300% for Web offset (PSO LWC Improved) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_300%25_ink_limit_(DL_PSOLWCimproved).kfpx

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Name	**Web offset, reduce to 300% ink limit (DeviceLink: PSO LWC Standard)**
Comment	Reduces the ink coverage to 300% for Web offset (PSO LWC Standard) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_300%25_ink_limit_(DL_PSOLWCstandard).kfpx
Name	**Web offset, reduce to 300% ink limit (DeviceLink: Web Coated SWOP grade 3)**
Comment	Reduces the ink coverage to 300% for Web offset (Web Coated SWOP grade 3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_300%25_ink_limit_(DL_WebcoatedSWOP-grade3).kfpx

Group: Create PDF layers

Name	Create separate layers for vector objects, text and images
Comment	Puts page objects on separate layers for vector objects, text and images.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Create_separate_layers_for_vector_objects%2C_text_and_images.kfpx
Name	Put all image objects on layers
Comment	Puts images on separate layers for color, grayscale and bitmaps.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Put_all_image_objects_on_layers.kfpx
Name	Put all text objects on a layer
Comment	Puts all text objects on a new layer.

Name	Create separate layers for vector objects, text and images
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Put_all_text_objects_on_a_layer.kfpx
Name	Put all transparent objects on layers
Comment	Puts all transparent objects on new layers according to the type of transparency (blend mode, constant alpha etc.)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Put_all_transparent_objects_on_layers.kfpx
Name	Put objects on layers according to their overprint settings
Comment	Creates layers with overprinting objects and objects that are set to knock out.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Put_objects_on_layers_according_to_their_overprint_settings.kfpx

Group: Digital printing and online publishing

Name	Convert for Digital printing (B_W)
Comment	Optimizes the current PDF document for digital printing. Converts all colors - including spot colors - to grayscale.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Digital_printing_and_online_publishing/Digital_printing_(B_W).kfpx
Name	Convert for Digital printing (color)
Comment	Optimizes the current PDF document for digital printing. Converts all colors - including spot colors - to CMYK.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Digital_printing_and_online_publishing/Digital_printing_(color).kfpx
Name	Convert for Online publishing
Comment	Converts the PDF for online publishing with optimization for size. Most PDFs will be significantly smaller then as all content not required for this purpose will be discarded and image resolution will be downsampled to 96 ppi. This approach makes the PDF as small as possible.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Digital_print-

Name	Convert for Digital printing (B_W)
	ing_and_online_publishing/Online_publishing.kfpx

Group: GWG 2022

Name	Digital Print
Comment	Based on GWG 2022 specifications. Variant: Digital Print
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Digital_Print.kfpx
Name	Large Format Print
Comment	Based on GWG 2022 specifications. Variant: Large Format Print
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Large_Format_Print.kfpx
Name	Magazine Ads CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: Magazine Ads CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Magazine_Ads_CMYK_and_RGB.kfpx
Name	Magazine Ads CMYK
Comment	Based on GWG 2022 specifications. Variant: Magazine Ads CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Magazine_Ads_CMYK.kfpx
Name	Newspaper Ads CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: Newspaper Ads CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Newspaper_Ads_CMYK_and_RGB.kfpx
Name	Newspaper Ads CMYK
Comment	Based on GWG 2022 specifications. Variant: Newspaper Ads CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Newspa-

Name	Digital Print
	per_Ads_CMYK.kfpx
Name	Packaging Flexo - Corrugated Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Corrugated Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Corrugated_Box.kfpx
Name	Packaging Flexo - Corrugated Display
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Corrugated Display
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Corrugated_Display.kfpx
Name	Packaging Flexo - Flexible
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Flexible
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Flexible.kfpx
Name	Packaging Flexo - Folding Carton Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Folding Carton Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Folding_Carton_Box.kfpx
Name	Packaging Flexo - Label
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Label
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Label.kfpx
Name	Packaging Flexo - Leaflet
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Leaflet
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Leaflet.kfpx
Name	Packaging Gravure - Corrugated Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Corrugated Box

Name	Digital Print
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Corrugated_Box.kfpx
Name	Packaging Gravure - Corrugated Display
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Corrugated Display
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Corrugated_Display.kfpx
Name	Packaging Gravure - Flexible
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Flexible
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Flexible.kfpx
Name	Packaging Gravure - Folding Carton Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Folding Carton Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Folding_Carton_Box.kfpx
Name	Packaging Gravure - Label
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Label
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Label.kfpx
Name	Packaging Gravure - Leaflet
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Leaflet
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Leaflet.kfpx
Name	Packaging Offset - Corrugated Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Corrugated Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Corrugated_Box.kfpx
Name	Packaging Offset - Corrugated Display

Name	Digital Print
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Corrugated Display
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Corrugated_Display.kfpx
Name	Packaging Offset - Flexible
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Flexible
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Flexible.kfpx
Name	Packaging Offset - Folding Carton Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Folding Carton Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Folding_Carton_Box.kfpx
Name	Packaging Offset - Label
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Label
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Label.kfpx
Name	Packaging Offset - Leaflet
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Leaflet
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Leaflet.kfpx
Name	SheetCMYK CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: SheetCMYK CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/SheetCMYK_CMYK_and_RGB.kfpx
Name	SheetCMYK CMYK
Comment	Based on GWG 2022 specifications. Variant: SheetCMYK CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/SheetCMYK_CMYK.kfpx

Name	Digital Print
Name	SheetSpot CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: SheetSpot CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/SheetSpot_CMYK_and_RGB.kfpx
Name	SheetSpot CMYK
Comment	Based on GWG 2022 specifications. Variant: SheetSpot CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/SheetSpot_CMYK.kfpx
Name	WebCMYK CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: WebCMYK CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebCMYK_CMYK_and_RGB.kfpx
Name	WebCMYK CMYK
Comment	Based on GWG 2022 specifications. Variant: WebCMYK CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebCMYK_CMYK.kfpx
Name	WebCMYKNews CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: WebCMYKNews CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebCMYKNews_CMYK_and_RGB.kfpx
Name	WebCMYKNews CMYK
Comment	Based on GWG 2022 specifications. Variant: WebCMYKNews CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebCMYKNews_CMYK.kfpx
Name	WebSpot CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: WebSpot CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Web-

Name	Digital Print
	Spot_CMYK_and_RGB.kfpx
Name	WebSpot CMYK
Comment	Based on GWG 2022 specifications. Variant: WebSpot CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebSpot_CMYK.kfpx

Group: PDF

Name	Uses PDF 2.0 features
Comment	Determines whether a PDF uses any print relevant features of PDF 2.0. Usually these features will be ignored by tools or devices that are not PDF 2.0 ready.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF/Uses_PDF_2.0_features.kfpx

Group: PDF analysis

Name	Analyze pages for effectively used plates
Comment	Analyzes all pages for effectively used plates. All pages are internally rendered, a plate is considered as being used if the minimum ink amount within at least one rectangle of 10 by 10 mm is reached. The threshold for the minimum ink amount can be specified during runtime. Device independent and all RGB colors should be converted into DeviceCMYK or DeviceGray beforehand.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/Analyze_pages_for_effectively_used_plates.kfpx
Name	Find CxF issues
Comment	CxF is an ISO standards that defines how to embed spectral measurements into a PDF Output Intent. This profile checks for some CxF related issues.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/Find_CxF_issues.kfpx
Name	List all hairlines

Name	Analyze pages for effectively used plates
Comment	CxF is an ISO standards that defines how to embed spectral measurements into a PDF Output Intent. This profile checks for some CxF related issues.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_all_hairlines.kfpx
Name	List black objects set to overprint
Comment	Reports all black objects that are set to overprint.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_black_objects_set_to_overprint.kfpx
Name	List color and grayscale images not within 250 to 450 ppi
Comment	Lists color and grayscale images with a resolution below 250 ppi and above 450 ppi. It depends on the image content whether this is sufficient.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_color_and_grayscale_images_not_within_250_to_450_ppi.kfpx
Name	**List invisible text objects**
Comment	Reports all text objects that are invisible. Invisible text is often used to overlay text (created by an OCR process) on top of scanned document pages.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_invisible_text_objects.kfpx
Name	**List objects using ICC/Lab/calibrated color**
Comment	Reports all page objects that use device independent color spaces (i.e. ICC based color spaces, Lab, CalRGB or CalGray color spaces).
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_objects_using_ICC_or_Lab_or_calibrated_color.kfpx
Name	**List page objects, grouped by type of object**
Comment	Reports page objects grouped by object type. Images are grouped by image resolution. This profile should only be used for one page at a time, as it may report a very large number of page objects.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_page_objects%2C_grouped_by_type_of_object.kfpx

Name	Analyze pages for effectively used plates
Name	**List potential font problems**
Comment	Lists possible font issues.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_potential_font_problems.kfpx
Name	**List potential overprint problems**
Comment	Visualizes overprint issues that might need further investigation. It might be useful to create a layer report from the results for better investigation.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_potential_overprint_problems.kfpx
Name	**List potential printing problems**
Comment	Lists potential printing problems.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_potential_printing_problems.kfpx
Name	**List rich black objects**
Comment	Lists all black objects which are also using Cyan, Magenta or Yellow.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_rich_black_objects.kfpx
Name	**List spot color objects**
Comment	Reports all spot color objects. Note: This profile does not report objects using Separation color spaces Cyan, Magenta, Yellow, Black, All or None.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_spot_color_objects.kfpx
Name	**List text using Courier inside Trim/BleedBox**
Comment	Reports all text objects which use Courier and which - partially or completely - lie inside the TrimBox or BleedBox. This can be used to search for fonts possibly substituted with Courier while ignoring page slugs which often use Courier but typically lie outside the TrimBox and BleedBox.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_text_using_Courier_inside_Trim_or_BleedBox.kfpx

Name	Analyze pages for effectively used plates
Name	**List text using non-embedded fonts**
Comment	Reports all text objects that use a font which is not embedded in the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_text_using_non-embedded_fonts.kfpx
Name	**List transparent objects**
Comment	Lists all page objects that use transparency. Objects are grouped by type of transparency.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_transparent_objects.kfpx
Name	**Will always hit**
Comment	-
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/Will_always_hit.kfpx

Group: PDF Fixups

Name	Convert all pages into CMYK images and preserve text information
Comment	Converts all page contents into CMYK images (150 ppi, high JPEG quality) and creates an invisible text copy for all text in order to preserve features like copying or searching of text.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Convert_all_pages_into_CMYK_images_and_preserve_text_information.kfpx
Name	Convert all pages into RGB images and preserve text information
Comment	Converts all page contents into RGB images (150 ppi, high JPEG quality) and creates an invisible text copy for all text in order to preserve features like copying or searching of text.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Convert_all_pages_into_RGB_images_and_preserve_text_information.kfpx
Name	Convert fonts to outlines

Name	Convert all pages into CMYK images and preserve text information
Comment	Converts fonts to outlines.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Convert_fonts_to_outlines.kfpx
Name	Derive page geometry boxes from crop marks
Comment	Inserts TrimBox and BleedBox on the basis of existing crop marks.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Derive_page_geometry_boxes_from_crop_marks.kfpx
Name	Downsample image resolution to 150 ppi (bitmaps to 300 ppi)
Comment	Downsamples image resolution of color and grayscale images to 150 ppi and image resolution of bitmap images to 300 ppi. Finally images are recompressed with low quality.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Downsample_image_resolution_to_150_ppi_(bitmaps_to_300_ppi).kfpx
Name	Downsample image resolution to 200 ppi (bitmaps to 600 ppi)
Comment	Downsamples image resolution of color and grayscale images to 200 ppi and image resolution of bitmap images to 600 ppi.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Downsample_image_resolution_to_200_ppi_(bitmaps_to_600_ppi).kfpx
Name	Downsample image resolution to 350 ppi (bitmaps to 1200 ppi)
Comment	Downsamples image resolution of color and grayscale images to 350 ppi and image resolution of bitmap images to 1200 ppi.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Downsample_image_resolution_to_350_ppi_(bitmaps_to_1200_ppi).kfpx
Name	Downsample image resolution to specified value
Comment	Downsamples image resolution of color, grayscale and bitmap images to values on the basis of settings that can be defined when running the profile. Finally images are recompressed with high quality.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Downsample_image_resolution_to_specified_value.kfpx

Name	Convert all pages into CMYK images and preserve text information
Name	Embed missing fonts from specified additional font folder only
Comment	If a PDF uses fonts which are not embedded into the PDF file they are embedded when using this fixup. The fonts have to be available in the additional font folder which can be defined when running the fixup.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Embed_missing_fonts_from_specified_additional_font_folder_only.kfpx
Name	Embed missing fonts
Comment	Embeds missing fonts in the PDF if they are available in the system.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Embed_missing_fonts.kfpx
Name	Fix potential font problems
Comment	Fixes possible font issues, where possible, to prevent bad output.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Fix_potential_font_problems.kfpx
Name	Fix problems in PDF tagging structure
Comment	Fixes possible font issues, where possible, to prevent bad output.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Fix_problems_in_PDF_tagging_structure.kfpx
Name	Flatten annotations and form fields
Comment	Flattens annotations and interactive form fields if present. The appearance of all annotations and all form fields will become a part of the page content.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Flatten_annotations_and_form_fields.kfpx
Name	Flatten overprints
Comment	Flattens overprinting objects and underlying objects so that overprints are maintained even if the output device is not capable of handling
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Flatten_overprints.kfpx

Name	Convert all pages into CMYK images and preserve text information
Name	Flatten transparency (high resolution)
Comment	Flattens transparency in the current PDF document using high resolution settings.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Flatten_transparency_(high_resolution).kfpx
Name	Recompress color and grayscale images to JPEG (high quality)
Comment	Recompresses color and grayscale images to JPEG with high quality.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Recompress_color_and_grayscale_images_to_JPEG_(high_quality).kfpx
Name	Recompress color and grayscale images to JPEG2000 (lossless)
Comment	Recompresses color and grayscale images to JPEG2000.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Recompress_color_and_grayscale_images_to_JPEG2000_(lossless).kfpx
Name	Recompress color and grayscale images to ZIP
Comment	Recompresses color and grayscale images to ZIP.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Recompress_color_and_grayscale_images_to_ZIP.kfpx
Name	Scale pages to A4 if not larger than A3
Comment	Scales pages of the PDF proportionally to A4 if their page size is not larger than A3.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Scale_pages_to_A4_if_not_larger_than_A3.kfpx
Name	Scale pages to A4
Comment	Scales all pages of the PDF proportionally to A4
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Scale_pages_to_A4.kfpx

Group: PDF version compatibility

Name	Compatible with PDF 1.2
Comment	Verifies whether the current document is compatible with PDF 1.2 and reports errors if features are used that are not supported in PDF 1.3.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.2.kfpx
Name	Compatible with PDF 1.3
Comment	Makes the current PDF compatible with PDF 1.3 and saves it as a PDF 1.3 document. Among other things flattens layers and transparency and reduces image bit depth from 16 to 8 bit if necessary.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.3.kfpx
Name	Compatible with PDF 1.4
Comment	Makes the current PDF compatible with PDF 1.4 and saves it as a PDF 1.4 document. Among other things flattens layers and reduces image bit depth from 16 to 8 bit if necessary.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.4.kfpx
Name	Compatible with PDF 1.5
Comment	Makes the current PDF compatible with PDF 1.5 and saves it as a PDF 1.5 document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.5.kfpx
Name	Compatible with PDF 1.6
Comment	Makes the current PDF compatible with PDF 1.6 and saves it as a PDF 1.6 document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.6.kfpx

Group: PDFA compliance

Name	Convert to PDF/A-1a
Comment	Converts the current document to PDF/A-1a.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-1a.kfpx
Name	Convert to PDF/A-1b (without fallback conversion)
Comment	Converts the current document to PDF/A-1b without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-1b_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-1b
Comment	Converts the current document to PDF/A-1b.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-1b.kfpx
Name	Convert to PDF/A-2a
Comment	Converts the current document to PDF/A-2a. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-2a.kfpx
Name	Convert to PDF/A-2b (without fallback conversion)
Comment	Converts the current document to PDF/A-2b without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-2b_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-2b
Comment	Converts the current document to PDF/A-2b. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/

Name	Convert to PDF/A-1a
	Convert_to_PDFA-2b.kfpx
Name	Convert to PDF/A-2u
Comment	Converts the current document to PDF/A-2u. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-2u.kfpx
Name	Convert to PDF/A-3a
Comment	Converts the current document to PDF/A-3a. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-3a.kfpx
Name	Convert to PDF/A-3b (without fallback conversion)
Comment	Converts the current document to PDF/A-3b without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-3b_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-3b
Comment	Converts the current document to PDF/A-3b. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-3b.kfpx
Name	Convert to PDF/A-3u
Comment	Converts the current document to PDF/A-3u. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-3u.kfpx
Name	Remove PDF/A information
Comment	PDF/A related entries in the PDF metadata will be removed so that the file will not be identified as a PDF/A file afterwards. However, no other modifications are being

Name	Convert to PDF/A-1a
	made so that the PDF may easily be converted to PDF/A again.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Remove_PDFA_information.kfpx
Name	Verify compliance with PDF/A-1a
Comment	Verifies compliance with PDF/A-1a for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-1a.kfpx
Name	Verify compliance with PDF/A-1b
Comment	Verifies compliance with PDF/A-1b for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-1b.kfpx
Name	Verify compliance with PDF/A-2a
Comment	Verifies compliance with PDF/A-2a for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-2a.kfpx
Name	Verify compliance with PDF/A-2b
Comment	Verifies compliance with PDF/A-2b for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-2b.kfpx
Name	Verify compliance with PDF/A-2u
Comment	Verifies compliance with PDF/A-2u for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-2u.kfpx
Name	Verify compliance with PDF/A-3a
Comment	Verifies compliance with PDF/A-3a for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-3a.kfpx

Name	Convert to PDF/A-1a
Name	Verify compliance with PDF/A-3b
Comment	Verifies compliance with PDF/A-3b for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-3b.kfpx
Name	Verify compliance with PDF/A-3u
Comment	Verifies compliance with PDF/A-3u for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-3u.kfpx
Name	Convert to PDF/A-4 (without fallback conversion)
Comment	Converts the current document to PDF/A-4 without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-4_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-4e (without fallback conversion)
Comment	Converts the current document to PDF/A-4e without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-4e_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-4f (without fallback conversion)
Comment	Converts the current document to PDF/A-4f without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-4f_(without_fallback_conversion).kfpx
Name	PDF/A-2u: Outline non-Unicode text if it is below specified share of all text
Comment	Converts all text without Unicode representation into outlines if the share of such text is below a specified share (in percentage) of all text. Some PDF creators create PDF where certain characters (such as bullet points) do not have Unicode representation. This Process Plan can be used to create files that formally have full Unicode representation.

Name	Convert to PDF/A-1a
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/PDFA-2u_-_Outline_non-Unicode_text.kfpx
Name	Verify compliance with PDF/A-4
Comment	Verifies compliance with PDF/A-4 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-4.kfpx
Name	Verify compliance with PDF/A-4e
Comment	Verifies compliance with PDF/A-4e for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-4e.kfpx
Name	Verify compliance with PDF/A-4f
Comment	Verifies compliance with PDF/A-4f for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-4f.kfpx
Name	Verify compliance with ZUGFeRD
Comment	ZUGFeRD is a German standard for electronic invoices. A ZUGFeRD invoice is a PDF/A-3 file with an embedded XML structure that allows for automatic processing of the invoice.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_ZUGFeRD.kfpx

Group: PDFE compliance

Name	Convert to PDF/E-1
Comment	Converts the current document to PDF/E-1.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFE_compliance/Convert_to_PDFE-1.kfpx
Name	Remove PDF/E information

Name	Convert to PDF/E-1
Comment	PDF/E related entries in the PDF metadata will be removed so that the file will not be identified as a PDF/E file afterwards. However, no other modifications are being made so that the PDF may easily be converted to PDF/E again.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFE_compliance/Remove_PDFE_information.kfpx
Name	Verify compliance with PDF/E-1
Comment	Verifies compliance with PDF/E-1 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFE_compliance/Verify_compliance_with_PDFE-1.kfpx

Group: PDFUA compliance

Name	Fix problems in PDF tagging structure
Comment	Fixes potential problems in the structure of tagged PDF documents.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFUA_compliance/Fix_problems_in_PDF_tagging_structure.kfpx
Name	Remove PDF/UA information
Comment	PDF/UA related entries in the PDF metadata will be removed so that the file will not be identified as a PDF/UA file afterwards. However, no other modifications are being made so that the PDF may easily be converted to PDF/UA again.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFUA_compliance/Remove_PDFUA_information.kfpx
Name	Verify compliance with PDF/UA-1 (syntax checks only)
Comment	Verifies compliance with PDF/UA-1 for the current document. PDF/UA validation does always require machine checks as well as semantic checks. This profile will perform all machine checks and allows you to open a window for interactive inspection.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFUA_compliance/Verify_compliance_with_PDFUA-1_(syntax_checks_only).kfpx

Group: PDFVT compliance

Name	Verify compliance with PDF/VT-1
Comment	Verifies compliance with PDF/VT-1 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFVT_compliance/Verify_compliance_with_PDFVT-1.kfpx
Name	Verify compliance with PDF/VT-2
Comment	Verifies compliance with PDF/VT-2 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFVT_compliance/Verify_compliance_with_PDFVT-2.kfpx
Name	Verify compliance with PDF/VT-3
Comment	Verifies compliance with PDF/VT-3 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFVT_compliance/Verify_compliance_with_PDFVT-3.kfpx

Group: PDFX compliance

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
Comment	Converts the current document to PDF/X-1a for Coated GRACoL 2006 as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-1a_(Coated_GRACoL_2006).kfpx
Name	Convert to PDF/X-1a (ISO Coated v2 (ECI))
Comment	Converts the current document to PDF/X-1a for ISO Coated v2 (ECI) as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
	Convert_to_PDFX-1a_(ISO_Coated_v2_(ECI)).kfpx
Name	Convert to PDF/X-1a (Japan Color 2001 Coated)
Comment	Converts the current document to PDF/X-1a for ISO Coated v2 (ECI) as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-1a_(Japan_Color_2001_Coated).kfpx
Name	**Convert to PDF/X-1a (PSO Coated v3 (ECI))**
Comment	Converts the current document to PDF/X-1a for PSO Coated v3 (ECI) as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-1a_(PSO_Coated_v3_(ECI)).kfpx
Name	Convert to PDF/X-1a (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-1a for US Web Coated (SWOP) v2 as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-1a_(US_Web_Coated_(SWOP)_v2).kfpx
Name	**Convert to PDF/X-3 (Coated GRACoL 2006)**
Comment	Converts the current document to PDF/X-3 for Coated GRACoL 2006 as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(Coated_GRACoL_2006).kfpx
Name	**Convert to PDF/X-3 (ISO Coated v2 (ECI))**
Comment	Converts the current document to PDF/X-3 for ISO Coated v2 (ECI) as the in-

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
	tended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to PDF/X-3 (Japan Color 2001 Coated)**
Comment	Converts the current document to PDF/X-3 for Japan Color 2001 Coated as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(Japan_Color_2001_Coated).kfpx
Name	**Convert to PDF/X-3 (PSO Coated v3 (ECI)**
Comment	Converts the current document to PDF/X-3 for PSO Coated v3 (ECI) as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(PSO_Coated_v3_(ECI).kfpx
Name	Convert to PDF/X-3 (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-3 for US Web Coated (SWOP) v2 as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(US_Web_Coated_(SWOP)_v2).kfpx
Name	**Convert to PDF/X-4 (Coated GRACoL 2006)**
Comment	Converts the current document to PDF/X-4 for Coated GRACoL 2006 as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(Coated_GRACoL_2006).kfpx

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
Name	**Convert to PDF/X-4 (ISO Coated v2 (ECI))**
Comment	Converts the current document to PDF/X-4 for ISO Coated v2 (ECI) as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to PDF/X-4 (Japan Color 2001 Coated)**
Comment	Converts the current document to PDF/X-4 for Japan Color 2001 Coated as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(Japan_Color_2001_Coated).kfpx
Name	**Convert to PDF/X-4 (PSO Coated v3 (ECI))**
Comment	Converts the current document to PDF/X-4 for PSO Coated v3 (ECI) as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(PSO_Coated_v3_(ECI)).kfpx
Name	**Convert to PDF/X-4 and create language layer views (ISO Coated v2 (ECI))**
Comment	Converts the current document to PDF/X-4 for ISO Coated v2 (ECI) as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed. The following layer views (OCCDs) are created: Layer name starts with EN or ends with EN; - English layer view; Layer name starts with DE or _DE - German layer view; Layer name starts with FR or ends _FR - French layer view; Layer name starts with IT or ends with _IT - Italian layer view & Layer name starts with ES or ends with _ES - Spanish layer view;
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_with_layer_views_(ISO_Coated_v2_(ECI))).kfpx
Name	Convert to PDF/X-4 (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-4 for US Web Coated (SWOP) v2 as the intended printing condition according to the PDF/X-4 standard. Transparency and layers

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
	are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(US_Web_Coated_(SWOP)_v2).kfpx
Name	**Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))**
Comment	Converts the current document to PDF/X-5n for 7C Indigo TAC370 (ColorLogic) as the intended printing condition according to the PDF/X-5n standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-5n_(7C_Indigo_TAC370_(ColorLogic)).kfpx
Name	**Remove PDF/X information**
Comment	PDF/X related entries in the PDF metadata will be removed so that the file will not be identified as a PDF/X file afterwards. However, no other modifications are being made so that the PDF may easily be converted to PDF/X again.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Remove_PDFX_information.kfpx
Name	**Verify compliance with PDF/X-1a**
Comment	Verifies compliance with PDF/X-1a:2003 for the current document. The PDF/X-1a:2003 standard also implies valid PDF/X-1a:2001 files as PDF/X-1a:2003 compliant.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-1a.kfpx
Name	**Verify compliance with PDF/X-3**
Comment	Verifies compliance with PDF/X-3:2003 for the current document. The PDF/X-3:2003 standard also implies valid PDF/X-1a:2001, PDF/X-1a:2003 or PDF/X-3:2002 files as PDF/X-3:2003 compliant.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-3.kfpx
Name	**Verify compliance with PDF/X-4**
Comment	Verifies compliance with PDF/X-4 for the current document.

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-4.kfpx
Name	**Verify compliance with PDF/X-4p**
Comment	Verifies compliance with PDF/X-4p for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-4p.kfpx
Name	**Verify compliance with PDF/X-5g**
Comment	Verifies compliance with PDF/X-5g for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-5g.kfpx
Name	**Verify compliance with PDF/X-5n**
Comment	Verifies compliance with PDF/X-5n for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-5n.kfpx
Name	**Verify compliance with PDF/X-5pg**
Comment	Verifies compliance with PDF/X-5pg for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-5pg.kfpx
Name	**Convert to PDF/X-6 (Coated GRACoL 2006)**
Comment	Converts the current document to PDF/X-6 for Coated GRACoL 2006 as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(Coated_GRACoL_2006).kfpx
Name	**Convert to PDF/X-6 (Coated CRACoL 2013)**
Comment	Converts the current document to PDF/X-6 for Coated GRACoL 2013 as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(Coated_GRACoL_2013).kfpx
Name	Convert to PDF/X-6 (ISO Coated v2 (ECI))
Comment	Converts the current document to PDF/X-6 for ISO Coated v2 (ECI) as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to PDF/X-6 (Japan Color 2001 Coated)**
Comment	Converts the current document to PDF/X-6 for Japan Color 2001 Coated as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(Japan_Color_2001_Coated).kfpx
Name	**Convert to PDF/X-6 (PSO Coated v3 (ECI))**
Comment	Converts the current document to PDF/X-6 for PSO Coated v3 (ECI) as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(PSO_Coated_v3_(ECI)).kfpx
Name	**Convert to PDF/X-6 (US Web Coated (SWOP) v2)**
Comment	Converts the current document to PDF/X-6 for US Web Coated (SWOP) v2 as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(US_Web_Coated_(SWOP)_v2).kfpx
Name	**Verify compliance with PDF/X-6**
Comment	Verifies compliance with PDF/X-6 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-6.kfpx

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
Name	**Verify compliance with PDF/X-6n**
Comment	Verifies compliance with PDF/X-6n for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-6n.kfpx
Name	**Verify compliance with PDF/X-6p**
Comment	Verifies compliance with PDF/X-6p for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-6p.kfpx

Group: PDFX and PDFA

Name	Convert to PDF/X-4 and PDF/A-2 (Coated GRACoL 2006)
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for Coated GRACoL 2006 as the intend dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(Coated_GRACoL_2006).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (Coated GRACoL 2013)
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for Coated GRACoL 2013 as the intend dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(Coated_GRACoL_2013).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (ISO Coated v2 (ECI))
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for ISO Coated v2 (ECI) as the intended dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(ISO_Coated_v2_(ECI)).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (Japan Color 2001 Coated)
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for Japan Color 2001 Coated as the int

Name	Convert to PDF/X-4 and PDF/A-2 (Coated GRACoL 2006)
	condition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(Japan_Color_2001_Coated).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (PSO Coated v3 (ECI))
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for PSO Coated v3 (ECI) as the intende dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(PSO_Coated_v3_(ECI)).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for US Web Coated (SWOP) v2 as the ir ing condition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(US_Web_Coated_(SWOP)_v2).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (Coated GRACoL 2006)
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for Coated GRACoL 2006 as the intend dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(Coated_GRACoL_2006).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (Coated GRACoL 2013)
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for Coated GRACoL 2013 as the intend dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(Coated_GRACoL_2013).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (ISO Coated v2 (ECI))
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for ISO Coated v2 (ECI) as the intended dition.

Name	Convert to PDF/X-4 and PDF/A-2 (Coated GRACoL 2006)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(ISO_Coated_v2_(ECI)).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (Japan Color 2001 Coated)
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for Japan Color 2001 Coated as the int condition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(Japan_Color_2001_Coated).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (PSO Coated v3 (ECI))
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for PSO Coated v3 (ECI) as the intende dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(PSO_Coated_v3_(ECI)).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for US Web Coated (SWOP) v2 as the ir ing condition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(US_Web_Coated_(SWOP)_v2).kfpx

Group: Packaging

Name	Packaging Flexo (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of the Ghent PDF Workgroup as defined in the 2015 specification'GWG_PackagingFlexo_2015'.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Packaging/Packaging_Flexo_(GWG_2015).kfpx
Name	Packaging Gravure (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of the

Name	Packaging Flexo (GWG 2015)
	Ghent PDF Workgroup as defined in the 2015 specification 'GWG_Packaging-Gravure_2015'.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Packaging/Packaging_Gravure_(GWG_2015).kfpx
Name	Packaging Offset (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of the Ghent PDF Workgroup as defined in the 2015 specification 'GWG_PackagingOffset_2015'.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Packaging/Packaging_Offset_(GWG_2015).kfpx

Group: Place content

Name	Place EAN 13 barcode with specified value
Comment	Places an EAN 13 code at the center of the page. Content is based on the value specified during runtime. The value has to consist of 12 or 13 digits; EAN 13 always uses 13 digits, the last being a checksum which will be added automatically if only 12 digits are specified. The barcode will be placed on a layer.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_EAN_13_barcode_with_specified_value.kfpx
Name	Place QR-Code with specified value
Comment	Places QR-Code at the center of the page. Content is based on the value specified during runtime. The QR-Code will be placed on a layer.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_QR-Code_with_specified_value.kfpx
Name	Place date (YYYY-MM-DD)
Comment	Places the current date into the upper right corner of each page. The default format is YYYY-MM-DD, however, formatting may be adapted by modifing the JavaScript Variable, which already contains code for other formats.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_date_(YYYY-MM-DD).kfpx

Name	Place EAN 13 barcode with specified value
Name	Place document name in gutter
Comment	Adds the name of the current document (minus the extension part) to the gutter of the document. Placement is 3 mm below the center of the trim box.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_document_name_in_gutter.kfpx
Name	Place job ID in gutter
Comment	Adds the job ID to the gutter of the document both in plain text format and as a Code 128 barcode. The default variable of the job ID is defined in the following variable: @var_jobid:'Job ID' 1234567890@.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_job_ID_in_gutter.kfpx
Name	Place page number in left or right corner
Comment	Places the page number at the lower left corner on even pages and at the lower right on odd pages. Each pager number will be followed by a delimiter symbol ' / ' and the total number of pages in the document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_page_number_in_left_or_right_corner.kfpx
Name	Place specified text at the center of each page
Comment	Places a text at the center of the page. The positioned text is based on the value specified during runtime.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_specified_text_at_the_center_of_each_page.kfpx
Name	Quick Check example
Comment	Puts file name and file path on the top left of every page in the current PDF (using pink spot colored Courier text). File name and file path are collected using a Quick Check process plan step. Using a "Place text" fixup in a second process plan step, file name and file path are picked up and combined into a suitable text using a JavaScript variable that retrieves the two values from the app.vars JavaScript application object's data structure.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Quick_Check_example.kfpx

Group: Preflight Certificate

Name	Verify Preflight Certificate
Comment	Verifies the existence and validity of a Preflight Certificate in the PDF document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Preflight_Certificate/Verify_Preflight_Certificate.kfpx

Group: Prepress

Name	Bring 100% black text to front (optionally check for visual changes)
Comment	Brings text using 100% black to front; at runtime it is possible to choose whether to do a befor in order to detect pages where bringing the text to the front causes a different appearance of t
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Bring_100%25_black_text_to_front_(optionally_check_for_visual_changes).kfpx
Name	Check and fix bleed
Comment	Analyses bleed and safety zones to determine required bleed. Safety zone starts at 2 mm insid 3;Tolerance: 25 Unit: mm Create bleed: Off; Page type: single; Developed by calibrate (office@c
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Check_and_fix_bleed.kfpx
Name	Magazine Ads (CMYK and RGB) (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of the Ghent P in the 2015 specification "GWG_MagazineAds_2015 CMYK + RGB".
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Magazine_Ads_(CMYK_and_RGB)_(GWG_2015).kfpx
Name	Magazine Ads (CMYK) (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification "GWG_MagazineAds_2015 CMYK".
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Magazine_Ads_(CMYK)_(GWG_2015).kfpx

Name	Bring 100% black text to front (optionally check for visual changes)
Name	**Newspaper Ads (CMYK and RGB) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_NewspaperAds_2015 CMYK + RGB”
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Newspaper_Ads_(CMYK_and_RGB)_(GWG_2015).kfpx
Name	**Newspaper Ads (CMYK) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_NewspaperAds_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Newspaper_Ads_(CMYK)_(GWG_2015).kfpx
Name	**Sheetfed offset (CMYK and RGB) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_SheetCmyk_2015 CMYK + RGB”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Sheetfed_offset_(CMYK_and_RGB)_(GWG_2015).kfpx
Name	**Sheetfed offset (CMYK and spot colors) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_SheetSpot_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Sheetfed_offset_(CMYK_and_spot_colors)_(GWG_2015).kfpx
Name	**Sheetfed offset (CMYK) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_SheetCmyk_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Sheetfed_offset_(CMYK)_(GWG_2015).kfpx
Name	**Sheetfed offset (CMYK, RGB and spot colors) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_SheetSpot_2015 CMYK + RGB”.

Name	Bring 100% black text to front (optionally check for visual changes)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Sheetfed_offset_(CMYK%2C_RGB_and_spot_colors)_(GWG_2015).kfpx
Name	**Web offset (CMYK and RGB) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebCmyk_2015 CMYK + RGB”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK_and_RGB)_(GWG_2015).kfpx
Name	**Web offset (CMYK and RGB, newsprint) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebCmykNews_2015 CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK_and_RGB%2C_newsprint)_(GWG_2015).kfpx
Name	**Web offset (CMYK and spot colors) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebSpot_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK_and_spot_colors)_(GWG_2015).kfpx
Name	**Web offset (CMYK) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebCmyk_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK)_(GWG_2015).kfpx
Name	**Web offset (CMYK, RGB and spot colors) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebSpot_2015 CMYK + RGB”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK%2C_RGB_and_spot_colors)_(GWG_2015).kfpx
Name	**Web offset (CMYK, newsprint) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th

Name	Bring 100% black text to front (optionally check for visual changes)
	group as defined in the 2015 specification “GWG_WebCmykNews_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK%2C_newsprint)_(GWG_2015).kfpx
Name	White underlay for print content on transparent foil (with bleed)
Comment	Creates a spot color plate “White underlay” where the tint value varies depending on the ink a transparent foil, print content – plus a small amount of ‘bleed’ around it – will be printed on th areas without print content remain transparent.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/White_underlay_for_parent_foil_(with_bleed).kfpx

Group: Process plans

Name	Check for PDF/A-2b compliance and convert if not compliant
Comment	Checks whether the PDF is compliant with the PDF/A-2b standard. The PDF is only modified ar report is created if conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Check_for_PDFA-2b_compliance_and_convert_if_not_compliant.kfpx
Name	Check for non-CMYK color, create a report, convert to CMYK (ISO Coated v2) and flatten transp
Comment	Checks whether the PDF uses spot colors or RGB, if this is the case a report with masks is creat jects can be indentified.
All non-CMYK colors will be converted into CMYK (ISO Coated v2) afterwards.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Check_for_non-CMYK_color%2C_create_a_report%2C_convert_to_CMYK_and_flatten_transparency.kfpx
Name	Convert pages with huge numbers of objects into CMYK images
Comment	Pages are analyzed whether limits for total number of page objects are exceeded in which cas age to reduce processing time when output.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Convert_pages_with_huge_numbers_of_objects_into_CMYK_images.kfpx
Name	Create DPart record information from headings

Name	Check for PDF/A-2b compliance and convert if not compliant
Comment	Identifies text in a specified size and adds a record start at pages with matching text in the DPa take advantage of such structures for caching and can significantly reduce processing time.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Create_DPart_record_information_from_headings.kfpx
Name	Convert to CMYK and spot colors and generate images of pre-separated pages
Comment	Converts the document to CMYK + spot colors and generates images of pre-separated pages (J
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Convert_to_CMYK_and_spot_colors_and_generate_images_of_pre-separated_pages.kfpx
Name	Convert to PDF/A-2b (any fallback conversions use the original file)
Comment	When regular PDF/A-2b conversion fails the original is attempted to convert by regenerating it by rasterizing all pages.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Convert_to_PDFA-2b_(any_fallback_conversions_use_the_original_file).kfpx
Name	Generate 3 mm bleed (pixel repetition on pages with text objects close to Trimbox, else mirror
Comment	Determines whether a page has text objects close to Trimbox in which case for that page bleed otherwise mirroring is used. The process plan uses a check that creates an array with page nu using pixel repetition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Generate_3_mm_bleed_(pixel_repetition_or_mirroring_depending_on_content).kfpx
Name	Prepare PDF for specified page size
Comment	Checks whether the TrimBox size of the pages matches the defined size and uses several appr not the case. This Process Plan will only work for PDF files where all pages have the same size.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Prepare_PDF_for_specified_page_size.kfpx
Name	Split PDF into half and impose as a brochure
Comment	Splits all pages in the middle (respecting single and double pages) and imposes the result for
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Split_PDF_into_half_and_impose_as_a_brochure.kfpx

Group: Processing Steps

Name	List Processing Steps metadata information
Comment	List Processing Steps metadata information
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Processing_steps/List_Processing_Steps_metadata_information.kfpx
Name	Verify compliance with Processing Steps (ISO 19593-1) for packaging and label
Comment	Verifies compliance with ISO 19593-1, Processing Steps for print files for packagings and labels
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Processing_steps/Verify_compliance_with_Processing_Steps_(ISO_19593-1)_for_packaging_and_label.kfpx

32.26 Modifying structure of bookmarks and DPart

pdfToolbox can extract existing DPart Metadata or bookmark structures in a JSON format so that you can modify them and inject them back into the PDF document. This functionality is available as part of Process Plans where you extract the structures using Quick Check and import them via the “Apply structures” action and directly on the command-line with pdfToolbox CLI using the following commands:

```
--modifystructures --extract
--modifystructures --apply
```

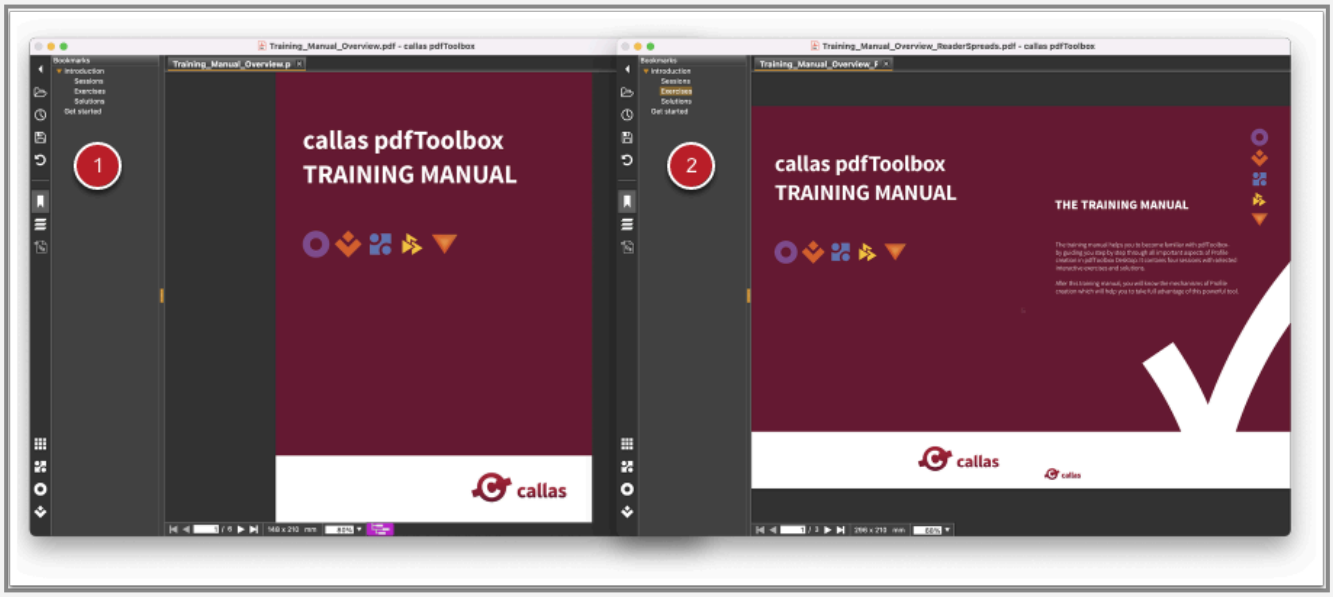
Click to copy

This article provides two examples of how to update the structures:

- Example 1: Modifying bookmark structure
- Example 2: Modifying DPart Metadata structure

Modifying bookmark structure

If you modify a document, such as converting a single-page booklet to reader spreads, adding new pages, or deleting existing pages, most of the time the bookmark structure is not updated and can no longer be used. Here is an example:



1. Document with single pages and existing bookmark structure
2. After conversion to reader spreads, the existing bookmarks won't work anymore because the bookmark structure is not updated accordingly.

To update the bookmark structure, follow the next steps. The modified testfile with the "old" bookmark structure can be downloaded here:

Training_Manual_Overview_ReaderSpreads.pdf

1. Extract current bookmark structure to JSON

Use the following command on the CLI to extract the existing bookmark structure to JSON:

```
pdfToolbox <sample.pdf> --modifystructures --extract=bookmarks
```

Click to copy

To modify the extracted bookmark structure, it is important to understand the different properties of the JSON objects:

`"bookmarks"`: Array that contains a list of all bookmarks found in the PDF file.

`"level"`: Indicates the nesting level of each bookmark as typically displayed in a PDF viewing program.

`"name"`: Indicates the name of the bookmark.

`"page"`: Specifies the page number that defines the destination of the bookmark.

`"open"`: True if the bookmark is initially open. This only has an effect for a bookmark hierarchy.

```
{
    "bookmarks" :
    [
        {
            "level" : 1,
            "name" : "Introduction",
            "open" : true,
            "page" : 0
        },
        {
            "level" : 2,
            "name" : "Sessions",
            "page" : 0
        },
        {
            "level" : 2,
            "name" : "Exercises",
            "page" : 0
        },
        {
            "level" : 2,
            "name" : "Solutions",
            "page" : 0
        },
        {
            "level" : 1,
            "name" : "Get started",
            "page" : 0
        }
    ]
}
```

2. Adjust bookmark structure

The bookmark structure should now be adjusted using algorithms for the specific use case. In our example, the page numbers have to be divided by 2 to so that the bookmarks refer to the correct pages in the reader spreads document. Also, the name of the first bookmark was changed from "Introduction" to "Manual content".



```
{
    "bookmarks" :
    [
        {
            "level" : 1,
            "name" : "Manual Content",
            "open" : true,
            "page" : 1
        },
        {
            "level" : 2,
            "name" : "Sessions",
            "page" : 2
        },
        {
            "level" : 2,
            "name" : "Exercises",
            "page" : 2
        },
        {
            "level" : 2,
            "name" : "Solutions",
            "page" : 3
        },
        {
            "level" : 1,
            "name" : "Get started",
            "page" : 3
        }
    ]
}
```

3. Apply new bookmark structure to PDF

Use the following command on the CLI to inject the new bookmark structure into the PDF:

```
pdfToolbox --modifystructures --apply=<JSON file> <PDF file>
```

Click to copy

Modifying DPart structure

DPart is page based metadata in a PDF file, stored in a hierarchy to group pages into “document parts” (which can group pages to "records"). It is organized in a tree so that it is easier for any PDF processors to identify pages or page ranges that have certain properties. pdfToolbox can display such metadata (as described in Display DPart Metadata).

When the page structure in a document is modified by any application, such as adding new pages or deleting existing pages, quite often the DPart structure is not updated.

In the example below (taken from the PDF/VT sample files at https://pdfa.org/resource/cal-poly-pdfvt-test-suite/) there is a PDF file that has 10 "document parts" / "records". Each "record" consists of 4 pages: the first two pages are a brochure and the last two pages are luggage tags.

Let us assume that we need to delete the luggage tags from each record which means that the DPart metadata will not fit anymore.



To update the bookmark structure, follow the next steps. The modified testfile with the "old" DPart metadata structure can be downloaded here:

Travel V1.0.1 - 10 small - Brochure.pdf

1. Extract current DPart structure to JSON

Use the following command on the CLI to extract the existing DPart structure to JSON:

```
pdfToolbox <sample.pdf> --modifystructures --extract=dpart
```

Click to copy

To modify the extracted DPart structure, it is important to understand the different entries:

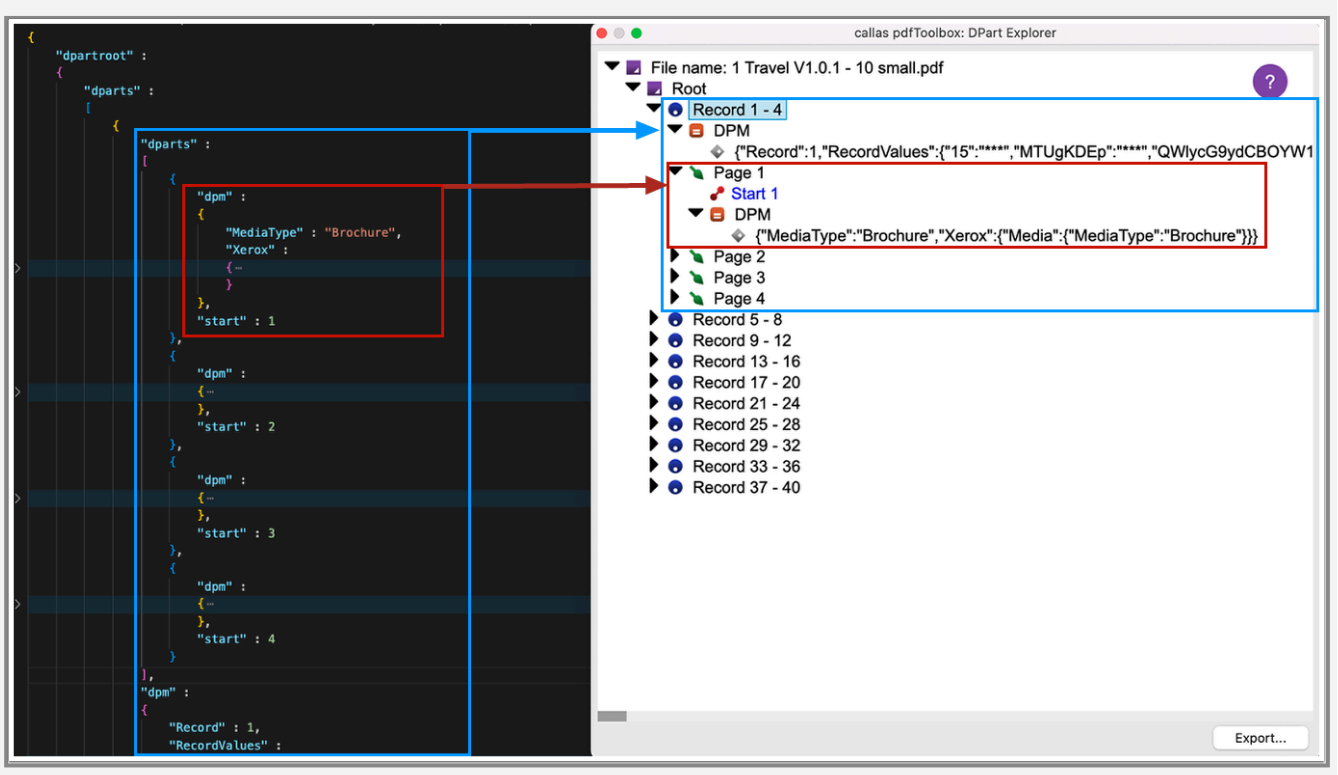
`"dpartroot"` : The root node of the hierarchy.

`"dparts"` : Can refer to a specific range of one or more PDF page objects, identified by their start and end keys.

`"dpm"` : The Document Part Metadata Dictionary contains the actual metadata related to the different parts of the document. The DPM refers to the DPart node in which it is defined and its pages.

`"start"` : The start key is the number of the first PDF page to which the DPart node refers.

`"end"` (optional): If a DPart node refers to not just one page but to a range of pages there is an end property (in addition to the start property) indicating the last page to which the node refers.



At the end there are two more entries:

`"nodenamelist"` (optional): Specifies the names/meaning of the DPart node levels in the tree hierarchy. These names can provide some explanation and could be used by software to display information. In our example we have three levels: The root, the Record level and the pages.

`"recordlevel"` (optional): Indicates on which level of the DPart hierarchy the record boundaries are, in our example RecordLevel is 1. Since the root is zero this indicates that the first level of the DPart hierarchy refers to the records. In our example that are ranges of always four pages specified via their start and page entries.

```
    "nodenamelist" :
    [
        "Root",
        "Record",
        "Page"
    ],
    "recordlevel" : 1
```

2. Adjust DPart structure

The DPart metadata for the luggage tags has been manually removed from the records. Now the records consist only of the two brochure entries. The page numbers (`"start"` keys) have also been adjusted:

Travel V1.0.1 - 10 small - Brochure - new structure.json



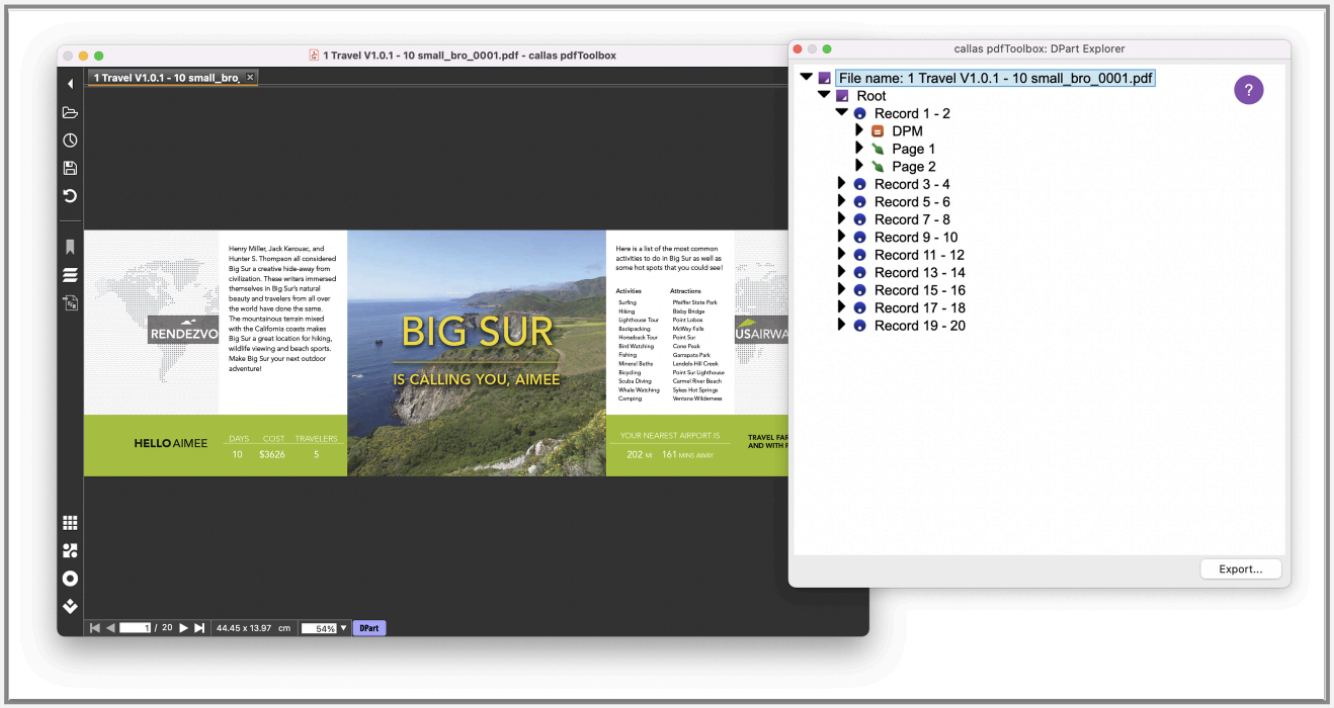
3. Apply new DPart structure to PDF

Use the following command on the CLI to inject the DPart structure into the PDF:

```
pdfToolbox --modifystructures --apply=<JSON file> <PDF file>
```

Click to copy

Now each record has only two pages, and the DPart metadata for all luggage tags in the DPart structure is deleted:


1 Travel V1.0.1 - 10 small_bro_0001.pdf - callas pdfToolbox
1 Travel V1.0.1 - 10 small_bro_
BIG SUR
IS CALLING YOU, AIMEE
HELLO AIMEE
DAYS COST TRAVELERS
10 $3626 5
callas pdfToolbox: DPart Explorer
File name: 1 Travel V1.0.1 - 10 small_bro_0001.pdf
Root
Record 1 - 2
DPM
Page 1
Page 2
Record 3 - 4
Record 5 - 6
Record 7 - 8
Record 9 - 10
Record 11 - 12
Record 13 - 14
Record 15 - 16
Record 17 - 18
Record 19 - 20
Export...
DPart

32.27 Handling Licensing through the License Server

In order to avoid having to activate a callas software product, you can use a reference to a running License Server in the call to that product. The `--licenseserver` option points the callas software product to the License Server so it can get permission to run your call. There are two License Server-based models:

Using the License Server

All commands related to the License Server are explained in the License Server manual: Using the License Server.

Using the License Server on-premise

All commands related to the License Server on-premise are explained in the License Server manual: Using the License Server on-premise.

32.28 Creating ZUGFeRD / Factur-X invoices

The German e-invoicing standard ZUGFeRD (Central User Guide Forum electronic Invoice Germany) defines a document, which contains an invoice in a human readable (PDF/A-3) and a machine readable format (XML). By using the PDF/A-3 standard the created file satisfies long-term archiving regulations.

It is possible to embed such a ZUGFeRD-XML file (created by an electronic billing or accounting system) into a PDF, setting the required ZUGFeRD-XMP metadata and making the PDF a valid PDF/A-3 in one run.

Factur-X is the corresponding French standard to ZUGFeRD 2.0 for electronic invoicing in EU countries.

Create a ZUGFeRD invoice using an invoice PDF and an invoice XML:

```
--zugferd --create=<ZUGFeRD XML file> <PDF file>
```

Adding a GiroCode during creation with a ZUGFeRD invoice using an invoice PDF and an invoice XML (GiroCode data will be determined from XML file):

```
--zugferd --create=<ZUGFeRD XML file>
--girocode=<lower left x>,<lower left y>,<size>,<unit>,<pagenumber>
<PDF file>
```

Parameters for GiroCode:

lower left x	Position from the lower left on x axis
lower left y	Position from the lower left on y axis

size	Size of resulting GiroCode (always quadratic)
unit	Optional unit for coordiantes and size: mm, inch, pt (default: mm)
pagenumber	Page, where the GiroCode will be positioned

Validate a PDF and the containing XML against the ZUGFeRD specification:

```
--zugferd <PDF file>
```

Extract the embedded XML from a ZUGFeRD PDF file:

```
--zugferd --extract <PDF file>
```

Create several reports from a ZUGFeRD invoice:

```
--zugferd --export=[<option>,<option>,...]
  [--resolution=<resolution in ppi>]
  [--imgformat=<image format>]
  <PDF file>
```

Options for ZUGFeRD reports:

NOPDFIMAGE	No images from the PDF pages

NOXML	XML invoice is not extracted
NOHTML	No HTML view of the XML
NOHTMLSYNOPSIS	No synopsis of PDF images and XML
NOVALIDATIONREPORT	No ZUGFeRD validation report

Addtional parameters for the export of PDFIMAGES:

RESOLUTION IN PPI	optional, resolution in ppi or width x height in pixel, e.g. 1024x800 (default: 72)
IMGFORMAT	optional, JPEG, PNG, TIFF, PDF (default: JPEG)

33. Using pdfToolbox in cloud environments

33.1 Using the callas pdfToolbox CLI Docker Image from Docker Hub

callas pdfToolbox CLI is now available as a pre-built Docker image on Docker Hub, simplifying deployment and ensuring consistency across environments. This guide explains how to use the image and configure licensing.

Prerequisites

- Docker installed on your system (Installation Guide)
- A valid callas license (via a License Server or an OEM agreement)

Quick Start

Pull the callas pdfToolbox CLI Docker image ...

```
docker pull callassoftware/pdftoolbox-cli
```

Click to copy
Run a basic command ...

```
docker run --rm callassoftware/pdftoolbox-cli pdftoolbox-cli --version
```

Click to copy

Licensing Setup

pdfToolbox-cli requires a License Server to manage activations. Docker containers are ephemeral, so license activation data cannot persist in the image itself.

Option 1:

- Use the callas License Server in the Cloud. Further information can be found here: Using the License Server
  - Specify the --licenseserver=licenseserver.callassoftware.com --lsmessage=<your wallet ID> options, e.g. ...

```
pdfToolbox --licenseserver=licenseserver.callassoftware.com --lsmessage=6d203314-e87e-11ef-9b89-db96358f5fef sample.kfpx sample.pdf -o=o.pdf
```

Click to copy

Option 2:

- Self-Hosted License Server
  - specify the --licenseserver=<your licenseserver IP> option, e.g. ...

```
pdfToolbox --licenseserver=192.168.188.42 sample.kfpx sample.pdf -o=o.pdf
```

Click to copy

Option 3:

- For embedded or high-volume use cases, contact callas for OEM agreements: support@callassoftware.com

Best Practices

Avoid the latest tag in production. Instead use versioned tags like callassoftware/pdfToolbox-cli:v16-1-661 for stability.

Troubleshooting

"License not found" errors: Verify the License Server is reachable and the specified license server is set correctly. Also verify that you have a valid cartridge installed

Docker-snippets repository

The docker-snippets repository on GitHub provides examples and instructions for creating custom Docker images. Find more information here.

33.2 Docker Snippets for callas pdfToolbox SDK

There are "ready-to-use" Docker images for pdfToolbox, pdfaPilot, and pdfChip on Docker Hub.

However, there is no equivalent for the pdfToolbox SDK/ pdfaPilot SDK (pdfEngine SDK) on Docker Hub. This is because it is not a standalone application but rather a software development kit intended for building custom applications.

Docker-snippets repository

The docker-snippets repository on GitHub contains an example Dockerfile-debian that demonstrates how to containerize a custom application using the callas pdfEngine SDK.

You can find further details and usage instructions here: https://github.com/callassoftware/docker-snippets/blob/main/other/pdfEngine-sdk/README.md

33.3 Using the License Server

Info

This article refers to the use of the **License Server as a SaaS solution**. If you are interested in the License Server on-premise, please refer to the following chapters:

- License Server on-premise: Installation, activation, deactivation
- License Server on-premise: Using the License Server

There is no need to activate the License Server or to install Cartridges in it. Instead, each user has a Wallet in which the associated resources (Cartridges) are saved. The Wallet is referenced via a Wallet ID, that the user will receive when the first Cartridge is purchased. (Cartridges can be purchased from those partners where also regular pdfToolbox Licenses are available.)

The URL for License Server is

licenseserver.callassoftware.com

Using the License Server

It is straightforward to use License Server on command line or in the SDK.

The parameter to reference a License Server needs to specify the URL of License Server and an additional parameter "`lsmessage`" needs to specify the Wallet ID.

Example

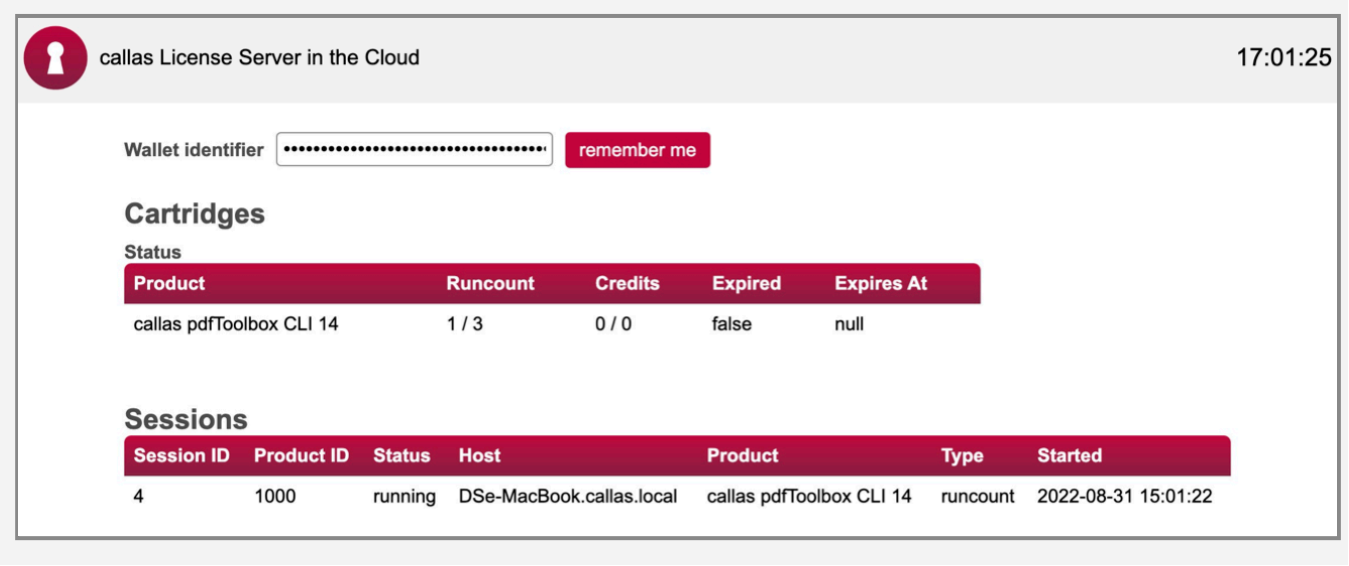
```
./pdfToolbox --licenseserver=licenseserver.callassoftware.com --lsmessage=<Wallet ID> sample.kfpx sample.pdf -o=output.pdf
```

Monitoring the License Server

In order to monitor what Cartridges are available for a Wallet ID and what Cartridges are currently in use the License Server webinterface can be used:

https://licenseserver.callassoftware.com

The Wallet identifier is then to be entered into the input field to display the information for that Wallet.



For **automated monitoring** purposes it is also possible to receive the monitoring data in a .json format.

The following example uses the *curl* command. This will return a list of all cartridges and a list of all current sessions.

Example

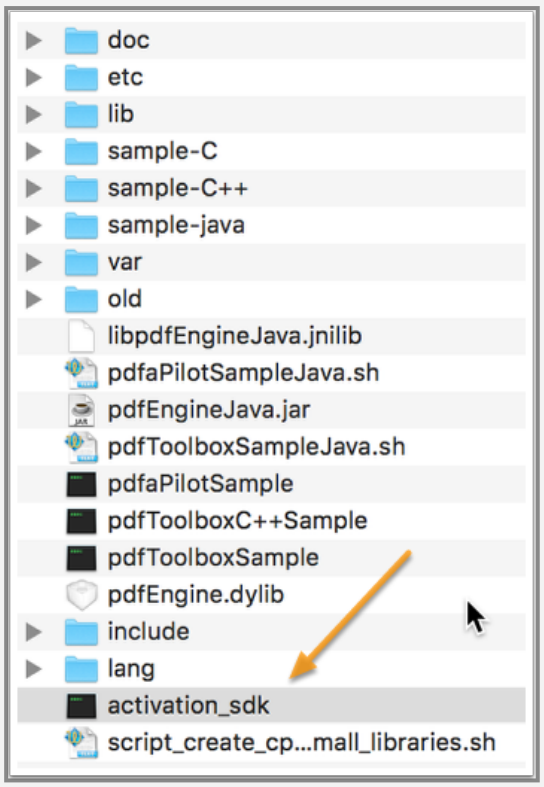
```
curl -XGET https://licenseserver.callassoftware.com/status.json -H "content-type:
application/json" -H "X-Wallet-ID: <the wallet ID>"
```

Click to copy

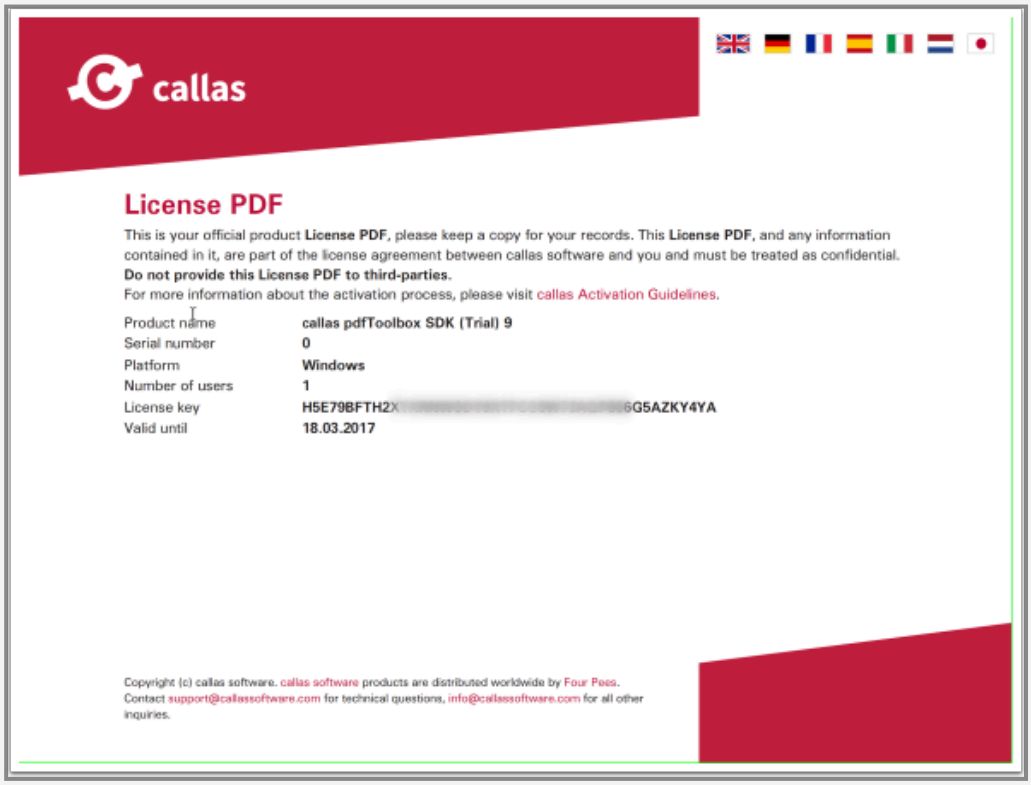
34. callas pdfToolbox SDK

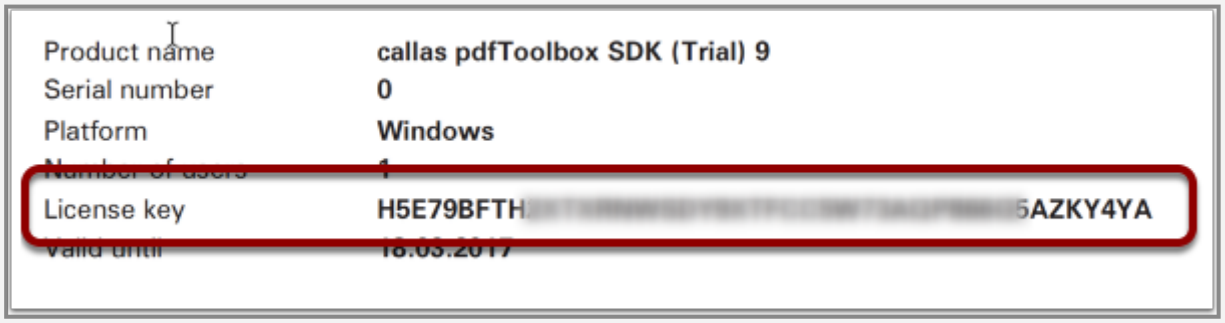
34.1 Activation and Deactivation of pdfToolbox SDK

After downloading and unarchiving the pdfToolbox SDK folder you will find a command line tool "activation_sdk" in the library folder.



You should also have received a License PDF together with the download path for the library. (If you do not have a License PDF, please send an email to sales@callassoftware.com.) The License PDF contains your License key as well as some other information.





In order to activate the library, use the command line activation tool with the --**keycode** parameter, your name, your company's name, and the **License PDF** file:

```
./activation_sdk --keycode <name> <company> <path to License PDF file>
```

Click to copy
Example:

```
./activation_sdk --keycode "Mary Smith" "The Printer Inc." /path/to/file/License_200012345.pdf
```

Click to copy

The tool will then output a license request with further instructions how the request has to be sent via email to the activation server. After you have done so, the activation server will send an **Activation PDF** back to you. You should then continue the installation using the --**activation** parameter and the **Activation PDF**:

```
./activation_sdk --activate <Path to Activation PDF>
```

Click to copy
Example:

```
./activation_sdk --activate /path/to/file/Activation.pdf
```

Click to copy

The License Key (as printed on the License PDF) is needed in each initialization call to the library as described in the SDK documentation. See first steps in the next article...

Deactivation of pdfToolbox SDK

As the activation (and the resulting license file) is bound to the hardware. It is necessary to deactivate a license on one machine before an activation takes place on the new machine.

In order to deactivate the license, use the --**deactivate** parameter and the **Activation code**:

```
./activation_sdk --deactivate <activation code>
```

Click to copy
The **<ativation code>** for the **deactivation** can be determined using the --**status** parameter:

```
./activation_sdk --status
```

Click to copy
To complete the deactivation, the output of the command has to be sent manually to the activation server by e-mail.

34.2 callas pdfEngine SDK: First steps

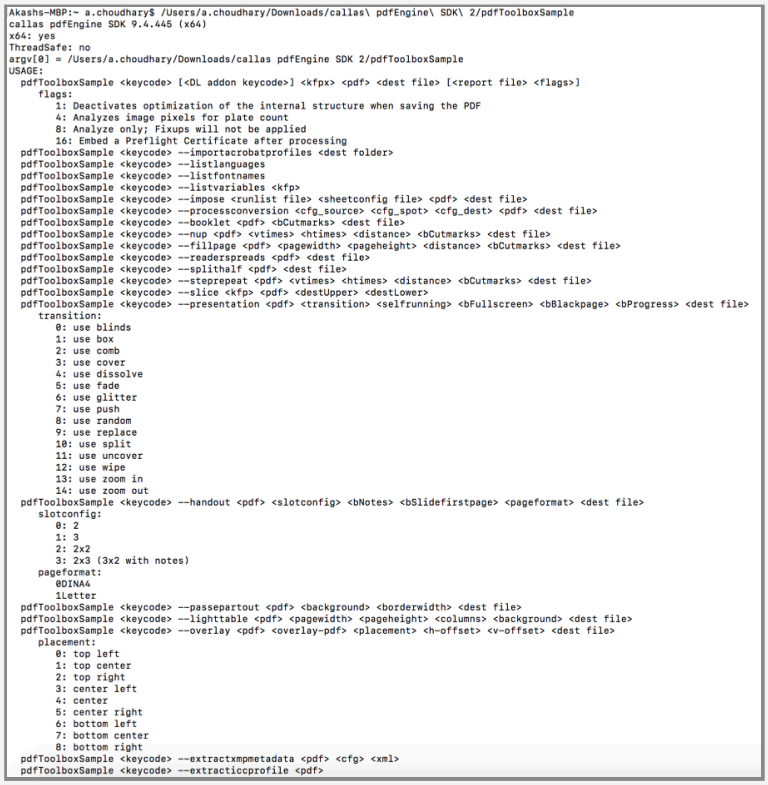
callas pdfEngine SDK is the name of callas' software development kit package name. The product related features (pdfToolbox or pdfaPilot) are activated using a license. This package includes full documentation and sample codes and contains executables and source code for all supported environments (C/C++, C# and Java).

To know more about the file components inside the package, go to File components and their use in pdfToolbox SDK article.

Executing a sample using callas pdfEngine SDK

Once you have activated your SDK using a license, you can run a sample, in this example 'pdfToolboxSample'. You can see below all the pre defined modes, along with the arguments that can be executed.

For example, pdfToolboxSample <keycode> --listlanguages



You can find a list of all predefined Profiles that come with the package here.

Executing a profile

In the example below, a profile 'Convert to PDFX-4 (PSO Coated v3 (ECI))' has been executed on the source file 'hello.pdf'.
The command looks like this:

```
/Users/a.choudhary/Downloads/callas pdfEngine SDK 2/pdfToolboxSample 'license code' './var/Profiles/PDFX compliance/Convert to PDFX-4 (PSO Coated v3 (ECI)).kfpx' './hello.pdf' './out.pdf'
```

Click to copy
The command has the apt arguments, which are:

1. path to the pdfToolboxSample (executable)

2. the keycode (from the license file that you might have received from callas or its resellers)

3. path to the profile that needs to be executed (.kfpx)

4. path to the input file

5. path and name of the output file

On executing:

```
Akashs-MBP:~ a.choudhary$ /Users/a.choudhary/Downloads/callas\ pdfEngine\ SDK\ 2/pdfToolboxSample JDEAXKQ3JAXN9HH695YC93QKHMWBMLPFTE65TJ6B
3CYS /Users/a.choudhary/Downloads/callas\ pdfEngine\ SDK\ 2/var/Profiles/PDFX\ compliance/Convert\ to\ PDFX-4\ \(PSO\ Coated\ v3\ \(ECI\)\
).kfpx /Users/a.choudhary/Downloads/callas\ pdfEngine\ SDK\ 2/hello.pdf out.pdf
callas pdfEngine SDK 9.4.445 (x64)
x64: yes
ThreadSafe: no
argv[0] = /Users/a.choudhary/Downloads/callas pdfEngine SDK 2/pdfToolboxSample
argv[1] = JDEAXKQ3JAXN9HH695YC93QKHMWBMLPFTE65TJ6B3CYS
argv[2] = /Users/a.choudhary/Downloads/callas pdfEngine SDK 2/var/Profiles/PDFX compliance/Convert to PDFX-4 (PSO Coated v3 (ECI)).kfpx
argv[3] = /Users/a.choudhary/Downloads/callas pdfEngine SDK 2/hello.pdf
argv[4] = out.pdf
callas pdfEngine SDK initialized
Running callas pdfEngine SDK samples...

Creating PRC engine
Loading Package
 Package: /Users/a.choudhary/Downloads/callas pdfEngine SDK 2/var/Profiles/PDFX compliance/Convert to PDFX-4 (PSO Coated v3 (ECI)).kfpx
10%
72%
73%
Verifying preferences: 76%
Verifying preferences: 79%
Verifying preferences: 82%
Verifying preferences: 85%
Verifying preferences: 89%
Verifying preferences: 92%
Verifying preferences: 95%
Verifying preferences: 98%
Verifying preferences: 100%
100%
Running profile
 PDF:     /Users/a.choudhary/Downloads/callas pdfEngine SDK 2/hello.pdf
5%
Saving PDF file out.pdf
Applying fixups: 11%
Applying fixups on page 1: 27%
Applying fixups on page 1: 30%
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
Fixup "Convert Color"
  Status: Success
```

Success!

34.3 Help? Displaying program information for pdfToolbox SDK

If you use the command line tool on command line with --help, you see the overview of options about all available commands for processing:

```
./activation_sdk --help
```

Click to copy

To get an overview about all the available options for a specific command, run:

```
./activation_sdk --help <command>
```

Click to copy

Commands like:

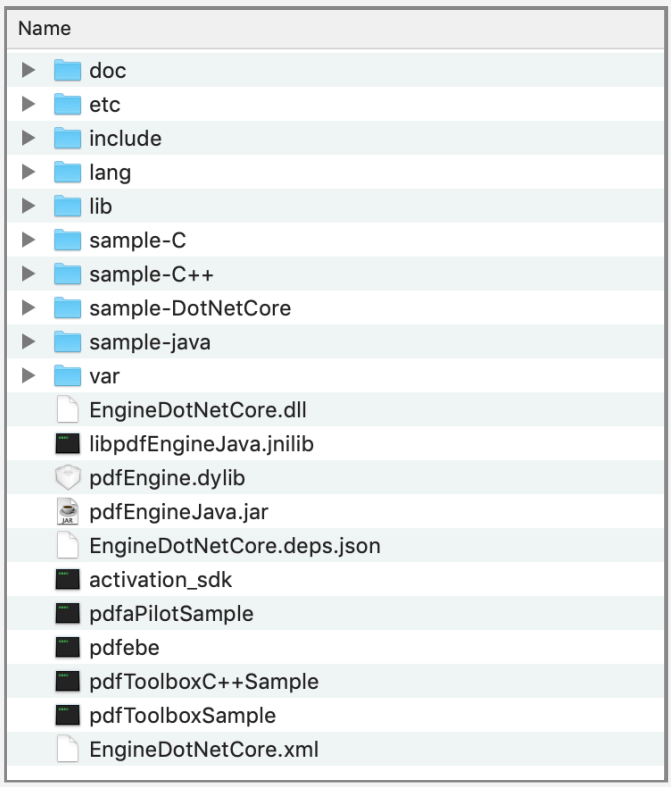
--version show version information

--activate Activate license

--deactivate Deactivate license

34.4 File components and their use in the SDK

Package too big? You can remove some components from the SDK and reduce the size of the software. Here is what you can delete without hampering the outcome of the software.



What to keep

For a smoother usage of the SDK, we recommend that you keep the following components

All			
Platform	Dependencies	Configuration	Language
Windows	all *.dll, .exe and *.ppi	etc	lang
OS X	lib	etc	lang
Unix	lib	etc	lang

Java Wrapper	
Windows	pdfEngineJava.jar and pdfEngineJava.dll
OS X	pdfEngineJava.jar and libpdfEngineJava.jnilib
Unix	pdfEngineJava.jar and libpdfEngineJava.so

.NET Wrapper	
Windows	pdfEngineDotNet.dll

.NET Core Wrapper (starting pdfEngine 12)	
Windows	EngineDotNetCore.*
MacOS	EngineDotNetCore.*
Linux	lib/EngineDotNetCore.*

What can be removed

The following components could be safely removed to reduce the delivery size.

Component	Functionality
doc	Documentation
include	C/C++ API Header
sample-C	C API Sample
sample-C++	C++ API Sample
sample-DotNet	.NET Sample
sample-java	Java Sample
var	Predefined configuration files (e.g. Profiles, Templates)

Functionality based

To further reduce the package size, the following subfolders can be removed from the 'etc' folder- depending on your requirements.

Sub-folder	Functionality
etc/Actions/Imposition	If no Arrange action is used
etc/Actions/LFP	If no Tiling or Grommet action is used
etc/Actions/PlaceContents	If no PlaceContent Fixups OR If no Place Barcode or Text Fixups are used
etc/APDFL	If no font embedding or no PDF/A conversion is used (or font situation is clear)
etc/Backgrounds	If no layer/image mask report is used
etc/Certify	If no Preflight certificate should be embedded
etc/ColorConversion	If no color conversion is used
etc/FontSubstitution	If no font substitution is used (for PDF/A)
etc/HtmlConverter	If no PDF report based on HTML template is used
etc/Inventory	If no inventory report is used
etc/MailConverter	If no emails are processed
etc/OCRTool	If no OCR needs to be done to text (Fixup: Create invisible text via OCR)
etc/PDFAExtSchema	If not PDF/A metadata entry needs to be embedded
etc/PDFOfficeTool	If no Office-files are processed
etc/PDFPSTool	If no PostScript-files are processed
etc/pmime	If no unknown files or wrong extensions are processed
etc/Reports	If no PDF/A-HTML Report or ZUGFeRD or CxF is used
etc/UnpackTool	If no archives (.zip) are processed or Profiles with Fixup

Sub-folder	Functionality
	"Place content on page" or Protected Profiles are used
etc/Variables	If no JavaScript post mortem debugger is used
etc/Visualizer	If no Comparison is used
etc/QuickTools	If no QuickFix or QuickCheck is required

34.5 .Net Core migration guide

The pdfEngine SDK v12 includes the new callas .Net Core language binding that replaces the existing callas .Net Framework language binding. It is available for multiple platforms (Windows, Linux, MacOS).

This migration guide is intended to support customers who are using existing applications developed with the previous, now obsolete *callas .Net Framework.*

The *callas .Net Core* language binding is not compatible with the *callas .Net Framework* language binding.

We use the term .Net Core when we actually mean .Net Standard. The reason is that the distinction between .Net Standard and .Net Framework often led to confusion.

callas .Net Framework language binding

The callas .Net Framework language binding assembly is still available inside the *var/legacy* subdirectory of the Engine SDK. However it is no longer maintained and has become a *legacy* interface.

Assembly

- pdfEngineDot-Net.dll

For existing .Net Framework applications the *var/legacy/pdfEngineDotNet.dll* assembly needs to be copied into the top level directory.

The following sample programs have been removed:

- pdfaPilotDotNet-Sample.exe
- pdfaPilotDotNet-SampleThread-Safe.exe
- pdfToolboxDot-NetSample.exe
- pdfToolboxDot-NetSam-pleThread-Safe.exe
- sample-DotNet (the directory containing the sample source code)

callas .Net Core language binding

Assembly

- EngineDotNetCore.dll
- EngineDotNetCore.deps.json
- EngineDotNetCore.xml

This assembly (and its companion files) reside either in the top level directory (on Windows and MacOS) or in the *lib* sub-folder (on Linux).

The following sample programs have been added:

- sample-DotNet-Core/pdfTool-boxSampleDot-NetCore
- sample-DotNet-Core/pdfaPilot-SampleDotNet-Core

- sample-DotNetCore/src (the directory containing the sample source code)

In addition there is a sample-DotNetCore/Readme.txt containing further Usage and Build Instructions.

Main differences

Namespace

The namespace has been changed from CallasSoftware.PDFEngine to CallasSoftware.PDFEngineCore.

Naming conventions

The *PTB_* prefix has been removed from all API functions, Structures and Enum Types. In addition some API functions have been renamed, e.g. *PTB_LanguageEnum* has been renamed to *EnumerateLanguages.*

In general the new API functions try to follow a *Verb*<Something> paradigm.

Callbacks

API Functions with callbacks no longer have a corresponding *userData* parameter. All user Data needs to be passed using *Lambdas.*

Migration Example

As an example the former PTB_ListLanguages API call is used to demonstrate how it can be migrated to the new EnumerateLanguages API call.

.Net Framework

```
        public static Boolean listLangCB (TStringID idLang, IntPtr userData) {
            Console.WriteLine ("Lang: " + getString (idLang));
            return true;
        }


        private static CError.PTB_EError doEnumerateLanguages () {
            CLib.PTB_LanguageCB_Net cbLang = new CLib.PTB_LanguageCB_Net (list-
LangCB);
            CLib.PTB_LanguageEnum (cbLang, IntPtr.Zero);
            return CError.PTB_EError.PTB_eerrNone;
        }
```

Click to copy

.Net Core

```
        EError DoEnumerateLanguages()
        {
            LanguageCB cb = (StringID id) =>
            {
                Console.WriteLine("Lang: " + GetString(id));
                return true;
            };
            EnumerateLanguages(cb);
            return EError.None;
        }
```

Click to copy

IDE users

The callas .Net Core language binding is no longer compatible with Visual Studio 2017. At least Visual Studio 2019 (or higher) is required for development with Visual Studio.

Another possible alternative is to use the open source Visual Studio Code IDE.

Summary

.Net Core is the new recommended API (actually it is a .Net Standard API).

Since the .Net Framework language binding assembly is still included in the installation package, there are two alternative upgrade paths for existing .Net framework applications to use pdfEngine SDK v12:

- migrate to the new .Net Core API (recommended)
- continue using the legacy .Net Framework API

34.6 Predefined Profiles

Below is a list of all predefined Profiles that are shipped with callas SDK/CLI. You can find these Profiles in the package folder under:

var/Profiles

For the use of Profiles in web applications, all Profiles are also available in a cloud space. The path for this can also be found in the list below.

List of Profiles based on their categories are listed below:

- Convert colors
- Convert colors using DeviceLink
- Create PDF layers
- Digital printing and online publishing
- GWG 2022
- PDF
- PDF analysis
- PDF Fixups
- PDF version compatibility
- PDFA compliance
- PDFE compliance
- PDFUA compliance
- PDFVT compliance
- PDFX compliance
- PDFX and PDFA
- Packaging
- Place content
- Preflight Certificate
- Prepress
- Process Plans
- Processing Steps

Group: Convert colors

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
Comment	Converts RGB objects to CMYK (PSO Coated v3) with a specified tolerance for R=G=B, i.e. objects that are converted to CMYK gray. Spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Con-

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
	vert_RGB_to_CMYK_(PSO_Coated_v3)_with_specified_R%3DG%3DB_tolerance.kfpx
Name	Convert registration color to CMYK black only
Comment	Converts all registration color objects to DeviceCMYK black only.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_registration_color_to_CMYK_black_only.kfpx
Name	**Convert spot to CMYK and replace overprint with transparency where needed**
Comment	When spot color objects are converted to CMYK the overprint rules change which might lead to a dramatically changed appearance.This Process Plan analyzes objects in context and replaces overprint with transparency where needed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_spot_to_CMYK_and_replace_overprint_with_transparency_where_needed.kfpx
Name	**Convert to CMYK only (Coated GRACoL 2006)**
Comment	Converts the current PDF document to CMYK only, using Coated GRACoL 2006 as destination profile. All spot colors are converted to CMYK as well.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK_only_(Coated_GRACoL_2006).kfpx
Name	**Convert to CMYK only (ISO Coated v2 (ECI))**
Comment	Converts the current PDF document to CMYK only, using ISO Coated v2 (ECI) as destination profile. All spot colors are converted to CMYK as well.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK_only_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to CMYK only (Japan Color 2001 Coated)**
Comment	Converts the current PDF document to CMYK only, using Japan Color 2001 Coated as destination profile. All spot colors are converted to CMYK as well.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK_only_(Japan_Color_2001_Coated).kfpx
Name	Convert color to PSO Coated v3 (ECI) (convert spot colors to CMYK)

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
Comment	Converts the current PDF document to CMYK only, using PSO Coated v3 (ECI) as destination profile. All spot colors are converted to CMYK as well.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK_only_(PSO_Coated_v3_(ECI).kfpx
Name	**Convert to CMYK, keep spot colors (Coated GRACoL 2006)**
Comment	Converts the current PDF document to CMYK, using Coated GRACoL 2006 as destination profile. All spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK%2C_keep_spot_colors_(Coated_GRACoL_2006).kfpx
Name	**Convert to CMYK, keep spot colors (ISO Coated v2 (ECI))**
Comment	Converts the current PDF document to CMYK, using ISO Coated v2 (ECI) as destination profile. All spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK%2C_keep_spot_colors_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to CMYK, keep spot colors (Japan Color 2001 Coated)**
Comment	Converts the current PDF document to CMYK, using Japan Color 2001 Coated as destination profile. All spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK%2C_keep_spot_colors_(Japan_Color_2001_Coated).kfpx
Name	**Convert color to PSO Coated v3 (ECI) (keep s**pot colors)
Comment	Converts the current PDF document to CMYK, using PSO Coated v3 (ECI) as destination profile. All spot colors are maintained.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_CMYK%2C_keep_spot_colors_(PSO_Coated_v3_(ECI)).kfpx
Name	**Convert to grayscale**
Comment	Converts all colors (including spot colors) in the current PDF document to grayscale.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_grayscale.kfpx

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
Name	**Convert to sRGB**
Comment	Converts all colors (including spot colors) in the current PDF document to sRGB.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_to_sRGB.kfpx
Name	**Convert using DeviceLink Office RGB to ISO Coated v2 (ECI)**
Comment	Converts the current PDF document to CMYK (ISO Coated v2 (ECI)) as destination profile according to the Fogra PSD (Process Standard Digital) guidelines. A special DeviceLink profile makes sure that line art elements defined in RGB keep their vivid color and that colors which are different in the original PDF have similar distances in the result. RGB Gray gets converted to pure Black, Black text is set to overprint. The DeviceLink profile has been developed for preparation files created with office applications for the print process.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Convert_using_DeviceLink_Office_RGB_to_ISO_Coated_v2_(ECI).kfpx
Name	**Map color with specified color values**
Comment	Maps colors in the current PDF document based on parameters specified during runtime.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Map_color_with_specified_color_values.kfpx
Name	**Overprint issues related to spot color if converted to CMYK**
Comment	The Profile identifies overprinting or overprinted spot color objects that are problematic when converted to CMYK. The underlying reason is that the overprint rules are much different for spot and CMYK objects.Two problematic areas are analyzed: 1.) OPM is off and 2.) OPM is on and at least one colorant (C, M, Y or K) is used by foreground and background object.These two areas are analyzed for spot color objects in front of or behind CMYK objects or other spot color objects.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Overprint_issues_related_to_spot_color_if_converted_to_CMYK.kfpx
Name	**Tone value adjustment in midtones by -10%**
Comment	Reduces the tone values in midtones by 10% for device dependent CMYK and spot colors.

Name	Convert RGB to CMYK (PSO Coated v3) with specified R=G=B tolerance
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors/Tone_value_adjustment_in_midtones_by_-10prct.kfpx

Group: Convert colors using DeviceLink

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Comment	Converts objects from CMYK exchange color space (eciCMYK) to Sheetfed offset (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/CMYK_exchange_CS_to_Sheetfed_(DL_eciCMYK_to_ISOcoatedv2).kfpx
Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to PSO Coated v3)**
Comment	Converts objects from CMYK exchange color space (eciCMYK) to Sheetfed offset (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/CMYK_exchange_CS_to_Sheetfed_(DL_eciCMYK_to_PSOcoatedv3).kfpx
Name	**RGB to CMYK (DeviceLink: Adobe RGB to ISO Coated v2)**
Comment	Converts objects from RGB (Adobe RGB) to CMYK (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/RGB_to_CMYK_(DL_AdobeRGB_to_ISOcoatedv2).kfpx
Name	**RGB to CMYK (DeviceLink: AdobeRGB to PSO Coated v3)**
Comment	Converts objects from RGB (Adobe RGB) to CMYK (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/RGB_to_CMYK_(DL_AdobeRGB_to_PSOcoatedv3).kfpx
Name	**RGB to CMYK (DeviceLink: sRGB to ISO Coated v2)**
Comment	Converts objects from RGB (sRGB) to CMYK (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/RGB_to_CMYK_(DL_sRGB_to_ISOcoatedv2).kfpx
Name	**RGB to CMYK (DeviceLink: sRGB to PSO Coated v3)**
Comment	Converts objects from RGB (sRGB) to CMYK (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/RGB_to_CMYK_(DL_sRGB_to_PSOcoatedv3).kfpx
Name	**Sheetfed offset to CMYK exchange color space (DeviceLink: ISO Coated v2 to eciCMYK)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to CMYK exchange color space (eciCMYK) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_CMYK_exchange_CS_(DL_ISOcoatedv2_to_eciCMYK).kfpx
Name	**Sheetfed offset to CMYK exchange color space (DeviceLink: PSO Coated v3 to eciCMYK)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to CMYK exchange color space (eciCMYK) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_CMYK_exchange_CS_(DL_PSOcoatedv3_to_eciCMYK).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: GRACoL2006 to Web Coated SWOP grade 3)**
Comment	Converts objects from Sheetfed offset (GRACoL 2006 Coated 1v2) to Web offset (Web Coated SWOP grade 3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_GRACoL2006_to_WebcoatedSWOPgrade3).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: GRACoL2006 to Web coated SWOP grade 5)**
Comment	Converts objects from Sheetfed offset (GRACoL 2006 Coated 1v2) to Web offset (Web Coated SWOP grade 5) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_GRACoL2006_to_WebcoatedSWOPgrade5).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to ISOnewspaper26v4)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (ISOnewspaper26v4) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_ISOnewspaper26v4).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to ISO Web Coated)**

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (ISO Web Coated) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_ISOwebcoated).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to PSO LWC Improved)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (PSO LWC Improved) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_PSOLWCimproved).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to PSO LWC Standard)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (PSO LWC Standard) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_PSOLWCstandard).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to PSO SC-B paper v3 (FOGRA54))**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (PSO SC-B paper v3 (FOGRA54)) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_PSOSC-Bpaperv3).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: ISO Coated v2 to SC Paper)**
Comment	Converts objects from Sheetfed offset (ISO Coated v2) to Web offset (SC Pa-

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	per) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_ISOcoatedv2_to_SCpaper).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to ISOnewspaper26v4)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (ISOnewspaper26v4) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_ISOnewspaper26v4).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to ISO Web Coated)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (ISO Web Coated) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_ISOwebcoated).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to PSO LWC Improved)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (PSO LWC Improved) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_PSOLWCimproved).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to PSO LWC Standard)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (PSO

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	LWC Standard) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_PSOLWCstandard).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to PSO SC-B Paper v3)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (PSO SC-B paper v3 (FOGRA54)) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_PSOSC-Bpaperv3).kfpx
Name	**Sheetfed offset to Web offset (DeviceLink: PSO Coated v3 to SC Paper)**
Comment	Converts objects from Sheetfed offset (PSO Coated v3) to Web offset (SC Paper) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed_to_Web_(DL_PSOcoatedv3_to_SCpaper).kfpx
Name	**Sheetfed offset, Europe to Japan (DeviceLink: ISO Coated v2 to Japan Color 2011 Coated)**
Comment	Converts objects from Europe (ISO Coated v2) to Japan (Japan Color 2011 Coated) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Europe_to_Japan_(DL_ISOcoatedv2_to_Japan2011Coated).kfpx
Name	**Sheetfed offset, Europe to Japan (DeviceLink: PSO Coated v3 to Japan Color 2011 Coated)**
Comment	Converts objects from Europe (PSO Coated v3) to Japan (Japan Color 2011

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	Coated) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Europe_to_Japan_(DL_PSOcoatedv3_to_Japan2011Coated).kfpx
Name	**Sheetfed offset, Europe to US (DeviceLink: ISO Coated v2 to GRACoL2006)**
Comment	Converts objects from Europe (ISO Coated v2) to US (GRACol 2006 Coated 1v2) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Europe_to_US_(DL_ISOcoatedv2_to_GRACoL2006).kfpx
Name	**Sheetfed offset, Europe to US (DeviceLink: PSO Coated v3 to GRACoL2006)**
Comment	Converts objects from Europe (PSO Coated v3) to US (GRACol 2006 Coated 1v2) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Europe_to_US_(DL_PSOcoatedv3_to_GRACoL2006).kfpx
Name	**Sheetfed offset, Japan to Europe (DeviceLink: Japan Color 2011 Coated to ISO Coated v2)**
Comment	Converts objects from Japan (Japan Color 2011 Coated) to Europe (ISO Coated v2) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Japan_to_Europe_(DL_Japan2011Coated_to_ISOcoatedv2).kfpx

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Name	**Sheetfed offset, Japan to Europe (DeviceLink: Japan Color 2011 Coated to PSO Coated v3)**
Comment	Converts objects from Japan (Japan Color 2011 Coated) to Europe (PSO Coated v3) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_Japan_to_Europe_(DL_Japan2011Coated_to_PSOcoatedv3).kfpx
Name	**Sheetfed offset, US to Europe (DeviceLink: GRACoL2006 to ISO Coated v2)**
Comment	Converts objects from US (GRACol 2006 Coated 1v2) to Europe (ISO Coated v2) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_US_to_Europe_(DL_GRACoL2006_to_ISOcoatedv2).kfpx
Name	**Sheetfed offset, US to Europe (DeviceLink: GRACoL2006 to PSO Coated v3)**
Comment	Converts objects from US (GRACol 2006 Coated 1v2) to Europe (PSO Coated v3) for Sheetfed offset using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_US_to_Europe_(DL_GRACoL2006_to_PSOcoatedv3).kfpx
Name	**Sheetfed offset, coated paper (DeviceLink: ISO Coated v2 to PSO Coated v3)**
Comment	Converts objects from Sheetfed offset according to ISO Coated v2 to PSO Coated v3 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_coated_paper_(DL_ISOcoatedv2_to_PSOcoatedv3).kfpx
Name	**Sheetfed offset, coated paper (DeviceLink: PSO Coated v3 to ISO Coated v2)**
Comment	Converts objects from Sheetfed offset according to PSO Coated v3 to ISO Coated v2 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed,_coated_paper_(DL_PSOcoatedv3_to_ISOcoatedv2).kfpx
Name	**Sheetfed offset, coated to uncoated (DeviceLink: PSO Coated v3 to PSO Uncoated v3)**
Comment	Converts objects from Sheetfed offset, coated paper (PSO Coated v3) to Sheetfed offset, uncoated paper (PSO Uncoated v3 (FOGRA52)) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_coated_to_uncoated_(DL_PSOcoatedv3_to_PSOuncoatedv3).kfpx
Name	**Sheetfed offset, reduce to 280% ink limit (DeviceLink: ISO Uncoated)**
Comment	Reduces the ink coverage to 280% for Sheetfed offset (ISO Uncoated) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_280%25_ink_limit_(DL_ISOuncoated).kfpx
Name	**Sheetfed offset, reduce to 280% ink limit (DeviceLink: PSO Uncoated v3)**
Comment	Reduces the ink coverage to 280% for Sheetfed offset (PSO Uncoated v3 (FOGRA52)) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_280%25_ink_limit_(DL_PSOuncoatedv3).kfpx
Name	**Sheetfed offset, reduce to 300% ink limit (DeviceLink: ISO Coated v2)**
Comment	Reduces the ink coverage to 300% for Sheetfed offset (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_300%25_ink_limit_(DL_ISOcoatedv2).kfpx
Name	**Sheetfed offset, reduce to 300% ink limit (DeviceLink: PSO Coated v3)**
Comment	Reduces the ink coverage to 300% for Sheetfed offset (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_300%25_ink_limit_(DL_PSOcoatedv3).kfpx
Name	**Sheetfed offset, reduce to 320% ink limit (DeviceLink: GRACoL2006)**
Comment	Reduces the ink coverage to 320% for Sheetfed offset (GRACoL 2006 Coated 1v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_320%25_ink_limit_(DL_GRACoL2006).kfpx
Name	**Sheetfed offset, reduce to 330% ink limit (DeviceLink: ISO Coated v2)**
Comment	Reduces the ink coverage to 330% for Sheetfed offset (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_330%25_ink_limit_(DL_ISOcoatedv2).kfpx

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Name	**Sheetfed offset, reduce to 340% ink limit (DeviceLink: Japan Color 2011)**
Comment	Reduces the ink coverage to 340% for Sheetfed offset (Japan Color 2011) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_reduce_to_340%25_ink_limit_(DL_Japan2011Coated).kfpx
Name	**Sheetfed offset, uncoated paper (DeviceLink: PSO Uncoated v2 to PSO Uncoated v3)**
Comment	Converts objects from Sheetfed offset, uncoated paper according to PSO Uncoated v2 to PSO Uncoated v3 (FOGRA52) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_uncoated_paper_(DL_PSOuncoatedv2_to_PSOuncoatedv3).kfpx
Name	**Sheetfed offset, uncoated paper (DeviceLink: PSO Uncoated v3 to PSO Uncoated v2)**
Comment	Converts objects from Sheetfed offset, uncoated paper according to PSO Uncoated v3 (FOGRA52) to PSO Uncoated v2 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Sheetfed%2C_uncoated_paper_(DL_PSOuncoatedv3_to_PSOuncoatedv2).kfpx
Name	**Web offset (DeviceLink: ISO Web Coated to ISOnewspaper26v4)**
Comment	Converts objects from Web offset according to ISO Web Coated to ISOnewspaper26v4 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web_(DL_ISOwebcoated_to_ISOnewspaper26v4).kfpx
Name	**Web (DeviceLink: PSOLWCimproved to ISOnewspaper26v4)**
Comment	Converts objects from Web offset according to PSO LWC Improved to ISOnewspaper26v4 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web_(DL_PSOLWCimproved_to_ISOnewspaper26v4).kfpx
Name	**Web offset (DeviceLink: Web Coated SWOP grade 3 to Web Coated SWOP grade 5)**
Comment	Converts objects from Web offset according to Web Coated SWOP grade 3 to Web Coated SWOP grade 5 using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web_(DL_WebcoatedSWOPgrade3_to_WebcoatedSWOPgrade5).kfpx
Name	**Web offset to Sheetfed offset (DeviceLink: PSO SC-B paper v3 (FOGRA54) to ISO Coated v2)**
Comment	Converts objects from Web offset (PSO SC-B paper v3 (FOGRA54)) to Sheetfed offset (ISO Coated v2) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web_to_Sheetfed_(DL_PSOSC-Bpaperv3_to_ISOcoatedv2).kfpx
Name	**Web offset to Sheetfed offset (DeviceLink: PSO SC-B paper v3 (FOGRA54) to PSO Coated v3)**
Comment	Converts objects from Web offset (PSO SC-B paper v3 (FOGRA54)) to Sheetfed offset (PSO Coated v3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_us-

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
	ing_DeviceLink/Web_to_Sheetfed_(DL_PSOSC-Bpaperv3_to_PSOcoatedv3).kfpx
Name	**Web offset, reduce to 240% ink limit (DeviceLink: ISOnewspaper26v4)**
Comment	Reduces the ink coverage to 240% for Web offset (ISOnewspaper26v4) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_240%25_ink_limit_(DL_ISOnewspaper26v4).kfpx
Name	**Web offset, reduce to 280% ink limit (DeviceLink: Web Coated SWOP grade 5)**
Comment	Reduces the ink coverage to 280% for Web offset (Web Coated SWOP grade 5) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_280%25_ink_limit_(DL_WebcoatedSWOP-grade5).kfpx
Name	**Web offset, reduce to 300% ink limit (DeviceLink: ISO Web Coated)**
Comment	Reduces the ink coverage to 300% for Web offset (ISO Web Coated) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_300%25_ink_limit_(DL_ISOwebcoated).kfpx
Name	**Web offset, reduce to 300% ink limit (DeviceLink: PSO LWC Improved)**
Comment	Reduces the ink coverage to 300% for Web offset (PSO LWC Improved) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_300%25_ink_limit_(DL_PSOLWCimproved).kfpx

Name	**CMYK exchange color space to Sheetfed offset (DeviceLink: eciCMYK to ISO Coated v2)**
Name	**Web offset, reduce to 300% ink limit (DeviceLink: PSO LWC Standard)**
Comment	Reduces the ink coverage to 300% for Web offset (PSO LWC Standard) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_300%25_ink_limit_(DL_PSOLWCstandard).kfpx
Name	**Web offset, reduce to 300% ink limit (DeviceLink: Web Coated SWOP grade 3)**
Comment	Reduces the ink coverage to 300% for Web offset (Web Coated SWOP grade 3) using a DeviceLink profile. DeviceLink profiles allow for more precise specification of the conversion process so that the results are often better than for regular ICC based color conversion.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Convert_colors_using_DeviceLink/Web%2C_reduce_to_300%25_ink_limit_(DL_WebcoatedSWOP-grade3).kfpx

Group: Create PDF layers

Name	Create separate layers for vector objects, text and images
Comment	Puts page objects on separate layers for vector objects, text and images.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Create_separate_layers_for_vector_objects%2C_text_and_images.kfpx
Name	Put all image objects on layers
Comment	Puts images on separate layers for color, grayscale and bitmaps.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Put_all_image_objects_on_layers.kfpx
Name	Put all text objects on a layer
Comment	Puts all text objects on a new layer.

Name	Create separate layers for vector objects, text and images
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Put_all_text_objects_on_a_layer.kfpx
Name	Put all transparent objects on layers
Comment	Puts all transparent objects on new layers according to the type of transparency (blend mode, constant alpha etc.)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Put_all_transparent_objects_on_layers.kfpx
Name	Put objects on layers according to their overprint settings
Comment	Creates layers with overprinting objects and objects that are set to knock out.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Create_PDF_layers/Put_objects_on_layers_according_to_their_overprint_settings.kfpx

Group: Digital printing and online publishing

Name	Convert for Digital printing (B_W)
Comment	Optimizes the current PDF document for digital printing. Converts all colors - including spot colors - to grayscale.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Digital_printing_and_online_publishing/Digital_printing_(B_W).kfpx
Name	Convert for Digital printing (color)
Comment	Optimizes the current PDF document for digital printing. Converts all colors - including spot colors - to CMYK.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Digital_printing_and_online_publishing/Digital_printing_(color).kfpx
Name	Convert for Online publishing
Comment	Converts the PDF for online publishing with optimization for size. Most PDFs will be significantly smaller then as all content not required for this purpose will be discarded and image resolution will be downsampled to 96 ppi. This approach makes the PDF as small as possible.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Digital_print-

Name	Convert for Digital printing (B_W)
	ing_and_online_publishing/Online_publishing.kfpx

Group: GWG 2022

Name	Digital Print
Comment	Based on GWG 2022 specifications. Variant: Digital Print
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Digital_Print.kfpx
Name	Large Format Print
Comment	Based on GWG 2022 specifications. Variant: Large Format Print
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Large_Format_Print.kfpx
Name	Magazine Ads CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: Magazine Ads CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Magazine_Ads_CMYK_and_RGB.kfpx
Name	Magazine Ads CMYK
Comment	Based on GWG 2022 specifications. Variant: Magazine Ads CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Magazine_Ads_CMYK.kfpx
Name	Newspaper Ads CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: Newspaper Ads CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Newspaper_Ads_CMYK_and_RGB.kfpx
Name	Newspaper Ads CMYK
Comment	Based on GWG 2022 specifications. Variant: Newspaper Ads CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Newspa-

Name	Digital Print
	per_Ads_CMYK.kfpx
Name	Packaging Flexo - Corrugated Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Corrugated Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Corrugated_Box.kfpx
Name	Packaging Flexo - Corrugated Display
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Corrugated Display
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Corrugated_Display.kfpx
Name	Packaging Flexo - Flexible
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Flexible
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Flexible.kfpx
Name	Packaging Flexo - Folding Carton Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Folding Carton Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Folding_Carton_Box.kfpx
Name	Packaging Flexo - Label
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Label
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Label.kfpx
Name	Packaging Flexo - Leaflet
Comment	Based on GWG 2022 specifications. Variant: Packaging Flexo -> Leaflet
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Flexo_-_Leaflet.kfpx
Name	Packaging Gravure - Corrugated Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Corrugated Box

Name	Digital Print
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Corrugated_Box.kfpx
Name	Packaging Gravure - Corrugated Display
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Corrugated Display
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Corrugated_Display.kfpx
Name	Packaging Gravure - Flexible
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Flexible
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Flexible.kfpx
Name	Packaging Gravure - Folding Carton Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Folding Carton Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Folding_Carton_Box.kfpx
Name	Packaging Gravure - Label
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Label
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Label.kfpx
Name	Packaging Gravure - Leaflet
Comment	Based on GWG 2022 specifications. Variant: Packaging Gravure -> Leaflet
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Gravure_-_Leaflet.kfpx
Name	Packaging Offset - Corrugated Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Corrugated Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Corrugated_Box.kfpx
Name	Packaging Offset - Corrugated Display

Name	Digital Print
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Corrugated Display
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Corrugated_Display.kfpx
Name	Packaging Offset - Flexible
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Flexible
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Flexible.kfpx
Name	Packaging Offset - Folding Carton Box
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Folding Carton Box
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Folding_Carton_Box.kfpx
Name	Packaging Offset - Label
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Label
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Label.kfpx
Name	Packaging Offset - Leaflet
Comment	Based on GWG 2022 specifications. Variant: Packaging Offset -> Leaflet
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Packaging_Offset_-_Leaflet.kfpx
Name	SheetCMYK CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: SheetCMYK CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/SheetCMYK_CMYK_and_RGB.kfpx
Name	SheetCMYK CMYK
Comment	Based on GWG 2022 specifications. Variant: SheetCMYK CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/SheetCMYK_CMYK.kfpx

Name	Digital Print
Name	SheetSpot CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: SheetSpot CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/SheetSpot_CMYK_and_RGB.kfpx
Name	SheetSpot CMYK
Comment	Based on GWG 2022 specifications. Variant: SheetSpot CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/SheetSpot_CMYK.kfpx
Name	WebCMYK CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: WebCMYK CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebCMYK_CMYK_and_RGB.kfpx
Name	WebCMYK CMYK
Comment	Based on GWG 2022 specifications. Variant: WebCMYK CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebCMYK_CMYK.kfpx
Name	WebCMYKNews CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: WebCMYKNews CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebCMYKNews_CMYK_and_RGB.kfpx
Name	WebCMYKNews CMYK
Comment	Based on GWG 2022 specifications. Variant: WebCMYKNews CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebCMYKNews_CMYK.kfpx
Name	WebSpot CMYK + RGB
Comment	Based on GWG 2022 specifications. Variant: WebSpot CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/Web-

Name	Digital Print
	Spot_CMYK_and_RGB.kfpx
Name	WebSpot CMYK
Comment	Based on GWG 2022 specifications. Variant: WebSpot CMYK
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/GWG_2022/WebSpot_CMYK.kfpx

Group: PDF

Name	Uses PDF 2.0 features
Comment	Determines whether a PDF uses any print relevant features of PDF 2.0. Usually these features will be ignored by tools or devices that are not PDF 2.0 ready.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF/Uses_PDF_2.0_features.kfpx

Group: PDF analysis

Name	Analyze pages for effectively used plates
Comment	Analyzes all pages for effectively used plates. All pages are internally rendered, a plate is considered as being used if the minimum ink amount within at least one rectangle of 10 by 10 mm is reached. The threshold for the minimum ink amount can be specified during runtime. Device independent and all RGB colors should be converted into DeviceCMYK or DeviceGray beforehand.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/Analyze_pages_for_effectively_used_plates.kfpx
Name	Find CxF issues
Comment	CxF is an ISO standards that defines how to embed spectral measurements into a PDF Output Intent. This profile checks for some CxF related issues.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/Find_CxF_issues.kfpx
Name	List all hairlines

Name	Analyze pages for effectively used plates
Comment	CxF is an ISO standards that defines how to embed spectral measurements into a PDF Output Intent. This profile checks for some CxF related issues.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_all_hairlines.kfpx
Name	List black objects set to overprint
Comment	Reports all black objects that are set to overprint.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_black_objects_set_to_overprint.kfpx
Name	List color and grayscale images not within 250 to 450 ppi
Comment	Lists color and grayscale images with a resolution below 250 ppi and above 450 ppi. It depends on the image content whether this is sufficient.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_color_and_grayscale_images_not_within_250_to_450_ppi.kfpx
Name	**List invisible text objects**
Comment	Reports all text objects that are invisible. Invisible text is often used to overlay text (created by an OCR process) on top of scanned document pages.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_invisible_text_objects.kfpx
Name	**List objects using ICC/Lab/calibrated color**
Comment	Reports all page objects that use device independent color spaces (i.e. ICC based color spaces, Lab, CalRGB or CalGray color spaces).
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_objects_using_ICC_or_Lab_or_calibrated_color.kfpx
Name	**List page objects, grouped by type of object**
Comment	Reports page objects grouped by object type. Images are grouped by image resolution. This profile should only be used for one page at a time, as it may report a very large number of page objects.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_page_objects%2C_grouped_by_type_of_object.kfpx

Name	Analyze pages for effectively used plates
Name	**List potential font problems**
Comment	Lists possible font issues.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_potential_font_problems.kfpx
Name	**List potential overprint problems**
Comment	Visualizes overprint issues that might need further investigation. It might be useful to create a layer report from the results for better investigation.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_potential_overprint_problems.kfpx
Name	**List potential printing problems**
Comment	Lists potential printing problems.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_potential_printing_problems.kfpx
Name	**List rich black objects**
Comment	Lists all black objects which are also using Cyan, Magenta or Yellow.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_rich_black_objects.kfpx
Name	**List spot color objects**
Comment	Reports all spot color objects. Note: This profile does not report objects using Separation color spaces Cyan, Magenta, Yellow, Black, All or None.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_spot_color_objects.kfpx
Name	**List text using Courier inside Trim/BleedBox**
Comment	Reports all text objects which use Courier and which - partially or completely - lie inside the TrimBox or BleedBox. This can be used to search for fonts possibly substituted with Courier while ignoring page slugs which often use Courier but typically lie outside the TrimBox and BleedBox.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_text_using_Courier_inside_Trim_or_BleedBox.kfpx

Name	Analyze pages for effectively used plates
Name	**List text using non-embedded fonts**
Comment	Reports all text objects that use a font which is not embedded in the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_text_using_non-embedded_fonts.kfpx
Name	**List transparent objects**
Comment	Lists all page objects that use transparency. Objects are grouped by type of transparency.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/List_transparent_objects.kfpx
Name	**Will always hit**
Comment	-
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_analysis/Will_always_hit.kfpx

Group: PDF Fixups

Name	Convert all pages into CMYK images and preserve text information
Comment	Converts all page contents into CMYK images (150 ppi, high JPEG quality) and creates an invisible text copy for all text in order to preserve features like copying or searching of text.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Convert_all_pages_into_CMYK_images_and_preserve_text_information.kfpx
Name	Convert all pages into RGB images and preserve text information
Comment	Converts all page contents into RGB images (150 ppi, high JPEG quality) and creates an invisible text copy for all text in order to preserve features like copying or searching of text.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Convert_all_pages_into_RGB_images_and_preserve_text_information.kfpx
Name	Convert fonts to outlines

Name	Convert all pages into CMYK images and preserve text information
Comment	Converts fonts to outlines.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Convert_fonts_to_outlines.kfpx
Name	Derive page geometry boxes from crop marks
Comment	Inserts TrimBox and BleedBox on the basis of existing crop marks.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Derive_page_geometry_boxes_from_crop_marks.kfpx
Name	Downsample image resolution to 150 ppi (bitmaps to 300 ppi)
Comment	Downsamples image resolution of color and grayscale images to 150 ppi and image resolution of bitmap images to 300 ppi. Finally images are recompressed with low quality.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Downsample_image_resolution_to_150_ppi_(bitmaps_to_300_ppi).kfpx
Name	Downsample image resolution to 200 ppi (bitmaps to 600 ppi)
Comment	Downsamples image resolution of color and grayscale images to 200 ppi and image resolution of bitmap images to 600 ppi.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Downsample_image_resolution_to_200_ppi_(bitmaps_to_600_ppi).kfpx
Name	Downsample image resolution to 350 ppi (bitmaps to 1200 ppi)
Comment	Downsamples image resolution of color and grayscale images to 350 ppi and image resolution of bitmap images to 1200 ppi.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Downsample_image_resolution_to_350_ppi_(bitmaps_to_1200_ppi).kfpx
Name	Downsample image resolution to specified value
Comment	Downsamples image resolution of color, grayscale and bitmap images to values on the basis of settings that can be defined when running the profile. Finally images are recompressed with high quality.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Downsample_image_resolution_to_specified_value.kfpx

Name	Convert all pages into CMYK images and preserve text information
Name	Embed missing fonts from specified additional font folder only
Comment	If a PDF uses fonts which are not embedded into the PDF file they are embedded when using this fixup. The fonts have to be available in the additional font folder which can be defined when running the fixup.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Embed_missing_fonts_from_specified_additional_font_folder_only.kfpx
Name	Embed missing fonts
Comment	Embeds missing fonts in the PDF if they are available in the system.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Embed_missing_fonts.kfpx
Name	Fix potential font problems
Comment	Fixes possible font issues, where possible, to prevent bad output.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Fix_potential_font_problems.kfpx
Name	Fix problems in PDF tagging structure
Comment	Fixes possible font issues, where possible, to prevent bad output.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Fix_problems_in_PDF_tagging_structure.kfpx
Name	Flatten annotations and form fields
Comment	Flattens annotations and interactive form fields if present. The appearance of all annotations and all form fields will become a part of the page content.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Flatten_annotations_and_form_fields.kfpx
Name	Flatten overprints
Comment	Flattens overprinting objects and underlying objects so that overprints are maintained even if the output device is not capable of handling
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Flatten_overprints.kfpx

Name	Convert all pages into CMYK images and preserve text information
Name	Flatten transparency (high resolution)
Comment	Flattens transparency in the current PDF document using high resolution settings.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Flatten_transparency_(high_resolution).kfpx
Name	Recompress color and grayscale images to JPEG (high quality)
Comment	Recompresses color and grayscale images to JPEG with high quality.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Recompress_color_and_grayscale_images_to_JPEG_(high_quality).kfpx
Name	Recompress color and grayscale images to JPEG2000 (lossless)
Comment	Recompresses color and grayscale images to JPEG2000.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Recompress_color_and_grayscale_images_to_JPEG2000_(lossless).kfpx
Name	Recompress color and grayscale images to ZIP
Comment	Recompresses color and grayscale images to ZIP.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Recompress_color_and_grayscale_images_to_ZIP.kfpx
Name	Scale pages to A4 if not larger than A3
Comment	Scales pages of the PDF proportionally to A4 if their page size is not larger than A3.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Scale_pages_to_A4_if_not_larger_than_A3.kfpx
Name	Scale pages to A4
Comment	Scales all pages of the PDF proportionally to A4
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_fixups/Scale_pages_to_A4.kfpx

Group: PDF version compatibility

Name	Compatible with PDF 1.2
Comment	Verifies whether the current document is compatible with PDF 1.2 and reports errors if features are used that are not supported in PDF 1.3.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.2.kfpx
Name	Compatible with PDF 1.3
Comment	Makes the current PDF compatible with PDF 1.3 and saves it as a PDF 1.3 document. Among other things flattens layers and transparency and reduces image bit depth from 16 to 8 bit if necessary.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.3.kfpx
Name	Compatible with PDF 1.4
Comment	Makes the current PDF compatible with PDF 1.4 and saves it as a PDF 1.4 document. Among other things flattens layers and reduces image bit depth from 16 to 8 bit if necessary.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.4.kfpx
Name	Compatible with PDF 1.5
Comment	Makes the current PDF compatible with PDF 1.5 and saves it as a PDF 1.5 document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.5.kfpx
Name	Compatible with PDF 1.6
Comment	Makes the current PDF compatible with PDF 1.6 and saves it as a PDF 1.6 document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDF_version_compatibility/Compatible_with_PDF_1.6.kfpx

Group: PDFA compliance

Name	Convert to PDF/A-1a
Comment	Converts the current document to PDF/A-1a.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-1a.kfpx
Name	Convert to PDF/A-1b (without fallback conversion)
Comment	Converts the current document to PDF/A-1b without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-1b_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-1b
Comment	Converts the current document to PDF/A-1b.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-1b.kfpx
Name	Convert to PDF/A-2a
Comment	Converts the current document to PDF/A-2a. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-2a.kfpx
Name	Convert to PDF/A-2b (without fallback conversion)
Comment	Converts the current document to PDF/A-2b without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-2b_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-2b
Comment	Converts the current document to PDF/A-2b. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/

Name	Convert to PDF/A-1a
	Convert_to_PDFA-2b.kfpx
Name	Convert to PDF/A-2u
Comment	Converts the current document to PDF/A-2u. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-2u.kfpx
Name	Convert to PDF/A-3a
Comment	Converts the current document to PDF/A-3a. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-3a.kfpx
Name	Convert to PDF/A-3b (without fallback conversion)
Comment	Converts the current document to PDF/A-3b without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-3b_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-3b
Comment	Converts the current document to PDF/A-3b. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-3b.kfpx
Name	Convert to PDF/A-3u
Comment	Converts the current document to PDF/A-3u. Layers, transparency and embedded files are preserved.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-3u.kfpx
Name	Remove PDF/A information
Comment	PDF/A related entries in the PDF metadata will be removed so that the file will not be identified as a PDF/A file afterwards. However, no other modifications are being

Name	Convert to PDF/A-1a
	made so that the PDF may easily be converted to PDF/A again.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Remove_PDFA_information.kfpx
Name	Verify compliance with PDF/A-1a
Comment	Verifies compliance with PDF/A-1a for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-1a.kfpx
Name	Verify compliance with PDF/A-1b
Comment	Verifies compliance with PDF/A-1b for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-1b.kfpx
Name	Verify compliance with PDF/A-2a
Comment	Verifies compliance with PDF/A-2a for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-2a.kfpx
Name	Verify compliance with PDF/A-2b
Comment	Verifies compliance with PDF/A-2b for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-2b.kfpx
Name	Verify compliance with PDF/A-2u
Comment	Verifies compliance with PDF/A-2u for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-2u.kfpx
Name	Verify compliance with PDF/A-3a
Comment	Verifies compliance with PDF/A-3a for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-3a.kfpx

Name	Convert to PDF/A-1a
Name	Verify compliance with PDF/A-3b
Comment	Verifies compliance with PDF/A-3b for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-3b.kfpx
Name	Verify compliance with PDF/A-3u
Comment	Verifies compliance with PDF/A-3u for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-3u.kfpx
Name	Convert to PDF/A-4 (without fallback conversion)
Comment	Converts the current document to PDF/A-4 without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-4_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-4e (without fallback conversion)
Comment	Converts the current document to PDF/A-4e without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-4e_(without_fallback_conversion).kfpx
Name	Convert to PDF/A-4f (without fallback conversion)
Comment	Converts the current document to PDF/A-4f without using any of the fallback options, which are normally applied when regular conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Convert_to_PDFA-4f_(without_fallback_conversion).kfpx
Name	PDF/A-2u: Outline non-Unicode text if it is below specified share of all text
Comment	Converts all text without Unicode representation into outlines if the share of such text is below a specified share (in percentage) of all text. Some PDF creators create PDF where certain characters (such as bullet points) do not have Unicode representation. This Process Plan can be used to create files that formally have full Unicode representation.

Name	Convert to PDF/A-1a
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/PDFA-2u_-_Outline_non-Unicode_text.kfpx
Name	Verify compliance with PDF/A-4
Comment	Verifies compliance with PDF/A-4 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-4.kfpx
Name	Verify compliance with PDF/A-4e
Comment	Verifies compliance with PDF/A-4e for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-4e.kfpx
Name	Verify compliance with PDF/A-4f
Comment	Verifies compliance with PDF/A-4f for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_PDFA-4f.kfpx
Name	Verify compliance with ZUGFeRD
Comment	ZUGFeRD is a German standard for electronic invoices. A ZUGFeRD invoice is a PDF/A-3 file with an embedded XML structure that allows for automatic processing of the invoice.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFA_compliance/Verify_compliance_with_ZUGFeRD.kfpx

Group: PDFE compliance

Name	Convert to PDF/E-1
Comment	Converts the current document to PDF/E-1.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFE_compliance/Convert_to_PDFE-1.kfpx
Name	Remove PDF/E information

Name	Convert to PDF/E-1
Comment	PDF/E related entries in the PDF metadata will be removed so that the file will not be identified as a PDF/E file afterwards. However, no other modifications are being made so that the PDF may easily be converted to PDF/E again.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFE_compliance/Remove_PDFE_information.kfpx
Name	Verify compliance with PDF/E-1
Comment	Verifies compliance with PDF/E-1 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFE_compliance/Verify_compliance_with_PDFE-1.kfpx

Group: PDFUA compliance

Name	Fix problems in PDF tagging structure
Comment	Fixes potential problems in the structure of tagged PDF documents.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFUA_compliance/Fix_problems_in_PDF_tagging_structure.kfpx
Name	Remove PDF/UA information
Comment	PDF/UA related entries in the PDF metadata will be removed so that the file will not be identified as a PDF/UA file afterwards. However, no other modifications are being made so that the PDF may easily be converted to PDF/UA again.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFUA_compliance/Remove_PDFUA_information.kfpx
Name	Verify compliance with PDF/UA-1 (syntax checks only)
Comment	Verifies compliance with PDF/UA-1 for the current document. PDF/UA validation does always require machine checks as well as semantic checks. This profile will perform all machine checks and allows you to open a window for interactive inspection.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFUA_compliance/Verify_compliance_with_PDFUA-1_(syntax_checks_only).kfpx

Group: PDFVT compliance

Name	Verify compliance with PDF/VT-1
Comment	Verifies compliance with PDF/VT-1 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFVT_compliance/Verify_compliance_with_PDFVT-1.kfpx
Name	Verify compliance with PDF/VT-2
Comment	Verifies compliance with PDF/VT-2 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFVT_compliance/Verify_compliance_with_PDFVT-2.kfpx
Name	Verify compliance with PDF/VT-3
Comment	Verifies compliance with PDF/VT-3 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFVT_compliance/Verify_compliance_with_PDFVT-3.kfpx

Group: PDFX compliance

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
Comment	Converts the current document to PDF/X-1a for Coated GRACoL 2006 as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-1a_(Coated_GRACoL_2006).kfpx
Name	Convert to PDF/X-1a (ISO Coated v2 (ECI))
Comment	Converts the current document to PDF/X-1a for ISO Coated v2 (ECI) as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
	Convert_to_PDFX-1a_(ISO_Coated_v2_(ECI)).kfpx
Name	Convert to PDF/X-1a (Japan Color 2001 Coated)
Comment	Converts the current document to PDF/X-1a for ISO Coated v2 (ECI) as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-1a_(Japan_Color_2001_Coated).kfpx
Name	**Convert to PDF/X-1a (PSO Coated v3 (ECI))**
Comment	Converts the current document to PDF/X-1a for PSO Coated v3 (ECI) as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-1a_(PSO_Coated_v3_(ECI)).kfpx
Name	Convert to PDF/X-1a (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-1a for US Web Coated (SWOP) v2 as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-1a_(US_Web_Coated_(SWOP)_v2).kfpx
Name	**Convert to PDF/X-3 (Coated GRACoL 2006)**
Comment	Converts the current document to PDF/X-3 for Coated GRACoL 2006 as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(Coated_GRACoL_2006).kfpx
Name	**Convert to PDF/X-3 (ISO Coated v2 (ECI))**
Comment	Converts the current document to PDF/X-3 for ISO Coated v2 (ECI) as the in-

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
	tended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to PDF/X-3 (Japan Color 2001 Coated)**
Comment	Converts the current document to PDF/X-3 for Japan Color 2001 Coated as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(Japan_Color_2001_Coated).kfpx
Name	**Convert to PDF/X-3 (PSO Coated v3 (ECI)**
Comment	Converts the current document to PDF/X-3 for PSO Coated v3 (ECI) as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(PSO_Coated_v3_(ECI).kfpx
Name	Convert to PDF/X-3 (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-3 for US Web Coated (SWOP) v2 as the intended printing condition. Converts calibrated colors and device dependent RGB to CMYK where necessary, while maintaining spot colors. Layers and transparency are flattened if present.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-3_(US_Web_Coated_(SWOP)_v2).kfpx
Name	**Convert to PDF/X-4 (Coated GRACoL 2006)**
Comment	Converts the current document to PDF/X-4 for Coated GRACoL 2006 as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(Coated_GRACoL_2006).kfpx

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
Name	**Convert to PDF/X-4 (ISO Coated v2 (ECI))**
Comment	Converts the current document to PDF/X-4 for ISO Coated v2 (ECI) as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to PDF/X-4 (Japan Color 2001 Coated)**
Comment	Converts the current document to PDF/X-4 for Japan Color 2001 Coated as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(Japan_Color_2001_Coated).kfpx
Name	**Convert to PDF/X-4 (PSO Coated v3 (ECI))**
Comment	Converts the current document to PDF/X-4 for PSO Coated v3 (ECI) as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(PSO_Coated_v3_(ECI)).kfpx
Name	**Convert to PDF/X-4 and create language layer views (ISO Coated v2 (ECI))**
Comment	Converts the current document to PDF/X-4 for ISO Coated v2 (ECI) as the intended printing condition according to the PDF/X-4 standard. Transparency and layers are allowed. The following layer views (OCCDs) are created: Layer name starts with EN or ends with EN; - English layer view; Layer name starts with DE or _DE - German layer view; Layer name starts with FR or ends _FR - French layer view; Layer name starts with IT or ends with _IT - Italian layer view & Layer name starts with ES or ends with _ES - Spanish layer view;
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_with_layer_views_(ISO_Coated_v2_(ECI))).kfpx
Name	Convert to PDF/X-4 (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-4 for US Web Coated (SWOP) v2 as the intended printing condition according to the PDF/X-4 standard. Transparency and layers

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
	are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-4_(US_Web_Coated_(SWOP)_v2).kfpx
Name	**Convert to PDF/X-5n (7C Indigo TAC370 (ColorLogic))**
Comment	Converts the current document to PDF/X-5n for 7C Indigo TAC370 (ColorLogic) as the intended printing condition according to the PDF/X-5n standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-5n_(7C_Indigo_TAC370_(ColorLogic)).kfpx
Name	**Remove PDF/X information**
Comment	PDF/X related entries in the PDF metadata will be removed so that the file will not be identified as a PDF/X file afterwards. However, no other modifications are being made so that the PDF may easily be converted to PDF/X again.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Remove_PDFX_information.kfpx
Name	**Verify compliance with PDF/X-1a**
Comment	Verifies compliance with PDF/X-1a:2003 for the current document. The PDF/X-1a:2003 standard also implies valid PDF/X-1a:2001 files as PDF/X-1a:2003 compliant.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-1a.kfpx
Name	**Verify compliance with PDF/X-3**
Comment	Verifies compliance with PDF/X-3:2003 for the current document. The PDF/X-3:2003 standard also implies valid PDF/X-1a:2001, PDF/X-1a:2003 or PDF/X-3:2002 files as PDF/X-3:2003 compliant.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-3.kfpx
Name	**Verify compliance with PDF/X-4**
Comment	Verifies compliance with PDF/X-4 for the current document.

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-4.kfpx
Name	**Verify compliance with PDF/X-4p**
Comment	Verifies compliance with PDF/X-4p for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-4p.kfpx
Name	**Verify compliance with PDF/X-5g**
Comment	Verifies compliance with PDF/X-5g for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-5g.kfpx
Name	**Verify compliance with PDF/X-5n**
Comment	Verifies compliance with PDF/X-5n for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-5n.kfpx
Name	**Verify compliance with PDF/X-5pg**
Comment	Verifies compliance with PDF/X-5pg for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-5pg.kfpx
Name	**Convert to PDF/X-6 (Coated GRACoL 2006)**
Comment	Converts the current document to PDF/X-6 for Coated GRACoL 2006 as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(Coated_GRACoL_2006).kfpx
Name	**Convert to PDF/X-6 (Coated CRACoL 2013)**
Comment	Converts the current document to PDF/X-6 for Coated GRACoL 2013 as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(Coated_GRACoL_2013).kfpx
Name	Convert to PDF/X-6 (ISO Coated v2 (ECI))
Comment	Converts the current document to PDF/X-6 for ISO Coated v2 (ECI) as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(ISO_Coated_v2_(ECI)).kfpx
Name	**Convert to PDF/X-6 (Japan Color 2001 Coated)**
Comment	Converts the current document to PDF/X-6 for Japan Color 2001 Coated as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(Japan_Color_2001_Coated).kfpx
Name	**Convert to PDF/X-6 (PSO Coated v3 (ECI))**
Comment	Converts the current document to PDF/X-6 for PSO Coated v3 (ECI) as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(PSO_Coated_v3_(ECI)).kfpx
Name	**Convert to PDF/X-6 (US Web Coated (SWOP) v2)**
Comment	Converts the current document to PDF/X-6 for US Web Coated (SWOP) v2 as the intended printing condition according to the PDF/X-6 standard. Transparency and layers are allowed.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Convert_to_PDFX-6_(US_Web_Coated_(SWOP)_v2).kfpx
Name	**Verify compliance with PDF/X-6**
Comment	Verifies compliance with PDF/X-6 for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-6.kfpx

Name	Convert to PDF/X-1a (Coated GRACoL 2006)
Name	**Verify compliance with PDF/X-6n**
Comment	Verifies compliance with PDF/X-6n for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-6n.kfpx
Name	**Verify compliance with PDF/X-6p**
Comment	Verifies compliance with PDF/X-6p for the current document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_compliance/Verify_compliance_with_PDFX-6p.kfpx

Group: PDFX and PDFA

Name	Convert to PDF/X-4 and PDF/A-2 (Coated GRACoL 2006)
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for Coated GRACoL 2006 as the intend dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(Coated_GRACoL_2006).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (Coated GRACoL 2013)
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for Coated GRACoL 2013 as the intend dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(Coated_GRACoL_2013).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (ISO Coated v2 (ECI))
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for ISO Coated v2 (ECI) as the intended dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(ISO_Coated_v2_(ECI)).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (Japan Color 2001 Coated)
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for Japan Color 2001 Coated as the int

Name	Convert to PDF/X-4 and PDF/A-2 (Coated GRACoL 2006)
	condition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(Japan_Color_2001_Coated).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (PSO Coated v3 (ECI))
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for PSO Coated v3 (ECI) as the intende dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(PSO_Coated_v3_(ECI)).kfpx
Name	Convert to PDF/X-4 and PDF/A-2 (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-4 and PDF/A-2 for US Web Coated (SWOP) v2 as the ir ing condition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-4_and_PDFA-2_(US_Web_Coated_(SWOP)_v2).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (Coated GRACoL 2006)
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for Coated GRACoL 2006 as the intend dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(Coated_GRACoL_2006).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (Coated GRACoL 2013)
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for Coated GRACoL 2013 as the intend dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(Coated_GRACoL_2013).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (ISO Coated v2 (ECI))
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for ISO Coated v2 (ECI) as the intended dition.

Name	Convert to PDF/X-4 and PDF/A-2 (Coated GRACoL 2006)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(ISO_Coated_v2_(ECI)).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (Japan Color 2001 Coated)
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for Japan Color 2001 Coated as the int condition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(Japan_Color_2001_Coated).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (PSO Coated v3 (ECI))
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for PSO Coated v3 (ECI) as the intende dition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(PSO_Coated_v3_(ECI)).kfpx
Name	Convert to PDF/X-6 and PDF/A-4 (US Web Coated (SWOP) v2)
Comment	Converts the current document to PDF/X-6 and PDF/A-4 for US Web Coated (SWOP) v2 as the ir ing condition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/PDFX_and_PDFA/Convert_to_PDFX-6_and_PDFA-4_(US_Web_Coated_(SWOP)_v2).kfpx

Group: Packaging

Name	Packaging Flexo (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of the Ghent PDF Workgroup as defined in the 2015 specification'GWG_PackagingFlexo_2015'.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Packaging/Packaging_Flexo_(GWG_2015).kfpx
Name	Packaging Gravure (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of the

Name	Packaging Flexo (GWG 2015)
	Ghent PDF Workgroup as defined in the 2015 specification 'GWG_Packaging-Gravure_2015'.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Packaging/Packaging_Gravure_(GWG_2015).kfpx
Name	Packaging Offset (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of the Ghent PDF Workgroup as defined in the 2015 specification 'GWG_PackagingOffset_2015'.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Packaging/Packaging_Offset_(GWG_2015).kfpx

Group: Place content

Name	Place EAN 13 barcode with specified value
Comment	Places an EAN 13 code at the center of the page. Content is based on the value specified during runtime. The value has to consist of 12 or 13 digits; EAN 13 always uses 13 digits, the last being a checksum which will be added automatically if only 12 digits are specified. The barcode will be placed on a layer.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_EAN_13_barcode_with_specified_value.kfpx
Name	Place QR-Code with specified value
Comment	Places QR-Code at the center of the page. Content is based on the value specified during runtime. The QR-Code will be placed on a layer.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_QR-Code_with_specified_value.kfpx
Name	Place date (YYYY-MM-DD)
Comment	Places the current date into the upper right corner of each page. The default format is YYYY-MM-DD, however, formatting may be adapted by modifing the JavaScript Variable, which already contains code for other formats.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_date_(YYYY-MM-DD).kfpx

Name	Place EAN 13 barcode with specified value
Name	Place document name in gutter
Comment	Adds the name of the current document (minus the extension part) to the gutter of the document. Placement is 3 mm below the center of the trim box.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_document_name_in_gutter.kfpx
Name	Place job ID in gutter
Comment	Adds the job ID to the gutter of the document both in plain text format and as a Code 128 barcode. The default variable of the job ID is defined in the following variable: @var_jobid:'Job ID' 1234567890@.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_job_ID_in_gutter.kfpx
Name	Place page number in left or right corner
Comment	Places the page number at the lower left corner on even pages and at the lower right on odd pages. Each pager number will be followed by a delimiter symbol ' / ' and the total number of pages in the document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_page_number_in_left_or_right_corner.kfpx
Name	Place specified text at the center of each page
Comment	Places a text at the center of the page. The positioned text is based on the value specified during runtime.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Place_specified_text_at_the_center_of_each_page.kfpx
Name	Quick Check example
Comment	Puts file name and file path on the top left of every page in the current PDF (using pink spot colored Courier text). File name and file path are collected using a Quick Check process plan step. Using a "Place text" fixup in a second process plan step, file name and file path are picked up and combined into a suitable text using a JavaScript variable that retrieves the two values from the app.vars JavaScript application object's data structure.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Place_content/Quick_Check_example.kfpx

Group: Preflight Certificate

Name	Verify Preflight Certificate
Comment	Verifies the existence and validity of a Preflight Certificate in the PDF document.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Preflight_Certificate/Verify_Preflight_Certificate.kfpx

Group: Prepress

Name	Bring 100% black text to front (optionally check for visual changes)
Comment	Brings text using 100% black to front; at runtime it is possible to choose whether to do a befor in order to detect pages where bringing the text to the front causes a different appearance of t
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Bring_100%25_black_text_to_front_(optionally_check_for_visual_changes).kfpx
Name	Check and fix bleed
Comment	Analyses bleed and safety zones to determine required bleed. Safety zone starts at 2 mm insid 3;Tolerance: 25 Unit: mm Create bleed: Off; Page type: single; Developed by calibrate (office@c
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Check_and_fix_bleed.kfpx
Name	Magazine Ads (CMYK and RGB) (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of the Ghent P in the 2015 specification "GWG_MagazineAds_2015 CMYK + RGB".
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Magazine_Ads_(CMYK_and_RGB)_(GWG_2015).kfpx
Name	Magazine Ads (CMYK) (GWG 2015)
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification "GWG_MagazineAds_2015 CMYK".
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Magazine_Ads_(CMYK)_(GWG_2015).kfpx

Name	Bring 100% black text to front (optionally check for visual changes)
Name	**Newspaper Ads (CMYK and RGB) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_NewspaperAds_2015 CMYK + RGB”
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Newspaper_Ads_(CMYK_and_RGB)_(GWG_2015).kfpx
Name	**Newspaper Ads (CMYK) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_NewspaperAds_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Newspaper_Ads_(CMYK)_(GWG_2015).kfpx
Name	**Sheetfed offset (CMYK and RGB) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_SheetCmyk_2015 CMYK + RGB”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Sheetfed_offset_(CMYK_and_RGB)_(GWG_2015).kfpx
Name	**Sheetfed offset (CMYK and spot colors) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_SheetSpot_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Sheetfed_offset_(CMYK_and_spot_colors)_(GWG_2015).kfpx
Name	**Sheetfed offset (CMYK) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_SheetCmyk_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Sheetfed_offset_(CMYK)_(GWG_2015).kfpx
Name	**Sheetfed offset (CMYK, RGB and spot colors) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_SheetSpot_2015 CMYK + RGB”.

Name	Bring 100% black text to front (optionally check for visual changes)
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Sheetfed_offset_(CMYK%2C_RGB_and_spot_colors)_(GWG_2015).kfpx
Name	**Web offset (CMYK and RGB) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebCmyk_2015 CMYK + RGB”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK_and_RGB)_(GWG_2015).kfpx
Name	**Web offset (CMYK and RGB, newsprint) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebCmykNews_2015 CMYK + RGB
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK_and_RGB%2C_newsprint)_(GWG_2015).kfpx
Name	**Web offset (CMYK and spot colors) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebSpot_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK_and_spot_colors)_(GWG_2015).kfpx
Name	**Web offset (CMYK) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebCmyk_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK)_(GWG_2015).kfpx
Name	**Web offset (CMYK, RGB and spot colors) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th group as defined in the 2015 specification “GWG_WebSpot_2015 CMYK + RGB”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK%2C_RGB_and_spot_colors)_(GWG_2015).kfpx
Name	**Web offset (CMYK, newsprint) (GWG 2015)**
Comment	Verifies compliance with - and applies fixups to achieve - the recommendations of th

Name	Bring 100% black text to front (optionally check for visual changes)
	group as defined in the 2015 specification “GWG_WebCmykNews_2015 CMYK”.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/Web_offset_(CMYK%2C_newsprint)_(GWG_2015).kfpx
Name	White underlay for print content on transparent foil (with bleed)
Comment	Creates a spot color plate “White underlay” where the tint value varies depending on the ink a transparent foil, print content – plus a small amount of ‘bleed’ around it – will be printed on th areas without print content remain transparent.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Prepress/White_underlay_for_parent_foil_(with_bleed).kfpx

Group: Process plans

Name	Check for PDF/A-2b compliance and convert if not compliant
Comment	Checks whether the PDF is compliant with the PDF/A-2b standard. The PDF is only modified ar report is created if conversion fails.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Check_for_PDFA-2b_compliance_and_convert_if_not_compliant.kfpx
Name	Check for non-CMYK color, create a report, convert to CMYK (ISO Coated v2) and flatten transp
Comment	Checks whether the PDF uses spot colors or RGB, if this is the case a report with masks is creat jects can be indentified.
All non-CMYK colors will be converted into CMYK (ISO Coated v2) afterwards.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Check_for_non-CMYK_color%2C_create_a_report%2C_convert_to_CMYK_and_flatten_transparency.kfpx
Name	Convert pages with huge numbers of objects into CMYK images
Comment	Pages are analyzed whether limits for total number of page objects are exceeded in which cas age to reduce processing time when output.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Convert_pages_with_huge_numbers_of_objects_into_CMYK_images.kfpx
Name	Create DPart record information from headings

Name	Check for PDF/A-2b compliance and convert if not compliant
Comment	Identifies text in a specified size and adds a record start at pages with matching text in the DPa take advantage of such structures for caching and can significantly reduce processing time.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Create_DPart_record_information_from_headings.kfpx
Name	Convert to CMYK and spot colors and generate images of pre-separated pages
Comment	Converts the document to CMYK + spot colors and generates images of pre-separated pages (J
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Convert_to_CMYK_and_spot_colors_and_generate_images_of_pre-separated_pages.kfpx
Name	Convert to PDF/A-2b (any fallback conversions use the original file)
Comment	When regular PDF/A-2b conversion fails the original is attempted to convert by regenerating it by rasterizing all pages.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Convert_to_PDFA-2b_(any_fallback_conversions_use_the_original_file).kfpx
Name	Generate 3 mm bleed (pixel repetition on pages with text objects close to Trimbox, else mirror
Comment	Determines whether a page has text objects close to Trimbox in which case for that page bleed otherwise mirroring is used. The process plan uses a check that creates an array with page nu using pixel repetition.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Generate_3_mm_bleed_(pixel_repetition_or_mirroring_depending_on_content).kfpx
Name	Prepare PDF for specified page size
Comment	Checks whether the TrimBox size of the pages matches the defined size and uses several appr not the case. This Process Plan will only work for PDF files where all pages have the same size.
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Prepare_PDF_for_specified_page_size.kfpx
Name	Split PDF into half and impose as a brochure
Comment	Splits all pages in the middle (respecting single and double pages) and imposes the result for
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Process_plans/Split_PDF_into_half_and_impose_as_a_brochure.kfpx

Group: Processing Steps

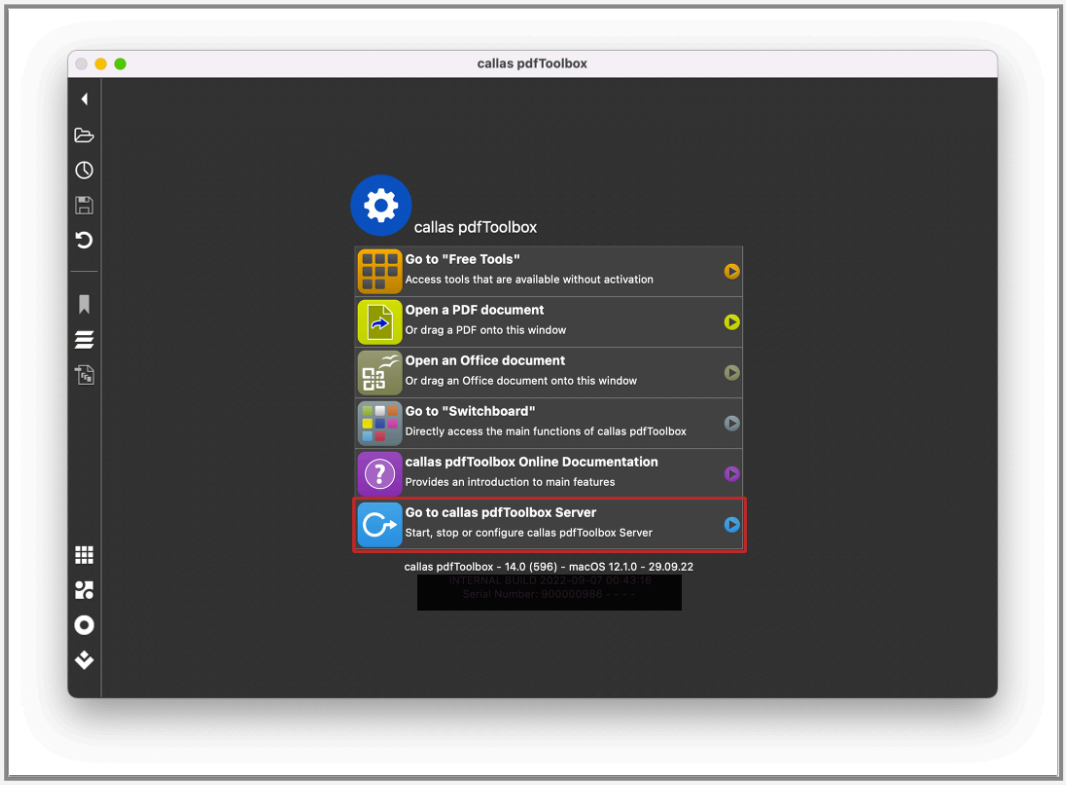
Name	List Processing Steps metadata information
Comment	List Processing Steps metadata information
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Processing_steps/List_Processing_Steps_metadata_information.kfpx
Name	Verify compliance with Processing Steps (ISO 19593-1) for packaging and label
Comment	Verifies compliance with ISO 19593-1, Processing Steps for print files for packagings and labels
URL	https://s3.eu-central-1.amazonaws.com/ccapi-sample/Profiles/Processing_steps/Verify_compliance_with_Processing_Steps_(ISO_19593-1)_for_packaging_and_label.kfpx

35. Server

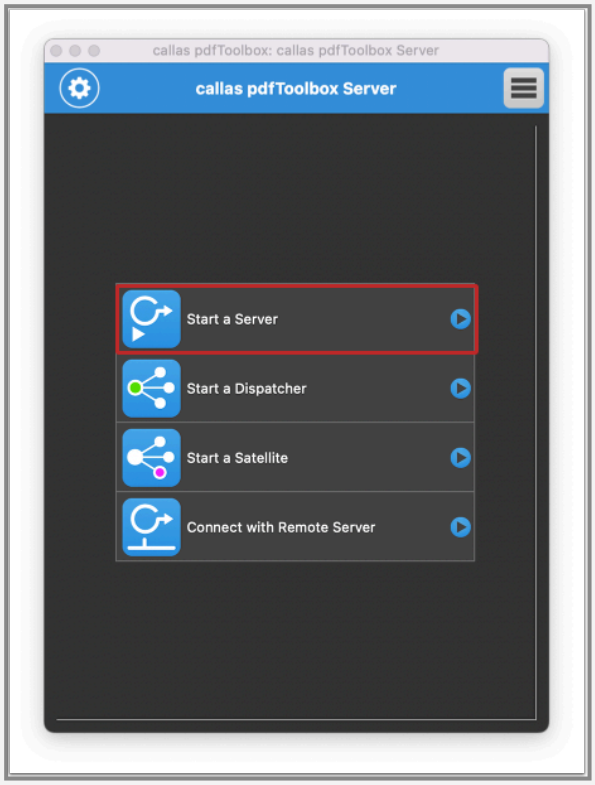
35.1 Introduction to pdfToolbox Server

pdfToolbox Server combines the user interface of pdfToolbox Desktop Version in interaction with the CLI component, which is automatically called in the background as soon as you start pdfToolbox Server.

Starting pdfToolbox Server



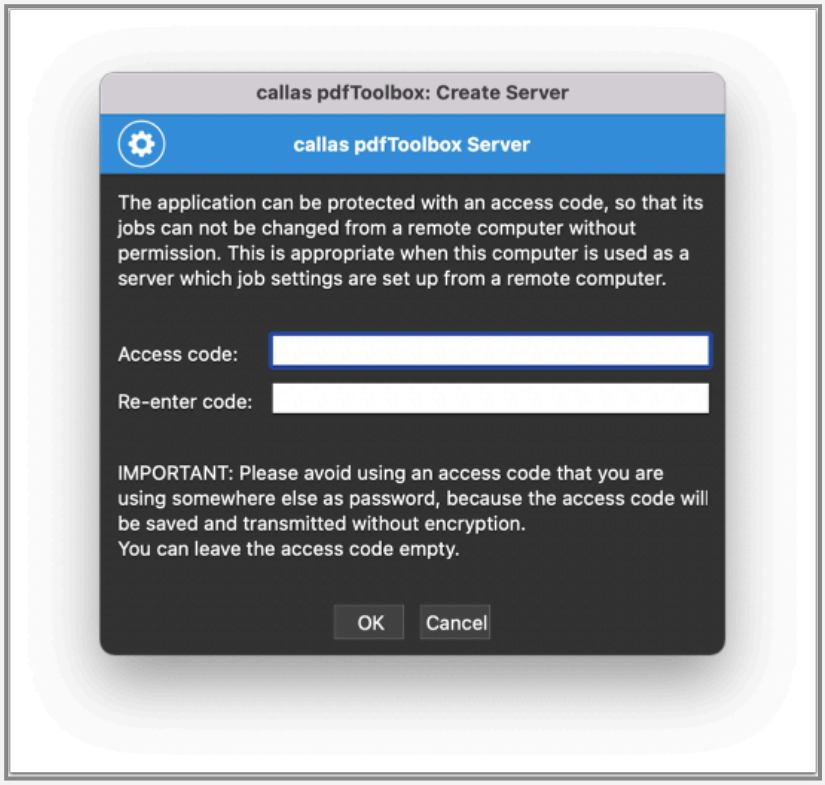
Start a Server



By clicking this button the CLI is started in the background (similar to the manual call with the `--server` command).

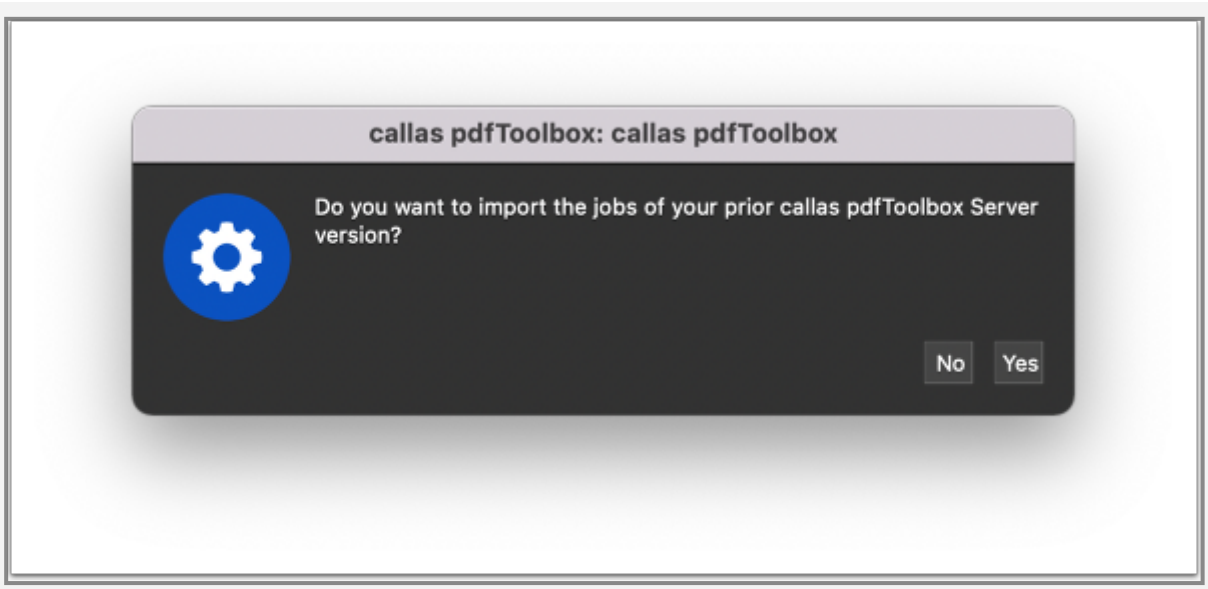
As soon as the process is running, jobs can already be processed.

Set access code



To protect the server and the server jobs from unauthorized access and changes, an access code can be assigned.

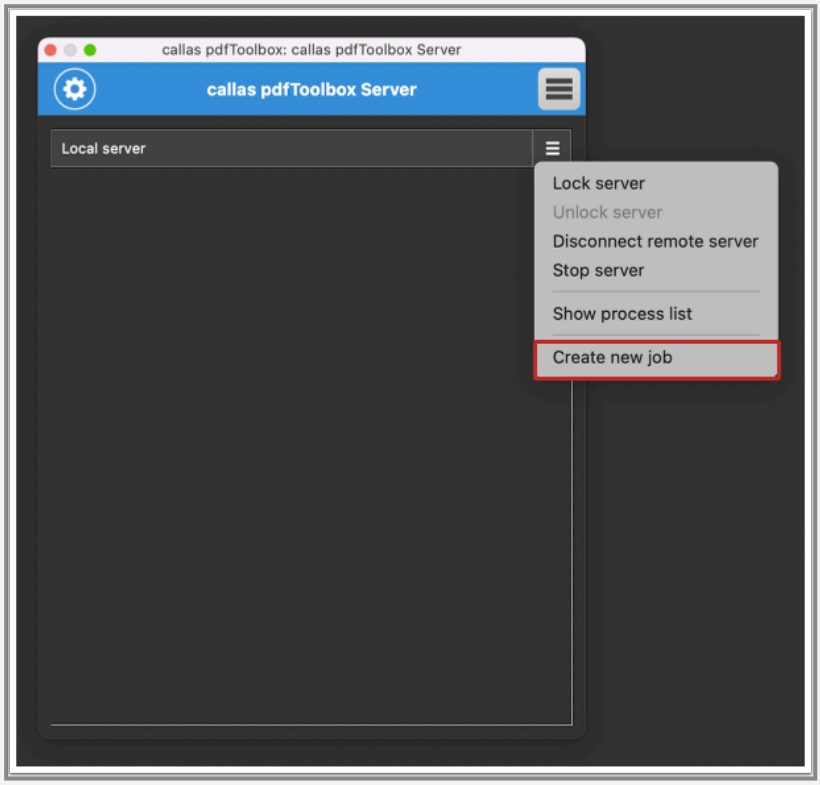
Automatic import of server jobs from previous version



If the program finds server settings and server jobs on the system from a previous version, these can be automatically imported into the new version.

Click "Yes" accordingly, otherwise no jobs will be imported if you click "No".

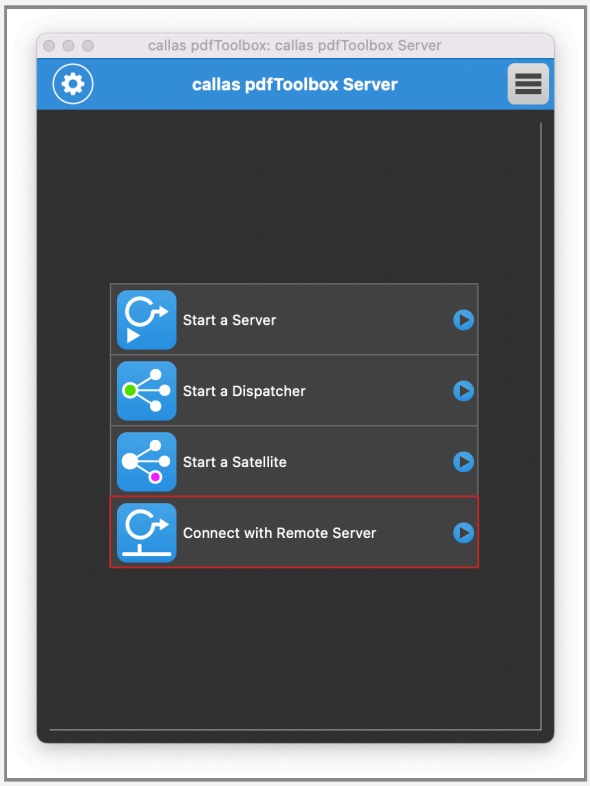
Create a new Server job



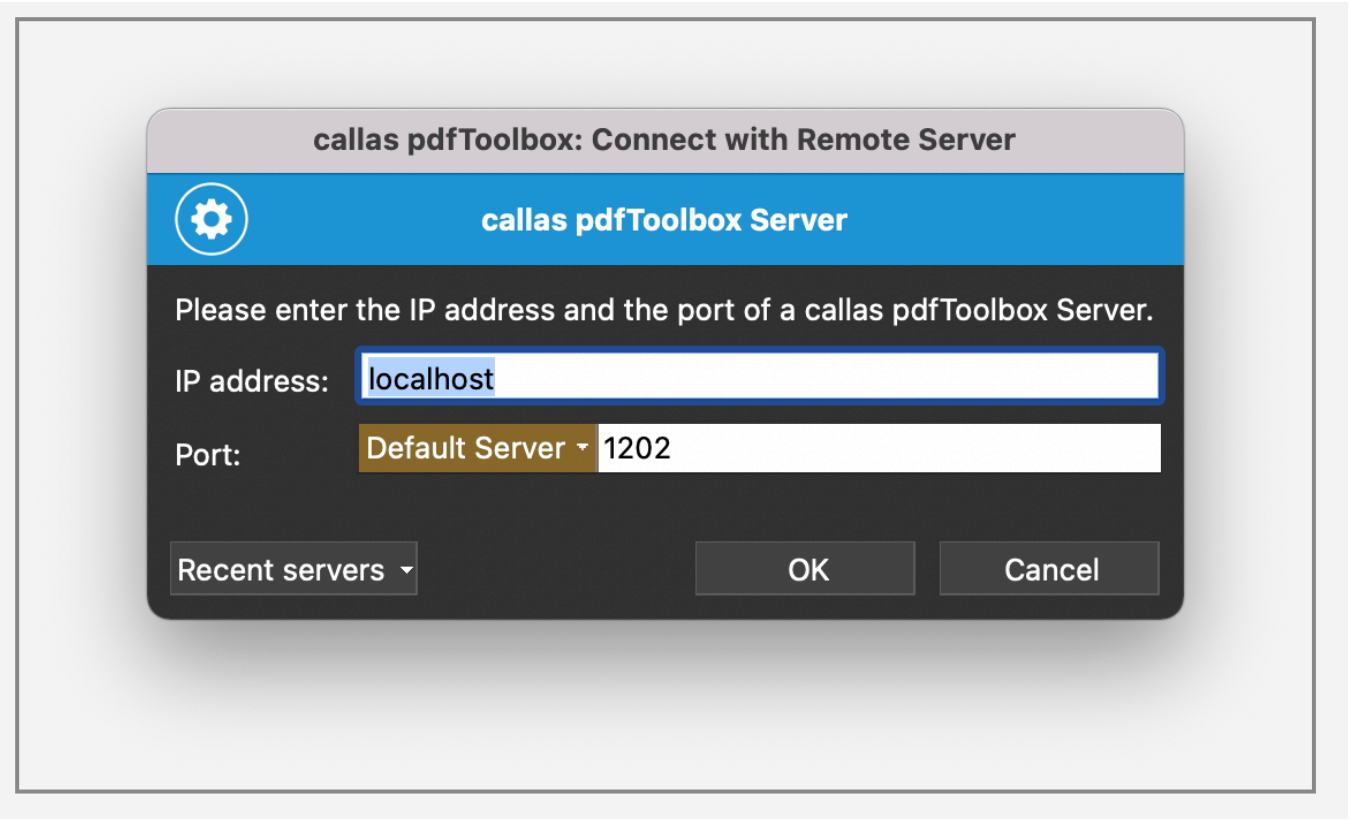
Once the server is running, a server job can be created.

35.2 Connect with Remote Server

If the server on which the processing is to take place is located on another computer, you can also connect to the system remotely.



IP and Port

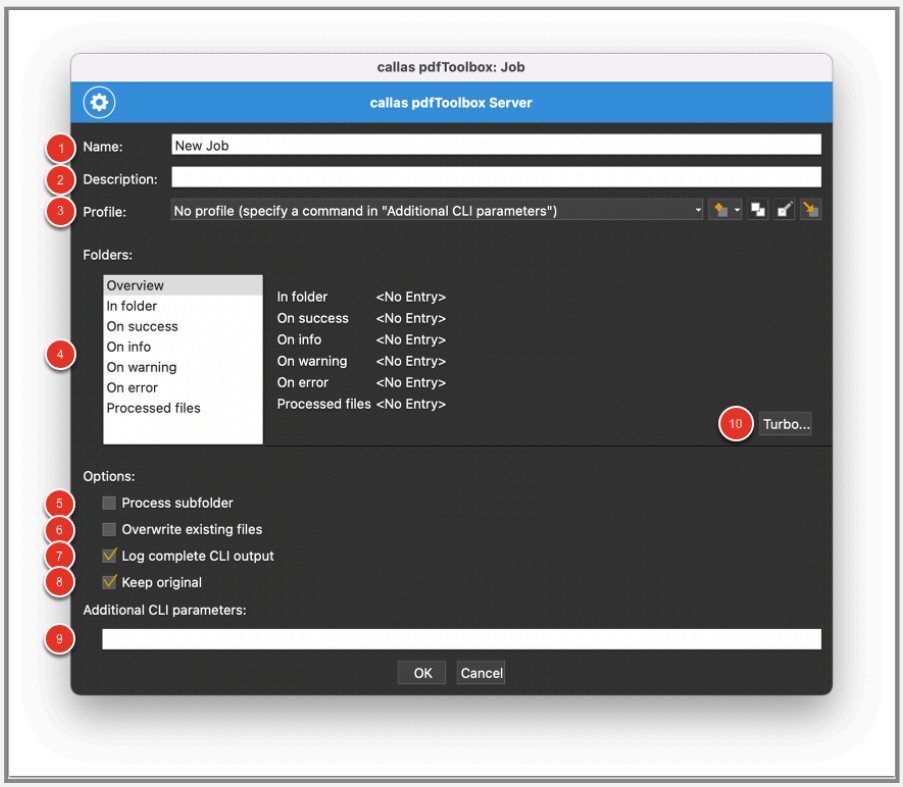


To connect to the target system the IP and the respective port must be specified.

Please note: If a firewall is used, the port must also be released accordingly.

35.3 Job settings

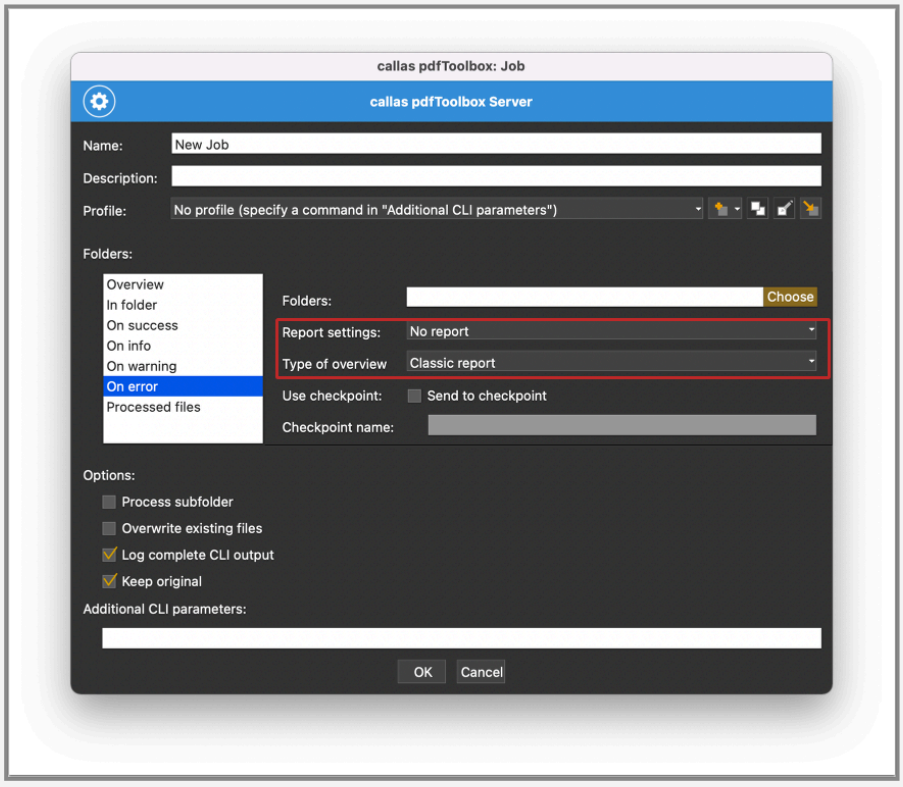
The server job has a variety of setting options, which are described below.



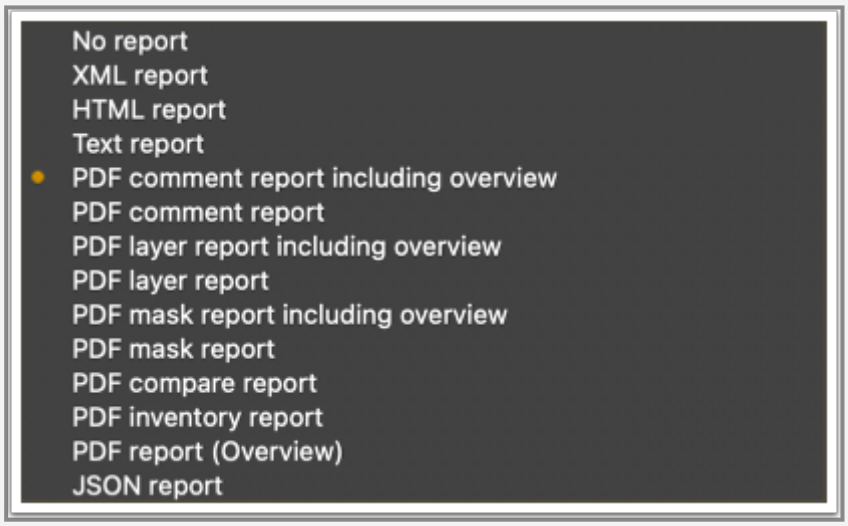
1. Name: Here you can specify a name for the job, this will then be displayed in the list of jobs.
2. Description: In addition to the name, an additional description can also be stored.
3. Profile: Existing profiles from the currently active library can be selected, new created, duplicated, exported, as well as imported from an external path.
4. Hotfolder: The Hotfolders are folders over which files are imported and accordingly output after processing, depending on the result of a profile or an action.
5. Process subfolder: By activating this option, nested directories can also be processed, otherwise only files located directly in the folder are processed.
6. Overwrite existing files: In the output folders (Success, Info, Warning, Error, Processed) new files are indexed with _0001, _0002 ... and stored next to them. If this option is activated, existing files with the same name are always overwritten.

7. Log complete CLI output: Under "Processed Files", a log file with the CLI output in text form is also created for each processed file.
8. Keep original: The original input file will be stored under "Processed files" as well.
9. Additional CLI parameters: As an alternative to selecting a profile, commands for the CLI can be specified here as well as other parameters that exist for a CLI command. Similar to the command call (via the command prompt/ terminal), the parameters must always be separated by a space.
10. Turbo button: When clicking this button, only one path must be specified. Within this folder, the subfolders (In-box, Success, Info, Warning, Error, Processed) are automatically created at the same time.

Create reports



Overview of possible report types



Use "non-PDF files" as input files

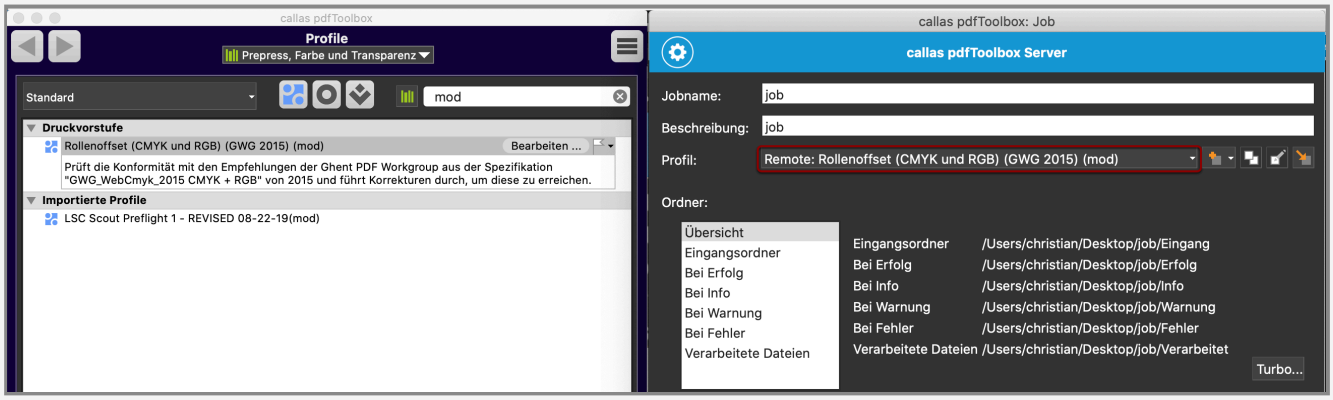
If files, that are not already a PDF file, are to be processed as input files (such as images or even Office files), the program tries to convert them automatically, if the format is supported.

Please also note the following article: Requirements for conversion to PDF

What does "Profile: Remote: ..." mean?

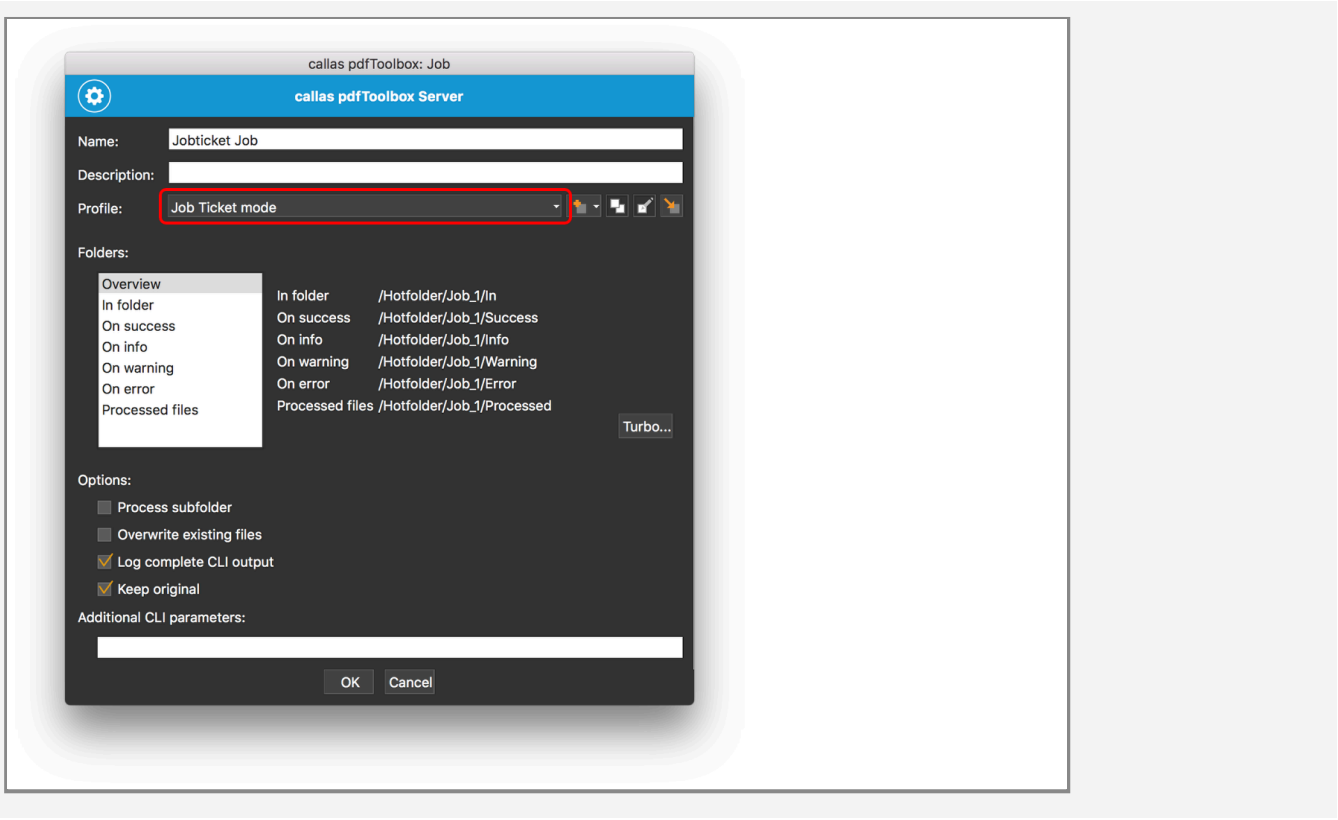
When editing a server job, you may have noticed that the addition "Remote:" was displayed under Profile. This indicates that the profile in the server settings is already different from the one under Profile and has therefore been changed outside the server settings.

The solution is to re-import or re-assign the profile.


callas pdfToolbox
Profile
Prepress, Farbe und Transparenz
Standard
mod
Druckvorstufe
Rollenoffset (CMYK und RGB) (GWG 2015) (mod)
Bearbeiten ...
Prüft die Konformität mit den Empfehlungen der Ghent PDF Workgroup aus der Spezifikation "GWG_WebCmyk_2015 CMYK + RGB" von 2015 und führt Korrekturen durch, um diese zu erreichen.
Importierte Profile
LSC Scout Preflight 1 - REVISED 08-22-19(mod)
callas pdfToolbox: Job
callas pdfToolbox Server
Jobname:
job
Beschreibung:
job
Profil:
Remote: Rollenoffset (CMYK und RGB) (GWG 2015) (mod)
Ordner:
Übersicht
Eingangsordner
Bei Erfolg
Bei Info
Bei Warnung
Bei Fehler
Verarbeitete Dateien
Eingangsordner /Users/christian/Desktop/job/Eingang
Bei Erfolg /Users/christian/Desktop/job/Erfolg
Bei Info /Users/christian/Desktop/job/Info
Bei Warnung /Users/christian/Desktop/job/Warnung
Bei Fehler /Users/christian/Desktop/job/Fehler
Verarbeitete Dateien /Users/christian/Desktop/job/Verarbeitet
Turbo...

35.4 Using Job Tickets

Usually, a Job of a Server needs incoming files in the specified "In" folder to start processing as defined in the respective Job settings.
The "Job Ticket mode" makes it possible to put just a ".jobticket"-file into the input hotfolder of the Server (instead of a PDF), which contains all necessary information about the file to be processed as well as optionally additional parameters like e.g. a Profile, additional CLI parameters or variables and their values.



When setting up a Job, just select "Job Ticket mode" instead of a Profil.
Using this mode, the Server will only process files with the extension ".jobticket" in the input folder of this Job. All other files will remain in the input folder and will not be processed.

The ".jobticket" file has to be formatted in JSON, containing all needed information for processing a file.

Job Ticket for Profile-based processing

As already mentioned, instead of placing an input file, which will be processed based on the fixed settings of a Job, the way a file is processed in a "Job Ticket"-Job is completely flexible - and there is no need to place the input file into the input folder as well.
All required components (like the input file, the Profile and maybe ressources referenced by Variables) can remain in their original location and have to be just referenced in the Job Ticket.

```
{
        "type": "jobticket",
                "params":
                [
                        "/Settings/Profiles/Place text.kfpx",
                        "/volumes/Production/Incoming/0815/DummyPDF.pdf",
                        "--setvariable=placetext:Text for DummyPDF",
                        "--report=xml",
                        "--report=template=overview,path=/0815/report"
                ]
}
```

Click to copy
Each Job Ticket has to start with the "type", which must be "jobticket".
Within the "params" section, almost all CLI parameters of pdfToolbox can be used.

Minimum content:

- As the Job itself does not define a Profile, a valid path to a Profile must be defined.
- Path to the file to be processed

All other parameters are optional.

Reports have to be defined in the .jobticket file, as they can not become set in the Job.
By default, the report will be saved into the folder of the respective severity of each process (Error, Warning, Info or Success).

However, the path to a report can also be defined in the .jobticket-file.

Job Ticket for Actions

When an Action (e.g. Save as image, ReDistill, Impose) shall be performed, the syntax of the .jobticket-file is quite similar. Instead of defining a path to a Profile, the respective Action command and the needed options have to be defined:

```
{
         "type": "jobticket",
        "params":
        [
                "--saveasimg",
                "E:\\Job2\\in\\Testfile.pdf",
                "--imgformat=jpeg",
                "--colorspace=CMYK",
                "--resolution=150"
        ]
}
```

Click to copy

```
{
        "type": "jobticket",
        "params":
        [
                "--mergepdf",
                "E:/Job_4711/in/file_001.pdf",
                "E:/Job_4711/in/file_002.pdf",
                "E:/Job_4711/in/file_003.pdf",
                "E:/Job_4711/in/file_004.pdf"
        ]
}
```

Click to copy

How to test a Job Tickets

To test the behavior of a .jobticket-file, they can be used with pdfToolbox on the command line:

```
pdfToolbox --jobticket <input.jobticket>
```

Syntax issues to be considered

Be aware of JSON-related typological issue and other implementation-based issues:

- Don't use (“) - but use (")
- Escape (\) on Windows → (\\)
  or use a normal slash (/) instead
- Escape Quotes (") → (\") when used as string
- Spaces ( ) must not be escaped

Limitations

The following output options are not supported:

- Outputfile
- Outputfolder
- Overwrite

Other currently not supported options:

- Cachefolder
- All kinds of distributed processing options
- Timeout settings



Everything from 'How to' to 'Do's and Dont's' of Job tickets in this video:

35.5 Sidecar files and their use when processing files with a Server-Job

With sidecar files, it is possible to add a file to the input hotfolder (that has the input PDF) which contains additional information in the form of Variables. This information can be used and modified during the hotfolder processing. The new (updated) values can be used to influence processing in later hotfolder steps.

Using Sidecar files

- When sidecar files shall be used, no new entry in the configuration of a Job is necessary - if a *.sidecar-file exists beside a PDF, their content is just used for processing.
- Values of the "variables" in the .sidecar-file are used as Variables for every Job.
- To use new values (e.g. calculated or set in previous hotfolder jobs) for Variables, they have to become grabbed up from the results area and it must be handled in the JavaScript Variable in that way, that their value overwrite the existing Variables during runtime.
- A .sidecar-file can not contain additional CLI commands or options.
- A .sidecar-file should exist at least simultaneously to the input file inside the input folder. It is recommended to copy the .sidecar-file before the input file to ensure the Server is aware of the sidecar file when scanning the input folder.

Syntax of Sidecar files

As already mentioned, optionally a .sidecar-file can be copied into the In folder beside the input file. The content of the .sidecar-file will be taken into account for processing. But the general parameters from the Job settings will control the way a file is handled.

The ".sidecar" file has to be formatted in JSON, containing all additional information for processing a file.

The .sidecar-file must have the exact, case-sensitive same name as the original file, e.g.:

- Inputfile.pdf
- Inputfile.pdf.sidecar

```
{
   "type":"sidecar",
   "variables":
   [
      {
         "key":"placetext",
         "value":"Some text from the initial sidecar file"
      },
      {
         "key":"activate_fixup",
         "1"
      }
   ]
}
```

Click to copy

Each Job Ticket has to start with the "type", which must be "sidecar".

Within the "variables" section new values for Variables can be defined, which will then be used for the specific file.

Updated content after a Job

The normal "variables" part will not be updated, if a Variable has changed during processing.
However, all changed Variable as well as details to every performed processing step is added to the .sidecar-file as shown in the sample below:

```
{
        "results" :
        [

                {
                        "command" : "< complete CLI command used >",
                        "folder_in" : "/Hotfolder/Job_2/In",
                        "jobname" : "Sidecar Job",
```

```
                        "returncode" : 5,
                        "variables" :
                        {
                                "placetext" : "Some text from the initial sidecar
file",
                                "placetext_new" : "Some text from the initial side-
car file and some additional text",
                                "script" : "Some text from the initial sidecar
file and some additional text"
                        }
                }
        ],
        "type" : "sidecar",
        "variables" :
        [

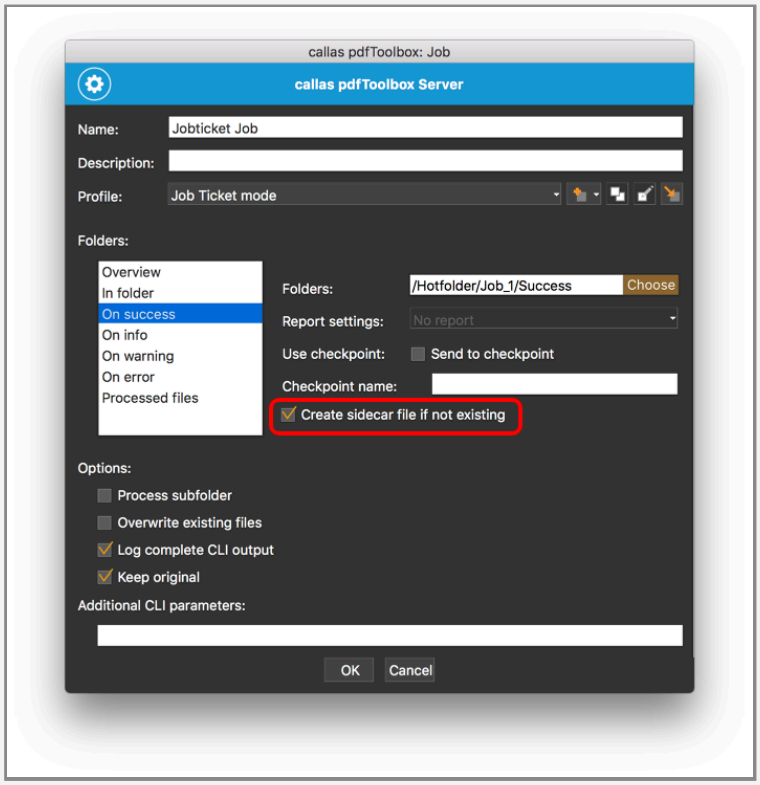
                {
                        "key" : "placetext",
                        "value" : "Some text from the initial sidecar file"
                },
      {
         "key":"activate_fixup",
         "1"
      }
        ]
}
```

Click to copy

In this sample, a new Variable "placetext_new" is created by a JavaScript-Variable in the Profile. It uses the supplied, new "placetext"-Variable and just adds some additional text.

Create Sidecar files from a Server-Job

It is possible to create a .sidecar-file if not existing by activating the respective option within the four settings for each severity. Creating such a .sidecar-file enables a pick-up of results by following steps.



Download of the used sample Profile

Place_text_and_add_some_text_to_a_new_Variabl.kfpx

Everything from 'How to' to 'Do's and Dont's' of Sidecar files in this video:

35.6 How to upgrade pdfToolbox Server successfully (without losing server jobs)?

Sometimes after upgrading to a newer version of callas pdfToolbox Server, the server jobs are not moved automatically. In that case, you can:

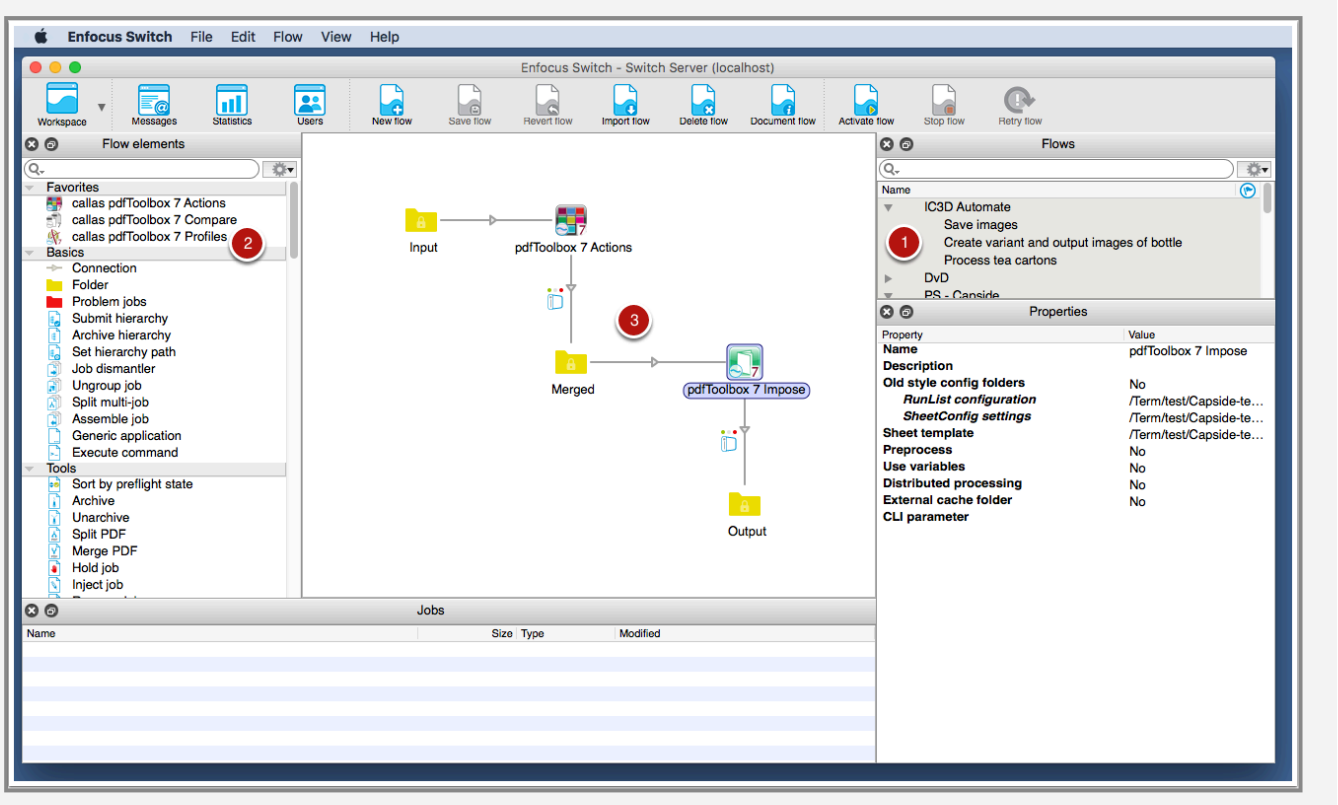
1. Go to the "Server" folder of the previous version of callas pdfToolbox Server:
   - MacOS:
     /Library/Application Support/callas software/callas pdfToolbox CLI (version)/Server/Jobs
   - Windows:
     C:\ProgramData\Application Support\callas software\callas pdfToolbox CLI (version)\Server\Jobs
2. Create a copy of the "Jobs" folder.
3. Go to the "Server" folder of the latest version of callas pdfToolbox Server.
4. Place the "Jobs" folder into the "Server" folder of the latest version of callas pdfToolbox Server.

35.7 pdfToolbox Server integration in automation systems

Instead of using pdfToolbox Server in hot folder mode, it can also be used integrated in automation systems such as Enfocus Switch. This article provides an introduction to how pdfToolbox can be used in such tools.

Enfocus Switch

Switch from Enfocus is a modular automation solution that can automate manual tasks; it includes both built-in tools (for FTP, renaming, grouping and ungrouping...) and connects to third-party tools. From the start, Switch has had a number of pdfToolbox configurators (the Switch name for the plug-ins that create a connection to a third-party tool).



1. The list of flows (the Switch terminology for a workflow that automates a series of tasks).

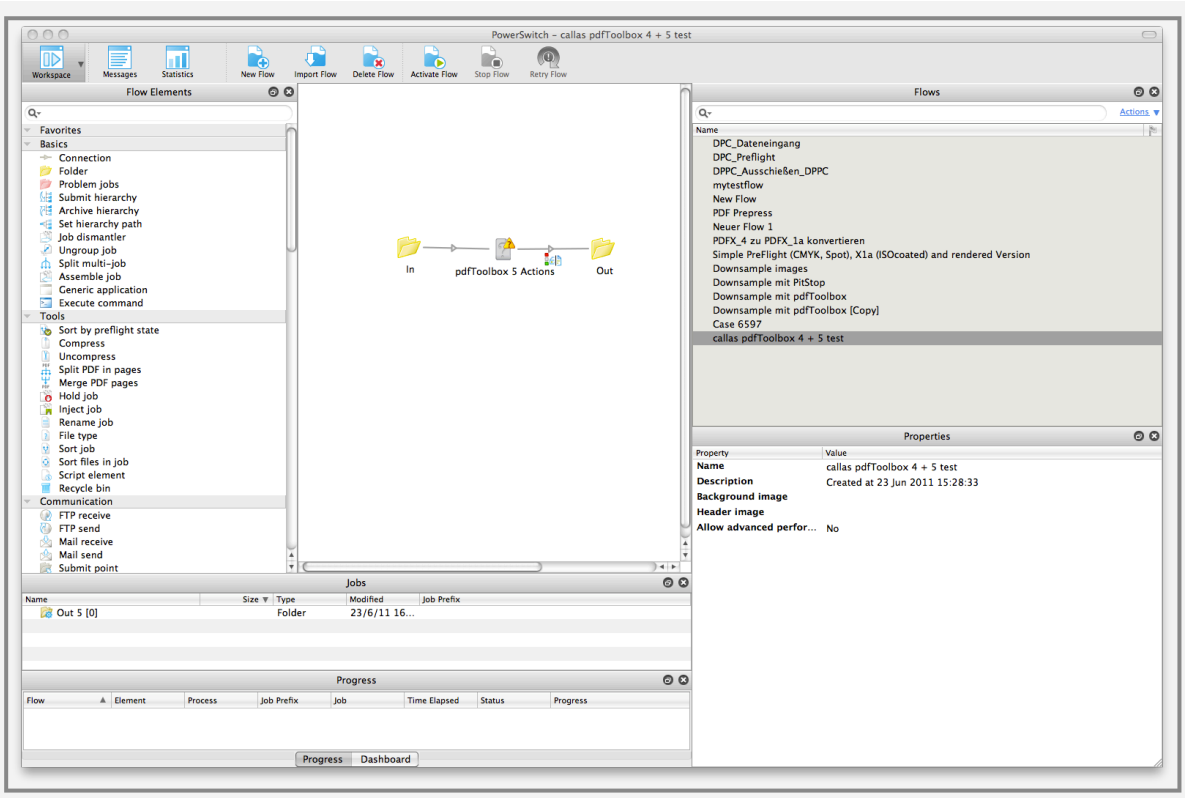
2. The list of built-in tools and configurators; at the top in the favorites section, three of the pdfToolbox configurators.
3. A workflow featuring two pdfToolbox configurators to automatically merge PDF files into a single file and subsequently impose them using the imposition engine built into pdfToolbox.

pdfToolbox provides the following configurators for Switch:

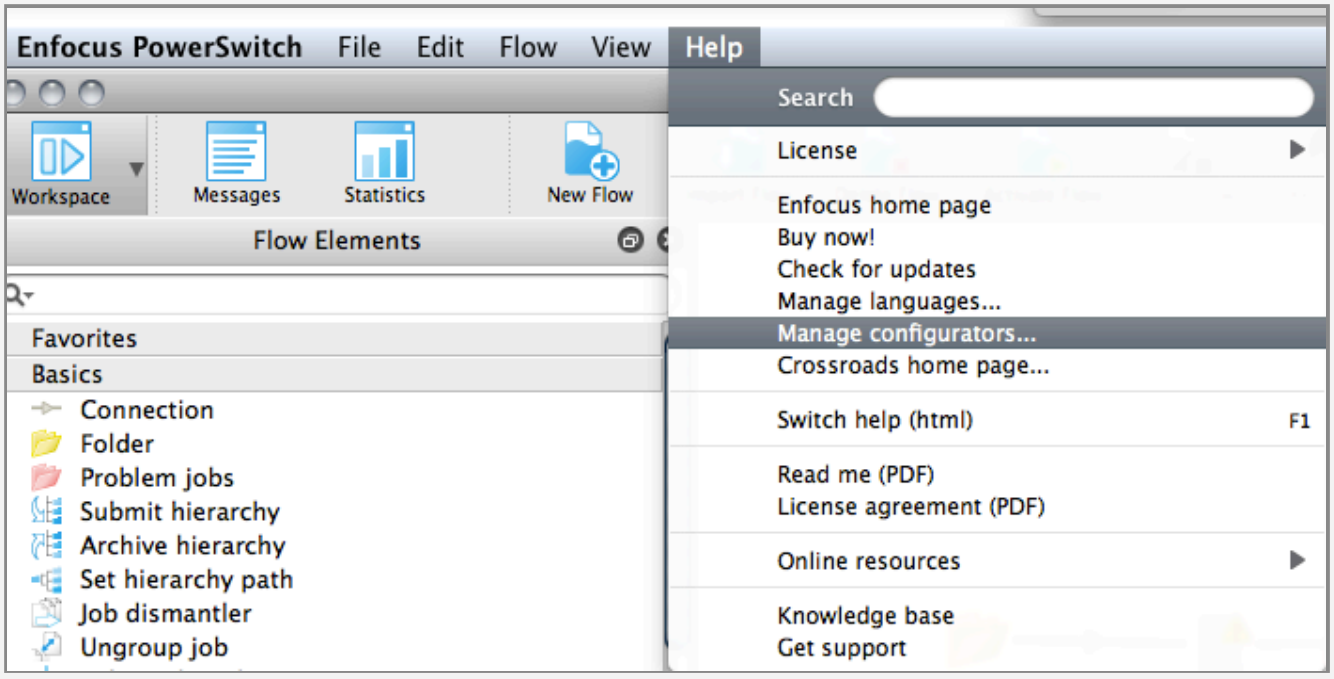
- Actions: runs one of the pdfToolbox actions (roughly equivalent to the functionality available in the Switchboard).
- Profiles: runs a pdfToolbox profile or process plan.
- Compare: compares two incoming PDF files and determines whether they are visually the same or different.
- Impose: runs the pdfToolbox imposition engine.
- ConvertColors: runs a color conversion using pdfToolbox.

Steps to integrate callas pdfToolbox with Enfocus Switch

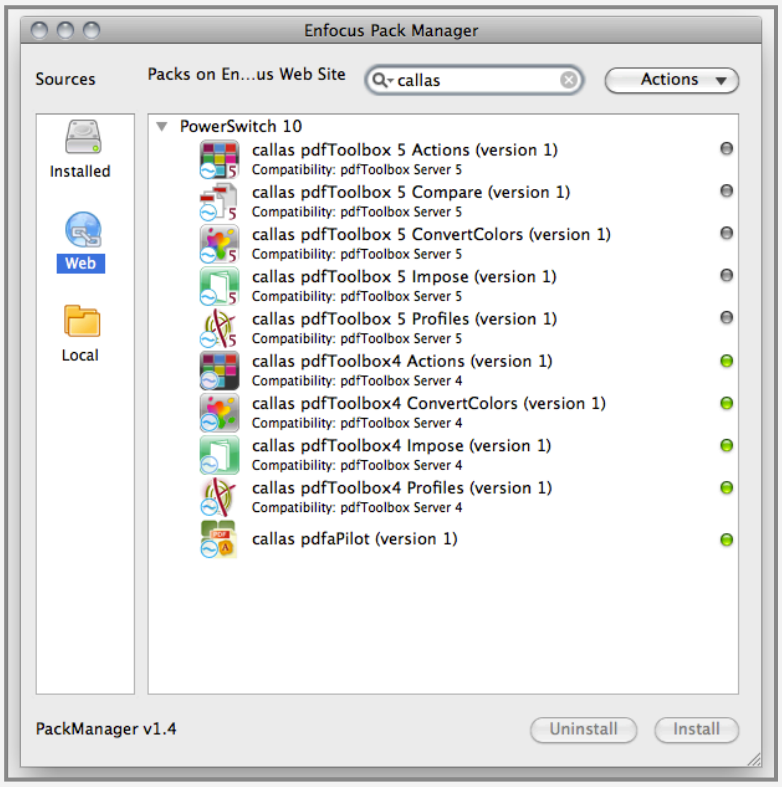
Open Enfocus Switch



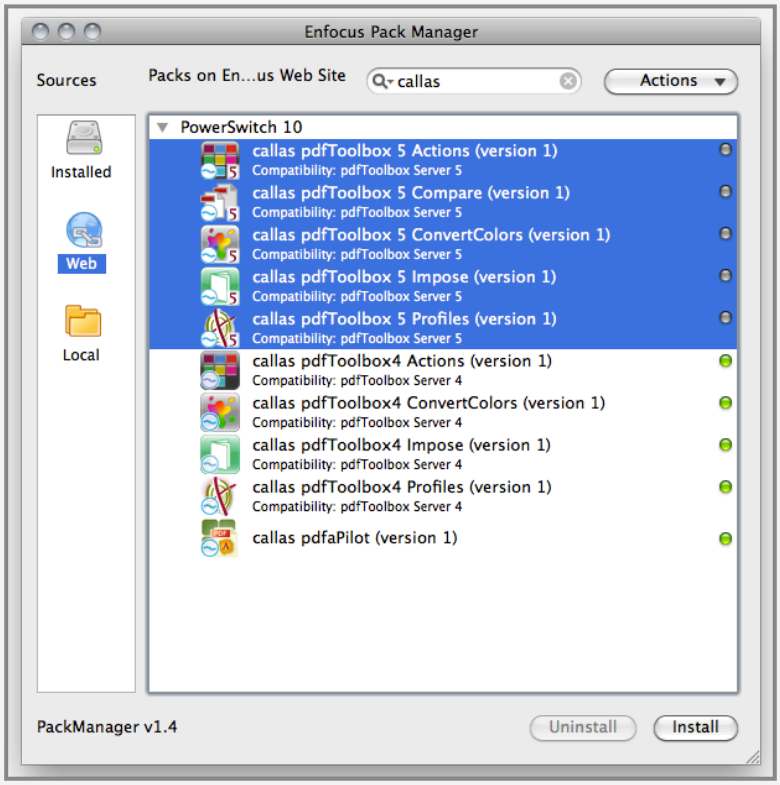
Go to “Help” => Manage configurators…:



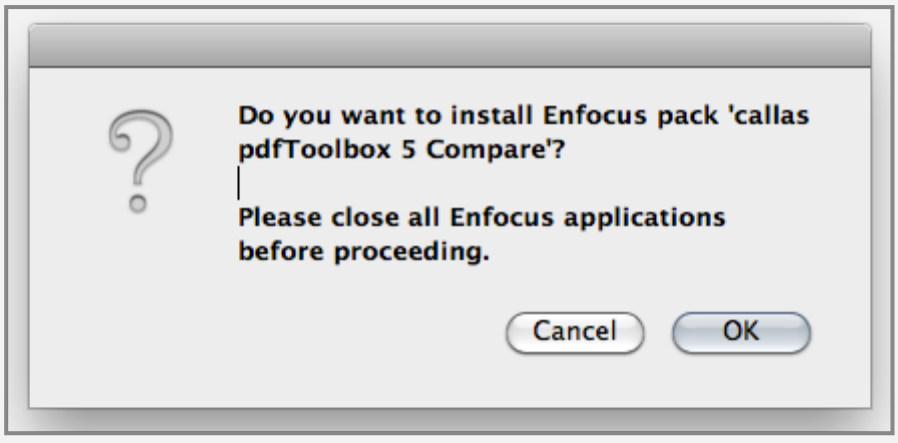
Open “Web” and search for callas:



Select the configurators you want and install them (there will be different versions of the configurators; select the ones for the callas application(s) you are using:



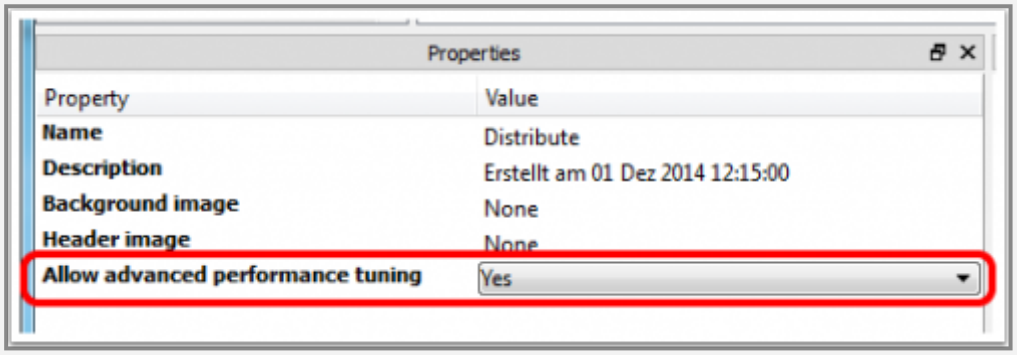
Finally restart Switch:



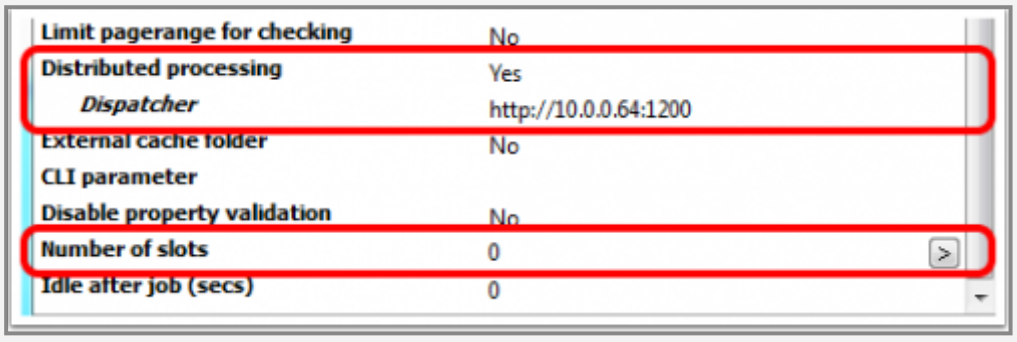
Using the optional extension “Distributed processing” of callas pdfToolbox, jobs can be processed on other machines (Satellites). The distribution of jobs is handled by the Dispatcher. This allows to distribute the load to selected machines.

For an optimal distribution of jobs within Enfocus Switch it has become figured out, that some additional settings should be done. The “Number of slots” in the pdfToolbox Configurator is important.

First “Allow advanced performance tuning” in the properties Flow should be set to “Yes”.



Additionally the “Number of slots” in the pdfToolbox Configurator should be set to “0”. This setting gives the Dispatcher the exklusive control of the distribution ob jobs in the network and the distribution normally done by Enfocus Switch will not take place.



36. Enfocus Switch

36.1 Using pdfToolbox in Enfocus Switch

Enfocus Switch is capable of using pdfToolbox CLI to automate all functionality offered by pdfToolbox. In order to do so, it uses configurators or Switch apps, depending on the version of Switch you are running.



You need to have pdfToolbox CLI installed on the same server as Enfocus Switch in order to use the configurators or Switch apps for pdfToolbox. You do not need to start pdfToolbox CLI in any way: this is done by Enfocus Switch when needed.

Configurators

If your version of Switch still supports configurators, you will see the pdfToolbox configurators in Enfocus Switch as soon as you launch it. You can also search for "pdfToolbox" in the "Flow elements" pane to filter and see only the pdfToolbox configurators.

Switch apps

If your version of Switch doesn't support configurators, you need to add the pdfToolbox apps from the Enfocus Switch App Store. In the App Store, search for "pdfToolbox" to find all pdfToolbox apps. The apps published by callas software are:

- callas pdfToolbox Actions
- callas pdfToolbox Compare
- callas pdfToolbox Profiles
- *callas pdfToolbox ColorConvert*
- *callas pdfToolbox Imposition*

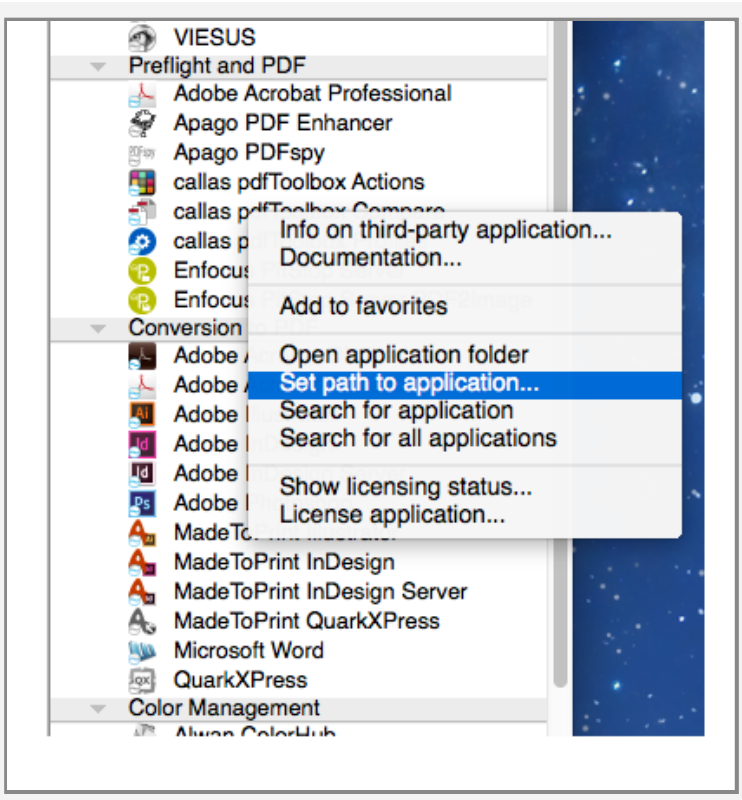
You should no longer use the "Imposition" or "ColorConvert" apps as they have been deprecated. Read the article on deprecated apps to know how to use their functionality in the three core apps. To work with the other apps, click on each

app in turn and use the "Get it now" button to install it on your Switch server. Once you've gone through that step, restart Switch and/or click the refresh button on the "Flow elements" pane. You can now start using the apps in your Enfocus Switch flows.

36.2 Setting the path for the pdfToolbox configurators

When using pdfToolbox in the Enfocus Switch workflow system, pdfToolbox Server or CLI has to be installed on that system. When using pdfToolbox in Switch, Switch automatically finds the path to the pdfToolbox CLI on the system. However, when you install a new version of pdfToolbox, or if you install pdfToolbox CLI only after you were already using Switch, you need to manually set the path to the version of pdfToolbox CLI you want to use with Switch.

In order to adjust the path, you have to select the pdfToolbox Configurator element and click on "Set path to application...". Then, set the path to the CLI version in the file system dialog shown.



Some important remarks with this:

- Make sure you select the path to the pdfToolbox CLI application in the system dialog, not the path to pdfToolbox Desktop. You will not get a warning while setting the path, but your flow will not work correctly. The pdfToolbox CLI

application lives in a folder called "cli" on both Windows and macOS.
- The way this is implemented in Switch means that you can use only one version of pdfToolbox CLI at a time. Changing the path using "Set path to application...", changes it for all configurators in all Switch flows.
- The different pdfToolbox configurators (Actions, Profiles, ColorConvert, Imposition, and Profiles) all use a different path. If you upgraded pdfToolbox CLI and you want all configurators to use the new version, you need to use the "Set path to application..." approach for *each configurator* individually.

What about Profiles used in Switch?

In the Profiles Configurator in Enfocus Switch, there are multiple ways to use a Profile:

- When a Profile is selected from a Library, setting the path to the new pdfToolbox version as shown above will automatically use the predefined Profile from the new installation path. The big advantage in this case is that the Profile which has been updated by callas will be used without any Flow adjustments needed.
- When a Profile is referenced via "Choose file" (or similar via a variable or script expression), the respective path to the ".kfpx"-Profile-file will remain unchanged and the processing will work as before. All pdfToolbox versions are backwards compatible - this means Profiles created by previous versions of the software will remain working. Also all kind of resources (e.g. ICC profiles or HTML-Templates) are contained in the .kfpx file (except when controlled via Variables).

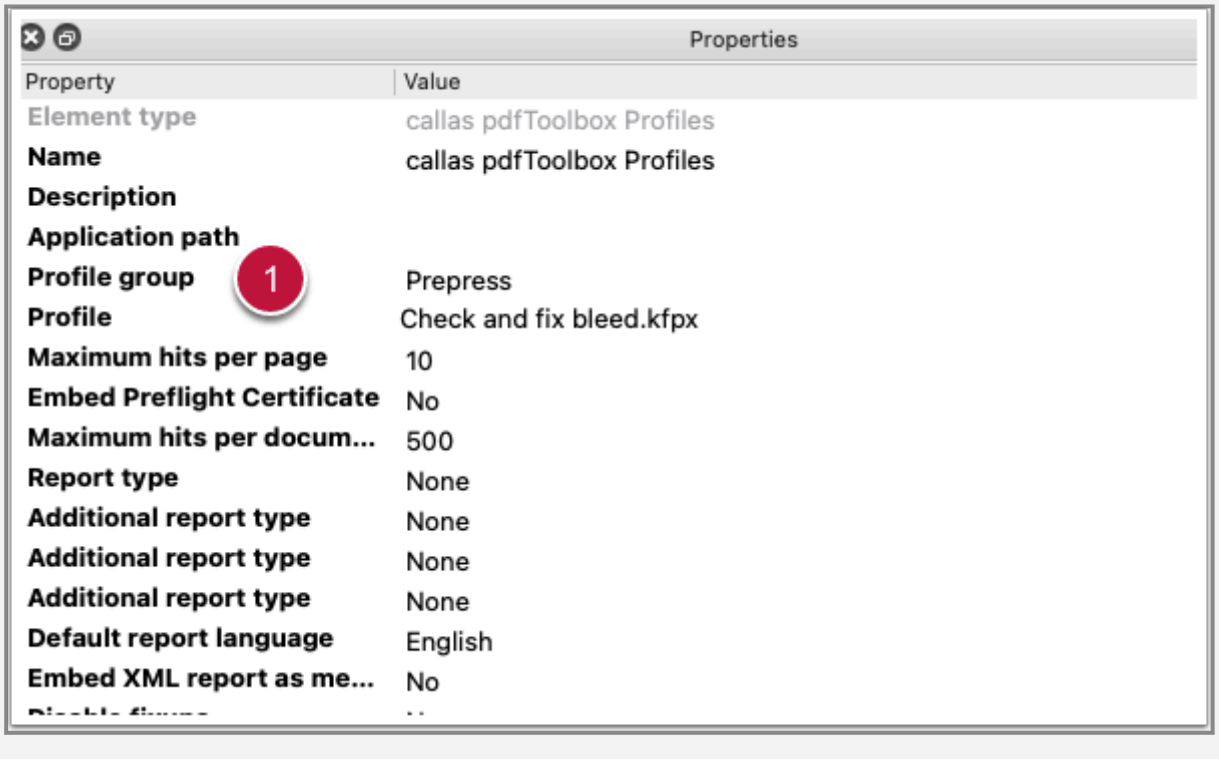
36.3 Setting the path for the pdfToolbox Switch apps

If you're using the pdfToolbox Switch apps in your workflows (as opposed to the configurators supported in older versions of Enfocus Switch), determining the path to the application works differently.

(To know how to work with the path to the configurators, read this article).

Where to set the path to pdfToolbox CLI

When using the pdfToolbox Switch apps, setting the application path has to happen on the actual pdfToolbox flow elements in a Switch flow.



The "Application path" property (1) controls which pdfToolbox CLI application is used when this flow element is used in a Switch flow.

Default and specific path

By default, the "Application path" property is empty. In that case, the app searches for the most recent version of pdfToolbox CLI on the system and uses that.

If you don't want to use the most recent version of pdfToolbox CLI on the Switch server, or your pdfToolbox CLI is installed in a non-standard location, put the full path to the pdfToolbox CLI you want to use in the "Application path" property. Make sure the path points to the pdfToolbox CLI application, and not the pdfToolbox Desktop application.

Mixing different versions

The fact that Enfocus Switch enforces this way of working for Switch apps, has advantages and disadvantages. While it means that pdfToolbox CLI version management now needs to happen on a flow element by flow element basis (unless you leave the property empty and use the most recent version automatically), this also means that you can mix different pdfToolbox versions if needed.

You could, for example:

- Leave the "Application path" property empty for almost all flow elements. They automatically use the most recent version of pdfToolbox CLI on the system.
- Set the "Application path" property of the flow elements in one Switch flow, or even a single flow element, to have it use an older version of the application. This can help with troubleshooting or overcoming a problem with the most recent version of the application.

36.4 Deprecated Switch apps

The pdfToolbox configurators, and the first version of the pdfToolbox apps for Switch, contained five different apps:

- callas pdfToolbox Actions
- callas pdfToolbox Compare
- callas pdfToolbox Profiles
- callas pdfToolbox ColorConvert
- callas pdfToolbox Imposition

Of these five apps, 2 have been deprecated and should no longer be used. This article explains what you should do instead.

pdfToolbox ColorConvert

the pdfToolbox ColorConvert app uses an old color conversion engine that has been deprecated and should no longer be used. Instead, use the pdfToolbox Profiles app with a pdfToolbox Preflight Profile or Process Plan. In both you can use all of the modern color conversion and manipulation fixups such as:

- Convert colors
- Convert colors using DeviceLink profiles
- Convert colors using n-channel profiles
- Map colors
- Map spot and process colors
- ...

The capabilities of these fixups far surpasses what was possible using the ColorConvert app.

pdfToolbox Imposition

The capabilities of the pdfToolbox Imposition app have been deprecated in favor of the pdfToolbox Actions app. In this pdfToolbox Actions app, there is now a new action type called "Impose" that gives you the same capabilities as the old Imposition app.

Alternatively, it is possible to use an "Impose" Action inside a Process Plan. This also offers the same capabilities, and has

the advantage that the imposition step can be combined with other Process Plan steps (for example to run fixups or checks before going to the imposition step).