

Bluebeam Revu eXtreme Script Reference Version 1.0.8

Bluebeam, Inc.
Published February 27, 2019
Applies to Bluebeam Revu® eXtreme®

This document is for informational purposes only and is provided by Bluebeam, Inc. The accuracy of the information is not guaranteed as Bluebeam products and corresponding reference documents continually evolve to adapt to market conditions. Bluebeam makes no warranties, express or implied, as to the information in this document. No portion of this document can be reproduced, distributed, archived or transmitted in any form, by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the express written permission of Bluebeam, Inc. Further, Bluebeam may have patents, patent applications, copyrights, trademarks, or other intellectual property covering the subject matter included in this document. Furnishing this document does not provide any license to these patents, trademarks, copyrights or other intellectual property. Any rights must be expressly provided in a written and authorized license agreement.

© 2019 Bluebeam, Inc. All rights reserved. Bluebeam, Revu, Bluebeam Q, Bluebeam Pushbutton Plus, Bluebeam Lite, and Bluebeam Pushbutton PDF are either registered trademarks or trademarks of Bluebeam, Inc. in the United States and other countries. All other trademarks are the property of their respective owners.

Introduction	3
Syntax	3
Commands	3
BalancePages	4
ColorProcess	4
ColumnsExport	4
ColumnsImport	5
DeleteFile	5
EmbedJavaScript	5
Export	5
FilePropertySet	6
Flatten	6
FormExport	7
FormImport	7
HeaderAndFooter	7
Import	8
InsertBlankPages	9
InsertPages	9
PageDelete	9
PageExtract	10
PageRotate	10
Print	10
PrintToFile	11
ReduceFileSize	12
Repair	12
ReplacePages	12
ReversePages	13
SetOpenPassword	13
SetPDFSecurity	13

SplitPages.....	14
Stamp.....	14
Thumbnail.....	15
Unflatten.....	15
UnlockPDFA.....	15

Introduction

Scripting is integrated into Bluebeam Revu eXtreme. Revu manages the files that the scripts will act on. Scripts can be run on the active document, or in batch mode. Revu will automatically open, save, and close the PDF files depending on the mode, and whether or not the script is set to commit changes.

A major advantage of scripting within Revu is that a script can be assigned to a toolbar button. When pushed the script is run on the active document. Some commands are only available when running Scripts from within Revu; Email is such a command. For example a script could be written that stamps, flattens, and launches an email with the PDF attached.

Syntax

Bluebeam Scripts are a series of commands that are single word identifiers followed by a comma delimited list of parameters enclosed in paranthesis.

For example:

```
SetPDFSecurity("abacadabra", 193)
```

In this example the PDF file being processed would have PDF Security applied such that only printing and copying are allowed.

There are several different types in Bluebeam Script:

Bool: Boolean values (true or false)

Number: Either an Integer or Real (1, 3.5, 0.2 ...)

String: Quoted list of characters ("document.pdf" ...) (the escape character is |)

Name: Unquoted list of characters containing lettings and numbers only. (Print, View, Flatten ...)

Color: A special string that represents a Color that is either a Color name such as "Black" or "Red", or a hex string such as

"#FF0000" which indicates Red, or an integer that contains the RGB values as packed bytes where B is the lowest byte.

Additionally comments may be added by using the '%' character. Any characters following the command character will be ignored until a new line character is encountered.

Commands

BalancePages

Description

Inserts blank pages into active document to balance the total number of pages to an odd, even, specific count, or specific page division

Parameters

pType [String]: Specifies how blank pages will be inserted at the end of the pdf file as follows:

- even = Inserts one page if needed to make count even
- odd = Inserts one page if needed to make count odd
- n = Inserts pages to make page count divisible by n, n is a number
- n = Inserts pages to make page count at least n pages, n is a number

pWidth [String, Optional]: Width of page in inches, last means width of last page

pHeight [String, Optional]: Height of page in inches, last means height of last page

pStyle [Number, Optional]: Page Style as follows:

- 0 = Blank
- 1 = Notebook
- 2 = 1/8" Grid
- 3 = 1/4" Grid
- 4 = Engineering Grid
- 5 = 0.5 cm Grid
- 6 = 1 cm Grid
- 7 = 1/2" Isometric Grid
- 8 = 0.5 cm Isometric Grid

Example

```
BalancePages("even")  
BalancePages(4, 8.5, 11, 1)
```

ColorProcess

Description

Converts page content colors to a color or gray scale.

Parameters

pStartColor [Color]: Start color to convert source colors to, usually darker color

pEndColor [Color]: End color to convert source colors to, usually lighter color

pScale [Bool, Optional]: Indicates that colors should be scaled from start to end

pProcessImages [Bool, Optional]: Images should be converted to new colors

pPageRange [String, Optional]: List or range of pages to be processed, -1 will process all pages, exp:
1,2,10-20

Example

```
ColorProcess("black", "white") % Convert to grayscale  
ColorProcess("Red", "white", true, true, "10-20")
```

ColumnsExport

Description

Exports the Custom Column definition of the active document to an .xml file.

Parameters

pFileName [String]: Filename to export the columns into

Example

```
ColumnsExport('columns.xml')
```

ColumnsImport

Description

Imports a Custom Column definition .xml file into the active document overwriting any existing Custom Columns. An .xml file can be generated by either the command ColumnsExport, or from within Bluebeam Revu.

Parameters

pFileName [String]: Filename of the Custom Column definition .xml file to import into the active document

Example

```
ColumnsImport("columns.xml")
```

DeleteFile

Description

Deletes a file from specified location.

Parameters

pFileName [String]: Filename to delete from the file system

Example

```
DeleteFile("c:\Directory\Filename.pdf")
```

EmbedJavaScript

Description

Embeds the specified JavaScript file as a document level script in the active document.

Parameters

pName [String]: Name of JavaScript Code

pFile [String]: JavaScript file to embed

Example

```
EmbedJavaScript("File.js")
```

Export

Description

Exports the markups in the active document to the specified output file optionally using a User ID to filter on.

Parameters

pOutputBAX [String, Optional]: Filename to export the markups into

pUserID [String, Optional]: User ID as used in bFX File Exchange to filter on when exporting markups

Example

```
Export("output.bax")
```

```
Export("output.bax", "12345")
```

FilePropertySet

Description

Sets a file property in the active document based on the specified key and value.

Parameters

pKey [String]: Key of file property to set

pValue [String]: Desired value of file property

Example

```
FilePropertySet("Author", "Homer J. Simpson")
```

Flatten

Description

Takes the active document and flattens all markups to be part of the page content.

Parameters

pRecoverable [Bool, Optional]: Specifies whether or not the flatten process is reversible

pFlags [Number, Optional]: Specifies what type of markups to flatten

Default = 8191

Image = 1

Ellipse = 2

Stamp = 4

Snapshot = 8

Text and Callout = 16

Ink and Highlighter = 32

Line and Dimension = 64

Measure Area = 128

Polyline = 256

Polygon and Cloud = 512

Rectangle = 1024

Text Markups = 2048

Group = 4096

File Attachment = 8192

Flags = 16384

Notes = 32768

Form Fields = 65536

Add together all values that should be flattened

pPageRange [String, Optional]: List or range of pages to be flattened, -1 will flatten all pages, exp:
1,2,10-20

pLayerName [String, Optional]: Layer Name to flatten markups to

Example

Flatten()

Flatten(true)

Flatten(true, 9) % Flattens Images (1) and Snapshots (8)

FormExport

Description

Exports the form data in the active document to a .xml, .csv, or .fdf file.

Parameters

pFileName [String]: Filename (.xml, .csv, or .fdf) to export the form data into

Example

FormExport("formdata.fdf")

FormImport

Description

Imports an FDF file containing form data into the active document.

Parameters

pFileName [String]: Filename of FDF file to import into the active document

Example

FormImport("formdata.fdf")

HeaderAndFooter

Description

Applies headers and footers to the active document. There are many codes that can be passed in as part of the header or footer text that will be dynamically substituted when the text is applied to the document.

Page Index Codes

<<1>>, <<1 of n>>, <<1/n>>, <<Page 1>>, <<Page 1 of n>>

Date Codes

<<M/d>>, <<M/d/yy>>, <<M/d/yyyy>>, <<MM/dd/yy>>, <<MM/dd/yyyy>>, <<d/M/yy>>, <<d/M/yyyy>>, <<dd/MM/yy>>, <<dd/MM/yyyy>>, <<MM/yy>>, <<MM/yyyy>>, <<ddd MMM d, yyyy>>, <<dddd MMMM d, yyyy>>, <<MM/dd/yyyy h:mm tt>>, <<dd/MM/yyyy HH:mm>>

Bates Numbering

<<Bates Number#Digits#Start#Prefix#Suffix>>

Examples:

<<Bates Number#6#1#_Prefix#_Suffix>>, <<Bates Number#6#1>>

File Properties

Headers and Footers also support pulling file property data from the PDF, any file property key can be used such as:

<<Title>>, <<Author>>, <<Client>> ...

These are additional special codes:

<<FileName>>, <<Path>>, <<PageLabel>>

Parameters

pTopLeft [String]: Header text for top left of page

pTopCenter [String]: Header text for top center of page

pTopRight [String]: Header text for top right of page

pBottomLeft [String]: Footer text for bottom left of page

pBottomCenter [String]: Footer text for bottom center of page

pBottomRight [String]: Footer text for bottom right of page.

pMarginLeft [Number, Optional]: Left margin in points (72 points per inch)

pMarginTop [Number, Optional]: Top margin in points (72 points per inch)

pMarginRight [Number, Optional]: Right margin in points (72 points per inch)

pMarginBottom [Number, Optional]: Bottom margin in points (72 points per inch)

pFont [String, Optional]: Name of font to use with header and footer

pSize [Number, Optional]: Size of font

pBold [Bool, Optional]: Emboldens font

pItalic [Bool, Optional]: Italicizes font

pUnderline [Bool, Optional]: Underlines text

pColor [Color, Optional]: Font color

pFitToContent [Bool, Optional]: Make content of page fit inside margins

pBatesOffset [Number, Optional]: The offset of the bates numbering

pBatesKey [String, Optional]: The unique key used to persistently store the last used Bates offset. Use this key to ensure that every bates number will be unique across documents.

pPageRange [String, Optional]: List or range of pages to apply the header and footer to, -1 will apply to all pages, exp: 1,2,10-20

Example

```
HeaderAndFooter("", "<<dddd MMMM d, yyyy>>","<<h:mm ss tt>>","<<Author>>","", "<<Page 1 of n>>",  
108, 28.8, 108, 48, "Blackadder ITC", 10.0, false, false, "Red",  
,false, 93, "1,3,5,10-20")
```

Import

Description

Imports the markups from list of files specified as parameters into the active document.

Parameters

pBAXorPDF [String, ...]: Filename of a bax or pdf file to import into the active document

Example

```
Import("markups1.bax", "markups2.bax" ...)
```



```
Import("revA.pdf" ...)  
Import("markups1.bax", "revB.pdf" ...)
```

InsertBlankPages

Description

Inserts new blank pages into the active document using the specified parameters for width, height, count and style. The default count is 1 and the default style is blank.

Parameters

pIndex [Number]: Page Index in the active document to insert pages after, 0 is before first page.

pWidth [Number]: Width of page in inches

pHeight [Number]: Height of page in inches

pCount [Number, Optional]: Number of pages to insert

pStyle [Number, Optional]: Page Style as follows:

- 0 = Blank
- 1 = Notebook
- 2 = 1/8" Grid
- 3 = 1/4" Grid
- 4 = Engineering Grid
- 5 = 0.5 cm Grid
- 6 = 1 cm Grid
- 7 = 1/2" Isometric Grid
- 8 = 0.5 cm Isometric Grid

Example

```
InsertBlankPages(0, 8.5, 11)  
InsertBlankPages(2, 8.5, 11, 10, 3)
```

InsertPages

Description

Inserts a PDF file into the active document using the specified parameters to control what additional data to be additionally imported such as bookmarks, file attachments, and file properties

Parameters

pIndex [Number]: Page Index in the active document to insert pages after, 0 is before first page.

pFileName [String]: Filename of document to insert

pBookmarks [Bool, Optional]: Insert bookmarks from inserted file, default is false

pAttachments [Bool, Optional]: Insert file attachments from inserted file, default is false

pProperties [Bool, Optional]: Merge document properties from inserted file, default is false

pLayers [Bool, Optional]: Merge document layers from inserted file, default is false

pUseFileName [Bool, Optional]: User file name as Page Label, default is false

Example

```
InsertPages(0, "Document.pdf")  
InsertPages(0, "Document.pdf", true, true, true, true)
```

PageDelete

Description

Deletes pages from the current document.

Parameters

pPageRange [String]: List or range of pages to delete. Cannot delete all pages. exp: 1,2,10-20

Example

```
PageDelete("1,2,10-20")
```

PageExtract

Description

Extracts pages from the currently active pdf document.

Parameters

pPageRange [String]: List or range of pages to Extract, -1 will extract all pages, exp: 1,2,10-20

pFileNameOrDirectory [String]: Filename or directory to save the extracted pages to

pPrefix [String, Optional]: A prefix that can be appended to the filename

pSuffix [String, Optional]: A suffix that can be appended to the filename

Example

```
PageExtract("1-3", "c:\Directory\file.pdf")
```

```
PageExtract("1,5,10-20", "c:\Directory")
```

```
PageExtract("1,5,10-20", "filename.pdf")
```

```
PageExtract("1,5,10-20", "", "prefix_", "_suffix")
```

PageRotate

Description

Rotates the active document pages by 90 degree increments.

Parameters

pRotations [Number]: Degrees to rotate pages by, must be multiple of 90

pPortrait [Bool, Optional]: Include portrait pages, default is true

pLandscape [Bool, Optional]: Include landscape pages, default is true

pPageRange [String, Optional]: List or range of pages to be Rotated, -1 will rotate all pages, exp:
1,2,10-20

Example

```
PageRotate(90)
```

```
PageRotate(-90, false, true, "10-20")
```

Print

Description

Prints the active document to a physical printer. There are only 3 syntaxes available for this function, see examples below. If advanced printing options are required, all 9 parameters must be specified.

Parameters

pPrinter [String, Optional]: Name of Printer
pPageSize [String, Optional]: Page size as it appears on Printer
pLandscape [Bool, Optional]: Whether to print landscape(true) or portrait(false)
pPageRange [String, Optional]: List or range of pages to be printed, -1 will print all pages, exp: 1,2,10-20
pAutoRotateAndCenter [Number, Optional]: Automatically rotated and center page content on paper.

-1 : Autorotate and center -90
0 : No autorotate and center
1 : Autorotate and center 90

pScaleType [Number, Optional]: Specifies how to scale when printing according to the following:

0 = None
1 = Fit to Paper
2 = Shrink large Images
3 = Custom

pCustomScale [Number, Optional]: If scale type is set to custom, this is the custom scale value (e.g. 0.5 would be 50%)

pDim [Bool, Optional]: Specifies whether to dim the content when printing

pCopies [Number, Optional]: Number of copies to print

Example

```
Print()  
Print("HP Laserjet")  
Print("HP Laserjet", "letter", false, "1-3", true, 1, 1, false, 1)
```

PrintToFile

Description

Prints the active document to a file. There are only 3 syntaxes available for this function, see examples below. If advanced printing options are required, all 10 parameters must be specified.

Parameters

pFileName [String]: File to print output to
pPrinter [String, Optional]: Name of Printer
pPageSize [String, Optional]: Page size as it appears on Printer
pLandscape [Bool, Optional]: Whether to print landscape(true) or portrait(false)
pPageRange [String, Optional]: List or range of pages to be printed, -1 will print all pages, exp: 1,2,10-20
pAutoRotateAndCenter [Number, Optional]: Automatically rotated and center page content on paper.

-1 : Autorotate and center -90
0 : No autorotate and center
1 : Autorotate and center 90

pScaleType [Number, Optional]: Specifies how to scale when printing according to the following:

0 = None
1 = Fit to Paper
2 = Shrink large Images
3 = Custom

pCustomScale [Number, Optional]: If scale type is set to custom, this is the custom scale value (e.g. 0.5 would be 50%)

pDim [Bool, Optional]: Specifies whether to dim the content when printing
pNumberOfCopies [Number, Optional]: Number of copies to print

Example

```
PrintToFile("out.prn")  
PrintToFile("out.prn", "HP Laserjet")  
PrintToFile("out.prn", "HP Laserjet", "letter", false, "1-3", true, 1, 1, false, 1)
```

ReduceFileSize

Description

Takes the active document and reduces file size.

Parameters

pReduceImageLevel [String, Optional]: Specifies the compression ratio. Possible values are Low, Medium and High, the default is Low.
pImageDPI [Number, Optional]: Specifies the Image's maximum DPI, the default is 150.
pDropFonts [Bool, Optional]: Specifies if Fonts will be dropped or not, the default is true.
pDropMiscellaneous [Bool, Optional]: Specifies if Metadata, Thumbnails, Private Data, Unused Resources will be dropped or not, the default is true.
pCompressAllStreams [Bool, Optional]: Compress All Streams, the default is true.

Example

```
ReduceFileSize()  
ReduceFileSize("High", 150, true, true, true)
```

Repair

Description

Runs a repair process on the active document using the specified options.

Parameters

pFixStripedImages [Bool]: Groups neighboring image stripes into a single image
pCombineStripedImages [Bool]: Attempts to merge groups of thin adjacent images into one image
pOptimizeSolidColorImages [Bool]: Converts single color images into vector rectangles
pProcessMasks [Bool]: Fixes AutoCAD files with Blend Modes and Masks
pRemoveTextClipping [Bool]: Fixes AutoCAD files with text clipping problems
pSimplifyClippingPaths [Bool]: Fixes Revit files with text clipping problems
pRepairFontIssues [Bool]: Fixes issues with fonts

Example

```
Repair(true, true, true, true, true, true, true)
```

ReplacePages

Description

Replaces pages in the current document with pages from the source document.

Parameters

pSourceFileName [String]: PDF document to get pages from

pSourcePages [String]: List or range of all source pages to use, -1 will use all pages, exp: 1,2,10-20
pPagesToReplace [String]: List or range of pages to replace, -1 will replace all pages, exp: 1,2,10-20
pContentOnly [Bool, Optional]: If true only the page content will be replaced leaving markups and hyperlinks

Example

```
ReplacePages("c:\Directory\test.pdf", "1", "1")  
ReplacePages("c:\Directory\test.pdf", "3,6", "4,7", true)
```

ReversePages

Description

Reverses all pages in the document.

Parameters

None

SetOpenPassword

Description

Sets open password on active document.

Parameters

pOpenPassword [String]: The open password need to open PDF
pEncryptionLevel [String, Optional]: Encryption Level to use. Values can be RC4, AES128 or AES256.

Example

```
SetOpenPassword("abacadabra")
```

SetPDFSecurity

Description

Applies security permissions to the active document.

Parameters

pPermissionPassword [String]: Password to lock pdf permissions
pFlags [Number]: Specifies what permission are allowed

Print = 1
PrintLowOnly = 2
FillForms = 4
EditMarkups = 8
EditDocument = 16
PageManipulation = 32
CopyContent = 64
Accessibility = 128

Add together values to set permissions

pOpenPassword [String, Optional]: Password used to open the pdf file

pEncryptionLevel [String, Optional]: Encryption Level to use. Values can be RC4, AES128 or AES256.

Example

```
SetPDFSecurity("master", 1)  
SetPDFSecurity("master", 13, "open")
```

SplitPages

Description

Extracts all pages in page range to individual files.

Parameters

pPageRange [String]: List or range of pages to Extract, -1 will extract all pages, exp: 1,2,10-20

pDirectory [String]: Directory to save the extracted pages to

pUsePageLabels [Bool, Optional]: Use page labels to name extracted pages as pdf files.

pPageFormat [String, Optional]: Format to number files names for multiple pages, if pUsePageLabels is true then this parameter will be ignored

Example

```
SplitPages("-1", "c:\Directory")  
SplitPages("1,5,10", "c:\Directory", true )  
SplitPages("1,5,10-20", "c:\Directory", false, " Page 001")
```

Stamp

Description

Places a stamp on the active document using the specified parameters.

Parameters

pFileName [String]: Filename of Stamp

pOrigin [String]: Origin of where to place the stamp as follows:

```
"upperleft"  
"upperright"  
"lowerleft"  
"lowerright"  
"center"  
"uppercenter"  
"lowercenter"
```

pXOffset [Number]: X Offset from origin in inches

pYOffset [Number]: Y Offset from origin in inches

pRotation [Number, Optional]: Rotation in Degrees

pScale [Number, Optional]: Scale (e.g. 0.5 would be 50%)

pOpacity [Number, Optional]: Opacity (0.4 would be 40% opacity)

pBlendMode [String, Optional]: Blend Mode as follows:

```
"normal"  
"multiply"  
"screen"  
"overlay"  
"darken"
```

"lighten"
"colordodge"
"colorburn"
"hardlight"
"softlight"
"difference"
"exclusion"
"luminosity"
"hue"
"saturation"
"color"

pPageRange [String, Optional]: List or range of pages to be stamped, -1 will stamp all pages, exp:
1,2,10-20

pLocked [Bool, Optional]: Specifies whether or not the stamp should be locked

Example

Stamp("mystamp.brx", "lowerright", 0.5, 1.0, 0, 1, 1, "normal")

Thumbnail

Description

Creates a thumbnail of given width and height and saves it to the specified filename. Can have an extension of most common image formats including (.bmp, .png, .jpg ...)

Parameters

pWidth [Number]: Desired width in pixels of output thumbnail image

pHeight [Number]: Desired height in pixels of output thumbnail image

pFileName [String]: Filename of desired output thumbnail image.

pPageFormat [String, Optional]: Suffix used when generatating thumbnails for multiple pages. " Page 001" would cause the resulting files to be named "File Page 001.png", "File Page 002.png" ...

pPageRange [String, Optional]: List or range of pages to have thumbnails generated for, -1 will generate thumbnails for all pages, exp: 1,2,10-20

pShowPopups [Bool, Optional]: Indicates that popups should be included

Example

thumbnail(320, 200, "thumbnail.png")

Unflatten

Description

Reverses the flattening process on the active document.

Parameters

pPageRange [String, Optional]: List or range of pages to unflatten, -1 will unflatten all pages, exp:
1,2,10-20

Example

Unflatten()

UnlockPDFA

Description

Unlocks the current PDF/A-1b document for editing.

Parameters

None

Example

UnlockPDFA()