NEEVIA

docCreator v5.0

API Reference

Table of Contents

License

NEEVIA TECHNOLOGY

ELECTRONIC END USER LICENSE AGREEMENT

For One (1) Computer/Server/Virtual Server

This is an End User License Agreement. This is a contract. If you install this software, you must abide by the terms of this agreement. This license is applicable to all software products sold by Neevia Technology (Neevia). The term software includes upgrades, modified versions or updates. This software is licensed and not sold. Only a personal, non-transferable and nonexclusive right to use the Neevia products is granted to the end user.

The following are definitions that should be noted by the user:

a. SERVER

This is a single computer owned, rented or leased by a single individual or entity on which one or more applications load and execute software in the memory space of that computer. Software is installed on a server for one or more users. All servers must be licensed to utilize Neevia software.

b. VIRTUAL SERVER

This is a single computer or a virtual machine (a software implementation of a machine that executes programs like a physical machine) that is owned, rented or leased by an individual or entity that turns around and rents or leases access to others. The virtual server may have one or more applications on it for the end users to use. The purpose of the virtual server is to give multiple users access to many software programs.

c. DEVELOPMENT

This means that you are programming a specific application or tool that will interact with the software that you are licensing from Neevia Technology.

THIS IS A CONTRACT BETWEEN YOU AND NEEVIA TECHNOLOGY. YOU SHOULD CAREFULLY READ THIS LICENSING AGREEMENT AND MUST ACCEPT ALL THE TERMS AND CONDITIONS BEFORE INSTALLING THIS NEEVIA SOFTWARE. BY INSTALLING THE SOFTWARE, YOU ARE AGREEING TO BE BOUND BY THE TERMS AND CONDITIONS OF THIS LICENSE. IF YOU DO NOT AGREE TO THE TERMS OF THIS LICENSE, DO NOT INSTALL THE SOFTWARE, AND DO NOT USE THE SOFTWARE. IF YOU VIOLATE THIS AGREEMENT, YOU WILL BE SUBJECT TO LEGAL ACTION BY NEEVIA TECHNOLOGY.

Subject to the payment of applicable license fees, Neevia Technology grants you a nonexclusive right to use its accompanying Neevia software product and related documents (the Software) in the terms and conditions provided as follow:

LICENSE

Until such time as Neevia Technology has issued a valid serial number to you, you may only use this software for a 60-day trial period. You agree to remove any copies of the software after the expiration of the trial period. No license is issued to you until you are issued a valid serial number.

(a) Home Use:
The primary user of each computer on which the Software is installed or used may also install the Software on one home or portable computer. However another person may not use the Software on a secondary computer at the same time the Software on the primary computer is being used.

(b) Server or Network Use:
You may store or install one (1) copy of the SOFTWARE on a storage device, such as a network server, for backup and archival purposes only. A license for the SOFTWARE may not be shared or used concurrently on different computers.

(c) Operating system or Language versions:
If you receive two or more copies of the Software with different operating systems or language versions, the total aggregate number of computers on which all versions of the Software are used may not exceed the Permitted Number of Computers. You may not rent, lease, sublicense, lend or transfer versions or copies of the Software you do not use, or Software contained on any unused media.

(d) Archiving:
You may make one copy of the Software solely for archival purposes. If the Software is an upgrade, you may use the Software only in conjunction with upgraded product. If you receive your first copy of the Software electronically, and a second copy on media afterward, the second copy can be used for archival purposes only.

You agree to surrender your license(s) if you violate this agreement. If you violate this agreement, you will not receive a refund upon termination of this license. You agree not to utilize our software to violate the copyright of any third parties. If you do violate the copyright of a third party utilizing our software, you agree to hold Neevia Technology harmless and will indemnify Neevia Technology for any such activity even if the violation is unintentional.

COPYRIGHT

The Software is owned by Neevia Technology and/or its suppliers, and is protected by the copyright and trademark laws of the United States and related applicable laws. You may not copy the Software except as set forth in the "License" section. Any copies that you are permitted to make pursuant to this Agreement must contain the same copyright and other proprietary notices that appear on or in the Software.

You may not rent, lease, sub-license, transfer, or sell the Software. You may not modify, translate, reverse engineer, decompile, disassemble, or create derivative works based on the Software, except to the extent applicable law expressly prohibits such foregoing restriction. You may use the trademarks to identify the Software owner's name, or to identify printed output produced by the Software. Such use of any trademark does not give you any rights of ownership in that trademark.

NO WARRANTY LICENSED SOFTWARE (S) - "AS IS"

The Software is provided AS IS. NEEVIA TECHNOLOGY AND ITS SUPPLIERS MAKE NO WARRANTIES, EXPRESS OR IMPLIED, AS TO THE MERCHANTABILITY, QUALITY, NONINFRINGEMENT OF THIRD PARTY RIGHTS, FITNESS FOR A PARTICULAR PURPOSE, AND THOSE ARISING BY STATUTE OR OTHERWISE IN LAW OR FROM A COURSE OF DEALING OR USAGE OF TRADE. THE ENTIRE RISK AS TO THE QUALITY, RESULTS BY USING THE SOFTWARE, AND PERFORMANCE OF THE SOFTWARE IS WITH THE END USER. Some states or jurisdictions do not allow the exclusion or limitation of incidental, consequential or special damages, or the exclusion of implied warranties or limitations on how long an implied warranty may last, so the above limitations may not apply to your or your company.

LIMITATION OF REMEDIES AND LIABILITY

NEEVIA TECHNOLOGY OR ITS SUPPLIERS OR RESELLERS SHALL NOT UNDER ANY CIRCUMSTANCE BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING BUT NOT LIMITED TO SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST SAVINGS, OR FOR ANY CLAIM BY A THIRD PARTY, ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE, EVEN IF NEEVIA TECHNOLOLGY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

GENERAL

This Agreement shall be construed, interpreted, and governed by the laws of the State of Florida, excluding the application of its conflicts of law rules. The United Nations Convention on Contracts of the International Sale of Goods will not govern this Agreement. If any part of this Agreement is found void and unenforceable, it will not affect the validity of the rest of the Agreement, which shall remain valid and enforceable according to its terms.

If you need to redistribute this product with your own software products, you need to contact Neevia and negotiate a separate licensing and royalty agreement.

You may not ship, transfer, or export the Software into any country or used in any manner prohibited by any export laws, restrictions or regulations.

UPGRADES

You must be properly licensed to install upgrades to Neevia Software products. Neevia upgrades replace and or supplement the previous product that formed the basis for your eligibility to for the upgrade. You may use the upgrade only in accordance with the terms of this Agreement. Upgrades may not be separated and used on separate computers.

GOVERNEMENT USERS

For United States government users, the Software and associated Documentation are deemed to be “commercial computer software” and “commercial computer documentation”, respectively pursuant to DFAR 227.7202 and FAR 12.212(b) as applicable.

ENTIRE AGREEMENT

You acknowledge that you have read this Agreement, understand it and agree to be bounded by its terms and conditions. It is the complete and exclusive statement of the Agreement between us, which supersedes any proposal or prior agreement, oral or written, and other communication between us relating to the subject matter of this Agreement.

Neevia docCreator API Reference

Neevia docCreator supports conversion through a COM object / .NET assembly.

Class ID: Neevia.docCreator

Example:

VBScript:
```
Set NVDC = CreateObject("Neevia.docCreator")
```

C#:
```
Neevia.docCreator NVDC = new Neevia.docCreator();
```

VB.NET:
```
Dim NVDC as New Neevia.docCreator()
```

NOTE: By default, the .NET assembly can be found in the **Program Files (x86)\neevia.com\docCreator\.NET** folder;

Methods

doSleep

Suspends execution of the current thread for a specified interval.

Syntax

NVDC.doSleep(sleepTime)

Parameters

sleepTime - specifies the amount of time, in milliseconds, for which to suspend execution.

setParameter

Sets a conversion parameter. (see Conversion parameters for a full list)

Syntax

NVDC.setParameter(paramName, paramValue)

Parameters

paramName - parameter name;
paramValue - parameter value (string);

Example

NVDC.setParameter("DocumentOutputFormat", "PDF")

getParameter

Returns the value of a conversion parameter previously set by the setParameter method.

Syntax

Res = NVDC.getParameter(paramName)

Parameters

paramName - parameter name;

getDefaultPrinter

Returns the default system printer.

Syntax

Res = NVDC.getDefaultPrinter

setDefaultPrinter

Sets the default system printer.

Syntax

Res = NVDC.setDefaultPrinter(printerName)

Parameters

printerName - printer name;

Remarks

Res <> 0 on error.

startPrinting

This method will create a new virtual printer.

Syntax

Res = NVDC.startPrinting

Remarks

Res <> 0 on error.

newPrinterName

Returns the name of the printer created by the startPrinting method.

Syntax

Res = NVDC.newPrinterName

newPortName

Returns the printer port associated with the printer created by the startPrinting method.

Syntax

Res = NVDC.newPortName

stopPrinting

Call this method when your application has completed printing to clear up the virtual printer and the variables that have been created by startPrinting method.

Syntax

Res = NVDC.stopPrinting

Remarks

Res <> 0 on error.

createStaticPrinter

Call this method to create a new system printer based on "Neevia docCreator" printer.

Syntax

Res = NVDC.createStaticPrinter(printerName, printerPort)

Parameters

printerName - printer name;
printerPort - printer port, Ex: "c:\testPort.ps";

Remarks

Res <> 0 on error.

deleteStaticPrinter

Deletes a printer from the system.

Syntax

Res = NVDC.deleteStaticPrinter(printerName)

Parameters

printerName - printer name;

Remarks

Res <> 0 on error.

createPrinterPort

Creates a new printer port.

Syntax

Res = NVDC.createPrinterPort(printerPort)

Parameters

printerPort - printer port, Ex: "c:\testPort.ps";

Remarks

Res <> 0 on error.

deletePrinterPort

Deletes a printer port from the system.

Syntax

Res = NVDC.deletePrinterPort(printerPort)

Parameters

printerPort - printer port, Ex: "c:\testPort.ps";

Remarks

Res <> 0 on error.

getPrinterPort

Returns the printer port associated with a specific printer.

Syntax

Res = NVDC.getPrinterPort(printerName)

Parameters

printerName - printer name;

Remarks

Res <> 0 on error.

setPrinterPort

Associates a printer port with a specific printer.

Syntax

Res = NVDC.getPrinterPort(printerName, printerPort)

Parameters

printerName - printer name;
printerPort - printer port, Ex: "c:\testPort.ps";

Remarks

Res <> 0 on error.

setInputDocument

Specifies the input document (must be PostScript, EPS or PDF) to convert. Use this method together with **create** method.

Syntax

Res = NVDC.setInputDocument(fileName, filePassword)

Parameters

fileNamer - input file;
filePassword - open password if input is an encrypted PDF file (this parameter is optional);

Remarks

Res <> 0 on error.

create

Call this method to create the output document.

Syntax

Res = NVDC.create

Remarks

Res <> 0 on error.

timeOut

Specifies the timeout period for the create method (in seconds).

Syntax

NVDC.timeOut = 360

Note: Can only be set before calling the Create method.

convertImage

This method converts an image into a different image format - see convertImageParam for supported parameters.

Syntax

Res = NVDC.convertImage(inputFile, outputFile)

Remarks

Res <> 0 on error.

Examples

To convert a JPEG image to a BMP raster image, use **NVDC.convertImage "image.jpg", "image.bmp"**
To convert a TIFF image to a PostScript, use **NVDC.convertImage "image.tif", "document.ps"**
To convert a TIFF image to a PDF document, use **NVDC.convertImage "image.tif", "document.pdf"**

linearizePDF

Optimizes an existing PDF file for fast web view.

Syntax

Res = NVDC.linearizePDF(inFile, outFile)

Parameters

inFile - full path to PDF file to linearize;
outFile - full path to destination file;

deletePDFpages

Deletes pages from a specified PDF document.

Syntax

Res = NVDC.deletePDFpages(inFile, outFile, fromPage, toPage)

Parameters

inFile - full path to PDF file to delete pages from;
outFile - full path to destination file;
fromPage - page number to start with;
toPage - page number to end with;

Remarks

Res<>0 on error.

rotatePDFpages

Rotates page(s) in a specified PDF document.

Syntax

Res = NVDC.rotatePDFpages(inFile, outFile, fromPage, toPage, rotate)

Parameters

inFile - full path to PDF file to rotate pages in;
outFile - full path to the destination file;
fromPage - page number to start with;
toPage - page number to end with;
rotate - rotate by (-270, -90, 0, 90, 180, 270) degrees;

Remarks

Res<>0 on error.

isPDFencrypted

Checks if a PDF file is encrypted.

Syntax

Res = NVDC.isPDFencrypted(filename)

Parameters

filename - path to file;

encryptPDF

Encrypts an existing PDF file.

Note: in the trial version all files will be encrypted with "neevia" as user and "owner" as password.

Syntax

```
Res = NVDC.encryptPDF(srcFile, destFile)
```

Parameters

srcFile - full path to the file that needs to be decrypted;
destFile - full path to the decrypted file;

Remarks

Res<>0 on error.

decryptPDF

Decrypts an existing PDF file.

Syntax

```
Res = NVDC.decryptPDF(srcFile, destFile, userPwd)
```

Parameters

srcFile - full path to the file that needs to be decrypted;
destFile - full path to the decrypted file;
userPwd - user password to be used in the decoding process;

Remarks

Res<>0 on error.

mergePDF

Merges two PDF files.

Syntax

```
Res = NVDC.mergePDF(firstFile, secondFile, outFile)
```

Parameters

firstFile - full path to first PDF file;
secondFile - full path to second PDF file;
outFile - full path to resulting file;

Remarks

Res<>0 on error.

mergeMultiplePDF

Merges multiple PDF files.

Syntax

```
Res = NVDC.mergeMultiplePDF(filesToMerge, destFile)
```

Parameters

filesToMerge - PDF files to merge, file names must be separated by +
destFile - output PDF file name;

Example

```
Res = NVDC.mergeMultiplePDF("c:\t1.pdf+c:\t2.pdf+c:\t3.pdf", "c:\out.pdf")
```

splitPDF

Splits an existing PDF.

Syntax

Res = NVDC.splitPDF(fileToSplit, destFolder)

Parameters

fileToSplit - path to input PDF file;
destFolder - path to destination folder;

Example

Res = NVDC.splitPDF("c:\t1.pdf", "c:\")

Remarks

Res<>0 on error.

extractPDFpages

Extracts pages from an existing PDF file.

Syntax

NVDC.extractPDFpages(fileIN, fileOUT, extractFROM, extractTO)

Parameters

fileIN - input PDF file name;
fileOUT - output file name;
extractFROM - extract from this page;
extractTO - extract to this page;

Example

Res = NVDC.extractPDFpages("c:\in.pdf", "c:\out.pdf", 1, 4)
(this will extract pages 1,2,3 and 4 from c:\in.pdf into c:\out.pdf)

Remarks

Res<>0 on error.

stampPDF

Stamps/watermarks an existing PDF document.

Syntax

Res = NVDC.stampPDF(fileToStamp, destFile)

Parameters

fileToStamp - path to input PDF file;
destFile - path to output PDF file;

Example

Res = NVDC.stampPDF("c:\in.pdf", "c:\out.pdf")

Remarks

Res<>0 on error.

mergeTIFF

Merges two TIFF files.

Syntax

```
Res = NVDC.mergeTIFF(firstFile, secondFile, outFile)
```

Parameters

firstFile - full path to first TIFF file;
secondFile - full path to second TIFF file;
outFile - full path to resulting file;

Remarks

Res<>0 on error.

mergeMultipleTIFF

Merges multiple TIFF files.

Syntax

```
Res = NVDC.mergeMultipleTIFF(filesToMerge, destFile)
```

Parameters

filesToMerge - TIFF files to merge, file names must be separated by +
destFile - output TIFF file name;

Example

```
Res = NVDC.mergeMultipleTIFF("c:\t1.tif+c:\t2.tif+c:\t3.tif", "c:\out.tif")
```

Remarks

Res<>0 on error.

splitTIFF

Splits an existing TIFF file.

Syntax

```
Res = NVDC.splitTIFF(fileToSplit, destFolder)
```

Parameters

fileToSplit - path to input TIFF file;
destFolder - path to destination folder;

Example

```
Res = NVDC.splitTIFF("c:\t1.tif", "c:\")
```

Remarks

Res<>0 on error.

extractTIFFpages

Extracts pages from an existing TIFF file.

Syntax

NVDC.extractTIFFpages(fileIN, fileOUT, extractFROM, extractTO)

Parameters

fileIN - input TIFF file name;
fileOUT - output TIFF file name;
extractFROM - extract from this page;
extractTO - extract to this page;

Example

Res = NVDC.extractTIFFpages("c:\in.tif", "c:\out.tif", 1, 4)
(this will extract pages 1,2,3 and 4 from c:\in.tif into c:\out.tif)

Remarks

Res<>0 on error.

getNumPages

Returns the number of pages in the specified PDF/TIFF document.

Syntax

Res = NVDC.getNumPages(fileName)

Parameters

filename - path to file;

getBaseName

Returns the file name (less any file extension) from a path.

Syntax

Res = NVDC.getBaseName(path)

Parameters

path - file path;

getExtensionName

Returns the file extension from file name.

Syntax

Res = NVDC.getExtensionName(filename)

fileExists

Checks if a specified file exists.

Syntax

Res = NVDC.fileExists(fileToCheck)

Parameters

fileToCheck - path to the file to check;

isFileInUse

Checks if a specified file is in use (locked).

Syntax

Res = NVDC.isFileInUse(fileToCheck)

Parameters

fileToCheck - path to the file to check;

fileCopy

Copies a file from source to destination.

Syntax

Res = NVDC.fileCopy(srcFile, destFile)

Parameters

srcFile - path to source file;
destFile - path to destination file;

fileDelete

Deletes a specified file.

Syntax

Res = NVDC.fileDelete(filename)

Parameters

filename - path to the file to delete;

createFolder

Creates a folder.

Syntax

Res = NVDC.createFolder(fldrName)

Parameters

fldrName - folder name;

copyFolder

Copies a folder from source to destination.

Syntax

Res = NVDC.copyFolder(srcFolder, destFolder)

Parameters

srcFolder - source path;
destFolder - destination path;

deleteFolder

Deletes a specified folder and its contents.

Syntax

Res = NVDC.deleteFolder(foldername)

Parameters

foldername - path to the folder to delete;

creatorPath

Returns the path to the docCreator.

Syntax

Res = NVDC.creatorPath

Data Type: String

GUID

Returns an unique identifier.

Syntax

Res = NVDC.GUID

Data Type: String

Conversion parameters - GENERAL

DocumentOutputName

Specifies the output file name (no path and extension). The file extension will be automatically added depending on the output format.

Syntax

NVDC.setParameter("DocumentOutputName", value)

Data type: String

Note: Can only be set before calling the Create method.

DocumentOutputFolder

Specifies the folder where the output document will go.

Syntax

NVDC.setParameter("DocumentOutputFolder", value)

Data type: String

Note: Can only be set before calling the Create method.

DocumentOutputFormat

Sets the output format.

Possible values: "PDF", "PDFA", “PDFA2”, “PDFA3”**,** “PDFA2U”, “PDFA3U” "PS", "EPS", "PNG", "BMP", "JPG", "TIFF", "PCX", "PSD", "PXL", "TEXT".

Syntax

NVDC.setParameter("DocumentOutputFormat", value)

Data type: String

Note: Can only be set before calling the Create method.

Note: when PDFA is specified the output will be PDF/A-1b. To obtain a PDF/A-2b or 3b file you’ll have to use PDFA2 or PDFA3. In case you need to create PDF/A files with text that contains Unicode equivalents please use PDFA2U or PDFA3U.

Known limitations:

TXT output format

The text extraction can be successful only if the printed document itself contains text information. If the document is an image, drawing, metafile, etc, docCreator will not be able to extract any text.

In some cases the printing application sends the text or parts of the text as glyphs. The glyphs codes cannot be converted back to character codes and the text file will contain unreadable characters.

The coordinates of the beginning text are reported by the printing application. The coordinates of the end of the text is calculated by the driver, based on the resolution, font, and the actual characters in the text. A small variation of 1 to 5 pixels is normal.

The coordinates of the text are saved as they are received from the printing application. Some applications, such as Quicken, change the coordinate system during printing. The part of the driver which generates the text output is not aware of this fact. In cases where the printing application changes the coordinate system, the coordinates saved to the text file may not be relative to the upper left corner of the image. There is no workaround for this issue.

When printing content of cells from Excel, the content of the cells is not separated.

Words cut in half. It is possible that the following line "This is a test line" is extracted from a document by docCreator and is saved to the text file as "T his is a te st lin e". This issue is caused by the printing application, which sends the text as different commands. The printer driver is not aware if the text is "correct", the text will be saved exactly how it is received from the printing application. This is most likely to happen with applications such as Word and Notepad, when for example one part of a word or sentence is using one font and another part of the text is using a different font, or a word was typed and then later edited.

When printing a document with Arabic text document with Unicode Text enabled, the Arabic text in the text output is garbage. The origin of the problem is that some languages use fonts that require shaping. Arabic is one of these. When an application prints Arabic text, no printer driver will receive the text itself but instead receives an array of glyph indices. The glyph index is font specific and points to a character location in the font. Because there is no way to convert back the glyph index into a Unicode character code, it is not possible to retrieve the Unicode text. It is caused by the design of the operating systems way of handling printing these kinds of fonts. This problem is found in all printer drivers that capture text, including the docCreator.

FirstPage

Specifies the first page to start the conversion with.

Syntax

NVDC.setParameter("FirstPage", value)

Data type: String

Note: Can only be set before calling the Create method.

LastPage

Specifies the last page to be converted.

Syntax

NVDC.setParameter("LastPage", value)

Data type: String

Note: Can only be set before calling the Create method.

UseCropBox

Specifies whether docCreator should use CropBox for paper size rather than MediaBox when converting PDF files.

Possible values: "true", "false".

Syntax

NVDC.setParameter("UseCropBox", value)

Data Type: String

Note: Can only be set before calling the Create method.

CropEPS

Specifies whether docCreator should crop the EPS/PostSCript files to the bounding box.
Possible values: "true", "false".

Syntax

NVDC.setParameter("CropEPS", value)

Data Type: String
Note: Can only be set before calling the Create method.

UseSystemFonts

Sets the output format.
Possible values: "0" - use only built-in fonts (the missing fonts will be substituted with Helvetica); "1" - use the font mapping file created during docCreator install to locate missing fonts; "2" - scan the <system>\fonts\ folder for missing fonts (very slow);

Syntax

NVDC.setParameter("UseSystemFonts", value)

Data type: String
Note: Can only be set before calling the Create method.

Conversion parameters - PostScript/EPS related

LanguageLevel

Specifies what PostScript language level docCreator should use when generating the output file. Possible values: "1" (PostScript language level 1),
"1.5" (PostScript language level 1.5),
"2" (PostScript language level 2),
"3" (PostScript language level 3).

Syntax

NVDC.setParameter("LanguageLevel", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PS" or "EPS".

Conversion parameters - Image related

TIFFType

Sets image type when output format is "TIFF".
Possible values:

"tiff24nc", (Color - 24bits RGB output, uncompressed)
"tiff32nc", (Color - 32bits CMYK output, uncompressed)
"tiff24lzw", (Color - 24bits RGB, LZW-compatible compression)
"tiff32lzw", (Color - 32bits CMYK, LZW-compatible compression)
"tiff24zip", (Color - 24bits RGB, ZIP (Deflate) compression)
"tiff32zip", (Color - 32bits CMYK, ZIP (Deflate) compression)
"tiffgray", (Grayscale - 8bits output, uncompressed)
"tiffgraylzw", (Grayscale - 8bits output, LZW-compatible compression)
"tiffgrayzip", (Grayscale - 8bits output, ZIP (Deflate) compression)
"tiffg3", (B&W - G3 fax encoding with EOLs)
"tiffg32d", (B&W - 2-D G3 fax encoding)
"tiffg4", (B&W - G4 fax encoding)
"tifflzw", (B&W - LZW-compatible compression)
"tiffpack". (B&W - PackBits compression)

Syntax

NVDC.setParameter("TiffType", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "TIFF".

MultiPageTIFF

Specifies whether docCreator should create multipage tiff files.
Possible values:

"true" (creates a multipage tiff file)
"false" (creates a tiff file for the each page in the input document)

Syntax

NVDC.setParameter("MultiPageTIFF", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "TIFF".

FillOrder

Sets fill order for the TIFF output format.
Possible values: "0" msb-to-lsb, "1" lsb-to-msb.

Syntax

NVDC.setParameter("FillOrder", value)

Data type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "TIFF".

JPGType

Sets image type when output format is "JPG".
Possible values:

"jpeg", (Color - 16m colors RGB output)
"jpegcmyk", (Color - CMYK output)
"jpeggray". (Grayscale output)

Syntax

NVDC.setParameter("JPGType", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "JPG".

JPGQuality

Sets image quality when output format is "JPG".
Possible values: "1"..."100".

Syntax

NVDC.setParameter("JPGQuality", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "JPG".

BMPType

Sets image type when output format is "BMP".
Possible values:

"bmp16m", (Color - 16m colors RGB output)
"bmp16", (Color - 16 colors RGB output)
"bmp256", (Color - 256 colors RGB output)
"bmpgray", (Grayscale output)
"bmpmono". (Monochrome output)

Syntax

NVDC.setParameter("BMPType", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "BMP".

PSDType

Sets image type when output format is "PSD".
Possible values:

"psdrgb", (Color - 24bits RGB output)
"psdcmyk". (Color - 32bits CMYK output)

Syntax

NVDC.setParameter("PSDType", value)
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PSD".

PCXType

Sets image type when output format is "PCX".
Possible values:

"pcx24b", (Color - 24bits RGB output)
"pcx16", (Color - 16 colors RGB output)
"pcx256", (Color - 256 colors RGB output)
"pcxcmyk", (Color - CMYK output)
"pcxgray", (Grayscale output)
"pcxmono". (Monochrome output)

Syntax

NVDC.setParameter("PCXType", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PCX".

PNGType

Sets image type when output format is "PNG".
Possible values:

"png16m", (Color - 16m colors output)
"png16", (Color - 16 colors output)
"png256", (Color - 256 colors output)
"pnggray", (Grayscale output)
"pngmono". (Monochrome output)

Syntax

NVDC.setParameter("PNGType", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PNG".

PXLType

Sets image type when output format is "PXL".
Possible values:

"pxlcolor", (Color - 24bits RGB output)
"pxlmono", (Monochrome output)

Syntax

NVDC.setParameter("PXLType", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PXL".

ScalePage

Specifies whether docCreator should scale the output document. If ScalePage is "false" output document will be stripped to ImgHeight/ImgWidth, if ScalePage is "true" output document will be scaled to ImgHeight/ImgWidth.

Syntax

NVDC.setParameter("ScalePage", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX".

ConstrainProportions

Specifies whether docCreator should constrain proportions when scaling the output document.
Possible values: "true", "false"

Syntax

NVDC.setParameter("ConstrainProportions", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX" and ScalePage is "true".

ScaleIfLarger

Instructs docCreator to perform scaling only if the input document is larger than the output. Possible values: "true", "false"

Syntax

NVDC.setParameter("ScaleIfLarger", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX" and ScalePage is "true".

PlaceContentIn

Specifies where docCreator should place the original page content on the scaled page.
Possible values: "0" (left-bottom corner), "1" (right-bottom corner), "2" (center), "3" (center-top),
"4" (center-bottom), "5" (left-center), "6" (right-center), "7" (left-top corner),
"8" (right-top corner)

Syntax

NVDC.setParameter("PlaceContentIn", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX" and ScalePage is "true".

ImgHeight

Specifies the output document height (in pixels).

Syntax

NVDC.setParameter("ImgHeight", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX".

ImgWidth

Specifies the output document width (in pixels).

Syntax

NVDC.setParameter("ImgWidth", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX".

ImgResH

Specifies the output document horizontal resolution (in dpi).

Syntax

NVDC.setParameter("ImgResH", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX".

ImgResV

Specifies the output document vertical resolution (in dpi).

Syntax

NVDC.setParameter("ImgResV", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX".

FileNameSuffix

Specifies the output filename suffix. If you use "**%d**" docCreator will add the page number to the file name. You can also control the number of digits used in the file name by replacing **%d** with **%0Nd** where **N** is the number of digits you want to use, for example %03d will force docCreator to produce files with names like this: 'filename001.jpg', ... , 'filename010.jpg', ... %04d will produce: 'filename0001.jpg', ... , 'filename0010.jpg', ...

Syntax

NVDC.setParameter("FileNameSuffix", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX".

TextAlphaBits

Controls the use of subsample antialiasing for text content. The subsampling box size should be 4 bits for optimum output, but smaller values can be used for faster rendering.
Possible Values: "**0**", "**1**", "**2**", "**4**"

Syntax

NVDC.setParameter("TextAlphaBits", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX".

GraphicsAlphaBits

Controls the use of subsample antialiasing for graphics content. The subsampling box size should be 4 bits for optimum output, but smaller values can be used for faster rendering.
Possible Values: "**0**", "**1**", "**2**", "**4**"

Syntax

NVDC.setParameter("GraphicsAlphaBits", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "TIFF", "JPG", "BMP", "PNG" or "PCX".
Because of the way antialiasing blends the edges of shapes into the background when they are drawn, some files that rely on joining separate filled polygons together to cover an area may not render as expected with Graphics antialiasing at 2 or 4 bits. If you encounter strange lines within solid areas, try rendering that file again with Graphic antialiasing at 1 bit.

Interpolate

Specifies whether the image parser should use image interpolation. Enabling image interpolation will result in higher quality for scaled images at the expense of speed.
Possible Values: "true", "false"

Syntax

NVDC.setParameter("Interpolate", value)

Data Type: String

UseWTS

Specifies whether Well Tempered Screening algorithm should be used for halftoning.
Possible Values: "true", "false"

Syntax

NVDC.setParameter("UseWTS", value)

Data Type: String

Note: the Well Tempered Screening algorithm is used for halftoning. If not enabled a rational tangent algorithm is chosen, which will typically result in significant differences between the screen angle and ruling requested, and actually rendered. Currently, the performance of WTS is reasonably good when rendering to a full page buffer, but not optimized for banded mode.

UseCIEColor

Specifies whether the image parser should remap the device-dependent color values through a CIE color space. This can improve the conversion of CMYK documents to RGB.
Possible Values: "true", "false"

Syntax

NVDC.setParameter("UseCIEColor", value)

Data Type: String

DitheringMethod

Specifies the dithering algorithm docCreator should use when producing B&W FAX TIFF files.
Possible Values: "0" (sierra), "1" (burkes), "2" (stucki), "3" (floyd), "4" (jarvis), "5" (cluster 6x6), "6" (cluster 8x8), "7" (cluster 16x16), "8" (bayer 4x4), "9" (bayer 8x8)

Syntax

NVDC.setParameter("DitheringMethod", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "FAX" or "SEND VIA FAX".

Conversion parameters - PDF/A related

OutputIntent

Specifies the PDF/A Output Intent.
Possible values: "srgb", "jc200103", "fogra27", "swop", "gray".

Syntax

NVDC.setParameter("OutputIntent", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDFA".

Conversion parameters - PDF related

PDFtoPDF

Specifies whether the PDF document set by the setInputDocument method has to be reparsed when converting into PDF.
Possible values: "true", "false".

Syntax

NVDC.setParameter("PDFtoPDF", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFMarksFile

Specifies the file to load PDFmark commands from.

Syntax

NVDC.setParameter("PDFMarksFile", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

LinearizePDF

Specifies whether the output PDF document should be linearized (optimized for fast web view).
Possible values: "true", "false".

Syntax

NVDC.setParameter("LinearizePDF", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

AttachOriginalFile

Specifies whether original file should be included as attachment in the output PDF.
Possible values: "true", "false"

Syntax

NVDC.setParameter("AttachOriginalFile", value)

Data Type: String

OptimizePDFfor

Possible values: "default", "screen", "printer", "prepress", "ebook".

For your convenience there are several sets of predefined settings for creating PDF files. These settings are designed to balance file size with quality, depending on how the PDF file is to be used:

• **Default** - is intended to be useful across a wide variety of uses, possibly at the expense of a larger output file. All color and grayscale images are downsampled at 72 dpi, monochrome images at 300 dpi; subsets of all fonts used in the file are embedded; and all information is compressed. PDF files created using the Default job option are compatible with Acrobat 4.0 (and later).

• **Screen** - intended for PDF files that will be displayed on the World Wide Web or an intranet, or that will be distributed through an e-mail system for on-screen viewing. This set of options uses compression, downsampling, and a relatively low resolution; converts all colors to RGB; maintains compatibility with Acrobat 3.0; to create a PDF file that is as small as possible. It also optimizes files for byte serving (fast web view).

• **Printer** - to be used for PDF files that are intended for desktop printers, digital copiers, publishing on a CD-ROM, or to send to a client as a publishing proof. In this set of options, file size is still important, but it is not the only objective. This set of options uses compression and downsampling to keep the file size down, but it also embeds subsets of all fonts used in the file, tags everything for color management, and prints to a medium resolution to create a reasonably accurate rendition of the original document.

• **Prepress** - intended for PDF files that will be printed as a high-quality final output to an imagesetter or, for example, a platesetter. In this case, file size is not a consideration. The objective is to maintain all the information in a PDF file that a commercial printer or service bureau will need to print the document correctly. This set of options downsamples color and grayscale images at 300 dpi, monochrome images at 1200 dpi, embeds subsets of all fonts used in the file, prints to a higher resolution, and uses other settings to preserve the maximum amount of information about the original document.

Syntax

NVDC.setParameter("OptimizePDFfor", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

DocumentResolution

Sets the output document resolution.

Possible values: "10"..."2400".

Syntax

NVDC.setParameter("DocumentResolution", value)

Data Type: String

Note: Can only be set before calling the Create method.

DocumentPaperSize

Sets the output document paper size. Use x to separate width and height. For example **10inx20in** will specify an output paper size with 10 inches in width and 20 inches in height.

Syntax

NVDC.setParameter("DocumentPaperSize", value)

Example - set the output document paper size to 8x11 inches

NVDC.setParameter("DocumentPaperSize", "8inx11in")

Example - set the output document paper size to 100x200 millimeters

NVDC.setParameter("DocumentPaperSize", "100mmx200mm")

Example - set the output document paper size to 3x5 centimeters

NVDC.setParameter("DocumentPaperSize", "3cmx5cm")

Data Type: String

Note: Can only be set before calling the Create method.

ScalePage

Specifies whether docCreator should scale the output document. If ScalePage is "false" output document will be stripped to DocumentPaperSize, if ScalePage is "true" output document will be scaled to DocumentPaperSize.

Syntax

NVDC.setParameter("ScalePage", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "PDF".

ConstrainProportions

Specifies whether docCreator should constrain proportions when scaling the output document.
Possible values: "true", "false"

Syntax

NVDC.setParameter("ConstrainProportions", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "PDF" and ScalePage is "true".

PlaceContentIn

Specifies where docCreator should place the original page content on the scaled page.
Possible values: "0" (left-bottom corner), "1" (right-bottom corner), "2" (center), "3" (center-top),
"4" (center-bottom), "5" (left-center), "6" (right-center), "7" (left-top corner),
"8" (right-top corner)

Syntax

NVDC.setParameter("PlaceContentIn", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is "PDF".

PDFVersion

Sets the compatibility level (a.k.a. PDF version) of the output PDF document.

Possible values: "1.3" (Acrobat 4-and-later compatible)
"1.4" (Acrobat 5-and-later compatible)
"1.5" (Acrobat 6-and-later compatible)
"1.6" (Acrobat 7-and-later compatible)
"1.7" (Acrobat 8-and-later compatible)
"2.0" (Acrobat 2.0 compatible)

Syntax

NVDC.setParameter("PDFVersion", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

DocumentTitle

Sets the title field in the output PDF document.

Syntax

NVDC.setParameter("DocumentTitle", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

DocumentSubject

Sets the subject field in the output PDF document.

Syntax

NVDC.setParameter("DocumentSubject", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

DocumentAuthor

Sets the author field in the output PDF document.

Syntax

NVDC.setParameter("DocumentAuthor", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

DocumentKeywords

Sets the keywords field in the output PDF document.

Syntax

NVDC.setParameter("DocumentKeywords", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFAutoRotatePage

Specifies whether docCreator should automatically rotate pages based on the orientation of the text.
Possible values: "None" (will disable the Auto-Rotate Pages option)
"PageByPage" (will rotate each page based on the direction of the text on that page)
"All" (will rotate all pages in the document based on the orientation of the majority of text)

Syntax

NVDC.setParameter("PDFAutoRotatePage", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFCompressPages

Specifies whether text and line art in the output PDF document should be compressed.
Possible values: "true", "false"

Syntax

NVDC.setParameter("PDFCompressPages", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFEmbedAllFonts

Specifies whether fonts in the output PDF document should be embedded.
Possible values: "true", "false"

Syntax

NVDC.setParameter("PDFEmbedAllFonts", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFSubsetFonts

Indicates whether to include in the output PDF document only the font characters that are used in the original document.
Possible values: "true", "false"

Syntax

NVDC.setParameter("PDFSubsetFonts", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFFontsMaxSubset

Sets the Subset Fonts threshold. If the percentage of used characters (compared with total characters of the particular font) exceeds this threshold, the entire font is embedded.
Possible values: "0"..."100".

Syntax

NVDC.setParameter("PDFFontsMaxSubset", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFProcessColorModel

Sets the color model for the output PDF document.
Possible values: "DeviceRGB", "DeviceCMYK", "DeviceGRAY", "LeaveUnchanged"

Syntax

NVDC.setParameter("PDFProcessColorModel", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

CompressColorImages

Specifies whether the color images in the output PDF document should be compressed.
Possible values: "true", "false"

Syntax

NVDC.setParameter("CompressColorImages", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

ColorCompressMethod

Sets the compression method for color images in the output PDF document.
Possible values: "Automatic", "JPEG-maximum", "JPEG-high", "JPEG-medium", "JPEG-low", "JPEG-minimum", "ZIP".

Syntax

NVDC.setParameter("ColorCompressMethod", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

CompressGrayImages

Specifies whether gray images in the output PDF document should be compressed.
Possible values: "true", "false"

Syntax

NVDC.setParameter("CompressGrayImages", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

GrayCompressMethod

Sets the compression method for gray images in the output PDF document.
Possible values: "Automatic", "JPEG-maximum", "JPEG-high", "JPEG-medium", "JPEG-low", "JPEG-minimum", "ZIP".

Syntax

NVDC.setParameter("GrayCompressMethod", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

CompressMonoImages

Specifies whether monochrome images in the output PDF document should be compressed.
Possible values: "true", "false"

Syntax

NVDC.setParameter("CompressMonoImages", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

MonoCompressMethod

Sets the compression method for monochrome images in the output PDF document.
Possible values:

"CCITT" (compress monochrome images using the CCITT group 4-fax compression)
"ZIP" (compress monochrome images using ZIP-compatible compression)

Syntax

NVDC.setParameter("MonoCompressMethod", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

ColorImageResolution

Sets the resolution level for color images in the output PDF document.
Possible values: "10"..."2400".

Syntax

NVDC.setParameter("ColorImageResolution", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

GrayImageResolution

Sets the resolution level for gray images in the output PDF document.
Possible values: "10"..."2400".

Syntax

NVDC.setParameter("GrayImageResolution", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

MonoImageResolution

Sets the resolution level for mono images in the output PDF document.
Possible values: "10"..."2400".

Syntax

NVDC.setParameter("MonoImageResolution", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

DownsampleColorImages

Specifies whether color images in the output PDF document should be downsampled.
Possible values: "true", "false"

Syntax

NVDC.setParameter("DownsampleColorImages", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

ColorImageDownsampleType

Possible values: "Bicubic", "Average", "Subsample".

Syntax

NVDC.setParameter("ColorImageDownsampleType", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

DownsampleGrayImages

Specifies whether gray images in the output PDF document should be downsampled.
Possible values: "true", "false"

Syntax

NVDC.setParameter("DownsampleGrayImages", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

GrayImageDownsampleType

Possible values: "Bicubic", "Average", "Subsample".

Syntax

NVDC.setParameter("GrayImageDownsampleType", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

DownsampleMonoImages

Specifies whether monochrome images in the output PDF document should be downsampled.
Possible values: "true", "false"

Syntax

NVDC.setParameter("DownsampleMonoImages", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

MonoImageDownsampleType

Possible values: "Bicubic", "Average", "Subsample".

Syntax

NVDC.setParameter("MonoImageDownsampleType", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

MaxInlineImageSize

Sets the maximum size of an inline image in bytes.
Default value: "4000"

Syntax

NVDC.setParameter("MaxInlineImageSize", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if DocumentOutputFormat is “PDF”. For images larger than this size, docCreator will create a XObject instead of embedding the image into the context stream. Note that redundant inline images must be embedded each time they occur in the document, while multiple references can be made to a single XObject image. Therefore it may be advantageous to set a small or zero value if the source document is expected to contain multiple identical images, reducing the size of the generated PDF.

ParseDSCComments

Intructs the conversion engine whether to parse PS/EPS DSC comments.
Possible values: "true", "false"

Syntax

NVDC.setParameter("ParseDSCComments", value)

Data Type: String

Note: Can only be set before calling the Create method.

DefaultRenderingIntent

Sets the default rendering intent.
Possible values: "0" (Default)
"1" (Perceptual)
"2" (Saturation)
"3" (RelativeColorimetric)
"4" (AbsoluteColorimetric)

Syntax

NVDC.setParameter("DefaultRenderingIntent", value)

Data Type: String

Note: Can only be set before calling the Create method.

PreserveOverprintSettings

Specifies whether overprint settings should be preserved.
Possible values: "true", "false"

Syntax

NVDC.setParameter("PreserveOverprintSettings", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

UCRandBGInfo

Specifies whether under color removal and black generation settings should be preserved.
Possible values: "true", "false"

Syntax

NVDC.setParameter("UCRandBGInfo", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

TransferFunctionInfo

Specifies whether transfer function information should be preserved.
Possible values: "true", "false"

Syntax

NVDC.setParameter("TransferFunctionInfo", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PreserveHaltoneInfo

Specifies whether halftone information should be preserved.
Possible values: "true", "false"

Syntax

NVDC.setParameter("PreserveHalftoneInfo", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

Conversion parameters - PDF Encryption

PDFEncryption

Specifies whether the output PDF document should be encrypted.
Possible values: "true", "false"

Syntax

NVDC.setParameter("PDFEncryption", value)

Data Type: String
Note: Can only be set before calling the Create method.

PDFEncryptionMethod

Sets the encryption method.
Possible values: "40" (low - 40 bits encryption, Acrobat 3-and-later compatible)
"rc4" (high - 128 bits encryption, Acrobat 5-and-later compatible)
"aes" (high - 128 bits encryption, Acrobat 6-and-later compatible)
"aes256" (high - 256 bits encryption, Acrobat 9-and-later compatible)

Syntax

NVDC.setParameter("PDFEncryptionMethod", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFEncryptMeta

Specifies whether metadata in the output PDF file should be encrypted.
Possible values: "true", "false"

Syntax

NVDC.setParameter("PDFEncryptMeta", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFUserPassword

Sets the user password in the output PDF document. Users will be asked to enter this password before Acrobat Reader allows them to view the PDF document.

Syntax

NVDC.setParameter("PDFUserPassword", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFOwnerPassword

Sets the output document owner password. This option will force the user of the PDF to enter a password before Acrobat Reader allows them to change the user password and security permissions.

Syntax

NVDC.setParameter("PDFOwnerPassword", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PDFPermissions

Sets PDF security permissions to use for encrypting output file.

Possible values:

"p" - document printing is denied
"c" - changing the document is denied
"s" - selection and copying of text and graphics is denied
"a" - adding or changing annotations (PDF comments) or form fields is denied

The following flags are defined for 128 bits and higher encryption:

"i" - disables editing of form fields
"e" - disables extraction of text and graphics
"d" - disables document assembly
"q" - disables high quality printing

Syntax

NVDC.setParameter("PDFPermissions", value)

Example

NVDC.setParameter("PDFPermissions", "p")

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

Conversion parameters - PDF viewer options

OpenAtPage

Specifies the page at which the output PDF document should open in PDF viewer.

Syntax

NVDC.setParameter("OpenAtpage", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

OpenMagnification

Specify the open magnification in % for the output document.

Possible values: "0" (Default)
"1" (Actual Size)
"2" (Fit Page)
"3" (Fit Width)
"4" (Fit Height)
"5" (Fit Visible)
"6" (Zoom 25%)
"7" (Zoom 50%)
"8" (Zoom 75%)
"9" (Zoom 100%)
"10" (Zoom 125%)
"11" (Zoom 150%)
"12" (Zoom 200%)
"13" (Zoom 400%)
"14" (Zoom 800%)
"15" (Zoom 1600%)
"16" (Zoom 2400%)
"17" (Zoom 3200%)
"18" (Zoom 6400%)

Syntax

NVDC.setParameter("OpenMagnification", value)

Data Type: String

Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

FullScreen

Specifies whether Acrobat Reader should maximize the document window and display output PDF documents without the menu bar, toolbar, or window controls.

Possible values: "true", "false"

Syntax

NVDC.setParameter("FullScreen", value)

Data Type: String

Note: take into account that if you hide the menu bar and toolbars users cannot apply commands and select tools unless they know the keyboard shortcuts.

PageMode
Specifies how output file should be displayed when opened in PDF viewer.
Possible values: "**0**" Default view
"**1**" Page only
"**2**" Outlines (bookmarks) visible
"**3**" Thumbnail images visible
"**4**" Optional content group (layers) panel visible
"**5**" Attachments panel visible

Syntax

NVDC.setParameter("PageMode", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

PageLayout
Specifies page layout to use when output file is opened in PDF viewer.
Possible values: "1" (Display one page at a time)
"2" (Display the pages in one column)
"3" (Display the pages in two columns, with odd numbered pages on the left)
"4" (Display the pages in two columns, with odd numbered pages on the right)
"5" (Display the pages two at a time, with odd numbered pages on the left)
"6" (Display the pages two at a time, with odd numbered pages on the right)

Syntax

NVDC.setParameter("PageLayout", value)

Data Type: String
Note: Can only be set before calling the Create method. Will have effect only if "DocumentOutputFormat" is "PDF".

HideMenuBar
Specifies whether Acrobat Reader should hide the menu bar when displaying the output PDF document.
Possible values: "true", "false"

Syntax

NVDC.setParameter("HideMenuBar", value)

Data Type: String

HideToolbar
Specifies whether Acrobat Reader should hide the toolbar when displaying PDF documents.
Possible values: "true", "false"

Syntax

NVDC.setParameter("HideToolbar", value)

Data Type: String

HideWindowUI
Specifies whether Acrobat Reader should hide the user interface when displaying PDF documents.
Possible values: "true", "false"

Syntax

NVDC.setParameter("HideWindowUI", value)

Data Type: String

FitWindow

Specifies whether Acrobat Reader should adjust the document window to fit snugly around the opening page when displaying PDF documents.
Possible values: "true", "false"

Syntax

NVDC.setParameter("FitWindow", value)

Data Type: String

CenterWindow

Specifies whether Acrobat Reader should position the window in the center of the screen area when displaying PDF documents.
Possible values: "true", "false"

Syntax

NVDC.setParameter("CenterWindow", value)

Data Type: String

Conversion parameters - Watermark/Stationery

StampText

Specifies the text to use as watermark. (See Appendix B for the list of supported variables)

Syntax

NVDC.setParameter("StampText", value)

Data Type: String

StampFile

Specifies the file to load stamp commands from. (See Appendix A for stamp file format info)

Syntax

NVDC.setParameter("StampFile", value)

Data Type: String

Example

NVDC.setParameter("StampFile", "c:\stamp.nsp")

Commands contained in stamp.nsp:

```
\stamp 0
\stampname SampleStamp
\text CONFIDENTIAL
\x center
\y center
\units 1
\fontname Arial
\fontcolor #00FF00
```

StampFontColor

Specifies the watermark font color.

Syntax

NVDC.setParameter("StampFontColor", value)

Example

NVDC.setParameter("StampFontColor", "#000000")

Data Type: String

StampFontName

Specifies the watermark font name.

Syntax

NVDC.setParameter("StampFontName", value)

Example

NVDC.setParameter("StampFontName", "Arial")

Data Type: String

StampFontSize

Specifies the watermark font size.

Syntax

NVDC.setParameter("StampFontSize", value)

Example

NVDC.setParameter("StampFontSize", "40")

Data Type: String

StampFontEmbed

Specifies whether fonts should be embedded.
Possible values: "true", "false"

Syntax

NVDC.setParameter("StampFontEmbed", value)

Data Type: String

StampFontSubset

Specifies whether fonts should be subset.
Possible values: "true", "false"

Syntax

NVDC.setParameter("StampFontSubset", value)

Data Type: String

StampTextRenderingMode

Specifies the text rendering mode.
Possible values:

"0" - Fill text, no stroke (default)
"1" - Stroke text, no fill
"2" - Fill then Stroke text
"3" - Invisible

Fill text, no stroke 

Stroke text, no fill 

Fill then stroke text 

Syntax

NVDC.setParameter("StampTextRenderingMode", value)

Data Type: String

StampFontEncoding

Specifies the font encoding.

Syntax

NVDC.setParameter("StampFontEncoding", value)

Data Type: String

StampScale

Specifies by how much (in percent) to scale the stamp. Default value: "100"

Syntax

NVDC.setParameter("StampScale", value)

Example

NVDC.setParameter("StampScale", "200")

Data Type: String

Note: will have effect only for Image and PDFOverlay stamps.

StampFontColor

Specifies font color in RGB colorspace.

Syntax

NVDC.setParameter("StampFontColor", value)

Example

NVDC.setParameter("StampFontColor", "#FF0000") (red)

Data Type: String

StampFontColorGray

Specifies font color in Gray colorspace.

Syntax

NVDC.setParameter("StampFontColorGray", value)

Example

NVDC.setParameter("StampFontColorGray", "#77")

Data Type: String

StampFontColorCMYK

Specifies font color in CMYK colorspace.

Syntax

NVDC.setParameter("StampFontColorCMYK", value)

Example

NVDC.setParameter("StampFontColorCMYK", "#000000FF") (black)

Data Type: String

StampStrokeColor

Specifies font stroke color in RGB colorspace.

Syntax

NVDC.setParameter("StampStrokeColor", value)

Example

NVDC.setParameter("StampStrokeColor", "#FF0000") (red)

Data Type: String

StampStrokeColorGray

Specifies font stroke color in Gray colorspace.

Syntax

NVDC.setParameter("StampStrokeColorGray", value)

Example

NVDC.setParameter("StampStrokeColorGray", "#77")

Data Type: String

StampStrokeColorCMYK

Specifies font stroke color in CMYK colorspace.

Syntax

NVDC.setParameter("StampStrokeColorCMYK", value)

Example

NVDC.setParameter("StampStrokeColorCMYK", "#000000FF") (black)

Data Type: String

StampStrokeWidth

Specifies the stroke width.

Syntax

NVDC.setParameter("StampStrokeWidth", value)

Data Type: String

StampRotate

Specifies the watermark rotation angle in degrees.

Syntax

NVDC.setParameter("StampRotate", value)

Data Type: String

StampOpacity

Specifies watermark opacity (transparency) level.
Possible values: "0" … "100". Default value: "100"

Syntax

NVDC.setParameter("StampOpacity", value)

Data Type: String

PlaceStampOnPages

Specifies the pages to place watermark/stationery on.
Note: page numbers must be separated by commas. To place the watermark/stationery on all pages specify "0".

Syntax

NVDC.setParameter("PlaceStampOnPages", value)

Example

NVDC.setParameter("PlaceStampOnPages", "1,3,7")
(will place watermark on pages 1, 3 and 7)

Data Type: String

StampUnits

Specifies measurement unit to use for the StampX and StampY properties.
Possible values: "0" (points), "1" (inches), "2" (centimeters), "3" (millimeters)

Syntax

NVDC.setParameter("StampUnits", value)

Data Type: String

StampX

Specifies the watermark X coordinate.

Syntax

NVDC.setParameter("StampX", value)

Data Type: String
Note: you can use as values, instead of numbers "**center**", "**left**" and "**right**" - as the name suggests they will position the watermark in the center of the page, on the left or right.

StampY

Specifies the watermark Y coordinate.

Syntax

NVDC.setParameter("StampY", value)

Data Type: String
Note: you can use as values, instead of numbers "**center**", "**top**" and "**bottom**" - as the name suggests they will position the watermark in the center of the page, on top or in the bottom.

StampWidth

Specifies the stamp width.

Syntax

NVDC.setParameter("StampWidth", value)

Data Type: String

Note: will have effect only for TextBox and Image stamps.

StampHeight

Specifies the stamp height.

Syntax

NVDC.setParameter("StampHeight", value)

Data Type: String

Note: will have effect only for TextBox and Image stamps.

StampTextBox

Specifies the watermark text to use inside a text box. (See Appendix B for the list of supported variables)

Syntax

NVDC.setParameter("StampTextBox", value)

Data Type: String

StampTextAlign

Specifies how the text to use inside a textbox stamp should be aligned.
Possbile values: "0" (left), "1" (center), "2" (right)

Syntax

NVDC.setParameter("StampTextAlign", value)

Data Type: String

StampWordWrap

Specifies whether to enable Word Wrap in case text does not fit in one line.
Possbile values: "true", "false"

Syntax

NVDC.setParameter("StampWordWrap", value)

Data Type: String

StampWebLink

Specifies the web address to go to when the stamp is clicked.

Syntax

NVDC.setParameter("StampWebLink", value)

Data Type: String

StampGoToPage

Specifies the page number to go to when the stamp is clicked.

Syntax

NVDC.setParameter("StampGoToPage", value)

Data Type: String

StampUseCropBox

Specifies whether to use the page crop box to position stamp.
Possbile values: "true", "false"

Syntax

NVDC.setParameter("StampUseCropBox", value)

Data Type: String

StampUsePageRotation

Specifies whether to use the page rotation parameter when placing stamp.
Possbile values: "true", "false"

Syntax

NVDC.setParameter("StampUsePageRotation", value)

Data Type: String

StampPlaceAs

Specifies how to place stamp on page.
Possible values: "0" as stamp: over page content (Default),
"1" as watermark: under page content

Syntax

NVDC.setParameter("StampPlaceAs", value)

Data Type: String

StampImage

Specifies the image file to use as stamp. Supported formats: GIF, PNG, JPEG, TIFF and BMP.

Syntax

NVDC.setParameter("StampImage", value)

Example

NVDC.setParameter("StampImage", "c:\image.gif")

Data Type: String

StampPDFOverlay

Specifies the PDF file to use as stationery.

Syntax

NVDC.setParameter("StampPDFOverlay", value)

Example

NVDC.setParameter("StampPDFOverlay", "c:\stationery.pdf")

Data Type: String

StampPDFOverlayPage

Specifies the page to use as overlay from the PDFOverlay file. Default value: "1"

Syntax

NVDC.setParameter("StampPDFOverlayPage", value)

Data Type: String

Conversion parameters - mergePDF / mergeMultiplePDF related

CreatePageBookmarks

Specifies whether bookmarks should be created during merging.
Possible values: "true", "false"

Syntax

NVDC.setParameter("CreatePageBookmarks", value)

Data Type: String
Note: Will have effect only when used in conjunction with the mergePDF method.

CreateNewBookmarks

Specifies whether new bookmarks should be created during merging.
Possible values: "true", "false"

Syntax

NVDC.setParameter("CreateNewBookmarks", value)

Data Type: String
Note: Will have effect only when used in conjunction with mergePDF method.

BookmarksFile

Specifies path to the file that contains the new bookmarks to add while merging files.

Syntax

NVDC.setParameter("BookmarksFile", value)

Data Type: String
Note: Will have effect only when used in conjunction with mergePDF method.

Conversion parameters - splitPDF related

SplitByBookmarks

Specifies whether PDF files should be split by bookmarks.
Possible values: "true", "false"

Syntax

NVDC.setParameter("SplitByBookmarks", value)

Data Type: String
Note: Will have effect only when used in conjunction with splitPDF method.

BkLevel

Specifies the lowest bookmark level to split by.
Default value: "1"

Syntax

NVDC.setParameter("BkLevel", value)

Data Type: String
Note: Will have effect only when used in conjunction with splitPDF method and SplitByBookmarks = true.

NameByBk

Specifies whether to name output PDF file(s) according to bookmark titles.
Possible values: "true", "false"

Syntax

NVDC.setParameter("nameByBk", value)

Data Type: String
Note: Will have effect only when used in conjunction with splitPDF method and SplitByBookmarks = true.

Conversion parameters - mergePDF, mergeMultiplePDF, splitPDF related

RemoveAnnotations

Specifies whether to remove text annotations from the output PDF file(s).
Possible values: "true", "false"

Syntax

NVDC.setParameter("RemoveAnnotations", value)

Data Type: String
Note: Will have effect only when used in conjunction with splitPDF or mergePDF methods.

RemoveAcroForms

Specifies whether to remove PDF forms from the output PDF file(s).
Possible values: "true", "false"

Syntax

NVDC.setParameter("RemoveAcroForms", value)

Data Type: String
Note: Will have effect only when used in conjunction with splitPDF or mergePDF methods.

RemovePageLabels

Specifies whether to remove page labels from the output PDF file(s).
Possible values: "true", "false"

Syntax

NVDC.setParameter("RemovePageLabels", value)

Data Type: String
Note: Will have effect only when used in conjunction with splitPDF or mergePDF methods.

RemoveLayers

Specifies whether to remove layers from the output PDF file(s).
Possible values: "true", "false"

Syntax

NVDC.setParameter("RemoveLayers", value)

Data Type: String
Note: Will have effect only when used in conjunction with splitPDF or mergePDF methods.

RemoveArticleThreads

Specifies whether to remove article threads from the output PDF file(s).
Possible values: "true", "false"

Syntax

NVDC.setParameter("RemoveArticleThreads", value)

Data Type: String
Note: Will have effect only when used in conjunction with splitPDF or mergePDF methods.

Conversion parameters - convertImage related

ImageRotate

Specifies how docCreator should handle image rotation/orientation.
Possible Values: "0" (preserve original), "1" (rotate landscape images 90 degrees), "2" (rotate landscape images -90 degrees), "3" (rotate portrait images 90 degrees), "4" (rotate portrait images -90 degrees)

Syntax

NVDC.setParameter("ImageRotate", value)

Data Type: String
Note: Will have effect only when used in conjunction with convertImage method.

OCR

Specifies whether docCreator should perform OCR (Optical Character Recognition) when converting images to PDF.
Possible values: "true", "false"

Syntax

NVDC.setParameter("OCR", value)

Data Type: String
Note: Will have effect only when used in conjunction with convertImage method.

OCRlang

Sets the OCR language.
Possible Values: "0" English, "1" Czech, "2" Danish, "3" Dutch, "4" Finnish, "5" French, "6" German, "7" Greek, "8" Hungarian, "9" Italian, "10" Japanese, "11" Korean, "12" Norwegian, "13" Polish, "14" Portuguese, "15" Russian, "16" Spanish, "17" Swedish, "18" Turkish, "19" Chinese Traditional, "20" Chinese Simplified

Syntax

NVDC.setParameter("OCRlang", value)

Data Type: String
Note: Will have effect only when used in conjunction with convertImage method and "OCR" = "true".

AutoRotate

Specifies whether the OCR engine should attempt to determine the orientation of the page.
Possible Values: "true", "false"

Syntax

NVDC.setParameter("AutoRotate", value)

Data Type: String
Note: Will have effect only when used in conjunction with convertImage method and "OCR" = "true".

AutoStraighten

Specifies whether the OCR engine should attempt to "de-skew" the page to correct for small angles of misalignment from the vertical.
Possible Values: "true", "false"

Syntax

NVDC.setParameter("AutoStraighten", value)

Data Type: String
Note: Will have effect only when used in conjunction with convertImage method and "OCR" = "true".

ConvertImageParam

Specifies additional conversion parameters for convertImage method.

Syntax

NVDC.setParameter("AutoStraighten", value)

Data Type: String
Note: Will have effect only when used in conjunction with convertImage method.

Possible Values:

-useimagepack forces docCreator to use Image Pack for all image conversions - by default docCreator will use the built-in image parser for jpeg, png, bmp, pcx and tiff images and Image Pack for the rest.

-colorize <value> colorize the image with the fill color
Specify the amount of colorization as a percentage. You can apply separate colorization values to the red, green, and blue channels of the image with a colorization value list delimited with slashes (e.g. 0/0/50).

-colors <value> preferred number of colors in the image
The actual number of colors in the image may be less than your request, but never more. Note, this is a color reduction option. Images with less unique colors than specified with this option will have any duplicate or unused colors removed. The ordering of an existing color palette may be altered. When converting an image from color to grayscale, convert the image to the gray colorspace before reducing the number of colors since doing so is most efficient.
Note: -dither, -colorspace affect the color reduction algorithm.

-colorspace <value> colorspace type
Choices are: CMYK, GRAY, HSL, HWB, OHTA, RGB, Transparent, XYZ, YCbCr, YIQ, YPbPr, or YUV.
Color reduction, by default, takes place in the RGB color space. Empirical evidence suggests that distances in color spaces such as YUV or YIQ correspond to perceptual color differences more closely than do distances in RGB space. These color spaces may give better results when color reducing an image. Refer to quantize for more details.

The Transparent color space behaves uniquely in that it preserves the matte channel of the image if it exists. The -colors or -monochrome option, or saving to a file format which requires color reduction, is required for this option to take effect.

-compress <type> image compression type
Choices are: None, BZip, Fax, Group4, JPEG, LZW, RLE or Zip.
Specify +compress to store the binary image in an uncompressed format. The default is the compression type of the specified image file.

-contrast enhance or reduce the image contrast
This option enhances the intensity differences between the lighter and darker elements of the image. Use -contrast to enhance the image or +contrast to reduce the image contrast. For a more pronounced effect you can repeat the option, for Ex: **NVDC.setParameter("ConvertImageParam", "-contrast -contrast")**

-crop <width>x<height>{+-}<x>{+-}<y>{%} preferred size and location of the cropped image
See -geometry for details about the geometry specification.

The width and height give the size of the image that remains after cropping, and x and y are offsets that give the location of the top left corner of the cropped image with respect to the original image. To specify the amount to be removed, use -shave instead.

If the x and y offsets are present, a single image is generated, consisting of the pixels from the cropping region. The offsets specify the location of the upper left corner of the cropping region measured downward and rightward with respect to the upper left corner of the image.

If the x and y offsets are omitted, a set of tiles of the specified geometry, covering the entire input image, is generated. The rightmost tiles and the bottom tiles are smaller if the specified geometry extends beyond the dimensions of the input image

-dither apply Floyd/Steinberg error diffusion to the image
The basic strategy of dithering is to trade intensity resolution for spatial resolution by averaging the intensities of several neighboring pixels. Images which suffer from severe contouring when reducing colors can be improved with this option.
The -colors or -monochrome option is required for this option to take effect.

-density <width>x<height> horizontal and vertical resolution in pixels of the image
This option specifies the image resolution to store while encoding a raster image. The default unit of measure is in dots per inch (DPI).

The default resolution is 72 dots per inch, which is equivalent to one point per pixel (Macintosh and Postscript standard). Computer screens are normally 72 or 96 dots per inch while printers typically support 150, 300, 600, or 1200 dots per inch. To determine the resolution of your display, use a ruler to measure the width of your screen in inches, and divide by the number of horizontal pixels (1024 on a 1024x768 display).

If the file format supports it, this option may be used to update the stored image resolution. Note that Photoshop stores and obtains image resolution from a proprietary embedded profile. If this profile is not stripped from the image, then Photoshop will continue to treat the image using its former resolution, ignoring the image resolution specified in the standard file header.

The density option is an attribute and does not alter the underlying raster image. It may be used to adjust the rendered size for desktop publishing purposes by adjusting the scale applied to the pixels. To resize the image so that it is the same size at a different resolution, use the -resample option.

-depth <value> depth of the image
This is the number of bits in a color sample within a pixel. The only acceptable values are 8 or 16. Use this option to specify the depth of raw images whose depth is unknown such as GRAY, RGB, or CMYK, or to change the depth of any image after it has been read.

-filter <type> use this type of filter when resizing an image
Use this option to affect the resizing operation of an image (see -geometry). Choose from these filters: Point, Box, Triangle, Hermite, Hanning, Hamming, Blackman, Gaussian, Quadratic, Cubic, Catrom, Mitchell, Lanczos, Bessel and Sinc.
The default filter is automatically selected to provide the best quality while consuming a reasonable amount of time. The Mitchell filter is used if the image supports a palette, supports a matte channel, or is being enlarged, otherwise the Lanczos filter is used.

-geometry <width>x<height>{+-}<x>{+-}<y>{%}{@}{!}{<}{>} preferred size and location of the Image
By default, the width and height are maximum values. That is, the image is expanded or contracted to fit the width and height value while maintaining the aspect ratio of the image. Append an exclamation point to the geometry to force the image size to exactly the size you specify. For example, if you specify 640x480! the image width is set to 640 pixels and height to 480.

If only the width is specified, the width assumes the value and the height is chosen to maintain the aspect ratio of the image. Similarly, if only the height is specified (e.g., -geometry x256), the width is chosen to maintain the aspect ratio.

To specify a percentage width or height instead, append %. The image size is multiplied by the width and height percentages to obtain the final image dimensions. To increase the size of an image, use a value greater than 100 (e.g. 125%). To decrease an image's size, use a percentage less than 100.

Use @ to specify the maximum area in pixels of an image.

Use > to change the dimensions of the image only if its width or height exceeds the geometry specification. < resizes the image only if both of its dimensions are less than the geometry specification. For example, if you specify '640x480>' and the image size is 256x256, the image size does not change. However, if the image is 512x512 or 1024x1024, it is resized to 480x480. Enclose the geometry specification in quotation marks to prevent the < or > from being interpreted by your shell as a file redirection.

-monochrome transform the image to black and white

-page <width>x<height>{+-}<x>{+-}<y>{%}{!}{<}{>} size and location of an image canvas
For convenience you can specify the page size by media (e.g. A4, Ledger, etc.). Otherwise, -page behaves much like -geometry (e.g. -page letter+43+43>).

This option is also used to place subimages when writing to a multi-image format that supports offsets, such as GIF89 and MNG. When used for this purpose the offsets are always measured from the top left corner of the canvas and are not affected by the -gravity option. To position a GIF or MNG image, use -page{+-}<x>{+-}<y> (e.g. -page +100+200). When writing to a MNG file, a -page option appearing ahead of the first image in the sequence with nonzero width and height defines the width and height values that are written in the MHDR chunk. Otherwise, the MNG width and height are computed from the bounding box that contains all images in the sequence. When writing a GIF89 file, only the bounding box method is used to determine its dimensions.

For a PostScript page, the image is sized as in -geometry and positioned relative to the lower left hand corner of the page by {+-}<xoffset>{+-}<y offset>. Use -page 612x792>, for example, to center the image within the page. If the image size exceeds the PostScript page, it is reduced to fit the page. The default page dimensions for a TEXT image is 612x792.

This option is used in concert with -density.

Use +page to remove the page settings for an image.

-quality <value> JPEG/MIFF/PNG compression level
For the JPEG image format, quality is 0 (lowest image quality and highest compression) to 100 (best quality but least effective compression). The default quality is 75. Use the -sampling-factor option to specify the factors for chroma downsampling.

For the MIFF image format, quality/10 is the zlib compression level, which is 0 (worst but fastest compression) to 9 (best but slowest). It has no effect on the image appearance, since the compression is always lossless.

For the JPEG-2000 image format, quality is mapped using a non-linear equation to the compression ratio required by the Jasper library. This non-linear equation is intended to loosely approximate the quality provided by the JPEG v1 format. The default quality value 75 results in a request for 16:1 compression. The quality value 100 results in a request for non-lossy compression.

For the MNG and PNG image formats, the quality value sets the zlib compression level (quality / 10) and filter-type (quality % 10). Compression levels range from 0 (fastest compression) to 100 (best but slowest). For compression level 0, the Huffman-only strategy is used, which is fastest but not necessarily the worst compression.

The default is quality is 75, which means nearly the best compression with adaptive filtering. The quality setting has no effect on the appearance of PNG and MNG images, since the compression is always lossless.

For further information, see the PNG specification.

When writing a JNG image with transparency, two quality values are required, one for the main image and one for the grayscale image that conveys the alpha channel. These are written as a single integer equal to the main image quality plus 1000 times the opacity quality. For example, if you want to use quality 75 for the main image and quality 90 to compress the opacity data, use -quality 90075.

-region <width>x<height>{+-}<x>{+-}<y> apply options to a portion of the image

-resample <horizontal>x<vertical> resample image to specified horizontal and vertical resolution
Resize the image so that its rendered size remains the same as the original at the specified target resolution. For example, if a 300 DPI image renders at 3 inches by 2 inches on a 300 DPI device, when the image has been resampled to 72 DPI, it will render at 3 inches by 2 inches on a 72 DPI device. Note that only a small number of image formats (e.g. JPEG, PNG, and TIFF) are capable of storing the image resolution. For formats which do not support an image resolution, the original resolution of the image must be specified via -density on the command line prior to specifying the resample resolution.

Note that Photoshop stores and obtains image resolution from a proprietary embedded profile. If this profile exists in the image, then Photoshop will continue to treat the image using its former resolution, ignoring the image resolution specified in the standard file header.

-rotate <degrees>{<}{>} apply Paeth image rotation to the image
Use > to rotate the image only if its width exceeds the height. < rotates the image only if its width is less than the height. For example, if you specify -rotate "-90>" and the image size is 480x640, the image is not rotated. However, if the image is 640x480, it is rotated by -90 degrees. If you use > or <, enclose it in quotation marks to prevent it from being misinterpreted as a file redirection.

-sampling-factor <horizontal_factor>x<vertical_factor> sampling factors used by JPEG encoder and YUV decoder/encoder.

This option specifies the sampling factors to be used by the JPEG encoder for chroma downsampling. If this option is omitted, the JPEG library will use its own default values. When reading or writing the YUV format, use -sampling-factor 2x1 to specify the 4:2:2 downsampling method.

-scale <geometry> scale the image
See -geometry for details about the geometry specification. -scale uses a simpler, faster algorithm. Offsets, if present in the geometry string, are ignored.

-sharpen <radius>{x<sigma>} sharpen the image
Use a Gaussian operator of the given radius and standard deviation (sigma).

-shave <width>x<height>{%} shave pixels from the image edges
Specify the width of the region to be removed from both sides of the image and the height of the regions to be removed from top and bottom.

-trim trim an image
This option removes any edges that are exactly the same color as the corner pixels. Use -fuzz to make trim remove edges that are nearly the same color as the corner pixels.
-fuzz <distance>{%} colors within this distance are considered equal

A number of algorithms search for a target color. By default the color must be exact. Use this option to match colors that are close to the target color in RGB space. For example, if you want to automatically trim the edges of an image with -trim but the image was scanned and the target background color may differ by a small amount. This option can account for these differences.

The distance can be in absolute intensity units or, by appending "%", as a percentage of the maximum possible intensity (255, 65535, or 4294967295).

Appendix A: Stamp File Format

Stamp file contains commands and text to be stamped into the PDF file. Commands are used to set certain attributes (like font, color etc).
Rules:

- Every command begins with a backslash (\) (there is no space after that);
- Case sensitivity is not relevant;
- Command and parameters are separated by spaces.

\stamp <value> - starts a stamp. <value> specifies the pages to stamp (leave blank if you want to stamp all the pages.

\stampname <value> - specifies the stamp name
\text <value> - specifies the text to stamp

\starttextbox <value> - Starts a Text Box (<value> is the text alignment: 1 - left, 2 - right, 3 - center)
.. text box text to stamp …
\endtextbox

\textrenderingmode <value> - specifies text rendering mode

\textgraycolor <value> - specifies text color (Gray colorspace)
\textrgbcolor <value> specifies text color (RGB colorpsace)
\textcmykcolor <value> - specifies text color (CMYK colorspace)

\strokegraycolor <value> - specifies stroke color (Gray colorspace)
\strokergbcolor <value> - specifies stroke color (RGB colorspace)
\strokecmykcolor <value> - specifies stroke color (CMYK colorspace)

\strokewidth <value> - specifies stroke width

\font <value> - specifies font name or font file
\fontsize <value> - specifies font size
\fontembed <value> - specifies if the font should be embedded or not
\fontsubset <value> - specifies if the font should be subsetted or not

\width <value> - specifies image or textbox width
\height <value> - specifies image or textbox height

\linespacing <value> - specifies line spacing for TextBox stamp

\image <value> - specifies image file to use as stamp

\importpdfpage <value> - specifies PDF file to use as PDF overlay stamp

\scale <value> - specifies scaling factor for image and PDF overlay stamps

\overlay - place stamp as overlay - over page content

\underlay - place stamp as watermark - behind page content
\canunstamp - stamp can be removed by the UnStampPDF method in the neeviaPDF PDFstamp COM object

\rotate <value> - rotate stamp by <value> degrees

\opacity <value> - specifies stamp opacity

\x <value> - specifies stamp's X coordinate
\y <value> - specifies stamp's Y coordinate
\units <value> - specifies the measurement units for \x, \y commands

Examples:
1. Stamp "WATERMARK" on first page at (1in, 1.5in), Arial font, green color

```
\stamp 1
\stampname Sample Stamp
\text WATERMARK
\x 1
\y 1.5
\units 1
\font Arial
\textrgbcolor #00FF00
```

2. Add bates numbering (5 digits) to each page at the bottom, Arial font, green color.

```
\stamp 0
\stampname Sample Stamp
\text  %BATES#DIGITS=5%
\x left
\y bottom
\units 1
\overlay
\font Arial
\textrgbcolor #00FF00
```

Appendix B: Variables supported by Text / TextBox stamps

Below is the list of variables supported by both Text and TextBox stamps:

%PAGE%	- current page number
%PAGES%	- total number of pages
%BATES#DIGITS=x#START=n%	- bates numbering Ex: %BATES#DIGITS=5%. Default number: 6 digits.
%FILENAME%	- name of the file
%WEEKDAY%	- full weekday name
%WEEKDAY_SHORT%	- abbreviated weekday name
%MONTH%	- month number (1-12)
%MONTHNAME%	- full month name
%MONTHNAME_SHORT%	- abbreviated month name
%YEAR%	- year with century (YYYY)
%YEAR_SHORT%	- year without century (YY)
%DAY%	- day of month
%DAY_YEAR%	- day of the year (1 -366)
%HOUR%	- hour (01- 12)
%HOURS%	- hour (00-23)
%MINUTES%	- minutes (00-59)
%SECONDS%	- seconds (00-59)
%AMPM%	- AM, PM
%DATE%	- local date representation. Ex: 11/01/07
%TIME%	- local time representation. Ex: 1:17:10 PM
%DATETIME%	- local date and time
%AUTHOR%	- document Author
%TITLE%	- document Title
%SUBJECT%	- document Subject
%KEYWORDS%	- document Keywords
%N%	- new line. This is valid for TextBox stamp only.

Appendix C: Paper sizes supported by PaperSize property

1 - Letter, 8 1/2 x 11 in.
2 - Letter Small, 8 1/2 x 11 in.
3 - Tabloid, 11 x 17 in.
4 - Ledger, 17 x 11 in.
5 - Legal, 8 1/2 x 14 in.
6 - Statement, 5 1/2 x 8 1/2 in.
7 - Executive, 7 1/2 x 10 1/2 in.
8 - A3, 297 x 420 mm
9 - A4, 210 x 297 mm
10 - A4 Small, 210 x 297 mm
11 - A5, 148 x 210 mm
12 - B4, 250 x 354 mm
13 - B5, 182 x 257 mm
14 - Folio, 8 1/2 x 13 in.
15 - Quarto, 215 x 275 mm
16 - 10 x 14 in.
17 - 11 x 17 in.
18 - Note, 8 1/2 x 11 in.
19 - Envelope #9, 3 7/8 x 8 7/8 in.
20 - Envelope #10, 4 1/8 x 9 1/2 in.
21 - Envelope #11, 4 1/2 x 10 3/8 in.
22 - Envelope #12, 4 1/2 x 11 in.
23 - Envelope #14, 5 x 11 1/2 in.
24 - C size sheet
25 - D size sheet
26 - E size sheet
27 - Envelope DL, 110 x 220 mm
29 - Envelope C3, 324 x 458 mm
30 - Envelope C4, 229 x 324 mm
28 - Envelope C5, 162 x 229 mm
31 - Envelope C6, 114 x 162 mm
32 - Envelope C65, 114 x 229 mm
33 - Envelope B4, 250 x 353 mm
34 - Envelope B5, 176 x 250 mm
35 - Envelope B6, 176 x 125 mm
36 - Envelope, 110 x 230 mm
37 - Envelope Monarch, 3 7/8 x 7 1/2 in.
38 - Envelope, 3 5/8 x 6 1/2 in.
39 - U.S. Standard Fanfold, 14 7/8 x 11 in.
40 - German Standard Fanfold, 8 1/2 x 12 in.
41 - German Legal Fanfold, 8 1/2 x 13 in.

Examples

Code samples for docCreator can be found here - https://neevia.com/support/examples/cr/