
mPDF Manual

Ian N Back

CONTENTS

About mPDF	2
Features v5	2
Requirements v5	4
Limitations	5
Credits	6
Licence	7
Installation & Setup	8
Installation v5.x	8
Installation v6.x	9
Reducing memory usage	12
Folders for temporary files	13
Getting Started	14
Creating your first file	14
HTML or PHP?	15
Fonts & Languages	16
About PDF files	16
Fonts in mPDF 5.x	17
Fonts in mPDF 6.x	22
OpenType layout (OTL)	26
Font names	30
Available Fonts v5.x	32
Available fonts v6	33
Choosing a configuration v5.x	35
Choosing a configuration v6.x	39
Automatic font selection	42
lang v5.x	44
lang 6.x	45
Input encoding	46
Fonts & Language cover v5.x	47
Arabic (RTL) text v5.x	48
Bidirectional (RTL) text v6.x	50
CJK Languages	52
Indic fonts v5.x	53
Default Font	54
Font substitution 5.x	55
Font substitution 6.x	56
Character substitution	57
Configuration	58
Configuration files v5.x	58
Configuration files v6.x	59
Configuration Methods	60
Configuration Variables	61
HTML support	62
HTML Tags	62
HTML Attributes	65
Custom HTML Tags	68
Tables	69
Tables	69
Table layout	71
Auto-layout algorithm	72
Border collapse	73
Paging	74
Page breaks	74
Double-sided documents	77
Page size & Orientation	78

<i>Page numbering</i>	79
<i>Using @page</i>	81
<i>Different page sizes</i>	85
Headers & Footers	86
Headers & Footers	86
Method 1	88
Method 2	93
Method 3	96
Method 4	101
<i>Page numbers & Date</i>	106
<i>Headers & Top margins</i>	107
<i>Browser compatible HTML</i>	108
<i>Rotated pages</i>	109
CSS & Stylesheets	110
<i>Introduction</i>	110
<i>Supported CSS</i>	112
<i>Default stylesheet</i>	125
<i>Named colours</i>	127
Setting PDF file properties	131
<i>Password protection</i>	131
<i>Document Metadata</i>	132
<i>PDF Version</i>	133
What Else Can I Do?	134
<i>Backgrounds & Borders</i>	134
<i>Fixed position blocks</i>	137
<i>Floating blocks</i>	139
<i>Hyphenation</i>	140
<i>Images</i>	142
<i>Kerning</i>	146
<i>Line-height</i>	147
<i>Line breaking</i>	149
<i>Lists</i>	150
<i>Text Justification</i>	152
<i>Annotations</i>	153
<i>Barcodes</i>	154
<i>Bookmarks</i>	155
<i>Columns</i>	156
<i>Forms</i>	157
<i>Index</i>	159
<i>Layers</i>	162
<i>Table of Contents</i>	163
<i>Watermarks</i>	165
<i>Graphs</i>	166
<i>Replaceable Aliases</i>	168
<i>CMYK colours</i>	170
<i>Importing files & Templates</i>	171
<i>Overwriting existing files</i>	172
<i>Writing non-HTML text</i>	173
<i>PDF/A1-b compliance</i>	174
<i>PDF/X-1a compliance</i>	177
<i>Capture HTML output</i>	180
<i>Saving to a database</i>	181
<i>Math Formulae with MathJax</i>	182
<i>Math with MathJax 2</i>	183
<i>Combining Diacritics</i>	185
Real life examples	190
<i>PDF from every page of website</i>	190
<i>Invoice</i>	192

Year Book	196
Colour Charts CMYK	199
E-mail a PDF file	200
A5 Booklet	202
Reserving x blank pages	203
Letterhead letters	204
User Input	207
Write directly to document	209
Direct writing to document	209
Troubleshooting	212
Known Issues	212
Slow!	213
Text is replaced	214
Reserved Terms	215
Browser incompatability	216
Error messages	217
Blank screen	218
Corrupt PDF file	219
Notice warnings	220
Image errors	221
Memory problems	222
Resizing	223
mPDF class fails to Initialise	224
Postscript printers	225
mPDF functions	226
mPDF()	226
Overview	229
AddColumn()	232
AddPage()	233
AddPageByArray()	238
AddSpotColor	244
AliasNbPages()	245
AliasNbPageGroups()	246
Annotation()	247
AutoSizeText()	250
Bookmark()	251
CircularText()	253
CreateIndex()	254
CreateReference()	257
DefFooterByName()	258
DefHeaderByName()	260
DefHTMLFooterByName()	262
DefHTMLHeaderByName()	263
Image()	264
ImportPage()	265
IndexEntry()	267
IndexEntrySee()	269
InsertIndex()	270
MultiCell()	271
Output()	272
OverWrite()	274
RestartDocTemplate()	277
RoundedRect()	278
SetAlpha()	279
SetAnchor2Bookmark()	281
SetAuthor()	282
SetFont()	283
SetBasePath()	285

SetColumns()	286
SetCompression()	288
SetCreator()	289
SetDefaultBodyCSS()	290
SetDefaultFont()	291
SetDirectionality()	292
SetDefaultFontSize()	293
SetDisplayMode()	294
SetDisplayPreferences()	296
SetDocTemplate()	297
SetFooter()	299
SetFooterByName()	302
SetHeader()	303
SetHeaderByName()	306
SetHTMLFooter()	308
SetHTMLFooterByName()	309
SetHTMLHeader()	310
SetHTMLHeaderByName()	311
SetImportUse()	313
SetKeywords()	314
SetPageTemplate()	315
SetProtection()	317
SetSourceFile()	319
SetSubject()	321
SetTitle()	322
SetVisibility()	324
SetWatermarkImage()	325
SetWatermarkText()	327
Shaded_box()	329
StartProgressBarOutput()	330
Thumbnail()	331
TOCpagebreak()	333
TOCpagebreakByArray()	338
TOC_Entry()	343
UseTemplate()	345
WriteBarcode()	348
WriteCell()	350
WriteFixedPosHTML()	351
WriteHTML()	353
WriteText()	356
FPDF Original Functions	357
HTML control tags	358
Overview	358
annotation	359
barcode	362
bookmark	365
columnbreak	367
columns	368
dottab	370
formfeed	371
htmlpagefooter	375
htmlpageheader	376
indexentry	377
indexinsert	379
jpggraph	382
pagebreak	387
pageheader	392
pagefooter	394

sethtmlpagefooter	396
sethtmlpageheader	398
setpagefooter	400
setpageheader	402
textcircle	404
tocentry	406
tocpagebreak	408
watermarkimage	413
watermarktext	415
mPDF Variables	417
Overview	417
adjustFontDescLineheight	422
aliasNbPg	423
aliasNbPgGp	424
allow_charset_conversion	425
allow_html_optional_endtags	427
allow_output_buffering	428
allowCJKorphans	429
allowCJKoverflow	430
anchor2Bookmark	431
annotMargin	432
annotOpacity	433
autoArabic	434
autoFontGroupSize	435
autoLangToFont	437
autoMarginPadding	438
autoPageBreak	439
autoScriptToLang	440
autoVietnamese	441
backupSubsFont	442
backupSIPFont	443
baseScript	444
biDirectional	445
bleedMargin	446
bookmarkStyles	447
charset_in	448
CJKforceend	449
collapseBlockMargins	450
cropMarkLength	451
cropMarkMargin	452
crossMarkMargin	453
CSSselectMedia	454
decimal_align	455
debug	456
debugfonts	457
displayDefaultOrientation	458
defaultPageNumStyle	459
dpi	460
enableImports	461
falseBoldWeight	462
forcePortraitHeaders	463
forcePortraitMargins	464
h2bookmarks	465
h2toc	466
ICCPprofile	467
img_dpi	468
incrementFPR1 [1-4]	469
iterationCounter	470

<i>jSmaxChar</i>	471
<i>jSmaxCharLast</i>	472
<i>jSmaxWordLast</i>	473
<i>jSWord</i>	474
<i>justifyB4br</i>	475
<i>keepColumns</i>	476
<i>keep_table_proportions</i>	477
<i>list_align_style</i>	478
<i>list_auto_mode</i>	479
<i>list_indent_default</i>	480
<i>list_indent_default_mpdf</i>	481
<i>list_indent_first_level</i>	482
<i>list_marker_offset</i>	483
<i>list_number_suffix</i>	484
<i>list_symbol_size</i>	485
<i>margBuffer</i>	486
<i>max_colH_correction</i>	487
<i>maxTTFFilesize</i>	488
<i>mirrorMargins</i>	489
<i>nbpqPrefix</i>	490
<i>nbpqSuffix</i>	491
<i>nonPrintMargin</i>	492
<i>normalLineheight</i>	493
<i>packTableData</i>	494
<i>pagenumPrefix</i>	495
<i>pagenumSuffix</i>	496
<i>PDFA</i>	497
<i>PDFAauto</i>	498
<i>PDFX</i>	499
<i>PDFXauto</i>	500
<i>percentSubset</i>	501
<i>printers_info</i>	502
<i>progressbar_altHTML</i>	503
<i>progressbar_heading</i>	504
<i>progressBar</i>	505
<i>repackageTTF</i>	506
<i>restoreBlockPagebreaks</i>	507
<i>restrictColorSpace</i>	508
<i>setAutoBottomMargin</i>	509
<i>setAutoTopMargin</i>	510
<i>showImageErrors</i>	511
<i>showStats</i>	512
<i>showWatermarkImage</i>	513
<i>showWatermarkText</i>	514
<i>SHYlang</i>	515
<i>simpleTables</i>	516
<i>smCapsScale</i>	517
<i>smCapsStretch</i>	518
<i>tableMinSizePriority</i>	519
<i>tabSpaces</i>	520
<i>title2annots</i>	521
<i>use_kwt</i>	522
<i>useAdobeCJK</i>	523
<i>useDictionaryLBR</i>	524
<i>useFixedNormalLineHeight</i>	525
<i>useFixedTextBaseline</i>	526
<i>useGraphs</i>	527
<i>useKerning</i>	528

<i>useLang</i>	529
<i>useSubstitutions</i>	531
<i>useTibetanLBR</i>	532
<i>watermark_font</i>	533
<i>watermarkImageAlpha</i>	534
<i>watermarkImgAlphaBlend</i>	535
<i>watermarkImgBehind</i>	536
<i>watermarkTextAlpha</i>	537
mPDF Utilities	538
<i>strcode2utf()</i>	538
<i>preparePreText()</i>	540
Codepages & Glyphs	542
<i>Win-1252</i>	542
<i>ASCII characters</i>	543
<i>Win-1251</i>	544
<i>ISO-8859-2</i>	545
<i>ISO-8859-4</i>	546
<i>ISO-8859-7</i>	547
<i>ISO-8859-9</i>	548
<i>ISO-8859/win comparison chart</i>	549
<i>ZapfDingbats (Adobe)</i>	552
<i>Symbols (Adobe)</i>	553
<i>Vietnamese</i>	554
<i>BIG-5 (Traditional Chinese)</i>	555
<i>HKCS (Hong Kong Character Set)</i>	562
<i>GBK (Simplified Chinese)</i>	565
<i>SHIFT_JIS (Japanese)</i>	573
<i>UHC (Korean)</i>	577
<i>Demo - Pangrams</i>	585
<i>Demo - Other languages</i>	588
<i>Demo - 1000 Character Classic</i>	589
<i>Demo - Other Unicode character</i>	591
<i>Equivalent codepages</i>	594
<i>iconv</i>	596
<i>mbstring</i>	598
<i>Unicode character planes</i>	599
<i>ISO 639-1 language codes</i>	607
<i>Useful links</i>	612
<i>Unicode coverage of Free Fonts</i>	613
PDF Files (Adobe)	619
<i>PDF reference</i>	619
Index	620

ABOUT MPDF

Features v5

Main features

- Accepts UTF-8 encoded HTML
- Supports almost all languages including **RTL** (arabic and hebrew), and **CJK** - (chinese-japanese-korean)
- Bookmarks
- CSS stylesheets
- Word spacing and character spacing for justification
- Nested block-level elements (e.g. P, DIV) including margins, borders, padding, line-height, background colours etc.
- Support (partial) for floating and fixed-position block-elements
- Page layout and orientation
- Text-justification and hyphenation
- Page numbering
- Odd and even paging with mirrored margins
- Page headers & footers
- Columns
- Tables - nested tables, rotated, or autosized to fit on a page
- Table of contents
- Index
- Watermarks
- Images in JPG, GIF, PNG, SVG, BMP or WMF format
- Password protection
- Annotations
- Barcodes (EAN13, UPC-A/E, Code 11, 39, 93, 128, Codabar, MSI, IMB, Planet, Postnet, RM4SCC etc.)
- Import another PDF file and use as a template
- Embedded font subsets
- PDF/A-1b support (ISO 19005-1:2005)
- PDF/X-1a support

More Information

mPDF has a number of enhancements over the original FPDF, HTML2FPDF and UPDF scripts:

UTF-8 encoded HTML is accepted as the standard input.

Right-to-left languages are supported, with automatic detection of RTL characters within a document. Transposes: tables, lists, text justification and table cell alignment, as well as full text reversal for RTL characters. Automatically detects non-RTL characters and displays these in original order.

Bookmarks and **Meta tag information** are supported in all character sets.

A single CSS stylesheet can be used for all pages, with **font substitution** automatically for CJK characters.

Character substitution can optionally be used to automatically replace any characters that do not exist in the current font.

Word spacing and character spacing are both used to justify text; works in unicode mode and CJK characters as well.

Nested block-level elements (e.g. P, DIV) are supported, with comprehensive CSS support e.g. margins, borders, padding, line-height, background colours etc.

CSS style attributes now fully support font, font-size, color, and background color (for highlighting) plus many more.

Table cell padding and borders are supported.

Text-indent for 1st line of paragraph, and hanging indents are supported.

List indenting can be defined.

Custom tags added - PAGEBREAK, COLUMNBREAK, INDEXENTRY

Multiple columns can be started and stopped anywhere on the page with column height adjusted (and optionally aligned to justify).

Tables can be **rotated**, or **autosized** - font-size is reduced if required to fit the page. Background colour for TR rows is supported.

Odd and even paging can be used with inner and outer margins alternated.

A more complex definition of **headers and footers** allows left/center/right parts to be defined, each with their own font-styles, and including code to allow the date/time to be inserted as well as page numbers.

A **table of contents** can be generated automatically, which can be inserted at the front of the document, based on custom tags used throughout the HTML code. (Based on Richard Bondi's extension to FPDF.)

An **Index** can be generated at the end of the document based on custom tags used throughout the HTML code.

Non-breaking space is supported when using fonts that have a character representing chr(160) (not all of them).

Watermarks as text or images can be used e.g. for 'DRAFT', and will appear as a transparency over other elements.

Intelligent **word-wrapping** will avoid 'orphan' punctuation or superscript moving to the next line.

Automatic hyphenation is optional.

All **HTML entities** are supported, as well as decimal and hex e.g. ' ≬ or ↤

Password protection can be set for the PDF file.

NB The original commands from FPDF can be used e.g. Write(), but some are altered to allow UTF-8 encoding and RTL text to be processed e.g. use WriteCell() and WriteMultiCell() instead of Cell() and MultiCell().

Requirements v5

mPDF requires a minimum of PHP 4.3.10 or PHP 5.0.3

mPDF v5.7.1 minimum is required for PHP>=5.5

(Since version 1, mPDF has been developed on PHP5 only, but should still be compatible with PHP4>=4.3.10)

PHP requires the mb_string module to be enabled (including mbregex, which needs to be explicitly enabled in some environments)

mPDF is **not** compatible with PHP function overloading (mbstring.func_overload)

As for FPDF, PHP requires zlib for compression.

Improved performance requires write permissions to the temporary cache folders.

Limitations

The script is (a lot) slower than the original FPDF and html2fpdf. Some of this is due to the inclusion of unicode font files (when used), but there is also an increase in processing time.

Tables

Block elements (e.g. DIV or P) are not supported inside tables. The content is displayed, but any CSS properties which apply to block elements are ignored (e.g. borders, padding, margins etc).

Block and in-line elements

All HTML elements are hard-coded to be treated as block or in-line elements (e.g. equivalent to CSS display:block or display:inline). This cannot be changed using CSS. See [HTML tags](#).

Special features

Several of the "special" features of mPDF are incompatible with each other e.g columns, fixed-position block elements, page-break-avoid:inside, Keep-with-table and rotated tables.

Other

Millimeters are the only accepted dimensions for defining page size and margins within mPDF (CSS stylesheets accept all usual units).

Blocks which are defined as position:absolute, fixed or float have only limited support (introduced v4.0).

See Also

- [Known issues](#)

Credits

All credit is due to the following:

- Olivier Plathey for the original FPDF class [<http://www.fpdf.org>]
- Renato Coelho for html2fpdf class [<http://html2fpdf.sourceforge.net>]
- Steven Wittens for the unicode fonts in UFPDF [<http://acko.net/node/56>]
- Nicola Asuni for TCPDF [<http://www.tcpdf.org>]
- Khaled Al-Sham'aa for ideas on joining Arabic letters [<http://www.ar-php.org>]

Including from FPDF extensions:

- Martin Hall-May for WMF support

Including from HTML2FPDF:

- Damon Kohler for the Flowing Block script
- Clément Lavoillotte for HTML-oriented FPDF idea
- Yamasoft for the gif.php class
- Jérôme Fenal for the _parsegif() function
- "VIETCOM" for the PDFTable code
- Yukihiro O. for the SetDash() function
- Ron Korving for the WordWrap() function
- Michel Poulain for the DisplayPreferences() function
- Seb for the _SetTextRendering() and SetTextOutline() functions
- Klemen VODOPIVEC for FPDF_Protection

Licence

License: GPL

A PHP class to generate PDF files from HTML with Unicode/UTF-8 and CJK support

Copyright (C) 2010 Ian Back

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License version 2 as published by
the Free Software Foundation.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

[<http://www.opensource.org/licenses/gpl-license.php>]

INSTALLATION & SETUP

Installation v5.x

First-time users

Installation:

- Download the .zip file and unzip it
- Create a folder e.g. /mpdf on your server
- Upload all of the files to the server, maintaining the folders as they are
- Ensure that you have write permissions set for the following folders:
 - /ttfontdata/
 - /tmp/
 - /graph_cache/

To test the installation, point your browser to the example files folder e.g.

[path_to_mpdf_folder]/mpdf/examples/

If you wish to install additional fonts please see the notes in [Fonts & Languages](#) for further instructions.

If you wish to define a different folder for temporary files rather than /tmp/ see the note on '[Folder for temporary files](#)'.

If you have problems, please read the section on [troubleshooting](#) in the manual.

Upgrading from version mPDF 5.0 Beta

There is no upgrade package from v5 beta because so many of the files have had at least minor changes. You can overwrite most of the files, taking care to keep a note of your 3 configuration files.

Important: You must delete all temporary files in the /ttfontdata/ folder.

Changes from 5.0 Beta

- config.php file has been changed (extra CJK characters to recognise CJK blocks)
- \$this->backupSubsFont (in config_fonts.php) optionally now takes an array
- no need to define 'cjk'=>true or 'sip|smp'=>true in config_fonts.php (ignored; cf. \$this->BMPonly)
- Indic language fonts have been altered to add Latin and Latin-1 Supplement characters
- progress bars now has an external progbar.css and configurable main heading
- added initial parameter new mPDF('+aCJK') or ' -aCJK' to override default useAdobeCJK at runtime
- QRCode is not included in main download (but as an extra package)

Earlier versions

If you have been using earlier versions of mPDF, most scripts should work as before. But note:

- Arial, Helvetica, Times and Courier are now treated like any other font
- the whole CSS font string is parsed e.g. style="font-family:'Lucida Grande';" will look for a font 'lucidagrande' and not 'lucida'

Configurable variables:

- \$mpdf->useSubstitutionsMB is now deprecated, but will work as an alias for \$mpdf->useSubstitutions
- \$mpdf->useOnlyCoreFonts is now deprecated and is ignored. Use new mPDF('c')
- \$this->use_CJK_only is now deprecated and is ignored. See \$this->useAdobeCJK and new mPDF('+aCJK') or ' -aCJK '

The initial parameters e.g. new mPDF('utf-8') have all changed. Old ones may be recognised, or will be ignored.

Installation v6.x

First-time users

Installation:

- Download the .zip file and unzip it
- Create a folder e.g. /mpdf on your server
- Upload all of the files to the server, maintaining the folders as they are
- Ensure that you have write permissions set for the following folders:
 - /ttfontdata/
 - /tmp/
 - /graph_cache/

To test the installation, point your browser to the example files folder e.g.
[path_to_mpdf_folder]/mpdf/examples/

If you wish to install additional fonts please see the notes in [Fonts & Languages](#) for further instructions.

If you wish to define a different folder for temporary files rather than /tmp/ see the note on '[Folder for temporary files](#)'.

If you have problems, please read the section on [troubleshooting](#) in the manual.

Upgrading from version mPDF 5.x

mPDF 6 has changed significantly from earlier version and it is recommended that a fresh install is used. You may wish to copy your previous config_* files and use them to update the new config files.

config_fonts.php - values of "indic" and "unAglyphs" from previous versions are now redundant.

config_lang2fonts.php - this is similar to the previous config_cp.php file; note however that \$unifont (NOT \$unifonts) must be only one font (not a comma-separated list as before).

Included fonts - the Indic fonts e.g. ind_bn_001.ttf are no longer required (nor do they work properly with mPDF 6).

useLang - this configurable variable, which used to be true by default, is now redundant. You may need to set: \$mpdf->autoLangToFont = true; for the same results.

SetFont() - is now redundant. You may need to set: \$mpdf->autoScriptToLang = true; for the same results.

Indexes - have been largely redefined. See the section above.

Lists - have been rewritten. See the section above.

Headers and Footers - have been rewritten. See the section above.

A number of old depracated aliases will no longer be supported. Warning errors have been added to prompt you to change to the updated form:

- \$mpdf->useOddEven - should now use - \$mpdf->mirrorMargins
- \$mpdf->useSubstitutionsMB - should now use - \$mpdf->useSubstitutions
- \$mpdf->AliasNbPg - should now use - \$mpdf->aliasNbPg
- \$mpdf->AliasNbPgGp - should now use - \$mpdf->aliasNbPgGp
- \$mpdf->BiDirectional - should now use - \$mpdf->biDirectional
- \$mpdf->Anchor2Bookmark - should now use - \$mpdf->anchor2Bookmark
- \$mpdf->KeepColumns - should now use - \$mpdf->keepColumns
- \$mpdf->UnvalidatedText - should now use - \$mpdf->watermarkText
- \$mpdf->TopicIsUnvalidated - should now use - \$mpdf->showWatermarkText
- \$mpdf->Reference - should now use - \$mpdf->IndexEntry

The following functions have been removed:

- setUnvalidatedText - should now use - SetWatermarkText()
- AddPages - should now use - AddPage() or HTML code methods
- startPageNums
- CreateReference and CreateIndex - cf. Index section above

Default style sheet

A new mpdf.css file includes defaults for LISTS top/bottom margins, and also examples for Indexes and ToCs. This now acts like a normal CSS file, including cascading selectors i.e. not just main tags. This is always read (if present), so acts as a secondary default CSS, but one which allows selectors. Styles added to this act like a user stylesheet when considering precedence e.g. cellSpacing and border-spacing.

Direct writing methods and OTL

WriteText() WriteCell() Watermark() AutoSizeText() and ShadedBox() DO support complex scripts and right-to-left text (RTL).

Write() does NOT support complex scripts or RTL (NB this is a change - Write() used to support RTL).

CircularText() does NOT support complex scripts or RTL.

MultiCell() DOES support complex scripts and RTL, but complex-script line-breaking MAY NOT be accurate.

MultiCell() does not support kerning and justification. NB This includes <textarea> in forms which uses MultiCell() internally.

<select> form objects also do NOT support kerning.

Page numbering

Page numbering i.e. by including {PAGENO} or {nbpg} in a header/footer, can use any of the number types as used for list-style e.g.

<pagebreak pagenumstyle="arabic-indic">

Short codes are recognised for the 5 most common:

- "1" - decimal
- "A" = upper-latin or upper-alpha
- "a" = lower-latin or lower-alpha
- "I" = upper-roman
- "i" = lower-roman

or any of the following: arabic-indic, hebrew, bengali, devanagari, gujarati, gurmukhi, kannada, malayalam, oriya, persian, tamil, telugu, thai, urdu, cambodian, khmer, lao, cjk-decimal

Note: A suitable font must be used in the header/footer in order to display the numbers in the selected script.

You can now set the pagenumberstyle from the beginning of the document by changing the configurable variable:

```
$this->defaultPageNumStyle = "arabic-indic"; // in config.php
$mpdf->defaultPageNumStyle = "arabic-indic"; // at runtime
```

Other Minor changes in mPDF 6

'hebrew', 'khmer', 'cambodian', 'lao', and 'cjk-decimal' are recognised as values for "list-style-type" in numbered lists.

CSS "text-outline" is now supported on TD/TH tags

Text wrapping in tables has been improved when using CJK scripts (chinese-japanese-korean).

Text underline and strikethrough can be used together: Hallo world. Either <u><s>...</s></u> or ... can be used

Added support for style="opacity:0.6;" in SVG - equivalent to: style="fill-opacity:0.6; stroke-opacity: 0.6;"

Added support for opacity="0.6" (as attribute) in SVG - previously only supported fill-opacity="0.6" stroke-opacity="0.6"

CSS position:absolute or fixed - rotate extended now to include rotate: 180; (previously just 90 or -90)

The default value of \$this->keep_table_proportions = true; in config.php has been changed (see effect on

Example 6 - nested table in top right cell).

Limited support has been added for SVG fonts embedded in SVG images (but not using @font-face rules) - see the separate Images demo file.

When using columns, the top margin is now collapsed at top of every column (not just first column of page).

The way mPDF handles optional end tags has been updated to be consistent with the [HTML5 specification](#) - previously not well defined for HTML4.

Changes to the way lists are handled means that text-align:justify may be inherited by lists from surrounding block elements (which did not happen previously). See LISTS above for more information.

Reducing memory usage

mPDF "Lite"

The mPDF main file `mpdf.php` is over 1MB in size. Simply to parse the mPDF class requires over 10MB of memory in PHP. This may not be a problem, but if your PHP configuration does not allow you to increase memory, or you envisage multiple calls on your server at the same time, you can consider producing a 'Lite' form of the `mpdf.php` script.

A utility script `compress.php` is included in mPDF in the root folder. This generates a new `mpdf.php` script omitting functions which you do not require. (It does not actually compress the file)

As from mPDF 6.0.1 this file will be distributed as `compress.php.distr` to avoid other users running the file unwantedly. To use `compress`, just rename the file `compress.php` (and remember to delete or rename it after you have finished using it).

As a guideline:

Version 5.1	Size of <code>mpdf.php</code> file	Memory usage in PHP
Full script	1170kB	12.75MB
Omitting functions, but including:	697kB	7.0MB
Tables, lists and images		
Omitting all optional functions	496kB	5.25MB

Firstly, rename the `mpdf.php` file as `mpdf_source.php`

Then point your browser to [path_to_mpdf]/compress.php

Follow the instructions on screen. This will produce (overwriting if necessary) a new `mpdf.php` file.

Do not delete the `mpdf_source` file, which will remain as your original.

Tip: Consider setting `$mpdf->simpleTables = true;` if you do not need complex table borders, or `$mpdf->packTableData = true;` if you do not mind the extra processing time.

Note: mPDF>=5.0 Ensure that you have set write permissions to temporary folders

Folders for temporary files

mPDF is configured to use [your_path_to_mpdf]/tmp/ as a folder to write temporary files (mainly for images), and [your_path_to_mpdf]/ttfondata/ as a folder to cache font metrics data. Write permissions should ideally be set on both these folders to allow read/write access for the script.

Images

If you wish to use a different folder for temporary files, you should define the constant `_MPDF_TEMP_PATH` before including the `mpdf.php` file e.g.

```
define("_MPDF_TEMP_PATH", ' ../../common/tempfiles/');
include("../mpdf.php");
$mpdf=new mPDF();
```

Images will still be processed without write permissions to this folder, but at considerable cost in processing time and memory usage.

Fonts

If you wish to use a different folder for temporary files, you should define the constant `_MPDF_TTFONTDATAPATH` before including the `mpdf.php` file

Fonts can still be used and embedded without write permissions to this folder, but at some cost in processing time and memory usage.

GETTING STARTED

Creating your first file

Getting started

The following PHP will produce the most basic example with mPDF.

Include the main file containing the mpdf class:

```
include('../mpdf.php');
```

Create an instance of the class:

```
$mpdf=new mPDF();
```

Write some HTML code:

```
$mpdf->WriteHTML('<p>Hallo World</p>');
```

Output a PDF file:

```
$mpdf->Output();
exit;
```

Notes

Note: `_MPDF_PATH` was required to be defined explicitly prior to mPDF 4.0 e.g.
`define('_MPDF_PATH','../').` From mPDF 4.0 the value should be automatically defined by the script itself when including the mpdf.php file.

For details and options for the Output command, see [Output\(\)](#)

HTML or PHP?

Most of the functions of mPDF can be achieved two ways: using PHP commands, or using custom HTML tags. Use whichever suits your purpose better, and you can always combine a mixture of the two.

If you are new to mPDF, I would recommend using HTML/CSS for everything - you can test most things out in a browser this way as you are writing it.

Example:

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->Bookmark('Start of the document');  
$mpdf->WriteHTML('<div>Section 1 text</div>');  
$mpdf->Output();  
  
?>
```

Example #2

```
<?php  
  
$html = '<bookmark content="Start of the Document" /><div>Section 1 text</div>';  
  
$mpdf=new mPDF();  
$mpdf->WriteHTML($html);  
$mpdf->Output();  
  
?>
```

Invalid HTML?

If you want to hide mPDF custom tags from a browser, enclose any mPDF code within:

```
<!--mpdf ... anything you want to write ... mpdf-->
```

mPDF will strip away the `<!--mpdf` tag and any following spaces, and the `mpdf-->` tag and any preceding spaces, and process all enclosed code e.g.

```
<!--mpdf <htmlheader id="header1"><h2>Section 2</h2></htmlheader> mpdf-->
```

See Also

[Capturing HTML](#)

FONTS & LANGUAGES

About PDF files

In order for users to be able to read a PDF file, they need to be able to access the necessary fonts/characters. They can do this in three ways:

- PDF files have certain standard fonts: Arial/Helvetica, Times and Courier in the win-1252 character set, and Zapfdingbats and Symbol character sets. These fonts should be available to any PDF reading program, and do not need to be embedded in the PDF document.
- Files using CJK (chinese-japanese-korean) characters can refer to standard fonts which are freely downloadable from the [Adobe website](#). Users will need to download these font-packs to read the document. The four Asian fonts provided by Adobe contain all the characters in codepages: SHIFT_JIS, UHC (cp949), GBK, or BIG5.
- Font information can be embedded in the file. It is possible to embed a subset of the font information selectively for only the characters used in the document, or the whole font file.

Fonts in mPDF 5.x

mPDF version 5 supports Truetype fonts, reading and embedding directly from the .ttf font files. Fonts must follow the Truetype specification and use Unicode mapping to the characters. Truetype collections (.ttc files) and Opentype files (.otf) in Truetype format are also supported.

Easy to add new fonts

1. Upload the Truetype font file to the fonts directory (/ttfonts)
 2. Define the font file details in the configuration file (config_fonts.php)
 3. Access the font by specifying it in your HTML code as the CSS font-family
 4. To use the font with specific languages, you need also to edit the configuration file (config_cp.php) - see the example

Example

You have 2 font files "Frutiger-Normal.ttf" and "FrutigerObl-Normal.ttf" which you want to be available in mPDF, and you will refer to them in HTML/CSS as "Frutiger".

1. Upload the 2 files to the fonts directory (/ttfonts)
 2. In the configuration file (config_fonts.php) add this to the array \$this->fontdata:

```
"frutiger" => array(  
    'R' => "Frutiger-Normal.ttf",  
    'I' => "FrutigerObl-Normal.ttf",  
).
```

3. In your HTML or CSS code use something like this:

<p style="font-family: Frutiger">....</p>

4. To use the font with specific languages, you need also to edit the configuration file (`config_cp.php`); let us imagine that Frutiger contains a full set of characters needed for the Thai language:

```
// THAI  
CASE "th": $unifonts = "frutiger,frutigerI"; break;
```

This will enable the Frutiger font in all of the following:

```
$mpdf = new mPDF('th');
```

```
$mpdf->SetAutoFont(AUTOFONT_THAIVIET);
```

```
<p lang="th"> . . . </p>
```

Full Unicode support

The DejaVu fonts distributed with mPDF contain an extensive set of characters, but it is easy to add fonts to access uncommon characters.

(The following characters will only appear in your browser if you have appropriate fonts installed on your computer.)

/miscellaneous-technical-symbols
 Miscellaneous Technical (Symbola)
 /miscellaneous-alphanumerics-quivira
 Enclosed Alphanumerics (Quivira)
 /miscellaneous-symbols-dejavusanscondensed
 Miscellaneous Symbols (DejaVuSansCondensed)

Names in brackets are open-source fonts which cover the Unicode ranges - see [Unicode coverage of free fonts](#) for details of these and other open-source fonts.

Complex scripts

Although all characters may be displayed, there are various reasons why a script may not appear as expected.

Right-to-left languages (Hebrew, Arabic etc.)

Arabic languages and Hebrew are written in a right-to-left direction (RTL). mPDF recognises both Arabic and Hebrew languages and reverses text direction automatically.

سلام عليكم שלום

Arabic languages (but not Hebrew) also change the form of the letter depending on its position in the text e.g. these are the initial, medial, final, and isolated forms of arabic letter 'ain':

ع ع ع ع

The isolated characters are contained in the Unicode block 'Arabic' U+0600 - U+06FF.

The initial, medial and final forms are contained in Unicode Blocks 'Arabic Presentation Forms' A and B (U+FB50 - U+FDFF, U+FE70 - U+FEFE). Note that quite a large number of fonts contain the isolated characters but not the presentation forms. Fonts used with mPDF must contain the 'Arabic Presentation Forms' in order to display arabic text correctly. mPDF automatically converts letters to their initial/medial/final forms in several languages: arabic, persian/farsi, urdu, sindhi and pashto.

Arabic text is used for many different languages e.g. persian/farsi, urdu, pashto etc. These languages often contain letters unique to that language. 'Arabic' fonts do not always contain the full set of arabic characters necessary for all languages.

Other RTL languages (using other alphabets) are reversed in order, but not otherwise processed, by mPDF e.g. Syriac, Thaana, N'Ko, and Samaritan.

Indic languages

Indic languages are also complex scripts which require some processing of characters before display. For example some vowels consist of 2 characters, to be placed before and after the adjacent consonant e.g.

U+0D1C + U+0D4C [vowel AU] = [written together as ജൌ]
 ജ + ഇ = ജീ

Consonant conjuncts are where two adjacent characters are written as a single 'conjunct' form e.g.

പ + ല്ല = പ്ല്ല

mPDF can support some of these languages, but requires specially prepared font files that are unique to mPDF. These are distributed with the mPDF package.

Supported languages: Bengali, Devanāgarī, Gujarāti, Gurmukhi, Kannada, Malayalam, Oriya, Tamil, Telugu

ଆসନ୍ତାକୁ ଆଲାଇକ୍ରୋମ ନମେତା ମହିଳାଙ୍କ ନମସ୍ତେ ବଣାକୁକମ୍!

Complex scripts **not** supported: Khmer, Sinhala, Tibetan, Myanmar (Burmese), Balinese

Vertical writing

Vertical writing is not supported by mPDF (e.g. Mongolian and Phags-pa) although the individual characters can be displayed using suitable fonts.

Combining diacritics

In Unicode, letters with diacritics (e.g. ÁáÀäÄä) are usually represented as a single character e.g. Unicode U+0196 is an A Umlaut. There are 4 blocks in Unicode of diacritics or 'marks' which can be used to combine with adjacent letters: Combining Diacritical Marks (U+0300 - U+036F), Combining Diacritical Marks Supplement (U+1DC0 - U+1DFF), Combining Marks for Symbols(U+20D0 - U+20FF) and Combining Half Marks (U+FE20 - U+FE2F).

Software applications may use special positioning information stored in OpenType font files to reposition the diacritic/mark depending on the context. mPDF does not support this repositioning and is dependent on the font design and original placement of the diacritic. (The following characters may display differently depending on your browser.)

Á á À à Ä ä ï (Precomposed characters)
 Á á À à Ä ä ï (Using diacritics)

It is recommended to use precomposed characters whenever possible with mPDF.

Unicode Supplementary Planes

The original Unicode allocated characters between x0000 and xFFFF (65,536 characters). This 'Basic Multilingual Plane' supported most characters in common use, including a large number of Unified Chinese-Japanese-Korean characters (CJK). Later the Unicode standard was extended to 16 Planes.

The first plane (plane 0), the Basic Multilingual Plane (BMP), is where most characters have been assigned so far.

Plane 1, the Supplementary Multilingual Plane (SMP), is mostly used for historic scripts such as Linear B, but is also used for musical and mathematical symbols.

Plane 2, the Supplementary Ideographic Plane (SIP), is used for about 40,000 Unified Han (CJK) Ideographs.

mPDF version 5 supports fonts containing characters from all Unicode Planes. By choosing the correct font, almost every single character from Unicode 5 can be displayed in a PDF file.

Unicode Supplementary Multilingual Plane (SMP or Plane 1) U+10000 - U+1FFFF

The Unicode Supplementary Multilingual Plane contains ranges such as Gothic text and Egyptian Hieroglyphics, as well as other (mainly) ancient scripts - see [Unicode coverage of free fonts](#) for full list.

mPDF uses a different method to embed fonts in the PDF file if they include characters from SMP or SIP, because the characters cannot be represented by a 4 character hex code 0000-FFFF. This method is less efficient than the default method, and it can be suppressed by adding the font name to the array `BMPonly` in the `config_fonts.php` configuration file.

Note that the DejaVu fonts distributed with mPDF do contain a few characters in the SMP plane, but most users will not require them and by default they are added to the array `BMPonly`.

Using CJK fonts in mPDF

Fonts containing CJK characters are large files, typically 10-30MB. Adobe provides a free download of an 'Asian font pack' allowing you to create PDF files without including (embedding) the font information in the file. This keeps the file size to a minimum and minimises resource usage on your website generating the PDF file. However, users will have to download the Adobe font packs to read the file, and other PDF software will not display the text correctly.

mPDF allows you to embed subsets of CJK fonts keeping file size down, although there is considerable memory usage to generate these files.

Some CJK fonts are broken up into 2 files because of the size of the files. One freely available font with almost complete coverage of all CJK characters (in both BMP and SIP) is 'Sun' available from Alan Wood's excellent website: <http://www.alanwood.net/unicode/fonts-east-asian.html>. This comes as 2 files, Sun-ExtA and Sun-ExtB (both about 20MB in size) containing the characters from BMP and SIP respectively.

mPDF allows you to treat these as one font by defining the second file as an SIP-extension of the first in the `config_fonts.php` configuration file.

This is an example of the entry in the config_fonts.php configuration file:

```
$this->fontdata = array(
...
    "sun-exta" => array(
        'R' => "Sun-ExtA.ttf",
        'sip-ext' => 'sun-extb',
    ),
    "sun-extb" => array(
        'R' => "Sun-ExtB.ttf",
    ),
...
);
```

This is a sample of HTML code containing CJK characters in both BMP and SIP - note only the sun-exta font-family needs to be referenced:

```
<div style="font-family:sun-extA;"> &#40706; &#40712; &#40727; &#x2320f; &#x23225; &#40742; &#40743;
&#x2322f; &#x23231; &#40761; &#40772; &#x23232; &#x23233; &#40773; &#40784; &#x23234; &#x23256;
&#40787; &#40794; &#x23262; &#x23281; &#40802; &#40809; &#x23289; &#x2328a; </div>
```

TrueType Collections

TrueType Collections (.ttc files) contain more than one font. mPDF treats each font separately by defining the TTCfontID array in the config_fonts.php configuration file.

This example uses the Windows MingLiU fonts, which consist of 2 files containing 6 fonts (note that mingliub is not a Bold variant):

Font collection file (mingliu.ttc) contains the following fonts:

- [1] MingLiU (mingliu) Regular
- [2] PMingLiU (pmingliu) Regular (Proportional)
- [3] MingLiU_HKSCS (mingliu_hkscs) Regular

Font collection file (mingliub.ttc) contains the following fonts:

- [1] MingLiU-ExtB (mingliu-extb) Regular
- [2] PMingLiU-ExtB (pmingliu-extb) Regular (Proportional)
- [3] MingLiU_HKSCS-ExtB (mingliu_hkscs-extb) Regular

This is the entry in the config_fonts.php configuration file:

```
$this->fontdata = array(
...
    "mingliu" => array(
        'R' => "mingliu.ttc",
        'TTCfontID' => array (
            'R' => 1,
        ),
        'sip-ext' => 'mingliu-extb',
    ),
    "pmingliu" => array(
        'R' => "mingliu.ttc",
        'TTCfontID' => array (
            'R' => 2,
        ),
        'sip-ext' => 'pmingliu-extb',
    ),
    "mingliu_hkscs" => array(
        'R' => "mingliu.ttc",
        'TTCfontID' => array (
            'R' => 3,
        ),
        'sip-ext' => 'mingliu_hkscs-extb',
    ),
    "mingliu-extb" => array(
```

```
'R' => "mingliub.ttc",
'TTCfontID' => array (
    'R' => 1,
),
),
"pmingliu-extb" => array(
    'R' => "mingliub.ttc",
    'TTCfontID' => array (
        'R' => 2,
),
),
),
"mingliu_hkscs-extb" => array(
    'R' => "mingliub.ttc",
    'TTCfontID' => array (
        'R' => 3,
),
),
),
...
);
```

This is an example of HTML code containing CJK characters from both BMP and SIP, and selecting the different fonts:

```
<div style="font-family:mingliu;"> &#40706; &#40742; &#40772; &#40784; &#40802; &#40809; &#x23289;
&#x2328a; </div>
<div style="font-family:mingliu_hkscs;"> &#40706; &#40742; &#40772; &#40784; &#40802; &#40809;
&#x23289; &#x2328a; </div>
<div style="font-family:pmingliu;"> &#40706; &#40742; &#40772; &#40784; &#40802; &#40809; &#x23289;
&#x2328a; </div>
```

Fonts in mPDF 6.x

mPDF supports Truetype fonts, reading and embedding directly from the .ttf font files. Fonts must follow the Truetype specification and use Unicode mapping to the characters. Truetype collections (.ttc files) and Opentype files (.otf) in Truetype format are also supported.

Easy to add new fonts

1. Upload the Truetype font file to the fonts directory (/ttfonts)
2. Define the font file details in the configuration file (config_fonts.php)
3. Access the font by specifying it in your HTML code as the CSS font-family
4. To use the font with specific languages, you need also to edit the configuration file (config_lang2fonts.php) - see the example

Example

You have 2 font files "Frutiger-Normal.ttf" and "FrutigerObl-Normal.ttf" which you want to be available in mPDF, and you will refer to them in HTML/CSS as "Frutiger".

1. Upload the 2 files to the fonts directory (/ttfonts)
2. In the configuration file (config_fonts.php) add this to the array \$this->fontdata:

```
"frutiger" => array(
    'R' => "Frutiger-Normal.ttf",
    'I' => "FrutigerObl-Normal.ttf",
),
```

For fonts which handle complex scripts and/or right-to-left text, see [OpenType layout \(OTL\)](#) for details of setting OTL use and support for kashida e.g.

```
"frutiger" => array(
    'R' => "Frutiger-Normal.ttf",
    'I' => "FrutigerObl-Normal.ttf",
    'useOTL' => 0xFF,
    'useKashida' => 75,
),
```

3. In your HTML or CSS code use something like this:

```
<p style="font-family: Frutiger">....</p>
```

4. To use the font with specific languages, you need also to edit the configuration file (config_lang2fonts.php); let us imagine that Frutiger contains a full set of characters needed for the Thai language:

```
// THAI
CASE "th": $unifont = "frutiger"; break;
```

This will enable the Frutiger font whenever the lang attribute is set, if the configurable variable autoLangToFont is set to true:

```
<p lang="th">...</p>
```

Full Unicode support

The DejaVu fonts distributed with mPDF contain an extensive set of characters - see [Unicode coverage of free fonts](#) for details of these and other open-source fonts.

Complex scripts

Right-to-left languages (Hebrew, Arabic etc.)

Arabic languages and Hebrew are written in a right-to-left direction (RTL). mPDF recognises both Arabic and

Hebrew languages and reverses text direction automatically.

سلام عليكم شلום

Arabic languages (but not Hebrew) also change the form of the letter depending on its position in the text e.g. these are the initial, medial, final, and isolated forms of arabic letter 'ain':

ع ع ع ع

Note: You must enable OpenType layout (OTL) features for a font to correctly display right-to-left scripts.

Indic languages, Lao, Tibetan etc.

Other complex scripts require some processing of characters before display. For example some vowels consist of 2 characters, to be placed before and after the adjacent consonant e.g.

U+0D1C + U+0D4C [vowel AU] = [written together as ജൌ]

ଜ + ଏୟ = ଜୟ

Consonant conjuncts are where two adjacent characters are written as a single 'conjunct' form e.g.

ପ + ଲ୍ଳ = ପ୍ଲ୍ଳ

Note: You must enable OpenType layout (OTL) features for a font to correctly display complex scripts.

Vertical writing

Vertical writing is not supported by mPDF (e.g. Mongolian and Phags-pa) although the individual characters can be displayed using suitable fonts.

Unicode Supplementary Planes

The original Unicode allocated characters between x0000 and xFFFF (65,536 characters). This 'Basic Multilingual Plane' supported most characters in common use, including a large number of Unified Chinese-Japanese-Korean characters (CJK). Later the Unicode standard was extended to 16 Planes.

The first plane (plane 0), the Basic Multilingual Plane (BMP), is where most characters have been assigned so far.

Plane 1, the Supplementary Multilingual Plane (SMP), is mostly used for historic scripts such as Linear B, but is also used for musical and mathematical symbols.

Plane 2, the Supplementary Ideographic Plane (SIP), is used for about 40,000 Unified Han (CJK) Ideographs.

mPDF supports fonts containing characters from all Unicode Planes. By choosing the correct font, almost every single character from Unicode 5 can be displayed in a PDF file.

Unicode Supplementary Multilingual Plane (SMP or Plane 1) U+10000 - U+1FFFF

The Unicode Supplementary Multilingual Plane contains ranges such as Gothic text and Egyptian Hieroglyphics, as well as other (mainly) ancient scripts - see [Unicode coverage of free fonts](#) for full list.

mPDF uses a different method to embed fonts in the PDF file if they include characters from SMP or SIP, because the characters cannot be represented by a 4 character hex code 0000-FFFF. This method is less efficient than the default method, and it can be suppressed by adding the font name to the array `BMPonly` in the `config_fonts.php` configuration file.

Note that the DejaVu fonts distributed with mPDF do contain a few characters in the SMP plane, but most users will not require them and by default they are added to the array `BMPonly`.

Using CJK fonts in mPDF

Fonts containing CJK characters are large files, typically 10-30MB. Adobe provides a free download of an 'Asian font pack' allowing you to create PDF files without including (embedding) the font information in the file. This

keeps the file size to a minimum and minimises resource usage on your website generating the PDF file. However, users will have to download the Adobe font packs to read the file, and other PDF software will not display the text correctly.

mPDF allows you to embed subsets of CJK fonts keeping file size down, although there is considerable memory usage to generate these files.

Some CJK fonts are broken up into 2 files because of the size of the files. One freely available font with almost complete coverage of all CJK characters (in both BMP and SIP) is 'Sun' available from Alan Wood's excellent website: <http://www.alanwood.net/unicode/fonts-east-asian.html>. This comes as 2 files, Sun-ExtA and Sun-ExtB (both about 20MB in size) containing the characters from BMP and SIP respectively.

mPDF allows you to treat these as one font by defining the second file as an SIP-extension of the first in the config_fonts.php configuration file.

This is an example of the entry in the config_fonts.php configuration file:

```
$this->fontdata = array(
...
    "sun-exta" => array(
        'R' => "Sun-ExtA.ttf",
        'sip-ext' => 'sun-extb',
    ),
    "sun-extb" => array(
        'R' => "Sun-ExtB.ttf",
    ),
...
);
```

This is a sample of HTML code containing CJK characters in both BMP and SIP - note only the sun-exta font-family needs to be referenced:

```
<div style="font-family:sun-exta;"> &#40706; &#40712; &#40727; &#x2320f; &#x23225; &#40742; &#40743;
&#x2322f; &#x23231; &#40761; &#40772; &#x23232; &#x23233; &#40773; &#40784; &#x23234; &#x23256;
&#40787; &#40794; &#x23262; &#x23281; &#40802; &#40809; &#x23289; &#x2328a; </div>
```

TrueType Collections

TrueType Collections (.ttc files) contain more than one font. mPDF treats each font separately by defining the TTCfontID array in the config_fonts.php configuration file.

This example uses the Windows MingLiU fonts, which consist of 2 files containing 6 fonts (note that mingliub is not a Bold variant):

Font collection file (mingliu.ttc) contains the following fonts:

- [1] MingLiU (mingliu) Regular
- [2] PMingLiU (pmingliu) Regular (Proportional)
- [3] MingLiU_HKSCS (mingliu_hkscs) Regular

Font collection file (mingliub.ttc) contains the following fonts:

- [1] MingLiU-ExtB (mingliu-extb) Regular
- [2] PMingLiU-ExtB (pmingliu-extb) Regular (Proportional)
- [3] MingLiU_HKSCS-ExtB (mingliu_hkscs-extb) Regular

This is the entry in the config_fonts.php configuration file:

```
$this->fontdata = array(
...
    "mingliu" => array(
        'R' => "mingliu.ttc",
        'TTCfontID' => array (
            'R' => 1,
        ),
        'sip-ext' => 'mingliu-extb',
```

```

),
"pmingliu" => array(
    'R' => "mingliu.ttc",
    'TTCfontID' => array (
        'R' => 2,
    ),
    'sip-ext' => 'pmingliu-extb',
),
"mingliu_hkscs" => array(
    'R' => "mingliu.ttc",
    'TTCfontID' => array (
        'R' => 3,
    ),
    'sip-ext' => 'mingliu_hkscs-extb',
),
"mingliu-extb" => array(
    'R' => "mingliub.ttc",
    'TTCfontID' => array (
        'R' => 1,
    ),
),
"pmingliu-extb" => array(
    'R' => "mingliub.ttc",
    'TTCfontID' => array (
        'R' => 2,
    ),
),
"mingliu_hkscs-extb" => array(
    'R' => "mingliub.ttc",
    'TTCfontID' => array (
        'R' => 3,
    ),
),
),
...
);

```

This is an example of HTML code containing CJK characters from both BMP and SIP, and selecting the different fonts:

```

<div style="font-family:mingliu;"> &#40706; &#40742; &#40772; &#40784; &#40802; &#40809; &#x23289;
&#x2328a; </div>
<div style="font-family:mingliu_hkscs;"> &#40706; &#40742; &#40772; &#40784; &#40802; &#40809;
&#x23289; &#x2328a; </div>
<div style="font-family:pmingliu;"> &#40706; &#40742; &#40772; &#40784; &#40802; &#40809; &#x23289;
&#x2328a; </div>

```

OpenType layout features (OTL)

OpenType layout features were introduced in mPDF >= 6.0

Advanced Typography

Many TrueType fonts contain OpenType Layout (OTL) tables. These Advanced Typographic tables contain additional information that extend the capabilities of the fonts to support high-quality international typography:

- OTL fonts support ligatures, positional forms, alternates, and other substitutions.
- OTL fonts include information to support features for two-dimensional positioning and glyph attachment.
- OTL fonts contain explicit script and language information, so a text-processing application can adjust its behavior accordingly.

mPDF 6 introduces the power and flexibility of the OpenType Layout font model into PDF. mPDF supports GSUB, GPOS and GDEF tables for now. mPDF does *not* support BASE and JSTF at present.

Other mPDF features to enhance complex scripts:

- Bidirectional (Bidi) algorithm for right-to-left (RTL) text
- support for Kashida for justification of arabic scripts
- partial support for CSS3 optional font features e.g. font-feature-settings, font-variant
- improved "autofont" capability to select fonts automatically for any script
- support for CSS :lang selector
- dictionary-based line-breaking for Lao, Thai and Khmer (U+200B is also supported)
- separate algorithm for Tibetan line-breaking

Note: There are other smart-font technologies around to deal with complex scripts, namely Graphite fonts (SIL International) and Apple Advanced Typography (AAT by Apple/Mac). mPDF 6 does not support these.

What can OTL Fonts do?

Support for OTL fonts allows the faithful display of almost all complex scripts:

- Arabic (السلام عليكم), Hebrew (שלום לך), Syriac (ܫܠام ܥܠܝܟ)
- Indic - Bengali (শান্তিকুমাৰ), Devanagari (नमस्ते), Gujarati (નમસ્કાર), Punjabi (ਮਿਚੀ ਆਰਾਮ), Kannada (ನಮಸ್ತಾ), Malayalam (നമബൈ), Oriya (ନମସ୍କର), Tamil (வணக்கம்), Telugu (నమస్కారం)
- Sinhala (අග්‍රාහෙනු), Thai (ສົວສົ່ງ), Lao (ສະບາຍດີ), Khmer (ជິກຳປະສົງ), Myanmar (မံတ္တာပမ္မ), Tibetan (བཀྲ ། རྒྱ ། རྒྱ ། རྒྱ ། རྒྱ)

OTL features allow:

- Joining and Reordering
- Complex syllables
- Ligatures
- Language-dependent substitutions
- Font features - Optional substitutions Stylistic Alternatives (salt)
- CSS control of discretionary OTL features
- Mark repositioning (and diacritics)
- Mark repositioning (and Contextual substitution)
- Complex Typography An example which utilises many different GSUB and GPOS features together - first without GSUB and GPOS:
- Text Justification using Kashida

A full list of feature tags is at <http://www.microsoft.com/typography/otspec/featurelist.htm>

In mPDF, the following features are on by default:

- GSUB features: locl ccmp pref blwf abvf pstf pres abvs blws psts haln rlig calt liga clig mset (all scripts)

- GSUB features: isol fina fin2 fin3 medi med2 init nukt akhn rphf rkrf half vatu cjct cfar (for appropriate scripts e.g. Indic, Arabic)
- GPOS features: abvm blwm mark mkmk curs cpsp dist requ [kern]

NB 'requ' is not listed in the Microsoft registry of Feature tags; however it is found in the Arial Unicode MS font (it repositions the baseline for punctuation in Kannada script).

kern is used in some fonts to reposition marks etc. and is essential for correct display, so in mPDF kern is on by default when any non-Latin script is used.

Complex scripts require a "shaping engine" to re-order glyphs and apply the OTL features by syllable. MS Word and Wordpad run on the Windows platform use "Uniscribe", whereas some browsers such as FireFox and OpenOffice use Pango/HarfBuzz. The different shaping engines (and indeed different versions of them) can produce different results.

Different applications have different defaults (on/off) for some of the features e.g. kerning.

When testing mPDF, if text does not appear as you expect, ensure that the font is installed on your computer, and view the HTML in a browser. Also try copying/pasting the text into Wordpad/Word/OpenOffice and ensure that the correct font has been applied.

Note that Wordpad sometimes substitutes a different font if it does not like the one you have chosen, and does not even indicate that the substitution has occurred.

CSS control of font features

See <http://www.w3.org/TR/css3-fonts/#font-rend-props> for information about CSS3 and font-features.

The following are supported in mPDF:

- font-variant-position
- font-variant-caps
- font-variant-ligatures
- font-variant-numeric
- font-variant-alternates - Only [normal | historical-forms] supported (i.e. most are NOT supported)
e.g. stylistic, styleset, character-variant, swash, ornaments, annotation (use font-feature-settings for these)
- font-variant - as above, and except for: east-asian-variant-values, east-asian-width-values, ruby
- font-language-override
- font-feature-settings

font-variant-east-asian is NOT supported

NB @font-face is NOT supported

NB @font-feature-values is NOT supported

Note: font-variant specifies a single property in CSS2, whereas in CSS3 it has become a shorthand for all the other font-variant-* properties. font-variant: small-caps was the form supported in CSS2, and will work in mPDF.

See notes later about font kerning.

Examples:

```
/* use small-cap alternate glyphs */
.smallcaps { font-feature-settings: "smcp" on; }

/* convert both upper and lowercase to small caps (affects punctuation also) */
.allsmallcaps { font-feature-settings: "c2sc", "smcp"; }

/* enable historical forms */
.hist { font-feature-settings: "hist"; }

/* disable common ligatures, usually on by default */
.noligs { font-feature-settings: "liga" 0; }
```

```

/* enable tabular (monospaced) figures */
td.tabular { font-feature-settings: "tnum"; }

/* enable automatic fractions */
.fractions { font-feature-settings: "frac"; }

/* use the second available swash character */
.swash { font-feature-settings: "swsh" 2; }

/* enable stylistic set 7 */
.fancystyle {
font-family: Gabriola; /* available on Windows 7, and on Mac OS */
font-feature-settings: "ss07";
}

```

How to use OTL in mPDF

In config_fonts.php there are 2 variables which affect OTL features for each font family e.g.:

```

"dejavusanscondensed" => array(
    'R' => "DejaVuSansCondensed.ttf",
    'B' => "DejaVuSansCondensed-Bold.ttf",
    'I' => "DejaVuSansCondensed-Oblique.ttf",
    'BI' => "DejaVuSansCondensed-BoldOblique.ttf",
    'useOTL' => 0xFF,
    'useKashida' => 75,
),

```

useOTL

useOTL should be set to an integer between 0 and 255. Each bit will enable OTL features for a different group of scripts:

Bit	dec	hex	Enabled
1	1	0x01	GSUB/GPOS - Latin script
2	2	0x02	GSUB/GPOS - Cyrillic script
3	4	0x04	GSUB/GPOS - Greek script
4	8	0x08	GSUB/GPOS - CJK scripts (excluding Hangul-Jamo)
5	16	0x10	(Reserved)
6	32	0x20	(Reserved)
7	64	0x40	(Reserved)
8	128	0x80	GSUB/GPOS - All other scripts (including all RTL scripts, complex scripts etc)

Setting useOTL to 0 (or omitting it) will disable all OTL features. Setting useOTL to 255 or 0xFF will enable OTL for all scripts. Setting useOTL to 0x82 will enable OTL features for Cyrillic and complex scripts.

In a font like Free Serif, it may be useful to enable OTL features for complex scripts, but disable OTL for Latin scripts (to save processing time). However, see above - this may disable kerning in Latin scripts in certain circumstances.

useKashida

useKashida should be set for arabic fonts if you wish to enable text justification using kashida. The value should be an integer between 0 and 100 and represents the percentage of additional space required to justify the text on a line as a ratio of kashida/inter-word spacing.

Choosing fonts to add to mPDF 6

Fonts with OTL need to have GDEF, GSUB and GPOS tables in the font file. Although TrueType font files are binary files, the table names and script/feature tags are written as ASCII characters; open the .ttf or .otf file in a text editor such as Windows Notepad, and you will see GDEF, GSUB and GPOS in the first few lines if they are present. You can also search the file to see if the script tags are present for your desired scripts cf. <http://www.microsoft.com/typography/otspec/scripttags.htm>.

Note: The OTL specification for Indic fonts was updated in 2005 to version 2. The v2 script tag for Bengali is "bng2" whereas prior to this it was "beng". Many open-source font files are still written for the old specification.

This is supported by mPDF, although v2 fonts give better results.

Note: mPDF does not support Graphite or AAT font features.

Configuring new fonts for mPDF 6

To add a font, first copy the font file to the /ttfonts/ folder.

Then edit config_fonts.php to add. See the manual for details if you are not already familiar with this.

If you wish to use this font with autoLangToFont, you also need to edit config_lang2fonts.php

Setting OTL use at runtime

mPDF caches some font information in the /ttfontdata/ folder to improve performance. This is regenerated if you change the value of useOTL for a font.

There may be circumstances when you wish to use OTL features with different scripts depending on the document e.g. for everyday use you may want to disable OTL for FreeSerif to save processing time, but on occasions use OTL for Indic and/or Arabic scripts. The recommended way to do this is to create 2 instances of the font e.g. in config_fonts.php:

```
"freeserif" => array(
'R' => "FreeSerif.ttf",
'B' => "FreeSerifBold.ttf",
'I' => "FreeSerifItalic.ttf",
'BI' => "FreeSerifBoldItalic.ttf",
'useOTL' => 0x00,
),
"freeserif2" => array(
'R' => "FreeSerif.ttf",
'B' => "FreeSerifBold.ttf",
'I' => "FreeSerifItalic.ttf",
'BI' => "FreeSerifBoldItalic.ttf",
'useOTL' => 0xFF, /* Uses OTL for all scripts */
'useKashida' => 75,
),
```

You could then either use this second font name in your stylesheets e.g.

```
<p style="font-family:freeserif2;">Hallo World (in Arabic)</p>
```

or, you could use font translation e.g.

```
$mpdf->fonttrans['freeserif'] = 'freeserif2';
```

Font names

CSS font-family name

Every font family has a name which is defined in the font file. This is the name by which your computer OS registers and recognises the font family.

This is also the name used in CSS e.g.

```
<p style="font-family: 'Trebuchet MS';">
```

mPDF font-family name

This is the name used by mPDF internally to process fonts. This could be anything you like, but by default mPDF will convert CSS font-family names by removing any spaces and changing to lowercase. Reading the CSS name above, mPDF will look for a "mPDF font-family name" of 'trebuchetms'.

This means that this will also work:

```
<p style="font-family: trebuchetms;">
```

Next it will look for a translation in `$this->fonttrans` in the `config_fonts.php` file. Imagine that we also wished to recognise 'Trebuchet', we would add:

```
$this->fonttrans = array(
...
    'trebuchet' => 'trebuchetms',
...
)
```

mPDF font-family names should therefore always be lower-case and contain no spaces. When mPDF needs to refer to a specific variant (bold, italic etc.) it will use the mPDF font-family name (lowercase) followed by 'B', 'I', or 'BI' (uppercase). The regular/normal Trebuchet MS will be 'trebuchetms', and the bold variant will be referred to as 'trebuchetmsB'.

These mPDF font names are used in other places:

- all the other configurable variables in the `config_fonts.php` use the mPDF font-family name
- use the mPDF font name in the `config_cp.php` file to make it selectively available in certain languages.
- if used in the PHP script e.g. `$mpdf = new mPDF("","", "trebuchetms");`

Font file name

To make a font available to mPDF, you need to specify the Truetype .ttf font files for each variant.

These should be defined in `config_fonts.php` in the array:

```
$this->fontdata = array(
...
    "trebuchetms" => array(
        'R' => "trebuc.ttf",
        'B' => "trebucbd.ttf",
        'I' => "trebucit.ttf",
        'BI' => "trebucbi.ttf",
    ),
...
)
```

Each font-family must have a Regular ['R'] file defined - the others ([‘B’]old, [‘I’]italic, [‘BI’]bold-italic) are optional.

mPDF will try to locate the font-file. If you have defined `_MPDF_SYSTEM_TTFONTS` at the top of the `config_fonts.php` file, it will first look for the font-file there. This is useful if you are running mPDF on a computer which already has a folder with TTF fonts in (e.g. on Windows)

If the font-file is not there, or `_MPDF_SYSTEM_TTFONTS` is not defined, mPDF will look in the folder `/[your_path_to_mpdf]/ttfonts/`

Note that the font-file names are case-sensitive and can contain capitals.

Available Fonts v5.x

The following fonts are distributed with mPDF 5.x

General

DejaVuSans	Good general coverage of common languages
DejaVuSansCondensed	
DejaVuSansMono	
DejaVuSerif	
DejaVuSerifCondensed	

Thai

Garuda	Thai (sans-serif)
Norasi	Thai (serif)

Indic languages

ind_bn_1_001	Bengali
ind_hi_1_001	Devanagari
ind_ml_1_001	Malayalam
ind_gu_1_001	Gujarati
ind_kn_1_001	Kannada
ind_or_1_001	Oriya
ind_pa_1_001	Punjabi (Gurmukhi)
ind_ta_1_001	Tamil
ind_te_1_001	Telugu

The following font names can be used with mPDF 5.x

Adobe Asian font-pack

GB	Chinese (Simplified) Adobe Asian font-pack containing all characters in codepage GBK (extension of GB2312)
BIG5	Chinese (Traditional) Adobe Asian font-pack containing all characters in codepage BIG-5
UHC	Korean Adobe Asian font-pack containing all characters in codepage UHC (cp949)
SJIS	Japanese Adobe Asian font-pack containing all characters in codepage SHIFT_JIS

Generic font-family

sans	sans-serif, serif and monospace are recognised generic font-families specified by CSS. These are
sans-serif	recognised by mPDF - defined in config_fonts.php
serif	
monospace	
mono	

For a list of languages covered by the fonts included with mPDF, see [Fonts and Language cover](#)

Available fonts v6

The following fonts are included with mPDF 6:

Font(s)	Download URL	Copyright / License	Coverage
DejaVuSans DejaVuSansCondensed DejaVuSerif DejaVuSerifCondensed DejaVuSansMono	http://dejavu-fonts.org	© Bitstream http://dejavu-fonts.org/wiki/License	[Numerous]
FreeSans FreeSerif FreeMono	http://www.gnu.org/software/freefont/	GNU GPL v3	[Numerous incl. Indic]
Quivira	http://www.quivira-font.com/	free for any use	Coptic Buhid Tagalog Tagbanwa Lisu
Abyssinica SIL	http://www.sil.org/resources/software_fonts/abyssinica-sil	SIL Open Font License	Ethiopic
XBRiyaz	(XW Zar fonts) http://wiki.irmug.org/index.php/XWZar	SIL Open Font License	Arabic
Taamey David CLM	http://opensiddur.org/tools/fonts/	GNU GPL 2	Hebrew
Estrangelo Edessa	http://www.bethmardutho.org/index.php/resources/fonts.html (SyrCOMEdessa.otf)	Adapted licence (free to use/share)	Syriac
Aegean	http://users.teilar.gr/~g1951d/	free for any use	Carian Lycian Lydian Phoenecian Ugaritic Linear B Old Italic
Jomolhari	https://sites.google.com/site/chrisfynn2/home/fonts/jomolhari	SIL Open Font License	Tibetan
Lohitkannada	https://fedorahosted.org/lohit/	SIL Open Font License	Kannada
Kaputaunicode	http://www.kaputa.com/slword/kaputaunicode.htm http://www.locallanguages.lk/sinhala_unicode_converters	Free Sri Lanka Web Community Center	Sinhala
Pothana2000	https://fedoraproject.org/wiki/Pothana2000_fonts	GNU GPL v2+	Telugu
Lateef	https://sil.org/resources/software_fonts/lateef	SIL Open Font License	Sindhi
Khmeros	http://www.khmeros.info/en/fonts (http://www.cambodia.org/fonts/)	LGPL Licence	Khmer
Dhyana	Google Fonts http://www.google.com/fonts/earlyaccess	SIL Open Font License	Lao
Tharlon	Google Fonts http://code.google.com/p/tharlon-font/	SIL Open Font License	Myanmar Tai Le
Padauk Book	http://www.sil.org/resources/software_fonts/padauk	SIL Open Font License	Myanmar
Ayar fonts	http://eng/ayarunicodegroup.org/	SIL Open Font License	Myanmar
ZawgyiOne	http://code.google.com/p/zawgyi/wiki/MyanmarFontDownload	Freely available. No licence information available	Myanmar
Garuda	http://www.hawaii.edu/thai/thaifonts/	Freely available. No licence information available	Thai
Sundanese Unicode	http://sabilulungan.org/aksara/	GNU GPL	Sundanese
Tai Heritage Pro	http://www.sil.org/resources/software_fonts/tai-heritage-pro	SIL Open Font License	Tai Viet
Sun-ExtA Sun-ExtB	http://www.alanwood.net/downloads/index.html	Freeware (Beijing ZhongYi Electronics Co)	Chinese Japanese Runic
Unbatang	http://kldp.net/projects/unfonts/download	GNU GPL	Korean
Aboriginal Sans	http://www.languagegeek.com/font/fontdownload.html	GNU GPL 3	Cree Canadian Aboriginal Inuktuit
MPH 2B Damase	http://www.alanwood.net/downloads/index.html	(Public domain)	Glagolitic Shavian Osmanya Kharoshthi Deseret
Aegyptus	http://users.teilar.gr/~g1951d/	free for any use	Egyptian Hieroglyphs
Akkadian	http://users.teilar.gr/~g1951d/	free for any use	Cuneiform
Eeyek Unicode	http://tabish.freeshell.org/eeyek/download.html	Freeware	Meetei Mayek

Font(s)	Download URL	Copyright / License	Coverage
Lannaalif	http://www.geocities.jp/simsheart_alif/taithamunicode.html	(Unclear)	Tai Tham
Daibanna SIL Book	http://www.sil.org/resources/software_fonts/dai-banna-sil	SIL Open Font License	New Tai Lue
KFGQPC Uthman Taha Naskh	http://fonts.qurancomplex.gov.sa/?page_id=42	https://www.ohloh.net/licenses/KFGQPC	Arabic (Koran/Quran)

The following font names can also be used with mPDF:

Adobe Asian font-pack

GB	Chinese (Simplified) Adobe Asian font-pack containing all characters in codepage GBK (extension of GB2312)
BIG5	Chinese (Traditional) Adobe Asian font-pack containing all characters in codepage BIG-5
UHC	Korean Adobe Asian font-pack containing all characters in codepage UHC (cp949)
SJIS	Japanese Adobe Asian font-pack containing all characters in codepage SHIFT_JIS

Generic font-family

sans	sans-serif, serif and monospace are recognised generic font-families specified by CSS.
sans-serif	These are recognised by mPDF - cf. config_fonts.php
serif	
monospace	
mono	

Choosing a configuration v5.x

(mPDF >= 5.0)

In mPDF there are a number of ways to configure your set-up. There is often a trade-off between file size, processing speed, appearance (support for different fonts), and reliability (i.e. ensuring that text is always displayed, at least in some form).

Some of the things you can change are:

- the initial parameter used to call mPDF e.g. `$mpdf=new mPDF('c')`
- configuration variables set in config.php see [mPDF Variables - Overview](#)
(many of those configuration variable can also be set at runtime on a 'per-script' basis)
- font details in config_fonts.php
- language/font details in config_cp.php
- initial style settings in config.php or mpdf.css

Main choices

Some of the major considerations are:

- whether to restrict the document to core non-embedded fonts
- whether, or how, to subset embedded fonts
- handling of languages which require special fonts, including auto_font and character substitution

Core non-embedded fonts

PDF files have certain standard fonts: Arial/Helvetica, Times and Courier in the win-1252 character set, and Zapfdingbats and Symbol character sets. These fonts should be available to any PDF reading program, and do not need to be embedded in the PDF document.

Advantages: Small file size, fast processing, small memory usage.

Disadvantages: Limited choice of fonts for appearance. Will not display characters which are not in the [win-1252 Symbols](#), or [Dingbats](#) codepages (suitable for most Western European languages).

To use core fonts only, use 'c' for the initial parameter:

```
$mpdf = new mPDF('c');
```

Embedded Unicode fonts

The alternative (the default setup) uses TrueType Unicode fonts, and the only limitation of characters to display is determined by the font files themselves.

Subsetting fonts

Fonts with good coverage of all characters you may require can be very large. If you embed the whole font file in the PDF document, the file can become very large - especially if you use a number of fonts. mPDF can embed subsets of the fonts i.e. just including the characters used in the PDF document.

Advantages of subsetting: Manageable file size (typically between 20-200kB)

Disadvantages of subsetting: Increase in processing time and memory usage (not always), as mPDF has to rebuild font files for each document.

By default, mPDF will embed subsets of fonts if less than 30% of the characters contained in the font are used in the document; otherwise it will embed the whole font file. You can override this by changing the configurable variable `$this->percentSubset`

For backwards compatibility, you can use '-s' or 's' in the initial parameter to override the config.php settings and force subsetting of all fonts e.g.

```
$mpdf = new mPDF('s');
$mpdf = new mPDF('ar-s'); // also defining arabic language code
$mpdf = new mPDF('utf8-s'); // for backwards compatibility - the utf-8 does nothing
```

Using core non-embedded fonts in a Unicode document

If your document uses Unicode fonts as above, you can force mPDF to use the core (non-embedded) PDF fonts in parts of the document by selecting the fontnames: `chelvetica`, `ccourier` and `ctimes` e.g.

```
<p style="font-family:chelvetica">This paragraph will use core fonts</p>
```

You could force mPDF to always use core fonts when Arial/Helvetica/Courier are specified, by editing the font translation variable `$this->fonttrans` in `config_fonts.php` e.g.:

```
$this->fonttrans = array(
    'arial' => 'chelvetica',
    'helvetica' => 'chelvetica',
    'timesnewroman' => 'ctimes',
    'times' => 'ctimes',
    'couriernew' => 'ccourier',
    'courier' => 'ccourier',
```

Languages which require special fonts

Indic languages always require special handling (cf. [Indic fonts](#)). Other than this, whether you need to do anything special is determined by the choice of fonts you use in the document, and whether they contain the necessary characters to display your text. The DejaVu fonts distributed with mPDF will usually display most Western and Eastern European languages, Cyrillic text, Baltic languages, Turkish, and Greek. Languages which usually need special consideration are: CJK (chinese - japanese - korean) languages, Vietnamese, Thai, and Arabic languages. With these, you need to tell mPDF to select a suitable font.

There are a number of different ways to do this:

- 1) Write your HTML code to specify the exact fonts needed:

```
<p style="font-family: Garuda">ເປົ້າມນຸ່ງຍື່ງສຸດປະເວົ້າລີຄຄູນຄ່າ</p>
<p style="font-family: BIG5">仝ວ້າ阿哀愛挨始</p>
<p style="font-family: sun-exta">仝ວ້າ阿哀愛挨始</p>
<p style="font-family: 'XB Riyaz'">البرادعي البرادعي</p>
<p style="font-family: ind_hi_1_001">ພහລາ ພັນາ</p>
```

- 2) Use the `lang` attribute to define the language. This causes mPDF to use a font as specified in the `config_cp.php` file. The current font-family will be used if it is available in the list defined as `$unifonts`, otherwise the first font specified in the `$unifonts` list will be selected.

```
<p lang="th">ເປົ້າມນຸ່ງຍື່ງສຸດປະເວົ້າລີຄຄູນຄ່າ</p>
<p lang="zh-CN">仝ວ້າ阿哀愛挨始</p>
<p lang="ar">البرادعي البرادعي</p>
<p lang="hi">ພහລາ ພັນາ</p>
```

- 3) If you set the language in the initial call to mPDF i.e.

```
$mpdf = new mPDF('th');
$mpdf = new mPDF('zh-CN'); // You can append +aCJK or -aCJK
$mpdf = new mPDF('ar');
$mpdf = new mPDF('hi');
```

then mPDF will restrict the fonts you can use for the whole document. Any of the fonts defined in `$unifonts` can be used:

```
// If config_cp.php defines:
CASE "th": $spacing = "C";
$unifonts = "garuda,garudaB,garudaI,garudaBI,norasi,norasiB,norasiI,norasiBI";
break;
```

These paragraphs will appear in Garuda (a sans-serif font) and Norasi (serif) respectively:

```
<p style="font-family:sans">เป็นมุขย์สุดประเสริฐเลิศคุณค่า</p>
<p style="font-family:serif">เป็นมุขย์สุดประเสริฐเลิศคุณค่า</p>
```

4) You can also set the language for the whole document by setting:

```
<body lang="th">
<body lang="zh-CN">
<body lang="ar">
<body lang="hi">
```

5) Use [SetAutoFont\(\)](#) to automatically detect these languages. AutoFont inspects the HTML code and inserts a span element to mark text which is auto-detected e.g.

```
$mpdf->SetAutoFont(AUTOFONT_ALL);
```

```
<p>This is Thai text: เป็นมุขย์สุดประเสริฐเลิศคุณค่า</p>
```

becomes:

```
<p>This is Thai text: <span lang="th" class="lang_th">เป็นมุขย์สุดประเสริฐเลิศคุณค่า</span></p>
```

The lang attribute means that mPDF will select a font as described above in 2). You can also use CSS stylesheets to apply additional styles e.g. changing the font-size.

Do not use AutoFont as well as methods (1) - (4).

6) Use [\\$useSubstitutions](#) to use character susbtitution. mPDF will inspect every character in the HTML code, and if the character is not represented in the current font, it will try to substitute it from one of the fonts defined in `$this->backupSubsFont` in config_fonts.php.

```
$this->backupSubsFont = array('dejavusanscondensed','norasi');
```

Which method should I use?

Let us call Western and Eastern European languages, Cyrillic text, Baltic languages, Turkish, and Greek - **common languages**, and CJK and Indic languages, Vietnamese, Thai, and Arabic - **special languages**.

Individually authored pages

If you are authoring a document in a common language which contains sections of text using special languages, the ideal method to use is 1) or 2).

Document in a Special language

If you are writing documents entirely in a special language, you should use method 3) or 4), but you will be restricted to selecting from the fonts defined in config_cp.php

Multilingual documents 'on-the-fly'

If you are creating a PDF document from a page such as this web-page or a forum board which is likely to contain some special language text, and it is not possible to mark-up the special language text, you should use method 5) AutoFont.

If the document is mainly in a common language and may contain only occasional words or characters, the alternative is to use method 6) character substitution. This will be slower than 5) if there are substantial sections in special languages.

It is possible to use methods 5) and 6) together, to ensure that all special characters are displayed.

Multilingual website

If you are using a multilingual website (Wiki?) which has document in different languages, but where each page will be in only one language, and you have a common stylesheet etc. you can use method 3). Use the language code (en-GB, zh-CN) from the website page selected to set up the mPDF class.

If you want to use core non-embedded fonts when possible (for Western European languages), you can add '-x' to the language string. This will select core fonts only when the language string is appropriate (as defined in

config_cp.php) e.g.

```
$mpdf = new mPDF('en-GB-x'); // will only use core non-embedded fonts
$mpdf = new mPDF('de-x'); // will only use core non-embedded fonts (German)
$mpdf = new mPDF('ar-x'); // behaves as though ('ar') called (Arabic)
$mpdf = new mPDF('ru-x'); // behaves as though ('ru') called (Russian)
```

See Also

- [RTL & Bidirectional text](#)
- [SetAutoFont\(\)](#) - Automatically detect language in the input HTML text and use appropriate fonts

There is a useful list of language/country codes at: <http://www.i18nguy.com/unicode/language-identifiers.html>.

Choosing a configuration v6.x

(mPDF >= 6.0)

In mPDF there are a number of ways to configure your set-up. There is often a trade-off between file size, processing speed, appearance (support for different fonts), and reliability (i.e. ensuring that text is always displayed, at least in some form).

Some of the things you can change are:

- the initial parameter used to call mPDF e.g. `$mpdf=new mPDF('c')`
- configuration variables set in config.php see [mPDF Variables - Overview](#)
(many of those configuration variable can also be set at runtime on a 'per-script' basis)
- font details in config_fonts.php
- language/font details in config_lang2fonts.php
- initial style settings in config.php or mpdf.css

Main choices

Some of the major considerations are:

- whether to restrict the document to core non-embedded fonts
- whether, or how, to subset embedded fonts
- handling of languages which require special fonts, including automatic font selection and character substitution

Core non-embedded fonts

PDF files have certain standard fonts: Arial/Helvetica, Times and Courier in the win-1252 character set, and Zapfdingbats and Symbol character sets. These fonts should be available to any PDF reading program, and do not need to be embedded in the PDF document.

Advantages: Small file size, fast processing, small memory usage.

Disadvantages: Limited choice of fonts for appearance. Will not display characters which are not in the [win-1252 Symbols](#), or [Dingbats](#) codepages (suitable for most Western European languages).

To use core fonts only, use 'c' for the initial parameter:

```
$mpdf = new mPDF( 'c' );
```

Embedded Unicode fonts

The alternative (the default setup) uses TrueType Unicode fonts, and the only limitation of characters to display is determined by the font files themselves.

Subsetting fonts

Fonts with good coverage of all characters you may require can be very large. If you embed the whole font file in the PDF document, the file can become very large - especially if you use a number of fonts. mPDF can embed subsets of the fonts i.e. just including the characters used in the PDF document.

Advantages of subsetting: Manageable file size (typically between 20-200kB)

Disadvantages of subsetting: Increase in processing time and memory usage (not always), as mPDF has to rebuild font files for each document.

By default, mPDF will embed subsets of fonts if less than 30% of the characters contained in the font are used in the document; otherwise it will embed the whole font file. You can override this by changing the configurable variable `$this->percentSubset`

For backwards compatibility, you can use 's' in the initial parameter to override the config.php settings and force subsetting of all fonts e.g.

```
$mpdf = new mPDF( 's' );
```

Using core non-embedded fonts in a Unicode document

If your document uses Unicode fonts as above, you can force mPDF to use the core (non-embedded) PDF fonts in parts of the document by selecting the fontnames: `chelvetica`, `ccourier` and `ctimes` e.g.

```
<p style="font-family:chelvetica">This paragraph will use core fonts</p>
```

You could force mPDF to always use core fonts when Arial/Helvetica/Courier are specified, by editing the font translation variable `$this->fonttrans` in `config_fonts.php` e.g.:

```
$this->fonttrans = array(
    'arial' => 'chelvetica',
    'helvetica' => 'chelvetica',
    'timesnewroman' => 'ctimes',
    'times' => 'ctimes',
    'couriernew' => 'ccourier',
    'courier' => 'ccourier',
```

Languages/Scripts which require special fonts

Most browsers / PC applications automatically select / substitute appropriate fonts when required. mPDF does not do this by default (additional processing resources required). there are two situations when you need to consider the method of font selection:

- the text contains characters which are not covered by the initial font selected
- the text contains complex script requiring special treatment i.e. [OpenType layout \(OTL\)](#)

The DejaVu fonts distributed with mPDF contain characters (glyphs) to display most Western and Eastern European languages, Cyrillic text, Baltic languages, Turkish, and Greek. Languages which usually need special consideration are: CJK (chinese - japanese - korean) languages, Indic languages, Vietnamese, Thai, and Arabic languages. With these, you need to tell mPDF to select a suitable font.

There are several different ways to do this:

- 1) Write your HTML code to specify the exact fonts needed:

```
<p style="font-family: Garuda">ເປົ້າມນຸ່ຍໍສຸດປະເວົ້າລືບຄຸນຄ່າ</p>
<p style="font-family: BIG5">全娃阿哀愛挨始</p>
<p style="font-family: sun-exta">全娃阿哀愛挨始</p>
<p style="font-family: 'XB Riyaz'">البرادعى البرادعى</p>
```

- 2) Write your HTML code using the `lang` attribute to define the language.

```
<p lang="th">ເປົ້າມນຸ່ຍໍສຸດປະເວົ້າລືບຄຸນຄ່າ</p>
<p lang="zh-CN">全娃阿哀愛挨始</p>
<p lang="ar">البرادعى البرادعى</p>
<p lang="hi">ପହଳା ପନ୍ତା</p>
```

This needs to be used in conjunction with either:

- `autoLangToFont`
- CSS stylesheet using the `:lang` selector

- 3) Use `autoScriptToLang` to mark up HTML text by inserting the `lang` attribute, based on the Unicode script block in question, and configurable values in `config_script2lang.php`

```
$mpdf->autoScriptToLang = true;
```

As for (2) this needs to be used in conjunction with either:

- `autoLangToFont`
- CSS stylesheet using the `:lang` selector

- 4) Use `$useSubstitutions` to use character substitution. mPDF will inspect every character in the HTML code, and if the character is not represented in the specified font, it will try to substitute it from one of the fonts

defined in `$this->backupSubsFont` in `config_fonts.php`.

```
$this->backupSubsFont = array('dejavusanscondensed', 'arialunicodems');
```

Which method should I use?

Individually authored pages

If you are authoring a document in a common language which contains sections of text using special languages, the ideal method to use is 1) or 2).

Multilingual documents 'on-the-fly'

If you are creating a PDF document from a page such as this web-page or a forum board which is likely to contain some special language text, and it is not possible to mark-up the special language text, you should use method 3).

If the document is mainly in a common language and may contain only occasional words or characters, the alternative is to use method 4) character substitution.

It is possible to use method 4) together with 1) 2) or 3), to ensure that all special characters are displayed.

See Also

- [RTL & Bidirectional text](#)

Automatic font selection

Note: This describes automatic font selection in mPDF >= v6.

mPDF has two functions which can be used together or separately:

`autoScriptToLang` - marks up HTML text using the `lang` attribute, based on the Unicode script block in question, and configurable values in `config_script2lang.php`.

`autoLangToFont` - selects the font to use, based on the HTML `lang` attribute, using configurable values in `config_lang2font.php`.

For automatic font selection, ideally we would choose the font based on the language in use. However it is actually impossible to determine the language used from a string of HTML text. The Unicode script block can be ascertained, and sometimes this tells us the language e.g. Telugu. However, Cyrillic script is used for example in many different languages. So the best we can do is base it on the script used. However, mPDF does this in two stages via the `lang` attribute, because this allows the options of using either of the stages alone or together:

```
<p>English русский язык</p>
```

↓ **autoScriptToLang** (`config_script2lang.php`) ↓

```
<p>English <span lang="und-Cyrl">русский язык</span> <span lang="ps">پښتو</span></p>
```

↓ **autoLangToFont** (`config_lang2fonts.php`) ↓

Uses "lang" to select font, and to determine OTL features applied

autoScriptToLang

```
$mpdf->autoScriptToLang = true;
$mpdf->baseScript = 1;
$mpdf->autoVietnamese = true;
$mpdf->autoArabic = true;
```

`$mpdf->baseScript = 1`; tells mPDF which Script to ignore. It is set by default to "1" which is for Latin script. In this mode, all scripts except Latin script are marked up with `lang` attribute. To select other scripts as the base, see the file `/classes/ucdn.php`

Using `autoScriptToLang`, mPDF detects text runs based on Unicode script block; using the values in `config_script2lang.php` it then encloses the text run within a `span` tag with the appropriate language attribute. For many scripts, the language cannot be determined: see the example above which recognises Cyrillic script and marks it up using `und-Cyrl`, which is a valid IETF tag, coding for language="undetermined", `script="Cyrillic"`.

Two optional refinements are added: Vietnamese text can often be recognised by the presence of certain characters which do not appear in other Latin script languages, and similarly analysis of the text can attempt to distinguish Arabic, Farsi, Pashto, Urdu and Sindhi. If active, the text will then be marked with a specific language tag e.g. "vi", "pa", "ur", "fa" etc.

These features can be disabled or enabled (default) using the variables `$mpdf->autoVietnamese` `$mpdf->autoArabic`, either in `config.php` or at runtime.

autoLangToFont

```
$mpdf->autoLangToFont = true;
```

You can edit the values in `config_lang2font.php` to specify which fonts are used for which `lang`.

Using text with multiple languages

Recommended ways to use multiple languages in mPDF:

1. If you have full control over the HTML, mark-up the text with the `lang` attribute and use CSS (`:lang` selector preferably); this method means that the language information can also be used by OTL for language dependent substitutions.
2. If you have no control over (user) HTML input and want to output faithfully, use both `autoScriptToLang` and `autoLangToFont`

It is preferable not to use `autoScriptToLang` and `autoLangToFont` unless they are necessary: they will result in increased processing time, and OTL tables will not be able to use language dependent substitutions when undefined languages are set e.g "und-Cyrl".

Note: As from mPDF 6.0 automatic font selection can be used within an SVG image. Control is separate from that of the rest of the document. For details, see the defined constants at the top of `classes/svg.php` file.

Using the HTML attribute - lang v5.x

When creating a multilingual document containing Arabic, Indic, **cjk**, Vietnamese or Thai languages, the text in these languages needs to be identified in order to correctly display them using appropriate fonts.

Marking up your HTML text using the lang attribute is one way to do this. The attribute can be used on any block or in-line tag e.g. `<div lang="zh-CN">....</div>` or `...`. It is also recognised in `<body>` or `<html>` tags.

Important: Support for the attribute *lang* is only active when the variable `$useLang` is set to **TRUE** (**DEFAULT**).

The language codes supported are determined by `config_cp.php` :

- en, ca, cy, da, de, es, eu, fr, ga, fi, is, it, nl, no, pt, sv
- cs, hr, hu, pl, ro, sk, sl
- bg, mk, ru, sr, uk
- et, kl, lt, lv
- el, tr
- vi, th
- CJK: ja, zh, zh-CN, zh-HK, zh-TW, ko
- RTL: he, ar, fa, ps, ur, sd
- INDIC: as, bn, gu, hi, kn, ks, ml, ne, or, pa, sd-IN, ta, te

See [ISO 639-1 language codes](#).

Codes such as "en-US" are supported, and interpreted as "en"

When a language is set, the following happens (determined by function `GetCodepage()`):

- available fonts are restricted to those which contain the necessary glyphs to display this language
- justification type (`$jSpacing`) is set when a block tag has the property `text-align:justify`; for span/inline elements the justification is disabled for the enclosing block element.

Note: Automatic language detection using `SetFont()` will detect language use in the text passed to mPDF, and add `..` tags to mark up. Using automatic language detection may add to the processing time when creating a large document.

See Also

- `useLang` - Specify whether to recognise and support the HTML attribute lang
- `SetFont()` - Use AutoFont to auto-detect text language in HTML input
- `autoFontGroupSize` - Specify the text chunk size to group when autodetecting text language

lang HTML attribute v6.x

The HTML lang attribute has a number of uses:

- when OTL tables are being used for a font, the language from the lang attribute is used to select which OTL features are applied;
- used in conjunction with CSS :lang selector to allow CSS styles to be applied;
- can be used in conjunction with autoLangToFont and autoScriptToLang (cf.)

mPDF supports use of the lang selector in CSS. All of the following are supported:

- :lang(fr)
- p:lang(fr)
- span:lang("syr")
- [lang="fr"]
- [lang='fr']
- p[lang=fr]
- p[lang="zh-TW"]

Note: [lang=zh] will match lang="zh-TW" and lang="zh-HK"

Limitation: class selectors and attribute selectors should be of equal specificity in CSS specification e.g.

```
:lang(syr) { color: blue; }
.syriac { color: red; }
```

should be of equal specificity, and thus apply whichever comes later in the CSS stylesheet. mPDF however gives :lang priority over .class

The use of the lang attribute and CSS selector is the recommended method for handling multi-lingual documents

Language tags

IETF tags should be used for lang which comply with the following:

- a 2 or 3 letter Language code, followed optionally by
- a hyphen and a 4 letter Script code, and or
- a hyphen and a 2 letter Region code
- i.e. [xx|xxx]{-Xxxx}{-XX}
- mPDF deals with IETF tags as case insensitive

Input encoding

mPDF accepts UTF-8 encoded text by default for all functions.

You can use the following to allow you to write html code encoded in other than utf-8 (in functions like `WriteHTML()`):

```
$mpdf->allow_charset_conversion=true; // Set by default to TRUE  
$mpdf->charset_in='windows-1252';
```

Note: `charset_in` requires codes recognised by the PHP function `iconv` i.e. windows-1252 not win-1252

If `allow_charset_conversion` is **TRUE** mPDF will also read the charset from the HTML header if present e.g.

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

Alternatively, you could convert the html to utf-8 encoding before passing it to mPDF, using any one of the PHP functions:

- `utf8_encode($ansi_encoded_html)` // only converts ISO-8859-1 to UTF-8
- `iconv('windows-1252', 'UTF-8', $ansi_encoded_html)`
- `mb_convert_encoding($ansi_encoded_html, 'UTF-8', 'windows-1252')`

Note the different order of the parameters, and the different codepage names used by the different functions. The codepage names recognised vary from platform to platform, and your PHP configuration.

A list of codepages recognised by `iconv()` can be found at <http://www.gnu.org/software/libiconv/>

In PHP5 you can list the codepages recognised by mb_functions using `mb_list_encodings()`.

Also note that each function has different ways of failing if illegal characters are encountered.

Fonts & Language cover v5.x

The following Open Source fonts are included with mPDF 5:

- *DejaVuSans*, *DejaVuSansCondensed*, *DejaVuSansMono*, *DejaVuSerif*, *DejaVuSerifCondensed*
- *Garuda* - Thai (sans-serif)
- *Norasi* - Thai (serif)
- *XBZar*, *XBRiyaz* - Arabic

Coverage of included fonts

		DejaVuSans (+condensed)	DejaVuSansMono	DejaVuSerif (+condensed)	Garuda (sans)	Norasi (serif)	XBZar	XBRiyaz
Western European: English, French, German, Spanish, Icelandic etc.	win-1252	Y	Y	Y	Y	Y	Y	Y
Central and Eastern Europe: Polish, Hungarian, Romanian	iso-8859-2	Y	Y	Y	-	-	-	-
Cyrillic: Russian, Bulgarian, Macedonian, Ukrainian etc.	win-1251	Y	Y	Y	-	-	-	-
Baltic: Latvian, Lithuanian, Estonian, Greenlandic	iso-8859-4	Y	Y	Y	-	-	-	-
Turkish	iso-8859-9	Y	Y	Y	-	-	-	-
Greek	iso-8859-7	Y	Y	Y	-	-	-	-
Arabic		Y	Y	-	-	-	Y	
Persian / Farsi		Y	Y	-	-	-	Y	
Urdu		[94%]	[97%]	-	-	-	Y	
Pashto		[83%]	[75%]	-	-	-	Y	
Hebrew		Y	-	-	-	-	-	
Thai		-	-	-	Y	-	-	
Vietnamese		Y	[62%]	[77%]	-	-	-	

Arabic (RTL) text v5.x

Note: Handling of RTL (right-to-left) languages was significantly rewritten for mPDF v5.1

Document Directionality - RTL versus LTR

The document has a baseline direction which is **LTR** or **RTL**; this determines:

- text alignment in blocks for which text-align has not been specifically set
- layout of mirrored page-margins, columns, ToC and Indexes, headers / footers

This base/document directionality is **LTR** by default, and can be set by any of the following:

```
$mpdf->SetDirectionality('rtl');
<html dir="rtl"> or <html style="direction: rtl;">
<body dir="rtl"> or <body style="direction: rtl;">
```

Base direction is an inherited CSS property, so will affect all content, unless direction is specified elsewhere.

Block-level Directionality

Direction can be set for any HTML block elements e.g. <div><p><table> etc using:

```
[HTML]
<div style="direction: rtl;">
or
[CSS stylesheet]
div.right { direction: rtl; }
```

Block-level direction *may* affect text alignment, and will also influence text reversal in **RTL** text.

Note that margin/padding are NOT reversed by direction i.e. left-margin will still be left-margin in **RTL** state.

Text alignment

The default value for text-align is "a nameless value which is dependent on direction". However, once text-align is specified, it is respected and inherited by all descendants.

Directionality in Tables

- direction can only be set on the top-level element of nested lists
- direction can only be set on <table>, NOT on <thead><tbody><td> etc.
- nested tables CAN have different directions

Text Bidirectionality

mPDF analyses any mixed text which contains **RTL** text. The text between HTML tags is divided into "chunks" of **LTR** and **RTL** text.

RTL text chunks are reversed (both letter- and word-order).

If (and only if) the direction of the block is **LTR** then the order of the chunks is reversed as well so that the sentence order is **RTL**.

This process when **RTL** arabic characters are present is fully automatic and unconfigurable. <bdo> etc has no effect.

However enclosing text in silent tags can sometimes help by altering the way the text is broken up into chunks to process e.g.:

```
english text <span>[arabic text]</span> english text
```

Fonts

Arabic is a complex script requiring processing before output. However any appropriate font can be used - as long as it contains the characters in Unicode blocks 'Arabic Presentation Forms' A and B (U+FB50 - U+FDFF, U+FE70 - U+FEFE). Note that quite a large number of fonts contain the isolated characters but not the presentation forms.

2 fonts are bundled with mPDF: XB Zar and XB Riyaz. These are 2 of a number of fonts available from

http://wiki.irmug.com/index.php/X_Series_2.

Note: The script handling Arabic text (RTL) was rewritten in mPDF 5.5 with improved support for Pashto/Sindhi/Urdu/Kurdish, especially for joining characters and added new presentation forms.

Non-unicode characters

Some characters in Pashto/Sindhi/Urdu/Kurdish do not have Unicode values for the final/initial/medial forms of the characters. However, some fonts include glyphs for these characters "un-mapped" to Unicode (including XB Zar and XB Riyaz, which are bundled with mPDF).

By editing config_fonts.php and adding to appropriate fonts:

```
'unAGlyphs' => true,
```

this will force mPDF to use unmapped glyphs. It requires the font file to include a Format 2.0 POST table which references the glyphs by name as e.g. uni067C.med or uni067C.medi

XB Riyaz, XB Zar, Arabic Typesetting (MS), Arial (MS) all contain this table. NB If you want to know if a font file is suitable, you can open a .ttf file in a text editor and search for "uni067C.med" - if it exists, it may work!

Using "unAGlyphs" forces subsetting of fonts, and will not work with SIP/SMP fonts (using characters beyond the Unicode BMP Plane).

mPDF maps these characters to part of the Private Use Area allocated by Unicode U+F500-F7FF. This could interfere with correct use if the font already utilises these codes (unlikely).

Alef Maksura

Detailed note on the Alef Maksura for advanced users:

U+0649 Alef Maksura only normally appears at the end of a word (in Arabic)

Initial and Medial forms exist in Unicode as FBE8 and FBE9 but are not in most fonts

So the final form is set in mPDF to show as FEF0; Initial and medial forms are shown as isolated/final, so that it does at least display.

It seems that Initial and Medial forms are used in Koranic text.

I have left options encoded in *function InitArabic()* if you want to alter - to make it double-joining, it also needs to be added to \$arabPrevLink as "\xd9\x89"

Note: mPDF deletes Unicode characters: U+200C,U+200D,U+200E,U+200F zero-width joiner/non-joiner, LTR and RTL marks so they will not appear - even though some fonts contain glyphs for these characters.

See Also

- [useLang](#) - Specify whether to recognise and support the HTML attribute lang
- [SetFont\(\)](#) - Use AutoFont to auto-detect text language in HTML input
- [autoFontGroupSize](#) - Specify the text chunk size to group when autodetecting text language
- [disableMultilingualJustify](#) - Specify whether to disable text justification in multilingual documents
- [lang](#) - Information on mPDF support for the HTML attribute lang

Bidirectional (RTL) text v6.x

Document Directionality - RTL versus LTR

The document has a baseline direction which is **LTR** or **RTL**; this determines:

- text alignment in blocks for which `text-align` has not been specifically set
- layout of mirrored page-margins, columns, ToC and Indexes, headers / footers

This base/document directionality is **LTR** by default, and can be set by any of the following:

```
$mpdf->SetDirectionality('rtl');
<html dir="rtl"> or <html style="direction: rtl;">
<body dir="rtl"> or <body style="direction: rtl;">
```

Base direction is an inherited CSS property, so will affect all content, unless direction is specified elsewhere.

Block-level Directionality

Direction can be set for any HTML block elements e.g. `<div><p><table>` etc using:

```
[HTML]
<div style="direction: rtl;">
or
[CSS stylesheet]
div.right { direction: rtl; }
```

Block-level direction *may* affect text alignment, and will also influence text reversal in **RTL** text.

Note that margin/padding are NOT reversed by direction i.e. left-margin will still be left-margin in **RTL** state.

Text alignment

The default value for `text-align` is "a nameless value which is dependent on direction". However, once `text-align` is specified, it is respected and inherited by all descendants.

Text Bidirectionality

Note: OpenType layout (OTL) features must be enabled on a font for it to display right-to-left script.

Bi-directional text is supported in mPDF.

1) The following Unicode characters are supported, and can be inserted directly in the text as HTML entities:

LRE	U+202A	LEFT-TO-RIGHT EMBEDDING	‪
RLE	U+202B	RIGHT-TO-LEFT EMBEDDING	‫
LRO	U+202D	LEFT-TO-RIGHT OVERRIDE	‭
RLO	U+202E	RIGHT-TO-LEFT OVERRIDE	‮
PDF	U+202C	POP DIRECTIONAL FORMATTING	‬
LRI	U+2066	LEFT-TO-RIGHT ISOLATE	⁦
RLI	U+2067	RIGHT-TO-LEFT ISOLATE	⁧
FSI	U+2068	FIRST STRONG ISOLATE	⁨
PDI	U+2069	POP DIRECTIONAL ISOLATE	⁩
LRM	U+200E	LEFT-TO-RIGHT MARK	‎
RLM	U+200F	RIGHT-TO-LEFT MARK	‏

2) The following HTML tags are supported:

- `<bdo>` (NB the "dir" attribute is mandatory on `<bdo>`)
- `<bdi>` (HTML5)

3) The CSS property "unicode-bidi" is supported with the following (CSS3) values: `normal | embed | isolate | bidi-override | isolate-override | plaintext`.

See <http://www.w3.org/TR/css3-writing-modes/#unicode-bidi> for more details.

"unicode-bidi" is supported on block level elements as well as in-line elements, but note that:

- the value is not inherited to child blocks
- using "embed" or "isolate" has no effect on block level boxes
- "isolate-override" is equivalent to "bidi-override" on block level boxes

NB dir="auto" is not supported generally, but it is supported for <bdi> (has the same effect as if omitted) to use First Strong Isolate (FSI).

Directionality can now be set on individual table cells <td style="direction:rtl;unicode-bidi:embed;"> or <td dir="rtl">

Equivalent methods

The following are equivalent methods:

EMBED

```
<span dir="rtl">...</span>
&#x202B;...&#x202C;
<span style="direction: rtl; unicode-bidi: embed">...</span>
```

OVERRIDE

```
<bdo dir="rtl">...</bdo>
&#x202E;...&#x202C;
<span dir="rtl" style="unicode-bidi: bidi-override">...</span>
<span style="direction: rtl; unicode-bidi: bidi-override">...</span>
```

ISOLATE

```
<bdi dir="ltr">...</bdi>
&#x2067;...&#x2069;
<span dir="rtl" style="unicode-bidi: isolate">...</span>
<span style="direction: rtl; unicode-bidi: isolate">...</span>
```

First Strong Isolate (FSI)

```
<bdi>...</bdi>
<bdi dir="auto">...</bdi>
&#x2068;...&#x2069;
<span dir="rtl" style="unicode-bidi: plaintext">...</span>
<span style="direction: rtl; unicode-bidi: plaintext">...</span>
```

First strong isolate (FSI)

FSI is useful when including text within a paragraph where the directionality of the text is unknown. For example, if you are printing out a catalogue from a database of book titles and the number of readers, when some book titles are in right-to-left script, you may use this template:

```
<li>Title: {TITLE} - {READERS} readers</li>
```

This would result in the following:

- Title: Alice in Wonderland - 12390 readers
- Title: 17890 - עליסה בארץ הפלאות, ספרו-ילדים מאת לואיס קרול readers

```
<li>Title: <bdi>{TITLE}</bdi> - {READERS} readers</li>
```

Using BDI will result in the following:

- Title: Alice in Wonderland - 12390 readers
- Title: 1790 - עליסה בארץ הפלאות, ספרו-ילדים מאת לואיס קרול readers

See Also

- [lang](#) - Information on mPDF support for the HTML attribute lang

CJK Languages

In order to view a file with non-embedded CJK (chinese-japanese-korean) fonts, you - and other users - need to download the Asian font pack for Adobe Reader: Chinese (Simplified), Chinese (Traditional), Korean, and Japanese from:

<http://www.adobe.com/support/downloads/product.jsp?platform=windows&product=10>

Indic fonts v5.x

Note: From mPDF >= 6.0 this has been superseded by the use of OpenType layout (OTL).

Special fonts are provided for several Indic languages.

	Unicode range	Font name	Languages
Bengali	0980-09FF	ind_bn_1_001	
Devanagari	0900-097F	ind_hi_1_001	Hindi, Sindhi, Nepali
Malayalam	0D00-0D7F	ind_ml_1_001	
Gujarati	0A80-0AFF	ind_gu_1_001	
Kannada	0C80-0CFF	ind_kn_1_001	
Oriya	0B00-0B7F	ind_or_1_001	
Punjabi	0A00-0A7F	ind_pa_1_001	(Gurmukhi)
Tamil	0B80-0BFF	ind_ta_1_001	
Telugu	0C00-0C7F	ind_te_1_001	

These fonts can be accessed using their font name e.g.

```
<p style="font-family: ind_ml_1_001;">ഞ്ചു</p>
```

NB Indic font support is font-specific; adding other Indic fonts to mPDF won't work.

Default Font

Default font & font-size

A default font and font-size are required for mPDF to function. These are determined by:

Declared when instantiating the mPDF class (see Getting Started). These "provisional" default values will be active if you are using functions to write directly to the PDF file e.g.

```
$mpdf=new mPDF('','A4',9,'dejavusans');  
$mpdf->WriteCell(110,5,'Hello World');
```

If you are using WriteHTML() these "provisional" default values are overridden, and defaults are set as follows:

- If mPDF('c') is set, the default font is set as 'helvetica', 'times' or 'courier'
- Otherwise the default font is set by the value in the default stylesheet \$default_CSS in config.php
- NB The default font-size is also set by the value in the default stylesheet \$default_CSS

If the secondary default CSS stylesheet mpdf.css contains values to define the BODY tag, these values will override the above e.g.

```
body { font-family: serif; font-size: 10pt; }
```

If a CSS stylesheet is used with WriteHTML() and contains values to define the BODY tag, these values will override the above.

In-line style used in the <body> tag will override the default values e.g.

```
<body style="font-family: serif; font-size: 10pt;">
```

Font substitution 5.x

(mPDF >= 5.0)

If a font or default font is declared in mPDF (including the default stylesheet, and any CSS stylesheets etc.) the font that is actually set is determined by:

Standard use

If called in HTML/CSS the font-family name is converted to an mPDF font-family name (see [Font names in 5.x](#))
e.g.

```
<p style="font-family: 'Trebuchet MS'; font-weight: bold;">
```

The font requested (including style) is checked to see if it is available to mPDF: set by the array
`$available_unifonts*`:

1. If the `font[style]` exists - selected e.g. `trebuchetms['B']`
2. If the `font[nostyle]` exists - selected e.g. `trebuchetms['R']`
3. Looks up the font-family in the three arrays `sans_fonts`, `serif_fonts`, and `mono_fonts` defined in `mpdf_config.php`, in this case looking for `'trebuchetms'`. If found, substitutes a font of similar type (sans-serif, serif, or mono) - the first font in the `sans_fonts`, `serif_fonts`, or `mono_fonts` arrays is used.
4. If no font has yet been selected, the first font in the array `$available_unifonts*` is selected

* The array `$available_unifonts` is initially derived from `$this->fontdata` in the `config_fonts.php` file. (`trebuchetms['B']` will be converted to `'trebuchetmsB'`) The array can be altered when certain languages are defined e.g. when using lang markup in the HTML code:

```
<p lang="ar">
```

If 'ar' (arabic) is set as the language, the default settings in `config_cp.php` define a restricted set of fonts which can be selected. In the example case, mPDF will look for `trebuchetmsB` and if not available, `trebuchetms`

Core fonts

If core fonts only are specified by using `$mpdf = new mPDF('c')`, then all font requests will be substituted by Arial/Helvetica, Times or Courier i.e. the core PDF fonts. mPDF determines whether the requested font is a sans-serif, serif or monospace font (as above), and substitutes accordingly.

Character (font) substitution

If some circumstances, individual characters are replaced by glyphs from another font - see [character \(font\) substitution](#).

Font substitution 6.x

If a font or default font is declared in HTML or CSS, the font that is actually set is determined by:

Standard use

If called in HTML/CSS the font-family name is converted to an mPDF font-family name (see [Font names](#)) e.g.

```
<p style="font-family: 'Trebuchet MS'; font-weight: bold;">
```

The font requested (including style) is checked to see if it is available to mPDF: set by the array `$available_unifonts*`:

1. If the font[style] exists - selected e.g. `trebuchetms['B']`
2. If the font[nostyle] exists - selected e.g. `trebuchetms['R']`
3. Looks up the font-family in the three arrays `sans_fonts`, `serif_fonts`, and `mono_fonts` defined in `mpdf_config.php`, in this case looking for `'trebuchetms'`. If found, substitutes a font of similar type (sans-serif, serif, or mono) - the first font in the `sans_fonts`, `serif_fonts`, or `mono_fonts` arrays is used.
4. If no font has yet been selected, the first font in the array `$available_unifonts*` is selected

* The array `$available_unifonts` is initially derived from `$this->fontdata` in the `config_fonts.php` file. (`trebuchetms['B']` will be converted to `'trebuchetmsB'`)

Core fonts

If core fonts only are specified by using `$mpdf = new mPDF('c')`, then all font requests will be substituted by Arial/Helvetica, Times or Courier i.e. the core PDF fonts. mPDF determines whether the requested font is a sans-serif, serif or monospace font (as above), and substitutes accordingly.

Character (font) substitution

If some circumstances, individual characters are replaced by glyphs from another font - see [character \(font\) substitution](#).

Character substitution

(mPDF >= 5.0)

Note: Prior to mPDF 5.0 there were 2 configurable variables, `$this->useSubstitutions` and `$this->useSubstitutionsMB`. controlling behaviour of core fonts and unicode fonts respectively. From mPDF 5.0, character substitution using core fonts is always ON and cannot be disabled by configurable variables. `$this->useSubstitutionsMB` is deprecated but it is recognised as an alias for `$this->useSubstitutions`. `$this->useSubstitutions` controls behaviour in Unicode font documents.

Core fonts

In documents using core fonts only, only characters included in the [win-1252](#) codepage are available in the Arial/Helvetica, Times or Courier fonts. If the document includes characters which are included in the other core Adobe fonts - [Symbols](#), or [Dingbats](#) - these will be substituted. Because they are displayed using a different font they may appear 'odd'.

Unicode fonts

In Unicode Truetype files, the limitation is whether the font file contains a "glyph" for each character in the document. Character substitution (i.e. substituting a different font solely to display that character) can be enabled by setting the configuration variable in `config.php`:

```
$this->useSubstitutions = true;
```

mPDF will try to find substitutions for any missing characters:

1. if the character is in Unicode Plane 2 (SIP) i.e. Unicode value > U+20000:
 1. looks in the sip-ext font file (see [Fonts in mPDF 5.x](#));
 2. looks in the font defined by `$this->backupSIPFont` in the `config_fonts.php` file
2. looks to see if the character is available in the core fonts: Arial/Helvetica, Times, Courier, Symbols or ZapfDingbats
3. looks in each of the the font(s) set by `$this->backupSubsFont` array in the `config_fonts.php` file

It is not recommended to enable this for regular use, as it will add to the processing time.

CONFIGURATION

Configuration files v5.x

(mPDF >= 4.0)

Three configuration files are included in the root folder of mPDF:

File name	Details	mPDF < 4.0
config.php	Configure most variables which affect mPDF. These values can be set at the beginning of individual scripts, but changes here will affect all of your PDF files.	These values were defined at the start of the <code>mpdf.php</code> file.
config_cp.php	Sets the options used for specified languages e.g. restricted fonts when appropriate.	This was contained in the function <code>GetCodepage()</code> in the <code>htmltoolkit.php</code> file.
config_fonts.php	Details of fonts installed in mPDF.	Previously in <code>mpdf_config.php</code> file.

Configuration files v6.x

(mPDF >= 6.0)

Four configuration files are included in the root folder of mPDF:

File name	Details
config.php	Configure most variables which affect mPDF. These values can be set at the beginning of individual scripts, but changes here will affect all of your PDF files. Also contains basic default CSS stylesheet properties.
config_fonts.php	Details of fonts installed in mPDF.
config_lang2fonts.php	Sets the options used for which fonts to use, when the text is defined by the lang attribute.
config_script2lang.php	Sets the options of which lang to mark up text with, when using automatic font selection based on the text script.

Configuration Methods

Refer to Reference >> mPDF Functions >> Overview

Configuration Variables

Refer to Reference >> mPDF Variables >> Overview

HTML SUPPORT

HTML Tags

HTML tags supported

The following HTML tags/elements are recognised and supported (to some extent) by mPDF. All HTML elements are hard-coded in mPDF to be treated as block or in-line elements (e.g. equivalent to CSS display:block or display:inline). This cannot be changed using CSS.

Tag	mPDF	Display type	CSS	Description
<!--...-->	✓			Defines a comment
<!DOCTYPE>				Defines the document type
<a>	✓	INLINE	✓	Defines a hyperlink
<abbr>				Defines an abbreviation
<acronym>	✓	INLINE	✓	Not supported in HTML5. Defines an acronym
<address>	✓	BLOCK	✓	Defines contact information for the author/owner of a document
<applet>				Not supported in HTML5. Defines an embedded applet
<area>				Defines an area inside an image-map
<article>	✓	BLOCK	✓	Defines an article
<aside>	✓	BLOCK	✓	Defines content aside from the page content
<audio>				Defines sound content
	✓	INLINE	✓	Defines bold text
<base>				Specifies the base URL/target for all relative URLs in a document
<basefont>				Not supported in HTML5. Specifies a default color, size, and font for all text in a document
<bdi>	✓	INLINE		Isolates a part of text that might be formatted in a different direction from other text outside it. mPDF>=6.0
<bdo>	✓	INLINE		Overrides the current text direction
<big>	✓	INLINE	✓	Not supported in HTML5. Defines big text
<blockquote>	✓	BLOCK	✓	Defines a section that is quoted from another source
<body>	✓		✓	Defines the document's body
 	✓		✓	Defines a single line break
<button>				Defines a clickable button
<canvas>				Used to draw graphics, on the fly, via scripting (usually JavaScript)
<caption>	✓	BLOCK/TABLE	✓	Defines a table caption
<center>	✓	BLOCK	✓	Not supported in HTML5. Defines centered text
<cite>	✓	INLINE	✓	Defines the title of a work
<code>	✓	INLINE	✓	Defines a piece of computer code
<col>				Specifies column properties for each column within a <colgroup> element
<colgroup>				Specifies a group of one or more columns in a table for formatting
<command>				Defines a command button that a user can invoke
<datalist>				Specifies a list of pre-defined options for input controls
<dd>	✓	BLOCK	✓	Defines a description of an item in a definition list
	✓	INLINE	✓	Defines text that has been deleted from a document
<details>	✓	BLOCK	✓	Defines additional details that the user can view or hide
<dfn>				Defines a definition term
<dir>				Not supported in HTML5. Defines a directory list

<div>	✓	BLOCK	✓	Defines a section in a document
<dl>	✓	BLOCK	✓	Defines a definition list
<dt>	✓	BLOCK	✓	Defines a term (an item) in a definition list
	✓	INLINE	✓	Defines emphasized text
<embed>				Defines a container for an external (non-HTML) application
<fieldset>	✓	FORMS	✓	Groups related elements in a form
<figcaption>	✓	BLOCK	✓	Defines a caption for a <figure> element
<figure>	✓	BLOCK	✓	Specifies self-contained content
	✓	INLINE	✓	Not supported in HTML5. Defines font, color, and size for text
<footer>	✓	BLOCK	✓	Defines a footer for a document or section
<form>	✓	FORMS	✓	Defines an HTML form for user input
<frame>				Not supported in HTML5. Defines a window (a frame) in a frameset
<frameset>				Not supported in HTML5. Defines a set of frames
<h1> to <h6>	✓	BLOCK	✓	Defines HTML headings
<head>	✓			Defines information about the document
<header>	✓	BLOCK	✓	Defines a header for a document or section
<hgroup>	✓	BLOCK	✓	Groups heading (<h1> to <h6>) elements
<hr>	✓		✓	Defines a thematic change in the content
<html>	✓			Defines the root of an HTML document
<i>	✓	INLINE	✓	Defines a part of text in an alternate voice or mood
<iframe>				Defines an inline frame
	✓	INLINE	✓	Defines an image
<input>	✓	FORMFIELD	✓	Defines an input control
<ins>	✓	INLINE	✓	Defines a text that has been inserted into a document
<kbd>	✓	INLINE	✓	Defines keyboard input
<keygen>				Defines a key-pair generator field (for forms)
<label>				Defines a label for an <input> element
<legend>	✓	FORMFIELD		Defines a caption for a <fieldset>, < figure>, or <details> element
	✓	BLOCK	✓	Defines a list item
<link>	✓			Defines the relationship between a document and an external resource
<main>	✓	BLOCK	✓	Defines a main section
<map>				Defines a client-side image-map
<mark>	✓	INLINE	✓	Defines marked/highlighted text
<menu>				Defines a list/menu of commands
<meta>				Defines metadata about an HTML document
<meter>	✓	INLINE		Defines a scalar measurement within a known range (a gauge)
<nav>	✓	BLOCK	✓	Defines navigation links
<noframes>				Not supported in HTML5. Defines an alternate content for users that do not support frames
<noscript>				Defines an alternate content for users that do not support client-side scripts
<object>				Defines an embedded object
	✓	BLOCK	✓	Defines an ordered list
<optgroup>				Defines a group of related options in a drop-down list
<option>	✓	FORMFIELD		Defines an option in a drop-down list
<output>				Defines the result of a calculation
<p>	✓	BLOCK	✓	Defines a paragraph
<param>				Defines a parameter for an object

<pre>	✓	BLOCK	Defines preformatted text
<progress>	✓		Represents the progress of a task
<q>	✓	INLINE	Defines a short quotation
<rp>			Defines what to show in browsers that do not support ruby annotations
<rt>			Defines an explanation/pronunciation of characters (for East Asian typography)
<ruby>			Defines a ruby annotation (for East Asian typography)
<s>	✓	INLINE	Defines text that is no longer correct
<samp>	✓	INLINE	Defines sample output from a computer program
<script>			Defines a client-side script
<section>	✓	BLOCK	Defines a section in a document
<select>	✓	FORMFIELD	Defines a drop-down list
<small>	✓	INLINE	Defines smaller text
<source>			Defines multiple media resources for media elements (<video> and <audio>)
	✓	INLINE	Defines a section in a document
<strike>	✓	INLINE	Not supported in HTML5. Defines strikethrough text
	✓	INLINE	Defines important text
<style>	✓		Defines style information for a document
<sub>	✓	INLINE	Defines subscripted text
<summary>	✓	BLOCK	Defines a visible heading for a <details> element
<sup>	✓	INLINE	Defines superscripted text
<table>	✓	TABLE	Defines a table
<tbody>	✓	TABLE	Groups the body content in a table
<td>	✓	TABLE	Defines a cell in a table
<textarea>	✓	FORMFIELD	Defines a multiline input control (text area)
<tfoot>	✓	TABLE	Groups the footer content in a table
<th>	✓	TABLE	Defines a header cell in a table
<thead>	✓	TABLE	Groups the header content in a table
<time>	✓	INLINE	Defines a date/time
<title>			Defines a title for the document
<tr>	✓	TABLE	Defines a row in a table
<track>			Defines text tracks for media elements (<video> and <audio>)
<tt>	✓	INLINE	Not supported in HTML5. Defines teletype text
<u>	✓	INLINE	Defines text that should be stylistically different from normal text
	✓	BLOCK	Defines an unordered list
<var>	✓	INLINE	Defines a variable
<video>			Defines a video or movie
<wbr>	✓		Defines a possible line-break (treated as identical to soft-hyphen) mPDF >=5.7

XHTML compatible forms are recognised e.g.

HTML Attributes

HTML tag Attributes (in-line) supported

Example of an HTML attribute: <div align="center">

mPDF supports attribute values in single or double quotes e.g. <div align="center"> or <div align='center'>

Minimised attributes are not supported e.g. <input type="checkbox" disabled />

XHTML specification is recommended for best compliance with mPDF.

Note: The attributes *class*, *id* and *style* are supported on most HTML tags: see [Supported CSS](#) for details.

HTML, BODY and ALL recognised tags	lang*	LANGUAGE-COUNTRY CODE *
	dir	rtl ltr (mPDF >= 6.0)
HTML, BODY	dir	rtl ltr (mPDF >= 5.0)
P, DIV	align	left center right justify
TABLE	border	1 0
	width	LENGTH
	align	left center right char char was added in mPDF 5.7
	char	Used in conjunction with align="char". Default if omitted is period ".." Non-ASCII characters can be defined with HTML entities e.g. · or · As per HTML 4 spec. Added mPDF 5.7
	bgcolor	#rrggbb
	cellSpacing	LENGTH
	cellPadding	LENGTH
	(repeat_header)	1 (removed from mPDF >= 5.4; use <thead>)
	autosize	FLOAT value >= 0 Shrinks a table to fit if width is too small to allow complete words to fit. The value (must be >=1) determines the maximum allowable factor to shrink i.e. autosize="2" will allow the font-size to be reduced to a minimum of 1/2 the original size. A value of 1 prevents automatic resizing of the table. (custom attribute)
	rotate	90 -90
TR	bgcolor	#rrggbb
TD, TH	width	LENGTH
	height	LENGTH (but not %)
	align	left center right
	vAlign	top middle bottom
	bgColor	#rrggbb
	colspan	INTEGER
	rowspan	INTEGER
	nowrap	nowrap
OL, UL	font-size	FONT-SIZE
OL	type	1 A a I i disc circle square
	start	INTEGER (mPDF >= 5.7)
UL	type	disc circle square

IMG	width, height	LENGTH
	max-height, max-width, min-height, min-width	LENGTH
	rotate	90 -90 180 90 = clockwise When width is specified e.g. width="3cm" this is applied to the rotated image i.e. will be width 3cm after rotating
HR	width	nn%
	align	left center right
	color	#rrggbb
FONT	face	FONT-FAMILY
	size	1 2 3 4 5 6 7 -1 +1
	color	#rrggbb
FORM	method	get post (default=post) (mPDF >= 5.3)
	action	URI (mPDF >= 5.3)
TEXTAREA	cols	INTEGER
	rows	INTEGER
	readonly	readonly
	required	required [HTML5] (mPDF >= 5.3)
	spellcheck	true false [HTML5] Default=false (mPDF >= 5.3)
	onChange	JAVASCRIPT In Active Forms; uses "Acrobat" Javascript (mPDF >= 5.3)
TEXTAREA, SELECT, INPUT	disabled	disabled
	title	TEXT
	name	TEXT Field names in Active Forms must only contain letters, numbers, colon(:), underscore(_) or hyphen(-). (This is largely as per HTML spec, but cannot contain period(.) as this is part of PDF spec for name heirarchies)
SELECT	size	INTEGER size = n rows visible. Default=1
	multiple	multiple
	required	required [HTML5] (mPDF >= 5.3)
	spellcheck	true false [HTML5] Default=false (mPDF >= 5.3) Only if also editable and size=1
	editable	editable [HTML5] Default=false (mPDF >= 5.3) Only if size=1
	onChange	JAVASCRIPT In Active Forms; uses "Acrobat" Javascript (mPDF >= 5.3)
OPTION	selected	selected
	value	TEXT
INPUT	type	text password hidden image button submit reset radio checkbox
INPUT (text, password, hidden, image, button, submit, reset)	size	INTEGER size=n=width as number of characters
	value	TEXT

INPUT (text, password)	maxlength readonly required spellcheck onChange	INTEGER readonly required [HTML5] (mPDF >= 5.3) true false [HTML5] Default=false (mPDF >= 5.3) JAVASCRIPT In Active Forms; uses "Acrobat" Javascript (mPDF >= 5.3)
INPUT (image, button, submit, reset)	alt src onClick	TEXT TEXT JAVASCRIPT In Active Forms; uses "Acrobat" Javascript (mPDF >= 5.3)
INPUT (button, submit, reset)	<i>noprint</i>	noprint (mPDF >= 5.3)
INPUT (radio, checkbox)	value	TEXT In Active Forms value(s) for radio buttons and checkboxes are required, and can only contain letters, numbers, colon(:), underscore(_), hyphen(-) or period(.)
DOTTAB	<i>outdent</i>	LENGTH (mPDF >= 5.7)
METER	value, max, min, low, high, optimum	FLOAT [HTML5 spec]
PROGRESS	value, max	FLOAT [HTML5 spec]
METER, PROGRESS	width, height type	Any alphanumeric string. If present, will select custom progress/meter formats - IF they have been defined by the user by editing script classes/meter.php

* *lang* is only recognised when the variable \$useLang is set to **TRUE** (**DEFAULT** is **TRUE**)

Note: Table page-break-inside, autosize values and rotate are only respected for that set on first level table of nested tables

Note: Attributes like nowrap, disabled, multiple, readonly, selected, checked are only supported in their formal/long form i.e. selected="selected"

Note: **LANGUAGE-COUNTRY CODE** accepts codes for *lang* according to the [HTML specification](#). e.g. "en", "en-US", "fr", "zh-HK". The settings for these languages are defined in the config_cp.php configuration file. Other codes for character sets (or codepages) are recognised - see [mPDF\(\)](#).

Changelog

Version	Description
2.0	TD border="1" was supported <= mPDF 1.3, but is not valid HTML and is now ignored
	TABLE align="..." with a rotated table - changed to set the alignment ignoring the rotation i.e. align=right sets the table to the right side of the page (looking as though it is bottom-aligned)
2.3	<i>lang</i> was added.
4.0	Default value of <i>lang</i> changed to TRUE
5.0	<i>dir</i> was added to HTML and BODY

Custom HTML Tags

mPDF uses a number of custom HTML tags (see [HTML Control Tags - Overview](#)).

In addition to those, the custom tags <ttz> <tts> and <tta> are reserved for use internally within the mPDF program to denote the core symbol, dingbat and arial characters.

Invalid HTML?

(mPDF >= 2.2)

If you want to hide mPDF custom tags from a browser, enclose any mPDF code within:

```
<!--mpdf ... anything you want to write ... mpdf-->
```

As these are the standard comment tags recognised by every browser, this will create valid HTML which should output correctly in any circumstances. mPDF will strip away the <!--mpdf tag and any following spaces, and the mpdf--> tag and any preceding spaces, and process all enclosed code.

Note: Almost all mPDF custom tags are self closing e.g. <pagebreak />. If you do not use self-closing tags, the HTML may not show correctly in a browser unless you hide the tags in special comments as above.

Most browsers will ignore (self-closing) tags that are not recognised, but <htmlpageheader> and <htmlpagefooter> are different, as they contain innerHTML e.g.

```
<htmlpageheader name="phname">HTML code in here</htmlpageheader>
```

The enclosed innerHTML will be output by a browser, so you need to hide it; either use the special comment tags as above, or an alternative is to use CSS, e.g. add to your stylesheet

```
htmlpageheader { display : none; }  
htmlpagefooter { display : none; }
```

Yet another way is to use in-line CSS:

```
<htmlpageheader name="phname" style="display: none;">HTML code in here</htmlpageheader>
```

TABLES

Tables

Styles

For a full list of CSS styles supported by mPDF for tables/cells, see [Valid CSS](#). Note the custom attribute "topntail" which I have added to add a horizontal border at the top and bottom of the table, and under the THEAD row if present.

Block-level tags (DIV, P etc) are ignored inside tables, including any CSS styles - inline CSS or stylesheet classes, id etc. To set text characteristics within a table/cell, either define the CSS for the table/cell, or use inline tags e.g.

Both models of border-collapse are supported (CSS border-collapse:collapse or separate), as well as cellSpacing and cellPadding.

Rotate

Tables can be rotated by 90 degrees clockwise or anticlockwise using either:

```
<table rotate="90|-90"> or  
<table style="rotate:90|-90;">  
(90 is clockwise, and -90 is anticlockwise)
```

Table rotation is disabled when columns are used.

Autosize

If a table's minimum width is too wide for the page/div/column, it will automatically reduce the font size in order to fit. The minimum width is calculated by the sum of column widths required to fit the longest single word in the column.

This function is ON by default, set in the config.php file as var \$shrink_tables_to_fit=1.4

The font size will only be reduced to a minimum size to avoid ridiculous results. The value of \$shrink_tables_to_fit is the maximum factor by which the font-size will be reduced.

If it is turned off e.g. \$mpdf->shrink_tables_to_fit=0; it is overridden by a specific declaration for a table e.g.:

```
<table autosize="2.4"> or  
<table style="autosize:2.4;">
```

A value of 1 will prevent any (unnecessary) resizing.

page-break-inside:avoid

If a table has the property page-break-inside:avoid and will not fit on the page, mPDF tries to shrink it to fit - up to a maximum "shrink-factor" set by the variable \$mpdf->shrink_tables_to_fit - default is 1.4 (i.e. about 70% original size). If this still won't fit, it moves it to the next page.

A shrunk table may not be what you want. You can prevent this resizing either by setting the maximum shrink-factor for a particular table e.g. <table autosize="1"> or by setting the variable for the whole document i.e. \$mpdf->shrink_tables_to_fit=1;
(Note that mPDF will always resize tables if it is the only way to fit a row or whole table onto a full page.)

Repeating Table Header row on new page

If a table is split onto more than one page, the first row of the table will be repeated at the top of the new page if either:

```
<table repeat_header="1"> or  
<thead> or <tfoot> is defined
```

Error reporting

NB 2 values are defined in the config.php file which control error reporting for tables:

```
$table_error_report = false; // Die and report error if table is too wide to contain whole words (even  
after autosizing)  
$table_error_report_param = ''; // Parameter which can be passed to show in error report i.e. chapter  
number being processed.
```

Table layout

mPDF will attempt to layout tables using the same algorithm recommended by the HTML specifications (see [Auto-layout algorithm](#)). However, the constraints of fitting content to the page size means that the recommended algorithm has to be altered.

In general, mPDF places more priority on producing a pleasing, efficiently laid out table than it does on respecting defined values e.g. `<table width="300px">` or `<td width="30%">`

If the result is not what you want, consider using some of the following to control the layout:

- change the maximum shrinkage factor allowed for an individual table e.g. `<table autosize="1.6">`
- prevent an individual table from resizing e.g. `<table autosize="1">`
- prevent all tables from resizing: `$mpdf->shrink_tables_to_fit=1;`
- non-breaking spaces
- `<td nowrap="nowrap">`
- `<td style="white-space:nowrap">`
- `<td style="width: 33%">`
- `<td style="width: 5cm">`
- `<table style="page-break-inside:avoid">` to force the table onto one page.
- `<table style="overflow: visible|hidden|wrap">` to control wide tables or tables with specific widths specified

Overriding the Auto- Layout

Other methods to override the default layout algorithm.

`$keep_table_proportions`

If the table width is set greater than the page width allows, mPDF will by default ignore any defined sizes and attempt to auto-layout the table to the page width. This will result in relative column widths etc. being lost.

If you wish to maintain the relative proportions of the table, set `$keep_table_proportions = TRUE`; this forces the table to be resized, but keep relative proportions. NB It also forces respect of cell widths set by %

`$ignore_table_percents`

Table and cell widths set as a percent value are respected when possible by mPDF - as long as the table layout meets the other constraints. This can lead to an ugly or inefficiently laid out table for a printed document.

Setting `$ignore_table_percents = TRUE`; will force mPDF to ignore any table or cell widths set as percent values.

`$ignore_table_width`

Table widths set as absolute length values are respected when possible by mPDF - as long as the table layout meets the other constraints. This can lead to an ugly or inefficiently laid out table for a printed document.

Setting `$ignore_table_width = TRUE`; will force mPDF to ignore any table widths set as absolute length values.

`$tableMinSizePriority`

If there is a conflict between respecting `page-break-inside:avoid` and respecting the maximum value allowed for `autosize`, the configurable variable `tableMinSizePriority` will determine which factor takes priority. (mPDF v>=4.6)

Auto-layout algorithm

Algorithm used for autosizing tables

For non-rotated tables

1. If the minimum width of the table is greater than the page width, the table is resized to fit the minimum width = page width.*
(Minimum width is calculated when each cell accommodates longest word (allowing for nowrap /))
This disregards the value of autosize.
2. If the height of the highest row is greater than a full page height, the table is resized to fit the highest row < page height.
(This is an approximation, as it scales the table size by a factor of less than $\sqrt{\text{maxrow} / \text{pageheight}}$).
This disregards the value of autosize.
3. The table is now printed - allowing it to break across pages - unless `page-break-inside: avoid` is set.
4. If `page-break-inside: avoid` is set:
 - if table height is greater than a full page, the size is reduced to fit the table on a full page, and printed.
(This is an approximation, as it scales the table size by a factor of less than $\sqrt{\text{tableheight} / \text{pageheight}}$).
This disregards the value of autosize.
 - if table height is greater than the remaining space on the page (but less than a full page):
 - if the table size can be reduced to fit in the remaining space, AND respect the autosize value (i.e. maximum scaling factor), it is scaled and printed in the remaining page space;
 - if not, a new page is started and the table is printed using the scaling factor set by 1 or 2.

If there is a conflict between respecting `page-break-inside: avoid` and respecting the maximum value allowed for `autosize`, the configurable variable `tableMinSizePriority` will determine which factor takes priority. (mPDF v>=4.6)

For rotated tables

1. If the whole table fits into the remaining height left on the page, print as it is.
2. If the minimum width of the table is greater than the page height remaining, the table is resized to fit the minimum width = page height remaining, as long as the whole table fits, and the scaling factor is within the value of autosize.
(Minimum width is calculated when each cell accommodates longest word - allowing for nowrap, etc.)
3. Else a new page is started, and the algorithm above followed (obviously containing size width becomes height etc.)

* From mPDF 4.3, this can be overridden using CSS property `overflow`:

```
<table style="overflow: auto | hidden | visible | wrap">
```

`auto` is the default, described above.

Border collapse

In the border-collapse=collapse mode, the following rules determine which border style "wins" in case of a conflict (closely follows CSS 2.1 specifications i.e. width >> CSS dominance (cell>table) >> Top/Left > Bottom/Right):

1. Borders with the 'border-style' of 'hidden' take precedence over all other conflicting borders.
2. Narrow borders are discarded in favor of wider ones.
3. Styles are preferred in this order: 'double', 'solid', 'dashed', 'dotted', 'ridge', 'outset', 'groove', and the lowest: 'inset'.
4. If border styles differ only in color, then a style set on a cell wins over one on a table.
5. When two elements of the same type conflict, then the one further to the left or top wins out.

Note that different browsers differ in the fine detail for complex cases.

PAGING

Page breaks

You can force a page break anywhere in the document either by using HTML code or PHP:

```
$mpdf->AddPage();
<pagebreak />
```

You can define or change all page characteristics when you add the new page:

- orientation
- margins
- numbering (on/off, style or suppress)
- header/footer

Note the special [TOCpagebreak\(\)](#) or [<tocpagebreak>](#) which are like [AddPage\(\)](#) and [<pagebreak>](#) whilst at the same time marking the position for a Table of Contents to be later inserted. CSS [@page](#) can also be used to define page breaks.

Forcing Page Breaks

You can set the CSS value for *page-break-before* to "always | left | right" for any block element (p, div etc). This will force a page-break, but take care that any enclosing (outer) block elements will be 'closed down' and their characteristics lost.

Avoiding Page Breaks

mPDF has limited scope to control when automatic page-breaks occur, and does not have 'widows' or 'orphans' protection.

page-break-inside

You can set the CSS value for *page-break-inside* to "avoid" for any block element. mPDF will try to avoid a page-break within the block, but this only works across a maximum of 2 pages, and is not compatible with table autosize or table rotate

"Keep-with-table"

```
$mpdf->use_kwt = true; // DEFAULT=false
```

If set to true, mPDF will automatically set *page-break-inside=avoid* for any H1-H6 header that *immediately* precedes a table, thus keeping the heading together with the table.

- automatically sets the table to fit on one page (i.e. *table:page-break-inside=avoid*) if it is a rotated table
- ignored when: columns on, *page-break-inside=avoid* for surrounding element, active Forms

Tables

If a table has the property *page-break-inside:avoid* and will not fit on the page, mPDF tries to shrink it to fit - up to a maximum "shrink-factor" set by the variable *\$mpdf->shrink_tables_to_fit* - default is 1.4 (i.e. about 70% original size). If this still won't fit, it moves it to the next page.

A shrunk table may not be what you want. You can prevent this resizing either by setting the maximum shrink-factor for a particular table e.g. *<table autosize="1">* or by setting the variable for the whole document i.e. *\$mpdf->shrink_tables_to_fit=1*;

(Note that mPDF will always resize tables if it is the only way to fit a row or whole table onto a full page.)

Headers and Footers

See the documentation for [Headers and Footers](#)

Types of page break

The handling of borders and padding at page breaks was updated in mPDF 6.0. mPDF has three types of page breaks:

- 1) "slice" - no border and no padding are inserted at a break. The effect is as though the element were rendered with no breaks present, and then sliced by the breaks afterward
- 2) "cloneall" - each page fragment is independently wrapped with the borders and padding of all open elements.
- 3) "clonebycss" - open elements which have the (custom) CSS property "box-decoration-break" set to "clone" are independently wrapped with their border and padding.

The difference between 2) and 3) is illustrated by this example:

```
<style>
div { border: 1px solid black; padding: 1em; }
.level1 { box-decoration-break: slice; }
.level2 { box-decoration-break: clone; }
.level3 { box-decoration-break: clone; }
</style>

<div class="level1">
<div class="level2">
<div class="level3">
<p style="page-break-after:always">...</p>
<p>...</p>
</div>
</div>
</div>
```

At the forced pagebreak which occurs after the P element:

If the page break type is "cloneall" - the three DIV elements will all be closed, by drawing the border and padding for each at the end of the page; the three DIV elements will be re-opened, drawing the borders and padding, at the top of the next page.

If the page break type is "clonebycss" - starting from the innermost element (div.level3) the DIV elements will have a border and padding at the end of the page if "box-decoration-break" is clone. In this case level2 and level 3 will be closed/cloned and level 1 will be sliced; the opposite will occur at the top of the next page.

Control of page breaks

Automatic page breaks (in flow of text)	Always "slice"
<tocpagebreak>	Always "cloneall"
<formfeed>	Always "slice"
If using columns	Always "cloneall"
Page break forced by change of @page selector	Always "cloneall"
<pagebreak>	Always "cloneall" if a change in page size or margins is specified. Otherwise page break type is determined by value of configurable variable: \$this->defaultPagebreakType. Default is "cloneall". Default can be overridden by attribute "page-break-type" e.g. <pagebreak page-break-type="clonebycss" />
Page breaks forced by: page-break-before or page-break-after	Page break type determined by value of configurable variable: \$this->defaultPagebreakType. Default is "cloneall".

Notes on page breaking

"box-decoration-break: slice | clone" was proposed for CSS3 in <http://www.w3.org/TR/2012/CR-css3-background-20120417/#the-box-decoration-break> but it appears that it may be withdrawn. Default is "slice"; it is not inherited.

"page-break-before" is not supported on <table>.

"page-break-before|after" is ignored if set on block elements inside a table.

Background images and gradients are not sliced (always cloned).

\$this->restoreBlockPagebreaks in config.php is now redundant.

Double-sided documents

Odd/Even pages and margins

mPDF can use alternating margins for **ODD** and **EVEN** pages (e.g. for **DOUBLE-SIDED** printing). If **\$mirrorMargins** (**\$useOddEven** in mPDF < 4.0) is set to **TRUE** or 1 before adding the first page, the document will mirror the left and right margin values on **ODD** and **EVEN** pages i.e. they become inner and outer margins. (this is automatically reversed for **RTL** languages).

NB Headers and footers use the **ODD** pages as default if this is not used.

The first page in a document will be **ODD** i.e. page number 1, and will appear as a **RIGHT** page in a **LTR** document (or **LEFT** page in a **RTL** document)

Examples

Example #1

```
<?php

// Define a document with default left-margin=30 and right-margin=10
$mpdf=new mPDF('','','', 0, '', 30, 10);
$mpdf->mirrorMargins = true;
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->AddPage();
// Now the right-margin (inner margin on page 2) = 30
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output();

?>
```

Page size & Orientation

Page sizes supported

When declaring an instance of [mPDF](#) class, you can specify the (default) page size and orientation for the document. The margins and orientation can be redefined throughout the document whenever you add a new page using [AddPage\(\)](#) or [<pagebreak>](#). It can also be set by CSS using [@page](#).

Examples

Example #1

```
<?php

$mpdf=new mPDF('','Legal');
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->AddPage('L'); // Adds a new page in Landscape orientation
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output();

?>
```

Example #2

```
// Define a default Landscape page size/format by name
$mpdf=new mPDF('utf-8', 'A4-L');

// Define a default page size/format by array - page will be 190mm wide x 236mm height
$mpdf=new mPDF('utf-8', array(190,236));

// Define a default page using all default values except "L" for Landscape orientation
$mpdf=new mPDF('','','0','','15',15,16,16,9,9,'L');
```

See Also

- [mPDF\(\)](#) - mPDF class constructor
- [AddPage\(\)](#) - Add a new page

Page numbering

{PAGENO} in a header or footer will be replaced by the document page number. By default this will be the actual page number in decimal numerals.

You can optionally:

- suppress page numbering - {PAGENO} will be replaced by a blank string - but counting will continue
- reset numbering - reset the numbering anywhere in the document*
- define/change the style of the numbering - as for lists it can be roman, alphabetical or decimal

From mPDF >= 3.0 the page numbering can be reset to any positive number. Prior to this, it was only possible to reset it to 1.

From mPDF 6.0 the page number style can include any of the values used for list-style-type

Page numbering changes are made when adding a new page using [AddPage\(\)](#) or [<pagebreak>](#) (or [TOCpagebreak\(\)](#) and [<tocpagebreak>](#)) or using [@page](#).

You could obviously also hide the page numbering by redefining a header or footer that does not include {PAGENO}.

Example #1

```
<?
$mpdf=new mPDF();

// Set a simple Footer including the page number
$mpdf->setFooter('{PAGENO}'');

$mpdf->WriteHTML('Section 1');

$mpdf->WriteHTML('<pagebreak suppress="on" />');
// You could also do this using
// $mpdf->AddPage('', '', '', '', 'on');

$mpdf->WriteHTML('Section 2 - No Footer');
$mpdf->WriteHTML('<pagebreak resetpagenum="1" pagenumstyle="a" suppress="off" />');

$mpdf->WriteHTML('Section 3 - Starting with page a');

$mpdf->Output();
?>
```

Changing page numbering from the start of the document

Note: From mPDF 6.0 the default page number style used from the start of the document can be set using [defaultPageNumStyle](#). The following text is now redundant.

If you want to set page numbering characteristics from the first page onwards, you should explicitly add the first page of the document using [AddPage\(\)](#). Note that this is normally not required, as mPDF creates a new first page automatically if required when first using [WriteHTML\(\)](#).

Example #2 In a complex document you could suppress the page numbering from the start for the cover page(s); then reset the number to 1, and set the style as lowercase Roman (i,ii,iii) for the foreword and introduction; then reset the style to decimal for the rest of the document:

```
<?
$mpdf=new mPDF();

// Double-side document - mirror margins
$mpdf->mirrorMargins = 1;

// Set a simple Footer including the page number
$mpdf->setFooter('{PAGENO}'');
```

```
// Turn off (suppress) page numbering from the start of the document
$mpdf->AddPage('', '', '', '', 'on');

$mpdf->WriteHTML('Your Front Cover Pages');

$mpdf->WriteHTML('<pagebreak type="NEXT-ODD" resetpagenum="1" pagenumstyle="i" suppress="off" />');
// You could also do this using
// $mpdf->AddPage('', 'NEXT-ODD', '1', 'i', 'off');

$mpdf->WriteHTML('Your Foreword and Introduction');
$mpdf->WriteHTML('<pagebreak type="NEXT-ODD" pagenumstyle="1" />');
$mpdf->WriteHTML('Your Book text');

$mpdf->Output();
?>
```

If using the initial [AddPage\(\)](#) causes problems (a blank first page), an alternative way to do this is:

```
$mpdf->PageNumSubstitutions[] = array('from'=>1, 'reset'=> 0, 'type'=>'I', 'suppress'=>'on');
```

See Also

- [defaultPageNumStyle](#) - Specify a default page numbering style from the start of the document
- [pagenumPrefix](#) - Specify text to precede page numbers generated by {PAGENO}
- [pagenumSuffix](#) - Specify text to follow page numbers generated by {PAGENO}
- [nbpgPrefix](#) - Specify text to precede page total generated by {nbpg}
- [nbpgSuffix](#) - Specify text to follow page numbers generated by {nbpg}
- [Replaceable aliases](#) -
- [aliasNbPg](#) - Specify the text to be replaced by the document page total
- [aliasNbPgGp](#) - Specify the text to be replaced by the group page total

Please also see [Table of Contents](#) for this special case.

Note: `startPageNums()` and `$showdefaultpagenos` are now deprecated and will do nothing (used prior to v1.3)

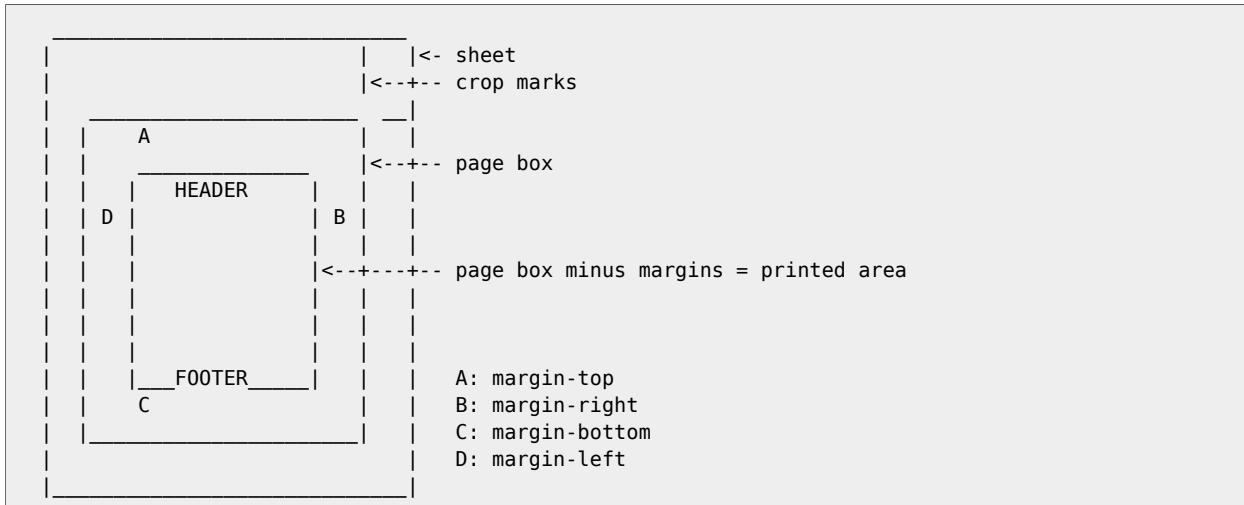
Using @page

(mPDF >= 4.2)

Note: If you are referencing an HTML header/footer, you must add the prefix 'html_' before the name.

About CSS Paged Media

The paged media model of CSS (<http://www.w3.org/TR/1998/REC-CSS2-19980512/page.html>) is used as a basis



The dimensions set when calling a new mPDF() set the Sheet size.

The Page-box size is assumed to be the same as the sheet size by default.

The page-box margins are therefore by default the left/right/top and bottom margins.

NB Page-box margins are INSIDE the page-box (unlike block elements in CSS).

Supported CSS selectors

The CSS @page selector is partially supported in mPDF with the following properties:

```
@page {
    size: 8.5in 11in; <length>{1,2} | auto | portrait | landscape ('em' 'ex' and % are not allowed;
    length values are width height
    margin: 10%; <any of the usual CSS values for margins> (% of page-box width for LR, of height for
    TB)
    margin-header: 5mm; <any of the usual CSS values for margins>
    margin-footer: 5mm; <any of the usual CSS values for margins>
    marks: crop | cross | none
    header: html_myHTMLHeaderOdd;
    footer: html_myHTMLFooterOdd;
    background: ...
    background-image: ...
    background-position ...
    background-repeat ...
    background-color ...
    background-gradient: ...
}
```

Notes

All properties except size are optional.

Three values for the *size* property set the page box to the same size as the sheet:

auto

The page box will be set to the size and orientation of the target sheet.

landscape

Overrides the target's orientation. The page box is the same size as the target, and the longer sides are horizontal.

portrait

Overrides the target's orientation. The page box is the same size as the target, and the shorter sides are horizontal.

The header and footer names refer to named headers/footers set in your document.

NB The prefix "html_" used before the name is used to denote a header/footer defined as HTML code.

If a header/Footer name is set as _blank (or any name that hasn't been defined) it will turn off Headers/Footers.

Crop marks indicate where the page should be cut. Cross marks (also known as register marks or registration marks) are used to align sheets.

If you have defined @page {} in the CSS, then the values for the margins will override the ones set calling a new mPDF().

IMPORTANT - if you define a @page {} but don't specify margins, they will be set to the initial margin values of mPDF.

If you set a page(-box) smaller than the sheet size, the margins are increased by the difference between the page-box and sheet size - automatically centering the page-box inside the sheet.

If you change page-box orientation, the sheet orientation will follow.

Note that block-style elements - and any styling associated with it - will be terminated at a page-break.

Pseudo-selectors

CSS pseudo-selectors :left :right and :first are recognised by mPDF and support the same properties as @page except:

- size
- margin-left
- margin-right
- odd-header-name
- even-header-name
- odd-footer-name
- even-footer-name

Example:

```
@page :right {
    margin-top: 3cm;
    margin-bottom: 4cm;
    header: html_myHeader;
}
```

Pseudo-selectors for page can change top, bottom, header and footer margins, but not left and right margins. mPDF can only cope with one set of (optionally mirrored) left/right margins.

Properties specified in a :first @page rule override those specified in :right (or :left) @page rules for the first page only

Named @page selectors

Named pages are also supported e.g.:

```
@page rotated { size: landscape; }
```

You can then refer to the named page in other CSS style sheets:

```
div.onitsside { page: rotated; page-break-before: right; }
```

<div class="onitsside"> will thus start a new right/odd page which will be in landscape.

Setting a named page

You can also set the page using parameters in:

- functions: AddPage() and TOCpagebreak()
- html tags: <tocpagebreak> <pagebreak> and <formfeed>

page-break-before

The CSS property 'page-break-before' is useful in conjunction with a named page definition.

`page-break-before: always|left|right;`

`always`

Always force a page break before the generated block element.

`left`

Force one or two page breaks before the generated block element so that the next page is formatted as a left/even page.

`right`

Force one or two page breaks before the generated block element so that the next page is formatted as a right/odd page.

So, for example, `page-break-before: right` is equivalent of `AddPage(... 'NEXT-ODD' ...)`

Example using Headers and Footers

```
$mpdf=new mPDF();
$mpdf->useOddEven = 1;
$html = '
<html>
<head>
<style>
@page {
    size: auto;
    odd-header-name: html_myHeader1;
    even-header-name: html_myHeader2;
    odd-footer-name: html_myFooter1;
    even-footer-name: html_myFooter2;
}
@page chapter2 {
    odd-header-name: html_Chapter2HeaderOdd;
    even-header-name: html_Chapter2HeaderEven;
    odd-footer-name: html_Chapter2FooterOdd;
    even-footer-name: html_Chapter2FooterEven;
}
@page noheader {
    odd-header-name: _blank;
    even-header-name: _blank;
    odd-footer-name: _blank;
    even-footer-name: _blank;
}
div.chapter2 {
    page-break-before: right;
    page: chapter2;
}
div.noheader {
    page-break-before: right;
    page: noheader;
}
</style>
</head>
<body>
<htmlpageheader name="myHeader1" style="display:none">
<div style="text-align: right; border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">My document</div>
```

```
</htmlpageheader>
<htmlpageheader name="myHeader2" style="display:none">
<div style="border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">My
document</div>
</htmlpageheader>
<htmlpagefooter name="myFooter1" style="display:none">
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt;
color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">{DATE j-m-Y}</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style:
italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">My document</td>
</tr></table>
</htmlpagefooter>
<htmlpagefooter name="myFooter2" style="display:none">
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt;
color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">My document</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style:
italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">{DATE j-m-Y}</td>
</tr></table>
</htmlpagefooter>

<htmlpageheader name="Chapter2HeaderOdd" style="display:none">
<div style="text-align: right; border-bottom: 1px solid #000000; font-weight: bold; font-size:
10pt;">Chapter 2</div>
</htmlpageheader>
<htmlpageheader name="Chapter2HeaderEven" style="display:none">
<div style="border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">Chapter 2</div>
</htmlpageheader>
<htmlpagefooter name="Chapter2FooterOdd" style="display:none">
<div style="text-align: right; font-weight: bold; font-size: 8pt; font-style: italic;">Chapter 2
Footer</div>
</htmlpagefooter>
<htmlpagefooter name="Chapter2FooterEven" style="display:none">
<div style="font-weight: bold; font-size: 8pt; font-style: italic;">Chapter 2 Footer</div>
</htmlpagefooter>

Hallo World

<div class="chapter2">Text of Chapter 2</div>

<div class="noheader">No-Header page</div>

</body></html>
';

$mpdf->WriteHTML($html);

$mpdf->Output();
```

Using different page sizes in document

(mPDF >= 4.3)

The page (sheet) size can be reset within the document.

Note: The @page model used by CSS and supported by mPDF allows the size of a "page-box" to be set, assuming a constant size of the "sheet" on which it is being printed. To avoid confusion, the term "sheet size" is preferred for the physical dimensions of the page of the document.

There are different ways to change sheet size:

1) Using CSS @page

```
<style>
@page { sheet-size: A3-L; }
@page bigger { sheet-size: 420mm 370mm; }
@page toc { sheet-size: A4; }

h1.bigsection {
    page-break-before: always;
    page: bigger;
}
</style>
```

You can reference the @page selectors using other CSS as above (and see [Using @page](#)) or setting attributes in:

- <tocpagebreak> e.g. <tocpagebreak ... page-selector="bigger" toc-page-selector="toc" ... />
- <pagebreak> e.g. <pagebreak ... page-selector="bigger" ... />

The functions [AddPage\(\)](#) and [TOCpagebreak\(\)](#) both have parameters which allow you to select a @page

2) Specify the sheet size when forcing a new page (HTML tags)

- <pagebreak> e.g. <pagebreak ... sheet-size="A3-L" ... />
- <tocpagebreak> e.g. <tocpagebreak ... sheet-size="420mm 370mm" toc-sheet-size="A4" ... />

Accepted values are:

- "A4", "A3", "Letter" etc
- "A4-L", "A3-L", "Letter-L" etc. for landscape
- a string of 2 dimensions e.g. "21cm 29.7cm" Can be px, pt, in, mm, cm: but not em, ex, or %

3) Specify the sheet size when forcing a new page (functions)

The functions [AddPage\(\)](#) and [TOCpagebreak\(\)](#) both have parameters to select sheet size.

Accepted values - same options as the original [mPDF\(\)](#) - are:

- "A4", "A3", "Letter" etc
- "A4-L", "A3-L", "Letter-L" etc. for landscape
- an array of 2 numbers which are the width then height in mm e.g. array(210,297)

Limitations

- mPDF will not carry over block-style elements across page-breaks.
- Does not work with <formfeed>, and disabled if using \$this->restoreBlockPageBreaks
- Cannot use @page { size: portrait | landscape; } at the same time as defining the sheet-size

HEADERS & FOOTERS

Headers & Footers

Page headers and page footers can be set for mPDF documents in a number of different ways. As mPDF has evolved, new methods have been added.

Note: As from mPDF v6.0 Headers and Footers are all now written internally as HTMLheaders/footers. The use of non-HTML headers and footers (methods 1 & 3) is deprecated, but remains supported. Non-HTML headers and footers are converted in mPDF to HTML equivalents. Default non-HTML headers will not clash with HTML headers, but named non-HTML headers WILL clash with (and overwrite) HTML headers of the same (equivalent) name e.g. `html_MyFooter == MyFooter` (non-HTML).

Types

RUNTIME - RUNTIME headers/footers are set as they are required throughout the document.

NAMED - NAMED headers/footers are defined at any time (before they are used) and given a 'name'; they can then be used/re-used by reference to that name at any time. They can be defined by either PHP script or custom HTML tags.

Non-HTML - NON-HTML headers/footers are set by text strings with no HTML mark up to define styles. Style changes (font-size, font-family, color) can be defined by changing mPDF variables.

HTML - HTML headers/footers are written in standard HTML code. They can only be defined outside HTML block tags (except `<body>`).

Note: **HTML** headers/footers are more intensive of processing time and memory. This probably makes little difference for an average/short document, but may be important for long/complex documents.

Methods

	NON-HTML	HTML
RUNTIME	<p>Method 1 <i>This is the simplest & quickest way to define a header/footer for the whole document if you need limited control over styling.</i></p> <p><code>SetHeader()</code> <code>SetFooter()</code></p> <p>There are several variants of this method, using string or array. The simplest form does not allow different header/footer for ODD and EVEN pages.</p> <p>Style can be defined by variables: <code>\$defaultheaderfontsize</code> <code>\$defaultheaderfontstyle</code> <code>\$defaultheaderline</code> <code>\$defaultfooterfontsize</code> <code>\$defaultfooterfontstyle</code> <code>\$defaultfooterline</code></p>	<p>Method 2 <i>The simplest & quickest way to program a header/footer once for the whole document that includes images or uses more complex layout styles.</i></p> <p><code>SetHTMLHeader()</code> <code>SetHTMLFooter()</code></p>

NAMED	Define	Method 3 <i>This method is useful if you do not need the flexibility of an HTML header/footer, but are changing headers/footers throughout the document.</i> DefHeaderByName() DefFooterByName() <pageheader> <pagefooter>	Method 4 <i>This is the best way for complex headers/footers with the advantage of HTML code, but you can easily change the headers/footers at any time during the document.</i> DefHTMLHeaderByName() DefHTMLFooterByName() <htmlpageheader> <htmlpagefooter>
Reference		<i>These methods are recommended when setting the header/footer at the start of a document.</i> SetHeaderByName() SetFooterByName() <setpageheader> <setpagefooter> @page	<i>These methods are recommended when setting the header/footer at the start of a document.</i> SetHTMLHeaderByName() SetHTMLFooterByName() <sethtmlpageheader> <sethtmlpagefooter> @page

Recommended when you wish to change the headers/footers during the document.
 These methods can reference any **NAMED** header or footer (**NON-HTML** or **HTML**)

AddPage() TOCpagebreak() <pagebreak> <tocpagebreak> @page

In any of the options, {PAGENO} or {DATE j-m-Y} can be used - which will be replaced by the page number or current date. j-m-Y can be replaced by any of the valid formats used in the php `date()` function.

See also: [forcePortraitHeaders](#)

Headers & Footers - Method 1

This uses **RUNTIME NON-HTML** headers & footers. This is the simplest & quickest way to define a header/footer for the whole document if you need limited control over styling. There are several variants of this method, using string or array. The simplest form does not allow different header/footer for **ODD** and **EVEN** pages.

Setting a Header/Footer at the start of a document

Variant #1 (Simplest form)

Use a single command with a string as parameter, to set a header/footer at the right margin of the page on **ODD** pages, and left margin for **EVEN** pages (if **DOUBLE-SIDED** document), or right margin for all pages.

```
$mpdf = new mPDF();
$mpdf->SetHeader('Document Title');
$mpdf->WriteHTML('Document text');
$mpdf->Output();
```

Variant #2 (Split string)

Set a header/footer in three parts. The text string defines three strings divided by '|' which will set a header at the left/centre/right margin of the page on **ODD** pages and right/centre/left margin for **EVEN** pages (if **DOUBLE-SIDED** document), or left/centre/right margin for all pages. Note the use of {PAGENO} which can be used in any type of header or footer.

```
$mpdf = new mPDF();
$mpdf->SetHeader('Document Title|Center Text|{PAGENO}');
$mpdf->SetFooter('Document Title');
$mpdf->WriteHTML('Document text');
$mpdf->Output();
```

Variant #3 (Controlling style with variables)

This is the same as Variant #2, but you can control some aspects of style for the headers/footers by altering certain mPDF variables.

```
$mpdf = new mPDF();
$mpdf->SetHeader('Document Title|Center Text|{PAGENO}');
$mpdf->SetFooter('Document Title');

$mpdf->defaultheaderfontsize=10;
$mpdf->defaultheaderfontstyle='B';
$mpdf->defaultheaderline=0;
$mpdf->defaultfooterfontsize=10;
$mpdf->defaultfooterfontstyle='BI';
$mpdf->defaultfooterline=0;

$mpdf->WriteHTML('Document text');
$mpdf->Output();
```

Variant #4 (Array) - DEPRACATED

Set a header/footer using an array of values. This allows greater control over styling. Recommended to use Variant #5, which is very similar, but specifies **ODD** and **EVEN** forms separately.

```
$arr = array (
    'odd' => array (
        'L' => array (
            'content' => '',
            'font-size' => 10,
            'font-style' => 'B',
            'font-family' => 'serif',
            'color'=>'#000000'
        ),
        'C' => array (
            'content' => '',
            'font-size' => 10,
            'font-style' => 'B',
            'font-family' => 'serif',
            'color'=>'#000000'
        )
    )
);
```

```

        'color'=>'#000000'
),
'R' => array (
    'content' => 'My document',
    'font-size' => 10,
    'font-style' => 'B',
    'font-family' => 'serif',
    'color'=>'#000000'
),
'line' => 1,
),
'even' => array ()
);
$mpdf->SetHeader($arr);

```

Variant #5 (Array)

Set a header/footer using an array of values. This allows greater control over styling. **ODD** and **EVEN** headers/footers are set separately using the second parameter of [SetHeader\(\)](#).

```

$arr = array (
'L' => array (
    'content' => 'Chapter 1',
    'font-size' => 10,
    'font-style' => 'B',
    'font-family' => 'serif',
    'color'=>'#000000'
),
'C' => array (
    'content' => '',
    'font-size' => 10,
    'font-style' => 'B',
    'font-family' => 'serif',
    'color'=>'#000000'
),
'R' => array (
    'content' => 'My document',
    'font-size' => 10,
    'font-style' => 'B',
    'font-family' => 'serif',
    'color'=>'#000000'
),
'line' => 1,
);
$mpdf->SetHeader($arr, '0'); // E for Even header

```

Note: When you are using the array form, any values that are not defined in the array use the document default values, not the defaultheader values (like the previous Simple form) i.e. an undefined font-size uses the document default of 10pt, not the \$defaultheaderfontsize of 8pt.

Although this looks complex, you could change one value easily throughout a document:

```

// following from above...
$mpdf->AddPage();
$arr['L']['content'] = 'Chapter 2';
$mpdf->SetHeader($arr, '0');

```

Changing Header/Footer during the document

This is where **RUNTIME** headers/footers get much more clumsy to use, whichever of the Variants above you are using. When a new page is added to the document (e.g. using [AddPage\(\)](#) or [<pagebreak>](#)) mPDF does the following:

- writes the footer for the current page
- starts the new page
- writes the header for the new page

Therefore to use any **RUNTIME** method you need to:

- change the header before the page-break
- change the footer after the page-break

```
$mpdf = new mPDF();
$mpdf->SetHeader('First section header');
$mpdf->SetFooter('First section footer');
$mpdf->WriteHTML('First section text...');

// Set the new Header before you AddPage
$mpdf->SetHeader('Second section header');
$mpdf->AddPage();
// Set the new Footer after you AddPage
$mpdf->SetFooter('Second section footer');

$mpdf->WriteHTML('Second section text...');

$mpdf->Output();
```

It gets even more complicated if you are using **DOUBLE-SIDED** document and you want to start the new section of your book on an **ODD** page:

```
$mpdf = new mPDF();
$mpdf->useOddEven = true;

$mpdf->SetHeader('First section header');
$mpdf->SetFooter('First section footer');
$mpdf->WriteHTML('First section text...');

// Use a conditional page-break to ensure you are on an EVEN page before changing the Header
$mpdf->AddPage('', 'E');

// Now you know that this page-break takes you to an ODD page
$mpdf->SetHeader('Second section header');
$mpdf->AddPage();
$mpdf->SetFooter('Second section footer');

$mpdf->WriteHTML('Second section text...');

$mpdf->Output();
```

Table of Contents

Using **RUNTIME** headers/footers with a Table of Contents gets so clumsy, it is recommended that you consider one of the **NAMED** methods. Here for the record is how you would do it:

```
$mpdf = new mPDF();
$mpdf->useOddEven = true;

// Set header and footer arrays for section:Introduction
$intro_header_odd_array = array(...);
$intro_header_even_array = array(...);
$intro_footer_odd_array = array(...);
$intro_footer_even_array = array(...);

// Set header and footer arrays for section:Main
$main_header_odd_array = array(...);
$main_header_even_array = array(...);
$main_footer_odd_array = array(...);
$main_footer_even_array = array(...);

// Set the headers/footers for the Introduction
$mpdf->setHeader($intro_header_odd_array, '0');
$mpdf->setHeader($intro_header_even_array, 'E');
$mpdf->setFooter($intro_footer_odd_array, '0');
$mpdf->setFooter($intro_footer_even_array, 'E');

$mpdf->AddPage(' ', ' ', 1, ' ', 'on'); // suppress page numbering for the introduction

$mpdf->WriteHTML('Introduction of document...');
```

```
$mpdf->AddPage('', 'E');

$mpdf->setHeader($main_header_odd_array, '0');
$mpdf->setHeader($main_header_even_array, 'E');

$mpdf->TOCpagebreak('', '', '', '', '', '', '', '', '', '', '', '', '', '',
    $toc-preHTML, $toc-postHTML, $toc-bookmarkText, 1, 'A', 'off');      // sets numbering to start at
A

$mpdf->setFooter($main_footer_odd_array, '0');
$mpdf->setFooter($main_footer_even_array, 'E');

$mpdf->WriteHTML('Main part of document...');

$mpdf->Output();
```

... and for historical purposes using depracated TOC functions:

```

$mpdf = new mPDF();
$mpdf->useOddEven = true;

// Set header and footer arrays for section:Introduction
$intro_header_odd_array = array(...);
$intro_header_even_array = array(...);
$intro_footer_odd_array = array(...);
$intro_footer_even_array = array(...);

// Set header and footer arrays for section:Main
$main_header_odd_array = array(...);
$main_header_even_array = array(...);
$main_footer_odd_array = array(...);
$main_footer_even_array = array(...);

// Set the headers/footers for the Introduction
$mpdf->setHeader($intro_header_odd_array,'0');
$mpdf->setHeader($intro_header_even_array,'E');
$mpdf->setFooter($intro_footer_odd_array,'0');
$mpdf->setFooter($intro_footer_even_array,'E');

$mpdf->AddPage('',' ', 1, '', 'on'); // suppress page numbering for the introduction

$mpdf->WriteHTML('Introduction of document...');

// Set some variables for the ToC - these are all now deprecacted
$mpdf->TOCheader = array();
$mpdf->TOCfooter = array();
$mpdf->TOCpreHTML = '<h2>Table of Contents</h2>';
$mpdf->TOCpostHTML = '<p>Text to come after the contenst list</p>';
$mpdf->TOCbookmarkText = 'Contents';

$mpdf->AddPage('','E');

$mpdf->setHeader($main_header_odd_array,'0');
$mpdf->setHeader($main_header_even_array,'E');
$mpdf->AddPage('',' ', 1, 'i', 'off'); // sets page numbering to start here at 1
$mpdf->setFooter($main_footer_odd_array,'0');
$mpdf->setFooter($main_footer_even_array,'E');

$mpdf->TOC(($tocfont, $tocfontsize, $tocindent, $resetpagenum, $pagenumstyle, $suppress,
$toc_orientation, $TOCusePaging, $TOCuseLinking));

$mpdf->WriteHTML('Main part of document...');

$mpdf->Output();

```

See Also

- SetHeader()
 - SetFooter()
 - \$defaultheaderfontsize
 - \$defaultheaderfontstyle

- `$defaultheaderline`
- `$defaultfooterfontsize`
- `$defaultfooterfontstyle`
- `$defaultfooterline`

Headers & Footers - Method 2

This uses **RUNTIME HTML** headers & footers. This is the simplest & quickest way to program a header/footer once for the whole document that includes images or uses more complex layout styles.

Setting Headers/Footers for the whole document

Use [SetHTMLHeader\(\)](#) and/or [SetHTMLFooter\(\)](#) to set HTML headers/footers before writing to the document.

Example #1 - Single-sided document

```
$mpdf = new mPDF();

// Define the Header/Footer before writing anything so they appear on the first page
$mpdf->SetHTMLHeader('<div style="text-align: right; font-weight: bold;">My document</div>');

$mpdf->SetHTMLFooter(
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt; color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">{DATE j-m-Y}</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style: italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">My document</td>
</tr></table>
');

$mpdf->WriteHTML('Hallo World');

$mpdf->Output();
```

Example #2 - Double-sided document

```
$mpdf = new mPDF();

$mpdf->useOddEven = 1; // Use different Odd/Even headers and footers and mirror margins

// Define the Headers before writing anything so they appear on the first page
$mpdf->SetHTMLHeader('<div style="text-align: right; font-weight: bold;">My document</div>', '0');

$mpdf->SetHTMLHeader('<div style="border-bottom: 1px solid #000000;">My document</div>', 'E');

$mpdf->SetHTMLFooter(
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt; color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">{DATE j-m-Y}</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style: italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">My document</td>
</tr></table>
'); // Note that the second parameter is optional : default = '0' for ODD

$mpdf->SetHTMLFooter(
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt; color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">My document</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style: italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">{DATE j-m-Y}</td>
</tr></table>
', 'E');

$mpdf->WriteHTML('Hallo World');

$mpdf->Output();
```

Changing Header/Footer during the document

This is where **RUNTIME** headers/footers get much more clumsy to use. When a new page is added to the document (e.g. using [AddPage\(\)](#) or [<pagebreak>](#)) mPDF does the following:

- writes the footer for the current page
- starts the new page
- writes the header for the new page

Therefore to use any **RUNTIME** method you need to:

- change the header before the page-break
- change the footer after the page-break

Example #1

```
// First ensure that you are on an Even page
$mpdf->AddPage('', 'E');

// Then set the headers for the next page before you add the page
$mpdf->SetHTMLHeader('<div style="text-align: right; border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">Chapter 2</div>', 'O');
$mpdf->SetHTMLHeader('<div style="border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">Chapter 2</div>', 'E');

$mpdf->AddPage();

$mpdf->SetHTMLFooter('<div style="text-align: right; font-weight: bold; font-size: 8pt; font-style: italic;">Chapter 2</div>', 'O');
$mpdf->SetHTMLFooter('<div style="font-weight: bold; font-size: 8pt; font-style: italic;">Chapter 2</div>', 'E');

$mpdf->WriteHTML('Rest of the document');
$mpdf->Output();
```

Example #2 - Turning a Header/Footer off

```
// If you want the changes to start on an ODD page
$mpdf->AddPage('', 'E');

$mpdf->SetHTMLHeader();
$mpdf->AddPage();
$mpdf->SetHTMLFooter();

$mpdf->WriteHTML('No-Header page');

$mpdf->Output();
```

Table of Contents

Using **RUNTIME** headers/footers with a Table of Contents is very clumsy, it is strongly recommended that you use one of the **NAMED** methods. Here for the record is how you would do it:

```
$mpdf = new mPDF();
$mpdf->useOddEven = true;

// Set the headers/footers for the Introduction
$mpdf->SetHTMLHeader('<div style="text-align: right; font-weight: bold;">Introduction</div>', 'O');
$mpdf->SetHTMLHeader('<div style="border-bottom: 1px solid #000000;">Introduction</div>', 'E');
$mpdf->SetHTMLFooter('<div style="font-weight: bold;">{PAGENO}</div>', 'O');
$mpdf->SetHTMLFooter('<div style="text-align: right;">{PAGENO}</div>', 'E');

$mpdf->AddPage('', '', 1, '', 'on'); // suppress page numbering for the introduction

$mpdf->WriteHTML('Introduction of document...');

$mpdf->AddPage('', 'E');

$mpdf->SetHTMLHeader('<div style="text-align: right; font-weight: bold;">Main</div>', 'O');
$mpdf->SetHTMLHeader('<div style="border-bottom: 1px solid #000000;">Main</div>', 'E');
```

...and for historical reference, using depracted TOC function:

```

$mpdf = new mPDF();
$mpdf->useOddEven = true;

// Set the headers/footers for the Introduction
$mpdf->SetHTMLHeader('<div style="text-align: right; font-weight: bold;">Introduction</div>','0');
$mpdf->SetHTMLHeader('<div style="border-bottom: 1px solid #000000;">Introduction</div>','E');
$mpdf->SetHTMLFooter('<div style="font-weight: bold;">{PAGENO}</div>','0');
$mpdf->SetHTMLFooter('<div style="text-align: right;">{PAGENO}</div>','E');

$mpdf->AddPage(' ', ' ', 1, ' ', 'on'); // suppress page numbering for the introduction

$mpdf->WriteHTML('Introduction of document...');

$mpdf->AddPage(' ','E');

$mpdf->SetHTMLHeader('<div style="text-align: right; font-weight: bold;">Main</div>','0');
$mpdf->SetHTMLHeader('<div style="border-bottom: 1px solid #000000;">Main</div>','E');
$mpdf->AddPage(' ', ' ', 1, 'i', 'off'); // sets page numbering to start here at 1
$mpdf->SetHTMLFooter('<div style="font-weight: bold;">Main - {PAGENO}</div>','0');
$mpdf->SetHTMLFooter('<div style="text-align: right;">Main - {PAGENO}</div>','E');

// Set some variables for the ToC - these are all now depracated
$mpdf->TOCheader = array();
$mpdf->TOCfooter = array();
$mpdf->TOCpreHTML = '<h2>Table of Contents</h2>';
$mpdf->TOCpostHTML = '<p>Text to come after the contenst list</p>';
$mpdf->TOCbookmarkText = 'Contents';

// Mark this current page as where the ToC is to be inserted
$mpdf->TOC($tocfont, $tocfontsize, $tocindent, $resetpagenum, $pagenumstyle, $suppress,
$toc_orientation, $TOCusePaging, $TOCuseLinking);

$mpdf->WriteHTML('Main part of document...');

$mpdf->Output();

```

See Also

- SetHTMLHeader()
 - SetHTMLFooter()

Headers & Footers - Method 3

This uses **NAMED NON-HTML** headers & footers. This method is useful if you do not need the flexibility of an HTML header/footer, but are changing headers/footers throughout the document.

These use the same array values as [Method 1](#) Variant #5.

Note that named headers are not specified as **ODD** or **EVEN** when they are defined, but only when they are selected.

Note: Do not name a header or footer starting with `html_` This prefix is reserved to identify an **HTML** header/footer.

Defining NAMED Headers/Footers

Example #1 - Using PHP

```
$arr1 = array (
    'L' => array (
        'content' => 'Chapter 1',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'C' => array (
        'content' => '',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'R' => array (
        'content' => 'My document',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'line' => 1,
);

$mpdf->DefHeaderByName('MyHeader1', $arr1);
```

Example #2 - Using Custom HTML tags

```
$mpdf=new mPDF();

// Define the Headers and Footers with names
$html =
<pageheader name="MyHeader1" content-left="" content-center="" content-right="My document" header-style="font-family: serif; font-size: 10pt; font-weight: bold; color: #000000;" line="on" />

<pagefooter name="MyFooter1" content-left="{DATE j-m-Y}" content-center="{PAGENO}/{nbpg}" content-right="My document" footer-style="font-family: serif; font-size: 8pt; font-weight: bold; font-style: italic; color: #000000;" />

<div>Now starts the document text... </div>
';

$mpdf->WriteHTML($html);
```

Referencing a Header/Footer by NAME

Once you have defined **NAMED** headers/footers for your document, you can reference them using:

- Methods for **NON-HTML** headers/footers only - **Recommended** when setting the first page header/footer at the start of a document (although they can also be used to change headers/footers during the document).
 - `SetHeaderByName()` - see Example #3
 - `SetFooterByName()` - see Example #3
 - `@page` - see Example #5
 - `<setpageheader>` - see Example #4
 - `<setpagefooter>` - see Example #4
- Methods to access any headers/footers (**HTML** or **NON-HTML**) - **Recommended** when changing header/footer during the document.
 - `AddPage()` - see Example #6
 - `<pagebreak>` - see Example #7
 - `@page` - see Example #5
 - `TOCpagebreak()` - see Example #8
 - `<tocpagebreak>` - see Example #9

Setting a named header at the start of a document

When using a **NAMED** header on the first page, remember that mPDF writes the header as the first page is started. This is usually when you first use `WriteHTML()` which automatically triggers an `AddPage()`.

Example #3 - SetHeaderByName()

```
$mpdf = new mPDF();

// Define a header named 'MyHeader1' here (as Example #1)

$mpdf->SetHeaderByName('MyHeader1');

$mpdf->WriteHTML('Document text');
$mpdf->Output();
```

In this example using custom HTML tags to set the **NON-HTML** header, notice that `<setpageheader>` has `show-this-page = 1`. This is because as soon as you call `WriteHTML()`, mPDF has added the first page, so this fixes the problem by forcing the header to show on the first page:

Example #4 - <setpageheader>

```
$mpdf = new mPDF();

$html = '
<pageheader name="MyHeader1" content-left="" content-center="" content-right="My document" header-
style="font-family: serif; font-size: 10pt; font-weight: bold; color: #000000;" line="on" />

<pagefooter name="MyFooter1" content-left="{DATE j-m-Y}" content-center="{PAGENO}/{nbpg}" content-
right="My document" footer-style="font-family: serif; font-size: 8pt; font-weight: bold; font-style:
italic; color: #000000;" />

<setpageheader name="MyHeader1" value="on" show-this-page="1" />
<setpagefooter name="MyFooter1" value="on" />

<div>Start of the document ... and all the rest</div>
';

$mpdf->WriteHTML($html);
$mpdf->Output();
```

Example #5 - @page

```
$mpdf=new mPDF();
```

```

$html = '
<html>
<head>
<style>
@page {
    size: auto;
    odd-header-name: MyHeader1;
    odd-footer-name: MyFooter1;
}
@page chapter2 {
    odd-header-name: MyHeader2;
    odd-footer-name: MyFooter2;
}
@page noheader {
    odd-header-name: _blank;
    odd-footer-name: _blank;
}
div.chapter2 {
    page-break-before: always;
    page: chapter2;
}
div.noheader {
    page-break-before: always;
    page: noheader;
}
</style>
</head>
<body>
<pageheader name="MyHeader1" content-right="My document" header-style="font-weight: bold; color: #000000;" line="on" />

<pagefooter name="MyFooter1" content-left="{DATE j-m-Y}" content-center="{PAGENO}/{nbpg}" footer-style="font-size: 8pt;" />

<pageheader name="MyHeader2" content-right="Chapter 2" header-style="font-weight: bold; color: #000000;" line="on" />

<pagefooter name="MyFooter2" content-left="{DATE j-m-Y}" content-center="2: {PAGENO}" footer-style="font-size: 8pt;" />

<div>Here is the text of the first chapter</div>
<div class="chapter2">Text of Chapter 2</div>
<div class="noheader">No-Header page</div>
</body></html>
';

$mpdf->WriteHTML($html);

$mpdf->Output();

```

Selecting a named header during the document

Example #6 - AddPage()

```

$mpdf->WriteHTML('Document text');

// In a SINGLE-SIDED document, the 'ODD' values set the default for all pages.
$mpdf->AddPage('','','','','','','','','','','','MyHeader2', '', 'MyFooter2', '', 1, 0, 1, 0);

$mpdf->WriteHTML('Document text');

// Turn Headers and Footers off
$mpdf->AddPage('','','','','','','','','','','',' ',' ',' ',' ',' ',' -1, 0, -1, 0);

$mpdf->WriteHTML('Document text with No Headers/Footers');

```

Example #7 - <pagebreak>

```
$html = '
<p>Document text</p>

<pagebreak odd-header-name="MyHeader2" odd-header-value="on" odd-footer-name="MyFooter2" odd-footer-value="on" />

<p>Text of Chapter 2/p>

<!-- TO TURN HEADER/FOOTER OFF FOR A NEW PAGE -->
<pagebreak odd-header-value="off" odd-footer-value="off" />

<p>No-Header page/p>
';

$mpdf->WriteHTML($html);
```

Table of Contents

Example #8 - TOCpagebreak()

```
$mpdf = new mPDF();

// Define headers here named 'MyHeader1', 'MyTOCHeader', 'MyTOCFooter', 'MyHeader2', 'MyFooter2' (as Example #1)

$mpdf->SetHeaderByName('MyHeader1');

$mpdf->WriteHTML('Introduction of document...');

$mpdf->TOCpagebreak ('', '', '', '', '', '', '', '', '', '', '', '', 'MyTOCHeader', '',
'MyTOCFooter', '', 1, 0 , 1, 0, '', '', '', '', '', '', '', '', '', '', '', '',
'MyHeader2', '', 'MyFooter2', '', 1, 0, 1,0);

$mpdf->WriteHTML('Main part of document...');

$mpdf->Output();
```

Example #9 - <tocpagebreak>

```
$html = '

<!-- Define headers etc. here named 'MyHeader1', 'MyTOCHeader', 'MyTOCFooter', 'MyHeader2',
'MyFooter2' (as Example #2) -->

<p>Introduction: Here starts the document</p>

<tocpagebreak toc-odd-header-name='MyTOCHeader' toc-odd-footer-name='MyTOCFooter' toc-odd-header-
value="1" toc-odd-footer-value="1" odd-header-name='MyHeader2' odd-header-value="1" odd-footer-
name='MyFooter2' odd-footer-value="1" />

<p>Text of Chapter 2... </p>
';

$mpdf->WriteHTML($html);
```

See Also

- [DefHeaderByName\(\)](#)
- [DefFooterByName\(\)](#)
- [<pageheader>](#)
- [<pagefooter>](#)
- [SetHeaderByName\(\)](#)
- [SetFooterByName\(\)](#)
- [<setpageheader>](#)
- [<setpagefooter>](#)
- [AddPage\(\)](#)

- [TOCpagebreak\(\)](#)
- [<pagebreak>](#)
- [<tocpagebreak>](#)
- [@page](#)

Headers & Footers - Method 4

This uses **NAMED HTML** headers & footers. This is the best way for complex headers/footers with the advantage of HTML code, but you can easily change the headers/footers at any time during the document.

Note that **NAMED HTML** headers are not specified as **ODD** or **EVEN** when they are defined, but only when they are selected.

Note: Do not name any header or footer starting with `html_`. This prefix is reserved to identify an **HTML** header/footer when passing its name in a reference.

Note: `AddPage()`, `TOCpagebreak()`, `<pagebreak>` `<tocpagebreak>` and `@page` can reference both **HTML** and **NON-HTML** headers/footers. When referring to an **HTML** header/footer you must add the prefix '`html_`' to distinguish them.

Defining NAMED HTML Headers/Footers

Example #1 - Using PHP

```
$mpdf=new mPDF();

$mpdf->DefHTMLHeaderByName('Chapter2Header','<div style="text-align: right; border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">Chapter 2</div>');

$mpdf->DefHTMLFooterByName('Chapter2Footer','<div style="text-align: right; font-weight: bold; font-size: 8pt; font-style: italic;">Chapter 2 Footer</div>');
```

Example #2 - Using Custom HTML tags

```
$mpdf=new mPDF();

$html = '
<htmlpageheader name="myHeader1">
<div style="text-align: right; border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">My document</div>
</htmlpageheader>

<htmlpageheader name="myHeader2">
<div style="border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">My
document</div>
</htmlpageheader>

<htmlpagefooter name="myFooter1">
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt;
color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">{DATE j-m-Y}</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style:
italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">My document</td>
</tr></table>
</htmlpagefooter>

<htmlpagefooter name="myFooter2">
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt;
color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">My document</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style:
italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">{DATE j-m-Y}</td>
</tr></table>
</htmlpagefooter>

';
```

Referencing a Header/Footer by NAME

Once you have defined **NAMED HTML** headers/footers for your document, you can reference them using:

- Methods for **HTML** headers/footers only - **Recommended** when setting the first page header/footer at the start of a document (although they can also be used to change headers/footers during the document).
 - [SetHTMLHeaderByName\(\)](#) - see Example #3
 - [SetHTMLFooterByName\(\)](#) - see Example #3
 - [@page](#) - see Example #5
 - [<sethtmlpageheader>](#) - see Example #4
 - [<sethtmlpagefooter>](#) - see Example #4
- Methods to access any headers/footers (**HTML** or **NON-HTML**) - **Recommended** when changing header/footer during the document.
 - [AddPage\(\)](#) - see Example #6
 - [<pagebreak>](#) - see Example #7
 - [@page](#) - see Example #5
 - [TOCpagebreak\(\)](#) - see Example #8
 - [<tocpagebreak>](#) - see Example #9

Setting a named HTML header at the start of a document

When using a **NAMED** header on the first page, remember that mPDF writes the header as the first page is started. This is usually when you first use `WriteHTML()` which automatically triggers an `AddPage()`.

Example #3 - SetHTMLHeaderByName()

```
$mpdf = new mPDF();

// Define an HTML header named 'MyHeader1' here (as Example #1)

$mpdf->SetHTMLHeaderByName('MyHeader1');

$mpdf->WriteHTML('Document text');
$mpdf->Output();
```

In this example using custom HTML tags to set the **HTML** header, notice that `<sethtmlpageheader>` has `show-this-page = 1`. This is because as soon as you call `WriteHTML()`, mPDF has added the first page, so this fixes the problem by forcing the header to show on the first page:

Example #4 - <sethtmlpageheader>

```
$mpdf = new mPDF();

$html = '
<htmlpageheader name="MyHeader1">
<div style="text-align: right; border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">My document</div>
</htmlpageheader>

<htmlpagefooter name="MyFooter1">
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt; color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">{DATE j-m-Y}</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style: italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">My document</td>
</tr></table>
</htmlpagefooter>

<sethtmlpageheader name="MyHeader1" value="on" show-this-page="1" />
<sethtmlpagefooter name="MyFooter1" value="on" />

<div>Start of the document ... and all the rest</div>
';

$mpdf->WriteHTML($html);
```

```
$mpdf->Output();
```

Example #5 - @page

```
$mpdf=new mPDF();

$html = '
<html>
<head>
<style>
@page {
    size: auto;
    odd-header-name: html_MyHeader1;
    odd-footer-name: html_MyFooter1;
}
@page chapter2 {
    odd-header-name: html_MyHeader2;
    odd-footer-name: html_MyFooter2;
}
@page noheader {
    odd-header-name: _blank;
    odd-footer-name: _blank;
}
div.chapter2 {
    page-break-before: always;
    page: chapter2;
}
div.noheader {
    page-break-before: always;
    page: noheader;
}
</style>
</head>
<body>
<htmlpageheader name="MyHeader1">
<div style="text-align: right; border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">My document</div>
</htmlpageheader>

<htmlpageheader name="MyHeader2">
<div style="border-bottom: 1px solid #000000; font-weight: bold; font-size: 10pt;">My
document</div>
</htmlpageheader>

<htmlpagefooter name="MyFooter1">
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt;
color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">{DATE j-m-Y}</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style:
italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">My document</td>
</tr></table>
</htmlpagefooter>

<htmlpagefooter name="MyFooter2">
<table width="100%" style="vertical-align: bottom; font-family: serif; font-size: 8pt;
color: #000000; font-weight: bold; font-style: italic;"><tr>
<td width="33%"><span style="font-weight: bold; font-style: italic;">My document</span></td>
<td width="33%" align="center" style="font-weight: bold; font-style:
italic;">{PAGENO}/{nbpg}</td>
<td width="33%" style="text-align: right; ">{DATE j-m-Y}</td>
</tr></table>
</htmlpagefooter>

<div>Here is the text of the first chapter</div>

<div class="chapter2">Text of Chapter 2</div>

<div class="noheader">No-Header page</div>
```

```
</body></html>
';
$mpdf->WriteHTML($html);
$mpdf->Output();
```

Selecting a named header during the document

Example #6 - AddPage()

```
$mpdf->WriteHTML('Document text');

// In a SINGLE-SIDED document, the 'ODD' values set the default for all pages.
$mpdf->AddPage('','','','','','','','','','','','html_MyHeader2','','html_MyFooter2','','1,0,1,0');

$mpdf->WriteHTML('Document text');

// Turn Headers and Footers off
$mpdf->AddPage('','','','','','','','','','','','','','','-1,0,-1,0');

$mpdf->WriteHTML('Document text with No Headers/Footers');
```

Example #7 - <pagebreak>

```
$html = '
<p>Document text</p>

<pagebreak odd-header-name="html_MyHeader2" odd-header-value="on" odd-footer-name="html_MyFooter2"
odd-footer-value="on" />

<p>Text of Chapter 2</p>

<!-- TO TURN HEADER/FOOTER OFF FOR A NEW PAGE -->
<pagebreak odd-header-value="off" odd-footer-value="off" />

<p>No-Header page</p>
';

$mpdf->WriteHTML($html);
```

Table of Contents

Example #8 - T0Cpagebreak()

```
$mpdf = new mPDF();

// Define HTML headers here named 'MyHeader1', 'MyT0CHeader', 'MyT0CFooter', 'MyHeader2',
'MyFooter2' (as Example #1)

$mpdf->SetHTMLHeaderByName('MyHeader1');

$mpdf->WriteHTML('Introduction of document...');

$mpdf->T0Cpagebreak ('','','','','','','','','','','','','','html_MyT0CHeader','','',
'html_MyT0CFooter','','1,0,1,0','','','','','','','','','','','','','html_MyHeader2','','',
'html_MyFooter2','','1,0,1,0');

$mpdf->WriteHTML('Main part of document...');

$mpdf->Output();
```

Example #9 - <tocpagebreak>

```
$html = '
```

```
<!-- Define HTML headers etc here named 'MyHeader1', 'MyTOCHeader', 'MyTOCFooter', 'MyHeader2',
'MyFooter2' (as Example #2) -->

<p>Introduction: Here starts the document</p>

<tocpagebreak toc-odd-header-name='html_MyTOCHeader' toc-odd-footer-name='html_MyTOCFooter' toc-odd-
header-value="1" toc-odd-footer-value="1" odd-header-name='html_MyHeader2' odd-header-value="1" odd-
footer-name='html_MyFooter2' odd-footer-value="1" />

<p>Text of Chapter 2... </p>
';

$mpdf->WriteHTML($html);
```

See Also

- [DefHTMLHeaderByName\(\)](#)
- [DefHTMLFooterByName\(\)](#)
- [<htmlpageheader>](#)
- [<htmlpagefooter>](#)
- [SetHTMLHeaderByName\(\)](#)
- [SetHTMLFooterByName\(\)](#)
- [<sethtmlpageheader>](#)
- [<sethtmlpagefooter>](#)
- [AddPage\(\)](#)
- [TOCpagebreak\(\)](#)
- [<pagebreak>](#)
- [<tocpagebreak>](#)
- [@page](#)

Page numbers & Date

In any page header or footer, {PAGENO} or {DATE j-m-Y} can be used - which will be replaced by the page number or current date. j-m-Y can be replaced by any of the valid formats used in the php [date\(\)](#) function.

The page number is the calculated number, taking regard of any resetting of numbering during the document.

You can also use the GLOBAL replacement aliases {nb} and {nbpq} (see [Replaceable Aliases](#)).

Headers & Top margins

(mPDF >= 4.0)

In standard usage, mPDF sets the following:

- top-margin = distance in mm from top of page to start of text (ignoring any headers)
- header-margin = distance in mm from top of page to start of header
- bottom-margin = distance in mm from bottom of page to bottom of text (ignoring any footers)
- footer-margin = distance in mm from bottom of page to bottom of footer

If you specify a header that extends further down the page than the top-margin, then the header and main text will overlap.

Alternative margin usage

The variables `$this->setAutoTopMargin` and `$this->setAutoBottomMargin` can be set in the config.php file. Both default to **FALSE**.

pad

If `$this->setAutoTopMargin` is set to '**pad**' then the value for margin-top is used to set a fixed distance in mm (padding) between the bottom of the header and top of the main text. The converse is true with `$this->setAutoBottomMargin`.

stretch

If `$this->setAutoTopMargin` is set to '**stretch**' then the margin-top sets a **minimum** distance in mm between the top of the page and the top of the main text, which expands if header is too large to fit. The value `$this->autoMarginPadding` (default = 2, configurable in config.php) defines an additional distance in mm used as padding between the header and main text. This is the behaviour seen in Microsoft Word.

The converse is true with `$this->setAutoBottomMargin`.

Experimental!

Use of this function is probably best considered experimental. It is incompatible with some of the other functionality in mPDF, and has some limitations:

- Using either pad or stretch, the values are set for the whole document - changes during the document will produce unpredictable results
- Incompatible with orientation changes, `forceportraitheaders` or `forceportraitmargins`
- Does not work with "write to current page" options for Headers
- Incompatible with CSS @paged media
- Cannot set a new Footer when already further down the page than there is room needed for footer
- Potential error if using {nbpg} to define page total in an HTML header or footer. The top/bottom margin is calculated by writing the HTML header using the current page no. If when the header is finally written, a page number total of e.g. 500 causes an extra line in the header/footer, this will not be allowed for.

Browser compatible HTML

If you wish to pass the same HTML to a browser as to mPDF, and it is showing incorrectly in the browser, please see [Custom tags](#).

Rotated pages

Special case - rotated Portrait headers on Landscape pages

```
$mpdf->forcePortraitHeaders = true;
```

This is a quick fix which rotates HTML headers and Footers (only - not normal ones) on landscape pages in the following conditions:

- document orientation is portrait
- when adding a page, you must only call \$mpdf->AddPage('L') or <pagebreak orientaion="landscape" />

If you try to set new margins/headers etc. for new landscape pages when forcePortrait=true, it will go wrong

CSS & STYLESHEETS

Stylesheets & CSS - Introduction

Default CSS styles are defined in config.php file (as var \$default_CSS) which is based on the recommended default values for HTML4 - <http://www.w3.org/TR/CSS21/sample.html> The appearance of a default mPDF document based on HTML should approximate to its appearance in a browser.

The following are supported (in order of ascending priority - lower ones in list overwrite higher):

- HTML attributes e.g. <div align="center"> (see [supported HTML attributes](#))
- CSS Stylesheets - included in header of HTML document or as <link /> or as @import()
 - - html tags e.g. p { font-size:12pt; color:#880000; }
 - - class e.g. .classname { font-size:9pt; }
 - - id e.g. #style { font-size:9pt; }
- In-line CSS style e.g. <p style="font-family:monospace;">

Note: Prior to mPDF 5.x HTML attributes overrode CSS styles.

CSS attributes, used in stylesheets or in-line, can define:

- most tags/elements e.g. div, p, body, table, span
- class-names e.g. p.mystylename { font-size:9pt; }
- id e.g. div#style { font-size:9pt; }

Tag, class and id can share a similar name e.g. p {...} .p {...} and #p {...} are handled uniquely

There is some support for 'cascaded' CSS e.g. div.topic table.type1 td {...}

- table, tr, th and td will only be recognised as the last items (as above)

- only 'block' elements (not 'in-line') can be included i.e. div.style1 a {...} will not work, nor will a#class1 {...}

For a full list of CSS attributes supported see [Supported CSS](#)

Note the OUTLINE style (which is not supported in most browsers) does work in mPDF e.g.

Using a stylesheet

The WriteHTML() method takes second parameter i.e. *mode*. See [WriteHTML\(\)](#) for details of this and other parameters.

mode

0 - Use this (default) if the text you pass is a complete HTML page including head and body and style definitions.

1 - Use this when you want to set a CSS stylesheet - see example below

2 - Write HTML code without the <head> information. Does not need to be contained in <body>

Example using a stylesheet

```
$stylesheet = file_get_contents('style.css');
$mpdf->WriteHTML($stylesheet,1);
$mpdf->WriteHTML($html,2);
```

Media selectors

mPDF supports media-dependent CSS styles as:

```
@media print {
  p { color: red; }
}

<style media="print">
  p { color: red; }
```

```
</style>  
  
<link rel="stylesheet" media="print" href="...." />
```

By default mPDF will match stylesheets set for "print" or "all" media. This can be changed by the configurable variable [CSSselectMedia](#).

Supported CSS

Supported CSS attributes - stylesheets or in-line

Unless otherwise stated, the following values are supported:

LENGTH: px, pc, pt, cm, mm, in, em, rem, ex and % (where appropriate) are supported. Default if no unit given is px.

FONT-SIZE: px, pc, pt, em, rem, ex, %, small, medium, large, x-small, x-large are supported. Default if no unit given is px.

Note: Support for "rem" was added mPDF 5.7. Unlike the CSS3 specification, the basic size used for rem in the document is based on the font-size set on the `<body>` element (rather than the `<html>` element).

Conversion from "px" is determined by the configurable variables [dpi](#) and [img_dpi](#)

"ex" uses an approximation of half the font height

FONT-FAMILY: Any font family defined in your configuration, as well as *sans*, *sans-serif*, *serif* or *monospace*.

COLOR: #rgb or #rrggb or rgb(255,255,255) or [colour names](#) e.g. 'black', 'gray' are supported.

Also supported are:

- rgb(255,255,255)
- rgba(255,255,255,1) [last value is transparency (alpha) - between 0-1]*
- rgb(100%,100%,100%)
- hsl(360,100%,100%)
- hsla(360,100%,100%,1) [last value is transparency (alpha) - between 0-1]*
- cmyk(100,100,100,100) [or 0-100%]
- cmyka(100,100,100,100,1) [or 0-100%; last value is transparency (alpha) - between 0-1]*
- device-cmyk(100,100,100,100) [or 0-100%]
- device-cmyka(100,100,100,100,1) [or 0-100%; last value is transparency (alpha) - between 0-1]*
- spot(COLOR NAME, 100%). (cf. [AddSpotColor](#))

mPDF >= 5.7 Spotcolor CMYK values can be defined as it is used e.g. color: spot(PANTONE 534 EC, 100%, 85, 65, 47, 9);

*Alpha values (transparency) are only supported on background colours - not text color

HTML Tag	Property	Values allowed & Notes
BODY	<code>direction</code>	<code>rtl ltr</code> (mPDF ≥ 5.0)
	COMMON TEXT STYLES	See below - all except <code>text-shadow</code>
	<code>margin-collapse</code>	<code>collapse none</code> (custom attribute - collapses top and bottom margins if at top or bottom of page)
	<code>line-height</code>	Line height as a factor of font-height. Usual values around 1.2 or 1.3 Also accepts px, pc, pt, cm, mm, in, em and % (mPDF ≥ 4.0)
	<code>background, background-image, background-position, background-repeat, background-color</code>	background is parsed as per the CSS specification. background-image in the form <code>url(image.png)</code> or <code>url('image.png')</code> or GRADIENT (mPDF ≥ 5.1). background-position is supported as per CSS2.1 background-repeat: repeat repeat-x repeat-y no-repeat background-attachment is parsed but has no effect. See note below. (cf. Backgrounds & borders) (All except background-color added mPDF 3.0)
	<code>background-image-resolution</code>	<code>normal [from-image <dpi>]</code> (mPDF ≥ 5.1) Custom tag, but as for image-resolution in CSS3
	<code>background-gradient</code>	Defines a linear or radial colour gradient for the background. linear <code>COLOR1 COLOR2 x1 y1 x2 y2</code> ; or radial <code>COLOR1 COLOR2 x1 y1 x2 y2 radius</code> ; x, y and radius are values between 0 and 1 (custom attribute) (cf. Backgrounds & borders)
	<code>background-image-opacity</code>	Value between 0 and 1.0 Sets the opacity/transparency of the background image.
	<code>background-image-resize</code>	0 - No resizing (default) 1 - Shrink-to-fit w (keep aspect ratio) 2 - Shrink-to-fit h (keep aspect ratio) 3 - Shrink-to-fit w and/or h (keep aspect ratio) 4 - Resize-to-fit w (keep aspect ratio) 5 - Resize-to-fit h (keep aspect ratio) 6 - Resize-to-fit w and h

HTML Tag	Property	Values allowed & Notes
All Block level tags:	margin*, margin-top, margin-bottom, margin-left, margin-right	LENGTH
P, DIV, H1-H6, OL, UL, ADDRESS, BLOCKQUOTE, CAPTION, CENTER, DL, DT, DD, FORM, ARTICLE, ASIDE, DETAILS, FIELDSET, FIGURE, FIGCAPTION, FOOTER, HEADER, HGROUP, NAV, SECTION, SUMMARY	padding*, padding-right, padding-left, padding-top, padding-bottom	LENGTH
	border, border-top, border-bottom, border-left, border-right	Size style and colour e.g. LENGTH solid dotted dashed double COLOR
	border-color*, border-top-color, border-right-color, border-bottom-color, border-left-color	COLOR (mPDF >= 4.0)
	border-width*, border-top-width, border-right-width, border-bottom-width, border-left-width	LENGTH (mPDF >= 4.0)
	border-style*, border-top-style, border-right-style, border-bottom-style, border-left-style	solid dotted dashed double (mPDF >= 4.0) (double was added mPDF 4.2)
	border-radius, border-top-left-radius, border-top-right-radius, border-bottom-right-radius, border-bottom-left-radius	LENGTH See draft CSS3 specification for use. NB Use of \$autoPadding which can automatically increase padding in block elements with border-radius set as required
	width	LENGTH mPDF does not support nested block elements which overlap (horizontally or vertically) i.e. the inner block must be contained by the outer block's physical dimensions. width and height are overridden if this is not the case.
	height	LENGTH mPDF does not support nested block elements which overlap (horizontally or vertically) i.e. the inner block must be contained by the outer block's physical dimensions. width and height are overridden if this is not the case. (Prior to mPDF v5.1: only supported if whole block is all on one page; will extend a block but not shorten it; will not force a pagebreak even if height should move onto next page.) NB % is interpreted as % of printable page height (inside margins)
	text-align	left center right justify
	page-break-after	auto avoid always left right (extended mPDF >= 5.4)
	page-break-inside	avoid (avoids a page-break within the block - only works across max. of 2 pages; not compatible with table autosize or table rotate)
	page-break-before	always left right
	display	none
	visibility	visible hidden <i>printonly</i> <i>screenonly</i> (mPDF >= 5.4)
	margin-collapse	collapse none (custom attribute - collapses top and bottom margins if at top or bottom of page)
	line-height	Line height as a factor of font-height. Usual values around 1.2 or 1.3 Also accepts px, pc, pt, cm, mm, in, em and % (mPDF >= 4.0) 'normal' also accepted (set by default to 1.3 in mPDF)
	line-stacking-strategy	inline-line-height block-line-height max-height grid-height As per draft CSS3 spec. (mPDF >= 6.0) Default = inline-line-height
	line-stacking-shift	consider-shifts disregard-shifts As per draft CSS3 spec. (mPDF >= 6.0) Default = consider-shifts
	background, background-image, background-position, background-repeat, background-color	background is parsed as per the CSS specification. background-image in the form url(image.png) or url('image.png') or GRADIENT (mPDF >= 5.1). background-position is supported as per CSS2.1 background-repeat: repeat repeat-x repeat-y no-repeat background-attachment is parsed but has no effect. See note below. (cf. Backgrounds & borders) (All except background-color added mPDF 3.0)
	background-clip	padding-box border-box (default=border-box) extended to include "content-box" from mPDF 5.7
	background-origin	padding-box content-box border-box (default padding-box) As per CSS3. mPDF >= 5.7
	background-size	[<length> <percentage> auto]{1,2} cover contain (default = auto) As per CSS3. mPDF >= 5.7
	background-image-resolution	normal [from-image <dpi>] (mPDF >= 5.1) Custom tag, but as for image-resolution in CSS3
	background-gradient	Defines a linear or radial colour gradient for the background. linear COLOR1 COLOR2 x1 y1 x2 y2; or radial COLOR1 COLOR2 x1 y1 x2 y2 radius; x, y and radius are values between 0 and 1 (custom attribute) (cf. Backgrounds & borders)
	background-image-opacity	Value between 0 and 1.0 Sets the opacity/transparency of the background image.
	background-image-resize	0 - No resizing (default) 1 - Shrink-to-fit w (keep aspect ratio) 2 - Shrink-to-fit h (keep aspect ratio) 3 - Shrink-to-fit w and/or h (keep aspect ratio) 4 - Resize-to-fit w (keep aspect ratio) 5 - Resize-to-fit h (keep aspect ratio) 6 - Resize-to-fit w and h
	box-shadow	As per CSS3 specification. 'inset' is not supported. (mPDF >= 5.4)
	box-decoration-break	slice clone (default = slice) Defines the handling of padding and borders at page breaks, when clonebycss set
	z-index	Sets a layer in the PDF document. Note: this is not the same as a "layer" in CSS terms. See Layers .

HTML Tag	Property	Values allowed & Notes
All Block level tags: P, DIV, H1-H6, OL, UL, ADDRESS, BLOCKQUOTE, CAPTION, CENTER, DL, DT, DD, FORM, ARTICLE, ASIDE, DETAILS, FIELDSET, FIGURE, FIGCAPTION, FOOTER, HEADER, HGROUP, NAV, SECTION, SUMMARY (contd.)	COMMON TEXT STYLES <i>text-indent</i>	See below Lengths first line of text in the paragraph/block. LENGTH (Indent is disabled in tables.) Negative value will give a 'hanging indent'.
	position	fixed absolute absolute - uses physical page as containing element; fixed - uses printable page (inside margins) as containing element. (mPDF >= 4.0) NB Only supported for top-level elements i.e. where the parent element is <body>. Fixed-position or floating elements nested inside other fixed-position or floating elements are not supported.
	overflow	visible hidden auto Applies to block elements with position fixed or absolute. auto - causes text to autofit within the block size (constrained if necessary to page edges). (mPDF >= 4.0)
	hyphens	none manual auto (default = manual) As per CSS. (mPDF >= 5.7)
	rotate	90 -90 Applies only to block elements with position fixed or absolute. (mPDF >= 5.0) Sizing and layout of the block element using top, left, bottom, right, width and height are applied BEFORE the element is rotated. The top and left co-ordinates are respected for positioning, except when bottom or right are specified without respective top or left when the bottom/right co-ordinates are used.
	top, left, bottom, right	LENGTH Applies to block elements with position fixed or absolute. (mPDF >= 4.0)
	float	left right Partially supported as for CSS2 property. NB Fixed-position or floating elements nested inside other fixed-position or floating elements are not supported.
	clear	left right both
	page	Specify a named page selector (see @page)
	direction	rtl ltr (mPDF >= 5.1)

HTML Tag	Property	Values allowed & Notes
LIST tags: OL, UL	list-style	['list-style-type' 'list-style-position' 'list-style-image'] mPDF >= 6.0
	list-style-type	1 A a I i disc circle square decimal lower-roman upper-roman lower-latin upper-latin lower-alpha upper-alpha none arabic-indic bengali cambodian cjk-decimal devanagari gujarati gurmukhi hebrew kannada khmer lao malayalam oriya persian telugu thai urdu tamil "1" - decimal "A" = upper-latin "a" = lower-latin "I" = upper-roman "i" = lower-roman Custom list-style-type is recognised e.g.: U+263Argb(255,0,0); - where U+263A is the Unicode HEX value of the character you want for the bullet - the character MUST be included in the font used for that list item - rgb() bit is optional
	list-style-image	<URI> mPDF >= 6.0
	list-style-position	inside outside (default = outside) mPDF >= 6.0
NB Lists are handled as block-level elements as from mPDF 6.0 (see above)		

HTML Tag	Property	Values allowed & Notes
LI	list-style	['list-style-type' 'list-style-position' 'list-style-image'] mPDF >= 6.0
	list-style-type	1 A a I i disc circle square decimal lower-roman upper-roman lower-latin upper-latin lower-alpha upper-alpha none arabic-indic bengali cambodian cjk-decimal devanagari gujarati gurmukhi hebrew kannada khmer lao malayalam oriya persian telugu thai urdu tamil "1" - decimal "A" = upper-latin "a" = lower-latin "I" = upper-roman "i" = lower-roman Custom list-style-type is recognised e.g.: U+263Argb(255,0,0); - where U+263A is the Unicode HEX value of the character you want for the bullet - the character MUST be included in the font used for that list item - rgb() bit is optional
	list-style-image	<URI> mPDF >= 6.0
	list-style-position	inside outside (default = outside) mPDF >= 6.0
NB Lists are handled as block-level elements as from mPDF 6.0 (see above)		

HTML Tag	Property	Values allowed & Notes
All Inline tags: SPAN, A, SUB, SUP, ACRONYM, BIG, SMALL, INS, S, STRIKE, DEL, STRONG, CITE, Q, EM, B, I, U, SAMP, CODE, KBD, TT, VAR, FONT, TIME, MARK	COMMON TEXT STYLES	See below
	border, border-top, border-bottom, border-left, border-right	Size style and colour e.g. LENGTH solid dotted dashed double COLOR (mPDF >= 5.4)
	border-color*, border-top-color, border-right-color, border-bottom-color, border-left-color	COLOR (mPDF >= 5.4)
	border-width*, border-top-width, border-right-width, border-bottom-width, border-left-width	LENGTH (mPDF >= 5.4)
	border-style*, border-top-style, border-right-style, border-bottom-style, border-left-style	solid dotted dashed double (mPDF >= 5.4)
	background-color, background	COLOR (only the color is supported in background)
	display	none (mPDF >= 5.0)
	visibility	hidden visible printonly screenonly (default = visible) (mPDF >= 5.7)
	hyphens	none manual auto (default = manual) As per CSS. (mPDF >= 5.7)

HTML Tag	Property	Values allowed & Notes
TABLE	<code>background-color</code> <code>background, background-image, background-position, background-repeat, background-color</code>	COLOR background is parsed as per the CSS specification. background-image in the form url('image.png') or url('image.png') or GRADIENT . background-position is supported as per CSS2.1 background-repeat: repeat repeat-x repeat-y no-repeat background-attachment is parsed but has no effect. See note below. (cf. Backgrounds & borders) (All except background-color added mPDF 5.1)
	<code>background-image-resolution</code>	normal [from-image <dpi>] (mPDF >= 5.1)
	<code>background-image-opacity</code>	Value between 0 and 1.0 Sets the opacity/transparency of the background image.
	COMMON TEXT STYLES	See below - except text-decoration, text-transform and text-shadow
	<code>border, border-right, border-left, border-top, border-bottom</code>	Size style and colour e.g. LENGTH double solid dashed dotted ridge outset groove inset hidden none COLOR NB Table & cell borders accept: double solid dashed dotted ridge outset groove inset (other block elements only support solid dotted dashed double)
	<code>border-color*, border-top-color, border-right-color, border-bottom-color, border-left-color</code>	COLOR (mPDF >= 4.0)
	<code>border-width*, border-top-width, border-right-width, border-bottom-width, border-left-width</code>	LENGTH (mPDF >= 4.0)
	<code>border-style*, border-top-style, border-right-style, border-bottom-style, border-left-style</code>	double solid dashed dotted ridge outset groove inset hidden none (mPDF >= 4.0)
	<code>padding*, padding-right, padding-left, padding-top, padding-bottom</code>	LENGTH (mPDF <= 1.3 incorrectly set cell-padding for all cells in table.) Sets table padding (only relevant when border-collapse:separate)
	<code>margin*, margin-right, margin-left, margin-top, margin-bottom</code>	LENGTH
	<code>width</code>	LENGTH
	<code>vertical-align</code>	top middle bottom (applies to all cells in table)
	<code>text-align</code>	left right center (applies to all cells in table; to centre a table on the page, you must use the align="center" attribute in the table tag)
	<code>border-collapse</code>	collapse separate
	<code>border-spacing</code>	Single or double values: 2px sets horizontal and vertical; 2px 3px sets horizontal=2 and vertical=3 (Same as cellSpacing attribute) Default=2px
	<code>empty-cells</code>	hide show (only relevant when border-collapse:separate)
	<code>page-break-inside</code>	avoid
	<code>page-break-after</code>	left right always (mPDF >= 5.4)
	<code>rotate</code>	rotates the table 90 degrees clockwise (90) or counter-clockwise (-90). Does not work with columns, and bookmarks will not be correctly placed (custom attribute)
	<code>autosize</code>	Shrinks a table to fit if width is too small to allow complete words to fit. The value (must be >=1) determines the maximum allowable factor to shrink i.e. autosize="2" will allow the font-size to be reduced to a minimum of 1/2 the original size. A value of 1 prevents automatic resizing of the table. (custom property)
	<code>topntail</code>	Sets border at top and bottom of table, and below THEAD row if present. (custom attribute)
	<code>thead-underline</code>	Sets border at bottom of THEAD row if present. (custom attribute added mPDF v1.1)
	<code>line-height</code>	Sets default line-height for table cells as factor or % (value e.g. 1.3)
	<code>line-stacking-strategy</code>	inline-line-height block-line-height max-height grid-height As per draft CSS3 spec. (mPDF >= 6.0) Default = inline-line-height
	<code>line-stacking-shift</code>	consider-shifts disregard-shifts As per draft CSS3 spec. (mPDF >= 6.0) Default = consider-shifts
	<code>overflow</code>	auto hidden visible wrap Controls table layout if minimum width is too wide for page.
	<code>direction</code>	rtl ltr (mPDF >= 5.1)

HTML Tag	Property	Values allowed & Notes
CAPTION	caption-side	top bottom (mPDF \geq 5.4) Right and left are not supported
THEAD, TFOOT	font-weight vertical-align	normal bold top middle bottom
	text-align	left center right
TR	background-color <i>background-image</i> , <i>background-image-opacity</i>	COLOR background is parsed as per the CSS specification. <i>background-image</i> in the form url('image.png') or url('image.png') or GRADIENT . <i>background-position</i> is supported as per CSS2.1 <i>background-repeat</i> : repeat repeat-x repeat-y no-repeat <i>background-attachment</i> is parsed but has no effect. See note below. (cf. Backgrounds & borders) (All except background-color added mPDF 5.1)
	border, border-right, border-left, border-top, border-bottom	Size style and colour e.g. LENGTH double solid dashed dotted ridge outset groove inset hidden none COLOR NB Table & cell borders accept: double solid dashed dotted ridge outset groove inset hidden none (other block elements only support solid dotted dashed double) (mPDF \geq 5.1)
	border-color*, border-top-color, border-right-color, border-bottom-color, border-left-color	COLOR (mPDF \geq 5.1)
	border-width*, border-top-width, border-right-width, border-bottom-width, border-left-width	LENGTH (mPDF \geq 5.1)
	border-style*, border-top-style, border-right-style, border-bottom-style, border-left-style	double solid dashed dotted ridge outset groove inset hidden none (mPDF \geq 5.1)
	<i>text-rotate</i>	Rotate the text inside table cells for this row. Accepted values are 1 - 90 for degrees anticlockwise, or -90 (only) for clockwise rotation. (custom attribute)
<code>:nth-child()</code>	SELECTOR	odd even <i>a</i> + <i>b</i> As per CSS3 specification

HTML Tag	Property	Values allowed & Notes
TD, TH	background, background-image, background-position, background-repeat, background-color	background is parsed as per the CSS specification. background-image in the form url('image.png') or url('image.png') or GRADIENT (mPDF \geq 5.1). background-position is supported as per CSS2.1 background-repeat: repeat repeat-x repeat-y no-repeat background-attachment is parsed but has no effect. See note below. (All except background-color added mPDF 3.2)
	<i>background-gradient</i>	Defines a linear or radial colour gradient for the background. linear COLOR1 COLOR2 x1 y1 x2 y2; or radial COLOR1 COLOR2 x1 y1 x2 y2 radius; x, y and radius are values between 0 and 1 (custom attribute) (cf. Backgrounds & borders) (Added mPDF 3.2)
	<i>background-image-opacity</i>	Value between 0 and 1.0 Sets the opacity/transparency of the background image.
	<i>background-image-resize</i>	0 - No resizing (default) 1 - Shrink-to-fit w (keep aspect ratio) 2 - Shrink-to-fit h (keep aspect ratio) 3 - Shrink-to-fit w and/or h (keep aspect ratio) 4 - Resize-to-fit w (keep aspect ratio) 5 - Resize-to-fit h (keep aspect ratio) 6 - Resize-to-fit w and h
	border, border-right, border-left, border-top, border-bottom	Size style and colour e.g. LENGTH double solid dashed dotted ridge outset groove inset hidden none COLOR NB Table & cell borders accept: double solid dashed dotted ridge outset groove inset hidden none (other block elements only support solid dotted dashed double)
	border-color*, border-top-color, border-right-color, border-bottom-color, border-left-color	COLOR (mPDF \geq 4.0)
	border-width*, border-top-width, border-right-width, border-bottom-width, border-left-width	LENGTH (mPDF \geq 4.0)
	border-style*, border-top-style, border-right-style, border-bottom-style, border-left-style	double solid dashed dotted ridge outset groove inset hidden none (mPDF \geq 4.0)
	padding*, padding-right, padding-left, padding-top, padding-bottom	LENGTH
	width	LENGTH
	height	LENGTH (but not %) (mPDF \geq 4.0)
COMMON TEXT STYLES		
	white-space	nowrap
	vertical-align	top middle bottom
	text-align	left right center From mPDF 5.7, also supports a decimal-mark alignment followed by a default e.g. text-align: '.' center; Non-ASCII characters should be defined using Unicode values: text-align: '\66B' center;
	<i>text-rotate</i>	Rotates the text in a table cell. Accepted values are 1-90 for degrees anticlockwise, or -90 (only) for clockwise rotation. (custom attribute)
	<i>line-stacking-strategy</i>	inline-line-height block-line-height max-height grid-height As per draft CSS3 spec. (mPDF \geq 6.0) Default = inline-line-height
	<i>line-stacking-shift</i>	consider-shifts disregard-shifts As per draft CSS3 spec. (mPDF \geq 6.0) Default = consider-shifts
	hyphens	none manual auto (default = manual) As per CSS. (mPDF \geq 5.7)
	:nth-child() SELECTOR	odd even an+b As per CSS3 specification
	direction	rtl ltr (mPDF \geq 6.0)

HTML Tag	Property	Values allowed & Notes
HR	width	LENGTH
	text-align	left right center
	color	COLOR
	height	i.e. line-width. LENGTH
	margin-top, margin-bottom	LENGTH
	margin-left, margin-right	auto 0 (only) are supported as an alternative way to align a HR of less than 100% width. Values for margin-left: 0; margin-right: auto; will align the HR to the left etc.
	clear	left right both
IMG	vertical-align	top middle bottom baseline text-bottom text-top (Full support only from mPDF 4.2)
	margin, margin-right, margin-left, margin-top, margin-bottom	LENGTH
	border, border-right, border-left, border-top, border-bottom	Must be full declaration of size style and colour e.g. 0.1pt solid dotted dashed #cccccc Will also accept #cccccc 0.1pt solid (which is generated by IE WYSIWYG editor) and (from mPDF 1.4) solid 3mm #000000 medium thin thick accepted for size
	padding*, padding-right, padding-left, padding-top, padding-bottom	LENGTH
	background-color	COLOR (mPDF >= 5.7.3)
	opacity	Value between 0 and 1.0 CSS3 recommended property (but already supported by FireFox) Sets the opacity/transparency of the image.
	image-orientation	ANGLE as deg, rad, or grad (as per draft CSS3) mPDF >= 5.1
	image-resolution	normal [from-image <dpi>](as per draft CSS3) mPDF >= 5.1
	image-rendering	auto crisp-edges optimizequality smooth (as per draft CSS3) (mPDF >= 6.0) Controls whether interpolation is on or off in PDF document. Once set for an image, subsequent use of the same image will use the initial setting for this property. auto (default) - uses the value set by config variable: \$interpolateImages crisp-edges - interpolation disabled optimizequality - interpolation enabled smooth - interpolation enabled
	gradient-mask	GRADIENT (see notes below) mPDF >= 5.1
	display	none
	visibility	visible hidden <i>printonly</i> <i>screenonly</i> (mPDF >= 5.4)
	transform	All CSS3 transform functions are supported except matrix() i.e. translate(), translateX(), translateY() skew(), skewX(), skewY() scale(), scaleX(), scaleY() rotate() cf. http://www.w3.org/TR/css3-transforms/#typedef-transform-function
SELECT	float	left right (cf. Images)
	z-index	Sets a layer in the PDF document. See Layers .
	(vspace, hspace)	Attributes - set values for margin-left/right or margin-top/bottom
	width, height	LENGTH . NB the inline attributes width="" and height="" are also supported
	min-width, min-height, max-width, max-height	LENGTH . (mPDF >= 5.6)
	font-family	FONT-FAMILY
	font-size	FONT-SIZE
	color	COLOR

HTML Tag	Property	Values allowed & Notes
TEXTAREA	width, height	LENGTH . NB the inline attributes cols="" and rows="" are also supported
	font-family	FONT-FAMILY
	font-size	FONT-SIZE
	color	COLOR
	vertical-align	top middle bottom baseline text-bottom text-top (Full support only from mPDF 4.2)
	border-color	COLOR Active Forms only (mPDF >= 5.3)
INPUT (type=IMAGE)	background-color	COLOR Active Forms only (mPDF >= 5.3)
	width, height	LENGTH
INPUT (type=BUTTON SUBMIT RESET)	font-family	FONT-FAMILY Active Forms only (mPDF >= 5.3)
	font-size	FONT-SIZE Active Forms only (mPDF >= 5.3)
	color	COLOR Active Forms only (mPDF >= 5.3)
	border-color	COLOR Active Forms only (mPDF >= 5.3)
	background-color	COLOR Active Forms only (mPDF >= 5.3)
INPUT (type=CHECKBOX RADIO)	font-size	FONT-SIZE (sets size of glyph). Not inherited.
	color	COLOR . Not inherited.
INPUT (type=PASSWORD TEXT)	width	LENGTH . NB the inline attribute size="" is also supported
	font-family	FONT-FAMILY
	font-size	FONT-SIZE
	color	COLOR
	border-color	COLOR Active Forms only (mPDF >= 5.3)
BR	background-color	COLOR Active Forms only (mPDF >= 5.3)
	clear	left right both
DOTTAB	<i>outdent</i>	LENGTH . (mPDF >= 5.7)
@page @page <named>	size	<length>{1,2} auto portrait landscape Note: 'em' 'ex' and % are not allowed Length values are width and height e.g. size: 8.5in 11in; or one value for a square. Sets the size of the 'page-box', which is usually used with a constant size sheet through the document, as in the CSS2 @paged media spec.
	sheet-size	<length>{2} A4 A4-L etc. Note: 'em' 'ex' and % are not allowed Length values are width and height e.g. size: 8.5in 11in; Any of the standard sheet sizes can be used (as for mPDF()) with the suffix '-L' for landscape
	odd-header-name, even-header-name, odd-footer-name, even-footer-name	A named Header or Footer. HTML headers/footers must precede the name with <i>html</i> _ NB This was the original form, and still takes preference over header and footer which can be set using the pseudo-selectors e.g. :right The name _default can be used to allow the current non-HTML header to remain unchanged (mPDF >= 5.1)
	margin, margin-left, margin-right, margin-top, margin-bottom	LENGTH (% refers to page-box width for left/right, of height for top/bottom)

HTML Tag	Property	Values allowed & Notes
<code>@page</code>	<code>margin-top, margin-bottom</code>	LENGTH (% refers to page-box width for left/right, of height for top/bottom) Note that left and right margins cannot be changed when using a page selector. <code>@page :first</code> is not recognised unless <code>@page</code> has been set with some CSS properties
<code>@page :right</code>	<code>margin-header, margin-footer</code>	LENGTH
<code>@page :left</code>	<code>marks</code>	<code>crop cross none</code> From mPDF 5.1 crop and cross can be used together.
<code>@page :first</code>	<code>header, footer</code>	A named Header or Footer. HTML headers/footers must precede the name with <code>html_</code> (from mPDF 4.2)
<code>@page <named></code>	<code>background, background-image, background-position, background-repeat, background-color</code>	background is parsed as per the CSS specification. background-image in the form <code>url(image.png)</code> or <code>url('image.png')</code> supported. background-position is supported as per CSS2.1 background-repeat: <code>repeat repeat-x repeat-y no-repeat</code> background-attachment is parsed but has no effect. See note below. (cf. Backgrounds & borders)
	<code>background-image-opacity</code>	Value between 0 and 1.0 Sets the opacity/transparency of the background image.
	<code>background-image-resize</code>	0 - No resizing (default) 1 - Shrink-to-fit w (keep aspect ratio) 2 - Shrink-to-fit h (keep aspect ratio) 3 - Shrink-to-fit w and/or h (keep aspect ratio) 4 - Resize-to-fit w (keep aspect ratio) 5 - Resize-to-fit h (keep aspect ratio) 6 - Resize-to-fit w and h
	<code>background-gradient</code>	Defines a linear or radial colour gradient for the background. linear <code>COLOR1 COLOR2 x1 y1 x2 y2</code> ; or radial <code>COLOR1 COLOR2 x1 y1 x2 y2 radius</code> ; x, y and radius are values between 0 and 1 (custom attribute) (cf. Backgrounds & borders)
	<code>resetpagenum</code>	INTEGER Reset page numbering. Use with <code>:first</code>
	<code>pagenumstyle</code>	<code>1 A a I i</code> See <code><pagebreak></code> for details
	<code>suppress</code>	<code>on off 1 0</code> See <code><pagebreak></code> for details

Common Text Styles

COMMON TEXT STYLES	font-family	FONT-FAMILY
	font-size	FONT-SIZE
	font-style	italic oblique normal
	font-weight	bold normal (only)
	font	Shorthand form, as per CSS2 specification (mPDF >= 4.0)
	font-variant	normal none [<common-lig-values> <discretionary-lig-values> <historical-lig-values> <contextual-alt-values> historical-forms small-caps all-small-caps petite-caps all-petite-caps unicase titling-caps] <numeric-figure-values> <numeric-spacing-values> <numeric-fraction-values> ordinal slashed-zero] (mPDF >= 6.0)
	font-variant-position	normal sub super (mPDF >= 6.0)
	font-variant-caps	normal small-caps all-small-caps petite-caps all-petite-caps unicase titling-caps (mPDF >= 6.0)
	font-variant-ligatures	normal none [<common-lig-values> <discretionary-lig-values> <historical-lig-values> <contextual-alt-values>] (mPDF >= 6.0) <common-lig-values> = [common-ligatures no-common-ligatures] <discretionary-lig-values> = [discretionary-ligatures no-discretionary-ligatures] <historical-lig-values> = [historical-ligatures no-historical-ligatures] <contextual-alt-values> = [contextual no-contextual]
	font-variant-numeric	normal [<numeric-figure-values> <numeric-spacing-values> <numeric-fraction-values> ordinal slashed-zero] (mPDF >= 6.0)
font-variant-alternates	font-variant-alternates	Only [normal historical-forms] supported (i.e. most are NOT supported) (mPDF >= 6.0)
	font-language-override	3-letter case-sensitive OpenType language system tag (mPDF >= 6.0)
	font-feature-settings	Comma separated list of <feature-tag-value> with the following syntax: <feature-tag-value> = <string> [<integer> on off]? (mPDF >= 6.0)
	font-kerning	auto normal none (mPDF >= 5.1) auto and normal both have the effect of turning kerning on, as long as <code>\$mpdf->useKerning = true;</code>
	vertical-align	super sub baseline LENGTH (mPDF >= 5.7.3)
	letter-spacing	normal LENGTH (mPDF >= 5.1)
	word-spacing	normal LENGTH (mPDF >= 5.1)
	color	COLOR
	text-decoration	underline line-through normal (line-through = strike-through) (mPDF >= 5.4) overline (mPDF >= 5.7.3)
	text-outline	thickness [blur] color; OR none (default) Blur is not supported. As per CSS3 (mPDF >= 5.7) Supported in tables >= 6.0
text-outline-width, <i>outline-width</i>	text-outline-width, <i>outline-width</i>	Width of the outline. LENGTH (text-outline-width as from mPDF >= 5.7) Supported in tables >= 6.0
	text-outline-color, <i>outline-color</i>	COLOR = colour of the inner part of the text e.g. #rrggbb. 'INVERT' is also accepted. (text-outline-color as from mPDF >= 5.7) Supported in tables >= 6.0
text-shadow	text-shadow	As per CSS3 specification. 'blur' is not supported. (mPDF >= 5.4)
	text-transform	capitalize uppercase lowercase (mPDF >= 5.4)
	unicode-bidi	normal embed isolate bidi-override isolate-override plaintext (mPDF >= 6.0) When used on block-level elements: - the value is not inherited to child blocks - using "embed" or "isolate" has no effect on block level-boxes - "isolate-override" is equivalent to "bidi-override" on block-level boxes

Notes:

* Margin, padding, border-width, border-color and border-style accept the various shorthand forms e.g.:

margin:1pt; will set all top, right, bottom and left values the same

margin:1pt 2pt; will set top and bottom to 1pt, left and right to 2pt

margin:1pt 2pt 3pt; will set top to 1pt, left and right to 2pt, and bottom to 3pt (mPDF >= 4.0)

margin:1pt 2pt 3pt 4pt; will set all values in order: top>right>bottom>left

NB Table page-break-inside, autosize values and rotate are only respected for that set on first level table of nested tables

Border

medium|thin|thick are accepted for size - converted to 1px, 3px, 5px

See Also

- [Backgrounds & Borders \(including Gradients\)](#)
- [HTML supported attributes](#)

Default stylesheet

mPDF uses a default "stylesheet" for default settings of style and layout; this is in config.php as a variable \$defaultCSS.

mPDF (>= 2.2) will load an optional file mpdf.css (located in the same directory as mpdf.php) which can be edited to change your default styles for PDF files.

mPDF >= 6.0 A new mpdf.css file includes defaults for Lists top/bottom margins, and also examples for Indexes and ToCs. This now acts like a normal CSS file, including cascading selectors i.e. not just main tags. This is always read (if present), so acts as a secondary default CSS, but one which allows selectors. Styles added to this act like a user stylesheet when considering precedence e.g. cellSpacing and border-spacing. (The following text describes behaviour prior to mPDF v6.0)

The file should be a valid CSS stylesheet, but will only support changing properties at element level i.e. P, DIV, TABLE, TD and not P.classname.

The mpdf.css file supplied is inactive as it has all the entries commented out by /* ... */

The following values will restore behaviour of pre-4.2 versions to 4.2:

```
img { margin: 0.83em 0; vertical-align: bottom; }
table { margin: 0.5em; }
textarea { vertical-align: top; }
```

The following values will restore behaviour of 2.2 versions:

```
body {
    font-family: sans-serif;
}
a {
    color: #000066;
    text-decoration: none;
}
table {
    border-collapse: collapse;
}
thead {
    vertical-align: bottom;
    text-align: center;
    font-weight: bold;
}
tfoot {
    text-align: center;
    font-weight: bold;
}
th {
    text-align: left;
    padding-left: 0.35em;
    padding-right: 0.35em;
    padding-top: 0.35em;
    padding-bottom: 0.35em;
    vertical-align: top;
}
td {
    padding-left: 0.35em;
    padding-right: 0.35em;
    padding-top: 0.35em;
    padding-bottom: 0.35em;
    vertical-align: top;
}
img {
    margin: 0.2em;
    vertical-align: middle;
}
```

Prior to mPDF 2.2

The original default styles (mPDF <=2.0) were extensively changed with improvements in CSS handling in mPDF 2.0 (this particularly changed table borders, and table cell alignment, as well as a serif default font-family).

In order to allow backwards compatibility, a secondary "stylesheet" - a variable \$defaultCSS2 was introduced in mPDF 2.0 - and the option to load this on initiating mPDF:

```
$mpdf->useDefaultCSS2 = true;
```

Secondary default CSS values used mPDF 2.0 - 2.1

```
var $defaultCSS2 = array(
    'BODY' => array(
        'FONT-FAMILY' => 'sans-serif',
    ),
    'A' => array(
        'COLOR' => '#000066',
        'TEXT-DECORATION' => '',
    ),
    'TABLE' => array(
        'BORDER-COLLAPSE' => 'collapse',
    ),
    'THEAD' => array(
        'VERTICAL-ALIGN' => 'bottom',
        'TEXT-ALIGN' => 'center',
        'FONT-WEIGHT' => 'bold',
    ),
    'TFOOT' => array(
        'TEXT-ALIGN' => 'center',
        'FONT-WEIGHT' => 'bold',
    ),
    'TH' => array(
        'TEXT-ALIGN' => '',
        'PADDING-LEFT' => '0.35em',
        'PADDING-RIGHT' => '0.35em',
        'PADDING-TOP' => '0.35em',
        'PADDING-BOTTOM' => '0.35em',
        'VERTICAL-ALIGN' => 'top',
    ),
    'TD' => array(
        'PADDING-LEFT' => '0.35em',
        'PADDING-RIGHT' => '0.35em',
        'PADDING-TOP' => '0.35em',
        'PADDING-BOTTOM' => '0.35em',
        'VERTICAL-ALIGN' => 'top',
    ),
    'IMG' => array(
        'MARGIN' => '0.2em',
        'VERTICAL-ALIGN' => 'middle',
    ),
);
);
```

Named colours

Named colours supported by mPDF in CSS

The table below provides a list of the named colours supported by mPDF when used in CSS contexts. The list is precisely the same as the [SVG 1.0 color keyword names](#). The two color swatches on the left illustrate setting the background color of a table cell in two ways: The first column uses the named color value, and the second column uses the respective numeric color value.

Named	Numeric	Color name	Hex rgb	Decimal
		aliceblue	#f0f8ff	240,248,255
		antiquewhite	#faebd7	250,235,215
		aqua	#00ffff	0,255,255
		aquamarine	#7fffd4	127,255,212
		azure	#f0ffff	240,255,255
		beige	#f5f5dc	245,245,220
		bisque	#ffe4c4	255,228,196
		black	#000000	0,0,0
		blanchedalmond	#ffebcd	255,235,205
		blue	#0000ff	0,0,255
		blueviolet	#8a2be2	138,43,226
		brown	#a52a2a	165,42,42
		burlywood	#deb887	222,184,135
		cadetblue	#5f9ea0	95,158,160
		chartreuse	#7fff00	127,255,0
		chocolate	#d2691e	210,105,30
		coral	#ff7f50	255,127,80
		cornflowerblue	#6495ed	100,149,237
		cornsilk	#fff8dc	255,248,220
		crimson	#dc143c	220,20,60
		cyan	#00ffff	0,255,255
		darkblue	#00008b	0,0,139
		darkcyan	#008b8b	0,139,139
		darkgoldenrod	#b8860b	184,134,11
		darkgray	#a9a9a9	169,169,169
		darkgreen	#006400	0,100,0
		darkgrey	#a9a9a9	169,169,169
		darkkhaki	#bdb76b	189,183,107
		darkmagenta	#8b008b	139,0,139
		darkolivegreen	#556b2f	85,107,47
		darkorange	#ff8c00	255,140,0
		darkorchid	#9932cc	153,50,204
		darkred	#8b0000	139,0,0
		darksalmon	#e9967a	233,150,122
		darkseagreen	#8fbcbf	143,188,143
		darkslateblue	#483d8b	72,61,139
		darkslategray	#2f4f4f	47,79,79
		darkslategrey	#2f4f4f	47,79,79
		darkturquoise	#00ced1	0,206,209
		darkviolet	#9400d3	148,0,211
		deeppink	#ff1493	255,20,147

Named	Numeric	Color name	Hex rgb	Decimal
		deepskyblue	#00bfff	0,191,255
		dimgray	#696969	105,105,105
		dimgrey	#696969	105,105,105
		dodgerblue	#1e90ff	30,144,255
		firebrick	#b22222	178,34,34
		floralwhite	#ffffaf	255,250,240
		forestgreen	#228b22	34,139,34
		fuchsia	#ff00ff	255,0,255
		gainsboro	#dcdcdc	220,220,220
		ghostwhite	#f8f8ff	248,248,255
		gold	#ffd700	255,215,0
		goldenrod	#daa520	218,165,32
		gray	#808080	128,128,128
		green	#008000	0,128,0
		greenyellow	#adff2f	173,255,47
		grey	#808080	128,128,128
		honeydew	#f0ffff	240,255,240
		hotpink	#ff69b4	255,105,180
		indianred	#cd5c5c	205,92,92
		indigo	#4b0082	75,0,130
		ivory	#fffff0	255,255,240
		khaki	#f0e68c	240,230,140
		lavender	#e6e6fa	230,230,250
		lavenderblush	#fff0f5	255,240,245
		lawngreen	#7fc000	124,252,0
		lemonchiffon	#ffffac	255,250,205
		lightblue	#add8e6	173,216,230
		lightcoral	#f08080	240,128,128
		lightcyan	#e0ffff	224,255,255
		lightgoldenrodyellow	#fafad2	250,250,210
		lightgray	#d3d3d3	211,211,211
		lightgreen	#90ee90	144,238,144
		lightgrey	#d3d3d3	211,211,211
		lightpink	#ffb6c1	255,182,193
		lightsalmon	#ffa07a	255,160,122
		lightseagreen	#20b2aa	32,178,170
		lightskyblue	#87cefa	135,206,250
		lightslategray	#778899	119,136,153
		lightslategrey	#778899	119,136,153
		lightsteelblue	#b0c4de	176,196,222
		lightyellow	#ffffe0	255,255,224
		lime	#00ff00	0,255,0
		limegreen	#32cd32	50,205,50
		linen	#faf0e6	250,240,230
		magenta	#ff00ff	255,0,255
		maroon	#800000	128,0,0
		mediumaquamarine	#66cdAA	102,205,170
		mediumblue	#0000cd	0,0,205
		mediumorchid	#ba55d3	186,85,211
		mediumpurple	#9370db	147,112,219

Named	Numeric	Color name	Hex rgb	Decimal
		mediumseagreen	#3cb371	60,179,113
		mediumslateblue	#7b68ee	123,104,238
		mediumspringgreen	#00fa9a	0,250,154
		mediumturquoise	#48d1cc	72,209,204
		mediumvioletred	#c71585	199,21,133
		midnightblue	#191970	25,25,112
		mintcream	#fffffa	245,255,250
		mistyrose	#ffe4e1	255,228,225
		moccasin	#ffe4b5	255,228,181
		navajowhite	#ffdead	255,222,173
		navy	#000080	0,0,128
		oldlace	#fdf5e6	253,245,230
		olive	#808000	128,128,0
		olivedrab	#6b8e23	107,142,35
		orange	#ffa500	255,165,0
		orangered	#ff4500	255,69,0
		orchid	#da70d6	218,112,214
		palegoldenrod	#eee8aa	238,232,170
		palegreen	#98fb98	152,251,152
		paleturquoise	#afeeee	175,238,238
		palevioletred	#db7093	219,112,147
		papayawhip	#ffefd5	255,239,213
		peachpuff	#ffdab9	255,218,185
		peru	#cd853f	205,133,63
		pink	#ffc0cb	255,192,203
		plum	#dda0dd	221,160,221
		powderblue	#b0e0e6	176,224,230
		purple	#800080	128,0,128
		red	#ff0000	255,0,0
		rosybrown	#bc8f8f	188,143,143
		royalblue	#4169e1	65,105,225
		saddlebrown	#8b4513	139,69,19
		salmon	#fa8072	250,128,114
		sandybrown	#f4a460	244,164,96
		seagreen	#2e8b57	46,139,87
		seashell	#fff5ee	255,245,238
		sienna	#a0522d	160,82,45
		silver	#c0c0c0	192,192,192
		skyblue	#87ceeb	135,206,235
		slateblue	#6a5acd	106,90,205
		slategray	#708090	112,128,144
		slategrey	#708090	112,128,144
		snow	#fffffa	255,250,250
		springgreen	#00ff7f	0,255,127
		steelblue	#4682b4	70,130,180
		tan	#d2b48c	210,180,140
		teal	#008080	0,128,128
		thistle	#d8bfd8	216,191,216
		tomato	#ff6347	255,99,71
		turquoise	#40e0d0	64,224,208

Named	Numeric	Color name	Hex rgb	Decimal
		violet	#ee82ee	238,130,238
		wheat	#f5deb3	245,222,179
		white	#ffffff	255,255,255
		whitesmoke	#f5f5f5	245,245,245
		yellow	#ffff00	255,255,0
		yellowgreen	#9acd32	154,205,50

SETTING PDF FILE PROPERTIES

Password protection

The security settings for the PDF file can be set in mPDF:

- determine whether the document is encrypted
- if encrypted, what permissions are granted to the user e.g.print and/or copy
- determine whether a user requires a password to open the document
- determine whether a password is required to modify the document (owner's password)

See Also

- [SetProtection\(\)](#) - Encrypts and sets the PDF document permissions

Document Metadata

A PDF file contains metadata about the title, author, subject, creation date, keywords. The title is usually shown in the top of the screen when a user views the file; the rest of the metadata can be accessed by viewing Document properties in Adobe Reader.

You can set the metadata directly using:

- [SetTitle\(\)](#)
- [SetAuthor\(\)](#)
- [SetCreator\(\)](#)
- [SetSubject\(\)](#)
- [SetKeywords\(\)](#)

Metadata is also set automatically if you pass full HTML code to [WriteHTML\(\)](#) including:

- title is read from <title>...</title> tags
- subject, keywords and author are read from <meta ... /> tags

Whichever is set later will override previous settings.

The text should be UTF-8 encoded, but should not contain HTML mark-up tags. [strcode2utf\(\)](#) is a useful function provided with mPDF which converts HTML numerical entities to UTF-8 encoded string.

Note: Adobe Reader uses system fonts to display the document metadata, therefore any Unicode text can be used, even if core fonts only are being used for the document.

Example

```
<?php  
  
$mpdf=new mPDF();  
  
$md = strcode2utf("ايلات &#1601;يما  
ايلات &#1601;يما");  
  
$mpdf->SetTitle($md);  
$mpdf->WriteHTML('<p>Hello World</p>');  
$mpdf->Output();  
  
?>
```

See Also

- [SetTitle\(\)](#) - Set document Title in metadata
- [SetAuthor\(\)](#) - Set document Author in metadata
- [SetCreator\(\)](#) - Set document Creator in metadata
- [SetSubject\(\)](#) - Set document Subject in metadata
- [SetKeywords\(\)](#) - Set document Keywords in metadata
- [strcode2utf\(\)](#) - Convert HTML numerical entities to UTF-8 encoded string

PDF Version

mPDF >=4.3 produces PDF files which are specified as version %PDF-1.4.

1.4 is the lowest specification which contains all the features used in mPDF - with a few minor exceptions:

- `SetDisplayPreferences($preferences)` allows the option to include "NoPrintScaling" (PDF-1.6)
- Using `mirrorMargins`, mPDF will set by default /Duplex /DuplexFlipLongEdge (PDF-1.7)
- The "Subject" field of Annotations (PDF-1.5)

Previous versions

mPDF <2.0 produced PDF files which were specified as version 1.3 (the file starts with %PDF-1.3).

mPDF >=2.0 was changed to specify %PDF-1.5, to allow for changes needed using Annotations.

WHAT ELSE CAN I DO?

Backgrounds & Borders

Backgrounds

Background colour (colour is the only property) can be set for in-line elements e.g.

3 types of background can be set for block elements (div, p etc), @page, and for the <body> element:

- background color
- background images (including gradients using CSS3 spec.)
- *background gradients (custom style for mPDF - not part of CSS specification - DEPRACATED)*

Background Images

mPDF supports most CSS properties to control background-images, including image resolution, opacity and transparency. See [Supported CSS](#).

Background-images support JPG, GIF, PNG, WMF and SVG images.

Background-images are disabled in columns, and when "page-break-inside: avoid" is used to keep a block together.

NB CSS2.1 states that the area for background-image should include the padding and the BORDER - IE7 does this, but Firefox 3 starts the tiling position 0 inside the border. mPDF complies with CSS and IE7.

Note: Background-color and background-image set on the <body> element will cover the whole page i.e. not inside the "margins". (cf. [Backgrounds & borders](#))

Gradients defined as background-image

(mPDF >= 5.1) Gradients can be defined as background-image: both the Mozilla or the draft CSS3 syntax are supported e.g.:

```
background: -moz-repeating-linear-gradient(red, blue 20px, red 40px)
background-image: -moz-repeating-linear-gradient(red, blue 20px, red 40px)
background: linear-gradient(top, #c7cdde, #f0f2ff);
background-image: linear-gradient(top, #c7cdde, #f0f2ff);
```

CSS3/Mozilla gradients in mPDF support: multiple colour-stops, opacity, repeating-gradients, and a number of options for defining the gradient axis (linear gradients) or shape and extent (radial gradients)

For more details see:

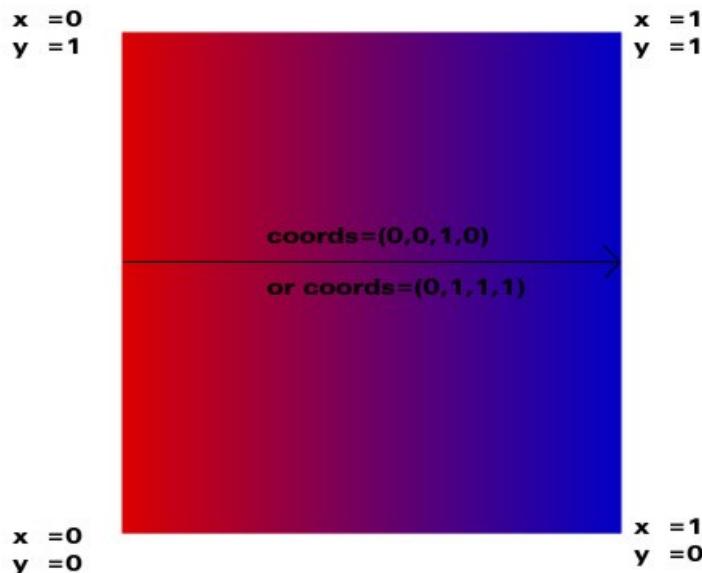
- Mozilla linear - <https://developer.mozilla.org/en/CSS/-moz-linear-gradient>
- Mozilla radial - <https://developer.mozilla.org/en/CSS/-moz-radial-gradient>
- Mozilla gradients use - https://developer.mozilla.org/en/Using_gradients
- CSS3 linear gradients - <http://dev.w3.org/csswg/css3-images/#linear-gradients>
- CSS3 radial gradients - <http://dev.w3.org/csswg/css3-images/#radial-gradients>

Background-gradient (Old form - DEPRACATED)

Background gradient can be set as a linear or radial gradient between two colours. Background gradients can be set on all block elements e.g. P, DIV, H1-H6, as well as @page and on BODY.

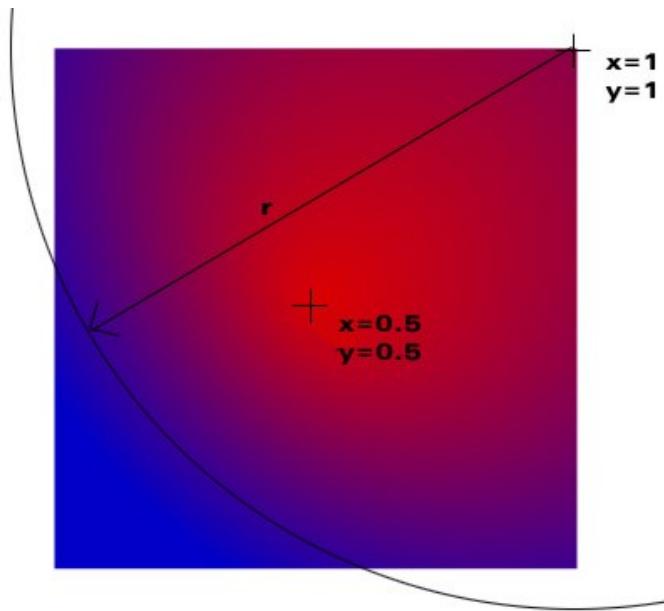
```
background-gradient: linear #c7cdde #f0f2ff 0 1 0 0.5;
```

The four numbers are coordinates in the form (x1, y1, x2, y2) which defines the gradient vector. x and y are values from 0 to 1, where 1 represents the height or width of the box as it is printed.



```
background-gradient: radial #00FFFF #FFFF00 0.5 0.5 0.5 0.5 0.65;
```

The five numbers are coordinates in the form (x_1, y_1, x_2, y_2, r) where (x_1, y_1) is the starting point of the gradient with color1, (x_2, y_2) is the center of the circle with color2, and r is the radius of the circle. (x_1, y_1) should be inside the circle, otherwise some areas will not be defined.



Borders

In addition to the standard CSS styles for borders, border-radius and background-clip are supported. these are defined in the [CSS3 draft specification](#).

Note: Border-radius does not work if Columns are being used, or if \$use_kwt is **TRUE** (keep-with-table).

The two length values of the 'border-*-radius' properties define the radii of a quarter ellipse that defines the shape of the corner of the outer border edge.

The first value is the horizontal radius e.g. in 'border-top-left-radius: 55pt 25pt' 55pt is radius of curve from top end of left border starting to go round to the top.

If the second length is omitted it is equal to the first (and the corner is thus a quarter circle). If either length is zero, the corner is square, not rounded.

The ‘border-radius’ shorthand sets all four ‘border-*radius’ properties. If values are given before and after the slash, then the values before the slash set the horizontal radius and the values after the slash set the vertical radius. If there is no slash, then the values set both radii equally. The four values for each radii are given in the order top-left, top-right, bottom-right, bottom-left. If bottom-left is omitted it is the same as top-right. If bottom-right is omitted it is the same as top-left. If top-right is omitted it is the same as top-left.

```
border-radius: 4em;
```

would be equivalent to

```
border-top-left-radius: 4em;
border-top-right-radius: 4em;
border-bottom-right-radius: 4em;
border-bottom-left-radius: 4em;
```

and

```
border-radius: 2em 1em 4em / 0.5em 3em;
```

would be equivalent to

```
border-top-left-radius: 2em 0.5em;
border-top-right-radius: 1em 3em;
border-bottom-right-radius: 4em 0.5em;
border-bottom-left-radius: 1em 3em;
```

Border for In-line elements

A border can be also be specified for in-line elements, but with more limited options for the properties. See [Supported CSS](#).

Auto-padding

Note that mPDF does not by default clip/wrap the content of the box to allow for the curved border (which CSS3 states should). Therefore you need to allow enough padding inside the box. Alternatively you can set the variable `$autoPadding` to **TRUE** (default=**FALSE**); this automatically increases the padding in block elements with `border-radius` set - as required.

Background clipping

```
background-clip : padding-box | border-box (default=border-box)
```

defines whether the background is painted to the inside or outside edge of the borders.

Example

```
div.rounded {
    border:1mm dashed #220044;
    background-color: #f0f2ff;
    border-radius: 3mm / 3mm;
    background-clip: border-box;
    padding: 1em;
}
```

Fixed position blocks

mPDF >= 4.0

Position

The CSS property "position" is **partially** supported in mPDF, allowing a block element (div etc.) to be placed at a fixed position on the page.

```
position: fixed|absolute
```

absolute - treats the whole physical page as the containing element

fixed - treats the 'printable page' (inside the margins) as the containing element

```
top|left|bottom|right: LENGTH (any valid length unit: em, mm, px, % etc.) DEFAULT = auto
```

rotate - (introduced mPDF 5.0) rotates the block element 90 degrees clockwise or anticlockwise

top|left|bottom|right|width|height are all used to set the size of the block element BEFORE rotation i.e. if the width is not specified, the left and right margins will be used to calculate the width. After rotation, *left|top* are used to position the top left corner of the (rotated) element. There is an exception if bottom or right are specified without the respective top or left values; in this case these values are used to position the bottom or right edge of the (rotated) element.

```
90|-90 DEFAULT = BLANK
```

Limitations:

- There is no equivalent concept of the viewport as in the browser (in CSS)
- All positioning is relative to the current page of the document being written
- Position is overridden if it would be off the page, so that the element displays within the containing element
- Overflow:auto causes text to autofit within the block size (additionally constrained if necessary to page edges).
- Fixed-position or floating elements nested inside other fixed-position or floating elements are not supported
- Probably INCOMPATIBLE with keep-with-table, columns etc.
- Annotations were disabled prior to mPDF 5.0

Note: width (including the value 'auto'), height, margin-left -right -top -bottom, padding-left -right -top -bottom, are all supported.

Note: Using overflow:auto can cause mPDF to be very slow, as it attempts to write the same HTML reiteratively until it finds a reasonable fit to the available size.

Overflow

The CSS property "overflow" determines how text is displayed if the block element size is too small for the text.

```
overflow: visible|hidden|auto DEFAULT = visible
```

visible - all text will show, even if it lies outside the defined block element

hidden - text will be 'clipped' so that overflowing text is not visible

auto - text will be automatically reduced in size if required to fit inside the block element

Example

```
Example #1
```

```
<div style="position: absolute; top: 50mm; left: 50mm; width: 100mm;">
```

```
This is text in a fixed position block element.  
</div>
```

Example #2 - Centres a block in the middle of the page

```
<style>  
.myfixed {  
    position: absolute;  
    overflow: visible;  
    left: 0;  
    right: 0;  
    width: 100mm; /* you must specify a width */  
    margin-top: auto;  
    margin-bottom: auto;  
    margin-left: auto;  
    margin-right: auto;  
    border: 1px solid #000088;  
    background-color: #EEDDDFF;  
}  
</style>  
<div class="myfixed">Cum sociis natoque penatibus et magnis dis parturient montes, nascetur  
ridiculus mus. Donec mattis lacus ac purus feugiat semper. Donec aliquet nunc odio, vitae  
pellentesque diam. Pellentesque sed velit lacus. Duis quis dui quis sem consectetur sollicitudin.  
Cras dolor quam, dapibus et pretium sit amet, elementum vel arcu.</div>
```

Example #3 - Rotated barcode at the bottom right corner of the page

```
<div style="position: fixed; right: 0mm; bottom: 0mm; rotate: -90;">  
<barcode code="978-0-9542246-0" class="barcode" />  
</div>
```

Floating blocks

mPDF >= 3.0

Float

The CSS property "float" is **partially** supported in mPDF, allowing block elements (p, div etc.) to be placed alongside one another. They can also be used to create "columns" that span more than one page.

```
float: right|left
```

Limitations:

- Float only works properly if a width is set for the float
- If no width is set, the width is set to the maximum available (full width, or less if floats already set)
- A block element next to a float has the padding adjusted so that content fits in the remaining width.
- Text next to float should wrap correctly, but backgrounds and borders will overlap and/or lie under the floats in a mess
- Does not work if Columns are being used.
- You cannot change the page margins/orientation etc. in middle of using floats
- Float is only supported on block elements (i.e. not SPAN etc.)

Margin-right can still be set for a float:right and vice-versa.

Note: The width that is set defines the width of the content-box. So if you have two floats with width=50% and either of them has padding, margin or border, they will not fit together on the page.

Clear

The CSS property "clear" can be set on any block element (p, div etc.), and also <hr> or
 elements.

```
clear: right|left|both
```

Example

```
<h4>CSS Float</h4>
<div>
Some text to start with

<div style="float: right; width: 28%;">
This is text that is set to float:right.
</div>
<div style="float: left; width: 54%;">
This is text that is set to float:left.
</div>

<div style="clear: both; margin: 0pt; padding: 0pt; "></div>
This is text that follows the clear:both.

</div>
```

Hyphenation

(mPDF >= 2.5)

Hyphenation was changed in mPDF 5.7 to support the CSS property `hyphens` (cf.)

The CSS property `hyphens` is supported on block elements (e.g. `<div>`), inline elements (e.g. ``) and `<td/th>`.

`hyphens: none | manual | auto`

Default = manual

The default can be changed by altering `$defaultCSS` in `config.php`

`hyphens: manual`

none

Words are not broken at line breaks, even if characters inside the word suggest line break points.

manual

Words are only broken at line breaks where there are characters inside the word that suggest line break opportunities. Characters can be explicit or conditional.

auto

Words can be broken at appropriate hyphenation points, as determined by characters inside the word, resources.

SHY inside the word take priority over hyphenation points determined by other resources.

Soft hyphens

The soft-hyphen character (U+00AD or ­) and the `<wbr>` tag (from mPDF 5.7) are supported in `WriteHTML()`.

Automatic hyphenation

Automatic hyphenation is set using CSS:

`hyphens: auto;`

Automatic hyphenation is based on the commonly used TeX algorithm and requires pattern files for each language. The following languages are supplied with mPDF 2.5:

Language	\$SHYlang
English	en (DEFAULT)
German	de
Spanish	es
Finnish	fi
French	fr
Italian	it
Dutch	nl
Portuguese	pl
Russian	ru
Swedish	sv

A pattern file for each language is found in the folder `/patterns/` and the variable `$SHYlanguages` needs to be updated in `config.php` if any additions are made.

Pattern checking can be fine-tuned by 4 variables if required:

	Default value	
\$SHYleftmin	2	Minimum number of characters allowed to the left of a hyphen.
\$SHYrightmin	2	Minimum number of characters allowed to the right of a hyphen.
\$SHYcharmin	2	Minimum number of characters of words to be checked.
\$SHYcharmax	10	Maximum number of characters in a pattern used for pattern checking (10 usually is more than enough). this is NOT the maximum length of words to be checked.

Example #1

```
<?php
$mpdf=new mPDF();
$mpdf->SHYlang = 'fr';
$mpdf->SHYleftmin = 3;
$mpdf->WriteHTML('<p style="hyphens: auto">La grande texte....</p>');
$mpdf->Output();
?>
```

Hyphenation Dictionary

If automatic hyphenation does not recognise a particular word, you can add words to a dictionary file with your own hyphenation. Edit the file "path to your mPDF/patterns/dictionary.txt" and add a new line for each word, marking the possible hyphenations with a forward slash. You can mark more than one place for each word e.g. "dis/es/tab/lis/h/men/tar/i/an/ism"

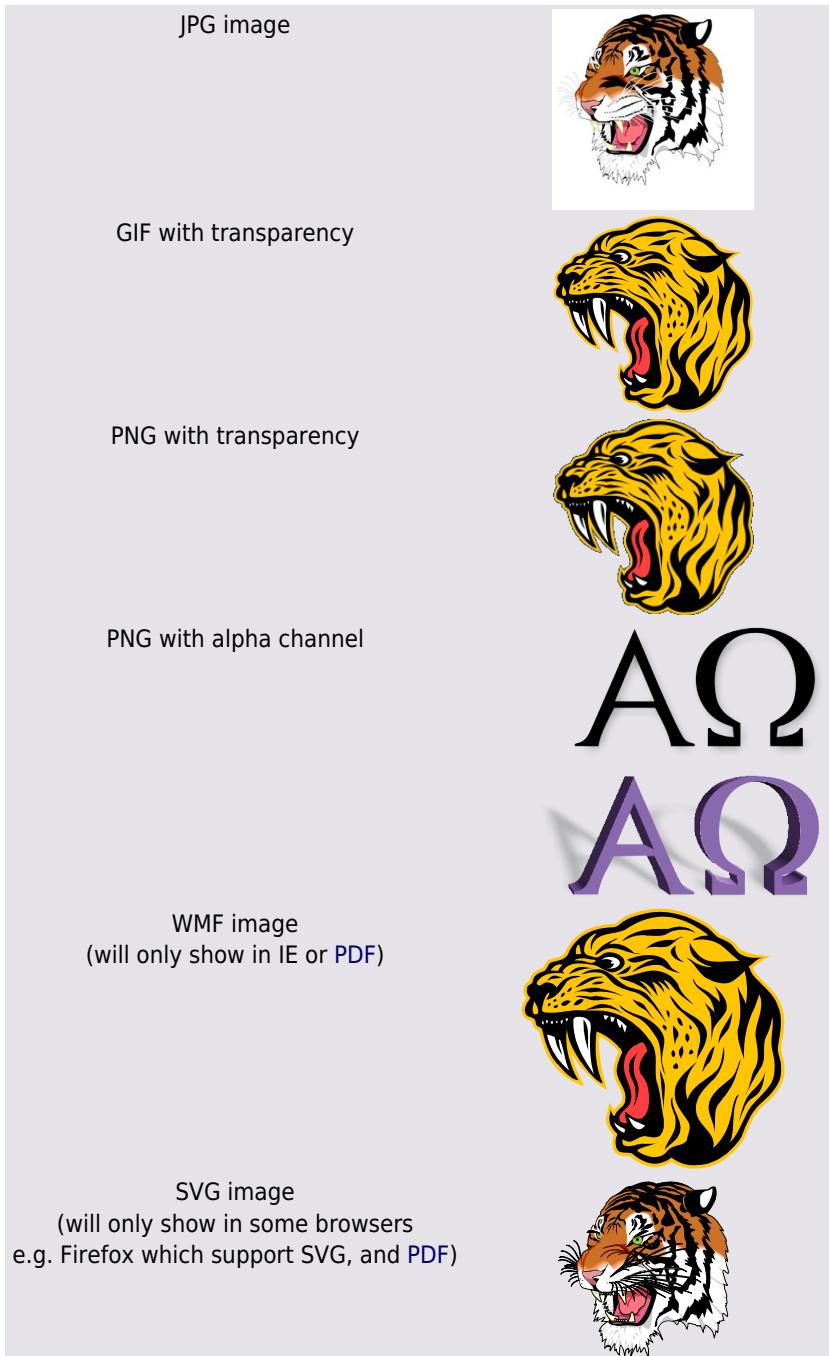
See Also

- **SHYlang** - Specify the language to use for automatic hyphenation

Images

Images are supported by mPDF: GIF, PNG, JPG, WMF, SVG, BMP and generated images from **PHP** scripts.

Transparent GIF images are supported, and so are interlaced and transparent PNG files, including transparency using alpha channel.



Debugging Errors: If you are having trouble with images not displaying correctly, set the variable `$mpdf->showImageErrors = true;`

Images are handled in mPDF as in-line elements.

Unlike the HTML specification, the width and height attributes of IMG will additionally take different dimensions e.g. mm, pt etc.

```
 // 90 pixels, just like HTML
```

```
 // non-standard support for dimensions
```

```
 // Can also use CSS
```

Images for which the size is not defined are output at a default dpi set in the config.php file:

```
$this->img_dpi = 96;
```

In addition, many CSS style properties are supported including *vertical-align*, as well as some custom attributes such as *opacity* and *rotate*. (See [Supported CSS and HTML attributes](#))

Images can be used in:

- In-line images, including in HTML headers & footers
- Watermarks
- CSS background-image

Size constraint

When writing HTML, image size is automatically constrained to current margins and page position. An extra parameter added to end of the Image() function allows you to override this:

```
$mpdf->Image('files/images/frontcover.jpg',0,0,210,297,'jpg','','true, false);
// the last "false" allows a full page picture
```

This is useful for e.g. a cover page for your document.

This can be achieved using HTML & CSS like this:

```
<div style="position: absolute; left:0; right: 0; top: 0; bottom: 0;">

</div>
```

Float and Text wrapping

mPDF partially supports the CSS style **float** with IMG elements. Text will wrap around these images, with certain limitations:

- only 1 float:right and 1 float:left image are allowed at a time i.e. you cannot overlap 2 right or 2 left
- the containing HTML element is extended at the bottom if necessary to enclose the last image (unlike your browser)
- the float is ignored if the image is too wide, inside a table, columns are on, or div page-break-inside: avoid is set
- disabled when columns are being used

If ignored, the image is output as a normal in-line element.

Performance

JPG images are quickest and most efficient.

PNG images with no alpha channel and not interlaced are next best.

GIF images are extremely slow if you do not have the appropriate GD library installed (above 2.0.8).

GIF images with GD library installed are quite fast.

PNG images which are interlaced or have alpha channel transparency, and GIF images use the GD library if it is installed.

Any process that requires the GD library uses a large amount of memory - to create a GD image in memory can use up to 10 x the file size (e.g. a 690K GIF file read into imagecreatefromstring() used about 5MB of PHP memory).

Dynamically generated Images

You can define a script which generates an image just as you can in HTML:

```

```

Embedded Image data

Embedded image data can be used either in elements or in CSS:background. gif, png and jpeg are supported.

```
[HTML]


[CSS stylesheet]
div { background: url(data:image/gif;base64,....) no-repeat 4px center; }
```

Image data as a Variable

A PHP variable containing image data can be passed directly to mPDF. You need to allocate the data to a class variable

(you can make any name up) e.g.

```
$mpdf->myvariable = file_get_contents('alpha.png');
```

The image can then be referred to by use of "var:varname" instead of a file name, either in src="" or direct to Image() function e.g.

```
$html = '';
$mpdf->WriteHTML($html);
```

```
$mpdf->Image('var: myvariable',0,0);
```

SVG images

mPDF partially supports SVG images, including as embedded HTML e.g.:

```
<p>This is an embedded SVG image:</p>
<svg>.....</svg>
```

All units with no dimension are taken as pixels.

Size of SVG image

The viewBox attribute of the <svg> element is used (if present) to determine the intrinsic size of the image.

If viewBox is not present, a width and/or height be specified e.g. width="400" height="200"

NB preserveAspectRatio is not supported.

If none of these are present, the intrinsic size will be set as the width of containing block (height = width).

Partially supported

- embedded images (but not embedded SVG/WMF [vector] images)
- tspan, tref
- <use ../>
- gradient on text (fill)
- <clipPath>
- embedded <style> elements*
- automatic font selection for text*

* As from mPDF 6.0 there is limited support for CSS classes and for automatic font selection (see the defined constants at the top of classes/svg.php file).

Not supported

- filters
- <marker>
- DOM
- <pattern>
- textlength, lengthadjust, toap or textPath
- gradient on stroke

Note: SVG images can be embedded within the HTML code. They may be a useful way to deal with some presentation issues not supported by HTML/CSS e.g. Text used with a transformation, or text used as a clipPath e.g.

```
<body>
<div>
```

```
...
<svg width="100%" height="100%" viewBox="0 0 480 360">
    <text x="100" y="-100" transform="rotate(60)" font-size="35">ROTATE</text>
    <rect id="test-frame" x="1" y="1" width="478" height="358" fill="none" stroke="#000000"/>
</svg>
...
<svg width="100%" height="100%" viewBox="0 0 480 360">
    <defs>
        <clipPath id="sample" clipPathUnits="userSpaceOnUse">
            <text x="45" y="270" font-size="100" font-family="Impact">Clip Test</text>
        </clipPath>
    </defs>
    <image xlink:href="bluesquidj.png" preserveAspectRatio="none" x="20" y="170" width="410"
height="160" clip-path="url(#sample)"/>
    <rect id="test-frame" x="1" y="1" width="478" height="358" fill="none" stroke="#000000"/>
</svg>
...
</div>
</body>
```

Kerning

(mPDF >= 6.0)

The control of Kerning is complicated! CSS3 allows for 2 methods of specifying kerning. In mPDF 6, these 2 methods have exactly the same effect:

- font-kerning: normal;
- font-feature-settings: 'kern' on;

TrueType fonts allow for 2 possible ways of including kerning data:

- OTL GPOS tables may contain kerning information
- The font may contain a separate kern table

Most fonts contain both or none, but they may exist independently.

If kerning is set to be active (by either of the CSS methods):

- if the useOTL value means that OTL GPOS tables are applied, then this method will be used;
- if not, then the separate kern table will be used - if it exists.

In Latin script, kerning will only be applied if specified by CSS. The configurable variable `useKerning` determines behaviour if `font-kerning: auto` is used (the default).

When using OTL tables, kerning is set to be on by default for non-LATIN script; this is because a number of fonts use information in the kern feature to reposition glyphs which are essential for correct display in complex scripts.

Limitation: if `useOTL` is set, but not for Latin script (e.g. = 0x02), and the text string contains more than one script, then kerning will not be applied to the Latin script text e.g. [Cyrillic text][Latin text][Cyrillic text]. This is because mPDF uses the presence of any repositioning applied to determine if kerning has been applied, otherwise using the alternative kern tables.

Line-height

The line-height of text can be controlled using the CSS property `line-height` (since mPDF 4). This used a fixed value for the line-height when `normal` was specified (the default CSS value) - set by the configurable variable `normalLineheight`. It also used a fixed proportion to determine the position of the text baseline (non-configurable).

From mPDF 6, you can (optionally) use font metrics derived from each font file to:

- Determine the height of a line when line-height is set to 'normal'
- Determine the glyph baseline (previously a fixed value)

Options are set by configurable variables in the `config.php` file:

Default settings in mPDF versions 6 - recommended especially for complex scripts with marks used above or below characters:

```
$this->useFixedNormalLineHeight = false;
$this->useFixedTextBaseline = false;
$this->adjustFontDescLineheight = 1.14;
```

Settings to be backwards compatible with mPDF versions < 6:

```
$this->useFixedNormalLineHeight = true;
$this->useFixedTextBaseline = true;
$this->normalLineheight = 1.33;
```

Using the font metrics will give approximately the same result as the fixed value for many standard Latin script fonts e.g. DejaVu Sans Condensed:

However, for some fonts the normal line-height using font metrics will be significantly taller, to account for the design of the font glyphs e.g. Khmer font:

For more information on how complex normal lineheights are, see Eric Meyers' website:
<http://meyerweb.com/eric/thoughts/2008/05/06/line-height-abnormal/> and <http://typophile.com/node/13081>

CSS control of line-height

From mPDF v 6.0 there are new controls for line-height using draft CSS3 properties. These can be set on all block level elements (P, DIV etc) and tables (TABLE/TD/TH).

line-stacking-strategy = `inline-line-height` | `block-line-height` | `max-height` | `grid-height`

- `inline-line-height` - [default] lineheight is initially calculated from the block-level font[-size]; the height is expanded by any inline content, including the calculated lineheight of that inline content;
- `block-line-height` - lineheight is fixed as the lineheight of the block-level font[-size];
- `max-height` - lineheight is initially calculated from the block-level font; the height is expanded by any inline content, EXCLUDING the calculated lineheight of that inline content;
- `grid-height` - lineheight is initially calculated from the block-level font; the height is expanded - AS MULTIPLES OF INITIAL LINEHEIGHT - by any inline content, EXCLUDING the calculated lineheight of that inline content;

Note: XSL has a similar property with the same name, which uses different but equivalent values: `line-height` instead of `inline-line-height`, `font-height` instead of `block-line-height`. It also uses `max-height`. The value `grid-height` is new to the CSS3 property.

line-stacking-shift = `consider-shifts` | `disregard-shifts`

This property determines whether to include or disregard the adjusted top- and bottom-edge of any characters that have a baseline-shift (e.g. superscript) when calculating lineheight.

Note: XSL has a similar property with a different name: `line-height-shift-adjustment` which uses the same values.

For more details see the [CSS3 draft specification](#).

Note for Advanced users

There are actually three possible metrics that can be used in a TrueType font file. The differences are summed up quite well in this article at <http://typophile.com/node/13081>. mPDF will by default use the usWinAscent and usWinDescent values to determine a 'normal' line-height, with two variations:

- if either the usWinAscent or usWinDescent are greater than the font bounding box (yMin yMax), then the values are reduced to equal the yMin/yMax values. NB this works as a fix with Myanmar Text (Windows 8 version) to give a line-height normal that is equivalent to that produced in browsers.
- if the USE_TYPO_METRICS bit is set on fsSelection (OS/2 table), this is telling the font to use the sTypo values and not the usWinAscent values. NB this works as a fix with Cambria Math font to give a normal line-height; at present, this is the only font I have found with this bit set; although note that MS WordPad and Windows FireFox browser use the big line-height from usWinAscent, whilst MS Word 2007 observes the fSelection value.

You can change the font metrics used by mPDF, by editing the defined constant (`_FONT_DESCRIPTOR`) at the top of the `mpdf.php` file:

- 'winTypo' uses sTypoAscender etc from the OS/2 table and is the one officially recommended - BUT
- 'win' use usWinAscent etc from OS/2 and in practice seems to be used most commonly in Windows environment; this is the default in mPDF;
- 'mac' uses Ascender etc from hhea table, and may be used to give results consistent with a Mac/OSX environment.

Finally, you can override values for Ascent, Descent and Leading for any specific font, by setting values in `config_font.php` e.g.

```
"cambriamath" => array(
    'R' => "cambria.ttf",
    'useOTL' => 0xFF,
    'TTCfontID' => array( 'R' => 2, ),
    'Ascent' => 950,
    'Descent' => -222,
    'Leading' => 0,
),
```

Note - The same values are used for all styles of the font-family. Descent values should be negative. All values should be given using a 1000 units per em scale, regardless of the UnitsPerEm used in the font design.

Notes

Remember that line-height for a TABLE has a default value (1.2) set in the `config.php` `defaultCSS`. This is left in for backwards compatibility. You can change this value to 'normal' for results consistent with most browsers.

Line-height in a `<textarea>` is fixed and defined in `classes/mpdfform.php` (= 1.2)

Details of the font metrics can be seen by inspecting the temporary font files e.g.
`/ttfondata/[fontname].mtx.php`.

Line breaking

The algorithm for determining automatic line breaks was completely rewritten in mPDF v6.0 - the following notes apply to mPDF >= 6.0

Line breaks will be allowed at:

- Spaces U+0020
- Word break U+200B
- Hyphen-minus U+002D when CSS hyphens set to "manual" or "auto", except when in a URL, or when following character is a > or a numeral
- Hard hyphen U+2010 when CSS hyphens set to "manual" or "auto"
- Soft hyphen U+00AD "" when CSS hyphens set to "manual" or "auto"
- [Automatic hyphenation](#) when CSS hyphens set to "auto"
- Between CJK characters, except CJK numerals, before "CJK-following" or after "CJK-leading" characters

Lao, Thai and Khmer text does not have space between words. By default, mPDF uses word dictionaries to determine appropriate opportunities for line-breaks. Users may turn this function off using the configurable variable `useDictionaryLBR`.

Alternatively users can insert the character U+200B (zero-width space) in the text to mark line-breaking opportunities manually.

Similarly for Tibetan script, mPDF 6 uses a simple algorithm to identify line-breaking opportunities after the characters U+0F0B (Tsheg) or U+0F0D. This can be overriden using the configurable variable `useTibetanLBR`.

Lists

(mPDF >= 6.0)

Lists are handled as for other block level elements, so you can apply any CSS properties usable on blocks (e.g. border, background, padding) to UL/OL and LI tags.

There is full support for CSS properties "list-style", "list-style-image", "list-style-type", and "list-style-position".

List Modes

There are two modes for lists: "mpdf" mode and "browser" mode. Mode is set using the configurable variable `list_auto_mode` in `config.php`

1) Browser mode gives the same display as most browsers. In this mode, the default list indentation is set by `padding "0 auto"` in the default CSS in `config.php`. "auto" equates to the value of `list_indent_default` in `config.php` - this is a "magic" value for padding, which is applied to either left or right depending on directionality of the list (rtl/ltr).

2) "mPDF" mode gives results compatible with versions of mPDF prior to v6.0. In this mode, the indentation is calculated differently: the outside edge of the list item is considered to be the outside edge of the bullet or number. For numbered lists, mPDF calculates the width of the largest number and this width is used to set the outside edge. The default list indentation of "auto" in mPDF mode is set by `list_indent_default_mpdf`. This value is added to the automatic calculated indentation. For backwards compatibility, `list_indent_first_level = 0;` can be used to prevent any indentation of the first list level.

The automatic indentation only works for bullets or numbered lists, and is ignored if "list-style-position: inside" is set, or images are used for markers.

Browser mode is set as the default - for backwards compatibility, change this to "mpdf".

List top & bottom margins

The default in browsers is to add a top and bottom margin to the outermost list only. This can be defined using CSS as:

```
ul, ol { margin-top: 0.83em; margin-bottom: 0.83em; }
ul ul, ul ol, ol ul, ol ol { margin-top: 0; margin-bottom: 0; }
```

This style is included in file `mpdf.css`

Versions of mPDF prior to v6.0 always added a top and bottom margin to the outermost list, (but no variation from this was possible). mPDF 6 is therefore backwards compatible re. the margins.

[NB The CSS styles are included in `mpdf.css`, because the `defaultCSS` values set in `config.php` only works on basic elements, and cannot use selectors such as "ol ol".]

Other configurable variables

Configurable variables are used to define size and offset for list bullets (i.e. disc, circle or square). The values can be any valid CSS size.

To specify a fixed bullet size and offset to give a similar appearance to most browsers, the default is set as:

```
$this->list_marker_offset = '5.5pt';
$this->list_symbol_size = '3.6pt';
```

To specify size and offset proportional to the list item's font size (compatible with versions of mPDF prior to v6.0), use:

```
$this->list_marker_offset = '0.45em';
$this->list_symbol_size = '0.31em';
```

Notes on Lists

The attribute `type` is case sensitive (whereas it is case insensitive in CSS). This allows the use of shorthand versions e.g. `type="A"` for uppercase alpha-numeric.

"list-style-type" is only inherited to child LI (not to child UL/OL); `list-style-image` and `-position` are fully inherited.

Lists in tables are basic, as block-level elements are not supported inside tables.

Properties like `text-align:justify` are inherited from surrounding elements, which will change the appearance of lists designed with earlier versions of mPDF.

The attribute `start="3"` (integer) works for "OL"; it is an official (though depracated) HTML attribute.

List bullets (`type = disc, circle or square`) are drawn rather than using font glyphs (from mPDF >= v6.0), for better consistency.

For maximum backwards compatibility with older versions of mPDF, change the following configurable variables in the `config.php` file:

mPDF 6.0 Default (Browser compatible)	Backwards Compatible <6.0
Lists \$this->list_auto_mode = 'browser'; \$this->list_marker_offset = '5.5pt'; \$this->list_symbol_size = '3.6pt';	\$this->list_auto_mode = 'mpdf'; \$this->list_marker_offset = '0.45em'; \$this->list_symbol_size = '0.31em';

Text Justification

Justifying text & Spacing

Character and Word spacing

When text-alignment is set to Justify (i.e. aligning to both right and left margins), word spacing and character spacing are normally used to justify text.

The default is to use a mix of character and word spacing. The ratio is set by variables defined at the top of the mpdf.php file:

```
var $jSWord = 0.4; // Percentage(/100) of spacing to allocate to Word vs. Character
var $jSmaxChar = 2; // Maximum spacing to allocate to character spacing. (0 = no maximum)
```

The maximum (`$jSmaxChar`) will prevent excessive, ugly and difficult-to-read character spacing.

CJK characters usually need to be character-spaced (as each character is like a word), and cursive scripts like Arabic or Indic texts need to be word-spaced. You can override some of the default settings also by using CSS letter-spacing:

```
div.arabic {
    text-align: justify;
    letter-spacing: 0;
}
```

Kashida in Arabic text

In Arabic text, justification can be achieved by elongating the line between characters - this is known as "kashida". Kashida can be used by configuring the font in config_fonts.php cf. [OpenType layout features \(OTL\)](#)

Non-breaking space (NBSP)

The non-breaking space character is supported when using all codepages other than CJK - unless the font chosen does not have a glyph/character representing the nbsp (? only FreeSerif); in these cases the nbsp is converted to a normal space. When supported, it prevents word wrapping and prevents collapse of white space as expected with HTML.

The last line

The last line of justified text is left-aligned, but when the penultimate line has been stretched to fit the page width it appears best if the last line also has increased spacing.

If the penultimate line was stretched considerably, the last line may look odd using the same spacing, so maximum values for the character-spacing (`$jSmaxCharLast`) and word-spacing (`$jSmaxWordLast`) can be set in the configuration file.

Justification before line-breaks

In a justified text block, an inline ``, `<textarea>`, `<input>`, or `<select>` element which causes a new line, force the previous line to be justified. However, a `<hr>` or `
` do not force justification of the preceding text.

This behaviour matches most browsers. To allow optional compliance with the behaviour of MS Word, which also justifies text before a `
` you can set the configurable variable `justifyB4br` to true.

Annotations

See

- [Annotation\(\)](#) - Insert an annotation in the document
- [<annotation>](#) - Custom HTML tag - equivalent to **Annotation**
- [\\$title2annots](#) - Convert the title attribute from all in-line elements to annotations

DRAFT

Barcodes

(mPDF >=4.0)

A variety of barcodes can be produced with mPDF. Please see the example file for more information.

Specifications in mPDF

Barcode type	code	Light margin / Quiet zone						Pr ratio
		X-dim	H-dim	Left	Right	Top/bottom	DAFT ^[5]	
EAN-13	EAN13, ISBN, ISSN	0.33	25.93	11X	7X	0		
UPC-A	UPCA	0.33	25.91	9X	9X	0		
UPC-E	UPCE	0.33	25.93	9X	7X	0		
EAN-8	EAN8	0.33	21.64	7X	7X	0		
Intelligent Mail barcode	IMB	0.508	3.68	3.175	3.175	0.711	2:2:3:1	^[4]
Royal Mail 4-state Customer barcode	RM4SCC	0.508	5.0	2	2	2	5:5:8:2	^[4]
Royal Mail 4-state Customer barcode (Dutch)	KIX	0.508	5.0	2	2	2	5:5:8:2	^[4]
POSTNET	POSTNET	0.508	3.175	3.175	3.175	1.016	5:2	
PLANET	PLANET	0.508	3.175	3.175	3.175	1.016	5:2	
Code 128	C128A, C128B, C128C	0.381	10	10X	10X	0		-
UCC/EAN-128 (GS1-128)	EAN128A, EAN128B, EAN128C	0.381	10	10X	10X	0		-
Code 39 (Code 3 of 9)	C39, C39+, C39E, C39E+	0.381	10	10X	10X	0		2.5 ^[1]
Standard 2 of 5	S25, S25+	0.381	10	10X	10X	0		3 ^[2]
Interleaved 2 of 5	I25, I25+, I25B, I25B+	0.381	10	10X	10X	0		2.5 ^[1]
Code 93	C93	0.381	10	10X	10X	0		-
MSI (Modified Plessey)	MSI, MSI+	0.381	10	12X	12X	0		-
CODABAR	CODABAR	0.381	10	10X	10X	0		2.5 ^[1]
Code 11	CODE11	0.381	10	10X	10X	0		3 ^[3]

All values in millimeters, unless specified as a factor of X-dim e.g. 10X.

X = width of narrowest bar (also known as module width)

Pr ratio is ratio of narrow bar to wide bar

^[1] Code specification: 1:2 - 1:3 (>2.2 if X<0.5mm)

^[2] Code specification: 1:3 - 1:4.5

^[3] Code specification: 2.24 - 3.5

^[4] Bars per inch (determines bar/gap ratio, gap width) = 22

^[5] Ratio of bar heights: Descender, Ascender, Full, Tracker (or Full:Half bar)

Bookmarks

Bookmarks

PDF document Bookmarks can be set in two ways:

```
<bookmark content="Buffalo" level="0" /> or  
$mpdf->Bookmark("Buffalo",0);
```

Bookmark(string content[, integer level])

- content
Text string (utf-8 encoded)
- level
Specify the level of this entry i.e. like heading1,2,3; but starts at level 0
Default = 0

NB Bookmarks use system fonts, therefore any Unicode text can be used, even if a unibyte codepage is being used for the document.

See - [bookmarkStyles](#)

DRAFT

Columns

Columns

Multiple columns can be used on a page. This can started and stopped anywhere in the document, by either:

```
$mPDF->SetColumns(nCols[, vAlign [, gap ]]); or  
<columns column-count="n" vAlign="justify" column-gap="n" />
```

nCols = Number of columns. Anything less than 2 sets columns off. (Required)

vAlign = Automatically adjusts height of columns to be equal if set to J or justify. Default Off. (Optional)

gap = gap in mm between columns. Default 5. (Optional)

<columnbreak /> <column_break /> or <newcolumn /> (synonymous) can be included to force a new column.
(This will automatically disable any justification or readjustment of column heights.)

The maximum ratio to adjust column height when justifying is defined at the top of the mpdf.php file - too large a value can give ugly results:

```
var $max_colH_correction = 1.15; (default value)
```

You can keep columns as they are i.e. 1st column will finish page then start on second, by setting

```
$mpdf->KeepColumns = true;
```

Limitations

Columns are incompatible with (and automatically disable): justification if a table is included, table rotation, collapsible margins for blocks e.g. top and bottom margins for a DIV will not collapse (default) at the top/bottom of a column. Support for block-level borders (DIV, P etc) was added in mPDF 3.0

Note: Block element borders (P,DIV etc) in columns may appear as disjointed lines. This is because they are written to the document line by line to allow repositioning when the columns are adjusted. The borders should appear correctly when "zoomed in" and when printing. Table cell borders may be similarly affected. Also note that the horizontal borders between table cells will only appear half width if a background color is used in the cell below, because the cell background may write over the bottom half of the border.

Note: To create columns that span more than one page, use [Floating blocks](#).

Forms

mPDF can generate a static view of HTML forms and their elements, or (from mPDF >=5.3) can produce an "Active Form".

"Active" Forms

Active forms can be generated which can either be printed, or the data submitted to a URI. The variable `$this->useActiveForms` should be set to **TRUE** either at run-time or in `config.php`

Compatibility & Limitations

Active forms have been tested with Adobe Reader 10, and Foxit 5.0 Some limitations and compatibility issues seem to be determined by the PDF reader, rather than by mPDF. This is especially true of the way the form fields are displayed, handling of non-ASCII text, and encryption.

Encrypted Active Forms do not work with Adobe Reader 10, but work fine in Foxit 5.0; I believe this to be due to the limited support of Adobe Reader for forms using the (old) PDF 1.4 specification i.e. mPDF.

Brief testing with the in-built Google Chrome PDF Reader also shows some limited functionality e.g. Submit doesn't work

Known limitations include:

- cannot save a completed form
- cannot save or export FDF data locally
- cannot "sign" PDF forms
- incompatible with rotated tables, HTML headers, and "keep-with-table" (is compatible with page-break:avoid - but not if rotated - and columns)
- cannot use SIP/SMP fonts for active form elements (ones which are subset as SIP/SMP)
- button images cannot be vector images (SVG or WMF)
- (active) radio buttons do not work inside a DIV fixed/absolute position (or with page-break-inside: avoid)

Creating a valid Active Form

A PDF document can only contain one active form e.g. submit will work on all fields in the document.

The method (GET/POST) and action (URI) are set when a `<form ...>` element is parsed, and remain active unless reset by another `<form>` element. All fields will be submitted as though from one form, whether they are enclosed within a `<form>` element or not.

All fields should have names.

Field names must only contain letters, numbers, colon(:), underscore(_) or hyphen(-). This is largely as per HTML spec, but cannot contain a period(.) as this is part of PDF spec. for name hierarchies.

Field names should usually be unique, except for radio buttons.

Duplicate field names can be used (e.g. to echo the text to a field elsewhere in the document), but fields with the same name must be of the same type, and have the same default value set.

Value(s) for radio buttons and checkboxes are required, and can only contain letters, numbers, colon(:), underscore(_), hyphen(-) or period(.)

Values in all other form fields can contain any unicode character (although obviously only win-1252 codepage if you are using core fonts only for the document). HTML entities e.g. ߾ are recommended.

See [HTML attributes](#) for details of attributes which can be set e.g. disabled, required etc.

See [supported CSS](#) for style properties which can be applied.

Exporting (submitting) data

Data from forms can be submitted to a URI in either HTML or XFDF format (cf. `config.php`). XFDF is a form of XML and is recommended, because of encoding issues. See `formsubmit.php` in the example folder for ideas on how to handle the submitted data.

NB A submitted radio button field name is doubled with an underscore i.e. "myButtonName_myButtonName"

If the export format is XFDF, the submitted data is always UTF-8 encoded.

If the export format is HTML, it is much more complicated. From a "core-fonts" only document, the submitted data uses PDFDocEncoding. (See `formsubmit.php` in the example folder for a conversion script.) But if the form contains any characters which are not in the PDFDocEncoding (similar to Windows-1252), Adobe Reader will decide which encoding to use(!?)

It is therefore recommend that you either use `mPDF('c')` and decode html POST from PDFDocEncoding, or use XFDF.

The default HTML submit method is POST; GET only seems to work from a PDF document opened in a standalone Reader (not in the browser).

You can specify a mailto address as a URI e.g. `action="mailto:email@address"` but you may find that it is blocked by the user's computer if using the HTML format.

Radio buttons

Disabled: if one radio button is set as disabled, mPDF will disable the whole group. PDF readers seem to handle this situation differently i.e. Adobe Reader 10 still allows selection of the disabled button, whilst Foxit disables the whole group.

Javascript

Javascript can be set for buttons using `onClick=""` but note this uses "Acrobat" Javascript. (You can download the Acrobat Javascript reference manual from the Adobe Developer's site).

For select, text and textarea you can use `onChange=""` which is triggered after the value has been changed.

Note: From mPDF >= 5.4, `<textarea>` and `<input type="text">` will accept javascript as: `onKeystroke`, `onValidate`, `onCalculate` and `onFormat`.
`onChange` is deprecated but works as `onCalculate` (for `<textarea>` and `<input>`).
Select still accepts `onChange`.

Unicode characters in JavaScript must be written by typing a backslash, a lowercase "u", and then the four digit hexadecimal number corresponding to the character's encoding in the utf-16 character set e.g. \u2042

Appearance of form fields

Adobe Reader 10 largely ignores any control one tries to place on the appearance of some form fields, and does its own thing. In general, the `font-size` set for the form field will determine its size, and for `text/textarea` and `select` fields, `color` will determine the font colour used. CSS values for `border-color` or `background-color` will work for (non-image) buttons, textarea and text fields. Other things like the border style and width can be altered by configurable variables in `config.php` but the level of control is disappointing.

Radio buttons and check-boxes use Adobe Reader's own icons, but Foxit uses information provided by the PDF file. The variable `$this->formUseZapD` determines whether ZapfDingbat symbols are used, or mPDF's appearance streams designed to mimic Adobe Reader's appearance.

Some components of interactive forms may be output in RGB colorspace even if you have specified `restrictColorSpace`. Since restricted colorSpace is mainly used for PDFA/PDFX files - which cannot contain active form fields anyway - this shouldn't matter.

Index

mPDF can generate an index at the end of document using:

- <indexentry content="Buffalo" /> to make index entries at the appropriate place in the HTML text
- <indexinsert ... /> generates and inserts the Index at the end of document

Note: Indexes have been completely rewritten in mPDF 6. The notes below refer to mPDF v6.0+

- <indexentry> or IndexEntry() should be used to create Index entries during document writing.
- <indexinsert> or InsertIndex() should be used to generate the Index at the end of the document.

When an Index is inserted in the PDF document using <indexinsert> or InsertIndex(), the Index is generated (internally) as HTML code in the following format (with the markup as shown):

```
<div class="mpdf_index_main">
<div class="mpdf_index_letter">A</div>
<div class="mpdf_index_entry">Aardvark<a class="mpdf_index_link" href="#page37">37</a>
</div>
...
</div>
```

CSS stylesheets can thus be used to control the layout of the Index e.g.:

```
/* For Index */
div.mpdf_index_main {
    line-height: normal;
    font-family: sans-serif;
}
div.mpdf_index_letter {
    line-height: normal;
    font-family: sans-serif;
    font-size: 1.8em;
    font-weight: bold;
    text-transform: uppercase;
    page-break-after: avoid;
    margin-top: 0.3em;
    margin-collapse: collapse;
}
div.mpdf_index_entry {
    line-height: normal;
    font-family: sans-serif;
    text-indent: -1.5em;
}
.a.mpdf_index_link {
    color: #000000;
    text-decoration: none;
}
```

A default stylesheet for Indexes is included in `mpdf.css`

Index Collation

In order to generate an Index with non-ASCII characters, entries need to be sorted accordingly (collation), and non-ASCII characters should map to the appropriate Dividing letter e.g.:

A

Alonso, Fernando

Álvarez, Isaac

Arroyo Molino, David

B

Benítez, Carlos

Entries in an Index can now be sorted using any of the Locale values available on your system. Set it using the "collation" property/parameter e.g.:

```
<indexinsert usedivletters="on" links="off" collation="es_ES.utf8" collation-group="Spanish_Spain"
```

```
/>
```

- or -

```
$mpdf->InsertIndex(true, false, "es_ES.utf8", "Spanish_Spain");
```

NB You should always choose a UTF-8 collation, even when you are using Core fonts or e.g. charset-in=win-1252, because mPDF handles all text internally as UTF-8 encoded.

You can see which Locales are available on your (Unix) system: <?php system('locale -a') ?>

Note: Index collation will probably not work on Windows servers because of the problems setting Locales under Windows.

If you have set your index to use Dividing letters, you can also determine how letters are grouped under a dividing letter. In the example index above, we want Ä to be grouped under the letter a/A. Set the "collation-group" using:

```
<indexinsert usedivletters="on" links="off" collation="es_ES.utf8" collation-group="Spanish_Spain"
/>
```

- or -

```
$mpdf->InsertIndex(true, false, "es_ES.utf8", "Spanish_Spain");
```

Values should be selected from the available file names in folder /collations/.

Note: This will not affect the overall order of entries, which is determined by the value of "collation".

Note: The groupings do not always match the order set by locale. This is because the data for collations has come from different sources. The files in /collations/ can be edited.

The array consists of [index]: unicode decimal value of character => unicode decimal value of character to group under: e.g. Ä [A tilde] (U+00C3) (decimal 195) => a (U+0061) (decimal 97). The target character should always be the lowercase form.

Non-ASCII chrcarters in Index entries

Note: htmlspecials_encode should be used to encode the text of content in <indexentry> - although not when using IndexEntry().

Columns

From mPDF 6.0, columns are not specified as part of the <indexinsert>, so a typical 2-column index might be produced by:

```
<pagebreak type="next-odd" />
<h2>Index</h2>
<columns column-count="2" column-gap="5" />
<indexinsert usedivletters="on" links="on" collation="en_US.utf8"
collationgroup="English_United_States" />
<columns column-count="1" />
```

Index Sub-entries

Index entries can contain sub-entries, separated by colons e.g.

```
<indexentry content="Mammals:elephants" />
```

A shorthand way of displaying subentries is set by default, which suppresses the main entry if > 1 subEntry. It can be disabled/enabled using the configurable variable \$this->indexUseSubentries in config.php.

This is the default appearance, with \$this->indexUseSubentries = false; -

```
Mammals 73
- elephants 142
- humans 173
Marsupials
- kangaroos 75
```

```
- wombats 86
```

Index entries can also include simple mark-up tags and/or more than one colon e.g:

```
<indexentry content="Mammals:&lt;b&gt;elephants&lt;/b&gt;:: breeding" />
```

which appears as:

```
Mammals  
- elephants: breeding 15
```

This is the appearance with \$this->indexUseSubentries = false;

```
Mammals 73  
Mammals, elephants 142  
Mammals, elephants: breeding 15  
Mammals, humans 173  
Marsupials, kangaroos 75  
Marsupials, wombats 86
```

Customised appearance (advanced)

Several variables set at beginning of function InsertIndex() in mpdf.php which could be changed to alter appearance of Index. e.g. spacer, and joiner characters.

Layers

(mPDF >= 5.7)

CSS "z-index" can be used to utilise layers in the PDF document.

CSS can set the z-index for any block element or image (default = 0). This does not work on block elements with fixed or absolute position.

Set the Initial state for each layer

You can set initial state="hidden" for a specific z-index (z), and/or specify a display name for the Layer e.g.

```
$mpdf->layerDetails[z]['state']='hidden'; // Set initial state of layer - "hidden" or nothing
```

```
$mpdf->layerDetails[z]['name']='Correct Answers';
```

- where z= the z-index (set by CSS)

Note:

- Using layers automatically changes the resulting PDF document to PDF 1.5 version (which is incompatible with PDFA and PDFX in mPDF).
- You cannot nest layers - inner values will be ignored

Display the Layers pane in PDF document viewer

`$mpdf->open_layer_pane` (set by default as `$this->open_layer_pane=false` in `config.php`) can be set to open the layers pane in the browser when the document is opened.

```
$mpdf->open_layer_pane = true;
```

Set Programmatically

If you are writing the PDF document using functions other than `WriteHTML()`, you can set the layers as follows:

```
$mpdf->BeginLayer($z-index);
...
$mpdf->EndLayer();
```

Reserved Layer Names

mPDF automatically adds layer names for visibility: "Print only", "Screen only", and "Hidden"; these only show when utilised.

Table of Contents

You can insert one or more tables of contents in the document using HTML or PHP code - see:

[TOCpagebreak\(\)](#) or <tocpagebreak> will insert a Table of Contents (ToC) at the current position.

[TOC_Entry\(\)](#) or <tocentry> can be used to mark entries for the ToC .

From mPDF >= 5.7, CSS styles can be used to control layout of the ToC.

At the end of the document, the ToC is generated, creating HTML code (internally) which looks like this:

```
<div class="mpdf_toc" id="mpdf_toc_0">
    <div class="mpdf_toc_level_0">
        <a class="mpdf_toc_a" href="#__mpdfinternallink_1"><span class="mpdf_toc_t_level_0">Section 1</span></a>
        <dottab outdent="2em" />
        <a class="mpdf_toc_a" href="#__mpdfinternallink_1"><span class="mpdf_toc_p_level_0">5</span></a>
    </div>
    <div class="mpdf_toc_level_1">
        <a class="mpdf_toc_a" href="#__mpdfinternallink_2"><span class="mpdf_toc_t_level_1">Chapter 1</span></a>
        <dottab outdent="2em" />
        <a class="mpdf_toc_a" href="#__mpdfinternallink_2"><span class="mpdf_toc_p_level_1">5</span></a>
    </div>
    <div class="mpdf_toc_level_2">
        <a class="mpdf_toc_a" href="#__mpdfinternallink_3"><span class="mpdf_toc_t_level_2">Topic 1</span></a>
        <dottab outdent="2em" />
        <a class="mpdf_toc_a" href="#__mpdfinternallink_3"><span class="mpdf_toc_p_level_2">5</span></a>
    </div>
</div>
```

The id is "0" for root/un-named ToC; otherwise it is lowercase of the name="" used for the ToC

An example CSS stylesheet for this:

Note: From mPDF v 6.0 this example stylesheet for ToCs was added to the file `mpdf.css` which is read by default as a secondary default CSS stylesheet

```
div.mpdf_toc {font-family: sans-serif; font-size: 11pt;}
.a.mpdf_toc_a {text-decoration: none; color: black;}
/* Whole line level 0 */
div.mpdf_toc_level_0 {line-height: 1.5; margin-left: 0; padding-right: 2em;}
/* padding-right should match e.g <dottab outdent="2em" /> 0 is default */
/* Title level 0 - may be inside <a> */
span.mpdf_toc_t_level_0 {font-weight: bold;}
/* Page no. level 0 - may be inside <a> */
span.mpdf_toc_p_level_0 {}
/* Whole line level 1 */
div.mpdf_toc_level_1 {margin-left: 2em; text-indent: -2em; padding-right: 2em;}
/* padding-right should match <dottab outdent="2em" /> 2em is default */
/* Title level 1 */
span.mpdf_toc_t_level_1 {font-style: italic; font-weight: bold;}
/* Page no. level 1 - may be inside <a> */
span.mpdf_toc_p_level_1 {}
/* Whole line level 2 */
div.mpdf_toc_level_2 {margin-left: 4em; text-indent: -2em; padding-right: 2em;}
/* padding-right should match <dottab outdent="2em" /> 2em is default */
/* Title level 2 */
span.mpdf_toc_t_level_2 {}
/* Page no. level 2 - may be inside <a> */
span.mpdf_toc_p_level_2 {}
```

For additional ToCs, these will have <toc name=""> attribute. Use the lowercase e.g. for name="Figures"

```
#mpdf_toc_figures {font-family:Calibri; font-size: 10pt;}
```

```
#mpdf_toc_figures span.mpdf_toc_t_level_0 {color: red; }
```

Note: If you have 2 ToCs immediately following each other, and wish to use pagenumstyle or suppress to control the following text, then you need to set those values on both of the <tocpagebreak> elements.

Automatically Generated ToC entries

You can automatically generate ToC entries from h1 - h6 tags, by setting the variable `h2toc`.

Only the default ToC will be used if more than 1 ToCs are defined for the document.

H1 - H6 must be written with uppercase when defining the array.

Example:

```
$mpdf->h2toc = array('H1'=>0, 'H2'=>1, 'H3'=>2);
```

NB This will ignores calls from inside ToC e.g. if <tocpagebreak toc-prehtml="<h3>Table of Contents</h3>" and H3 is set to auto-generate a ToC entry - these will be ignored.

Watermarks

You can add a watermark to the page(s) either text (e.g. DRAFT) and/or a background image.

Set the following before writing the HTML code. NB The watermark is added when writing the footer at the end of a page, so it can be set anytime before ending the page/document.

```
$mpdf->SetWatermarkText('DRAFT'); // Will cope with UTF-8 encoded text
```

```
$mpdf->watermark_font = 'DejaVuSansCondensed'; // Uses default font if left blank
```

You can alter the transparency values (default = 0.2) using

```
$mpdf->watermarkTextAlpha = 0.1;  
$mpdf->watermarkImageAlpha = 0.5;
```

A watermark image is set by default to print on top of the page contents. The opacity setting will alter the appearance of the text behind the image. You can optionally set the watermark to appear behind the page contents using `watermarkImgBehind`, but note that the image will be hidden by any background colour specified, including table cells and the page background.

Note: In version 4.4 `watermarkImgBehind` was unintentionally set to **TRUE** in the config.php file

Set the watermark(s) to show using: [showWatermarkText](#) or [showWatermarkImage](#)

Note: From mPDF >=3.0 you can alternatively use the CSS style for background-image on the `<body>` tag to create a sort of watermark, although this does not support opacity. The difference is that text, tables etc are written over the top of a background-image; a watermark is actually printed over the top of everything else, but is semi-transparent.

See

- [SetWatermarkText\(\)](#) - Set the text to use as a Watermark
- [<watermarktext>](#) - HTML equivalent to SetWatermarkText()
- [SetWatermarkImage\(\)](#) - Set the image to use as a Watermark
- [<watermarkimage>](#) - HTML equivalent to SetWatermarkImage()
- [watermarkImageAlpha](#) - Specifies the transparency (alpha value) for the watermark image
- [watermarkTextAlpha](#) - Specifies the transparency (alpha value) for the watermark text
- [showWatermarkText](#) - Specifies whether or not to show/print the watermark text
- [showWatermarkImage](#) - Specifies whether or not to show/print the watermark image
- [watermark_font](#) - Specifies the font to use for Watermark text
- [watermarkImgBehind](#) - Specify whether to show the watermark image behind the page contents

Graphs

mPDF can generate graphs from table data. This requires the **JPGraph** program, which is an open-source PHP library available from <http://www.aditus.nu/jpgraph/>

Graphs have only been tested with the PHP5 version of JPGraph.

Installation

JPGraph needs to be installed on your server. Follow the instructions for installing JPGraph, including configuring the script to access the TTF fonts necessary to produce JGraphs.

Note: Check that you have a subfolder in your /mpdf/ folder named /graph_cache/ and make sure it is writeable.

Using Graphs in mPDF

See the example below. The table containing data must precede the graph. Nested tables will be ignored.

Example #1

```
<?php

include("../mpdf.php");

// This must be defined before including mpdf.php file
define("_JPGRAPH_PATH", '../..../jpgraph_5/src/');

// Change these if necessary to the name of font files you can access from JPGraph
define("_TTF_FONT_NORMAL", 'arial.ttf');
define("_TTF_FONT_BOLD", 'arialbd.ttf');

$mpdf=new mPDF();

// This must be set to allow mPDF to parse table data
$mpdf->useGraphs = true;

$html = '
<table id="tbl_1"><tbody>
<tr><td></td><td><b>Female</b></td><td><b>Male</b></td></tr>
<tr><td>35 - 44</td><td><b>4</b></td><td><b>2</b></td></tr>
<tr><td>45 - 54</td><td><b>5</b></td><td><b>7</b></td></tr>
<tr><td>55 - 64</td><td><b>21</b></td><td><b>18</b></td></tr>
<tr><td>65 - 74</td><td><b>11</b></td><td><b>14</b></td></tr>
<tr><td>75 - 84</td><td><b>10</b></td><td><b>10</b></td></tr>
<tr><td>85 - 94</td><td><b>2</b></td><td><b>1</b></td></tr>
<tr><td>95 - 104</td><td><b>1</b></td><td><b>1</b></td></tr>
<tr><td>TOTAL</td><td>54</td><td>52</td></tr>
</tbody></table>

<jpgraph table="tbl_1" type="bar" title="New subscriptions" label-y="% patients" label-x="Age group"
series="cols" data-row-end="-1" show-values="1" width="600" legend-overlap="1" hide-grid="1" hide-y-
axis="1" />
';

$mpdf->WriteHTML($html);
$mpdf->Output();
exit;

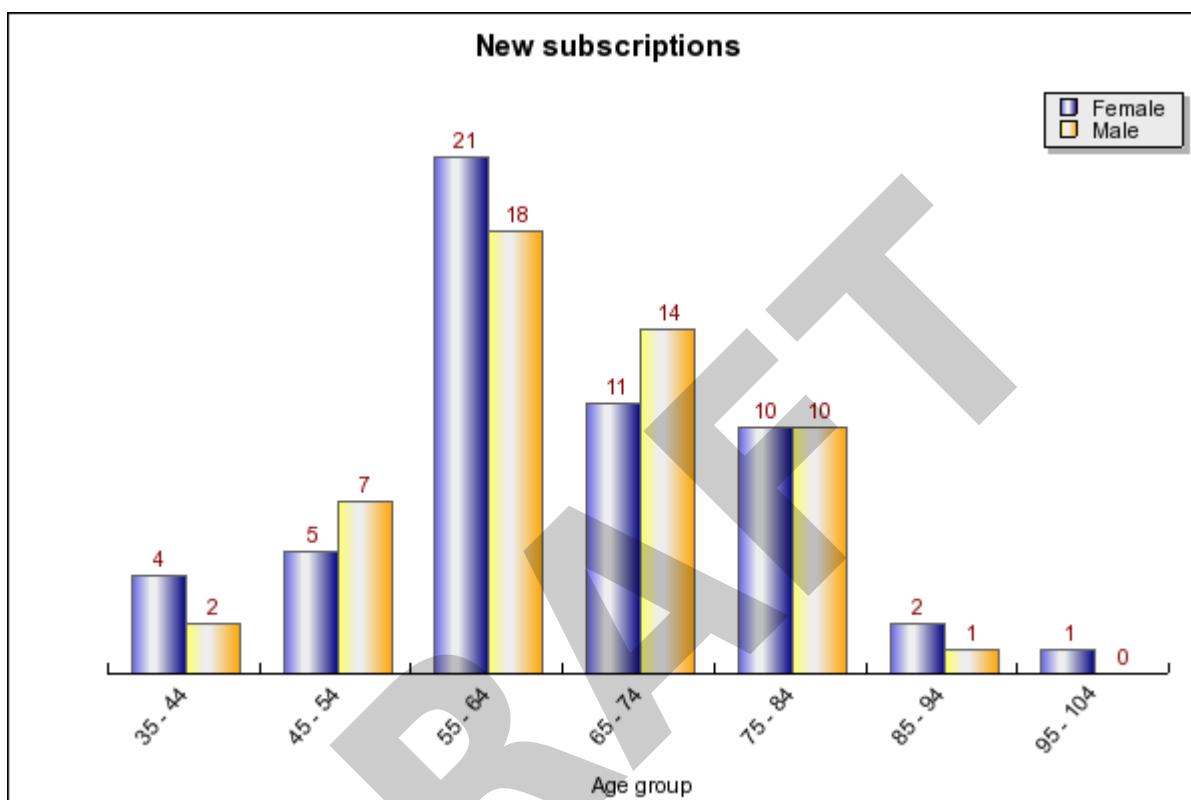
?>
```

Result

This should produce a PDF file like this:

	Female	Male
35 - 44	4	2

45 - 54	5	7
55 - 64	21	18
65 - 74	11	14
75 - 84	10	10
85 - 94	2	1
95 - 104	1	
TOTAL	54	52



Note that graphs can be rotated as with images using `<jpgraph rotate="90" ...` (see [HTML attributes](#))

Replaceable Aliases

Note: Prior to mPDF 6.0 you could include `{nb}` and `{nbpg}` anywhere in the text of the document, including headers/footers.

In mPDF v6.0+ these will only be replaced when used in headers/footers.

There are several placemarkers you can include, which will be replaced when the PDF file is output:

`{nb}`

Will be replaced by the total number of pages in the document. This disregards any changes you make to page numbering with [AddPage\(\)](#) or [<pagebreak>](#) and will always return the actual number of pages in the PDF document. You can change this alias (in case you wish to use the text "`{nb}`" in your document) using [AliasNbPages\(\)](#). The alias is case-sensitive.

`{nbpg}`

Will be replaced by the total number of pages in the document or page group. If you have reset the page numbering with [AddPage\(\)](#) or [<pagebreak>](#) the total number of pages in the current page group will be used (up to where the numbering is reset) rather than the total number of pages in the whole document. You can change this alias (in case you wish to use the text "`{nbpg}`" in your document) using [AliasNbPageGroups\(\)](#). The alias is case-sensitive. The numbering style can be changed using [AddPage\(\)](#) or [<pagebreak>](#).

`{PAGENO}`

Will be replaced by the Page number - but **only** in headers/footers. If you have reset the page numbering with [AddPage\(\)](#) or [<pagebreak>](#) the calculated document page number will be used rather than the physical PDF document page number. You cannot change this alias. The alias is case-sensitive. The numbering style can be changed using [AddPage\(\)](#) or [<pagebreak>](#).

`{DATE d/m/Y}`

Will be replaced by today's date - but **only** in headers/footers. The alias is case-sensitive. The date format can be defined using any of the values in the PHP function [date\(\)](#). There must be a space after `{DATE`

Example: `{DATE j-m-Y H:m}`

`{colsum} {colsumN}`

Place in cell inside a table footer i.e. `<tfoot><td>`. The total of values in the corresponding column will be output at the bottom of every page, and the end of the table. Default output is an integer. An optional integer **N** after `colsum` will specify a fixed number of decimal places. (mPDF >= 5.4)

`{iteration varname}`

Place in cell inside table header i.e. `<thead><td>`. You can use any alphanumeric string as a unique varname. (When processed by mPDF it will be replaced by `"_varname_"` to avoid collision with an mPDF internal variable. The length of the alphanumeric string will determine the length of the placeholder used to space the text i.e. if your counter will reach 1000 you should use a string of at least 4-5 characters in length.

See Also

- [pagenumPrefix](#) - Specify text to precede page numbers generated by `{PAGENO}`
- [pagenumSuffix](#) - Specify text to follow page numbers generated by `{PAGENO}`

- **nbpqPrefix** - Specify text to precede page total generated by {nbpq }
- **nbpqSuffix** - Specify text to follow page numbers generated by {nbpq }
- **Page numbering** -
- **aliasNbPg** - Specify the text to be replaced by the document page total
- **aliasNbPgGp** - Specify the text to be replaced by the group page total

CMYK colours

CMYK Colors

Functions - SetDrawColor(), SetTextColor() and SetFillColor() all take an optional 4th parameter.

If defined this will interpret the input as CMYK color i.e.

```
SetDrawColor(15,82,0,10)
```

NB all values are out of 100 - not 255 as for RGB

Importing files & Templates

(mPDF >= 2.3)

Using an extension of mPDF, pages from external PDF files can be imported into 'templates' and used throughout the current document. This can be used for:

- page templates e.g. statements or invoices
- letterheads (see also [Headers & Footers](#))
- a whole document template
- create thumbnails as handouts etc. from a document you have produced

Note: On limited testing, mPDF appears to import any PDF file (with a version <= 1.4, or one produced by mPDF) as long as it is not password protected. Text and images are imported, but links, bookmarks etc. are not. LZW encoding is supported from mPDF 4.3 onwards

Tip: mPDF imports all embedded fonts required for the document, even if they are the same as the ones used in the document being written. Try to keep the file size of the external source PDF file down to a minimum.

Note: Prior to mPDF 4.3, this required calling `mPDFI()`. The functions have now been incorporated into the main `mpdf.php` file, but you must use `SetImportUse()` to enable them.

See

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [ImportPage\(\)](#) - Import a page from an external PDF file
- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template

Overwriting existing files

(mPDF >= 2.3)

Using the class extension (mPDFI), an existing PDF file can be overwritten, replacing specified text with alternatives. For example you may have created a long complex PDF file, and you wish to produce copies with an individual number on each copy without having to re-generate the whole document each time.

Note: OverWrite() was written to work on PDF files produced by mPDF. It will work with encrypted files, as long as the same encryption properties are used for the new document. I have also used it successfully with a number of external PDF files, including those produced by Acrobat Distiller (but not if they are encrypted). It will not work with embedded font subsets.

See

- [OverWrite\(\)](#) - Replace specified text strings in an existing PDF file
- [mPDFI\(\)](#) - Class constructor for importing files and templates

Writing non-HTML text

mPDF is optimised to output HTML code, i.e. parsing the HTML markup. If you want to create a PDF file from a long text file (e.g. a PHP script file) preserving **TABs** and multiple spaces, use the function `preparePreText()`. Note this is not part of the mPDF class.

This will surround the text with `<pre>` tags whilst preventing any `<pre>` tags included in the text from being parsed. It also allows use of a text string marker (*formfeed*) to be replaced by a formfeed in the output file.

See

- [preparePreText\(\)](#)

PDF/A1-b compliance

(mPDF >= 4.3)

PDF/A1-b is a file format for the long-term archiving of electronic documents. It is based on the PDF Reference Version 1.4 from Adobe Systems Inc. and is defined by ISO 19005-1:2005

A key element to this reproducibility is the requirement for PDF/A documents to be 100% self-contained. All of the information necessary for displaying the document in the same manner every time is embedded in the file. This includes, but is not limited to, all content (text, raster images and vector graphics), fonts, and color information. A PDF/A document is not permitted to be reliant on information from external sources (e.g. font programs and hyperlinks).

Other key elements to PDF/A compatibility include:

- All fonts must be embedded and also must be legally embeddable for unlimited, universal rendering. This also applies to the Adobe standard fonts such as Times, Courier or Helvetica.
- Colorspaces specified in a device-independent manner.
- Encryption is not allowed.
- Use of standards-based metadata is mandated.
- PDF Version must be specified as 1.4

The standard specifies two levels of compliance for PDF files:

- PDF/A-1a - Level A compliance in Part 1
- PDF/A-1b - Level B compliance in Part 1

PDF/A-1b has the objective of ensuring reliable reproduction of the visual appearance of the document. PDF/A-1a includes all the requirements of PDF/A-1b and additionally requires that document structure be included (also known as being "tagged"), with the objective of ensuring that document content can be searched and repurposed.

A PDF/A document can be identified as such through PDF/A-specific metadata located in the "<http://www.aiim.org/pdfa/ns/id/>" namespace. However, claiming to be PDF/A and being so are not necessarily the same.

Important: mPDF is **not guaranteed** to produce fully PDF/A1-b compliant files in all circumstances. It is the users responsibility to check compliance if this is essential.

mPDF and PDF/A1-b compliance

You can make mPDF produce mPDF/A1-b compliant files by setting:

```
$this->PDFA = true; // Edit in config.php or set at runtime
```

Some changes were made in mPDF 4.3 which affect all files (PDF/A or not) to improve compliance with PDF-1.4 specification:

- file version changed to PDF-1.4*
- newline removed at end of file after %%EOF
- an /ID object is added to the PDF trailer (this is optional in PDF-1.4. spec.) when the file is not encrypted
- a binary file marker (consisting of a comment line with 4 characters > 127 ASCII) is added just after the first line (recommended n PDF-1.4 spec.)

* The PDF file version was changed to 1.5 in previous versions when 'active forms were introduced as an experiment'. These are no longer viable, and the rest of mPDF generated files meet 1.4 specification.

Colorspaces and ICC Profiles

PDF files handle colours internally in a number of different colorspaces. PDFA files must have one defined (and embedded) colour profile for one colorspace.

The 3 colour RGB colorspace is used most commonly in PDF/A documents, although you could select a 4-colour CMYK colorspace - but you cannot use both together, and SPOT colours are not permitted. RGB colorspace is the default in mPDF for a PDF/A1-b document.

It is likely that most content in a PDF file will be RGB unless you have specifically defined otherwise e.g. a CMYK JPG image file, or colours defined as CMYK() etc.

Colorspaces can be altered - see [restrictColorSpace](#).

An ICC Color profile must be embedded in the file.

The ICC Color profile will likely change the appearance of colours in your document. The ICC profile that is included with mPDF is an open licence color profile from the International Color Consortium (<http://www.color.org>) which seems to change colours a little as possible for an sRGB profile. Other sources of free ICC profiles are:

- <http://www.eci.org>
- <http://www.color.org/profiles2.xalter>
- http://www.adobe.com/support/downloads/iccprofiles/iccprofiles_win.html

You can change the color profile used by mPDF by adding a *.icc file to the [mpdf]/iccprofiles/ folder. Then edit config.php e.g.

```
$this->ICCPProfile = 'eciRGB_v2';
```

Fonts

All fonts must be embedded in a PDF/A file. mPDF normally uses the core Adobe fonts (Helvetica, Times, Courier, Symbol and Zapfdingbats) in a number of situations, and these font files are not available to embed. This affects:

- \$useCoreFontsOnly is not permitted
- lists cannot use Zapfdingbats font for bullets
- form element <select> cannot use Zapfdingbats character as the drop-down symbol
- ISBN/ISSN barcode cannot use Helvetica for the ISBN number (above the bar)

Also, the fonts GB, BIG5, UHC and SJIS which reference the free CJK font pack from Adobe cannot be used

Transparency and Annotations

Because PDF/A-1b has the objective of ensuring reliable reproduction of the visual appearance of the document, no objects may be transparent or semi-transparent. This affects:

- watermarks text or images
- images with the CSS property *opacity* set
- annotation markers

Making it happen!

When \$this->PDFA is set to true, the following changes will automatically and silently happen:

- list bullets will use the characters from the current font (if available) e.g. • ° ■
- the default font (which in some circumstances is set to a core Adobe font) is set to an alternative embedded font
- annotations (including hyperlinks) have their 'flag' changed to /F 28 which should force printing (this seems to make no difference but is required for PDFA compliance)
- a /Charset entry is included for each font when using embedded subsets
- a Metadata object is added to the file
- an ICC sRGB Color Profile is added as /OutputIntent

Only the first change may make a discernible change to the displayed document. All other changes are to the file structure.

Warnings

Some problems can be fixed by mPDF, but will cause a change to the appearance of your document. By default they will generate warning messages. Once you have assessed the warnings, you can direct mPDF to make these changes automatically by setting in your script:

```
$mpdf->PDFAuto = true; // Overrides warnings making changes when possible to force PDFA1-b compliance
```

The following issues will cause a warning message when you try to generate a PDFA file:

Problem detected	Action taken by mPDF
List bullets cannot be substituted from current font	Bullets substituted by a hyphen "-"
Character substitution (\$useSubstitutions or \$useSubstitutionsMB) is enabled. This would attempt to substitute missing characters using core Adobe fonts which are disallowed.	Character substitution is disabled
Annotation markers are present; these cannot be semi-transparent and may hide underlying text.	Annotation markers are placed in the right margin
Images which have CSS property opacity set less than 1 - PDFA does not allow transparency. NB GIF or PNG images with a simple transparency set (not alpha-channel mask) are valid.	Opacity is changed to 1 (no transparency)
JPG images in CMYK colour are not permitted.	Converted to RGB (only if GD library installed). NB This may significantly change the colour and appearance of the image
Core fonts (Times, Helvetica, Courier) are specified directly e.g. in CSS stylesheet	Font will be substituted with an available sans, serif, or mono font.
Barcode of ISBN, ISSN or EAN-13 type which specifies the code to appear above the barcode (normally uses Helvetica font).	Substitutes available sans-serif font
Form element <select> which normally uses a Zapfdingbats character for the drop-down symbol.	An equal sign "=" will be substituted if ▼ is not available in the default sans-serif font.

Fatal Errors

Some issues cannot be fixed automatically by mPDF and will generate fatal errors:

- \$useCoreFontsOnly is set as **TRUE** (cannot embed core fonts)
- BIG5, SJIS, UHC or GB fonts cannot be used (cannot be embedded)
- Watermarks - text or image - are not permitted (transparency is disallowed so will make text unreadable)
- Using CMYK colour in functions SetTextColor() SetDrawColor() or SetFillColor()
- PNG images with alpha channel transparency ('masks' not allowed)
- encryption is enabled

See Also

- [PDF/X-1a compliance](#)
- [PDFA - Create PDF/A1-b compliant document](#)
- [PDFX - Create PDF/X-1a compliant document](#)
- [PDFAauto - Specify whether to automatically fix issues to create PDF/A1-b compliant document](#)
- [PDFXauto - Specify whether to automatically fix issues to create PDF/X-1a compliant document](#)
- [ICCProfile - Specify the ICC profile for the chosen colorspace used in the document](#)
- [restrictColorSpace - Specify whether to automatically limit the colorspaces used](#)

Useful resources

On-line PDFA Validator: <http://www.validatepdfa.com/online.htm>

Useful info on PDFA: <http://www.pdfa.org/doku.php?id=pdfa:en:techdoc>

PDF/X-1a compliance

(mPDF >= 5.1)

PDF/X-1a is a file format to facilitate printing of electronic documents.

Two key elements to this function are the requirement for PDF/X documents to be 100% self-contained, and all images need to be CMYK or spot colors.

A PDF/X document can be identified as such through PDF/X-specific metadata located in the document. However, claiming to be PDF/X compliant and being so are not necessarily the same.

Important: mPDF is **not guaranteed** to produce fully PDF/X-1a compliant files in all circumstances. It is the users responsibility to check compliance if this is essential.

mPDF and PDF/X-1a compliance

You can make mPDF produce mPDF/X-1a:2003 compliant files by setting:

```
$this->PDFX = true; // Edit in config.php or set at runtime
```

Colorspaces and ICC Profiles

PDF files handle colours internally in a number of different colorspaces. PDFX files can only use CMYK and spot colors, and a colorspace profile must be embedded in the document.

It is likely that most content in a PDF file will be RGB unless you have specifically defined otherwise e.g. a CMYK JPG image file, or colours defined as CMYK() etc.

Colorspaces can be altered using [restrictColorSpace](#).

An ICC Color profile must be embedded in the file.

A default CMYK ICC profile is included with mPDF: SWOP2006_Coated5v2.icc

Sources of free ICC profiles are:

- <http://www.color.org>
- <http://www.eci.org>
- <http://www.color.org/profiles2.xalter>
- http://www.adobe.com/support/downloads/iccprofiles/iccprofiles_win.html

You can change the color profile used by mPDF by adding a *.icc file to the [mpdf]/iccprofiles/ folder. It must be a 4-colour CMYK profile. Then edit config.php e.g.

```
$this->ICCPProfile = 'SWOP2006_Coated5v2';
```

Fonts

All fonts must be embedded in a PDF/X file. mPDF normally uses the core Adobe fonts (Helvetica, Times, Courier, Symbol and Zapfdingbats) in a number of situations, and these font files are not available to embed. This affects:

- \$useCoreFontsOnly is not permitted
- lists cannot use Zapfdingbats font for bullets
- form element <select> cannot use Zapfdingbats character as the drop-down symbol
- ISBN/ISSN barcode cannot use Helvetica for the ISBN number (above the bar)

Also, the fonts GB, BIG5, UHC and SJIS which reference the free CJK font pack from Adobe cannot be used

Transparency and Annotations

Because PDF/X-1a has the objective of ensuring reliable reproduction of the document, no objects may be transparent or semi-transparent. This affects:

- watermarks text or images
- images with the CSS property *opacity* set
- annotation markers

Making it happen!

When \$this->PDFX is set to true, the following changes will automatically and silently happen:

- list bullets will use the characters from the current font (if available) e.g. • ° ■
- the default font (which in some circumstances is set to a core Adobe font) is set to an alternative embedded font
- annotations (including hyperlinks) have their 'flag' changed to /F 28 which should force printing (this seems to make no difference but is required for PDFX compliance)
- a /Charset entry is included for each font when using embedded subsets
- a Metadata object is added to the file
- an ICC Color Profile is added as /OutputIntent

Only the first change may make a discernible change to the displayed document. All other changes are to the file structure.

Warnings

Some problems can be fixed by mPDF, but will cause a change to the appearance of your document. By default they will generate warning messages. Once you have assessed the warnings, you can direct mPDF to make these changes automatically by setting in your script:

```
$mpdf->PDFXauto = true; // Overrides warnings making changes when possible to force PDFX-1a compliance
```

The following issues will cause a warning message when you try to generate a PDFX file:

Problem detected	Action taken by mPDF
List bullets cannot be substituted from current font	Bullets substituted by a hyphen "-"
Character substitution (\$useSubstitutions or \$useSubstitutionsMB) is enabled. This would attempt to substitute missing characters using core Adobe fonts which are disallowed.	Character substitution is disabled
Annotation markers are present; these cannot be semi-transparent and may hide underlying text.	Annotation markers are placed in the right margin
Images which have CSS property opacity set less than 1 - PDFX does not allow transparency. NB GIF or PNG images with a simple transparency set (not alpha-channel mask) are valid.	Opacity is changed to 1 (no transparency)
Core fonts (Times, Helvetica, Courier) are specified directly e.g. in CSS stylesheet	Font will be substituted with an available sans, serif, or mono font.
Barcode of ISBN, ISSN or EAN-13 type which specifies the code to appear above the barcode (normally uses Helvetica font).	Substitutes available sans-serif font
Form element <select> which normally uses a Zapfdingbats character for the drop-down symbol.	An equal sign "=" will be substituted if ▼ is not available in the default sans-serif font.

Fatal Errors

Some issues cannot be fixed automatically by mPDF and will generate fatal errors:

- \$useCoreFontsOnly is set as **TRUE** (cannot embed core fonts)
- BIG5, SJIS, UHC or GB fonts cannot be used (cannot be embedded)
- Watermarks - text or image - are not permitted (transparency is disallowed so will make text unreadable)
- PNG images with alpha channel transparency ('masks' not allowed)
- encryption is enabled

See Also

- PDF/A1-b compliance
- PDFA - Create PDF/A1-b compliant document
- PDFX - Create PDF/X-1a compliant document

- **PDFAuto** - Specify whether to automatically fix issues to create PDF/A1-b compliant document
- **PDFXauto** - Specify whether to automatically fix issues to create PDF/X-1a compliant document
- **ICCProfile** - Specify the ICC profile for the chosen colorspace used in the document
- **restrictColorSpace** - Specify whether to automatically limit the colorspaces used

Capture HTML output

One way of outputting a webpage to mPDF without re-writing your scripts too much, is to buffer the output:

```
include("../mpdf.php");
$mpdf=new mPDF();
// Buffer the following html with PHP so we can store it to a variable later
ob_start();

// This is where your script would normally output the HTML using echo or print

// Now collect the output buffer into a variable
$html = ob_get_contents();
ob_end_clean();

// send the captured HTML from the output buffer to the mPDF class for processing
$mpdf->WriteHTML($html);

$mpdf->Output();
exit;
```

Saving to a database

If you want to save the created file to a database (not necessarily recommended), one user has suggested the following:

```
$s = $mpdf->Output('','S');
$data = bin2hex($s);
$query = "INSERT INTO aaa SET content=0x".$data." ";
```

When you want to output the PDF file:

```
// Get the database content into the variable $data
$pdf = pack("H*", $data );
header('Content-Type: application/pdf');
header('Content-Length: '.strlen($pdf));
header('Content-disposition: inline; filename="'.$name.'"');
header('Cache-Control: public, must-revalidate, max-age=0');
header('Pragma: public');
header('Expires: Sat, 26 Jul 1997 05:00:00 GMT');
header('Last-Modified: '.gmdate('D, d M Y H:i:s').' GMT');
echo $pdf;
exit;
```

Math Formulae with MathJax

mPDF does not support MathML or LaTeX math formulae directly. However you can include math in a PDF document making use of [MathJax](#) following the steps below.

Note: See alternative method to this in [Math with MathJax 2](#)

Note: This version has an updated script.

MathJax is a Javascript program which renders math equations in a browser from either MathML or LaTeX sources. You need to use MathJax first to render the equation, and MathJax needs to be configured to offer SVG as an output option. To demonstrate, start with a sample page such as
<http://www.mathjax.org/demos/tex-samples/>

- 1) Open the HTML page in your browser (allowing MathJax to render the equations).
- 2) Right click over one of the equations and select: Math Settings >> Math Renderer >> SVG (This re-renders all the equations n the page in SVG format),
- 3) Save the page including the javascript-processed code. In Firefox, select File >> Save Page As... (This doesn't work in IE which only saves the original HTML code prior to processing).
- 4) Edit the file to make any necessary changes e.g. CSS stylesheets.
- 5) Run the following script to produce your PDF document (assumes you saved the file as TeXSample.htm). This adjusts the SVG code produced to allow mPDF to display it:

```
$mpdf=new mPDF('');  
  
$html = file_get_contents('TeXSample.htm');  
  
preg_match('/<svg[^>]*>\s*(<defs.*?>.*?</defs>)\s*</svg>/', $html, $m);  
$defs = $m[1];  
  
$html = preg_replace('/<svg[^>]*>\s*<defs.*?</defs>\s*</svg>/' , '' , $html);  
  
$html = preg_replace('/(<svg[^>]*>)/' , "\\\1". $defs, $html);  
  
preg_match_all('/<svg([^>]*style="(.*)")/' , $html, $m);  
for ($i=0;$i<count($m[0]);$i++) {  
    $style=$m[2][$i];  
    preg_match('/width: (.*)/',$style, $wr);  
    $w = $mpdf->ConvertSize($wr[1],0,$mpdf->FontSize) * $mpdf->dpi/25.4;  
    preg_match('/height: (.*)/',$style, $hr);  
    $h = $mpdf->ConvertSize($hr[1],0,$mpdf->FontSize) * $mpdf->dpi/25.4;  
    $replace = '<svg'.$m[1][$i].' width="'.$w.'" height="'.$h.'" style="'.$style.'";  
    $html = str_replace($m[0][$i], $replace, $html);  
}  
  
$mpdf->WriteHTML($html);  
$mpdf->Output();  
exit;
```

See an example of output: <http://mpdf1.com/common/mpdf/examples/MathJaxSample.htm>

Math with MathJax 2

Note: This is an alternative method to the one in [Math Formulae with MathJax](#)

Note: This version has an updated script which should work in all browsers.

1) Adapt the page in which you are testing/writing the Math formulae

- Ideally set the default output to SVG. Otherwise the user will need to select this from: Math Settings >> Math Renderer >> SVG
- Add code for a button to send the processed SVG to your PHP script

Example of MathJax page

```
<!DOCTYPE html>
<html>
<head>

<!-- This line adds MathJax to the page with default SVG output --&gt;
&lt;script type="text/javascript"
src="http://cdn.mathjax.org/mathjax/latest/MathJax.js?config=TeX-AMS-MML_SVG"&gt;&lt;/script&gt;

&lt;/head&gt;
&lt;body&gt;

&lt;h3&gt;The Cauchy-Schwarz Inequality (TeX)&lt;/h3&gt;
\[\left( \sum_{k=1}^n a_k b_k \right)^2 \leq \left( \sum_{k=1}^n a_k^2 \right) \left( \sum_{k=1}^n b_k^2 \right) \]

&lt;h3&gt;Standard Deviation (MathML)&lt;/h3&gt;
&lt;math
display="block"&gt;&lt;mrow&gt;&lt;mi&gt;\#x03c3;&lt;/mi&gt;&lt;mo&gt;=&lt;/mo&gt;&lt;msqrt&gt;&lt;mfrac&gt;&lt;mrow&gt;&lt;mn&gt;1&lt;/mn&gt;&lt;/mrow&gt;&lt;mrow&gt;&lt;mi&gt;N&lt;/mi&gt;&lt;/mrow&gt;&lt;/mfrac&gt;&lt;mstyle
displaystyle="true"&gt;&lt;munderover&gt;&lt;mrow&gt;&lt;mo&gt;\#x2211;&lt;/mo&gt;&lt;/mrow&gt;&lt;mrow&gt;&lt;mi&gt;i&lt;/mi&gt;&lt;mo&gt;=&lt;/mo&gt;&lt;mn&gt;1&lt;/mn&gt;&lt;/mrow&gt;&lt;mrow&gt;&lt;mi&gt;N&lt;/mi&gt;&lt;/mrow&gt;&lt;/munderover&gt;&lt;mrow&gt;&lt;msup&gt;&lt;mrow&gt;&lt;mi&gt;x&lt;/mi&gt;&lt;/mrow&gt;&lt;mrow&gt;&lt;mi&gt;i&lt;/mi&gt;&lt;/mrow&gt;&lt;/msup&gt;&lt;/mrow&gt;&lt;mstyle
stretchy="false"&gt;(&lt;/mo&gt;&lt;msub&gt;&lt;mrow&gt;&lt;mi&gt;x&lt;/mi&gt;&lt;/mrow&gt;&lt;/msub&gt;&lt;mo&gt;\#x2212;&lt;/mo&gt;&lt;mrow&gt;&lt;mi&gt;i&lt;/mi&gt;&lt;/mrow&gt;&lt;/mrow&gt;&lt;msup&gt;&lt;mrow&gt;&lt;mi&gt;i&lt;/mi&gt;&lt;/mrow&gt;&lt;/msup&gt;&lt;/mrow&gt;&lt;/mstyle&gt;&lt;/mrow&gt;&lt;/msqrt&gt;&lt;mo&gt;.&lt;/mo&gt;&lt;/mrow&gt;&lt;/math&gt;

&lt;h3&gt;Inline equation (TeX)&lt;/h3&gt;
&lt;p&gt;Finally, while display equations look good for a page of samples, the ability to mix math and text in a paragraph is also important. This expression <math>\sqrt{3x-1} + (1+x)^2 is an example of an inline equation. As you see, MathJax equations can be used this way as well, without unduly disturbing the spacing between lines.</p>

<!-- This block of code adds a button to send the processed HTML code to your script:
example_test.php -->
<div id="mpdf-create">
<form autocomplete="off" action="example_test.php" method="POST" id="pdfform"
onSubmit="document.getElementById('bodydata').value=encodeURIComponent(document.body.innerHTML);">
<input type="submit" value="PDF" name="submit"/>
<input type="hidden" value="" id="bodydata" name="bodydata" />
</form>
</div>

</body>
</html>
```

2) Now you need a PHP script (in this example: `example_test.php`) which processes the output code from MathJax so that it is readable by mPDF:

Example of 1st part of example_test.php

```
// You should include a check for unwanted external referrers to prevent
```

```
// calls on this script from external websites!

$mpdf=new mPDF('');

$html = $_POST['bodydata'];
$html = urldecode($html);

preg_match('/<svg[^>]*>\s*(<defs.*?>.*?</defs>)\s*</svg>/', $html, $m);
$defs = $m[1];

$html = preg_replace('/<svg[^>]*>\s*<defs.*?</defs>\s*</svg>/' , '' , $html);

$html = preg_replace('/(<svg[^>]*>)/' , "\\\1".$defs , $html);

preg_match_all('/<svg([^\>]*)style="(.*?)"/' , $html, $m);
for ($i=0;$i<count($m[0]);$i++) {
    $style=$m[2][$i];
    preg_match('/width: (.*)/',$style, $wr);
    $w = $mpdf->ConvertSize($wr[1],0,$mpdf->FontSize) * $mpdf->dpi/25.4;
    preg_match('/height: (.*)/',$style, $hr);
    $h = $mpdf->ConvertSize($hr[1],0,$mpdf->FontSize) * $mpdf->dpi/25.4;
    $replace = '<svg'.$m[1][$i].' width="'.$w.'" height="'.$h.'" style="'.$m[2][$i].'";
    $html = str_replace($m[0][$i],$replace,$html);
}

}
```

3a) Finally you can create a PDF document directly based on the MathJax web page submitted:

Example of 2nd part of example_test.php creating a PDF document

```
// ADD a stylesheet
$stylesheet =
/* This helps alignment for inline equations */
img { vertical-align: middle; }
/* This sets padding for display equations (but not in-line ones) */
.MathJax_SVG_Display { padding: 1em 0; }
/* This prevents the Create PDF button being reproduced in the PDF document */
/* Use this method to suppress other parts of the web-page from displaying */
#mpdf-create { display: none; }
/* Add any other CSS styling here for the rest of the document */
/* The CSS/stylesheets information from the original page is not accessible here */
';

$mpdf->WriteHTML($stylesheet,1);

$mpdf->WriteHTML($html);
$mpdf->Output();
exit;
```

3b) Or you could output the prepared SVG code suitable for including directly in your PDF documents:

Example of 2nd part of example_test.php to output the code to a browser

```
...
// To output SVG files (one for each formula) readable by mPDF as text output
header('Content-type: text/plain');
preg_match_all('/<svg(.*)</svg>/' , $html, $m);
for ($i=0;$i<count($m[0]);$i++) {
    $svg = $m[0][$i];
    $svg = preg_replace('/>/','>\n' , $svg);      // Just add some new lines
    echo $svg . "\n\n";
}
exit;
```

See an example: <http://mpdf1.com/common/mpdf/examples/MathJaxSample.htm>

Combining Diacritics

Note: From mPDF v6.0, support for [OpenType layout \(OTL\)](#) makes this section redundant if you use the OTL capability available in most fonts.

In Unicode, letters with diacritics (e.g. ÁáÀàÄä) are usually represented as a single character e.g. Unicode U+0196 is an A Umlaut. There are 4 blocks in Unicode of diacritics or 'marks' which can be used to combine with adjacent letters: Combining Diacritical Marks (U+0300 - U+036F), Combining Diacritical Marks Supplement (U+1DC0 - U+1DFF), Combining Marks for Symbols(U+20D0 - U+20FF) and Combining Half Marks (U+FE20 - U+FE2F).

Software applications use special positioning information stored in OpenType font files to reposition the diacritic/mark depending on the context. mPDF (< v 6.0) does not support this repositioning and is dependent on the font design and original placement of the diacritic:

It is recommended to use precomposed characters whenever possible with mPDF. If not you could use this quick patch to automatically combine diacritics:

In mpdf.php file, function WriteHTML()

Just after the lines:

```
$e = strcode2utf($e);
$e = $this->lesser_entity_decode($e);
```

Add this line:

```
$e = strtr($e, $this->compat);
```

Then add the following in your script (use `$this->compat` if you use it in the `config.php` file) after declaring the class `$mpdf`:

```
$mpdf->compat = array ( "A\xcc\x80"=>"\"x\c3\x80", "A\xcc\x81"=>"\"x\c3\x81", "A\xcc\x82"=>"\"x\c3\x82",  
"A\xcc\x83"=>"\"x\c3\x83", "A\xcc\x88"=>"\"x\c3\x84", "A\xcc\x8a"=>"\"x\c3\x85", "C\xcc\xa7"=>"\"x\c3\x87",  
"E\xcc\x80"=>"\"x\c3\x88", "E\xcc\x81"=>"\"x\c3\x89", "E\xcc\x82"=>"\"x\c3\x8a", "E\xcc\x88"=>"\"x\c3\x8b",  
"I\xcc\x80"=>"\"x\c3\x8c", "I\xcc\x81"=>"\"x\c3\x8d", "I\xcc\x82"=>"\"x\c3\x8e", "I\xcc\x88"=>"\"x\c3\x8f",  
"N\xcc\x83"=>"\"x\c3\x91", "0\xcc\x80"=>"\"x\c3\x92", "0\xcc\x81"=>"\"x\c3\x93", "0\xcc\x82"=>"\"x\c3\x94",  
"0\xcc\x83"=>"\"x\c3\x95", "0\xcc\x88"=>"\"x\c3\x96", "U\xcc\x80"=>"\"x\c3\x99", "U\xcc\x81"=>"\"x\c3\x9a",  
"U\xcc\x82"=>"\"x\c3\x9b", "U\xcc\x88"=>"\"x\c3\x9c", "Y\xcc\x81"=>"\"x\c3\x9d", "a\xcc\x80"=>"\"x\c3\xa0",  
"a\xcc\x81"=>"\"x\c3\xa1", "a\xcc\x82"=>"\"x\c3\xa2", "a\xcc\x83"=>"\"x\c3\xa3", "a\xcc\x88"=>"\"x\c3\xa4",  
"a\xcc\x8a"=>"\"x\c3\xa5", "c\xcc\x7"=>"\"x\c3\xa7", "e\xcc\x80"=>"\"x\c3\xa8", "e\xcc\x81"=>"\"x\c3\xa9",  
"e\xcc\x82"=>"\"x\c3\xaa", "e\xcc\x88"=>"\"x\c3\xab", "i\xcc\x80"=>"\"x\c3\xac", "i\xcc\x81"=>"\"x\c3\xad",  
"i\xcc\x82"=>"\"x\c3\xae", "i\xcc\x88"=>"\"x\c3\xaf", "n\xcc\x83"=>"\"x\c3\xb1", "o\xcc\x80"=>"\"x\c3\xb2",  
"o\xcc\x81"=>"\"x\c3\xb3", "o\xcc\x82"=>"\"x\c3\xb4", "o\xcc\x83"=>"\"x\c3\xb5", "o\xcc\x88"=>"\"x\c3\xb6",  
"u\xcc\x80"=>"\"x\c3\xb9", "u\xcc\x81"=>"\"x\c3\xba", "u\xcc\x82"=>"\"x\c3\xbb", "u\xcc\x88"=>"\"x\c3\xbc",  
"y\xcc\x81"=>"\"x\c3\xbd", "y\xcc\x88"=>"\"x\c3\xbf", "A\xcc\x84"=>"\"x\c4\x80", "a\xcc\x84"=>"\"x\c4\x81",  
"A\xcc\x86"=>"\"x\c4\x82", "a\xcc\x86"=>"\"x\c4\x83", "A\xcc\x88"=>"\"x\c4\x84", "a\xcc\x88"=>"\"x\c4\x85",  
"C\xcc\x81"=>"\"x\c4\x86", "c\xcc\x81"=>"\"x\c4\x87", "C\xcc\x82"=>"\"x\c4\x88", "c\xcc\x82"=>"\"x\c4\x89",  
"C\xcc\x87"=>"\"x\c4\x8a", "c\xcc\x87"=>"\"x\c4\x8b", "C\xcc\x8c"=>"\"x\c4\x8c", "c\xcc\x8c"=>"\"x\c4\x8d",  
"D\xcc\x8c"=>"\"x\c4\x8e", "d\xcc\x8c"=>"\"x\c4\x8f", "E\xcc\x84"=>"\"x\c4\x92", "e\xcc\x84"=>"\"x\c4\x93",  
"E\xcc\x86"=>"\"x\c4\x94", "e\xcc\x86"=>"\"x\c4\x95", "E\xcc\x87"=>"\"x\c4\x96", "e\xcc\x87"=>"\"x\c4\x97",  
"E\xcc\x8a"=>"\"x\c4\x98", "e\xcc\x8a"=>"\"x\c4\x99", "E\xcc\x8c"=>"\"x\c4\x9a", "e\xcc\x8c"=>"\"x\c4\x9b",  
"G\xcc\x82"=>"\"x\c4\x9c", "g\xcc\x82"=>"\"x\c4\x9d", "G\xcc\x86"=>"\"x\c4\x9e", "g\xcc\x86"=>"\"x\c4\x9f",  
"G\xcc\x87"=>"\"x\c4\xa0", "g\xcc\x87"=>"\"x\c4\xa1", "G\xcc\x87"=>"\"x\c4\xab", "g\xcc\x87"=>"\"x\c4\xad",  
"H\xcc\x82"=>"\"x\c4\xa4", "h\xcc\x82"=>"\"x\c4\xa5", "I\xcc\x83"=>"\"x\c4\xab", "i\xcc\x83"=>"\"x\c4\xac",  
"I\xcc\x84"=>"\"x\c4\xaa", "i\xcc\x84"=>"\"x\c4\xab", "I\xcc\x86"=>"\"x\c4\xac", "i\xcc\x86"=>"\"x\c4\xad",  
"I\xcc\x88"=>"\"x\c4\xae", "i\xcc\x88"=>"\"x\c4\xaf", "I\xcc\x87"=>"\"x\c4\xb0", "J\xcc\x82"=>"\"x\c4\xb4",  
"j\xcc\x82"=>"\"x\c4\xb5", "K\xcc\x7"=>"\"x\c4\xb6", "k\xcc\x7"=>"\"x\c4\xb7", "L\xcc\x81"=>"\"x\c4\xb9",  
"l\xcc\x81"=>"\"x\c4\xba", "L\xcc\x7"=>"\"x\c4\xbb", "l\xcc\x7"=>"\"x\c4\xbc", "L\xcc\x8c"=>"\"x\c4\xbd",  
"l\xcc\x8c"=>"\"x\c4\xbe", "N\xcc\x81"=>"\"x\c5\x83", "n\xcc\x81"=>"\"x\c5\x84", "N\xcc\x87"=>"\"x\c5\x85",  
"n\xcc\x87"=>"\"x\c5\x86", "N\xcc\x8c"=>"\"x\c5\x87", "n\xcc\x8c"=>"\"x\c5\x88", "0\xcc\x84"=>"\"x\c5\x8c",  
"o\xcc\x84"=>"\"x\c5\x8d", "0\xcc\x86"=>"\"x\c5\x8e", "o\xcc\x86"=>"\"x\c5\x8f", "0\xcc\x8b"=>"\"x\c5\x90",  
"o\xcc\x8b"=>"\"x\c5\x91", "R\xcc\x81"=>"\"x\c5\x94", "r\xcc\x81"=>"\"x\c5\x95", "R\xcc\x87"=>"\"x\c5\x96",  
"r\xcc\x87"=>"\"x\c5\x97", "R\xcc\x8c"=>"\"x\c5\x98", "r\xcc\x8c"=>"\"x\c5\x99", "S\xcc\x81"=>"\"x\c5\x9a",  
"s\xcc\x81"=>"\"x\c5\x9b", "S\xcc\x82"=>"\"x\c5\x9c", "s\xcc\x82"=>"\"x\c5\x9d", "S\xcc\x87"=>"\"x\c5\x9e",  
"s\xcc\x87"=>"\"x\c5\x9f", "S\xcc\x8c"=>"\"x\c5\xa0", "s\xcc\x8c"=>"\"x\c5\xab", "T\xcc\x87"=>"\"x\c5\xab",  
"t\xcc\x87"=>"\"x\c5\xab", "T\xcc\x8c"=>"\"x\c5\xab", "t\xcc\x8c"=>"\"x\c5\xab", "U\xcc\x83"=>"\"x\c5\xab",
```

```
"u\xcc\x83"=>"\xc5\xaa", "U\xcc\x84"=>"\xc5\xaa", "u\xcc\x84"=>"\xc5\xab", "U\xcc\x86"=>"\xc5\xac",
"u\xcc\x86"=>"\xc5\xad", "U\xcc\x8a"=>"\xc5\xae", "u\xcc\x8a"=>"\xc5\xaf", "U\xcc\x8b"=>"\xc5\xb0",
"u\xcc\x8b"=>"\xc5\xb1", "U\xcc\x8a"=>"\xc5\xb2", "u\xcc\x8a"=>"\xc5\xb3", "W\xcc\x82"=>"\xc5\xb4",
"l\xcc\x82"=>"\xc5\xb5", "Y\xcc\x82"=>"\xc5\xb6", "y\xcc\x82"=>"\xc5\xb7", "Y\xcc\x88"=>"\xc5\xb8",
"Z\xcc\x81"=>"\xc5\xb9", "z\xcc\x81"=>"\xc5\xba", "Z\xcc\x87"=>"\xc5\xbb", "z\xcc\x87"=>"\xc5\xbc",
"Z\xcc\x8c"=>"\xc5\xbd", "z\xcc\x8c"=>"\xc5\xbe", "0\xcc\x9b"=>"\xc6\xaa", "o\xcc\x9b"=>"\xc6\xaa",
"U\xcc\x9b"=>"\xc6\xaf", "u\xcc\x9b"=>"\xc6\xb0", "A\xcc\x8c"=>"\xc7\x8d", "a\xcc\x8c"=>"\xc7\x8e",
"I\xcc\x8c"=>"\xc7\x8f", "i\xcc\x8c"=>"\xc7\x90", "0\xcc\x8c"=>"\xc7\x91", "o\xcc\x8c"=>"\xc7\x92",
"U\xcc\x8c"=>"\xc7\x93", "u\xcc\x8c"=>"\xc7\x94", "\xc3\x9c\xcc\x84"=>"\xc7\x95",
"\xc3\xbc\xcc\x84"=>"\xc7\x96", "\xc3\x9c\xcc\x81"=>"\xc7\x97", "\xc3\xbc\xcc\x81"=>"\xc7\x98",
"\xc3\x9c\xcc\x8c"=>"\xc7\x99", "\xc3\xbc\xcc\x8c"=>"\xc7\x9a", "\xc3\x9c\xcc\x80"=>"\xc7\x9b",
"\xc3\xbc\xcc\x80"=>"\xc7\x9c", "\xc3\x84\xcc\x84"=>"\xc7\x9e", "\xc3\xaa\xcc\x84"=>"\xc7\x9f",
"\xc8\xa6\xcc\x84"=>"\xc7\xaa", "\xc8\xa7\xcc\x84"=>"\xc7\xal", "\xc3\x86\xcc\x84"=>"\xc7\xaa",
"\xc3\xaa\xcc\x84"=>"\xc7\xaa", "G\xcc\x8c"=>"\xc7\xaa", "g\xcc\x8c"=>"\xc7\xaa",
"K\xcc\x8c"=>"\xc7\xaa", "k\xcc\x8c"=>"\xc7\xaa", "0\xcc\x8a"=>"\xc7\xaa", "o\xcc\x8a"=>"\xc7\xab",
"\xc7\xaa\xcc\x84"=>"\xc7\xac", "\xc7\xab\xcc\x84"=>"\xc7\xad", "\xc6\xb7\xcc\x8c"=>"\xc7\xae",
"\xca\x92\xcc\x8c"=>"\xc7\xaf", "j\xcc\x8c"=>"\xc7\xb0", "G\xcc\x81"=>"\xc7\xb4",
"g\xcc\x81"=>"\xc7\xb5", "N\xcc\x80"=>"\xc7\xb8", "n\xcc\x80"=>"\xc7\xb9",
"\xc3\x85\xcc\x81"=>"\xc7\xba", "\xc3\xaa\xcc\x81"=>"\xc7\xbb", "\xc3\x86\xcc\x81"=>"\xc7\xbc",
"\xc3\xaa\xcc\x81"=>"\xc7\xbd", "\xc3\x98\xcc\x81"=>"\xc7\xbe", "\xc3\xb8\xcc\x81"=>"\xc7\xbf",
"A\xcc\x8f"=>"\xc8\x80", "a\xcc\x8f"=>"\xc8\x81", "A\xcc\x91"=>"\xc8\x82", "a\xcc\x91"=>"\xc8\x83",
"E\xcc\x8f"=>"\xc8\x84", "e\xcc\x8f"=>"\xc8\x85", "E\xcc\x91"=>"\xc8\x86", "e\xcc\x91"=>"\xc8\x87",
"I\xcc\x8f"=>"\xc8\x88", "i\xcc\x8f"=>"\xc8\x89", "I\xcc\x91"=>"\xc8\x8a", "i\xcc\x91"=>"\xc8\x8b",
"0\xcc\x8f"=>"\xc8\x8c", "o\xcc\x8f"=>"\xc8\x8d", "0\xcc\x91"=>"\xc8\x8e", "o\xcc\x91"=>"\xc8\x8f",
"R\xcc\x8f"=>"\xc8\x90", "r\xcc\x8f"=>"\xc8\x91", "R\xcc\x91"=>"\xc8\x92", "r\xcc\x91"=>"\xc8\x93",
"U\xcc\x8f"=>"\xc8\x94", "u\xcc\x8f"=>"\xc8\x95", "U\xcc\x91"=>"\xc8\x96", "u\xcc\x91"=>"\xc8\x97",
"S\xcc\xaa"=>"\xc8\x98", "s\xcc\xaa"=>"\xc8\x99", "T\xcc\xaa"=>"\xc8\x9a", "t\xcc\xaa"=>"\xc8\x9b",
"H\xcc\x8c"=>"\xc8\x9e", "h\xcc\x8c"=>"\xc8\x9f", "A\xcc\x87"=>"\xc8\xaa", "a\xcc\x87"=>"\xc8\x7",
"E\xcc\xaa"=>"\xc8\xaa", "e\xcc\xaa"=>"\xc8\xaa", "\xc3\x96\xcc\x84"=>"\xc8\xaa",
"\xc3\xb6\xcc\x84"=>"\xc8\xab", "\xc3\x95\xcc\x84"=>"\xc8\xac", "\xc3\xb5\xcc\x84"=>"\xc8\xad",
"0\xcc\x87"=>"\xc8\xae", "o\xcc\x87"=>"\xc8\xaf", "\xc8\xae\xcc\x84"=>"\xc8\xb0",
"\xc8\xaf\xcc\x84"=>"\xc8\xb1", "Y\xcc\x84"=>"\xc8\xb2", "y\xcc\x84"=>"\xc8\xb3",
"\xc8\x88\xcc\x81"=>"\xc8\x84", "\xc2\x88\xcc\x81"=>"\xc8\x85", "\xe9\x91\xcc\x81"=>"\xe8\x86",
"\xe9\x95\xcc\x81"=>"\xe9\x88", "\xe9\x97\xcc\x81"=>"\xe9\x89", "\xe9\x99\xcc\x81"=>"\xe8\x8a",
"\xe9\x9f\xcc\x81"=>"\xe9\x8c", "\xe9\xaa\xcc\x81"=>"\xe9\x8e", "\xe9\xaa\xcc\x81"=>"\xe9\x8f",
"\xcf\x8a\xcc\x81"=>"\xe9\x90", "\xe9\x99\xcc\x88"=>"\xe9\xaa", "\xe9\xaa\xcc\x88"=>"\xe9\xab",
"\xe9\xb1\xcc\x81"=>"\xe9\xac", "\xe9\xb5\xcc\x81"=>"\xe9\xad", "\xe9\xb7\xcc\x81"=>"\xe9\xae",
"\xe9\xb9\xcc\x81"=>"\xe9\xaf", "\xe9\xb8\xcc\x81"=>"\xe9\xb0", "\xe9\xb9\xcc\x88"=>"\xe9\x8a",
"\xe9\x85\xcc\x88"=>"\xe9\x8b", "\xe9\xbf\xcc\x81"=>"\xe9\x8c", "\xe9\x85\xcc\x81"=>"\xe9\x8d",
"\xe9\x89\xcc\x81"=>"\xe9\x8e", "\xe9\x92\xcc\x81"=>"\xe9\x93", "\xe9\x92\xcc\x88"=>"\xe9\x94",
"\xd0\x95\xcc\x80"=>"\xd0\x80", "\xd0\x95\xcc\x88"=>"\xd0\x81", "\xd0\x93\xcc\x81"=>"\xd0\x83",
"\xd0\x86\xcc\x88"=>"\xd0\x87", "\xd0\x9a\xcc\x81"=>"\xd0\x8c", "\xd0\x98\xcc\x80"=>"\xd0\x8d",
"\xd0\xa3\xcc\x86"=>"\xd0\x8e", "\xd0\x98\xcc\x86"=>"\xd0\x99", "\xd0\xb8\xcc\x86"=>"\xd0\xb9",
"\xd0\xb5\xcc\x80"=>"\xd1\x90", "\xd0\xb5\xcc\x88"=>"\xd1\x91", "\xd0\xb3\xcc\x81"=>"\xd1\x93",
"\xd1\x96\xcc\x88"=>"\xd1\x97", "\xd0\xba\xcc\x81"=>"\xd1\x9c", "\xd0\xb8\xcc\x80"=>"\xd1\x9d",
"\xd1\x83\xcc\x86"=>"\xd1\x9e", "\xd1\xb4\xcc\x8f"=>"\xd1\xb6", "\xd1\xb5\xcc\x8f"=>"\xd1\xb7",
"\xd0\x96\xcc\x86"=>"\xd3\x81", "\xd0\xb6\xcc\x86"=>"\xd3\x82", "\xd0\x90\xcc\x86"=>"\xd3\x90",
"\xd0\xb0\xcc\x86"=>"\xd3\x91", "\xd0\x90\xcc\x88"=>"\xd3\x92", "\xd0\xb0\xcc\x88"=>"\xd3\x93",
"\xd0\x95\xcc\x86"=>"\xd3\x96", "\xd0\xb5\xcc\x86"=>"\xd3\x97", "\xd3\x98\xcc\x88"=>"\xd3\x9a",
"\xd3\x99\xcc\x88"=>"\xd3\x9b", "\xd0\x96\xcc\x88"=>"\xd3\x9c", "\xd0\xb6\xcc\x88"=>"\xd3\x9d",
"\xd0\x97\xcc\x88"=>"\xd3\x9e", "\xd0\xb7\xcc\x88"=>"\xd3\x9f", "\xd0\x98\xcc\x84"=>"\xd3\xaa",
"\xd0\xb8\xcc\x84"=>"\xd3\xaa", "\xd0\x98\xcc\x88"=>"\xd3\x4", "\xd0\xb8\xcc\x88"=>"\xd3\x5",
"\xd0\x9e\xcc\x88"=>"\xd3\xaa", "\xd0\xbe\xcc\x88"=>"\xd3\x7", "\xd3\xaa\xcc\x88"=>"\xd3\xaa",
"\xd3\xaa\xcc\x88"=>"\xd3\xab", "\xd0\xad\xcc\x88"=>"\xd3\xac", "\xd1\x8d\xcc\x88"=>"\xd3\xad",
"\xd0\xaa\xcc\x84"=>"\xd3\xae", "\xd1\x83\xcc\x84"=>"\xd3\xaf", "\xd0\xaa\xcc\x88"=>"\xd3\xb0",
"\xd1\x83\xcc\x88"=>"\xd3\xb1", "\xd0\xaa\xcc\x8b"=>"\xd3\xb2", "\xd1\x83\xcc\x8b"=>"\xd3\xb3",
"\xd0\xaa\xcc\x88"=>"\xd3\xb4", "\xd1\x87\xcc\x88"=>"\xd3\xb5", "\xd0\xab\xcc\x88"=>"\xd3\xb8",
"\xd1\x8b\xcc\x88"=>"\xd3\xb9", "A\xcc\xaa"=>"\xe1\xb8\x80", "a\xcc\xaa"=>"\xe1\xb8\x81",
"B\xcc\x87"=>"\xe1\xb8\x82", "b\xcc\x87"=>"\xe1\xb8\x83", "B\xcc\xaa"=>"\xe1\xb8\x84",
"\b\xcc\xaa"=>"\xe1\xb8\x85", "B\xcc\xb1"=>"\xe1\xb8\x86", "b\xcc\xb1"=>"\xe1\xb8\x87",
"\xc3\x87\xcc\x81"=>"\xe1\xb8\x88", "\xc3\xaa\x7\xcc\x81"=>"\xe1\xb8\x89", "D\xcc\x87"=>"\xe1\xb8\x8a",
"\d\xcc\x87"=>"\xe1\xb8\x8b", "D\xcc\xaa"=>"\xe1\xb8\x8c", "d\xcc\xaa"=>"\xe1\xb8\x8d",
"D\xcc\xb1"=>"\xe1\xb8\x8e", "d\xcc\xb1"=>"\xe1\xb8\x8f", "D\xcc\xaa"=>"\xe1\xb8\x90",
"\d\xcc\xaa"=>"\xe1\xb8\x91", "D\xcc\xad"=>"\xe1\xb8\x92", "d\xcc\xad"=>"\xe1\xb8\x93",
"\xc4\x92\xcc\x80"=>"\xe1\xb8\x94", "\xc4\x93\xcc\x80"=>"\xe1\xb8\x95",
"\xc4\x92\xcc\x81"=>"\xe1\xb8\x96", "\xc4\x93\xcc\x81"=>"\xe1\xb8\x97", "E\xcc\xad"=>"\xe1\xb8\x98",
"\e\xcc\xad"=>"\xe1\xb8\x99", "E\xcc\xb0"=>"\xe1\xb8\x9a", "e\xcc\xb0"=>"\xe1\xb8\x9b",
"\xc8\xaa\xcc\x86"=>"\xe1\xb8\x9c", "\xc8\xaa\xcc\x86"=>"\xe1\xb8\x9d", "F\xcc\x87"=>"\xe1\xb8\x9e",
"\f\xcc\x87"=>"\xe1\xb8\x9f", "G\xcc\x84"=>"\xe1\xb8\xaa", "g\xcc\x84"=>"\xe1\xb8\xaa",
"\H\xcc\x87"=>"\xe1\xb8\xab", "h\xcc\x87"=>"\xe1\xb8\xac", "H\xcc\xaa"=>"\xe1\xb8\xad",
"\h\xcc\xaa"=>"\xe1\xb8\xab", "\h\xcc\x88"=>"\xe1\xb8\xac", "h\xcc\x88"=>"\xe1\xb8\xad",
"\h\xcc\xaa"=>"\xe1\xb8\xab", "\h\xcc\xaa"=>"\xe1\xb8\xac", "H\xcc\xae"=>"\xe1\xb8\xaa",
"\h\xcc\xae"=>"\xe1\xb8\xab", "I\xcc\xb0"=>"\xe1\xb8\xac", "i\xcc\xb0"=>"\xe1\xb8\xad",
```

```

"\xc3\x8f\xcc\x81"=>"\xe1\xb8\xae", "\xc3\xaf\xcc\x81"=>"\xe1\xb8\xaf", "K\xcc\x81"=>"\xe1\xb8\xb0",
"K\xcc\x81"=>"\xe1\xb8\xb1", "K\xcc\xaa"=>"\xe1\xb8\xb2", "K\xcc\xab"=>"\xe1\xb8\xb3",
"K\xcc\xb1"=>"\xe1\xb8\xb4", "k\xcc\xb1"=>"\xe1\xb8\xb5", "L\xcc\xab"=>"\xe1\xb8\xb6",
"\l\xcc\xab"=>"\xe1\xb8\xb7", "\xe1\xb8\xb6\xcc\x84"=>"\xe1\xb8\xb8",
"\l\xcc\xb1"=>"\xe1\xb8\xb9", "L\xcc\xb1"=>"\xe1\xb8\xba", "l\xcc\xb1"=>"\xe1\xb8\xbb",
"L\xcc\xad"=>"\xe1\xb8\xbc", "l\xcc\xad"=>"\xe1\xb8\xbd", "M\xcc\x81"=>"\xe1\xb8\xbe",
"m\xcc\x81"=>"\xe1\xb8\xbf", "M\xcc\x87"=>"\xe1\xb9\x80", "m\xcc\x87"=>"\xe1\xb9\x81",
"m\xcc\xab"=>"\xe1\xb9\x82", "m\xcc\xab"=>"\xe1\xb9\x83", "N\xcc\x87"=>"\xe1\xb9\x84",
"n\xcc\x87"=>"\xe1\xb9\x85", "N\xcc\xab"=>"\xe1\xb9\x86", "n\xcc\xab"=>"\xe1\xb9\x87",
"N\xcc\xb1"=>"\xe1\xb9\x88", "n\xcc\xb1"=>"\xe1\xb9\x89", "N\xcc\xad"=>"\xe1\xb9\x8a",
"n\xcc\xad"=>"\xe1\xb9\x8b", "\xc3\x95\xcc\x81"=>"\xe1\xb9\x8c", "\xc3\xb5\xcc\x81"=>"\xe1\xb9\x8d",
"\xc3\x95\xcc\x88"=>"\xe1\xb9\x8e", "\xc3\xb5\xcc\x88"=>"\xe1\xb9\x8f",
"\xc5\x8c\xcc\x80"=>"\xe1\xb9\x90", "\xc5\x8d\xcc\x81"=>"\xe1\xb9\x91",
"\xc5\x8c\xcc\x81"=>"\xe1\xb9\x92", "\xc5\x8d\xcc\x81"=>"\xe1\xb9\x93", "P\xcc\x81"=>"\xe1\xb9\x94",
"p\xcc\x81"=>"\xe1\xb9\x95", "P\xcc\x87"=>"\xe1\xb9\x96", "p\xcc\x87"=>"\xe1\xb9\x97",
"R\xcc\x87"=>"\xe1\xb9\x98", "r\xcc\x87"=>"\xe1\xb9\x99", "R\xcc\xab"=>"\xe1\xb9\x9a",
"r\xcc\xab"=>"\xe1\xb9\x9b", "\xe1\xb9\x9a\xcc\x84"=>"\xe1\xb9\x9c",
"\xe1\xb9\x9b\xcc\x84"=>"\xe1\xb9\x9d", "R\xcc\xb1"=>"\xe1\xb9\x9e", "r\xcc\xb1"=>"\xe1\xb9\x9f",
"S\xcc\x87"=>"\xe1\xb9\xaa", "s\xcc\x87"=>"\xe1\xb9\xab", "S\xcc\xab"=>"\xe1\xb9\xac",
"s\xcc\xab"=>"\xe1\xb9\xad", "\xc5\x9a\xcc\x87"=>"\xe1\xb9\xad", "\xc5\x9b\xcc\x87"=>"\xe1\xb9\xac",
"\xc5\xaa\xcc\x87"=>"\xe1\xb9\xaa", "\xc5\xaa\xcc\x87"=>"\xe1\xb9\xad",
"\xe1\xb9\xaa\xcc\x87"=>"\xe1\xb9\xaa", "\xe1\xb9\xab\xcc\x87"=>"\xe1\xb9\xac",
"T\xcc\x87"=>"\xe1\xb9\xaa", "t\xcc\x87"=>"\xe1\xb9\xab", "T\xcc\xab"=>"\xe1\xb9\xac",
"t\xcc\xab"=>"\xe1\xb9\xad", "T\xcc\xb1"=>"\xe1\xb9\xae", "t\xcc\xb1"=>"\xe1\xb9\xaf",
"T\xcc\xad"=>"\xe1\xb9\xb0", "t\xcc\xad"=>"\xe1\xb9\xb1", "U\xcc\xab"=>"\xe1\xb9\xb2",
"U\xcc\xab"=>"\xe1\xb9\xb3", "U\xcc\xb0"=>"\xe1\xb9\xb4", "u\xcc\xb0"=>"\xe1\xb9\xb5",
"U\xcc\xad"=>"\xe1\xb9\xb6", "u\xcc\xad"=>"\xe1\xb9\xb7", "\xc5\xaa\xcc\x81"=>"\xe1\xb9\xb8",
"\xc5\xaa\xcc\x81"=>"\xe1\xb9\xb9", "\xc5\xaa\xcc\x88"=>"\xe1\xb9\xba",
"\xc5\xab\xcc\x88"=>"\xe1\xb9\xbb", "\v\xcc\x83"=>"\xe1\xb9\xbc", "\v\xcc\x83"=>"\xe1\xb9\xbd",
"\v\xcc\xab"=>"\xe1\xb9\xbe", "\v\xcc\xab"=>"\xe1\xb9\xbf", "\w\xcc\x80"=>"\xe1\xba\x80",
"\w\xcc\x80"=>"\xe1\xba\x81", "\w\xcc\x81"=>"\xe1\xba\x82", "\w\xcc\x81"=>"\xe1\xba\x83",
"\w\xcc\x88"=>"\xe1\xba\x84", "\w\xcc\x88"=>"\xe1\xba\x85", "\w\xcc\x87"=>"\xe1\xba\x86",
"\w\xcc\x87"=>"\xe1\xba\x87", "\w\xcc\x87"=>"\xe1\xba\x88", "\w\xcc\xab"=>"\xe1\xba\x89",
"X\xcc\x87"=>"\xe1\xba\x8a", "x\xcc\x87"=>"\xe1\xba\x8b", "X\xcc\x88"=>"\xe1\xba\x8c",
"x\xcc\x88"=>"\xe1\xba\x8d", "Y\xcc\x87"=>"\xe1\xba\x8e", "y\xcc\x87"=>"\xe1\xba\x8f",
"Z\xcc\x82"=>"\xe1\xba\x90", "z\xcc\x82"=>"\xe1\xba\x91", "Z\xcc\xab"=>"\xe1\xba\x92",
"z\xcc\xab"=>"\xe1\xba\x93", "z\xcc\xb1"=>"\xe1\xba\x94", "z\xcc\xb1"=>"\xe1\xba\x95",
"h\xcc\xb1"=>"\xe1\xba\x96", "t\xcc\x88"=>"\xe1\xba\x97", "\w\xcc\x8a"=>"\xe1\xba\x98",
"y\xcc\x8a"=>"\xe1\xba\x99", "\xc5\xbf\xcc\x87"=>"\xe1\xba\x9b", "A\xcc\xab"=>"\xe1\xba\xab",
"a\xcc\xab"=>"\xe1\xba\xaa", "A\xcc\x89"=>"\xe1\xba\xab", "a\xcc\x89"=>"\xe1\xba\xab",
"\xc3\x82\xcc\x81"=>"\xe1\xba\xab", "\xc3\xaa\xcc\x81"=>"\xe1\xba\xab",
"\xc3\x82\xcc\x80"=>"\xe1\xba\xab", "\xc3\xaa\xcc\x80"=>"\xe1\xba\xab",
"\xc3\x82\xcc\x89"=>"\xe1\xba\xab", "\xc3\xaa\xcc\x89"=>"\xe1\xba\xab",
"\xc3\x82\xcc\x83"=>"\xe1\xba\xaa", "\xc3\xaa\xcc\x83"=>"\xe1\xba\xab",
"\xe1\xba\xab\xcc\x82"=>"\xe1\xba\xab", "\xe1\xba\xaa\xcc\x82"=>"\xe1\xba\xab\xad",
"\xc4\x82\xcc\x81"=>"\xe1\xba\xab", "\xc4\x83\xcc\x81"=>"\xe1\xba\xab",
"\xc4\x82\xcc\x80"=>"\xe1\xba\xab", "\xc4\x83\xcc\x80"=>"\xe1\xba\xab",
"\xc4\x82\xcc\x89"=>"\xe1\xba\xb2", "\xc4\x83\xcc\x89"=>"\xe1\xba\xb3",
"\xc4\x82\xcc\x83"=>"\xe1\xba\xb4", "\xc4\x83\xcc\x83"=>"\xe1\xba\xb5",
"\xe1\xba\xab\xcc\x86"=>"\xe1\xba\xb6", "\xe1\xba\xaa\xcc\x86"=>"\xe1\xba\xb7",
"E\xcc\xab"=>"\xe1\xba\xb8", "e\xcc\xab"=>"\xe1\xba\xb9", "E\xcc\x89"=>"\xe1\xba\xba",
"e\xcc\x89"=>"\xe1\xba\xbb", "E\xcc\x83"=>"\xe1\xba\xbc", "e\xcc\x83"=>"\xe1\xba\xbd",
"\xc3\x8a\xcc\x81"=>"\xe1\xba\xbe", "\xc3\xaa\xcc\x81"=>"\xe1\xba\xbf",
"\xc3\x8a\xcc\x80"=>"\xe1\xbb\x80", "\xc3\xaa\xcc\x80"=>"\xe1\xbb\x81",
"\xc3\x8a\xcc\x89"=>"\xe1\xbb\x82", "\xc3\xaa\xcc\x89"=>"\xe1\xbb\x83",
"\xc3\x8a\xcc\x83"=>"\xe1\xbb\x84", "\xc3\xaa\xcc\x83"=>"\xe1\xbb\x85",
"\xe1\xba\xb8\xcc\x82"=>"\xe1\xbb\x86", "\xe1\xba\xb9\xcc\x82"=>"\xe1\xbb\x87",
"I\xcc\x89"=>"\xe1\xbb\x88", "i\xcc\x89"=>"\xe1\xbb\x89", "I\xcc\xab"=>"\xe1\xbb\x8a",
"i\xcc\xab"=>"\xe1\xbb\x8b", "o\xcc\xab"=>"\xe1\xbb\x8c", "o\xcc\xab"=>"\xe1\xbb\x8d",
"o\xcc\x89"=>"\xe1\xbb\x8e", "o\xcc\x89"=>"\xe1\xbb\x8f", "\xc3\x94\xcc\x81"=>"\xe1\xbb\x90",
"\xc3\xb4\xcc\x81"=>"\xe1\xbb\x91", "\xc3\x94\xcc\x80"=>"\xe1\xbb\x92",
"\xc3\xb4\xcc\x80"=>"\xe1\xbb\x93", "\xc3\x94\xcc\x89"=>"\xe1\xbb\x94",
"\xc3\xb4\xcc\x89"=>"\xe1\xbb\x95", "\xc3\x94\xcc\x83"=>"\xe1\xbb\x96",
"\xc3\xb4\xcc\x83"=>"\xe1\xbb\x97", "\xe1\xbb\x8c\xcc\x82"=>"\xe1\xbb\x98",
"\xe1\xbb\x8d\xcc\x82"=>"\xe1\xbb\x99", "\xc6\xaa\xcc\x81"=>"\xe1\xbb\x9a",
"\xc6\xaa\xcc\x81"=>"\xe1\xbb\x9b", "\xc6\xaa\xcc\x80"=>"\xe1\xbb\x9c",
"\xc6\xaa\xcc\x80"=>"\xe1\xbb\x9d", "\xc6\xaa\xcc\x89"=>"\xe1\xbb\x9e",
"\xc6\xaa\xcc\x89"=>"\xe1\xbb\x9f", "\xc6\xaa\xcc\x83"=>"\xe1\xbb\xab",
"\xc6\xaa\xcc\x83"=>"\xe1\xbb\xab", "\xc6\xaa\xcc\xab"=>"\xe1\xbb\xab",
"\xc6\xaa\xcc\xab"=>"\xe1\xbb\xab", "\u\xcc\xab"=>"\xe1\xbb\xab", "u\xcc\xab"=>"\xe1\xbb\xab",
"\u\xcc\x89"=>"\xe1\xbb\xab", "u\xcc\x89"=>"\xe1\xbb\xab", "\xc6\xaf\xcc\x81"=>"\xe1\xbb\xab",
"\xc6\xbb\xcc\x81"=>"\xe1\xbb\xab", "\xc6\xaf\xcc\x80"=>"\xe1\xbb\xaa",
"\xc6\xbb\xcc\x80"=>"\xe1\xbb\xab", "\xc6\xaf\xcc\x89"=>"\xe1\xbb\xac",

```

```

"\xc6\xb0\xcc\x89"=>"\xe1\xbb\xad", "\xc6\xaf\xcc\x83"=>"\xe1\xbb\xae",
"\xc6\xb0\xcc\x83"=>"\xe1\xbb\xaf", "\xc6\xaf\xcc\xa3"=>"\xe1\xbb\xb0",
"\xc6\xb0\xcc\xa3"=>"\xe1\xbb\xb1", "Y\xcc\x80"=>"\xe1\xbb\xb2", "y\xcc\x80"=>"\xe1\xbb\xb3",
"Y\xcc\xa3"=>"\xe1\xbb\xb4", "y\xcc\xa3"=>"\xe1\xbb\xb5", "Y\xcc\x89"=>"\xe1\xbb\xb6",
"y\xcc\x89"=>"\xe1\xbb\xb7", "Y\xcc\x83"=>"\xe1\xbb\xb8", "y\xcc\x83"=>"\xe1\xbb\xb9",
"\xce\xb1\xcc\x93"=>"\xe1\xbc\x80", "\xce\xb1\xcc\x94"=>"\xe1\xbc\x81",
"\xe1\xbc\x80\xcc\x80"=>"\xe1\xbc\x82", "\xe1\xbc\x81\xcc\x80"=>"\xe1\xbc\x83",
"\xe1\xbc\x80\xcc\x81"=>"\xe1\xbc\x84", "\xe1\xbc\x81\xcc\x81"=>"\xe1\xbc\x85",
"\xe1\xbc\x80\xcd\x82"=>"\xe1\xbc\x86", "\xe1\xbc\x81\xcd\x82"=>"\xe1\xbc\x87",
"\xce\x91\xcc\x93"=>"\xe1\xbc\x88", "\xce\x91\xcc\x94"=>"\xe1\xbc\x89",
"\xe1\xbc\x88\xcc\x80"=>"\xe1\xbc\x8a", "\xe1\xbc\x89\xcc\x80"=>"\xe1\xbc\x8b",
"\xe1\xbc\x88\xcc\x81"=>"\xe1\xbc\x8c", "\xe1\xbc\x89\xcc\x81"=>"\xe1\xbc\x8d",
"\xe1\xbc\x88\xcd\x82"=>"\xe1\xbc\x8e", "\xe1\xbc\x89\xcd\x82"=>"\xe1\xbc\x8f",
"\xce\xb5\xcc\x93"=>"\xe1\xbc\x90", "\xce\xb5\xcc\x94"=>"\xe1\xbc\x91",
"\xe1\xbc\x90\xcc\x80"=>"\xe1\xbc\x92", "\xe1\xbc\x91\xcc\x80"=>"\xe1\xbc\x93",
"\xe1\xbc\x90\xcc\x81"=>"\xe1\xbc\x94", "\xe1\xbc\x91\xcc\x81"=>"\xe1\xbc\x95",
"\xce\x95\xcc\x93"=>"\xe1\xbc\x98", "\xce\x95\xcc\x94"=>"\xe1\xbc\x99",
"\xe1\xbc\x98\xcc\x80"=>"\xe1\xbc\x9a", "\xe1\xbc\x99\xcc\x80"=>"\xe1\xbc\x9b",
"\xe1\xbc\x98\xcc\x81"=>"\xe1\xbc\x9c", "\xe1\xbc\x99\xcc\x81"=>"\xe1\xbc\x9d",
"\xce\xb7\xcc\x93"=>"\xe1\xbc\xaa", "\xce\xb7\xcc\x94"=>"\xe1\xbc\xab",
"\xe1\xbc\xaa\xcc\x80"=>"\xe1\xbc\xac", "\xe1\xbc\xaa\xcc\x80"=>"\xe1\xbc\xad",
"\xe1\xbc\xaa\xcd\x82"=>"\xe1\xbc\xae", "\xe1\xbc\xaa\xcd\x82"=>"\xe1\xbc\xaf",
"\xce\xb9\xcc\x93"=>"\xe1\xbc\xb0", "\xce\xb9\xcc\x94"=>"\xe1\xbc\xb1",
"\xe1\xbc\xb0\xcc\x80"=>"\xe1\xbc\xb2", "\xe1\xbc\xb1\xcc\x80"=>"\xe1\xbc\xb3",
"\xe1\xbc\xb0\xcc\x81"=>"\xe1\xbc\xb4", "\xe1\xbc\xb1\xcc\x81"=>"\xe1\xbc\xb5",
"\xe1\xbc\xb0\xcd\x82"=>"\xe1\xbc\xb6", "\xe1\xbc\xb1\xcd\x82"=>"\xe1\xbc\xb7",
"\xce\x99\xcc\x93"=>"\xe1\xbc\xb8", "\xce\x99\xcc\x94"=>"\xe1\xbc\xb9",
"\xe1\xbc\xb8\xcc\x80"=>"\xe1\xbc\xba", "\xe1\xbc\xb9\xcc\x80"=>"\xe1\xbc\xbb",
"\xe1\xbc\xb8\xcc\x81"=>"\xe1\xbc\xbc", "\xe1\xbc\xb9\xcc\x81"=>"\xe1\xbc\xbd",
"\xe1\xbc\xb8\xcd\x82"=>"\xe1\xbc\xbe", "\xe1\xbc\xb9\xcd\x82"=>"\xe1\xbc\xbf",
"\xce\xbf\xcc\x93"=>"\xe1\xbd\x80", "\xce\xbf\xcc\x94"=>"\xe1\xbd\x81",
"\xe1\xbd\x80\xcc\x80"=>"\xe1\xbd\x82", "\xe1\xbd\x81\xcc\x80"=>"\xe1\xbd\x83",
"\xe1\xbd\x80\xcc\x81"=>"\xe1\xbd\x84", "\xe1\xbd\x81\xcc\x81"=>"\xe1\xbd\x85",
"\xce\x9f\xcc\x93"=>"\xe1\xbd\x88", "\xce\x9f\xcc\x94"=>"\xe1\xbd\x89",
"\xe1\xbd\x88\xcc\x80"=>"\xe1\xbd\x8a", "\xe1\xbd\x89\xcc\x80"=>"\xe1\xbd\x8b",
"\xe1\xbd\x88\xcc\x81"=>"\xe1\xbd\x8c", "\xe1\xbd\x89\xcc\x81"=>"\xe1\xbd\x8d",
"\xcf\x85\xcc\x93"=>"\xe1\xbd\x90", "\xcf\x85\xcc\x94"=>"\xe1\xbd\x91",
"\xe1\xbd\x90\xcc\x80"=>"\xe1\xbd\x92", "\xe1\xbd\x91\xcc\x80"=>"\xe1\xbd\x93",
"\xe1\xbd\x90\xcc\x81"=>"\xe1\xbd\x94", "\xe1\xbd\x91\xcc\x81"=>"\xe1\xbd\x95",
"\xe1\xbd\x90\xcd\x82"=>"\xe1\xbd\x96", "\xe1\xbd\x91\xcd\x82"=>"\xe1\xbd\x97",
"\xce\xab\xcc\x94"=>"\xe1\xbd\x99", "\xe1\xbd\x99\xcc\x80"=>"\xe1\xbd\x9b",
"\xe1\xbd\x99\xcc\x81"=>"\xe1\xbd\x9d", "\xe1\xbd\x99\xcd\x82"=>"\xe1\xbd\x9f",
"\xcf\x89\xcc\x93"=>"\xe1\xbd\xaa", "\xcf\x89\xcc\x94"=>"\xe1\xbd\xab",
"\xe1\xbd\xaa\xcc\x80"=>"\xe1\xbd\xac", "\xe1\xbd\xaa\xcc\x81"=>"\xe1\xbd\xad",
"\xe1\xbd\xaa\xcd\x82"=>"\xe1\xbd\xae", "\xe1\xbd\xaa\xcd\x82"=>"\xe1\xbd\xaf",
"\xe1\xbd\xcc\x80"=>"\xe1\xbd\xb0", "\xe1\xbd\xcc\x80"=>"\xe1\xbd\xb2",
"\xe1\xbd\xcc\x80"=>"\xe1\xbd\xb4", "\xe1\xbd\xcc\x80"=>"\xe1\xbd\xb6",
"\xe1\xbd\xcc\x80"=>"\xe1\xbd\xb8", "\xe1\xbd\xcc\x80"=>"\xe1\xbd\xba",
"\xe1\xbd\xcc\x80"=>"\xe1\xbd\xbc", "\xe1\xbd\xcc\x80"=>"\xe1\xbe\x80",
"\xe1\xbc\x81\xcd\x85"=>"\xe1\xbe\x81", "\xe1\xbc\x82\xcd\x85"=>"\xe1\xbe\x82",
"\xe1\xbc\x83\xcd\x85"=>"\xe1\xbe\x83", "\xe1\xbc\x84\xcd\x85"=>"\xe1\xbe\x84",
"\xe1\xbc\x85\xcd\x85"=>"\xe1\xbe\x85", "\xe1\xbc\x86\xcd\x85"=>"\xe1\xbe\x86",
"\xe1\xbc\x87\xcd\x85"=>"\xe1\xbe\x87", "\xe1\xbc\x88\xcd\x85"=>"\xe1\xbe\x88",
"\xe1\xbc\x89\xcd\x85"=>"\xe1\xbe\x89", "\xe1\xbc\x8a\xcd\x85"=>"\xe1\xbe\x8a",
"\xe1\xbc\x8b\xcd\x85"=>"\xe1\xbe\x8b", "\xe1\xbc\x8c\xcd\x85"=>"\xe1\xbe\x8c",
"\xe1\xbc\x8d\xcd\x85"=>"\xe1\xbe\x8d", "\xe1\xbc\x8e\xcd\x85"=>"\xe1\xbe\x8e",
"\xe1\xbc\x8f\xcd\x85"=>"\xe1\xbe\x8f", "\xe1\xbc\xaa\xcd\x85"=>"\xe1\xbe\x90",
"\xe1\xbc\xab\xcd\x85"=>"\xe1\xbe\x91", "\xe1\xbc\xac\xcd\x85"=>"\xe1\xbe\x92",
"\xe1\xbc\xab\xcd\x85"=>"\xe1\xbe\x93", "\xe1\xbc\xac\xcd\x85"=>"\xe1\xbe\x94",
"\xe1\xbc\xab\xcd\x85"=>"\xe1\xbe\x95", "\xe1\xbc\xac\xcd\x85"=>"\xe1\xbe\x96",
"\xe1\xbc\xab\xcd\x85"=>"\xe1\xbe\x97", "\xe1\xbc\xac\xcd\x85"=>"\xe1\xbe\x98",
"\xe1\xbc\xab\xcd\x85"=>"\xe1\xbe\x99", "\xe1\xbc\xaa\xcd\x85"=>"\xe1\xbe\x9a",
"\xe1\xbc\xab\xcd\x85"=>"\xe1\xbe\x9b", "\xe1\xbc\xac\xcd\x85"=>"\xe1\xbe\x9c",

```

```
"\xe1\xbc\xad\xcd\x85"=>"\xe1\xbe\x9d", "\xe1\xbc\xae\xcd\x85"=>"\xe1\xbe\x9e",
"\xe1\xbc\xaf\xcd\x85"=>"\xe1\xbe\x9f", "\xe1\xbd\xa0\xcd\x85"=>"\xe1\xbe\xa0",
"\xe1\xbd\xa1\xcd\x85"=>"\xe1\xbe\xa1", "\xe1\xbd\xa2\xcd\x85"=>"\xe1\xbe\xa2",
"\xe1\xbd\xa3\xcd\x85"=>"\xe1\xbe\xa3", "\xe1\xbd\xa4\xcd\x85"=>"\xe1\xbe\xa4",
"\xe1\xbd\xa5\xcd\x85"=>"\xe1\xbe\xa5", "\xe1\xbd\xa6\xcd\x85"=>"\xe1\xbe\xa6",
"\xe1\xbd\xa7\xcd\x85"=>"\xe1\xbe\xa7", "\xe1\xbd\xa8\xcd\x85"=>"\xe1\xbe\xa8",
"\xe1\xbd\xaa\xcd\x85"=>"\xe1\xbe\xaa", "\xe1\xbd\xaa\xcd\x85"=>"\xe1\xbe\xaa",
"\xe1\xbd\xab\xcd\x85"=>"\xe1\xbe\xab", "\xe1\xbd\xac\xcd\x85"=>"\xe1\xbe\xac",
"\xe1\xbd\xad\xcd\x85"=>"\xe1\xbe\xad", "\xe1\xbd\xae\xcd\x85"=>"\xe1\xbe\xae",
"\xe1\xbd\xaf\xcd\x85"=>"\xe1\xbe\xaf", "\xe1\xbd\xcc\x86"=>"\xe1\xbe\xb0",
"\xe1\xbd\xcc\x84"=>"\xe1\xbe\xb1", "\xe1\xbd\xb0\xcd\x85"=>"\xe1\xbe\xb2",
"\xe1\xbd\xcd\x85"=>"\xe1\xbe\xb3", "\xe1\xac\xcd\x85"=>"\xe1\xbe\xb4",
"\xe1\xbd\xcd\x82"=>"\xe1\xbe\xb6", "\xe1\xbe\xb6\xcd\x85"=>"\xe1\xbe\xb7",
"\xe1\xcc\x86"=>"\xe1\xbe\xb8", "\xe1\xcc\x84"=>"\xe1\xbe\xb9",
"\xe1\xcc\x80"=>"\xe1\xbe\xba", "\xe1\xcc\x85"=>"\xe1\xbe\xbc",
"\xe2\xa8\xcd\x82"=>"\xe1\xbf\x81", "\xe1\xbd\xb4\xcd\x85"=>"\xe1\xbf\x82",
"\xe1\xbd\xcd\x85"=>"\xe1\xbf\x83", "\xe1\xae\xcd\x85"=>"\xe1\xbf\x84",
"\xe1\xbd\xcd\x82"=>"\xe1\xbf\x86", "\xe1\xbf\x86\xcd\x85"=>"\xe1\xbf\x87",
"\xe1\xcc\x80"=>"\xe1\xbf\x88", "\xe1\xcc\x80"=>"\xe1\xbf\x8a",
"\xe1\xcc\x85"=>"\xe1\xbf\x8c", "\xe1\xbe\xbf\xcc\x80"=>"\xe1\xbf\x8d",
"\xe1\xbe\xbf\xcc\x81"=>"\xe1\xbf\x8e", "\xe1\xbe\xbf\xcd\x82"=>"\xe1\xbf\x8f",
"\xe1\xbd\xcc\x86"=>"\xe1\xbf\x90", "\xe1\xbd\xcc\x84"=>"\xe1\xbf\x91",
"\xe1\xcc\x80"=>"\xe1\xbf\x92", "\xe1\xbd\xcd\x82"=>"\xe1\xbf\x96",
"\xe1\xcc\x8a\xcd\x82"=>"\xe1\xbf\x97", "\xe1\xcc\x99\xcc\x86"=>"\xe1\xbf\x98",
"\xe1\xcc\x99\xcc\x84"=>"\xe1\xbf\x99", "\xe1\xcc\x99\xcc\x80"=>"\xe1\xbf\x9a",
"\xe1\xbf\xbe\xcc\x80"=>"\xe1\xbf\x9d", "\xe1\xbf\xbe\xcc\x81"=>"\xe1\xbf\x9e",
"\xe1\xbf\xbe\xcd\x82"=>"\xe1\xbf\x9f", "\xe1\xcc\x85\xcc\x86"=>"\xe1\xbf\xa0",
"\xe1\xcc\x85\xcc\x84"=>"\xe1\xbf\xa1", "\xe1\xcc\x8b\xcc\x80"=>"\xe1\xbf\xa2",
"\xe1\xcc\x81\xcc\x93"=>"\xe1\xbf\xa4", "\xe1\xcc\x81\xcc\x94"=>"\xe1\xbf\xa5",
"\xe1\xcc\x85\xcd\x82"=>"\xe1\xbf\xa6", "\xe1\xcc\x8b\xcd\x82"=>"\xe1\xbf\xa7",
"\xe1\xcc\x86"=>"\xe1\xbf\xa8", "\xe1\xcc\x84"=>"\xe1\xbf\xa9",
"\xe1\xcc\x80"=>"\xe1\xbf\xaa", "\xe1\xcc\x81\xcc\x94"=>"\xe1\xbf\xac",
"\xe2\xa8\xcc\x80"=>"\xe1\xbf\xad", "\xe1\xbd\xbc\xcd\x85"=>"\xe1\xbf\xb2",
"\xe1\xcc\x89\xcd\x85"=>"\xe1\xbf\xb3", "\xe1\xcc\x8e\xcd\x85"=>"\xe1\xbf\xb4",
"\xe1\xcc\x89\xcd\x82"=>"\xe1\xbf\xb6", "\xe1\xbf\xb6\xcd\x85"=>"\xe1\xbf\xb7",
"\xe1\xcc\x9f\xcc\x80"=>"\xe1\xbf\xb8", "\xe1\xcc\x9a\xcc\x80"=>"\xe1\xbf\xba",
"\xe1\xcc\x9a\xcd\x85"=>"\xe1\xbf\xbc");
```

REAL LIFE EXAMPLES

PDF from every page of website

If you want to add a link to pages on your website to create a PDF version of the page, here is one way to do it.

(If you have already set up a CSS stylesheet for media="print" which works for mPDF, you can omit the first 2 steps.)

Write a stylesheet suitable for presenting your webpage in mPDF. Most webpages have a header and menu on the left or right which you will not want to appear in the PDF document. You can use display: none to omit these elements. From mPDF >= 5.0 this will also work for inline elements. You may have something like this:

```
#myheader, #mysearchbar, #myleftcol, #myfooter { display: none; }
span.abstract_link { display: none; }

#maincontents {
    float:none;
    margin:0px;
    overflow:auto;
    padding:0;
    width:100%;
}
```

Save this file as e.g. mypdf.css

Then add this line to your webpages. This should be at the end of the document <head> section, to override any preceding style definitions:

```
<link href="mypdf.css" type="text/css" rel="stylesheet" media="mpdf" />
```

NB The media="mpdf" means that the stylesheet will be ignored by browsers, but can be selected for use by mPDF - see below.

Now create a file e.g. makepdf.php and add the following script. Note you will need to edit the lines in italics.

```
<?php

// Define relative path from this script to mPDF
define('_MPDF_PATH','../../common/mpdf/');

include(_MPDF_PATH . "mpdf.php");
$url = urldecode($_REQUEST['url']);

// To prevent anyone else using your script to create their PDF files
if (!preg_match('/^http://\w+\.\w+\.\w+\.com/', $url)) { die("Access denied"); }

// For $_POST i.e. forms with fields
if (count($_POST)>0) {
    $ch = curl_init($url);
    curl_setopt($ch, CURLOPT_HEADER, 0);
    curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1 );
    foreach($_POST AS $name=>$post) {
        $formvars = array($name=>$post." \n");
    }
    curl_setopt($ch, CURLOPT_POSTFIELDS, $formvars);
    $html = curl_exec($ch);
    curl_close($ch);
}
else if (ini_get('allow_url_fopen')) {
    $html = file_get_contents($url);
}
else {
    $ch = curl_init($url);
    curl_setopt($ch, CURLOPT_HEADER, 0);
    curl_setopt ( $ch , CURLOPT_RETURNTRANSFER , 1 );
    $html = curl_exec($ch);
    curl_close($ch);
}
```

```
$mpdf=new mPDF('');
$mpdf->useSubstitutions = true; // optional - just as an example
$mpdf->SetHeader($url.'||Page {PAGENO}'); // optional - just as an example
$mpdf->CSSselectMedia='mpdf'; // assuming you used this in the document header
$mpdf->setBasePath($url);
$mpdf->WriteHTML($html);
$mpdf->Output();
exit;

?>
```

Now, the link from your webpages. This code can be inserted anywhere on the page. (The <![CDATA[bit is to make the page compatible with XHTML.) This code will work if your webpages are simple files e.g. myfile.html or if they are selected using variables in the URI (i.e. HTTP GET method) e.g. myfile.php?cc=1&var=35

```
<script language="javascript" type="text/javascript">
/* <![CDATA[ */
    document.write('<a href="makepdf.php?url=' + encodeURIComponent(location.href) + '">');
    document.write('Create PDF file of this page');
    document.write('</a>');
/* ]]> */
</script>
```

Use this to generate the code, if the webpages are selected using variables hidden from the URI e.g. HTTP POST method using a form:

```
$mpdf_link =
<script language="javascript" type="text/javascript">
/* <![CDATA[ */
    document.write('<form method="POST" action="makepdf.php?url=' +
encodeURIComponent(location.href) + '>');
';
foreach($_POST AS $name=>$post) {
    $mpdf_link .= 'document.write('<input type="hidden" name="'.$name.'" value="'.$post.'" />');
'."\n";
}
$mpdf_link .= '
    document.write('<input type="submit" name="submit" value="Create PDF file of this page"
/>');
    document.write('</form>');
/* ]]> */
</script>
';

echo $mpdf_link;
```

Invoice

This example will produce a tabulated invoice with headers and footers. See the result.

```
<?php

include("../mpdf.php");

$mpdf=new mPDF('win-1252','A4','','',20,15,48,25,10,10);
$mpdf->useOnlyCoreFonts = true; // false is default
$mpdf->SetProtection(array('print'));
$mpdf->SetTitle("Acme Trading Co. - Invoice");
$mpdf->SetAuthor("Acme Trading Co.");
$mpdf->SetWatermarkText("Paid");
$mpdf->showWatermarkText = true;
$mpdf->watermark_font = 'DejaVuSansCondensed';
$mpdf->watermarkTextAlpha = 0.1;
$mpdf->SetDisplayMode('fullpage');

$html =
<html>
<head>
<style>
body {font-family: sans-serif;
      font-size: 10pt;
}
p {   margin: 0pt;
}
td { vertical-align: top; }
.items td {
    border-left: 0.1mm solid #000000;
    border-right: 0.1mm solid #000000;
}
table thead td { background-color: #EEEEEE;
    text-align: center;
    border: 0.1mm solid #000000;
}
.items td.blanktotal {
    background-color: #FFFFFF;
    border: 0mm none #000000;
    border-top: 0.1mm solid #000000;
    border-right: 0.1mm solid #000000;
}
.items td.totals {
    text-align: right;
    border: 0.1mm solid #000000;
}
</style>
</head>
<body>

<!--mpdf
<htmlpageheader name="myheader">
<table width="100%"><tr>
<td width="50%" style="color:#0000BB;"><span style="font-weight: bold; font-size: 14pt;">Acme
Trading Co.</span><br />123 Anystreet<br />Your City<br />GD12 4LP<br /><span style="font-size:
15pt;">&#9742;</span> 01777 123 567</td>
<td width="50%" style="text-align: right;">Invoice No.<br /><span style="font-weight: bold; font-
size: 12pt;">0012345</span></td>
</tr></table>
</htmlpageheader>

<htmlpagefooter name="myfooter">
<div style="border-top: 1px solid #000000; font-size: 9pt; text-align: center; padding-top: 3mm; ">
Page {PAGENO} of {nb}
</div>
</htmlpagefooter>

<sethtmlpageheader name="myheader" value="on" show-this-page="1" />
<sethtmlpagefooter name="myfooter" value="on" />
$mpdf-->
```

```
<div style="text-align: right">Date: '.date('jS F Y').'</div>

<table width="100%" style="font-family: serif;" cellpadding="10">
<tr>
<td width="45%" style="border: 0.1mm solid #888888;"><span style="font-size: 7pt; color: #555555; font-family: sans;">>SOLD TO:</span><br /><br />345 Anotherstreet<br />Little Village<br />Their City<br />CB22 6S0</td>
<td width="10%">&ampnbsp</td>
<td width="45%" style="border: 0.1mm solid #888888;"><span style="font-size: 7pt; color: #555555; font-family: sans;">>SHIP TO:</span><br /><br />345 Anotherstreet<br />Little Village<br />Their City<br />CB22 6S0</td>
</tr>
</table>

<table class="items" width="100%" style="font-size: 9pt; border-collapse: collapse;" cellpadding="8">
<thead>
<tr>
<td width="15%">REF. NO.</td>
<td width="10%">QUANTITY</td>
<td width="45%">DESCRIPTION</td>
<td width="15%">UNIT PRICE</td>
<td width="15%">AMOUNT</td>
</tr>
</thead>
<tbody>
<!-- ITEMS HERE -->
<tr>
<td align="center">MF1234567</td>
<td align="center">1</td>
<td>Large pack Hoover bags</td>
<td align="right">&pound;2.56</td>
<td align="right">&pound;25.60</td>
</tr>
<tr>
<td align="center">MX37801982</td>
<td align="center">1</td>
<td>Womans waterproof jacket<br />Options - Red and charcoal.</td>
<td align="right">&pound;112.56</td>
<td align="right">&pound;112.56</td>
</tr>
<tr>
<td align="center">MR7009298</td>
<td align="center">25</td>
<td>Steel nails; oval head; 30mm x 3mm. Packs of 1000.</td>
<td align="right">&pound;12.26</td>
<td align="right">&pound;325.60</td>
</tr>
<tr>
<td align="center">MF1234567</td>
<td align="center">10</td>
<td>Large pack Hoover bags</td>
<td align="right">&pound;2.56</td>
<td align="right">&pound;25.60</td>
</tr>
<tr>
<td align="center">MX37801982</td>
<td align="center">1</td>
<td>Womans waterproof jacket<br />Options - Red and charcoal.</td>
<td align="right">&pound;112.56</td>
<td align="right">&pound;112.56</td>
</tr>
<tr>
<td align="center">MR7009298</td>
<td align="center">25</td>
<td>Steel nails; oval head; 30mm x 3mm. Packs of 1000.</td>
<td align="right">&pound;12.26</td>
<td align="right">&pound;325.60</td>
</tr>
<tr>
<td align="center">MF1234567</td>
<td align="center">10</td>
```

```
<td>Large pack Hoover bags</td>
<td align="right">&pound;2.56</td>
<td align="right">&pound;25.60</td>
</tr>
<tr>
<td align="center">MX37801982</td>
<td align="center">1</td>
<td>Womans waterproof jacket<br />Options - Red and charcoal.</td>
<td align="right">&pound;112.56</td>
<td align="right">&pound;112.56</td>
</tr>
<tr>
<td align="center">MR7009298</td>
<td align="center">25</td>
<td>Steel nails; oval head; 30mm x 3mm. Packs of 1000.</td>
<td align="right">&pound;12.26</td>
<td align="right">&pound;325.60</td>
</tr>
<tr>
<td align="center">MF1234567</td>
<td align="center">10</td>
<td>Large pack Hoover bags</td>
<td align="right">&pound;2.56</td>
<td align="right">&pound;25.60</td>
</tr>
<tr>
<td align="center">MX37801982</td>
<td align="center">1</td>
<td>Womans waterproof jacket<br />Options - Red and charcoal.</td>
<td align="right">&pound;112.56</td>
<td align="right">&pound;112.56</td>
</tr>
<tr>
<td align="center">MR7009298</td>
<td align="center">25</td>
<td>Steel nails; oval head; 30mm x 3mm. Packs of 1000.</td>
<td align="right">&pound;12.26</td>
<td align="right">&pound;325.60</td>
</tr>
<tr>
<td align="center">MF1234567</td>
<td align="center">10</td>
<td>Large pack Hoover bags</td>
<td align="right">&pound;2.56</td>
<td align="right">&pound;25.60</td>
</tr>
<tr>
<td align="center">MX37801982</td>
<td align="center">1</td>
<td>Womans waterproof jacket<br />Options - Red and charcoal.</td>
<td align="right">&pound;112.56</td>
<td align="right">&pound;112.56</td>
</tr>
<tr>
<td align="center">MF1234567</td>
<td align="center">10</td>
<td>Large pack Hoover bags</td>
<td align="right">&pound;2.56</td>
<td align="right">&pound;25.60</td>
</tr>
<tr>
<td align="center">MX37801982</td>
<td align="center">1</td>
<td>Womans waterproof jacket<br />Options - Red and charcoal.</td>
<td align="right">&pound;112.56</td>
<td align="right">&pound;112.56</td>
</tr>
<tr>
<td align="center">MR7009298</td>
<td align="center">25</td>
<td>Steel nails; oval head; 30mm x 3mm. Packs of 1000.</td>
<td align="right">&pound;12.26</td>
<td align="right">&pound;325.60</td>
```

```

</tr>
<tr>
<td align="center">MR7009298</td>
<td align="center">25</td>
<td>Steel nails; oval head; 30mm x 3mm. Packs of 1000.</td>
<td align="right">&pound;12.26</td>
<td align="right">&pound;325.60</td>
</tr>
<tr>
<td align="center">MF1234567</td>
<td align="center">10</td>
<td>Large pack Hoover bags</td>
<td align="right">&pound;2.56</td>
<td align="right">&pound;25.60</td>
</tr>
<tr>
<td align="center">MX37801982</td>
<td align="center">1</td>
<td>Womans waterproof jacket<br />Options - Red and charcoal.</td>
<td align="right">&pound;112.56</td>
<td align="right">&pound;112.56</td>
</tr>
<tr>
<td align="center">MR7009298</td>
<td align="center">25</td>
<td>Steel nails; oval head; 30mm x 3mm. Packs of 1000.</td>
<td align="right">&pound;12.26</td>
<td align="right">&pound;325.60</td>
</tr>
<!-- END ITEMS HERE -->
<tr>
<td class="blanktotal" colspan="3" rowspan="6"></td>
<td class="totals">Subtotal:</td>
<td class="totals">&pound;1825.60</td>
</tr>
<tr>
<td class="totals">Tax:</td>
<td class="totals">&pound;18.25</td>
</tr>
<tr>
<td class="totals">Shipping:</td>
<td class="totals">&pound;42.56</td>
</tr>
<tr>
<td class="totals">><b>TOTAL :</b></td>
<td class="totals">><b>&pound;1882.56</b></td>
</tr>
<tr>
<td class="totals">Deposit:</td>
<td class="totals">>&pound;100.00</td>
</tr>
<tr>
<td class="totals">><b>Balance due:</b></td>
<td class="totals">><b>&pound;1782.56</b></td>
</tr>
</tbody>
</table>
<div style="text-align: center; font-style: italic;">Payment terms: payment due in 30 days</div>
</body>
</html>
';
$mpdf->WriteHTML($html);

$mpdf->Output(); exit;

exit;

?>

```

Year Book

This script was written to create a YearBook, where each entry must fit on a quarter page. See the [result](#).

```
<?php

// First write all your entries to a PDF file, forcing each entry to fit on one page
include("../mpdf.php");

// Define the maximum containing box width & height for each text box as it will appear on the final
page (no padding or margin here)
$pw = 80;           // Width
$ph = 110;          // Height
$minK = 0.7;         // Maximum scaling factor 0.7 = 70%
$inc = 0.01;        // Increment to change scaling factor 0.05 = 5%
$spacing = 10;       // millimetres (vertically and horizontally between boxes in output) shrinks if
boxes too big
$border = 3;         // millimetres round final boxes (-1 for no border)
$align = 'T';         // T(op) or M(iddle) for content of final output boxes

// Only change the first parameter of the next line if required e.g. utf-8
$mpdf = new mPDF('', array(($pw*(1/$minK)),($ph*(1/$minK))), ',', 0, ($pw*(1/$minK))- $pw, 0, ($ph*(1/$minK))-$ph, 0, 0);

$pph = array();

// FOR EACH ENTRY FOR YOUR YEARBOOK saving the page height in $pph (where $html is the HTML code for
the entry):
//     $pph[$i] = SinglePage($html, $pw, $ph, $minK);

//=====
// .. but we will use this for an example

$html1 =
<style>
div { text-align: justify; }
</style>
<h2>Joanne Smith 2002-2007</h2><div>This is the normal text in the div: Nulla felis erat, imperdiet
eu, ullamcorper non, nonummy quis, elit. Suspendisse potenti. Ut a eros orci. Morbi feugiat pulvinar
dolor. Cras odio. Donec mattis, nisi id euismod auctor, neque metus pellentesque,  risus at eleifend lacus sapien et
risus. Phasellus metus. Phasellus feugiat, lectus ac aliquam molestie, leo lacus tincidunt turpis,
vel aliquam quam odio et sapien. Mauris ante pede, auctor ac, suscipit quis, malesuada sed, nulla. Integer sit amet odio sit
amet lectus lectus euismod. Donec et nulla. Sed quis orci. </div>
';

$html2 =
<style>
div { text-align: justify; }
</style>
<h2>Tim Another 2001-2007</h2><div>This is the normal text in the div: Nulla felis erat, imperdiet
eu, ullamcorper non, nonummy quis, elit. Suspendisse potenti. Ut a eros at ligula vehicula pretium.
Maecenas feugiat pede vel risus. et lectus. Fusce eleifend neque sit amet erat. Integer consectetur
nulla non orci. Morbi feugiat pulvinar dolor. Cras odio. Donec mattis, nisi id euismod auctor, neque
metus pellentesque,  risus at
eleifend lacus sapien et risus. Phasellus metus, suscipit quis, malesuada sed, nulla. Integer sit
amet odio sit amet lectus lectus euismod. Donec et nulla. Sed quis orci. <br />
Morbi feugiat pulvinar dolor. Cras odio. Donec mattis, nisi id euismod auctor, neque metus
pellentesque risus, at eleifend lacus sapien et risus. Phasellus metus. Phasellus feugiat, lectus ac
aliquam molestie, leo lacus tincidunt turpis, vel aliquam quam odio et sapien. Mauris ante pede,
auctor ac, suscipit quis, malesuada sed, nulla. Integer sit amet odio sit amet lectus lectus
euismod. Donec et nulla. Sed quis orci. </div>
';

for($i=1; $i<=10; $i++) {
    // $html = $html;
    if ($i % 3 == 1) { $html = $html2; }
    else { $html = $html1; }
    $pph[$i] = SinglePage($html, $pw, $ph, $minK) ;      // $pph saves the actual height of each page
```

```

}

//=====
// Save the pages to a file
$mpdf->Output('test.pdf', 'F');

// Now collate those pages using mPDFI - 4 pages to one page
include("../mpdfi/mpdfi.php");

$mpdf=new mPDFI();
$mpdf->SetDisplayMode('fullpage');

$mpdf->SetHeader('{DATE j-m-Y}|My Yearbook 2005|{PAGENO}');
$mpdf->SetFooter('|Printed using mPDF|');

$pagecount = $mpdf->SetSourceFile('test.pdf');
for($i=1; $i<=$pagecount; $i++) {
    if ($i % 4 == 1) { $mpdf->AddPage(); }
    $pgheight = $mpdf->h - $mpdf->tMargin - $mpdf->bMargin;
    $hspacing = min($spacing,($mpdf->pgwidth - $pw*2) );
    $vspacing = min($spacing,($pgheight - $ph*2) );
    $x1 = $mpdf->lMargin + ($mpdf->pgwidth/2 - $hspacing/2 - $pw)/2;
    $x2 = $mpdf->lMargin + $mpdf->pgwidth/2 + $hspacing/2 + ($mpdf->pgwidth/2 - $hspacing/2 - $pw)/2;
    $y1 = $mpdf->tMargin + ($pgheight /2 - $vspacing/2 - $ph)/2;
    $y2 = $mpdf->tMargin + $pgheight /2 + $vspacing/2 + ($pgheight /2 - $vspacing/2 - $ph)/2;
    if ($i % 4 == 1) { $x = $x1; $y = $y1; }
    else if ($i % 4 == 2) { $x = $x2; $y = $y1; }
    else if ($i % 4 == 3) { $x = $x1; $y = $y2; }
    else if ($i % 4 == 0) { $x = $x2; $y = $y2; }
    $tplIdx = $mpdf->ImportPage($i, 0,0,$pw,$pph[$i]);

    if ($align=='T') { $mpdf->UseTemplate($tplIdx, $x, $y, $pw, $pph[$i]); }
    else { $mpdf->UseTemplate($tplIdx, $x, ($y + ((($ph - $pph[$i])/2)), $pw, $pph[$i]); }

    if ($border >= 0) { $mpdf->Rect($x-$border, $y-$border, $pw+2*$border, $ph+2*$border); }
}

$mpdf->Output();

exit;

//=====
function SinglePage($html, $pw, $ph, $minK=1, $inc=0.1) {
// returns height of page
global $mpdf;
$mpdf->AddPage('','','','','','','',',$mpdf->w - $pw)', ',$mpdf->h - $ph),0,0);
$k = 1;

$currenpage = $mpdf->page;
$mpdf->WriteHTML($html);

$newpage = $mpdf->page;
while($currenpage != $newpage) {
    for($u=0;$u<=($newpage-$currenpage);$u++) {
        // DELETE PAGE - the added page
        unset($mpdf->pages[$mpdf->page]);
        if ($mpdf->ktAnnots[$mpdf->page]) { unset( $mpdf->ktAnnots[$mpdf->page] ); }
        if ($mpdf->tbrot_Annots[$mpdf->page]) { unset( $mpdf->tbrot_Annots[$mpdf->page] ); }
        if ($mpdf->kwt_Annots[$mpdf->page]) { unset( $mpdf->kwt_Annots[$mpdf->page] ); }
        if ($mpdf->PageAnnots[$mpdf->page]) { unset( $mpdf->PageAnnots[$mpdf->page] ); }
        if ($mpdf->ktBlock[$mpdf->page]) { unset( $mpdf->ktBlock[$mpdf->page] ); }
        if ($mpdf->PageLinks[$mpdf->page]) { unset( $mpdf->PageLinks[$mpdf->page] ); }
        if ($mpdf->pageoutput[$mpdf->page]) { unset( $mpdf->pageoutput[$mpdf->page] ); }
        // Go to page before - so can addpage
        $mpdf->page--;
    }
    // mPDF 2.4 Float Images
    if (count($mpdf->floatbuffer)) {
        $mpdf->objectbuffer[] = $mpdf->floatbuffer['objattr'];
        $mpdf->printobjectbuffer(false);
        $mpdf->objectbuffer = array();
        $mpdf->floatbuffer = array();
        $mpdf->float = false;
    }
}

```

```
}

$k += $inc;
if ((1/$k) < $minK) { die("Page no. ".$mpdf->page." is too large to fit"); }
$w = $pw * $k;
$h = $ph * $k;
$mpdf->_beginpage('', '', ($mpdf->w - $w), '', ($mpdf->h - $h));
$currenpage = $mpdf->page;

$mpdf->_out('2 J');
$mpdf->_out(sprintf('%2f w', 0.1*$mpdf->k));
$mpdf->SetFont($mpdf->default_font, '', $mpdf->default_font_size, true, true); // forces
write
$mpdf->SetDrawColor(0);
$mpdf->SetFillColor(255);
$mpdf->SetTextColor(0);
$mpdf->ColorFlag=false;

// Start Transformation
$mpdf->StartTransform();
$mpdf->transformScale((100/$k), (100/$k), 0, 0);

$mpdf->WriteHTML($html);

$newpage = $mpdf->page;

//Stop Transformation
$mpdf->StopTransform();
}
return ($mpdf->y / $k);
}
?>
```

Colour Charts CMYK

This example produces colour charts for all the CMYK colours (incrementing each value by 10). See the [result](#).

```
<?php

include("../mpdf.php");

$mpdf=new mPDF('win-1252','A4-L');

$mpdf->useOnlyCoreFonts = true;
$mpdf->SetDisplayMode('fullpage');

$mpdf->SetFontSize(6);

$pm = 8; // page margin
$w = 20;
$h = 10;
$m = 5;

for($k=0;$k<=8;$k++) { // Black - page group
    for($y=0;$y<=10;$y++) { // Yellow - page group
        $mpdf->AddPage();
        for($i=0;$i<=10;$i++) { // Rows (Magenta)
            for($j=0; $j<=10; $j++) { // Cols (Cyan)
                $mpdf->SetXY($pm+($j*($w+$m)), $pm+($i*($h+$m)));
                $mpdf->SetFillColor((($j*10), ($i*10), ($y*10), ($k*10)));
                $mpdf->Cell($w,$h,'',0,0,'L',1);
                $mpdf->SetXY($pm+($j*($w+$m)), $h+$pm+($i*($h+$m)));
                $txt = 'C:'.($j*10).' M:'.($i*10).' Y:'.($y*10).' K:'.($k*10);
                $mpdf->Cell($w,4,$txt,0,0,'L');

            }
        }
    }
}

$mpdf->Output('mpdf.pdf','I');

exit;

?>
```

E-mail a PDF file

This example shows how to create a PDF file and e-mail it:

```
<?php

include("../mpdf.php"); //Include mPDF Class

$mpdf=new mPDF(); // Create new mPDF Document

//Beginning Buffer to save PHP variables and HTML tags
ob_start();

// Function Date
$day = date('d');
$month = date('m');
$year = date('Y');

switch ($month)
{
case 1: $month = "January"; break;
case 2: $month = "February"; break;
case 3: $month = "March"; break;
case 4: $month = "April"; break;
case 5: $month = "May"; break;
case 6: $month = "June"; break;
case 7: $month = "July"; break;
case 8: $month = "August"; break;
case 9: $month = "September"; break;
case 10: $month = "October"; break;
case 11: $month = "November"; break;
case 12: $month = "December"; break;
}

echo "Hello World
Today is $month $day, $year";

$html = ob_get_contents();
ob_end_clean();
//Here convert the encode for UTF-8, if you prefer the ISO-8859-1 just change for
$mpdf->WriteHTML($html);
$mpdf->WriteHTML(utf8_encode($html));

$content = $mpdf->Output('', 'S');

$content = chunk_split(base64_encode($content));
$mailto = 'mailto@mailto.com'; //Mailto here
$from_name = 'ACME Corps Ltd'; //Name of sender mail
$from_mail = 'mailfrom@mailfrom.com'; //Mailfrom here
$subject = 'subjecthere';
$message = 'mailmessage';
$filename = "yourfilename-".date("d-m-Y_H-i",time()); //Your Filenname with local date and time

//Headers of PDF and e-mail
$boundary = "XYZ- " . date("dmYis") . "-ZYX";

$header = "--$boundary\r\n";
$header .= "Content-Transfer-Encoding: 8bits\r\n";
$header .= "Content-Type: text/html; charset=ISO-8859-1\r\n\r\n"; // or utf-8
$header .= "$message\r\n";
$header .= "--$boundary\r\n";
$header .= "Content-Type: application/pdf; name=\"".$filename."\"\r\n";
$header .= "Content-Disposition: attachment; filename=\"".$filename."\"\r\n";
$header .= "Content-Transfer-Encoding: base64\r\n\r\n";
$header .= "$content\r\n";
$header .= "--$boundary--\r\n";

$header2 = "MIME-Version: 1.0\r\n";
$header2 .= "From: ".$from_name." \r\n";
$header2 .= "Return-Path: $from_mail\r\n";
$header2 .= "Content-type: multipart/mixed; boundary=\"$boundary\"\r\n";
```

```
$header2 .= "$boundary\r\n";  
  
mail($mailto,$subject,$header,$header2, "-r".$from_mail);  
  
$mpdf->Output($filename , 'I');  
exit;  
  
?>
```

Note: Submitted by a user. See [message](#) in the Forum.

A5 Booklet

This script was written to create a new PDF file based on a pre-existing PDF document, converting an A4 document into an A5 booklet ready for duplex printing. Page order is adjusted, and page orientation is rotated so that it prints a landscape booklet.

```
<?php

include("../mpdf.php");

$mpdf=new mPDF('','A4-L','','',0,0,0,0,0,0);
$mpdf->SetImportUse();
$ow = $mpdf->h;
$oh = $mpdf->w;
$pw = $mpdf->w / 2;
$ph = $mpdf->h;

$mpdf->SetDisplayMode('fullpage');

$pagecount = $mpdf->SetSourceFile('A4sourcefile.pdf');
$pp = GetBookletPages($pagecount);

foreach($pp AS $v) {
    $mpdf->AddPage();
    if ($v[0]>0 && $v[0]<=$pagecount) {
        $tplIdx = $mpdf->ImportPage($v[0], 0,0,$ow,$oh);
        $mpdf->UseTemplate($tplIdx, 0, 0, $pw, $ph);
    }
    if ($v[1]>0 && $v[1]<=$pagecount) {
        $tplIdx = $mpdf->ImportPage($v[1], 0,0,$ow,$oh);
        $mpdf->UseTemplate($tplIdx, $pw, 0, $pw, $ph);
    }
}

$mpdf->Output();

exit;

function GetBookletPages($np, $backcover=true) {
    $lastpage = $np;
    $np = 4*ceil($np/4);
    $pp = array();
    for ($i=1; $i<=$np/2; $i++) {
        $p1 = $np - $i + 1;
        if ($backcover) {
            if ($i == 1) { $p1 = $lastpage; }
            else if ($p1 >= $lastpage) { $p1 = 0; }
        }
        if ($i % 2 == 1) {
            $pp[] = array( $p1, $i );
        }
        else {
            $pp[] = array( $i, $p1 );
        }
    }
    return $pp;
}

?>
```

Reserving x blank pages

One user wanted to reserve several blank pages on the left for advertisements, whilst flowing the document on the other pages.

There isn't a 'proper' way to do it, but a few lines of script in `mpdf.php` may work:

About 20 lines above `function PageNo()` - at the end of `function AddPage()` - find:

```
//RESET BLOCK BORDER TOP
if (!$this->ColActive) {
    for($bl=1;$bl<=$this->blklvl;$bl++) {
        $this->blk[$bl]['y0'] = $this->y;
```

Just before this, insert

```
if ($this->page % 2 == 0 && $this->page < 5) {
    $this->AddPage();
    $this->MarginCorrection = 0;
}
```

You can obviously change:

`$this->page % 2 == 1` will leave odd pages blank

`$this->page < 5` will leave blank alternate pages up to and including page number 4

Letterhead letters

Using the CSS @page selector, these are a couple of similar methods for producing multiple letters into 1 PDF file, where each letter uses a letterhead for page one then plain paper for the remaining pages.

Method 1

```
$header = '
<!--mpdf
<htmlpageheader name="letterheader">
<table width="100%" style=" font-family: sans-serif;"><tr>
<td width="50%" style="color:#0000BB; "><span style="font-weight: bold; font-size: 14pt;">Acme
Trading Co.</span><br />123 Anystreet<br />Your City<br />GD12 4LP<br /><span style="font-size:
15pt;">@</span> 01777 123 567</td>
<td width="50%" style="text-align: right; vertical-align: top;">Invoice No.<br /><span style="font-
weight: bold; font-size: 12pt;">0012345</span></td>
</tr></table>
<div style="margin-top: 1cm; text-align: right; font-family: sans-serif;">{DATE jS F Y}</div>
</htmlpageheader>

<htmlpagefooter name="letterfooter2">
<div style="border-top: 1px solid #000000; font-size: 9pt; text-align: center; padding-top: 3mm;
font-family: sans-serif; ">
Page {PAGENO} of {nbpg}
</div>
</htmlpagefooter>

mpdf-->

<style>
@page {
margin-top: 2.5cm;
margin-bottom: 2.5cm;
margin-left: 2cm;
margin-right: 2cm;
header: html_letterheader;
footer: _blank;
resetpagenum: 1;
background-color: pink;
}

@page :first {
margin-top: 8cm;
margin-bottom: 4cm;
header: html_letterheader;
footer: _blank;
resetpagenum: 1;
background-color: lightblue;
}
@page letterhead {
margin-top: 2.5cm;
margin-bottom: 2.5cm;
margin-left: 2cm;
margin-right: 2cm;
header: html_letterheader;
footer: html_letterfooter2;
background-color: pink;
}

@page letterhead :first {
margin-top: 8cm;
margin-bottom: 4cm;
header: html_letterheader;
footer: _blank;
resetpagenum: 1;
background-color: lightblue;
}
</style>
';

$letter = 'Dear Sir or Madam,<br />
Contents of your letter...
<pagebreak />
... more letter on page 2 ...'
```

```

<pagebreak />
... more letter on page 3 ...

';

$mpdf->WriteHTML($header);

$mpdf->WriteHTML($letter);
$mpdf->WriteHTML('<pagebreak page-selector="letterhead" />');
$mpdf->WriteHTML($letter);
$mpdf->WriteHTML('<pagebreak page-selector="letterhead" />');
$mpdf->WriteHTML($letter);

$mpdf->Output();

```

Method 2

```

$header =
<!--mpdf
<htmlpageheader name="letterheader">
<table width="100%" style=" font-family: sans-serif;"><tr>
<td width="50%" style="color:#0000BB; "><span style="font-weight: bold; font-size: 14pt;">Acme
Trading Co.</span><br />123 Anystreet<br />Your City<br />GD12 4LP<br /><span style="font-size:
15pt;">@</span> 01777 123 567</td>
<td width="50%" style="text-align: right; vertical-align: top;">Invoice No.<br /><span style="font-
weight: bold; font-size: 12pt;">0012345</span></td>
</tr></table>
<div style="margin-top: 1cm; text-align: right; font-family: sans-serif;">{DATE jS F Y}</div>
</htmlpageheader>

<htmlpagefooter name="letterfooter2">
<div style="border-top: 1px solid #000000; font-size: 9pt; text-align: center; padding-top: 3mm;
font-family: sans-serif; ">
Page {PAGENO} of {nbpg}
</div>
</htmlpagefooter>

$mpdf-->

<style>
@page {
margin-top: 2.5cm;
margin-bottom: 2.5cm;
margin-left: 2cm;
margin-right: 2cm;
footer: html_letterfooter2;
background-color: pink;
}

@page :first {
margin-top: 8cm;
margin-bottom: 4cm;
header: html_letterheader;
footer: _blank;
resetpagenum: 1;
background-color: yellow;
}
@page letterhead :first {
margin-top: 8cm;
margin-bottom: 4cm;
header: html_letterheader;
footer: _blank;
resetpagenum: 1;
background-color: lightblue;
}
.letter {
page-break-before: always;
page: letterhead;
}
</style>
';

```

```
$firstletter = '<div>Dear Sir or Madam,<br />
Contents of your letter...
<pagebreak />
... more letter on page 2 ...
<pagebreak />
... more letter on page 3 ...
</div>
';

$letter = '<div class="letter">Dear Sir or Madam,<br />
Contents of your letter...
<pagebreak />
... more letter on page 2 ...
<pagebreak />
... more letter on page 3 ...
</div>
';

$mpdf->WriteHTML($header);

$mpdf->WriteHTML($firstletter);
$mpdf->WriteHTML($letter);
$mpdf->WriteHTML($letter);

$mpdf->Output();
```

Creating a PDF file from User Input

These scripts allow you to present a form to the user, who can enter text and upload an image; these are displayed first in the browser, with the option to create a PDF file from the output. These scripts should only be considered the basis of a full script and will need adapting considerably. In particular, note that the uploaded image files may need to be deleted at some point.

example_userinput.php

```
<?php

$html = '
<html>
<body>
<form action="example_userinput2.php" method="post" enctype="multipart/form-data">
Enter text:
<br />
<textarea name="text" id="text"></textarea>
<br />
<label for="file">Choose Image to upload:</label> <input type="file" name="file" id="file" />
<br />
<input type="submit" name="submit" value="Submit" />
</form>

</body>
</html>
';

echo $html;

exit;

?>
```

example_userinput2.php

```
<?php

if (($_FILES["file"]["type"] == "image/gif" || $_FILES["file"]["type"] == "image/jpeg")
&& $_FILES["file"]["size"] < 20000)  {
    // If the destination file already exists, it will be overwritten
    move_uploaded_file($_FILES["file"]["tmp_name"], "../tmp/" . $_FILES["file"]["name"]);
}
else {
    echo "Invalid file";
}

$html =' 
<html>
<body>
<div>' . $_POST['text'] . '</div>


<form action="example_userinput3.php" method="post" enctype="multipart/form-data">
<textarea style="display:none" name="text" id="text">' . $_POST['text'] . '</textarea>
<input type="hidden" name="filename" id="filename" value="' . $_FILES["file"]["name"] . '" />
<input type="submit" name="submit" value="Create PDF file" />
</form>

</body>
</html>
';

echo $html;

exit;
```

```
?>
```

```
example_userinput3.php
```

```
<?php

define('_MPDF_PATH', '../');

include("../mpdf.php");
$mpdf=new mPDF(' ');

$html =
<html>
<body>
<div>' . $_POST['text'] . '</div>

</body>
</html>
';

$mpdf->WriteHTML($html);

$mpdf->Output();

exit;

?>
```

WRITE DIRECTLY TO DOCUMENT

Direct writing to document

Other Methods

mPDF is optimised to accept HTML code and CSS styles. Apart from WriteHTML() there are other methods that can be used to write to the PDF document, but these do not always have full functionality. These are methods available in the original FPDF and successors, and if these are all you are using you may find that you do not need to use mPDF with its extra functions that slow the program down - see FPDF and UFPDF for further details.

The methods Cell() and Text() from FPDF are still present, but should not be used directly as they will not cope with UTF-8 encoded text. Use the WriteCell() and WriteText() methods instead.

All the methods described below handle UTF-8 encoded text, and all but AutosizeText() and watermark() will reverse RTL (right-to-left) text when appropriate.

Direct writing methods and OTL (updated: mPDF >= 6.0)

- WriteText() WriteCell() Watermark() AutoSizeText() and ShadedBox() DO support complex scripts and right-to-left text (RTL).
- Write() does NOT support complex scripts or RTL (NB this is a change - Write() used to support RTL).
- CircularText() does NOT support complex scripts or RTL.
- MultiCell() DOES support complex scripts and RTL, but complex-script line-breaking MAY NOT be accurate.
- MultiCell() does not support kerning and justification. NB This includes <textarea> in forms which uses MultiCell() internally.
- <select> form objects also do NOT support kerning.

Text containing HTML entities, as well as decimal and hex e.g. & apos; & #8812; or & #x21a4; can be used in all of these methods, by setting:

```
$mpdf->text_input_as_HTML = true; (default = false)
```

This will convert all the above to their appropriate characters, otherwise the text will be output as it is.

For most of the methods, you are referred to the originals at [FPDF >> Manual](#) for more information.

WriteCell(float w, float h, string text[, mixed border[, integer ln[, string align[, integer fill[, mixed link[, float returnx]]]]]])
--

Writes a single line of text directly to the PDF document at the current position.

See the details for Cell() at FPDF. An additional parameter returnx has been added; if ln is set, the current position moves not to the left margin, but to the value set as returnx.

WriteText(float w, float h, string text)

Writes a single line of text directly to the PDF document at a specified position.

See the details for Text() at FPDF.

MultiCell(float w, float h, string text[, mixed border[, string align[, integer fill[, mixed link[, string directionality[, boolean encoded]]]]]])

Writes a block of text directly to the PDF document at the current position. Lines are wrapped at the margins.

See the details for MultiCell() at FPDF. Two additional parameters have been added:

- directionality
Set to 'rtl' if using RTL language (right-to-left)
Default = 'ltr'
- encoded
When set to false (default), UTF-8 encoded text will be appropriately converted to the chosen output file format. It should only be set to true, when the input text has already been converted internally within the program.
Default = false

```
SetX(float x)
SetY(float y)
SetXY(float x, float y)
```

Sets the co-ordinates for the current position to write. Note only milimeters can be used as units. X and Y are measured from the top-left corner of the page.

See the details these methods at FPDF

```
AutosizeText(string text, float width, string font, string style[, float fontsize])
```

Writes a single line of text directly to the PDF document at the current position.

Font size will be automatically reduced to fit width (but is not increased).

NB Does not reverse RTL text

- text
UTF-8 encoded text to write. Single line only.
- width
Width of text in millimeters. The font size will be reduced if required to fit this size.
- font
Font family to use
- style
Font style used [blank for normal]|i|b|bi
- fontsize
Maximm font size in points (pt)
Default = 72

```
watermark(string text[, float angle[, float fontsize[, float alpha]]])
```

Writes a single line of text centred on the page, which can be rotated and transparent i.e. a watermark.

NB Does not reverse RTL text.

- text
UTF-8 encoded text
- angle
Angle of rotation for text (anticlockwise rotation from horizontal)
Default = 45
- fontsize
Font size in points (pt)
Default = 96
- alpha
Transparency
Default = 0.2

```
RoundedRect(float x, float y, float w, float h, float radius[, string style])
```

Draws a rectangle with rounded corners directly to the PDF document at the specified position.

- x
Abscissa of left edge of box - value in millimeters
- y
Ordinate of top edge of box - value in millimeters
- w
Width of the box - in millimeters
- h
Width of the box - in millimeters
- radius
Radius of the rounded corners
- style
Box style: D or empty string - draw border (default); F - fill; DF or FD - draw and fill
Default = " i.e. border, no fill

```
shaded_box(string title[, string font[, float fontstyle[, float fontsize[, float width[, string style[, float radius[, string backgroundcolor[, string color[, float padding]]]]]]]]])
```

Writes a single line of text surrounded by a box directly to the PDF document at the current position.
The box can have rounded corners, and be filled with background-colour.

- **title**
UTF-8 encoded text (single line)
- **font**
Font family
Default = " i.e. default document font
- **fontstyle**
Font style as one of B (bold), I (italic), BI (bold-italic) or blank for normal
Default = 'B' i.e. bold
- **fontsize**
Font size in points (pt)
Default = " i.e. default document font size
- **width**
Width of the box - any units acceptable in CSS can be used e.g. pt, px, mm, % (of page width)
Default = '70%'
- **style**
Box style: D or empty string - draw border (default); F - fill; DF or FD - draw and fill
Default = 'DF' i.e. border and fill
- **radius**
Radius of the rounded corners
Default = 2.5
- **backgroundcolor**
Fill colour for the box - as #rrggbb
Default = '#FFFFFF'
- **color**
Text colour - as #rrggbb
Default = '#000000'
- **padding**
Padding between text and box border, in millimeters
Default = 2

See Also

- [preparePreText\(\)](#) - Prepares text to be output ignoring the HTML markup

TROUBLESHOOTING

Known Issues

Crashing with no error message

Ensure mbregex is enabled as part of mb_string. Apparently this is enabled by default when you enable mbstring in most cases, however with cPanel and some other non-standard environments this might not be the case, so people have to explicitly look for and enable mbregex (i.e. compile PHP with `--enable-mbregex`).

Blank pages or some sections missing

If you pass a large chunk of code to WriteHTML() whether as CSS styles or the main HTML code, you may get a blank page output, or that section of code missing.

The PHP function preg_replace() has a maximum string length it will parse (by default this is often about 100000 characters). Over this, PHP silently returns a null value. So long strings of code will be replaced by nothing!

You may be able to increase the value of `pcre.backtrack_limit` at runtime if your system allows; alternatively, break your HTML into chunks and pass them one at a time to WriteHTML()

`pcre.backtrack_limit` is configurable from PHP $\geq 5.2.0$

The default value was increased from 100,000 to 1,000,000 from PHP $\geq 5.3.7$

```
ini_set("pcre.backtrack_limit","1000000");
```

Keep-with-table

If `use_kwt ("keep-with-table")` is set, and a heading element precedes a table inside a div with border/background set: it doesn't work e.g.

```
<div style="border: 1px solid #000000; background-color: #EEEEFF;"><h2>Title</h2><table...>
```

Program dies with no error message when generating a large PDF file

A timeout due to Apache configuration 'TimeOut' will cause the script to terminate with no error message, despite increasing the PHP time limit etc.

See also [Blank screen](#) for a bug when using localhost

Problems fixed from mPDF ≥ 5.0

Indic Fonts - ASCII characters

The Indic fonts (added mPDF 4.0) do not contain the basic ASCII characters: a-z, A-Z, and in some: ` and \$

The font files have been edited to add these characters if you are using embedded font subsets (so all ASCII chars show), but they will not be available if you are not using subsets. In this case you need to mark up HTML text with `lang` or `font-family`.

Adobe Reader 7 error reading file with embedded SVG image

With some SVG images, Adobe Reader 7 throws an error - "Problem with Type 3 font, form or pattern".

See Also

- [Limitations](#)

Slow!

mPDF is quite slow. Large tables seem to be the biggest burden, especially if they have borders. The other thing that is slow is rewriting sections repeatedly to automatically re-size them e.g. a block element with position: fixed and overflow: auto

On the other hand, I have used mPDF to produce a 400 page book, complete with a few images, 40 or so small tables, a table of contents and Index in approx 90 secs.

Consider the following:

- make sure you have upgraded to the latest version of mPDF
- turn off error_reporting() in PHP, including error_logging
- set \$mpdf->useSubstitutions=false;
- avoid using .gif image files if you do not have the GD library installed as part of PHP
- set \$mpdf->simpleTables = true; if you do not need complex table borders (mPDF >= 4.3)
- consider if you can use core PDF fonts rather than embedded fonts
- avoid using CSS "page-break-inside:avoid"

Images

JPG images are quickest, as the data requires no processing beyond extracting the metadata (width, height, colour-space etc.).

PNG images with no alpha channel and not interlaced are next best.

GIF files are extremely slow if you do not have the appropriate GD library (above 2.0.8). With GD they are quite fast.

Note that any images that require the GD library are processed quite fast - but use a large amount of memory - to create a GD image in memory can use up to 10 x the file size (e.g. a 690K GIF file read into imagecreatefromstring() used about 5MB of PHP memory). PNG images with alpha channel transparency or interlaced, require the GD library to process them.

Depending on your setup, you may find that performance can be improved a lot by using relative paths for images rather than absolute paths e.g.

Other things which are slow

Tables - the more rows and columns the worse it gets; if the size of the table requires it to be resized to fit on the page, this takes longer.

Using "overflow: auto" on fixed-position divs, as it tries to rewrite the data repeatedly until it finds a best fit.

Text is replaced

If some of your text is unexpectedly replaced by numbers or date-strings, check Replaceable Aliases.

Reserved Terms

If some of your text is unexpectedly replaced by numbers or date-strings, check Replaceable Aliases.

Browser incompatability

If you wish to pass the same HTML to a browser as to mPDF, and it is showing incorrectly in the browser, please see [Custom tags](#).

Error messages

"Output has already been sent from the script - PDF file generation aborted."

If you see this message it means that the script has sent output to the browser before starting to generate the PDF file.

Most likely causes are:

- a PHP error message - this should be displayed in your browser giving details of the problem
- inadvertent whitespace in your PHP script files e.g. leaving space before or after the PHP tags <?php ?>
- you are using object_buffering to generate content for your PDF file - see below

If no error message appears, try setting:

```
$mpdf->debug = true;
```

Object buffering

In order to catch error messages and prevent them being included in a PDF file (which will be corrupted), mPDF 2.5 introduced a method to detect whether there had been any output from the script prior to generating the PDF file in Output(). This includes checking for ob_get_contents() - a PHP function to check if there is any output in the object-buffer.

If you use object_buffering in the process of preparing the text for mPDF, this will falsely trigger the error message. If this is the case, add the following to your script to prevent it:

```
$mpdf->allow_output_buffering = true;
```

Blank screen

If you get nothing but a blank screen on your browser, it may be because there is a script error. Turn on debugging at the start of your script:

```
<?php  
include("../mpdf.php");  
$mpdf=new mPDF();  
  
$mpdf->debug = true;  
  
$mpdf->WriteHTML("Hallo World");  
$mpdf->Output();  
?>
```

See also [Known issues](#) and [Corrupt PDF file](#).

localhost problems / Vista 64 bit

Since updating to Windows Vista x64bit for development, all files containing images or external stylesheets crashed or timeout.

See the useful thread at: <http://www.wampserver.com/phorum/read.php?2,28291>

To generate images and retrieve external stylesheets, mPDF needs to access files using fopen() etc. Changing "localhost" to "127.0.0.1" resolved the problem.

Generating graphs with JGraph

JGraph outputs error messages as images generated by the script. If the error message is "out of memory" then you just get a blank screen. Try increasing your memory_limit.

Corrupt PDF file

If you get a message saying "Corrupt PDF file: does not start with %PDF", it may be because your script - either mPDF or an error in your PHP code - has output an error message. The browser is expecting a file in PDF format, which should start with "%PDF" and instead it gets some text like an error message.

To show error message(s):

```
<?
$mpdf=new mPDF();
$mpdf->debug = true;
...
$mpdf->Output();
?>
```

Adding exit

A number of errors can be caused by not explicitly ending your script with `exit;`

Notice warnings

If you get something like:

Notice: Undefined index: win-1252 in D:\Program Files\Zend\Apache2\htdocs\mpdf\mpdf.php on line 3741

you need to suppress NOTICE warnings using e.g.

```
error_reporting(E_ALL ^ E_NOTICE);
// or
error_reporting(0);
```

Include this at the start of your script before including the mpdf.php file.

Image errors

If images cannot be displayed correctly, you should see a  on the page. Please see [Images](#) for further information, and a method to debug the problem.

Background-images will degrade silently ie. they do not show the  icon.

Image files containing spaces may cause problems. Change the spaces to %20 using e.g. `urlencode()`

Memory problems

mPDF uses a lot of memory on the server. If you get an error message that you have exceeded your memory limits, try the following:

- it is more efficient in very long documents to process the HTML code in small chunks rather than as one large HTML string
- use `ini_set("memory_limit", "128M")` or similar at the top of your script to allocate more memory
- generate a smaller `mpdf.php` file - see [Reducing Memory Usage](#)
- avoid using GIF image files, or PNG images with the alpha channel (transparency) or interlaced PNG images
- set `simpleTables` if you are using large tables and do not require complex borders or padding
- setting `packTabledata` can save considerably on memory usage, BUT at a significant cost in processing time
- setting `cacheTables = TRUE` makes the most efficient use of memory, but requires temporary data written to files (which may be resource intensive) and will add significantly to processing time
- use a limited number of fonts, avoiding large font file sizes

PHP 5.3.x on Windows

Memory can become exhausted rapidly when running PHP 5.3.x on Windows. I believe this may be a bug in the Windows version of PHP. A script that exhausts 256Mb memory on Windows may only use 18Mb when run on Linux. It appears to happen most often (or exclusively) when using tables.

So if you are only using Windows in a test environment and use Linux for production, you should consider setting the memory limit to maximum on Windows e.g. `ini_set("memory_limit", "-1")`.

Small changes to your code e.g. the CSS applied to table elements can have dramatic, but unpredictable effects on the memory usage. For example changing an attribute on a table cell to a css style.

Otherwise you will need to refer to the tips above, especially `simpleTables`, `packTabledata`, or `cacheTables`

One user has reported this problem disappearing when updating from 5.3.8 to PHP 5.3.10, so this may be a bug that is fixed?

Resizing

mPDF automatically resizes content in some circumstances. Using the default settings and CSS properties, the following are resized:

- tables will resize so that the tallest row (cell) will fit on a page [this is the only one that cannot be overridden]
- tables will resize to fit the minimum width into the available width (minimum width of a table is when no words are broken) - override using CSS `<table style="overflow: hidden|visible|wrap">`
- tables will resize to fit the table into the remaining available space left on a page, as long as it is within the limit set by configurable variable `$this->shrink_tables_to_fit` - this can be prevented by setting this value to "1" in config.php or `<table autosize="1">`
- block elements (e.g. `<div>`) with `position:fixed` or `position:absolute` and `overflow:auto` (the default) will resize the contents if required to fit on the page - override by changing the value of `overflow`
- images will resize if necessary to fit onto a page

Tables may also resize if you set the CSS property `page-break-inside: avoid`

mPDF class fails to Initialise

mPDF still retains use of a constructor function with the same name as the class i.e.

```
class mPDF {  
    ...  
    function mPDF(...)
```

This means it is still compatible with PHP4, and is backwards compatible using PHP5 - **UNLESS** you are using namespaces in PHP >= 5.3

In this case you will need to rename:

```
function mPDF(...){
```

as

```
__construct(...){
```

Postscript printers

If you have problems with PDF files printing on PostScript printers, this may be helped by setting the configurable variable in config.php:

```
$this->repackageTTF = false;
```

This forces mPDF to remake the font file using only core tables when Embedding full TTF font files.

MPDF FUNCTIONS

mPDF()

(mPDF >= 5.0)

mPDF – Initialise an instance of mPDF class

Description

```
class mPDF ([ string $mode [, mixed $format [, float $default_font_size [, string $default_font [, float $margin_left , float $margin_right , float $margin_top , float $margin_bottom , float $margin_header , float $margin_footer [, string $orientation ]]]]]])
```

Initialise an instance of mPDF class.

Parameters

mode

This parameter specifies the mode of the new document.

DEFAULT: BLANK

Codepage Values (case-insensitive)

BLANK

"c"
"...-x"
"...-s" or "s"
"...+aCJK" or "+aCJK"
"...-aCJK" or "-aCJK"

where ... can be any string. Only language/country codes will have any effect, but other strings are parsed for backwards compatibility (but have no effect).

Only some combinations make sense. See [Choosing a configuration](#) for more details.

Country/Language code values (case-insensitive)

Country/language codes are defined in config_cp.php

A country/language code can be passed as e.g. "en-GB" or "en_GB" or "en"

Note: If the *mode* is set by passing a country/language string, this may also set: available fonts, text justification, and directionality **RTL** (as determined by config_cp.php)

Note: There is a useful list of language/country codes at:

<http://www.i18nguy.com/unicode/language-identifiers.html>

format

format can be specified either as a pre-defined page size, or as an array of width and height in millimetres (see Example #2 below).

DEFAULT: "A4"

Values (case-insensitive)

A0 - A10, B0 - B10, C0 - C10
4A0, 2A0, RA0 - RA4, SRA0 - SRA4
Letter, Legal, Executive, Folio

Demy, Royal
 A (Type A paperback 111x178mm)
 B (Type B paperback 128x198mm)
 Ledger, Tabloid*

All of the above values can be suffixed with "-L" to force a Landscape page orientation document e.g. "A4-L".

If *format* is defined as a string, the final *orientation* parameter will be ignored.

*Ledger and Tabloid are standard formats with the same page size but different orientation (Ledger is landscape, and Tabloid is portrait). mPDF treats these identically; if you wish to use Ledger, you should specify "Ledger-L" for landscape.

default_font_size

Sets the default document font size in **points** (pt)

BLANK or omitted or 0 uses the default value set in *defaultCSS*.

default_font

Sets the default font-family for the new document.

BLANK or omitted uses default value set in *defaultCSS* unless *codepage* has been set to "win-1252". If *codepage*="win-1252", the appropriate core Adobe font will be set i.e. Helvetica, Times, or Courier.

margin_left *margin_right* *margin_top* *margin_bottom* *margin_header* *margin_footer*

Sets the page margins for the new document.

All values should be specified as **LENGTH** in millimetres.

If you are creating a **DOUBLE-SIDED** document, the margin values specified will be used for **ODD** pages; left and right margins will be mirrored for **EVEN** pages.

BLANK or omitted uses the default values.

DEFAULT Values

margin_left 15
margin_right 15
margin_top 16
margin_bottom 16
margin_header 9
margin_footer 9

orientation

This attribute specifies the default page orientation of the new document if *format* is defined as an array. This value will be ignored if *format* is a string value.

DEFAULT: "P"

Values (case-insensitive)

P: **DEFAULT** Portrait

L: Landscape

Changelog

Version	Description
2.0	The <i>orientation</i> parameter was added.
5.0	The <i>mode</i> parameter renamed (from <i>codepage</i>), and recognised values changed

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF();
$mpdf->WriteHTML('<p>Hello World</p>');
$mpdf->Output('filename.pdf');
exit;

?>
```

Example #2

```
// Define a new mPDF document using utf-8 fonts
$mpdf=new mPDF('utf-8');

// Define a new mPDF document using win-1252 fonts based on a language/country code
$mpdf=new mPDF('en-GB');

// Define a Landscape page size/format by name
$mpdf=new mPDF('utf-8', 'A4-L');

// Define a page size/format by array - page will be 190mm wide x 236mm height
$mpdf=new mPDF('utf-8', array(190,236));

// Define a page using all default values except "L" for Landscape orientation
$mpdf=new mPDF('', '', 0, '', 15, 15, 16, 16, 9, 9, 'L');
```

Notes

Note: `_MPDF_PATH` was required to be defined explicitly prior to mPDF 4.0 e.g.
`define('_MPDF_PATH','..').` From mPDF 4.0 the value should be automatically defined by the script itself when including the `mpdf.php` file.

See Also

- [WriteHTML\(\)](#) - Write HTML to the document
- [Output\(\)](#) - Finalise and output the document

Overview

mPDF Functions by Category

Main controls

[mPDF](#) - Main class constructor

[mPDFI](#) - [Depracated] - see [SetImportUse\(\)](#)

[WriteHTML](#) - Write HTML code to the document

[Output](#) - Finalise the document and send it to specified destination

Configuration

[SetImportUse](#) - Enables use of templates and imported PDF files (was mPDFI)

[StartProgressBarOutput](#) - Show progress bars whilst generating PDF file

[SetAnchor2Bookmark](#) - Specifies how PDF Book marks are created from HTML anchors

[SetBasePath](#) - Specifies a base URL for mPDF to interpret relative URLs

[SetCompression](#) - Specifies that mPDF should compress the data for the PDF file

[SetDefaultFont](#) -

[SetDefaultBodyCSS](#) -

[SetDefaultFontSize](#) -

[SetDirectionality](#) -

[SetDisplayMode](#) - Specify the initial Display Mode when the PDF file is opened

[SetAutoFont](#) -

[RestrictUnicodeFonts](#) -

[SetUserRights](#) -

Document Metadata

[SetTitle](#) - Set the document title

[SetAuthor](#) - Set the document author

[SetCreator](#) - Set the document creator

[SetSubject](#) - Set the document subject

[SetKeywords](#) - Set the document keywords

Encryption & Passwords

[SetProtection](#) - Encrypts and sets the PDF document permissions

[SetUserRights](#) -

Watermarks (page backgrounds)

[SetWatermarkImage](#) - Set an image to use as a Watermark

[SetWatermarkText](#) - Set the text to use as a Watermark

Page Numbering

[AliasNbPages](#) -Defines the placeholder used to insert total number of pages into the document

[AliasNbPageGroups](#) -Defines the placeholder used to insert total page number into the document

Colours

[AddSpotColor](#) - Define a Spot Colour to be referenced in the document

The following methods have HTML equivalents e,g, <pagebreak /> and AddPage():

Paging

[AddPage](#) - Add a new page

[AddPageByArray](#) - Add a new page using array of parameters

Bookmarks (Outlines)

[Bookmark](#) -Add a Bookmark to the document

Annotations

[Annotation](#) -Add an Annotation to the document

Index

[CreateIndex](#) -Generate an Index for the document (mPDF < v6.0)

[InsertIndex](#) - Generate an Index for the document (mPDF >= v6.0)

[IndexEntry](#) -Insert an Index entry for the document

[IndexEntrySee](#) - Insert a cross-reference entry for the document Index

Columns

[SetColumns](#) -Control the use of multiple columns on the page

[AddColumn](#) - Start a new Column

Page Headers & Footers

[DefFooterByName](#) - Define a page footer by name

[DefHeaderByName](#) - Define a page header by name

[DefHTMLFooterByName](#) - Define an HTML page footer by name

[DefHTMLHeaderByName](#) - Define an HTML page header by name

[SetFooter](#) - Set a page footer

[SetFooterByName](#) - Set a page footer by name

[SetHeader](#) - Set a page header

[SetHeaderByName](#) - Set a page header by name

[SetHTMLFooter](#) - Set an HTML page footer

[SetHTMLFooterByName](#) - Set an HTML page footer by name

[SetHTMLHeader](#) - Set an HTML page header

[SetHTMLHeaderByName](#) - Set an HTML page header by name

Table of Contents

[TOCpagebreak](#) - Insert a table of contents in the document

[TOCpagebreakByArray](#) - Insert a table of contents in the document using an array of parameters

[TOC_Entry](#) - Insert an entry for the Table of Contents

Other Miscellaneous methods:

Replacing text in existing PDF files

[OverWrite](#) - Replace specified text strings in an existing PDF file

Control Visibility

[SetVisibility](#) - Control the visibility of subsequent objects

Import PDF files & Templates

[mPDFI](#) - [Depracated] - see [SetImportUse\(\)](#)

[Thumbnail](#) - Print thumbnails of an external PDF file

[SetSourceFile](#) - Specify the source PDF file used to import pages into the document

[ImportPage](#) - Import a page from an external PDF file

[UseTemplate](#) - Insert an imported page from an external PDF file

[SetPageTemplate](#) - Specify a page from an external PDF file to use as a template

[SetDocTemplate](#) - Specify an external PDF file to use as a template

Barcode

[WriteBarcode](#) - Write an EAN-13 (ISBN-13) barcode

Write directly to page without HTML

[AutosizeText](#) -

[RoundedRect](#) -

[WriteCell](#) -

[WriteText](#) -

[Shaded_box](#) -

[CircularText](#) - NB This does have an HTML equivalent as <textcircle>

AddColumn()

(mPDF >= 2.2)

AddColumn – Start a new Column

Description

```
void AddColumn ( )
```

Start a new Column in the document. Columns must be set using [SetColumns\(\)](#) or [<columns>](#). Height justification for the Columns is disabled when column breaks are set explicitly.

Note: Columns are incompatible with (and automatically disable): borders for block-level elements (DIV, P etc), table rotation, and collapsible margins for blocks e.g. top and bottom margins for a DIV will not collapse (default) at the top/bottom of a column.

Parameters

No parameters

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetColumns(2);  
$mpdf->WriteHTML('<p>Some text...</p>');  
$mpdf->AddColumn();  
$mpdf->WriteHTML('<p>Next column...</p>');  
$mpdf=Output();  
  
?>
```

See Also

- [<columnbreak>](#) - HTML equivalent to AddColumn()
- [SetColumns\(\)](#) - Control the use of multiple columns on the page
- [<columns>](#) - Control the use of multiple columns on the page

AddPage()

(mPDF \geq 1.0)

AddPage – Add a new page

Note: A new function `AddPageByArray()` was added in mPDF 5.0 which is recommended as much simpler to use.

Description

```
void AddPage ([ string $orientation [, string $type [, string $resetpagenum [, string $pagenumstyle [,  
string $suppress [, float $margin-left [, float $margin-right [, float $margin-top [, float $margin-bottom [, float  
$margin-header [, float $margin-footer [, string $odd-header-name [, string $even-header-name [, string  
$odd-footer-name [, string $even-footer-name [, mixed $odd-header-value [, mixed $even-header-value [,  
mixed $odd-footer-value [, mixed $even-footer-value [, string $pageselector [, mixed $sheet-size  
]]]]]]]]]]]]]]])
```

Add a new page to the document. The parameter `type` can specify certain conditions which determine how many pages are added. If writing a **DOUBLE-SIDED** document, a conditional page-break (`type="E"` or `"O"`) will add a new page only if required to make the current page match the type (i.e. **ODD** or **EVEN**); a page-break with `type="NEXT-ODD"` or `"NEXT-EVEN"` will add one or two pages as required to make the current page match the type (i.e. **ODD** or **EVEN**).

Number of pages added:

		DOUBLE-SIDED	
type	SINGLE-SIDED	Currently ODD page	Currently EVEN page
BLANK	1	1	1
O or ODD	0	0	1
E or EVEN	0	1	0
NEXT-ODD	1	2	1
NEXT-EVEN	1	1	2

Note: If no new page is added, the other parameters will be ignored e.g. resetting page numbers/styles, margins and headers/footers. If 2 pages are added, the changes in page numbers/styles, margins and headers/footers will start on the final added page.

Note: From mPDF >= 3.0 the page numbering can be reset to any positive number. Prior to this, it was only possible to reset it to 1.

Parameters

orientation = L|P

This attribute specifies the orientation of the new page.
BLANK or omitted leaves the current orientation unchanged

Values (case-insensitive)
L or landscape: Landscape
P or portrait: Portrait

type = E|O|even|odd|next-odd|next-even

If type is specified as "E" or "O" when writing a **DOUBLE-SIDED** document, the page-break is conditional; a new page will only be added if necessary to meet the specified condition.
 If type is specified as "NEXT-ODD" or "NEXT-EVEN" when writing a **DOUBLE-SIDED** document, either one or two pages are added as necessary to meet the specified condition.
 If not writing a **DOUBLE-SIDED** document, a page-break type="E" or "O" will be ignored.
BLANK or omitted will force a new page unconditionally.

Values (case-insensitive)

O or ODD: Add a new page if required to make current page an **ODD** one.
 E or EVEN: Add a new page if required to make current page an **EVEN** one.
 NEXT-ODD: Add one or two pages as required to make the current page **ODD**.
 NEXT-EVEN: Add one or two pages as required to make the current page **EVEN**.

resetpagenum = 1 - ∞

Sets/resets the document page number to *resetpagenum* starting on the new page. (The value must be a positive integer).
BLANK or omitted or 0 leaves the current page number sequence unchanged.

pagenumstyle = 1|A|a|i

Sets/resets the page numbering style (values as for lists)
BLANK or omitted leaves the current page number style unchanged.

Values (case-sensitive)

1: Decimal - 1,2,3,4...
 A: Alpha uppercase - A,B,C,D...
 a: Alpha lowercase - a,b,c,d...
 I: Roman uppercase - I, II, III, IV...
 i: Roman lowercase - i, ii, iii, iv...

suppress = on|off|1|0

suppress=on will suppress document page numbers from the new page onwards (until *suppress=off* is used)
BLANK or omitted leaves the current condition unchanged.

Values (case-insensitive)

1 or on: Suppress (hide) page numbers from the new page forwards.
 0 or off: Show page numbers from the new page forwards.

margin-left
margin-right
margin-top
margin-bottom
margin-header
margin-footer

Sets the page margins from the new page forwards.

All values should be specified as **LENGTH** in millimetres.

If you are writing a **DOUBLE-SIDED** document, the margin values will be used for **ODD** pages; left and right margins will be mirrored for **EVEN** pages.

BLANK or omitted leaves the current margin unchanged. NB "0" (zero) will set the margin to zero.

odd-header-name
even-header-name

odd-footer-name
even-footer-name

Selects a header or footer by name to use from the new page forwards. The header/footer must already have been defined using [DefHeaderByName\(\)](#), [DefFooterByName\(\)](#), [DefHTMLHeaderByName\(\)](#), or [DefHTMLFooterByName\(\)](#).

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted leaves the header/footer unchanged. NB **BLANK** will not unset the header. Set *odd-header-value* to -1 to turn the header off.

Note: You must add the prefix 'html_' before the name if it is a HTMLHeader.

odd-header-value
even-header-value
odd-footer-value
even-footer-value

Specify whether to show or hide the named header or footer from the new page forwards. The header/footer must already have been defined using [DefHeaderByName\(\)](#), [DefFooterByName\(\)](#), [DefHTMLHeaderByName\(\)](#), or [DefHTMLFooterByName\(\)](#).

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted or 0 leaves the header/footer state unchanged.

Values (case-insensitive)

1 or on: Start using the selected header/footer from the new page onwards.

-1 or off: Start the selected header from the new page onwards.

page selector

Select a named CSS @page.

BLANK or omitted or leaves the CSS page unchanged.

See [Using @page](#) for more information

sheet-size

sheet-size can be specified either as a pre-defined page size, or as an array of width and height in millimetres e.g. array(210,297).

DEFAULT: BLANK - makes no change to the current sheet-size

Values (case-insensitive)

A0 - A10, B0 - B10, C0 - C10

4A0, 2A0, RA0 - RA4, SRA0 - SRA4

Letter, Legal, Executive, Folio

Demy, Royal

A (Type A paperback 111x178mm)

B (Type B paperback 128x198mm)

All of the above values can be suffixed with "-L" to force a Landscape page orientation document e.g. "A4-L"

Note: If you use the array() form for *sheet-size*, then you must:

- specify the width less than the height i.e. the dimensions of the page in portrait orientation; and
- explicitly define the *orientation* as L or P

[Changelog](#)

Version	Description
1.3	Parameters <i>resetpagenum</i> , <i>pagenumstyle</i> and <i>suppress</i> were added.
2.0	Parameters <i>margin-left</i> , <i>margin-right</i> , <i>margin-top</i> , <i>margin-bottom</i> , <i>margin-header</i> , <i>margin-footer</i> , <i>odd-header-name</i> , <i>odd-header-value</i> , <i>even-header-name</i> , <i>even-header-value</i> , <i>odd-footer-name</i> , <i>odd-footer-value</i> , <i>even-footer-name</i> , <i>even-footer-value</i> were added.
2.2	Values NEXT-ODD and NEXT-EVEN accepted for parameter <i>type</i> .
2.2	Parameters <i>type</i> and <i>orientation</i> changed to be case-insensitive.
3.0	<i>resetpagenum</i> changed to allow positive integers above 1
4.2	Parameter <i>pageselector</i> was added
4.3	Parameter <i>sheet-size</i> was added

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->WriteHTML('Your Introduction');
$mpdf->AddPage();
$mpdf->WriteHTML('Your Book text');

?>
```

Example #2 - Resetting page numbering with a new style

```
<?php

$mpdf=new mPDF();
$mpdf->useOddEven = 1;
$mpdf->AddPage('','','1','i','on');
$mpdf->WriteHTML('Your Book text');

?>
```

Example #3 - Defining new margins and page orientation

```
<?php

$mpdf=new mPDF();
$mpdf->WriteHTML('Your Introduction');
$mpdf->AddPage('L','','','','',50,50,50,50,10,10);
$mpdf->WriteHTML('Your Book text');

?>
```

Example #4 - Changing headers/footers

```
<?php

$mpdf=new mPDF();
$mpdf->useOddEven = 1;

$mpdf->defHeaderByName('myHeader', array (
    'L' => array (),
    'R' => array (),
    'C' => array ('content' => 'Chapter 2', 'font-style' => 'B', 'font-family' => 'serif'),
    'line' => 1,
));

$mpdf->defHTMLHeaderByName('myHeader2','<div style="text-align: center; font-weight: bold;">Chapter
2</div>');
```

```
$mpdf->WriteHTML('Your Introduction');

// Selects new headers for ODD and EVEN pages to use from the new page onwards
// Note the html_ prefix before the named HTML header
$mpdf->AddPage('', 'NEXT-ODD', '', '', '', '', '', '', '', 'myHeader', 'html_myHeader2', '', '', 1, 1,
0, 0);

$mpdf->WriteHTML('Your Book text');

// Turns all headers/footers off from new page onwards
$mpdf->AddPage('', 'NEXT-EVEN', '', '', '', '', '', '', '', '', '', '', '-1,-1,-1,-1);

$mpdf->WriteHTML('End section of book with no headers');

?>
```

Notes

Note: Unlike FPDF **AddPage()** does not need to be called at the beginning of the document if you are writing HTML code to the document. [WriteHTML\(\)](#) will automatically add the first page to a new document.

Note: **pagebreak** can be used as an **HTML** equivalent of both [AddPage\(\)](#).

See Also

- <pagebreak> - Custom HTML tag - equivalent to [AddPage\(\)](#)

AddPageByArray()

(mPDF >= 5.0)

AddPageByArray — Add a new page using an array of parameters

Description

```
void AddPageByArray ([ array $arr ])
```

Add a new page to the document using an array of parameters which are all optional. The parameter *type* can specify certain conditions which determine how many pages are added. If writing a **DOUBLE-SIDED** document, a conditional page-break (*type*="E" or "O") will add a new page only if required to make the current page match the type (i.e. **ODD** or **EVEN**); a page-break with *type*="NEXT-ODD" or "NEXT-EVEN" will add one or two pages as required to make the current page match the type (i.e. **ODD** or **EVEN**).

Number of pages added:

DOUBLE-SIDED			
type	SINGLE-SIDED	Currently ODD page	Currently EVEN page
BLANK	1	1	1
O or ODD	0	0	1
E or EVEN	0	1	0
NEXT-ODD	1	2	1
NEXT-EVEN	1	1	2

Note: If no new page is added, the other parameters will be ignored e.g. resetting page numbers/styles, margins and headers/footers. If 2 pages are added, the changes in page numbers/styles, margins and headers/footers will start on the final added page.

Note: From mPDF >= 3.0 the page numbering can be reset to any positive number. Prior to this, it was only possible to reset it to 1.

Note: The description below uses the same parameter names as used in AddPage(), but these were not recognised in mPDF < 5.4. The examples use different parameter names. From mPDF >=5.4 both are supported. See below for a list of alternatives.

Parameters

orientation = L|P

This attribute specifies the orientation of the new page.
BLANK or omitted leaves the current orientation unchanged

Values (case-insensitive)
L or landscape: Landscape
P or portrait: Portrait

type = E|O|even|odd|next-odd|next-even

If *type* is specified as "E" or "O" when writing a **DOUBLE-SIDED** document, the page-break is conditional; a new page will only be added if necessary to meet the specified condition.
If *type* is specified as "NEXT-ODD" or "NEXT-EVEN" when writing a **DOUBLE-SIDED** document, either one or two pages are added as necessary to meet the specified condition.

If not writing a **DOUBLE-SIDED** document, a page-break type="E" or "O" will be ignored.
BLANK or omitted will force a new page unconditionally.

Values (case-insensitive)

O or ODD: Add a new page if required to make current page an **ODD** one.
E or EVEN: Add a new page if required to make current page an **EVEN** one.
NEXT-ODD: Add one or two pages as required to make the current page **ODD**.
NEXT-EVEN: Add one or two pages as required to make the current page **EVEN**.

resetpagenum = 1 - ∞

Sets/resets the document page number to *resetpagenum* starting on the new page. (The value must be a positive integer).
BLANK or omitted or 0 leaves the current page number sequence unchanged.

pagenumstyle = 1|A|a||i

Sets/resets the page numbering style (values as for lists)
BLANK or omitted leaves the current page number style unchanged.

Values (case-sensitive)

1: Decimal - 1,2,3,4...
A: Alpha uppercase - A,B,C,D...
a: Alpha lowercase - a,b,c,d...
I: Roman uppercase - I, II, III, IV...
i: Roman lowercase - i, ii, iii, iv...

suppress = on|off|1|0

suppress=on will suppress document page numbers from the new page onwards (until *suppress=off* is used)
BLANK or omitted leaves the current condition unchanged.

Values (case-insensitive)

1 or on: Suppress (hide) page numbers from the new page forwards.
0 or off: Show page numbers from the new page forwards.

margin-left
margin-right
margin-top
margin-bottom
margin-header
margin-footer

Sets the page margins from the new page forwards.

All values should be specified as **LENGTH** in millimetres.

If you are writing a **DOUBLE-SIDED** document, the margin values will be used for **ODD** pages; left and right margins will be mirrored for **EVEN** pages.

BLANK or omitted leaves the current margin unchanged. NB "0" (zero) will set the margin to zero.

odd-header-name
even-header-name
odd-footer-name
even-footer-name

Selects a header or footer by name to use from the new page forwards. The header/footer must already have been defined using [DefHeaderByName\(\)](#), [DefFooterByName\(\)](#), [DefHTMLHeaderByName\(\)](#), or

DefHTMLFooterByName()

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted leaves the header/footer unchanged. NB **BLANK** will not unset the header. Set *odd-header-value* to -1 to turn the header off.

Note: You must add the prefix 'html_' before the name if it is a HTMLHeader.

odd-header-value

even-header-value

odd-footer-value

even-footer-value

Specify whether to show or hide the named header or footer from the new page forwards. The header/footer must already have been defined using **DefHeaderByName()**, **DefFooterByName()**, **DefHTMLHeaderByName()**, or **DefHTMLFooterByName()**.

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted or 0 leaves the header/footer state unchanged.

Values (case-insensitive)

1 or on: Start using the selected header/footer from the new page onwards.

-1 or off: Start the selected header from the new page onwards.

page selector

Select a named CSS @page.

BLANK or omitted or leaves the CSS page unchanged.

See [Using @page](#) for more information

sheet-size

sheet-size can be specified either as a pre-defined page size, or as an array of width and height in millimetres e.g. array(210,297).

DEFAULT: BLANK - makes no change to the current sheet-size

Values (case-insensitive)

A0 - A10, B0 - B10, C0 - C10

4A0, 2A0, RA0 - RA4, SRA0 - SRA4

Letter, Legal, Executive, Folio

Demy, Royal

A (Type A paperback 111x178mm)

B (Type B paperback 128x198mm)

All of the above values can be suffixed with "-L" to force a Landscape page orientation document e.g. "A4-L"

Note: If you use the array() form for *sheet-size*, then you must:

- specify the width less than the height i.e. the dimensions of the page in portrait orientation; and
- explicitly define the *orientation* as L or P

Changelog

Version	Description
5.0	Function was added.
5.4	Alternative parameter names added

Alternative parameter names

type	condition
margin-left	mgl
margin-right	mgr
margin-top	mgt
margin-bottom	mgb
margin-header	mgh
margin-footer	mgf
odd-header-name	ohname
even-header-name	ehname
odd-footer-name	ofname
even-footer-name	efname
odd-header-value	ohvalue
even-header-value	ehvalue
odd-footer-value	ofvalue
even-footer-value	efvalue
pageselector	pagesel
sheet-size	newformat

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->WriteHTML('Your Introduction');
$mpdf->AddPage();
$mpdf->WriteHTML('Your Book text');

?>
```

Example #2 - Resetting page numbering with a new style

```
<?php

$mpdf=new mPDF();
$mpdf->useOddEven = 1;
$mpdf->AddPageByArray(array(
    'resetpagenum' => '1',
    'pagenumstyle' => 'i',
    'suppress' => 'on',
));
$mpdf->WriteHTML('Your Book text');

?>
```

Example #3 - Defining new margins and page orientation

```
<?php
```

```
$mpdf=new mPDF();
$mpdf->WriteHTML('Your Introduction');
$mpdf->AddPageByArray(array(
    'orientation' => 'L',
    'mgl' => '50',
    'mgr' => '50',
    'mgt' => '50',
    'mgb' => '50',
    'mgh' => '10',
    'mgf' => '10',
));
$mpdf->WriteHTML('Your Book text');

?>
```

Example #4 - Changing headers/footers

```
<?php

$mpdf=new mPDF();
$mpdf->useOddEven = 1;

$mpdf->defHeaderByName('myHeader', array (
'L' => array (),
'R' => array (),
'C' => array ('content' => 'Chapter 2', 'font-style' => 'B', 'font-family' => 'serif'),
'line' => 1,
));

$mpdf->defHTMLHeaderByName('myHeader2','<div style="text-align: center; font-weight: bold;">Chapter
2</div>');

$mpdf->WriteHTML('Your Introduction');

// Selects new headers for ODD and EVEN pages to use from the new page onwards
// Note the html_ prefix before the named HTML header
$mpdf->AddPageByArray(array(
    'condition' => 'NEXT-ODD',
    'ohname' => 'myHeader',
    'ehname' => 'html_myHeader2',
    'ohvalue' => 1,
    'ehvalue' => 1,
));

$mpdf->WriteHTML('Your Book text');

// Turns all headers/footers off from new page onwards
$mpdf->AddPage('', 'NEXT-ODD', '', '', '', '', '', '', '', '', '', '', '', '', '-1,-1,-1,-1);
$mpdf->AddPageByArray(array(
    'condition' => 'NEXT-ODD',
    'ohvalue' => -1,
    'ehvalue' => -1,
    'ofvalue' => -1,
    'efvalue' => -1,
));

$mpdf->WriteHTML('End section of book with no headers');

?>
```

Example #5 - Blank template as example

```
<?php

$mpdf=new mPDF();
$mpdf->WriteHTML('Introduction');
$mpdf->AddPageByArray(array(
    'orientation' => '',
    'condition' => '',
    'resetpagenum' => '',
    'pagenumstyle' => '' ,
```

```
'suppress' => '',
'mgl' => '',
'mgr' => '',
'mgt' => '',
'mgb' => '',
'mgh' => '',
'mgf' => '',
'ohname' => '',
'ehname' => '',
'ofname' => '',
'efname' => '',
'ohvalue' => 0,
'ehvalue' => 0,
'ofvalue' => 0,
'efvalue' => 0,
'pagesel' => '',
'newformat' => '',
));
$mpdf->WriteHTML('Chapter 1 ...');
$mpdf=Output();
?>
```

Notes

Note: Unlike FPDF **AddPage()** or **AddPageByArray()** does not need to be called at the beginning of the document if you are writing HTML code to the document. [WriteHTML\(\)](#) will automatically add the first page to a new document.

See Also

- <pagebreak> - Custom HTML tag - equivalent to **AddPage()** or **AddPageByArray()**

AddSpotColor

(mPDF >= 5.1)

AddSpotColor – Define a Spot colour

Description

```
void AddSpotColor ( string $name , int $c , int $m , int $y , int $k )
```

Define a Spot colour which can be used in the document. Spot colours need to be defined at the start of the script.

Parameters

name

Specifies the name used for the spot colour.

c, m, y, k

Specifies the CMYK values to be used to display in the document or if the spot colour is not available.
DEFAULT: 0

Examples

Example #1

```
<?php  
  
$mpdf->AddSpotColor('PANTONE 534 EC',85,65,47,9);  
  
?>  
  
HTML  
<p style="color: spot(PANTONE 300 EC,80%);">This will be printed using PANTONE 300 EC at 80%  
tint</p>
```

AliasNbPages()

(mPDF >= 1.0)

AliasNbPages - Defines the placeholder used to insert total number of pages into the document

Description

```
void AliasNbPages ( string $text )
```

Set the value for the variable string `aliasNbPg` which is used as a placeholder used to insert total number of pages into the document.

Parameters

`text`

Defines the text for the variable `aliasNbPg`.

DEFAULT: {nb}

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->AliasNbPages('[pagetotal]');  
$mpdf->WriteHTML('<p>There are [pagetotal] pages in this document</p>');  
$mpdf->Output();  
  
?>
```

Example #2

```
$mpdf->AliasNbPages('[pagetotal]');  
  
is the exact equivalent of:  
  
$mpdf->aliasNbPg = '[pagetotal]';
```

See Also

- [Replaceable Aliases](#)
- [AliasNbPageGroups\(\)](#) - Sets the placeholder alias for the total number of pages in a document or page group

AliasNbPageGroups()

(mPDF >= 2.0)

AliasNbPageGroups - Defines the placeholder used to insert total page number into the document

Description

```
void AliasNbPageGroups ( string $text )
```

Set the value for the variable string `aliasNbPgGp` which is used as a placeholder used to insert total page number into the document. If you have reset the page numbering with `AddPage()` or `<pagebreak>` the total number of pages in the current page group will be used (up to where the numbering is reset) rather than the total number of pages in the whole document.

Parameters

`text`

Defines the text for the variable `aliasNbPgGp`.

DEFAULT: {nbpg}

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->AliasNbPageGroups('[pagetotal]');  
$mpdf->WriteHTML('<p>There are [pagetotal] pages in this page group</p>');  
$mpdf->Output();  
  
?>
```

Example #2

```
$mpdf->AliasNbPageGroups('[pagetotal]');  
  
is the exact equivalent of:  
  
$mpdf->aliasNbPgGp = '[pagetotal]';
```

See Also

- [Replaceable Aliases](#)
- [AliasNbPages\(\)](#) - Sets the placeholder alias for the total number of pages in the document

Annotation()

(mPDF >= 2.2)

Annotation – Add an Annotation to the document

Description

```
void Annotation ( string $text [, float $x , float $y [, string $icon [, string $author [, string $subject [, float $opacity [, array $colarray [, mixed $popup ]]]]]]])
```

Adds an Annotation to the document. An annotation is like a Tooltip on a webpage. The Annotation marker, like those of "Sticky Notes" in Adobe Reader. When the reader passes the cursor over, it will display a popup text box.

The exact position on the page can be specified using *x* and *y*, or left to position automatically. If *x* and *y* are not specified, the Annotation will be inserted at the current position of writing in the document. The *x* position (horizontal) can be overridden by the variable [\\$annotMargin](#), which can be used to force the Annotation marker to display in the right margin.

Note: All text to do with an annotation (text, author, subject) is rendered with the system font and can therefore contain any Unicode character even if the document font restricts to a specific codepage.

**** SetUserRights was removed in mPDF 2.4 ****

Annotations cannot be moved or deleted by the reader

Parameters

text

This parameter specifies the text to appear in the popup text box

x

Sets the *x* position of the (bottom left edge of the) Annotation marker, set in mm from the left of the page.

BLANK or omitted or 0 uses the current writing position on the page, unless overridden by [\\$annotMargin](#).

y

Sets the *y* position of the (bottom left edge of the) Annotation marker, set in mm from the top of the page. When Annotation markers are used within the text ([\\$annotMargin=FALSE](#)), the marker is raised by the current lineheight to appear above the text.

BLANK or omitted or 0 uses the current writing position on the page.

icon

Sets the appearance of the Annotation marker.

BLANK or omitted uses **DEFAULT** i.e. 'Note'

Values (case sensitive)

- Note
- Comment
- Help
- Insert
- Key
- NewParagraph

Paragraph
DEFAULT: Note

Note: The default is "Comment" when using Annotations from HTML markup when title2annots is **TRUE**

author

This specifies the name of the Author which will appear at the top of the popup text box.
DEFAULT: BLANK

subject

This specifies the text to appear in the Annotation properties.
DEFAULT: BLANK

opacity

Sets the opacity of the Annotation marker. Values must be greater than 0 and <= 1.
BLANK or omitted or 0: sets the opacity to the value of **annotOpacity** (**DEFAULT** 0.5), unless **annotMargin** forces the Annotations to appear in the margin, when the **DEFAULT** is 1

colarray

An array containing RGB color specification, which determines the colour of the Annotation marker background
DEFAULT array(255,255,0) i.e. Yellow

popup

Specify whether to show the popup box for the annotation when the PDF document is opened, and optional specify its dimensions and/or position.
BLANK or omitted, 0 or "0" - the popup box is not shown.
Any other value forces the popup box to appear when the document is opened.

An array of 2 numbers will set the X and Y position in mm e.g. *popup*=array(30, 30) will show a popup box with the top left corner 30mm from the top of the page and 30mm from the left of the page.
An array of 4 numbers will set the X and Y position and also the width and height in mm e.g.
popup=array(30, 30, 80, 50) will show a popup box with the top left corner 30mm from the top of the page and 30mm from the left of the page, a width of 80mm and a height of 50mm.
Note that the PDF Reader (e.g. Adobe Reader) may reposition the popup box as it pleases.

Changelog

Version	Description
2.2	The function was added.
2.4	Annotations cannot be moved or deleted
4.3	Parameter <i>popup</i> was added

Examples

Example #1

```
<?php
```

```
$mpdf=new mPDF();
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Annotation("Text annotation example");
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');

?>
```

Example #2

```
$mpdf=new mPDF();

// The Annotation markers will appear 10mm in from the right margin of the page
$mpdf->annotMargin = 10;

// The Annotation markers need no transparency as they appear in the margins
$mpdf->annotOpacity = 1;

$mpdf->WriteHTML('<p>Hallo World</p>');

$mpdf->Annotation("Text annotation example\nCharacters test:\xd1\x87\xd0\xb5
\xd0\xbf\xd1\x83\xd1\x85\xd1\x8a\xd1\x82", 145, 24, 'Comment', "Ian Back", "My Subject", 0.7,
array(127, 127, 255));

$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');
```

See Also

- [annotMargin](#) - Specify the x (horizontal) placement of Annotation markers
- [annotOpacity](#) - Specifiy the default opacity used for Annotation markers
- [<annotation>](#) - Custom HTML tag - equivalent to **Annotation**
- [title2annots](#) - Convert all HTML element *title* attributes to Annotations

AutosizeText()

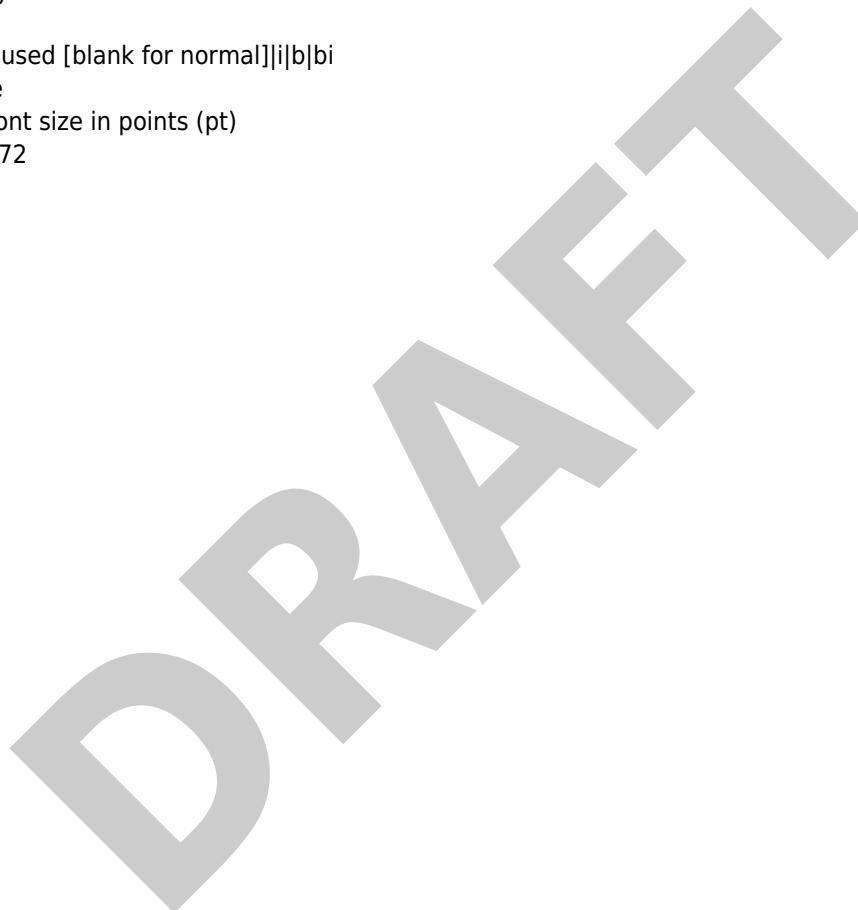
AutosizeText(string text, float width, string font, string style[, float fontsize])

Writes a single line of text directly to the PDF document at the current position.

Font size will be automatically reduced to fit width (but is not increased).

NB Does not reverse RTL text

- **text**
UTF-8 encoded text to write. Single line only.
- **width**
Width of text in millimeters. The font size will be reduced if required to fit this size.
- **font**
Font family to use
- **style**
Font style used [blank for normal]|i|b|bi
- **fontsize**
Maximm font size in points (pt)
Default = 72

A large, semi-transparent watermark text "DRAFT" is rotated diagonally from bottom-left to top-right across the page.

Bookmark()

(mPDF >= 1.0)

Bookmark – Add a Bookmark to the document

Description

```
void Bookmark ( string $content [, int $level [, float $y ]])
```

Add a Bookmark to the document. Bookmarks appear in Adobe Reader and link to specific points in the text. The target is set as the current writing position in the document when the Bookmark is defined.

Note: Bookmarks use Adobe Reader system fonts, therefore any Unicode text can be used, even if a unibyte codepage is being used for the document.

Parameters

content

Specifies the text to appear as a Bookmark.

content cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <annotation content="This is < 40" />

It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.

level

level specifies the "tree" level for the Bookmark. The top level is 0. See Example 2 below. Accepts an integer from 0 to the maximum depth you wish.

DEFAULT: 0

y

y specifies the y-coordinate on the page for the Bookmark. The top of the page is 0. The default is the current writing position on the page.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->Bookmark('Start of the document');  
$mpdf->WriteHTML('<div>Section 1 text</div>');  
$mpdf->Output('filename.pdf');  
  
?>
```

Example #2

```
$mpdf=new mPDF();  
  
$mpdf->Bookmark('Section 1', 0);  
$mpdf->WriteHTML('<div>Section 1 text</div>');  
  
$mpdf->Bookmark('Chapter 1', 1);
```

```
$mpdf->WriteHTML('<div>Chapter 1 text</div>');

$mpdf->Bookmark('Chapter 2', 1);
$mpdf->WriteHTML('<div>Chapter 2 text</div>');

$mpdf->Bookmark('Section 2', 0);
$mpdf->WriteHTML('<div>Section 2 text</div>');

$mpdf->Bookmark('Chapter 3', 1);
$mpdf->WriteHTML('<div>Chapter 3 text</div>');

$mpdf->Output('filename.pdf');

This will produce a Bookmark tree in Adobe Reader:
+ Section 1
  + Chapter 1
  + Chapter 2
+ Section 2
  + Chapter 3
```

Notes

Note: To set the Bookmark for a Table of Contents, see *toc-bookmarkText* in [TOCpagebreak\(\)](#).

See Also

- <[bookmark](#)> - Custom HTML tag - equivalent to [Bookmark\(\)](#)

CircularText()

mPDF >= 5.4

```
function CircularText($x, $y, $r, $text, $align='top|bottom', fontfamily='', fontsize=0, fontstyle='',  
$kerning=120, $fontwidth=100, $divider='') {
```

x: abscissa of center

y: ordinate of center

r: radius of circle

text: text to be printed

align: text alignment: top or bottom. Default value: top

divider: optional character string to divide top and bottom text

kerning: (fixed) spacing between letters in percentage. Default value: 120. Zero is not allowed.

fontwidth: width of letters in percentage. Default value: 100. Zero is not allowed

- uses automatic Kerning between letters if useKerning == true

See [<textcircle>](#) for more details

CreateIndex()

(mPDF >= 2.2)

CreateIndex — Generate an Index for the document

Note: This function is removed in mPDF v6.0 and replaced by [InsertIndex\(\)](#)

Description

```
void CreateIndex ([ int $numberofcolumns [, float $fontsize [, float $linespacing [, float $offset [, int $usedivletters [, float $divlettersfontsize [, float $columngap [, string $font [, string $divletterfont [, boolean $uselinking ]]]]]]]]])
```

Inserts an Index for the document based on index entries made using <indexentry> or [CreateIndex\(\)](#).

Note: Prior to mPDF 2.2 the function CreateReference() was used. CreateIndex() is now the preferred form.

Note: *uselinking* was added in mPDF 3.0

Parameters

numberofcolumns

Set the number of (vertical) columns to use for the Index
BLANK or omitted or 0 or 1 uses the whole page is used as one column.
DEFAULT: 1

fontsize

Sets the font size for the Index in **points** (pt)
BLANK or omitted or 0 uses the default font-size for the document.

linespacing

Sets the line-height used for index entries. Usual values between 1.0 and 1.4.
BLANK or omitted or 0 uses the default value.
DEFAULT: 1.2 (changed from 1.0 in mPDF < 3.0)

offset

Sets the text indent (in mm) for subsequent lines, if an index entry flows onto two or more lines.
BLANK or omitted uses a default value of 3mm.

usedivletters

Defines whether to divide index entries starting with the same letter, using a (large) letter as a heading.
DEFAULT: 1

Values

- 1: show dividing letters in the Index
- 0: do not show dividing letters in the Index
- BLANK** or omitted uses a default value of 1

divlettersize

Sets the font size for the dividing letters in **points** (pt)
BLANK or omitted or 0 uses the 1.8 times the default font-size for the document.

columngap

Sets the gap between columns (if set) in millimeters.
BLANK or omitted uses the default value.
DEFAULT: 5 (mm)

font

Set the font-family for the Index.
BLANK or omitted uses default font-family for the document.

divletterfont

Set the font-family for the dividing letters in the Index.
BLANK or omitted uses default font-family for the document.

uselinking

Specify whether to add hyperlinks (internal links) to the entries in the document Index.
TRUE or 1: add links to Index
BLANK or omitted, 0 or **FALSE**: do not add links to the Index
DEFAULT: **FALSE**

Changelog

Version	Description
2.2	Function was added as a synonym for CreateReference().
3.0	<i>uselinking</i> parameter was added
3.0	Default value for <i>linespacing</i> changed to 1.2

Examples

Example #1

```
<?php

$mpdf=new mPDF();

$mpdf->WriteHTML('<p>Beginning bit of document...</p>');
$mpdf->IndexEntry("Buffalo");
$mpdf->WriteHTML('<p>Your text which refers to a buffalo, which you would like to see in the Index</p>');

$mpdf->AddPage();
$mpdf->WriteHTML('<h2>Index</h2>',2);
$mpdf->CreateIndex(2, '', '', 3, 1, '', 5, 'serif','sans-serif');

$mpdf=Output();

?>
```

Note: There is no HTML equivalent of CreateIndex()

See Also

- [IndexEntry\(\)](#) - Add an Index entry in the document
- [<indexentry>](#) - Mark an Index entry in the document

CreateReference()

(mPDF >= 1.0)

CreateRefence — Generate an Index - DEPRACATED / Removed in mPDF v6.0

Description

```
void CreateReference ( )
```

Note: **CreateReference()** is now depracated in favour of the better-named [CreateIndex\(\)](#).

Changelog

Version	Description
1.0	Function was added.
2.2	Depracated in favour of CreateIndex()

See Also

- [CreateIndex\(\)](#) - Generate an Index

DefFooterByName()

(mPDF >= 2.0)

DefFooterByName - Define a page footer with a given name

Description

```
void DefFooterByName ([ string $name [, array $footer ]])
```

Define a page footer with a given name. Named footers can be referenced and set later in the document e.g. SetFooterByName()

Note: Do not name any header or footer starting with `html_`. This prefix is reserved to identify an **HTML** header/footer when passing its name in a reference.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

User-defined name for the footer. If *name* = **BLANK** the name '_default' is used.

footer

This parameter specifies the content of the page footer as an array.

DEFAULT: array() above

Values in the array

content: TEXT STRING

font-size: FLOAT font size in pts

font-style: B|I|BI|BLANK STRING

font-family: Any available font-family

color: CSS '#RRGGBB' string

line: 0|1 - specify whether to draw a line above the footer

```
$footer = array (
    'L' => array (
        'content' => '',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'C' => array (
        'content' => '',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'R' => array (
        'content' => 'My document',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
);
```

```
    'line' => 1,  
);
```

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 3](#)

See Also

- [DefHeaderByName\(\)](#)
- [<pagefooter>](#)
- [SetFooterByName\(\)](#)
- [<setpagefooter>](#)

DefHeaderByName()

(mPDF >= 2.0)

DefHeaderByName - Define a page header with a given name

Description

```
void DefHeaderByName ([ string $name [, array $header ]])
```

Define a page header with a given name. Named headers can be referenced and set later in the document e.g. SetHeaderByName()

Note: Do not name any header or footer starting with `html_`. This prefix is reserved to identify an **HTML** header/footer when passing its name in a reference.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

User-defined name for the header. If *name* = **BLANK** the name '`_default`' is used.

header

This parameter specifies the content of the page header as an array.

DEFAULT: array()

Values in the array

content: TEXT STRING

font-size: FLOAT font size in pts

font-style: B|I|BI|BLANK STRING

font-family: Any available font-family

color: CSS '#RRGGBB' string

line: 0|1 - specify whether to draw a line under the Header

```
$header = array (
    'L' => array (
        'content' => '',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'C' => array (
        'content' => '',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'R' => array (
        'content' => 'My document',
        'font-size' => 10,
        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
);
```

```
    'line' => 1,  
);
```

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 3](#)

See Also

- [DefFooterByName\(\)](#)
- [<pageheader>](#)
- [SetHeaderByName\(\)](#)
- [<setpageheader>](#)

DefHTMLFooterByName()

(mPDF >= 2.0)

DefHTMLFooterByName – Define an HTML page footer with a given name

Description

```
void DefHTMLFooterByName ( string $name [, string $html ] )
```

Define an HTML page footer with a given name. Named footers can be referenced and set later in the document e.g. [SetHTMLFooterByName\(\)](#)

Note: Do not name any header or footer starting with `html_`. This prefix is reserved to identify an **HTML** header/footer when passing its name in a reference.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

User-defined name for the footer. If *name* = **BLANK** the name '_default' is used.

html

This parameter specifies the content of the page footer as a string of valid HTML code.

DEFAULT: BLANK

Changelog

Version	Description
---------	-------------

2.0	The function was added.
-----	-------------------------

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 4](#)

See Also

- [DefHTMLHeaderByName\(\)](#)
- [<htmlpagefooter>](#)
- [SetHTMLFooterByName\(\)](#)
- [<sethtmlpagefooter>](#)

DefHTMLHeaderByName()

(mPDF >= 2.0)

DefHTMLHeaderByName – Define an HTML page header with a given name

Description

```
void DefHTMLHeaderByName ( string $name [, string $html ] )
```

Define an HTML page header with a given name. Named headers can be referenced and set later in the document e.g. [SetHTMLHeaderByName\(\)](#)

Note: Do not name any header or footer starting with `html_`. This prefix is reserved to identify an **HTML** header/footer when passing its name in a reference.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

User-defined name for the header. If *name* = **BLANK** the name '`_default`' is used.

html

This parameter specifies the content of the page header as a string of valid HTML code.

DEFAULT: BLANK

Changelog

Version Description

2.0	The function was added.
-----	-------------------------

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 4](#)

See Also

- [DefHTMLFooterByName\(\)](#)
- [`<htmlpageheader>`](#)
- [SetHTMLHeaderByName\(\)](#)
- [`<sethtmlpageheader>`](#)

Image()

See [Images](#).

`Image($filename, $x, $y, $width, $height, $ext, $href_link, $paint=true, $constrain=true, $is_watermark=false)`

```
$mpdf->Image('files/images/frontcover.jpg',0,0,210,297,'jpg','','true, false);  
// the last "false" allows a full page picture
```

See Also

- [SetAlpha\(\)](#) - Set the opacity and blend mode for Images

A large, light gray watermark reading "DRAFT" diagonally across the page.

ImportPage()

(mPDF >= 2.3)

ImportPage – Import a page from an external PDF file

Description

```
int ImportPage ( int $pageno [, float $crop_x [, float $crop_y [, float $crop_w [, float $crop_h [, string  
$boxname ]]]]])
```

Import a page, or part of a page, from an external PDF file. The external source file must first be set with [SetSourceFile\(\)](#). A 'template' is created in mPDF which stores the image of this page, ready to insert into the document.

Parameters

pageno

This parameter specifies the page number from the source PDF file to import. *pageno* should be a positive integer value.

DEFAULT: 1

crop_x

Specifies the x-coordinate (abscissa) for the page of the source PDF file, when importing a 'cropped' page into the template. Value in millimetres.

DEFAULT: 0

crop_y

Specifies the y-coordinate (ordinate) for the page of the source PDF file, when importing a 'cropped' page into the template. Value in millimetres.

DEFAULT: 0

crop_w

Specifies the width in millimetres when importing a 'cropped' page into the template.

DEFAULT: **NULL** uses the full page width from the source file

crop_h

Specifies the height in millimetres when importing a 'cropped' page into the template.

DEFAULT: **NULL** uses the full page height from the source file

boxname

boxname is currently not used.

Return Value

ImportPage() returns an ID for the template which it has created. This ID can be used at any time to insert the template into the document with [UseTemplate\(\)](#) or [SetPageTemplate\(\)](#)

Changelog

Version	Description
2.3	Function was added.

Examples

Example #1 - Using a full page

```
<?php

include("../mpdf.php");

$mpdf=new mPDF();
$mpdf->SetImportUse();

$pagecount = $mpdf->SetSourceFile('logoheader.pdf');
$tplId = $mpdf->ImportPage($pagecount);

$mpdf->UseTemplate($tplId);

$mpdf->WriteHTML('Hallo World');

$mpdf->Output();

?>
```

Example #2 - Using a 'cropped' page

```
<?php

include("../mpdf.php");

$mpdf=new mPDF();
$mpdf->SetImportUse();

$pagecount = $mpdf->SetSourceFile('testfile.pdf');

$tplId = $mpdf->ImportPage($pagecount, 50, 50, 100, 100);

$mpdf->UseTemplate($tplId, '', '', 100, 100);

$mpdf->Output();

?>
```

See Also

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template

IndexEntry()

(mPDF >= 2.2)

IndexEntry - Insert an Index entry for the document

Description

```
void IndexEntry ( string $content [ string $xref ])
```

Insert an Index entry for the document Index, referencing the current writing position in the document. If *xref* is set, it will appear as a cross-referencing entry in the index as for [IndexEntrySee\(\)](#).

Note: The Index must be generated explicitly at the end of the document using [CreateIndex\(\)](#) at some point before [Output\(\)](#) is called.

Note: Prior to mPDF 2.2 the function [Reference\(\)](#) was used. [IndexEntry\(\)](#) is now the preferred form.

Parameters

content

This parameter sets the text as it will appear in the Index entry.

content cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <annotation content="This is < 40" />

It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.

Text entries passed in the form "Subject:Subcategory" will appear in the Index as "Subject, Subcategory"

REQUIRED

xref

This parameter sets the text used as a cross-reference. Text should be UTF-8 encoded.

xref cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <indexentry xref="< 40" />

It is recommended that you use e.g. `htmlspecialchars($xref, ENT_QUOTES)` for this.

Text entries passed in the form "Subject:Subcategory" will appear in the Index as "Subject, Subcategory"

OPTIONAL

Changelog

Version Description

Version	Description
3.0	<i>xref</i> parameter added.

Examples

Example #1

```
<?php

$mpdf=new mPDF();

$mpdf->WriteHTML('<p>Beginning bit of document...</p>');
$mpdf->IndexEntry("Buffalo");
$mpdf->WriteHTML('<p>Your text which refers to a buffalo, which you would like to see in the Index</p>');

$mpdf->AddPage();
$mpdf->WriteHTML('<h2>Index</h2>',2);
$mpdf->CreateIndex(2, '', '', 3, 1, '', 5, 'serif','sans-serif');
```

```
$mpdf=Output();  
?>
```

Example #2

```
$mpdf->IndexEntry("Dromedary", "Camel:types");  
  
// This will produce an entry in the Index under 'Dromedary' appearing as:  
  
Dromedary - see Camel, types
```

Notes

Note: <indexentry> may be a preferred form, as it will allow more precise identification of the position and page - the <indexentry> can be placed just next to the appropriate word.

Recommended placement

Recommended placement of Index Entries is just after the first word following the opening tag of the block element:

```
<h2>First<indexentry... /> word of a heading or block</h2>
```

or alternatively just after the opening tag of the block element:

```
<h2><indexentry... />Heading or block</h2>
```

or just after a word to be marked:

```
... this is a word<indexentry... /> in the middle of text to be marked ...
```

See Also

- [IndexEntrySee\(\)](#) - Insert a cross-reference entry for the document Index
- [<indexentry>](#) - Mark an Index entry in the document
- [CreateIndex\(\)](#) - Generate a document Index

IndexEntrySee()

(mPDF >= 2.2)

IndexEntrySee - Insert a cross-reference entry for the document Index

Description

```
void IndexEntrySee ( string $content , string $see_content )
```

Insert a cross-reference entry for the document Index i.e. "Dromedary - see Camel".

Note: The Index must be generated explicitly at the end of the document using [CreateIndex\(\)](#) at some point before [Output\(\)](#) is called.

Parameters

content

This parameter sets the text as it will appear in the Index entry. Text should be UTF-8 encoded. Text entries passed in the form "Subject:Subcategory" will appear in the Index as "Subject, Subcategory".

REQUIRED

see_content

This parameter sets the text used as the cross-reference. Text should be UTF-8 encoded. Text entries passed in the form "Subject:Subcategory" will appear in the Index as "Subject, Subcategory"

REQUIRED

Examples

Example #1

```
$mpdf->IndexEntry("Dromedary", "Camel:types");  
  
// This will produce an entry in the Index under 'Dromedary' appearing as:  
  
Dromedary - see Camel, types
```

See Also

- [IndexEntry\(\)](#) - Mark an Index entry in the document
- [<indexentry>](#) - Mark an Index entry in the document
- [CreateIndex\(\)](#) - Generate a document Index

InsertIndex()

(mPDF >= 6.0)

InsertIndex — Generate an Index for the document

Description

```
void InsertIndex ([ int $usedivletters [, boolean $uselinking [, string $indexCollationLocale [, string $indexCollationGroup ]]]])
```

Inserts an Index for the document based on index entries made using <indexentry> or [IndexEntry\(\)](#).

Parameters

usedivletters

Defines whether to divide index entries starting with the same letter, using a (large) letter as a heading.
DEFAULT: 1

Values

1: show dividing letters in the Index
 0: do not show dividing letters in the Index
BLANK or omitted uses a default value of 1

uselinking

Specify whether to add hyperlinks (internal links) to the entries in the document Index.
TRUE or 1: add links to Index
BLANK or omitted, 0 or **FALSE**: do not add links to the Index
DEFAULT: FALSE

indexCollationLocale

Set a Locale to determine the overall sort order of index entries e.g. 'en_GB.utf8'. Available options are determined by the locales available in your system configuration. Always use a utf-8 locale.
BLANK or omitted uses current locale set in your system.

indexCollationGroup

If you have set your index to use Dividing letters, this value will determine how letters are grouped under a dividing letter. Values should be selected from the files in folder /collations/ e.g. 'English_United_Kingdom'
 NB This will not affect the overall order of entries, which is determined by the value above.
BLANK or omitted - grouping occurs under the first letter of the index entries.

Changelog

Version	Description
6.0	Function was added

See Also

- [Indexes](#)

MultiCell()

```
MultiCell(float w, float h, string text[, mixed border[, string align[, integer fill[, mixed link[, string directionality[, boolean encoded]]]]]])
```

Writes a block of text directly to the PDF document at the current position. Lines are wrapped at the margins. See the details for MultiCell() at FPDF. Two additional parameters have been added:

- **directionality**

Set to 'rtl' if using RTL language (right-to-left)

Default = 'ltr'

- **encoded**

When set to false (default), UTF-8 encoded text will be appropriately converted to the chosen output file format. It should only be set to true, when the input text has already been converted internally within the program.

Default = false

A large, semi-transparent watermark text "DRAFT" is positioned diagonally across the page, starting from the bottom-left and ending near the top-right. The text is in a bold, sans-serif font and is rendered in a light gray color.

Output()

(FPDF)

Output - Finalise the document and send it to specified destination

Description

```
string Output ([ string $filename , string $dest ])
```

Send the document to a given destination: browser, file or string. In the case of browser, the plug-in may be used (if present) or a download ("Save as" dialog box) may be forced.

Parameters

filename

The name of the file. If not specified, the document will be sent to the browser (destination I).
BLANK or omitted: "doc.pdf"

dest

Destination where to send the document.

DEFAULT: "I" i.e. Browser

I: send the file inline to the browser. The plug-in is used if available. The name given by *filename* is used when one selects the "Save as" option on the link generating the PDF.

D: send to the browser and force a file download with the name given by *filename*.

F: save to a local file with the name given by *filename* (may include a path).

S: return the document as a string. *filename* is ignored.

Note: You can use the 'S' option to e-mail a PDF file - see example under [E-mail a PDF file](#).

Examples

Example #1

```
<?php
// Sends output inline to browser
$mpdf=new mPDF();
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output();

?>
```

Example #2

```
<?php
// Saves file on the server as 'filename.pdf'
$mpdf=new mPDF();
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf','F');

?>
```

Example #3 - Sending file as e-mail (and also to browser)

```
$mpdf=new mPDF();
```

```
$mpdf->WriteHTML($html);

$content = $mpdf->Output('', 'S');

$content = chunk_split(base64_encode($content));
$mailto = 'recipient@domain.com';
$from_name = 'Your name';
$from_mail = 'sender@domain.com';
$replyto = 'sender@domain.com';
$uid = md5(uniqid(time()));
$subject = 'Your e-mail subject here';
$message = 'Your e-mail message here';
$filename = 'filename.pdf';

$header = "From: ".$from_name." <".$from_mail.">\r\n";
$header .= "Reply-To: ".$replyto."\r\n";
$header .= "MIME-Version: 1.0\r\n";
$header .= "Content-Type: multipart/mixed; boundary=\\"". $uid."\\r\\n\\r\\n";
$header .= "This is a multi-part message in MIME format.\r\n";
$header .= "--".$uid."\r\n";
$header .= "Content-type:text/plain; charset=iso-8859-1\r\n";
$header .= "Content-Transfer-Encoding: 7bit\r\n\r\n";
$header .= $message."\r\n\r\n";
$header .= "--".$uid."\r\n";
$header .= "Content-Type: application/pdf; name=\"".$filename."\"\r\n";
$header .= "Content-Transfer-Encoding: base64\r\n";
$header .= "Content-Disposition: attachment; filename=\"".$filename."\"\r\n\r\n";
$header .= $content."\r\n\r\n";
$header .= "--".$uid"--";
$is_sent = @mail($mailto, $subject, "", $header);

$mpdf->Output();
exit;
```

OverWrite()

(mPDFI >= 2.3)

OverWrite – Replace specified text strings in an existing PDF file

Description

```
mixed OverWrite ( string $sourcefile , mixed $search , mixed $replacement [, string $dest [, string $file_out ]]] )
```

Using the class extension mPDFI, an existing PDF file can be overwritten, replacing specified text with alternatives. For example you may have created a long complex PDF file, and you wish to produce copies with an individual number on each copy without having to re-generate the whole document each time.

`Overwrite()` does not re-flow the text from the source file. If the *replacement* string is longer than the *search* string, it may overlap the following text.

Note: `OverWrite()` has only been tested to work on PDF files produced by mPDF. It will work with encrypted files, as long as the same encryption properties are used for the new document.

Note: If you want the final PDF file to be encrypted, you need to encrypt the original source file. Make sure that you specify a password otherwise mPDF uses a random password and `OverWrite()` will not be able to access the text.

Note: From mPDF >= 5.3 a unique encryption key is generated each time you create a PDF file. So to use encryption you need to save variables when you create the original file. See Example 2.

Parameters

sourcefile

This parameter specifies the source PDF file to use. *sourcefile* should be a relative path to a local file.

search

The pattern to search for. It can be either a string or an array with strings. Must only contain only ASCII characters.

If the document is utf-8 mode, the search patterns must not exist in text with justified alignment. (Justified text is achieved in mPDF by varying the character spacing for each **SPACE** between words; this breaks up the text in the PDF file.)

replacement

The string or an array with strings to replace. *replacement* can contain any utf-8 encoded characters. If this parameter is a string and the *search* parameter is an array, only the first *search* element will be replaced by the *replacement* string, any extra *search*s will be replaced by an empty string. If both *search* and *replacement* parameters are arrays, each *search* will be replaced by the *replacement* counterpart. If there are fewer elements in the *replacement* array than in the *search* array, any extra *search*s will be replaced by an empty string.

dest

dest specifies the destination for the generated PDF document.

DEFAULT: "D"

Values

D: download the PDF file
 I: serves in-line to the browser
 S: returns the PDF document as a string
 F: save as file *file_out*

sourcefile

This parameter specifies the filename for the output PDF file. No path should be included unless *dest* is set as "F".
DEFAULT: "mpdf.pdf"

Return Value

OverWrite() returns the PDF file as a string if *dest* is set to "S".

Changelog

Version	Description
2.3	Function was added.

Examples

Example #1

```
<?php

include("../mpdf.php");

// Must set codepage (e.g. UTF-8 or Core fonts) the same as for original document
// The rest of the parameters do nothing
$mpdf=new mPDF('');
$mpdf->SetImportUse();

// forces no subsetting - otherwise the inserted characters may not be contained in a subset font
$mpdf->percentSubset = 0;

$search = array(
    'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX',
    'XXXXXXXXXXXXXXXXXXXXXXXXXXXXZXXXX'
);

$replacement = array(
    "personalised for Jos\xc3\xa9 Bloggs",
    "COPYRIGHT: Licensed to Jos\xc3\xa9 Bloggs"
);

$mpdf->OverWrite('test.pdf', $search, $replacement, 'I', 'mpdf.pdf' ) ;

?>
```

Example #2 Using encryption

```
<?php

include("../mpdf.php");

$mpdf=new mPDF('');
$mpdf->percentSubset = 0;

$mpdf->SetProtection(array(),'','bread'); // Need to specify a password
$mpdf->WriteHTML('<p>This copy is XXXXXXXXXXXXXXXXXXXXXXXXX</p>');
```

```
$mpdf->Output('test.pdf','F');

// Have to save various encryption keys, which are uniquely generated each document
$uid = $mpdf->uniqid;
$oval = $mpdf->Ovalue;
$encKey = $mpdf->encryption_key;
$uval = $mpdf->Uvalue;
$pval = $mpdf->Pvalue;
$RC128 = $mpdf->useRC128encryption;

unset( $mpdf );
//=====================================================================
$mpdf=new mPDF('');
$mpdf->SetImportUse();

// Re-instate saved encryption keys from original document
$mpdf->encrypted = true;
$mpdf->useRC128encryption = $RC128;
$mpdf->uniqid = $uid ;
$mpdf->Ovalue = $oval ;
$mpdf->encryption_key = $encKey ;
$mpdf->Uvalue = $uval ;
$mpdf->Pvalue = $pval ;

$search = array(
    'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
);

$replacement = array(
    "personalised for Jos\xc3\xa9 Bloggs"
);

$mpdf->OverWrite('test.pdf', $search, $replacement, 'I', 'mpdf.pdf' ) ;
exit;

?>
```

See Also

- [mPDFI\(\)](#) - Class constructor for importing files and templates
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [ImportPage\(\)](#) - Import a page from an external PDF file
- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template
- [RestartDocTemplate\(\)](#) - Re-start the use of a Document template from the next page

RestartDocTemplate()

(mPDFI >= 2.4)

RestartDocTemplate - Re-start the use of a Document template from the next page

Description

```
void RestartDocTemplate ( )
```

Restart the use of a document template (set by [SetDocTemplate\(\)](#)) from the next page.

Parameters

none

Changelog

Version	Description
2.4	Function was added.

Examples

Example #1

```
<?php

include("../mpdf.php");

$mpdf=new mPDF();
$mpdf->SetImportUse();

// Document template consisting of 2 pages, first with logo and addresses, 2nd with a simple header
$mpdf->SetDocTemplate('logoheader.pdf',true);

$mpdf->AddPage();

$mpdf->WriteHTML($firstletter);

$mpdf->RestartDocTemplate();
$mpdf->AddPage();
$mpdf->WriteHTML($secondletter);

$mpdf->RestartDocTemplate();
$mpdf->AddPage();
$mpdf->WriteHTML($thirdletter);

$mpdf->Output();

?>
```

See Also

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template

RoundedRect()

RoundedRect(float x, float y, float w, float h, float radius[, string style])

Draws a rectangle with rounded corners directly to the PDF document at the specified position.

- **x**
Abscissa of left edge of box - value in millimeters
- **y**
Ordinate of top edge of box - value in millimeters
- **w**
Width of the box - in millimeters
- **h**
Height of the box - in millimeters
- **radius**
Radius of the rounded corners
- **style**
Box style: D or empty string - draw border (default); F - fill; DF or FD - draw and fill
Default = "" i.e. border, no fill

DRAFT

SetAlpha()

(mPDF >= 1.0)

SetAlpha - Set the opacity for Images

Description

```
void SetAlpha ( float $alpha [, integer $blend ])
```

Set the opacity and blend mode for Images

Parameters

alpha

This parameter specifies the opacity for any subsequent images written to the current document.

Values

Float: 0 (transparent) to 1 (opaque)

blend

This parameter specifies the blend mode.

Values

STRING - One of the following:

Normal
Multiply
Screen
Overlay
Darken
Lighten
ColorDodge
ColorBurn
HardLight
SoftLight
Difference
Exclusion
Hue
Saturation
Color
Luminosity

DEFAULT: Normal

Changelog

Version	Description
1.0	Function was added.

Examples

Example #1

```
<?php
```

```
include("../mpdf.php");
$mpdf=new mPDF();

$mpdf->SetAlpha(0.5);
$mpdf->WriteHTML('');
$mpdf->SetAlpha(1);

// This produces the identical result as the last 3 lines
// $mpdf->WriteHTML('');

$mpdf->Output();
exit;

?>
```

See Also

- [Image\(\)](#) - Write an image to the current document

SetAnchor2Bookmark()

(mPDF >= 1.0)

SetAnchor2Bookmark – Specifies whether PDF Bookmarks are created automatically from HTML anchors

Description

```
void SetAnchor2Bookmark ( int $mode )
```

Specifies whether PDF Bookmarks are created from HTML anchors (e.g.) . This function simply sets the variable \$anchor2Bookmark

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

mode

Values

- 0 - does not generate a bookmark
- 1 - generate a bookmark using the text value of the *name* attribute

DEFAULT: 0

Changelog

Version Description

Version	Description
3.0	<i>mode</i> = 2 removed

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->SetAnchor2Bookmark(1);
$mpdf->WriteHTML('<h1><a name="Section 1" />The title</h1>');
$mpdf->Output();

?>

This will create a Bookmark in the PDF document: "Section 1"
```

Notes

Note: Prior to mPDF 3.0 you could specify *mode* = 2 which added the page number to the bookmark e.g. Introduction (p.32). This was removed as it did not accurately handle pagebreaks etc.

SetAuthor()

(mPDF >= 1.0)

SetAuthor – Set the document Author

Description

```
void SetAuthor ( string $text )
```

Set the Author for the document. This metadata can be seen when inspecting the document properties in Adobe Reader.

Note: Adobe Reader uses system fonts to display the document metadata, therefore any Unicode text can be used, even if a unibyte codepage is being used for the document.

Parameters

text

Defines the text to appear as the Author. The text should be UTF-8 encoded, but should not contain HTML mark-up tags. [strcode2utf\(\)](#) is a useful function provided with mPDF which converts HTML numerical entities to UTF-8 encoded string.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetAuthor('My Name');  
$mpdf->WriteHTML('<p>Hallo World</p>');  
$mpdf->Output('filename.pdf');  
  
?>
```

Example #2

```
// htmltoolkit contains a function strcode2utf() to convert htmlentities to UTF-8 encoded text  
  
<?php  
  
$mpdf=new mPDF();  
  
$md = strcode2utf("ايلات &#1601;يما  
ايلات &#1601;يما");  
  
$mpdf->SetAuthor($md);  
$mpdf->WriteHTML('<p>Hallo World</p>');  
$mpdf->Output('filename.pdf');  
  
?>
```

See Also

- [SetTitle\(\)](#) - Set document Title in metadata
- [SetCreator\(\)](#) - Set document Creator in metadata
- [SetSubject\(\)](#) - Set document Subject in metadata
- [SetKeywords\(\)](#) - Set document Keywords in metadata
- [strcode2utf\(\)](#) - Convert HTML numerical entities to UTF-8 encoded string

SetAutoFont()

(mPDF >= 2.3 <= 5.7)

SetAutoFont - Automatically detect language in the input HTML text and use appropriate fonts

Description

```
void SetAutoFont ( int $flag )
```

Turns on the AutoFont function, which automatically detects language in the input HTML text and uses appropriate fonts.

Note: This function was removed in mPDF 6.0 Use [autoLangToFont](#) for the same results

AutoFont uses `...` to mark text which is auto-detected. See [lang](#) for further details, but note that SetAutoFont also:

- disables support for `text-align=justify`
- sets `$useLang=TRUE`
- sets `$biDirectional=TRUE` (if `AUTOFONT_RTL` is set)

Note: Using automatic language detection adds considerable processing time when creating a large document.

Note: It is better to mark the HTML text yourself using block tags e.g. `<p lang=""` rather than relying on mPDF to set `<span lang=""` if possible; when set at block level, it allows appropriate changes to text alignment, character spacing and text-justification.

Note: Automatic language detection is based on analysis of the characters in the text. It is therefore not precise, and in particular it may be difficult to distinguish between different arabic languages (arabic, farsi, urdu, pashto etc), CJK languages (even between Chinese and Japanese text in same text chunk).

Note: Chinese Traditional and Simplified Chinese share so many characters that mPDF does not even try to distinguish which is being used. If AutoFont is being used and a chinese languages is identified, it is marked as `...`.

The font actually used for default chinese is determined by the settings in function `GetCodepage()` in `config_cp.php`; by default this is Chinese Simplified (GB, GBK).

Note: SetAutoFont is only valid when using UTF-8 as the codepage for the document.

Parameters

flag

This parameter specifies which languages are auto-detected. Either an integer or one of the defined constants can be used.

`SetAutoFont(0)` will turn off any auto-detection. Bitwise operators can be used with the defined constants e.g.

`SetAutoFont(AUTOFONT_CJK | AUTOFONT_THAIVIET)` will turn on autodetection for CJK languages and Thai and Vietnamese.

Values

`AUTOFONT_CJK = 1` = Any CJK languages (Chinese - Japanese - Korean)

`AUTOFONT_THAIVIET = 2` = Thai and Vietnamese

`AUTOFONTRTL = 4` = RTL languages i.e. Hebrew and Arabic, including Pashto, Urdu etc.

AUTOFONT_INDIC = 8 = Indic languages such as Devanagari

AUTOFONT_ALL = 15 = All of the above

DEFAULT: AUTOFONT_ALL (15)

Changelog

Version Description

4.0 AutoFont marks up the with `class="lang_xx"` as well as `lang="xx"`

2.3 Function was added.

2.3 Internal variable `$pregRTLchars` was altered for better detection of arabic/hebrew

6.0 Removed in favour of autoLangToFont

Examples

Example #1

```
<?php

include("../mpdf.php");
$mpdf=new mPDF('utf-8');

$html =
<p>Most of this text is in English, but has occasional words in Chinese:其貢獻在 or Vietnamese: Một khảo
sát mới cho biết, or maybe even Arabic:</p>
<p>البرادعي -12 - البرادعي</p>
<p>其貢獻在國際間亦備受肯定, 2005年</p>
';

$mpdf->SetAutoFont();

$mpdf->WriteHTML($html);

$mpdf->Output();

?>
```

See Also

- [useLang](#) - Specify whether to recognise and support the HTML attribute lang
- [autoFontGroupSize](#) - Specify the text chunk size to group when autodetecting text language
- [disableMultilingualJustify](#) - Specify whether to disable text justification in multilingual documents
- [lang](#) - Information on mPDF support for the HTML attribute lang

SetBasePath()

(mPDF >= 1.0)

SetBasePath - Specifies a base URL for mPDF to interpret relative URLs

Description

```
void SetBasePath ( string $url )
```

Specifies that mPDF should interpret any URLs in the HTML code relative to this *url*. Otherwise relative paths will be based on the current script. This is important for hyperlinks, external stylesheets, and images.

Note: From mPDF 5.7 <base href=""> is parsed in the HTML and used to set the base path.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

url

A full URL should be provided, but the following will all work:

```
http://www.mydomain.com  
http://www.mydomain.com/index.php  
http://www.mydomain.com/path/
```

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
  
$url = "http://www.php.net/manual/en/function.iconv.php";  
$html = file_get_contents($url);  
$mpdf->setBasePath($url);  
// Now any relative paths e.g.  will be parsed relative to a  
// base path of "http://www.php.net/manual/en/"  
  
$mpdf->WriteHTML($html);  
$mpdf->Output();  
  
?>
```

SetColumns()

(mPDF >= 1.0)

SetColumns – Control use of Columns on the page

Description

```
void SetColumns ( int $nCols [, string $vAlign [, float $gap ]])
```

Define, start or stop Columns in the document.

Note: The maximum ratio to adjust column height when justifying is set by `$max_colH_correction` - too large a value can give ugly results

Note: If you are setting HTMLHeaders or HTMLFooters, this will cancel any columns you have set; you need to call SetColumns() after commands like `SetHTMLHeader()` etc.

Parameters

nCols

Set the number of (vertical) columns to use on a page

BLANK or omitted or 0 or 1 turns Columns OFF i.e. the whole page is used as one column.

DEFAULT: 1

vAlign

Automatically adjusts height of columns to be equal if set to J or justify.

BLANK or omitted turns vertical-alignment OFF

Values (case-insensitive)

J or justify

DEFAULT: ""

gap

Set the gap between columns in millimeters

BLANK or omitted uses default value.

DEFAULT: 5 (mm)

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mPDF->SetColumns(2, 'J', 3);
$mpdf->WriteHTML('<p>Some text...</p>');
$mpdf->AddColumn();
$mpdf->WriteHTML('<p>Next column...</p>');
$mpdf=Output();

?>
```

See Also

- [AddColumn\(\)](#) - Start a new Column
- [<columnbreak>](#) - Start a new Column
- [<columns>](#) - Control the use of multiple columns on the page - HTML equivalent of SetColumns()

SetCompression()

(mPDF >= 1.0)

SetCompression - Specifies that mPDF should compress the data for the PDF file

Description

```
void SetCompression ( boolean $flag )
```

Specifies that mPDF should compress the data for the PDF file. This makes a smaller PDF file and is set by default to **TRUE** on initialising the mPDF class.

Parameters

flag

TRUE or FALSE to specify whether mPDF should compress the data for the PDF file.

DEFAULT: TRUE

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetCompression(false);  
$mpdf->WriteHTML('<p>Hallo World</p>');  
$mpdf->Output('filename.pdf');  
  
?>
```

SetCreator()

(mPDF >= 1.0)

SetCreator - Set the document Creator

Description

```
void SetCreator ( string $text )
```

Set the Creator for the document. This metadata can be seen when inspecting the document properties in Adobe Reader.

Note: Adobe Reader uses system fonts to display the document metadata, therefore any Unicode text can be used, even if a unibyte codepage is being used for the document.

Parameters

text

Defines the text to appear as a Creator. The text should be UTF-8 encoded, but should not contain HTML mark-up tags. [strcode2utf\(\)](#) is a useful function provided with mPDF which converts HTML numerical entities to UTF-8 encoded string.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetCreator('My Creator');  
$mpdf->WriteHTML('<p>Hallo World</p>');  
$mpdf->Output('filename.pdf');  
  
?>
```

Example #2

```
// html toolkit contains a function strcode2utf() to convert htmlentities to UTF-8 encoded text  
  
<?php  
  
$mpdf=new mPDF();  
  
$md = strcode2utf("ايلات &#1601;يما  
ايلات &#1601;يما");  
  
$mpdf->SetCreator($md);  
$mpdf->WriteHTML('<p>Hallo World</p>');  
$mpdf->Output('filename.pdf');  
  
?>
```

See Also

- [SetTitle\(\)](#) - Set document Title in metadata
- [SetAuthor\(\)](#) - Set document Author in metadata
- [SetSubject\(\)](#) - Set document Subject in metadata
- [SetKeywords\(\)](#) - Set document Keywords in metadata
- [strcode2utf\(\)](#) - Convert HTML numerical entities to UTF-8 encoded string

SetDefaultBodyCSS()

(mPDF >= 4.2)

SetDefaultBodyCSS - Change default CSS properties at runtime

Description

```
void SetDefaultBodyCSS ( string $property , string $value )
```

Change default CSS properties at runtime. This changes the default CSS stylesheet values for the BODY element.

Parameters

property

Specifies the CSS property to set. Any valid CSS property that mPDF supports for the BODY element e.g.
font-family, font-size, color
Case-insensitive

value

Specifies the value for the given property.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetDefaultBodyCSS('color', '#880000');  
$mpdf->WriteHTML('Hallow World should be in red');  
$mpdf->Output();  
  
?>
```

SetDefaultFont()

(mPDF <= 2.2)

SetDefaultFont — [depracated]

Note: Although this function is still available, it is not needed, nor recommended. Default font and other layout styles can be altered if required by editing the file mpdf.css

See [Default stylesheet](#) for more information.

SetDirectionality()

(mPDF >= 1.0)

SetDirectionality - Set the document RTL state

Description

```
void SetDirectionality ( string $dir )
```

Set the document RTL state. This defines the default alignment of tables, columns, text justification, page layout etc. See [RTL & Bidirectional Text](#) for more details. If SetDirectionality('rtl') is set, *Text Bidirectionality* is automatically turned on.

Parameters

dir

Defines the directionality of the document
BLANK or omitted (or any value other than RTL) sets the default of LTR

Values (case-insensitive)

LTR

RTL

DEFAULT: LTR

Examples

Example #1

```
<?php

$mpdf=new mPDF('utf-8');
$mpdf->SetDirectionality('rtl');
$mpdf->WriteHTML('<p>dag סקרן שט בים מאוכזב ולפטע מצא חברה</p>');
$mpdf->Output();

?>
```

SetDefaultFontSize()

(mPDF <= 2.2)

SetDefaultFontSize — [depracated]

Note: Although this function is still available, it is not needed, nor recommended. Default font size and other layout styles can be altered if required by editing the file mpdf.css

See [Default stylesheet](#) for more information.

SetDisplayMode()

(mPDF >= 1.0)

SetDisplayMode - Specify the initial Display Mode when the PDF file is opened in Adobe Reader

Description

```
void SetDisplayMode ( mixed $zoom [, string $layout ])
```

Specify the initial Display Mode when the PDF file is opened in Adobe Reader. When the user opens the finished file in Adobe Reader, these values will determine the initial appearance and layout.

Parameters

zoom

This parameter specifies the magnification (zoom) of the display when the document is opened.

Values (case-sensitive)

fullpage: Fit a whole page in the screen

fullwidth: Fit the width of the page in the screen

real: Display at real size

default: User's default setting in Adobe Reader

INTEGER : Display at a percentage zoom (e.g. 90 will display at 90% zoom)

layout

Specify the page layout to be used when the document is opened.

DEFAULT: "continuous"

Values (case-sensitive)

single: Display one page at a time

continuous: Display the pages in one column

two: Display the pages in two columns (first page determined by document direction (e.g. RTL))

twoleft: Display the pages in two columns, with the first page displayed on the left side (mPDF >= 5.2)

tworight: Display the pages in two columns, with the first page displayed on the right side (mPDF >= 5.2)

default: User's default setting in Adobe Reader

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetDisplayMode('fullwidth');  
$mpdf->WriteHTML('<p>Hello World</p>');  
$mpdf->Output('filename.pdf');  
  
?>
```

Example #2

```
// Display at 90% zoom - note the 90 is a number not a string  
$mpdf->SetDisplayMode(90);
```

```
// Display two pages side by side (book style)
$mpdf->SetDisplayMode('fullpage','two');
```

Changelog

Version	Description
1.0	Function was added.
5.2	2 options for layout parameter added (twoleft and tworight)

See Also

- [SetDisplayPreferences\(\)](#) - Defines the way the document shall be presented on the screen

SetDisplayPreferences()

(mPDF >= 3.0)

SetDisplayPreferences – Defines the way the document shall be presented on the screen

Description

```
void SetDisplayPreferences ( string $prefs )
```

Specify the way the document shall be presented on the screen when the PDF file is opened in Adobe Reader. When the user opens the finished file in Adobe Reader, these values will determine the initial appearance and layout.

Parameters

prefs

This parameter takes a string containing any one or more of the possible values separated by any characters (e.g. comma, forward slash or space). All the options start as **FALSE**, and can only be set **TRUE** by this command. Each setting is added to those previously set.

Values (case-sensitive)

FullScreen: Full-screen mode, with no menu bar, window controls, or any other window visible
HideMenubar: whether to hide the conforming reader's menu bar when the document is active
HideToolbar: whether to hide the conforming reader's tool bars when the document is active
HideWindowUI: whether to hide user interface elements in the document's window (such as scroll bars and navigation controls), leaving only the document's contents displayed
DisplayDocTitle: whether the window's title bar should display the document title taken from the Title entry of the document information dictionary. If false, the title bar will instead display the name of the PDF file containing the document
CenterWindow: whether to position the document's window in the center of the screen
FitWindow: whether to resize the document's window to fit the size of the first displayed page
NoPrintScaling: overrides the user's default setting to scale the printing size (mPDF >= 5.1)

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetDisplayPreferences('/HideMenubar/HideToolbar/DisplayDocTitle');  
$mpdf->WriteHTML('<p>Hallo World</p>');  
$mpdf->Output('filename.pdf');  
  
?>
```

See Also

- [SetDisplayMode\(\)](#) - Specify the initial Display Mode when the PDF file is opened in Adobe Reader

SetDocTemplate()

(mPDF >= 2.3)

SetDocTemplate - Specify an external PDF file to use as a template

Description

```
void SetDocTemplate ( [ string $file [, boolean $continue ]])
```

Specify an external PDF file to use as a template. Each page of the external source PDF file will be used as a template for the corresponding page in your new document. If the current mPDF document has more pages than the external PDF source document, the last page will (optionally) continue to be used for any remaining pages.

Parameters

file

This parameter specifies the source PDF file used as the template document. *file* should be a relative path to a local file.

DEFAULT: BLANK

continue = 1|0|TRUE|FALSE

If **TRUE** (or any positive value) it forces the last page of the source file to continue to be used as a template, if the current mPDF document contains more pages than the source file.

DEFAULT: FALSE

Note: If you want to turn the template off, just use `$mpdf->SetDocTemplate()` with no parameters.

Changelog

Version Description

2.3 Function was added.

Examples

Example #1

```
<?php
include("../mpdf.php");

$mpdf=new mPDF();
$mpdf->SetImportUse();

$mpdf->SetDocTemplate('logoheader.pdf',true);

// Do not add page until doc template set, as it is inserted at the start of each page
$mpdf->AddPage();

$mpdf->WriteHTML('Hallo World');

// Subsequent pages from logoheader.pdf will be inserted on all subsequent pages

$mpdf->Output();

?>
```

See Also

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [RestartDocTemplate\(\)](#) - Re-start the use of a Document template from the next page
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template

SetFooter()

(mPDF >= 1.0)

SetFooter - Sets a page header

Description

```
void SetFooter ([ mixed $footer [, string $side ]])
```

Set a page footer.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

footer

This parameter specifies the content of the page footer. It can accept a string or array. If a **BLANK** string or **NULL** or array() is passed, this will clear the page footer.

DEFAULT: array()

Values

A simple text string is set as content for the **RIGHT** margin. If **DOUBLE-SIDED** document, this is mirrored on **EVEN** pages i.e. **LEFT** margin.

A text string containing 2 characters '|' - will be split into three strings and set as content for the left|centre|right parts of the footer e.g. *header*="Chapter 1|{PAGENO}|Book Title".

If **DOUBLE-SIDED** document, this is mirrored on **EVEN** pages i.e. right|centre|left.

An array can be in two forms. The first form includes information for both **ODD** and **EVEN** footers, and is the expected form if *side* = **BLANK**.

Values

content: **TEXT STRING**

font-size: **FLOAT** font size in **pts**

font-style: **B|I|BI|BLANK STRING**

font-family: Any available font-family

color: CSS '#RRGGBB' string

line: 0|1 - specify whether to draw a line above the Footer

```
$footer = array (
    'odd' => array (
        'L' => array (
            'content' => '',
            'font-size' => 10,
            'font-style' => 'B',
            'font-family' => 'serif',
            'color'=>'#000000'
        ),
        'C' => array (
            'content' => '',
            'font-size' => 10,
            'font-style' => 'B',
            'font-family' => 'serif',
            'color'=>'#000000'
        ),
        'R' => array (
            'content' => 'My document',
            'font-size' => 10,

```

```

        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'line' => 1,
),
'even' => array ()
);

```

The second form includes information for either **ODD** or **EVEN** footers, and must be accompanied by a valid value for *side* = O|E

```

$footer = array (
'L' => array (
'content' => '',
'font-size' => 10,
'font-style' => 'B',
'font-family' => 'serif',
'color'=>'#000000'
),
'C' => array (
'content' => '',
'font-size' => 10,
'font-style' => 'B',
'font-family' => 'serif',
'color'=>'#000000'
),
'R' => array (
'content' => 'My document',
'font-size' => 10,
'font-style' => 'B',
'font-family' => 'serif',
'color'=>'#000000'
),
'line' => 1,
);

```

side

Specify whether to set the footer for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: BLANK

Values (case-sensitive)

O - set the footer for **ODD** pages

E - set the footer for **EVEN** pages

BLANK - sets both **ODD** or **EVEN** page footers

Changelog

Version	Description
---------	-------------

2.0	The <i>side</i> parameter was added.
-----	--------------------------------------

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 1](#)

See Also

- [SetHeader\(\)](#)
- [\\$defaultfooterfontsize](#)
- [\\$defaultfooterfontstyle](#)

- `$defaultfooterline`

SetFooterByName()

(mPDF >= 2.0)

SetFooterByName – Sets a page footer by a given name

Description

```
void SetFooterByName ( string $name [, string $side ] )
```

Sets a page footer that has previously been defined by name.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

This parameter specifies the name of a previously defined page footer. If a **BLANK** string or **NULL** is passed, mPDF will use the value '_default' if such a page footer exists.

side

Specify whether to set the footer for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'O'

Values (case-sensitive)

O - set the footer for **ODD** pages in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

E - set the footer for **EVEN** pages

DEFAULT - sets the footer for **ODD** in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 3](#)

See Also

- [DefFooterByName\(\)](#)
- [<pagefooter>](#)
- [SetHeaderByName\(\)](#)
- [<setpagefooter>](#)

SetHeader()

(mPDF >= 1.0)

SetHeader - Sets a page header

Description

```
void SetHeader ([ mixed $header [, string $side [, boolean $write ]]])
```

Set a page header.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

header

This parameter specifies the content of the page header. It can accept a string or array. If a **BLANK** string or **NULL** or **array()** is passed, this will clear the page header.

DEFAULT: **array()**

Values

A simple text string is set as content for the **RIGHT** margin. If **DOUBLE-SIDED** document, this is mirrored on **EVEN** pages i.e. **LEFT** margin.

A text string containing 2 characters '|' - will be split into three strings and set as content for the left|centre|right parts of the header e.g. *header*="Chapter 1|{PAGENO}|Book Title".

If **DOUBLE-SIDED** document, this is mirrored on **EVEN** pages i.e. right|centre|left.

An array can be in two forms. The first form includes information for both **ODD** and **EVEN** headers, and is the expected form if *side = BLANK*.

Values

content: TEXT STRING

font-size: FLOAT font size in **pts**

font-style: B|I|BI|BLANK STRING

font-family: Any available font-family

color: CSS '#RRGGBB' string

line: 0|1 - specify whether to draw a line under the Header

```
$header = array (
    'odd' => array (
        'L' => array (
            'content' => '',
            'font-size' => 10,
            'font-style' => 'B',
            'font-family' => 'serif',
            'color'=>'#000000'
        ),
        'C' => array (
            'content' => '',
            'font-size' => 10,
            'font-style' => 'B',
            'font-family' => 'serif',
            'color'=>'#000000'
        ),
        'R' => array (
            'content' => 'My document',
            'font-size' => 10,

```

```

        'font-style' => 'B',
        'font-family' => 'serif',
        'color'=>'#000000'
    ),
    'line' => 1,
),
'even' => array ()
);

```

The second form includes information for either **ODD** or **EVEN** headers, and must be accompanied by a valid value for *side* = O|E

```

$header = array (
'L' => array (
'content' => '',
'font-size' => 10,
'font-style' => 'B',
'font-family' => 'serif',
'color'=>'#000000'
),
'C' => array (
'content' => '',
'font-size' => 10,
'font-style' => 'B',
'font-family' => 'serif',
'color'=>'#000000'
),
'R' => array (
'content' => 'My document',
'font-size' => 10,
'font-style' => 'B',
'font-family' => 'serif',
'color'=>'#000000'
),
'line' => 1,
);

```

side

Specify whether to set the header for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: BLANK

Values (case-sensitive)

O - set the header for **ODD** pages

E - set the header for **EVEN** pages

BLANK - sets both **ODD** or **EVEN** page headers

write

If **TRUE** it forces the Header to be written immediately to the current page. Use if the header is being set after the new page has been added.

DEFAULT: FALSE

Note: *write* forces the appropriate header to be written. If you have just defined an **ODD**-sided header and the document is currently writing to an **EVEN**-sided page, the **EVEN** header will be output.

Changelog

Version Description

2.0 The *side* and *write* parameters were added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 1](#)

See Also

- [SetFooter\(\)](#)
- [\\$defaultheaderfontsize](#)
- [\\$defaultheaderfontstyle](#)
- [\\$defaultheaderline](#)

SetHeaderByName()

(mPDF >= 2.0)

SetHeaderByName – Sets a page header by a given name

Description

```
void SetHeaderByName ( string $name [, string $side [, boolean $write ]])
```

Sets a page header that has previously been defined by name.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

This parameter specifies the name of a previously defined page header. If a **BLANK** string or **NULL** is passed, mPDF will use the value '_default' if such a page header exists.

side

Specify whether to set the header for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'O'

Values (case-sensitive)

O - set the header for **ODD** pages in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

E - set the header for **EVEN** pages

DEFAULT - sets the header for **ODD** in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

write

If **TRUE** it forces the Header to be written immediately to the current page. Use if the header is being set after the new page has been added.

DEFAULT: FALSE

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 3](#)

See Also

- [DefHeaderByName\(\)](#)
- [<pageheader>](#)

- [SetFooterByName\(\)](#)
- [`<setpageheader>`](#)

SetHTMLFooter()

(mPDF >= 1.2)

SetHTMLFooter – Sets an HTML page footer

Description

```
void SetHTMLFooter ( string $html [, string $side ])
```

Set an HTML page footer.

Parameters

header

This parameter specifies the content of the page footer as a string of valid HTML code.

DEFAULT: BLANK

side

Specify whether to set the footer for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'O'

Values (case-sensitive)

O - set the footer for **ODD** pages

E - set the footer for **EVEN** pages

BLANK - set the footer for **ODD** pages

Note: Important Difference - [SetHeader\(\)](#) and [SetFooter\(\)](#) called without specifying a *side* sets both **ODD** & **EVEN** headers/footers; [SetHTMLHeader\(\)](#) and [SetHTMLFooter\(\)](#) without a *side* - set **ODD** page header/footer only as default

Changelog

Version	Description
1.2	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 2](#)

See Also

- [SetHTMLHeader\(\)](#)

SetHTMLFooterByName()

(mPDF >= 2.0)

SetHTMLFooterByName – Sets an HTML page footer by a given name

Description

```
void SetHTMLFooterByName ( string $name [, string $side ])
```

Sets an HTML page footer that has previously been defined by name.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

This parameter specifies the name of a previously defined HTML page footer. If a **BLANK** string or **NULL** is passed, mPDF will use the value '_default' if such a page footer exists.

side

Specify whether to set the footer for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'O'

Values (case-sensitive)

O - set the footer for **ODD** pages in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

E - set the footer for **EVEN** pages

DEFAULT - sets the footer for **ODD** in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 4](#)

See Also

- [DefHTMLFooterByName\(\)](#)
- [`<htmlpagefooter>`](#)
- [SetHTMLHeaderByName\(\)](#)
- [`<sethtmlpagefooter>`](#)

SetHTMLHeader()

(mPDF >= 1.2)

SetHTMLHeader - Sets an HTML page header

Description

```
void SetHTMLHeader ( string $html [, string $side [, boolean $write ]])
```

Set an HTML page header.

Parameters

header

This parameter specifies the content of the page header as a string of valid HTML code.

DEFAULT: BLANK

side

Specify whether to set the header for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: BLANK

Values (case-sensitive)

O - set the header for **ODD** pages

E - set the header for **EVEN** pages

BLANK - sets **ODD** page headers

write

If **TRUE** it forces the Header to be written immediately to the current page. Use if the header is being set after the new page has been added.

DEFAULT: FALSE

Note: Important Difference - [SetHeader\(\)](#) and [SetFooter\(\)](#) called without specifying a *side* sets both **ODD** & **EVEN** headers/footers; [SetHTMLHeader\(\)](#) and [SetHTMLFooter\(\)](#) without a *side* - sets **ODD** page header/footer only as default

Changelog

Version	Description
1.2	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 2](#)

See Also

- [SetHTMLFooter\(\)](#)

SetHTMLHeaderByName()

(mPDF >= 2.0)

SetHTMLHeaderByName - Sets an HTML page header by a given name

Description

```
void SetHTMLHeaderByName ( string $name [, string $side [, boolean $write ]] )
```

Sets an HTML page header that has previously been defined by name.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

This parameter specifies the name of a previously defined HTML page header. If a **BLANK** string or **NULL** is passed, mPDF will use the value '`_default`' if such a page header exists.

side

Specify whether to set the header for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'O'

Values (case-sensitive)

O - set the header for **ODD** pages in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

E - set the header for **EVEN** pages

DEFAULT - sets the header for **ODD** in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

write

If **TRUE** it forces the Header to be written immediately to the current page. Use if the header is being set after the new page has been added.

DEFAULT: FALSE

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 4](#)

See Also

- [DefHTMLHeaderByName\(\)](#)
- [<htmlpageheader>](#)

- [SetHTMLFooterByName\(\)](#)
- [`<sethtmlpageheader>`](#)

SetImportUse()

(mPDF >= 4.3)

SetImportUse - Enable the use of imported PDF files or templates

Description

```
void SetImportUse( )
```

Enable the use of imported PDF files or templates. This causes additional files (classes) to be loaded, enabling several functions allowing you to import PDF files into the document you are writing, and using templates.

Note: Prior to mPDF 4.3, this required calling mPDFI(). The functions have now been incorporated into the main mpdf.php file, but you must use SetImportUse() to enable them.

Changelog

Version	Description
4.3	Function added.

Examples

Example #1

```
<?php  
  
include("../mpdf.php");  
  
$mpdf=new mPDF();  
$mpdf->SetImportUse();  
  
$mpdf->WriteHTML('You can use this just like normal, but also import and use templates...');  
  
$mpdf->Output();  
  
?>
```

See Also

- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [ImportPage\(\)](#) - Import a page from an external PDF file
- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template
- [RestartDocTemplate\(\)](#) - Re-start the use of a Document template from the next page

SetKeywords()

(mPDF >= 1.0)

SetKeywords - Set the document metadata Keywords

Description

```
void SetKeywords ( string $text )
```

Set Keywords for the document metadata. This metadata can be seen when inspecting the document properties in Adobe Reader.

Note: Adobe Reader uses system fonts to display the document metadata, therefore any Unicode text can be used, even if a unibyte codepage is being used for the document.

Parameters

text

Defines the text to appear as Keywords. The text should be UTF-8 encoded, but should not contain HTML mark-up tags. [strcode2utf\(\)](#) is a useful function provided with mPDF which converts HTML numerical entities to UTF-8 encoded string.

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->SetKeywords('My Keywords, More Keywords');
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');

?>
```

Example #2

```
// html toolkit contains a function strcode2utf() to convert htmlentities to UTF-8 encoded text

<?php

$mpdf=new mPDF();

$md = strcode2utf("ايلات &#1601;يما
&#1575;يلات &#1601;يما");

$mpdf->SetKeywords($md);
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');

?>
```

See Also

- [SetTitle\(\)](#) - Set document Title in metadata
- [SetAuthor\(\)](#) - Set document Author in metadata
- [SetCreator\(\)](#) - Set document Creator in metadata
- [SetSubject\(\)](#) - Set document Subject in metadata
- [strcode2utf\(\)](#) - Convert HTML numerical entities to UTF-8 encoded string

SetPageTemplate()

(mPDF >= 2.3)

SetPageTemplate – Specify a page from an external PDF file to use as a template

Description

```
void SetPageTemplate ([ int $templateID ])
```

Specify a page from an external PDF file to use as a template. The page must have already been stored as a 'template' using [SetSourceFile\(\)](#). Once a 'page template' has been set, the template is inserted on every subsequent page of the document. The template is added to the page at the same time as a new page is started (in the Header).

Parameters

templateID

This parameter specifies the ID of the page template to use. Value must be a valid template ID from [SetSourceFile\(\)](#).

DEFAULT or **BLANK** will clear the template, so subsequent pages will not have the template added.

Changelog

Version	Description
2.3	Function was added.

Examples

Example #1

```
<?php

include("../mpdf.php");

$mpdf=new mPDF();
$mpdf->SetImportUse();

$pagecount = $mpdf->SetSourceFile('logoheader.pdf');
$tplId = $mpdf->ImportPage($pagecount);

$mpdf->SetPageTemplate($tplId);

// Do not add page until page template set, as it is inserted at the start of each page
$mpdf->AddPage();

$mpdf->WriteHTML('Hallo World');

// The template $tplId will be inserted on all subsequent pages until (optionally)
// $mpdf->SetPageTemplate();

$mpdf->Output();

?>
```

See Also

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document

- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template

SetProtection()

(mPDF >= 1.0)

SetProtection - Encrypts and sets the PDF document permissions

Description

```
void SetProtection ( array $permissions [, string $user_password [, string $owner_password [, integer $length ]]] )
```

Encrypts and sets the PDF document permissions for the PDF file, together with user and owner passwords.

Note: A default mPDF document is not encrypted, and grants full permissions to the end-user e.g. copying, printing, modifying.

Parameters

permissions

This parameter is an array which specifies the permissions granted to the end-user. A blank array should be passed to deny all permissions to the user. The latter 4 permissions were added in mPDF >=5.3
Using any of these last 4 permissions require 128-bit encryption and will force this mode, regardless of any value set for *length*.

Note: If 128-bit encryption is used (whether by specifying *length*=128 or by using any of the 4 latter permissions), the use of *print* will only allow low-resolution printing from the document; you must specify *print-highres* to allow full resolution printing.

Values (case-sensitive)

An array including any, all or none of the following. The values included are those permissions allowed:

copy
print
modify
annot-forms

fill-forms
extract
assemble
print-highres

user_password

Specify a password required for a user to open the PDF file.

BLANK or omitted - No password is required to open the PDF document.

owner_password

Specify a password which will allow full access and permissions to the PDF file.

If omitted - A random password is generated by mPDF

length

Specify the bit-length used for encryption. Two values are possible, 40 and 128. The 4 latter permissions (see above) require 128-bit encryption, and setting any of these will automatically set *length* as 128,

overriding any value specified.

DEFAULT: 40 - use 40-bit encryption

VALUES

40

128

Changelog

Version	Description
2.5	cjk files can be encrypted
3.2	Empty (blank array) <i>permissions</i> array correctly handled.
5.3	Additional <i>permissions</i> added, and <i>length</i> parameter added enabling 128-bit encryption

Examples

Example #1

```
<?php

$mpdf=new mPDF();
// Encrypt the file and grant no permissions to the user to copy, print etc.
// The user will be able to open the file as no password is specified
// Owner cannot access full rights because no owner_password was set
$mpdf->SetProtection(array());
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');

?>
```

Example #2

```
// Encrypt the file and grant no permissions to the user
// The user will need to use "UserPassword" to open the file
// Owner has full rights using the password "MyPassword"
$mpdf->SetProtection(array(), 'UserPassword', 'MyPassword');

// Encrypt the file and grant permissions to the user to copy and print
// No password is required to open the document
// Owner has full rights using the password "MyPassword"
$mpdf->SetProtection(array('copy','print'), '', 'MyPassword');
```

See Also

- [SetUserRights\(\)](#)

SetSourceFile()

(mPDF >= 2.3)

SetSourceFile - Specify the source PDF file used to import pages into the document

Description

```
int SetSourceFile ( string $file )
```

Specify the source PDF file used to import pages into the document.

Parameters

file

This parameter specifies the source PDF file used to import pages into the document. **Note:** *file* should be a relative path to a local file.

Return Value

SetSourceFile() returns the number of pages in the source file.

Changelog

Version	Description
---------	-------------

2.3	Function was added.
-----	---------------------

Examples

Example #1

```
<?php  
  
include("../mpdf.php");  
  
$mpdf=new mPDF();  
$mpdf->SetImportUse();  
  
$pagecount = $mpdf->SetSourceFile('logoheader.pdf');  
  
// Import the last page of the source PDF file  
$tplId = $mpdf->ImportPage($pagecount);  
  
$mpdf->UseTemplate($tplId);  
  
$mpdf->WriteHTML('Hallo World');  
  
$mpdf->Output();  
  
?>
```

See Also

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [ImportPage\(\)](#) - Import a page from an external PDF file

- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template

SetSubject()

(mPDF >= 1.0)

SetSubject - Set the document metadata Subject

Description

```
void SetSubject ( string $text )
```

Set the Subject for the document. This metadata can be seen when inspecting the document properties in Adobe Reader.

Note: Adobe Reader uses system fonts to display the document metadata, therefore any Unicode text can be used, even if a unibyte codepage is being used for the document.

Parameters

text

Defines the text to appear as the document Subject. The text should be UTF-8 encoded, but should not contain HTML mark-up tags. [strcode2utf\(\)](#) is a useful function provided with mPDF which converts HTML numerical entities to UTF-8 encoded string.

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->SetSubject('My Subject');
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');

?>
```

Example #2

```
// htmltoolkit contains a function strcode2utf() to convert htmlentities to UTF-8 encoded text

<?php

$mpdf=new mPDF();

$md = strcode2utf("ايلات &#1601;يما
&#1575;يلات &#1601;يما");

$mpdf->SetSubject($md);
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');

?>
```

See Also

- [SetTitle\(\)](#) - Set document Title in metadata
- [SetAuthor\(\)](#) - Set document Author in metadata
- [SetCreator\(\)](#) - Set document Creator in metadata
- [SetKeywords\(\)](#) - Set document Keywords in metadata
- [strcode2utf\(\)](#) - Convert HTML numerical entities to UTF-8 encoded string

SetTitle()

(mPDF >= 1.0)

SetTitle – Set the document title

Description

```
void SetTitle ( string $text )
```

Set the title for the document. The title is displayed at the top of the Adobe Reader screen when viewing the PDF file, and is included in the document metadata, which can be seen when inspecting the document properties in Adobe Reader.

Note: Adobe Reader uses system fonts to display the document metadata, therefore any Unicode text can be used, even if a unibyte codepage is being used for the document.

Note: The *title* tag from the header of an HTML document will override this value when you use [WriteHTML\(\)](#).

Parameters

text

Defines the text to appear as a Title. The text should be UTF-8 encoded, but should not contain HTML mark-up tags. [strcode2utf\(\)](#) is a useful function provided with mPDF which converts HTML numerical entities to UTF-8 encoded string.

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->SetTitle('My Title');
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');

?>
```

Example #2

```
// html toolkit contains a function strcode2utf() to convert htmlentities to UTF-8 encoded text

<?php

$mpdf=new mPDF();

$md = strcode2utf("ايلات &#1601;يما
&#1575;يلات &#1601;يما");

$mpdf->SetTitle($md);
$mpdf->WriteHTML('<p>Hallo World</p>');
$mpdf->Output('filename.pdf');

?>
```

See Also

- [SetAuthor\(\)](#) - Set document Author in metadata
- [SetCreator\(\)](#) - Set document Creator in metadata
- [SetSubject\(\)](#) - Set document Subject in metadata
- [SetKeywords\(\)](#) - Set document Keywords in metadata
- [strcode2utf\(\)](#) - Convert HTML numerical entities to UTF-8 encoded string

SetVisibility()

(mPDF >= 5.4)

SetVisibility – Set the visibility of subsequent objects

Description

```
void SetVisibility ( string $visibility )
```

Set the visibility of subsequent objects

Parameters

visibility

This parameter specifies the visibility.

Values

STRING - One of the following:

visible
hidden
printonly
screenonly

DEFAULT: visible

Changelog

Version	Description
5.4	Function was added.

Examples

Example #1

```
<?php

...

$mpdf->SetVisibility('printonly');
$mpdf->WriteHTML('<p>This text will only be visible when the document is printed</p>');
$mpdf->SetVisibility('screenonly');
$mpdf->WriteHTML('<p>This text will only be visible on screen, and will not be included when the
document is printed</p>');
...

?>
```

See Also

- [Supported CSS - 'visibility'](#)

SetWatermarkImage()

(mPDF >= 2.2)

SetWatermarkImage - Set an image to use as a Watermark

Description

```
void SetWatermarkImage ( string $src [, float $alpha [, mixed $size [, mixed $position ]]] )
```

Set an image to use as a Watermark. The watermark is a semi-transparent background printed on each page, used for text such as "DRAFT" or a background image. The watermark will be added to each page when the Footer is printed if the variable `showWatermarkImage` is set to 1 or **TRUE**.

Parameters

src

This parameter specifies the image file to use for the watermark. This can be a full URI or use a relative path

alpha

This parameter defines the transparency value (alpha) to use for the watermark. The Value should be between 0 and 1.

DEFAULT: 0.2

size

This parameter takes either a pre-defined string, an integer, or an array of width and height. Defines the size of the watermark.

Values

D: default i.e. original size of image - may depend on `img_dpi`

P: Resize to fit the full page size, keeping aspect ratio

F: Resize to fit the print-area (frame) respecting current page margins, keeping aspect ratio

INT: Resize to full page size minus a margin set by this integer in millimeters, keeping aspect ratio
`array($width, $height)`: Specify a size; units in millimeters

DEFAULT: "D"

position

This parameter takes either a pre-defined string or an array of x and y. Defines the position of the watermark on the page.

Values

P: Centred on the whole page area

F: Centred on the page print-area (frame) respecting page margins

`array($x, $y)`: Specify a position; units in millimeters

DEFAULT: "P"

Changelog

Version	Description
2.2	The function was added.

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->SetWatermarkImage('../images/background.jpg');
$mpdf->showWatermarkImage = true;
$mpdf->WriteHTML('<p>Hallo World</p>');

?>
```

Example #2 - Using a Watermark as a Header

```
<?php

$mpdf=new mPDF('','A4','','','',20,20,50,10);
// Setting transparency to 1, and exact positioning, you can use a Watermark Image as a 'Header'
// Note that the page top-margin is set to accomodate the image
$mpdf->SetWatermarkImage('http://www.yourdomain.com/images/logo.jpg', 1, '', array(160,10));
$mpdf->showWatermarkImage = true;
$mpdf->WriteHTML('<p>Hallo World</p>');

?>
```

See Also

- [SetWatermarkText\(\)](#) - Set the text to use as a Watermark
- [watermarkImageAlpha](#) - Specifies the transparency (alpha value) for the watermark image
- [watermarkTextAlpha](#) - Specifies the transparency (alpha value) for the watermark text
- [showWatermarkText](#) - Specifies whether or not to show/print the watermark text
- [showWatermarkImage](#) - Specifies whether or not to show/print the watermark image
- [watermark_font](#) - Specifies the font to use for Watermark text

SetWatermarkText()

(mPDF >= 2.2)

SetWatermarkText – Set the text to use as a Watermark

Description

```
void SetWatermarkText ( [ string $text [, float $alpha ]])
```

Set the text to use as a Watermark. The watermark is a semi-transparent background printed on each page, used for text such as "DRAFT". The watermark will be added to each page when the Footer is printed if the variable `showWatermark` is set to 1 or **TRUE**.

Note: Prior to mPDF 2.2 the function `setUnvalidatedText()` was used. `SetWatermarkText()` is now the preferred form.

Parameters

`text`

This parameter defines the text to use for the watermark. The text should be UTF-8 encoded, but should not contain HTML mark-up tags. If the text is blank, it will clear the watermark text, so nothing appears.

DEFAULT: BLANK

`alpha`

This parameter defines the transparency value (alpha) to use for the watermark: either text or image. The Value should be between 0 and 1.

DEFAULT: 0.2

Changelog

Version	Description
2.2	The function was added.
2.2	Parameter <code>alpha</code> was added
3.0	Parameter <code>text</code> changed to be optional
3.0	No parameters are required from v3.0 onwards e.g. \$mpdf->SetWatermarkText(); can be used as well as: \$mpdf->SetWatermarkText('');

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->SetWatermarkText('DRAFT');
$mpdf->showWatermarkText = true;
$mpdf->WriteHTML('<p>Hallo World</p>');

?>
```

Example #2

```
// html toolkit contains a function strcode2utf() to convert htmlentities to UTF-8 encoded text
```

See Also

- <watermarktext> - HTML equivalent of SetWatermarkText()
 - **SetWatermarkImage()** - Set an image to use as a Watermark
 - **watermarkImageAlpha** - Specifies the transparency (alpha value) for the watermark image
 - **watermarkTextAlpha** - Specifies the transparency (alpha value) for the watermark text
 - **showWatermarkText** - Specifies whether or not to show/print the watermark text
 - **showWatermarkImage** - Specifies whether or not to show/print the watermark image
 - **watermark_font** - Specifies the font to use for Watermark text

Shaded_box()

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Shaded_box(string title[, string font[, float fontstyle[, float fontsize[, float width[, string style[, float radius[, string backgroundcolor[, string color[, float padding]]]]]]]]])

Writes a single line of text surrounded by a box directly to the PDF document at the current position. The box can have rounded corners, and be filled with background-colour.

- **title**
UTF-8 encoded text (single line)
- **font**
Font family
Default = "" i.e. default document font
- **fontstyle**
Font style as one of B (bold), I (italic), BI (bold-italic) or blank for normal
Default = 'B' i.e. bold
- **fontsize**
Font size in points (pt)
Default = "" i.e. default document font size
- **width**
Width of the box - any units acceptable in CSS can be used e.g. pt, px, mm, % (of page width)
Default = '70%'
- **style**
Box style: D or empty string - draw border (default); F - fill; DF or FD - draw and fill
Default = 'DF' i.e. border and fill
- **radius**
Radius of the rounded corners
Default = 2.5
- **backgroundcolor**
Fill colour for the box - as #rrggbb
Default = '#FFFFFF'
- **color**
Text colour - as #rrggbb
Default = '#000000'
- **padding**
Padding between text and box border, in millimeters
Default = 2

This method accepts UTF-8 encoded text, and will reverse RTL (right-to-left) text when appropriate.

Text containing HTML entities, as well as decimal and hex e.g. ' & #8812; or & #x21a4; can be used, by setting:

`$mpdf->text_input_as_HTML = true; (default = false)`

This will convert all the above to their appropriate characters, otherwise the text will be output as it is.

StartProgressBarOutput()

(mPDF >= 4.2)

StartProgressBarOutput - Enable progress bars to be shown during file generation

Description

```
void StartProgressBarOutput ( [ string $mode ] )
```

Enable progress bars to be shown during file generation. Not recommended for general use, but may be helpful for development purposes, or for slow document generation. To set this value globally you can edit the value for `progressBar` in the configuration file `config.php`

Note: You may need to define the constant `_MPDF_URI` if you use progress bars. The constant `_MPDF_URI` is needed to redirect the user to the PDF file (and prior to mPDF 5.0 to locate a javascript file within the progress bar script). It must be either a relative path (e.g. `'../'`) or a full URI (e.g. `'http://www.mydomain.com/mpdf/'`). If you do not define it before calling `mPDF()` mPDF will assign it the same value as `_MPDF_PATH`. This is fine if you have used a relative path. `_MPDF_PATH` requires either a relative path or a filesystem real path (e.g. `'/homepages/27/d84233457/htdocs/'`)

Parameters

`mode`

Values (case-insensitive)

1 or **BLANK** or omitted: Shows 1 progress bar (simple form) = **DEFAULT**

2: Shows multiple progress bars for detailed examination of progress

Examples

Example #1

```
<?php

define('_MPDF_URI','../');
// must be a relative or absolute URI - not a file system path

$mpdf=new mPDF();
$mpdf->StartProgressBarOutput(2);
$mpdf->WriteHTML('You will hardly have time to see the progress bars!');
$mpdf->Output();

?>
```

See Also

- `progressBar` -Specify whether to show progress bars during file generation
- `progbar_heading` - define customised heading for progress bars
- `progbar_altHTML` - define customised HTML for progress bars

Thumbnail()

(mPDFI >= 2.3)

Thumbnail - Create thumbnails of an external PDF file and insert in current document

Description

```
void Thumbnail ( string $file [, integer $numberperrow [, float $spacing ]])
```

Create thumbnails of an external PDF file and insert in current document.

Parameters

file

This parameter specifies the source PDF file to import. *file* should be a relative path to a local file.

numberperrow

number specifies the number of thumbnails to print in each **row**.

DEFAULT: 3

spacing

Specifies the spacing (vertical and horizontal) between thumbnails in millimetres.

DEFAULT: 10

Changelog

Version Description

Version	Description
2.3	Function was added.

Examples

Example #1

```
<?php
include("../mpdf.php");

$mpdf=new mPDF();
$mpdf->SetImportUse();

$mpdf->Thumbnail('testfile.pdf', 4);

$mpdf->Output();

?>
```

See Also

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [ImportPage\(\)](#) - Import a page from an external PDF file
- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template
- [RestartDocTemplate\(\)](#) - Re-start the use of a Document template from the next page

TOCpagebreak()

(mPDF >= 2.0)

TOCpagebreak – Insert a table of contents in the document

Note: A new function `TOCpagebreakByArray()` was added in mPDF 5.0 which is recommended as much simpler to use.

Description

Add a new page to the document, marking the point at which a Table of Contents (**ToC**) will be inserted in the document at the end of writing. The numerous parameters specify both paging details for the continuing document, and for the **ToC** when it is generated.

Note: From mPDF 5.7 the layout of a table of contents can be controlled using CSS. *font*, *font-size* and *indent* have become redundant. They are kept as parameters to allow backwards compatibility, but any values set for these are ignored.

Note: When writing a **DOUBLE-SIDED** document, the **ToC** will always start on an **ODD** page. Therefore there is no option to specify the pagebreak type as in [AddPage\(\)](#) - using **TOCpagebreak()** will always continue the document on an **ODD** page.

Note: Page numbering is always suppressed in the **ToC**.

Note: The **ToC** is generated at the end of the document. Unless otherwise specified, the **ToC** will inherit the page margins, headers/footers and orientation of the last page written to the document.

Note: From mPDF 2.3 you can include more than one **Toc** in the document using the parameter *name*.

Note: If **TOCpagebreak()** is used at the start of a blank (**ODD**) page, no new page(s) will be added. This was added in mPDF 2.3 to allow a **ToC** to be placed on the first page, or to allow a **ToC** to follow another **ToC**. In this case, any properties for the continuing document are ignored. If you define several **ToCs** following immediately on from one another, set the properties in the first **ToC** you define.

Parameters

The initial parameters specify characteristics for the **TOC**, which is generated automatically at the end of the document when **Output()** is called.

font

From mPDF >= 5.7 CSS styles are used to control layout of **ToCs**. Any value set for this is ignored.

Set the font-family for the **ToC**.

BLANK or omitted uses default font-family for the document.

font-size

From mPDF >= 5.7 CSS styles are used to control layout of **ToCs**. Any value set for this is ignored.

Sets the font size for the **ToC** in **points** (pt)

BLANK or omitted or 0 uses the default font-size for the document.

indent

From mPDF >= 5.7 CSS styles are used to control layout of **ToCs**. Any value set for this is ignored.

Sets the value in millimetres to indent each level of the **ToC** from the left margin.

BLANK or omitted uses a default value of 5mm.

paging = **TRUE|1|FALSE|0**

Specify whether to show page numbers in the **ToC**.

BLANK or omitted uses a default value of **TRUE**.

DEFAULT: TRUE

Values

TRUE or 1: show page numbers in the **ToC**.

FALSE or 0: do not show page numbers in the **ToC**.

links = **TRUE|1|FALSE|0**

Specify whether to generate hyperlinks in the **ToC**.

BLANK or omitted uses a default value of **FALSE**.

DEFAULT: FALSE

Values

TRUE or 1: show hyperlinks in the **ToC**.

FALSE or 0: do not show hyperlinks in the **ToC**.

toc-orientation

This attribute specifies the orientation of the **ToC** pages.

BLANK or omitted leaves the orientation unchanged i.e. at the end of the document (before the **ToC** is generated)

Values (case-insensitive)

L or landscape: Landscape

P or portrait: Portrait

toc-margin-left

toc-margin-right

toc-margin-top

toc-margin-bottom

toc-margin-header

toc-margin-footer

Set the page margins for the **ToC**.

All values should be specified as **LENGTH** in millimetres.

If you are writing a **DOUBLE-SIDED** document, the margin values will be used for **ODD** pages; left and right margins will be mirrored for **EVEN** pages.

BLANK or omitted leaves the current margin unchanged i.e. the margins current at the end of the document.

"0" (zero) will set the margin to zero.

toc-odd-header-name

toc-even-header-name

toc-odd-footer-name

toc-even-footer-name

Selects a header or footer by name to use for the **ToC**. The header/footer must already have been defined using [DefHeaderByName\(\)](#), [DefFooterByName\(\)](#), [DefHTMLHeaderByName\(\)](#), or [DefHTMLFooterByName\(\)](#). If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted leaves the header/footer unchanged. NB **BLANK** will not unset the header. Set *toc-odd-header-value* to -1 to turn the header off.

Note: You must add the prefix 'html_' before the name if it is a HTMLHeader.

toc-odd-header-value

toc-even-header-value

toc-odd-footer-value

toc-even-footer-value

Specify whether to show a header or footer in the **ToC**. The header/footer must already have been defined using [DefHeaderByName\(\)](#), [DefFooterByName\(\)](#), [DefHTMLHeaderByName\(\)](#), or [DefHTMLFooterByName\(\)](#).

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted or 0 leaves the header/footer state unchanged.

Values (case-insensitive)

1 or on: Show the selected header/footer in the **ToC**.

-1 or off: Hide the selected header/footer in the **ToC**.

toc-preHTML

Specify the HTML code to appear before the **ToC** e.g. '<h1>Contents</h1>'. Note that in contrast with the HTML equivalent <[tocpagebreak](#)> the text does not need to use HTML-entities.

BLANK or omitted will enter no text

toc-postHTML

Specify the HTML code to appear after the **ToC** e.g. '<p>Comments to go below the ToC</p>'. Note that in contrast with the HTML equivalent <[tocpagebreak](#)> the text does not need to use HTML-entities.

BLANK or omitted will enter no text.

toc-bookmarkText

Specify the text as it will appear as a **BOOKMARK** for the **ToC** e.g. 'Content list'.

BLANK or omitted will not create a **BOOKMARK**.

name

Specify which **Toc** to include at this point, if using more than one **Toc** in the document. *name* can be any alphanumeric characters (except just "0") and is case-insensitive.
BLANK or omitted or 0 uses the default **ToC**.

toc-pageselector

Select a named CSS @page for the **ToC**.
BLANK or omitted or leaves the CSS page unchanged.

See [Using @page](#) for more information

toc-sheet-size

sheet-size can be specified either as a pre-defined page size, or as an array of width and height in millimetres e.g. array(210,297).

DEFAULT: **BLANK** - makes no change to the current sheet-size

Values (case-insensitive)

A0 - A10, B0 - B10, C0 - C10
 4A0, 2A0, RA0 - RA4, SRA0 - SRA4
 Letter, Legal, Executive, Folio
 Demy, Royal
 A (Type A paperback 111x178mm)
 B (Type B paperback 128x198mm)

All of the above values can be suffixed with "-L" to force a Landscape page orientation document e.g. "A4-L"

outdent

Set a negative indent for the last line of each **ToC** entry.

Values should be **BLANK** string or any valid CSS **LENGTH**.

This will cause the line to extend beyond the right margin; you can prevent this by setting **PADDING-RIGHT** equal to this value.

DEFAULT 0

toc-resetpagenum = 1 - ∞

Sets/resets the document page number to *resetpagenum* starting on the **ToC**. (The value must be a positive integer).

BLANK or omitted or 0 leaves the preceding page number sequence unchanged.

toc-pagenumstyle = 1|A|a|I|i|[+ any value supported for list-style-type]

Sets/resets the page numbering style to use in the **ToC** (values as for cf. [lists](#))
BLANK or omitted leaves the current page number style unchanged.

Values (case-sensitive)

1: Decimal - 1,2,3,4...
 A: Alpha uppercase - A,B,C,D...
 a: Alpha lowercase - a,b,c,d...
 I: Roman uppercase - I, II, III, IV...
 i: Roman lowercase - i, ii, iii, iv...

toc-suppress = **TRUE|FALSE|1|0**

suppress=on will suppress document page numbers in the **ToC**

BLANK or omitted leaves the current condition unchanged.

Values (case-insensitive)

1 or **TRUE**: Suppress (hide) page numbers in the **ToC**

0 or **FALSE**: Show page numbers in the **ToC**

The rest of the parameters are defined exactly as for [AddPage\(\)](#). Note that these parameters define page numbering, margins, headers/footers for the document as it continues from this point on; in the final document this will be the part of the document immediately after the **ToC**.

Please refer to [AddPage\(\)](#) for further details. But **note** there are differences in the order of the parameters especially take care with \$orientation

Changelog

Version	Description
2.0	Function was added.
2.2	Default values for <i>font-size</i> , <i>paging</i> and <i>links</i> were redefined.
2.3	<i>name</i> attribute was added.
4.3	Parameters <i>pageselector</i> , <i>sheet-size</i> , <i>toc-pageselector</i> and <i>toc-sheet-size</i> were added
5.7	<i>outdent</i> parameter added <i>font</i> , <i>font-size</i> and <i>indent</i> redundant
6.0	Parameters added: <i>toc-resetpagenum</i> <i>toc-pagenumstyle</i> <i>toc-suppress</i>

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->WriteHTML('Introduction');
$mpdf->TOCpagebreak();

$mpdf->TOC_Entry("Chapter 1",0);
$mpdf->WriteHTML('Chapter 1 ...');
$mpdf=Output();

?>
```

See Also

- [TOC_Entry\(\)](#) - Add an entry for Table of Contents
- [AddPage\(\)](#) - Add a new page
- <[tocpagebreak](#)> - Custom HTML tag - equivalent to **TOCpagebreak()**

TOCpagebreakByArray()

(mPDF >= 5.0)

TOCpagebreakByArray — Insert a table of contents in the document using an array of parameters

Description

```
void TOCpagebreakByArray ([ array $arr ])
```

Add a new page to the document using an array of (optional) parameters, marking the point at which a Table of Contents (**ToC**) will be inserted in the document at the end of writing. The numerous parameters specify both paging details for the continuing document, and for the **ToC** when it is generated.

Note: From mPDF 5.7 the layout of a table of contents can be controlled using CSS. *font font-size* and *indent* have become redundant.

Note: When writing a **DOUBLE-SIDED** document, the **ToC** will always start on an **ODD** page. Therefore there is no option to specify the pagebreak type as in [AddPage\(\)](#) - using **TOCpagebreakByArray()** will always continue the document on an **ODD** page.

Note: Page numbering is always suppressed in the **ToC**.

Note: The **ToC** is generated at the end of the document. Unless otherwise specified, the **ToC** will inherit the page margins, headers/footers and orientation of the last page written to the document.

Note: You can include more than one **ToC** in the document using the parameter *name*.

Note: If **TOCpagebreakByArray()** is used at the start of a blank (**ODD**) page, no new page(s) will be added. This was added in mPDF 2.3 to allow a **ToC** to be placed on the first page, or to allow a **ToC** to follow another **ToC**. In this case, any properties for the continuing document are ignored. If you define several **ToCs** following immediately on from one another, set the properties in the first **ToC** you define.

Parameters

The initial parameters specify characteristics for the **ToC**, which is generated automatically at the end of the document when [Output\(\)](#) is called.

paging = **TRUE|1|FALSE|0**

Specify whether to show page numbers in the **ToC**.
BLANK or omitted uses a default value of **TRUE**.
DEFAULT: **TRUE**

Values

TRUE or 1: show page numbers in the **ToC**.
FALSE or 0: do not show page numbers in the **ToC**.

links = **TRUE|1|FALSE|0**

Specify whether to generate hyperlinks in the **ToC**.
BLANK or omitted uses a default value of **FALSE**.
DEFAULT: **FALSE**

Values

TRUE or 1: show hyperlinks in the **ToC**.

FALSE or 0: do not show hyperlinks in the **ToC**.

toc-orientation

This attribute specifies the orientation of the **ToC** pages.

BLANK or omitted leaves the orientation unchanged i.e. at the end of the document (before the **ToC** is generated)

Values (case-insensitive)

L or landscape: Landscape

P or portrait: Portrait

toc-margin-left
toc-margin-right
toc-margin-top
toc-margin-bottom
toc-margin-header
toc-margin-footer

Set the page margins for the **ToC**.

All values should be specified as **LENGTH** in millimetres.

If you are writing a **DOUBLE-SIDED** document, the margin values will be used for **ODD** pages; left and right margins will be mirrored for **EVEN** pages.

BLANK or omitted leaves the current margin unchanged i.e. the margins current at the end of the document.

"0" (zero) will set the margin to zero.

outdent

Set a negative indent for the last line of each **ToC** entry.

Values should be **BLANK** string or any valid CSS **LENGTH**.

This will cause the line to extend beyond the right margin; you can prevent this by setting **PADDING-RIGHT** equal to this value.

DEFAULT 0

toc-odd-header-name
toc-even-header-name
toc-odd-footer-name
toc-even-footer-name

Selects a header or footer by name to use for the **ToC**. The header/footer must already have been defined using [DefHeaderByName\(\)](#), [DefFooterByName\(\)](#), [DefHTMLHeaderByName\(\)](#), or [DefHTMLFooterByName\(\)](#).

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted leaves the header/footer unchanged. NB **BLANK** will not unset the header. Set *toc-odd-header-value* to -1 to turn the header off.

Note: You must add the prefix 'html_' before the name if it is a **HTMLHeader**.

toc-odd-header-value
toc-even-header-value
toc-odd-footer-value
toc-even-footer-value

Specify whether to show a header or footer in the **ToC**. The header/footer must already have been defined using [DefHeaderByName\(\)](#), [DefFooterByName\(\)](#), [DefHTMLHeaderByName\(\)](#), or [DefHTMLFooterByName\(\)](#).

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted or 0 leaves the header/footer state unchanged.

Values (case-insensitive)

- 1 or on: Show the selected header/footer in the **ToC**.
- 1 or off: Hide the selected header/footer in the **ToC**.

toc-preHTML

Specify the HTML code to appear before the **ToC** e.g. '<h1>Contents</h1>'. Note that in contrast with the HTML equivalent <[tocpagebreak](#)> the text does not need to use HTML-entities.

BLANK or omitted will enter no text

toc-postHTML

Specify the HTML code to appear after the **ToC** e.g. '<p>Comments to go below the ToC</p>'. Note that in contrast with the HTML equivalent <[tocpagebreak](#)> the text does not need to use HTML-entities.

BLANK or omitted will enter no text.

toc-bookmarkText

Specify the text as it will appear as a **BOOKMARK** for the **ToC** e.g. 'Content list'.

BLANK or omitted will not create a **BOOKMARK**.

name

Specify which **ToC** to include at this point, if using more than one **ToC** in the document. *name* can be any alphanumeric characters (except just "0") and is case-insensitive.

BLANK or omitted or 0 uses the default **ToC**.

toc-pageselector

Select a named CSS @page for the **ToC**.

BLANK or omitted or leaves the CSS page unchanged.

See [Using @page](#) for more information

toc-sheet-size

sheet-size can be specified either as a pre-defined page size, or as an array of width and height in millimetres e.g. array(210,297).

DEFAULT: BLANK - makes no change to the current sheet-size

Values (case-insensitive)

- A0 - A10, B0 - B10, C0 - C10
- 4A0, 2A0, RA0 - RA4, SRA0 - SRA4
- Letter, Legal, Executive, Folio
- Demy, Royal
- A (Type A paperback 111x178mm)
- B (Type B paperback 128x198mm)

All of the above values can be suffixed with "-L" to force a Landscape page orientation document e.g. "A4-L"

The rest of the parameters are defined exactly as for [AddPageByArray\(\)](#). Note that these parameters define page numbering, margins, headers/footers for the document as it continues from this point on;

in the final document this will be the part of the document immediately after the **ToC**.
Please refer to [AddPageByArray\(\)](#) for further details.

Changelog

Version	Description
5.0	Function was added.
5.7	<i>outdent</i> parameter added <i>font</i> , <i>font-size</i> and <i>indent</i> redundant

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->WriteHTML('Introduction');
$mpdf->TOCpagebreakByArray();

$mpdf->TOC_Entry("Chapter 1",0);
$mpdf->WriteHTML('Chapter 1 ...');
$mpdf=Output();

?>
```

Example #2 - Blank template as example

```
<?php

$mpdf=new mPDF();
$mpdf->WriteHTML('Introduction');
$mpdf->TOCpagebreakByArray(array(
    'tocfont' => '',
    'tocfontsize' => '',
    'tocindent' => '',
    'TOCusePaging' => true,
    'TOCuseLinking' => '',
    'toc_orientation' => '',
    'toc_mgl' => '',
    'toc_mgr' => '',
    'toc_mgt' => '',
    'toc_mgb' => '',
    'toc_mgh' => '',
    'toc_mgf' => '',
    'toc_ohname' => '',
    'toc_ehname' => '',
    'toc_ofname' => '',
    'toc_efname' => '',
    'toc_ohvalue' => 0,
    'toc_ehvalue' => 0,
    'toc_ofvalue' => 0,
    'toc_efvalue' => 0,
    'toc_preHTML' => '',
    'toc_postHTML' => '',
    'toc_bookmarkText' => '',
    'resetpagenum' => '',
    'pagenumstyle' => '',
    'suppress' => '',
    'orientation' => '',
    'mgl' => '',
    'mgr' => '',
    'mgt' => '',
    'mgb' => '',
    'mgh' => '',
    'mgf' => '',
    'ohname' => ''))
```

```
'ehname' => '',
'ofname' => '',
'efname' => '',
'ohvalue' => 0,
'ehvalue' => 0,
'ofvalue' => 0,
'efvalue' => 0,
'toc_id' => 0,
'pagesel' => '',
'toc_pagesel' => '',
'sheetsize' => '',
'toc_sheetsize' => '',
));
$mpdf->TOC_Entry("Chapter 1",0);
$mpdf->WriteHTML('Chapter 1 ...');
$mpdf=Output();
?>
```

See Also

- [TOC_Entry\(\)](#) - Add an entry for Table of Contents
- [AddPage\(\)](#) - Add a new page
- [AddPageByArray\(\)](#) - Add a new page using an array of parameters
- <[tocpagebreak](#)> - Custom HTML tag - equivalent to **TOCpagebreakByArray()**

TOC_Entry()

(mPDF >= 1.0)

TOC_Entry - Insert an entry for the Table of Contents

Description

```
void TOC_Entry ( string $content [, int $level [, string $name ]])
```

Insert an entry for the Table of Contents referencing the current writing position in the document.

Note: The position for the Table of Contents must be specified using [TOCpagebreak\(\)](#) or [<tocpagebreak>](#) at some point before [Output\(\)](#) is called.

Note: From mPDF 2.3 you can use more than one **ToC** in the document using the parameter *name*.

Parameters

content

This parameter sets the text as it will appear in the ToC entry.

content cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <annotation content="This is < 40" />

It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.

REQUIRED

level

Specify the level of this entry i.e. like heading1,2,3

Starts at level 0

DEFAULT: 0

name

Specify which **ToC** to add this entry, if using more than one **ToC** in the document. *name* can be any alphanumeric characters (except just "0") and is case-insensitive.

BLANK or omitted or 0 uses the default **ToC**.

Values (case-insensitive)

"ALL" will add this entry to every ToC active in the document.

Changelog

Version	Description
---------	-------------

2.3	<i>name</i> attribute was added.
-----	----------------------------------

Examples

Example #1

<pre><?php \$mpdf=new mPDF(); \$mpdf->WriteHTML('Introduction'); \$mpdf->TOCpagebreak();</pre>

```
$mpdf->TOC_Entry("Chapter 1",0);
$mpdf->WriteHTML('Chapter 1 ...');
$mpdf=Output();
?>
```

Notes

Note: Since mPDF 2.0 **insertTOC()** should **not** be called at the end of the document. Output() will automatically generate the ToC if it has been defined with either or **TOCpagebreak()** or **<tocpagebreak>**.

See Also

- **TOCpagebreak()** - Insert a Table of Contents in the document
- **<tocpagebreak>** - Insert a Table of Contents in the document
- **<tocentry>** - Mark a ToC entry in the document

UseTemplate()

(mPDF >= 2.3)

UseTemplate - Insert an imported page from an external PDF file

Description

```
array UseTemplate ( int $templateID [, float $x [, float $y [, float $width [, float $height ]]]])
```

Insert an imported page/template from an external PDF file into the current document. The page, or 'cropped' page, must have already been stored as a 'template' using [SetSourceFile\(\)](#). The template is inserted on the current page of the document. UseTemplate() returns an array of height and width of the imported page as it is printed (see Example #1).

Note: The template will be printed onto the page as the bottom 'layer' i.e. anything else written to that page by mPDF will be written on top of the template. NB If you use WriteHTML() and have a background-color set on BODY this will hide the template from view e.g. <body style="background-color:#FFFFFF;">

Note: If you are using automatic header-margins, you need to set the header before starting the first page; if you start the document with UseTemplate() this will move it to page 1, so the order needs to be:

```
$mpdf = new mPDF();
$mpdf->SetImportUse();
$mpdf->SetHTMLHeader($header);
$pagecount = $mpdf->SetSourceFile('logoheader.pdf');
$tplIdx = $mpdf->ImportPage($pagecount);
$mpdf->UseTemplate($tplIdx);
$mpdf->WriteHTML($html);
```

Parameters

templateID

This parameter specifies the ID of the page template to insert.

x

Sets the x co-ordinate (abscissa) to output the template. Value should be specified as **LENGTH** in millimetres.

DEFAULT NULL - Sets x co-ordinate to 0

BLANK or omitted - uses default (0)

-1: Uses current writing position in document

y

Sets the y co-ordinate (ordinate) to output the template. Value should be specified as **LENGTH** in millimetres.

DEFAULT NULL - Sets y co-ordinate to 0

BLANK or omitted - uses default (0)

-1: Uses current writing position in document

width

Specifies the width for the template to appear on the page. Value should be specified as **LENGTH** in millimetres.

DEFAULT or 0 will output the template at the original size (if neither *width* nor *height* are set) or if *height* is set, the *width* is automatically set to ouput the template in proportion to the original.

height

Specifies the height for the template to appear on the page. Value should be specified as **LENGTH** in millimetres.

DEFAULT or 0 will output the template at the original size (if neither *width* nor *height* are set) or if *width* is set, the *height* is automatically set to ouput the template in proportion to the original.

Return Value

UseTemplate() returns an array of the calculated *width* and *height*.

Changelog

Version Description

2.3 Function was added.

Examples

Example #1

```
<?php

include("../mpdf.php");

$mpdf=new mPDF();
$mpdf->SetImportUse();

// Add First page
$mpdf->AddPage();

$pagecount = $mpdf->SetSourceFile('logoheader.pdf');
$tplId = $mpdf->ImportPage($pagecount);

$actualsize = $mpdf->UseTemplate($tplId);

// The height of the template as it was printed is returned as $actualsize['h']
// The width of the template as it was printed is returned as $actualsize['w']

$mpdf->WriteHTML('Hallo World');

$mpdf->Output();

?>
```

Example #2 - Using a 'cropped' page

```
<?php

include("../mpdf.php");

$mpdf=new mPDF();
$mpdf->SetImportUse();

$pagecount = $mpdf->SetSourceFile('testfile.pdf');

$tplId = $mpdf->ImportPage($pagecount, 50, 50, 100, 100);

$mpdf->UseTemplate($tplId, '', '', 100, 100);

$mpdf->Output();
```

```
?>
```

See Also

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [ImportPage\(\)](#) - Import a page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template
- [RestartDocTemplate\(\)](#) - Re-start the use of a Document template from the next page

WriteBarcode()

(mPDF >= 2.0)

mPDF - Write an EAN-13 (ISBN-13) barcode

Description

```
void writeBarcode ( string $code [, int $showisbn [, float $x [, float $y [, float $size [, float $border [, float $padding_left , float $padding_right , float $padding_top , float $padding_bottom ]]]]]])
```

Write an EAN-13 barcode. Useful information about the EAN-13 (ISBN-13) specification can be found at http://www.gs1uk.org/downloads/bar_code/Bar coding getting it right.pdf

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

code

This parameter specifies the EAN-13 (ISBN-13) code. Accepts 12 or 13 digits (i.e. with or without the check digit) and may optionally contain hyphens e.g. 978-09542-2461-5 or 978095422461

showisbn

Specif whether to show the EAN-13 (ISBN-13) code **above** the barcode. NB The numbers will always appear below the bars, but the code abive the bars is optional.

Values

1 (or any positive value): show the EAN-13 code

0 zero: Hide the EAN-13 code

BLANK or omitted: 1

x

Sets the x (horizontal) position for the barcode.

BLANK or omitted uses the current writing position in the document.

y

Sets the y (vertical) position for the barcode.

BLANK or omitted uses the current writing position in the document.

size

This parameter specifies the size of the barcode relative to the standard. Values between 0.8 and 2.0 (80% to 200%) are accepted.

DEFAULT: 1

border

This parameter specifies whether or not to show a border around the barcode.

Values

1 or **TRUE** (or any positive value) will set a border

0 or FALSE or BLANK will omit the border

DEFAULT: "0" i.e. No border

padding_left
padding_right
padding_top
padding_bottom

Sets the padding around the barcode.

All values should be specified as **LENGTH** in millimetres

BLANK or omitted uses the default values.

DEFAULT Values

padding_left 1
padding_right 1
padding_top 2
padding_bottom 2

Changelog

Version Description

2.0 The function was added.

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->writeBarcode('978-1234-567-890');
$mpdf->Output();

?>
```

WriteCell()

```
WriteCell(float w, float h, string text[, mixed border[, integer ln[, string align[, integer fill[, mixed link[, float returnx]]]]]])
```

Writes a single line of text directly to the PDF document at the current position.

See the details for Cell() at FPDF. An additional parameter returnx has been added; if In is set, the current position moves not to the left margin, but to the value set as returnx.

The methods Cell() and Text() from FPDF are still present, but should not be used directly as they will not cope with UTF-8 encoded text. Use the WriteCell() and WriteText() methods instead.

This method accepts UTF-8 encoded text, and will reverse RTL (right-to-left) text when appropriate.

Text containing HTML entities, as well as decimal and hex e.g. ' ≬ or ↤ can be used, by setting:

\$mpdf->text_input_as_HTML = true; (default = false)

This will convert all the above to their appropriate characters, otherwise the text will be output as it is.

For further information, you are referred to the originals at [FPDF >> Manual](#).

WriteFixedPosHTML()

(mPDF >= 4.0)

WriteFixedPosHTML — Write HTML to a fixed position on the current page

Description

```
void WriteFixedPosHTML ( string $html , float $x , float $y , float $w , float $h [, string $overflow ] )
```

Write HTML to a fixed position on the current page.

Parameters

html

This parameter specifies the text to write to the document - parsed as HTML code

x

Sets the x position of the (left edge) of the block element, set in mm from the left of the page.

y

Sets the y position of the (top edge) of the block element, set in mm from the top of the page.

w

Sets the width of the block element, in mm.

h

Sets the height of the block element, in mm.

overflow

Specifies how to handle text which would not fit inside the block element, with its dimensions as specified.

Values

visible: show all text, even if it spills over outside the dimensions of the block element

hidden: hide any text (clip) which spills over outside the dimensions of the block element

auto: force text to be reduced in size if required to fit inside the dimensions of the block element

BLANK uses the default value of 'visible'

Changelog

Version Description

4.0	Function was added.
-----	---------------------

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
  
$mpdf->WriteHTML('<p>Beginning bit of document...</p>');  
  
$mpdf->WriteFixedPosHTML('<p>This text will appear just where I want it!</p>', 30, 120, 50, 90,  
'auto');  
  
$mpdf=Output();  
  
?>
```

See Also

- [WriteHTML](#) - Add an Index entry in the document
- [Fixed position elements](#) - About fixed-position elements (using CSS)

DRAFT

WriteHTML()

(mPDF >= 1.0)

WriteHTML — Write HTML code to the document

Description

```
void WriteHTML ( string $html [, int $mode [, boolean $initialise [, boolean $close ]]] )
```

Write *html* code to the document.

Note: Prior to mPDF 4.2 a fatal error was caused if *html* was passed as a **NULL** value, **FALSE** or an undefined variable.

Parameters

html

UTF-8 encoded HTML code to write to the document.

mode

Controls what parts of the *html* code is parsed.

Values

- 0 - Parses a whole *html* document
- 1 - Parses the *html* as styles and stylesheets only
- 2 - Parses the *html* as output elements only
- 3 - (For internal use only - parses the *html* code without writing to document)
- 4 - (For internal use only - writes the *html* code to a buffer)

DEFAULT: 0

Mode #0 (DEFAULT)

Metadata:

- title is read from <title>...</title> tags
- subject, keywords and author are read from <meta ...

Charset:

- if *\$allow_charset_conversion* = **TRUE** and a charset= statement is present, mPDF will attempt to convert all the following HTML input from the specified charset to UTF-8

CSS styles:

- any CSS found between <style>...</style> tags
- stylesheets specified by @import url(*.css)
- stylesheets specified by <link rel="stylesheet" href=""

NB Stylesheets with media="all" or media ="screen" will always be parsed.

The variable *\$disablePrintCSS* will determine whether stylesheets media="print" are parsed or not.

Anything between <style> tags is then discarded.

If <body> tags are found, all *html* outside these tags are discarded, and the rest is parsed as content for the document.

If no <body> tags are found, all remaining *html* is parsed as content.

Mode #1

The *html* input is only parsed as CSS style information only.

The code does not have to be surrounded by <style> tags, so you can pass the contents of a stylesheet directly - see Example #1.

Mode #2

If <body> tags are found, all *html* outside these tags are discarded, and the rest is parsed as content for the document.

If no <body> tags are found, all *html* is parsed as content.

Prior to mPDF 4.2 the default CSS was not parsed when using mode #2

initialise

Set **TRUE** or **FALSE** to determine whether to initialise all buffers, starting all HTML elements from new.

See example 2 for use. You must start with a WriteHTML() that calls *initialise=TRUE*

DEFAULT: TRUE

close

Set **TRUE** or **FALSE** to specify whether all HTML elements are closed, and buffers cleared. See example 2 for use. You must end with a WriteHTML() that calls *close=TRUE*

DEFAULT: TRUE

Changelog

Version Description

2.0	Using WriteHTML without the <i>mode</i> parameter no longer clears any CSS styles already imported.
2.1	Parameters <i>initialise</i> and <i>close</i> introduced.
4.2	Accepts NULL string as parameter without error. Parses default CSS when using <i>mode</i> as 2

Examples

Example #1

```
<?php
$mpdf=new mPDF();

$stylesheet = file_get_contents('style.css');
$mpdf->WriteHTML($stylesheet,1);
$mpdf->WriteHTML('<p>Hallo World</p>', 2);

$mpdf->Output();
?>
```

Example #2

```
// You can write parts of HTML elements by using the initialise and close parameters:

$mpdf->WriteHTML('<p>This is the beginning...', 2, true, false);
$mpdf->WriteHTML('...this is the middle...', 2, false, false);
$mpdf->WriteHTML('...and this is the end</p>', 2, false, true);
```

See Also

- [allow_charset_conversion](#) - attempts to read any charset declaration in the HTML code
- [disablePrintCSS](#) - prevents stylesheets set for print media being parsed
- [ignore_invalid_utf8](#) - prevents mPDF from failing if text contains invalid UTF-8 characters
- [charset_in](#) - specify the input text character set if not UTF-8

- **biDirectional** - specify whether mPDF should test for **RTL** text
- **allow_html_optional_endtags** -specify whether mPDF should try to accommodate optional HTML endtags
- **restoreBlockPagebreaks** - keep current HTML tags/CSS styles active when forcing a page-break or formfeed

WriteText()

WriteText(float w, float h, string text)

Writes a single line of text directly to the PDF document at a specified position.
See the details for Text() at FPDF.

The methods Cell() and Text() from FPDF are still present, but should not be used directly as they will not cope with UTF-8 encoded text. Use the WriteCell() and WriteText() methods instead.

This method accepts UTF-8 encoded text, and will reverse RTL (right-to-left) text when appropriate.

Text containing HTML entities, as well as decimal and hex e.g. ' & #8812; or & #x21a4; can be used, by setting:

`$mpdf->text_input_as_HTML = true;` (default = false)

This will convert all the above to their appropriate characters, otherwise the text will be output as it is.

For further information, you are referred to the originals at [FPDF >> Manual](#).

FPDF Original Functions

mPDF is based on [FPDF](#) and has developed to support HTML input. However you can still use the original functions to write directly to the document. The links below take you straight to the FPDF Reference Manual pages:

[Cell](#) - print a cell*
[Image](#) - output an image
[Line](#) - draw a line
[MultiCell](#) - print text with line breaks*
[Rect](#) - draw a rectangle
[SetDrawColor](#) - set drawing color
[SetFillColor](#) - set filling color
[SetFont](#) - set font
[SetFont](#) - set font size
[SetLineWidth](#) - set line width
[SetTextColor](#) - set text color
[SetX](#) - set current x position
[SetXY](#) - set current x and y positions
[SetY](#) - set current y position
[Text](#) - print a string
[Write](#) - print flowing text*

*Note that these functions require text to be encoded if required, and do not automatically reverse RTL text, or convert HTML entities to characters. See the mPDF functions [WriteCell\(\)](#) and [WriteText\(\)](#) which do much the same thing, but support UTF-8 input regardless of the output encoding of your PDF file.

HTML CONTROL TAGS

Overview

mPDF Custom HTML tags by Category

Paging

pagebreak -

formfeed - Add a new page keeping current HTML tags/CSS styles active

Annotation

annotation -

Graphs

jgraph -

Circular Text

textcircle -

Bookmarks

bookmark -

Columns

columnbreak -

columns -

Index

indexentry -

indexinsert -

Table of Contents

tocentry -

tocpagebreak -

Headers & Footers

htmlpagefooter -

htmlpageheader -

pagefooter -

pageheader -

sethtmlpagefooter -

sethtmlpageheader -

setpageheader -

setpagefooter -

Barcode

barcode -

Other

dottab - Insert dots to following (right-aligned) text

annotation

(mPDF >= 2.2)

annotation – Add an Annotation to the document

Description

```
<annotation content [ pos-x ] [ pos-y ] [ icon ] [ author ] [ subject ] [ opacity ] [ color ] [ popup ] [ file ] />
```

Adds an Annotation to the document. An annotation is like a Tooltip on a webpage. The Annotation marker, like those of "Sticky Notes" in Adobe Reader. When the reader passes the cursor over, it will display a popup text box.

The exact position on the page can be specified using *x* and *y*, or left to position automatically. If *x* and *y* are not specified, the Annotation will be inserted at the current position of writing in the document. The *x* position (horizontal) can be overridden by the variable [annotMargin](#), which can be used to force the Annotation marker to display in the right margin.

The attribute *file* can be used to specify a file (any type) which is to be embedded inside the PDF document.

Note: All text to do with an annotation (text, author, subject) is rendered with the system font and can therefore contain any Unicode character even if the document font restricts to a specific codepage.

Annotations cannot be moved or deleted by the reader.

Parameters

content

This parameter specifies the text to appear in the popup text box.

content cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <annotation content="This is < 40" />

It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.

pos-x

Sets the *x* position of the (bottom left edge of the) Annotation marker, set in mm from the left of the page.

BLANK or omitted or 0 uses the current writing position on the page, unless overridden by [\\$annotMargin](#).

pos-y

Sets the *y* position of the (bottom left edge of the) Annotation marker, set in mm from the top of the page. When Annotation markers are used within the text ([annotMargin=FALSE](#)), the marker is raised by the current lineheight to appear above the text.

BLANK or omitted or 0 uses the current writing position on the page.

icon

Sets the appearance of the Annotation marker.

BLANK or omitted uses **DEFAULT** i.e. 'Note'

Values (case sensitive)

Note

Comment

Help

Insert
Key
NewParagraph
Paragraph
DEFAULT: Note

Note: The default is "Comment" when using Annotations from HTML markup when title2annots is **TRUE**

author

This specifies the name of the Author which will appear at the top of the popup text box.
DEFAULT: BLANK

subject

This specifies the text to appear in the Annotation properties.
DEFAULT: BLANK

opacity

Sets the opacity of the Annotation marker. Values must be greater than 0 and <= 1.
BLANK or omitted or 0: sets the opacity to the value of **annotOpacity** (**DEFAULT** 0.5), unless **annotMargin** forces the Annotations to appear in the margin, when the **DEFAULT** is 1

color

Specify the colour of the Annotation marker background.

Values

Any valid CSS color recognised by mPDF:

#RRGGBB

#RGB

rgb(255,255,255)

Named colours such as white, yellow, blue etc.

DEFAULT Yellow

popup

Specify whether to show the popup box for the annotation when the PDF document is opened, and optional specify its dimensions and/or position.
BLANK or omitted or "0" - the popup box is not shown.
Any other value forces the popup box to appear when the document is opened.

Values

<nn nn> - A string of 2 numbers (separated by spaces) will set the X and Y position in mm e.g. '30 30' will show a popup box with the top left corner 30mm from the top of the page and 30mm from the left of the page.

<nn nn nn nn> - A string of 4 numbers (separated by spaces) will set the X and Y position and also the width and height in mm e.g. '30 30 80 50' will show a popup box with the top left corner 30mm from the top of the page and 30mm from the left of the page, a width of 80mm and a height of 50mm.

Note that the PDF Reader (e.g. Adobe Reader) may reposition the popup box as it pleases.

file

file specifies the name of a file to embed in the PDF document.

DEFAULT: BLANK

Changelog

Version	Description
2.2	The function was added.
2.4	Annotations cannot be moved or deleted. (function SetUserRights removed)
4.3	Attribute <i>popup</i> was added
5.1	Attribute <i>file</i> was added

Examples

Example #1

```
<?php

$mpdf=new mPDF();

$html = '<p>This is a paragraph about violas<annotation content="Violas are like big violins" />
about which I know very little.</p>';

$mpdf->WriteHTML($html);
$mpdf->Output();

?>
```

Example #2

```
$mpdf=new mPDF();

// The Annotation markers will appear 10mm in from the right margin of the page
$mpdf->annotMargin = 10;

// The Annotation markers need no transparency as they appear in the margins
$mpdf->annotOpacity = 1;

$html = '<p>This is a paragraph about violas<annotation content="Violas are like big violins" />
about which I know very little.</p>';

$mpdf->Output();
```

See Also

- [annotMargin](#) - Specify the x (horizontal) placement of Annotation markers
- [annotOpacity](#) - Specify the default opacity used for Annotation markers
- [Annotation\(\)](#) - PHP equivalent to <annotation>
- [title2annots](#) - Convert all HTML element *title* attributes to Annotations

barcode

(mPDF >= 4.0)

barcode - Add a Barcode to the document

Description

```
<barcode code [ type ] [ text ] [ size ] [ height ] [ pr ] />
```

Add a Barcode to the document.

Note: See [barcodes](#) and the example file for further information.

Attributes

code

Specifies the code to translate to a barcode.

code for EAN13 / ISBN / ISSN / UPCA / UPCE can contain hyphens '-' but no other characters are allowed.

Check-digits can be optionally included for EAN13 / ISBN / ISSN / UPCA / UPCE

REQUIRED

type

type specifies the type of barcode required.

DEFAULT: EAN13

Values

EAN13, ISBN, ISSN, UPCA, UPCE, EAN8

EAN13P2, ISBNP2, ISSNP2, UPCAP2, UPCEP2, EAN8P2 (with EAN-2 supplement code i.e. 01-99)

EAN13P5, ISBNP5, ISSNP5, UPCAP5, UPCEP5, EAN8P5 (with EAN-5 supplement code e.g. 90000)

(UPCE needs the UPCA code entered)

IMB, RM4SCC, KIX, POSTNET, PLANET

C128A, C128B, C128C

EAN128A, EAN128B, EAN128C

C39, C39+, C39E, C39E+

S25, S25+, I25, I25+, I25B, I25B+

C93

MSI, MSI+

CODABAR

CODE11

Note: Type with a + at the end includes check-digits.

text

EAN13 only

Specifies whether to show the the code at the top of an EAN13 barcode.

Note that ISBN and ISSN always show the text, prefixed with ISBN or ISSN.

Values: 1 or 0

DEFAULT: 0

size

Specifies the size of the barcode.

size (*float*) will scale the nominal size of the barcode as a factor of 1

`size="1.5"` will generate a barcode one and half times the height and width of the nominal size set in mPDF

NB Sizes between 0.8 and 2.0 are recommended for EAN13 and similar barcodes.

DEFAULT: 1

height

Specifies the height of the barcode.

`height (float)` will determine the relative height of the barcode as a factor of 1

The height factor is applied after the `size`

`size="2" height="0.5"` will generate a barcode of twice the nominal width, but with the nominal height.

NB Ignored for Postcode barcodes

DEFAULT: 1

pr

Specifies the print ratio i.e. narrow:wide bar width for some types of barcode.

Valid for: C39 (Code 39), Standard and Interleaved 2 of 5 (S25, I25 etc.), CODABAR and CODE11

DEFAULT: Varies between 2.5 and 3.0 dependent on barcode specification (see [barcodes](#))

Changelog

Version Description

4.0 The function was added.

Examples

Examples

```
<barcode code="978-0-9542246-0" type="ISBN" height="0.66" text="1" />
```

```
<barcode code="04210000526" type="UPCE" />
// Note the UPC-A code is required which is converted to UPCE
```

```
<barcode code="978-0-9542246-0-8 07" type="ISSNP2" text="1" />
```

```
<barcode code="01234567094987654321-01234567891" type="IMB" />
```

```
<barcode code="SN34RD1A" type="RM4SCC" />
```

```
<barcode code="54321068" type="I25" />
```

```
<barcode code="A34698735B" type="CODABAR" />
```

Notes

Note: The following CSS properties can be set on the `<barcode>` element as though it were standard HTML:

`vertical-align` (default: middle)

`border*`

`margin*`

`padding*` (default 2mm for EAN13)

`color` (default black)

`background-color` (default white)

NB padding has defaults as 0(mm) if not specified

Padding is in addition to any specified quiet zones/light margins.

Example with CSS

```
<style>
.barcode {
    padding: 1.5mm;
    margin: 0;
    vertical-align: top;
    color: #000044;
}
.barcodecell {
    text-align: center;
    vertical-align: middle;
}
</style>

<div class="barcodecell"><barcode code="54321068" type="I25" class="barcode" /></div>
```

Note: mPDF will generate a Checkdigit for most barcodes if required, which is added to the barcode. If you need to know what the checkdigit is for a particular barcode, you could do the following:

Example - Generating a checkdigit

```
// Must not contain any - or spaces
include('../classes/barcode.php');
$bc = new PDFBarcode();
echo $bc->getChecksum('9344543204454', 'C93');
exit;
```

See Also

- [Barcodes](#) - More information on types of barcode

bookmark

(mPDF >= 1.0)

bookmark – Add a Bookmark to the document

Description

```
<bookmark content [ level ] />
```

Add a Bookmark to the document. Bookmarks appear in Adobe Reader and link to specific points in the text. The target is set as the current writing position in the document when the Bookmark is defined.

Note: Bookmarks use Adobe Reader system fonts, therefore any Unicode text can be used, even if a unibyte codepage is being used for the document.

Attributes

content

Specifies the text to appear as a Bookmark.

content cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <bookmark content="< 40" />

It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.

level

level specifies the "tree" level for the Bookmark. The top level is 0. See Example 2 below. Accepts an integer from 0 to the maximum depth you wish.

DEFAULT: 0

Examples

Example #1

```
<html>
<bookmark content="Start of the Document" />
<div>Section 1 text</div>
</html>
```

Example #2

```
<html>

<bookmark content="Section 1" />
<div>Section 1 text</div>

<bookmark content="Chapter 1" />
<div>Chapter 1 text</div>

<bookmark content="Chapter 2" />
<div>Chapter 2 text</div>

<bookmark content="Section 2" />
<div>Section 2 text</div>

<bookmark content="Chapter 3" />
<div>Chapter 3 text</div>

</html>
```

```
This will produce a Bookmark tree in Adobe Reader:  
+ Section 1  
  + Chapter 1  
  + Chapter 2  
+ Section 2  
  + Chapter 3
```

Notes

Note: To set the Bookmark for a Table of Contents, see *toc-bookmarkText* in <tocpagebreak>.

Recommended placement

Recommended placement of Bookmarks is just after the first word following the opening tag of the block element:

```
<h2>First<bookmark... /> word of a heading or block</h2>
```

or alternatively just after the opening tag of the block element:

```
<h2><bookmark... />Heading or block</h2>
```

or just after a word to be marked:

```
... this is a word<bookmark... /> in the middle of text to be marked ...
```

Automatically Generated Bookmarks

You can automatically generate bookmarks from h1 - h6 tags, by setting the variable *\$h2bookmarks*.

Define arrays with e.g. the tag => Bookmark level

Remember bookmark levels start at 0.

H1 - H6 must be written with uppercase when defining the array.

Example:

```
$mpdf->h2bookmarks = array('H1'=>0, 'H2'=>1, 'H3'=>2);
```

See Also

- [Bookmark\(\)](#) - PHP equivalent to <bookmark>

columnbreak

(mPDF >= 1.0)

columnbreak - Start a new Column

Description

```
<columnbreak />
```

Start a new Column in the document. Columns must be set using [SetColumns\(\)](#) or [<columns>](#). Height justification for the Columns is disabled when column breaks are set explicitly.

Note: Columns are incompatible with (and automatically disable): borders for block-level elements (DIV, P etc), table rotation, and collapsible margins for blocks e.g. top and bottom margins for a DIV will not collapse (default) at the top/bottom of a column.

Attributes

No attributes

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetColumns(2);  
$mpdf->WriteHTML('<p>Some text...</p><columnbreak /><p>Next column...</p>');  
$mpdf=Output();  
  
?>
```

See Also

- [AddColumn\(\)](#) - PHP equivalent to <columnbreak>
- [SetColumns\(\)](#) - Control the use of multiple columns on the page
- [<columns>](#) - Control the use of multiple columns on the page

columns

(mPDF >= 1.0)

columns - Control use of Columns on the page

Description

```
<columns column-count [ vAlign ] [ column-gap ] />
```

Define, start or stop Columns in the document.

Note: The maximum ratio to adjust column height when justifying is set by `$max_colH_correction` - too large a value can give ugly results

Note: If you are setting HTMLHeaders or HTMLFooters, this will cancel any columns you have set; you need to call `SetColumns()` or use `<columns>` after commands like `SetHTMLHeader()` etc.

Parameters

column-count

Set the number of (vertical) columns to use on a page

BLANK or omitted or 0 or 1 turns Columns OFF i.e. the whole page is used as one column.

DEFAULT: 1

vAlign

Automatically adjusts height of columns to be equal if set to J or justify.

BLANK or omitted turns vertical-alignment OFF

Values (case-insensitive)

J or justify

DEFAULT: ""

column-gap

Set the gap between columns in millimeters

BLANK or omitted uses default value.

DEFAULT: 5 (mm)

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->WriteHTML('<columns column-count="3" vAlign="J" column-gap="7" />');
$mpdf->WriteHTML('<p>Some text...</p>');
$mpdf->WriteHTML('<columnbreak />');
$mpdf->WriteHTML('<p>Next column...</p>');
$mpdf=Output();

?>
```

See Also

- [AddColumn\(\)](#) - Start a new Column
- [<columnbreak>](#) - Start a new Column
- [SetColumns\(\)](#) - Control the use of multiple columns on the page - PHP equivalent of <columns>

dottab

(mPDF >= 4.3)

dottab – Insert dots to following (right-aligned) text

Description

```
<dottab [ outdent ] />
```

Insert a string of dots to fill the space to the text which follows `<dottab />`, which is right-aligned. This gives the appearance seen in e.g. a table of contents, or a menu with prices. A minimum of 3 dots (with spaces either side) is inserted. If the text which follows the `<dottab />` will not fit on a single line, the default tab of ' ... ' is added, and text is not right-aligned.

Note: If a space is required after the `<dottab>` use a non-breaking space

From mPDF 5.7+ CSS styles can be applied to `<dottab>` as for an inline element.

Attributes

outdent

Takes any valid CSS **LENGTH** e.g. "2em"

DEFAULT: 0

Changelog

Version Description

4.3	This tag was added.
5.7	<i>outdent</i> was added

Examples

Example #1

```
$menuitem = '<p>Chilli con carne <dottab />&nbsp;£7.95</p>';  
$mpdf->WriteHTML($menuitem);
```

Example #2

To right-align the text which follows the `<dottab>` whilst indenting any preceding lines:

```
$menuitem = '<p style="padding-right: 3em">Chilli con carne etc. etc. <dottab outdent="3em" />&nbsp;£7.95</p>';
```

formfeed

(mPDF >= 2.3)

formfeed — Add a new page keeping current HTML tags/CSS styles active

Description

```
<formfeed [ orientation ] [ type ] [ resetpagenum ] [ pagenumstyle ] [ suppress ]
[ margin-left ] [ margin-right ] [ margin-top ] [ margin-bottom ] [ margin-header ] [ margin-footer ]
[ odd-header-name ] [ odd-header-value ] [ even-header-name ] [ even-header-value ] [ odd-footer-name ] [
odd-footer-value ] [ even-footer-name ] [ even-footer-value ] [ page-selector ] />
```

Add a new page to the document. The attributes are the same as for <pagebreak>, but whereas pagebreak by default closes any open HTML tags, and does not continue CSS styles after the pagebreak, formfeed reinstates the active properties and styles on the new page.

The attribute **type** can specify certain conditions which determine how many pages are added. If writing a **DOUBLE-SIDED** document, a conditional formfeed (**type="E"** or **"O"**) will add a new page only if required to make the current page match the type (i.e. **ODD** or **EVEN**); a formfeed with **type="NEXT-ODD"** or **"NEXT-EVEN"** will add one or two pages as required to make the current page match the type (i.e. **ODD** or **EVEN**).

Number of pages added:

DOUBLE-SIDED			
type	SINGLE-SIDED	Currently ODD page	Currently EVEN page
BLANK	1	1	1
O or ODD	0	0	1
E or EVEN	0	1	0
NEXT-ODD	1	2	1
NEXT-EVEN	1	1	2

Note: If no new page is added, the other parameters will be ignored e.g. resetting page numbers/styles, margins and headers/footers. If 2 pages are added, any changes in page numbers/styles, margins and headers/footers will start on the final added page.

Note: From mPDF >= 3.0 the page numbering can be reset to any positive number. Prior to this, it was only possible to reset it to 1.

Attributes

orientation = L|P|landscape|portrait

This attribute specifies the orientation of the new page.
BLANK or omitted leaves the current orientation unchanged

Values (case-insensitive)
L or landscape: Landscape
P or portrait: Portrait

type = E|O|even|odd|next-odd|next-even

If *type* is specified when writing a **DOUBLE-SIDED** document, the formfeed is conditional; a new page will only be added if necessary to meet the specified condition.
If not writing a **DOUBLE-SIDED** document, a formfeed *type="E"* will be ignored, whilst a formfeed *type="O"* will always force a new page.

BLANK or omitted will force a new page unconditionally.

Values (case-insensitive)

O or odd: Add a new page if required to make current page an **ODD** one.

E or even: Add a new page if required to make current page an **EVEN** one.

NEXT-ODD: Add one or two pages as required to make the current page **ODD**.

NEXT-EVEN: Add one or two pages as required to make the current page **EVEN**.

resetpagenum = 1 - ∞

Sets/resets the document page number to *resetpagenum* starting on the new page. (The value must be a positive integer).

BLANK or omitted or 0 leaves the current page number sequence unchanged.

pagenumstyle = 1|A|a|i

Sets/resets the page numbering style (values as for lists)

BLANK or omitted leaves the current page number style unchanged.

Values (case-sensitive)

1: Decimal - 1,2,3,4...

A: Alpha uppercase - A,B,C,D...

a: Alpha lowercase - a,b,c,d...

I: Roman uppercase - I, II, III, IV...

i: Roman lowercase - i, ii, iii, iv...

suppress = on|off|1|0

suppress=on will suppress document page numbers from the new page onwards (until *suppress=off* is used)

BLANK or omitted leaves the current condition unchanged.

Values (case-insensitive)

1 or on: Suppress (hide) page numbers from the new page forwards.

0 or off: Show page numbers from the new page forwards.

margin-left
margin-right
margin-top
margin-bottom
margin-header
margin-footer

Sets the page margins from the new page forwards.

All values should be specified as **LENGTH** in any valid CSS form.

If you are writing a **DOUBLE-SIDED** document, the margin values will be used for **ODD** pages; left and right margins will be mirrored for **EVEN** pages.

BLANK or omitted leaves the current margin unchanged. NB "0" (zero) will set the margin to zero.

odd-header-name
even-header-name
odd-footer-name
even-footer-name

Selects a header or footer by name to use from the new page forwards. The header/footer must already have been defined using [<pageheader>](#), [<pagefooter>](#), [<htmlpageheader>](#), or [<htmlpagefooter>](#).

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for

EVEN will be ignored.

BLANK or omitted leaves the header/footer unchanged. NB "" will not unset the header. Use *odd-header-value* to turn the header off.

Note: You must add the prefix 'html_' before the name if it is a HTMLHeader.

odd-header-value

even-header-value

odd-footer-value

even-footer-value

Specify whether to show or hide the named header or footer from the new page forwards. The header/footer must already have been defined using [<pageheader>](#), [<pagefooter>](#), [<htmlpageheader>](#), or [<htmlpagefooter>](#).

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted or 0 leaves the header/footer state unchanged.

Values (case-insensitive)

1 or on: Start using the selected header/footer from the new page onwards.

-1 or off: Start the selected header from the new page onwards.

pageselector

Select a named CSS @page.

BLANK or omitted or leaves the CSS page unchanged.

See [Using @page](#) for more information

Changelog

Version	Description
2.3	Function was added.
3.0	<code>resetpagenum</code> changed to allow positive integers above 1
4.2	Parameter <code>page-selector</code> was added

Examples

Example #1

```
<html>
<div style="border:1px solid blue; font-size: 14pt">Text of introduction...
<formfeed />
This text will start on a new page, and will also have a blue border etc...</div>
</html>
```

Notes

Note: See [<pagebreak>](#) for further examples using the attributes. There is no PHP equivalent of [<formfeed>](#) but you can use `$restoreBlockPagebreaks`

See Also

- [AddPage\(\)](#) - Add one or more (conditional) pages to the document

- <pagebreak> - Forces a new page
- [restoreBlockPagebreaks](#) - forces <pagebreak> to act in the same way as <formfeed>

htmlpagefooter

(mPDF >= 2.0)

htmlpagefooter - Define an HTML page footer with a given name

Description

```
<htmlpagefooter name > html </htmlpagefooter>
```

Define an HTML page footer with a given name. Named footer can be referenced and set later in the document e.g. <sethtmlpagefooter>

Note: Do not name any header or footer starting with `html_`. This prefix is reserved to identify an **HTML** header/footer when passing its name in a reference.

Note: Remember that, unlike most mPDF custom tags which are self-closing with `/>`, `htmlpageheader` and `htmlpagefooter` require end tags. If you wish to make the HTML code compatible with browsers, see [Custom tags](#)

Attributes

`name`

This attribute is a text string to use as the name for this footer.
If name is **BLANK** or omitted, it is set as '`_default`'.

Content

`html`

Any valid HTML code can be enclosed between the tags, and will be parsed by mPDF as for any other content.

Changelog

Version	Description
---------	-------------

2.0	The tag was added.
-----	--------------------

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 4](#)

See Also

- [DefHTMLFooterByName\(\)](#)
- [<htmlpageheader>](#)
- [SetHTMLFooterByName\(\)](#)
- [<sethtmlpagefooter>](#)
- [@page](#)

htmlpageheader

(mPDF >= 2.0)

htmlpageheader – Define an HTML page header with a given name

Description

```
<htmlpageheader name > html </htmlpageheader>
```

Define an HTML page header with a given name. Named headers can be referenced and set later in the document e.g. [<sethtmlpageheader>](#)

Note: Do not name any header or footer starting with `html_`. This prefix is reserved to identify an **HTML** header/footer when passing its name in a reference.

Note: Remember that, unlike most mPDF custom tags which are self-closing with `/>`, `htmlpageheader` and `htmlpagefooter` require end tags. If you wish to make the HTML code compatible with browsers, see [Custom tags](#)

Attributes

`name`

This attribute is a text string to use as the name for this header.
If name is **BLANK** or omitted, it is set as '`_default`'.

Content

`html`

Any valid HTML code can be enclosed between the tags, and will be parsed by mPDF as for any other content.

Changelog

Version	Description
---------	-------------

2.0	The tag was added.
-----	--------------------

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 4](#)

See Also

- [DefHTMLHeaderByName\(\)](#)
- [<htmlpagefooter>](#)
- [SetHTMLHeaderByName\(\)](#)
- [<sethtmlpageheader>](#)
- [@page](#)

indexentry

(mPDF >= 1.0)

indexentry - Insert an Index entry for the document

Description

```
<indexentry content [ xref ] />
```

Insert an Index entry for the document Index, referencing the current writing position in the document. If *xref* is set, it will appear as a cross-referencing entry in the index as for [IndexEntrySee\(\)](#).

Note: The Index must be generated explicitly at the end of the document using [CreateIndex\(\)](#) at some point before [Output\(\)](#) is called.

Attributes

content

This attribute sets the text as it will appear in the Index entry. Text should be UTF-8 encoded.
content cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <indexentry content="< 40" />
It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.

REQUIRED

xref

This attribute sets the text used as a cross-reference. Text should be UTF-8 encoded.
xref cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <indexentry xref="< 40" />
It is recommended that you use e.g. `htmlspecialchars($xref, ENT_QUOTES)` for this.
Text entries passed in the form "Subject:Subcategory" will appear in the Index as "Subject, Subcategory"
OPTIONAL

Changelog

Version Description

Version	Description
3.0	<i>xref</i> attribute added.

Examples

Example #1

```
<?php

$mpdf=new mPDF();

$mpdf->WriteHTML('<p>Beginning bit of document...</p>');
$mpdf->WriteHTML('<p>Your text which refers to a buffalo,<indexentry content="Buffalo" /> which you would like to see in the Index</p>');

$mpdf->AddPage();
$mpdf->WriteHTML('<h2>Index</h2>',2);
$mpdf->CreateIndex(2, '', '', 3, 1, '', 5, 'serif','sans-serif');

$mpdf=Output();

?>
```

Example #2

```
$mpdf->WriteHTML('<p><indexentry content="Dromedary" xref="Camel:types" />The dromedary is a type of camel</p>');

// This will produce an entry in the Index under 'Dromedary' appearing as:

Dromedary - see Camel, types
```

Notes

Note: <indexentry> may be a preferred form to [IndexEntry\(\)](#), as it will allow more precise identification of the position and page - the <indexentry> can be placed just next to the appropriate word.

Recommended placement

Recommended placement of Index Entries is just after the first word following the opening tag of the block element:

```
<h2>First<indexentry... /> word of a heading or block</h2>
```

or alternatively just after the opening tag of the block element:

```
<h2><indexentry... />Heading or block</h2>
```

or just after a word to be marked:

```
... this is a word<indexentry... /> in the middle of text to be marked ...
```

See Also

- [IndexEntry\(\)](#) - Mark an Index entry in the document
- [CreateIndex\(\)](#) - Generate a document Index

indexinsert

(mPDF >= 3.0)

indexinsert — Generate an Index for the document

Description

```
<indexinsert [ font ] [ font-size ] [ line-spacing ] [ cols ] [ gap ] [ offset ] [ links ] [ usedivletters ] [ div-font ] [ div-font-size ] />
```

From mPDF v 6.0 onwards, the attributes have changed. See [Indexes](#) for more information. The following is for mPDF >= v6.0:

```
<indexinsert [ links ] [ usedivletters ] [ collation ] [ collation-group ] />
```

Inserts an Index for the document based on index entries made using [<indexentry>](#) or [CreateIndex\(\)](#).

Attributes

font

Set the font-family for the Index.

BLANK or omitted uses default font-family for the document.

font-size

Sets the font size for the Index in **points** (pt)

BLANK or omitted or 0 uses the default font-size for the document.

line-spacing

Sets the line-height used for index entries. Usual values between 1.0 and 1.4.

BLANK or omitted or 0 uses the default value of the document.

DEFAULT: 1.2

cols

Set the number of (vertical) columns to use for the Index

BLANK or omitted or 0 or 1 uses the whole page is used as one column.

DEFAULT: 1

gap

Sets the gap between columns (if set) in millimeters.

BLANK or omitted uses the default value.

DEFAULT: 5 (mm)

offset

Sets the text indent (in mm) for subsequent lines, if an index entry flows onto two or more lines.

BLANK or omitted uses a default value of 3mm.

links = 1|on|0|off

Specify whether to generate hyperlinks to the pages in the Index.

BLANK or omitted uses a default value of OFF.

DEFAULT: OFF

Values (case-insensitive)

ON or 1: show hyperlinks in the Index

OFF or 0: do not show hyperlinks in the Index

usedivletters = 1|on|0|off|-1

Specify whether to insert a CAPITAL letter to divide each group of entries starting with the same letter in the Index.

BLANK or omitted uses a default value of ON.

DEFAULT: ON

Values (case-insensitive)

ON or 1 or Omitted: show dividing letters in the Index.

OFF or 0 or -1: do not show dividing letters in the Index.

div-font

Set the font-family for the dividing letters in the Index.

NB Will always appear in **BOLD** style.

BLANK or omitted uses default font-family for the document.

div-font-size

Sets the font size for the dividing letters in the Index in **points** (pt)

BLANK or omitted or 0 uses the default font-size for the document.

indexCollationLocale

Set a Locale to determine the overall sort order of index entries e.g. 'en_GB.utf8'. Available options are determined by the locales available in your system configuration. Always use a utf-8 locale.

BLANK or omitted uses current locale set in your system.

indexCollationGroup

If you have set your index to use Dividing letters, this value will determine how letters are grouped under a dividing letter. Values should be selected from the files in folder /collations/ e.g. 'English_United_Kingdom'

NB This will not affect the overall order of entries, which is determined by the value above.

BLANK or omitted - grouping occurs under the first letter of the index entries.

Changelog

Version	Description
3.0	Tag was added.
6.0	Parameters removed: <i>font</i> <i>font-size</i> <i>line-spacing</i> <i>cols</i> <i>gap</i> <i>offset</i> <i>links</i> <i>usedivletters</i> <i>div-font</i> <i>div-font-size</i> Parameters added: <i>collation</i> <i>collation-group</i>

Examples

Example #1 (mPDF >= 6.0)

```
<html>
<p>Text of document...</p>
<p><indexentry content="Buffalo" />Your text which refers to a buffalo, which you would like to see
in the Index</p>
```

```
<p>...rest of document</p>
<pagebreak />
<h2>Index</h2>
<indexinsert usedivletters="on" links="on" collation="en_US.utf8" collation-
group="English_United_States"/>
</html>
```

See Also

- [Indexes](#)

jpgraph

(mPDF >= 2.4)

jpgraph — Generate a graph from table data (requires [JPGraph](#) integration)

Description

```
<jpgraph [ table ][ type ][ stacked ][ dpi ][ title ][ splines ][ bandw ][ antialias ][ label-y ][ label-x ][ axis-x ][ axis-y ][ percent ][ series ][ data-col-begin ][ data-row-begin ][ data-col-end ][ data-row-end ][ show-values ][ width ][ height ][ legend-overlap ][ hide-grid ][ hide-y-axis ] />
```

Generates and inserts a graph into the document at the current writing position. <jpgraph> must follow the table which it refers to (not necessarily immediately). Requires *useGraphs* set to **TRUE**.

Note: This requires [JPGraph](#) to be installed on the server. See [Graphs](#) for further information.

Attributes

table

This attribute (optionally) specifies the table "id" or "name" from which to use data.

BLANK or omitted - uses data from the most recent table (in order of the HTML code being parsed) as long as the table did not have an "id" or "name" defined.

type

Specifies the type of graph.

BLANK or omitted uses the default value.

Values (case-insensitive)

- bar
- horiz_bar (horizontal bar graph)
- line
- radar
- pie
- pie3d
- xy
- scatter

DEFAULT: bar

Graphs of type xy or scatter will expect exactly two columns/rows of numerical data - giving X and Y co-ordinates respectively. In the xy graph, the x values need to be in numerical order.

stacked = 1|0

Specifies whether to "stack" bars in graphs of type *bar* or *horizontal-bar*.

BLANK or omitted uses default value.

DEFAULT: 0 (OFF)

dpi

Sets the image resolution of the graph in dots per inch (dpi). NB Large values will use extensive amounts of memory.

BLANK or omitted uses default value.

Values**INTEGER:** between 50 - 2400**DEFAULT:** 150*title*

Specifies a text string to use atitle for the graph

splines = 1|0

Specify whether to smooth lines for xy-type line graphs

DEFAULT: 0*bandw* = 1|0

Specify whether to create a black and white graph

DEFAULT: 0 (colour)*antialias* = 1|0

Specify whether to use antialias in generating the graphs.

If antialias is used better quality curves are produced, but graph lines will only be 1px wide - which will be very thin when using higher resolutions e.g. 300dpi (this is a limitation set by JGraph)

DEFAULT: 1 (use antialias) for all types except *line* and *radar*.*label-y*

Specifies a text string to use a label across the y-axis

label-x

Specifies a text string to use a label across the x-axis

axis-x

Specify the scale or type of x-axis.

Values (case-insensitive)

text: uses text labels for the x-axis

lin: use a linear scale

log: use a logarithmic scale

DEFAULT: text (except if splines are set when it will default to 'lin')*axis-y*

Specify the scale or type of y-axis.

Values (case-insensitive)

lin: use a linear scale for the y-axis

percent: show a percent sign on a linear scale

log: use a logarithmic scale

DEFAULT: lin*percent* = 1|0

Specify whether to graph the data as percentages of the series total. This useful if you have 2 series of data to compare such as the number of cycle accidents per age group compared with the population broken down by age group.

DEFAULT: 0

series

Specify whether the table data has the data series in columns or rows.

Values (case-sensitive)

cols: data series are read from table columns

rows: data series are read from table rows

DEFAULT:cols

data-col-begin

Specify the column number to start reading data

Values

INTEGER:

DEFAULT: 2 (except *scatter* and *xy* and *series='cols'*, when = 1)

data-row-begin

Specify the row number to start reading data

Values

INTEGER:

DEFAULT: 2 (except *scatter* and *xy* and *series='rows'*, when = 1)

data-col-end

Specify the last column number to contain data.

Values

0: Read data up to, and including, the last column

POSITIVE INTEGER: Specify the last column by number to include data

NEGATIVE INTEGER: Specify the column reading from the last column e.g. "-2" = 2nd column from last

DEFAULT: 0

data-row-end

Specify the last row number to contain data.

Values

0: Read data up to, and including, the last row

POSITIVE INTEGER: Specify the last row by number to include data

NEGATIVE INTEGER: Specify the row reading from the last row e.g. "-2" = 2nd row from last

DEFAULT: 0

show-values = 1|0

Specify whether to show the value for each data point

DEFAULT: 0

width
height

Specify width and/or height fro the graph. If only one is specified, the graph is resized in proportion to the default sizings.

Values

Any valid CSS value including 100%, 300px etc. If no units are defined, pixels are assumed.

DEFAULT: Values are set according to graph type (in graph.php)

```
$defsize['pie'] = array('w' => 600, 'h' => 300);
$defsize['pie3d'] = array('w' => 600, 'h' => 300);
$defsize['radar'] = array('w' => 600, 'h' => 300);
$defsize['line'] = array('w' => 600, 'h' => 400);
$defsize['xy'] = array('w' => 600, 'h' => 400);
$defsize['scatter'] = array('w' => 600, 'h' => 400);
$defsize['bar'] = array('w' => 600, 'h' => 400);
$defsize['horiz_bar'] = array('w' => 600, 'h' => 500);
```

legend-overlap = 1|0

Specify whether to overlap the legend box over the graph (ignored for *pie*, *pie3d* and *radar*)

DEFAULT: 0

hide-grid = 1|0

Specify whether to hide the grid lines (ignored for *pie*, *pie3d* and *radar*)

DEFAULT: 0

hide-y-axis = 1|0

Specify whether to hide the whole y-axis - including the grid lines (ignored for *pie*, *pie3d* and *radar*)

DEFAULT: 0

Note: Other attributes or styles supported by can be used, except for *width* and *height* (which are ignored) and of course *src*.

Changelog

Version Description

2.4 The function was added.

Examples

Example #1

```
<?php
include("../mpdf.php");

define("_JPGRAPH_PATH", '../../../../../jpgraph_5/src/'); // must define this before including mpdf.php file
define("_TTF_FONT_NORMAL", 'arial.ttf');
define("_TTF_FONT_BOLD", 'arialbd.ttf');

$mpdf=new mPDF();
$mpdf->useGraphs = true;
```

```
$html = '
<table id="tbl_1"><tbody>
<tr><td></td><td><b>Female</b></td><td><b>Male</b></td></tr>
<tr><td>35 - 44</td><td><b>4</b></td><td><b>2</b></td></tr>
<tr><td>45 - 54</td><td><b>5</b></td><td><b>7</b></td></tr>
<tr><td>55 - 64</td><td><b>21</b></td><td><b>18</b></td></tr>
<tr><td>65 - 74</td><td><b>11</b></td><td><b>14</b></td></tr>
<tr><td>75 - 84</td><td><b>10</b></td><td><b>10</b></td></tr>
<tr><td>85 - 94</td><td><b>2</b></td><td><b>1</b></td></tr>
<tr><td>95 - 104</td><td><b>1</b></td><td><b>1</b></td></tr>
<tr><td>TOTAL</td><td>54</td><td>52</td></tr>
</tbody></table>

<jpgraph table="tbl_1" type="bar" stacked="0" dpi="300" title="New subscriptions" splines="1"
bandw="0" antialias="1" label-y="% patients" label-x="Age group" axis-x="text" axis-y="lin"
percent="0" series="cols" data-col-begin="2" data-row-begin="2" data-col-end="0" data-row-end="-1"
show-values="1" width="600" legend-overlap="1" hide-grid="1" hide-y-axis="1" />
';

$mpdf->WriteHTML($html );
$mpdf->Output();
exit;
?>
```

See Also

- [useGraphs](#) - Parse table data from the HTML, and allow the use of <jpgraph>
- [Graphs](#) - More about JPGraph and graphs

pagebreak

(mPDF >= 1.0)

pagebreak — Add a new page

Description

```
<pagebreak [ orientation ] [ type ] [ resetpagenum ] [ pagenumstyle ] [ suppress ]
[ margin-left ] [ margin-right ] [ margin-top ] [ margin-bottom ] [ margin-header ] [ margin-footer ]
[ odd-header-name ] [ odd-header-value ] [ even-header-name ] [ even-header-value ] [ odd-footer-name ] [
odd-footer-value ] [ even-footer-name ] [ even-footer-value ] [ page-selector ] [ sheet-size ] [ page-break-type ]
/>
```

Add a new page to the document. All properties set by a pagebreak will continue on subsequent pages until reset.

The attribute **type** can specify certain conditions which determine how many pages are added. If writing a **DOUBLE-SIDED** document, a conditional page-break (`type="E"` or `"O"`) will add a new page only if required to make the current page match the type (i.e. **ODD** or **EVEN**); a page-break with `type="NEXT-ODD"` or `"NEXT-EVEN"` will add one or two pages as required to make the current page match the type (i.e. **ODD** or **EVEN**).

Number of pages added:

DOUBLE-SIDED			
type	SINGLE-SIDED	Currently ODD page	Currently EVEN page
BLANK	1	1	1
O or ODD	0	0	1
E or EVEN	0	1	0
NEXT-ODD	1	2	1
NEXT-EVEN	1	1	2

Note: If no new page is added, the other parameters will be ignored e.g. resetting page numbers/styles, margins and headers/footers. If 2 pages are added, any changes in page numbers/styles, margins and headers/footers will start on the final added page.

Note: `<newpage>` and `<page_break>` are synonymous. `<pagebreak>` is the preferred form.

Note: From mPDF >= 3.0 the page numbering can be reset to any positive number. Prior to this, it was only possible to reset it to 1.

Attributes

`orientation` = L|P|landscape|portrait

This attribute specifies the orientation of the new page.
BLANK or omitted leaves the current orientation unchanged

Values (case-insensitive)

L or landscape: Landscape

P or portrait: Portrait

`type` = E|O|even|odd|next-odd|next-even

If `type` is specified when writing a **DOUBLE-SIDED** document, the page-break is conditional; a new page will only be added if necessary to meet the specified condition.

If not writing a **DOUBLE-SIDED** document, a page-break `type="E"` will be ignored, whilst a page-break `type="O"` will always force a new page.
BLANK or omitted will force a new page unconditionally.

Values (case-insensitive)

O or odd: Add a new page if required to make current page an **ODD** one.
E or even: Add a new page if required to make current page an **EVEN** one.
NEXT-ODD: Add one or two pages as required to make the current page **ODD**.
NEXT-EVEN: Add one or two pages as required to make the current page **EVEN**.

`resetpagenum = 1 - ∞`

Sets/resets the document page number to `resetpagenum` starting on the new page. (The value must be a positive integer).

BLANK or omitted or 0 leaves the current page number sequence unchanged.

`pagenumstyle = 1|A|a|I|i|[+ any value supported for list-style-type]`

Sets/resets the page numbering style (values as for cf. [lists](#))

BLANK or omitted leaves the current page number style unchanged.

Values (case-sensitive)

1: Decimal - 1,2,3,4...
A: Alpha uppercase - A,B,C,D...
a: Alpha lowercase - a,b,c,d...
I: Roman uppercase - I, II, III, IV...
i: Roman lowercase - i, ii, iii, iv...

`suppress = on|off|1|0`

`suppress=on` will suppress document page numbers from the new page onwards (until `suppress=off` is used)

BLANK or omitted leaves the current condition unchanged.

Values (case-insensitive)

1 or on: Suppress (hide) page numbers from the new page forwards.
0 or off: Show page numbers from the new page forwards.

`margin-left`
`margin-right`
`margin-top`
`margin-bottom`
`margin-header`
`margin-footer`

Sets the page margins from the new page forwards.

All values should be specified as **LENGTH** in any valid CSS form.

If you are writing a **DOUBLE-SIDED** document, the margin values will be used for **ODD** pages; left and right margins will be mirrored for **EVEN** pages.

BLANK or omitted leaves the current margin unchanged. NB "0" (zero) will set the margin to zero.

`odd-header-name`
`even-header-name`
`odd-footer-name`
`even-footer-name`

Selects a header or footer by name to use from the new page forwards. The header/footer must already have been defined using `<pageheader>`, `<pagefooter>`, `<htmlpageheader>`, or `<htmlpagefooter>`. If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored. **BLANK** or omitted leaves the header/footer unchanged. NB "" will not unset the header. Use `odd-header-value` to turn the header off.

Note: You must add the prefix 'html_' before the name if it is a HTMLHeader.

odd-header-value
even-header-value
odd-footer-value
even-footer-value

Specify whether to show or hide the named header or footer from the new page forwards. The header/footer must already have been defined using `<pageheader>`, `<pagefooter>`, `<htmlpageheader>`, or `<htmlpagefooter>`. If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored. **BLANK** or omitted or 0 leaves the header/footer state unchanged.

Values (case-insensitive)

1 or on: Start using the selected header/footer from the new page onwards.
-1 or off: Stop using the selected header from the new page onwards.

page-selector

Select a named CSS @page.
BLANK or omitted or leaves the CSS page unchanged.

See [Using @page](#) for more information

sheet-size

sheet-size can be specified either as a pre-defined page size, or as two **LENGTH** values separated by a space, representing width and height e.g. '210mm 297mm'. em, ex and % are not accepted. Note that this is different from the 'size' property of the page-box used with the CSS @page selector.

DEFAULT: BLANK - makes no change to the current sheet-size

Values (case-insensitive)

A0 - A10, B0 - B10, C0 - C10
4A0, 2A0, RA0 - RA4, SRA0 - SRA4
Letter, Legal, Executive, Folio
Demy, Royal
A (Type A paperback 111x178mm)
B (Type B paperback 128x198mm)
`<LENGTH>{2}`

All of the pre-defined values can be suffixed with "-L" to force a Landscape page orientation document e.g. "A4-L"

page-break-type = slice|clone|clonebycss

slice - no border and no padding are inserted at a break. The effect is as though the element were rendered with no breaks present, and then sliced by the breaks afterward
cloneall - each page fragment is independently wrapped with the borders and padding of all open elements

clonebycss - open elements which have the (custom) CSS property "box-decoration-break" set to "clone" are independently wrapped with their border and padding
BLANK or omitted - default page break type is used - as specified by `defaultPagebreakType`

Changelog

Version	Description
1.3	Values NEXT-ODD and NEXT-EVEN for <code>type</code> were added. Parameters <code>resetpagenum</code> , <code>pagenumstyle</code> and <code>suppress</code> were added.
2.0	Parameters <code>margin-left</code> , <code>margin-right</code> , <code>margin-top</code> , <code>margin-bottom</code> , <code>margin-header</code> , <code>margin-footer</code> , <code>odd-header-name</code> , <code>odd-header-value</code> , <code>even-header-name</code> , <code>even-header-value</code> , <code>odd-footer-name</code> , <code>odd-footer-value</code> , <code>even-footer-name</code> , <code>even-footer-value</code> were added.
3.0	<code>resetpagenum</code> changed to allow positive integers above 1
4.2	Parameter <code>page-selector</code> was added
4.3	Parameter <code>sheet-size</code> was added
6.0	Parameter <code>page-break-type</code> was added

Examples

Example #1

```
<html>
<p>Text of introduction...</p>
<pagebreak type="NEXT-ODD" resetpagenum="1" pagenumstyle="i" suppress="off" />
<p>Text of main book...</p>
</html>
```

Example #2 - Defining new margins and page orientation

```
<html>

<p>Text of introduction...</p>
<pagebreak orientation="landscape" margin-left="60mm" margin-right="40mm" margin-top="55mm" margin-bottom="30mm" margin-header="12mm" margin-footer="12mm" />
<p>Text of main book...</p>

</html>
```

Example #3 - Changing headers/footers

```
<html>

<pageheader name="myHeader1" content-center="My document" header-style="font-weight: bold;" line="on" />

<htmlpageheader name="myHeader2">
<div style="text-align: center; font-weight: bold;">My document</div>
</htmlpageheader>

<p>Text of introduction...</p>

<pagebreak type="NEXT-ODD" odd-header-name="myHeader1" odd-header-value="1" even-header-name="html_myHeader2" even-header-value="1" odd-footer-value="-1" even-footer-value="-1" />

<p>Text of main book...</p>
</html>
```

Notes

Note: `pagebreak` can be used as an **HTML** equivalent of `AddPage()`.

See Also

- [AddPage\(\)](#) - Add one or more (conditional) pages to the document
- [Page breaks](#)
- [`<formfeed>`](#) - Forces a new page keeping current HTML tags/CSS active
- [restoreBlockPageBreaks](#) - Configure pagebreak to act like formfeed

pageheader

(mPDF >= 2.0)

pageheader – Define a page header with a given name

Description

```
<pageheader [ name ] [ content-left ] [ content-center ] [ content-right ] [ header-style ] [ header-style-left ]
[ header-style-center ] [ header-style-right ] [ line ] />
```

Define a page header with a given name. Named headers can be referenced and set later in the document e.g.
`<setpageheader>`

Note: Do not name any header or footer starting with `html_` This prefix is reserved to identify an HTML header/footer when passing its name in a reference.

Attributes

name

This attribute is a text string to use as the name for this header.
If name is **BLANK** or omitted, it is set as '`_default`'.

content-left
content-center
content_right

Defines the text to appear in the page header.
At least one 'content-' must be defined.

Values

content-left: Text to appear at left margin
content-center: Text to appear in centre of page
content-right: Text to appear at right margin

header-style
header-style-left
header-style-center
header-style-right

This attribute can optionally set CSS style properties for the page header.
header-style will set the same style for left, right and center content, whereas *header-style-left*, *header-style-center* and *header-style-right* set the style for one part of the content only.

Values

Valid CSS inline style declaration but only 5 properties can be set:

font-family
font-size
font-weight
font-style
color

If style is not set, the default values for the document are used.

line

If set to "1" or any positive value, a line will be drawn below the header.

Changelog

Version	Description
2.0	The tag was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 3](#)

See Also

- [DefHeaderByName\(\)](#)
- [<pagefooter>](#)
- [SetHeaderByName\(\)](#)
- [<setpageheader>](#)

pagefooter

(mPDF >= 2.0)

pagefooter - Define a page footer with a given name

Description

```
<pagefooter [ name ] [ content-left ] [ content-center ] [ content-right ] [ footer-style ] [ footer-style-left ] [ footer-style-center ] [ footer-style-right ] [ line ] />
```

Define a page footer with a given name. Named footers can be referenced and set later in the document e.g.
<setpagefooter>

Note: Do not name any header or footer starting with `html_` This prefix is reserved to identify an **HTML** header/footer when passing its name in a reference.

Attributes

name

This attribute is a text string to use as the name for this footer.
If name is **BLANK** or omitted, it is set as '`_default`'.

content-left
content-center
content_right

Defines the text to appear in the page footer.
At least one 'content-' must be defined.

Values

content-left: Text to appear at left margin
content-center: Text to appear in centre of page
content-right: Text to appear at right margin

footer-style
footer-style-left
footer-style-center
footer-style-right

This attribute can optionally set CSS style properties for the page footer.
footer-style will set the same style for left, right and center content, whereas *footer-style-left*, *footer-style-center* and *footer-style-right* set the style for one part of the content only.

Values

Valid CSS inline style declaration but only 5 properties can be set:

font-family
font-size
font-weight
font-style
color

If style is not set, the default values for the document are used.

line

If set to "1" or any positive value, a line will be drawn above the footer.

Changelog

Version	Description
2.0	The tag was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 3](#)

See Also

- [DefFooterByName\(\)](#)
- [<pageheader>](#)
- [SetFooterByName\(\)](#)
- [<setpagefooter>](#)

sethtmlpagefooter

(mPDF >= 2.0)

sethtmlpagefooter – Set an HTML page footer by a given name

Description

```
< sethtmlpagefooter [ name ] [ page ] [ value ] />
```

Sets an HTML page footer that has previously been defined by name.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

This parameter specifies the name of a previously defined HTML page footer. If a **BLANK** string or **NULL** is passed, mPDF will use the value '_default' if such a page footer exists.
The *name* does not need to be defined if you are setting the value to -1 or 'off'

page

Specify whether to set the footer for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'ODD'

Note: setting this value to **BLANK** will not clear the footer; set *value* to -1 or off to cancel the header

Values (case-insensitive)

O or ODD - set the footer for **ODD** pages in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

E or EVEN - set the footer for **EVEN** pages

ALL - sets the footer for both **ODD** and **EVEN** pages.

If the *page* value is **BLANK** or omitted - sets the footer for **ODD** in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

value

Specify whether to start or stop showing the named footer from the current position in the document.

Values (case-insensitive)

1 or ON - start using this named footer

-1 or OFF - stop using a footer

DEFAULT - **BLANK** is ignored, making no changes to the current state

Changelog

Version Description

2.0	The function was added.
-----	-------------------------

Examples

For examples and further information please see:

- [Headers & Footers](#)

- Headers & Footers - Method 4

See Also

- [DefHTMLFooterByName\(\)](#)
- [`<htmlpagefooter>`](#)
- [SetHTMLFooterByName\(\)](#)
- [`<sethtmlpageheader>`](#)

sethtmlpageheader

(mPDF >= 2.0)

sethtmlpageheader - Set an HTML page header by a given name

Description

```
< sethtmlpageheader [ name ] [ page ] [ value ] [ show-this-page ] />
```

Sets an HTML page header that has previously been defined by name.

Parameters

name

This parameter specifies the name of a previously defined HTML page header. If a **BLANK** string or **NULL** is passed, mPDF will use the value '`_default`' if such a page header exists.

The *name* does not need to be defined if you are setting the value to -1 or 'off'

page

Specify whether to set the header for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'ODD'

Note: setting this value to **BLANK** will not clear the header; set *value* to -1 or off to cancel the header

Values (case-insensitive)

O or ODD - set the header for **ODD** pages in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

E or EVEN - set the header for **EVEN** pages

ALL - sets the header for both **ODD** and **EVEN** pages.

If the *page* value is **BLANK** or omitted - sets the header for **ODD** in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

value

Specify whether to start or stop showing the named header from the current position in the document.

Values (case-insensitive)

1 or ON - start using this named header

-1 or OFF - stop using a header

DEFAULT - **BLANK** is ignored, making no changes to the current state

show-this-page

If *show-this-page* is set to "1" (or any positive value) it forces the header to be written immediately to the current page. Use if the header is being set after the new page has been added.

DEFAULT: **BLANK**

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 4](#)

See Also

- [DefHTMLHeaderByName\(\)](#)
- [<htmlpageheader>](#)
- [SetHTMLFooterByName\(\)](#)
- [<sethtmlpagefooter>](#)

setpagefooter

(mPDF >= 2.0)

setpagefooter – Set a page footer by a given name

Description

```
< setpagefooter [ name ] [ page ] [ value ] />
```

Sets a page footer that has previously been defined by name.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

This parameter specifies the name of a previously defined page footer. If a **BLANK** string or **NULL** is passed, mPDF will use the value '_default' if such a page footer exists.
The *name* does not need to be defined if you are setting the value to -1 or 'off'

page

Specify whether to set the footer for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'ODD'

Note: setting this value to **BLANK** will not clear the footer; set *value* to -1 or off to cancel the header

Values (case-insensitive)

O or ODD - set the footer for **ODD** pages in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

E or EVEN - set the footer for **EVEN** pages

ALL - sets the footer for both **ODD** and **EVEN** pages.

If the *page* value is **BLANK** or omitted - sets the footer for **ODD** in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

value

Specify whether to start or stop showing the named footer from the current position in the document.

Values (case-insensitive)

1 or ON - start using this named footer

-1 or OFF - stop using a footer

DEFAULT - **BLANK** is ignored, making no changes to the current state

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 3](#)

See Also

- [DefFooterByName\(\)](#)
- [<pagefooter>](#)
- [SetFooterByName\(\)](#)
- [<setpageheader>](#)
- [@page](#)

setpageheader

(mPDF >= 2.0)

setpageheader - Set a page header by a given name

Description

```
< setpageheader [ name ] [ page ] [ value ] [ show-this-page ] />
```

Sets a page header that has previously been defined by name.

Note: This function/method was altered in mPDF 2.2 by capitalising the first letter of the name. As function/method names in PHP have hitherto been case-insensitive, this should not cause any problems, but it is recommended where possible to use the preferred spelling.

Parameters

name

This parameter specifies the name of a previously defined page header. If a **BLANK** string or **NULL** is passed, mPDF will use the value '_default' if such a page header exists.
The *name* does not need to be defined if you are setting the value to -1 or 'off'

page

Specify whether to set the header for **ODD** or **EVEN** pages in a **DOUBLE-SIDED** document.

DEFAULT: 'ODD'

Note: setting this value to **BLANK** will not clear the header; set *value* to -1 or 'off' to cancel the header

Values (case-insensitive)

O or ODD - set the header for **ODD** pages in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

E or EVEN - set the header for **EVEN** pages

ALL - sets the header for both **ODD** and **EVEN** pages.

If the *page* value is **BLANK** or omitted - sets the header for **ODD** in a **DOUBLE-SIDED** document, or for both **ODD** and **EVEN** in a **SINGLE-SIDED** document.

value

Specify whether to start or stop showing the named header from the current position in the document.

Values (case-insensitive)

1 or ON - start using this named header

-1 or OFF - stop using a header

DEFAULT - **BLANK** is ignored, making no changes to the current state

show-this-page

If *show-this-page* is set to "1" (or any positive value) it forces the header to be written immediately to the current page. Use if the header is being set after the new page has been added.

DEFAULT: **BLANK**

Note: *show-this-page* forces the appropriate header to be written. If you have just defined an **ODD**-sided header and the document is currently writing to an **EVEN**-sided page, the **EVEN** header will be output.

Changelog

Version	Description
2.0	The function was added.

Examples

For examples and further information please see:

- [Headers & Footers](#)
- [Headers & Footers - Method 3](#)

See Also

- [DefHeaderByName\(\)](#)
- [<pageheader>](#)
- [SetHeaderByName\(\)](#)
- [<setpagefooter>](#)
- [@page](#)

textcircle

(mPDF >= 5.4)

textcircle - Draw a circle using specified text

Description

```
<textcircle r top-text bottom-text [ divider ][ space-width ][ char-width ][ href ] />
```

Draw a circle using specified text. One or both of top-text and/or bottom-text must be defined. The radius and font-size are user-defined, whilst the width and height of the generated object will be calculated from these values. Font-size should be set for `<textcircle>` using in-line CSS or specified in a CSS stylesheet as for any standard HTML tag. Other CSS styles supported on Circular Text are: border, margin, padding, color, background-color, font-family, font-size, font-weight, font-style, display, visibility, and opacity.

Note: CSS style are not inherited from parent elements.

From mPDF >= 5.6 the CSS property `font-size` can be set to `auto`. This automatically sizes text to fill a semicircle (if both top and bottom set) or a full circle (if only one set).

Circular Text is displayed as though an in-line element. Automatic kerning will be applied to the text if `useKerning` is **TRUE**.

Note: Prior to mPDF 5.6 the textcircle was displayed with a white background. in >= 5.6 the background has been changed to transparent by default.

Parameters

r

Radius of circle. Any valid **LENGTH** can be entered

top-text

This parameter defines text which will be centred on the top of the circle.
top-text cannot contain any of the characters: `<` `>` `&` `'` or `"` and must use the appropriate HTML entities e.g. `<textcircle top-text="Brian's document" />` It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.
 Either top-text or bottom-text (or both) must be defined.

DEFAULT: BLANK

bottom-text

This parameter defines text which will be centred on the bottom of the circle.
bottom-text cannot contain any of the characters: `<` `>` `&` `'` or `"` and must use the appropriate HTML entities e.g. `<textcircle bottom-text="Brian's document" />` It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.
 Either top-text or bottom-text (or both) must be defined.

DEFAULT: BLANK

divider

This parameter defines an optional string of characters which will be used to divide top and bottom text of the circle.
divider cannot contain any of the characters: `<` `>` `&` `'` or `"`. Use HTML entities for special characters or

non-ASCII characters e.g. <textcircle divider="•" />

DEFAULT: BLANK

space-width

This parameter should be specified as an integer defining the (fixed) letter-spacing as a percentage of normal.

DEFAULT: 120

char-width

This parameter should be specified as an integer defining the width of each character as a percentage of normal.

DEFAULT: 100

Changelog

Version	Description
5.4	The tag was added.
5.6	Transparent background fixed. Support for divider added. Support for "font: auto" added.

Examples

Example #1

```
<?php
...
$mpdf->WriteHTML('<textcircle r="30mm" space-width="120" char-width="150"
top-text="• Circular Text •" bottom-text="Circular Text"
style="background-color: #FFAAAA; border:1px solid red; padding: 0.3em; margin: 0.3em; color:
#000000; font-size: 21pt; font-weight:bold; font-family: Arial" />');
...
?>
```

See Also

- [CircularText\(\)](#) - PHP equivalent to <textcircle>

tocentry

(mPDF >= 1.0)

tocentry - Insert an entry for the Table of Contents

Description

```
<tocentry content [ level ] [ name ] />
```

Insert an entry for the Table of Contents referencing the current writing position in the document.

Note: The position for the Table of Contents must be specified using [TOCpagebreak\(\)](#) or [<tocpagebreak>](#) at some point before [Output\(\)](#) is called.

Note: From mPDF 2.3 you can use more than one **ToC** in the document using the attribute *name*.

Parameters

content

This parameter sets the text as it will appear in the ToC entry. Text should be UTF-8 encoded.
content cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <tocentry content="< 40" />
It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.

REQUIRED

level

Specify the level of this entry (i.e. like heading 1,2,3) as a positive integer

Starts at level 0

DEFAULT: 0

name

Specify which **ToC** to add this entry, if using more than one **ToC** in the document. *name* can be any alphanumeric characters (except just "0") and is case-insensitive.

BLANK or omitted or 0 uses the default **ToC**.

Values (case-insensitive)

"ALL" will add this entry to every ToC active in the document.

Changelog

Version Description

2.3 *name* attribute was added.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->WriteHTML('Introduction');
```

```
$mpdf->WriteHTML('<tocpagebreak />');

$mpdf->WriteHTML('<tocentry content="Chapter 1" />');
$mpdf->WriteHTML('Chapter 1 ...');
$mpdf=Output();

?>
```

Notes

Note: Since mPDF 2.0 **insertTOC()** should **not** be called at the end of the document. **Output()** will automatically generate the ToC if it has been defined with either or **TOCpagebreak()** or **<tocpagebreak>**.

Recommended placement

Recommended placement of ToC Entries is just after the first word following the opening tag of the block element:

```
<h2>First<tocentry... /> word of a heading or block</h2>
```

or alternatively just after the opening tag of the block element:

```
<h2><tocentry... />Heading or block</h2>
```

or just after a word to be marked:

```
... this is a word<tocentry... /> in the middle of text to be marked ...
```

See Also

- [TOCpagebreak\(\)](#) - Insert a Table of Contents in the document
- [<tocpagebreak>](#) - Insert a Table of Contents in the document
- [TOC_Entry\(\)](#) - Mark a ToC entry in the document

tocpagebreak

(mPDF >= 2.0)

tocpagebreak — Insert a table of contents

Description

```
<tocpagebreak [ paging ] [ links ] [ toc-orientation ] [ toc-margin-left ] [ toc-margin-right ] [ toc-margin-top ]
[ toc-margin-bottom ] [ toc-margin-header ] [ toc-margin-footer ] [ toc-odd-header-name ] [ toc-even-header-
name ] [ toc-odd-footer-name ] [ toc-even-footer-name ] [ toc-odd-header-value ] [ toc-even-header-value ] [
toc-odd-footer-value ] [ toc-even-footer-value ] [ toc-preHTML ] [ toc-postHTML ] [ toc-bookmarkText ] [ name ] [
toc-page-selector ] [ toc-sheet-size ] [ toc-resetpagenum ] [ toc-pagenumstyle ] [ toc-suppress ]
[ orientation ] [ resetpagenum ] [ pagenumstyle ] [ suppress ]
[ margin-left ] [ margin-right ] [ margin-top ] [ margin-bottom ] [ margin-header ] [ margin-footer ]
[ odd-header-name ] [ odd-header-value ] [ even-header-name ] [ even-header-value ] [ odd-footer-name ] [
odd-footer-value ] [ even-footer-name ] [ even-footer-value ] [ page-selector ] [ sheet-size ] [ outdent ] />
```

Add a new page to the document, marking the point at which a Table of Contents (**ToC**) will be inserted in the document at the end of writing. The numerous parameters specify both paging details for the continuing document, and for the **ToC** when it is generated.

Note: From mPDF 5.7 the layout of a table of contents can be controlled using CSS. *font font-size* and *indent* have become redundant.

Note: When writing a **DOUBLE-SIDED** document, the **ToC** will always start on an **ODD** page. Therefore there is no option to specify the pagebreak type as in <pagebreak> - using <tocpagebreak> will always continue the document on an **ODD** page.

Note: Page numbering was suppressed in the **ToC** prior to mPDF v 6.0. From v6.0 onwards, you can specify the page numbering throughout the **ToC**.

Note: The **ToC** is generated at the end of the document. Unless otherwise specified, the **ToC** will inherit the page margins, headers/footers and orientation of the last page written to the document.

Note: From mPDF 2.3 you can include more than one **ToC** in the document using the attribute *name*.

Note: If <tocpagebreak> occurs at the start of a blank (**ODD**) page, no new page(s) will be added. This was added in mPDF 2.3 to allow a **ToC** to be placed on the first page, or to allow a **ToC** to follow another **ToC**. In this case, any properties for the continuing document are ignored. If you define several **ToCs** following immediately on from one another, set the properties in the first **ToC** you define (including the *resetpagenum*).

Attributes

The first set of attributes specify characteristics for the **ToC**, which is generated automatically at the end of the document when **Output()** is called.

paging = 1|on|0|off

Specify whether to show page numbers in the **ToC**.
BLANK or omitted uses a default value of **TRUE**.
DEFAULT: ON

Values (case-insensitive)
ON or 1: show page numbers in the **ToC**.

OFF or 0: do not show page numbers in the **ToC**.

links = 1|on|0|off

Specify whether to generate hyperlinks in the **ToC**.

BLANK or omitted uses a default value of **FALSE**.

DEFAULT: OFF

Values (case-insensitive)

ON or **1:** show hyperlinks in the **ToC**.

OFF or **0:** do not show hyperlinks in the **ToC**.

toc-orientation

This attribute specifies the orientation of the **ToC** pages.

BLANK or omitted leaves the orientation unchanged i.e. at the end of the document (before the **ToC** is generated)

Values (case-insensitive)

L or landscape: Landscape

P or portrait: Portrait

toc-margin-left
toc-margin-right
toc-margin-top
toc-margin-bottom
toc-margin-header
toc-margin-footer

Set the page margins for the **ToC**.

Values can be specified using any valid CSS **LENGTH** e.g. px, pt, em, mm.

If you are writing a **DOUBLE-SIDED** document, the margin values will be used for **ODD** pages; left and right margins will be mirrored for **EVEN** pages.

BLANK or omitted leaves the current margin unchanged i.e. the margins current at the end of the document.

"0" (zero) will set the margin to zero.

toc-odd-header-name
toc-even-header-name
toc-odd-footer-name
toc-even-footer-name

Selects a header or footer by name to use for the **ToC**. The header/footer must already have been defined using **defHeaderByName()**, **defFooterByName()**, **defHTMLHeaderByName()**, or **defHTMLFooterByName()**.

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted leaves the header/footer unchanged. NB **BLANK** will not unset the header. Set *toc-odd-header-value* to -1 to turn the header off.

Note: You must add the prefix 'html_' before the name if it is a **HTMLHeader**.

toc-odd-header-value
toc-even-header-value
toc-odd-footer-value
toc-even-footer-value

Specify whether to show a header or footer in the **ToC**. The header/footer must already have been defined using **defHeaderByName()**, **defFooterByName()**, **defHTMLHeaderByName()**, or

defHTMLFooterByName()

If you are writing a **SINGLE-SIDED** document, the values for **ODD** will be used for all pages, and values for **EVEN** will be ignored.

BLANK or omitted or 0 leaves the header/footer state unchanged.

Values (case-insensitive)

1 or on: Show the selected header/footer in the **ToC**.

-1 or off: Hide the selected header/footer in the **ToC**.

toc-preHTML

Specify the HTML code to appear before the **ToC**.

The HTML code cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <tocpagebreak toc-prehtml="<h1>Contents</h1>">

It is recommended that you use `htmlspecialchars('Your html code', ENT_QUOTES)` for this.

BLANK or omitted will enter no text

toc-postHTML

Specify the HTML code to appear after the **ToC**.

The HTML code cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <tocpagebreak toc-posthtml="<p>Comments after the Contents table</p>">

It is recommended that you use `htmlspecialchars('Your html code', ENT_QUOTES)` for this.

BLANK or omitted will enter no text.

toc-bookmarkText

Specify the text as it will appear as a **BOOKMARK** for the **ToC** e.g. 'Content list'.

The text cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <tocpagebreak toc-bookmarkText="Contents table >>">

It is recommended that you use `htmlspecialchars('Your bookmark text', ENT_QUOTES)` for this.

BLANK or omitted will not create a **BOOKMARK**.

name

Specify which **ToC** to include at this point, if using more than one **ToC** in the document. *name* can be any alphanumeric characters (except just "0") and is case-insensitive.

BLANK or omitted or 0 uses the default **ToC**.

toc-page-selector

Select a named CSS @page for the **ToC**.

BLANK or omitted or leaves the CSS page unchanged.

See [Using @page](#) for more information

toc-sheet-size

toc-sheet-size can be specified either as a pre-defined page size, or as two **LENGTH** values separated by a space, representing width and height e.g. '210mm 297mm'. em, ex and % are not accepted. Note that this is different from the 'size' property of the page-box used with the CSS @page selector.

DEFAULT: BLANK - makes no change to the current sheet-size

Values (case-insensitive)

A0 - A10, B0 - B10, C0 - C10

4A0, 2A0, RA0 - RA4, SRA0 - SRA4

Letter, Legal, Executive, Folio
 Demy, Royal
 A (Type A paperback 111x178mm)
 B (Type B paperback 128x198mm)
 <LENGTH>{2}

All of the pre-defined values can be suffixed with "-L" to force a Landscape page orientation document e.g. "A4-L"

toc-resetpagenum = 1 - ∞

Sets/resets the document page number to *resetpagenum* starting on the **Toc**. (The value must be a positive integer).

BLANK or omitted or 0 leaves the preceding page number sequence unchanged.

toc-pagenumstyle = 1|A|a||I|i|[+ any value supported for list-style-type]

Sets/resets the page numbering style to use in the **Toc** (values as for cf. [lists](#))

BLANK or omitted leaves the current page number style unchanged.

Values (case-sensitive)

- 1: Decimal - 1,2,3,4...
- A: Alpha uppercase - A,B,C,D...
- a: Alpha lowercase - a,b,c,d...
- I: Roman uppercase - I, II, III, IV...
- i: Roman lowercase - i, ii, iii, iv...

toc-suppress = on|off|1|0

suppress=on will suppress document page numbers in the **Toc**

BLANK or omitted leaves the current condition unchanged.

Values (case-insensitive)

1 or on: Suppress (hide) page numbers in the **Toc**

0 or off: Show page numbers in the **Toc**

The rest of the attributes are defined exactly as for [`<pagebreak>`](#). Note that these attributes define page numbering, margins, headers/footers for the document as it continues from this point on; in the final document this will be the part of the document immediately after the **Toc**.

Please refer to [`<pagebreak>`](#) for further details.

Changelog

Version	Description
2.0	Tag was added.
2.2	Default values for <i>font-size</i> , <i>paging</i> and <i>links</i> were redefined.
2.3	<i>name</i> attribute was added.
3.0	<i>toc-bookmarkText</i> changed to decode <i>htmlspecialchars</i>
4.3	Parameters <i>page-selector</i> , <i>sheet-size</i> , <i>toc-page-selector</i> and <i>toc-sheet-size</i> were added
5.7	<i>outdent</i> parameter added <i>font</i> , <i>font-size</i> and <i>indent</i> redundant
6.0	Parameters added: <i>toc-resetpagenum</i> <i>toc-pagenumstyle</i> <i>toc-suppress</i>

Examples

Example #1

```
<html>
<p>Text of introduction...</p>
<tocpagebreak />
<p><tocentry content="Chapter 1" />Text of main book...</p>
</html>
```

See Also

- **<tocentry>**- Add an entry for Table of Contents
- **<pagebreak>** - Add a new page
- **TOCpagebreak()** - PHP equivalent of **<tocpagebreak>**

watermarkimage

(mPDF >= 3.0)

watermarkimage - Set an image to use as a Watermark

Description

```
<watermarkimage src [ alpha ] [ size ] [ position ] />
```

Set an image to use as a Watermark. The watermark is a semi-transparent background printed on each page, used for text such as "DRAFT" or a background image. The watermark will be added to each page when the Footer is printed if the variable `showWatermarkImage` is set to 1 or **TRUE**.

Parameters

src

This parameter specifies the image file to use for the watermark. This can be a full URI or use a relative path

alpha

This parameter defines the transparency value (alpha) to use for the watermark. The Value should be between 0 and 1.

DEFAULT: 0.2

size

This parameter takes either a pre-defined string, an integer, or an array of width and height. Defines the size of the watermark.

Values

D: default i.e. original size of image - may depend on `img_dpi`

P: Resize to fit the full page size, keeping aspect ratio

F: Resize to fit the print-area (frame) respecting current page margins, keeping aspect ratio

INT: Resize to full page size minus a margin set by this integer in millimeters, keeping aspect ratio

2 comma-separated numbers (`$width, $height`): Specify a size; units in millimeters

DEFAULT: "D"

position

This parameter takes either a pre-defined string or an array of x and y. Defines the position of the watermark on the page.

Values

P: Centred on the whole page area

F: Centred on the page print-area (frame) respecting page margins

2 comma-separated numbers (`$x, $y`): Specify a position; units in millimeters

DEFAULT: "P"

Changelog

Version	Description
3.0	The tag was added.

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->showWatermarkText = true;
$mpdf->WriteHTML('<watermarkimage src="images/background.png" alpha="0.4" size="200,250" />');
$mpdf->WriteHTML('<p>Hallo World</p>');

?>
```

See Also

- [SetWatermarkImage\(\)](#) - PHP equivalent to <watermarkimage>
- [watermarkImageAlpha](#) - Specifies the transparency (alpha value) for the watermark image
- [watermarkTextAlpha](#) - Specifies the transparency (alpha value) for the watermark text
- [showWatermarkText](#) - Specifies whether or not to show/print the watermark text
- [showWatermarkImage](#) - Specifies whether or not to show/print the watermark image
- [watermark_font](#) - Specifies the font to use for Watermark text

watermarktext

(mPDF >= 3.0)

watermarktext - Set the text to use as a Watermark

Description

```
<watermarktext content [ alpha ] />
```

Set the text to use as a Watermark. The watermark is a semi-transparent background printed on each page, used for text such as "DRAFT". The watermark will be added to each page when the Footer is printed if the variable `showWatermark` is set to 1 or **TRUE**.

Parameters

content

This parameter defines the text to use for the watermark.

content cannot contain any of the characters: < > & ' or " and must use the appropriate HTML entities e.g. <watermarktext content="Brian's document" /> It is recommended that you use `htmlspecialchars('Content', ENT_QUOTES)` for this.

If the text is blank, it will clear the watermark text, so nothing appears.

DEFAULT: BLANK

alpha

This parameter defines the transparency value (alpha) to use for the watermark: either text or image. The Value should be between 0 and 1.

DEFAULT: 0.2

Changelog

Version Description

Version	Description
3.0	The tag was added.

Examples

Example #1

```
<?php
$mpdf=new mPDF();
$mpdf->showWatermarkText = true;
$mpdf->WriteHTML('<watermarktext content="DRAFT" alpha="0.4" />');
$mpdf->WriteHTML(' <p>Hallo World</p>');
?>
```

See Also

- [SetWatermarkText\(\)](#) - PHP equivalent to <watermarktext>
- [SetWatermarkImage\(\)](#) - Set an image to use as a Watermark
- [watermarkImageAlpha](#) - Specifies the transparency (alpha value) for the watermark image
- [watermarkTextAlpha](#) - Specifies the transparency (alpha value) for the watermark text
- [showWatermarkText](#) - Specifies whether or not to show/print the watermark text

- [showWatermarkImage](#) - Specifies whether or not to show/print the watermark image
- [watermark_font](#) - Specifies the font to use for Watermark text

MPDF VARIABLES

Overview

mPDF Variables by Category

Most of the following variables are set as defaults in the configuration file config.php. When set there, they are written as e.g.:

```
$this->variablename = true;
```

Alternatively you can redefine the variable in individual scripts; here it will be written as e.g. (assuming your class object is "mpdf"):

```
$mpdf->variablename = true;
```

Category	Variable	Default value	Scope ^[1]	Version	Notes
Paging	mirrorMargins	0	DOCUMENT	4.0	The alias <code>useOddEven</code> was used before v4.0
	restoreBlockPagebreaks	FALSE		2.3	Removed v6.0
	forcePortraitMargins	FALSE		2.3	
	displayDefaultOrientation	FALSE		2.3	
	autoPageBreak	TRUE		3.1	
	setAutoTopMargin	FALSE		4.0	
	setAutoBottomMargin	FALSE		4.0	
	autoMarginPadding	2		4.0	
	margBuffer	2		5.5	Allows an (empty) end of block to extend beyond the bottom margin by this amount (mm)
	printers_info	FALSE		5.1	Adds date and page info for printer when using <code>@page</code> and <code>"marks:crop;"</code>
Page numbering	bleedMargin	5	DOCUMENT	5.1	
	crossMarkMargin	5		5.1	Distance of cross mark from margin in mm
	cropMarkMargin	8		5.1	Distance of crop mark from margin in mm
	cropMarkLength	18		5.1	Default length in mm of crop line
	nonPrintMargin	8		5.1	Non-printable border at edge of paper sheet in mm
	aliasNbPg	"{nb}"			NB The default value will not appear correctly in a PDF version of this page; it is substituted by the number of pages.
	aliasNbPgGp	"{nbpq}"			NB The default value will not appear correctly in a PDF version of this page; it is substituted by the number of pages.
	pagenumPrefix	""		3.0	
	pagenumSuffix	""		3.0	
	nbpqPrefix	""		3.0	
	nbpqSuffix	""		3.0	
	defaultPageNumStyle	"1"		6.0	

Category	Variable	Default value	Scope ^[1]	Version	Notes
Fonts, Languages and Character sets	percentSubset	30	DOCUMENT	5.0	Control subsetting behaviour for fonts
	useKerning	FALSE	DOCUMENT	5.4	Set to TRUE to enable CSS support for <i>font-kerning</i>
	maxTTFFilesize	2000	DOCUMENT	5.0	Control subsetting behaviour for fonts
	allow_charset_conversion	TRUE			
	charset_in	NULL			
	biDirectional	FALSE			
	use_CJK_only	FALSE			Removed mPDF 5.0
	useAdobeCJK	TRUE	DOCUMENT*	5.0	* You must only change this variable in the config.php file. Use initial parameter to change at runtime e.g. \$mpdf=new mPDF(' +aCJK');
	autoFontGroupSize	2		2.3	Removed in v6.0
	useLang	TRUE		2.3	Default value FALSE before v4.0 Removed in v6.0
	autoScriptToLang	FALSE	DOCUMENT	6.0	Replaces SetAutoFont function
	autoLangToFont	FALSE	DOCUMENT	6.0	Replaces useLang
	baseScript	1	DOCUMENT	6.0	
	autoArabic	FALSE	DOCUMENT	6.0	
	autoVietnamese	FALSE	DOCUMENT	6.0	
	disableMultilingualJustify	TRUE		2.3	Removed in v6.0
	falseBoldWeight	5		4.2	Weight for bold text when using an artificial (outline) bold
	smCapsScale	0.75		5.0	Factor of 1 to scale capital letters
	smCapsStretch	115		5.0	% to stretch small caps horizontally
	backupSubsFont	array('dejavusanscondensed')		5.0	Set in config_fonts.php
	backupSIPFont	NOT DEFINED		5.0	Set in config_fonts.php
Configuration	useOnlyCoreFonts	FALSE	DOCUMENT	(3.0)	Removed mPDF 5.0 - Use \$mpdf=new mPDF(' c '); The alias use_embeddedfonts_1252 was used before v4.0
	repackageTTF	FALSE	DOCUMENT	5.2	
	useSubstitutions	FALSE		(4.0)	NB Altered behaviour mPDF >= 5.0 Default value TRUE before v4.0

Category	Variable	Default value	Scope ^[1]	Version	Notes
	useSubstitutionsMB	FALSE		4.2	Removed in mPDF 5.0 Use useSubstitutions instead. Substitute missing characters in UTF-8(multibyte) documents - from core fonts
	collapseBlockMargins	TRUE		4.2	Allows top and bottom margins to collapse between block elements
	dpi	96		4.5	Specifies size conversion for objects with size set by "px"
	enableImports	FALSE	DOCUMENT*	4.3	Enable Imported PDF files (templates) [was mPDFI] * You must only change this variable in the config.php file. Use SetImportUse() to change at runtime.
	allow_output_buffering	FALSE	DOCUMENT	3.0	
	allow_html_optional_endtags	TRUE	DOCUMENT		
	ignore_invalid_utf8	FALSE			
	text_input_as_HTML	FALSE	DOCUMENT		
	showStats	FALSE	DOCUMENT	4.0	
	progressBar	FALSE	DOCUMENT*	4.2	Shows progress-bars whilst generating file * You must only change this variable in the config.php file. Use StartProgressBarOutput() to set at runtime.
	progbar_heading	"mPDF file progress"	DOCUMENT	5.0	Customise the use of progress-bars
	progbar_altHTML	""	DOCUMENT	5.0	Customise the use of progress-bars
	incrementFPR1 [1-4]	10,20,30,50		4.2	
Debugging	debug	FALSE	DOCUMENT	3.1	
	debugfonts	FALSE	DOCUMENT	5.0	Show errors and warning notes for fonts
	showImageErrors	FALSE	DOCUMENT	3.0	
PDF/A1-b, PDF/X-1a Colorspace	PDFA	FALSE	DOCUMENT	4.3	
	PDFAAuto	FALSE	DOCUMENT	4.3	
	PDFX	FALSE	DOCUMENT	5.1	
	PDFXAuto	FALSE	DOCUMENT	5.1	
	ICCProfile	""	DOCUMENT	4.3	
	restrictColorSpace	0	DOCUMENT	5.1	
Annotations	title2annots	FALSE	DOCUMENT	2.2	
	annotMargin	NULL	DOCUMENT	2.2	
	annotOpacity	0.5	DOCUMENT	2.2	
Bookmarks (Outlines)	anchor2Bookmark	0	DOCUMENT		
	h2bookmarks	array()	DOCUMENT	5.7	Automatically generate bookmarks from Heading elements H1-H6

Category	Variable	Default value	Scope ^[1]	Version	Notes
	h2toc	array()	DOCUMENT	5.7	Automatically generate ToC entries from Heading elements H1-H6
CSS & Styles	CSSselectMedia	"print"	DOCUMENT	4.4	
	disablePrintCSS	NULL	DOCUMENT		DEPRACATED from >= 4.4
	rtlCSS	2	DOCUMENT		REMOVED from v 5.1
	useDefaultCSS2		DOCUMENT		DEPRACATED from >= 2.2
Page Headers & Footers	defaultfooterfontsize	8			
	defaultfooterfontstyle	"BI"			
	defaultfooterline	1			
	defaultheaderfontsize	8			
	defaultheaderfontstyle	"BI"			
	defaultheaderline	1			
	footer_line_spacing	0.25			
	header_line_spacing	0.25			
	forcePortraitHeaders	FALSE	DOCUMENT		
	headerPageNoMarker	"! "	DOCUMENT		DEPRACATED from >= 4.0
Tables	simpleTables	FALSE	DOCUMENT	4.3	
	packTableData	FALSE	DOCUMENT	4.4	
	cacheTables	FALSE	DOCUMENT	5.4	Removed v6.0
	tableMinSizePriority	FALSE		4.6	
	ignore_table_percents	FALSE		2.2	
	ignore_table_widths	FALSE		2.2	
	keep_table_proportions	FALSE		2.2	
	shrink_tables_to_fit	1.4			
	table_error_report	FALSE			
	table_error_report_param	""			
	use_kwt	FALSE		2.0	
	iterationCounter	FALSE	DOCUMENT	5.0	Enables the use of a replaceable iteration counter in table headers or footers
	decimal_align	cf.	DOCUMENT	5.7	Array of characters enabled to align table columns
Images	img_dpi	96			
Text Spacing & Justification	normalLineheight	1.33		4.2	Value used for line-height when CSS specified as 'normal'
	useFixedNormalLineHeight	FALSE		6.0	
	useFixedTextBaseline	FALSE		6.0	
	adjustFontDescLineheight	1.14		6.0	
	jSmaxChar	2			
	jSmaxWordLast	2		5.1	
	jSmaxCharLast	1		5.1	
	jSpacing	NULL			DEPRACATED from >= 5.1
	jSword	0.4			
	orphansAllowed	5			Removed mPDF 5.7
	allowCJKorphans	TRUE		5.2	Wrapping of CJK text
	allowCJKoverflow	FALSE		5.2	Wrapping of CJK text
	CJKforceend	FALSE		5.7	Wrapping of CJK text

Category	Variable	Default value	Scope ^[1]	Version	Notes
	tabSpaces	8		2.3	
	justifyB4br	FALSE		4.4	Justify the line before a when using text-align: justify
Hyphenation	hyphenate	FALSE		2.5	Removed mPDF 5.7
	hyphenateTables	FALSE		2.5	Removed mPDF 5.7
	SHYlang	"en"		2.5	
Columns	keepColumns	FALSE			
	max_colH_correction	1.15			
Lists	list_auto_mode	"browser"		6.0	
	list_indent_default	"40px"		6.0	
	list_indent_default_mpdf	"0em"		6.0	
	list_marker_offset	"5.5pt"		6.0	
	list_symbol_size	"3.6pt"		6.0	
	list_align_style	"R"		2.1	Removed mPDF 6.0
	list_indent_first_level	0			
	list_number_suffix	".."		2.1	
Watermarks	showWatermarkImage	NULL		2.2	
	showWatermarkText	NULL		2.2	
	watermark_font	""			
	watermarkImageAlpha	0.2		2.2	Can be changed by SetWatermarkImage()
	watermarkImgAlphaBlend	"Normal"		4.5	
	watermarkImgBehind	FALSE		4.4	Place watermark images behind page contents
	watermarkTextAlpha	0.2		2.2	
Borders	autoPadding	FALSE	DOCUMENT	3.0	
Bookmarks	bookmarkStyles	array()	DOCUMENT	5.4	Specify appearance of Bookmarks in PDF reader

[1] Variables with scope marked as **DOCUMENT** should only be set once at the beginning of the document. All others can be changed during the course of creating the document.

adjustFontDescLineheight

(mPDF >= 6.0)

Description

```
void adjustFontDescLineheight
```

Specify a factor by which to multiply the font metrics, when determining the text lineheight. When set to 1, the text lineheight will equal the values set in the font for Ascent + Descent.

For more information, see [Line-height](#).

Values

adjustFontDescLineheight

Values

FLOAT : use the when font metrics are being used to set the text line-height. Usual value between 1 and 1.2

DEFAULT: 1.14

Changelog

Version Description

6.0	Variable was added.
-----	---------------------

See Also

[Line-height](#)

aliasNbPg

Variable which defines the text to be replaced by the total page number in the document.

Default = {nb}

You can assign this value directly, or use the function: [AliasNbPages\(\)](#)

Note: This variable originally started with an uppercase letter (<= mPDF 2.2). Support for the uppercase version was removed in mPDF >= 6.0

See Also

- [pagenumPrefix](#) - Specify text to precede page numbers generated by {PAGENO}
- [pagenumSuffix](#) - Specify text to follow page numbers generated by {PAGENO}
- [nbpgPrefix](#) - Specify text to precede page total generated by {nbpg}
- [nbpgSuffix](#) - Specify text to follow page numbers generated by {nbpg}
- [Page numbering](#) -
- [Replaceable aliases](#) -
- [aliasNbPgGp](#) - Specify the text to be replaced by the group page total

aliasNbPgGp

Variable which defines the text to be replaced by the total page number in the page group.

Default = {nbpg}

You can assign this value directly, or use the function: [AliasNbPageGroups\(\)](#)

Note: This variable originally started with an uppercase letter (<= mPDF 2.2). Support for the uppercase version was removed in mPDF >= 6.0

See Also

- [pagenumPrefix](#) - Specify text to precede page numbers generated by {PAGENO}
- [pagenumSuffix](#) - Specify text to follow page numbers generated by {PAGENO}
- [nbpgPrefix](#) - Specify text to precede page total generated by {nbpg}
- [nbpgSuffix](#) - Specify text to follow page numbers generated by {nbpg}
- [Page numbering](#) -
- [Replaceable aliases](#) -
- [aliasNbPg](#) - Specify the text to be replaced by the document page total

allow_charset_conversion

(mPDF >= 1.0)

Description

boolean **allow_charset_conversion**

When **TRUE**, mPDF will attempt to parse the character set of any input HTML. You can also use it together with *charset_in* to manually set an input encoding.

Values

allow_charset_conversion = **TRUE|FALSE**

Values

TRUE: DEFAULT Parse the character set of any input text from the HTML, or allow setting of the value *charset_in*

FALSE: Expect all text input as UTF-8 encoding.

Examples

Example #1

```
<?php

$html = '
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" dir="ltr">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-4" />
<title>Document in Lithuanian</title>
</head>
<body>
... the body of the document encoded in ISO-8859-4 ...
</body>
</html>';

$mpdf=new mPDF();
$mpdf->allow_charset_conversion = true;
$mpdf->WriteHTML($html);

$mpdf->Output();

?>
```

Example #2

```
<?php

$html = '... the body of the document encoded in ISO-8859-4 ...';

$mpdf=new mPDF();
$mpdf->allow_charset_conversion = true;
$mpdf->charset_in = 'iso-8859-4';
$mpdf->WriteHTML($html);

$mpdf->Output();

?>
```

Note: mPDF will convert pages with character sets which work with the PHP function [iconv\(\)](#)

See Also

- [WriteHTML\(\)](#) - Write HTML code to document
- [charset_in](#) - character encoding of input text

allow_html_optional_endtags

(mPDF >= 1.0)

Description

boolean **allow_html_optional_endtags**

When **TRUE**, mPDF will attempt to parse the input HTML allowing for omitted end-tags. Some end tags are optional in HTML 4 (see <http://www.w3.org/TR/1998/REC-html40-19980424/index/elements.html>). These include (not exclusively): P, LI, DD, DT, TR, TD

Note: If the HTML is incorrectly "formed" i.e. with illegal nesting of elements, this may do better or worse than the default.

Values

allow_html_optional_endtags = **TRUE|FALSE**

Values

TRUE: DEFAULT Parse the input HTML allowing for omitted end-tags

FALSE: Expect well formed (X)HTML with closing tags on all elements.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->allow_html_optional_endtags = true;  
$mpdf->WriteHTML('<p>Hallo World');  
$mpdf->WriteHTML('<p>Hallo World');  
$mpdf->WriteHTML('<p>Hallo World');  
$mpdf->Output();  
  
?>
```

allow_output_buffering

(mPDF >= 3.0)

Description

boolean **allow_output_buffering**

When **TRUE**, mPDF will ignore any content in the object buffer - ob_get_contents() - when outputting the PDF file. By default, any buffered output will be treated as an error message and will abort the PDF file output and display any errors to the browser.

Note: If the variable *debug* is set to **FALSE** (default) this variable has no effect.

Values

allow_output_buffering = **FALSE|TRUE**

Values

FALSE: DEFAULT Any buffered output from the script will trigger an error message and abort production of the PDF file.

TRUE: Ignores any content in the object buffer when outputting the PDF file.

See Also

- [Error Messages](#)

allowCJKorphans

(mPDF >= 5.7)

CJK line-breaking is implemented in mPDF roughly according to accepted rules.

Configurable variables allow fine control of behaviour (except in tables):

```
$this->allowCJKorphans = true; // FALSE=always wrap to next line; TRUE=squeeze or overflow  
$this->allowCJKoverflow = false; // FALSE=squeeze; TRUE=overflow (only selected)
```

See also [allowCJKoverflow](#) and [CJKforceend](#)

DRAFT

allowCJKoverflow

(mPDF >= 5.7)

CJK line-breaking is implemented in mPDF roughly according to accepted rules.

Configurable variables allow fine control of behaviour (except in tables):

```
$this->allowCJKorphans = true; // FALSE=always wrap to next line; TRUE=squeeze or overflow  
$this->allowCJKoverflow = false; // FALSE=squeeze; TRUE=overflow (only selected)
```

See also [allowCJKorphans](#) and [CJKforceend](#)

DRAFT

anchor2Bookmark

Note: This variable originally started with an uppercase letter (<= mPDF 2.2). Support for the uppercase version was removed in mPDF >= 6.0

See Also

- [SetAnchor2Bookmark\(\)](#) - Specifies how PDF Book marks are created from HTML anchors

DRAFT

annotMargin

```
float $annotMargin = 0; // default position for Annotations;  
0 = where it was defined;  
(+ve value float) Distance from right margin of page to show annotations;
```

See Also

- [Annotation\(\)](#) - Add an Annotation to the document

DRAFT

annotOpacity

float \$annotOpacity = 0.5; // default Opacity for Annotations;

Value >0 - <= 1

Suggest 0.5 if using in-line with text, or 1 if using in the margins (with annotMargin <> 0)

See Also

- [Annotation\(\)](#) - Add an Annotation to the document

A large, semi-transparent watermark text "DRAFT" is rotated approximately 45 degrees counter-clockwise across the page. The letters are bold and have a light gray fill.

autoArabic

(mPDF >= 6.0)

Description

boolean **autoArabic**

Tells mPDF whether to try and distinguish between Arabic languages when using `autoScriptToLang`.

Analysis of the text will attempt to distinguish Arabic, Farsi, Pashto, Urdu and Sindhi. If active, the text will then be marked with a specific language tag e.g. "pa", "ur", "fa" etc.

Values

`autoArabic = TRUE|FALSE`

Values

FALSE: **DEFAULT** Arabic script will be marked with the attribute `lang='und-Arab'` when using `autoScriptToLang`.

TRUE: mPDF will attempt to distinguish Arabic, Farsi, Pashto, Urdu and Sindhi, and text will then be marked with a specific language tag e.g. "pa", "ur", "fa" etc.

See Also

- [autoScriptToLang](#) - marks up HTML text using the `lang` attribute, based on the Unicode script block in question
- [Automatic Font selection](#)

autoFontSize

(mPDF >= 2.3 <6.0)

autoFontSize - Specify the chunk size of text to group when auto-detecting languages using SetAutoFont

Value

```
void autoFontSize
```

Specify the chunk size of text to group when auto-detecting languages using [SetAutoFont\(\)](#).

Note: This variable is removed from mPDF v 6.0

Bigger chunks (3) allows reversal of whole sentences of RTL text, not just letters in individual words; the disadvantage is that it may include bits of other languages either side, forcing them in the font used for the "foreign" language.

Smaller chunks (1) - analysing word by word - takes more processing time, and cannot reverse RTL sentences. In text with CJK language, it makes it harder for mPDF to correctly identify between e.g. Korean and Chinese which share some characters. Thus words may be identified alternately as Korean or Chinese.

Values

autoFontSize

Values

1: individual words are analysed

2: words are analysed to see if they are distinctive of a particular language, and then surrounding text that is compatible is grouped together with these words

3: as big chunks as possible are grouped, including ASCII characters and punctuation

DEFAULT: 2

Changelog

Version Description

2.3	Variable was added.
-----	---------------------

Examples

Example #1

```
<?php

include("../mpdf.php");
$mpdf=new mPDF('utf-8');

$html = "
<style>
p { font-family: FreeSerif; }
</style>
<p>Most of this text is in English, but has occasional words in Chinese:来自商务 or Vietnamese: Một khảo sát mới cho biết, or maybe even Arabic: إلا يبيض</p>
<p>الإيبيض "تجبر بشدة"</p>
<p>其截至 WHO 年底 2005 答</p>
";

$mpdf->SetAutoFont();

$mpdf->autoFontSize = 1;
$mpdf->WriteHTML($html);
```

```
$mpdf->autoFontSize = 2;
$mpdf->WriteHTML($html);

$mpdf->autoFontSize = 3;
$mpdf->WriteHTML($html);

$html2 = "<p>In this example, the word boundaries from different languages are already defined by
marking with &lt;span&ampgt tags</p>
<p>Most of this text is in English, but has occasional words in Chinese:<span>来自商务</span> or
Vietnamese: <span>Một khảo sát mới cho biết</span>, or maybe even Arabic: <span>الإيصال</span></p>
";
$mpdf->WriteHTML($html2);

$mpdf->Output();

?>
```

See the result of this as a PDF file

See Also

- [useLang](#) - Specify whether to recognise and support the HTML attribute lang
- [SetAutoFont\(\)](#) - Use AutoFont to auto-detect text language in HTML input
- [disableMultilingualJustify](#) - Specify whether to disable text justification in multilingual documents
- [lang](#) - Information on mPDF support for the HTML attribute lang

autoLangToFont

(mPDF >= 6.0)

Description

boolean **autoLangToFont**

When **TRUE**, selects the font to use, based on the HTML lang attribute, using configurable values in config_lang2font.php

Values

autoLangToFont = TRUE|FALSE

Values

TRUE: **DEFAULT** selects the font to use, based on the HTML lang attribute, using configurable values in config_lang2font.php

FALSE: Font selection unaffected.

See Also

- [autoScriptToLang](#) - marks up HTML text using the lang attribute, based on the Unicode script block
- [Automatic Font selection](#)

autoMarginPadding

(mPDF >= 4.0)

Description

mixed **autoMarginPadding**

Specify padding between top-margin and header in 'stretch' mode. When `setAutoTopMargin` or `setAutoBottomMargin` are set to 'stretch' `autoMarginPadding` defines the minimum distance in mm that will be forced between the bottom of the header and the top of the main text (or bottom of text and footer).

Values

autoMarginPadding

Values

FLOAT Value in millimeters

DEFAULT 2

See Also

- [Headers & Top margins](#)
- [setAutoTopMargin -Specify mode of determining top-margin position](#)
- [setAutoBottomMargin -Specify mode of determining bottom-margin position](#)

autoPageBreak

(mPDF >= 3.1)

autoPageBreak – Specify whether to allow automatic page breaks

Description

```
void autoPageBreak
```

Specify whether to allow automatic page breaks. By default, mPDF creates page breaks when required in the document. Setting the value to FALSE allows an oversized object (image etc.) to overwrite the footer and/or the bottom margin of the page.

Values

autoPageBreak = **TRUE** | **FALSE**

Values

TRUE : enables automatic page breaks

FALSE: disable automatic page breaks

DEFAULT: **TRUE**

Changelog

Version	Description
3.1	Variable was added.

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF();
$html = ' ';
$mpdf->autoPageBreak = false;
$mpdf->AddPage();
$mpdf->WriteHTML($html);
$mpdf->Output();
?>
```

Note: This variable existed in the script before v3.1 but did not allow paging to be turned off. The variable was altered in mPDF 3.1 by making the first letter of the variable lowercase.

autoScriptToLang

(mPDF >= 6.0)

Description

boolean **autoScriptToLang**

When **TRUE**, marks up HTML text using the lang attribute, based on the Unicode script block in question, and configurable values in config_script2lang.php.

Values

autoScriptToLang = TRUE|FALSE

Values

TRUE: **DEFAULT** mark up HTML text using the lang attribute, based on the Unicode script block in question, and configurable values in config_script2lang.php.

FALSE: No markup applied.

See Also

- [autoLangToFont](#) - selects the font to use, based on the HTML lang attribute, using configurable values in config_lang2font.php
- [Automatic Font selection](#)

autoVietnamese

(mPDF >= 6.0)

Description

boolean **autoVietnamese**

Tells mPDF whether to try and distinguish the Vietnamese language when using `autoScriptToLang`.

Vietnamese is written using mainly Latin script, and cannot therefore be distinguished from other languages using Latin script. However there are a few characters which are unique to Vietnamese and may enable mPDF to distinguish text in a Vietnamese language.

Because a number of fonts do not contain these unique characters, you may wish mPDF to try and detect Vietnamese so a special font can be selected.

Note: If `autoVietnamese` is **TRUE** mPDF will mark up text detected as Vietnamese even if `baseScript` is set as "1" - ignoring most Latin scripts.

Values

`autoVietnamese` = **TRUE|FALSE**

Values

FALSE: DEFAULT Vietnamese text will be treated as Latin text when using `autoScriptToLang`

TRUE: mPDF will attempt to distinguish Vietnamese, and text will then be marked with `lang= "vi"`

See Also

- [autoScriptToLang](#) - marks up HTML text using the `lang` attribute, based on the Unicode script block in question
- [Automatic Font selection](#)

backupSubsFont

mPDF >= 5.0

Optional array containing font(s) to use for missing characters when using useSubstitutions.

Default set in config_fonts.php

Default value: array('dejavusanscondensed')

Only relevant when using subsets (otherwise would add very large file), and doesn't do Indic or arabic text

More than 1 font can be specified but each will add to the processing time of the script

Names used are as defined in \$this->fontdata in config_fonts.php

Examples:

```
$this->backupSubsFont = array('dejavusanscondensed','arialunicodems','sun-exta'); // this will recognise  
most scripts
```

```
$this->backupSubsFont = array('dejavusanscondensed','arialunicodems'); // good default - REQUIRES Arial  
Unicode MS
```

DRAFT

backupSIPFont

mPDF >= 5.0

Optional font name to use for CJK characters in Plane 2 Unicode (> U+20000) when using useSubstitutions.

Default set in config_fonts.php

Default value: [blank]

Names used are as defined in \$this->fontdata in config_fonts.php

Use a font like hannom or sun-extb if available

Example:

```
$this->backupSIPFont = 'sun-extb';
```

A large, semi-transparent watermark text "DRAFT" is printed diagonally across the page from bottom-left to top-right. The letters are bold and have a light gray fill.

baseScript

(mPDF >= 6.0)

Description

boolean **baseScript**

Tells mPDF which Script to ignore when using `autoScriptToLang`.

Values

`baseScript = INTEGER`

Values

INTEGER: (**DEFAULT = 1**) Number representing the script block to be ignored when using `autoScriptToLang`. It is set by default to "1" which is for Latin script. In this mode, all scripts except Latin script are marked up with "lang" attribute. To select other scripts as the base, see the file `/classes/ucdn.php`

See Also

- [autoScriptToLang](#) - marks up HTML text using the lang attribute, based on the Unicode script block in question
- [Automatic Font selection](#)

biDirectional

This variable is set by mPDF when it detects characters from a script written in a right-to-left direction to allow processing of bidirectional text. It does not normally need to be defined by the user, but can be, in order to force the bidirectional algorithm to be applied.

Note: This variable originally started with an uppercase letter (<= mPDF 2.2). Support for the uppercase version was removed in mPDF >= 6.0

See Also

- [RTL & Bidirectional text](#)

A large, semi-transparent gray watermark-style text reading "DRAFT" in a bold, sans-serif font. The text is oriented diagonally, sloping upwards from the bottom-left towards the top-right. It serves as a placeholder or draft indicator for the page content.

bleedMargin

mPDF >= 5.1

Set the default bleed margin in millimeters used in @page media.

Default set in config.php

Default value: 5

Values: Integer, 0 and above. In millimeters

DRAFT

bookmarkStyles

mPDF >= 5.4

Bookmarks can be styled by adding code as below to your script. You can define a colour (array of RGB) and/or a font-style (B, I, or BI) for each level (starting at 0). Results may depend on the PDF Reader you are using.

Example

Example #1 - In config.php file

```
<?php  
  
$this->bookmarkStyles = array(  
0 => array('color'=> array(0,64,128), 'style'=>'B'),  
1 => array('color'=> array(128,0,0), 'style'=>''),  
2 => array('color'=> array(0,128,0), 'style'=>'I'),  
);  
  
?>
```

charset_in

(mPDF >= 1.0)

Description

string **charset_in**

Defines the character encoding of any input HTML. Use it together with [allow_charset_conversion](#) to manually set an input encoding.

Values

charset_in

Values (case-insensitive)

Any string value allowed which is valid for the PHP function [iconv\(\)](#). This appears to vary depending on the local configuration. See the manual entry for [iconv](#) for usual values. You may need to use cp1252 or windows-1252 instead of win-1252, or iso-88591 instead of iso-8859-1.

BLANK or omitted: Expect all text input as UTF-8 encoding.

Examples

Example #1

```
<?php

$html = '... the body of the document encoded in ISO-8859-4 ...';

$mpdf=new mPDF();
$mpdf->allow_charset_conversion = true;
$mpdf->charset_in = 'iso-8859-4';
$mpdf->WriteHTML($html);

$mpdf->Output();

?>
```

See Also

- [WriteHTML\(\)](#) - Write HTML code to document
- [allow_charset_conversion](#) - Activates character encoding conversion of input text
- [iconv](#) - list of values accepted for this variable

CJKforceend

(mPDF >= 5.7)

CJK line-breaking is implemented in mPDF roughly according to accepted rules.

Configurable variables allow fine control of behaviour:

```
$this->CJKforceend = false; // Forces overflowing punctuation to hang outside right margin (used with CJK script)
```

```
$this->allowCJKorphans = true; // FALSE=always wrap to next line; TRUE=squeeze or overflow  
$this->allowCJKoverflow = false; // FALSE=squeeze; TRUE=overflow (only selected)
```

IF \$this->allowCJKorphans == true AND \$this->allowCJKoverflow == true AND \$this->CJKforceend == true AND text-align:justify
will force hanging punctuation to hang outside right margin.

See also [allowCJKoverflow](#) and [allowCJKorphans](#)

collapseBlockMargins

(mPDF >= 4.2)

`collapseBlockMargins` – Specify whether to collapse (vertical) margins between block elements

Description

```
void collapseBlockMargins
```

Specify whether to collapse (vertical) margins between block elements. In line with CSS specification, the top/bottom margins of adjoining block-style elements are collapsed to the larger of the two. This works between all block elements such as DIV, P, H1-6 etc. and also lists and tables.

NB Firefox does not collapse margins above and below tables, but IE8 does.

Values

`collapseBlockMargins = TRUE | FALSE`

Values

TRUE : enable collapse

FALSE: disable collapse

DEFAULT: **TRUE**

Changelog

Version Description

4.2	Variable was added.
-----	---------------------

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF();

$html = '
<div style="margin-bottom: 3em;">This DIV has a bottom margin defined as 3em</div>
<div style="margin-top: 2em;">This DIV has a top margin defined as 2em. The space between these DIVs
will collapse to 3em</div>';

$mpdf->WriteHTML($html);
$mpdf->Output();
?>
```

Note: The collapse of margins at the top and bottom of the page is unaffected by this variable. This is set by the custom CSS property `margin-collapse: collapse|none` which can be defined in the `defaultCSS` variable in `config.php` or elsewhere.

cropMarkLength

mPDF >= 5.1

Default length in mm of crop line

Default set in config.php

Default value: 18

DRAFT

cropMarkMargin

mPDF >= 5.1

Distance of crop mark from margin in mm

Default set in config.php

Default value: 8

DRAFT

crossMarkMargin

mPDF >= 5.1

Distance of cross mark from margin in mm

Default set in config.php

Default value: 5

DRAFT

CSSselectMedia

(mPDF >= 4.4)

CSSselectMedia - Selects which media-dependent CSS stylesheets to use

Description

```
void CSSselectMedia
```

Selects which media-dependent CSS stylesheets to use. mPDF supports internal and external CSS stylesheets, if the media property is set to "all", or matches the value of *CSSselectMedia*. The @media selector within stylesheets is also supported.

Note: This variable can be changed either in the configuration file config.php or at runtime

Values

CSSselectMedia

Values

"print"
"screen"
[any other valid CSS @media selector, except "all"]
DEFAULT: "print"

Changelog

Version	Description
4.4	Variable was added.

decimal_align

Decimal Mark alignment in table columns

CSS text-align supports decimal mark characters in table cells (TD and TH). HTML attributes "align" and "char" can also be used.

Characters to be used for alignment must be defined in the array \$this->decimal_align in config.php

By default these are : period "." comma "," middot "\B7" and arabic decimal mark "\66B"

```
$this->decimal_align = array('DP'=>'.', 'DC'=>', ', 'DM'=>"\xc2\xb7", 'DA'=>"\xd9\xab", 'DD'=>'-' );
```

The default character is a period.

To add additional characters, edit config.php

- The UTF-8 representation of any non-ASCII characters must be used
- Use any unused 2 character code starting with D for the array key

debug

(mPDF >= 3.1)

debug - Turn on debugging messages

Description

```
void debug
```

Specify whether to show debugging messages. If you are having problems with mPDF, set *debug* to TRUE to show error and warning messages that may otherwise be suppressed.

Values

debug = **TRUE** | **FALSE**

Values

TRUE : enable debugging

FALSE: disable debugging

DEFAULT: FALSE

Changelog

Version Description

3.1	Variable was added.
-----	---------------------

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF();
$mpdf->debug = true;
$mpdf->WriteHTML("Hello World");
$mpdf->Output();
?>
```

Note: This was introduced in mPDF 3.1 as the method used by mPDF to catch error messages inadvertently picked up suppressed errors such as those caused by @fopen() - even though the @ is deliberately there to prevent interruption of the script.

See Also

- [showImageErrors](#) - show/hide error reporting for problems with Images
- [allow_output_buffering](#) - prevent error mesages when using output buffering

debugfonts

mPDF >= 5.0

Show errors and warning notes for fonts.

Default set in config.php

Default value: False

Values: true|false

DRAFT

displayDefaultOrientation

`$mpdf->displayDefaultOrientation = true; (default false)`

If default page orientation is P(ortrait), any L(andscape) pages in the document are displayed in Adobe Reader rotated to appear in portrait. Reverse is true if default orientation is landscape.

DRAFT

defaultPageNumStyle

(mPDF >= 6.0)

Description

```
void defaultPageNumStyle
```

Specifies a default page number style to use from the start of the document.

For more information, see [page numbering](#).

Values

defaultPageNumStyle

Values

(Uses the same values as for list-style-type)

1 | A | a | I | i | disc | circle | square | decimal | lower-roman | upper-roman | lower-latin | upper-latin | lower-alpha | upper-alpha | none
arabic-indic | bengali | cambodian | cjk-decimal | devanagari | gujarati | gurmukhi | hebrew |kannada | khmer | lao | malayalam | oriya | persian | telugu | thai | urdu | tamil
"1" - decimal
"A" = upper-latin
"a" = lower-latin
"I" = upper-roman
"i" = lower-roman
DEFAULT: "1"

Changelog

Version	Description
6.0	Variable was added.

See Also

- [Page numbering](#)

dpi

(mPDF >= 4.5)

Description

```
void dpi
```

Specify how to convert sizes specified in "px" units (pixels). As a print medium, PDF documents do not have any inherent size for pixels. Any values set for e.g. font-size, border-width etc. need to be converted to a real length. The dots-per-inch (dpi) affects all conversions from pixels **except images**, which are set by a separate variable *img_dpi*

Note: It is recommended that the values for *dpi* and *img_dpi* are the same.

Note: This variable can be changed in the configuration file config.php

Values

dpi

Values

INTEGER : set conversion for pixel - dots per inch

DEFAULT: 96

Changelog

Version Description

Version	Description
4.5	Variable was added.

See Also

- [img_dpi](#) - Specify conversion for image sizes set in pixels

enableImports

(mPDF >= 4.3)

enableImports - Enable the use of imported PDF files or templates

Description

```
void enableImports
```

Enable the use of imported PDF files or templates. This causes additional files (classes) to be loaded, enabling several functions allowing you to import PDF files into the document you are writing, and using templates.

Note: You should only change this variable in the configuration file config.php If you want to set this at runtime, use SetImportUse()

Note: Prior to mPDF 4.3, this required calling mPDFI(). The functions have now been incorporated into the main mpdf.php file, but you must set this variable to enable them.

Values

enableImports = **TRUE** | **FALSE**

Values

TRUE : enable import/template functions

FALSE: disabled

DEFAULT: **FALSE**

Changelog

Version	Description
4.3	Variable was added.

See Also

- [SetImportUse\(\)](#) - Enable the use of imported PDF files or templates
- [Thumbnail\(\)](#) - Print thumbnails of an external PDF file
- [SetSourceFile\(\)](#) - Specify the source PDF file used to import pages into the document
- [ImportPage\(\)](#) - Import a page from an external PDF file
- [UseTemplate\(\)](#) - Insert an imported page from an external PDF file
- [SetPageTemplate\(\)](#) - Specify a page from an external PDF file to use as a template
- [SetDocTemplate\(\)](#) - Specify an external PDF file to use as a template
- [RestartDocTemplate\(\)](#) - Re-start the use of a Document template from the next page

falseBoldWeight

(mPDF >= 4.2)

falseBoldWeight - Specify weight used for bold text when using an artificial (outline) bold

Description

```
void falseBoldWeight
```

Specify weight used for bold text when using an artificial (outline) bold. If bold text is set by **** and the current font does not have a font file for the bold variant, an artificial bold is created by stroking the outline of the characters. This variable sets the width of the line and thus the "weight" of the bold text. Values between 0 and 10 are recommended.

Values

falseBoldWeight

Values

INTEGER : set weight of bold text

DEFAULT: 5

Changelog

Version Description

4.2	Variable was added.
-----	---------------------

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF();
$mpdf->falseBoldWeight = 8;
$mpdf->WriteHTML('<p style="font-family: mysimplefont"><b>Hello World</b><p>');
$mpdf->Output();
?>
```

Note: False bold text has the same character widths as the normal text.

forcePortraitHeaders

=====

Special case - rotated Portrait headers on Landscape pages

=====

```
$mpdf->forcePortraitHeaders = true;
```

In v2.3 this was updated to work for non-HTML headers/footers as well.

This is a quick fix which rotates HTML headers and Footers on landscape pages in the following conditions:

- document orientation is portrait
- when adding a page, you must only call `$mpdf->AddPage('L')` or `<pagebreak orientaion="landscape" />` - if you try to set new margins/headers etc. for new landscape pages when `forcePortrait=true`, it will go wrong

DRAFT

forcePortraitMargins

mPDF >= 2.3

=====

Special case - rotated margins on Landscape pages

=====

\$mpdf->forcePortraitMargins = true; (default = false)

If default page orientation is P(ortrait), then adding new page L(andscape) causes the left/right margins to be used as top/bottom, and vice-versa.

(NB similar to \$forcePortraitHeaders which does the same but also rotates the [HTML] headers/footers.)

NB - if using OddEven (double-sided document), it is the Portrait orientation L/R margins that alternate.

When adding a page, you must only call \$mpdf->AddPage('L') or <pagebreak orientaion="landscape" /> - if you try to set new margins/headers etc. for new landscape pages when forcePortraitMargins=true, it will go wrong

DRAFT

h2bookmarks

(mPDF >= 5.7)

Description

boolean **h2bookmarks**

When a valid array is set, mPDF will automatically generate PDF Bookmarks from all heading elements (H1 - H6) written to the document using `WriteHTML()`.

Any number of levels may be specified, but the first level should be 0 and other levels should be consecutive.

Values

h2bookmarks

Values

An array should be specified mapping the headings to the required level of Bookmarks
e.g. `array('H1'=>0, 'H2'=>1, 'H3'=>2)`

DEFAULT: `array()`

h2toc

(mPDF >= 5.7)

Description

boolean **h2toc**

When a valid array is set, mPDF will automatically generate entries for a Table of Contents using all heading elements (H1 - H6) written to the document using `WriteHTML()`.

Any number of levels may be specified, but the first level should be 0 and other levels should be consecutive.

Values

h2toc

Values

An array should be specified mapping the headings to the required level of Table of Contents
e.g. `array('H1'=>0, 'H2'=>1, 'H3'=>2)`

DEFAULT: `array()`

ICCProfile

(mPDF >= 4.3)

ICCProfile - Specify ICC colour profile to use when creating PDF/A1-b or PDF/X-1a (mPDF >= 5.1) compliant documents

Description

```
void ICCProfile
```

Specify ICC colour profile to use when creating PDF/A1-b or PDF/X-1a compliant documents. The file must exist in /iccprofiles/ folder

Either a 3-colour RGB profile or a 4-colour CMYK profile should be used depending on other settings.

The default colorspace for a PDF/A1-b document is RGB. The default for a PDF/X-1a document is CMYK.

Specify the name without the .icc extension. sRGB_IEC61966-2-1 is used as the default file for PDF/A1-b documents if none is specified (a profile must be specified for PDF/X-1a)

Values

ICCProfile = **STRING**

Name of a valid ICC colour profile in the /iccprofiles/ folder - without the .icc extension.

DEFAULT: BLANK

Changelog

Version	Description
---------	-------------

4.3	Variable was added.
-----	---------------------

See Also

- [PDF/A1-b compliance](#)
- [PDF/X-1a compliance](#)
- [PDFA - Create PDF/A1-b compliant document](#)
- [PDFX - Create PDF/X-1a compliant document](#)
- [PDFAauto - Specify whether to automatically fix issues to create PDF/A1-b compliant document](#)
- [PDFXauto - Specify whether to automatically fix issues to create PDF/X-1a compliant document](#)
- [restrictColorSpace - Specify whether to automatically limit the colorspaces used](#)

img_dpi

(mPDF >= 1.0)

img_dpi - Specify size conversion for images using pixels

Description

```
void img_dpi
```

Specify how to convert image sizes specified in "px" units (pixels). As a print medium, PDF documents do not have any inherent size for pixels. Width and height values of images set in pixels need to be converted to a real length. The dots-per-inch (dpi) affects the conversion from pixels for images; other objects using pixels as a length, are set by a separate variable *dpi*

Note: It is recommended that the values for *dpi* and *img_dpi* are the same.

Note: This variable can be changed in the configuration file config.php

Values

img_dpi

Values

INTEGER : set conversion for pixel - dots per inch

DEFAULT: 96

Changelog

Version Description

1.0	Variable was added.
-----	---------------------

See Also

- [dpi](#) - Specify conversion for other values set in pixels

incrementFPR1 [1-4]

(mPDF >= 4.2)

incrementFPR1 - Adjust auto-fit for fixed position block elements

incrementFPR2

incrementFPR3

incrementFPR4

Description

```
void incrementFPR1
```

When writing a block element with `position:fixed` and `overflow:auto`, mPDF scales it down to fit in the space by repeatedly rewriting it and making adjustments. These values give the adjustments used, depending how far out the previous guess was. The lower the number, the quicker it will finish, but the less accurate the fit may be.

FPR1 is for coarse adjustments, and FPR4 for fine adjustments when it is getting closer.

Values

incrementFPR1 [1-4]

Values

INTEGER : recommended between 10-100

DEFAULT: 10, 20, 30, 50

Changelog

Version Description

Version	Description
4.2	Variables were added.

iterationCounter

mPDF >= 5.0

Allow use of {iteration varname} in THEAD

Default set in config.php or at runtime

Default value: FALSE

Values: TRUE|FALSE

DRAFT

jSmaxChar

Maximum spacing to allocate to character spacing. (0 = no maximum)

Set in config.php or at runtime

Default value: 2

DRAFT

jSmaxCharLast

Maximum character spacing allowed (carried over) when finishing a last line.

Default set in config.php

```
$this->jSmaxCharLast = 1;
```

DRAFT

jSmaxWordLast

Maximum word spacing allowed (carried over) when finishing a last line.

Default set in config.php

```
$this->jSmaxWordLast = 2;
```

DRAFT

jSWord

Proportion (/1) of space (when justifying margins) to allocate to Word vs. Character

DRAFT

justifyB4br

(mPDF >= 4.4)

justifyB4br - Specify whether to justify line of text before a linebreak

Description

```
void justifyB4br
```

Specify whether to justify line of text before a linebreak. This only makes a difference when the text-alignment of the current block is set as "text-align: justify".

In a justified text block, a `
` linebreak will not cause the preceding line to be justified. This behaviour matches most browsers. However, to allow optional compliance with the behaviour of MS Word, which also justifies text before a `
` you can set this variable to **TRUE**.

Values

`justifyB4br = TRUE | FALSE`

Values

TRUE : justify line of text before a linebreak

FALSE: do not justify line of text before a linebreak

DEFAULT: **FALSE**

Changelog

Version	Description
4.4	Variable was added.

keepColumns

Note: This variable originally started with an uppercase letter (<= mPDF 2.2). Support for the uppercase version was removed in mPDF >= 6.0

Set this variable to **TRUE** and columns will be written successively i.e. there will be no balancing of the length of columns.

```
$mpdf->keepColumns = true;
```

DRAFT

keep_table_proportions

If table width is set, and is > page width, setting this value to true forces the table to keep relative sizes when resizing.

It also forces respect of cell widths set by %

DRAFT

list_align_style

(mPDF >= 2.1 < 6.0)

Note: This was removed in mPDF 6.0

Specifies text alignment of numbers in numbered lists (default Right)

\$mpdf->list_align_style = 'L';

Default = "R"

(This can be altered at run time, but is not changeable through stylesheets or in-line style)

DRAFT

list_auto_mode

(mPDF >= 6.0)

Description

```
void list_auto_mode
```

Specify whether to use mPDF custom method of automatic indentation of lists, or standard browser-compatible. The custom mPDF method is ignored if `list-style-position: inside`, or image used for marker (or custom U+).

For more information, see [Lists](#).

Values

list_auto_mode

Values

browser - list display will conform to standard browser behaviour for automatic indentation of lists

mpdf - list display will be consistent with mPDF behaviour prior to v6.0

DEFAULT: browser

Changelog

Version Description

6.0	Variable was added.
-----	---------------------

See Also

- [Lists](#)

list_indent_default

(mPDF >= 6.0)

Description

```
void list_indent_default
```

Define the default indentation of a list item, when in standard 'browser' list mode.

For more information, see [Lists](#).

Values

list_indent_default

Values

LENGTH: Any valid CSS length value is permitted e.g. "10pt" or "3em"

DEFAULT: '40px'

Changelog

Version	Description
---------	-------------

6.0	Variable was added.
-----	---------------------

See Also

[Lists](#)

list_indent_default_mpdf

(mPDF >= 6.0)

Description

```
void list_indent_default_mpdf
```

Define the default indentation of a list item, when in (backwards-compatible) 'mpdf' list mode. Note that in mpdf mode, the real indentation is calculated by adding the value of this property to the width of the widest list-marker.

For more information, see [Lists](#).

Values

list_indent_default_mpdf

Values

LENGTH: Any valid CSS length value is permitted e.g. "10pt" or "3em"

DEFAULT: '0em'

Changelog

Version	Description
---------	-------------

6.0	Variable was added.
-----	---------------------

See Also

[Lists](#)

list_indent_first_level

Description

```
void list_indent_first_level
```

Specify whether to indent the first level of a list. From mPDF >= 6.0, this will only apply if you are using the "mpdf" list mode.

For more information, see [Lists](#).

Values

list_indent_first_level= 1|0

Values

- 1: **DEFAULT** - Indent the first level of a list (when using the "mpdf" list mode)
- 0: No indentation.

See Also

[Lists](#)

list_marker_offset

(mPDF >= 6.0)

Description

```
void list_marker_offset
```

Define the offset of a list marker from the list-item content, when using "bullet" markers i.e. disc/circle/square.

For more information, see [Lists](#).

Values

list_marker_offset

Values

LENGTH: Any valid CSS length value is permitted e.g. "10pt" or "3em"

DEFAULT: '5.5pt'

Changelog

Version Description

6.0	Variable was added.
-----	---------------------

See Also

[Lists](#)

list_number_suffix

mPDF >= 2.1

Specifies content to follow a numbered list marker e.g. '.' gives 1. or IV. whereas ')' gives 1) or a)

```
$mpdf->list_number_suffix = '!';
```

Default = "."

(This can be altered at run time, but is not changeable through stylesheets or in-line style)

DRAFT

list_symbol_size

(mPDF >= 6.0)

Description

```
void list_symbol_size
```

Define the size of a list marker, when using "bullet" markers i.e. disc/circle/square.

For more information, see [Lists](#).

Values

list_symbol_size

Values

LENGTH: Any valid CSS length value is permitted e.g. "10pt" or "3em"

DEFAULT: '3.6pt'

Changelog

Version	Description
---------	-------------

6.0	Variable was added.
-----	---------------------

See Also

[Lists](#)

margBuffer

(mPDF >= 5.5)

Allows an (empty) end of block to extend beyond the bottom margin by this amount (mm). Avoids just the border/background-color of the end of a block being moved on to next page.

DEFAULT 2

DRAFT

max_colH_correction

(mPDF >= 1.0)

max_colH_correction - Sets maximum ratio to allow when adjusting column heights

Description

max_colH_correction (1.15 | float)

The maximum ratio to adjust column height when justifying - too large a value can give ugly results

Note: The vAlign parameter of <columnbreak> or SetColumns() must be set to J or justify

Values

value	description
default	1.15
range	Values above 1.0

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->max_colH_correction = 1.3;  
  
?>
```

maxTTFFFilesize

mPDF >= 5.0

Control fonts/subsetting

// Set maximum size of TTF font file to allow non-subsets - in kB

// Used to avoid e.g. Arial Unicode MS (perhaps used for substituteCharsMB) to ever be fully embedded

// NB Free serif is 1.5MB, most files are <= 600kB (most 200-400KB)

Default set in config.php

Default value: 2000

DRAFT

mirrorMargins

(mPDF >= 4.0)

Description

Set to 1, the document will mirror the left and right margin values on odd and even pages i.e. they become inner and outer margins. (this is automatically reversed for RTL languages).

NB Headers and footers use the 'Odd' pages as default if this is not used.

Default = 0

See Also

- Double-sided documents

DRAFT

nbpgPrefix

(mPDF >= 3.0)

Description

string **nbpgPrefix**

Specify text to precede the page total generated by {nbpg }. (*There should be no space after the 'g'. This is put here to allow it to print in mPDF*)

Note: This is only recommended in non-HTML headers and footers. Although the text is added correctly in HTML headers & footers, the text alignment is not readjusted after substitution e.g. if it used in the right margin.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->pagenumPrefix = 'Page number ' ;  
$mpdf->pagenumSuffix = ' - ' ;  
$mpdf->nbpgPrefix = ' out of ' ;  
$mpdf->nbpgSuffix = ' pages' ;  
  
$mpdf->SetHeader('{PAGENO}{nbpg}') ;  
  
$mpdf->WriteHTML("Hallo World");  
  
$mpdf->Output();  
  
?>
```

Will output a header:
"Page number 1 - out of 1 pages"

See Also

- [pagenumPrefix](#) - Specify text to precede page numbers generated by {PAGENO}
- [pagenumSuffix](#) - Specify text to follow page numbers generated by {PAGENO}
- [nbpgSuffix](#) - Specify text to follow page numbers generated by {nbpg }
- [Page numbering](#) -
- [Replaceable aliases](#) -
- [aliasNbPg](#) - Specify the text to be replaced by the document page total
- [aliasNbPgGp](#) - Specify the text to be replaced by the group page total

nbpgSuffix

(mPDF >= 3.0)

Description

string **nbpgPrefix**

Specify text to follow the page total generated by {nbpg}. (There should be no space after the 'g'. This is put here to allow it to print in mPDF)

Note: This is only recommended in non-HTML headers and footers. Although the text is added correctly in HTML headers & footers, the text alignment is not readjusted after substitution e.g. if it used in the right margin.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->pagenumPrefix = 'Page number ';  
$mpdf->pagenumSuffix = ' - ';  
$mpdf->nbpgPrefix = ' out of ';  
$mpdf->nbpgSuffix = ' pages';  
  
$mpdf->SetHeader('{PAGENO}{nbpg}');  
  
$mpdf->WriteHTML("Hallo World");  
  
$mpdf->Output();  
  
?>
```

Will output a header:
"Page number 1 - out of 1 pages"

See Also

- [pagenumPrefix](#) - Specify text to precede page numbers generated by {PAGENO}
- [pagenumSuffix](#) - Specify text to follow page numbers generated by {PAGENO}
- [nbpgPrefix](#) - Specify text to precede page total generated by {nbpg}
- [Page numbering](#) -
- [Replaceable aliases](#) -
- [aliasNbPg](#) - Specify the text to be replaced by the document page total
- [aliasNbPgGp](#) - Specify the text to be replaced by the group page total

nonPrintMargin

mPDF >= 5.1

Non-printable border at edge of paper sheet in mm

Default set in config.php

Default value: 8

When using Crop- and cross-marks in @page media

DRAFT

normalLineheight

(mPDF >= 4.2)

normalLineheight - Define the default line-height

Description

```
void normalLineheight
```

This variable defines the default line-height used when the CSS property line-height is set to normal (default).

Note: From mPDF v 6.0 onwards, this value will only be used when the variable useFixedNormalLineHeight is set to **TRUE**

For more information, see [Line-height](#).

Values

normalLineheight

Values

FLOAT : Usual value between 1.1 and 1.5

DEFAULT: 1.33

Changelog

Version Description

4.2	Variable was added.
-----	---------------------

See Also

[Line-height](#)

Further reading

- <http://office.microsoft.com/en-us/word/HP100165231033.aspx>
- <http://typophile.com/node/13081>

packTableData

(mPDF >= 4.4)

packTableData – Use binary packing of table data to reduce memory usage

Description

```
void packTableData
```

Processing tables uses large amounts of internal memory, as the value are stored in an array. Enabling packTableData causes mPDF to pack the table data into a binary form saving considerable memory. However, the conversion to and from binary data takes a significant amount of time, and can increase processing time.

Note: This variable can be changed either in the configuration file config.php or at runtime

Values

packTableData = **TRUE** | **FALSE**

Values

TRUE : use binary packing of table data

FALSE: does not use binary packing of table data

DEFAULT: FALSE

Changelog

Version	Description
---------	-------------

4.4	Variable was added.
-----	---------------------

See Also

- [simpleTables](#) - Disables complex table borders etc. to improve performance

pagenumPrefix

(mPDF >= 3.0)

Description

string **pagenumPrefix**

Specify text to precede the page number when using {PAGENO} to insert page numbers in headers or footers.

Note: This is only recommended in non-HTML headers and footers. Although the text is added correctly in HTML headers & footers, the text alignment is not readjusted after substitution e.g. if it used in the right margin.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->pagenumPrefix = 'Page number ';  
$mpdf->pagenumSuffix = ' - ';  
$mpdf->nbpqPrefix = ' out of ';  
$mpdf->nbpqSuffix = ' pages';  
  
$mpdf->SetHeader('{PAGENO}{nbpq}');  
  
$mpdf->WriteHTML("Hallo World");  
  
$mpdf->Output();  
  
?>
```

Will output a header:
"Page number 1 - out of 1 pages"

See Also

- [pagenumSuffix](#) - Specify text to follow page numbers generated by {PAGENO}
- [nbpqPrefix](#) - Specify text to precede page total generated by {nbpq}
- [nbpqSuffix](#) - Specify text to follow page numbers generated by {nbpq}
- [Page numbering](#) -
- [Replaceable aliases](#) -
- [aliasNbPg](#) - Specify the text to be replaced by the document page total
- [aliasNbPgGp](#) - Specify the text to be replaced by the group page total

pagenumSuffix

(mPDF >= 3.0)

Description

```
string pagenumSuffix
```

Specify text to follow the page number when using {PAGENO} to insert page numbers in headers or footers.

Note: This is only recommended in non-HTML headers and footers. Although the text is added correctly in HTML headers & footers, the text alignment is not readjusted after substitution e.g. if it used in the right margin.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->pagenumPrefix = 'Page number ';  
$mpdf->pagenumSuffix = ' - ';  
$mpdf->nbpqPrefix = ' out of ';  
$mpdf->nbpqSuffix = ' pages';  
  
$mpdf->SetHeader('{PAGENO}{nbpq}');  
  
$mpdf->WriteHTML("Hallo World");  
  
$mpdf->Output();  
  
?>
```

Will output a header:
"Page number 1 - out of 1 pages"

See Also

- [pagenumPrefix](#) - Specify text to precede page numbers generated by {PAGENO}
- [nbpqPrefix](#) - Specify text to precede page total generated by {nbpq}
- [nbpqSuffix](#) - Specify text to follow page numbers generated by {nbpq}
- [Page numbering](#) -
- [Replaceable aliases](#) -
- [aliasNbPg](#) - Specify the text to be replaced by the document page total
- [aliasNbPgGp](#) - Specify the text to be replaced by the group page total

PDFA

(mPDF >= 4.3)

PDFA - Create PDF/A1-b compliant document

Description

```
void PDFA
```

Specify whether to create a PDF/A1-b compliant document.

Values

PDFA = **TRUE** | **FALSE**

Values

TRUE : create PDF/A1-b compliant document

FALSE: normal PDF document

DEFAULT: FALSE

Changelog

Version	Description
4.3	Variable was added.

See Also

- [PDF/A1-b compliance](#)
- [PDF/X-1a compliance](#)
- [PDFX - Create PDF/X-1a compliant document](#)
- [PDFAauto - Specify whether to automatically fix issues to create PDF/A1-b compliant document](#)
- [PDFXauto - Specify whether to automatically fix issues to create PDF/X-1a compliant document](#)
- [ICCProfile - Specify the ICC profile for the chosen colorspace used in the document](#)
- [restrictColorSpace - Specify whether to automatically limit the colorspaces used](#)

PDFAuto

(mPDF >= 4.3)

PDFAuto - Specify whether to automatically fix issues to create PDF/A1-b compliant document

Description

```
void PDFAuto
```

Specify whether to automatically fix issues to create PDF/A1-b compliant document

Values

PDFAuto = **TRUE** | **FALSE**

Values

TRUE : automatically fix issues to create PDF/A1-b compliant document

FALSE: show warning messages if changes are required for compliance

DEFAULT: FALSE

Changelog

Version	Description
4.3	Variable was added.

See Also

- [PDF/A1-b compliance](#)
- [PDF/X-1a compliance](#)
- [PDFA - Create PDF/A1-b compliant document](#)
- [PDFX - Create PDF/X-1a compliant document](#)
- [PDFXauto - Specify whether to automatically fix issues to create PDF/X-1a compliant document](#)
- [ICCProfile - Specify the ICC profile for the chosen colorspace used in the document](#)
- [restrictColorSpace - Specify whether to automatically limit the colorspaces used](#)

PDFX

(mPDF >= 5.1)

PDFX - Create PDF/X-1a compliant document

Description

```
void PDFX
```

Specify whether to create a PDF/X-1a compliant document.

Values

PDFX = **TRUE** | **FALSE**

Values

TRUE : create PDF/X-1a compliant document

FALSE: normal PDF document

DEFAULT: FALSE

Changelog

Version	Description
---------	-------------

5.1	Variable was added.
-----	---------------------

See Also

- [PDF/A1-b compliance](#)
- [PDF/X-1a compliance](#)
- [PDFA - Create PDF/A1-b compliant document](#)
- [PDFAauto - Specify whether to automatically fix issues to create PDF/A1-b compliant document](#)
- [PDFXauto - Specify whether to automatically fix issues to create PDF/X-1a compliant document](#)
- [ICCProfile - Specify the ICC profile for the chosen colorspace used in the document](#)
- [restrictColorSpace - Specify whether to automatically limit the colorspaces used](#)

PDFXauto

(mPDF >= 5.1)

PDFXauto - Specify whether to automatically fix issues to create PDF/X-1a compliant document

Description

```
void PDFXauto
```

Specify whether to automatically fix issues to create PDF/X-1a compliant document

Values

PDFXauto = **TRUE** | **FALSE**

Values

TRUE : automatically fix issues to create PDF/X-1a compliant document

FALSE: show warning messages if changes are required for compliance

DEFAULT: FALSE

Changelog

Version	Description
---------	-------------

5.1	Variable was added.
-----	---------------------

See Also

- [PDF/A1-b compliance](#)
- [PDF/X-1a compliance](#)
- [PDFA - Create PDF/A1-b compliant document](#)
- [PDFX - Create PDF/X-1a compliant document](#)
- [PDFAuto - Specify whether to automatically fix issues to create PDF/A1-b compliant document](#)
- [ICCPProfile - Specify the ICC profile for the chosen colorspace used in the document](#)
- [restrictColorSpace - Specify whether to automatically limit the colorspaces used](#)

percentSubset

mPDF >= 5.0

Control fonts/subsetting

// If not -s (i.e. forced subset) this value determines whether to subset or not

// 0 - 100 = percent characters

// i.e. if ==40, mPDF will embed whole font if >40% characters in that font

// or embed subset if <40% characters

// 0 will force whole file to be embedded

// 100 will force always to subset

Default set in config.php

Default value: 30

DRAFT

printers_info

mPDF >= 5.1

Adds date and page info for printer when using @page and "marks:crop;"

Default set in config.php

Default value: False

Values: true|false

DRAFT

progbar_altHTML

mPDF >= 5.0

Customisable Progress bar

Default set in config.php

Default value: "

```
// e.g. '<html><body>Creating PDF file. Please wait...</body></html>'  
// Should include <html> and <body> but NOT end tags  
// Can include <head> and link to stylesheet etc.
```

See Also

- [StartProgressBarOutput\(\)](#) - show progress bars during file generation
- [progressBar](#) -Specify whether to show progress bars during file generation
- [progbar_heading](#) - define customised heading for progress bars

progar_heading

mPDF >= 5.0

Customisable Progress bar

Default set in config.php

Default value: 'mPDF file progress'

See Also

- [StartProgressBarOutput\(\)](#) - show progress bars during file generation
- [progressBar](#) -Specify whether to show progress bars during file generation
- [progar_altHTML](#) - define customised HTML for progress bars

DRAFT

progressBar

(mPDF >= 4.2)

progressBar -Specify whether to show progress bars during file generation

Description

```
void progressBar
```

Specify whether to show progress bars during file generation. Not recommended for general use, but may be helpful for development purposes, or for slow document generation.

Note: You should only change this variable in the configuration file config.php If you want to set this at runtime, use [StartProgressBarOutput\(\)](#)

Note: You may need to define _MPDF_URI if you are using progress bars - see [StartProgressBarOutput\(\)](#)

Values

progressBar = 2 | 1 | FALSE

Values

2: display multiple progress bars for detailed examination of progress

1 : display 1 progress bar (simple form)

FALSE: disable progress bars

DEFAULT: FALSE

Changelog

Version Description

Version	Description
4.2	Variable was added.

See Also

- [StartProgressBarOutput\(\)](#) - show progress bars during file generation
- [progbar_heading](#) - define customised heading for progress bars
- [progbar_altHTML](#) - define customised HTML for progress bars

repackageTTF

When Embedding full TTF font files, setting this variable to TRUE forces mPDF to remake the font file using only core tables.

This may improve function with some PostScript printers (GhostScript/GSView)

`$this->repackageTTF = false;(default)`

Does not work with TTC font collections

Produces a slightly smaller PDF file; increased processing time.

A large, semi-transparent watermark text "DRAFT" is rotated approximately 45 degrees counter-clockwise. The letters are bold and have a light gray fill. They are positioned in the upper-left quadrant of the page, with the letters overlapping each other.

restoreBlockPagebreaks

(mPDF >= 2.3 <= 6.0)

Description

boolean **restoreBlockPageBreaks**

Specifies whether or not to restore open HTML block elements after a forced pagebreak. When a pagebreak is forced by [AddPage\(\)](#) or [<pagebreak>](#), mPDF by default will close any HTML block elements, expecting the HTML code to start afresh after the pagebreak. If this value is set to **TRUE** mPDF will attempt to carry over any CSS style values for the current block elements and continue after the pagebreak.

Note: This variable was removed in mPDF 6.0 See [Page Breaks](#) for more information

Values

restoreBlockPageBreaks = **TRUE|FALSE**

Values

TRUE: Restore block element properties after a pagebreak

FALSE: DEFAULT Do not restore block elements after a pagebreak

Values of 1 or 0 can also be used

Changelog

Version Description

2.3 The variable was added.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->restoreBlockPageBreaks = true;  
$mpdf->WriteHTML('<div class="firstlevel"><div class="secondlevel"><p>Hallo World</p>' );  
  
$mpdf->AddPage();  
  
$mpdf->WriteHTML('<p>Hallo World</p></div></div>' );  
  
?>
```

See Also

- [WriteHTML\(\)](#) - Write HTML code to the document
- [<formfeed>](#) - Equivalent to pagebreak with *restoreBlockPageBreaks* set to **TRUE**

restrictColorSpace

(mPDF >= 5.1)

restrictColorSpace - Specify whether to automatically limit the colorspaces used

Description

```
void PDFAuto
```

Specify whether to automatically limit the colorspaces used when creating PDF/A1-b or PDF/X-1a compliant documents. PDF files can contain objects using different colorSpaces e.g. Grayscale, RGB and CMYK. By default, mPDF creates PDF files using the colours as they are specified: font colour may be set (e.g. #880000) as an RGB colour, and the file may contain JPG images in RGB or CMYK format.

In some circumstances, you may wish to create a PDF file with restricted colorSpaces e.g. printers will often want files which contain only CMYK, spot colours, or grayscale, but not RGB. Using restrictColorSpace will attempt to convert every colour value used in the document to the permitted colorSpace(s). Almost everything including images will be converted (except BMP images), and the conversion of images may take significant time.

Values

restrictColorSpace = 1|2|3

Values

- 1: allow GRAYSCALE only [convert CMYK/RGB->gray]
- 2: allow RGB / SPOT COLOR / Grayscale [convert CMYK->RGB]
- 3: allow CMYK / SPOT COLOR / Grayscale [convert RGB->CMYK]
- 0 or any other value: no restriction is made on colorspace used

DEFAULT: 0

Changelog

Version Description

Version	Description
5.1	Variable was added.

See Also

- [PDF/A1-b compliance](#)
- [PDF/X-1a compliance](#)
- [PDFA - Create PDF/A1-b compliant document](#)
- [PDFX - Create PDF/X-1a compliant document](#)
- [PDFAauto - Specify whether to automatically fix issues to create PDF/A1-b compliant document](#)
- [PDFXauto - Specify whether to automatically fix issues to create PDF/X-1a compliant document](#)
- [ICCProfile - Specify the ICC profile for the chosen colorspace used in the document](#)

setAutoBottomMargin

(mPDF >= 4.0)

Description

mixed **setAutoBottomMargin**

Specify the behaviour defining the bottom-margin of the document. When *setAutoBottomMargin* is set to 'stretch' then *autoMarginPadding* defines the minimum distance in mm that will be forced between the top of the footer and the bottom of the main text.

Values

setAutoBottomMargin

Values

pad - the value for margin-bottom is used to set a fixed distance in mm (padding) between the top of the footer and the bottom of the main text

stretch - margin-bottom sets a **minimum** distance in mm between the bottom of the page and the bottom of the main text, which expands if the footer is too large to fit.

FALSE - the defined value for margin-bottom is respected even if the footer overlaps the main body of the document.

DEFAULT FALSE

See Also

- [Headers & Top margins](#)
- [setAutoTopMargin](#) -Specify mode of determining top-margin position
- [autoMarginPadding](#) - Specify padding between top-margin and header in automatic mode

setAutoTopMargin

(mPDF >= 4.0)

Description

mixed **setAutoTopMargin**

Specify the behaviour defining the top-margin of the document. When *setAutoTopMargin* is set to 'stretch' then *autoMarginPadding* defines the minimum distance in mm that will be forced between the bottom of the header and the top of the main text.

Values

setAutoTopMargin

Values

pad - the value for margin-top is used to set a fixed distance in mm (padding) between the bottom of the header and top of the main text

stretch - margin-top sets a **minimum** distance in mm between the top of the page and the top of the main text, which expands if header is too large to fit.

FALSE - the defined value for margin-top is respected even if the header overlaps the main body of the document.

DEFAULT FALSE

See Also

- [Headers & Top margins](#)
- [setAutoBottomMargin](#) -Specify mode of determining bottom-margin position
- [autoMarginPadding](#) - Specify padding between top-margin and header in automatic mode

showImageErrors

mPDF >= 3.0

Set to **TRUE** for optional error reporting for problems with Images.

DRAFT

showStats

(mPDF >= 4.0)

showStats - Display performance data for the generated PDF file

Value

```
void showStats
```

Specify whether to show performance data. Useful if you are developing with mPDF, and want to test different configurations.

Values

showStats = **TRUE** | **FALSE**

Values

TRUE : enable display of data

FALSE: disabled

DEFAULT: **FALSE**

Changelog

Version Description

4.0 Variable was added.

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF();
$mpdf->showStats = true;
$mpdf->WriteHTML("Hello World");
$mpdf->Output();
?>
```

This will suppress output of the PDF file, and display data on the browser such as:

```
Generated in 0.45 seconds
Compiled in 0.46 seconds (total)
Peak Memory usage 10.25MB
PDF file size 37kB
Number of fonts 6
```

See Also

- [debug](#) - specify whether to show errors
- [showImageErrors](#) - show/hide error reporting for problems with Images
- [allow_output_buffering](#) - prevent error messages when using output buffering

showWatermarkImage

(mPDF >= 2.2)

Description

boolean **showWatermarkImage**

Specifies whether or not to show/print the watermark image on each page. The file for the watermark must be defined using [SetWatermarkImage\(\)](#). The watermark is added to the document at the end of each page.

Values

`showWatermarkImage = TRUE|FALSE`

If `showWatermarkImage` is **TRUE** or 1 the watermark image will be added to each page of the document. The value can be changed during the document to turn the watermark on and off on different pages.

Values

TRUE: Show/print the watermark image

FALSE: DEFAULT Do not show/print the watermark image

Values of 1 or 0 can also be used

Changelog

Version Description

2.2 The variable was added.

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->SetWatermarkImage('images/tiger.jpg');
$mpdf->showWatermarkImage = true;
$mpdf->WriteHTML('<p>Hallo World</p>');

// As showWatermark is true when the first page is finished, a watermark is added
$mpdf->AddPage();

// When the second page is finished the watermark will not appear.
$mpdf->showWatermarkImage = false;
$mpdf->WriteHTML('<p>Hallo World</p>');

?>
```

See Also

- [SetWatermarkText\(\)](#) - Set the text to use as a Watermark
- [SetWatermarkImage\(\)](#) - Set an image to use as a Watermark
- [watermarkImageAlpha](#) - Specifies the transparency (alpha value) for the watermark image
- [watermarkTextAlpha](#) - Specifies the transparency (alpha value) for the watermark text
- [showWatermarkText](#) - Specifies whether or not to show/print the watermark text
- [watermark_font](#) - Specifies the font to use for Watermark text

showWatermarkText

(mPDF >= 2.2)

Description

boolean showWatermarkText

Specifies whether or not to show/print the watermark text on each page. The text for the watermark must be defined using [SetWatermarkText\(\)](#). The watermark is added to the document at the end of each page.

Note: This superceded TopicIsUnvalidated in mPDF 2.2 as an alias, and is now the preferred form.

Values

showWatermarkText = TRUE|FALSE

If *showWatermarkText* is **TRUE** or 1 the watermark text will be added to each page of the document. The value can be changed during the document to turn the watermark on and off on different pages.

Values

TRUE: Show/print the watermark text

FALSE: DEFAULT Do not show/print the watermark text

Values of 1 or 0 can also be used

Changelog

Version Description

2.2 The variable was added.

Examples

Example #1

```
<?php

$mpdf=new mPDF();
$mpdf->SetWatermarkText('DRAFT');
$mpdf->showWatermarkText = true;
$mpdf->WriteHTML('<p>Hallo World</p>');

// As showWatermark is true when the first page is finished, a watermark is added
$mpdf->AddPage();

// When the second page is finished the watermark will not appear.
$mpdf->showWatermarkText = false;
$mpdf->WriteHTML('<p>Hallo World</p>');

?>
```

See Also

- [SetWatermarkText\(\)](#) - Set the text to use as a Watermark
- [SetWatermarkImage\(\)](#) - Set an image to use as a Watermark
- [watermarkImageAlpha](#) - Specifies the transparency (alpha value) for the watermark image
- [watermarkTextAlpha](#) - Specifies the transparency (alpha value) for the watermark text
- [showWatermarkImage](#) - Specifies whether or not to show/print the watermark image
- [watermark_font](#) - Specifies the font to use for Watermark text

SHYlang

(mPDF >= 2.5)

Please see [Hyphenation](#).

DRAFT

simpleTables

(mPDF >= 4.3)

simpleTables - Disables complex table borders etc. to improve performance

Description

```
void simpleTables
```

Specify whether to disable complex table borders etc. to improve performance. The border for all table cells will be the same (although separate values can be used for -top, -left etc.). A separate table border may still be specified (if border-collapse is not used).

This may improve performance considerably for large tables, reducing memory use and increasing processing speed by approximately 30%

Note: Prior to mPDF 4.4 using simpleTables also disabled padding, background-images, background-color and rotated text. These were re-introduced with almost no loss of performance.

Values

simpleTables = **TRUE** | **FALSE**

Values

TRUE : disable complex table borders etc.

FALSE: allow full CSS support for tables

DEFAULT: FALSE

Changelog

Version Description

4.3	Variable was added.
-----	---------------------

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF();
$mpdf->simpleTables = true;
$mpdf->WriteHTML($table);
$mpdf->Output();
?>
```

smCapsScale

mPDF >= 5.0

Control SmallCaps appearance

Default set in config.php

Default value: 0.75; // Factor of 1 to scale capital letters

DRAFT

smCapsStretch

mPDF >= 5.0

Control SmallCaps appearance

Default set in config.php

Default value: 115; // % to stretch small caps horizontally

DRAFT

tableMinSizePriority

(mPDF >= 4.6)

Description

mixed **tableMinSizePriority**

Forces mPDF to respect the value set as autosize when displaying a table. If a table has `page-break-inside:avoid` set but cannot be made to fit onto a page without exceeding the resizing factor set by `autosize:` the value of `tableMinSizePriority` will determine which factor wins out.

Values

`tableMinSizePriority`

Values

`TRUE | FALSE`

`DEFAULT = FALSE`

tabSpaces

(mPDF >= 2.3)

Description

```
int tabSpaces
```

Specifies the number of spaces to substitute for a **TAB** character when parsing HTML input between `<pre>...</pre>` tags. The default value (8) is consistent with the HTML specification, but many programs including Windows NotePad uses a value of 6.

Values

tabSpaces

Values

Integer value greater than 0

DEFAULT: 8

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->tabSpaces = 6;  
$html = file_get_content('NotePad.txt');  
$mpdf->WriteHTML('<pre>' . $html . '</pre>');  
$mpdf->Output();  
  
?>
```

title2annots

title2annots=true (default=false)

Convert the title attribute from all in-line elements to annotations. In-line elements include span, acronym, i, delete, cite, abbr, q, samp, code, var; also picks up from img element, but not block elements such as div, p, h1-6, or form elements, nor any table elements.

See Also

- [Annotation\(\)](#) - Add an Annotation to the document
- [<annotation>](#) - Custom HTML tag - equivalent to **Annotation**

DRAFT

use_kwt

mPDF >= 2.0

Keep with table

Default false

If true: function AdjustHTML() adds an attribute KEEP-WITH-TABLE to any <table> which is immediately preceded by a <h1-6> element.

This attempts to keep the heading and table together on one page.

Can be used together with <table style="page-break-inside: avoid;">

Automatically sets the table to fit on one page (i.e. page-break-inside=avoid) if it is a rotated table.

Ignored if: Heading is longer than one line of text on the page, Columns are on, Keep-block-together active (i.e. page-break-inside=avoid for surrounding BLOCK).

DRAFT

useAdobeCJK

(mPDF >= 5.0)

Description

boolean **useAdobeCJK**

When **TRUE**, forces mPDF to use the free Adobe CJK Asian fonts, thus keeping the PDF file size to a minimum. This affects text defines using the CSS *lang* property, which includes the use of AutoFont. Thus a CSS stylesheet defining `lang="ja"` will be substituted by the Adobe Japanese font. (This will not prevent the use of other CJK fonts if specified by font-family.)

The precise effect it has on different languages/fonts will be specified in the `config_cp.php` configuration file.

Note: This value can only be set inside the `config.php` configuration file. To change the value at runtime, you must use `$mpdf = new mPDF(' -aCJK');` to set as **FALSE** or `$mpdf = new mPDF(' +aCJK');` to set as **TRUE**

Values

`useAdobeCJK = TRUE|FALSE`

Values

TRUE: DEFAULT Use the free Adobe CJK Asian fonts

FALSE: Use normal embedded fonts

useDictionaryLBR

(mPDF >= 6.0)

Description

boolean **useDictionaryLBR**

Specify whether to use mPDF word dictionaries to determine appropriate places for line breaks when using Lao, Thai or Khmer text.

For more information, see [Line breaking](#).

Values

useDictionaryLBR = **FALSE|TRUE**

Values

TRUE: DEFAULT use mPDF word dictionaries to determine appropriate places for line breaks when using Lao, Thai or Khmer text.

FALSE: word dictionaries not used. Line breaks will only be allowed according to usual line-breaking rules.

Changelog

Version Description

6.0 Variable was added.

See Also

- [Line breaking](#)

useFixedNormalLineHeight

(mPDF >= 6.0)

Description

boolean **useFixedNormalLineHeight**

Specify whether to use a fixed value for the line-height of text, when CSS property line-height is set to `normal` (default).

For more information, see [Line-height](#).

Values

`useFixedNormalLineHeight = FALSE|TRUE`

Values

FALSE: **DEFAULT** use the font metrics to determine the line-height of text, when CSS property line-height is set to `normal` (default).

TRUE: use a fixed value for the line-height of text, when CSS property line-height is set to `normal` (default).

Changelog

Version	Description
6.0	Variable was added.

See Also

- [Line-height](#)

useFixedTextBaseline

(mPDF >= 6.0)

Description

boolean **useFixedTextBaseline**

Specify whether to use a fixed value to set the position of the text baseline.

For more information, see [Line-height](#).

Values

`useFixedTextBaseline= FALSE|TRUE`

Values

FALSE: **DEFAULT** use the font metrics to set the position of the text baseline.

TRUE: use a fixed value to set the position of the text baseline.

Changelog

Version	Description
6.0	Variable was added.

See Also

[Line-height](#)

useGraphs

(mPDF >= 2.4)

Description

boolean **useGraphs**

When **TRUE**, mPDF will parse table data and allow the use of [<jpgraph>](#) to generate graphs from the data. This should only be set to **TRUE** when required to conserve memory and processing time.

Note: Graphs require [JPGraph](#) to be installed on the server. See [Graphs](#) for further information.

Values

`useGraphs = TRUE|FALSE`

Values

TRUE: Parse table data from the HTML, and allow the use of [<jpgraph>](#)

FALSE: **DEFAULT**

Examples

See [<jpgraph>](#) for example

See Also

- [<jpgraph>](#) - Write HTML code to document
- [Graphs](#) - More about JPGraph and graphs

useKerning

(mPDF >= 5.4)

Description

boolean **useKerning**

When **TRUE**, mPDF will support the CSS style *font-kerning*. This should only be set to **TRUE** when required to conserve memory and processing time.

Values

useKerning = **TRUE|FALSE**

Values

TRUE: Enable support for CSS style *font-kerning*

FALSE: **DEFAULT**

See Also

- Kerning

useLang

(mPDF >= 2.3 <= 5.7)

useLang - Specify whether to recognise/support the HTML attribute *lang*

Description

```
void useLang
```

Specify whether to recognise/support the HTML attribute *lang*.

See [lang](#) for more details.

Note: This variable was removed in mPDF 6.0 [autoScriptToLang](#) should be used for the same effect.

Note: The default value was changed to **TRUE** in version 4.0

Note: *lang* is a useful way to select appropriate fonts for some languages. Automatic font selection using [SetAutoFont\(\)](#) marks up the HTML with the *lang* attribute, so useLang is required. Using automatic font selection adds considerable processing time when creating a large document. Automatic font selection is only valid when using UTF-8 as the codepage for the document.

Values

`useLang = TRUE | FALSE`

Values

TRUE : recognise/support the HTML attribute *lang*.

FALSE: does not recognise/support the HTML attribute *lang*.

DEFAULT: TRUE

Changelog

Version Description

2.3 Variable was added.

4.0 Default value changed to **TRUE**

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF('utf-8');

$html =
<p>Start with some English text</p>
<p lang="ar">
هل ستسفر الجهود الدبلوماسية الجارية عن حلول؟ وكيف تنظر للاتهامات لبعض هذه الدول بالتدخل في الشأن العراقي، والتورط في دعم عمليات
</p><!-- العنف؟ والى اي مدى يبدو الوضع في العراق انعكasa للصراعات الإقليمية في المنطقة؟
<p>And again in English</p>
';

$mpdf->useLang = true;

$mpdf->WriteHTML($html);

$mpdf->Output();
```

```
?>
```

See Also

- [SetAutoFont\(\)](#) - Use AutoFont to auto-detect text language in HTML input
- [autoFontGroupSize](#) - Specify the text chunk size to group when autodetecting text language
- [disableMultilingualJustify](#) - Specify whether to disable text justification in multilingual documents
- [lang](#) - Information on mPDF support for the HTML attribute lang

useSubstitutions

(mPDF >= 4.2)

useSubstitutions – Specify whether to substitute missing characters in UTF-8 (multibyte) documents

Description

```
void useSubstitutions
```

Specify whether to substitute missing characters in UTF-8(multibyte) documents. Characters which cannot be displayed in the current set font, will be substituted by characters in the Adobe core fonts (Symbol, Zapfdingbats etc.), or the backup font(s) specified in the variable in `$this->backupSubsFont` the config_fonts.php configuration file.

Note: Prior to mPDF 5.0 useSubstitutions controlled the behaviour of character substitution when using Adobe core fonts, and useSubstitutionsMB was used for character susbstition in multibyte/utf-8 documents. For mPDF >= 5.0 character substitution for documents using core fonts is always enabled.

Values

`useSubstitutions = TRUE | FALSE`

Values

TRUE : enable substitution

FALSE: disable substitution

DEFAULT: FALSE

Changelog

Version Description

4.2	Variable was added. Controlled behaviour when using Adobe core fonts
5.0	Changed to control behaviour of multibyte documents

Examples

Example #1

```
<?php
include("../mpdf.php");
$mpdf=new mPDF('UTF-8');
$mpdf->useSubstitutions = true;
$mpdf->WriteHTML("Hallo World");
$mpdf->Output();
?>
```

Note: This may add significantly to the processing time for large files.

useTibetanLBR

(mPDF >= 6.0)

Description

boolean **useTibetanLBR**

Specify whether to use mPDF algorithm to determine appropriate places for line breaks when using Tibetan text.

mPDF uses a simple algorithm to identify line-breaking opportunities after the characters U+0F0B (Tsheg) or U+0F0D.

For more information, see [Line breaking](#).

Values

*useTibetanLBR = **FALSE|TRUE***

Values

TRUE: **DEFAULT** use mPDF algorithm to determine appropriate places for line breaks when using Tibetan text.

FALSE: algorithm not used. Line breaks will only be allowed according to usual line-breaking rules.

Changelog

Version	Description
6.0	Variable was added.

See Also

[Line breaking](#)

watermark_font

(mPDF >= 1.0)

Description

```
string watermark_font
```

Specifies the font to use for the watermark on each page. The **BOLD** style of the font is used by default.

Values

watermark_font

Define as any of the available font-families.

DEFAULT or **BLANK** uses the default font-family for the document.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetWatermarkText('DRAFT');  
$mpdf->showWatermarkText = true;  
$mpdf->watermark_font = 'DejaVuSansCondensed';  
$mpdf->WriteHTML('<p>Hallo World</p>');  
  
?>
```

See Also

- [SetWatermarkText\(\)](#) - Set the text to use as a Watermark
- [SetWatermarkImage\(\)](#) - Set an image to use as a Watermark
- [watermarkImageAlpha](#) - Specifies the transparency (alpha value) for the watermark image
- [watermarkTextAlpha](#) - Specifies the transparency (alpha value) for the watermark text
- [showWatermarkText](#) - Specifies whether or not to show/print the watermark text
- [showWatermarkImage](#) - Specifies whether or not to show/print the watermark image

watermarkImageAlpha

(mPDF >= 2.2)

Description

```
string watermarkTextAlpha
```

Specifies the transparency (alpha value) to use for the watermark text on each page.

Values

watermarkTextAlpha

Define as a value between 0 and 1.

DEFAULT or **BLANK** : 0.2

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetWatermarkImage('images/logo.jpg');  
$mpdf->showWatermarkImage = true;  
$mpdf->watermarkImageAlpha = 0.1;  
$mpdf->WriteHTML('<p>Hallo World</p>');  
  
?>
```

See Also

- [SetWatermarkText\(\)](#) - Set the text to use as a Watermark
- [SetWatermarkImage\(\)](#) - Set an image to use as a Watermark
- [watermarkTextAlpha](#) - Specifies the transparency (alpha value) for the watermark text
- [showWatermarkText](#) - Specifies whether or not to show/print the watermark text
- [showWatermarkImage](#) - Specifies whether or not to show/print the watermark image
- [watermark_font](#) - Specifies the font to use for Watermark text

watermarkImgAlphaBlend

(mPDF >= 4.5)

watermarkImgAlphaBlend - Specify the blend mode for overlying watermark images

Description

```
void watermarkImgAlphaBlend
```

Specify the blend mode for overlying watermark images. Different blend modes work with different types of images. The blend modes are those specified in the [PDF reference](#). The value of Normal gives acceptable results for most cases. 'Multiply' may produce better results with overlying WMF or SVG images.

Note: This variable can be changed either in the configuration file config.php or at runtime

Values

watermarkImgAlphaBlend

Values

Normal
Multiply
Screen
Overlay
Darken
Lighten
ColorDodge
ColorBurn
HardLight
SoftLight
Difference
Exclusion
DEFAULT: Normal

Changelog

Version Description

Version	Description
4.5	Variable was added.

See Also

- [watermarkImgBehind](#) - Specify whether to place watermark images behind page contents

watermarkImgBehind

(mPDF >= 4.4)

watermarkImgBehind – Specify whether to place watermark images behind page contents

Description

```
void watermarkImgBehind
```

Specify whether to place watermark images behind page contents

Note: This variable can be changed either in the configuration file config.php or at runtime

Note: This variable was unintentionally set to **TRUE** in the config.php file released with mPDF 4.4 In version 4.5 it was changed to **FALSE**

Values

watermarkImgBehind = **TRUE** | **FALSE**

Values

TRUE : place watermark images behind page contents

FALSE: place watermark images in front of all page contents

DEFAULT: FALSE

Changelog

Version Description

4.4 Variable was added.

4.5 Default value changed to **FALSE**

See Also

- [watermarkImgAlphaBlend](#) - Specify the blend mode for overlaid watermark images

watermarkTextAlpha

(mPDF >= 2.2)

Description

```
string watermarkTextAlpha
```

Specifies the transparency (alpha value) to use for the watermark text on each page.

Values

watermarkTextAlpha

Define as a value between 0 and 1.

DEFAULT or **BLANK** : 0.2

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
$mpdf->SetWatermarkText('DRAFT');  
$mpdf->showWatermarkText = true;  
$mpdf->watermarkTextAlpha = 0.1;  
$mpdf->WriteHTML('<p>Hallo World</p>');  
  
?>
```

See Also

- [SetWatermarkText\(\)](#) - Set the text to use as a Watermark
- [SetWatermarkImage\(\)](#) - Set an image to use as a Watermark
- [watermarkImageAlpha](#) - Specifies the transparency (alpha value) for the watermark image
- [showWatermarkText](#) - Specifies whether or not to show/print the watermark text
- [showWatermarkImage](#) - Specifies whether or not to show/print the watermark image
- [watermark_font](#) - Specifies the font to use for Watermark text

MPDF UTILITIES

strcode2utf()

(htmltoolkit >=1.0)

strcode2utf - Convert HTML numerical entities to UTF-8 encoded string

Description

```
string strcode2utf ( string $text [, boolean $low ])
```

Returns a UTF-8 encoded string.

Note: This function is not part of the mPDF class, and is located in the htmltoolkit.php file. It is called without the class prefix e.g. \$mpd->

Note: **strcode2utf** is useful for preparing text to be used as a Watermark, or for the metadata properties of Title, Author, Creator and Keywords - which require UTF-8 encoded strings with no entities.

Parameters

text

The input string, containing HTML numerical entities e.g. Ά or •

low

Specifies whether to convert HTML numerical entities of ASCII characters (<128).

DEFAULT: TRUE

Values

TRUE: Convert all HTML numerical entities to UTF-8 characters

FALSE: Only convert characters above codepoint 127

Note: This function does not convert named character entities like & " or »

Return value

Returns a UTF-8 encoded string.

Examples

Example #1

```
<?php  
  
$mpdf=new mPDF();  
  
$wm = strcode2utf("ايلات &#1601;يما  
ايلات &#1601;يما");  
  
$mpdf->SetWatermarkText($wm);  
$mpdf->showWatermark = true;  
$mpdf->WriteHTML('<p>Hallo World</p>');  
  
?>
```

See Also

- [SetWatermarkText\(\)](#) - Sets the text to use for a Watermark
- [SetTitle\(\)](#) - Set document Title in metadata
- [SetAuthor\(\)](#) - Set document Author in metadata
- [SetCreator\(\)](#) - Set document Creator in metadata
- [SetSubject\(\)](#) - Set document Subject in metadata
- [SetKeywords\(\)](#) - Set document Keywords in metadata

preparePreText()

(html toolkit >= 2.4)

PreparePreText - Prepares text to be output ignoring the HTML markup

Description

```
string preparePreText ( string $text [, string $formfeed ])
```

Prepares text to be output ignoring the HTML markup. This is useful to output a large text file (e.g. a PHP script file) using [WriteHTML\(\)](#). This will surround the text with `<pre>` tags whilst preventing `<pre>` tags included in the text from being parsed. It also allows use of a text string marker (*formfeed*) to be replaced by a formfeed in the output file.

Note: Prior to mPDF 5.1 you should use the *\$mode* parameter of [WriteHTML\(\)](#) as '2' to avoid parsing the text for style tags. See example below.

Parameters

text

This parameter specifies the text string to prepare.

formfeed

formfeed specifies the string to be replaced by a formfeed in the output file.

DEFAULT: "FF//"

Return value

Returns a text string.

Changelog

Version	Description
2.4	Function was added.

Examples

Example #1

```
<?php
include("../mpdf.php");

$text = file_get_contents('scriptfile.php');
$text = preparePreText($text);

// Default spaces/tab in mPDF is 8 as specified by HTML spec.
// Notepad and other text editors use a value of 6
$mpdf->tabSpaces = 6;

// If 'scriptfile.php' is iso-8859-1 or win-1252 encoded, use
$mpdf->allow_charset_conversion=true;
$mpdf->charset_in='windows-1252';

$mpdf->WriteHTML($text, 2);

$mpdf->Output();
```

```
exit;  
?>
```

See Also

- [WriteHTML\(\)](#) - Write HTML code to a PDF file
- `tabSpaces` - Specifies the number of spaces to substitute for a **TAB** character
- `allow_charset_conversion` - Parse the character set of any input text from the HTML, or allow setting of the value `charset_in`
- `charset_in` - Define the character encoding of any input HTML

CODEPAGES & GLYPHS

Win-1252

Windows 1252

(CP1252)

Latin / Western European

Western European languages including English, Italian, Spanish, Portuguese, French, German, Dutch, Danish, Swedish, Norwegian, and Icelandic, which use the Latin alphabet.

! " # \$ % & ' () * + , - . /
 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
 @ A B C D E F G H I J K L M N O
 P Q R S T U V W X Y Z [\] ^ _
 ` a b c d e f g h i j k l m n o
 p q r s t u v w x y z { | } ~
 i ¢ £ ¤ ¥ ¡ § ¨ © ª « ¬ ® ¯ °
 ± ² ³ ´ μ ¶ . , ¹ º » » ¼ ½ ¾ Ł Á
 Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï Ð
 Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý þ ß à
 á â ã ä å æ ç è é ê ë ì í î ï ð
 ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ œ
 œ Š š Ý Ž ž f ^ ~ - — ‘ ’ „ “ ”
 „ † ‡ • … ‰ < > € ™

Note: windows-1252 (win-1252, cp1252) contains all the characters in ISO-8859-1 plus additional ones

ASCII characters

ASCII

```
! " # $ % & ' ( ) * + , - . /  
0 1 2 3 4 5 6 7 8 9 : ; < = > ?  
@ A B C D E F G H I J K L M N O  
P Q R S T U V W X Y Z [ \ ] ^ _  
' a b c d e f g h i j k l m n o  
p q r s t u v w x y z { | } ~
```

Win-1251

Windows 1251

(CP1251)

Cyrillic

Languages that use the Cyrillic alphabet such as Russian, Bulgarian, Serbian, Macedonian and Bulgarian.

Generally considered to give better coverage than ISO-8859-5.

```
_ ! " # $ % & ' ( ) * + , - . /  
0 1 2 3 4 5 6 7 8 9 : ; < = > ?  
@ A B C D E F G H I J K L M N O  
P Q R S T U V W X Y Z [ \ ] ^ _  
` a b c d e f g h i j k l m n o  
p q r s t u v w x y z { | } ~  
Ђ Ѓ „ … т ъ € %о Ђ Ѓ Ђ Ѓ  
Ђ ‘ ’ “ ” • — ™ Ђ Ѓ Ђ Ѓ  
Ў ѕ Ј ѕ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ  
Ѡ ± І І Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ Ѓ  
Ѡ Б В Г Д Е Ж З И Й К Л М Н О П  
Ѡ С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я  
Ѡ б в г д е ж з и й к л м н о п  
Ѡ с т у ф х ц ч ш щ є ў ѿ є ѿ ѿ
```

ISO-8859-2

ISO-8859-2

Latin-2

Central European

Supports Central and Eastern European languages that use the Latin alphabet, including Bosnian, Polish, Croatian, Czech, Slovak, Slovenian, Serbian, and Hungarian.

! " # \$ % & ' () * + , - . /
0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z [\] ^ _
` a b c d e f g h i j k l m n o
p q r s t u v w x y z { | } ~
¤ § “ ” ° ‘ ’ , Á Â Ä Ç É Ë Í Î Ó
Ô Ö × Ú Ü Ý ß á â ä ç é ë í î ó
ô ö ÷ ú ü ý Ä å Á ç É Ë Í Î Ó
Đ đ Ë è Ê ē Í Í Ł Ł Ł Ł N ñ Ñ ñ
Őő Óó Úú Ýý Íí Ł Ł Ł Ł N ñ Ñ ñ
Őő Óó Úú Ýý Íí Ł Ł Ł Ł N ñ Ñ ñ

ISO-8859-4

ISO-8859-4

Latin-4

North European

Estonian, Latvian, Lithuanian, Greenlandic, and Sami.

! " # \$ % & ' () * + , - . /
0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z [\] ^ _
` a b c d e f g h i j k l m n o
p q r s t u v w x y z { | } ~
¤ § “ ” - ° ‘ ’ , Á Â Ã Ä Å Æ É Ë
í ï Ô Õ Ö × Ø Ú Û Ü ß á â ä å å
æ é ë í î ô õ ö ÷ ø ú û ü Å á ä Å
ä Ç č Đ đ Ė ê È è Ë ë G ñ Í ï Í
í î ï K ï k L ! N ñ ï ï Õ ò R ð
š š T ð Ü ù Ù ù U ù U ù Ž ž

ISO-8859-7

ISO-8859-7

Latin/Greek

Covers the modern Greek language (monotonic orthography). Can also be used for Ancient Greek written without accents or in monotonic orthography, but lacks the diacritics for polytonic orthography.

```
! " # $ % & ' ( ) * + , - . /  
0 1 2 3 4 5 6 7 8 9 : ; < = > ?  
@ A B C D E F G H I J K L M N O  
P Q R S T U V W X Y Z [ \ ] ^ _  
' a b c d e f g h i j k l m n o  
p q r s t u v w x y z { | } ~  
' ' € € ™ | § “ © „ “ —  
° ± ² ³ ‘ ’ Α · Ε Ή 1 » Ο ½ Υ Ω  
Ϊ Α Β Γ Δ Ε Ζ Η Θ Ι Κ Λ Μ Ν Ξ Ο  
Π Ρ Σ Τ Υ Φ Χ Ψ Ω ī Ÿ á é ñ í  
Ӧ α β γ δ ε ڇ ڻ θ ڻ κ λ μ ν ڻ ڻ  
ڻ ρ ڻ σ ڻ τ ڻ ϕ ڻ χ ڻ ψ ڻ ω ڻ ü ڻ ó ڻ ú ڻ ó
```

ISO-8859-9

ISO-8859-9

Latin-5

Turkish

Largely the same as ISO-8859-1, replacing the rarely used Icelandic letters with Turkish ones. It is also used for Kurdish.

```
! " # $ % & ' ( ) * + , - . /  
0 1 2 3 4 5 6 7 8 9 : ; < = > ?  
@ A B C D E F G H I J K L M N O  
P Q R S T U V W X Y Z [ \ ] ^ _  
' a b c d e f g h i j k l m n o  
p q r s t u v w x y z { | } ~  
i ¢ f ¤ ¥ ¡ § “ © ª « ¬ ® ¯ °  
± ² ³ ´ μ ¶ · ¸ ¹ º » ¼ ½ ¾ Ł À  
Á Â Ã Ä Å Æ Ç È É Ë Ì Í Î Ï Ñ  
Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü ß à á â ã  
ä å æ ç è é ê ë ì í î ï ñ ò ó ô  
õ ö ÷ ø ù ú û ü ÿ Ĝ ġ ì í Š ſ
```

ISO-8859/win comparison chart

ISO-8859 codepages

Dec	Hex	1	2	3	4	5	6	7	8	9	10	11	13	14	15	16
160 A0		Non-breaking space (NBSP)														
161 A1	i Ä H Ä Ë	'		i Ä n "	ß	i Ä										
162 A2	¢ ^ ^ K Į	'	¢	¢	Ē	ゅ	¢	b	¢	a						
163 A3	£ Ł £ R Ģ	£	£	£	G	ゅ	£	£	£	£						
164 A4	¤ ¤ ¤ ¤ € ¤	¤	€	¤	¤	ī	¤	¤	¤	¤	€	€				
165 A5	¥ Ł Ī S Dp	¥	¥	¥	ī	ſ	"	č	¥	"						
166 A6	ı Ś H L ı	ı	ı	ı	K	ň	ı	D	Ś	Ś						
167 A7	§ § § § Ī	§	§	§	§	ג	§	§	§	§						
168 A8	" " " J	"	"	"	L	ž	Ø	W	š	š						
169 A9	© Š İ Ś Ľ	©	©	©	Đ	њ	©	©	©	©						
170 AA	¤ Š Š Ě H	.	x	¤	Š	ž	R	W	¤	Š						
171 AB	« Č Ě G H	«	«	«	F	ž	«	d	«	«						
172 AC	¬ Ž Ĵ F K ,	¬	¬	¬	Z	ž	¬	Y	¬	Ž						
173 AD	soft hyphen (SHY)														SHY	
174 AE	® Ž Ž Ÿ	®	®	®	Ū	݂	®	®	®	®	ž					
175 AF	‐ Ž Ž ‐ Ŭ	‐	‐	‐	ῃ	݂	AE	Ÿ	‐	Ž						
176 B0	° ° ° ° A	°	°	°	°	݂	°	F	°	°						
177 B1	± ą į ą B	±	±	±	ą	ନ	±	ଫ	±							
178 B2	² ² ² ² B	²	²	²	²	ē	݂	²	Ğ	²	Č					
179 B3	³ ³ ³ ³ Г	³	³	³	³	ǵ	ଣ	³	g	³	ଳ					
180 B4	‘ ‘ ‘ ‘ Д	‘	‘	‘	‘	ī	ର	“	M	ž						
181 B5	μ ĩ μ ĩ E	μ	μ	μ	ĩ	ର	μ	ମ	μ	”						
182 B6	¶ ſ ĥ ¡ Ж	¶	ſ	ĥ	¡	Ж	¶	¶	¶	¶	¶					
183 B7	· · · · З	·	·	·	·	ନ	·	P	·	·						
184 B8	„ „ „ „ И	„	„	„	„	ର	„	Ø	W	ž	ž					
185 B9	¹ Š I Š Й	¹	Š	I	Š	Й	¹	¹	đ	ନ	¹	p	¹	č		
186 BA	º ſ ē K	º	ſ	ē	K	ନ	º	š	u	ର	w	º	ſ			
187 BB	» ū ğ ġ L :	»	»	»	»	ශ	ශ	ශ	ශ	ශ	ශ	»	»	»		
188 BC	¼ ž ĵ ĭ M	¼	ž	ć	đ	M	¼	¼	ž	ନ	¼	ỳ	Œ	Œ		
189 BD	½ “ ½ ౏ H	½	“	½	ଏ	H	½	½	—	ଏ	½	ଶ	œ	œ		
190 BE	¾ ž ž O	¾	ž	ž	O	ନ	¾	ū	ନ	¾	ଶ	ଯ	ଯ	ଯ		
191 BF	ż ż ḷ П ئ	ż	ż	ż	پ	ئ	ż	ż	ڻ	ڻ	ڻ	ڻ	ڻ	ڻ		
192 C0	À Á Ñ Ä P ī	À	Á	Á	Ñ	Ä	ନ	ା	ା	ା	ା	ା	ା	ା		
193 C1	Á Á Á Á C ୧ A	ା	ା	ା	ା	C	୧	A	ା	ନ	ି	ା	ା	ା		
194 C2	Â Â Â Â T ī B	Â	Â	Â	Â	T	ି	B	Â	Â	ନ	ା	ା	ା		
195 C3	Ã Ã Ã Y ī G	Ã	Ã	Ã	Y	ି	G	Ã	Ã	ଜ	ଚ	Ã	Ã	Ã		
196 C4	Ä Ä Ä Ä F ଡ Δ	Ä	Ä	Ä	Ä	F	ଡ	Δ	Ä	Ä	ନ	ା	ା	ା		
197 C5	Å Ł Ć Å X ! E	Å	Ł	Ć	Å	X	!	E	Å	Å	ନ	ା	ା	ା	ଚ	
198 C6	Æ Č Ć Æ ҃ Z	Æ	Č	Ć	Æ	҃	Z	Æ	Æ	ନ	ଏ	Æ	Æ	Æ		
199 C7	Ҫ Ҫ Ҫ ڶ H	Ҫ	Ҫ	Ҫ	ڶ	H	ڶ	ڶ	Ӗ	Ҫ	Ҫ	Ҫ	Ҫ	Ҫ		
200 C8	Ӗ Ĉ Ĕ Ĕ Ҫ ଶ ପ ଥ	Ӗ	Ҫ	߇	߇	߇	߇	߇	߇	߇	߇	߇	߇	߇		
201 C9	܇ ܇ ܇ ܇ ܇ ܇ ܇ ܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇		
202 CA	܇ ܇ ܇ ܇ ܇ ܇ ܇ ܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇		
203 CB	܇ ܇ ܇ ܇ ܇ ܇ ܇ ܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇	܇		

204 CC	ì Ě ì É ь җ M	ì É ҆ G ì ì ì
205 CD	í í í í є ڦ N	í í ۽ ڪ í í í
206 CE	î î î î ў ڙ ڦ	î î ۽ ڦ î î î
207 CF	ї ð ï î я ڏ O	ї ï ۽ ڦ ڻ î î î
208 D0	đ Đ Đ a ڏ P	ڇ Đ ڙ Š ڻ Đ Đ
209 D1	ñ ñ ñ ñ ٻ ڦ P	ڻ ñ ڙ ñ ñ ñ ñ
210 D2	ò ñ ò õ в ڙ	ò õ ڳ ñ ò ò ò
211 D3	ó ó ó ڪ گ س ڦ	ó ó ڦ ó ó ó ó
212 D4	ô ô ô ô د ڦ T	ô ô ڦ ô ô ô ô
213 D5	ő ô ڳ ڳ e ص Y	ő ô ڦ ڦ ڦ ڦ ڦ
214 D6	ö ö ö ö ج ڦ ف	ö ö ڦ ö ö ö ö
215 D7	× × × × ز ڦ X	× ù ڦ × ٿ × ſ
216 D8	ø ř ڳ ø ڦ ڦ	ø ø , ڦ ø ø ū
217 D9	ù ú ù ú ڦ ڦ	ù ú ڦ ڦ ڦ ڦ ڦ
218 DA	ú ú ú ú ک ڦ	ú ú . ڦ ú ú ú
219 DB	û ú ú ú ل ڦ	û ú ڦ ڦ ڦ ڦ
220 DC	ü ü ü ü م ڦ	ü ü ڦ ڦ ڦ ڦ
221 DD	ý ý ü ü ن ڦ	ڦ ý ž ý ý ڦ
222 DE	پ ٿ ڪ ڻ ڦ o	ڦ پ ڦ ڦ ڦ ڦ
223 DF	ڦ ڦ ڦ ڦ ڦ ڦ	ڦ ڦ ڦ ڦ ڦ ڦ
224 E0	à á à á p - ú	à á à á l a à à
225 E1	á á á á c ڦ	ڦ á á ڦ l á á á
226 E2	â â â â ت ڦ	ڦ â â â â ڦ â â â
227 E3	ã ä ä ä y ڦ	ڦ ä ä ä ڦ ڦ ä ä ä
228 E4	ä ä ä ä ф J	ڦ ä ä ä ڦ ڦ ä ä ä
229 E5	å í ç å x ڦ	ڦ å å å ڦ ڦ å å å
230 E6	æ á á æ ڦ ڦ	ڦ æ æ æ ڦ ڦ æ æ æ
231 E7	ç ç ç i ڦ ڦ	ڦ ç ç ڦ ڦ è è è
232 E8	è č è č ش ڦ	ڦ è č ڦ ڦ è è è
233 E9	é é é é Ѣ ڦ	ڦ é é ڦ ڦ é é é
234 EA	ê ê ê ê ڦ ڦ	ڦ ê ê ê ڦ ڦ ê ê ê
235 EB	ë ë ë ë ڦ ڦ	ڦ ë ë ë ڦ ڦ ë ë ë
236 EC	ì ē ì è ڦ ڦ	ڦ ì è ڦ ڦ ڦ ڦ
237 ED	í í í í є ڦ	ڦ í í ڦ ڦ ڦ ڦ
238 EE	î î î î ў ڦ	ڦ î î ڦ ڦ ڦ ڦ
239 EF	ї ð ï î я ڦ	ڦ ð ï î ڦ ڦ ڦ ڦ
240 F0	ð đ đ ð ڦ	ڦ ð ڦ ڦ ڦ ڦ ڦ
241 F1	ñ ñ ñ ñ є ڦ	ڦ ñ ñ ڦ ڦ ڦ ڦ
242 F2	ò ñ ò õ ڦ	ڦ ñ ڦ ڦ ڦ ڦ ڦ
243 F3	ó ó ó ڪ ڦ	ڦ ó ó ڦ ڦ ó ó ó
244 F4	ô ô ô ô ڦ	ڦ ô ô ڦ ڦ ô ô ô
245 F5	ő ڳ ڳ s ڦ	ڦ ڳ ڦ ڦ ڦ ڦ
246 F6	ö ö ö ö ڦ	ڦ ö ö ڦ ڦ ö ö ö
247 F7	÷ ÷ ÷ ÷ ڦ	ڦ ÷ ڦ ڦ ڦ ڦ ڦ
248 F8	ø ř ڳ ڦ j	ڦ ø ڦ ڦ ڦ ڦ ڦ
249 F9	ù ú ù ڦ ڦ	ڦ ú ڦ ڦ ڦ ڦ ڦ
250 FA	ú ú ú ú ڦ	ڦ ú ڦ ڦ ڦ ڦ ڦ
251 FB	û ú ú ú ڦ	ڦ ú ڦ ڦ ڦ ڦ ڦ
252 FC	ü ü ü ü ڦ	ڦ ü ü ڦ ڦ ü ü ü

253 FD	ý	ý	ü	ü	§	ú	LRM	ı	ý	ž	ý	ý	ę	
254 FE	þ	þ	ş	ş	ü	ý	ó	RLM	ş	þ	ž	ý	þ	þ
255 FF	ÿ	ÿ	·	·	·	þ	ÿ	к	’	ÿ	ÿ	ÿ	ÿ	

Additional characters in win-1252

8-	€	,	f	"	...	†	‡	^	%o	Š	<	Œ	Ž	
	20AC	201A	0192	201E	2026	2020	2021	02C6	2030	0160	2039	0152	017D	
128	130	131	132	133	134	135	136	137	138	139	140	142		
	'	'	"	"	•	-	-	~	™	š	>	œ	ž	ÿ
9-		2018	2019	201C	201D	2022	2013	2014	02DC	2122	0161	203A	0153	017E 0178
	145	146	147	148	149	150	151	152	153	154	155	156	158	159

Win-1251

8-	Ђ	Ѓ	,	Ѓ	"	...	†	‡	€	%o	Љ	<	Њ	Ќ	Ћ	Џ	
	0402	0403	201A	0453	201E	2026	2020	2021	20AC	2030	0409	2039	040A	040C	040B	040F	
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143		
	ђ	'	'	"	"	•	-	-	™	љ	>	њ	ќ	ћ	џ	Џ	
9-		0452	2018	2019	201C	201D	2022	2013	2014		2122	0459	203A	045A	045C	045B	045F
	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	
	NBSP	Ӧ	Ӧ	J	Ӧ	Ӧ	Ӧ	Ӧ	Ӧ	Ӧ	Ӧ	Ӧ	Ӧ	Ӧ	Ӧ	Ӧ	
A-	00A0	0040	040E	045E	0408	00A4	0490	00A6	00A7	0401	00A9	0404	00AB	00AC	00AD	00AE	0407
	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	
	°	±	I	i	ѓ	μ	¶	·	ë	№	€	»	«	¬	SHY	®	
B-	00B0	00B1	0406	0456	0491	00B5	00B6	00B7	0451	2116	0454	00BB	0458	0405	0455	0457	
	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	
	А	Б	В	Г	Д	Е	Ж	З	И	҃	К	Л	М	Н	О	П	
C-	0410	0411	0412	0413	0414	0415	0416	0417	0418	0419	041A	041B	041C	041D	041E	041F	
	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	
	Р	С	Т	Ү	Ф	Х	Ц	Ч	Ш	Щ	҂	Ы	҃	҄	҅	҆	
D-	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	042A	042B	042C	042D	042E	042F	
	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	
	а	б	в	г	д	е	ж	з	и	҃	к	л	м	н	о	п	
E-	0430	0431	0432	0433	0434	0435	0436	0437	0438	0439	043A	043B	043C	043D	043E	043F	
	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	
	р	с	т	ү	ф	х	ц	ч	ш	щ	҂	ы	҃	҄	҅	҆	
F-	0440	0441	0442	0443	0444	0445	0446	0447	0448	0449	044A	044B	044C	044D	044E	044F	
	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	

See Also

UNICODE Full list characters

<http://www.unicode.org/Public/UNIDATA/UnicodeData.txt>

Mappings - sources

<http://www-950.ibm.com/software/globalization/icu/demo/converters?s=ALL>

<http://www.unicode.org/Public/MAPPINGS/VENDORS/>

ZapfDingbats (Adobe)

Zapf Dingbats

A comprehensive grid of black icons and symbols, likely from a font set. The symbols include: directional arrows (up, down, left, right, diagonal), numbers (1-10), geometric shapes (diamond, circle, square, triangle, plus, minus, asterisk, percent), religious and cultural symbols (crosses, Star of David, etc.), floral patterns, musical notes, and various abstract and functional icons (key, envelope, telephone, etc.).

Symbols (Adobe)

Symbols

! # % & () * + , . / 0 1 2 3
4 5 6 7 8 9 : ; < = > ? [] _ {
| } © ¬ ® ° ± µ × ÷ f A B Γ Δ E
Ζ Η Θ Ι Κ Λ Μ Ν Ξ Ο Π Ρ Σ Τ Υ Φ
Χ Ψ Ω α β γ δ ε ζ η θ ι κ λ μ ν
ξ ο π ρ ξ σ τ υ φ χ ψ ω θ γ φ ϖ
• ... ' " / Ι β Ι ™ Ω Ι ← ↑ → ↓ ↔
↔ ← ↑ → ↓ ↔ ∀ ∃ ∅ Δ ∇ ∈ ∉ ∃ Π
Σ − / * √ ∞ ∞ ∙ ∧ ∨ ∏ ∏ ∫ ∵ ~ ≈
≈ ≠ ≡ ≤ ≥ ⊂ ⊄ ⊆ ⊇ ⊕ ⊗ ⊥ ∙ ∫ ∬
< > ♠ ♣ ♥ ♦

Vietnamese

The Vietnamese alphabets are listed in several noncontiguous Unicode ranges: Basic Latin {U+0000..U+007F}, Latin-1 Supplement {U+0080..U+00FF}, Latin Extended-A, -B {U+0100..U+024F}, Latin Extended Additional {U+1E00..U+1EFF}, and Combining Diacritical Marks {U+0300.. U+036F}.

Latin-1 Supplement

ÀÁÂÃÈÉÊÌÍÒÓÔÕÙÚÝ

à á â ã è é ê ì í ò ó ô õ ù ú ý

Latin Extended-A

Ăă Đđ Īī Ūū

The following letters are specific to Vietnamese:

Latin Extended-B {U+01A0,01A1,U+01AF,01B0}

σ σ Ο ο

Latin Extended Additional {U+1EA0 - U+1EF1}

ꝑ Ꝓ ꝓ Ꝕ ꝕ Ꝗ ꝗ Ꝙ ꝙ Ꝛ ꝛ Ꝝ Ꝟ ꝟ Ꝡ Ꝡ Ꝡ

Ү ү Ү ү Ү ү Ү ү Ү ү Ү ү Ү ү

The Vietnamese *d*̣*ong* currency symbol is ₫ (U+20AB).

BIG-5 (Traditional Chinese)

辰弛芋朕儂的菟箇督楠能賠篾蕕錢賦憤返方冒妹耗毛諭幼絡隆麟廊亂侖傲菁勁叟哄單嚮埃夥娶孺岫巒廬狃挽慷懿抻捩斃量枅梃櫟櫟櫻毋沱淞
 達地瀦直停滴登答特南納買罰磬琵苗負墳編放棒埋妙孟癒傭洛粒鱗妻乘來傴胄劫雙咬喘噬姪壽嫵學忿巍廩忤悖慳懶拊捫擅斂晰枋榔梆械樂毆沮淬
 但知樗拂低敵渡等得畷濃獵伐盤昆秒譜噴篇捧某麻脈妄油預雷疏輪露乖佯傅同劭參咥啾噦娘壺婉懈岑嶼辭忻俊慄懶拗撻擒數哲粉條椹樞櫧榦沂涓
 叽談駐鳥亭擢杜痘匿駢煤八番枇病芙吻片抱望魔稔茂愈輿裸留臨路又侑餽冉劬厥咸啻擗壹姪孽峩廸悄愴憲掉據敲晨杪柵楮櫻櫻殷泗潤
 只男酌頂鶴摘斗燈鴟楂之梅肌挽微描膚分偏庖暴磨蓑模渝与螺琳賂并併傀冀辨厖萼喟噶坡壺嫋孰屹嶷廢微悚懃懦抃掏撼敝晨杼梭楔櫧櫻殳沂澑
 蜘段註長釣泥徒湯銅鍋乃媒幡飯尾廟腐鮒蔑崩房摩湊摸柳余羅溜燐櫓卽佩做今劑厂哇喊噪咾壯媯孩中嶽塵徭懶愿懷拔掣撈敞晤杰楷楫槧櫻殼沾淇
 菁檀衷眺吊鼎屠淘道捺粘培櫨頌備豹符物瞽峰忙盆蜜面清予淀流淋魯丕佻偈兩劈卷咆咯噬坏壘娜孥屬嶮廚惶惶懶懶拔擢撈敞晤杰楷楫槧櫻殼沾淇
 鐸鍛柱諜嬌邸吐東童雜捻輩箸範非票普淵癖呆妨本箕棉役雄浴掠倫蓮碗侈假兌剿下呶唳噫坎壙娥孕屐嶢廖徙恫愴惺找掎撕效晉杞梓櫟櫟櫻殼沾淇
 託誕抽調坪蹄電搭瞳乍念肺箱販避瓢斧覆壁包坊奔已免厄郵沃葎力聯蕨佶偃兒剩兀咀唸嘸圓壑姚子屎嶂廬從恬愍慄抉掖攬畋晏杆梳榆樂檳殞沒涕
 琢蛋忠蝶潰諦田投洞那年背函藩費漂敷複僻俸剖幌魅滅矢邑欲立領練詫佇兀剗卉呻啞嘲嗇壓妍孀屆蘄廂徇恂復懈扼捏擗攸易杓桎櫟櫻殼沾淇
 濡耽宙腸椿訂灤悼撞奈熱牌駁般誹標扶腹貞倣傍堦未牝野遊抑率陵簾瓦佗們几剪卅咒啖嘶圖壅姜嬢屁龜廁徑恤愍懷授搜搏收昂束栲榦櫻殼沾淇
 拓綻仲脹鐸艇殿島憧鈍寧盃莫繁被彪怖福米苦亡殆味姪耶誘慾律量煉鷺尙俯儻剔區皆焯嘴團墮嬢嬢尹嵬庠很恃惱應扛捍擎攷朶朶棠榕櫃辱汎浹
 托端中眺綴締宕導曇禰杯縛畔肥俵府服陞簿乏勃蔓鳴爺裕養陸遼漣惑佛俾儻剞劂匱叩啜營圓壞姆嬾尸帽麼徊恣惻懊扣挾摧攫昵霸桀棹榜櫻殼殃沛浚
 宅短嫡町薦程顛套堂頓濡敗爆班罷謬布幅閉母鵬鉗漫銘夜祐陽離諒憐賄估倬儻刺匯呱售噎圍燠姐嬪龍嵎并彌恍愷懶扞捐搏擎杳臘柳模藁殄沁浹
 灑淡秩朝佃汀纏唐騰沌姪馬迫汎碑媛夫伏柄慕鋒牧繭明門猶蓉痴稜烈和价倣儻劄匕呴啖嗽圍墅妝嬪專崙幘不恪愀憑戳挈攝攘昊曷楓接槃檢歹沂渢
 鷹歎逐暢櫬梯添刀頭敦任芭薄汎皮逼埠落弊墓邦朴迄命紋涌葉璃瞭劣倭仟倅儻刮匏吝啧嗾噬國增妣跚九嵒开徂恆惺撫孔拏搨攤旻朦框棣榑檠殃沚涓
 題旦蓄挑塚提店凍陶惇尿罵舶板疲筆付葺平募豐撲沫名悶湧耀理療列論仞倥儕刪匐吩咐噴囿堙奸嫻寶睂幘彙恚惶憚戰拱搦攀早曩枮棕梨槧欝歎澗
 醒探筑懲通挺展冬鐙屯如婆粕斑泌畢不風幣輔訪墨末冥問柚羊梨涼麗肋仗倪儻刎匍匐喫呐哭嗷罔壺奩寰崔帳彗快愆憔戮拮搓擺无曦枹棧櫟櫻殼斃沴渢
 第坦竹彫痛抵天党透隙入破箔搬比必瓶楓兵補褒卜抹娘貴有窯李梁零麓彷彿儻將岐慢彭恁惱憊戲挂搶攏呆曰粒椒榻檄歸汪洒
 大嘆畜張鉢悌填倒逃西乳琶白帆斐弼敏封併甫蜂僕又婿尤揖用履料鈴六仆倨僵凰匈吼哽噴口箜奠嬌寥崛幘彙恚惶憚戰拱搦攀早曩枮棕梨槧欝歎澗
 台丹築弔追弟典怒踏寅日派泊叛披肘頻部丙捕蓬北亦椋餅憂熔吏寮苓郎仄倚價處勿吭唔嗤囁壺奢嫗寢崔帳彌怕惄慵戛指搓擺无曦枹棧櫟櫻殼斃沴渢
 隊狸置喋津帝徹努藤突賑把伯伴彼疋斌武紛鞭縫貿膜牟儲友曜濫了冷牢亢俛饒冰飭吁曷喚囁堋奐嫗寢崗帷彈怛悵慄菱拂揄擗施暎櫻櫻櫻殼斃沴渢
 鯛樽螭帖椎庭轍土膳鳶虹杷拍半扉膝貧葡文鋪芳防鮀霧目幽洋蘭僚怜老亶俚僭潤勵呀哦嘎嚙堦奚媽實翠帶弸怎惆憇戌拊揶擗旁暉相棘榦櫻殼斃沴渢
 黯鰐致寵墜底撤度討櫻肉播博判悲颺瀕舞雱保胞鉢枕矛木宥楊藍亮嶺狼毫俑僭冽勲吽哥嗟囁埋奎寤寢崔帳彌怕惄慵戛指搓擺无曦枹棧櫟櫻殼斃沴渢
 貸迦池丁貨呈鑄都到毒汝覩秤犒否終鰐附奮勉泡膨哩務網佑庸卵慮令榔于俟傳冢勞叮咫喇囁埒夸媚宸峩已升忿悸怨憤拆揀擂斟暎楊梵楞樣欝毫泛
 袋異智貯珍貞適途統篤難農這鳩卑鼻蛭阜焚便法肪枚眠盲唯容酪侶類朗舒倪僕冤昂變咨喃嚴埔夭婪宦岷巫廳忝惡慚懼拿揩擘斛煊柯桴榔樊欸毬汨
 苔奪恥著沈荆笛賭糖秃軟膾陪闋匪美蒜赴扮遍朋紡昧民猛輸妖落龍累弄豫儘僉冕勍曼哈啼嚙塊夬婢它峩巖廩忱悒慘懼擎擎撻變暉枷櫻林櫻櫻櫻殼斃沴渢

HKSCS (Hong Kong Supplementary Character Set)

GBK (Simplified Chinese)

樵障色辛遂政積栴膳匝驛多逮奪斷忠牒摶抵典砾等峇鈍圭播這畠藩被組付風併保法肪妹密緬耶猶羊藍旅麟練倭舒修儻涼劈卒吭呖喚址墅夸
 樟鐘職身衰成石栓繕層霜他退辰彈宙潮梅悌鉄砥祷銅曇屎把陪烟般肥媛不楓內鞭朋紡埋岬綿爺涌窯濫處鱗簾論事併偕儼潤劔區听曷喃嘵坼塹卒姪
 梢鍾織診翠性析撰禪宋鎗遜貸達壇仲朝塚弟迭鍛痘道吞如巴賠肌繁罷姬瓶封聞弁方冒麻箕棉夜湧用欄侶隣煉錄豫佩會儼冽劔仁呀咷啼寥坎堙夭嫋
 松鉦燭親粹征昔扇全爽遭村袋但団中暢通廷轍都當葡萄入蚤壳幡畔緋桧敏部文婉放棒魔已免治袖熔嵐龍輪漣肋丁佻假儻況劔匱吽咤單鳴址塙夬
 晶醬殖薪睡姓斥戰然奏送損苔叩鍛着挑痛庭撤途燈胴遁乳覩買櫨班秘逼頻蕪霧勉捧某磨魅滅也有溶卯竜臨憐祿亂侘偃僵冰劍匱吁哂唧嚙坏夥婢
 昭賞植芯炊勢戚川漸壯走尊腿只誕嫡懲鉗底徹賭灯童豚日農獵苦犯碑筆賓葡紛便抱望摩未牝夕揖洋亂隆琳恋麓乘侏僥僂冲剽匯叱咷噶塙夢娜
 昌象拭臣水制惜尖善喪裝孫胎蛸蛋茶徵追帝哲菟涛瞳汎廿膾煤肇版皮畢貧舞糞遍庖暴盆味姪門憂様酪粒燐廉六乖侈們儕沴剿匣叭咨啾噪圓瑩旁娑
 屝詳飾紳推淒席專前倉藻存滯夙胆室彫槌定溺登湯洞敦虹腦棟箸汎疲必瀨武粉返崩房凡蔓鳴紋悠楊落疏淋裂郎又佶俯儕決劄仁叨哈啻噬嗇塙隻娟
 捷詔埴秦帥是隻宣鮮叢蒼揃泰茸耽秩張椎堤鏘渡淘撞惇肉能梅硃汜泌弼浜撫奮辺峯忙翻漫銘悶幽曜絡留林烈蟬ノ仔俾儘ノ剩匕叮哄喟嘯圖堡爻娥
 掌証囑神垂歛脆占閃双葬其替諾綻逐弔墜呈適杜盜憧屯賑納媒箱板比肘斌侮焚編峰忘本滿迷問宥擁洛琉厘劣聾并佗倬儂幕剏匏變咬喊噤團場夕姚
 招訟錠真吹瀨稅千銑創莊袖戴濁筭蓄疔津貞笛斗棟導頓勾濃培函斑斐菱彬附扮篇宝帽奔慢盟貫友搖雷溜倫列老、佝倩儕寫剪匐曼咥咯噫圓媯壽姪
 抄裳釀疹逗世齐先錢僧草卒態鐸端筑帳陳剃的徒榜堂滯逐惱倍麦搬披膝品阜憤片奉妨幌万明勑勇揚賴流綠歷籠卯佛倡儻冢剔匍叟咸喀嚙圍塙壺妍
 承衝讓申厨寸靜仙遷鼠聰統怠託短竹帖鎮儕滴屠桃同酉式囊配駁帆批彥鰐赴墳交報坊掘膚命戾侵庸菜劉力曆狼个估倣僵寇剏甸雙粵喙嘲國堋堦姜
 彰蕉蒸深岡擢青蝉選迺綜族待琢炭畜寵貨停敵妬東動寅尼埜輩莫叛扉毬蛭蛭賦噴偏呆剖殆爾名尤佑容來略領齡牢丕佚伴價冤刺匈纂哇唧嘶圈塙壹姨
 廰蔣穰浸醉澄醒舌踐阻總賊帶濯湛築喋珍低擢塗搭働苦二之肺縛反悲疋蒜負吻餽包傍沒併冥餅唯妖裸掠陵麗漏丐伉倅僅「剋勿參咆唳嘴圍塙壺姆
 床菖疊榛須裙逝絕賤訴糟屬岱汎淡馳凋沈亭摘堵投闌鳶汝迺背爆半彼匹鋸譜分蔑俸亡勃迄娘勿輸幼螺葎量靈浪弋价倥僭冕剏勺厶吩咐營圉聖壯姐
 庄肖狀森諷雀請雪詮蘇窓俗耐拓歎遲兆朕鶴泥吐悼騰屆難乃牌漠判庇裨錨芙鮒警倣乏釦沫婿空諭傭羅立遼零榔腕仟倪僭蕡荆勸厥咄啞器囿塙嬖
 尚紹淨晉笥頗誓說薦組相速對抉旦蜘丁直釣鼎兔嶋頭榦軟粘盆曝伴妃柊苗膚物別菩鵬穆末椋目癒預淀率諒隸樓碗仞倔僥胄剏勵廝呶啖噎罔埒壤俵
 少粧杖新勒昔誠節船素瘦足堆托探致貯抄吊釘電島陶突楠燃杯迫隼否鼻秒腐仏碧簿鳳睦抹鶴默油輿翼律良鈴朗灣仞倨僕同剏勲厥咀焯唶罔捕墁佞
 小笑条振陣相西窃舛粗爭測体宅担置著勅爪鄭田宕鐙凸南撲敗薄蛤卑美病符沸癱母飽牧又霧木愈譽翌陸糧苓弄椀仗倚僖冉刮勑夏呻啜嘛厖塊妝
 將章擾慎迅杉製設腺租燥束驛啄坦稚苧鳥紬邸灑套透椽瞬捻排舶犒匪眉描父払壁暮鋒朴俣矛儲渝与浴離稜礼廊蕨彷倣儂冊刪飭廁咒售嗽口埃壘妁
 宵称情心訊据声折羨祖漕捉駄卓嘆痴猪頂嬌遜殿塘逃枥繩念拌粕嘶蛩廟浮弗僻戌邦撲亦牟蒙鑑余沃里瞭玲婁菴仆悌傳回刲勲厥咀焯唶罔捕墁佞
 媚祥常審腎雞聖摸纖礎槽息陀瀧单池瀧長壺蹄伝塔踏蹠駢年廢箔鳩蕃昆豹普淵貞慕豐墨樹無耗敷予欲裡療怜勞託仄俐儕口刎勑厂呷唧噴囁併塙嬖
 妾礁嬢寢尽趨精接線疎槍即橈滴丹智樗銚坪諦点唐豆独楂熱俳白閥磬枇評斧覆米墓訪卜鱈夢網柳夕抑裏猶嶺露鰐仍俚僉冀剏勞卷呱嘵噉塙塙獎
 燐硝壤娠甚枢盛拙箭疏巢則舵鷹誰恥駐跳潰訂顛刀膳毒鍋貓馬泊筏盤微表敷複陞募褒僕枉務盲靖融慾痢涼勵路亘从俑傲兮刲勲卻吆喰嘵塙塙塙
 詈省場唇尋數生切穿狙曹側柁題樽弛鑄超椿艇転凍討篤捺葱芭柏拔番尾票扶腹閉穗蜂北鮪眠猛躍雄養璃梁冷賂瓦賣俛僕愈刃勍哿咎咷噴囁併塙嬖
 咨症城侵壬嵩牲碩旋楚早促打醍鰐地耐譏鍔締貼冬藤禿灘寧罵拍罰晚備瓢怖福蔽輔蓬頰枕民毛訥郵陽理料例炉鷺毫俘傅兩函勁卮呵哺嗤囁塙塙
 商照刺信塵崇清蹟煽曾操造惰第狸知註調綴程甜党蕩督謎祢婆博伐挽簸漂府服並補萌吠膜耗孟藥邑遙梨寮伶櫟梓京俎餽競口券口咏哭嗜囁塙塙
 升湘丞辱人隨棲責染蛆掃臘唾代述男注賑柘汀店奴統德乍認派萩堯煩非彪富副平捕縫貿哩稔模厄裕要吏僚累鍊賄亟儘偬兌夙劬口吩咐喚嚴塙塙
 匠涉丈蝕震錘晴脊洗塑搜憎訖鯛豎段柱聽漬梯展土糖得內忍波矧澆采避俵婦伏幣圃砲貌每蓑摸矢祐蓉利亮淚連亞命倂兒處辨卉呐啼鳴嚙垂塙嬖
 勝消上食針錐星績淺增想增汰黛異檀昼眺佃提天度箇匿那妊杷秤鉢範費謬夫蕗屏鋪烹謀枚湊麵弥由葉覽了望蓮話于來做兀几辨出吮哦喚嚮塙塙
 償沼鞘触進醉整籍泉粳惣像太隊脫暖抽町櫬挺填努答鵠奈任霸蠅八販誹百埠葺兵舖泡膨昧蜜面野猷耀蘭虜璫聯和式佯偈几凜劑卅吼哥喇嚙坏塙塙
 夾

SHIFT_JIS (Japanese)

貢染姐掃臥睡代述男注脹柘汀店奴統德乍認派萩發煩非彪富副平捕縫貿哩稔模厄裕要吏僚累鍊賄亟儘尙兌夙劬已盼唔喚嚴壘燠奐婪學員崑昂廟拂脊洗塑搜憎訖鯛堅段柱聽瀆梯展土糖得內忍波矧澆采避俵婦伏幣圃砲貌每蓑摸矢祐蓉利亮淚連亞命倘兒處辨卉呐唏鳴嚶垂樽奕婢孵屎崛帀廡徂績浅增想增汰黛異檀昼眺佃提天度箇匿那妊杷秤鉢範費謬夫落墀鋪烹謀枚湊麵弥由葉覽了望蓮話于來做兀几辨出吮哦嘵嚮墘墟奇娶孳屆峯厄廢往籍泉癮惣像太隊脫暖抽町櫈挺填努答鵠奈任霸蠅八販誂百埠葺兵舖泡膨昧蜜面野猷耀蘭虜瑠聯和式佯偈儿凜劑卅吼哥喇嚦坏壻夾娘孰屁寄已塵彷積梅膳匝騷多逮奪斷忠牒摶抵典砾等峽鈍圭播這畠藩被紐付風併保法肪妹密繡耶猶羊藍旅麟練倭舒侑修儻涼劈卒吭咁喰嘵址墅夸婉孩尹崗巫廢石栓繕層霜他退辰彈宙潮梅悌鉄砥禱銅曇尿把陪烟般肥媛不楓丙鞭朋紡埋岬綿爺涌窯濫慮鱗簾論事佰偕儼潤劔區听曷喃噏坼塹卒姪孥尸岬《廝彭析撰禪宋鎗遜貸達壇仲朝塚弟迭鍛痘道香如巴賠肌繁罷姬瓶封聞弁方冒麻箕棉夜湧用欄侶隣煉錄豫佩會儼冽劔仁呀咷啼嚦坏壻夾娘孰屁寄已塵彷昔扇全爽遭村袋但団中暢通廷轍都當葡萄入蚤壳幡畔紺松敏部文婉放棒魔已免治柚熔嵐龍輪漣肋丨佻假儻況劔險吽咤單嘵妯塘夬嫋孚尤峪巒廖彙斥戰然奏送損苔叩鍛着挑痛庭撤途燈胴迺乳視買櫨班秘逼頻蕪霧勉捧某磨魅滅也有溶卵竈臨憐祿亂侘偃僵冰劍匱吁哂唧噏坏毀夥娉孕尠島巔廢彗戚川漸壯走尊腿只誕嫡懲鎚底徹賭灯童豚日農狼苦犯碑筆賓葡紛便抱望摩未牝夕揖洋乱隆琳恋麓乘侏兩傍冲剽匯叱咫喘嘴扒塗夢娜子尔峭巍廐彖惜尖善喪裝孫胎蛸蛋茶微追帝哲菟涛瞳沌甘膿煤肇版皮畢貧舞糞遍庖暴盆味姪門憂樣酪粒燐廉六乖侈們儕沴剿匣叭咨啾噪圜瑩旁娑嬌對哽巒廸旦席專前倉藻存滯肅胆室彫槌定溺登湯洞敦虹腦模箸汎疲必瀕武粉返崩房凡蔓鳴紋悠楊落疏淋裂郎又佶俯儻決剖𠃚叨哈音噬嗇塙負娟嬪專峽嶼廂弯隻宣鮮叢蒼揃泰茸耽秩張椎堤鎗渡淘撞惇肉能梅硃氾弼浜撫奮邊峯忙翻漫銘悶幽曜絡留林烈蟬丨仔俾鑑洩剩匕叮咁喟嘯圖堡爻娥嬢將峩巔廐彎脆占閃双葬其替諾綻逐弔墜呈適杜盜憧屯賑納媒箱板比肘斌侮焚編峰忘本滿迷問宥擁洛琉厘劣聾井佗倬儂幕剗匏變咬喊噤團場夕姚嬪冠峙窿庠彌稅千銑創莊袖戴濁筭蓄序津貞笛斗棟導嚙勾濃培函斑斐菱彬附扮篇宝帽奔慢盟貫友搖雷溜倫列老、尙倩偶寫剪匐曼咥咯噫圓堦壽姪嬪寶峩嶽广彈斉先錢僧草卒態鐸端筑帳陳剃的徒榜堂灘述惱倍麦搬披膝品阜憤片奉妨幌万明勑勇揚賴流綠歷筆卯佛倡儻冢剔匍叟咸喀嚦圍壺姪嬪寶帖嶮麼彌逝絕賤訴糟屬岱沵淡馳凋沈亭摘堵投鬪鳶汝迺背爆半彼匹鈎譜分蔑俸亡勃迄娘勿輸幼螺葎量靈浪式价倥偬冕到勺厶吩咐唸營固聖姐嬪寢峯巒彌彌請雪詮蘇窓俗耐拓歎遲兆朕鶴泥吐悼騰届難乃牌漠判庇裨錨芙鮒賦噴偏呆剖殆爾名尤佑容來略領齡牢不佚伴價冤刺匈篡哇唧嘶圈坪壹姨嬪寫岷嶝并弭誓說薦組相速對抯旦蜘丁直釣鼎兎嶋頭榦軟粘盃曝伴妃柊苗膚物別菩鵬穆末掠目癒預淀率諒隸樓碗仞倔饒胃剗勵嘶呶啖噎罔埒壞僕嬪實岫巒幙弋誠節船素瘦足堆托探致貯抄吊釘電島陶突楠燃杯迫隼否鼻秒腐仏碧簿鳳睦抹鵠默油輿翼律良鈴朗湾尙倨僞罔劄勳厥咀咤喰圖埔墁僕嬪寢峯巒巒彌彌西窓舛粗爭測体宅担置著勅爪鄭田宕鐙凸南撲敗薄蛤卑美病符沸癖母飽牧又霧木愈譽翌陸糧苓弄椀仗倚僖冉刮勑夏呻啜嘛罔埆壘妝嫖寐岔羞幙製設腺租燥束驛啄坦稚芋鳥紬邸澱套透橡啜捨排舶墮匪眉描父払壁暮鋒朴俣矛儲偷与浴離稜礼廊蕨仇俾冊刪飭廁咒售嗽口埃壘灼嫩寃岑嵬幙声折羨祖漕捉駄卓嘆痴猪頂媾遁殿塘逃柵繩念拝粕嘶蛩廟浮弗僻戊邦撲亦牟蒙鑑余沃里瞭玲婁藁仆悌傳回刲勳厥咀咤喰圖埔墁僕嬪寢峯巒巒彌彌聖摵纏礎槽息陀瀧单池瀧長壠蹄伝塔踏讌馴年廢箔鳩蕃毘豹普淵貢慕豐墨樹無耗敷予欲裡療怜勞詫仄俐僕口刎勑厂呷唧噴囁墘壅墘壘奐奐寇屹岡精接線疎槍即橈淹丹智橈跳坪諦点唐豆独櫛熱俳白閥磐枇評斧覆米墓訪卜鱒夢網柳夕抑裏猶嶺露鰐仍俚僉冀剗勞卷呱嘵嗷贈塙壑獎嫣寃劣嵒幙盛拙箭疏巢則舵鷹誰恥駐跳潰訂顛刀膳毒鍋猫馬泊筏盤微表敷複陞募褒僕枉務盲靖融慾痢涼励路亘从俑傲今刲勳卻吆喰嘵嚦墘堵墘壓奧媽宸此嵌幄爻牛碩旋楚早促打醍餚地酌謀鐸締貼冬藤禿灘寧罵拍罰晚備瓢怖福蔽蓬頰枕民毛訥郵陽理料例炉鷺毫俘傅兩函勁卮呵哺嗤囁垓墮奢嫊它屬嵩帶廳清蹟煽曾操造惰第狸知註調綴程甜党蕩督謎称婆博伐挽簸漂府服並補萌吠膜耗孟藥邑遙梨寮伶櫓梓京俎攸競口券口咏哭嗜囁炮堵奘嫊媾宀屏嶮帛廐正跡煎曾搔贈妥大谷值衷蝶葛禎纏倒董特雍爾破剥髮飯桶冰布幅柄甫芳防幕妙妄約遊踊李凌令魯惑亢俟傀競凰劫乍呴哮嘎嚼墘墻奐奐寇屹岡栖赤潛措挿藏墮台棚談虫腸迂碇添怒到浣風濡琶伯醜頌飛標富復弊步胞鉢槓脈茂役誘謠履兩類呂脇一倪偷免凭劭準吝哽嗟囁坡壞奎媚孝屐崔帙廐

UHC (Korean)

車綏汁順庄衝辱塵雀隻撰疏綜他題淡逐朝綴艇電投瞳謎粘媒八盤琵錨赴奮弁縫貿抹明也猷踊裡諒裂賄侘僞剽變喚垓奧宸帽彈悖憚擔撻曼杆楫槧氈謝樹柔醇少蕉蝕刃頗脆扇狙糟遜醍歎蓄暢漬締田悼洞乍燃培畷番昆苗賦焚婉砲貌又命門猶要裏良烈歪侏僖剿曼嗚坡奠宦嵌弩悛憔拿擇昊朶楸檣麾者授戎遵小菖食仁菅醒川楚窓村第旦筑挑佃程灤嶼撞那撲倍肌晚枇秒負扮烹謀亦名紋涌蓉痢糧劣話侈傳剩雙喇岱奢孺嵩弑悚惺擎擎擅旱朮楷檢毫紗呪十巡宵肖色人杉逝尖曾相損大探竹懲櫬禎殿島憧奈捻配幡挽微病譜憤便泡膨桓冥悶湧葉璃稜列和信僂剪纂喻址奚學崔廳悍惠拗擒无臘楹檄毬社受充純娟紹職震据請宣措燥尊台坦畜彫塚碇点宕導鈍念輩肇飯尾描芙墳遍法肪枕娘問柚耀理瞭麗倭併僉剔參啼坼奎孵崑廬恙標拔據旒朦棠憶毓煮儒住盾妾粧纖針雛誓占塑漕孫代嘆築張通汀貼套堂曇年肺箸頌備廟膚噴返朋紡膜鵝貢有羊梨療零論佛傲刺厥單坎奐孰崗廢恬慝抒撈旌胥棹樸毋斜首什潤嘗笑燭進趨誠千膳槽存黛丹馳弔痛梯纏塘同吞熱背箱煩飛豹腐吻編方冒幕霧尤揖窯李梁隸肋佚傅剋夏喘嗇奕孩峽塵恂抉撥旁曷棗櫟毆赦酒醜準唱章殖辛嵩西先繕槍其隊誰蠅帳鎋提添塔動頓貓牌函範非評符分篇放棒枚矛勿憂用履料鈴麓伉愧剝廁喊圖夾孚峙廚恤慾扼撓斷曰椒橙殼捨趣集淳商祥植身崇製仙全曹袖逮樽致帖追挺店唐騰遁寧盃駁販避表父物片捧某昧牟目悠熔吏寮玲六价偷刮卷喀團天孕帖廣恃慙扁攬研曦棕橄殷射腫酬殉哨礁拭診瑞聖舌然早卒退狸置寵槌抵展刀頭豚濡杯莫藩費票浮沸偏抱望妹無木幽溶利凌怜聾仟做刪卞喙圓壽子岷廖恣慘戰擎斟曠棧樓殲縞種酋楯召硝飾親錘精絕漸操族貸谷稚凋椎悌天凍陶沌認敗縛般誹瓢普弗蔑崩暴埋夢儲宥洋蘭僚嶺老仗偈刎卽啖圍堵孀岫廐恍慷戮搏斛暖棍樣殲芝珠週旬升省墳薪錐盛雪善掃賊袋棚痴兆墜弟典冬透敦忍排爆繁被漂斧淵警峯房麻務蒙友楊藍亮冷狼仄偕凰卉啣國壹嬪岑廸恪慄截搗變暉棘樊殲柴狩輯循匠症鋏芯遂生節前想俗苔豎池丁津廷迭倒逃惇妊佯漠畔肥冰敷覆別峰忙魔眠耗勇曜濫了例牢仍會處區哺圈壺嬪屹廁怎憑戡攝斃噦梵樞殘篠殊蹴准勝照蒸臣衰牲設鮮層速腿異智貯陳庭轍怒踏屯任馬曝班罷標扶複碧奉忘磨民網優擁欄虜伶漏亶假涼匯哭圓壯嬖屬庠快愴戌搖斂曉條權殃質朱讐駿償焦杖紳翠正折閃宋足胎奪恥著貨底撤奴豆滌尿芭迫犯絢彪怖腹癱報帽摩妙盲佑揚嵐旅令浪亢偃冽匣哮罔壘嬪孱怡慎戍渝數暹梔槿殄疾手襲舜傷湘擾秦睡栖接銑爽測泰辰弛苧珍帝徹土膳酉如罵薄版秘俵府福璧包妨益脈猛唯庸卵慮類朗亞倆羃匕哥囑壞嬌屎幢忿愿戈揶敲擊梭樂歟漆守衆竣除消情神炊棲拙遷奏束替達地猪沈定哲度討寅入婆舶汎碑謬布服僻俸坊凡稔毛輸容酪侶累弄于們冕匏咤嚼壘嫩屆幟惠慊戀揆敞陽棓榔歸悉取蒐瞬鋤沼常疹水晴切選喪捉戴但知樗朕堤溺努藤鳶乳破粕汜皮百富幅貢倣剖本蓑孟諭妖落龍瑠廊舒俯冀匐咫嚴擴端尹幘忖愧懼揀敍喧梟榴歡室主舟春怨樵場申推星碩詮倉息叩值駐直呈鑰砥蕩突日琶箔板疲紐婦復米苦傍奔湊妄癪幼絡隆麟婁豫倬兮匍咨嚮嫂尸幄徨愴懿掉敖暉桿榕歟叱若秋術助晶丈森逗性責腺鼠造耐濁檀注長停滴賭糖篤尼霸拍叛披畢不風炳戊鳳勃箕面柳余螺琉臨櫓又倚兌勵咸噫哈嚙壓媚龍帶徧懃懶掀敞陽棓榔歸失弱習俊序梢冗浸垂政跡船創側待茸男註鳥剃笛都到橡肉波泊搬比逼埠伏閉母乏堦密模愈預雷疏隣路乘倪兩匈哄噶壅婢對帛徘徊懶捏攸量桎樞歇執寂終述女松丞榛吹成赤舛僧促岱諾段衷頂儂的途統毒賑杷柏帆斐筆付葺蔽暮鵬殆岬摸愉輿裸留輪賂乖倨競勸咬嘯墮娶專帑徒愍懦搜收晟桀槧飲竺爵拾塾薯昇鐘振諷勢籍箭訴像楨琢鍛忠諜鶴摘杜等特難巴萩判悲菱賓部幣募邦牧未棉約融翼劉淋連碗俎儼勤咀嘴墟娟寥巖徊惶憇拱攘晝枢槐欒七錫秀出諸昭上晉醉征脊羨遡贈堆鐸暖柱跳低敵登箇秃二播博反批必敏楓弊慕飽穆已緬靖予羅溜琳魯丕俚兒勳咆嘲壞娜寶已徇惻懋捐攬皓析寨鬱軸酌洲熟諸昌障新須姓績線阻憎陀託壇抽超亭擢渡答督汝把伯半扉弼頻封平墓鋒睦魅綿躍夕淀流燐呂腕俑兀飭咐嘶懊娑寫巫徑惺愴拯攀晤枸槁欒式杓修祝署掌鉦審陣淒石旋組遭打拓蛋仲蝶吊鼎徒燈匿楠農這蛤庇彥瀕葡丙補訪撲蔓滅厄郵浴掠厘聯蕨倪儻勍呱嗽塹姚寢巍彷愕懷拉擺晁東楞櫻鹿尺州淑庶招賞姵迅是析煽素送惰托耽中腸爪釘屠湯峽南膾陪鳩妃疋浜舞聞甫褒墨漫牝矢邑沃立倫練藁侖儼勁咎嗾塹姪實嶼彭悽應拋擲晝柯榆櫛汐勺就宿渚抄象唇訊畝昔煎粗走妥宅綻着脹紬鄭妬淘銅馴能賠閥否匹斌武文捕蜂卜慢姪野遊欲率力簾鰐來儕辨呵嘔塹姜寤嶽彙悴懊毋擴暎枷榔欒耳蛇囚銃所廠訟伸尋摺惜泉礎葬太瀧炭室牒椿諦吐東胴捺之煤伐蕃美蛭附糞保芳吠沫迷夜祐養離量憐惑併儕劖吼嗜奸寇嶽惡憑拜舉袒枉楮櫻而遮需重初床裳尻壬澄席栓疎草多鷹湛秩潮鷄訂兔搭童灘乃梅鉢磐眉蒜阜粉鞭胞防末盟治由陽里遼廉脇佩饒劍叭嗟垠獎冤嵬彌惻憊拈擘杳杞楔蘂

Demo - Pangrams

Pangrams - (from Wikipedia)

English

The quick brown fox jumps over a lazy dog

Bulgarian

Жълтата дюля беше щастлива, че пухът, който цъфна, замръзна като гъон.

За миг бях в чужд плюшен скърцащ фотьойл.

Catalan

Jove xef, porti whisky amb quinze glaçons d'hidrogen, coi!

Aqueix betzol, Jan, comprava whisky de figa

Czech

Příliš žluťoučký kůň úpěl dábelské ódy

Danish

Høj bly gom vandt fræk sexquiz på wc

Dutch

Doch Bep, flink sexy qua vorm, zwijgt

Pa's wijze lynx bezag vroom het fikse aquaduct

Finnish

Törkylempiä vongahdus

French

Portez ce vieux whisky au juge blond qui fume

Bâchez la queue du wagon-taxi avec les pyjamas du fakir

Voyez le brick géant que j'examine près du wharf

German

Victor jagt zwölf Boxkämpfer quer über den großen Sylter Deich

"Fix, Schwyz!" quäkt Jürgen blöd vom Paß

""Falsches Üben von Xylophonmusik quält jeden größeren Zwerg"

Greek

Γαζίες καὶ μυρτιές δὲν θὰ βρῶ πιὰ στὸ χρυσαφὶ ξέφωτο

Ξεσκεπάζω τὴν ψυχοφθόρα βδελυγμία.

Ζαφείρι δέξου πάγκαλο, βαθῶν ψυχῆς τὸ σῆμα.

Hebrew

dag skran shat bim maoczab olpetu mazia chbra

או הנסה אלהים, לבוא לךת לו גוי מקרוב גוי, במסת באחת ובמוותים ובמלחמה וביד חזקה ובזרע נטויה, ובמוראים גדלים: ככל אשר-עשה לכם יהוה אלהיכם, במצרים--לענין

לכן חכו לי נאם יהוה ליום קומי לעד, כי משפטיו לאסף גוים לקבצי ממלכות, לשפך עליהם זעמי כל חרון אפי, כי באש קנאתי תאכל כל הארץ

שפּן אכל קצַת גָזֶר בְטֻעַם חִסָּה, וְדִי.

Hungarian

Egy hútlen vejét fülönctsípő, dühös mexikói úr Wesselényinél máról Quitóban.

Icelandic

Kæmi ný öxi hér ykist þjófum nú bæði víl og ádrepa

Irish

D'fhuasail íosa Úrmhac na hÓighe Beannaithe pór Éava agus Ádhaimh

D'fuasail íosa Úrmac na hÓige Beannaite pór Éaba agus Ádairí

Italian

"Quel fez sghembo copre davanti"

"Ma la volpe col suo balzo ha raggiunto il quieto Fido"

"Quel vituperabile xenofobo zelante assaggia il whisky ed esclama: alleluja!"

Japanese

Iroha Uta

いろはにはへと ちりぬるを わかよたれそ つねならむ うゐのおくやま けふこえて あさきゆめみし 烫
ひもせず

色は匂へど 散りぬるを 我が世誰ぞ 常ならむ 有為の奥山 今日越えて 浅き夢見じ 酔ひもせず (ん)

Tori Naku Uta

とりなくこゑす ゆめさせ みよあけわたる ひんかしを そらいろはえて おきつへに ほふねむれゐぬ
もやのうち

鳥啼く声す 夢覚ませ 見よ明け渡る 東を 空色栄えて 沖つ辺に 帆船群れゐぬ 霧の中

Ametsuchi No Uta

あめ つち ほし そら / やま かは みね たに / くも きり むろ こけ / ひと いぬ うへ すゑ / ゆわ さる お
ふせよ / えのえ*を なれ ゐて

天地 星空 / 山 川 峰 谷 / 雲 霧 室 苔 / 人 犬 上 末 / 硫 黄 猿 生 ふ 為 よ / 檻 の 枝 を 飼 れ 居 て

Taini no Uta

たゐにいて なつむわれをそ きみめすと あさりおひゆく やましろの うちゑへるこら もはほせよ えふ
ねかけぬ

田居に出で 菜摘むわれをぞ 君召すと 求食り追ひゆく 山城の 打酔へる子ら 藻葉干せよ え舟繋けぬ

Korean

키스의 고유조건은 입술끼리 만나야 하고 특별한 기술은 필요치 않다.

Lithuanian

Įlinkdama fechtuotojo špaga sublykčiojusi pragréžé apvalų arbūzą

Norwegian

Vår sære Zulu fra badeøya spilte jo whist og quickstep i min taxi.

Høvdingens kjære squaw får litt pizza i Mexico by

Polish

Pójdźże, kiń tę chmurność w głęb flaszy!

Pchnąć w tę łódź jeża lub ośm skrzyń fig.

Mężny bądz, chroń pułk twój i sześć flag.

Portuguese

Blitz prende ex-vesgo com cheque fajuto.

Gazeta publica hoje no jornal uma breve nota de faxina na quermesse.

À noite, vovô Kowalsky vê o ímã cair no pé do pingüim queixoso e vovó põe açúcar no chá de tâmaras do jabuti feliz.

Luís argüia à Júlia que «brações, fé, chá, óxido, pôr, zângão» eram palavras do português.

Romanian

Gheorghe, obezul, a reușit să obțină jucându-se un flux în Quebec de o mie kilowatioră.

Russian

В чащах юга жил бы цитрус? Да, но фальшивый экземпляр!

(Using quasiobsolete spelling for last word to include ъ) В чащах юга жил бы цитрус? Да, но фальшивый
экземпляръ!

Эх, чужак! Общий съём цен шляп (юфть) — вдрызг!
Экс-граф? Плюш изъят. Бьём чуждый цен хвощ!
Съешь ещё этих мягких французских булок, да выпей же чаю.
Широкая электрификация южных губерний даст мощный толчок подъёму сельского хозяйства.

Serbian

Љубазни фењерција чајавог лица хоће да ми покаже штос.

Ljubazni fenjerdžija čađavog lica hoće da mi pokaže štos.

Slovene

Šerif bo za vajo spet kuhal domače žgance

Piškur molče grabi fižol z dna cezijeve hoste

Spanish

El veloz murciélagos hindú comía feliz cardillo y kiwi. La cigüeña tocaba el saxofón detrás del palenque de paja.

El pingüino Wenceslao hizo kilómetros bajo exhaustiva lluvia y frío, añoraba a su querido cachorro.

Jovencillo emponzoñado de whisky: ¡qué figurota exhibe!

Ese libro explica en su epígrafe las hazañas y aventuras de Don Quijote de la Mancha en Kuwait.

Queda gazpacho, fibra, látex, jamón, kiwi y viñas.

Whisky bueno: iexcitad mi frágil pequeña vejez!

Swedish

Flygande bäckasiner söka hwila på mjuka tuvor.

Yxskaftbud, ge vår wczonmö iqhjälp.

Thai

ເມື່ອມານຸ່ຍົດປະໂຫວີສະເພີແລັກຄຸນດໍາ ກ່ວາບຮຣດາຜູ້ງສັຕ້ງເດັ່ນຈານ ຈົງຝ່າທີ່ນີ້ພັດທະນາວິຊາການ ອຢ່າລ້າງພລາວູຖາເຂົ່າໜ່າມົກ໌າໄຄຣ ໄນເຖື່ອ^ໄ ໂກງໂກຮແໜ່ງໜັດຢືດຢັດດໍາ ທັດອກັນເໜີ້ອນກີ່ພາວ້າໜາສັຍ ປົງປົບຕິປະພາດຕິກົງກໍາຫັດໃຈ ພູດຈາໃຫ້ຈະໆ ຈໍາ ນ່າພັ້ງເຂົ່າ

Turkish

Pijamalı hasta yağız şoföre çabucak güvendi.

Ukrainian

Чуєш їх, доцю, га? Кумедна ж ти, прощайся без Ґольфів!

Жебракують філософи при ґанку церкви в Гадячі, ще й шатро їхнє п'яне знаємо.

Demo - Other languages

Arabic

Arabic Supplement

Arabic Presentation Forms A

Arabic Presentation Forms B

Hebrew

... אַבְגָּד הַוְזָחֶת יִדְכְּלֵם מִנְסָעָף אַצְקָר שְׁתַׁוְוִי'"'

Hebrew Presentation Forms

עד חכמים רת ששה שאבגדה ווֹט יכלאך רשות ב-5 פ"א

Thai

Bengali

Gujarati

આ આ દ્વારા લિખિત ક્રમ એંઝો ઓર્ડર ક અગધ ક થાય છ જ મ અટ ઠ ક દેણ ત થાય દ ધ ન પ ક બ ભ મ એ ર૦૧૩૨૫૪૫૬૭૮૮

Malavalam

ଆ ଆ ହୁ ହୁଣ ଉ ହୁ ଫୁ ନ କ କା କୁ କ ବ ଗ ଲି ଏ ଚ ରି ଜୀ ଯେ ତେ ୮୦ ଧ ଯ ଶ ଗୁ ତ ମି ୩ ଧ ନ ପ ନମ ବୁ କ ମ ଯ ର ର ଲ ଛୁ ଫ ବ ଶ ଶ ଶ ଶ ହ ଏ ମି ହୁ ନ କି ରି ବୁ ଯ ଫ

Tamil

அ ஆ இ ஈ உ ஹ ர ற ல ள மு வ ண ஷ ஸ ஹ ய 0 க உ ந ச ஞ சு எ அ க ய ள கு வ மீ ஹ ட பு ஸ வ ஞ ஞ நீ

Vietnamese

The Vietnamese alphabets are listed in several noncontiguous Unicode ranges:

Thousand Character Classic (千字文)

南朝梁敕员外散骑侍郎 周兴嗣 撰

天地玄黄，宇宙洪荒，日月盈昃，辰宿列张，
寒来暑往，秋收冬藏，闰余成岁，律吕调阳，
云腾致雨，露结为霜，金生丽水，玉出崐崙，
剑号巨阙，珠称夜光，果珍李柰，菜重芥姜，
海咸河淡，鳞潜羽翔，龙师火帝，鸟官人皇，
始制文字，乃服衣裳，推位让国，有虞陶唐，
吊民伐罪，周发殷汤，坐朝问道，垂拱平章，
爱育黎首，臣伏戎羌，遐迩壹体，率宾归王，
鸣凤在竹，白驹食场，化被草木，赖及万方，
盖此身发，四大五常，恭惟鞠养，岂敢毁伤，
女慕贞洁，男效才良，知过必改，得能莫忘，
罔谈彼短，靡恃己长，信使可覆，器欲难量，
墨悲丝染，诗赞羔羊，景行维贤，克念作圣，
德建名立，形端表正，空谷传声，虚堂习听，
祸因恶积，福缘善庆，尺璧非宝，寸阴是竟，
资父事君，曰严与敬，孝当竭力，忠则尽命，
临深履薄，夙兴温清，似兰斯馨，如松之盛，
川流不息，渊澄取映，容止若思，言辞安定，
笃初诚美，慎终宜令，荣业所基，籍甚无竟，
学优登仕，摄职从政，存以甘棠，去而益咏，
乐殊贵贱，礼别尊卑，上和下睦，夫唱妇随，
外受傅训，入奉母仪，诸姑伯叔，犹子比儿，
孔怀兄弟，同气连枝，交友投分，切磨箴规，
仁慈隐恻，造次弗离，节义廉退，颠沛匪亏，
性静情逸，心动神疲，守真志满，逐物意移，
坚持雅操，好爵自縻，都邑华夏，东西二京，
背邙面洛，浮渭据泾，宫殿盘郁，楼观飞惊，
图写禽兽，画彩仙灵，丙舍傍启，甲帐对楹，
肆筵设席，鼓瑟吹笙，升阶纳陛，弁转疑星，
右通广内，左达承明，既集坟典，亦聚群英，
杜稿钟隶，漆书壁经，府罗将相，路侠槐卿，
户封八县，家给千兵，高冠陪辇，驱毂振缨，
世禄侈富，车驾肥轻，策功茂实，勒碑刻铭，
磻溪伊尹，佐时阿衡，奄宅曲阜，微旦孰营，
桓公匡合，济弱扶倾，绮回汉惠，说感武丁，
俊乂密勿，多士寔宁，晋楚更霸，赵魏困横，
假途灭虢，践土会盟，何遵约法，韩弊烦刑，
起翦頽牧，用军最精，宣威沙漠，驰誉丹青，
九州禹迹，百郡秦并，岳宗泰岱，禅主云亭，
雁门紫塞，鸡田赤城，昆池碣石，巨野洞庭，
旷远绵邈，岩岫杳冥，治本于农，务兹稼穡，
俶载南亩，我艺黍稷，税熟贡新，劝赏黜陟，
孟轲敦素，史鱼秉直，庶几中庸，劳谦谨敕，
聆音察理，鉴貌辨色，贻厥嘉猷，勉其祗植，
省躬讥诫，宠增抗极，殆辱近耻，林皋幸即，
两疏见机，解组谁逼，索居闲处，沉默寂寥，
求古寻论，散虑逍遙，欣奏累遣，戚谢欢招，
渠荷的历，园莽抽条，枇杷晚翠，梧桐早凋，
陈根委翳，落叶飘摇，游鹍独运，凌摩絳霄，
耽读玩市，寓目囊箱，易輶攸畏，属耳垣墙，
具膳饭餐，适口充肠，饱饫烹宰，饥厌糟糠，
亲戚故旧，老少异粮，妾御绩纺，侍巾帷房，
纨扇圆絜，银烛炜煌，昼眠夕寐，蓝筭象床，
弦歌酒宴，接杯举觞，矫手顿足，悦豫且康，
嫡后嗣续，祭祀烝尝，稽颡再拜，悚惧恐惶，
笺牒简要，顾答审详，骸垢想浴，执热愿凉，
驴骡犊特，駸跃超骧，诛斩賊盜，捕获叛亡，
布射辽丸，嵇琴阮啸，恬笔伦纸，钩巧任钓，
释纷利俗，并皆佳妙，毛施淑姿，工顰妍笑，

年矢每催，曦晖朗曜，璇玑悬斡，晦魄环照，
指薪修祜，永绥吉劭，矩步引领，俯仰廊庙，
束带矜庄，徘徊瞻眺，孤陋寡闻，愚蒙等诮，
谓语助者，焉哉乎也。

Demo - Other Unicode characters

Most of the following is from <http://www.w3.org/2001/06/utf-8-test/UTF-8-demo.html> :

Olde English long 's'

Paradife Loſt

Mathematics and Sciences

ƒ E·da = Q, n → ∞, $\sum f(i) = \prod g(i)$, $\forall x \in \mathbb{R}$: $[x] = -[-x]$, $\alpha \wedge \neg\beta = \neg(\neg\alpha \vee \beta)$,
 $\mathbb{N} \subseteq \mathbb{N}_0 \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R} \subset \mathbb{C}$, $\perp < a \neq b \equiv c \leq d \ll T \Rightarrow (A \Leftrightarrow B)$,
 $2H_2 + O_2 \rightleftharpoons 2H_2O$, $R = 4.7 \text{ k}\Omega$, $\emptyset 200 \text{ mm}$

Linguistics and dictionaries

ði intə'næʃənəl fə'nɛtik əsouəsi'eɪʃn
Y [Ypsilɔn], Yen [jɛn], Yoga ['jo:gə]

APL

((V\|V)=ιρV)/V←,V ⊢←ι→ρΔ∇⊃~[][]

Typography

- ‘single’ and “double” quotes
- Curly apostrophes: “We’ve been here”
- Latin-1 apostrophe and accents: ' ``
- „deutsche“ „Anführungszeichen“
- †, ‡, ‰, •, 3–4, —, –5/+5, ™, ...
- ASCII safety test: 1|||, 0OD, 8B
- the euro symbol: 14.95 €

Greek (in Polytonic)

The Greek anthem

Σὲ γνωρίζω ἀπὸ τὴν κόψη
τοῦ σπαθιοῦ τὴν τρομερή,
σὲ γνωρίζω ἀπὸ τὴν ὅψη
ποὺ μὲ βία μετράει τὴ γῆ.

Απ' τὰ κόκκαλα βγαλμένη
τῶν Ἑλλήνων τὰ ιερά
καὶ σὰν πρῶτα ἀνδρειωμένη
χαῖρε, ὦ χαῖρε, Ἐλευθεριά!

From a speech of Demosthenes in the 4th century BC
Οὐχὶ ταύτα παρίσταται μοι γιγνώσκειν, ὡς ἄνδρες Ἀθηναῖοι,
ὅταν τ' εἰς τὰ πράγματα ἀποβλέψω καὶ ὅταν πρὸς τοὺς
λόγους οὓς ἀκούω· τοὺς μὲν γάρ λόγους περὶ τοῦ
τιμωρήσασθαι Φίλιππον ὄρῳ γιγνομένους, τὰ δὲ πράγματ'
εἰς τοῦτο προήκοντα, ὥσθ' ὅπως μὴ πεισόμεθ' αὐτοὶ
πρότερον κακῶς σκέψασθαι δέον. οὐδέν οὖν ἄλλο μοι δοκοῦσιν
οἱ τὰ τοιαῦτα λέγοντες ἢ τὴν ύποθεσιν, περὶ ἣς βουλεύεσθαι,
οὐχὶ τὴν οὖσαν παριστάντες ὑμῖν ἀμαρτάνειν. ἔγὼ δέ, ὅτι μέν
ποτ' ἔξην τῇ πόλει καὶ τὰ αὐτῆς ἔχειν ἀσφαλῶς καὶ Φίλιππον
τιμωρήσασθαι, καὶ μάλ' ἀκριβῶς οἶδα· ἐπ' ἔμοῦ γάρ, οὐ πάλαι
γέγονεν ταῦτ' ἀμφότερα· νῦν μέντοι πέπεισμαι τοῦθ' ἱκανὸν
προλαβεῖν ήμῖν εἴναι τὴν πρώτην, ὅπως τοὺς συμμάχους

σώσομεν. έὰν γὰρ τοῦτο βεβαίως ὑπάρξῃ, τότε καὶ περὶ τοῦ τίνα τιμωρήσεται τις καὶ ὃν τρόπον ἔξεσται σκοπεῖν· πρὶν δὲ τὴν ἀρχὴν ὄρθως ὑποθέσθαι, μάταιον ἡγοῦμαι περὶ τῆς τελευτῆς ὄντινοῦν ποιεῖσθαι λόγον.

Δημοσθένους, Γ' Ὁλυνθιακὸς

Georgian

From a Unicode conference invitation

გთხოვთ ახლავე გაიაროთ რეგისტრაცია Unicode-ის მეათე საერთაშორისო კონფერენციაზე დასასწრებად, რომელიც გაიმართება 10-12 მარტს, ქ. მაიცხი, გერმანიაში. კონფერენცია შეპკრებს ერთად მსოფლიოს ექსპერტებს ისეთ დარგებში როგორიცაა ინტერნეტი და Unicode-ი, ინტერნაციონალიზაცია და ლოკალიზაცია, Unicode-ის გამოყენება ოპერაციულ სისტემებსა, და გამოყენებით პროგრამებში, შრიფტებში, ტექსტების დამუშავებასა და მრავალენოვან კომპიუტერულ სისტემებში.

Russian

From a Unicode conference invitation

Зарегистрируйтесь сейчас на Десятую Международную Конференцию по Unicode, которая состоится 10-12 марта 1997 года в Майнце в Германии. Конференция соберет широкий круг экспертов по вопросам глобального Интернета и Unicode, локализации и интернационализации, воплощению и применению Unicode в различных операционных системах и программных приложениях, шрифтах, верстке и многоязычных компьютерных системах.

Thai (UCS Level 2)

Excerpt from a poetry on The Romance of The Three Kingdoms (a Chinese classic 'San Gua')

◎ ແຜ່ນດີນເຂັ້ມສົ່ງໂຄຣມແສນສັງເວົຊ ພຣະປາກເກສກອງນູ້ກຸ້ຂຶ້ນໄໝ່
ສືບສອງກັ້ວຍກັ້ວຍກັ້ວຍກັ້ວຍກັ້ວຍກັ້ວຍກັ້ວຍ
ທຽບນັບຄືອັນທີ່ເປັນທີ່ພື້ນ ບ້ານເມືອງຈິງວິປະຕິບເປັນນັກຫາ
ໄຊຈົ່ນເຮັກທັພທີ່ຫວ່າມີເມືອງມາ ໂມາຍຈະຝ່າມດ້ວຍຕົວສຳຄັນ
ເໜືອນຂັບໄສໄລ່ເສື່ອຈາກເຄຫາ ວັບໝາມປໍາເຂົ້າມາເລຍອາສັນ
ຝ່າຍອ້ອງອຸ້ນຍຸແຍກໃຫ້ແຕກກັນ ໃຊ້ສາວນັ້ນເປັນໜັນນັ້ນຊື່ນໜາວໃຈ
ພລັນລີຈູຍກູຍກູຍກູຍກູຍກູຍກູຍກູຍກູຍກູຍກູຍກູຍກູຍກູຍກູຍ
ຕ້ອງນຽມາຟ້າພັນຈະບຣລັຍ ຖ້າຫາໄຄຣຄໍາຫຼັກບຣລັງກໍ ພ

Ethiopian

Proverbs in the Amharic language

ሰላም አይታረስ የንሥ አይከለስ፡፡
በዚ ካለኝ እንዲአዲቱ በቁመጥኝ፡፡
በተ የለበሩ ቅዱጥና ነው፡፡
ዶሁ በካልሙ ቅዱ ባይጠጣ ጉማት በገደለው፡፡
የእና መለምታ በቅበ አይታገም፡፡
አይተ በበት ዘዋጥሙ፡፡
በተረጋጋሙ ይደረግማሙ፡፡
ቀበ በቀበ፡ ደንቃሳል በለጻና ይረዳል፡፡
ድር በምብር እንበት የስር፡፡
ስወ እንደበት እንደ እንደ ተረበቱ አይተዳደርም፡፡

እናገር የከራተውን ጉርጻ ማዕዘዎች አይደርም::
የነረበት ልማ፡ በያየት ይስቀ ባየቤት ያጠልቸ፡
ሥራ ከመኖታት ልጻን እኩታት፡፡
ዓባይ ማደረግ የለው፡ የንድ ይዘ ይዘረል፡፡
የእለላም እና መካ የአዋሽ እና የርክ፡፡
ተኑጂለ በተፏ ተመልስ ባኅ፡፡
ወዳጅሁ ማር በሆነ መረጃሁ አጥላለው፡፡
እናገሩን በፍረሰኝ ለክ አርጋ፡፡

Runes

HM կրթ թե՛ր հմ ԵՆԱՄ Ի՛ թբմ ՐԵՒՄ ՒՐԵՊՄԵՐՈՒՄ ՊԻ՛ ԹԵ ՊՄՅԵ

(Old English, which transcribed into Latin reads 'He cwaeth that he bude thaem lande northweardum with tha Westsae.' and means 'He said that he lived in the northern land near the Western Sea.'

Braille

Greetings in various languages

Hello world. Καλημέρα κόσμε. ハローワールド。

Equivalent codepages

'Codepage'	mb_encoding	Languages
win-1252	windows-1252	Western European
win-1251	windows-1251	Cyrillic: Russian, Bulgarian, Byelorussian, Macedonian, Serbian, Ukrainian
iso-8859-2	iso-8859-2	Central and Eastern Europe: Czech, Hungarian, Polish, Romanian
iso-8859-4	iso-8859-4	Baltic: Latvian, Lithuanian, Estonian, Greenlandic
iso-8859-7	iso-8859-7	Greek
iso-8859-9	iso-8859-9	Turkish
UTF-8	UTF-8	(ALL)
BIG5	BIG-5	Chinese Traditional (Hong Kong)
GBK	CP936	Chinese Simplified (China)
SHIFT_JIS	SJIS	Japanese
UHC	UHC	Korean

Equivalent encodings

The following list gives equivalent encodings (different names for the same encoding), one per line.

- US-ASCII, ASCII, ISO646-US, ISO_646.IRV:1991, ISO-IR-6, ANSI_X3.4-1968, ANSI_X3.4-1986, CP367, IBM367, US, csASCII
- UCS-2, ISO-10646-UCS-2, csUnicode
- UCS-2BE, UNICODEBIG, UNICODE-1-1, csUnicode11
- UCS-2LE, UNICODELITTLE
- UCS-4, ISO-10646-UCS-4, csUCS4
- UTF-7, UNICODE-1-1-UTF-7, csUnicode11UTF7
- ISO-8859-1, ISO_8859-1, ISO_8859-1:1987, ISO-IR-100, CP819, IBM819, LATIN1, L1, csISOLatin1, ISO8859-1
- ISO-8859-2, ISO_8859-2, ISO_8859-2:1987, ISO-IR-101, LATIN2, L2, csISOLatin2, ISO8859-2
- ISO-8859-3, ISO_8859-3, ISO_8859-3:1988, ISO-IR-109, LATIN3, L3, csISOLatin3, ISO8859-3
- ISO-8859-4, ISO_8859-4, ISO_8859-4:1988, ISO-IR-110, LATIN4, L4, csISOLatin4, ISO8859-4
- ISO-8859-5, ISO_8859-5, ISO_8859-5:1988, ISO-IR-144, CYRILLIC, csISOLatinCyrillic, ISO8859-5
- ISO-8859-6, ISO_8859-6, ISO_8859-6:1987, ISO-IR-127, ECMA-114, ASMO-708, ARABIC, csISOLatinArabic, ISO8859-6
- ISO-8859-7, ISO_8859-7, ISO_8859-7:1987, ISO-IR-126, ECMA-118, ELOT_928, GREEK8, GREEK, csISOLatinGreek, ISO8859-7
- ISO-8859-8, ISO_8859-8, ISO_8859-8:1988, ISO-IR-138, HEBREW, csISOLatinHebrew, ISO8859-8
- ISO-8859-9, ISO_8859-9, ISO_8859-9:1989, ISO-IR-148, LATIN5, L5, csISOLatin5, ISO8859-9
- ISO-8859-10, ISO_8859-10, ISO_8859-10:1992, ISO-IR-157, LATIN6, L6, csISOLatin6, ISO8859-10
- ISO-8859-13, ISO_8859-13, ISO-IR-179, LATIN7, L7, ISO8859-13
- ISO-8859-14, ISO_8859-14, ISO_8859-14:1998, ISO-IR-199, LATIN8, L8, ISO-CELTIC, ISO8859-14
- ISO-8859-15, ISO_8859-15, ISO_8859-15:1998, ISO-IR-203, ISO8859-15
- ISO-8859-16, ISO_8859-16, ISO_8859-16:2000, ISO-IR-226, ISO8859-16
- KOI8-R, csKOI8R
- CP1250, WINDOWS-1250, MS-EE
- CP1251, WINDOWS-1251, MS-CYRL
- CP1252, WINDOWS-1252, MS-ANSI
- CP1253, WINDOWS-1253, MS-GREEK
- CP1254, WINDOWS-1254, MS-TURK
- CP1255, WINDOWS-1255, MS-HEBR
- CP1256, WINDOWS-1256, MS-ARAB
- CP1257, WINDOWS-1257, WINBALTRIM
- CP1258, WINDOWS-1258
- CP850, IBM850, 850, csPC850Multilingual
- CP862, IBM862, 862, csPC862LatinHebrew
- CP866, IBM866, 866, csIBM866
- MACINTOSH, MAC, csMacintosh
- HP-ROMAN8, ROMAN8, R8, csHPRoman8

- CP1133, IBM-CP1133
- TIS-620, TIS620, TIS620-0, TIS620.2529-1, TIS620.2533-0, TIS620.2533-1, ISO-IR-166
- CP874, WINDOWS-874
- VISCII, VISCII1.1-1, csVISCII
- TCVN, TCVN-5712, TCVN5712-1, TCVN5712-1:1993
- JIS_C6220-1969-RO, ISO646-JP, ISO-IR-14, JP, csISO14JISC6220ro
- JIS_X0201, JISX0201-1976, X0201, csHalfWidthKatakana
- JIS_X0208, JIS_X0208-1983, JIS_X0208-1990, JIS0208, X0208, ISO-IR-87, JIS_C6226-1983, csISO87JISX0208
- JIS_X0212, JIS_X0212.1990-0, JIS_X0212-1990, X0212, ISO-IR-159, csISO159JISX02121990
- GB_1988-80, ISO646-CN, ISO-IR-57, CN, csISO57GB1988
- GB_2312-80, ISO-IR-58, csISO58GB231280, CHINESE
- ISO-IR-165, CN-GB-ISOIR165
- KSC_5601, KS_C_5601-1987, KS_C_5601-1989, ISO-IR-149, csKSC56011987, KOREAN
- EUC-JP, EUCJP, Extended_UNIX_Code_Packed_Format_for_Japanese, csEUCPkdFmtJapanese
- SHIFT_JIS, SHIFT-JIS, SJIS, MS_KANJI, csShiftJIS
- ISO-2022-JP, csISO2022JP
- ISO-2022-JP-2, csISO2022JP2
- EUC-CN, EUCCN, GB2312, CN-GB, csGB2312
- GBK, CP936
- ISO-2022-CN, csISO2022CN
- HZ, HZ-GB-2312
- EUC-TW, EUCTW, csEUCTW
- BIG5, BIG-5, BIG-FIVE, BIGFIVE, CN-BIG5, csBig5
- BIG5-HKSCS, BIG5HKSCS
- EUC-KR, EUCKR, csEUCKR
- CP949, UHC
- JOHAB, CP1361
- ISO-2022-KR, csISO2022KR

From : <http://www.datamystic.com/textpipe/htmlhelp/conversion%20filters/fromUnicode.htm>

iconv - Supported codepages

The following codepages are supported by the iconv function:

European languages

- ASCII
- ISO-8859-{1,2,3,4,5,7,9,10,13,14,15,16}
- KOI8-R, KOI8-U, KOI8-RU
- CP{1250,1251,1252,1253,1254,1257}
- CP{850,866}
- Mac{Roman,CentralEurope,Iceland,Croatian,Romania}
- Mac{Cyrillic,Ukraine,Greek,Turkish}
- Macintosh

Semitic languages

- ISO-8859-{6,8}
- CP{1255,1256}
- CP862
- Mac{Hebrew,Arabic}

Japanese

- EUC-JP, SHIFT_JIS, CP932, ISO-2022-JP, ISO-2022-JP-2, ISO-2022-JP-1

Chinese

- EUC-CN, HZ, GBK, GB18030, EUC-TW, BIG5, CP950, BIG5-HKSCS, ISO-2022-CN, ISO-2022-CN-EXT

Korean

- EUC-KR, CP949, ISO-2022-KR, JOHAB

Armenian

- ARMSCII-8

Georgian

- Georgian-Academy, Georgian-PS

Tajik

- KOI8-T

Thai

- TIS-620, CP874, MacThai

Laotian

- MuleLao-1, CP1133

Vietnamese

- VISCI, TCVN, CP1258

Platform specifics

- HP-ROMAN8, NEXTSTEP

Full Unicode

- UTF-8
- UCS-2, UCS-2BE, UCS-2LE
- UCS-4, UCS-4BE, UCS-4LE
- UTF-16, UTF-16BE, UTF-16LE
- UTF-32, UTF-32BE, UTF-32LE

- UTF-7

See <http://www.gnu.org/software/libiconv/>

mbstring - Supported codepages

Currently the following character encodings are supported by the mbstring module. Any of those Character encodings can be specified in the encoding parameter of mbstring functions.

- UCS-4, UCS-4BE, UCS-4LE
- UCS-2, UCS-2BE, UCS-2LE
- UTF-32, UTF-32BE, UTF-32LE
- UTF-16, UTF-16BE, UTF-16LE
- UTF-7, UTF7-IMAP,
- UTF-8
- ASCII
- EUC-JP
- SJIS
- eucJP-win, SJIS-win
- ISO-2022-JP
- JIS
- ISO-8859-1, ISO-8859-2, ISO-8859-3, ISO-8859-4, ISO-8859-5, ISO-8859-6, ISO-8859-7, ISO-8859-8, ISO-8859-9, ISO-8859-10, ISO-8859-13, ISO-8859-14, ISO-8859-15
- byte2be, byte2le
- byte4be, byte4le
- BASE64
- HTML-ENTITIES
- 7bit
- 8bit
- EUC-CN
- CP936
- HZ
- EUC-TW
- CP950
- BIG-5
- EUC-KR
- UHC (CP949)
- ISO-2022-KR
- Windows-1251 (CP1251)
- Windows-1252 (CP1252)
- CP866 (IBM866)
- KOI8-R

For the module to be enabled, php must have been compiled with `--enable-mbstring`

See <http://uk.php.net/manual/en/ref.mbstring.php>

Unicode character planes

As of Unicode 5.1, The BMP includes the following scripts:

- [Basic Latin](#) (0000–007F)
- [Latin-1 Supplement](#) (0080–00FF)
- [Latin Extended-A](#) (0100–017F)
- [Latin Extended-B](#) (0180–024F)
- [IPA Extensions](#) (0250–02AF)
- [Spacing Modifier Letters](#) (02B0–02FF)
- [Combining Diacritical Marks](#) (0300–036F)
- [Greek and Coptic](#) (0370–03FF)
- [Cyrillic](#) (0400–04FF)
- [Cyrillic Supplement](#) (0500–052F)
- [Armenian](#) (0530–058F)
- [Hebrew](#) (0590–05FF)
- [Arabic](#) (0600–06FF)
- [Syriac](#) (0700–074F)
- [Arabic Supplement](#) (0750–077F)
- [Thaana](#) (0780–07BF)
- [N'Ko \(Mandenkan\)](#) (07C0–07FF)
- [Arabic Extended-A](#) (08A0–08FF)
- Indic scripts:
 - [Devanagari](#) (0900–097F)
 - [Bengali](#) (0980–09FF)
 - [Gurmukhi](#) (0A00–0A7F)
 - [Gujarati](#) (0A80–0AFF)
 - [Oriya](#) (0B00–0B7F)
 - [Tamil](#) (0B80–0BFF)
 - [Telugu](#) (0C00–0C7F)
 - [Kannada](#) (0C80–0CFF)
 - [Malayalam](#) (0D00–0D7F)
 - [Sinhala](#) (0D80–0DFF)
- [Thai](#) (0E00–0E7F)
- [Lao](#) (0E80–0EFF)
- [Tibetan](#) (0F00–0FFF)
- [Burmese \(Myanmar\)](#) (1000–109F)
- [Georgian](#) (10A0–10FF)
- [Hangul Jamo](#) (1100–11FF)
- [Ethiopic](#) (1200–137F)
- [Ethiopic Supplement](#) (1380–139F)
- [Cherokee](#) (13A0–13FF)
- [Unified Canadian Aboriginal Syllabics](#) (1400–167F)
- [Ogham](#) (1680–169F)
- [Runic](#) (16A0–16FF)
- Philippine scripts:
 - [Tagalog](#) (1700–171F)
 - [Hanunóo](#) (1720–173F)
 - [Buhid](#) (1740–175F)
 - [Tagbanwa](#) (1760–177F)
- [Khmer](#) (1780–17FF)
- [Mongolian](#) (1800–18AF)
- [Limbu](#) (1900–194F)
- [Tai Le](#) (1950–197F)
- [New Tai Lue](#) (1980–19DF)
- [Khmer Symbols](#) (19E0–19FF)
- [Buginese](#) (1A00–1A1F)
- [Balinese](#) (1B00–1B7F)

- Sundanese (1B80-1BBF)
- Lepcha (Rong) (1C00-1C4F)
- Ol Chiki (Santali / Ol Cemet') (1C50-1C7F)
- Phonetic Extensions (1D00-1D7F)
- Phonetic Extensions Supplement (1D80-1DBF)
- Combining Diacritical Marks Supplement (1DC0-1DFF)
- Latin Extended Additional (1E00-1EFF)
- Greek Extended (1F00-1FFF)
- Symbols:
 - General Punctuation (2000-206F)
 - Superscripts and Subscripts (2070-209F)
 - Currency Symbols (20A0-20CF)
 - Combining Diacritical Marks for Symbols (20D0-20FF)
 - Letterlike Symbols (2100-214F)
 - Number Forms (2150-218F)
 - Arrows (2190-21FF)
 - Mathematical Operators (2200-22FF)
 - Miscellaneous Technical (2300-23FF)
 - Control Pictures (2400-243F)
 - Optical Character Recognition (2440-245F)
 - Enclosed Alphanumerics (2460-24FF)
 - Box Drawing (2500-257F)
 - Block Elements (2580-259F)
 - Geometric Shapes (25A0-25FF)
 - Miscellaneous Symbols (2600-26FF)
 - Dingbats (2700-27BF)
 - Miscellaneous Mathematical Symbols-A (27C0-27EF)
 - Supplemental Arrows-A (27F0-27FF)
 - Braille Patterns (2800-28FF)
 - Supplemental Arrows-B (2900-297F)
 - Miscellaneous Mathematical Symbols-B (2980-29FF)
 - Supplemental Mathematical Operators (2A00-2AFF)
 - Miscellaneous Symbols and Arrows (2B00-2BFF)
- Glagolitic (2C00-2C5F)
- Latin Extended-C (2C60-2C7F)
- Coptic (2C80-2CFF)
- Georgian Supplement (2D00-2D2F)
- Tifinagh (2D30-2D7F)
- Ethiopic Extended (2D80-2DDF)
- Cyrillic Extended-A (2DE0-2DFF)
- Supplemental Punctuation (2E00-2E7F)
- CJK Radicals Supplement (2E80-2EFF)
- Kangxi Radicals (2F00-2FDF)
- Ideographic Description Characters (2FF0-2FFF)
- CJK Symbols and Punctuation (3000-303F)
- Hiragana (3040-309F)
- Katakana (30A0-30FF)
- Bopomofo (3100-312F)
- Hangul Compatibility Jamo (3130-318F)
- Kanbun (3190-319F)
- Bopomofo Extended (31A0-31BF)
- CJK Strokes (31C0-31EF)
- Katakana Phonetic Extensions (31F0-31FF)
- Enclosed CJK Letters and Months (3200-32FF)
- CJK Compatibility (3300-33FF)
- CJK Unified Ideographs Extension A (3400-4DBF)
- Yijing Hexagram Symbols (4DC0-4DFF)
- CJK Unified Ideographs (4E00-9FFF)
- Yi Syllables (A000-A48F)

- Yi Radicals (A490–A4CF)
- Vai (A500–A63F)
- Cyrillic Extended-B (A640–A69F)
- Modifier Tone Letters (A700–A71F)
- Latin Extended-D (A720–A7FF)
- Syloti Nagri (A800–A82F)
- Phags-pa (A840–A87F)
- Saurashtra (A880–A8DF)
- Kayah Li (A900–A92F)
- Rejang (A930–A95F)
- Cham (AA00–AA5F)
- Hangul Syllables (AC00–D7AF)
- High Surrogates (D800–DB7F)
- High Private Use Surrogates (DB80–DBFF)
- Low Surrogates (DC00–DFFF)
- Private Use Area (E000–F8FF)
- CJK Compatibility Ideographs (F900–FAFF)
- Alphabetic Presentation Forms (FB00–FB4F)
- Arabic Presentation Forms-A (FB50–FDFF)
- Variation Selectors (FE00–FE0F)
- Vertical Forms (FE10–FE1F)
- Combining Half Marks (FE20–FE2F)
- CJK Compatibility Forms (FE30–FE4F)
- Small Form Variants (FE50–FE6F)
- Arabic Presentation Forms-B (FE70–FEFF)
- Halfwidth and Fullwidth Forms (FF00–FFEF)
- Specials (FFF0–FFFF)

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F															
00	C0 Controls			Basic Latin			C1 Controls			Latin 1 Supplement																				
01	Latin Extended-A								Latin Extended-B																					
02	Latin Extended-B				IPA Extensions					Spacing Modifiers																				
03	Combining Diacritics								Greek																					
04	Cyrillic																													
05	Cyrillic Sup.		Armenian								Hebrew																			
06	Arabic																													
07	Syriac		Arabic Sup.			Thaana			N'Ko																					
08	Samaritan		(Mandaic)		???	???	???	???	(Arabic Extended-A)																					
09	Devanagari																													
0A	Gurmukhi																													
0B	Oriya																													
0C	Telugu																													
0D	Malayalam																													
0E	Thai																													
0F	Tibetan																													
10	Myanmar								Georgian																					
11	Hangul Jamo																													
12	Ethiopic																													
13	Ethiopic				Eth. Sup.					Cherokee																				
14	Unified Canadian Aboriginal Syllabics																													
15	Unified Canadian Aboriginal Syllabics																													
16	Unified Canadian Aboriginal Syllabics					Ogham			Runic																					
17	Tagalog	Hanunóo	Buhid	Tagbanwa				Khmer																						

18	Mongolian			Canadian Syllabics Ext.																	
19	Limbu	Tai Le	New Tai Lue			Khmer															
1A	Buginese	Tai Tham			???	???	???	???	???												
1B	Balinese			Sundanese			(Batak)														
1C	Lepcha	Ol Chiki	???	???	???	???	(Sund)	Vedic Extensions													
1D	Phonetic Extensions			Phonetic Ext. Sup.			Comb. Diacritics Sup.														
1E	Latin Extended Additional																				
1F	Greek Extended																				
20	General Punctuation		Subs/Supers		Currency		Diac. Symbs.														
21	Letterlike Symbols		Number Forms			Arrow															
22	Mathematical Symbols																				
23	Miscellaneous Technical																				
24	Control Pictures	OCR	Enclosed Alphanumerics																		
25	Box Drawing			Blocks		Geometric Shapes															
26	Miscellaneous Symbols																				
27	Dingbats			MiscMathA			Arrows														
28	Braille Patterns																				
29	Supplemental Arrows-B			Misc. Mathematical Symbols-B																	
2A	Supplemental Mathematical Operators																				
2B	Miscellaneous Symbols and Arrows																				
2C	Glagolitic	Latn Ext-C		Coptic																	
2D	Georgian Sup.	Tifinagh	Ethiopic Extended			Cyr Ext-A															
2E	Supplemental Punctuation			CJK Radicals																	
2F	Kangxi Radicals			???			IDC														
30	CJK Syms. & Punct.	Hiragana		Katakana																	
31	Bopomofo	Hangul Compatibility Jamo	Kbn	Bpmf Ext.	CJK Strokes	Kk.															
32	Enclosed CJK Letters & Months																				
33	CJK Compatibility																				
34	CJK Unified Ideographs Extension A																				
35	CJK Unified Ideographs Extension A																				
36	CJK Unified Ideographs Extension A																				
37	CJK Unified Ideographs Extension A																				
38	CJK Unified Ideographs Extension A																				
39	CJK Unified Ideographs Extension A																				
3A	CJK Unified Ideographs Extension A																				
3B	CJK Unified Ideographs Extension A																				
3C	CJK Unified Ideographs Extension A																				
3D	CJK Unified Ideographs Extension A																				
3E	CJK Unified Ideographs Extension A																				
3F	CJK Unified Ideographs Extension A																				
40	CJK Unified Ideographs Extension A																				
41	CJK Unified Ideographs Extension A																				
42	CJK Unified Ideographs Extension A																				
43	CJK Unified Ideographs Extension A																				
44	CJK Unified Ideographs Extension A																				
45	CJK Unified Ideographs Extension A																				
46	CJK Unified Ideographs Extension A																				
47	CJK Unified Ideographs Extension A																				
48	CJK Unified Ideographs Extension A																				

49	CJK Unified Ideographs Extension A	
4A	CJK Unified Ideographs Extension A	
4B	CJK Unified Ideographs Extension A	
4C	CJK Unified Ideographs Extension A	
4D	CJK Unified Ideographs Extension A	Yijing Hexagrams
4E	CJK Unified Ideographs	
4F	CJK Unified Ideographs	
50	CJK Unified Ideographs	
51	CJK Unified Ideographs	
52	CJK Unified Ideographs	
53	CJK Unified Ideographs	
54	CJK Unified Ideographs	
55	CJK Unified Ideographs	
56	CJK Unified Ideographs	
57	CJK Unified Ideographs	
58	CJK Unified Ideographs	
59	CJK Unified Ideographs	
5A	CJK Unified Ideographs	
5B	CJK Unified Ideographs	
5C	CJK Unified Ideographs	
5D	CJK Unified Ideographs	
5E	CJK Unified Ideographs	
5F	CJK Unified Ideographs	
60	CJK Unified Ideographs	
61	CJK Unified Ideographs	
62	CJK Unified Ideographs	
63	CJK Unified Ideographs	
64	CJK Unified Ideographs	
65	CJK Unified Ideographs	
66	CJK Unified Ideographs	
67	CJK Unified Ideographs	
68	CJK Unified Ideographs	
69	CJK Unified Ideographs	
6A	CJK Unified Ideographs	
6B	CJK Unified Ideographs	
6C	CJK Unified Ideographs	
6D	CJK Unified Ideographs	
6E	CJK Unified Ideographs	
6F	CJK Unified Ideographs	
70	CJK Unified Ideographs	
71	CJK Unified Ideographs	
72	CJK Unified Ideographs	
73	CJK Unified Ideographs	
74	CJK Unified Ideographs	
75	CJK Unified Ideographs	
76	CJK Unified Ideographs	
77	CJK Unified Ideographs	
78	CJK Unified Ideographs	
79	CJK Unified Ideographs	

7A	CJK Unified Ideographs					
7B	CJK Unified Ideographs					
7C	CJK Unified Ideographs					
7D	CJK Unified Ideographs					
7E	CJK Unified Ideographs					
7F	CJK Unified Ideographs					
80	CJK Unified Ideographs					
81	CJK Unified Ideographs					
82	CJK Unified Ideographs					
83	CJK Unified Ideographs					
84	CJK Unified Ideographs					
85	CJK Unified Ideographs					
86	CJK Unified Ideographs					
87	CJK Unified Ideographs					
88	CJK Unified Ideographs					
89	CJK Unified Ideographs					
8A	CJK Unified Ideographs					
8B	CJK Unified Ideographs					
8C	CJK Unified Ideographs					
8D	CJK Unified Ideographs					
8E	CJK Unified Ideographs					
8F	CJK Unified Ideographs					
90	CJK Unified Ideographs					
91	CJK Unified Ideographs					
92	CJK Unified Ideographs					
93	CJK Unified Ideographs					
94	CJK Unified Ideographs					
95	CJK Unified Ideographs					
96	CJK Unified Ideographs					
97	CJK Unified Ideographs					
98	CJK Unified Ideographs					
99	CJK Unified Ideographs					
9A	CJK Unified Ideographs					
9B	CJK Unified Ideographs					
9C	CJK Unified Ideographs					
9D	CJK Unified Ideographs					
9E	CJK Unified Ideographs					
9F	CJK Unified Ideographs					
A0	Yi					
A1	Yi					
A2	Yi					
A3	Yi					
A4	Yi	Yi Radicals	Lisu			
A5	Vai					
A6	Vai	Cyrillic Extended-B	Bamum			
A7	Mod. Tone		Latin Extended-D			
A8	Syloti Nagri	IndNo	Phags-pa	Saurashtra	Deva Ext.	
A9	Kayah Li	Rejang	HangulA	Javanese	???	???
AA	Cham		Mymr ExtA	Tai Viet	(Mtei Ext)	

	(Ethiopic Ext-A)	(Warang Citi)	???	???	¿Beria?	Meetei Mayek
AC		Hangul Syllables				
AD		Hangul Syllables				
AE		Hangul Syllables				
AF		Hangul Syllables				
B0		Hangul Syllables				
B1		Hangul Syllables				
B2		Hangul Syllables				
B3		Hangul Syllables				
B4		Hangul Syllables				
B5		Hangul Syllables				
B6		Hangul Syllables				
B7		Hangul Syllables				
B8		Hangul Syllables				
B9		Hangul Syllables				
BA		Hangul Syllables				
BB		Hangul Syllables				
BC		Hangul Syllables				
BD		Hangul Syllables				
BE		Hangul Syllables				
BF		Hangul Syllables				
C0		Hangul Syllables				
C1		Hangul Syllables				
C2		Hangul Syllables				
C3		Hangul Syllables				
C4		Hangul Syllables				
C5		Hangul Syllables				
C6		Hangul Syllables				
C7		Hangul Syllables				
C8		Hangul Syllables				
C9		Hangul Syllables				
CA		Hangul Syllables				
CB		Hangul Syllables				
CC		Hangul Syllables				
CD		Hangul Syllables				
CE		Hangul Syllables				
CF		Hangul Syllables				
D0		Hangul Syllables				
D1		Hangul Syllables				
D2		Hangul Syllables				
D3		Hangul Syllables				
D4		Hangul Syllables				
D5		Hangul Syllables				
D6		Hangul Syllables				
D7	Hangul Syllables				Hangul Jamo Extended-B	
D8		High-half zone of UTF-16				
D9		High-half zone of UTF-16				
DA		High-half zone of UTF-16				
DB		High-half zone of UTF-16				

DC	Low-half zone of UTF-16	
DD	Low-half zone of UTF-16	
DE	Low-half zone of UTF-16	
DF	Low-half zone of UTF-16	
E0	Private Use Zone	
E1	Private Use Zone	
E2	Private Use Zone	
E3	Private Use Zone	
E4	Private Use Zone	
E5	Private Use Zone	
E6	Private Use Zone	
E7	Private Use Zone	
E8	Private Use Zone	
E9	Private Use Zone	
EA	Private Use Zone	
EB	Private Use Zone	
EC	Private Use Zone	
ED	Private Use Zone	
EE	Private Use Zone	
EF	Private Use Zone	
F0	Private Use Zone	
F1	Private Use Zone	
F2	Private Use Zone	
F3	Private Use Zone	
F4	Private Use Zone	
F5	Private Use Zone	
F6	Private Use Zone	
F7	Private Use Zone	
F8	Private Use Zone	
F9	CJK Compatibility Ideographs	
FA	CJK Compatibility Ideographs	
FB	Alphabetic Pres. Forms	Arabic Presentation Forms A
FC		Arabic Presentation Forms A
FD	Arabic Presentation Forms A	Nonchars. APF-A
FE	Vars. Vert. Half CJKcomp Small	Arabic Presentation Forms B
FF	Halfwidth & Fullwidth Forms	Spec.

From <http://www.unicode.org/roadmaps/bmp/>

ISO 639-1 language codes

The language/country codes recognised by mPDF are configurable in the config_cp.php file.

ISO 639-1 Code	Language	Recognised by mPDF	Country
aa	Afar		
ab	Abkhazian		
af	Afrikaans		
ak	Akan		
sq	Albanian		
am	Amharic		
ar	Arabic	YES	
an	Aragonese		
hy	Armenian		
as	Assamese	YES	
av	Avaric		
ae	Avestan		
ay	Aymara		
az	Azerbaijani		
ba	Bashkir		
bm	Bambara		
eu	Basque		
be	Belarusian		
bn	Bengali	YES	
bh	Bihari		
bi	Bislama		
bo	Tibetan		
bs	Bosnian		
br	Breton		
bg	Bulgarian	YES	
my	Burmese		
ca	Catalan; Valencian	YES	
cs	Czech	YES	
ch	Chamorro		
ce	Chechen		
zh	Chinese	YES	
cv	Chuvash		
kw	Cornish		
co	Corsican		
cr	Cree		
cy	Welsh	YES	
cs	Czech	YES	
da	Danish	YES	
de	German	YES	
dv	Divehi; Dhivehi; Maldivian		
nl	Dutch; Flemish	YES	
dz	Dzongkha		
el	Greek, Modern (1453-)	YES	
en	English	YES	

ISO 639-1 Code	Language	Recognised by mPDF	Country
eo	Esperanto		
et	Estonian	YES	
eu	Basque	YES	
ee	Ewe		
fo	Faroese		
fa	Persian	YES	
fj	Fijian		
fi	Finnish	YES	
fr	French	YES	
ff	Fulah		
ka	Georgian		
de	German	YES	
gd	Gaelic; Scottish Gaelic		
ga	Irish	YES	
gl	Galician		
gv	Manx		
el	Greek, Modern (1453-)	YES	
gu	Gujarati	YES	
ht	Haitian; Haitian Creole		
ha	Hausa		
he	Hebrew	YES	
hz	Herero		
hi	Hindi	YES	
ho	Hiri Motu		
hr	Croatian	YES	
hu	Hungarian	YES	
hy	Armenian		
ig	Igbo		
is	Icelandic	YES	
io	Ido		
ie	Interlingue; Occidental		
ia	Interlingua		
id	Indonesian		
ik	Inupiaq		
is	Icelandic	YES	
it	Italian	YES	
jv	Javanese		
ja	Japanese	YES	
kl	Kalaallisut; Greenlandic	YES	
kn	Kannada	YES	
ks	Kashmiri	YES	
ka	Georgian		
kr	Kanuri		
kk	Kazakh		
km	Central Khmer		
ki	Kikuyu; Gikuyu		
rw	Kinyarwanda		

ISO 639-1 Code	Language	Recognised by mPDF	Country
ky	Kirghiz; Kyrgyz		
kv	Komi		
kg	Kongo		
ko	Korean	YES	
kj	Kuanyama; Kwanyama		
ku	Kurdish		
lo	Lao		
la	Latin		
lv	Latvian	YES	
li	Limburgan; Limburger; Limburgish		
In	Lingala		
lt	Lithuanian	YES	
lb	Luxembourgish; Letzeburgesch		
lu	Luba-Katanga		
lg	Ganda		
mk	Macedonian	YES	
mh	Marshallese		
ml	Malayalam	YES	
mi	Maori		
mr	Marathi		
ms	Malay		
mk	Macedonian	YES	
mg	Malagasy		
mt	Maltese		
mn	Mongolian		
mi	Maori		
ms	Malay		
my	Burmese		
na	Nauru		
nv	Navajo; Navaho		
nr	Ndebele, South; South Ndebele		
nd	Ndebele, North; North Ndebele		
ng	Ndonga		
ne	Nepali	YES	
nl	Dutch; Flemish	YES	
nn	Norwegian Nynorsk		
nb	Bokmål, Norwegian		
no	Norwegian	YES	
ny	Chichewa; Chewa; Nyanja		
oc	Occitan		
oj	Ojibwa		
or	Oriya	YES	
om	Oromo		
os	Ossetian; Ossetic		
pa	Punjabi; Panjabi	YES	
fa	Persian	YES	
pi	Pali		

ISO 639-1 Code	Language	Recognised by mPDF	Country
pl	Polish	YES	
pt	Portuguese	YES	
ps	Pushto; Pashto	YES	
qu	Quechua		
rm	Romansh		
ro	Romanian; Moldavian; Moldovan	YES	
rn	Rundi		
ru	Russian	YES	
sg	Sango		
sa	Sanskrit		
si	Sinhala; Sinhalese		
sk	Slovak	YES	
sl	Slovenian	YES	
se	Northern Sami		
sm	Samoan		
sn	Shona		
sd	Sindhi	YES	-IN, -PK
so	Somali		
st	Sotho, Southern		
es	Spanish; Castilian	YES	
sq	Albanian		
sc	Sardinian		
sr	Serbian	YES	
ss	Swati		
su	Sundanese		
sw	Swahili		
sv	Swedish	YES	
ty	Tahitian		
ta	Tamil	YES	
tt	Tatar		
te	Telugu	YES	
tg	Tajik		
tl	Tagalog		
th	Thai	YES	
bo	Tibetan		
ti	Tigrinya		
to	Tonga (Tonga Islands)		
tn	Tswana		
ts	Tsonga		
tk	Turkmen		
tr	Turkish	YES	
tw	Twi		
ug	Uighur; Uyghur		
uk	Ukrainian	YES	
ur	Urdu	YES	
uz	Uzbek		
ve	Venda		

ISO 639-1 Code	Language	Recognised by mPDF	Country
vi	Vietnamese	YES	
vo	Volapük		
cy	Welsh	YES	
wa	Walloon		
wo	Wolof		
xh	Xhosa		
yi	Yiddish		
yo	Yoruba		
za	Zhuang; Chuang		
zh	Chinese	YES	-TW, -HK, -CN
zu	Zulu		

Useful links

Fonts frequency (browsers)

- <http://www.codestyle.org/css/font-family/sampler-CombinedResults.shtml>

Coverage of FreeSans fonts

- <http://www.gnu.org/software/freefont/coverage.html>

Coverage of DejaVu fonts (enclosed files in downloads)

- <http://sourceforge.net/projects/dejavu/>

Unicode coverage Free Fonts

Fonts with full coverage of Unicode Ranges

Unicode Block	Fonts with full coverage
BASIC LATIN (U+0000-U+007E)	dejavu*, free*
LATIN-1 SUPPLEMENT (U+00A0-U+00FF)	dejavu*, free*
LATIN EXTENDED-A (U+0100-U+017F)	dejavu*, free*
LATIN EXTENDED-B (U+0180-U+024F)	dejavusans(+cond), freeserif
IPA EXTENSIONS (U+0250-U+02AF)	dejavu*, free*
SPACING MODIFIER LETTERS (U+02B0-U+02FF)	free*
COMBINING DIACRITICAL MARKS (U+0300-U+036F)	freesans, freeserif
Special positioning required	
GREEK (U+0370-U+03FF)	dejavusans(+cond)
CYRILLIC (U+0400-U+04FF)	dejavusans(+cond), freeserif
CYRILLIC SUPPLEMENT (U+0500-U+052F)	charissil, quivira
ARMENIAN (U+0530-U+058F)	dejavusans(+cond), freemono, freesans
HEBREW (U+0590-U+05FF)	freeserif, quivira
Special processing required	
ARABIC (U+0600-U+06FF)	sun-exta (>90%)
Special processing required	dejavusans(+cond), xbriyaz, xbzar (>50%)
SYRIAC (U+0700-U+074F)	freesans (>90%)
Special processing required	
ARABIC SUPPLEMENT (U+0750-U+077F)	
Special processing required	
THAANA (U+0780-U+07BF)	mph2bdamase
Special processing required	
N'KO (MANDENKAN) (U+07C0-U+07FF)	dejavusans(+cond) (>90%)
SAMARITAN (U+0800-U+083E)	
DEVANAGARI (U+0900-U+097F)	<i>ind_hi_1_001</i>
Special processing required	
BENGALI (U+0980-U+09FF)	<i>ind_bn_1_001</i>
Special processing required	
GURMUKHI (U+0A00-U+0A7F)	<i>ind_pa_1_001</i>
Special processing required	
GUJARATI (U+0A80-U+0AFF)	<i>ind_gu_1_001</i>
Special processing required	
ORIYA (U+0B00-U+0B7F)	<i>ind_or_1_001</i>
Special processing required	
TAMIL (U+0B80-U+0BFF)	<i>ind_ta_1_001</i>
Special processing required	
TELUGU (U+0C00-U+0C7F)	<i>ind_te_1_001</i>
Special processing required	
KANNADA (U+0C80-U+0CFF)	<i>ind_kn_1_001</i>
Special processing required	
MALAYALAM (U+0D00-U+0D7F)	<i>ind_ml_1_001</i>
Special processing required	
SINHALA (U+0D80-U+0DFF)	kaputaunicode
Special processing required	
THAI (U+0E00-U+0E7F)	bitstreamcyberbit, freeserif, garuda, norasi, quivira
LAO (U+0E80-U+0EFF)	dejavusans(+cond)
TIBETAN (U+0F00-U+0FFF)	tibetanunicode (>90%)
Special processing required	
MYANMAR (U+1000-U+109F)	
Special processing required	
GEORGIAN (U+10A0-U+10FF)	dejavusans(+cond), dejavuserif(+cond)

Unicode Block	Fonts with full coverage
HANGUL JAMO (U+1100-U+11FF)	unbatang
ETHIOPIC (U+1200-U+137F)	abyssinicasil
ETHIOPIC SUPPLEMENT (U+1380-U+139F)	abyssinicasil
CHEROKEE (U+13A0-U+13FF)	aboriginalsans, aboriginalserif, freeserif, mph2bdamase, quivira
UNIFIED CANADIAN ABORIGINAL SYLLABICS (U+1400-U+167F)	aboriginalsans, aboriginalserif
OGHAM (U+1680-U+169F)	aboriginalserif, dejavusans(+cond), quivira
RUNIC (U+16A0-U+16FF)	aboriginalserif, freemono, quivira
TAGALOG (PHILIPPINE) (U+1700-U+171F)	quivira
HANUNOO (PHILIPPINE) (U+1720-U+173F)	freeserif, mph2bdamase, quivira
BUHID (PHILIPPINE) (U+1740-U+175F)	quivira
TAGBANWA (PHILIPPINE) (U+1760-U+177F)	quivira
KHMER (U+1780-U+17FF)	khmerosbattambang
Special processing required	
MONGOLIAN (U+1800-U+18AF)	sun-exta (>90%)
Vertical positioning required	
CANADIAN SYLLABICS (U+18B0-U+18F5)	aboriginalsans, aboriginalserif
LIMBU (U+1900-U+194F)	mph2bdamase
TAI LE (U+1950-U+197F)	freeserif, mph2bdamase
NEW TAI LUE (U+1980-U+19DF)	daibannasilbook (>90%)
KHMER SYMBOLS (U+19E0-U+19FF)	
Special processing required	
BUGINESE (U+1A00-U+1A1F)	freeserif, mph2bdamase
TAI THAM (U+1A20-U+1AAF)	lannaalif
BALINESE (U+1B00-U+1B7F)	
Special processing required	
SUNDANESE (U+1B80-U+1BBF)	sundaneseunicode
LEPCHA (RONG) (U+1C00-U+1C4F)	
OL CHIKI (SANTALI / OL CEMET) (U+1C50-U+1C7F)	
VEDIC EXTENSIONS (U+1CD0-U+1CFF)	
PHONETIC EXTENSIONS (U+1D00-U+1D7F)	quivira
PHONETIC EXTENSIONS SUPPLEMENT (U+1D80-U+1DBF)	charissil, quivira
COMBINING DIACRITICAL MARKS SUPPLEMENT (U+1DC0-U+1DFF)	
Special positioning required	
LATIN EXTENDED ADDITIONAL (U+1E00-U+1EFF)	charissil, quivira
GREEK EXTENDED (U+1F00-U+1FFF)	dejavu*, free*
GENERAL PUNCTUATION (U+2000-U+206F)	bitstreamcyberbit, freemono, symbola
SUPERSCRIPTS AND SUBSCRIPTS (U+2070-U+209F)	bitstreamcyberbit, charissil, dejavu*, free*, quivira
CURRENCY SYMBOLS (U+20A0-U+20CF)	bitstreamcyberbit, quivira
COMBINING MARKS FOR SYMBOLS (U+20D0-U+20FF)	bitstreamcyberbit, freeserif
Special positioning required	
LETTERLIKE SYMBOLS (U+2100-U+214F)	bitstreamcyberbit, quivira
NUMBER FORMS (U+2150-U+218F)	bitstreamcyberbit, quivira
ARROWS (U+2190-U+21FF)	dejavu*, quivira
MATHEMATICAL OPERATORS (U+2200-U+22FF)	bitstreamcyberbit, dejavusans(+cond), freeserif, quivira
MISCELLANEOUS TECHNICAL (U+2300-U+23FF)	bitstreamcyberbit, quivira
CONTROL PICTURES (U+2400-U+243F)	bitstreamcyberbit, quivira
OPTICAL CHARACTER RECOGNITION (U+2440-U+245F)	bitstreamcyberbit, freemono, quivira
ENCLOSED ALPHANUMERICS (U+2460-U+24FF)	bitstreamcyberbit, quivira

Unicode Block	Fonts with full coverage
BOX DRAWING (U+2500-U+257F)	bitstreamcyberbit, dejavu*, freemono, quivira
BLOCK ELEMENTS (U+2580-U+259F)	bitstreamcyberbit, dejavu*, freemono, freesans, quivira
GEOMETRIC SHAPES (U+25A0-U+25FF)	bitstreamcyberbit, dejavu*, freemono, freeserif, quivira
MISCELLANEOUS SYMBOLS (U+2600-U+26FF)	bitstreamcyberbit, symbola
DINGBATS (U+2700-U+27BF)	symbola
MISCELLANEOUS MATHEMATICAL SYMBOLS-A (U+27C0-U+27EF)	quivira, symbola
SUPPLEMENTAL ARROWS-A (U+27F0-U+27FF)	dejavusans(+cond), dejavuserif(+cond), quivira, symbola
BRAILLE PATTERNS (U+2800-U+28FF)	dejavusans(+cond), dejavuserif(+cond), freemono, quivira
SUPPLEMENTAL ARROWS-B (U+2900-U+297F)	dejavuserif(+cond), quivira, symbola
MISCELLANEOUS MATHEMATICAL SYMBOLS-B (U+2980-U+29FF)	quivira, symbola
SUPPLEMENTAL MATHEMATICAL OPERATORS (U+2A00-U+2AFF)	quivira, symbola
MISCELLANEOUS SYMBOLS AND ARROWS (U+2B00-U+2BFF)	quivira, symbola
GLAGOLITIC (U+2C00-U+2C5F)	mph2bdamase
LATIN EXTENDED-C (U+2C60-U+2C7F)	quivira
COPTIC (U+2C80-U+2CFF)	quivira
GEORGIAN SUPPLEMENT (U+2D00-U+2D2F)	dejavuserif(+cond), mph2bdamase, quivira
TIFINAGH (U+2D30-U+2D7F)	dejavusans(+cond), mph2bdamase, quivira
ETHIOPIC EXTENDED (U+2D80-U+2DDF)	abyssinicasil
CYRILLIC EXTENDED-A (U+2DE0-U+2DFF)	quivira
SUPPLEMENTAL PUNCTUATION (U+2E00-U+2E7F)	symbola
CJK RADICALS SUPPLEMENT (U+2E80-U+2EFF)	hannoma, sun-exta
KANGXI RADICALS (U+2F00-U+2FDF)	hannoma, sun-exta
IDEOGRAPHIC DESCRIPTION CHARACTERS (U+2FF0-U+2FFF)	hannoma, sun-exta
CJK SYMBOLS AND PUNCTUATION (U+3000-U+303F)	bitstreamcyberbit, sun-exta
HIRAGANA (U+3040-U+309F)	bitstreamcyberbit, sun-exta
KATAKANA (U+30A0-U+30FF)	bitstreamcyberbit, sun-exta
BOPOMOFO (U+3100-U+312F)	bitstreamcyberbit
HANGUL COMPATIBILITY JAMO (U+3130-U+318F)	bitstreamcyberbit, sun-exta, unbatang
KANBUN (U+3190-U+319F)	bitstreamcyberbit, sun-exta
BOPOMOFO EXTENDED (U+31A0-U+31BF)	bitstreamcyberbit, sun-exta
CJK STROKES (U+31C0-U+31EF)	bitstreamcyberbit, sun-exta
KATAKANA PHONETIC EXTENSIONS (U+31F0-U+31FF)	bitstreamcyberbit, hannoma, sun-exta
ENCLOSED CJK LETTERS AND MONTHS (U+3200-U+32FF)	bitstreamcyberbit
CJK COMPATIBILITY (U+3300-U+33FF)	bitstreamcyberbit
CJK UNIFIED IDEOGRAPH-EXTENSION A (U+3400-U+4DB5)	hannoma, sun-exta
YIJING HEXAGRAM SYMBOLS (U+4DC0-U+4DFF)	dejavusans(+cond), quivira, sun-exta, symbola
CJK UNIFIED IDEOGRAPH-EXTENSION B (U+4E00-U+9FFF)	bitstreamcyberbit, hannoma, sun-exta (>90%)
YI SYLLABLES (U+A000-U+A48F)	sun-exta
YI RADICALS (U+A490-U+A4CF)	sun-exta
LISU (U+A4D0-U+A4FF)	quivira
VAI (U+A500-U+A63F)	wakor
CYRILLIC EXTENDED-B (U+A640-U+A69F)	quivira

Unicode Block	Fonts with full coverage
BAMUM (U+A6A0-U+A6FF)	
MODIFIER TONE LETTERS (U+A700-U+A71F)	charissil, quivira
LATIN EXTENDED-D (U+A720-U+A7FF)	quivira
SYLOTI NAGRI (U+A800-U+A82F)	mph2bdamase
PHAGS-PA (U+A840-U+A87F)	
Vertical positioning required	
SAURASHTRA (U+A880-U+A8DF)	
KAYAH LI (U+A900-U+A92F)	
REJANG (U+A930-U+A95F)	
HANGUL CHOSEONG (U+A960-U+A97F)	unbatang
JAVANESE (U+A980-U+A9DF)	
CHAM (U+AA00-U+AA5F)	
MYANMAR (U+AA60-U+AA7B)	
Special processing required	
TAI VIET (U-AA80-U+AADF)	taiheritagepro
MEETEI MAYEK (U+ABC0-U+ABF9)	meetemayek
HANGUL SYLLABLES (U+AC00-U+D7FF)	unbatang
CJK COMPATIBILITY IDEOGRAPHHS (U+F900-U+FAFF)	sun-exta (>90%)
ALPHABETIC PRESENTATION FORMS (U+FB00-U+FB4F)	dejavusans(+cond), quivira
ARABIC PRESENTATION FORMS-A (U+FB50-U+FDFF)	
VARIATION SELECTORS (U+FE00-U+FE0F)	bitstreamcyberbit, charissil, dejavusans(+cond), dejavuserif(+cond)
VERTICAL FORMS (U+FE10-U+FE1F)	bitstreamcyberbit, sun-exta, symbola, unbatang
COMBINING HALF MARKS (U+FE20-U+FE2F)	bitstreamcyberbit, symbola
Special positioning required	
CJK COMPATIBILITY FORMS (U+FE30-U+FE4F)	bitstreamcyberbit, hannoma, symbola
SMALL FORM VARIANTS (U+FE50-U+FE6F)	bitstreamcyberbit, sun-exta
ARABIC PRESENTATION FORMS-B (U+FE70-U+FEFE)	bitstreamcyberbit, dejavusans(+cond), dejavusansmono
SPECIALS (U+FEFF-U+FEFF)	bitstreamcyberbit, charissil, dejavusans(+cond), dejavusansmono, freeserif
HALFWIDTH AND FULLWIDTH FORMS (U+FF00-U+FFEF)	sun-exta (>90%)
SPECIALS (U+FFF0-U+FFFD)	charissil, dejavu*, symbola
Unicode Plane 1 (SMP) Supplementary Multilingual Plane	
LINEAR B SYLLABARY (U+10000-U+1007F)	aegean
LINEAR B IDEOGRAMS (U+10080-U+100FF)	aegean
AEGEAN NUMBERS (U+10100-U+1013F)	aegean
ANCIENT GREEK NUMBERS (U+10140-U+1018F)	aegean, quivira
ANCIENT SYMBOLS (U+10190-U+101CF)	aegean, quivira
PHAISTOS DISC (U+101D0-U+101FF)	aegean
LYCIAN (U+10280-U+1029F)	aegean, quivira
CARIAN (U+102A0-U+102DF)	aegean, quivira
OLD ITALIC (U+10300-U+1032F)	aegean, mph2bdamase, quivira
GOTHIC (U+10330-U+1034F)	freeserif, mph2bdamase, quivira
UGARITIC (U+10380-U+1039F)	aegean, freesans, mph2bdamase
OLD PERSIAN (U+103A0-U+103DF)	aegean, freesans, mph2bdamase
DESERET (U+10400-U+1044F)	mph2bdamase (>90%)
SHAVIAN (U+10450-U+1047F)	mph2bdamase

Unicode Block	Fonts with full coverage
OSMANYA (U+10480-U+104AF)	mph2bdamase
CYPRIOT SYLLABARY (U+10800-U+1083F)	aegean, mph2bdamase
PHOENICIAN (U+10900-U+1091F)	aegean, freesans
LYDIAN (U+10920-U+1093F)	aegean, quivira
KHAROSHTHI (U+10A00-U+10A5F)	mph2bdamase
KAITHI (U+11080-U+110CF)	
CUNEIFORM (SUMERO-AKKADIAN) (U+12000-U+123FF)	akkadian
CUNEIFORM NUMBERS AND PUNCTUATION (U+12400-U+1247F)	akkadian
EGYPTIAN HIEROGLYPHS (U+13000-U+1342F)	aegyptus
BYZANTINE MUSICAL SYMBOLS (U+1D000-U+1D0FF)	freeserif
MUSICAL SYMBOLS (U+1D100-U+1D1FF)	freeserif
ANCIENT GREEK MUSICAL NOTATION (U+1D200-U+1D24F)	aegean, quivira
TAI XUAN JING SYMBOLS (U+1D300-U+1D35F)	dejavusans, quivira, sun-extb
COUNTING ROD NUMERALS (U+1D360-U+1D37F)	symbola
MATHEMATICAL ALPHANUMERIC SYMBOLS (U+1D400-U+1D7FF)	freeserif, symbola
MAHJONG TILES (U+1F000-U+1F02F)	freeserif, symbola
DOMINO TILES (U+1F030-U+1F09F)	freeserif, quivira, symbola
Unicode Plane 2 (SIP) Supplementary Ideographic Plane	
CJK UNIFIED IDEOGRAPHS EXTENSION B (U+20000-U+2A6DF)	hannomb, sun-extb
CJK UNIFIED IDEOGRAPHS EXTENSION C (U+2A700-U+2B734)	sun-extb
CJK COMPATIBILITY IDEOGRAPHS SUPPLEMENT (U+2F800-U+2FA1F)	hannomb, sun-extb

Font name	Font file	Available from
charissil	CharisSILR.ttf	http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&cat_id=FontDownloads
quivira	Quivira.ttf	http://www.quivira-font.com/
symbola	Symbola.otf	http://users.teilar.gr/~g1951d/
bitstreamcyberbit	Cyberbit.ttf	http://aol-4.vo.llnwd.net/pub/communicator/extras/fonts/windows/
mph2bdamase	damase_v.2.ttf	http://www.wazu.jp/gallery/views/View_MP2BDamase.html
garuda	Garuda.ttf	ftp://linux.thai.net/pub/thailinux/software/thai-ttf/
norasi	Norasi.ttf	ftp://linux.thai.net/pub/thailinux/software/thai-ttf/
kaputaunicode	kaputaunicode.ttf	http://info.lk/slword/kaputaunicode.htm
tibetanunicode	TibetanUnicode.ttf	http://www.popdict.com/dict_tibetan.htm#fonts
abyssinicasil	Abyssinica_SIL.ttf	http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&cat_id=FontDownloads
aboriginalsans, aboriginalserif	AboriginalSansREGULAR.ttf AboriginalSerifREGULAR.ttf	http://www.languagegeek.com/font/fontdownload.html#Full_Unicode
khmerosbattambang	KhmerOS_battambang.ttf	http://sourceforge.net/projects/khmer/files/Fonts%20-%20KhmerOS/
daibannasilbook	DBSILBR.ttf	http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&cat_id=FontDownloads
lannaalif	Iannaalif-v1-03.ttf	http://www.geocities.jp/simsheart_alif/taithamunicode.html
sundaneseunicode	SundaneseUnicode-1.0.5.ttf	http://sabilulungan.org/aksara/
wakor	Wakor.ttf	http://www.evertype.com/fonts/vai/
taiheritagepro	TaiHeritagePro.ttf	http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&cat_id=FontDownloads
meeteimayek	Eeyek.ttf	http://tabish.freeshell.org/eeyek/download.html
aegean	Aegean.otf	http://users.teilar.gr/~g1951d/
akkadian	Akkadian.otf	http://users.teilar.gr/~g1951d/
aegyptus	Aegyptus.otf	http://users.teilar.gr/~g1951d/
hannoma / hannomb	HAN NOM A.ttf HAN NOM B.ttf	http://vietunicode.sourceforge.net/fonts/fonts_hannom.html
sun-exta / sun-extb	Sun-ExtA.ttf Sun-ExtB.ttf	http://www.alanwood.net/unicode/fonts-east-asian.html

Font name	Font file	Available from
unbatang	UnBatang_0613.ttf	http://people.ktug.or.kr/~nomos/mine/UnBatang_0613.ttf
xbzar / xbriyaz	XB Riyaz.ttf XB Zar.ttf	http://wiki.irmug.com/index.php/X_Series_2

PDF FILES (ADOBE)

PDF reference

'PDF Reference' provides a description of the Portable Document Format, and can be downloaded from Adobe at http://www.adobe.com/devnet/pdf/pdf_reference.html and http://www.adobe.com/devnet/pdf/pdf_reference_archive.html

PDF files produced from this program are marked as %PDF1.4

PDF Reference Manual Edition	PDF File Version	Acrobat Reader Version
3rd	%PDF1.4	5.x
4th	%PDF1.5	6.x
5th	%PDF1.6	7.x
6th	%PDF1.7	8.x
PDF 32000-1:2008 1st edition	%PDF1.7	9.x

INDEX

A

A5 Booklet 202
About PDF files 16
AddColumn() 232
AddPage() 233
AddPageByArray() 238
AddSpotColor 244
adjustFontDesclineheight 422
AliasNbPageGroups() 246
AliasNbPages() 245
aliasNbPg 423
aliasNbPgGp 424
allow_charset_conversion 425
allowCJkorphans 429
allowCjKoverflow 430
allow_html_optional_endtags 427
allow_output_buffering 428
anchor2Bookmark 431
annotation 359
Annotation() 247
Annotations 153
annotMargin 432
annotOpacity 433
Arabic (RTL) text v5.x 48
ASCII characters 543
autoArabic 434
autoFontSize 435
autoLangToFont 437
Auto-layout algorithm 72
autoMarginPadding 438
Automatic font selection 42
autoPageBreak 439
autoScriptToLang 440
AutoSizeText() 250
autoVietnamese 441
Available Fonts v5.x 32
Available fonts v6 33

B

Backgrounds & Borders 134
backupSIPFont 443
backupSubsFont 442
barcode 362
Barcodes 154
baseScript 444
biDirectional 445
Bidirectional (RTL) text v6.x 50
BIG-5 (Traditional Chinese) 555
Blank screen 218
bleedMargin 446
bookmark 365
Bookmark() 251
Bookmarks 155
bookmarkStyles 447
Border collapse 73
Browser compatible HTML 108
Browser incompatability 216

C

Capture HTML output 180
Character substitution 57
charset_in 448
Choosing a configuration v5.x 35
Choosing a configuration v6.x 39
CircularText() 253
CJKforceend 449
CJK Languages 52
CMYK colours 170

collapseBlockMargins 450
Colour Charts CMYK 199
columnbreak 367
columns 368
Columns 156
Combining Diacritics 185
Configuration files v5.x 58
Configuration files v6.x 59
Configuration Methods 60
Configuration Variables 61
Corrupt PDF file 219
CreateIndex() 254
CreateReference() 257
Creating your first file 14
Credits 6
cropMarkLength 451
cropMarkMargin 452
crossMarkMargin 453
CSSselectMedia 454
Custom HTML Tags 68

D

debug 456
debugfonts 457
decimal_align 455
Default Font 54
defaultPageNumStyle 459
Default stylesheet 125
DefFooterByName() 258
DefHeaderByName() 260
DefHTMLFooterByName() 262
DefHTMLHeaderByName() 263
Demo - 1000 Character Classic 589
Demo - Other languages 588
Demo - Other Unicode character 591
Demo - Pangrams 585
Different page sizes 85
Direct writing to document 209
displayDefaultOrientation 458
Document Metadata 132
dottab 370
Double-sided documents 77
dpi 460

E

E-mail a PDF file 200
enableImports 461
Equivalent codepages 594
Error messages 217

F

falseBoldWeight 462
Features v5 2
Fixed position blocks 137
Floating blocks 139
Folders for temporary files 13
Font names 30
Fonts in mPDF 5.x 17
Fonts in mPDF 6.x 22
Fonts & Language cover v5.x 47
Font substitution 5.x 55
Font substitution 6.x 56
forcePortraitHeaders 463
forcePortraitMargins 464
formfeed 371
Forms 157
FPDF Original Functions 357

G

GBK (Simplified Chinese) 565
Graphs 166

H

h2bookmarks 465
h2toc 466
Headers & Footers 86
Headers & Top margins 107
HKCS (Hong Kong Character Set) 562
HTML Attributes 65
HTML or PHP? 15
htmlpagefooter 375
htmlpageheader 376
HTML Tags 62
Hyphenation 140

I

ICCPProfile 467
iconv 596
Image() 264
Image errors 221
Images 142
img_dpi 468
Importing files & Templates 171
ImportPage() 265
incrementFPR1 [1-4] 469
Index 159
indexentry 377
IndexEntry() 267
IndexEntrySee() 269
indexinsert 379
Indic fonts v5.x 53
Input encoding 46
InsertIndex() 270
Installation v5.x 8
Installation v6.x 9
Introduction 110
Invoice 192
ISO 639-1 language codes 607
ISO-8859-2 545
ISO-8859-4 546
ISO-8859-7 547
ISO-8859-9 548
ISO-8859/win comparison chart 549
iterationCounter 470

J

jpgraph 382
jSmaxChar 471
jSmaxCharLast 472
jSmaxWordLast 473
jSWord 474
justifyB4br 475

K

keepColumns 476
keep_table_proportions 477
Kerning 146
Known Issues 212

L

lang 6.x 45
lang v5.x 44
Layers 162
Letterhead letters 204
Licence 7
Limitations 5
Line breaking 149
Line-height 147
list_align_style 478
list_auto_mode 479

list_indent_default 480
list_indent_default_mpdf 481
list_indent_first_level 482
list_marker_offset 483
list_number_suffix 484
Lists 150
list_symbol_size 485

M

margBuffer 486
Math Formulae with MathJax 182
Math with MathJax 2 183
max_colH_correction 487
maxTTFFilesize 488
mbstring 598
Memory problems 222
Method 1 88
Method 2 93
Method 3 96
Method 4 101
mirrorMargins 489
mPDF() 226
mPDF class fails to Initialise 224
MultiCell() 271

N

Named colours 127
nbpgPrefix 490
nbpgSuffix 491
nonPrintMargin 492
normalLineheight 493
Notice warnings 220

O

OpenType layout (OTL) 26
Output() 272
Overview 229, 358, 417
OverWrite() 274
Overwriting existing files 172

P

packTableData 494
pagebreak 387
Page breaks 74
pagefooter 394
pageheader 392
Page numbering 79
Page numbers & Date 106
pagenumPrefix 495
pagenumSuffix 496
Page size & Orientation 78
Password protection 131
PDFA 497
PDF/A1-b compliance 174
PDFAAuto 498
PDF from every page of website 190
PDF reference 619
PDF Version 133
PDFX 499
PDF/X-1a compliance 177
PDFXauto 500
percentSubset 501
Postscript printers 225
preparePreText() 540
printers_info 502
progressbar_altHTML 503
progressbar_heading 504
progressBar 505

R

Reducing memory usage 12
repackageTTF 506
Replaceable Aliases 168

Requirements v5 4
Reserved Terms 215
Reserving x blank pages 203
Resizing 223
RestartDocTemplate() 277
restoreBlockPagebreaks 507
restrictColorSpace 508
Rotated pages 109
RoundedRect() 278

S

Saving to a database 181
SetAlpha() 279
SetAnchor2Bookmark() 281
SetAuthor() 282
setAutoBottomMargin 509
SetAutoFont() 283
setAutoTopMargin 510
SetBasePath() 285
SetColumns() 286
SetCompression() 288
SetCreator() 289
SetDefaultBodyCSS() 290
SetDefaultFont() 291
SetDefaultFontSize() 293
SetDirectionality() 292
SetDisplayMode() 294
SetDisplayPreferences() 296
SetDocTemplate() 297
SetFooter() 299
SetFooterByName() 302
SetHeader() 303
SetHeaderByName() 306
SetHTMLFooter() 308
SetHTMLFooterByName() 309
SetHTMLHeader() 310
SetHTMLHeaderByName() 311
sethtmlpagefooter 396
sethtmlpageheader 398
SetImportUse() 313
SetKeywords() 314
setpagefooter 400
setpageheader 402
SetPageTemplate() 315
SetProtection() 317
SetSourceFile() 319
SetSubject() 321
SetTitle() 322
SetVisibility() 324
SetWatermarkImage() 325
SetWatermarkText() 327
Shaded box() 329
SHIFT_ЈЅ (Japanese) 573
showImageErrors 511
showStats 512
showWatermarkImage 513
showWatermarkText 514
SHYlang 515
simpleTables 516
Slow! 213
smCapsScale 517
smCapsStretch 518
StartProgressBarOutput() 330
strcode2utf() 538
Supported CSS 112
Symbols (Adobe) 553

T

Table layout 71
tableMinSizePriority 519
Table of Contents 163
Tables 69
tabSpaces 520
textcircle 404
Text is replaced 214
Text Justification 152
Thumbnail() 331
title2annots 521
tocentry 406
TOC_Entry() 343
tocpagebreak 408
TOCpagebreak() 333
TOCpagebreakByArray() 338

U

UHC (Korean) 577
Unicode character planes 599
Unicode coverage of Free Fonts 613
useAdobeCJK 523
useDictionaryLBR 524
useFixedNormalLineHeight 525
useFixedTextBaseline 526
Useful links 612
useGraphs 527
useKerning 528
use_kwt 522
useLang 529
User Input 207
useSubstitutions 531
UseTemplate() 345
useTibetanLBR 532
Using @page 81

V

Vietnamese 554

W

watermark font 533
watermarkImage 413
watermarkImageAlpha 534
watermarkImgAlphaBlend 535
watermarkImgBehind 536
Watermarks 165
watermarkText 415
watermarkTextAlpha 537
Win-1251 544
Win-1252 542
WriteBarcode() 348
WriteCell() 350
WriteFixedPosHTML() 351
WriteHTML() 353
WriteText() 356
Writing non-HTML text 173
WYSIWYG editor 120

Y

Year Book 196

Z

ZapfDingbats (Adobe) 552

