

PHP Pdf Creation

Module-free creation of Pdf documents
from within PHP

developed by R&OS Ltd

Contents

Introduction	1
Changes	1
version 009	1
version 008	1
version 007	2
version 006	2
version 005	3

getTextWidth	22
addTextWrap	22
saveState	22
restoreState	22
openObject	22
reopenObject	22
closeObject	23
addObject	23
stopObject	23
addInfo	23
setPreferences	23
setEncryption	23
addImage	23
addJpegFromFile	24
addPngFromFile	24
output	24
openHere	24
selectFont	25
setFontFamily	26
	27
addLink	27
addInternalLink	27
addDestination	28
	28
Misc	29
Callback functions	29
	30
FAQs	32
Whats all this crap I see, my page is broken ?	32
What page sizes can I use ?	32
What is this <i>Call to undefined function: imagecreatefromjpeg()</i> error about ?	32
Does this work with accented characters ?	33

Introduction

This class is designed to provide a **non-module**, non-commercial alternative to dynamically creating pdf documents from within PHP.

Obviously this will not be quite as quick as the module alternatives, but it is surprisingly fast, this

- adjusted the size of an A4 page to be more accurate.
- fixed the table title alignment to stay centered on the table if the table position is fixed on the page
- used callback functions to add clickable links to the text.

Please excuse this blatant 'plug', but if your company wishes some customization of these routines for your purposes, R&OS can do this at very reasonable rates, just drop us a line at info@ros.co.nz.

'EXECUTIVE' (521.86,756.00), 'FOLIO' (612.00,936.00)

`ezSetDy(dy [,mod])`

Changes the vertical position of the writing point by a set amount, so to move the pointer 10 s4o3p down the page (making a gap in the writing), use:

```
ezSetDy(-10)
```

If this movement makes the writing location below the bottom margin, then a new page will automatically be made, and the pointer moved to the top of it.

The optional parameter 'mod' can be set to the value 'makeSpace', which means that if a new page is forced, then the pointer will be moved the distance 'dy' on the new page as well. The intention of this is if you needed 100 s4o3p of space to draw a picture, then doing:

```
ezSetDy(-100, 'makeSpace')
```

guarantees that there will be 100 s4o3p of space above the final writing point.

ezTable

`y=ezTable(array data,[array cols],[title],[array options])`

The easy way to throw a table of information onto the page, can be used with just the data variable, which must contain a two dimensional array of data. This function was made with data extracted from database queries in mind, so is expecting it in that format, a two dimensional array with the first array having one entry for each row (and each of those is another array).

The table will start writing from the current writing point, and will proceed until all the data has been present. By default, borders will be drawn, alternate lines will be shaded gray, and the table will wrap over pages, re-printing the headers at the top of each page.

The return value from the function is the y-position of the writing pointer after the table has been added to the document.

The other options are described here:

`$cols` (optional) is an associative array, the keys are the names of the columns from `$data` to be present `p` (and in that order), the values are the titles to be given to the columns, if this is not wanted but you do want later options then "" (the empty string) is a suitable placeholder.

`$title` (optional) is the title to be put on the top of the table

`$options` is an associative array which can contain:

'colGap' => 5 , the space between the text and the column lines in each column

and in the other order, also have column headings and a table heading.

```
$pdf->ezTable($data,array('type'=>'Type','name'=><i>Alias</i>'))
```



```
, 'Some LOTR Characters'
```

Now a version with the width specified to be too small, so that the content has to wrap, and the table oriented over the the right.

number a a a a a a a a a a a a a a a more	Name	Type
1	gandalf	wizard
2	bilbo	hobbit
3	frodo	hobbit
4	saruman	bad dude
5	sauron	really bad dude

A simple example, showing the table expanding to fit the specified width, and having links on one of the table columns.

```
$pdf->ezTable($data,array('type'=>'','name'=>''),' ',
    ,array('showHeadings'=>0
    , 'shaded'=>0
    , 'width'=>400
    , 'cols'=>array('name'=>array('link'=>'url')) ));
```

wizard	gandalf
hobbit	bilbo
hobbit	frodo
bad dude	

`$border['join']` is the join type as specified in `setLineStyle`. The default is 'round'.
`$border['color']` is an associative array for specifying the line color of the border.
The values are as specified in `setStrokeColor` and should be assigned to:
`$border['color']['red']`, `$border['color']['green']` and `$border['color']['blue']` respectively.

ezStartPageNumbers

`setNum = ezStartPageNumbers(x,y,size,[pos],[pattern],[num])`

Add page numbers on the pages from here, place then on the 'pos' side of the coordinates (x,y) (pos can be 'left' or 'right').

Use the given 'pattern' for display, where {PAGENUM} and {TOTALPAGENUM} are replaced as required, by default the pattern is set to '{PAGENUM} of {TOTALPAGENUM}'

If \$num is set, then make the first page this number set,5,H3'pattern' fo66ages frwille assdjust to:

```
$pdf->ezNewPage();  
$pdf->ezNewPage();  
$pdf->ezNewPage();  
$j=$pdf->ezNewStarte(Numbers(300,400,20,'',' ',1
```

ezStopPageNumbers

`ezStopPageNumbers([stopTotal],[next],[setNum])`

In version 009 this function was enhanced to include a number of extra parameters:

'stopTotal' => 0 or 1 (default 0), stops the totaling for the page numbering set. So for example if you start numbering a 10 page document on the second page, and stop it on the 4th page, with stopTotal set to 1, then the numbers will be reported as "x of 3".

'next' => 0 or 1, stops on the next page, not this one.

'setNum' => (defaults to 0) define which set number is to be stopped, this is the number returned from the *ezStartPageNumbers* command.

ezColumnsStop

ezColumnsStop()

This stops multi-column mode, it will leave the writing point at whatever level it was at, it is recommended that an ezNewPage() command is executed straight after this command, but for flexibility this is left up to the individual consumer.

inline codes

There are a few callback functions (see the base class functions for an explanation of callback

It is similar to *alink* except that instead of providing a URL the label of a pre-created destination should be used.

`<c:ilink:destination_label>text to be clickable</c:ilink>`

```
// place required to be marked
$pdf->addDestination('xxxxyyyzzz','Fit');
// add lots of stuff, new pages etc, then...
$pdf->ezText('<c:ilink:xxxxyyyzzz>R&OS pdf class</c:ilink>');
```

Click here to go to the 5th item on the table of contents.

Note that the code for the example and the actual one shown are not identical for technical reasons.

Base Class Functions

addText

`addText(x,y,size,text,[angle=0],[adjust=0])`

Add the text at a particular location on the page, noting that the origin on the axes in a pdf document is in the lower left corner by default.

An angle can be supplied as this will do the obvious (in degrees).

'adjust', gives the value of units to be added to the width of each space within the text. This is used mainly to support the justification options within the ezpdf ezText function.

directives are similar the basic html:

`bold text`

`<i>italic text</i>`

`<i>bold italic text</i>`

Note that there can be no spaces within the directives, and that they must be in lower case.

By default, these will work only with the supplied fonts, and when the font was selected it must have been specified with the '.afm' suffix. For more information about why this is and how you can customise this behaviour see the **setFontFamily** command.

setLineStyle

```
setLineStyle([width],[cap],[join],[dash],[phase])
```

This sets the line drawing style.



`rectangle(x1,y1,width,height)`

Obvious.

newPage

`id=newPage([il4.i,id,pos])`

Starts a new page and returns the id of the page contents, this can be safely ignored, but storing it will allow the il4.iiion of more ilformaion back ilto the page later, through the use of the 'reopenObject' funcion.

The comm 12.is usually used without any of the opions to simply add a new page to the end of the current bunch, but with the opions can be used to il4.i a page within the existing pages. The 'il4.i'

previous newPage comm 12.(this is actually the object id of the contents of a page). 'pos' will

Returns a number which is the distance that the descender goes beneath the Baseline, for a normal character set this is a negative number.

getTextWidth

```
x=getTextWidth(size,text)
```

Returns the width of the given text string at the given size.

addTextWrap

```
a=addTextWrap(x,y,width,size,text,[justification='left'][,angle=0])
```

closeObject

`closeObject()`

Close the currently open object. Further writes will now go to the current page.

addObject

`addObject(id,[options='add'])`

Add the object specified by id to the current page (default). If a string is supplied in options, then the following may be specified:

'add' - add to the current page only.

'all' - add to every page from the current one on.

'odd' - add to all odd numbered pages from now on.

'even' - add to all even numbered pages from now on.

'next' - add to just the next page.

'nextodd' - add to all odd numbered pages from the next one.

'nexteven' - add to all even numbered pages from the next one.

stopObject

`stopObject(id)`

If the object (id) has been appearing on pages up to now, then stop it, this page will be the last one
astcurrent paglo4will dwilntasupit next one.

addImage(img,x,y,w,[h],[quality=75])

'Fit' *fit the page to the view*

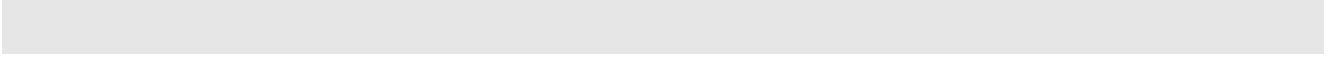
'FitH' top

addDestination

`addDestination(label,style[,a][,b][,c])`

This marks a point in the document as a potential destination for an internal link, the 'label' can be any string, though should be unique for the document, else confusion may ensue. The remainder of the

Misc



and some additional user contributed notes are (thanks Andrew):

$1/72''$ is 0.3528mm or 1 point

1 point was historically 0.0138 inches, a little under $1/72''$

10mm is 28.35 points

A4 is 210 x 297 mm or 595.28 x 841.89 points

FAQs

Whats all this crap I see, my page is broken ?

If when you try to view your pdf document, you just get a whole lot of "junk", that looks like this:

```
<</Type/Outlines/Count0>>endobj30obj<</Type/Pages/Kids[60R]...  
(and a whole lot more)
```

This is most likely due to a problem with IE not refreshing the mime type of a page. What can happen is if you have had an error in the creation of the page in the past, then you fix the error, but just keep refreshing the page, IE does not re-register the mime-type of the page as a pdf, but keeps it being a text page (like it was when there was the error).

The solution is simple. Shut the browser (copying the URL before you do it makes life easier), then open a new browser window and paste the URL into it then press enter, that it is a pdf page) Tj ET BT 50.000 410.986 Td ET

update - this functionality has been added, see the documentation for the addJpegFromFile command, there is also support for PNG files (addPngFromFile)

Does this work with accented characters ?

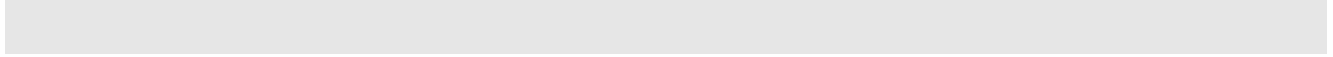
Being a slightly ignorant english speaker I completely forgot about

```
$pdf->selectFont('./fonts/Helvetica');  
$pdf->addText(30,400,30,'Hello World');  
$pdf->stream();
```

Can I put the fonts directory somewhere else ?

Yes, the selectFont function will take an absolute path so you can specify where you would like the

```
$y=$y-$height;
$pdf->addText(30,$y,$size,$demotext);
}
for ($angle=0;$angle<360;$angle=$angle+20){
    $r=rand(0,100)/100;
    $g=rand(0,100)/100;
    $b=rand(0,100)/100;
    $pdf->setColor($r,$g,$b);
    $pdf->addText(300+cos(deg2rad($angle))*40,300-
sin(deg2rad($angle))*40,20,$demotext,$angle);
}
$pdf->stream();
```



R2L Right to left (including vertical writing systems
such as Chinese, Japanese, and Korean)


```

</html>
';

    // also have a look through the directory, and remove the files that are
    // older than a week
    if ($d = @opendir($dir)) {
        while (($file = readdir($d)) != false) {
            if (substr($file,0,4)=="PDF_"){
                // then check to see if this one is too old
                $ftime = filemtime($dir.'/'.$file);
                if (time()-$ftime > 3600*24){
                    unlink($dir.'/'.$file);
                }
            }
        }
        closedir($d);
    }
}

```

Note that this code will also scan the folder and will remove any files which are more than a day old.

How do I get the Euro character to show in my text?

The Euro character seems to be outside the normal character set, so you need to replace one of your 0-255 characters with it, using the options of the selectFont command (noting that these only work the **first time** that the function is called for each font).

Here is an example which replaces the "!" character:

```

<?php
include ('class.ezpdf.php');

$pdf =& new Cezpdf('a4');

$euro_diff = array(33=>'Euro');
$pdf->selectFont('./fonts/Helvetica.afm',
    array('encoding'=>'WinAnsiEncoding','differences'=>$euro_diff));

$pdf->ezText('**!!!!!!**',100,array('justification'=>'full'));

$pdf->ezText($pdf->messages,12);

if (isset($d) && $d){
    $pdfcode = $pdf->ezOutput(1);
    $pdfcode = str_replace("\n","<br>",htmlspecialchars($pdfcode));
    echo '<html><body>';
}

```

Just for the reference (FAQ? :-), I've found more references about using the Euro in ASCII-Tables:

<http://www.dragoman.org/euro/>

replaces the "^"-char (0x5E) by the Euro-Symbol (U+20AC) and calls the new Table "Euro-ASCII"

<http://lists.w3.org/Archives/Public/www-international/1997OctDec/att-0077/01-euro.html>

replaces the "|" (0x7C) by the Euro

http://www.alphaquark.com/Traduction/En_Euro.htm

replaces the ascii-char at 128 (0x80)

My Linux-console (vim + xxd) uses 0xa4...

How do I get the symbolic fonts (eg zapfdingbats) working?

It seems that these need to have the encoding set to "none" in order to work, this should be done on the first selectFont command for the font, and will look something like:

```
$pdf->selectFont('ZapfDingbats.afm','none');
```

or:

```
$pdf->selectFont('ZapfDingbats.afm',array('encoding'=>'none'));
```

How do I make a table from my database information?

This is relatively simple, you must populate an array from the database, and then use that to make the pdf table. As there seems to be a lot of trouble with this, here is an example:

```
<?php
// test the table functions
error_reporting(E_ALL);
include('class.ezpdf.php');
```

