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 encrpyted 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 mPDFI('');
$mpdf->SetImportUse();

// forces no subsetting - otherwise the inserted characters may not be contained in a subset
font
$mpdf->percentSubset = 0;

$search = array(
        'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX',
        'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXZZZZZZZZ'
);


$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 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX</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(
```

```
        'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
);

$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

Printed on Wed 05 Aug 2015 12:14:51 GMT +0100 (DST)