# Adding new fonts and encodings

This tutorial explains how to use TrueType, OpenType and Type1 fonts so that you are not limited to the standard fonts anymore. The other benefit is that you can choose the text encoding, which allows you to use other languages than the Western ones (the standard fonts support only cp1252 aka windows-1252).

For OpenType, only the format based on TrueType is supported (not the one based on Type1).

For Type1, you will need the corresponding AFM file (it is usually provided with the font).

Adding a new font requires two steps:

* Generation of the font definition file
* Declaration of the font in the script

## Generation of the font definition file

The first step consists in generating a PHP file containing all the information needed by FPDF; in addition, the font file is compressed. To do this, a helper script is provided in the makefont directory of the package: makefont.php. It contains the following function:

MakeFont(**string** fontfile [, **string** enc [, **boolean** embed [, **boolean** subset]]]) fontfile

Path to the .ttf, .otf or .pfb file.

enc

Name of the encoding to use. Default value: cp1252.

embed

Whether to embed the font or not. Default value: true.

subset

Whether to subset the font or not. Default value: true.

The first parameter is the name of the font file. The extension must be either .ttf, .otf or .pfb and determines the font type. If your Type1 font is in ASCII format (.pfa), you can convert it to binary (.pfb) with the help of [t1utils](http://www.lcdf.org/~eddietwo/type/#t1utils).

For Type1 fonts, the corresponding .afm file must be present in the same directory.

The encoding defines the association between a code (from 0 to 255) and a character. The first 128 are always the same and correspond to ASCII; the following are variable. Encodings are stored in .map files. The available ones are:

* cp1250 (Central Europe)
* cp1251 (Cyrillic)
* cp1252 (Western Europe)
* cp1253 (Greek)
* cp1254 (Turkish)
* cp1255 (Hebrew)
* cp1257 (Baltic)
* cp1258 (Vietnamese)
* cp874 (Thai)
* ISO-8859-1 (Western Europe)
* ISO-8859-2 (Central Europe)
* ISO-8859-4 (Baltic)
* ISO-8859-5 (Cyrillic)
* ISO-8859-7 (Greek)
* ISO-8859-9 (Turkish)
* ISO-8859-11 (Thai)
* ISO-8859-15 (Western Europe)
* ISO-8859-16 (Central Europe)
* KOI8-R (Russian)
* KOI8-U (Ukrainian)

Of course, the font must contain the characters corresponding to the selected encoding.

The third parameter indicates whether the font should be embedded in the PDF or not. When a font is not embedded, it is searched in the system. The advantage is that the PDF file is smaller; on the other hand, if it is not available, then a substitution font is used. So you should ensure that the needed font is installed on the client systems. Embedding is the recommended option to guarantee a correct rendering.

The last parameter indicates whether subsetting should be used, that is to say, whether only the characters from the selected encoding should be kept in the embedded font. As a result, the size of the PDF file can be greatly reduced, especially if the original font was big.

After you have called the function (create a new file for this and include makefont.php), a .php file is created, with the same name as the font file. You may rename it if you wish. If the case of embedding, the font file is compressed and gives a second file with .z as extension (except if the compression function is not available, it requires Zlib). You may rename it too, but in this case you have to change the variable $file in the .php file accordingly.

Example:

<?php  
require('makefont/makefont.php');  
  
MakeFont('C:\\Windows\\Fonts\\comic.ttf','cp1252');  
?>

which gives the files comic.php and comic.z.

Then copy the generated files to the font directory. If the font file could not be compressed, copy it directly instead of the .z version.

Another way to call MakeFont() is through the command line:

php makefont\makefont.php C:\Windows\Fonts\comic.ttf cp1252

Finally, for TrueType and OpenType fonts, you can also generate the files [online](http://www.fpdf.org/makefont/) instead of doing it manually.

## Declaration of the font in the script

The second step is simple. You just need to call the [AddFont()](http://docs.google.com/doc/addfont.htm) method:

$pdf->AddFont('Comic','','comic.php');

And the font is now available (in regular and underlined styles), usable like the others. If we had worked with Comic Sans MS Bold (comicbd.ttf), we would have written:

$pdf->AddFont('Comic','B','comicbd.php');

## Example

Now let's see a complete example. We will use the [Ceviche One](https://fonts.google.com/specimen/Ceviche+One) font. The first step is the generation of the font files:

<?php  
require('makefont/makefont.php');  
  
MakeFont('CevicheOne-Regular.ttf','cp1252');  
?>

The script produces the following output:

Font file compressed: CevicheOne-Regular.z

Font definition file generated: CevicheOne-Regular.php

Alternatively we could have used the command line:

php makefont\makefont.php CevicheOne-Regular.ttf cp1252

or used the online generator.

We can now copy the two generated files to the font directory and write the script:

<?php  
require('fpdf.php');  
  
$pdf = new FPDF();  
$pdf->AddFont('CevicheOne','','CevicheOne-Regular.php');  
$pdf->AddPage();  
$pdf->SetFont('CevicheOne','',45);  
$pdf->Write(10,'Enjoy new fonts with FPDF!');  
$pdf->Output();  
?>

[[Run]](http://docs.google.com/tuto7.php)