fonts: tutorial

Maarten Gelderman

# How to install a Type1 Font using fontinst

**abstract**
In this brief tutorial I will describe how a postscript Type 1 font can be made available to TEX using the fontinst-utility (version 1.8). I will not delve into the technical details of this program or the exact functionality of each of the font files used during this process, as those aspects of font-installation are described in other articles in this issue of MAPS. I will not delve into the advanced features of fontinst either, but rather will provide a hands-on tutorial.

**keywords**
fontinst, Type 1 font

Traditionally typesetting in TEX is done using bitmap fonts – pk-fonts – generated by the METAFONT-program, the most famous or notorious of which is the Computer Modern font, originally developed by Donald E. Knuth for usage with TEX. For typesetting involving advanced mathematics Computern Modern will remain the font of choice, although some commercial alternatives (Times and Lucida from Y&Y (`http://www.yandy.com`) and Helvetiva from Micropress (`http://www.micropress-inc.com`)) are available. For typesetting ordinary text, one may well prefer to use some other, probably commercial font. The majority of those fonts are available in either TrueType or Type 1 (PostScript) format. In this article I will describe how to setup one of the latter fonts, for usage with TEX.[1] Contrary to perceived wisdom, adding an additional PostScript font to your TEX-installation is remarkably uncomplicated.

## What do you need?

Of course you need a working TEX-installation. However, that will not suffice. I will also assume that you are using dvips to convert your dvi-files to printable postscript, which in turn supposes that you have a PostScript interpreter, most probably a PostScript-printer, GhostScript or Acrobat Distiller, available. Furthermore, you need the fontinst-utility, which is available from

`ftp://ftp.tex.ac.uk/tex-archive/fonts/utilities/fontinst`

or a similar directory in the CTAN-mirrors `ftp.dante.de` and `ftp.cs.ruu.nl`. I you are using a 'normal' TEX-distribution in combination with a PostScript printer or GhostScript, it is likely that those components have already been installed properly, otherwise you will have to take care of that yourself.

Of course, you also need the font you want to install. A font will typically exist of at least two files. The first eight letters of both file names are identical, the last three, the file names extension, differ. The first file has the file extention `.pfa` or `.pfb`. This is

1. If you want to use TrueType fonts wit TEX, the most easy thing to do seems to use either the program `ttf2pk`, which is available from the `fonts/utilities`-directory of CTAN and from `ftp.freetype.org` or to use `ttftot42`, which is available from `http://ftp.giga.or.at/pub/nih/ttftot42`. Those programs convert your TrueType fonts to ordinary TEX and PostScript fonts, respectively. The latter program 'embeds' the TrueType font in a so-called Type42-font, which can probably be considered a special kind of Type 1 font as far as your TEX implementation is concerned.

the valuable part of the font, the part that contains the letters and that you will probably have to pay for, unless it was supplied as a package deal with some piece of software. The second file you need has the extension `.afm`. It contains information about the main characteristics of the font. If you bought your font, you probably have both files available, if you got the `.pfa` or `.pfb` file as part of some other product, the `.afm` file may be missing. Fortunately, `.afm`-files are distributed for free, and can mosttimes be found at CTAN or the Internet-site of the font-supplier. If you have to obtain your `.afm`-files from one of those sources because it was not supplied with the `.pfa/b` file, make sure that not only the name but also the supplier of the `.afm`-file is identical to that of the `.pfa/b`-file (some popular fonts like Garamond are available from multiple suppliers and the Garamond from Adobe is definitely different from for instance the Monotype variant of this typeface).

A I said in the previous paragraph, you need at least two files to setup the font for usage with TEX. However, the combination of two files, although sufficient technically, can hardly be called a font. The reason is simple. Roughly speaking, in computer parlance a new font is used for each situation in which a single glyph (say an 'a') has a different shape. So your computer needs a font file for the regular a, *the italic a*, **the bold a**, and ***the bold italic a***. Consequently, what is a single font in human perception, is usually made up of at least eight files, two for the regular shaped glyphs, two for the italic, two for the bold and two for the bold italic glyphs. Provided that you have at least one single combination of a `.pfa/b` and `.afm` file, fontinst will happily process them, but you yourself won't probably be satisfied with the result.

## Renaming the font-files

Let's suppose that the font files are available. In my case I want to install an Adobe Garamond, which I got together with my scanner. I have got four couples of two files:

- gdbi___.pfb and gdbi___.afm (the bold italic glyphs),
- gdb____.pfb and gdb____.afm (the bold glyphs),
- gdi____.pfb and gdi____.afm (the italic glyphs), and
- gdrg___.pfb and gdrg___.afm (the regular shaped glyphs).

First copy those eight files to a separate directory, so that you can do no harm to the original files.

Now the most difficult part of the installation process has to be carried out. We have to find appropriate names for the font files. A first requirement is that the name you give to your fonts is unique. If you want to mess up your TEX-installation, ignore this requirement and be prepared for some rather unpleasant surprises. A second requirement, which helps you to conform to the first one is that the name you give to your font files is compliant with the font naming scheme developed by Karl Berry, which can be found at

`ftp://ftp.tex.ac.uk/tex-archive/info/fontname`.

This scheme gives an 'algorithm' for generating 'informative' names to fonts within the eight character limit imposed upon us by some old-style operating systems. The name of a font is composed as follows (for a complete and more detailed outline see Karl Berry's original article):

**The first letter** of the file name is used to indicate which font foundry produced the font we are naming. The Garamond font I am installing here, is produced by Adobe and in Karl Berry's scheme this implies that the first letter of the font name will be a `p`. Had the foundry be Linotype, the first letter would have been an `l`, Bitstream results in a `b`, Monotype in an `m`, etc.

**The second and third letter** of the name are used to condense the name of the font. In this case I choose `ad` for Adobe Garamond. A rather large, but of course incomplete, list of names is available in the original scheme.

**The fourth and possibly fifth letter** are used to indicate the shape/variant/width of the font. Again the list of possibilities is huge but when we are installing an ordinary font, the choice is not too complicated. We use the `r` for the regular shape, an `ri` for the italics, a `b` for the bold shape and `bi` for the bold italics.

**The next two letters** are used to indicate the 'font encoding'. The `pfa/b` files contain a huge number of glyphs. In principle they may be arranged in any random order. Let's for the sake of simplicity assume that the files only contain the letters a to z. One font designer may choose to make the a the first letter in his font, the b the second etc. Another may start with the z and count down. The number of possibilities is sheer endless. Without knowledge of the way the glyphs are arranged (the encoding), the font files are virtually useless. The most easy way to find out how the glyphs are arranged is to examine the contents of the `.afm`-files (`.afm`-files are regular ASCII-files, you can load them directly into your favorite editor). Those files contain information similar to that listed below:

```
StartFontMetrics 2.0
Comment  Copyright (c) 1989 Adobe Systems Incorporated.
Comment Creation Date:Wed Jul 12 19:25:18 PDT 1989
FontName AGaramond-Regular
EncodingScheme AdobeStandardEncoding
FullName Adobe Garamond Regular
FamilyName AdobeGaramond
Weight Regular
```

The information we are looking for is 'EncodingScheme AdobeStandardEncoding'. According to Karl Berry's scheme we now have to use `8a` as the next two letters of our file name. Other more or less frequently occurring coding schemes are Adobe Expert (`8x`), TeXBase1 (`8r`), and TeXnANSI (`8y`).

**Optional last letter** The optional last letter is not needed in this example, but can be used to indicate the width of a font. Some fonts have a narrow, a condensed, a wide or an extended variant in which case we have to add `n`, `c`, `w`, and `x` respectively to the file name.

Now we are ready to rename the font files into `padr8a`, `padb8a`, `padri8a`, and `padbi8a` for the regular, bold, italic and bold italic shape respectively. Although strictly speaking it is only necessary to rename the `.afm`-files, I personally prefer to rename the `pfa/b` files as well in order to retain their status as matched couples. Use the `rename` or `mv` command or whatever functionality you operating system provides to give the (copies of the) font files the appropriate names.

## Running fontinst

Now we have to prepare a TEX-file which will be used by fontinst to process our font files. The contents of this file is rather simple:

```
\input fontinst.sty
\latinfamily{pad}{}
\bye
```

The first line asks TEX to input the fontinst-utility and the next line tells fontinst to process the fontfiles. Note that the three letters `pad` are the first three letters of the name of our

fontfiles and probably will be different if you are processing your own files. Fontinst automatically takes care of the different shapes and encodings of the files to be input, with two exceptions: (1) if you want to install a family using expertfonts you have to add the letter `x` and (2) if you want to install a family using expert fonts with oldstyle digits you have to add the letter `j`. In the two latter cases the command to use will be `\latinfamily{padx}{}` and `\latinfamily{padj}{}`, respectively. Run the file through TEX, look at your terminal and wait (possibly for quite a long time).

TEX will generate an extraordinary amount of messages on your screen. Simply ignore them. The fontinst utility is trying to install almost any imaginable glyph and any imaginable variant of the font at hand. A lot of those glyphs and variants will obviously not be available to you and so fontinst complains and in some cases even finds a solution for lacking glyphs and variants: lacking ligatures will be replaced by their composing characters and slanted and fake small caps variants of a font will automatically be generated if necessary.

Usually TEX will finish without any fatal error messages and a directory listing will show you that some new files have been generated. However, ignoring typos, two situations may occur in which this is not the case. Firstly, TEX may complain about the definition of the `\bye`-command. In this case you are using an old version of fontinst. Upgrade to a newer one. Second, TEX may complain about a lack of memory. Upgrade to a larger TEX or adapt the configuration of the TEX-version you are using.[2]

## Processing the output of fontinst

Part of the output generated by fontinst needs an additional conversion before it can be used by TEX. This conversion is done by issuing the following two commands from within your command-shell (Dos-syntax on the left, Unix (bash) on the right):

```
for %f in (*.pl) do pltotf %f          for f in *.vpl; do vptovf $f;done
for %f in (*.vpl) do vptovf %f         for f in *.pl; do pltotf $f;done
```

Probably some messages about rounding errors will appear on your screen, ignore them. Just check whether some new `.tfm` and `.vf` files have been generated, which will usually be the case.[3] Again a couple of things may go wrong, in particular on Dos and Windows computers. If either pltotf or vptotf is a batchfile add `call` between do and the program name. A second possible risk is that both programs are somewhat oldfashioned and don't like the syntax used above. As you might have grasped pltotf reads a `.pl` file and converts it to a `.tfm` file. Some older pltotf's require you to enter the full name of both the `.pl` file to be read and the `.tfm` file to be generated. Similarly vptovf reads a `.vpl`-file and converts it to two other files a `.vf` and a `.tfm` file. Some oldfashioned programs, again, require you to explicitly specify the names of all files to be read and generated.

## Putting everything in place

Almost all required files have been generated by now. The main thing to be done is place them in the correct directory (when you are using a more advanced operating system like Unix, you will probably need system administrator permissions to complete this final stage). Before moving the newly generated files, let's make sure that we do not generate conflicts by using duplicate filenames. Just execute the following commands (I assume

2. How to do this is largely dependent on your TEX-implementation. If you do not want to upgrade the most easy solution may be to run the fontinst-job on someone elses computer and copy the output afterwards.
3. To be more accurate, for each of the four couples of files used in this case five `.tfm`-files will be generated and three `.vf`-files, furthermore some files for fake fonts will be made.

that you use the TEX Directory Structure; TDS, replace texmf by the appropriate directory on your machine):

```
for %f in (*.tfm) do dir texmf\%f/s          for f in *.tfm; do find /texmf -name $f;done
for %f in (*.vf) do dir texmf\%f/s           for f in *.vf; do find /texmf -name $f;done
for %f in (*.fd) do dir texmf\%f/s           for f in *.fd; do find /texmf -name $f;done
```

You have to move all .tfm-files to a subdirectory of texmf/fonts/tfm;[4] the .vf-files go to a subdirectory of texmf/fonts/vf and the .fd files to a subdirectory of texmf/tex/latex. The .pfb-files also have to be placed in an appropriate directory. Although not strictly required it might be prudent to put them into a subdirectory of texmf/fonts/type1. After moving all those files, you may have to rebuild your hash-tables by typing mktexlsr (just try it, if it is not necessary it will not harm either).

Finally you have to inform dvips about your intent to use your new font by editing psfonts.map (which is probably located in texmf/dvips/base. Add the following lines:

```
padr8r  AGaramond-Regular    "TeXBase1Encoding ReEncodeFont " <8r.enc \
                                        <padr8a.pfb
padri8r AGaramond-Italic     "TeXBase1Encoding ReEncodeFont " <8r.enc \
                                        <padri8a.pfb
padb8r  AGaramond-Bold       "TeXBase1Encoding ReEncodeFont " <8r.enc \
                                        <padb8a.pfb
padbi8r AGaramond-BoldItalic "TeXBase1Encoding ReEncodeFont " <8r.enc \
                                        <padbi8a.pfb
padro8r AGaramond-Regular    "0.167 SlantFont TexBase1Encoding ReEncod\
                                        eFont " <8r.enc <padr8a.pfb
padbo8r AGaramond-Bold       "0.167 SlantFont TeXBase1Encoding ReEncod\
                                        eFont " <8r.enc <padb8a.pfb
```

The preceding lines have been constructed as follows. The first column contains the name of the font according to the Karl Berry scheme, but this time in TexBase1Encoding (8r). The next column contains the name of the font. This name should be copied verbatim from the fontname contained in the .afm file (see the listing on page 39). The next column contains some PostScript commands that convert the font to the desired encoding. The final two columns are used to tell dvips that it should add two files to all output it generates containing the font concerned: the encoding file and the file with the font itself.

The last two lines to be added to dvips are somewhat different from the other four. Most PostScript fonts do not come with a slanted version. Those last two lines are used to generate one. The first column contains the Karl Berry name of the slanted font, the next column the fontname of the font that will be used to generate the slanted font (this same font is added to the PostScript file in the final column). The column with the PostScript commands now contains the additional text '0.167 SlantFont', that generates a slanted version on the fly.

## Using the font

Now you are ready to use the font. Just add the command

```
\renewcommand{\rmdefault}{pad}
```

to the preamble of your LATEX document and your newly installed font will be used to typeset it. Probably you will want to take some additional measures in order to ensure that your

4. The exact choice of a subdirectory does not matter, however, you may want to give it a name that makes some sense.

documents appearance will be acceptable. The Garamond font, for instance, looks absolutely ugly when used with the standard LaTeX-classes. Using the classes of the American Mathematical Society (e.g., `\usepackage{amsart}`) already is a considerable improvement. If you use a typewriter font regularly, you may want to switch to Courier instead of the regular Computer Modern typewriter, which is way too black. However, Courier is too large, so you will have to scale it. To accomplish this, I made a file `ot1zcr.fd` by renaming and modifying the `ot1pcr.fd`. By adding `\renewcommand{\ttdefault}{zcr}` to my preamble I obtain the scaled typewriter font instead of the regular one. The interpretation of the stuff in the font definition file underneath is left as an exercise to the reader (the main feature of which is a scalefactor of 89%, further information can be found in Siep Kroonenberg's article elsewhere in this issue).

```
%Filename: ot1zcr.fd
%Created by: Maarten from ot1pcr.fd

%THIS FILE SHOULD BE PUT IN A TEX INPUTS DIRECTORY

\ProvidesFile{ot1zcr.fd}
   [1997/09/30 Fontinst v1.6 font definitions for OT1/zcr.]
   % Modified by Maarten

\DeclareFontFamily{OT1}{zcr}{}

\DeclareFontShape{OT1}{zcr}{b}{n}{ <-> s * [0.89] pcrb7t }{}

\DeclareFontShape{OT1}{zcr}{b}{sc}{ <-> s * [0.89] pcrbc7t }{}

\DeclareFontShape{OT1}{zcr}{b}{sl}{ <-> s * [0.89] pcrbo7t }{}

\DeclareFontShape{OT1}{zcr}{m}{n}{ <-> s * [0.89] pcrr7t }{}

\DeclareFontShape{OT1}{zcr}{m}{sc}{ <-> s * [0.89] pcrrc7t }{}

\DeclareFontShape{OT1}{zcr}{m}{sl}{ <-> s * [0.89] pcrro7t }{}

\DeclareFontShape{OT1}{zcr}{bx}{n}{<->ssub * zcr/b/n}{}
\DeclareFontShape{OT1}{zcr}{bx}{sc}{<->ssub * zcr/b/sc}{}
\DeclareFontShape{OT1}{zcr}{bx}{sl}{<->ssub * zcr/b/sl}{}
\DeclareFontShape{OT1}{zcr}{b}{it}{<->ssub * zcr/b/sl}{}
\DeclareFontShape{OT1}{zcr}{bx}{it}{<->ssub * zcr/b/it}{}
\DeclareFontShape{OT1}{zcr}{l}{n}{<->ssub * zcr/m/n}{}
\DeclareFontShape{OT1}{zcr}{l}{sc}{<->ssub * zcr/m/sc}{}
\DeclareFontShape{OT1}{zcr}{l}{sl}{<->ssub * zcr/m/sl}{}
\DeclareFontShape{OT1}{zcr}{m}{it}{<->ssub * zcr/m/sl}{}
\DeclareFontShape{OT1}{zcr}{l}{it}{<->ssub * zcr/m/it}{}

\DeclareFontShape{OT1}{zcr}{m}{ui}{<->ssub * zcr/m/it}{}
\DeclareFontShape{OT1}{zcr}{b}{ui}{<->ssub * zcr/b/it}{}
\endinput
```