The PracTEX Journal
TPJ 2005 No 02, 2005-04-15
Rev. 2005-04-16

OpenType installation basics for ConTEXt

Adam T. Lindsay

To make a sweeping generalization, ConTEXt tends to attract people who are interested in customizing their own documents. While LATEX's use is dominated by pre-cooked document layouts, using templates for academic publishing, ConTEXt seems to attract those people looking for just a little bit more flexibility. It is not surprising, then, that many ConTEXt users – both professional and personal users – tire of dealing with the default Computer Modern font family, and look to install their own fonts in order to imbue their documents with their own personality.

Font purchasing has been a tricky business. Fonts have traditionally been available in two different formats (PostScript and TrueType) for each of two different platforms (different encapsulating file formats for PC and Mac). Periodically, modern updates to those font formats have appeared, promising new features with that new technology (e.g., Multiple Masters, GX), only to have withered and died in the marketplace. So designers and others who buy fonts may tire of the fractured market of multiple font formats, but they should understandably be a little leery of new font formats.

It is within this environment that the OpenType font format was conceived. It attempts to unify TrueType and PostScript imaging models, and unifies the file formats so that there is one file designed to work on any platform. It unifies across many encodings by utilizing Unicode as the native font encoding. It unifies across expert encodings, small caps fonts, and other fonts with alternative glyphs by introducing OpenType features, which allow fairly sophisticated glyph replacement procedures. (Much has been written on OpenType elsewhere, so I won't go on.) In short, the format is being portrayed as a "future-proof" font format: if you are buying a font today, you can feel more secure that your purchase will retain its usefulness when it is an OpenType font.

This article examines the basic steps necessary for OpenType font `.otf` installation, with a focus on ConTEXt-oriented tools. Along the way, I will give overviews of the general font installation workflow and of the TEXfont font installation script. The article assumes a fair amount of confidence at the command line and a properly-configured ConTEXt installation. The instructions are slanted toward using ConTEXt's preferred pdfTEX engine, so if you use other engines like dvips or dvipdfmx, I must assume you know why you are doing so and how to adapt such instructions for your own needs.

1 Prepare yourself

There are a few general steps when doing any ConTEXt font installation:

Collect the fonts in a directory for installation. Typically, TEXfont will take the fonts as they come, with no special naming schemes. At most, on my Macintosh, I occasionally need to change file extensions from upper to lower case (e.g., change `Optima.TTF` to `Optima.ttf`). Windows users often need to remove underscores from their filenames.

Decide on encodings and, especially with OpenType fonts, choose font features to use. Font encodings are a surprisingly complex topic, but what you most need to know is this: although OpenType fonts deal with the gigantic Unicode character set well, traditional TEX only deals with 256 characters at a time. ConTEXt and TEXfont have the `texnansi` encoding as the default, and it works well for most European languages. Again, if you have a preference other than this default, I have to assume that you have and understand your reasons.

Run TEXfont from the command line with the decisions made. This is discussed further below.

Verify the run by processing the sample file TEXfont generates.

Make typescripts for using the font repeatably within ConTEXt. Typescripts are best discussed in detail elsewhere, but I do give the basics to get you started.

2 Running TEXfont

There are four major command line options for the `texfont` command that you should remember.

```
texfont --makepath --install --vendor=<fontfoundry> --co=<fontfamily>
```

The `--makepath` option ensures that TEXfont can go ahead and create the directories so that it can put everything in its right place. The `--install` option tells TEXfont to go ahead and copy the font files into your texmf tree. The `--vendor=` argument allows you to give a label to the font's source (e.g., 'public' or the name of a font foundry, such as 'adobe'). The `--collection=` argument allows you to give a label to the specific font family (e.g., 'torunska' or 'myriadpro'). The labeling arguments don't contain critical information, but they *are* the directory names that help keep your fonts organized. In other words, you can put whatever you like, but if you want to be able to come back and understand things later, it's best to be consistent with names.

As TEXfont accepts abbreviated command line switches, this set of options is very commonly abbreviated to:

```
texfont --ma --in --ve=<fontfoundry> --co=<fontfamily>
```

If you're using TEXfont to install Type1 PostScript fonts, then you can create a temporary directory containing all of the `.pfb` and `.afm` files in one font family. Run the above command, and TEXfont will work at putting everything in its right place.

2.1 . . . for OpenType fonts

OpenType support in TEXfont builds on Eddie Kohler's excellent LCDF typetools package[1]. There should be binary installations available for most TEX distributions nowadays. The typetools, centered around doing clever things with `.otf` fonts, lend their name to the next command line option you need to know: `--lcdf`. The `--lcdf` switch basically tells TEXfont that you have a bunch of `.otf` files and you

[1] available and documented at http://www.lcdf.org/type/. Be sure to use a recent version (2.26 as of this writing)–version 2.2, in wide distribution on the Macintosh, causes problems for TEXfont users.

would like it to use the LCDF typetools as its helper[2]. Another relevant switch for OpenType fonts is `--preproc`, for pre-processing the OpenType files to make them into Type1 PostScript files[3].

The last TEXfont option that we need to know about is `--variant=`. When used with the `--lcdf` switch, this argument takes a comma-separated list of four-letter OpenType features available in the font. These features are what make the OpenType format sexy: you can use small caps, old-style figures, and fancy ligatures all from the same source font file. TEXfont treats each set of features applied to an OpenType font as a different font: TEX's historical limitations mean that it can't handle all features all at once.

You will want to enter two OpenType features almost every time in the TEXfont `--variant` option: '`liga`' and '`kern`,' meaning 'ligatures' and 'kerning.' Two other features that appear in many 'Pro' OpenType fonts are '`smcp`' and '`onum`,' or 'small caps' and 'old-style numbers.' Eddie Kohler's site gives further definitions and demonstrations of these features.

2.2 For example...

It's best to work through an example. Although the Antykwa Toruńska fonts are already available for TEX use, we're going to use them in our worked example because they act like much more expensive, commercial OpenType fonts. Download the "Antykwa Torunska Open Type" fonts from:

```
http://www.janusz.nowacki.strefa.pl/torunska-e.html
```

and unzip them. Move into the newly created `antt-otf` directory, and run TEXfont:

```
texfont --ma --in --ve=public --co=torunska --lcdf --pre --va=liga,kern
```

[2]The reason why the LCDF typetools aren't used by default is that I wrote more primitive OpenType support using FontForge into an earlier version of TEXfont. Since then, Eddie's tools have appeared and raced ahead of FontForge in terms of OpenType capabilities.

[3]Unlike other type conversion steps, this is designed to be non-lossy: the CFF format within `.otf` files is actually Type1 dressed up in different clothing. As a result, you most likely *do* want to use this switch, as the resultant `.pfb` file will be usable in many more TEXy situations. Technically, many .ttf files can be OpenType fonts as well, but they are best treated with TrueType font utilities, and are beyond the scope of this article. Also note that `.otf` files are usable without any conversion step by pdfTEX, but that since it has trouble sub-setting such fonts, this can lead to unmanageable file sizes.

TEXfont will then leap into life, scrolling by with several pages of text[4]. All you really need to know at this point is that behind the scenes, TEXfont:

- applies the features to the encoding, making replacements and ligatures as necessary and creating a new, font-specific `.enc` file,
- converts the font metrics into a form TEX can use, resulting in a `.tfm` file,
- converts the glyphs into `.pfb` files,
- makes a `.map` file[5] to associate `.tfm` files with `.pfb` and encoding files,
- puts everything into its right place, and
- creates a test file in the current directory.

Let's turn our attention to the test file. Following our instructions, you should now have a file called `texnansi-LIGA-KERN-public-torunska.tex` in your directory. Run `texexec` on that file:

```
texexec --pdf --mode=compact --once texnansi-LIGA-KERN-public-torunska.tex
```

Open the resulting PDF, and you should see page after page of 16×16 grids of characters in different weights and styles.

Let's run TEXfont once more, in order to get small caps:

```
texfont --ma --in --ve=public --co=torunska --lcdf --pre --va=liga,kern,smcp
```

2.3 Making a typescript

Let's make a quick typescript to be sure that we can use the fonts in other documents. Create a new file in a text editor, and call it '`type-torunska.tex`'. Copy the first four `\definefontsynonym` lines from the automatically-generated test file into a typescript declaration:

[4]You may notice warnings like '*GPOS Pair Positioning coverage format error*' scroll by. I believe that is an error specific to the OpenType Antykwa Toruńska fonts that will hopefully vanish in future versions.

[5]You may notice that there is what, on first glance, appears to be gobbledygook in this file near the ReEncodeFont directive. This is not nonsense or an error, but an auto-generated hash as a by-product of the LCDF typetools creating a custom mapping.

```
\starttypescript[serif][torunska][texnansi]
 \definefontsynonym[AntykwaTorunska-Bold]
  [texnansi-LIGA-KERN-AntykwaTorunska-Bold]        [encoding=texnansi]
 \definefontsynonym[AntykwaTorunska-BoldItalic]
  [texnansi-LIGA-KERN-AntykwaTorunska-BoldItalic]  [encoding=texnansi]
 \definefontsynonym[AntykwaTorunska-Italic]
  [texnansi-LIGA-KERN-AntykwaTorunska-Italic]      [encoding=texnansi]
 \definefontsynonym[AntykwaTorunska-Regular]
  [texnansi-LIGA-KERN-AntykwaTorunska-Regular]     [encoding=texnansi]
```

Add a modified line from the small caps demonstration file, and then close the definition:

```
 \definefontsynonym[AntykwaTorunska-Caps]
  [texnansi-LIGA-KERN-SMCP-AntykwaTorunska-Regular][encoding=texnansi]
\stoptypescript
```

The above typescript is known as an 'encoding' typescript, as it associates a symbolic name with an actual font file, and also tells ConTEXt to use a specific encoding. We hook into the text style system with a 'name' typescript, which associates canonical internal font names with the font-specific names:

```
\starttypescript[serif][torunska][name]
 \setups[font:fallback:serif]
 \definefontsynonym[Serif]          [AntykwaTorunska-Regular]
 \definefontsynonym[SerifItalic]    [AntykwaTorunska-Italic]
 \definefontsynonym[SerifBold]      [AntykwaTorunska-Bold]
 \definefontsynonym[SerifBoldItalic][AntykwaTorunska-BoldItalic]
 \definefontsynonym[SerifCaps]      [AntykwaTorunska-Caps]
\stoptypescript
```

Finally, we devise a short typescript that loads the font map file whenever we call the typescripts:

```
\starttypescript[map][torunska][texnansi]
 \loadmapfile[texnansi-LIGA-KERN-public-torunska.map]
 \loadmapfile[texnansi-LIGA-KERN-SMCP-public-torunska.map]
\stoptypescript
```

With the `type-torunska.tex` file containing those three basic typescripts, all that remains is to call it from a test file:

```
\usetypescriptfile[type-torunska]
\definetypeface[mine][rm][serif][torunska][default][encoding=texnansi]
\setupbodyfont [mine,12pt]
\starttext
Regular, {\it italic}, {\bf bold}, {\bi bold italics}, and {\sc small
caps}.\par
\input ward
\stoptext
```

The intricacies of typescript definitions and usage will have to be left for another article, but I hope the above provides enough of a template to get started.

3 Troubleshooting

There are a few common problems that TEXfont beginners run into. I can't address them all, but here are the two that I encounter most:

permissions If your texmf tree is only writable by a systems administrator, then you may get a message like this shortly before TEXfont gives up: *'mktexlsr: /usr/local/teTeX/share/texmf.local/ls-R: no write permission. Skipping...'* On a system such as my UNIX-like Macintosh, I would prefix the same `texfont` command with `sudo`.

unknown FONTROOT TEXfont tries really hard to find the default location for your fonts, but with some installations it still fails to find where to put the fonts. If this is the case, re-run `texfont`, adding the argument:

```
--fontroot=/path/to/your/texmflocaltree
```

4 Keep experimenting

TEXfont is a versatile tool, and it is fairly forgiving of its inputs. It works extremely well at slurping a whole directory full of fonts as they arrive straight from the

foundry. It has been adapted to OpenType fonts by leveraging the cleverness embedded in the LCDF typetools. The only thing you can destroy when experimenting is your free disk space!

If you would like to know more, the basic ConTEXt font manuals[6] are densely packed with information. There is another article discussing more advanced issues in OpenType support printed in the Dutch TEX User Group's MAPS, issue #31, and there is an earlier version of that article on my website[7]. I'm more than willing to field questions via the ConTEXt mailing list, as well, and can update this article based on users' feedback. OpenType and TEX is still a new development, and so can be subject to changing features, but I hope this article introduces users to the possibilities of this font format of the future.

[6] http://www.pragma-ade.com/general/manuals/mtexfont.pdf and http://www.pragma-ade.com/general/manuals/mfonts.pdf

[7] http://homepage.mac.com/atl/tex/