Font Installation: Agfa Eaglefeather to Linotype Zapfino

William F. Adams
ATLIS, 75 Utley Drive, Suite 110, Camp Hill, PA 17055
wadams@atlis.com

Abstract

Here's how I managed to get Agfa Eaglefeather and Linotype Zapfino installed in TEX and some of the things which I learned, from A to Z.

Beginning First, acquire the fonts. I'd purchased Agfa Eaglefeather back when it first came out to do the signage for a Frank Lloyd Wright exhibition at the Longwood Center for the Visual Arts. I've always wanted Zapfino since first hearing of it, so was thrilled to learn it was bundled with Mac OS X.

Choose your font vendor carefully. Considerations include:

Are they on the "short list" of font vendors which are listed in Fontname (`http://www.tug.org/fontname/`? This eases the process of installation, Agfa for example isn't, and I was having trouble getting the install of Eaglefeather started until I hit upon the idea of lumping it in with Monotype Octavian which I'd installed previously, on the theory that assimilation works both ways (Monotype was bought out by Agfa).

Are `.afm` (adobe font metric) files provided/ available? Not all vendors do this, especially for fonts provided in Mac format. If `.afm` files are not available, one can make one's own, with certain caveats. The typical case is a Mac `lwfn` printer font and bitmap screen font pair, in which case the `bmp2afm` utility from `CTAN` and other file archives will extract the metric and kerning information. For the Windows `.pfm` (postscript font metric) format files similar utilities are available. Additionally, most font editors can access this information and write `.afm` files. `pfaedit` is an open source font editing/creation program recently announced on SourceForge (`http://www.sourceforge.net`).

Who made the font and why? Does the font vendor acknowledge the font's designer at least? Make royalty payments? Provide information which helps in learning how the font was first used (or ideally tutorial exemplars such as the lovely booklets which accompanied the Adobe Originals), where it was first used (a few fonts are still named for the book in which they were first used...) and when so that one may write a decent colophon?

As implied above, it is the font's vendor which, in the Fontname scheme, determines the beginning of the font name. For Agfa-Monotype Eaglefeather, "M", for Apple-Linotype Zapfino, "E". the appropriately named sub-directories (mef and ezo) for the fonts are placed in directories for the relevant vendor (monotype and apple). If you are using a font from a vendor to which no letter or number is assigned, I would suggest using "z" for misc instead of attempting to assign something oneself—not only does this raise the spectre of incompatibility should the number later be formally assigned, it seems not to work without reconfiguration of the TEX Directory Structure settings.

Conversion As a graphic designer by trade, I've always purchased Mac format fonts, working on the theory it's easier to convert from Mac to PC/Unix than in other directions. (This habit was formed, of course, before `.pdf`) There are a number of tools for this, I use a combination of: Ares FontHopper (to get from Mac to PC format), and Type Designer by Manfred Albracht (to regularise the encoding and generate a `.afm`), and of course the afore-mentioned `bmp2afm` and other programs. Of notable interest for those with older TEX systems, `ttf2afm`, a part of the FreeType project allows `.dvi` processors to make use of TrueType outline fonts by converting them into bitmap fonts.

Zapfino for MacOS X is provided in the new .dfont resource fork-less format, which is as yet unique to MacOS X. Fortunately, the font tools group at Apple has created "Fork Switcher", a small utility program which switches a font from resource to data fork. Zapfino is also available from Linotype in a more prosaic Type 1, multiple font format, but the Apple version has many additional characters (a total of 1,286) and promises to eventually provide elegant access to all them and more which are yet to be created...

Fork Switcher is available from `http://fonts.apple.com`. Unfortunately, Ares FontHopper has been discontinued since Adobe purchased Ares, and DTP Software became the German distributor for Yuri Armola's FontLab program, (`http://www.pyrus.com` in the U.S., or `http://www.fontlab.com`) which subsumed the code and features of Type Designer. (Note: This latter Type Designer is *not* the font editor which once upon a time was sold by Letraset.)

Decisions Eaglefeather was rather straightforward once I'd worked out a solution to its lacking an Expert font set and having "merely" a Small Caps font. (I made my own Expert fonts. This is probably a variant of the "If all one has is a hammer..." maxim.)

I chose "ef" for the name, since it seems unlikely I'd need Egyptienne (the canonical font for this letterpair). Arguably, I should use an unused letterpair, since TEX insists on making a folder for egyptien `.tfm`s.

Zapfino has proven more complex, and even now, I've not reached an equitable solution. Unfortunately, despite its having over 1,000 unique glyphs, very few of them are actually wired up (for example, there's no way to access the old-style figures) for usage in TextEdit.app or WorldText.app (née GXWrite from Apple's QuickDraw/GX)—this program is available in the `Extras` folder of the Developers' Applications (if you didn't get a Developer CD with your copy of MacOS X, the Tools are available as a download with (free) membership in the Apple Developers' Connection) in MacOS X at this time (10.1 has not yet been released as of this writing). Sadly, this is confirmed by my usage of Zapfino in TEXGX. Although installation there was almost painfully easy (switch font fork with Fork Switcher, drag to System Folder:Fonts, start TEXGX, use font), GX does not provide access to font features for which layout tables have not been written, and at this time these are limited to the binary options (to ligature or not to ligature) which Apple's TextEdit.app and other programs which use the `nstext` object make available. One additional consideration here is that the font after conversion bears special handling since it is set for installable embedding which seems *not* to be covered by Apple's software license. Correspondence with Apple asking for guidance or clarification on this issue has thus far gone unanswered—it's worth noting though that Apple removed font tables from their system fonts so as to preclude the conversion and installation of same in other operating systems.

Eaglefeather David Siegel, the designer of this font under license from the Frank Lloyd Wright Foundation has written "The Story of Eaglefeather" which is available from his website, `http://www.dsiegel.com`. By the time this is printed a `.pdf` version set in the font itself should be available from my personal web site `http://members.aol.com/willadams`.

Font Installation: `fontinst` I must confess that I use this amazing macro package by Alan Jeffrey as a "black box" and merely feed it appropriately named collections of `.afm`s and a TEX file consisting of `\input fontinst.sty`, an appropriate `\latinfamily` command, and `\bye`. In addition to the manual, I would especially recommend reading Tako Hoekwater and Ulrik Vieth's presentation for EuroTeX '99 "Surviving the TEX font encoding mess" (filename `et99-font-tutorial.pdf`) which is included in the pre-release `fontinst` documentation directory on `CTAN`.

It is my understanding after extensive reading of the manual and the `weave`d source of the package itself (included with the pre-release version on `CTAN`, filename `fisource.pdf`) that it can be controlled by `.etx` files, and hope to eventually find the time to experiment with that. Alan Hoenig's wonderful (but sadly out-of-print) book *TEX Unbound* has extensive tutorials on this as well. One observation before I begin a description of low-level direct manipulation which a better understanding of `fontinst` might obviate: `.etx` files are created in the course of font installation by `fontinst`, and if a second (tweaked) install is attempted, they will be used in lieu of (updated) `.afm` files which have the same filename.

General Tweaking Making the O FLOAT.

This is specifically suggested for adjustment in "The Story of Eaglefeather", and I did so—couldn't resist. For those with a similar weakness, we start with the `vpl` for the roman font, and construct an LO ligature First, find an empty character slot. Since there's no "Eng" at `D 141`, we'll use that. (My apologies to those whose native tongue requires this character.)

```
(CHARACTER D 141 (COMMENT Ng)
   (CHARWD R 5.0)
   (CHARHT R 5.0)
   (CHARDP R 0.0)
   (MAP
      (SETRULE R 5.0 R 5.0)
      (SPECIAL Warning:
               missing glyph 'Eng')
      )
   )
```

and add in the small cap "o" we'll be inserting.

Find a character to replace and grab the replacement character:

```
(CHARACTER D 111 (COMMENT o)
   (CHARWD R 5.81995)
   (CHARHT R 4.8999)
   (CHARDP R 0.10999)
   (MAP
      (SETCHAR D 111) (COMMENT o)
      )
}
```

Merge the two entries:

```
(CHARACTER D 141 (COMMENT small cap o)
   (CHARWD R 5.81995)
   (CHARHT R 4.8999)
   (CHARDP R 0.10999)
   (MAP
      (SETCHAR D 111) (COMMENT o)
      )
   )
```

and then tell TEX from where to draw the character:

```
(CHARACTER D 141 (COMMENT small cap o)
   (CHARWD R 5.81995)
   (CHARHT R 4.8999)
   (CHARDP R 0.10999)
   (MAP
      (SELECTFONT D 1)
            (COMMENT mefr8x at 10.0pt)
      (SETCHAR D 111) (COMMENT o)
      )
   )
```

Unfortunately, KRN only allows us to move characters horizontally, so a character slot must be dedicated to this as far as I know. So, we create the ligature character:

```
(CHARACTER D 173 (COMMENT L_o)
   (CHARWD R 6.875)
   (CHARHT R 6.39999)
   (CHARDP R 0.0299)
   (MAP
      (PUSH)
      (MOVERIGHT R 1.75)
      (MOVEUP R 2.2)
      (SETCHAR D 111) (COMMENT o)
      (POP)
      (SETCHAR D 76) (COMMENT L)
      )
   )
```

Then we go back and insert a reasonable KRN for A. . .

After doing this, we must note that even though it was missing, “ng” was kerned (with the values for ‘n’ I suppose), so we want to delete such kerns. Similarly, we delete any kerns for where we put the “o”. This part could be automated by fontinst, I believe. . .

Kerning after the L_o pair hopefully won’t need too much adjusting.

Next, noting that “L” is CHARACTER D 76, we modify the LIGKERN entry for o which is already there, changing it from a kern to a ligature; from: (KRN D 111 R -0.48999) (COMMENT o) to: (LIG D 111 D 173) (COMMENT o L_o).

Hyphenation One of the things which fontinst doesn’t do for the fonts which it installs is a zero-width hyphen. Fortunately, as Don Hosek of Quixote http://www.quixote.com, publisher of the magazine *Serif: The Magazine of Fine Type and Typography* noted in a post to comp.text.tex ages ago, this is quite easy to do, just:

- open up the appropriate .vpl (e.g., mefr9d.vpl)
- find the “hyphenchar” if it exists, e.g.,

  ```
  (CHARACTER D 127 (COMMENT hyphenchar)
     (CHARWD R 2.90991)
     (CHARHT R 3.05994)
     (CHARDP R 0.0)
     (MAP
        (SETCHAR D 45) (COMMENT hyphen)
        )
     )
  ```

- zero out CHARWD (or adjust it to the desired value if having the hyphen hang all the way out is not desirable)
- if “hyphenchar” doesn’t exist, copy the real hyphen into a blank character slot. Note: CHARDP was −2.5 for the real thing. . . not sure why it should be zeroed out for this.

Then, in one’s TEX file, one must set the hyphenchar in each font definition macro to the correct slot, e.g., \hyphenchar\efroman = 127.

Installation Once fontinst has turned out the basic font property list files (both normal, .pl for regular fonts and .vpl for virtual fonts), one must convert these into the .tfm (TEX Font Metrics) and .vf (virtual fonts) files which TeX itself will use by using plttotf and vptovf. Doing this is specific to one’s system/TEX implementation and is well-described in the fontinst manual. There are comments which I copy which describe the appropriate command for each file, but the executable is misnamed in them (pltotfm vice pltotf and vpltovf vice vptovf).

Then, the files should be stored away in the proper places, the fonts added to the dvips config.ps file by way of an added map file and one’s TEX filename database refreshed.

Justification A thing which bears noting on TEX's method of justification is that it does not alter inter-character spacing—this ensures that ligatures will be visually compatible with the text in which they reside. Readers with access to a copy of MacOS X can graphically explore this by setting "office" in the Snyder Bold font bundled with MacOS X in TextEdit, selecting it and turning on "all ligatures".

Kerning A font installed in TEX will inherit the kerning from the relevant `.afm` files (if any). Kerning should be tested with a text which contains all of the letter triplets used in the intended language, as well as as sensible Cap/lowercase pairs. Such a file for English which makes use of Webster.app, Digital Shakespeare and the *King James Version of the Bible* is available at `http://members.aol.com/willadams/typography/textsamples`

Ligatures Only the "basic" f-ligatures are installed by `fontinst` for a typical font since those are the only ones normally available. Eaglefeather only has f-ligatures as actual glyphs, while others can be created by careful kerning of select letterpairs.

Zapfino on the other hand, has a plethora of ligatures (if only one could get at them).

Quote

> TeX started out with a highly irrational and non-composable design, which was partly helped and yet a bit furthered by the fine LATEX NFSS. Making sense out of it all is most difficult.
>
> One concludes that human language and their alphabets cannot be coldly reduced to integer codes. The letter is kept; the spirit is lost.
>
> *Dr. Richard Kinch* responding to Javier Bezos in the thread "TEX and Unicode = where are we?" in `comp.text.tex` Wednesday 25 April 2001 12:42:01 PDT

Samples and Testing The canonical way to test TEX fonts is "`tex testfont`". This provides a number of options for providing various character charts and text samples which are fairly rigorous, and quite good at exposing problems (e.g., missing accents, but only for those using plain TEX-like methods of accessing accented characters) with a font which might otherwise be left until crunch time.

Stephen Moye has also done some wonderful macros for generating specimen pages – `typespec01-10` – which are available on `CTAN` where plain TEX-contributed macros are stored.

Zapfino Zapfino had its origins in Prof. Zapf's 1944 sketchbook, when he was a mapping officer during World War II. A previous attempt to render those letterforms as type, Virtuosa Script for D. Stempel had been rather compromised by the limitations of hot metal matrices, especially the swash letters.

This design was revived when David Siegel in 1993, after working on the Euler project with Prof. Zapf, and after graduating approached him about a chaotic calligraphic typeface based upon an example done for the Society of Typographic Arts in Chicago. Remembering the page from his sketchbook, Prof. Zapf saw the chance for a design without compromises due to the advantages afforded by digital type technology.

Digitization was done by Gino Lee, but production halted due to personal problems, and languished until Prof. Zapf showed the design to Linotype. It was then rendered as a traditional, multiple alphabet typeface family.

Apple's having John Hudson of Tiro Type re-encode these multiple fonts as a single entity brings this full circle for their new MacOS X and its "ATSUI" (Apple Typographic System for Unicode Information), and Zapfino is, or rather will be, a full-featured hand-writing font...