

Using OpenType fonts in Lua \TeX

Will Robertson

June 28, 2011

1 Introduction

This is an example of using the `fontspec` package in Lua \TeX (which is \TeX running on the Lua \TeX engine). Follow along in the source to see how it works.

You're probably familiar with using pdf \TeX and selecting a few different fonts by loading a few different packages, such as `mathpazo` for Palatino and `mathptmx` for Times New Roman. Lua \TeX with the `fontspec` package allows you to use not just pre-packaged \TeX fonts, but almost every font installed in your computer.

To start, put `\usepackage{fontspec}` in the preamble of your document. At the very beginning, you'll be interested in using three main commands:

```
\setmainfont      \setsansfont      \setmonofont
```

These three commands specify the fonts to use for the default roman, sans serif, and typewriter (or monospaced) fonts, respectively. They all take the same arguments: one optional argument to specify font features and a mandatory argument for the font name. A typical example is

```
\setmainfont[Numbers=OldStyle]{TeX Gyre Pagella}
```

2 Installing fonts

The first time that you run `fontspec`, a database of font names must be created based on the fonts installed in your computer. This can take several minutes, so don't worry if it seems like Lua \TeX has hung while compiling your document. It's only slow the first time.

Lua \TeX will find fonts in the following folders:

```
~/Library/Fonts      /Library/Fonts
/System/Library/Fonts /Network/Library/Fonts
```

as well as in any folders searched by \TeX , including

```
~/Library/texmf/fonts/opentype/*/*  
/usr/local/texlive/texmf-local/fonts/opentype/*/*
```

Lua \TeX is able to load fonts with extensions .otf, .ttf, and .dfont. Unfortunately many of the bundled fonts in Mac OS X 10.6 (Snow Leopard) are distributed as .ttc fonts, which currently do not work in Lua \TeX . These include ‘Hoefler Text’, ‘Cochin’, ‘American Typewriter’, and others.

3 Loading fonts

Fonts may be loaded by font name or by file name. For example, the following two commands are equivalent:

```
\setmainfont{TeX Gyre Pagella}  
\setmainfont{texgyrepagella-regular.otf}
```

However, when you load fonts by their name (not filename), fontspec will attempt to automatically locate any accompanying bold and italic fonts. If you need to specify these manually, the syntax is

```
\setmainfont [  
  ItalicFont      = texgyrepagella-italic.otf ,  
  BoldFont        = texgyrepagella-bold.otf ,  
  BoldItalicFont  = texgyrepagella-bolditalic.otf ,  
] {texgyrepagella-regular.otf}
```

See the fontspec documentation for further details.

4 Font features

OpenType fonts can be created with in-built font features that allow customisation in the way the font looks. For example, your font might have both lining numerals (0123456789) and old-style numerals (012356789). You would choose between these with either, respectively,

```
\setmainfont[Numbers=Lining]{TeX Gyre Pagella}  
\setmainfont[Numbers=OldStyle]{TeX Gyre Pagella}
```

\TeX users are used to typing punctuation with keyboard symbols such as ‘dash dash dash’ (---) for em-dash (—), and straight quotes (‘ ’) for curly ones (‘ ’). To get the same effects when using Lua \TeX fonts, use the following feature:

```
\setmainfont[Ligatures=TeX]{TeX Gyre Pagella}
```

Unlike regular \TeX , you may also scale the relative sizes of the fonts as you load them. The amount of scaling can be specified either numerical (`Scale=1.1`) or `fontspec` can automatically scale a font to match either the uppercase or lowercase letters of the default font; either one of

```
\setmainfont[Scale=MatchUppercase]{TeX Gyre Pagella}  
\setmainfont[Scale=MatchLowercase]{TeX Gyre Pagella}
```

All of the font features above can be combined as you like, just separate multiple features with commas:

```
\setsansfont  
[Ligatures=TeX, Scale=MatchLowercase]  
{Latin Modern Sans}
```

There are many, many font features you can choose from. See the `fontspec` documentation for a full listing.

5 Using Unicode

$\text{Lua}\TeX$ fully supports Unicode input, which makes it ideal for multilingual documents. As a very brief example, consider the following:

ein Apfel ein Tag hält den Doktor weg
ένα μήλο ημερησίως κρατά το γιατρό μακριά
яблоко день держит доктора отсутствующе

Take a look at the source to this document; it looks exactly like the output; each letter can be typed directly without needing macros for the accents or letters.

Multilingual typesetting can become a little more involved than just typing in the Unicode letters; it's important to adjust the hyphenation for different languages and it's often necessary to switch fonts for different scripts. But you get the idea.