XeTeX: TeX *plus* Unicode *plus* OpenType… *minus* complexity

Jonathan Kew

SIL International

November 17, 2007

History

April 2004: first public release of XeTeX 0.3 (for Mac OS X only)

- built-in support for Unicode
- access to all fonts installed on the user's computer
- AAT (Apple Advanced Typography) for typographic features

History

April 2004: first public release of XeTeX 0.3 (for Mac OS X only)

- built-in support for Unicode
- access to all fonts installed on the user's computer
- AAT (Apple Advanced Typography) for typographic features

February 2005: XeTeX reaches version 0.9

- OpenType layout features
- mature LaTeX support packages

History

April 2004: first public release of XeTeX 0.3 (for Mac OS X only)

- built-in support for Unicode
- access to all fonts installed on the user's computer
- AAT (Apple Advanced Typography) for typographic features

February 2005: XeTeX reaches version 0.9

- OpenType layout features
- mature LaTeX support packages

April 2006 (BachoTeX): XeTeX for Linux (first public release)

June 2006: XeTeX built for Windows by Akira Kakuto

History

April 2004: first public release of XeTeX 0.3 (for Mac OS X only)

- built-in support for Unicode
- access to all fonts installed on the user's computer
- AAT (Apple Advanced Typography) for typographic features

February 2005: XeTeX reaches version 0.9

- OpenType layout features
- mature LaTeX support packages

April 2006 (BachoTeX): XeTeX for Linux (first public release)

June 2006: XeTeX built for Windows by Akira Kakuto

February 2007: TeX Live 2007 includes XeTeX for all platforms

- many thanks to Karl Berry, and all TL builders and testers!

Character sets: **7 bits**, 8 bits, 16 bits, more…

The original TeX82 program used a 7-bit input text encoding, and supported 8-bit character codes internally for font access (up to 256 characters per font, though Knuth's own fonts only used 128).

A 7-bit input encoding, supplemented by conventions such as ``` ``...'' ``` for "…", `---` for —, etc., was designed primarily for English typesetting.

For European languages needing accented letters, etc., the Computer Modern fonts and Plain TeX supported various macros (`\'a \`e \u\i \^o \~u` etc.), which became a de facto standard for representing such letters.

Adaptations (e.g., Babel) for other languages sometimes added language-specific macros such as the active `"` character to provide more convenient input.

Character sets: 7 bits, **8 bits**, 16 bits, more…

TeX 3.0 extended the program to have 8-bit character support throughout, including text input as well as font access.

Simplified the support of European languages, including input text using 8-bit national codepages, and hyphenation of words including accented characters.

For multilingual documents, even using Latin script, it can be difficult to find an 8-bit encoding that provides all the necessary characters.

Adding non-Latin scripts to the mix still leads to great complexity: a variety of custom encodings, fonts, pre-processors, macros, etc.

Character sets: 7 bits, 8 bits, **16 bits**, more…

Unicode was initially conceived as a 16-bit character set, supporting up to 65,536 characters, that would include all the characters of all the world's scripts, eliminating the need for multiple codepages, and simplifying data interchange.

The major early effort to support Unicode in a TeX system was Omega, which was initially a 16-bit system (though it has since grown). However, Omega never really achieved stability and widespread use.

With time, it became clear that 65,536 characters are not sufficient, and Unicode was extended; it is now *incorrect* to think of Unicode as a 16-bit character set.

Character sets: 7 bits, 8 bits, 16 bits, **more...**

The Unicode standard now defines around 100,000 distinct characters, and the number is still growing. The largest possible Unicode character code is U+10FFFF, which requires 21 bits to represent as a simple integer.

There are multiple ways to represent these codes in a file, with UTF-8 (a variable-length sequence of 8-bit bytes) usually being the most convenient.

One key feature of the UTF-8 encoding form for Unicode is that the 7-bit ASCII character set is unchanged; the exact same byte values are still used to represent these 128 characters.

Both XeTeX and luaTeX now support the full Unicode character set. Although they use different internal representations, both read UTF-8 text files natively.

Typesetting Unicode text with XeTeX

Accented characters (many more than in any legacy codepage):

```
\font\txt="Charis SIL" at 14pt \txt
Óðinn átti tvá brœðr. Hét annarr Vé, en annarr Vílir.
Hej Slované, ještě naše slovanská řeč žije.
Dünyayı verelim çocuklara hiç değilse bir günlüğüne.
Kuř béga Šešùpė, kuř Nẽmunas tẽka, taĩ mũsų tėvỹnė, ...
```

Typesetting Unicode text with XeTeX

Accented characters (many more than in any legacy codepage):

```
\font\txt="Charis SIL" at 14pt \txt
Óðinn átti tvá brœðr. Hét annarr Vé, en annarr Vílir.
Hej Slované, ještě naše slovanská řeč žije.
Dünyayı verelim çocuklara hiç değilse bir günlüğüne.
Kuř béga Šešùpė, kuř Nẽmunas tẽka, taĩ mũsų tėvỹnė, ...
```

Óðinn átti tvá brœðr. Hét annarr Vé, en annarr Vílir.

Hej Slované, ještě naše slovanská řeč žije.

Dünyayı verelim çocuklara hiç değilse bir günlüğüne.

Kuř béga Šešùpė, kuř Nẽmunas tẽka, taĩ mũsų tėvỹnė, gražì Lietuvà.

Complex scripts: Arabic, Indic, etc.

As well as the extended Latin character set, complex non-Latin scripts are supported through industry-standard Unicode encoding and fonts:

```
\fontspec{Charis SIL}Русский
\fontspec{Devanagari MT}हिन्दी
\fontspec{STKaiti}汉语
\fontspec[Script=Arabic]{Noori Nastaliq MT}
    \beginR اُردُو قيلعتسن \endR
\fontspec{AppleGothic}한국어
\fontspec{Gentium}Ελληνικά
\fontspec{Lucida Grande}עברית
\fontspec{Hiragino Mincho Pro}日本語
```

Complex scripts: Arabic, Indic, etc.

As well as the extended Latin character set, complex non-Latin scripts are supported through industry-standard Unicode encoding and fonts:

```
\fontspec{Charis SIL}Русский
\fontspec{Devanagari MT}हिन्दी
\fontspec{STKaiti}汉语
\fontspec[Script=Arabic]{Noori Nastaliq MT}
    \beginR اُردُو نستعلیق \endR
\fontspec{AppleGothic}한국어
\fontspec{Gentium}Ελληνικά
\fontspec{Lucida Grande}עברית
\fontspec{Hiragino Mincho Pro}日本語
```

Русский
हिन्दी
汉语
اُردُو نستعلیق
한국어
Ελληνικά
עברית
日本語

Far Eastern scripts (Chinese, Japanese, Korean)

They're just more characters; no special effort required:

Far Eastern scripts (Chinese, Japanese, Korean)

They're just more characters; no special effort required:

```
\font\han="STSong" at 16pt
\font\rom="Gentium" at 8pt

\def\hc#1#2{\vbox{\hbox{\han #1}
   \hbox{\kern10pt\rom #2}}}

\vbox{\hc{書く}{ka-ku}
   \hc{最も}{motto-mo}
   \hc{最後}{sai-go}
   \hc{働く}{hatara-ku}
   \hc{海}{umi}}
```

Far Eastern scripts (Chinese, Japanese, Korean)

They're just more characters; no special effort required:

```
\font\han="STSong" at 16pt
\font\rom="Gentium" at 8pt

\def\hc#1#2{\vbox{\hbox{\han #1}
   \hbox{\kern10pt\rom #2}}}

\vbox{\hc{書く}{ka-ku}
   \hc{最も}{motto-mo}
   \hc{最後}{sai-go}
   \hc{働く}{hatara-ku}
   \hc{海}{umi}}
```

書く
ka-ku
最も
motto-mo
最後
sai-go
働く
hatara-ku
海
umi

Vertical text with XeTeX

```
\font\body="STKaiti:vertical" at 7pt \body
\font\bold="STHeiti:vertical" at 7pt
\font\title="STHeiti:vertical" at 10pt
\centerline{\title 三　国　演　义} \medskip
\centerline{\bold〔明〕罗贯中} \smallskip
\leftline{词曰：}
滚滚长江东逝水，浪花淘尽英雄。是非成败转头空：青山依旧在，几度夕阳红。
白发渔樵江渚上，惯看秋月春风。一壶浊酒喜相逢：古今多少事，都付笑谈中。
\medskip
\centerline{\bold 第一回} \smallskip
\centerline{\bold 宴桃园豪杰三结义　　斩黄巾英雄首立功}\medskip
话说天下大势，分久必合，合久必分：周末七国分争，并入于秦；及秦灭之后，
楚、汉分争，又并入于汉；汉朝自高祖斩白蛇而起义，一统天下，后来光武中兴
，传至献帝，遂分为三国。推其致乱之由，殆始于桓、灵二帝。桓帝禁锢善类，
崇信宦官。及桓帝崩，灵帝即位，大将军窦武、太傅陈蕃，共相辅佐；时有宦官
曹节等弄权，窦武、陈蕃谋诛之，机事不密，反为所害，中涓自此愈横。\par
```

Vertical text with XeTeX

```
\font\body="STKaiti:vertical" at 7pt \body
\font\bold="STHeiti:vertical" at 7pt
\font\title="STHeiti:vertical" at 10pt
\centerline{\title 三　国　演　义} \medskip
\centerline{\bold〔明〕罗贯中} \smallskip
\leftline{词曰：}
滚滚长江东逝水，浪花淘尽英雄。是非成败转头空：青山依旧在，几度夕阳红。
白发渔樵江渚上，惯看秋月春风。一壶浊酒喜相逢：古今多少事，都付笑谈中。
\medskip
\centerline{\bold 第一回} \smallskip
\centerline{\bold 宴桃园豪杰三结义　　斩黄巾英雄首立功}\medskip
话说天下大势，分久必合，合久必分：周末七国分争，并入于秦；及秦灭之后，
楚、汉分争，又并入于汉；汉朝自高祖斩白蛇而起义，一统天下，后来光武中兴
，传至献帝，遂分为三国。推其致乱之由，殆始于桓、灵二帝。桓帝禁锢善类，
崇信宦官。及桓帝崩，灵帝即位，大将军窦武、太傅陈蕃，共相辅佐；时有宦官
曹节等弄权，窦武、陈蕃谋诛之，机事不密，反为所害，中涓自此愈横。\par
```

三　国　演　义

〔明〕罗贯中

词曰：

滚滚长江东逝水，浪花淘尽英雄。是非成败转头空：青山依旧在，几度夕阳红。

白发渔樵江渚上，惯看秋月春风。一壶浊酒喜相逢：古今多少事，都付笑谈中。

第一回

宴桃园豪杰三结义　　斩黄巾英雄首立功

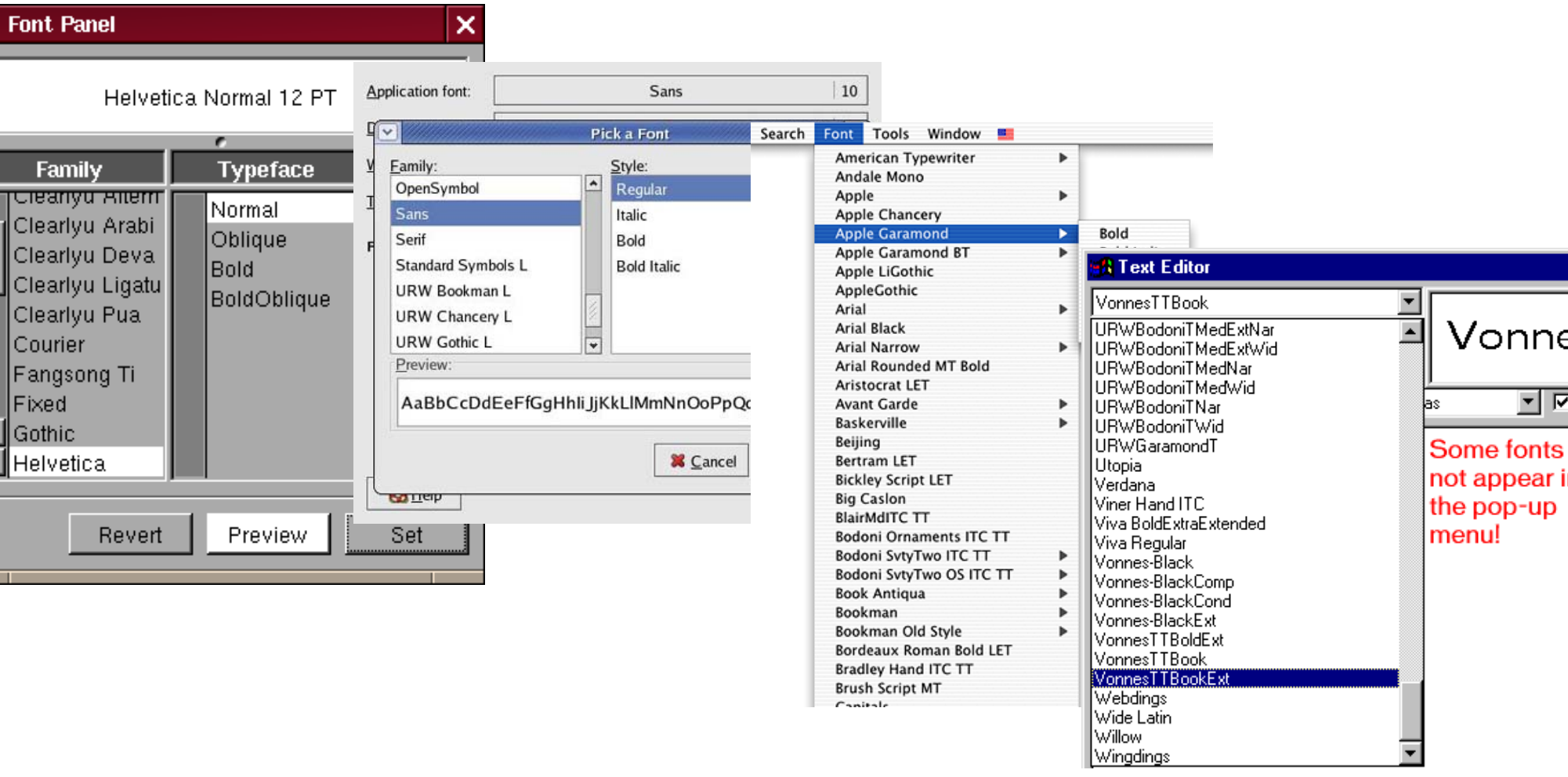
话说天下大势，分久必合，合久必分：周末七国分争，并入于秦；及秦灭之后，楚、汉分争，又并入于汉；汉朝自高祖斩白蛇而起义，一统天下，后来光武中兴，传至献帝，遂分为三国。推其致乱之由，殆始于桓、灵二帝。桓帝禁锢善类，崇信宦官。及桓帝崩，灵帝即位，大将军窦武、太傅陈蕃，共相辅佐；时有宦官曹节等弄权，窦武、陈蕃谋诛之，机事不密，反为所害，中涓自此愈横。

Font access

How difficult should it be to specify the font you want to use?

Font access

How difficult should it be to specify the font you want to use?



Font access

How difficult should it be to specify the font you want to use?

```
\font\helv = phvr8t
\font\timesrom = ptmr8t
\font\timesital = ptmri8t
\font\charter = bchr8t at 12pt
```

Font access

How difficult should it be to specify the font you want to use?

```
\font\helv = "Helvetica"
\font\timesrom = "Times New Roman"
\font\timesital = "Times New Roman Italic"
\font\charter = "Bitstream Charter" at 12pt

\font\garamond = "Adobe Garamond Pro"
\font\chancery = "Apple Chancery"
\font\dingbats = "Zapf Dingbats"
\font\japanese = "Hiragino Mincho Pro W3"
\font\chinese = "STKaiti" at 16pt
\font\arabic = "Scheherazade"
```

Advanced features of modern fonts

Professional-quality OpenType fonts may include many features that were formerly available only as separate, custom-encoded ‘expert’ fonts, such as small capitals, lining and oldstyle numerals, extra ligatures and swash glyphs, etc.

Advanced features of modern fonts

Professional-quality OpenType fonts may include many features that were formerly available only as separate, custom-encoded ‘expert’ fonts, such as small capitals, lining and oldstyle numerals, extra ligatures and swash glyphs, etc.

A font specification in XeTeX can include ‘tags’ that enable such features, if supported by the font:

```
\font\A="Adobe Garamond Pro"
\A Adobe Garamond Pro: 12345
\font\B="Adobe Garamond Pro:+smcp"
\B Garamond with Small Caps
\font\C="Adobe Garamond Pro:+onum"
\C Oldstyle Numerals: 12345
```

Advanced features of modern fonts

Professional-quality OpenType fonts may include many features that were formerly available only as separate, custom-encoded 'expert' fonts, such as small capitals, lining and oldstyle numerals, extra ligatures and swash glyphs, etc.

A font specification in XeTeX can include 'tags' that enable such features, if supported by the font:

```
\font\A="Adobe Garamond Pro"
\A Adobe Garamond Pro: 12345
\font\B="Adobe Garamond Pro:+smcp"
\B Garamond with Small Caps
\font\C="Adobe Garamond Pro:+onum"
\C Oldstyle Numerals: 12345
```

Adobe Garamond Pro: 12345

Garamond with Small Caps

Oldstyle Numerals: 12345

OpenType families with optical sizes

Where an OpenType font family includes optical size information, XeTeX will automatically load the appropriate face for the size being used:

```
\font\A="Brioso Pro" at 7pt
\A This is \fontname\A

\font\B="Brioso Pro" at 11pt
\B This is \fontname\B

\font\C="Brioso Pro" at 16pt
\C This is \fontname\C

\font\D="Brioso Pro" at 24pt
\D This is \fontname\D
```

OpenType families with optical sizes

Where an OpenType font family includes optical size information, XeTeX will automatically load the appropriate face for the size being used:

```
\font\A="Brioso Pro" at 7pt
\A This is \fontname\A

\font\B="Brioso Pro" at 11pt
\B This is \fontname\B

\font\C="Brioso Pro" at 16pt
\C This is \fontname\C

\font\D="Brioso Pro" at 24pt
\D This is \fontname\D
```

This is "Brioso Pro Caption" at 7.0pt

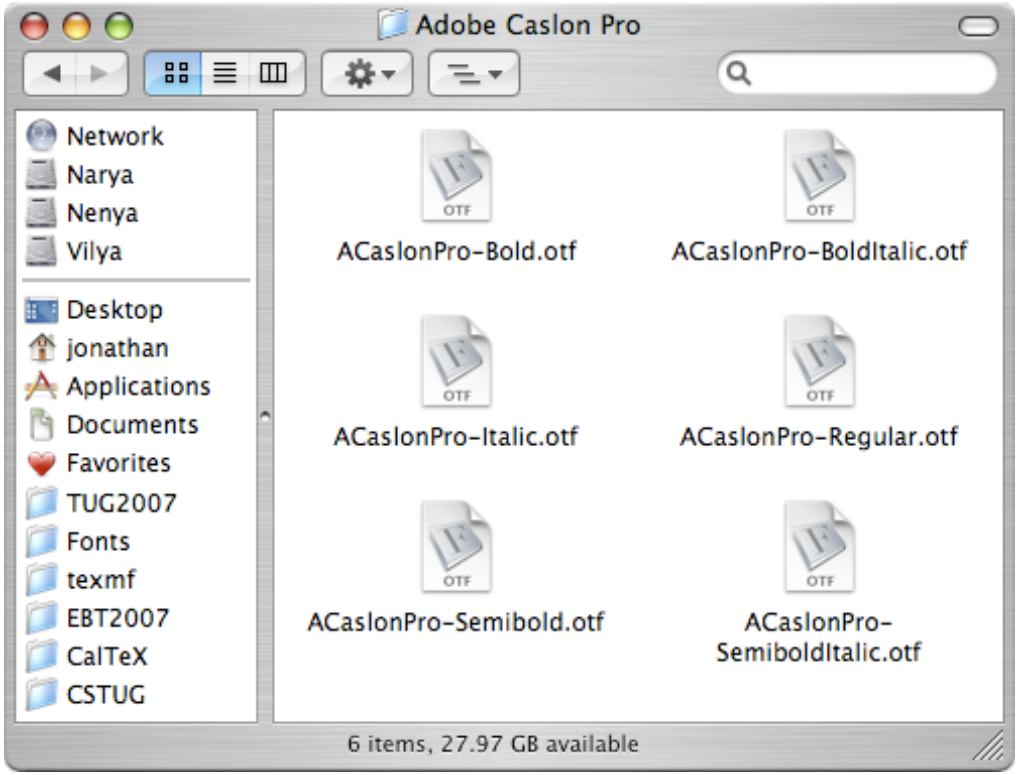
This is "Brioso Pro" at 11.0pt

This is "Brioso Pro Subhead" at 16.0pt

This is "Brioso Pro Display" at 24.0pt

Installing a new font

How about installing a new font, purchased or downloaded as an OpenType or TrueType file?



Installing a new font

How about installing a new font, purchased or downloaded as an OpenType or TrueType file?

The traditional way:

- Rename font files according to a cryptic convention
- Possibly convert fonts to Type 1 (`.pfb`) format
- Generate metrics and virtual font descriptions
- Compile the metrics and virtual fonts into binary form
- Install fonts, metrics, virtual fonts into proper subdirectories of the texmf hierarchy
- Create map files for dvips, pdftex, dvipdfm, xdvi, …
- Install map files and add entries to the drivers' configurations
- Update hash tables if installing system-wide

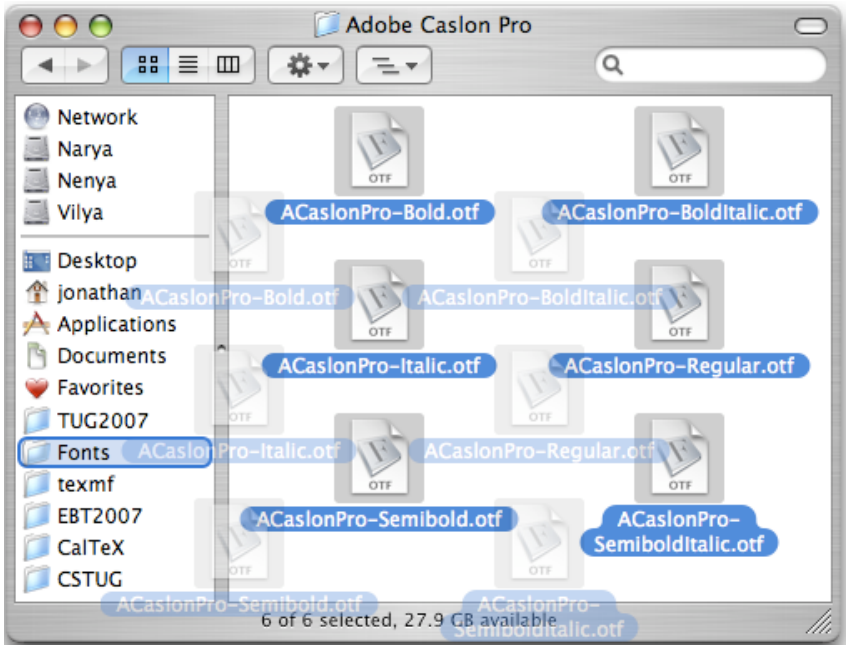
See the 100-page *Font Installation Guide* and other documents for more details.

Installing a new font

How about installing a new font, purchased or downloaded as an OpenType or TrueType file?

The XeTeX way:

- Drag the font files to the system's Fonts folder



- Ready to use via `\font\caslon = "Adobe Caslon Pro"`

Using LaTeX with the XeTeX engine

Simply run `xelatex` instead of `pdflatex`.

Using LaTeX with the XeTeX engine

Simply run `xelatex` instead of `pdflatex`.

- normal LaTeX document classes and packages are used to format the document
- the key difference is that Unicode text and fonts can be used with no `\inputenc` package or special font setup

Using LaTeX with the XeTeX engine

Simply run `xelatex` instead of `pdflatex`.

- normal LaTeX document classes and packages are used to format the document
- the key difference is that Unicode text and fonts can be used with no `\inputenc` package or special font setup

LaTeX is primarily a document markup and formatting language built on TeX; meanwhile, XeTeX extends the TeX engine to better handle Unicode and fonts. So they usually work well together.

Using LaTeX with the XeTeX engine

Simply run `xelatex` instead of `pdflatex`.

- normal LaTeX document classes and packages are used to format the document
- the key difference is that Unicode text and fonts can be used with no `\inputenc` package or special font setup

LaTeX is primarily a document markup and formatting language built on TeX; meanwhile, XeTeX extends the TeX engine to better handle Unicode and fonts. So they usually work well together.

Common problem areas include:

- 8-bit text encodings supported via `inputenc` (convert to Unicode instead)
- custom-encoded fonts that are not Unicode-compliant
- packages that depend on a specific device driver (usually drawing packages that use PostScript output)

Using LaTeX with the XeTeX engine

```
\documentclass{beamer}
\usetheme{Singapore}
\usefonttheme{serif} \usecolortheme{crane}

\usepackage{fontspec}
\setmainfont{Optima}
\setmonofont[Scale=MatchLowercase]{Andale Mono WT J}

\usepackage{xunicode,xltxtra,url,parskip,mdwlist}

\title{\XeTeX:\\
  $\phantom{\hbox{…}}$\TeX\ \emph{plus} Unicode
       \emph{plus} OpenType…\\ \emph{minus} complexity}
\author{Jonathan Kew}
\institute{SIL International}
\date{November 17, 2007}

\begin{document}
\maketitle
```

Using LaTeX with the XeTeX engine

XeTeX:
TeX *plus* Unicode *plus* OpenType…
minus complexity

Jonathan Kew

SIL International

November 17, 2007

fontspec: a simple interface to fonts

XeTeX makes it easy to access a single font, at a specific size, by using the font name in a `\font` command.

However, LaTeX really needs more than this. To change the font of a LaTeX document, we may need to change not only the Roman but also the Bold and Italic faces, used for things like `\emph` or headings, at many different sizes (`\normalsize`, `\small`, `\footnotesize`, etc.)

Normally this is handled by a font-specific package such as `garamond.sty`, together with `.fd` files that link font family abbreviations, style options, encodings, and sizes to specific real or virtual fonts. Creating all this infrastructure accounts for much of the complexity of LaTeX font setup.

fontspec: a simple interface to fonts

The `fontspec` package by Will Robertson eliminates all this complexity for XeTeX users, by providing high-level font selection macros that rely on XeTeX's capabilities behind the scenes.

fontspec: a simple interface to fonts

The `fontspec` package by Will Robertson eliminates all this complexity for XeTeX users, by providing high-level font selection macros that rely on XeTeX's capabilities behind the scenes.

To change the typefaces used in a LaTeX document, a typical preamble can simply include a few lines such as:

```
\usepackage{fontspec}

\setmainfont{Adobe Caslon Pro}
\setsansfont{Lucida Sans}
\setmonofont{Lucida Console}
```

(Usually, the option `[Mapping=tex-text]` should be added to the declarations, for compatibility with TeX typing conventions.)

fontspec: a powerful interface to fonts

Many additional options are available to access advanced features of OpenType fonts. For example, it is possible to:

- specify default features to be applied to all fonts
- automatically scale typefaces to match ex-height, cap-height, etc.
- define new font family commands (like `\rmfamily`, `\sffamily`, etc.) that respect the LaTeX environment but access any available fonts
- precisely control which faces are used for Bold, Italic, etc., in a complex font family
- specify script- and language-specific options for fonts that support these features

See the extensive `fontspec` documentation for more details.

xunicode for compatibility

To use LaTeX input conventions for accents and other 'special' characters with Unicode-encoded fonts, we need to redefine these macros to access the correct Unicode codepoints, otherwise LaTeX will try to print them from a variety of custom-encoded fonts.

Ross Moore's package `xunicode` does this for many common LaTeX control sequences; it includes a couple of thousand definitions such as:

```
\DeclareEncodedCompositeCharacter{\UTFencname}{\`}{0300}{02CB}  % Combining grave accent
\DeclareEncodedCompositeCharacter{\UTFencname}{\'}{0301}{02CA}  % Combining acute accent
\DeclareEncodedCompositeCharacter{\UTFencname}{\^}{0302}{02C6}  % Combining circumflex accent
\DeclareEncodedCompositeCharacter{\UTFencname}{\~}{0303}{02DC}  % Combining tilde
\DeclareEncodedCompositeCharacter{\UTFencname}{\=}{0304}{02C9}  % Combining macron

\DeclareUTFcharacter[\UTFencname]{x00A1}{\textexclamdown}
\DeclareUTFcharacter[\UTFencname]{x00A2}{\textcent}
\DeclareUTFcharacter[\UTFencname]{x00A3}{\textsterling}
\DeclareUTFcharacter[\UTFencname]{x00A4}{\textcurrency}
\DeclareUTFcharacter[\UTFencname]{x00A5}{\textyen}
```

arabxetex

Package by François Charette providing an ArabTeX-like interface for typesetting languages in Arabic script with XeLaTeX. Input text is either Unicode Arabic or a Latin transliteration.

arabxetex

Package by François Charette providing an ArabTeX-like interface for typesetting languages in Arabic script with XeLaTeX. Input text is either Unicode Arabic or a Latin transliteration.

مِنَ ٱلْقُرآنِ ٱلْكَرِيمِ، سُورَةُ ٱلسَّجْدَةِ ١٥–١٦:

إِنَّمَا يُؤْمِنُ بِئَايـٰتِنَا ٱلَّذِينَ إِذَا ذُكِّرُوا بِهَا خَرُّوا سُجَّداً وَسَبَّحُوا بِحَمْدِ رَبِّهِمْ وَهُمْ لَا يَسْتَكْبِرُونَ ۩ ﴿١٥﴾
تَتَجَافَىٰ جُنُوبُهُمْ عَنِ ٱلْمَضَاجِعِ يَدْعُونَ رَبَّهُمْ خَوْفاً وَطَمَعاً وَمِمَّا رَزَقْنٰهُمْ يُنْفِقُونَ ﴿١٦﴾

arabxetex

Package by François Charette providing an ArabTeX-like interface for typesetting languages in Arabic script with XeLaTeX. Input text is either Unicode Arabic or a Latin transliteration.

مِنَ ٱلْقُرآنِ ٱلْكَرِيمِ، سُورَةُ ٱلسَّجْدَةِ ١٥–١٦:

إِنَّمَا يُؤْمِنُ بِئَايـٰتِنَا ٱلَّذِينَ إِذَا ذُكِّرُوا بِهَا خَرُّوا سُجَّداً وَسَبَّحُوا بِحَمْدِ رَبِّهِمْ وَهُمْ لَا يَسْتَكْبِرُونَ ۩ ﴿١٥﴾
تَتَجَافَىٰ جُنُوبُهُمْ عَنِ ٱلْمَضَاجِعِ يَدْعُونَ رَبَّهُمْ خَوْفاً وَطَمَعاً وَمِمَّا رَزَقْنٰهُمْ يُنْفِقُونَ ﴿١٦﴾

اَنبیاءِ بنی اِسرائیل وِچوں حضرت یُونُس ہِک جلیل القدر نبی ہن۔ آپ دا دَورِ نبوّت ۷۸۲ /۷۸۱ق م کنُوں ۷۵۳ق م تک ہے۔ اُوں ویلے مُملکتِ بنی اِسرائیل دا بادشاہ یُربعام ثانی ہا۔ آپ اِیں مُملکت وِچ ناصرت دے نال "جات حِفر" دے مقام تے رَہندے ہَن۔اَج کَل اِیں جاہ کُوں "خربۃُالزُّرَع" آہدِن۔ ایہ مقام جِھیلِ جلیل دے مغربی پاسے بارَھاں مِیل اتیں ناصرۃ دے شُمال مشرقی پاسے ترائے مِیل دے فاصلے تے ہے۔ اِتّھوں دے کھنڈرات دی کُھدائی دے دَوران پتّا لگّے جو ایہ مقام کانسی دے آخری دَور تقریباً ۱۵۵۰ق م کنُوں ۱۲۰۰ق م تک آباد ہا۔ وَلا ڈُوجھی دفعہ ایہ تقریباً ۱۲۰۰ق م کنُوں ۶۰۰ق م تک آباد ریہا۔ حضرت یُونُس ہیں ڈُوجھے دَور وِچ ہَن۔ اِنہیں کھنڈرات کنُوں ذرا شُمالی پاسے ہِک وستی جیندا ناں ہے "مشہد"۔ روایت دے مُطابق حضرت یُونُس دا مزار اِتھائیں ہے۔

jspacing, zhspacing

`jspacing`: package by Miyata Shigeru implementing detailed rules for Japanese character spacing (e.g., around punctuation marks).

> ユニコードは、プラットフォームに係わらず、プログラムに係わらず、言語に係わらず、すべての文字に独立した番号を与えます。ユニコード標準は、アップル、ヒューレットパッカード、IBM、ジャストシステム、マイクロソフト、オラクル、SAP、サン、サイベースなどの産業界の主導的企業と他の多くの企業に採用されています。

jspacing, zhspacing

`jspacing`: package by Miyata Shigeru implementing detailed rules for Japanese character spacing (e.g., around punctuation marks).

ユニコードは、プラットフォームに係わらず、プログラムに係わらず、言語に係わらず、すべての文字に独立した番号を与えます。ユニコード標準は、アップル、ヒューレットパッカード、IBM、ジャストシステム、マイクロソフト、オラクル、SAP、サン、サイベースなどの産業界の主導的企業と他の多くの企業に採用されています。

ユニコードは、プラットフォームに係わらず、プログラムに係わらず、言語に係わらず、すべての文字に独立した番号を与えます。ユニコード標準は、アップル、ヒューレットパッカード、IBM、ジャストシステム、マイクロソフト、オラクル、SAP、サン、サイベースなどの産業界の主導的企業と他の多くの企業に採用されています。

`zhspacing`: similar package for Chinese, by Yin Dian.

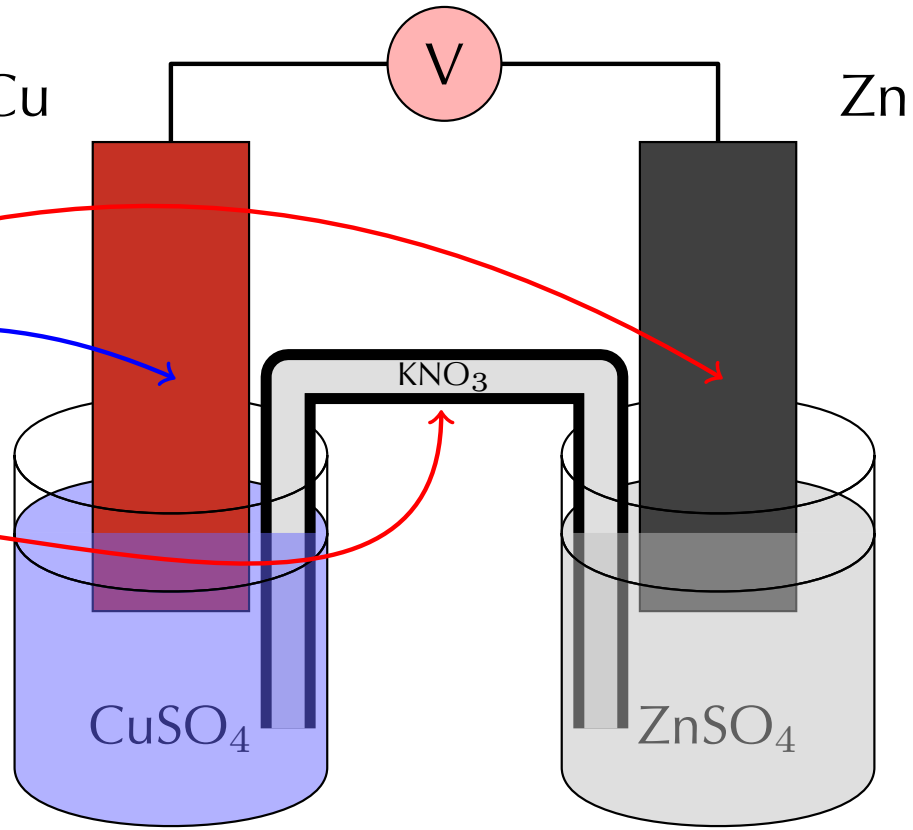
Drawing packages such as T*ikZ*

T*ikZ* works because `pgf` supports the `dvipdfm(x)` driver:

Drawing packages such as T*ikZ*

T*ikZ* works because `pgf` supports the `dvipdfm(x)` driver:

- Anode
- Cathode
- Saline bridge



(example figure from `http://www.fauskes.net/nb/pgftikzexamples/`)

driver-dependent packages

Many driver-dependent packages now support XeTeX, so that they work correctly regardless of the TeX engine used.

graphics	supported via the `xetex.def` driver (by Ross Moore)
color	supported via the `xetex.def` driver
hyperref	recognizes XeTeX and uses appropriate `\specials`
geometry	configuration file treats XeTeX like pdfTeX
crop	configuration file treats XeTeX like pdfTeX
pdfpages	supports the XeTeX primitives for PDF inclusion
pgf/TikZ	configuration file treats XeTeX like dvipdfm
pstricks	custom driver (by Miyata Shigeru) works with xdvipdfmx

Using XeTeX with ConTeXt

`texexec` supports an `--xtx` option to use XeTeX in place of the default pdfTeX engine. This allows the use of XeTeX font names and other options; see the `contextgarden.net` wiki.

Using XeTeX with ConTeXt

`texexec` supports an `--xtx` option to use XeTeX in place of the default pdfTeX engine. This allows the use of XeTeX font names and other options; see the `contextgarden.net` wiki.

```
\definetypeface[Serapion][rm][Xserif][Serapion Pro]
\setupbodyfont[Serapion, 12pt]

\starttypescript[serif][didot][uc]
\definefontsynonym[DidotRegular]['Didot']  [encoding=uc]
\definefontsynonym[DidotItalic] ['Didot/I'][encoding=uc]
\definefontsynonym[DidotBold]   ['Didot/B'][encoding=uc]
\definefontsynonym[DidotCaps]   ['Didot;
   Letter Case=Small Capitals;Ligatures=!Common Ligatures']
   [encoding=uc]
\stoptypescript

\definedfont["Hoefler Text:mapping=tex-text;
              Style Options=Engraved Text;
              Letter Case=All Capitals;color=229966" at 24pt]
```

Including PDF images

The primitive `\XeTeXpdffile` *[options]* *"filename"* includes a page from a PDF file.

[option] keywords include:

- `page` *number* (default is first page in the PDF)
- `crop`, `media`, `bleed`, `trim`, `art` select bounding box to use
- `scaled`, `xscaled`, `yscaled` *factor* (1000 = natural size)
- `width`, `height` *dimen*
- `rotated` *degrees*

From TeX's point of view, the image is like a big character; put it into an `\hbox` to measure height and width with macros.

Including PDF images

The primitive `\XeTeXpdffile` *[options]* *"filename"* includes a page from a PDF file.

[option] keywords include:

- `page` *number* (default is first page in the PDF)
- `crop`, `media`, `bleed`, `trim`, `art` select bounding box to use
- `scaled`, `xscaled`, `yscaled` *factor* (1000 = natural size)
- `width`, `height` *dimen*
- `rotated` *degrees*

From TeX's point of view, the image is like a big character; put it into an `\hbox` to measure height and width with macros.

New in XeTeX 0.997: `\XeTeXpdfpagecount` *"filename"* accesses the number of pages in the given file; allows packages such as `pdfpages` to iterate over a complete file.

JPEG, PNG, and BMP images

For raster images, use `\XeTeXpicfile` *[options] "filename"*

Options are the same as for PDF files, except for the PDF-specific bounding box and page selection options.

Again, XeTeX knows the dimensions of the image, so by putting it in a box and examining `\ht` and `\wd`, macros can do appropriate layout.

The LaTeX graphics package automatically recognizes XeTeX and uses these commands to handle pictures.

JPEG, PNG, and BMP images

For raster images, use `\XeTeXpicfile` *[options] "filename"*

Options are the same as for PDF files, except for the PDF-specific bounding box and page selection options.

Again, XeTeX knows the dimensions of the image, so by putting it in a box and examining `\ht` and `\wd`, macros can do appropriate layout.

The LaTeX graphics package automatically recognizes XeTeX and uses these commands to handle pictures.

On Mac OS X, many additional image formats can be used with the `xdv2pdf` output driver, thanks to QuickTime, but for cross-platform compatibility JPEG and PNG are recommended.

Support for EPS

It is also possible to use EPS images, because the `xdvipdfmx` driver can automatically convert these to PDF using Ghostscript or another distiller (see the `D` option in `dvipdfmx.cfg`).

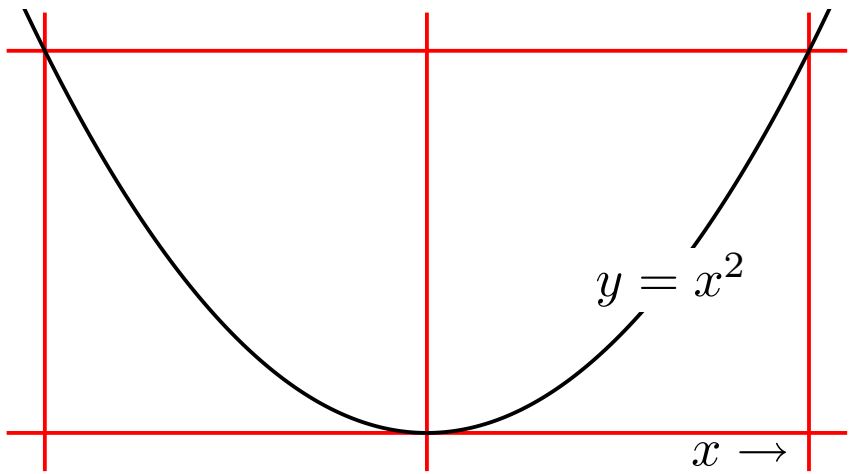
Currently, this works with the LaTeX graphics package and the newest `xetex.def` driver definition. Other packages may need to be updated to accept EPS, as XeTeX did not originally support this format.

Support for EPS

It is also possible to use EPS images, because the `xdvipdfmx` driver can automatically convert these to PDF using Ghostscript or another distiller (see the `D` option in `dvipdfmx.cfg`).

Currently, this works with the LaTeX graphics package and the newest `xetex.def` driver definition. Other packages may need to be updated to accept EPS, as XeTeX did not originally support this format.

```
\includegraphics
    [width=\hsize]
        {combined.eps}
```



TeX's math typesetting features

Essentially, a math formula is parsed into an *mlist* of various types of items: operators, variables, fractions, relations, etc. Then TeX converts this to nested hboxes and vboxes, using various font styles and positioning rules.

The basic operation of putting a character into a math formula is `\mathchar`, which takes a 16-bit value that is made up of a type (0–7), family (0-15), and character code (0–255). The family resolves to an appropriate font depending on `\textfont`, `\scriptfont`, and `\scriptscriptfont` settings.

For more complex elements, there are `\delimiter` and `\radical`, which take longer numbers with additional fields for small and large variants.

TeX's math typesetting features

Math typesetting also depends on special font metrics in the 'symbol' and 'extension' fonts that include lists of variable-sized and extensible operators, delimiters, etc., in addition to the metrics of standard text fonts.

To allow characters to be entered directly in math formulas, instead of always using `\mathchar` or macros that expand to this, each character also has a `\mathcode` that determines what type of character it is, which math family it belongs to, etc. Similarly, there is `\delcode` for characters that are to behave as delimiters.

Math processing is described in detail in Appendix G of *The TeXbook*.

Extending TeX for Unicode math

Unicode now provides standardized character codes for a huge number of math symbols, including styled letters (bold, fraktur, calligraphic, blackboard bold, etc.) that may be used for special purposes, as well as operators, relations, and so on.

The first step in allowing XeTeX to use Unicode for math typesetting is to extend the various math-related codes to support the full range of Unicode characters. To retain compatibility with old TeX macro files, this is done with new primitives `\XeTeXmathcode`, `\XeTeXmathchar`, `\XeTeXdelcode`, etc., that accept larger character codes.

At the same time, the number of permitted math font families has been increased from 16 to 256, to provide additional flexibility for macro package writers.

Extending TeX for Unicode math

```
\XeTeXmathcode`\-="2 "2 "2212 % minus sign
\XeTeXmathcode`\/="0 "1 "2215 % division slash
\XeTeXmathcode`\.="0 "1 `. % period
\XeTeXmathcode`\,="0 "1 `, % comma
\XeTeXmathcode`\∑="1 "2 `∑ % sum

\XeTeXmathchardef\sum="1 "2 `∑
\XeTeXmathchardef\prod="1 "2 `∏
\XeTeXmathchardef\intop="1 "2 `∫
\XeTeXmathchardef\ointop="1 "2 `∮
\XeTeXmathchardef\infty="1 "2 `∞

\def\lceil{\XeTeXdelimiter"4 "3 "2308 } % ceiling
\def\rceil{\XeTeXdelimiter"5 "3 "2309 }
\def\lfloor{\XeTeXdelimiter"4 "3 "230A } % floor
\def\rfloor{\XeTeXdelimiter"5 "3 "230B }

\def\sqrt{\XeTeXradical"3 "221A }
```

OpenType math fonts

With OpenType, it is possible for a single font to contain all the thousands of mathematical characters. In addition, a variety of metrics are needed to control the positioning and spacing of the parts of a formula.

Traditionally, TeX used a number of `\fontdimen` parameters from the `tfm` files for the math symbol and extension fonts. Equivalent information was simply not available with most other fonts.

With the release of Word 2007, Microsoft has defined a format for embedding such math-specific metrics in OpenType fonts. The first example of such a math-capable OpenType font is Cambria Math, included with a number of current MS products (including some freely-downloadable viewers).

OpenType math fonts

XeTeX is now able to use the OpenType 'MATH' table to control math typesetting, so that OpenType math fonts like Cambria can automatically be laid out correctly, with no additional metrics files.

$$f(x) = a_0 + \sum_{n=1}^{\infty} \left(a_n \cos \frac{n\pi x}{L} + b_n \sin \frac{n\pi x}{L} \right)$$

$$f(x) = a_0 + \sum_{n=1}^{\infty} \left(a_n \cos \frac{n\pi x}{L} + b_n \sin \frac{n\pi x}{L} \right)$$

Handling text in native fonts

When using native system fonts (OpenType, AAT, Graphite), the minimal unit of text from the TeX point of view is the *native word* (a new node type).

Such a 'native word' might not correspond exactly to a natural-language word; it is simply a run of adjacent characters using a single font.

From TeX's point of view, the word is like a box, with width, height, and depth, but its contents cannot be unboxed or otherwise inspected.

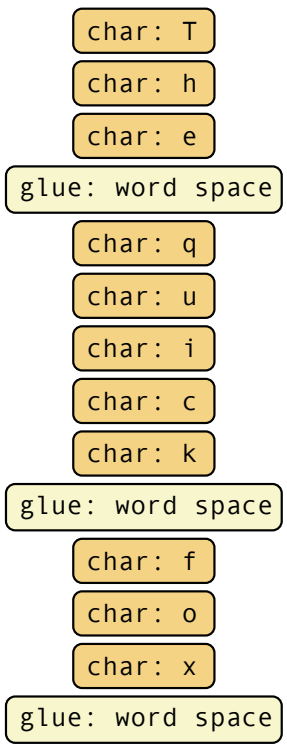
All layout within the word, like ligatures, kerning, or complex script behavior, is handled by a text layout engine (ATSUI, ICU Layout, or Graphite), which determines the actual glyphs to be rendered and their positioning.

Building a paragraph

TeX collects characters and spaces (glue) into a *horizontal list* which will be broken into lines.

Building a paragraph

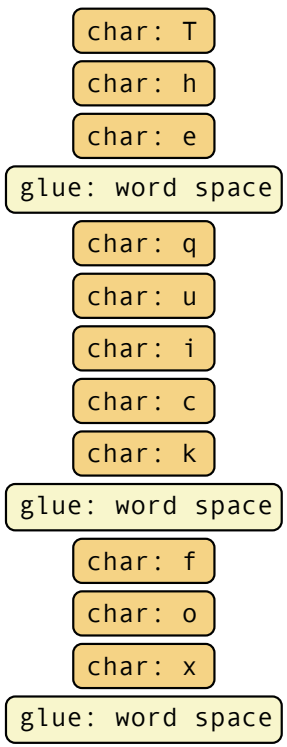
TeX collects characters and spaces (glue) into a *horizontal list* which will be broken into lines.



Building a paragraph

TeX collects characters and spaces (glue) into a *horizontal list* which will be broken into lines.

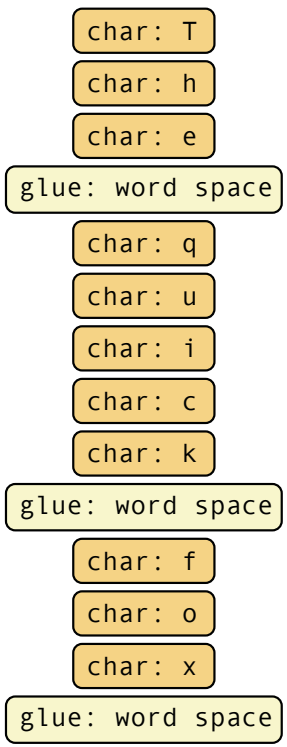
XeTeX gathers each run of characters in the same font into a *word*, so the horizontal list now consists of words and glue.

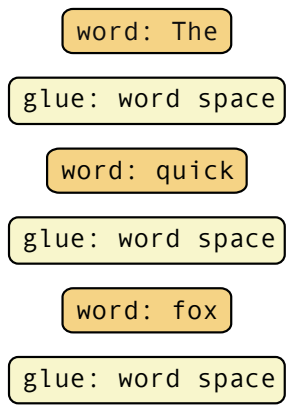


Building a paragraph

TeX collects characters and spaces (glue) into a *horizontal list* which will be broken into lines.

XeTeX gathers each run of characters in the same font into a *word*, so the horizontal list now consists of words and glue.





Hyphenation with system fonts

The horizontal list contains word nodes and glue:

`Two` `glue` `different` `glue` `foxes`

Hyphenation with system fonts

The horizontal list contains word nodes and glue:

`Two` `glue` `different` `glue` `foxes`

If hyphenation is needed, the word node has to be deconstructed, and discretionary nodes inserted:

`Two` `glue` `dif` `hyphen?` `fer` `hyphen?` `ent` `glue` `foxes`

Hyphenation with system fonts

The horizontal list contains word nodes and glue:

`Two` `glue` `different` `glue` `foxes`

If hyphenation is needed, the word node has to be deconstructed, and discretionary nodes inserted:

`Two` `glue` `dif` `hyphen?` `fer` `hyphen?` `ent` `glue` `foxes`

Now the line-break routine can use the discretionary breaks:

`Two` `glue` `dif` ` ` `fer` `-`

`ent` `glue` `foxes`

Two differ-
ent foxes

Hyphenation with system fonts

The horizontal list contains word nodes and glue:

`Two` `glue` `different` `glue` `foxes`

If hyphenation is needed, the word node has to be deconstructed, and discretionary nodes inserted:

`Two` `glue` `dif` `hyphen?` `fer` `hyphen?` `ent` `glue` `foxes`

Now the line-break routine can use the discretionary breaks:

`Two` `glue` `dif` `␣` `fer` `-`

`ent` `glue` `foxes`

Two differ-
ent foxes

To avoid disrupting the low-level text layout, we must reassemble word fragments wherever a discretionary break was not used:

`Two` `glue` `differ-`

`ent` `glue` `foxes`

Two differ-
ent foxes

Questions... and answers?

- Contact information
  - mailto:jonathan_kew@sil.org
- XeTeX web site and mailing list
  - http://scripts.sil.org/xetex
  - http://tug.org/mailman/listinfo/xetex
- Source repository for ongoing development
  - http://scripts.sil.org/svn-public/xetex/TRUNK

Questions... and answers?

- Contact information
  - mailto:jonathan_kew@sil.org
- XeTeX web site and mailing list
  - http://scripts.sil.org/xetex
  - http://tug.org/mailman/listinfo/xetex
- Source repository for ongoing development
  - http://scripts.sil.org/svn-public/xetex/TRUNK

ዩኒኮድ ምንድን ነው? ؟"يونِكود" ما هي الشفرة الموحدة 什麼是Unicode(統一碼/標準萬國碼)? Što je Unicode? რა არის უნიკოდი? Τί εἶναι τὸ Unicode; ?מה זה יוניקוד यूनिकोड क्या है? Hvað er Unicode? ユニコードとは何か? 유니코드에 대해? ؟چیست یونیکُد Что такое Unicode? Unicode คืออะไร? ዩኒኮድ ኣንታይ ኢዩ?