

X_ƎTeX Live

Jonathan Kew

SIL International

Horsleys Green

High Wycombe

Bucks HP14 3XL, England

jonathan_kew (at) sil dot org

1 X_ƎTeX in TeX Live

The release of TeX Live 2007 marked a milestone for the X_ƎTeX project, as the first major TeX distribution to include X_ƎTeX (version 0.996) as an integral part. Prior to this, X_ƎTeX was a tool that could be added to a TeX setup, but version and configuration differences meant that it was difficult to ensure smooth integration in all cases, and it was only available for users who specifically chose to seek it out and install it. (One exception to this is the MacTeX package, which has included X_ƎTeX for the past year or so, but this was just one distribution on one platform.) Integration in TeX Live, in contrast, provides near-universal availability and a more standardized configuration, which should simplify setup, use and support.

Special thanks to Karl Berry for his encouragement and support through this process, and to all the TeX Live builders and testers on various platforms who helped to make this possible.

1.1 Key features

The two most significant features of X_ƎTeX as found in TeX Live remain the same as they have been since its first appearance: support for the use of the host operating system's fonts (PostScript, TrueType, or OpenType) with no TeX-specific setup, and including layout features defined in the fonts; and extensive support for Unicode, including complex Asian and other scripts. With this release, users on all platforms have the option of using the same OpenType fonts in TeX documents as in mainstream GUI applications, including access to all the rich typographic features found in modern fonts.

As an example of the simplicity X_ƎTeX brings to font usage, consider the present article. This is written using the `lugproc` class. Running this in X_ƎLaTeX, the lines:

```
\usepackage{fontspec}
\setmainfont[Mapping=tex-text]
    {Adobe Garamond Pro}
\setmonofont[Scale=MatchLowercase]
    {Andale Mono WT J}
```

Note: This article is based on the author's presentations at both the EuroBachTeX 2007 and TUG 2007 conferences, but is printed in a single *Proceedings* issue to avoid duplication.

in the preamble are sufficient to set the typefaces throughout the document. These fonts were installed by simply dropping the `.otf` or `.ttf` files in the computer's Fonts folder; no `.tfm`, `.fd`, `.sty`, `.map`, or other TeX-related files had to be created or installed.

Release 0.996 of X_ƎTeX also provides some enhancements over earlier, pre-TeX Live versions. In particular, there are new primitives for low-level access to glyph information (useful during font development and testing); some preliminary support for the use of OpenType math fonts (such as the Cambria Math font shipped with MS Office 2007); and a variety of bug fixes.

1.2 Hyphenation setup

A long-standing problem with integrating X_ƎTeX has been the variety of hyphenation patterns for various languages, which are written using a variety of character encodings and various ways to represent those encodings in 7-bit or 8-bit files. Because X_ƎTeX interprets 8-bit text files as Unicode (UTF-8) by default, many old hyphenation files cannot be read as-is. This in turn meant that the X_ƎLaTeX format could fail to build, depending on the user's language configuration.

Older releases of X_ƎTeX made some attempt to address this by including modified versions of some of the hyphenation files from TeX, adapted to load correctly as Unicode patterns. However, ensuring that these were installed in the right place for X_ƎTeX to find them (without affecting other engines or replacing standard files) was problematic.

In TeX Live 2007, this situation has been addressed by modifying the `language.dat` file so that hyphenation files are loaded via "wrappers" (except for those that are simple ASCII files, which are already Unicode-compatible). The wrapper files, provided in TeX Live in `texmf-dist/tex/generic/xu-hyphen`, test whether the format is being built by X_ƎTeX, and if so they redefine the input encoding and/or `\catcodes`, active character definitions, etc., so that the patterns will be loaded as Unicode data. Figure 1 shows an example of such a wrapper file; in this case, the German vowels with umlauts and the ß character need Unicode-compliant definitions, in place of those found in the original hyphenation file. The precise details vary, of course, depending on the structure and encoding

```

% xu-dehyphn.tex
% Wrapper for XeTeX to read dehyphn.tex
% Jonathan Kew, 2006-08-17
% Public domain
\begingroup
\expandafter\ifx\csname XeTeXrevision\endcsname\relax
\else
  \catcode`\?=7
  % Define the accent macro " in such a way that it
  % expands to single letters in Unicode
  \catcode`\`=13
  \def"#1{\ifx#1a??e4\else \ifx#1o??f6\else \ifx#1u??fc\else
    \errmessage{Hyphenation pattern file corrupted!}%
    \fi\fi\fi}
  % - patterns with umlauts are ok
  \def\n#1{#1}
  % - define \3 to be character "00DF (\ss in Unicode)
  \def\3{??df}
  % - define \9 to throw an error
  \def\9{\errmessage{Hyphenation pattern file corrupted!}}
  % - duplicated patterns to support font encoding OT1 are not wanted
  \def\c#1{}
  %
  \let\PATTERNS=\patterns
  \def\patterns{% at the \patterns command in dehyphn.tex...
    \endgroup % end group containing local definitions from dehyphn
    \begingroup % and start our own (to match \endgroup in dehyphn)
    \PATTERNS % and then load the real patterns
  }
\fi
\input dehyphn.tex
\endgroup
\endinput

```

Figure 1: `xu-dehyphn.tex`, a typical hyphenation wrapper file from the TeX Live setup

of the pattern file being loaded, but similar techniques can generally be used.

In the longer term, reorganization and standardization of the hyphenation files, perhaps co-ordinated with work in OpenOffice.org (which uses a very similar hyphenation algorithm) would be a useful project. However, this will require not only a good understanding of the language and encoding issues, but also interaction with license holders or maintainers of all the existing patterns. Meanwhile, the current setup with `xu-` wrappers has proved to be a workable interim solution.

1.3 Package configuration

Another common problem for XeTeX users in the past has been that some popular L^ATeX packages (e.g., `graphics`, `color`, `geometry`, `crop`, `hyperref`, and others) depend on knowing the intended output driver (direct PDF generation with `pdfTeX`, `dvips`, `dvipdfm`, etc.) in order to use the correct implementation-specific methods to control the output. Many such packages attempt to detect the TeX engine in use and automatically choose the appropriate driver. However, with XeTeX being a new engine, existing packages were unaware of it.

This situation is improving, as some major packages have added a test for XeTeX and now choose the appropriate driver options. For others, including important cases like `geometry` and `crop`, TL2007 includes configuration files in the `xelatex` subtree that provide the proper setup. In most cases, therefore, users should find that the packages work transparently in XeTeX just as with other TeX engines and drivers.

One important package that did not work transparently with XeTeX in the TL2007 release is `pgf`; however, since the release in February, `pgf` has also been updated so that it recognizes the XeTeX engine automatically.

1.4 The ArabXeTeX package

A new package by François Charette provides an ArabTeX-like interface for typesetting languages in Arabic script, using standard Unicode-based fonts. As shown in figure 2, this supports both literal Unicode input of Arabic text, and ArabTeX transliterations, and can work with any OpenType font, including complex calligraphic styles such as Nastaliq script. This package was created after the current TeX Live release, but can be obtained from CTAN and works with the existing XeTeX version.

```
% preamble...
\usepackage{arabxetex}
\newfontfamily\arabicfont
  [Script=Arabic]{Scheherazade}
\newfontfamily\urdufont
  [Script=Arabic,Scale=0.75]{Noori Nastaliq MT}
% body...
\begin{arab}[fullvoc]
mina 'l-qur'Ani 'l-karImi,
  sUraTu 'l-ssajdaTi 15--16:
% ...etc...
\end{arab}
\begin{urdu}[voc]
یُونُسُ حَضْرَتِ وَعْجُوں بِنِیْ اِسْرَائِیْلِ اَبِیْءِ
% ...etc...
```

Result:

مِنَ الْقُرْآنِ الْكَرِيمِ، سُورَةُ السَّجْدَةِ ١٥-١٦:

إِنَّمَا يُؤْمِنُ بِآيَاتِنَا الَّذِينَ إِذَا ذُكِرُوا بِهَا حُزُّوا وَسَخُوا بِحَمْدِ رَبِّهِمْ وَهُمْ لَا يَسْتَكْبِرُونَ ۝
تَتَجَافَى جُنُوبُهُمْ عَنِ الْمَضَاجِعِ يَدْعُونَ رَبَّهُمْ خَوْفًا وَطَمَعًا وَمِمَّا رَزَقْنَاهُمْ يُنفِقُونَ ﴿١٥﴾

انجیاء بنی اسرائیل و یونس حضرت یونس کب علیہ السلام نبی ہیں۔ آپ دا دور نبوت ۸۱۷/۸۱۶ ق م کنوں ۵۳ ق م تک ہے۔ اوں ویلے مملکت بنی اسرائیل دا بادشاہ یربعام ثانی ہا۔ آپ لیں مملکت وچ ناصر تے ہال "جات حضر" دے زمانہ تے رہندے ہن۔ آج کل لیں جاہ کنوں "خریبا الزورع" آیدن۔ ایہ مقام جمیل جمیل دے مغربی پاسے پازحان نیل اتیں ناصر دے شمال مشرقی پاسے ترانے نیل دے فاصلے تے ہے۔ انھوں دے کنڈرات دی ٹھڈائی دے دوران پتھلے جو ایہ مقام کانسی دے آخری دور تقریباً ۱۵۵۰ ق م کنوں ۱۴۰۰ ق م تک آہد ہا۔ ولا پوجھی دفعہ ایہ تقریباً ۱۴۰۰ ق م کنوں ۶۰۰ ق م تک آہد رہندا۔ حضرت یونس دے دور تقریباً ۱۴۰۰ ق م کنوں ۶۰۰ ق م تک آہد ہا۔ روایت دے مطابق حضرت یونس دا مزار اٹھائیں ہے۔

Figure 2: Examples of ArabX_YTeX input and typeset output

2 Beyond TL2007 and X_YTeX 0.996

In parallel with the integration of X_YTeX 0.996 into T_EX Live, there has been continuing development of the next version of X_YTeX itself and the associated drivers and support files. Release 0.997 (preliminary code is in the Subversion source repository at the time of writing) will include several new and enhanced features, a few of which are described here.

2.1 PSTricks graphics

One of the limitations of X_YTeX has been that it natively generates .xdv or "extended DVI" output, which needs to be converted to PDF by a special X_YTeX-specific output driver. This excludes the use of the dvips+Ghostscript output path, and therefore also prevents the use of packages that rely on writing PostScript `\special` commands that Ghostscript or a PostScript printer will interpret.

The most important such package, judging by discussion on the mailing lists, is probably PSTricks, which is widely used for special drawing and graphic effects. Thanks to recent work by Miyata Shigeru, the `xdvipdfmx` driver used with X_YTeX has been extended to support most PSTricks features (with a few exceptions), and therefore standard PSTricks pictures, plots, etc., can be used in X_YTeX. This is achieved by extracting the PostScript code and running Ghostscript (or another process, according to

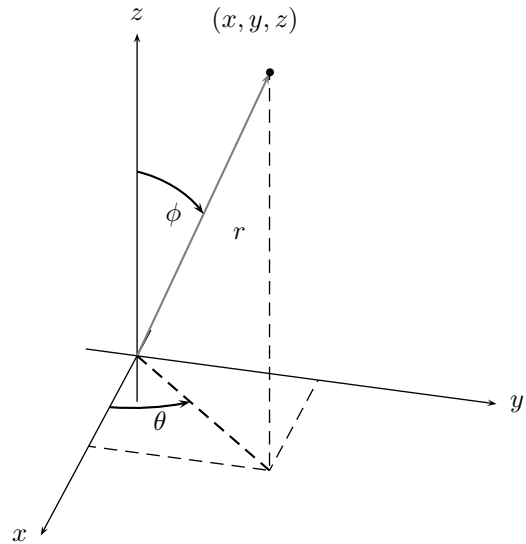


Figure 3: Example of a PSTricks plot embedded in a X_YTeX document (from <http://tug.org/PSTricks/main.cgi?file=pst-plot/3D/examples>)

the driver's .cfg file) to convert this to PDF which can then be embedded in the document, as illustrated in figure 3. While this technique is currently quite slow, it does at least permit the use of such graphics. However, users may find that other graphics packages such as the PGF-based T_ik_Z provide better performance in many cases.

2.2 Unicode math extensions

New in X_YTeX 0.996, and more complete in 0.997, is support for use of the full range of Unicode math characters, including the styled math alphabets in Plane 1 as well as the large number of mathematical symbols. T_EX's `\mathcode`, `\delcode` and related tables have been enlarged, and the number of math families is increased from 16 to 256. A small example of the use of Unicode characters in math mode is shown in figure 4; work is in progress to design and implement a L^AT_EX package to provide extensive and well-integrated support, building on the primitive facilities now available in the engine.

T_EX's math codes contain three distinct components, representing the character class (ordinary character, large operator, binary operator, relation, etc.), the math family to be used, and the actual character code. T_EX compresses this information into a single 16-bit value, with 3 bits for the class, 4 for the family, and 8 for the character code, normally expressed as 4 hex digits (see *The T_EXbook*, p. 154). X_YTeX packs a 3-bit class, 8-bit family, and 21-bit Unicode value into a single 32-bit code, but as the example in figure 4 shows, it allows the components to be specified separately for clarity as they no longer map neatly onto individual hex digits.

```

% set up Cambria Math for roman, symbol and extension families
\font\1="Cambria Math:script=math" at 10pt
\font\2="Cambria Math:script=math;+ssty=0" at 7pt
\font\3="Cambria Math:script=math;+ssty=1" at 5pt
\textfont0=\1 \scriptfont0=\2 \scriptscriptfont0=\3
\textfont2=\1 \scriptfont2=\2 \scriptscriptfont2=\3
\textfont3=\1 \scriptfont3=\2 \scriptscriptfont3=\3

% use Cambria Math with italic mapping for family 1
\font\1="Cambria Math:script=math;mapping=math-italic" at 10pt
\font\2="Cambria Math:script=math;mapping=math-italic;+ssty=0" at 7pt
\font\3="Cambria Math:script=math;mapping=math-italic;+ssty=1" at 5pt
\textfont1=\1 \scriptfont1=\2 \scriptscriptfont1=\3

% set mathcodes (many are predefined in xetex.fmt)
\XeTeXmathcode\`-="2 "2 "2212 % minus sign
\XeTeXmathcode\`Σ="1 "2 `Σ % summation

% some control sequences...
\XeTeXmathchardef\sum="1 "2 `Σ \XeTeXmathchardef\prod="1 "2 `Π
\XeTeXmathchardef\intop="1 "2 `j \XeTeXmathchardef\infty="1 "2 `∞
\XeTeXmathchardef\geq="3 "2 `≥ \XeTeXmathchardef\leq="3 "2 `≤
\XeTeXmathchardef\pi="7 "1 `π

% using Unicode characters in math
$$ f(x) = a_0 + \sum^{\infty}_{n=1} \left( a_n \cos\{\frac{n\pi x}{L}\} + b_n \sin\{\frac{n\pi x}{L}\} \right) $$

```

Result, using an OpenType math font:

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

Figure 4: Defining and using Unicode math characters

When using a complete OpenType math font such as Cambria Math, it may be necessary to load the font several times with different character mappings and OpenType features.

2.3 Inter-character token insertion

A new feature in X_ƒTEX version 0.997 is the ability to insert arbitrary token lists in between normal text characters, without complex macro programming. This is designed primarily to support requirements of Japanese and Chinese typography, where special spacing controls are needed in certain cases such as between ideographs and adjacent punctuation characters.

To support this feature, each character has a “class” known as `\XeTeXcharclass`, a bit like an extra `\catcode`, but ignored by normal TEX operations. But whenever two printable text characters occur next to each other, X_ƒTEX will check their class values, and if a token list has been defined for this class pair it will be inserted between the characters. Such a token list may contain arbitrary TEX material, although the most useful possibilities are probably various forms of `\skip` and `\penalty` (to control spacing and breaking), and font changes (making it possible to

automatically switch fonts for different scripts within Unicode text, without requiring embedded markup).

For example, the default `xetex` and `xelatex` formats initialize most `\XeTeXcharclass` values to zero, but assign all the CJK ideographs to class 1. We can take advantage of this to allow Chinese characters to be included in running text without additional markup, even though the default body font does not support them; a simple example is shown in figure 5. While this technique is not a universal substitute for proper language and font markup in the source document, it can greatly simplify the author’s task in some mixed-script situations.

2.4 Graphite font support

The initial version of X_ƒTEX, on Mac OS X only, supported special font features such as contextual swashes, ligatures, alternate glyphs, etc., by means of Apple’s AAT font technology. Later, support for OpenType font features was added, based on the ICU layout library; this enabled X_ƒTEX to provide complex font support across multiple platforms.

A third font layout technology, designed to support the requirements of non-Latin scripts, minority languages, and scripts not yet in Unicode, is SIL’s Graphite system (<http://scripts.sil.org/RenderingGraphite>).

