Typesetting multilingual text with `lualatex` and `xelatex`

Nelson H. F. Beebe

Department of Mathematics
University of Utah
Salt Lake City, UT 84112, USA

08 January 2020

Abstract

Producing multilingual verbatim displays in `lualatex` and `xelatex` from Unicode UTF-8 input should be straightforward, because both of those extensions of TEX and LATEX are native Unicode engines. Nevertheless, there remains a technological limitation of inadequate support for most non-Latin scripts in typewriter fonts. Multilingual prose is easier, but may still be limited by incomplete character coverage in Unicode fonts.

Section 16 of this document discusses typesetting of bidirectional texts in two languages.

This document is typeset by `lualatex` with a main font *Noto Serif*, and a typewriter font *Source Code Pro*. However, some sections illustrate completely different font choices.

Contents

Acknowledgments

This document has benefited from helpful exchanges with several friends in the TEX User Group and elsewhere. In alphabetical order by family name, they are: Barbara Beeton (USA), Karl Berry (USA), Efthymios (Tim) Folias (USA), Paul Hardy (USA), Richard Koch (USA), Frank Mittelback (Germany), Ross Moore (Australia), and Arnold (Aharon) Robbins (Israel). The author bears full responsibility for any mistakes in this document.

1 Introduction

TEX's final revision[1] in 1989 moved its character model from 7 bits to 8 bits, and its fonts remained with a limit of 256 characters. While that is enough for a single European language, it is inadequate for handling multilingual text without excessive font switching.

The LuaTEX[2] and XETEX[3] programs are significant extensions of TEX that completely replace its character and font mechanisms, basing them on the Unicode character set. When preloaded with the LATEX macro package, the executables for those extensions are called `lualatex` and `xelatex`.

In standard TEX, input characters are expected to match one-to-one with glyphs in a font. When such a match is impossible, they must be handled by remapping a command name, such as `\euro`, to a glyph (here, €) in a specified font, or by using TEX's category codes (*catcodes*) that allow selected characters to be designated *active*, which effectively means that they run TEX commands, rather than standing for themselves.

TEX fonts are accompanied by TEX font metric files (`*.tfm`) that contain additional information about the glyphs, including their dimensions, normal intercharacter spacing, special spacing between selected pairs of characters (called *kerning*), and ligatures, which are glyphs used to represent certain sequences of characters, such as *fi*, *ffi*, *fl*, and *ffl*, as well as the two- and three-hyphen input representation of en-dash (–) and em-dash (—). If the ligatures are broken, TEX produces instead *fi*, *ffi*, *fl*, *ffl*, -- and ---.

With traditional TEX typesetting, ligature breaking is normally done with input bracing: `{shelf}ful`, `shel{f}ful`, or `shelf{}ful`. *The TEXbook* suggests that an italic correction may look better in some fonts: `shelf\/ful`.

With Unicode fonts, however, TEX does not see individual characters in words, so the breaking must be done by boxing, or adding zero-width kerns: `\mbox{shelf}ful`, `shelf\null ful`, or `shelf{\kern 0pt}ful`. Here, `\null` is a standard TEX macro that is defined as an empty box: `\hbox{}`.

2 Platform (in)dependence of typesetting

TEX's typesetting was carefully designed to be system independent: all internal calculations that relate to line breaking and page breaking are done in exact integer arithmetic. Further, because TEX's font and font metric files are identical across all operating systems and CPUs, its typesetting is *platform independent*, so documents are *archival*: they look the same no matter where or when they were produced. Of course, macro definitions sometimes evolve, so typesetting files with higher-level markup, such as LATEX and ConTEXt, might then produce visible differences over time.

[1] See `https://tug.org/TUGboat/tb10-3/tb25knut.pdf` and `https://tug.org/TUGboat/tb11-4/tb30knut.pdf`.
[2] See `https://tug.org/TUGboat/tb28-3/tb90hoekwater-luatex.pdf`.
[3] See `https://tug.org/TUGboat/tb26-2/kew.pdf`.

With Unicode, however, typesetting is no longer under complete control by TEX. Instead, LuaTEX and XETEX assemble a word to be typeset, then hand it off to a *platform-dependent* Unicode library layer that uses information in a Unicode font, which includes dimension, kerning, and ligature information, but in addition may include glyph variants, as well as the ability to stretch or shrink letters to fit a particular horizontal size, as is commonly done in Arabic. Unicode fonts may also support alternate writing directions, such as right-to-left for Arabic and Hebrew, boustrophedon (back-and-forth) for Ancient Greek, vertical right-to-left for classical Chinese, Japanese and Korean, and vertical left-to-right for Mongolian, as well as color (notably, for *emojis*, like smiley and frowny faces).

There are currently four main Unicode libraries, from Microsoft in the Windows operating system, from Apple in macOS, IBM's `libicu` (International Components for Unicode) in various Unix family systems, and HarfBuzz that is available on all of those. When TEX Live 2020 is officially released, there should be `luahbtex` and `luahblatex` executables for TEX and LATEX typesetting with the HarfBuzz library.[4]

With each of the Unicode libraries, slightly different results might be expected for word typesetting, and thus, line breaking and page breaking almost certainly differ with each. In addition, they are evolving software products over which TEX has no control, so even on a single platform, software upgrades of those libraries, or of Unicode font files, potentially change how a particular document is typeset. That means that reproducing the typeset appearance of documents may be impossible, except within relatively short time frames when software and font updates are avoided.

When document production stability is needed, as it is for many large organizations and publishers, the best solution is likely to carry out the typesetting with a *virtual machine* whose software and fonts are frozen, and for security, made a standalone network-free system, or else isolated in a private network that is invisible to the outside world. The lightweight machines called *containers* in Linux, *jails* in FreeBSD, and *zones* in the Solaris family do not provide adequate isolation, because they share software components with the underlying operating system.

3 Unicode and fonts

The Unicode character set requires at most 21 bits for each character number, which would permit up to $2^{21} = 2\,097\,152$ characters. However, restrictions needed for compressed encodings reduce that limit to 1 111 998. That is more than enough to support all of the world's known writing systems: the Unicode 12.1 Standard of May 2019 defines 'only' 137 994 characters.

The Unicode encoding includes three *Private Use Areas*, character ranges U+E000–U+F8FF (6 400), U+F0000–U+FFFFD (65 534), and U+100000–U+10FFFD (65 534), giving a total of 137 468 slots that will never be assigned glyphs, or character properties. Characters in those areas only have meaning when producer and consumer parties agree on them.

[4] See `https://tug.org/TUGboat/tb40-1/tb124hosny-harfbuzz.pdf`.

There are more than 20,000 commercially available fonts, yet only a few of them have been extended to support Unicode, and they rarely cover the large character repertoires needed for Chinese, Japanese, Korean, and the many alphabetic scripts of Africa, the Indian subcontinent, some indigenous languages in North America, and Utah's own Deseret alphabet. The offerings in free fonts are much more limited.

Among free serif and sans serif fonts in early 2020, it appears that one of the largest glyph collections may be in the Google *Noto* family, and for typewriter fonts, in Adobe *Source Code Pro*. Other large families include James Kass' *Code2000*, *Code2001*, and *Code2002* fonts, and the GNU *FreeFont* collection.[5] The Kass fonts were originally distributed as shareware, with a modest fee requested for use. Some Web documents report that they are now covered by a free license, but I have not yet been able to clarify their current license status.

There is an even larger glyph collection in GNU *Unifont*,[6] developed by Roman Czyborra, Paul Hardy, and others, but it was created by converting 16 × 16 bitmap fonts from many free sources into a Unicode font that is suitable for a fallback for screen display of otherwise-missing glyphs, but not for high-quality typesetting. We illustrate *Unifont* in Section 11.

With the Unicode typesetting engines derived from TEX, families of fonts are chosen by using their designer names, or their file names, and they generally automatically include stylistic variants, such as **bold**, *italic*, sans serif, *slanted*, `monospaced or typewriter`, and so on. Thus, the preamble of this document selected the fonts with just three simple commands:

```
\setmainfont {Noto Serif}
\setmonofont {Source Code Pro} [Scale = MatchLowercase]
\setsansfont {Noto Sans}       [Scale = MatchLowercase]
```

The scaling of the second two improves their appearance when used near the prose font.

4 Editor and terminal support for Unicode

To make it possible to work with the large glyph repertoire of Unicode, you need a suitable text editor, and a terminal program that can display and input Unicode characters. For this document, I used the standard `xterm` program, then typed Ctl-Right-Mouse to get a popup window from which I selected *TrueType Fonts*, *Unicode Encoding*, and *Unicode Fonts*. For editing the document, I used `emacs`.

I found the multilingual text samples on the Web, and simply cut-and-pasted them from a Web browser into the document file. While `emacs` shows the Asian characters in the samples, `xterm` does not. Switching to `gnome-terminal`, `mate-terminal`, `uxterm`, or `xfce4-terminal` allows screen display of the Asian characters.

[5] See `https://www.gnu.org/software/freefont`.
[6] See `https://savannah.gnu.org/projects/unifont/`.

emacs has powerful input methods that allow entry of text in many different scripts, but I did not need to use them for this document.

5 Finding, inspecting, and viewing Unicode fonts

The primary formats for Unicode fonts are TrueType (*.ttc and *.ttf) files and OpenType (*.otf) files. The .ttc extension is used for TrueType Collections, which are files that can contain multiple TrueType fonts.

Fonts can be found by the TEX engines extended for Unicode using the normal environment variables for file search paths:

```
% kpsewhich --help-formats
...
opentype fonts: .otf .OTF  [variables: OPENTYPEFONTS
  TEXFONTS] [original path (from texmf.cnf) =
  $TEXMFDOTDIR:$TEXMF/fonts/{opentype,truetype}\
  //:$OSFONTDIR//]
...
truetype fonts: .ttf .ttc .TTF .TTC .dfont  [variables:
  TTFONTS TEXFONTS] [original path (from texmf.cnf) =
  $TEXMFDOTDIR:$TEXMF/fonts/{truetype,opentype}\
  //:$OSFONTDIR//]
...
```

For typesetting this document, I set just one variable:

```
# For csh / fish / tcsh:
% setenv TEXFONTS .:/usr/share/fonts//:

# For ash / bash / dash / ksh / mksh / pdksh / sh / zsh:
$ TEXFONTS=.:/usr/share/fonts//:
$ export TEXFONTS
```

The doubled slash ending a pathname means to search its subdirectories, and the final colon in the value means that the system default value is implicitly appended. That way, both system fonts, and fonts in the TEX Live tree, can be found.

Fonts can also be found by the underlying Unicode library layers, not controllable through user-definable environment variables, but rather by system configuration files whose absolute pathnames are compiled into the libraries.

Based on experiments in a large test farm with hundreds of different operating systems, it appears that modern Unix-family systems use the fontconfig package that supplies a configuration file, fonts.conf, found in various locations, such as /etc/fonts

(Linux, OpenBSD, and Solaris family), `/usr/local/etc/fonts` (FreeBSD family), and `/usr/pkg/etc/fontconfig` (NetBSD). That file contains information about directory trees where fonts are stored, which you can display like this:

```
% grep "<dir>" /etc/fonts/fonts.conf
<dir>/usr/share/fonts</dir>
<dir>/usr/share/X11/fonts/Type1</dir>
<dir>/usr/share/X11/fonts/TTF</dir>
<dir>/usr/local/share/fonts</dir>
<dir>~/.fonts</dir>
```

The fontconfig package also supplies several executables:

`fc-cache`	build font information cache files
`fc-cat`	read font information cache files
`fc-conflist`	show the ruleset files information on the system
`fc-list`	list available fonts
`fc-match`	match available fonts
`fc-pattern`	parse and show pattern
`fc-query`	query font files
`fc-scan`	scan font files or directories
`fc-validate`	validate font files

Unfortunately, their manual-page descriptions are unreasonably brief, so you are likely to need online Web pages[7] for more documentation.

For fonts that are *native* to the platform, nothing special needs to be done. Users of word processors, Web browsers, and editors just expect to be able to select font names from menus that are constructed on-the-fly by enumerating all of the fonts found in system catalogs. To make system fonts available for use in TEX engines, environment variables are still needed, as we did with `TEXFONTS`.

To add more font directories to your system, there are two choices: either add them to the standard font directory, such as `/usr/share/fonts`, or add their directory names to a file `local.conf` in the same directory as `fonts.conf`. Later system updates might change the latter file, but would preserve your local additions. Then run the `fc-cache` program to update the cache that makes font access faster. Those actions all require `root` (administrator) access.

At my site, we added a directory `/usr/share/fonts/local` that is a symbolic link to an NFS-mounted directory on our main fileserver, allowing our hundreds of systems to access the locally added fonts without needing private copies of them. That choice also means that major system upgrades that might entirely replace the `/usr/share/fonts` directory do not destroy our local font archive. We just need to restore the symbolic links for the `local` subdirectory.

[7] See `https://www.freedesktop.org/wiki/Software/fontconfig/`.

Two important tools, `ttfdump` and `otfinfo`, can be used to extract information from TrueType and OpenType fonts, and their output can be voluminous. Here is how I counted the number of characters and glyphs in some fonts used in this document:

```
% ttfdump /usr/share/fonts/local/unifont/          \
         unifont-12.1.04/unifont-12.1.04.ttf |    \
         grep -c 'Char '
57087

% ttfdump /usr/share/fonts/google-noto/            \
         NotoSerif-Regular.ttf | grep -c 'Char '
2189

% ttfdump /usr/share/fonts/google-noto/            \
         NotoSerif-Regular.ttf | grep -c '^Glyph '
2414

% cd /usr/share/fonts/adobe-source-code-pro
% foreach f ( *.otf )
     printf "%-31s\t" $f ; otfinfo -g $f | wc -l
  end
SourceCodePro-Black.otf         1585
SourceCodePro-BlackIt.otf       1288
SourceCodePro-Bold.otf          1585
SourceCodePro-BoldIt.otf        1288
SourceCodePro-ExtraLight.otf    1585
SourceCodePro-ExtraLightIt.otf  1288
SourceCodePro-It.otf            1288
SourceCodePro-Light.otf         1585
SourceCodePro-LightIt.otf       1288
SourceCodePro-Medium.otf        1585
SourceCodePro-MediumIt.otf      1288
SourceCodePro-Regular.otf       1585
SourceCodePro-Semibold.otf      1585
SourceCodePro-SemiboldIt.otf    1288
```

With a similar loop over all of the OpenType fonts in the `/usr/share/fonts` tree, I made a list of fonts sorted by increasing glyph counts:

```
$ for f in `find /usr/share/font* -name '*.otf' `
  do
      printf "%-63s\t" $f ; otfinfo -g $f | wc -l
  done | sort -k2,2n -k1,1
/usr/share/fonts/mathjax/MathJax_Vector-Bold.otf            2
/usr/share/fonts/mathjax/MathJax_Vector-Regular.otf         2
/usr/share/fonts/mathjax/STIXMathJax_Shapes-BoldItalic.otf  7
```

```
...
/usr/share/fonts/source-sans-pro/SourceSansPro-Semibold.otf 1942
/usr/share/fonts/stix/STIX-Regular.otf                      3780
/usr/share/fonts/stix/STIXMath-Regular.otf                  4226
```

That list *excludes* fonts with Chinese, Japanese, and Korean characters, because with the `-g` option, the `otfinfo` tool complained `CID-keyed fonts not supported`.

You can ask for general font information with

```
% otfinfo -i /usr/share/fonts/adobe-source-han-sans-cn/          \
            SourceHanSansCN-Regular.otf
Family:               Source Han Sans CN Regular
Subfamily:            Regular
Full name:            Source Han Sans CN Regular
PostScript name:      SourceHanSansCN-Regular
Preferred family:     Source Han Sans CN
Version:              Version 1.000;PS 1;hotconv 1.0.78;          \
                      makeotf.lib2.5.61930
Unique ID:            1.000;ADBE;SourceHanSansCN-Regular;ADOBE
Description:          Dr. Ken Lunde (project architect,           \
                      glyph set definition & overall production); \
                      Masataka HATTORI □□□□                       \
                      (production & ideograph elements)
Designer:             Ryoko NISHIZUKA □□□□ (kana & ideographs);   \
                      Paul D. Hunt (Latin, Greek & Cyrillic);     \
                      Wenlong ZHANG □□□ (bopomofo);               \
                      Sandoll Communication □□□□□□□□,              \
                      Soo-young JANG □□□ & Joo-yeon KANG □□□      \
                      (hangul elements, letters & syllables)
Manufacturer:         Adobe Systems Incorporated
Vendor URL:           http://www.adobe.com/type/
...
```

and get an idea of the script coverage with

```
% otfinfo -s /usr/share/fonts/adobe-source-han-sans-cn/    \
            SourceHanSansCN-Regular.otf
DFLT      Default
cyrl      Cyrillic
grek      Greek
hani      CJK Ideographic
hani.ZHS  CJK Ideographic/Chinese Simplified
kana      Hiragana/Katakana
latn      Latin
```

The `ttx` program from the fonttools package can display information from TrueType and OpenType files, as well as export the entire font to XML where you can view, and even change, its details, and import the XML back into a new font file.

To find what system fonts are available, run the `fc-list` command, saving the output in a file, or piping it into a pager or a search filter:

```
% fc-list | grep -i devanagari
/usr/share/fonts/google-noto/NotoSansDevanagari-Bold.ttf: \
    Noto Sans Devanagari:style=Bold
/usr/share/fonts/google-noto/NotoSansDevanagari-Bold.ttf: \
    Noto Sans Devanagari:style=Bold
/usr/share/fonts/google-noto/NotoSansDevanagariUI-Regular.ttf: \
    Noto Sans Devanagari UI:style=Regular
...
```

You can ask your TEX implementation where it looks for OpenType and TrueType fonts like this:

```
% kpsewhich --show-path=.otf
long search path output

% kpsewhich --show-path=.ttc
long search path output

% kpsewhich --show-path=.ttf
long search path output
```

You can count how many OpenType font files are available in a TEX Live tree like this:

```
% fgrep -c .otf $prefix/texlive/2020/ls-R
1452
```

That example assumes that `$prefix` expands to the root of the directory tree that holds locally installed software.

If you do not know where that tree is found on your system, just ask where the `tex` executable is found, like this:

```
% \which -a tex
/usr/uumath/texlive/2020/bin/x86_64-linux/tex
/usr/uumath/bin/tex
/usr/bin/tex
/bin/tex

% file /bin /usr/bin
/bin:     symbolic link to usr/bin
/usr/bin: directory
```

The initial backslash on the command tells the Unix shell to ignore any built-in `which` command, and instead look for a shell function, or an executable program. The `-a`

option means that `which` should report *all* instances of the program in the `PATH` list. The output shown from my site finds four `tex` programs, but the last two are identical because of the symbolic link for `bin`. Thus, I find that `prefix` is `/usr/uumath`.[8]

To find the location of a particular font file, you need two commands:

```
% grep -i 'freemono.*otf' $prefix/texlive/2020/ls-R
FreeMonoBoldOblique.otf
FreeMonoOblique.otf
FreeMonoBold.otf
FreeMono.otf

% kpsewhich FreeMono.otf
/usr/local/texlive/2020/texmf-dist/fonts/opentype/  \
public/gnu-freefont/FreeMono.otf
```

If the `gnome-font-viewer` program is installed on your system, you can run it to display a compact summary of available fonts. Use the mouse to select a particular font to get a new panel that shows sample text in various sizes.

Finally, you can view, and even edit (subject, of course, to file permissions and copyrights), the contents of font files like this:

```
% fontforge unifont-12.1.04.ttf
creates separate windows with a graphical user interface
```

5.1 Apple macOS systems

On Apple macOS, the `Font Book` program provides a view of fonts that are installed in `/System/Library/Fonts` and `/Library/Fonts`. The first contains the base system fonts, and the second contains fonts added later.

`Font Book` also supports installing fonts: for an unprivileged user, that just copies files from a specified directory into the user directory `$HOME/Library/Fonts`, where they are then normally available only for that user. If both `.ttf` and `.otf` files for a single font are installed in a font directory, you get a warning, with an offer to resolve duplicates. The TrueType format is the default, even though the newer OpenType format offers more features.

In a `Font Book` window, there are four icons in the top left toolbar: (i) letters and digits in the current font selection, (ii) a table of glyphs, (iii) sample text (*Lorem ipsum dolor ...*), and (iv) information about the font. For the first three of those, a *Size* icon

[8] Although the GNU default value of `prefix` is `/usr/local`, we long ago abandoned that default, because we support multiple operating systems, and some of them, notably from the BSD family, usurp the `/usr/local` tree for vendor package installations.

in the upper right of the window allows choice of the glyph size. Menus in the screen **Font Book** toolbar provide additional options, and show keyboard shortcuts. The menu under **Font Book** has a *Preferences* item that allows you to change the default location for font installation.

macOS automatically maintains several caches of font information that speed access to font files. If cache corruption is suspected, after terminating all running user programs, an administrator command, `atsutil databases -remove`, can be used to delete the font caches, after which a system restart is required. The caches are rebuilt during startup. Other solutions to the problem can be found on the Web.[9]

5.2 Microsoft Windows 10 systems

On Microsoft Windows, system fonts are stored in the directory `c:\Windows\Fonts`, and an automatically maintained cache file `StaticCache.dat` there speeds access to fonts. That is not the only location where fonts are found: numerous directories under `c:\Windows` also contain fonts installed by various software packages.

If you have sufficient filesystem space, consider installing the complete font offerings available for the operating system. From the *Settings* popup window, choose *Fonts*, and then *Download fonts for all languages*. At the time of writing this, there are then about 750 font files, requiring about 620MB of storage.

There appears to be no simple user-accessible tool to update the font cache file, and online documentation suggests that recovering from a corrupted cache is a perilous operation. If you think you might need to do so, search the vendor site for advice.[10]

Microsoft maintains an extensive Web site about typography[11] that may be helpful in understanding font technologies and locating fonts. In particular, that site has a link to a 2760-page manual with information about font licensing, lists of system fonts supplied with various versions of Windows, and samples of the vendor font products.

5.3 TrueType font collections

Font files whose names end with `.ttc` are *collections* of fonts. Surprisingly, neither `ttfdump` nor `ttx` appears to offer a way to list their names. However, if you run `fontforge` on such a file, it puts up a box asking you to choose the font to view. For example, for the file `PTSerif.ttc`, the box contains

[9] See `https://bit.ly/2R1cfgu` or `https://tinyurl.com/yguy3a4d`, and `https://bit.ly/2MZx3DN` or `https://tinyurl.com/yhq7oslz`.
[10] See, for example, `https://bit.ly/39PPgx8` or `https://tinyurl.com/yjgofan2`.
[11] See `https://docs.microsoft.com/en-us/typography/`.

```
There are multiple fonts in this file, pick one
PT Serif
PT Serif Italic
PT Serif Bold Italic
PT Serif Bold
```

The fonts are numbered from zero, so you could use the third one with commands like this:

```
% usually in preamble:
\defaultfontfeatures { Extension = .ttc }
...
% at point of font switch:
\setmainfont {PT Serif} [ FontIndex = 2 ]
```

Providing a default file extension is convenient if you need to change fonts often in a single document.

Alternatively, you can select all of them like this:

```
\setmainfont {PTSerif.ttc}
             [
                 % Path = /home/jones/Library/Fonts/,
                 UprightFeatures    = { FontIndex = 0 },
                 ItalicFeatures     = { FontIndex = 1 },
                 BoldItalicFeatures = { FontIndex = 2 },
                 BoldFeatures       = { FontIndex = 3 }
             ]
```

It is never a good idea to embed absolute, or system-dependent, pathnames into documents, but the commented `Path` setting shows how you could do so if the font file is not otherwise found.

The four font variants would then be automatically selected with the usual LaTeX `\rmfamily`, `\itshape`, `\itshape\bfseries`, and `\bfseries` declarations, or with the commands `\textrm{...}`, `\textit{...}`, `\textit{\textbf{...}}`, and `\textbf{...}`.

6 Prose font samples

In order to access glyphs with Chinese, Japanese, and Korean characters displayed in this section, I had to change the main font on the GNU/Linux system where this document was typeset to a related font that includes those characters, but also uses a sans serif style for Latin letters:

```
\setmainfont {NotoSansCJKsc}
```

This is \textrm{...} plain text. The quick brown fox jumped over the lazy dog. THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.

This block in a quotation environment contains the \setmainfont change:

> English: Hello, good day!
> Greek: Χα□ρετε, καλ□ μ□ρα
> Hindi: □□□□□□
> Icelandic: Halló, góður dagur!
> Chinese: 你好, 早晨
> Japanese: こんにちは
> Korean: 안녕하세요
> Russian: Здравствуйте, добрый день!

The X-boxes in that block indicate missing glyphs in the font, namely, the Devanagari letters needed for Hindi, and some of the accented Greek letters.

7 Monospaced font samples

This block in a quotation environment contains the \setmainfont change with a \ttfamily declaration:

> English: Hello, good day!
> Greek: Χαίρετε, καλή μέρα
> Hindi: □□□□□□
> Icelandic: Halló, góður dagur!
> Chinese: □□, □□
> Japanese: □□□□□
> Korean: □□□□□
> Russian: Здравствуйте, добрый день!

This block in a quotation environment contains the \setmainfont change, as well as a \bfseries command:

> **bold English: Hello, good day!**
> **bold Greek: Χα□ρετε, καλ□ μ□ρα**
> **bold Hindi: □□□□□□**
> **bold Icelandic: Halló, góður dagur!**
> **bold Chinese: 你好, 早晨**
> **bold Japanese: こんにちは**
> **bold Korean: 안녕하세요**
> **bold Russian: Здравствуйте, добрый день!**

This is a verbatim environment inside a quotation environment, preceded by the \setmainfont change:

```
English:   Hello, good day!
Greek:     Χαίρετε, καλή μέρα!
Hindi:     □□□□□□
Icelandic: Halló, góður dagur!
Chinese:   □□, □□
Japanese:  □□□□□
Korean:    □□□□□
Russian:   Здравствуйте, добрый день!
```

Notice that the Chinese, Japanese, and Korean glyphs are not available in the *Source Code Pro* typewriter font. However, the accented letters in the Greek text are correctly displayed, whereas they were lost in our prose font.

8 Arial Unicode MS samples

Before the start of this section, I changed the main font family with the input

```
\setmainfont {Arial Unicode MS}
```

That font[12] has a large repertoire of 38 918 glyphs. It is not a free font, but can be found in a vendor font directory on Apple macOS. Although it was commissioned by Microsoft to extend a popular sans serif typeface to cover a large portion of Unicode, it is not a system font in the Windows operating system. It was, however, included in several Microsoft desktop products between 2000 and 2010, where it was installed in `c:\Windows\Fonts\ArialUni.ttf`. Microsoft no longer distributes or supports it, but updated versions are reportedly available from Monotype.[13]

Because Arial Unicode MS was designed as a fallback font to supply glyphs missing in other, smaller, fonts, it does not contain the usual style variations of bold, italic, small caps, and so on: its only shape is sans serif.

This is `\textrm{...}` plain text. The quick brown fox jumped over the lazy dog. THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.

This is a `quotation` environment in normal prose:

> English: Hello, good day!
> Greek: Χαίρετε, καλή μέρα
> Hindi: नमस्ते
> Icelandic: Halló, góður dagur!

[12] See `https://en.wikipedia.org/wiki/Arial_Unicode_MS`.
[13] See `https://docs.microsoft.com/en-us/typography/font-list/arial-unicode-ms`.

Chinese: 你好, 早晨
Japanese: こんにちは
Korean: 안녕하세요
Russian: Здравствуйте, добрый день!

The output correctly shows all of the glyphs. There is no separate typewriter font in Arial Unicode MS, so those tests are omitted.

9 Code2000 samples

Before the start of this section, I changed the font families with the input

```
\setmainfont {Code2000}
\setmonofont {Code2000} [Scale = MatchLowercase]
```

This is \textrm{...} plain text. The quick brown fox jumped over the lazy dog. THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.

This is a quotation environment in normal prose:

English: Hello, good day!
Greek: Χαίρετε, καλή μέρα
Hindi: नमस्ते
Icelandic: Halló, góður dagur!
Chinese: 你好, 早晨
Japanese: こんにちは
Korean: 안녕하세요
Russian: Здравствуйте, добрый день!

The output correctly shows all of the glyphs. There is no separate typewriter font in Code2000, so those tests are omitted.

10 GNU FreeFont samples

Before the start of this section, I changed the main and typewriter font families to FreeFont,[14] with the input

```
\setmainfont {FreeSerif}
\setmonofont {FreeMono} [Scale = MatchLowercase]
```

[14] See https://www.gnu.org/software/freefont/.

This is \textrm{...} plain text. The quick brown fox jumped over the lazy dog. THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.

This is a quotation environment in normal prose:

> English: Hello, good day!
> Greek: Χαίρετε, καλή μέρα
> Hindi: नमस्ते
> Icelandic: Halló, góður dagur!
> Chinese: ??, ??
> Japanese: ?????
> Korean: ?????
> Russian: Здравствуйте, добрый день!

This is a verbatim environment inside a quotation environment:

```
English:   Hello, good day!
Greek:     Χαίρετε, καλή μέρα!
Hindi:     ???????
Icelandic: Halló, góður dagur!
Chinese:   ??, ??
Japanese:  ?????
Korean:    ?????
Russian:   Здравствуйте, добрый день!
```

11 GNU Unifont samples

Before the start of this section, I changed the main and typewriter font families to Unifont, with the input

```
\setmainfont {unifont-12.1.04}
\setmonofont {unifont-12.1.04} [Scale = MatchLowercase]
```

There have been dozens of releases of Unifont; the selected version is the latest available at the time of writing this.

This is \textrm{...} plain text. The quick brown fox jumped over the lazy dog. THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.

These samples set in a quotation environment with explicit line breaks show that Unifont appears to be a mostly monospaced (typewriter) font within a single alphabet, and might be usable in LaTeX verbatim environments:

```
.....    ,,,,,
,,,,,,,,,    ,,,,,
:::::    ;;;;;
#####    $$$$$
%%%%%    &&&&&
(((((    )))))
{{{{{    }}}}}
[[[[[    ]]]]]
<<<<<    >>>>>
*****    +++++
-----    /////
|||||    =====
!!!!!    ?????
IIIII    iiiii
MMMMM    mmmmm
WWWWW    wwwww
0123456789
1234567890
2345678901
3456789012
4567890123
ABCDEFGHIJKLMNOPQRSTUVWXYZ    abcdefghijklmnopqrstuvwxyz
BCDEFGHIJKLMNOPQRSTUVWXYZA    bcdefghijklmnopqrstuvwxyza
CDEFGHIJKLMNOPQRSTUVWXYZAB    cdefghijklmnopqrstuvwxyzab
DEFGHIJKLMNOPQRSTUVWXYZABC    defghijklmnopqrstuvwxyzabc
EFGHIJKLMNOPQRSTUVWXYZABCD    efghijklmnopqrstuvwxyzabcd
```

Notice that the Unifont commas, apostrophes, and quotation mark characters are narrower than other characters. Those characters are common in computer input and output, so verbatim displays in Unifont would have noticeable, and unwanted, irregularities.

This is a quotation environment in normal prose:

> English: Hello, good day!
> Greek: Χαίρετε, καλή μέρα
> Hindi: नमस्ते
> Icelandic: Halló, góður dagur!
> Chinese: 你好, 早晨
> Japanese: こんにちは
> Korean: 안녕하세요
> Russian: Здравствуйте, добрый день!

Unifont supplies all of the glyphs needed for the eight languages illustrated here.

This is a verbatim environment inside a quotation environment:

```
English:   Hello, good day!
Greek:     Χαίρετε, καλή μέρα!
Hindi:     नमस्ते
Icelandic: Halló, góður dagur!
Chinese:   你好, 早晨
Japanese:  こんにちは
Korean:    안녕하세요
Russian:   Здравствуйте, добрый день!
```

Once again, we got complete glyph coverage, because we chose the same Unifont name in the `\setmainfont` and `\setmonofont` commands.

12 Linux Libertine O samples

Before the start of this section, I changed the main and typewriter font families with the input

```
\setmainfont {Linux Libertine O}
\setmonofont {Linux Libertine Mono O} [Scale = MatchLowercase]
```

This is `\textrm{...}` plain text. The quick brown fox jumped over the lazy dog. THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.

This is a `quotation` environment in normal prose:

> English: Hello, good day!
> Greek: Χαίρετε, καλή μέρα
> Hindi: □□□□□□
> Icelandic: Halló, góður dagur!
> Chinese: □□, □□
> Japanese: □□□□□
> Korean: □□□□□
> Russian: Здравствуйте, добрый день!

This is a `verbatim` environment inside a `quotation` environment:

> ```
> English:   Hello, good day!
> Greek:     □□□□□□□, □□□□ □□□□!
> Hindi:     □□□□□□
> Icelandic: Halló, góður dagur!
> Chinese:   □□, □□
> Japanese:  □□□□□
> Korean:    □□□□□
> Russian:   □□□□□□□□□□□□, □□□□□□ □□□□!
> ```

13 Roboto samples

Before the start of this section, I changed the main and typewriter font families with the input

```
\setmainfont {Roboto-Regular}
\setmonofont {RobotoMono-Regular} [Scale = MatchLowercase]
```

This is `\textrm{...}` plain text. The quick brown fox jumped over the lazy dog. THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.

This is a `quotation` environment in normal prose:

> English: Hello, good day!
> Greek: Χαίρετε, καλή μέρα
> Hindi: □□□□□□
> Icelandic: Halló, góður dagur!
> Chinese: □□, □□
> Japanese: □□□□□
> Korean: □□□□□
> Russian: Здравствуйте, добрый день!

This is a `verbatim` environment inside a `quotation` environment:

```
English:   Hello, good day!
Greek:     Χαίρετε, καλή μέρα!
Hindi:
Icelandic: Halló, góður dagur!
Chinese:     ,
Japanese:
Korean:
Russian:   Здравствуйте, добрый день!
```

Notice that missing glyphs in the typewriter font are not marked. That is indeed unfortunate, because it means that any use of the font would need careful proofreading. Fortunately, the typeset log file records those deficiencies with messages like this one:

```
Missing character: There is no X (U+03A7) in font
LinuxLibertineMonoO:mode=node;script=latn;language=DFLT;!
```

14 Typesetting text in multiple scripts

This section returns to the main document fonts, with these commands

Table 1: Multilingual text *with* language tagging.

English: All human beings are born free and equal in dignity and rights. They are endowed with reason and conscience and should act towards one another in a spirit of brotherhood.	**Greek**: Όλοι οι άνθρωποι γεννιούνται ελεύθεροι και ίσοι στην αξιοπρέπεια και τα δικαιώματα. Είναι προικισμένοι με λογική και συνείδηση, και οφείλουν να συμπεριφέρονται μεταξύ τους με πνεύμα αδελφοσύνης.	**Icelandic**: Hver maður er borinn frjáls og jafn öðrum að virðingu og réttindum. Menn eru gæddir vitsmunum og samvizku, og ber þeim að breyta bróðurlega hverjum við annan.	**Polish**: Wszyscy ludzie rodzą się wolni i równi pod względem swej godności i swych praw. Są oni obdarzeni rozumem i sumieniem i powinni postępować wobec innych w duchu braterstwa.	**Russian**: Все люди рождаются свободными и равными в своем достоинстве и правах. Они наделены разумом и совестью и должны поступать в отношении друг друга в духе братства.

```
\setmainfont {Noto Serif}
\setmonofont {Source Code Pro} [Scale = MatchLowercase]
```

issued just before the heading.

In most of this document, we use only short fragments of text outside the initial 128-character ASCII block of Unicode, and apart from the fonts themselves, no special additional support is needed.

If you need to set longer fragments of multilingual text, however, then you should use an additional powerful package, babel. It supports scores of languages, with separate hyphenation rules for each of them. If you have text in, say, five particular languages, you can use this command in the LaTeX document preamble:

```
\usepackage [greek, icelandic, polish,
             russian, english]          {babel}
```

The last language listed is the default one. You can then switch from one to another with a `\selectlanguage` command, as in Table 1 with a paragraph from Article 01 of the United Nations *Universal Declaration of Human Rights*. We intentionally chose a short line width in the boxes to force hyphenation. The text is typeset with input like this:

```
\newcommand {\boxspace} {\hskip 1em plus 0.5em minus 0.5em}
\newlength  {\boxwidth}
\setlength  {\boxwidth} {0.175\textwidth}
```

Table 2: Multilingual text *without* language tagging.

English: All human beings are born free and equal in dignity and rights. They are endowed with reason and conscience and should act towards one another in a spirit of brotherhood.	**Greek**: Όλοι οι άνθρωποι γεννιούνται ελεύθεροι και ίσοι στην αξιοπρέπεια και τα δικαιώματα. Είναι προικισμένοι με λογική και συνείδηση, και οφείλουν να συμπεριφέρονται μεταξύ τους με πνεύμα αδελφοσύνης.	**Icelandic**: Hver maður er borinn frjáls og jafn öðrum að virðingu og réttindum. Menn eru gæddir vitsmunum og samvizku, og ber þeim að breyta bróðurlega hverjum við annan.	**Polish**: Wszyscy ludzie rodzą się wolni i równi pod względem swej godności i swych praw. Są oni obdarzeni rozumem i sumieniem i powinni postępować wobec innych w duchu braterstwa.	**Russian**: Все люди рождаются свободными и равными в своем достоинстве и правах. Они наделены разумом и совестью и должны поступать в отношении друг друга в духе братства.

```
\begin{center}
    \parbox [t] {\boxwidth} {textbf{English}:
                             \selectlanguage {english}...}%
    \boxspace
    \parbox [t] {\boxwidth} {textbf{Greek}:
                             \selectlanguage {greek}...}%
    \boxspace
    \parbox [t] {\boxwidth} {textbf{Icelandic}:
                             \selectlanguage {icelandic}...}%
    \boxspace
    \parbox [t] {\boxwidth} {textbf{Polish}:
                             \selectlanguage {polish}...}%
    \boxspace
    \parbox [t] {\boxwidth} {textbf{Russian}:
                             \selectlanguage {russian}...}%
\end{center}
```

Table 2 shows the same example, but with the `\selectlanguage` commands removed. There are fewer hyphenations in the last three languages, and because the patterns that chose them come from the default of English, they are likely to be wrong. Right justification of narrow paragraphs without adequate hyphenation support produces lots of ugly whitespace, a phenomenon familiar to all newspaper readers.

You might wonder why the \selectlanguage command is needed. Could TEX not determine the language, and thus its hyphenation rules, from the code block of the Unicode text? The answer is no, because most alphabetic scripts have been used in multiple languages and countries, and hyphenation practices differ. Here is one example for regions of English:

American	coura-geously
Australian	cour-ageously
British	cour-ageously
Canadian	coura-geously
Syllabic	cour-a-geous-ly

Another example comes from the German language reform of 1996 that was jointly adopted by the governments of Austria, Germany, and Switzerland. It is called *Die neue Regelung der Rechtschreibung* (*The new spelling rules*).[15] The rules are encoded in law, and changed spelling in some words, including those that differed among countries. The changes reduced the use of the ß (*Eszett* or *scharfes S*) character,[16] and modified hyphenation rules.

The German-language countries are not the only instance of spelling and hyphenation reforms: there is a long list of such actions in numerous regions recorded at

https://en.wikipedia.org/wiki/Spelling_reform

15 OpenType font attributes

Unicode fonts contain many additional user-accessible features, but until this section, we have not exploited any of them. Table 3 shows the same five narrow paragraphs as before, without the \setlanguage commands, but now we select some OpenType font attributes. We require letter boxes to expand by 20% of their normal widths, and we scale the normal width, stretch, and shrink of interword spaces and punctuation, with this command:

```
\setmainfont {Noto Serif}
             [
                 LetterSpace = 20,
                 WordSpace = {1.50, 1.75, 1.75},
                 PunctuationSpace = WordSpace
             ]
```

[15] See the German language description in https://de.wikipedia.org/wiki/Reform_der_deutschen_Rechtschreibung_von_1996. The law remains controversial, and has substantial opposition, so the rules may see future modification, or even withdrawal.

[16] See https://en.wikipedia.org/wiki/%C3%9F.

Table 3: Multilingual text *without* language tagging, but *with* modified font attributes. To avoid text overlapping into the next column, a discretionary hyphen has been added to each of the long Greek words *προικισμένοι*, *συμπεριφέρονται*, and *αδελφοσύνης*.

English:	**Greek**:	**Icelandic**:	**Polish**:	**Russian**:
All human beings are born free and equal in dignity and rights. They are endowed with reason and conscience and should act towards one another in a spirit of brotherhood.	Όλοι οι άνθρωποι γεννιούνται ελεύθεροι και ίσοι στην αξιοπρέπεια και τα δικαιώματα. Είναι προικισμένοι με λογική και συνείδηση, και οφείλουν να συμπεριφέρονται μεταξύ τους με πνεύμα αδελφοσύνης.	Hver maður er borinn frjáls og jafn öðrum að virðingu og réttindum. Menn eru gæddir vitsmunum og samvizku, og ber þeim að breyta bróðurlega hverjum við annan.	Wszyscy ludzie rodzą się wolni i równi pod względem swej godności i swych praw. Są oni obdarzeni rozumem i sumieniem i powinni postępować wobec innych w duchu braterstwa.	Все люди рождаются свободными и равными в своем достоинстве и правах. Они наделены разумом и совестью и должны поступать в отношении друг друга в духе братства.

16 Mixed writing directions

The luabidi package makes it straightforward to set text in left-to-right and right-to-left scripts, using `\setLTR` and `\setRTL` to switch writing direction.

In `emacs` version 24 (2011) or later, although the Hebrew text is stored in reading order, it is automatically displayed as right-to-left text. Older `emacs` versions show Hebrew letters, but display them left to right. To give an idea of the difficulty that this poses for the typist, here is that last sentence in reversed order: wohs snoisrev `scame` redlO .thgir ot tfel meht yalpsid tub ,srettel werbeH

Only three of the free fonts at my site have *Hebrew* in their names — *Droid Sans Hebrew*, *Noto Sans Hebrew*, and *Noto Serif Hebrew* — and they are available in *Bold* and *Regular* variants. However, when I attempted to use them in this section, I discovered that they

lack punctuation, so period and comma are displayed as empty boxes. They also lack Latin letters, making them unsuitable for mixed language text.

Next, I tried *Arial Unicode MS*, a sans serif font. It has the Hebrew letters, but I found that it was hard to distinguish some of the glyphs.

Several Hebrew letters have similar shapes that are distinguished by serifs, and their loss in a sans serif font causes readability problems.

To understand why, look at the Hebrew alphabet[17] in Tables 4 and 5. That alphabet has 22 letters, but 5 have final forms, just as Greek has variant forms ϖ, φ, and ς for π (pi), ϕ (phi), and σ (sigma). The tables display enlarged Hebrew letters to make it easier to see their fine details.

The tables do not tell everything about the alphabet: in particular, the Semitic languages Arabic and Hebrew traditionally omit vowels in normal writing, but the letters can be decorated with dots (Hebrew *dagesh*) and apostrophes (Hebrew *geresh*) to indicate pronunciation. The Unicode character set has separate encoding blocks of *presentation forms* for the ornamented letters of those scripts.

Because I am unfamiliar with Hebrew, I found it easiest to prepare the table using hexadecimal values of the letters, with input like this:

```
\newcommand {\he} [1]
            {{\setmainfont{Noto Serif Hebrew Regular}%
              \LARGE #1}}
\begin{tabular}{ccccc}
    Alef          & Bet           & Gimel         &
    Dalet         & He            \\
    \he{^^^^05d0} & \he{^^^^05d1} & \he{^^^^05d2} &
    \he{^^^^05d3} & \he{^^^^05d4} \\[2ex]
...
\end{tabular}
```

LuaTEX and XETEX both use the four-circumflex prefix on Unicode four-digit hexadecimal character numbers.

You can also represent arbitrary Unicode characters with the `\Uchar` command that expects a following integer. Thus, `\Uchar 1488` is another way to represent the Unicode hexadecimal character, `U+05d0`, for the letter *alef*. Of course, unless the current font has a glyph for that character, you may need to temporarily switch fonts, as we did with our `\he{...}` wrapper macro.

The last possible Unicode character is `U+10ffff` (decimal 1 114 111). Attempts to use larger decimal values for `\Uchar` produce a complaint from the Unicode typesetting engines like this:

[17] See also `https://en.wikipedia.org/wiki/Hebrew_alphabet`.

Table 4: Hebrew alphabet displayed with \LARGE characters in *Noto Serif Hebrew Regular*. When two letters are shown under a name, the lower one is the *final form* used at the end of a word. The letter ordering is opposite of that in normal Hebrew.

Alef	Bet	Gimel	Dalet	He
א	ב	ג	ד	ה
Waw/Vav	**Zayin**	**Chet**	**Tet**	**Yod**
ו	ז	ח	ט	י
Kaf	**Lamed**	**Mem**	**Nun**	**Samech**
כ	ל	מ	נ	ס
ך		ם	ן	
Ayin	**Pay**	**Tsade**	**Qof**	**Resh**
ע	פ	צ	ק	ר
	ף	ץ		
Shin	**Tav**			
ש	ת			

Table 5: Hebrew alphabet displayed with \LARGE characters in *Noto Sans Hebrew Regular*.

Alef	Bet	Gimel	Dalet	He
א	ב	ג	ד	ה
Waw/Vav	**Zayin**	**Chet**	**Tet**	**Yod**
ו	ז	ח	ט	י
Kaf	**Lamed**	**Mem**	**Nun**	**Samech**
כ	ל	מ	נ	ס
ך		ם	ן	
Ayin	**Pay**	**Tsade**	**Qof**	**Resh**
ע	פ	צ	ק	ר
	ף	ץ		
Shin	**Tav**			
ש	ת			

Table 6: Bilingual bidirectional text *without* language tagging.

English: Everyone is entitled to all the rights and freedoms set forth in this Declaration, without distinction of any kind, such as race, colour, sex, language, religion, political or other opinion, national or social origin, property, birth or other status.
Furthermore, no distinction shall be made on the basis of the political, jurisdictional or international status of the country or territory to which a person belongs, whether it be independent, trust, non-self-governing or under any other limitation of sovereignty.

Hebrew: כל אדם זכאי לזכויות ולחרויות שנקבעו בהכרזש זו ללא הפליה כלשהיא מטעמי גזע, צבע, מין, לשון, דת, דעה פוליטית או דעה בבעיות אחרות, בגלל מוצא לאומי או חברתי, קנין, לידה או מעמד אחר.
גדולה מזו, לא יופלה אדם על פי מעמדה המדיני, על פי סמכותה או על פי מעמדה הבינלאומי של המדינה או הארץ שאליה הוא שייך, בין שהארץ היא עצמאית, ובין שהיא נתונה לנאמנות, בין שהיא נטולת שלטון עצמי ובין שריבונותה מוגבלת כל הגבלה אחרת.

```
*\Uchar 1114111\relax
% OK, no complaints

*\Uchar 1114112\relax
! Bad character code (1114112).
<to be read again>
                   \relax
<*> \Uchar 1114112\relax

?
```

More Web searches led to a font archive at the *Open Siddur Project*,[18] where I found a `fonts-master.zip` file with 472 font files covering Arabic, Hebrew, Latin, and other alphabets. After examining several of them with `fontforge`, I chose the serif font *Shlomo* for the Hebrew text in the rest of this section.

Table 6 shows Article 02 of the *Universal Declaration of Human Rights* in English[19] and Hebrew,[20] without language selection. The Hebrew block begins with a `\setRTL` command, and that part of the example is set in *Shlomo*.

The placement of the `\setRTL` command is significant: if we move it *after* the input `\textbf{Hebrew}:`, then the block begins like this:

[18] See `https://opensiddur.org/help/fonts`.
[19] See `https://unicode.org/udhr/d/udhr_eng.html`.
[20] See `https://unicode.org/udhr/d/udhr_heb.html`.

Hebrew: כל אדם זכאי לזכויות ולחרויות שנקבעו בהכרזש זו ללא הפליה כלשהיא מטעמי גזע, צבע, מין, לשון, דת, דעה פוליטית או דעה בבעיות אחרות, בגלל מוצא לאומי או חברתי, קנין, לידה או מעמד אחר.

Even if you do not read Hebrew, you should be able to see that words have been reordered on the line. That is a feature of bidirectional typesetting that is familiar to readers of right-to-left languages. They are accustomed to seeing acronyms, foreign-language phrases, mathematics, and (what English-language speakers call 'arabic') numbers written in their 'normal' left-to-right directions. The frequent need for such mixtures is why modern Chinese, Japanese, and Korean typesetting is moving from historical text orderings to horizontal left-to-right display to reduce eye motion in reading.

17 Transliteration and translation

When you use additional scripts in your document, you can ensure that foreign language names are spelled correctly, and you can supply original language text that may be helpful for pasting into search boxes in Web browsers and library catalogs.

However, you should not assume that all of your readers can understand the extra scripts. You can help them by transcribing the foreign text into the main document language. The Google Translate[21] service supplies transliterations of non-Latin scripts, and for some, even offers an audio rendition. For English–Russian transliteration, there is a useful online service[22] that works in both directions.

Transliterations and translations can be often be supplied inline, like this: `She gave a friendly Greek greeting, Χαίρετε (Chaírete (Hello)).`

In bibliographic databases, it is good practice to identify the language as well, as in this example from a bibliography about Albert Einstein:

```
title =         "Теорияа Относительности: Общедоступное
                Излущение (Teoriaa Otnositel'nosti:
                Obshchedostypnoe Izloshenie) ({Russian})
                [{On} the Special and General Theory of
                Relativity in Common Understanding]",
```

[21] See `https://translate.google.com/`.
[22] See `https://translit.ru/`.

18 Further reading

There is much more to be said about Unicode fonts and their user-modifiable attributes. A good first start for LaTeX authors is Will Robertson's tutorial, *The fontspec package*, available in TeX Live trees at `texmf-dist/doc/latex/fontspec/fontspec.pdf`.

There are online encyclopedia articles about Unicode encodings and font formats at

```
https://en.wikipedia.org/wiki/OpenType
https://en.wikipedia.org/wiki/Private_Use_Areas
https://en.wikipedia.org/wiki/TrueType
https://en.wikipedia.org/wiki/UTF-8
https://en.wikipedia.org/wiki/Unicode_font
https://en.wikipedia.org/wiki/Unicode
```

The master site for the Unicode consortium at `https://home.unicode.org/` has numerous links to Unicode resources, including code charts at

```
https://www.unicode.org/charts/
```

Writing directions in East Asian scripts are described in encyclopedia articles at

```
https://en.wikipedia.org/wiki/Horizontal_and_vertical_
writing_in_East_Asian_scripts
https://en.wikipedia.org/wiki/Mongolian_writing_systems
```

There is more historical variation than the vertical right-to-left, and horizontal left-to-right, that I described earlier, but those two practices cover the majority of printed text in that region.

There is a master repository of hyphenation rules of many languages in TeX typesetting at

```
http://www.hyphenation.org/
```