

The fonttable package

Author: Peter Wilson, Herries Press

Maintainer: Will Robertson

will dot robertson at latex-project dot org

v1.6c 2017/06/06

Abstract

The package lets you typeset the characters in a font in tabular and/or running text forms.

Contents

1	Introduction	1
2	The package	1
2.1	Table and texts	2
2.2	Testing a glyph	4
3	The code	5
3.1	Table and texts	5
3.2	Testing a glyph	13
	Bibliography	15

1 Introduction

The fonttable package lets you typeset a font’s character set in tabular and/or running text forms.

This manual is typeset according to the conventions of the L^AT_EX DOC-STRIP utility which enables the automatic extraction of the L^AT_EX macro source files [MG04].

2 The package

The package provides commands to typeset a table of all the glyphs in a given font and to typeset an example of regular text. For font designers it provides commands to typeset a ‘test’ glyph among sets of glyphs from the font.

`\fnthours` As a convenience, `\fnthours` prints the time of day when the file was processed; it uses the 24 hour clock notation. (The macro `\today` prints the date when the file was processed.)

2.1 Table and texts

`\fonttable` The command `\fonttable{⟨testfont⟩}` typesets a table showing all the glyphs in the `⟨testfont⟩`, where `⟨testfont⟩` is the name of a font file¹ like `cmr10` (for Computer Modern Roman) or `pzdr` (for Zapf Dingbats).

NOTE: The `mftinc` package [Pak05] for pretty-printing METAFONT code also defines a `\fonttable` macro that is akin to this one. If you want to use both packages together then you can use the following general procedure for when a macro `\macro` is defined in both `packA` and `packB` packages.

```
\usepackage{packA}
\let\macroA\macro%   save packA's definition
\let\macro\relax%    undefine \macro
\usepackage{packB}%  now it's packB's definition of \macro
...
\macro % use the packB definition
\macroA % use the packA definition
```

`\xfonttable` The command `\xfonttable{⟨encoding⟩}{⟨family⟩}{⟨series⟩}{⟨shape⟩}` typesets a table showing all the glyphs in the font with encoding `⟨encoding⟩` (e.g., T1 or OMS), family `⟨family⟩` (e.g., `pp1` for Palatino or `cmbrs` for CM Bright Math (OMS)), font series `⟨series⟩` (e.g., `sb` for semibold of `m` for medium), and font shape `⟨shape⟩` (e.g., `n` for normal or `sc` for small caps). For example:
`\xfonttable{U}{pzd}{m}{n}`
 for Zapf Dingbats.

`\pikfont` The command² `\pikfont{⟨encoding⟩}{⟨family⟩}{⟨series⟩}{⟨shape⟩}` selects the font with encoding `⟨encoding⟩` (e.g., T1 or OMS), family `⟨family⟩` (e.g., `pp1` for Palatino or `cmbrs` for CM Bright Math (OMS)), font series `⟨series⟩` (e.g., `sb` for semibold of `m` for medium), and font shape `⟨shape⟩` (e.g., `n` for normal or `sc` for small caps). For example:

```
\pikfont{T1}{pp1}{m}{sc}
```

for Palatino small caps. The size of the font corresponds to the current setting (e.g., `\footnotesize`, `\normalsize`, `\Large`). It can also be changed after being selected by the incantation

```
\fontsize{⟨size⟩}{⟨baselineskip⟩}\selectfont
```

¹More precisely, the name of a `.tfm` file.

²The name was chosen in an attempt to avoid clashes with other macros that might perform similar functions.

where $\langle size \rangle$ is the normal height and $\langle baselineskip \rangle$ is the distance between text lines; the measurement system is **pts** but just use numbers with no units specified.

For example:

```
\fontsize{12}{15}\selectfont
```

for a 12pt font with 15pts between baselines.

If you are unsure about the meaning of the various arguments of `\xfonttable` and `\pikfont` see *The Companion* [MG04, Chapter 7] or the *LaTeX2e font selection* manual (`fntguide.tex`; try `texdoc fntguide`).

`\fontrange` The package attempts to populate the table with a maximum of 256 glyphs, numbered from 0 to 255. The `\fontrange{\langle low \rangle}{\langle high \rangle}` declaration changes this by reducing the range so that it extends from $\langle low \rangle$ to $\langle high \rangle$, where $\langle low \rangle$ should be at least 0 and $\langle high \rangle$ at most 256, and $\langle low \rangle$ less than $\langle high \rangle$.

The table is composed of blocks of sixteen characters. If necessary the value of $\langle low \rangle$ is adjusted lower and $\langle high \rangle$ is adjusted higher to match this block structure. For example, if you wanted a table of the lower 128 characters then `\fontrange{0}{127}` would do the job, while the upper half of a 256 character font could be tabulated via `\fontrange{128}{255}`.

`\decimals` Normally each cell in the table includes the decimal number of the position in the (256) character set. `\nodecimals` turns off this numbering and `\decimals` turns it on. The default is `\decimals`.

`\hexoct` Normally the columns and rows in the table are numbered using hexadecimal and octal numbers. These can be turned off by `\nohexoct` and turned on again with `\hexoct`, which is the default.

`\fetablewidth` The font table's width is the length `\fetablewidth`, which by default is set to the normal textwidth (or more exactly, to `\hsize`). The table itself is left aligned. However, if `\nohexoct` is in effect the width of the table is its natural width.

`\fntcolwidth` When `\nohexoct` is in effect the minimum width of a table column is `\fntcolwidth`. This is initially declared as

```
\setwidth{\fntcolwidth}{0.08\fetablewidth}
```

`\fonttext` The command `\fonttext{\langle testfont \rangle}` typesets an example text using the $\langle testfont \rangle$ (e.g. `cmr10`).

`\simpletext` The example text can be just a paragraph and a line of capitals, or include more complex accented words as well. Following the declaration `\fulltext` the complex words are included as well as the example paragraph. The default is `\simpletext` for just the paragraph.

`\regulartext` The command `\regulartext{\langle fontspec \rangle}` typesets the example text using $\langle fontspec \rangle$, for example `\rmfamily\itshape` or `\pikfont{T1}{pnc}{m}{it}`.

`\fonttexts` The macro `\fonttexts{\langle testfont \rangle}{\langle text \rangle}` typesets $\langle text \rangle$ using the $\langle testfont \rangle$ (e.g., `cmr10`). Similarly the macro `\regulartexts{\langle fontspec \rangle}{\langle text \rangle}` typesets $\langle text \rangle$ using $\langle fontspec \rangle$ (e.g., `\rmfamily\itshape` or `\pikfont{T1}{ppl}{m}{it}`).

`\germanparatext` `\germanparatext` expands to a German language paragraph, borrowed from the `blindtext` package [Lik05]. `\latinparatext` expands to one version of a paragraph of the traditional *lorem ipsum* dummy Latin text. Either, or both, of these could be used as the $\langle text \rangle$ argument to `\fonttexts` or `\regulartexts`.

NOTE: These were originally called `\germantext` and `\latintext` but on 2009/05/14 I was told that the `babel` package defines `\latintext`, which causes

unexpected results if it is used in the same document as this package. To try and be on the safe side I renamed `\germantext` as well as `\latintext`.

`\aztext` `\aztext` expands to the lowercase Latin alphabet a to z, and `\AZtext` is the corresponding command for the uppercase A to Z. The macros `\digitstext` and `\puncttext` expand respectively to the digits 0 to 9, and to the typical punctuation marks. In all cases there is a space between each character.

2.2 Testing a glyph

The macros here are a reimplementaion of Donald Knuth's `testfont.tex`, which is available from CTAN.

In the following, the value of a glyph argument can be specified as its location in the font (i.e., as a decimal number). With a few exceptions, if the glyph is within the visible ASCII range (33–126) it may instead be specified by the ASCII character prefixed with a single open quote mark³ (`'`). The exceptions are nos: 37 (`%`), 92 (`\`) 123 (`{`) and 125 (`}`) (but there may be others). In any case, the glyph representing the character `p` can be specified either as `'p` or as 112.

The glyphs are taken from the current font. If the font does not have Latin alphabet glyphs in the ASCII locations then in the descriptions below phrases like 'lowercase alphabet' or 'uppercase alphabet' or 'digits', should be taken to mean (the glyphs in) those locations.

`\glyphmixture` `\glyphmixture{<T>}{<S>}{<E>}` typesets the `<T>` (test) glyph between the glyphs in the range from `<S>` (start) to `<E>` (end). For example
`\glyphmixture{'e}'{'f}'{'g}` will produce
 efeeffeeffef
 egeeggeeeeggge

`\glyphalternation` `\glyphalternation{<T>}{<S>}{<E>}` typesets the `<T>` glyph alternately between each glyph in the range from `<S>` to `<E>`. For example
`\glyphalternation{'e}'{'f}'{'g}` will produce
 efefefefefefefe
 egegegegegegege

`\glyphseries` `\glyphseries{<T>}{<S>}{<E>}` typesets the `<T>` glyph between the glyphs in the range from `<S>` to `<E>`. For example
`\glyphseries{'e}'{'f}'{'h}` will produce
 efgehe

`\glyphalphabet` `\glyphalphabet{<T>}` typesets the `<T>` glyph between each letter of the lowercase Latin alphabet plus a few others. `\GLYPHALPHABET{<T>}` does the same but using the uppercase Latin alphabet. For example, the output of
`\glyphalphabet}{'3}` is like
 3a3b3c3d3e3f3g...3z3Ø3~3!3"3

`\glyphlowers` `\glyphlowers` takes each character of the lowercase alphabet in turn as a test glyph and sets it interspersed among the other lowercase characters.
`\glyphlowers` `\glyphuppers` and `\glyphdigits` are similar except that they use the uppercase alphabet and the ten digits instead. For example, `\glyphdigits` produces

³Sometimes called a 'backquote'.

output like
 000102030405060708090
 101112131415161718191
 202122232425262728292
 ...
 909192939495969798999

`\glyphpunct` `\glyphpunct` sets a collection of words with an assortment of punctuation marks.

3 The code

1 `(*pack)`

3.1 Table and texts

Most of the code below is an edited version of code used in `nfssfont.tex` for displaying aspects of the set of glyphs in a font.

```

\sevenrm A small fixed size roman font.
        2 \providecommand*\sevenrm{\fontsize{7}{9pt}\rmfamily}

\fontm Counts and a dimen.
\fontn 3 \newcount\fontm \newcount\fontn \newcount\fontp \newdimen\fontdim
\fontp 4
\fontdim
\fonttable \fonttable{font} typesets a table of all the glyphs in the font (e.g., auncl10).
        5 \newcommand*\fonttable[1]{%
        6   \def\fontname{#1}%
        7   \bgroup
        8   \fontstartfont
        9   \ftable
       10   \egroup}
       11

\pikfont \pikfont{encoding}{family}{series}{shape} selects the font with encoding,
        family, series and shape.
       12 \DeclareRobustCommand\pikfont[4]{%
       13   \fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}\selectfont}
       14

\fonttable \fonttable{encoding}{family}{series}{shape} typesets a table of all
the glyphs in the font with encoding, family, series and shape (e.g.,
\fonttable{T1}{pnc}{m}{it} for New Century Schoolbook italic). The original
code for the macro was supplied by Enrico Gregorio.
       15 \newcommand*\fonttable[4]{%
       16   \begingroup
       17     \pikfont{#1}{#2}{#3}{#4}%
       18     \edef\fontname{\fontname\font}%

```

New: strip any size information from the fontname (which could be, e.g., either ‘cmr10’ or ‘cmr10_{at}10pt’.) This wasn’t necessary before because we didn’t explicitly choose the font size; it was inferred automatically.

```

19 \edef\@tempa{ \string a\string t}%
20 \edef\@tempb{\noexpand\in@\@tempa-\f@tfontname}%
21 \@tempb
22 \ifin@
23 \edef\f@tfontname{\expandafter\f@tstripsize\f@tfontname}%
24 \fi

```

End new code, and finish as before:

```

25 \normalfont
26 \f@tstartfont
27 \f@table
28 \endgroup
29 }

```

`\f@tstripsize` Needed above.

```

30 \edef\@tempa{%
31 \def\noexpand\f@tstripsize
32 ##1 \string a\string t##2\string p\string t{##1}%
33 }
34 \@tempa

```

`\f@tstartfont` Sets up for a font table.

```

35 \newcommand*\f@tstartfont-{}%

```

New: scale the font by 0.01% to (attempt to) avoid TeX’s font optimisation. This becomes a problem in Spanish babel, say, when `\textfont\fam` changes when `cmr10` has been loaded under a different name, here. (And the `\textfont` can no longer be parsed correctly. See: <http://latex-alive.tumblr.com/post/3229118083/texs-font-loading-optimisation>)

```

36 \@tempdima=\f@size pt
37 \font\f@ttestfont=\f@tfontname\space at 0.9999\@tempdima\relax

```

Continue as before:

```

38 \f@ttestfont \f@tsetbaselineskip
39 \ifdim\fontdimen6\f@ttestfont<10pt\relax
40 \rightskip=0pt plus 20pt\relax
41 \else
42 \rightskip=0pt plus 2em\relax
43 \fi
44 \spaceskip=\fontdimen2\f@ttestfont % space between words (\raggedright)
45 \xspaceskip=\fontdimen2\f@ttestfont
46 \advance\xspaceskip by\fontdimen7\f@ttestfont
47 }

```

`\f@tsetbaselineskip`

```

48 \newcommand*\f@tsetbaselineskip-{\setbox0=\hbox{\f@tn=0
49 \loop\char\f@tn \ifnum \f@tn<255 \advance\f@tn 1 \repeat}

```

```

50 \baselineskip=6pt \advance\baselineskip\ht0 \advance\baselineskip\dp0 }
51

\ftoct \ftoct{<onum>} typesets the octal constant <onum>.
52 \newcommand*{\ftoct}[1]{\hbox{\rmfamily\'}\kern-.2em\itshape
53      #1/\kern.05em}} % octal constant

\fthex \fthex{<hnum>} typesets the hexadecimal constant <hnum>.
54 \newcommand*{\fthex}[1]{\hbox{\rmfamily\H}\ttfamily#1}} % hexadecimal constant

\ftsetdigs \ftsetdigs
55 \def\ftsetdigs#1"#2{\gdef\h{#2}% \h=hex prefix; \0\1=corresponding octal
56 \ftm=\ftn \divide\ftm by 64 \xdef\0{\the\ftm}%
57 \multiply\ftm by-64 \advance\ftm by\ftn \divide\ftm by 8 \xdef\1{\the\ftm}}

\fttestrow \fttestrow checks if there are any characters in the next block of 16 slots.
58 \newcommand*{\fttestrow}{\setbox0=\hbox{\penalty 1\def\{\char"\h}%
59 \0\1\2\3\4\5\6\7\8\9\A\B\C\D\E\F%
60 \global\ftp=\lastpenalty}} % \ftp=1 if none of the characters exist
61

\ifhexoct Flag for (not) setting hex and octal numbers.
  \hexoct
\nohexoct 62 \newif\ifhexoct
63 \newcommand*{\hexoct}{\hexocttrue}
64 \newcommand*{\nohexoct}{\hexoctfalse}
65 \hexoct
66

\ftoddlinenum \ftodddline
67 \newcommand*{\ftodddline}{\cr
68 \noalign{\nointerlineskip}
69 \multispan{19}\hrulefill&
70 \setbox0=\hbox{\lower 2.3pt\hbox{\fthex{\h x}}}\smash{\box0}
71 \cr
72 \noalign{\nointerlineskip}}
73

\iff@tskipping
\ftskippingtrue 74 \newif\iff@tskipping
\ftskippingfalse 75

\fonrange \fonrange{<low>}{<high>} sets the character range to be output.
76 \newcommand*{\fonrange}[2]{%
77 \ifnum#1<#2\relax

Set \ftlow to the nearest multiple of 16 that is at or below <low>, but first make
sure that it will be at least 0.

78 \ifnum#1<\z@
79 \ftm=\z@

```

```

80 \else
81   \f@tm=#1
82   \divide \f@tm \sixt@0n
83   \multiply \f@tm \sixt@0n
84 \fi
85 \edef\f@tlow{\the\f@tm}

```

Set `\f@thigh` to the nearest multiple of 16 at or above *(high)*, finally making sure that its maximum is 256.

```

86 \f@tm=#2
87 \divide \f@tm \sixt@0n
88 \advance \f@tm \@ne
89 \multiply \f@tm \sixt@0n
90 \ifnum \f@tm > \@ccclvi \f@tm=\@ccclvi \fi
91 \edef\f@thigh{\the\f@tm}
92 \else
93   \PackageError{fonttable}{%
94     Improper values for fontrange. Default values substituted}{\@ehc}
95   \def\f@tlow{0} \def\f@thigh{256}
96 \fi}
97 \fontrange{0}{256}
98

```

`\f@tloopforsixteen` `\f@tloopforsixteen` sets up a block of sixteen character slots.

```

99 \newcommand*{\f@tloopforsixteen}{%
100 \ifnum\f@tn<\f@tlow \global\f@tn=\f@tlow\fi
101 \loop\f@tskippingfalse
102 \ifnum\f@tn<\f@thigh \f@tm=\f@tn \divide\f@tm \sixt@0n \chardef\next=\f@tm
103 \expandafter\f@tsetdigs\meaning\next \f@ttestrow
104 \ifnum\f@tp=\@ne \f@tskippingtrue \fi\fi
105 \iff@tskipping \global\advance\f@tn \sixt@0n \repeat}
106

```

`\f@tevenline` `\f@tevenline` gets next non-empty set of a block of 16 characters. It either calls `\f@tmorechart` to print them, or `\f@tendchart` to finish off the table if all 256 potential characters have been processed.

`\f@tevenlinenonum` does something similar when no external numbers are printed.

```

107 \newcommand*{\f@tevenline}{%
108 \f@tloopforsixteen
109 \ifnum\f@tn=\f@thigh \let\next=\f@tendchart\else\let\next=\f@tmorechart\fi
110 \next}
111 \newcommand*{\f@tevenlinenonum}{%
112 \f@tloopforsixteen
113 \ifnum\f@tn=\f@thigh
114   \\hline
115 \else
116   \\hline
117   \f@tmorechartnonum
118 \fi}

```



```

153 \f@tevenlinenonum
154 \end{tabular}}
155 \newcommand*{\f@table}{\ifhexoct\f@tftablenum\else\f@tftablenonum\fi}
156

```

`\f@tendchart` `\f@tendchart` sets the last line of an externally numbered table with the relevant hex digits.

```

157 \newcommand*{\f@tendchart}{\cr\noalign{\hrule}
158 \raise11.5pt\null&&&\f@thex 8&&\f@thex 9&&\f@thex A&&\f@thex B&
159 &\f@thex C&&\f@thex D&&\f@thex E&&\f@thex F&\cr
160 \egroup$$\par}
161

```

`\f@tpsg` `\f@tpsg` typesets a single glyph, possibly with its decimal slot number. `\f@placechar` is the function to typeset the glyph with its number that is internally defined as `\f@placedecimal` if decimals are to be shown.

```

162 \newcommand*{\f@tpsg}{%
163 \setbox\z@=\hbox{\f@placechar{\char\f@tn}{\the\f@tn}}%
164 \ifdim\ht\z@>7.5pt\relax
165 \f@treposition
166 \else
167 \ifdim\dp\z@>2.5pt\relax
168 \f@treposition
169 \fi
170 \fi
171 \box\z@
172 \global\advance\f@tn\@ne
173 }

```

Change this definition to adjust the typesetting of the decimal numbers:

```

174 \newcommand*\f@placedecimal[2]{#1\ {\tiny #2}}

```

`\decimals` Following `\decimals`, which is the default, decimal numbers are printed in the table. Following `\nodecimals` they are not printed.

```

175 \newcommand*\f@nodecimals}{%
176 \renewcommand*\f@placechar{\@firstoftwo}%
177 }

178 \newcommand*\f@decimals}{%
179 \renewcommand*\f@placechar{\f@placedecimal}%
180 }
181 \newcommand*\f@placechar{}
182 \decimals

```

`\f@treposition` `\f@treposition`

```

183 \newcommand*{\f@treposition}{\setbox0=\vbox{\kern2pt\box0}\f@tdim=\dp0
184 \advance\f@tdim 2pt \dp0=\f@tdim}
185

```

`\fonttext` `\fonttext{}` typesets `\knutext` using `` (e.g. `auncl10`).

```
186 \def\fonttext#1{%
187   \def\fontname{#1}%
188   \bgroup
189   \fontstartfont
190   \knutext
191   \egroup}
192
```

`\regulartext` `\regulartext{<fontspec>}` typesets `\knutext` using `<fontspec>` (e.g., `\aunclfamily`).

```
193 \def\regulartext#1{%
194   \bgroup
195   #1
196   \knutext
197   \egroup}
198
```

`\knutext` Deathless prose from Knuth for testing a font. It includes `\moreknutext`, `\capknutext`, and `\knunames`.

```
199 \def\knutext{{
200 On November 14, 1885, Senator \& Mrs.~Leland Stanford called together
201 at their San Francisco mansion the 24~prominent men who had been
202 chosen as the first trustees of The Leland Stanford Junior University.
203 They handed to the board the Founding Grant of the University, which
204 they had executed three days before. This document---with various
205 amendments, legislative acts, and court decrees---remains as the
206 University's charter. In bold, sweeping language it stipulates that
207 the objectives of the University are 'to qualify students for
208 personal success and direct usefulness in life; and to promote the
209 publick welfare by exercising an influence in behalf of humanity and
210 civilization, teaching the blessings of liberty regulated by law, and
211 inculcating love and reverence for the great principles of government
212 as derived from the inalienable rights of man to life, liberty, and
213 the pursuit of happiness.'
214
215 \moreknutext
216
217 \capknutext
218
219 \knunames
220 \par}}
221
```

`\@moreknutext` Some more text with a variety of ligatures and accents.

```
222 \def\@moreknutext{'But aren't Kafka's Schlo{\ss} and {\AE}sop's
223 {\OE}uvres often na{"i}ve vis-\`a-vis the d{\ae}monic ph{\oe}nix's
224 official r\`ole in fluffy souffl\`es? }
225
```

```

\@capkntext Text using only capital letters and some punctuation.
\capkntext 226 \newcommand{\@capkntext}{%
227 (!'THE DAZED BROWN FOX QUICKLY GAVE 12345--67890 JUMPS!)}
228 \let\capkntext\@capkntext
229

\@knunames Lots of accents masquerading in personal names.
230 \def\@knunames{ {\AA}ngel\aa\ Beatrice Claire
231 Diana \’Erica Fran\c{c}oise Ginette H\’el\’ene Iris
232 Jackie K\=aren {\L}au\ra Mar{\’\i}a N\H{a}ta{\l}{\u\i}e {\0}ctave
233 Pauline Qu\~eneau Roxanne Sabine T\~a{\’\j}a Ur\v{s}ula
234 Vivian Wendy Xanthippe Yv{\o}nne Z\”azilie\par}
235

\guillemotleft Just in case the French quotes are not defined, as they are called for in the subse-
\guillemotright quent \germantext.
\flqq 236 \DeclareTextSymbol{\guillemotleft}{OT1}{‘\’}
\frqq 237 \DeclareTextSymbol{\guillemotright}{OT1}{‘\’}
238 \providecommand{\flqq}{\guillemotleft}
239 \providecommand{\frqq}{\guillemotright}
240

\germantext Text from the Blindtext package.
\germanparatext 241 \providecommand*\germantext}{%
242 \PackageWarning{fonttable}{\protect\germantext\space is deprecated,
243 \MessageBreak use \protect\germanparatext\space instead}}
244 \newcommand*\germanparatext}{%
245 Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer
246 diesen Text liest, ist selbst schuld. Der Text gibt lediglich den
247 Grauwert der Schrift an. Ist das wirklich so? Ist es
248 gleich\~g\”ul\~tig ob ich schreibe: \frqq Dies ist ein
249 Blindtext\flqq\ oder \frqq Huardest gefburn\flqq? Kjift --
250 mitnichten! Ein Blindtext bietet mir wichtige Informationen. An
251 ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie
252 harmonisch die Figuren zueinander stehen und pr\”u\~fe, wie breit
253 oder schmal sie l\”auft. Ein Blindtext sollte m\”og\~lichst viele
254 verschiedene Buchstaben enthalten und in der Originalsprache
255 gesetzt sein. Er mu\ss\ keinen Sinn ergeben, sollte aber lesbar
256 sein. Fremdsprachige Texte wie \frqq Lorem ipsum\flqq\ dienen
257 nicht dem eigentlichen Zweck, da sie eine
258 falsche Anmutung vermitteln.\par}
259

\latintext The traditional printers’ text.
\latinparatext 260 \providecommand*\latintext}{%
261 \PackageWarning{fonttable}{\protect\latintext\space may be overridden by the
262 babel package \MessageBreak use
263 \protect\latinparatext\space instead}}
264 \newcommand*\latinparatext}{%

```

```

265 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam
266 lobortis facilisis sem. Nullam nec mi et neque pharetra
267 sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper,
268 felis non sodales commodo, lectus velit ultrices augue, a
269 dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie
270 ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in
271 sapien. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
272 Duis fringilla tristique neque. Sed interdum libero ut metus.
273 Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit
274 amet ante lobortis sollicitudin. Praesent blandit blandit mauris.
275 Praesent lectus tellus, aliquet aliquam, luctus a, egestas a,
276 turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum
277 turpis accumsan semper.\par}
278

```

```

\simpletext \simpletext kills off \moreknuttext and \knunames. \fulltext restores \moreknuttext
\fulltext and \knunames. Make \fulltext the default.
\moreknuttext 279 \newcommand*{\simpletext}{\let\moreknuttext\relax \let\knunames\relax}
\knunames 280 \newcommand*{\fulltext}{\let\moreknuttext@\moreknuttext \let\knunames@\knunames}
281 \fulltext
282

fonttexts \fonttexts{<font>}{<text>} typesets <text> using <font> (e.g. auncl10).
283 \def\fonttexts#1#2{%
284 \def\fontname{#1}%
285 \bgroup
286 \fontstartfont
287 #2
288 \egroup}
289

\regulartexts \regulartext{<fontspec>}{<text>} typesets <text> using <fontspec> (e.g., \aunclfamily).
290 \def\regulartexts#1#2{%
291 \bgroup
292 #1 #2
293 \egroup}
294

\aztext The various characters used for Latin texts.
\AZtext 295 \newcommand*{\aztext}{a b c d e f g h i j k l m n o p q r s t u v w x y z}
\digitstext 296 \newcommand*{\AZtext}{A B C D E F G H I J K L M N O P Q R S T U V W X Y Z}
\punctext 297 \newcommand*{\digitstext}{0 1 2 3 4 5 6 7 8 9}
298 \newcommand*{\punctext}{' ! @ \ $ \% * ( ) \_ - + = [ ] < > \{ \} : ; ' , . ? /}
299

```

3.2 Testing a glyph

This is a reimplementaion of Donald Knuth's `testfont.tex` which is available from CTAN and there is also a commented version in Appendix H of *The META-FONT Book*.

```

\fnthours The time of day on a 24 hour clock.
\fttwodigits 300 %%% using \@tempcnta for Knuth's \m and \@tempcntb for his \n
301 \newcommand*\fnthours{\@tempcntb=\time \divide\@tempcntb 60
302 \@tempcnta=-\@tempcntb \multiply\@tempcnta 60 \advance\@tempcnta \time
303 \fttwodigits\@tempcntb:\fttwodigits\@tempcnta}
304 \newcommand*\fttwodigits}[1]{\ifnum #1<10 0\fi \number#1}
305

\ftgettsechars \ftgettsechars{⟨T⟩}{⟨S⟩}{⟨E⟩} gets three characters and \chardefs \fttchar
\fttchar to ⟨T⟩ (the test character), \ftschar to ⟨S⟩ (start character) and \ftechar to
\ftschar ⟨E⟩ (the end character).
\ftechar 306 \newcommand*\ftgettsechars}[3]{%
307 \chardef\fttchar=#1 \chardef\ftschar=#2 \chardef\ftechar=#3}
308

\glyphmixture \glyphmixture{⟨T⟩}{⟨S⟩}{⟨E⟩} sets a mix of ⟨T⟩ within the glyph range from
\ftmixpattern ⟨S⟩ to ⟨E⟩ according to the pattern \ftmixpattern. The work is done by
\ftdomix \ftdomix.
309 \newcommand*\glyphmixture}[3]{\ftgettsechars{#1}{#2}{#3}%
310 \ftdomix\ftmixpattern}
311 \newcommand*\ftmixpattern}{\0\1\0\0\1\1\0\0\0\1\1\1\0\1}
312 \newcommand*\ftdomix}[1]{\par\chardef\0=\fttchar \@tempcntb=\ftschar
313 \loop \chardef\1=\@tempcntb #1\endgraf
314 \ifnum \@tempcntb<\ftechar \advance\@tempcntb \@ne \repeat}
315

\glyphalternation These are similar to \glyphmixture and \ftmixpattern except that the glyphs
\ftaltpattern are alternated.
316 \newcommand*\glyphalternation}[3]{\ftgettsechars{#1}{#2}{#3}%
317 \ftdomix\ftaltpattern}
318 \newcommand*\ftaltpattern}{\0\1\0\1\0\1\0\1\0\1\0\1\0\1\0\1\0}
319

\ftdisc For breaking long lines so that the test character will be at the end of one line
and repeated at the start of the next one.
320 \newcommand*\ftdisc{\discretionary{\fttchar}{\fttchar}{\fttchar}}
321

\glyphseries \glyphseries{⟨T⟩}{⟨S⟩}{⟨E⟩} puts the test character ⟨T⟩ between all the others
\ftdoseries in the range ⟨S⟩ to ⟨E⟩. The work is done by \ftdoseries.
322 \newcommand*\glyphseries}[3]{\ftgettsechars{#1}{#2}{#3}%
323 \ftdisc\ftdoseries\ftschar\ftechar\par}
324 \newcommand*\ftdoseries}[2]{\@tempcntb=#1\relax
325 \loop\char\@tempcntb\ftdisc
326 \ifnum\@tempcntb<#2\advance\@tempcntb \@ne \repeat}
327

```

`\glyphalphabet` `\glyphalphabet{⟨T⟩}` inserts the test glyph `⟨T⟩` between the lowercase alphabetic characters. Similarly `\GLYPHALPHABET{⟨T⟩}` does the same with the uppercase characters. The work is done by, respectively, `\f@tcomplower` and `\f@tcompupper`.

```
328 \newcommand*\glyphalphabet{\f@tcomplower}
329 \newcommand*\GLYPHALPHABET{\f@tcompupper}
330 \newcommand*\f@tcomplower[1]{\chardef\f@ttchar=#1
331 \f@tdisc\f@tdoseries{a}{z}\f@tdoseries{31}{34}\par}
332 \newcommand*\f@tcompupper[1]{\chardef\f@ttchar=#1
333 \f@tdisc\f@tdoseries{A}{Z}\f@tdoseries{35}{37}\par}
334
```

`\glyphlowers` These macros generate an extended mix of characters of a particular kind. The work is done by `\f@tdocomprehensive` with `\f@tclc`, `\f@tcuc`, and `\f@tdgs` setting up the glyph sets.

```
\f@tclc 335 \newcommand*\glyphlowers{\f@tdocomprehensive\f@tclc{a}{z}{31}{34}}
\f@tcuc 336 \newcommand*\glyphuppers{\f@tdocomprehensive\f@tcuc{A}{Z}{35}{37}}
\f@tdgs 337 \newcommand*\glyphdigits{\f@tdocomprehensive\f@tdgs{0}{4}{5}{9}}
\f@tdocomprehensive 338 \newcommand*\f@tdocomprehensive[5]{\par\chardef\f@ttchar=#2
339 \loop{#1} \ifnum\f@ttchar<#3\@tempcnta=\f@ttchar\advance\@tempcnta \@ne
340 \chardef\f@ttchar=\@tempcnta \repeat
341 \chardef\f@ttchar=#4
342 \loop{#1} \ifnum\f@ttchar<#5\@tempcnta=\f@ttchar\advance\@tempcnta \@ne
343 \chardef\f@ttchar=\@tempcnta \repeat}
344 \newcommand*\f@tclc{\f@tdisc\f@tdoseries{a}{z}\f@tdoseries{31}{34}\par}
345 \newcommand*\f@tcuc{\f@tdisc\f@tdoseries{A}{Z}\f@tdoseries{35}{37}\par}
346 \newcommand*\f@tdgs{\f@tdisc\f@tdoseries{0}{9}\par}
347
```

`\glyphpunct` `\glyphpunct` sets punctuation marks in combination with different sorts of letters.

`\f@tdopunct` The work is done by `\f@tdopunct`.

```
348 \newcommand*\glyphpunct{\par\f@tdopunct{min}\f@tdopunct{pig}\f@tdopunct{hid}
349 \f@tdopunct{HIE}\f@tdopunct{TIP}\f@tdopunct{fluff}
350 \ $1,234.56 + 7/8 = 9% @ \#0\par}
351 \newcommand*\f@tdopunct[1]{#1,\ #1:\ #1;\ #'#1'\
352 ?'#1?\ !'#1!\ (#1)\ [#1]\ #1*\ #1.\par}
353
```

The end of the package.

```
354 </pack>
```

Bibliography

- [Lik05] Knut Lickert. *Blindtext.sty: Creating text for testing / Texterzeugung zum testen*. October 2005. (Available from CTAN in [macros/latex/contrib/blindtext](#))

- [MG04] Frank Mittelbach and Michel Goossens. *The LaTeX Companion*. Second edition. Addison-Wesley Publishing Company, 2004.
- [Pak05] Scott Pakin ‘The mftinc package’, January 2005. (Available from CTAN in `macros/latex/contrib/mftinc`)

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols	A	<code>\f@size</code> 36
<code>\"</code> 223, 234, 248, 252, 253	<code>\AA</code> 230	<code>\f@taltpattern</code> <u>316</u>
<code>\#</code> 350	<code>\aa</code> 230	<code>\f@tchartline</code>
<code>\\$</code> 298, 350	<code>\AE</code> 222 121, 122, <u>127</u>
<code>\%</code> 350	<code>\ae</code> 223	<code>\f@tchartstrut</code>
<code>\&</code> 200, 298	<code>\AZtext</code> <i>3</i> , <u>295</u> 130, <u>132</u> , 144
<code>\-</code> 248, 252, 253	<code>\aztext</code> <i>3</i> , <u>295</u>	<code>\f@tcllc</code> <u>335</u>
<code>\/</code> 53		<code>\f@tcol</code> <u>137</u>
<code>\=</code> 232	B	<code>\f@tcomplower</code> <u>328</u>
<code>\@capknuttext</code> <u>226</u>	<code>\baselineskip</code> 50	<code>\f@tcompupper</code> <u>328</u>
<code>\@cclvi</code> 90	C	<code>\f@tcuc</code> <u>335</u>
<code>\@ehc</code> 94	<code>\c</code> 231	<code>\f@tdgs</code> <u>335</u>
<code>\@firstoftwo</code> 176	<code>\capknuttext</code> 217, <u>226</u>	<code>\f@tdim</code> <i>3</i> , 183, 184
<code>\@knunames</code> <u>230</u> , 280	<code>\char</code> 49, 58, 163, 325	<code>\f@tdisc</code> <u>320</u> , 323, 325,
<code>\@moreknuttext</code> <u>222</u> , 280	<code>\chardef</code> 102, 307, 312,	331, 333, 344–346
<code>\@tempa</code> 19, 20, 30, 34	313, 330, 332,	<code>\f@tdocomprehensive</code> <u>335</u>
<code>\@tempb</code> 20, 21	338, 340, 341, 343	<code>\f@tdomix</code> <u>309</u> , 317
<code>\@tempcntb</code> 300–303,		<code>\f@tdopunct</code> <u>348</u>
312–314, 324–326	D	<code>\f@tdoseries</code> <u>322</u> ,
<code>\{</code> 298	<code>\decimals</code> <i>3</i> , <u>175</u>	331, 333, 344–346
<code>\}</code> 298	<code>\DeclareRobustCommand</code>	<code>\f@tchar</code> . <u>306</u> , 314, 323
<code>\^</code> 224, 233 12	<code>\f@tendchart</code> 109, <u>157</u>
<code>_</code> 298	<code>\DeclareTextSymbol</code> .	<code>\f@tevenline</code>
<code>\‘</code> 223, 231, 237 236, 237 <u>107</u> , 122, 148
<code>\~</code> 233	<code>\digitstext</code> <i>3</i> , <u>295</u>	<code>\f@tevenlinenonum</code> .
	<code>\discretionary</code> 320 <u>107</u> , 125, 153
Numbers	<code>\dp</code> 50, 167, 183, 184	<code>\f@tfontname</code> . 6, 18,
<code>\0</code> 55, 56,	E	20, 23, 37, 187, 284
128, 311, 312, 318	<code>\edef</code> 18–20, 23, 30, 85, 91	<code>\f@tftablenonum</code> . . . <u>142</u>
<code>\1</code> 55, 57, 121,	<code>\endgraf</code> 313	<code>\f@tftablenum</code> <u>142</u>
128, 311, 313, 318		<code>\f@tgettsechars</code> . . .
	F 306, 309, 316, 322
<code>_</code> 174, 230, 249,	<code>\f@placechar</code>	<code>\f@thex</code> . <u>54</u> , 70, 158, 159
255, 256, 351, 352 <u>162</u> , 176, 179, 181	<code>\f@thigh</code>
	<code>\f@placadecimal</code> <u>162</u> , 179	91, 95, 102, 109, 113

- `\f@tloopforsixteen` .
 99, 108, 112
`\f@tlow` 85, 95, 100
`\f@tm` 3, 56, 57, 79, 81–
 83, 85–91, 102, 121
`\f@tmixpattern` 309
`\f@tmorechart` . 109, 120
`\f@tmorechartnonum` .
 117, 120
`\f@tn` 3, 48, 49,
 56, 57, 100, 102,
 105, 109, 113,
 142, 150, 163, 172
`\f@toct` ... 52, 128, 147
`\f@toddl`ine 67, 121
`\f@toddlinenum` 67
`\f@tp` 3, 60, 104
`\f@tpsg` ... 128, 130, 162
`\f@trepos`ition
 165, 168, 183
`\f@tschar` . 306, 312, 323
`\f@tsetbaselineskip`
 38, 48
`\f@tsetdigs` 55, 103
`\f@tsimpleline`
 124, 125, 127
`\f@tsskip`pingfalse .
 74, 101
`\f@tsskip`pingtrue 74, 104
`\f@tstartchartnonum`
 137, 152
`\f@tstartfont`
 8, 26, 35, 189, 286
`\f@tstripsize` ... 23, 30
`\f@ttchar`
 . 306, 312, 320,
 330, 332, 338–343
`\f@ttestfont`
 37–39, 44–46
`\f@ttestrow` 58, 103
`\f@ttwodigits` 300
`\flqq` 236, 249, 256
`\fntcolwidth` 3, 132, 138
`\fnthours` 1, 300
`\font` 18, 37
`\fontdimen` .. 39, 44–46
`\fontencoding` 13
`\fontfamily` 13
`\fontname` 18
`\fontrange` 2, 76
`\fontseries` 13
`\fontshape` 13
`\fonttable` 1, 5
`\fonttext` 3, 186
`\fonttexts` . 3, 283, 283
`\frqq` . 236, 248, 249, 256
`\ftable` 9, 27, 142
`\ftablewidth` 3, 132, 143
`\fulltext` 3, 279
- G**
- `\germanparatext` . 3, 241
`\germantext` 241
`\GLYPHALPHABET` .. 4, 328
`\glyphalphabet` .. 4, 328
`\glyphalternation` 4, 316
`\glyphdigits` 4, 335
`\glyphowers` 4, 335
`\glyphmixture` ... 4, 309
`\glyphpunct` 4, 348
`\glyphseries` 4, 322
`\glyphuppers` 335
`\guillemotleft` ... 236
`\guillemotright` ... 236
- H**
- `\H` 54, 232
`\h` 55, 58, 70
`\hexoct` 3, 62
`\hexoctfalse` 64
`\hexocttrue` 63
`\hline` 114, 116, 124
`\hrulefill` 69
`\ht` 50, 164
- I**
- `\i` 223, 232
`\iff@tsskip`ping . 74, 105
`\ifhexoct` 62, 155
`\ifin@` 22
`\in@` 20
`\itshape` 52
- J**
- `\j` 233
- K**
- `\knunames` 219, 279
`\knutext` .. 190, 196, 199
- L**
- `\L` 232
`\l` 232
`\lastpenalty` 60
`\latinparatext` .. 3, 260
`\latintext` 260
`\loop` 49, 101,
 313, 325, 339, 342
`\lower` 70, 132, 146
- M**
- `\m` 300
`\meaning` 103
`\MessageBreak` . 243, 262
`\moreknutext` .. 215, 279
`\multicolumn` 138
`\multiply` 57, 83, 89, 302
`\multispan` 69
- N**
- `\n` 300
`\newcount` 3
`\newif` 62, 74
`\next` . 102, 103, 109, 110
`\noalign` 68, 72, 120, 157
`\nodecimals` 3, 175
`\noexpand` 20, 31
`\nohexoct` 3, 62
`\nointerlineskip` 68, 72
`\normalfont` 25
- O**
- `\O` 232
`\o` 234
`\OE` 223
`\oe` 223
- P**
- `\PackageError` 93
`\PackageWarning` 242, 261
`\pikfont` 2, 12, 17
`\providecommand` . 2,
 238, 239, 241, 260
`\punctext` 3, 295
- R**
- `\raise` 158
`\regulartext` 3, 193
`\regulartexts` ... 3, 290
`\repeat` 49, 105,
 314, 326, 340, 343

S	T	V
<code>\sevenrm</code> 2	<code>\tabskip</code> 144	<code>\v</code> 233
<code>\simpletext</code> 3, <u>279</u>	<code>\time</code> 301, 302	<code>\vbox</code> 132, 183
<code>\sixt@n</code> 82,	<code>\tiny</code> 174	<code>\vrule</code> 145
83, 87, 89, 102, 105	<code>\ttfamily</code> 54	X
<code>\smash</code> 70		<code>\xdef</code> 56, 57, 121
<code>\spaceskip</code> 44	U	<code>\xfonttable</code> 2, <u>15</u>
<code>\ss</code> 222, 255	<code>\u</code> 232	<code>\xspaceskip</code> 45, 46
<code>\string</code> 19, 32		