The **fonttable** package*

Author: Peter Wilson, Herries Press
Maintainer: Will Robertson
`will dot robertson at latex-project dot org`

2009/10/15

Abstract

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

Contents

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 LATEX DOC-STRIP utility which enables the automatic extraction of the LATEX macro source files [MG04].

---

*This file has version number v1.6, last revised 2009/10/15.

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 file[1] 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 defintion
\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., `ppl` 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 command[2]
`\pikfont{`⟨*encoding*⟩`}{`⟨*family*⟩`}{`⟨*series*⟩`}{`⟨*shape*⟩`}`
selects the font with encoding ⟨*encoding*⟩ (e.g., T1 or OMS), family ⟨*family*⟩ (e.g., `ppl` 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:

---

[1]More precisely, the name of a `.tfm` file.
[2]The name was chosen in an attempt to avoid clashes with other macros that might perform similar functions.

`\pikfont{T1}{ppl}{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`
where ⟨*size*⟩ is the normal height and ⟨*baselineskip*⟩ 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 selec-
tion* 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{`⟨*low*⟩`}{`⟨*high*⟩`}` declaration changes
this by reducing the range so that it extends from ⟨*low*⟩ to ⟨*high*⟩, where ⟨*low*⟩
should be at least 0 and ⟨*high*⟩ at most 256, and ⟨*low*⟩ less than ⟨*high*⟩.

   The table is composed of blocks of sixteen characters. If necessary the value
of ⟨*low*⟩ is adjusted lower and ⟨*high*⟩ 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
`\nodecimals`   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
`\nohexoct`   and octal numbers. These can be turned off by `\nohexoct` and turned on again
with `\hexoct`, which is the default.

`\ftablewidth`   The font table's width is the length `\ftablewidth`, 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\ftablewidth}`

`\fonttext`   The command `\fonttext{`⟨*testfont*⟩`}` typesets an example text using the
⟨*testfont*⟩ (e.g. `cmr10`).

`\simpletext`   The example text can be just a paragraph and a line of capitals, or include
`\fulltext`   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{`⟨*fontspec*⟩`}` typesets the example text using
⟨*fontspec*⟩, for example `\rmfamily\itshape` or `\pikfont{T1}{pnc}{m}{it}`.

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

`\germanparatext`   `\germanparatext` expands to a German language paragraph, borrowed from
`\latinparatext`

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 ⟨*text*⟩ 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
\AZtext   corresponding command for the uppercase A to Z. The macros \digitstext and
\digitstext   \punctext expand respectively to the digits 0 to 9, and to the typical punctuation
\punctext   marks. In all cases there is a space between each character.

2.2   Testing a glyph

The macros here are a reimplementation 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 mark[3] ('). 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
efeeffeeefffef
egeeggeeeggg g

\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
efegehe

\glyphalphabet   \glyphalphabet{⟨*T*⟩} typesets the ⟨*T*⟩ glyph between each letter of the low-
\GLYPHALPHABET   ercase Latin alphabet plus a few others. \GLYPHALPHABET{⟨*T*⟩} does the same but using the uppercase Latin alphabet. For example, the output of

---

[3]Sometimes called a 'backquote'.

`\glyphalphabet}{‘3}` is like
`3a3b3c3d3e3f3g...3z3Ø3˜3!3"3`

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

`00010203040506070809O`
`10111213141516171819I`
`202122232425262728292`
...
`9091929394959697989999`

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

3   The code

```
1 ⟨∗pack⟩
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{fonttable}[2009/10/15 v1.6 displays a font]
4
```

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.

```
5 \providecommand*{\sevenrm}{\fontsize{7}{9pt}\rmfamily}
```

\f@tm
\f@tn
\f@tp
\f@tdim
Counts and a dimen.

```
6 \newcount\f@tm \newcount\f@tn \newcount\f@tp \newdimen\f@tdim
7
```

\fonttable   `\fonttable{⟨font⟩}` typesets a table of all the glyphs in the ⟨font⟩ (e.g., auncl10).

```
8 \newcommand*{\fonttable}[1]{%
9   \def\f@tfontname{#1}%
10   \bgroup
11   \f@tstartfont
12   \ftable
13   \egroup}
14
```

\pikfont   `\pikfont{⟨encoding⟩}{⟨family⟩}{⟨series⟩}{⟨shape⟩}` selects the font with ⟨encoding⟩, ⟨family⟩, ⟨series⟩ and ⟨shape⟩.

```
15 \DeclareRobustCommand{\pikfont}[4]{%
16   \fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}\selectfont}
17
```

\xfonttable   \xfonttable{⟨*encoding*⟩}{⟨*family*⟩}{⟨*series*⟩}{⟨*shape*⟩} typesets a table of all
               the glyphs in the font with ⟨*encoding*⟩, ⟨*family*⟩, ⟨*series*⟩ and ⟨*shape*⟩ (e.g.,
               \xfonttable{T1}{pnc}{m}{it} for New Century Schoolbook italic). The original
               code for the macro was supplied by Enrico Gregorio.

```
18 \newcommand*{\xfonttable}[4]{\bgroup
19   \pikfont{#1}{#2}{#3}{#4}%
20   \edef\f@tfontname{\fontname\font}\normalfont
21   \f@tstartfont
22   \ftable
23   \egroup}
24
```

\f@tstartfont   Sets up for a font table.

```
25 \newcommand*{\f@tstartfont}{\font\f@ttestfont=\f@tfontname
26   \f@ttestfont \f@tsetbaselineskip
27   \ifdim\fontdimen6\f@ttestfont<10pt \rightskip=0pt plus 20pt
28   \else\rightskip=0pt plus 2em \fi
29   \spaceskip=\fontdimen2\f@ttestfont % space between words (\raggedright)
30   \xspaceskip=\fontdimen2\f@ttestfont \advance\xspaceskip
31   by\fontdimen7\f@ttestfont}
32
```

\f@tsetbaselineskip

```
33 \newcommand*{\f@tsetbaselineskip}{\setbox0=\hbox{\f@tn=0
34   \loop\char\f@tn \ifnum \f@tn<255 \advance\f@tn 1 \repeat}
35   \baselineskip=6pt \advance\baselineskip\ht0 \advance\baselineskip\dp0 }
36
```

\f@toct   \f@toct{⟨*onum*⟩} typesets the octal constant ⟨*onum*⟩.

```
37 \newcommand*{\f@toct}[1]{\hbox{\rmfamily\'{}\kern-.2em\itshape
38           #1\/\kern.05em}} % octal constant
```

\f@thex   \f@thex{⟨*hnum*⟩} typesets the hexadecimal constant ⟨*hnum*⟩.

```
39 \newcommand*{\f@thex}[1]{\hbox{\rmfamily\H{}\ttfamily#1}} % hexadecimal constant
```

\f@tsetdigs   \f@tsetdigs

```
40 \def\f@tsetdigs#1"#2{\gdef\h{#2}% \h=hex prefix; \0\1=corresponding octal
41   \f@tm=\f@tn \divide\f@tm by 64 \xdef\0{\the\f@tm}%
42   \multiply\f@tm by-64 \advance\f@tm by\f@tn \divide\f@tm by 8 \xdef\1{\the\f@tm}}
```

\f@ttestrow   \f@ttestrow checks if there are any characters in the next block of 16 slots.

```
43 \newcommand*{\f@ttestrow}{\setbox0=\hbox{\penalty 1\def\\{\char"\h}%
44   \\0\\1\\2\\3\\4\\5\\6\\7\\8\\9\\A\\B\\C\\D\\E\\F%
45   \global\f@tp=\lastpenalty}} % \f@tp=1 if none of the characters exist
46
```

\ifhexoct   Flag for (not) setting hex and octal numbers.
\hexoct
\nohexoct   47 \newif\ifhexoct

```
48 \newcommand*{\hexoct}{\hexocttrue}
49 \newcommand*{\nohexoct}{\hexoctfalse}
50 \hexoct
51
```

\f@toddlinenum   \f@toddline

```
52 \newcommand*{\f@toddline}{\cr
53   \noalign{\nointerlineskip}
54   \multispan{19}\hrulefill&
55   \setbox0=\hbox{\lower 2.3pt\hbox{\f@thex{\h x}}}\smash{\box0}
56   \cr
57   \noalign{\nointerlineskip}}
58
```

\iff@tskipping
\f@tskippingtrue   `59 \newif\iff@tskipping`
\f@tskippingfalse   `60`

\fontrange   \fontrange{⟨low⟩}{⟨high⟩} sets the character range to be output.

```
61 \newcommand*{\fontrange}[2]{%
62   \ifnum#1<#2\relax
```

Set \f@tlow to the nearest multiple of 16 that is at or below ⟨low⟩, but first make sure that it will be at least 0.

```
63   \ifnum#1<\z@
64     \f@tm=\z@
65   \else
66     \f@tm=#1
67     \divide \f@tm \sixt@@n
68     \multiply \f@tm \sixt@@n
69   \fi
70   \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.

```
71   \f@tm=#2
72   \divide \f@tm \sixt@@n
73   \advance \f@tm \@ne
74   \multiply \f@tm \sixt@@n
75   \ifnum \f@tm > \@cclvi  \f@tm=\@cclvi \fi
76   \edef\f@thigh{\the\f@tm}
77 \else
78   \PackageError{fonttable}{%
79     Improper values for fontrange. Default values substituted}{\@ehc}
80   \def\f@tlow{0} \def\f@thigh{256}
81   \fi}
82 \fontrange{0}{256}
83
```

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

```
84 \newcommand*{\f@tloopforsixteen}{%
85    \ifnum\f@tn<\f@tlow \global\f@tn=\f@tlow\fi
86    \loop\f@tskippingfalse
87    \ifnum\f@tn<\f@thigh \f@tm=\f@tn \divide\f@tm \sixt@@n \chardef\next=\f@tm
88    \expandafter\f@tsetdigs\meaning\next \f@ttestrow
89    \ifnum\f@tp=\@ne \f@tskippingtrue \fi\fi
90    \iff@tskipping \global\advance\f@tn \sixt@@n \repeat}
91
```

\f@tevenline   \f@tevenline gets next non-empty set of a block of 16 characters. It either calls
\f@tevenlinenonum \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.

```
92 \newcommand*{\f@tevenline}{%
93    \f@tloopforsixteen
94    \ifnum\f@tn=\f@thigh \let\next=\f@tendchart\else\let\next=\f@tmorechart\fi
95    \next}
96 \newcommand*{\f@tevenlinenonum}{%
97    \f@tloopforsixteen
98    \ifnum\f@tn=\f@thigh
99       \\\hline
100   \else
101      \\\hline
102      \f@tmorechartnonum
103   \fi}
104
```

\f@tmorechart   \f@tmorechart sets two lines of the table, and \f@tmorechartnonum does the
\f@tmorechartnonum same when there are no external numbers.

```
105 \newcommand*{\f@tmorechart}{\cr\noalign{\hrule\penalty5000}
106 \f@tchartline \f@toddline \f@tm=\1 \advance\f@tm 1 \xdef\1{\the\f@tm}
107 \f@tchartline \f@tevenline}
108 \newcommand*{\f@tmorechartnonum}{%
109    \f@tsimpleline \\ \hline
110    \f@tsimpleline \f@tevenlinenonum}
111
```

\f@tchartline   \f@tchartline does a line of the table, including external numbers, and
\f@tsimpleline   \f@tsimpleline does an unnumbered line.

```
112 \newcommand*{\f@tchartline}{%
113    &\f@toct{\0\1x}&&\f@tpsg{}&&\f@tpsg{}&&\f@tpsg{}&&\f@tpsg{}&&\f@tpsg{}&&\f@tpsg{}&&\f@tp
114 \newcommand*{\f@tsimpleline}{%
115    \f@tpsg{}\f@tchartstrut& \f@tpsg{} & \f@tpsg{} & \f@tpsg{} & \f@tpsg{} & \f@tpsg{} & \f@
116
```

\f@tchartstrut   \f@tchartstrut is a strut used in each table line. \ftablewidth is width of an
\ftablewidth   externally numbered table. \fntcolwidth is the minimum width of a column in
\fntcolwidth   an unnumbered table.

```
117 \newcommand*{\f@tchartstrut}{\lower4.5pt\vbox to14pt{}}
118 \newdimen\ftablewidth
119   \ftablewidth=\hsize
120 \newdimen\fntcolwidth
121   \setlength{\fntcolwidth}{0.08\ftablewidth}
```

\f@tcol   \f@tstartchartnonum is a table line of spaces, with no verticals.

\f@tstartchartnonum
```
122 \newcommand*{\f@tcol}{%
123   \multicolumn{1}{c}{\hspace*{\fntcolwidth}}}
124 \newcommand*{\f@tstartchartnonum}{%
125   \f@tcol &\f@tcol &\f@tcol &\f@tcol &\f@tcol &\f@tcol &\f@tcol &\f@tcol}
126
```

\ftable   \ftable sets a complete character table. The actual code is in either \f@tftablenum
\f@tftablenum   or \f@tftablenonum for externally numbered or plain tables, respectively.

\f@tftablenonum
```
127 \newcommand*{\f@tftablenum}{$$\global\f@tn=\z@
128   \halign to\ftablewidth\bgroup
129     \f@tchartstrut##\tabskip0pt plus10pt&
130     &\hfil##\hfil&\vrule##\cr
131   \lower6.5pt\null
132   &&&\f@toct0&&\f@toct1&&\f@toct2&&\f@toct3&&\f@toct4&&\f@toct5&&\f@toct6&&\f@toct7&%
133   \f@tevenline}
134 \newcommand*{\f@tftablenonum}{%
135   \global\f@tn=\z@
136   \begin{tabular}{|c|c|c|c|c|c|c|c|}
137     \f@tstartchartnonum
138     \f@tevenlinenonum
139   \end{tabular}}
140 \newcommand*{\ftable}{\ifhexoct\f@tftablenum\else\f@tftablenonum\fi}
141
```

\f@tendchart   \f@tendchart sets the last line of an externally numbered table with the relevant
hex digits.
```
142 \newcommand*{\f@tendchart}{\cr\noalign{\hrule}
143   \raise11.5pt\null&&&\f@thex 8&&\f@thex 9&&\f@thex A&&\f@thex B&
144   &\f@thex C&&\f@thex D&&\f@thex E&&\f@thex F&\cr
145   \egroup$$\par}
146
```

\f@tpsg   \f@tpsg typesets a single glyph, possibly with its decimal slot number. \f@placechar
\f@placechar   is the function to typeset the glyph with its number that is internally defined as
\f@placedecimal   \f@placedecimal if decimals are to be shown.
```
147 \newcommand*{\f@tpsg}{%
148   \setbox\z@=\hbox{\f@placechar{\char\f@tn}{\the\f@tn}}%
149   \ifdim\ht\z@>7.5pt\relax
150     \f@treposition
151   \else
152     \ifdim\dp\z@>2.5pt\relax
153       \f@treposition
```

```
154     \fi
155   \fi
156   \box\z@
157   \global\advance\f@tn\@ne
158 }
```

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

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

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

```
160 \newcommand*{\nodecimals}{%
161   \renewcommand*\f@placechar{\@firstoftwo}%
162 }
```

```
163 \newcommand{\decimals}{%
164   \renewcommand*\f@placechar{\f@placedecimal}%
165 }
166 \newcommand*\f@placechar{}
167 \decimals
```

\f@treposition   \f@treposition

```
168 \newcommand*{\f@treposition}{\setbox0=\vbox{\kern2pt\box0}\f@tdim=\dp0
169   \advance\f@tdim 2pt \dp0=\f@tdim}
170
```

\fonttext   \fonttext{⟨font⟩} typesets \knutext using ⟨font⟩ (e.g. auncl10).

```
171 \def\fonttext#1{%
172   \def\f@tfontname{#1}%
173   \bgroup
174   \f@tstartfont
175   \knutext
176   \egroup}
177
```

\regulartext   \regulartext{⟨fontspec⟩} typesets \knutext using ⟨fontspec⟩ (e.g., \aunclfamily).

```
178 \def\regulartext#1{%
179   \bgroup
180   #1
181   \knutext
182   \egroup}
183
```

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

```
184 \def\knutext{{
185 On November 14, 1885, Senator \& Mrs.~Leland Stanford called together
186 at their San Francisco mansion the 24~prominent men who had been
187 chosen as the first trustees of The Leland Stanford Junior University.
188 They handed to the board the Founding Grant of the University, which
```

```
189 they had executed three days before. This document---with various
190 amendments, legislative acts, and court decrees---remains as the
191 University's charter.  In bold, sweeping language it stipulates that
192 the objectives of the University are ``to qualify students for
193 personal success and direct usefulness in life; and to promote the
194 publick welfare by exercising an influence in behalf of humanity and
195 civilization, teaching the blessings of liberty regulated by law, and
196 inculcating love and reverence for the great principles of government
197 as derived from the inalienable rights of man to life, liberty, and
198 the pursuit of happiness.''
199
200 \moreknutext
201
202 \capknutext
203
204 \knunames
205 \par}}
206
```

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

```
207 \def\@moreknutext{?`But aren't Kafka's Schlo{\ss} and {\AE}sop's
208 {\OE}uvres often na{\"\i}ve vis-\`a-vis the d{\ae}monic ph{\oe}nix's
209 official r\^ole in fluffy souffl\'es? }
210
```

\@capknutext   Text using only capital letters and some punctutation.
\capknutext
```
211 \newcommand{\@capknutext}{%
212 (!`THE DAZED BROWN FOX QUICKLY GAVE 12345--67890 JUMPS!)}
213 \let\capknutext\@capknutext
214
```

\@knunames   Lots of accents masquerading in personal names.

```
215 \def\@knunames{ {\AA}ngel\aa\ Beatrice Claire
216   Diana \'Erica Fran\c{c}oise Ginette H\'el\'ene Iris
217   Jackie K\=aren {\L}au\.ra Mar{\'\i}a N\H{a}ta{\l}{\u\i}e {\O}ctave
218   Pauline Qu\^eneau Roxanne Sabine T\~a{\'\j}a Ur\v{s}ula
219   Vivian Wendy Xanthippe Yv{\o}nne Z\"azilie\par}
220
```

\guillemotleft   Just in case the french quotes are not defined, as they are called for in the subse-
\guillemotright   quent \germantext.
\flqq
```
221 \DeclareTextSymbol{\guillemotleft}{OT1}{`\'}
```
\frqq
```
222 \DeclareTextSymbol{\guillemotright}{OT1}{`\`}
223 \providecommand{\flqq}{\guillemotleft}
224 \providecommand{\frqq}{\guillemotright}
225
```

\germantext   Text from the Blindtext package.
\germanparatext
```
226 \providecommand*{\germantext}{%
```

```
227    \PackageWarning{fonttable}{\protect\germantext\space is deprecated,
228                      \MessageBreak use \protect\germanparatext\space instead}}
229 \newcommand*{\germanparatext}{%
230 Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer
231 diesen Text liest, ist selbst schuld. Der Text gibt lediglich den
232 Grauwert der Schrift an. Ist das wirklich so? Ist es
233 gleich\-g\"ul\-tig ob ich schreibe: \frqq Dies ist ein
234 Blindtext\flqq\ oder \frqq Huardest gefburn\flqq? Kjift --
235 mitnichten! Ein Blindtext bietet mir wichtige Informationen. An
236 ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie
237 harmonisch die Figuren zueinander stehen und pr\"u\-fe, wie breit
238 oder schmal sie l\"auft. Ein Blindtext sollte m\"og\-lichst viele
239 verschiedene Buchstaben enthalten und in der Originalsprache
240 gesetzt sein. Er mu\ss\ keinen Sinn ergeben, sollte aber lesbar
241 sein. Fremdsprachige Texte wie \frqq Lorem ipsum\flqq\ dienen
242 nicht dem eigentlichen Zweck, da sie eine
243 falsche Anmutung vermitteln.\par}
244
```

\latintext   The traditional printers' text.

\latinparatext
```
245 \providecommand*{\latintext}{%
246    \PackageWarning{fonttable}{\protect\latintext\space may be overriden by the
247     babel package \MessageBreak use
248                      \protect\latinparatext\space instead}}
249 \newcommand*{\latinparatext}{%
250 Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam
251 lobortis facilisis sem. Nullam nec mi et neque pharetra
252 sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper,
253 felis non sodales commodo, lectus velit ultrices augue, a
254 dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie
255 ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in
256 sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
257 Duis fringilla tristique neque. Sed interdum libero ut metus.
258 Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit
259 amet ante lobortis sollicitudin. Praesent blandit blandit mauris.
260 Praesent lectus tellus, aliquet aliquam, luctus a, egestas a,
261 turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum
262 turpis accumsan semper.\par}
263
```

\simpletext   \simpletext kills off \moreknutext and \knunames. \fulltext restores \moreknutext
\fulltext   and \knunames. Make \fulltext the default.

\moreknutext
\knunames
```
264 \newcommand*{\simpletext}{\let\moreknutext\relax \let\knunames\relax}
265 \newcommand*{\fulltext}{\let\moreknutext\@moreknutext \let\knunames\@knunames}
266 \fulltext
267
```

fonttexts   \fonttexts{⟨font⟩}{⟨text⟩} typesets ⟨text⟩ using ⟨font⟩ (e.g. auncl10).

```
268 \def\fonttexts#1#2{%
```

```
269    \def\f@tfontname{#1}%
270    \bgroup
271    \f@tstartfont
272    #2
273    \egroup}
274
```

\regulartexts   \regulartext{⟨*fontspec*⟩}{⟨*text*⟩} typesets ⟨*text*⟩ using ⟨*fontspec*⟩ (e.g., \aunclfamily).

```
275 \def\regulartexts#1#2{%
276    \bgroup
277    #1 #2
278    \egroup}
279
```

\aztext   The various characters used for Latin texts.
\AZtext
\digitstext
\punctext

```
280 \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}
281 \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}
282 \newcommand*{\digitstext}{0 1 2 3 4 5 6 7 8 9}
283 \newcommand*{\punctext}{' ! @ \$ \& * ( ) \_ - + =  [ ] < > \{ \} : ; ' , . ? /}
284
```

3.2   Testing a glyph

This is a reimplementation 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.
\f@ttwodigits

```
285 %%%%%%% using \@tempcnta for Knuth's \m and \@tempcntb for his \n
286 \newcommand*{\fnthours}{\@tempcntb=\time \divide\@tempcntb 60
287    \@tempcnta=-\@tempcntb \multiply\@tempcnta 60 \advance\@tempcnta \time
288    \f@ttwodigits\@tempcntb:\f@ttwodigits\@tempcnta}
289 \newcommand*{\f@ttwodigits}[1]{\ifnum #1<10 0\fi \number#1}
290
```

\f@tgettsechars   \f@tgettsechars{⟨*T*⟩}{⟨*S*⟩}{⟨*E*⟩} gets three characters and \chardefs \f@ttchar
\f@ttchar   to ⟨*T*⟩ (the test character), \f@tschar to ⟨*S*⟩ (start character) and \f@techar to
\f@tschar   ⟨*E*⟩ (the end character).
\f@techar

```
291 \newcommand*{\f@tgettsechars}[3]{%
292    \chardef\f@ttchar=#1 \chardef\f@tschar=#2 \chardef\f@techar=#3}
293
```

\glyphmixture   \glyphmixture{⟨*T*⟩}{⟨*S*⟩}{⟨*E*⟩} sets a mix of ⟨*T*⟩ within the glyph range from
\f@tmixpattern   ⟨*S*⟩ to ⟨*E*⟩ according to the pattern \f@tmixpattern.   The work is done by
\f@tdomix   \f@tdomix.

```
294 \newcommand*{\glyphmixture}[3]{\f@tgettsechars{#1}{#2}{#3}%
295                               \f@tdomix\f@tmixpattern}
296 \newcommand*{\f@tmixpattern}{\0\1\0\0\1\1\0\0\1\1\1\0\1}
```

```
297 \newcommand*{\f@tdomix}[1]{\par\chardef\0=\f@ttchar \@tempcntb=\f@tschar
298     \loop \chardef\1=\@tempcntb #1\endgraf
299     \ifnum \@tempcntb<\f@techar \advance\@tempcntb \@ne \repeat}
300
```

\glyphalternation   These are similar to \glyphmixture and \f@tmixpattern except that the glyphs
\f@taltpattern   are alternated.

```
301 \newcommand*{\glyphalternation}[3]{\f@tgettsechars{#1}{#2}{#3}%
302                                     \f@tdomix\f@taltpattern}
303 \newcommand*{\f@taltpattern}{\0\1\0\1\0\1\0\1\0\1\0\1\0\1\0\1\0}
304
```

\f@tdisc   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.

```
305 \newcommand*{\f@tdisc}{\discretionary{\f@ttchar}{\f@ttchar}{\f@ttchar}}
306
```

\glyphseries   \glyphseries{⟨T⟩}{⟨S⟩}{⟨E⟩} puts the test character ⟨T⟩ between all the others
\f@tdoseries   in the range ⟨S⟩ to ⟨E⟩. The work is done by \f@tdoseries.

```
307 \newcommand*{\glyphseries}[3]{\f@tgettsechars{#1}{#2}{#3}%
308     \f@tdisc\f@tdoseries\f@tschar\f@techar\par}
309 \newcommand*{\f@tdoseries}[2]{\@tempcntb=#1\relax
310     \loop\char\@tempcntb\f@tdisc
311         \ifnum\@tempcntb<#2\advance\@tempcntb \@ne \repeat}
312
```

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

```
313 \newcommand*{\glyphalphabet}{\f@tcomplower}
314 \newcommand*{\GLYPHALPHABET}{\f@tcompupper}
315 \newcommand*{\f@tcomplower}[1]{\chardef\f@ttchar=#1
316     \f@tdisc\f@tdoseries{`a}{`z}\f@tdoseries{31}{34}\par}
317 \newcommand*{\f@tcompupper}[1]{\chardef\f@ttchar=#1
318     \f@tdisc\f@tdoseries{`A}{`Z}\f@tdoseries{35}{37}\par}
319
```

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

```
320 \newcommand*{\glyphlowers}{\f@tdocomprehensive\f@tclc{`a}{`z}{31}{34}}
321 \newcommand*{\glyphuppers}{\f@tdocomprehensive\f@tcuc{`A}{`Z}{35}{37}}
322 \newcommand*{\glyphdigits}{\f@tdocomprehensive\f@tdgs{`0}{`4}{`5}{`9}}
323 \newcommand*{\f@tdocomprehensive}[5]{\par\chardef\f@ttchar=#2
324     \loop{#1} \ifnum\f@ttchar<#3\@tempcnta=\f@ttchar\advance\@tempcnta \@ne
325     \chardef\f@ttchar=\@tempcnta \repeat
326     \chardef\f@ttchar=#4
```

```
327    \loop{#1} \ifnum\f@ttchar<#5\@tempcnta=\f@ttchar\advance\@tempcnta \@ne
328    \chardef\f@ttchar=\@tempcnta \repeat}
329 \newcommand*{\f@tclc}{\f@tdisc\f@tdoseries{'a}{'z}\f@tdoseries{31}{34}\par}
330 \newcommand*{\f@tcuc}{\f@tdisc\f@tdoseries{'A}{'Z}\f@tdoseries{35}{37}\par}
331 \newcommand*{\f@tdgs}{\f@tdisc\f@tdoseries{'0}{'9}\par}
332
```

`\glyphpunct`
`\f@tdopunct`

\glyphpunct sets punctuation marks in combination with different sorts of letters. The work is done by \f@tdopunct.

```
333 \newcommand*{\glyphpunct}{\par\f@tdopunct{min}\f@tdopunct{pig}\f@tdopunct{hid}
334                    \f@tdopunct{HIE}\f@tdopunct{TIP}\f@tdopunct{fluff}
335    \$1,234.56 + 7/8 = 9\% @ \#0\par}
336 \newcommand*{\f@tdopunct}[1]{#1,\ #1:\ #1;\ '#1'\
337    ?'#1?\ !'#1!\ (#1)\ [#1]\ #1*\ #1.\par}
338
```

The end of the package.

```
339 ⟨/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.