$\LaTeX\,2_\varepsilon$ font selection

© Copyright 1995, $\LaTeX$3 Project Team.
All rights reserved.

12 December 1995

Contents

1 Introduction

This document describes the new font selection features of the LaTeX Document Preparation System. It is intended for package writers who want to write font-loading packages similar to `times` or `latexsym`.

This document is only a brief introduction to the new facilities and is intended for package writers who are familiar with TeX fonts and LaTeX packages. It is *neither* a user-guide *nor* a reference manual for fonts in LaTeX 2_ε.

1.1 LaTeX 2_ε fonts

The most important difference between LaTeX 2.09 and LaTeX 2_ε is the way that fonts are selected. In LaTeX 2.09, the Computer Modern fonts were built into the LaTeX format, and so customizing LaTeX to use other fonts was a major effort.

In LaTeX 2_ε, very few fonts are built into the format, and there are commands to load new text and math fonts. Packages such as `times` or `latexsym` allow authors to access these fonts. This document describes how to write similar font-loading packages.

The LaTeX 2_ε font selection system was first released as the 'New Font Selection Scheme' (NFSS) in 1989, and then in release 2 in 1993. LaTeX 2_ε includes NFSS release 2 as standard.

1.2 Overview

This document contains an overview of the new font commands of LaTeX.

Section 2 describes the commands for selecting fonts in classes and packages. It lists the five LaTeX font attributes, and lists the commands for selecting fonts. It also describes how to customize the author commands such as `\textrm` and `\textit` to suit your document design.

Section 3 explains the commands for controlling LaTeX math fonts. It describes how to specify new math fonts and new math symbols.

Section 4 explains how to install new fonts into LaTeX. It shows how LaTeX font attributes are turned into TeX font names, and how to specify your own fonts using font definition files.

Section 5 discusses text font encodings. It describes how to declare a new encoding and how to define commands, such as `\AE` or `\"`, which have different definitions in different encodings, depending on whether ligatures, etc. are available in the encoding.

Section 6 covers font miscellanea. It describes how LaTeX performs font substitution, how to customize fonts that are preloaded in the LaTeX format, and the naming conventions used in LaTeX font selection.

1.3 Further information

For a general introduction to LaTeX, including the new features of LaTeX 2_ε, you should read *LaTeX: A Document Preparation System*, Leslie Lamport, Addison Wesley, 2nd ed, 1994.

A more detailed description of the LaTeX font selection scheme is to be found in *The LaTeX Companion*, Goossens, Mittelbach and Samarin, Addison Wesley, 1994.

The LaTeX font selection scheme is based on TeX, which is described by its developer in *The TeXbook*, Donald E. Knuth, Addison Wesley, 1986, revised in 1991 to include the features of TeX 3.

Sebastian Rahtz's `psnfss` software contains the software for using a large number of Type 1 fonts (including the Adobe Laser Writer 35 and the Monotype CD-ROM fonts) in LaTeX. It should be available from the same source as your copy of LaTeX.

The `psnfss` software uses fonts generated by Alan Jeffrey's `fontinst` software. This can convert fonts from Adobe Font Metric format into a format readable by LaTeX, including the generation of the `.fd` files described in Section 4. The `fontinst` software should be available from the same source as your copy of LaTeX.

Whenever practical, LaTeX uses the font naming scheme described in *Filenames for fonts*, Karl Berry, *TUGboat* 11(4), 1990.

The class-writer's guide *LaTeX 2_ε for Class and Package Writers* describes the new LaTeX features for writers of document classes and packages and is kept in `clsguide.tex`. Configuring LaTeX is covered by the guide *Configuration options for LaTeX 2_ε* in `cfgguide.tex` whilst the philosophy behind our policy on modifying LaTeX is described in *Modifying LaTeX* in `modguide.tex`.

We are gradually turning the source code for LaTeX into a LaTeX document *LaTeX: the program*. This document includes an index of LaTeX commands and can be typeset from `source2e.tex`.

For more information about TeX and LaTeX, please contact your local TeX Users Group, or the international TeX Users Group. Here are some useful addresses.

> TeX Users Group, 1850 Union Street #1637, San Francisco, CA 94123, USA
> Fax: +1 415 982 8559 Email: `tug@tug.org`
>
> UK TUG, 1 Eymore Close, Selly Oak, Birmingham B29 4LB, UK
> Fax: +44 121 476 2159 Email: `uktug-enquiries@tex.ac.uk`

2 Text fonts

This section describes the commands available to class and package writers for specifying and selecting fonts.

2.1 Text font attributes

Every text font in LaTeX has five *attributes*:

encoding This specifies the order that characters appear in the font. The two most common text encodings used in LaTeX are Knuth's 'TeX text' encoding, and the 'TeX text extended' encoding developed by the TeX Users Group members during a TeX Conference at Cork in 1990 (hence its informal name 'Cork encoding').

family The name for a collection of fonts, usually grouped under a common name by the font foundry. For example, 'Adobe Times', 'ITC Garamond', and Knuth's 'Computer Modern Roman' are all font families.

series How heavy or expanded a font is. For example, 'medium weight', 'narrow' and 'bold extended' are all series.

shape The form of the letters within a font family. For example, 'italic', 'oblique' and 'upright' (sometimes called 'roman') are all font shapes.

size The design size of the font, for example '10pt'.

The possible values for these attributes are given short acronyms by LaTeX. The most common values for the font encoding are:

`OT1`	TeX text
`T1`	TeX extended text
`OML`	TeX math italic
`OMS`	TeX math symbols
`OMX`	TeX math large symbols
`U`	Unknown
`L`$\langle xx \rangle$	A local encoding

The 'local' encodings are intended for font encodings which are only locally available, for example a font containing an organisation's logo in various sizes.

There are far too many font families to list them all, but some common ones are:

`cmr`	Computer Modern Roman
`cmss`	Computer Modern Sans
`cmtt`	Computer Modern Typewriter
`cmm`	Computer Modern Math Italic
`cmsy`	Computer Modern Math Symbols
`cmex`	Computer Modern Math Extensions
`ptm`	Adobe Times
`phv`	Adobe Helvetica
`pcr`	Adobe Courier

The most common values for the font series are:

`m`	Medium
`b`	Bold
`bx`	Bold extended
`sb`	Semi-bold
`c`	Condensed

The most common values for the font shape are:

`n`	Normal (that is 'upright' or 'roman')
`it`	Italic
`sl`	Slanted (or 'oblique')
`sc`	Caps and small caps

The font size is specified as a dimension, for example `10pt` or `1.5in` or `3mm`. These five parameters specify every LaTeX font, for example:

LaTeX specification	*Font*	*TeX font name*
`OT1 cmr m n 10pt`	Computer Modern Roman 10pt	`cmr10`
`OT1 cmss m sl 12pt`	Computer Modern Sans Oblique 12pt	`cmssi12`
`OML cmm m it 10pt`	Computer Modern Math Italic 10pt	`cmmi10`
`T1 ptm b it 18pt`	Adobe Times Bold Italic 18pt	`ptmbq at 18pt`

These five parameters are displayed whenever LaTeX gives an overfull box warning, for example:

```
Overfull \hbox (3.80855pt too wide) in paragraph at lines 314--318
[]\OT1/cmr/m/n/10 Normally [] and [] will be iden-ti-cal,
```

The author commands for fonts set the five attributes:

Author command	*Attribute*	*Value in* `article` *class*
`\textrm{..}` or `\rmfamily`	family	`cmr`
`\textsf{..}` or `\sffamily`	family	`cmss`
`\texttt{..}` or `\ttfamily`	family	`cmtt`
`\textmd{..}` or `\mdseries`	series	`m`
`\textbf{..}` or `\bfseries`	series	`bx`
`\textup{..}` or `\upshape`	shape	`n`
`\textit{..}` or `\itshape`	shape	`it`
`\textsl{..}` or `\slshape`	shape	`sl`
`\textsc{..}` or `\scshape`	shape	`sc`
`\tiny`	size	`5pt`
`\scriptsize`	size	`7pt`
`\footnotesize`	size	`8pt`
`\small`	size	`9pt`
`\normalsize`	size	`10pt`
`\large`	size	`12pt`
`\Large`	size	`14.4pt`
`\LARGE`	size	`17.28pt`
`\huge`	size	`20.74pt`
`\Huge`	size	`24.88pt`

The values used by these commands are determined by the document class, using the parameters defined in Section 2.4.

Note that there are no author commands for selecting new encodings. These should be provided by packages, such as the `fontenc` package.

This section does not explain how LaTeX font specifications are turned into TeX font names. This is described in Section 4.

2.2 Selection commands

The low-level commands used to select a text font are as follows.

`\fontencoding` {⟨*encoding*⟩}
`\fontfamily` {⟨*family*⟩}
`\fontseries` {⟨*series*⟩}
`\fontshape` {⟨*shape*⟩}
`\fontsize` {⟨*size*⟩} {⟨*baselineskip*⟩}

Each of these commands sets one of the font attributes; `\fontsize` also sets `\baselineskip`. The actual font in use is not altered by these commands, but the current attributes are used to determine which font to use after the next `\selectfont` command.

`\selectfont`

Selects a text font, based on the current values of the font attributes.

Warning: there *must* be a `\selectfont` command immediately after any settings of the font parameters by (some of) the five `\font`⟨*parameter*⟩ commands, before any following text. For example, it is legal to say:

```
\fontfamily{ptm}\fontseries{b}\selectfont Some text.
```

but it is *not* legal to say:

```
\fontfamily{ptm} Some \fontseries{b}\selectfont text.
```

You may get unexpected results if you put text between a `\font`⟨*parameter*⟩ command and a `\selectfont`.

`\usefont` {⟨*encoding*⟩} {⟨*family*⟩} {⟨*series*⟩} {⟨*shape*⟩}

A short hand for the equivalent `\font`... commands followed by a call to `\selectfont`.

2.3 Internals

The current values of the font attributes are held in internal macros.

```
\f@encoding
\f@family
\f@series
\f@shape
\f@size
\f@baselineskip
\tf@size
\sf@size
\ssf@size
```

These hold the current values of the encoding, the family, the series, the shape, the size, the baseline skip, the main math size, the 'script' math size and the 'scriptscript' math size. The last three are accessible only within a formula; outside of math they may contain arbitrary values.

For example, to set the size to 12 without changing the baseline skip:

```
\fontsize{12}{\f@baselineskip}
```

However, you should *never* alter the values of the internal commands directly; they must only be modified using the low-level commands like `\fontfamily`, `\fontseries`, etc. If you disobey this warning you might produce code that loops.

2.4 Parameters for author commands

The parameter values set by author commands such as `\textrm` and `\rmfamily`, etc. are not hard-wired into LaTeX; instead these commands use the values of a number of parameters set by the document class and packages. For example, `\rmdefault` is the name of the default family selected by `\textrm` and `\rmfamily`. Thus to set a document in Adobe Times, Helvetica and Courier, the document designer specifies:

```
\renewcommand{\rmdefault}{ptm}
\renewcommand{\sfdefault}{phv}
\renewcommand{\ttdefault}{pcr}
```

```
\encodingdefault
\familydefault
\seriesdefault
\shapedefault
```

The encoding, family, series and shape of the main body font. By default these are `OT1`, `\rmdefault`, `m` and `n`. Note that since the default family is `\rmdefault`, this means that changing `\rmdefault` will change the main body font of the document.

```
\rmdefault
\sfdefault
\ttdefault
```

The families selected by `\textrm`, `\rmfamily`, `\textsf`, `\sffamily`, `\texttt` and `\ttfamily`. By default these are `cmr`, `cmss` and `cmtt`.

```
\bfdefault
\mddefault
```

The series selected by `\textbf`, `\bfseries`, `\textmd` and `\mdseries`. By default these are `bx` and `m`. These values are suitable for the default families used. If other fonts are used as standard document fonts (for example, certain PostScript fonts) it might be necessary to adjust the value of `\bfdefault` to `b` since only a few such families have a 'bold extended' series. An alternative (taken for the fonts provided by `psnfss`) is to define silent substitutions from `bx` series to `b` series with special `\DeclareFontShape` declarations and the `ssub` size function, see Section 4.4.

```
\itdefault
\sldefault
\scdefault
\updefault
```

The shapes selected by `\textit`, `\itshape`, `\textsl`, `\slshape`, `\textsc`, `\scshape`, `\textup` and `\upshape`. By default these are `it`, `sl`, `sc` and `n`.

Note that there are no parameters for the size commands. These should be defined directly in class files, for example:

```
\renewcommand{\normalsize}{\fontsize{10}{12}\selectfont}
```

More elaborate examples (setting additional parameters when the text size is changed) can be found in `classes.dtx` the source documentation for the classes `article`, `report`, and `book`.

2.5 Special font declaration commands

`\DeclareFixedFont` {⟨*cmd*⟩} {⟨*ENC*⟩} {⟨*family*⟩} {⟨*series*⟩} {⟨*shape*⟩} {⟨*size*⟩}

Declares command ⟨*cmd*⟩ to be a font switch which selects the font that is specified by the attributes ⟨*ENC*⟩, ⟨*family*⟩, ⟨*series*⟩, ⟨*shape*⟩, and ⟨*size*⟩.

The font is selected without any adjustments to baselineskip and other surrounding conditions.

This example makes `{\picturechar .}` select a small dot very quickly:

```
\DeclareFixedFont{\picturechar}{OT1}{cmr}{m}{n}{5}
```

`\DeclareTextFontCommand {`$\langle$*cmd*$\rangle$`} {`$\langle$*font-switches*$\rangle$`}`

Declares command $\langle$*cmd*$\rangle$ to be a font command with one argument. The current font attributes are locally modified by $\langle$*font-switches*$\rangle$ and then the argument of $\langle$*cmd*$\rangle$ is typeset in the resulting new font.

Commands defined by `\DeclareTextFontCommand` automatically take care of any necessary italic correction (on either side).

The following example shows how `\textrm` is defined by the kernel.

```
\DeclareTextFontCommand{\textrm}{\rmfamily}
```

To define a command that always typeset its argument in the italic shape of the main document font you could declare:

```
\DeclareTextFontCommand{\normalit}{\normalfont\itshape}
```

This declaration can be used to change the meaning of a command; if $\langle$*cmd*$\rangle$ is already defined, a log that it has been redefined is put in the transcript file.

`\DeclareOldFontCommand {`$\langle$*cmd*$\rangle$`} {`$\langle$*text-switch*$\rangle$`} {`$\langle$*math-switch*$\rangle$`}`

Declares command $\langle$*cmd*$\rangle$ to be a font switch (i.e. used with the syntax `{`$\langle$*cmd*$\rangle$`...}`) having the definition $\langle$*text-switch*$\rangle$ when used in text and the definition $\langle$*math-switch*$\rangle$ when used in a formula. Math alphabet commands, like `\mathit`, when used within $\langle$*math-switch*$\rangle$ should not have an argument. Their use in this argument causes their semantics to change so that they here act as a font switch, as required by the usage of the $\langle$*cmd*$\rangle$.

This declaration is useful for setting up commands like `\rm` to behave as they did in LaTeX 2.09. We strongly urge you *not* to misuse this declaration to invent new font commands.

The following example defines `\it` to produce the italic shape of the main document font if used in text and to switch to the font that would normally be produced by the math alphabet `\mathit` if used in a formula.

```
\DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
```

This declaration can be used to change the meaning of a command; if $\langle$*cmd*$\rangle$ is already defined, a log that it has been redefined is put in the transcript file.

3 Math fonts

This section describes the commands available to class and package writers for specifying math fonts and math commands.

3.1 Math font attributes

The selection of fonts within math mode is quite different to that of text fonts.

Some math fonts are selected explicitly by one-argument commands such as `\mathsf{max}` or `\mathbf{vec}`: these are called *math alphabets*. These math alphabet commands affect only the font used for letters and symbols of type `\mathalpha` (see Section 3.6); other symbols within the argument will be left unchanged. The predefined math alphabets are:

Alphabet	*Description*	*Example*
`\mathnormal`	default	$abcXYZ$
`\mathrm`	roman	abcXYZ
`\mathbf`	bold roman	$\mathbf{abcXYZ}$
`\mathsf`	sans serif	abcXYZ
`\mathit`	text italic	abcXYZ
`\mathtt`	typewriter	$\mathtt{abcXYZ}$
`\mathcal`	calligraphic	$\mathcal{XYZ}$

Other math fonts are selected implicitly by TeX for symbols, with commands such as `\oplus` (producing $\oplus$) or with straight characters like `>` or `+`. Fonts containing such math symbols are called *math symbol fonts*. The predefined math symbol fonts are:

Symbol font	*Description*	*Example*
`operators`	symbols from `\mathrm`	$[+]$
`letters`	symbols from `\mathnormal`	$<< \star >>$
`symbols`	most LaTeX symbols	$\leq * \geq$
`largesymbols`	large symbols	$\sum\prod\int$

Some math fonts are both *math alphabets* and *math symbol fonts*, for example `\mathrm` and `operators` are the same font, and `\mathnormal` and `letters` are the same font.

Math fonts in LaTeX have the same five attributes as text fonts: encoding, family, series, shape and size. However, there are no commands that allow the attributes to be individually changed. Instead, the conversion from math fonts to these five attributes is controlled by the *math version*. For example, the `normal` math version maps:

Math font		*External font*
Alphabets	*Symbol fonts*	*Attributes*
`\mathnormal`	`letters`	`OML cmm m it`
`\mathrm`	`operators`	`OT1 cmr m n`
`\mathcal`	`symbols`	`OMS cmsy m n`
	`largesymbols`	`OMX cmex m n`
`\mathbf`		`OT1 cmr bx n`
`\mathsf`		`OT1 cmss m n`
`\mathit`		`OT1 cmr m it`
`\mathtt`		`OT1 cmtt m n`

The `bold` math version is similar except that it contains bold fonts. The command `\boldmath` selects the `bold` math version.

Math versions can only be changed outside of math mode.

The two predefined math versions are:

`normal`	the default math version
`bold`	the bold math version

Packages may define new math alphabets, math symbol fonts, and math versions. This section describes the commands for writing such packages.

3.2 Selection commands

There are no commands for selecting symbol fonts. Instead, these are selected indirectly through symbol commands like `\oplus`. Section 3.6 explains how to define symbol commands.

```
\mathnormal{⟨math⟩}
\mathcal{⟨math⟩}
\mathrm{⟨math⟩}
\mathbf{⟨math⟩}
\mathsf{⟨math⟩}
\mathit{⟨math⟩}
\mathtt{⟨math⟩}
```

Each math alphabet is a command which can only be used inside math mode. For example, `$x + \mathsf{y} + \mathcal{Z}$` produces $x + \mathsf{y} + \mathcal{Z}$.

```
\mathversion{⟨version-name⟩}
```

This command selects a math version; it can only be used outside math mode. For example, `\boldmath` is defined to be `\mathversion{bold}`.

3.3 Declaring math versions

```
\DeclareMathVersion {⟨version-name⟩}
```

Defines ⟨*version-name*⟩ to be a math version.

The newly declared version is initialised with the defaults for all symbol fonts and math alphabets declared so far (see the commands `\DeclareSymbolFont` and `\DeclareMathAlphabet`).

If used on an already existing version, an information message is written to the transcript file and all previous `\SetSymbolFont` or `\SetMathAlphabet` declarations for this version are overwritten by the math alphabet and symbol font defaults, i.e. one ends up with a virgin math version.

Example:

```
\DeclareMathVersion{normal}
```

3.4 Declaring math alphabets

`\DeclareMathAlphabet {`⟨*math-alph*⟩`} {`⟨*encoding*⟩`} {`⟨*family*⟩`} {`⟨*series*⟩`} {`⟨*shape*⟩`}`

Defines ⟨*math-alph*⟩ to be a new math alphabet.

The arguments ⟨*encoding*⟩ ⟨*family*⟩ ⟨*series*⟩ ⟨*shape*⟩ are the default values for this math alphabet in all math versions; these can be reset later for a particular math version by a `\SetMathAlphabet` command. If ⟨*shape*⟩ is empty then the ⟨*math-alph*⟩ is declared to be invalid in all versions, unless it is set by a later `\SetMathAlphabet` command.

Checks that ⟨*math-alph*⟩ can be used and that ⟨*encoding*⟩ is a valid encoding scheme.

In these examples, `\foo` is defined everywhere but `\baz`, by default, is defined nowhere.

```
\DeclareMathAlphabet{\foo}{OT1}{cmtt}{m}{n}
\DeclareMathAlphabet{\baz}{OT1}{}{}{}
```

`\SetMathAlphabet {`⟨*math-alph*⟩`} {`⟨*version-name*⟩`}`
`{`⟨*encoding*⟩`} {`⟨*family*⟩`} {`⟨*series*⟩`} {`⟨*shape*⟩`}`

Changes, or sets, the font for the math alphabet ⟨*math-alph*⟩ in math version ⟨*version-name*⟩ to ⟨*encoding*⟩⟨*family*⟩⟨*series*⟩⟨*shape*⟩.

Checks that ⟨*math-alph*⟩ is a math alphabet, ⟨*version-name*⟩ is a math version and ⟨*encoding*⟩ is a known encoding scheme.

This example defines `\baz` for the 'normal' math version only:

```
\SetMathAlphabet\baz{normal}{OT1}{cmss}{m}{n}
```

Note that this declaration is not used for all math alphabets: Section 3.5 describes `\DeclareSymbolFontAlphabet`, which is used to set up math alphabets contained in fonts which have are declared as symbol fonts.

3.5 Declaring symbol fonts

`\DeclareSymbolFont {`⟨*sym-font-name*⟩`} {`⟨*encoding*⟩`} {`⟨*family*⟩`} {`⟨*series*⟩`} {`⟨*shape*⟩`}`

Defines ⟨*sym-font-name*⟩ to be a new symbol font.

The arguments ⟨*encoding*⟩ ⟨*family*⟩ ⟨*series*⟩ ⟨*shape*⟩ are the default values for this symbol font in all math versions; these can be reset later for a particular math version by a `\SetSymbolFont` command.

Checks that ⟨*sym-font-name*⟩ can be used and that ⟨*encoding*⟩ is a declared encoding scheme.

For example, the following sets up the first four standard math symbol fonts:

```
\DeclareSymbolFont{operators}{OT1}{cmr}{m}{n}
\DeclareSymbolFont{letters}{OML}{cmm}{m}{it}
\DeclareSymbolFont{symbols}{OMS}{cmsy}{m}{n}
\DeclareSymbolFont{largesymbols}{OMX}{cmex}{m}{n}
```

`\SetSymbolFont` {⟨*sym-font-name*⟩} {⟨*version name*⟩}
{⟨*encoding*⟩} {⟨*family*⟩} {⟨*series*⟩} {⟨*shape*⟩}

Changes the symbol font ⟨*sym-font-name*⟩ for math version ⟨*version name*⟩ to ⟨*encoding*⟩ ⟨*family*⟩ ⟨*series*⟩ ⟨*shape*⟩.

Checks that ⟨*sym-font-name*⟩ is a symbol font, ⟨*version name*⟩ is a known math version and ⟨*encoding*⟩ is a declared encoding scheme.

For example, the following come from the set up of the 'bold' math version:

```
\SetSymbolFont{operators}{bold}{OT1}{cmr}{bx}{n}
\SetSymbolFont{letters}{bold}{OML}{cmm}{b}{it}
```

`\DeclareSymbolFontAlphabet` {⟨*math-alph*⟩} {⟨*sym-font-name*⟩}

Allows the previously declared symbol font ⟨*sym-font-name*⟩ to be also the math alphabet ⟨*id*⟩ (in all math versions).

This declaration should be used in preference to **`\DeclareMathAlphabet`** and **`\SetMathAlphabet`** when a math alphabet is the same as a symbol font; this is because it makes better use of the limited number (only 16) of TeX's math groups.

Checks that ⟨*math-alph*⟩ can be defined and that ⟨*sym-font-name*⟩ is a symbol font.

Example:

```
\DeclareSymbolFontAlphabet{\mathrm}{operators}
\DeclareSymbolFontAlphabet{\mathcal}{symbols}
```

3.6 Declaring math symbols

`\DeclareMathSymbol` {⟨*symbol*⟩} {⟨*type*⟩} {⟨*sym-font-name*⟩} {⟨*slot*⟩}

The ⟨*symbol*⟩ can be either a single character such as '`>`', or a macro name, such as **`\sum`**.

Defines the ⟨*symbol*⟩ to be a math symbol of type ⟨*type*⟩ in slot ⟨*slot*⟩ of symbol font ⟨*sym-font-name*⟩. The ⟨*type*⟩ can be given as a number or as a command:

Type	*Meaning*	*Example*
`0` or `\mathord`	Ordinary	α
`1` or `\mathop`	Large operator	$\sum$
`2` or `\mathbin`	Binary operation	$\times$
`3` or `\mathrel`	Relation	$\leq$
`4` or `\mathopen`	Opening	$\langle$
`5` or `\mathclose`	Closing	$\rangle$
`6` or `\mathpunct`	Punctuation	;
`7` or `\mathalpha`	Alphabet character	A

Only symbols of type `\mathalpha` will be affected by math alphabet commands: within the argument of a math alphabet command they will produce the character in slot ⟨*slot*⟩ of that math alphabet's font. Symbols of other types will always produce the same symbol (within one math version).

`\DeclareMathSymbol` allows a macro ⟨*symbol*⟩ to be redefined only if it was previously defined to be a math symbol. It also checks that the ⟨*sym-font-name*⟩ is a declared symbol font.

Example:

```
\DeclareMathSymbol{\alpha}{0}{letters}{"0B}
\DeclareMathSymbol{\lessdot}{\mathbin}{AMSb}{"0C}
\DeclareMathSymbol{\alphld}{\mathalpha}{AMSb}{"0C}
```

`\DeclareMathDelimiter` {⟨*cmd*⟩} {⟨*type*⟩} {⟨*sym-font-name-1*⟩} {⟨*slot-1*⟩}
{⟨*sym-font-name-2*⟩} {⟨*slot-2*⟩}

Defines ⟨*cmd*⟩ to be a math delimiter where the small variant is in slot ⟨*slot-1*⟩ of symbol font ⟨*sym-font-name-1*⟩ and the large variant is in slot ⟨*slot-2*⟩ of symbol font ⟨*sym-font-name-2*⟩. Both symbol fonts must have been declared previously.

Checks that ⟨*sym-font-name-i*⟩ are both declared symbol fonts.

If TEX is not looking for a delimiter, ⟨*cmd*⟩ is treated just as if it had been defined with `\DeclareMathSymbol` using ⟨*type*⟩,⟨*sym-font-name-1*⟩ and ⟨*slot-1*⟩. In other words, if a command is defined as a delimiter then this automatically defines it as a math symbol.

Example:

```
\DeclareMathDelimiter{\langle}{\mathopen}{symbols}{"68}
                                         {largesymbols}{"0A}
```

`\DeclareMathAccent` {⟨*cmd*⟩} {⟨*type*⟩} {⟨*sym-font-name*⟩} {⟨*slot*⟩}

Defines ⟨*cmd*⟩ to act as a math accent.

The accent character comes from slot ⟨*slot*⟩ in ⟨*sym-font-name*⟩. The ⟨*type*⟩ can be either `\mathord` or `\mathalpha`; in the latter case the accent character changes font when used in a math alphabet.

Example:

```
\DeclareMathAccent{\acute}{\mathalpha}{operators}{"13}
\DeclareMathAccent{\vec}{\mathord}{letters}{"7E}
```

`\DeclareMathRadical {⟨cmd⟩} {⟨sym-font-name-1⟩} {⟨slot-1⟩} {⟨sym-font-name-2⟩} {⟨slot-2⟩}`

Defines ⟨*cmd*⟩ to be a radical where the small variant is in slot ⟨*slot-1*⟩ of symbol font ⟨*sym-font-name-1*⟩ and the large variant is in slot ⟨*slot-2*⟩ of symbol font ⟨*sym-font-name-2*⟩. Both symbol fonts must have been declared previously.

Example (probably the only use for it!):

```
\DeclareMathRadical{\sqrt}{symbols}{"70}{largesymbols}{"70}
```

3.7 Declaring math sizes

`\DeclareMathSizes {⟨t-size⟩} {⟨mt-size⟩} {⟨s-size⟩} {⟨ss-size⟩}`

Declares that ⟨*mt-size*⟩ is the (main) math text size, ⟨*s-size*⟩ is the 'script' size and ⟨*ss-size*⟩ the 'scriptscript' size to be used in math, when ⟨*t-size*⟩ is the current text size. For text sizes for which no such declaration is given the 'script' and 'scriptscript' size will be calculated and then fonts are loaded for the calculated sizes or the best approximation (this may result in a warning message).

Normally, ⟨*t-size*⟩ and ⟨*mt-size*⟩ will be identical; however, if, for example, PostScript text fonts are mixed with bit-map math fonts then you may not have available a ⟨*mt-size*⟩ for every ⟨*t-size*⟩.

Example:

```
\DeclareMathSizes{13.82}{12.4}{10}{7}
```

4 Font installation

This section explains how LaTeX's font attributes are turned into TeX font specifications.

4.1 Font definition files

The description of how LaTeX font attributes are turned into TeX fonts is usually kept in a *font definition* (`.fd`) file. The `.fd` file for family ⟨*family*⟩ in encoding ⟨*encoding*⟩ is called ⟨*ENC*⟩⟨*family*⟩`.fd`, for example `OT1cmr.fd` for Computer Modern Roman or `T1ptm.fd` for Adobe Times.

Whenever LaTeX encounters an encoding/family combination that it does not know (e.g. if the document designer says `\fontfamily{ptm}\selectfont`) then LaTeX attempts to load the appropriate `.fd` file. "Not known" means: there was no `\DeclareFontFamily` declaration issued for this encoding/family combination.

The declarations in the `.fd` file are responsible for telling LaTeX how to load fonts for that encoding/family combination. If the `.fd` file could not be found, a warning is issued and font substitutions are made.

4.2 Font definition file commands

The `.fd` files should contain only commands from this section. Note that those commands can also be used outside a `.fd` file: they can be put in package or class files, or even in the preamble of a document.

`\ProvidesFile{`⟨*file-name*⟩`}[`⟨*release-info*⟩`]`

The `.fd` file should announce itself with a `\ProvidesFile` command, as described in *LaTeX 2ε for Class and Package Writers*. For example:

```
\ProvidesFile{T1ptm.fd}[1994/06/01 Adobe Times font definitions]
```

`\DeclareFontFamily {`⟨*encoding*⟩`} {`⟨*family*⟩`} {`⟨*loading-settings*⟩`}`

Declares a font family ⟨*family*⟩ to be available in encoding scheme ⟨*encoding*⟩.

The ⟨*loading-settings*⟩ are executed immediately after loading any font with this encoding and family.

Checks that ⟨*encoding*⟩ was previously declared.

This example refers to the Computer Modern Typewriter font family in the Cork encoding:

```
\DeclareFontFamily{T1}{cmtt}{\hyphenchar\font=-1}
```

Each `.fd` file should contain exactly one `\DeclareFontFamily` command, and it should be for the appropriate encoding/family combination.

`\DeclareFontShape {`⟨*encoding*⟩`} {`⟨*family*⟩`} {`⟨*series*⟩`} {`⟨*shape*⟩`}` `{`⟨*loading-info*⟩`} {`⟨*loading-settings*⟩`}`

Declares a font shape combination; here ⟨*loading-info*⟩ contains the information that combines sizes with external fonts. The syntax is complex and is described in Section 4.3 below.

The ⟨*loading-settings*⟩ are executed after loading any font with this font shape. They are executed immediately after the 'loading-settings' which were declared by `\DeclareFontFamily` and so they can be used to overwrite the settings made at the family level.

Checks that the combination ⟨*encoding*⟩⟨*family*⟩ was previously declared via `\DeclareFontFamily`.

Example:

```
\DeclareFontShape{OT1}{cmr}{m}{sl}{%
          <5-8> sub * cmr/m/n
          <8> cmsl8
          <9> cmsl9
          <10> <10.95> cmsl10
          <12> <14.4> <17.28> <20.74> <24.88> cmsl12
          }{}
```

An `.fd` file can contain any number of `\DeclareFontShape` commands, which should be for the appropriate ⟨*encoding*⟩ and ⟨*family*⟩.

4.3 Font file loading information

The information which tells LaTeX exactly which font (`.tfm`) files to load is contained in the ⟨*loading-info*⟩ part of a `\DeclareFontShape` declaration. This part consists of one or more ⟨*fontshape-decl*⟩s, each of which has the following form:

⟨*fontshape-decl*⟩ ::= ⟨*size-infos*⟩ ⟨*font-info*⟩
⟨*size-infos*⟩ ::= ⟨*size-infos*⟩ ⟨*size-info*⟩ | ⟨*size-info*⟩
⟨*size-info*⟩ ::= "`<`" ⟨*number-or-range*⟩ "`>`"
⟨*font-info*⟩ ::= [⟨*size-function*⟩ "`*`"] ["`[`" ⟨*optarg*⟩ "`]`"] ⟨*fontarg*⟩

The ⟨*number-or-range*⟩ denotes the size or size-range for which this entry applies.

If it contains a hyphen it is a range: lower bound on the left (if missing, zero implied), upper bound on the right (if missing, ∞ implied). For ranges, the upper bound is *not* included in the range and the lower bound is.

Examples:

`<10>`	simple size	10pt only
`<-8>`	range	all sizes less than 8pt
`<8-14.4>`	range	all sizes greater than or equal to 8pt but less than 14.4pt
`<14.4->`	range	all sizes greater than or equal 14.4pt

If more than one ⟨*size-info*⟩ entry follows without any intervening ⟨*font-info*⟩, they all share the next ⟨*font-info*⟩.

The ⟨*size-function*⟩, if present, handles the use of ⟨*font-info*⟩. If not present, the 'empty' ⟨*size-function*⟩ is assumed.

All the ⟨*size-info*⟩s are inspected in the order in which they appear in the font shape declaration. If a ⟨*size-info*⟩ matches the requested size, its ⟨*size-function*⟩ is executed. If `\external@font` is non-empty afterwards this process stops, otherwise the next ⟨*size-info*⟩ is inspected. (See also `\DeclareSizeFunction`.)

If this process does not lead to a non-empty `\external@font`, LaTeX tries the nearest simple size. If the entry contains only ranges an error is returned.

4.4 Size functions

LaTeX provides the following size functions, whose 'inputs' are ⟨*fontarg*⟩ and ⟨*optarg*⟩ (when present).

'' (empty) Load the external font ⟨*fontarg*⟩ at the user-requested size. If ⟨*optarg*⟩ is present, it is used as the scale-factor.

s Like the empty function but without terminal warnings, only loggings.

gen Generates the external font from ⟨*fontarg*⟩ followed by the user-requested size, e.g. `<8> <9> <10> gen * cmtt`

sgen Like the 'gen' function but without terminal warnings, only loggings.

genb Generates the external font from ⟨*fontarg*⟩ followed by the user-requested size, using the conventions of the 'ec' fonts. e.g. `<10.98> genb * dctt` produces `dctt1098`. New feature 1995/12/01

sgenb Like the 'genb' function but without terminal warnings, only loggings. New feature 1995/12/01

sub Tries to load a font from a different font shape declaration given by ⟨*fontarg*⟩ in the form ⟨*family*⟩/⟨*series*⟩/⟨*shape*⟩.

ssub Silent variant of 'sub', only loggings.

subf Like the empty function but issues a warning that it has to substitute the external font ⟨*fontarg*⟩ because the desired font shape was not available in the requested size.

ssubf Silent variant of 'subf', only loggings.

fixed Load font ⟨*fontarg*⟩ as is, disregarding the user-requested size. If present, ⟨*optarg*⟩ gives the "at ... pt" size to be used.

sfixed Silent variant of 'fixed', only loggings.

Examples for the use of most of the above size functions can be found in the file `cmfonts.fdd`—the source for the standard `.fd` files describing the Computer Modern fonts by Donald Knuth.

`\DeclareSizeFunction {`⟨*name*⟩`} {`⟨*code*⟩`}`

Declares a size-function ⟨*name*⟩ for use in `\DeclareFontShape` commands. The interface is still under development but there should be no real need to a define new size functions.

The ⟨*code*⟩ is executed when the size or size-range in `\DeclareFontShape` matches the user-requested size.

The arguments of the size-function are automatically parsed and placed into `\mandatory@arg` and `\optional@arg` for use in ⟨*code*⟩. Also available, of course, is `\f@size`, which is the user-requested size.

To signal success ⟨*code*⟩ must define the command `\external@font` to contain the external name and any scaling options (if present) for the font to be loaded.

This example sets up the 'empty' size function (simplified):

```
\DeclareSizeFunction{}
        {\edef\external@font{\mandatory@arg\space at\f@size}
```

5 Encodings

This section explains how to declare and use new font encodings and how to declare commands for use with particular encodings.

5.1 The fontenc package

Users can select new font encodings using the `fontenc` package. The `fontenc` package has options for encodings; the last option becomes the default encoding. For example, to use the `OT2` (Washington University Cyrillic encoding) and `T1` encodings, with `T1` as the default, an author types:

```
\usepackage[OT2,T1]{fontenc}
```

This package loads the *encoding definition files* (⟨*ENC*⟩`enc.def` files) for each font encoding ⟨*ENC*⟩ given as an option but not already declared; it also sets `\encodingdefault` to be the last encoding in the option list.

LaTeX currently predefines the `OT1` and `T1` text encodings, and provides the files `OT1enc.def` and `T1enc.def`. Other encoding set-ups might be added to the distribution at a later stage.

Thus the example above loads the file `OT2enc.def` and sets `\encodingdefault` to `T1`.

Note: If you wish to use T1-encoded fonts other than the 'cmr' family then you may need to load the package (e.g. `times`) that selects the fonts *before* loading `fontenc` (this prevents the system from attempting to load any T1-encoded fonts from the 'cmr' family).

5.2 Encoding definition files

Warning: Some aspects of the contents of font definition files are still under development. Therefore, the current versions of the files `OT1enc.def` and `T1enc.def` are temporary versions and should not be used as models for producing further such files. For further information you should read the documentation in `ltoutenc.dtx`.

The declarations in the encoding definition file are responsible for declaring the encoding and telling LaTeX how to produce characters in this encoding.

The ⟨*ENC*⟩`enc.def` files should contain only commands from this section. As with the font definition file commands, it is also possible (although normally not necessary) to use these declarations directly within a class or package file.

`\ProvidesFile{`⟨*file-name*⟩`}[`⟨*release-info*⟩`]`

The ⟨*ENC*⟩`enc.def` file should announce itself with a `\ProvidesFile` command, described in *LaTeX2ε for Class and Package Writers*. For example:

```
\ProvidesFile{OT2enc.def}
             [1994/06/01 Washington University Cyrillic encoding]
```

`\DeclareFontEncoding {`⟨*encoding*⟩`} {`⟨*text-settings*⟩`} {`⟨*math-settings*⟩`}`

Declares a new encoding scheme ⟨*encoding*⟩.

The ⟨*text-settings*⟩ are declarations which are executed every time `\selectfont` changes the encoding to be ⟨*encoding*⟩.

The ⟨*math-settings*⟩ are similar but are for math alphabets. They are executed whenever a math alphabet with this encoding is called.

Spaces within the arguments are ignored to avoid surplus spaces in the document. If a real space is necessary use `\space`.

Example:

```
\DeclareFontEncoding{OT1}{}{}
```

Some author commands need to change their definition depending on which encoding is currently in use. For example, in the `OT1` encoding, the letter 'Æ' is in slot `"1D`, whereas in the `T1` encoding it is in slot `"C6`. So the definition of `\AE` has to change depending on whether the current encoding is `OT1` or `T1`. The following commands allow this to happen.

`\DeclareTextCommand {`⟨*cmd*⟩`} {`⟨*encoding*⟩`} [`⟨*num*⟩`] [`⟨*default*⟩`] {`⟨*definition*⟩`}`

This command is like `\newcommand`, except that it defines a command which is specific to one encoding. For example, the definition of `\aa` in the `OT1` encoding is:

```
\DeclareTextCommand{\aa}{OT1}{{\accent23a}}
```

`\DeclareTextCommands` takes the same optional arguments as `\newcommand`.

The resulting command is robust, even if the code in ⟨*definition*⟩ is fragile.

It does not produce an error if the command has already been defined but logs the redefinition in the transcript file.

`\ProvideTextCommand {`⟨*cmd*⟩`} {`⟨*encoding*⟩`} [`⟨*num*⟩`] [`⟨*default*⟩`] {`⟨*definition*⟩`}` New feature 1994/12/01

This command is the same as `\DeclareTextCommand`, except that if ⟨*cmd*⟩ is already defined in encoding ⟨*encoding*⟩, then the definition is ignored.

`\DeclareTextSymbol` {⟨*cmd*⟩} {⟨*encoding*⟩} {⟨*slot*⟩}

This command defines a text symbol with slot ⟨*slot*⟩ in the encoding. For example, the definition of `\ss` in the `OT1` encoding is:

```
\DeclareTextSymbol{\ss}{OT1}{25}
```

It does not produce an error if the command has already been defined but logs the redefinition in the transcript file.

`\DeclareTextAccent` {⟨*cmd*⟩} {⟨*encoding*⟩} {⟨*slot*⟩}

This command declares a text accent, with the accent taken from slot ⟨*slot*⟩ in the encoding. For example, the definition of `\"` in the `OT1` encoding is:

```
\DeclareTextAccent{\"}{OT1}{127}
```

It does not produce an error if the command has already been defined but logs the redefinition in the transcript file.

`\DeclareTextComposite` {⟨*cmd*⟩} {⟨*encoding*⟩} {⟨*letter*⟩} {⟨*slot*⟩}

This command declares that the composite letter formed from applying ⟨*cmd*⟩ to ⟨*letter*⟩ is defined to be simply slot ⟨*slot*⟩ in the encoding. The ⟨*letter*⟩ should be a single letter (such as `a`) or a single command (such as `\i`).

For example, the definition of `\'{a}` in the `T1` encoding could be declared like this:

```
\DeclareTextComposite{\'}{T1}{a}{225}
```

The ⟨*cmd*⟩ should have been previously declared for this encoding, either with using `\DeclareTextAccent`, or as a one-argument `\DeclareTextCommand`.

`\DeclareTextCompositeCommand` {⟨*cmd*⟩} {⟨*encoding*⟩} {⟨*letter*⟩} {⟨*definition*⟩} New feature 1994/12/01

This is a more general form of `\DeclareTextComposite`, which allows for an arbitrary ⟨*definition*⟩, not just a ⟨*slot*⟩. The main use for this is to allow accents on `i` to act like accents on `\i`, for example:

```
\DeclareTextCompositeCommand{\'}{OT1}{i}{\'\i}
```

It has the same restrictions as `\DeclareTextComposite`.

5.3 Default definitions

The commands in ⟨*ENC*⟩`enc.def` files allow encoding-specific commands to be defined, but they do not allow commands to be used in other encodings. For example, the OMS encoding contains the symbol '§', but we need to be able to use the command `\S` in any encoding, not just OMS. New description 1994/12/01

To allow this, LaTeX has commands for giving default definitions for commands, which are used when the command is not defined in the current encoding. For example, the default encoding for `\S` is OMS, and so in an encoding (such as OT1) which does not contain `\S`, the OMS version is selected. But in an encoding (such as T1) which does contain `\S`, the version for that encoding is used.

Note: These commands should *not* occur in ⟨*ENC*⟩`enc.def` files, since these should only define commands for that encoding. They should instead be placed in packages.

`\DeclareTextCommandDefault {`⟨*cmd*⟩`} {`⟨*definition*⟩`}` New feature 1994/12/01

This command allows an encoding-specific command to be given a default definition. For example, the default definition for `\copyright` is defined be be a circled 'c' with:

```
\DeclareTextCommandDefault{\copyright}{\textcircled{c}}
```

`\DeclareTextAccentDefault {`⟨*cmd*⟩`} {`⟨*encoding*⟩`}`
`\DeclareTextSymbolDefault {`⟨*cmd*⟩`} {`⟨*encoding*⟩`}` New feature 1994/12/01

These commands allow an encoding-specific command to be given a default encoding. For example, the default encoding for `\"` and `\ae` is set to be OT1 by:

```
\DeclareTextAccentDefault{\"}{OT1}
\DeclareTextSymbolDefault{\ae}{OT1}
```

Note that `\DeclareTextAccentDefault` can be used on any one-argument encoding-specific command, not just those defined with `\DeclareTextAccent`. Similarly, `\DeclareTextSymbolDefault` can be used on any encoding-specific command with no arguments, not just those defined with `\DeclareTextSymbol`.

For more examples of these definitions, see `ltoutenc.dtx`.

`\ProvideTextCommandDefault {`⟨*cmd*⟩`} {`⟨*definition*⟩`}` New feature 1994/12/01

This command is the same as `\DeclareTextCommandDefault`, except that if the command already has a default definition, then the definition is ignored. This is useful to give 'faked' definitions of symbols which may be given 'real' definitions by other packages. For example, a package might give a fake definition of `\textonequarter` by saying:

```
\ProvideTextCommandDefault{\textonequarter}{$\m@th\frac14$}
```

5.4 Encoding defaults

`\DeclareFontEncodingDefaults {⟨text-settings⟩} {⟨math-settings⟩}`

Declares ⟨*text-settings*⟩ and ⟨*math-settings*⟩ for all encoding schemes. These are executed before the encoding scheme dependent ones are executed so that one can use the defaults for the major cases and overwrite them if necessary using `\DeclareFontEncoding`.

If `\relax` is used as an argument, the current setting of this default is left unchanged.

This example is used by amsfonts.sty for accent positioning; it changes only the math settings:

```
\DeclareFontEncodingDefaults{\relax}{\def\accentclass@{7}}
```

`\DeclareFontSubstitution {⟨encoding⟩} {⟨family⟩} {⟨series⟩} {⟨shape⟩}`

Declares the default values for font substitution which will be used when a font with encoding ⟨*encoding*⟩ should be loaded but no font can be found with the current attributes.

These substitutions are local to the encoding scheme because the encoding scheme is never substituted! They are tried in the order ⟨*shape*⟩ then ⟨*series*⟩ and finally ⟨*family*⟩.

If no defaults are set up for an encoding, the values given by `\DeclareErrorFont` are used.

The font specification for ⟨*encoding*⟩⟨*family*⟩⟨*series*⟩⟨*shape*⟩ must have been defined by `\DeclareFontShape` before the `\begin{document}` is reached.

Example:

```
\DeclareFontSubstitution{T1}{cmr}{m}{n}
```

5.5 Case changing

`\MakeUppercase {⟨text⟩}`
`\MakeLowercase {⟨text⟩}`

TeX provides two primitives `\uppercase` and `\lowercase` for changing the case of text. Unfortunately, these TeX primitives do not change the case of characters accessed by commands like `\ae` or `\aa`. To overcome this problem, LaTeX provides these two commands. New feature 1995/06/01

In the long run, we would like to use all-caps fonts rather than any command like `\MakeUppercase` but this is not possible at the moment because such fonts do not exist.

For further details, see `clsguide.tex`.

In order that upper/lower-casing will work reasonably well, and in order to provide any correct hyphenation, LaTeX 2_ε *must* use, throughout a document, the same fixed table for changing case. The table used is designed for the font encoding `T1`; this works well with the standard TeX fonts for all Latin alphabets but will cause problems when using other alphabets. New description 1995/12/01

6 Miscellanea

This section covers the remaining font commands of LaTeX.

6.1 Font substitution

`\DeclareErrorFont {`⟨*encoding*⟩`} {`⟨*family*⟩`} {`⟨*series*⟩`} {`⟨*shape*⟩`} {`⟨*size*⟩`}`

Declares ⟨*encoding*⟩⟨*family*⟩⟨*series*⟩⟨*shape*⟩ to be the font shape used in cases where the standard substitution mechanism fails (i.e. would loop). For the standard mechanism see the command `\DeclareFontSubstitution` above.

The font specification for ⟨*encoding*⟩⟨*family*⟩⟨*series*⟩⟨*shape*⟩ must have been defined by `\DeclareFontShape` before the `\begin{document}` is reached.

Example:

```
\DeclareErrorFont{OT1}{cmr}{m}{n}{10}
```

`\fontsubfuzz`

This parameter is used to decide whether or not to produce a terminal warning if a font size substitution takes place. If the difference between the requested and the chosen size is less than `\fontsubfuzz` the warning is only written to the transcript file. The default value is `0.4pt`. This can be redefined with `\renewcommand`, for example:

```
\renewcommand{\fontsubfuzz}{0pt}   % always warn
```

6.2 Preloading

`\DeclarePreloadSizes {`⟨*encoding*⟩`} {`⟨*family*⟩`} {`⟨*series*⟩`} {`⟨*shape*⟩`} {`⟨*size-list*⟩`}`

Specifies the fonts that should be preloaded by the format. These commands should be put in a `preload.cfg` file, which is read in when the LaTeX format is being built. Read `preload.dtx` for more information on how to built such a configuration file.

Example:

```
\DeclarePreloadSizes{OT1}{cmr}{m}{sl}{10,10.95,12}
```

6.3 Naming conventions

- Math alphabet commands all start with `\math...`: examples are `\mathbf`, `\mathcal`, etc.
- The text font changing commands with arguments all start with `\text...`: e.g. `\textbf` and `\textrm`. The exception to this is `\emph`, since it occurs very commonly in author documents and so deserves a shorter name.
- Names for encoding schemes are strings of up to three letters, all upper case. The LaTeX3 project reserves the use of encodings starting with `T` (standard 256-long text encodings), `M` (standard 256-long math encodings), `S` (standard 256-long symbol encodings), `OT` (standard 128-long text encodings) and `OM` (standard 128-long math encodings). Please do not use the above starting letters for non-portable encodings. If new standard encoding emerge then we shall add them in a later release of LaTeX.

  Encoding schemes which are local to a site should start with `L`.
- Font family names should contain up to five lower case letters. Where possible, these should conform to the *Filenames for fonts* font naming scheme.
- Font series names should contain up to four lower case letters.
- Font shapes should contain up to two letters lower case.
- Names for symbol fonts are built from lower and upper case letters with no restriction.

Whenever possible, you should use the series and shape names suggested in *The LaTeX Companion* since this will make it easier to combine new fonts with existing fonts.

Where possible, text symbols should be named as `\text` followed by the Adobe glyph name: for example `\textonequarter` or `\textsterling`. Similarly, math symbols should be named as `\math` followed by the glyph name, for example `\mathonequarter` or `\mathsterling`. Commands which can be used in text or math can then be defined using `\ifmmode`, for example:

New description 1994/12/01

```
\DeclareRobustCommand{\pounds}{%
   \ifmmode \mathsterling \else \textsterling \fi
}
```

Note that commands defined in this way must be robust, in case they get put into a section title or other moving argument.

References

[1] Michel Goossens, Frank Mittelbach and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1994.

[2] Donald E. Knuth. Typesetting concrete mathematics. *TUGboat*, 10(1):31–36, April 1989.

[3] Leslie Lamport. *LaTeX: A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, second edition, 1994.