LaTeX 2$_\varepsilon$ for authors

© Copyright 1994, LaTeX3 Project Team.
All rights reserved.

June 1994

Contents

1 Introduction

Welcome to LaTeX 2$_\varepsilon$, the new standard version of the LaTeX Document Preparation System.

This document describes how to take advantage of the new features of LaTeX, and how to process your old LaTeX documents with LaTeX 2$_\varepsilon$. However, this document is only a brief introduction to the new facilities and is intended for authors who are already familiar with the old version of LaTeX. It is *neither* a user-guide *nor* a reference manual for LaTeX 2$_\varepsilon$.

1.1 LaTeX 2$_\varepsilon$—The new LaTeX release

The previous version of LaTeX was known as LaTeX 2.09. Over the years many extensions have been developed for LaTeX. This is, of course, a sure sign of its continuing popularity but it has had one unfortunate result: incompatible LaTeX formats came into use at different sites. This included 'standard LaTeX 2.09', LaTeX built with the *New Font Selection Scheme* (NFSS), SliTeX, AMS-LaTeX, and so on. Thus, to process documents from various places, a site maintainer was forced to keep multiple versions of the LaTeX program. In addition, when looking at a source file it was not always clear for which format the document was written.

To put an end to this unsatisfactory situation, LaTeX 2$_\varepsilon$ has been produced; it brings all such extensions back under a single format and thus prevents the proliferation of mutually incompatible dialects of LaTeX 2.09. With LaTeX 2$_\varepsilon$ the 'new font selection scheme' is standard and, for example, amstex (formerly the AMS-LaTeX format) or slides (formerly the SliTeX format) are simply extension packages, all working with the same base format.

The introduction of a new release also made it possible to add a small number of often-requested features and to make the task of writing packages and classes simpler.

1.2 LaTeX3—The long-term future of LaTeX

LaTeX 2$_\varepsilon$ is the consolidation step in a comprehensive reimplementation of the LaTeX system. The next major release of LaTeX will be LaTeX3, which will include a radical overhaul of the document designers' and package writers' interface to LaTeX.

LaTeX3 is a long-term research project but, until it is completed, the project team are committed to the active maintenance of LaTeX 2$_\varepsilon$. Thus the experience gained from the production and maintenance of LaTeX 2$_\varepsilon$ will be a major influence on the design of LaTeX3.

1.3 Overview

This document contains an overview of the new structure and features of LaTeX. It is *not* a self-contained document, as it contains only the features of LaTeX which have changed since version 2.09. You should read this document in conjunction with an introduction to LaTeX.

Section 2 contains an overview of the new structure of LaTeX documents. It describes how classes and packages work and how class and package options can be used. It lists the standard packages and classes which come with LaTeX.

Section 3 describes the new commands available to authors in LaTeX 2$_\varepsilon$.

Section 4 shows how to process old LaTeX documents with LaTeX 2$_\varepsilon$.

Section 5 contains advice on dealing with problems you may encounter in running LaTeX 2$_\varepsilon$. It lists some error messages which are new in LaTeX 2$_\varepsilon$ and it describes some of the more common problems and how to cure them, or where to find further information.

1.4 Further information

For a general introduction to LaTeX, including the new features of LaTeX 2$_\varepsilon$, you should read *LaTeX: A Document Preparation System* by Leslie Lamport [3].

A more detailed description of the new features of LaTeX, including an overview of more than 150 packages, is to be found in *The LaTeX Companion* by Michel Goossens, Frank Mittelbach and Alexander Samarin [1].

For more information about the many new LaTeX packages you should read the package documentation, which should be available from the same source as your copy of LaTeX.

There are a number of documentation files which accompany every copy of LaTeX. A copy of *LaTeX News* will come out with each six-monthly release of LaTeX; it will be found in the files `ltnews*.tex`. The class- and package-writer's guide *LaTeX 2ε for Class and Package Writers* describes the new LaTeX features for writers of document classes and packages; it is in `clsguide.tex`. The guide *LaTeX 2ε Font Selection* describes the LaTeX font selection scheme for class- and package-writers; it is in `fntguide.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, P. O. Box 869, Santa Barbara, CA 93102-0869, USA, Fax: +1 805 963 8358, E-mail: tug@tug.org.

2 Classes and packages

This section describes the new structure of LaTeX documents and the new types of file: *classes* and *packages*.

2.1 What are classes and packages?

The main difference between LaTeX 2.09 and LaTeX 2ε is in the commands before `\begin{document}`.

In LaTeX 2.09, documents had *styles*, such as `article` or `book`, and *options*, such as `twoside` or `epsfig`. These were indicated by the `\documentstyle` command:

```
\documentstyle[⟨options⟩]{⟨style⟩}
```

For example, to specify a two-sided article with encapsulated PostScript figures, you said:

```
\documentstyle[twoside,epsfig]{article}
```

However, there were two different types of document style option: *built-in options* such as `twoside`; and *packages* such as `epsfig.sty`. These were very different, since any LaTeX document style could use the `epsfig` package but only document styles which declared the `twoside` option could use it.

To avoid this confusion, LaTeX 2ε differentiates between built-in options and packages. These are given by the new `\documentclass` and `\usepackage` commands:

```
\documentclass[⟨options⟩]{⟨class⟩}
\usepackage[⟨options⟩]{⟨packages⟩}
```

For example, to specify a two-sided article with encapsulated PostScript figures, you now write:

```
\documentclass[twoside]{article}
\usepackage{epsfig}
```

You can load more than one package with a single `\usepackage` command; for example, rather than writing:

```
\usepackage{epsfig}
\usepackage{multicol}
```

you can specify:

```
\usepackage{epsfig,multicol}
```

Note that LaTeX 2_ε still understands the LaTeX 2.09 `\documentstyle` command. This command causes LaTeX 2_ε to enter *LaTeX 2.09 compatibility mode*, which is described in Section 4.

You should not, however, use the `\documentstyle` command for new documents because this compatibility mode is very slow and the new features of LaTeX 2_ε are not available in this mode.

To help differentiate between classes and packages, document classes now end with `.cls` rather than `.sty`. Packages still end with `.sty`, since most LaTeX 2.09 packages work well with LaTeX 2_ε.

2.2 Class and package options

In LaTeX 2.09, only document styles could have options such as `twoside` or `draft`. In LaTeX 2_ε, both classes and packages are allowed to have options. For example, to specify a two-sided article with graphics using the `dvips` driver, you write:

```
\documentclass[twoside]{article}
\usepackage[dvips]{graphics}
```

It is possible for packages to share common options. For example, you may could, in addition, load the `color` package by specifying:

```
\documentclass[twoside]{article}
\usepackage[dvips]{graphics}
\usepackage[dvips]{color}
```

But because `\usepackage` allows more than one package to be listed, this can be shortened to:

```
\documentclass[twoside]{article}
\usepackage[dvips]{graphics,color}
```

In addition, packages will also use each option given to the `\documentclass` command (if they know what to do with it), so you could also write:

```
\documentclass[twoside,dvips]{article}
\usepackage{graphics,color}
```

Class and package options are covered in more detail in *The LaTeX Companion* and in *LaTeX 2ε for Class and Package Writers*.

2.3 Standard classes

The following classes are distributed with LaTeX:

article The `article` class described in *LaTeX: A Document Preparation System*.

book The `book` class described in *LaTeX: A Document Preparation System*.

report The `report` class described in *LaTeX: A Document Preparation System*.

slides The `slides` class described in *LaTeX: A Document Preparation System*, formerly SliTeX.

ltxdoc The document class for documenting the LaTeX program, based on `article`.

ltxguide The document class for *LaTeX 2ε for Authors* and *LaTeX 2ε for Class and Package Writers*, based on `article`. The document you are reading now uses the `ltxguide` class. The layout for this class is likely to change in future releases of LaTeX.

ltnews The document class for the *LaTeX News* information sheet, based on `article`. The layout for this class is likely to change in future releases of LaTeX.

proc A document class for proceedings, based on `article`. Formerly the `proc` package.

2.4 Standard packages

The following packages are distributed with LaTeX:

doc Allows the documentation of LaTeX programs. Described in `doc.dtx` and in *The LaTeX Companion*.

exscale Provides scaled versions of the math extension font. Described in `ltexscale.dtx`.

fontenc Specifies which font encoding LaTeX should use. Described in `ltoutenc.dtx`.

ifthen Provides commands of the form 'if...then do...otherwise do...'. Described in `ifthen.dtx` and *The LaTeX Companion*.

latexsym LaTeX 2ε no longer loads the LaTeX symbol font by default. To access it, you should use the `latexsym` package. Described in `latexsym.dtx` and in *The LaTeX Companion*; see also Section 5.

makeidx Provides commands for producing indexes. Described in *LaTeX: A Document Preparation System* and in *The LaTeX Companion*.

newlfont Emulates the font commands of LaTeX 2.09 with the New Font Selection Scheme. Described in *The LaTeX Companion*.

oldlfont Emulates the font commands of LaTeX 2.09. Described in *The LaTeX Companion*.

syntonly Processes a document without typesetting it. Described in `syntonly.dtx` and in *The LaTeX Companion*.

tracefnt Allows you to control how much information about LaTeX's font loading is displayed. Described in *The LaTeX Companion*.

2.5 Related software

In addition, the following software should be available from the same distributor as your copy of LaTeX 2ε:

amslatex Advanced mathematical typesetting from the American Mathematical Society

babel Supports typesetting in many languages.

color Provides support for colour.

graphics Inclusion of graphics files.

mfnfss Typesetting with bit-map (Metafont) fonts.

psnfss Typesetting with Type 1 (PostScript) fonts.

tools Miscellaneous packages written by the LaTeX3 project team.

These packages come with documentation and each of them is also described in at least one of the books *The LaTeX Companion* and *LaTeX: A Document Preparation System*.

2.5.1 Tools

This collection of packages includes the following (some may have slightly different names on some systems):

array Extended versions of the environments array, tabular and tabular*, with many extra features.

dcolumn Alignment on 'decimal points' in tabular entries. Requires the array package.

delarray Adds 'large delimiters' around arrays. Requires array.

hhline Finer control over horizontal rules in tables. Requires array.

longtable Multi-page tables. (Does not require array, but it uses the extended features if both are loaded.)

tabularx Defines a tabularx environment that is similar to tabular* but it modifies the column widths, rather than the inter-column space, to achieve the desired table width.

afterpage Place text after the current page.

enumerate Extended version of the enumerate environment.

fontsmpl Package and test file for producing 'font samples'

ftnright Place all footnotes in the right hand column in two-column mode.

indentfirst Indent The first paragraph of sections etc.

multicol Typeset text in columns, with the length of the final columns 'balanced'.

rawfonts Preload fonts using the old internal font names of LaTeX2.09. See Section 5.2

somedefs Selective handling of package options. (Used in rawfonts.sty.)

showkeys Prints the 'keys' used by `\label`, `\ref`, `\cite` etc., useful whilst drafting.

theorem Flexible declaration of 'theorem-like' environments.

varioref 'Smart' handling of page references.

verbatim Flexible extension of the verbatim environment.

xspace Smart space command that helps you to avoid the common mistake of missing spaces after command names.

3 Commands

This section describes the new commands available in LaTeX 2ε. They are covered in more detail in *LaTeX: A Document Preparation System* and in *The LaTeX Companion.*

3.1 Initial commands

Initial commands can appear only before the `\documentclass` line.

```
\begin{filecontents} {⟨file-name⟩}
⟨file-contents⟩
\end{filecontents}
```

The `filecontents` environment is intended for bundling within a single document file the contents of packages, options, or other files. When the document file is run through LaTeX 2ε the body of this environment is written verbatim (preceded by a comment line) to a file whose name is given as the environment's only argument. However, if that file already exists then nothing happens except for an information message.

Only normal ASCII text characters (i.e. 7-bit visible text) should be included in a `filecontents` environment. Anything else, such as tab characters, form-feeds or 8-bit characters, should not be included in a

`filecontents` environment.

Tabs and form feeds produce a warning, explaining that they are turned into spaces or blank lines, respectively. What happens to 8-bit characters depends on the TeX installation and is in general unpredictable.

The `filecontents` environment is used for including LaTeX files. For other plain text files (such as Encapsulated PostScript files), you should use the `filecontents*` environment which does not add a comment line.

These environments are allowed only before `\documentclass`. This ensures that any packages that have been bundled in the document are present when needed.

3.2 Preamble commands

The changes to the preamble commands are intentionally designed to make LaTeX 2ε documents look clearly different from old documents. The commands should be used only before `\begin{document}`.

```
\documentclass [⟨option-list⟩] {⟨class-name⟩} [⟨release-date⟩]
```

This command replaces the LaTeX2.09 command `\documentstyle`.

There must be exactly one `\documentclass` command in a document; and it must come after the `filecontents` environments, if any, but before any other commands.

The ⟨*option-list*⟩ is a list of options, each of which may modify the formatting of elements which are defined in the ⟨*class-name*⟩ file, as well as those in all following `\usepackage` commands (see below).

The optional argument ⟨*release-date*⟩ can be used to specify the earliest desired release date of the class file; it should contain a date in the format YYYY/MM/DD. If a version of the class older than this date is found, a warning is issued.

For example, to specify a two-column article, using a version of `article.cls` released after June 1994, you specify:

```
\documentclass[twocolumn]{article}[1994/06/01]
```

```
\documentstyle [⟨option-list⟩] {⟨class-name⟩}
```

This command is still supported for compatibility with old files. It is essentially the same as `\documentclass` except that it invokes *LaTeX 2.09 compatibility mode*. It also causes any options in the ⟨*option-list*⟩ that are not processed by the class file to be loaded as packages after the class has been loaded. See Section 4 for more details on LaTeX 2.09 compatibility mode.

```
\usepackage [⟨option-list⟩] {⟨package-name⟩} [⟨release-date⟩]
```

Any number of `\usepackage` commands is allowed. Each package file (as denoted by ⟨*package-name*⟩) defines new elements (or modifies those defined in the class file loaded by the ⟨*class-name*⟩ argument of the `\documentclass` command). A package file thus extends the range of documents which can be processed.

The ⟨*option-list*⟩ argument can contain a list of options, each of which can modify the formatting of elements which are defined in this ⟨*package-name*⟩ file.

As above, ⟨*release-date*⟩ can contain the earliest desired release date of the package file in the format YYYY/MM/DD; if an older version of the package is found, a warning is issued.

For example, to load the `graphics` package for the `dvips` driver, using a version of `graphics.sty` released after June 1994, you write:

```
\usepackage[dvips]{graphics}[1994/06/01]
```

Each package is loaded only once. If the same package is requested more than once, nothing happens in the second or following attempt unless the package has been requested with options that were not given in the original `\usepackage`. If such extra options are specified then an error message is produced. See Section 5 how to resolve this problem.

As well as processing the options given in the ⟨*option-list*⟩ of the `\usepackage` command, each package processes the ⟨*option-list*⟩ of the `\documentclass` command as well. This means that any option which should be processed by every package (to be precise, by every package that specifies an action for it) can be specified just once, in the `\documentclass` command, rather than being repeated for each package that needs it.

```
\listfiles
```

If this command is placed in the preamble then a list of the files read in (as a result of processing the document) will be displayed on the terminal (and in the log file) at the end of the run. Where possible, a short description will also be produced.

Warning: this command will list only files which were read using LaTeX commands such as `\input{`⟨*file*⟩`}` or `\include{`⟨*file*⟩`}`. If the file was read using the primitive TeX file input command then it will not be listed.

```
\setcounter{errorcontextlines} {⟨num⟩}
```

TeX3 introduced a new primitive `\errorcontextlines` which controls the format of error messages. LaTeX 2$_\varepsilon$ provides an interface to this through the standard `\setcounter` command. As most LaTeX users do not want to see the internal definitions of LaTeX commands each time they make an error, LaTeX 2$_\varepsilon$ sets this to -1 by default.

3.3 Document structure

The `book` document class introduces new commands to indicate document structure.

```
\frontmatter
\mainmatter
\backmatter
```

These commands indicate the beginning of the front matter (title page, table of contents and prefaces), main matter (main text) and back matter (bibliography, indexes and colophon).

3.4 Definitions

In LaTeX, commands can have both mandatory and optional arguments, for example in:

```
\documentclass[11pt]{article}
```

the `11pt` argument is optional, whereas the `article` class name is mandatory.

In LaTeX 2.09 users could define commands with arguments, but these had to be mandatory arguments. With LaTeX 2_{ε}, users can now define commands and environments which also have one optional argument.

```
\newcommand {⟨cmd⟩} [⟨num⟩] [⟨default⟩] {⟨definition⟩}
\renewcommand {⟨cmd⟩} [⟨num⟩] [⟨default⟩] {⟨definition⟩}
```

These commands have a new, second, optional argument; this is used for defining commands which themselves take one optional argument. This new argument is best introduced by means of a simple (and hence not very practical) example:

```
\newcommand{\example}[2][YYY]{Mandatory arg: #2;
                              Optional arg: #1.}
```

This defines `\example` to be a command with two arguments, referred to as `#1` and `#2` in the `{`⟨*definition*⟩`}`—nothing new so far. But by adding a second optional argument to this `\newcommand` (the `[YYY]`) the first argument (`#1`) of the newly defined command `\example` is made optional with its default value being `YYY`.

Thus the usage of `\example` is either:

```
\example{BBB}
```

which prints:

> Mandatory arg: BBB; Optional arg: YYY.

or:

```
\example[XXX]{AAA}
```

which prints:

> Mandatory arg: AAA; Optional arg: XXX.

The default value of the optional argument is `YYY`. This value is specified as the `[`⟨*default*⟩`]` argument of the `\newcommand` that created `\example`.

As another more useful example, the definition:

```
\newcommand{\seq}[2][n]{\lbrace #2_{0},\ldots,\,#2_{#1} \rbrace}
```

means that the input `$\seq{a}$` produces the formula $\{a_0,\ldots,\,a_n\}$, whereas the input `$\seq[k]{x}$` produces the formula $\{x_0,\ldots,\,x_k\}$.

In summary, the command:

```
\newcommand {⟨cmd⟩} [⟨num⟩] [⟨default⟩] {⟨definition⟩}
```

defines ⟨*cmd*⟩ to be a command with ⟨*num*⟩ arguments, the first of which is optional and has default value ⟨*default*⟩.

Note that there can only be one optional argument but, as before, there can be up to nine arguments in total.

```
\newenvironment {⟨cmd⟩} [⟨num⟩] [⟨default⟩] {⟨beg-def⟩} {⟨end-def⟩}
\renewenvironment {⟨cmd⟩} [⟨num⟩] [⟨default⟩] {⟨beg-def⟩} {⟨end-def⟩}
```

LaTeX 2ε also supports the creation of environments that have one optional argument. Thus the syntax of these two commands has been extended in the same way as that of `\newcommand`.

```
\providecommand {⟨cmd⟩} [⟨num⟩] [⟨default⟩] {⟨definition⟩}
```

This takes the same arguments as `\newcommand`. If ⟨*cmd*⟩ is already defined then the existing definition is kept; but if it is currently undefined then the effect of `\providecommand` is to define ⟨*cmd*⟩ just as if `\newcommand` had been used.

3.5 Boxes

These next three commands for making LR-boxes all existed in LaTeX 2.09. They have been enhanced in two ways.

```
\makebox [⟨width⟩] [⟨pos⟩] {⟨text⟩}
\framebox [⟨width⟩] [⟨pos⟩] {⟨text⟩}
\savebox {⟨cmd⟩} [⟨width⟩] [⟨pos⟩] {⟨text⟩}
```

One small but far-reaching change for LaTeX 2ε is that, within the ⟨*width*⟩ argument only, four special lengths can be used. These are all dimensions of the box that would be produced by using simply `\mbox{`⟨*text*⟩`}`:

`\height` its height above the baseline;

`\depth` its depth below the baseline;

`\totalheight` the sum of `\height` and `\depth`;

`\width` its width.

Thus, to put 'hello' in the centre of a box of twice its natural width, you would use:

```
\makebox[2\width]{hello}
```

Or you could put f into a square box, like this: $\boxed{f}$

```
\framebox{\makebox[\totalheight]{\itshape f\/}}
```

Note that it is the total width of the framed box, including the frame, which is set to `\totalheight`.

The other change is a new possibility for ⟨*pos*⟩: `s` has been added to `l` and `r`. If ⟨*pos*⟩ is `s` then the text is stretched the full length of the box, making use of any 'rubber lengths' (including any inter-word spaces) in the box's contents. If no such 'rubber length' is present, an 'underfull box' will probably be produced.

```
\parbox [⟨pos⟩] [⟨height⟩] [⟨inner-pos⟩] {⟨width⟩} {⟨text⟩}
\begin{minipage} [⟨pos⟩] [⟨height⟩] [⟨inner-pos⟩] {⟨width⟩}
⟨text⟩
\end{minipage}
```

As for the box commands above, `\height`, `\width`, etc. may be used in the `[`⟨*height*⟩`]` argument to denote the natural dimensions of the box.

The ⟨*inner-pos*⟩ argument is new in LaTeX 2ε. It is the vertical equivalent to the ⟨*pos*⟩ argument for `\makebox`, etc, determining the position of ⟨*text*⟩ within the box. The ⟨*inner-pos*⟩ may be any one of `t`, `b`, `c`, or `s`, denoting top, bottom, centred, or 'stretched' alignment respectively. When the ⟨*inner-pos*⟩ argument is not specified, LaTeX gives it same value as ⟨*pos*⟩ (this could be the latter's default value).

```
\begin{lrbox} {⟨cmd⟩}
⟨text⟩
\end{lrbox}
```

This is an environment which does not directly print anything. Its effect is to save the typeset ⟨*text*⟩ in the bin ⟨*cmd*⟩. Thus it is like `\sbox {⟨cmd⟩} {⟨text⟩}`, except that any white space before or after the contents ⟨*text*⟩ is ignored.

This is very useful as it enables both the `\verb` command and the `verbatim` environment to be used within ⟨*text*⟩.

It also makes it possible to define, for example, a 'framed box' environment. This is done by first using this environment to save some text in a bin ⟨*cmd*⟩ and then calling `\fbox{\usebox{⟨cmd⟩}}`.

The following example defines an environment, called `fmpage`, that is a framed version of `minipage`.

```
\newsavebox{\fmbox}
\newenvironment{fmpage}[1]
  {\begin{lrbox}{\fmbox}\begin{minipage}{#1}}
  {\end{minipage}\end{lrbox}\fbox{\usebox{\fmbox}}}
```

3.6 Measuring things

The first of these next commands was in LaTeX 2.09. The two new commands are the obvious analogues.

```
\settowidth {⟨length-cmd⟩} {⟨lr text⟩}
\settoheight {⟨length-cmd⟩} {⟨lr text⟩}
\settodepth {⟨length-cmd⟩} {⟨lr text⟩}
```

3.7 Controlling page breaks

Sometimes it is necessary, for a final version of a document, to 'help' LaTeX break the pages in the best way. LaTeX 2.09 had a variety of commands for this situation: `\clearpage`, `\pagebreak` etc. LaTeX2ε provides, in addition, commands which can produce longer pages as well as shorter ones.

```
\enlargethispage {⟨size⟩}
\enlargethispage* {⟨size⟩}
```

These commands increase the height of a page (from its normal value of `\textheight`) by the specified amount ⟨*size*⟩, a rigid length. This change affects *only* the current page.

This can be used, for example, to allow an extra line to be fitted onto the page or, with a negative length, to produce a page shorter than normal.

The star form also shrinks any vertical white space on the page as much as possible, so as to fit the maximum amount of text on the page.

3.8 Floats

There is a new command, `\suppressfloats`, and a new 'float specifier'. These will enable people to gain better control of LaTeX's float placement algorithm.

`\suppressfloats [`⟨*placement*⟩`]`

This command stops any further floating environments from being placed on the current page. With an optional argument, which should be either `t` or `b` (not both), this restriction applies only to putting further floats at the top or at the bottom. Any floats which would normally be placed on this page are placed on the next page instead.

The extra float location specifier: `!`

This can be used, along with at least one of `h`, `t`, `b` and `p`, in the location optional argument of a float.

If a `!` is present then, just for this particular float, whenever it is processed by the float mechanism the following are ignored:

- all restrictions on the number of floats which can appear;
- all explicit restrictions on the amount of space on a text page which may be occupied by floats or must be occupied by text.

The mechanism will, however, still attempt to ensure that pages are not overfull and that floats of the same type are printed in the correct order.

Note that its presence has no effect on the production of float pages.

A `!` specifier overrides the effect of any `\suppressfloats` command for this particular float.

3.9 Font changing: text

LaTeX 2ε has a very different font selection scheme to LaTeX 2.09. In this section, we give a brief description of the new commands. A more detailed description with examples is given in *The LaTeX Companion*, and the interface for class- and package-writers is described in *LaTeX 2ε Font Selection*.

```
\rmfamily
\sffamily
\ttfamily
\mdseries
\bfseries
\upshape
\itshape
\slshape
\scshape
```

These are font commands whose use is the same as the commands `\rm`, `\bf`, etc. The difference is that each command changes just one attribute of the font (the attribute changed is part of the name). One result of this is that, for example, `\bfseries\itshape` produces both a change of series and a change of shape, to give a bold italic font.

```
\textrm{⟨text⟩}
\textsf{⟨text⟩}
\texttt{⟨text⟩}
\textmd{⟨text⟩}
\textbf{⟨text⟩}
\textup{⟨text⟩}
\textit{⟨text⟩}
\textsl{⟨text⟩}
\textsc{⟨text⟩}
\emph{⟨text⟩}
```

These are one-argument commands; they take as an argument the text which is to be typeset in the particular font. They also automatically insert italic corrections where appropriate; if you do not like the result, you can add an italic correction with `\/` or remove it with `\nocorr`. The `\nocorr` should always be the first or last thing within the `{`⟨*text*⟩`}` argument.

3.10 Font changing: math

Most of the fonts used within math mode do not need to be explicitly invoked; but to use letters from a range of fonts, the following class of commands is provided.

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

These are also one-argument commands which take as an argument the letters which are to be typeset in the particular font. The argument is processed in math mode so spaces within it will be ignored. Only letters, digits and accents have their font changed, for example `$\mathbf{\tilde A \times 1}$` produces $\tilde{\mathbf{A}} \times \mathbf{1}$.

3.11 Ensuring math mode

```
\ensuremath {⟨math commands⟩}
```

In LaTeX 2.09, if you wanted a command to work both in math mode and in text mode, the suggested method was to define something like:

```
\newcommand{\Gp}{\mbox{$G_p$}}
```

Unfortunately, the `\mbox` stops `\Gp` changing size correctly in (for instance) subscripts or a fraction.

In LaTeX 2_ε you can define it thus:

```
\newcommand{\Gp}{\ensuremath{G_p}}
```

Now `\Gp` will work correctly in all contexts.

This is because the `\ensuremath` does nothing, producing simply `{G_p}`, when `\Gp` is used within math mode; but it ensures that math mode is entered (and exited) as required when `\Gp` is used in text mode.

3.12 Logos

```
\LaTeX
\LaTeXe
```

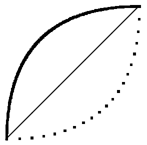
`\LaTeX` (producing 'LaTeX') is still the 'main' logo command, but if you need to refer to the new features, you can write `\LaTeXe` (producing 'LaTeX 2_ε').

3.13 Picture commands

```
\qbezier[⟨N⟩](⟨AX⟩,⟨AY⟩)(⟨BX⟩,⟨BY⟩)(⟨CX⟩,⟨CY⟩)
 \bezier{⟨N⟩}(⟨AX⟩,⟨AY⟩)(⟨BX⟩,⟨BY⟩)(⟨CX⟩,⟨CY⟩)
```

The `\qbezier` command can be used in `picture` mode to draw a quadratic bezier curve from position $(\langle AX\rangle,\langle AY\rangle)$ to $(\langle CX\rangle,\langle CY\rangle)$ with control point $(\langle BX\rangle,\langle BY\rangle)$. The optional argument $\langle N\rangle$ gives the number of points on the curve.

For example, the diagram:



is drawn with:

```
\begin{picture}(50,50)
   \thicklines
   \qbezier(0,0)(0,50)(50,50)
   \qbezier[20](0,0)(50,0)(50,50)
   \thinlines
   \put(0,0){\line(1,1){50}}
\end{picture}
```

The `\bezier` command is the same, except that the argument $\langle N\rangle$ is not optional. It is provided for compatibility with the LaTeX 2.09 `bezier` document style option.

3.14 Old commands

```
\samepage
```

The `\samepage` command still exists but is no longer being maintained. This is because it only ever worked erratically; it does not guarantee that there will be no page-breaks within its scope; and it can cause footnotes and marginals to be wrongly placed.

We recommend using `\enlargethispage` in conjunction with page-break commands such as `\newpage` and `\pagebreak` to help control page breaks.

```
\SLiTeX
```

Since SliTeX no longer exists, the logo is no longer defined in the LaTeX kernel. A suitable replacement is `\textsc{Sli\TeX}`. The SliTeX logo is defined in LaTeX 2.09 compatibility mode.

```
\mho \Join \Box \Diamond \leadsto
\sqsubset \sqsupset \lhd \unlhd \rhd \unrhd
```

These symbols are contained in the LaTeX symbol font, which was automatically loaded by LaTeX 2.09. However, TeX has room for only sixteen math font families; thus many users discovered that they ran out. Because of this, LaTeX does not load the LaTeX symbol font unless you use the `latexsym` package.

These symbols are also made available, using different fonts, by the `amsfonts` package, which also defines a large number of other symbols. It is supplied by the American Mathematical Society.

The `latexsym` package is loaded automatically in LaTeX 2.09 compatibility mode.

4 LaTeX 2.09 documents

LaTeX 2$_{\varepsilon}$ can process (almost) any LaTeX 2.09 document, by entering *LaTeX 2.09 compatibility mode*. Nothing has changed, you run LaTeX in the same way you always did, and you will get much the same results.

The reason for the 'almost' is that some LaTeX 2.09 packages made use of low-level unsupported features of LaTeX. If you discover such a package, you should find out if it has been updated to work with LaTeX 2$_{\varepsilon}$. Most packages will still work with LaTeX 2$_{\varepsilon}$—the easiest way to find out whether a package still works is to try it!

LaTeX 2.09 compatibility mode is a comprehensive emulation of LaTeX 2.09, but at the cost of time. Documents can run up to 50% slower in compatibility mode than they did under LaTeX 2.09. In addition, many of the new features of LaTeX 2$_{\varepsilon}$ are not available in LaTeX 2.09 compatibility mode.

4.1 Font selection problems

When using compatibility mode, it is possible that you will find problems with font-changing commands in some old documents. These problems are of two types:

- producing error messages;
- not producing the font changes you expected.

In case of error messages it is possible that the document (or an old style file used therein) contains references to old internal commands which are no longer defined, see Section 5.2 for more information if this is the case.

LaTeX 2.09 allowed sites to customize their LaTeX installation, which resulted in documents producing different results on different LaTeX installations. LaTeX 2$_\varepsilon$ no longer allows so much customization but, for compatibility with old documents, the local configuration file `latex209.cfg` is loaded every time LaTeX 2$_\varepsilon$ enters LaTeX 2.09 compatibility mode.

For example, if your site was customized to use the New Font Selection Scheme (NFSS) with the `oldlfont` option, then you can make LaTeX 2$_\varepsilon$ emulate this by creating a `latex209.cfg` file containing the command `\ds@oldlfont`. Similarly, to emulate NFSS with the `newlfont` option, you can create a `latex209.cfg` file containing `\ds@newlfont`.

4.2 Native mode

To run an old documents faster, and use the new features of LaTeX 2$_\varepsilon$, you should try using *LaTeX 2$_\varepsilon$ native mode.* This is done by replacing the command:

```
\documentstyle[⟨options⟩,⟨packages⟩]{⟨class⟩}
```

with:

```
\documentclass[⟨options⟩]{⟨class⟩}
\usepackage{latexsym,⟨packages⟩}
```

However, some documents which can be processed in LaTeX 2.09 compatibility mode may not work in native mode. Some LaTeX 2.09 packages will only work with LaTeX 2$_\varepsilon$ in 2.09 compatibility mode. Some documents will cause errors because of LaTeX 2$_\varepsilon$'s improved error detection abilities.

But most LaTeX 2.09 documents can be processed by LaTeX 2$_\varepsilon$'s native mode with the above change. Again, the easiest way to find out whether your documents can be processed in native mode is to try it!

5 Problems

This section describes some of the things which may go wrong when using LaTeX 2$_\varepsilon$, and what you can do about it.

5.1 New error messages

LaTeX 2$_\varepsilon$ has a number of new error messages. Please also note that many error messages now produce further helpful information if you press `h` in response to the error prompt.

`Option clash for package` *⟨package⟩*`.`

The named package has been loaded twice with different options. If you enter `h` you will be told what the options were, for example, if your document contained:

```
\usepackage[foo]{fred}
\usepackage[baz]{fred}
```

then you will get the error message:

```
Option clash for package fred.
```

and typing `h` at the `?` prompt will give you:

```
The package fred has already been loaded with options:
  [foo]
There has now been an attempt to load it with options:
  [baz]
Adding the line:
  \usepackage[foo,baz]{fred}
to your document may fix this.
Try typing <return> to proceed.
```

The cure is, as suggested, to load the package with both sets of options. Note that since LaTeX packages can call other packages, it is possible to get a package option clash without explicitly requesting the same package twice.

`Command` *⟨command⟩* `not provided in base NFSS.`

The *⟨command⟩* is not provided by default in LaTeX 2$_\varepsilon$. This error is generated by using one of the commands:

```
\mho \Join \Box \Diamond \leadsto
\sqsubset \sqsupset \lhd \unlhd \rhd \unrhd
```

which are now part of the `latexsym` package. The cure is to add:

```
\usepackage{latexsym}
```

in the preamble of your document.

`LaTeX2e command` *⟨command⟩* `in LaTeX 2.09 document.`

The *⟨command⟩* is a LaTeX 2$_\varepsilon$ command but this is a LaTeX 2.09 document. The cure is to replace the command by a LaTeX 2.09 command, or to run document in native mode, as described in Section 4.2.

```
NFSS release 1 command \newmathalphabet found.
```

The command `\newmathalphabet` was used by the New Font Selection Scheme Release 1 but it has now been replaced by `\DeclareMathAlphabet`, the use of which is described in *LaTeX 2ε Font Selection*.

The best cure is to update the package which contained the `\newmathalphabet` command. Find out if there is a new release of the package, or (if you wrote the package yourself) consult *LaTeX 2ε Font Selection* for the new syntax of font commands.

If there is no updated version of the package then you can cure this error by using the `newlfont` or `oldlfont` package, which tells LaTeX which version of `\newmathalphabet` should be emulated.

You should use `oldlfont` if the document selects math fonts with syntax such as this:

`{\cal A}`, etc.

Use `newlfont` if the document's syntax is like this:

`\cal{A}`, etc.

```
Text for \verb command ended by end of line.
```

The `\verb` command has been begun but not ended on that line. This usually means that you have forgotten to put in the end-character of the `\verb` command.

```
Illegal use of \verb command.
```

The `\verb` command has been used inside the argument of another command. This has never been allowed in LaTeX—often producing incorrect output without any warning—and so LaTeX 2ε produces an error message.

5.2 Old internal commands

A number of LaTeX 2.09 internal commands have been removed, since their functionality is now provided in a different way. See *LaTeX 2ε for Class and Package Writers* for more details of the new, supported interface for class and package writers.

```
\tenrm \elvrm \twlrm ...
\tenbf \elvbf \twlbf ...
\tensf \elvsf \twlsf ...
⋮
```

These commands provided access to the seventy fonts pre-loaded by LaTeX 2.09. In contrast, LaTeX 2ε normally pre-loads at most fourteen fonts, which saves a lot of font memory; but a consequence is that any LaTeX file which used the above commands to directly access fonts will no longer work.

Their use will usually produce an error message such as:

```
! Undefined control sequence.
l.5 \tenrm
```

The cure for this is to update the document to use the new font-changing commands provided by LaTeX 2ε; these are described in *LaTeX 2ε Font Selection*.

If this is not possible then, as a last resort, you can use the `rawfonts` package, which loads the seventy LaTeX 2.09 fonts and provides direct access to them using the old commands. This takes both time and memory. If you do not wish to load all seventy fonts, you can select some of them by using the `only` option to `rawfonts`. For example, to load only `tenrm` and `tenbf` you write:

```
\usepackage[only,tenrm,tenbf]{rawfonts}
```

The `rawfonts` package is distributed with the LaTeX tools software, see Section 2.4.

5.3 Old files

One of the more common mistakes in running LaTeX is to read in old versions of packages instead of the new versions. If you get an incomprehensible error message from a standard package, make sure you are loading the most recent version of the package. You can find out which version of the package has been loaded by looking in the log file for a line like:

```
Package ‘fred’ 1994/06/01 v0.01 Fred’s package.
```

You can use the ⟨*release-date*⟩ options to `\documentclass` and `\usepackage` to make sure that you are getting a suitably recent copy of the document class or package. This is useful when sending a document to another site, which may have out-of-date software.

5.4 Where to go for more help

If you can't find the answer for your problem here, try looking in *LaTeX: A Document Preparation System* or *The LaTeX Companion.* If you have a problem with installing LaTeX, look in the installation guide files which come with the distribution.

If this doesn't help, contact your local LaTeX guru or local LaTeX mailing list.

If you think you've discovered a bug then please report it! First, you should find out if the problem is with a third-party package or class. If the problem is caused by a package or class other than those listed in Section 2 then please report the problem to the author of the package or class, not to the LaTeX3 project team.

If the bug really is with core LaTeX then you should create a *short*, *self-contained* document which exhibits the problem. You should run a *recent* (less than a year old) version of LaTeX on the file and then run LaTeX on `latexbug.tex`. This will create an error report which you should send, together with the sample document and log file, to the LaTeX bugs address which can be found in the file `latexbug.tex` or `bugs.txt`.

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.