

# The fixltx2e and fix-cm packages\*

Frank Mittelbach, David Carlisle, Chris Rowley, Walter Schmidt†

2014/04/27

## Abstract

These packages provides fixes to L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> which are desirable but cannot be integrated into the L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> kernel or the font definition files directly as they would produce a version incompatible to earlier releases (either in formatting or functionality).

By providing these fixes in the form of packages, users can benefit from them without the danger that their documents will fail or produce unexpected results at other sites since the documents contain a clear indication (the `\usepackage` line, preferably with a required date) that the fixes are needed.

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Using fixltx2e . . . . .	4
1.2	Using fix-cm . . . . .	4
<b>2</b>	<b>Fixes added</b>	<b>4</b>
2.1	2-col: 1-col fig can come before earlier 2-col fig (pr/2346) . . . . .	4
2.1.1	Notes on the Implementation Strategy . . . . .	4
2.2	Wrong header for twocolumn (pr/2613) . . . . .	5
2.2.1	Notes on the Implementation Strategy . . . . .	5
2.3	\@ discards spaces when moving (pr/3039) . . . . .	6
2.4	\setlength produces error if used with registers like \dimen0 (pr/3066) . . . . .	6
2.5	\addpenalty ruins flush-bottom (pr/3073) . . . . .	6
<b>3</b>	<b>Fixes added for 2003/06/01</b>	<b>6</b>
3.1	\fnsymbol should use text symbols if possible (pr/3400) . . . . .	6
3.2	No hyphenation in first word after float environment (pr/3498) . .	7
3.3	Allowing \emph to produce small caps, etc . . . . .	7
3.4	Using EC fonts (T1 encoding) makes my documents look bl**dy horrible (from c.t.t.)	
	I can't use arbitrary sizes with CM fonts (from c.t.t.) . . . . .	7
3.4.1	What fix-cm does . . . . .	7
3.4.2	How to load the package . . . . .	8
3.4.3	Usage notes . . . . .	8

---

\*This file has version number v1.1p, last revised 2014/04/27.

†Walter wrote fix-cm

<b>4</b>	<b>Fixes added for 2005/12/01</b>	<b>9</b>
4.1	<code>\textsubscript</code> not defined in latex.ltx (pr/3492) . . . . .	9
4.2	<code>\DeclareMathSizes</code> only take pts. (pr/3693) . . . . .	9
4.3	<code>\addpenalty</code> ruins flush-bottom (pr/3073) . . . . .	9
4.4	<code>\footnotemark[x]</code> crashes with fixltx2e.sty (pr/3752) . . . . .	9
4.4.1	Notes on the implementation strategy . . . . .	9
4.5	Fewer fragile commands . . . . .	10
4.5.1	Notes on the implementation strategy . . . . .	10
<b>5</b>	<b>Fixes added for 2014/05/01</b>	<b>11</b>
5.1	Check the optional arguments of floats . . . . .	11
5.2	Infinite glue found (pr/4023 and pr/2346) . . . . .	11
<b>6</b>	<b>Implementation</b>	<b>11</b>
6.1	2-col: 1-col fig can come before earlier 2-col fig (pr/2346) Wrong headline for twocolumn (pr/2613) . . . . .	11
6.1.1	Preserving Marks . . . . .	11
6.1.2	Preserving Float Order . . . . .	12
6.2	<code>\@</code> discards spaces when moving (pr3039) . . . . .	17
6.3	<code>\setlength</code> produces error if used with registers like <code>\dimen0</code> (pr/3066) . . . . .	17
6.4	<code>\addpenalty</code> ruins flush-bottom (pr/3073) . . . . .	17
6.5	<code>\fnsymbol</code> should use text symbols if possible (pr/3400) . . . . .	18
6.6	No hyphenation in first word after float environment(pr/3498) . . .	19
6.7	Allowing <code>\emph</code> to produce small caps, etc . . . . .	19
6.8	<code>\textsubscript</code> not defined in latex.ltx (pr/3492) . . . . .	19
6.9	<code>\DeclareMathSizes</code> only take pts. (pr/3693) . . . . .	19
6.10	Fewer fragile macros . . . . .	20
6.11	Using EC fonts (T1 encoding) makes my documents look bl**dy horrible . . . . .	21
6.11.1	Preliminaries . . . . .	21
6.11.2	T1 encoding . . . . .	21
6.11.3	TS1 encoding . . . . .	24
6.11.4	OT1 encoding . . . . .	27
6.11.5	OML and OMS encoded math fonts . . . . .	29
6.11.6	$\TeX$ symbols . . . . .	29
6.12	Check the optional argument to floats . . . . .	30

# 1 Introduction

In the newsletter `ltnews07.tex`, which accompanied the  $\text{\LaTeX}$  2 <sub>$\epsilon$</sub>  maintenance release of June 1997, we wrote:

Many of the problem reports we receive concerning the standard classes are not concerned with bugs but are suggesting, more or less politely, that the design decisions embodied in them are ‘not optimal’ and asking us to modify them.

There are several reasons why we have decided not to make such changes to these files.

- However misguided, the current behaviour is clearly what was intended when these classes were designed.
- It is not good practice to change such aspects of ‘standard classes’ because many people will be relying on them.

We have therefore decided not to even consider making such modifications, nor to spend time justifying that decision. This does not mean that we do not agree that there are many deficiencies in the design of these classes, but we have many tasks with higher priority than continually explaining why the standard classes for  $\text{\LaTeX}$  cannot be changed.

Back then we probably should have said that this decision also covers changes to the  $\text{\LaTeX}$  kernel and font definitions, if the change results in noticeable differences in the formatting of documents or otherwise produces severe incompatibilities between releases. The important point to stress here is that “people rely on the fact that a document formatted at one site produces identical output at a different site”. By fixing a certain problem in version  $\langle date \rangle$ , people making use of the fix will get incorrectly formatted documents if they send them to others who still run on a version prior to  $\langle date \rangle$ .

In theory one could get around this by adding a line like

```
\NeedsTeXFormat{latex2e}[\langle date \rangle]
```

on top of the document. However, this fails for two reasons. Firstly, most people will not be aware that they make use of a feature or fix that is only available in their version of  $\text{\LaTeX}$ ; and thus do not add such a line in their documents. Secondly, even if there is such a line the receiving site might not be able to upgrade their  $\text{\LaTeX}$  in time to process the document properly (the latter is a sad fact of life).

By providing the `fixltx2e` and `fix-cm` packages we hope to help people in this respect since, when they are used, a document will contain a clear indication that special features/fixes are needed and if the receiving site does not have the packages available (or not available with the right version) it is far easier to download and install them from some archive than to upgrade  $\text{\LaTeX}$  in a rush.

The packages are independent from each other and deal with different subjects: `fixltx2e` provides general changes to the  $\text{\LaTeX}$  kernel, while `fix-cm` improves the definitions of the Computer Modern font families.

We will try to maintain the packages in such a way that they can be used with all maintenance releases of  $\text{\LaTeX}$  2 <sub>$\epsilon$</sub>  so that, if urgently needed, people can simply add them to the current directory in case they cannot upgrade their  $\text{\LaTeX}$  for whatever reason.

The packages are **NOT** provided so that people can stop upgrading their  $\text{\LaTeX}$  system. They will contain only fixes of a certain nature, others will still go into the kernel and extensions in form of packages, and support files will still be added to the base system at regular intervals.

## 1.1 Using fixltx2e

To use the fixltx2e package include the line

```
\usepackage{fixltx2e}[\langle date \rangle]
```

into the preamble of your document, where  $\langle date \rangle$  is the date of the fixltx2e package that you are using. This way your document will produce a warning if processed at a site that only has an older version of of this package.

## 1.2 Using fix-cm

To use the fix-cm package, load it *before* `\documentclass`, and use the command `\RequirePackage` to do so, rather than the normal `\usepackage`:

```
\RequirePackage{fix-cm}  
\documentclass ...
```

**Do not to load any other package before the document class**, unless you have a thorough understanding of the L<sup>A</sup>T<sub>E</sub>X internals and know exactly what you are doing!

## 2 Fixes added

This section describes all the fixes/features that have been added to the initial release of the package. If applicable the bug report info (see `bugs.txt`) is given.

### 2.1 2-col: 1-col fig can come before earlier 2-col fig (pr/2346)

```
>Number:      2346  
>Category:    latex  
>Synopsis:    2-col: 1-col fig can come before earlier 2-col fig  
>Arrival-Date: Wed Dec 18 15:41:07 1996  
>Originator:  w.l.kleb@larc.nasa.gov (bil kleb)  
>Description:  
as documented in lamport's book, p. 198, concerning figure  
placement, "a figure will not be printed before an earlier  
figure, and a table will not be printed before an earlier  
table." however, there is a footnote stating, "However,  
in two-column page style, a single-column figure can come before  
an earlier double-column figure, and vice versa."
```

```
this twocolumn behavior is undesireable---at least by me and  
most professional organizations i publish in. ed snyzter developed  
a hack fix for 2.09 several years ago which links the two  
counters, but i have not run across a similar "fix" for 2e...
```

Originally fixed in package fix2col which was merged into this package. Documentation and code from this package have been merged into this file.

#### 2.1.1 Notes on the Implementation Strategy

The standard output routine maintains two lists of floats that have been ‘deferred’ for later consideration. One list for single column floats, and one for double column floats (which are always immediately put onto their deferred list). This mechanism means that L<sup>A</sup>T<sub>E</sub>X ‘knows’ which type of float is contained in each box by the list that it is processing, but having two lists means that there is no mechanism for preserving the order between the floats in each list.

The solution to this problem consists of two small changes to the output routine.

Firstly, abandon the ‘double column float list’ `\@dbldeferlist` and change every command where it is used so that instead the same `\@deferlist` is used as for single column floats. That one change ensures that double and single column floats stay in the same sequence, but as  $\text{\LaTeX}$  no longer ‘knows’ whether a float is double or single column, it will happily insert a double float into a single column, overprinting the other column, or the margin.

The second change is to provide an alternative mechanism for recording the two column floats.  $\text{\LaTeX}$  already has a compact mechanism for recording float information, an integer count register assigned to each float records information about the ‘type’ of float ‘figure’, ‘table’ and the position information ‘htp’ etc.

The type information is stored in the ‘high’ bits, one bit position (above ‘32’) allocated to each float type. The ‘low’ bits store information about the allowed positions, one bit each allocated for `h t b p`. In the  $\text{\LaTeX}$ 2.09 system, the bit corresponding to ‘16’ formed a ‘boundary’ between these two sets of information, and it was never actually used by the system. Ed Sznyter’s `fixfloats` package not unreasonably used this position to store the double column information, setting the bit for double column floats. Then at each point in the output routine at which a float is committed to a certain region, an additional check must be made to check that the float is (or is not) double column. If it spans the wrong number of columns it is deferred rather than being added.

Unfortunately the bit ‘16’ is not available in  $\text{\LaTeX}$ 2<sub>ε</sub>. It is used to encode the extra float position possibility ‘!’ that was added in that system. It would be possible to use position ‘32’ and to move the flags for ‘table’, ‘figure’,... up one position, to start at 64, but this would mean that in principle one less float type would be supported, and more importantly is likely to break any other packages that assume anything about the output routine internals. So here I instead use another mechanism for flagging double column floats: By default all floats have depth 0pt. This package arranges that double column ones have depth 1sp. This information may then be used in the same manner as in the `fixfloats` package, to defer any floats that are not of the correct column spanning type.

## 2.2 Wrong header for twocolumn (pr/2613)

```
>Number:      2613
>Category:    latex
>Synopsis:    wrong headline for twocolumn
>Arrival-Date: Mon Sep 22 16:41:09 1997
>Originator:   daniel@cs.uni-bonn.de (Daniel Reischert)
>Description:
When setting the document in two columns
the headline shows the top mark of the second column,
but it should show the top mark of the first column.
```

Originally fixed in package `fix2col` which was merged into this package. Documentation and code from this package have been merged into this file.

### 2.2.1 Notes on the Implementation Strategy

The standard  $\text{\LaTeX}$  twocolumn system works internally by making each column a separate ‘page’ that is passed independently to  $\text{\TeX}$ ’s pagebreaker. (Unlike say the `multicol` package, where all columns are gathered together and then split into columns later, using `\vsplit`.) This means that the primitive  $\text{\TeX}$  marks that are normally used for header information, are globally reset after the first column. By default  $\text{\LaTeX}$  does nothing about this. A good solution is provided by Piet van Oostrum (building on earlier work of Joe Pallas) in his `fixmarks` package.

After the first column box has been collected the mark information for that box is saved, so that any `\firstmark` can be ‘artificially’ used to set the page-level marks after the second column has been collected. (The second column

`\firstmark` is not normally required.) Unfortunately  $\TeX$  does not provide a direct way of knowing if any marks are in the page, `\firstmark` always has a value from previous pages, even if there is no mark in this page. The solution is to make a copy of the box and then `\vsplit` it so that any marks show up as `\splitfirstmark`.

The use of `\vsplit` does mean that the output routine will globally change the value of `\splitfirstmark` and `\splitbotmark`. The `fixmarks` package goes to some trouble to save and restore these values so that the output routine does *not* change the values. This part of `fixmarks` is not copied here as it is quite costly (having to be run on every page) and there is no reason why anyone writing code using `\vsplit` should allow the output routine to be triggered before the split marks have been accessed.

## 2.3 `\@` discards spaces when moving (pr/3039)

```
>Number:      3039
>Category:    latex
>Synopsis:    \@ discards spaces when moving
>Arrival-Date: Sat May 22 09:01:06 1999
>Originator:  asnd@triumf.ca (Donald Arseneau)
>Description:
The \@ command expands to \spacefactor\@m in auxiliary files,
which then ignores following spaces when it is reprocessed.
```

## 2.4 `\setlength` produces error if used with registers like `\dimen0` (pr/3066)

```
>Number:      3066
>Category:    latex
>Synopsis:    \setlength{\dimen0}{10pt}
>Arrival-Date: Tue Jul 6 15:01:06 1999
>Originator:  oberdiek@ruf.uni-freiburg.de (Heiko Oberdiek)
>Description:
The current implementation of \setlength causes an error,
because the length specification isn't terminated properly.
More safe:
\def\setlength#1#2{#1=#2\relax}
```

## 2.5 `\addpenalty` ruins flush-bottom (pr/3073)

```
>Number:      3073
>Category:    latex
>Synopsis:    \addpenalty ruins flush-bottom
>Arrival-Date: Sat Jul 17 05:11:05 1999
>Originator:  asnd@triumf.ca (Donald Arseneau)
>Description:
Just to keep in mind for further development eh?
A page break at an \addpenalty after \vspace does *not*
give a flush-bottom page. (The intent of \addpenalty is
apparently just to preserve the flush bottom by putting
the breakpoint `above' the skip.)
```

# 3 Fixes added for 2003/06/01

## 3.1 `\fnsymbol` should use text symbols if possible (pr/3400)

```
>Number:      3400
>Category:    latex
>Synopsis:    \fnsymbol should use text symbols if possible
>Arrival-Date: Fri Jan 04 20:41:00 CET 2002
```

>Originator: was@VR-Web.de (Walter Schmidt)

The `\fnsymbol` command can be used in both text and math mode. The symbols produced are, however, always taken from the math fonts. As a result, they may not match the text fonts, even if the symbols are actually available, for instance from the TS1 encoding. Since `\fnsymbol` is primarily used for footnotes in text, this should be fixed, IMO.

### 3.2 No hyphenation in first word after float environment (pr/3498)

>Number: 3498  
>Category: latex  
>Synopsis: No hyphenation in first word after float environment  
>Arrival-Date: Thu Jan 30 13:21:00 CET 2003  
>Originator: h.harders@tu-bs.de (Harald Harders)

If a float environment (figure, table) is written within a paragraph, the first word after the environment is not hyphenated.

### 3.3 Allowing `\emph` to produce small caps, etc

By default `\em` or `\emph` switches to roman in an italic context but some designers prefer a switch to small caps in that case. This can be achieved by setting `\emminnershape`, e.g.,

```
\renewcommand\emminnershape{\scshape}
```

### 3.4 Using EC fonts (T1 encoding) makes my documents look bl\*\*dy horrible (from c.t.t.) I can't use arbitrary sizes with CM fonts (from c.t.t.)

No I'm not trying to collect any cites from the news group discussion on this topic. In a nutshell, if one adds

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

to a document that uses the Computer Modern typefaces, then not only the T1 encoding is used but the fonts used in the document look noticeably different. This is due to the fact that the EC fonts have more font series designs, e.g. a 14.4pt bold etc and those get used in the standard `.fd` files, while with Computer Modern (in OT1 encoding) such sizes were scaled versions of smaller sizes—with a noticeable different look and feel.

So we provide a package `fix-cm` to ensure that comparable definitions are used. In addition to that, the package `fix-cm` also enables continuous scaling of the CM fonts. This package was written by Walter Schmidt.

#### 3.4.1 What `fix-cm` does

Loading the package `fix-cm` changes the font definitions of the Computer Modern fonts, in order to achieve the following effects:

- The appearance of the T1 and TS1 encoded CM fonts (aka 'EC') is made as similar as possible to the traditional (OT1 encoded) ones. Particularly, a number of broken or ugly design sizes are no longer used, the look of the bold sans serif typeface at large sizes is considerably improved, and mismatches between the text fonts and the corresponding math fonts are avoided. As a side effect, PostScript and PDF documents may become smaller, because fewer fonts need to be embedded.

- The Computer Modern fonts are made available with arbitrary sizes.
- Only those design sizes of the fonts will be used, that are normally available in Type1 format, too. You need not load the extra package `cmmib57` for this purpose.

The package acts on the following font families:

- The text font families `cmr`, `cmss`, `cmtt` and `cmvtt` with OT1, T1 and TS1 encoding.
- The default math fonts used by  $\text{\LaTeX}$ , i.e., the font families `cmm` with encoding OML and `cms` with encoding OMS.
- The symbols used by the package `latexsym`, i.e., the font family `lasy`.

Note that the package does *not* act on:

- Font families such as CM Fibonacci, CM Dunhill etc., which are provided for experimental purposes or for fun only.
- CM text fonts with character sets other than Latin, e.g., Cyrillic. Loading of the required font and encoding definitions while the fonts are not actually used, would not be a good idea. This should be addressed by particular packages or by changing the standard FDs of these fonts.
- Extra math fonts such as the AMS symbol fonts. While they match the style of Computer Modern, they are frequently used in conjunction with other font families, too. Thus, `fix-cm` is obviously not the right place to make sure that they can be scaled continuously. Ask the maintainers of these fonts to provide this feature, which is badly needed!
- The math extension font `cmex`. Whether or not this font should be scaled is a question of its own, and there are other packages (`exscale`, `amsmath`, `amsfonts`) to take care of it.

### 3.4.2 How to load the package

The package should be loaded *before* `\documentclass`, using the command `\RequirePackage{fix-cm}`, rather than the normal `\usepackage`. Rationale: If the package is loaded in the preamble, a preceding package or even the code of the document class may have used any of the CM fonts already. However, the definitions of those fonts, that are already in use, cannot be changed any more.

### 3.4.3 Usage notes

In contrast to what you may expect, `fix-cm` does *not* ensure that line and page breaks stay the same, when you switch an existing document from OT1 to T1 encoding. The package does not turn off all of the additional design sizes in the EC fonts collection: Those, that contribute considerably to the typographical quality and do not conflict with the math fonts, are—indeed—used.

Be careful when using arbitrary, non-standard font sizes with applications that need bitmap fonts: You may end up with lots of possibly huge `.pk` files. Also, `Metafont` chokes sometimes on extremely small or large sizes, because of arithmetic problems.

`fix-cm` supersedes the experimental packages `cmsd` and `fix-ec`, which are no longer distributed.

The packages `type1cm` and `type1ec` must not be loaded additionally; they enable only continuous scaling.



## 4 Fixes added for 2005/12/01

### 4.1 `\textsubscript` not defined in latex.ltx (pr/3492)

>Number: 3492  
>Category: latex  
>Synopsis: `\textsubscript` not defined in latex.ltx  
>Arrival-Date: Tue Jan 14 23:01:00 CET 2003  
>Originator: tgakic@chem.tue.nl (Ionel Mugurel Ciobica)

I use `\textsubscript` much more often than `\textsuperscript`, and `\textsubscript` it is not defined in latex.ltx. Could you please consider including the definition of `\textsubscript` in the latex.ltx for the next versions of LaTeX. Thank you.

### 4.2 `\DeclareMathSizes` only take pts. (pr/3693)

>Number: 3693  
>Category: latex  
>Synopsis: `\DeclareMathSizes` only take pts.  
>Arrival-Date: Fri Jun 11 16:21:00 CEST 2004  
>Originator: moho01ab@student.cbs.dk (Morten Hoegholm)

The last three arguments of `\@DeclareMathSizes` cannot take a dimension as argument, making it inconsistent with the rest of the font changing commands and itself, as the second argument can take a dimension specification.

### 4.3 `\addpenalty` ruins flush-bottom (pr/3073)

>Number: 3073  
>Category: latex  
>Synopsis: `\addpenalty` ruins flush-bottom  
>Arrival-Date: 20 Oct 2005 14:46:35 -0700  
>Originator: asnd@triumf.ca (Donald Arseneau)  
>Description:  
The (revised) definition of `\addpenalty` has been incorporated into fixltx2e, but now Plamen Tanovski has found a problem: since the `\vskip` is increased by the previous depth, consecutive `\addpenalty` and `\addvspace` commands keep enlarging the `\vskip`.

### 4.4 `\footnotemark[x]` crashes with fixltx2e.sty (pr/3752)

>Number: 3752  
>Category: tools  
>Synopsis: feature `\footnotemark[x]` crashes with fixltx2e.sty  
>Arrival-Date: Fri Dec 17 10:11:00 +0100 2004  
>Originator: stefan.pofahl@zsw-bw.de (Stefan Pofahl)

If I use `/fnsymbol` together with fixltx2e.sty I can not use optional parameter `[num]`  
`\footnotemark[1]` is not showing the mark number 1 but the mark `\value{footnote}`.

This bug was related to pr/3400, where `\@fnsymbol` was made robust.

#### 4.4.1 Notes on the implementation strategy

Pr/3400 made `\@fnsymbol` decide between text-mode and math-mode, which requires a certain level of robustness somewhere as the decision between text and math must be made at typesetting time and not when inside `\protected@edef`

or similar commands. One way of dealing with this is to make sure the value seen by `\@fnsymbol` is a fully expanded number, which could be handled by code such as

```
\def\@fnsymbol#1{\expandafter\@fnsymbol
\expandafter{\the\c@#1\endcsname}}
```

This would be a good solution if everybody used the high level commands only by writing code like `\fnsymbol{footnote}`. Unfortunately many classes (including the standard classes) and packages use the internal forms directly as in `\@fnsymbol\c@footnote` so the easy solution of changing `\fnsymbol` would break code that had worked for the past 20 years.

Therefore the implementation here makes `\@fnsymbol` itself a non-robust command again and instead uses a new robust command `\TextOrMath`, which will take care of typesetting either the math or the text symbol. In order to do so, we face an age old problem and unsolvable problem in T<sub>E</sub>X: A reliable test for math mode that doesn't destroy kerning. Fortunately this problem can be solved when using eT<sub>E</sub>X so if you use this as engine for your L<sup>A</sup>T<sub>E</sub>X format, as recommended by the L<sup>A</sup>T<sub>E</sub>X3 Project, you will get a fully functioning `\TextOrMath` command with no side effects. If you use regular T<sub>E</sub>X as engine for your L<sup>A</sup>T<sub>E</sub>X format then we have to choose between the lesser of two evils: 1) breaking ligatures and preventing kerning or 2) face the risk of choosing text-mode at the beginning of an alignment cell, which was supposed to be math-mode. We have decided upon 1) as is customary for regular robust commands in L<sup>A</sup>T<sub>E</sub>X.

## 4.5 Fewer fragile commands

```
>Number:      3816
>Category:    latex
>Synopsis:    Argument of \@sect has an extra }.
>Arrival-Date: Sat Oct 22 23:11:01 +0200 2005
>Originator:  susi@uriah.heep.sax.de (Susanne Wunsch)
```

Use of a `\raisebox` in `\section{}` produces the error message mentioned in the subject.

PR latex/1738 described a similar problem, which has been solved 10 years ago. Protecting the `\raisebox` with `\protect` solved my problem as well, but wouldn't it make sense to have a similar fix as in the PR?

It is particularly confusing, that an unprotected `\raisebox` in a `\section*-environment` works fine, while in a `\section-environment` produces error.

While not technically a bug, in this day and age there are few reasons why commands taking optional arguments should not be robust.

### 4.5.1 Notes on the implementation strategy

Rather than changing the kernel macros to be robust, we have decided to add the macro `\MakeRobust` in fixltx2e so that users can easily turn fragile macros into robust ones. A macro `\foo` is made robust by doing the simple `\MakeRobust{\foo}`. fixltx2e makes the following kernel macros robust: `\(, \)`, `\[, \]`, `\makebox`, `\savebox`, `\framebox`, `\parbox`, `\rule` and `\raisebox`.

## 5 Fixes added for 2014/05/01

### 5.1 Check the optional arguments of floats

By default LaTeX silently ignores unknown letters in the optional arguments of floats. `\begin{figure}[tB]` the B is ignored so it acts like `\begin{figure}[t]` However `\begin{figure}[B]` does *not* act like `\begin{figure}[]` as the check for an empty argument, or unsupplied argument, is earlier. `[]` causes the default float placement to be used, but `[B]` means that *no* float area is allowed and so the float will not be placed until the next `\clearpage` or end of document, no warning is given.

This package adds a check on each letter, and if it not one of `!tbhp` then an error is given and the code acts as if `p` had been used, so that the float may be placed somewhere.

### 5.2 Infinite glue found (pr/4023 and pr/2346)

The fix for pr/2346 had an issue when used in conjunction with `\enlargethispage` as the latter introduced an infinite negative glue at the bottom of the page. That in turn make a `\vsplit` operation to get at the column marks invalid.

## 6 Implementation

We require at least a somewhat sane version of LaTeX 2<sub>ε</sub>. Earlier ones where really quite different from one another.

```
1 (*fixltx2e)
2 \NeedsTeXFormat{LaTeX2e}[1996/06/01]
```

### 6.1 2-col: 1-col fig can come before earlier 2-col fig (pr/2346) Wrong headline for twocolumn (pr/2613)

Originally fixed in package `fix2col` which was merged into this package. Code and documentation are straight copies from that package.

#### 6.1.1 Preserving Marks

This is just a change to the single command `\@outputdblcol` so that it saves mark information for the first column and restores it in the second column.

```
3 \def\@outputdblcol{%
4   \if@firstcolumn
5     \global\@firstcolumnfalse
```

Save the left column

```
6   \global\setbox\@leftcolumn\copy\@outputbox
7   Remember the marks from the first column
7   \splitmaxdepth\maxdimen
8   \vbadness\maxdimen
```

In case of `\enlargethispage` we will have infinite negative glue at the bottom of the page (coming from `\vss`) and that will earn us an error message if we `\vsplit` to get at the marks. So we need to remove the last glue (if any) at the end of `\@outputbox` as we are only interested in marks that change doesn't matter.

```
9   \setbox\@outputbox\vbox{\unvbox\@outputbox\unskip}%
10  \setbox\@outputbox\vsplit\@outputbox to\maxdimen
```

One minor difference from the current `fixmarks`, pass the marks through a token register to stop any `#` tokens causing an error in a `\def`.

```
11  \toks@\expandafter{\topmark}%
12  \xdef\@firstcoltopmark{\the\toks@}%
```

```

13   \toks@\expandafter{\splitfirstmark}%
14   \xdef\@firstcolfirstmark{\the\toks@}%

```

This test does not work if truly empty marks have been inserted, but  $\text{\LaTeX}$  marks should always have (at least) two brace groups. (Except before the first mark is used, when the marks are empty, but that is OK here.)

```

15   \ifx\@firstcolfirstmark\@empty
16     \global\let\@setmarks\relax
17   \else
18     \gdef\@setmarks{%
19       \let\firstmark\@firstcolfirstmark
20       \let\topmark\@firstcoltopmark}%
21   \fi

```

End of change

```

22   \else
23     \global\@firstcolumntrue
24     \setbox\@outputbox\vbox{%
25       \hb@xt@\textwidth{%
26         \hb@xt@\columnwidth{\box\@leftcolumn \hss}%
27       \hfil

```

The color of the `\vrule` should be `\normalcolor` as to not inherit the color from the column.

```

28         {\normalcolor\vrule \@width\columnseprule}%
29       \hfil
30       \hb@xt@\columnwidth{\box\@outputbox \hss}}}%
31   \@combinedblfloats

```

Override current first and top with those of first column if necessary

```

32   \setmarks

```

End of change

```

33   \@outputpage
34   \begingroup
35     \dblfloatplacement
36     \startdblcolumn
37     \whiles\if@colmade \fi{\@outputpage\startdblcolumn}%
38   \endgroup
39   \fi}

```

### 6.1.2 Preserving Float Order

Changes `\dbldeferlist` to `\deferlist` are not explicitly noted but are flagged by blank comment lines around the changed line.

```

40 \def\enddblfloat{%
41   \if@twocolumn
42     \endfloatbox
43     \ifnum\floatpenalty <\z@
44       \largefloatcheck

```

Force the depth of two column float boxes.

```

45   \global\dp\@currbox1sp %

```

What follows is essentially `\endfloat` without a starting `\endfloatbox`.

```

46   \@cons\currlist\currbox
47   \ifnum\floatpenalty <-\@Mii
48     \penalty -\@Miv
49     \@tempdima\prevdepth
50     \vbox{}%
51     \prevdepth\@tempdima
52     \penalty\floatpenalty
53   \else
54     \adjust{\penalty -\@Miv \vbox{}\penalty\floatpenalty}\@Esphack

```

```

55     \fi
56   \else
57     \end@float
58   \fi
59 }

```

Test if the float box has the wrong width. (Actually as noted above the test is for a conventional depth setting rather than for the width of the float).

```

60 \def\@testwrongwidth #1{%
61   \ifdim\dp#1=f@depth
62   \else
63     \global\@testtrue
64   \fi}

```

Normally looking for single column floats, which have zero depth.

```

65 \let\f@depth\z@

```

but when making two column float area, look for floats with 1sp depth.

```

66 \def\@dblfloatplacement{\global\@dbltopnum\c@dbltopnumber
67   \global\@dbltoproom \dbltopfraction\@colht
68   \@textmin \@colht
69   \advance \@textmin -\@dbltoproom
70   \@fpmin \dblfloatpagefraction\textheight
71   \@fptop \dblfpptop
72   \@fpsep \dblfpsep
73   \@fpbot \dblfpbot
74   \def\f@depth{1sp}}

```

All the remaining changes are replacing the double column defer list or inserting the extra test `\@testwrongwidth{<box>}` at suitable places. That is at places where a box is taken off the deferlist.

```

75 \def \@doclearpage {%
76   \ifvoid\footins
77     \setbox\@tempboxa\vsplit\@cclv to\z@ \unvbox\@tempboxa
78     \setbox\@tempboxa\box\@cclv
79     \xdef\@deferlist{\@toplist\@botlist\@deferlist}%
80     \global \let \@toplist \@empty
81     \global \let \@botlist \@empty
82     \global \@colroom \@colht
83     \ifx \@currlist\@empty
84     \else
85       \@latexerr{Float(s) lost}\@ehb
86       \global \let \@currlist \@empty
87     \fi
88     \@makefcolumn\@deferlist
89     \@whiles\if@colmade \fi{\@opcol\@makefcolumn\@deferlist}%
90     \if@twocolumn
91       \if@firstcolumn
92         \xdef\@deferlist{\@dbltoplist\@deferlist}%
93         \global \let \@dbltoplist \@empty
94         \global \@colht \textheight
95         \begingroup
96           \@dblfloatplacement
97           \@makefcolumn\@deferlist
98           \@whiles\if@colmade \fi{\@outputpage
99             \makefcolumn\@deferlist}%
100       \endgroup
101     \else
102       \vbox{}\clearpage
103     \fi
104   \fi

```

the next line is needed to avoid losing floats in certain circumstances a single call to the original `\doclearpage` will now no longer output all floats.

```

105     \ifx\@deferlist\@empty \else\clearpage \fi
106   \else
107     \setbox\@cclv\vbox{\box\@cclv\vfil}%
108     \@makecol\@opcol
109     \clearpage
110   \fi
111 }

```

```

112 \def \@startdblcolumn {%
113   \@tryfcolumn \@deferlist
114   \if@fcolmade
115   \else
116     \begingroup
117     \let \reserved@b \@deferlist
118     \global \let \@deferlist \@empty
119     \let \@elt \@sdblcolelt
120     \reserved@b
121   \endgroup
122 \fi
123 }

```

```

124 \def\@addtonextcol{%
125   \begingroup
126   \@insertfalse
127   \@setfloattypescounts
128   \ifnum \@fpstype=8
129   \else
130     \ifnum \@fpstype=24
131     \else
132       \@flsettextmin
133       \@reqcolroom \ht\@currbox
134       \advance \@reqcolroom \@textmin
135       \ifdim \@colroom>\@reqcolroom
136         \@flsetnum \@colnum
137         \ifnum\@colnum>\z@
138           \@bitor\@currtype\@deferlist
139           \@testwrongwidth\@currbox
140           \if@test
141           \else
142             \@addtotoporbot
143           \fi
144         \fi
145       \fi
146     \fi
147   \fi
148   \if@insert
149   \else
150     \@cons\@deferlist\@currbox
151   \fi
152 \endgroup
153 }

```

```

154 \def\@addtodblcol{%
155   \begingroup
156   \@insertfalse
157   \@setfloattypescounts
158   \@getfpsbit \tw@
159   \ifodd\@tempcnta
160     \@flsetnum \@dbltopnum
161     \ifnum \@dbltopnum>\z@
162       \@tempswafalse
163       \ifdim \@dbltoproom>\ht\@currbox

```

```

164         \@tempwattrue
165     \else
166         \ifnum \@fpstype<\sist@@n
167             \advance \@dbltoproom \@textmin
168             \ifdim \@dbltoproom>\ht\@currbox
169                 \@tempwattrue
170             \fi
171             \advance \@dbltoproom -\@textmin
172         \fi
173     \fi
174     \if@tempswa
175         \@bitor \@currtype \@deferlist
not in fixfloats?
176         \@testwrongwidth\@currbox
177     \if@test
178     \else
179         \@tempdima -\ht\@currbox
180         \advance\@tempdima
181             -\ifx \@dbltoplist\@empty \dbltextfloatsep \else
182                                     \dblfloatsep \fi
183         \global \advance \@dbltoproom \@tempdima
184         \global \advance \@colht \@tempdima
185         \global \advance \@dbltopnum \m@ne
186         \@cons \@dbltoplist \@currbox
187         \@inserttrue
188     \fi
189 \fi
190 \fi
191 \fi
192 \if@insert
193 \else
194     \@cons\@deferlist\@currbox
195 \fi
196 \endgroup
197 }

```

```

198 \def \@addtocurcol {%
199     \@insertfalse
200     \@setfloattypecounts
201     \ifnum \@fpstype=8
202     \else
203         \ifnum \@fpstype=24
204         \else
205             \@flsettextmin
206             \advance \@textmin \@textfloatsheight
207             \@reqcolroom \@pageht
208             \ifdim \@textmin>\@reqcolroom
209                 \@reqcolroom \@textmin
210             \fi
211             \advance \@reqcolroom \ht\@currbox
212             \ifdim \@colroom>\@reqcolroom
213                 \@flsetnum \@colnum
214                 \ifnum \@colnum>\z@
215                     \@bitor\@currtype\@deferlist

```

We need to defer the float also if its width doesn't fit.

```

216         \@testwrongwidth\@currbox
217     \if@test
218     \else
219         \@bitor\@currtype\@botlist
220     \if@test
221         \@addtobot

```

```

222         \else
223         \ifodd \count\@currbox
224         \advance \@reqcolroom \intextsep
225         \ifdim \@colroom>\@reqcolroom
226         \global \advance \@colnum \m@ne
227         \global \advance \@textfloatsheight \ht\@currbox
228         \global \advance \@textfloatsheight 2\intextsep
229         \@cons \@midlist \@currbox
230         \if@nobreak
231         \nobreak
232         \@nobreakfalse
233         \everypar{}%
234         \else
235         \addpenalty \interlinepenalty
236         \fi
237         \vskip \intextsep
238         \box\@currbox
239         \penalty\interlinepenalty
240         \vskip\intextsep
241         \ifnum\outputpenalty <-\@Mii \vskip -\parskip\fi
242         \outputpenalty \z@
243         \@inserttrue
244         \fi
245         \fi
246         \if@insert
247         \else
248         \@addtotoporbot
249         \fi
250         \fi
251         \fi
252         \fi
253         \fi
254         \fi
255         \fi
256         \if@insert
257         \else
258         \@resetfyps
259         \@cons\@deferlist\@currbox
260         \fi
261 }

262 \def\@xtryfc #1{%
263   \@next\reserved@a\@trylist{ }{ }%
264   \@currtype \count #1%
265   \divide\@currtype\@xxxii
266   \multiply\@currtype\@xxxii
267   \@bitor \@currtype \@failedlist
268   \@testfp #1%

269   \@testwrongwidth #1%

270   \ifdim \ht #1>\@colht
271     \@testtrue
272   \fi
273   \if@test
274     \@cons\@failedlist #1%
275   \else
276     \@ytryfc #1%
277   \fi}

278 \def\@ztryfc #1{%
279   \@tempcnta\count #1%
280   \divide\@tempcnta\@xxxii
281   \multiply\@tempcnta\@xxxii
282   \@bitor \@tempcnta {\@failedlist \@flfail}%

```



```

283 \testfp #1%
    not in fixfloats?
284 \testwrongwidth #1%
285 \tempdimb\tempdima
286 \advance\tempdimb\ht #1%
287 \advance\tempdimb\fpsep
288 \ifdim \tempdimb >\colht
289 \testtrue
290 \fi
291 \iftest
292 \cons\flfail #1%
293 \else
294 \cons\flsucceed #1%
295 \tempdima\tempdimb
296 \fi}

```

## 6.2 \@ discards spaces when moving (pr3039)

\@ Ensure that \@m can't eat spaces. Alternative would be to make \@ robust but that takes more space.

```

297 \def\@{\spacefactor\@m{}}

```

## 6.3 \setlength produces error if used with registers like \dimen0 (pr/3066)

\setlength Add space after register (#1) but only if this is still the original definition. When, for example, calc was already loaded this wouldn't be a good idea any more.

```

298 \def\@tempa#1#2{#1#2\relax}
299 \ifx\setlength\@tempa
300 \def\setlength#1#2{#1 #2\relax}
301 \fi

```

## 6.4 \addpenalty ruins flush-bottom (pr/3073)

\addpenalty Fix provided by Donald (though the original fix was not good enough). In 2005 Plamen Tanovski discovered that this fix wasn't good enough either as the \vskip kept getting bigger if several \addpenalty commands followed each other. Donald kindly send a new fix.

```

302 \def\addpenalty#1{%
303 \ifvmode
304 \if@minipage
305 \else
306 \if@nobreak
307 \else
308 \ifdim\lastskip=\z@
309 \penalty#1\relax
310 \else
311 \@tempskipb\lastskip

```

We have to make sure the final \vskip seen by TeX is the correct one, namely \@tempskipb. However we may have to adjust for \prevdepth when placing the penalty but that should not affect the skip we pass on to TeX.

```

312 \begingroup
313 \advance \@tempskipb
314 \ifdim\prevdepth>\maxdepth\maxdepth\else

```

If \prevdepth is -1000pt due to \nointerlineskip we better not add it!

```

315 \ifdim \prevdepth = -\@m\p@ \z@ \else \prevdepth \fi
316 \fi
317 \vskip -\@tempskipb

```

```

318         \penalty#1%
319         \vskip\@tempskipb
320     \endgroup
321     \vskip -\@tempskipb
322     \vskip \@tempskipb
323 \fi
324 \fi
325 \fi
326 \else
327     \@noitemerr
328 \fi}

```

## 6.5 \fnsymbol should use text symbols if possible (pr/3400)

**\fnsymbol** This macro is another example of an ever recurring problem in  $\TeX$ : Determining if something is text-mode or math-mode. It is imperative for the decision between text and math to be delayed until the actual typesetting is done as the code in question may go through an `\edef` or `\write` where an `\ifmmode` test would be executed prematurely. Hence in the implementation below, `\fnsymbol` is not robust in itself but the parts doing the actual typesetting are.

In the case of `\fnsymbol` we make use of the robust command `\TextOrMath` which takes two arguments and typesets the first if in text-mode and the second if in math-mode. Note that in order for this command to make the correct decision, it must insert a `\relax` token if run under regular  $\TeX$ , which ruins any kerning between the preceding characters and whatever awaits typesetting. If you use  $\text{\texttt{e}\TeX}$  as engine for  $\text{\texttt{L}\TeX}$  (as recommended) this unfortunate side effect is not present.

```

329 \def\fnsymbol#1{%
330     \ifcase#1\or \TextOrMath\textasteriskcentered *\or
331     \TextOrMath \textdagger \dagger\or
332     \TextOrMath \textdaggerdbl \ddagger \or
333     \TextOrMath \textsection \mathsection\or
334     \TextOrMath \textparagraph \mathparagraph\or
335     \TextOrMath \textbardbl |\or
336     \TextOrMath {\textasteriskcentered\textasteriskcentered}{**}\or
337     \TextOrMath {\textdagger\textdagger}{\dagger\dagger}\or
338     \TextOrMath {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}\else
339     \@ctrerr \fi
340 }

```

**\TextOrMath** When using regular  $\TeX$ , we make this command robust so that it always selects the correct branch in an `\ifmmode` switch with the usual disadvantage of ruining kerning. For the application we use it for here that shouldn't matter. The alternative would be to mimic `\IeC` from `inputenc` but then it will have the disadvantage of choosing the wrong branch if appearing at the beginning of an alignment cell. However, users of  $\text{\texttt{e}\TeX}$  will be pleasantly surprised to get the best of both worlds and no bad side effects.

First some code for checking if we are running  $\text{\texttt{e}\TeX}$  but making sure not to permanently turn `\eTeXversion` into `\relax`.

```

341 \begingroup\expandafter\expandafter\expandafter\endgroup
342 \expandafter\ifx\csname eTeXversion\endcsname\relax

```

In case of ordinary  $\TeX$  we define `\TextOrMath` as a robust command but make sure it always grabs its arguments. If we didn't do this it might very well gobble spaces in the input stream.

```

343 \DeclareRobustCommand\TextOrMath{%
344     \ifmmode \expandafter\@secondoftwo
345     \else \expandafter\@firstoftwo \fi}
346 \protected@edef\TextOrMath#1#2{\TextOrMath{#1}{#2}}
347 \else

```

For eTeX the situation is similar. The robust macro is a hidden one so that we again avoid problems of gobbling spaces in the input.

```

348 \protected\expandafter\def\csname TextOrMath\space\endcsname{%
349   \ifmode   \expandafter\@secondoftwo
350   \else     \expandafter\@firstoftwo \fi}
351 \edef\TextOrMath#1#2{%
352   \expandafter\noexpand\csname TextOrMath\space\endcsname
353   {#1}{#2}}
354 \fi

```

## 6.6 No hyphenation in first word after float environment (pr/3498)

`\@esphack` Fix suggested by Donald Arseneau.

```

\@Esphack 355 \def\@esphack{%
356   \relax
357   \ifhmode
358     \spacefactor\@savsf
359     \ifdim\@savsk>\z@
360       \nobreak \hskip\z@skip % <-----
361       \ignorespaces
362     \fi
363   \fi}

364 \def\@Esphack{%
365   \relax
366   \ifhmode
367     \spacefactor\@savsf
368     \ifdim\@savsk>\z@
369       \nobreak \hskip\z@skip % <-----
370       \@ignoretrue
371       \ignorespaces
372     \fi
373   \fi}

```

## 6.7 Allowing `\emph` to produce small caps, etc

```

\em
\emminnershape 374 \DeclareRobustCommand\em
375                 {\@nomath\em \ifdim \fontdimen\@ne\font >\z@
376                 \emminnershape \else \itshape \fi}
377 \def\emminnershape{\upshape}

```

## 6.8 `\textsubscript` not defined in latex.ltx (pr/3492)

`\textsubscript` This macro is almost identical to `\textsuperscript` from the kernel.

```

378 \DeclareRobustCommand*\textsubscript[1]{%
379   \@textsubscript{\selectfont#1}}
380 \def\@textsubscript#1{%
381   {\m@th\ensuremath_{\mbox{\fontsize\sf@size\z@#1}}}}

```

## 6.9 `\DeclareMathSizes` only take pts. (pr/3693)

`\@DeclareMathSizes` This fix given by Michael J. Downes on comp.text.tex on 2002/10/17 allows the user to have settings such as `\DeclareMathSizes{9.5dd}{9.5dd}{7.4dd}{6.6dd}`.

```

382 \def\@DeclareMathSizes #1#2#3#4#5{%
383   \@defaultunits\dimen@ #2pt\relax\@nnil
384   \if $3$%
385     \expandafter\let\csname S@\strip@pt\dimen@\endcsname\math@fontsfalse
386   \else
387     \@defaultunits\dimen@ii #3pt\relax\@nnil
388     \@defaultunits\@tempdima #4pt\relax\@nnil

```

```

389 \defaultunits\@tempdimb #5pt\relax\@nnil
390 \toks@{#1}%
391 \expandafter\xdef\csname S@\strip@pt\dimen@\endcsname{%
392 \gdef\noexpand\tf@size{\strip@pt\dimen@ii}%
393 \gdef\noexpand\s@size{\strip@pt\@tempdima}%
394 \gdef\noexpand\ssf@size{\strip@pt\@tempdimb}%
395 \the\toks@
396 }%
397 \fi
398 }

```

## 6.10 Fewer fragile macros

`\MakeRobust` The macro firstly checks if the controls sequence in question exists at all.

```

399 \providecommand*\MakeRobust[1]{%
400 \ifundefined{\expandafter\@gobble\string#1}{%
401 \latex@error{The control sequence '\string#1' is undefined!%
402 \MessageBreak There is nothing here to make robust}%
403 \@eha
404 }%

```

Then we check if the macro is already robust. We do this by testing if the internal name for a robust macro is defined, namely `\foo_`. If it is already defined do nothing, otherwise set `\foo_` equal to `\foo` and redefine `\foo` so that it acts like a macro defined with `\DeclareRobustCommand`.

```

405 {%
406 \ifundefined{\expandafter\@gobble\string#1\space}%
407 {%
408 \expandafter\let\csname
409 \expandafter\@gobble\string#1\space\endcsname=#1%
410 \edef\reserved@a{\string#1}%
411 \def\reserved@b{#1}%
412 \edef\reserved@b{\expandafter\strip@prefix\meaning\reserved@b}%
413 \edef#1{%
414 \ifx\reserved@a\reserved@b
415 \noexpand\x@protect\noexpand#1%
416 \fi
417 \noexpand\protect\expandafter\noexpand
418 \csname\expandafter\@gobble\string#1\space\endcsname}%
419 }%
420 {\latex@info{The control sequence '\string#1' is already robust}}%
421 }%
422 }

```

Here we make some kernel macros robust.

```

423 \MakeRobust\(  

424 \MakeRobust\  

425 \MakeRobust\  

426 \MakeRobust\  

427 \MakeRobust\makebox  

428 \MakeRobust\savebox  

429 \MakeRobust\framebox  

430 \MakeRobust\parbox  

431 \MakeRobust\rule  

432 \MakeRobust\raisebox  

433 \fixltx2e)

```

## 6.11 Using EC fonts (T1 encoding) makes my documents look bl\*\*dy horrible

### 6.11.1 Preliminaries

The L<sup>A</sup>T<sub>E</sub>X kernel does not declare the font encoding TS1. However, we are going to set up font definitions for this encoding, so we have to declare it now.

```
434 \fix-cm
435 \input{ts1enc.def}
```

In case the package is loaded in the preamble, any of the CM fonts may have been used already and cannot be redefined. Yet we try to intercept at least the problem that is most likely to occur, i.e., a hidden `\normalfont`. Most of the standard definitions are ok, but those for T1 encoding and 10.95pt need to be removed:

```
436 \expandafter \let \csname T1/cmr/m/n/10.95\endcsname \relax
437 \expandafter \let \csname T1/cmss/m/n/10.95\endcsname \relax
438 \expandafter \let \csname T1/cmtt/m/n/10.95\endcsname \relax
439 \expandafter \let \csname T1/cmvtt/m/n/10.95\endcsname \relax
```

`fix-cm` may still fail, if the EC fonts are preloaded in the L<sup>A</sup>T<sub>E</sub>X format file. This situation is, however, very unlikely and could occur only with a customized format.

The remainder of the package is enclosed in a group, where the catcodes are guaranteed to be appropriate for the processing of font definitions.

```
440 \begingroup
441 \nfss@catcodes
```

### 6.11.2 T1 encoding

#### CM Roman

```
442 \DeclareFontFamily{T1}{cmr}{}
443 \DeclareFontShape{T1}{cmr}{m}{n}{
444     <-6>      ecrm0500
445     <6-7>     ecrm0600
446     <7-8>     ecrm0700
447     <8-9>     ecrm0800
448     <9-10>    ecrm0900
449     <10-12>   ecrm1000
450     <12-17>   ecrm1200
451     <17->     ecrm1728
452 }{}
453 \DeclareFontShape{T1}{cmr}{m}{sl}{
454     <-6>      ecsl0500
455     <6-7>     ecsl0600
456     <7-8>     ecsl0700
457     <8-9>     ecsl0800
458     <9-10>    ecsl0900
459     <10-12>   ecsl1000
460     <12-17>   ecsl1200
461     <17->     ecsl1728
462 }{}
463 \DeclareFontShape{T1}{cmr}{m}{it}{
464     <-8>      ecti0700
465     <8-9>     ecti0800
466     <9-10>    ecti0900
467     <10-12>   ecti1000
468     <12-17>   ecti1200
469     <17->     ecti1728
470 }{}
471 \DeclareFontShape{T1}{cmr}{m}{sc}{
472     <-6>      eccc0500
```

```

473         <6-7>      eccc0600
474         <7-8>      eccc0700
475         <8-9>      eccc0800
476         <9-10>     eccc0900
477         <10-12>    eccc1000
478         <12-17>    eccc1200
479         <17->      eccc1728
480         }{}
481 \DeclareFontShape{T1}{cmr}{m}{ui}{
482         <-8>      ecui0700
483         <8-9>      ecui0800
484         <9-10>     ecui0900
485         <10-12>    ecui1000
486         <12-17>    ecui1200
487         <17->      ecui1728
488         }{}
489 \DeclareFontShape{T1}{cmr}{b}{n}{
490         <-6>      ecrb0500
491         <6-7>      ecrb0600
492         <7-8>      ecrb0700
493         <8-9>      ecrb0800
494         <9-10>     ecrb0900
495         <10-12>    ecrb1000
496         <12-17>    ecrb1200
497         <17->      ecrb1728
498         }{}
499 \DeclareFontShape{T1}{cmr}{bx}{n}{
500         <-6>      ecbx0500
501         <6-7>      ecbx0600
502         <7-8>      ecbx0700
503         <8-9>      ecbx0800
504         <9-10>     ecbx0900
505         <10-12>    ecbx1000
506         <12->      ecbx1200
507         }{}
508 \DeclareFontShape{T1}{cmr}{bx}{sl}{
509         <-6>      ecbl0500
510         <6-7>      ecbl0600
511         <7-8>      ecbl0700
512         <8-9>      ecbl0800
513         <9-10>     ecbl0900
514         <10-12>    ecbl1000
515         <12->      ecbl1200
516         }{}
517 \DeclareFontShape{T1}{cmr}{bx}{it}{
518         <-8>      ecbi0700
519         <8-9>      ecbi0800
520         <9-10>     ecbi0900
521         <10-12>    ecbi1000
522         <12->      ecbi1200
523         }{}
524 \DeclareFontShape{T1}{cmr}{bx}{sc}{
525         <-6>      ecx0500
526         <6-7>      ecx0600
527         <7-8>      ecx0700
528         <8-9>      ecx0800
529         <9-10>     ecx0900
530         <10-12>    ecx1000
531         <12->      ecx1200
532         }{}
533 %

```

## CM Sans

```
534 \DeclareFontFamily{T1}{cmss}{}
535 \DeclareFontShape{T1}{cmss}{m}{n}{
536     <-9>      ecss0800
537     <9-10>    ecss0900
538     <10-12>   ecss1000
539     <12-17>   ecss1200
540     <17->     ecss1728
541 }{}
542 \DeclareFontShape{T1}{cmss}{m}{sl}{
543     <-9>      ecsi0800
544     <9-10>    ecsi0900
545     <10-12>   ecsi1000
546     <12-17>   ecsi1200
547     <17->     ecsi1728
548 }{}
549 \DeclareFontShape{T1}{cmss}{m}{it}
550     {<->ssub*cmss/m/sl}{}
551 \DeclareFontShape{T1}{cmss}{m}{sc}
552     {<->sub*cmr/m/sc}{}
553 \DeclareFontShape{T1}{cmss}{sbc}{n}{
554     <->      ecssdc10
555 }{}
556 \DeclareFontShape{T1}{cmss}{bx}{n}{
557     <-10>     ecsx0900
558     <10->     ecsx1000
559 }{}
560 \DeclareFontShape{T1}{cmss}{bx}{sl}{
561     <-10>     ecso0900
562     <10->     ecso1000
563 }{}
564 \DeclareFontShape{T1}{cmss}{bx}{it}
565     {<->ssub*cmss/bx/sl}{}

```

The following substitutions are not provided in the default .fd files. I have included them, so that you can easily use the EC fonts with the default bold series being **b** rather than **bx**.

```
566 \DeclareFontShape{T1}{cmss}{b}{n}
567     {<->ssub*cmss/bx/n}{}
568 \DeclareFontShape{T1}{cmss}{b}{sl}
569     {<->ssub*cmss/bx/sl}{}
570 \DeclareFontShape{T1}{cmss}{b}{it}
571     {<->ssub*cmss/bx/sl}{}

```

## CM Typewriter

```
572 \DeclareFontFamily{T1}{cmtt}{\hyphenchar \font\m@ne}
573 \DeclareFontShape{T1}{cmtt}{m}{n}{
574     <-9>      ectt0800
575     <9-10>    ectt0900
576     <10-12>   ectt1000
577     <12-17>   ectt1200
578     <17->     ectt1728
579 }{}
580 \DeclareFontShape{T1}{cmtt}{m}{it}{
581     <-9>      ecit0800
582     <9-10>    ecit0900
583     <10-12>   ecit1000
584     <12-17>   ecit1200
585     <17->     ecit1728
586 }{}
587 \DeclareFontShape{T1}{cmtt}{m}{sl}{
588     <-9>      ecst0800

```

```

589      <9-10>  ecst0900
590      <10-12> ecst1000
591      <12-17> ecst1200
592      <17->   ecst1728
593    }{}
594 \DeclareFontShape{T1}{cmtt}{m}{sc}{
595      <-9>      ectc0800
596      <9-10>    ectc0900
597      <10-12>   ectc1000
598      <12-17>   ectc1200
599      <17->     ectc1728
600    }{}
601 \DeclareFontShape{T1}{cmtt}{bx}{n}{
602     {<->sub * cmtt/m/n}{}
603 \DeclareFontShape{T1}{cmtt}{bx}{it}{
604     {<->sub * cmtt/m/it}{}
605 \DeclareFontShape{T1}{cmtt}{bx}{sl}{
606     {<->sub * cmtt/m/sl}{}

```

Substitutions not provided in the default .fd files:

```

607 \DeclareFontShape{T1}{cmtt}{b}{n}{
608     {<->sub * cmtt/m/n}{}
609 \DeclareFontShape{T1}{cmtt}{b}{it}{
610     {<->sub * cmtt/m/it}{}
611 \DeclareFontShape{T1}{cmtt}{b}{sl}{
612     {<->sub * cmtt/m/sl}{}

```

### CM Typewriter (var.)

```

613 \DeclareFontFamily{T1}{cmvtt}{}
614 \DeclareFontShape{T1}{cmvtt}{m}{n}{
615     <-9>      ecvt0800
616     <9-10>    ecvt0900
617     <10-12>   ecvt1000
618     <12-17>   ecvt1200
619     <17->     ecvt1728
620   }{}
621 \DeclareFontShape{T1}{cmvtt}{m}{it}{
622     <-9>      ecvi0800
623     <9-10>    ecvi0900
624     <10-12>   ecvi1000
625     <12-17>   ecvi1200
626     <17->     ecvi1728
627   }{}

```

### 6.11.3 TS1 encoding

#### CM Roman

```

628 \DeclareFontFamily{TS1}{cmr}{\hyphenchar\font\m@ne}
629 \DeclareFontShape{TS1}{cmr}{m}{n}{
630     <-6>      tcrm0500
631     <6-7>     tcrm0600
632     <7-8>     tcrm0700
633     <8-9>     tcrm0800
634     <9-10>    tcrm0900
635     <10-12>   tcrm1000
636     <12-17>   tcrm1200
637     <17->     tcrm1728
638   }{}
639 \DeclareFontShape{TS1}{cmr}{m}{sl}{
640     <-6>      tcs10500
641     <6-7>     tcs10600
642     <7-8>     tcs10700

```



```

643         <8-9>    tcs10800
644         <9-10>   tcs10900
645         <10-12>  tcs11000
646         <12-17> tcs11200
647         <17->    tcs11728
648     }{}
649 \DeclareFontShape{TS1}{cmr}{m}{it}{
650     <-8>        tcti0700
651     <8-9>       tcti0800
652     <9-10>      tcti0900
653     <10-12>     tcti1000
654     <12-17>     tcti1200
655     <17->       tcti1728
656 }{}
657 \DeclareFontShape{TS1}{cmr}{m}{ui}{
658     <-8>        tcui0700
659     <8-9>       tcui0800
660     <9-10>      tcui0900
661     <10-12>     tcui1000
662     <12-17>     tcui1200
663     <17->       tcui1728
664 }{}
665 \DeclareFontShape{TS1}{cmr}{b}{n}{
666     <-6>        tcrb0500
667     <6-7>       tcrb0600
668     <7-8>       tcrb0700
669     <8-9>       tcrb0800
670     <9-10>      tcrb0900
671     <10-12>     tcrb1000
672     <12-17>     tcrb1200
673     <17->       tcrb1728
674 }{}
675 \DeclareFontShape{TS1}{cmr}{bx}{n}{
676     <-6>        tcbx0500
677     <6-7>       tcbx0600
678     <7-8>       tcbx0700
679     <8-9>       tcbx0800
680     <9-10>      tcbx0900
681     <10-12>     tcbx1000
682     <12->       tcbx1200
683 }{}
684 \DeclareFontShape{TS1}{cmr}{bx}{sl}{
685     <-6>        tcbl0500
686     <6-7>       tcbl0600
687     <7-8>       tcbl0700
688     <8-9>       tcbl0800
689     <9-10>      tcbl0900
690     <10-12>     tcbl1000
691     <12->       tcbl1200
692 }{}
693 \DeclareFontShape{TS1}{cmr}{bx}{it}{
694     <-8>        tcbi0700
695     <8-9>       tcbi0800
696     <9-10>      tcbi0900
697     <10-12>     tcbi1000
698     <12->       tcbi1200
699 }{}

```

## CM Sans

```

700 \DeclareFontFamily{TS1}{cmss}{\hyphenchar\font\m@ne}
701 \DeclareFontShape{TS1}{cmss}{m}{n}{
702     <-9>        tcss0800

```

```

703      <9-10>  tcss0900
704      <10-12> tcss1000
705      <12-17> tcss1200
706      <17->   tcss1728
707    }{}
708 \DeclareFontShape{TS1}{cmss}{m}{it}{
709     {<->ssub*cmss/m/sl}{}
710 \DeclareFontShape{TS1}{cmss}{m}{sl}{
711     <-9>      tcsi0800
712     <9-10>   tcsi0900
713     <10-12>  tcsi1000
714     <12-17> tcsi1200
715     <17->   tcsi1728
716   }{}
717 \DeclareFontShape{TS1}{cmss}{sbc}{n}{
718     <->      tcssdc10
719   }{}
720 \DeclareFontShape{TS1}{cmss}{bx}{n}{
721     <-10>    tcsx0900
722     <10->    tcsx1000
723   }{}
724 \DeclareFontShape{TS1}{cmss}{bx}{sl}{
725     <-10>    tcso0900
726     <10->    tcso1000
727   }{}
728 \DeclareFontShape{TS1}{cmss}{bx}{it}{
729     {<->ssub*cmss/bx/sl}{}

```

Substitutions not provided in the default .fd files:

```

730 \DeclareFontShape{TS1}{cmss}{b}{n}{
731     {<->ssub*cmss/bx/n}{}
732 \DeclareFontShape{TS1}{cmss}{b}{sl}{
733     {<->ssub*cmss/bx/sl}{}
734 \DeclareFontShape{TS1}{cmss}{b}{it}{
735     {<->ssub*cmss/bx/sl}{}

```

## CM Typewriter

```

736 \DeclareFontFamily{TS1}{cmtt}{\hyphenchar \font\m@ne}
737 \DeclareFontShape{TS1}{cmtt}{m}{n}{
738     <-9>      tctt0800
739     <9-10>   tctt0900
740     <10-12>  tctt1000
741     <12-17>  tctt1200
742     <17->   tctt1728
743   }{}
744 \DeclareFontShape{TS1}{cmtt}{m}{it}{
745     <-9>      tcit0800
746     <9-10>   tcit0900
747     <10-12>  tcit1000
748     <12-17>  tcit1200
749     <17->   tcit1728
750   }{}
751 \DeclareFontShape{TS1}{cmtt}{m}{sl}{
752     <-9>      tcst0800
753     <9-10>   tcst0900
754     <10-12>  tcst1000
755     <12-17>  tcst1200
756     <17->   tcst1728
757   }{}
758 \DeclareFontShape{TS1}{cmtt}{bx}{n}{
759     {<->sub * cmtt/m/n}{}
760 \DeclareFontShape{TS1}{cmtt}{bx}{it}{

```

```

761      {<->sub * cmtt/m/it}{ }
762 \DeclareFontShape{TS1}{cmtt}{bx}{sl}{
763      {<->sub * cmtt/m/sl}{ }

```

Substitutions not provided in the default .fd files:

```

764 \DeclareFontShape{TS1}{cmtt}{b}{n}{
765      {<->sub * cmtt/m/n}{ }
766 \DeclareFontShape{TS1}{cmtt}{b}{it}{
767      {<->sub * cmtt/m/it}{ }
768 \DeclareFontShape{TS1}{cmtt}{b}{sl}{
769      {<->sub * cmtt/m/sl}{ }

```

### CM Typewriter (var.)

```

770 \DeclareFontFamily{TS1}{cmvtt}{ }
771 \DeclareFontShape{TS1}{cmvtt}{m}{n}{
772      <-9>      tcvt0800
773      <9-10>    tcvt0900
774      <10-12>   tcvt1000
775      <12-17>   tcvt1200
776      <17->     tcvi1728
777      }{ }
778 \DeclareFontShape{TS1}{cmvtt}{m}{it}{
779      <-9>      tcvi0800
780      <9-10>    tcvi0900
781      <10-12>   tcvi1000
782      <12-17>   tcvi1200
783      <17->     tcvi1728
784      }{ }

```

### 6.11.4 OT1 encoding

#### CM Roman

```

785 \DeclareFontFamily{OT1}{cmr}{\hyphenchar\font45 }
786 \DeclareFontShape{OT1}{cmr}{m}{n}{
787      <-6>      cmr5
788      <6-7>     cmr6
789      <7-8>     cmr7
790      <8-9>     cmr8
791      <9-10>    cmr9
792      <10-12>   cmr10
793      <12-17>   cmr12
794      <17->     cmr17
795      }{ }
796 \DeclareFontShape{OT1}{cmr}{m}{sl}{
797      <-9>      cmsl8
798      <9-10>    cmsl9
799      <10-12>   cmsl10
800      <12->     cmsl12
801      }{ }
802 \DeclareFontShape{OT1}{cmr}{m}{it}{
803      <-8>      cmti7
804      <8-9>     cmti8
805      <9-10>    cmti9
806      <10-12>   cmti10
807      <12->     cmti12
808      }{ }
809 \DeclareFontShape{OT1}{cmr}{m}{sc}{
810      <->      cmcsc10
811      }{ }
812 \DeclareFontShape{OT1}{cmr}{m}{ui}{
813      <->      cmu10
814      }{ }

```

```

815 \DeclareFontShape{OT1}{cmr}{b}{n}{
816     <->      cmb10
817 }{}
818 \DeclareFontShape{OT1}{cmr}{bx}{n}{
819     <-6>      cmbx5
820     <6-7>     cmbx6
821     <7-8>     cmbx7
822     <8-9>     cmbx8
823     <9-10>    cmbx9
824     <10-12>   cmbx10
825     <12->     cmbx12
826 }{}
827 \DeclareFontShape{OT1}{cmr}{bx}{sl}{
828     <->      cmbxsl10
829 }{}
830 \DeclareFontShape{OT1}{cmr}{bx}{it}{
831     <->      cmbxti10
832 }{}
833 \DeclareFontShape{OT1}{cmr}{bx}{ui}
834     {<->sub*cmr/m/ui}{}

```

## CM Sans

```

835 \DeclareFontFamily{OT1}{cmss}{\hyphenchar\font45 }
836 \DeclareFontShape{OT1}{cmss}{m}{n}{
837     <-9>      cmss8
838     <9-10>    cmss9
839     <10-12>   cmss10
840     <12-17>  cmss12
841     <17->    cmss17
842 }{}
843 \DeclareFontShape{OT1}{cmss}{m}{it}
844     {<->sub*cmss/m/sl}{}
845 \DeclareFontShape{OT1}{cmss}{m}{sl}{
846     <-9>      cmssi8
847     <9-10>    cmssi9
848     <10-12>   cmssi10
849     <12-17>   cmssi12
850     <17->    cmssi17
851 }{}
852 \DeclareFontShape{OT1}{cmss}{m}{sc}
853     {<->sub*cmr/m/sc}{}
854 \DeclareFontShape{OT1}{cmss}{m}{ui}
855     {<->sub*cmr/m/ui}{}
856 \DeclareFontShape{OT1}{cmss}{sbc}{n}{
857     <->      cmssdc10
858 }{}
859 \DeclareFontShape{OT1}{cmss}{bx}{n}{
860     <->      cmssbx10
861 }{}
862 \DeclareFontShape{OT1}{cmss}{bx}{ui}
863     {<->sub*cmr/bx/ui}{}

```

## CM Typewriter

```

864 \DeclareFontFamily{OT1}{cmtt}{\hyphenchar\font\m@ne}
865 \DeclareFontShape{OT1}{cmtt}{m}{n}{
866     <-9>      cmtt8
867     <9-10>    cmtt9
868     <10-12>   cmtt10
869     <12->     cmtt12
870 }{}
871 \DeclareFontShape{OT1}{cmtt}{m}{it}{

```

```

872      <->      cmitt10
873      }{}
874 \DeclareFontShape{OT1}{cmitt}{m}{sl}{
875      <->      cmslitt10
876      }{}
877 \DeclareFontShape{OT1}{cmitt}{m}{sc}{
878      <->      cmisc10
879      }{}
880 \DeclareFontShape{OT1}{cmitt}{m}{ui}
881      {<->ssub*cmitt/m/it}{}
882 \DeclareFontShape{OT1}{cmitt}{bx}{n}
883      {<->ssub*cmitt/m/n}{}
884 \DeclareFontShape{OT1}{cmitt}{bx}{it}
885      {<->ssub*cmitt/m/it}{}
886 \DeclareFontShape{OT1}{cmitt}{bx}{ui}
887      {<->ssub*cmitt/m/it}{}

```

### CM Typewriter (var.)

```

888 \DeclareFontFamily{OT1}{cmvtt}{\hyphenchar\font45 }
889 \DeclareFontShape{OT1}{cmvtt}{m}{n}{
890      <->      cmvtt10
891      }{}
892 \DeclareFontShape{OT1}{cmvtt}{m}{it}{
893      <->      cmvtti10
894      }{}

```

### 6.11.5 OML and OMS encoded math fonts

```

895 \DeclareFontFamily{OML}{cmm}{\skewchar\font127 }
896 \DeclareFontShape{OML}{cmm}{m}{it}{
897      <-6>      cmmi5
898      <6-7>     cmmi6
899      <7-8>     cmmi7
900      <8-9>     cmmi8
901      <9-10>    cmmi9
902      <10-12>   cmmi10
903      <12->     cmmi12
904      }{}
905 \DeclareFontShape{OML}{cmm}{b}{it}{<-6>cmmib5<6-8>cmmib7<8->cmmib10}{}
906 \DeclareFontShape{OML}{cmm}{bx}{it}
907      {<->ssub*cmm/b/it}{}
908 \DeclareFontFamily{OMS}{cmsy}{\skewchar\font48 }
909 \DeclareFontShape{OMS}{cmsy}{m}{n}{
910      <-6>      cmsy5
911      <6-7>     cmsy6
912      <7-8>     cmsy7
913      <8-9>     cmsy8
914      <9-10>    cmsy9
915      <10->     cmsy10
916      }{}
917 \DeclareFontShape{OMS}{cmsy}{b}{n}{<-6>cmsyb5<6-8>cmsyb7<8->cmsyb10}{}

```

### 6.11.6 L<sup>A</sup>T<sub>E</sub>X symbols

```

918 \DeclareFontFamily{U}{lasy}{}
919 \DeclareFontShape{U}{lasy}{m}{n}{
920      <-6>      lasy5
921      <6-7>     lasy6
922      <7-8>     lasy7
923      <8-9>     lasy8
924      <9-10>    lasy9
925      <10->     lasy10
926      }{}

```

```

927 \DeclareFontShape{U}{lasy}{b}{n}{
928     <-10>    ssub * lasy/m/n
929     <10->    lasyb10
930 }{}
931 \endgroup
932 </fix-cm>

```

## 6.12 Check the optional argument to floats

The default definition of `\@xfloat` allows `\begin{figure}[abt23WD]` silently ignoring all but `t`. If you use `\begin{figure}[T]` you get no warning but the float is not allowed *anywhere* so will go to the end of document (or `\clearpage`). This change gives an error message for undefined options.

```

933 \def\@xfloat #1[#2]{%
934   \@nodocument
935   \def \@capttype {#1}%
936   \def \@fps {#2}%
937   \@onelevel@sanitize \@fps
938   \def \reserved@b {}%
939   \ifx \reserved@b \@fps
940     \@fpsadddefault
941   \else
942     \ifx \@fps \@empty
943       \@fpsadddefault
944     \fi
945   \fi
946   \ifhmode
947     \@bsphack
948     \@floatpenalty -\@Mii
949   \else
950     \@floatpenalty-\@Miii
951   \fi
952   \ifinner
953     \@parmoderr\@floatpenalty\z@
954   \else
955     \@next\@currbox\@freelist
956     {%
957       \@tempcnta \sixt@@n
958       \expandafter \@tfor \expandafter \reserved@a
959       \expandafter :\expandafter =\@fps
960       \do

```

Start of changes, use a nested if structure, ending in an error.

```

961       {%
962         \if \reserved@a h%
963           \ifodd \@tempcnta
964             \else
965               \advance \@tempcnta \@ne
966             \fi
967         \else\if \reserved@a t%
968           \@setfpsbit \tw@
969         \else\if \reserved@a b%
970           \@setfpsbit 4%
971         \else\if \reserved@a p%
972           \@setfpsbit 8%
973         \else\if \reserved@a !%
974           \ifnum \@tempcnta>15
975             \advance\@tempcnta -\sixt@@n\relax
976           \fi
977         \else
978           \latex@error{Unknown float option `\'reserved@a'}%
979           {Option `\'reserved@a' ignored and `p' used.}%

```

```

980          \@setfpsbit 8%
981          \fi\fi\fi\fi\fi
982      }%
End of changes
983      \@tempcntb \csname ftype@\@capttype \endcsname
984      \multiply \@tempcntb \@xxxii
985      \advance \@tempcnta \@tempcntb
986      \global \count\@currbox \@tempcnta
987      }%
988      \@fltovf
989  \fi
990  \global \setbox\@currbox
991      \color@vbox
992      \normalcolor
993      \vbox \bgroup
994          \hsize\columnwidth
995          \@parboxrestore
996          \@floatboxreset
997  }

```