The fixltx2e and fix-cm packages[*]

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

2014/09/29

Abstract

These packages provides fixes to LaTeX 2_ε which are desirable but cannot be integrated into the LaTeX 2_ε 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

[†]This file has version number v1.1s, last revised 2014/09/29.
[†]Walter wrote fix-cm

1 Introduction

In the newsletter `ltnews07.tex`, which accompanied the LaTeX 2$_\varepsilon$ 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 LaTeX cannot be changed.

Back then we probably should have said that this decision also covers changes to the 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 ⟨*date*⟩, 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 ⟨*date*⟩.

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

```
\NeedsTeXFormat{latex2e}[⟨date⟩]
```

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 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 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 LaTeX in a rush.

The packages are independent from each other and deal with different subjects: fixltx2e provides general changes to the 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 LaTeX 2$_\varepsilon$ so that, if urgently needed, people can simply add them to the current directory in case they cannot upgrade their LaTeX for whatever reason.

The packages are **NOT** provided so that people can stop upgrading their 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}[⟨date⟩]
```

into the preamble of your document, where ⟨*date*⟩ 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 LaTeX 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 LaTeX ‘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 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. 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 LaTeX2.09 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 LaTeX 2_ε. 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 LaTeX twocolumn system works internally by making each column a separate ‘page’ that is passed independently to 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 TeX marks that are normally used for header information, are globally reset after the first column. By default 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 \eminnershape, e.g.,

```
\renewcommand\eminnershape{\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 14.4 pt 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 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\csname 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.

`\TextOrMath` 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 TeX: A reliable test for math mode that doesn't destroy kerning. Fortunately this problem can be solved when using eTeX so if you use this as engine for your LaTeX format, as recommended by the LaTeX3 Project, you will get a fully functioning `\TextOrMath` command with no side effects. If you use regular TeX as engine for your LaTeX 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 LaTeX.

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

`\MakeRobust` 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 Fixes added after 2014/05/01

6.1 Within counters only reset next level down (pr4393)

This is actually implicitly documented behavior in the LaTeX Manual that states that `\stepcounter` resets all counters marked “within”. However it means that if, for example, theorems are numbered within sections and you start a new chapter in a book, the section counter is reset to zero but the theorem counter is not until the first section appears. Thus a theorem directly within the chapter body (without a new section) would show an incremented number relative to the last theorem of the previous chapter.

For this reason we are now resetting all levels of within in one go even if that means that some of these resets may happen several times unnecessarily.

7 Implementation

We require at least a somewhat sane version of LaTeX 2$_\varepsilon$. Earlier ones where really quite different from one another.

```
⟨*fixltx2e⟩
\NeedsTeXFormat{LaTeX2e}[1996/06/01]
```

7.1 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.

7.1.1 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.

```
\def\@outputdblcol{%
  \if@firstcolumn
    \global\@firstcolumnfalse
```

Save the left column

```
    \global\setbox\@leftcolumn\copy\@outputbox
```

Remember the marks from the first column

```
    \splitmaxdepth\maxdimen
    \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 thek last glue (if any) at the end of `\@outputbox` as we are only interested in marks that change doesn't matter.

```
     \setbox\@outputbox\vbox{\unvbox\@outputbox\unskip}%
     \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`.

```
    \toks@\expandafter{\topmark}%
    \xdef\@firstcoltopmark{\the\toks@}%
    \toks@\expandafter{\splitfirstmark}%
    \xdef\@firstcolfirstmark{\the\toks@}%
```

This test does not work if truly empty marks have been inserted, but 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.)

```
    \ifx\@firstcolfirstmark\@empty
      \global\let\@setmarks\relax
    \else
      \gdef\@setmarks{%
        \let\firstmark\@firstcolfirstmark
        \let\topmark\@firstcoltopmark}%
    \fi
```

End of change

```
  \else
    \global\@firstcolumntrue
    \setbox\@outputbox\vbox{%
     \hb@xt@\textwidth{%
        \hb@xt@\columnwidth{\box\@leftcolumn \hss}%
        \hfil
```

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

```
        {\normalcolor\vrule \@width\columnseprule}%
        \hfil
       \hb@xt@\columnwidth{\box\@outputbox \hss}}}%
  \@combinedblfloats
```

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

```
    \@setmarks
```

End of change

```
    \@outputpage
    \begingroup
      \@dblfloatplacement
      \@startdblcolumn
      \@whilesw\if@fcolmade \fi{\@outputpage\@startdblcolumn}%
    \endgroup
  \fi}
```

7.1.2 Preserving Float Order

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

```
\def\end@dblfloat{%
  \if@twocolumn
    \@endfloatbox
    \ifnum\@floatpenalty <\z@
      \@largefloatcheck
```

Force the depth of two column float boxes.

```
      \global\dp\@currbox1sp %
```

What follows is essentially `\end@float` without a starting `\@endfloatbox`.

```
      \@cons\@currlist\@currbox
      \ifnum\@floatpenalty <-\@Mii
        \penalty -\@Miv
        \@tempdima\prevdepth
        \vbox{}%
        \prevdepth\@tempdima
        \penalty\@floatpenalty
      \else
        \vadjust{\penalty -\@Miv \vbox{}\penalty\@floatpenalty}\@Esphack
      \fi
    \fi
  \else
    \end@float
  \fi
}
```

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).

```
\def\@testwrongwidth #1{%
  \ifdim\dp#1=\f@depth
  \else
    \global\@testtrue
  \fi}
```

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

```
\let\f@depth\z@
```

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

```
\def\@dblfloatplacement{\global\@dbltopnum\c@dbltopnumber
   \global\@dbltoproom \dbltopfraction\@colht
   \@textmin \@colht
   \advance \@textmin -\@dbltoproom
   \@fpmin \dblfloatpagefraction\textheight
   \@fptop \@dblfptop
   \@fpsep \@dblfpsep
   \@fpbot \@dblfpbot
   \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.

```
\def \@doclearpage {%
     \ifvoid\footins
       \setbox\@tempboxa\vsplit\@cclv to\z@ \unvbox\@tempboxa
       \setbox\@tempboxa\box\@cclv
       \xdef\@deferlist{\@toplist\@botlist\@deferlist}%
       \global \let \@toplist \@empty
       \global \let \@botlist \@empty
       \global \@colroom \@colht
       \ifx \@currlist\@empty
       \else
          \@latexerr{Float(s) lost}\@ehb
          \global \let \@currlist \@empty
       \fi
       \@makefcolumn\@deferlist
       \@whilesw\if@fcolmade \fi{\@opcol\@makefcolumn\@deferlist}%
       \if@twocolumn
         \if@firstcolumn
           \xdef\@deferlist{\@dbltoplist\@deferlist}%
           \global \let \@dbltoplist \@empty
           \global \@colht \textheight
           \begingroup
              \@dblfloatplacement
              \@makefcolumn\@deferlist
              \@whilesw\if@fcolmade \fi{\@outputpage
                                        \@makefcolumn\@deferlist}%
           \endgroup
         \else
           \vbox{}\clearpage
         \fi
       \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.

```
      \ifx\@deferlist\@empty \else\clearpage \fi
    \else
      \setbox\@cclv\vbox{\box\@cclv\vfil}%
      \@makecol\@opcol
      \clearpage
    \fi
}
\def \@startdblcolumn {%
  \@tryfcolumn \@deferlist
  \if@fcolmade
  \else
    \begingroup
      \let \reserved@b \@deferlist
      \global \let \@deferlist \@empty
      \let \@elt \@sdblcolelt
      \reserved@b
    \endgroup
  \fi
}
\def\@addtonextcol{%
  \begingroup
   \@insertfalse
   \@setfloattypecounts
   \ifnum \@fpstype=8
   \else
     \ifnum \@fpstype=24
     \else
       \@flsettextmin
       \@reqcolroom \ht\@currbox
       \advance \@reqcolroom \@textmin
       \ifdim \@colroom>\@reqcolroom
         \@flsetnum \@colnum
         \ifnum\@colnum>\z@
            \@bitor\@currtype\@deferlist
            \@testwrongwidth\@currbox
            \if@test
            \else
              \@addtotoporbot
            \fi
         \fi
       \fi
     \fi
   \fi
   \if@insert
   \else
     \@cons\@deferlist\@currbox
   \fi
  \endgroup
}
\def\@addtodblcol{%
  \begingroup
   \@insertfalse
   \@setfloattypecounts
   \@getfpsbit \tw@
   \ifodd\@tempcnta
     \@flsetnum \@dbltopnum
     \ifnum \@dbltopnum>\z@
       \@tempswafalse
       \ifdim \@dbltoproom>\ht\@currbox
         \@tempswatrue
       \else
         \ifnum \@fpstype<\sixt@@n
           \advance \@dbltoproom \@textmin
           \ifdim \@dbltoproom>\ht\@currbox
             \@tempswatrue
           \fi
           \advance \@dbltoproom -\@textmin
         \fi
       \fi
       \if@tempswa
           \@bitor \@currtype \@deferlist
```

not in fixfloats?

```
          \@testwrongwidth\@currbox
```

```
           \if@test
           \else
              \@tempdima -\ht\@currbox
              \advance\@tempdima
                -\ifx \@dbltoplist\@empty \dbltextfloatsep \else
                                          \dblfloatsep \fi
              \global \advance \@dbltoproom \@tempdima
              \global \advance \@colht \@tempdima
              \global \advance \@dbltopnum \m@ne
              \@cons \@dbltoplist \@currbox
              \@inserttrue
           \fi
       \fi
     \fi
   \fi
   \if@insert
   \else
     \@cons\@deferlist\@currbox
   \fi
  \endgroup
}
\def \@addtocurcol {%
   \@insertfalse
   \@setfloattypecounts
   \ifnum \@fpstype=8
   \else
     \ifnum \@fpstype=24
     \else
       \@flsettextmin
       \advance \@textmin \@textfloatsheight
       \@reqcolroom \@pageht
       \ifdim \@textmin>\@reqcolroom
         \@reqcolroom \@textmin
       \fi
```

```
      \advance \@reqcolroom \ht\@currbox
      \ifdim \@colroom>\@reqcolroom
        \@flsetnum \@colnum
        \ifnum \@colnum>\z@
          \@bitor\@currtype\@deferlist
```

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

```
          \@testwrongwidth\@currbox
          \if@test
          \else
            \@bitor\@currtype\@botlist
            \if@test
              \@addtobot
            \else
              \ifodd \count\@currbox
                \advance \@reqcolroom \intextsep
                \ifdim \@colroom>\@reqcolroom
                  \global \advance \@colnum \m@ne
                  \global \advance \@textfloatsheight \ht\@currbox
                  \global \advance \@textfloatsheight 2\intextsep
                  \@cons \@midlist \@currbox
                  \if@nobreak
                    \nobreak
                    \@nobreakfalse
                    \everypar{}%
                  \else
                    \addpenalty \interlinepenalty
                  \fi
                  \vskip \intextsep
                  \box\@currbox
                  \penalty\interlinepenalty
                  \vskip\intextsep
                  \ifnum\outputpenalty <-\@Mii \vskip -\parskip\fi
                  \outputpenalty \z@
                  \@inserttrue
                \fi
              \fi
              \if@insert
              \else
                \@addtotoporbot
              \fi
            \fi
          \fi
        \fi
      \fi
    \fi
  \fi
  \if@insert
  \else
    \@resethfps
    \@cons\@deferlist\@currbox
  \fi
}
\def\@xtryfc #1{%
  \@next\reserved@a\@trylist{}{}%
  \@currtype \count #1%
  \divide\@currtype\@xxxii
  \multiply\@currtype\@xxxii
  \@bitor \@currtype \@failedlist
  \@testfp #1%
  \@testwrongwidth #1%
  \ifdim \ht #1>\@colht
     \@testtrue
  \fi
  \if@test
    \@cons\@failedlist #1%
  \else
    \@ytryfc #1%
  \fi}
```

```
\def\@ztryfc #1{%
  \@tempcnta\count #1%
  \divide\@tempcnta\@xxxii
  \multiply\@tempcnta\@xxxii
  \@bitor \@tempcnta {\@failedlist \@flfail}%
  \@testfp #1%
```

not in fixfloats?

```
  \@testwrongwidth #1%
```

```
  \@tempdimb\@tempdima
  \advance\@tempdimb\ht #1%
  \advance\@tempdimb\@fpsep
  \ifdim \@tempdimb >\@colht
    \@testtrue
  \fi
  \if@test
    \@cons\@flfail #1%
  \else
    \@cons\@flsucceed #1%
    \@tempdima\@tempdimb
  \fi}
```

7.2 `\@` discards spaces when moving (pr3039)

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

```
\def\@{\spacefactor\@m{}}
```

7.3 `\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.

```
\def\@tempa#1#2{#1#2\relax}
\ifx\setlength\@tempa
  \def\setlength#1#2{#1 #2\relax}
\fi
```

7.4 `\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.

```
\def\addpenalty#1{%
  \ifvmode
    \if@minipage
    \else
      \if@nobreak
      \else
        \ifdim\lastskip=\z@
          \penalty#1\relax
      \else
        \@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.

```
        \begingroup
          \advance \@tempskipb
            \ifdim\prevdepth>\maxdepth\maxdepth\else
```

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

```
              \ifdim \prevdepth = -\@m\p@ \z@ \else \prevdepth \fi
            \fi
          \vskip -\@tempskipb
          \penalty#1%
          \vskip\@tempskipb
        \endgroup
        \vskip -\@tempskipb
        \vskip \@tempskipb
      \fi
    \fi
  \fi
\else
  \@noitemerr
\fi}
```

7.5 `\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 eTeX as engine for LaTeX (as recommended) this unfortunate side effect is not present.

```
\def\@fnsymbol#1{%
  \ifcase#1\or \TextOrMath\textasteriskcentered *\or
  \TextOrMath \textdagger \dagger\or
  \TextOrMath \textdaggerdbl \ddagger \or
  \TextOrMath \textsection  \mathsection\or
  \TextOrMath \textparagraph \mathparagraph\or
  \TextOrMath \textbardbl \|\or
  \TextOrMath {\textasteriskcentered\textasteriskcentered}{**}\or
  \TextOrMath {\textdagger\textdagger}{\dagger\dagger}\or
  \TextOrMath {\textdaggerdbl\textdaggerdbl}{\ddagger\ddagger}\else
  \@ctrerr \fi
}
```

`\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 wil have the disadvantage of choosing the wrong branch if appearing at the beginning of an alignment cell. However, users of eTeX 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 eTeX but making sure not to permanently turn `\eTeXversion` into `\relax`.

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\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.

```
\DeclareRobustCommand\TextOrMath{%
  \ifmmode  \expandafter\@secondoftwo
  \else     \expandafter\@firstoftwo  \fi}
\protected@edef\TextOrMath#1#2{\TextOrMath{#1}{#2}}
\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.

```
\protected\expandafter\def\csname TextOrMath\space\endcsname{%
  \ifmmode  \expandafter\@secondoftwo
  \else     \expandafter\@firstoftwo  \fi}
\edef\TextOrMath#1#2{%
  \expandafter\noexpand\csname TextOrMath\space\endcsname
  {#1}{#2}}
\fi
```

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

`\@esphack` `\@Esphack` Fix suggested by Donald Arseneau.

```
\def\@esphack{%
  \relax
  \ifhmode
    \spacefactor\@savsf
    \ifdim\@savsk>\z@
      \nobreak \hskip\z@skip  % <------
      \ignorespaces
    \fi
  \fi}
\def\@Esphack{%
  \relax
  \ifhmode
    \spacefactor\@savsf
    \ifdim\@savsk>\z@
      \nobreak \hskip\z@skip  % <------
      \@ignoretrue
      \ignorespaces
    \fi
   \fi}
```

7.7 Allowing `\emph` to produce small caps, etc

`\em` `\eminnershape`

```
\DeclareRobustCommand\em
        {\@nomath\em \ifdim \fontdimen\@ne\font >\z@
                         \eminnershape \else \itshape \fi}
\def\eminnershape{\upshape}
```

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

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

```
\DeclareRobustCommand*\textsubscript[1]{%
  \@textsubscript{\selectfont#1}}
\def\@textsubscript#1{%
  {\m@th\ensuremath{_{\mbox{\fontsize\sf@size\z@#1}}}}}
```

7.9 `\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}`.

```
\def\@DeclareMathSizes #1#2#3#4#5{%
  \@defaultunits\dimen@ #2pt\relax\@nnil
  \if $#3$%
    \expandafter\let\csname S@\strip@pt\dimen@\endcsname\math@fontsfalse
  \else
    \@defaultunits\dimen@ii #3pt\relax\@nnil
    \@defaultunits\@tempdima #4pt\relax\@nnil
    \@defaultunits\@tempdimb #5pt\relax\@nnil
    \toks@{#1}%
    \expandafter\xdef\csname S@\strip@pt\dimen@\endcsname{%
      \gdef\noexpand\tf@size{\strip@pt\dimen@ii}%
      \gdef\noexpand\sf@size{\strip@pt\@tempdima}%
      \gdef\noexpand\ssf@size{\strip@pt\@tempdimb}%
      \the\toks@
    }%
  \fi
}
```

7.10 Fewer fragile macros

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

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

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`.

```
  {%
    \@ifundefined{\expandafter\@gobble\string#1\space}%
    {%
      \expandafter\let\csname
      \expandafter\@gobble\string#1\space\endcsname=#1%
      \edef\reserved@a{\string#1}%
      \def\reserved@b{#1}%
      \edef\reserved@b{\expandafter\strip@prefix\meaning\reserved@b}%
      \edef#1{%
        \ifx\reserved@a\reserved@b
          \noexpand\x@protect\noexpand#1%
        \fi
        \noexpand\protect\expandafter\noexpand
        \csname\expandafter\@gobble\string#1\space\endcsname}%
    }%
    {\@latex@info{The control sequence `\string#1' is already robust}}%
   }%
}
```

Here we make some kernel macros robust.

```
\MakeRobust\(
\MakeRobust\)
\MakeRobust\[
\MakeRobust\]
\MakeRobust\makebox
\MakeRobust\savebox
```

```
\MakeRobust\framebox
\MakeRobust\parbox
\MakeRobust\rule
\MakeRobust\raisebox
⟨/fixltx2e⟩
```

7.11 Using EC fonts (T1 encoding) makes my documents look bl**dy horrible

7.11.1 Preliminaries

The LaTeX 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.

```
⟨*fix-cm⟩
\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 10.95 pt need to be removed:

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

fix-cm may still fail, if the EC fonts are preloaded in the LaTeX 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.

```
\begingroup
\nfss@catcodes
```

7.11.2 T1 encoding

CM Roman

```
\DeclareFontFamily{T1}{cmr}{}
\DeclareFontShape{T1}{cmr}{m}{n}{
        <-6>    ecrm0500
        <6-7>   ecrm0600
        <7-8>   ecrm0700
        <8-9>   ecrm0800
        <9-10>  ecrm0900
        <10-12> ecrm1000
        <12-17> ecrm1200
        <17->   ecrm1728
      }{}
\DeclareFontShape{T1}{cmr}{m}{sl}{
        <-6>    ecsl0500
        <6-7>   ecsl0600
        <7-8>   ecsl0700
        <8-9>   ecsl0800
        <9-10>  ecsl0900
        <10-12> ecsl1000
        <12-17> ecsl1200
        <17->   ecsl1728
      }{}
\DeclareFontShape{T1}{cmr}{m}{it}{
        <-8>    ecti0700
        <8-9>   ecti0800
        <9-10>  ecti0900
        <10-12> ecti1000
        <12-17> ecti1200
        <17->   ecti1728
      }{}
\DeclareFontShape{T1}{cmr}{m}{sc}{
        <-6>    eccc0500
        <6-7>   eccc0600
        <7-8>   eccc0700
        <8-9>   eccc0800
        <9-10>  eccc0900
        <10-12> eccc1000
        <12-17> eccc1200
        <17->   eccc1728
               }{}
\DeclareFontShape{T1}{cmr}{m}{ui}{
        <-8>    ecui0700
        <8-9>   ecui0800
        <9-10>  ecui0900
        <10-12> ecui1000
        <12-17> ecui1200
        <17->   ecui1728
      }{}
\DeclareFontShape{T1}{cmr}{b}{n}{
        <-6>    ecrb0500
        <6-7>   ecrb0600
        <7-8>   ecrb0700
        <8-9>   ecrb0800
        <9-10>  ecrb0900
        <10-12> ecrb1000
        <12-17> ecrb1200
        <17->   ecrb1728
      }{}
\DeclareFontShape{T1}{cmr}{bx}{n}{
        <-6>    ecbx0500
        <6-7>   ecbx0600
        <7-8>   ecbx0700
        <8-9>   ecbx0800
        <9-10>  ecbx0900
        <10-12> ecbx1000
        <12->   ecbx1200
      }{}
\DeclareFontShape{T1}{cmr}{bx}{sl}{
        <-6>    ecbl0500
        <6-7>   ecbl0600
        <7-8>   ecbl0700
        <8-9>   ecbl0800
        <9-10>  ecbl0900
        <10-12> ecbl1000
        <12->   ecbl1200
      }{}
\DeclareFontShape{T1}{cmr}{bx}{it}{
        <-8>    ecbi0700
        <8-9>   ecbi0800
        <9-10>  ecbi0900
        <10-12> ecbi1000
        <12->   ecbi1200
      }{}
\DeclareFontShape{T1}{cmr}{bx}{sc}{
        <-6>    ecxc0500
        <6-7>   ecxc0600
        <7-8>   ecxc0700
        <8-9>   ecxc0800
        <9-10>  ecxc0900
        <10-12> ecxc1000
        <12->   ecxc1200
      }{}
%
```

CM Sans

```
\DeclareFontFamily{T1}{cmss}{}
\DeclareFontShape{T1}{cmss}{m}{n}{
        <-9>    ecss0800
        <9-10>  ecss0900
        <10-12> ecss1000
        <12-17> ecss1200
        <17->   ecss1728
      }{}
\DeclareFontShape{T1}{cmss}{m}{sl}{
        <-9>    ecsi0800
        <9-10>  ecsi0900
        <10-12> ecsi1000
        <12-17> ecsi1200
        <17->   ecsi1728
      }{}
\DeclareFontShape{T1}{cmss}{m}{it}
       {<->ssub*cmss/m/sl}{}
\DeclareFontShape{T1}{cmss}{m}{sc}
       {<->sub*cmr/m/sc}{}
\DeclareFontShape{T1}{cmss}{sbc}{n}{
        <->     ecssdc10
       }{}
\DeclareFontShape{T1}{cmss}{bx}{n}{
        <-10>   ecsx0900
        <10->   ecsx1000
      }{}
\DeclareFontShape{T1}{cmss}{bx}{sl}{
        <-10>   ecso0900
        <10->   ecso1000
      }{}
\DeclareFontShape{T1}{cmss}{bx}{it}
       {<->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`.

```
\DeclareFontShape{T1}{cmss}{b}{n}
       {<->ssub*cmss/bx/n}{}
\DeclareFontShape{T1}{cmss}{b}{sl}
       {<->ssub*cmss/bx/sl}{}
\DeclareFontShape{T1}{cmss}{b}{it}
       {<->ssub*cmss/bx/sl}{}
```

CM Typewriter

```
\DeclareFontFamily{T1}{cmtt}{\hyphenchar \font\m@ne}
\DeclareFontShape{T1}{cmtt}{m}{n}{
        <-9>    ectt0800
        <9-10>  ectt0900
        <10-12> ectt1000
        <12-17> ectt1200
        <17->   ectt1728
      }{}
\DeclareFontShape{T1}{cmtt}{m}{it}{
        <-9>    ecit0800
        <9-10>  ecit0900
        <10-12> ecit1000
        <12-17> ecit1200
        <17->   ecit1728
      }{}
\DeclareFontShape{T1}{cmtt}{m}{sl}{
        <-9>    ecst0800
        <9-10>  ecst0900
        <10-12> ecst1000
        <12-17> ecst1200
        <17->   ecst1728
      }{}
\DeclareFontShape{T1}{cmtt}{m}{sc}{
        <-9>    ectc0800
        <9-10>  ectc0900
        <10-12> ectc1000
        <12-17> ectc1200
        <17->   ectc1728
      }{}
\DeclareFontShape{T1}{cmtt}{bx}{n}
      {<->sub * cmtt/m/n}{}
\DeclareFontShape{T1}{cmtt}{bx}{it}
      {<->sub * cmtt/m/it}{}
\DeclareFontShape{T1}{cmtt}{bx}{sl}
      {<->sub * cmtt/m/sl}{}
```

Substitutions not provided in the default `.fd` files:

```
\DeclareFontShape{T1}{cmtt}{b}{n}
      {<->sub * cmtt/m/n}{}
\DeclareFontShape{T1}{cmtt}{b}{it}
      {<->sub * cmtt/m/it}{}
\DeclareFontShape{T1}{cmtt}{b}{sl}
      {<->sub * cmtt/m/sl}{}
```

CM Typewiter (var.)

```
\DeclareFontFamily{T1}{cmvtt}{}
\DeclareFontShape{T1}{cmvtt}{m}{n}{
        <-9>    ecvt0800
        <9-10>  ecvt0900
        <10-12> ecvt1000
        <12-17> ecvt1200
        <17->   ecvt1728
      }{}
\DeclareFontShape{T1}{cmvtt}{m}{it}{
        <-9>    ecvi0800
        <9-10>  ecvi0900
        <10-12> ecvi1000
        <12-17> ecvi1200
        <17->   ecvi1728
      }{}
```

7.11.3 TS1 encoding

CM Roman

```
\DeclareFontFamily{TS1}{cmr}{\hyphenchar\font\m@ne}
\DeclareFontShape{TS1}{cmr}{m}{n}{
        <-6>    tcrm0500
        <6-7>   tcrm0600
        <7-8>   tcrm0700
        <8-9>   tcrm0800
        <9-10>  tcrm0900
        <10-12> tcrm1000
        <12-17> tcrm1200
        <17->   tcrm1728
      }{}
\DeclareFontShape{TS1}{cmr}{m}{sl}{
        <-6>    tcsl0500
        <6-7>   tcsl0600
        <7-8>   tcsl0700
        <8-9>   tcsl0800
        <9-10>  tcsl0900
        <10-12> tcsl1000
        <12-17> tcsl1200
        <17->   tcsl1728
      }{}
\DeclareFontShape{TS1}{cmr}{m}{it}{
        <-8>    tcti0700
        <8-9>   tcti0800
        <9-10>  tcti0900
        <10-12> tcti1000
        <12-17> tcti1200
        <17->   tcti1728
      }{}
\DeclareFontShape{TS1}{cmr}{m}{ui}{
        <-8>    tcui0700
        <8-9>   tcui0800
        <9-10>  tcui0900
        <10-12> tcui1000
        <12-17> tcui1200
        <17->   tcui1728
      }{}
\DeclareFontShape{TS1}{cmr}{b}{n}{
        <-6>    tcrb0500
        <6-7>   tcrb0600
        <7-8>   tcrb0700
        <8-9>   tcrb0800
        <9-10>  tcrb0900
        <10-12> tcrb1000
        <12-17> tcrb1200
        <17->   tcrb1728
      }{}
\DeclareFontShape{TS1}{cmr}{bx}{n}{
        <-6>    tcbx0500
        <6-7>   tcbx0600
        <7-8>   tcbx0700
        <8-9>   tcbx0800
        <9-10>  tcbx0900
        <10-12> tcbx1000
        <12->   tcbx1200
      }{}
\DeclareFontShape{TS1}{cmr}{bx}{sl}{
        <-6>    tcbl0500
        <6-7>   tcbl0600
        <7-8>   tcbl0700
        <8-9>   tcbl0800
        <9-10>  tcbl0900
        <10-12> tcbl1000
        <12->   tcbl1200
      }{}
\DeclareFontShape{TS1}{cmr}{bx}{it}{
        <-8>    tcbi0700
        <8-9>   tcbi0800
        <9-10>  tcbi0900
        <10-12> tcbi1000
```

```
          <12->   tcbi1200
        }{}
```

CM Sans

```
\DeclareFontFamily{TS1}{cmss}{\hyphenchar\font\m@ne}
\DeclareFontShape{TS1}{cmss}{m}{n}{
          <-9>    tcss0800
          <9-10>  tcss0900
          <10-12> tcss1000
          <12-17> tcss1200
          <17->   tcss1728
        }{}
\DeclareFontShape{TS1}{cmss}{m}{it}
        {<->ssub*cmss/m/sl}{}
\DeclareFontShape{TS1}{cmss}{m}{sl}{
          <-9>    tcsi0800
          <9-10>  tcsi0900
          <10-12> tcsi1000
          <12-17> tcsi1200
          <17->   tcsi1728
        }{}
\DeclareFontShape{TS1}{cmss}{sbc}{n}{
          <->     tcssdc10
         }{}
\DeclareFontShape{TS1}{cmss}{bx}{n}{
          <-10>   tcsx0900
          <10->   tcsx1000
        }{}
\DeclareFontShape{TS1}{cmss}{bx}{sl}{
          <-10>   tcso0900
          <10->   tcso1000
        }{}
\DeclareFontShape{TS1}{cmss}{bx}{it}
        {<->ssub*cmss/bx/sl}{}
```

Substitutions not provided in the default `.fd` files:

```
\DeclareFontShape{TS1}{cmss}{b}{n}
        {<->ssub*cmss/bx/n}{}
\DeclareFontShape{TS1}{cmss}{b}{sl}
        {<->ssub*cmss/bx/sl}{}
\DeclareFontShape{TS1}{cmss}{b}{it}
        {<->ssub*cmss/bx/sl}{}
```

CM Typewriter

```
\DeclareFontFamily{TS1}{cmtt}{\hyphenchar \font\m@ne}
\DeclareFontShape{TS1}{cmtt}{m}{n}{
          <-9>    tctt0800
          <9-10>  tctt0900
          <10-12> tctt1000
          <12-17> tctt1200
          <17->   tctt1728
        }{}
\DeclareFontShape{TS1}{cmtt}{m}{it}{
          <-9>    tcit0800
          <9-10>  tcit0900
          <10-12> tcit1000
          <12-17> tcit1200
          <17->   tcit1728
        }{}
\DeclareFontShape{TS1}{cmtt}{m}{sl}{
          <-9>    tcst0800
          <9-10>  tcst0900
          <10-12> tcst1000
          <12-17> tcst1200
          <17->   tcst1728
        }{}
  \DeclareFontShape{TS1}{cmtt}{bx}{n}
         {<->sub * cmtt/m/n}{}
  \DeclareFontShape{TS1}{cmtt}{bx}{it}
         {<->sub * cmtt/m/it}{}
  \DeclareFontShape{TS1}{cmtt}{bx}{sl}
         {<->sub * cmtt/m/sl}{}
```

Substitutions not provided in the default `.fd` files:

```
\DeclareFontShape{TS1}{cmtt}{b}{n}
       {<->sub * cmtt/m/n}{}
\DeclareFontShape{TS1}{cmtt}{b}{it}
       {<->sub * cmtt/m/it}{}
\DeclareFontShape{TS1}{cmtt}{b}{sl}
       {<->sub * cmtt/m/sl}{}
```

CM Typewriter (var.)

```
\DeclareFontFamily{TS1}{cmvtt}{}
\DeclareFontShape{TS1}{cmvtt}{m}{n}{
        <-9>    tcvt0800
        <9-10>  tcvt0900
        <10-12> tcvt1000
        <12-17> tcvt1200
        <17->   tcvi1728
      }{}
\DeclareFontShape{TS1}{cmvtt}{m}{it}{
        <-9>    tcvi0800
        <9-10>  tcvi0900
        <10-12> tcvi1000
        <12-17> tcvi1200
        <17->   tcvi1728
      }{}
```

7.11.4 OT1 encoding

CM Roman

```
\DeclareFontFamily{OT1}{cmr}{\hyphenchar\font45 }
\DeclareFontShape{OT1}{cmr}{m}{n}{
        <-6>    cmr5
        <6-7>   cmr6
        <7-8>   cmr7
        <8-9>   cmr8
        <9-10>  cmr9
        <10-12> cmr10
        <12-17> cmr12
        <17->   cmr17
      }{}
\DeclareFontShape{OT1}{cmr}{m}{sl}{
        <-9>    cmsl8
        <9-10>  cmsl9
        <10-12> cmsl10
        <12->   cmsl12
      }{}
\DeclareFontShape{OT1}{cmr}{m}{it}{
        <-8>    cmti7
        <8-9>   cmti8
        <9-10>  cmti9
        <10-12> cmti10
        <12->   cmti12
      }{}
\DeclareFontShape{OT1}{cmr}{m}{sc}{
        <->     cmcsc10
      }{}
\DeclareFontShape{OT1}{cmr}{m}{ui}{
        <->     cmu10
      }{}
\DeclareFontShape{OT1}{cmr}{b}{n}{
        <->     cmb10
      }{}
\DeclareFontShape{OT1}{cmr}{bx}{n}{
        <-6>    cmbx5
        <6-7>   cmbx6
        <7-8>   cmbx7
        <8-9>   cmbx8
        <9-10>  cmbx9
        <10-12> cmbx10
        <12->   cmbx12
      }{}
\DeclareFontShape{OT1}{cmr}{bx}{sl}{
        <->     cmbxsl10
      }{}
\DeclareFontShape{OT1}{cmr}{bx}{it}{
        <->     cmbxti10
      }{}
\DeclareFontShape{OT1}{cmr}{bx}{ui}
      {<->sub*cmr/m/ui}{}
```

CM Sans

```
\DeclareFontFamily{OT1}{cmss}{\hyphenchar\font45 }
\DeclareFontShape{OT1}{cmss}{m}{n}{
        <-9>    cmss8
        <9-10>  cmss9
        <10-12> cmss10
        <12-17> cmss12
        <17->   cmss17
      }{}
\DeclareFontShape{OT1}{cmss}{m}{it}
      {<->sub*cmss/m/sl}{}
\DeclareFontShape{OT1}{cmss}{m}{sl}{
        <-9>    cmssi8
        <9-10>  cmssi9
        <10-12> cmssi10
        <12-17> cmssi12
        <17->   cmssi17
      }{}
\DeclareFontShape{OT1}{cmss}{m}{sc}
       {<->sub*cmr/m/sc}{}
\DeclareFontShape{OT1}{cmss}{m}{ui}
       {<->sub*cmr/m/ui}{}
\DeclareFontShape{OT1}{cmss}{sbc}{n}{
        <->     cmssdc10
      }{}
\DeclareFontShape{OT1}{cmss}{bx}{n}{
        <->     cmssbx10
      }{}
\DeclareFontShape{OT1}{cmss}{bx}{ui}
       {<->sub*cmr/bx/ui}{}
```

CM Typewriter

```
\DeclareFontFamily{OT1}{cmtt}{\hyphenchar \font\m@ne}
```

```
\DeclareFontShape{OT1}{cmtt}{m}{n}{
        <-9>    cmtt8
        <9-10>  cmtt9
        <10-12> cmtt10
        <12->   cmtt12
      }{}
\DeclareFontShape{OT1}{cmtt}{m}{it}{
        <->     cmitt10
      }{}
\DeclareFontShape{OT1}{cmtt}{m}{sl}{
        <->     cmsltt10
      }{}
\DeclareFontShape{OT1}{cmtt}{m}{sc}{
        <->     cmtcsc10
      }{}
\DeclareFontShape{OT1}{cmtt}{m}{ui}
       {<->ssub*cmtt/m/it}{}
\DeclareFontShape{OT1}{cmtt}{bx}{n}
       {<->ssub*cmtt/m/n}{}
\DeclareFontShape{OT1}{cmtt}{bx}{it}
       {<->ssub*cmtt/m/it}{}
\DeclareFontShape{OT1}{cmtt}{bx}{ui}
       {<->ssub*cmtt/m/it}{}
```

CM Typewriter (var.)

```
\DeclareFontFamily{OT1}{cmvtt}{\hyphenchar\font45 }
\DeclareFontShape{OT1}{cmvtt}{m}{n}{
        <->     cmvtt10
      }{}
\DeclareFontShape{OT1}{cmvtt}{m}{it}{
        <->     cmvtti10
      }{}
```

7.11.5 OML and OMS encoded math fonts

```
\DeclareFontFamily{OML}{cmm}{\skewchar\font127 }
\DeclareFontShape{OML}{cmm}{m}{it}{
        <-6>    cmmi5
        <6-7>   cmmi6
        <7-8>   cmmi7
        <8-9>   cmmi8
        <9-10>  cmmi9
        <10-12> cmmi10
        <12->   cmmi12
      }{}
\DeclareFontShape{OML}{cmm}{b}{it}{<-6>cmmib5<6-8>cmmib7<8->cmmib10}{}
\DeclareFontShape{OML}{cmm}{bx}{it}
       {<->ssub*cmm/b/it}{}
```

```
\DeclareFontFamily{OMS}{cmsy}{\skewchar\font48 }
\DeclareFontShape{OMS}{cmsy}{m}{n}{
        <-6>    cmsy5
        <6-7>   cmsy6
        <7-8>   cmsy7
        <8-9>   cmsy8
        <9-10>  cmsy9
        <10->   cmsy10
      }{}
\DeclareFontShape{OMS}{cmsy}{b}{n}{<-6>cmbsy5<6-8>cmbsy7<8->cmbsy10}{}
```

7.11.6 LaTeX symbols

```
\DeclareFontFamily{U}{lasy}{}
\DeclareFontShape{U}{lasy}{m}{n}{
        <-6>    lasy5
        <6-7>   lasy6
        <7-8>   lasy7
        <8-9>   lasy8
        <9-10>  lasy9
        <10->   lasy10
      }{}
\DeclareFontShape{U}{lasy}{b}{n}{
        <-10>   ssub * lasy/m/n
        <10->   lasyb10
      }{}
```

```
\endgroup
⟨/fix-cm⟩
```

7.12 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.

```
⟨*fixltx2e⟩
```

```
\def\@xfloat #1[#2]{%
  \@nodocument
  \def \@captype {#1}%
   \def \@fps {#2}%
   \@onelevel@sanitize \@fps
   \def \reserved@b {!}%
   \ifx \reserved@b \@fps
     \@fpsadddefault
   \else
     \ifx \@fps \@empty
       \@fpsadddefault
     \fi
   \fi
   \ifhmode
     \@bsphack
     \@floatpenalty -\@Mii
   \else
     \@floatpenalty-\@Miii
   \fi
  \ifinner
     \@parmoderr\@floatpenalty\z@
  \else
    \@next\@currbox\@freelist
      {%
       \@tempcnta \sixt@@n
       \expandafter \@tfor \expandafter \reserved@a
         \expandafter :\expandafter =\@fps
         \do
```

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

```
          {%
           \if \reserved@a h%
             \ifodd \@tempcnta
             \else
               \advance \@tempcnta \@ne
             \fi
           \else\if \reserved@a t%
             \@setfpsbit \tw@
           \else\if \reserved@a b%
             \@setfpsbit 4%
           \else\if \reserved@a p%
             \@setfpsbit 8%
           \else\if \reserved@a !%
             \ifnum \@tempcnta>15
               \advance\@tempcnta -\sixt@@n\relax
             \fi
           \else
             \@latex@error{Unknown float option `\reserved@a'}%
             {Option `\reserved@a' ignored and `p' used.}%
             \@setfpsbit 8%
           \fi\fi\fi\fi\fi
           }%
```

End of changes

```
      \@tempcntb \csname ftype@\@captype \endcsname
      \multiply \@tempcntb \@xxxii
      \advance \@tempcnta \@tempcntb
      \global \count\@currbox \@tempcnta
      }%
   \@fltovf
  \fi
  \global \setbox\@currbox
   \color@vbox
     \normalcolor
     \vbox \bgroup
       \hsize\columnwidth
       \@parboxrestore
       \@floatboxreset
}
```

7.13 Within counters only reset next level down

Rather than resetting the "within" counter to zero we set it to −1 and then run `\stepcounter` that moves it to 0 and also initiates resetting the next level down.

```
\def\@stpelt#1{\global\csname c@#1\endcsname \m@ne\stepcounter{#1}}
⟨/fixltx2e⟩
```