Changes to the LaTeX 2ε format

31 March 2016

Contents

1 Introduction

This document describes changes that have been made to the LaTeX format since the 2014/05/01 LaTeX release.

As announced in LaTeX News 22, the 2015 LaTeX, release adopts a new policy. Improvements and bug fixes will be made to the format sources, with the latexrelease package being available to revert changes to use definitions from an earlier format.

2 Changes introduced in 2017/01/01

A new test is added during format making that eTeX extensions are available. As Noted in LaTeX news, eTeX will now be required to build LaTeX.

Further updates tracking changes for LuaTeX 1.0.

Thhe definition of `\showhyphens` is changed in formats built with Xe-TeX, as the original version, inherited from plain TeX does not work with Xe-TeX.

Changes to the default encoding used by LuaTeX and XeTeXformats to be TU (Unicode) rather than OT1 (7 bit legacy TeX encoding).

3 Changes introduced in 2016/03/31 patch 3

Fixes to `\newinsert` and `\extrafloats`.

4 Changes introduced in 2016/03/31 patch 2

Adjustments to `\c{g}` in OT1 encoding.

5 Changes introduced in 2016/03/31 patch 1

Adjust the upper limit for Character Class allocation in Xe-TeX to 4096 to match a change in Xe-TeX.

6 Changes introduced in 2016/03/31

Modify picture mode as suggested in latex/4452 to avoid leaders of almost zero length.

Modify the checks in `\DeclareMathSymbol` and related commands so that they do not give errors with new LuaTeX releases.

7 Changes introduced in 2016/02/01

Adjustments to LuaTeX support to match changes to the LuaTeX engine, and to the character class allocation in XeTeX.

Load Unicode data from new generic `unicode-data` distribution.

8 Changes introduced in 2015/10/01 patch 2

This release fixes the behaviour of the allocation mechanism if the switch from the standard to extended pool takes place within a group.

9 Changes introduced in 2015/10/01 patch 1

This release allows latexrelease to revert the LuaTeX-specific changes, in particular fixing an incorrect date in part of the mechanism and adding a method to disable callback management entirely.

10 Changes introduced in LaTeX 2015/10/01

10.1 LuaTeX allocation

Almost all changes at this release relate to incorporating allocation macros for luatex into the format as done for etex and xetex in 2015/01/01. For details see `ltluatex.dtx` or LaTeX News 22. `ltluatex.dtx` now forms a new chapter (N) in the documented sources, `source2e.tex`.

10.2 Increased number of floats

The default float list has been increased from 18 to 52 registers if eTeX is available. The list can be increased further using `\extrafloats` however this default allocation uses classic registers below 256 so the registers are also available for `\newinsert` as described below.

10.3 Improved \newinsert

The command `\newinsert` has been extended to take registers from the lists of free float registers once the classic register allocation is used up. This should make it highly unlikely to get "no room" errors on register allocation assuming the format is used with an e-TeX based TeX engine.

10.4 New accent, \textcommabelow (pr/4414)

The command `\textcommabelow` has been added. This is mainly intended for Romanian letters Ș ș Ț ț.

10.5 Unicode 8

The file `unicode-letters.def` used to initialise character data in Unicode TEX variants has been regenerated from data files updated to Unicode 8.

11 Changes introduced in 2015/01/01 patch 2

There were no changes to the format at this release, but the sources were fixed to fix bug latex/4434 affecting bottom float positioning if the latexrelease package was used.

12 Changes introduced in 2015/01/01 patch 1

This release re-introduces the "Patch Level" scheme for identifying releases between main "dated" releases. Early LaTeX 2_ε releases included a mechanism whereby updates could be provided by a *patch file*. This was mainly intended to allow updates to be made without downloading the full sources again, which was an important consideration with download speeds and costs at the time.

The new mechanism incorporates any changes directly into the sources, but having the patch level identified in the banner allows the LaTeX release to be identified, even if (as in this case) most of the changes do not affect the format but affect other base packages such as latexrelease and inputenc. The patch level is shown in the banner at the start of the job, but does not affect the date handling of the `\IncludeInRelease` mechanism.

Apart from re-arranging the version banner, the only change in the format is that `\newtoks` was accidentally defined twice, using the old and new allocation scheme described in Section 13.2. The old definition is now only in the latexrelease package, for use when emulating old formats.

13 Changes between LaTeX releases 2014/05/01 and 2015/01/01[1]

13.1 Support for LaTeX version changes

`\includeInRelease{`⟨*date*⟩`}[`⟨*date*⟩`]{`⟨*label*⟩`}{`⟨*message*⟩`}{`⟨*code*⟩`}`

[1] Much of this text is taken from fixltx2e package which was formerly used to make such changes available separately.

The `\includeInRelease` command has been added to support backward and forward compatibility for the LaTeX format. It supports the declaration of conditional code that can be loaded based on options given to the latexrelease package. Its use is described in detail in the latexrelease package documentation.

13.2 New Allocation Code

Previously `\newcount` and related commands were based on classic TeX and only allocated in the range 0–255. This was extended (in different ways) for e-TeX in the `etex` package and in the `xelatex.ini` and `lualatex.ini` files used in those formats. Related to this the number of boxes allocated to store floats was limited. This was extended to a certain extent in the `morefloats` package (by Don Hosek and H.-Martin Muench) but the new allocation incorporates float allocation directly and supports much larger float lists using the extended registers.

The new code allocates registers in the full extended range ($2^{15} - 1$ for etex and xelatex, $2^{16} - 1$ for lualatex. In addition a new command `\extrafloats` is provided.

`\extrafloats{`⟨*number*⟩`}`

This allocates additional registers for the LaTeX float system to hold figures and tables etc. Similar functionality has been available via the morefloats package but this is a different implementation using extended e-TeX registers when available so allows many more registers to be reserved for floats as they are allocated from a pool of 32 or 64 thousand rather than 256 registers, depending on the engine in use.

`\newmarks{`⟨*command*⟩`}`

e-TeX only, previously available via the etex package. Allocates commands to use the extended e-TeX mark mechanism.

`\newXeTeXintercharclass{`⟨*command*⟩`}`

Xe-TeX only, previously in the Xe-LaTeX format, but added via `xelatex.ini` not part of the core release. Allocates commands to use the Xe-TeX character class mechanism.

13.3 e-TeX tracing if available

`\loggingall` (Usually used via `\tracingall`) is extended to enable additional e-TeX tracing if e-TeX is available. (`\tracingall` extension has been available as part of the `etex` package previously).

Also based on code from the `etex` package, a command `\tracingnone` is added to reverse the effects of `\tracingall` and turn off all primitive TeX tracing. A new command `\hideoutput` has been added which resets the tracing parameters set by `\showoutput`.

13.4 \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:     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.
```

13.5 \@ 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:     Donald Arseneau
>Description:
The \@ command expands to \spacefactor\@m in auxiliary files,
which then ignores following spaces when it is reprocessed.
```

13.6 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:     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 undesirable---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.

13.6.1 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.

13.7 Infinite glue found (pr/4023 and pr/2346)

The fix for pr/2346 did not work as intended 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.

13.8 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 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.

13.8.1 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 page breaker. (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.

13.9 `\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:     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}
```

13.10 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:     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.

13.10.1 Notes on the implementation strategy

Rather than changing the kernel macros to be robust, we have decided to add the macro `\MakeRobust`\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`.

. . . TODO . . . fleqn vesion of `\[\]`

13.11 `\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:     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.)
```

13.12 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.

13.13 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.

13.14 `\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:     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.
```

13.15 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:    Harald Harders

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

13.16 \fnsymbol should use text symbols (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:     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.
```

13.17 \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

 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.

13.17.1 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.

Therefore the implementation here makes `\@fnsymbol` itself a non-robust command again and instead uses a new robust command `\TextOrMath\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.