Introduction

Auto-detecting key–value arguments 1
Encoding subsets for TS1 encoded fonts 1

New or improved commands

Code improvements

Support for slanted small caps in the EC-fonts 1
EC sans serif at small sizes 2
Detect nested minipage environments 2
LuaTeX callback efficiency improvement 2
Improve \l3docstrip integration into docstrip 2
Improve font series handling with incorrect .fd files 2

Bug fixes

Prevent LaTeX from loosing a \smash 2
Resolve an issue with \mathchoice and localalphabets 2

Changes to packages in the amsmath category

Changes to packages in the graphics category

Fix a \mathcolor bug 2

Changes to packages in the tools category

array: Correctly identify single-line m-cells 3

Introduction

to be written

Auto-detecting key–value arguments

To allow extension of the core \LaTeX{} syntax, \ltcmd now supports a =... modifier when grabbing arguments. This modifier instructs \LaTeX{} that the argument should be passed to the underlying code as a set of keyvals. If the argument does not “look like” a set of keyvals, it will be converted into a single key–value pair, with the argument to = specifying the name of that key. For example, the \caption command could be defined as

\begin{verbatim}
\DeclareDocumentCommand\caption{s ={short-text}+O{#3} +m}
{...}
\end{verbatim}

which would mean that if the optional argument does not contain keyval data, it will be converted to a single keyval pair with the key name short-text.

Arguments which begin with =, are always interpreted as keyvals even if they do not contain further = signs.

Any = signs enclosed within $...$ or \(...\), i.e. in inline math mode, are ignored, meaning that only = outside of math mode will generally cause interpretation as keyval material.

In case the argument contains a “textual” = sign that is mistaken as key/value indicator you can hide it using a brace group as you would do in other places, e.g.,

\begin{verbatim}
\caption[{Use of = signs}]
{Use of = signs in optional arguments}
\end{verbatim}

However, because a = sign in math mode are already ignored, this should seldom be necessary.

Encoding subsets for TS1 encoded fonts

The text companion encoding TS1 is unfortunately not very faithfully supported in fonts that are not close cousins to the Computer Modern fonts. It was therefore necessary to provide the notion of “sub-encodings” on a per font basis. These sub-encodings are declared for a font family with the help of a \DeclareEncodingSubset declaration, see [5] for details.

Maintainers of font bundles that include TS1 encoded font files should add an appropriate declaration into the corresponding ts1\family.fd file, because otherwise the default subencoding is assumed, which is probably disabling too many glyphs that are actually available in the font.\footnote{github issue 905}

New or improved commands

Support for slanted small caps in the EC-fonts

Since some time \LaTeX{} supports the combination of the shapes small caps and italic/slanted. The EC-fonts contain slanted small caps fonts but using them required the loading of an external package. Suitable font definitions have now been added to t1cmd.fd and so from now on

\begin{verbatim}
\textsc{\textsl{slanted small}}
\textsc{\textit{italic small caps}}
\bfseries
\end{verbatim}

Changes to packages in the tools category

array: Correctly identify single-line m-cells

1The \LaTeX{} format contains declarations for many font families already. This was done in 2020 to quickstart the use of the symbols in the kernel, but it is really the wrong place for such declarations. Thus, for new fonts the declarations should be placed into the corresponding .fd files.
EC sans serif at small sizes
The EC (T1 encoded Computer Modern) sans serif fonts have errors at small sizes: the medium weight is bolder and wider than the bold extended. This makes them unusable at these small sizes. The default .fd file has therefore been adjusted to use a scaled down 8pt font instead. (github issue 879)

Detect nested minipage environments
Nesting of minipage environments is only partially supported in \LaTeX and can lead to incorrect output, such as overfull boxes or footnotes appearing in the wrong place; see [1, p. 106]. However, until now there was no warning if that happened. This has been changed and the environment now warns if you nest it in another minipage environment that already contains footnotes. (github issue 168)

\LaTeX callback efficiency improvement
The mechanism for providing the pre/post_mlist_to_hlist_filter callbacks in \LaTeX has been improved to make it more reusable and to avoid overhead if these callbacks are not used. (github issue 168)

Improve l3docstrip integration into docstrip
In 2020 we merged l3docstrip.tex into docstrip.tex to support the \%<@@= (module) syntax of expl3; see [2]. However, this support was incomplete, because it didn’t cover docstrip lines of the form \%<+,\ldots> or \%<+-,\ldots>. This was never noticed until now, because usually \%<+,\ldots> blocks are used. Now all lines in a .dtx file are subject to the @@ replacement approach. (github issue 830)

Improve font series handling with incorrect .fd files
By convention, the font series value is supposed to contain no m, unless you refer to the “medium” series (which is represented by a single m). For example, one should write c for “medium weight, condensed width” and not mc. This was one of the many space-conserving methods necessary in the early days of \LaTeX2ε.

Some older .fd files do not obey that convention but use mc, bm, etc., in their declarations. As a result, some font selection scheme functionality was not working when confronted with such .fd files. We have therefore augmented \DeclareSymbolFont and \SetSymbolFont to strip their series argument from any surplus m so that they do not unnecessarily trigger font substitutions. Regardless of this support such .fd files should get fixed by their maintainers. (github issue 918)

Bug fixes

Prevent \TeX from loosing a \smash
When \TeX is typesetting a fraction, it will rebox the material in either numerator or the denominator depending on which is wider. If the repackaged part consists of a single box, that box gets new dimensions and if it was built using a \smash that effect vanishes (because a smash is nothing other than zeroing some box dimension, which now got undone). For example, in the line
\[
\frac{1}{2} \neq \frac{1}{\smash{2^X}}
\]
the 2 in the denominators was not always at the same vertical position, because the second \smash was ignored due to reboxing:
\[
\frac{1}{2} \not \stackrel{\neq}{\smash{2^X}}
\]
The differences are subtle but noticeable. This is now corrected and the \smash is always honored. Thus now you get this output:
\[
\frac{1}{2} \not \stackrel{\neq}{2^X}
\]

Resolve an issue with \mathchoice and localalphabets
The code for keeping a number of math alphabets local (introduced in 2021; see [3]) used \aftergroup to do some cleanup actions after a formula had finished. Unfortunately, \aftergroup can’t be used inside the arguments of the \mathchoice primitive and as a result one got low-level errors if the freezing happened in such a place. The implementation was therefore revised to avoid the \aftergroup approach altogether. (github issue 921)

Changes to packages in the amsmath category

Changes to packages in the graphics category

Fix a \mathcolor bug
The \mathcolor command introduced in [4] needs to scan for following sub and superscripts, but if it did so at the end of an alignment cell, e.g., in a array environment, the k was evaluated too early causing some internal errors. This is now properly guarded for. (github issue 901)
Changes to packages in the tools category

array: Correctly identify single-line m-cells

Cells in m-columns that only contain a single line are supposed to behave like single-line p-cells and align at the same baseline. To test for the condition, \texttt{array} used to compare the height of the cell to the height of the strut used for the table rows. However, the height of that strut depends on the setting of \texttt{\textbackslash arraystretch} and if you made this negative (or very large) the test came out wrong. Therefore, we now test against the height of a normal strut to ensure that single-line cells are correctly identified as such (unless their content is truly very tall, in which case aligning is pointless anyway).

\texttt{(github issue 766)}

References


[2] \LaTeX{} Project Team: \textit{\LaTeX} 2ε news 32.  

[3] \LaTeX{} Project Team: \textit{\LaTeX} 2ε news 34.  
https://latex-project.org/news/latex2e-news/ltnews34.pdf

[4] \LaTeX{} Project Team: \textit{\LaTeX} 2ε news 35.  

[5] \LaTeX{} Project Team: \textit{\LaTeX} 2ε font selection.  
https://latex-project.org/help/documentation/