LaTeX News

Issue 33, June 2021 — Draft Version

Contents

Introduction

The focus of the June 2020 release is to provide further important building blocks for producing reliable tagged PDF output in the future (see [1]), they are discussed in the next two sections. In addition we included a number of smaller enhancement and fixes that are outlined on the next pages. As usual, more detail can on individual changes can be found in the `changes.txt` files in the distribution and, of course, in the documented sources [2].

Extending the hook concept to paragraphs

Largely triggered by the need for better control of paragraph text processing, in particular when producing tagged PDF output, we have extended the paragraph processing of LaTeX so that the kernel gains control both at the start and the end of each paragraph. This is done in a manner that is (or should be) transparent to packages and user documents.

Beside the internal control points for exclusive use of the LaTeX kernel we also implemented four public hooks that can be used by packages or user via the hook management declarations to achieve special effects or implement manipulations that in the past were only possible through redefinitions of `\everypar` or `\par` with the usual issue that such changes would conflict with changes in other packages.

The documentation of the hooks together with a few examples is provided in `ltpara-doc.pdf` and for those who want to study the (quite interesting) code is found

in `ltpara-code.pdf`. Additionally it is included as part of the full kernel documentation in `source2e.pdf`.

Extending the hook concept to commands

Up to now the hook management covered hooks for a few core areas, such as hooks for the `\shipout` process or those in the `document` environment, as well as generic hooks for file loading (helpful for patching) and for arbitrary environments (executed by `\begin` and `\end`).

This has now been extended to add `/before` and `/after` hooks to any (document-level) command—in theory at that is. In practice the new generic `cmd` hooks, especially the `cmd/.../after` hooks may fail with commands that are too complex to be automatically patched and break them if the hook is filled with code. The restrictions are documented in `ltcmdhooks-doc.pdf`.

However, given that these hooks are mainly meant for package developers to provide a better interoperability between different packages and between packages and the LaTeX kernel, these restrictions are of minor importance: for commands where the mechanism can't be applied, one is in the same situation as before and for all others there will be a noticeable improvement. This is especially important for our big project providing accessible and tagged PDF output [1], because for this we will eventually have to patch many third-party packages and this is only feasible, if it can be done in controlled and standardized ways.

Other changes to the LaTeX kernel

Adjusting `itemize` labels with `\labelitemfont`

The command `\labelitemfont` was in fact already introduced with the LaTeX release 2020-02-02, but back then we forgot to describe it, so we do this now. Its purpose is to resolve some bad formatting issues with the `itemize` environment and at the same time make it easier to adjust its layout if necessary. What could happen in the past was the `itemize` labels, e.g., the •, would sometimes react to surrounding font changes and could suddenly change shape, for example to •.

Now `\labelitemfont` is applied to each label defaulting to `\normalfont` which will prevent this behavior. By choosing a different settings other effects can be achieved, for example

```
\renewcommand\labelitemfont
   {\normalfont\fontfamily{lmss}\selectfont}
\renewcommand\labelitemfont
   {\rmfamily\normalshape}
```

The first will take the symbols from Latin Modern Sans so that you get ▪, −, ∗ and ·, while the second variant freezes the font family and shape, but leave the series variable, so that an `itemize` in a bold context would show bolder symbols. Making it empty would give you the buggy old behavior back. *(github issue 497)*

A note on file names with spaces, dots or UTF-8 characters

In one of the recent LaTeX releases we improved the interface for specifying file names so that they can now safely contain spaces (as is common on Windows but also elsewhere), UTF-8 characters outside the ASCII range as well as names with several dots in it. In the past this was only possible by applying a special syntax (in cases of spaces), not at all for most UTF-8 characters and not in all circumstances for files with several dots.

However, TeX has a built-in rule saying that you can leave out the extension if it is `.tex`. Because of that `\input{file}` or `\input{file.tex}` both load `file.tex` if it exists. While this is convenient most of the time it is a little awkward in some scenarios (for example, when both `file` and `file.tex` exist) and also when you manually try to implement that rule.

LaTeX therefore had one special syntax for `\include` and `\includeonly`: they always expected that their arguments contains a file name[1] without its extension, which had to be `.tex`. Thus when you mistakenly wrote `\include{mychap.tex}` (for example, when you changed from `\input` to `\include` somewhere), LaTeX went ahead and looked for the file `mychap.tex.tex` for inclusion and tried to write support information to the file `mychap.tex.aux`. The reason was that `\include` had to construct both physical file names from the argument and it didn't bother to do something special about the extension `.tex`.

As a side effect of the new implementation this has now changed and the argument of `\include` now gets the extension `.tex` removed if it was used. Thus `\include{mychap.tex}` now loads `mychap.tex` and no longer looks for `mychap.tex.tex`. *(github issue 486)*

`\end{document}` should always start in v-mode

Until now `\end{document}` executed the code from the `\AtEndDocument` hook as its first action. This meant that it was executed in horizontal mode if the user left no empty line after the last paragraph. As a result one could get a spurious space added, for example, when that code contained a `\write` statement. This was fixed and now `\enddocument` first issues a `\par` to ensure that it always starts out in vertical mode. *(github issue 385)*

Allow extra space between name and address in letter class

The `\opening` command in the letter class expects the name and address to be separated by `\\` but it didn't allow to use an optional argument at this point to add some extra space after the name. The coding has now been slightly altered to allow for this. *(github issue 427)*

[1] In case of `\includeonly` a comma separated list of such names.

Add a Lua callback to ltshipout to provide a uniform location for applying custom attributes

Just before shipping out a page, a new LuaTeX callback `pre_shipout_filter` is now called to allow final adjustments to the box to be shipped out. This is particularly for LuaTeX packages which flag certain elements of the page (e.g. using attributes or properties) in order to apply certain effects to these elements at shipout. An example for this is the luacolor package which could insert the color commands using this callback.

Improved copy & paste support for pdfTeX documents

When compiling with pdfTeX, additional information is added to the PDF file to improve copying from and searching in text. This especially allows ligatures to copy correctly from pdfTeX generated PDF files in most cases.

Since this has been integrated into the kernel, most documents should no longer need to load the cmap package or input `glyphtounicode`. *(github issue 465)*

Provide a hook in `\selectfont`

After `\selectfont` has altered the font we run a hook so that packages can make final adjustments. This functionality was originally provided by the everysel package, the new implementation is slightly different and uses the standard hook management. *(github issue 444)*

Delay change of font series and shape to `\selectfont` call

With the NFSS extensions introduced in 2020 the font series and shape settings can be influenced by changes to the font family. The setting is therefore delayed until `\selectfont` is executed to avoid unnecessary or incorrect substitutions that may otherwise happen due to the order of declarations. *(github issue 444)*

Allow `\nocite` in preamble

A natural place for `\nocite{*}` would be the preamble of the document, but for historical reasons LaTeX issued an error message if it was placed there. From the new release on it is now allowed in the preamble. *(github issue 424)*

Shipping out a page while bypassing hooks

In the 2020 October release several hooks were added to the page shipout process, e.g., to add some background or foreground material to some or all pages. We now also added a `\RawShipout` command that bypasses most of these hooks during the shipout. Some essential internal bookkeeping still takes place such as updating the `totalpages` counter or adding `shipout/firstpage` or `shipout/lastpage` material if the page happens to be the first or last.

Robust commands in filename arguments

The filename handling has been modified so that `\string` is applied while normalizing robust commands while determining the file name. Previously `\input{\sqrt{2}}` would cause LaTeX to loop indefinitely. With the new behavior it accesses `sqrt {2}.tex`. *(github issue 481)*

Additional support for Unicode characters from the Latin Extended Additional block

LaTeX is quite capable of typesetting characters such as "ṃ", but until now it lacked the Unicode mappings for some characters that are used to write Sanskrit words in Latin transliteration (as seen in books about yoga, Buddhist philosophy, etc.). These have now been added so that such characters can be entered directly instead of resorting to `\d{m}` and so forth. *(github issue 484)*

Always have color groups set up

To use color in LaTeX certain constructs, especially boxes, need an extra layer of groups to ensure that the color setting does not *escape* and continue outside the box when it shouldn't. To arrange for this the LaTeX kernel defined a number of commands, e.g., `\color@begingroup` to be used in such places. They have been initially no-ops and only the color packages redefined them to become real groups. This arrangement complicates the coding as one has to account for a group being there (or not there) depending of what is loaded in the document. So now the kernel already adds the groups. *(github issue 488)*

Execute `\par` at the end of `\marginpar` arguments

In preparation for tagged PDF it is important to properly tag all paragraphs and this requires running code at the beginning and end of each. At the end of a paragraph this is done inside the `\par` command, but the way `\marginpar` was coded, LaTeX ended the marginal note without ever explicitly calling `\par`. This has now been changed.

Another case where this issue caused problems was the lineno package where the last line was not numbered if the `\marginpar` ended without a `\par` in the document. *(github issue 489)*

Producing several footnote marks to one footnote

It is sometimes necessary to reference the same footnote several times, i.e., produce several footnote marks with the same number or symbol. This is now always possible by placing a `\label` into the `\footnote` and reference it with the command `\footref` elsewhere. This way marks referring to footnotes anywhere on the page (including those in `minipage`s) can be generated. In the past this command was only available with certain classes or when loading the footmisc package. *(github issue 482)*

Providing the raw option list of packages or documentclass to key/value handlers

LaTeX 2ε has always normalized space in option lists so `\documentclass[ a4paper , 12pt ]{article}` processed the intended options `a4paper` and `12pt`.

Unfortunately the mechanism used was designed for the simple option names of the standard option processing. Many classes and packages now use extended *keyval* processing, however this white space normalization makes this difficult: `[bb=1 2 3 4]` which might be expected to pass a bounding box of four numbers is normalized to `bb=1234` and `[bb={1 2 3 4}]` which might be expected to quote the spaces results in low level TeX parsing errors.

For compatibility reasons, the standard option processing has not been changed however the original un-normalized package and class option lists are now saved. They are not used in the standard processing, however extended package option systems may use these "raw" option list macros if they are defined.

The one change affecting the standard processing is that the low level error mentioned above is now avoided as values (any tokens to the right of an = sign) are removed from consideration from the "unused option list". In this release `clip=true` and `clip=false` both contribute `clip` to the list of options that have been used. *(github issue 85)*

Poor man's `\textasteriskcentered` if missing

The `\textasteriskcentered` symbol, used as part of the set of footnote symbols in LaTeX, is assumed to be implemented by every font in the `TS1` encoding (when pdfTeX is used) or in the `TU` encoding for the Unicode engines. Unfortunately, that assumption is not correct for all fonts, for example, for the `stix2` fonts don't offer the glyph, with the result that one gets missing glyphs when using `\thanks` etc.

For that reason the definition for `\textasteriskcentered` was altered to check if there is a glyph in the right position and if not a normal "*" is used, slightly enlarged and lowered. That may not be perfect in all cases, but certainly better than nothing show up. *(github issue 502)*

Provide more "dashes" in encodings `OT1`, `T1` and `TU`

When pasting in text from external sources one sometimes encounters the Unicode characters `"2011` (non-breaking hyphen), `"2012` (figure dash) and `"2015` (horizontal bar) in addition to the common `"2013` (en-dash) and `"2014` (em-dash). In the past the first three characters produced an error message when used with pdfTeX. Now they typeset an approximation (as they are unavailable in `OT1` or `T1` encoded fonts used by pdfTeX), e.g., the figure dash is approximated by an en-dash.

In Unicode engines they either work (if contained in the selected Unicode font) or typeset nothing and produce a "Missing character" warning in the log file.

However, what works in all engines now, is to access the characters via the command names `\textnonbreakinghyphen`, `\textfiguredash` and `\texthorizontalbar`, respectively. *(github issue 404)*

`filecontents` with UTF-8 characters in file name

Since a few releases back, the `filecontents` environment allows writing a file with UTF-8 characters in its name. However there was a bug that would not allow *over*writing a file with UTF-8 characters in the name. This has been fixed and now `filecontents` allows any characters in the file name. *(github issue 415)*

Extending latexrelease to declare an entire module

In the 2020-10-01 release, LaTeX's new hook management system was added to the kernel (see [3]) and, as with all changes to the kernel, it was added to latexrelease, so that it is possible to roll back to a date where such module didn't exist yet, or roll forward from an older release and have the hook management system by loading the latexrelease package.

However rolling back from a later release to the 2020-10-01 release didn't quite work because it would try to define all the commands from lthooks again, and that would result in errors, as usual with commands defined with `\newcommand` or in the case of lthooks, `\cs_new:Npn`.

To solve this issue, now completely new modules can be defined in latexrelease using `\NewModuleRelease` and then when rolling back or forward it will know if the code of the module has to be read or completely ignored. More details can be found in the latexrelease documentation (`texdoc latexrelease`). *(github issue 479)*

Small fix for rolling back prior to 2020-02-02

Whereas the latexrelease package can usually emulate an older LaTeX kernel without much problem, rolling back to before the 2020-02-02 release didn't work properly because the management of the `\ExplSyntaxOn/Off` status for packages cannot be removed by the rollback without messing up catcodes after an expl3-based package is loaded. This has been fixed and now rollback is more careful not to leave ExplSyntaxOn after a package ends. *(github issue 504)*

Add `\tracingstacklevels` and `\tracinglostchars=3` to `\tracingall`

In July 2020 David Jones suggested an extension to TeX engines, that added the possibility to set `\tracinglostchars=3` to have an error in case some character is missing from a font. In previous years, the warning for a missing character would be silently

printed to the `.log` file (if `\tracinglostchars` > 0) and to the terminal (if > 1). This extension was added for TeX Live and MiKTeX (except in Knuth's TeX, of course) and now with `\tracinglostchars` > 2 you get an error on a missing glyph.

Later, in January 2021, Petr Olšák suggested yet another extension, a new primitive parameter `\tracingstacklevels` that, when positive (and when `\tracingmacros` is also positive), will print a visual indication of the macro nesting level in TeX's tracing information.

Both these changes were incorporated to LaTeX's debugging macros `\tracingall` and `\tracingnone`, so when you use them, the new extensions are automatically activated/deactivated if your TeX distributions has a recent enough engine. An example document demonstrating these parameters is available in the linked GitHub issue. *(github issue 524)*

Make `\\` generally robust

In 2018 most LaTeX user-level commands were made robust including `\\`. However, `\\` is redefined in various environments and not all cases were caught, in particular its use as row delimiter in `tablular` structures. This has now been corrected and `\\` should be robust in all standard circumstances. Doing that also fixed one anomaly present in the past: in a tabular preamble of the form

```
{l>{raggedright}p{10cm}r}
```

a `\\` in the second column would have the definition used by `\raggedright` and would not indicate the (premature) end of the `tabular`, e.g.,

```
a & b1 \\ b2 & c \\
```

would be a single row in that `tabular`. However, writing

```
a &    \\ b2 & c \\
```

would give you two rows: due to the scanning process the `\\` directly following the `&` still had the "end the row" meaning and not the "start a new line in the second column" meaning.

With `\\` now robust, the scanning after `&` ends when the command is seen and the second column is started and so now both lines above consistently produce a single `tabular` row.

As before, you can use `\raggedright \arraybackslash` in the `tabular` preamble to ensure that `\\` is always interpreted as a row separator when used in the column or you could use `\tabularnewline` to explicitly ask for a new row even when `\\` has a different meaning in the current column. *(github issue 548)*

Changes to packages in the graphics category

Removed spurious warning for generic graphics rules

A previous release mistakenly caused a warning to appear when loading a graphics file with an unknown extension through a generic graphics rule. The warning would incorrectly say that the file was not found, whereas the file would be included correctly. The warning now doesn't show up in that case. *(github issue 516)*

Fixed loading `gzipped` PostScript graphics files

A previous release mistakenly changed the file searching mechanism and compressed graphics files would raise an error when being loaded with `\includegraphics`. This has been fixed and now `gzip`ped graphics load correctly. *(github issue 519)*

Changes to packages in the tools category

layout: Support extra language options

The package now recognizes `japanese` and `romanian` as language options. *(github issues 353 and 529)*

array and longtable: Make `\\` generally robust

The fix for this issue was also applied to these packages, see above. *(github issue 548)*

longtable: General bug fix update

Minor update to longtable to fix bugs reported. Notably the possibility of incorrect page breaks if floats appear on the same page that a `longtable` starts. As this may affect page breaking in existing documents, a rollback to longtable 4.13 (`longtable-2020-01-07.sty`) is supported. *(gnats issue tools/2914 3396 3512) (github issue 133 183 464)*

trace: Add `\tracingstacklevels` and `\tracinglostchars=3` to `\traceon`

The enhancement mentioned earlier was also added to the trace package. *(github issue 524)*

bm: Better support for commands with optional arguments

Some uses of optional arguments that were supported by `\bm` stopped being supported (in 2004) when `\kernel@ifnextchar` was used internally by the format instead of `\@ifnextchar`. This update handles both versions of this command and restores the original behaviour.

In addition package options for guiding the use of "poor man's bold" in fallback situations were added. *(github issue 554)*

Changes to packages in the amsmath category

The fix for issue 548 was also applied in amsmath, see above. *(github issue 548)*

References

[1] Frank Mittelbach and Chris Rowley: *LaTeX Tagged PDF — A blueprint for a large project.* https://latex-project.org/publications/indexbyyear/2020/

[2] *LaTeX documentation on the LaTeX Project Website.* https://latex-project.org/help/documentation/

[3] LaTeX Project Team: *LaTeX 2ε news 32.* https://latex-project.org/news/latex2e-news/ltnews32.pdf