

L^AT_EX3 News

Issue 12, January 2020 (L^AT_EX release 2020-01-01)

Contents

| | |
|---|----------|
| Introduction | 1 |
| New features in expl3 | 1 |
| A new argument specifier: <code>e-type</code> | 1 |
| New functions | 1 |
| String conversion moves to <code>expl3</code> | 1 |
| Case changing of text | 2 |
| Notable fixes and changes | 2 |
| File name parsing | 2 |
| Message formatting | 2 |
| Key inheritance | 2 |
| Floating point juxtaposition | 2 |
| Changing box dimensions | 2 |
| More functions moved to stable | 2 |
| Deprecations | 2 |
| Internal improvements | 2 |
| Cross-module functions | 2 |
| The backend | 2 |
| Better support for (u)pT_EX | 2 |
| Options | 3 |
| Engine requirements | 3 |
| Documentation | 3 |
| News | 3 |
| ChangeLog | 3 |
| Changes in xparse | 3 |
| New experimental modules | 3 |
| l3build changes | 3 |

Introduction

There has been quite a gap since the last *L^AT_EX3 News* (Issue 11, February 2018), and so there is quite a bit to cover here. Luckily, one of the things there *is* to cover is that we are using a more formalised approach for logging changes, so writing up what has happened is a bit easier. (By mistake L^AT_EX3 News 11 itself did not get *published* when written, but is now available: we have kept the information it contains separate as it is a good summary of the work that had happened in 2017.)

Work has continued apace across the L^AT_EX3 codebase in the last (nearly) two years. A lot of this is ultimately focussed on making the core of `expl3` even

more stable: *squeezing* out more experimental ideas, refining ones we have and making it a serious option for core L^AT_EX programming.

As a result of these activities, the L^AT_EX3 programming layer will be available as part of the kernel of L^AT_EX 2_ε from 2020-02-02 onwards, i.e., can be used without explicitly loading `expl3`. See *L^AT_EX News 31* [2] for more details on this.

New features in `expl3`

A new argument specifier: `e-type`

During 2018, the team worked with the T_EX Live, X_ƎT_EX and (u)pT_EX developers to add the `\expanded` primitive to pdfT_EX, X_ƎT_EX and (u)pT_EX. This primitive was originally suggested for pdfT_EX v1.50 (never released), and was present in LuaT_EX from the start of that project.

Adding `\expanded` lets us create a new argument specifier: `e-type` expansion. This is *almost* the same as `x-type`, but is itself expandable. (It also doesn't need doubled `#` tokens.) That's incredibly useful for creating function-like macros: you can ensure that *everything* is expanded in an argument before you go near it, with not an `\expandafter` in sight.

New functions

New programming tools have appeared in various places across `expl3`. The highlights are

- Shuffling of sequences to allow randomization
- Arrays of integers and floating point values; these have constant-time access
- Functions to return values after system shell usage
- Expandable access to file information, including file size, MD5 hash and modification date

For the latter, we have revised handling of file names considerably. There is now support for finding files in expansion contexts (by using the `\(pdf)filesize` primitive). Spaces and quotes in file names are now fully normalised, in a similar manner to the approach used by the latest L^AT_EX 2_ε kernel.

String conversion moves to `expl3`

In addition to entirely new functions, the team has moved the `l3str-convert` module from the `l3experimental` bundle into the `expl3` core. This module is essential for dealing with the need to produce UTF-16 and UTF-32 strings in some contexts, and also offers built-in escape for url and PDF strings.

Case changing of text

Within `expl3`, the team have renamed and reworked the ideas from `\tl_upper_case:n` and so on, creating a new module `l3text`. This is a “final” home for functions to manipulate *text*; token lists that can reasonably be expected to expand to plain text plus limited markup, for example emphasis and labels/references. Moving these functions, we have also made a small number of changes in other modules to give consistent names to functions: see the change log for full details.

Over time, we anticipate that functions for other textual manipulation will be added to this module.

Notable fixes and changes

File name parsing

The functions for parsing file names have been entirely rewritten, partly as this is required for the expandable access to file information mentioned above. The new code correctly deals with spaces and quote marks in file names and splits the path/name/extension.

Message formatting

The format of messages in `expl3` was originally quite text-heavy, the idea being that they would stand out in the `.log` file. However, this made them hard to find by a regular expression search, and was very different from the $\text{\LaTeX} 2_{\epsilon}$ message approach. The formatting of `expl3` messages has been aligned with that from the $\text{\LaTeX} 2_{\epsilon}$ kernel, such that IDE scripts and similar will be able to find and extract them directly.

Key inheritance

A number of changes have been made to the inheritance code for keys, to allow inheritance to work “as expected” in (almost) all cases.

Floating point juxtaposition

Implicit multiplication by juxtaposition, such as `2pi`, is now handled separately from parenthetic values. Thus for example `1in/1cm` is treated as equal to `(1in)/(1cm)` and thus yields `2.54`, and `1/2(pi+pi)` is equal to `pi`.

Changing box dimensions

\TeX ’s handling of boxes is subtly different from other registers, and this shows up in particular when you want to resize a box. To bring treatment of boxes, or rather the grouping behavior of boxes, into line with other registers, we have made some internal changes to how functions such as `\box_set_wd:N` are implemented. This will be transparent for “well-behaved” use cases of these functions.

More functions moved to stable

A large number of functions which were introduced as candidates have been evaluated and moved to stable status. The team hopes to move all functions in `expl3`

to stable status, or move them out of the core, over the coming months.

Deprecations

There have been two notable sets of deprecations over the past 18 months. First, we have rationalised all of the “raw” primitive names to the form `\tex_<name>:D`. This means that the older names, starting `\pdftex_...`, `\xetex_...`, etc., have been removed.

Secondly, the use of integer constants, which dates back to the earliest days of `expl3`, is today more likely to make the code harder to read than anything else. Speed improvements in engines mean that the tiny enhancements in reading such constants are no longer required. Thus for example `\c_two` is deprecated in favour of simply using `2`.

In parallel with this, a number of older `.sty` files have been removed. These older files provided legacy stubs for files which have now been integrated in the `expl3` core. They have now had sufficient time to allow users to update their code.

Internal improvements

Cross-module functions

The team introduced the idea of internal module functions some time ago. Within the kernel, there are places where functions need to be used in multiple modules. To make the nature of the kernel interactions clearer, we have worked on several aspects

- Reducing as far as possible cross-module functions
- Making more generally-useful functions public, for example scan marks
- Creating an explicit cross-kernel naming convention for functions which are internal but are essential to use in multiple kernel modules

The backend

Creating graphics, working with color, setting up hyperlinks and so on require backend-specific code. Here, backends are for example `dvips`, `xdvipdfmx` and the direct PDF mode in `pdf \TeX` and `Lua \TeX` . These functions are needed across the $\text{\LaTeX} 3$ codebase and have to be updated separately from the `expl3` core. To facilitate that, we have split those sources into a separate bundle, which can be updated as required.

At the same time, the code these files contain is very low-level and is best described as internal. We have re-structured how the entire set of functions are referred to such that they are now internal for the area they implement, for example image inclusion, box affine transformations, etc.

Better support for (u)p \TeX

The developers behind (u)p \TeX (Japanese \TeX) have recently enhanced their English documentation (see

<https://github.com/texjporg/ptex-manual>). Using this new information, we have been able to make internal adjustments to `expl3` to better support these engines.

Options

A new option `undo-recent-deprecations` is now available for cases where a document (or package) requires some `expl3` functions that have been formally removed after deprecation. This is to allow *temporary* work-arounds for documents to be compiled whilst code is begin updated.

The “classical” options for selecting backends (`dvips`, `pdfTeX`, etc.) are now recognised in addition to the native key–value versions. This should make it much easier to use the `expl3` image and color support as it is brought up to fully-workable standards.

Engine requirements

The minimum engine versions needed to use `expl3` have been incremented a little:

- `pdfTeX` v1.40
- `XƎTeX` v0.99992
- `LuaTeX` v0.95
- `ε-(u)pTeX` mid-2012

The team have also worked with the `XƎTeX` and `(u)pTeX` developers to standardise the set of post-`ε-TeX` utility primitives that are available: the so-called “`pdfTeX` utilities”. These are now available in all supported engines, and in time will all be *required*. This primarily impacts `XƎTeX`, which gained most of these primitives in the 2019 `TeX` Live cycle. (Examples are the random number primitives and expandable file data provision.) See *L^AT_EX News 31* [2] for more.

Documentation

News

The *L^AT_EX 3 News* files were until recently only used to create PDF files on the team website [1]. We have now integrated those into the `l3kernel` (`expl3` core) bundle. The news files cover all of `LATEX 3` files, as the core files are always available.

ChangeLog

Since the start of 2018, the team have commenced a comprehensive change log for each of the bundles which make up the `LATEX 3` code. These are simple Markdown text files, which means that they can be displayed formatted in web views.

Changes in `xparse`

A number of new features have been added to `xparse`. To allow handling of the fact that skipping spaces may be required only in some cases when searching

for optional arguments, a new modifier `!` is available in argument specifiers. This causes `xparse` to *require* that an optional argument follows immediately with no intervening spaces.

There is a new argument type purely for environments: `b`-type for collecting a `\begin... \end` pair, i.e., collecting the body of an environment. This is similar in concept to the `environ` package, but is integrated directly into `xparse`.

Finally, it is now possible to refer to one argument as the default for another optional one, for example `\NewDocumentCommand{\caption}{0{#2} m} ...`

New experimental modules

A number of new experimental modules have been added within the `l3experimental` bundle:

`l3benchmark` Performance-testing system using the timing function in modern `TeX` engines

`l3cctab` Category code tables for all engines, not just `LuaTeX`

`l3color` Color support, similar in interface to `xcolor`

`l3draw` Creation of drawings, inspired by `pgf`, but using the `LATEX 3` FPU for calculations

`l3pdf` Support for PDF features such as compression, hyperlinks, etc.

`l3sys-shell` Shell escape functions for file manipulation

`l3build` changes

The `l3build` tool for testing and releasing `TeX` packages has seen a number of incremental improvements. It is now available directly as a script in `TeX` Live and `MiKTTeX`, meaning you can call it simply as `l3build target`

Accompanying this, we have added support for installing scripts and script `man` files.

There is a new `upload` target that can take a zip file and send it to CTAN: you just have to fill in release information for *this* upload at the prompts.

Testing using PDF files rather than logs has been heavily revised: this is vital for work on PDF tagging.

There is also better support for complex directory structures, including the ability to manually specify TDS location for all installed files. This is particularly targeted at packages with both generic and format-specific files to install.

References

- [1] *L^AT_EX Project Website*. <https://latex-project.org/>
- [2] *L^AT_EX 2_ε release newsletters on the L^AT_EX Project Website*. <https://latex-project.org/news/latex2e-news/>