Move of sources from Subversion to Git

Version identifiers 1

expl3 updates and extensions 1

l3sort moves to the kernel 1

Boolean functions 1

Revision of l3file 1

Detection of \cs_generate_variant:Nn errors 1

Accessing random data 2

More powerful debugging 2

Mark-up changes in l3doc 2

l3build updates 2

Move of sources from Subversion to Git

The \LaTeX{} team have used a variety of version control systems over the life of the \LaTeX{}3 sources. For a long time we maintained the \LaTeX{}3 sources in Subversion (svn) but also provided a read-only clone of them on GitHub using SubGit from TMate Software [1] to synchronize the two repositories—a solution that worked very well.

We have now retired the Subversion repository and completely moved over to Git, with the master \LaTeX{}3 repository hosted on GitHub: https://github.com/latex3/latex3. This new approach means we are (slowly) adopting some new approaches to development, for example branches and accepting pull requests.

Version identifiers

Following this change, we have removed Subversion $Id$ lines from the \LaTeX{}3 sources. At present, we will be retaining \GetIdInfo as there are several possible use cases. The \LaTeX{}3 sources now have only release date strings as identifiers. However, the team recommend that package authors include version information directly in \ProvidesExplPackage (or similar) lines.

expl3 updates and extensions

Work has continued on the codebase over the last year, with both small changes/fixes and more substantial changes taking place. The following sections summarise some of the more notable changes.

l3sort moves to the kernel

Sorting is an important ability, and for some time the team have provided a stand-alone l3sort to support this. The functionality has seen wide take up, and so has now been integrated directly into the kernel. This took place in parallel with some interface changes to “round out” the code.

Boolean functions

For some time, the team have been aware that boolean expressions can fail in certain circumstances, leading to low-level errors. This is linked to two features of the long-standing \bool_if:n(TF) function: expandable operation and short-circuit evaluation.

Addressing that has meant two changes: altering \bool_if:n(TF) to always evaluate each part of the expression, and introducing new short-circuit functions without the issue. The latter are lazy in expl3 terms:

- \bool_lazy_all:n(TF)
- \bool_lazy_and:nn(TF)
- \bool_lazy_any:n(TF)
- \bool_lazy_or:nn(TF)

These new, stable functions are now the recommended way of handling boolean evaluations. Package authors are encouraged to employ these new functions as appropriate.

Revision of l3file

Large parts of l3file have been revised to give a better separation of path/file/extension. This has resulted in the addition of a number of new support functions and variables.

At the same time, new experimental functions have been added to utilise a number of useful primitives in pdf\TeX: \file_get_mdfive_hash:nN, \file_get_size:nN and \file_get_timestamp:nN. Currently, Xe\TeX does not support getting file size/timestamp information: this is available in other engines.

Paralleling these changes, we have added (experimental) support for shell escape to the l3sys module, most notably \sys_shell_now:n. A range of test booleans are also available to check whether shell escape is enabled.

Detection of \cs_generate_variant:Nn errors

The ability to generate variants is an important feature of expl3. At the same time, there are crucial aspects

of this approach that can be misunderstood by users. In particular, the requirement that variants map correctly to an underlying $N$- or $n$-type base function is sometimes misunderstood.

To help detect and correct these cases, $\texttt{\cs_generate_variant:Nn}$ now carries out error checking on its arguments, and raises a warning where it is mis-applied. At present, the team have avoided making this an error as it is likely to be seen by end users rather than directly by package developers. In time, we are likely to revisit this and tighten up further on this key requirement.

**Accessing random data**

To support randomised data selection, we have introduced a family of experimental functions which use underlying engine support for random values, and provide one entry at random from the data type. At the same time, we have addressed some issues with uniformity stemming from the random number function used by pdftex and inherited by other engines. This means that expl3’s FPU will generate pseudo-random values across the range of possible outputs.

**More powerful debugging**

A new set of debugging functions have been added to the kernel. These allow debug code to be enabled locally using the new option $\texttt{enable-debug}$ along with functions $\texttt{\debug_on:n}$ and $\texttt{\debug_off:n}$. Accompanying this change, we have improved the handling of global/local consistency in variable setting.

**Mark-up changes in l3doc**

Since the introduction of the $\__$ syntax to mark internal functions, the need for explicit markup of internal material in sources has been negated. As such, we have now dropped the requirement to mark internal material with $\texttt{[aux]}$ when using l3doc. Instead, the status of functions and variables is auto-detected from the presence of $\__$. For cases where non-standard names are used for internal code, the mark-up $\texttt{[int]}$ is retained, e.g.

\begin{macro}\[int]\{l@expl@enable@debug@bool\}