

The l3docstrip package

Code extraction and manipulation

The L^AT_EX3 Project*

Released 2018/03/05

1 Extending DocStrip

The l3docstrip module adds L^AT_EX3 extensions to the DocStrip program for extracting code from .dtx. As such, this documentation should be read along with that for DocStrip.

2 Internal functions and variables

An important consideration for L^AT_EX3 development is separating out public and internal functions. Functions and variables which are private to one module should not be used or modified by any other module. As T_EX does not have any formal namespacing system, this requires a convention for indicating which functions in a code-level module are public and which are private.

Using l3docstrip allows internal functions to be indicated using a “two part” system. Within the .dtx file, internal functions may be indicated using @@ in place of the module name, for example

```
\cs_new_protected:Npn \@@_some_function:nn #1#2
{
  % Some code here
}
\tl_new:N \l_@@_internal_tl
```

To extract the code using l3docstrip, the “guard” concept used by DocStrip is extended by introduction of the syntax %<@@=(*module*)>. The *module* name then replaces the @@ when the code is extracted, so that

```
%<*package>
%<@@=foo>
\cs_new_protected:Npn \@@_some_function:nn #1#2
{
  % Some code here
}
\tl_new:N \l_@@_internal_tl
%</package>
```

*E-mail: latex-team@latex-project.org

is extracted as

```
\cs_new_protected:Npn \__foo_some_function:nn #1#2
{
  % Some code here
}
\tl_new:N \l__foo_internal_tl
```

where the `__` indicates that the functions and variables are internal to the `foo` module.

Use `@@@` to obtain `@` in the output (`@@@@` to get `@`). For longer pieces of code the replacement can be completely suppressed by giving an empty module name, namely using the syntax `%<@=>`.