penlight

Lua libraries for use in LuaLaTeX

Kale Ewasiuk (`kalekje@gmail.com`)

2022–04–16

The official documentation for the Lua library can be found here:
`https://lunarmodules.github.io/Penlight`

Required Package Option

The first option sent to this package MUST be one of:
`[penlight]`   or   `[pl]`.
All Penlight sub-modules are then available under this global variable by either
`penlight.XYZ` or `pl.XYZ`

texlua usage

If you want to use Penlight (and extras) with the `texlua` intrepreter (no document made,
only for Lua files, useful for testing), you can access it by setting `__SKIP_TEX__ = true`
and adding the package to path. For example:

```
package.path = package.path .. ';'..'path/to/texmf/tex/latex/penlight/?.lua'
penlight = require('penlight')
-- below is optional
__SKIP_TEX__ = true  --only required if you want to use
                     --penlightextras without a LaTeX run
-- __PL_NO_GLOBALS__ = true -- optional, skip global definitions
require('penlightextras')
```

Additional Package Options

stringx	will import additional string functions into the string meta table. this will be ran in pre-amble: `require('pl.stringx').import()`
format	allows % operator for Python-style string formating this will be ran in pre-amble: `require('pl.stringx').format_operator()` `https://lunarmodules.github.io/Penlight/libraries/pl.stringx.html#format_operator`
func	allows placehold expressions eg. `_1+1` to be used this will be ran in pre-amble: `penlight.utils.import('pl.func')` `https://lunarmodules.github.io/Penlight/libraries/pl.func`
extrasnoglobals	does the above three (`func,stringx,format`); adds some additional functions to `penlight` module; and adds the `pl.tex` sub-module.
extras	does the above `extrasnoglobals` but makes many of the functions global variables.

Extras

If `extras` is used, the following Lua globals will be defined:

Misc stuff

`__SKIP_TEX__`	If using package with `texlua`, set this global before loading `penlight`
`__PL_NO_GLOBALS__`	If using package with `texlua` and you don't want to set some globals (described in next sections), set this global before to `true` loading `penlight`
`hasval(x)`	Python-like boolean testing
`COMP'xyz'()`	Python-like comprehensions: `https://lunarmodules.github.io/Penlight/libraries/pl.comprehension.html`
`math.mod(n,d)`, `math.mod2(n)`	math modulous
`string.totable(s)`	string a table of characters
`kpairs(t)`, `npairs(t)`	iterate over keys only, or include nil value from table ipairs

`pl.utils.filterfiles(dir,filt,rec)`	Get files from dir and apply glob-like filters. Set rec to `true` to include sub directories

`pl.tex.` module is added

`add_bkt_cnt(n)`, `close_bkt_cnt(n)`, `reset_bkt_cnt` functions to keep track of adding curly brackets as strings. `add` will return `n` (default 1) {'s and increment a counter. `close` will return `n` }'s (default will close all brackets) and decrement.

`_NumBkts` internal integer for tracking the number of brackets

`opencmd(cs)` prints `\cs` { and adds to the bracket counters.

`_xNoValue,_xTrue,_xFalse`: `xparse` equivalents for commands

`prt(x)`,`prtn(x)` print without or with a newline at end. Tries to help with special characters or numbers printing.

`prtl(l)`,`prtt(t)` print a literal string, or table

`wrt(x)`, `wrtn(x)` write to log

`help_wrt(s1, s2)` pretty-print something to console. S2 is a flag to help you find.

`prt_array2d(tt)` pretty print a 2d array

`pkgwarn(pkg, msg1, msg2)` throw a package warning

`pkgerror(pkg, msg1, msg2, stop)` throw a package error. If stop is true, immediately ceases compile.

`defcmd(cs, val)` like `\gdef`

`newcmd(cs, val)` like `\newcommand`

`renewcmd(cs, val)` like `\renewcommand`

`prvcmd(cs, val)` like `\providecommand`

`deccmd(cs, dft, overwrite)` declare a command. If `dft` (default) is `nil`, `cs` is set to a package warning saying `'cs' was declared and used in document, but never set`. If `overwrite` is true, it will overwrite an existing command (using `defcmd`), otherwise, it will throw error like `newcmd`.

global extras

If `extras` is used and NOT `extrasnoglobals`, then some globals are set.

All `pl.tex` modules are added.

`hasval`, `COMP`, `kpairs`, `npairs` are globals.

`pl.tablex` functions are added to the `table` table.