The luatexbase-loader package

Heiko Oberdiek (primary author of luatex)
Élie Roux, Manuel Pégourié-Gonnard, Philipp Gesang*

https://github.com/lualatex/luatexbase
lualatex-dev@tug.org

v0.6 2013-05-11

Abstract

Lua modules are loaded using the `require()` function which, similarly to TeX's `\input`, takes care of locating the file to be loaded. This package adapts the way the files are searched in order to accommodate the TDS as well as usual Lua naming conventions.

For higher-level functions related to Lua modules, see luatexbase-modutils, which also loads the present package.

1 Documentation

Starting with LuaTeX 0.45, `require()` uses Kpathsea for file searching when the library is initialised (which is always the case in TeX mode, unless explicitly disabled by the user). However, it did not respect the Lua convention that `require("foo.bar")` should look for `foo/bar.lua` until version 0.60. LuaTeX 0.74 ships with Lua 5.2 that has a different loading system.

The aim of this package is to have a coherent behaviour between versions of LuaTeX, and to get the loaded file's name printed in the output (LaTeXstyle). The first versions did ensure backward compatibilty to LuaTeX 0.25 but as LuaTeX development is quite fast, this version supports only LuaTeX 0.45 and higher.

More precisely, when asked for `foo.bar` (or `foo.bar.lua`), it first looks for file `foo/bar` using Kpathsea with the format `lua`, and then for `foo.bar`, removing the possible extension.

2 Implementation

2.1 TeX package

```
1 ⟨∗texpackage⟩
```

2.1.1 Preliminaries

Catcode defenses and reload protection.

```
2 \begingroup\catcode61\catcode48\catcode32=10\relax% = and space
3   \catcode123 1 % {
4   \catcode125 2 % }
```

---

*See "History" in luatexbase.pdf for details.

1

```
 5    \catcode 35 6 % #
 6    \toks0\expandafter{\expandafter\endlinechar\the\endlinechar}%
 7    \edef\x{\endlinechar13}%
 8    \def\y#1 #2 {%
 9      \toks0\expandafter{\the\toks0 \catcode#1 \the\catcode#1}%
10      \edef\x{\x \catcode#1 #2}}%
11   \y  13  5 % carriage return
12   \y  61 12 % =
13   \y  32 10 % space
14   \y 123  1 % {
15   \y 125  2 % }
16   \y  35  6 % #
17   \y  64 11 % @ (letter)
18   \y  10 12 % new line ^^J
19   \y  34 12 % "
20   \y  39 12 % '
21   \y  40 12 % (
22   \y  41 12 % )
23   \y  44 12 % ,
24   \y  45 12 % -
25   \y  46 12 % .
26   \y  47 12 % /
27   \y  58 12 % :
28   \y  91 12 % [
29   \y  93 12 % ]
30   \y  94  7 % ^
31   \y  95  8 % _
32   \y  96 12 % `
33   \toks0\expandafter{\the\toks0 \relax\noexpand\endinput}%
34   \edef\y#1{\noexpand\expandafter\endgroup%
35     \noexpand\ifx#1\relax \edef#1{\the\toks0}\x\relax%
36     \noexpand\else \noexpand\expandafter\noexpand\endinput%
37     \noexpand\fi}%
38 \expandafter\y\csname luatexbase@loader@sty@endinput\endcsname%
```

Package declaration.

```
39 \begingroup
40   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
41     \def\x#1[#2]{\immediate\write16{Package: #1 #2}}
42   \else
43     \let\x\ProvidesPackage
44   \fi
45 \expandafter\endgroup
46 \x{luatexbase-loader}[2013/05/11 v0.6 Lua module loader for LuaTeX]
```

Make sure LuaTEX is used.

```
47 \begingroup\expandafter\expandafter\expandafter\endgroup
48 \expandafter\ifx\csname RequirePackage\endcsname\relax
49   \input ifluatex.sty
50 \else
51   \RequirePackage{ifluatex}
52 \fi
53 \ifluatex\else
54   \begingroup
```

```
55    \expandafter\ifx\csname PackageError\endcsname\relax
56      \def\x#1#2#3{\begingroup \newlinechar10
57        \errhelp{#3}\errmessage{Package #1 error: #2}\endgroup}
58    \else
59      \let\x\PackageError
60    \fi
61    \expandafter\endgroup
62    \x{luatexbase-loader}{LuaTeX is required for this package. Aborting.}{%
63      This package can only be used with the LuaTeX engine^^J%
64      (command 'lualatex' or 'luatex').^^J%
65      Package loading has been stopped to prevent additional errors.}
66    \expandafter\luatexbase@loader@sty@endinput%
67 \fi
```

2.1.2 Main content

First load luatexbase-compat.

```
68 \begingroup\expandafter\expandafter\expandafter\endgroup
69 \expandafter\ifx\csname RequirePackage\endcsname\relax
70   \input luatexbase-compat.sty
71 \else
72   \RequirePackage{luatexbase-compat}
73 \fi
```

Load the supporting Lua module. This one doesn't follow the usual naming conventions, since it won't be loaded with the usual functions for obvious bootstraping reasons.

```
74 \luatexbase@directlua{%
75   local file = "luatexbase.loader.lua"
76   local path = assert(kpse.find_file(file, 'lua'),
77     "File '"..file.."' not found")
78   texio.write_nl("("..path..")")
79   dofile(path)}
```

That's all, folks!

```
80 \luatexbase@loader@sty@endinput%
81 ⟨/texpackage⟩
```

2.2 Lua module

```
82 ⟨*luamodule⟩
83 luatexbase          = luatexbase or { }
84 local luatexbase    = luatexbase
```

Just in case it's called from a TEXLua script...
```
85 kpse.set_program_name("luatex")
```

In LATEX, it's traditional to print the included file paths. We don't want to do this for scripts using texlua... Currently there is no perfect check of texlua vs. luatex, so we check for the token table.
```
86 local print_included_path = false
87 if token then
88   print_included_path = true
89 end
```

Emulate (approximatively) kpse's lua format search. More precisely, combine the search path of `texmfscripts` and `tex` in order to approximate `LUAINPUTS`. But we need to handle suffixes ourselves.

`lua_suffixes` is taken verbatim from Kpathsea's source (`tex-file.c`, constant `LUA_SUFFIXES`).[1]

```
90 local lua_suffixes = {
91   ".luc", ".luctex", ".texluc", ".lua", ".luatex", ".texlua",
92 }
```

Auxiliary function for suffixes: says if `suffix` is a suffix of `name`.

```
93 local function ends_with(suffix, name)
94    return name:sub(-suffix:len()) == suffix
95 end
```

Auxiliary function for suffixes: returns the basename of a file if it end by one of the suffixes.

```
96 local function basename(name)
97   for _, suffix in ipairs(lua_suffixes) do
98     if ends_with(suffix, name) then
99       return name:sub(1, -(suffix:len()+1))
100     end
101   end
102   return name
103 end
```

The main function, emulating the behaviour of `packages.searchers[2]`, with a small improvement that eliminates the possible extension.

```
104 local function find_module_file(mod)
105   local compat = basename(mod):gsub('%.', '/')
106   return kpse.find_file(compat, 'lua') or kpse.find_file(mod, 'lua')
107 end
```

Combined searcher, using primarily the new kpse searcher, and the original as a fall-back. Starting from LuaTeX 0.75, Lua 5.2 is used. Among the changes, `package.loaders` is renamed as `package.searchers`.

The main improvement is thus the printing of the filename in the output.

```
108 local package_loader_two
109 if not package.searchers then
110   package.searchers = package.loaders
111 end
112 package_loader_two = package.searchers[2]
113 local function load_module(mod)
114   local file = find_module_file(mod)
115   if not file then
116     local msg = "\n\t[luatexbase.loader] Search failed"
117     local ret = package_loader_two(mod)
118     if type(ret) == 'string' then
119       return msg..ret
120     elseif type(ret) == 'nil' then
121       return msg
122     else
123       return ret
124     end
125   end
126   local loader, error = loadfile(file)
```

---

[1]Last checked 2013-04-12.

```
127  if not loader then
128    return "\n\t[luatexbase.loader] Loading error:\n\t"..error
129  end
130  if print_included_path then
131    texio.write_nl("("..file..")")
132  end
133  return loader
134 end
```

Finally install this combined loader as loader 2.

```
135 package.searchers[2] = load_module
136 ⟨/luamodule⟩
```

3   Test files

A dummy lua file for tests.

```
137 ⟨*testdummy⟩
138 return true
139 ⟨/testdummy⟩
```

Check that the package loads properly, under both LaTeX and Plain TeX, and load a dummy module in the current diretory.

```
140 ⟨testplain⟩\input luatexbase-loader.sty
141 ⟨testlatex⟩\RequirePackage{luatexbase-loader}
142 ⟨*testplain, testlatex⟩
143 \catcode64 11
144 \luatexbase@directlua{require "test-loader"}
145 \luatexbase@directlua{require "test-loader.sub"}
146 ⟨/testplain, testlatex⟩
147 ⟨testplain⟩\bye
148 ⟨testlatex⟩\stop
```