

The luatexbase-loader package

Manuel Pégourié-Gonnard
mpg@elzevir.fr

Élie Roux
elie.roux@telecom-bretagne.eu

v0.2a 2010-05-27

Abstract

Lua modules are loaded using the `require()` function which, similarly to \TeX 's `\input`, takes care of locating the file and load it, but also makes a few supplementary checks, for example to avoid loading the same module twice. 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 Lua \TeX 0.45.0, `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 does not respect the Lua convention that `require("foo.bar")` should look for `foo/bar.lua`.¹ This package implements such behaviour.

More precisely, it implements a new kpse searcher that looks for file `foo/bar` using Kpathsea with the format `lua` (that is, search along `LUAINPUTS` and try the following extensions: `.luc`, `.luctex`, `.texluc`, `.lua`, `.luatex`, `.texlua`). If this search fails, it falls back to `foo.bar` (along the same path with the same extensions).

Also, older versions of Lua \TeX , such as 0.25.4 (\TeX Live 2008), don't know about the `lua` format for kpse searching. So, an emulator for this function is provided. The emulator is not perfect, in particular it may find more results than the normal `lua` format search.² In order to ensure more homogeneous results across versions, this emulator is used as a fall-back when the real `lua` format search doesn't find any result.

Finally, a combined version of this new kpse searcher and the original function at `package.loaders[2]` (using first the new loader, then the old one if the new doesn't return any result) is installed as `package.loaders[2]`.

2 Implementation

2.1 \TeX package

¹Support for that has been added in rev 3558 of Lua \TeX , currently unreleased but probably part of Lua \TeX 0.54.

²An may also fail to find the file in particular cases, see comments in the implementation for details.

```
1 (*texpackage)
```

2.1.1 Preliminaries

Reload protection, especially for Plain T_EX.

```
2          \csname lltxb@loader@loaded\endcsname
3 \expandafter\let\csname lltxb@loader@loaded\endcsname\endinput
```

Catcode defenses.

```
4 \begingroup
5   \catcode123 1 % {
6   \catcode125 2 % }
7   \catcode 35 6 % #
8   \toks0{}%
9   \def\x{}%
10  \def\y#1 #2 {%
11    \toks0\expandafter{\the\toks0 \catcode#1 \the\catcode#1}%
12    \edef\x{\x \catcode#1 #2}}%
13  \y 123 1 % {
14  \y 125 2 % }
15  \y 35 6 % #
16  \y 10 12 % ^^J
17  \y 34 12 % "
18  \y 36 3 % $ $
19  \y 39 12 % '
20  \y 40 12 % (
21  \y 41 12 % )
22  \y 42 12 % *
23  \y 43 12 % +
24  \y 44 12 % ,
25  \y 45 12 % -
26  \y 46 12 % .
27  \y 47 12 % /
28  \y 60 12 % <
29  \y 61 12 % =
30  \y 64 11 % @ (letter)
31  \y 62 12 % >
32  \y 95 12 % _ (other)
33  \y 96 12 % `
34  \edef\y#1{\endgroup\edef#1{\the\toks0\relax}\x}%
35 \expandafter\y\csname lltxb@loader@AtEnd\endcsname
```

Package declaration.

```
36 \begingroup
37   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
38     \def\x#1[#2]{\immediate\write16{Package: #1 #2}}
39   \else
40     \let\x\ProvidesPackage
41   \fi
42 \expandafter\endgroup
43 \x{luatexbase-loader}[2010/05/27 v0.2a Lua module loader for LuaTEX]
```

Make sure Lua_T_EX is used.

```
44 \begingroup\expandafter\expandafter\expandafter\endgroup
45 \expandafter\ifx\csname RequirePackage\endcsname\relax
46   \input ifluatex.sty
47 \else
48   \RequirePackage{ifluatex}
49 \fi
50 \ifluatex\else
51   \begingroup
52     \expandafter\ifx\csname PackageWarningNoLine\endcsname\relax
53       \def\x#1#2{\begingroup\newlinechar10
54         \immediate\write16{Package #1 warning: #2}\endgroup}
55     \else
56       \let\x\PackageWarningNoLine
57     \fi
58   \expandafter\endgroup
59   \x{luatexbase-loader}{LuaTeX is required for this package. Aborting.}
60   \lltxb@loader@AtEnd
61   \expandafter\endinput
62 \fi
```

2.1.2 Main content

First load luatexbase-compat.

```
63 \begingroup\expandafter\expandafter\expandafter\endgroup
64 \expandafter\ifx\csname RequirePackage\endcsname\relax
65   \input luatexbase-compat.sty
66 \else
67   \RequirePackage{luatexbase-compat}
68 \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 bootstrapping reasons.

```
69 \luatexbase@directlua{%
70   local file = "luatexbase.loader.lua"
71   local path = assert(kpse.find_file(file, 'tex'),
72     "File '..file..' no found")
73   texio.write_nl("(..path..)")
74   dofile(path)}
```

That's all, folks!

```
75 \lltxb@loader@AtEnd
76 \</texpackage>
```

2.2 Lua module

```
77 <luamodule>
78 module('luatexbase', package.seeall)
```

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`),³.

```

79 local lua_suffixes = {
80   ".luc", ".luctex", ".texluc", ".lua", ".luatex", ".texlua",
81 }

```

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

```

82 local function ends_with(suffix, name)
83   return name:sub(-suffix:len()) == suffix
84 end

```

The search function first builds the list of filenames to be search. For the lua format, kpse always adds a suffix if no (known) suffix is present, so we do the same.

```

85 function find_file_lua_emul(name)
86   local search_list = {}
87   for _, suffix in ipairs(lua_suffixes) do
88     if ends_with(suffix, name) then
89       search_list = { name }
90       break
91     else
92       table.insert(search_list, name..suffix)
93     end
94   end

```

Now look for each file in this list.

```

95   for _, search_name in ipairs(search_list) do
96     local f = kpse.find_file(search_name, 'texmfscripts')
97     or kpse.find_file(search_name, 'tex')

```

There is a problem with using the `tex` search format: kpse will try to add suffixes from the `TEX_SUFFIXES` list, which may lead to problems if a file like `<name>.lua.tex` exists. We prevent that by checking if the file found ends with the required name. So `<name>.lua` will still be hidden by `<name>.lua.tex` but it seems less bad not to find it than to return an incorrect result.

```

98     if f and ends_with(search_name, f) then
99       return f
100     end
101   end
102 end

```

If lua search format is available, use it with emulation as a fall-back, or just use emulation.

```

103 local find_file_lua
104 if pcall('kpse.find_file', 'dummy', 'lua') then
105   find_file_lua = function (name)
106     return kpse.find_file(name, 'lua') or find_file_lua_emul(name)
107   end
108 else
109   find_file_lua = function (name)
110     return find_file_lua_emul(name)

```

³Unchanged since 2007-07-06, last checked 2010-05-10.

```

111 end
112 end

```

Find the full path corresponding to a module name.

```

113 local function find_module_file(mod)
114   return find_file_lua(mod:gsub('%.', '/'), 'lua')
115   or find_file_lua(mod, 'lua')
116 end

```

Combined searcher, using primarily the new kpse searcher, and the original as a fall-back.

```

117 local package_loader_two = package.loaders[2]
118 local function load_module(mod)
119   local file = find_module_file(mod)
120   if not file then
121     local msg = "\n\t[luatexbase.loader] Search failed"
122     local ret = package_loader_two(mod)
123     if type(ret) == 'string' then
124       return msg..ret
125     elseif type(ret) == 'nil' then
126       return msg
127     else
128       return ret
129     end
130   end
131   local loader, error = loadfile(file)
132   if not loader then
133     return "\n\t[luatexbase.loader] Loading error:\n\t"..error
134   end
135   texio.write_nl("(..file..)")
136   return loader
137 end

```

Finally install this combined loader as loader 2.

```

138 package.loaders[2] = load_module
139 </luamodule>

```

3 Test files

A dummy lua file for tests.

```

140 <*testdummy>
141 return true
142 </testdummy>

```

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

```

143 <testplain>\input luatexbase-loader.sty
144 <testlatex>\RequirePackage{luatexbase-loader}
145 <*testplain,testlatex>
146 \catcode64 11
147 \luatexbase@directlua{require "test-loader"}
148 \luatexbase@directlua{require "test-loader.sub"}

```

```
149 </testplain, testlatex>  
150 <testplain>\bye  
151 <testlatex>\stop
```