

The lt3luabridge package: Lua without LuaTeX

Vít Starý Novotný*

Released 2024-07-03

The `lt3luabridge` `expl3` [2] package provides support for executing Lua code in LuaTeX or any other TeX engine that exposes the shell. The package provides interfaces to plain TeX, L^ATeX, and ConTeXt formats:

```
\documentclass{standalone}
\usepackage{lt3luabridge}
\begin{document}
$ 1 + 2 = \luabridgeExecute{ print(1 + 2) } $
\end{document}
```

The package was previously part of the Markdown package [1], where it has been battle-tested since 2016. Since 2022, `lt3luabridge` has also been available as a separate package.

1 Loading the package

Use the `\input lt3luabridge\relax` command to load the package from plain TeX, use the `\usepackage{lt3luabridge}` command to load the package from L^ATeX, and use the `\usemodule[t]{lt3luabridge}` command to load the package from ConTeXt.

2 Executing Lua code

The interface for executing Lua code mimics the `\lua_now:n` function from `l3luatex`.

`\luabridge_now:n` `\luabridge_now:n` $\langle token list \rangle$

`\luabridge_now:e`

New: 2022-06-26

Updated: 2022-07-31

The $\langle token list \rangle$ is first tokenized by TeX, which includes converting line ends to spaces in the usual TeX manner and which respects currently-applicable TeX category codes. The resulting $\langle Lua input \rangle$ is passed to the Lua interpreter for processing. Each `\luabridge_now:n` block is treated by Lua as a separate chunk. The Lua interpreter executes the $\langle Lua input \rangle$ immediately, and in an expandable manner.

Unlike `\lua_now:n`, `\luabridge_now:n` may execute $\langle Lua input \rangle$ in a separate process from TeX. Therefore, you should not interact with TeX from $\langle Lua input \rangle$ or create global variables. The only exception is the standard output produced by the `print()` Lua function like in the example at the top of this page. The standard output of `print()` will be inserted into TeX's input stream.

*E-mail: witiko@mail.muni.cz

`\luabridgeExecute` `\luabridgeExecute` $\langle token list \rangle$
New: 2022-06-26 The `\luabridgeExecute` document command aliases the `\luabridge_now:e` function.
Updated: 2022-07-31

`\luabridge_tl_set:Nn` `\luabridge_tl_set:Nn` $\langle tl var \rangle \langle token list \rangle$
New: 2024-02-14 Like `\lua_now:n` but the result of executing the Lua code is stored in $\langle tl var \rangle$ instead of being inserted into \TeX 's input stream.

3 Setting and getting the method to execute Lua code

There are several methods that can be used to execute Lua code. This section describes the interface that the package provides to set the preferred method or to determine which method was used.

`\g_luabridge_method_int` This variable controls the method used to execute Lua code. The variable is set automatically when the package is loaded and changing the value of the variable afterwards has no effect. However, we can set the value of the variable before loading the package to one of the constants described below.

`\c_luabridge_method_shell_int`
New: 2022-07-31
Use shell escape through the `\write18` \TeX command to execute Lua code.

`\c_luabridge_method_directlua_int`
New: 2022-06-26
Use the `\directlua` primitive of $\text{Lua}\TeX$ to execute Lua code.

4 Setting and getting the filenames of helper files

When shell escape is used to execute Lua code, several helper files are needed to shuffle around code and output. The following variables and constants are undefined when the `\directlua` primitive of $\text{Lua}\TeX$ is used to execute Lua code.

`\g_luabridge_output_dirname_str`
New: 2022-06-26
This variable controls the output directory that will store the helper files. The variable should be set to the same value as the `-output-directory` parameter of the \TeX engine.

`\c_luabridge_default_output_dirname_str`
New: 2022-06-26
Updated: 2024-07-03
This constant is the default value of `\g_luabridge_output_dirname_str`.

`\g_luabridge_helper_script_filename_str`

New: 2022-06-26

This variable controls the filename of a helper Lua script that will be executed from the shell using the \TeX Lua interpreter.

`\c_luabridge_default_helper_script_filename_str`

New: 2022-06-26

This constant is the default value of `\g_luabridge_helper_script_filename_str`.

`\g_luabridge_error_output_filename_str`

New: 2022-06-26

This variable controls the filename of a helper file that will contain the error output produced by the `texlua` interpreter (if any).

`\c_luabridge_default_error_output_filename_str`

New: 2022-06-26

This constant is the default value of `\g_luabridge_error_output_filename_str`.

5 Plain \TeX implementation

This section contains the implementation for plain \TeX using generic `expl3`.

```
1 <@@=luabridge>
2 <*generic-package>
3 \ifx\ExplSyntaxOn\undefined
4   \input expl3-generic\relax
5 \fi
6 \ExplSyntaxOn
7 \int_const:Nn
8   \c_luabridge_method_directlua_int
9   { 0 }
10 \int_const:Nn
11   \c_luabridge_method_shell_int
12   { 1 }
13 \int_if_exist:NF
14   \g_luabridge_method_int
15   {
16     \int_new:N
17       \g_luabridge_method_int
18     \sys_if_engine luatex:TF
19       {
20         \int_gset_eq:NN
21           \g_luabridge_method_int
22           \c_luabridge_method_directlua_int
23       }
24     {
25       \int_gset_eq:NN
26         \g_luabridge_method_int
```

```

27         \c_luabridge_method_shell_int
28     }
29 }
30 \msg_new:nnn
31 { luabridge }
32 { method-shell }
33 {
34     Using-shell-escape-as-the-bridging-method
35 }
36 \msg_new:nnn
37 { luabridge }
38 { method-directlua }
39 {
40     Using-direct-Lua-access-as-the-bridging-method
41 }
42 \msg_new:nnn
43 { luabridge }
44 { unknown-method }
45 {
46     Unknown-bridging-method:~#1
47 }
48 \int_case:nnF
49 { \g_luabridge_method_int }
50 {
51     { \c_luabridge_method_shell_int }
52     {
53         \msg_info:nn
54         { luabridge }
55         { method-shell }
56     }
57     { \c_luabridge_method_directlua_int }
58     {
59         \msg_info:nn
60         { luabridge }
61         { method-directlua }
62     }
63 }
64 {
65     \cs_generate_variant:Nn
66     \msg_error:nnn
67     { nnV }
68     \msg_error:nnV
69     { luabridge }
70     { unknown-method }
71     \g_luabridge_method_int
72 }
73 \int_compare:nNnT
74 { \g_luabridge_method_int }
75 =
76 { \c_luabridge_method_shell_int }
77 {

```

Instead of assuming the current working directory as the output directory, try to determine the output directory from the environmental variable `TEXMF_OUTPUT_DIRECTORY`,

which is automatically defined by T_EX engines and accessible from child processes.

```

78  \sys_if_platform_unix:TF
79  {
80    \str_const:Nn
81    \c_luabridge_default_output_dirname_str
82    { $TEXMF_OUTPUT_DIRECTORY }
83  }
84  {
85    \sys_if_platform_windows:TF
86    {
87      \str_set:Nn
88      \l_tmpa_str
89      { TEXMF_OUTPUT_DIRECTORY }
90      \str_put_left:NV
91      \l_tmpa_str
92      \c_percent_str
93      \str_put_right:NV
94      \l_tmpa_str
95      \c_percent_str
96      \str_const:NV
97      \c_luabridge_default_output_dirname_str
98      \l_tmpa_str
99    }
100   {
101     \str_const:Nn
102     \c_luabridge_default_output_dirname_str
103     { . }
104   }
105 }
106 \str_const:Nx
107 \c_luabridge_default_helper_script_filename_str
108 { \jobname.luabridge.lua }
109 \str_const:Nx
110 \c_luabridge_default_error_output_filename_str
111 { \jobname.luabridge.err }
112 \str_if_exist:NF
113 \g_luabridge_output_dirname_str
114 {
115   \str_new:N
116   \g_luabridge_output_dirname_str
117   \str_gset_eq:NN
118   \g_luabridge_output_dirname_str
119   \c_luabridge_default_output_dirname_str
120 }
121 \str_if_exist:NF
122 \g_luabridge_helper_script_filename_str
123 {
124   \str_gset_eq:NN
125   \g_luabridge_helper_script_filename_str
126   \c_luabridge_default_helper_script_filename_str
127 }
128 \str_if_exist:NF
129 \g_luabridge_error_output_filename_str
130 {

```

```

131     \str_gset_eq:NN
132     \g_luabridge_error_output_filename_str
133     \c_luabridge_default_error_output_filename_str
134 }
135 \cs_new:Nn
136 \luabridge_tl_set:Nn
137 {
138     \iow_open:NV
139     \g_tmpa_iow
140     \g_luabridge_helper_script_filename_str
141     \msg_info:nnV
142     { luabridge }
143     { writing-helper-script }
144     \g_luabridge_helper_script_filename_str

```

Escape " and \ in the Lua code, so that we can represent it as a double-quoted string that we can pass into the load() Lua built-in and fail gracefully if the Lua code fails to compile.

```

145     \tl_set:Nx
146     \l_tmpa_tl
147     { \tl_to_str:n { #2 } }
148     \regex_replace_all:nnN
149     { [\\"] }
150     { \\0 }
151     \l_tmpa_tl
152     \tl_set:Nx
153     \l_tmpa_tl
154     {
155         local~ran_ok, err = pcall(function()
156             local~ran_ok, kpse = pcall(require,~"kpse")
157             if~ran_ok~then~kpse.set_program_name("luatex") end~
158             assert(load(" \exp_not:V \l_tmpa_tl "))( )
159         end)
160         if~not~ran_ok~then~
161             local~file = io.open("
162                 \g_luabridge_output_dirname_str /
163                 \g_luabridge_error_output_filename_str
164                 ", "w")
165             if~file~then~
166                 file:write(err .. " \iow_char:N \\ n ")
167                 file:close()
168             end~
169             print('
170                 \iow_char:N \\ \iow_char:N \\ begingroup
171                 \iow_char:N \\ \iow_char:N \\ ExplSyntaxOn
172                 \iow_char:N \\ \iow_char:N \\ csname~
173                 msg_error:nnv\iow_char:N \\ \iow_char:N \\ endcsname
174                 { luabridge }
175                 { failed-to-execute }
176                 { g_luabridge_output_dirname_str }
177                 { g_luabridge_error_output_filename_str }
178                 \iow_char:N \\ \iow_char:N \\ endgroup
179                 ')
180             end

```

```

181     }
182     \iow_now:NV
183     \g_tmpa_iow
184     \l_tmpa_tl
185     \iow_close:N
186     \g_tmpa_iow
187     \msg_info:nnV
188     { luabridge }
189     { executing-helper-script }
190     \g_luabridge_helper_script_filename_str
191     \sys_get_shell:xnNTF
192     {

```

If the environmental variable `TEXMF_OUTPUT_DIRECTORY` is undefined, use the current working directory (`.`) instead.

```

193     \str_if_eq:NNTF
194     \g_luabridge_output_dirname_str
195     \c_luabridge_default_output_dirname_str
196     {
197     \sys_if_platform_windows:TF
198     {
199         if~not~defined~TEXMF_OUTPUT_DIRECTORY~(
200             texlua~
201             \g_luabridge_helper_script_filename_str
202         )~else~(
203             texlua~
204             \g_luabridge_output_dirname_str /
205             \g_luabridge_helper_script_filename_str
206         )
207     }
208     {
209     \sys_if_platform_unix:T
210     {
211         TEXMF_OUTPUT_DIRECTORY =
212         ${TEXMF_OUTPUT_DIRECTORY:-.} \iow_newline:
213     }
214     texlua~
215     \g_luabridge_output_dirname_str /
216     \g_luabridge_helper_script_filename_str
217     }
218     }
219     {
220     texlua~
221     \g_luabridge_output_dirname_str /
222     \g_luabridge_helper_script_filename_str
223     }
224     }
225     { }
226     #1
227     {
228     }
229     {
230     \msg_error:nn
231     { luabridge }

```

```

232         { level-disabled }
233     }
234 }
235 \prg_generate_conditional_variant:Nnn
236   \sys_get_shell:nnN
237   { xnN }
238   { TF }
239 \cs_generate_variant:Nn
240   \msg_info:nnn
241   { nnV }
242 \cs_generate_variant:Nn
243   \msg_error:nnnn
244   { nnvv }
245 \cs_generate_variant:Nn
246   \iow_open:Nn
247   { NV }
248 \cs_generate_variant:Nn
249   \iow_now:Nn
250   { NV }
251 \msg_new:nnn
252   { luabridge }
253   { writing-helper-script }
254   {
255     Writing~a~helper~Lua~script~to~file~#1
256   }
257 \msg_new:nnn
258   { luabridge }
259   { executing-helper-script }
260   {
261     Executing~a~helper~Lua~script~from~file~#1
262   }
263 \msg_new:nnnn
264   { luabridge }
265   { failed-to-execute }
266   {
267     An~error~was~encountered~while~executing~Lua~code
268   }
269   {
270     For~further~clues,~examine~file~#1 / #2
271   }
272 \msg_new:nnnn
273   { luabridge }
274   { level-disabled }
275   {
276     Shell~escape~seems~to~be~disabled
277   }
278   {
279     You~may~need~to~run~TeX~with~the~---shell-escape~or~the~
280     --enable-write18~flag,~or~write~shell_escape=t~in~the~
281     texmf.cnf~file.
282   }
283 }
284 \int_compare:nNnT
285   { \g_luabridge_method_int }

```



```

286 =
287 { \c_luabridge_method_directlua_int }
288 {
289   \cs_new:Nn
290     \luabridge_tl_set:Nn
291     {
292       \tl_set:Nn
293         \l_tmpa_tl
294         { #2 }
295       \tl_set:Nx
296         \l_tmpa_tl
297         {
298           _ENV = setmetatable({}, {__index = _ENV})
299           local~function~print(input)
300             input = tostring(input)
301             local~output = {}
302             for~line~in~input:gmatch("[^
303               \iow_char:N \\ r
304               \iow_char:N \\ n
305             ]+") do~
306               table.insert(output, line)
307             end~
308             tex.print(output)
309           end~
310           \exp_not:V \l_tmpa_tl
311         }
312       \tl_set:Nf
313         #1
314         {
315           \lua_now:V
316           \l_tmpa_tl
317         }
318     }
319   \cs_generate_variant:Nn
320     \lua_now:n
321     { V }
322 }
323 \cs_new:Nn
324   \luabridge_now:n
325   {
326     \luabridge_tl_set:Nn
327       \l_tmpb_tl
328       { #1 }
329     \tl_use:N
330       \l_tmpb_tl
331   }
332 \cs_new_protected:Npn
333   \luabridgeExecute
334   #1
335   {
336     \luabridge_now:e
337     { #1 }
338   }
339 \cs_generate_variant:Nn

```

```

340 \luabridge_now:n
341 { e }
342 \ExplSyntaxOff
343 </generic-package>

```

6 L^AT_EX implementation

This section contains the implementation for L^AT_EX.

```

344 <*latex-package>
345 \RequirePackage{expl3}
346 \ProvidesExplPackage
347   {lt3luabridge}%
348   {2024-07-03}%
349   {2.2.0}%
350   {An expl3 package that allows you to execute Lua code in LuaTeX or any other
351     TeX engine that exposes the shell}
352 \input lt3luabridge\relax
353 </latex-package>

```

7 ConT_EXt implementation

This section contains the implementation for ConT_EXt. ConT_EXt MkII, MkIV, and later formats are supported.

```

354 <*context-package>
355 \writestatus{loading}{ConTeXt User Module / lt3luabridge}
356 \startmodule[lt3luabridge]
357 \unprotect
358 \input lt3luabridge\relax
359 </context-package>

```

References

- [1] Vít Novotný. *Markdown. A package for converting and rendering markdown documents inside T_EX*. Version 2.15.2-0-gb238dbc. May 31, 2022. URL: <https://ctan.org/pkg/markdown> (visited on 06/26/2022).
- [2] The L^AT_EX Team. *expl3. Wrapper package for experimental L^AT_EX3*. June 16, 2022. URL: <https://ctan.org/pkg/expl3> (visited on 06/26/2022).

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

Symbols	Numbers
\ \ 149 , 150 , 166 , 170 , 171 , 172 , 173 , 178 , 303 , 304	\ 0 150

C	
cs commands:	
\cs_generate_variant:Nn	
..... 65, 239, 242, 245, 248, 319, 339	
\cs_new:Nn	135, 289, 323
\cs_new_protected:Npn	332
D	
\directlua	2
E	
exp commands:	
\exp_not:n	158, 310
\ExplSyntaxOff	342
\ExplSyntaxOn	3, 6
F	
\fi	5
I	
\ifx	3
\input	4, 352, 358
int commands:	
\int_case:nnTF	48
\int_compare:nNnTF	73, 284
\int_const:Nn	7, 10
\int_gset_eq:NN	20, 25
\int_if_exist:NTF	13
\int_new:N	16
iow commands:	
\iow_char:N	166, 170, 171, 172, 173, 178, 303, 304
\iow_close:N	185
\iow_newline:	212
\iow_now:Nn	182, 249
\iow_open:Nn	138, 246
\g_tmpa_iow	139, 183, 186
J	
\jobname	108, 111
L	
lua commands:	
\lua_now:n	1, 2, 315, 320
luabridge commands:	
\c_luabridge_default_error_- output_filename_str ..	3, 110, 133
\c_luabridge_default_helper_- script_filename_str ..	3, 107, 126
\c_luabridge_default_output_- dirname_str ..	2, 81, 97, 102, 119, 195
\g_luabridge_error_output_- filename_str	3, 129, 132, 163
\g_luabridge_helper_script_- filename_str	3, 122, 125, 140, 144, 190, 201, 205, 216, 222
\c_luabridge_method_directlua_- int	2, 8, 22, 57, 287
\g_luabridge_method_int 2, 14, 17, 21, 26, 49, 71, 74, 285
\c_luabridge_method_shell_int	2, 11, 27, 51, 76
\luabridge_now:n ..	1, 2, 324, 336, 340
\g_luabridge_output_dirname_str	2, 113, 116, 118, 162, 194, 204, 215, 221
\luabridge_tl_set:Nn ..	2, 136, 290, 326
\luabridgeExecute	2, 333
M	
msg commands:	
\msg_error:nn	230
\msg_error:nnn	66, 68
\msg_error:nnnn	243
\msg_info:nn	53, 59
\msg_info:nnn	141, 187, 240
\msg_new:nnn	30, 36, 42, 251, 257
\msg_new:nnnn	263, 272
P	
prg commands:	
\prg_generate_conditional_- variant:Nnn	235
\ProvidesExplPackage	346
R	
regex commands:	
\regex_replace_all:nnN	148
\relax	4, 352, 358
\RequirePackage	345
S	
\startmodule	356
str commands:	
\c_percent_str	92, 95
\str_const:Nn ...	80, 96, 101, 106, 109
\str_gset_eq:NN	117, 124, 131
\str_if_eq:NNTF	193
\str_if_exist:NTF	112, 121, 128
\str_new:N	115
\str_put_left:Nn	90
\str_put_right:Nn	93
\str_set:Nn	87
\l_tmpa_str	88, 91, 94, 98
sys commands:	
\sys_get_shell:nnN	236
\sys_get_shell:nnNTF	191
\sys_if_engine luatex:TF	18

\sys_if_platform_unix:TF	78, 209	\l_tmpb_tl	327, 330
\sys_if_platform_windows:TF . .	85, 197		
		T	U
tl commands:		\undefined	3
\tl_set:Nn	145, 152, 292, 295, 312	\unprotect	357
\tl_to_str:n	147		
\tl_use:N	329	W	
\l_tmpa_tl	146,	\write18	2
151, 153, 158, 184, 293, 296, 310, 316		\writestatus	355