# The luacolor package

# Heiko Oberdiek\*

### 2023-08-18 v1.18

#### Abstract

Package luacolor implements color support based on LuaTeX's node attributes.

# Contents

	שטע	cumentation	2													
	1.1	Introduction	2													
	1.2	Usage	2													
	1.3	Limitations	2													
2	Imp	plementation	3													
	2.1	Catcodes and identification	3													
	2.2	Check for LuaT <sub>E</sub> X	3													
	2.3	Check for disabled colors	4													
	2.4	Load module and check version	4													
	2.5	Find driver	4													
	2.6	Attribute setting	5													
	2.7	Whatsit insertion	5													
	2.8	\pdfxform/\saveboxresource support	6													
	2.9	Lua module	7													
		2.9.1 Driver detection	7													
		2.9.2 Color strings	8													
		2.9.3 Attribute register	9													
		2.9.4 Whatsit insertion	9													
3	Inst	Installation 1														
	3.1	Download	11													
		Bundle installation	4.4													
	3.2	Buildie installation	11													
	$\frac{3.2}{3.3}$	Package installation	12													
	-	Package installation														
	3.3	Package installation	12													
4	3.3 3.4 3.5	Package installation	12 12													
4	3.3 3.4 3.5 His	Package installation	12 12 12													
4	3.3 3.4 3.5 <b>His</b> [200	Package installation Refresh file name databases Some details for the interested  tory  [7/12/12 v1.0]	12 12 12 12													
4	3.3 3.4 3.5 <b>His</b> [200 [200	Package installation Refresh file name databases Some details for the interested  tory [7/12/12 v1.0]	12 12 12 12 12													
4	3.3 3.4 3.5 <b>His</b> [200 [200	Package installation Refresh file name databases Some details for the interested  tory  7/12/12 v1.0	12 12 12 12 12 12													
4	3.3 3.4 3.5 <b>His</b> [200 [201 [201	Package installation Refresh file name databases Some details for the interested  tory [7/12/12 v1.0]	12 12 12 12 12 12 12 13													
4	3.3 3.4 3.5 <b>His</b> [200 [200 [201 [201 [201	Package installation Refresh file name databases Some details for the interested  tory [7/12/12 v1.0]	12 12 12 12 12 12 12 13 13													
4	3.3 3.4 3.5 <b>His</b> [200 [201 [201 [201 [201	Package installation Refresh file name databases Some details for the interested  tory  17/12/12 v1.0]	12 12 12 12 12 12 12 13 13 13													
4	3.3 3.4 3.5 <b>His</b> [200 [201 [201 [201 [201 [201	Package installation Refresh file name databases Some details for the interested  tory [7/12/12 v1.0] [9/04/10 v1.1] [0/03/09 v1.2] [0/12/13 v1.3] [1/03/29 v1.4] [1/04/22 v1.5] [1/04/23 v1.6]	12 12 12 12 12 12 13 13 13													
4	3.3 3.4 3.5 <b>His</b> [200 [200 [201 [201 [201 [201 [201	Package installation Refresh file name databases Some details for the interested  tory  17/12/12 v1.0]	12 12 12 12 12 12 13 13 13 13													
4	3.3 3.4 3.5 His: [200 [201 [201 [201 [201 [201 [201 [201	Package installation Refresh file name databases Some details for the interested  tory  17/12/12 v1.0] 19/04/10 v1.1] 10/03/09 v1.2] 10/12/13 v1.3] 1/03/29 v1.4] 1/04/22 v1.5] 1/04/23 v1.6] 1/10/22 v1.7]	12 12 12 12 12 12 13 13 13 13 13													

 $<sup>{\</sup>rm *Please\ report\ any\ issues\ at\ https://github.com/ho-tex/luacolor/issues}$ 

[2016/05]	/16 v	$^{1.10}$																														13
[2018/11]	/22 v	[1.11]																														14
[2019/07]	/25  v	[1.12]																														14
[2019/11]	/29 v	[1.13]																														14
[2020-02-	22 v	1.14]																														14
[2020-02-	24 v	1.15]																														14
[2020-04-	04 v	1.16]																														14
[2021-02-	17 v	1.17]																														14
[2023-08-	18 v	1.18]																														14
	[2018/11 [2019/07 [2019/11 [2020-02- [2020-04- [2021-02-	[2018/11/22 v [2019/07/25 v [2019/11/29 v [2020-02-22 v [2020-02-24 v [2020-04-04 v [2021-02-17 v]	[2018/11/22 v1.11] [2019/07/25 v1.12] [2019/11/29 v1.13] [2020-02-22 v1.14] [2020-02-24 v1.15] [2020-04-04 v1.16] [2021-02-17 v1.17]	[2018/11/22 v1.11] . [2019/07/25 v1.12] . [2019/11/29 v1.13] . [2020-02-22 v1.14] . [2020-02-24 v1.15] . [2020-04-04 v1.16] . [2021-02-17 v1.17]	[2018/11/22 v1.11] [2019/07/25 v1.12] [2019/11/29 v1.13] [2020-02-22 v1.14] [2020-02-24 v1.15] [2020-04-04 v1.16] [2021-02-17 v1.17]	[2018/11/22 v1.11] [2019/07/25 v1.12] [2019/11/29 v1.13] [2020-02-22 v1.14] [2020-02-24 v1.15] [2020-04-04 v1.16] [2021-02-17 v1.17]	[2018/11/22 v1.11] [2019/07/25 v1.12] [2019/11/29 v1.13] [2020-02-22 v1.14] [2020-02-24 v1.15] [2020-04-04 v1.16] [2021-02-17 v1.17]	[2018/11/22 v1.11] [2019/07/25 v1.12] [2019/11/29 v1.13] [2020-02-22 v1.14] [2020-02-24 v1.15] [2020-04-04 v1.16] [2021-02-17 v1.17]	[2018/11/22 v1.11] [2019/07/25 v1.12] [2019/11/29 v1.13] [2020-02-22 v1.14] [2020-02-24 v1.15] [2020-04-04 v1.16] [2021-02-17 v1.17]	[2018/11/22 v1.11] [2019/07/25 v1.12] [2019/11/29 v1.13] [2020-02-22 v1.14] [2020-02-24 v1.15] [2020-04-04 v1.16] [2021-02-17 v1.17]	[2018/11/22 v1.11]	2016/05/16 v1.10																				

14

# 1 Documentation

#### 1.1 Introduction

This package uses a LuaTEX's attribute register to to annotate nodes with color information. If a color is set, then the attribute register is set to this color and all nodes created in its scope (current group) are annotated with this attribute. Now the color property behaves much the same way as the font property.

### 1.2 Usage

5 Index

Package color is loaded automatically by this package luacolor. If you need a special driver option or you prefer package xcolor, then load it before package luacolor, for example:

```
\usepackage[dvipdfmx]{xcolor}
```

The package luacolor is loaded without options:

```
\usepackage{luacolor}
```

It is able to detect PDF mode and DVI drivers are differentiated by its color specials. Therefore the package do need driver options.

Then it redefines the color setting commands to set attributes instead of whatsits for color.

At last the attribute annotations of the nodes in the output box must be analyzed to insert the necessary color whatsits. (With older LuaTEX that lack the appropriate callback function the package atbegshi is used to get control before a box is shipped out.)

```
	extstyle 	ext
```

Macro \luacolorProcessBox processes the box  $\langle box \rangle$  in the previously described manner. It is automatically called for pages, but not for XForm objects. Before passing a box to \pdfxform, call \luacolorProcessBox first.

### 1.3 Limitations

Ligatures with different colored components: Package luacolor sees the ligature after the paragraph building and page breaking, when a page is to be shipped out. Therefore it cannot break ligatures, because the components might occupy different space. Therefore it is the responsibility of the ligature forming process to deal with different colored glyphs that form a ligature. The user can avoid the problem entirely by explicitly breaking the ligature at the places where the color changes.

. .

# 2 Implementation

1 (\*package)

### 2.1 Catcodes and identification

```
2 \begingroup\catcode61\catcode48\catcode32=10\relax%
   \catcode13=5 % ^^M
    \endlinechar=13 %
    \catcode123=1 % {
    \catcode125=2 % }
    \catcode64=11 % @
    \def\x{\endgroup
8
      \expandafter\edef\csname LuaCol@AtEnd\endcsname{%
9
        \endlinechar=\the\endlinechar\relax
10
11
        \catcode13=\the\catcode13\relax
12
        \catcode32=\the\catcode32\relax
13
        \catcode35=\the\catcode35\relax
14
        \catcode61=\the\catcode61\relax
15
        \catcode64=\the\catcode64\relax
16
        \catcode123=\the\catcode123\relax
17
         \catcode125=\the\catcode125\relax
18
      }%
19
    }%
20 \x\catcode61\catcode48\catcode32=10\relax%
21 \catcode13=5 % ^^M
22 \endlinechar=13 %
23 \catcode35=6 % #
24 \catcode64=11 % @
25 \catcode123=1 \% \{
26 \catcode125=2 % }
27 \def\TMP@EnsureCode#1#2{%
    \edef\LuaCol@AtEnd{%
28
      \LuaCol@AtEnd
29
      \catcode#1=\the\catcode#1\relax
30
31
    }%
    \catcode#1=#2\relax
32
33 }
34 \TMP@EnsureCode{34}{12}% "
35 \TMP@EnsureCode{39}{12}% '
36 \TMP@EnsureCode{40}{12}% (
37 \TMP@EnsureCode{41}{12}% )
38 \TMP@EnsureCode{42}{12}% *
39 \TMP@EnsureCode{43}{12}% +
40 \TMP@EnsureCode{44}{12}% ,
41 \TMP@EnsureCode{45}{12}% -
42 \TMP@EnsureCode{46}{12}% .
43 \TMP@EnsureCode{47}{12}% /
44 \TMP@EnsureCode{58}{12}%:
45 \TMP@EnsureCode{60}{12}% <
46 \TMP@EnsureCode{62}{12}% >
47 \TMP@EnsureCode{91}{12}% [
48 \TMP@EnsureCode{93}{12}% ]
49 \TMP@EnsureCode\{95\}\{12\}\% _ (other!)
50 \TMP@EnsureCode{96}{12}\% '
51 \edef\LuaCol@AtEnd{\LuaCol@AtEnd\noexpand\endinput}
  Package identification.
52 \NeedsTeXFormat{LaTeX2e}
53 \ProvidesPackage{luacolor}%
    [2023-08-18 v1.18 Color support via LuaTeX's attributes (HO)]
```

### 2.2 Check for LuaT<sub>E</sub>X

Without LuaTeX there is no point in using this package.

```
55 \RequirePackage{color}
56 \ifx\directlua\@undefined
57 \PackageError{luacolor}{%
58 This package may only be run using LuaTeX%
59 }\@ehc
60 \expandafter\LuaCol@AtEnd
61 \fi%
```

### 2.3 Check for disabled colors

```
62 \ifcolors@
63 \else
    \PackageWarningNoLine{luacolor}{%
64
65
      Colors are disabled by option 'monochrome',
66
    \def\set@color{}%
67
   \def\reset@color{}%
68
   \def\set@page@color{}%
70
   \def\define@color#1#2{}%
71
   \expandafter\LuaCol@AtEnd
72 \fi%
```

### 2.4 Load module and check version

```
73 \directlua{%
74 require("luacolor")%
75 }
76 \begingroup
    \edef\x{\directlua{tex.write("2023-08-18 v1.18")}}%
78
    \left\{ \right\}
79
      \directlua{%
80
         if oberdiek.luacolor.getversion then %
           oberdiek.luacolor.getversion()%
81
         end%
82
      }%
83
    }%
84
85
    \int ifx\x\y
    \else
86
       \PackageError{luacolor}{%
87
         Wrong version of lua module.\MessageBreak
88
89
         Package version: \x\MessageBreak
90
         Lua module: \y
      }\@ehc
91
    \fi
92
93 \endgroup
```

### 2.5 Find driver

```
94 \ifnum\outputmode=\@ne
95 \else
96
     \begingroup
       \def\current@color{}%
97
       \def\reset@color{}%
98
 99
       \setbox\z@=\hbox{%
100
          \begingroup
101
            \set@color
102
          \endgroup
       }%
103
       \edef\reserved@a{%
104
         \directlua{%
105
           oberdiek.luacolor.dvidetect()%
106
107
         }%
       }%
108
```

```
109
                           \ifx\reserved@a\@empty
                             \PackageError{luacolor}{%
                    110
                               DVI driver detection failed because of \MessageBreak
                    111
                    112
                               unrecognized color \string\special
                    113
                             }\@ehc
                    114
                             \endgroup
                    115
                             \expandafter\expandafter\LuaCol@AtEnd
                    116
                           \else
                             \PackageInfo{luacolor}{%
                    117
                               Type of color \string\special: \reserved@a
                    118
                             \@gobble}%
                    119
                           \fi%
                    120
                    121
                         \endgroup
                    122 \fi
                    2.6
                          Attribute setting
  \LuaCol@Attribute
                    123 \newattribute\LuaCol@Attribute
                    124 \let\LuaCol@setattribute\setattribute
                    125 \directlua{%
                    oberdiek.luacolor.setattribute(\number\allocationnumber)%
                    127 }
         \set@color change 2023-08-18: added \reset@color so that \mathcolor can gobble it, issue
                    7.
                    128 \protected\def\set@color{%
                        \LuaCol@setattribute\LuaCol@Attribute{%
                    129
                           \directlua{%
                    130
                             oberdiek.luacolor.get("\luaescapestring{\current@color}")%
                    131
                    132
                           }%
                    133 }%
                    134 \aftergroup\reset@color
                    135 }
       \reset@color
                    136 \def\reset@color{}
                    2.7 Whatsit insertion
\luacolorProcessBox
                    137 \def\luacolorProcessBox#1{%
                        \directlua{%
                    139
                           oberdiek.luacolor.process(\number#1)%
                        }%
                    140
                    141 }
                    142 \directlua{%
                    143 if luatexbase.callbacktypes.pre_shipout_filter then
                    144
                           token.get_next()
                    145
                        end
                    146 }\@secondoftwo\@gobble{
                        \RequirePackage{atbegshi}[2011/01/30]
                    148 \AtBeginShipout{%
                           \luacolorProcessBox\AtBeginShipoutBox
                    149
                    150
                        }
                    151 }
```

Set default color.

152 \set@color

### 2.8 \pdfxform/\saveboxresource support

```
153 \ifnum\outputmode=\@ne
       \let\LuaCol@org@pdfxform\saveboxresource
First we need some helpers to allow expandable code to parse keyword style ar-
guments:
       \def\LuaCol@iii@i@ii#1#2#3{#3{#1}{#2}}
155
       \def\LuaCol@ii@i#1#2{{#2#1}}
156
       \def\LuaCol@if@keyword#1#2#3{%
157
         \expanded{\unexpanded{\LuaCol@iii@i@ii{#2}{#3}}\expandafter}%
158
         \directlua{%
159
160
           token.put_next(token.create(token.scan_keyword(token.scan_string())
161
           and '@firstoftwo'
           or '@secondoftwo'))
163
         }{#1}%
       }
164
The following macro scans a integer and expands to a token equivalent to a chardef
whose value corresponds to the scanned integer. This allows the integer to be
passed around as a undelimited argument.
       \def\LuaCol@scan@number{%
165
166
         \directlua{
167
           token.put_next(token.new(token.scan_int(), token.command_id'char_given'))
168
         }%
       }
169
TeX primitives like \saveboxresource read braced arguments in a special way.
Especially they expand everything until they find a left brace. To simulate this,
we use Lua to expand everything else:
170
       \def\LuaCol@scan@tobrace{%
171
         \directlua{
           local relax, space = token.command_id'relax', token.command_id'spacer'
172
173
           local t
174
           repeat
             t = token.scan_token()
175
           until not (t.command == relax or t.command == space)
176
177
           token.put_next(t)
178
         }%
       }
179
       \def\LuaCol@scan@boxresource@i#1#2{%
180
         \LuaCol@if@keyword{attr}{%
181
           \expanded{\unexpanded{\LuaCol@scan@boxresource@iI{#1#2attr}}%
182
              \expandafter\expandafter\expandafter}%
183
           \LuaCol@scan@tobrace
184
         }{%
185
           \LuaCol@scan@boxresource@ii{#1#2}%
186
187
         }%
       }
188
       \def\LuaCol@scan@boxresource@iI#1#2{\LuaCol@scan@boxresource@ii{#1{#2}}}
189
       \def\LuaCol@scan@boxresource@ii#1{%
190
191
         \LuaCol@if@keyword{resources}{%
           \expanded{\unexpanded{\LuaCol@scan@boxresource@iiI{#1resources}}%
192
             \expandafter\expandafter\expandafter}%
193
           \LuaCol@scan@tobrace
194
         }{%
195
           \LuaCol@scan@boxresource@iii{#1}%
196
197
         }%
198
       \def\LuaCol@scan@boxresource@iiI#1#2{\LuaCol@scan@boxresource@iii{#1{#2}}}
199
       \def\LuaCol@scan@boxresource@iii#1{%
200
         \LuaCol@if@keyword{margin}{%
201
           \expanded{\unexpanded{\LuaCol@scan@boxresource@iv{#1margin }}%
202
203
             \expandafter\expandafter\expandafter}%
```

```
\LuaCol@scan@number
204
205
206
            \LuaCol@scan@boxresource@iv{#1}{}%
207
208
209
       \def\LuaCol@scan@boxresource@iv#1#2{%
210
         \expanded{\unexpanded{\LuaCol@scan@boxresource@v{#1#2}}%
211
           \expandafter\expandafter\expandafter}%
212
         \LuaCol@scan@number
       }
213
       \def\LuaCol@scan@boxresource@v#1#2{%
214
         \luacolorProcessBox{#2}%
215
216
         \LuaCol@org@pdfxform#1#2%
217
```

This could be written in Lua, but at least upto LuaTEX 1.11, feeding back too many tokens from Lua to TEX triggers a segmentation fault. This is written in Lua so the integer setting is expandable and does not interfere with a preceding \immediate.

```
219
       \protected\def\saveboxresource{%
220
         \LuaCol@if@keyword{type}{%
221
            \expandafter
            \expanded{\unexpanded{\LuaCol@scan@boxresource@i{type }}%
222
              \expandafter\expandafter\expandafter}%
223
            \LuaCol@scan@number
224
         }{%
225
            \LuaCol@scan@boxresource@i{}{}%
226
227
         }%
228
       }
Legacy alias.
229
       \let\pdfxform\saveboxresource
230 \fi
231 \LuaCol@AtEnd%
232 (/package)
```

### 2.9 Lua module

```
233 (*lua)
```

Box zero contains a hbox with the color hat is analyzed to get the prefix for the color setting happened.

```
234 oberdiek = oberdiek or {}
235 local luacolor = oberdiek.luacolor or {}
236 oberdiek.luacolor = luacolor
```

#### getversion()

```
237 function luacolor.getversion()
238 tex.write("2023-08-18 v1.18")
239 end
```

### 2.9.1 Driver detection

```
240 local ifpdf = tonumber(tex.outputmode or tex.pdfoutput) > 0
241 \; {\sf local \; prefix}
242 local prefixes = {
            = "color ",
243 dvips
    dvipdfm = "pdf:sc ",
244
     truetex = "textcolor:",
245
    pctexps = "ps::",
246
247 }
248 local patterns = {
    ["^color "]
                              = "dvips",
249
     ["^pdf: *begincolor "] = "dvipdfm",
250
```

```
["^pdf: *bcolor "]
                                         = "dvipdfm",
            251
                 ["^pdf: *bc "]
                                          = "dvipdfm",
            252
                 ["^pdf: *setcolor "] = "dvipdfm",

["^pdf: *scolor "] = "dvipdfm",

["^pdf: *sc "] = "dvipdfm",
            253
            254
            255
            256 ["^textcolor:"]
                                         = "truetex",
            257 ["^ps::"]
                                          = "pctexps",
            258 }
     info()
            259 local function info(msg, term)
            260 local target = "log"
            261
                 if term then
                 target = "term and log"
            262
            263
            264 texio.write_nl(target, "Package luacolor info: " .. msg .. ".")
            265 texio.write_nl(target, "")
            266 \; \mathrm{end}
dvidetect()
            267 function luacolor.dvidetect()
            268 local v = tex.box[0]
                 assert(v.id == node.id("hlist"))
            269
                 for v in node.traverse_id(node.id("whatsit"), v.head) do
            271
                   if v and v.subtype == node.subtype("special") then
            272
                      local data = v.data
                      for pattern, driver in pairs(patterns) do
            273
            274
                        if string.find(data, pattern) then
                          prefix = prefixes[driver]
            275
                          tex.write(driver)
            276
            277
                          return
            278
                       end
            279
                      end
                      info("\\special{" .. data .. "}", true)
            280
            281
                      return
            282
                    end
            283 end
            284 info("Missing \\special", true)
            285 end
            2.9.2 Color strings
            286 \log 1 map = {
            287 \quad n = 0,
            288 }
      get()
            289 function luacolor.get(color)
            290 tex.write("" .. luacolor.getvalue(color))
            291 end
 getvalue()
            292 function luacolor.getvalue(color)
            293 local n = map[color]
            294 if not n then
            295
                 n = map.n + 1
            296
                 map.n = n
            297
                 map[n] = color
                 map[color] = n
            298
            299 end
            300 return n
            301 end
```

#### 2.9.3 Attribute register

```
setattribute()
               302 local attribute
               303 function luacolor.setattribute(attr)
               304 attribute = attr
               305 end
getattribute()
               306 function luacolor.getattribute()
               307 return attribute
               308 end
               2.9.4 Whatsit insertion
               309 local LIST = 1
               310 local LIST_LEADERS = 2
               311 local LIST_DISC = 3
               312 local COLOR = 4
               313 local NOCOLOR = 5
               314 local RULE = node.id("rule")
               315 local node_types = {
               316 [node.id("hlist")] = LIST,
               317 [node.id("vlist")] = LIST,
               318 [node.id("rule")] = COLOR,
               319 [node.id("glyph")] = COLOR,
               320
                   [node.id("disc")] = LIST_DISC,
               321
                    [node.id("whatsit")] = {
               322
                      [node.subtype("pdf_colorstack")] =
               323
                        function(n)
                          return n.stack == 0 and NOCOLOR or nil
               324
               325
                        end,
                      [node.subtype("special")] = COLOR,
               326
               327
                      [node.subtype("pdf_literal")] = COLOR,
               328
                      [node.subtype("pdf_save")] = COLOR,
                      [node.subtype("pdf_restore")] = COLOR, -- probably not needed
               330 -- TODO (DPC)
                                   [node.subtype("pdf_refximage")] = COLOR,
               331 },
                   [node.id("glue")] =
               332
               333
                     function(n)
                        if n.subtype >= 100 then -- leaders
               334
                         if n.leader.id == RULE then
               335
                            return COLOR
               336
                          else
               337
                           return LIST_LEADERS
               338
               339
                          end
               340
                        end
               341
                      end,
               342 }
   get_type()
               343 local function get_type(n)
               344 local ret = node_types[n.id]
               345 if type(ret) == 'table' then
               346
                   ret = ret[n.subtype]
               347 end
               348 if type(ret) == 'function' then
               349
                    ret = ret(n)
               350
                   end
                   return ret
               351
               352 \; \text{end}
               353 local mode = 2 -- luatex.pdfliteral.direct
```

```
354 local WHATSIT = node.id("whatsit")
                        355 local SPECIAL = node.subtype("special")
                        356 local PDFLITERAL = node.subtype("pdf_literal")
                        357 local DRY_FALSE = false
                        358 local DRY_TRUE = true
traverse()
                        359 local function traverse(list, color, dry)
                        360 if not list then
                        361
                                      return color
                        362
                        363
                                  local head
                        364
                                  if get_type(list) == LIST then
                        365
                                     head = list.head
                                 elseif get_type(list) == LIST_DISC then
                        366
                                     head = list.replace
                        367
                        368
                                  else
                                       texio.write_nl("!!! Error: Wrong list type: " .. node.type(list.id))
                        369
                        370
                                       return color
                        371
                                  end
                        372 (debug)texio.write_nl("traverse: " .. node.type(list.id))
                                   for n in node.traverse(head) do
                        374 (debug)texio.write_nl(" node: " .. node.type(n.id))
                                       local t = get_type(n)
                        375
                        376 \langle debug \rangle texio.write\_nl("TYPE "..tostring(t).. " "..tostring(node.type(node.getid(n))).." ".. tostring(node.type(node.getid(n))).." ".. tostring(node.type(node.getid(n))).." ".. tostring(node.type(node.getid(n))).." ".. tostring(node.type(node.getid(n))).." ".. tostring(node.type(node.getid(n)))..." ".. tostring(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.type(node.
                        377
                                       if t == LIST or t == LIST_DISC then
                        378
                                            color = traverse(n, color, dry)
                                       elseif t == LIST_LEADERS then
                        379
                                            local color_after = traverse(n.leader, color, DRY_TRUE)
                        380
                                            if color == color_after then
                        381
                        382
                                                traverse(n.leader, color, DRY_FALSE or dry)
                        383
                                            else
                                                traverse(n.leader, '', DRY_FALSE or dry)
                        The color status is unknown here, because the leader box will or will not be set.
                        385
                                                color = ''
                        386
                                            end
                        387
                                       elseif t == COLOR then
                                            local v = node.has_attribute(n, attribute)
                        389
                                            if v then
                        390
                                                local newColor = map[v]
                                                if newColor ~= color then
                        391
                                                     color = newColor
                        392
                                                    if dry == DRY_FALSE then
                        393
                                                         local newNode
                        394
                                                         if ifpdf then
                        395
                                                             newNode = node.new(WHATSIT, PDFLITERAL)
                        396
                        397
                                                             newNode.mode = mode
                                                             newNode.data = color
                        398
                        399
                                                             newNode = node.new(WHATSIT, SPECIAL)
                        400
                        401
                                                             newNode.data = prefix .. color
                        402
                                                         end
                        403
                                                         head = node.insert_before(head, n, newNode)
                        404
                                                     end
                        405
                                                end
                        406
                                            end
                                        elseif t == NOCOLOR then
                        407
                                            color = ''
                        408
                        409
                        410
                                   if get_type(list) == LIST then
                        411
```

412

list.head = head

```
else
           413
           414
                  list.replace = head
           415
               end
               return color
           416
           417 \; \mathrm{end}
process()
           418 function luacolor.process(box)
           419 local color = ""
                local list = tex.getbox(box)
           421
               traverse(list, color, DRY_FALSE)
           422 end
           423
           424 \ \text{if luatexbase.callbacktypes.pre\_shipout\_filter then}
           425 luatexbase.add_to_callback("pre_shipout_filter", function(list)
                  traverse(list, "", DRY_FALSE)
           426
                  return true
           427
           428 end, "luacolor.process")
           429 end
```

For recent versions of luaotfload, we can register a callback to control how coloring glyph is handled for the color feature.

```
430 if luaotfload.set_colorhandler then
431 local set_attribute = node.direct.set_attribute
432 luaotfload.set_colorhandler(function(head, n, color)
433 set_attribute(n, attribute, luacolor.getvalue(color))
434 return head, n
435 end)
436 end
437 ⟨/lua⟩
```

# 3 Installation

#### 3.1 Download

**Package.** This package is available on CTAN<sup>1</sup>:

CTAN:macros/latex/contrib/luacolor/luacolor.dtx The source file.

CTAN:macros/latex/contrib/luacolor/luacolor.pdf Documentation.

**Bundle.** All the packages of the bundle 'luacolor' are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

```
CTAN:install/macros/latex/contrib/luacolor.tds.zip
```

TDS refers to the standard "A Directory Structure for TEX Files" (CTAN:pkg/tds). Directories with texmf in their name are usually organized this way.

### 3.2 Bundle installation

Unpacking. Unpack the luacolor.tds.zip in the TDS tree (also known as texmf tree) of your choice. Example (linux):

```
unzip luacolor.tds.zip -d ~/texmf
```

Script installation. Check the directory TDS:scripts/luacolor/ for scripts that need further installation steps.

<sup>1</sup>CTAN:pkg/luacolor

### 3.3 Package installation

**Unpacking.** The .dtx file is a self-extracting docstrip archive. The files are extracted by running the .dtx through plain  $T_FX$ :

```
tex luacolor.dtx
```

**TDS.** Now the different files must be moved into the different directories in your installation TDS tree (also known as texmf tree):

```
\label{luacolor.sty} \begin{array}{l} {\tt luacolor.sty} \to {\tt tex/latex/luacolor/luacolor.sty} \\ {\tt luacolor.lua} \to {\tt scripts/luacolor/luacolor.lua} \\ {\tt luacolor.pdf} \to {\tt doc/latex/luacolor/luacolor.pdf} \\ {\tt luacolor.dtx} \to {\tt source/latex/luacolor/luacolor.dtx} \end{array}
```

If you have a docstrip.cfg that configures and enables docstrip's TDS installing feature, then some files can already be in the right place, see the documentation of docstrip.

#### 3.4 Refresh file name databases

If your TEX distribution (TEX Live, MiKTEX, ...) relies on file name databases, you must refresh these. For example, TEX Live users run texhash or mktexlsr.

#### 3.5 Some details for the interested

Unpacking with LATEX. The .dtx chooses its action depending on the format:

plain TEX: Run docstrip and extract the files.

LATEX: Generate the documentation.

If you insist on using LATEX for docstrip (really, docstrip does not need LATEX), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{luacolor.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

Generating the documentation. You can use both the .dtx or the .drv to generate the documentation. The process can be configured by the configuration file ltxdoc.cfg. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfIAT<sub>F</sub>X:

```
pdflatex luacolor.dtx
makeindex -s gind.ist luacolor.idx
pdflatex luacolor.dtx
makeindex -s gind.ist luacolor.idx
pdflatex luacolor.dtx
```

# 4 History

```
[2007/12/12 v1.0]
```

• First public version.

### [2009/04/10 v1.1]

• Fixes for changed syntax of \directlua in LuaTeX 0.36.

# [2010/03/09 v1.2]

• Adaptation for package luatex 2010/03/09 v0.4.

# [2010/12/13 v1.3]

- Support for \pdfxform added.
- Loaded package luatexbase-attr recognized.
- Update for LuaTeX: 'list' fields renamed to 'head' in v0.65.0.

# [2011/03/29 v1.4]

• Avoid whatsit insertion if option monochrome is used (thanks Manuel Pégourié-Gonnard).

# [2011/04/22 v1.5]

- Bug fix by Manuel Pégourié-Gonnard: A typo prevented the detection of whatsits and applying color changes for \pdfliteral and \special nodes that might contain typesetting material.
- Bug fix by Manuel Pégourié-Gonnard: Now colors are also applied to leader boxes.
- Unnecessary color settings are removed for leaders boxes, if after the leader box the color has not changed. The costs are a little runtime, leader boxes are processed twice.
- Additional whatsits that are colored: pdf\_refximage.
- Workaround for bug with node.insert\_before removed for the version after LuaT<sub>E</sub>X 0.65, because bug was fixed in 0.27. (Thanks Manuel Pégourié-Gonnard.)

# [2011/04/23 v1.6]

- Bug fix for nested leader boxes.
- Bug fix for leader boxes that change color, but are not set because of missing place.
- Version check for Lua module added.

### [2011/10/22 v1.7]

• Lua functions getattribute and getvalue added to tell other external Lua functions the attribute register number for coloring.

# [2011/11/01 v1.8]

• Use of node.subtype instead of magic numbers.

# [2016/05/13 v1.9]

- More use of node.subtype instead of magic numbers.
- luatex 85 updates

### [2016/05/16 v1.10]

• Documentation updates.

### [2018/11/22 v1.11]

- handle issue 43.
- removed pre-0.65 stuff

# [2019/07/25 v1.12]

• removed uses of module function, see PR70

# [2019/11/29 v1.13]

- Documentation updates.
- Use iftex directly.

### [2020-02-22 v1.14]

- Drop use of iftex ltxcmds and infwarerr.
- Assume Itluatex preloaded into format (true since 2015).
- Patch \saveboxresource rather than \pdfxform (keep old name as alias).
- Grab the number via Lua so that a \immediate prefix still works with \saveboxresource/\pdfxform.
- Added handler for the color feature of luaotfload

### [2020-02-24 v1.15]

• Grab all possible arguments for \saveboxresource/\pdfxform

# [2020-04-04 v1.16]

• Reset color after pdf\_colorstack whatsits.

# [2021-02-17 v1.17]

• Use  $\LaTeX$   $2_{\varepsilon}$ 's new pre\_shipout\_filter callback if it's available to allow coloring background and foreground layer material

### [2023-08-18 v1.18]

• added \reset@color to \set@color for \mathcolor, issue 7.

### 5 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; plain numbers refer to the code lines where the entry is used.

${f Symbols}$	${f A}$
\@ehc 59, 91, 113	\aftergroup 134
\@empty 109	\allocationnumber 126
\@gobble 119, 146	\AtBeginShipout 148
\@ne 94, 153	\AtBeginShipoutBox 149
$\verb \@secondoftwo  146$	${f C}$
\Qundefined $56$	\catcode
\\ 280, 284	3, 5, 6, 7, 11, 12, 13, 14, 15, 16,

17, 20, 21, 23, 24, 25, 26, 30, 32	\luacolorProcessBox 2, <u>137</u> , 149, <u>215</u>
\csname 9	\luaescapestring 131
\current@color 97, 131	
_	M
D	\MessageBreak 88, 89, 111
\define@color	N
\directlua 56, 73, 77, 79, 105,	\NeedsTeXFormat
125, 130, 138, 142, 159, 166, 171 \dvidetect() 267	\newattribute
\dv1detect() <u>201</u>	\number 126, 139
E	,
\endcsname9	O
\endinput 51	\outputmode 94, 153
\endlinechar 4, 10, 22	
\expanded . 158, 182, 192, 202, 210, 222	P
• , , , , ,	\PackageError 57, 87, 110
$\mathbf{G}$	\PackageInfo 117
\get() <u>289</u>	\PackageWarningNoLine 64
\get_type() <u>343</u>	\pdfxform 229
\getattribute() <u>306</u>	\process()
\getvalue() <u>292</u>	\protected 128, 219
\getversion() <u>237</u>	\ProvidesPackage 53
**	R
Н	\RequirePackage 55, 147
\hbox 99	\reserved@a 104, 109, 118
I	\reset@color 68, 98, 134, <u>136</u>
\ifcolors@ 62	
\ifnum 94, 153	${f S}$
\ifx 56, 85, 109	\saveboxresource 154, 219, 229
\info() 259	\set@color $67, 101, \underline{128}, 152$
<u> </u>	\set@page@color 69
${f L}$	\setattribute 124
\LuaCol@AtEnd 28, 29, 51, 60, 71, 115, 231	\setattribute() <u>302</u>
\LuaCol@Attribute <u>123</u> , 129	\setbox 99
\LuaCol@if@keyword	\special 112, 118
$\dots \dots 157, 181, 191, 201, 220$	Т
\LuaCol@ii@i	
\ T C - 1 @ : : : @ : @ : :	\the 10, 11, 12, 13, 14, 15, 16, 17, 30
\LuaCol@iii@i@ii 155, 158	\the 10, 11, 12, 13, 14, 15, 16, 17, 30 \TMP@EnsureCode 27,
\LuaCol@org@pdfxform 154, 216	\TMP@EnsureCode 27,
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i	\TMP@EnsureCode
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i	\TMP@EnsureCode 27,
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i	\TMP@EnsureCode
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i	\text{TMP@EnsureCode} \tag{27}, \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i	\TMP@EnsureCode
\LuaCol@org@pdfxform	\text{TMP@EnsureCode} \tag{27,} \\ 34, 35, 36, 37, 38, 39, 40, 41,} \\ 42, 43, 44, 45, 46, 47, 48, 49, 50 \\ \text{traverse()} \tag{59} \\ \tag{U} \\ \unexpanded  158, 182, 192, 202, 210, 222 \\ \tag{22}
\LuaCol@org@pdfxform	\TMP@EnsureCode
\LuaCol@org@pdfxform	\text{TMP@EnsureCode} \tag{27,} \\ 34, 35, 36, 37, 38, 39, 40, 41,} \\ 42, 43, 44, 45, 46, 47, 48, 49, 50 \\ \text{traverse()} \tag{59} \\ \tag{U} \\ \unexpanded  158, 182, 192, 202, 210, 222 \\ \tag{22}
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i 180, 222, 226 \LuaCol@scan@boxresource@ii 182, 189 \LuaCol@scan@boxresource@ii 186, 189, 190 \LuaCol@scan@boxresource@iiI 192, 199 \LuaCol@scan@boxresource@iii 196, 199, 200 \LuaCol@scan@boxresource@iv	\text{TMP@EnsureCode} \tag{27,} \\ 34, 35, 36, 37, 38, 39, 40, 41,} \\ 42, 43, 44, 45, 46, 47, 48, 49, 50 \\ \traverse() \tag{59} \\ \tag{U} \\ \unexpanded \text{158, 182, 192, 202, 210, 222} \\ \text{X} \\ \tag{x} \tag{8, 20, 77, 85, 89}
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i 180, 222, 226 \LuaCol@scan@boxresource@ii 182, 189 \LuaCol@scan@boxresource@ii 186, 189, 190 \LuaCol@scan@boxresource@iiI 192, 199 \LuaCol@scan@boxresource@ii 196, 199, 200 \LuaCol@scan@boxresource@iv 202, 206, 209	\text{TMP@EnsureCode} \dots \dots 27, \\ 34, 35, 36, 37, 38, 39, 40, 41, \\ 42, 43, 44, 45, 46, 47, 48, 49, 50 \\ \traverse() \dots
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i 180, 222, 226 \LuaCol@scan@boxresource@ii 182, 189 \LuaCol@scan@boxresource@ii 186, 189, 190 \LuaCol@scan@boxresource@iiI 192, 199 \LuaCol@scan@boxresource@ii 196, 199, 200 \LuaCol@scan@boxresource@iv 202, 206, 209 \LuaCol@scan@boxresource@v .210, 214	\text{TMP@EnsureCode} \tag{27,} \\ 34, 35, 36, 37, 38, 39, 40, 41,} \\ 42, 43, 44, 45, 46, 47, 48, 49, 50 \\ \traverse() \tag{59} \\ \tag{U} \\ \unexpanded \text{158, 182, 192, 202, 210, 222} \\ \text{X} \\ \tag{x} \tag{8, 20, 77, 85, 89}
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i 180, 222, 226 \LuaCol@scan@boxresource@iI 182, 189 \LuaCol@scan@boxresource@iI 182, 189 \LuaCol@scan@boxresource@iI 192, 199 \LuaCol@scan@boxresource@iII 192, 199 \LuaCol@scan@boxresource@iII 192, 199 \LuaCol@scan@boxresource@iII 196, 199, 200 \LuaCol@scan@boxresource@iV 202, 206, 209 \LuaCol@scan@boxresource@v 210, 214 \LuaCol@scan@number 165, 204, 212, 224	\text{TMP@EnsureCode} \dots \dots 27, \\ 34, 35, 36, 37, 38, 39, 40, 41, \\ 42, 43, 44, 45, 46, 47, 48, 49, 50 \\ \traverse() \dots
\LuaCol@org@pdfxform 154, 216 \LuaCol@scan@boxresource@i 180, 222, 226 \LuaCol@scan@boxresource@ii 182, 189 \LuaCol@scan@boxresource@ii 186, 189, 190 \LuaCol@scan@boxresource@iiI 192, 199 \LuaCol@scan@boxresource@ii 196, 199, 200 \LuaCol@scan@boxresource@iv 202, 206, 209 \LuaCol@scan@boxresource@v .210, 214	\text{TMP@EnsureCode} \dots \dots 27, \\ 34, 35, 36, 37, 38, 39, 40, 41, \\ 42, 43, 44, 45, 46, 47, 48, 49, 50 \\ \traverse() \dots