

# The luacolor package

Heiko Oberdiek\*  
<heiko.oberdiek at googlemail.com>

2018/11/22 v1.11

## Abstract

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

## Contents

<b>1</b>	<b>Documentation</b>	<b>2</b>
1.1	Introduction . . . . .	2
1.2	Usage . . . . .	2
1.3	Limitations . . . . .	2
<b>2</b>	<b>Implementation</b>	<b>3</b>
2.1	Catcodes and identification . . . . .	3
2.2	Check for LuaTeX . . . . .	4
2.3	Check for disabled colors . . . . .	4
2.4	Load module and check version . . . . .	4
2.5	Find driver . . . . .	5
2.6	Attribute setting . . . . .	5
2.7	Whatsit insertion . . . . .	6
2.8	\pdfxform support . . . . .	6
2.9	Lua module . . . . .	6
2.9.1	Driver detection . . . . .	7
2.9.2	Color strings . . . . .	8
2.9.3	Attribute register . . . . .	8
2.9.4	Whatsit insertion . . . . .	8
<b>3</b>	<b>Test</b>	<b>10</b>
3.1	Catcode checks for loading . . . . .	10
3.2	Driver detection . . . . .	12
<b>4</b>	<b>Installation</b>	<b>12</b>
4.1	Download . . . . .	12
4.2	Bundle installation . . . . .	12
4.3	Package installation . . . . .	13
4.4	Refresh file name databases . . . . .	13
4.5	Some details for the interested . . . . .	13
<b>5</b>	<b>Catalogue</b>	<b>14</b>

---

\*Please report any issues at <https://github.com/ho-tex/oberdiek/issues>

<b>6 History</b>	<b>14</b>
[2007/12/12 v1.0]	14
[2009/04/10 v1.1]	14
[2010/03/09 v1.2]	14
[2010/12/13 v1.3]	14
[2011/03/29 v1.4]	14
[2011/04/22 v1.5]	15
[2011/04/23 v1.6]	15
[2011/10/22 v1.7]	15
[2011/11/01 v1.8]	15
[2016/05/13 v1.9]	15
[2016/05/16 v1.10]	15
[2018/11/22 v1.11]	15
<b>7 Index</b>	<b>15</b>

# 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

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 what-sits for color.

At last the attribute annotations of the nodes in the output box must be analyzed to insert the necessary color what-sits. Currently LuaTeX lacks an appropriate callback function. Therefore package `atbegshi` is used to get control before a box is shipped out.

`\luacolorProcessBox {⟨box⟩}`

Macro `\luacolorProcessBox` processes the box `⟨box⟩` 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%
3 \catcode13=5 % ^^M
4 \endlinechar=13 %
5 \catcode123=1 % {
6 \catcode125=2 % }
7 \catcode64=11 % @
8 \def\x{\endgroup
9 \expandafter\edef\csname LuaCol@AtEnd\endcsname{%
10 \endlinechar=\the\endlinechar\relax
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{%
28 \edef\LuaCol@AtEnd{%
29 \LuaCol@AtEnd
30 \catcode#1=\the\catcode#1\relax
31 }%
32 \catcode#1=#2\relax
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}%
54 [2018/11/22 v1.11 Color support via LuaTeX's attributes (HO)]

```

## 2.2 Check for LuaTeX

Without LuaTeX there is no point in using this package.

```

55 \RequirePackage{infwarerr}[2010/04/08]%
56 \RequirePackage{ifluatex}[2010/03/01]%
57 \RequirePackage{ifpdf}[2011/01/30]%
58 \RequirePackage{itxcmds}[2011/04/18]%
59 \RequirePackage{color}

```

require ltuatex rather than luatex package support for LuaTeX allocations.

```

60 \ifluatex
61   \ifx\newattribute\@undefined
62     \RequirePackage{ltuatex}%
63   \fi
64 \else
65   \@PackageError{luacolor}{%
66     This package may only be run using LuaTeX%
67   }\@ehc
68   \expandafter\LuaCol@AtEnd
69 \fi%

```

\LuaCol@directlua

```

70 \let\LuaCol@directlua\directlua

```

## 2.3 Check for disabled colors

```

71 \ifcolors@
72 \else
73   \@PackageWarningNoLine{luacolor}{%
74     Colors are disabled by option `monochrome'%
75   }%
76   \def\set@color{}%
77   \def\reset@color{}%
78   \def\set@page@color{}%
79   \def\define@color#1#2{}%
80   \expandafter\LuaCol@AtEnd
81 \fi%

```

## 2.4 Load module and check version

```

82 \LuaCol@directlua{%
83   require("luacolor")%
84 }

85 \begingroup
86   \edef\x{\LuaCol@directlua{tex.write("2018/11/22 v1.11")}}%
87   \edef\y{%
88     \LuaCol@directlua{%
89       if oberdiek.luacolor.getversion then %
90         oberdiek.luacolor.getversion()%
91       end%
92     }%
93   }%
94   \ifx\x\y
95   \else
96     \@PackageError{luacolor}{%
97       Wrong version of lua module.\MessageBreak
98       Package version: \x\MessageBreak
99       Lua module: \y
100    }\@ehc
101   \fi

```

```
102 \endgroup
```

## 2.5 Find driver

```
103 \ifpdf
104 \else
105 \begingroup
106 \def\current@color{}%
107 \def\reset@color{}%
108 \setbox\z@=\hbox{%
109 \begingroup
110 \set@color
111 \endgroup
112 }%
113 \edef\reserved@a{%
114 \LuaCol@directlua{%
115 oberdiek.luacolor.dvidetect()}%
116 }%
117 }%
118 \ifx\reserved@a\@empty
119 \@PackageError{luacolor}{%
120 DVI driver detection failed because of\MessageBreak
121 unrecognized color \string\special
122 }{\@ehc
123 \endgroup
124 \expandafter\expandafter\expandafter\LuaCol@AtEnd
125 \else
126 \@PackageInfoNoLine{luacolor}{%
127 Type of color \string\special: \reserved@a
128 }%
129 \fi%
130 \endgroup
131 \fi
```

## 2.6 Attribute setting

```
\LuaCol@Attribute
```

```
132 \ltx@ifundefined{newluatexattribute}{%
133 \newattribute\LuaCol@Attribute
134 }{%
135 \newluatexattribute\LuaCol@Attribute
136 }
137 \ltx@ifundefined{setluatexattribute}{%
138 \let\LuaCol@setattribute\setattribute
139 }{%
140 \let\LuaCol@setattribute\setluatexattribute
141 }
142 \LuaCol@directlua{%
143 oberdiek.luacolor.setattribute(\number\allocationnumber)%
144 }
```

```
\set@color
```

```
145 \protected\def\set@color{%
146 \LuaCol@setattribute\LuaCol@Attribute{%
147 \LuaCol@directlua{%
148 oberdiek.luacolor.get("\luaescapestring{\current@color}")%
149 }%
150 }%
151 }
```

```
\reset@color
```

```
152 \def\reset@color{}
```

## 2.7 Whatsit insertion

```
\luacolorProcessBox
153 \def\luacolorProcessBox#1{%
154   \LuaCol@directlua{%
155     oberdiek.luacolor.process(\number#1)%
156   }%
157 }

158 \RequirePackage{atbegshi}[2011/01/30]
159 \AtBeginShipout{%
160   \luacolorProcessBox\AtBeginShipoutBox
161 }

    Set default color.
162 \set@color
```

## 2.8 \pdfxform support

```
163 \ifpdf
164   \ifx\pdfxform\@undefined
165     \let\pdfxform\saveboxresource
166   \fi
167   \ltx@ifundefined{pdfxform}{%
168     \directlua{%
169       tex.enableprimitives('{%
170         'pdfxform','pdflastxform','pdfrefxform'%
171       })%
172     }%
173   }-%
174   \ltx@ifundefined{protected}{%
175     \directlua{tex.enableprimitives('{{'protected'}})}%
176   }-%
177   \ltx@ifundefined{pdfxform}{%
178     \@PackageWarning{luacolor}{\string\pdfxform\space not found}%
179   }-%
180   \let\LuaCol@org@pdfxform\pdfxform
181   \begingroup\expandafter\expandafter\expandafter\endgroup
182   \expandafter\ifx\csname protected\endcsname\relax
183     \@PackageWarning{luacolor}{\string\protected\space not found}%
184   \else
185     \expandafter\protected
186   \fi
187   \def\pdfxform{%
188     \begingroup
189     \afterassignment\LuaCol@pdfxform
190     \count@=%
191   }%
192   \def\LuaCol@pdfxform{%
193     \luacolorProcessBox\count@
194     \LuaCol@org@pdfxform\count@
195     \endgroup
196   }%
197 }-%
198 \fi

199 \LuaCol@AtEnd%
200 (/package)
```

## 2.9 Lua module

```
201 (*lua)
```

Box zero contains a `\hbox` with the color `\special`. That is analyzed to get the prefix for the color setting `\special`.

```

202 module("oberdiek.luacolor", package.seeall)
getversion()
203 function getversion()
204   tex.write("2018/11/22 v1.11")
205 end

2.9.1 Driver detection

206 local ifpdf
207 if tonumber(tex.outputmode or tex.pdfoutput) > 0 then
208   ifpdf = true
209 else
210   ifpdf = false
211 end
212 local prefix
213 local prefixes = {
214   dvips = "color ",
215   dvipdfm = "pdf:sc ",
216   truetex = "textcolor:",
217   pctexps = "ps::",
218 }
219 local patterns = {
220   ["^color "] = "dvips",
221   ["^pdf: *begincolor "] = "dvipdfm",
222   ["^pdf: *bcolor "] = "dvipdfm",
223   ["^pdf: *bc "] = "dvipdfm",
224   ["^pdf: *setcolor "] = "dvipdfm",
225   ["^pdf: *scolor "] = "dvipdfm",
226   ["^pdf: *sc "] = "dvipdfm",
227   ["^textcolor:"] = "truetex",
228   ["^ps::"] = "pctexps",
229 }

info()
230 local function info(msg, term)
231   local target = "log"
232   if term then
233     target = "term and log"
234   end
235   texio.write_nl(target, "Package luacolor info: " .. msg .. ".")
236   texio.write_nl(target, "")
237 end

dvidetect()
238 function dvidetect()
239   local v = tex.box[0]
240   assert(v.id == node.id("hlist"))
241   for v in node.traverse_id(node.id("whatsit"), v.head) do
242     if v and v.subtype == node.subtype("special") then
243       local data = v.data
244       for pattern, driver in pairs(patterns) do
245         if string.find(data, pattern) then
246           prefix = prefixes[driver]
247           tex.write(driver)
248           return
249         end
250       end
251       info("\\special{" .. data .. "}", true)
252       return
253     end
254   end
255   info("Missing \\special", true)
256 end

```

## 2.9.2 Color strings

```
257 local map = {
258   n = 0,
259 }

get()

260 function get(color)
261   tex.write("'" .. getvalue(color))
262 end

getvalue()

263 function getvalue(color)
264   local n = map[color]
265   if not n then
266     n = map.n + 1
267     map.n = n
268     map[n] = color
269     map[color] = n
270   end
271   return n
272 end
```

## 2.9.3 Attribute register

```
setattribute()

273 local attribute
274 function setattribute(attr)
275   attribute = attr
276 end

getattribute()

277 function getattribute()
278   return attribute
279 end
```

## 2.9.4 Whatsit insertion

```
280 local LIST = 1
281 local LIST_LEADERS = 2
282 local COLOR = 3
283 local RULE = node.id("rule")
284 local node_types = {
285   [node.id("hlist")] = LIST,
286   [node.id("vlist")] = LIST,
287   [node.id("rule")] = COLOR,
288   [node.id("glyph")] = COLOR,
289   [node.id("disc")] = COLOR,
290   [node.id("whatsit")] = {
291     [node.subtype("special")] = COLOR,
292     [node.subtype("pdf_literal")] = COLOR,
293     [node.subtype("pdf_save")] = COLOR,
294     [node.subtype("pdf_restore")] = COLOR, -- probably not needed
295     -- TODO (DPC) [node.subtype("pdf_refximage")] = COLOR,
296   },
297   [node.id("glue")] =
298     function(n)
299       if n.subtype >= 100 then -- leaders
300         if n.leader.id == RULE then
301           return COLOR
302         else
303           return LIST_LEADERS
304         end
305       end
306     end
```



```

305     end
306   end,
307 }

get_type()

308 local function get_type(n)
309   local ret = node_types[n.id]
310   if type(ret) == 'table' then
311     ret = ret[n.subtype]
312   end
313   if type(ret) == 'function' then
314     ret = ret(n)
315   end
316   return ret
317 end

318 local mode = 2 -- luatex.pdfiteral.direct
319 local WHATSIT = node.id("whatsit")
320 local SPECIAL = node.subtype("special")
321 local PDFLITERAL = node.subtype("pdf_literal")
322 local DRY_FALSE = false
323 local DRY_TRUE = true

traverse()

324 local function traverse(list, color, dry)
325   if not list then
326     return color
327   end
328   if get_type(list) ~= LIST then
329     texio.write_nl("!!! Error: Wrong list type: " .. node.type(list.id))
330     return color
331   end
332   (debug)texio.write_nl("traverse: " .. node.type(list.id))
333   local head = list.head
334   for n in node.traverse(head) do
335     (debug)texio.write_nl(" node: " .. node.type(n.id))
336     local t = get_type(n)
337     (debug)texio.write_nl("TYPE " .. tostring(t) .. " " .. tostring(node.type(node.getid(n))) .. " " .. tostring(node.get-
      subtype(n)))
338     if t == LIST then
339       color = traverse(n, color, dry)
340     elseif t == LIST_LEADERS then
341       local color_after = traverse(n.leader, color, DRY_TRUE)
342       if color == color_after then
343         traverse(n.leader, color, DRY_FALSE or dry)
344       else
345         traverse(n.leader, "", DRY_FALSE or dry)
346       % The color status is unknown here, because the leader box
347       % will or will not be set.
348       color = ""
349     end
350     elseif t == COLOR then
351       local v = node.has_attribute(n, attribute)
352       if v then
353         local newColor = map[v]
354         if newColor ~= color then
355           color = newColor
356         if dry == DRY_FALSE then
357           local newNode
358           if ifpdf then
359             newNode = node.new(WHATSIT, PDFLITERAL)
360             newNode.mode = mode
361             newNode.data = color

```

```

362     else
363         newNode = node.new(WHATSIT, SPECIAL)
364         newNode.data = prefix .. color
365     end
366     head = node.insert_before(head, n, newNode)
367 end
368 end
369 end
370 end
371 end
372 list.head = head
373 return color
374 end

```

process()

```

375 function process(box)
376 local color = ""
377 local list = tex.getbox(box)
378 traverse(list, color, DRY_FALSE)
379 end

380 </lua>

```

### 3 Test

```

381 <*test1>
382 \documentclass{article}
383 \usepackage{color}
384 </test1>

```

#### 3.1 Catcode checks for loading

```

385 <*test1>
386 \catcode`\{=1 %
387 \catcode`\}=2 %
388 \catcode`\#=6 %
389 \catcode`\@=11 %
390 \expandafter\ifx\csname count@\endcsname\relax
391 \countdef\count@=255 %
392 \fi
393 \expandafter\ifx\csname @gobble\endcsname\relax
394 \long\def\@gobble#1{}%
395 \fi
396 \expandafter\ifx\csname @firstofone\endcsname\relax
397 \long\def\@firstofone#1{#1}%
398 \fi
399 \expandafter\ifx\csname loop\endcsname\relax
400 \expandafter\@firstofone
401 \else
402 \expandafter\@gobble
403 \fi
404 {%
405 \def\loop#1\repeat{%
406 \def\body{#1}%
407 \iterate
408 }%
409 \def\iterate{%
410 \body
411 \let\next\iterate
412 \else
413 \let\next\relax
414 \fi

```

```

415 \next
416 }%
417 \let\repeat=\fi
418 }%
419 \def\RestoreCatcodes{}
420 \count@=0 %
421 \loop
422 \edef\RestoreCatcodes{%
423 \RestoreCatcodes
424 \catcode\the\count@=\the\catcode\count@\relax
425 }%
426 \ifnum\count@<255 %
427 \advance\count@ 1 %
428 \repeat
429
430 \def\RangeCatcodeInvalid#1#2{%
431 \count@=#1\relax
432 \loop
433 \catcode\count@=15 %
434 \ifnum\count@<#2\relax
435 \advance\count@ 1 %
436 \repeat
437 }
438 \def\RangeCatcodeCheck#1#2#3{%
439 \count@=#1\relax
440 \loop
441 \ifnum#3=\catcode\count@
442 \else
443 \errmessage{%
444 Character \the\count@\space
445 with wrong catcode \the\catcode\count@\space
446 instead of \number#3%
447 }%
448 \fi
449 \ifnum\count@<#2\relax
450 \advance\count@ 1 %
451 \repeat
452 }
453 \def\space{ }
454 \expandafter\ifx\csname LoadCommand\endcsname\relax
455 \def\LoadCommand{\input luacolor.sty\relax}%
456 \fi
457 \def\Test{%
458 \RangeCatcodeInvalid{0}{47}%
459 \RangeCatcodeInvalid{58}{64}%
460 \RangeCatcodeInvalid{91}{96}%
461 \RangeCatcodeInvalid{123}{255}%
462 \catcode`\@=12 %
463 \catcode`\=\=0 %
464 \catcode`\%=14 %
465 \LoadCommand
466 \RangeCatcodeCheck{0}{36}{15}%
467 \RangeCatcodeCheck{37}{37}{14}%
468 \RangeCatcodeCheck{38}{47}{15}%
469 \RangeCatcodeCheck{48}{57}{12}%
470 \RangeCatcodeCheck{58}{63}{15}%
471 \RangeCatcodeCheck{64}{64}{12}%
472 \RangeCatcodeCheck{65}{90}{11}%
473 \RangeCatcodeCheck{91}{91}{15}%
474 \RangeCatcodeCheck{92}{92}{0}%
475 \RangeCatcodeCheck{93}{96}{15}%
476 \RangeCatcodeCheck{97}{122}{11}%

```

```

477 \RangeCatcodeCheck{123}{255}{15}%
478 \RestoreCatcodes
479 }
480 \Test
481 \csname @@end\endcsname
482 \end
483 </test1>

```

### 3.2 Driver detection

```

484 <*test2>
485 \NeedsTeXFormat{LaTeX2e}
486 \ifcsname driver\endcsname
487 \expandafter\PassOptionsToPackage\expandafter{\driver}{color}%
488 \pdfoutput=0 %
489 \fi
490 \documentclass{minimal}
491 \usepackage{luacolor}[2018/11/01]
492 \csname @@end\endcsname
493 \end
494 </test2>

495 <*test3>
496 \NeedsTeXFormat{LaTeX2e}
497 \documentclass{minimal}
498 \usepackage{luacolor}[2018/11/01]
499 \usepackage{qstest}
500 \IncludeTests{*}
501 \LogTests{log}{*}{*}
502 \makeatletter

503 \@@end
504 </test3>

```

## 4 Installation

### 4.1 Download

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

[CTAN:macros/latex/contrib/oberdiek/luacolor.dtx](#) The source file.

[CTAN:macros/latex/contrib/oberdiek/luacolor.pdf](#) Documentation.

**Bundle.** All the packages of the bundle ‘oberdiek’ 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/oberdiek.tds.zip](#)

*TDS* refers to the standard “A Directory Structure for T<sub>E</sub>X Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

### 4.2 Bundle installation

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

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

---

<sup>1</sup><http://ctan.org/pkg/luacolor>

**Script installation.** Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package `attachfile2` comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

### 4.3 Package installation

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

```
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):

```
luacolor.sty → tex/latex/oberdiek/luacolor.sty
luacolor.lua → scripts/oberdiek/luacolor.lua
luacolor.pdf → doc/latex/oberdiek/luacolor.pdf
luacolor.dtx → source/latex/oberdiek/luacolor.dtx
```

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`.

### 4.4 Refresh file name databases

If your `TEX` distribution (te`TEX`, mik`TEX`, ...) relies on file name databases, you must refresh these. For example, te`TEX` users run `texhash` or `mktextlsr`.

### 4.5 Some details for the interested

**Unpacking with L<sup>A</sup>T<sub>E</sub>X.** The `.dtx` chooses its action depending on the format:

**plain T<sub>E</sub>X:** Run `docstrip` and extract the files.

**L<sup>A</sup>T<sub>E</sub>X:** Generate the documentation.

If you insist on using L<sup>A</sup>T<sub>E</sub>X for `docstrip` (really, `docstrip` does not need L<sup>A</sup>T<sub>E</sub>X), 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 pdfL<sup>A</sup>T<sub>E</sub>X:

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

## 5 Catalogue

The following XML file can be used as source for the [T<sub>E</sub>X Catalogue](#). The elements `caption` and `description` are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is `luacolor.xml`.

```
505 (*catalogue)
506 <?xml version='1.0' encoding='us-ascii'?>
507 <!DOCTYPE entry SYSTEM 'catalogue.dtd'>
508 <entry datestamp='$Date$' modifier='$Author$' id='luacolor'>
509   <name>luacolor</name>
510   <caption>Color support based on LuaTeX's node attributes.</caption>
511   <authorref id='auth:oberdiek' />
512   <copyright owner='Heiko Oberdiek' year='2007,2009-2011' />
513   <license type='lppl1.3' />
514   <version number='1.11' />
515   <description>
516     This package implements color support based on LuaTeX's node
517     attributes.
518   <p />
519   The package is part of the <xref refid='oberdiek'>oberdiek</xref> bundle.
520 </description>
521 <documentation details='Package documentation'
522   href='ctan:/macros/latex/contrib/oberdiek/luacolor.pdf' />
523 <ctan file='true' path='/macros/latex/contrib/oberdiek/luacolor.dtx' />
524 <miktex location='oberdiek' />
525 <texlive location='oberdiek' />
526 <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip' />
527 </entry>
528 </catalogue>
```

## 6 History

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

- First public version.

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

- Fixes for changed syntax of `\directlua` in LuaT<sub>E</sub>X 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 LuaT<sub>E</sub>X: ‘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 `\pdfifliteral` 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 LuaTeX 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

## 7 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.

<b>Symbols</b>	<code>\%</code> .....	<i>464</i>	
<code>\#</code> .....	<i>388</i>	<code>\@</code> .....	<i>389, 462</i>

<code>\@end</code>	503	<code>\ifcname</code>	486
<code>\@PackageError</code>	65, 96, 119	<code>\ifuatex</code>	60
<code>\@PackageInfoNoLine</code>	126	<code>\ifnum</code>	426, 434, 441, 449
<code>\@PackageWarning</code>	178, 183	<code>\ifpdf</code>	103, 163
<code>\@PackageWarningNoLine</code>	73	<code>\ifx</code>	61, 94, 118, 164, 182, 390, 393, 396, 399, 454
<code>\@ehc</code>	67, 100, 122	<code>\IncludeTests</code>	500
<code>\@empty</code>	118	<code>\info()</code>	230
<code>\@firstofone</code>	397, 400	<code>\input</code>	455
<code>\@gobble</code>	394, 402	<code>\iterate</code>	407, 409, 411
<code>\@undefined</code>	61, 164		
<code>\@</code>	251, 255, 463		
<code>\{</code>	386		
<code>\}</code>	387		
		<b>L</b>	
<b>A</b>		<code>\LoadCommand</code>	455, 465
<code>\advance</code>	427, 435, 450	<code>\LogTests</code>	501
<code>\afterassignment</code>	189	<code>\loop</code>	405, 421, 432, 440
<code>\allocationnumber</code>	143	<code>\ltx@ifundefined</code>	132, 137, 167, 174, 177
<code>\AtBeginShipout</code>	159	<code>\LuaCol@AtEnd</code>	
<code>\AtBeginShipoutBox</code>	160		28, 29, 51, 68, 80, 124, 199
		<code>\LuaCol@Attribute</code>	132, 146
<b>B</b>		<code>\LuaCol@directlua</code>	
<code>\body</code>	406, 410		70, 82, 86, 88, 114, 142, 147, 154
		<code>\LuaCol@org@pdfxform</code>	180, 194
<b>C</b>		<code>\LuaCol@pdfxform</code>	189, 192
<code>\catcode</code>	2, 3, 5, 6, 7, 11, 12, 13, 14, 15, 16, 17, 20, 21, 23, 24, 25, 26, 30, 32, 386, 387, 388, 389, 424, 433, 441, 445, 462, 463, 464	<code>\LuaCol@setattribute</code>	138, 140, 146
<code>\count@</code>	190, 193, 194, 391, 420, 424, 426, 427, 431, 433, 434, 435, 439, 441, 444, 445, 449, 450	<code>\luacolorProcessBox</code>	2, 153, 160, 193
<code>\countdef</code>	391	<code>\luaescapestring</code>	148
<code>\csname</code>	9, 182, 390, 393, 396, 399, 454, 481, 492		
<code>\current@color</code>	106, 148	<b>M</b>	
		<code>\makeatletter</code>	502
<b>D</b>		<code>\MessageBreak</code>	97, 98, 120
<code>\define@color</code>	79		
<code>\directlua</code>	70, 168, 175	<b>N</b>	
<code>\documentclass</code>	382, 490, 497	<code>\NeedsTeXFormat</code>	52, 485, 496
<code>\driver</code>	487	<code>\newattribute</code>	61, 133
<code>\dvidetect()</code>	238	<code>\newluatexattribute</code>	135
		<code>\next</code>	411, 413, 415
<b>E</b>		<code>\number</code>	143, 155, 446
<code>\end</code>	482, 493		
<code>\endcsname</code>	9, 182, 390, 393, 396, 399, 454, 481, 486, 492	<b>P</b>	
<code>\endinput</code>	51	<code>\PassOptionsToPackage</code>	487
<code>\endlinechar</code>	4, 10, 22	<code>\pdfoutput</code>	488
<code>\errmessage</code>	443	<code>\pdfxform</code>	164, 165, 178, 180, 187
		<code>\process()</code>	375
<b>G</b>		<code>\protected</code>	145, 183, 185
<code>\get()</code>	260	<code>\ProvidesPackage</code>	53
<code>\get_type()</code>	308		
<code>\getattribute()</code>	277	<b>R</b>	
<code>\getvalue()</code>	263	<code>\RangeCatcodeCheck</code>	
<code>\getversion()</code>	203		438, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477
		<code>\RangeCatcodeInvalid</code>	
<b>H</b>			430, 458, 459, 460, 461
<code>\hbox</code>	108	<code>\repeat</code>	405, 417, 428, 436, 451
		<code>\RequirePackage</code>	55, 56, 57, 58, 59, 62, 158
<b>I</b>		<code>\reserved@a</code>	113, 118, 127
<code>\ifcolors@</code>	71	<code>\reset@color</code>	77, 107, 152
		<code>\RestoreCatcodes</code>	419, 422, 423, 478
		<b>S</b>	
		<code>\saveboxresource</code>	165
		<code>\set@color</code>	76, 110, 145, 162
		<code>\set@page@color</code>	78
		<code>\setattribute</code>	138
		<code>\setattribute()</code>	273
		<code>\setbox</code>	108



<code>\setluatexattribute</code> .....	140		
<code>\space</code> .....	178, 183, 444, 445, 453		
<code>\special</code> .....	121, 127		
<b>T</b>			
<code>\Test</code> .....	457, 480		
<code>\the</code> .....	10, 11, 12,		
	13, 14, 15, 16, 17, 30, 424, 444, 445		
<code>\TMP@EnsureCode</code> .....	27,		
	34, 35, 36, 37, 38, 39, 40, 41,		
	42, 43, 44, 45, 46, 47, 48, 49, 50		
<code>\traverse()</code> .....	<u>324</u>		
		<b>U</b>	
		<code>\usepackage</code> .....	383, 491, 498, 499
		<b>X</b>	
		<code>\x</code> .....	8, 20, 86, 94, 98
		<b>Y</b>	
		<code>\y</code> .....	87, 94, 99
		<b>Z</b>	
		<code>\z@</code> .....	108