

The luacolor package

Heiko Oberdiek
<heiko.oberdiek at gmail.com>

2011/04/22 v1.5

Abstract

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

Contents

1	Documentation	2
1.1	Introduction	2
1.2	Usage	2
1.3	Limitations	2
2	Implementation	2
2.1	Catcodes and identification	2
2.2	Check for LuaTeX	3
2.3	Check for disabled colors	4
2.4	Find driver	4
2.5	Attribute setting	5
2.6	Whatsit insertion	5
2.7	\pdfxform support	5
2.8	Lua module	6
2.8.1	Driver detection	6
2.8.2	Color strings	7
2.8.3	Attribute register	7
2.8.4	Whatsit insertion	7
3	Test	9
3.1	Catcode checks for loading	9
3.2	Driver detection	11
4	Installation	11
4.1	Download	11
4.2	Bundle installation	12
4.3	Package installation	12
4.4	Refresh file name databases	12
4.5	Some details for the interested	12
5	History	13
	[2007/12/12 v1.0]	13
	[2009/04/10 v1.1]	13
	[2010/03/09 v1.2]	13
	[2010/12/13 v1.3]	13
	[2011/03/29 v1.4]	13
	[2011/04/22 v1.5]	13
6	Index	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

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 [2011/04/22 v1.5 Coloring based on LuaTeX's node attributes (H0)]

```

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{ltxcmds}[2011/04/18]%
59 \RequirePackage{color}

60 \ifluatex
61   \ltx@ifpackageloaded{luatexbase-attr}{%
62   }{%
63     \RequirePackage{luatex}[2010/03/09]%
64   }%
65 \else

```

```

66 \PackageError{luacolor}{%
67   This package may only be run using LuaTeX%
68 }{\@ehc
69 \expandafter\LuaCol@AtEnd
70 \fi%

```

`\LuaCol@directlua`

```

71 \ifnum\luatexversion<36 %
72 \def\LuaCol@directlua{\directlua0 }%
73 \else
74 \let\LuaCol@directlua\directlua
75 \fi

```

2.3 Check for disabled colors

```

76 \ifcolors@
77 \else
78 \PackageWarningNoLine{luacolor}{%
79   Colors are disabled by option ‘monochrome’%
80 }%
81 \def\set@color{}%
82 \def\reset@color{}%
83 \def\set@page@color{}%
84 \def\define@color#1#2{}%
85 \expandafter\LuaCol@AtEnd
86 \fi%

```

2.4 Find driver

```

87 \LuaCol@directlua{%
88   require("oberdiek.luacolor\ifnum\luatexversion<65 -pre065\fi")%
89 }
90 \ifpdf
91 \else
92 \begingroup
93 \def\current@color{}%
94 \def\reset@color{}%
95 \setbox\z@=\hbox{%
96   \begingroup
97   \set@color
98   \endgroup
99 }%
100 \edef\reserved@a{%
101   \LuaCol@directlua{%
102     oberdiek.luacolor.dvidetect()%
103   }%
104 }%
105 \ifx\reserved@a\@empty
106 \PackageError{luacolor}{%
107   DVI driver detection failed because of\MessageBreak
108   unrecognized color \string\special
109 }{\@ehc
110 \endgroup
111 \expandafter\expandafter\expandafter\LuaCol@AtEnd
112 \else
113 \PackageInfoNoLine{luacolor}{%
114   Type of color \string\special: \reserved@a
115 }%
116 \fi%
117 \endgroup
118 \fi

```

2.5 Attribute setting

```
\LuaCol@Attribute
119 \ltx@ifundefined{newluatexattribute}{%
120   \newattribute\LuaCol@Attribute
121 }{%
122   \newluatexattribute\LuaCol@Attribute
123 }
124 \ltx@ifundefined{setluatexattribute}{%
125   \let\LuaCol@setattribute\setattribute
126 }{%
127   \let\LuaCol@setattribute\setluatexattribute
128 }
129 \LuaCol@directlua{%
130   oberdiek.luacolor.setattribute(\number\allocationnumber)%
131 }

\set@color
132 \protected\def\set@color{%
133   \LuaCol@setattribute\LuaCol@Attribute{%
134     \LuaCol@directlua{%
135       oberdiek.luacolor.get("\luatexluaescapestring{\current@color}")%
136     }%
137   }%
138 }

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

2.6 Whatsit insertion

```
\luacolorProcessBox
140 \def\luacolorProcessBox#1{%
141   \LuaCol@directlua{%
142     oberdiek.luacolor.process(\number#1)%
143   }%
144 }

145 \RequirePackage{atbegshi}[2011/01/30]
146 \AtBeginShipout{%
147   \luacolorProcessBox\AtBeginShipoutBox
148 }

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

2.7 \pdfxform support

```
150 \ifpdf
151   \ltx@ifundefined{pdfxform}{%
152     \ifnum\luatexversion>36 %
153       \directlua{%
154         tex.enableprimitives('',{%
155           'pdfxform','pdflastxform','pdfrefxform'%
156         })%
157       }%
158     \fi
159   }{}%
160   \ltx@ifundefined{protected}{%
161     \ifnum\luatexversion>36 %
162       \directlua{tex.enableprimitives('',{'protected'})}%
163     \fi
```

```

164 }{}%
165 \ltx@ifundefined{pdfxform}{%
166   \@PackageWarning{luacolor}{\string\pdfxform\space not found}%
167 }{%
168   \let\LuaCol@org@pdfxform\pdfxform
169   \begingroup\expandafter\expandafter\expandafter\endgroup
170   \expandafter\ifx\csname protected\endcsname\relax
171     \@PackageWarning{luacolor}{\string\protected\space not found}%
172   \else
173     \expandafter\protected
174   \fi
175   \def\pdfxform{%
176     \begingroup
177     \afterassignment\LuaCol@pdfxform
178     \count@=%
179   }%
180   \def\LuaCol@pdfxform{%
181     \luacolorProcessBox\count@
182     \LuaCol@org@pdfxform\count@
183   \endgroup
184 }%
185 }%
186 \fi

187 \LuaCol@AtEnd%
188 \</package>

```

2.8 Lua module

```
189 <*lua>
```

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

```
190 module("oberdiek.luacolor", package.seeall)
```

2.8.1 Driver detection

```

191 local ifpdf
192 if tonumber(tex.pdfoutput) > 0 then
193   ifpdf = true
194 else
195   ifpdf = false
196 end
197 local prefix
198 local prefixes = {
199   dvips    = "color ",
200   dvipdfm  = "pdf:sc ",
201   truetex  = "textcolor:",
202   pctexps  = "ps:.",
203 }
204 local patterns = {
205   ["^color "]      = "dvips",
206   ["^pdf: *begincolor "] = "dvipdfm",
207   ["^pdf: *bcolor "]   = "dvipdfm",
208   ["^pdf: *bc "]       = "dvipdfm",
209   ["^pdf: *setcolor "]  = "dvipdfm",
210   ["^pdf: *scolor "]    = "dvipdfm",
211   ["^pdf: *sc "]       = "dvipdfm",
212   ["^textcolor:"]      = "truetex",
213   ["^ps:."]           = "pctexps",
214 }

215 local function info(msg, term)
216   local target = "log"
217   if term then
218     target = "term and log"

```

```

219 end
220 texio.write_nl(target, "Package luacolor info: " .. msg .. ".")
221 texio.write_nl(target, "")
222 end

223 function dvidetect()
224   local v = tex.box[0]
225   assert(v.id == node.id("hlist"))
226   for v in node.traverse_id(node.id("whatsit"), v.head) do
227   for v in node.traverse_id(node.id("whatsit"), v.list) do
228     if v and v.subtype == 3 then -- special
229       local data = v.data
230       for pattern, driver in pairs(patterns) do
231         if string.find(data, pattern) then
232           prefix = prefixes[driver]
233           tex.write(driver)
234           return
235         end
236       end
237       info("\\special{" .. data .. "}", true)
238       return
239     end
240   end
241   info("Missing \\special", true)
242 end

```

2.8.2 Color strings

```

243 local map = {
244   n = 0,
245 }
246 function get(color)
247   local n = map[color]
248   if not n then
249     n = map.n + 1
250     map.n = n
251     map[n] = color
252     map[color] = n
253   end
254   tex.write("'" .. n)
255 end

```

2.8.3 Attribute register

```

256 local attribute
257 function setattribute(attr)
258   attribute = attr
259 end

```

2.8.4 Whatsit insertion

```

260 local LIST = 1
261 local LIST_LEADERS = 2
262 local COLOR = 3
263 local RULE = node.id("rule")
264 local node_types = {
265   [node.id("hlist")] = LIST,
266   [node.id("vlist")] = LIST,
267   [node.id("rule")] = COLOR,
268   [node.id("glyph")] = COLOR,
269   [node.id("disc")] = COLOR,
270   [node.id("whatsit")] = {
271     [3] = COLOR, -- special
272     [8] = COLOR, -- pdf_literal
273     [14] = COLOR, -- pdf_refximage
274   },

```

```

275 [node.id("glue")] =
276   function(n)
277     if n.subtype >= 100 then -- leaders
278       if n.leader.id == RULE then
279         return COLOR
280       else
281         return LIST_LEADERS
282       end
283     end
284   end,
285 }
286 local function get_type(n)
287   local ret = node_types[n.id]
288   if type(ret) == 'table' then
289     ret = ret[n.subtype]
290   end
291   if type(ret) == 'function' then
292     ret = ret(n)
293   end
294   return ret
295 end
296 local mode = 2 -- luatex.pdfliteral.direct
297 local WHATSIT = node.id("whatsit")
298 local SPECIAL = 3
299 local PDFLITERAL = 8
300 local DRY_FALSE = 0
301 local DRY_TRUE = 1
302 local function traverse(list, color, dry)
303   if not list then
304     return color
305   end
306   if get_type(list) ~= LIST then
307     texio.write_nl("!!! Error: Wrong list type: " .. node.type(list.id))
308     return color
309   end
310   (debug)texio.write_nl("traverse: " .. node.type(list.id))
311   (lpre065) local head = list.head
312   (pre065) local head = list.list
313   for n in node.traverse(head) do
314     (debug)texio.write_nl(" node: " .. node.type(n.id))
315     local t = get_type(n)
316     if t == LIST then
317       color = traverse(n, color, DRY_FALSE)
318     elseif t == LIST_LEADERS then
319       local color_after = traverse(n.leader, color, DRY_TRUE)
320       if color == color_after then
321         traverse(n.leader, color, DRY_FALSE)
322       else
323         traverse(n.leader, '', DRY_FALSE)
324       end
325     elseif t == COLOR then
326       local v = node.has_attribute(n, attribute)
327       if v then
328         local newColor = map[v]
329         if newColor ~= color then
330           color = newColor
331           if dry == DRY_FALSE then
332             local newNode
333             if ifpdf then
334               newNode = node.new(WHATSIT, PDFLITERAL)
335               newNode.mode = mode
336               newNode.data = color

```



```

337         else
338             newNode = node.new(WHATSIT, SPECIAL)
339             newNode.data = prefix .. color
340         end
341     <!*pre065>
342     head = node.insert_before(head, n, newNode)
343 </!pre065>
344 <!*pre065>
345         if head == n then
346             newNode.next = head
347             local old_prev = head.prev
348             head.prev = newNode
349             head = newNode
350             head.prev = old_prev
351         else
352             head = node.insert_before(head, n, newNode)
353         end
354 </pre065>
355     end
356 end
357 end
358 end
359 end
360 <!pre065> list.head = head
361 <pre065> list.list = head
362 return color
363 end

364 function process(box)
365     local color = ""
366     local list = tex.getbox(box)
367     traverse(list, color, DRY_FALSE)
368 end

369 </lua>

```

3 Test

```

370 <!*test1>
371 \documentclass{article}
372 \usepackage{color}
373 </test1>

```

3.1 Catcode checks for loading

```

374 <!*test1>
375 \catcode'\{=1 %
376 \catcode'\}=2 %
377 \catcode'\#=6 %
378 \catcode'\@=11 %
379 \expandafter\ifx\csname count@\endcsname\relax
380     \countdef\count@=255 %
381 \fi
382 \expandafter\ifx\csname @gobble\endcsname\relax
383     \long\def\@gobble#1{}%
384 \fi
385 \expandafter\ifx\csname @firstofone\endcsname\relax
386     \long\def\@firstofone#1{#1}%
387 \fi
388 \expandafter\ifx\csname loop\endcsname\relax
389     \expandafter\@firstofone
390 \else
391     \expandafter\@gobble
392 \fi

```

```

393 {%
394   \def\loop#1\repeat{%
395     \def\body{#1}%
396     \iterate
397   }%
398   \def\iterate{%
399     \body
400     \let\next\iterate
401     \else
402     \let\next\relax
403     \fi
404     \next
405   }%
406   \let\repeat=\fi
407 }%
408 \def\RestoreCatcodes{}
409 \count@=0 %
410 \loop
411   \edef\RestoreCatcodes{%
412     \RestoreCatcodes
413     \catcode\the\count@=\the\catcode\count@\relax
414   }%
415   \ifnum\count@<255 %
416     \advance\count@ 1 %
417   \repeat
418
419 \def\RangeCatcodeInvalid#1#2{%
420   \count@=#1\relax
421   \loop
422     \catcode\count@=15 %
423     \ifnum\count@<#2\relax
424       \advance\count@ 1 %
425     \repeat
426 }
427 \def\RangeCatcodeCheck#1#2#3{%
428   \count@=#1\relax
429   \loop
430     \ifnum#3=\catcode\count@
431     \else
432       \errmessage{%
433         Character \the\count@\space
434         with wrong catcode \the\catcode\count@\space
435         instead of \number#3%
436       }%
437     \fi
438     \ifnum\count@<#2\relax
439       \advance\count@ 1 %
440     \repeat
441 }
442 \def\space{ }
443 \expandafter\ifx\csname LoadCommand\endcsname\relax
444   \def\LoadCommand{\input luacolor.sty\relax}%
445 \fi
446 \def\Test{%
447   \RangeCatcodeInvalid{0}{47}%
448   \RangeCatcodeInvalid{58}{64}%
449   \RangeCatcodeInvalid{91}{96}%
450   \RangeCatcodeInvalid{123}{255}%
451   \catcode'\@=12 %
452   \catcode'\=0 %
453   \catcode'\%=14 %
454   \LoadCommand

```

```

455 \RangeCatcodeCheck{0}{36}{15}%
456 \RangeCatcodeCheck{37}{37}{14}%
457 \RangeCatcodeCheck{38}{47}{15}%
458 \RangeCatcodeCheck{48}{57}{12}%
459 \RangeCatcodeCheck{58}{63}{15}%
460 \RangeCatcodeCheck{64}{64}{12}%
461 \RangeCatcodeCheck{65}{90}{11}%
462 \RangeCatcodeCheck{91}{91}{15}%
463 \RangeCatcodeCheck{92}{92}{0}%
464 \RangeCatcodeCheck{93}{96}{15}%
465 \RangeCatcodeCheck{97}{122}{11}%
466 \RangeCatcodeCheck{123}{255}{15}%
467 \RestoreCatcodes
468 }
469 \Test
470 \csname @@end\endcsname
471 \end
472 </test1>

```

3.2 Driver detection

```

473 <*test2>
474 \NeedsTeXFormat{LaTeX2e}
475 \ifcsname driver\endcsname
476   \expandafter\PassOptionsToPackage\expandafter{\driver}{color}%
477   \pdfoutput=0 %
478 \fi
479 \documentclass{minimal}
480 \usepackage{luacolor}[2011/04/22]
481 \csname @@end\endcsname
482 \end
483 </test2>
484 <*test3>
485 \NeedsTeXFormat{LaTeX2e}
486 \documentclass{minimal}
487 \usepackage{luacolor}[2011/04/22]
488 \usepackage{qstest}
489 \IncludeTests{*}
490 \LogTests{log}{*}{*}
491 \makeatletter
492 \@@end
493 </test3>

```

4 Installation

4.1 Download

Package. This package is available on CTAN¹:

[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_EX Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

¹[ftp://ftp.ctan.org/tex-archive/](http://ftp.ctan.org/tex-archive/)

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
```

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 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$:

```
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):

<code>luacolor.sty</code>	\rightarrow <code>tex/latex/oberdiek/luacolor.sty</code>
<code>oberdiek.luacolor.lua</code>	\rightarrow <code>scripts/oberdiek/oberdiek.luacolor.lua</code>
<code>luacolor.lua</code>	\rightarrow <code>scripts/oberdiek/luacolor.lua</code>
<code>oberdiek.luacolor-pre065.lua</code>	\rightarrow <code>scripts/oberdiek/oberdiek.luacolor-pre065.lua</code>
<code>luacolor-pre065.lua</code>	\rightarrow <code>scripts/oberdiek/luacolor-pre065.lua</code>
<code>luacolor.pdf</code>	\rightarrow <code>doc/latex/oberdiek/luacolor.pdf</code>
<code>test/luacolor-test1.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/luacolor-test1.tex</code>
<code>test/luacolor-test2.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/luacolor-test2.tex</code>
<code>test/luacolor-test3.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/luacolor-test3.tex</code>
<code>luacolor.dtx</code>	\rightarrow <code>source/latex/oberdiek/luacolor.dtx</code>

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 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ distribution (`te $\mathrm{T}_{\mathrm{E}}\mathrm{X}$` , `mik $\mathrm{T}_{\mathrm{E}}\mathrm{X}$` , ...) relies on file name databases, you must refresh these. For example, `te $\mathrm{T}_{\mathrm{E}}\mathrm{X}$` users run `texhash` or `mktextlsr`.

4.5 Some details for the interested

Attached source. The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is `pdftk`, e.g. unpack the file into the current directory:

```
pdftk luacolor.pdf unpack_files output .
```

Unpacking with $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$. The `.dtx` chooses its action depending on the format:

plain $\mathrm{T}_{\mathrm{E}}\mathrm{X}$: Run `docstrip` and extract the files.

$\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$: Generate the documentation.

If you insist on using $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ for `docstrip` (really, `docstrip` does not need $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$), 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^AT_EX:

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

5 History

[2007/12/12 v1.0]

- First public version.

[2009/04/10 v1.1]

- Fixes for changed syntax of `\directlua` in LuaT_EX 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_EX: ‘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_EX 0.65, because bug was fixed in 0.27. (Thanks Manuel Pégourié-Gonnard.)

6 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.

Symbols		I	
\#	377	\ifcolors@	76
\%	453	\ifcsname	475
\@	378, 451	\ifluatex	60
\@end	492	\ifnum	71, 88, 152, 161, 415, 423, 430, 438
\@PackageError	66, 106	\ifpdf	90, 150
\@PackageInfoNoLine	113	\ifx	105, 170, 379, 382, 385, 388, 443
\@PackageWarning	166, 171	\IncludeTests	489
\@PackageWarningNoLine	78	\input	444
\@ehc	68, 109	\iterate	396, 398, 400
\@empty	105		
\@firstofone	386, 389	L	
\@gobble	383, 391	\LoadCommand	444, 454
\\	237, 241, 452	\LogTests	490
\{	375	\loop	394, 410, 421, 429
\}	376	\ltx@ifpackageloaded	61
		\ltx@ifundefined	119, 124, 151, 160, 165
A		\LuaCol@AtEnd	28, 29, 51, 69, 85, 111, 187
\advance	416, 424, 439	\LuaCol@Attribute	119, 133
\afterassignment	177	\LuaCol@directlua	
\allocationnumber	130		71, 87, 101, 129, 134, 141
\AtBeginShipout	146	\LuaCol@org@pdfxform	168, 182
\AtBeginShipoutBox	147	\LuaCol@pdfxform	177, 180
		\LuaCol@setattribute	125, 127, 133
B		\luacolorProcessBox	2, 140, 147, 181
\body	395, 399	\luatexluaescapestring	135
		\luatexversion	71, 88, 152, 161
C		M	
\catcode	2, 3, 5, 6, 7, 11, 12, 13, 14, 15, 16, 17, 20, 21, 23, 24, 25, 26, 30, 32, 375, 376, 377, 378, 413, 422, 430, 434, 451, 452, 453	\makeatletter	491
\count@	178, 181, 182, 380, 409, 413, 415, 416, 420, 422, 423, 424, 428, 430, 433, 434, 438, 439	\MessageBreak	107
\countdef	380		
\csname	9, 170, 379, 382, 385, 388, 443, 470, 481	N	
\current@color	93, 135	\NeedsTeXFormat	52, 474, 485
		\newattribute	120
D		\newluatexattribute	122
\define@color	84	\next	400, 402, 404
\directlua	72, 74, 153, 162	\number	130, 142, 435
\documentclass	371, 479, 486		
\driver	476	P	
E		\PassOptionsToPackage	476
\end	471, 482	\pdfoutput	477
\endcsname	9, 170, 379, 382, 385, 388, 443, 470, 475, 481	\pdfxform	166, 168, 175
\endinput	51	\protected	132, 171, 173
\endlinechar	4, 10, 22	\ProvidesPackage	53
\errmessage	432		
H		R	
\hbox	95	\RangeCatcodeCheck	
			427, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466
		\RangeCatcodeInvalid	
			419, 447, 448, 449, 450
		\repeat	394, 406, 417, 425, 440
		\RequirePackage	
			55, 56, 57, 58, 59, 63, 145
		\reserved@a	100, 105, 114
		\reset@color	82, 94, 139
		\RestoreCatcodes	408, 411, 412, 467

S		<code>\TMP@EnsureCode</code> 27,
<code>\set@color</code> 81, 97, <u>132</u> , 149		34, 35, 36, 37, 38, 39, 40, 41,
<code>\set@page@color</code> 83		42, 43, 44, 45, 46, 47, 48, 49, 50
<code>\setattribute</code> 125		
<code>\setbox</code> 95	U	
<code>\setluatexattribute</code> 127	<code>\usepackage</code> 372, 480, 487, 488	
<code>\space</code> 166, 171, 433, 434, 442		
<code>\special</code> 108, 114	X	
T		<code>\x</code> 8, 20
<code>\Test</code> 446, 469		
<code>\the</code> 10, 11, 12,	Z	
13, 14, 15, 16, 17, 30, 413, 433, 434	<code>\z@</code> 95	