

The luacolor package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2007/12/12 v1.0

Abstract

Package luacolor implements color support based on L^AT_EX's node attributes.

Contents

1	Documentation	2
1.1	Introduction	2
1.2	Usage	2
2	Implementation	2
2.1	Catcodes and identification	2
2.2	Check for L ^A T _E X	3
2.3	Check for disabled colors	3
2.4	Find driver	3
2.5	Attribute setting	4
2.6	Whatsit insertion	4
2.7	Lua module	5
2.7.1	Driver detection	5
2.7.2	Color strings	6
2.7.3	Attribute register	6
2.7.4	Whatsit insertion	6
3	Test	7
3.1	Catcode checks for loading	7
3.2	Driver detection	8
3.3	Short test for plain-T _E X	9
4	Installation	9
4.1	Download	9
4.2	Bundle installation	9
4.3	Package installation	10
4.4	Refresh file name databases	10
4.5	Some details for the interested	10
5	History	11
	[2007/12/12 v1.0]	11
6	Index	11

1 Documentation

1.1 Introduction

This package uses a L^AT_EX'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 L^AT_EX 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.

2 Implementation

```
1 (*package)
```

2.1 Catcodes and identification

```
2 \begingroup
3   \catcode123 1 % {
4   \catcode125 2 % }
5   \def\x{\endgroup
6     \expandafter\edef\csname LuaCol@AtEnd\endcsname{%
7       \catcode35 \the\catcode35\relax
8       \catcode64 \the\catcode64\relax
9       \catcode123 \the\catcode123\relax
10      \catcode125 \the\catcode125\relax
11    }%
12  }%
13 \x
14 \catcode35 6 % #
15 \catcode64 11 % @
16 \catcode123 1 % {
17 \catcode125 2 % }
18 \def\TMP@EnsureCode#1#2{%
19   \edef\LuaCol@AtEnd{%
20     \LuaCol@AtEnd
```

```

21 \catcode#1 \the\catcode#1\relax
22 }%
23 \catcode#1 #2\relax
24 }
25 \TMP@EnsureCode{34}{12}% "
26 \TMP@EnsureCode{39}{12}% '
27 \TMP@EnsureCode{40}{12}% (
28 \TMP@EnsureCode{41}{12}% )
29 \TMP@EnsureCode{42}{12}% *
30 \TMP@EnsureCode{43}{12}% +
31 \TMP@EnsureCode{44}{12}% ,
32 \TMP@EnsureCode{45}{12}% -
33 \TMP@EnsureCode{46}{12}% .
34 \TMP@EnsureCode{47}{12}% /
35 \TMP@EnsureCode{58}{12}% :
36 \TMP@EnsureCode{60}{12}% <
37 \TMP@EnsureCode{61}{12}% =
38 \TMP@EnsureCode{62}{12}% >
39 \TMP@EnsureCode{95}{12}% _ (other!)
40 \TMP@EnsureCode{96}{12}% `
41 \edef\LuaCol@AtEnd{%
42 \LuaCol@AtEnd
43 \noexpand\endinput
44 }

Package identification.
45 \NeedsTeXFormat{LaTeX2e}
46 \ProvidesPackage{luacolor}%
47 [2007/12/12 v1.0 Coloring based on LuaTeX's node attributes (H0)]

```

2.2 Check for LuaTeX

Without L^AT_EX there is no point in using this package.

```

48 \RequirePackage{infwarerr}[2007/09/09]%
49 \RequirePackage{ifluatex}[2007/12/12]%
50 \RequirePackage{color}

51 \ifluatex
52 \RequirePackage{luatex}[2007/12/12]%
53 \else
54 \@PackageError{luacolor}{%
55 This package may only be run using LuaTeX%
56 }\@ehc
57 \expandafter\LuaCol@AtEnd
58 \fi

```

2.3 Check for disabled colors

```

59 \ifcolors@
60 \else
61 \@PackageWarningNoLine{luacolor}{%
62 Colors are disabled by option 'monochrome'%
63 }%
64 \expandafter\LuaCol@AtEnd
65 \fi

```

2.4 Find driver

```

66 \directlua0{%
67 require("oberdiek.luacolor")%
68 }
69 \RequirePackage{ifpdf}[2007/09/09]
70 \ifpdf
71 \else
72 \begingroup

```

```

73 \def\current@color{}%
74 \def\reset@color{}%
75 \setbox\z@=\hbox{%
76 \begingroup
77 \set@color
78 \endgroup
79 }%
80 \edef\reserved@a{%
81 \directlua0{%
82 oberdiek.luacolor.dvidetect()}%
83 }%
84 }%
85 \ifx\reserved@a\@empty
86 \@PackageError{luacolor}{%
87 DVI driver detection failed because of\MessageBreak
88 unrecognized color \string\special
89 }\@ehc
90 \endgroup
91 \expandafter\expandafter\expandafter\LuaCol@AtEnd
92 \else
93 \@PackageInfoNoLine{luacolor}{%
94 Type of color \string\special: \reserved@a
95 }%
96 \fi
97 \endgroup
98 \fi

```

2.5 Attribute setting

\LuaCol@Attribute

```

99 \newattribute\LuaCol@Attribute
100 \directlua0{%
101 oberdiek.luacolor.setattribute(\number\allocationnumber)%
102 }

```

\set@color

```

103 \protected\def\set@color{%
104 \setattribute\LuaCol@Attribute{%
105 \directlua0{%
106 oberdiek.luacolor.get("\luaescapestring{\current@color}")%
107 }%
108 }%
109 }

```

\reset@color

```

110 \def\reset@color{}

```

2.6 Whatsit insertion

\luacolorProcessBox

```

111 \def\luacolorProcessBox#1{%
112 \directlua0{%
113 oberdiek.luacolor.process(\number#1)%
114 }%
115 }

116 \RequirePackage{atbegshi}[2007/09/09]
117 \AtBeginShipout{%
118 \luacolorProcessBox\AtBeginShipoutBox
119 }

Set default color.
120 \set@color

```

```

121 \LuaCol@AtEnd
122 </package>

```

2.7 Lua module

```

123 <lua>

```

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

```

124 module("oberdiek.luacolor", package.seeall)

```

2.7.1 Driver detection

```

125 local ifpdf
126 if tonumber(tex.pdfoutput) > 0 then
127   ifpdf = true
128 else
129   ifpdf = false
130 end
131 local prefix
132 local prefixes = {
133   dvips    = "color ",
134   dvipdfm  = "pdf:sc ",
135   truetex  = "textcolor:",
136   pctexps  = "ps::",
137 }
138 local patterns = {
139   ["^color "]      = "dvips",
140   ["^pdf: *begincolor "] = "dvipdfm",
141   ["^pdf: *bcolor "]   = "dvipdfm",
142   ["^pdf: *bc "]       = "dvipdfm",
143   ["^pdf: *setcolor "] = "dvipdfm",
144   ["^pdf: *scolor "]   = "dvipdfm",
145   ["^pdf: *sc "]       = "dvipdfm",
146   ["^textcolor:"]      = "truetex",
147   ["^ps::"]            = "pctexps",
148 }
149 local function info(msg, term)
150   local target = "log"
151   if term then
152     target = "term and log"
153   end
154   texio.write_nl(target, "Package luacolor info: " .. msg .. ".")
155   texio.write_nl(target, "")
156 end
157 function dvidetect()
158   local v = tex.box[0]
159   assert(v.id == node.id("hlist"))
160   for v in node.traverse_id(node.id("whatsit"), v.list) do
161     if v and v.subtype == 3 then -- special
162       local data = v.data
163       for pattern, driver in pairs(patterns) do
164         if string.find(data, pattern) then
165           prefix = prefixes[driver]
166           tex.write(driver)
167           return
168         end
169       end
170       info("\\special{" .. data .. "}", true)
171       return
172     end
173   end
174   info("Missing \\special", true)

```

```
175 end
```

2.7.2 Color strings

```
176 local map = {
177   n = 0,
178 }
179 function get(color)
180   local n = map[color]
181   if not n then
182     n = map.n + 1
183     map.n = n
184     map[n] = color
185     map[color] = n
186   end
187   tex.write("'" .. n)
188 end
```

2.7.3 Attribute register

```
189 local attribute
190 function setattribute(attr)
191   attribute = attr
192 end
```

2.7.4 Whatsit insertion

```
193 function process(box)
194   local color = ""
195   local list = tex.getbox(box)
196   traverse(list, color)
197 end

198 local LIST = 1
199 local COLOR = 2
200 local type = {
201   [node.id("hlist")] = LIST,
202   [node.id("vlist")] = LIST,
203   [node.id("rule")] = COLOR,
204   [node.id("glyph")] = COLOR,
205   [node.id("disc")] = COLOR,
206 }
207 local subtype = {
208   [3] = COLOR, -- special
209   [8] = COLOR, -- pdf_literal
210 }
211 local mode = 2 -- luatex.pdf_literal.direct
212 local WHATSIT = node.id("whatsit")
213 local SPECIAL = 3
214 local PDFLITERAL = 8

215 function traverse(list, color)
216   if not list then
217     return color
218   end
219   if type[list.id] ~= LIST then
220     texio.write_nl("!!! Error: Wrong list type: " .. node.type(list.id))
221     return color
222   end
223   (debug)texio.write_nl("traverse: " .. node.type(list.id))
224   local head = list.list
225   for n in node.traverse(head) do
226     (debug)texio.write_nl(" node: " .. node.type(n.id))
227     local type = type[n.id]
228     if type == LIST then
229       color = traverse(n, color)
230     elseif type == COLOR
```

```

231         or (type == WHATSIT
232             and subtype[n.subtype]) then
233     local v = node.has_attribute(n, attribute)
234     if v then
235         local newColor = map[v]
236         if newColor ~= color then
237             color = newColor
238             local newNode
239             if ifpdf then
240                 newNode = node.new(WHATSIT, PDFLITERAL)
241                 newNode.mode = mode
242                 newNode.data = color
243             else
244                 newNode = node.new(WHATSIT, SPECIAL)
245                 newNode.data = prefix .. color
246             end
247             if head == n then
248                 newNode.next = head
249                 local old_prev = head.prev
250                 head.prev = newNode
251                 head = newNode
252                 head.prev = old_prev
253             else
254                 head = node.insert_before(head, n, newNode)
255             end
256         end
257     end
258 end
259 end
260 list.list = head
261 return color
262 end
263 </lua>

```

3 Test

```

264 <test1>
265 \documentclass{article}
266 \usepackage{color}
267 </test1>

```

3.1 Catcode checks for loading

```

268 <test1>
269 \catcode'\{=1 %
270 \catcode'\}=2 %
271 \catcode'\#=6 %
272 \catcode'\@=11 %
273 \expandafter\ifx\csname count@\endcsname\relax
274   \countdef\count@=255 %
275 \fi
276 \expandafter\ifx\csname @gobble\endcsname\relax
277   \long\def@gobble#1{}%
278 \fi
279 \expandafter\ifx\csname @firstofone\endcsname\relax
280   \long\def@firstofone#1{#1}%
281 \fi
282 \expandafter\ifx\csname loop\endcsname\relax
283   \expandafter@firstofone
284 \else
285   \expandafter@gobble
286 \fi
287 {%

```

```

288 \def\loop#1\repeat{%
289   \def\body{#1}%
290   \iterate
291 }%
292 \def\iterate{%
293   \body
294   \let\next\iterate
295   \else
296     \let\next\relax
297   \fi
298   \next
299 }%
300 \let\repeat=\fi
301 }%
302 \def\RestoreCatcodes{}
303 \count@=0 %
304 \loop
305   \edef\RestoreCatcodes{%
306     \RestoreCatcodes
307     \catcode\the\count@=\the\catcode\count@\relax
308   }%
309   \ifnum\count@<255 %
310     \advance\count@ 1 %
311   \repeat
312 }
313 \def\RangeCatcodeInvalid#1#2{%
314   \count@=#1\relax
315   \loop
316     \catcode\count@=15 %
317     \ifnum\count@<#2\relax
318       \advance\count@ 1 %
319     \repeat
320 }
321 \expandafter\ifx\csname LoadCommand\endcsname\relax
322   \def\LoadCommand{\input luacolor.sty\relax}%
323 \fi
324 \def\Test{%
325   \RangeCatcodeInvalid{0}{47}%
326   \RangeCatcodeInvalid{58}{64}%
327   \RangeCatcodeInvalid{91}{96}%
328   \RangeCatcodeInvalid{123}{255}%
329   \catcode'\@=12 %
330   \catcode'\=0 %
331   \catcode'\{=1 %
332   \catcode'\}=2 %
333   \catcode'\#=6 %
334   \catcode'\[=12 %
335   \catcode'\]=12 %
336   \catcode'\%=14 %
337   \catcode'\_ =10 %
338   \catcode\l3=5 %
339   \LoadCommand
340   \RestoreCatcodes
341 }
342 \Test
343 \csname @@end\endcsname
344 \end
345 </test1>

```

3.2 Driver detection

```

346 <*test2>
347 \NeedsTeXFormat{LaTeX2e}

```

```

348 \ifcsname driver\endcsname
349 \expandafter\PassOptionsToPackage\expandafter{\driver}{color}%
350 \pdfoutput=0 %
351 \fi
352 \documentclass{minimal}
353 \usepackage{luacolor}[2007/12/12]
354 \csname @@end\endcsname
355 \end
356 \test2}

357 <*test3>
358 \NeedsTeXFormat{LaTeX2e}

359 \documentclass{minimal}
360 \usepackage{luacolor}[2007/12/12]
361 \usepackage{qstest}
362 \IncludeTests{*}
363 \LogTests{log}{*}{*}
364 \makeatletter

365 \@@end
366 \test3}

```

3.3 Short test for plain-TeX

```

367 <*test4>
368 \input luacolor.sty\relax
369 \newluastate\TestLuaState
370 \newattribute\TestAttr
371 \setattribute\TestAttr{10}
372 \unsetattribute\TestAttr
373 \newcatcodetable\TestCTa
374 \begingroup
375 \SetCatcodeRange{'A'}{'Z'}{12}%
376 \endgroup
377 \BeginCatcodeRegime\__CT__LaTeX
378 \EndCatcodeRegime
379 \end
380 \test4}

```

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 TeX 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
```

¹<http://ftp.ctan.org/tex-archive/>

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
oberdiek.luacolor.lua → scripts/oberdiek/oberdiek.luacolor.lua
luacolor.lua      → scripts/oberdiek/luacolor.lua
luacolor.pdf      → doc/latex/oberdiek/luacolor.pdf
test/luacolor-test1.tex → doc/latex/oberdiek/test/luacolor-test1.tex
test/luacolor-test2.tex → doc/latex/oberdiek/test/luacolor-test2.tex
test/luacolor-test3.tex → doc/latex/oberdiek/test/luacolor-test3.tex
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

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 L^AT_EX. The `.dtx` chooses its action depending on the format:

plain-`TEX`: Run `docstrip` and extract the files.

L^AT_EX: Generate the documentation.

If you insist on using L^AT_EX for `docstrip` (really, `docstrip` does not need L^AT_EX), 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.

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; numbers in *roman* refer to the code lines where the entry is used.

Symbols	
\#	271, 333
\%	336
\@	272, 329
\@@end	365
\@PackageError	54, 86
\@PackageInfoNoLine	93
\@PackageWarningNoLine	61
\@ehc	56, 89
\@empty	85
\@firstofone	280, 283
\@gobble	277, 285
\[334
\]	170, 174, 330
\{	269, 331
\}	270, 332
\]	335
_	377
_	337
A	
\advance	310, 318
\allocationnumber	101
\AtBeginShipout	117
\AtBeginShipoutBox	118
B	
\BeginCatcodeRegime	377
\body	289, 293
C	
\catcode	3, 4, 7, 8, 9, 10, 14, 15, 16, 17, 21, 23, 269, 270, 271,
	272, 307, 316, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338
\count@	274, 303,
	307, 309, 310, 314, 316, 317, 318
\countdef	274
\csname	6,
	273, 276, 279, 282, 321, 343, 354
\current@color	73, 106
D	
\directlua	66, 81, 100, 105, 112
\documentclass	265, 352, 359
\driver	349
E	
\end	344, 355, 379
\EndCatcodeRegime	378
\endcsname	6, 273,
	276, 279, 282, 321, 343, 348, 354
\endinput	43
H	
\hbox	75
I	
\ifcolors@	59
\ifcsname	348
\ifluatex	51
\ifnum	309, 317
\ifpdf	70
\ifx	85, 273, 276, 279, 282, 321
\IncludeTests	362
\input	322, 368
\iterate	290, 292, 294

L	
<code>\LoadCommand</code>	322, 339
<code>\LogTests</code>	363
<code>\loop</code>	288, 304, 315
<code>\LuaCol@AtEnd</code>	
. . .	19, 20, 41, 42, 57, 64, 91, 121
<code>\LuaCol@Attribute</code>	99, 104
<code>\luacolorProcessBox</code>	2, 111, 118
<code>\luaescapestring</code>	106
M	
<code>\makeatletter</code>	364
<code>\MessageBreak</code>	87
N	
<code>\NeedsTeXFormat</code>	45, 347, 358
<code>\newattribute</code>	99, 370
<code>\newcatcodetable</code>	373
<code>\newluastate</code>	369
<code>\next</code>	294, 296, 298
<code>\number</code>	101, 113
P	
<code>\PassOptionsToPackage</code>	349
<code>\pdfoutput</code>	350
<code>\protected</code>	103
<code>\ProvidesPackage</code>	46
R	
<code>\RangeCatcodeInvalid</code>	
.	313, 325, 326, 327, 328
<code>\repeat</code>	288, 300, 311, 319
<code>\RequirePackage</code>	48, 49, 50, 52, 69, 116
<code>\reserved@a</code>	80, 85, 94
<code>\reset@color</code>	74, 110
<code>\RestoreCatcodes</code>	302, 305, 306, 340
S	
<code>\set@color</code>	77, 103, 120
<code>\setattribute</code>	104, 371
<code>\setbox</code>	75
<code>\SetCatcodeRange</code>	375
<code>\special</code>	88, 94
T	
<code>\Test</code>	324, 342
<code>\TestAttr</code>	370, 371, 372
<code>\TestCTa</code>	373
<code>\TestLuaState</code>	369
<code>\the</code>	7, 8, 9, 10, 21, 307
<code>\TMP@EnsureCode</code>	
.	18, 25, 26, 27, 28, 29, 30,
.	31, 32, 33, 34, 35, 36, 37, 38, 39, 40
U	
<code>\unsetattribute</code>	372
<code>\usepackage</code>	266, 353, 360, 361
X	
<code>\x</code>	5, 13
Z	
<code>\z@</code>	75