

# The luacolor package

Heiko Oberdiek  
<heiko.oberdiek at gmail.com>

2011/03/29 v1.4

## 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
<b>2</b>	<b>Implementation</b>	<b>2</b>
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	4
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
<b>3</b>	<b>Test</b>	<b>8</b>
3.1	Catcode checks for loading	8
3.2	Driver detection	10
<b>4</b>	<b>Installation</b>	<b>10</b>
4.1	Download	10
4.2	Bundle installation	11
4.3	Package installation	11
4.4	Refresh file name databases	11
4.5	Some details for the interested	11
<b>5</b>	<b>History</b>	<b>12</b>
	[2007/12/12 v1.0]	12
	[2009/04/10 v1.1]	12
	[2010/03/09 v1.2]	12
	[2010/12/13 v1.3]	12
	[2011/03/29 v1.4]	12
<b>6</b>	<b>Index</b>	<b>12</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.

## 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 [2011/03/29 v1.4 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}[2007/09/09]%
56 \RequirePackage{ifluatex}[2009/04/10]%
57 \RequirePackage{ifpdf}[2010/01/28]%
58 \RequirePackage{ltxcmds}[2010/04/26]%
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 \RequirePackage{ifpdf}[2007/09/09]
91 \ifpdf
92 \else
93   \begingroup
94     \def\current@color{}%
95     \def\reset@color{}%
96     \setbox\z@=\hbox{%
97       \begingroup
98         \set@color
99       \endgroup
100     }%
101     \edef\reserved@a{%
102       \LuaCol@directlua{%
103         oberdiek.luacolor.dvidetect()%
104       }%
105     }%
106     \ifx\reserved@a\@empty
107       \@PackageError{luacolor}{%
108         DVI driver detection failed because of\MessageBreak
109         unrecognized color \string\special
110       }{\@ehc
111       \endgroup
112       \expandafter\expandafter\expandafter\LuaCol@AtEnd
113     \else
114       \@PackageInfoNoLine{luacolor}{%
115         Type of color \string\special: \reserved@a
116       }%
117     \fi%
118   \endgroup
119 \fi
```

## 2.5 Attribute setting

\LuaCol@Attribute

```
120 \ltx@ifundefined{newluatexattribute}{%
121   \newattribute\LuaCol@Attribute
122 }{%
123   \newluatexattribute\LuaCol@Attribute
124 }
125 \ltx@ifundefined{setluatexattribute}{%
126   \let\LuaCol@setattribute\setattribute
127 }{%
128   \let\LuaCol@setattribute\setluatexattribute
129 }
130 \LuaCol@directlua{%
```

```

131 oberdiek.luacolor.setattribute(\number\allocationnumber)%
132 }

\set@color

133 \protected\def\set@color{%
134   \LuaCol@setattribute\LuaCol@Attribute{%
135     \LuaCol@directlua{%
136       oberdiek.luacolor.get("\luatexluaescapestring{\current@color}")%
137     }%
138   }%
139 }

\reset@color

140 \def\reset@color{}

```

## 2.6 Whatsit insertion

```

\luacolorProcessBox

141 \def\luacolorProcessBox#1{%
142   \LuaCol@directlua{%
143     oberdiek.luacolor.process(\number#1)%
144   }%
145 }

146 \RequirePackage{atbegshi}[2007/09/09]
147 \AtBeginShipout{%
148   \luacolorProcessBox\AtBeginShipoutBox
149 }

Set default color.

150 \set@color

```

## 2.7 \pdfxform support

```

151 \ifpdf
152   \ltx@ifundefined{pdfxform}{%
153     \ifnum\luatexversion>36 %
154       \directlua{%
155         tex.enableprimitives('',{%
156           'pdfxform','pdflastxform','pdfrefxform'%
157         })%
158       }%
159     \fi
160   }{}%
161   \ltx@ifundefined{protected}{%
162     \ifnum\luatexversion>36 %
163       \directlua{tex.enableprimitives('',{'protected'})}%
164     \fi
165   }{}%
166   \ltx@ifundefined{pdfxform}{%
167     \@PackageWarning{luacolor}{\string\pdfxform\space not found}%
168   }{%
169     \let\LuaCol@org@pdfxform\pdfxform
170     \begingroup\expandafter\expandafter\expandafter\endgroup
171     \expandafter\ifx\csname protected\endcsname\relax
172       \@PackageWarning{luacolor}{\string\protected\space not found}%
173     \else
174       \expandafter\protected
175     \fi
176     \def\pdfxform{%
177       \begingroup
178       \afterassignment\LuaCol@pdfxform

```

```

179     \count@=%
180 }%
181 \def\LuaCol@pdfxform{%
182     \luacolorProcessBox\count@
183     \LuaCol@org@pdfxform\count@
184 \endgroup
185 }%
186 }%
187 \fi

188 \LuaCol@AtEnd%
189 \end{package}

```

## 2.8 Lua module

```

190 (*lua)

```

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

```

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

```

### 2.8.1 Driver detection

```

192 local ifpdf
193 if tonumber(tex.pdfoutput) > 0 then
194     ifpdf = true
195 else
196     ifpdf = false
197 end
198 local prefix
199 local prefixes = {
200     dvips = "color ",
201     dvipdfm = "pdf:sc ",
202     truetex = "textcolor:",
203     pctexps = "ps::",
204 }
205 local patterns = {
206     ["^color "] = "dvips",
207     ["^pdf: *begincolor "] = "dvipdfm",
208     ["^pdf: *bcolor "] = "dvipdfm",
209     ["^pdf: *bc "] = "dvipdfm",
210     ["^pdf: *setcolor "] = "dvipdfm",
211     ["^pdf: *scolor "] = "dvipdfm",
212     ["^pdf: *sc "] = "dvipdfm",
213     ["^textcolor:"] = "truetex",
214     ["^ps::"] = "pctexps",
215 }
216 local function info(msg, term)
217     local target = "log"
218     if term then
219         target = "term and log"
220     end
221     texio.write_nl(target, "Package luacolor info: " .. msg .. ".")
222     texio.write_nl(target, "")
223 end
224 function dvidetect()
225     local v = tex.box[0]
226     assert(v.id == node.id("hlist"))
227     (!pre065) for v in node.traverse_id(node.id("whatsit"), v.head) do
228     (pre065) for v in node.traverse_id(node.id("whatsit"), v.list) do
229         if v and v.subtype == 3 then -- special
230             local data = v.data
231             for pattern, driver in pairs(patterns) do
232                 if string.find(data, pattern) then
233                     prefix = prefixes[driver]

```

```

234         tex.write(driver)
235     return
236 end
237 end
238     info("\\special{" .. data .. "}", true)
239     return
240 end
241 end
242     info("Missing \\special", true)
243 end

```

## 2.8.2 Color strings

```

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

```

## 2.8.3 Attribute register

```

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

```

## 2.8.4 Whatsit insertion

```

261 function process(box)
262     local color = ""
263     local list = tex.getbox(box)
264     traverse(list, color)
265 end
266 local LIST = 1
267 local COLOR = 2
268 local type = {
269     [node.id("hlist")] = LIST,
270     [node.id("vlist")] = LIST,
271     [node.id("rule")] = COLOR,
272     [node.id("glyph")] = COLOR,
273     [node.id("disc")] = COLOR,
274 }
275 local subtype = {
276     [3] = COLOR, -- special
277     [8] = COLOR, -- pdf_literal
278 }
279 local mode = 2 -- luatex.pdf_literal.direct
280 local WHATSIT = node.id("whatsit")
281 local SPECIAL = 3
282 local PDFLITERAL = 8
283 function traverse(list, color)
284     if not list then
285         return color
286     end
287     if type[list.id] ~= LIST then
288         texio.write_nl("!!! Error: Wrong list type: " .. node.type(list.id))
289         return color

```

```

290 end
291 (debug)texio.write_nl("traverse: " .. node.type(list.id))
292 (!pre065) local head = list.head
293 (pre065) local head = list.list
294 for n in node.traverse(head) do
295 (debug)texio.write_nl(" node: " .. node.type(n.id))
296 local type = type[n.id]
297 if type == LIST then
298 color = traverse(n, color)
299 elseif type == COLOR
300 or (type == WHATSIT
301 and subtype[n.subtype]) then
302 local v = node.has_attribute(n, attribute)
303 if v then
304 local newColor = map[v]
305 if newColor ~= color then
306 color = newColor
307 local newNode
308 if ifpdf then
309 newNode = node.new(WHATSIT, PDFLITERAL)
310 newNode.mode = mode
311 newNode.data = color
312 else
313 newNode = node.new(WHATSIT, SPECIAL)
314 newNode.data = prefix .. color
315 end
316 if head == n then
317 newNode.next = head
318 local old_prev = head.prev
319 head.prev = newNode
320 head = newNode
321 head.prev = old_prev
322 else
323 head = node.insert_before(head, n, newNode)
324 end
325 end
326 end
327 end
328 end
329 (!pre065) list.head = head
330 (pre065) list.list = head
331 return color
332 end
333 </lua>

```

### 3 Test

```

334 (*test1)
335 \documentclass{article}
336 \usepackage{color}
337 </test1>

```

#### 3.1 Catcode checks for loading

```

338 (*test1)
339 \catcode'\{=1 %
340 \catcode'\}=2 %
341 \catcode'\#=6 %
342 \catcode'\@=11 %
343 \expandafter\ifx\csname count@\endcsname\relax
344 \countdef\count@=255 %
345 \fi
346 \expandafter\ifx\csname @gobble\endcsname\relax

```

```

347 \long\def\@gobble#1{}%
348 \fi
349 \expandafter\ifx\csname @firstofone\endcsname\relax
350 \long\def\@firstofone#1{#1}%
351 \fi
352 \expandafter\ifx\csname loop\endcsname\relax
353 \expandafter\@firstofone
354 \else
355 \expandafter\@gobble
356 \fi
357 {%
358 \def\loop#1\repeat{%
359 \def\body{#1}%
360 \iterate
361 }%
362 \def\iterate{%
363 \body
364 \let\next\iterate
365 \else
366 \let\next\relax
367 \fi
368 \next
369 }%
370 \let\repeat=\fi
371 }%
372 \def\RestoreCatcodes{}
373 \count@=0 %
374 \loop
375 \edef\RestoreCatcodes{%
376 \RestoreCatcodes
377 \catcode\the\count@=\the\catcode\count@\relax
378 }%
379 \ifnum\count@<255 %
380 \advance\count@ 1 %
381 \repeat
382 }
383 \def\RangeCatcodeInvalid#1#2{%
384 \count@=#1\relax
385 \loop
386 \catcode\count@=15 %
387 \ifnum\count@<#2\relax
388 \advance\count@ 1 %
389 \repeat
390 }
391 \def\RangeCatcodeCheck#1#2#3{%
392 \count@=#1\relax
393 \loop
394 \ifnum#3=\catcode\count@
395 \else
396 \errmessage{%
397 Character \the\count@\space
398 with wrong catcode \the\catcode\count@\space
399 instead of \number#3%
400 }%
401 \fi
402 \ifnum\count@<#2\relax
403 \advance\count@ 1 %
404 \repeat
405 }
406 \def\space{ }
407 \expandafter\ifx\csname LoadCommand\endcsname\relax
408 \def\LoadCommand{\input luacolor.sty\relax}%

```

```

409 \fi
410 \def\Test{%
411   \RangeCatcodeInvalid{0}{47}%
412   \RangeCatcodeInvalid{58}{64}%
413   \RangeCatcodeInvalid{91}{96}%
414   \RangeCatcodeInvalid{123}{255}%
415   \catcode'\@=12 %
416   \catcode'\=0 %
417   \catcode'\%=14 %
418   \LoadCommand
419   \RangeCatcodeCheck{0}{36}{15}%
420   \RangeCatcodeCheck{37}{37}{14}%
421   \RangeCatcodeCheck{38}{47}{15}%
422   \RangeCatcodeCheck{48}{57}{12}%
423   \RangeCatcodeCheck{58}{63}{15}%
424   \RangeCatcodeCheck{64}{64}{12}%
425   \RangeCatcodeCheck{65}{90}{11}%
426   \RangeCatcodeCheck{91}{91}{15}%
427   \RangeCatcodeCheck{92}{92}{0}%
428   \RangeCatcodeCheck{93}{96}{15}%
429   \RangeCatcodeCheck{97}{122}{11}%
430   \RangeCatcodeCheck{123}{255}{15}%
431   \RestoreCatcodes
432 }
433 \Test
434 \csname @@end\endcsname
435 \end
436 </test1>

```

## 3.2 Driver detection

```

437 (*test2)
438 \NeedsTeXFormat{LaTeX2e}
439 \ifcsname driver\endcsname
440   \expandafter\PassOptionsToPackage\expandafter{\driver}{color}%
441   \pdfoutput=0 %
442 \fi
443 \documentclass{minimal}
444 \usepackage{luacolor}[2011/03/29]
445 \csname @@end\endcsname
446 \end
447 </test2>
448 (*test3)
449 \NeedsTeXFormat{LaTeX2e}
450 \documentclass{minimal}
451 \usepackage{luacolor}[2011/03/29]
452 \usepackage{qstest}
453 \IncludeTests{*}
454 \LogTests{log}{*}{*}
455 \makeatletter
456 \@@end
457 </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.

---

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

**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
```

**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 T<sub>E</sub>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>	→ <code>tex/latex/oberdiek/luacolor.sty</code>
<code>oberdiek.luacolor.lua</code>	→ <code>scripts/oberdiek/oberdiek.luacolor.lua</code>
<code>luacolor.lua</code>	→ <code>scripts/oberdiek/luacolor.lua</code>
<code>oberdiek.luacolor-pre065.lua</code>	→ <code>scripts/oberdiek/oberdiek.luacolor-pre065.lua</code>
<code>luacolor-pre065.lua</code>	→ <code>scripts/oberdiek/luacolor-pre065.lua</code>
<code>luacolor.pdf</code>	→ <code>doc/latex/oberdiek/luacolor.pdf</code>
<code>test/luacolor-test1.tex</code>	→ <code>doc/latex/oberdiek/test/luacolor-test1.tex</code>
<code>test/luacolor-test2.tex</code>	→ <code>doc/latex/oberdiek/test/luacolor-test2.tex</code>
<code>test/luacolor-test3.tex</code>	→ <code>doc/latex/oberdiek/test/luacolor-test3.tex</code>
<code>luacolor.dtx</code>	→ <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 T<sub>E</sub>X distribution (teT<sub>E</sub>X, mikT<sub>E</sub>X, ...) relies on file name databases, you must refresh these. For example, teT<sub>E</sub>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 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 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).

## 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

\# ..... 341

<code>\%</code> .....	417	<code>\input</code> .....	408
<code>\@</code> .....	342, 415	<code>\iterate</code> .....	360, 362, 364
<code>\@@end</code> .....	456		
<code>\@PackageError</code> .....	66, 107		
<code>\@PackageInfoNoLine</code> .....	114		
<code>\@PackageWarning</code> .....	167, 172		
<code>\@PackageWarningNoLine</code> .....	78		
<code>\@ehc</code> .....	68, 110		
<code>\@empty</code> .....	106		
<code>\@firstofone</code> .....	350, 353		
<code>\@gobble</code> .....	347, 355		
<code>\@</code> .....	238, 242, 416		
<code>\{</code> .....	339		
<code>\}</code> .....	340		
	<b>A</b>		
<code>\advance</code> .....	380, 388, 403		
<code>\afterassignment</code> .....	178		
<code>\allocationnumber</code> .....	131		
<code>\AtBeginShipout</code> .....	147		
<code>\AtBeginShipoutBox</code> .....	148		
	<b>B</b>		
<code>\body</code> .....	359, 363		
	<b>C</b>		
<code>\catcode</code> 2, 3, 5, 6, 7, 11, 12, 13, 14, 15, 16, 17, 20, 21, 23, 24, 25, 26, 30, 32, 339, 340, 341, 342, 377, 386, 394, 398, 415, 416, 417			
<code>\count@</code> ... 179, 182, 183, 344, 373, 377, 379, 380, 384, 386, 387, 388, 392, 394, 397, 398, 402, 403			
<code>\countdef</code> .....	344		
<code>\csname</code> .....	9, 171, 343, 346, 349, 352, 407, 434, 445		
<code>\current@color</code> .....	94, 136		
	<b>D</b>		
<code>\define@color</code> .....	84		
<code>\directlua</code> .....	72, 74, 154, 163		
<code>\documentclass</code> .....	335, 443, 450		
<code>\driver</code> .....	440		
	<b>E</b>		
<code>\end</code> .....	435, 446		
<code>\endcsname</code> .....	9, 171, 343, 346, 349, 352, 407, 434, 439, 445		
<code>\endinput</code> .....	51		
<code>\endlinechar</code> .....	4, 10, 22		
<code>\errmessage</code> .....	396		
	<b>H</b>		
<code>\hbox</code> .....	96		
	<b>I</b>		
<code>\ifcolors@</code> .....	76		
<code>\ifcsname</code> .....	439		
<code>\ifluatex</code> .....	60		
<code>\ifnum</code> 71, 88, 153, 162, 379, 387, 394, 402			
<code>\ifpdf</code> .....	91, 151		
<code>\ifx</code> .. 106, 171, 343, 346, 349, 352, 407			
<code>\IncludeTests</code> .....	453		
	<b>L</b>		
<code>\LoadCommand</code> .....	408, 418		
<code>\LogTests</code> .....	454		
<code>\loop</code> .....	358, 374, 385, 393		
<code>\ltx@ifpackageloaded</code> .....	61		
<code>\ltx@ifundefined</code> 120, 125, 152, 161, 166			
<code>\LuaCol@AtEnd</code> 28, 29, 51, 69, 85, 112, 188			
<code>\LuaCol@Attribute</code> .....	120, 134		
<code>\LuaCol@directlua</code> .....	71, 87, 102, 130, 135, 142		
<code>\LuaCol@org@pdfxform</code> .....	169, 183		
<code>\LuaCol@pdfxform</code> .....	178, 181		
<code>\LuaCol@setattribute</code> ..	126, 128, 134		
<code>\luacolorProcessBox</code> ..	2, 141, 148, 182		
<code>\luatexluaescapestring</code> .....	136		
<code>\luatexversion</code> .....	71, 88, 153, 162		
	<b>M</b>		
<code>\makeatletter</code> .....	455		
<code>\MessageBreak</code> .....	108		
	<b>N</b>		
<code>\NeedsTeXFormat</code> .....	52, 438, 449		
<code>\newattribute</code> .....	121		
<code>\newluatexattribute</code> .....	123		
<code>\next</code> .....	364, 366, 368		
<code>\number</code> .....	131, 143, 399		
	<b>P</b>		
<code>\PassOptionsToPackage</code> .....	440		
<code>\pdfoutput</code> .....	441		
<code>\pdfxform</code> .....	167, 169, 176		
<code>\protected</code> .....	133, 172, 174		
<code>\ProvidesPackage</code> .....	53		
	<b>R</b>		
<code>\RangeCatcodeCheck</code> .....	391, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430		
<code>\RangeCatcodeInvalid</code> .....	383, 411, 412, 413, 414		
<code>\repeat</code> .....	358, 370, 381, 389, 404		
<code>\RequirePackage</code> .....	55, 56, 57, 58, 59, 63, 90, 146		
<code>\reserved@a</code> .....	101, 106, 115		
<code>\reset@color</code> .....	82, 95, 140		
<code>\RestoreCatcodes</code> ..	372, 375, 376, 431		
	<b>S</b>		
<code>\set@color</code> .....	81, 98, 133, 150		
<code>\set@page@color</code> .....	83		
<code>\setattribute</code> .....	126		
<code>\setbox</code> .....	96		
<code>\setluatexattribute</code> .....	128		
<code>\space</code> .....	167, 172, 397, 398, 406		
<code>\special</code> .....	109, 115		
	<b>T</b>		
<code>\Test</code> .....	410, 433		
<code>\the</code> .....	10, 11, 12, 13, 14, 15, 16, 17, 30, 377, 397, 398		

<code>\TMP@EnsureCode</code>	.....	27,		<b>X</b>
34, 35, 36, 37, 38, 39, 40, 41,			<code>\x</code>	..... 8, 20
42, 43, 44, 45, 46, 47, 48, 49, 50				
				<b>Z</b>
<b>U</b>				
<code>\usepackage</code>	.....	336, 444, 451, 452	<code>\z@</code>	..... 96