The luacolor package

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

2018/11/22 v1.11

Abstract

Package luacolor implements color support based on Lua\TeX{}’s node attributes.

Contents

1 Documentation 2
  1.1 Introduction ........................................... 2
  1.2 Usage ................................................ 2
  1.3 Limitations ......................................... 2

2 Implementation 3
  2.1 Catcodes and identification ............................. 3
  2.2 Check for Lua\TeX{} ..................................... 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

3 Test 10
  3.1 Catcode checks for loading .............................. 10
  3.2 Driver detection ...................................... 12

4 Installation 12
  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

5 Catalogue 14

*Please report any issues at https://github.com/ho-tex/oberdiek/issues
1 Documentation

1.1 Introduction

This package uses a Lua\TeX{}'s attribute register 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 whatsis for color.

At last the attribute annotations of the nodes in the output box must be analyzed to insert the necessary color whatsis. Currently Lua\TeX{} 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

2.1 Catcodes and identification

\begin{verbatim}
\begingroup\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode123=1 % {
\catcode125=2 % }
\catcode64=11 % @
\def\x\endgroup
\expandafter\edef\csname LuaCol@AtEnd\endcsname{%
\endlinechar=\the\endlinechar\relax
\catcode13=\the\catcode13\relax
\catcode32=\the\catcode32\relax
\catcode35=\the\catcode35\relax
\catcode61=\the\catcode61\relax
\catcode64=\the\catcode64\relax
\catcode123=\the\catcode123\relax
\catcode125=\the\catcode125\relax
}%
\x\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode32=6 % 
\catcode123=1 % {
\catcode125=2 % }
\def\TMP@EnsureCode#1#2{%
\edef\LuaCol@AtEnd{\LuaCol@AtEnd
\catcode#1=\the\catcode#1\relax
}\catcode#1=#2\relax
}%
\TMP@EnsureCode{34}{12}%;
\TMP@EnsureCode{39}{12}%;
\TMP@EnsureCode{40}{12}%;
\TMP@EnsureCode{41}{12}%;
\TMP@EnsureCode{42}{12}%;
\TMP@EnsureCode{43}{12}%;
\TMP@EnsureCode{44}{12}%;
\TMP@EnsureCode{45}{12}%;
\TMP@EnsureCode{46}{12}%;
\TMP@EnsureCode{47}{12}%;
\TMP@EnsureCode{58}{12}%;
\TMP@EnsureCode{60}{12}%;
\TMP@EnsureCode{62}{12}%;
\TMP@EnsureCode{91}{12}%;
\TMP@EnsureCode{93}{12}%;
\TMP@EnsureCode{95}{12}%_ (other!)
\edef\LuaCol@AtEnd{\LuaCol@AtEnd\noexpand\endinput}
\end{verbatim}

Package identification.
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{luacolor}
[2018/11/22 v1.11 Color support via LuaTeX's attributes (HO)]

\section{Check for Lua\TeX X}
Without Lua\TeX X there is no point in using this package.
\RequirePackage{infwarerr}[2010/04/08]%
\RequirePackage{ifluatex}[2010/03/01]%
\RequirePackage{ifpdf}[2011/01/30]%
\RequirePackage{ltxcmds}[2011/04/18]%
\RequirePackage{color}
require \texttt{ltluatex} rather than \texttt{luatex} package support for Lua\TeX X allocations.
\ifluatex
\if\newattribute\@undefined
\RequirePackage{ltluatex}\
\fi
\else
\PackageError{luacolor}{This package may only be run using Lua\TeX X}{@ehc}
\expandafter\LuaCol@AtEnd
\fi
\let\LuaCol@directlua\directlua

\section{Check for disabled colors}
\ifcolors@
\else
\PackageWarningNoLine{luacolor}{Colors are disabled by option `monochrome'}
\def\set@color{}
\def\reset@color{}
\def\set@page@color{}
\def\define@color#1#2{}
\expandafter\LuaCol@AtEnd
\fi

\section{Load module and check version}
\LuaCol@directlua{}
require('luacolor')
\begingroup
\edef\x{\LuaCol@directlua{tex.write("2018/11/22 v1.11")}}
\edef\y{}\LuaCol@directlua{}
\edef\y{\LuaCol@directlua{if oberdiek.luacolor.getversion then oberdiek.luacolor.getversion() end}}
\ifx\x\y
\else
\PackageError{luacolor}{Wrong version of lua module.\MessageBreak Package version: \x\MessageBreak Lua module: \y}{@ehc}
\fi

2.5 Find driver

\begin{verbatim}
2.6 Attribute setting

\LuaCol@Attribute
\set@color
\reset@color
\end{verbatim}
2.7 Whatst insertion

\def\luacolorProcessBox#1{%  
\LuaCol@directlua{%  
oberdiek.luacolor.process(\number#1)%  
}%  
}%  
\RequirePackage{atbegshi}[2011/01/30]  
\AtBeginShipout{%  
\luacolorProcessBox\AtBeginShipoutBox%  
}%  
Set default color.
\set@color

2.8 \pdfform support

\ifpdf  
\if\pdfform@undefined  
\let\pdfform\saveboxresource  
\fi%  
\ltx@IfUndefined{pdfxform}{%  
\directlua{\text-enableprimitives('',{\pdfform,\pdflastxform,\pdfformref}'\})}%  
\}{%  
\ltx@IfUndefined{protected}{%  
\directlua{\text-enableprimitives('',{'protected'})}%  
\}{%  
\@PackageWarning{luacolor}{\string\pdfxform\space not found}%  
\}%  
\let\LuaCol@org@pdfxform\pdfform  
\begingroup\expandafter\expandafter\expandafter\endgroup%  
\if\protected\relax%  
\@PackageWarning{luacolor}{\string\protected\space not found}%  
\else%  
\protected%  
\def\pdfxform{%  
\begingroup%  
\afterassignment\LuaCol@pdfxform  
\count@=%  
\_lua@directlua{\luacolorProcessBox\count@\LuaCol@org@pdfxform\count@}%  
\endgroup%  
\@PackageWarning{luacolor}{\string\pdfxform\space not found}%  
\endgroup
\expandafter\protected%  
\fi%  
\directlua{\text-enableprimitives('',{'protected'})}%  
\}{%  
\pdfform%  
\begingroup\expandafter\expandafter\expandafter\endgroup%  
\expandafter\ifx\csname protected\endcsname\relax%  
\@PackageWarning{luacolor}{\string\protected\space not found}%  
\else%  
\protected%  
\directlua{\text-enableprimitives('',{'protected'})}%  
\}{%  
\pdfform%  
\begingroup\expandafter\expandafter\expandafter\endgroup%  
\expandafter\ifx\csname protected\endcsname\relax%  
\@PackageWarning{luacolor}{\string\protected\space not found}%  
\else%  
\protected%  
\directlua{\text-enableprimitives('',{'protected'})}%  
\}{%  
\pdfform%  
\begingroup\expandafter\expandafter\expandafter\endgroup%  
\expandafter\ifx\csname protected\endcsname\relax%  
\@PackageWarning{luacolor}{\string\protected\space not found}%  
\else%  
\protected%  
\directlua{\text-enableprimitives('',{'protected'})}%  
\}{%  
\endgroup
\fi%  
\endgroup
\fi

2.9 Lua module

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: *color "] = "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("whatst"), 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

```lua
local map = {
    n = 0,
}

function get(color)
    tex.write("" .. getvalue(color))
end

function getvalue(color)
    local n = map[color]
    if not n then
        n = map.n + 1
        map.n = n
        map[n] = color
        map[color] = n
    end
    return n
end
```

2.9.3 Attribute register

```lua
function setattribute(attr)
    attribute = attr
end

function getattribute()
    return attribute
end
```

2.9.4 Whatsit insertion

```lua
local LIST = 1
local LIST_LEADERS = 2
local COLOR = 3
local RULE = node.id("rule")
local node_types = {
    [node.id("hlist")] = LIST,
    [node.id("vlist")] = LIST,
    [node.id("rule")] = COLOR,
    [node.id("glyph")] = COLOR,
    [node.id("disc")] = COLOR,
    [node.id("whatsit")]:={
        [node.subtype("special")] = COLOR,
        [node.subtype("pdf_literal")] = COLOR,
        [node.subtype("pdf_save")] = COLOR,
        [node.subtype("pdf_restore")] = COLOR, -- probably not needed
        -- TODO (DPC) [node.subtype("pdf_refximage")] = COLOR,
        --
    },
    [node.id("glue")]=function(n)
        if n.subtype >= 100 then -- leaders
            if n.leader.id == RULE then
                return COLOR
            else
                return LIST_LEADERS
            end
        end
        return n
    end
```
get_type()

```
local function get_type(n)
    local ret = node_types[n.id]
    if type(ret) == 'table' then
        ret = ret[n.subtype]
    end
    if type(ret) == 'function' then
        ret = ret(n)
    end
    return ret
end
```

```
mode = 2 -- luatex.pdfliteral.direct
local WHATSIT = node.id("whatsit")
local SPECIAL = node.subtype("special")
local PDFLITERAL = node.subtype("pdf_literal")
local DRY_FALSE = false
local DRY_TRUE = true
```

traverse()

```
local function traverse(list, color, dry)
    if not list then
        return color
    end
    if get_type(list) ~= LIST then
        texio.write_nl("!!! Error: Wrong list type: " .. node.type(list.id))
        return color
    end
    local head = list.head
    for n in node.traverse(head) do
        local t = get_type(n)
        if t == LIST then
            color = traverse(n, color, dry)
        elseif t == LIST_LEADERS then
            local color_after = traverse(n.leader, color, DRY_TRUE)
            if color == color_after then
                traverse(n.leader, color, DRY_FALSE or dry)
            else
                traverse(n.leader, '', DRY_FALSE or dry)
            end
        elseif t == COLOR then
            local v = node.has_attribute(n, attribute)
            if v then
                local newColor = map[v]
                if newColor ~= color then
                    color = newColor
                    if ifpdf then
                        newNode = node.new(WHATSIT, PDFLITERAL)
                        newNode.mode = mode
                        newNode.data = color
                    end
                end
            end
        end
    end
    texio.write_nl("traverse: " .. node.type(list.id))
    return color
end
```
else
    newNode = node.new(WHATSIT, SPECIAL)
    newNode.data = prefix .. color
end
head = node.insert_before(head, n, newNode)
end
end
end
end
head = node.insert_before(head, n, newNode)
end
end
end
end
list.head = head
return color
end
process()
function process(box)
    local color ="
    local list = tex.getbox(box)
    traverse(list, color, DRY_FALSE)
    end
end
})(/lua)

3 Test

\documentclass{article}
\usepackage{color}
\})(/test1)

3.1 Catcode checks for loading
\})(/test1)
\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`@=11 %
\expandafter\ifx\csname count@\endcsname\relax
\countdef\count@=255 %
\fi
\expandafter\ifx\csname @gobble\endcsname\relax
\long\def\@gobble#1{}% 
\fi
\expandafter\ifx\csname @firstofone\endcsname\relax
\long\def\@firstofone#1{#1}% 
\fi
\expandafter\ifx\csname loop\endcsname\relax
\else
\expandafter\@gobble
\fi
(\% 
\def\loop#1\repeat(\%
\def\body(#1)% 
\iterate 
)%
\def\iterate(\%
\body 
\let\next\iterate
\else 
\let\next\relax
\fi
\next
\let\repeat=\fi\%
\def\RestoreCatcodes{}
\count@=0 \%
\loop
\edef\RestoreCatcodes{\RestoreCatcodes\catcode\the\count@=\the\catcode\count@\relax\}%
\ifnum\count@<255 \%
\advance\count@ 1 \%
\repeat
\def\RangeCatcodeInvalid#1#2{\count@=#1\relax\loop\catcode\count@=15 \%
\ifnum\count@<#2\relax\repeat\}
\def\RangeCatcodeCheck#1#2#3{\count@=#1\relax\loop\ifnum#3=\catcode\count@\else\errmessage{Character \the\count@ with wrong catcode \the\catcode\count@ instead of \number#3}\fi\ifnum\count@<#2\relax\repeat\}
def\space{ }\expandafter\ifx\csname LoadCommand\endcsname\relax\def\LoadCommand{\input luacolor.sty}\relax\%
\fi\def\Test{%\RangeCatcodeInvalid{0}{47}%\RangeCatcodeInvalid{58}{64}%\RangeCatcodeInvalid{91}{96}%\RangeCatcodeInvalid{123}{255}%\catcode`@=12 \%\catcode`\%=0 \%\catcode`\%=14 \%
\LoadCommand\RangeCatcodeCheck{0}{36}{15}\%
\RangeCatcodeCheck{37}{37}{14}\%
\RangeCatcodeCheck{48}{47}{15}\%
\RangeCatcodeCheck{58}{63}{15}\%
\RangeCatcodeCheck{64}{64}{12}\%
\RangeCatcodeCheck{65}{90}{11}\%
\RangeCatcodeCheck{91}{91}{15}\%
\RangeCatcodeCheck{92}{92}{0}\%
\RangeCatcodeCheck{93}{96}{15}\%
\RangeCatcodeCheck{97}{122}{11}%
3.2 Driver detection

4.1 Download

Package. This package is available on CTAN\footnote{http://ctan.org/pkg/luacolor}:


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/macrolatex/oberdiek.tds.zip

TDS refers to the standard “A Directory Structure for \TeX Files” (CTAN:tds/tds.pdf). Directories with \texttt{texmf} in their name are usually organized this way.

4.2 Bundle installation

Unpacking. Unpack the \texttt{oberdiek.tds.zip} in the TDS tree (also known as \texttt{texmf} tree) of your choice. Example (linux):

\begin{verbatim}
    unzip oberdiek.tds.zip -d ~/texmf
\end{verbatim}

\footnote{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):

```bash
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 (teTeX, miktex, ...) relies on file name databases, you must refresh these. For example, teTeX users run texhash or mktexlsr.

4.5 Some details for the interested

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

plain TeX: Run docstrip and extract the files.

\LaTeX: Generate the documentation.

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

```
lualatex \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 pdf\LaTeX:

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

13
5 Catalogue

The following XML file can be used as source for the \TeX{} Catalogue. The elements \texttt{caption} and \texttt{description} are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is \texttt{luacolor.xml}.

\begin{verbatim}
<catalogue>
<?xml version='1.0' encoding='us-ascii'?>
<!DOCTYPE entry SYSTEM 'catalogue.dtd'>
<entry datestamp='$Date$' modifier='$Author$' id='luacolor'>
  <name>luacolor</name>
  <caption>Color support based on LuaTeX's node attributes.</caption>
  <authorref id='auth:oberdiek'/>
  <copyright owner='Heiko Oberdiek' year='2007,2009-2011'/>
  <license type='lppl1.3'/>
  <version number='1.11'/>
  <description>
    This package implements color support based on LuaTeX's node attributes.
  </description>
  <documentation details='Package documentation'
    href='ctan:/macros/latex/contrib/oberdiek/luacolor.pdf'/>
  <ctan file='true' path='/macros/latex/contrib/oberdiek/luacolor.dtx'/>
  <miktex location='oberdiek'/>
  <texlive location='oberdiek'/>
  <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip'/>
</entry>
</catalogue>
\end{verbatim}

6 History

[2007/12/12 v1.0]
- First public version.

[2009/04/10 v1.1]
- Fixes for changed syntax of \texttt{\directlua} in Lua\TeX{} 0.36.

[2010/03/09 v1.2]
- Adaptation for package \texttt{lualatex} 2010/03/09 v0.4.

[2010/12/13 v1.3]
- Support for \texttt{\pdfxform} added.
- Loaded package \texttt{lualatex-base-attrib} recognized.
- Update for Lua\TeX{}: ‘list’ fields renamed to ‘head’ in v0.65.0.

[2011/03/29 v1.4]
- Avoid whatsis insertion if option \texttt{monochrome} is used (thanks Manuel Pégourié-Gonnard).
[2011/04/22 v1.5]

- Bug fix by Manuel Pégourié-Gonnard: A typo prevented the detection of whatsitss 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 whatsitss 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.

<table>
<thead>
<tr>
<th>Symbols</th>
<th>%</th>
<th>464</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>388</td>
<td>&amp;</td>
</tr>
<tr>
<td>Command</td>
<td>Page(s)</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>\setluatexattribute</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td>\space</td>
<td>178, 183, 444, 445, 453</td>
<td></td>
</tr>
<tr>
<td>\special</td>
<td>121, 127</td>
<td></td>
</tr>
<tr>
<td>\Test</td>
<td>457, 480</td>
<td></td>
</tr>
<tr>
<td>\the</td>
<td>10, 11, 12, 13, 14, 15, 16, 17, 30, 424, 444, 445</td>
<td></td>
</tr>
<tr>
<td>\TMP@EnsureCode</td>
<td>27, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50</td>
<td></td>
</tr>
<tr>
<td>\traverse()</td>
<td>324</td>
<td></td>
</tr>
<tr>
<td>\usepackage</td>
<td>383, 491, 498, 499</td>
<td></td>
</tr>
<tr>
<td>\x</td>
<td>8, 20, 86, 94, 98</td>
<td></td>
</tr>
<tr>
<td>\y</td>
<td>87, 94, 99</td>
<td></td>
</tr>
<tr>
<td>\z@</td>
<td>108</td>
<td></td>
</tr>
</tbody>
</table>