

# The `luatex` package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2007/12/12 v0.1

## Abstract

This package manages the new and extended features and resources that `LUATEX` provides. Examples are attributes and catcode tables.

## Contents

<b>1</b>	<b>Documentation</b>	<b>2</b>
1.1	Introduction	2
1.1.1	<code>L<sub>A</sub>T<sub>E</sub>X</code>	2
1.1.2	plain- <code>T<sub>E</sub>X</code>	2
1.2	Register allocation	3
1.2.1	Register with 16 bit	3
1.2.2	Insertions	3
1.3	Lua states	3
1.4	Attributes	3
1.5	Catcode tables	4
1.5.1	Interface proposal	4
1.6	Lua module loading	5
1.6.1	Package <code>luatex-loader</code>	6
<b>2</b>	<b>Implementation</b>	<b>6</b>
2.1	Reload check and package identification	6
2.2	Catcodes	7
2.3	Check for <code>LUA<sub>T</sub>EX</code>	8
2.4	Inherit support for $\epsilon$ - <code>T<sub>E</sub>X</code>	8
2.5	Adaption of $\epsilon$ - <code>T<sub>E</sub>X</code> 's register allocation	8
2.6	plain- <code>T<sub>E</sub>X</code> compatibility	9
2.7	Lua states	11
2.8	Attributes	11
2.8.1	Allocation	11
2.8.2	Interface	11
2.9	Catcode tables	12
2.9.1	Allocation	12
2.9.2	<code>\SetCatcodeRange</code>	12
2.9.3	Predefined catcode tables	13
2.9.4	Number stack	13
2.9.5	Catcode regime macros	14
2.10	Lua module loader	14
2.11	Lua script	16

<b>3</b>	<b>Test</b>	<b>16</b>
3.1	Catcode checks for loading	16
3.2	Catcode tables	18
3.2.1	Predefined catcode tables	18
3.2.2	Catcode table number stack	19
3.2.3	Catcode table stack	19
3.2.4	Catcode regime macros	20
3.3	Attribute allocation	20
3.4	Lua states	20
3.5	Short test for plain- $\text{\TeX}$	20
<b>4</b>	<b>Installation</b>	<b>21</b>
4.1	Download	21
4.2	Bundle installation	21
4.3	Package installation	21
4.4	Refresh file name databases	22
4.5	Some details for the interested	22
<b>5</b>	<b>History</b>	<b>22</b>
	[2007/12/12 v0.1]	22
<b>6</b>	<b>Index</b>	<b>23</b>

# 1 Documentation

## 1.1 Introduction

$\text{\TeX}$  provides global resources such as registers. But it does not provide an interface for managing these resources. For example, two packages want to use a counter register. If they take the same register number, then the use of both packages will conflict and they cannot be used together. Therefore formats such as plain- $\text{\TeX}$  or  $\text{\LaTeX}$  implement an allocation scheme for registers. A package reserves with `\newcount` an unused register number for its own exclusive use.

Nowadays  $\text{\TeX}$  is not alone anymore:  $\varepsilon\text{-}\text{\TeX}$ , pdf $\text{\TeX}$  and other compilers for  $\text{\TeX}$  are developed that extend and add new features and resources.

Now  $\text{\LaTeX}$  has reached beta state. It inherits most of pdf $\text{\TeX}$ 's features including  $\varepsilon\text{-}\text{\TeX}$ . Also it implements new concepts such as attributes or catcode tables.

### 1.1.1 $\text{\LaTeX}$

$\text{\LaTeX} 2_{\varepsilon}$  is frozen and therefore refuses to even notice the new  $\text{\TeX}$  variants. Not even the old  $\varepsilon\text{-}\text{\TeX}$  is supported by its kernel. At least there is a third party package `etex` that manages the new  $\varepsilon\text{-}\text{\TeX}$  resources.

This package tries to do the same for  $\text{\LaTeX}$  and starts to support at least a few of the new features.

### 1.1.2 plain- $\text{\TeX}$

$\text{\LaTeX}$  has inherited its resource handling from plain- $\text{\TeX}$ . The interface is basically the same: `\newcount`, ... Therefore this package tries to follow this tradition by providing compatibility to plain- $\text{\TeX}$ . It can be loaded with plain- $\text{\TeX}$  and defines at least some of the features that this packages provides for  $\text{\LaTeX}$ .

## 1.2 Register allocation

### 1.2.1 Register with 16 bit

Because L<sup>A</sup>T<sub>E</sub>X is a super set of  $\epsilon$ -T<sub>E</sub>X regarding registers, the register allocation scheme should not conflict with package `etex`. Therefore this package is loaded to inherit its allocation scheme. The only change is currently that the limit is increased to 65536 registers for the following register classes:

- `count`
- `dimen`
- `skip`
- `muskip`
- `marks`
- `toks`
- `box`

This affects the number of global and local registers. Because it is done in a package and not in the kernel, it is possible that someone loads package `etex` before uses the local allocation variants. This will prevent the extension for this register class. If more registers are needed, just load package `luatex` earlier.

### 1.2.2 Insertions

Insertions need four registers `\count`, `\dimen`, `\skip`, and `\box` with the same number. Usually they are allocated downwards from 254, 253, ... Also `\newcount`, `\newdimen`, ... fill up these register numbers from below before switching to higher register numbers by package `etex`. When this occurs, no insertions can be allocated anymore.

Therefore `\newcount`, `\newdimen`, `\newskip`, and `\newbox` are replaced by their global variants (`\globcount`, ...) that use the higher numbers immediately, leaving the room for insertions. There should not be an efficiency penalty because L<sup>A</sup>T<sub>E</sub>X stores the registers of a class in the same Lua table unlike  $\epsilon$ -T<sub>E</sub>X, where registers below 256 are stored in an array and higher numbers are put in a tree structure.

## 1.3 Lua states

`\newluastate {<cmd>}`

Macro `\newluastate` reserves a new Lua state and stores the number in `\cmd`.

## 1.4 Attributes

Nodes can have custom attributes in L<sup>A</sup>T<sub>E</sub>X. These attributes are organized by a new register class. As the other registers up to  $2^{16}$  attributes are supported. An attribute value can be negative that means the attribute is not set. Otherwise T<sub>E</sub>X's range of non-negative integers up to  $2^{31}$  are available.

`\newattribute {<cmd>}`

Macro `\newattribute` defines command `<cmd>` using `\attributedef` using an new attribute number. The new attribute is initially unset.

`\setattribute {<cmd>} {<value>}`

Macro `\setattribute` locally sets attribute command `<cmd>` to the number `<value>`. Valid values range from  $-1$  until  $2^{31}$  (the upper limit is the same as for other T<sub>E</sub>X integer numbers).

`\unsetAttribute {⟨cmd⟩}`

Macro `\unsetAttribute` clears the attribute command `⟨cmd⟩`.

## 1.5 Catcode tables

L<sup>A</sup>T<sub>E</sub>X introduces catcode tables as new feature, see documentation. There is need for discussion, how to deal best:

- `\initcatcodetable` and `\setcatcodetable` act globally.
- `\catcodetable` causes an error if used with an uninitialized catcode table.
- Large catcode table numbers should be avoided because of performance breakdown.
- Use case L<sup>A</sup>T<sub>E</sub>X package: The package must not be surprised by changed catcodes and must not surprise by changing catcodes accidentally. Catcode tables could offer a solution. At the begin a catcode regime with standard catcodes is established and the old one is restored afterwards.
- Use case: L<sup>A</sup>T<sub>E</sub>X's `tex.print` might be used with a catcode table number, for example a table where all entries have catcode “other”.
- Readonly catcode tables.
- Is there is a need for local allocations? (Package `etex`'s `\loc` variants are not used in T<sub>E</sub>X Live 2007.)

### 1.5.1 Interface proposal

The idea: `\newcatcodetable` allocates odd numbered catcode tables. Even numbered tables are managed as stack. Also some catcode tables are defined. These must not be changed.

`\newcatcodetable {⟨cmd⟩}`

Macro `\newcatcodetable` reserves a new catcode table and remembers its number in `⟨cmd⟩`. The catcode table is initialized with ini-T<sub>E</sub>X's catcodes.

`\CatcodeTableIniTeX`  
`\CatcodeTableString`  
`\CatcodeTableOther`  
`\CatcodeTableLaTeX`

These are catcode tables and must not be changed. `\CatcodeTableIniTeX` contains the catcode settings of ini-T<sub>E</sub>X. `\CatcodeTableString` follows T<sub>E</sub>X's convention of `\string`, `\meaning` and friends. The space gets catcode 10 (space), the other characters have catcode 12 (other). In `\CatcodeTableOther` all entries have catcode 12 (other). `\CatcodeTableLaTeX` contains the setting of a pure L<sup>A</sup>T<sub>E</sub>X format ('at' is other).

`\CatcodeTableStack`  
`\IncCatcodeTableStack`  
`\DecCatcodeTableStack`

`\CatcodeTableStack` is the stack pointer. Initially it is catcode table zero. `\IncCatcodeTableStack` and `\DecCatcodeTableStack` increments and decrements the stack pointer. Currently `\IncCatcodeTableStack` does not initialize a

new catcode table. Both increment and decrement operations do not set a catcode table.

`\PushCatcodeTableNumStack`  
`\PopCatcodeTableNumStack`

It can be handy to have a global stack for catcode table numbers to deal with the global assignment property of `\initcatcodetable` and `\savecatcodetable`. `\PushCatcodeTableNumStack` pushes the current catcode table on the stack. `\PopCatcodeTableNumStack` pops the topmost number off the number stack to set the current catcode table. Catcode table zero is used in case of an empty stack.

`\BeginCatcodeRegime { $\langle catcodetable \rangle$ }`  
`\EndCatcodeRegime`

`\BeginCatcodeRegime` remembers the current catcode table number. Then it creates and uses a fresh catcode table on the stack that is initialized by  $\langle catcodetable \rangle$ :

```
\PushCatcodeTableNumStack
\catcodetable $\langle catcodetable \rangle$  \IncCatcodeTableStack
\savecatcodetable\CatcodeTableStack
\catcodetable\CatcodeTableStack
```

`\EndCatcodeRegime` drops the catcode table, created by `\BeginCatcodeRegime` and sets the catcode table that was active before:

```
\DecCatcodeTableStack
\PopCatcodeTableNumStack
```

These macros solve the use case, described earlier for a  $\text{\LaTeX}$  package:

```
% package foobar.sty
\BeginCatcodeRegime\CatcodeTableLaTeX
\makeatletter
% ... package contents ...
\EndCatcodeRegime
% end of package
```

If the package wants to change catcodes after its loading, `\AtBeginDocument` or `\AtEndOfPackage` can be used.

`\SetCatcodeRange { $\langle from \rangle$ } { $\langle to \rangle$ } { $\langle catcode \rangle$ }`

The catcodes of characters in range from  $\langle from \rangle$  to inclusive  $\langle to \rangle$  are set to  $\langle catcode \rangle$ .

## 1.6 Lua module loading

Currently  $\text{\LaTeX}$  (version 0.20) does not support Lua script files inside `TDS:scripts//`, because Lua's mechanism for module loading does not use the `kpathsea` library. Therefore this packages appends a `kpse` loader to the list of Lua's module loaders. It finds the module  $\langle module \rangle$  by

```
kpse.find_file(" $\langle module \rangle$ .lua", "texmfscripts")
```

Unhappily `kpathsea` does not support directory components in a file name. Therefore the Lua convention is not followed to replace dots in the module name by the directory separator.

Example: A Lua script of a package `foobar` wants the following modules:

```
require("foobar.hello.world")
require("org.somewhere.xyz")
```

Then they can be find in:

```
TDS:scripts/foobar/foobar.hello.world.lua
TDS:scripts/foobar/org.somewhere.xyz.lua
```

I would have preferred the following locations, following lua conventions, e.g.:

```
TDS:scripts/foobar/hello/world.lua
TDS:scripts/foobar/org/somewhere/xyz.lua
```

But I do not know, how to achieve this in a reliable way using kpathsea.

### 1.6.1 Package luatex-loader

If someone do not need or want package luatex but it's extension for module loading, then he can use package luatex-loader. Both plain-TeX and L<sup>A</sup>T<sub>E</sub>X are supported.

## 2 Implementation

```
1 <package>
```

### 2.1 Reload check and package identification

Reload check, especially if the package is not used with L<sup>A</sup>T<sub>E</sub>X.

```
2 \begingroup
3 \catcode44 12 % ,
4 \catcode45 12 % -
5 \catcode46 12 % .
6 \catcode58 12 % :
7 \catcode64 11 % @
8 \expandafter\let\expandafter\x\csname ver@luatex.sty\endcsname
9 \ifcase 0%
10 \ifx\x\relax % plain
11 \else
12 \ifx\x\empty % LaTeX
13 \else
14 1%
15 \fi
16 \fi
17 \else
18 \catcode35 6 % #
19 \catcode123 1 % {
20 \catcode125 2 % }
21 \expandafter\ifx\csname PackageInfo\endcsname\relax
22 \def\x#1#2{%
23 \immediate\write-1{Package #1 Info: #2.}%
24 }%
25 \else
26 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
27 \fi
28 \x{luatex}{The package is already loaded}%
29 \endgroup
30 \expandafter\endinput
31 \fi
32 \endgroup
```

Package identification:

```
33 \begingroup
34 \catcode35 6 % #
```

```

35 \catcode40 12 % (
36 \catcode41 12 % )
37 \catcode44 12 % ,
38 \catcode45 12 % -
39 \catcode46 12 % .
40 \catcode47 12 % /
41 \catcode58 12 % :
42 \catcode64 11 % @
43 \catcode123 1 % {
44 \catcode125 2 % }
45 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
46   \def\x#1#2#3[#4]{\endgroup
47     \immediate\write-1{Package: #3 #4}%
48     \xdef#1{#4}%
49   }%
50 \else
51   \def\x#1#2[#3]{\endgroup
52     #2[{#3}]%
53     \ifx#1\relax
54       \xdef#1{#3}%
55     \fi
56   }%
57 \fi
58 \expandafter\x\csname ver@luatex.sty\endcsname
59 \ProvidesPackage{luatex}%
60 [2007/12/12 v0.1 LuaTeX basic definition package (HO)]

```

## 2.2 Catcodes

```

61 \begingroup
62 \catcode123 1 % {
63 \catcode125 2 % }
64 \def\x{\endgroup
65   \expandafter\edef\csname LuT@AtEnd\endcsname{%
66     \catcode35 \the\catcode35\relax
67     \catcode64 \the\catcode64\relax
68     \catcode123 \the\catcode123\relax
69     \catcode125 \the\catcode125\relax
70   }%
71 }%
72 \x
73 \catcode35 6 % #
74 \catcode64 11 % @
75 \catcode123 1 % {
76 \catcode125 2 % }
77 \def\TMP@EnsureCode#1#2{%
78   \edef\LuT@AtEnd{%
79     \LuT@AtEnd
80     \catcode#1 \the\catcode#1\relax
81   }%
82   \catcode#1 #2\relax
83 }
84 \TMP@EnsureCode{10}{12}% ^^J
85 \TMP@EnsureCode{34}{12}% "
86 \TMP@EnsureCode{36}{3}% $
87 \TMP@EnsureCode{39}{12}% '
88 \TMP@EnsureCode{40}{12}% (
89 \TMP@EnsureCode{41}{12}% )
90 \TMP@EnsureCode{42}{12}% *
91 \TMP@EnsureCode{43}{12}% +
92 \TMP@EnsureCode{44}{12}% ,
93 \TMP@EnsureCode{45}{12}% -

```

```

94 \TMP@EnsureCode{46}{12}% .
95 \TMP@EnsureCode{47}{12}% /
96 \TMP@EnsureCode{60}{12}% <
97 \TMP@EnsureCode{61}{12}% =
98 \TMP@EnsureCode{62}{12}% >
99 \TMP@EnsureCode{95}{12}% _ (other!)
100 \TMP@EnsureCode{96}{12}% '

```

## 2.3 Check for LuaTeX

Without L<sup>A</sup>T<sub>E</sub>X there is no point in using this package.

```

101 \begingroup\expandafter\expandafter\expandafter\endgroup
102 \expandafter\ifx\csname RequirePackage\endcsname\relax
103   \input infwarerr.sty\relax
104   \input ifluatex.sty\relax
105 \else
106   \RequirePackage{infwarerr}[2007/09/09]%
107   \RequirePackage{ifluatex}[2007/12/12]%
108 \fi

109 \ifluatex
110 \else
111   \@PackageError{luatex}{%
112     This package may only be run using LuaTeX%
113   }\@ehc
114   \LuT@AtEnd
115   \expandafter\endinput
116 \fi

```

## 2.4 Inherit support for $\epsilon$ -T<sub>E</sub>X

Package etex is not compatible for plain-T<sub>E</sub>X. But it could be present if a format is used that is based on etex.src. Therefore we only load the package in case of L<sup>A</sup>T<sub>E</sub>X and tests its presence independently of the format by looking for \et@xins.

```

117 \begingroup\expandafter\expandafter\expandafter\endgroup
118 \expandafter\ifx\csname et@xins\endcsname\relax
119 \else
120   \RequirePackage{etex}[1998/03/26]%
121 \fi

```

## 2.5 Adaption of $\epsilon$ -T<sub>E</sub>X's register allocation

$\epsilon$ -T<sub>E</sub>X has increased the number of T<sub>E</sub>X registers from 2<sup>8</sup> (256) to 2<sup>15</sup> (32768) for a register class. L<sup>A</sup>T<sub>E</sub>X extends the limit further to 2<sup>16</sup> (65536). The allocation scheme of package etex is not changed. But this can be subject for discussion.

If a register class hasn't registered any local registers yet, then the limit can safely be pushed to 65536.

```

122 \begingroup\expandafter\expandafter\expandafter\endgroup
123 \expandafter\ifx\csname et@xins\endcsname\relax
124   \@PackageWarningNoLine{luatex}{%
125     Support for eTeX is not loaded (etex.src)%
126   }%
127 \else
128   \def\LuT@temp#1{%
129     \ifnum\count27#1=32768 %
130       \count27#1=65536 %
131     \fi
132   }%
133   \LuT@temp0%
134   \LuT@temp1%
135   \LuT@temp2%
136   \LuT@temp3%

```



```

137 \LuT@temp4%
138 \LuT@temp5%
139 \LuT@temp6%

```

$\varepsilon$ -TEX uses an array for the first 256 registers and then a tree structure. L<sup>A</sup>T<sub>E</sub>X stores all registers of a class in one Lua table. There shouldn't be large performance differences. This allows starting immediately in the extended area, leaving room for insertions.

```

140 \let\newcount\globcount
141 \let\newdimen\globdimen
142 \let\newskip\globskip
143 \let\newbox\globbox
144 \fi

```

## 2.6 plain-TEX compatibility

`\@empty`

```

145 \expandafter\ifx\csname @empty\endcsname\relax
146 \def\@empty{}%
147 \fi

```

`\@gobble`

```

148 \expandafter\ifx\csname @gobble\endcsname\relax
149 \long\def\@gobble#1{}%
150 \fi

```

`\@firstofone`

```

151 \expandafter\ifx\csname @firstofone\endcsname\relax
152 \long\def\@firstofone#1{#1}%
153 \fi

```

`\@firstoftwo`

```

154 \expandafter\ifx\csname @firstoftwo\endcsname\relax
155 \long\def\@firstoftwo#1#2{#1}%
156 \fi

```

`\@car`

```

157 \expandafter\ifx\csname @car\endcsname\relax
158 \def\@car#1#2\@nil{#1}%
159 \fi

```

`\@cdr`

```

160 \expandafter\ifx\csname @cdr\endcsname\relax
161 \def\@cdr#1#2\@nil{#2}%
162 \fi

```

`\@ifstar`

```

163 \expandafter\ifx\csname @ifstar\endcsname\relax
164 \def\@ifstar#1{%
165 \ifnextchar*\@firstoftwo{#1}}%
166 }%

```

`\@ifnextchar`

```

167 \long\def\@ifnextchar#1#2#3{%
168 \let\reserved@d=#1%
169 \def\reserved@a{#2}%
170 \def\reserved@b{#3}%
171 \futurelet\@let@token\@ifnch
172 }%

```

```

\@ifnch
173 \def\@ifnch{%
174 \ifx\@let@token\@sptoken
175 \let\reserved@c\@xifnch
176 \else
177 \ifx\@let@token\reserved@d
178 \let\reserved@c\reserved@a
179 \else
180 \let\reserved@c\reserved@b
181 \fi
182 \fi
183 \reserved@c
184 }%

\@sptoken
185 \let\LuT@temp\:%
186 \def\:{\let\@sptoken= }%
187 \: % explicit space

\@xifnch
188 \def\:{\@xifnch}%
189 \expandafter\def\: {%
190 \futurelet\@let@token\@ifnch
191 }%
192 \let\:\LuT@temp
193 \fi

\@tempcnta
194 \expandafter\ifx\csname @tempcnta\endcsname\relax
195 \csname newcount\endcsname\@tempcnta
196 \fi

\@tempcntb
197 \expandafter\ifx\csname @tempcntb\endcsname\relax
198 \csname newcount\endcsname\@tempcntb
199 \fi

\LuT@newcommand
200 \begingroup\expandafter\expandafter\expandafter\endgroup
201 \expandafter\ifx\csname newcommand\endcsname\relax
202 \def\LuT@newcommand#1[#2]#3{%
203 \ifx#1\@undefined
204 \let#1\relax
205 \else
206 \ifx#1\relax
207 \else
208 \@PackageError{luatex}{%
209 \string#1 is already defined.\MessageBreak
210 Redefinition is skipped%
211 }\@ehc
212 \fi
213 \fi
214 \ifx#1\relax
215 \ifcase#2 %
216 \def#1{#3}%
217 \or
218 \def#1##1{#3}%
219 \or
220 \def#1##1##2{#3}%
221 \or
222 \def#1##1##2##3{#3}%

```

```

223     \or
224     \@INTERNAL@ERROR
225     \fi
226     \fi
227 }%
228 \else
229 \def\LuT@newcommand{\newcommand*}%
230 \fi

```

## 2.7 Lua states

\LuT@AllocLuaState

```

231 \newcount\LuT@AllocLuaState
232 \LuT@AllocLuaState=\z@

```

\newluastate

```

233 \LuT@newcommand\newluastate[1]{%
234   \ifnum\LuT@AllocLuaState<65535 %
235     \global\advance\LuT@AllocLuaState\@ne
236     \allocationnumber\LuT@AllocLuaState
237     \global\chardef#1=\allocationnumber
238     \wlog{\string#1=\string\luastate\the\allocationnumber}%
239   \else
240     \errmessage{No room for a new \string\luastate}%
241   \fi
242 }

```

## 2.8 Attributes

### 2.8.1 Allocation

\LuT@AllocAttribute

```

243 \newcount\LuT@AllocAttribute
244 \LuT@AllocAttribute=\m@ne

```

\newattribute

```

245 \LuT@newcommand\newattribute[1]{%
246   \ifnum\LuT@AllocAttribute<65535 %
247     \global\advance\LuT@AllocAttribute\@ne
248     \allocationnumber\LuT@AllocAttribute
249     \global\attributedef#1=\allocationnumber
250     \unsetattribute{#1}%
251     \wlog{\string#1=\string\attribute\the\allocationnumber}%
252   \else
253     \errmessage{No room for a new \string\attribute}%
254   \fi
255 }

```

### 2.8.2 Interface

\setattribute

```

256 \LuT@newcommand\setattribute[2]{%
257   #1=\numexpr#2\relax
258 }

```

\unsetattribute

```

259 \LuT@newcommand\unsetattribute[1]{%
260   #1=\m@ne
261 }

```

## 2.9 Catcode tables

### 2.9.1 Allocation

`\LuT@AllocCatcodeTable`

```
262 \newcount\LuT@AllocCatcodeTable
263 \LuT@AllocCatcodeTable=\m@ne
264 \newcount\CatcodeTableStack
265 \CatcodeTableStack=\z@
```

`\newcatcodetable`

```
266 \LuT@newcommand\newcatcodetable[1]{%
267   \ifnum\LuT@AllocCatcodeTable<1114110 % 0x10FFF is maximal \chardef
268     % or < 268435455 % 228 - 1
269   \global\advance\LuT@AllocCatcodeTable by\tw@
270   \allocationnumber=\LuT@AllocCatcodeTable
271   \global\chardef#1=\allocationnumber
272   \wlog{%
273     \string#1=\string\catcodetable\the\allocationnumber
274   }%
275   \else
276     \errmessage{No room for a new \string\catcodetable}%
277   \fi
278 }%
```

`\IncCatcodeTableStack`

```
279 \LuT@newcommand\IncCatcodeTableStack[0]{%
280   \ifnum\CatcodeTableStack<268435454 %
281     \global\advance\CatcodeTableStack by\tw@
282   \else
283     \@PackageError{luatex}{%
284       Catcode table stack overflow%
285     }\@ehd
286   \fi
287 }
```

`\DecCatcodeTableStack`

```
288 \LuT@newcommand\DecCatcodeTableStack[0]{%
289   \ifnum\CatcodeTableStack>\z@
290     \global\advance\CatcodeTableStack by-2 %
291   \else
292     \@PackageError{luatex}{%
293       Catcode table stack is empty%
294     }\@ehd
295   \fi
296 }
```

### 2.9.2 \SetCatcodeRange

`\SetCatcodeRange`

```
297 \LuT@newcommand\SetCatcodeRange[3]{%
298   \edef\LuT@temp{%
299     \noexpand\@tempcnta=\the\@tempcnta
300     \noexpand\@tempcntb=\the\@tempcntb
301     \noexpand\count@=\the\count@
302     \relax
303   }%
304   \@tempcnta=\numexpr#1\relax
305   \@tempcntb=\numexpr#2\relax
306   \count@=\numexpr#3\relax
307   \loop
308     \unless\ifnum\@tempcnta>\@tempcntb
```

```

309     \catcode\@tempcnta=\count@
310     \advance\@tempcnta by \@ne
311     \repeat
312     \LuT@temp
313 }

```

### 2.9.3 Predefined catcode tables

```

314 \newcatcodetable\CatcodeTableIniTeX
315 \newcatcodetable\CatcodeTableString
316 \newcatcodetable\CatcodeTableOther
317 \newcatcodetable\CatcodeTableLaTeX

318 \initcatcodetable\CatcodeTableIniTeX
319 \begingroup
320   \def\@makeoother#1{\catcode#1=12\relax}%
321   \@firstofone{%
322     \catcodetable\CatcodeTableIniTeX
323     \begingroup
324       \SetCatcodeRange{0}{8}{15}%
325       \catcode9=10 % tab
326       \catcode11=15 %
327       \catcode12=13 % form feed
328       \SetCatcodeRange{14}{31}{15}%
329       \catcode35=6 % hash
330       \catcode36=3 % dollar
331       \catcode38=4 % ampersand
332       \catcode94=7 % circumflex
333       \catcode95=8 % underscore
334       \catcode123=1 % brace left
335       \catcode125=2 % brace right
336       \catcode126=13 % tilde
337       \catcode127=15 %
338       \savecatcodetable\CatcodeTableLaTeX
339     \endgroup
340     \@makeoother{0}% nul
341     \@makeoother{13}% carriage return
342     \@makeoother{37}% percent
343     \@makeoother{92}% backslash
344     \@makeoother{127}%
345     \SetCatcodeRange{65}{90}{12}% A-Z
346     \SetCatcodeRange{97}{122}{12}% a-z
347     \savecatcodetable\CatcodeTableString
348     \@makeoother{32}% space
349     \savecatcodetable\CatcodeTableOther
350   \endgroup
351 }%

```

### 2.9.4 Number stack

`\LuT@NumStackEmpty` A special empty stack value because of `\@cdr`'s brace removal.

```

352 \def\LuT@NumStackEmpty{0}

```

`\LuT@NumStack`

```

353 \let\LuT@NumStack\LuT@NumStackEmpty

```

`\PushCatcodeTableNumStack`

```

354 \LuT@newcommand\PushCatcodeTableNumStack[0]{%
355   \xdef\LuT@NumStack{%
356     {\the\catcodetable}\LuT@NumStack
357   }%
358 }

```

`\PopCatcodeTableNumStack`

```

359 \LuT@newcommand\PopCatcodeTableNumStack[0]{%
360   \ifx\LuT@NumStack\LuT@NumStackEmpty
361     \@PackageWarning{luatex}{Empty catcode table number stack}%
362     \catcodetable\z@
363   \else
364     \catcodetable=\expandafter\@car\LuT@NumStack\@nil\relax
365     \xdef\LuT@NumStack{%
366       \expandafter\@cdr\LuT@NumStack\@nil
367     }%
368   \fi
369 }

```

## 2.9.5 Catcode regime macros

`\BeginCatcodeRegime`

```

370 \LuT@newcommand\BeginCatcodeRegime[1]{%
371   \PushCatcodeTableNumStack
372   \catcodetable=\numexpr#1\relax
373   \IncCatcodeTableStack
374   \savecatcodetable\CatcodeTableStack
375   \catcodetable\CatcodeTableStack
376 }

```

`\EndCatcodeRegime`

```

377 \LuT@newcommand\EndCatcodeRegime[0]{%
378   \DecCatcodeTableStack
379   \PopCatcodeTableNumStack
380 }

```

## 2.10 Lua module loader

```

381 \begingroup\expandafter\expandafter\expandafter\endgroup
382 \expandafter\ifx\csname RequirePackage\endcsname\relax
383   \input luatex-loader.sty\relax
384 \else
385   \RequirePackage{luatex-loader}[2007/12/12]%
386 \fi

387 \LuT@AtEnd
388 </package>

389 <*loader>

```

Reload check, especially if the package is not used with L<sup>A</sup>T<sub>E</sub>X.

```

390 \begingroup
391   \catcode44 12 % ,
392   \catcode45 12 % -
393   \catcode46 12 % .
394   \catcode58 12 % :
395   \catcode64 11 % @
396   \expandafter\let\expandafter\x\csname ver@luatex-loader.sty\endcsname
397   \ifcase 0%
398     \ifx\x\relax % plain
399     \else
400       \ifx\x\empty % LaTeX
401       \else
402         1%
403       \fi
404     \fi
405   \else
406     \catcode35 6 % #
407     \catcode123 1 % {

```

```

408 \catcode125 2 % }
409 \expandafter\ifx\csname PackageInfo\endcsname\relax
410 \def\x#1#2{%
411 \immediate\write-1{Package #1 Info: #2.}%
412 }%
413 \else
414 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
415 \fi
416 \x{luatex-loader}{The package is already loaded}%
417 \endgroup
418 \expandafter\endinput
419 \fi
420 \endgroup
Package identification:
421 \begingroup
422 \catcode35 6 % #
423 \catcode40 12 % (
424 \catcode41 12 % )
425 \catcode44 12 % ,
426 \catcode45 12 % -
427 \catcode46 12 % .
428 \catcode47 12 % /
429 \catcode58 12 % :
430 \catcode64 11 % @
431 \catcode123 1 % {
432 \catcode125 2 % }
433 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
434 \def\x#1#2#3[#4]{\endgroup
435 \immediate\write-1{Package: #3 #4}%
436 \xdef#1{#4}%
437 }%
438 \else
439 \def\x#1#2[#3]{\endgroup
440 #2[{#3}]%
441 \ifx#1\relax
442 \xdef#1{#3}%
443 \fi
444 }%
445 \fi
446 \expandafter\x\csname ver@luatex-loader.sty\endcsname
447 \ProvidesPackage{luatex-loader}%
448 [2007/12/12 v0.1 Lua module loader (HO)]
449 \begingroup
450 \catcode10 12 % ^^J
451 \catcode34 12 % "
452 \catcode39 12 % '
453 \catcode40 12 % (
454 \catcode41 12 % )
455 \catcode44 12 % ,
456 \catcode46 12 % .
457 \catcode61 12 % =
458 \catcode95 12 % _ (other!)
459 \catcode96 12 % '
460 \endlinechar=10 %
461 \directlua0{%
462 do
463 local script = "oberdiek.luatex.lua"
464 local file = kpse.find_file(script, "texmfscripts")
465 if file then
466 texio.write_nl("(" .. file .. ")")
467 dofile(file)
468 else

```

```

469         error("File '" .. script .. "' not found")
470     end
471 end
472 }%
473 \endgroup%
474 </loader>

```

## 2.11 Lua script

Currently L<sup>A</sup>T<sub>E</sub>X does not use KPSE when searching for module files. The following Lua script implements a workaround. It extends `package.loader` by another search method. Modules are found by the module name with extension `..lua` similar to

```
kpsewhich --format=texmfscripts <module>.lua
```

Unhappily `kpsewhich` does not support directory components in the file name. Therefore a module `a.b.c` cannot be installed as `a/b/c.lua`. The script must be named `a.b.c.lua`.

```

475 <lua>

476 module("oberdiek.luatex", package.seeall)

477 function kpse_module_loader(module)
478     local script = module .. ".lua"
479     local file = kpse.find_file(script, "texmfscripts")
480     if file then
481         local loader, error = loadfile(file)
482         if loader then
483             texio.write_nl("(" .. file .. ")")
484             return loader
485         end
486         return "\n\t[oberdiek.luatex.kpse_module_loader] Loading error:\n\t"
487             .. error
488     end
489     return "\n\t[oberdiek.luatex.kpse_module_loader] Search failed"
490 end
491 table.insert(package.loaders, kpse_module_loader)
492 </lua>

```

## 3 Test

```

493 <test2>
494 \documentclass{article}
495 \def\LoadCommand{%
496     \RequirePackage{luatex}[2007/12/12]%
497 }
498 </test2>

499 <test3>
500 \documentclass{article}
501 \def\LoadCommand{%
502     \RequirePackage{luatex-loader}[2007/12/12]%
503 }
504 </test3>

```

### 3.1 Catcode checks for loading

```

505 <test1>
506 \catcode'\{=1 %
507 \catcode'\}=2 %
508 \catcode'\#=6 %
509 \catcode'\@=11 %

```



```

510 \expandafter\ifx\csname count@\endcsname\relax
511   \countdef\count@=255 %
512 \fi
513 \expandafter\ifx\csname @gobble\endcsname\relax
514   \long\def\@gobble#1{}%
515 \fi
516 \expandafter\ifx\csname @firstofone\endcsname\relax
517   \long\def\@firstofone#1{#1}%
518 \fi
519 \expandafter\ifx\csname loop\endcsname\relax
520   \expandafter\@firstofone
521 \else
522   \expandafter\@gobble
523 \fi
524 {%
525   \def\loop#1\repeat{%
526     \def\body{#1}%
527     \iterate
528   }%
529   \def\iterate{%
530     \body
531     \let\next\iterate
532   \else
533     \let\next\relax
534   \fi
535   \next
536 }%
537 \let\repeat=\fi
538 }%
539 \def\RestoreCatcodes{}
540 \count@=0 %
541 \loop
542   \edef\RestoreCatcodes{%
543     \RestoreCatcodes
544     \catcode\the\count@=\the\catcode\count@\relax
545   }%
546 \ifnum\count@<255 %
547   \advance\count@ 1 %
548 \repeat
549
550 \def\RangeCatcodeInvalid#1#2{%
551   \count@=#1\relax
552   \loop
553     \catcode\count@=15 %
554   \ifnum\count@<#2\relax
555     \advance\count@ 1 %
556   \repeat
557 }
558 \expandafter\ifx\csname LoadCommand\endcsname\relax
559   \def\LoadCommand{\input luatex.sty\relax}%
560 \fi
561 \def\Test{%
562   \RangeCatcodeInvalid{0}{47}%
563   \RangeCatcodeInvalid{58}{64}%
564   \RangeCatcodeInvalid{91}{96}%
565   \RangeCatcodeInvalid{123}{255}%
566   \catcode'\@=12 %
567   \catcode'\=0 %
568   \catcode'\{=1 %
569   \catcode'\}=2 %
570   \catcode'\#=6 %
571   \catcode'\[=12 %

```

```

572 \catcode'\]=12 %
573 \catcode'\%=14 %
574 \catcode'\ =10 %
575 \catcode13=5 %
576 \LoadCommand
577 \RestoreCatcodes
578 }
579 \Test
580 \csname @@end\endcsname
581 \end
582 </test1>

```

## 3.2 Catcode tables

### 3.2.1 Predefined catcode tables

```

583 <*test4>
584 \NeedsTeXFormat{LaTeX2e}

```

Remember L<sup>A</sup>T<sub>E</sub>X's initial catcodes in count registers starting at \TestLaTeX.

```

585 \count0=0 %
586 \chardef\TestLaTeX=1000 %
587 \chardef\TestMax=300 %
588 \loop
589 \count\numexpr\TestLaTeX+\count0\relax=\catcode\count0 %
590 \ifnum\count0<\TestMax
591 \advance\count0 by 1 %
592 \repeat
593 \documentclass{minimal}
594 \usepackage{luatex}[2007/12/12]
595 \usepackage{qstest}
596 \IncludeTests{*}
597 \LogTests{log}{*}{*}
598 \makeatletter
599 \def\Check#1{%
600 \Expect*{\the\count@=\the\catcode\count@}%
601 *{\the\count@=#1}%
602 }
603 \newcount\scratch
604 \def\Test#1#2{%
605 \begin{qstest}{CatcodeTable#1}{CatcodeTable#1}%
606 \catcodetable\csname CatcodeTable#1\endcsname
607 \count@=\z@
608 \loop
609 \scratch=#2\relax
610 \Expect*{\the\count@=\the\catcode\count@}%
611 *{\the\count@=\the\scratch}%
612 \ifnum\count@<\TestMax
613 \advance\count@\@ne
614 \repeat
615 \end{qstest}%
616 }
617 \Test{LaTeX}{\the\count\numexpr\TestLaTeX+\count@}
618 \Test{String}{\ifnum\count@=32 10\else 12\fi}
619 \Test{Other}{12}
620 \initcatcodetable99 %
621 \Test{IniTeX}{%
622 0\relax
623 \begingroup
624 \catcodetable99 %
625 \global\scratch=\the\catcode\count@
626 \endgroup
627 }

```

### 3.2.2 Catcode table number stack

```

628 \begin{qstest}{CatcodeTableNumStack}{CatcodeTableNumStack}
629   \def\TestStack#1{%
630     \Expect*{\LuT@NumStack}{#1}%
631   }%
632   \TestStack{0}%
633   \PushCatcodeTableNumStack
634   \TestStack{{0}0}%
635   \@firstofone{%
636     \begingroup
637       \initcatcodetable12 %
638       \catcodetable12 %
639       \PushCatcodeTableNumStack
640       \TestStack{{12}{0}0}%
641       \PopCatcodeTableNumStack
642       \TestStack{{0}0}%
643       \PopCatcodeTableNumStack
644       \TestStack{0}%
645       \def\TestWarning{Missing empty stack warning}%
646       \def\@PackageWarning#1#2{\def\TestWarning{empty stack}}%
647       \PopCatcodeTableNumStack
648       \TestStack{0}%
649       \Expect*{\TestWarning}{empty stack}%
650     \endgroup
651   }%
652 \end{qstest}

```

### 3.2.3 Catcode table stack

```

653 \begin{qstest}{CatcodeTableStack}{CatcodeTableStack}
654   \def\TestStack#1{%
655     \Expect*{\the\CatcodeTableStack}{#1}%
656   }%
657   \TestStack{0}%
658   \IncCatcodeTableStack
659   \TestStack{2}%
660   \IncCatcodeTableStack
661   \TestStack{4}%
662   \begingroup
663     \IncCatcodeTableStack
664     \TestStack{6}%
665   \endgroup
666   \TestStack{6}%
667   \begingroup
668     \DecCatcodeTableStack
669     \TestStack{4}%
670   \endgroup
671   \TestStack{4}%
672   \DecCatcodeTableStack
673   \TestStack{2}%
674   \DecCatcodeTableStack
675   \TestStack{0}%
676   \begingroup
677     \def\TestError{Missing error}%
678     \def\@PackageError#1#2#3{%
679       \def\TestError{Empty stack}%
680     }%
681     \DecCatcodeTableStack
682     \TestStack{0}%
683     \Expect*{\TestError}{Empty stack}%
684   \endgroup
685 \end{qstest}

```

### 3.2.4 Catcode regime macros

```

686 \begin{qstest}{CatcodeRegime}{CatcodeRegime}
687   \def\TestStacks#1#2#3{%
688     \Expect*{\the\catcodetable}{#1}%
689     \Expect*{\the\CatcodeTableStack}{#2}%
690     \Expect*{\LuT@NumStack}{#3}%
691   }%
692   \TestStacks{0}{0}{0}%
693   \catcode'\|=7 %
694   \BeginCatcodeRegime\CatcodeTableLaTeX
695     \TestStacks{2}{2}{00}%
696     \Expect*{\the\catcode'\|}{12}%
697   \EndCatcodeRegime
698   \TestStacks{0}{0}{0}%
699   \Expect*{\the\catcode'\|}{7}%
700 \end{qstest}

```

### 3.3 Attribute allocation

```

701 \begin{qstest}{Attributes}{Attributes}
702   \newattribute\TestAttr
703   \Expect*{\meaning\TestAttr}%
704     *{\string\attribute\number\allocationnumber}%
705   \Expect*{\the\allocationnumber}{0}%
706   \begingroup
707     \newattribute\TestAttr
708     \Expect*{\the\allocationnumber}{1}%
709   \endgroup
710   \Expect*{\the\allocationnumber}{0}%
711   \Expect*{\meaning\TestAttr}*{\string\attribute1}%
712   \Expect*{\the\TestAttr}{-1}%
713   \def\Test#1{%
714     \setattribute\TestAttr{#1}%
715     \Expect*{\the\TestAttr}{#1}%
716   }%
717   \Test{0}%
718   \Test{1}%
719   \Test{-1}%
720   \Test{123}%
721   \unsetattribute\TestAttr
722   \Expect*{\the\TestAttr}{-1}%
723   \begingroup
724     \Expect*{\the\TestAttr}{-1}%
725     \Test{1234}%
726   \endgroup
727   \Expect*{\the\TestAttr}{-1}%
728 \end{qstest}

```

### 3.4 Lua states

```

729 \begin{qstest}{LuaState}{LuaState}
730   \newluastate\TestLuaState
731   \Expect*{\number\TestLuaState}{1}%
732   \newluastate\TestLuaState
733   \Expect*{\number\TestLuaState}{2}%
734 \end{qstest}

735 @@end
736 /test4

```

### 3.5 Short test for plain-TeX

```

737 (*test5)
738 \input luatex.sty\relax
739 \newluastate\TestLuaState

```

```

740 \newattribute\TestAttr
741 \setattribute\TestAttr{10}
742 \unsetattribute\TestAttr
743 \newcatcodetable\TestCTa
744 \begingroup
745   \SetCatcodeRange{'A'}{'Z'}{12}%
746 \endgroup
747 \BeginCatcodeRegime\CatcodeTableLaTeX
748 \EndCatcodeRegime
749 \end
750 </test5>

```

## 4 Installation

### 4.1 Download

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

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

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

**Bundle.** All the packages of the bundle ‘oberdiek’ are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

[CTAN:install/macros/latex/contrib/oberdiek.tds.zip](#)

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

### 4.2 Bundle installation

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

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

**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 luatex.dtx
```

**TDS.** Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

---

<sup>1</sup><http://ftp.ctan.org/tex-archive/>

```

luatex.sty          → tex/generic/oberdiek/luatex.sty
luatex-loader.sty   → tex/generic/oberdiek/luatex-loader.sty
oberdiek.luatex.lua  → scripts/oberdiek/oberdiek.luatex.lua
luatex.pdf          → doc/latex/oberdiek/luatex.pdf
test/luatex-test1.tex → doc/latex/oberdiek/test/luatex-test1.tex
test/luatex-test2.tex → doc/latex/oberdiek/test/luatex-test2.tex
test/luatex-test3.tex → doc/latex/oberdiek/test/luatex-test3.tex
test/luatex-test4.tex → doc/latex/oberdiek/test/luatex-test4.tex
test/luatex-test5.tex → doc/latex/oberdiek/test/luatex-test5.tex
luatex.dtx          → source/latex/oberdiek/luatex.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  $\text{T}_{\text{E}}\text{X}$  distribution (te $\text{T}_{\text{E}}\text{X}$ , mik $\text{T}_{\text{E}}\text{X}$ , ...) relies on file name databases, you must refresh these. For example, te $\text{T}_{\text{E}}\text{X}$  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 luatex.pdf unpack_files output .
```

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

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

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

If you insist on using  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  for `docstrip` (really, `docstrip` does not need  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{luatex.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 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ :

```

pdflatex luatex.dtx
makeindex -s gind.ist luatex.idx
pdflatex luatex.dtx
makeindex -s gind.ist luatex.idx
pdflatex luatex.dtx

```

## 5 History

[2007/12/12 v0.1]

- 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	C
\# ..... 508, 570	\catcode ..... 3, 4, 5, 6, 7, 18, 19, 20, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 62, 63, 66, 67, 68, 69, 73, 74, 75, 76, 80, 82, 309, 320, 325, 326, 327, 329, 330, 331, 332, 333, 334, 335, 336, 337, 391, 392, 393, 394, 395, 406, 407, 408, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 506, 507, 508, 509, 544, 553, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 589, 600, 610, 625, 693, 696, 699
\% ..... 573	\catcodetable 273, 276, 322, 356, 362, 364, 372, 375, 606, 624, 638, 688
\: ..... 185, 186, 187, 188, 189, 192	\CatcodeTableIniTeX .. 4, 314, 318, 322
\@ ..... 509, 566	\CatcodeTableLaTeX . 317, 338, 694, 747
\@@end ..... 735	\CatcodeTableOther ..... 316, 349
\@INTERNAL@ERROR ..... 224	\CatcodeTableStack 4, 264, 265, 280, 281, 289, 290, 374, 375, 655, 689
\@PackageError 111, 208, 283, 292, 678	\CatcodeTableString ..... 315, 347
\@PackageWarning ..... 361, 646	\chardef ..... 237, 267, 271, 586, 587
\@PackageWarningNoLine ..... 124	\Check ..... 599
\@car ..... 157, 364	\count 129, 130, 585, 589, 590, 591, 617
\@cdr ..... 160, 366	\count@ ..... 301, 306, 309, 511, 540, 544, 546, 547, 551, 553, 554, 555, 600, 601, 607, 610, 611, 612, 613, 617, 618, 625
\@ehc ..... 113, 211	\countdef ..... 511
\@ehd ..... 285, 294	\csname . 8, 21, 45, 58, 65, 102, 118, 123, 145, 148, 151, 154, 157, 160, 163, 194, 195, 197, 198, 201, 382, 396, 409, 433, 446, 510, 513, 516, 519, 558, 580, 606
\@empty ..... 145	
\@firstofone .. 151, 321, 517, 520, 635	
\@firstoftwo ..... 154, 165	
\@gobble ..... 148, 514, 522	
\@ifnch ..... 171, 173, 190	
\@ifnextchar ..... 165, 167	
\@ifstar ..... 163	
\@let@token ..... 171, 174, 177, 190	
\@makeoother ..... . 320, 340, 341, 342, 343, 344, 348	
\@ne ..... 235, 247, 310, 613	
\@nil ..... 158, 161, 364, 366	
\@sptoken ..... 174, 185	
\@tempcnta 194, 299, 304, 308, 309, 310	
\@tempcntb ..... 197, 300, 305, 308	
\@undefined ..... 203	
\@xifnch ..... 175, 188	
\[ ..... 571	
\ \ ..... 567	
\{ ..... 506, 568	
\} ..... 507, 569	
\] ..... 572	
\  ..... 693, 696, 699	
\_ ..... 574	
A	D
\advance ..... 235, 247, 269, 281, 290, 310, 547, 555, 591, 613	\DecCatcodeTableStack ..... ..... 288, 378, 668, 672, 674, 681
\allocationnumber ..... . 236, 237, 238, 248, 249, 251, 270, 271, 273, 704, 705, 708, 710	\directlua ..... 461
\attribute ..... 251, 253, 704, 711	\documentclass ..... 494, 500, 593
\attributedef ..... 249	
B	E
\begin .... 605, 628, 653, 686, 701, 729	\empty ..... 12, 400
\BeginCatcodeRegime .. 5, 370, 694, 747	\end 581, 615, 652, 685, 700, 728, 734, 749
\body ..... 526, 530	\EndCatcodeRegime .... 377, 697, 748
	\endcsname 8, 21, 45, 58, 65, 102, 118, 123, 145, 148, 151, 154, 157, 160, 163, 194, 195, 197, 198, 201, 382, 396, 409, 433, 446, 510, 513, 516, 519, 558, 580, 606
	\endinput ..... 30, 115, 418
	\endlinechar ..... 460
	\errmessage ..... 240, 253, 276

\Expect	600, 610, 630, 649, 655, 683, 688, 689, 690, 696, 699, 703, 705, 708, 710, 711, 712, 715, 722, 724, 727, 731, 733
<b>F</b>	
\futurelet	171, 190
<b>G</b>	
\gloabox	143
\globcount	140
\globdimen	141
\globskip	142
<b>I</b>	
\ifcase	9, 215, 397
\ifluatex	109
\ifnum	129, 234, 246, 267, 280, 289, 308, 546, 554, 590, 612, 618
\ifx	10, 12, 21, 45, 53, 102, 118, 123, 145, 148, 151, 154, 157, 160, 163, 174, 177, 194, 197, 201, 203, 206, 214, 360, 382, 398, 400, 409, 433, 441, 510, 513, 516, 519, 558
\immediate	23, 47, 411, 435
\IncCatcodeTableStack	279, 373, 658, 660, 663
\IncludeTests	596
\initcatcodetable	318, 620, 637
\input	103, 104, 383, 559, 738
\iterate	527, 529, 531
<b>L</b>	
\LoadCommand	495, 501, 559, 576
\LogTests	597
\loop	307, 525, 541, 552, 588, 608
\luastate	238, 240
\LuT@AllocAttribute	243, 246, 247, 248
\LuT@AllocCatcodeTable	262, 267, 269, 270
\LuT@AllocLuaState	231, 234, 235, 236
\LuT@AtEnd	78, 79, 114, 387
\LuT@newcommand	200, 233, 245, 256, 259, 266, 279, 288, 297, 354, 359, 370, 377
\LuT@NumStack	353, 355, 356, 360, 364, 365, 366, 630, 690
\LuT@NumStackEmpty	352, 353, 360
\LuT@temp	128, 133, 134, 135, 136, 137, 138, 139, 185, 192, 298, 312
<b>M</b>	
\m@ne	244, 260, 263
\makeatletter	598
\meaning	703, 711
\MessageBreak	209
<b>N</b>	
\n	486, 489
\NeedsTeXFormat	584
\newattribute	3, 245, 702, 707, 740
\newbox	143
\newcatcodetable	4, 266, 314, 315, 316, 317, 743
\newcommand	229
\newcount	140, 231, 243, 262, 264, 603
\newdimen	141
\newluastate	3, 233, 730, 732, 739
\newskip	142
\next	531, 533, 535
\number	704, 731, 733
\numexpr	257, 304, 305, 306, 372, 589, 617
<b>P</b>	
\PackageInfo	26, 414
\PopCatcodeTableNumStack	359, 379, 641, 643, 647
\ProvidesPackage	59, 447
\PushCatcodeTableNumStack	5, 354, 371, 633, 639
<b>R</b>	
\RangeCatcodeInvalid	550, 562, 563, 564, 565
\repeat	311, 525, 537, 548, 556, 592, 614
\RequirePackage	106, 107, 120, 385, 496, 502
\reserved@a	169, 178
\reserved@b	170, 180
\reserved@c	175, 178, 180, 183
\reserved@d	168, 177
\RestoreCatcodes	539, 542, 543, 577
<b>S</b>	
\savecatcodetable	338, 347, 349, 374
\scratch	603, 609, 611, 625
\setattribute	3, 256, 714, 741
\SetCatcodeRange	5, 297, 324, 328, 345, 346, 745
<b>T</b>	
\t	486, 489
\Test	561, 579, 604, 617, 618, 619, 621, 713, 717, 718, 719, 720, 725
\TestAttr	702, 703, 707, 711, 712, 714, 715, 721, 722, 724, 727, 740, 741, 742
\TestCTa	743
\TestError	677, 679, 683
\TestLaTeX	586, 589, 617
\TestLuaState	730, 731, 732, 733, 739
\TestMax	587, 590, 612
\TestStack	629, 632, 634, 640, 642, 644, 648, 654, 657, 659, 661, 664, 666, 669, 671, 673, 675, 682
\TestStacks	687, 692, 695, 698
\TestWarning	645, 646, 649
\the	66, 67, 68, 69, 80, 238, 251, 273, 299, 300, 301, 356, 544, 600, 601, 610, 611, 617, 625, 655, 688, 689, 696, 699, 705, 708, 710, 712, 715, 722, 724, 727
\TMP@EnsureCode	77, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100
\tw@	269, 281



U		X	
<code>\unless</code>	..... 308	<code>\x</code>	..... 8, 10, 12, 22, 26,
<code>\unsetattribute</code>	. 4, 250, 259, 721, 742		28, 46, 51, 58, 64, 72, 396, 398,
<code>\usepackage</code>	..... 594, 595		400, 410, 414, 416, 434, 439, 446
W		Z	
<code>\wlog</code>	..... 238, 251, 272		
<code>\write</code>	..... 23, 47, 411, 435	<code>\z@</code>	..... 232, 265, 289, 362, 607