

# The `luatex` package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2009/12/02 v0.3

## 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	17
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-TeX	21
<b>4</b>	<b>Installation</b>	<b>21</b>
4.1	Download	21
4.2	Bundle installation	21
4.3	Package installation	22
4.4	Refresh file name databases	22
4.5	Some details for the interested	22
<b>5</b>	<b>History</b>	<b>23</b>
[2007/12/12 v0.1]		23
[2009/04/10 v0.2]		23
[2009/12/02 v0.3]		23
<b>6</b>	<b>Index</b>	<b>23</b>

# 1 Documentation

## 1.1 Introduction

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-TeX or L<sup>A</sup>TeX implement an allocation scheme for registers. A package reserves with `\newcount` an unused register number for its own exclusive use.

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

Now L<sup>A</sup>TeX has reached beta state. It inherits most of pdfTeX's features including  $\varepsilon$ -TeX. Also it implements new concepts such as attributes or catcode tables.

### 1.1.1 L<sup>A</sup>TeX

L<sup>A</sup>TeX 2 $\varepsilon$  is frozen and therefore refuses to even notice the new TeX variants. Not even the old  $\varepsilon$ -TeX is supported by its kernel. At least there is a third party package `etex` that manages the new  $\varepsilon$ -TeX resources.

This package tries to do the same for L<sup>A</sup>TeX and starts to support at least a few of the new features.

### 1.1.2 plain-TeX

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

## 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 {⟨catcodetable⟩}
\EndCatcodeRegime
```

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

```
\PushCatcodeTableNumStack
\catcodetable⟨catcodetable⟩ \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 `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 {⟨from⟩} {⟨to⟩} {⟨catcode⟩}
```

The catcodes of characters in range from `⟨from⟩` to inclusive `⟨to⟩` are set to `⟨catcode⟩`.

## 1.6 Lua module loading

Currently `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 `⟨module⟩` by

```
kpse.find_file("⟨module⟩.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 \catcode123 1 % {
9 \catcode125 2 % }
10 \expandafter\let\expandafter\x\csname ver@luatex.sty\endcsname
11 \ifx\x\relax % plain-TeX, first loading
12 \else
13 \def\empty{}%
14 \ifx\x\empty % LaTeX, first loading,
15 % variable is initialized, but \ProvidesPackage not yet seen
16 \else
17 \catcode35 6 % #
18 \expandafter\ifx\csname PackageInfo\endcsname\relax
19 \def\x#1#2{%
20 \immediate\write-1{Package #1 Info: #2.}%
21 }%
22 \else
23 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
24 \fi
25 \x{luatex}{The package is already loaded}%
26 \aftergroup\endinput
27 \fi
28 \fi
29 \endgroup
```

Package identification:

```
30 \begingroup
31 \catcode35 6 % #
32 \catcode40 12 % (
33 \catcode41 12 % )
34 \catcode44 12 % ,
```

```

35 \catcode45 12 % -
36 \catcode46 12 % .
37 \catcode47 12 % /
38 \catcode58 12 % :
39 \catcode64 11 % @
40 \catcode91 12 % [
41 \catcode93 12 % ]
42 \catcode123 1 % {
43 \catcode125 2 % }
44 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
45   \def\x#1#2#3[#4]{\endgroup
46     \immediate\write-1{Package: #3 #4}%
47     \xdef#1{#4}%
48   }%
49 \else
50   \def\x#1#2[#3]{\endgroup
51     #2[#{#3}]%
52     \ifx#1\@undefined
53       \xdef#1{#3}%
54     \fi
55     \ifx#1\relax
56       \xdef#1{#3}%
57     \fi
58   }%
59 \fi
60 \expandafter\x\csname ver@luatex.sty\endcsname
61 \ProvidesPackage{luatex}%
62 [2009/12/02 v0.3 LuaTeX basic definition package (HO)]

```

## 2.2 Catcodes

```

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

```

```

94 \TMP@EnsureCode{44}{12}% ,
95 \TMP@EnsureCode{45}{12}% -
96 \TMP@EnsureCode{46}{12}% .
97 \TMP@EnsureCode{47}{12}% /
98 \TMP@EnsureCode{60}{12}% <
99 \TMP@EnsureCode{61}{12}% =
100 \TMP@EnsureCode{62}{12}% >
101 \TMP@EnsureCode{95}{12}% _ (other!)
102 \TMP@EnsureCode{96}{12}% ‘

```

## 2.3 Check for LuaTeX

Without LuaTeX there is no point in using this package.

```

103 \begingroup\expandafter\expandafter\expandafter\endgroup
104 \expandafter\ifx\csname RequirePackage\endcsname\relax
105   \input infwarerr.sty\relax
106   \input ifluatex.sty\relax
107 \else
108   \RequirePackage{infwarerr}[2007/09/09]%
109   \RequirePackage{ifluatex}[2009/04/10]%
110 \fi

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

```

## 2.4 Inherit support for $\varepsilon$ -TeX

Package etex is not compatible for plain-TeX. 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>TeX and tests its presence independently of the format by looking for \et@xins.

```

119 \begingroup\expandafter\expandafter\expandafter\endgroup
120 \expandafter\ifx\csname RequirePackage\endcsname\relax
121 \else
122   \RequirePackage{etex}[1998/03/26]%
123 \fi

```

## 2.5 Adaption of $\varepsilon$ -TeX's register allocation

$\varepsilon$ -TeX has increased the number of TeX registers from  $2^8$  (256) to  $2^{15}$  (32768) for a register class. LuaTeX extends the limit further to  $2^{16}$  (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.

```

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

```



```

137 \LuT@temp2%
138 \LuT@temp3%
139 \LuT@temp4%
140 \LuT@temp5%
141 \LuT@temp6%

```

$\varepsilon$ -T<sub>E</sub>X 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.

```

142 \let\newcount\globcount
143 \let\newdimen\globdimen
144 \let\newskip\globskip
145 \let\newbox\globbox
146 \fi

```

## 2.6 plain-T<sub>E</sub>X compatibility

`\@empty`

```

147 \expandafter\ifx\csname @empty\endcsname\relax
148 \def\@empty{}%
149 \fi

```

`\@gobble`

```

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

```

`\@firstofone`

```

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

```

`\@firstoftwo`

```

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

```

`\@car`

```

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

```

`\@cdr`

```

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

```

`\@ifstar`

```

165 \expandafter\ifx\csname @ifstar\endcsname\relax
166 \def\@ifstar#1{%
167 \ifnextchar*\@firstoftwo{#1}}%
168 }%

```

`\@ifnextchar`

```

169 \long\def\@ifnextchar#1#2#3{%
170 \let\reserved@d=#1%
171 \def\reserved@a{#2}%
172 \def\reserved@b{#3}%
173 \futurelet\@let@token\@ifnch
174 }%

```

```

\@ifnch
175 \def\@ifnch{%
176 \ifx\@let@token\@sptoken
177 \let\reserved@c\@xifnch
178 \else
179 \ifx\@let@token\reserved@d
180 \let\reserved@c\reserved@a
181 \else
182 \let\reserved@c\reserved@b
183 \fi
184 \fi
185 \reserved@c
186 }%

\@sptoken
187 \let\LuT@temp\:%
188 \def\:{\let\@sptoken= }%
189 \: % explicit space

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

\@tempcnta
196 \expandafter\ifx\csname @tempcnta\endcsname\relax
197 \csname newcount\endcsname\@tempcnta
198 \fi

\@tempcntb
199 \expandafter\ifx\csname @tempcntb\endcsname\relax
200 \csname newcount\endcsname\@tempcntb
201 \fi

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

```

```

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

```

## 2.7 Lua states

\LuT@AllocLuaState

```

233 \newcount\LuT@AllocLuaState
234 \LuT@AllocLuaState=\z@

```

\newluastate

```

235 \LuT@newcommand\newluastate[1]{%
236   \ifnum\LuT@AllocLuaState<65535 %
237     \global\advance\LuT@AllocLuaState\@ne
238     \allocationnumber\LuT@AllocLuaState
239     \global\chardef#1=\allocationnumber
240     \wlog{\string#1=\string\luastate\the\allocationnumber}%
241   \else
242     \errmessage{No room for a new \string\luastate}%
243   \fi
244 }

```

## 2.8 Attributes

### 2.8.1 Allocation

\LuT@AllocAttribute

```

245 \newcount\LuT@AllocAttribute
246 \LuT@AllocAttribute=\m@ne

```

\newattribute

```

247 \LuT@newcommand\newattribute[1]{%
248   \ifnum\LuT@AllocAttribute<65535 %
249     \global\advance\LuT@AllocAttribute\@ne
250     \allocationnumber\LuT@AllocAttribute
251     \global\attributedef#1=\allocationnumber
252     \unsetattribute{#1}%
253     \wlog{\string#1=\string\attribute\the\allocationnumber}%
254   \else
255     \errmessage{No room for a new \string\attribute}%
256   \fi
257 }

```

### 2.8.2 Interface

\setattribute

```

258 \LuT@newcommand\setattribute[2]{%
259   #1=\numexpr#2\relax
260 }

```

\unsetattribute

```

261 \ifnum\luatexversion<37
262   \LuT@newcommand\LuT@UnsetAttributeValue[0]{}%
263   \let\LuT@UnsetAttributeValue\m@ne
264 \else

```

```

265 \LuT@newcommand\LuT@UnsetAttributeValue[0]{-2147483647 }%
266 \fi
267 \LuT@newcommand\unsetattribute[1]{%
268   #1=\LuT@UnsetAttributeValue
269 }

```

## 2.9 Catcode tables

### 2.9.1 Allocation

\LuT@AllocCatcodeTable

```

270 \newcount\LuT@AllocCatcodeTable
271 \LuT@AllocCatcodeTable=\m@ne
272 \newcount\CatcodeTableStack
273 \CatcodeTableStack=\z@

```

\newcatcodetable

```

274 \LuT@newcommand\newcatcodetable[1]{%
275   \ifnum\LuT@AllocCatcodeTable<1114110 % 0x10FFFF is maximal \chardef
276     % or < 268435455 %  $2^{28} - 1$ 
277     \global\advance\LuT@AllocCatcodeTable by\tw@
278     \allocationnumber=\LuT@AllocCatcodeTable
279     \global\chardef#1=\allocationnumber
280     \wlog{%
281       \string#1=\string\catcodetable\the\allocationnumber
282     }%
283   \else
284     \errmessage{No room for a new \string\catcodetable}%
285   \fi
286 }%

```

\IncCatcodeTableStack

```

287 \LuT@newcommand\IncCatcodeTableStack[0]{%
288   \ifnum\CatcodeTableStack<268435454 %
289     \global\advance\CatcodeTableStack by\tw@
290   \else
291     \@PackageError{luatex}{%
292       Catcode table stack overflow%
293     }\@ehd
294   \fi
295 }

```

\DecCatcodeTableStack

```

296 \LuT@newcommand\DecCatcodeTableStack[0]{%
297   \ifnum\CatcodeTableStack>\z@
298     \global\advance\CatcodeTableStack by-2 %
299   \else
300     \@PackageError{luatex}{%
301       Catcode table stack is empty%
302     }\@ehd
303   \fi
304 }

```

### 2.9.2 \SetCatcodeRange

\SetCatcodeRange

```

305 \LuT@newcommand\SetCatcodeRange[3]{%
306   \edef\LuT@temp{%
307     \noexpand\@tempcnta=\the\@tempcnta
308     \noexpand\@tempcntb=\the\@tempcntb
309     \noexpand\count@=\the\count@

```

```

310     \relax
311 }%
312 \@tempcnta=\numexpr#1\relax
313 \@tempcntb=\numexpr#2\relax
314 \count@=\numexpr#3\relax
315 \loop
316   \unless\ifnum\@tempcnta>\@tempcntb
317     \catcode\@tempcnta=\count@
318     \advance\@tempcnta by \@ne
319   \repeat
320 \LuT@temp
321 }

```

### 2.9.3 Predefined catcode tables

```

322 \newcatcodetable\CatcodeTableIniTeX
323 \newcatcodetable\CatcodeTableString
324 \newcatcodetable\CatcodeTableOther
325 \newcatcodetable\CatcodeTableLaTeX

326 \initcatcodetable\CatcodeTableIniTeX
327 \begingroup
328   \def\@makeother#1{\catcode#1=12\relax}%
329   \@firstofone{%
330     \catcodetable\CatcodeTableIniTeX
331     \begingroup
332       \SetCatcodeRange{0}{8}{15}%
333       \catcode9=10 % tab
334       \catcode11=15 %
335       \catcode12=13 % form feed
336       \SetCatcodeRange{14}{31}{15}%
337       \catcode35=6 % hash
338       \catcode36=3 % dollar
339       \catcode38=4 % ampersand
340       \catcode94=7 % circumflex
341       \catcode95=8 % underscore
342       \catcode123=1 % brace left
343       \catcode125=2 % brace right
344       \catcode126=13 % tilde
345       \catcode127=15 %
346       \savecatcodetable\CatcodeTableLaTeX
347     \endgroup
348     \@makeother{0}% nul
349     \@makeother{13}% carriage return
350     \@makeother{37}% percent
351     \@makeother{92}% backslash
352     \@makeother{127}%
353     \SetCatcodeRange{65}{90}{12}% A-Z
354     \SetCatcodeRange{97}{122}{12}% a-z
355     \savecatcodetable\CatcodeTableString
356     \@makeother{32}% space
357     \savecatcodetable\CatcodeTableOther
358   \endgroup
359 }%

```

### 2.9.4 Number stack

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

```

360 \def\LuT@NumStackEmpty{0}

```

`\LuT@NumStack`

```

361 \let\LuT@NumStack\LuT@NumStackEmpty

```

`\PushCatcodeTableNumStack`

```

362 \LuT@newcommand\PushCatcodeTableNumStack[0]{%
363   \xdef\LuT@NumStack{%
364     {\the\catcodetable}\LuT@NumStack
365   }%
366 }
```

`\PopCatcodeTableNumStack`

```

367 \LuT@newcommand\PopCatcodeTableNumStack[0]{%
368   \ifx\LuT@NumStack\LuT@NumStackEmpty
369     \@PackageWarning{luatex}{Empty catcode table number stack}%
370     \catcodetable\z@
371   \else
372     \catcodetable=\expandafter\@car\LuT@NumStack\@nil\relax
373     \xdef\LuT@NumStack{%
374       \expandafter\@cdr\LuT@NumStack\@nil
375     }%
376   \fi
377 }
```

### 2.9.5 Catcode regime macros

`\BeginCatcodeRegime`

```

378 \LuT@newcommand\BeginCatcodeRegime[1]{%
379   \PushCatcodeTableNumStack
380   \catcodetable=\numexpr#1\relax
381   \IncCatcodeTableStack
382   \savecatcodetable\CatcodeTableStack
383   \catcodetable\CatcodeTableStack
384 }
```

`\EndCatcodeRegime`

```

385 \LuT@newcommand\EndCatcodeRegime[0]{%
386   \DecCatcodeTableStack
387   \PopCatcodeTableNumStack
388 }
```

## 2.10 Lua module loader

```

389 \begingroup\expandafter\expandafter\expandafter\endgroup
390 \expandafter\ifx\csname RequirePackage\endcsname\relax
391   \input luatex-loader.sty\relax
392 \else
393   \RequirePackage{luatex-loader}[2009/12/02]%
394 \fi

395 \LuT@AtEnd
396 </package>

397 <*loader>

    Reload check, especially if the package is not used with LATEX.
398 \begingroup
399   \catcode44 12 % ,
400   \catcode45 12 % -
401   \catcode46 12 % .
402   \catcode58 12 % :
403   \catcode64 11 % @
404   \catcode123 1 % {
405   \catcode125 2 % }
406   \expandafter\let\expandafter\x\csname ver@luatex-loader.sty\endcsname
407   \ifx\x\relax % plain-TeX, first loading
408   \else
```

```

409 \def\empty{}%
410 \ifx\x\empty % LaTeX, first loading,
411 % variable is initialized, but \ProvidesPackage not yet seen
412 \else
413 \catcode35 6 % #
414 \expandafter\ifx\csname PackageInfo\endcsname\relax
415 \def\x#1#2{%
416 \immediate\write-1{Package #1 Info: #2.}%
417 }%
418 \else
419 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
420 \fi
421 \x{luatex-loader}{The package is already loaded}%
422 \aftergroup\endinput
423 \fi
424 \fi
425 \endgroup
Package identification:
426 \begingroup
427 \catcode35 6 % #
428 \catcode40 12 % (
429 \catcode41 12 % )
430 \catcode44 12 % ,
431 \catcode45 12 % -
432 \catcode46 12 % .
433 \catcode47 12 % /
434 \catcode58 12 % :
435 \catcode64 11 % @
436 \catcode91 12 % [
437 \catcode93 12 % ]
438 \catcode123 1 % {
439 \catcode125 2 % }
440 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
441 \def\x#1#2#3[#4]{\endgroup
442 \immediate\write-1{Package: #3 #4}%
443 \xdef#1{#4}%
444 }%
445 \else
446 \def\x#1#2[#3]{\endgroup
447 #2[#3]}%
448 \ifx#1\undefined
449 \xdef#1{#3}%
450 \fi
451 \ifx#1\relax
452 \xdef#1{#3}%
453 \fi
454 }%
455 \fi
456 \expandafter\x\csname ver@luatex-loader.sty\endcsname
457 \ProvidesPackage{luatex-loader}%
458 [2009/12/02 v0.3 Lua module loader (H0)]
459 \begingroup
460 \catcode10 12 % ^^J
461 \catcode34 12 % "
462 \catcode39 12 % '
463 \catcode40 12 % (
464 \catcode41 12 % )
465 \catcode44 12 % ,
466 \catcode46 12 % .
467 \catcode60 12 % <
468 \catcode61 12 % =
469 \catcode95 12 % _ (other!)

```

```

470 \catcode96 12 % ‘
471 \endlinechar=10 %
472 \ifnum\luatexversion<36 %
473   \directlua0%
474 \else %
475   \expandafter\directlua %
476 \fi %
477 {%
478   do
479     local script = "oberdiek.luatex.lua"
480     local file = kpse.find_file(script, "texmfscripts")
481     if file then
482       texio.write_nl("(" .. file .. ")")
483       dofile(file)
484     else
485       error("File ‘" .. script .. ’’ not found")
486     end
487   end
488 }%
489 \endgroup%
490 </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`.

```

491 <lua>
492 module("oberdiek.luatex", package.seeall)
493 function kpse_module_loader(module)
494   local script = module .. ".lua"
495   local file = kpse.find_file(script, "texmfscripts")
496   if file then
497     local loader, error = loadfile(file)
498     if loader then
499       texio.write_nl("(" .. file .. ")")
500       return loader
501     end
502     return "\n\t[oberdiek.luatex.kpse_module_loader] Loading error:\n\t"
503         .. error
504   end
505   return "\n\t[oberdiek.luatex.kpse_module_loader] Search failed"
506 end
507 table.insert(package.loaders, kpse_module_loader)
508 </lua>

```

## 3 Test

```

509 <test2>
510 \documentclass{article}
511 \def\LoadCommand{%
512   \RequirePackage{luatex}[2009/12/02]%
513 }

```



```

514 </test2>
515 <*test3>
516 \documentclass{article}
517 \def\LoadCommand{%
518   \RequirePackage{luatex-loader}[2009/12/02]%
519 }
520 </test3>

```

### 3.1 Catcode checks for loading

```

521 <*test1>
522 \catcode'\{=1 %
523 \catcode'\}=2 %
524 \catcode'\#=6 %
525 \catcode'\@=11 %
526 \expandafter\ifx\csname count@\endcsname\relax
527   \countdef\count@=255 %
528 \fi
529 \expandafter\ifx\csname @gobble\endcsname\relax
530   \long\def\@gobble#1{%
531 \fi
532 \expandafter\ifx\csname @firstofone\endcsname\relax
533   \long\def\@firstofone#1{#1}%
534 \fi
535 \expandafter\ifx\csname loop\endcsname\relax
536   \expandafter\@firstofone
537 \else
538   \expandafter\@gobble
539 \fi
540 {%
541   \def\loop#1\repeat{%
542     \def\body{#1}%
543     \iterate
544   }%
545   \def\iterate{%
546     \body
547     \let\next\iterate
548   \else
549     \let\next\relax
550   \fi
551   \next
552 }%
553 \let\repeat=\fi
554 }%
555 \def\RestoreCatcodes{}
556 \count@=0 %
557 \loop
558   \edef\RestoreCatcodes{%
559     \RestoreCatcodes
560     \catcode\the\count@=\the\catcode\count@\relax
561   }%
562 \ifnum\count@<255 %
563   \advance\count@ 1 %
564 \repeat
565
566 \def\RangeCatcodeInvalid#1#2{%
567   \count@=#1\relax
568   \loop
569     \catcode\count@=15 %
570   \ifnum\count@<#2\relax
571     \advance\count@ 1 %
572   \repeat
573 }

```

```

574 \expandafter\ifx\csname LoadCommand\endcsname\relax
575   \def\LoadCommand{\input luatex.sty\relax}%
576 \fi
577 \def\Test{%
578   \RangeCatcodeInvalid{0}{47}%
579   \RangeCatcodeInvalid{58}{64}%
580   \RangeCatcodeInvalid{91}{96}%
581   \RangeCatcodeInvalid{123}{255}%
582   \catcode'\@=12 %
583   \catcode'\=0 %
584   \catcode'\{=1 %
585   \catcode'\}=2 %
586   \catcode'\#=6 %
587   \catcode'\[=12 %
588   \catcode'\]=12 %
589   \catcode'\%=14 %
590   \catcode'\ =10 %
591   \catcode13=5 %
592   \LoadCommand
593   \RestoreCatcodes
594 }
595 \Test
596 \csname @@end\endcsname
597 \end
598 </test1>

```

## 3.2 Catcode tables

### 3.2.1 Predefined catcode tables

```

599 <*test4>
600 \NeedsTeXFormat{LaTeX2e}

```

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

```

601 \count0=0 %
602 \chardef\TestLaTeX=1000 %
603 \chardef\TestMax=300 %
604 \loop
605   \count\numexpr\TestLaTeX+\count0\relax=\catcode\count0 %
606   \ifnum\count0<\TestMax
607     \advance\count0 by 1 %
608   \repeat
609 \documentclass{minimal}
610 \usepackage{luatex}[2009/12/02]
611 \usepackage{qstest}
612 \IncludeTests{*}
613 \LogTests{log}{*}{*}
614 \makeatletter
615 \def\Check#1{%
616   \Expect*{\the\count@=\the\catcode\count@}%
617   *{\the\count@=#1}%
618 }
619 \newcount\scratch
620 \def\Test#1#2{%
621   \begin{qstest}{CatcodeTable#1}{CatcodeTable#1}%
622     \catcodetable\csname CatcodeTable#1\endcsname
623     \count@=\z@
624     \loop
625       \scratch=#2\relax
626       \Expect*{\the\count@=\the\catcode\count@}%
627       *{\the\count@=\the\scratch}%
628     \ifnum\count@<\TestMax
629       \advance\count@\@ne
630     \repeat

```

```

631 \end{qstest}}%
632 }
633 \Test{LaTeX}{\the\count\numexpr\TestLaTeX+\count@}
634 \Test{String}{\ifnum\count@=32 10\else 12\fi}
635 \Test{Other}{12}
636 \initcatcodetable99 %
637 \Test{IniTeX}{%
638   0\relax
639   \begingroup
640     \catcodetable99 %
641     \global\scratch=\the\catcode\count@
642   \endgroup
643 }

```

### 3.2.2 Catcode table number stack

```

644 \begin{qstest}{CatcodeTableNumStack}{CatcodeTableNumStack}
645   \def\TestStack#1{%
646     \Expect*{\LuT@NumStack}{#1}%
647   }%
648   \TestStack{0}%
649   \PushCatcodeTableNumStack
650   \TestStack{{0}0}%
651   \@firstofone{%
652     \begingroup
653       \initcatcodetable12 %
654       \catcodetable12 %
655       \PushCatcodeTableNumStack
656       \TestStack{{12}{0}0}%
657       \PopCatcodeTableNumStack
658       \TestStack{{0}0}%
659       \PopCatcodeTableNumStack
660       \TestStack{0}%
661       \def\TestWarning{Missing empty stack warning}%
662       \def\@PackageWarning#1#2{\def\TestWarning{empty stack}}%
663       \PopCatcodeTableNumStack
664       \TestStack{0}%
665       \Expect*{\TestWarning}{empty stack}%
666     \endgroup
667   }%
668 \end{qstest}

```

### 3.2.3 Catcode table stack

```

669 \begin{qstest}{CatcodeTableStack}{CatcodeTableStack}
670   \def\TestStack#1{%
671     \Expect*{\the\CatcodeTableStack}{#1}%
672   }%
673   \TestStack{0}%
674   \IncCatcodeTableStack
675   \TestStack{2}%
676   \IncCatcodeTableStack
677   \TestStack{4}%
678   \begingroup
679     \IncCatcodeTableStack
680     \TestStack{6}%
681   \endgroup
682   \TestStack{6}%
683   \begingroup
684     \DecCatcodeTableStack
685     \TestStack{4}%
686   \endgroup
687   \TestStack{4}%
688   \DecCatcodeTableStack
689   \TestStack{2}%

```

```

690 \DecCatcodeTableStack
691 \TestStack{0}%
692 \begingroup
693   \def\TestError{Missing error}%
694   \def\@PackageError#1#2#3{%
695     \def\TestError{Empty stack}%
696   }%
697   \DecCatcodeTableStack
698   \TestStack{0}%
699   \Expect*{\TestError}{Empty stack}%
700 \endgroup
701 \end{qstest}

```

### 3.2.4 Catcode regime macros

```

702 \begin{qstest}{CatcodeRegime}{CatcodeRegime}
703   \def\TestStacks#1#2#3{%
704     \Expect*{\the\catcodetable}{#1}%
705     \Expect*{\the\CatcodeTableStack}{#2}%
706     \Expect*{\LuT@NumStack}{#3}%
707   }%
708   \TestStacks{0}{0}{0}%
709   \catcode'\|=7 %
710   \BeginCatcodeRegime\CatcodeTableLaTeX
711     \TestStacks{2}{2}{0}%
712     \Expect*{\the\catcode'\|}{12}%
713   \EndCatcodeRegime
714   \TestStacks{0}{0}{0}%
715   \Expect*{\the\catcode'\|}{7}%
716 \end{qstest}

```

### 3.3 Attribute allocation

```

717 \begin{qstest}{Attributes}{Attributes}
718   \newattribute\TestAttr
719   \Expect*{\meaning\TestAttr}%
720     *{\string\attribute\number\allocationnumber}%
721   \Expect*{\the\allocationnumber}{0}%
722   \begingroup
723     \newattribute\TestAttr
724     \Expect*{\the\allocationnumber}{1}%
725   \endgroup
726   \Expect*{\the\allocationnumber}{0}%
727   \Expect*{\meaning\TestAttr}*{\string\attribute1}%
728   \Expect*{\the\TestAttr}*{\number\LuT@UnsetAttributeValue}%
729   \def\Test#1{%
730     \setattribute\TestAttr{#1}%
731     \Expect*{\the\TestAttr}{#1}%
732   }%
733   \Test{0}%
734   \Test{1}%
735   \Test{-1}%
736   \Test{123}%
737   \unsetattribute\TestAttr
738   \Expect*{\the\TestAttr}*{\number\LuT@UnsetAttributeValue}%
739   \begingroup
740     \Expect*{\the\TestAttr}*{\number\LuT@UnsetAttributeValue}%
741     \Test{1234}%
742   \endgroup
743   \Expect*{\the\TestAttr}*{\number\LuT@UnsetAttributeValue}%
744 \end{qstest}

```

### 3.4 Lua states

```

745 \begin{qstest}{LuaState}{LuaState}

```

```

746 \newluastate\TestLuaState
747 \Expect*{\number\TestLuaState}{1}%
748 \newluastate\TestLuaState
749 \Expect*{\number\TestLuaState}{2}%
750 \end{qstest}

751 \@@end
752 \</test4>

```

### 3.5 Short test for plain-TeX

```

753 (*test5)
754 \input luatex.sty\relax
755 \newluastate\TestLuaState
756 \newattribute\TestAttr
757 \setattribute\TestAttr{10}
758 \unsetattribute\TestAttr
759 \newcatcodetable\TestCTa
760 \begingroup
761   \SetCatcodeRange{'A'}{'Z'}{12}%
762 \endgroup
763 \BeginCatcodeRegime\CatcodeTableLaTeX
764 \EndCatcodeRegime
765 \end
766 \</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 TeX Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

### 4.2 Bundle installation

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

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

**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/
```

---

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

### 4.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain- $\text{\TeX}$ :

```
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):

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

### 4.5 Some details for the interested

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

```
pdftk luatex.pdf unpack_files output .
```

**Unpacking with  $\text{\LaTeX}$ .** The `.dtx` chooses its action depending on the format:

**plain- $\text{\TeX}$ :** Run `docstrip` and extract the files.

**$\text{\LaTeX}$ :** Generate the documentation.

If you insist on using  $\text{\LaTeX}$  for `docstrip` (really, `docstrip` does not need  $\text{\LaTeX}$ ), 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 $\text{\LaTeX}$` :

```
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.

[2009/04/10 v0.2]

- Requires package `ifluatex` in version 2.0 to ensure `\luatexversion`.
- Updates the call of `\directlua`, the syntax has changed in L<sup>A</sup>T<sub>E</sub>X 0.36.

[2009/12/02 v0.3]

- Unsetting of attributes updated for L<sup>A</sup>T<sub>E</sub>X 0.37.

## 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

<code>\#</code> .....	524, 586	<code>\_</code> .....	590
<code>\%</code> .....	589		
<code>\:</code> .....	187, 188, 189, 190, 191, 194		
<code>\@</code> .....	525, 582		
<code>\@end</code> .....	751		
<code>\@INTERNAL@ERROR</code> .....	226		
<code>\@PackageError</code> .....	113, 210, 291, 300, 694		
<code>\@PackageWarning</code> .....	369, 662		
<code>\@PackageWarningNoLine</code> .....	126		
<code>\@car</code> .....	159, 372		
<code>\@cdr</code> .....	162, 374		
<code>\@ehc</code> .....	115, 213		
<code>\@ehd</code> .....	293, 302		
<code>\@empty</code> .....	147		
<code>\@firstofone</code> ..	153, 329, 533, 536, 651		
<code>\@firstoftwo</code> .....	156, 167		
<code>\@gobble</code> .....	150, 530, 538		
<code>\@ifnch</code> .....	173, 175, 192		
<code>\@ifnextchar</code> .....	167, 169		
<code>\@ifstar</code> .....	165		
<code>\@let@token</code> .....	173, 176, 179, 192		
<code>\@makeoother</code> .....	328, 348, 349, 350, 351, 352, 356		
<code>\@ne</code> .....	237, 249, 318, 629		
<code>\@nil</code> .....	160, 163, 372, 374		
<code>\@sptoken</code> .....	176, 187		
<code>\@tempcnta</code> .....	196, 307, 312, 316, 317, 318		
<code>\@tempcntb</code> .....	199, 308, 313, 316		
<code>\@undefined</code> .....	52, 205, 448		
<code>\@xifnch</code> .....	177, 190		
<code>\[</code> .....	587		
<code>\]</code> .....	583		
<code>\{</code> .....	522, 584		
<code>\}</code> .....	523, 585		
<code>\]</code> .....	588		
<code>\ </code> .....	709, 712, 715		
		<b>A</b>	
		<code>\advance</code> .....	237, 249, 277, 289, 298, 318, 563, 571, 607, 629
		<code>\aftergroup</code> .....	26, 422
		<code>\allocationnumber</code> .....	238, 239, 240, 250, 251, 253, 278, 279, 281, 720, 721, 724, 726
		<code>\attribute</code> .....	253, 255, 720, 727
		<code>\attributedef</code> .....	251
		<b>B</b>	
		<code>\begin</code> ....	621, 644, 669, 702, 717, 745
		<code>\BeginCatcodeRegime</code> ..	5, 378, 710, 763
		<code>\body</code> .....	542, 546
		<b>C</b>	
		<code>\catcode</code> ..	3, 4, 5, 6, 7, 8, 9, 17, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 64, 65, 68, 69, 70, 71, 75, 76, 77, 78, 82, 84, 317, 328, 333, 334, 335, 337, 338, 339, 340, 341, 342, 343, 344, 345, 399, 400, 401, 402, 403, 404, 405, 413, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 522, 523, 524, 525, 560, 569, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 605, 616, 626, 641, 709, 712, 715
		<code>\catcodetable</code> ..	281, 284, 330, 364, 370, 372, 380, 383, 622, 640, 654, 704
		<code>\CatcodeTableIniTeX</code> ..	4, 322, 326, 330
		<code>\CatcodeTableLaTeX</code> ..	325, 346, 710, 763
		<code>\CatcodeTableOther</code> .....	324, 357

<code>\CatcodeTableStack</code>	4, 272, 273, 288, 289, 297, 298, 382, 383, 671, 705
<code>\CatcodeTableString</code>	323, 355
<code>\chardef</code>	239, 275, 279, 602, 603
<code>\Check</code>	615
<code>\count</code>	131, 132, 601, 605, 606, 607, 633
<code>\count@</code>	309, 314, 317, 527, 556, 560, 562, 563, 567, 569, 570, 571, 616, 617, 623, 626, 627, 628, 629, 633, 634, 641
<code>\countdef</code>	527
<code>\csname</code>	10, 18, 44, 60, 67, 104, 120, 125, 147, 150, 153, 156, 159, 162, 165, 196, 197, 199, 200, 203, 390, 406, 414, 440, 456, 526, 529, 532, 535, 574, 596, 622
<b>D</b>	
<code>\DecCatcodeTableStack</code>	296, 386, 684, 688, 690, 697
<code>\directlua</code>	473, 475
<code>\documentclass</code>	510, 516, 609
<b>E</b>	
<code>\empty</code>	13, 14, 409, 410
<code>\end</code>	597, 631, 668, 701, 716, 744, 750, 765
<code>\EndCatcodeRegime</code>	385, 713, 764
<code>\endcsname</code>	10, 18, 44, 60, 67, 104, 120, 125, 147, 150, 153, 156, 159, 162, 165, 196, 197, 199, 200, 203, 390, 406, 414, 440, 456, 526, 529, 532, 535, 574, 596, 622
<code>\endinput</code>	26, 117, 422
<code>\endlinechar</code>	471
<code>\errmessage</code>	242, 255, 284
<code>\Expect</code>	616, 626, 646, 665, 671, 699, 704, 705, 706, 712, 715, 719, 721, 724, 726, 727, 728, 731, 738, 740, 743, 747, 749
<b>F</b>	
<code>\futurelet</code>	173, 192
<b>G</b>	
<code>\globbox</code>	145
<code>\globcount</code>	142
<code>\globdimen</code>	143
<code>\globskip</code>	144
<b>I</b>	
<code>\ifcase</code>	217
<code>\ifluatex</code>	111
<code>\ifnum</code>	131, 236, 248, 261, 275, 288, 297, 316, 472, 562, 570, 606, 628, 634
<code>\ifx</code>	11, 14, 18, 44, 52, 55, 104, 120, 125, 147, 150, 153, 156, 159, 162, 165, 176, 179, 196, 199, 203, 205, 208, 216, 368, 390, 407, 410, 414, 440, 448, 451, 526, 529, 532, 535, 574
<code>\immediate</code>	20, 46, 416, 442
<code>\IncCatcodeTableStack</code>	287, 381, 674, 676, 679
<code>\IncludeTests</code>	612
<code>\initcatcodetable</code>	326, 636, 653
<code>\input</code>	105, 106, 391, 575, 754
<code>\iterate</code>	543, 545, 547
<b>L</b>	
<code>\LoadCommand</code>	511, 517, 575, 592
<code>\LogTests</code>	613
<code>\loop</code>	315, 541, 557, 568, 604, 624
<code>\luastate</code>	240, 242
<code>\luatexversion</code>	261, 472
<code>\LuT@AllocAttribute</code>	245, 248, 249, 250
<code>\LuT@AllocCatcodeTable</code>	270, 275, 277, 278
<code>\LuT@AllocLuaState</code>	233, 236, 237, 238
<code>\LuT@AtEnd</code>	80, 81, 116, 395
<code>\LuT@newcommand</code>	202, 235, 247, 258, 262, 265, 267, 274, 287, 296, 305, 362, 367, 378, 385
<code>\LuT@NumStack</code>	361, 363, 364, 368, 372, 373, 374, 646, 706
<code>\LuT@NumStackEmpty</code>	360, 361, 368
<code>\LuT@temp</code>	130, 135, 136, 137, 138, 139, 140, 141, 187, 194, 306, 320
<code>\LuT@UnsetAttributeValue</code>	262, 263, 265, 268, 728, 738, 740, 743
<b>M</b>	
<code>\m@ne</code>	246, 263, 271
<code>\makeatletter</code>	614
<code>\meaning</code>	719, 727
<code>\MessageBreak</code>	211
<b>N</b>	
<code>\n</code>	502, 505
<code>\NeedsTeXFormat</code>	600
<code>\newattribute</code>	3, 247, 718, 723, 756
<code>\newbox</code>	145
<code>\newcatcodetable</code>	4, 274, 322, 323, 324, 325, 759
<code>\newcommand</code>	231
<code>\newcount</code>	142, 233, 245, 270, 272, 619
<code>\newdimen</code>	143
<code>\newluastate</code>	3, 235, 746, 748, 755
<code>\newskip</code>	144
<code>\next</code>	547, 549, 551
<code>\number</code>	720, 728, 738, 740, 743, 747, 749
<code>\numexpr</code>	259, 312, 313, 314, 380, 605, 633
<b>P</b>	
<code>\PackageInfo</code>	23, 419
<code>\PopCatcodeTableNumStack</code>	367, 387, 657, 659, 663
<code>\ProvidesPackage</code>	15, 61, 411, 457
<code>\PushCatcodeTableNumStack</code>	5, 362, 379, 649, 655
<b>R</b>	
<code>\RangeCatcodeInvalid</code>	566, 578, 579, 580, 581
<code>\repeat</code>	319, 541, 553, 564, 572, 608, 630



