

# The ltxcmds package

Heiko Oberdiek  
<heiko.oberdiek at googlemail.com>

2011/03/16 v1.18

## Abstract

The package `ltxcmds` exports some utility macros from the L<sup>A</sup>T<sub>E</sub>X kernel into a separate namespace and also provides them for other formats such as plain-T<sub>E</sub>X.

## Contents

<b>1</b>	<b>Documentation</b>	<b>2</b>
1.1	Introduction . . . . .	2
1.2	Numbers . . . . .	3
1.3	Argument killers . . . . .	3
1.4	Argument grabbers . . . . .	3
1.5	List helpers . . . . .	4
1.6	Tail recursion . . . . .	4
1.7	Empty macro . . . . .	4
1.8	Characters . . . . .	4
1.9	Boolean switch . . . . .	5
1.10	Command definitions . . . . .	5
1.11	Stripping . . . . .	5
1.12	File management . . . . .	6
	1.12.1 File extensions . . . . .	6
	1.12.2 Load check . . . . .	6
	1.12.3 Version date check . . . . .	6
1.13	Macro additions . . . . .	7
1.14	Macro <code>\ltx@ifnextchar</code> . . . . .	7
1.15	<code>\ltx@leavevmode</code> , <code>\ltx@mbbox</code> . . . . .	7
1.16	Expandable test for emptiness . . . . .	7
1.17	Stripping spaces . . . . .	7
1.18	Check for emptiness of boxes . . . . .	8
<b>2</b>	<b>Implementation</b>	<b>8</b>
2.1	Identification . . . . .	8
2.2	Numbers . . . . .	10
2.3	Argument killers . . . . .	10
2.4	Argument grabbers . . . . .	11
2.5	List helpers . . . . .	12
2.6	Tail recursion . . . . .	13
2.7	Empty macro . . . . .	13
2.8	Characters . . . . .	13
2.9	Boolean switch . . . . .	14
2.10	Command definitions . . . . .	15
2.11	Stripping . . . . .	15
2.12	File management . . . . .	16
	2.12.1 File extensions . . . . .	16

2.12.2	Load check	16
2.12.3	Version date check	16
2.13	Macro additions	18
2.14	Macro <code>\ltx@ifnextchar</code>	18
2.15	<code>\ltx@leavevmode</code> , <code>\ltx@mbox</code>	19
2.16	Help macros	19
2.17	Expandable test for emptiness	20
2.17.1	Vanilla $\TeX$	20
2.17.2	With <code>\detokenize</code>	21
2.17.3	<code>\ltx@ifblank</code>	21
2.18	<code>\ltx@zapspace</code>	21
2.19	<code>\ltx@ifBoxEmpty</code>	21
<b>3</b>	<b>Test</b>	<b>23</b>
3.1	Catcode checks for loading	23
3.2	Test <code>\ltx@GobbleNum</code>	24
3.3	Test <code>\ltx@ifempty</code>	27
3.4	Test <code>\ltx@zap@space</code>	29
3.5	Test <code>\ltx@ifBoxEmpty</code>	30
<b>4</b>	<b>Installation</b>	<b>32</b>
4.1	Download	32
4.2	Bundle installation	32
4.3	Package installation	32
4.4	Refresh file name databases	33
4.5	Some details for the interested	33
<b>5</b>	<b>References</b>	<b>33</b>
<b>6</b>	<b>History</b>	<b>34</b>
[2009/08/05 v1.0]		34
[2009/12/12 v1.1]		34
[2010/01/28 v1.2]		34
[2010/03/01 v1.3]		34
[2010/03/09 v1.4]		34
[2010/04/08 v1.5]		34
[2010/04/16 v1.6]		35
[2010/04/26 v1.7]		35
[2010/09/11 v1.8]		35
[2010/10/25 v1.9]		35
[2010/10/31 v1.10]		35
[2010/11/12 v1.11]		35
[2010/12/02 v1.12]		35
[2010/12/04 v1.13]		35
[2010/12/07 v1.14]		35
[2010/12/12 v1.15]		35
[2011/02/04 v1.16]		35
[2011/02/05 v1.17]		35
[2011/03/16 v1.18]		36
<b>7</b>	<b>Index</b>	<b>36</b>

# 1 Documentation

## 1.1 Introduction

Many of my packages also support other formats such as plain- $\TeX$ . Because I am rather familiar with the utility macros from  $\LaTeX$ 's kernel (e.g. `\@gobble`,

\@firstoftwo), I found myself rewriting them again and again, because they are lacking in plain- $\TeX$ .

Therefore this package provides often used macros and similar ones with the name prefix \ltx@. This avoids also faulty redefinitions. I remember an example where a package redefined \@firstoftwo with forgetting \long.

## 1.2 Numbers

\ltx@zero	→	0
\ltx@one	→	1
\ltx@two	→	2
\ltx@ccclv	→	255
\ltx@minusone	→	-1

These commands are numbers 0, 1, 2, 255 and -1. They are not digits and a space is not gobbled afterwards. Macro \ltx@minusone is available since version 2010/12/12 v1.15.

## 1.3 Argument killers

\ltx@gobble {⟨1⟩}	→
\ltx@gobbletwo {⟨1⟩} {⟨2⟩}	→
\ltx@gobblethree {⟨1⟩} {⟨2⟩} {⟨3⟩}	→
\ltx@gobblefour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}	→

\ltx@GobbleNum {⟨num⟩} {⟨1⟩} {⟨2⟩} ... {⟨⟨num⟩⟩}	→
--	---

The first argument ⟨num⟩ of macro \ltx@GobbleNum specifies, how many following arguments are eaten. Macro \ltx@GobbleNum is expandable in exact two expansion steps.

## 1.4 Argument grabbers

\ltx@firstofone {⟨1⟩}	→	⟨1⟩
\ltx@firstoftwo {⟨1⟩} {⟨2⟩}	→	⟨1⟩
\ltx@secondoftwo {⟨1⟩} {⟨2⟩}	→	⟨2⟩
\ltx@firstofthree {⟨1⟩} {⟨2⟩} {⟨3⟩}	→	⟨1⟩
\ltx@secondofthree {⟨1⟩} {⟨2⟩} {⟨3⟩}	→	⟨2⟩
\ltx@thirdofthree {⟨1⟩} {⟨2⟩} {⟨3⟩}	→	⟨3⟩
\ltx@firstoffour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}	→	⟨1⟩
\ltx@secondoffour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}	→	⟨2⟩
\ltx@thirdoffour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}	→	⟨3⟩
\ltx@fourthoffour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}	→	⟨4⟩

Macros \ltx@firstofthree, \ltx@secondofthree and \ltx@thirdofthree were added in version 2010/11/12 v1.11. Macros \ltx@firstoffour, ..., \ltx@fourthoffour were added in version 2011/02/04 v1.16.

## 1.5 List helpers

<code>\ltx@carzero ... \@nil</code>	$\rightarrow$
<code>\ltx@cdrzero ... \@nil</code>	$\rightarrow \dots$

<code>\ltx@car {\langle 1 \rangle} ... \@nil</code>	$\rightarrow \langle 1 \rangle$
<code>\ltx@cdr {\langle 1 \rangle} ... \@nil</code>	$\rightarrow \dots$

<code>\ltx@cartwo {\langle 1 \rangle} {\langle 2 \rangle} ... \@nil</code>	$\rightarrow \langle 1 \rangle \langle 2 \rangle$
<code>\ltx@cdrtwo {\langle 1 \rangle} {\langle 2 \rangle} ... \@nil</code>	$\rightarrow \dots$

<code>\ltx@carthree {\langle 1 \rangle} {\langle 2 \rangle} {\langle 3 \rangle} ... \@nil</code>	$\rightarrow \langle 1 \rangle \langle 2 \rangle \langle 3 \rangle$
<code>\ltx@cdrthree {\langle 1 \rangle} {\langle 2 \rangle} {\langle 3 \rangle} ... \@nil</code>	$\rightarrow \dots$

<code>\ltx@carfour {\langle 1 \rangle} {\langle 2 \rangle} {\langle 3 \rangle} {\langle 4 \rangle} ... \@nil</code>	$\rightarrow \langle 1 \rangle \langle 2 \rangle \langle 3 \rangle \langle 4 \rangle$
<code>\ltx@cdrfour {\langle 1 \rangle} {\langle 2 \rangle} {\langle 3 \rangle} {\langle 4 \rangle} ... \@nil</code>	$\rightarrow \dots$

<code>\ltx@CarNum {\langle num \rangle} {\langle 1 \rangle} ... {\langle \langle num \rangle \rangle} {\langle \langle num \rangle + 1 \rangle} ... \@nil</code>	$\rightarrow \{\langle 1 \rangle\} \dots \{\langle \langle num \rangle \rangle\} \dots$
<code>\ltx@CdrNum {\langle num \rangle} {\langle 1 \rangle} ... {\langle \langle num \rangle \rangle} {\langle \langle num \rangle + 1 \rangle} ... \@nil</code>	$\rightarrow \{\langle \langle num \rangle + 1 \rangle\} \dots$

Macros `\ltx@CarNum` and `\ltx@CdrNum` are expandable in exact two expansion steps.

## 1.6 Tail recursion

<code>\ltx@ReturnAfterFi {\langle 1 \rangle} \fi</code>	$\rightarrow \backslash fi \langle 1 \rangle$
<code>\ltx@ReturnAfterElseFi {\langle 1 \rangle} \else {\langle 2 \rangle} \fi</code>	$\rightarrow \backslash fi \langle 1 \rangle$

## 1.7 Empty macro

<code>\ltx@empty</code>	$\rightarrow$
-------------------------	---------------

## 1.8 Characters

<code>\ltx@space</code>	$\rightarrow$	<code>␣</code>
<code>\ltx@percentchar</code>	$\rightarrow$	<code>%</code>
<code>\ltx@backslashchar</code>	$\rightarrow$	<code>\</code>
<code>\ltx@hashchar</code>	$\rightarrow$	<code>#</code> (since v1.7)
<code>\ltx@leftbracechar</code>	$\rightarrow$	<code>{</code> (since v1.8)
<code>\ltx@rightbracechar</code>	$\rightarrow$	<code>}</code> (since v1.8)

## 1.9 Boolean switch

`\ltx@newif {⟨cmd⟩}`

`\ltx@newif` defines a new boolean switch `⟨cmd⟩` like `\newif`. Unlike plain  $\text{\TeX}$ 's `\newif`, `\ltx@newif` is not `\outer`. The command `⟨cmd⟩` must start with the two characters `if`.

`\ltx@newglobalif {⟨cmd⟩}`

`\ltx@newglobalif` defines a new boolean switch `⟨cmd⟩` like `\ltx@newif`. However the switch setting commands, `⟨cmd⟩` without the prefix `if` and followed by `true` or `false` are acting globally.

## 1.10 Command definitions

`\ltx@ifundefined {⟨cmd⟩} {⟨yes⟩} {⟨no⟩}`

If  $\varepsilon\text{-}\text{\TeX}$  is available, `\ifcsname` is used that does not have the side effect of defining undefined commands with meaning of `\relax`. This command is always expandable. Change in version 1.1: Also the meaning `\relax` is always considered “undefined”.

`\ltx@ifUndefined {⟨cmd⟩} {⟨yes⟩} {⟨no⟩}`

If  $\varepsilon\text{-}\text{\TeX}$  is available, `\ifcsname` is used that does not have the side effect of defining undefined commands with meaning of `\relax`. Also it always checks for the meaning of `\relax` and considers this as undefined. This macro is not expandable without  $\varepsilon\text{-}\text{\TeX}$ .

`\ltx@LocalExpandAfter`

It expands the token after the next token but in a local context. That is the difference to `\expandafter`. The local context discards the side effect of `\csname` and let the command undefined after the expansion step.

## 1.11 Stripping

`\ltx@RemovePrefix`  
`\ltx@StripPrefix`

All tokens up to and including the next available character ‘>’ are thrown away. Usually it is used to strip the first part of the output of the commands `\meaning` or `\pdflastmatch`. Macro `\ltx@RemovePrefix` has the same meaning as  $\text{\LaTeX}$ 's `\strip@prefix`, whereas macro `\ltx@StripPrefix` expands the next token once before stripping the prefix.

`\ltx@onelevel@sanitize {⟨macro⟩}`

Macro `\ltx@onelevel@sanitize` provides  $\text{\LaTeX}$ 's `\@onelevel@sanitize`. The macro is expanded once and the contents is converted to characters with catcode 12 (other) and space tokens with catcode 10 (space). Then the sanitized contents is stored into the macro again. Since version 1.12.

## 1.12 File management

All macros in this section are expandable like the counterparts of the L<sup>A</sup>T<sub>E</sub>X kernel. Also they can be used after the preamble.

### 1.12.1 File extensions

<code>\ltx@clsextension</code>
<code>\ltx@pkgextension</code>

Macros `\ltx@clsextension` and `\ltx@styextension` stores the strings `cls` and `sty`. In opposite to L<sup>A</sup>T<sub>E</sub>X's `\@clsextension` and `\@styextension` they can also be used after `\begin{document}`.

### 1.12.2 Load check

<code>\ltx@ifclassloaded {&lt;class&gt;} {&lt;yes&gt;} {&lt;no&gt;}</code>
<code>\ltx@ifpackageloaded {&lt;package&gt;} {&lt;yes&gt;} {&lt;no&gt;}</code>

Macros `\ltx@ifclassloaded`/`\ltx@ifpackageloaded` execute `<yes>`, if the `<class>` or `<package>` is loaded, otherwise `<no>` is called. Both `<class>` and `<package>` are specified without extension. The macros can also be used after `\begin{document}`.

<code>\ltx@iffileloaded {&lt;file&gt;} {&lt;yes&gt;} {&lt;no&gt;}</code>
--

If L<sup>A</sup>T<sub>E</sub>X's `\ProvidesFile` macro was called before using `<file>` as argument, then `\ltx@iffileloaded` calls `<yes>`, otherwise `<no>`. Therefore it is possible that the `<file>` is loaded, but `<no>` is executed because of a missing `\ProvidesFile`. The L<sup>A</sup>T<sub>E</sub>X kernel does not have a counterpart of `\ltx@iffileloaded`.

Note that the file name used in `\ProvidesFile` and `\ltx@iffileloaded` must match. For example, if T<sub>E</sub>X's default extension `.tex` was given in the first command, then it must also specified in the latter command and vice versa.

### 1.12.3 Version date check

<code>\ltx@ifclasslater {&lt;class&gt;} {&lt;date&gt;} {&lt;yes&gt;} {&lt;no&gt;}</code>
<code>\ltx@ifpackagelater {&lt;package&gt;} {&lt;date&gt;} {&lt;yes&gt;} {&lt;no&gt;}</code>
<code>\ltx@iffilelater {&lt;file&gt;} {&lt;date&gt;} {&lt;yes&gt;} {&lt;no&gt;}</code>

If a `\ProvidesClass`/`\ProvidesPackage`/`\ProvidesFile` command with exact the same class/package/file was executed before with an optional argument that starts with a L<sup>A</sup>T<sub>E</sub>X version date, then this version date is compared with the argument `<date>`. If they are equal or if the version date is the later date, then `<yes>` is called. In all other cases `<no>` is executed.

A L<sup>A</sup>T<sub>E</sub>X date has the format `YYYY/MM/DD` with `YYYY` as year with four digits, `MM` as month with two digits and `DD` as day with two digits. If pdfT<sub>E</sub>X's `\pdfmatch` is available, then it is used to detect the version date, to reject invalid date formats and to reject some invalid dates. Dates before 1994/01/01 are always invalid, because version dates are introduced with L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> in 1994.

### 1.13 Macro additions

`\ltx@GlobalAppendToMacro {<cmd>} {<addition>}`  
`\ltx@LocalAppendToMacro {<cmd>} {<addition>}`

The `<addition>` is appended to the parameterless macro `<cmd>`. If `<cmd>` is undefined or has the meaning `\relax`, then it will be initialized as empty macro before. Due to a bug `<addition>` must not contain `\par` before version 2010/10/25 v1.9.

### 1.14 Macro `\ltx@ifnextchar`

`\ltx@ifnextchar {<char>} {<yes>} {<no>}`

If next character is `<char>` then `<yes>` is called, otherwise `<no>`. The character is not removed.

### 1.15 `\ltx@leavevmode`, `\ltx@mbox`

`\ltx@leavevmode`

Macro `\ltx@leavevmode` calls pdfTeX's `\quitvmode`. Otherwise `\leavevmode` is used and defined if it is necessary.

`\ltx@mbox`

Macro `\ltx@mbox` reimplements `\mbox` with two changes. Instead of `\leavevmode` it uses `\ltx@leavevmode` and stops right after `\hbox`. Especially it does not grab the argument and allows the extended syntax of `\hbox`.

### 1.16 Expandable test for emptiness

`\ltx@ifempty {<stuff>} {<yes>} {<no>}`

Macro `\ltx@ifempty` checks in exact two expansion steps whether `<stuff>` is empty or contains token. Depending on the result `<yes>` or `<no>` is executed. The token in `<stuff>` may contain `\par` and unmatched conditionals (`\if`, `\else`, `\fi`, ...). Since version 2010/11/12 v1.11.

`\ltx@ifblank {<stuff>} {<yes>} {<no>}`

Macro `\ltx@ifblank` tests in exact two expansion steps if `<stuff>` is empty or contain only blank spaces. In this case argument `<yes>` is called. If `<stuff>` contains other tokens than spaces then `<no>` is executed. Since version 2010/12/04 v1.13.

### 1.17 Stripping spaces

`\ltx@zapspace {<stuff>}`

Macro `\ltx@zapspace` strips spaces from `<stuff>` that are not hidden inside curly braces. Like L<sup>A</sup>T<sub>E</sub>X's `\zap@space` it is expandable. Differences:

- Syntax: `\zap@space` also expects a space token and `\@empty` after `<stuff>`.
- Macro `\ltx@zapspace` is expandable in exact two expansion steps.

- Macro `\ltx@zapspace` always retains curly braces.
- Macro `\zap@space` has a bug. It stops stripping spaces after a token group in curly braces if the first two tokens inside the group are equal.
- Macro `\ltx@zapspace` also works with `\par` and conditionals (`\if`, `\else`, `\fi`, ...).

Macro `\ltx@zapspace` is available since version 2010/12/07 v1.14.

## 1.18 Check for emptiness of boxes

`\ltx@ifboxempty {<box register number>} {<yes>} {<no>}`

Macro `\ltx@ifboxempty` calls `<yes>` if the box exists (`\ifvoid` returns false) and the box does not contain any content. Otherwise if the box is void or contains something, then `<no>` is executed. Thus being empty means that the box exists and is either an `\hbox` or a `\vbox` and may even have dimensions other than 0.0 pt, but the box does not contain anything. Macro `\ltx@ifboxempty` is available since 2010/02/04 v1.16.

`\ltx@ifboxvoidoreempty {<box register number>} {<yes>} {<no>}`

Macro `\ltx@ifboxvoidoreempty` calls `<yes>` if the box is either void or does not contain any content. Otherwise `<no>` is executed. Macro `\ltx@ifboxvoidoreempty` is available since 2010/02/04 v1.16.

## 2 Implementation

### 2.1 Identification

```
1 <*package>
```

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

```
2 \begingroup\catcode61\catcode48\catcode32=10\relax%
3 \catcode13=5 % ^^M
4 \endlinechar=13 %
5 \catcode35=6 % #
6 \catcode39=12 % '
7 \catcode44=12 % ,
8 \catcode45=12 % -
9 \catcode46=12 % .
10 \catcode58=12 % :
11 \catcode64=11 % @
12 \catcode123=1 % {
13 \catcode125=2 % }
14 \expandafter\let\expandafter\x\csname ver@ltxcmds.sty\endcsname
15 \ifx\x\relax % plain-TeX, first loading
16 \else
17 \def\empty{}%
18 \ifx\x\empty % LaTeX, first loading,
19 % variable is initialized, but \ProvidesPackage not yet seen
20 \else
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{ltxcmds}{The package is already loaded}%
29      \aftergroup\endinput
30    \fi
31  \fi
32 \endgroup%

Package identification:
33 \begingroup\catcode61\catcode48\catcode32=10\relax%
34   \catcode13=5 % ^^M
35   \endlinechar=13 %
36   \catcode35=6 % #
37   \catcode39=12 % '
38   \catcode40=12 % (
39   \catcode41=12 % )
40   \catcode44=12 % ,
41   \catcode45=12 % -
42   \catcode46=12 % .
43   \catcode47=12 % /
44   \catcode58=12 % :
45   \catcode64=11 % @
46   \catcode91=12 % [
47   \catcode93=12 % ]
48   \catcode123=1 % {
49   \catcode125=2 % }
50   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
51     \def\x#1#2#3[#4]{\endgroup
52       \immediate\write-1{Package: #3 #4}%
53       \xdef#1{#4}%
54     }%
55   \else
56     \def\x#1#2[#3]{\endgroup
57       #2[#{#3}]%
58       \ifx#1\@undefined
59         \xdef#1{#3}%
60       \fi
61       \ifx#1\relax
62         \xdef#1{#3}%
63       \fi
64     }%
65   \fi
66 \expandafter\x\csname ver@ltxcmds.sty\endcsname
67 \ProvidesPackage{ltxcmds}%
68 [2011/03/16 v1.18 LaTeX kernel commands for general use (HO)]%

69 \begingroup\catcode61\catcode48\catcode32=10\relax%
70   \catcode13=5 % ^^M
71   \endlinechar=13 %
72   \catcode123=1 % {
73   \catcode125=2 % }
74   \catcode64=11 % @
75   \def\x{\endgroup
76     \expandafter\edef\csname LTXcmds@AtEnd\endcsname{%
77       \endlinechar=\the\endlinechar\relax
78       \catcode13=\the\catcode13\relax
79       \catcode32=\the\catcode32\relax
80       \catcode35=\the\catcode35\relax
81       \catcode61=\the\catcode61\relax
82       \catcode64=\the\catcode64\relax
83       \catcode123=\the\catcode123\relax
84       \catcode125=\the\catcode125\relax
85     }%
86   }%
87 \x\catcode61\catcode48\catcode32=10\relax%
88 \catcode13=5 % ^^M

```

```

89 \endlinechar=13 %
90 \catcode35=6 % #
91 \catcode64=11 % @
92 \catcode123=1 % {
93 \catcode125=2 % }
94 \def\TMP@EnsureCode#1#2{%
95   \edef\LTXcmds@AtEnd{%
96     \LTXcmds@AtEnd
97     \catcode#1=\the\catcode#1\relax
98   }%
99   \catcode#1=#2\relax
100 }
101 \TMP@EnsureCode{36}{3}% $
102 \TMP@EnsureCode{38}{4}% &
103 \TMP@EnsureCode{40}{12}% (
104 \TMP@EnsureCode{41}{12}% )
105 \TMP@EnsureCode{45}{12}% -
106 \TMP@EnsureCode{46}{12}% .
107 \TMP@EnsureCode{47}{12}% /
108 \TMP@EnsureCode{60}{12}% <
109 \TMP@EnsureCode{62}{12}% >
110 \TMP@EnsureCode{91}{12}% [
111 \TMP@EnsureCode{96}{12}% ‘
112 \TMP@EnsureCode{93}{12}% ]
113 \TMP@EnsureCode{94}{12}% ^ (superscript) (!)
114 \TMP@EnsureCode{124}{12}% |
115 \edef\LTXcmds@AtEnd{\LTXcmds@AtEnd\noexpand\endinput}

```

## 2.2 Numbers

```

\ltx@zero
116 \chardef\ltx@zero=0 %

\ltx@one
117 \chardef\ltx@one=1 %

\ltx@two
118 \chardef\ltx@two=2 %

\ltx@active
119 \chardef\ltx@active=13 %

\ltx@ccclv
120 \chardef\ltx@ccclv=255 %

\ltx@minusone
121 \def\ltx@minusone{%
122   -\ltx@one
123 }

```

## 2.3 Argument killers

```

\ltx@gobble
124 \long\def\ltx@gobble#1{}

\ltx@gobbletwo
125 \long\def\ltx@gobbletwo#1#2{}

\ltx@gobblethree
126 \long\def\ltx@gobblethree#1#2#3{}

```

```

\ltx@gobblefour
127 \long\def\ltx@gobblefour#1#2#3#4{}

\ltx@GobbleNum
128 \def\ltx@GobbleNum#1{%
129   \romannumeral
130   \csname ltx@zero%
131   \expandafter\LTXcmds@GobbleNum
132   \romannumeral\LTXcmds@num{#1}000{m\endcsname}%
133 }

\LTXcmds@GobbleNum
134 \def\LTXcmds@GobbleNum#1{%
135   \csname LTXcmds@G#1\LTXcmds@GobbleNum
136 }

\LTXcmds@Gm
137 \long\def\LTXcmds@Gm#1{%
138   \endcsname
139 }

```

## 2.4 Argument grabbers

```

\ltx@firstofone
140 \long\def\ltx@firstofone#1{#1}

\ltx@firstoftwo
141 \long\def\ltx@firstoftwo#1#2{#1}

\ltx@secondoftwo
142 \long\def\ltx@secondoftwo#1#2{#2}

\ltx@firstofthree
143 \long\def\ltx@firstofthree#1#2#3{#1}

\ltx@secondofthree
144 \long\def\ltx@secondofthree#1#2#3{#2}

\ltx@thirdofthree
145 \long\def\ltx@thirdofthree#1#2#3{#3}%

\ltx@firstoffour
146 \long\def\ltx@firstoffour#1#2#3#4{#1}

\ltx@secondoffour
147 \long\def\ltx@secondoffour#1#2#3#4{#2}

\ltx@thirdoffour
148 \long\def\ltx@thirdoffour#1#2#3#4{#3}%

\ltx@fourthoffour
149 \long\def\ltx@fourthoffour#1#2#3#4{#4}%

```

## 2.5 List helpers

```

\ltx@car
150 \long\def\ltx@car#1#2\@nil{#1}

\ltx@cdr
151 \long\def\ltx@cdr#1#2\@nil{#2}

\ltx@carzero
152 \long\def\ltx@carzero#1\@nil{}%

\ltx@cdrzero
153 \long\def\ltx@cdrzero#1\@nil{#1}%

\ltx@cartwo
154 \long\def\ltx@cartwo#1#2#3\@nil{#1#2}

\ltx@cdrtwo
155 \long\def\ltx@cdrtwo#1#2#3\@nil{#3}

\ltx@carthree
156 \long\def\ltx@carthree#1#2#3#4\@nil{#1#2#3}

\ltx@cdrthree
157 \long\def\ltx@cdrthree#1#2#3#4\@nil{#4}

\ltx@carfour
158 \long\def\ltx@carfour#1#2#3#4#5\@nil{#1#2#3#4}

\ltx@cdrfour
159 \long\def\ltx@cdrfour#1#2#3#4#5\@nil{#5}

\ltx@CarNum
160 \def\ltx@CarNum#1{%
161   \romannumeral
162   \csname LTXcmds@CarNumFinish%
163   \expandafter\LTXcmds@CarNum
164   \romannumeral\LTXcmds@num{#1}000{x\endcsname}%
165 }

\LTXcmds@CarNum
166 \def\LTXcmds@CarNum#1{%
167   \csname LTXcmds@C#1\LTXcmds@CarNum
168 }

\LTXcmds@Cm
169 \long\def\LTXcmds@Cm#1#2{%
170   \endcsname{#1#2}%
171 }

\LTXcmds@Cx
172 \def\LTXcmds@Cx#1{%
173   \endcsname{}%
174 }

\LTXcmds@CarNumFinish
175 \long\def\LTXcmds@CarNumFinish#1#2\@nil{%
176   \ltx@zero
177   #1%
178 }

```

`\ltx@CdrNum`

```
179 \def\ltx@CdrNum#1{%
180   \romannumeral0%
181   \expandafter\expandafter\expandafter\LTXcmds@CdrNum
182   \ltx@GobbleNum{#1}%
183 }
```

`\LTXcmds@CdrNum`

```
184 \long\def\LTXcmds@CdrNum#1\@nil{ #1}%
```

## 2.6 Tail recursion

`\ltx@ReturnAfterFi`

```
185 \long\def\ltx@ReturnAfterFi#1\fi{\fi#1}
```

`\ltx@ReturnAfterElseFi`

```
186 \long\def\ltx@ReturnAfterElseFi#1\else#2\fi{\fi#1}
```

## 2.7 Empty macro

`\ltx@empty`

```
187 \def\ltx@empty{}
```

## 2.8 Characters

`\ltx@space`

```
188 \def\ltx@space{ }
```

`\ltx@percentchar`

```
189 \begingroup
190   \lccode'0='\% \relax
191 \lowercase{\endgroup
192   \def\ltx@percentchar{0}%
193 }
```

`\ltx@backslashchar`

```
194 \begingroup
195   \lccode'0='\\ \relax
196 \lowercase{\endgroup
197   \def\ltx@backslashchar{0}%
198 }
```

`\ltx@hashchar`

```
199 \begingroup
200   \lccode'0='\# \relax
201 \lowercase{\endgroup
202   \def\ltx@hashchar{0}%
203 }
```

`\ltx@leftbracechar`

```
204 \begingroup
205   \lccode'0='{ \relax
206 \lowercase{\endgroup
207   \def\ltx@leftbracechar{0}%
208 }
```

`\ltx@rightbracechar`

```
209 \begingroup
210 \lccode'0='}\relax
211 \lowercase{\endgroup
212 \def\ltx@rightbracechar{0}%
213 }
```

## 2.9 Boolean switch

`\ltx@newif`

```
214 \def\ltx@newif#1{%
215   \begingroup
216   \escapechar=-1 %
217   \expandafter\endgroup
218   \expandafter\LTXcmds@newif\string#1\@nil
219 }
```

`\LTXcmds@newif`

```
220 \begingroup
221 \escapechar=-1 %
222 \expandafter\endgroup
223 \expandafter\def\expandafter\LTXcmds@newif\string\if#1\@nil{%
224   \expandafter\edef\csname#1true\endcsname{%
225     \let
226     \expandafter\noexpand\csname if#1\endcsname
227     \noexpand\iftrue
228   }%
229   \expandafter\edef\csname#1false\endcsname{%
230     \let
231     \expandafter\noexpand\csname if#1\endcsname
232     \noexpand\iffalse
233   }%
234   \csname#1false\endcsname
235 }
```

`\ltx@newglobalif`

```
236 \def\ltx@newglobalif#1{%
237   \begingroup
238   \escapechar=-1 %
239   \expandafter\endgroup
240   \expandafter\LTXcmds@newglobalif\string#1\@nil
241 }
```

`\LTXcmds@newglobalif`

```
242 \begingroup
243 \escapechar=-1 %
244 \expandafter\endgroup
245 \expandafter
246 \def\expandafter\LTXcmds@newglobalif\string\if#1\@nil{%
247   \expandafter\edef\csname#1true\endcsname{%
248     \global\let
249     \expandafter\noexpand\csname if#1\endcsname
250     \noexpand\iftrue
251   }%
252   \expandafter\edef\csname#1false\endcsname{%
253     \global\let
254     \expandafter\noexpand\csname if#1\endcsname
255     \noexpand\iffalse
256   }%
257   \csname#1false\endcsname
258 }
```

## 2.10 Command definitions

```
\ltx@LocalExpandAfter

259 \def\ltx@LocalExpandAfter{%
260   \begingroup
261     \expandafter\expandafter\expandafter
262   \endgroup
263   \expandafter
264 }

265 \ltx@LocalExpandAfter
266 \ifx\csname ifcsname\endcsname\relax

\ltx@ifundefined

267 \def\ltx@ifundefined#1{%
268   \expandafter\ifx\csname #1\endcsname\relax
269     \expandafter\ltx@firstoftwo
270   \else
271     \expandafter\ltx@secondoftwo
272   \fi
273 }%

\ltx@ifUndefined

274 \def\ltx@ifUndefined#1{%
275   \begingroup\expandafter\expandafter\expandafter\endgroup
276   \expandafter\ifx\csname #1\endcsname\relax
277     \expandafter\ltx@firstoftwo
278   \else
279     \expandafter\ltx@secondoftwo
280   \fi
281 }%

282 \expandafter\ltx@gobble
283 \else
284 \expandafter\ltx@firstofone
285 \fi
286 {%

\ltx@ifundefined

287 \def\ltx@ifundefined#1{%
288   \ifcsname #1\endcsname
289     \expandafter\ifx\csname #1\endcsname\relax
290       \expandafter\expandafter\expandafter\ltx@firstoftwo
291     \else
292       \expandafter\expandafter\expandafter\ltx@secondoftwo
293     \fi
294   \else
295     \expandafter\ltx@firstoftwo
296   \fi
297 }%

\ltx@ifUndefined

298 \let\ltx@ifUndefined\ltx@ifundefined
299 }
```

## 2.11 Stripping

```
\ltx@RemovePrefix

300 \def\ltx@RemovePrefix#1>{}
```

`\ltx@StripPrefix`

```
301 \def\ltx@StripPrefix{%
302   \expandafter\ltx@RemovePrefix
303 }
```

`\ltx@onelevel@sanitize`

```
304 \def\ltx@onelevel@sanitize#1{%
305   \edef#1{%
306     \expandafter
307     \ltx@RemovePrefix\meaning#1%
308   }%
309 }
```

## 2.12 File management

### 2.12.1 File extensions

`\ltx@clsextension`

```
310 \def\ltx@clsextension{cls}
```

`\ltx@pkgextension`

```
311 \def\ltx@pkgextension{sty}
```

### 2.12.2 Load check

`\ltx@iffileloaded`

```
312 \def\ltx@iffileloaded#1{%
313   \ltx@ifundefined{ver@#1}\ltx@secondoftwo\ltx@firstoftwo
314 }
```

`\ltx@ifclassloaded`

```
315 \def\ltx@ifclassloaded#1{%
316   \ltx@iffileloaded{#1.\ltx@clsextension}%
317 }
```

`\ltx@ifpackageloaded`

```
318 \def\ltx@ifpackageloaded#1{%
319   \ltx@iffileloaded{#1.\ltx@pkgextension}%
320 }
```

### 2.12.3 Version date check

`\ltx@iffilelater`

```
321 \def\ltx@iffilelater#1#2{%
322   \ltx@iffileloaded{#1}{%
323     \expandafter\LTxcmds@iflater\expandafter{%
324       \number
325       \expandafter\expandafter\expandafter\LTxcmds@ParseVersion
326       \expandafter\expandafter\expandafter{%
327         \csname ver@#1\endcsname
328       }%
329     \expandafter}\expandafter{%
330       \number
331       \expandafter\LTxcmds@ParseVersion\expandafter{#2}%
332     }%
333   }\ltx@secondoftwo
334 }
```

\LTXcmds@IfLater

```

335 \def\LTXcmds@IfLater#1#2{%
336   \ifcase 0%
337     \ifnum#1<19940101 %
338     \else
339       \ifnum#2<19940101 %
340       \else
341         \ifnum#2>#1 %
342         \else
343           1%
344         \fi
345       \fi
346     \fi
347     \ltx@space
348     \expandafter\ltx@secondoftwo
349   \else
350     \expandafter\ltx@firstoftwo
351   \fi
352 }

```

\ltx@ifclasslater

```

353 \def\ltx@ifclasslater#1{%
354   \ltx@iffilelater{#1.\ltx@clsextension}%
355 }

```

\ltx@ifpackagelater

```

356 \def\ltx@ifpackagelater#1{%
357   \ltx@iffilelater{#1.\ltx@pkgextension}%
358 }

359 \ltx@ifundefined{pdfmatch}{%

```

\LTXcmds@ParseVersion

```

360 \def\LTXcmds@ParseVersion#1{%
361   \LTXcmds@@ParseVersion#10000/00/00\@nil
362 }%

```

\LTXcmds@@ParseVersion

```

363 \def\LTXcmds@@ParseVersion#1#2#3#4/#5#6/#7#8#9\@nil{%
364   #1#2#3#4#5#6#7#8%
365 }%

366 }{%

```

\LTXcmds@ParseVersion

```

367 \def\LTXcmds@ParseVersion#1{%
368   \ifnum\pdfmatch{%
369     ~%
370     (199[4-9] | [2-9] [0-9] [0-9] [0-9])/%
371     (0[1-9] | 1[0-2])/%
372     (0[1-9] | [1-2] [0-9] | 3[0-1])%
373   }{#1}=1 %
374   \ltx@StripPrefix\pdfmatch1 %
375   \ltx@StripPrefix\pdfmatch2 %
376   \ltx@StripPrefix\pdfmatch3 %
377   \else
378     0%
379   \fi
380 }%

381 }

```

## 2.13 Macro additions

`\ltx@GlobalAppendToMacro`

```
382 \long\def\ltx@GlobalAppendToMacro#1#2{%
383   \ifx\ltx@undefined#1%
384     \let#1\ltx@empty
385   \else
386     \ifx\relax#1%
387       \let#1\ltx@empty
388     \fi
389   \fi
390   \begingroup
391     \toks0\expandafter{#1#2}%
392     \xdef#1{\the\toks0}%
393   \endgroup
394 }
```

`\ltx@LocalAppendToMacro`

```
395 \long\def\ltx@LocalAppendToMacro#1#2{%
396   \global\let\LTxcmds@gtemp#1%
397   \ifx\ltx@undefined\LTxcmds@gtemp
398     \global\let\LTxcmds@gtemp\ltx@empty
399   \else
400     \ifx\relax\LTxcmds@gtemp
401       \global\let\LTxcmds@gtemp\ltx@empty
402     \fi
403   \fi
404   \begingroup
405     \toks0\expandafter{\LTxcmds@gtemp#2}%
406     \xdef\LTxcmds@gtemp{\the\toks0}%
407   \endgroup
408   \let#1\LTxcmds@gtemp
409 }
```

## 2.14 Macro `\ltx@ifnextchar`

`\ltx@ifnextchar`

```
410 \long\def\ltx@ifnextchar#1#2#3{%
411   \begingroup
412   \let\LTxcmds@CharToken= #1\relax
413   \toks\ltx@zero{#2}%
414   \toks\ltx@two{#3}%
415   \futurelet\LTxcmds@LetToken\LTxcmds@ifnextchar
416 }
```

`\LTxcmds@ifnextchar`

```
417 \def\LTxcmds@ifnextchar{%
418   \ifx\LTxcmds@LetToken\LTxcmds@CharToken
419     \expandafter\endgroup\the\toks\expandafter\ltx@zero
420   \else
421     \ifx\LTxcmds@LetToken\LTxcmds@SpaceToken
422       \expandafter\expandafter\expandafter\LTxcmds@@ifnextchar
423     \else
424       \expandafter\endgroup\the\toks
425       \expandafter\expandafter\expandafter\ltx@two
426     \fi
427   \fi
428 }
```

`\LTxcmds@@ifnextchar`

```
429 \begingroup
```

```

430 \def\x#1{\endgroup
431 \def\LTXcmds@ifnextchar#1{%
432 \futurelet\LTXcmds@LetToken\LTXcmds@ifnextchar
433 }%
434 }%
435 \x{ }

```

`\LTXcmds@SpaceToken`

```

436 \begingroup
437 \def\x#1{\endgroup
438 \let\LTXcmds@SpaceToken= #1%
439 }%
440 \x{ }

```

## 2.15 `\ltx@leavevmode`, `\ltx@mbox`

`\ltx@leavevmode`

```

441 \ltx@ifundefined{quitvmode}{%
442 \ltx@ifundefined{leavevmode}{%
443 \ltx@ifundefined{voidb@x}{%
444 \ltx@ifundefined{newbox}{%
445 \def\ltx@leavevmode{%
446 \begingroup
447 \setbox\ltx@zero=\hbox{}}%
448 \begingroup
449 \setbox\ltx@zero=\hbox{\box\ltx@zero}%
450 \endgroup
451 \unhbox\ltx@zero
452 \endgroup
453 }%
454 }{%
455 \csname newbox\endcsname\LTXcmds@VoidBox
456 \ifvoid\LTXcmds@VoidBox
457 \else
458 \setbox\LTXcmds@VoidBox=\hbox{}}%
459 \begingroup
460 \setbox\LTXcmds@VoidBox=\hbox{\box\LTXcmds@VoidBox}%
461 \endgroup
462 \fi
463 \def\ltx@leavevmode{\unhbox\LTXcmds@VoidBox}%
464 }%
465 }{%
466 \def\ltx@leavevmode{\unhbox\voidb@x}%
467 }%
468 }{%
469 \let\ltx@leavevmode\leavevmode
470 }%
471 }{%
472 \let\ltx@leavevmode\quitvmode
473 }

```

`\ltx@mbox`

```

474 \def\ltx@mbox{%
475 \ltx@leavevmode
476 \hbox
477 }

```

## 2.16 Help macros

`\LTXcmds@num`

```

478 \ltx@ifundefined{numexpr}{%

```

```

479 \def\LTXcmds@num#1{%
480   \expandafter\ltx@firstofone\expandafter{%
481     \number#1%
482   }%
483 }%
484 }{%
485 \def\LTXcmds@num#1{%
486   \expandafter\ltx@firstofone\expandafter{%
487     \the\numexpr#1%
488   }%
489 }%
490 }

```

## 2.17 Expandable test for emptiness

```

491 \ltx@ifundefined{detokenize}{%

```

### 2.17.1 Vanilla T<sub>E</sub>X

`\ltx@ifempty` The macro is based on `\@ifempty` of Robert R. Schneck [1] and `\@ifnull` of Ulrich Diez [2]. There are three cases to consider:

1. #1 is empty,
2. #1 is not empty and the first token is not a begingroup character,
3. #1 starts with a begingroup character (catcode 1).

```

492 \def\LTXcmds@temp#1{%
493   \long\def\ltx@ifempty##1{%
494     \romannumeral0%
495     \iffalse{\fi
496       \expandafter\ltx@gobble\expandafter{%
497         \expandafter\string##1}%
498       \expandafter\ltx@gobble\string
499     }%
500     \expandafter\ltx@firstofthree\expandafter
501     {\iffalse}\fi
502     \expandafter#1\ltx@secondoftwo
503   }%
504   \expandafter#1\ltx@firstoftwo
505 }%

```

`\ltx@ifblank`

```

506 \long\def\ltx@ifblank##1{%
507   \romannumeral0%
508   \iffalse{\fi
509     \expandafter\expandafter\expandafter\ltx@gobble
510     \expandafter\expandafter\expandafter{%
511       \expandafter\expandafter\expandafter{%
512         \expandafter\string\ltx@gobble##1.%
513       }%
514       \expandafter\ltx@gobble\string
515     }%
516     \expandafter\ltx@firstofthree\expandafter
517     {\iffalse}\fi
518     \expandafter#1\ltx@secondoftwo
519   }%
520   \expandafter#1\ltx@firstoftwo
521 }%
522 }%
523 \LTXcmds@temp{ }%
524 }{%

```

### 2.17.2 With \detokenize

Ahmed Musa provided `\ifstrempy` using `\detokenize` and `\pdfstrcmp` [3]. Ulrich Diez, GL, Heiko Oberdiek improved it further by removing `\pdfstrcmp` and taking three arguments [4, 5, 6, 7, 8].

`\ltx@ifempty`

```
525 \long\def\ltx@ifempty#1{%
526   \romannumeral%
527   \csname
528     LTXcmds@ifempty%
529     \ifcat$\detokenize{#1}$%
530     @%
531     \fi
532   \endcsname
533 }%
```

`\LTXcmds@ifempty@`

```
534 \long\def\LTXcmds@ifempty@#1#2{0 #1}%
```

`\LTXcmds@ifempty`

```
535 \long\def\LTXcmds@ifempty#1#2{0 #2}%
```

### 2.17.3 \ltx@ifblank

`\ltx@ifblank`

```
536 \long\def\ltx@ifblank#1{%
537   \romannumeral%
538   \csname
539     LTXcmds@ifempty%
540     \ifcat$\detokenize\expandafter{\ltx@gobble#1.}$%
541     @%
542     \fi
543   \endcsname
544 }%
545 }
```

## 2.18 \ltx@zapspace

`\ltx@zapspace`

```
546 \long\def\ltx@zapspace#1{%
547   \romannumeral
548   \LTXcmds@zapspace\ltx@zero#1 \@nil
549 }
```

`\LTXcmds@zapspace`

```
550 \long\def\LTXcmds@zapspace#1 #2\@nil{%
551   \ltx@ifempty{#2}{%
552     #1%
553   }{%
554     \LTXcmds@zapspace#1#2\@nil
555   }%
556 }
```

### 2.19 \ltx@ifBoxEmpty

In case of  $\varepsilon$ -TeX the test for an empty box is done via `\lastnodetype` as suggested by David Kastrup [9].

```
557 \ltx@ifUndefined{lastnodetype}{%
```

```

558 \catcode'\$=9 %
559 \catcode'\&=14 %
560 }{ %
561 \catcode'\$=14 %
562 \catcode'\&=9 %
563 }

```

`\ltx@ifboxempty`

```

564 \def\ltx@ifboxempty#1{%
565 \ifvoid#1\relax
566 \expandafter\ltx@secondoftwo
567 \else

```

Implementation using  $\varepsilon$ -TeX's `\lastnodetype`.

```

568 & \begin{group}
569 & \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%
570 & \ifhmode\unhcopy\else\unvcopy\fi#1\relax
571 & \expandafter
572 & }%
573 & \expandafter\endgroup
574 & \ifnum\lastnodetype<\ltx@zero
575 & \expandafter\expandafter\expandafter\ltx@firstoftwo
576 & \else
577 & \expandafter\expandafter\expandafter\ltx@secondoftwo
578 & \fi

```

Implementation without  $\varepsilon$ -TeX using a signature at the beginning of the test box.

```

579 $ \begin{group}
580 $ \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%
581 $ \penalty\ltx@one
582 $ \ifhmode\unhcopy\else\unvcopy\fi#1\relax
583 $ \expandafter
584 $ }%
585 $ \ifnum\lastpenalty=\ltx@one

```

Box 0 has been changed and is restored by closing the group.

```

586 $ \endgroup
587 $ \begin{group}
588 $ \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%
589 $ \penalty\ltx@two
590 $ \ifhmode\unhcopy\else\unvcopy\fi#1\relax
591 $ \expandafter
592 $ }%
593 $ \ifnum\lastpenalty=\ltx@two
594 $ \def\next{\endgroup\expandafter\ltx@firstoftwo}%
595 $ \else
596 $ \def\next{\endgroup\expandafter\ltx@secondoftwo}%
597 $ \fi
598 $ \else
599 $ \def\next{\endgroup\expandafter\ltx@secondoftwo}%
600 $ \fi
601 $ \next
602 \fi
603 }

```

`\ltx@ifboxvoidoreempty`

```

604 \def\ltx@ifboxvoidoreempty#1{%
605 \ifvoid#1\relax
606 \expandafter\ltx@thirdoffour
607 \fi
608 \ltx@ifboxempty{#1}%
609 }

```

```

610 \LTXcmds@AtEnd%
611 \</package>

```

## 3 Test

### 3.1 Catcode checks for loading

```

612 \<*test1>

613 \catcode'\{=1 %
614 \catcode'\}=2 %
615 \catcode'\#=6 %
616 \catcode'\@=11 %
617 \expandafter\ifx\csname count@\endcsname\relax
618   \countdef\count@=255 %
619 \fi
620 \expandafter\ifx\csname @gobble\endcsname\relax
621   \long\def\@gobble#1{}%
622 \fi
623 \expandafter\ifx\csname @firstofone\endcsname\relax
624   \long\def\@firstofone#1{#1}%
625 \fi
626 \expandafter\ifx\csname loop\endcsname\relax
627   \expandafter\@firstofone
628 \else
629   \expandafter\@gobble
630 \fi
631 {%
632   \def\loop#1\repeat{%
633     \def\body{#1}%
634     \iterate
635   }%
636   \def\iterate{%
637     \body
638     \let\next\iterate
639   \else
640     \let\next\relax
641   \fi
642   \next
643 }%
644 \let\repeat=\fi
645 }%
646 \def\RestoreCatcodes{}
647 \count@=0 %
648 \loop
649   \edef\RestoreCatcodes{%
650     \RestoreCatcodes
651     \catcode\the\count@=\the\catcode\count@\relax
652   }%
653 \ifnum\count@<255 %
654   \advance\count@ 1 %
655 \repeat
656
657 \def\RangeCatcodeInvalid#1#2{%
658   \count@=#1\relax
659   \loop
660     \catcode\count@=15 %
661   \ifnum\count@<#2\relax
662     \advance\count@ 1 %
663   \repeat
664 }
665 \def\RangeCatcodeCheck#1#2#3{%

```

```

666 \count@=#1\relax
667 \loop
668 \ifnum#3=\catcode\count@
669 \else
670 \errmessage{%
671 Character \the\count@\space
672 with wrong catcode \the\catcode\count@\space
673 instead of \number#3%
674 }%
675 \fi
676 \ifnum\count@<#2\relax
677 \advance\count@ 1 %
678 \repeat
679 }
680 \def\space{ }
681 \expandafter\ifx\csname LoadCommand\endcsname\relax
682 \def\LoadCommand{\input ltxcmds.sty\relax}%
683 \fi
684 \def\Test{%
685 \RangeCatcodeInvalid{0}{47}%
686 \RangeCatcodeInvalid{58}{64}%
687 \RangeCatcodeInvalid{91}{96}%
688 \RangeCatcodeInvalid{123}{255}%
689 \catcode'\@=12 %
690 \catcode'\=0 %
691 \catcode'\%=14 %
692 \LoadCommand
693 \RangeCatcodeCheck{0}{36}{15}%
694 \RangeCatcodeCheck{37}{37}{14}%
695 \RangeCatcodeCheck{38}{47}{15}%
696 \RangeCatcodeCheck{48}{57}{12}%
697 \RangeCatcodeCheck{58}{63}{15}%
698 \RangeCatcodeCheck{64}{64}{12}%
699 \RangeCatcodeCheck{65}{90}{11}%
700 \RangeCatcodeCheck{91}{91}{15}%
701 \RangeCatcodeCheck{92}{92}{0}%
702 \RangeCatcodeCheck{93}{96}{15}%
703 \RangeCatcodeCheck{97}{122}{11}%
704 \RangeCatcodeCheck{123}{255}{15}%
705 \RestoreCatcodes
706 }
707 \Test
708 \csname @@end\endcsname
709 \end
710 </test1>

```

### 3.2 Test \ltx@GobbleNum

```

711 <*test-gobble>
712 \catcode'\{=1 %
713 \catcode'\}=2 %
714 \catcode'\#=6 %
715 \expandafter\ifx\csname RequirePackage\endcsname\relax
716 \input ltxcmds.sty\relax
717 \else
718 \RequirePackage{ltxcmds}[2011/03/16]%
719 \fi
720 \catcode'\@=11 %
721 \def\msg#{\immediate\write16}%
722 \msg{[Test \string\ltx@GobbleNum]}%
723 \long\def\Test#1=#2\\{%
724 \edef\StrA{\ltx@GobbleNum#1}%
725 \expandafter\expandafter\expandafter\def

```

```

726 \expandafter\expandafter\expandafter\StrAA
727 \expandafter\expandafter\expandafter{\ltx@GobbleNum#1}%
728 \edef\StrB{#2}%
729 \ifx\StrA\StrB
730   \ifx\StrAA\StrB
731     \msg{* ok.}%
732   \else
733     \msg{StrAA: \StrAA}%
734     \msg{StrB: \StrB}%
735     \errhelp{Test: #1=#2}%
736     \errmessage{Test (two expansions) failed}%
737   \fi
738 \else
739   \msg{StrA: \StrA}%
740   \msg{StrB: \StrB}%
741   \errhelp{Test: #1=#2}%
742   \errmessage{Test (edef) failed!}%
743 \fi
744 }
745 \Test0abc=abc\\
746 \Test1abc=bc\\
747 \Test2abc=c\\
748 \Test3abcd=d\\
749 \Test4abcde=e\\
750 \Test5abcdef=f\\
751 \Test6abcdefg=g\\
752 \Test7abcdefgh=h\\
753 \Test8abcdefghi=i\\
754 \Test9abcdefghij=j\\
755 \Test{10}0123456789X=X\\
756 \Test{12}abcdefghijkln=m\\
757 \Test{700}%
758 012345678901234567890123456789012345678901234567890123456789%
759 012345678901234567890123456789012345678901234567890123456789%
760 012345678901234567890123456789012345678901234567890123456789%
761 012345678901234567890123456789012345678901234567890123456789%
762 012345678901234567890123456789012345678901234567890123456789%
763 012345678901234567890123456789012345678901234567890123456789%
764 012345678901234567890123456789012345678901234567890123456789%
765 012345678901234567890123456789012345678901234567890123456789%
766 012345678901234567890123456789012345678901234567890123456789%
767 012345678901234567890123456789012345678901234567890123456789%
768 X=X\\
769 \Test{-1}abc=abc\\
770 \Test2\par\par\relax=\relax\\
771
772 \begingroup
773   \count1=2 %
774   \Test{\count1}abc=c\\
775 \endgroup
776
777 \ltx@ifUndefined{numexpr}{%
778 }{%
779   \Test{1+1}abc=c\\
780 }
781
782 \msg{[Test \string\ltx@CdrNum]}%
783 \long\def\Test#1=#2\{%
784   \edef\StrA{\ltx@CdrNum#1\@nil}%
785   \expandafter\expandafter\expandafter\def
786   \expandafter\expandafter\expandafter\StrAA
787   \expandafter\expandafter\expandafter{\ltx@CdrNum#1\@nil}%

```

```

788 \edef\StrB{#2}%
789 \ifx\StrA\StrB
790 \ifx\StrAA\StrB
791 \msg{* ok.}%
792 \else
793 \msg{StrAA: \meaning\StrAA}%
794 \msg{StrB: \meaning\StrB}%
795 \errhelp{Test: #1=#2}%
796 \errmessage{Test (two expansions) failed}%
797 \fi
798 \else
799 \msg{StrA: \StrA}%
800 \msg{StrB: \StrB}%
801 \errhelp{Test: #1=#2}%
802 \errmessage{Test (edef) failed!}%
803 \fi
804 }
805 \Test0abc=abc\\
806 \Test1abc=bc\\
807 \Test2abc=c\\
808 \Test3abcd=d\\
809 \Test4abcde=e\\
810 \Test5abcdef=f\\
811 \Test6abcdefg=g\\
812 \Test7abcdefgh=h\\
813 \Test8abcdefghi=i\\
814 \Test9abcdefghij=j\\
815 \Test{10}0123456789X=X\\
816 \Test{12}abcdefghijklm=m\\
817 \Test{700}%
818 012345678901234567890123456789012345678901234567890123456789%
819 012345678901234567890123456789012345678901234567890123456789%
820 012345678901234567890123456789012345678901234567890123456789%
821 012345678901234567890123456789012345678901234567890123456789%
822 012345678901234567890123456789012345678901234567890123456789%
823 012345678901234567890123456789012345678901234567890123456789%
824 012345678901234567890123456789012345678901234567890123456789%
825 012345678901234567890123456789012345678901234567890123456789%
826 012345678901234567890123456789012345678901234567890123456789%
827 012345678901234567890123456789012345678901234567890123456789%
828 X=X\\
829 \Test{-1}abc=abc\\
830 \Test2\par\par\relax=\relax\\
831
832 \msg{[Test \string\ltx@CarNum]}%
833 \long\def\Test#1=#2\\{%
834 \edef\StrA{\ltx@CarNum#1\@nil}%
835 \expandafter\expandafter\expandafter\def
836 \expandafter\expandafter\expandafter\StrAA
837 \expandafter\expandafter\expandafter{\ltx@CarNum#1\@nil}%
838 \edef\StrB{#2}%
839 \ifx\StrA\StrB
840 \ifx\StrAA\StrB
841 \msg{* ok.}%
842 \else
843 \msg{StrAA: \meaning\StrAA}%
844 \msg{StrB: \meaning\StrB}%
845 \errhelp{Test: #1=#2}%
846 \errmessage{Test (two expansions) failed}%
847 \fi
848 \else
849 \msg{StrA: \StrA}%

```

```

850 \msg{StrB: \StrB}%
851 \errhelp{Test: #1=#2}%
852 \errmessage{Test (edef) failed!}%
853 \fi
854 }
855 \Test0abc=\\
856 \Test1abc=a\\
857 \Test2abc=ab\\
858 \Test3abc=abc\\
859 \Test3abcd=abcd\\
860 \Test4abcde=abcd\\
861 \Test{10}0123456789X=0123456789\\
862 \Test{12}abcdefghijklm=abcdefghijkl\\
863 \Test{700}%
864 0123456789012345678901234567890123456789012345678901234567890123456789%
865 0123456789012345678901234567890123456789012345678901234567890123456789%
866 0123456789012345678901234567890123456789012345678901234567890123456789%
867 0123456789012345678901234567890123456789012345678901234567890123456789%
868 0123456789012345678901234567890123456789012345678901234567890123456789%
869 0123456789012345678901234567890123456789012345678901234567890123456789%
870 0123456789012345678901234567890123456789012345678901234567890123456789%
871 0123456789012345678901234567890123456789012345678901234567890123456789%
872 0123456789012345678901234567890123456789012345678901234567890123456789%
873 0123456789012345678901234567890123456789012345678901234567890123456789%
874 X=%
875 0123456789012345678901234567890123456789012345678901234567890123456789%
876 0123456789012345678901234567890123456789012345678901234567890123456789%
877 0123456789012345678901234567890123456789012345678901234567890123456789%
878 0123456789012345678901234567890123456789012345678901234567890123456789%
879 0123456789012345678901234567890123456789012345678901234567890123456789%
880 0123456789012345678901234567890123456789012345678901234567890123456789%
881 0123456789012345678901234567890123456789012345678901234567890123456789%
882 0123456789012345678901234567890123456789012345678901234567890123456789%
883 0123456789012345678901234567890123456789012345678901234567890123456789%
884 0123456789012345678901234567890123456789012345678901234567890123456789%
885 \\
886 \Test{-1}abc=\\
887 \Test2\par\par\relax=\par\par\\
888 \csname @@end\endcsname\end
889 \</test-gobble>

```

### 3.3 Test \ltx@ifempty

```

890 \<test-ifempty>
891 \catcode'\{=1 %
892 \catcode'\}=2 %
893 \catcode'\#=6 %
894 \catcode'\@=11 %
895 \errorcontextlines=1000 %
896 \begingroup\expandafter\expandafter\expandafter\endgroup
897 \expandafter\ifx\csname RequirePackage\endcsname\relax
898 \input ltxcmds.sty\relax
899 \else
900 \RequirePackage{ltxcmds}[2011/03/16]%
901 \fi
902 \def\msg#\<immediate\write16>
903 \def\TestY{Y}
904 \def\TestN{N}
905 \msg{* \string\ltx@ifempty}
906 \long\def\test#1{%
907 \begingroup
908 % Calculate expected test result via macro definition
909 \def\Stuff{#1}%

```

```

910 \ifx\Stuff\ltx@empty
911 \def\StuffEmpty{\Y}%
912 \else
913 \def\StuffEmpty{\N}%
914 \fi
915 % Test \ltx@ifempty
916 \expandafter\expandafter\expandafter\def
917 \expandafter\expandafter\expandafter\TestEmpty
918 \expandafter\expandafter\expandafter{%
919 \ltx@ifempty{#1}{\Y}{\N}%
920 }%
921 \ifx\StuffEmpty\TestEmpty
922 \msg{* Test OK}%
923 \else
924 \ltx@ifundefined{detokenize}{}{%
925 \msg{Stuff: [\detokenize{\Stuff}]}%
926 }%
927 \errmessage{Test failed!}%
928 \fi
929 \endgroup
930 }
931 \test{}
932 \test{a}
933 \test{abc}
934 \test{\par}
935 \test{ }
936 \test{\if}
937 \test{{\if}}
938 \test{\else}
939 \test{{\else}}
940 \test{\fi}
941 \test{{}\fi}
942 \test{\or\ifcase}
943 \test{{}}
944 \test{{a}}
945 \test{{}abc}
946 \test{{\par}}
947 \test{{}\par}

948 \def\SpaceTwo#1{%
949 \def\SpaceTwo{#1#1}%
950 }\SpaceTwo{ }
951 \msg{* \string\ltx@ifblank}
952 \long\def\test#1{%
953 \begin{group}
954 % Calculate expected test result via macro definition
955 \def\Stuff{#1}%
956 \ifx\Stuff\ltx@empty
957 \def\StuffEmpty{\Y}%
958 \else
959 \ifx\Stuff\ltx@space
960 \def\StuffEmpty{\Y}%
961 \else
962 \ifx\Stuff\SpaceTwo
963 \def\StuffEmpty{\Y}%
964 \else
965 \def\StuffEmpty{\N}%
966 \fi
967 \fi
968 \fi
969 % Test \ltx@ifblank
970 \expandafter\expandafter\expandafter\def
971 \expandafter\expandafter\expandafter\TestEmpty

```

```

972 \expandafter\expandafter\expandafter{%
973 \ltx@ifblank{#1}{\Y}{\N}%
974 }%
975 \ifx\StuffEmpty\TestEmpty
976 \msg{* Test OK}%
977 \else
978 \ltx@ifundefined{detokenize}{\{%
979 \msg{Stuff: [\detokenize{\Stuff}]]}%
980 }%
981 \errmessage{Test failed!}%
982 \fi
983 \endgroup
984 }
985 \test{}
986 \test{a}
987 \test{\if}
988 \test{\else}
989 \test{\fi}
990 \test{ \fi}
991 \test{\par}
992 \test{ \par}
993 \test{ {} }
994 \test{ {} }
995 \def\x#1{%
996 \test{#1#1}%
997 \test{#1#1{}}%
998 \test{#1#1\par}%
999 \test{#1#1\else}%
1000 }\x{ }
1001 \csname @@end\endcsname\end
1002 \</test-ifempty>

```

### 3.4 Test \ltx@zap@space

```

1003 <*test-zap@space>
1004 \catcode'\{=1 %
1005 \catcode'\}=2 %
1006 \catcode'\#=6 %
1007 \catcode'\@=11 %
1008 \errorcontextlines=1000 %
1009 \begingroup\expandafter\expandafter\expandafter\endgroup
1010 \expandafter\ifx\csname RequirePackage\endcsname\relax
1011 \input ltxcmds.sty\relax
1012 \else
1013 \RequirePackage{ltxcmds}[2011/03/16]%
1014 \fi
1015 \def\msg#{\immediate\write16}
1016 \def\space{ }
1017 \def\empty{}
1018 \msg{* \string\ltx@zap@space}
1019 \long\def\test#1#2{%
1020 \begingroup
1021 \def\TestInput{#1}%
1022 \def\TestExpected{#2}%
1023 % Test \ltx@zap@space
1024 \expandafter\expandafter\expandafter\def
1025 \expandafter\expandafter\expandafter\TestResult
1026 \expandafter\expandafter\expandafter{%
1027 \ltx@zap@space{#1}%
1028 }%
1029 \ifx\TestResult\TestExpected
1030 \msg{* Test OK}%
1031 \else

```

```

1032      \ltx@onelevel@sanitize\TestInput
1033      \ltx@onelevel@sanitize\TestExpected
1034      \ltx@onelevel@sanitize\TestResult
1035      \msg{* Input: \space\space\space[\TestInput]]}%
1036      \msg{ \space Result: \space\space[\TestResult]]}%
1037      \msg{ \space Expected: [\TestExpected]]}%
1038      \errmessage{Test failed!}%
1039    \fi
1040  \endgroup
1041 }
1042 \long\def\etest#1#2{%
1043   \begingroup
1044     \edef\x{\endgroup
1045       \noexpand\test{#1}{#2}%
1046     }%
1047   \x
1048 }
1049 \catcode'\~ =13 %
1050 \let~\noexpand

1051 \test{}{}
1052 \test{}{}{}
1053 \test{ }{}{}
1054 \test{ { } }{}
1055 \test{ { } }{}{}
1056 \test{ { } }{}{}
1057 \test{ { } }{}{}
1058 \test{a {b} c}{a{b}c}
1059 \test{a bb ccc}{abbccc}
1060 \test{{a} {bb} {ccc}}{{a}{bb}{ccc}}
1061 \test{\par}{\par}
1062 \test{\if}{\if}
1063 \test{\space}{\space}
1064 \etest{\par\space\par}{\par\par}
1065 \etest{~\empty\space~\empty}{~\empty~\empty}
1066 \etest{~\fi\space~\else\space}{~\fi~\else}

1067 \csname @@end\endcsname\end
1068 </test-zapspac>

```

### 3.5 Test \ltx@ifBoxEmpty

```

1069 <*test-ifboxempty>
1070 \catcode'\{=1 %
1071 \catcode'\}=2 %
1072 \catcode'\#=6 %
1073 \catcode'\@=11 %
1074 \begingroup\expandafter\expandafter\expandafter\endgroup
1075 \expandafter\ifx\csname RequirePackage\endcsname\relax
1076   \input ltxcmds.sty\relax
1077 \else
1078   \RequirePackage{ltxcmds}[2011/03/16]%
1079 \fi
1080 \def\msg#{\immediate\write16}
1081 % make box 0 void
1082 \begingroup
1083   \setbox0=\box0 %
1084 \endgroup
1085 \ifvoid0 %
1086 \else
1087   \errmessage{Voiding box 0 failed}%
1088 \fi
1089 \setbox2=\box0 %
1090 \def\test#1#2{%
1091   \@test{#1}{#2}%

```

```

1092 \@@test{#1}{#2}%
1093 \chardef\x=#1%
1094 \@test\x{#2}%
1095 \@@test\x{#2}%
1096 }
1097 \def\@test#1#2{%
1098 \begingroup
1099 \setbox9=\hbox{%
1100 \def\TestExpected{#2}%
1101 \ltx@ifboxempty{#1}{%
1102 \def\TestResult{Y}%
1103 }{%
1104 \def\TestResult{N}%
1105 }%
1106 \ifx\TestExpected\TestResult
1107 \msg{* Test passed.}%
1108 \else
1109 \errmessage{Test failed!}%
1110 \fi
1111 }%
1112 \ifdim\wd9=0pt %
1113 \else
1114 \errmessage{Unwanted space?}%
1115 \fi
1116 \endgroup
1117 }
1118 \def\@@test#1#2{%
1119 \begingroup
1120 \setbox9=\hbox{%
1121 \def\TestExpected{#2}%
1122 \ifvoid#1\def\TestExpected{Y}\fi
1123 \ltx@ifboxvoidoreempty{#1}{%
1124 \def\TestResult{Y}%
1125 }{%
1126 \def\TestResult{N}%
1127 }%
1128 \ifx\TestExpected\TestResult
1129 \msg{* Test passed.}%
1130 \else
1131 \errmessage{Test failed!}%
1132 \fi
1133 }%
1134 \ifdim\wd9=0pt %
1135 \else
1136 \errmessage{Unwanted space?}%
1137 \fi
1138 \endgroup
1139 }
1140 \test0N
1141 \test2N
1142 \setbox0=\hbox{}
1143 \test0Y
1144 \setbox2=\hbox{}
1145 \test2Y
1146 \setbox0=\vbox{}
1147 \test0Y
1148 \setbox2=\vbox{}
1149 \test0Y
1150 \setbox0=\hbox{ }%
1151 \test0N
1152 \setbox2=\hbox{ }%
1153 \test2N

```

```

1154 \setbox0=\hbox{\penalty1}%
1155 \test0N
1156 \setbox2=\hbox{\penalty1}%
1157 \test2N
1158 \csname @@end\endcsname\end
1159 \</test-ifboxempty>

```

## 4 Installation

### 4.1 Download

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

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

[CTAN:macros/latex/contrib/oberdiek/ltxcmds.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 ltxcmds.dtx
```

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

<code>ltxcmds.sty</code>	→ <code>tex/generic/oberdiek/ltxcmds.sty</code>
<code>ltxcmds.pdf</code>	→ <code>doc/latex/oberdiek/ltxcmds.pdf</code>
<code>test/ltxcmds-test1.tex</code>	→ <code>doc/latex/oberdiek/test/ltxcmds-test1.tex</code>
<code>test/ltxcmds-test-gobble.tex</code>	→ <code>doc/latex/oberdiek/test/ltxcmds-test-gobble.tex</code>
<code>test/ltxcmds-test-ifempty.tex</code>	→ <code>doc/latex/oberdiek/test/ltxcmds-test-ifempty.tex</code>
<code>test/ltxcmds-test-zapspc.tex</code>	→ <code>doc/latex/oberdiek/test/ltxcmds-test-zapspc.tex</code>
<code>test/ltxcmds-test-ifboxempty.tex</code>	→ <code>doc/latex/oberdiek/test/ltxcmds-test-ifboxempty.tex</code>
<code>ltxcmds.dtx</code>	→ <code>source/latex/oberdiek/ltxcmds.dtx</code>

If you have a `docstrip.cfg` that configures and enables `docstrip`’s TDS installing feature, then some files can already be in the right place, see the documentation of `docstrip`.

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

## 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 `mktexlsr`.

## 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 ltxcmds.pdf unpack_files output .
```

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

**plain  $\text{T}_{\text{E}}\text{X}$ :** 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{ltxcmds.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 ltxcmds.dtx
makeindex -s gind.ist ltxcmds.idx
pdflatex ltxcmds.dtx
makeindex -s gind.ist ltxcmds.idx
pdflatex ltxcmds.dtx
```

## 5 References

- [1] Robert R. Schneck: *Re: \ifempty solution (was Macro puzzle: maximally general \ifempty)*; newsgroup `comp.text.tex`, `news:3eef1ada_6@corp.newsgroups.com`, 2003-06-17.  
<http://groups.google.com/group/comp.text.tex/msg/be03a159ec374895>
- [2] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:ibk3t8$ee7$1@news.albasani.net`, 2010-11-12.  
<http://groups.google.com/group/comp.text.tex/msg/803bd57221a04996>
- [3] Ahmed Musa: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:f5496afe-40ed-42bd-b629-a2419ecf7c0d@o14g2000prn.googlegroups.com`, 2010-12-03.  
<http://groups.google.com/group/comp.text.tex/msg/fbf7d61a0c3a807d>
- [4] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idbo94$uka$1@four.albasani.net`, 2010-12-03.  
<http://groups.google.com/group/comp.text.tex/msg/0c230ee479487962>

- [5] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idbpu4$cg1$1@news.albasani.net`, 2010-12-03.  
<http://groups.google.com/group/comp.text.tex/msg/bbef4263390d647b>
- [6] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idd4ga$r83$1@four.albasani.net`, 2010-12-04.  
<http://groups.google.com/group/comp.text.tex/msg/00dfd1ec103cd272>
- [7] GL: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:4cfa2e27$0$7389$426a74cc@news.free.fr`, 2010-12-04.  
<http://groups.google.com/group/comp.text.tex/msg/d3a75995c1cf267e>
- [8] Heiko Oberdiek: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:iddhq1$3kj$1@news.eternal-september.org`, 2010-12-04.  
<http://groups.google.com/group/comp.text.tex/msg/5f7a23e3ab70e347>
- [9] David Kastrup: *How to detect if \vbox is empty*; newsgroup `comp.text.tex`, 2011-02-04.  
<http://groups.google.com/group/comp.text.tex/msg/8d3cb89496a4d86d>

## 6 History

### [2009/08/05 v1.0]

- First version.

### [2009/12/12 v1.1]

- Short title shortened.
- `\ltx@ifUndefined` added.

### [2010/01/28 v1.2]

- `\ltx@RemovePrefix` and `\ltx@StripPrefix` added.
- `\ltx@ifclassloaded`, `\ltx@ifpackageloaded`, `\ltx@iffileloaded`, `\ltx@ifclasslater`, `\ltx@ifpackagelater`, `\ltx@iffilelater`, `\ltx@clsextension`, `\ltx@pkgextension` added.
- `\ltx@GlobalAppendToMacro`, `\ltx@LocalAppendToMacro` added.

### [2010/03/01 v1.3]

- `\ltx@newif` added.
- `\ltx@ifnextchar` added.
- Numbers `\ltx@zero`, `\ltx@one`, `\ltx@two`, `\ltx@cclv` added.

### [2010/03/09 v1.4]

- `\ltx@pkgextension` and `\ltx@clsextension` are hardcoded to avoid trouble with `\@onlypreamble`.

### [2010/04/08 v1.5]

- `\ltx@cartwo`, `\ltx@cdrtwo`, `\ltx@carthree`, `\ltx@cdrthree`, `\ltx@carfour`, `\ltx@cdrfour` added.
- `\ltx@ReturnAfterFi` and `\ltx@ReturnAfterElseFi` fixed.

[2010/04/16 v1.6]

- `\ltx@leavevmode`, `\ltx@mbox` added.

[2010/04/26 v1.7]

- `\ltx@GobbleNum`, `\ltx@CdrNum`, `\ltx@CarNum` added.
- `\ltx@carzero`, `\ltx@cdrzero` added.
- `\ltx@hashchar` added.

[2010/09/11 v1.8]

- `\ltx@leftbracechar`, `\ltx@rightbracechar` added.

[2010/10/25 v1.9]

- `\ltx@LocalAppendToMacro` and `\ltx@GlobalAppendToMacro` are now `\long`.

[2010/10/31 v1.10]

- `\ltx@newglobalif` added.

[2010/11/12 v1.11]

- `\ltx@ifempty` added.
- `\ltx@firstofthree`, `\ltx@secondofthree`, `\ltx@thirdofthree` added.

[2010/12/02 v1.12]

- `\ltx@onelevel@sanitize` added.
- `\LTXcmds@num` fixed for the case with `\numexpr` (bug found by GL).

[2010/12/04 v1.13]

- `\ltx@ifblank` added.
- Optimization for `\ltx@ifempty`.

[2010/12/07 v1.14]

- `\ltx@zapspace` added.

[2010/12/12 v1.15]

- `\ltx@minusone` added.

[2011/02/04 v1.16]

- `\ltx@ifBoxEmpty` and `\ltx@ifBoxVoidOrEmpty` added.
- `\ltx@firstoffour`, ..., `\ltx@fourthoffour` added.

[2011/02/05 v1.17]

- `\ltx@ifBoxEmpty`: an empty box may have non-zero dimensions.

[2011/03/16 v1.18]

- `\ltx@ifclasslater` fixed.

## 7 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; plain numbers refer to the code lines where the entry is used.

Symbols	
<code>\#</code> . . . . .	200, 615, 714, 893, 1006, 1072
<code>\\$</code> . . . . .	558, 561
<code>\%</code> . . . . .	190, 691
<code>\&amp;</code> . . . . .	559, 562
<code>\@</code> . . . . .	616, 689, 720, 894, 1007, 1073
<code>\@test</code> . . . . .	1092, 1095, 1118
<code>\@firstofone</code> . . . . .	624, 627
<code>\@gobble</code> . . . . .	621, 629
<code>\@nil</code> . . . . .	150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 175, 184, 218, 223, 240, 246, 361, 363, 548, 550, 554, 784, 787, 834, 837
<code>\@test</code> . . . . .	1091, 1094, 1097
<code>\@undefined</code> . . . . .	58
<code>\</code> . . . . .	195, 690, 723, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 768, 769, 770, 774, 779, 783, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 828, 829, 830, 833, 855, 856, 857, 858, 859, 860, 861, 862, 885, 886, 887
<code>\{</code> . . . . .	205, 613, 712, 891, 1004, 1070
<code>\}</code> . . . . .	210, 614, 713, 892, 1005, 1071
<code>\~</code> . . . . .	1049
A	
<code>\advance</code> . . . . .	654, 662, 677
<code>\aftergroup</code> . . . . .	29
B	
<code>\body</code> . . . . .	633, 637
<code>\box</code> . . . . .	449, 460, 1083, 1089
C	
<code>\catcode</code> . . . . .	2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 33, 34, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 69, 70, 72, 73, 74, 78, 79, 80, 81, 82, 83, 84, 87, 88, 90, 91, 92, 93, 97, 99, 558, 559, 561, 562, 613, 614, 615, 616, 651, 660, 668, 672, 689, 690, 691, 712, 713, 714, 720, 891, 892, 893, 894, 1004, 1005, 1006, 1007, 1049, 1070, 1071, 1072, 1073
<code>\chardef</code> . . . . .	116, 117, 118, 119, 120, 1093
<code>\count</code> . . . . .	773, 774
<code>\count@</code> . . . . .	618, 647, 651, 653, 654, 658, 660, 661, 662, 666, 668, 671, 672, 676, 677
<code>\countdef</code> . . . . .	618
<code>\csname</code> . . . . .	14, 21, 50, 66, 76, 130, 135, 162, 167, 224, 226, 229, 231, 234, 247, 249, 252, 254, 257, 266, 268, 276, 289, 327, 455, 527, 538, 617, 620, 623, 626, 681, 708, 715, 888, 897, 1001, 1010, 1067, 1075, 1158
D	
<code>\detokenize</code> . . . . .	529, 540, 925, 979
E	
<code>\empty</code> . . . . .	17, 18, 1017, 1065
<code>\end</code> . . . . .	709, 888, 1001, 1067, 1158
<code>\endcsname</code> . . . . .	14, 21, 50, 66, 76, 132, 138, 164, 170, 173, 224, 226, 229, 231, 234, 247, 249, 252, 254, 257, 266, 268, 276, 288, 289, 327, 455, 532, 543, 617, 620, 623, 626, 681, 708, 715, 888, 897, 1001, 1010, 1067, 1075, 1158
<code>\endinput</code> . . . . .	29, 115
<code>\endlinechar</code> . . . . .	4, 35, 71, 77, 89
<code>\errhelp</code> . . . . .	735, 741, 795, 801, 845, 851
<code>\errmessage</code> . . . . .	670, 736, 742, 796, 802, 846, 852, 927, 981, 1038, 1087, 1109, 1114, 1131, 1136
<code>\errorcontextlines</code> . . . . .	895, 1008
<code>\escapechar</code> . . . . .	216, 221, 238, 243
<code>\etest</code> . . . . .	1042, 1064, 1065, 1066
F	
<code>\futurelet</code> . . . . .	415, 432
H	
<code>\hbox</code> . . . . .	447, 449, 458, 460, 476, 569, 580, 588, 1099, 1120, 1142, 1144, 1150, 1152, 1154, 1156
I	
<code>\if</code> . . . . .	223, 246, 936, 937, 987, 1062
<code>\ifcase</code> . . . . .	336, 942
<code>\ifcat</code> . . . . .	529, 540
<code>\ifcsname</code> . . . . .	288
<code>\ifdim</code> . . . . .	1112, 1134
<code>\iffalse</code> . . . . .	232, 255, 495, 501, 508, 517
<code>\ifhbox</code> . . . . .	569, 580, 588
<code>\ifhmode</code> . . . . .	570, 582, 590
<code>\ifnum</code> . . . . .	337, 339, 341, 368, 574, 585, 593, 653, 661, 668, 676
<code>\iftrue</code> . . . . .	227, 250

\ifvoid	456, 565, 605, 1085, 1122	\ltx@ifpackagelater	356
\ifx	15, 18, 21, 50, 58, 61, 266, 268, 276, 289, 383, 386, 397, 400, 418, 421, 617, 620, 623, 626, 681, 715, 729, 730, 789, 790, 839, 840, 897, 910, 921, 956, 959, 962, 975, 1010, 1029, 1075, 1106, 1128	\ltx@ifpackageloaded	318
\immediate	23, 52, 721, 902, 1015, 1080	\ltx@ifundefined	5, 274, 298, 359, 441, 442, 443, 444, 478, 491, 557, 777, 924, 978
\input	682, 716, 898, 1011, 1076	\ltx@ifundefined	5, 267, 287, 298, 313
\iterate	634, 636, 638	\ltx@leavevmode	7, 441, 475
<b>L</b>			
\lastnodetype	574	\ltx@leftbracechar	204
\lastpenalty	585, 593	\ltx@LocalAppendToMacro	395
\lccode	190, 195, 200, 205, 210	\ltx@LocalExpandAfter	5, 259, 265
\leavevmode	469	\ltx@mbox	7, 474
\letLTxcmds@gtemp	401	\ltx@minusone	121
\LoadCommand	682, 692	\ltx@newglobalif	5, 236
\loop	632, 648, 659, 667	\ltx@newif	5, 214
\lowercase	191, 196, 201, 206, 211	\ltx@one	117, 122, 581, 585
\ltx@active	119	\ltx@onelevel@sanitize	5, 304, 1032, 1033, 1034
\ltx@backslashchar	194	\ltx@percentchar	189
\ltx@car	4, 150	\ltx@pkgextension	311, 319, 357
\ltx@carfour	4, 158	\ltx@RemovePrefix	5, 300, 302, 307
\ltx@CarNum	4, 160, 832, 834, 837	\ltx@ReturnAfterElseFi	186
\ltx@carthree	4, 156	\ltx@ReturnAfterFi	4, 185
\ltx@cartwo	4, 154	\ltx@rightbracechar	209
\ltx@carzero	4, 152	\ltx@secondoffour	147
\ltx@cclv	120	\ltx@secondofthree	144
\ltx@cdr	151	\ltx@secondoftwo	142, 271, 279, 292, 313, 333, 348, 502, 518, 566, 577, 596, 599
\ltx@cdrfour	159	\ltx@space	4, 188, 347, 959
\ltx@CdrNum	179, 782, 784, 787	\ltx@StripPrefix	301, 374, 375, 376
\ltx@cdrthree	157	\ltx@thirdoffour	148, 606
\ltx@cdrtwo	155	\ltx@thirdofthree	145
\ltx@cdrzero	153	\ltx@two	118, 414, 425, 589, 593
\ltx@clsextension	6, 310, 316, 354	\ltx@undefined	383, 397
\ltx@empty	4, 187, 384, 387, 398, 401, 910, 956	\ltx@zapspace	7, 546, 1018, 1023, 1027
\ltx@firstoffour	146	\ltx@zero	3, 116, 176, 413, 419, 447, 449, 451, 548, 569, 574, 580, 588
\ltx@firstofone	3, 140, 284, 480, 486	\LTxcmds@@ifnextchar	422, 429
\ltx@firstofthree	143, 500, 516	\LTxcmds@@ParseVersion	361, 363
\ltx@firstoftwo	141, 269, 277, 290, 295, 313, 350, 504, 520, 575, 594	\LTxcmds@AtEnd	95, 96, 115, 610
\ltx@fourthoffour	149	\LTxcmds@CarNum	163, 166
\ltx@GlobalAppendToMacro	7, 382	\LTxcmds@CarNumFinish	175
\ltx@gobble	3, 124, 282, 496, 498, 509, 512, 514, 540	\LTxcmds@CdrNum	181, 184
\ltx@gobblefour	127	\LTxcmds@CharToken	412, 418
\ltx@GobbleNum	3, 128, 182, 722, 724, 727	\LTxcmds@Cm	169
\ltx@gobblethree	126	\LTxcmds@Cx	172
\ltx@gobbletwo	125	\LTxcmds@Gm	137
\ltx@hashchar	199	\LTxcmds@GobbleNum	131, 134
\ltx@ifblank	7, 506, 536, 951, 969, 973	\LTxcmds@gtemp	396, 397, 398, 400, 405, 406, 408
\ltx@ifBoxEmpty	8, 564, 608, 1101	\LTxcmds@ifempty	535
\ltx@ifBoxVoidOrEmpty	8, 604, 1123	\LTxcmds@ifempty@	534
\ltx@ifclasslater	6, 353	\LTxcmds@iflater	323, 335
\ltx@ifclassloaded	6, 315	\LTxcmds@ifnextchar	415, 417, 432
\ltx@ifempty	7, 492, 525, 551, 905, 915, 919	\LTxcmds@LetToken	415, 418, 421, 432
\ltx@iffilelater	321, 354, 357	\LTxcmds@newglobalif	240, 242
\ltx@iffileloaded	6, 312, 316, 319, 322	\LTxcmds@newif	218, 220
\ltx@ifnextchar	7, 410	\LTxcmds@num	132, 164, 478
		\LTxcmds@ParseVersion	325, 331, 360, 367
		\LTxcmds@SpaceToken	421, 436
		\LTxcmds@temp	492, 523

<code>\LTXcmds@VoidBox</code>	455, 456, 458, 460, 463	<code>\StuffEmpty</code>	911, 913, 921, 957, 960, 963, 965, 975
<code>\LTXcmds@zapspace</code>	548, 550		
<b>M</b>		<b>T</b>	
<code>\meaning</code>	307, 793, 794, 843, 844	<code>\Test</code>	684, 707, 723, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 757, 769, 770, 774, 779, 783, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 829, 830, 833, 855, 856, 857, 858, 859, 860, 861, 862, 863, 886, 887
<code>\msg</code>	721, 722, 731, 733, 734, 739, 740, 782, 791, 793, 794, 799, 800, 832, 841, 843, 844, 849, 850, 902, 905, 922, 925, 951, 976, 979, 1015, 1018, 1030, 1035, 1036, 1037, 1080, 1107, 1129	<code>\test</code>	906, 931, 932, 933, 934, 935, 936, 937, 938, 939, 940, 941, 942, 943, 944, 945, 946, 947, 952, 985, 986, 987, 988, 989, 990, 991, 992, 993, 994, 996, 997, 998, 999, 1019, 1045, 1051, 1052, 1053, 1054, 1055, 1056, 1057, 1058, 1059, 1060, 1061, 1062, 1063, 1090, 1140, 1141, 1143, 1145, 1147, 1149, 1151, 1153, 1155, 1157
<b>N</b>		<code>\TestEmpty</code>	917, 921, 971, 975
<code>\N</code>	904, 913, 919, 965, 973	<code>\TestExpected</code>	1022, 1029, 1033, 1037, 1100, 1106, 1121, 1122, 1128
<code>\next</code>	594, 596, 599, 601, 638, 640, 642	<code>\TestInput</code>	1021, 1032, 1035
<code>\number</code>	324, 330, 481, 673	<code>\TestN</code>	904
<code>\numexpr</code>	487	<code>\TestResult</code>	1025, 1029, 1034, 1036, 1102, 1104, 1106, 1124, 1126, 1128
<b>P</b>		<code>\TestY</code>	903
<code>\PackageInfo</code>	26	<code>\the</code>	77, 78, 79, 80, 81, 82, 83, 84, 97, 392, 406, 419, 424, 487, 651, 671, 672
<code>\par</code>	770, 830, 887, 934, 946, 947, 991, 992, 998, 1061, 1064	<code>\TMP@EnsureCode</code>	94, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114
<code>\pdflastmatch</code>	374, 375, 376	<code>\toks</code>	391, 392, 405, 406, 413, 414, 419, 424
<code>\pdfmatch</code>	368	<b>U</b>	
<code>\penalty</code>	581, 589, 1154, 1156	<code>\unhbox</code>	451, 463, 466
<code>\ProvidesPackage</code>	19, 67	<code>\unhcopy</code>	570, 582, 590
<b>Q</b>		<code>\unvcopy</code>	570, 582, 590
<code>\quitvmode</code>	472	<b>V</b>	
<b>R</b>		<code>\vbox</code>	569, 580, 588, 1146, 1148
<code>\RangeCatcodeCheck</code>	665, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704	<code>\voidb@x</code>	466
<code>\RangeCatcodeInvalid</code>	657, 685, 686, 687, 688	<b>W</b>	
<code>\repeat</code>	632, 644, 655, 663, 678	<code>\wd</code>	1112, 1134
<code>\RequirePackage</code>	718, 900, 1013, 1078	<code>\write</code>	23, 52, 721, 902, 1015, 1080
<code>\RestoreCatcodes</code>	646, 649, 650, 705	<b>X</b>	
<code>\romannumeral</code>	129, 132, 161, 164, 180, 494, 507, 526, 537, 547	<code>\x</code>	14, 15, 18, 22, 26, 28, 51, 56, 66, 75, 87, 430, 435, 437, 440, 995, 1000, 1044, 1047, 1093, 1094, 1095
<b>S</b>		<b>Y</b>	
<code>\setbox</code>	447, 449, 458, 460, 569, 580, 588, 1083, 1089, 1099, 1120, 1142, 1144, 1146, 1148, 1150, 1152, 1154, 1156	<code>\Y</code>	903, 911, 919, 957, 960, 963, 973
<code>\space</code>	671, 672, 680, 1016, 1035, 1036, 1037, 1063, 1064, 1065, 1066		
<code>\SpaceTwo</code>	948, 949, 950, 962		
<code>\StrA</code>	724, 729, 739, 784, 789, 799, 834, 839, 849		
<code>\StrAA</code>	726, 730, 733, 786, 790, 793, 836, 840, 843		
<code>\StrB</code>	728, 729, 730, 734, 740, 788, 789, 790, 794, 800, 838, 839, 840, 844, 850		
<code>\Stuff</code>	909, 910, 925, 955, 956, 959, 962, 979		