

# The ltxcmds package

Heiko Oberdiek\*

2019/12/15 v1.24

## Abstract

The package `ltxcmds` exports some utility macros from the `LATEX` kernel into a separate namespace and also provides them for other formats such as `plain-TEX`.

## Contents

<b>1</b>	<b>Documentation</b>	<b>3</b>
1.1	Introduction . . . . .	3
1.2	Numbers . . . . .	3
1.3	Scratch registers . . . . .	3
1.4	Argument killers . . . . .	3
1.5	Argument grabbers . . . . .	4
1.6	List helpers . . . . .	5
1.7	Tail recursion . . . . .	5
1.8	Empty macro . . . . .	6
1.9	Characters . . . . .	6
1.10	Boolean switch . . . . .	6
1.11	Command definitions . . . . .	6
1.12	Stripping . . . . .	7
1.13	File management . . . . .	7
	1.13.1 File extensions . . . . .	7
	1.13.2 Load check . . . . .	7
	1.13.3 Version date check . . . . .	8
1.14	Macro additions . . . . .	8
1.15	Next character detection . . . . .	8
1.16	<code>\ltx@leavevmode</code> , <code>\ltx@ombox</code> . . . . .	9
1.17	Expandable test for emptiness . . . . .	9
1.18	Stripping spaces . . . . .	9
1.19	Check for emptiness of boxes . . . . .	10
<b>2</b>	<b>Implementation</b>	<b>10</b>
2.1	Identification . . . . .	10
2.2	Numbers . . . . .	12
2.3	Scratch registers . . . . .	12
2.4	Argument killers . . . . .	14
2.5	Argument grabbers . . . . .	15
2.6	List helpers . . . . .	15
2.7	Tail recursion . . . . .	17

---

\*Please report any issues at <https://github.com/ho-tex/ltxcmds/issues>

2.8	Empty macro	17
2.9	Characters	17
2.10	Boolean switch	18
2.11	Command definitions	19
2.12	Stripping	20
2.13	File management	20
2.13.1	File extensions	20
2.13.2	Load check	20
2.13.3	Version date check	21
2.14	Macro additions	22
2.15	Next character detection	23
2.16	<code>\ltx@leavevmode</code> , <code>\ltx@inbox</code>	24
2.17	Help macros	25
2.18	Expandable test for emptiness	25
2.18.1	Vanilla $\TeX$	25
2.18.2	With <code>\detokenize</code>	26
2.18.3	<code>\ltx@ifblank</code>	26
2.19	<code>\ltx@zapspace</code>	27
2.20	<code>\ltx@ifboxempty</code>	27
<b>3</b>	<b>Installation</b>	<b>28</b>
3.1	Download	28
3.2	Bundle installation	29
3.3	Package installation	29
3.4	Refresh file name databases	29
3.5	Some details for the interested	29
<b>4</b>	<b>References</b>	<b>30</b>
<b>5</b>	<b>History</b>	<b>30</b>
	[2009/08/05 v1.0]	30
	[2009/12/12 v1.1]	30
	[2010/01/28 v1.2]	31
	[2010/03/01 v1.3]	31
	[2010/03/09 v1.4]	31
	[2010/04/08 v1.5]	31
	[2010/04/16 v1.6]	31
	[2010/04/26 v1.7]	31
	[2010/09/11 v1.8]	31
	[2010/10/25 v1.9]	31
	[2010/10/31 v1.10]	31
	[2010/11/12 v1.11]	32
	[2010/12/02 v1.12]	32
	[2010/12/04 v1.13]	32
	[2010/12/07 v1.14]	32
	[2010/12/12 v1.15]	32
	[2011/02/04 v1.16]	32
	[2011/02/05 v1.17]	32
	[2011/03/16 v1.18]	32
	[2011/04/14 v1.19]	32
	[2011/04/18 v1.20]	32
	[2011/08/22 v1.21]	32
	[2011/11/09 v1.22]	33
	[2016/05/16 v1.23]	33

# 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

<code>\ltx@zero</code>	→ 0
<code>\ltx@one</code>	→ 1
<code>\ltx@two</code>	→ 2
<code>\ltx@ccclv</code>	→ 255
<code>\ltx@minusone</code>	→ -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 Scratch registers

Following the conventions of plain  $\TeX$  and  $\LaTeX$  the first ten registers are free to use. Even numbered registers are for local, odd numbered for global use.

<code>\ltx@(Loc, Glob)(Toks, Dimen, Skip)(A, B, C, D, E)</code>
---

The name consists of the prefix `\ltx@`, then `Loc` or `Glob` for local or global usage follows. The register type is given by `Toks` for token register, `Dimen` for dimen register and `Skip` for skip register. As last part the registers are numbered from A to E. Example: `\ltx@LocToksA`.

Since 2011/04/14 v1.19.

## 1.4 Argument killers

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

<code>\ltx@GobbleNum {⟨num⟩} {⟨1⟩} {⟨2⟩} ... {⟨⟨num⟩⟩}</code>	→
---	---

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

## 1.5 Argument grabbers

<code>\ltx@firstofone {⟨1⟩}</code>	→	$\langle 1 \rangle$
<code>\ltx@firstoftwo {⟨1⟩} {⟨2⟩}</code>	→	$\langle 1 \rangle$
<code>\ltx@secondoftwo {⟨1⟩} {⟨2⟩}</code>	→	$\langle 2 \rangle$
<code>\ltx@firstofthree {⟨1⟩} {⟨2⟩} {⟨3⟩}</code>	→	$\langle 1 \rangle$
<code>\ltx@secondofthree {⟨1⟩} {⟨2⟩} {⟨3⟩}</code>	→	$\langle 2 \rangle$
<code>\ltx@thirdofthree {⟨1⟩} {⟨2⟩} {⟨3⟩}</code>	→	$\langle 3 \rangle$
<code>\ltx@firstoffour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}</code>	→	$\langle 1 \rangle$
<code>\ltx@secondoffour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}</code>	→	$\langle 2 \rangle$
<code>\ltx@thirdoffour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}</code>	→	$\langle 3 \rangle$
<code>\ltx@fourthoffour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩}</code>	→	$\langle 4 \rangle$

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.

## 1.6 List helpers

<code>\ltx@carzero ... \@nil</code>	→
<code>\ltx@cdrzero ... \@nil</code>	→ ...

<code>\ltx@car {⟨1⟩} ... \@nil</code>	→ ⟨1⟩
<code>\ltx@cdr {⟨1⟩} ... \@nil</code>	→ ...

<code>\ltx@cartwo {⟨1⟩} {⟨2⟩} ... \@nil</code>	→ ⟨1⟩⟨2⟩
<code>\ltx@carsecond {⟨1⟩} {⟨2⟩} ... \@nil</code>	→ ⟨2⟩
<code>\ltx@cdrtwo {⟨1⟩} {⟨2⟩} ... \@nil</code>	→ ...

<code>\ltx@carthree {⟨1⟩} {⟨2⟩} {⟨3⟩} ... \@nil</code>	→ ⟨1⟩⟨2⟩⟨3⟩
<code>\ltx@carthird {⟨1⟩} {⟨2⟩} {⟨3⟩} ... \@nil</code>	→ ⟨3⟩
<code>\ltx@cdrthree {⟨1⟩} {⟨2⟩} {⟨3⟩} ... \@nil</code>	→ ...

<code>\ltx@carfour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩} ... \@nil</code>	→ ⟨1⟩⟨2⟩⟨3⟩⟨4⟩
<code>\ltx@carfourth {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩} ... \@nil</code>	→ ⟨4⟩
<code>\ltx@cdrfour {⟨1⟩} {⟨2⟩} {⟨3⟩} {⟨4⟩} ... \@nil</code>	→ ...

<code>\ltx@CarNum {⟨num⟩} {⟨1⟩} ... {⟨⟨num⟩⟩} {⟨⟨num⟩+1⟩} ... \@nil</code>	→ {⟨1⟩} ... {⟨⟨num⟩⟩} ...
<code>\ltx@CarNumth {⟨num⟩} {⟨1⟩} ... {⟨⟨num⟩⟩} {⟨⟨num⟩+1⟩} ... \@nil</code>	→ {⟨⟨num⟩⟩} ...
<code>\ltx@CdrNum {⟨num⟩} {⟨1⟩} ... {⟨⟨num⟩⟩} {⟨⟨num⟩+1⟩} ... \@nil</code>	→ {⟨⟨num⟩+1⟩} ...

Macros with uppercase letters are expandable in two expansion steps. Changes in version 2019/12/15 v1.24:

- Macros `\ltx@carsecond`, `\ltx@carthird`, `\ltx@carfourth`, `\ltx@CarNumth` added.
- Macros `\ltx@cdr`, `\ltx@cdrtwo`, `\ltx@cdrthree`, `\ltx@cdrfour`, `\ltx@CdrNum` are expandable in two expansion steps and retain spaces and braces after the first gobbled arguments.

## 1.7 Tail recursion

<code>\ltx@ReturnAfterFi {⟨1⟩} \fi</code>	→ \fi ⟨1⟩
<code>\ltx@ReturnAfterElseFi {⟨1⟩} \else {⟨2⟩} \fi</code>	→ \fi ⟨1⟩

## 1.8 Empty macro

<code>\ltx@empty</code>	→
-------------------------	---

## 1.9 Characters

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

## 1.10 Boolean switch

<code>\ltx@newif {&lt;cmd&gt;}</code>
---------------------------------------

`\ltx@newif` defines a new boolean switch `<cmd>` like `\newif`. Unlike plain  $\text{T}_{\text{E}}\text{X}$ 's `\newif`, `\ltx@newif` is not `\outer`. The command `<cmd>` must start with the two characters `if`.

<code>\ltx@newglobalif {&lt;cmd&gt;}</code>
---

`\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.

## 1.11 Command definitions

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

If  $\varepsilon\text{-T}_{\text{E}}\text{X}$  is available, `\ifcurname` 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”.

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

If  $\varepsilon\text{-T}_{\text{E}}\text{X}$  is available, `\ifcurname` 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  $\varepsilon\text{-T}_{\text{E}}\text{X}$ .

<code>\ltx@LocalExpandAfter</code>
------------------------------------

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 `\curname` and let the command undefined after the expansion step.

## 1.12 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 `\pdfastmatch`. Macro `\ltx@RemovePrefix` has the same meaning as L<sup>A</sup>T<sub>E</sub>X’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 L<sup>A</sup>T<sub>E</sub>X’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 then sanitized contents is stored into the macro again. Since version 1.12.

## 1.13 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.

### 1.13.1 File extensions

```
\ltx@clsextension  
\ltx@pkgeextension
```

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}`.

### 1.13.2 Load check

```
\ltx@ifclassloaded {<class>} {<yes>} {<no>}  
\ltx@ifpackageloaded {<package>} {<yes>} {<no>}
```

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}`.

```
\ltx@iffileloaded {<file>} {<yes>} {<no>}
```

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.

### 1.13.3 Version date check

```
\ltx@ifclasslater {<class>} {<date>} {<yes>} {<no>}
\ltx@ifpackagelater {<package>} {<date>} {<yes>} {<no>}
\ltx@iffilelater {<file>} {<date>} {<yes>} {<no>}
```

If a `\ProvidesClass`/`\ProvidesPackage`/`\ProvidesFile` command with exact the same class/package/file was executed before with an optional argument that starts with a  $\LaTeX$  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  $\LaTeX$  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 `pdfTeX`'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  $\LaTeX 2_{\epsilon}$  in 1994.

### 1.14 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 beforehand. Due to a bug `<addition>` must not contain `\par` before version 2010/10/25 v1.9.

```
\ltx@GlobalPrependToMacro {<cmd>} {<addition>}
\ltx@LocalPrependToMacro {<cmd>} {<addition>}
```

The `<addition>` is prepended to the parameterless macro `<cmd>`. If `<cmd>` is undefined or has the meaning `\relax`, then it will be initialized as empty macro beforehand. The macros were added in version 2011/08/22 v1.21.

### 1.15 Next character detection

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

If next character is `<char>` then `<yes>` is called, otherwise `<no>`. The character is not removed. Spaces are silently removed when looking for `<char>` as  $\LaTeX$ 's version `\kernel@ifnextchar` does. But there are also small differences:

- The space can be used as `<char>`. In this case optional spaces before `<char>` are not supported of course.
- If the optional space is a command that is a character (defined by `\let` or `\futurelet`), then `\kernel@ifnextchar` breaks with an  $\TeX$  error. `\ltx@ifnextchar` silently removes this token as optional space.

Since 2010/03/01 v1.3.

`\ltx@ifnextchar@nospace`  $\{\langle char \rangle\} \{\langle yes \rangle\} \{\langle no \rangle\}$

Macro `\ltx@ifnextchar@nospace` behaves like macro `\ltx@ifnextchar` with the exception that optional spaces are not supported before  $\langle char \rangle$ . Since 2011/04/14 v1.19.

## 1.16 `\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`.

## 1.17 Expandable test for emptiness

`\ltx@ifempty`  $\{\langle stuff \rangle\} \{\langle yes \rangle\} \{\langle no \rangle\}$

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

`\ltx@ifblank`  $\{\langle stuff \rangle\} \{\langle yes \rangle\} \{\langle no \rangle\}$

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

## 1.18 Stripping spaces

`\ltx@zapspace`  $\{\langle stuff \rangle\}$

Macro `\ltx@zapspace` strips spaces from  $\langle stuff \rangle$  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  $\langle stuff \rangle$ .
- 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.

## 1.19 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 LATEX.
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 [2019/12/15 v1.24 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@cclv
120 \chardef\ltx@cclv=255 %

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

```

## 2.3 Scratch registers

```

\ltx@LocToksA
124 \toksdef\ltx@LocToksA=0 %

\ltx@LocToksB
125 \toksdef\ltx@LocToksB=2 %

```

```

\ltx@LocToksC
126 \toksdef\ltx@LocToksC=4 %

\ltx@LocToksD
127 \toksdef\ltx@LocToksD=6 %

\ltx@LocToksE
128 \toksdef\ltx@LocToksE=8 %

\ltx@GlobToksA
129 \toksdef\ltx@GlobToksA=1 %

\ltx@GlobToksB
130 \toksdef\ltx@GlobToksB=3 %

\ltx@GlobToksC
131 \toksdef\ltx@GlobToksC=5 %

\ltx@GlobToksD
132 \toksdef\ltx@GlobToksD=7 %

\ltx@GlobToksE
133 \toksdef\ltx@GlobToksE=9 %

\ltx@LocDimenA
134 \dimendef\ltx@LocDimenA=0 %

\ltx@LocDimenB
135 \dimendef\ltx@LocDimenB=2 %

\ltx@LocDimenC
136 \dimendef\ltx@LocDimenC=4 %

\ltx@LocDimenD
137 \dimendef\ltx@LocDimenD=6 %

\ltx@LocDimenE
138 \dimendef\ltx@LocDimenE=8 %

\ltx@GlobDimenA
139 \dimendef\ltx@GlobDimenA=1 %

\ltx@GlobDimenB
140 \dimendef\ltx@GlobDimenB=3 %

\ltx@GlobDimenC
141 \dimendef\ltx@GlobDimenC=5 %

\ltx@GlobDimenD
142 \dimendef\ltx@GlobDimenD=7 %

\ltx@GlobDimenE
143 \dimendef\ltx@GlobDimenE=9 %

\ltx@LocSkipA
144 \skipdef\ltx@LocSkipA=0 %

```

```

\ltx@LocSkipB
145 \skipdef\ltx@LocSkipB=2 %

\ltx@LocSkipC
146 \skipdef\ltx@LocSkipC=4 %

\ltx@LocSkipD
147 \skipdef\ltx@LocSkipD=6 %

\ltx@LocSkipE
148 \skipdef\ltx@LocSkipE=8 %

\ltx@GlobSkipA
149 \skipdef\ltx@GlobSkipA=1 %

\ltx@GlobSkipB
150 \skipdef\ltx@GlobSkipB=3 %

\ltx@GlobSkipC
151 \skipdef\ltx@GlobSkipC=5 %

\ltx@GlobSkipD
152 \skipdef\ltx@GlobSkipD=7 %

\ltx@GlobSkipE
153 \skipdef\ltx@GlobSkipE=9 %

```

## 2.4 Argument killers

```

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

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

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

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

\ltx@GobbleNum
158 \def\ltx@GobbleNum#1{%
159   \romannumeral
160   \csname ltx@zero%
161   \expandafter\LTXcmds@GobbleNum
162   \romannumeral\LTXcmds@num{#1}000{m\endcsname}%
163 }

\LTXcmds@GobbleNum
164 \def\LTXcmds@GobbleNum#1{%
165   \csname LTXcmds@G#1\LTXcmds@GobbleNum
166 }

\LTXcmds@Gm
167 \long\def\LTXcmds@Gm#1{%
168   \endcsname
169 }

```

## 2.5 Argument grabbers

```
\ltx@firstofone
170 \long\def\ltx@firstofone#1{#1}

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

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

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

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

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

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

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

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

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

## 2.6 List helpers

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

\ltx@cdrzero
181 \long\def\ltx@cdrzero#1\@nil{#1}

\ltx@cdrzero
182 \def\ltx@cdrzero{%
183   \romannumeral\ltx@cdrzero\ltx@zero
184 }

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

\ltx@cdr
186 \long\def\ltx@cdr#1{%
187   \romannumeral\ltx@cdr#1\ltx@zero
188 }

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

```

\ltx@carsecond
    190 \long\def\ltx@carsecond#1#2#3\@nil{#2}

\ltx@cdrtwo
    191 \long\def\ltx@cdrtwo#1#2{%
    192 \romannumeral\LTxcmds@cdrzero\ltx@zero
    193 }

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

\ltx@carthird
    195 \long\def\ltx@carthird#1#2#3#4\@nil{#3}

\ltx@cdrthree
    196 \long\def\ltx@cdrthree#1#2#3{%
    197 \romannumeral\LTxcmds@cdrzero\ltx@zero
    198 }

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

\ltx@carfourth
    200 \long\def\ltx@carfourth#1#2#3#4#5\@nil{#4}

\ltx@cdrfour
    201 \long\def\ltx@cdrfour#1#2#3#4{%
    202 \romannumeral\LTxcmds@cdrzero\ltx@zero
    203 }

\ltx@CarNum
    204 \def\ltx@CarNum#1{%
    205 \romannumeral
    206 \csname LTxcmds@CarNumFinish%
    207 \expandafter\LTxcmds@CarNum
    208 \romannumeral\LTxcmds@num{#1}000{x\endcsname}%
    209 }

\LTxcmds@CarNum
    210 \def\LTxcmds@CarNum#1{%
    211 \csname LTxcmds@C#1\LTxcmds@CarNum
    212 }

\LTxcmds@Cm
    213 \long\def\LTxcmds@Cm#1#2{%
    214 \endcsname{#1#2}%
    215 }

\LTxcmds@Cx
    216 \def\LTxcmds@Cx#1{%
    217 \endcsname{#1}%
    218 }

\LTxcmds@CarNumFinish
    219 \long\def\LTxcmds@CarNumFinish#1#2\@nil{#1%
    220 \ltx@zero
    221 #1%
    222 }

```

```

\ltx@CarNumth
223 \def\ltx@CarNumth#1{%
224   \romannumeral
225   \expandafter\expandafter\expandafter
226   \LTXcmds@CarNumth
227   \ltx@GobbleNum{#1}{}%
228 }

```

```

\LTXcmds@CarNumth
229 \long\def\LTXcmds@CarNumth#1#2\@nil{%
230   \ltx@zero
231   #1%
232 }

```

```

\ltx@CdrNum
233 \def\ltx@CdrNum#1{%
234   \romannumeral%
235   \expandafter\expandafter\expandafter\ltx@cdrzero
236   \expandafter\expandafter\expandafter\ltx@zero
237   \ltx@GobbleNum{#1}%
238 }

```

## 2.7 Tail recursion

```

\ltx@ReturnAfterFi
239 \long\def\ltx@ReturnAfterFi#1\fi{\fi#1}

```

```

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

```

## 2.8 Empty macro

```

\ltx@empty
241 \def\ltx@empty{}

```

## 2.9 Characters

```

\ltx@space
242 \def\ltx@space{ }

```

```

\ltx@percentchar
243 \begingroup
244   \lccode'0='%\relax
245 \lowercase{\endgroup
246   \def\ltx@percentchar{0}%
247 }

```

```

\ltx@backslashchar
248 \begingroup
249   \lccode'0='\\relax
250 \lowercase{\endgroup
251   \def\ltx@backslashchar{0}%
252 }

```

`\ltx@hashchar`

```
253 \begingroup
254 \lccode'0='#\relax
255 \lowercase{\endgroup
256 \def\ltx@hashchar{0}%
257 }
```

`\ltx@leftbracechar`

```
258 \begingroup
259 \lccode'0='{}\relax
260 \lowercase{\endgroup
261 \def\ltx@leftbracechar{0}%
262 }
```

`\ltx@rightbracechar`

```
263 \begingroup
264 \lccode'0='}\relax
265 \lowercase{\endgroup
266 \def\ltx@rightbracechar{0}%
267 }
```

## 2.10 Boolean switch

`\ltx@newif`

```
268 \def\ltx@newif#1{%
269 \begingroup
270 \escapechar=-1 %
271 \expandafter\endgroup
272 \expandafter\LTXcmds@newif\string#1\@nil
273 }
```

`\LTXcmds@newif`

```
274 \begingroup
275 \escapechar=-1 %
276 \expandafter\endgroup
277 \expandafter\def\expandafter\LTXcmds@newif\string\if#1\@nil{%
278 \expandafter\edef\csname#1true\endcsname{%
279 \let
280 \expandafter\noexpand\csname if#1\endcsname
281 \noexpand\iftrue
282 }%
283 \expandafter\edef\csname#1false\endcsname{%
284 \let
285 \expandafter\noexpand\csname if#1\endcsname
286 \noexpand\iffalse
287 }%
288 \csname#1false\endcsname
289 }
```

`\ltx@newglobalif`

```
290 \def\ltx@newglobalif#1{%
291 \begingroup
292 \escapechar=-1 %
293 \expandafter\endgroup
294 \expandafter\LTXcmds@newglobalif\string#1\@nil
295 }
```

`\LTXcmds@newglobalif`

```
296 \begingroup
297   \escapechar=-1 %
298 \expandafter\endgroup
299 \expandafter
300 \def\expandafter\LTXcmds@newglobalif\string\if#1\@nil{%
301   \expandafter\edef\csname#1true\endcsname{%
302     \global\let
303     \expandafter\noexpand\csname if#1\endcsname
304     \noexpand\iftrue
305   }%
306 \expandafter\edef\csname#1false\endcsname{%
307   \global\let
308   \expandafter\noexpand\csname if#1\endcsname
309   \noexpand\iffalse
310 }%
311 \csname#1false\endcsname
312 }
```

## 2.11 Command definitions

`\ltx@LocalExpandAfter`

```
313 \def\ltx@LocalExpandAfter{%
314   \begingroup
315   \expandafter\expandafter\expandafter
316   \endgroup
317   \expandafter
318 }

319 \ltx@LocalExpandAfter
320 \ifx\csname ifcsname\endcsname\relax
```

`\ltx@ifundefined`

```
321 \def\ltx@ifundefined#1{%
322   \expandafter\ifx\csname #1\endcsname\relax
323     \expandafter\ltx@firstoftwo
324   \else
325     \expandafter\ltx@secondoftwo
326   \fi
327 }
```

`\ltx@ifUndefined`

```
328 \def\ltx@ifUndefined#1{%
329   \begingroup\expandafter\expandafter\expandafter\endgroup
330   \expandafter\ifx\csname #1\endcsname\relax
331     \expandafter\ltx@firstoftwo
332   \else
333     \expandafter\ltx@secondoftwo
334   \fi
335 }%

336 \expandafter\ltx@gobble
337 \else
338 \expandafter\ltx@firstofone
339 \fi
340 {%
```

```

\ltx@ifundefined
341 \def\ltx@ifundefined#1{%
342   \ifcsname #1\endcsname
343   \expandafter\ifx\csname #1\endcsname\relax
344   \expandafter\expandafter\expandafter\ltx@firstoftwo
345   \else
346   \expandafter\expandafter\expandafter\ltx@secondoftwo
347   \fi
348   \else
349   \expandafter\ltx@firstoftwo
350   \fi
351 }%

```

```

\ltx@ifUndefined
352 \let\ltx@ifUndefined\ltx@ifundefined
353 }

```

## 2.12 Stripping

```

\ltx@RemovePrefix
354 \def\ltx@RemovePrefix#1>{}

\ltx@StripPrefix
355 \def\ltx@StripPrefix{%
356   \expandafter\ltx@RemovePrefix
357 }

```

```

\ltx@onelevel@sanitize
358 \def\ltx@onelevel@sanitize#1{%
359   \edef#1{%
360     \expandafter
361     \ltx@RemovePrefix\meaning#1%
362   }%
363 }

```

## 2.13 File management

### 2.13.1 File extensions

```

\ltx@clsextension
364 \def\ltx@clsextension{cls}

\ltx@pkgextension
365 \def\ltx@pkgextension{sty}

```

### 2.13.2 Load check

```

\ltx@iffileloaded
366 \def\ltx@iffileloaded#1{%
367   \ltx@ifundefined{ver@#1}\ltx@secondoftwo\ltx@firstoftwo
368 }

\ltx@ifclassloaded
369 \def\ltx@ifclassloaded#1{%
370   \ltx@iffileloaded{#1.\ltx@clsextension}%
371 }

```

\ltx@ifpackageloaded

```
372 \def\ltx@ifpackageloaded#1{%  
373   \ltx@iffileloaded{#1.\ltx@pkgextension}%  
374 }
```

### 2.13.3 Version date check

\ltx@iffilelater

```
375 \def\ltx@iffilelater#1#2{%  
376   \ltx@iffileloaded{#1}%  
377   \expandafter\LTXcmds@IfLater\expandafter{%  
378     \number  
379     \expandafter\expandafter\expandafter\LTXcmds@ParseVersion  
380     \expandafter\expandafter\expandafter{%  
381       \csname ver@#1\endcsname  
382     }%  
383   \expandafter}\expandafter{%  
384     \number  
385     \expandafter\LTXcmds@ParseVersion\expandafter{#2}%  
386   }%  
387 } \ltx@secondoftwo  
388 }
```

\LTXcmds@IfLater

```
389 \def\LTXcmds@IfLater#1#2{%  
390   \ifcase 0%  
391     \ifnum#1<19940101 %  
392     \else  
393       \ifnum#2<19940101 %  
394       \else  
395         \ifnum#2>#1 %  
396         \else  
397           1%  
398         \fi  
399       \fi  
400     \fi  
401     \ltx@space  
402     \expandafter\ltx@secondoftwo  
403   \else  
404     \expandafter\ltx@firstoftwo  
405   \fi  
406 }
```

\ltx@ifclasslater

```
407 \def\ltx@ifclasslater#1{%  
408   \ltx@iffilelater{#1.\ltx@clsextension}%  
409 }
```

\ltx@ifpackagelater

```
410 \def\ltx@ifpackagelater#1{%  
411   \ltx@iffilelater{#1.\ltx@pkgextension}%  
412 }  
  
413 \ltx@ifundefined{pdfmatch}{%
```

\LTXcmds@ParseVersion

```
414   \def\LTXcmds@ParseVersion#1{%  
415     \LTXcmds@@ParseVersion#10000/00/00\@nil  
416   }%
```

\LTXcmds@ParseVersion

```
417 \def\LTXcmds@ParseVersion#1#2#3#4/#5#6/#7#8#9\@nil{%
418   #1#2#3#4#5#6#7#8%
419 }%
420 }{%
```

\LTXcmds@ParseVersion

```
421 \def\LTXcmds@ParseVersion#1{%
422   \ifnum\pdfmatch{%
423     ~%
424     (199[4-9] | [2-9] [0-9] [0-9] [0-9])/%
425     (0[1-9] | 1[0-2])/%
426     (0[1-9] | [1-2] [0-9] | 3[0-1])%
427   }{#1}=1 %
428   \ltx@StripPrefix\pdflastmatch1 %
429   \ltx@StripPrefix\pdflastmatch2 %
430   \ltx@StripPrefix\pdflastmatch3 %
431   \else
432     0%
433   \fi
434 }%
435 }
```

## 2.14 Macro additions

\ltx@GlobalAppendToMacro

```
436 \long\def\ltx@GlobalAppendToMacro#1#2{%
437   \ifx\ltx@undefined#1%
438     \let#1\ltx@empty
439   \else
440     \ifx\relax#1%
441       \let#1\ltx@empty
442     \fi
443   \fi
444   \begingroup
445     \ltx@LocToksA\expandafter{#1#2}%
446     \xdef#1{\the\ltx@LocToksA}%
447   \endgroup
448 }
```

\ltx@LocalAppendToMacro

```
449 \long\def\ltx@LocalAppendToMacro#1#2{%
450   \global\let\LTXcmds@gtemp#1%
451   \ifx\ltx@undefined\LTXcmds@gtemp
452     \global\let\LTXcmds@gtemp\ltx@empty
453   \else
454     \ifx\relax\LTXcmds@gtemp
455       \global\let\LTXcmds@gtemp\ltx@empty
456     \fi
457   \fi
458   \begingroup
459     \ltx@LocToksA\expandafter{\LTXcmds@gtemp#2}%
460     \xdef\LTXcmds@gtemp{\the\ltx@LocToksA}%
461   \endgroup
462   \let#1\LTXcmds@gtemp
463 }
```

`\ltx@GlobalPrependToMacro`

```
464 \long\def\ltx@GlobalPrependToMacro#1#2{%
465   \ifx\ltx@undefined#1%
466     \let#1\ltx@empty
467   \else
468     \ifx\relax#1%
469       \let#1\ltx@empty
470     \fi
471   \fi
472   \begingroup
473     \ltx@LocToksA{#2}%
474     \ltx@LocToksB\expandafter{#1}%
475     \xdef#1{\the\ltx@LocToksA\the\ltx@LocToksB}%
476   \endgroup
477 }
```

`\ltx@LocalPrependToMacro`

```
478 \long\def\ltx@LocalPrependToMacro#1#2{%
479   \global\let\LTXcmds@gtemp#1%
480   \ifx\ltx@undefined\LTXcmds@gtemp
481     \global\let\LTXcmds@gtemp\ltx@empty
482   \else
483     \ifx\relax\LTXcmds@gtemp
484       \global\let\LTXcmds@gtemp\ltx@empty
485     \fi
486   \fi
487   \begingroup
488     \ltx@LocToksA{#2}%
489     \ltx@LocToksB\expandafter{\LTXcmds@gtemp}%
490     \xdef\LTXcmds@gtemp{\the\ltx@LocToksA\the\ltx@LocToksB}%
491   \endgroup
492   \let#1\LTXcmds@gtemp
493 }
```

## 2.15 Next character detection

`\ltx@ifnextchar`

```
494 \long\def\ltx@ifnextchar#1#2#3{%
495   \begingroup
496   \let\LTXcmds@CharToken= #1\relax
497   \ltx@LocToksA{\endgroup#2}%
498   \ltx@LocToksB{\endgroup#3}%
499   \futurelet\LTXcmds@LetToken\LTXcmds@ifnextchar
500 }
```

`\LTXcmds@ifnextchar`

```
501 \def\LTXcmds@ifnextchar{%
502   \ifx\LTXcmds@LetToken\LTXcmds@CharToken
503     \the\expandafter\ltx@LocToksA
504   \else
505     \expandafter
506       \ifx\csname LTXcmds@LetToken\endcsname\LTXcmds@SpaceToken
507         \expandafter\expandafter\expandafter\LTXcmds@ifnextchar
508       \else
509         \the\expandafter\expandafter\expandafter\ltx@LocToksB
510     \fi
511   \fi
512 }
```

`\LTXcmds@ifnextchar` `\futurelet` does not distinguish between a character and a command that is a character (defined by using `\let` or `\futurelet`). Therefore the space is caught by `\romannumeral` with negative character constant that gobbles one optional space.

```
513 \def\LTXcmds@ifnextchar{%
514   \expandafter\futurelet
515   \expandafter\LTXcmds@LetToken
516   \expandafter\LTXcmds@ifnextchar
517   \romannumeral-'\.%
518 }
```

`\LTXcmds@SpaceToken`

```
519 \ltx@firstofone{\let\LTXcmds@SpaceToken= } %
```

`\ltx@ifnextchar@nospace`

```
520 \long\def\ltx@ifnextchar@nospace#1#2#3{%
521   \begingroup
522   \let\LTXcmds@CharToken= #1\relax
523   \ltx@LocToksA{\endgroup#2}%
524   \ltx@LocToksB{\endgroup#3}%
525   \futurelet\LTXcmds@LetToken\LTXcmds@ifnextchar@nospace
526 }
```

`\LTXcmds@ifnextchar@nospace`

```
527 \def\LTXcmds@ifnextchar@nospace{%
528   \the
529   \ifx\LTXcmds@LetToken\LTXcmds@CharToken
530     \expandafter\ltx@LocToksA
531   \else
532     \expandafter\ltx@LocToksB
533   \fi
534 }
```

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

`\ltx@leavevmode`

```
535 \ltx@ifundefined{quitvmode}{%
536   \ltx@ifundefined{leavevmode}{%
537     \ltx@ifundefined{voidb@x}{%
538       \ltx@ifundefined{newbox}{%
539         \def\ltx@leavevmode{%
540           \begingroup
541             \setbox\ltx@zero=\hbox{}%
542           \begingroup
543             \setbox\ltx@zero=\hbox{\box\ltx@zero}%
544           \endgroup
545           \unhbox\ltx@zero
546         \endgroup
547       }%
548     }{%
549       \csname newbox\endcsname\LTXcmds@VoidBox
550     \ifvoid\LTXcmds@VoidBox
551     \else
552       \setbox\LTXcmds@VoidBox=\hbox{}%
553     \begingroup
554       \setbox\LTXcmds@VoidBox=\hbox{\box\LTXcmds@VoidBox}%
555     \endgroup
```

```

556     \fi
557     \def\ltx@leavevmode{\unhbox\LTxcmds@VoidBox}%
558   }%
559 }{%
560   \def\ltx@leavevmode{\unhbox\voidb@x}%
561   }%
562 }{%
563   \let\ltx@leavevmode\leavevmode
564   }%
565 }{%
566   \let\ltx@leavevmode\quitvmode
567 }

```

\ltx@mbox

```

568 \def\ltx@mbox{%
569   \ltx@leavevmode
570   \hbox
571 }

```

## 2.17 Help macros

\LTxcmds@num

```

572 \ltx@ifundefined{numexpr}{%
573   \def\LTxcmds@num#1{%
574     \expandafter\ltx@firstofone\expandafter{%
575       \number#1%
576     }%
577   }%
578 }{%
579   \def\LTxcmds@num#1{%
580     \expandafter\ltx@firstofone\expandafter{%
581       \the\numexpr#1%
582     }%
583   }%
584 }

```

## 2.18 Expandable test for emptiness

```
585 \ltx@ifundefined{detokenize}{%
```

### 2.18.1 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).

```

586   \def\LTxcmds@temp#1{%
587     \long\def\ltx@ifempty##1{%
588       \romannumeral0%
589       \iffalse\fi
590       \expandafter\ltx@gobble\expandafter{%
591         \expandafter{\string##1}%
592         \expandafter\ltx@gobble\string
593       }%
594       \expandafter\ltx@firstofthree\expandafter
595       {\iffalse}\fi
596       \expandafter#1\ltx@secondoftwo

```

```

597     }%
598     \expandafter#1\ltx@firstoftwo
599     }%

```

`\ltx@ifblank`

```

600     \long\def\ltx@ifblank##1{%
601         \romannumeral0%
602         \iffalse{\fi
603             \expandafter\expandafter\expandafter\ltx@gobble
604             \expandafter\expandafter\expandafter{%
605                 \expandafter\expandafter\expandafter{%
606                     \expandafter\string\ltx@gobble##1.%
607                 }%
608             \expandafter\ltx@gobble\string
609         }%
610         \expandafter\ltx@firstofthree\expandafter
611         {\iffalse}\fi
612         \expandafter#1\ltx@secondoftwo
613     }%
614     \expandafter#1\ltx@firstoftwo
615 }%
616 }%
617 \LTXcmds@temp{ }%
618 }{%

```

### 2.18.2 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`

```

619     \long\def\ltx@ifempty#1{%
620         \romannumeral%
621         \csname
622             LTXcmds@ifempty%
623             \ifcat$\detokenize{#1}$%
624             @%
625             \fi
626         \endcsname
627     }%

```

`\LTXcmds@ifempty@`

```

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

```

`\LTXcmds@ifempty`

```

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

```

### 2.18.3 `\ltx@ifblank`

`\ltx@ifblank`

```

630     \long\def\ltx@ifblank#1{%
631         \romannumeral%
632         \csname
633             LTXcmds@ifempty%
634             \ifcat$\detokenize\expandafter{\ltx@gobble#1.}$%

```

```

635     @%
636     \fi
637     \endcsname
638 }%

639 }

```

## 2.19 \ltx@zapspace

\ltx@zapspace

```

640 \long\def\ltx@zapspace#1{%
641     \romannumeral
642     \LTXcmds@zapspace\ltx@zero#1 \@nil
643 }

```

\LTXcmds@zapspace

```

644 \long\def\LTXcmds@zapspace#1 #2\@nil{%
645     \ltx@ifempty{#2}{%
646         #1%
647     }{%
648         \LTXcmds@zapspace#1#2\@nil
649     }%
650 }

```

## 2.20 \ltx@ifboxempty

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

```

651 \ltx@ifundefined{lastnodetype}{%
652     \catcode'\$=9 %
653     \catcode'\&=14 %
654 }{%
655     \catcode'\$=14 %
656     \catcode'\&=9 %
657 }

```

\ltx@ifboxempty

```

658 \def\ltx@ifboxempty#1{%
659     \ifvoid#1\relax
660     \expandafter\ltx@secondoftwo
661     \else

```

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

```

662 & \begingroup
663 &     \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%
664 &         \ifhmode\unhcopy\else\unvcopy\fi#1\relax
665 &     \expandafter
666 &     }%
667 & \expandafter\endgroup
668 & \ifnum\lastnodetype<\ltx@zero
669 &     \expandafter\expandafter\expandafter\ltx@firstoftwo
670 & \else
671 &     \expandafter\expandafter\expandafter\ltx@secondoftwo
672 & \fi

```

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

```

673 $ \begingroup
674 $     \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%

```

```

675 $      \penalty\ltx@one
676 $      \ifhmode\unhcopy\else\unvcopy\fi#1\relax
677 $      \expandafter
678 $      }%
679 $      \ifnum\lastpenalty=\ltx@one
Box 0 has been changed and is restored by closing the group.
680 $      \endgroup
681 $      \begingroup
682 $      \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%
683 $      \penalty\ltx@two
684 $      \ifhmode\unhcopy\else\unvcopy\fi#1\relax
685 $      \expandafter
686 $      }%
687 $      \ifnum\lastpenalty=\ltx@two
688 $      \def\next{\endgroup\expandafter\ltx@firstoftwo}%
689 $      \else
690 $      \def\next{\endgroup\expandafter\ltx@secondoftwo}%
691 $      \fi
692 $      \else
693 $      \def\next{\endgroup\expandafter\ltx@secondoftwo}%
694 $      \fi
695 $      \next
696 $      \fi
697 }

```

`\ltx@ifboxvoidoreempty`

```

698 \def\ltx@ifboxvoidoreempty#1{%
699   \ifvoid#1\relax
700     \expandafter\ltx@thirdoffour
701   \fi
702   \ltx@ifboxempty{#1}%
703 }

704 \LTXcmds@AtEnd%
705 \endpackage

```

## 3 Installation

### 3.1 Download

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

[CTAN:macros/latex/contrib/ltxcmds/ltxcmds.dtx](https://ctan.org/ctan/packages/macros/latex/contrib/ltxcmds/ltxcmds.dtx) The source file.

[CTAN:macros/latex/contrib/ltxcmds/ltxcmds.pdf](https://ctan.org/ctan/packages/macros/latex/contrib/ltxcmds/ltxcmds.pdf) Documentation.

**Bundle.** All the packages of the bundle ‘ltxcmds’ 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/ltxcmds.tds.zip](https://ctan.org/ctan/packages/install/macros/latex/contrib/ltxcmds.tds.zip)

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

---

<sup>1</sup>[CTAN:pkg/ltxcmds](https://ctan.org/ctan/packages/pkg/ltxcmds)

## 3.2 Bundle installation

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

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

## 3.3 Package installation

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

```
tex ltxcmds.dtx
```

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

```
ltxcmds.sty → tex/generic/ltxcmds/ltxcmds.sty
ltxcmds.pdf → doc/latex/ltxcmds/ltxcmds.pdf
ltxcmds.dtx → source/latex/ltxcmds/ltxcmds.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`.

## 3.4 Refresh file name databases

If your  $\TeX$  distribution ( $\TeX$  Live, MiK $\TeX$ , ...) relies on file name databases, you must refresh these. For example,  $\TeX$  Live users run `texhash` or `mktexlsr`.

## 3.5 Some details for the interested

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

**plain  $\TeX$ :** Run `docstrip` and extract the files.

**$\LaTeX$ :** Generate the documentation.

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

```
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 $\LaTeX$` :

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

## 4 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.  
<https://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$1@news.albasani.net`, 2010-11-12.  
<https://groups.google.com/group/comp.text.tex/msg/803bd57221a04996>
- [3] Ahmed Musa: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:f5496afe-40ed-42bd-b629-a2419ecf7c0d@o14g2000prn.googlegroups.com`, 2010-12-03.  
<https://groups.google.com/group/comp.text.tex/msg/fbf7d61a0c3a807d>
- [4] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idbo94$uka$1@four.albasani.net`, 2010-12-03.  
<https://groups.google.com/group/comp.text.tex/msg/0c230ee479487962>
- [5] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idbpu4$cg1$1@news.albasani.net`, 2010-12-03.  
<https://groups.google.com/group/comp.text.tex/msg/bbef4263390d647b>
- [6] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idd4ga$r83$1@four.albasani.net`, 2010-12-04.  
<https://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.  
<https://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.  
<https://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.  
<https://groups.google.com/group/comp.text.tex/msg/8d3cb89496a4d86d>

## 5 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@ifboxvoidoreempty` 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.

[2011/04/14 v1.19]

- `\ltx@ifnextchar`: detection of optional spaces modified.
- `\ltx(Loc, Glob)(Toks, Dimen, Skip)(A, B, C, D, E)` added.

[2011/04/18 v1.20]

- `\ltx@ifnextchar` with conditional support (thanks GL for bug report).

[2011/08/22 v1.21]

- `\ltx@GlobalPrependToMacro`, `\ltx@LocalPrependToMacro` added (feature request of Martin Münch).

[2011/11/09 v1.22]

- `\ltx@carsecond`, `\ltx@carthird`, `\ltx@carfourth`, `\ltx@CarNumth` added.
- `\ltx@cdrzero`, `\ltx@cdr`, `\ltx@cdrtwo`, `csltx@cdrthree`, `\ltx@cdrfour`, `\ltx@CdrNum` modified to retain braces and spaces. They are expandable in two expansion steps.

[2016/05/16 v1.23]

- Documentation updates.

[2019/12/15 v1.24]

- Documentation updates.

## 6 Index

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

Symbols		D	
<code>\#</code> .....	254	<code>\detokenize</code> .....	623, 634
<code>\\$</code> .....	652, 655	<code>\dimendef</code> .....	134, 135, 136, 137, 138, 139, 140, 141, 142, 143
<code>\%</code> .....	244	E	
<code>\&amp;</code> .....	653, 656	<code>\empty</code> .....	17, 18
<code>\.</code> .....	517	<code>\endcsname</code> .....	14, 21, 50, 66, 76, 162, 168, 208, 214, 217, 278, 280, 283, 285, 288, 301, 303, 306, 308, 311, 320, 322, 330, 342, 343, 381, 506, 549, 626, 637
<code>\@nil</code> 180, 181, 185, 189, 190, 194, 195, 199, 200, 219, 229, 272, 277, 294, 300, 415, 417, 642, 644, 648		<code>\endinput</code> .....	29, 115
<code>\@undefined</code> .....	58	<code>\endlinechar</code> .....	4, 35, 71, 77, 89
<code>\%</code> .....	249	<code>\escapechar</code> .....	270, 275, 292, 297
<code>\{</code> .....	259	F	
<code>\}</code> .....	264	<code>\futurelet</code> .....	499, 514, 525
A		H	
<code>\aftergroup</code> .....	29	<code>\hbox</code> .....	541, 543, 552, 554, 570, 663, 674, 682
B		I	
<code>\box</code> .....	543, 554	<code>\if</code> .....	277, 300
C		<code>\ifcase</code> .....	390
<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, 652, 653, 655, 656	<code>\ifcat</code> .....	623, 634
<code>\chardef</code> .....	116, 117, 118, 119, 120	<code>\ifcsname</code> .....	342
<code>\csname</code> .....	14, 21, 50, 66, 76, 160, 165, 206, 211, 278, 280, 283, 285, 288, 301, 303, 306, 308, 311, 320, 322, 330, 343, 381, 506, 549, 621, 632	<code>\iffalse</code> ..	286, 309, 589, 595, 602, 611
		<code>\ifhbox</code> .....	663, 674, 682
		<code>\ifhmode</code> .....	664, 676, 684
		<code>\ifnum</code> ..	391, 393, 395, 422, 668, 679, 687
		<code>\iftrue</code> .....	281, 304
		<code>\ifvoid</code> .....	550, 659, 699

<code>\ifx</code>	15, 18, 21, 50, 58, 61, 320, 322, 330, 343, 437, 440, 451, 454, 465, 468, 480, 483, 502, 506, 529	<code>\ltx@GlobToksD</code>	132
<code>\immediate</code>	23, 52	<code>\ltx@GlobToksE</code>	133
<b>L</b>			
<code>\lastnodetype</code>	668	<code>\ltx@gobble</code>	3, 154, 336, 590, 592, 603, 606, 608, 634
<code>\lastpenalty</code>	679, 687	<code>\ltx@gobblefour</code>	157
<code>\lccode</code>	244, 249, 254, 259, 264	<code>\ltx@GobbleNum</code>	3, 158, 227, 237
<code>\leavevmode</code>	563	<code>\ltx@gobblethree</code>	156
<code>\letLTXcmds@gtmp</code>	455, 484	<code>\ltx@gobbletwo</code>	155
<code>\lowercase</code>	245, 250, 255, 260, 265	<code>\ltx@hashchar</code>	253
<code>\ltx@(Loc, Glob) (Toks, Dimen, Skip) (A, B, C, D, E)</code>	3	<code>\ltx@ifblank</code>	9, 600, 630
<code>\ltx@active</code>	119	<code>\ltx@ifBoxEmpty</code>	10, 658, 702
<code>\ltx@backslashchar</code>	248	<code>\ltx@ifBoxVoidOrEmpty</code>	10, 698
<code>\ltx@car</code>	5, 185	<code>\ltx@ifclasslater</code>	8, 407
<code>\ltx@carfour</code>	5, 199	<code>\ltx@ifclassloaded</code>	7, 369
<code>\ltx@carfourth</code>	200	<code>\ltx@ifempty</code>	9, 586, 619, 645
<code>\ltx@CarNum</code>	5, 204	<code>\ltx@iffilelater</code>	375, 408, 411
<code>\ltx@CarNumth</code>	223	<code>\ltx@iffileloaded</code>	7, 366, 370, 373, 376
<code>\ltx@carsecond</code>	190	<code>\ltx@ifnextchar</code>	8, 494
<code>\ltx@carthird</code>	195	<code>\ltx@ifnextchar@nospace</code>	9, 520
<code>\ltx@carthree</code>	5, 194	<code>\ltx@ifpackagelater</code>	410
<code>\ltx@cartwo</code>	5, 189	<code>\ltx@ifpackageloaded</code>	372
<code>\ltx@carzero</code>	5, 180	<code>\ltx@ifUndefined</code>	6, 328, 352, 413, 535, 536, 537, 538, 572, 585, 651
<code>\ltx@cclv</code>	120	<code>\ltx@ifundefined</code>	6, 321, 341, 352, 367
<code>\ltx@cdr</code>	186	<code>\ltx@leavevmode</code>	9, 535, 569
<code>\ltx@cdrfour</code>	201	<code>\ltx@leftbracechar</code>	258
<code>\ltx@CdrNum</code>	233	<code>\ltx@LocalAppendToMacro</code>	449
<code>\ltx@cdrthree</code>	196	<code>\ltx@LocalExpandAfter</code>	6, 313, 319
<code>\ltx@cdrtwo</code>	191	<code>\ltx@LocalPrependToMacro</code>	478
<code>\ltx@cdrzero</code>	182, 235	<code>\ltx@LocDimenA</code>	134
<code>\ltx@clsextension</code>	7, 364, 370, 408	<code>\ltx@LocDimenB</code>	135
<code>\ltx@empty</code>	6, 241, 438, 441, 452, 455, 466, 469, 481, 484	<code>\ltx@LocDimenC</code>	136
<code>\ltx@firstoffour</code>	176	<code>\ltx@LocDimenD</code>	137
<code>\ltx@firstofone</code>	4, 170, 338, 519, 574, 580	<code>\ltx@LocDimenE</code>	138
<code>\ltx@firstofthree</code>	173, 594, 610	<code>\ltx@LocSkipA</code>	144
<code>\ltx@firstoftwo</code>	171, 323, 331, 344, 349, 367, 404, 598, 614, 669, 688	<code>\ltx@LocSkipB</code>	145
<code>\ltx@fourthoffour</code>	179	<code>\ltx@LocSkipC</code>	146
<code>\ltx@GlobalAppendToMacro</code>	8, 436	<code>\ltx@LocSkipD</code>	147
<code>\ltx@GlobalPrependToMacro</code>	8, 464	<code>\ltx@LocSkipE</code>	148
<code>\ltx@GlobDimenA</code>	139	<code>\ltx@LocToksA</code>	124, 445, 446, 459, 460, 473, 475, 488, 490, 497, 503, 523, 530
<code>\ltx@GlobDimenB</code>	140	<code>\ltx@LocToksB</code>	125, 474, 475, 489, 490, 498, 509, 524, 532
<code>\ltx@GlobDimenC</code>	141	<code>\ltx@LocToksC</code>	126
<code>\ltx@GlobDimenD</code>	142	<code>\ltx@LocToksD</code>	127
<code>\ltx@GlobDimenE</code>	143	<code>\ltx@LocToksE</code>	128
<code>\ltx@GlobSkipA</code>	149	<code>\ltx@mbox</code>	9, 568
<code>\ltx@GlobSkipB</code>	150	<code>\ltx@minusone</code>	121
<code>\ltx@GlobSkipC</code>	151	<code>\ltx@newglobalif</code>	6, 290
<code>\ltx@GlobSkipD</code>	152	<code>\ltx@newif</code>	6, 268
<code>\ltx@GlobSkipE</code>	153	<code>\ltx@one</code>	117, 122, 675, 679
<code>\ltx@GlobToksA</code>	129	<code>\ltx@onelevel@sanitize</code>	7, 358
<code>\ltx@GlobToksB</code>	130	<code>\ltx@percentchar</code>	243
<code>\ltx@GlobToksC</code>	131	<code>\ltx@pkgextension</code>	365, 373, 411
		<code>\ltx@RemovePrefix</code>	7, 354, 356, 361
		<code>\ltx@ReturnAfterElseFi</code>	240
		<code>\ltx@ReturnAfterFi</code>	5, 239

<code>\ltx@rightbracechar</code> .....	<a href="#">263</a>	<b>N</b>	
<code>\ltx@secondoffour</code> .....	<a href="#">177</a>	<code>\next</code> .....	<a href="#">688, 690, 693, 695</a>
<code>\ltx@secondofthree</code> .....	<a href="#">174</a>	<code>\number</code> .....	<a href="#">378, 384, 575</a>
<code>\ltx@secondoftwo</code> .....	<a href="#">172, 325, 333, 346, 367, 387,</a>	<code>\numexpr</code> .....	<a href="#">581</a>
	<a href="#">402, 596, 612, 660, 671, 690, 693</a>	<b>P</b>	
<code>\ltx@space</code> .....	<a href="#">6, 242, 401</a>	<code>\PackageInfo</code> .....	<a href="#">26</a>
<code>\ltx@StripPrefix</code> ..	<a href="#">355, 428, 429, 430</a>	<code>\pdflastmatch</code> .....	<a href="#">428, 429, 430</a>
<code>\ltx@thirdoffour</code> .....	<a href="#">178, 700</a>	<code>\pdfmatch</code> .....	<a href="#">422</a>
<code>\ltx@thirdofthree</code> .....	<a href="#">175</a>	<code>\penalty</code> .....	<a href="#">675, 683</a>
<code>\ltx@two</code> .....	<a href="#">118, 683, 687</a>	<code>\ProvidesPackage</code> .....	<a href="#">19, 67</a>
<code>\ltx@undefined</code> ....	<a href="#">437, 451, 465, 480</a>	<b>Q</b>	
<code>\ltx@zapspace</code> .....	<a href="#">9, 640</a>	<code>\quitvmode</code> .....	<a href="#">566</a>
<code>\ltx@zero</code> ...	<a href="#">3, 116, 183, 187, 192,</a>	<b>R</b>	
	<a href="#">197, 202, 220, 230, 236, 541,</a>	<code>\romannumeral</code> ..	<a href="#">159, 162, 183, 187,</a>
	<a href="#">543, 545, 642, 663, 668, 674, 682</a>		<a href="#">192, 197, 202, 205, 208, 224,</a>
<code>\LTXcmds@ifnextchar</code> .....	<a href="#">507, 513</a>		<a href="#">234, 517, 588, 601, 620, 631, 641</a>
<code>\LTXcmds@ParseVersion</code> ....	<a href="#">415, 417</a>	<b>S</b>	
<code>\LTXcmds@AtEnd</code> .....	<a href="#">95, 96, 115, 704</a>	<code>\setbox</code> .....	<a href="#">541, 543, 552, 554, 663, 674, 682</a>
<code>\LTXcmds@CarNum</code> .....	<a href="#">207, 210</a>	<code>\skipdef</code> .....	<a href="#">144, 145, 146,</a>
<code>\LTXcmds@CarNumFinish</code> .....	<a href="#">219</a>		<a href="#">147, 148, 149, 150, 151, 152, 153</a>
<code>\LTXcmds@CarNumth</code> .....	<a href="#">226, 229</a>	<b>T</b>	
<code>\LTXcmds@cdrzero</code> .....	<a href="#">181, 183, 187, 192, 197, 202</a>	<code>\the</code> .....	<a href="#">77, 78,</a>
<code>\LTXcmds@CharToken</code> .	<a href="#">496, 502, 522, 529</a>		<a href="#">79, 80, 81, 82, 83, 84, 97, 446,</a>
<code>\LTXcmds@Cm</code> .....	<a href="#">213</a>		<a href="#">460, 475, 490, 503, 509, 528, 581</a>
<code>\LTXcmds@Cx</code> .....	<a href="#">216</a>	<code>\TMP@EnsureCode</code> .....	<a href="#">94, 101,</a>
<code>\LTXcmds@Gm</code> .....	<a href="#">167</a>		<a href="#">102, 103, 104, 105, 106, 107,</a>
<code>\LTXcmds@GobbleNum</code> .....	<a href="#">161, 164</a>		<a href="#">108, 109, 110, 111, 112, 113, 114</a>
<code>\LTXcmds@gttemp</code> .....	<a href="#">450,</a>	<code>\toksdef</code> .....	<a href="#">124, 125, 126,</a>
	<a href="#">451, 452, 454, 459, 460, 462,</a>		<a href="#">127, 128, 129, 130, 131, 132, 133</a>
	<a href="#">479, 480, 481, 483, 489, 490, 492</a>	<b>U</b>	
<code>\LTXcmds@ifempty</code> .....	<a href="#">629</a>	<code>\unhbox</code> .....	<a href="#">545, 557, 560</a>
<code>\LTXcmds@ifempty@</code> .....	<a href="#">628</a>	<code>\unhcopy</code> .....	<a href="#">664, 676, 684</a>
<code>\LTXcmds@ifLater</code> .....	<a href="#">377, 389</a>	<code>\unvcopy</code> .....	<a href="#">664, 676, 684</a>
<code>\LTXcmds@ifnextchar</code> ...	<a href="#">499, 501, 516</a>	<b>V</b>	
<code>\LTXcmds@ifnextchar@nospace</code>	<a href="#">525, 527</a>	<code>\vbox</code> .....	<a href="#">663, 674, 682</a>
<code>\LTXcmds@LetToken</code> .....	<a href="#">499, 502, 515, 525, 529</a>	<code>\voidb@x</code> .....	<a href="#">560</a>
<code>\LTXcmds@newglobalif</code> .....	<a href="#">294, 296</a>	<b>W</b>	
<code>\LTXcmds@newif</code> .....	<a href="#">272, 274</a>	<code>\write</code> .....	<a href="#">23, 52</a>
<code>\LTXcmds@num</code> .....	<a href="#">162, 208, 572</a>	<b>X</b>	
<code>\LTXcmds@ParseVersion</code> .....	<a href="#">379, 385, 414, 421</a>	<code>\x</code> .....	<a href="#">14, 15, 18, 22, 26, 28, 51, 56, 66, 75, 87</a>
<code>\LTXcmds@SpaceToken</code> .....	<a href="#">506, 519</a>		
<code>\LTXcmds@temp</code> .....	<a href="#">586, 617</a>		
<code>\LTXcmds@VoidBox</code> .....	<a href="#">549, 550, 552, 554, 557</a>		
<code>\LTXcmds@zapspace</code> .....	<a href="#">642, 644</a>		
<b>M</b>			
<code>\meaning</code> .....	<a href="#">361</a>		