The ltxcmds package

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

2016/05/16 v1.23

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

1 Documentation **3**
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 . . . 5
1.9 Characters . . . 6
1.10 Boolean switch . . . 6
1.11 Command definitions . . . 6
1.12 Stripping . . . 6
1.13 File management . . . 7
1.13.1 File extensions . . . 7
1.13.2 Load check . . . 7
1.13.3 Version date check . . . 7
1.14 Macro additions . . . 8
1.15 Next character detection . . . 8
1.16 \ltx@leavevmode, \ltx@mbox . . . 8
1.17 Expandable test for emptiness . . . 9
1.18 Stripping spaces . . . 9
1.19 Check for emptiness of boxes . . . 9

2 Implementation **10**
2.1 Identification . . . 10
2.2 Numbers . . . 11
2.3 Scratch registers . . . 12
2.4 Argument killers . . . 14
2.5 Argument grabbers . . . 14
2.6 List helpers . . . 15
2.7 Tail recursion . . . 16
2.8 Empty macro . . . 16
2.9 Characters . . . 17
2.10 Boolean switch . . . 17

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

2.11 Command definitions . . . 18
2.12 Stripping . . . 19
2.13 File management . . . 19
2.13.1 File extensions . . . 19
2.13.2 Load check . . . 20
2.13.3 Version date check . . . 20
2.14 Macro additions . . . 21
2.15 Next character detection . . . 22
2.16 \ltx@leavevmode, \ltx@mbox . . . 23
2.17 Help macros . . . 24
2.18 Expandable test for emptiness . . . 24
2.18.1 Vanilla TEX . . . 24
2.18.2 With \detokenize . . . 25
2.18.3 \ltx@ifblank . . . 25
2.19 \ltx@zapspace . . . 26
2.20 \ltx@IfBoxEmpty . . . 26

3 Test **27**
3.1 Catcode checks for loading . . . 27
3.2 Test \ltx@GobbleNum . . . 29
3.3 Test \ltx@ifempty . . . 32
3.4 Test \ltx@zap@space . . . 33
3.5 Test \ltx@IfBoxEmpty . . . 34
3.6 Test for next character detection . . . 36
3.7 Test for list helpers . . . 39

4 Installation **40**
4.1 Download . . . 40
4.2 Bundle installation . . . 40
4.3 Package installation . . . 41
4.4 Refresh file name databases . . . 41
4.5 Some details for the interested . . . 41

5 Catalogue **42**

6 References **42**

7 History **43**
[2009/08/05 v1.0] . . . 43
[2009/12/12 v1.1] . . . 43
[2010/01/28 v1.2] . . . 43
[2010/03/01 v1.3] . . . 43
[2010/03/09 v1.4] . . . 43
[2010/04/08 v1.5] . . . 43
[2010/04/16 v1.6] . . . 43
[2010/04/26 v1.7] . . . 44
[2010/09/11 v1.8] . . . 44
[2010/10/25 v1.9] . . . 44
[2010/10/31 v1.10] . . . 44
[2010/11/12 v1.11] . . . 44
[2010/12/02 v1.12] . . . 44
[2010/12/04 v1.13] . . . 44
[2010/12/07 v1.14] . . . 44
[2010/12/12 v1.15] . . . 44
[2011/02/04 v1.16] . . . 44
[2011/02/05 v1.17] . . . 44
[2011/03/16 v1.18] . . . 44
[2011/04/14 v1.19] . . . 45

[2011/04/18 v1.20] 45
[2011/08/22 v1.21] 45
[2011/11/09 v1.22] 45
[2016/05/16 v1.23] 45

8 Index **45**

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@cclv`	→	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 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.

`\ltx@(Loc,Glob)(Toks,Dimen,Skip)(A,B,C,D,E)`

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

`\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 fol-

lowing arguments are eaten. Macro \ltx@GobbleNum is expandable in exact two expansion steps.

1.5 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.

1.6 List helpers

\ltx@carzero ... \@nil	→	
\ltx@cdrzero ... \@nil	→	...

\ltx@car {⟨*1*⟩} ... \@nil	→	⟨*1*⟩
\ltx@cdr {⟨*1*⟩} ... \@nil	→	...

\ltx@cartwo {⟨*1*⟩} {⟨*2*⟩} ... \@nil	→	⟨*1*⟩⟨*2*⟩
\ltx@carsecond {⟨*1*⟩} {⟨*2*⟩} ... \@nil	→	⟨*2*⟩
\ltx@cdrtwo {⟨*1*⟩} {⟨*2*⟩} ... \@nil	→	...

\ltx@carthree {⟨*1*⟩} {⟨*2*⟩} {⟨*3*⟩} ... \@nil	→	⟨*1*⟩⟨*2*⟩⟨*3*⟩
\ltx@carthird {⟨*1*⟩} {⟨*2*⟩} {⟨*3*⟩} ... \@nil	→	⟨*3*⟩
\ltx@cdrthree {⟨*1*⟩} {⟨*2*⟩} {⟨*3*⟩} ... \@nil	→	...

\ltx@carfour {⟨*1*⟩} {⟨*2*⟩} {⟨*3*⟩} {⟨*4*⟩} ... \@nil	→	⟨*1*⟩⟨*2*⟩⟨*3*⟩⟨*4*⟩
\ltx@carfourth {⟨*1*⟩} {⟨*2*⟩} {⟨*3*⟩} {⟨*4*⟩} ... \@nil	→	⟨*4*⟩
\ltx@cdrfour {⟨*1*⟩} {⟨*2*⟩} {⟨*3*⟩} {⟨*4*⟩} ... \@nil	→	...

\ltx@CarNum {⟨*num*⟩} {⟨*1*⟩} ... {⟨⟨*num*⟩⟩} {⟨⟨*num*⟩+*1*⟩} ... \@nil	→	{⟨*1*⟩} ... {⟨⟨*num*⟩⟩} ...
\ltx@CarNumth {⟨*num*⟩} {⟨*1*⟩} ... {⟨⟨*num*⟩⟩} {⟨⟨*num*⟩+*1*⟩} ... \@nil	→	{⟨⟨*num*⟩⟩} ...
\ltx@CdrNum {⟨*num*⟩} {⟨*1*⟩} ... {⟨⟨*num*⟩⟩} {⟨⟨*num*⟩+*1*⟩} ... \@nil	→	{⟨⟨*num*⟩+*1*⟩} ...

Macros with uppercase letters are expandable in two expansion steps. Changes in version 2016/05/16 v1.23:

- 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

\ltx@ReturnAfterFi {⟨*1*⟩} \fi	→	\fi ⟨*1*⟩
\ltx@ReturnAfterElseFi {⟨*1*⟩} \else {⟨*2*⟩} \fi	→	\fi ⟨*1*⟩

1.8 Empty macro

\ltx@empty	→	

1.9 Characters

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

1.10 Boolean switch

\ltx@newif {⟨*cmd*⟩}

\ltx@newif defines a new boolean switch ⟨*cmd*⟩ like \newif. Unlike plain 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.

1.11 Command definitions

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

If ε-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 ε-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 ε-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.

1.12 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 \pdflastmatch. Macro \ltx@RemovePrefix has the same meaning as 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 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 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 LaTeX kernel. Also they can be used after the preamble.

1.13.1 File extensions

\ltx@clsextension
\ltx@pkgextension

Macros \ltx@clsextension and \ltx@styextension stores the strings cls and sty. In opposite to LaTeX'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 LaTeX'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 LaTeX 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 TeX'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_ε 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 {⟨*char*⟩} {⟨*yes*⟩} {⟨*no*⟩}

Macro \ltx@ifnextchar@nospace behaves like macro \ltx@ifnextchar with the exception that optional spaces are not supported before ⟨*char*⟩. 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 {⟨*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.

1.18 Stripping spaces

\ltx@zapspace {⟨*stuff*⟩}

Macro \ltx@zapspace strips spaces from ⟨*stuff*⟩ that are not hidden inside curly braces. Like LaTeX'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.

1.19 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@IfBoxVoidOrEmpty {⟨*box register number*⟩} {⟨*yes*⟩} {⟨*no*⟩}

Macro \ltx@IfBoxVoidOrEmpty calls ⟨*yes*⟩ if the box is either void or does not contain any content. Otherwise ⟨*no*⟩ is executed. Macro \ltx@IfBoxVoidOrEmpty 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
```

```
    #2[{#3}]%
    \ifx#1\@undefined
      \xdef#1{#3}%
    \fi
    \ifx#1\relax
      \xdef#1{#3}%
    \fi
  }%
 \fi
\expandafter\x\csname ver@ltxcmds.sty\endcsname
\ProvidesPackage{ltxcmds}%
 [2016/05/16 v1.23 LaTeX kernel commands for general use (HO)]%
\begingroup\catcode61\catcode48\catcode32=10\relax%
 \catcode13=5 % ^^M
 \endlinechar=13 %
 \catcode123=1 % {
 \catcode125=2 % }
 \catcode64=11 % @
 \def\x{\endgroup
   \expandafter\edef\csname LTXcmds@AtEnd\endcsname{%
     \endlinechar=\the\endlinechar\relax
     \catcode13=\the\catcode13\relax
     \catcode32=\the\catcode32\relax
     \catcode35=\the\catcode35\relax
     \catcode61=\the\catcode61\relax
     \catcode64=\the\catcode64\relax
     \catcode123=\the\catcode123\relax
     \catcode125=\the\catcode125\relax
   }%
 }%
\x\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode64=11 % @
\catcode123=1 % {
\catcode125=2 % }
\def\TMP@EnsureCode#1#2{%
 \edef\LTXcmds@AtEnd{%
  \LTXcmds@AtEnd
  \catcode#1=\the\catcode#1\relax
 }%
 \catcode#1=#2\relax
}
\TMP@EnsureCode{36}{3}% $
\TMP@EnsureCode{38}{4}% &
\TMP@EnsureCode{40}{12}% (
\TMP@EnsureCode{41}{12}% )
\TMP@EnsureCode{45}{12}% -
\TMP@EnsureCode{46}{12}% .
\TMP@EnsureCode{47}{12}% /
\TMP@EnsureCode{60}{12}% <
\TMP@EnsureCode{62}{12}% >
\TMP@EnsureCode{91}{12}% [
\TMP@EnsureCode{96}{12}% `
\TMP@EnsureCode{93}{12}% ]
\TMP@EnsureCode{94}{12}% ^ (superscript) (!)
\TMP@EnsureCode{124}{12}% |
\edef\LTXcmds@AtEnd{\LTXcmds@AtEnd\noexpand\endinput}
```

2.2 Numbers

\ltx@zero

```
\chardef\ltx@zero=0 %
```
\ltx@zero
```
\chardef\ltx@one=1 %
```
\ltx@two
```
\chardef\ltx@two=2 %
```
\ltx@active
```
\chardef\ltx@active=13 %
```
\ltx@cclv
```
\chardef\ltx@cclv=255 %
```
\ltx@minusone
```
\def\ltx@minusone{%
  -\ltx@one
}
```

2.3 Scratch registers

\ltx@LocToksA
```
\toksdef\ltx@LocToksA=0 %
```
\ltx@LocToksB
```
\toksdef\ltx@LocToksB=2 %
```
\ltx@LocToksC
```
\toksdef\ltx@LocToksC=4 %
```
\ltx@LocToksD
```
\toksdef\ltx@LocToksD=6 %
```
\ltx@LocToksE
```
\toksdef\ltx@LocToksE=8 %
```
\ltx@GlobToksA
```
\toksdef\ltx@GlobToksA=1 %
```
\ltx@GlobToksB
```
\toksdef\ltx@GlobToksB=3 %
```
\ltx@GlobToksC
```
\toksdef\ltx@GlobToksC=5 %
```
\ltx@GlobToksD
```
\toksdef\ltx@GlobToksD=7 %
```
\ltx@GlobToksE
```
\toksdef\ltx@GlobToksE=9 %
```
\ltx@LocDimenA
```
\dimendef\ltx@LocDimenA=0 %
```
\ltx@LocDimenB
```
\dimendef\ltx@LocDimenB=2 %
```
\ltx@LocDimenC
```
\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
```
\long\def\ltx@gobble#1{}
```
\ltx@gobbletwo
```
\long\def\ltx@gobbletwo#1#2{}
```
\ltx@gobblethree
```
\long\def\ltx@gobblethree#1#2#3{}
```
\ltx@gobblefour
```
\long\def\ltx@gobblefour#1#2#3#4{}
```
\ltx@GobbleNum
```
\def\ltx@GobbleNum#1{%
  \romannumeral
  \csname ltx@zero%
  \expandafter\LTXcmds@GobbleNum
  \romannumeral\LTXcmds@num{#1}000{m\endcsname}%
}
```
\LTXcmds@GobbleNum
```
\def\LTXcmds@GobbleNum#1{%
  \csname LTXcmds@G#1\LTXcmds@GobbleNum
}
```
\LTXcmds@Gm
```
\long\def\LTXcmds@Gm#1{%
  \endcsname
}
```

2.5 Argument grabbers

\ltx@firstofone
```
\long\def\ltx@firstofone#1{#1}
```
\ltx@firstoftwo
```
\long\def\ltx@firstoftwo#1#2{#1}
```
\ltx@secondoftwo
```
\long\def\ltx@secondoftwo#1#2{#2}
```
\ltx@firstofthree
```
\long\def\ltx@firstofthree#1#2#3{#1}
```
\ltx@secondofthree
```
\long\def\ltx@secondofthree#1#2#3{#2}
```
\ltx@thirdofthree
```
\long\def\ltx@thirdofthree#1#2#3{#3}%
```
\ltx@firstoffour
```
\long\def\ltx@firstoffour#1#2#3#4{#1}
```
\ltx@secondoffour
```
\long\def\ltx@secondoffour#1#2#3#4{#2}
```
\ltx@thirdoffour
```
\long\def\ltx@thirdoffour#1#2#3#4{#3}%
```
\ltx@fourthoffour
```
\long\def\ltx@fourthoffour#1#2#3#4{#4}%
```

2.6 List helpers

\ltx@carzero

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

\LTXcmds@cdrzero

```
\long\def\LTXcmds@cdrzero#1\@nil{#1}
```

\ltx@cdrzero

```
\def\ltx@cdrzero{%
  \romannumeral\LTXcmds@cdrzero\ltx@zero
}
```

\ltx@car

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

\ltx@cdr

```
\long\def\ltx@cdr#1{%
  \romannumeral\LTXcmds@cdrzero\ltx@zero
}
```

\ltx@cartwo

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

\ltx@carsecond

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

\ltx@cdrtwo

```
\long\def\ltx@cdrtwo#1#2{%
  \romannumeral\LTXcmds@cdrzero\ltx@zero
}
```

\ltx@carthree

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

\ltx@carthird

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

\ltx@cdrthree

```
\long\def\ltx@cdrthree#1#2#3{%
  \romannumeral\LTXcmds@cdrzero\ltx@zero
}
```

\ltx@carfour

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

\ltx@carfourth

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

\ltx@cdrfour

```
\long\def\ltx@cdrfour#1#2#3#4{%
  \romannumeral\LTXcmds@cdrzero\ltx@zero
}
```

\ltx@CarNum

```
\def\ltx@CarNum#1{%
  \romannumeral
  \csname LTXcmds@CarNumFinish%
  \expandafter\LTXcmds@CarNum
  \romannumeral\LTXcmds@num{#1}000{x\endcsname}%
}
```

\LTXcmds@CarNum

```
\def\LTXcmds@CarNum#1{%
  \csname LTXcmds@C#1\LTXcmds@CarNum
}
```

\LTXcmds@Cm

```
\long\def\LTXcmds@Cm#1#2{%
  \endcsname{#1#2}%
}
```

\LTXcmds@Cx

```
\def\LTXcmds@Cx#1{%
  \endcsname{}%
}
```

\LTXcmds@CarNumFinish

```
\long\def\LTXcmds@CarNumFinish#1#2\@nil{%
  \ltx@zero
  #1%
}
```

\ltx@CarNumth

```
\def\ltx@CarNumth#1{%
  \romannumeral
  \expandafter\expandafter\expandafter
  \LTXcmds@CarNumth
  \ltx@GobbleNum{#1}{}%
}
```

\LTXcmds@CarNumth

```
\long\def\LTXcmds@CarNumth#1#2\@nil{%
  \ltx@zero
  #1%
}
```

\ltx@CdrNum

```
\def\ltx@CdrNum#1{%
  \romannumeral%
  \expandafter\expandafter\expandafter\ltx@cdrzero
  \expandafter\expandafter\expandafter\ltx@zero
  \ltx@GobbleNum{#1}%
}
```

2.7 Tail recursion

\ltx@ReturnAfterFi

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

\ltx@ReturnAfterElseFi

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

2.8 Empty macro

\ltx@empty

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

2.9 Characters

\ltx@space

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

\ltx@percentchar

```
\begingroup
  \lccode`0=`\%\relax
\lowercase{\endgroup
  \def\ltx@percentchar{0}%
}
```

\ltx@backslashchar

```
\begingroup
  \lccode`0=`\\\relax
\lowercase{\endgroup
  \def\ltx@backslashchar{0}%
}
```

\ltx@hashchar

```
\begingroup
  \lccode`0=`\#\relax
\lowercase{\endgroup
  \def\ltx@hashchar{0}%
}
```

\ltx@leftbracechar

```
\begingroup
  \lccode`0=`\{\relax
\lowercase{\endgroup
  \def\ltx@leftbracechar{0}%
}
```

\ltx@rightbracechar

```
\begingroup
  \lccode`0=`\}\relax
\lowercase{\endgroup
  \def\ltx@rightbracechar{0}%
}
```

2.10 Boolean switch

\ltx@newif

```
\def\ltx@newif#1{%
  \begingroup
    \escapechar=-1 %
  \expandafter\endgroup
  \expandafter\LTXcmds@newif\string#1\@nil
}
```

\LTXcmds@newif

```
\begingroup
  \escapechar=-1 %
\expandafter\endgroup
\expandafter\def\expandafter\LTXcmds@newif\string\if#1\@nil{%
  \expandafter\edef\csname#1true\endcsname{%
    \let
    \expandafter\noexpand\csname if#1\endcsname
    \noexpand\iftrue
  }%
  \expandafter\edef\csname#1false\endcsname{%
```

```
    \let
    \expandafter\noexpand\csname if#1\endcsname
    \noexpand\iffalse
  }%
  \csname#1false\endcsname
}
```

\ltx@newglobalif

```
\def\ltx@newglobalif#1{%
  \begingroup
    \escapechar=-1 %
  \expandafter\endgroup
  \expandafter\LTXcmds@newglobalif\string#1\@nil
}
```

\LTXcmds@newglobalif

```
\begingroup
  \escapechar=-1 %
\expandafter\endgroup
\expandafter
\def\expandafter\LTXcmds@newglobalif\string\if#1\@nil{%
  \expandafter\edef\csname#1true\endcsname{%
    \global\let
    \expandafter\noexpand\csname if#1\endcsname
    \noexpand\iftrue
  }%
  \expandafter\edef\csname#1false\endcsname{%
    \global\let
    \expandafter\noexpand\csname if#1\endcsname
    \noexpand\iffalse
  }%
  \csname#1false\endcsname
}
```

2.11 Command definitions

\ltx@LocalExpandAfter

```
\def\ltx@LocalExpandAfter{%
  \begingroup
    \expandafter\expandafter\expandafter
  \endgroup
  \expandafter
}

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

\ltx@ifundefined

```
  \def\ltx@ifundefined#1{%
    \expandafter\ifx\csname #1\endcsname\relax
      \expandafter\ltx@firstoftwo
    \else
      \expandafter\ltx@secondoftwo
    \fi
  }%
```

\ltx@IfUndefined

```
  \def\ltx@IfUndefined#1{%
    \begingroup\expandafter\expandafter\expandafter\endgroup
    \expandafter\ifx\csname #1\endcsname\relax
      \expandafter\ltx@firstoftwo
    \else
```

```
      \expandafter\ltx@secondoftwo
    \fi
  }%
  \expandafter\ltx@gobble
\else
  \expandafter\ltx@firstofone
\fi
{%
```

\ltx@ifundefined

```
  \def\ltx@ifundefined#1{%
    \ifcsname #1\endcsname
      \expandafter\ifx\csname #1\endcsname\relax
        \expandafter\expandafter\expandafter\ltx@firstoftwo
      \else
        \expandafter\expandafter\expandafter\ltx@secondoftwo
      \fi
    \else
      \expandafter\ltx@firstoftwo
    \fi
  }%
```

\ltx@IfUndefined

```
  \let\ltx@IfUndefined\ltx@ifundefined
}
```

2.12 Stripping

\ltx@RemovePrefix

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

\ltx@StripPrefix

```
\def\ltx@StripPrefix{%
  \expandafter\ltx@RemovePrefix
}
```

\ltx@onelevel@sanitize

```
\def\ltx@onelevel@sanitize#1{%
  \edef#1{%
    \expandafter
    \ltx@RemovePrefix\meaning#1%
  }%
}
```

2.13 File management

2.13.1 File extensions

\ltx@clsextension

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

\ltx@pkgextension

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

2.13.2 Load check

\ltx@iffileloaded

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

\ltx@ifclassloaded

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

\ltx@ifpackageloaded

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

2.13.3 Version date check

\ltx@iffilelater

```
\def\ltx@iffilelater#1#2{%
  \ltx@iffileloaded{#1}{%
    \expandafter\LTXcmds@IfLater\expandafter{%
      \number
      \expandafter\expandafter\expandafter\LTXcmds@ParseVersion
      \expandafter\expandafter\expandafter{%
        \csname ver@#1\endcsname
      }%
    \expandafter}\expandafter{%
      \number
      \expandafter\LTXcmds@ParseVersion\expandafter{#2}%
    }%
  }\ltx@secondoftwo
}
```

\LTXcmds@IfLater

```
\def\LTXcmds@IfLater#1#2{%
  \ifcase 0%
      \ifnum#1<19940101 %
      \else
        \ifnum#2<19940101 %
        \else
          \ifnum#2>#1 %
          \else
            1%
          \fi
        \fi
      \fi
      \ltx@space
    \expandafter\ltx@secondoftwo
  \else
    \expandafter\ltx@firstoftwo
  \fi
}
```

\ltx@ifclasslater

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

\ltx@ifpackagelater

```
\def\ltx@ifpackagelater#1{%
```

```
  \ltx@iffilelater{#1.\ltx@pkgextension}%
}

\ltx@IfUndefined{pdfmatch}{%
```

\LTXcmds@ParseVersion

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

\LTXcmds@@ParseVersion

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

\LTXcmds@ParseVersion

```
  \def\LTXcmds@ParseVersion#1{%
    \ifnum\pdfmatch{%
      ^%
      (199[4-9]|[2-9][0-9][0-9][0-9])/%
      (0[1-9]|1[0-2])/%
      (0[1-9]|[1-2][0-9]|3[0-1])%
    }{#1}=1 %
      \ltx@StripPrefix\pdflastmatch1 %
      \ltx@StripPrefix\pdflastmatch2 %
      \ltx@StripPrefix\pdflastmatch3 %
    \else
      0%
    \fi
  }%
}
```

2.14 Macro additions

\ltx@GlobalAppendToMacro

```
\long\def\ltx@GlobalAppendToMacro#1#2{%
  \ifx\ltx@undefined#1%
    \let#1\ltx@empty
  \else
    \ifx\relax#1%
      \let#1\ltx@empty
    \fi
  \fi
  \begingroup
    \ltx@LocToksA\expandafter{#1#2}%
    \xdef#1{\the\ltx@LocToksA}%
  \endgroup
}
```

\ltx@LocalAppendToMacro

```
\long\def\ltx@LocalAppendToMacro#1#2{%
  \global\let\LTXcmds@gtemp#1%
  \ifx\ltx@undefined\LTXcmds@gtemp
    \global\let\LTXcmds@gtemp\ltx@empty
  \else
    \ifx\relax\LTXcmds@gtemp
      \global\letLTXcmds@gtemp\ltx@empty
    \fi
  \fi
```

```
  \begingroup
    \ltx@LocToksA\expandafter{\LTXcmds@gtemp#2}%
    \xdef\LTXcmds@gtemp{\the\ltx@LocToksA}%
  \endgroup
  \let#1\LTXcmds@gtemp
}
```

\ltx@GlobalPrependToMacro

```
\long\def\ltx@GlobalPrependToMacro#1#2{%
  \ifx\ltx@undefined#1%
    \let#1\ltx@empty
  \else
    \ifx\relax#1%
      \let#1\ltx@empty
    \fi
  \fi
  \begingroup
    \ltx@LocToksA{#2}%
    \ltx@LocToksB\expandafter{#1}%
    \xdef#1{\the\ltx@LocToksA\the\ltx@LocToksB}%
  \endgroup
}
```

\ltx@LocalPrependToMacro

```
\long\def\ltx@LocalPrependToMacro#1#2{%
  \global\let\LTXcmds@gtemp#1%
  \ifx\ltx@undefined\LTXcmds@gtemp
    \global\let\LTXcmds@gtemp\ltx@empty
  \else
    \ifx\relax\LTXcmds@gtemp
      \global\letLTXcmds@gtemp\ltx@empty
    \fi
  \fi
  \begingroup
    \ltx@LocToksA{#2}%
    \ltx@LocToksB\expandafter{\LTXcmds@gtemp}%
    \xdef\LTXcmds@gtemp{\the\ltx@LocToksA\the\ltx@LocToksB}%
  \endgroup
  \let#1\LTXcmds@gtemp
}
```

2.15 Next character detection

\ltx@ifnextchar

```
\long\def\ltx@ifnextchar#1#2#3{%
  \begingroup
  \let\LTXcmds@CharToken= #1\relax
  \ltx@LocToksA{\endgroup#2}%
  \ltx@LocToksB{\endgroup#3}%
  \futurelet\LTXcmds@LetToken\LTXcmds@ifnextchar
}
```

\LTXcmds@ifnextchar

```
\def\LTXcmds@ifnextchar{%
  \ifx\LTXcmds@LetToken\LTXcmds@CharToken
    \the\expandafter\ltx@LocToksA
  \else
    \expandafter
      \ifx\csname LTXcmds@LetToken\endcsname\LTXcmds@SpaceToken
      \expandafter\expandafter\expandafter\LTXcmds@@ifnextchar
    \else
      \the\expandafter\expandafter\expandafter\ltx@LocToksB
```

```
    \fi
  \fi
}
```

\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 catched by \romannumeral with negative character constant that gobbles one optional space.

```
\def\LTXcmds@@ifnextchar{%
  \expandafter\futurelet
  \expandafter\LTXcmds@LetToken
  \expandafter\LTXcmds@ifnextchar
  \romannumeral-`\.%
}
```

\LTXcmds@SpaceToken

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

\ltx@ifnextchar@nospace

```
\long\def\ltx@ifnextchar@nospace#1#2#3{%
  \begingroup
  \let\LTXcmds@CharToken= #1\relax
  \ltx@LocToksA{\endgroup#2}%
  \ltx@LocToksB{\endgroup#3}%
  \futurelet\LTXcmds@LetToken\LTXcmds@ifnextchar@nospace
}
```

\LTXcmds@ifnextchar@nospace

```
\def\LTXcmds@ifnextchar@nospace{%
  \the
  \ifx\LTXcmds@LetToken\LTXcmds@CharToken
    \expandafter\ltx@LocToksA
  \else
    \expandafter\ltx@LocToksB
  \fi
}
```

2.16 \ltx@leavevmode, \ltx@mbox

\ltx@leavevmode

```
\ltx@IfUndefined{quitvmode}{%
  \ltx@IfUndefined{leavevmode}{%
    \ltx@IfUndefined{voidb@x}{%
      \ltx@IfUndefined{newbox}{%
        \def\ltx@leavevmode{%
          \begingroup
            \setbox\ltx@zero=\hbox{}%
            \begingroup
              \setbox\ltx@zero=\hbox{\box\ltx@zero}%
            \endgroup
            \unhbox\ltx@zero
          \endgroup
        }%
      }{%
        \csname newbox\endcsname\LTXcmds@VoidBox
        \ifvoid\LTXcmds@VoidBox
        \else
          \setbox\LTXcmds@VoidBox=\hbox{}%
          \begingroup
            \setbox\LTXcmds@VoidBox=\hbox{\box\LTXcmds@VoidBox}%
          \endgroup
        \fi
```

```
      \def\ltx@leavevmode{\unhbox\LTXcmds@VoidBox}%
    }%
  }{%
    \def\ltx@leavevmode{\unhbox\voidb@x}%
  }%
 }{%
  \let\ltx@leavevmode\leavevmode
 }%
}{%
 \let\ltx@leavevmode\quitvmode
}
```

\ltx@mbox

```
\def\ltx@mbox{%
 \ltx@leavevmode
 \hbox
}
```

2.17 Help macros

\LTXcmds@num

```
\ltx@IfUndefined{numexpr}{%
 \def\LTXcmds@num#1{%
  \expandafter\ltx@firstofone\expandafter{%
   \number#1%
  }%
 }%
}{%
 \def\LTXcmds@num#1{%
  \expandafter\ltx@firstofone\expandafter{%
   \the\numexpr#1%
  }%
 }%
}
```

2.18 Expandable test for emptiness

```
\ltx@IfUndefined{detokenize}{%
```

2.18.1 Vanilla TEX

\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).

```
 \def\LTXcmds@temp#1{%
  \long\def\ltx@ifempty##1{%
   \romannumeral0%
   \iffalse{\fi
    \expandafter\ltx@gobble\expandafter{%
     \expandafter{\string##1}%
     \expandafter\ltx@gobble\string
    }%
    \expandafter\ltx@firstofthree\expandafter
    {\iffalse}\fi
    \expandafter#1\ltx@secondoftwo
   }%
   \expandafter#1\ltx@firstoftwo
  }%
```

\ltx@ifblank

```
    \long\def\ltx@ifblank##1{%
      \romannumeral0%
      \iffalse{\fi
        \expandafter\expandafter\expandafter\ltx@gobble
        \expandafter\expandafter\expandafter{%
          \expandafter\expandafter\expandafter{%
            \expandafter\string\ltx@gobble##1.%
          }%
          \expandafter\ltx@gobble\string
        }%
        \expandafter\ltx@firstofthree\expandafter
        {\iffalse}\fi
        \expandafter#1\ltx@secondoftwo
      }%
      \expandafter#1\ltx@firstoftwo
    }%
  }%
  \LTXcmds@temp{ }%
}{%
```

2.18.2 With \detokenize

Ahmed Musa provided \ifstrempty 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

```
  \long\def\ltx@ifempty#1{%
    \romannumeral%
    \csname
      LTXcmds@ifempty%
      \ifcat$\detokenize{#1}$%
        @%
      \fi
    \endcsname
  }%
```

\LTXcmds@ifempty@

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

\LTXcmds@ifempty

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

2.18.3 \ltx@ifblank

\ltx@ifblank

```
  \long\def\ltx@ifblank#1{%
    \romannumeral%
    \csname
      LTXcmds@ifempty%
      \ifcat$\detokenize\expandafter{\ltx@gobble#1.}$%
        @%
      \fi
    \endcsname
  }%
```

```
}
```

2.19 \ltx@zapspace

\ltx@zapspace

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

\LTXcmds@zapspace

```
\long\def\LTXcmds@zapspace#1 #2\@nil{%
  \ltx@ifempty{#2}{%
    #1%
  }{%
    \LTXcmds@zapspace#1#2\@nil
  }%
}
```

2.20 \ltx@IfBoxEmpty

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

```
\ltx@IfUndefined{lastnodetype}{%
  \catcode`\$=9 %
  \catcode`\&=14 %
}{%
  \catcode`\$=14 %
  \catcode`\&=9 %
}
```

\ltx@IfBoxEmpty

```
\def\ltx@IfBoxEmpty#1{%
  \ifvoid#1\relax
    \expandafter\ltx@secondoftwo
  \else
```

Implementation using ε-TEX's \lastnodetype.

```
&   \begingroup
&     \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%
&       \ifhmode\unhcopy\else\unvcopy\fi#1\relax
&       \expandafter
&     }%
&   \expandafter\endgroup
&   \ifnum\lastnodetype<\ltx@zero
&     \expandafter\expandafter\expandafter\ltx@firstoftwo
&   \else
&     \expandafter\expandafter\expandafter\ltx@secondoftwo
&   \fi
```

Implementation without ε-TEX using a signature at the beginning of the test box.

```
$   \begingroup
$     \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%
$       \penalty\ltx@one
$       \ifhmode\unhcopy\else\unvcopy\fi#1\relax
$       \expandafter
$     }%
$     \ifnum\lastpenalty=\ltx@one
```

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

```
$       \endgroup
$       \begingroup
$       \setbox\ltx@zero=\ifhbox#1\hbox\else\vbox\fi{%
$         \penalty\ltx@two
$         \ifhmode\unhcopy\else\unvcopy\fi#1\relax
```

```
$         \expandafter
$       }%
$       \ifnum\lastpenalty=\ltx@two
$         \def\next{\endgroup\expandafter\ltx@firstoftwo}%
$       \else
$         \def\next{\endgroup\expandafter\ltx@secondoftwo}%
$       \fi
$     \else
$       \def\next{\endgroup\expandafter\ltx@secondoftwo}%
$     \fi
$   \next
  \fi
}
```

\ltx@IfBoxVoidOrEmpty

```
\def\ltx@IfBoxVoidOrEmpty#1{%
  \ifvoid#1\relax
    \expandafter\ltx@thirdoffour
  \fi
  \ltx@IfBoxEmpty{#1}%
}

\LTXcmds@AtEnd%
⟨/package⟩
```

3 Test

3.1 Catcode checks for loading

```
⟨*test1⟩

\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`\@=11 %
\expandafter\ifx\csname count@\endcsname\relax
  \countdef\count@=255 %
\fi
\expandafter\ifx\csname @gobble\endcsname\relax
  \long\def\@gobble#1{}%
\fi
\expandafter\ifx\csname @firstofone\endcsname\relax
  \long\def\@firstofone#1{#1}%
\fi
\expandafter\ifx\csname loop\endcsname\relax
  \expandafter\@firstofone
\else
  \expandafter\@gobble
\fi
{%
  \def\loop#1\repeat{%
    \def\body{#1}%
    \iterate
  }%
  \def\iterate{%
    \body
      \let\next\iterate
    \else
      \let\next\relax
    \fi
    \next
  }%
  \let\repeat=\fi
```

```
}%
\def\RestoreCatcodes{}
\count@=0 %
\loop
  \edef\RestoreCatcodes{%
    \RestoreCatcodes
    \catcode\the\count@=\the\catcode\count@\relax
  }%
\ifnum\count@<255 %
  \advance\count@ 1 %
\repeat

\def\RangeCatcodeInvalid#1#2{%
  \count@=#1\relax
  \loop
    \catcode\count@=15 %
  \ifnum\count@<#2\relax
    \advance\count@ 1 %
  \repeat
}
\def\RangeCatcodeCheck#1#2#3{%
  \count@=#1\relax
  \loop
    \ifnum#3=\catcode\count@
    \else
      \errmessage{%
        Character \the\count@\space
        with wrong catcode \the\catcode\count@\space
        instead of \number#3%
      }%
    \fi
  \ifnum\count@<#2\relax
    \advance\count@ 1 %
  \repeat
}
\def\space{ }
\expandafter\ifx\csname LoadCommand\endcsname\relax
  \def\LoadCommand{\input ltxcmds.sty\relax}%
\fi
\def\Test{%
  \RangeCatcodeInvalid{0}{47}%
  \RangeCatcodeInvalid{58}{64}%
  \RangeCatcodeInvalid{91}{96}%
  \RangeCatcodeInvalid{123}{255}%
  \catcode`\@=12 %
  \catcode`\\=0 %
  \catcode`\%=14 %
  \LoadCommand
  \RangeCatcodeCheck{0}{36}{15}%
  \RangeCatcodeCheck{37}{37}{14}%
  \RangeCatcodeCheck{38}{47}{15}%
  \RangeCatcodeCheck{48}{57}{12}%
  \RangeCatcodeCheck{58}{63}{15}%
  \RangeCatcodeCheck{64}{64}{12}%
  \RangeCatcodeCheck{65}{90}{11}%
  \RangeCatcodeCheck{91}{91}{15}%
  \RangeCatcodeCheck{92}{92}{0}%
  \RangeCatcodeCheck{93}{96}{15}%
  \RangeCatcodeCheck{97}{122}{11}%
  \RangeCatcodeCheck{123}{255}{15}%
  \RestoreCatcodes
}
```

```
\Test
\csname @@end\endcsname
\end
```

⟨/test1⟩

3.2 Test \ltx@GobbleNum

```
⟨*test-gobble⟩
\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input ltxcmds.sty\relax
\else
  \RequirePackage{ltxcmds}[2016/05/16]%
\fi
\catcode`\@=11 %
\def\msg#{\immediate\write16}%
\msg{[Test \string\ltx@GobbleNum]}%
\long\def\Test#1=#2\\{%
  \edef\StrA{\ltx@GobbleNum#1}%
  \expandafter\expandafter\expandafter\def
  \expandafter\expandafter\expandafter\StrAA
  \expandafter\expandafter\expandafter{\ltx@GobbleNum#1}%
  \edef\StrB{#2}%
  \ifx\StrA\StrB
    \ifx\StrAA\StrB
      \msg{* ok.}%
    \else
      \msg{StrAA: \StrAA}%
      \msg{StrB: \StrB}%
      \errhelp{Test: #1=#2}%
      \errmessage{Test (two expansions) failed}%
    \fi
  \else
    \msg{StrA: \StrA}%
    \msg{StrB: \StrB}%
    \errhelp{Test: #1=#2}%
    \errmessage{Test (edef) failed!}%
  \fi
}
\Test0abc=abc\\
\Test1abc=bc\\
\Test2abc=c\\
\Test3abcd=d\\
\Test4abcde=e\\
\Test5abcdef=f\\
\Test6abcdefg=g\\
\Test7abcdefgh=h\\
\Test8abcdefghi=i\\
\Test9abcdefghij=j\\
\Test{10}0123456789X=X\\
\Test{12}abcdefghijklm=m\\
\Test{700}%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
```

```
0123456789012345678901234567890123456789012345678901234567890123456789%
X=X\\
\Test{-1}abc=abc\\
\Test2\par\par\relax=\relax\\

\begingroup
  \count1=2 %
  \Test{\count1}abc=c\\%
\endgroup

\ltx@IfUndefined{numexpr}{%
}{%
  \Test{1+1}abc=c\\%
}

\msg{[Test \string\ltx@CdrNum]}%
\long\def\Test#1=#2\\{%
  \edef\StrA{\ltx@CdrNum#1\@nil}%
  \expandafter\expandafter\expandafter\def
  \expandafter\expandafter\expandafter\StrAA
  \expandafter\expandafter\expandafter{\ltx@CdrNum#1\@nil}%
  \edef\StrB{#2}%
  \ifx\StrA\StrB
    \ifx\StrAA\StrB
      \msg{* ok.}%
    \else
      \msg{StrAA: \meaning\StrAA}%
      \msg{StrB: \meaning\StrB}%
      \errhelp{Test: #1=#2}%
      \errmessage{Test (two expansions) failed}%
    \fi
  \else
    \msg{StrA: \StrA}%
    \msg{StrB: \StrB}%
    \errhelp{Test: #1=#2}%
    \errmessage{Test (edef) failed!}%
  \fi
}
\Test0abc=abc\\
\Test1abc=bc\\
\Test2abc=c\\
\Test3abcd=d\\
\Test4abcde=e\\
\Test5abcdef=f\\
\Test6abcdefg=g\\
\Test7abcdefgh=h\\
\Test8abcdefghi=i\\
\Test9abcdefghij=j\\
\Test{10}0123456789X=X\\
\Test{12}abcdefghijklm=m\\
\Test{700}%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
X=X\\
```

```
\Test{-1}abc=abc\\
\Test2\par\par\relax=\relax\\

\msg{[Test \string\ltx@CarNum]}%
\long\def\Test#1=#2\\{%
  \edef\StrA{\ltx@CarNum#1\@nil}%
  \expandafter\expandafter\expandafter\def
  \expandafter\expandafter\expandafter\StrAA
  \expandafter\expandafter\expandafter{\ltx@CarNum#1\@nil}%
  \edef\StrB{#2}%
  \ifx\StrA\StrB
    \ifx\StrAA\StrB
      \msg{* ok.}%
    \else
      \msg{StrAA: \meaning\StrAA}%
      \msg{StrB: \meaning\StrB}%
      \errhelp{Test: #1=#2}%
      \errmessage{Test (two expansions) failed}%
    \fi
  \else
    \msg{StrA: \StrA}%
    \msg{StrB: \StrB}%
    \errhelp{Test: #1=#2}%
    \errmessage{Test (edef) failed!}%
  \fi
}
\Test0abc=\\
\Test1abc=a\\
\Test2abc=ab\\
\Test3abc=abc\\
\Test3abcd=abc\\
\Test4abcde=abcd\\
\Test{10}0123456789X=0123456789\\
\Test{12}abcdefghijklm=abcdefghijkl\\
\Test{700}%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
X=%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
0123456789012345678901234567890123456789012345678901234567890123456789%
\\
\Test{-1}abc=\\
\Test2\par\par\relax=\par\par\\
\csname @@end\endcsname\end
⟨/test-gobble⟩
```

3.3 Test \ltx@ifempty

```
⟨*test-ifempty⟩
\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`\@=11 %
\errorcontextlines=1000 %
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input ltxcmds.sty\relax
\else
  \RequirePackage{ltxcmds}[2016/05/16]%
\fi
\def\msg#{\immediate\write16}
\def\TestY{\Y}
\def\TestN{\N}
\msg{* \string\ltx@ifempty}
\long\def\test#1{%
  \begingroup
    % Calculate expected test result via macro definition
    \def\Stuff{#1}%
    \ifx\Stuff\ltx@empty
      \def\StuffEmpty{\Y}%
    \else
      \def\StuffEmpty{\N}%
    \fi
    % Test \ltx@ifempty
    \expandafter\expandafter\expandafter\def
    \expandafter\expandafter\expandafter\TestEmpty
    \expandafter\expandafter\expandafter{%
      \ltx@ifempty{#1}{\Y}{\N}%
    }%
    \ifx\StuffEmpty\TestEmpty
      \msg{* Test OK}%
    \else
      \ltx@IfUndefined{detokenize}{}{%
        \msg{Stuff: [\detokenize{\Stuff}]}%
      }%
      \errmessage{Test failed!}%
    \fi
  \endgroup
}
\test{}
\test{a}
\test{abc}
\test{\par}
\test{ }
\test{\if}
\test{{\if}}
\test{\else}
\test{{\else}}
\test{\fi}
\test{{}\fi}
\test{\or\ifcase}
\test{{}}
\test{{a}}
\test{{}abc}
\test{{\par}}
\test{{}\par}
\def\SpaceTwo#1{%
  \def\SpaceTwo{#1#1}%
}\SpaceTwo{ }
```

```
\msg{* \string\ltx@ifblank}
\long\def\test#1{%
  \begingroup
    % Calculate expected test result via macro definition
    \def\Stuff{#1}%
    \ifx\Stuff\ltx@empty
      \def\StuffEmpty{\Y}%
    \else
      \ifx\Stuff\ltx@space
        \def\StuffEmpty{\Y}%
      \else
        \ifx\Stuff\SpaceTwo
          \def\StuffEmpty{\Y}%
        \else
          \def\StuffEmpty{\N}%
        \fi
      \fi
    \fi
    % Test \ltx@ifblank
    \expandafter\expandafter\expandafter\def
    \expandafter\expandafter\expandafter\TestEmpty
    \expandafter\expandafter\expandafter{%
      \ltx@ifblank{#1}{\Y}{\N}%
    }%
    \ifx\StuffEmpty\TestEmpty
      \msg{* Test OK}%
    \else
      \ltx@IfUndefined{detokenize}{}{%
        \msg{Stuff: [\detokenize{\Stuff}]}%
      }%
      \errmessage{Test failed!}%
    \fi
  \endgroup
}
\test{}
\test{a}
\test{\if}
\test{\else}
\test{\fi}
\test{ \fi}
\test{\par}
\test{ \par}
\test{{}}
\test{ {}}
\def\x#1{%
  \test{#1#1}%
  \test{#1#1{}}%
  \test{#1#1\par}%
  \test{#1#1\else}%
}\x{ }
\csname @@end\endcsname\end
⟨/test-ifempty⟩
```

3.4 Test \ltx@zap@space

```
⟨*test-zapspace⟩
\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`\@=11 %
\errorcontextlines=1000 %
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
```

```
  \input ltxcmds.sty\relax
\else
  \RequirePackage{ltxcmds}[2016/05/16]%
\fi
\def\msg#{\immediate\write16}
\def\space{ }
\def\empty{}
\msg{* \string\ltx@zapspace}
\long\def\test#1#2{%
  \begingroup
    \def\TestInput{#1}%
    \def\TestExpected{#2}%
    % Test \ltx@zapspace
    \expandafter\expandafter\expandafter\def
    \expandafter\expandafter\expandafter\TestResult
    \expandafter\expandafter\expandafter{%
      \ltx@zapspace{#1}%
    }%
    \ifx\TestResult\TestExpected
      \msg{* Test OK}%
    \else
      \ltx@onelevel@sanitize\TestInput
      \ltx@onelevel@sanitize\TestExpected
      \ltx@onelevel@sanitize\TestResult
      \msg{* Input: \space\space\space[\TestInput]}%
      \msg{ \space Result: \space\space[\TestResult]}%
      \msg{ \space Expected: [\TestExpected]}%
      \errmessage{Test failed!}%
    \fi
  \endgroup
}
\long\def\etest#1#2{%
  \begingroup
    \edef\x{\endgroup
      \noexpand\test{#1}{#2}%
    }%
  \x
}
\catcode`\~=13 %
\let~\noexpand

\test{}{}
\test{{}}{{}}
\test{ {}}{{}}
\test{{ }}{{ }}
\test{{} }{{}}
\test{ {} }{{}}
\test{ { } }{{ }}
\test{a {b} c}{a{b}c}
\test{a bb ccc}{abbccc}
\test{{a} {bb} {ccc}}{{a}{bb}{ccc}}
\test{\par}{\par}
\test{\if}{\if}
\test{\space}{\space}
\etest{\par\space\par}{\par\par}
\etest{~\empty\space~\empty}{~\empty~\empty}
\etest{~\fi\space~\else\space}{~\fi~\else}

\csname @@end\endcsname\end
⟨/test-zapspace⟩
```

3.5 Test \ltx@IfBoxEmpty

```
⟨*test-ifboxempty⟩
\catcode`\{=1 %
```

```
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`\@=11 %
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input ltxcmds.sty\relax
\else
  \RequirePackage{ltxcmds}[2016/05/16]%
\fi
\def\msg#{\immediate\write16}
 % make box 0 void
\begingroup
  \setbox0=\box0 %
\endgroup
\ifvoid0 %
\else
  \errmessage{Voiding box 0 failed}%
\fi
\setbox2=\box0 %
\def\test#1#2{%
  \@test{#1}{#2}%
  \@@test{#1}{#2}%
  \chardef\x=#1%
  \@test\x{#2}%
  \@@test\x{#2}%
}
\def\@test#1#2{%
  \begingroup
    \setbox9=\hbox{%
      \def\TestExpected{#2}%
      \ltx@IfBoxEmpty{#1}{%
        \def\TestResult{Y}%
      }{%
        \def\TestResult{N}%
      }%
      \ifx\TestExpected\TestResult
        \msg{* Test passed.}%
      \else
        \errmessage{Test failed!}%
      \fi
    }%
    \ifdim\wd9=0pt %
    \else
      \errmessage{Unwanted space?}%
    \fi
  \endgroup
}
\def\@@test#1#2{%
  \begingroup
    \setbox9=\hbox{%
      \def\TestExpected{#2}%
      \ifvoid#1\def\TestExpected{Y}\fi
      \ltx@IfBoxVoidOrEmpty{#1}{%
        \def\TestResult{Y}%
      }{%
        \def\TestResult{N}%
      }%
      \ifx\TestExpected\TestResult
        \msg{* Test passed.}%
      \else
        \errmessage{Test failed!}%
      \fi
```

```
    }%
    \ifdim\wd9=0pt %
    \else
      \errmessage{Unwanted space?}%
    \fi
  \endgroup
}
\test0N
\test2N
\setbox0=\hbox{}
\test0Y
\setbox2=\hbox{}
\test2Y
\setbox0=\vbox{}
\test0Y
\setbox2=\vbox{}
\test0Y
\setbox0=\hbox{ }%
\test0N
\setbox2=\hbox{ }%
\test2N
\setbox0=\hbox{\penalty1}%
\test0N
\setbox2=\hbox{\penalty1}%
\test2N
\csname @@end\endcsname\end
⟨/test-ifboxempty⟩
```

3.6 Test for next character detection

```
⟨*test-nextchar⟩
\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`\@=11 %
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input ltxcmds.sty\relax
  \input eolgrab.sty\relax
\else
  \RequirePackage{ltxcmds}[2016/05/16]%
  \RequirePackage{eolgrab}[2011/01/12]%
\fi
\def\msg#{\immediate\write16}
\begingroup
  \def\x#1{%
    \endgroup
    \let\TestSpaceToken= #1\relax
  }%
\x{ }
\def\TestSpace{ }
\begingroup
  \lccode32=65 % space -> A
\lowercase{%
  \endgroup
  \def\TestSpaceA{ }%
}
\def\TestCatch{%
  \eolgrab\@TestCatch
}
\def\@TestCatch#1{%
  \begingroup
    \def\x{#1}%
```

```
    \ifx\x\ltx@empty
    \else
      \ltx@onelevel@sanitize\x
      \errmessage{Unparsed stuff on line [\x]}%
    \fi
  \endgroup
}
\def\TestCmdM#1{%
  \TestCheckType{M}%
  \TestCatch
}
\def\TestCmdOM[#1]#2{%
  \TestCheckType{O}%
  \TestCatch
}
\def\TestCheckType#1{%
  \if\TestCmdType#1\relax
  \else
    \errmessage{Wrong type #1, expected: \TestCmdType}%
  \fi
}
\def\TestCmd#1{%
  \def\TestCmdType{#1}%
  \ltx@ifnextchar[\TestCmdOM\TestCmdM
}
\def\TestCmdExp#1{%
  \expandafter\TestCmd\expandafter#1%
}
\outer\def\TestOuter{}
\TestCmd O[o]{m}
\TestCmd M{m}
\TestCmd O [o]{m}
\TestCmd M {m}
\def\x#1{\def\x{#1#1}}\x{ }
\TestCmdExp O\x[o]{m}
\TestCmdExp M\x{m}
\def\x#1{\def\x{#1#1#1#1}}\x{ }
\TestCmdExp O\x[o]{m}
\TestCmdExp M\x{m}
\def\x{\TestSpaceToken}
\TestCmdExp O\x[o]{m}
\TestCmdExp M\x{m}
\def\x{\TestSpaceToken\TestSpaceToken\TestSpaceToken}
\TestCmdExp O\x[o]{m}
\TestCmdExp M\x{m}
\TestCmd M\TestSpace
\TestOuter
\TestCmd M \TestSpace
\TestOuter
\TestCmd M\iftrue
\TestOuter
\TestCmd M\iffalse
\TestOuter
\TestCmd M\else
\TestOuter
\TestCmd M\fi
\TestOuter
\TestCmd M \iftrue
\TestOuter
\TestCmd M \iffalse
\TestOuter
\TestCmd M \else
```

```
\TestOuter
%
\def\TestCmd#1{%
  \def\TestCmdType{#1}%
  \ltx@ifnextchar@nospace[\TestCmdOM\TestCmdM
}
\TestCmd O[o]{m}
\TestCmd M{m}
\TestCmd M [
\TestOuter
\TestCmd M {m}
\TestCmd M\iftrue
\TestOuter
\TestCmd M\iffalse
\TestOuter
\TestCmd M\else
\TestCmd M\fi
\TestOuter
\TestOuter
%
\def\TestCmd#1{%
  \def\TestCmdType{#1}%
  \ltx@ifnextchar(\TestCmdPM\TestCmdM
}
\def\TestCmdPM(#1)#2{%
  \TestCheckType{P}%
  \TestCatch
}
\TestCmd P(p){m}
\TestCmd M{m}
\TestCmd P (p){m}
\TestCmd M {m}
%
\def\TestCmd#1{%
  \def\TestCmdType{#1}%
  \ltx@ifnextchar{ }\TestCmdSM\TestCmdM
}
\def\TestCmdSM#1#{%
  \TestCheckType{S}%
  \begingroup
    \let\x= #1\relax
    \ifx\x\TestSpaceToken
    \else
      \errmessage{unexpected space token: \meaning#1}%
    \fi
  \endgroup
  \def\TestCmdType{M}%
  \TestCmdM
}
\TestCmd S {m}
\TestCmd M{m}
\def\x#1{\def\x{#1#1}}\x{ }
\TestCmdExp S\x{m}
%
\def\TestCmd#1{%
  \def\TestCmdType{#1}%
  \ltx@ifnextchar\iffalse\TestCmdIM\TestCmdM
}
\def\TestCmdIM\iffalse#1{%
  \TestCheckType{I}%
  \TestCatch
}
```

```
\TestCmd M\iftrue
\TestOuter
\TestCmd M \iftrue
\TestCmd I\iffalse\iffalse
\TestCmd I \iffalse\iffalse
\TestOuter
%
\def\TestCmd#1{%
  \def\TestCmdType{#1}%
  \ltx@ifnextchar@nospace\iffalse\TestCmdIM\TestCmdM
}
\TestCmd M\iftrue
\TestOuter
\TestCmd I\iffalse\iffalse
\TestOuter
\csname @@end\endcsname\end
⟨/test-nextchar⟩
```

3.7 Test for list helpers

```
⟨*test-carcdr⟩
\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`\@=11 %
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input ltxcmds.sty\relax
  \input eolgrab.sty\relax
\else
  \RequirePackage{ltxcmds}[2016/05/16]%
  \RequirePackage{eolgrab}[2011/01/12]%
\fi
\def\msg#{\immediate\write16}
\def\space{ }
\long\def\Test#1#2#3{%
  \begingroup
    \def\TestExpected{#3}%
    \expandafter\expandafter\expandafter\def
    \expandafter\expandafter\expandafter\TestResult
    \expandafter\expandafter\expandafter{%
      #1#2\@nil
    }%
    \ifx\TestResult\TestExpected
    \else
      \msg{\string\TestExpected: [\meaning\TestExpected]}%
      \msg{\string\TestResult: \space\space[\meaning\TestResult]}%
      \errmessage{Test failed!}%
    \fi
  \endgroup
}
\Test\ltx@carzero{abc}{}
\Test\ltx@carzero{}{}
\Test\ltx@carzero{\par\par}{}
\Test\ltx@cdrzero{}{}
\Test\ltx@cdrzero{abc}{abc}
\Test\ltx@cdrzero{ \par}{ \par}
\Test\ltx@cdrzero{\@empty}{\@empty}
\Test\ltx@cdrzero{{}}{{}}
\Test\ltx@car{abc}{a}
\Test\ltx@car{\par}{\par}
\Test\ltx@cdr{abc}{bc}
\Test\ltx@cdr{a \par}{ \par}
```

```
\Test\ltx@cdr{a\@empty}{\@empty}
\Test\ltx@cartwo{abc}{ab}
\Test\ltx@cartwo{\par\@empty}{\par\@empty}
\Test\ltx@carsecond{abc}{b}
\Test\ltx@carsecond{\@empty b\@empty}{b}
\Test\ltx@carsecond{\par\par\par}{\par}
\Test\ltx@cdrtwo{abc}{c}
\Test\ltx@cdrtwo{ab \par}{ \par}
\Test\ltx@cdrtwo{ab\@empty}{\@empty}
\Test\ltx@cdrtwo{ab{}}{{}}
\Test\ltx@cdrthree{abcdefg}{defg}
\Test\ltx@cdrfour{abcdefg}{efg}
\Test{\ltx@CdrNum{5}}{abcdefg}{fg}
\Test{\ltx@CdrNum{0}}{\par}{\par}
\Test{\ltx@CdrNum{0}}{\@empty}{\@empty}
\Test{\ltx@CdrNum{0}}{{}}{{}}
\Test{\ltx@CdrNum{0}}{ }{ }
\Test{\ltx@CdrNum{2}}{abcd}{cd}
\Test{\ltx@CdrNum{2}}{\vbox\par\hbox\par}{\hbox\par}
\Test{\ltx@carthree}{abcdefg}{abc}
\Test{\ltx@carfour}{abcdefg}{abcd}
\Test{\ltx@CarNum{5}}{abcdefg}{abcde}
\Test{\ltx@CarNum{2}}{\@empty\par}{\@empty\par}
\Test\ltx@carthird{abcdefg}{c}
\Test\ltx@carfourth{abcdefg}{d}
\Test{\ltx@CarNumth{5}}{abcdefg}{e}
\Test{\ltx@CarNumth{2}}{\@empty\@empty\@empty}{\@empty}
\Test{\ltx@CarNumth{2}}{\par\par\par}{\par}
\Test{\ltx@CarNumth{2}}{ab}{b}
\csname @@end\endcsname\end
⟨/test-carcdr⟩
```

4 Installation

4.1 Download

Package. This package is available on CTAN[1]:

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 TEX Files" (CTAN:tds/tds.pdf). Directories with texmf in their name are usually organized this way.

4.2 Bundle installation

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

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

[1] http://ctan.org/pkg/ltxcmds

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 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/oberdiek/ltxcmds.sty
ltxcmds.pdf                  → doc/latex/oberdiek/ltxcmds.pdf
test/ltxcmds-test1.tex       → doc/latex/oberdiek/test/ltxcmds-test1.tex
test/ltxcmds-test-gobble.tex → doc/latex/oberdiek/test/ltxcmds-test-gobble.tex
test/ltxcmds-test-ifempty.tex → doc/latex/oberdiek/test/ltxcmds-test-ifempty.tex
test/ltxcmds-test-zapspace.tex → doc/latex/oberdiek/test/ltxcmds-test-zapspace.tex
test/ltxcmds-test-ifboxempty.tex → doc/latex/oberdiek/test/ltxcmds-test-ifboxempty.tex
test/ltxcmds-test-nextchar.tex → doc/latex/oberdiek/test/ltxcmds-test-nextchar.tex
test/ltxcmds-test-carcdr.tex → doc/latex/oberdiek/test/ltxcmds-test-carcdr.tex
ltxcmds.dtx                  → source/latex/oberdiek/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.

4.4 Refresh file name databases

If your TEX distribution (teTEX, mikTEX, ...) relies on file name databases, you must refresh these. For example, teTEX users run texhash or mktexlsr.

4.5 Some details for the interested

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

plain TEX: Run docstrip and extract the files.

LATEX: Generate the documentation.

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

```
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 pdfLATEX:

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

5 Catalogue

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

```
⟨*catalogue⟩
<?xml version='1.0' encoding='us-ascii'?>
<!DOCTYPE entry SYSTEM 'catalogue.dtd'>
<entry datestamp='$Date$' modifier='$Author$' id='ltxcmds'>
  <name>ltxcmds</name>
  <caption>Some LaTeX kernel commands for general use.</caption>
  <authorref id='auth:oberdiek'/>
  <copyright owner='Heiko Oberdiek' year='2009-2011'/>
  <license type='lppl1.3'/>
  <version number='1.23'/>
  <description>
    This package exports some utility macros
    from the LaTeX kernel into a separate namespace and
    also makes them available for other formats such as plain TeX.
    <p/>
    The package is part of the <xref refid='oberdiek'>oberdiek</xref>
    bundle.
  </description>
  <documentation details='Package documentation'
      href='ctan:/macros/latex/contrib/oberdiek/ltxcmds.pdf'/>
  <ctan file='true' path='/macros/latex/contrib/oberdiek/ltxcmds.dtx'/>
  <miktex location='oberdiek'/>
  <texlive location='oberdiek'/>
  <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip'/>
</entry>
⟨/catalogue⟩
```

6 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:idbo94uka1@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

7 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.

[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.

8 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

\# 254, 709, 808, 987, 1100, 1166, 1257, 1431
\$ 652, 655
\% 244, 785
\& 653, 656
\. 517
\@ 710, 783, 814, 988, 1101, 1167, 1258, 1432
\@@test 1186, 1189, 1212
\@TestCatch 1282, 1284
\@empty 1465, 1471, 1473, 1475, 1479, 1485, 1493, 1497
\@firstofone 718, 721
\@gobble 715, 723
\@nil . . 180, 181, 185, 189, 190, 194, 195, 199, 200, 219, 229, 272, 277, 294, 300, 415, 417, 642, 644, 648, 878, 881, 928, 931, 1449
\@test 1185, 1188, 1191
\@undefined 58
\\ 249, 784, 817, 839, 840, 841, 842, 843, 844, 845, 846, 847, 848, 849, 850, 862, 863, 864, 868, 873, 877, 899, 900, 901, 902, 903, 904, 905, 906, 907, 908, 909, 910, 922, 923, 924, 927, 949, 950, 951, 952, 953, 954, 955, 956, 979, 980, 981
\{ 259, 707, 806, 985, 1098, 1164, 1255, 1429
\} 264, 708, 807, 986, 1099, 1165, 1256, 1430
\~ 1143

A

\advance 748, 756, 771
\aftergroup 29

B

\body 727, 731
\box 543, 554, 1177, 1183

C

\catcode 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, 707, 708, 709, 710, 745, 754, 762, 766, 783, 784, 785, 806, 807, 808, 814, 985, 986, 987, 988, 1098, 1099, 1100, 1101, 1143, 1164, 1165, 1166, 1167, 1255, 1256, 1257, 1258, 1429, 1430, 1431, 1432

\chardef . . 116, 117, 118, 119, 120, 1187
\count 867, 868
\count@ 712, 741, 745, 747, 748, 752, 754, 755, 756, 760, 762, 765, 766, 770, 771
\countdef 712
\csname 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, 711, 714, 717, 720, 775, 802, 809, 982, 991, 1095, 1104, 1161, 1169, 1252, 1260, 1426, 1434, 1500

D

\detokenize 623, 634, 1019, 1073
\dimendef 134, 135, 136, 137, 138, 139, 140, 141, 142, 143

E

\empty 17, 18, 1111, 1159
\end 803, 982, 1095, 1161, 1252, 1426, 1500
\endcsname . 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, 711, 714, 717, 720, 775, 802, 809, 982, 991, 1095, 1104, 1161, 1169, 1252, 1260, 1426, 1434, 1500
\endinput 29, 115
\endlinechar 4, 35, 71, 77, 89
\eolgrab 1282
\errhelp . . . 829, 835, 889, 895, 939, 945
\errmessage 764, 830, 836, 890, 896, 940, 946, 1021, 1075, 1132, 1181, 1203, 1208, 1225, 1230, 1290, 1305, 1392, 1455
\errorcontextlines 989, 1102
\escapechar 270, 275, 292, 297
\etest 1136, 1158, 1159, 1160

F

\futurelet 499, 514, 525

H

\hbox 541, 543, 552, 554, 570, 663, 674, 682, 1193, 1214, 1236, 1238, 1244, 1246, 1248, 1250, 1489

I

\if 277, 300, 1030, 1031, 1081, 1156, 1303
\ifcase 390, 1036
\ifcat 623, 634
\ifcsname 342
\ifdim 1206, 1228
\iffalse 286, 309, 589, 595, 602, 611, 1338, 1346, 1362, 1405, 1407, 1414, 1415, 1420, 1424
\ifhbox 663, 674, 682
\ifhmode 664, 676, 684
\ifnum 391, 393, 395, 422, 668, 679, 687, 747, 755, 762, 770
\iftrue 281, 304, 1336, 1344, 1360, 1411, 1413, 1422
\ifvoid 550, 659, 699, 1179, 1216
\ifx 15, 18, 21, 50, 58, 61, 320, 322, 330, 343, 437, 440, 451, 454, 465, 468, 480, 483, 502, 506, 529, 711, 714, 717, 720, 775, 809, 823, 824, 883, 884, 933, 934, 991, 1004, 1015, 1050, 1053, 1056, 1069, 1104, 1123, 1169, 1200, 1222, 1260, 1287, 1390, 1434, 1451
\immediate 23, 52, 815, 996, 1109, 1174, 1267, 1441
\input 776, 810, 992, 1105, 1170, 1261, 1262, 1435, 1436
\iterate 728, 730, 732

L

\lastnodetype 668
\lastpenalty 679, 687
\lccode . . 244, 249, 254, 259, 264, 1276
\leavevmode 563
\letLTXcmds@gtemp 455, 484
\LoadCommand 776, 786
\loop 726, 742, 753, 761
\lowercase 245, 250, 255, 260, 265, 1277
\ltx@(Loc,Glob)(Toks,Dimen,Skip)(A,B,C,D,E) *3*
\ltx@active 119
\ltx@backslashchar 248
\ltx@car *5*, 185, 1467, 1468
\ltx@carfour *5*, 199, 1491
\ltx@carfourth 200, 1495
\ltx@CarNum *5*, 204, 926, 928, 931, 1492, 1493
\ltx@CarNumth 223, 1496, 1497, 1498, 1499
\ltx@carsecond . . 190, 1474, 1475, 1476
\ltx@carthird 195, 1494
\ltx@carthree *5*, 194, 1490
\ltx@cartwo *5*, 189, 1472, 1473
\ltx@carzero . . *5*, 180, 1459, 1460, 1461
\ltx@cclv 120
\ltx@cdr 186, 1469, 1470, 1471
\ltx@cdrfour 201, 1482
\ltx@CdrNum 233, 876, 878, 881, 1483, 1484, 1485, 1486, 1487, 1488, 1489
\ltx@cdrthree 196, 1481
\ltx@cdrtwo 191, 1477, 1478, 1479, 1480
\ltx@cdrzero 182, 235, 1462, 1463, 1464, 1465, 1466
\ltx@clsextension *7*, 364, 370, 408
\ltx@empty *5*, 241, 438, 441, 452, 455, 466, 469, 481, 484, 1004, 1050, 1287
\ltx@firstoffour 176
\ltx@firstofone *4*, 170, 338, 519, 574, 580
\ltx@firstofthree 173, 594, 610
\ltx@firstoftwo . . 171, 323, 331, 344, 349, 367, 404, 598, 614, 669, 688

\ltx@fourthoffour 179
\ltx@GlobalAppendToMacro . . . 8, 436
\ltx@GlobalPrependToMacro . . . 8, 464
\ltx@GlobDimenA 139
\ltx@GlobDimenB 140
\ltx@GlobDimenC 141
\ltx@GlobDimenD 142
\ltx@GlobDimenE 143
\ltx@GlobSkipA 149
\ltx@GlobSkipB 150
\ltx@GlobSkipC 151
\ltx@GlobSkipD 152
\ltx@GlobSkipE 153
\ltx@GlobToksA 129
\ltx@GlobToksB 130
\ltx@GlobToksC 131
\ltx@GlobToksD 132
\ltx@GlobToksE 133
\ltx@gobble 3, 154, 336, 590, 592, 603, 606, 608, 634
\ltx@gobblefour 157
\ltx@GobbleNum 3, 158, 227, 237, 816, 818, 821
\ltx@gobblethree 156
\ltx@gobbletwo 155
\ltx@hashchar 253
\ltx@ifblank 9, 600, 630, 1045, 1063, 1067
\ltx@IfBoxEmpty 9, 658, 702, 1195
\ltx@IfBoxVoidOrEmpty . . 9, 698, 1217
\ltx@ifclasslater 7, 407
\ltx@ifclassloaded 7, 369
\ltx@ifempty 9, 586, 619, 645, 999, 1009, 1013
\ltx@iffilelater 375, 408, 411
\ltx@iffileloaded . . . 7, 366, 370, 373, 376
\ltx@ifnextchar 8, 494, 1310, 1371, 1384, 1405
\ltx@ifnextchar@nospace 8, 520, 1353, 1420
\ltx@ifpackagelater 410
\ltx@ifpackageloaded 372
\ltx@IfUndefined 6, 328, 352, 413, 535, 536, 537, 538, 572, 585, 651, 871, 1018, 1072
\ltx@ifundefined . . 6, 321, 341, 352, 367
\ltx@leavevmode 8, 535, 569
\ltx@leftbracechar 258
\ltx@LocalAppendToMacro 449
\ltx@LocalExpandAfter 6, 313, 319
\ltx@LocalPrependToMacro 478
\ltx@LocDimenA 134
\ltx@LocDimenB 135
\ltx@LocDimenC 136
\ltx@LocDimenD 137
\ltx@LocDimenE 138
\ltx@LocSkipA 144
\ltx@LocSkipB 145
\ltx@LocSkipC 146
\ltx@LocSkipD 147
\ltx@LocSkipE 148
\ltx@LocToksA 124, 445, 446, 459, 460, 473, 475, 488, 490, 497, 503, 523, 530
\ltx@LocToksB 125, 474, 475, 489, 490, 498, 509, 524, 532
\ltx@LocToksC 126
\ltx@LocToksD 127
\ltx@LocToksE 128
\ltx@mbox 9, 568
\ltx@minusone 121
\ltx@newglobalif 6, 290
\ltx@newif 6, 268
\ltx@one 117, 122, 675, 679
\ltx@onelevel@sanitize 7, 358, 1126, 1127, 1128, 1289
\ltx@percentchar 243
\ltx@pkgextension 365, 373, 411
\ltx@RemovePrefix . . . 6, 354, 356, 361
\ltx@ReturnAfterElseFi 240
\ltx@ReturnAfterFi 5, 239
\ltx@rightbracechar 263
\ltx@secondoffour 177
\ltx@secondofthree 174
\ltx@secondoftwo 172, 325, 333, 346, 367, 387, 402, 596, 612, 660, 671, 690, 693
\ltx@space 6, 242, 401, 1053
\ltx@StripPrefix . . . 355, 428, 429, 430
\ltx@thirdoffour 178, 700
\ltx@thirdofthree 175
\ltx@two 118, 683, 687
\ltx@undefined 437, 451, 465, 480
\ltx@zapspace . 9, 640, 1112, 1117, 1121
\ltx@zero 3, 116, 183, 187, 192, 197, 202, 220, 230, 236, 541, 543, 545, 642, 663, 668, 674, 682
\LTXcmds@@ifnextchar 507, 513
\LTXcmds@@ParseVersion 415, 417
\LTXcmds@AtEnd 95, 96, 115, 704
\LTXcmds@CarNum 207, 210
\LTXcmds@CarNumFinish 219
\LTXcmds@CarNumth 226, 229
\LTXcmds@cdrzero 181, 183, 187, 192, 197, 202
\LTXcmds@CharToken 496, 502, 522, 529
\LTXcmds@Cm 213
\LTXcmds@Cx 216
\LTXcmds@Gm 167
\LTXcmds@GobbleNum 161, 164
\LTXcmds@gtemp 450, 451, 452, 454, 459, 460, 462, 479, 480, 481, 483, 489, 490, 492
\LTXcmds@ifempty 629
\LTXcmds@ifempty@ 628
\LTXcmds@IfLater 377, 389
\LTXcmds@ifnextchar . . . 499, 501, 516
\LTXcmds@ifnextchar@nospace 525, 527
\LTXcmds@LetToken 499, 502, 515, 525, 529
\LTXcmds@newglobalif 294, 296
\LTXcmds@newif 272, 274
\LTXcmds@num 162, 208, 572

\LTXcmds@ParseVersion 379, 385, 414, 421
\LTXcmds@SpaceToken 506, 519
\LTXcmds@temp 586, 617
\LTXcmds@VoidBox 549, 550, 552, 554, 557
\LTXcmds@zapspace 642, 644

M

\meaning 361, 887, 888, 937, 938, 1392, 1453, 1454
\msg 815, 816, 825, 827, 828, 833, 834, 876, 885, 887, 888, 893, 894, 926, 935, 937, 938, 943, 944, 996, 999, 1016, 1019, 1045, 1070, 1073, 1109, 1112, 1124, 1129, 1130, 1131, 1174, 1201, 1223, 1267, 1441, 1453, 1454

N

\N 998, 1007, 1013, 1059, 1067
\next . . 688, 690, 693, 695, 732, 734, 736
\number 378, 384, 575, 767
\numexpr 581

O

\outer 1315

P

\PackageInfo 26
\par 864, 924, 981, 1028, 1040, 1041, 1085, 1086, 1092, 1155, 1158, 1461, 1464, 1468, 1470, 1473, 1476, 1478, 1484, 1489, 1493, 1498
\pdflastmatch 428, 429, 430
\pdfmatch 422
\penalty 675, 683, 1248, 1250
\ProvidesPackage 19, 67

Q

\quitvmode 566

R

\RangeCatcodeCheck 759, 787, 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798
\RangeCatcodeInvalid 751, 779, 780, 781, 782
\repeat 726, 738, 749, 757, 772
\RequirePackage 812, 994, 1107, 1172, 1264, 1265, 1438, 1439
\RestoreCatcodes . . 740, 743, 744, 799
\romannumeral . 159, 162, 183, 187, 192, 197, 202, 205, 208, 224, 234, 517, 588, 601, 620, 631, 641

S

\setbox 541, 543, 552, 554, 663, 674, 682, 1177, 1183, 1193, 1214, 1236, 1238, 1240, 1242, 1244, 1246, 1248, 1250
\skipdef 144, 145, 146, 147, 148, 149, 150, 151, 152, 153
\space 765, 766, 774, 1110, 1129, 1130, 1131, 1157, 1158, 1159, 1160, 1442, 1454
\SpaceTwo 1042, 1043, 1044, 1056
\StrA 818, 823, 833, 878, 883, 893, 928, 933, 943
\StrAA 820, 824, 827, 880, 884, 887, 930, 934, 937
\StrB 822, 823, 824, 828, 834, 882, 883, 884, 888, 894, 932, 933, 934, 938, 944
\Stuff 1003, 1004, 1019, 1049, 1050, 1053, 1056, 1073
\StuffEmpty 1005, 1007, 1015, 1051, 1054, 1057, 1059, 1069

T

\Test 778, 801, 817, 839, 840, 841, 842, 843, 844, 845, 846, 847, 848, 849, 850, 851, 863, 864, 868, 873, 877, 899, 900, 901, 902, 903, 904, 905, 906, 907, 908, 909, 910, 911, 923, 924, 927, 949, 950, 951, 952, 953, 954, 955, 956, 957, 980, 981, 1443, 1459, 1460, 1461, 1462, 1463, 1464, 1465, 1466, 1467, 1468, 1469, 1470, 1471, 1472, 1473, 1474, 1475, 1476, 1477, 1478, 1479, 1480, 1481, 1482, 1483, 1484, 1485, 1486, 1487, 1488, 1489, 1490, 1491, 1492, 1493, 1494, 1495, 1496, 1497, 1498, 1499
\test 1000, 1025, 1026, 1027, 1028, 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036, 1037, 1038, 1039, 1040, 1041, 1046, 1079, 1080, 1081, 1082, 1083, 1084, 1085, 1086, 1087, 1088, 1090, 1091, 1092, 1093, 1113, 1139, 1145, 1146, 1147, 1148, 1149, 1150, 1151, 1152, 1153, 1154, 1155, 1156, 1157, 1184, 1234, 1235, 1237, 1239, 1241, 1243, 1245, 1247, 1249, 1251
\TestCatch 1281, 1296, 1300, 1375, 1409
\TestCheckType 1295, 1299, 1302, 1374, 1387, 1408
\TestCmd . . 1308, 1313, 1316, 1317, 1318, 1319, 1332, 1334, 1336, 1338, 1340, 1342, 1344, 1346, 1348, 1351, 1355, 1356, 1357, 1359, 1360, 1362, 1364, 1365, 1369, 1377, 1378, 1379, 1380, 1382, 1398, 1399, 1403, 1411, 1413, 1414, 1415, 1418, 1422, 1424
\TestCmdExp 1312, 1321, 1322, 1324, 1325, 1327, 1328, 1330, 1331, 1401
\TestCmdIM 1405, 1407, 1420
\TestCmdM 1294, 1310, 1353, 1371, 1384, 1396, 1405, 1420
\TestCmdOM 1298, 1310, 1353
\TestCmdPM 1371, 1373

\TestCmdSM 1384, 1386
\TestCmdType . . 1303, 1305, 1309, 1352, 1370, 1383, 1395, 1404, 1419
\TestEmpty 1011, 1015, 1065, 1069
\TestExpected 1116, 1123, 1127, 1131, 1194, 1200, 1215, 1216, 1222, 1445, 1451, 1453
\TestInput 1115, 1126, 1129
\TestN 998
\TestOuter 1315, 1333, 1335, 1337, 1339, 1341, 1343, 1345, 1347, 1349, 1358, 1361, 1363, 1366, 1367, 1412, 1416, 1423, 1425
\TestResult 1119, 1123, 1128, 1130, 1196, 1198, 1200, 1218, 1220, 1222, 1447, 1451, 1454
\TestSpace 1274, 1332, 1334
\TestSpaceA 1279
\TestSpaceToken 1271, 1326, 1329, 1390
\TestY 997
\the 77, 78, 79, 80, 81, 82, 83, 84, 97, 446, 460, 475, 490, 503, 509, 528, 581, 745, 765, 766
\TMP@EnsureCode 94, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114
\toksdef 124, 125, 126, 127, 128, 129, 130, 131, 132, 133

U

\unhbox 545, 557, 560
\unhcopy 664, 676, 684
\unvcopy 664, 676, 684

V

\vbox . . 663, 674, 682, 1240, 1242, 1489
\voidb@x 560

W

\wd 1206, 1228
\write 23, 52, 815, 996, 1109, 1174, 1267, 1441

X

\x 14, 15, 18, 22, 26, 28, 51, 56, 66, 75, 87, 1089, 1094, 1138, 1141, 1187, 1188, 1189, 1269, 1273, 1286, 1287, 1289, 1290, 1320, 1321, 1322, 1323, 1324, 1325, 1326, 1327, 1328, 1329, 1330, 1331, 1389, 1390, 1400, 1401

Y

\Y 997, 1005, 1013, 1051, 1054, 1057, 1067