

The ltxcmds package

Heiko Oberdiek*

2016/05/16 v1.23

Abstract

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

Contents

1	Documentation	3
1.1	Introduction	3
1.2	Numbers	3
1.3	Scratch registers	3
1.4	Argument killers	4
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@mbox</code>	9
1.17	Expandable test for emptiness	9
1.18	Stripping spaces	9
1.19	Check for emptiness of boxes	10
2	Implementation	10
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/oberdiek/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@mbox</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
3	Test	28
3.1	Catcode checks for loading	28
3.2	Test <code>\ltx@GobbleNum</code>	30
3.3	Test <code>\ltx@ifempty</code>	33
3.4	Test <code>\ltx@zap@space</code>	35
3.5	Test <code>\ltx@ifBoxEmpty</code>	36
3.6	Test for next character detection	38
3.7	Test for list helpers	41
4	Installation	42
4.1	Download	42
4.2	Bundle installation	43
4.3	Package installation	43
4.4	Refresh file name databases	43
4.5	Some details for the interested	43
5	References	44
6	History	45
	[2009/08/05 v1.0]	45
	[2009/12/12 v1.1]	45
	[2010/01/28 v1.2]	45
	[2010/03/01 v1.3]	45
	[2010/03/09 v1.4]	45
	[2010/04/08 v1.5]	45
	[2010/04/16 v1.6]	45
	[2010/04/26 v1.7]	45
	[2010/09/11 v1.8]	45
	[2010/10/25 v1.9]	46
	[2010/10/31 v1.10]	46
	[2010/11/12 v1.11]	46
	[2010/12/02 v1.12]	46
	[2010/12/04 v1.13]	46
	[2010/12/07 v1.14]	46

[2010/12/12 v1.15]	46
[2011/02/04 v1.16]	46
[2011/02/05 v1.17]	46
[2011/03/16 v1.18]	46
[2011/04/14 v1.19]	46
[2011/04/18 v1.20]	46
[2011/08/22 v1.21]	47
[2011/11/09 v1.22]	47
[2016/05/16 v1.23]	47

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	\rightarrow	0
\ltx@one	\rightarrow	1
\ltx@two	\rightarrow	2
\ltx@ccclv	\rightarrow	255
\ltx@minusone	\rightarrow	-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.

$\text{\ltx@}(\text{Loc},\text{Glob})(\text{Toks},\text{Dimen},\text{Skip})(\text{A},\text{B},\text{C},\text{D},\text{E})$
--

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

<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>	\rightarrow
<code>\ltx@cdrzero ... \@nil</code>	\rightarrow ...

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

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

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

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

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

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

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

1.8 Empty macro

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

1.9 Characters

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

1.10 Boolean switch

<code>\ltx@newif {<cmd>}</code>

`\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`.

<code>\ltx@newglobalif {<cmd>}</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 {<cmd>} {<yes>} {<no>}</code>
--

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”.

<code>\ltx@ifundefined {<cmd>} {<yes>} {<no>}</code>
--

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`.

<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 `\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 L^AT_EX’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^AT_EX’s `\@onelevel@sanitize`. The macro is expanded once and the contents is converted to characters with catcode 12 (other) and space tokens with catcode 10 (space). Then the sanitized contents is stored into the macro again. Since version 1.12.

1.13 File management

All macros in this section are expandable like the counterparts of the L^AT_EX 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 L^AT_EX’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^AT_EX’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^AT_EX 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_EX’s default extension `.tex` was given in the first command, then it must also be 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 L^AT_EX'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@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 [2016/05/16 v1.23 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{}%

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

\ltx@cdrzero
182 \def\ltx@cdrzero{%
183   \romannumeral\LTXcmds@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\LTXcmds@cdrzero\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\LTxcms@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\LTxcms@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\LTxcms@cdrzero\ltx@zero
203 }

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

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

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

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

\LTxcms@CarNumFinish
219 \long\def\LTxcms@CarNumFinish#1#2\@nil{%
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_EX

\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 ε -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 ε -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 ε -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@ifBoxVoidOrEmpty

```

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

704 \LTXcmds@AtEnd%
705 \endpackage

```

3 Test

3.1 Catcode checks for loading

```

706 (*test1)

707 \catcode`\{=1 %
708 \catcode`\}=2 %
709 \catcode`\#=6 %
710 \catcode`\@=11 %
711 \expandafter\ifx\csname count@\endcsname\relax
712   \countdef\count@=255 %
713 \fi
714 \expandafter\ifx\csname @gobble\endcsname\relax
715   \long\def\@gobble#1{}%
716 \fi
717 \expandafter\ifx\csname @firstofone\endcsname\relax
718   \long\def\@firstofone#1{#1}%
719 \fi
720 \expandafter\ifx\csname loop\endcsname\relax
721   \expandafter\@firstofone
722 \else

```

```

723 \expandafter\@gobble
724 \fi
725 {%
726 \def\loop#1\repeat{%
727 \def\body{#1}%
728 \iterate
729 }%
730 \def\iterate{%
731 \body
732 \let\next\iterate
733 \else
734 \let\next\relax
735 \fi
736 \next
737 }%
738 \let\repeat=\fi
739 }%
740 \def\RestoreCatcodes{}
741 \count@=0 %
742 \loop
743 \edef\RestoreCatcodes{%
744 \RestoreCatcodes
745 \catcode\the\count@=\the\catcode\count@\relax
746 }%
747 \ifnum\count@<255 %
748 \advance\count@ 1 %
749 \repeat
750
751 \def\RangeCatcodeInvalid#1#2{%
752 \count@=#1\relax
753 \loop
754 \catcode\count@=15 %
755 \ifnum\count@<#2\relax
756 \advance\count@ 1 %
757 \repeat
758 }
759 \def\RangeCatcodeCheck#1#2#3{%
760 \count@=#1\relax
761 \loop
762 \ifnum#3=\catcode\count@
763 \else
764 \errmessage{%
765 Character \the\count@\space
766 with wrong catcode \the\catcode\count@\space
767 instead of \number#3%
768 }%
769 \fi
770 \ifnum\count@<#2\relax
771 \advance\count@ 1 %
772 \repeat
773 }
774 \def\space{ }
775 \expandafter\ifx\csname LoadCommand\endcsname\relax
776 \def\LoadCommand{\input ltxcmds.sty\relax}%
777 \fi
778 \def\Test{%
779 \RangeCatcodeInvalid{0}{47}%
780 \RangeCatcodeInvalid{58}{64}%

```

```

781 \RangeCatcodeInvalid{91}{96}%
782 \RangeCatcodeInvalid{123}{255}%
783 \catcode`\@=12 %
784 \catcode`\=0 %
785 \catcode`\%=14 %
786 \LoadCommand
787 \RangeCatcodeCheck{0}{36}{15}%
788 \RangeCatcodeCheck{37}{37}{14}%
789 \RangeCatcodeCheck{38}{47}{15}%
790 \RangeCatcodeCheck{48}{57}{12}%
791 \RangeCatcodeCheck{58}{63}{15}%
792 \RangeCatcodeCheck{64}{64}{12}%
793 \RangeCatcodeCheck{65}{90}{11}%
794 \RangeCatcodeCheck{91}{91}{15}%
795 \RangeCatcodeCheck{92}{92}{0}%
796 \RangeCatcodeCheck{93}{96}{15}%
797 \RangeCatcodeCheck{97}{122}{11}%
798 \RangeCatcodeCheck{123}{255}{15}%
799 \RestoreCatcodes
800 }
801 \Test
802 \csname @@end\endcsname
803 \end
804 </test1>

```

3.2 Test \ltx@GobbleNum

```

805 <*test-gobble>
806 \catcode`\{=1 %
807 \catcode`\}=2 %
808 \catcode`\#=6 %
809 \expandafter\ifx\csname RequirePackage\endcsname\relax
810 \input ltxcmds.sty\relax
811 \else
812 \RequirePackage{ltxcmds}[2016/05/16]%
813 \fi
814 \catcode`\@=11 %
815 \def\msg#{\immediate\write16}%
816 \msg{[Test \string\ltx@GobbleNum]}%
817 \long\def\Test#1=#2\{\%
818 \edef\StrA{\ltx@GobbleNum#1}%
819 \expandafter\expandafter\expandafter\def
820 \expandafter\expandafter\expandafter\StrAA
821 \expandafter\expandafter\expandafter{\ltx@GobbleNum#1}%
822 \edef\StrB{#2}%
823 \ifx\StrA\StrB
824 \ifx\StrAA\StrB
825 \msg{* ok.}%
826 \else
827 \msg{StrAA: \StrAA}%
828 \msg{StrB: \StrB}%
829 \errhelp{Test: #1=#2}%
830 \errmessage{Test (two expansions) failed}%
831 \fi
832 \else
833 \msg{StrA: \StrA}%
834 \msg{StrB: \StrB}%
835 \errhelp{Test: #1=#2}%
836 \errmessage{Test (edef) failed!}%

```

```

837 \fi
838 }
839 \Test0abc=abc\\
840 \Test1abc=bc\\
841 \Test2abc=c\\
842 \Test3abcd=d\\
843 \Test4abcde=e\\
844 \Test5abcdef=f\\
845 \Test6abcdefg=g\\
846 \Test7abcdefgh=h\\
847 \Test8abcdefghi=i\\
848 \Test9abcdefghij=j\\
849 \Test{10}0123456789X=X\\
850 \Test{12}abcdefghijkmlm=m\\
851 \Test{700}%
852 0123456789012345678901234567890123456789012345678901234567890123456789%
853 0123456789012345678901234567890123456789012345678901234567890123456789%
854 0123456789012345678901234567890123456789012345678901234567890123456789%
855 0123456789012345678901234567890123456789012345678901234567890123456789%
856 0123456789012345678901234567890123456789012345678901234567890123456789%
857 0123456789012345678901234567890123456789012345678901234567890123456789%
858 0123456789012345678901234567890123456789012345678901234567890123456789%
859 0123456789012345678901234567890123456789012345678901234567890123456789%
860 0123456789012345678901234567890123456789012345678901234567890123456789%
861 0123456789012345678901234567890123456789012345678901234567890123456789%
862 X=X\\
863 \Test{-1}abc=abc\\
864 \Test2\par\par\relax=\relax\\
865
866 \begingroup
867   \count1=2 %
868   \Test{\count1}abc=c\\%
869 \endgroup
870
871 \ltx@ifundefined{numexpr}{%
872 }{%
873   \Test{1+1}abc=c\\%
874 }
875
876 \msg{[Test \string\ltx@CdrNum]}%
877 \long\def\Test#1=#2\\{%
878   \edef\StrA{\ltx@CdrNum#1\@nil}%
879   \expandafter\expandafter\expandafter\def
880     \expandafter\expandafter\expandafter\StrAA
881     \expandafter\expandafter\expandafter{\ltx@CdrNum#1\@nil}%
882   \edef\StrB{#2}%
883   \ifx\StrA\StrB
884     \ifx\StrAA\StrB
885       \msg{* ok.}%
886     \else
887       \msg{StrAA: \meaning\StrAA}%
888       \msg{StrB: \meaning\StrB}%
889       \errhelp{Test: #1=#2}%
890       \errmessage{Test (two expansions) failed}%
891   \fi
892 \else
893   \msg{StrA: \StrA}%
894   \msg{StrB: \StrB}%

```

```

895 \errhelp{Test: #1=#2}%
896 \errmessage{Test (edef) failed!}%
897 \fi
898 }
899 \Test0abc=abc\\
900 \Test1abc=bc\\
901 \Test2abc=c\\
902 \Test3abcd=d\\
903 \Test4abcde=e\\
904 \Test5abcdef=f\\
905 \Test6abcdefg=g\\
906 \Test7abcdefgh=h\\
907 \Test8abcdefghi=i\\
908 \Test9abcdefghij=j\\
909 \Test{10}0123456789X=X\\
910 \Test{12}abcdefghijklm=m\\
911 \Test{700}%
912 012345678901234567890123456789012345678901234567890123456789%
913 012345678901234567890123456789012345678901234567890123456789%
914 012345678901234567890123456789012345678901234567890123456789%
915 012345678901234567890123456789012345678901234567890123456789%
916 012345678901234567890123456789012345678901234567890123456789%
917 012345678901234567890123456789012345678901234567890123456789%
918 012345678901234567890123456789012345678901234567890123456789%
919 012345678901234567890123456789012345678901234567890123456789%
920 012345678901234567890123456789012345678901234567890123456789%
921 012345678901234567890123456789012345678901234567890123456789%
922 X=X\\
923 \Test{-1}abc=abc\\
924 \Test2\par\par\relax=\relax\\
925
926 \msg{[Test \string\ltx@CarNum]}%
927 \long\def\Test#1=#2\\{%
928 \edef\StrA{\ltx@CarNum#1\@nil}%
929 \expandafter\expandafter\expandafter\def
930 \expandafter\expandafter\expandafter\StrAA
931 \expandafter\expandafter\expandafter{\ltx@CarNum#1\@nil}%
932 \edef\StrB{#2}%
933 \ifx\StrA\StrB
934 \ifx\StrAA\StrB
935 \msg{* ok.}%
936 \else
937 \msg{StrAA: \meaning\StrAA}%
938 \msg{StrB: \meaning\StrB}%
939 \errhelp{Test: #1=#2}%
940 \errmessage{Test (two expansions) failed}%
941 \fi
942 \else
943 \msg{StrA: \StrA}%
944 \msg{StrB: \StrB}%
945 \errhelp{Test: #1=#2}%
946 \errmessage{Test (edef) failed!}%
947 \fi
948 }
949 \Test0abc=\\
950 \Test1abc=a\\
951 \Test2abc=ab\\
952 \Test3abc=abc\\

```



```

953 \Test3abcd=abc\\
954 \Test4abcde=abcd\\
955 \Test{10}0123456789X=0123456789\\
956 \Test{12}abcdefghijklm=abcdefghijklm\\
957 \Test{700}%
958 012345678901234567890123456789012345678901234567890123456789%
959 012345678901234567890123456789012345678901234567890123456789%
960 012345678901234567890123456789012345678901234567890123456789%
961 012345678901234567890123456789012345678901234567890123456789%
962 012345678901234567890123456789012345678901234567890123456789%
963 012345678901234567890123456789012345678901234567890123456789%
964 012345678901234567890123456789012345678901234567890123456789%
965 012345678901234567890123456789012345678901234567890123456789%
966 012345678901234567890123456789012345678901234567890123456789%
967 012345678901234567890123456789012345678901234567890123456789%
968 X=%
969 012345678901234567890123456789012345678901234567890123456789%
970 012345678901234567890123456789012345678901234567890123456789%
971 012345678901234567890123456789012345678901234567890123456789%
972 012345678901234567890123456789012345678901234567890123456789%
973 012345678901234567890123456789012345678901234567890123456789%
974 012345678901234567890123456789012345678901234567890123456789%
975 012345678901234567890123456789012345678901234567890123456789%
976 012345678901234567890123456789012345678901234567890123456789%
977 012345678901234567890123456789012345678901234567890123456789%
978 012345678901234567890123456789012345678901234567890123456789%
979 \\
980 \Test{-1}abc=\\
981 \Test2\par\par\relax=\par\par\\
982 \csname @@end\endcsname\end
983 \end{test-gobble}

```

3.3 Test \ltx@ifempty

```

984 (*test-ifempty)
985 \catcode`\{=1 %
986 \catcode`\}=2 %
987 \catcode`\#=6 %
988 \catcode`\@=11 %
989 \errorcontextlines=1000 %
990 \begingroup\expandafter\expandafter\expandafter\endgroup
991 \expandafter\ifx\csname RequirePackage\endcsname\relax
992 \input ltxcmds.sty\relax
993 \else
994 \RequirePackage{ltxcmds}[2016/05/16]%
995 \fi
996 \def\msg#{\immediate\write16}
997 \def\TestY{Y}
998 \def\TestN{N}
999 \msg{* \string\ltx@ifempty}
1000 \long\def\test#1{%
1001 \begingroup
1002 % Calculate expected test result via macro definition
1003 \def\Stuff{#1}%
1004 \ifx\Stuff\ltx@ifempty
1005 \def\StuffEmpty{Y}%
1006 \else
1007 \def\StuffEmpty{N}%
1008 \fi

```

```

1009 % Test \ltx@ifempty
1010 \expandafter\expandafter\expandafter\def
1011 \expandafter\expandafter\expandafter\TestEmpty
1012 \expandafter\expandafter\expandafter{%
1013 \ltx@ifempty{#1}{\Y}{\N}%
1014 }%
1015 \ifx\StuffEmpty\TestEmpty
1016 \msg{* Test OK}%
1017 \else
1018 \ltx@ifundefined{detokenize}{}{%
1019 \msg{Stuff: [\detokenize{\Stuff}]}%
1020 }%
1021 \errmessage{Test failed!}%
1022 \fi
1023 \endgroup
1024 }
1025 \test{}
1026 \test{a}
1027 \test{abc}
1028 \test{\par}
1029 \test{ }
1030 \test{\if}
1031 \test{{\if}}
1032 \test{\else}
1033 \test{{\else}}
1034 \test{\fi}
1035 \test{{}\fi}
1036 \test{\or\ifcase}
1037 \test{{}}
1038 \test{{a}}
1039 \test{{}abc}
1040 \test{{\par}}
1041 \test{{}\par}

1042 \def\SpaceTwo#1{%
1043 \def\SpaceTwo{#1#1}%
1044 }\SpaceTwo{ }
1045 \msg{* \string\ltx@ifblank}
1046 \long\def\test#1{%
1047 \begingroup
1048 % Calculate expected test result via macro definition
1049 \def\Stuff{#1}%
1050 \ifx\Stuff\ltx@empty
1051 \def\StuffEmpty{\Y}%
1052 \else
1053 \ifx\Stuff\ltx@space
1054 \def\StuffEmpty{\Y}%
1055 \else
1056 \ifx\Stuff\SpaceTwo
1057 \def\StuffEmpty{\Y}%
1058 \else
1059 \def\StuffEmpty{\N}%
1060 \fi
1061 \fi
1062 \fi
1063 % Test \ltx@ifblank
1064 \expandafter\expandafter\expandafter\def
1065 \expandafter\expandafter\expandafter\TestEmpty
1066 \expandafter\expandafter\expandafter{%

```

```

1067 \ltx@ifblank{#1}{\Y}{\N}%
1068 }%
1069 \ifx\StuffEmpty\TestEmpty
1070 \msg{* Test OK}%
1071 \else
1072 \ltx@ifundefined{detokenize}{}{%
1073 \msg{Stuff: [\detokenize{\Stuff}]]}%
1074 }%
1075 \errmessage{Test failed!}%
1076 \fi
1077 \endgroup
1078 }
1079 \test{}
1080 \test{a}
1081 \test{\if}
1082 \test{\else}
1083 \test{\fi}
1084 \test{ \fi}
1085 \test{\par}
1086 \test{ \par}
1087 \test{ {} }
1088 \test{ {} }
1089 \def\x#1{%
1090 \test{#1#1}%
1091 \test{#1#1{}}%
1092 \test{#1#1\par}%
1093 \test{#1#1\else}%
1094 } \x{ }
1095 \csname @@end\endcsname\end
1096 /test-ifempty>

```

3.4 Test \ltx@zap@space

```

1097 <*test-zap@space>
1098 \catcode`\{=1 %
1099 \catcode`\}=2 %
1100 \catcode`\#=6 %
1101 \catcode`\@=11 %
1102 \errorcontextlines=1000 %
1103 \begingroup\expandafter\expandafter\expandafter\endgroup
1104 \expandafter\ifx\csname RequirePackage\endcsname\relax
1105 \input ltxcmds.sty\relax
1106 \else
1107 \RequirePackage{ltxcmds}[2016/05/16]%
1108 \fi
1109 \def\msg#{\immediate\write16}
1110 \def\space{ }
1111 \def\empty{}
1112 \msg{* \string\ltx@zap@space}
1113 \long\def\test#1#2{%
1114 \begingroup
1115 \def\TestInput{#1}%
1116 \def\TestExpected{#2}%
1117 % Test \ltx@zap@space
1118 \expandafter\expandafter\expandafter\def
1119 \expandafter\expandafter\expandafter\TestResult
1120 \expandafter\expandafter\expandafter{%
1121 \ltx@zap@space{#1}%
1122 }%

```

```

1123 \ifx\TestResult\TestExpected
1124 \msg{* Test OK}%
1125 \else
1126 \ltx@onelevel@sanitize\TestInput
1127 \ltx@onelevel@sanitize\TestExpected
1128 \ltx@onelevel@sanitize\TestResult
1129 \msg{* Input: \space\space\space[\TestInput]]}%
1130 \msg{ \space Result: \space\space[\TestResult]]}%
1131 \msg{ \space Expected: [\TestExpected]]}%
1132 \errmessage{Test failed!}%
1133 \fi
1134 \endgroup
1135 }
1136 \long\def\etest#1#2{%
1137 \begingroup
1138 \edef\x{\endgroup
1139 \noexpand\test{#1}{#2}%
1140 }%
1141 \x
1142 }
1143 \catcode`\~13 %
1144 \let~\noexpand

1145 \test{}{}
1146 \test{{}}{{}}
1147 \test{ {} }{{}}
1148 \test{{ } }{{ }}
1149 \test{{} }{{}}
1150 \test{ {} }{{}}
1151 \test{ { } }{{ }}
1152 \test{a {b} c}{a{b}c}
1153 \test{a bb ccc}{abbccc}
1154 \test{{a} {bb} {ccc}}{{a}{bb}{ccc}}
1155 \test{\par}{\par}
1156 \test{\if}{\if}
1157 \test{\space}{\space}
1158 \etest{\par\space\par}{\par\par}
1159 \etest{~\empty\space~\empty}{~\empty~\empty}
1160 \etest{~\fi\space~\else\space}{~\fi~\else}

1161 \csname @@end\endcsname\end
1162 /test-zapspac

```

3.5 Test \ltx@ifBoxEmpty

```

1163 \def\test-ifboxempty{
1164 \catcode`\{=1 %
1165 \catcode`\}=2 %
1166 \catcode`\#=6 %
1167 \catcode`\@=11 %
1168 \begingroup\expandafter\expandafter\expandafter\endgroup
1169 \expandafter\ifx\csname RequirePackage\endcsname\relax
1170 \input ltxcmds.sty\relax
1171 \else
1172 \RequirePackage{ltxcmds}[2016/05/16]%
1173 \fi
1174 \def\msg#{\immediate\write16}
1175 % make box 0 void
1176 \begingroup
1177 \setbox0=\box0 %

```

```

1178 \endgroup
1179 \ifvoid0 %
1180 \else
1181   \errmessage{Voiding box 0 failed}%
1182 \fi
1183 \setbox2=\box0 %
1184 \def\test#1#2{%
1185   \@test{#1}{#2}%
1186   @@test{#1}{#2}%
1187   \chardef\x=#1%
1188   \@test\x{#2}%
1189   @@test\x{#2}%
1190 }
1191 \def\@test#1#2{%
1192   \begingroup
1193     \setbox9=\hbox{%
1194       \def\TestExpected{#2}%
1195       \ltx@ifBoxEmpty{#1}{%
1196         \def\TestResult{Y}%
1197       }{%
1198         \def\TestResult{N}%
1199       }%
1200       \ifx\TestExpected\TestResult
1201         \msg{* Test passed.}%
1202       \else
1203         \errmessage{Test failed!}%
1204       \fi
1205     }%
1206     \ifdim\wd9=0pt %
1207     \else
1208       \errmessage{Unwanted space?}%
1209     \fi
1210   \endgroup
1211 }
1212 \def\@@test#1#2{%
1213   \begingroup
1214     \setbox9=\hbox{%
1215       \def\TestExpected{#2}%
1216       \ifvoid#1\def\TestExpected{Y}\fi
1217       \ltx@ifBoxVoidOrEmpty{#1}{%
1218         \def\TestResult{Y}%
1219       }{%
1220         \def\TestResult{N}%
1221       }%
1222       \ifx\TestExpected\TestResult
1223         \msg{* Test passed.}%
1224       \else
1225         \errmessage{Test failed!}%
1226       \fi
1227     }%
1228     \ifdim\wd9=0pt %
1229     \else
1230       \errmessage{Unwanted space?}%
1231     \fi
1232   \endgroup
1233 }
1234 \testON
1235 \test2N

```

```

1236 \setbox0=\hbox{}
1237 \test0Y
1238 \setbox2=\hbox{}
1239 \test2Y
1240 \setbox0=\vbox{}
1241 \test0Y
1242 \setbox2=\vbox{}
1243 \test0Y
1244 \setbox0=\hbox{ }%
1245 \test0N
1246 \setbox2=\hbox{ }%
1247 \test2N
1248 \setbox0=\hbox{\penalty1}%
1249 \test0N
1250 \setbox2=\hbox{\penalty1}%
1251 \test2N
1252 \csname @@end\endcsname\end
1253 /test-ifboxempty>

```

3.6 Test for next character detection

```

1254 (*test-nextchar)
1255 \catcode`\{=1 %
1256 \catcode`\}=2 %
1257 \catcode`\#=6 %
1258 \catcode`\@=11 %
1259 \begingroup\expandafter\expandafter\expandafter\endgroup
1260 \expandafter\ifx\csname RequirePackage\endcsname\relax
1261   \input ltxcmds.sty\relax
1262   \input eolgrab.sty\relax
1263 \else
1264   \RequirePackage{ltxcmds}[2016/05/16]%
1265   \RequirePackage{eolgrab}[2011/01/12]%
1266 \fi
1267 \def\msg#{\immediate\write16}
1268 \begingroup
1269   \def\x#1{%
1270     \endgroup
1271     \let\TestSpaceToken= #1\relax
1272   }%
1273 \x{ }
1274 \def\TestSpace{ }
1275 \begingroup
1276   \lccode32=65 % space -> A
1277 \lowercase{%
1278   \endgroup
1279   \def\TestSpaceA{ }%
1280 }
1281 \def\TestCatch{%
1282   \eolgrab\@TestCatch
1283 }
1284 \def\@TestCatch#1{%
1285   \begingroup
1286     \def\x{#1}%
1287     \ifx\x\ltx@empty
1288     \else
1289       \ltx@onelevel@sanitize\x
1290       \errmessage{Unparsed stuff on line [\x]}%
1291     \fi

```

```

1292 \endgroup
1293 }
1294 \def\TestCmdM#1{%
1295   \TestCheckType{M}%
1296   \TestCatch
1297 }
1298 \def\TestCmdOM[#1]#2{%
1299   \TestCheckType{O}%
1300   \TestCatch
1301 }
1302 \def\TestCheckType#1{%
1303   \if\TestCmdType#1\relax
1304   \else
1305     \errmessage{Wrong type #1, expected: \TestCmdType}%
1306   \fi
1307 }
1308 \def\TestCmd#1{%
1309   \def\TestCmdType{#1}%
1310   \ltx@ifnextchar[\TestCmdOM\TestCmdM
1311 }
1312 \def\TestCmdExp#1{%
1313   \expandafter\TestCmd\expandafter#1%
1314 }
1315 \outer\def\TestOuter{}
1316 \TestCmd O[o]{m}
1317 \TestCmd M{m}
1318 \TestCmd O [o]{m}
1319 \TestCmd M {m}
1320 \def\x#1{\def\x{#1}}\x{ }
1321 \TestCmdExp O\x[o]{m}
1322 \TestCmdExp M\x{m}
1323 \def\x#1{\def\x{#1#1#1}}\x{ }
1324 \TestCmdExp O\x[o]{m}
1325 \TestCmdExp M\x{m}
1326 \def\x{\TestSpaceToken}
1327 \TestCmdExp O\x[o]{m}
1328 \TestCmdExp M\x{m}
1329 \def\x{\TestSpaceToken\TestSpaceToken\TestSpaceToken}
1330 \TestCmdExp O\x[o]{m}
1331 \TestCmdExp M\x{m}
1332 \TestCmd M\TestSpace
1333 \TestOuter
1334 \TestCmd M \TestSpace
1335 \TestOuter
1336 \TestCmd M\iftrue
1337 \TestOuter
1338 \TestCmd M\iffalse
1339 \TestOuter
1340 \TestCmd M\else
1341 \TestOuter
1342 \TestCmd M\fi
1343 \TestOuter
1344 \TestCmd M \iftrue
1345 \TestOuter
1346 \TestCmd M \iffalse
1347 \TestOuter
1348 \TestCmd M \else
1349 \TestOuter

```

```

1350 %
1351 \def\TestCmd#1{%
1352   \def\TestCmdType{#1}%
1353   \ltx@ifnextchar@nospace[\TestCmdOM\TestCmdM
1354 }
1355 \TestCmd O[o]{m}
1356 \TestCmd M{m}
1357 \TestCmd M [
1358 \TestOuter
1359 \TestCmd M {m}
1360 \TestCmd M\iftrue
1361 \TestOuter
1362 \TestCmd M\iffalse
1363 \TestOuter
1364 \TestCmd M\else
1365 \TestCmd M\fi
1366 \TestOuter
1367 \TestOuter
1368 %
1369 \def\TestCmd#1{%
1370   \def\TestCmdType{#1}%
1371   \ltx@ifnextchar(\TestCmdPM\TestCmdM
1372 }
1373 \def\TestCmdPM(#1)#2{%
1374   \TestCheckType{P}%
1375   \TestCatch
1376 }
1377 \TestCmd P(p){m}
1378 \TestCmd M{m}
1379 \TestCmd P (p){m}
1380 \TestCmd M {m}
1381 %
1382 \def\TestCmd#1{%
1383   \def\TestCmdType{#1}%
1384   \ltx@ifnextchar{ }\TestCmdSM\TestCmdM
1385 }
1386 \def\TestCmdSM#1#{%
1387   \TestCheckType{S}%
1388   \begingroup
1389     \let\x= #1\relax
1390     \ifx\x\TestSpaceToken
1391       \else
1392         \errmessage{unexpected space token: \meaning#1}%
1393       \fi
1394   \endgroup
1395   \def\TestCmdType{M}%
1396   \TestCmdM
1397 }
1398 \TestCmd S {m}
1399 \TestCmd M{m}
1400 \def\x#1{\def\x{#1}}\x{ }
1401 \TestCmdExp S\x{m}
1402 %
1403 \def\TestCmd#1{%
1404   \def\TestCmdType{#1}%
1405   \ltx@ifnextchar\iffalse\TestCmdIM\TestCmdM
1406 }
1407 \def\TestCmdIM\iffalse#1{%

```



```

1408 \TestCheckType{I}%
1409 \TestCatch
1410 }
1411 \TestCmd M\iftrue
1412 \TestOuter
1413 \TestCmd M \iftrue
1414 \TestCmd I\iffalse\iffalse
1415 \TestCmd I \iffalse\iffalse
1416 \TestOuter
1417 %
1418 \def\TestCmd#1{%
1419 \def\TestCmdType{#1}%
1420 \ltx@ifnextchar@nospace\iffalse\TestCmdIM\TestCmdM
1421 }
1422 \TestCmd M\iftrue
1423 \TestOuter
1424 \TestCmd I\iffalse\iffalse
1425 \TestOuter
1426 \csname @@end\endcsname\end
1427 /test-nextchar)

```

3.7 Test for list helpers

```

1428 (*test-carcdr)
1429 \catcode`\{=1 %
1430 \catcode`\}=2 %
1431 \catcode`\#=6 %
1432 \catcode`\@=11 %
1433 \begingroup\expandafter\expandafter\expandafter\endgroup
1434 \expandafter\ifx\csname RequirePackage\endcsname\relax
1435 \input ltxcmds.sty\relax
1436 \input eolgrab.sty\relax
1437 \else
1438 \RequirePackage{ltxcmds}[2016/05/16]%
1439 \RequirePackage{eolgrab}[2011/01/12]%
1440 \fi
1441 \def\msg#{\immediate\write16}
1442 \def\space{ }
1443 \long\def\Test#1#2#3{%
1444 \begingroup
1445 \def\TestExpected{#3}%
1446 \expandafter\expandafter\expandafter\def
1447 \expandafter\expandafter\expandafter\TestResult
1448 \expandafter\expandafter\expandafter{%
1449 #1#2\@nil
1450 }%
1451 \ifx\TestResult\TestExpected
1452 \else
1453 \msg{\string\TestExpected: [\meaning\TestExpected]}%
1454 \msg{\string\TestResult: \space\space[\meaning\TestResult]}%
1455 \errmessage{Test failed!}%
1456 \fi
1457 \endgroup
1458 }
1459 \Test\ltx@carzero{abc}{ }
1460 \Test\ltx@carzero{ }{ }
1461 \Test\ltx@carzero{\par\par}{ }
1462 \Test\ltx@cdrzero{ }{ }
1463 \Test\ltx@cdrzero{abc}{abc}

```

```

1464 \Test\ltx@cdrzero{ \par}{ \par}
1465 \Test\ltx@cdrzero{\@empty}{\@empty}
1466 \Test\ltx@cdrzero{}}{}
1467 \Test\ltx@car{abc}{a}
1468 \Test\ltx@car{ \par}{ \par}
1469 \Test\ltx@cdr{abc}{bc}
1470 \Test\ltx@cdr{a \par}{ \par}
1471 \Test\ltx@cdr{a \@empty}{\@empty}
1472 \Test\ltx@cartwo{abc}{ab}
1473 \Test\ltx@cartwo{ \par \@empty}{ \par \@empty}
1474 \Test\ltx@carsecond{abc}{b}
1475 \Test\ltx@carsecond{\@empty b \@empty}{b}
1476 \Test\ltx@carsecond{ \par \par \par}{ \par}
1477 \Test\ltx@cdrtwo{abc}{c}
1478 \Test\ltx@cdrtwo{ab \par}{ \par}
1479 \Test\ltx@cdrtwo{ab \@empty}{\@empty}
1480 \Test\ltx@cdrtwo{ab{}}{}
1481 \Test\ltx@cdrthree{abcdefg}{defg}
1482 \Test\ltx@cdrfour{abcdefg}{efg}
1483 \Test{\ltx@CdrNum{5}}{abcdefg}{fg}
1484 \Test{\ltx@CdrNum{0}}{ \par}{ \par}
1485 \Test{\ltx@CdrNum{0}}{ \@empty}{ \@empty}
1486 \Test{\ltx@CdrNum{0}}{ }{ }
1487 \Test{\ltx@CdrNum{0}}{ }{ }
1488 \Test{\ltx@CdrNum{2}}{abcd}{cd}
1489 \Test{\ltx@CdrNum{2}}{ \vbox \par \hbox \par}{ \hbox \par}
1490 \Test{\ltx@carthree}{abcdefg}{abc}
1491 \Test{\ltx@carfour}{abcdefg}{abcd}
1492 \Test{\ltx@CarNum{5}}{abcdefg}{abcde}
1493 \Test{\ltx@CarNum{2}}{ \@empty \par}{ \@empty \par}
1494 \Test\ltx@carthird{abcdefg}{c}
1495 \Test\ltx@carfourth{abcdefg}{d}
1496 \Test{\ltx@CarNumth{5}}{abcdefg}{e}
1497 \Test{\ltx@CarNumth{2}}{ \@empty \@empty \par}{ \@empty}
1498 \Test{\ltx@CarNumth{2}}{ \par \par \par}{ \par}
1499 \Test{\ltx@CarNumth{2}}{ab}{b}
1500 \csname @@end\endcsname\end
1501 \end{test-carcdr}

```

4 Installation

4.1 Download

Package. This package is available on CTAN¹:

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

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

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

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

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

¹[CTAN:pkg/ltxcmds](#)

4.2 Bundle installation

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

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

Script installation. Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps.

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):

<code>ltxcmds.sty</code>	\rightarrow <code>tex/generic/oberdiek/ltxcmds.sty</code>
<code>ltxcmds.pdf</code>	\rightarrow <code>doc/latex/oberdiek/ltxcmds.pdf</code>
<code>test/ltxcmds-test1.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/ltxcmds-test1.tex</code>
<code>test/ltxcmds-test-gobble.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/ltxcmds-test-gobble.tex</code>
<code>test/ltxcmds-test-ifempty.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/ltxcmds-test-ifempty.tex</code>
<code>test/ltxcmds-test-zapspace.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/ltxcmds-test-zapspace.tex</code>
<code>test/ltxcmds-test-ifboxempty.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/ltxcmds-test-ifboxempty.tex</code>
<code>test/ltxcmds-test-nextchar.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/ltxcmds-test-nextchar.tex</code>
<code>test/ltxcmds-test-carcdr.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/ltxcmds-test-carcdr.tex</code>
<code>ltxcmds.dtx</code>	\rightarrow <code>source/latex/oberdiek/ltxcmds.dtx</code>

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

4.4 Refresh file name databases

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

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 pdfL^AT_EX:

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

5 References

- [1] Robert R. Schneck: *Re: \ifempty solution (was Macro puzzle: maximally general \ifempty)*; newsgroup `comp.text.tex`, `news:3eef1ada_6@corp.newsgroups.com`, 2003-06-17.
<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$1@news.albasani.net`, 2010-11-12.
<https://groups.google.com/group/comp.text.tex/msg/803bd57221a04996>
- [3] Ahmed Musa: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:f5496afe-40ed-42bd-b629-a2419ecf7c0d@o14g2000prn.googlegroups.com`, 2010-12-03.
<https://groups.google.com/group/comp.text.tex/msg/fbf7d61a0c3a807d>
- [4] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idbo94uka1@four.albasani.net`, 2010-12-03.
<https://groups.google.com/group/comp.text.tex/msg/0c230ee479487962>
- [5] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idbpu4cgi1@news.albasani.net`, 2010-12-03.
<https://groups.google.com/group/comp.text.tex/msg/bbef4263390d647b>
- [6] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup `comp.text.tex`, `news:idd4ga$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>

6 History

[2009/08/05 v1.0]

- First version.

[2009/12/12 v1.1]

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

[2010/01/28 v1.2]

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

[2010/03/01 v1.3]

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

[2010/03/09 v1.4]

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

[2010/04/08 v1.5]

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

[2010/04/16 v1.6]

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

[2010/04/26 v1.7]

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

[2010/09/11 v1.8]

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

[2010/10/25 v1.9]

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

[2010/10/31 v1.10]

- `\ltx@newglobalif` added.

[2010/11/12 v1.11]

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

[2010/12/02 v1.12]

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

[2010/12/04 v1.13]

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

[2010/12/07 v1.14]

- `\ltx@zapspace` added.

[2010/12/12 v1.15]

- `\ltx@minusone` added.

[2011/02/04 v1.16]

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

[2011/02/05 v1.17]

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

[2011/03/16 v1.18]

- `\ltx@ifclasslater` fixed.

[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.

7 Index

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

Symbols			
<code>\#</code>	254, 709, 808, 987, 1100, 1166, 1257, 1431	<code>\}</code> 264, 708, 807, 986, 1099, 1165, 1256, 1430	
<code>\\$</code>	652, 655	<code>\~</code> 1143	
<code>\%</code>	244, 785	A	
<code>\&</code>	653, 656	<code>\advance</code>	748, 756, 771
<code>\.</code>	517	<code>\aftergroup</code>	29
<code>\@</code>	710, 783, 814, 988, 1101, 1167, 1258, 1432	B	
<code>\@test</code>	1186, 1189, 1212	<code>\body</code>	727, 731
<code>\@TestCatch</code>	1282, 1284	<code>\box</code>	543, 554, 1177, 1183
<code>\@empty</code>	1465, 1471, 1473, 1475, 1479, 1485, 1493, 1497	C	
<code>\@firstofone</code>	718, 721	<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, 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	
<code>\@gobble</code>	715, 723	<code>\chardef</code> .	116, 117, 118, 119, 120, 1187
<code>\@nil</code> ..	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	<code>\count</code>	867, 868
<code>\@test</code>	1185, 1188, 1191	<code>\count@</code>	712, 741, 745, 747, 748, 752, 754, 755, 756, 760, 762, 765, 766, 770, 771
<code>\@undefined</code>	58	<code>\countdef</code>	712
<code>\</code>	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	<code>\csname</code>	14, 21, 50, 66, 76, 160, 165, 206, 211, 278,
<code>\{</code>	259, 707, 806, 985, 1098, 1164, 1255, 1429		

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@gttemp 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@ccclv 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 6, 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

<code>\ltx@firstoftwo</code>	171 , 323 , 331 , 344 , 349 , 367 , 404 , 598 , 614 , 669 , 688	<code>\ltx@LocSkipA</code>	144
<code>\ltx@fourthoffour</code>	179	<code>\ltx@LocSkipB</code>	145
<code>\ltx@GlobalAppendToMacro</code>	8 , 436	<code>\ltx@LocSkipC</code>	146
<code>\ltx@GlobalPrependToMacro</code>	8 , 464	<code>\ltx@LocSkipD</code>	147
<code>\ltx@GlobDimenA</code>	139	<code>\ltx@LocSkipE</code>	148
<code>\ltx@GlobDimenB</code>	140	<code>\ltx@LocToksA</code>	124 , 445 , 446 , 459 , 460 , 473 , 475 , 488 , 490 , 497 , 503 , 523 , 530
<code>\ltx@GlobDimenC</code>	141	<code>\ltx@LocToksB</code>	125 , 474 , 475 , 489 , 490 , 498 , 509 , 524 , 532
<code>\ltx@GlobDimenD</code>	142	<code>\ltx@LocToksC</code>	126
<code>\ltx@GlobDimenE</code>	143	<code>\ltx@LocToksD</code>	127
<code>\ltx@GlobSkipA</code>	149	<code>\ltx@LocToksE</code>	128
<code>\ltx@GlobSkipB</code>	150	<code>\ltx@mbox</code>	9 , 568
<code>\ltx@GlobSkipC</code>	151	<code>\ltx@minusone</code>	121
<code>\ltx@GlobSkipD</code>	152	<code>\ltx@newglobalif</code>	6 , 290
<code>\ltx@GlobSkipE</code>	153	<code>\ltx@newif</code>	6 , 268
<code>\ltx@GlobToksA</code>	129	<code>\ltx@one</code>	117 , 122 , 675 , 679
<code>\ltx@GlobToksB</code>	130	<code>\ltx@onelevel@sanitize</code>	7 , 358 , 1126 , 1127 , 1128 , 1289
<code>\ltx@GlobToksC</code>	131	<code>\ltx@percentchar</code>	243
<code>\ltx@GlobToksD</code>	132	<code>\ltx@pkgextension</code>	365 , 373 , 411
<code>\ltx@GlobToksE</code>	133	<code>\ltx@RemovePrefix</code>	7 , 354 , 356 , 361
<code>\ltx@gobble</code>	4 , 154 , 336 , 590 , 592 , 603 , 606 , 608 , 634	<code>\ltx@ReturnAfterElseFi</code>	240
<code>\ltx@gobblefour</code>	157	<code>\ltx@ReturnAfterFi</code>	5 , 239
<code>\ltx@GobbleNum</code>	4 , 158 , 227 , 237 , 816 , 818 , 821	<code>\ltx@rightbracechar</code>	263
<code>\ltx@gobblethree</code>	156	<code>\ltx@secondoffour</code>	177
<code>\ltx@gobbletwo</code>	155	<code>\ltx@secondofthree</code>	174
<code>\ltx@hashchar</code>	253	<code>\ltx@secondoftwo</code>	172 , 325 , 333 , 346 , 367 , 387 , 402 , 596 , 612 , 660 , 671 , 690 , 693
<code>\ltx@ifblank</code>	9 , 600 , 630 , 1045 , 1063 , 1067	<code>\ltx@space</code>	6 , 242 , 401 , 1053
<code>\ltx@ifBoxEmpty</code>	10 , 658 , 702 , 1195	<code>\ltx@StripPrefix</code>	355 , 428 , 429 , 430
<code>\ltx@ifBoxVoidOrEmpty</code>	10 , 698 , 1217	<code>\ltx@thirdoffour</code>	178 , 700
<code>\ltx@ifclasslater</code>	8 , 407	<code>\ltx@thirdofthree</code>	175
<code>\ltx@ifclassloaded</code>	7 , 369	<code>\ltx@two</code>	118 , 683 , 687
<code>\ltx@ifempty</code>	9 , 586 , 619 , 645 , 999 , 1009 , 1013	<code>\ltx@undefined</code>	437 , 451 , 465 , 480
<code>\ltx@iffilelater</code>	375 , 408 , 411	<code>\ltx@zapspace</code>	9 , 640 , 1112 , 1117 , 1121
<code>\ltx@iffileloaded</code>	7 , 366 , 370 , 373 , 376	<code>\ltx@zero</code>	3 , 116 , 183 , 187 , 192 , 197 , 202 , 220 , 230 , 236 , 541 , 543 , 545 , 642 , 663 , 668 , 674 , 682
<code>\ltx@ifnextchar</code>	8 , 494 , 1310 , 1371 , 1384 , 1405	<code>\LTXcmds@@ifnextchar</code>	507 , 513
<code>\ltx@ifnextchar@nospace</code>	9 , 520 , 1353 , 1420	<code>\LTXcmds@@ParseVersion</code>	415 , 417
<code>\ltx@ifpackagelater</code>	410	<code>\LTXcmds@AtEnd</code>	95 , 96 , 115 , 704
<code>\ltx@ifpackageloaded</code>	372	<code>\LTXcmds@CarNum</code>	207 , 210
<code>\ltx@ifUndefined</code>	6 , 328 , 352 , 413 , 535 , 536 , 537 , 538 , 572 , 585 , 651 , 871 , 1018 , 1072	<code>\LTXcmds@CarNumFinish</code>	219
<code>\ltx@ifundefined</code>	6 , 321 , 341 , 352 , 367	<code>\LTXcmds@CarNumth</code>	226 , 229
<code>\ltx@leavevmode</code>	9 , 535 , 569	<code>\LTXcmds@cdrzero</code>	181 , 183 , 187 , 192 , 197 , 202
<code>\ltx@leftbracechar</code>	258	<code>\LTXcmds@CharToken</code>	496 , 502 , 522 , 529
<code>\ltx@LocalAppendToMacro</code>	449	<code>\LTXcmds@Cm</code>	213
<code>\ltx@LocalExpandAfter</code>	6 , 313 , 319	<code>\LTXcmds@Cx</code>	216
<code>\ltx@LocalPrependToMacro</code>	478	<code>\LTXcmds@Gm</code>	167
<code>\ltx@LocDimenA</code>	134	<code>\LTXcmds@GobbleNum</code>	161 , 164
<code>\ltx@LocDimenB</code>	135	<code>\LTXcmds@gtemp</code>	450 , 451 , 452 , 454 , 459 , 460 , 462 , 479 , 480 , 481 , 483 , 489 , 490 , 492
<code>\ltx@LocDimenC</code>	136	<code>\LTXcmds@ifempty</code>	629
<code>\ltx@LocDimenD</code>	137		
<code>\ltx@LocDimenE</code>	138		

<code>\LTxcmds@ifempty@</code>	628	<code>\RestoreCatcodes</code> . .	740, 743, 744, 799
<code>\LTxcmds@iflater</code>	377, 389	<code>\romannumeral</code> . .	159, 162, 183, 187,
<code>\LTxcmds@ifnextchar</code> . . .	499, 501, 516		192, 197, 202, 205, 208, 224,
<code>\LTxcmds@ifnextchar@nospace</code>	525, 527		234, 517, 588, 601, 620, 631, 641
<code>\LTxcmds@lettoken</code>		S	
	499, 502, 515, 525, 529	<code>\setbox</code>	541, 543,
<code>\LTxcmds@newglobalif</code>	294, 296		552, 554, 663, 674, 682, 1177,
<code>\LTxcmds@newif</code>	272, 274		1183, 1193, 1214, 1236, 1238,
<code>\LTxcmds@num</code>	162, 208, 572		1240, 1242, 1244, 1246, 1248, 1250
<code>\LTxcmds@parseversion</code>		<code>\skipdef</code>	144, 145, 146,
	379, 385, 414, 421		147, 148, 149, 150, 151, 152, 153
<code>\LTxcmds@spacetoken</code>	506, 519	<code>\space</code>	765, 766,
<code>\LTxcmds@temp</code>	586, 617		774, 1110, 1129, 1130, 1131,
<code>\LTxcmds@voidbox</code> 549, 550, 552, 554, 557			1157, 1158, 1159, 1160, 1442, 1454
<code>\LTxcmds@zapspace</code>	642, 644	<code>\SpaceTwo</code>	1042, 1043, 1044, 1056
M		<code>\StrA</code>	818, 823,
<code>\meaning</code>	361,		833, 878, 883, 893, 928, 933, 943
	887, 888, 937, 938, 1392, 1453, 1454	<code>\StrAA</code>	820, 824,
<code>\msg</code>	815, 816, 825, 827, 828,		827, 880, 884, 887, 930, 934, 937
	833, 834, 876, 885, 887, 888,	<code>\StrB</code>	822, 823,
	893, 894, 926, 935, 937, 938,		824, 828, 834, 882, 883, 884,
	943, 944, 996, 999, 1016, 1019,		888, 894, 932, 933, 934, 938, 944
	1045, 1070, 1073, 1109, 1112,	<code>\Stuff</code>	1003, 1004,
	1124, 1129, 1130, 1131, 1174,		1019, 1049, 1050, 1053, 1056, 1073
	1201, 1223, 1267, 1441, 1453, 1454	<code>\StuffEmpty</code>	1005, 1007,
N			1015, 1051, 1054, 1057, 1059, 1069
<code>\N</code>	998, 1007, 1013, 1059, 1067	T	
<code>\next</code> . .	688, 690, 693, 695, 732, 734, 736	<code>\Test</code> 778, 801, 817, 839, 840, 841, 842,	
<code>\number</code>	378, 384, 575, 767		843, 844, 845, 846, 847, 848,
<code>\numexpr</code>	581		849, 850, 851, 863, 864, 868,
O			873, 877, 899, 900, 901, 902,
<code>\outer</code>	1315		903, 904, 905, 906, 907, 908,
P			909, 910, 911, 923, 924, 927,
<code>\PackageInfo</code>	26		949, 950, 951, 952, 953, 954,
<code>\par</code> 864, 924, 981, 1028, 1040, 1041,			955, 956, 957, 980, 981, 1443,
	1085, 1086, 1092, 1155, 1158,		1459, 1460, 1461, 1462, 1463,
	1461, 1464, 1468, 1470, 1473,		1464, 1465, 1466, 1467, 1468,
	1476, 1478, 1484, 1489, 1493, 1498		1469, 1470, 1471, 1472, 1473,
<code>\pdflastmatch</code>	428, 429, 430		1474, 1475, 1476, 1477, 1478,
<code>\pdfmatch</code>	422		1479, 1480, 1481, 1482, 1483,
<code>\penalty</code>	675, 683, 1248, 1250		1484, 1485, 1486, 1487, 1488,
<code>\ProvidesPackage</code>	19, 67		1489, 1490, 1491, 1492, 1493,
Q			1494, 1495, 1496, 1497, 1498, 1499
<code>\quitvmode</code>	566	<code>\test</code>	1000, 1025, 1026,
R			1027, 1028, 1029, 1030, 1031,
<code>\RangeCatcodeCheck</code>			1032, 1033, 1034, 1035, 1036,
	759, 787, 788, 789, 790, 791,		1037, 1038, 1039, 1040, 1041,
	792, 793, 794, 795, 796, 797, 798		1046, 1079, 1080, 1081, 1082,
<code>\RangeCatcodeInvalid</code>			1083, 1084, 1085, 1086, 1087,
	751, 779, 780, 781, 782		1088, 1090, 1091, 1092, 1093,
<code>\repeat</code>	726, 738, 749, 757, 772		1113, 1139, 1145, 1146, 1147,
<code>\RequirePackage</code>	812, 994,		1148, 1149, 1150, 1151, 1152,
	1107, 1172, 1264, 1265, 1438, 1439		1153, 1154, 1155, 1156, 1157,
			1184, 1234, 1235, 1237, 1239,
			1241, 1243, 1245, 1247, 1249, 1251
		<code>\TestCatch</code> 1281, 1296, 1300, 1375, 1409	

<code>\TestCheckType</code>	<code>\TestSpaceToken</code> 1271, 1326, 1329, 1390
1295, 1299, 1302, 1374, 1387, 1408	<code>\TestY</code> 997
<code>\TestCmd</code> . . . 1308, 1313, 1316, 1317,	<code>\the</code> 77, 78, 79, 80, 81, 82,
1318, 1319, 1332, 1334, 1336,	83, 84, 97, 446, 460, 475, 490,
1338, 1340, 1342, 1344, 1346,	503, 509, 528, 581, 745, 765, 766
1348, 1351, 1355, 1356, 1357,	<code>\TMP@EnsureCode</code> 94, 101,
1359, 1360, 1362, 1364, 1365,	102, 103, 104, 105, 106, 107,
1369, 1377, 1378, 1379, 1380,	108, 109, 110, 111, 112, 113, 114
1382, 1398, 1399, 1403, 1411,	<code>\toksdef</code> 124, 125, 126,
1413, 1414, 1415, 1418, 1422, 1424	127, 128, 129, 130, 131, 132, 133
<code>\TestCmdExp</code> 1312, 1321, 1322, 1324,	
1325, 1327, 1328, 1330, 1331, 1401	U
<code>\TestCmdIM</code> 1405, 1407, 1420	<code>\unhbox</code> 545, 557, 560
<code>\TestCmdM</code> 1294, 1310,	<code>\unhcopy</code> 664, 676, 684
1353, 1371, 1384, 1396, 1405, 1420	<code>\unvcopy</code> 664, 676, 684
<code>\TestCmdOM</code> 1298, 1310, 1353	
<code>\TestCmdPM</code> 1371, 1373	V
<code>\TestCmdSM</code> 1384, 1386	<code>\vbox</code> . . . 663, 674, 682, 1240, 1242, 1489
<code>\TestCmdType</code> . . . 1303, 1305, 1309,	<code>\voidb@x</code> 560
1352, 1370, 1383, 1395, 1404, 1419	
<code>\TestEmpty</code> 1011, 1015, 1065, 1069	W
<code>\TestExpected</code> 1116,	<code>\wd</code> 1206, 1228
1123, 1127, 1131, 1194, 1200,	<code>\write</code> 23,
1215, 1216, 1222, 1445, 1451, 1453	52, 815, 996, 1109, 1174, 1267, 1441
<code>\TestInput</code> 1115, 1126, 1129	
<code>\TestN</code> 998	X
<code>\TestOuter</code> 1315, 1333, 1335,	<code>\x</code> 14, 15, 18, 22, 26, 28, 51,
1337, 1339, 1341, 1343, 1345,	56, 66, 75, 87, 1089, 1094, 1138,
1347, 1349, 1358, 1361, 1363,	1141, 1187, 1188, 1189, 1269,
1366, 1367, 1412, 1416, 1423, 1425	1273, 1286, 1287, 1289, 1290,
<code>\TestResult</code> 1119, 1123,	1320, 1321, 1322, 1323, 1324,
1128, 1130, 1196, 1198, 1200,	1325, 1326, 1327, 1328, 1329,
1218, 1220, 1222, 1447, 1451, 1454	1330, 1331, 1389, 1390, 1400, 1401
<code>\TestSpace</code> 1274, 1332, 1334	
<code>\TestSpaceA</code> 1279	Y
	<code>\Y</code> 997, 1005, 1013, 1051, 1054, 1057, 1067