The ltxcmds package

Heiko Oberdiek*

2020-05-10 v1.25

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

---

[*] Please report any issues at `https://github.com/ho-tex/ltxcmds/issues`

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 following 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 2020-05-10 v1.25:

- 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

```
⟨*package⟩
```

Reload check, especially if the package is not used with LaTeX.

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode35=6 % #
  \catcode39=12 % ’
  \catcode44=12 % ,
  \catcode45=12 % -
  \catcode46=12 % .
  \catcode58=12 % :
  \catcode64=11 % @
  \catcode123=1 % {
  \catcode125=2 % }
  \expandafter\let\expandafter\x\csname ver@ltxcmds.sty\endcsname
  \ifx\x\relax % plain-TeX, first loading
  \else
    \def\empty{}%
    \ifx\x\empty % LaTeX, first loading,
      % variable is initialized, but \ProvidesPackage not yet seen
    \else
      \expandafter\ifx\csname PackageInfo\endcsname\relax
        \def\x#1#2{%
          \immediate\write-1{Package #1 Info: #2.}%
        }%
      \else
        \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
      \fi
      \x{ltxcmds}{The package is already loaded}%
      \aftergroup\endinput
    \fi
  \fi
\endgroup%
```

Package identification:

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode35=6 % #
  \catcode39=12 % '
  \catcode40=12 % (
  \catcode41=12 % )
  \catcode44=12 % ,
  \catcode45=12 % -
  \catcode46=12 % .
  \catcode47=12 % /
  \catcode58=12 % :
  \catcode64=11 % @
  \catcode91=12 % [
  \catcode93=12 % ]
  \catcode123=1 % {
  \catcode125=2 % }
  \expandafter\ifx\csname ProvidesPackage\endcsname\relax
    \def\x#1#2#3[#4]{\endgroup
      \immediate\write-1{Package: #3 #4}%
      \xdef#1{#4}%
    }%
  \else
    \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}%
  [2020-05-10 v1.25 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@one`

```
\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

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

\ltx@LocDimenE

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

\ltx@GlobDimenA

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

\ltx@GlobDimenB

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

\ltx@GlobDimenC

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

\ltx@GlobDimenD

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

\ltx@GlobDimenE

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

\ltx@LocSkipA

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

`\ltx@LocSkipB`

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

`\ltx@LocSkipC`

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

`\ltx@LocSkipD`

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

`\ltx@LocSkipE`

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

`\ltx@GlobSkipA`

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

`\ltx@GlobSkipB`

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

`\ltx@GlobSkipC`

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

`\ltx@GlobSkipD`

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

`\ltx@GlobSkipE`

```
\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

changed 2020-05-10 to adapt to dates with dashes (ISO) The core of the commands are copies from the latex commands.

`\ltx@ifl@ter`
`\ltx@parse@version@`

```
\def\ltx@ifl@ter#1#2{%
  \expandafter\ltx@ifl@t@r
    \csname ver@#2.#1\endcsname}
\def\ltx@ifl@t@r#1#2{%
  \ifnum\expandafter\ltx@parse@version@#1//00\@nil<%
        \expandafter\ltx@parse@version@#2//00\@nil
    \expandafter\@secondoftwo
  \else
    \expandafter\@firstoftwo
  \fi}
\def\ltx@parse@version@#1{\ltx@parse@version0#1}
\def\ltx@parse@version#1/#2/#3#4#5\@nil{%
\ltx@parse@version@dash#1-#2-#3#4\@nil
}
\def\ltx@parse@version@dash#1-#2-#3#4#5\@nil{%
  \if\relax#2\relax\else#1\fi#2#3#4 }
```

`\ltx@iffilelater`

```
\def\ltx@iffilelater#1{\expandafter\ltx@ifl@t@r\csname ver@#1\endcsname}
```

`\ltx@ifclasslater`

```
\def\ltx@ifclasslater{\ltx@ifl@ter\ltx@clsextension}
```

`\ltx@ifpackagelater`

```
\def\ltx@ifpackagelater{\ltx@ifl@ter\ltx@pkgextension}
```

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 Installation

3.1 Download

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

CTAN:macros/latex/contrib/ltxcmds/ltxcmds.dtx The source file.

CTAN:macros/latex/contrib/ltxcmds/ltxcmds.pdf Documentation.

3.2 Package installation

The package is at best installed with the package manager of the TEX system. Manual installation is possible too:

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

```
tex ltxcmds.dtx
```

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

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

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

3.3 Refresh file name databases

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

3.4 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.

---

[1] CTAN:pkg/ltxcmds

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
```

4 References

[1] Robert R. Schneck: *Re: \ifempty solution (was Macro puzzle: maximally general \ifempty)*; newsgroup comp.text.tex, news:3eef1ada_6@corp.newsgroups.com, 2003-06-17.
https://groups.google.com/group/comp.text.tex/msg/be03a159ec374895

[2] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup comp.text.tex, news:ibk3t8$ee7$1@news.albasani.net, 2010-11-12.
https://groups.google.com/group/comp.text.tex/msg/803bd57221a04996

[3] Ahmed Musa: *Re: TeX refuses to strip outer braces in argument*; newsgroup comp.text.tex, news:f5496afe-40ed-42bd-b629-a2419ecf7c0d@o14g2000prn.googlegroups.com, 2010-12-03.
https://groups.google.com/group/comp.text.tex/msg/fbf7d61a0c3a807d

[4] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup comp.text.tex, news:idbo94$uka$1@four.albasani.net, 2010-12-03.
https://groups.google.com/group/comp.text.tex/msg/0c230ee479487962

[5] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup comp.text.tex, news:idbpu4$cg1$1@news.albasani.net, 2010-12-03.
https://groups.google.com/group/comp.text.tex/msg/bbef4263390d647b

[6] Ulrich Diez: *Re: TeX refuses to strip outer braces in argument*; newsgroup comp.text.tex, news:idd4ga$r83$1@four.albasani.net, 2010-12-04.
https://groups.google.com/group/comp.text.tex/msg/00dfd1ec103cd272

[7] GL: *Re: TeX refuses to strip outer braces in argument*; newsgroup comp.text.tex, news:4cfa2e27$0$7389$426a74cc@news.free.fr, 2010-12-04.
https://groups.google.com/group/comp.text.tex/msg/d3a75995c1cf267e

[8] Heiko Oberdiek: *Re: TeX refuses to strip outer braces in argument*; newsgroup comp.text.tex, news:iddhq1$3kj$1@news.eternal-september.org, 2010-12-04.
https://groups.google.com/group/comp.text.tex/msg/5f7a23e3ab70e347

[9] David Kastrup: *How to detect if \vbox is empty*; newsgroup comp.text.tex, 2011-02-04.
https://groups.google.com/group/comp.text.tex/msg/8d3cb89496a4d86d

5 History

[2009/08/05 v1.0]

- First version.

[2009/12/12 v1.1]

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

[2010/01/28 v1.2]

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

[2010/03/01 v1.3]

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

[2010/03/09 v1.4]

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

[2010/04/08 v1.5]

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

[2010/04/16 v1.6]

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

[2010/04/26 v1.7]

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

[2010/09/11 v1.8]

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

[2010/10/25 v1.9]

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

[2010/10/31 v1.10]

- `\ltx@newglobalif` added.

[2010/11/12 v1.11]

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

[2010/12/02 v1.12]

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

[2010/12/04 v1.13]

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

[2010/12/07 v1.14]

- `\ltx@zapspace` added.

[2010/12/12 v1.15]

- `\ltx@minusone` added.

[2011/02/04 v1.16]

- `\ltx@IfBoxEmpty` and `\ltx@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.

[2019/12/15 v1.24]

- Documentation updates.

[2020-05-10 v1.25]

- Changed the definitions of `\ltx@iffilelater`, `\ltx@ifpackagelater` and `\ltx@ifclasslater` to support dates in ISO format in same way as the LaTeX kernel does it since 2017. The commands now use the same test as the LaTeX kernel. `\pdfmatch` is no longer used with pdftex, and the tests for dates before 1994 have been removed

6 Index

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