

The ltxcmds package

Heiko Oberdiek
<heiko.oberdiek at googlemail.com>

2010/04/08 v1.5

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	2
1.1	Introduction	2
1.2	Numbers	2
1.3	Argument killers	2
1.4	Argument grabbers	3
1.5	List helpers	3
1.6	Tail recursion	3
1.7	Empty macro	3
1.8	Characters	3
1.9	Boolean switch	4
1.10	Command definitions	4
1.11	Stripping	4
1.12	File management	4
	1.12.1 File extensions	4
	1.12.2 Load check	5
	1.12.3 Version date check	5
1.13	Macro additions	5
1.14	Macro \ltx@ifnextchar	5
2	Implementation	6
2.1	Identification	6
2.2	Numbers	7
2.3	Argument killers	7
2.4	Argument grabbers	8
2.5	List helpers	8
2.6	Tail recursion	8
2.7	Empty macro	8
2.8	Characters	9
2.9	Boolean switch	9
2.10	Command definitions	9
2.11	Stripping	10
2.12	File management	10
	2.12.1 File extensions	10
	2.12.2 Load check	11
	2.12.3 Version date check	11
2.13	Macro additions	12
2.14	Macro \ltx@ifnextchar	13

3	Test	13
3.1	Catcode checks for loading	13
4	Installation	15
4.1	Download	15
4.2	Bundle installation	15
4.3	Package installation	15
4.4	Refresh file name databases	16
4.5	Some details for the interested	16
5	History	16
[2009/08/05 v1.0]	16
[2009/12/12 v1.1]	16
[2010/01/28 v1.2]	17
[2010/03/01 v1.3]	17
[2010/03/09 v1.4]	17
[2010/04/08 v1.5]	17
6	Index	17

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
\ltx@one
\ltx@two
\ltx@cclv

These commands are numbers 0, 1, 2 and 255. They are not digits and a space is not gobbled afterwards.

1.3 Argument killers

$\text{\ltx@gobble} \{ \langle 1 \rangle \}$	\rightarrow
$\text{\ltx@gobbletwo} \{ \langle 1 \rangle \} \{ \langle 2 \rangle \}$	\rightarrow
$\text{\ltx@gobblethree} \{ \langle 1 \rangle \} \{ \langle 2 \rangle \} \{ \langle 3 \rangle \}$	\rightarrow
$\text{\ltx@gobblefour} \{ \langle 1 \rangle \} \{ \langle 2 \rangle \} \{ \langle 3 \rangle \} \{ \langle 4 \rangle \}$	\rightarrow

1.4 Argument grabbers

<code>\ltx@firstofone {⟨1⟩}</code>	\rightarrow	$\langle 1 \rangle$
<code>\ltx@firstoftwo {⟨1⟩} {⟨2⟩}</code>	\rightarrow	$\langle 1 \rangle$
<code>\ltx@secondoftwo {⟨1⟩} {⟨2⟩}</code>	\rightarrow	$\langle 2 \rangle$

1.5 List helpers

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

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

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

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

1.6 Tail recursion

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

1.7 Empty macro

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

1.8 Characters

<code>\ltx@space</code>
<code>\ltx@percentchar</code>
<code>\ltx@backslashchar</code>

1.9 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`.

1.10 Command definitions

`\ltx@ifundefined {<cmd>} {<yes>} {<no>}`

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

`\ltx@ifundefined {<cmd>} {<yes>} {<no>}`

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

`\ltx@LocalExpandAfter`

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

1.11 Stripping

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

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

1.12 File management

All macros in this section are expandable like the counterparts of the \LaTeX kernel. Also they can be used after the preamble.

1.12.1 File extensions

`\ltx@clsextension`
`\ltx@pkgextension`

If `\@clsextension/\@pkgextension` exists then `\ltx@clsextension/\ltx@pkgextension` returns this macro, otherwise the result is `cls/sty`.

1.12.2 Load check

<code>\ltx@ifclassloaded {⟨class⟩} {⟨yes⟩} {⟨no⟩}</code> <code>\ltx@ifpackageloaded {⟨package⟩} {⟨yes⟩} {⟨no⟩}</code>
--

If the $\langle class \rangle / \langle package \rangle$ are loaded the macros `\ltx@ifclassloaded`/`\ltx@ifpackageloaded` call the $\langle yes \rangle$ argument. Otherwise $\langle no \rangle$ is executed. Both $\langle class \rangle$ and $\langle package \rangle$ are specified without extension.

<code>\ltx@iffileloaded {⟨file⟩} {⟨yes⟩} {⟨no⟩}</code>
--

If L^AT_EX's `\ProvidesFile` macro was called before using $\langle file \rangle$ as argument, then `\ltx@iffileloaded` calls $\langle yes \rangle$, otherwise $\langle no \rangle$. Therefore it is possible that the $\langle file \rangle$ is loaded, but $\langle no \rangle$ 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.

1.12.3 Version date check

<code>\ltx@ifclasslater {⟨class⟩} {⟨date⟩} {⟨yes⟩} {⟨no⟩}</code> <code>\ltx@ifpackagelater {⟨package⟩} {⟨date⟩} {⟨yes⟩} {⟨no⟩}</code> <code>\ltx@iffilelater {⟨file⟩} {⟨date⟩} {⟨yes⟩} {⟨no⟩}</code>
--

If a `\ProvidesClass`/`\ProvidesPackage`/`\ProvidesFile` command with exact the same class/package/file was executed before with an optional argument that starts with a L^AT_EX version date, then this version date is compared with the argument $\langle date \rangle$. If they are equal or if the version date is the later date, then $\langle yes \rangle$ is called. In all other cases $\langle no \rangle$ is executed.

A L^AT_EX date has the format YYYY/MM/DD with YYYY as year with four digits, MM as month with two digits and DD as day with two digits. If pdfT_EX's `\pdfmatch` is available, then it is used to detect the version date, to reject invalid date formats and to reject some invalid dates. Dates before 1994/01/01 are always invalid, because version dates are introduced with L^AT_EX 2_ε in 1994.

1.13 Macro additions

<code>\ltx@GlobalAppendToMacro {⟨cmd⟩} {⟨addition⟩}</code> <code>\ltx@LocalAppendToMacro {⟨cmd⟩} {⟨addition⟩}</code>

The $\langle addition \rangle$ is appended to the parameterless macro $\langle cmd \rangle$. If $\langle cmd \rangle$ is undefined or has the meaning `\relax`, then it will be initialized as empty macro before.

1.14 Macro `\ltx@ifnextchar`

<code>\ltx@ifnextchar {⟨char⟩} {⟨yes⟩} {⟨no⟩}</code>
--

If next character is $\langle char \rangle$ then $\langle yes \rangle$ is called, otherwise $\langle no \rangle$. The character is not removed.

2 Implementation

2.1 Identification

```
1 (*package)
```

Reload check, especially if the package is not used with L^AT_EX.

```
2 \begingroup
3  \catcode44 12 % ,
4  \catcode45 12 % -
5  \catcode46 12 % .
6  \catcode58 12 % :
7  \catcode64 11 % @
8  \catcode123 1 % {
9  \catcode125 2 % }
10 \expandafter\let\expandafter\x\csname ver@ltxcmds.sty\endcsname
11 \ifx\x\relax % plain-TeX, first loading
12 \else
13   \def\empty{}%
14   \ifx\x\empty % LaTeX, first loading,
15     % variable is initialized, but \ProvidesPackage not yet seen
16   \else
17     \catcode35 6 % #
18     \expandafter\ifx\csname PackageInfo\endcsname\relax
19       \def\x#1#2{%
20         \immediate\write-1{Package #1 Info: #2.}%
21       }%
22     \else
23       \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
24     \fi
25     \x{ltxcmds}{The package is already loaded}%
26   \aftergroup\endinput
27 \fi
28 \fi
29 \endgroup
```

Package identification:

```
30 \begingroup
31  \catcode35 6 % #
32  \catcode40 12 % (
33  \catcode41 12 % )
34  \catcode44 12 % ,
35  \catcode45 12 % -
36  \catcode46 12 % .
37  \catcode47 12 % /
38  \catcode58 12 % :
39  \catcode64 11 % @
40  \catcode91 12 % [
41  \catcode93 12 % ]
42  \catcode123 1 % {
43  \catcode125 2 % }
44  \expandafter\ifx\csname ProvidesPackage\endcsname\relax
45    \def\x#1#2#3[#4]{\endgroup
46      \immediate\write-1{Package: #3 #4}%
47      \xdef#1{#4}%
48    }%
49  \else
50    \def\x#1#2[#3]{\endgroup
51      #2[#{#3}]%
52      \ifx#1\@undefined
53        \xdef#1{#3}%
54      \fi
55      \ifx#1\relax
56        \xdef#1{#3}%
57      \fi
58    }%
59  \fi
60 \endgroup
```

```

57      \fi
58    }%
59  \fi
60 \expandafter\x\csname ver@ltxcmds.sty\endcsname
61 \ProvidesPackage{ltxcmds}%
62 [2010/04/08 v1.5 LaTeX kernel commands for general use (H0)]
63 \begingroup
64   \catcode123 1 % {
65   \catcode125 2 % }
66   \def\x{\endgroup
67     \expandafter\edef\csname LTXcmds@AtEnd\endcsname{%
68       \catcode35 \the\catcode35\relax
69       \catcode64 \the\catcode64\relax
70       \catcode123 \the\catcode123\relax
71       \catcode125 \the\catcode125\relax
72     }%
73   }%
74 \x
75 \catcode35 6 % #
76 \catcode64 11 % @
77 \catcode123 1 % {
78 \catcode125 2 % }
79 \def\TMP@EnsureCode#1#2{%
80   \edef\LTXcmds@AtEnd{%
81     \LTXcmds@AtEnd
82     \catcode#1 \the\catcode#1\relax
83   }%
84   \catcode#1 #2\relax
85 }
86 \TMP@EnsureCode{40}{12}% (
87 \TMP@EnsureCode{41}{12}% )
88 \TMP@EnsureCode{45}{12}% -
89 \TMP@EnsureCode{46}{12}% .
90 \TMP@EnsureCode{47}{12}% /
91 \TMP@EnsureCode{60}{12}% <
92 \TMP@EnsureCode{61}{12}% =
93 \TMP@EnsureCode{62}{12}% >
94 \TMP@EnsureCode{91}{12}% [
95 \TMP@EnsureCode{96}{12}% '
96 \TMP@EnsureCode{93}{12}% ]
97 \TMP@EnsureCode{94}{12}% ^ (superscript) (!)
98 \TMP@EnsureCode{124}{12}% |

```

2.2 Numbers

```

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

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

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

\ltx@cc1v
102 \chardef\ltx@cc1v=255 %

```

2.3 Argument killers

```

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

```

`\ltx@gobbletwo`

104 `\long\def\ltx@gobbletwo#1#2{}`

`\ltx@gobblethree`

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

`\ltx@gobblefour`

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

2.4 Argument grabbers

`\ltx@firstofone`

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

`\ltx@firstoftwo`

108 `\long\def\ltx@firstoftwo#1#2{#1}`

`\ltx@secondoftwo`

109 `\long\def\ltx@secondoftwo#1#2{#2}`

2.5 List helpers

`\ltx@car`

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

`\ltx@cdr`

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

`\ltx@cartwo`

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

`\ltx@cdrtwo`

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

`\ltx@carthree`

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

`\ltx@cdrthree`

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

`\ltx@carfour`

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

`\ltx@cdrfour`

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

2.6 Tail recursion

`\ltx@ReturnAfterFi`

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

`\ltx@ReturnAfterElseFi`

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

2.7 Empty macro

`\ltx@empty`

120 `\def\ltx@empty{}`

2.8 Characters

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

\ltx@percentchar
122 \begingroup
123   \lccode'0='%\relax
124 \lowercase{\endgroup
125   \def\ltx@percentchar{0}%
126 }

\ltx@backslashchar
127 \begingroup
128   \lccode'0='\\relax
129 \lowercase{\endgroup
130   \def\ltx@backslashchar{0}%
131 }
```

2.9 Boolean switch

```
\ltx@newif
132 \def\ltx@newif#1{%
133   \begingroup
134     \escapechar=-1 %
135   \expandafter\endgroup
136   \expandafter\LTXcmds@newif\string#1\@nil
137 }

\LTXcmds@newif
138 \begingroup
139   \escapechar=-1 %
140 \expandafter\endgroup
141 \expandafter\def\expandafter\LTXcmds@newif\string\if#1\@nil{%
142   \expandafter\edef\csname#1true\endcsname{%
143     \let
144     \expandafter\noexpand\csname if#1\endcsname
145     \noexpand\iftrue
146   }%
147   \expandafter\edef\csname#1false\endcsname{%
148     \let
149     \expandafter\noexpand\csname if#1\endcsname
150     \noexpand\iffalse
151   }%
152   \csname#1false\endcsname
153 }
```

2.10 Command definitions

```
\ltx@LocalExpandAfter
154 \def\ltx@LocalExpandAfter{%
155   \begingroup
156     \expandafter\expandafter\expandafter
157   \endgroup
158   \expandafter
159 }

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

```

\ltx@ifundefined
162 \def\ltx@ifundefined#1{%
163 \expandafter\ifx\csname #1\endcsname\relax
164 \expandafter\ltx@firstoftwo
165 \else
166 \expandafter\ltx@secondoftwo
167 \fi
168 }%

\ltx@ifUndefined
169 \def\ltx@ifUndefined#1{%
170 \begingroup\expandafter\expandafter\expandafter\endgroup
171 \expandafter\ifx\csname #1\endcsname\relax
172 \expandafter\ltx@firstoftwo
173 \else
174 \expandafter\ltx@secondoftwo
175 \fi
176 }%

177 \expandafter\ltx@gobble
178 \else
179 \expandafter\ltx@firstofone
180 \fi
181 {%

\ltx@ifundefined
182 \def\ltx@ifundefined#1{%
183 \ifcsname #1\endcsname
184 \expandafter\ifx\csname #1\endcsname\relax
185 \expandafter\expandafter\expandafter\ltx@firstoftwo
186 \else
187 \expandafter\expandafter\expandafter\ltx@secondoftwo
188 \fi
189 \else
190 \expandafter\ltx@firstoftwo
191 \fi
192 }%

\ltx@ifUndefined
193 \let\ltx@ifUndefined\ltx@ifundefined
194 }

```

2.11 Stripping

```

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

\ltx@StripPrefix
196 \def\ltx@StripPrefix{%
197 \expandafter\ltx@RemovePrefix
198 }

```

2.12 File management

2.12.1 File extensions

```

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

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

```

2.12.2 Load check

`\ltx@iffileloaded`

```
201 \def\ltx@iffileloaded#1{%  
202   \ltx@ifundefined{ver@#1}\ltx@secondoftwo\ltx@firstoftwo  
203 }
```

`\ltx@ifclassloaded`

```
204 \def\ltx@ifclassloaded#1{%  
205   \ltx@iffileloaded{#1.\ltx@clsextension}%  
206 }
```

`\ltx@ifpackageloaded`

```
207 \def\ltx@ifpackageloaded#1{%  
208   \ltx@iffileloaded{#1.\ltx@pkgextension}%  
209 }
```

2.12.3 Version date check

`\ltx@iffilelater`

```
210 \def\ltx@iffilelater#1#2{%  
211   \ltx@iffileloaded{#1}{%  
212     \expandafter\LTXcmds@IfLater\expandafter{%  
213       \number  
214       \expandafter\expandafter\expandafter\LTXcmds@ParseVersion  
215       \expandafter\expandafter\expandafter{%  
216         \csname ver@#1\endcsname  
217       }%  
218     \expandafter}\expandafter{%  
219       \number  
220       \expandafter\LTXcmds@ParseVersion\expandafter{#2}%  
221     }%  
222   }\ltx@secondoftwo  
223 }
```

`\LTXcmds@IfLater`

```
224 \def\LTXcmds@IfLater#1#2{%  
225   \ifcase 0%  
226     \ifnum#1<19940101 %  
227     \else  
228       \ifnum#2<19940101 %  
229       \else  
230         \ifnum#2>#1 %  
231         \else  
232           1%  
233         \fi  
234       \fi  
235     \fi  
236     \ltx@space  
237     \expandafter\ltx@secondoftwo  
238   \else  
239     \expandafter\ltx@firstoftwo  
240   \fi  
241 }
```

`\ltx@ifclasslater`

```
242 \def\ltx@ifclasslater#1{%  
243   \ltx@ifclasslater{#1.\ltx@clsextension}%  
244 }
```

```

\ltx@ifpackagelater
245 \def\ltx@ifpackagelater#1{%
246   \ltx@iffilelater{#1.\ltx@pkgextension}%
247 }

248 \ltx@ifundefined{pdfmatch}{%

\LTxcmds@ParseVersion
249 \def\LTxcmds@ParseVersion#1{%
250   \LTxcmds@@ParseVersion#10000/00/00\@nil
251 }%

\LTxcmds@@ParseVersion
252 \def\LTxcmds@@ParseVersion#1#2#3#4/#5#6/#7#8#9\@nil{%
253   #1#2#3#4#5#6#7#8%
254 }%

255 }{%

\LTxcmds@ParseVersion
256 \def\LTxcmds@ParseVersion#1{%
257   \ifnum\pdfmatch{%
258     ~%
259     (199[4-9] | [2-9] [0-9] [0-9] [0-9])/%
260     (0[1-9] | 1[0-2])/%
261     (0[1-9] | [1-2] [0-9] | 3[0-1])%
262   }-#{1}=1 %
263   \ltx@StripPrefix\pdfmatch1 %
264   \ltx@StripPrefix\pdfmatch2 %
265   \ltx@StripPrefix\pdfmatch3 %
266   \else
267     0%
268   \fi
269 }%
270 }

```

2.13 Macro additions

```

\ltx@GlobalAppendToMacro
271 \def\ltx@GlobalAppendToMacro#1#2{%
272   \ifx\ltx@undefined#1%
273     \let#1\ltx@empty
274   \else
275     \ifx\relax#1%
276       \let#1\ltx@empty
277     \fi
278   \fi
279   \begingroup
280     \toks0\expandafter{#1#2}%
281     \xdef#1{\the\toks0}%
282   \endgroup
283 }

\ltx@LocalAppendToMacro
284 \def\ltx@LocalAppendToMacro#1#2{%
285   \global\let\LTxcmds@gtemp#1%
286   \ifx\ltx@undefined\LTxcmds@gtemp
287     \global\let\LTxcmds@gtemp\ltx@empty
288   \else
289     \ifx\relax\LTxcmds@gtemp
290       \global\let\LTxcmds@gtemp\ltx@empty

```

```

291   \fi
292 \fi
293 \begingroup
294   \toks0\expandafter{\LTXcmds@gtmp#2}%
295   \xdef\LTXcmds@gtmp{\the\toks0}%
296 \endgroup
297 \let#1\LTXcmds@gtmp
298 }

```

2.14 Macro \ltx@ifnextchar

\ltx@ifnextchar

```

299 \long\def\ltx@ifnextchar#1#2#3{%
300   \begingroup
301   \let\LTXcmds@CharToken= #1\relax
302   \toks\ltx@zero{#2}%
303   \toks\ltx@two{#3}%
304   \futurelet\LTXcmds@LetToken\LTXcmds@ifnextchar
305 }

```

\LTXcmds@ifnextchar

```

306 \def\LTXcmds@ifnextchar{%
307   \ifx\LTXcmds@LetToken\LTXcmds@CharToken
308     \expandafter\endgroup\the\toks\expandafter\ltx@zero
309   \else
310     \ifx\LTXcmds@LetToken\LTXcmds@SpaceToken
311       \expandafter\expandafter\expandafter\LTXcmds@@ifnextchar
312     \else
313       \expandafter\endgroup\the\toks
314       \expandafter\expandafter\expandafter\ltx@two
315     \fi
316   \fi
317 }

```

\LTXcmds@@ifnextchar

```

318 \begingroup
319   \def\x#1{\endgroup
320     \def\LTXcmds@@ifnextchar#1{%
321       \futurelet\LTXcmds@LetToken\LTXcmds@ifnextchar
322     }%
323   }%
324 \x{ }

```

\LTXcmds@SpaceToken

```

325 \begingroup
326   \def\x#1{\endgroup
327     \let\LTXcmds@SpaceToken= #1%
328   }%
329 \x{ }

330 \LTXcmds@AtEnd
331 \endpackage

```

3 Test

3.1 Catcode checks for loading

```

332 \test1

333 \catcode'\{=1 %
334 \catcode'\}=2 %

```

```

335 \catcode'\#=6 %
336 \catcode'\@=11 %
337 \expandafter\ifx\csname count@\endcsname\relax
338   \countdef\count@=255 %
339 \fi
340 \expandafter\ifx\csname @gobble\endcsname\relax
341   \long\def@gobble#1{%
342 \fi
343 \expandafter\ifx\csname @firstofone\endcsname\relax
344   \long\def\@firstofone#1{#1}%
345 \fi
346 \expandafter\ifx\csname loop\endcsname\relax
347   \expandafter\@firstofone
348 \else
349   \expandafter@gobble
350 \fi
351 {%
352   \def\loop#1\repeat{%
353     \def\body{#1}%
354     \iterate
355   }%
356   \def\iterate{%
357     \body
358     \let\next\iterate
359   \else
360     \let\next\relax
361   \fi
362   \next
363 }%
364 \let\repeat=\fi
365 }%
366 \def\RestoreCatcodes{
367 \count@=0 %
368 \loop
369   \edef\RestoreCatcodes{%
370     \RestoreCatcodes
371     \catcode\the\count@=\the\catcode\count@\relax
372   }%
373 \ifnum\count@<255 %
374   \advance\count@ 1 %
375 \repeat
376
377 \def\RangeCatcodeInvalid#1#2{%
378   \count@=#1\relax
379   \loop
380     \catcode\count@=15 %
381   \ifnum\count@<#2\relax
382     \advance\count@ 1 %
383   \repeat
384 }
385 \expandafter\ifx\csname LoadCommand\endcsname\relax
386   \def\LoadCommand{\input ltxcmds.sty\relax}%
387 \fi
388 \def\Test{%
389   \RangeCatcodeInvalid{0}{47}%
390   \RangeCatcodeInvalid{58}{64}%
391   \RangeCatcodeInvalid{91}{96}%
392   \RangeCatcodeInvalid{123}{255}%
393   \catcode'\@=12 %
394   \catcode'\=0 %
395   \catcode'\{=1 %
396   \catcode'\}=2 %

```

```

397 \catcode'\#=6 %
398 \catcode'\[=12 %
399 \catcode'\]=12 %
400 \catcode'\%=14 %
401 \catcode'\_ =10 %
402 \catcode13=5 %
403 \LoadCommand
404 \RestoreCatcodes
405 }
406 \Test
407 \csname @@end\endcsname
408 \end
409 </test1>

```

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 \TeX Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

4.2 Bundle installation

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

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

Script installation. Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package `attachfile2` comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```

chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/

```

4.3 Package installation

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

```
tex ltxcmds.dtx
```

¹<http://ftp.ctan.org/tex-archive/>

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

```
ltxcmds.sty      → tex/generic/oberdiek/ltxcmds.sty
ltxcmds.pdf      → doc/latex/oberdiek/ltxcmds.pdf
test/ltxcmds-test1.tex → doc/latex/oberdiek/test/ltxcmds-test1.tex
ltxcmds.dtx      → source/latex/oberdiek/ltxcmds.dtx
```

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

4.4 Refresh file name databases

If your $\text{T}_{\text{E}}\text{X}$ distribution (`te $\text{T}_{\text{E}}\text{X}$` , `mik $\text{T}_{\text{E}}\text{X}$` , ...) relies on file name databases, you must refresh these. For example, `te $\text{T}_{\text{E}}\text{X}$` users run `texhash` or `mktextlsr`.

4.5 Some details for the interested

Attached source. The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is `pdftk`, e.g. unpack the file into the current directory:

```
pdftk ltxcmds.pdf unpack_files output .
```

Unpacking with $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$. The `.dtx` chooses its action depending on the format:

plain $\text{T}_{\text{E}}\text{X}$: Run `docstrip` and extract the files.

$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$: Generate the documentation.

If you insist on using $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ for `docstrip` (really, `docstrip` does not need $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{ltxcmds.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

Generating the documentation. You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with `pdf $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$` :

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

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.

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; numbers in roman refer to the code lines where the entry is used.

Symbols		C	
<code>\#</code>	335, 397	<code>\catcode</code>	3, 4, 5, 6, 7, 8, 9, 17, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 64, 65, 68, 69, 70, 71, 75, 76, 77, 78, 82, 84, 333, 334, 335, 336, 371, 380, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402
<code>\%</code>	123, 400	<code>\chardef</code>	99, 100, 101, 102
<code>\@</code>	336, 393	<code>\count@</code>	338, 367, 371, 373, 374, 378, 380, 381, 382
<code>\@firstofone</code>	344, 347	<code>\countdef</code>	338
<code>\@gobble</code>	341, 349	<code>\csname</code>	10, 18, 44, 60, 67, 142, 144, 147, 149, 152, 161, 163, 171, 184, 216, 337, 340, 343, 346, 385, 407
<code>\@nil</code>	110, 111, 112, 113, 114, 115, 116, 117, 136, 141, 250, 252		
<code>\@undefined</code>	52		
<code>\[</code>	398		
<code>\]</code>	128, 394		
<code>\{</code>	333, 395		
<code>\}</code>	334, 396		
<code>\]</code>	399		
		E	
<code>_</code>	401	<code>\empty</code>	13, 14
		<code>\end</code>	408
A		<code>\endcsname</code>	10, 18, 44, 60, 67, 142, 144, 147, 149, 152, 161, 163, 171, 183, 184, 216, 337, 340, 343, 346, 385, 407
<code>\advance</code>	374, 382	<code>\endinput</code>	26
<code>\aftergroup</code>	26	<code>\escapechar</code>	134, 139
B			
<code>\body</code>	353, 357		

F	
<code>\futurelet</code>	304, 321
I	
<code>\if</code>	141
<code>\ifcase</code>	225
<code>\ifcsname</code>	183
<code>\iffalse</code>	150
<code>\ifnum</code>	226, 228, 230, 257, 373, 381
<code>\iftrue</code>	145
<code>\ifx</code> .	11, 14, 18, 44, 52, 55, 161, 163, 171, 184, 272, 275, 286, 289, 307, 310, 337, 340, 343, 346, 385
<code>\immediate</code>	20, 46
<code>\input</code>	386
<code>\iterate</code>	354, 356, 358
L	
<code>\lccode</code>	123, 128
<code>\letLTxcmds@gtemp</code>	290
<code>\LoadCommand</code>	386, 403
<code>\loop</code>	352, 368, 379
<code>\lowercase</code>	124, 129
<code>\ltx@backslashchar</code>	127
<code>\ltx@car</code>	3, 110
<code>\ltx@carfour</code>	3, 116
<code>\ltx@carthree</code>	3, 114
<code>\ltx@cartwo</code>	3, 112
<code>\ltx@cclv</code>	102
<code>\ltx@cdr</code>	111
<code>\ltx@cdrfour</code>	117
<code>\ltx@cdrthree</code>	115
<code>\ltx@cdrtwo</code>	113
<code>\ltx@clsextension</code> ...	4, 199, 205, 243
<code>\ltx@empty</code> ..	3, 120, 273, 276, 287, 290
<code>\ltx@firstofone</code>	3, 107, 179
<code>\ltx@firstoftwo</code> 108, 164, 172, 185, 190, 202, 239
<code>\ltx@GlobalAppendToMacro</code>	5, 271
<code>\ltx@gobble</code>	2, 103, 177
<code>\ltx@gobblefour</code>	106
<code>\ltx@gobblethree</code>	105
<code>\ltx@gobbletwo</code>	104
<code>\ltx@ifclasslater</code>	5, 242
<code>\ltx@ifclassloaded</code>	5, 204
<code>\ltx@iffilelater</code>	210, 246
<code>\ltx@iffileloaded</code> 5, 201, 205, 208, 211	
<code>\ltx@ifnextchar</code>	5, 299
<code>\ltx@ifpackagelater</code>	245
<code>\ltx@ifpackageloaded</code>	207
<code>\ltx@ifUndefined</code>	4, 169, 193, 248
<code>\ltx@ifundefined</code> 4, 162, 182, 193, 202	
<code>\ltx@LocalAppendToMacro</code>	284
<code>\ltx@LocalExpandAfter</code> ...	4, 154, 160
<code>\ltx@newif</code>	4, 132
<code>\ltx@one</code>	100
<code>\ltx@percentchar</code>	122
<code>\ltx@pkgextension</code>	200, 208, 246
<code>\ltx@RemovePrefix</code>	4, 195, 197
<code>\ltx@ReturnAfterElseFi</code>	119
<code>\ltx@ReturnAfterFi</code>	3, 118
<code>\ltx@secondoftwo</code> 109, 166, 174, 187, 202, 222, 237
<code>\ltx@space</code>	3, 121, 236
<code>\ltx@StripPrefix</code> ..	196, 263, 264, 265
<code>\ltx@two</code>	101, 303, 314
<code>\ltx@undefined</code>	272, 286
<code>\ltx@zero</code>	2, 99, 302, 308
<code>\LTxcmds@ifnextchar</code>	311, 318
<code>\LTxcmds@@ParseVersion</code>	250, 252
<code>\LTxcmds@AtEnd</code>	80, 81, 330
<code>\LTxcmds@CharToken</code>	301, 307
<code>\LTxcmds@gtemp</code> 285, 286, 287, 289, 294, 295, 297
<code>\LTxcmds@ifLater</code>	212, 224
<code>\LTxcmds@ifnextchar</code> ...	304, 306, 321
<code>\LTxcmds@LetToken</code> ..	304, 307, 310, 321
<code>\LTxcmds@newif</code>	136, 138
<code>\LTxcmds@ParseVersion</code> 214, 220, 249, 256
<code>\LTxcmds@SpaceToken</code>	310, 325
N	
<code>\next</code>	358, 360, 362
<code>\number</code>	213, 219
P	
<code>\PackageInfo</code>	23
<code>\pdflastmatch</code>	263, 264, 265
<code>\pdfmatch</code>	257
<code>\ProvidesPackage</code>	15, 61
R	
<code>\RangeCatcodeInvalid</code> 377, 389, 390, 391, 392
<code>\repeat</code>	352, 364, 375, 383
<code>\RestoreCatcodes</code> ..	366, 369, 370, 404
T	
<code>\Test</code>	388, 406
<code>\the</code>	68, 69, 70, 71, 82, 281, 295, 308, 313, 371
<code>\TMP@EnsureCode</code> ...	79, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98
<code>\toks</code>	280, 281, 294, 295, 302, 303, 308, 313
W	
<code>\write</code>	20, 46
X	
<code>\x</code>	10, 11, 14, 19, 23, 25, 45, 50, 60, 66, 74, 319, 324, 326, 329