

The pdfescape package

Heiko Oberdiek
<heiko.oberdiek at gmail.com>

2011/11/25 v1.13

Abstract

This package implements pdfTeX's escape features (`\pdfescapehex`, `\pdfunescapehex`, `\pdfescapeiname`, `\pdfescapestring`) using TeX or ϵ -TeX.

Contents

1	Documentation	2
1.1	Additional unescape macros	2
1.2	Sanitizing macro	3
2	Implementation	3
2.1	Reload check and package identification	3
2.2	Catcodes	4
2.3	Load package	5
2.4	Sanitizing	5
2.4.1	Space characters	6
2.4.2	Space normalization	6
2.5	<code>\EdefUnescapeName</code>	6
2.6	<code>\EdefUnescapeString</code>	8
2.7	User macros (pdfTeX analogues)	11
2.8	Help macros	12
2.8.1	Characters	12
2.8.2	Switch for ϵ -TeX	13
2.9	Conversions	13
2.9.1	Conversion to hex string	13
2.9.2	Character code to octal number	14
2.9.3	Unpack hex string	14
2.9.4	Conversion to PDF name	15
2.9.5	Conversion to PDF string	16
3	Test	17
3.1	Catcode checks for loading	17
3.2	Macro tests	19
3.3	Test with <code>\pdfescape...</code> commands	19
3.4	Test without <code>\pdfescape...</code> , with ϵ -TeX	19
3.5	Test without <code>\pdfescape...</code> and ϵ -TeX	19
3.6	Test with LuaTeX	19
3.7	Check/ensure test preconditions	19
3.7.1	Check <code>\pdfescape...</code>	19
3.7.2	Check ϵ -TeX	19
3.7.3	Check LuaTeX	20
3.8	Common part	20
3.8.1	Test for iniTeX	26

4	Installation	28
4.1	Download	28
4.2	Bundle installation	28
4.3	Package installation	29
4.4	Refresh file name databases	29
4.5	Some details for the interested	29
5	Catalogue	30
6	History	30
	[2007/02/21 v1.0]	30
	[2007/02/25 v1.1]	30
	[2007/03/20 v1.2]	31
	[2007/04/11 v1.3]	31
	[2007/04/21 v1.4]	31
	[2007/08/27 v1.5]	31
	[2007/09/09 v1.6]	31
	[2007/10/27 v1.7]	31
	[2007/11/11 v1.8]	31
	[2010/03/01 v1.9]	31
	[2010/11/12 v1.10]	31
	[2011/01/30 v1.11]	31
	[2011/04/04 v1.12]	31
	[2011/11/25 v1.13]	31
7	Index	32

1 Documentation

```
\EdefEscapeHex {<cmd>} {<string>}
\EdefUnescapeHex {<cmd>} {<string>}
\EdefEscapeName {<cmd>} {<string>}
\EdefEscapeString {<cmd>} {<string>}
```

These commands converts $\langle string \rangle$ and stores the result in macro $\langle cmd \rangle$. The conversion result is the same as the conversion of the corresponding pdfTeX's primitives. Note that the argument $\langle string \rangle$ is expanded before the conversion.

For example, if pdfTeX ≥ 1.30 is present, then `\EdefEscapeHex` becomes to:

```
\def\EdefEscapeHex#1#2{%
  \edef#1{\pdfescapehex{#2}}%
}
```

The package provides implementations for the case that pdfTeX is not present (or too old). Even ε -TeX can be missing, however it is used if it is detected.

Babel. The input strings may contain shorthand characters of package `babel`.

1.1 Additional unescape macros

```
\EdefUnescapeName {<cmd>} {<string>}
```

Sequences of a hash sign with two hexadecimal digits are converted to the corresponding character (PDF-1.2). A hash sign that is not followed by two hexadecimal digits is left unchanged. The catcodes in the result string follow TeX's conventions. The space has catcode 10 (space) and the other characters have catcode 12 (other).

```
\EdefUnescapeString {\langle cmd\rangle} {\langle string\rangle}
```

Macro $\langle cmd \rangle$ stores the unescaped string in $\langle string \rangle$. All the rules for literal strings are implemented, see PDF specification. The catcodes in the result string follow TeX's conventions.

1.2 Sanitizing macro

```
\EdefSanitize {\langle cmd\rangle} {\langle string\rangle}
```

Argument $\langle string \rangle$ is expanded, converted to a string of tokens with catcode 12 (other) and space tokens, and stored in macro $\langle cmd \rangle$.

2 Implementation

```
1 \langle *package\rangle
```

2.1 Reload check and package identification

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

```
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@pdfescape.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{pdfescape}{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@pdfescape.sty\endcsname
67 \ProvidesPackage{pdfescape}%
68 [2011/11/25 v1.13 Implements pdfTeX's escape features (H0)]%

```

2.2 Catcodes

```

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 PE@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#3{%
95   \edef\PE@AtEnd{%
96     \PE@AtEnd
97     #1#2=\the#1#2\relax
98   }%
99   #1#2=#3\relax

```

```

100 }
101 \TMP@EnsureCode\catcode{0}{12}% ^^@
102 \TMP@EnsureCode\catcode{34}{12}% "
103 \TMP@EnsureCode\catcode{36}{3}% $
104 \TMP@EnsureCode\catcode{38}{4}% &
105 \TMP@EnsureCode\catcode{39}{12}% '
106 \TMP@EnsureCode\catcode{42}{12}% *
107 \TMP@EnsureCode\catcode{45}{12}% -
108 \TMP@EnsureCode\catcode{46}{12}% .
109 \TMP@EnsureCode\catcode{47}{12}% /
110 \TMP@EnsureCode\catcode{60}{12}% <
111 \TMP@EnsureCode\catcode{62}{12}% >
112 \TMP@EnsureCode\catcode{91}{12}% [
113 \TMP@EnsureCode\catcode{93}{12}% ]
114 \TMP@EnsureCode\catcode{94}{7}% ^
115 \TMP@EnsureCode\catcode{96}{12}% `
116 \TMP@EnsureCode\uccode{34}{0}% "
117 \TMP@EnsureCode\uccode{48}{0}% 0
118 \TMP@EnsureCode\uccode{61}{0}% =
119 \edef\PE@AtEnd{\PE@AtEnd\noexpand\endinput}

```

2.3 Load package

```

120 \begingroup\expandafter\expandafter\expandafter\endgroup
121 \expandafter\ifx\csname RequirePackage\endcsname\relax
122   \def\TMP@RequirePackage#1[#2]{%
123     \begingroup\expandafter\expandafter\expandafter\endgroup
124     \expandafter\ifx\csname ver@#1.sty\endcsname\relax
125       \input #1.sty\relax
126     \fi
127   }%
128   \TMP@RequirePackage{ltxcms}[2010/04/08]%
129 \else
130   \RequirePackage{ltxcms}[2010/04/08]%
131 \fi

```

2.4 Sanitizing

`\EdefSanitize` Macro `\EdefSanitize` takes #2, entirely converts it to token with catcode 12 (other) and stores the result in macro #1.

```

132 \begingroup\expandafter\expandafter\expandafter\endgroup
133 \expandafter\ifx\csname detokenize\endcsname\relax
134   \long\def\EdefSanitize#1#2{%
135     \begingroup
136       \csname @safe@activestruel\endcsname
137       \edef#1{#2}%
138       \PE@onelevel@sanitize#1%
139     \expandafter\endgroup
140     \expandafter\def\expandafter#1\expandafter{#1}%
141   }%
142   \begingroup\expandafter\expandafter\expandafter\endgroup
143   \expandafter\ifx\csname @onelevel@sanitize\endcsname\relax
144     \def\PE@onelevel@sanitize#1{%
145       \edef#1{\expandafter\PE@strip@prefix\meaning#1}%
146     }%
147     \def\PE@strip@prefix#1>{>%
148   \else
149     \let\PE@onelevel@sanitize\@onelevel@sanitize
150   \fi
151 \else
152   \long\def\EdefSanitize#1#2{%
153     \begingroup
154       \csname @safe@activestruel\endcsname
155       \edef#1{#2}%

```

```

156   \expandafter\endgroup
157   \expandafter\def\expandafter#1\expandafter{%
158     \detokenize\expandafter{#1}%
159   }%
160 }%
161 \def\PE@onelevel@sanitize#1{%
162   \edef#1{\detokenize\expandafter{#1}}%
163 }%
164 \fi

```

`\PE@sanitize` Macro `\PE@sanitize` is only defined for compatibility with version 1.4. Its use is deprecated.

```
165 \let\PE@sanitize\EdefSanitize
```

2.4.1 Space characters

`\PE@space@other`

```

166 \begingroup
167 \catcode`\ =12\relax%
168 \def\x{\endgroup\def\PE@space@other{ }}\x\relax

```

`\PE@space@space`

```
169 \def\PE@space@space{ }
```

2.4.2 Space normalization

`\PE@SanitizeSpaceOther`

```

170 \def\PE@SanitizeSpaceOther#1{%
171   \edef#1{\expandafter\PE@SpaceToOther#1 \relax}%
172 }

```

`\PE@SpaceToOther`

```

173 \def\PE@SpaceToOther#1 #2\relax{%
174   #1%
175   \ifx\#2\%
176   \else
177     \PE@space@other
178     \ltx@ReturnAfterFi{%
179       \PE@SpaceToOther#2\relax
180     }%
181   \fi
182 }

```

2.5 `\EdefUnescapeName`

`\EdefUnescapeName`

```

183 \def\EdefUnescapeName#1#2{%
184   \EdefSanitize#1{#2}%
185   \PE@SanitizeSpaceOther#1%
186   \PE@UnescapeName#1%
187   \PE@onelevel@sanitize#1%
188 }

```

`\PE@UnescapeName`

```

189 \begingroup
190 \catcode`\$=6 % hash
191 \catcode`\#=12 % other
192 \gdef\PE@UnescapeName$1{%
193   \begingroup
194     \PE@InitUccodeHexDigit
195     \def\PE@result{}%

```

```

196     \expandafter\PE@DeName$1#\relax\relax
197   \expandafter\endgroup
198   \expandafter\def\expandafter$1\expandafter{\PE@result}%
199 }%
200 \gdef\PE@DeName$1#$2$3{%
201   \ifx\relax$2%
202     \edef\PE@result{\PE@result$1}%
203     \let\PE@next\relax
204   \else
205     \ifx\relax$3%
206       % wrong escape sequence in input
207       \edef\PE@result{\PE@result$1#}%
208       \let\PE@next\relax
209     \else
210       \uppercase{%
211         \def\PE@testA{$2}%
212         \def\PE@testB{$3}%
213       }%
214       \ifcase\ifcase\expandafter\PE@TestUcHexDigit\PE@testA
215         \ifcase\expandafter\PE@TestUcHexDigit\PE@testB
216           \ltx@zero
217         \else
218           \ltx@one
219         \fi
220       \else
221         \ltx@one
222       \fi
223       \uccode\ltx@zero="\PE@testA\PE@testB\relax
224       \uppercase{%
225         \def\PE@temp{^^@}%
226       }%
227       \uccode\ltx@zero=\ltx@zero
228       \edef\PE@result{\PE@result$1\PE@temp}%
229       \let\PE@next\PE@DeName
230     \else
231       % wrong escape sequence in input
232       \edef\PE@result{\PE@result$1#}%
233       \def\PE@next{\PE@DeName$2$3}%
234     \fi
235   \fi
236 \fi
237 \PE@next
238 }%
239 \endgroup

```

\PE@InitUccodeHexDigit

```

240 \def\PE@InitUccodeHexDigit{%
241   \uccode`a=`A\relax
242   \uccode`b=`B\relax
243   \uccode`c=`C\relax
244   \uccode`d=`D\relax
245   \uccode`e=`E\relax
246   \uccode`f=`F\relax
247   \uccode`A=\ltx@zero
248   \uccode`B=\ltx@zero
249   \uccode`C=\ltx@zero
250   \uccode`D=\ltx@zero
251   \uccode`E=\ltx@zero
252   \uccode`F=\ltx@zero
253   \uccode`0=\ltx@zero
254   \uccode`1=\ltx@zero
255   \uccode`2=\ltx@zero
256   \uccode`3=\ltx@zero

```

```

257 \uccode`4=\ltx@zero
258 \uccode`5=\ltx@zero
259 \uccode`6=\ltx@zero
260 \uccode`7=\ltx@zero
261 \uccode`8=\ltx@zero
262 \uccode`9=\ltx@zero
263 }

```

\PE@TestUcHexDigit

```

264 \def\PE@TestUcHexDigit#1{%
265 \ifnum`#1<48 % 0
266 \ltx@one
267 \else
268 \ifnum`#1>70 % F
269 \ltx@one
270 \else
271 \ifnum`#1>57 % 9
272 \ifnum`#1<65 % A
273 \ltx@one
274 \else
275 \ltx@zero
276 \fi
277 \else
278 \ltx@zero
279 \fi
280 \fi
281 \fi
282 }

```

2.6 \EdefUnescapeString

\EdefUnescapeString

```

283 \def\EdefUnescapeString#1#2{%
284 \EdefSanitize#1{#2}%
285 \PE@SanitizeSpaceOther#1%
286 \PE@NormalizeLineEnd#1%
287 \PE@UnescapeString#1%
288 \PE@onelevel@sanitize#1%
289 }

```

```

290 \begingroup
291 \uccode`\8=10 % lf
292 \uccode`\9=13 % cr
293 \def\x#1#2{\endgroup

```

\PE@NormalizeLineEnd

```

294 \def\PE@NormalizeLineEnd##1{%
295 \def\PE@result{}%
296 \expandafter\PE@@NormalizeLineEnd##1#2\relax
297 \let##1\PE@result
298 }%

```

\PE@@NormalizeLineEnd

```

299 \def\PE@@NormalizeLineEnd##1##2##3{%
300 \ifx\relax##2%
301 \edef\PE@result{\PE@result##1}%
302 \let\PE@next\relax
303 \else
304 \edef\PE@result{\PE@result##1#1}%
305 \ifx#1##2% lf
306 \let\PE@next\PE@@NormalizeLineEnd
307 \else

```



```

308     \def\PE@next{\PE@@NormalizeLineEnd##2}%
309     \fi
310     \fi
311     \PE@next
312 }%
313 }%
314 \uppercase{%
315   \x 89%
316 }

317 \beginingroup
318 \catcode`\|=0 %
319 \catcode`\|=12 %

```

\PE@UnescapeString

```

320 |gdef|PE@UnescapeString#1{%
321   |beginingroup
322   |def|PE@result{}%
323   |expandafter|PE@DeString#1\|relax
324   |expandafter|endgroup
325   |expandafter|def|expandafter#1|expandafter{|PE@result}%
326 }%

```

\PE@DeString

```

327 |gdef|PE@DeString#1\#2{%
328   |ifx|relax#2%
329   |edef|PE@result{|PE@result#1}%
330   |let|PE@next|relax
331   |else
332   |if n#2%
333     |uccode|ltx@zero=10 %
334   |elseif r#2%
335     |uccode|ltx@zero=13 %
336   |elseif t#2%
337     |uccode|ltx@zero=9 %
338   |elseif b#2%
339     |uccode|ltx@zero=8 %
340   |elseif f#2%
341     |uccode|ltx@zero=12 %
342   |else
343     |uccode|ltx@zero|=ltx@zero
344   |fi|fi|fi|fi|fi
345   |ifnum|uccode|ltx@zero>|ltx@zero
346     |uppercase{%
347       |edef|PE@temp{^^@}%
348     }%
349   |edef|PE@result{|PE@result#1|PE@temp}%
350   |let|PE@next|PE@DeString
351   |else
352     |if\#2% backslash
353     |edef|PE@result{|PE@result#1}%
354     |let|PE@next|PE@CheckEndBackslash
355   |else
356     |ifnum`#2=10 % linefeed
357     |edef|PE@result{|PE@result#1}%
358     |let|PE@next|PE@DeString
359   |else
360     |ifcase|PE@TestOctDigit#2%
361     |edef|PE@result{|PE@result#1}%
362     |def|PE@next{|PE@OctI#2}%
363   |else
364     |edef|PE@result{|PE@result#1#2}%
365     |let|PE@next|PE@DeString

```

```

366         |fi
367         |fi
368     |fi
369     |fi
370     |fi
371     |PE@next
372 }%

```

\PE@CheckEndBackslash

```

373 |gdef|PE@CheckEndBackslash#1{%
374     |ifx|relax#1%
375     |else
376         |edef|PE@result{|PE@result\}%
377         |expandafter|PE@DeString|expandafter#1%
378     |fi
379 }%

```

```

380 |endgroup

```

\PE@TestOctDigit

```

381 \def\PE@TestOctDigit#1{%
382     \ifnum`#1<48 % 0
383         \ltx@one
384     \else
385         \ifnum`#1>55 % 7
386             \ltx@one
387         \else
388             \ltx@zero
389         \fi
390     \fi
391 }

```

\PE@OctI

```

392 \def\PE@OctI#1#2{%
393     \ifcase\PE@TestOctDigit#2%
394         \def\PE@next{\PE@OctII{#1#2}}%
395     \else
396         \def\PE@next{\PE@OctAll#1#2}%
397     \fi
398     \PE@next
399 }

```

\PE@OctII

```

400 \def\PE@OctII#1#2{%
401     \ifcase\PE@TestOctDigit#2%
402         \def\PE@next{\PE@OctIII#1#2}%
403     \else
404         \def\PE@next{\PE@OctAll{#1}#2}%
405     \fi
406     \PE@next
407 }

```

```

408 \ltx@ifundefined{numexpr}{%
409     \catcode`\$=9 %
410     \catcode`\&=14 %
411 }{%
412     \catcode`\$=14 %
413     \catcode`\&=9 %
414 }

```

\PE@OctIII

```

415 \def\PE@OctIII#1#2#3{%

```

```

416 \ifnum#1<4 %
417   \def\PE@next{\PE@OctAll{#1#2#3}}%
418 \else
419 $   \count\ltx@cclv#1 %
420 $   \advance\count\ltx@cclv -4 %
421   \edef\PE@next{%
422     \noexpand\PE@OctAll{%
423 $     \the\count\ltx@cclv
424 &     \the\numexpr#1-4\relax
425     #2#3%
426   }%
427 }%
428 \fi
429 \PE@next
430 }

```

\PE@OctAll

```

431 \def\PE@OctAll#1{%
432   \uccode\ltx@zero='#1\relax
433   \uppercase{%
434     \edef\PE@result{\PE@result^^@}%
435   }%
436   \PE@DeString
437 }

```

2.7 User macros (pdfTeX analogues)

```

438 \begingroup\expandafter\expandafter\expandafter\endgroup
439 \expandafter\ifx\csname RequirePackage\endcsname\relax
440   \def\TMP@RequirePackage#1[#2]{%
441     \begingroup\expandafter\expandafter\expandafter\endgroup
442     \expandafter\ifx\csname ver@#1.sty\endcsname\relax
443       \input #1.sty\relax
444     \fi
445   }%
446   \TMP@RequirePackage{pdftexcmds}[2007/11/11]%
447 \else
448   \RequirePackage{pdftexcmds}[2007/11/11]%
449 \fi

450 \begingroup\expandafter\expandafter\expandafter\endgroup
451 \expandafter\ifx\csname pdf@escapehex\endcsname\relax

```

\EdefEscapeHex

```

452 \long\def\EdefEscapeHex#1#2{%
453   \EdefSanitize#1{#2}%
454   \PE@SanitizeSpaceOther#1%
455   \PE@EscapeHex#1%
456 }%

```

\EdefUnescapeHex

```

457 \def\EdefUnescapeHex#1#2{%
458   \EdefSanitize#1{#2}%
459   \PE@UnescapeHex#1%
460 }%

```

\EdefEscapeName

```

461 \long\def\EdefEscapeName#1#2{%
462   \EdefSanitize#1{#2}%
463   \PE@SanitizeSpaceOther#1%
464   \PE@EscapeName#1%
465 }%

```

```

\edefEscapeString
466 \long\def\EdefEscapeString#1#2{%
467   \EdefSanitize#1{#2}%
468   \PE@SanitizeSpaceOther#1%
469   \PE@EscapeString#1%
470   }%

471 \else

\PE@edefbabel Help macro that adds support for babel's shorthand characters.
472 \long\def\PE@edefbabel#1#2#3{%
473   \begingroup
474   \csname @save@activestruel\endcsname
475   \edef#1{#2{#3}}%
476   \expandafter\endgroup
477   \expandafter\def\expandafter#1\expandafter{#1}%
478   }%

\EdefEscapeHex
479 \long\def\EdefEscapeHex#1#2{%
480   \PE@edefbabel#1\pdf@escapehex{#2}%
481   }%

\EdefUnescapeHex
482 \def\EdefUnescapeHex#1#2{%
483   \PE@edefbabel#1\pdf@unescapehex{#2}%
484   }%

\EdefEscapeName
485 \long\def\EdefEscapeName#1#2{%
486   \PE@edefbabel#1\pdf@escapename{#2}%
487   }%

\EdefEscapeString
488 \long\def\EdefEscapeString#1#2{%
489   \PE@edefbabel#1\pdf@escapestring{#2}%
490   }%

491 \expandafter\PE@AtEnd
492 \fi%

```

2.8 Help macros

2.8.1 Characters

Special characters with catcode 12 (other) are created and stored in macros.

```

\PE@hash
493 \edef\PE@hash{\string#}

\PE@backslash
494 \begingroup
495 \escapechar=-1 %
496 \edef\x{\endgroup
497   \def\noexpand\PE@backslash{\string\\}}%
498 }
499 \x

```

2.8.2 Switch for -T_EX

```
500 \ltx@newif\ifPE@etex
501 \begingroup\expandafter\expandafter\expandafter\endgroup
502 \expandafter\ifx\csname numexpr\endcsname\relax
503 \else
504   \PE@etextrue
505 \fi
```

2.9 Conversions

2.9.1 Conversion to hex string

\PE@EscapeHex

```
506 \ifPE@etex
507   \def\PE@EscapeHex#1{%
508     \edef#1{\expandafter\PE@ToHex#1\relax}%
509   }%
510 \else
511   \def\PE@EscapeHex#1{%
512     \def\PE@result{%
513       \expandafter\PE@ToHex#1\relax
514       \let#1\PE@result
515     }%
516 \fi
```

\PE@ToHex

```
517 \def\PE@ToHex#1{%
518   \ifx\relax#1%
519   \else
520     \PE@HexChar{#1}%
521   \expandafter\PE@ToHex
522 \fi
523 }%
```

\PE@HexChar

```
524 \ifPE@etex
525   \def\PE@HexChar#1{%
526     \PE@HexDigit{\numexpr\dimexpr.0625\dimexpr`#1sp\relax\relax\relax}%
527     \PE@HexDigit{%
528       \numexpr`#1-16*\dimexpr.0625\dimexpr`#1sp\relax\relax\relax
529     }%
530   }%
531 \else
532   \def\PE@HexChar#1{%
533     \dimen0=`#1sp%
534     \dimen2=.0625\dimen0 %
535     \advance\dimen0-16\dimen2 %
536     \edef\PE@result{%
537       \PE@result
538       \PE@HexDigit{\dimen2 }%
539       \PE@HexDigit{\dimen0 }%
540     }%
541   }%
542 \fi
```

\PE@HexDigit

```
543 \def\PE@HexDigit#1{%
544   \expandafter\string
545   \ifcase#1%
546     0\or 1\or 2\or 3\or 4\or 5\or 6\or 7\or 8\or 9\or
547     A\or B\or C\or D\or E\or F%
548   \fi
549 }
```

2.9.2 Character code to octal number

`\PE@OctChar`

```
550 \ifPE@etex
551 \def\PE@OctChar#1{%
552   \expandafter\PE@@OctChar
553     \the\numexpr\dimexpr.015625\dimexpr`#1sp\relax\relax
554     \expandafter\relax
555     \expandafter\relax
556     \the\numexpr\dimexpr.125\dimexpr`#1sp\relax\relax\relax
557     \relax
558     #1%
559 }%
560 \def\PE@@OctChar#1\relax#2\relax#3{%
561   \PE@backslash
562   #1%
563   \the\numexpr#2-8*#1\relax
564   \the\numexpr\dimexpr`#3sp\relax-8*#2\relax
565 }%
566 \else
567 \def\PE@OctChar#1{%
568   \dimen0=`#1sp%
569   \dimen2=.125\dimen0 %
570   \dimen4=.125\dimen2 %
571   \advance\dimen0-8\dimen2 %
572   \advance\dimen2-8\dimen4 %
573   \edef\PE@result{%
574     \PE@result
575     \PE@backslash
576     \number\dimen4 %
577     \number\dimen2 %
578     \number\dimen0 %
579   }%
580 }%
581 \fi
```

2.9.3 Unpack hex string

`\PE@UnescapeHex`

```
582 \def\PE@UnescapeHex#1{%
583   \begingroup
584   \PE@InitUccodeHexDigit
585   \def\PE@result{%
586     \expandafter\PE@DeHex#1\relax\relax
587   \expandafter\endgroup
588   \expandafter\def\expandafter#1\expandafter{\PE@result}%
589 }
```

`\PE@DeHex`

```
590 \def\PE@DeHex#1#2{%
591   \ifx#2\relax
592     \ifx#1\relax
593       \let\PE@next\relax
594     \else
595       \uppercase{%
596         \def\PE@testA{#1}%
597       }%
598       \ifcase\expandafter\PE@TestUcHexDigit\PE@testA
599         \def\PE@next{%
600           \PE@DeHex#10\relax\relax
601         }%
602       \else
```

```

603     \let\PE@next\relax
604   \fi
605 \fi
606 \else
607   \uppercase{%
608     \def\PE@testA{#1}%
609     \def\PE@testB{#2}%
610   }%
611   \ifcase\expandafter\PE@TestUcHexDigit\PE@testA
612     \ifcase\expandafter\PE@TestUcHexDigit\PE@testB
613       \uccode\ltx@zero="\PE@testA\PE@testB\relax
614       \ifnum\uccode\ltx@zero=32 %
615         \let\PE@temp\PE@space@space
616       \else
617         \uppercase{%
618           \def\PE@temp{^^@}%
619         }%
620       \fi
621     \edef\PE@result{\PE@result\PE@temp}%
622     \let\PE@next\PE@DeHex
623   \else
624     % invalid input sequence
625     \def\PE@next{%
626       \PE@DeHex#1%
627     }%
628   \fi
629 \else
630   % invalid input sequence
631   \def\PE@next{\PE@DeHex#2}%
632 \fi
633 \fi
634 \PE@next
635 }

```

2.9.4 Conversion to PDF name

\PE@EscapeName

```

636 \ifPE@etex
637   \def\PE@EscapeName#1{%
638     \edef#1{\expandafter\PE@EscapeNameTokens#1\relax}%
639   }%
640 \else
641   \def\PE@EscapeName#1{%
642     \def\PE@result{}%
643     \expandafter\PE@EscapeNameTokens#1\relax
644     \let#1\PE@result
645   }%
646 \fi

```

\PE@EscapeNameTokens

```

647 \def\PE@EscapeNameTokens#1{%
648   \ifx\relax#1%
649   \else
650     \ifnum`#1<33 %
651       \ifcase`#1 %
652         % drop illegal zero
653       \else
654         \PE@EscapeNameAdd\PE@hash
655         \PE@HexChar#1%
656       \fi
657     \else
658       \ifnum`#1>126 %

```

```

659     \PE@EscapeNameAdd\PE@hash
660     \PE@HexChar#1%
661     \else \ifnum`#1=35 \PE@EscapeNameHashChar 23% #
662     \else\ifnum`#1=37 \PE@EscapeNameHashChar 25% %
663     \else\ifnum`#1=40 \PE@EscapeNameHashChar 28% (
664     \else\ifnum`#1=41 \PE@EscapeNameHashChar 29% )
665     \else\ifnum`#1=47 \PE@EscapeNameHashChar 2F% /
666     \else\ifnum`#1=60 \PE@EscapeNameHashChar 3C% <
667     \else\ifnum`#1=62 \PE@EscapeNameHashChar 3E% >
668     \else\ifnum`#1=91 \PE@EscapeNameHashChar 5B% [
669     \else\ifnum`#1=93 \PE@EscapeNameHashChar 5D% ]
670     \else\ifnum`#1=123 \PE@EscapeNameHashChar 7B% {
671     \else\ifnum`#1=125 \PE@EscapeNameHashChar 7D% }
672     \else
673     \PE@EscapeNameAdd{#1}%
674     \fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi
675     \fi
676     \fi
677     \expandafter\PE@EscapeNameTokens
678     \fi
679 }%
680 \def\PE@EscapeNameHashChar#1#2{%
681 \PE@EscapeNameAdd{\PE@hash\string#1\string#2}%
682 }%

```

\PE@EscapeNameAdd

```

683 \ifPE@etex
684 \def\PE@EscapeNameAdd#1{#1}%
685 \else
686 \def\PE@EscapeNameAdd#1{%
687 \edef\PE@result{%
688 \PE@result
689 #1%
690 }%
691 }%
692 \fi

```

2.9.5 Conversion to PDF string

\PE@EscapeString

```

693 \ifPE@etex
694 \def\PE@EscapeString#1{%
695 \edef#1{\expandafter\PE@EscapeStringTokens#1\relax}%
696 }%
697 \else
698 \def\PE@EscapeString#1{%
699 \begingroup
700 \def\PE@result{}%
701 \expandafter\PE@EscapeStringTokens#1\relax
702 \expandafter\endgroup
703 \expandafter\def\expandafter#1\expandafter{\PE@result}%
704 }%
705 \fi

```

\PE@EscapeStringTokens

```

706 \def\PE@EscapeStringTokens#1{%
707 \ifx\relax#1%
708 \else
709 \ifnum`#1<33 %
710 \PE@OctChar#1%
711 \else
712 \ifnum`#1>126 %

```



```

713     \PE@OctChar#1%
714     \else \ifnum`#1=40 \PE@EscapeStringAdd{\string\}% (
715     \else\ifnum`#1=41 \PE@EscapeStringAdd{\string\)}% )
716     \else\ifnum`#1=92 \PE@EscapeStringAdd{\string\}% \
717     \else
718     \PE@EscapeStringAdd{#1}%
719     \fi\fi\fi
720     \fi
721     \fi
722     \expandafter\PE@EscapeStringTokens
723     \fi
724 }%

```

\PE@EscapeStringAdd

```

725 \ifPE@etex
726 \def\PE@EscapeStringAdd#1{#1}%
727 \else
728 \def\PE@EscapeStringAdd#1{%
729 \edef\PE@result{%
730 \PE@result
731 #1%
732 }%
733 }%
734 \fi

735 \PE@AtEnd%
736 </package>

```

3 Test

3.1 Catcode checks for loading

```

737 <*test1>
738 \catcode`\{=1 %
739 \catcode`\}=2 %
740 \catcode`\#=6 %
741 \catcode`\@=11 %
742 \expandafter\ifx\csname count@\endcsname\relax
743 \countdef\count@=255 %
744 \fi
745 \expandafter\ifx\csname @gobble\endcsname\relax
746 \long\def\@gobble#1{}%
747 \fi
748 \expandafter\ifx\csname @firstofone\endcsname\relax
749 \long\def\@firstofone#1{#1}%
750 \fi
751 \expandafter\ifx\csname loop\endcsname\relax
752 \expandafter\@firstofone
753 \else
754 \expandafter\@gobble
755 \fi
756 {%
757 \def\loop#1\repeat{%
758 \def\body{#1}%
759 \iterate
760 }%
761 \def\iterate{%
762 \body
763 \let\next\iterate
764 \else
765 \let\next\relax

```

```

766     \fi
767     \next
768   }%
769   \let\repeat=\fi
770 }%
771 \def\RestoreCatcodes{}
772 \count@=0 %
773 \loop
774   \edef\RestoreCatcodes{%
775     \RestoreCatcodes
776     \catcode\the\count@=\the\catcode\count@\relax
777   }%
778 \ifnum\count@<255 %
779   \advance\count@ 1 %
780 \repeat
781
782 \def\RangeCatcodeInvalid#1#2{%
783   \count@=#1\relax
784   \loop
785     \catcode\count@=15 %
786   \ifnum\count@<#2\relax
787     \advance\count@ 1 %
788   \repeat
789 }
790 \def\RangeCatcodeCheck#1#2#3{%
791   \count@=#1\relax
792   \loop
793     \ifnum#3=\catcode\count@
794     \else
795       \errmessage{%
796         Character \the\count@\space
797         with wrong catcode \the\catcode\count@\space
798         instead of \number#3%
799       }%
800     \fi
801   \ifnum\count@<#2\relax
802     \advance\count@ 1 %
803   \repeat
804 }
805 \def\space{ }
806 \expandafter\ifx\csname LoadCommand\endcsname\relax
807 \def\LoadCommand{\input pdfescape.sty\relax}%
808 \fi
809 \def\Test{%
810   \RangeCatcodeInvalid{0}{47}%
811   \RangeCatcodeInvalid{58}{64}%
812   \RangeCatcodeInvalid{91}{96}%
813   \RangeCatcodeInvalid{123}{255}%
814   \catcode`\@=12 %
815   \catcode`\=0 %
816   \catcode`\%=14 %
817   \LoadCommand
818   \RangeCatcodeCheck{0}{36}{15}%
819   \RangeCatcodeCheck{37}{37}{14}%
820   \RangeCatcodeCheck{38}{47}{15}%
821   \RangeCatcodeCheck{48}{57}{12}%
822   \RangeCatcodeCheck{58}{63}{15}%
823   \RangeCatcodeCheck{64}{64}{12}%
824   \RangeCatcodeCheck{65}{90}{11}%
825   \RangeCatcodeCheck{91}{91}{15}%
826   \RangeCatcodeCheck{92}{92}{0}%
827   \RangeCatcodeCheck{93}{96}{15}%

```

```

828 \RangeCatcodeCheck{97}{122}{11}%
829 \RangeCatcodeCheck{123}{255}{15}%
830 \RestoreCatcodes
831 }
832 \Test
833 \csname @@end\endcsname
834 \end
835 </test1>

```

3.2 Macro tests

```

836 <*test2 j test3 j test4 j test5>
837 \NeedsTeXFormat{LaTeX2e}
838 \makeatletter

```

3.3 Test with \pdfescape... commands

```

839 <*test2>
840 \ProvidesFile{pdfescape-test2.tex}%
841 [2011/11/25 v1.13 Test with \string\pdfescape... commands]%
842 </test2>

```

3.4 Test without \pdfescape..., with -TeX

```

843 <*test3>
844 \ProvidesFile{pdfescape-test3.tex}%
845 [2011/11/25 v1.13 Test without \string\pdfescape..., with e-TeX]%
846 </test3>

```

3.5 Test without \pdfescape... and -TeX

```

847 <*test4>
848 \ProvidesFile{pdfescape-test4.tex}%
849 [2011/11/25 v1.13 Test without \string\pdfescape... and e-TeX]%
850 </test4>

```

3.6 Test with LuaTeX

```

851 <*test5>
852 \ProvidesFile{pdfescape-test5.tex}%
853 [2011/11/25 v1.13 Test with LuaTeX]%
854 </test5>

```

3.7 Check/ensure test preconditions

3.7.1 Check \pdfescape...

```

855 <*test2>
856 \@ifundefined{pdfescapehex}{%
857 \PackageError{pdfescape-test2}{%
858 Missing \string\pdfescape... commands%
859 }{Test aborted.}%
860 \stop
861 }{}
862 </test2>

863 <*test3 j test4>
864 \let\pdfescapehex\@undefined
865 \let\pdfunescapehex\@undefined
866 \let\pdfescapename\@undefined
867 \let\pdfescapestring\@undefined
868 </test3 j test4>

```

3.7.2 Check -TeX

```

869 <*test3>
870 \@ifundefined{numexpr}{%
871 \PackageError{pdfescape-test3}{%

```

```

872   Missing \eTeX
873   }{Test aborted.}%
874   \stop
875 }{}
876 </test3>

```

Package `qstest` uses ε -TeX, thus ε -TeX's features can only be disabled later during loading of package `pdfescape`.

3.7.3 Check LuaTeX

```

877 <*test5>
878 \@ifundefined{directlua}{%
879   \PackageError{pdfescape-test5}{%
880     Missing LuaTeX%
881   }{Test aborted.}%
882   \stop
883 }{}
884 </test5>

```

3.8 Common part

The files for testing uses the framework, provided by the new package `qstest` of David Kastrup.

```

885 \RequirePackage{qstest}
886 \IncludeTests{*}
887 \LogTests[log]{*}{*}
888
889 \newcommand*\ExpectVar}[2]{%
890   \ifx#1#2%
891   \else
892     \begingroup
893       \@onelevel@sanitize#1%
894       \@onelevel@sanitize#2%
895       \typeout{[#1] <> [#2]}% hash-ok
896     \endgroup
897   \fi
898   \Expect*\ifx#1#2true\else false\fi}{true}%
899 }
900
901 \makeatletter
902 \begingroup
903   \gdef\AllBytes{}%
904   \count@=0 %
905   \catcode0=12 %
906   \@whilenum\count@<256 \do{%
907     \lccode0=\count@
908     \ifnum\count@=32 %
909       \xdef\AllBytes{\AllBytes\space}%
910     \else
911       \lowercase{%
912         \xdef\AllBytes{\AllBytes^^@}%
913       }%
914     \fi
915     \advance\count@ by 1 %
916   }%
917 \endgroup
918 \newcommand*\AllBytesHex}{%
919   000102030405060708090A0B0C0D0E0F%
920   101112131415161718191A1B1C1D1E1F%
921   202122232425262728292A2B2C2D2E2F%
922   303132333435363738393A3B3C3D3E3F%
923   404142434445464748494A4B4C4D4E4F%
924   505152535455565758595A5B5C5D5E5F%
925   606162636465666768696A6B6C6D6E6F%

```

```

926 707172737475767778797A7B7C7D7E7F%
927 808182838485868788898A8B8C8D8E8F%
928 909192939495969798999A9B9C9D9E9F%
929 A0A1A2A3A4A5A6A7A8A9AAABACADAEAF%
930 B0B1B2B3B4B5B6B7B8B9ABBBBCDBEBF%
931 C0C1C2C3C4C5C6C7C8C9CACBCCDCECF%
932 D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF%
933 E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF%
934 F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF%
935 }
936 \@onelevel@sanitize\AllBytesHex
937 \expandafter\lowercase\expandafter{%
938 \expandafter\newcommand\expandafter*\expandafter\AllBytesHexLC
939 \expandafter{\AllBytesHex}%
940 }
941 \newcommand*{\AllBytesName}{%
942 \begingroup
943 \catcode\#=12 %
944 \xdef\AllBytesName{%
945 #01#02#03#04#05#06#07#08#09#0A#0B#0C#0D#0E#0F%
946 #10#11#12#13#14#15#16#17#18#19#1A#1B#1C#1D#1E#1F%
947 #20!"#23$#25&'#28#29*+,-.#2F%
948 0123456789:;#3C=#3E?%
949 @ABCDEFGHIJKLMNO%
950 PQRSTUVWXYZ#5B\@backslashchar#5D^_%
951 `abcdefghijklmno%
952 pqrstuvwxyz#7B|#7D\string~#7F%
953 #80#81#82#83#84#85#86#87#88#89#8A#8B#8C#8D#8E#8F%
954 #90#91#92#93#94#95#96#97#98#99#9A#9B#9C#9D#9E#9F%
955 #A0#A1#A2#A3#A4#A5#A6#A7#A8#A9#AA#AB#AC#AD#AE#AF%
956 #B0#B1#B2#B3#B4#B5#B6#B7#B8#B9#BA#BB#BC#BD#BE#BF%
957 #C0#C1#C2#C3#C4#C5#C6#C7#C8#C9#CA#CB#CC#CD#CE#CF%
958 #D0#D1#D2#D3#D4#D5#D6#D7#D8#D9#DA#DB#DC#DD#DE#DF%
959 #E0#E1#E2#E3#E4#E5#E6#E7#E8#E9#EA#EB#EC#ED#EE#EF%
960 #F0#F1#F2#F3#F4#F5#F6#F7#F8#F9#FA#FB#FC#FD#FE#FF%
961 }%
962 \endgroup
963 \@onelevel@sanitize\AllBytesName
964
965 \newcommand*{\AllBytesString}{%
966 \begingroup
967 \def\|{|}%
968 \edef\%{\@percentchar}%
969 \catcode\|=0 %
970 \catcode\#=12 %
971 \catcode\~=12 %
972 \catcode\|=12 %
973 |xdef|AllBytesString{%
974 \000\001\002\003\004\005\006\007\010\011\012\013\014\015\016\017%
975 \020\021\022\023\024\025\026\027\030\031\032\033\034\035\036\037%
976 \040!"#$|%&'(\)*+,-./%
977 0123456789:;<=>?%
978 @ABCDEFGHIJKLMNO%
979 PQRSTUVWXYZ[\]^_%
980 `abcdefghijklmno%
981 pqrstuvwxyz{|}~\177%
982 \200\201\202\203\204\205\206\207\210\211\212\213\214\215\216\217%
983 \220\221\222\223\224\225\226\227\230\231\232\233\234\235\236\237%
984 \240\241\242\243\244\245\246\247\250\251\252\253\254\255\256\257%
985 \260\261\262\263\264\265\266\267\270\271\272\273\274\275\276\277%
986 \300\301\302\303\304\305\306\307\310\311\312\313\314\315\316\317%
987 \320\321\322\323\324\325\326\327\330\331\332\333\334\335\336\337%

```

```

988     \340\341\342\343\344\345\346\347\350\351\352\353\354\355\356\357%
989     \360\361\362\363\364\365\366\367\370\371\372\373\374\375\376\377%
990   }%
991 |endgroup
992 \@onelevel@sanitize\AllBytesString
993
994 <*test4>
995 \let\org@detokenize\detokenize
996 \let\detokenize\undefined
997 \let\org@numexpr\numexpr
998 \let\numexpr\undefined
999 </test4>
1000 \RequirePackage{pdfescape}
1001 <*test4>
1002 \let\detokenize\org@detokenize
1003 \let\numexpr\org@numexpr
1004 </test4>
1005
1006 \begin{qstest}{all-hex}{\AllBytes, escapehex}
1007   \EdefEscapeHex\x{\AllBytes}%
1008   \Expect*{\x}*{\AllBytesHex}%
1009   \ExpectVar\x\AllBytesHex
1010 \end{qstest}
1011
1012 \begin{qstest}{all-unhex}{\AllBytesHex, unescapehex}
1013   \EdefUnescapeHex\x{\AllBytesHex}%
1014   \Expect*{\x}*{\AllBytes}%
1015   \ExpectVar\x\AllBytes
1016 \end{qstest}
1017
1018 \begin{qstest}{all-unhex-lc}{\AllBytesHexLC, unescapehex, lowercase}
1019   \EdefUnescapeHex\x{\AllBytesHexLC}%
1020   \Expect*{\x}*{\AllBytes}%
1021   \ExpectVar\x\AllBytes
1022 \end{qstest}
1023
1024 \begin{qstest}{unhex-incomplete}{unescapehex, incomplete}
1025   \EdefUnescapeHex\x{4}%
1026   \Expect*{\x}{@}%
1027 \end{qstest}
1028
1029 \begin{qstest}{unhex-space}{unescapehex, space}
1030   \EdefUnescapeHex\x{20}%
1031   \Expect*{\x}{ }%
1032   \ExpectVar\x\space
1033 \end{qstest}
1034
1035 \begin{qstest}{unhex-spaces}{unescapehex, spaces}
1036   \EdefUnescapeHex\x{204020204120}%
1037   \def\y#1{%
1038     \edef\z{#1\string @#1#1\string A#1}%
1039   }\y{ }%
1040   \Expect*{\x}*{\z}%
1041   \ExpectVar\x\z
1042 \end{qstest}
1043
1044 \begin{qstest}{unhex-hash}{unescapehex, hash}
1045   \catcode`\#=12 %
1046   \EdefUnescapeHex\x{#20}%
1047   \ExpectVar\x\space
1048 \end{qstest}
1049

```

```

1050 \begin{qstest}{unhex-invalid}{unescapehex, invalid}
1051   \def\test#1#2{%
1052     \EdefUnescapeHex\x{#1}%
1053     \edef\y{#2}%
1054     \@onelevel@sanitize\y
1055     \ExpectVar\x\y
1056   }%
1057 <*test2>
1058   \edef\x{\pdfunescapehex{4X}}%
1059   \edef\y{\string @}%
1060   \ifx\x\y
1061   \else
1062     \def~{\space}%
1063     \typeout{*****}%
1064     \typeout{* Your pdfTeX contains bug 777.~~~*}%
1065     \typeout{* This test is redefined as dummy, *}%
1066     \typeout{* because it triggers the bug.~~~*}%
1067     \typeout{*****}%
1068     \def\test#1#2{%
1069       \fi
1070 </test2>
1071   \test{X}{}%
1072   \test{XY}{}%
1073   \test{XYZ}{}%
1074   \test{A}{^~a0}%
1075   \test{AX}{^^a0}%
1076   \test{XA}{^^a0}%
1077   \test{XXAXX}{^^a0}%
1078 \end{qstest}
1079
1080 \begin{qstest}{all-name}{\AllBytes, escapename}
1081   \EdefEscapeName\x{\AllBytes}%
1082   \Expect*{\x}*{\AllBytesName}%
1083   \ExpectVar\x\AllBytesName
1084 \end{qstest}
1085
1086 \begin{qstest}{all-string}{\AllBytes, escapestring}
1087   \EdefEscapeString\x{\AllBytes}%
1088   \Expect*{\x}*{\AllBytesString}%
1089   \ExpectVar\x\AllBytesString
1090 \end{qstest}
1091
1092 \begin{qstest}{uchexdigit}{unescape, uppercase hex digit}
1093   \catcode\@=11 %
1094   \catcode0=12 %
1095   \def\test#1#2{%
1096     \uccode0=#1\relax
1097     \uppercase{%
1098       \def\x{^^@}%
1099     }%
1100     \Expect*{%
1101       \ifcase\expandafter\PE@TestUcHexDigit\x
1102         true%
1103       \else
1104         false%
1105       \fi
1106     }{#2}%
1107   }%
1108   \def\range#1#2#3{%
1109     \count0=#1\relax
1110     \loop
1111     \ifnum\count0<#2\relax

```

```

1112     \test{\count0}{#3}%
1113     \advance\count0 by 1 %
1114   \repeat
1115 }%
1116 \range{0}{47}{false}%
1117 \range{48}{57}{true}%
1118 \range{58}{64}{false}%
1119 \range{65}{70}{true}%
1120 \range{71}{255}{false}%
1121 \end{qstest}
1122
1123 \begin{qstest}{unescapename}{unescapename}
1124   \def\test#1#2{%
1125     \EdefUnescapeName\x{#1}%
1126     \edef\y{#2}%
1127     \@onelevel@sanitize\y
1128     \ExpectVar\x\y
1129   }%
1130   \catcode`\#=12 %
1131   \catcode0=12 %
1132   \test{}{}%
1133   \test{x}{x}%
1134   \test{xy}{xy}%
1135   \test{#}{#}%
1136   \test{##}{##}%
1137   \test{###}{###}%
1138   \test{####}{####}%
1139   \test{#x}{#x}%
1140   \test{#xy}{#xy}%
1141   \test{#1}{#1}%
1142   \test{#40}{@}%
1143   \test{#400}{@0}%
1144   \test{#4x0}{#4x0}%
1145   \test{#ab}{~ab}%
1146   \test{#00}{~@}%
1147   \test{x#40y#40z}{x@y@z}%
1148   \test{#40#40#40#40}{@@@}%
1149   \test{a#x}{a#x}%
1150   \test{a#xy}{a#xy}%
1151   \test{a#1}{a#1}%
1152   \test{a#40}{a@}%
1153   \test{a#400}{a@0}%
1154   \test{#20}{ }%
1155   \test{a#20}{a }%
1156   \test{a#20b}{a b}%
1157   \test{a#20#20#20b}{a \space\space b}%
1158 \end{qstest}
1159
1160 \begin{qstest}{unescapestring}{unescapestring}
1161   \def\test#1#2{%
1162     \EdefUnescapeString\x{#1}%
1163     \edef\y{#2}%
1164     \@onelevel@sanitize\y
1165     \ExpectVar\x\y
1166   }%
1167   \catcode0=12 %
1168   \def\DefChar#1#2{%
1169     \begingroup
1170     \uccode0=#2\relax
1171     \uppercase{\endgroup
1172     \def#1{~@}%
1173     }%

```



```

1174 }%
1175 \DefChar\nul{0}%
1176 \DefChar\one{1}%
1177 \DefChar\bel{8}%
1178 \DefChar\tab{9}%
1179 \DefChar\lf{10}%
1180 \DefChar\ff{12}%
1181 \DefChar\cr{13}%
1182 \DefChar\{92}%
1183 \test{}{ }%
1184 \test{a}{a}%
1185 \test{\}{ }%
1186 \test{\}{ }%
1187 \test{\}{ }%
1188 \test{\000}{\nul}%
1189 \test{\b}{\bel}%
1190 \test{\t}{\tab}%
1191 \test{\n}{\lf}%
1192 \test{\f}{\ff}%
1193 \test{\r}{\cr}%
1194 \test{\}{ }%
1195 \test{\}{ }%
1196 \test{\040}{ }%
1197 \test{\100}{@}%
1198 \test{\40}{ }%
1199 \test{\1}{\one}%
1200 \test{\01}{\one}%
1201 \test{\001}{\one}%
1202 \test{\18}{\one8}%
1203 \test{\018}{\one8}%
1204 \test{\0018}{\one8}%
1205 \test{x}{x}%
1206 \test{x}{x}%
1207 \test{x}{x}%
1208 \test{x\000}{x\nul}%
1209 \test{x\b}{x\bel}%
1210 \test{x\t}{x\tab}%
1211 \test{x\n}{x\lf}%
1212 \test{x\f}{x\ff}%
1213 \test{x\r}{x\cr}%
1214 \test{x}{x}%
1215 \test{x}{x}%
1216 \test{x\040}{x }%
1217 \test{x\100}{x@}%
1218 \test{x\40}{x }%
1219 \test{x\1}{x\one}%
1220 \test{x\01}{x\one}%
1221 \test{x\001}{x\one}%
1222 \test{x\18}{x\one8}%
1223 \test{x\018}{x\one8}%
1224 \test{x\0018}{x\one8}%
1225 \test{\b\t\n\f\r\(\)\000\040}{%
1226 \bel\tab\lf\ff\cr()\nul\space
1227 }%
1228 \test{\lf}{ }%
1229 \test{x\lf}{x}%
1230 \test{\cr}{\lf}%
1231 \test{\cr\lf}{\lf}%
1232 \test{\lf}{\lf}%
1233 \test{\lf\cr}{\lf\lf}%
1234 \test{x\cr}{x\lf}%
1235 \test{x\cr\lf}{x\lf}%

```

```

1236 \test{x\lf}{x\lf}%
1237 \test{x\lf\cr}{x\lf\lf}%
1238 \test{x\\\cr\lf y\cr}{xy\lf}%
1239 %
1240 \test{\\409}{ 9}%
1241 \test{\\800}{800}%
1242 \test{\\900}{900}%
1243 \test{\\578}{/8}%
1244 \test{\\477}{?}%
1245 \test{\\377}{~ff}%
1246 \test{\\777}{^^ff}%
1247 \test{\\7777}{^^ff7}%
1248 \end{qstest}
1249 \stop
1250 </test2 j test3 j test4 j test5>

```

3.8.1 Test for iniTeX

```

1251 <*test6>
1252 \input pdfescape.sty\relax
1253 \catcode`\{=1 %
1254 \catcode`\}=2 %
1255 \catcode`\#=6 %
1256 \catcode`\^=7 %
1257 \catcode`\@=11 %
1258 \begingroup
1259 \catcode`\@=11 %
1260 \countdef\count@=255 %
1261 \def\space{ }%
1262 \long\def\@whilenum#1\do #2{%
1263   \ifnum #1\relax
1264     #2\relax
1265     \@iwhilenum{#1\relax#2\relax}%
1266   \fi
1267 }%
1268 \long\def\@iwhilenum#1{%
1269   \ifnum #1%
1270     \expandafter\@iwhilenum
1271   \else
1272     \expandafter\ltx@gobble
1273   \fi
1274   {#1}%
1275 }%
1276 \gdef\AllBytes{}%
1277 \count@=0 %
1278 \catcode0=12 %
1279 \@whilenum\count@<256 \do{%
1280   \lccode0=\count@
1281   \ifnum\count@=32 %
1282     \xdef\AllBytes{\AllBytes\space}%
1283   \else
1284     \lowercase{%
1285       \xdef\AllBytes{\AllBytes^^@}%
1286     }%
1287   \fi
1288   \advance\count@ by 1 %
1289 }%
1290 \endgroup
1291 \def\AllBytesHex{%
1292 000102030405060708090A0B0C0D0E0F%
1293 101112131415161718191A1B1C1D1E1F%
1294 202122232425262728292A2B2C2D2E2F%

```

```

1295 303132333435363738393A3B3C3D3E3F%
1296 404142434445464748494A4B4C4D4E4F%
1297 505152535455565758595A5B5C5D5E5F%
1298 606162636465666768696A6B6C6D6E6F%
1299 707172737475767778797A7B7C7D7E7F%
1300 808182838485868788898A8B8C8D8E8F%
1301 909192939495969798999A9B9C9D9E9F%
1302 A0A1A2A3A4A5A6A7A8A9AAABACADAEAF%
1303 B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF%
1304 C0C1C2C3C4C5C6C7C8C9CACBCCDCECF%
1305 D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF%
1306 E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF%
1307 F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF%
1308 }
1309 \ltx@onelevel@sanitize\AllBytesHex
1310 \expandafter\lowercase\expandafter{%
1311 \expandafter\def\expandafter\AllBytesHexLC
1312 \expandafter{\AllBytesHex}%
1313 }
1314 \begingroup
1315 \catcode\#=12 %
1316 \xdef\AllBytesName{%
1317 #01#02#03#04#05#06#07#08#09#0A#0B#0C#0D#0E#0F%
1318 #10#11#12#13#14#15#16#17#18#19#1A#1B#1C#1D#1E#1F%
1319 #20!"#$%&'#28#29*+,-.#2F%
1320 0123456789:;#3C=#3E?%
1321 @ABCDEFGHIJKLMNO%
1322 PQRSTUVWXYZ#5B\ltx@backslashchar#5D^_%
1323 `abcdefghijklmnop%
1324 pqrstuvwxyz#7B|#7D\string~#7F%
1325 #80#81#82#83#84#85#86#87#88#89#8A#8B#8C#8D#8E#8F%
1326 #90#91#92#93#94#95#96#97#98#99#9A#9B#9C#9D#9E#9F%
1327 #A0#A1#A2#A3#A4#A5#A6#A7#A8#A9#AA#AB#AC#AD#AE#AF%
1328 #B0#B1#B2#B3#B4#B5#B6#B7#B8#B9#BA#BB#BC#BD#BE#BF%
1329 #C0#C1#C2#C3#C4#C5#C6#C7#C8#C9#CA#CB#CC#CD#CE#CF%
1330 #D0#D1#D2#D3#D4#D5#D6#D7#D8#D9#DA#DB#DC#DD#DE#DF%
1331 #E0#E1#E2#E3#E4#E5#E6#E7#E8#E9#EA#EB#EC#ED#EE#EF%
1332 #F0#F1#F2#F3#F4#F5#F6#F7#F8#F9#FA#FB#FC#FD#FE#FF%
1333 }%
1334 \endgroup
1335 \ltx@onelevel@sanitize\AllBytesName
1336 \edef\AllBytesFromName{\expandafter\ltx@gobble\AllBytes}
1337
1338 \begingroup
1339 \def\|{|}%
1340 \edef%\{\ltx@percentchar}%
1341 \catcode\|=0 %
1342 \catcode\#=12 %
1343 \catcode\~ =12 %
1344 \catcode\|=12 %
1345 |xdef|AllBytesString{%
1346 \000\001\002\003\004\005\006\007\010\011\012\013\014\015\016\017%
1347 \020\021\022\023\024\025\026\027\030\031\032\033\034\035\036\037%
1348 \040!"#$%&'(\)*+,-./%
1349 0123456789:;<=>?%
1350 @ABCDEFGHIJKLMNO%
1351 PQRSTUVWXYZ[\]\^_%
1352 `abcdefghijklmnop%
1353 pqrstuvwxyz{|}~\177%
1354 \200\201\202\203\204\205\206\207\210\211\212\213\214\215\216\217%
1355 \220\221\222\223\224\225\226\227\230\231\232\233\234\235\236\237%
1356 \240\241\242\243\244\245\246\247\250\251\252\253\254\255\256\257%

```

```

1357 \260\261\262\263\264\265\266\267\270\271\272\273\274\275\276\277%
1358 \300\301\302\303\304\305\306\307\310\311\312\313\314\315\316\317%
1359 \320\321\322\323\324\325\326\327\330\331\332\333\334\335\336\337%
1360 \340\341\342\343\344\345\346\347\350\351\352\353\354\355\356\357%
1361 \360\361\362\363\364\365\366\367\370\371\372\373\374\375\376\377%
1362 }%
1363 |endgroup
1364 \ltx@onelevel@sanitize\AllBytesString
1365 \def\msg#{\immediate\write16}
1366 \def\Test#1#2#3{%
1367 \begingroup
1368 #1\TestResult{#2}%
1369 \ifx\TestResult#3%
1370 \else
1371 \newlinechar=10 %
1372 \msg{Expect:^^J#3}%
1373 \msg{Result:^^J\TestResult}%
1374 \errmessage{\string#2 -\string#1-> \string#3}%
1375 \fi
1376 \endgroup
1377 }
1378 \Test\EdefEscapeHex\AllBytes\AllBytesHex
1379 \Test\EdefUnescapeHex\AllBytesHex\AllBytes
1380 \Test\EdefEscapeName\AllBytes\AllBytesName
1381 \Test\EdefUnescapeName\AllBytesName\AllBytesFromName
1382 \Test\EdefEscapeString\AllBytes\AllBytesString
1383 \Test\EdefUnescapeString\AllBytesString\AllBytes
1384 \csname @@end\endcsname\end
1385 </test6>

```

4 Installation

4.1 Download

Package. This package is available on CTAN¹:

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

[CTAN:macros/latex/contrib/oberdiek/pdfescape.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.

4.2 Bundle installation

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

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

¹<http://ftp.ctan.org/tex-archive/>

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 pdfescape.dtx
```

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

```
pdfescape.sty      → tex/generic/oberdiek/pdfescape.sty
pdfescape.pdf      → doc/latex/oberdiek/pdfescape.pdf
test/pdfescape-test1.tex → doc/latex/oberdiek/test/pdfescape-test1.tex
test/pdfescape-test2.tex → doc/latex/oberdiek/test/pdfescape-test2.tex
test/pdfescape-test3.tex → doc/latex/oberdiek/test/pdfescape-test3.tex
test/pdfescape-test4.tex → doc/latex/oberdiek/test/pdfescape-test4.tex
test/pdfescape-test5.tex → doc/latex/oberdiek/test/pdfescape-test5.tex
test/pdfescape-test6.tex → doc/latex/oberdiek/test/pdfescape-test6.tex
pdfescape.dtx      → source/latex/oberdiek/pdfescape.dtx
```

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

4.4 Refresh file name databases

If your \TeX distribution (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

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 pdfescape.pdf unpack_files output .
```

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{pdfescape.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 pdfescape.dtx
makeindex -s gind.ist pdfescape.idx
pdflatex pdfescape.dtx
makeindex -s gind.ist pdfescape.idx
pdflatex pdfescape.dtx
```

5 Catalogue

The following XML file can be used as source for the [T_EX Catalogue](#). The elements `caption` and `description` are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is `pdfescape.xml`.

```
1386 (*catalogue)
1387 <?xml version='1.0' encoding='us-ascii'?>
1388 <!DOCTYPE entry SYSTEM 'catalogue.dtd'>
1389 <entry datestamp='$Date$' modifier='$Author$' id='pdfescape'>
1390   <name>pdfescape</name>
1391   <caption>Implements pdfTeX's escape features using TeX or e-TeX.</caption>
1392   <authorref id='auth:oberdiek'>/>
1393   <copyright owner='Heiko Oberdiek' year='2007,2010,2011'>/>
1394   <license type='lpl1.3'>/>
1395   <version number='1.13'>/>
1396   <description>
1397     This package implements <xref refid='pdftex'>pdfTeX</xref>'s
1398     escape features (<tt>\pdfescapehex</tt>, <tt>\pdfunescapehex</tt>,
1399     <tt>\pdfescapename</tt>, <tt>\pdfescapestring</tt>) using TeX or
1400     e-TeX.
1401   <p/>
1402   The package is part of the <xref refid='oberdiek'>oberdiek</xref>
1403   bundle.
1404 </description>
1405 <documentation details='Package documentation'
1406   href='ctan:/macros/latex/contrib/oberdiek/pdfescape.pdf'>/>
1407 <ctan file='true' path='/macros/latex/contrib/oberdiek/pdfescape.dtx'>/>
1408 <miktex location='oberdiek'>/>
1409 <texlive location='oberdiek'>/>
1410 <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip'>/>
1411 </entry>
1412 </catalogue>
```

6 History

[2007/02/21 v1.0]

- First version.

[2007/02/25 v1.1]

- Test files added.
- `\EdefUnescapeHex` supports lowercase letters.
- Fix: `\EdefEscapeName{^^@}`

- Fix: `\EdefEscapeName{\string#}`
- Fix for `\EdefUnescapeHex` in case of incomplete hex string.
- Fix: `\EdefUnescapeHex` generates space tokens with catcode 10 (space) in all cases.
- Fix: `\EdefEscapeHex` and `\EdefEscapeName` now generate tokens with catcode 12 (other) only.

[2007/03/20 v1.2]

- Fix: Wrong year in `\ProvidesPackage`.

[2007/04/11 v1.3]

- Line ends sanitized.

[2007/04/21 v1.4]

- `\EdefUnescapeName` and `\EdefUnescapeString` added.

[2007/08/27 v1.5]

- `\EdefSanitize` added (replaces `\PE@sanitize`).

[2007/09/09 v1.6]

- Fix in catcode setup.

[2007/10/27 v1.7]

- More efficient `\EdefSanitize`.

[2007/11/11 v1.8]

- Use of package `pdftexcmds` for Lua \TeX support.

[2010/03/01 v1.9]

- Compatibility with `ini \TeX` .

[2010/11/12 v1.10]

- Use of package `ltxcmds`.
- Fix for compatibility with `ini \TeX` .

[2011/01/30 v1.11]

- Already loaded package files are not input in plain \TeX .

[2011/04/04 v1.12]

- Further fixes for compatibility for `ini \TeX` .
- Test file for `ini \TeX` added.

[2011/11/25 v1.13]

- Higher order bit of octal sequences in `\EdefUnescapeString` ignored according to the PDF specification (Bug found by Bruno Le Floch).

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>	191, 327, 352, 740, 943, 970, 1045, 1130, 1255, 1315, 1342
<code>\\$</code>	190, 409, 412
<code>\%</code>	816, 968, 1340
<code>\&</code>	410, 413
<code>\(</code>	714, 976, 1348
<code>\)</code>	715, 976, 1348
<code>\@</code>	741, 814, 1093, 1257, 1259
<code>\@backslashchar</code>	950
<code>\@firstofone</code>	749, 752
<code>\@gobble</code>	746, 754
<code>\@ifundefined</code>	856, 870, 878
<code>\@iwhilenum</code>	1265, 1268, 1270
<code>\@onelevel@sanitize</code> ...	149, 893, 894, 936, 963, 992, 1054, 1127, 1164
<code>\@percentchar</code>	968
<code>\@undefined</code> 58, 864, 865, 866, 867, 996, 998
<code>\@whilenum</code>	906, 1262, 1279
<code>\\</code>	175, 319, 497, 716, 815, 972, 979, 1182, 1185, 1186, 1187, 1188, 1189, 1190, 1191, 1192, 1193, 1194, 1195, 1196, 1197, 1198, 1199, 1200, 1201, 1202, 1203, 1204, 1205, 1206, 1207, 1208, 1209, 1210, 1211, 1212, 1213, 1214, 1215, 1216, 1217, 1218, 1219, 1220, 1221, 1222, 1223, 1224, 1225, 1226, 1228, 1229, 1238, 1240, 1241, 1242, 1243, 1244, 1245, 1246, 1247, 1344, 1351
<code>\{</code>	738, 1253
<code>\}</code>	376, 739, 1254
<code>\^</code>	1256
<code>\ </code>	318, 323, 967, 969, 1339, 1341
<code>\~</code>	971, 1343
Numbers	
<code>\0</code>	974, 975, 976, 1346, 1347, 1348
<code>\1</code>	981, 1353
<code>\2</code>	982, 983, 984, 985, 1354, 1355, 1356, 1357
<code>\3</code>	986, 987, 988, 989, 1358, 1359, 1360, 1361
<code>\8</code>	291
<code>\9</code>	292
<code>_</code>	167, 716
A	
<code>\advance</code>	420, 535, 571, 572, 779, 787, 802, 915, 1113, 1288
<code>\aftergroup</code>	29
<code>\AllBytes</code>	903, 909, 912, 1006, 1007, 1014, 1015, 1020, 1021, 1080, 1081, 1086, 1087, 1276, 1282, 1285, 1336, 1378, 1379, 1380, 1382, 1383
<code>\AllBytesFromName</code>	1336, 1381
<code>\AllBytesHex</code>	918, 936, 939, 1008, 1009, 1012, 1013, 1291, 1309, 1312, 1378, 1379
<code>\AllBytesHexLC</code> ..	938, 1018, 1019, 1311
<code>\AllBytesName</code>	941, 944, 963, 1082, 1083, 1316, 1335, 1380, 1381
<code>\AllBytesString</code>	965, 992, 1088, 1089, 1364, 1382, 1383
B	
<code>\begin</code>	1006, 1012, 1018, 1024, 1029, 1035, 1044, 1050, 1080, 1086, 1092, 1123, 1160
<code>\bel</code>	1177, 1189, 1209, 1226
<code>\body</code>	758, 762
C	
<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, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 167, 190, 191, 318, 319, 409, 410, 412, 413, 738, 739, 740, 741, 776, 785, 793, 797, 814, 815, 816, 905, 943, 969, 970, 971, 972, 1045, 1093, 1094, 1130, 1131, 1167, 1253, 1254, 1255, 1256, 1257, 1259, 1278, 1315, 1341, 1342, 1343, 1344
<code>\count</code>	419, 420, 423, 1109, 1111, 1112, 1113
<code>\count@</code>	743, 772, 776, 778, 779, 783, 785, 786, 787, 791, 793, 796, 797, 801, 802, 904, 906, 907, 908, 915, 1260, 1277, 1279, 1280, 1281, 1288
<code>\countdef</code>	743, 1260
<code>\cr</code> ..	1181, 1193, 1213, 1226, 1230, 1231, 1233, 1234, 1235, 1237, 1238
<code>\csname</code>	14, 21, 50, 66, 76, 121, 124, 133, 136, 143, 154, 439, 442, 451, 474, 502, 742, 745, 748, 751, 806, 833, 1384
D	
<code>\DefChar</code>	1168, 1175, 1176, 1177, 1178, 1179, 1180, 1181, 1182
<code>\detokenize</code> ..	158, 162, 995, 996, 1002

`\dimen` . 533, 534, 535, 538, 539, 568,
569, 570, 571, 572, 576, 577, 578
`\dimexpr` 526, 528, 553, 556, 564
`\do` 906, 1262, 1279

E

`\EdefEscapeHex` . 2, [452](#), [479](#), 1007, 1378
`\EdefEscapeName` . . [461](#), [485](#), 1081, 1380
`\EdefEscapeString` [466](#), [488](#), 1087, 1382
`\EdefSanitize` 3, [132](#),
165, 184, 284, 453, 458, 462, 467
`\EdefUnescapeHex`
. [457](#), [482](#), 1013, 1019,
1025, 1030, 1036, 1046, 1052, 1379
`\EdefUnescapeName` . 2, [183](#), 1125, 1381
`\EdefUnescapeString` 3, [283](#), 1162, 1383
`\empty` 17, 18
`\end` 834, 1010, 1016, 1022,
1027, 1033, 1042, 1048, 1078,
1084, 1090, 1121, 1158, 1248, 1384
`\endcsname` 14, 21, 50,
66, 76, 121, 124, 133, 136, 143,
154, 439, 442, 451, 474, 502,
742, 745, 748, 751, 806, 833, 1384
`\endinput` 29, 119
`\endlinechar` 4, 35, 71, 77, 89
`\errmessage` 795, 1374
`\escapechar` 495
`\eTeX` 872
`\Expect` 898, 1008, 1014, 1020,
1026, 1031, 1040, 1082, 1088, 1100
`\ExpectVar` 889,
1009, 1015, 1021, 1032, 1041,
1047, 1055, 1083, 1089, 1128, 1165

F

`\ff` 1180, 1192, 1212, 1226

G

`\gdef` 192, 200, 903, 1276

I

`\ifcase` 214, 215, 393,
401, 545, 598, 611, 612, 651, 1101
`\ifnum` 265, 268, 271, 272, 382,
385, 416, 614, 650, 658, 661,
662, 663, 664, 665, 666, 667,
668, 669, 670, 671, 709, 712,
714, 715, 716, 778, 786, 793,
801, 908, 1111, 1263, 1269, 1281
`\ifPE@etex` 500,
506, 524, 550, 636, 683, 693, 725
`\ifx` 15, 18, 21, 50, 58, 61, 121, 124,
133, 143, 175, 201, 205, 300,
305, 439, 442, 451, 502, 518,
591, 592, 648, 707, 742, 745,
748, 751, 806, 890, 898, 1060, 1369
`\immediate` 23, 52, 1365
`\includeTests` 886
`\input` 125, 443, 807, 1252
`\iterate` 759, 761, 763

L

`\lccode` 907, 1280

`\lf` 1179, 1191, 1211, 1226,
1228, 1229, 1230, 1231, 1232,
1233, 1234, 1235, 1236, 1237, 1238
`\LoadCommand` 807, 817
`\LogTests` 887
`\loop` 757, 773, 784, 792, 1110
`\lowercase` 911, 937, 1284, 1310
`\ltx@backslashchar` 1322
`\ltx@ccclv` 419, 420, 423
`\ltx@gobble` 1272, 1336
`\ltx@ifUndefined` 408
`\ltx@newif` 500
`\ltx@one` 218, 221, 266, 269, 273, 383, 386
`\ltx@onelevel@sanitize`
. 1309, 1335, 1364
`\ltx@percentchar` 1340
`\ltx@ReturnAfterFi` 178
`\ltx@zero` 216, 223, 227, 247, 248, 249,
250, 251, 252, 253, 254, 255,
256, 257, 258, 259, 260, 261,
262, 275, 278, 388, 432, 613, 614

M

`\makeatletter` 838, 901
`\meaning` 145
`\msg` 1365, 1372, 1373

N

`\NeedsTeXFormat` 837
`\newcommand` . . . 889, 918, 938, 941, 965
`\newlinechar` 1371
`\next` 763, 765, 767
`\nul` 1175, 1188, 1208, 1226
`\number` 576, 577, 578, 798
`\numexpr` 424, 526, 528,
553, 556, 563, 564, 997, 998, 1003

O

`\one` 1176, 1199,
1200, 1201, 1202, 1203, 1204,
1219, 1220, 1221, 1222, 1223, 1224
`\org@detokenize` 995, 1002
`\org@numexpr` 997, 1003

P

`\PackageError` 857, 871, 879
`\PackageInfo` 26
`\pdf@escapehex` 480
`\pdf@escapename` 486
`\pdf@escapestring` 489
`\pdf@unescapehex` 483
`\pdfescape` 841, 845, 849, 858
`\pdfescapehex` 864, 1398
`\pdfescapename` 866, 1399
`\pdfescapestring` 867, 1399
`\pdfunescapehex` 865, 1058, 1398
`\PE@@NormalizeLineEnd` 296, [299](#)
`\PE@@OctChar` 552, 560
`\PE@AtEnd` 95, 96, 119, 491, 735
`\PE@backslash` [494](#), 561, 575
`\PE@CheckEndBackslash` [373](#)
`\PE@DeHex` 586, [590](#)
`\PE@DeName` 196, 200, 229, 233

\PE@DeString 327, 436
\PE@edefbabel .. 472, 480, 483, 486, 489
\PE@EscapeHex 455, 506
\PE@EscapeName 464, 636
\PE@EscapeNameAdd
..... 654, 659, 673, 681, 683
\PE@EscapeNameHashChar
..... 661, 662, 663, 664, 665,
666, 667, 668, 669, 670, 671, 680
\PE@EscapeNameTokens .. 638, 643, 647
\PE@EscapeString 469, 693
\PE@EscapeStringAdd
..... 714, 715, 716, 718, 725
\PE@EscapeStringTokens . 695, 701, 706
\PE@etextrtrue 504
\PE@hash 493, 654, 659, 681
\PE@HexChar 520, 524, 655, 660
\PE@HexDigit .. 526, 527, 538, 539, 543
\PE@InitUccodeHexDigit . 194, 240, 584
\PE@next 203, 208, 229, 233, 237, 302,
306, 308, 311, 394, 396, 398,
402, 404, 406, 417, 421, 429,
593, 599, 603, 622, 625, 631, 634
\PE@NormalizeLineEnd 286, 294
\PE@OctAll 396, 404, 417, 422, 431
\PE@OctChar 550, 710, 713
\PE@OctI 392
\PE@OctII 394, 400
\PE@OctIII 402, 415
\PE@onelevel@sanitize
..... 138, 144, 149, 161, 187, 288
\PE@result 195, 198, 202,
207, 228, 232, 295, 297, 301,
304, 434, 512, 514, 536, 537,
573, 574, 585, 588, 621, 642,
644, 687, 688, 700, 703, 729, 730
\PE@sanitize 165
\PE@SanitizeSpaceOther
..... 170, 185, 285, 454, 463, 468
\PE@space@other 166, 177
\PE@space@space 169, 615
\PE@SpaceToOther 171, 173
\PE@strip@prefix 145, 147
\PE@temp 225, 228, 615, 618, 621
\PE@testA 211,
214, 223, 596, 598, 608, 611, 613
\PE@testB . 212, 215, 223, 609, 612, 613
\PE@TestOctDigit 381, 393, 401
\PE@TestUcHexDigit
214, 215, 264, 598, 611, 612, 1101
\PE@ToHex 508, 513, 517
\PE@UnescapeHex 459, 582
\PE@UnescapeName 186, 189
\PE@UnescapeString 287, 320
\ProvidesFile 840, 844, 848, 852
\ProvidesPackage 19, 67

R

\range 1108, 1116, 1117, 1118, 1119, 1120
\RangeCatcodeCheck
. 790, 818, 819, 820, 821, 822,
823, 824, 825, 826, 827, 828, 829

\RangeCatcodeInvalid
..... 782, 810, 811, 812, 813
\repeat .. 757, 769, 780, 788, 803, 1114
\RequirePackage ... 130, 448, 885, 1000
\RestoreCatcodes .. 771, 774, 775, 830

S

\space 796, 797, 805, 909, 1032,
1047, 1062, 1157, 1226, 1261, 1282
\stop 860, 874, 882, 1249

T

\TAB 1178, 1190, 1210, 1226
\TEST 809, 832, 1366,
1378, 1379, 1380, 1381, 1382, 1383
\TEST 1051, 1068, 1071, 1072, 1073,
1074, 1075, 1076, 1077, 1095,
1112, 1124, 1132, 1133, 1134,
1135, 1136, 1137, 1138, 1139,
1140, 1141, 1142, 1143, 1144,
1145, 1146, 1147, 1148, 1149,
1150, 1151, 1152, 1153, 1154,
1155, 1156, 1157, 1161, 1183,
1184, 1185, 1186, 1187, 1188,
1189, 1190, 1191, 1192, 1193,
1194, 1195, 1196, 1197, 1198,
1199, 1200, 1201, 1202, 1203,
1204, 1205, 1206, 1207, 1208,
1209, 1210, 1211, 1212, 1213,
1214, 1215, 1216, 1217, 1218,
1219, 1220, 1221, 1222, 1223,
1224, 1225, 1228, 1229, 1230,
1231, 1232, 1233, 1234, 1235,
1236, 1237, 1238, 1240, 1241,
1242, 1243, 1244, 1245, 1246, 1247
\TESTResult 1368, 1369, 1373
\THE 77, 78, 79,
80, 81, 82, 83, 84, 97, 423, 424,
553, 556, 563, 564, 776, 796, 797
\TMP@EnsureCode
.. 94, 101, 102, 103, 104, 105,
106, 107, 108, 109, 110, 111,
112, 113, 114, 115, 116, 117, 118
\TMP@RequirePackage 122, 128, 440, 446
\TYPEOUT
895, 1063, 1064, 1065, 1066, 1067

U

\UCCODE 116, 117, 118,
223, 227, 241, 242, 243, 244,
245, 246, 247, 248, 249, 250,
251, 252, 253, 254, 255, 256,
257, 258, 259, 260, 261, 262,
291, 292, 432, 613, 614, 1096, 1170
\UPPERCASE 210, 224,
314, 433, 595, 607, 617, 1097, 1171

W

\WRITE 23, 52, 1365

X

\X 14, 15, 18, 22, 26, 28, 51, 56,
66, 75, 87, 168, 293, 315, 496,

499, 1007, 1008, 1009, 1013,	
1014, 1015, 1019, 1020, 1021,	
1025, 1026, 1030, 1031, 1032,	
1036, 1040, 1041, 1046, 1047,	
1052, 1055, 1058, 1060, 1081,	
1082, 1083, 1087, 1088, 1089,	
1098, 1101, 1125, 1128, 1162, 1165	
	Y
\y	1037, 1039,
	1053, 1054, 1055, 1059, 1060,
	1126, 1127, 1128, 1163, 1164, 1165
	Z
\z	1038, 1040, 1041