

# The pdfescape package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2007/04/21 v1.4

## Abstract

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

## Contents

<b>1</b>	<b>Documentation</b>	<b>2</b>
1.1	Additional unescape macros	2
<b>2</b>	<b>Implementation</b>	<b>2</b>
2.1	Catcodes	2
2.2	Reload check and package identification	3
2.3	Sanitizing	4
2.3.1	Space characters	4
2.3.2	Space normalization	5
2.4	<code>\EdefUnescapeName</code>	5
2.5	<code>\EdefUnescapeString</code>	7
2.6	User macros (pdfTeXanalogues)	9
2.7	Help macros	10
2.7.1	Characters	10
2.7.2	Switch for $\epsilon$ -TeX	11
2.8	Conversions	11
2.8.1	Conversion to hex string	11
2.8.2	Character code to octal number	12
2.8.3	Unpack hex string	12
2.8.4	Conversion to PDF name	13
2.8.5	Conversion to PDF string	14
<b>3</b>	<b>Test</b>	<b>15</b>
3.1	Test with <code>\pdfescape...</code> commands	15
3.2	Test without <code>\pdfescape...</code> , with $\epsilon$ -TeX	15
3.3	Test without <code>\pdfescape...</code> and $\epsilon$ -TeX	15
3.4	Check/ensure test preconditions	16
3.4.1	Check <code>\pdfescape...</code>	16
3.4.2	Check $\epsilon$ -TeX	16
3.5	Common part	16
<b>4</b>	<b>Installation</b>	<b>22</b>
4.1	Download	22
4.2	Bundle installation	22
4.3	Package installation	23
4.4	Refresh file name databases	23
4.5	Some details for the interested	23

<b>5 History</b>	<b>24</b>
[2007/02/21 v1.0]	24
[2007/02/25 v1.1]	24
[2007/03/20 v1.2]	24
[2007/04/11 v1.3]	24
[2007/04/21 v1.4]	24
<b>6 Index</b>	<b>24</b>

## 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  $\geq 1.30$  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  $\varepsilon$ -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 {⟨cmd⟩} {⟨string⟩}
```

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.

## 2 Implementation

```
1 ⟨*package⟩
```

### 2.1 Catcodes

```
2 \expandafter\edef\curname PE@AtEnd\endcurname{%
3   \catcode64 \the\catcode64\relax
4 }
```

```

5 \catcode64 11 % @
6 \def\PE@EnsureCode#1#2#3{%
7   \edef\PE@AtEnd{%
8     \PE@AtEnd
9     #1#2 \the#1#2\relax
10  }%
11  #1#2 #3\relax
12 }
13 \PE@EnsureCode\catcode{0}{12}% ^^@
14 \PE@EnsureCode\catcode{34}{12}% "
15 \PE@EnsureCode\catcode{42}{12}% *
16 \PE@EnsureCode\catcode{45}{12}% -
17 \PE@EnsureCode\catcode{46}{12}% .
18 \PE@EnsureCode\catcode{60}{12}% <
19 \PE@EnsureCode\catcode{61}{12}% =
20 \PE@EnsureCode\catcode{62}{12}% >
21 \PE@EnsureCode\catcode{94}{7}% ^
22 \PE@EnsureCode\catcode{96}{12}% ‘
23 \PE@EnsureCode\uccode{34}{0}% "
24 \PE@EnsureCode\uccode{48}{0}% 0
25 \PE@EnsureCode\uccode{61}{0}% =

```

## 2.2 Reload check and package identification

Reload check, especially if the package is not used with L<sup>A</sup>T<sub>E</sub>X.

```

26 \begingroup
27   \catcode44 12 % ,
28   \catcode45 12 % -
29   \catcode46 12 % .
30   \catcode58 12 % :
31   \catcode64 11 % @
32   \expandafter\let\expandafter\x\csname ver@pdfescape.sty\endcsname
33   \ifcase 0%
34     \ifx\x\relax % plain
35     \else
36       \ifx\x\empty % LaTeX
37       \else
38         1%
39       \fi
40     \fi
41   \else
42     \expandafter\ifx\csname PackageInfo\endcsname\relax
43     \def\x#1#2{%
44       \immediate\write-1{Package #1 Info: #2.}%
45     }%
46   \else
47     \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
48   \fi
49   \x{pdfescape}{The package is already loaded}%
50 \endgroup
51 \expandafter\endinput
52 \fi
53 \endgroup

```

Package identification:

```

54 \begingroup
55   \catcode40 12 % (
56   \catcode41 12 % )
57   \catcode44 12 % ,
58   \catcode45 12 % -
59   \catcode46 12 % .
60   \catcode47 12 % /
61   \catcode58 12 % :

```

```

62 \catcode64 11 % @
63 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
64   \def\x#1#2#3[#4]{\endgroup
65     \immediate\write-1{Package: #3 #4}%
66     \xdef#1{#4}%
67   }%
68 \else
69   \def\x#1#2[#3]{\endgroup
70     #2[{#3}]%
71     \ifx#1\relax
72       \xdef#1{#3}%
73     \fi
74   }%
75 \fi
76 \expandafter\x\csname ver@pdfescape.sty\endcsname
77 \ProvidesPackage{pdfescape}%
78 [2007/04/21 v1.4 Provides hex, PDF name and string conversions (H0)]

```

## 2.3 Sanitizing

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

```

79 \begingroup\expandafter\expandafter\expandafter\endgroup
80 \expandafter\ifx\csname detokenize\endcsname\relax
81   \long\def\PE@sanitize#1#2{%
82     \begingroup
83       \csname @safe@activetrue\endcsname
84       \edef#1{#2}%
85       \PE@onelevel@sanitize#1%
86     \expandafter\endgroup
87     \expandafter\def\expandafter#1\expandafter{#1}%
88   }%
89 \begingroup\expandafter\expandafter\expandafter\endgroup
90 \expandafter\ifx\csname @onelevel@sanitize\endcsname\relax
91   \def\PE@onelevel@sanitize#1{%
92     \edef#1{\expandafter\PE@strip@prefix\meaning#1}%
93   }%
94   \def\PE@strip@prefix#1>{#1}%
95 \else
96   \let\PE@onelevel@sanitize\@onelevel@sanitize
97 \fi
98 \else
99   \long\def\PE@sanitize#1#2{%
100     \begingroup
101       \csname @safe@activetrue\endcsname
102       \edef#1{#2}%
103       \PE@onelevel@sanitize#1%
104     \expandafter\endgroup
105     \expandafter\def\expandafter#1\expandafter{#1}%
106   }%
107   \def\PE@onelevel@sanitize#1{%
108     \edef#1{\detokenize\expandafter{#1}}%
109   }%
110 \fi

```

### 2.3.1 Space characters

`\PE@space@other`

```

111 \begingroup
112 \catcode'\ =12\relax%
113 \def\x{\endgroup\def\PE@space@other{ }}\x\relax

```

\PE@space@space

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

### 2.3.2 Space normalization

\PE@SanitizeSpaceOther

```
115 \def\PE@SanitizeSpaceOther#1{%
116   \edef#1{\expandafter\PE@SpaceToOther#1 \relax}%
117 }
```

\PE@SpaceToOther

```
118 \def\PE@SpaceToOther#1 #2\relax{%
119   #1%
120   \ifx\#2\%
121     \else
122       \PE@space@other
123       \@ReturnAfterFi{%
124         \PE@SpaceToOther#2\relax
125       }%
126     \fi
127 }
```

\@ReturnAfterFi

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

## 2.4 \EdefUnescapeName

\EdefUnescapeName

```
129 \def\EdefUnescapeName#1#2{%
130   \PE@sanitize#1{#2}%
131   \PE@SanitizeSpaceOther#1%
132   \PE@UnescapeName#1%
133   \PE@onelevel@sanitize#1%
134 }
```

\PE@UnescapeName

```
135 \begingroup
136   \catcode'\$=6 % hash
137   \catcode'\#=12 % other
138   \gdef\PE@UnescapeName$1{%
139     \begingroup
140       \PE@InitUccodeHexDigit
141       \def\PE@result{%
142         \expandafter\PE@DeName$1#\relax\relax
143       \expandafter\endgroup
144       \expandafter\def\expandafter$1\expandafter{\PE@result}%
145     }%
146     \gdef\PE@DeName$1#$2$3{%
147       \ifx\relax$2%
148         \edef\PE@result{\PE@result$1}%
149         \let\PE@next\relax
150       \else
151         \ifx\relax$3%
152           % wrong escape sequence in input
153           \edef\PE@result{\PE@result$1#}%
154           \let\PE@next\relax
155         \else
156           \uppercase{%
157             \def\PE@testA{$2}%
158             \def\PE@testB{$3}%

```

```

159     }%
160     \ifcase\ifcase\expandafter\PE@TestUcHexDigit\PE@testA
161         \ifcase\expandafter\PE@TestUcHexDigit\PE@testB
162             \z@
163         \else
164             \@ne
165         \fi
166     \else
167         \@ne
168     \fi
169     \uccode\z@="\PE@testA\PE@testB\relax
170     \uppercase{%
171         \def\PE@temp{^^@}%
172     }%
173     \uccode\z@=\z@
174     \edef\PE@result{\PE@result$1\PE@temp}%
175     \let\PE@next\PE@DeName
176 \else
177     % wrong escape sequence in input
178     \edef\PE@result{\PE@result$1#}%
179     \def\PE@next{\PE@DeName$2$3}%
180 \fi
181 \fi
182 \fi
183 \PE@next
184 }%
185 \endgroup

```

\PE@InitUccodeHexDigit

```

186 \def\PE@InitUccodeHexDigit{%
187     \uccode'a='A\relax
188     \uccode'b='B\relax
189     \uccode'c='C\relax
190     \uccode'd='D\relax
191     \uccode'e='E\relax
192     \uccode'f='F\relax
193     \uccode'A=\z@
194     \uccode'B=\z@
195     \uccode'C=\z@
196     \uccode'D=\z@
197     \uccode'E=\z@
198     \uccode'F=\z@
199     \uccode'0=\z@
200     \uccode'1=\z@
201     \uccode'2=\z@
202     \uccode'3=\z@
203     \uccode'4=\z@
204     \uccode'5=\z@
205     \uccode'6=\z@
206     \uccode'7=\z@
207     \uccode'8=\z@
208     \uccode'9=\z@
209 }

```

\PE@TestUcHexDigit

```

210 \def\PE@TestUcHexDigit#1{%
211     \ifnum'#1<48 % 0
212         \@ne
213     \else
214         \ifnum'#1>70 % F
215             \@ne
216         \else

```

```

217     \ifnum'#1>57 % 9
218     \ifnum'#1<65 % A
219     \@ne
220     \else
221     \z@
222     \fi
223     \else
224     \z@
225     \fi
226     \fi
227 \fi
228 }

```

## 2.5 \EdefUnescapeString

\EdefUnescapeString

```

229 \def\EdefUnescapeString#1#2{%
230   \PE@sanitize#1{#2}%
231   \PE@SanitizeSpaceOther#1%
232   \PE@NormalizeLineEnd#1%
233   \PE@UnescapeString#1%
234   \PE@onelevel@sanitize#1%
235 }

```

```

236 \begingroup
237   \uccode'\8=10 % lf
238   \uccode'\9=13 % cr
239 \def\x#1#2{\endgroup

```

\PE@NormalizeLineEnd

```

240 \def\PE@NormalizeLineEnd##1{%
241   \def\PE@result{}%
242   \expandafter\PE@@NormalizeLineEnd##1#2\relax
243   \let##1\PE@result
244 }%

```

\PE@@NormalizeLineEnd

```

245 \def\PE@@NormalizeLineEnd##1#2##2{%
246   \ifx\relax##2%
247     \edef\PE@result{\PE@result##1}%
248     \let\PE@next\relax
249   \else
250     \edef\PE@result{\PE@result##1#1}%
251     \ifx#1##2% lf
252       \let\PE@next\PE@@NormalizeLineEnd
253     \else
254       \def\PE@next{\PE@@NormalizeLineEnd##2}%
255     \fi
256   \fi
257   \PE@next
258 }%
259 }%
260 \uppercase{%
261   \x 89%
262 }

263 \begingroup
264   \catcode'\|=0 %
265   \catcode'\|=12 %

```

\PE@UnescapeString

```

266 |gdef\PE@UnescapeString#1{%

```

```

267 |begingroup
268 |def|PE@result{}%
269 |expandafter|PE@DeString#1\|relax
270 |expandafter|endgroup
271 |expandafter|def|expandafter#1|expandafter{|PE@result}%
272 }%

```

\PE@DeString

```

273 |gdef|PE@DeString#1\#2{%
274 |ifx|relax#2%
275 |edef|PE@result{|PE@result#1}%
276 |let|PE@next|relax
277 |else
278 |if n#2%
279 |uccode|z@=10 %
280 |else|if r#2%
281 |uccode|z@=13 %
282 |else|if t#2%
283 |uccode|z@=9 %
284 |else|if b#2%
285 |uccode|z@=8 %
286 |else|if f#2%
287 |uccode|z@=12 %
288 |else
289 |uccode|z@=|z@
290 |fi|fi|fi|fi|fi
291 |ifnum|uccode|z@>|z@
292 |uppercase{%
293 |edef|PE@temp{^^@}%
294 }%
295 |edef|PE@result{|PE@result#1|PE@temp}%
296 |let|PE@next|PE@DeString
297 |else
298 |if\#2% backslash
299 |edef|PE@result{|PE@result#1}%
300 |let|PE@next|PE@CheckEndBackslash
301 |else
302 |ifnum'#2=10 % linefeed
303 |edef|PE@result{|PE@result#1}%
304 |let|PE@next|PE@DeString
305 |else
306 |ifcase|PE@TestOctDigit#2%
307 |edef|PE@result{|PE@result#1}%
308 |def|PE@next{|PE@OctI#2}%
309 |else
310 |edef|PE@result{|PE@result#1#2}%
311 |let|PE@next|PE@DeString
312 |fi
313 |fi
314 |fi
315 |fi
316 |fi
317 |PE@next
318 }%

```

\PE@CheckEndBackslash

```

319 |gdef|PE@CheckEndBackslash#1{%
320 |ifx|relax#1%
321 |else
322 |edef|PE@result{|PE@result\}%
323 |expandafter|PE@DeString|expandafter#1%
324 |fi

```



```

325 }%

326 |endgroup

\PE@TestOctDigit

327 \def\PE@TestOctDigit#1{%
328   \ifnum'#1<48 % 0
329     \@ne
330   \else
331     \ifnum'#1>55 % 7
332       \@ne
333     \else
334       \z@
335     \fi
336   \fi
337 }

\PE@OctI

338 \def\PE@OctI#1#2{%
339   \ifcase\PE@TestOctDigit#2%
340     \def\PE@next{\PE@OctII{#1#2}}%
341   \else
342     \def\PE@next{\PE@OctAll{#1#2}}%
343   \fi
344   \PE@next
345 }

\PE@OctII

346 \def\PE@OctII#1#2{%
347   \ifcase\PE@TestOctDigit#2%
348     \def\PE@next{\PE@OctAll{#1#2}}%
349   \else
350     \def\PE@next{\PE@OctAll{#1#2}}%
351   \fi
352   \PE@next
353 }

\PE@OctAll

354 \def\PE@OctAll#1{%
355   \uccode\z@='#1\relax
356   \uppercase{%
357     \edef\PE@result{\PE@result^^@}%
358   }%
359   \PE@DeString
360 }

```

## 2.6 User macros (pdfTeX analogues)

```

361 \begingroup\expandafter\expandafter\expandafter\endgroup
362 \expandafter\ifx\csname pdfescapehex\endcsname\relax

\EdefEscapeHex

363 \long\def\EdefEscapeHex#1#2{%
364   \PE@sanitize#1{#2}%
365   \PE@SanitizeSpaceOther#1%
366   \PE@EscapeHex#1%
367 }%

\EdefUnescapeHex

368 \def\EdefUnescapeHex#1#2{%
369   \PE@sanitize#1{#2}%
370   \PE@UnescapeHex#1%
371 }%

```

\EdefEscapeName

```
372 \long\def\EdefEscapeName#1#2{%
373   \PE@sanitize#1{#2}%
374   \PE@SanitizeSpaceOther#1%
375   \PE@EscapeName#1%
376 }%
```

\EdefEscapeString

```
377 \long\def\EdefEscapeString#1#2{%
378   \PE@sanitize#1{#2}%
379   \PE@SanitizeSpaceOther#1%
380   \PE@EscapeString#1%
381 }%
```

382 \else

\PE@edefbabel Help macro that adds support for babel's shorthand characters.

```
383 \long\def\PE@edefbabel#1#2#3{%
384   \begingroup
385   \csname @save@activetrue\endcsname
386   \edef#1{#2{#3}}%
387   \expandafter\endgroup
388   \expandafter\def\expandafter#1\expandafter{#1}%
389 }%
```

\EdefEscapeHex

```
390 \long\def\EdefEscapeHex#1#2{%
391   \PE@edefbabel#1\pdfescapehex{#2}%
392 }%
```

\EdefUnescapeHex

```
393 \def\EdefUnescapeHex#1#2{%
394   \PE@edefbabel#1\pdfunescapehex{#2}%
395 }%
```

\EdefEscapeName

```
396 \long\def\EdefEscapeName#1#2{%
397   \PE@edefbabel#1\pdfescapename{#2}%
398 }%
```

\EdefEscapeString

```
399 \long\def\EdefEscapeString#1#2{%
400   \PE@edefbabel#1\pdfescapestring{#2}%
401 }%

402 \PE@AtEnd
403 \expandafter\endinput
404 \fi
```

## 2.7 Help macros

### 2.7.1 Characters

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

\PE@hash

```
405 \edef\PE@hash{\string#}
```

\PE@backslash

```
406 \begingroup
407   \escapechar=-1 %
408 \edef\x{\endgroup
409   \def\noexpand\PE@backslash{\string\\}%
410 }
411 \x
```

## 2.7.2 Switch for $\varepsilon$ -T<sub>E</sub>X

```
412 \newif\ifPE@etex
413 \begingroup\expandafter\expandafter\expandafter\endgroup
414 \expandafter\ifx\csname numexpr\endcsname\relax
415   \else
416     \PE@etextrue
417 \fi
```

## 2.8 Conversions

### 2.8.1 Conversion to hex string

\PE@EscapeHex

```
418 \ifPE@etex
419   \def\PE@EscapeHex#1{%
420     \edef#1{\expandafter\PE@ToHex#1\relax}%
421   }%
422 \else
423   \def\PE@EscapeHex#1{%
424     \def\PE@result{%
425       \expandafter\PE@ToHex#1\relax
426     }\let#1\PE@result
427   }%
428 \fi
```

\PE@ToHex

```
429 \def\PE@ToHex#1{%
430   \ifx\relax#1%
431   \else
432     \PE@HexChar{#1}%
433     \expandafter\PE@ToHex
434   \fi
435 }%
```

\PE@HexChar

```
436 \ifPE@etex
437   \def\PE@HexChar#1{%
438     \PE@HexDigit{\numexpr\dimexpr.0625\dimexpr'#1sp\relax\relax}%
439     \PE@HexDigit{%
440       \numexpr'#1-16*\dimexpr.0625\dimexpr'#1sp\relax\relax
441     }%
442   }%
443 \else
444   \def\PE@HexChar#1{%
445     \dimen0='#1sp%
446     \dimen2=.0625\dimen0 %
447     \advance\dimen0-16\dimen2 %
448     \edef\PE@result{%
449       \PE@result
450       \PE@HexDigit{\dimen2 }%
451       \PE@HexDigit{\dimen0 }%
452     }%
453   }%
454 \fi
```

\PE@HexDigit

```
455 \def\PE@HexDigit#1{%
456   \expandafter\string
457   \ifcase#1%
458     0\or 1\or 2\or 3\or 4\or 5\or 6\or 7\or 8\or 9\or
459     A\or B\or C\or D\or E\or F%
460   \fi
461 }
```

## 2.8.2 Character code to octal number

\PE@OctChar

```
462 \ifPE@etex
463   \def\PE@OctChar#1{%
464     \expandafter\PE@@OctChar
465     \the\numexpr\dimexpr.015625\dimexpr'#1sp\relax\relax
466     \expandafter\relax
467     \expandafter\relax
468     \the\numexpr\dimexpr.125\dimexpr'#1sp\relax\relax\relax
469     \relax
470     #1%
471   }%
472   \def\PE@@OctChar#1\relax#2\relax#3{%
473     \PE@backslash
474     #1%
475     \the\numexpr#2-8*#1\relax
476     \the\numexpr\dimexpr'#3sp\relax-8*#2\relax
477   }%
478 \else
479   \def\PE@OctChar#1{%
480     \dimen0='#1sp%
481     \dimen2=.125\dimen0 %
482     \dimen4=.125\dimen2 %
483     \advance\dimen0-8\dimen2 %
484     \advance\dimen2-8\dimen4 %
485     \edef\PE@result{%
486       \PE@result
487       \PE@backslash
488       \number\dimen4 %
489       \number\dimen2 %
490       \number\dimen0 %
491     }%
492   }%
493 \fi
```

## 2.8.3 Unpack hex string

\PE@UnescapeHex

```
494 \def\PE@UnescapeHex#1{%
495   \begingroup
496     \PE@InitUccodeHexDigit
497     \def\PE@result{%
498       \expandafter\PE@DeHex#1\relax\relax
499     \expandafter\endgroup
500     \expandafter\def\expandafter#1\expandafter{\PE@result}%
501 }
```

\PE@DeHex

```
502 \def\PE@DeHex#1#2{%
503   \ifx#2\relax
504     \ifx#1\relax
```

```

505     \let\PE@next\relax
506   \else
507     \uppercase{%
508       \def\PE@testA{#1}%
509     }%
510     \ifcase\expandafter\PE@TestUcHexDigit\PE@testA
511       \def\PE@next{%
512         \PE@DeHex#10\relax\relax
513       }%
514     \else
515       \let\PE@next\relax
516     \fi
517   \fi
518 \else
519   \uppercase{%
520     \def\PE@testA{#1}%
521     \def\PE@testB{#2}%
522   }%
523   \ifcase\expandafter\PE@TestUcHexDigit\PE@testA
524     \ifcase\expandafter\PE@TestUcHexDigit\PE@testB
525       \uccode\z@="\PE@testA\PE@testB\relax
526       \ifnum\uccode\z@=32 %
527         \let\PE@temp\PE@space@space
528       \else
529         \uppercase{%
530           \def\PE@temp{^^@}%
531         }%
532       \fi
533       \edef\PE@result{\PE@result\PE@temp}%
534       \let\PE@next\PE@DeHex
535     \else
536       % invalid input sequence
537       \def\PE@next{%
538         \PE@DeHex#1%
539       }%
540     \fi
541   \else
542     % invalid input sequence
543     \def\PE@next{\PE@DeHex#2}%
544   \fi
545 \fi
546 \PE@next
547 }

```

#### 2.8.4 Conversion to PDF name

\PE@EscapeName

```

548 \ifPE@etex
549   \def\PE@EscapeName#1{%
550     \edef#1{\expandafter\PE@EscapeNameTokens#1\relax}%
551   }%
552 \else
553   \def\PE@EscapeName#1{%
554     \def\PE@result{}%
555     \expandafter\PE@EscapeNameTokens#1\relax
556     \let#1\PE@result
557   }%
558 \fi

```

\PE@EscapeNameTokens

```

559 \def\PE@EscapeNameTokens#1{%
560   \ifx\relax#1%

```



617 \fi

\PE@EscapeStringTokens

```
618 \def\PE@EscapeStringTokens#1{%
619   \ifx\relax#1%
620   \else
621     \ifnum'#1<33 %
622       \PE@OctChar#1%
623     \else
624       \ifnum'#1>126 %
625         \PE@OctChar#1%
626       \else \ifnum'#1=40 \PE@EscapeStringAdd{\string\}% (
627         \else\ifnum'#1=41 \PE@EscapeStringAdd{\string\)}% )
628         \else\ifnum'#1=92 \PE@EscapeStringAdd{\string\\}% \
629         \else
630           \PE@EscapeStringAdd{#1}%
631         \fi\fi\fi
632       \fi
633     \fi
634   \expandafter\PE@EscapeStringTokens
635 \fi
636 }%
```

\PE@EscapeStringAdd

```
637 \ifPE@etex
638 \def\PE@EscapeStringAdd#1{#1}%
639 \else
640 \def\PE@EscapeStringAdd#1{%
641   \edef\PE@result{%
642     \PE@result
643     #1%
644   }%
645 }%
646 \fi

647 \PE@AtEnd
648 \end{package}
```

### 3 Test

```
649 \test1 | test2 | test3
650 \NeedsTeXFormat{LaTeX2e}
651 \makeatletter
```

#### 3.1 Test with \pdfescape... commands

```
652 \test1
653 \ProvidesFile{pdfescape-test1.tex}%
654   [2007/04/21 v1.4 Test with \string\pdfescape... commands]%
655 \test1
```

#### 3.2 Test without \pdfescape..., with $\varepsilon$ -TeX

```
656 \test2
657 \ProvidesFile{pdfescape-test2.tex}%
658   [2007/04/21 v1.4 Test without \string\pdfescape..., with  $\varepsilon$ -TeX]%
659 \test2
```

#### 3.3 Test without \pdfescape... and $\varepsilon$ -TeX

```
660 \test3
661 \ProvidesFile{pdfescape-test3.tex}%
662   [2007/04/21 v1.4 Test without \string\pdfescape... and  $\varepsilon$ -TeX]%
```

```
663 </test3>
```

### 3.4 Check/ensure test preconditions

#### 3.4.1 Check \pdfescape...

```
664 <*test1>
665 \@ifundefined{pdfescapehex}{%
666   \PackageError{pdfescape-test1}{%
667     Missing \string\pdfescape... commands%
668   }{Test aborted.}%
669   \stop
670 }{}
671 </test1>

672 <*test2 | test3>
673 \let\pdfescapehex\@undefined
674 \let\pdfunescapehex\@undefined
675 \let\pdfescapename\@undefined
676 \let\pdfescapestring\@undefined
677 </test2 | test3>
```

#### 3.4.2 Check $\varepsilon$ -TeX

```
678 <*test2>
679 \@ifundefined{numexpr}{%
680   \PackageError{pdfescape-test2}{%
681     Missing \eTeX
682   }{Test aborted.}%
683   \stop
684 }{}
685 </test2>
```

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

### 3.5 Common part

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

```
686 \RequirePackage{qstest}
687 \IncludeTests{*}
688 \LogTests{log}{*}{*}
689
690 \newcommand*{\ExpectVar}[2]{%
691   \ifx#1#2%
692   \else
693     \begingroup
694       \@onelevel@sanitize#1%
695       \@onelevel@sanitize#2%
696       \typeout{[#1] <> [#2]}% hash-ok
697     \endgroup
698   \fi
699   \Expect*{\ifx#1#2true\else false\fi}{true}%
700 }
701
702 \makeatletter
703 \begingroup
704   \gdef\AllBytes{}%
705   \count@=0 %
706   \catcode@=12 %
707   \@whilenum\count@<256 \do{%
708     \lccode@=\count@
709     \ifnum\count@=32 %
710       \xdef\AllBytes{\AllBytes\space}%
711     \else
```



```

712     \lowercase{%
713     \xdef\AllBytes{\AllBytes^^@}%
714     }%
715     \fi
716     \advance\count@ by 1 %
717 }%
718 \endgroup
719 \newcommand*{\AllBytesHex}{%
720 000102030405060708090A0B0C0D0E0F%
721 101112131415161718191A1B1C1D1E1F%
722 202122232425262728292A2B2C2D2E2F%
723 303132333435363738393A3B3C3D3E3F%
724 404142434445464748494A4B4C4D4E4F%
725 505152535455565758595A5B5C5D5E5F%
726 606162636465666768696A6B6C6D6E6F%
727 707172737475767778797A7B7C7D7E7F%
728 808182838485868788898A8B8C8D8E8F%
729 909192939495969798999A9B9C9D9E9F%
730 A0A1A2A3A4A5A6A7A8A9AABACADAFAF%
731 B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF%
732 C0C1C2C3C4C5C6C7C8C9CACCCDCFCF%
733 D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF%
734 E0E1E2E3E4E5E6E7E8E9EAECEDEEEF%
735 F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF%
736 }
737 \@onelevel@sanitize\AllBytesHex
738 \expandafter\lowercase\expandafter{%
739 \expandafter\newcommand\expandafter*\expandafter\AllBytesHexLC
740 \expandafter{\AllBytesHex}%
741 }
742 \newcommand*{\AllBytesName}{%
743 \begingroup
744 \catcode'\# =12 %
745 \xdef\AllBytesName{%
746 #01#02#03#04#05#06#07#08#09#0A#0B#0C#0D#0E#0F%
747 #10#11#12#13#14#15#16#17#18#19#1A#1B#1C#1D#1E#1F%
748 #20!"#23$#25&'#28#29*+,-.#2F%
749 0123456789:;#3C=#3E?%
750 @ABCDEFGHIJKLMNO%
751 PQRSTUVWXYZ#5B\@backslashchar#5D^_%
752 'abcdefghijklmnop%
753 pqrstuvwxyz#7B|#7D\string~#7F%
754 #80#81#82#83#84#85#86#87#88#89#8A#8B#8C#8D#8E#8F%
755 #90#91#92#93#94#95#96#97#98#99#9A#9B#9C#9D#9E#9F%
756 #A0#A1#A2#A3#A4#A5#A6#A7#A8#A9#AA#AB#AC#AD#AE#AF%
757 #B0#B1#B2#B3#B4#B5#B6#B7#B8#B9#BA#BB#BC#BD#BE#BF%
758 #C0#C1#C2#C3#C4#C5#C6#C7#C8#C9#CA#CB#CC#CD#CE#CF%
759 #D0#D1#D2#D3#D4#D5#D6#D7#D8#D9#DA#DB#DC#DD#DE#DF%
760 #E0#E1#E2#E3#E4#E5#E6#E7#E8#E9#EA#EB#EC#ED#EE#EF%
761 #F0#F1#F2#F3#F4#F5#F6#F7#F8#F9#FA#FB#FC#FD#FE#FF%
762 }%
763 \endgroup
764 \@onelevel@sanitize\AllBytesName
765
766 \newcommand*{\AllBytesString}{%
767 \begingroup
768 \def\|{|}%
769 \edef\%{\@percentchar}%
770 \catcode'\| =0 %
771 \catcode'\# =12 %
772 \catcode'\~ =12 %
773 \catcode'\ =12 %

```

```

774 |xdef|AllBytesString{%
775     \000\001\002\003\004\005\006\007\010\011\012\013\014\015\016\017%
776     \020\021\022\023\024\025\026\027\030\031\032\033\034\035\036\037%
777     \040!"#$%&'(\)*+,-./%
778     0123456789:;<=>?%
779     @ABCDEFGHJKLMNO%
780     PQRSTUVWXYZ[\]^_%
781     'abcdefghijklmnopqrstuvwxyz%
782     pqrstuvwxyz{|}~\177%
783     \200\201\202\203\204\205\206\207\210\211\212\213\214\215\216\217%
784     \220\221\222\223\224\225\226\227\230\231\232\233\234\235\236\237%
785     \240\241\242\243\244\245\246\247\250\251\252\253\254\255\256\257%
786     \260\261\262\263\264\265\266\267\270\271\272\273\274\275\276\277%
787     \300\301\302\303\304\305\306\307\310\311\312\313\314\315\316\317%
788     \320\321\322\323\324\325\326\327\330\331\332\333\334\335\336\337%
789     \340\341\342\343\344\345\346\347\350\351\352\353\354\355\356\357%
790     \360\361\362\363\364\365\366\367\370\371\372\373\374\375\376\377%
791 }%
792 |endgroup
793 \@onelevel@sanitize\AllBytesString
794
795 <*test3>
796 \let\org@detokenize\detokenize
797 \let\detokenize\@undefined
798 \let\org@numexpr\numexpr
799 \let\numexpr\@undefined
800 </test3>
801 \RequirePackage{pdfescape}
802 <*test3>
803 \let\detokenize\org@detokenize
804 \let\numexpr\org@numexpr
805 </test3>
806
807 \begin{qstest}{all-hex}{\AllBytes, escapehex}
808   \EdefEscapeHex\x{\AllBytes}%
809   \Expect*{\x}*{\AllBytesHex}%
810   \ExpectVar\x\AllBytesHex
811 \end{qstest}
812
813 \begin{qstest}{all-unhex}{\AllBytesHex, unescapehex}
814   \EdefUnescapeHex\x{\AllBytesHex}%
815   \Expect*{\x}*{\AllBytes}%
816   \ExpectVar\x\AllBytes
817 \end{qstest}
818
819 \begin{qstest}{all-unhex-lc}{\AllBytesHexLC, unescapehex, lowercase}
820   \EdefUnescapeHex\x{\AllBytesHexLC}%
821   \Expect*{\x}*{\AllBytes}%
822   \ExpectVar\x\AllBytes
823 \end{qstest}
824
825 \begin{qstest}{unhex-incomplete}{unescapehex, incomplete}
826   \EdefUnescapeHex\x{4}%
827   \Expect*{\x}{@}%
828 \end{qstest}
829
830 \begin{qstest}{unhex-space}{unescapehex, space}
831   \EdefUnescapeHex\x{20}%
832   \Expect*{\x}{ }%
833   \ExpectVar\x\space
834 \end{qstest}
835

```

```

836 \begin{qstest}{unhex-spaces}{unescapehex, spaces}
837   \EdefUnescapeHex\x{204020204120}%
838   \def\y#1{%
839     \edef\z{#1\string @#1#1\string A#1}%
840   }\y{ }%
841   \Expect*{\x}*{\z}%
842   \ExpectVar\x\z
843 \end{qstest}
844
845 \begin{qstest}{unhex-hash}{unescapehex, hash}
846   \catcode'\#=12 %
847   \EdefUnescapeHex\x{#20}%
848   \ExpectVar\x\space
849 \end{qstest}
850
851 \begin{qstest}{unhex-invalid}{unescapehex, invalid}
852   \def\test#1#2{%
853     \EdefUnescapeHex\x{#1}%
854     \edef\y{#2}%
855     \@onelevel@sanitize\y
856     \ExpectVar\x\y
857   }%
858   \langle *test1\rangle
859   \edef\x{\pdfunescapehex{4X}}%
860   \edef\y{\string @}%
861   \ifx\x\y
862   \else
863     \def~{\space}%
864     \typeout{*****}%
865     \typeout{* Your pdfTeX contains bug 777.~~~*}%
866     \typeout{* This test is redefined as dummy, *}%
867     \typeout{* because it triggers the bug.~~~~*}%
868     \typeout{*****}%
869     \def\test#1#2{%
870       \fi
871     }\test1\rangle
872     \test{X}{}%
873     \test{XY}{}%
874     \test{XYZ}{}%
875     \test{A}{~^a0}%
876     \test{AX}{~^a0}%
877     \test{XA}{~^a0}%
878     \test{XXAXX}{~^a0}%
879 \end{qstest}
880
881 \begin{qstest}{all-name}{\AllBytes, escapename}
882   \EdefEscapeName\x{\AllBytes}%
883   \Expect*{\x}*{\AllBytesName}%
884   \ExpectVar\x\AllBytesName
885 \end{qstest}
886
887 \begin{qstest}{all-string}{\AllBytes, escapestring}
888   \EdefEscapeString\x{\AllBytes}%
889   \Expect*{\x}*{\AllBytesString}%
890   \ExpectVar\x\AllBytesString
891 \end{qstest}
892
893 \begin{qstest}{uchexdigit}{unescape, uppercase hex digit}
894   \catcode'\@=11 %
895   \catcode0=12 %
896   \def\test#1#2{%
897     \uccode0=#1\relax

```

```

898 \uppercase{%
899 \def\x{^^@}%
900 }%
901 \Expect*{%
902 \ifcase\expandafter\PE@TestUcHexDigit\x
903 true%
904 \else
905 false%
906 \fi
907 }{#2}%
908 }%
909 \def\range#1#2#3{%
910 \count0=#1\relax
911 \loop
912 \ifnum\count0<#2\relax
913 \test{\count0}{#3}%
914 \advance\count0 by 1 %
915 \repeat
916 }%
917 \range{0}{47}{false}%
918 \range{48}{57}{true}%
919 \range{58}{64}{false}%
920 \range{65}{70}{true}%
921 \range{71}{255}{false}%
922 \end{qstest}
923
924 \begin{qstest}{unescapename}{unescapename}
925 \def\test#1#2{%
926 \EdefUnescapeName\x{#1}%
927 \edef\y{#2}%
928 \@onelevel@sanitize\y
929 \ExpectVar\x\y
930 }%
931 \catcode'\#=12 %
932 \catcode0=12 %
933 \test{}{}%
934 \test{x}{x}%
935 \test{xy}{xy}%
936 \test{#}{#}%
937 \test{##}{##}%
938 \test{###}{###}%
939 \test{####}{####}%
940 \test{#x}{#x}%
941 \test{#xy}{#xy}%
942 \test{#1}{#1}%
943 \test{#40}{@}%
944 \test{#400}{@0}%
945 \test{#4x0}{#4x0}%
946 \test{#ab}{^^ab}%
947 \test{#00}{^^@}%
948 \test{x#40y#40z}{x@y@z}%
949 \test{#40#40#40#40}{@@@@}%
950 \test{a#x}{a#x}%
951 \test{a#xy}{a#xy}%
952 \test{a#1}{a#1}%
953 \test{a#40}{a@}%
954 \test{a#400}{a@0}%
955 \test{#20}{ }%
956 \test{a#20}{a }%
957 \test{a#20b}{a b}%
958 \test{a#20#20#20b}{a \space\space b}%
959 \end{qstest}

```

```

960
961 \begin{qstest}{unescapestring}{unescapestring}
962   \def\test#1#2{%
963     \EdefUnescapeString\x{#1}%
964     \edef\y{#2}%
965     \@onelevel@sanitize\y
966     \ExpectVar\x\y
967   }%
968   \catcode0=12 %
969   \def\DefChar#1#2{%
970     \begingroup
971       \uccode0=#2\relax
972       \uppercase{\endgroup
973         \def#1{^^@}%
974       }%
975   }%
976   \DefChar\nul{0}%
977   \DefChar\one{1}%
978   \DefChar\bel{8}%
979   \DefChar\tab{9}%
980   \DefChar\lf{10}%
981   \DefChar\ff{12}%
982   \DefChar\cr{13}%
983   \DefChar\{\92}%
984   \test{}{}%
985   \test{a}{a}%
986   \test{\}{}%
987   \test{\\}{}%
988   \test{\\y}{\y}%
989   \test{\\000}{\nul}%
990   \test{\\b}{\bel}%
991   \test{\\t}{\tab}%
992   \test{\\n}{\lf}%
993   \test{\\f}{\ff}%
994   \test{\\r}{\cr}%
995   \test{\\}{}%
996   \test{\\}{}%
997   \test{\\040}{ }%
998   \test{\\100}{@}%
999   \test{\\40}{ }%
1000  \test{\\1}{\one}%
1001  \test{\\01}{\one}%
1002  \test{\\001}{\one}%
1003  \test{\\18}{\one8}%
1004  \test{\\018}{\one8}%
1005  \test{\\0018}{\one8}%
1006  \test{x\\}{x}%
1007  \test{x\\\\}{x\\}%
1008  \test{x\\y}{x\y}%
1009  \test{x\\000}{x\nul}%
1010  \test{x\\b}{x\bel}%
1011  \test{x\\t}{x\tab}%
1012  \test{x\\n}{x\lf}%
1013  \test{x\\f}{x\ff}%
1014  \test{x\\r}{x\cr}%
1015  \test{x\\}{x}%
1016  \test{x\\}{x}%
1017  \test{x\\040}{x }%
1018  \test{x\\100}{x@}%
1019  \test{x\\40}{x }%
1020  \test{x\\1}{x\one}%
1021  \test{x\\01}{x\one}%

```

```

1022 \test{x\001}{x\one}%
1023 \test{x\18}{x\one8}%
1024 \test{x\018}{x\one8}%
1025 \test{x\0018}{x\one8}%
1026 \test{\b\t\n\f\r\(\)\000\040}{%
1027 \bel\tab\lf\ff\cr()\nul\space
1028 }%
1029 \test{\\lf}{}%
1030 \test{x\\lf}{x}%
1031 \test{\cr}{\lf}%
1032 \test{\cr\lf}{\lf}%
1033 \test{\lf}{\lf}%
1034 \test{\lf\cr}{\lf\lf}%
1035 \test{x\cr}{x\lf}%
1036 \test{x\cr\lf}{x\lf}%
1037 \test{x\lf}{x\lf}%
1038 \test{x\lf\cr}{x\lf\lf}%
1039 \test{x\\cr\lf y\cr}{xy\lf}%
1040 \end{qstest}
1041 \stop
1042 </test1 | test2 | test3>

```

## 4 Installation

### 4.1 Download

**Package.** This package is available on CTAN<sup>1</sup>:

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

*TDS* refers to the standard “A Directory Structure for T<sub>E</sub>X 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/

```

---

<sup>1</sup><http://ftp.ctan.org/tex-archive/>

### 4.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain- $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ :

```
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
pdfescape-test1.tex → doc/latex/oberdiek/pdfescape-test1.tex
pdfescape-test2.tex → doc/latex/oberdiek/pdfescape-test2.tex
pdfescape-test3.tex → doc/latex/oberdiek/pdfescape-test3.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  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  distribution (`te $\mathrm{T}_{\mathrm{E}}\mathrm{X}$` , `mik $\mathrm{T}_{\mathrm{E}}\mathrm{X}$` , ...) relies on file name databases, you must refresh these. For example, `te $\mathrm{T}_{\mathrm{E}}\mathrm{X}$`  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 pdfescape.pdf unpack_files output .
```

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

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

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

If you insist on using  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$  for `docstrip` (really, `docstrip` does not need  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ), 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 `pdf $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$` :

```
pdfflatex pdfescape.dtx
makeindex -s gind.ist pdfescape.idx
pdfflatex pdfescape.dtx
makeindex -s gind.ist pdfescape.idx
pdfflatex pdfescape.dtx
```

## 5 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.

## 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	
\# . . .	137, 273, 298, 744, 771, 846, 931
\\$ . . . . .	136
\% . . . . .	769
\( . . . . .	626, 777
\) . . . . .	627, 777
\@ . . . . .	894
@ReturnAfterFi . . . . .	123, <u>128</u>
@backslashchar . . . . .	751
@ifundefined . . . . .	665, 679
\@ne ..	164, 167, 212, 215, 219, 329, 332
\@onelevel@sanitize . . . . .	96, 694,
	695, 737, 764, 793, 855, 928, 965
\@percentchar . . . . .	769
\@undefined	673, 674, 675, 676, 797, 799
\@whilenum . . . . .	707
\\ . . . . .	120, 265, 409,
	628, 773, 780, 983, 986, 987,
	988, 989, 990, 991, 992, 993,
	994, 995, 996, 997, 998, 999,
	1000, 1001, 1002, 1003, 1004,
	1005, 1006, 1007, 1008, 1009,
	1010, 1011, 1012, 1013, 1014,
	1015, 1016, 1017, 1018, 1019,
	1020, 1021, 1022, 1023, 1024,
	1025, 1026, 1027, 1029, 1030, 1039
\}	. . . . . 322
\	. . . . . 264, 269, 768, 770
\~	. . . . . 772
Numbers	
\0 . . . . .	775, 776, 777
\1 . . . . .	782
\2 . . . . .	783, 784, 785, 786
\3 . . . . .	787, 788, 789, 790
\8 . . . . .	237
\9 . . . . .	238



<code>\_</code> . . . . .	112, 628	<b>G</b>	<code>\gdef</code> . . . . .	138, 146, 704
<b>A</b>		<b>I</b>		
<code>\advance</code> . . . . .	447, 483, 484, 716, 914	<code>\ifcase</code> . . . . .	33, 160, 161, 339, 347, 457, 510, 523, 524, 563, 902	
<code>\AllBytes</code> . . . . .	704, 710, 713, 807, 808, 815, 816, 821, 822, 881, 882, 887, 888	<code>\ifnum</code> . . . . .	211, 214, 217, 218, 328, 331, 526, 562, 570, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 621, 624, 626, 627, 628, 709, 912	
<code>\AllBytesHex</code> . . . . .	719, 737, 740, 809, 810, 813, 814	<code>\ifPE@etex</code> . . . . .	412, 418, 436, 462, 548, 595, 605, 637	
<code>\AllBytesHexLC</code> . . . . .	739, 819, 820	<code>\ifx</code> . . . . .	34, 36, 42, 63, 71, 80, 90, 120, 147, 151, 246, 251, 362, 414, 430, 503, 504, 560, 619, 691, 699, 861	
<code>\AllBytesName</code> . . . . .	742, 745, 764, 883, 884	<code>\immediate</code> . . . . .	44, 65	
<code>\AllBytesString</code> . . . . .	766, 793, 889, 890	<code>\IncludeTests</code> . . . . .	687	
<b>B</b>		<b>L</b>		
<code>\begin</code> . . . . .	807, 813, 819, 825, 830, 836, 845, 851, 881, 887, 893, 924, 961	<code>\lccode</code> . . . . .	708	
<code>\bel</code> . . . . .	978, 990, 1010, 1027	<code>\lf</code> . . . . .	980, 992, 1012, 1027, 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036, 1037, 1038, 1039	
<b>C</b>		<code>\LogTests</code> . . . . .	688	
<code>\catcode</code> . . . . .	3, 5, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 27, 28, 29, 30, 31, 55, 56, 57, 58, 59, 60, 61, 62, 112, 136, 137, 264, 265, 706, 744, 770, 771, 772, 773, 846, 894, 895, 931, 932, 968	<code>\loop</code> . . . . .	911	
<code>\count</code> . . . . .	910, 912, 913, 914	<code>\lowercase</code> . . . . .	712, 738	
<code>\count@</code> . . . . .	705, 707, 708, 709, 716	<b>M</b>		
<code>\cr</code> . . . . .	982, 994, 1014, 1027, 1031, 1032, 1034, 1035, 1036, 1038, 1039	<code>\makeatletter</code> . . . . .	651, 702	
<code>\csname</code> . . . . .	2, 32, 42, 63, 76, 80, 83, 90, 101, 362, 385, 414	<code>\meaning</code> . . . . .	92	
<b>D</b>		<b>N</b>		
<code>\DefChar</code> . . . . .	969, 976, 977, 978, 979, 980, 981, 982, 983	<code>\NeedsTeXFormat</code> . . . . .	650	
<code>\detokenize</code> . . . . .	108, 796, 797, 803	<code>\newcommand</code> . . . . .	690, 719, 739, 742, 766	
<code>\dimen</code> . . . . .	445, 446, 447, 450, 451, 480, 481, 482, 483, 484, 488, 489, 490	<code>\newif</code> . . . . .	412	
<code>\dimexpr</code> . . . . .	438, 440, 465, 468, 476	<code>\nul</code> . . . . .	976, 989, 1009, 1027	
<code>\do</code> . . . . .	707	<code>\number</code> . . . . .	488, 489, 490	
<b>E</b>		<code>\numexpr</code> . . . . .	438, 440, 465, 468, 475, 476, 798, 799, 804	
<code>\EdefEscapeHex</code> . . . . .	2, 363, 390, 808	<b>O</b>		
<code>\EdefEscapeName</code> . . . . .	372, 396, 882	<code>\one</code> . . . . .	977, 1000, 1001, 1002, 1003, 1004, 1005, 1020, 1021, 1022, 1023, 1024, 1025	
<code>\EdefEscapeString</code> . . . . .	377, 399, 888	<code>\org@detokenize</code> . . . . .	796, 803	
<code>\EdefUnescapeHex</code> . . . . .	368, 393, 814, 820, 826, 831, 837, 847, 853	<code>\org@numexpr</code> . . . . .	798, 804	
<code>\EdefUnescapeName</code> . . . . .	2, 129, 926	<b>P</b>		
<code>\EdefUnescapeString</code> . . . . .	2, 229, 963	<code>\PackageError</code> . . . . .	666, 680	
<code>\empty</code> . . . . .	36	<code>\PackageInfo</code> . . . . .	47	
<code>\end</code> . . . . .	811, 817, 823, 828, 834, 843, 849, 879, 885, 891, 922, 959, 1040	<code>\pdfescape</code> . . . . .	654, 658, 662, 667	
<code>\endcsname</code> . . . . .	2, 32, 42, 63, 76, 80, 83, 90, 101, 362, 385, 414	<code>\pdfescapehex</code> . . . . .	391, 673	
<code>\endinput</code> . . . . .	51, 403	<code>\pdfescapeiname</code> . . . . .	397, 675	
<code>\escapechar</code> . . . . .	407	<code>\pdfescapestring</code> . . . . .	400, 676	
<code>\eTeX</code> . . . . .	681	<code>\pdfunescapehex</code> . . . . .	394, 674, 859	
<code>\Expect</code> . . . . .	699, 809, 815, 821, 827, 832, 841, 883, 889, 901	<code>\PE@@NormalizeLineEnd</code> . . . . .	242, 245	
<code>\ExpectVar</code> . . . . .	690, 810, 816, 822, 833, 842, 848, 856, 884, 890, 929, 966	<code>\PE@@OctChar</code> . . . . .	464, 472	
<b>F</b>		<code>\PE@AtEnd</code> . . . . .	7, 8, 402, 647	
<code>\ff</code> . . . . .	981, 993, 1013, 1027	<code>\PE@backslash</code> . . . . .	406, 473, 487	
		<code>\PE@CheckEndBackslash</code> . . . . .	319	
		<code>\PE@DeHex</code> . . . . .	498, 502	
		<code>\PE@DeName</code> . . . . .	142, 146, 175, 179	

\PE@DeString	273, 359	\repeat	915
\PE@edefbabel	383, 391, 394, 397, 400	\RequirePackage	686, 801
\PE@EnsureCode	6, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25	<b>S</b>	
\PE@EscapeHex	366, 418	\space	710, 833, 848, 863, 958, 1027
\PE@EscapeName	375, 548	\stop	669, 683, 1041
\PE@EscapeNameAdd	566, 571, 585, 593, 595	<b>T</b>	
\PE@EscapeNameHashChar	573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 592	\tab	979, 991, 1011, 1027
\PE@EscapeNameTokens	550, 555, 559	\test	852, 869, 872, 873, 874, 875, 876, 877, 878, 896, 913, 925, 933, 934, 935, 936, 937, 938, 939, 940, 941, 942, 943, 944, 945, 946, 947, 948, 949, 950, 951, 952, 953, 954, 955, 956, 957, 958, 962, 984, 985, 986, 987, 988, 989, 990, 991, 992, 993, 994, 995, 996, 997, 998, 999, 1000, 1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008, 1009, 1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017, 1018, 1019, 1020, 1021, 1022, 1023, 1024, 1025, 1026, 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036, 1037, 1038, 1039
\PE@EscapeString	380, 605	\the	3, 9, 465, 468, 475, 476
\PE@EscapeStringAdd	626, 627, 628, 630, 637	\typeout	696, 864, 865, 866, 867, 868
\PE@EscapeStringTokens	607, 613, 618	<b>U</b>	
\PE@etexttrue	416	\uccode	23, 24, 25, 169, 173, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 237, 238, 355, 525, 526, 897, 971
\PE@hash	405, 566, 571, 593	\uppercase	156, 170, 260, 356, 507, 519, 529, 898, 972
\PE@HexChar	432, 436, 567, 572	<b>W</b>	
\PE@HexDigit	438, 439, 450, 451, 455	\write	44, 65
\PE@InitUccodeHexDigit	140, 186, 496	<b>X</b>	
\PE@next	149, 154, 175, 179, 183, 248, 252, 254, 257, 340, 342, 344, 348, 350, 352, 505, 511, 515, 534, 537, 543, 546	\x	32, 34, 36, 43, 47, 49, 64, 69, 76, 113, 239, 261, 408, 411, 808, 809, 810, 814, 815, 816, 820, 821, 822, 826, 827, 831, 832, 833, 837, 841, 842, 847, 848, 853, 856, 859, 861, 882, 883, 884, 888, 889, 890, 899, 902, 926, 929, 963, 966
\PE@NormalizeLineEnd	232, 240	<b>Y</b>	
\PE@OctAll	342, 348, 350, 354	\y	838, 840, 854, 855, 856, 860, 861, 927, 928, 929, 964, 965, 966
\PE@OctChar	462, 622, 625	<b>Z</b>	
\PE@OctI	338	\z	839, 841, 842
\PE@OctII	340, 346	\z@	162, 169, 173, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 221, 224, 334, 355, 525, 526
\PE@onelevel@sanitize	85, 91, 96, 103, 107, 133, 234	<b>R</b>	
\PE@result	141, 144, 148, 153, 174, 178, 241, 243, 247, 250, 357, 424, 426, 448, 449, 485, 486, 497, 500, 533, 554, 556, 599, 600, 612, 615, 641, 642	\range	909, 917, 918, 919, 920, 921
\PE@sanitize	79, 130, 230, 364, 369, 373, 378	<b>R</b>	
\PE@SanitizeSpaceOther	115, 131, 231, 365, 374, 379	<b>R</b>	
\PE@space@other	111, 122	<b>R</b>	
\PE@space@space	114, 527	<b>R</b>	
\PE@SpaceToOther	116, 118	<b>R</b>	
\PE@strip@prefix	92, 94	<b>R</b>	
\PE@temp	171, 174, 527, 530, 533	<b>R</b>	
\PE@testA	157, 160, 169, 508, 510, 520, 523, 525	<b>R</b>	
\PE@testB	158, 161, 169, 521, 524, 525	<b>R</b>	
\PE@TestOctDigit	327, 339, 347	<b>R</b>	
\PE@TestUcHexDigit	160, 161, 210, 510, 523, 524, 902	<b>R</b>	
\PE@ToHex	420, 425, 429	<b>R</b>	
\PE@UnescapeHex	370, 494	<b>R</b>	
\PE@UnescapeName	132, 135	<b>R</b>	
\PE@UnescapeString	233, 266	<b>R</b>	
\ProvidesFile	653, 657, 661	<b>R</b>	
\ProvidesPackage	77	<b>R</b>	