

The pdfescape package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2007/09/09 v1.6

Abstract

This package implements pdf_{TEX}'s escape features (`\pdfescapehex`, `\pdfunescapehex`, `\pdfescapename`, `\pdfescapestring`) using $\text{T}_{\text{E}}\text{X}$ or $\varepsilon\text{-T}_{\text{E}}\text{X}$.

Contents

1	Documentation	2
1.1	Additional unescape macros	2
1.2	Sanitizing macro	2
2	Implementation	3
2.1	Reload check and package identification	3
2.2	Catcodes	4
2.3	Sanitizing	4
2.3.1	Space characters	5
2.3.2	Space normalization	5
2.4	<code>\EdefUnescapeName</code>	5
2.5	<code>\EdefUnescapeString</code>	7
2.6	User macros (pdf _{TEX} analogues)	10
2.7	Help macros	11
2.7.1	Characters	11
2.7.2	Switch for $\varepsilon\text{-T}_{\text{E}}\text{X}$	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	13
2.8.4	Conversion to PDF name	14
2.8.5	Conversion to PDF string	15
3	Test	15
3.1	Catcode checks for loading	15
3.2	Macro tests	16
3.3	Test with <code>\pdfescape...</code> commands	16
3.4	Test without <code>\pdfescape...</code> , with $\varepsilon\text{-T}_{\text{E}}\text{X}$	16
3.5	Test without <code>\pdfescape...</code> and $\varepsilon\text{-T}_{\text{E}}\text{X}$	16
3.6	Check/ensure test preconditions	17
3.6.1	Check <code>\pdfescape...</code>	17
3.6.2	Check $\varepsilon\text{-T}_{\text{E}}\text{X}$	17
3.7	Common part	17
4	Installation	23
4.1	Download	23
4.2	Bundle installation	23
4.3	Package installation	24
4.4	Refresh file name databases	24
4.5	Some details for the interested	24

5 History	25
[2007/02/21 v1.0]	25
[2007/02/25 v1.1]	25
[2007/03/20 v1.2]	25
[2007/04/11 v1.3]	25
[2007/04/21 v1.4]	25
[2007/08/27 v1.5]	25
[2007/09/09 v1.6]	25
6 Index	25

1 Documentation

<code>\EdefEscapeHex {⟨cmd⟩} {⟨string⟩}</code> <code>\EdefUnescapeHex {⟨cmd⟩} {⟨string⟩}</code> <code>\EdefEscapeName {⟨cmd⟩} {⟨string⟩}</code> <code>\EdefEscapeString {⟨cmd⟩} {⟨string⟩}</code>
--

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

<code>\EdefUnescapeName {⟨cmd⟩} {⟨string⟩}</code>

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).

<code>\EdefUnescapeString {⟨cmd⟩} {⟨string⟩}</code>

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

<code>\EdefSanitize {⟨cmd⟩} {⟨string⟩}</code>

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 <*package>
```

2.1 Reload check and package identification

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

```
2 \begingroup
3 \catcode44 12 % ,
4 \catcode45 12 % -
5 \catcode46 12 % .
6 \catcode58 12 % :
7 \catcode64 11 % @
8 \expandafter\let\expandafter\x\csname ver@pdfescape.sty\endcsname
9 \ifcase 0%
10 \ifx\x\relax % plain
11 \else
12 \ifx\x\empty % LaTeX
13 \else
14 1%
15 \fi
16 \fi
17 \else
18 \expandafter\ifx\csname PackageInfo\endcsname\relax
19 \def\x#1#2{%
20 \immediate\write-1{Package #1 Info: #2.}%
21 }%
22 \else
23 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
24 \fi
25 \x{pdfescape}{The package is already loaded}%
26 \endgroup
27 \expandafter\endinput
28 \fi
29 \endgroup
```

Package identification:

```
30 \begingroup
31 \catcode40 12 % (
32 \catcode41 12 % )
33 \catcode44 12 % ,
34 \catcode45 12 % -
35 \catcode46 12 % .
36 \catcode47 12 % /
37 \catcode58 12 % :
38 \catcode64 11 % @
39 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
40 \def\x#1#2#3[#4]{\endgroup
41 \immediate\write-1{Package: #3 #4}%
42 \xdef#1{#4}%
43 }%
44 \else
45 \def\x#1#2[#3]{\endgroup
46 #2[#{#3}]%
47 \ifx#1\relax
48 \xdef#1{#3}%
49 \fi
50 }%
51 \fi
52 \expandafter\x\csname ver@pdfescape.sty\endcsname
53 \ProvidesPackage{pdfescape}%
54 [2007/09/09 v1.6 Provides hex, PDF name and string conversions (H0)]
```

2.2 Catcodes

```

55 \expandafter\edef\csname PE@AtEnd\endcsname{%
56   \catcode64 \the\catcode64\relax
57 }
58 \catcode64 11 % @
59 \def\TMP@EnsureCode#1#2#3{%
60   \edef\PE@AtEnd{%
61     \PE@AtEnd
62     #1#2 \the#1#2\relax
63   }%
64   #1#2 #3\relax
65 }
66 \TMP@EnsureCode\catcode{0}-{12}% ^^@
67 \TMP@EnsureCode\catcode{34}-{12}% "
68 \TMP@EnsureCode\catcode{39}-{12}% '
69 \TMP@EnsureCode\catcode{42}-{12}% *
70 \TMP@EnsureCode\catcode{45}-{12}% -
71 \TMP@EnsureCode\catcode{46}-{12}% .
72 \TMP@EnsureCode\catcode{60}-{12}% <
73 \TMP@EnsureCode\catcode{61}-{12}% =
74 \TMP@EnsureCode\catcode{62}-{12}% >
75 \TMP@EnsureCode\catcode{94}-{7}% ^
76 \TMP@EnsureCode\catcode{96}-{12}% ‘
77 \TMP@EnsureCode\uccode{34}-{0}% "
78 \TMP@EnsureCode\uccode{48}-{0}% 0
79 \TMP@EnsureCode\uccode{61}-{0}% =

```

2.3 Sanitizing

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

```

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

```

```
111 \fi
```

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

```
112 \let\PE@sanitize\EdefSanitize
```

2.3.1 Space characters

`\PE@space@other`

```
113 \begingroup
114 \catcode'\ =12\relax%
115 \def\x{\endgroup\def\PE@space@other{ }}\x\relax
```

`\PE@space@space`

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

2.3.2 Space normalization

`\PE@SanitizeSpaceOther`

```
117 \def\PE@SanitizeSpaceOther#1{%
118 \edef#1{\expandafter\PE@SpaceToOther#1 \relax}%
119 }
```

`\PE@SpaceToOther`

```
120 \def\PE@SpaceToOther#1 #2\relax{%
121 #1%
122 \ifx\#2\%
123 \else
124 \PE@space@other
125 \@ReturnAfterFi{%
126 \PE@SpaceToOther#2\relax
127 }%
128 \fi
129 }
```

`\@ReturnAfterFi`

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

2.4 \EdefUnescapeName

`\EdefUnescapeName`

```
131 \def\EdefUnescapeName#1#2{%
132 \EdefSanitize#1{#2}%
133 \PE@SanitizeSpaceOther#1%
134 \PE@UnescapeName#1%
135 \PE@onelevel@sanitize#1%
136 }
```

`\PE@UnescapeName`

```
137 \begingroup
138 \catcode'\$=6 % hash
139 \catcode'\#=12 % other
140 \gdef\PE@UnescapeName$1{%
141 \begingroup
142 \PE@InitUccodeHexDigit
143 \def\PE@result{}%
144 \expandafter\PE@DeName$1\relax\relax
145 \expandafter\endgroup
146 \expandafter\def\expandafter$1\expandafter{\PE@result}%
147 }%
```

```

148 \gdef\PE@DeName$1#$2$3{%
149   \ifx\relax$2%
150     \edef\PE@result{\PE@result$1}%
151     \let\PE@next\relax
152   \else
153     \ifx\relax$3%
154       % wrong escape sequence in input
155       \edef\PE@result{\PE@result$1#}%
156       \let\PE@next\relax
157     \else
158       \uppercase{%
159         \def\PE@testA{$2}%
160         \def\PE@testB{$3}%
161       }%
162       \ifcase\ifcase\expandafter\PE@TestUcHexDigit\PE@testA
163         \ifcase\expandafter\PE@TestUcHexDigit\PE@testB
164           \z@
165         \else
166           \@ne
167         \fi
168       \else
169         \@ne
170       \fi
171       \uccode\z@="\PE@testA\PE@testB\relax
172       \uppercase{%
173         \def\PE@temp{^^@}%
174       }%
175       \uccode\z@=\z@
176       \edef\PE@result{\PE@result$1\PE@temp}%
177       \let\PE@next\PE@DeName
178     \else
179       % wrong escape sequence in input
180       \edef\PE@result{\PE@result$1#}%
181       \def\PE@next{\PE@DeName$2$3}%
182     \fi
183   \fi
184 \fi
185 \PE@next
186 }%
187 \endgroup

```

\PE@InitUccodeHexDigit

```

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

```

```

209 \uccode'8=\z@
210 \uccode'9=\z@
211 }

```

\PE@TestUcHexDigit

```

212 \def\PE@TestUcHexDigit#1{%
213 \ifnum'#1<48 % 0
214 \@ne
215 \else
216 \ifnum'#1>70 % F
217 \@ne
218 \else
219 \ifnum'#1>57 % 9
220 \ifnum'#1<65 % A
221 \@ne
222 \else
223 \z@
224 \fi
225 \else
226 \z@
227 \fi
228 \fi
229 \fi
230 }

```

2.5 \EdefUnescapeString

\EdefUnescapeString

```

231 \def\EdefUnescapeString#1#2{%
232 \EdefSanitize#1{#2}%
233 \PE@SanitizeSpaceOther#1%
234 \PE@NormalizeLineEnd#1%
235 \PE@UnescapeString#1%
236 \PE@onelevel@sanitize#1%
237 }

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

```

\PE@NormalizeLineEnd

```

242 \def\PE@NormalizeLineEnd##1{%
243 \def\PE@result{}%
244 \expandafter\PE@@NormalizeLineEnd##1#2\relax
245 \let##1\PE@result
246 }%

```

\PE@@NormalizeLineEnd

```

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

```

```

260 }%
261 }%
262 \uppercase{%
263   \x 89%
264 }

265 \begingroup
266   \catcode'\|=0 %
267   \catcode'\|=12 %

```

\PE@UnescapeString

```

268 |gdef|PE@UnescapeString#1{%
269   |begingroup
270     |def|PE@result{%
271       |expandafter|PE@DeString#1|relax
272     |expandafter|endgroup
273   |expandafter|def|expandafter#1|expandafter{|PE@result}%
274 }%

```

\PE@DeString

```

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

```



```

317     |fi
318   |fi
319   |PE@next
320 }%

\PE@CheckEndBackslash

321 |gdef|PE@CheckEndBackslash#1{%
322   |ifx|relax#1%
323   |else
324     |edef|PE@result{|PE@result\}%
325     |expandafter|PE@DeString|expandafter#1%
326   |fi
327 }%

328 |endgroup

\PE@TestOctDigit

329 \def\PE@TestOctDigit#1{%
330   \ifnum'#1<48 % 0
331     \@ne
332   \else
333     \ifnum'#1>55 % 7
334       \@ne
335     \else
336       \z@
337     \fi
338   \fi
339 }

\PE@OctI

340 \def\PE@OctI#1#2{%
341   \ifcase\PE@TestOctDigit#2%
342     \def\PE@next{\PE@OctII{#1#2}}%
343   \else
344     \def\PE@next{\PE@OctAll#1#2}%
345   \fi
346   \PE@next
347 }

\PE@OctII

348 \def\PE@OctII#1#2{%
349   \ifcase\PE@TestOctDigit#2%
350     \def\PE@next{\PE@OctAll{#1#2}}%
351   \else
352     \def\PE@next{\PE@OctAll{#1}#2}%
353   \fi
354   \PE@next
355 }

\PE@OctAll

356 \def\PE@OctAll#1{%
357   \uccode\z@='#1\relax
358   \uppercase{%
359     \edef\PE@result{\PE@result^^@}%
360   }%
361   \PE@DeString
362 }

```

2.6 User macros (pdfTeX analogues)

```

363 \begingroup\expandafter\expandafter\expandafter\endgroup
364 \expandafter\ifx\csname pdfescapehex\endcsname\relax

\edefEscapeHex
365 \long\def\edefEscapeHex#1#2{%
366   \edefSanitize#1{#2}%
367   \PE@SanitizeSpaceOther#1%
368   \PE@EscapeHex#1%
369 }%

\edefUnescapeHex
370 \def\edefUnescapeHex#1#2{%
371   \edefSanitize#1{#2}%
372   \PE@UnescapeHex#1%
373 }%

\edefEscapeName
374 \long\def\edefEscapeName#1#2{%
375   \edefSanitize#1{#2}%
376   \PE@SanitizeSpaceOther#1%
377   \PE@EscapeName#1%
378 }%

\edefEscapeString
379 \long\def\edefEscapeString#1#2{%
380   \edefSanitize#1{#2}%
381   \PE@SanitizeSpaceOther#1%
382   \PE@EscapeString#1%
383 }%

384 \else

\PE@edefbabel Help macro that adds support for babel's shorthand characters.
385 \long\def\PE@edefbabel#1#2#3{%
386   \begingroup
387     \csname @save@activetrue\endcsname
388     \edef#1{#2{#3}}%
389   \expandafter\endgroup
390   \expandafter\def\expandafter#1\expandafter{#1}%
391 }%

\edefEscapeHex
392 \long\def\edefEscapeHex#1#2{%
393   \PE@edefbabel#1\pdfescapehex{#2}%
394 }%

\edefUnescapeHex
395 \def\edefUnescapeHex#1#2{%
396   \PE@edefbabel#1\pdfunescapehex{#2}%
397 }%

\edefEscapeName
398 \long\def\edefEscapeName#1#2{%
399   \PE@edefbabel#1\pdfescapename{#2}%
400 }%

\edefEscapeString
401 \long\def\edefEscapeString#1#2{%
402   \PE@edefbabel#1\pdfescapestring{#2}%
403 }%

```

```

404 \PE@AtEnd
405 \expandafter\endinput
406 \fi

```

2.7 Help macros

2.7.1 Characters

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

```

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

\PE@backslash
408 \begingroup
409 \escapechar=-1 %
410 \edef\x{\endgroup
411 \def\noexpand\PE@backslash{\string\\}%
412 }
413 \x

```

2.7.2 Switch for ε -TeX

```

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

```

2.8 Conversions

2.8.1 Conversion to hex string

```

\PE@EscapeHex
420 \ifPE@etex
421 \def\PE@EscapeHex#1{%
422 \edef#1{\expandafter\PE@ToHex#1\relax}%
423 }%
424 \else
425 \def\PE@EscapeHex#1{%
426 \def\PE@result{%
427 \expandafter\PE@ToHex#1\relax
428 \let#1\PE@result
429 }%
430 \fi

\PE@ToHex
431 \def\PE@ToHex#1{%
432 \ifx\relax#1%
433 \else
434 \PE@HexChar{#1}%
435 \expandafter\PE@ToHex
436 \fi
437 }%

\PE@HexChar
438 \ifPE@etex
439 \def\PE@HexChar#1{%
440 \PE@HexDigit{\numexpr\dimexpr.0625\dimexpr'#1sp\relax\relax\relax}%
441 \PE@HexDigit{%
442 \numexpr'#1-16*\dimexpr.0625\dimexpr'#1sp\relax\relax\relax

```

```

443 }%
444 }%
445 \else
446 \def\PE@HexChar#1{%
447 \dimen0='#1sp%
448 \dimen2=.0625\dimen0 %
449 \advance\dimen0-16\dimen2 %
450 \edef\PE@result{%
451 \PE@result
452 \PE@HexDigit{\dimen2 }%
453 \PE@HexDigit{\dimen0 }%
454 }%
455 }%
456 \fi

```

\PE@HexDigit

```

457 \def\PE@HexDigit#1{%
458 \expandafter\string
459 \ifcase#1%
460 0\or 1\or 2\or 3\or 4\or 5\or 6\or 7\or 8\or 9\or
461 A\or B\or C\or D\or E\or F%
462 \fi
463 }

```

2.8.2 Character code to octal number

\PE@OctChar

```

464 \ifPE@etex
465 \def\PE@OctChar#1{%
466 \expandafter\PE@@OctChar
467 \the\numexpr\dimexpr.015625\dimexpr'#1sp\relax\relax
468 \expandafter\relax
469 \expandafter\relax
470 \the\numexpr\dimexpr.125\dimexpr'#1sp\relax\relax\relax
471 \relax
472 #1%
473 }%
474 \def\PE@@OctChar#1\relax#2\relax#3{%
475 \PE@backslash
476 #1%
477 \the\numexpr#2-8*#1\relax
478 \the\numexpr\dimexpr'#3sp\relax-8*#2\relax
479 }%
480 \else
481 \def\PE@OctChar#1{%
482 \dimen0='#1sp%
483 \dimen2=.125\dimen0 %
484 \dimen4=.125\dimen2 %
485 \advance\dimen0-8\dimen2 %
486 \advance\dimen2-8\dimen4 %
487 \edef\PE@result{%
488 \PE@result
489 \PE@backslash
490 \number\dimen4 %
491 \number\dimen2 %
492 \number\dimen0 %
493 }%
494 }%
495 \fi

```

2.8.3 Unpack hex string

\PE@UnescapeHex

```
496 \def\PE@UnescapeHex#1{%
497   \begingroup
498     \PE@InitUccodeHexDigit
499     \def\PE@result{%
500       \expandafter\PE@DeHex#1\relax\relax
501     \expandafter\endgroup
502   \expandafter\def\expandafter#1\expandafter{\PE@result}%
503 }
```

\PE@DeHex

```
504 \def\PE@DeHex#1#2{%
505   \ifx#2\relax
506     \ifx#1\relax
507       \let\PE@next\relax
508     \else
509       \uppercase{%
510         \def\PE@testA{#1}%
511       }%
512       \ifcase\expandafter\PE@TestUcHexDigit\PE@testA
513         \def\PE@next{%
514           \PE@DeHex#10\relax\relax
515         }%
516       \else
517         \let\PE@next\relax
518       \fi
519     \fi
520   \else
521     \uppercase{%
522       \def\PE@testA{#1}%
523       \def\PE@testB{#2}%
524     }%
525     \ifcase\expandafter\PE@TestUcHexDigit\PE@testA
526       \ifcase\expandafter\PE@TestUcHexDigit\PE@testB
527         \uccode\z@="\PE@testA\PE@testB\relax
528         \ifnum\uccode\z@=32 %
529           \let\PE@temp\PE@space@space
530         \else
531           \uppercase{%
532             \def\PE@temp{^^@}%
533           }%
534         \fi
535       \edef\PE@result{\PE@result\PE@temp}%
536       \let\PE@next\PE@DeHex
537     \else
538       % invalid input sequence
539       \def\PE@next{%
540         \PE@DeHex#1%
541       }%
542     \fi
543   \else
544     % invalid input sequence
545     \def\PE@next{\PE@DeHex#2}%
546   \fi
547 \fi
548 \PE@next
549 }
```

\PE@EscapeName

```
\PE@EscapeNameTokens
```

```
\PE@EscapeNameAdd
```

14

```

605 }%
606 \fi

```

2.8.5 Conversion to PDF string

\PE@EscapeString

```

607 \ifPE@etex
608 \def\PE@EscapeString#1{%
609 \edef#1{\expandafter\PE@EscapeStringTokens#1\relax}%
610 }%
611 \else
612 \def\PE@EscapeString#1{%
613 \begingroup
614 \def\PE@result{%
615 \expandafter\PE@EscapeStringTokens#1\relax
616 \expandafter\endgroup
617 \expandafter\def\expandafter#1\expandafter{\PE@result}%
618 }%
619 \fi

```

\PE@EscapeStringTokens

```

620 \def\PE@EscapeStringTokens#1{%
621 \ifx\relax#1%
622 \else
623 \ifnum'#1<33 %
624 \PE@OctChar#1%
625 \else
626 \ifnum'#1>126 %
627 \PE@OctChar#1%
628 \else \ifnum'#1=40 \PE@EscapeStringAdd{\string\}% (
629 \else\ifnum'#1=41 \PE@EscapeStringAdd{\string\)}% )
630 \else\ifnum'#1=92 \PE@EscapeStringAdd{\string\\}% \
631 \else
632 \PE@EscapeStringAdd{#1}%
633 \fi\fi\fi
634 \fi
635 \fi
636 \expandafter\PE@EscapeStringTokens
637 \fi
638 }%

```

\PE@EscapeStringAdd

```

639 \ifPE@etex
640 \def\PE@EscapeStringAdd#1{#1}%
641 \else
642 \def\PE@EscapeStringAdd#1{%
643 \edef\PE@result{%
644 \PE@result
645 #1%
646 }%
647 }%
648 \fi

649 \PE@AtEnd
650 \</package>

```

3 Test

3.1 Catcode checks for loading

```

651 \<*test1>

```

```

652 \catcode'\@=11 %
653 \def\RestoreCatcodes{}
654 \count@=0 %
655 \loop
656   \edef\RestoreCatcodes{%
657     \RestoreCatcodes
658     \catcode\the\count@=\the\catcode\count@\relax
659   }%
660 \ifnum\count@<255 %
661   \advance\count@\@ne
662 \repeat
663
664 \def\RangeCatcodeInvalid#1#2{%
665   \count@=#1\relax
666   \loop
667     \catcode\count@=15 %
668     \ifnum\count@<#2\relax
669       \advance\count@\@ne
670     \repeat
671 }
672 \def\Test{%
673   \RangeCatcodeInvalid{0}{47}%
674   \RangeCatcodeInvalid{58}{64}%
675   \RangeCatcodeInvalid{91}{96}%
676   \RangeCatcodeInvalid{123}{255}%
677   \catcode'\@=12 %
678   \catcode'\=0 %
679   \catcode'\{=1 %
680   \catcode'\}=2 %
681   \catcode'\#=6 %
682   \catcode'\[=12 %
683   \catcode'\]=12 %
684   \catcode'\%=14 %
685   \catcode'\ =10 %
686   \catcode13=5 %
687   \input pdfescape.sty\relax
688   \RestoreCatcodes
689 }
690 \Test
691 \csname @@end\endcsname
692 \end
693 </test1>

```

3.2 Macro tests

```

694 <*test2 | test3 | test4>
695 \NeedsTeXFormat{LaTeX2e}
696 \makeatletter

```

3.3 Test with \pdfescape... commands

```

697 <*test2>
698 \ProvidesFile{pdfescape-test1.tex}%
699   [2007/09/09 v1.6 Test with \string\pdfescape... commands]%
700 </test2>

```

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

```

701 <*test3>
702 \ProvidesFile{pdfescape-test2.tex}%
703   [2007/09/09 v1.6 Test without \string\pdfescape..., with  $\varepsilon$ -TeX]%
704 </test3>

```

3.5 Test without \pdfescape... and ε -TeX


```

705 <*test4>
706 \ProvidesFile{pdfescape-test3.tex}%
707 [2007/09/09 v1.6 Test without \string\pdfescape... and e-TeX]%
708 </test4>

```

3.6 Check/ensure test preconditions

3.6.1 Check \pdfescape...

```

709 <*test2>
710 \@ifundefined{pdfescapehex}{%
711   \PackageError{pdfescape-test1}{%
712     Missing \string\pdfescape... commands%
713   }{Test aborted.}%
714   \stop
715 }{}
716 </test2>

717 <*test3 | test4>
718 \let\pdfescapehex\@undefined
719 \let\pdfunescapehex\@undefined
720 \let\pdfescapename\@undefined
721 \let\pdfescapestring\@undefined
722 </test3 | test4>

```

3.6.2 Check ε -TeX

```

723 <*test3>
724 \@ifundefined{numexpr}{%
725   \PackageError{pdfescape-test2}{%
726     Missing \eTeX
727   }{Test aborted.}%
728   \stop
729 }{}
730 </test3>

```

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

3.7 Common part

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

```

731 \RequirePackage{qstest}
732 \IncludeTests{*}
733 \LogTests{log}{*}{*}
734
735 \newcommand*{\ExpectVar}[2]{%
736   \ifx#1#2%
737   \else
738     \begingroup
739       \@onelevel@sanitize#1%
740       \@onelevel@sanitize#2%
741       \typeout{[#1] <> [#2]}% hash-ok
742     \endgroup
743   \fi
744   \Expect*{\ifx#1#2true\else false\fi}{true}%
745 }
746
747 \makeatletter
748 \begingroup
749   \gdef\AllBytes{}%
750   \count@=0 %
751   \catcode0=12 %
752   \@whilenum\count@<256 \do{%
753     \lccode0=\count@

```

```

754 \ifnum\count@=32 %
755 \xdef\AllBytes{\AllBytes\space}%
756 \else
757 \lowercase{%
758 \xdef\AllBytes{\AllBytes^^@}%
759 }%
760 \fi
761 \advance\count@ by 1 %
762 }%
763 \endgroup
764 \newcommand*{\AllBytesHex}{%
765 000102030405060708090A0B0C0D0E0F%
766 101112131415161718191A1B1C1D1E1F%
767 202122232425262728292A2B2C2D2E2F%
768 303132333435363738393A3B3C3D3E3F%
769 404142434445464748494A4B4C4D4E4F%
770 505152535455565758595A5B5C5D5E5F%
771 606162636465666768696A6B6C6D6E6F%
772 707172737475767778797A7B7C7D7E7F%
773 808182838485868788898A8B8C8D8E8F%
774 909192939495969798999A9B9C9D9E9F%
775 A0A1A2A3A4A5A6A7A8A9AABACADAFAF%
776 B0B1B2B3B4B5B6B7B8B9ABBBBCDBEBF%
777 C0C1C2C3C4C5C6C7C8C9CACBCCDCECF%
778 D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF%
779 E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF%
780 F0F1F2F3F4F5F6F7F8F9FABFBCFDFEFF%
781 }
782 \@onelevel@sanitize\AllBytesHex
783 \expandafter\lowercase\expandafter{%
784 \expandafter\newcommand\expandafter*\expandafter\AllBytesHexLC
785 \expandafter{\AllBytesHex}%
786 }
787 \newcommand*{\AllBytesName}{%
788 \begingroup
789 \catcode'\# =12 %
790 \xdef\AllBytesName{%
791 #01#02#03#04#05#06#07#08#09#0A#0B#0C#0D#0E#0F%
792 #10#11#12#13#14#15#16#17#18#19#1A#1B#1C#1D#1E#1F%
793 #20!"#23$#25&'#28#29*+,-.#2F%
794 0123456789:;#3C=#3E?%
795 @ABCDEFGHIJKLMNO%
796 PQRSTUVWXYZ#5B\@backslashchar#5D^_%
797 'abcdefghijklmnopqrstuvwxyz%
798 pqrstuvwxyz#7B|#7D\string~#7F%
799 #80#81#82#83#84#85#86#87#88#89#8A#8B#8C#8D#8E#8F%
800 #90#91#92#93#94#95#96#97#98#99#9A#9B#9C#9D#9E#9F%
801 #A0#A1#A2#A3#A4#A5#A6#A7#A8#A9#AA#AB#AC#AD#AE#AF%
802 #B0#B1#B2#B3#B4#B5#B6#B7#B8#B9#BA#BB#BC#BD#BE#BF%
803 #C0#C1#C2#C3#C4#C5#C6#C7#C8#C9#CA#CB#CC#CD#CE#CF%
804 #D0#D1#D2#D3#D4#D5#D6#D7#D8#D9#DA#DB#DC#DD#DE#DF%
805 #E0#E1#E2#E3#E4#E5#E6#E7#E8#E9#EA#EB#EC#ED#EE#EF%
806 #F0#F1#F2#F3#F4#F5#F6#F7#F8#F9#FA#FB#FC#FD#FE#FF%
807 }%
808 \endgroup
809 \@onelevel@sanitize\AllBytesName
810
811 \newcommand*{\AllBytesString}{%
812 \begingroup
813 \def\{|}%
814 \edef\%{\@percentchar}%
815 \catcode'\|=0 %

```

```

816 \catcode'\#=12 %
817 \catcode'\~=12 %
818 \catcode'\'=12 %
819 |xdef|AllBytesString{%
820 \000\001\002\003\004\005\006\007\010\011\012\013\014\015\016\017%
821 \020\021\022\023\024\025\026\027\030\031\032\033\034\035\036\037%
822 \040!"#$%&'(\)*+,-./%
823 0123456789:;<=>?%
824 @ABCDEFGHIJKLMNO%
825 PQRSTUVWXYZ[\]^_%
826 'abcdefghijklmno%
827 pqrstuvwxyz{|}~\177%
828 \200\201\202\203\204\205\206\207\210\211\212\213\214\215\216\217%
829 \220\221\222\223\224\225\226\227\230\231\232\233\234\235\236\237%
830 \240\241\242\243\244\245\246\247\250\251\252\253\254\255\256\257%
831 \260\261\262\263\264\265\266\267\270\271\272\273\274\275\276\277%
832 \300\301\302\303\304\305\306\307\310\311\312\313\314\315\316\317%
833 \320\321\322\323\324\325\326\327\330\331\332\333\334\335\336\337%
834 \340\341\342\343\344\345\346\347\350\351\352\353\354\355\356\357%
835 \360\361\362\363\364\365\366\367\370\371\372\373\374\375\376\377%
836 }%
837 |endgroup
838 \@onelevel@sanitize\AllBytesString
839
840 <*test4>
841 \let\org@detokenize\detokenize
842 \let\detokenize\@undefined
843 \let\org@numexpr\numexpr
844 \let\numexpr\@undefined
845 </test4>
846 \RequirePackage{pdfescape}
847 <*test4>
848 \let\detokenize\org@detokenize
849 \let\numexpr\org@numexpr
850 </test4>
851
852 \begin{qstest}{all-hex}{\AllBytes, escapehex}
853 \EdefEscapeHex\x{\AllBytes}%
854 \Expect*{\x}*{\AllBytesHex}%
855 \ExpectVar\x\AllBytesHex
856 \end{qstest}
857
858 \begin{qstest}{all-unhex}{\AllBytesHex, unescapehex}
859 \EdefUnescapeHex\x{\AllBytesHex}%
860 \Expect*{\x}*{\AllBytes}%
861 \ExpectVar\x\AllBytes
862 \end{qstest}
863
864 \begin{qstest}{all-unhex-lc}{\AllBytesHexLC, unescapehex, lowercase}
865 \EdefUnescapeHex\x{\AllBytesHexLC}%
866 \Expect*{\x}*{\AllBytes}%
867 \ExpectVar\x\AllBytes
868 \end{qstest}
869
870 \begin{qstest}{unhex-incomplete}{unescapehex, incomplete}
871 \EdefUnescapeHex\x{4}%
872 \Expect*{\x}{@}%
873 \end{qstest}
874
875 \begin{qstest}{unhex-space}{unescapehex, space}
876 \EdefUnescapeHex\x{20}%
877 \Expect*{\x}{ }%

```

```

878 \ExpectVar\x\space
879 \end{qstest}
880
881 \begin{qstest}{unhex-spaces}{unescapehex, spaces}
882 \EdefUnescapeHex\x{204020204120}%
883 \def\y#1{%
884   \edef\z{#1\string @#1#1\string A#1}%
885 } \y{ }%
886 \Expect*{\x}*{\z}%
887 \ExpectVar\x\z
888 \end{qstest}
889
890 \begin{qstest}{unhex-hash}{unescapehex, hash}
891 \catcode'\#=12 %
892 \EdefUnescapeHex\x{#20}%
893 \ExpectVar\x\space
894 \end{qstest}
895
896 \begin{qstest}{unhex-invalid}{unescapehex, invalid}
897 \def\test#1#2{%
898   \EdefUnescapeHex\x{#1}%
899   \edef\y{#2}%
900   \@onelevel@sanitize\y
901   \ExpectVar\x\y
902 }%
903 \test2
904 \edef\x{\pdfunescapehex{4X}}%
905 \edef\y{\string @}%
906 \ifx\x\y
907 \else
908   \def~{\space}%
909   \typeout{*****}%
910   \typeout{* Your pdfTeX contains bug 777.~~~~*}%
911   \typeout{* This test is redefined as dummy, *}%
912   \typeout{* because it triggers the bug.~~~~*}%
913   \typeout{*****}%
914   \def\test#1#2{%
915     \fi
916   }
917 \test{X}{}%
918 \test{XY}{}%
919 \test{XYZ}{}%
920 \test{A}{~^a0}%
921 \test{AX}{~^a0}%
922 \test{XA}{~^a0}%
923 \test{XXAXX}{~^a0}%
924 \end{qstest}
925
926 \begin{qstest}{all-name}{\AllBytes, escapename}
927 \EdefEscapeName\x{\AllBytes}%
928 \Expect*{\x}*{\AllBytesName}%
929 \ExpectVar\x\AllBytesName
930 \end{qstest}
931
932 \begin{qstest}{all-string}{\AllBytes, escapestring}
933 \EdefEscapeString\x{\AllBytes}%
934 \Expect*{\x}*{\AllBytesString}%
935 \ExpectVar\x\AllBytesString
936 \end{qstest}
937
938 \begin{qstest}{uchexdigit}{unescape, uppercase hex digit}
939 \catcode'\@=11 %

```

```

940 \catcode0=12 %
941 \def\test#1#2{%
942   \uccode0=#1\relax
943   \uppercase{%
944     \def\x{^^@}%
945   }%
946   \Expect*{%
947     \ifcase\expandafter\PE@TestUcHexDigit\x
948     true%
949     \else
950     false%
951     \fi
952   }{#2}%
953 }%
954 \def\range#1#2#3{%
955   \count0=#1\relax
956   \loop
957   \ifnum\count0<#2\relax
958     \test{\count0}{#3}%
959     \advance\count0 by 1 %
960   \repeat
961 }%
962 \range{0}{47}{false}%
963 \range{48}{57}{true}%
964 \range{58}{64}{false}%
965 \range{65}{70}{true}%
966 \range{71}{255}{false}%
967 \end{qstest}
968
969 \begin{qstest}{unescapename}{unescapename}
970   \def\test#1#2{%
971     \EdefUnescapeName\x{#1}%
972     \edef\y{#2}%
973     \@onelevel@sanitize\y
974     \ExpectVar\x\y
975   }%
976   \catcode'\#=12 %
977   \catcode0=12 %
978   \test{}{}%
979   \test{x}{x}%
980   \test{xy}{xy}%
981   \test{#}{#}%
982   \test{##}{##}%
983   \test{###}{###}%
984   \test{####}{####}%
985   \test{#x}{#x}%
986   \test{#xy}{#xy}%
987   \test{#1}{#1}%
988   \test{#40}{@}%
989   \test{#400}{@0}%
990   \test{#4x0}{#4x0}%
991   \test{#ab}{^^ab}%
992   \test{#00}{^^@}%
993   \test{x#40y#40z}{x@y@z}%
994   \test{#40#40#40#40}{@@@@}%
995   \test{a#x}{a#x}%
996   \test{a#xy}{a#xy}%
997   \test{a#1}{a#1}%
998   \test{a#40}{a@}%
999   \test{a#400}{a@0}%
1000 \test{#20}{ }%
1001 \test{a#20}{a }%

```

```

1002 \test{a#20b}{a b}%
1003 \test{a#20#20#20b}{a \space\space b}%
1004 \end{qstest}
1005
1006 \begin{qstest}{unescapestring}{unescapestring}
1007 \def\test#1#2{%
1008     \EdefUnescapeString\x{#1}%
1009     \edef\y{#2}%
1010     \@onelevel@sanitize\y
1011     \ExpectVar\x\y
1012 }%
1013 \catcode0=12 %
1014 \def\DefChar#1#2{%
1015     \begingroup
1016         \uccode0=#2\relax
1017         \uppercase{\endgroup
1018             \def#1{^^@}%
1019         }%
1020 }%
1021 \DefChar\nul{0}%
1022 \DefChar\one{1}%
1023 \DefChar\bel{8}%
1024 \DefChar\tab{9}%
1025 \DefChar\lf{10}%
1026 \DefChar\ff{12}%
1027 \DefChar\cr{13}%
1028 \DefChar\{\{92}%
1029 \test{}{}%
1030 \test{a}{a}%
1031 \test{\}\}%
1032 \test{\\\}\}%
1033 \test{\\\y}{\y}%
1034 \test{\\000}{\nul}%
1035 \test{\\b}{\bel}%
1036 \test{\\t}{\tab}%
1037 \test{\\n}{\lf}%
1038 \test{\\f}{\ff}%
1039 \test{\\r}{\cr}%
1040 \test{\\(){}%
1041 \test{\\)}{)}%
1042 \test{\\040}{ }%
1043 \test{\\100}{@}%
1044 \test{\\40}{ }%
1045 \test{\\1}{\one}%
1046 \test{\\01}{\one}%
1047 \test{\\001}{\one}%
1048 \test{\\18}{\one8}%
1049 \test{\\018}{\one8}%
1050 \test{\\0018}{\one8}%
1051 \test{x\\}{x}%
1052 \test{x\\\}\}{x\\}%
1053 \test{x\\\y}{x\y}%
1054 \test{x\\000}{x\nul}%
1055 \test{x\\b}{x\bel}%
1056 \test{x\\t}{x\tab}%
1057 \test{x\\n}{x\lf}%
1058 \test{x\\f}{x\ff}%
1059 \test{x\\r}{x\cr}%
1060 \test{x\\(){x}%
1061 \test{x\\)}{x}%
1062 \test{x\\040}{x }%
1063 \test{x\\100}{x@}%

```

```

1064 \test{x\\40}{x }%
1065 \test{x\\1}{x\one}%
1066 \test{x\\01}{x\one}%
1067 \test{x\\001}{x\one}%
1068 \test{x\\18}{x\one8}%
1069 \test{x\\018}{x\one8}%
1070 \test{x\\0018}{x\one8}%
1071 \test{\\b\\t\\n\\f\\r\\(\\)\\000\\040}{%
1072 \bel\tab\lf\ff\cr()\\nul\space
1073 }%
1074 \test{\\lf}{}%
1075 \test{x\\lf}{x}%
1076 \test{\cr}{\lf}%
1077 \test{\cr\lf}{\lf}%
1078 \test{\lf}{\lf}%
1079 \test{\lf\cr}{\lf\lf}%
1080 \test{x\cr}{x\lf}%
1081 \test{x\cr\lf}{x\lf}%
1082 \test{x\lf}{x\lf}%
1083 \test{x\lf\cr}{x\lf\lf}%
1084 \test{x\\cr\lf y\cr}{xy\lf}%
1085 \end{qstest}
1086 \stop
1087 </test2 | test3 | test4>

```

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:macros/latex/contrib/oberdiek/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
```

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/
```

¹<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- \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
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-test4.tex → doc/latex/oberdiek/pdfescape-test4.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 , mi \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 pdf \LaTeX :

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

[2007/08/27 v1.5]

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

[2007/09/09 v1.6]

- Fix in catcode setup.

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		214, 217, 221, 331, 334, 661, 669
<code>\#</code>	139, 275, 300, 681, 789, 816, 891, 976	<code>\@onelevel@sanitize</code> 97, 739,
<code>\\$</code> 138	740, 782, 809, 838, 900, 973, 1010
<code>\%</code> 684, 814	<code>\@percentchar</code> 814
<code>\(</code> 628, 822	<code>\@undefined</code> 718, 719, 720, 721, 842, 844
<code>\)</code> 629, 822	<code>\@whilenum</code> 752
<code>\@</code> 652, 677, 939	<code>\[</code> 682
<code>\@ReturnAfterFi</code> 125, <u>130</u>	<code>\%</code> 122, 267, 411, 630, 678, 818, 825,
<code>\@backslashchar</code> 796	1028, 1031, 1032, 1033, 1034,
<code>\@ifundefined</code> 710, 724	1035, 1036, 1037, 1038, 1039,
<code>\@ne</code> 166, 169,	1040, 1041, 1042, 1043, 1044,

1045, 1046, 1047, 1048, 1049, 1050, 1051, 1052, 1053, 1054, 1055, 1056, 1057, 1058, 1059, 1060, 1061, 1062, 1063, 1064, 1065, 1066, 1067, 1068, 1069, 1070, 1071, 1072, 1074, 1075, 1084			E
\{ 679		\EdefEscapeHex 2, 365, 392, 853	
\} 324, 680		\EdefEscapeName 374, 398, 927	
\] 683		\EdefEscapeString 379, 401, 933	
\ 266, 271, 813, 815		\EdefSanitize 2, 80, 112, 132, 232, 366, 371, 375, 380	
\~ 817		\EdefUnescapeHex 370, 395, 859, 865, 871, 876, 882, 892, 898	
		\EdefUnescapeName 2, 131, 971	
		\EdefUnescapeString 2, 231, 1008	
		\empty 12	
		\end 692, 856, 862, 868, 873, 879, 888, 894, 924, 930, 936, 967, 1004, 1085	
Numbers		\endcsname 8, 18, 39, 52, 55, 81, 84, 91, 102, 364, 387, 416, 691	
\0 820, 821, 822		\endinput 27, 405	
\1 827		\escapechar 409	
\2 828, 829, 830, 831		\eTeX 726	
\3 832, 833, 834, 835		\Expect 744, 854, 860, 866, 872, 877, 886, 928, 934, 946	
\8 239		\ExpectVar 735, 855, 861, 867, 878, 887, 893, 901, 929, 935, 974, 1011	
\9 240			
_ 114, 630, 685			
	A		F
\advance 449, 485, 486, 661, 669, 761, 959		\ff 1026, 1038, 1058, 1072	
\AllBytes 749, 755, 758, 852, 853, 860, 861, 866, 867, 926, 927, 932, 933			G
\AllBytesHex 764, 782, 785, 854, 855, 858, 859		\gdef 140, 148, 749	
\AllBytesHexLC 784, 864, 865			I
\AllBytesName . 787, 790, 809, 928, 929		\ifcase 9, 162, 163, 341, 349, 459, 512, 525, 526, 565, 947	
\AllBytesString ... 811, 838, 934, 935		\ifnum 213, 216, 219, 220, 330, 333, 528, 564, 572, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 623, 626, 628, 629, 630, 660, 668, 754, 957	
	B	\ifPE@etex 414, 420, 438, 464, 550, 597, 607, 639	
\begin . 852, 858, 864, 870, 875, 881, 890, 896, 926, 932, 938, 969, 1006		\ifx 10, 12, 18, 39, 47, 81, 91, 122, 149, 153, 248, 253, 364, 416, 432, 505, 506, 562, 621, 736, 744, 906	
\bel 1023, 1035, 1055, 1072		\immediate 20, 41	
	C	\IncludeTests 732	
\catcode 3, 4, 5, 6, 7, 31, 32, 33, 34, 35, 36, 37, 38, 56, 58, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 114, 138, 139, 266, 267, 652, 658, 667, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 751, 789, 815, 816, 817, 818, 891, 939, 940, 976, 977, 1013		\input 687	
\count 955, 957, 958, 959			L
\count@ 654, 658, 660, 661, 665, 667, 668, 669, 750, 752, 753, 754, 761		\lccode 753	
\cr .. 1027, 1039, 1059, 1072, 1076, 1077, 1079, 1080, 1081, 1083, 1084		\lf 1025, 1037, 1057, 1072, 1074, 1075, 1076, 1077, 1078, 1079, 1080, 1081, 1082, 1083, 1084	
\csname 8, 18, 39, 52, 55, 81, 84, 91, 102, 364, 387, 416, 691		\LogTests 733	
	D	\loop 655, 666, 956	
\DefChar 1014, 1021, 1022, 1023, 1024, 1025, 1026, 1027, 1028		\lowercase 757, 783	
\detokenize 109, 841, 842, 848			M
\dimen . 447, 448, 449, 452, 453, 482, 483, 484, 485, 486, 490, 491, 492		\makeatletter 696, 747	
\dimexpr 440, 442, 467, 470, 478		\meaning 93	
\do 752			N
			\NeedsTeXFormat 695
			\newcommand ... 735, 764, 784, 787, 811
			\newif 414
			\nul 1021, 1034, 1054, 1072

\number 490, 491, 492
 \numexpr 440, 442,
 467, 470, 477, 478, 843, 844, 849

O

\one 1022, 1045,
 1046, 1047, 1048, 1049, 1050,
 1065, 1066, 1067, 1068, 1069, 1070
 \org@detokenize 841, 848
 \org@numexpr 843, 849

P

\PackageError 711, 725
 \PackageInfo 23
 \pdfescape 699, 703, 707, 712
 \pdfescapehex 393, 718
 \pdfescapename 399, 720
 \pdfescapestring 402, 721
 \pdfunescapehex 396, 719, 904
 \PE@@NormalizeLineEnd 244, 247
 \PE@@OctChar 466, 474
 \PE@AtEnd 60, 61, 404, 649
 \PE@backslash 408, 475, 489
 \PE@CheckEndBackslash 321
 \PE@DeHex 500, 504
 \PE@DeName 144, 148, 177, 181
 \PE@DeString 275, 361
 \PE@edefbabel . 385, 393, 396, 399, 402
 \PE@EscapeHex 368, 420
 \PE@EscapeName 377, 550
 \PE@EscapeNameAdd
 568, 573, 587, 595, 597
 \PE@EscapeNameHashChar
 575, 576, 577, 578, 579,
 580, 581, 582, 583, 584, 585, 594
 \PE@EscapeNameTokens .. 552, 557, 561
 \PE@EscapeString 382, 607
 \PE@EscapeStringAdd
 628, 629, 630, 632, 639
 \PE@EscapeStringTokens 609, 615, 620
 \PE@etextrtrue 418
 \PE@hash 407, 568, 573, 595
 \PE@HexChar 434, 438, 569, 574
 \PE@HexDigit .. 440, 441, 452, 453, 457
 \PE@InitUccodeHexDigit 142, 188, 498
 \PE@next 151, 156, 177,
 181, 185, 250, 254, 256, 259,
 342, 344, 346, 350, 352, 354,
 507, 513, 517, 536, 539, 545, 548
 \PE@NormalizeLineEnd 234, 242
 \PE@OctAll 344, 350, 352, 356
 \PE@OctChar 464, 624, 627
 \PE@OctI 340
 \PE@OctII 342, 348
 \PE@onelevel@sanitize
 86, 92, 97, 104, 108, 135, 236
 \PE@result 143, 146, 150,
 155, 176, 180, 243, 245, 249,
 252, 359, 426, 428, 450, 451,
 487, 488, 499, 502, 535, 556,
 558, 601, 602, 614, 617, 643, 644
 \PE@sanitize 112

\PE@SanitizeSpaceOther
 117, 133, 233, 367, 376, 381
 \PE@space@other 113, 124
 \PE@space@space 116, 529
 \PE@SpaceToOther 118, 120
 \PE@strip@prefix 93, 95
 \PE@temp 173, 176, 529, 532, 535
 \PE@testA 159,
 162, 171, 510, 512, 522, 525, 527
 \PE@testB . 160, 163, 171, 523, 526, 527
 \PE@TestOctDigit 329, 341, 349
 \PE@TestUcHexDigit
 162, 163, 212, 512, 525, 526, 947
 \PE@ToHex 422, 427, 431
 \PE@UnescapeHex 372, 496
 \PE@UnescapeName 134, 137
 \PE@UnescapeString 235, 268
 \ProvidesFile 698, 702, 706
 \ProvidesPackage 53

R

\range 954, 962, 963, 964, 965, 966
 \RangeCatcodeInvalid
 664, 673, 674, 675, 676
 \repeat 662, 670, 960
 \RequirePackage 731, 846
 \RestoreCatcodes .. 653, 656, 657, 688

S

\space .. 755, 878, 893, 908, 1003, 1072
 \stop 714, 728, 1086

T

\tab 1024, 1036, 1056, 1072
 \Test 672, 690
 \test 897, 914, 917, 918, 919, 920, 921,
 922, 923, 941, 958, 970, 978,
 979, 980, 981, 982, 983, 984,
 985, 986, 987, 988, 989, 990,
 991, 992, 993, 994, 995, 996,
 997, 998, 999, 1000, 1001, 1002,
 1003, 1007, 1029, 1030, 1031,
 1032, 1033, 1034, 1035, 1036,
 1037, 1038, 1039, 1040, 1041,
 1042, 1043, 1044, 1045, 1046,
 1047, 1048, 1049, 1050, 1051,
 1052, 1053, 1054, 1055, 1056,
 1057, 1058, 1059, 1060, 1061,
 1062, 1063, 1064, 1065, 1066,
 1067, 1068, 1069, 1070, 1071,
 1074, 1075, 1076, 1077, 1078,
 1079, 1080, 1081, 1082, 1083, 1084
 \the 56, 62, 467, 470, 477, 478, 658
 \TMP@EnsureCode 59, 66, 67, 68, 69,
 70, 71, 72, 73, 74, 75, 76, 77, 78, 79
 \typeout .. 741, 909, 910, 911, 912, 913

U

\uccode 77, 78,
 79, 171, 175, 189, 190, 191, 192,
 193, 194, 195, 196, 197, 198,
 199, 200, 201, 202, 203, 204,

	205, 206, 207, 208, 209, 210,	887, 892, 893, 898, 901, 904,
	239, 240, 357, 527, 528, 942, 1016	906, 927, 928, 929, 933, 934,
\uppercase 158, 172,	935, 944, 947, 971, 974, 1008, 1011
	262, 358, 509, 521, 531, 943, 1017	
		Y
	W	\y 883, 885, 899, 900, 901, 905,
\write 20, 41	906, 972, 973, 974, 1009, 1010, 1011
	X	Z
\x 8, 10, 12, 19,	\z 884, 886, 887
	23, 25, 40, 45, 52, 115, 241, 263,	\z@ ... 164, 171, 175, 195, 196, 197,
	410, 413, 853, 854, 855, 859,	198, 199, 200, 201, 202, 203,
	860, 861, 865, 866, 867, 871,	204, 205, 206, 207, 208, 209,
	872, 876, 877, 878, 882, 886,	210, 223, 226, 336, 357, 527, 528