

The pdfescape package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2007/04/11 v1.3

Abstract

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

Contents

| | | |
|----------|--|-----------|
| 1 | Documentation | 2 |
| 2 | Implementation | 2 |
| 2.1 | Reload check and package identification | 2 |
| 2.2 | Preparations | 3 |
| 2.2.1 | Catcodes | 3 |
| 2.3 | User macros | 3 |
| 2.4 | Help macros | 4 |
| 2.4.1 | Characters | 4 |
| 2.4.2 | Switch for ϵ -TeX | 5 |
| 2.5 | Sanitizing | 5 |
| 2.6 | Conversions | 6 |
| 2.6.1 | Conversion to hex string | 6 |
| 2.6.2 | Character code to octal number | 7 |
| 2.6.3 | Unpack hex string | 7 |
| 2.6.4 | Conversion to PDF name | 8 |
| 2.6.5 | Conversion to PDF string | 9 |
| 3 | Test | 10 |
| 3.1 | Test with <code>\pdfescape...</code> commands | 10 |
| 3.2 | Test without <code>\pdfescape...</code> , with ϵ -TeX | 10 |
| 3.3 | Test without <code>\pdfescape...</code> and ϵ -TeX | 10 |
| 3.4 | Check/ensure test preconditions | 10 |
| 3.4.1 | Check <code>\pdfescape...</code> | 10 |
| 3.4.2 | Check ϵ -TeX | 10 |
| 3.5 | Common part | 10 |
| 4 | Installation | 13 |
| 4.1 | Download | 13 |
| 4.2 | Bundle installation | 13 |
| 4.3 | Package installation | 14 |
| 4.4 | Refresh file name databases | 14 |
| 4.5 | Some details for the interested | 14 |
| 5 | History | 15 |
| | [2007/02/21 v1.0] | 15 |
| | [2007/02/25 v1.1] | 15 |
| | [2007/03/20 v1.2] | 15 |
| | [2007/04/11 v1.3] | 15 |

1 Documentation

```

\edefEscapeHex {⟨cmd⟩} {⟨string⟩}
\edefUnescapeHex {⟨cmd⟩} {⟨string⟩}
\edefEscapeName {⟨cmd⟩} {⟨string⟩}
\edefEscapeString {⟨cmd⟩} {⟨string⟩}

```

These commands convert $\langle string \rangle$ and store 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 $\epsilon = 1.30$ 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`.

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

```

Package identification:

```

25 \begingroup
26   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
27     \def\x#1#2#3[#4]{\endgroup
28       \immediate\write-1{Package: #3 #4}%
29       \xdef#1{#4}%

```

```

30     }%
31   \else
32     \def\x#1#2[#3]{\endgroup
33       #2[#3]}%
34     \ifx#1\relax
35       \xdef#1{#3}%
36     \fi
37   }%
38 \fi
39 \expandafter\x\csname ver@pdfescape.sty\endcsname
40 \ProvidesPackage{pdfescape}%
41 [2007/04/11 v1.3 Provides hex, PDF name and string conversions (H0)]

```

2.2 Preparations

2.2.1 Catcodes

```

42 \expandafter\edef\csname PE@AtEnd\endcsname{%
43   \catcode64 \the\catcode64\relax
44 }
45 \catcode64 11 % @
46 \def\PE@EnsureCode#1#2#3{%
47   \edef\PE@AtEnd{%
48     \PE@AtEnd
49     #1#2 \the#1#2\relax
50   }%
51   #1#2 #3\relax
52 }
53 \PE@EnsureCode\catcode{0}{12}% ^^@
54 \PE@EnsureCode\catcode{34}{12}% "
55 \PE@EnsureCode\catcode{42}{12}% *
56 \PE@EnsureCode\catcode{45}{12}% -
57 \PE@EnsureCode\catcode{46}{12}% .
58 \PE@EnsureCode\catcode{60}{12}% <
59 \PE@EnsureCode\catcode{61}{12}% =
60 \PE@EnsureCode\catcode{62}{12}% >
61 \PE@EnsureCode\catcode{94}{7}% ^
62 \PE@EnsureCode\catcode{96}{12}% ‘
63 \PE@EnsureCode\uccode{34}{0}% "
64 \PE@EnsureCode\uccode{48}{0}% 0
65 \PE@EnsureCode\uccode{61}{0}% =

```

2.3 User macros

```

66 \begingroup\expandafter\expandafter\expandafter\endgroup
67 \expandafter\ifx\csname pdfescapehex\endcsname\relax

```

\EdefEscapeHex

```

68   \long\def\EdefEscapeHex#1#2{%
69     \PE@sanitize#1{#2}%
70     \edef#1{\expandafter\PE@SpaceToOther#1 \relax}%
71     \PE@EscapeHex#1%
72   }%

```

\EdefUnescapeHex

```

73   \def\EdefUnescapeHex#1#2{%
74     \PE@sanitize#1{#2}%
75     \PE@UnescapeHex#1%
76   }%

```

\EdefEscapeName

```

77   \long\def\EdefEscapeName#1#2{%
78     \PE@sanitize#1{#2}%

```

```

79     \edef#1{\expandafter\PE@SpaceToOther#1 \relax}%
80     \PE@EscapeName#1%
81 }%

\DefEscapeString

82 \long\def\DefEscapeString#1#2{%
83     \PE@sanitize#1{#2}%
84     \edef#1{\expandafter\PE@SpaceToOther#1 \relax}%
85     \PE@EscapeString#1%
86 }%

87 \else

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

88 \long\def\PE@edefbabel#1#2#3{%
89     \begingroup
90     \csname @save@activetrue\endcsname
91     \edef#1{#2{#3}}%
92     \expandafter\endgroup
93     \expandafter\def\expandafter#1\expandafter{#1}%
94 }

\DefEscapeHex

95 \long\def\DefEscapeHex#1#2{%
96     \PE@edefbabel#1\pdfescapehex{#2}%
97 }%

\DefUnescapeHex

98 \def\DefUnescapeHex#1#2{%
99     \PE@edefbabel#1\pdfunescapehex{#2}%
100 }%

\DefEscapeName

101 \long\def\DefEscapeName#1#2{%
102     \PE@edefbabel#1\pdfescapename{#2}%
103 }%

\DefEscapeString

104 \long\def\DefEscapeString#1#2{%
105     \PE@edefbabel#1\pdfescapestring{#2}%
106 }%

107 \PE@AtEnd
108 \expandafter\endinput
109 \fi

```

2.4 Help macros

2.4.1 Characters

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

```

\PE@hash

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

\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{ }

```

\PE@backslash

```

115 \begingroup
116   \long\def\@gobble#1{}%
117   \escapechar=92 %
118 \edef\x{\endgroup
119   \def\noexpand\PE@backslash{\expandafter\@gobble\string\\}%
120 }
121 \x

```

2.4.2 Switch for ε -TeX

```

122 \newif\ifPE@etex
123 \begingroup\expandafter\expandafter\expandafter\endgroup
124 \expandafter\ifx\csname numexpr\endcsname\relax
125 \else
126   \PE@etextrue
127 \fi

```

2.5 Sanitizing

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

```

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

```

\PE@SpaceToOther

```

157 \def\PE@SpaceToOther#1 #2\relax{%
158   #1%
159   \ifx\\#2\\%
160   \else
161     \PE@space@other
162     \@ReturnAfterFi{%
163       \PE@SpaceToOther#2\relax
164     }%
165   \fi
166 }

```

\@ReturnAfterFi

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

2.6 Conversions

2.6.1 Conversion to hex string

\PE@EscapeHex

```
168 \ifPE@etex
169   \def\PE@EscapeHex#1{%
170     \edef#1{\expandafter\PE@ToHex#1\relax}%
171   }%
172 \else
173   \def\PE@EscapeHex#1{%
174     \def\PE@result{%
175       \expandafter\PE@ToHex#1\relax
176       \let#1\PE@result
177     }%
178 \fi
```

\PE@ToHex

```
179 \def\PE@ToHex#1{%
180   \ifx\relax#1%
181   \else
182     \PE@HexChar{#1}%
183     \expandafter\PE@ToHex
184   \fi
185 }%
```

\PE@HexChar

```
186 \ifPE@etex
187   \def\PE@HexChar#1{%
188     \PE@HexDigit{\numexpr\dimexpr.0625\dimexpr'#1sp\relax\relax\relax}%
189     \PE@HexDigit{%
190       \numexpr'#1-16*\dimexpr.0625\dimexpr'#1sp\relax\relax\relax
191     }%
192   }%
193 \else
194   \def\PE@HexChar#1{%
195     \dimen0='#1sp%
196     \dimen2=.0625\dimen0 %
197     \advance\dimen0-16\dimen2 %
198     \edef\PE@result{%
199       \PE@result
200       \PE@HexDigit{\dimen2 }%
201       \PE@HexDigit{\dimen0 }%
202     }%
203   }%
204 \fi
```

\PE@HexDigit

```
205 \def\PE@HexDigit#1{%
206   \expandafter\string
207   \ifcase#1%
208     0\or 1\or 2\or 3\or 4\or 5\or 6\or 7\or 8\or 9\or
209     A\or B\or C\or D\or E\or F%
210   \fi
211 }
```

2.6.2 Character code to octal number

\PE@OctChar

```
212 \ifPE@etex
213   \def\PE@OctChar#1{%
214     \expandafter\PE@OctChar
215       \the\numexpr\dimexpr.015625\dimexpr'#1sp\relax\relax
216       \expandafter\relax
217       \expandafter\relax
218       \the\numexpr\dimexpr.125\dimexpr'#1sp\relax\relax\relax
219       \relax
220       #1%
221   }%
222   \def\PE@@OctChar#1\relax#2\relax#3{%
223     \PE@backslash
224     #1%
225     \the\numexpr#2-8*#1\relax
226     \the\numexpr\dimexpr'#3sp\relax-8*#2\relax
227   }%
228 \else
229   \def\PE@OctChar#1{%
230     \dimen0='#1sp%
231     \dimen2=.125\dimen0 %
232     \dimen4=.125\dimen2 %
233     \advance\dimen0-8\dimen2 %
234     \advance\dimen2-8\dimen4 %
235     \edef\PE@result{%
236       \PE@result
237       \PE@backslash
238       \number\dimen4 %
239       \number\dimen2 %
240       \number\dimen0 %
241     }%
242   }%
243 \fi
```

2.6.3 Unpack hex string

\PE@UnescapeHex

```
244 \def\PE@UnescapeHex#1{%
245   \begingroup
246     \def\PE@result{}%
247     \expandafter\PE@DeHex#1\relax\relax
248   \expandafter\endgroup
249   \expandafter\def\expandafter#1\expandafter{\PE@result}%
250 }
```

\PE@DeHex

```
251 \def\PE@DeHex#1#2{%
252   \ifx#2\relax
253     \ifx#1\relax
254       \else
255         \PE@DeHex#10\relax\relax
256       \fi
257     \else
258       \uppercase{\lccode0="#1#2}\relax
259       \ifnum\lccode0=32 %
260         \edef\PE@result{\PE@result\PE@space@space}%
261       \else
262         \lowercase{\def\PE@temp{^^@}}%
263         \edef\PE@result{\PE@result\PE@temp}%
264       \fi
```

```

265     \expandafter\PE@DeHex
266     \fi
267 }

```

2.6.4 Conversion to PDF name

```
\PE@EscapeName
```

```

268 \ifPE@etex
269   \def\PE@EscapeName#1{%
270     \edef#1{\expandafter\PE@EscapeNameTokens#1\relax}%
271   }%
272 \else
273   \def\PE@EscapeName#1{%
274     \def\PE@result{}%
275     \expandafter\PE@EscapeNameTokens#1\relax
276     \let#1\PE@result
277   }%
278 \fi

```

```
\PE@EscapeNameTokens
```

```

279 \def\PE@EscapeNameTokens#1{%
280     \ifx\relax#1%
281     \else
282     \ifnum'#1<33 %
283     \ifcase'#1 %
284     % drop illegal zero
285     \else
286     \PE@EscapeNameAdd\PE@hash
287     \PE@HexChar#1%
288     \fi
289     \else
290     \ifnum'#1>126 %
291     \PE@EscapeNameAdd\PE@hash
292     \PE@HexChar#1%
293     \else \ifnum'#1=35 \PE@EscapeNameHashChar 23% #
294     \else\ifnum'#1=37 \PE@EscapeNameHashChar 25% %
295     \else\ifnum'#1=40 \PE@EscapeNameHashChar 28% (
296     \else\ifnum'#1=41 \PE@EscapeNameHashChar 29% )
297     \else\ifnum'#1=47 \PE@EscapeNameHashChar 2F% /
298     \else\ifnum'#1=60 \PE@EscapeNameHashChar 3C% <
299     \else\ifnum'#1=62 \PE@EscapeNameHashChar 3E% >
300     \else\ifnum'#1=91 \PE@EscapeNameHashChar 5B% [
301     \else\ifnum'#1=93 \PE@EscapeNameHashChar 5D% ]
302     \else\ifnum'#1=123 \PE@EscapeNameHashChar 7B% {
303     \else\ifnum'#1=125 \PE@EscapeNameHashChar 7D% }
304     \else
305     \PE@EscapeNameAdd{#1}%
306     \fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi
307     \fi
308     \fi
309     \expandafter\PE@EscapeNameTokens
310     \fi
311 }%
312 \def\PE@EscapeNameHashChar#1#2{%
313     \PE@EscapeNameAdd{\PE@hash\string#1\string#2}%
314 }%

```

```
\PE@EscapeNameAdd
```

```

315 \ifPE@etex
316   \def\PE@EscapeNameAdd#1{#1}%
317 \else
318   \def\PE@EscapeNameAdd#1{%

```



```

319 \edef\PE@result{%
320 \PE@result
321 #1%
322 }%
323 }%
324 \fi

```

2.6.5 Conversion to PDF string

\PE@EscapeString

```

325 \ifPE@etex
326 \def\PE@EscapeString#1{%
327 \edef#1{\expandafter\PE@EscapeStringTokens#1\relax}%
328 }%
329 \else
330 \def\PE@EscapeString#1{%
331 \begingroup
332 \def\PE@result{%
333 \expandafter\PE@EscapeStringTokens#1\relax
334 \expandafter\endgroup
335 \expandafter\def\expandafter#1\expandafter{\PE@result}%
336 }%
337 \fi

```

\PE@EscapeStringTokens

```

338 \def\PE@EscapeStringTokens#1{%
339 \ifx\relax#1%
340 \else
341 \ifnum'#1<33 %
342 \PE@OctChar#1%
343 \else
344 \ifnum'#1>126 %
345 \PE@OctChar#1%
346 \else \ifnum'#1=40 \PE@EscapeStringAdd{\string\}% (
347 \else\ifnum'#1=41 \PE@EscapeStringAdd{\string\)}% )
348 \else\ifnum'#1=92 \PE@EscapeStringAdd{\string\\}% \
349 \else
350 \PE@EscapeStringAdd{#1}%
351 \fi\fi\fi
352 \fi
353 \fi
354 \expandafter\PE@EscapeStringTokens
355 \fi
356 }%

```

\PE@EscapeStringAdd

```

357 \ifPE@etex
358 \def\PE@EscapeStringAdd#1{#1}%
359 \else
360 \def\PE@EscapeStringAdd#1{%
361 \edef\PE@result{%
362 \PE@result
363 #1%
364 }%
365 }%
366 \fi

367 \PE@AtEnd
368 \end{package}

```

3 Test

```
369 <*test1 | test2 | test3>
370 \NeedsTeXFormat{LaTeX2e}
371 \makeatletter
```

3.1 Test with \pdfescape... commands

```
372 <*test1>
373 \ProvidesFile{pdfescape-test1.tex}%
374 [2007/04/11 v1.3 Test with \string\pdfescape... commands]%
375 </test1>
```

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

```
376 <*test2>
377 \ProvidesFile{pdfescape-test2.tex}%
378 [2007/04/11 v1.3 Test without \string\pdfescape..., with e-TeX]%
379 </test2>
```

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

```
380 <*test3>
381 \ProvidesFile{pdfescape-test3.tex}%
382 [2007/04/11 v1.3 Test without \string\pdfescape... and e-TeX]%
383 </test3>
```

3.4 Check/ensure test preconditions

3.4.1 Check \pdfescape...

```
384 <*test1>
385 \@ifundefined{pdfescapehex}{%
386   \PackageError{pdfescape-test1}{%
387     Missing \string\pdfescape... commands%
388   }{Test aborted.}%
389   \stop
390 }{}
391 </test1>

392 <*test2 | test3>
393 \let\pdfescapehex\@undefined
394 \let\pdfunescapehex\@undefined
395 \let\pdfescapename\@undefined
396 \let\pdfescapestring\@undefined
397 </test2 | test3>
```

3.4.2 Check ε -TeX

```
398 <*test2>
399 \@ifundefined{numexpr}{%
400   \PackageError{pdfescape-test2}{%
401     Missing \eTeX
402   }{Test aborted.}%
403   \stop
404 }{}
405 </test2>
```

Package `qstest` uses ε -TeX, thus ε -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.

```
406 \RequirePackage{qstest}
407 \IncludeTests{*}
408 \LogTests{lgout}{*}{*}
```

```

409
410 \newcommand*{\ExpectVar}[2]{%
411   \Expect*{\ifx#1#2true\else false\fi}{true}%
412 }
413
414 \makeatletter
415 \begingroup
416   \gdef\AllBytes{%
417     \count@=0 %
418     \catcode0=12 %
419     \@whilenum\count@<256 \do{%
420       \lccode0=\count@
421       \ifnum\count@=32 %
422         \xdef\AllBytes{\AllBytes\space}%
423       \else
424         \lowercase{%
425           \xdef\AllBytes{\AllBytes^^@}%
426         }%
427       \fi
428       \advance\count@ by 1 %
429     }%
430 \endgroup
431 \newcommand*{\AllBytesHex}{%
432   000102030405060708090A0B0C0D0E0F%
433   101112131415161718191A1B1C1D1E1F%
434   202122232425262728292A2B2C2D2E2F%
435   303132333435363738393A3B3C3D3E3F%
436   404142434445464748494A4B4C4D4E4F%
437   505152535455565758595A5B5C5D5E5F%
438   606162636465666768696A6B6C6D6E6F%
439   707172737475767778797A7B7C7D7E7F%
440   808182838485868788898A8B8C8D8E8F%
441   909192939495969798999A9B9C9D9E9F%
442   A0A1A2A3A4A5A6A7A8A9AABACADAFAF%
443   B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF%
444   C0C1C2C3C4C5C6C7C8C9CACBCCDCECF%
445   D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF%
446   E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF%
447   F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF%
448 }
449 \@onelevel@sanitize\AllBytesHex
450 \expandafter\lowercase\expandafter{%
451   \expandafter\newcommand\expandafter*\expandafter\AllBytesHexLC
452     \expandafter{\AllBytesHex}%
453 }
454 \newcommand*{\AllBytesName}{%
455 \begingroup
456   \catcode'\#12 %
457   \xdef\AllBytesName{%
458     #01#02#03#04#05#06#07#08#09#0A#0B#0C#0D#0E#0F%
459     #10#11#12#13#14#15#16#17#18#19#1A#1B#1C#1D#1E#1F%
460     #20!"#23$#25&'#28#29*+,-.#2F%
461     0123456789:;#3C=#3E?%
462     @ABCDEFGHIJKLMNO%
463     PQRSTUVWXYZ#5B\@backslashchar#5D^_%
464     `abcdefghijklmnopqrstuvwxyz%
465     pqrstuvwxyz7B|#7D\string~#7F%
466     #80#81#82#83#84#85#86#87#88#89#8A#8B#8C#8D#8E#8F%
467     #90#91#92#93#94#95#96#97#98#99#9A#9B#9C#9D#9E#9F%
468     #A0#A1#A2#A3#A4#A5#A6#A7#A8#A9#AA#AB#AC#AD#AE#AF%
469     #B0#B1#B2#B3#B4#B5#B6#B7#B8#B9#BA#BB#BC#BD#BE#BF%
470     #C0#C1#C2#C3#C4#C5#C6#C7#C8#C9#CA#CB#CC#CD#CE#CF%

```

```

471      #D0#D1#D2#D3#D4#D5#D6#D7#D8#D9#DA#DB#DC#DD#DE#DF%
472      #E0#E1#E2#E3#E4#E5#E6#E7#E8#E9#EA#EB#EC#ED#EE#EF%
473      #F0#F1#F2#F3#F4#F5#F6#F7#F8#F9#FA#FB#FC#FD#FE#FF%
474  }%
475 \endgroup
476 \@onelevel@sanitize\AllBytesName
477
478 \newcommand*{\AllBytesString}{%
479 \begin{group
480   \def\{|}%
481   \edef%\{@percentchar}%
482   \catcode'\|=0 %
483   \catcode'\#=12 %
484   \catcode'\~=12 %
485   \catcode'\|=12 %
486   \xdef\AllBytesString{%
487     \000\001\002\003\004\005\006\007\010\011\012\013\014\015\016\017%
488     \020\021\022\023\024\025\026\027\030\031\032\033\034\035\036\037%
489     \040!"#$%&'(\)*+,-./%
490     0123456789:;<=>?%
491     @ABCDEFGHIJKLMNO%
492     PQRSTUVWXYZ[\]^_%
493     'abcdefghijklmno%
494     pqrstuvwxyz{|}~\177%
495     \200\201\202\203\204\205\206\207\210\211\212\213\214\215\216\217%
496     \220\221\222\223\224\225\226\227\230\231\232\233\234\235\236\237%
497     \240\241\242\243\244\245\246\247\250\251\252\253\254\255\256\257%
498     \260\261\262\263\264\265\266\267\270\271\272\273\274\275\276\277%
499     \300\301\302\303\304\305\306\307\310\311\312\313\314\315\316\317%
500     \320\321\322\323\324\325\326\327\330\331\332\333\334\335\336\337%
501     \340\341\342\343\344\345\346\347\350\351\352\353\354\355\356\357%
502     \360\361\362\363\364\365\366\367\370\371\372\373\374\375\376\377%
503   }%
504 \endgroup
505 \@onelevel@sanitize\AllBytesString
506
507 <*test3>
508 \let\org@detokenize\detokenize
509 \let\detokenize@undefined
510 \let\org@numexpr\numexpr
511 \let\numexpr@undefined
512 </test3>
513 \RequirePackage{pdfescape}
514 <*test3>
515 \let\detokenize\org@detokenize
516 \let\numexpr\org@numexpr
517 </test3>
518
519 \begin{qstest}{all-hex}{\AllBytes, escapehex}
520   \EdefEscapeHex\x{\AllBytes}%
521   \Expect*{\x}*{\AllBytesHex}%
522   \ExpectVar\x\AllBytesHex
523 \end{qstest}
524
525 \begin{qstest}{all-unhex}{\AllBytesHex, unescapehex}
526   \EdefUnescapeHex\x{\AllBytesHex}%
527   \Expect*{\x}*{\AllBytes}%
528   \ExpectVar\x\AllBytes
529 \end{qstest}
530
531 \begin{qstest}{all-unhex-lc}{\AllBytesHexLC, unescapehex, lowercase}
532   \EdefUnescapeHex\x{\AllBytesHexLC}%

```

```

533 \Expect*{\x}*{\AllBytes}%
534 \ExpectVar\x\AllBytes
535 \end{qstest}
536
537 \begin{qstest}{unhex-incomplete}{unescapehex, incomplete}
538 \EdefUnescapeHex\x{4}%
539 \Expect*{\x}-{0}%
540 \end{qstest}
541
542 \begin{qstest}{unhex-space}{unescapehex, space}
543 \EdefUnescapeHex\x{20}%
544 \Expect*{\x}-{ }%
545 \ExpectVar\x\space
546 \end{qstest}
547
548 \begin{qstest}{unhex-spaces}{unescapehex, spaces}
549 \EdefUnescapeHex\x{204020204120}%
550 \def\y#1{%
551 \edef\z{#1\string @#1#1\string A#1}%
552 } \y{ }%
553 \Expect*{\x}*{\z}%
554 \ExpectVar\x\z
555 \end{qstest}
556
557 \begin{qstest}{all-name}{\AllBytes, escapename}
558 \EdefEscapeName\x{\AllBytes}%
559 \Expect*{\x}*{\AllBytesName}%
560 \ExpectVar\x\AllBytesName
561 \end{qstest}
562
563 \begin{qstest}{all-string}{\AllBytes, escapestring}
564 \EdefEscapeString\x{\AllBytes}%
565 \Expect*{\x}*{\AllBytesString}%
566 \ExpectVar\x\AllBytesString
567 \end{qstest}
568
569 \stop
570 </test1 | test2 | test3>

```

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](#)

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
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 `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 L^AT_EX. The `.dtx` chooses its action depending on the format:

plain-`TEX`: Run `docstrip` and extract the files.

L^AT_EX: Generate the documentation.

If you insist on using L^AT_EX for `docstrip` (really, `docstrip` does not need L^AT_EX), 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 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.

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 | | |
|------------------------------------|-----------------|---|
| <code>\#</code> | 456, 483 | <code>\@ifundefined</code> 385, 399 |
| <code>\%</code> | 481 | <code>\@onelevel@sanitize</code> 145, 449, 476, 505 |
| <code>\(</code> | 346, 489 | <code>\@percentchar</code> 481 |
| <code>\)</code> | 347, 489 | <code>\@undefined</code> 393, 394, 395, 396, 509, 511 |
| <code>\@ReturnAfterFi</code> | 162, <u>167</u> | <code>\@whilenum</code> 419 |
| <code>\@backslashchar</code> | 463 | <code>\@</code> 119, 159, 348, 485, 492 |
| <code>\@gobble</code> | 116, 119 | <code>\ </code> 480, 482 |
| | | <code>\~</code> 484 |

| | |
|------------------------------|---|
| Numbers | |
| \0 | 487, 488, 489 |
| \1 | 494 |
| \2 | 495, 496, 497, 498 |
| \3 | 499, 500, 501, 502 |
| _ | 112, 348 |
| A | |
| \advance | 197, 233, 234, 428 |
| \AllBytes | 416, 422, 425, 519, 520, 527, 528, 533, 534, 557, 558, 563, 564 |
| \AllBytesHex | 431, 449, 452, 521, 522, 525, 526 |
| \AllBytesHexLC | 451, 531, 532 |
| \AllBytesName | 454, 457, 476, 559, 560 |
| \AllBytesString | 478, 505, 565, 566 |
| B | |
| \begin | 519, 525, 531, 537, 542, 548, 557, 563 |
| C | |
| \catcode | 43, 45, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 112, 418, 456, 482, 483, 484, 485 |
| \count@ | 417, 419, 420, 421, 428 |
| \csname | 3, 13, 26, 39, 42, 67, 90, 124, 129, 132, 139, 150 |
| D | |
| \detokenize | 152, 508, 509, 515 |
| \dimen | 195, 196, 197, 200, 201, 230, 231, 232, 233, 234, 238, 239, 240 |
| \dimexpr | 188, 190, 215, 218, 226 |
| \do | 419 |
| E | |
| \EdefEscapeHex | 2, 68, 95, 520 |
| \EdefEscapeName | 77, 101, 558 |
| \EdefEscapeString | 82, 104, 564 |
| \EdefUnescapeHex | 73, 98, 526, 532, 538, 543, 549 |
| \empty | 7 |
| \end | 523, 529, 535, 540, 546, 555, 561, 567 |
| \endcsname | 3, 13, 26, 39, 42, 67, 90, 124, 129, 132, 139, 150 |
| \endinput | 22, 108 |
| \escapechar | 117 |
| \eTeX | 401 |
| \Expect | 411, 521, 527, 533, 539, 544, 553, 559, 565 |
| \ExpectVar | 410, 522, 528, 534, 545, 554, 560, 566 |
| G | |
| \gdef | 416 |
| I | |
| \ifcase | 4, 207, 283 |
| \ifnum | 259, 282, 290, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 341, 344, 346, 347, 348, 421 |
| \ifPE@etex | 122, 168, 186, 212, 268, 315, 325, 357 |
| \ifx | 5, 7, 13, 26, 34, 67, 124, 129, 139, 159, 180, 252, 253, 280, 339, 411 |
| \immediate | 15, 28 |
| \IncludeTests | 407 |
| L | |
| \lccode | 258, 259, 420 |
| \LogTests | 408 |
| \lowercase | 262, 424, 450 |
| M | |
| \makeatletter | 371, 414 |
| \meaning | 141 |
| N | |
| \NeedsTeXFormat | 370 |
| \newcommand | 410, 431, 451, 454, 478 |
| \newif | 122 |
| \number | 238, 239, 240 |
| \numexpr | 188, 190, 215, 218, 225, 226, 510, 511, 516 |
| O | |
| \org@detokenize | 508, 515 |
| \org@numexpr | 510, 516 |
| P | |
| \PackageError | 386, 400 |
| \PackageInfo | 18 |
| \pdfescape | 374, 378, 382, 387 |
| \pdfescapehex | 96, 393 |
| \pdfescapename | 102, 395 |
| \pdfescapestring | 105, 396 |
| \pdfunescapehex | 99, 394 |
| \PE@OctChar | 214, 222 |
| \PE@AtEnd | 47, 48, 107, 367 |
| \PE@backslash | 115, 223, 237 |
| \PE@DeHex | 247, 251 |
| \PE@edefbabel | 88, 96, 99, 102, 105 |
| \PE@EnsureCode | 46, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65 |
| \PE@EscapeHex | 71, 168 |
| \PE@EscapeName | 80, 268 |
| \PE@EscapeNameAdd | 286, 291, 305, 313, 315 |
| \PE@EscapeNameHashChar | 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 312 |
| \PE@EscapeNameTokens | 270, 275, 279 |
| \PE@EscapeString | 85, 325 |
| \PE@EscapeStringAdd | 346, 347, 348, 350, 357 |
| \PE@EscapeStringTokens | 327, 333, 338 |
| \PE@etexttrue | 126 |
| \PE@hash | 110, 286, 291, 313 |
| \PE@HexChar | 182, 186, 287, 292 |
| \PE@HexDigit | 188, 189, 200, 201, 205 |
| \PE@OctChar | 212, 342, 345 |

