The pdfescape package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2007/04/21 v1.4

Abstract

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

Contents

1 Documentation **2**
1.1 Additional unescape macros . . . 2

2 Implementation **2**
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 `\EdefUnescapeName` . . . 5
2.5 `\EdefUnescapeString` . . . 7
2.6 User macros (pdfTeXanalogues) . . . 9
2.7 Help macros . . . 10
2.7.1 Characters . . . 10
2.7.2 Switch for ε-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

3 Test **15**
3.1 Test with `\pdfescape...` commands . . . 15
3.2 Test without `\pdfescape...`, with ε-TeX . . . 15
3.3 Test without `\pdfescape...` and ε-TeX . . . 15
3.4 Check/ensure test preconditions . . . 16
3.4.1 Check `\pdfescape...` . . . 16
3.4.2 Check ε-TeX . . . 16
3.5 Common part . . . 16

4 Installation **22**
4.1 Download . . . 22
4.2 Bundle installation . . . 22
4.3 Package installation . . . 22
4.4 Refresh file name databases . . . 23
4.5 Some details for the interested . . . 23

5 History **23**
[2007/02/21 v1.0] 23
[2007/02/25 v1.1] 24
[2007/03/20 v1.2] 24
[2007/04/11 v1.3] 24
[2007/04/21 v1.4] 24

6 Index **24**

1 Documentation

`\EdefEscapeHex` {⟨*cmd*⟩} {⟨*string*⟩}
`\EdefUnescapeHex` {⟨*cmd*⟩} {⟨*string*⟩}
`\EdefEscapeName` {⟨*cmd*⟩} {⟨*string*⟩}
`\EdefEscapeString` {⟨*cmd*⟩} {⟨*string*⟩}

These commands converts ⟨*string*⟩ and stores the result in macro ⟨*cmd*⟩. The conversion result is the same as the conversion of the corresponding pdfTEX's primitives. Note that the argument ⟨*string*⟩ is expanded before the conversion.

For example, if pdfTEX >= 1.30 is present, then `\EdefEscapeHex` becomes to:

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

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

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

1.1 Additional unescape macros

`\EdefUnescapeName` {⟨*cmd*⟩} {⟨*string*⟩}

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

`\EdefUnescapeString` {⟨*cmd*⟩} {⟨*string*⟩}

Macro ⟨*cmd*⟩ stores the unescaped string in ⟨*string*⟩. 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\csname PE@AtEnd\endcsname{%
3   \catcode64 \the\catcode64\relax
4 }
```

```
\catcode64 11 % @
\def\PE@EnsureCode#1#2#3{%
  \edef\PE@AtEnd{%
    \PE@AtEnd
    #1#2 \the#1#2\relax
  }%
  #1#2 #3\relax
}
\PE@EnsureCode\catcode{0}{12}% ^^@
\PE@EnsureCode\catcode{34}{12}% "
\PE@EnsureCode\catcode{42}{12}% *
\PE@EnsureCode\catcode{45}{12}% -
\PE@EnsureCode\catcode{46}{12}% .
\PE@EnsureCode\catcode{60}{12}% <
\PE@EnsureCode\catcode{61}{12}% =
\PE@EnsureCode\catcode{62}{12}% >
\PE@EnsureCode\catcode{94}{7}% ^
\PE@EnsureCode\catcode{96}{12}% `
\PE@EnsureCode\uccode{34}{0}% "
\PE@EnsureCode\uccode{48}{0}% 0
\PE@EnsureCode\uccode{61}{0}% =
```

2.2 Reload check and package identification

Reload check, especially if the package is not used with LaTeX.

```
\begingroup
  \catcode44 12 % ,
  \catcode45 12 % -
  \catcode46 12 % .
  \catcode58 12 % :
  \catcode64 11 % @
  \expandafter\let\expandafter\x\csname ver@pdfescape.sty\endcsname
  \ifcase 0%
    \ifx\x\relax % plain
    \else
      \ifx\x\empty % LaTeX
      \else
        1%
      \fi
    \fi
  \else
    \expandafter\ifx\csname PackageInfo\endcsname\relax
      \def\x#1#2{%
        \immediate\write-1{Package #1 Info: #2.}%
      }%
    \else
      \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
    \fi
    \x{pdfescape}{The package is already loaded}%
    \endgroup
    \expandafter\endinput
  \fi
\endgroup
```

Package identification:

```
\begingroup
  \catcode40 12 % (
  \catcode41 12 % )
  \catcode44 12 % ,
  \catcode45 12 % -
  \catcode46 12 % .
  \catcode47 12 % /
  \catcode58 12 % :
```

```
  \catcode64 11 % @
  \expandafter\ifx\csname ProvidesPackage\endcsname\relax
    \def\x#1#2#3[#4]{\endgroup
      \immediate\write-1{Package: #3 #4}%
      \xdef#1{#4}%
    }%
  \else
    \def\x#1#2[#3]{\endgroup
      #2[{#3}]%
      \ifx#1\relax
        \xdef#1{#3}%
      \fi
    }%
  \fi
\expandafter\x\csname ver@pdfescape.sty\endcsname
\ProvidesPackage{pdfescape}%
  [2007/04/21 v1.4 Provides hex, PDF name and string conversions (HO)]
```

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`.

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname detokenize\endcsname\relax
  \long\def\PE@sanitize#1#2{%
    \begingroup
      \csname @safe@activestrue\endcsname
      \edef#1{#2}%
      \PE@onelevel@sanitize#1%
    \expandafter\endgroup
    \expandafter\def\expandafter#1\expandafter{#1}%
  }%
  \begingroup\expandafter\expandafter\expandafter\endgroup
  \expandafter\ifx\csname @onelevel@sanitize\endcsname\relax
    \def\PE@onelevel@sanitize#1{%
      \edef#1{\expandafter\PE@strip@prefix\meaning#1}%
    }%
    \def\PE@strip@prefix#1>{}%
  \else
    \let\PE@onelevel@sanitize\@onelevel@sanitize
  \fi
\else
  \long\def\PE@sanitize#1#2{%
    \begingroup
      \csname @safe@activestrue\endcsname
      \edef#1{#2}%
      \PE@onelevel@sanitize#1%
    \expandafter\endgroup
    \expandafter\def\expandafter#1\expandafter{#1}%
  }%
  \def\PE@onelevel@sanitize#1{%
    \edef#1{\detokenize\expandafter{#1}}%
  }%
\fi
```

2.3.1 Space characters

`\PE@space@other`

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

\PE@space@space

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

2.3.2 Space normalization

\PE@SanitizeSpaceOther

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

\PE@SpaceToOther

```
\def\PE@SpaceToOther#1 #2\relax{%
  #1%
  \ifx\\#2\\%
  \else
    \PE@space@other
    \@ReturnAfterFi{%
      \PE@SpaceToOther#2\relax
    }%
  \fi
}
```

\@ReturnAfterFi

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

2.4 \EdefUnescapeName

\EdefUnescapeName

```
\def\EdefUnescapeName#1#2{%
  \PE@sanitize#1{#2}%
  \PE@SanitizeSpaceOther#1%
  \PE@UnescapeName#1%
  \PE@onelevel@sanitize#1%
}
```

\PE@UnescapeName

```
\begingroup
  \catcode`\$=6  % hash
  \catcode`\#=12 % other
  \gdef\PE@UnescapeName$1{%
    \begingroup
      \PE@InitUccodeHexDigit
      \def\PE@result{}%
      \expandafter\PE@DeName$1#\relax\relax
    \expandafter\endgroup
    \expandafter\def\expandafter$1\expandafter{\PE@result}%
  }%
  \gdef\PE@DeName$1#$2$3{%
    \ifx\relax$2%
      \edef\PE@result{\PE@result$1}%
      \let\PE@next\relax
    \else
      \ifx\relax$3%
        % wrong escape sequence in input
        \edef\PE@result{\PE@result$1#}%
        \let\PE@next\relax
      \else
        \uppercase{%
          \def\PE@testA{$2}%
          \def\PE@testB{$3}%
```

```
        }%
        \ifcase\ifcase\expandafter\PE@TestUcHexDigit\PE@testA
                 \ifcase\expandafter\PE@TestUcHexDigit\PE@testB
                   \z@
                 \else
                   \@ne
                 \fi
               \else
                 \@ne
               \fi
          \uccode\z@="\PE@testA\PE@testB\relax
          \uppercase{%
            \def\PE@temp{^^@}%
          }%
          \uccode\z@=\z@
          \edef\PE@result{\PE@result$1\PE@temp}%
          \let\PE@next\PE@DeName
        \else
          % wrong escape sequence in input
          \edef\PE@result{\PE@result$1#}%
          \def\PE@next{\PE@DeName$2$3}%
        \fi
      \fi
    \fi
    \PE@next
  }%
\endgroup
```

\PE@InitUccodeHexDigit

```
\def\PE@InitUccodeHexDigit{%
  \uccode`a=`A\relax
  \uccode`b=`B\relax
  \uccode`c=`C\relax
  \uccode`d=`D\relax
  \uccode`e=`E\relax
  \uccode`f=`F\relax
  \uccode`A=\z@
  \uccode`B=\z@
  \uccode`C=\z@
  \uccode`D=\z@
  \uccode`E=\z@
  \uccode`F=\z@
  \uccode`0=\z@
  \uccode`1=\z@
  \uccode`2=\z@
  \uccode`3=\z@
  \uccode`4=\z@
  \uccode`5=\z@
  \uccode`6=\z@
  \uccode`7=\z@
  \uccode`8=\z@
  \uccode`9=\z@
}
```

\PE@TestUcHexDigit

```
\def\PE@TestUcHexDigit#1{%
  \ifnum`#1<48 % 0
    \@ne
  \else
    \ifnum`#1>70 % F
      \@ne
    \else
```

```
      \ifnum`#1>57 % 9
        \ifnum`#1<65 % A
          \@ne
        \else
          \z@
        \fi
      \else
        \z@
      \fi
    \fi
  \fi
}
```

2.5 \EdefUnescapeString

\EdefUnescapeString

```
\def\EdefUnescapeString#1#2{%
  \PE@sanitize#1{#2}%
  \PE@SanitizeSpaceOther#1%
  \PE@NormalizeLineEnd#1%
  \PE@UnescapeString#1%
  \PE@onelevel@sanitize#1%
}

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

\PE@NormalizeLineEnd

```
  \def\PE@NormalizeLineEnd##1{%
    \def\PE@result{}%
    \expandafter\PE@@NormalizeLineEnd##1#2\relax
    \let##1\PE@result
  }%
```

\PE@@NormalizeLineEnd

```
  \def\PE@@NormalizeLineEnd##1#2##2{%
    \ifx\relax##2%
      \edef\PE@result{\PE@result##1}%
      \let\PE@next\relax
    \else
      \edef\PE@result{\PE@result##1#1}%
      \ifx#1##2% lf
        \let\PE@next\PE@@NormalizeLineEnd
      \else
        \def\PE@next{\PE@@NormalizeLineEnd##2}%
      \fi
    \fi
    \PE@next
  }%
}%
\uppercase{%
  \x 89%
}

\begingroup
  \catcode`\|=0 %
  \catcode`\\=12 %
```

\PE@UnescapeString

```
  |gdef|PE@UnescapeString#1{%
```

```
    |begingroup
      |def|PE@result{}%
      |expandafter|PE@DeString#1\|relax
    |expandafter|endgroup
    |expandafter|def|expandafter#1|expandafter{|PE@result}%
  }%
```

\PE@DeString

```
  |gdef|PE@DeString#1\#2{%
    |ifx|relax#2%
      |edef|PE@result{|PE@result#1}%
      |let|PE@next|relax
    |else
      |if n#2%
        |uccode|z@=10 %
      |else|if r#2%
        |uccode|z@=13 %
      |else|if t#2%
        |uccode|z@=9 %
      |else|if b#2%
        |uccode|z@=8 %
      |else|if f#2%
        |uccode|z@=12 %
      |else
        |uccode|z@=|z@
      |fi|fi|fi|fi|fi
      |ifnum|uccode|z@>|z@
        |uppercase{%
          |edef|PE@temp{^^@}%
        }%
        |edef|PE@result{|PE@result#1|PE@temp}%
        |let|PE@next|PE@DeString
      |else
        |if\#2% backslash
          |edef|PE@result{|PE@result#1}%
          |let|PE@next|PE@CheckEndBackslash
        |else
          |ifnum`#2=10 % linefeed
            |edef|PE@result{|PE@result#1}%
            |let|PE@next|PE@DeString
          |else
            |ifcase|PE@TestOctDigit#2%
              |edef|PE@result{|PE@result#1}%
              |def|PE@next{|PE@OctI#2}%
            |else
              |edef|PE@result{|PE@result#1#2}%
              |let|PE@next|PE@DeString
            |fi
          |fi
        |fi
      |fi
    |fi
    |PE@next
  }%
```

\PE@CheckEndBackslash

```
  |gdef|PE@CheckEndBackslash#1{%
    |ifx|relax#1%
    |else
      |edef|PE@result{|PE@result\}%
      |expandafter|PE@DeString|expandafter#1%
    |fi
```

```
  }%
|endgroup
```

\PE@TestOctDigit

```
\def\PE@TestOctDigit#1{%
  \ifnum`#1<48 % 0
    \@ne
  \else
    \ifnum`#1>55 % 7
      \@ne
    \else
      \z@
    \fi
  \fi
}
```

\PE@OctI

```
\def\PE@OctI#1#2{%
  \ifcase\PE@TestOctDigit#2%
    \def\PE@next{\PE@OctII{#1#2}}%
  \else
    \def\PE@next{\PE@OctAll#1#2}%
  \fi
  \PE@next
}
```

\PE@OctII

```
\def\PE@OctII#1#2{%
  \ifcase\PE@TestOctDigit#2%
    \def\PE@next{\PE@OctAll{#1#2}}%
  \else
    \def\PE@next{\PE@OctAll{#1}#2}%
  \fi
  \PE@next
}
```

\PE@OctAll

```
\def\PE@OctAll#1{%
  \uccode\z@=’#1\relax
  \uppercase{%
    \edef\PE@result{\PE@result^^@}%
  }%
  \PE@DeString
}
```

2.6 User macros (pdfTEX analogues)

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname pdfescapehex\endcsname\relax
```

\EdefEscapeHex

```
  \long\def\EdefEscapeHex#1#2{%
    \PE@sanitize#1{#2}%
    \PE@SanitizeSpaceOther#1%
    \PE@EscapeHex#1%
  }%
```

\EdefUnescapeHex

```
  \def\EdefUnescapeHex#1#2{%
    \PE@sanitize#1{#2}%
    \PE@UnescapeHex#1%
  }%
```

\EdefEscapeName

```
  \long\def\EdefEscapeName#1#2{%
    \PE@sanitize#1{#2}%
    \PE@SanitizeSpaceOther#1%
    \PE@EscapeName#1%
  }%
```

\EdefEscapeString

```
  \long\def\EdefEscapeString#1#2{%
    \PE@sanitize#1{#2}%
    \PE@SanitizeSpaceOther#1%
    \PE@EscapeString#1%
  }%

\else
```

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

```
  \long\def\PE@edefbabel#1#2#3{%
    \begingroup
      \csname @save@activestrue\endcsname
      \edef#1{#2{#3}}%
    \expandafter\endgroup
    \expandafter\def\expandafter#1\expandafter{#1}%
  }%
```

\EdefEscapeHex

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

\EdefUnescapeHex

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

\EdefEscapeName

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

\EdefEscapeString

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

  \PE@AtEnd
  \expandafter\endinput
\fi
```

2.7 Help macros

2.7.1 Characters

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

\PE@hash

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

\PE@backslash

```
\begingroup
  \escapechar=-1 %
\edef\x{\endgroup
  \def\noexpand\PE@backslash{\string\\}%
}
\x
```

2.7.2 Switch for ε-TEX

```
\newif\ifPE@etex
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname numexpr\endcsname\relax
\else
  \PE@etextrue
\fi
```

2.8 Conversions

2.8.1 Conversion to hex string

\PE@EscapeHex

```
\ifPE@etex
  \def\PE@EscapeHex#1{%
    \edef#1{\expandafter\PE@ToHex#1\relax}%
  }%
\else
  \def\PE@EscapeHex#1{%
    \def\PE@result{}%
    \expandafter\PE@ToHex#1\relax
    \let#1\PE@result
  }%
\fi
```

\PE@ToHex

```
\def\PE@ToHex#1{%
  \ifx\relax#1%
  \else
    \PE@HexChar{#1}%
    \expandafter\PE@ToHex
  \fi
}%
```

\PE@HexChar

```
\ifPE@etex
  \def\PE@HexChar#1{%
    \PE@HexDigit{\numexpr\dimexpr.0625\dimexpr`#1sp\relax\relax\relax}%
    \PE@HexDigit{%
      \numexpr`#1-16*\dimexpr.0625\dimexpr`#1sp\relax\relax\relax
    }%
  }%
\else
  \def\PE@HexChar#1{%
    \dimen0=`#1sp%
    \dimen2=.0625\dimen0 %
    \advance\dimen0-16\dimen2 %
    \edef\PE@result{%
      \PE@result
      \PE@HexDigit{\dimen2 }%
      \PE@HexDigit{\dimen0 }%
    }%
  }%
\fi
```

\PE@HexDigit

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

2.8.2 Character code to octal number

\PE@OctChar

```
\ifPE@etex
  \def\PE@OctChar#1{%
    \expandafter\PE@@OctChar
        \the\numexpr\dimexpr.015625\dimexpr`#1sp\relax\relax
            \expandafter\relax
        \expandafter\relax
        \the\numexpr\dimexpr.125\dimexpr`#1sp\relax\relax\relax
        \relax
        #1%
  }%
  \def\PE@@OctChar#1\relax#2\relax#3{%
    \PE@backslash
    #1%
    \the\numexpr#2-8*#1\relax
    \the\numexpr\dimexpr`#3sp\relax-8*#2\relax
  }%
\else
  \def\PE@OctChar#1{%
    \dimen0=`#1sp%
    \dimen2=.125\dimen0 %
    \dimen4=.125\dimen2 %
    \advance\dimen0-8\dimen2 %
    \advance\dimen2-8\dimen4 %
    \edef\PE@result{%
      \PE@result
      \PE@backslash
      \number\dimen4 %
      \number\dimen2 %
      \number\dimen0 %
    }%
  }%
\fi
```

2.8.3 Unpack hex string

\PE@UnescapeHex

```
\def\PE@UnescapeHex#1{%
  \begingroup
    \PE@InitUccodeHexDigit
    \def\PE@result{}%
    \expandafter\PE@DeHex#1\relax\relax
  \expandafter\endgroup
  \expandafter\def\expandafter#1\expandafter{\PE@result}%
}
```

\PE@DeHex

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

```
      \let\PE@next\relax
    \else
      \uppercase{%
        \def\PE@testA{#1}%
      }%
      \ifcase\expandafter\PE@TestUcHexDigit\PE@testA
        \def\PE@next{%
          \PE@DeHex#10\relax\relax
        }%
      \else
        \let\PE@next\relax
      \fi
    \fi
  \else
    \uppercase{%
      \def\PE@testA{#1}%
      \def\PE@testB{#2}%
    }%
    \ifcase\expandafter\PE@TestUcHexDigit\PE@testA
      \ifcase\expandafter\PE@TestUcHexDigit\PE@testB
        \uccode\z@="\PE@testA\PE@testB\relax
        \ifnum\uccode\z@=32 %
          \let\PE@temp\PE@space@space
        \else
          \uppercase{%
            \def\PE@temp{^^@}%
          }%
        \fi
        \edef\PE@result{\PE@result\PE@temp}%
        \let\PE@next\PE@DeHex
      \else
        % invalid input sequence
        \def\PE@next{%
          \PE@DeHex#1%
        }%
      \fi
    \else
      % invalid input sequence
      \def\PE@next{\PE@DeHex#2}%
    \fi
  \fi
  \PE@next
}
```

2.8.4 Conversion to PDF name

\PE@EscapeName

```
\ifPE@etex
  \def\PE@EscapeName#1{%
    \edef#1{\expandafter\PE@EscapeNameTokens#1\relax}%
  }%
\else
  \def\PE@EscapeName#1{%
    \def\PE@result{}%
    \expandafter\PE@EscapeNameTokens#1\relax
    \let#1\PE@result
  }%
\fi
```

\PE@EscapeNameTokens

```
\def\PE@EscapeNameTokens#1{%
  \ifx\relax#1%
```

```
  \else
    \ifnum`#1<33 %
      \ifcase`#1 %
        % drop illegal zero
      \else
        \PE@EscapeNameAdd\PE@hash
        \PE@HexChar#1%
      \fi
    \else
      \ifnum`#1>126 %
        \PE@EscapeNameAdd\PE@hash
        \PE@HexChar#1%
      \else  \ifnum`#1=35  \PE@EscapeNameHashChar 23% #
        \else\ifnum`#1=37  \PE@EscapeNameHashChar 25% %
        \else\ifnum`#1=40  \PE@EscapeNameHashChar 28% (
        \else\ifnum`#1=41  \PE@EscapeNameHashChar 29% )
        \else\ifnum`#1=47  \PE@EscapeNameHashChar 2F% /
        \else\ifnum`#1=60  \PE@EscapeNameHashChar 3C% <
        \else\ifnum`#1=62  \PE@EscapeNameHashChar 3E% >
        \else\ifnum`#1=91  \PE@EscapeNameHashChar 5B% [
        \else\ifnum`#1=93  \PE@EscapeNameHashChar 5D% ]
        \else\ifnum`#1=123 \PE@EscapeNameHashChar 7B% {
        \else\ifnum`#1=125 \PE@EscapeNameHashChar 7D% }
        \else
          \PE@EscapeNameAdd{#1}%
        \fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi
      \fi
    \fi
    \expandafter\PE@EscapeNameTokens
  \fi
}%
\def\PE@EscapeNameHashChar#1#2{%
  \PE@EscapeNameAdd{\PE@hash\string#1\string#2}%
}%
```

`\PE@EscapeNameAdd`

```
\ifPE@etex
  \def\PE@EscapeNameAdd#1{#1}%
\else
  \def\PE@EscapeNameAdd#1{%
    \edef\PE@result{%
      \PE@result
      #1%
    }%
  }%
\fi
```

2.8.5 Conversion to PDF string

`\PE@EscapeString`

```
\ifPE@etex
  \def\PE@EscapeString#1{%
    \edef#1{\expandafter\PE@EscapeStringTokens#1\relax}%
  }%
\else
  \def\PE@EscapeString#1{%
    \begingroup
      \def\PE@result{}%
      \expandafter\PE@EscapeStringTokens#1\relax
    \expandafter\endgroup
    \expandafter\def\expandafter#1\expandafter{\PE@result}%
  }%
```

```
\fi
```

\PE@EscapeStringTokens

```
\def\PE@EscapeStringTokens#1{%
  \ifx\relax#1%
  \else
    \ifnum`#1<33 %
      \PE@OctChar#1%
    \else
      \ifnum`#1>126 %
        \PE@OctChar#1%
      \else  \ifnum`#1=40 \PE@EscapeStringAdd{\string\(}% (
        \else\ifnum`#1=41 \PE@EscapeStringAdd{\string\)}% )
        \else\ifnum`#1=92 \PE@EscapeStringAdd{\string\\}% \
        \else
          \PE@EscapeStringAdd{#1}%
        \fi\fi\fi
      \fi
    \fi
    \expandafter\PE@EscapeStringTokens
  \fi
}%
```

\PE@EscapeStringAdd

```
\ifPE@etex
  \def\PE@EscapeStringAdd#1{#1}%
\else
  \def\PE@EscapeStringAdd#1{%
    \edef\PE@result{%
      \PE@result
      #1%
    }%
  }%
\fi
```

```
\PE@AtEnd
```

⟨/package⟩

3 Test

⟨*test1 | test2 | test3⟩

```
\NeedsTeXFormat{LaTeX2e}
\makeatletter
```

3.1 Test with \pdfescape... commands

⟨*test1⟩

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

⟨/test1⟩

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

⟨*test2⟩

```
\ProvidesFile{pdfescape-test2.tex}%
    [2007/04/21 v1.4 Test without \string\pdfescape..., with e-TeX]%
```

⟨/test2⟩

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

⟨*test3⟩

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

```
⟨/test3⟩
```

3.4 Check/ensure test preconditions

3.4.1 Check \pdfescape...

```
⟨*test1⟩
\@ifundefined{pdfescapehex}{%
  \PackageError{pdfescape-test1}{%
    Missing \string\pdfescape... commands%
  }{Test aborted.}%
  \stop
}{}
⟨/test1⟩
```

```
⟨*test2 | test3⟩
\let\pdfescapehex\@undefined
\let\pdfunescapehex\@undefined
\let\pdfescapename\@undefined
\let\pdfescapestring\@undefined
⟨/test2 | test3⟩
```

3.4.2 Check ε-TEX

```
⟨*test2⟩
\@ifundefined{numexpr}{%
  \PackageError{pdfescape-test2}{%
    Missing \eTeX
  }{Test aborted.}%
  \stop
}{}
⟨/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.

```
\RequirePackage{qstest}
\IncludeTests{*}
\LogTests{log}{*}{*}

\newcommand*{\ExpectVar}[2]{%
  \ifx#1#2%
  \else
    \begingroup
      \@onelevel@sanitize#1%
      \@onelevel@sanitize#2%
      \typeout{[#1] <> [#2]}% hash-ok
    \endgroup
  \fi
  \Expect*{\ifx#1#2true\else false\fi}{true}%
}

\makeatletter
\begingroup
  \gdef\AllBytes{}%
  \count@=0 %
  \catcode0=12 %
  \@whilenum\count@<256 \do{%
    \lccode0=\count@
    \ifnum\count@=32 %
      \xdef\AllBytes{\AllBytes\space}%
    \else
```

```
      \lowercase{%
        \xdef\AllBytes{\AllBytes^^@}%
      }%
    \fi
    \advance\count@ by 1 %
  }%
\endgroup
\newcommand*{\AllBytesHex}{%
  000102030405060708090A0B0C0D0E0F%
  101112131415161718191A1B1C1D1E1F%
  202122232425262728292A2B2C2D2E2F%
  303132333435363738393A3B3C3D3E3F%
  404142434445464748494A4B4C4D4E4F%
  505152535455565758595A5B5C5D5E5F%
  606162636465666768696A6B6C6D6E6F%
  707172737475767778797A7B7C7D7E7F%
  808182838485868788898A8B8C8D8E8F%
  909192939495969798999A9B9C9D9E9F%
  A0A1A2A3A4A5A6A7A8A9AAABACADAEAF%
  B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF%
  C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF%
  D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF%
  E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF%
  F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF%
}
\@onelevel@sanitize\AllBytesHex
\expandafter\lowercase\expandafter{%
  \expandafter\newcommand\expandafter*\expandafter\AllBytesHexLC
      \expandafter{\AllBytesHex}%
}
\newcommand*{\AllBytesName}{}
\begingroup
  \catcode`\#=12 %
  \xdef\AllBytesName{%
    #01#02#03#04#05#06#07#08#09#0A#0B#0C#0D#0E#0F%
    #10#11#12#13#14#15#16#17#18#19#1A#1B#1C#1D#1E#1F%
    #20!"#23$#25&'#28#29*+,-.#2F%
    0123456789:;#3C=#3E?%
    @ABCDEFGHIJKLMNO%
    PQRSTUVWXYZ#5B\@backslashchar#5D^_%
    `abcdefghijklmno%
    pqrstuvwxyz#7B|#7D\string~#7F%
    #80#81#82#83#84#85#86#87#88#89#8A#8B#8C#8D#8E#8F%
    #90#91#92#93#94#95#96#97#98#99#9A#9B#9C#9D#9E#9F%
    #A0#A1#A2#A3#A4#A5#A6#A7#A8#A9#AA#AB#AC#AD#AE#AF%
    #B0#B1#B2#B3#B4#B5#B6#B7#B8#B9#BA#BB#BC#BD#BE#BF%
    #C0#C1#C2#C3#C4#C5#C6#C7#C8#C9#CA#CB#CC#CD#CE#CF%
    #D0#D1#D2#D3#D4#D5#D6#D7#D8#D9#DA#DB#DC#DD#DE#DF%
    #E0#E1#E2#E3#E4#E5#E6#E7#E8#E9#EA#EB#EC#ED#EE#EF%
    #F0#F1#F2#F3#F4#F5#F6#F7#F8#F9#FA#FB#FC#FD#FE#FF%
  }%
\endgroup
\@onelevel@sanitize\AllBytesName

\newcommand*{\AllBytesString}{}
\begingroup
  \def\|{|}%
  \edef\%{\@percentchar}%
  \catcode`\|=0 %
  \catcode`\#=12 %
  \catcode`\~=12 %
  \catcode`\\=12 %
```

```
  |xdef|AllBytesString{%
    \000\001\002\003\004\005\006\007\010\011\012\013\014\015\016\017%
    \020\021\022\023\024\025\026\027\030\031\032\033\034\035\036\037%
    \040!"#$|%&'\(\)*+,-./%
    0123456789:;<=>?%
    @ABCDEFGHIJKLMNO%
    PQRSTUVWXYZ[\\]^_%
    `abcdefghijklmno%
    pqrstuvwxyz{|}~\177%
    \200\201\202\203\204\205\206\207\210\211\212\213\214\215\216\217%
    \220\221\222\223\224\225\226\227\230\231\232\233\234\235\236\237%
    \240\241\242\243\244\245\246\247\250\251\252\253\254\255\256\257%
    \260\261\262\263\264\265\266\267\270\271\272\273\274\275\276\277%
    \300\301\302\303\304\305\306\307\310\311\312\313\314\315\316\317%
    \320\321\322\323\324\325\326\327\330\331\332\333\334\335\336\337%
    \340\341\342\343\344\345\346\347\350\351\352\353\354\355\356\357%
    \360\361\362\363\364\365\366\367\370\371\372\373\374\375\376\377%
  }%
|endgroup
\@onelevel@sanitize\AllBytesString

⟨*test3⟩
\let\org@detokenize\detokenize
\let\detokenize\@undefined
\let\org@numexpr\numexpr
\let\numexpr\@undefined
⟨/test3⟩
\RequirePackage{pdfescape}
⟨*test3⟩
\let\detokenize\org@detokenize
\let\numexpr\org@numexpr
⟨/test3⟩

\begin{qstest}{all-hex}{\AllBytes, escapehex}
  \EdefEscapeHex\x{\AllBytes}%
  \Expect*{\x}*{\AllBytesHex}%
  \ExpectVar\x\AllBytesHex
\end{qstest}

\begin{qstest}{all-unhex}{\AllBytesHex, unescapehex}
  \EdefUnescapeHex\x{\AllBytesHex}%
  \Expect*{\x}*{\AllBytes}%
  \ExpectVar\x\AllBytes
\end{qstest}

\begin{qstest}{all-unhex-lc}{\AllBytesHexLC, unescapehex, lowercase}
  \EdefUnescapeHex\x{\AllBytesHexLC}%
  \Expect*{\x}*{\AllBytes}%
  \ExpectVar\x\AllBytes
\end{qstest}

\begin{qstest}{unhex-incomplete}{unescapehex, incomplete}
  \EdefUnescapeHex\x{4}%
  \Expect*{\x}{@}%
\end{qstest}

\begin{qstest}{unhex-space}{unescapehex, space}
  \EdefUnescapeHex\x{20}%
  \Expect*{\x}{ }%
  \ExpectVar\x\space
\end{qstest}

```

```
\begin{qstest}{unhex-spaces}{unescapehex, spaces}
  \EdefUnescapeHex\x{204020204120}%
  \def\y#1{%
    \edef\z{#1\string @#1#1\string A#1}%
  }\y{ }%
  \Expect*{\x}*{\z}%
  \ExpectVar\x\z
\end{qstest}

\begin{qstest}{unhex-hash}{unescapehex, hash}
  \catcode`\#=12 %
  \EdefUnescapeHex\x{#20}%
  \ExpectVar\x\space
\end{qstest}

\begin{qstest}{unhex-invalid}{unescapehex, invalid}
  \def\test#1#2{%
    \EdefUnescapeHex\x{#1}%
    \edef\y{#2}%
    \@onelevel@sanitize\y
    \ExpectVar\x\y
  }%
⟨*test1⟩
  \edef\x{\pdfunescapehex{4X}}%
  \edef\y{\string @}%
  \ifx\x\y
  \else
    \def~{\space}%
    \typeout{**********************************}%
    \typeout{* Your pdfTeX contains bug 777.~~~~*}%
    \typeout{* This test is redefined as dummy, *}%
    \typeout{* because it triggers the bug.~~~~~*}%
    \typeout{**********************************}%
    \def\test#1#2{}%
  \fi
⟨/test1⟩
  \test{X}{}%
  \test{XY}{}%
  \test{XYZ}{}%
  \test{A}{^^a0}%
  \test{AX}{^^a0}%
  \test{XA}{^^a0}%
  \test{XXAXX}{^^a0}%
\end{qstest}

\begin{qstest}{all-name}{\AllBytes, escapename}
  \EdefEscapeName\x{\AllBytes}%
  \Expect*{\x}*{\AllBytesName}%
  \ExpectVar\x\AllBytesName
\end{qstest}

\begin{qstest}{all-string}{\AllBytes, escapestring}
  \EdefEscapeString\x{\AllBytes}%
  \Expect*{\x}*{\AllBytesString}%
  \ExpectVar\x\AllBytesString
\end{qstest}

\begin{qstest}{uchexdigit}{unescape, uppercase hex digit}
  \catcode`\@=11 %
  \catcode0=12 %
  \def\test#1#2{%
    \uccode0=#1\relax
```

```
    \uppercase{%
      \def\x{^^@}%
    }%
    \Expect*{%
       \ifcase\expandafter\PE@TestUcHexDigit\x
         true%
       \else
         false%
       \fi
    }{#2}%
  }%
  \def\range#1#2#3{%
    \count0=#1\relax
    \loop
    \ifnum\count0<#2\relax
      \test{\count0}{#3}%
      \advance\count0 by 1 %
    \repeat
  }%
  \range{0}{47}{false}%
  \range{48}{57}{true}%
  \range{58}{64}{false}%
  \range{65}{70}{true}%
  \range{71}{255}{false}%
\end{qstest}

\begin{qstest}{unescapename}{unescapename}
  \def\test#1#2{%
    \EdefUnescapeName\x{#1}%
    \edef\y{#2}%
    \@onelevel@sanitize\y
    \ExpectVar\x\y
  }%
  \catcode`\#=12 %
  \catcode0=12 %
  \test{}{}%
  \test{x}{x}%
  \test{xy}{xy}%
  \test{#}{#}%
  \test{##}{##}%
  \test{###}{###}%
  \test{####}{####}%
  \test{#x}{#x}%
  \test{#xy}{#xy}%
  \test{#1}{#1}%
  \test{#40}{@}%
  \test{#400}{@0}%
  \test{#4x0}{#4x0}%
  \test{#ab}{^^ab}%
  \test{#00}{^^@}%
  \test{x#40y#40z}{x@y@z}%
  \test{#40#40#40#40}{@@@@}%
  \test{a#x}{a#x}%
  \test{a#xy}{a#xy}%
  \test{a#1}{a#1}%
  \test{a#40}{a@}%
  \test{a#400}{a@0}%
  \test{#20}{ }%
  \test{a#20}{a }%
  \test{a#20b}{a b}%
  \test{a#20#20#20b}{a \space\space b}%
\end{qstest}
```

```
\begin{qstest}{unescapestring}{unescapestring}
  \def\test#1#2{%
    \EdefUnescapeString\x{#1}%
    \edef\y{#2}%
    \@onelevel@sanitize\y
    \ExpectVar\x\y
  }%
  \catcode0=12 %
  \def\DefChar#1#2{%
    \begingroup
      \uccode0=#2\relax
    \uppercase{\endgroup
      \def#1{^^@}%
    }%
  }%
  \DefChar\nul{0}%
  \DefChar\one{1}%
  \DefChar\bel{8}%
  \DefChar\tab{9}%
  \DefChar\lf{10}%
  \DefChar\ff{12}%
  \DefChar\cr{13}%
  \DefChar\\{92}%
  \test{}{}%
  \test{a}{a}%
  \test{\\}{}%
  \test{\\\\}{\\}%
  \test{\\\\y}{\\y}%
  \test{\\000}{\nul}%
  \test{\\b}{\bel}%
  \test{\\t}{\tab}%
  \test{\\n}{\lf}%
  \test{\\f}{\ff}%
  \test{\\r}{\cr}%
  \test{\\(}{(}%
  \test{\\)}{)}%
  \test{\\040}{ }%
  \test{\\100}{@}%
  \test{\\40}{ }%
  \test{\\1}{\one}%
  \test{\\01}{\one}%
  \test{\\001}{\one}%
  \test{\\18}{\one8}%
  \test{\\018}{\one8}%
  \test{\\0018}{\one8}%
  \test{x\\}{x}%
  \test{x\\\\}{x\\}%
  \test{x\\\\y}{x\\y}%
  \test{x\\000}{x\nul}%
  \test{x\\b}{x\bel}%
  \test{x\\t}{x\tab}%
  \test{x\\n}{x\lf}%
  \test{x\\f}{x\ff}%
  \test{x\\r}{x\cr}%
  \test{x\\(}{x(}%
  \test{x\\)}{x)}%
  \test{x\\040}{x }%
  \test{x\\100}{x@}%
  \test{x\\40}{x }%
  \test{x\\1}{x\one}%
  \test{x\\01}{x\one}%
```

```
  \test{x\\001}{x\one}%
  \test{x\\18}{x\one8}%
  \test{x\\018}{x\one8}%
  \test{x\\0018}{x\one8}%
  \test{\\b\\t\\n\\f\\r\\(\\)\\\\\\000\\040}{%
    \bel\tab\lf\ff\cr()\\\nul\space
  }%
  \test{\\\lf}{}%
  \test{x\\\lf}{x}%
  \test{\cr}{\lf}%
  \test{\cr\lf}{\lf}%
  \test{\lf}{\lf}%
  \test{\lf\cr}{\lf\lf}%
  \test{x\cr}{x\lf}%
  \test{x\cr\lf}{x\lf}%
  \test{x\lf}{x\lf}%
  \test{x\lf\cr}{x\lf\lf}%
  \test{x\\\cr\lf y\cr}{xy\lf}%
\end{qstest}
\stop
```

⟨/test1 | test2 | test3⟩

4 Installation

4.1 Download

Package. This package is available on CTAN[1]:

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
```

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
```

[1] ftp://ftp.ctan.org/tex-archive/

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

```
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 (teTEX, mikTEX, ...) relies on file name databases, you must refresh these. For example, teTEX users run `texhash` or `mktexlsr`.

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 pdfLATEX:

```
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.

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, 128
\@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

\␣ ... 112, 628

A

\advance ... 447, 483, 484, 716, 914
\AllBytes 704, 710, 713, 807, 808, 815, 816, 821, 822, 881, 882, 887, 888
\AllBytesHex ... 719, 737, 740, 809, 810, 813, 814
\AllBytesHexLC ... 739, 819, 820
\AllBytesName . 742, 745, 764, 883, 884
\AllBytesString ... 766, 793, 889, 890

B

\begin . 807, 813, 819, 825, 830, 836, 845, 851, 881, 887, 893, 924, 961
\bel 978, 990, 1010, 1027

C

\catcode 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
\count 910, 912, 913, 914
\count@ 705, 707, 708, 709, 716
\cr 982, 994, 1014, 1027, 1031, 1032, 1034, 1035, 1036, 1038, 1039
\csname 2, 32, 42, 63, 76, 80, 83, 90, 101, 362, 385, 414

D

\DefChar 969, 976, 977, 978, 979, 980, 981, 982, 983
\detokenize 108, 796, 797, 803
\dimen . 445, 446, 447, 450, 451, 480, 481, 482, 483, 484, 488, 489, 490
\dimexpr 438, 440, 465, 468, 476
\do 707

E

\EdefEscapeHex *2*, 363, 390, 808
\EdefEscapeName 372, 396, 882
\EdefEscapeString 377, 399, 888
\EdefUnescapeHex 368, 393, 814, 820, 826, 831, 837, 847, 853
\EdefUnescapeName *2*, 129, 926
\EdefUnescapeString *2*, 229, 963
\empty 36
\end . . . 811, 817, 823, 828, 834, 843, 849, 879, 885, 891, 922, 959, 1040
\endcsname 2, 32, 42, 63, 76, 80, 83, 90, 101, 362, 385, 414
\endinput 51, 403
\escapechar 407
\eTeX 681
\Expect 699, 809, 815, 821, 827, 832, 841, 883, 889, 901
\ExpectVar 690, 810, 816, 822, 833, 842, 848, 856, 884, 890, 929, 966

F

\ff 981, 993, 1013, 1027

G

\gdef 138, 146, 704

I

\ifcase 33, 160, 161, 339, 347, 457, 510, 523, 524, 563, 902
\ifnum 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
\ifPE@etex 412, 418, 436, 462, 548, 595, 605, 637
\ifx 34, 36, 42, 63, 71, 80, 90, 120, 147, 151, 246, 251, 362, 414, 430, 503, 504, 560, 619, 691, 699, 861
\immediate 44, 65
\IncludeTests 687

L

\lccode 708
\lf 980, 992, 1012, 1027, 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036, 1037, 1038, 1039
\LogTests 688
\loop 911
\lowercase 712, 738

M

\makeatletter 651, 702
\meaning 92

N

\NeedsTeXFormat 650
\newcommand . . . 690, 719, 739, 742, 766
\newif 412
\nul 976, 989, 1009, 1027
\number 488, 489, 490
\numexpr 438, 440, 465, 468, 475, 476, 798, 799, 804

O

\one 977, 1000, 1001, 1002, 1003, 1004, 1005, 1020, 1021, 1022, 1023, 1024, 1025
\org@detokenize 796, 803
\org@numexpr 798, 804

P

\PackageError 666, 680
\PackageInfo 47
\pdfescape 654, 658, 662, 667
\pdfescapehex 391, 673
\pdfescapename 397, 675
\pdfescapestring 400, 676
\pdfunescapehex 394, 674, 859
\PE@@NormalizeLineEnd 242, 245
\PE@@OctChar 464, 472
\PE@AtEnd 7, 8, 402, 647
\PE@backslash 406, 473, 487
\PE@CheckEndBackslash 319
\PE@DeHex 498, 502
\PE@DeName 142, 146, 175, 179
\PE@DeString 273, 359
\PE@edefbabel . 383, 391, 394, 397, 400
\PE@EnsureCode 6, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25
\PE@EscapeHex 366, 418
\PE@EscapeName 375, 548
\PE@EscapeNameAdd 566, 571, 585, 593, 595
\PE@EscapeNameHashChar 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 592

\PE@EscapeNameTokens .. 550, 555, 559
\PE@EscapeString 380, 605
\PE@EscapeStringAdd .....
..... 626, 627, 628, 630, 637
\PE@EscapeStringTokens 607, 613, 618
\PE@etextrue 416
\PE@hash 405, 566, 571, 593
\PE@HexChar 432, 436, 567, 572
\PE@HexDigit .. 438, 439, 450, 451, 455
\PE@InitUccodeHexDigit 140, 186, 496
\PE@next 149, 154, 175,
179, 183, 248, 252, 254, 257,
340, 342, 344, 348, 350, 352,
505, 511, 515, 534, 537, 543, 546
\PE@NormalizeLineEnd 232, 240
\PE@OctAll 342, 348, 350, 354
\PE@OctChar 462, 622, 625
\PE@OctI 338
\PE@OctII 340, 346
\PE@onelevel@sanitize .....
.... 85, 91, 96, 103, 107, 133, 234
\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
\PE@sanitize .....
.. 79, 130, 230, 364, 369, 373, 378
\PE@SanitizeSpaceOther .....
..... 115, 131, 231, 365, 374, 379
\PE@space@other 111, 122
\PE@space@space 114, 527
\PE@SpaceToOther 116, 118
\PE@strip@prefix 92, 94
\PE@temp 171, 174, 527, 530, 533
\PE@testA 157,
160, 169, 508, 510, 520, 523, 525
\PE@testB . 158, 161, 169, 521, 524, 525
\PE@TestOctDigit 327, 339, 347
\PE@TestUcHexDigit .....
. 160, 161, 210, 510, 523, 524, 902
\PE@ToHex 420, 425, 429
\PE@UnescapeHex 370, 494
\PE@UnescapeName 132, 135
\PE@UnescapeString 233, 266
\ProvidesFile 653, 657, 661
\ProvidesPackage 77

R

\range 909, 917, 918, 919, 920, 921
\repeat 915
\RequirePackage 686, 801

S

\space ... 710, 833, 848, 863, 958, 1027
\stop 669, 683, 1041

T

\tab 979, 991, 1011, 1027
\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
\the 3, 9, 465, 468, 475, 476
\typeout .. 696, 864, 865, 866, 867, 868

U

\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
\uppercase 156, 170,
260, 356, 507, 519, 529, 898, 972

W

\write 44, 65

X

\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

Y

\y 838, 840, 854, 855, 856, 860,
861, 927, 928, 929, 964, 965, 966

Z

\z 839, 841, 842
\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