

# The `pdftexcmds` package

Heiko Oberdiek

<oberdiek@uni-freiburg.de>

2009/09/22 v0.5

## Abstract

LUAT<sub>E</sub>X provides most of the commands of pdfT<sub>E</sub>X 1.40. However a number of utility functions are removed. This package tries to fill the gap and implements some of the missing primitive using Lua.

## Contents

|   |           |
|---|-----------|
| <b>1 Documentation</b>  | <b>2</b>  |
| 1.1 General principles . . . . .  | 2         |
| 1.2 Macros . . . . .  | 3         |
| 1.2.1 Experimental . . . . .  | 4         |
| <b>2 Implementation</b>   | <b>4</b>  |
| 2.1 Reload check and package identification . . . . .                         | 4         |
| 2.2 Catcodes . . . . .  | 5         |
| 2.3 Load package infwarerr . . . . .  | 6         |
| 2.4 Without LUAT <sub>E</sub> X . . . . .                                     | 6         |
| 2.5 \pdf@primitive, \pdf@ifprimitive . . . . .                                | 7         |
| 2.5.1 Using LUAT <sub>E</sub> X's <code>tex.enableprimitives</code> . . . . . | 7         |
| 2.5.2 Trying various names to find the primitives . . . . .                   | 8         |
| 2.5.3 Result . . . . .  | 9         |
| 2.6 XeT <sub>E</sub> X . . . . .  | 9         |
| 2.7 Load Lua module . . . . .   | 9         |
| 2.8 Lua functions . . . . .   | 10        |
| 2.9 Lua module . . . . .  | 12        |
| <b>3 Test</b>   | <b>17</b> |
| 3.1 Catcode checks for loading . . . . .                                      | 17        |
| <b>4 Installation</b>   | <b>18</b> |
| 4.1 Download . . . . .  | 18        |
| 4.2 Bundle installation . . . . .   | 19        |
| 4.3 Package installation . . . . .  | 19        |
| 4.4 Refresh file name databases . . . . .                                     | 19        |
| 4.5 Some details for the interested . . . . .                                 | 19        |
| <b>5 History</b>  | <b>20</b> |
| [2007/11/11 v0.1] . . . . .   | 20        |
| [2007/11/12 v0.2] . . . . .   | 20        |
| [2007/12/12 v0.3] . . . . .   | 20        |
| [2009/04/10 v0.4] . . . . .   | 20        |
| [2009/09/22 v0.5] . . . . .   | 20        |
| <b>6 Index</b>  | <b>20</b> |

# 1 Documentation

Some primitives of pdfTeX are not defined by LUATeX. This package implements macro based solutions using Lua code for the following missing pdfTeX primitives;

- `\pdfstrcmp`
- `\pdfunescapehex`
- `\pdfescapehex`
- `\pdfescapename`
- `\pdfescapestring`
- `\pdffilesize`
- `\pdffilemoddate`
- `\pdffiledump`
- `\pdfmdfivesum`
- `\immediate\write18`

The original names of the primitives cannot be used:

- The syntax for their arguments cannot easily simulated by macros. The primitives using key words such as `file` (`\pdfmdfivesum`) or `offset` and `length` (`\pdffiledump`) and uses `<general text>` for the other arguments. Using token registers assignments, `<general text>` could be catched. However, the simulated primitives are expandable and register assignments would destroy this important property. (`<general text>` allows something like `\expandafter\bgroup ... \egroup`.)
- The original primitives can be expanded using one expansion step. The new macros need two expansion steps because of the additional macro expansion.

Example:

```
\expandafter\foo\pdffilemoddate{file}
vs.
\expandafter\expandafter\expandafter
\foo\pdf@filemoddate{file}
```

LUATeX isn't stable yet and thus the status of this package is *experimental*. Feedback is welcome.

## 1.1 General principles

**Naming convention:** Usually this package defines a macro `\pdf@{cmd}` if pdfTeX provides `\pdf{cmd}`.

**Arguments:** The order of arguments in `\pdf@{cmd}` is the same as for the corresponding primitive of pdfTeX. The arguments are ordinary undelimited TeX arguments, no `<general text>` and without additional keywords.

**Expandability:** The macro `\pdf@{cmd}` is expandable if the corresponding pdfTeX primitive has this property. Exact two expansion steps are necessary (first is the macro expansion) except for `\pdf@primitive` and `\pdf@ifprimitive`. The latter ones are not macros, but have the direct meaning of the primitive.

**Without Luatex:** The macros `\pdf@{cmd}` are mapped to the commands of pdfTeX if they are available. Otherwise they are undefined.

**Availability:** The macros that the packages provides are undefined, if the necessary primitives are not found and cannot be implemented by Lua.

## 1.2 Macros

```
\pdfstrcmp {\⟨stringA⟩} {\⟨stringB⟩}
```

Same as `\pdfstrcmp{\⟨stringA⟩}{\⟨stringB⟩}`.

```
\pdfunescapehex {\⟨string⟩}
```

Same as `\pdfunescapehex{\⟨string⟩}`. The argument is a byte string given in hexadecimal notation. The result are character tokens from 0 until 255 with catcode 12 and the space with catcode 10.

```
\pdfescapehex {\⟨string⟩}  
\pdfescapestring {\⟨string⟩}  
\pdfescapename {\⟨string⟩}
```

Same as the primitives of pdf<sub>T</sub>E<sub>X</sub>. However pdf<sub>T</sub>E<sub>X</sub> does not know about characters with codes 256 and larger. Thus the string is treated as byte string, characters with more than eight bits are ignored.

```
\pdffilesize {\⟨filename⟩}
```

Same as `\pdffilesize{\⟨filename⟩}`.

```
\pdffilemoddate {\⟨filename⟩}
```

Same as `\pdffilemoddate{\⟨filename⟩}`.

```
\pdffiledump {\⟨offset⟩} {\⟨length⟩} {\⟨filename⟩}
```

Same as `\pdffiledump offset ⟨offset⟩ length ⟨length⟩ {\⟨filename⟩}`. Both ⟨offset⟩ and ⟨length⟩ must not be empty, but must be a valid T<sub>E</sub><sub>X</sub> number.

```
\pdfmdfivesum {\⟨string⟩}
```

Same as `\pdfmdfivesum{\⟨string⟩}`. Keyword `file` is supported by macro `\pdffilemdfivesum`.

```
\pdffilemdfivesum {\⟨filename⟩}
```

Same as `\pdfmdfivesum file{\⟨filename⟩}`.

```
\pdfshellescape
```

Same as `\pdfshellescape`. It expands to 1 if external commands can be executed and 0 otherwise. In pdf<sub>T</sub>E<sub>X</sub> external commands must be enabled first by command line option or configuration option. In LUAT<sub>E</sub>X option `--safer` disables the execution of external commands.

```
\pdfsystem {\⟨cmdline⟩}
```

It is a wrapper for `\immediate\write18` in pdf<sub>T</sub>E<sub>X</sub> or `os.execute` in LUAT<sub>E</sub>X.

In theory `os.execute` returns a status number. But its meaning is quite undefined. Are there some reliable properties? Does it make sense to provide an user interface to this status exit code?

```
\pdf@primitive \cmd
```

Same as `\pdfprimitive` in pdfTeX or LUATEX. In XeTeX the primitive is called `\primitive`. Despite the current definition of the command `\cmd`, it's meaning as primitive is used.

```
\pdf@ifprimitive \cmd
```

Same as `\ifpdfprimitive` in pdfTeX or LUATEX. XeTeX calls it `\ifprimitive`. It is a switch that checks if the command `\cmd` has it's primitive meaning.

### 1.2.1 Experimental

```
\pdf@unescapehexnative {\langle string\rangle}
\pdf@escapehexnative {\langle string\rangle}
\pdf@escapenamenative {\langle string\rangle}
\pdf@mdfivesumnative {\langle string\rangle}
```

The variants without `native` in the macro name are supposed to be compatible with pdfTeX. However characters with more than eight bits are not supported and are ignored. If LUATEX is running, then its UTF-8 coded strings are used. Thus the full unicode character range is supported. However the result differs from pdfTeX for characters with eight or more bits.

```
\pdf@pipe {\langle cmdline\rangle}
```

It calls `\langle cmdline\rangle` and returns the output of the external program in the usual manner as byte string (catcode 12, space with catcode 10). The Lua documentation says, that the used `io.popen` may not be available on all platforms. Then macro `\pdf@pipe` is undefined.

## 2 Implementation

1 `(*package)`

### 2.1 Reload check and package identification

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

```
2 \begingroup
3   \catcode44 12 % ,
4   \catcode45 12 % -
5   \catcode46 12 % .
6   \catcode58 12 % :
7   \catcode64 11 % @
8   \catcode123 1 % {
9   \catcode125 2 % }
10  \expandafter\let\expandafter\x\csname ver@pdftexcmds.sty\endcsname
11  \ifx\x\relax % plain-TeX, first loading
12  \else
13    \def\empty{}%
14    \ifx\x\empty % LaTeX, first loading,
15      % variable is initialized, but \ProvidesPackage not yet seen
16    \else
```

```

17      \catcode35 6 % #
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{pdftexcmds}{The package is already loaded}%
26      \aftergroup\endinput
27  \fi
28 \fi
29 \endgroup

```

Package identification:

```

30 \begingroup
31   \catcode35 6 % #
32   \catcode40 12 % (
33   \catcode41 12 % )
34   \catcode44 12 % ,
35   \catcode45 12 % -
36   \catcode46 12 % .
37   \catcode47 12 % /
38   \catcode58 12 % :
39   \catcode64 11 % @
40   \catcode91 12 % [
41   \catcode93 12 % ]
42 \catcode123 1 % {
43 \catcode125 2 % }
44 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
45     \def\x#1#2#3[#4]{\endgroup
46         \immediate\write-1{Package: #3 #4}%
47         \xdef#1[#4]%
48     }%
49 \else
50     \def\x#1#2[#3]{\endgroup
51         #2[#3]%
52         \ifx#1\undefined
53             \xdef#1[#3]%
54         \fi
55         \ifx#1\relax
56             \xdef#1[#3]%
57         \fi
58     }%
59 \fi
60 \expandafter\x\csname ver@pdftexcmds.sty\endcsname
61 \ProvidesPackage{pdftexcmds}%
62 [2009/09/22 v0.5 LuaTeX support for pdfTeX utility functions (HO)]

```

## 2.2 Catcodes

```

63 \begingroup
64   \catcode123 1 % {
65   \catcode125 2 % }
66   \def\x{\endgroup
67       \expandafter\edef\csname pdftexcmds@AtEnd\endcsname{%
68           \catcode35 \the\catcode35\relax
69           \catcode64 \the\catcode64\relax
70           \catcode123 \the\catcode123\relax
71           \catcode125 \the\catcode125\relax
72       }%
73   }%
74 \x

```

```

75 \catcode35 6 % #
76 \catcode64 11 % @
77 \catcode123 1 % {
78 \catcode125 2 % }
79 \def\TMP@EnsureCode#1#2{%
80   \edef\pdftexcmds@AtEnd{%
81     \pdftexcmds@AtEnd
82     \catcode#1 \the\catcode#1\relax
83   }%
84   \catcode#1 #2\relax
85 }
86 \TMP@EnsureCode{10}{12}%
87 \TMP@EnsureCode{33}{12}%
88 \TMP@EnsureCode{34}{12}%
89 \TMP@EnsureCode{39}{12}%
90 \TMP@EnsureCode{40}{12}%
91 \TMP@EnsureCode{41}{12}%
92 \TMP@EnsureCode{42}{12}%
93 \TMP@EnsureCode{43}{12}%
94 \TMP@EnsureCode{44}{12}%
95 \TMP@EnsureCode{45}{12}%
96 \TMP@EnsureCode{46}{12}%
97 \TMP@EnsureCode{47}{12}%
98 \TMP@EnsureCode{58}{12}%
99 \TMP@EnsureCode{60}{12}%
100 \TMP@EnsureCode{61}{12}%
101 \TMP@EnsureCode{62}{12}%
102 \TMP@EnsureCode{94}{7}%
103 \TMP@EnsureCode{95}{12}%
104 \TMP@EnsureCode{96}{12}%
105 \TMP@EnsureCode{126}{12}%
106 \edef\pdftexcmds@AtEnd{%
107   \pdftexcmds@AtEnd
108   \escapechar=\number\escapechar\relax
109 }
110 \escapechar=92 %

```

### 2.3 Load package **infwarerr**

```

111 \begingroup\expandafter\expandafter\expandafter\endgroup
112 \expandafter\ifx\csname RequirePackage\endcsname\relax
113   \input infwarerr.sty\relax
114   \input ifluatex.sty\relax
115 \else
116   \RequirePackage{infwarerr}[2007/09/09]%
117   \RequirePackage{ifluatex}[2009/04/10]%
118 \fi

```

### 2.4 Without LuaTeX

```

119 \ifluatex
120 \else
121   \@PackageInfoNoLine{\pdftexcmds}{LuaTeX not detected}%
122   \def\pdftexcmds@nopdftex{%
123     \@PackageInfoNoLine{\pdftexcmds}{pdfTeX >= 1.30 not detected}%
124     \let\pdftexcmds@nopdftex\relax
125   }%
126   \def\pdftexcmds@temp#1{%
127     \begingroup\expandafter\expandafter\expandafter\endgroup
128     \expandafter\ifx\csname pdf#1\endcsname\relax
129       \pdftexcmds@nopdftex
130     \else
131       \expandafter\def\csname pdf@#1\expandafter\endcsname
132       \expandafter##\expandafter{%

```

```

133      \csname pdf#1\endcsname
134    }%
135  \fi
136 }%
137 \pdftexcmds@temp{strcmp}%
138 \pdftexcmds@temp{escapehex}%
139 \let\pdf@escapehexnative\pdf@escapehex
140 \pdftexcmds@temp{unescapehex}%
141 \let\pdf@unescapehexnative\pdf@unescapehex
142 \pdftexcmds@temp{escapestring}%
143 \pdftexcmds@temp{escapename}%
144 \pdftexcmds@temp{filesize}%
145 \pdftexcmds@temp{filemoddate}%
146 \begingroup\expandafter\expandafter\expandafter\endgroup
147 \expandafter\ifx\csname pdfshellescape\endcsname\relax
148   \pdftexcmds@nopdftex
149 \else
150   \def\pdf@shellescape{%
151     \pdfshellescape
152   }%
153 \fi
154 \begingroup\expandafter\expandafter\expandafter\endgroup
155 \expandafter\ifx\csname pdffiledump\endcsname\relax
156   \pdftexcmds@nopdftex
157 \else
158   \def\pdf@filedump#1#2#3{%
159     \pdffiledump offset#1 length#2{#3}%
160   }%
161 \fi
162 \begingroup\expandafter\expandafter\expandafter\endgroup
163 \expandafter\ifx\csname pdfmdfivesum\endcsname\relax
164   \pdftexcmds@nopdftex
165 \else
166   \def\pdf@mdfivesum#\pdfmdfivesum}%
167   \let\pdf@mdfivesumnative\pdf@mdfivesum
168   \def\pdf@filemdfivesum#\pdfmdfivesum file}%
169 \fi
170 \def\pdf@system#{%
171   \immediate\write18%
172 }%
173 \fi

```

## 2.5 \pdf@primitive, \pdf@ifprimitive

Since version 1.40.0 pdfTeX has \pdfprimitive and \ifpdfprimitive. And \pdfprimitive was fixed in version 1.40.4.

XeTeX provides them under the name \primitive and \ifprimitive. LUATEX knows both name variants, but they have possibly to be enabled first (`tex.enableprimitives`).

Depending on the format TeX Live uses a prefix `luatex`.

Caution: \let must be used for the definition of the macros, especially because of \ifpdfprimitive.

### 2.5.1 Using LuATEX's `tex.enableprimitives`

```

174 \ifluatex
\pdftexcmds@directlua
175 \ifnum\luatexversion<36 %
176   \def\pdftexcmds@directlua{\directlua0 }%
177 \else
178   \let\pdftexcmds@directlua\directlua
179 \fi

```

```

180 \begingroup
181   \newlinechar=10 %
182   \endlinechar=\newlinechar
183   \pdftexcmds@directlua{%
184     if tex.enableprimitives then
185       tex.enableprimitives('pdf@', {'primitive', 'ifprimitive'})
186     end
187   }%
188 \endgroup %

189 \fi

```

### 2.5.2 Trying various names to find the primitives

\pdftexcmds@strip@prefix

```

190 \def\pdftexcmds@strip@prefix#1>{}

191 \def\pdftexcmds@temp#1#2#3{%
192   \begingroup\expandafter\expandafter\expandafter\endgroup
193   \expandafter\ifx\csname pdf@#1\endcsname\relax
194     \begingroup
195       \def\x{\#3}%
196       \edef\x{\expandafter\pdftexcmds@strip@prefix\meaning\x}%
197       \escapechar=-1 %
198       \edef\y{\expandafter\meaning\csname#2\endcsname}%
199     \expandafter\endgroup
200     \ifx\x\y
201       \expandafter\let\csname pdf@#1\expandafter\endcsname
202         \csname #2\endcsname
203     \fi
204   \fi
205 }

```

\pdf@primitive

```

206 \pdftexcmds@temp{primitive}{pdfprimitive}{pdfprimitive}%
207 \pdftexcmds@temp{primitive}{primitive}{primitive}%
208 \pdftexcmds@temp{primitive}{luatexprimitive}{pdfprimitive}%
209 \pdftexcmds@temp{primitive}{luatexpdfprimitive}{pdfprimitive}%

```

\pdf@ifprimitive

```

210 \pdftexcmds@temp{ifprimitive}{ifpdfprimitive}{ifpdfprimitive}%
211 \pdftexcmds@temp{ifprimitive}{ifprimitive}{ifprimitive}%
212 \pdftexcmds@temp{ifprimitive}{luatexifprimitive}{ifpdfprimitive}%
213 \pdftexcmds@temp{ifprimitive}{luatexifpdfprimitive}{ifpdfprimitive}%

```

Disable broken \pdfprimitive.

```

214 \begingroup
215   \expandafter\ifx\csname pdf@primitive\endcsname\relax
216   \else
217     \expandafter\ifx\csname pdftexversion\endcsname\relax
218     \else
219       \ifnum\pdftexversion=140 %
220         \expandafter\ifx\csname pdftexrevision\endcsname\relax
221         \else
222           \ifnum\pdftexrevision<4 %
223             \endgroup
224             \let\pdf@primitive\@undefined
225             \PackageInfoNoLine{\pdftexcmds}{%
226               \string\pdf@primitive disabled, because\MessageBreak
227               \string\pdfprimitive\space is broken until pdfTeX 1.40.4%
228             }%
229           \begingroup

```

```

230           \fi
231           \fi
232           \fi
233           \fi
234   \fi
235 \endgroup

```

### 2.5.3 Result

```

236 \begingroup
237  \@PackageInfoNoLine{pdftexcmds}{%
238    \string\pdf@primitive\space is %
239    \expandafter\ifx\csname pdf@primitive\endcsname\relax not \fi
240    available%
241  }%
242  \@PackageInfoNoLine{pdftexcmds}{%
243    \string\pdf@ifprimitive\space is %
244    \expandafter\ifx\csname pdf@ifprimitive\endcsname\relax not \fi
245    available%
246  }%
247 \endgroup

```

## 2.6 XeTeX

Look for primitives `\shellescape`, `\strcmp`.

```

248 \def\pdftexcmds@temp#1{%
249   \begingroup\expandafter\expandafter\expandafter\endgroup
250   \expandafter\ifx\csname pdf@#1\endcsname\relax
251     \begingroup
252       \escapechar=-1 %
253       \edef\x{\expandafter\meaning\csname#1\endcsname}%
254       \def\y{#1}%
255       \def\z##1->{}{%
256         \edef\y{\expandafter\z\meaning\y}%
257       \expandafter\endgroup
258       \ifx\x\y
259         \expandafter\def\csname pdf@#1\expandafter\endcsname
260         \expandafter{%
261           \csname#1\endcsname
262         }%
263       \fi
264     }%
265   }%
266 \pdftexcmds@temp{shellescape}%
267 \pdftexcmds@temp{strcmp}%

268 \ifluatex
269 \else
270   \pdftexcmds@AtEnd
271   \expandafter\endinput
272 \fi

```

## 2.7 Load Lua module

```

273 \begingroup\expandafter\expandafter\expandafter\endgroup
274 \expandafter\ifx\csname RequirePackage\endcsname\relax
275   \input luatex-loader.sty\relax
276 \else
277   \RequirePackage{luatex-loader}[2009/04/10]%
278 \fi
279 \pdftexcmds@directluaf{%
280   require("oberdiek.pdftexcmds")%
281 }

```

## 2.8 Lua functions

```
\pdftexcmds@toks
282 \begingroup\expandafter\expandafter\expandafter\endgroup
283 \expandafter\ifx\csname newtoks\endcsname\relax
284   \toksdef\pdftexcmds@toks=0 %
285 \else
286   \csname newtoks\endcsname\pdftexcmds@toks
287 \fi

288 \ifnum\luatexversion<36 %
289 \else
290   \catcode`\0=9 %
291 \fi

\pdf@strcmp
292 \long\def\pdf@strcmp#1#2{%
293   \directlua0{%
294     oberdiek.pdftexcmds.strptime("\luaescapestring{#1}",%
295       "\luaescapestring{#2}")%
296   }%
297 }%

\pdf@escapehex
298 \long\def\pdf@escapehex#1{%
299   \directlua0{%
300     oberdiek.pdftexcmds.escapehex("\luaescapestring{#1}", "byte")%
301   }%
302 }%

\pdf@escapehexnative
303 \long\def\pdf@escapehexnative#1{%
304   \directlua0{%
305     oberdiek.pdftexcmds.escapehex("\luaescapestring{#1}")%
306   }%
307 }%

\pdf@unescapehex
308 \def\pdf@unescapehex#1{%
309   \the\expandafter\pdftexcmds@toks
310   \directlua0{%
311     oberdiek.pdftexcmds.toks="pdftexcmds@toks"%
312     oberdiek.pdftexcmds.unescapehex("\luaescapestring{#1}", "byte")%
313   }%
314 }%

\pdf@unescapehexnative
315 \def\pdf@unescapehexnative#1{%
316   \the\expandafter\pdftexcmds@toks
317   \directlua0{%
318     oberdiek.pdftexcmds.toks="pdftexcmds@toks"%
319     oberdiek.pdftexcmds.unescapehex("\luaescapestring{#1}")%
320   }%
321 }%

\pdf@escapestring
322 \long\def\pdf@escapestring#1{%
323   \directlua0{%
324     oberdiek.pdftexcmds.escapestring("\luaescapestring{#1}", "byte")%
325   }%
326 }
```

```

\pdf@escapename
327 \long\def\pdf@escapename#1{%
328   \directlua{%
329     oberdiek.pdftexcmds.escapename("\luaescapestring{\#1}", "byte")%
330   }%
331 }

\pdf@escapenamenative
332 \long\def\pdf@escapenamenative#1{%
333   \directlua{%
334     oberdiek.pdftexcmds.escapename("\luaescapestring{\#1}")%
335   }%
336 }

\pdf@filesize
337 \def\pdf@filesize#1{%
338   \directlua{%
339     oberdiek.pdftexcmds.filesize("\luaescapestring{\#1}")%
340   }%
341 }

\pdf@filemoddate
342 \def\pdf@filemoddate#1{%
343   \directlua{%
344     oberdiek.pdftexcmds.filemoddate("\luaescapestring{\#1}")%
345   }%
346 }

\pdf@filedump
347 \def\pdf@filedump#1#2#3{%
348   \directlua{%
349     oberdiek.pdftexcmds.filedump("\luaescapestring{\number#1}",%
350       "\luaescapestring{\number#2}",%
351       "\luaescapestring{\#3}")%
352   }%
353 }

\pdf@mdfivesum
354 \long\def\pdf@mdfivesum#1{%
355   \directlua{%
356     oberdiek.pdftexcmds.mdfivesum("\luaescapestring{\#1}", "byte")%
357   }%
358 }

\pdf@mdfivesumnative
359 \long\def\pdf@mdfivesumnative#1{%
360   \directlua{%
361     oberdiek.pdftexcmds.mdfivesum("\luaescapestring{\#1}")%
362   }%
363 }

\pdf@filemdfivesum
364 \def\pdf@filemdfivesum#1{%
365   \directlua{%
366     oberdiek.pdftexcmds.filemdfivesum("\luaescapestring{\#1}")%
367   }%
368 }

\pdf@shellescape
369 \def\pdf@shellescape{%

```

```

370  \directlua{%
371      oberdiek.pdftexcmds.shellescape()%%
372  }%
373 }

\pdf@system

374 \def\pdf@system#1{%
375     \directlua{%
376         oberdiek.pdftexcmds.system("\luaescapestring{#1}")%
377     }%
378 }

\pdf@lastsystemstatus

379 \def\pdf@lastsystemstatus{%
380     \directlua{%
381         oberdiek.pdftexcmds.lastsystemstatus()%%
382     }%
383 }

\pdf@lastsystemexit

384 \def\pdf@lastsystemexit{%
385     \directlua{%
386         oberdiek.pdftexcmds.lastsystemexit()%%
387     }%
388 }

389 \catcode`\0=12 %

\pdf@pipe Check availability of io.popen first.

390 \ifnum0%
391     \pdftexcmds@directlua{%
392         if io.popen then %
393             tex.write("1")%
394         end%
395     }%
396     =1 %
397     \def\pdf@pipe#1{%
398         \the\expandafter\pdftexcmds@toks
399         \pdftexcmds@directlua{%
400             oberdiek.pdftexcmds.toks="pdftexcmds@toks"%
401             oberdiek.pdftexcmds.pipe("\luaescapestring{#1}")%
402         }%
403     }%
404 \fi

405 \pdftexcmds@AtEnd
406 
```

## 2.9 Lua module

```

407 <*lua>
408 module("oberdiek.pdftexcmds", package.seeall)
409 local systemexitstatus
410 function strcmp(A, B)
411     if A == B then
412         tex.write("0")
413     elseif A < B then
414         tex.write("-1")
415     else
416         tex.write("1")
417     end

```

```

418 end
419 local function utf8_to_byte(str)
420   local i = 0
421   local n = string.len(str)
422   local t = {}
423   while i < n do
424     i = i + 1
425     local a = string.byte(str, i)
426     if a < 128 then
427       table.insert(t, string.char(a))
428     else
429       if a >= 192 and i < n then
430         i = i + 1
431         local b = string.byte(str, i)
432         if b < 128 or b >= 192 then
433           i = i - 1
434           elseif a == 194 then
435             table.insert(t, string.char(b))
436           elseif a == 195 then
437             table.insert(t, string.char(b + 64))
438           end
439         end
440       end
441     end
442     return table.concat(t)
443 end
444 function escapehex(str, mode)
445   if mode == "byte" then
446     str = utf8_to_byte(str)
447   end
448   tex.write((string.gsub(str, ".",
449     function (ch)
450       return string.format("%02X", string.byte(ch))
451     end
452   )))
453 end

```

See procedure unescapehex in file `utils.c` of pdfTeX. Caution: `tex.write` ignores leading spaces.

```

454 function unescapehex(str, mode)
455   local a = 0
456   local first = true
457   local result = {}
458   for i = 1, string.len(str), 1 do
459     local ch = string.byte(str, i)
460     if ch >= 48 and ch <= 57 then
461       ch = ch - 48
462     elseif ch >= 65 and ch <= 70 then
463       ch = ch - 55
464     elseif ch >= 97 and ch <= 102 then
465       ch = ch - 87
466     else
467       ch = nil
468     end
469     if ch then
470       if first then
471         a = ch * 16
472         first = false
473       else
474         table.insert(result, a + ch)
475         first = true
476       end
477     end

```

```

478     end
479     if not first then
480       table.insert(result, a)
481     end
482     if mode == "byte" then
483       local utf8 = {}
484       for i, a in ipairs(result) do
485         if a < 128 then
486           table.insert(utf8, a)
487         else
488           if a < 192 then
489             table.insert(utf8, 194)
490             a = a - 128
491           else
492             table.insert(utf8, 195)
493             a = a - 192
494           end
495           table.insert(utf8, a + 128)
496         end
497       end
498     result = utf8
499   end
500   tex.settoks(toks, string.char(unpack(result)))
501 end

```

See procedure `escapestring` in file `utils.c` of pdfTeX.

```

502 function escapestring(str, mode)
503   if mode == "byte" then
504     str = utf8_to_byte(str)
505   end
506   tex.write((string.gsub(str, ".", ,
507     function (ch)
508       local b = string.byte(ch)
509       if b < 33 or b > 126 then
510         return string.format("\\%o", b)
511       end
512       if b == 40 or b == 41 or b == 92 then
513         return "\\ " .. ch
514       end

```

Lua 5.1 returns the match in case of return value `nil`.

```

515     return nil
516   end
517   )))
518 end

```

See procedure `escapename` in file `utils.c` of pdfTeX.

```

519 function escapename(str, mode)
520   if mode == "byte" then
521     str = utf8_to_byte(str)
522   end
523   tex.write((string.gsub(str, ".", ,
524     function (ch)
525       local b = string.byte(ch)
526       if b == 0 then

```

In Lua 5.0 `nil` could be used for the empty string, But `nil` returns the match in Lua 5.1, thus we use the empty string explicitly.

```

527       return ""
528     end
529     if b <= 32 or b >= 127
530       or b == 35 or b == 37 or b == 40 or b == 41
531       or b == 47 or b == 60 or b == 62 or b == 91
532       or b == 93 or b == 123 or b == 125 then
533       return string.format("#%.2X", b)

```

```

534     else
Lua 5.1 returns the match in case of return value nil.
535         return nil
536     end
537     end
538   )))
539 end
540 function filesize(filename)
541   local foundfile = kpse.find_file(filename, "tex", true)
542   if foundfile then
543     local size = lfs.attributes(foundfile, "size")
544     if size then
545       tex.write(size)
546     end
547   end
548 end
See procedure makepdftime in file utils.c of pdfTeX.
549 function filemoddate(filename)
550   local foundfile = kpse.find_file(filename, "tex", true)
551   if foundfile then
552     local date = lfs.attributes(foundfile, "modification")
553     if date then
554       local d = os.date("*t", date)
555       if d.sec >= 60 then
556         d.sec = 59
557       end
558       local u = os.date("!*t", date)
559       local off = 60 * (d.hour - u.hour) + d.min - u.min
560       if d.year ~= u.year then
561         if d.year > u.year then
562           off = off + 1440
563         else
564           off = off - 1440
565         end
566       elseif d.yday ~= u.yday then
567         if d.yday > u.yday then
568           off = off + 1440
569         else
570           off = off - 1440
571         end
572       end
573       local timezone
574       if off == 0 then
575         timezone = "Z"
576       else
577         local hours = math.floor(off / 60)
578         local mins = math.abs(off - hours * 60)
579         timezone = string.format("%+03d'%02d'", hours, mins)
580       end
581       tex.write(string.format("D:%04d%02d%02d%02d%02d%s",
582           d.year, d.month, d.day, d.hour, d.min, d.sec, timezone))
583     end
584   end
585 end
586 function filedump(offset, length, filename)
587   length = tonumber(length)
588   if length and length > 0 then
589     local foundfile = kpse.find_file(filename, "tex", true)
590     if foundfile then
591       offset = tonumber(offset)
592       if not offset then
593         offset = 0

```

```

594     end
595     local filehandle = io.open(foundfile, "r")
596     if filehandle then
597         if offset > 0 then
598             filehandle:seek("set", offset)
599         end
600         local dump = filehandle:read(length)
601         escapehex(dump)
602     end
603 end
604 end
605 end
606 function md5fivesum(str, mode)
607     if mode == "byte" then
608         str = utf8_to_byte(str)
609     end
610     escapehex(md5.sum(str))
611 end
612 function filemd5fivesum(filename)
613     local foundfile = kpse.find_file(filename, "tex", true)
614     if foundfile then
615         local filehandle = io.open(foundfile, "r")
616         if filehandle then
617             local contents = filehandle:read("*a")
618             escapehex(md5.sum(contents))
619         end
620     end
621 end
622 function shellescape()
623     if os.execute then
624         tex.write("1")
625     else
626         tex.write("0")
627     end
628 end
629 function system(cmdline)
630     systemexitstatus = nil
631     texio.write_nl("log", "system(" .. cmdline .. ") ")
632     if os.execute then
633         texio.write("log", "executed.")
634         systemexitstatus = os.execute(cmdline)
635     else
636         texio.write("log", "disabled.")
637     end
638 end
639 function lastsystemstatus()
640     local result = tonumber(systemexitstatus)
641     if result then
642         local x = math.floor(result / 256)
643         tex.write(result - 256 * math.floor(result / 256))
644     end
645 end
646 function lastsystemexit()
647     local result = tonumber(systemexitstatus)
648     if result then
649         tex.write(math.floor(result / 256))
650     end
651 end
652 function pipe(cmdline)
653     local result
654     systemexitstatus = nil
655     texio.write_nl("log", "pipe(" .. cmdline .. ") ")

```

```

656 if io.popen then
657   texio.write("log", "executed.")
658   local handle = io.popen(cmdline, "r")
659   if handle then
660     result = handle:read("*a")
661     handle:close()
662   end
663 else
664   texio.write("log", "disabled.")
665 end
666 if result then
667   tex.settoks(toks, result)
668 else
669   tex.settoks(toks, "")
670 end
671 end
672 
```

### 3 Test

#### 3.1 Catcode checks for loading

```

673 <*test1>
674 \catcode`\'=1 %
675 \catcode`\}=2 %
676 \catcode`\#=6 %
677 \catcode`\@=11 %
678 \expandafter\ifx\csname count@\endcsname\relax
679   \countdef\count@=255 %
680 \fi
681 \expandafter\ifx\csname @gobble\endcsname\relax
682   \long\def\@gobble#1{}%
683 \fi
684 \expandafter\ifx\csname @firstofone\endcsname\relax
685   \long\def\@firstofone#1{#1}%
686 \fi
687 \expandafter\ifx\csname loop\endcsname\relax
688   \expandafter\@firstofone
689 \else
690   \expandafter\@gobble
691 \fi
692 {%
693   \def\loop#1\repeat{%
694     \def\body{#1}%
695     \iterate
696   }%
697   \def\iterate{%
698     \body
699     \let\next\iterate
700   \else
701     \let\next\relax
702   \fi
703   \next
704 }%
705   \let\repeat=\fi
706 }%
707 \def\RestoreCatcodes{}
708 \count@=0 %
709 \loop
710   \edef\RestoreCatcodes{%
711     \RestoreCatcodes
712     \catcode\the\count@=\the\catcode\count@\relax

```

```

713  }%
714 \ifnum\count@<255 %
715   \advance\count@ 1 %
716 \repeat
717
718 \def\RangeCatcodeInvalid#1#2{%
719   \count@=#1\relax
720   \loop
721     \catcode\count@=15 %
722   \ifnum\count@<#2\relax
723     \advance\count@ 1 %
724   \repeat
725 }
726 \expandafter\ifx\csname LoadCommand\endcsname\relax
727   \def\LoadCommand{\input pdftexcmds.sty\relax}%
728 \fi
729 \def\Test{%
730   \RangeCatcodeInvalid{0}{47}%
731   \RangeCatcodeInvalid{58}{64}%
732   \RangeCatcodeInvalid{91}{96}%
733   \RangeCatcodeInvalid{123}{255}%
734   \catcode`\@=12 %
735   \catcode`\\=0 %
736   \catcode`{=1 %
737   \catcode`}=2 %
738   \catcode`\#=6 %
739   \catcode`\[=12 %
740   \catcode`\]=12 %
741   \catcode`\%=14 %
742   \catcode`\ =10 %
743   \catcode13=5 %
744   \LoadCommand
745   \RestoreCatcodes
746 }
747 \Test
748 \csname @@end\endcsname
749 \end
750 </test1>

```

## 4 Installation

### 4.1 Download

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

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

[CTAN:macros/latex/contrib/oberdiek/pdftexcmds.pdf](http://CTAN:macros/latex/contrib/oberdiek/pdftexcmds.pdf) Documentation.

**Bundle.** All the packages of the bundle ‘oberdiek’ are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

[CTAN:install/macros/latex/contrib/oberdiek.tds.zip](http://CTAN:install/macros/latex/contrib/oberdiek.tds.zip)

TDS refers to the standard “A Directory Structure for T<sub>E</sub>X Files” ([CTAN:tds/tds.pdf](http://CTAN:tds/tds.pdf)). Directories with `texmf` in their name are usually organized this way.

---

<sup>1</sup>[ftp://ftp.ctan.org/tex-archive/](http://ftp.ctan.org/tex-archive/)

## 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 `TDSScripts/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-T<sub>E</sub>X:

```
tex pdftexcmds.dtx
```

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

|                                      |   |
|--------------------------------------|---|
| <code>pdftexcmds.sty</code>          | → <code>tex/generic/oberdiek/pdftexcmds.sty</code>      |
| <code>oberdiek.pdftexcmds.lua</code> | → <code>scripts/oberdiek/oberdiek.pdftexcmds.lua</code> |
| <code>pdftexcmds.lua</code>          | → <code>scripts/oberdiek/pdftexcmds.lua</code>          |
| <code>pdftexcmds.pdf</code>          | → <code>doc/latex/oberdiek/pdftexcmds.pdf</code>        |
| <code>pdftexcmds.dtx</code>          | → <code>source/latex/oberdiek/pdftexcmds.dtx</code>     |

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 T<sub>E</sub>X distribution (teT<sub>E</sub>X, mikT<sub>E</sub>X, ...) relies on file name databases, you must refresh these. For example, teT<sub>E</sub>X 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 pdftexcmds.pdf unpack_files output .
```

**Unpacking with L<sup>A</sup>T<sub>E</sub>X.** The `.dtx` chooses its action depending on the format:

**plain-T<sub>E</sub>X:** Run `docstrip` and extract the files.

**L<sup>A</sup>T<sub>E</sub>X:** Generate the documentation.

If you insist on using L<sup>A</sup>T<sub>E</sub>X for `docstrip` (really, `docstrip` does not need L<sup>A</sup>T<sub>E</sub>X), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{pdftexcmds.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<sup>A</sup>T<sub>E</sub>X:

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

## 5 History

[2007/11/11 v0.1]

- First version.

[2007/11/12 v0.2]

- Short description fixed.

[2007/12/12 v0.3]

- Organization of Lua code as module.

[2009/04/10 v0.4]

- Adaptation for syntax change of `\directlua` in LUAT<sub>E</sub>X 0.36.

[2009/09/22 v0.5]

- `\pdf@primitive`, `\pdf@ifprimitive` added.
- XeT<sub>E</sub>X's variants are detected for `\pdf@shellescape`, `\pdf@strcmp`, `\pdf@primitive`, `\pdf@ifprimitive`.

## 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                                    | Numbers                              |
|--|--------------------------------------|
| <code>\#</code> . . . . .                  | <code>676, 738</code>                |
| <code>\%</code> . . . . .                  | <code>741</code>                     |
| <code>\@</code> . . . . .                  | <code>677, 734</code>                |
| <code>\@PackageInfoNoLine</code> . . . . . | <code>121, 123, 225, 237, 242</code> |
| <code>\@firstofone</code> . . . . .        | <code>685, 688</code>                |
| <code>\@gobble</code> . . . . .            | <code>682, 690</code>                |
| <code>\@undefined</code> . . . . .         | <code>52, 224</code>                 |
| <code>\[</code> . . . . .                  | <code>739</code>                     |
| <code>\\"</code> . . . . .                 | <code>510, 513, 735</code>           |
| <code>\{</code> . . . . .                  | <code>674, 736</code>                |
| <code>\}</code> . . . . .                  | <code>675, 737</code>                |
| <code>\]</code> . . . . .                  | <code>740</code>                     |
|  |                                      |
|  | <b>A</b>                             |
|  | <code>\advance</code> . . . . .      |
|  | <code>715, 723</code>                |
|  | <code>\aftergroup</code> . . . . .   |
|  | <code>26</code>                      |
|  | <b>B</b>                             |
|  | <code>\body</code> . . . . .         |
|  | <code>694, 698</code>                |
|  | <b>C</b>                             |
|  | <code>\catcode</code> . . . . .      |
|  | <code>3, 4, 5,</code>                |

|  |   |
|--|---|
| 6, 7, 8, 9, 17, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 64, 65, 68, 69, 70, 71, 75, 76, 77, 78, 82, 84, 290, 389, 674, 675, 676, 677, 712, 721, 734, 735, 736, 737, 738, 739, 740, 741, 742, 743 | P   |
| \count@ . . . . . 679, 708, 712, 714, 715, 719, 721, 722, 723  | \PackageInfo . . . . . 23   |
| \countdef . . . . . 679  | \pdf@escapehex . . . . . 3, 139, 298  |
| \csname . . . . . 10, 18, 44, 60, 67, 112, 128, 131, 133, 147, 155, 163, 193, 198, 201, 202, 215, 217, 220, 239, 244, 250, 253, 259, 261, 274, 283, 286, 678, 681, 684, 687, 726, 748                        | \pdf@escapename . . . . . 139, 303  |
|  | \pdf@escapenamenative . . . . . 327   |
|  | \pdf@escapenamenative . . . . . 332   |
|  | \pdf@escapestring . . . . . 322   |
|  | \pdf@filedump . . . . . 3, 158, 347   |
|  | \pdf@filemdfivesum . . . . . 3, 168, 364  |
|  | \pdf@filemoddate . . . . . 3, 342   |
|  | \pdf@filesize . . . . . 3, 337  |
|  | \pdf@ifprimitive . . . . . 4, 210, 243  |
|  | \pdf@lastsystemexit . . . . . 384   |
|  | \pdf@lastsystemstatus . . . . . 379   |
|  | \pdf@cmdfivesum . . . . . 3, 166, 167, 354  |
|  | \pdf@cmdfivesumnative . . . . . 167, 359  |
|  | \pdf@pipe . . . . . 4, 390  |
|  | \pdf@primitive . . . . . 4, 206, 224, 226, 238  |
|  | \pdf@shellescape . . . . . 3, 150, 369  |
|  | \pdf@strcmp . . . . . 3, 292  |
|  | \pdf@system . . . . . 3, 170, 374   |
|  | \pdf@unescapehex . . . . . 3, 141, 308  |
|  | \pdf@unescapehexnative . . . . . 4, 141, 315  |
|  | \pdffiledump . . . . . 159  |
|  | \pdfmdfivesum . . . . . 166, 168  |
|  | \pdfprimitive . . . . . 227   |
|  | \pdfshellescape . . . . . 151   |
|  | \pdftexcmds@AtEnd . . . . . 80, 81, 106, 107, 270, 405  |
|  | \pdftexcmds@directlua . . . . . 175, 183, 279, 391, 399   |
|  | \pdftexcmds@nopdftex . . . . . 122, 124, 129, 148, 156, 164   |
|  | \pdftexcmds@strip@prefix . . . . . 190, 196   |
|  | \pdftexcmds@temp . . . . . 126, 137, 138, 140, 142, 143, 144, 145, 191, 206, 207, 208, 209, 210, 211, 212, 213, 248, 266, 267 |
|  | \pdftexcmds@toks . . . . . 282, 309, 316, 398   |
|  | \pdftexrevision . . . . . 222   |
|  | \pdftexversion . . . . . 219  |
|  | \ProvidesPackage . . . . . 15, 61   |
|  | R   |
|  | \RangeCatcodeInvalid . . . . . 718, 730, 731, 732, 733  |
|  | \repeat . . . . . 693, 705, 716, 724  |
|  | \RequirePackage . . . . . 116, 117, 277   |
|  | \RestoreCatcodes . . . . . 707, 710, 711, 745   |
|  | S   |
|  | \space . . . . . 227, 238, 243  |
|  | T   |
|  | \Test . . . . . 729, 747  |
|  | \the . . . . . 68, 69, 70, 71, 82, 309, 316, 398, 712   |
|  | \TMP@EnsureCode . . . . . 79, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105            |
|  | \toksdef . . . . . 284  |
|  | W   |
|  | \write . . . . . 20, 46, 171  |

| <b>X</b>  | <b>Y</b>                         |
|---|----------------------------------|
|   | \y ..... 198, 200, 254, 256, 258 |
| \x ... 10, 11, 14, 19, 23, 25, 45, 50,<br>60, 66, 74, 195, 196, 200, 253, 258 | \z ..... 255, 256                |