

# The embedfile package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2007/04/11 v1.1

## Abstract

This package embeds files to a PDF document. Currently pdfTeX 1.30 in PDF mode is supported only.

## Contents

<b>1</b>	<b>Documentation</b>	<b>1</b>
1.1	Introduction	1
1.1.1	Future development	2
1.2	User interface	2
1.3	Examples	3
1.3.1	plain-TeX	3
1.4	Package dtx-attach	3
<b>2</b>	<b>Implementation</b>	<b>4</b>
2.1	Reload check and package identification	4
2.2	Tools	5
2.3	Check for recent pdfTeX in PDF mode	5
2.4	Key value definitions	6
2.5	Embed the file	6
<b>3</b>	<b>Installation</b>	<b>9</b>
3.1	Download	9
3.2	Bundle installation	9
3.3	Package installation	10
3.4	Refresh file name databases	10
3.5	Some details for the interested	10
<b>4</b>	<b>References</b>	<b>11</b>
<b>5</b>	<b>History</b>	<b>11</b>
	[2006/08/16 v1.0]	11
	[2007/04/11 v1.1]	11
<b>6</b>	<b>Index</b>	<b>11</b>

## 1 Documentation

### 1.1 Introduction

The PDF format ([3]) allows the inclusion of files inside the PDF document. The included files can be bound to an annotation on a page. Or they can be recorded in a sorted list of embedded files. The packages `attachfile` or `attachfile2` follow the first approach, this package uses the latter method.

### 1.1.1 Future development

My dream is a large package that merges the features of all these packages mentioned before:

- Files can be attached to a page.
- Files can be attached to the document.
- An easy user interface for simple, common tasks and beginners.
- An interface for the advanced users that want to setup every detail.
- Support of many drivers (pdftex, dvips, dvipdfm, ...).
- ...

However, I have not managed to take the time for this project. Instead:

- First I experimented with package `attachfile`, adding driver support, fixing bugs, .... The result is currently named as `attachfile2`. It uses an external script to get file properties (size, date, checksum, ...).
- In order to avoid an external program for getting basic file properties I provided a patch “EscapeAndOther” for pdfTeX that was accepted for version 1.30.
- This package closes a gap left by the packages for attaching files and allows the embedding of files to the document. Also it makes use of the new primitives of pdfTeX.

## 1.2 User interface

This package `embedfile` can be used with both L<sup>A</sup>T<sub>E</sub>X and plain-T<sub>E</sub>X. See [subsubsection 1.3.1](#) that explains the use with plain-T<sub>E</sub>X by an example. In L<sup>A</sup>T<sub>E</sub>X the package is loaded as usually. There are no options.

```
\usepackage{embedfile}
```

`\embedfile [options] {file}`

The macro `\embedfile` includes file *file* and attaches it to the PDF document. At the end of the document the sorted list of embedded files are written. Thus you can safely use `\embedfile` before `\end{document}`. Embedding files using `\AtEndDocument` will only work, if `\AtEndDocument` is called before loading the package `embedfile`.

The *options* are give as key value pairs. The following keys are supported:

**filespec** This allows to override the file name that appears in the PDF file. If you are using other than simple file names (8bit, path separators, ...), look into the PDF specification ([3]). There are rules how these file names must be written/encoded.

**filesystem** This sets the entry /FS in the file specification dictionary, see PDF specification ([3]). Example: `filesystem=URL`.

**mimetype** This sets the mime type ([4]) of the file, see [subsubsection 1.3.1](#) for examples and [5] for a list of officially registered types.

**desc** The description for the file.

**stringmethod** The package must convert the values of the keys `filespec` and `desc` into a PDF string. If `hyperref` is found, then its `\pdfstringdef` will be used, otherwise pdfTeX’s `\pdfescapestring` is used. Value `psd` forces the use of `\pdfstringdef`, value `escape` the use of `\pdfescapestring`.

## 1.3 Examples

### 1.3.1 plain-TeX

The package can be used with plain-TeX. Because it uses package `keyval` and `\@ifnextchar`, it needs a little help from `miniltx.tex`.

If additionally package `keyval` (`graphicx`) is needed, load it first. Then package `embedfile` avoids a duplicate loading of package `keyval`.

Because plain-TeX does not provide a hook at end of the document, you have to call `\embedfilefinish` manually at the end after the last embedded file.

```
1 (*exampleplain)
2 % Load packages
3 \input miniltx
4 % \def\Gin@driver{pdftex.def}
5 % \input graphicx.sty
6 \input embedfile.sty
7 \resetatcatcode
8
9 % default setting
10 \embedfilesetup{
11   mimetype=text/plain
12 }
13
14 % Embed files
15 \embedfile[
16   filespec=example.tex,
17   desc={Source code (plain-TeX) of this example}
18 ]{embedfile-example-plain.tex}
19
20 \embedfile[
21   desc={Source of package 'embedfile'}
22 ]{embedfile.dtx}
23
24 \embedfile[
25   mimetype=application/pdf,
26   desc={Documentation of package 'embedfile'}
27 ]{embedfile.pdf}
28
29 % Some text
30 This example document contains three embedded files.
31
32 % End of document
33 \embedfilefinish % don't forget
34 \bye
35 /exampleplain)
```

### 1.4 Package `dtx-attach`

Package `dtx-attach` is just a small application of package `embedfile`. I am using it for the CTAN documentation of my packages in [CTAN:macros/latex/contrib/oberdiek/](https://ctan.org/ctan/ma/macros/latex/contrib/oberdiek/). It also serves as small example for the use of the package with L<sup>A</sup>T<sub>E</sub>X.

```
36 (*dtxattach)
37 \NeedsTeXFormat{LaTeX2e}
38 \ProvidesPackage{dtx-attach}
39 [2007/04/11 v1.1 Embed \string\jobname.dtx (H0)]%
40 \RequirePackage{embedfile}[2007/04/11]
41 \embedfile[%
42   stringmethod=escape,%
43   mimetype=plain/text,%
44   desc={LaTeX docstrip source archive for package '\jobname'}%
45 ]{\jobname.dtx}
46 /dtxattach)
```

## 2 Implementation

47 `\package`

### 2.1 Reload check and package identification

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

```
48 \begingroup
49 \catcode44 12 % ,
50 \catcode45 12 % -
51 \catcode46 12 % .
52 \catcode58 12 % :
53 \catcode64 11 % @
54 \expandafter\let\expandafter\x\csname ver@embedfile.sty\endcsname
55 \ifcase 0%
56 \ifx\x\relax % plain
57 \else
58 \ifx\x\empty % LaTeX
59 \else
60 1%
61 \fi
62 \fi
63 \else
64 \expandafter\ifx\csname PackageInfo\endcsname\relax
65 \def\x#1#2{%
66 \immediate\write-1{Package #1 Info: #2.}%
67 }%
68 \else
69 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
70 \fi
71 \x{embedfile}{The package is already loaded}%
72 \endgroup
73 \expandafter\endinput
74 \fi
75 \endgroup
```

Package identification:

```
76 \begingroup
77 \catcode40 12 % (
78 \catcode41 12 % )
79 \catcode44 12 % ,
80 \catcode45 12 % -
81 \catcode46 12 % .
82 \catcode47 12 % /
83 \catcode58 12 % :
84 \catcode64 11 % @
85 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
86 \def\x#1#2#3[#4]{\endgroup
87 \immediate\write-1{Package: #3 #4}%
88 \xdef#1{#4}%
89 }%
90 \else
91 \def\x#1#2[#3]{\endgroup
92 #2[#{#3}]%
93 \ifx#1\relax
94 \xdef#1{#3}%
95 \fi
96 }%
97 \fi
98 \expandafter\x\csname ver@embedfile.sty\endcsname
99 \ProvidesPackage{embedfile}%
100 [2007/04/11 v1.1 embed files into PDF (H0)]
101 \edef\EmbedFileRestoreCatcodes{%
```

```

102 \catcode39 \the\catcode39 % '
103 \catcode40 \the\catcode40 % (
104 \catcode41 \the\catcode41 % )
105 \catcode47 \the\catcode47 % /
106 \catcode60 \the\catcode60 % <
107 \catcode62 \the\catcode62 % >
108 \catcode64 \the\catcode64 % @
109 \catcode91 \the\catcode91 % [
110 \catcode93 \the\catcode93 % ]
111 \catcode96 \the\catcode96 % '
112 }
113 \catcode39 12 % '
114 \catcode40 12 % (
115 \catcode41 12 % )
116 \catcode47 12 % /
117 \catcode60 12 % <
118 \catcode62 12 % >
119 \catcode64 11 % @
120 \catcode91 12 % [
121 \catcode93 12 % ]
122 \catcode96 12 % '

```

## 2.2 Tools

`\EmbedFile@PackageError`

```

123 \begingroup\expandafter\expandafter\expandafter\endgroup
124 \expandafter\ifx\csname PackageError\endcsname\relax
125 \def\EmbedFile@PackageError#1#2{%
126 \errhelp{#2}%
127 \errmessage{Package embedfile Error: #1.}%
128 }%
129 \else
130 \def\EmbedFile@PackageError#1#2{%
131 \begingroup
132 \let\on@line\empty
133 \PackageError{embedfile}{#1}{#2}%
134 \endgroup
135 }%
136 \fi

```

`\EmbedFile@RequirePackage`

```

137 \begingroup\expandafter\expandafter\expandafter\endgroup
138 \expandafter\ifx\csname RequirePackage\endcsname\relax
139 \def\EmbedFile@RequirePackage#1#2{%
140 \expandafter\ifx\csname #2\endcsname\relax
141 \input #1.sty\relax
142 \fi
143 }%
144 \else
145 \def\EmbedFile@RequirePackage#1#2{%
146 \expandafter\ifx\csname #2\endcsname\relax
147 \RequirePackage{#1}%
148 \fi
149 }%
150 \fi

```

## 2.3 Check for recent pdfTeX in PDF mode

Load package ifpdf and check mode.

```

151 \EmbedFile@RequirePackage{ifpdf}{ifpdf}
152 \ifpdf

```

```

153 \else
154   \EmbedFile@PackageError{%
155     Missing pdfTeX in PDF mode%
156   }{%
157     Currently other drivers are not supported. Package loading is aborted.%
158   }%
159   \EmbedFileRestoreCatcodes
160   \expandafter\endinput
161 \fi

Check version.

162 \begingroup\expandafter\expandafter\expandafter\endgroup
163 \expandafter\ifx\csname pdffilesize\endcsname\relax
164   \EmbedFile@PackageError{%
165     Unsupported pdfTeX version%
166   }{%
167     At least version 1.30 is necessary. Package loading is aborted.%
168   }%
169   \EmbedFileRestoreCatcodes
170   \expandafter\endinput
171 \fi

```

## 2.4 Key value definitions

```

172 \EmbedFile@RequirePackage{keyval}{define@key}

173 \def\EmbedFile@DefineKey#1#2{%
174   \define@key{EmbedFile}{#1}{%
175     \expandafter\def\csname EmbedFile@#1\endcsname{##1}%
176   }%
177   \expandafter\def\csname EmbedFile@#1\endcsname{#2}%
178 }

```

Subtype of the embedded file (optional).

```
179 \EmbedFile@DefineKey{mimetype}{}
```

File specification string.

```
180 \EmbedFile@DefineKey{filespec}{\EmbedFile@file}
```

File system (optional).

```
181 \EmbedFile@DefineKey{filesystem}{}
```

Description (optional).

```
182 \EmbedFile@DefineKey{desc}{}
```

Method for converting text to PDF strings.

```

183 \EmbedFile@DefineKey{stringmethod}{%
184   \ifx\pdfstringdef\@undefined
185     escape%
186   \else
187     \ifx\pdfstringdef\relax
188       escape%
189     \else
190       psd%
191     \fi
192   \fi
193 }

```

```
\embedfilesetup
```

```

194 \def\embedfilesetup{%
195   \setkeys{EmbedFile}%
196 }

```

## 2.5 Embed the file

```
\embedfile
```

```

197 \def\embedfile{%
198   \@ifnextchar[\EmbedFile@embedfile{\EmbedFile@embedfile[]}%
199 }

\EmbedFile@embedfile

200 \def\EmbedFile@embedfile[#1]#2{%
201   \ifEmbedFile@finished
202     \EmbedFile@PackageError{%
203       \string\embedfile\space after \string\embedfilefinish
204     }{%
205       The list of embedded files is already written.%
206     }%
207   \else
208     \begingroup
209     \def\EmbedFile@file{#2}%
210     \setkeys{EmbedFile}{#1}%
211     \expandafter\ifx\expandafter\\pdffilesize{\EmbedFile@file}\\%
212     \EmbedFile@PackageError{%
213       File '\EmbedFile@file' not found%
214     }{%
215       The unknown file is not embedded.%
216     }%
217   \else
218     \EmbedFile@convert\EmbedFile@filespec\EmbedFile@@filespec
219     \ifx\EmbedFile@desc\empty
220       \let\EmbedFile@@desc\empty
221     \else
222       \EmbedFile@convert\EmbedFile@desc\EmbedFile@@desc
223     \fi
224     \immediate\pdfobj stream attr{%
225       /Type/EmbeddedFile%
226       \ifx\EmbedFile@mimetype\empty
227         \else
228           /Subtype/\pdfescapename{\EmbedFile@mimetype}%
229         \fi
230       /Params<<%
231         /ModDate(\pdffilemoddate{\EmbedFile@file})%
232         /Size \pdffilesize{\EmbedFile@file}%
233         /Checksum<\pdfmdfivesum file{\EmbedFile@file}>%
234       >>%
235     }file{\EmbedFile@file}\relax
236     \immediate\pdfobj{%
237       <<%
238         /Type/Filespec%
239         \ifx\EmbedFile@filesystem\empty
240         \else
241           /FS/\pdfescapename{\EmbedFile@filesystem}%
242         \fi
243         /F(\EmbedFile@@filespec)%
244         \ifx\EmbedFile@@desc\empty
245         \else
246           /Desc(\EmbedFile@@desc)%
247         \fi
248         /EF<<%
249           /F \the\pdflastobj\space 0 R%
250         >>%
251       >>%
252     }%
253     \EmbedFile@add{%
254       \EmbedFile@@filespec
255     }\the\pdflastobj\space 0 R%
256   \fi
257 \endgroup

```

```

258 \fi
259 }

\EmbedFile@convert
260 \def\EmbedFile@convert#1#2{%
261 \ifnum\pdfstrcmp{\EmbedFile@stringmethod}{psd}=0 %
262 \pdfstringdef\EmbedFile@temp{#1}%
263 \let#2\EmbedFile@temp
264 \else
265 \edef#2{\pdfescapestring{#1}}%
266 \fi
267 }

268 \global\let\EmbedFile@list\empty

\EmbedFile@add
Sorting is done by the insertion sort algorithm. Probably the sorting could be done
more reliable. However, the PDF specification is not too clear to me regarding
precise sorting rules (how to deal with different encodings, escaped characters,
...).
269 \def\EmbedFile@add#1#2{%
270 \begingroup
271 \edef\key{\pdfescapehex{#1}}%
272 \ifx\EmbedFile@list\empty
273 \xdef\EmbedFile@list{\noexpand\do{\key}{#2}}%
274 \else
275 \def\do##1##2{%
276 \ifnum\pdfstrcmp{##1}{\key}>0 %
277 \edef\x{%
278 \toks@{%
279 \the\toks@%
280 \noexpand\do{\key}{#2}%
281 \noexpand\do{##1}{##2}%
282 }%
283 }%
284 \x
285 \def\do####1####2{%
286 \toks@\expandafter{\the\toks@\do{####1}{####2}}%
287 }%
288 \def\stop{%
289 \xdef\EmbedFile@list{\the\toks@}%
290 }%
291 \else
292 \toks@\expandafter{\the\toks@\do{##1}{##2}}%
293 \fi
294 }%
295 \def\stop{%
296 \xdef\EmbedFile@list{\the\toks@\noexpand\do{\key}{#2}}%
297 }%
298 \toks@{}%
299 \EmbedFile@list\stop
300 \fi
301 \endgroup
302 }

303 \newif\ifEmbedFile@finished

\embedfilefinish
304 \def\embedfilefinish{%
305 \ifEmbedFile@finished
306 \EmbedFile@PackageError{%
307 Too many invocations of \string\embedfilefinish
308 }{%

```

```

309     The list of embedded files is already written.%
310 }%
311 \else
312   \ifx\EmbedFile@list\empty
313   \else
314     \global\EmbedFile@finishedtrue
315     \begingroup
316       \def\do##1##2{%
317         <##1>##2%
318       }%
319       \immediate\pdfobj{%
320         <<%
321           /Names[\EmbedFile@list]%
322         >>%
323       }%
324       \pdfnames{%
325         /EmbeddedFiles \the\pdflastobj\space 0 R%
326       }%
327     \endgroup
328   \fi
329 \fi
330 }

331 \begingroup\expandafter\expandafter\expandafter\endgroup
332 \expandafter\ifx\csname AtEndDocument\endcsname\relax
333 \else
334   \AtEndDocument{\embedfilefinish}%
335 \fi

336 \EmbedFileRestoreCatcodes
337 </package>

```

## 3 Installation

### 3.1 Download

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

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

[CTAN:macros/latex/contrib/oberdiek/embedfile.pdf](#) Documentation.

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

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

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

### 3.2 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
```

---

<sup>1</sup><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/
```

### 3.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain- $\TeX$ :

```
tex embedfile.dtx
```

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

<code>embedfile.sty</code>	$\rightarrow$ <code>tex/latex/oberdiek/embedfile.sty</code>
<code>dtx-attach.sty</code>	$\rightarrow$ <code>tex/latex/oberdiek/dtx-attach.sty</code>
<code>embedfile.pdf</code>	$\rightarrow$ <code>doc/latex/oberdiek/embedfile.pdf</code>
<code>embedfile-example-plain.tex</code>	$\rightarrow$ <code>doc/latex/oberdiek/embedfile-example-plain.tex</code>
<code>embedfile.dtx</code>	$\rightarrow$ <code>source/latex/oberdiek/embedfile.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`.

### 3.4 Refresh file name databases

If your  $\TeX$  distribution (te $\TeX$ , mi $\TeX$ , ...) relies on file name databases, you must refresh these. For example, te $\TeX$  users run `texhash` or `mktextlsr`.

### 3.5 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 embedfile.pdf unpack_files output .
```

**Unpacking with L $\TeX$ .** The `.dtx` chooses its action depending on the format:

**plain- $\TeX$ :** Run `docstrip` and extract the files.

**L $\TeX$ :** Generate the documentation.

If you insist on using L $\TeX$  for `docstrip` (really, `docstrip` does not need L $\TeX$ ), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{embedfile.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 $\TeX$ :

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

## 4 References

- [1] Scott Pakin: *The attachfile package*; 2005/02/20 v1.2; [CTAN:macros/latex/contrib/attachfile/](#).
- [2] Heiko Oberdiek: *The attachfile2 package*; 2006/08/16 v2.2; [CTAN:macros/latex/contrib/oberdiek/attachfile2.pdf](#).
- [3] Adobe Systems Incorporated: *PDF Reference, Fifth Edition, Version 1.6*, November 2004; [http://partners.adobe.com/public/developer/pdf/index\\_reference.html](http://partners.adobe.com/public/developer/pdf/index_reference.html).
- [4] Network Working Group: RFC 2046, *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*, November 1996; <http://www.rfc-editor.org/>.
- [5] IANA (Internet Assigned Numbers Authority): *MIME Media Types*, May 2006; <http://www.iana.org/assignments/media-types/>.

## 5 History

[2006/08/16 v1.0]

- First public version.

[2007/04/11 v1.1]

- 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		D	
<code>\@ifnextchar</code>	198	<code>\define@key</code>	174
<code>\@undefined</code>	184	<code>\do</code>	273, 275,
<code>\@</code>	211		280, 281, 285, 286, 292, 296, 316
A		E	
<code>\AtEndDocument</code>	334	<code>\embedfile</code>	2, 15, 20, 24, 41, 197, 203
B		<code>\EmbedFile@desc</code>	220, 222, 244, 246
<code>\bye</code>	34	<code>\EmbedFile@filespec</code>	218, 243, 254
C		<code>\EmbedFile@add</code>	253, 269
<code>\catcode</code>	49, 50, 51, 52, 53, 77, 78, 79, 80, 81, 82, 83, 84, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122	<code>\EmbedFile@convert</code>	218, 222, 260
<code>\csname</code>	54, 64, 85, 98, 124, 138, 140, 146, 163, 175, 177, 332	<code>\EmbedFile@DefineKey</code>	173, 179, 180, 181, 182, 183
		<code>\EmbedFile@desc</code>	219, 222
		<code>\EmbedFile@embedfile</code>	198, 200
		<code>\EmbedFile@file</code>	180,
			209, 211, 213, 231, 232, 233, 235
		<code>\EmbedFile@filespec</code>	218

\EmbedFile@filesystem	239, 241	N	
\EmbedFile@finishedtrue	314	\NeedsTeXFormat	37
\EmbedFile@list	268, 272, 273, 289, 296, 299, 312, 321	\newif	303
\EmbedFile@mimetype	226, 228	O	
\EmbedFile@PackageError	123, 154, 164, 202, 212, 306	\on@line	132
\EmbedFile@RequirePackage	137, 151, 172	P	
\EmbedFile@stringmethod	261	\PackageError	133
\EmbedFile@temp	262, 263	\PackageInfo	69
\embedfilefinish	33, 203, 304, 334	\pdfescapehex	271
\EmbedFileRestoreCatcodes	101, 159, 169, 336	\pdfescapename	228, 241
\embedfilesetup	10, 194	\pdfescapestring	265
\empty	58, 132, 219, 220, 226, 239, 244, 268, 272, 312	\pdffilemoddate	231
\encsname	140	\pdffilesize	211, 232
\endcsname	54, 64, 85, 98, 124, 138, 146, 163, 175, 177, 332	\pdflastobj	249, 255, 325
\endinput	73, 160, 170	\pdfmdfivesum	233
\errhelp	126	\pdfnames	324
\errmessage	127	\pdfobj	224, 236, 319
G		\pdfstrcmp	261, 276
\Gin@driver	4	\pdfstringdef	184, 187, 262
I		\ProvidesPackage	38, 99
\ifcase	55	R	
\ifEmbedFile@finished	201, 303, 305	\RequirePackage	40, 147
\ifnum	261, 276	\resetatcatcode	7
\ifpdf	152	S	
\ifx	56, 58, 64, 85, 93, 124, 138, 140, 146, 163, 184, 187, 211, 219, 226, 239, 244, 272, 312, 332	\setkeys	195, 210
\immediate	66, 87, 224, 236, 319	\space	203, 249, 255, 325
\input	3, 5, 6, 141	\stop	288, 295, 299
J		T	
\jobname	39, 44, 45	\the	102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 249, 255, 279, 286, 289, 292, 296, 325
K		\toks@	278, 279, 286, 289, 292, 296, 298
\key	271, 273, 276, 280, 296	W	
		\write	66, 87
		X	
		\x	54, 56, 58, 65, 69, 71, 86, 91, 98, 277, 284