

The embedfile package

Heiko Oberdiek
<heiko.oberdiek at gmail.com>

2010/03/01 v2.5

Abstract

This package embeds files to a PDF document. Currently the only supported driver is pdf \TeX \geq 1.30 in PDF mode.

Contents

1	Documentation	2
1.1	Introduction	2
1.1.1	Future development	2
1.2	User interface	2
1.3	Collection support (PDF 1.7)	4
1.4	Export of object references	5
1.4.1	Example	5
1.5	Examples	6
1.5.1	plain \TeX	6
1.5.2	Collection example	6
1.6	Package <code>dtx-attach</code>	7
2	Implementation	8
2.1	Reload check and package identification	8
2.2	Catcodes	9
2.3	Tools	9
2.4	Check for recent pdf \TeX in PDF mode	10
2.5	Strings	10
2.6	Switches	11
2.7	Key value definitions	12
2.8	Embed the file	17
3	Test	21
3.1	Catcode checks for loading	21
3.2	Simple test	23
3.3	Test for ini- \TeX	23
4	Installation	24
4.1	Download	24
4.2	Bundle installation	24
4.3	Package installation	24
4.4	Refresh file name databases	25
4.5	Some details for the interested	25
5	References	25

6 History	26
[2006/08/16 v1.0]	26
[2007/04/11 v1.1]	26
[2007/09/09 v1.2]	26
[2007/10/28 v2.0]	26
[2007/10/29 v2.1]	26
[2007/11/11 v2.2]	26
[2007/11/25 v2.3]	26
[2009/09/25 v2.4]	26
[2010/03/01 v2.5]	27
7 Index	28

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^AT_EX and plain T_EX. See [subsection 1.5.1](#) that explains the use with plain T_EX by an example. In L^AT_EX 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 (8-bit, path separators, ...), look into the PDF specification ([3]). There are rules how these file names must be written/encoded. Avoid 8-bit characters and other special characters, the behaviour is currently undefined. Use option `ucfilespec` for more funny file names. The string method, see below, is `escape` since version 2.4.

This name is also used as entry in a name tree (see PDF specification: `/EmbeddedFiles`). Therefore the value for `filespec` must be unique among all embedded files. Also key `initialfiles` refers to this name, if the file name and the value of `filespec` are different.

ucfilespec Since PDF 1.7 the file name may be provided in Unicode. The conversion of the option value into a PDF string is controlled by option `stringmethod`.

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 [subsection 1.5.1](#) 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 `ucfilespec` and `desc` into a PDF string (before version 2.4: `filespec` and `desc`). 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`.

<key>.value Sets the value of a collection item property, see [section 1.3](#).

<key>.prefix Sets the prefix of a collection item property, see [section 1.3](#).

id The value must be an unique name. Macros `\embedfileifobjectexists` and `\embedfilegetobject` are using this name later.

`\embedfilefinish`

The list of all embedded files must be added as data structure in the PDF file. In case of `LATEX` this is automatically done. The package uses `\AtEndDocument`. Then the list of all files should be known. However, plain `TEX` does not know about `\AtEndDocument`. Thus the user must call `\embedfilefinish` at the end of the document after the last file is embedded.

`\embedfilesetup {options}`

Options for `\embedfile` and collection support can be set in `\embedfilesetup`.

1.3 Collection support (PDF 1.7)

Since PDF 1.7 the embedded files can form a *collection* (sometimes referred as *package*), the main document is called *cover sheet*. See PDF specification 8.2.4 “Collections” and 3.10.5 “Collection items” [3].

Usually Acrobat Reader 7 or 8 shows the embedded files in a table at the bottom with the following columns:

Name	Description	Modified	Size
...

If the files form a collection, then they are displayed in a table left or top (depending on option `view`, see `\embedfilesetup`).

Collection support is enabled automatically, if it is used.

`\embedfilesetup` $\{ \langle options \rangle \}$

The following options are supported in addition to options for `\embedfile`:

view If the PDF file contains a collection, then Acrobat Reader 8 shows a line at the top below the menu bar and the toolbar. It shows the current selected file, icons for changing the view mode, an options menu. The initial mode how the collection is presented is set by this option `view`. The following modes/values are supported, the default is **details**:

details The full collection table is displayed at the top below the collection bar.

tile The files of the collection are shown in tile mode on the left.

hidden The collection table is not shown.

initialfile Selects the file that is initially presented. Especially useful for an embedded PDF file that is then shown instead of the cover document. There must be an `\embedfile` command somewhere whose value for key `filespec` is used here. The `\embedfile` command can drop option `filespec` if the file name is not different.

`\embedfilefield` $\{ \langle key \rangle \} \{ \langle options \rangle \}$

Macro `\embedfilefield` defines a column/field in the collection table. The name of the field is $\langle key \rangle$.

type sets the type of the field. The supported values are:

text A text field. Its value is set in `\embedfile` by option $\langle key \rangle$.value.

date A date field. Its value is set in `\embedfile` by option $\langle key \rangle$.value. A special format is required, see “3.8.3 Dates” [3].

number A field with an integer or float number. Its value is set in `\embedfile` by option $\langle key \rangle$.value.

file The file name of the embedded file.

desc The description text of the embedded file. It is set in `\embedfile` by option `desc`.

moddate The modification date of the embedded file.

size The size of the embedded file.

All types allow the use of a prefix that is disregarded by sorting. The prefix for this field is set in `\embedfile` by option $\langle key \rangle$.prefix.

title sets the column title.

visible controls whether the column is presented:

true shows the column.

false hides the column.

Default: **true**

edit Allows the editing of field values. Does not seem to have an effect for Acrobat Reader.

true enables the feature, if available (depends on the PDF viewer).

false disables the feature.

Default: **false**

The order of `\embedfilefield` statements defines the order of the columns.

`\embedfilesort {<key-sort-list>}`

The sort order of the embedded files are controlled by macro `\embedfilesort`. `<key-sort-list>` defines the sort order. The key is a field name defined by `\embedfilefield`. Its value is either **ascending** or **descending**. The default is **ascending**.

1.4 Export of object references

Caution: This feature is still experimental. It may be even removed in future versions. Therefore feedback would be nice, if someone has a useful application for this feature.

Object numbers are saved, if `id` is given in `\embedfile`. The following objects are supported:

- `EmbeddedFile`
- `Filespec`

`\embedfileifobjectexists {<id>} {<type>} {<then>} {<else>}`

Macro `\embedfileifobjectexists` tests whether object of `<type>` is available for the embedded file identified by `<id>`.

`\embedfilegetobject {<id>} {<type>}`

Macro `\embedfilegetobject` expands to the full object reference object of `<type>` for the embedded file identified by `<id>`.

1.4.1 Example

```
\embedfile[id={foo}]{foo.pdf}
\embedfileifobjectexists{foo}{Filespec}{%
  \typeout{%
    FileSpec object for 'foo': %
    \embedfilegetobject{foo}{Filespec}%
  }%
}{%
  \typeout{No Filespec object for 'foo'}%
}
```

1.5 Examples

1.5.1 plain T_EX

The package can be used with plain T_EX. It can be used with or without 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 T_EX 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 %<<END
3 % Load packages
4 \input miniltx
5 % \def\Gin@driver{pdftex.def}
6 % \input graphicx.sty
7 \input embedfile.sty
8 \resetatcatcode
9
10 % default setting
11 \embedfilesetup{
12   mimetype=text/plain
13 }
14
15 % Embed files
16 \embedfile[
17   filespec=example.tex,
18   desc={Source code (plain-TeX) of this example}
19 ]{embedfile-example-plain.tex}
20
21 \embedfile[
22   desc={Source of package 'embedfile'}
23 ]{embedfile.dtx}
24
25 \embedfile[
26   mimetype=application/pdf,
27   desc={Documentation of package 'embedfile'}
28 ]{embedfile.pdf}
29
30 % Some text
31 This example document contains three embedded files.
32
33 % End of document
34 \embedfilefinish % don't forget
35 \bye
36 %END
37 </exampleplain)
```

1.5.2 Collection example

```
38 (*examplecollection)
39 %<<END
40 \NeedsTeXFormat{LaTeX2e}
41 \documentclass{article}
42 \usepackage[bookmarks=false]{hyperref}
43 % provides \pdfstringdef that is then used by 'title' and
44 % other keys.
45 \usepackage{embedfile}[2010/03/01]
46 \embedfilesetup{
47   view=details,
48   initialfile=embedfile.pdf
49 }
50 \embedfilefield{file}{
```

```

51 type=file,
52 title={File name}
53 }
54 \embedfilefield{description}{
55 type=desc,
56 title={Description}
57 }
58 \embedfilefield{date}{
59 type=moddate,
60 title={Date}
61 }
62 \embedfilefield{size}{
63 type=size,
64 title={Size}
65 }
66 \embedfilefield{type}{
67 type=text,
68 title={Type},
69 visible=false
70 }
71 \embedfilesort{
72 type,
73 date=descending
74 }
75 \begin{document}
76 An example for embedded files as collection.
77 You need Acrobat Reader 8 or higher.
78
79 \embedfile[
80 desc={Source file of package 'embedfile'},
81 description.prefix={Package: },
82 type.value={DTX}
83 ]{embedfile.dtx}
84
85 \embedfile[
86 desc={Documentation of package 'embedfile'},
87 description.prefix={Package: },
88 type.value={PDF}
89 ]{embedfile.pdf}
90
91 \embedfile[
92 desc={The source for this example},
93 description.prefix={Example: },
94 type.value={TEX}
95 ]{\jobname.tex}
96
97 \end{document}
98 %END
99 </examplecollection>

```

1.6 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:macros/latex/contrib/oberdiek/). It also serves as small example for the use of the package with L^AT_EX.

```

100 (*dtxattach)
101 \NeedsTeXFormat{LaTeX2e}
102 \ProvidesPackage{dtx-attach}
103 [2010/03/01 v2.5 Embed \string\jobname.dtx (HO)]%
104 \RequirePackage{embedfile}[2010/03/01]
105 \embedfile[%
106 stringmethod=escape,%
107 mimetype=plain/text,%

```

```

108 desc={LaTeX docstrip source archive for package ‘\jobname’}%
109 ]{\jobname.dtx}
110 </dtxattach>

```

2 Implementation

```
111 <*package>
```

2.1 Reload check and package identification

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

```

112 \begingroup
113 \catcode44 12 % ,
114 \catcode45 12 % -
115 \catcode46 12 % .
116 \catcode58 12 % :
117 \catcode64 11 % @
118 \catcode123 1 % {
119 \catcode125 2 % }
120 \expandafter\let\expandafter\x\csname ver@embedfile.sty\endcsname
121 \ifx\x\relax % plain-TeX, first loading
122 \else
123 \def\empty{}%
124 \ifx\x\empty % LaTeX, first loading,
125 % variable is initialized, but \ProvidesPackage not yet seen
126 \else
127 \catcode35 6 % #
128 \expandafter\ifx\csname PackageInfo\endcsname\relax
129 \def\x#1#2{%
130 \immediate\write-1{Package #1 Info: #2.}%
131 }%
132 \else
133 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
134 \fi
135 \x{embedfile}{The package is already loaded}%
136 \aftergroup\endinput
137 \fi
138 \fi
139 \endgroup

```

Package identification:

```

140 \begingroup
141 \catcode35 6 % #
142 \catcode40 12 % (
143 \catcode41 12 % )
144 \catcode44 12 % ,
145 \catcode45 12 % -
146 \catcode46 12 % .
147 \catcode47 12 % /
148 \catcode58 12 % :
149 \catcode64 11 % @
150 \catcode91 12 % [
151 \catcode93 12 % ]
152 \catcode123 1 % {
153 \catcode125 2 % }
154 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
155 \def\x#1#2#3[#4]{\endgroup
156 \immediate\write-1{Package: #3 #4}%
157 \xdef#1{#4}%
158 }%
159 \else
160 \def\x#1#2[#3]{\endgroup
161 #2[#{#3}]%

```



```

162     \ifx#1\@undefined
163     \xdef#1{#3}%
164     \fi
165     \ifx#1\relax
166     \xdef#1{#3}%
167     \fi
168   }%
169   \fi
170 \expandafter\x\csname ver@embedfile.sty\endcsname
171 \ProvidesPackage{embedfile}%
172 [2010/03/01 v2.5 embed files into PDF (HO)]

```

2.2 Catcodes

```

173 \begingroup
174 \catcode123 1 % {
175 \catcode125 2 % }
176 \def\x{\endgroup
177   \expandafter\edef\csname EmFi@AtEnd\endcsname{%
178     \catcode35 \the\catcode35\relax
179     \catcode64 \the\catcode64\relax
180     \catcode123 \the\catcode123\relax
181     \catcode125 \the\catcode125\relax
182   }%
183 }%
184 \x
185 \catcode35 6 % #
186 \catcode64 11 % @
187 \catcode123 1 % {
188 \catcode125 2 % }
189 \def\TMP@EnsureCode#1#2{%
190   \edef\EmFi@AtEnd{%
191     \EmFi@AtEnd
192     \catcode#1 \the\catcode#1\relax
193   }%
194   \catcode#1 #2\relax
195 }
196 \TMP@EnsureCode{39}{12}% '
197 \TMP@EnsureCode{40}{12}% (
198 \TMP@EnsureCode{41}{12}% )
199 \TMP@EnsureCode{44}{12}% ,
200 \TMP@EnsureCode{46}{12}% .
201 \TMP@EnsureCode{47}{12}% /
202 \TMP@EnsureCode{58}{12}% :
203 \TMP@EnsureCode{60}{12}% <
204 \TMP@EnsureCode{61}{12}% =
205 \TMP@EnsureCode{62}{12}% >
206 \TMP@EnsureCode{91}{12}% [
207 \TMP@EnsureCode{93}{12}% ]
208 \TMP@EnsureCode{96}{12}% `

```

2.3 Tools

\EmFi@RequirePackage

```

209 \begingroup\expandafter\expandafter\expandafter\endgroup
210 \expandafter\ifx\csname RequirePackage\endcsname\relax
211   \def\EmFi@RequirePackage#1[#2]{%
212     \input #1.sty\relax
213   }%
214 \else
215   \let\EmFi@RequirePackage\RequirePackage
216 \fi

```

`\EmFi@Error`

```
217 \EmFi@RequirePackage{infwarerr}[2007/09/09]%
218 \def\EmFi@Error{%
219   \@PackageError{embedfile}%
220 }
```

2.4 Check for recent pdfTeX in PDF mode

Load package ifpdf and check mode.

```
221 \EmFi@RequirePackage{ifpdf}[2007/09/09]
222 \ifpdf
223 \else
224   \EmFi@Error{%
225     Missing pdfTeX in PDF mode%
226   }{%
227     Currently other drivers are not supported. %
228     Package loading is aborted.%
229   }%
230   \EmFi@AtEnd
231   \expandafter\endinput
232 \fi

233 \EmFi@RequirePackage{pdftexcmds}[2007/11/11]
234 \EmFi@RequirePackage{ltxcmds}[2010/03/01]
235 \EmFi@RequirePackage{kvsetkeys}[2010/03/01]
236 \EmFi@RequirePackage{kvdefinekeys}[2010/03/01]
```

Check version.

```
237 \begingroup\expandafter\expandafter\expandafter\endgroup
238 \expandafter\ifx\csname pdf@filesize\endcsname\relax
239   \EmFi@Error{%
240     Unsupported pdfTeX version%
241   }{%
242     At least version 1.30 is necessary. Package loading is aborted.%
243   }%
244   \EmFi@AtEnd
245   \expandafter\endinput
246 \fi
```

2.5 Strings

Minimal version of package pdfescape is 2007/08/27 v1.5 because of `\EdefSanitize`.

```
247 \EmFi@RequirePackage{pdfescape}[2007/11/11]
248 \def\EmFi@temp#1{%
249   \expandafter\EdefSanitize\csname EmFi@S@#1\endcsname{#1}%
250 }
```

`\EmFi@details`

```
251 \EmFi@temp{details}%
```

`\EmFi@tile`

```
252 \EmFi@temp{tile}%
```

`\EmFi@hidden`

```
253 \EmFi@temp{hidden}%
```

`\EmFi@S@text`

```
254 \EmFi@temp{text}
```

`\EmFi@S@date`

```
255 \EmFi@temp{date}
```

```

\EmFi@S@number
256 \EmFi@temp{number}

\EmFi@S@file
257 \EmFi@temp{file}

\EmFi@S@desc
258 \EmFi@temp{desc}

\EmFi@S@moddate
259 \EmFi@temp{moddate}

\EmFi@S@creationdate
260 \EmFi@temp{creationdate}

\EmFi@S@size
261 \EmFi@temp{size}

\EmFi@S@ascending
262 \EmFi@temp{ascending}

\EmFi@S@descending
263 \EmFi@temp{descending}

\EmFi@S@true
264 \EmFi@temp{true}

\EmFi@S@false
265 \EmFi@temp{false}

```

2.6 Switches

```

\ifEmFi@collection
266 \ltx@newif\ifEmFi@collection

\ifEmFi@sort
267 \ltx@newif\ifEmFi@sort

\ifEmFi@visible
268 \ltx@newif\ifEmFi@visible

\ifEmFi@edit
269 \ltx@newif\ifEmFi@edit

\ifEmFi@item
270 \ltx@newif\ifEmFi@item

\ifEmFi@finished
271 \ltx@newif\ifEmFi@finished

\ifEmFi@id
272 \ltx@newif\ifEmFi@id

```

2.7 Key value definitions

`\EmFi@GlobalKey`

```
273 \def\EmFi@GlobalKey#1#2{%
274   \global\expandafter\let\csname KV@#1@#2\expandafter\endcsname
275   \csname KV@#1@#2\endcsname
276 }
```

`\EmFi@GlobalDefaultKey`

```
277 \def\EmFi@GlobalDefaultKey#1#2{%
278   \EmFi@GlobalKey{#1}{#2}%
279   \global\expandafter\let
280   \csname KV@#1@#2@default\expandafter\endcsname
281   \csname KV@#1@#2@default\endcsname
282 }
```

`\EmFi@DefineKey`

```
283 \def\EmFi@DefineKey#1#2{%
284   \kv@define@key{EmFi}{#1}{%
285     \expandafter\def\csname EmFi@#1\endcsname{##1}%
286   }%
287   \expandafter\def\csname EmFi@#1\endcsname{#2}%
288 }
```

Subtype of the embedded file (optional).

```
289 \EmFi@DefineKey{mimetype}{}
```

File specification string.

```
290 \EmFi@DefineKey{filespec}{\EmFi@file}
```

File specification string in Unicode.

```
291 \EmFi@DefineKey{ucfilespec}{}
```

File system (optional).

```
292 \EmFi@DefineKey{filesystem}{}
```

Description (optional).

```
293 \EmFi@DefineKey{desc}{}
```

Method for converting text to PDF strings.

```
294 \EmFi@DefineKey{stringmethod}{%
295   \ifx\pdfstringdef\@undefined
296     escape%
297   \else
298     \ifx\pdfstringdef\relax
299       escape%
300     \else
301       psd%
302     \fi
303   \fi
304 }
```

Option id as key for object numbers.

```
305 \kv@define@key{EmFi}{id}{%
306   \def\EmFi@id{#1}%
307   \EmFi@idtrue
308 }
```

`\EmFi@defobj`

```
309 \def\EmFi@defobj#1{%
310   \ifEmFi@id
311     \expandafter\xdef\csname EmFi@#1@\EmFi@id\endcsname{%
312       \the\pdflastobj\ltx@space 0 R%
313     }%
314   \fi
315 }
```

\embedfileifobjectexists

```
316 \def\embedfileifobjectexists#1#2{%
317   \expandafter\ifx\csname EmFi@#2@#1\endcsname\relax
318   \expandafter\ltx@secondoftwo
319   \else
320   \expandafter\ltx@firstoftwo
321   \fi
322 }
```

\embedfilegetobject

```
323 \def\embedfilegetobject#1#2{%
324   \embedfileifobjectexists{#1}{#2}{%
325     \csname EmFi@#2@#1\endcsname
326   }{%
327     O O R%
328   }%
329 }
```

Initial view of the collection.

```
330 \kv@define@key{EmFi}{view}[]{%
331   \EdefSanitize\EmFi@temp{#1}%
332   \def\EmFi@next{%
333     \global\EmFi@collectiontrue
334   }%
335   \ifx\EmFi@temp\ltx@empty
336     \let\EmFi@view\EmFi@S@details
337   \else\ifx\EmFi@temp\EmFi@S@details
338     \let\EmFi@view\EmFi@S@details
339   \else\ifx\EmFi@temp\EmFi@S@tile
340     \let\EmFi@view\EmFi@S@tile
341   \else\ifx\EmFi@temp\EmFi@S@hidden
342     \let\EmFi@view\EmFi@S@hidden
343   \else
344     \let\EmFi@next\relax
345     \EmFi@Error{%
346       Unknown value '\EmFi@temp' for key 'view'.\MessageBreak
347       Supported values: 'details', 'tile', 'hidden'.%
348     }\@ehc
349   \fi\fi\fi\fi
350   \EmFi@next
351 }
352 \EmFi@DefineKey{initialfile}{}
```

\embedfilesetup

```
353 \def\embedfilesetup{%
354   \ifEmFi@finished
355     \def\EmFi@next##1{%
356       \EmFi@Error{%
357         \string\embedfilefield\ltx@space after \string\embedfilefinish
358       }{%
359         The list of embedded files is already written.%
360       }%
361     \else
362     \def\EmFi@next{%
363       \kvsetkeys{EmFi}%
364     }%
365     \fi
366     \EmFi@next
367 }
```

\EmFi@schema

```
368 \def\EmFi@schema{}
```

```

\EmFi@order
369 \gdef\EmFi@order{0}

\EmFi@@order
370 \let\EmFi@@order\relax

\EmFi@fieldlist
371 \def\EmFi@fieldlist{}

\EmFi@sortcase
372 \def\EmFi@sortcase{0}%

\embedfilefield
373 \def\embedfilefield#1#2{%
374   \ifEmFi@finished
375     \EmFi@Error{%
376       \string\embedfilefield\ltx@space after \string\embedfilefinish
377     }{%
378       The list of embedded files is already written.%
379     }%
380   \else
381     \global\EmFi@collectiontrue
382     \EdefSanitize\EmFi@key{#1}%
383     \expandafter\ifx\csname KV@EmFi@\EmFi@key.prefix\endcsname\relax
384     \begingroup
385       \count@=\EmFi@order
386       \advance\count@ 1 %
387       \xdef\EmFi@order{\the\count@}%
388       \let\EmFi@title\EmFi@key
389       \let\EmFi@type\EmFi@S@text
390       \EmFi@visibletrue
391       \EmFi@editfalse
392       \kvsetkeys{EmFiFi}{#2}%
393       \EmFi@convert\EmFi@title\EmFi@title
394       \xdef\EmFi@schema{%
395         \EmFi@schema
396         /\pdf@escapename{\EmFi@key}<<%
397         /Subtype/%
398         \ifx\EmFi@type\EmFi@S@date D%
399         \else\ifx\EmFi@type\EmFi@S@number N%
400         \else\ifx\EmFi@type\EmFi@S@file F%
401         \else\ifx\EmFi@type\EmFi@S@desc Desc%
402         \else\ifx\EmFi@type\EmFi@S@moddate ModDate%
403         \else\ifx\EmFi@type\EmFi@S@creationdate CreationDate%
404         \else\ifx\EmFi@type\EmFi@S@size Size%
405         \else S%
406         \fi\fi\fi\fi\fi\fi\fi
407         /N(\EmFi@title)%
408         \EmFi@@order{\EmFi@order}%
409         \ifEmFi@visible
410         \else
411           /V false%
412         \fi
413         \ifEmFi@edit
414         /E true%
415         \fi
416         >>%
417       }%
418     \let\do\relax
419     \xdef\EmFi@fieldlist{%
420       \EmFi@fieldlist

```

```

421     \do{\EmFi@key}%
422 }%
423 \ifx\EmFi@type\EmFi@S@text
424     \kv@define@key{EmFi}{\EmFi@key.value}{%
425     \EmFi@itemtrue
426     \def\EmFi@temp{##1}%
427     \EmFi@convert\EmFi@temp\EmFi@temp
428     \expandafter\def\csname EmFi@V@#1%
429     \expandafter\endcsname\expandafter{%
430     \expandafter(\EmFi@temp)%
431     }%
432 }%
433 \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
434 \else\ifx\EmFi@type\EmFi@S@date
435     \kv@define@key{EmFi}{\EmFi@key.value}{%
436     \EmFi@itemtrue
437     \def\EmFi@temp{##1}%
438     \EmFi@convert\EmFi@temp\EmFi@temp
439     \expandafter\def\csname EmFi@V@#1%
440     \expandafter\endcsname\expandafter{%
441     \expandafter(\EmFi@temp)%
442     }%
443 }%
444 \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
445 \else\ifx\EmFi@type\EmFi@S@number
446     \kv@define@key{EmFi}{\EmFi@key.value}{%
447     \EmFi@itemtrue
448     \expandafter\edef\csname EmFi@V@#1\endcsname{ ##1}%
449     }%
450     \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
451 \fi\fi\fi
452 \kv@define@key{EmFi}{\EmFi@key.prefix}{%
453     \EmFi@itemtrue
454     \expandafter\def\csname EmFi@P@#1\endcsname{##1}%
455 }%
456 \EmFi@GlobalKey{EmFi}{\EmFi@key.prefix}%
457 \kv@define@key{EmFiSo}{\EmFi@key}[ascending]{%
458     \edef\EmFi@temp{##1}%
459     \ifx\EmFi@temp\EmFi@S@ascending
460     \def\EmFi@temp{true}%
461     \else\ifx\EmFi@temp\EmFi@S@descending
462     \def\EmFi@temp{false}%
463     \else
464     \def\EmFi@temp{}%
465     \EmFi@Error{%
466         Unknown sort order '\EmFi@temp'.\MessageBreak
467         Supported values: '\EmFi@S@ascending', %
468         '\EmFi@S@descending
469     }\@ehc
470 \fi\fi
471 \ifx\EmFi@temp\ltx@empty
472 \else
473     \xdef\EmFi@sortkeys{%
474     \EmFi@sortkeys
475     /\pdf@escapename{#1}%
476     }%
477     \ifx\EmFi@sortorders\ltx@empty
478     \global\let\EmFi@sortorders\EmFi@temp
479     \gdef\EmFi@sortcase{1}%
480     \else
481     \xdef\EmFi@sortorders{%
482     \EmFi@sortorders

```

```

483         \ltx@space
484         \EmFi@temp
485     }%
486     \xdef\EmFi@sortcase{2}%
487     \fi
488     \fi
489     }%
490     \EmFi@GlobalDefaultKey{EmFiSo}\EmFi@key
491 \endgroup
492 \else
493     \EmFi@Error{%
494         Field '\EmFi@key' is already defined%
495     }\@ehc
496     \fi
497 \fi
498 }

499 \kv@define@key{EmFiFi}{type}{%
500     \EdefSanitize\EmFi@temp{#1}%
501     \ifx\EmFi@temp\EmFi@S@text
502         \let\EmFi@type\EmFi@temp
503     \else\ifx\EmFi@temp\EmFi@S@date
504         \let\EmFi@type\EmFi@temp
505     \else\ifx\EmFi@temp\EmFi@S@number
506         \let\EmFi@type\EmFi@temp
507     \else\ifx\EmFi@temp\EmFi@S@file
508         \let\EmFi@type\EmFi@temp
509     \else\ifx\EmFi@temp\EmFi@S@desc
510         \let\EmFi@type\EmFi@temp
511     \else\ifx\EmFi@temp\EmFi@S@moddate
512         \let\EmFi@type\EmFi@temp
513     \else\ifx\EmFi@temp\EmFi@S@creationdate
514         \let\EmFi@type\EmFi@temp
515     \else\ifx\EmFi@temp\EmFi@S@size
516         \let\EmFi@type\EmFi@temp
517     \else
518         \EmFi@Error{%
519             Unknown type '\EmFi@temp'.\MessageBreak
520             Supported types: 'text', 'date', 'number', 'file',\MessageBreak
521             'desc', 'moddate', 'creationdate', 'size'%
522         }%
523     \fi\fi\fi\fi\fi\fi\fi\fi
524 }

525 \kv@define@key{EmFiFi}{title}{%
526     \def\EmFi@title{#1}%
527 }

```

\EmFi@setboolean

```

528 \def\EmFi@setboolean#1#2{%
529     \EdefSanitize\EmFi@temp{#2}%
530     \ifx\EmFi@temp\EmFi@S@true
531         \csname EmFi@#1true\endcsname
532     \else
533         \ifx\EmFi@temp\EmFi@S@false
534             \csname EmFi@#1false\endcsname
535         \else
536             \EmFi@Error{%
537                 Unknown value '\EmFi@temp' for key '#1'.\MessageBreak
538                 Supported values: 'true', 'false'%
539             }\@ehc
540         \fi
541     \fi
542 }

```



```

543 \kv@define@key{EmFiFi}{visible}[true]{%
544   \EmFi@setboolean{visible}{#1}%
545 }
546 \kv@define@key{EmFiFi}{edit}[true]{%
547   \EmFi@setboolean{edit}{#1}%
548 }

```

\EmFi@sortkeys

```
549 \def\EmFi@sortkeys{}
```

\EmFi@sortorders

```
550 \def\EmFi@sortorders{}
```

\embedfilesort

```

551 \def\embedfilesort{%
552   \kvsetkeys{EmFiSo}%
553 }

```

2.8 Embed the file

\embedfile

```

554 \def\embedfile{%
555   \ltx@ifnextchar[\EmFi@embedfile{\EmFi@embedfile[]}]%
556 }

```

\EmFi@embedfile

```

557 \def\EmFi@embedfile[#1]#2{%
558   \ifEmFi@finished
559     \EmFi@Error{%
560       \string\embedfile\ltx@space after \string\embedfilefinish
561     }{%
562       The list of embedded files is already written.%
563     }%
564   \else
565     \begingroup
566     \def\EmFi@file{#2}%
567     \kvsetkeys{EmFi}{#1}%
568     \expandafter\expandafter\expandafter
569     \ifx\expandafter\expandafter\expandafter
570       \\pdf@filesize{\EmFi@file}\\\\%
571       \EmFi@Error{%
572         File '\EmFi@file' not found%
573       }{%
574         The unknown file is not embedded.%
575       }%
576     \else
577       \edef\EmFi@@filespec{%
578         \pdf@escapestring{\EmFi@filespec}%
579       }%
580       \ifx\EmFi@ucfilespec\ltx@empty
581         \let\EmFi@@ucfilespec\ltx@empty
582       \else
583         \EmFi@convert\EmFi@ucfilespec\EmFi@@ucfilespec
584       \fi
585       \ifx\EmFi@desc\ltx@empty
586         \let\EmFi@@desc\ltx@empty
587       \else
588         \EmFi@convert\EmFi@desc\EmFi@@desc
589       \fi
590       \ifEmFi@item
591         \let\do\EmFi@do

```

```

592     \immediate\pdfobj{%
593     <<%
594     \EmFi@fieldlist
595     >>%
596     }%
597     \edef\EmFi@ci{\the\pdflastobj}%
598     \fi
599     \immediate\pdfobj stream attr{%
600     /Type/EmbeddedFile%
601     \ifx\EmFi@mimetype\ltx@empty
602     \else
603     /Subtype/\pdf@escapename{\EmFi@mimetype}%
604     \fi
605     /Params<<%
606     /ModDate(\pdf@filemoddate{\EmFi@file})%
607     /Size \pdf@filesize{\EmFi@file}%
608     /Checksum<\pdf@filemdfivesum{\EmFi@file}>%
609     >>%
610     }file{\EmFi@file}\relax
611     \EmFi@defobj{EmbeddedFile}%
612     \immediate\pdfobj{%
613     <<%
614     /Type/Filespec%
615     \ifx\EmFi@filesystem\ltx@empty
616     \else
617     /FS/\pdf@escapename{\EmFi@filesystem}%
618     \fi
619     /F(\EmFi@@filespec)%
620     \ifx\EmFi@@ucfilespec\ltx@empty
621     \else
622     /UF(\EmFi@@ucfilespec)%
623     \fi
624     \ifx\EmFi@@desc\ltx@empty
625     \else
626     /Desc(\EmFi@@desc)%
627     \fi
628     /EF<<%
629     /F \the\pdflastobj\ltx@space 0 R%
630     >>%
631     \ifEmFi@item
632     /CI \EmFi@ci\ltx@space 0 R%
633     \fi
634     >>%
635     }%
636     \EmFi@defobj{Filespec}%
637     \EmFi@add{%
638     \EmFi@@filespec
639     }{\the\pdflastobj\ltx@space 0 R}%
640     \fi
641     \endgroup
642     \fi
643 }

```

\EmFi@do

```

644 \def\EmFi@do#1{%
645   \expandafter\ifx\csname EmFi@P@#1\endcsname\relax
646   \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
647   \else
648     /\pdf@escapename{#1}\csname EmFi@V@#1\endcsname
649   \fi
650   \else
651     /\pdf@escapename{#1}<<%
652     \expandafter\ifx\csname EmFi@V@#1\endcsname\relax

```

```

653     \else
654         /D\csname EmFi@V@#1\endcsname
655     \fi
656     /P(\csname EmFi@P@#1\endcsname)%
657     >>%
658 \fi
659 }

```

\EmFi@convert

```

660 \def\EmFi@convert#1#2{%
661     \ifnum\pdf@strcmp{\EmFi@stringmethod}{psd}=0 %
662         \pdfstringdef\EmFi@temp{#1}%
663         \let#2\EmFi@temp
664     \else
665         \edef#2{\pdf@escapestring{#1}}%
666     \fi
667 }

668 \global\let\EmFi@list\ltx@empty

```

\EmFi@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, ...).

```

669 \def\EmFi@add#1#2{%
670     \begingroup
671     \ifx\EmFi@list\ltx@empty
672         \xdef\EmFi@list{\noexpand\do{#1}{#2}}%
673     \else
674         \def\do##1##2{%
675             \ifnum\pdf@strcmp{##1}{#1}>0 %
676                 \edef\x{%
677                     \toks@{%
678                         \the\toks@%
679                         \noexpand\do{#1}{#2}%
680                         \noexpand\do{##1}{##2}%
681                     }%
682                 }%
683                 \x
684             \def\do###1###2{%
685                 \toks@\expandafter{\the\toks@\do{###1}{###2}}%
686             }%
687             \def\stop{%
688                 \xdef\EmFi@list{\the\toks@}%
689             }%
690         \else
691             \toks@\expandafter{\the\toks@\do{##1}{##2}}%
692         \fi
693     }%
694     \def\stop{%
695         \xdef\EmFi@list{\the\toks@\noexpand\do{#1}{#2}}%
696     }%
697     \toks@{}%
698     \EmFi@list\stop
699 \fi
700 \endgroup
701 }

```

\embedfilefinish

```

702 \def\embedfilefinish{%
703     \ifEmFi@finished
704         \EmFi@Error{%

```

```

705     Too many invocations of \string\embedfilefinish
706   }-%
707     The list of embedded files is already written.%
708   }%
709   \else
710     \ifx\EmFi@list\ltx@empty
711     \else
Write /EmbeddedFiles entry.
712     \global\EmFi@finishedtrue
713     \begingroup
714     \def\do##1##2{%
715       (##1)##2%
716     }%
717     \immediate\pdfobj{%
718       <<%
719         /Names[\EmFi@list]%
720       >>%
721     }%
722     \pdfnames{%
723       /EmbeddedFiles \the\pdflastobj\ltx@space 0 R%
724     }%
725     \endgroup
Write collection objects.
726     \ifx\EmFi@initialfile\ltx@empty
727     \else
728       \EmFi@collectiontrue
729     \fi
730     \ifEmFi@collection
731       \ifx\EmFi@initialfile\ltx@empty
732         \let\EmFi@@initialfile\ltx@empty
733       \else
734         \edef\EmFi@@initialfile{%
735           \pdf@escapestring{\EmFi@initialfile}%
736         }%
737       \fi
Look for initial file among the embedded files.
738     \begingroup
739     \let\f=N%
740     \def\do##1##2{%
741       \def\x{##1}%
742       \ifx\x\EmFi@@initialfile
743         \let\f=Y%
744       \let\do\ltx@gobbletwo
745     \fi
746   }%
747   \EmFi@list
748   \expandafter\endgroup
749   \ifx\f Y%
750   \else
751     \@PackageWarningNoLine{embedfile}{%
752       Missing initial file '\EmFi@initialfile'\MessageBreak
753       among the embedded files%
754     }%
755     \let\EmFi@initialfile\ltx@empty
756     \let\EmFi@@initialfile\ltx@empty
757   \fi
758   \ifcase\EmFi@sortcase
759     \def\EmFi@temp{%
760   \or
761     \def\EmFi@temp{%
762       /S\EmFi@sortkeys

```

```

763         /A \EmFi@sortorders
764     }%
765 \else
766     \def\EmFi@temp{%
767         /S[\EmFi@sortkeys]%
768         /A[\EmFi@sortorders]%
769     }%
770 \fi
771 \def\EmFi@@order##1{%
772     \ifnum\EmFi@order>1 %
773         /O ##1%
774     \fi
775 }%
776 \immediate\pdfobj{%
777     <<%
778         \ifx\EmFi@schema\ltx@empty
779         \else
780             /Schema<<\EmFi@schema>>%
781         \fi
782         \ifx\EmFi@@initialfile\ltx@empty
783         \else
784             /D(\EmFi@@initialfile)%
785         \fi
786         \ifx\EmFi@view\EmFi@S@tile
787             /View/T%
788         \else\ifx\EmFi@view\EmFi@S@hidden
789             /View/H%
790         \fi\fi
791         \ifx\EmFi@temp\ltx@empty
792             \EmFi@temp
793         \else
794             /Sort<<\EmFi@temp>>%
795         \fi
796     >>%
797 }%
798 \pdfcatalog{%
799     /Collection \the\pdflastobj\ltx@space0 R%
800 }%
801 \fi
802 \fi
803 \fi
804 }

805 \begingroup\expandafter\expandafter\expandafter\endgroup
806 \expandafter\ifx\csname AtEndDocument\endcsname\relax
807 \else
808     \AtEndDocument{\embedfilefinish}%
809 \fi

810 \EmFi@AtEnd
811 </package>

```

3 Test

3.1 Catcode checks for loading

```

812 (*test1)
813 \catcode'\{=1 %
814 \catcode'\}=2 %
815 \catcode'\#=6 %
816 \catcode'\@=11 %
817 \expandafter\ifx\csname count@\endcsname\relax

```

```

818 \countdef\count@=255 %
819 \fi
820 \expandafter\ifx\csname @gobble\endcsname\relax
821 \long\def@gobble#1{%
822 \fi
823 \expandafter\ifx\csname @firstofone\endcsname\relax
824 \long\def@firstofone#1{#1}%
825 \fi
826 \expandafter\ifx\csname loop\endcsname\relax
827 \expandafter@firstofone
828 \else
829 \expandafter@gobble
830 \fi
831 {%
832 \def\loop#1\repeat{%
833 \def\body{#1}%
834 \iterate
835 }%
836 \def\iterate{%
837 \body
838 \let\next\iterate
839 \else
840 \let\next\relax
841 \fi
842 \next
843 }%
844 \let\repeat=\fi
845 }%
846 \def\RestoreCatcodes{}
847 \count@=0 %
848 \loop
849 \edef\RestoreCatcodes{%
850 \RestoreCatcodes
851 \catcode\the\count@=\the\catcode\count@\relax
852 }%
853 \ifnum\count@<255 %
854 \advance\count@ 1 %
855 \repeat
856
857 \def\RangeCatcodeInvalid#1#2{%
858 \count@=#1\relax
859 \loop
860 \catcode\count@=15 %
861 \ifnum\count@<#2\relax
862 \advance\count@ 1 %
863 \repeat
864 }
865 \expandafter\ifx\csname LoadCommand\endcsname\relax
866 \def\LoadCommand{\input embedfile.sty\relax}%
867 \fi
868 \def\Test{%
869 \RangeCatcodeInvalid{0}{47}%
870 \RangeCatcodeInvalid{58}{64}%
871 \RangeCatcodeInvalid{91}{96}%
872 \RangeCatcodeInvalid{123}{255}%
873 \catcode'\@=12 %
874 \catcode'\=0 %
875 \catcode'\{=1 %
876 \catcode'\}=2 %
877 \catcode'\#=6 %
878 \catcode'\ [=12 %
879 \catcode'\]=12 %

```

```

880 \catcode'\%=14 %
881 \catcode'\ =10 %
882 \catcode13=5 %
883 \LoadCommand
884 \RestoreCatcodes
885 }
886 \Test
887 \csname @@end\endcsname
888 \end
889 </test1>

```

3.2 Simple test

```

890 <*test2>
891 \input embedfile.sty\relax
892 \embedfile[%
893   stringmethod=escape,%
894   mimetype=plain/text,%
895   desc={LaTeX docstrip source archive for package 'embedfile'},%
896   id={embedfile.dtx}%
897 ]{embedfile.dtx}
898 \nopagenumbers
899 Test (plain-TeX): {\tt embedfile.dtx} should be embedded.%
900
901 \def\Test#1{%
902   \par
903   \embedfileifobjectexists{embedfile.dtx}{#1}{%
904     Object #1 (embedfile.dtx): %
905     \embedfilegetobject{embedfile.dtx}{#1}%
906   }{%
907     \errmessage{Missing object #1 (embedfile.dtx)}%
908   }%
909 }
910 \Test{EmbeddedFile}
911 \Test{Filespec}
912 \embedfilefinish
913 \bye
914 </test2>
915 <*test3>
916 \NeedsTeXFormat{LaTeX2e}
917 \let\SavedJobname\jobname
918 \def\jobname{embedfile}
919 \RequirePackage{dtx-attach}[2010/03/01]
920 \let\jobname\SavedJobname
921 \documentclass{minimal}
922 \begin{document}
923 Test (\LaTeX): \texttt{embedfile.dtx} should be embedded.%
924 \end{document}
925 </test3>

```

3.3 Test for ini-TeX

```

926 <*test4>
927 \catcode'\{=1 %
928 \catcode'\}=2 %
929 \input ifluatex.sty %
930 \ifluatex
931   \directlua{%
932     tex.enableprimitives('', {%
933       'pdflastobj',%
934       'pdfnames',%
935       'pdfobj',%
936       'pdfoutput'%

```

```

937   })%
938 }%
939 \fi
940 \pdfoutput=1 %
941 \input embedfile.sty %
942 \shipout\hbox{}
943 \embedfile[%
944   stringmethod=escape,%
945   mimetype=plain/text,%
946   desc={iniTeX source},%
947 ]{\jobname.tex}
948 \embedfilefinish
949 \end
950 </test4>

```

4 Installation

4.1 Download

Package. This package is available on CTAN¹:

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

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

4.2 Bundle installation

Unpacking. Unpack the `oberdiek.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip oberdiek.tds.zip -d ~/texmf
```

Script installation. Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package `attachfile2` comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

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_EX:

```
tex embedfile.dtx
```

¹<http://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):

```

embedfile.sty           → tex/generic/oberdiek/embedfile.sty
dtx-attach.sty         → tex/generic/oberdiek/dtx-attach.sty
embedfile.pdf          → doc/latex/oberdiek/embedfile.pdf
embedfile-example-plain.tex → doc/latex/oberdiek/embedfile-example-plain.tex
embedfile-example-collection.tex → doc/latex/oberdiek/embedfile-example-collection.tex
test/embedfile-test1.tex → doc/latex/oberdiek/test/embedfile-test1.tex
test/embedfile-test2.tex → doc/latex/oberdiek/test/embedfile-test2.tex
test/embedfile-test3.tex → doc/latex/oberdiek/test/embedfile-test3.tex
test/embedfile-test4.tex → doc/latex/oberdiek/test/embedfile-test4.tex
embedfile.dtx          → source/latex/oberdiek/embedfile.dtx

```

If you have a `docstrip.cfg` that configures and enables `docstrip`'s TDS installing feature, then some files can already be in the right place, see the documentation of `docstrip`.

4.4 Refresh file name databases

If your \TeX distribution (`te \TeX` , `mik \TeX` , ...) relies on file name databases, you must refresh these. For example, `te \TeX` users run `texhash` or `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 embedfile.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{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 `pdf \LaTeX` :

```

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

```

5 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, Sixth Edition, Version 1.7*, Oktober 2006; http://www.adobe.com/devnet/pdf/pdf_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/>.

6 History

[2006/08/16 v1.0]

- First public version.

[2007/04/11 v1.1]

- Line ends sanitized.

[2007/09/09 v1.2]

- Fixes for plain-TeX, wrapper for package keyval added.
- Catcode section rewritten.

[2007/10/28 v2.0]

- Collection support added (PDF 1.7).

[2007/10/29 v2.1]

- Export of object references by adding new option `id` and new macros `\embedfileifobjectexists` and `\embedfilegetobject`.

[2007/11/11 v2.2]

- Use of package `pdftexcmds` for LuaTeX support.

[2007/11/25 v2.3]

- Fix in use of `\pdf@filesize`, bug introduced in previous version.

[2009/09/25 v2.4]

- Bug fix: If `hyperref` is used with option `unicode`, the Unicode encoded file name causes trouble. Therefore `\pdfstringdef` is now never used for option `filespec`, always method `escape` is applied (Peter Cibulka).
- Bug fix for `initialfile`.
- Bug fix for file names in `/EmbeddedFiles`.
- New option `ucfilespec` for file name support in Unicode (since PDF 1.7).

[2010/03/01 v2.5]

- Compatibility for ini- \TeX .
- Package `keyval` replaced by packages `kvsetkeys` and `kvdefinekeys` because of compatibility for ini- \TeX .
- TDS location moved from `TDS:tex/latex/oberdiek/embedfile.sty` to `TDS:tex/generic/oberdiek/eml`

7 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; plain numbers refer to the code lines where the entry is used.

Symbols	
\#	815, 877
\%	880
\@	816, 873
\@PackageError	219
\@PackageWarningNoLine	751
\@ehc	348, 469, 495, 539
\@firstofone	824, 827
\@gobble	821, 829
\@undefined	162, 295
\[878
\\	570, 874
\{	813, 875, 927
\}	814, 876, 928
\]	879
_	881
A	
\advance	386, 854, 862
\aftergroup	136
\atEndDocument	808
B	
\begin	75, 922
\body	833, 837
\bye	35, 913
C	
\catcode	113, 114, 115, 116, 117, 118, 119, 127, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 174, 175, 178, 179, 180, 181, 185, 186, 187, 188, 192, 194, 813, 814, 815, 816, 851, 860, 873, 874, 875, 876, 877, 878, 879, 880, 881, 882, 927, 928
\count@	385, 386, 387, 818, 847, 851, 853, 854, 858, 860, 861, 862
\countdef	818
\csname	120, 128, 154, 170, 177, 210, 238, 249, 274, 275, 280, 281, 285, 287, 311, 317, 325, 383, 428, 439, 448, 454, 531, 534, 645, 646, 648, 652, 654, 656, 806, 817, 820, 823, 826, 865, 887
D	
\directlua	931
\do	418, 421, 591, 672, 674, 679, 680, 684, 685, 691, 695, 714, 740, 744
\documentclass	41, 921
E	
\EdefSanitize	249, 331, 382, 448, 458, 500, 529
\embedfile	3, 16, 21, 25, 79, 85, 91, 105, <u>554</u> , 560, 892, 943
\embedfilefield	4, 50, 54, 58, 62, 66, 357, <u>373</u>
\embedfilefinish	3, 34, 357, 376, 560, <u>702</u> , 808, 912, 948
\embedfilegetobject	5, <u>323</u> , 905
\embedfileifobjectexists	5, <u>316</u> , 324, 903
\embedfilessetup	3, 4, 11, 46, <u>353</u>
\embedfilesort	5, 71, <u>551</u>
\EmFi@desc	586, 588, 624, 626
\EmFi@filespec	577, 619, 638
\EmFi@initialfile	732, 734, 742, 756, 782, 784
\EmFi@order	370, 408, 771
\EmFi@ucfilespec	581, 583, 620, 622
\EmFi@add	637, <u>669</u>
\EmFi@atEnd	190, 191, 230, 244, 810
\EmFi@ci	597, 632
\EmFi@collectiontrue	333, 381, 728
\EmFi@convert	393, 427, 438, 583, 588, <u>660</u>
\EmFi@DefineKey	283, 289, 290, 291, 292, 293, 294, 352
\EmFi@defobj	309, 611, 636
\EmFi@desc	585, 588
\EmFi@details	<u>251</u>
\EmFi@do	591, <u>644</u>
\EmFi@editfalse	391
\EmFi@embedfile	555, <u>557</u>
\EmFi@Error	217, 224, 239, 345, 356, 375, 465, 493, 518, 536, 559, 571, 704
\EmFi@fieldlist	371, 419, 420, 594
\EmFi@file	290, 566, 570, 572, 606, 607, 608, 610
\EmFi@filespec	578
\EmFi@filesystem	615, 617
\EmFi@finishedtrue	712
\EmFi@GlobalDefaultKey	277, 490
\EmFi@GlobalKey	273, 278, 433, 444, 450, 456
\EmFi@hidden	<u>253</u>
\EmFi@id	306, 311
\EmFi@idtrue	307
\EmFi@initialfile	726, 731, 735, 752, 755
\EmFi@itemtrue	425, 436, 447, 453
\EmFi@key	382, 383, 388, 396, 421, 424, 433, 435, 444, 446, 450, 452, 456, 457, 490, 494
\EmFi@list	668, 671, 672, 688, 695, 698, 710, 719, 747
\EmFi@mimetype	601, 603
\EmFi@next	332, 344, 350, 355, 362, 366

<code>\EmFi@order</code>	369, 385, 387, 408, 772
<code>\EmFi@RequirePackage</code>	209,
	217, 221, 233, 234, 235, 236, 247
<code>\EmFi@S@ascending</code>	262, 459, 467
<code>\EmFi@S@creationdate</code>	260, 403, 513
<code>\EmFi@S@date</code>	255, 398, 434, 503
<code>\EmFi@S@desc</code>	258, 401, 509
<code>\EmFi@S@descending</code>	263, 461, 468
<code>\EmFi@S@details</code>	336, 337, 338
<code>\EmFi@S@false</code>	265, 533
<code>\EmFi@S@file</code>	257, 400, 507
<code>\EmFi@S@hidden</code>	341, 342, 788
<code>\EmFi@S@moddate</code>	259, 402, 511
<code>\EmFi@S@number</code>	256, 399, 445, 505
<code>\EmFi@S@size</code>	261, 404, 515
<code>\EmFi@S@text</code>	254, 389, 423, 501
<code>\EmFi@S@tile</code>	339, 340, 786
<code>\EmFi@S@true</code>	264, 530
<code>\EmFi@schema</code>	368, 394, 395, 778, 780
<code>\EmFi@setboolean</code>	528, 544, 547
<code>\EmFi@sortcase</code>	372, 479, 486, 758
<code>\EmFi@sortkeys</code> 473, 474, 549, 762, 767	
<code>\EmFi@sortorders</code>	
	477, 478, 481, 482, 550, 763, 768
<code>\EmFi@stringmethod</code>	661
<code>\EmFi@temp</code>	248,
	251, 252, 253, 254, 255, 256,
	257, 258, 259, 260, 261, 262,
	263, 264, 265, 331, 335, 337,
	339, 341, 346, 426, 427, 430,
	437, 438, 441, 458, 459, 460,
	461, 462, 464, 466, 471, 478,
	484, 500, 501, 502, 503, 504,
	505, 506, 507, 508, 509, 510,
	511, 512, 513, 514, 515, 516,
	519, 529, 530, 533, 537, 662,
	663, 759, 761, 766, 791, 792, 794
<code>\EmFi@tile</code>	252
<code>\EmFi@title</code>	388, 393, 407, 526
<code>\EmFi@type</code>	
	389, 398, 399, 400, 401, 402,
	403, 404, 423, 434, 445, 502,
	504, 506, 508, 510, 512, 514, 516
<code>\EmFi@ucfilespec</code>	580, 583
<code>\EmFi@view</code> 336, 338, 340, 342, 786, 788	
<code>\EmFi@visibletrue</code>	390
<code>\empty</code>	123, 124
<code>\end</code>	97, 888, 924, 949
<code>\endcsname</code>	
	120, 128, 154, 170, 177, 210,
	238, 249, 274, 275, 280, 281,
	285, 287, 311, 317, 325, 383,
	429, 440, 448, 454, 531, 534,
	645, 646, 648, 652, 654, 656,
	806, 817, 820, 823, 826, 865, 887
<code>\endinput</code>	136, 231, 245
<code>\errmessage</code>	907
F	
<code>\f</code>	739, 743, 749
G	
<code>\gdef</code>	369, 479
<code>\Gin@driver</code>	5
H	
<code>\hbox</code>	942
I	
<code>\ifcase</code>	758
<code>\ifEmFi@collection</code>	266, 730
<code>\ifEmFi@edit</code>	269, 413
<code>\ifEmFi@finished</code> 271, 354, 374, 558, 703	
<code>\ifEmFi@id</code>	272, 310
<code>\ifEmFi@item</code>	270, 590, 631
<code>\ifEmFi@sort</code>	267
<code>\ifEmFi@visible</code>	268, 409
<code>\ifluatex</code>	930
<code>\ifnum</code>	661, 675, 772, 853, 861
<code>\ifpdf</code>	222
<code>\ifx</code>	121, 124, 128, 154, 162, 165,
	210, 238, 295, 298, 317, 335,
	337, 339, 341, 383, 398, 399,
	400, 401, 402, 403, 404, 423,
	434, 445, 459, 461, 471, 477,
	501, 503, 505, 507, 509, 511,
	513, 515, 530, 533, 569, 580,
	585, 601, 615, 620, 624, 645,
	646, 652, 671, 710, 726, 731,
	742, 749, 778, 782, 786, 788,
	791, 806, 817, 820, 823, 826, 865
<code>\immediate</code>	
	130, 156, 592, 599, 612, 717, 776
<code>\input</code>	4, 6, 7, 212, 866, 891, 929, 941
<code>\iterate</code>	834, 836, 838
J	
<code>\jobname</code>	95,
	103, 108, 109, 917, 918, 920, 947
K	
<code>\kv@define@key</code>	
	284, 305, 330, 424, 435,
	446, 452, 457, 499, 525, 543, 546
<code>\kvsetkeys</code>	363, 392, 552, 567
L	
<code>\LaTeX</code>	923
<code>\LoadCommand</code>	866, 883
<code>\loop</code>	832, 848, 859
<code>\ltx@empty</code>	335, 471, 477,
	580, 581, 585, 586, 601, 615,
	620, 624, 668, 671, 710, 726,
	731, 732, 755, 756, 778, 782, 791
<code>\ltx@firstoftwo</code>	320
<code>\ltx@gobbletwo</code>	744
<code>\ltx@ifnextchar</code>	555
<code>\ltx@newif</code>	
	266, 267, 268, 269, 270, 271, 272
<code>\ltx@secondoftwo</code>	318
<code>\ltx@space</code>	312, 357, 376,
	483, 560, 629, 632, 639, 723, 799
M	
<code>\MessageBreak</code>	
	346, 466, 519, 520, 537, 752

N		<code>\RestoreCatcodes</code> .. 846, 849, 850, 884
<code>\NeedsTeXFormat</code>	40, 101, 916	S
<code>\next</code>	838, 840, 842	<code>\SavedJobname</code>
<code>\nopagenumbers</code>	898	<code>\shipout</code>
P		<code>\stop</code>
<code>\PackageInfo</code>	133	T
<code>\par</code>	902	<code>\Test</code>
<code>\pdf@escapename</code>		<code>\TeX</code>
.....	396, 475, 603, 617, 648, 651	<code>\texttt</code>
<code>\pdf@escapestring</code>	578, 665, 735	<code>\the</code>
<code>\pdf@filemdfivesum</code>	608
<code>\pdf@filemoddate</code>	606	312, 387, 597, 629, 639, 678,
<code>\pdf@filesize</code>	570, 607	685, 688, 691, 695, 723, 799, 851
<code>\pdf@strcmp</code>	661, 675	<code>\TMP@EnsureCode</code>
<code>\pdfcatalog</code>	798
<code>\pdflastobj</code> 312, 597, 629, 639, 723, 799		196, 197, 198, 199, 200, 201,
<code>\pdfnames</code>	722	202, 203, 204, 205, 206, 207, 208
<code>\pdfobj</code>	592, 599, 612, 717, 776	<code>\toks@</code> 677, 678, 685, 688, 691, 695, 697
<code>\pdfoutput</code>	940	<code>\tt</code>
<code>\pdfstringdef</code>	43, 295, 298, 662	899
<code>\ProvidesPackage</code>	102, 125, 171	U
R		<code>\usepackage</code>
<code>\RangeCatcodeInvalid</code>
.....	857, 869, 870, 871, 872	W
<code>\repeat</code>	832, 844, 855, 863	<code>\write</code>
<code>\RequirePackage</code>	104, 215, 919
<code>\resetatcatcode</code>	8	X
		<code>\x</code>
	
		120, 121,
		124, 129, 133, 135, 155, 160,
		170, 176, 184, 676, 683, 741, 742