

The embedfile package

Heiko Oberdiek*

<heiko.oberdiek at gmail.com>

2018/11/01 v2.8

Abstract

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

Contents

1	Documentation	2
1.1	Introduction	2
1.1.1	Future development	2
1.2	User interface	3
1.3	Collection support (PDF 1.7)	4
1.4	Export of object references	5
1.4.1	Example	6
1.5	Examples	6
1.5.1	plain TeX	6
1.5.2	Collection example	6
1.6	Package dtx-attach	8
2	Implementation	8
2.1	Reload check and package identification	8
2.2	Catcodes	9
2.3	Tools	10
2.4	Check for recent pdfTeX in PDF mode	10
2.5	Strings	11
2.6	Switches	12
2.7	Key value definitions	12
2.8	Embed the file	17
3	Test	22
3.1	Catcode checks for loading	22
3.2	Simple test	24
3.3	Test for ini-TeX	24
4	Installation	25
4.1	Download	25
4.2	Bundle installation	25
4.3	Package installation	25
4.4	Refresh file name databases	26
4.5	Some details for the interested	26
5	Catalogue	26
6	References	27

*Please report any issues at <https://github.com/ho-tex/oberdiek/issues>

7 History	27
[2006/08/16 v1.0]	27
[2007/04/11 v1.1]	27
[2007/09/09 v1.2]	27
[2007/10/28 v2.0]	27
[2007/10/29 v2.1]	27
[2007/11/11 v2.2]	27
[2007/11/25 v2.3]	28
[2009/09/25 v2.4]	28
[2010/03/01 v2.5]	28
[2011/04/13 v2.6]	28
[2016/05/15 v2.7]	28
[2018/11/01 v2.8]	28
8 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 \LaTeX and plain \TeX . See [subsubsection 1.5.1](#) that explains the use with plain \TeX by an example. In \LaTeX the package is loaded as usually. There are no options.

```
\usepackage{embedfile}
```

`\embedfile [\langle options \rangle] {\langle file \rangle}`

The macro `\embedfile` includes file *\langle file \rangle* 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 *\langle options \rangle* 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 `string-method`.

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 1.5.1](#) 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 `pdf\TeX's \pdfescapestring` is used. Value `psd` forces the use of `\pdfstringdef`, value `escape` the use of `\pdfescapestring`.

\langle key \rangle.value Sets the value of a collection item property, see [section 1.3](#).

\langle key \rangle.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
...

Acrobat Reader 10 shows the embedded files in the left panel and adds a new column for the compressed 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 {⟨options⟩}`

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 {⟨key⟩} {⟨options⟩}`

Macro `\embedfilefield` defines a column/field in the collection table. The name of the field is `⟨key⟩`.

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

text A text field. Its value is set in `\embedfile` by option `⟨key⟩.value`.

date A date field. Its value is set in `\embedfile` by option `⟨key⟩.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 `⟨key⟩.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 `<key>.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}[2018/11/01]
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/author/marcus/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 [2018/11/01 v2.8 Embed \string\jobname.dtx (HO)]%
104 \RequirePackage{embedfile}[2018/11/01]
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\catcode61\catcode48\catcode32=10\relax%
113 \catcode13=5 % ^^M
114 \endlinechar=13 %
115 \catcode35=6 % #
116 \catcode39=12 % '
117 \catcode44=12 % ,
118 \catcode45=12 % -
119 \catcode46=12 % .
120 \catcode58=12 % :
121 \catcode64=11 % @
122 \catcode123=1 % {
123 \catcode125=2 % }
124 \expandafter\let\expandafter\x\csname ver@embedfile.sty\endcsname
125 \ifx\x\relax % plain-TeX, first loading
126 \else
127 \def\empty{}%
128 \ifx\x\empty % LaTeX, first loading,
129 % variable is initialized, but \ProvidesPackage not yet seen
130 \else
131 \expandafter\ifx\csname PackageInfo\endcsname\relax
132 \def\x#1#2{%
133 \immediate\write-1{Package #1 Info: #2.}%
134 }%
135 \else
136 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
137 \fi
138 \x{embedfile}{The package is already loaded}%
139 \aftergroup\endinput
140 \fi
141 \fi
142 \endgroup%
```

Package identification:

```
143 \begingroup\catcode61\catcode48\catcode32=10\relax%
144 \catcode13=5 % ^^M
145 \endlinechar=13 %
146 \catcode35=6 % #
147 \catcode39=12 % '
```



```

148 \catcode40=12 % (
149 \catcode41=12 % )
150 \catcode44=12 % ,
151 \catcode45=12 % -
152 \catcode46=12 % .
153 \catcode47=12 % /
154 \catcode58=12 % :
155 \catcode64=11 % @
156 \catcode91=12 % [
157 \catcode93=12 % ]
158 \catcode123=1 % {
159 \catcode125=2 % }
160 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
161 \def\x#1#2#3[#4]{\endgroup
162 \immediate\write-1{Package: #3 #4}%
163 \xdef#1{#4}%
164 }%
165 \else
166 \def\x#1#2[#3]{\endgroup
167 #2[{#3}]%
168 \ifx#1\@undefined
169 \xdef#1{#3}%
170 \fi
171 \ifx#1\relax
172 \xdef#1{#3}%
173 \fi
174 }%
175 \fi
176 \expandafter\x\csname ver@embedfile.sty\endcsname
177 \ProvidesPackage{embedfile}%
178 [2018/11/01 v2.8 Embed files into PDF (HO)]%

```

2.2 Catcodes

```

179 \begingroup\catcode61\catcode48\catcode32=10\relax%
180 \catcode13=5 % ^^M
181 \endlinechar=13 %
182 \catcode123=1 % {
183 \catcode125=2 % }
184 \catcode64=11 % @
185 \def\x{\endgroup
186 \expandafter\edef\csname EmFi@AtEnd\endcsname{%
187 \endlinechar=\the\endlinechar\relax
188 \catcode13=\the\catcode13\relax
189 \catcode32=\the\catcode32\relax
190 \catcode35=\the\catcode35\relax
191 \catcode61=\the\catcode61\relax
192 \catcode64=\the\catcode64\relax
193 \catcode123=\the\catcode123\relax
194 \catcode125=\the\catcode125\relax
195 }%
196 }%
197 \x\catcode61\catcode48\catcode32=10\relax%
198 \catcode13=5 % ^^M
199 \endlinechar=13 %
200 \catcode35=6 % #
201 \catcode64=11 % @
202 \catcode123=1 % {
203 \catcode125=2 % }
204 \def\TMP@EnsureCode#1#2{%
205 \edef\EmFi@AtEnd{%
206 \EmFi@AtEnd

```

```

207 \catcode#1=\the\catcode#1\relax
208 }%
209 \catcode#1=#2\relax
210 }
211 \TMP@EnsureCode{39}{12}% '
212 \TMP@EnsureCode{40}{12}% (
213 \TMP@EnsureCode{41}{12}% )
214 \TMP@EnsureCode{44}{12}% ,
215 \TMP@EnsureCode{46}{12}% .
216 \TMP@EnsureCode{47}{12}% /
217 \TMP@EnsureCode{58}{12}% :
218 \TMP@EnsureCode{60}{12}% <
219 \TMP@EnsureCode{62}{12}% >
220 \TMP@EnsureCode{91}{12}% [
221 \TMP@EnsureCode{93}{12}% ]
222 \TMP@EnsureCode{96}{12}% `
223 \edef\EmFi@AtEnd{\EmFi@AtEnd\noexpand\endinput}

```

2.3 Tools

`\EmFi@RequirePackage`

```

224 \begingroup\expandafter\expandafter\expandafter\endgroup
225 \expandafter\ifx\csname RequirePackage\endcsname\relax
226 \def\EmFi@RequirePackage#1[#2]{%
227 \input #1.sty\relax
228 }%
229 \else
230 \let\EmFi@RequirePackage\RequirePackage
231 \fi

```

`\EmFi@Error`

```

232 \EmFi@RequirePackage{infwarerr}[2007/09/09]%
233 \def\EmFi@Error{%
234 \@PackageError{embedfile}%
235 }

```

Luatex compat

```

236 \ifx\pdfextension\@undefined\else
237 \protected\def\pdflastobj {\numexpr\pdffeedback lastobj\relax}
238 \protected\def\pdfnames {\pdfextension names }
239 \protected\def\pdfobj {\pdfextension obj }
240 \let\pdfoutput \outputmode
241 \fi

```

2.4 Check for recent pdfTeX in PDF mode

Load package ifpdf and check mode.

```

242 \EmFi@RequirePackage{ifpdf}[2007/09/09]
243 \ifpdf
244 \else
245 \EmFi@Error{%
246 Missing pdfTeX in PDF mode%
247 }{%
248 Currently other drivers are not supported. %
249 Package loading is aborted.%
250 }%
251 \expandafter\EmFi@AtEnd
252 \fi%

253 \EmFi@RequirePackage{pdftexcmds}[2007/11/11]
254 \EmFi@RequirePackage{ltxcmds}[2010/03/01]
255 \EmFi@RequirePackage{kvsetkeys}[2010/03/01]
256 \EmFi@RequirePackage{kvdefinekeys}[2010/03/01]

```

Check version.

```
257 \begingroup\expandafter\expandafter\expandafter\endgroup
258 \expandafter\ifx\csize pdf@filesize\endcsize\relax
259 \EmFi@Error{%
260   Unsupported pdfTeX version%
261 }{%
262   At least version 1.30 is necessary. Package loading is aborted.%
263 }%
264 \expandafter\EmFi@AtEnd
265 \fi%
```

2.5 Strings

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

```
266 \EmFi@RequirePackage{pdfescape}[2007/11/11]
267 \def\EmFi@temp#1{%
268   \expandafter\EdefSanitize\csize EmFi@S@#1\endcsize{#1}%
269 }

\EmFi@details
270 \EmFi@temp{details}%

\EmFi@tile
271 \EmFi@temp{tile}%

\EmFi@hidden
272 \EmFi@temp{hidden}%

\EmFi@S@text
273 \EmFi@temp{text}

\EmFi@S@date
274 \EmFi@temp{date}

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

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

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

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

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

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

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

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

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

\EmFi@S@false
284 \EmFi@temp{false}
```

2.6 Switches

```
\ifEmFi@collection
285 \ltx@newif\ifEmFi@collection

\ifEmFi@sort
286 \ltx@newif\ifEmFi@sort

\ifEmFi@visible
287 \ltx@newif\ifEmFi@visible

\ifEmFi@edit
288 \ltx@newif\ifEmFi@edit

\ifEmFi@item
289 \ltx@newif\ifEmFi@item

\ifEmFi@finished
290 \ltx@newif\ifEmFi@finished

\ifEmFi@id
291 \ltx@newif\ifEmFi@id
```

2.7 Key value definitions

```
\EmFi@GlobalKey
292 \def\EmFi@GlobalKey#1#2{%
293   \global\expandafter\let\csname KV@#1@#2\expandafter\endcsname
294   \csname KV@#1@#2\endcsname
295 }

\EmFi@GlobalDefaultKey
296 \def\EmFi@GlobalDefaultKey#1#2{%
297   \EmFi@GlobalKey{#1}{#2}%
298   \global\expandafter\let
299   \csname KV@#1@#2@default\expandafter\endcsname
300   \csname KV@#1@#2@default\endcsname
301 }

\EmFi@DefineKey
302 \def\EmFi@DefineKey#1#2{%
303   \kv@define@key{EmFi}{#1}{%
304     \expandafter\def\csname EmFi@#1\endcsname{##1}%
305   }%
306   \expandafter\def\csname EmFi@#1\endcsname{#2}%
307 }
```

Subtype of the embedded file (optional).

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

File specification string.

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

File specification string in Unicode.

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

File system (optional).

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

Description (optional).

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

Method for converting text to PDF strings.

```
313 \EmFi@DefineKey{stringmethod}{%
314   \ifx\pdfstringdef\@undefined
315     escape%
316   \else
317     \ifx\pdfstringdef\relax
318       escape%
319     \else
320       psd%
321     \fi
322   \fi
323 }
```

Option id as key for object numbers.

```
324 \kv@define@key{EmFi}{id}{%
325   \def\EmFi@id{#1}%
326   \EmFi@idtrue
327 }
```

\EmFi@defobj

```
328 \def\EmFi@defobj#1{%
329   \ifEmFi@id
330     \expandafter\xdef\csname EmFi@#1@\EmFi@id\endcsname{%
331       \the\pdflastobj\ltx@space 0 R%
332     }%
333   \fi
334 }
```

\embedfileifobjectexists

```
335 \def\embedfileifobjectexists#1#2{%
336   \expandafter\ifx\csname EmFi@#2@#1\endcsname\relax
337     \expandafter\ltx@secondoftwo
338   \else
339     \expandafter\ltx@firstoftwo
340   \fi
341 }
```

\embedfilegetobject

```
342 \def\embedfilegetobject#1#2{%
343   \embedfileifobjectexists{#1}{#2}{%
344     \csname EmFi@#2@#1\endcsname
345   }{%
346     0 0 R%
347   }%
348 }
```

Initial view of the collection.

```
349 \kv@define@key{EmFi}{view}[]{%
350   \EdefSanitize\EmFi@temp{#1}%
351   \def\EmFi@next{%
352     \global\EmFi@collectiontrue
353   }%
354   \ifx\EmFi@temp\ltx@empty
355     \let\EmFi@view\EmFi@S@details
356   \else\ifx\EmFi@temp\EmFi@S@details
357     \let\EmFi@view\EmFi@S@details
358   \else\ifx\EmFi@temp\EmFi@S@tile
359     \let\EmFi@view\EmFi@S@tile
360   \else\ifx\EmFi@temp\EmFi@S@hidden
361     \let\EmFi@view\EmFi@S@hidden
362   \else
363     \let\EmFi@next\relax
```

```

364 \EmFi@Error{%
365     Unknown value `\'EmFi@temp' for key `view'.\MessageBreak
366     Supported values: `details', `tile', `hidden'.%
367 } \@ehc
368 \fi\fi\fi\fi
369 \EmFi@next
370 }
371 \EmFi@DefineKey{initialfile}{-}

\embedfilesetup
372 \def\embedfilesetup{%
373 \ifEmFi@finished
374 \def\EmFi@next##1{-%
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 \def\EmFi@next{%
382 \kvsetkeys{EmFi}%
383 }%
384 \fi
385 \EmFi@next
386 }

\EmFi@schema
387 \def\EmFi@schema{-}

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

\EmFi@@order
389 \let\EmFi@@order\relax

\EmFi@fieldlist
390 \def\EmFi@fieldlist{-}

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

\embedfilefield
392 \def\embedfilefield#1#2{%
393 \ifEmFi@finished
394 \EmFi@Error{%
395 \string\embedfilefield\ltx@space after \string\embedfilefinish
396 }{-%
397     The list of embedded files is already written.%
398 }%
399 \else
400 \global\EmFi@collectiontrue
401 \edefSanitize\EmFi@key{#1}%
402 \expandafter\ifx\csname KV@EmFi@\EmFi@key.prefix\endcsname\relax
403 \begingroup
404 \count@=\EmFi@order
405 \advance\count@ 1 %
406 \xdef\EmFi@order{\the\count@}%
407 \let\EmFi@title\EmFi@key
408 \let\EmFi@type\EmFi@S@text
409 \EmFi@visibletrue
410 \EmFi@editfalse

```

```

411 \kvsetkeys{EmFiFi}{#2}%
412 \EmFi@convert\EmFi@title\EmFi@title
413 \xdef\EmFi@schema{%
414   \EmFi@schema
415   /\pdf@escapename{\EmFi@key}<<%
416   /Subtype/%
417   \ifx\EmFi@type\EmFi@S@date D%
418   \else\ifx\EmFi@type\EmFi@S@number N%
419   \else\ifx\EmFi@type\EmFi@S@file F%
420   \else\ifx\EmFi@type\EmFi@S@desc Desc%
421   \else\ifx\EmFi@type\EmFi@S@moddate ModDate%
422   \else\ifx\EmFi@type\EmFi@S@creationdate CreationDate%
423   \else\ifx\EmFi@type\EmFi@S@size Size%
424   \else S%
425   \fi\fi\fi\fi\fi\fi
426   /N(\EmFi@title)%
427   \EmFi@@order{\EmFi@order}%
428   \ifEmFi@visible
429   \else
430     /V false%
431   \fi
432   \ifEmFi@edit
433     /E true%
434   \fi
435   >>%
436 }%
437 \let\do\relax
438 \xdef\EmFi@fieldlist{%
439   \EmFi@fieldlist
440   \do{\EmFi@key}%
441 }%
442 \ifx\EmFi@type\EmFi@S@text
443   \kv@define@key{EmFi}{\EmFi@key.value}{%
444     \EmFi@itemtrue
445     \def\EmFi@temp{##1}%
446     \EmFi@convert\EmFi@temp\EmFi@temp
447     \expandafter\def\csname EmFi@V@#1%
448     \expandafter\endcsname\expandafter{%
449       \expandafter(\EmFi@temp)%
450     }%
451   }%
452   \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
453 \else\ifx\EmFi@type\EmFi@S@date
454   \kv@define@key{EmFi}{\EmFi@key.value}{%
455     \EmFi@itemtrue
456     \def\EmFi@temp{##1}%
457     \EmFi@convert\EmFi@temp\EmFi@temp
458     \expandafter\def\csname EmFi@V@#1%
459     \expandafter\endcsname\expandafter{%
460       \expandafter(\EmFi@temp)%
461     }%
462   }%
463   \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
464 \else\ifx\EmFi@type\EmFi@S@number
465   \kv@define@key{EmFi}{\EmFi@key.value}{%
466     \EmFi@itemtrue
467     \expandafter\edef\csname EmFi@V@#1\endcsname{ ##1}%
468   }%
469   \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
470 \fi\fi\fi
471 \kv@define@key{EmFi}{\EmFi@key.prefix}{%
472   \EmFi@itemtrue

```

```

473     \expandafter\def\csname EmFi@P@#1\endcsname{##1}%
474 }%
475 \EmFi@GlobalKey{EmFi}{\EmFi@key.prefix}%
476 \kv@define@key{EmFiSo}{\EmFi@key}[ascending]{%
477     \EdefSanitize\EmFi@temp{##1}%
478     \ifx\EmFi@temp\EmFi@S@ascending
479         \def\EmFi@temp{true}%
480     \else\ifx\EmFi@temp\EmFi@S@descending
481         \def\EmFi@temp{false}%
482     \else
483         \def\EmFi@temp{}%
484         \EmFi@Error{%
485             Unknown sort order ``\EmFi@temp'.\MessageBreak
486             Supported values: ``\EmFi@S@ascending', %
487             ``\EmFi@S@descending
488         }\@ehc
489     \fi\fi
490     \ifx\EmFi@temp\ltx@empty
491     \else
492         \xdef\EmFi@sortkeys{%
493             \EmFi@sortkeys
494             /\pdf@escapename{#1}%
495         }%
496         \ifx\EmFi@sortorders\ltx@empty
497             \global\let\EmFi@sortorders\EmFi@temp
498             \gdef\EmFi@sortcase{1}%
499         \else
500             \xdef\EmFi@sortorders{%
501                 \EmFi@sortorders
502                 \ltx@space
503                 \EmFi@temp
504             }%
505             \xdef\EmFi@sortcase{2}%
506         \fi
507     \fi
508 }%
509 \EmFi@GlobalDefaultKey{EmFiSo}\EmFi@key
510 \endgroup
511 \else
512     \EmFi@Error{%
513         Field ``\EmFi@key' is already defined%
514     }\@ehc
515 \fi
516 \fi
517 }

518 \kv@define@key{EmFiFi}{type}{%
519     \EdefSanitize\EmFi@temp{##1}%
520     \ifx\EmFi@temp\EmFi@S@text
521         \let\EmFi@type\EmFi@temp
522     \else\ifx\EmFi@temp\EmFi@S@date
523         \let\EmFi@type\EmFi@temp
524     \else\ifx\EmFi@temp\EmFi@S@number
525         \let\EmFi@type\EmFi@temp
526     \else\ifx\EmFi@temp\EmFi@S@file
527         \let\EmFi@type\EmFi@temp
528     \else\ifx\EmFi@temp\EmFi@S@desc
529         \let\EmFi@type\EmFi@temp
530     \else\ifx\EmFi@temp\EmFi@S@moddate
531         \let\EmFi@type\EmFi@temp
532     \else\ifx\EmFi@temp\EmFi@S@creationdate
533         \let\EmFi@type\EmFi@temp
534     \else\ifx\EmFi@temp\EmFi@S@size

```



```

535 \let\EmFi@type\EmFi@temp
536 \else
537 \EmFi@Error{%
538   Unknown type `\'EmFi@temp'.\MessageBreak
539   Supported types: `text', `date', `number', `file',\MessageBreak
540   `desc', `moddate', `creationdate', `size'%
541 }%
542 \fi\fi\fi\fi\fi\fi\fi
543 }

544 \kv@define@key{EmFiFi}{title}{%
545 \def\EmFi@title{#1}%
546 }

```

\EmFi@setboolean

```

547 \def\EmFi@setboolean#1#2{%
548 \edefSanitize\EmFi@temp{#2}%
549 \ifx\EmFi@temp\EmFi@S@true
550 \csname EmFi@#1true\endcsname
551 \else
552 \ifx\EmFi@temp\EmFi@S@false
553 \csname EmFi@#1false\endcsname
554 \else
555 \EmFi@Error{%
556   Unknown value `\'EmFi@temp' for key `#1'.\MessageBreak
557   Supported values: `true', `false'%
558 } \@ehc
559 \fi
560 \fi
561 }

562 \kv@define@key{EmFiFi}{visible}[true]{%
563 \EmFi@setboolean{visible}{#1}%
564 }

565 \kv@define@key{EmFiFi}{edit}[true]{%
566 \EmFi@setboolean{edit}{#1}%
567 }

```

\EmFi@sortkeys

```

568 \def\EmFi@sortkeys{}

```

\EmFi@sortorders

```

569 \def\EmFi@sortorders{}

```

\embedfilesort

```

570 \def\embedfilesort{%
571 \kvsetkeys{EmFiSo}%
572 }

```

2.8 Embed the file

\embedfile

```

573 \def\embedfile{%
574 \ltx@ifnextchar[\EmFi@embedfile{\EmFi@embedfile[]}%
575 }

```

\EmFi@embedfile

```

576 \def\EmFi@embedfile[#1]#2{%
577 \ifEmFi@finished
578 \EmFi@Error{%
579 \string\embedfile\ltx@space after \string\embedfilefinish
580 }{%

```

```

581     The list of embedded files is already written.%
582 }%
583 \else
584 \begingroup
585 \def\EmFi@file{#2}%
586 \kvsetkeys{EmFi}{#1}%
587 \expandafter\expandafter\expandafter
588 \ifx\expandafter\expandafter\expandafter
589   \pdf@filesize{\EmFi@file}\%
590   \EmFi@Error{%
591     File '\EmFi@file' not found%
592   }{%
593     The unknown file is not embedded.%
594   }%
595 \else
596 \edef\EmFi@@filespec{%
597   \pdf@escapestring{\EmFi@filespec}%
598 }%
599 \ifx\EmFi@ucfilespec\ltx@empty
600   \let\EmFi@@ucfilespec\ltx@empty
601 \else
602   \EmFi@convert\EmFi@ucfilespec\EmFi@@ucfilespec
603 \fi
604 \ifx\EmFi@desc\ltx@empty
605   \let\EmFi@@desc\ltx@empty
606 \else
607   \EmFi@convert\EmFi@desc\EmFi@@desc
608 \fi
609 \ifEmFi@item
610   \let\do\EmFi@do
611   \immediate\pdfobj{%
612     <<%
613       \EmFi@fieldlist
614     >>%
615   }%
616   \edef\EmFi@ci{\the\pdf@lastobj}%
617 \fi
618 \immediate\pdfobj stream attr{%
619   /Type/EmbeddedFile%
620   \ifx\EmFi@mimetype\ltx@empty
621   \else
622     /Subtype/\pdf@escapename{\EmFi@mimetype}%
623   \fi
624   /Params<<%
625     /ModDate(\pdf@filemoddate{\EmFi@file})%
626     /Size \pdf@filesize{\EmFi@file}%
627     /Checksum<\pdf@filemdfivesum{\EmFi@file}>%
628   >>%
629 }file{\EmFi@file}\relax
630 \EmFi@defobj{EmbeddedFile}%
631 \immediate\pdfobj{%
632   <<%
633     /Type/Filespec%
634     \ifx\EmFi@filesystem\ltx@empty
635     \else
636       /FS/\pdf@escapename{\EmFi@filesystem}%
637     \fi
638     /F(\EmFi@@filespec)%
639     \ifx\EmFi@@ucfilespec\ltx@empty
640     \else
641       /UF(\EmFi@@ucfilespec)%
642     \fi

```

```

643     \ifx\EmFi@@desc\ltx@empty
644     \else
645     /Desc(\EmFi@@desc)%
646     \fi
647     /EF<<%
648     /F \the\pdfastobj\ltx@space 0 R%
649     >>%
650     \ifEmFi@item
651     /CI \EmFi@ci\ltx@space 0 R%
652     \fi
653     >>%
654     }%
655     \EmFi@defobj{Filespec}%
656     \EmFi@add{%
657     \EmFi@@filespec
658     }{\the\pdfastobj\ltx@space 0 R}%
659     \fi
660     \endgroup
661     \fi
662 }

```

\EmFi@do

```

663 \def\EmFi@do#1{%
664   \expandafter\ifx\csname EmFi@P@#1\endcsname\relax
665   \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
666   \else
667   /\pdf@escapename{#1}\csname EmFi@V@#1\endcsname
668   \fi
669   \else
670   /\pdf@escapename{#1}<<%
671   \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
672   \else
673   /D\csname EmFi@V@#1\endcsname
674   \fi
675   /P(\csname EmFi@P@#1\endcsname)%
676   >>%
677   \fi
678 }

```

\EmFi@convert

```

679 \def\EmFi@convert#1#2{%
680   \ifnum\pdf@strcmp{\EmFi@stringmethod}{psd}=0 %
681   \pdfstringdef\EmFi@temp{#1}%
682   \let#2\EmFi@temp
683   \else
684   \edef#2{\pdf@escapestring{#1}}%
685   \fi
686 }

687 \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, ...).

```

688 \def\EmFi@add#1#2{%
689   \begingroup
690   \ifx\EmFi@list\ltx@empty
691   \xdef\EmFi@list{\noexpand\do{#1}{#2}}%
692   \else
693   \def\do##1##2{%
694     \ifnum\pdf@strcmp{##1}{#1}>0 %
695     \edef\x{%

```

```

696     \toks@{%
697     \the\toks@%
698     \noexpand\do{##1}{##2}%
699     \noexpand\do{###1}{###2}%
700     }%
701 }%
702 \x
703 \def\do###1###2{%
704     \toks@\expandafter{\the\toks@\do{###1}{###2}}%
705 }%
706 \def\stop{%
707     \xdef\EmFi@list{\the\toks@}%
708 }%
709 \else
710     \toks@\expandafter{\the\toks@\do{##1}{##2}}%
711 \fi
712 }%
713 \def\stop{%
714     \xdef\EmFi@list{\the\toks@\noexpand\do{##1}{##2}}%
715 }%
716 \toks@{}%
717 \EmFi@list\stop
718 \fi
719 \endgroup
720 }

```

\embedfilefinish

```

721 \def\embedfilefinish{%
722     \ifEmFi@finished
723     \EmFi@Error{%
724         Too many invocations of \string\embedfilefinish
725     }{%
726         The list of embedded files is already written.%
727     }%
728 \else
729     \ifx\EmFi@list\ltx@empty
730     \else

```

Write /EmbeddedFiles entry.

```

731     \global\EmFi@finishedtrue
732     \begingroup
733     \def\do##1##2{%
734         (##1)##2%
735     }%
736     \immediate\pdfobj{%
737         <<%
738         /Names[\EmFi@list]%
739         >>%
740     }%
741     \pdfnames{%
742         /EmbeddedFiles \the\pdflastobj\ltx@space 0 R%
743     }%
744     \endgroup

```

Write collection objects.

```

745     \ifx\EmFi@initialfile\ltx@empty
746     \else
747     \EmFi@collectiontrue
748     \fi
749     \ifEmFi@collection
750     \ifx\EmFi@initialfile\ltx@empty
751     \let\EmFi@initialfile\ltx@empty
752     \else

```

```

753 \edef\EmFi@@initialfile{%
754 \pdf@escapestring{\EmFi@initialfile}%
755 }%
756 \fi

```

Look for initial file among the embedded files.

```

757 \begingroup
758 \let\f=N%
759 \def\do##1##2{%
760 \def\x{##1}%
761 \ifx\x\EmFi@@initialfile
762 \let\f=Y%
763 \let\do\ltx@gobbletwo
764 \fi
765 }%
766 \EmFi@list
767 \expandafter\endgroup
768 \ifx\f Y%
769 \else
770 \@PackageWarningNoLine{embedfile}{%
771 Missing initial file ` \EmFi@initialfile'\MessageBreak
772 among the embedded files%
773 }%
774 \let\EmFi@initialfile\ltx@empty
775 \let\EmFi@@initialfile\ltx@empty
776 \fi
777 \ifcase\EmFi@sortcase
778 \def\EmFi@temp{}%
779 \or
780 \def\EmFi@temp{%
781 /S\EmFi@sortkeys
782 /A \EmFi@sortorders
783 }%
784 \else
785 \def\EmFi@temp{%
786 /S[\EmFi@sortkeys]%
787 /A[\EmFi@sortorders]%
788 }%
789 \fi
790 \def\EmFi@@order##1{%
791 \ifnum\EmFi@order>1 %
792 /O ##1%
793 \fi
794 }%
795 \immediate\pdfobj{%
796 <<%
797 \ifx\EmFi@schema\ltx@empty
798 \else
799 /Schema<<\EmFi@schema>>%
800 \fi
801 \ifx\EmFi@@initialfile\ltx@empty
802 \else
803 /D(\EmFi@@initialfile)%
804 \fi
805 \ifx\EmFi@view\EmFi@S@tile
806 /View/T%
807 \else\ifx\EmFi@view\EmFi@S@hidden
808 /View/H%
809 \fi\fi
810 \ifx\EmFi@temp\ltx@empty
811 \EmFi@temp
812 \else
813 /Sort<<\EmFi@temp>>%

```

```

814     \fi
815     >>%
816   }%
817   \pdfcatalog{%
818     /Collection \the\pdflastobj\ltx@space0 R%
819   }%
820   \fi
821   \fi
822   \fi
823 }

824 \begingroup\expandafter\expandafter\expandafter\endgroup
825 \expandafter\ifx\csname AtEndDocument\endcsname\relax
826 \else
827   \AtEndDocument{\embedfilefinish}%
828 \fi

829 \EmFi@AtEnd%
830 \endpackage

```

3 Test

3.1 Catcode checks for loading

```

831 \test1

832 \catcode`\{=1 %
833 \catcode`\}=2 %
834 \catcode`\#=6 %
835 \catcode`\@=11 %
836 \expandafter\ifx\csname count@\endcsname\relax
837   \countdef\count@=255 %
838 \fi
839 \expandafter\ifx\csname @gobble\endcsname\relax
840   \long\def\@gobble#1{%
841     \fi
842   \expandafter\ifx\csname @firstofone\endcsname\relax
843     \long\def\@firstofone#1{#1}%
844   \fi
845   \expandafter\ifx\csname loop\endcsname\relax
846     \expandafter\@firstofone
847   \else
848     \expandafter\@gobble
849   \fi
850 {%
851   \def\loop#1\repeat{%
852     \def\body{#1}%
853     \iterate
854   }%
855   \def\iterate{%
856     \body
857     \let\next\iterate
858   \else
859     \let\next\relax
860   \fi
861   \next
862 }%
863 \let\repeat=\fi
864 }%
865 \def\RestoreCatcodes{
866 \count@=0 %
867 \loop
868 \edef\RestoreCatcodes{%

```

```

869 \RestoreCatcodes
870 \catcode\the\count@=\the\catcode\count@\relax
871 }%
872 \ifnum\count@<255 %
873 \advance\count@ 1 %
874 \repeat
875
876 \def\RangeCatcodeInvalid#1#2{%
877 \count@=#1\relax
878 \loop
879 \catcode\count@=15 %
880 \ifnum\count@<#2\relax
881 \advance\count@ 1 %
882 \repeat
883 }
884 \def\RangeCatcodeCheck#1#2#3{%
885 \count@=#1\relax
886 \loop
887 \ifnum#3=\catcode\count@
888 \else
889 \errmessage{%
890 Character \the\count@\space
891 with wrong catcode \the\catcode\count@\space
892 instead of \number#3%
893 }%
894 \fi
895 \ifnum\count@<#2\relax
896 \advance\count@ 1 %
897 \repeat
898 }
899 \def\space{ }
900 \expandafter\ifx\csname LoadCommand\endcsname\relax
901 \def\LoadCommand{\input embedfile.sty\relax}%
902 \fi
903 \def\Test{%
904 \RangeCatcodeInvalid{0}{47}%
905 \RangeCatcodeInvalid{58}{64}%
906 \RangeCatcodeInvalid{91}{96}%
907 \RangeCatcodeInvalid{123}{255}%
908 \catcode`\@=12 %
909 \catcode`\=0 %
910 \catcode`\%=14 %
911 \LoadCommand
912 \RangeCatcodeCheck{0}{36}{15}%
913 \RangeCatcodeCheck{37}{37}{14}%
914 \RangeCatcodeCheck{38}{47}{15}%
915 \RangeCatcodeCheck{48}{57}{12}%
916 \RangeCatcodeCheck{58}{63}{15}%
917 \RangeCatcodeCheck{64}{64}{12}%
918 \RangeCatcodeCheck{65}{90}{11}%
919 \RangeCatcodeCheck{91}{91}{15}%
920 \RangeCatcodeCheck{92}{92}{0}%
921 \RangeCatcodeCheck{93}{96}{15}%
922 \RangeCatcodeCheck{97}{122}{11}%
923 \RangeCatcodeCheck{123}{255}{15}%
924 \RestoreCatcodes
925 }
926 \Test
927 \csname @@end\endcsname
928 \end
929 </test1>

```

3.2 Simple test

```
930 <*test2>
931 \input embedfile.sty\relax
932 \embedfile[%
933   stringmethod=escape,%
934   mimetype=plain/text,%
935   desc={LaTeX docstrip source archive for package `embedfile'},%
936   id={embedfile.dtx}%
937 ]{embedfile.dtx}
938 \nopagenumbers
939 Test (plain-TeX): {\tt embedfile.dtx} should be embedded.%
940
941 \def\Test#1{%
942   \par
943   \embedfileifobjectexists{embedfile.dtx}{#1}{%
944     Object #1 (embedfile.dtx): %
945     \embedfilegetobject{embedfile.dtx}{#1}%
946   }{%
947     \errmessage{Missing object #1 (embedfile.dtx)}%
948   }%
949 }
950 \Test{EmbeddedFile}
951 \Test{Filespec}
952 \embedfilefinish
953 \bye
954 </test2>

955 <*test3>
956 \NeedsTeXFormat{LaTeX2e}
957 \let\SavedJobname\jobname
958 \def\jobname{embedfile}
959 \RequirePackage{dtx-attach}[2018/11/01]
960 \let\jobname\SavedJobname
961 \documentclass{minimal}
962 \begin{document}
963   Test (\LaTeX): \texttt{embedfile.dtx} should be embedded.%
964 \end{document}
965 </test3>
```

3.3 Test for ini-TeX

```
966 <*test4>
967 \catcode`\{=1 %
968 \catcode`\}=2 %
969 \input ifuatex.sty %
970 \ifuatex
971   \directlua{%
972     tex.enableprimitives('',{%
973       'pdflastobj',%
974       'pdfnames',%
975       'pdfobj',%
976       'pdfoutput'%
977     })%
978   }%
979 \ifx\pdfextension\undefined\else
980   \protected\def\pdflastobj {\numexpr\pdffeedback lastobj\relax}
981   \protected\def\pdfnames   {\pdfextension names }
982   \protected\def\pdfobj     {\pdfextension obj }
983   \let\pdfoutput           \outputmode
984 \fi
985 \fi
986 \pdfoutput=1 %
987 \input embedfile.sty %
```



```

988 \shipout\hbox{}
989 \embedfile[%
990   stringmethod=escape,%
991   mimetype=plain/text,%
992   desc={iniTeX source},%
993 ]{\jobname.tex}
994 \embedfilefinish
995 \end
996 </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
```

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

¹<http://ctan.org/pkg/embedfile>

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

4.4 Refresh file name databases

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

4.5 Some details for the interested

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 Catalogue

The following XML file can be used as source for the [\$\text{\TeX}\$ Catalogue](#). The elements `caption` and `description` are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is `embedfile.xml`.

```

997 <*catalogue>
998 <?xml version='1.0' encoding='us-ascii'?>
999 <!DOCTYPE entry SYSTEM 'catalogue.dtd'>
1000 <entry datestamp='$Date$' modifier='$Author$' id='embedfile'>
1001   <name>embedfile</name>
1002   <caption>Embed files into PDF.</caption>
1003   <authorref id='auth:oberdiek' />
1004   <copyright owner='Heiko Oberdiek' year='2006-2011' />
1005   <license type='lppl1.3' />
1006   <version number='2.8' />
1007   <description>
1008     This package embeds files in a PDF document, using the PDF
1009     format's embedding operation (note the contrast with the attach
1010     operation used by the <xref refid='attachfile'>attachfile</xref>
1011     and <xref refid='attachfile2'>attachfile2</xref> packages).
1012     Currently only <xref refid='pdftex'>pdfTeX</xref> &gt;=1.30, in
1013     PDF mode, is supported.
1014
```

```

1015 <p/>
1016 The package is part of the <xref refid='oberdiek'>oberdiek</xref>
1017 bundle.
1018 </description>
1019 <documentation details='Package documentation'
1020 href='ctan:/macros/latex/contrib/oberdiek/embedfile.pdf' />
1021 <ctan file='true' path='/macros/latex/contrib/oberdiek/embedfile.dtx' />
1022 <miktex location='oberdiek' />
1023 <texlive location='oberdiek' />
1024 <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip' />
1025 </entry>
1026 </catalogue>

```

6 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/>.

7 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/embedfile.sty`.

[2011/04/13 v2.6]

- Docu fixes (thanks Hans-Martin Münch).

[2016/05/15 v2.7]

- LuaTeX compatibility

[2018/11/01 v2.8]

- Remove `luatex85` package dependency.

8 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		<code>\AtEndDocument</code> 827
<code>\#</code>	834	
<code>\%</code>	910	
<code>\@</code>	835, 908	
<code>\@PackageError</code>	234	
<code>\@PackageWarningNoLine</code>	770	
<code>\@ehc</code>	367, 488, 514, 558	
<code>\@firstofone</code>	843, 846	
<code>\@gobble</code>	840, 848	
<code>\@undefined</code>	168, 236, 314	
<code>\</code>	589, 909	
<code>\{</code>	832, 967	
<code>\}</code>	833, 968	
A		
<code>\advance</code>	405, 873, 881, 896	
<code>\aftergroup</code>	139	
B		
<code>\begin</code>	75, 962	
<code>\body</code>	852, 856	
<code>\bye</code>	35, 953	
C		
<code>\catcode</code>	112, 113, 115, 116, 117, 118, 119, 120, 121, 122, 123, 143, 144, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 179, 180, 182, 183, 184, 188, 189, 190, 191, 192, 193, 194, 197, 198, 200, 201, 202, 203, 207, 209, 832, 833, 834, 835, 870, 879, 887, 891, 908, 909, 910, 967, 968	

<code>\count@</code>	404, 405, 406, 837, 866, 870, 872, 873, 877, 879, 880, 881, 885, 887, 890, 891, 895, 896
<code>\countdef</code>	837
<code>\csname</code> 124, 131, 160, 176, 186, 225, 258, 268, 293, 294, 299, 300, 304, 306, 330, 336, 344, 402, 447, 458, 467, 473, 550, 553, 664, 665, 667, 671, 673, 675, 825, 836, 839, 842, 845, 900, 927	
D	
<code>\directlua</code>	971
<code>\do</code> 437, 440, 610, 691, 693, 698, 699, 703, 704, 710, 714, 733, 759, 763	
<code>\documentclass</code>	41, 961
E	
<code>\EdefSanitize</code>	268, 350, 401, 467, 477, 519, 548
<code>\embedfile</code>	3, 16, 21, 25, 79, 85, 91, 105, 573, 579, 932, 989
<code>\embedfilefield</code>	4, 50, 54, 58, 62, 66, 376, 392
<code>\embedfilefinish</code>	3, 34, 376, 395, 579, 721, 827, 952, 994
<code>\embedfilegetobject</code>	5, 342, 945
<code>\embedfileifobjectexists</code> 5, 335, 343, 943	
<code>\embedfilessetup</code>	4, 4, 11, 46, 372
<code>\embedfilesort</code>	5, 71, 570
<code>\EmFi@@desc</code>	605, 607, 643, 645
<code>\EmFi@@filespec</code>	596, 638, 657
<code>\EmFi@initialfile</code>	751, 753, 761, 775, 801, 803
<code>\EmFi@@order</code>	389, 427, 790
<code>\EmFi@@ucfilespec</code>	600, 602, 639, 641
<code>\EmFi@add</code>	656, 688
<code>\EmFi@AtEnd</code> 205, 206, 223, 251, 264, 829	
<code>\EmFi@ci</code>	616, 651
<code>\EmFi@collectiontrue</code>	352, 400, 747
<code>\EmFi@convert</code>	412, 446, 457, 602, 607, 679
<code>\EmFi@DefineKey</code>	302, 308, 309, 310, 311, 312, 313, 371
<code>\EmFi@defobj</code>	328, 630, 655
<code>\EmFi@desc</code>	604, 607
<code>\EmFi@details</code>	270
<code>\EmFi@do</code>	610, 663
<code>\EmFi@editfalse</code>	410
<code>\EmFi@embedfile</code>	574, 576
<code>\EmFi@Error</code>	232, 245, 259, 364, 375, 394, 484, 512, 537, 555, 578, 590, 723
<code>\EmFi@fieldlist</code>	390, 438, 439, 613
<code>\EmFi@file</code>	309, 585, 589, 591, 625, 626, 627, 629
<code>\EmFi@filespec</code>	597
<code>\EmFi@filesystem</code>	634, 636
<code>\EmFi@finishedtrue</code>	731
<code>\EmFi@GlobalDefaultKey</code>	296, 509
<code>\EmFi@GlobalKey</code>	292, 297, 452, 463, 469, 475
<code>\EmFi@hidden</code>	272
<code>\EmFi@id</code>	325, 330
<code>\EmFi@idtrue</code>	326
<code>\EmFi@initialfile</code> 745, 750, 754, 771, 774	
<code>\EmFi@itemtrue</code>	444, 455, 466, 472
<code>\EmFi@key</code>	401, 402, 407, 415, 440, 443, 452, 454, 463, 465, 469, 471, 475, 476, 509, 513
<code>\EmFi@list</code>	687, 690, 691, 707, 714, 717, 729, 738, 766
<code>\EmFi@mimetype</code>	620, 622
<code>\EmFi@next</code> 351, 363, 369, 374, 381, 385	
<code>\EmFi@order</code>	388, 404, 406, 427, 791
<code>\EmFi@RequirePackage</code>	224, 232, 242, 253, 254, 255, 256, 266
<code>\EmFi@S@ascending</code>	281, 478, 486
<code>\EmFi@S@creationdate</code>	279, 422, 532
<code>\EmFi@S@date</code>	274, 417, 453, 522
<code>\EmFi@S@desc</code>	277, 420, 528
<code>\EmFi@S@descending</code>	282, 480, 487
<code>\EmFi@S@details</code>	355, 356, 357
<code>\EmFi@S@false</code>	284, 552
<code>\EmFi@S@file</code>	276, 419, 526
<code>\EmFi@S@hidden</code>	360, 361, 807
<code>\EmFi@S@moddate</code>	278, 421, 530
<code>\EmFi@S@number</code>	275, 418, 464, 524
<code>\EmFi@S@size</code>	280, 423, 534
<code>\EmFi@S@text</code>	273, 408, 442, 520
<code>\EmFi@S@tile</code>	358, 359, 805
<code>\EmFi@S@true</code>	283, 549
<code>\EmFi@schema</code>	387, 413, 414, 797, 799
<code>\EmFi@setboolean</code>	547, 563, 566
<code>\EmFi@sortcase</code>	391, 498, 505, 777
<code>\EmFi@sortkeys</code> 492, 493, 568, 781, 786	
<code>\EmFi@sortorders</code>	496, 497, 500, 501, 569, 782, 787
<code>\EmFi@stringmethod</code>	680
<code>\EmFi@temp</code>	267, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 350, 354, 356, 358, 360, 365, 445, 446, 449, 456, 457, 460, 477, 478, 479, 480, 481, 483, 485, 490, 497, 503, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 538, 548, 549, 552, 556, 681, 682, 778, 780, 785, 810, 811, 813
<code>\EmFi@tile</code>	271
<code>\EmFi@title</code>	407, 412, 426, 545
<code>\EmFi@type</code>	408, 417, 418, 419, 420, 421, 422, 423, 442, 453, 464, 521, 523, 525, 527, 529, 531, 533, 535
<code>\EmFi@ucfilespec</code>	599, 602
<code>\EmFi@view</code> 355, 357, 359, 361, 805, 807	
<code>\EmFi@visibletrue</code>	409
<code>\empty</code>	127, 128
<code>\end</code>	97, 928, 964, 995
<code>\endcsname</code>	124, 131, 160, 176, 186, 225, 258, 268, 293, 294, 299, 300,

304, 306, 330, 336, 344, 402, 448, 459, 467, 473, 550, 553, 664, 665, 667, 671, 673, 675, 825, 836, 839, 842, 845, 900, 927	639, 643, 687, 690, 729, 745, 750, 751, 774, 775, 797, 801, 810
\endinput 139, 223	\ltx@firstoftwo 339
\endlinechar . . . 114, 145, 181, 187, 199	\ltx@gobbletwo 763
\errmessage 889, 947	\ltx@ifnextchar 574
F	\ltx@newif 285, 286, 287, 288, 289, 290, 291
\f 758, 762, 768	\ltx@secondoftwo 337
G	\ltx@space 331, 376, 395, 502, 579, 648, 651, 658, 742, 818
\gdef 388, 498	M
\Gin@driver 5	\MessageBreak 365, 485, 538, 539, 556, 771
H	N
\hbox 988	\NeedsTeXFormat 40, 101, 956
I	\next 857, 859, 861
\ifcase 777	\nopagenumbers 938
\ifEmFi@collection 285, 749	\number 892
\ifEmFi@edit 288, 432	\numexpr 237, 980
\ifEmFi@finished 290, 373, 393, 577, 722	O
\ifEmFi@id 291, 329	\outputmode 240, 983
\ifEmFi@item 289, 609, 650	P
\ifEmFi@sort 286	\PackageInfo 136
\ifEmFi@visible 287, 428	\par 942
\ifluatex 970	\pdf@escapename 415, 494, 622, 636, 667, 670
\ifnum . 680, 694, 791, 872, 880, 887, 895	\pdf@escapestring 597, 684, 754
\ifpdf 243	\pdf@filemdfivesum 627
\ifx 125, 128, 131, 160, 168, 171, 225, 236, 258, 314, 317, 336, 354, 356, 358, 360, 402, 417, 418, 419, 420, 421, 422, 423, 442, 453, 464, 478, 480, 490, 496, 520, 522, 524, 526, 528, 530, 532, 534, 549, 552, 588, 599, 604, 620, 634, 639, 643, 664, 665, 671, 690, 729, 745, 750, 761, 768, 797, 801, 805, 807, 810, 825, 836, 839, 842, 845, 900, 979	\pdf@filemoddate 625
\immediate 133, 162, 611, 618, 631, 736, 795	\pdf@filesize 589, 626
\input . . 4, 6, 7, 227, 901, 931, 969, 987	\pdf@strcmp 680, 694
\iterate 853, 855, 857	\pdfcatalog 817
J	\pdfextension 236, 238, 239, 979, 981, 982
\jobname 95, 103, 108, 109, 957, 958, 960, 993	\pdffeedback 237, 980
K	\pdflastobj 237, 331, 616, 648, 658, 742, 818, 980
\kv@define@key 303, 324, 349, 443, 454, 465, 471, 476, 518, 544, 562, 565	\pdfnames 238, 741, 981
\kvsetkeys 382, 411, 571, 586	\pdfobj 239, 611, 618, 631, 736, 795, 982
L	\pdfoutput 240, 983, 986
\LaTeX 963	\pdfstringdef 43, 314, 317, 681
\LoadCommand 901, 911	\protected . 237, 238, 239, 980, 981, 982
\loop 851, 867, 878, 886	\ProvidesPackage 102, 129, 177
\ltx@empty 354, 490, 496, 599, 600, 604, 605, 620, 634,	R
	\RangeCatcodeCheck 884, 912, 913, 914, 915, 916, 917, 918, 919, 920, 921, 922, 923
	\RangeCatcodeInvalid 876, 904, 905, 906, 907
	\repeat 851, 863, 874, 882, 897
	\RequirePackage 104, 230, 959
	\resetatcatcode 8
	\RestoreCatcodes . . 865, 868, 869, 924
	S
	\SavedJobname 957, 960
	\shipout 988
	\space 890, 891, 899
	\stop 706, 713, 717

T	\tt	939
\Test		903, 926, 941, 950, 951
\TeX	U	939
\texttt	\undefined	979
\the	\usepackage	42, 45
187, 188, 189, 190, 191,	W	
192, 193, 194, 207, 331, 406,	\write	133, 162
616, 648, 658, 697, 704, 707,		
710, 714, 742, 818, 870, 890, 891	X	
\TMP@EnsureCode	\x	124, 125,
. 204, 211, 212, 213, 214, 215,		128, 132, 136, 138, 161, 166,
216, 217, 218, 219, 220, 221, 222		176, 185, 197, 695, 702, 760, 761
\toks@ . 696, 697, 704, 707, 710, 714, 716		