

The embedfile package

Heiko Oberdiek*

<heiko.oberdiek at gmail.com>

2016/05/15 v2.7

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
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 (pdf_{tex}, 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 pdf_{TeX} 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 pdf_{TeX}.

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 \pdfTeX '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}[2016/05/15]
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/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 [2016/05/15 v2.7 Embed \string\jobname.dtx (HO)]%
104 \RequirePackage{embedfile}[2016/05/15]
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 [2016/05/15 v2.7 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 \EmFi@RequirePackage{luatex85}[2016/01/01]
238 \fi

```

2.4 Check for recent pdfTeX in PDF mode

Load package ifpdf and check mode.

```

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

250 \EmFi@RequirePackage{pdftexcmds}[2007/11/11]
251 \EmFi@RequirePackage{ltxcmds}[2010/03/01]
252 \EmFi@RequirePackage{kvsetkeys}[2010/03/01]
253 \EmFi@RequirePackage{kvdefinekeys}[2010/03/01]

```

Check version.

```

254 \begingroup\expandafter\expandafter\expandafter\endgroup

```

```

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

```

2.5 Strings

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

```

263 \EmFi@RequirePackage{pdfescape}[2007/11/11]

264 \def\EmFi@temp#1{%
265   \expandafter\EdefSanitize\csize EmFi@S@#1\endcsize{#1}%
266 }

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```

2.6 Switches

```
\ifEmFi@collection
282 \ltx@newif\ifEmFi@collection

\ifEmFi@sort
283 \ltx@newif\ifEmFi@sort

\ifEmFi@visible
284 \ltx@newif\ifEmFi@visible

\ifEmFi@edit
285 \ltx@newif\ifEmFi@edit

\ifEmFi@item
286 \ltx@newif\ifEmFi@item

\ifEmFi@finished
287 \ltx@newif\ifEmFi@finished

\ifEmFi@id
288 \ltx@newif\ifEmFi@id
```

2.7 Key value definitions

```
\EmFi@GlobalKey
289 \def\EmFi@GlobalKey#1#2{%
290   \global\expandafter\let\csname KV@#1@#2\expandafter\endcsname
291   \csname KV@#1@#2\endcsname
292 }

\EmFi@GlobalDefaultKey
293 \def\EmFi@GlobalDefaultKey#1#2{%
294   \EmFi@GlobalKey{#1}{#2}%
295   \global\expandafter\let
296   \csname KV@#1@#2@default\expandafter\endcsname
297   \csname KV@#1@#2@default\endcsname
298 }

\EmFi@DefineKey
299 \def\EmFi@DefineKey#1#2{%
300   \kv@define@key{EmFi}{#1}{%
301     \expandafter\def\csname EmFi@#1\endcsname{##1}%
302   }%
303   \expandafter\def\csname EmFi@#1\endcsname{#2}%
304 }
```

Subtype of the embedded file (optional).

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

File specification string.

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

File specification string in Unicode.

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

File system (optional).

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

Description (optional).

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

Method for converting text to PDF strings.

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

Option id as key for object numbers.

```
321 \kv@define@key{EmFi}{id}{%  
322   \def\EmFi@id{#1}%  
323   \EmFi@idtrue  
324 }
```

\EmFi@defobj

```
325 \def\EmFi@defobj#1{%  
326   \ifEmFi@id  
327     \expandafter\xdef\cename EmFi@#1@\EmFi@id\endcename{%  
328       \the\pdflastobj\ltx@space 0 R%  
329     }%  
330   \fi  
331 }
```

\embedfileifobjectexists

```
332 \def\embedfileifobjectexists#1#2{%  
333   \expandafter\ifx\cename EmFi@#2@#1\endcename\relax  
334   \expandafter\ltx@secondoftwo  
335   \else  
336     \expandafter\ltx@firstoftwo  
337   \fi  
338 }
```

\embedfilegetobject

```
339 \def\embedfilegetobject#1#2{%  
340   \embedfileifobjectexists{#1}{#2}{%  
341     \cename EmFi@#2@#1\endcename  
342   }{%  
343     0 0 R%  
344   }%  
345 }
```

Initial view of the collection.

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

```

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

\embedfilesetup
369 \def\embedfilesetup{%
370   \ifEmFi@finished
371     \def\EmFi@next##1{ }%
372     \EmFi@Error{%
373       \string\embedfilefield\ltx@space after \string\embedfilefinish
374     }{%
375       The list of embedded files is already written.%
376     }%
377   \else
378     \def\EmFi@next{%
379       \kvsetkeys{EmFi}%
380     }%
381   \fi
382 \EmFi@next
383 }

\EmFi@schema
384 \def\EmFi@schema{ }

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

\EmFi@@order
386 \let\EmFi@@order\relax

\EmFi@fieldlist
387 \def\EmFi@fieldlist{ }

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

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

```

```

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

```

```

470     \expandafter\def\csname EmFi@P@#1\endcsname{##1}%
471 }%
472 \EmFi@GlobalKey{EmFi}{\EmFi@key.prefix}%
473 \kv@define@key{EmFiSo}{\EmFi@key}[ascending]{%
474   \EdefSanitize\EmFi@temp{##1}%
475   \ifx\EmFi@temp\EmFi@S@ascending
476     \def\EmFi@temp{true}%
477   \else\ifx\EmFi@temp\EmFi@S@descending
478     \def\EmFi@temp{false}%
479   \else
480     \def\EmFi@temp{}%
481     \EmFi@Error{%
482       Unknown sort order `~\EmFi@temp'.\MessageBreak
483       Supported values: `~\EmFi@S@ascending', %
484       ~\EmFi@S@descending
485     }\@ehc
486   \fi\fi
487   \ifx\EmFi@temp\ltx@empty
488   \else
489     \xdef\EmFi@sortkeys{%
490       \EmFi@sortkeys
491       /\pdf@escapename{#1}%
492     }%
493     \ifx\EmFi@sortorders\ltx@empty
494       \global\let\EmFi@sortorders\EmFi@temp
495       \gdef\EmFi@sortcase{1}%
496     \else
497       \xdef\EmFi@sortorders{%
498         \EmFi@sortorders
499         \ltx@space
500         \EmFi@temp
501       }%
502       \xdef\EmFi@sortcase{2}%
503     \fi
504   \fi
505 }%
506 \EmFi@GlobalDefaultKey{EmFiSo}\EmFi@key
507 \endgroup
508 \else
509   \EmFi@Error{%
510     Field `~\EmFi@key' is already defined%
511   }\@ehc
512 \fi
513 \fi
514 }

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

```



```

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

541 \kv@define@key{EmFiFi}{title}{%
542 \def\EmFi@title{#1}%
543 }

```

\EmFi@setboolean

```

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

559 \kv@define@key{EmFiFi}{visible}[true]{%
560 \EmFi@setboolean{visible}{#1}%
561 }

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

```

\EmFi@sortkeys

```

565 \def\EmFi@sortkeys{}

```

\EmFi@sortorders

```

566 \def\EmFi@sortorders{}

```

\embedfilesort

```

567 \def\embedfilesort{%
568 \kvsetkeys{EmFiSo}%
569 }

```

2.8 Embed the file

\embedfile

```

570 \def\embedfile{%
571 \ltx@ifnextchar{\EmFi@embedfile{\EmFi@embedfile[]}}%
572 }

```

\EmFi@embedfile

```

573 \def\EmFi@embedfile[#1]#2{%
574 \ifEmFi@finished
575 \EmFi@Error{%
576 \string\embedfile\ltx@space after \string\embedfilefinish
577 }{%

```

```

578     The list of embedded files is already written.%
579 }%
580 \else
581 \begingroup
582 \def\EmFi@file{#2}%
583 \kvsetkeys{EmFi}{#1}%
584 \expandafter\expandafter\expandafter
585 \ifx\expandafter\expandafter\expandafter
586   \pdf@filesize{\EmFi@file}\%
587   \EmFi@Error{%
588     File '\EmFi@file' not found%
589   }{%
590     The unknown file is not embedded.%
591   }%
592 \else
593 \edef\EmFi@@filespec{%
594   \pdf@escapestring{\EmFi@filespec}%
595 }%
596 \ifx\EmFi@ucfilespec\ltx@empty
597   \let\EmFi@@ucfilespec\ltx@empty
598 \else
599   \EmFi@convert\EmFi@ucfilespec\EmFi@@ucfilespec
600 \fi
601 \ifx\EmFi@desc\ltx@empty
602   \let\EmFi@@desc\ltx@empty
603 \else
604   \EmFi@convert\EmFi@desc\EmFi@@desc
605 \fi
606 \ifEmFi@item
607   \let\do\EmFi@do
608   \immediate\pdfobj{%
609     <<%
610       \EmFi@fieldlist
611     >>%
612   }%
613   \edef\EmFi@ci{\the\pdf@lastobj}%
614 \fi
615 \immediate\pdfobj stream attr{%
616   /Type/EmbeddedFile%
617   \ifx\EmFi@mimetype\ltx@empty
618   \else
619     /Subtype/\pdf@escapename{\EmFi@mimetype}%
620   \fi
621   /Params<<%
622     /ModDate(\pdf@filemoddate{\EmFi@file})%
623     /Size \pdf@filesize{\EmFi@file}%
624     /Checksum<\pdf@filemdfivesum{\EmFi@file}>%
625   >>%
626 }file{\EmFi@file}\relax
627 \EmFi@defobj{EmbeddedFile}%
628 \immediate\pdfobj{%
629   <<%
630     /Type/Filespec%
631     \ifx\EmFi@filesystem\ltx@empty
632     \else
633       /FS/\pdf@escapename{\EmFi@filesystem}%
634     \fi
635     /F(\EmFi@@filespec)%
636     \ifx\EmFi@@ucfilespec\ltx@empty
637     \else
638       /UF(\EmFi@@ucfilespec)%
639     \fi

```

```

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

```

\EmFi@do

```

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

```

\EmFi@convert

```

676 \def\EmFi@convert#1#2{%
677   \ifnum\pdf@stricmp{\EmFi@stringmethod}{psd}=0 %
678   \pdfstringdef\EmFi@temp{#1}%
679   \let#2\EmFi@temp
680   \else
681     \edef#2{\pdf@escapestring{#1}}%
682   \fi
683 }

684 \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, ...).

```

685 \def\EmFi@add#1#2{%
686   \begingroup
687   \ifx\EmFi@list\ltx@empty
688     \xdef\EmFi@list{\noexpand\do{#1}{#2}}%
689   \else
690     \def\do##1##2{%
691       \ifnum\pdf@stricmp{##1}{#1}>0 %
692         \edef\x{%

```

```

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

```

\embedfilefinish

```

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

```

Write /EmbeddedFiles entry.

```

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

```

Write collection objects.

```

742   \ifx\EmFi@initialfile\ltx@empty
743   \else
744     \EmFi@collectiontrue
745   \fi
746   \ifEmFi@collection
747     \ifx\EmFi@initialfile\ltx@empty
748       \let\EmFi@initialfile\ltx@empty
749     \else

```

```

750     \edef\EmFi@@initialfile{%
751         \pdf@escapestring{\EmFi@initialfile}%
752     }%
753     \fi

```

Look for initial file among the embedded files.

```

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

```

```

811     \fi
812     >>%
813   }%
814   \pdfcatalog{%
815     /Collection \the\pdfastobj\ltx@space0 R%
816   }%
817   \fi
818   \fi
819   \fi
820 }

821 \begingroup\expandafter\expandafter\expandafter\endgroup
822 \expandafter\ifx\csname AtEndDocument\endcsname\relax
823 \else
824   \AtEndDocument{\embedfilefinish}%
825 \fi

826 \EmFi@AtEnd%
827 \</package>

```

3 Test

3.1 Catcode checks for loading

```

828 <*test1>

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

```

```

866 \RestoreCatcodes
867 \catcode\the\count@=\the\catcode\count@\relax
868 }%
869 \ifnum\count@<255 %
870 \advance\count@ 1 %
871 \repeat
872
873 \def\RangeCatcodeInvalid#1#2{%
874 \count@=#1\relax
875 \loop
876 \catcode\count@=15 %
877 \ifnum\count@<#2\relax
878 \advance\count@ 1 %
879 \repeat
880 }
881 \def\RangeCatcodeCheck#1#2#3{%
882 \count@=#1\relax
883 \loop
884 \ifnum#3=\catcode\count@
885 \else
886 \errmessage{%
887 Character \the\count@\space
888 with wrong catcode \the\catcode\count@\space
889 instead of \number#3%
890 }%
891 \fi
892 \ifnum\count@<#2\relax
893 \advance\count@ 1 %
894 \repeat
895 }
896 \def\space{ }
897 \expandafter\ifx\csname LoadCommand\endcsname\relax
898 \def\LoadCommand{\input embedfile.sty\relax}%
899 \fi
900 \def\Test{%
901 \RangeCatcodeInvalid{0}{47}%
902 \RangeCatcodeInvalid{58}{64}%
903 \RangeCatcodeInvalid{91}{96}%
904 \RangeCatcodeInvalid{123}{255}%
905 \catcode`\@=12 %
906 \catcode`\=0 %
907 \catcode`\%=14 %
908 \LoadCommand
909 \RangeCatcodeCheck{0}{36}{15}%
910 \RangeCatcodeCheck{37}{37}{14}%
911 \RangeCatcodeCheck{38}{47}{15}%
912 \RangeCatcodeCheck{48}{57}{12}%
913 \RangeCatcodeCheck{58}{63}{15}%
914 \RangeCatcodeCheck{64}{64}{12}%
915 \RangeCatcodeCheck{65}{90}{11}%
916 \RangeCatcodeCheck{91}{91}{15}%
917 \RangeCatcodeCheck{92}{92}{0}%
918 \RangeCatcodeCheck{93}{96}{15}%
919 \RangeCatcodeCheck{97}{122}{11}%
920 \RangeCatcodeCheck{123}{255}{15}%
921 \RestoreCatcodes
922 }
923 \Test
924 \csname @@end\endcsname
925 \end
926 </test1>

```

3.2 Simple test

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

952 <*test3>
953 \NeedsTeXFormat{LaTeX2e}
954 \let\SavedJobname\jobname
955 \def\jobname{embedfile}
956 \RequirePackage{dtx-attach}[2016/05/15]
957 \let\jobname\SavedJobname
958 \documentclass{minimal}
959 \begin{document}
960   Test (\LaTeX): \texttt{embedfile.dtx} should be embedded.%
961 \end{document}
962 </test3>
```

3.3 Test for ini-TeX

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



```

985 \shipout\hbox{}
986 \embedfile[%
987   stringmethod=escape,%
988   mimetype=plain/text,%
989   desc={iniTeX source},%
990 ]{\jobname.tex}
991 \embedfilefinish
992 \end
993 </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 (`te \TeX` , `mik \TeX` , ...) relies on file name databases, you must refresh these. For example, `te \TeX` 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`.

```
994 <*catalogue>
995 <?xml version='1.0' encoding='us-ascii'?>
996 <!DOCTYPE entry SYSTEM 'catalogue.dtd'>
997 <entry datestamp='$Date$' modifier='$Author$' id='embedfile'>
998   <name>embedfile</name>
999   <caption>Embed files into PDF.</caption>
1000   <authorref id='auth:oberdiek'/>
1001   <copyright owner='Heiko Oberdiek' year='2006-2011'/>
1002   <license type='lppl1.3'/>
1003   <version number='2.7'/>
1004   <description>
1005     This package embeds files in a PDF document, using the PDF
1006     format's embedding operation (note the contrast with the attach
1007     operation used by the <xref refid='attachfile'>attachfile</xref>
1008     and <xref refid='attachfile2'>attachfile2</xref> packages).
1009     Currently only <xref refid='pdftex'>pdf $\text{\TeX}$ </xref> &gt;=1.30, in
1010     PDF mode, is supported.
1011   </description>
```

```

1012 <p/>
1013 The package is part of the <xref refid='oberdiek'>oberdiek</xref>
1014 bundle.
1015 </description>
1016 <documentation details='Package documentation'
1017 href='ctan:/macros/latex/contrib/oberdiek/embedfile.pdf'/>
1018 <ctan file='true' path='/macros/latex/contrib/oberdiek/embedfile.dtx'/>
1019 <miktex location='oberdiek'/>
1020 <texlive location='oberdiek'/>
1021 <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip'/>
1022 </entry>
1023 </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

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>\#</code>	831	<code>\body</code>	849, 853
<code>\%</code>	907	<code>\bye</code>	35, 950
<code>\@</code>	832, 905	C	
<code>\@PackageError</code>	234	<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, 829, 830, 831, 832, 867, 876, 884, 888, 905, 906, 907, 964, 965
<code>\@PackageWarningNoLine</code>	767	<code>\count@</code>	401, 402, 403, 834, 863, 867, 869, 870, 874, 876, 877, 878, 882, 884, 887, 888, 892, 893
<code>\@ehc</code>	364, 485, 511, 555	<code>\countdef</code>	834
<code>\@firstofone</code>	840, 843	<code>\csname</code>	124, 131, 160, 176, 186, 225, 255, 265, 290, 291, 296, 297, 301, 303, 327, 333, 341, 399,
<code>\@gobble</code>	837, 845		
<code>\@undefined</code>	168, 236, 311		
<code>\%</code>	586, 906		
<code>\{</code>	829, 964		
<code>\}</code>	830, 965		
A			
<code>\advance</code>	402, 870, 878, 893		
<code>\aftergroup</code>	139		
<code>\AtEndDocument</code>	824		
B			
<code>\begin</code>	75, 959		

444, 455, 464, 470, 547, 550, 661, 662, 664, 668, 670, 672, 822, 833, 836, 839, 842, 897, 924	
D	
\directlua	968
\do 434, 437, 607, 688, 690, 695, 696, 700, 701, 707, 711, 730, 756, 760	
\documentclass	41, 958
E	
\EdefSanitize	265, 347, 398, 464, 474, 516, 545
\embedfile	3, 16, 21, 25, 79, 85, 91, 105, 570, 576, 929, 986
\embedfilefield	4, 50, 54, 58, 62, 66, 373, 389
\embedfilefinish	3, 34, 373, 392, 576, 718, 824, 949, 991
\embedfilegetobject	5, 339, 942
\embedfileifobjectexists 5, 332, 340, 940	
\embedfilessetup	4, 4, 11, 46, 369
\embedfilesort	5, 71, 567
\EmFi@@desc	602, 604, 640, 642
\EmFi@@filespec	593, 635, 654
\EmFi@@initialfile	748, 750, 758, 772, 798, 800
\EmFi@@order	386, 424, 787
\EmFi@@ucfilespec	597, 599, 636, 638
\EmFi@add	653, 685
\EmFi@AtEnd 205, 206, 223, 248, 261, 826	
\EmFi@ci	613, 648
\EmFi@collectiontrue	349, 397, 744
\EmFi@convert	409, 443, 454, 599, 604, 676
\EmFi@DefineKey	299, 305, 306, 307, 308, 309, 310, 368
\EmFi@defobj	325, 627, 652
\EmFi@desc	601, 604
\EmFi@details	267
\EmFi@do	607, 660
\EmFi@editfalse	407
\EmFi@embedfile	571, 573
\EmFi@Error	232, 242, 256, 361, 372, 391, 481, 509, 534, 552, 575, 587, 720
\EmFi@fieldlist	387, 435, 436, 610
\EmFi@file	306, 582, 586, 588, 622, 623, 624, 626
\EmFi@filespec	594
\EmFi@filesystem	631, 633
\EmFi@finishedtrue	728
\EmFi@GlobalDefaultKey	293, 506
\EmFi@GlobalKey	289, 294, 449, 460, 466, 472
\EmFi@hidden	269
\EmFi@id	322, 327
\EmFi@idtrue	323
\EmFi@initialfile 742, 747, 751, 768, 771	
\EmFi@itemtrue	441, 452, 463, 469
\EmFi@key	398, 399, 404, 412, 437, 440, 449, 451, 460, 462, 466, 468, 472, 473, 506, 510
\EmFi@list	684, 687, 688, 704, 711, 714, 726, 735, 763
\EmFi@mimetype	617, 619
\EmFi@next 348, 360, 366, 371, 378, 382	
\EmFi@order	385, 401, 403, 424, 788
\EmFi@RequirePackage	224, 232, 237, 239, 250, 251, 252, 253, 263
\EmFi@S@ascending	278, 475, 483
\EmFi@S@creationdate	276, 419, 529
\EmFi@S@date	271, 414, 450, 519
\EmFi@S@desc	274, 417, 525
\EmFi@S@descending	279, 477, 484
\EmFi@S@details	352, 353, 354
\EmFi@S@false	281, 549
\EmFi@S@file	273, 416, 523
\EmFi@S@hidden	357, 358, 804
\EmFi@S@moddate	275, 418, 527
\EmFi@S@number	272, 415, 461, 521
\EmFi@S@size	277, 420, 531
\EmFi@S@text	270, 405, 439, 517
\EmFi@S@tile	355, 356, 802
\EmFi@S@true	280, 546
\EmFi@schema	384, 410, 411, 794, 796
\EmFi@setboolean	544, 560, 563
\EmFi@sortcase	388, 495, 502, 774
\EmFi@sortkeys 489, 490, 565, 778, 783	
\EmFi@sortorders	493, 494, 497, 498, 566, 779, 784
\EmFi@stringmethod	677
\EmFi@temp	264, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 347, 351, 353, 355, 357, 362, 442, 443, 446, 453, 454, 457, 474, 475, 476, 477, 478, 480, 482, 487, 494, 500, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 535, 545, 546, 549, 553, 678, 679, 775, 777, 782, 807, 808, 810
\EmFi@tile	268
\EmFi@title	404, 409, 423, 542
\EmFi@type	405, 414, 415, 416, 417, 418, 419, 420, 439, 450, 461, 518, 520, 522, 524, 526, 528, 530, 532
\EmFi@ucfilespec	596, 599
\EmFi@view 352, 354, 356, 358, 802, 804	
\EmFi@visibletrue	406
\empty	127, 128
\end	97, 925, 961, 992
\endcsname	124, 131, 160, 176, 186, 225, 255, 265, 290, 291, 296, 297, 301, 303, 327, 333, 341, 399, 445, 456, 464, 470, 547, 550, 661, 662, 664, 668, 670, 672, 822, 833, 836, 839, 842, 897, 924
\endinginput	139, 223
\endlinechar	114, 145, 181, 187, 199
\errmessage	886, 944

F		M	
<code>\f</code>	755, 759, 765	<code>\ltx@space</code>	328, 373, 392, 499, 576, 645, 648, 655, 739, 815
G		N	
<code>\gdef</code>	385, 495	<code>\NeedsTeXFormat</code>	40, 101, 953
<code>\Gin@driver</code>	5	<code>\next</code>	854, 856, 858
H		<code>\nopagenumbers</code>	935
<code>\hbox</code>	985	<code>\number</code>	889
I		<code>\numexpr</code>	977
<code>\ifcase</code>	774	O	
<code>\ifEmFi@collection</code>	282, 746	<code>\outputmode</code>	980
<code>\ifEmFi@edit</code>	285, 429	P	
<code>\ifEmFi@finished</code>	287, 370, 390, 574, 719	<code>\PackageInfo</code>	136
<code>\ifEmFi@id</code>	288, 326	<code>\par</code>	939
<code>\ifEmFi@item</code>	286, 606, 647	<code>\pdf@escapename</code>	412, 491, 619, 633, 664, 667
<code>\ifEmFi@sort</code>	283	<code>\pdf@escapestring</code>	594, 681, 751
<code>\ifEmFi@visible</code>	284, 425	<code>\pdf@filemdfivesum</code>	624
<code>\ifluatex</code>	967	<code>\pdf@filemoddate</code>	622
<code>\ifnum</code>	677, 691, 788, 869, 877, 884, 892	<code>\pdf@filesize</code>	586, 623
<code>\ifpdf</code>	240	<code>\pdf@strcmp</code>	677, 691
<code>\ifx</code>	125, 128, 131, 160, 168, 171, 225, 236, 255, 311, 314, 333, 351, 353, 355, 357, 399, 414, 415, 416, 417, 418, 419, 420, 439, 450, 461, 475, 477, 487, 493, 517, 519, 521, 523, 525, 527, 529, 531, 546, 549, 585, 596, 601, 617, 631, 636, 640, 661, 662, 668, 687, 726, 742, 747, 758, 765, 794, 798, 802, 804, 807, 822, 833, 836, 839, 842, 897, 976	<code>\pdfcatalog</code>	814
<code>\immediate</code>	133, 162, 608, 615, 628, 733, 792	<code>\pdfextension</code>	236, 976, 978, 979
<code>\input</code> ..	4, 6, 7, 227, 898, 928, 966, 984	<code>\pdffeedback</code>	977
<code>\iterate</code>	850, 852, 854	<code>\pdflastobj</code>	328, 613, 645, 655, 739, 815, 977
J		<code>\pdfnames</code>	738, 978
<code>\jobname</code>	95, 103, 108, 109, 954, 955, 957, 990	<code>\pdfobj</code> ...	608, 615, 628, 733, 792, 979
K		<code>\pdfoutput</code>	980, 983
<code>\kv@define@key</code>	300, 321, 346, 440, 451, 462, 468, 473, 515, 541, 559, 562	<code>\pdfstringdef</code>	43, 311, 314, 678
<code>\kvsetkeys</code>	379, 408, 568, 583	<code>\protected</code>	977, 978, 979
L		<code>\ProvidesPackage</code>	102, 129, 177
<code>\LaTeX</code>	960	R	
<code>\LoadCommand</code>	898, 908	<code>\RangeCatcodeCheck</code>	881, 909, 910, 911, 912, 913, 914, 915, 916, 917, 918, 919, 920
<code>\loop</code>	848, 864, 875, 883	<code>\RangeCatcodeInvalid</code>	873, 901, 902, 903, 904
<code>\ltx@empty</code>	351, 487, 493, 596, 597, 601, 602, 617, 631, 636, 640, 684, 687, 726, 742, 747, 748, 771, 772, 794, 798, 807	<code>\repeat</code>	848, 860, 871, 879, 894
<code>\ltx@firstoftwo</code>	336	<code>\RequirePackage</code>	104, 230, 956
<code>\ltx@gobbletwo</code>	760	<code>\resetatcatcode</code>	8
<code>\ltx@ifnextchar</code>	571	<code>\RestoreCatcodes</code> ..	862, 865, 866, 921
<code>\ltx@newif</code>	282, 283, 284, 285, 286, 287, 288	S	
<code>\ltx@secondoftwo</code>	334	<code>\SavedJobname</code>	954, 957
T		<code>\shipout</code>	985
<code>\Test</code>	900, 923, 938, 947, 948	<code>\space</code>	887, 888, 896
<code>\TeX</code>	936	<code>\stop</code>	703, 710, 714
<code>\texttt</code>	960	T	
<code>\the</code>	187, 188, 189, 190, 191, 192, 193, 194, 207, 328, 403, 613, 645, 655, 694, 701, 704, 707, 711, 739, 815, 867, 887, 888	T	

\TMP@EnsureCode	W
. 204, 211, 212, 213, 214, 215,	\write 133, 162
216, 217, 218, 219, 220, 221, 222	
\toks@ . 693, 694, 701, 704, 707, 711, 713	
\tt 936	X
U	\x 124, 125,
\undefined 976	128, 132, 136, 138, 161, 166,
\usepackage 42, 45	176, 185, 197, 692, 699, 757, 758