

# The embedfile package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2007/11/25 v2.3

## Abstract

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

## Contents

<b>1</b>	<b>Documentation</b>	<b>2</b>
1.1	Introduction	2
1.1.1	Future development	2
1.2	User interface	2
1.3	Collection support (PDF 1.7)	3
1.4	Export of object references	5
1.4.1	Example	5
1.5	Examples	5
1.5.1	plain-TeX	5
1.5.2	Collection example	6
1.6	Package dtx-attach	7
<b>2</b>	<b>Implementation</b>	<b>7</b>
2.1	Reload check and package identification	7
2.2	Catcodes	8
2.3	Tools	9
2.4	Check for recent pdfTeX in PDF mode	9
2.5	Strings	10
2.6	Switches	11
2.7	Key value definitions	11
2.8	Embed the file	18
<b>3</b>	<b>Test</b>	<b>22</b>
3.1	Catcode checks for loading	22
3.2	Simple test	24
<b>4</b>	<b>Installation</b>	<b>24</b>
4.1	Download	24
4.2	Bundle installation	25
4.3	Package installation	25
4.4	Refresh file name databases	25
4.5	Some details for the interested	25
<b>5</b>	<b>References</b>	<b>26</b>

<b>6 History</b>	<b>26</b>
[2006/08/16 v1.0]	26
[2007/04/11 v1.1]	26
[2007/09/09 v1.2]	26
[2007/10/28 v2.0]	26
[2007/10/29 v2.1]	26
[2007/11/11 v2.2]	26
[2007/11/25 v2.3]	27
<b>7 Index</b>	<b>27</b>

# 1 Documentation

## 1.1 Introduction

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

### 1.1.1 Future development

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

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

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

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

## 1.2 User interface

This package `embedfile` can be used with both  $\text{\LaTeX}$  and plain- $\text{\TeX}$ . See [subsubsection 1.5.1](#) that explains the use with plain- $\text{\TeX}$  by an example. In  $\text{\LaTeX}$  the package is loaded as usually. There are no options.

```
\usepackage{embedfile}
```

`\embedfile` [*options*] {*file*}

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

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

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

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

**mimetype** This sets the mime type ([4]) of the file, see [subsection 1.5.1](#) for examples and [5] for a list of officially registered types.

**desc** The description for the file.

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

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

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

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

`\embedfilefinish`

The list of all embedded files must be added as data structure in the PDF file. In case of L<sup>A</sup>T<sub>E</sub>X this is automatically done. The package uses `\AtEndDocument`. Then the list of all files should be known. However, plain-T<sub>E</sub>X does not know about `\AtEndDocument`. Thus the user must call `\embedfilefinish` at the end of the document after the last file is embedded.

`\embedfilesetup` {*options*}

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

### 1.3 Collection support (PDF 1.7)

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

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

Name	Description	Modified	Size
...	...	...	...

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

Collection support is enabled automatically, if it is used.

`\embedfilesetup {⟨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.

`\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-TeX

The package can be used with plain-TeX. 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-TeX does not provide a hook at end of the document, you have to call `\embedfilefinish` manually at the end after the last embedded file.

```
1 ⟨*exampleplain⟩
2 %<<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}[2007/11/25]
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/](#). It also serves as small example for the use of the package with  $\text{\LaTeX}$ .

```

100 (*dtxattach)
101 \NeedsTeXFormat{LaTeX2e}
102 \ProvidesPackage{dtx-attach}
103 [2007/11/25 v2.3 Embed \string\jobname.dtx (HO)]%
104 \RequirePackage{embedfile}[2007/11/25]
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  $\text{\LaTeX}$ .

```

112 \begingroup
113  \catcode44 12 % ,
114  \catcode45 12 % -
115  \catcode46 12 % .

```

```

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

```

Package identification:

```

140 \begingroup
141   \catcode35 6 % #
142   \catcode40 12 % (
143   \catcode41 12 % )
144   \catcode44 12 % ,
145   \catcode45 12 % -
146   \catcode46 12 % .
147   \catcode47 12 % /
148   \catcode58 12 % :
149   \catcode64 11 % @
150   \catcode91 12 % [
151   \catcode93 12 % ]
152   \catcode123 1 % {
153   \catcode125 2 % }
154   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
155     \def\x#1#2#3[#4]{\endgroup
156       \immediate\write-1{Package: #3 #4}%
157       \xdef#1{#4}%
158     }%
159   \else
160     \def\x#1#2[#3]{\endgroup
161       #2[#{#3}]%
162       \ifx#1\@undefined
163         \xdef#1{#3}%
164       \fi
165       \ifx#1\relax
166         \xdef#1{#3}%
167       \fi
168     }%
169   \fi
170 \expandafter\x\csname ver@embedfile.sty\endcsname
171 \ProvidesPackage{embedfile}%
172 [2007/11/25 v2.3 embed files into PDF (H0)]

```

## 2.2 Catcodes

```

173 \begingroup

```



```

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

```

## 2.3 Tools

\EmFi@RequirePackage

```

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

```

\EmFi@Error

```

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

```

## 2.4 Check for recent pdfTeX in PDF mode

Load package ifpdf and check mode.

```

221 \EmFi@RequirePackage{ifpdf}[2007/09/09]
222 \ifpdf
223 \else
224 \EmFi@Error{%
225 Missing pdfTeX in PDF mode%

```

```

226 }{%
227     Currently other drivers are not supported. %
228     Package loading is aborted.%
229 }%
230 \EmFi@AtEnd
231 \expandafter\endinput
232 \fi

233 \EmFi@RequirePackage{pdftexcmds}[2007/11/11]

Check version.

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

```

## 2.5 Strings

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

```

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

245 \def\EmFi@temp#1{%
246     \expandafter\EdefSanitize\csname EmFi@S@#1\endcsname{#1}%
247 }

\EmFi@details

248 \EmFi@temp{details}%

\EmFi@tile

249 \EmFi@temp{tile}%

\EmFi@hidden

250 \EmFi@temp{hidden}%

\EmFi@S@text

251 \EmFi@temp{text}

\EmFi@S@date

252 \EmFi@temp{date}

\EmFi@S@number

253 \EmFi@temp{number}

\EmFi@S@file

254 \EmFi@temp{file}

\EmFi@S@desc

255 \EmFi@temp{desc}

\EmFi@S@moddate

256 \EmFi@temp{moddate}

\EmFi@S@creationdate

257 \EmFi@temp{creationdate}

```

```

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

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

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

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

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

```

## 2.6 Switches

```

\ifEmFi@collection
263 \newif\ifEmFi@collection

\ifEmFi@initialfile
264 \newif\ifEmFi@initialfile

\ifEmFi@sort
265 \newif\ifEmFi@sort

\ifEmFi@visible
266 \newif\ifEmFi@visible

\ifEmFi@edit
267 \newif\ifEmFi@edit

\ifEmFi@item
268 \newif\ifEmFi@item

\ifEmFi@finished
269 \newif\ifEmFi@finished

\ifEmFi@id
270 \newif\ifEmFi@id

```

## 2.7 Key value definitions

```

271 \expandafter\ifx\csname define@key\endcsname\relax
272 \chardef\EmFi@plain=\z@
273 \def\EmFi@temp#1{%
274 \begingroup\expandafter\expandafter\expandafter\endgroup
275 \expandafter\ifx\csname#1\endcsname\relax
276 \chardef\EmFi@plain=\@ne
277 \fi
278 }%
279 \EmFi@temp{NeedsTeXFormat}%
280 \EmFi@temp{ProvidesPackage}%
281 \EmFi@temp{DeclareOption}%
282 \EmFi@temp{ExecuteOptions}%
283 \EmFi@temp{ProcessOptions}%
284 \ifnum\EmFi@plain=\@ne
285 \def\EmFi@temp#1{%
286 \expandafter\let\csname EmFi@Org#1\expandafter\endcsname

```

```

287             \csname#1\endcsname
288     \expandafter\def\csname#1\endcsname
289     }%
290     \EmFi@temp{NeedsTeXFormat}#1{}%
291     \EmFi@temp{ProvidesPackage}#1[#2]{}% hash-ok
292     \EmFi@temp{DeclareOption}#1{}%
293     \EmFi@temp{ExecuteOptions}#1{}%
294     \EmFi@temp{ProcessOptions}{}%

\KV@errx LATEX's option processing is not available with plain-TEX. Thus we define the de-
fault error command \KV@errx here, also using package infwarerr's \@PackageError.

295     \def\KV@errx#1{%
296         \@PackageError{keyval}{#1}\@ehc
297     }%

```

Other macros from L<sup>A</sup>T<sub>E</sub>X's kernel that are used by package keyval.

```

\@ifnextchar

298     \expandafter\ifx\csname @ifnextchar\endcsname\relax
299     \def\@ifnextchar#1#2#3{%
300         \let\reserved@d=#1%
301         \def\reserved@a{#2}%
302         \def\reserved@b{#3}%
303         \futurelet\@let@token\@ifnch
304     }%
305     \def\@ifnch{%
306         \ifx\@let@token\@sptoken
307             \let\reserved@c\@xifnch
308         \else
309             \ifx\@let@token\reserved@d
310                 \let\reserved@c\reserved@a
311             \else
312                 \let\reserved@c\reserved@b
313             \fi
314         \fi
315         \reserved@c
316     }%
317     \begingroup
318         \def\:{\global\let\@sptoken=}%
319         \: % this makes \@sptoken a space token
320         \def\:{\@xifnch}%
321         \expandafter\gdef\:{%
322             \futurelet\@let@token\@ifnch
323         }%
324     \endgroup
325     \fi

\@namedef

326     \expandafter\ifx\csname @namedef\endcsname\relax
327     \def\@namedef#1{%
328         \expandafter\def\csname#1\endcsname
329         }%
330     \fi

331     \fi
332     \EmFi@RequirePackage{keyval}[1999/03/16]%
333     \ifnum\EmFi@plain=\@ne
334         \def\EmFi@temp#1{%
335             \expandafter\let\csname#1\expandafter\endcsname
336             \csname EmFi@Org#1\endcsname
337         }%
338         \EmFi@temp{NeedsTeXFormat}%
339         \EmFi@temp{ProvidesPackage}%

```

```

340 \EmFi@temp{DeclareOption}%
341 \EmFi@temp{ExecuteOptions}%
342 \EmFi@temp{ProcessOptions}%
343 \fi
344 \fi

```

\EmFi@GlobalKey

```

345 \def\EmFi@GlobalKey#1#2{%
346 \global\expandafter\let\csname KV@#1@#2\expandafter\endcsname
347 \csname KV@#1@#2\endcsname
348 }

```

\EmFi@GlobalDefaultKey

```

349 \def\EmFi@GlobalDefaultKey#1#2{%
350 \EmFi@GlobalKey{#1}{#2}%
351 \global\expandafter\let
352 \csname KV@#1@#2@default\expandafter\endcsname
353 \csname KV@#1@#2@default\endcsname
354 }

```

\EmFi@DefineKey

```

355 \def\EmFi@DefineKey#1#2{%
356 \define@key{EmFi}{#1}{%
357 \expandafter\def\csname EmFi@#1\endcsname{##1}%
358 }%
359 \expandafter\def\csname EmFi@#1\endcsname{#2}%
360 }

```

Subtype of the embedded file (optional).

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

```

File specification string.

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

```

File system (optional).

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

```

Description (optional).

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

```

Method for converting text to PDF strings.

```

365 \EmFi@DefineKey{stringmethod}{%
366 \ifx\pdfstringdef\@undefined
367 escape%
368 \else
369 \ifx\pdfstringdef\relax
370 escape%
371 \else
372 psd%
373 \fi
374 \fi
375 }

```

Option id as key for object numbers.

```

376 \define@key{EmFi}{id}{%
377 \def\EmFi@id{#1}%
378 \EmFi@idtrue
379 }

```

\EmFi@defobj

```

380 \def\EmFi@defobj#1{%
381 \ifEmFi@id
382 \expandafter\xdef\csname EmFi@#1@\EmFi@id\endcsname{%

```

```

383     \the\pdflastobj\space 0 R%
384   }%
385 \fi
386 }

```

\embedfileifobjectexists

```

387 \def\embedfileifobjectexists#1#2{%
388   \expandafter\ifx\csname EmFi@#2@#1\endcsname\relax
389     \expandafter\@secondoftwo
390   \else
391     \expandafter\@firstoftwo
392   \fi
393 }

```

\@firstoftwo

```

394 \expandafter\ifx\csname @firstoftwo\endcsname\relax
395   \long\def\@firstoftwo#1#2{#1}%
396 \fi

```

\@secondoftwo

```

397 \expandafter\ifx\csname @secondoftwo\endcsname\relax
398   \long\def\@secondoftwo#1#2{#2}%
399 \fi

```

\embedfilegetobject

```

400 \def\embedfilegetobject#1#2{%
401   \embedfileifobjectexists{#1}{#2}{%
402     \csname EmFi@#2@#1\endcsname
403   }{%
404     0 0 R%
405   }%
406 }

```

Initial view of the collection.

```

407 \define@key{EmFi}{view}[]{%
408   \EdefSanitize\EmFi@temp{#1}%
409   \def\EmFi@next{%
410     \global\EmFi@collectiontrue
411   }%
412   \ifx\EmFi@temp\empty
413     \let\EmFi@view\EmFi@S@details
414   \else\ifx\EmFi@temp\EmFi@S@details
415     \let\EmFi@view\EmFi@S@details
416   \else\ifx\EmFi@temp\EmFi@S@tile
417     \let\EmFi@view\EmFi@S@tile
418   \else\ifx\EmFi@temp\EmFi@S@hidden
419     \let\EmFi@view\EmFi@S@hidden
420   \else
421     \let\EmFi@next\relax
422     \EmFi@Error{%
423       Unknown value '\EmFi@temp' for key 'view'.\MessageBreak
424       Supported values: 'details', 'tile', 'hidden'.%
425     }\@ehc
426   \fi\fi\fi\fi
427   \EmFi@next
428 }
429 \EmFi@DefineKey{initialfile}{}

```

\embedfilessetup

```

430 \def\embedfilessetup{%
431   \ifEmFi@finished

```

```

432     \def\EmFi@next##1{%
433     \EmFi@Error{%
434         \string\embedfilefield\space after \string\embedfilefinish
435     }{%
436         The list of embedded files is already written.%
437     }%
438     \else
439     \def\EmFi@next{%
440         \setkeys{EmFi}%
441     }%
442     \fi
443     \EmFi@next
444 }

\EmFi@schema
445 \def\EmFi@schema{}

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

\EmFi@@order
447 \let\EmFi@@order\relax

\EmFi@fieldlist
448 \def\EmFi@fieldlist{}

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

\embedfilefield
450 \def\embedfilefield#1#2{%
451     \ifEmFi@finished
452     \EmFi@Error{%
453         \string\embedfilefield\space after \string\embedfilefinish
454     }{%
455         The list of embedded files is already written.%
456     }%
457     \else
458     \global\EmFi@collectiontrue
459     \edefSanitize\EmFi@key{#1}%
460     \expandafter\ifx\csname KV@EmFi@\EmFi@key.prefix\endcsname\relax
461     \begingroup
462     \count@=\EmFi@order
463     \advance\count@ 1 %
464     \xdef\EmFi@order{\the\count@}%
465     \let\EmFi@title\EmFi@key
466     \let\EmFi@type\EmFi@S@text
467     \EmFi@visibletrue
468     \EmFi@editfalse
469     \setkeys{EmFiFi}{#2}%
470     \EmFi@convert\EmFi@title\EmFi@title
471     \xdef\EmFi@schema{%
472         \EmFi@schema
473         /\pdf@escapename{\EmFi@key}<<%
474         /Subtype/%
475         \ifx\EmFi@type\EmFi@S@date D%
476         \else\ifx\EmFi@type\EmFi@S@number N%
477         \else\ifx\EmFi@type\EmFi@S@file F%
478         \else\ifx\EmFi@type\EmFi@S@desc Desc%
479         \else\ifx\EmFi@type\EmFi@S@moddate ModDate%
480         \else\ifx\EmFi@type\EmFi@S@creationdate CreationDate%

```

```

481         \else\ifx\EmFi@type\EmFi@S@size Size%
482         \else S%
483         \fi\fi\fi\fi\fi\fi\fi
484         /N(\EmFi@title)%
485         \EmFi@@order{\EmFi@order}%
486         \ifEmFi@visible
487         \else
488             /V false%
489         \fi
490         \ifEmFi@edit
491             /E true%
492         \fi
493     >>%
494 }%
495 \let\do\relax
496 \xdef\EmFi@fieldlist{%
497     \EmFi@fieldlist
498     \do{\EmFi@key}%
499 }%
500 \ifx\EmFi@type\EmFi@S@text
501     \define@key{EmFi}{\EmFi@key.value}{%
502         \EmFi@itemtrue
503         \def\EmFi@temp{##1}%
504         \EmFi@convert\EmFi@temp\EmFi@temp
505         \expandafter\def\csname EmFi@V@#1%
506         \expandafter\endcsname\expandafter{%
507             \expandafter(\EmFi@temp)%
508         }%
509     }%
510     \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
511 \else\ifx\EmFi@type\EmFi@S@date
512     \define@key{EmFi}{\EmFi@key.value}{%
513         \EmFi@itemtrue
514         \def\EmFi@temp{##1}%
515         \EmFi@convert\EmFi@temp\EmFi@temp
516         \expandafter\def\csname EmFi@V@#1%
517         \expandafter\endcsname\expandafter{%
518             \expandafter(\EmFi@temp)%
519         }%
520     }%
521     \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
522 \else\ifx\EmFi@type\EmFi@S@number
523     \define@key{EmFi}{\EmFi@key.value}{%
524         \EmFi@itemtrue
525         \expandafter\edef\csname EmFi@V@#1\endcsname{ ##1}%
526     }%
527     \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
528 \fi\fi\fi
529 \define@key{EmFi}{\EmFi@key.prefix}{%
530     \EmFi@itemtrue
531     \expandafter\def\csname EmFi@P@#1\endcsname{##1}%
532 }%
533 \EmFi@GlobalKey{EmFi}{\EmFi@key.prefix}%
534 \define@key{EmFiSo}{\EmFi@key}[ascending]{%
535     \edef\EmFi@temp{##1}%
536     \ifx\EmFi@temp\EmFi@S@ascending
537         \def\EmFi@temp{true}%
538     \else\ifx\EmFi@temp\EmFi@S@descending
539         \def\EmFi@temp{false}%
540     \else
541         \def\EmFi@temp{}%
542     \EmFi@Error{%

```



```

543         Unknown sort order '\EmFi@temp'.\MessageBreak
544         Supported values: '\EmFi@S@ascending', %
545         '\EmFi@S@descending
546     }\@ehc
547 \fi\fi
548 \ifx\EmFi@temp\empty
549 \else
550     \xdef\EmFi@sortkeys{%
551         \EmFi@sortkeys
552         /\pdf@escapename{#1}%
553     }%
554     \ifx\EmFi@sortorders\empty
555         \global\let\EmFi@sortorders\EmFi@temp
556         \gdef\EmFi@sortcase{1}%
557     \else
558         \xdef\EmFi@sortorders{%
559             \EmFi@sortorders
560             \space
561             \EmFi@temp
562         }%
563         \xdef\EmFi@sortcase{2}%
564     \fi
565 \fi
566 }%
567 \EmFi@GlobalDefaultKey{EmFiSo}\EmFi@key
568 \endgroup
569 \else
570     \EmFi@Error{%
571         Field '\EmFi@key' is already defined%
572     }\@ehc
573 \fi
574 \fi
575 }

576 \define@key{EmFiFi}{type}{%
577     \EdefSanitize\EmFi@temp{#1}%
578     \ifx\EmFi@temp\EmFi@S@text
579         \let\EmFi@type\EmFi@temp
580     \else\ifx\EmFi@temp\EmFi@S@date
581         \let\EmFi@type\EmFi@temp
582     \else\ifx\EmFi@temp\EmFi@S@number
583         \let\EmFi@type\EmFi@temp
584     \else\ifx\EmFi@temp\EmFi@S@file
585         \let\EmFi@type\EmFi@temp
586     \else\ifx\EmFi@temp\EmFi@S@desc
587         \let\EmFi@type\EmFi@temp
588     \else\ifx\EmFi@temp\EmFi@S@moddate
589         \let\EmFi@type\EmFi@temp
590     \else\ifx\EmFi@temp\EmFi@S@creationdate
591         \let\EmFi@type\EmFi@temp
592     \else\ifx\EmFi@temp\EmFi@S@size
593         \let\EmFi@type\EmFi@temp
594     \else
595         \EmFi@Error{%
596             Unknown type '\EmFi@temp'.\MessageBreak
597             Supported types: 'text', 'date', 'number', 'file',\MessageBreak
598             'desc', 'moddate', 'creationdate', 'size'%
599         }%
600     \fi\fi\fi\fi\fi\fi\fi\fi
601 }

602 \define@key{EmFiFi}{title}{%
603     \def\EmFi@title{#1}%
604 }

```

\EmFi@setboolean

```
605 \def\EmFi@setboolean#1#2{%
606   \EdefSanitize\EmFi@temp{#2}%
607   \ifx\EmFi@temp\EmFi@S@true
608     \csname EmFi@#1true\endcsname
609   \else
610     \ifx\EmFi@temp\EmFi@S@false
611       \csname EmFi@#1false\endcsname
612     \else
613       \EmFi@Error{%
614         Unknown value ‘\EmFi@temp’ for key ‘#1’.\MessageBreak
615         Supported values: ‘true’, ‘false’%
616       }\@ehc
617     \fi
618   \fi
619 }

620 \define@key{EmFiFi}{visible}[true]{%
621   \EmFi@setboolean{visible}{#1}%
622 }

623 \define@key{EmFiFi}{edit}[true]{%
624   \EmFi@setboolean{edit}{#1}%
625 }
```

\EmFi@sortkeys

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

\EmFi@sortorders

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

\embedfilesort

```
628 \def\embedfilesort{%
629   \setkeys{EmFiSo}%
630 }
```

## 2.8 Embed the file

\embedfile

```
631 \def\embedfile{%
632   \@ifnextchar[\EmFi@embedfile{\EmFi@embedfile[]}%
633 }
```

\EmFi@embedfile

```
634 \def\EmFi@embedfile[#1]#2{%
635   \ifEmFi@finished
636     \EmFi@Error{%
637       \string\embedfile\space after \string\embedfilefinish
638     }{%
639       The list of embedded files is already written.%
640     }%
641   \else
642     \begingroup
643     \def\EmFi@file{#2}%
644     \ifx\EmFi@file\EmFi@initialfile
645       \global\EmFi@initialfiletrue
646     \fi
647     \setkeys{EmFi}{#1}%
648     \expandafter\expandafter\expandafter
649     \ifx\expandafter\expandafter\expandafter
650       \\pdf@filesize{\EmFi@file}\\%
```

```

651     \EmFi@Error{%
652         File '\EmFi@file' not found%
653     }{%
654         The unknown file is not embedded.%
655     }%
656 \else
657     \EmFi@convert\EmFi@filespec\EmFi@@filespec
658     \ifx\EmFi@desc\empty
659         \let\EmFi@@desc\empty
660     \else
661         \EmFi@convert\EmFi@desc\EmFi@@desc
662     \fi
663     \ifEmFi@item
664         \let\do\EmFi@do
665         \immediate\pdfobj{%
666             <<%
667                 \EmFi@fieldlist
668             >>%
669         }%
670         \edef\EmFi@ci{\the\pdflastobj}%
671     \fi
672     \immediate\pdfobj stream attr{%
673         /Type/EmbeddedFile%
674         \ifx\EmFi@mimetype\empty
675         \else
676             /Subtype/\pdf@escapename{\EmFi@mimetype}%
677         \fi
678         /Params<<%
679             /ModDate(\pdf@filemoddate{\EmFi@file})%
680             /Size \pdf@filesize{\EmFi@file}%
681             /Checksum<\pdf@filemdfivesum{\EmFi@file}>%
682         >>%
683     }file{\EmFi@file}\relax
684     \EmFi@defobj{EmbeddedFile}%
685     \immediate\pdfobj{%
686         <<%
687             /Type/Filespec%
688             \ifx\EmFi@filesystem\empty
689             \else
690                 /FS/\pdf@escapename{\EmFi@filesystem}%
691             \fi
692             /F(\EmFi@@filespec)%
693             \ifx\EmFi@@desc\empty
694             \else
695                 /Desc(\EmFi@@desc)%
696             \fi
697             /EF<<%
698                 /F \the\pdflastobj\space 0 R%
699             >>%
700             \ifEmFi@item
701                 /CI \EmFi@ci\space 0 R%
702             \fi
703             >>%
704         }%
705     \EmFi@defobj{Filespec}%
706     \EmFi@add{%
707         \EmFi@@filespec
708     }{\the\pdflastobj\space 0 R}%
709     \fi
710 \endgroup
711 \fi
712 }

```

\EmFi@do

```
713 \def\EmFi@do#1{%
714   \expandafter\ifx\csname EmFi@P@#1\endcsname\relax
715     \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
716       \else
717         /\pdf@escapename{#1}\csname EmFi@V@#1\endcsname
718       \fi
719     \else
720       /\pdf@escapename{#1}<<%
721       \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
722         \else
723           /D\csname EmFi@V@#1\endcsname
724         \fi
725       /P(\csname EmFi@P@#1\endcsname)%
726     >>%
727   \fi
728 }
```

\EmFi@convert

```
729 \def\EmFi@convert#1#2{%
730   \ifnum\pdf@strcmp{\EmFi@stringmethod}{psd}=0 %
731     \pdfstringdef\EmFi@temp{#1}%
732     \let#2\EmFi@temp
733   \else
734     \edef#2{\pdf@escapestring{#1}}%
735   \fi
736 }

737 \global\let\EmFi@list\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, ...).

```
738 \def\EmFi@add#1#2{%
739   \begingroup
740     \edef\key{\pdf@escapehex{#1}}%
741     \ifx\EmFi@list\empty
742       \xdef\EmFi@list{\noexpand\do{\key}{#2}}%
743     \else
744       \def\do##1##2{%
745         \ifnum\pdf@strcmp{##1}{\key}>0 %
746           \edef\x{%
747             \toks@{%
748               \the\toks@%
749               \noexpand\do{\key}{#2}%
750               \noexpand\do{##1}{##2}%
751             }%
752           }%
753           \x
754         \def\do####1####2{%
755           \toks@\expandafter{\the\toks@\do{####1}{####2}}%
756         }%
757         \def\stop{%
758           \xdef\EmFi@list{\the\toks@}%
759         }%
760       \else
761         \toks@\expandafter{\the\toks@\do{##1}{##2}}%
762       \fi
763     }%
764     \def\stop{%
765       \xdef\EmFi@list{\the\toks@\noexpand\do{\key}{#2}}%
```

```

766     }%
767     \toks@{ }%
768     \EmFi@list\stop
769     \fi
770 \endgroup
771 }

```

\embedfilefinish

```

772 \def\embedfilefinish{%
773   \ifEmFi@finished
774     \EmFi@Error{%
775       Too many invocations of \string\embedfilefinish
776     }{%
777       The list of embedded files is already written.%
778     }%
779   \else
780     \ifx\EmFi@list\empty
781     \else

```

Write /EmbeddedFiles entry.

```

782     \global\EmFi@finishedtrue
783     \begingroup
784     \def\do##1##2{%
785       <##1>##2%
786     }%
787     \immediate\pdfobj{%
788       <<%
789       /Names[\EmFi@list]%
790       >>%
791     }%
792     \pdfnames{%
793       /EmbeddedFiles \the\pdflastobj\space 0 R%
794     }%
795 \endgroup

```

Write collection objects.

```

796   \ifEmFi@initialfile
797     \EmFi@collectiontrue
798   \fi
799   \ifEmFi@collection
800     \ifEmFi@initialfile
801     \else
802       \ifx\EmFi@initialfile\empty
803         \EmFi@convert\EmFi@initialfile\EmFi@initialfile
804       \else
805         \@PackageWarningNoLine{embedfile}{%
806           Missing initial file '\EmFi@initialfile'\MessageBreak
807           among the embedded files%
808         }%
809         \EmFi@initialfilefalse
810       \fi
811     \fi
812     \ifcase\EmFi@sortcase
813       \def\EmFi@temp{ }%
814     \or
815       \def\EmFi@temp{%
816         /S\EmFi@sortkeys
817         /A \EmFi@sortorders
818       }%
819     \else
820       \def\EmFi@temp{%
821         /S[\EmFi@sortkeys]%
822         /A[\EmFi@sortorders]%

```

```

823     }%
824   \fi
825   \def\EmFi@@order##1{%
826     \ifnum\EmFi@order>1 %
827       /O ##1%
828     \fi
829   }%
830   \immediate\pdfobj{%
831     <<%
832       \ifx\EmFi@schema\empty
833       \else
834         /Schema<<\EmFi@schema>>%
835       \fi
836       \ifEmFi@initialfile
837         /D(\EmFi@initialfile)%
838       \fi
839       \ifx\EmFi@view\EmFi@S@tile
840         /View/T%
841       \else\ifx\EmFi@view\EmFi@S@hidden
842         /View/H%
843       \fi\fi
844       \ifx\EmFi@temp\empty
845         \EmFi@temp
846       \else
847         /Sort<<\EmFi@temp>>%
848       \fi
849     >>%
850   }%
851   \pdfcatalog{%
852     /Collection \the\pdflastobj\space0 R%
853   }%
854 \fi
855 \fi
856 \fi
857 }

858 \begingroup\expandafter\expandafter\expandafter\endgroup
859 \expandafter\ifx\csname AtEndDocument\endcsname\relax
860 \else
861   \AtEndDocument{\embedfilefinish}%
862 \fi

863 \EmFi@AtEnd
864 \end{package}

```

## 3 Test

### 3.1 Catcode checks for loading

```

865 \test1\

866 \catcode'\{=1 %
867 \catcode'\}=2 %
868 \catcode'\#=6 %
869 \catcode'\@=11 %
870 \expandafter\ifx\csname count@\endcsname\relax
871   \countdef\count@=255 %
872 \fi
873 \expandafter\ifx\csname @gobble\endcsname\relax
874   \long\def@gobble#1{%
875 \fi
876 \expandafter\ifx\csname @firstofone\endcsname\relax
877   \long\def@firstofone#1{#1}%

```

```

878 \fi
879 \expandafter\ifx\csname loop\endcsname\relax
880 \expandafter\@firstofone
881 \else
882 \expandafter\@gobble
883 \fi
884 {%
885 \def\loop#1\repeat{%
886 \def\body{#1}%
887 \iterate
888 }%
889 \def\iterate{%
890 \body
891 \let\next\iterate
892 \else
893 \let\next\relax
894 \fi
895 \next
896 }%
897 \let\repeat=\fi
898 }%
899 \def\RestoreCatcodes{}
900 \count@=0 %
901 \loop
902 \edef\RestoreCatcodes{%
903 \RestoreCatcodes
904 \catcode\the\count@=\the\catcode\count@\relax
905 }%
906 \ifnum\count@<255 %
907 \advance\count@ 1 %
908 \repeat
909
910 \def\RangeCatcodeInvalid#1#2{%
911 \count@=#1\relax
912 \loop
913 \catcode\count@=15 %
914 \ifnum\count@<#2\relax
915 \advance\count@ 1 %
916 \repeat
917 }
918 \expandafter\ifx\csname LoadCommand\endcsname\relax
919 \def\LoadCommand{\input embedfile.sty\relax}%
920 \fi
921 \def\Test{%
922 \RangeCatcodeInvalid{0}{47}%
923 \RangeCatcodeInvalid{58}{64}%
924 \RangeCatcodeInvalid{91}{96}%
925 \RangeCatcodeInvalid{123}{255}%
926 \catcode'\@=12 %
927 \catcode'\=0 %
928 \catcode'\{=1 %
929 \catcode'\}=2 %
930 \catcode'\#=6 %
931 \catcode'\[=12 %
932 \catcode'\]=12 %
933 \catcode'\%=14 %
934 \catcode'\ =10 %
935 \catcode13=5 %
936 \LoadCommand
937 \RestoreCatcodes
938 }
939 \Test

```

```

940 \csname @@end\endcsname
941 \end
942 </test1>

3.2 Simple test

943 <*test2>
944 \input embedfile.sty\relax
945 \embedfile[%
946   stringmethod=escape,%
947   mimetype=plain/text,%
948   desc={LaTeX docstrip source archive for package 'embedfile'},%
949   id={embedfile.dtx}%
950 ]{embedfile.dtx}
951 \nopagenumbers
952 Test (plain-TeX): {\tt embedfile.dtx} should be embedded.%
953
954 \def\Test#1{%
955   \par
956   \embedfileifobjectexists{embedfile.dtx}{#1}{%
957     Object #1 (embedfile.dtx): %
958     \embedfilegetobject{embedfile.dtx}{#1}%
959   }{%
960     \errmessage{Missing object #1 (embedfile.dtx)}%
961   }%
962 }
963 \Test{EmbeddedFile}
964 \Test{Filespec}
965 \embedfilefinish
966 \bye
967 </test2>

968 <*test3>
969 \NeedsTeXFormat{LaTeX2e}
970 \let\SavedJobname\jobname
971 \def\jobname{embedfile}
972 \RequirePackage{dtx-attach}[2007/11/25]
973 \let\jobname\SavedJobname
974 \documentclass{minimal}
975 \begin{document}
976   Test (\LaTeX): \texttt{embedfile.dtx} should be embedded.%
977 \end{document}
978 </test3>

```

## 4 Installation

### 4.1 Download

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

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

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

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

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

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

---

<sup>1</sup><http://ftp.ctan.org/tex-archive/>



## 4.2 Bundle installation

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

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

**Script installation.** Check the directory `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-`TEX`:

```
tex embedfile.dtx
```

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

<code>embedfile.sty</code>	→ <code>tex/latex/oberdiek/embedfile.sty</code>
<code>dtx-attach.sty</code>	→ <code>tex/latex/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>embedfile.dtx</code>	→ <code>source/latex/oberdiek/embedfile.dtx</code>

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

## 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

**Attached source.** The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is `pdftk`, e.g. unpack the file into the current directory:

```
pdftk embedfile.pdf unpack_files output .
```

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

**plain-`TEX`:** Run `docstrip` and extract the files.

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

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

```
latex \let\install=y\input{embedfile.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

**Generating the documentation.** You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfL<sup>A</sup>T<sub>E</sub>X:

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

## 5 References

- [1] Scott Pakin: *The attachfile package*; 2005/02/20 v1.2; [CTAN:macros/latex/contrib/attachfile/](http://ctan.org/ctan/ctan-macros/latex/contrib/attachfile/).
- [2] Heiko Oberdiek: *The attachfile2 package*; 2006/08/16 v2.2; [CTAN:macros/latex/contrib/oberdiek/attachfile2.pdf](http://ctan.org/ctan/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](http://www.adobe.com/devnet/pdf/pdf_reference.html).
- [4] Network Working Group: RFC 2046, *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*, November 1996; <http://www.rfc-editor.org/>.
- [5] IANA (Internet Assigned Numbers Authority): *MIME Media Types*, May 2006; <http://www.iana.org/assignments/media-types/>.

## 6 History

### [2006/08/16 v1.0]

- First public version.

### [2007/04/11 v1.1]

- Line ends sanitized.

### [2007/09/09 v1.2]

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

### [2007/10/28 v2.0]

- Collection support added (PDF 1.7).

### [2007/10/29 v2.1]

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

### [2007/11/11 v2.2]

- Use of package `pdftexcmds` for L<sup>A</sup>T<sub>E</sub>X support.

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

## 7 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols			
<code>\#</code> .....	868, 930	<code>\csname</code> <i>120, 128, 154, 170, 177, 210,</i>	
<code>\%</code> .....	933	<i>235, 246, 271, 275, 286, 287,</i>	
<code>\:</code> .....	<i>318, 319, 320, 321</i>	<i>288, 298, 326, 328, 335, 336,</i>	
<code>\@</code> .....	869, 926	<i>346, 347, 352, 353, 357, 359,</i>	
<code>\@PackageError</code> .....	219, 296	<i>382, 388, 394, 397, 402, 460,</i>	
<code>\@PackageWarningNoLine</code> .....	805	<i>505, 516, 525, 531, 608, 611,</i>	
<code>\@ehc</code> .....	<i>296, 425, 546, 572, 616</i>	<i>714, 715, 717, 721, 723, 725,</i>	
<code>\@firstofone</code> .....	877, 880	<i>859, 870, 873, 876, 879, 918, 940</i>	
<code>\@firstoftwo</code> .....	<i>391, 394</i>		
<code>\@gobble</code> .....	874, 882	<b>D</b>	
<code>\@ifnch</code> .....	<i>303, 305, 322</i>	<code>\define@key</code> <i>356, 376, 407, 501, 512,</i>	
<code>\@ifnextchar</code> .....	<i>298, 632</i>	<i>523, 529, 534, 576, 602, 620, 623</i>	
<code>\@let@token</code> .....	<i>303, 306, 309, 322</i>	<code>\do</code> .....	<i>495, 498, 664, 742, 744,</i>
<code>\@namedef</code> .....	<i>326</i>	<i>749, 750, 754, 755, 761, 765, 784</i>	
<code>\@ne</code> .....	<i>276, 284, 333</i>	<code>\documentclass</code> .....	<i>41, 974</i>
<code>\@secondoftwo</code> .....	<i>389, 397</i>	<b>E</b>	
<code>\@sptoken</code> .....	<i>306, 318, 319</i>	<code>\EdefSanitize</code> .....	
<code>\@undefined</code> .....	<i>162, 366</i>	<i>246, 408, 459, 525, 535, 577, 606</i>	
<code>\@xifnch</code> .....	<i>307, 320</i>	<code>\embedfile</code> .....	<i>3, 16,</i>
<code>\[</code> .....	931	<i>21, 25, 79, 85, 91, 105, 631, 637, 945</i>	
<code>\]</code> .....	<i>650, 927</i>	<code>\embedfilefield</code> .....	
<code>\{</code> .....	<i>866, 928</i>	<i>4, 50, 54, 58, 62, 66, 434, 450</i>	
<code>\}</code> .....	<i>867, 929</i>	<code>\embedfilefinish</code> .....	<i>3,</i>
<code>\]</code> .....	<i>932</i>	<i>34, 434, 453, 637, 772, 861, 965</i>	
<code>\_</code> .....	<i>934</i>	<code>\embedfilegetobject</code> .....	<i>5, 400, 958</i>
<b>A</b>		<code>\embedfileifobjectexists</code> .....	<i>5, 387, 401, 956</i>
<code>\advance</code> .....	<i>463, 907, 915</i>	<code>\embedfilessetup</code> .....	<i>3, 4, 11, 46, 430</i>
<code>\aftergroup</code> .....	<i>136</i>	<code>\embedfilesort</code> .....	<i>5, 71, 628</i>
<code>\AtEndDocument</code> .....	<i>861</i>	<code>\EmFi@desc</code> .....	<i>659, 661, 693, 695</i>
<b>B</b>		<code>\EmFi@filespec</code> .....	<i>657, 692, 707</i>
<code>\begin</code> .....	<i>75, 975</i>	<code>\EmFi@order</code> .....	<i>447, 485, 825</i>
<code>\body</code> .....	<i>886, 890</i>	<code>\EmFi@cadd</code> .....	<i>706, 738</i>
<code>\bye</code> .....	<i>35, 966</i>	<code>\EmFi@AtEnd</code> ...	<i>190, 191, 230, 241, 863</i>
<b>C</b>		<code>\EmFi@ci</code> .....	<i>670, 701</i>
<code>\catcode</code> <i>113, 114, 115, 116, 117, 118,</i>		<code>\EmFi@collectiontrue</code> ..	<i>410, 458, 797</i>
<i>119, 127, 141, 142, 143, 144,</i>		<code>\EmFi@convert</code> .....	<i>470, 504, 515, 657, 661, 729, 803</i>
<i>145, 146, 147, 148, 149, 150,</i>		<code>\EmFi@DefineKey</code> .....	
<i>151, 152, 153, 174, 175, 178,</i>		<i>355, 361, 362, 363, 364, 365, 429</i>	
<i>179, 180, 181, 185, 186, 187,</i>		<code>\EmFi@defobj</code> .....	<i>380, 684, 705</i>
<i>188, 192, 194, 866, 867, 868,</i>		<code>\EmFi@desc</code> .....	<i>658, 661</i>
<i>869, 904, 913, 926, 927, 928,</i>		<code>\EmFi@details</code> .....	<i>248</i>
<i>929, 930, 931, 932, 933, 934, 935</i>		<code>\EmFi@do</code> .....	<i>664, 713</i>
<code>\chardef</code> .....	<i>272, 276</i>	<code>\EmFi@editfalse</code> .....	<i>468</i>
<code>\count@</code> ...	<i>462, 463, 464, 871, 900,</i>	<code>\EmFi@embedfile</code> .....	<i>632, 634</i>
<i>904, 906, 907, 911, 913, 914, 915</i>		<code>\EmFi@Error</code> .....	
<code>\countdef</code> .....	<i>871</i>	<i>217, 224, 236, 422, 433, 452,</i>	
		<i>542, 570, 595, 613, 636, 651, 774</i>	
		<code>\EmFi@fieldlist</code> ...	<i>448, 496, 497, 667</i>

\EmFi@file	..... 362, 643, 644, 650, 652, 679, 680, 681, 683	596, 606, 607, 610, 614, 731, 732, 813, 815, 820, 844, 845, 847
\EmFi@filespec	..... 657	\EmFi@tile ..... 249
\EmFi@filesystem	..... 688, 690	\EmFi@title ..... 465, 470, 484, 603
\EmFi@finishedtrue	..... 782	\EmFi@type .....
\EmFi@GlobalDefaultKey	.... 349, 567	. 466, 475, 476, 477, 478, 479,
\EmFi@GlobalKey	..... ..... 345, 350, 510, 521, 527, 533	480, 481, 500, 511, 522, 579, 581, 583, 585, 587, 589, 591, 593
\EmFi@hidden	..... 250	\EmFi@view 413, 415, 417, 419, 839, 841
\EmFi@id	..... 377, 382	\EmFi@visibletrue ..... 467
\EmFi@idtrue	..... 378	\empty ..... 123, 124, 412, 548, 554, 658, 659, 674, 688, 693, 737, 741, 780, 802, 832, 844
\EmFi@initialfile	..... ..... 644, 802, 803, 806, 837	\end ..... 97, 941, 977
\EmFi@initialfilefalse	..... 809	\endcsname .....
\EmFi@initialfiletrue	..... 645	. 120, 128, 154, 170, 177, 210, 235, 246, 271, 275, 286, 287, 288, 298, 326, 328, 335, 336, 346, 347, 352, 353, 357, 359, 382, 388, 394, 397, 402, 460, 506, 517, 525, 531, 608, 611, 714, 715, 717, 721, 723, 725, 859, 870, 873, 876, 879, 918, 940
\EmFi@itemtrue	.... 502, 513, 524, 530	\endinput ..... 136, 231, 242
\EmFi@key	..... 459, 460, 465, 473, 498, 501, 510, 512, 521, 523, 527, 529, 533, 534, 567, 571	\errmessage ..... 960
\EmFi@list	..... 737, 741, 742, 758, 765, 768, 780, 789	F
\EmFi@mimetype	..... 674, 676	\futurelet ..... 303, 322
\EmFi@next	409, 421, 427, 432, 439, 443	G
\EmFi@order	... 446, 462, 464, 485, 826	\gdef ..... 321, 446, 556
\EmFi@plain	..... 272, 276, 284, 333	\Gin@driver ..... 5
\EmFi@RequirePackage	..... ..... 209, 217, 221, 233, 244, 332	I
\EmFi@S@ascending	.... 259, 536, 544	\ifcase ..... 812
\EmFi@S@creationdate	.. 257, 480, 590	\ifEmFi@collection ..... 263, 799
\EmFi@S@date	..... 252, 475, 511, 580	\ifEmFi@edit ..... 267, 490
\EmFi@S@desc	..... 255, 478, 586	\ifEmFi@finished 269, 431, 451, 635, 773
\EmFi@S@descending	.... 260, 538, 545	\ifEmFi@id ..... 270, 381
\EmFi@S@details	..... 413, 414, 415	\ifEmFi@initialfile 264, 796, 800, 836
\EmFi@S@false	..... 262, 610	\ifEmFi@item ..... 268, 663, 700
\EmFi@S@file	..... 254, 477, 584	\ifEmFi@sort ..... 265
\EmFi@S@hidden	..... 418, 419, 841	\ifEmFi@visible ..... 266, 486
\EmFi@S@moddate	..... 256, 479, 588	\ifnum 284, 333, 730, 745, 826, 906, 914
\EmFi@S@number	.... 253, 476, 522, 582	\ifpdf ..... 222
\EmFi@S@size	..... 258, 481, 592	\ifx ..... 121, 124, 128, 154, 162, 165, 210, 235, 271, 275, 298, 306, 309, 326, 366, 369, 388, 394, 397, 412, 414, 416, 418, 460, 475, 476, 477, 478, 479, 480, 481, 500, 511, 522, 536, 538, 548, 554, 578, 580, 582, 584, 586, 588, 590, 592, 607, 610, 644, 649, 658, 674, 688, 693, 714, 715, 721, 741, 780, 802, 832, 839, 841, 844, 859, 870, 873, 876, 879, 918
\EmFi@S@text	..... 251, 466, 500, 578	\immediate .....
\EmFi@S@tile	..... 416, 417, 839	. 130, 156, 665, 672, 685, 787, 830
\EmFi@S@true	..... 261, 607	\input ..... 4, 6, 7, 212, 919, 944
\EmFi@schema	.. 445, 471, 472, 832, 834	\iterate ..... 887, 889, 891
\EmFi@setboolean	.... 605, 621, 624	J
\EmFi@sortcase	.... 449, 556, 563, 812	\jobname 95, 103, 108, 109, 970, 971, 973
\EmFi@sortkeys	550, 551, 626, 816, 821	
\EmFi@sortorders	..... . 554, 555, 558, 559, 627, 817, 822	
\EmFi@stringmethod	..... 730	
\EmFi@temp	..... 245, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 273, 279, 280, 281, 282, 283, 285, 290, 291, 292, 293, 294, 334, 338, 339, 340, 341, 342, 408, 412, 414, 416, 418, 423, 503, 504, 507, 514, 515, 518, 535, 536, 537, 538, 539, 541, 543, 548, 555, 561, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593,	

<b>K</b>	
\key . . . . .	740, 742, 745, 749, 765
\KV@errx . . . . .	295
<b>L</b>	
\LaTeX . . . . .	976
\LoadCommand . . . . .	919, 936
\loop . . . . .	885, 901, 912
<b>M</b>	
\MessageBreak . . . . .	
. . . . .	423, 543, 596, 597, 614, 806
<b>N</b>	
\NeedsTeXFormat . . . . .	40, 101, 969
\newif . . . . .	263,
. . . . .	264, 265, 266, 267, 268, 269, 270
\next . . . . .	891, 893, 895
\nopagenumbers . . . . .	951
<b>P</b>	
\PackageInfo . . . . .	133
\par . . . . .	955
\pdf@escapehex . . . . .	740
\pdf@escapename . . . . .	
. . . . .	473, 552, 676, 690, 717, 720
\pdf@escapestring . . . . .	734
\pdf@filemdfivesum . . . . .	681
\pdf@filemoddate . . . . .	679
\pdf@filesize . . . . .	650, 680
\pdf@strcmp . . . . .	730, 745
\pdfcatalog . . . . .	851
\pdflastobj . . . . .	383, 670, 698, 708, 793, 852
\pdfnames . . . . .	792
\pdfobj . . . . .	665, 672, 685, 787, 830
\pdfstringdef . . . . .	43, 366, 369, 731
\ProvidesPackage . . . . .	102, 125, 171
<b>R</b>	
\RangeCatcodeInvalid . . . . .	
. . . . .	910, 922, 923, 924, 925
<b>S</b>	
\SavedJobname . . . . .	970, 973
\setkeys . . . . .	440, 469, 629, 647
\space . . . . .	383, 434, 453,
. . . . .	560, 637, 698, 701, 708, 793, 852
\stop . . . . .	757, 764, 768
<b>T</b>	
\Test . . . . .	921, 939, 954, 963, 964
\TeX . . . . .	952
\texttt . . . . .	976
\the . . . . .	178, 179, 180, 181, 192,
. . . . .	383, 464, 670, 698, 708, 748,
. . . . .	755, 758, 761, 765, 793, 852, 904
\TMP@EnsureCode . . . . .	189,
. . . . .	196, 197, 198, 199, 200, 201,
. . . . .	202, 203, 204, 205, 206, 207, 208
\toks@ . . . . .	747, 748, 755, 758, 761, 765, 767
\tt . . . . .	952
<b>U</b>	
\usepackage . . . . .	42, 45
<b>W</b>	
\write . . . . .	130, 156
<b>X</b>	
\x . . . . .	120, 121, 124, 129, 133, 135,
. . . . .	155, 160, 170, 176, 184, 746, 753
<b>Z</b>	
\z@ . . . . .	272
<b>repeat . . . . .</b>	
<b>RequirePackage . . . . .</b>	
<b>reserved@a . . . . .</b>	
<b>reserved@b . . . . .</b>	
<b>reserved@c . . . . .</b>	
<b>reserved@d . . . . .</b>	
<b>resetatcatcode . . . . .</b>	
<b>RestoreCatcodes . . . . .</b>	