

The embedfile package

Heiko Oberdiek*

2023-01-12 v2.12

Abstract

This package embeds files to a PDF document. Currently the only supported drivers are pdfTeX \geq 1.30 and luaTeX in PDF mode.

Contents

1	Documentation	2
1.1	Introduction	2
1.1.1	Future development	2
1.2	User interface	2
1.3	Collection support (PDF 1.7)	4
1.4	Export of object references	5
1.4.1	Example	5
1.5	Examples	6
1.5.1	plain TeX	6
1.5.2	Collection example	6
1.6	Package dtx-attach	7
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	Installation	22
3.1	Download	22
3.2	Bundle installation	23
3.3	Package installation	23
3.4	Refresh file name databases	23
3.5	Some details for the interested	23
4	References	24
5	History	24
	[2006/08/16 v1.0]	24
	[2007/04/11 v1.1]	24
	[2007/09/09 v1.2]	24
	[2007/10/28 v2.0]	24
	[2007/10/29 v2.1]	24
	[2007/11/11 v2.2]	24

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

[2007/11/25 v2.3]	24
[2009/09/25 v2.4]	24
[2010/03/01 v2.5]	25
[2011/04/13 v2.6]	25
[2016/05/15 v2.7]	25
[2018/11/01 v2.8]	25
[2019/12/03 v2.9]	25
[2020-04-14 v2.10]	25
[2020-04-24 v2.11]	25
[2023-01-12 v2.12]	25

6 Index **25**

1 Documentation

1.1 Introduction

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

1.1.1 Future development

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

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

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

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

1.2 User interface

This package `embedfile` can be used with both L^AT_EX and plain T_EX. See [subsection 1.5.1](#) that explains the use with plain T_EX by an example. In L^AT_EX the package is loaded as usually. There are no options.

```
\usepackage{embedfile}
```

`\embedfile [options] {file}`

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

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

filespec This allows to override the file name that appears in the PDF file. If you are using other than simple file names (8-bit, path separators, ...), look into the PDF specification ([3]). There are rules how these file names must be written/encoded. Avoid 8-bit characters and other special characters, the behaviour is currently undefined. Use option `ucfilespec` for more funny file names. The string method, see below, is `escape` since version 2.4.

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

ucfilespec Since PDF 1.7 the file name may be provided in Unicode. It must be provided for PDF/A-3. By default the `filespec` is used. The conversion of the option value into a PDF string is controlled by option `stringmethod`. Non-ascii chars can look funny if `hyperref` is not loaded!

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.

afrelationship This adds the `/AFRelationship` key to the `filespec` dictionary. The value is a pdf name with or without the leading slash. Typical values are `Source`, `Data`, `Alternative`, `Schema` or `Unspecified`. Mandatory for PDF/A-3.

stringmethod The package must convert the values of the keys `ucfilespec` and `desc` into a PDF string (before version 2.4: `filespec` and `desc`). If `hyperref` is found, then its `\pdfstringdef` will be used, otherwise pdf \TeX 's `\pdfescapestring` is used. Value `psd` forces the use of `\pdfstringdef`, value `escape` the use of `\pdfescapestring`.

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

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

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

`\embedfilefinish`

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

`\embedfilesetup {options}`

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

1.3 Collection support (PDF 1.7)

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

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

Name	Description	Modified	Size
...

Acrobat Reader 10 shows the embeded 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` $\{\langle options \rangle\}$

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

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

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

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

hidden The collection table is not shown.

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

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

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

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

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

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

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

file The file name of the embedded file.

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

moddate The modification date of the embedded file.

size The size of the embedded file.

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

title sets the column title.

visible controls whether the column is presented:

true shows the column.

false hides the column.

Default: **true**

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

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

false disables the feature.

Default: **false**

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

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

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

1.4 Export of object references

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

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

- `EmbeddedFile`
- `Filespec`

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

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

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

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

1.4.1 Example

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

1.5 Examples

1.5.1 plain T_EX

The package can be used with plain T_EX. It can be used with or without help from `miniltx.tex`.

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

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

```
1 (*exampleplain)
2 %<<END
3 % Load packages
4 \input miniltx
5 % \def\Gin@driver{pdftex.def}
6 % \input graphicx.sty
7 \input embedfile.sty
8 \resetatcatcode
9
10 % default setting
11 \embedfilesetup{
12   mimetype=text/plain
13 }
14
15 % Embed files
16 \embedfile[
17   filespec=example.tex,
18   desc={Source code (plain-TeX) of this example}
19 ]{embedfile-example-plain.tex}
20
21 \embedfile[
22   desc={Source of package 'embedfile'}
23 ]{embedfile.dtx}
24
25 \embedfile[
26   mimetype=application/pdf,
27   desc={Documentation of package 'embedfile'}
28 ]{embedfile.pdf}
29
30 % Some text
31 This example document contains three embedded files.
32
33 % End of document
34 \embedfilefinish % don't forget
35 \bye
36 %END
37 </exampleplain)
```

1.5.2 Collection example

```
38 (*examplecollection)
39 %<<END
40 \NeedsTeXFormat{LaTeX2e}
41 \documentclass{article}
42 \usepackage[bookmarks=false]{hyperref}
43 % provides \pdfstringdef that is then used by 'title' and
44 % other keys.
45 \usepackage{embedfile}[2019/12/03]
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:pkg/oberdiek](#). It also serves as small example for the use of the package with \LaTeX .

```

100 (*dtxattach)
101 \NeedsTeXFormat{LaTeX2e}
102 \ProvidesPackage{dtx-attach}
103 [2023-01-12 v2.12 Embed \string\jobname.dtx (HO)]%
104 \RequirePackage{embedfile}[2019/12/03]
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 [2023-01-12 v2.12 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 \ifcsname EmFi@Error\endcsname
234 \else
235   \def\EmFi@Error{%
236     \@PackageError{embedfile}%
237   }
238 \fi

```

Luatex compat

```

239 \ifx\pdfextension\undefined\else
240   \protected\def\pdflastobj {\numexpr\pdffeedback lastobj\relax}
241   \protected\def\pdfnames   {\pdfextension names }
242   \protected\def\pdfobj     {\pdfextension obj }
243   \protected\def\pdfcatalog {\pdfextension catalog }
244   \let\pdfoutput           \outputmode
245 \fi

```

2.4 Check for recent pdfTeX in PDF mode

Load package iftex and check mode.

```

246 \EmFi@RequirePackage{iftex}[2019/11/07]
247 \ifpdf
248 \else
249   \EmFi@Error{%
250     Missing pdfTeX or luaTeX in PDF mode%
251   }{%
252     Currently other drivers are not supported. %
253     Package loading is aborted.%
254   }%
255   \expandafter\EmFi@AtEnd
256 \fi%
257 \EmFi@RequirePackage{pdftexcmds}[2007/11/11]
258 \EmFi@RequirePackage{ltxcmds}[2010/03/01]
259 \EmFi@RequirePackage{kvsetkeys}[2010/03/01]
260 \EmFi@RequirePackage{kvdefinekeys}[2010/03/01]

```

Check version.

```

261 \begingroup\expandafter\expandafter\expandafter\endgroup
262 \expandafter\ifx\csname pdf@filesize\endcsname\relax
263   \EmFi@Error{%
264     Unsupported pdfTeX version%
265   }{%
266     At least version 1.30 is necessary. Package loading is aborted.%
267   }%
268   \expandafter\EmFi@AtEnd
269 \fi%

```

2.5 Strings

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

```
270 \EmFi@RequirePackage{pdfescape}[2007/11/11]
271 \def\EmFi@temp#1{%
272   \expandafter\EdefSanitize\csname EmFi@S@#1\endcsname{#1}%
273 }
```

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

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

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

```
\EmFi@S@text
277 \EmFi@temp{text}
```

```
\EmFi@S@date
278 \EmFi@temp{date}
```

```
\EmFi@S@number
279 \EmFi@temp{number}
```

```
\EmFi@S@file
280 \EmFi@temp{file}
```

```
\EmFi@S@desc
281 \EmFi@temp{desc}
```

```
\EmFi@S@afrelationship
282 \EmFi@temp{afrelationship}
```

```
\EmFi@S@moddate
283 \EmFi@temp{moddate}
```

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

```
\EmFi@S@size
285 \EmFi@temp{size}
```

```
\EmFi@S@ascending
286 \EmFi@temp{ascending}
```

```
\EmFi@S@descending
287 \EmFi@temp{descending}
```

```
\EmFi@S@true
288 \EmFi@temp{true}
```

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

2.6 Switches

```
\ifEmFi@collection
290 \ltx@newif\ifEmFi@collection

\ifEmFi@sort
291 \ltx@newif\ifEmFi@sort

\ifEmFi@visible
292 \ltx@newif\ifEmFi@visible

\ifEmFi@edit
293 \ltx@newif\ifEmFi@edit

\ifEmFi@item
294 \ltx@newif\ifEmFi@item

\ifEmFi@finished
295 \ltx@newif\ifEmFi@finished

\ifEmFi@id
296 \ltx@newif\ifEmFi@id
```

2.7 Key value definitions

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

\EmFi@GlobalDefaultKey
301 \def\EmFi@GlobalDefaultKey#1#2{%
302   \EmFi@GlobalKey{#1}{#2}%
303   \global\expandafter\let
304   \csname KV@#1@#2@default\expandafter\endcsname
305   \csname KV@#1@#2@default\endcsname
306 }

\EmFi@DefineKey
307 \def\EmFi@DefineKey#1#2{%
308   \kv@define@key{EmFi}{#1}{%
309     \expandafter\def\csname EmFi@#1\endcsname{##1}%
310   }%
311   \expandafter\def\csname EmFi@#1\endcsname{#2}%
312 }

Subtype of the embedded file (optional).
313 \EmFi@DefineKey{mimetype}{}

File specification string.
314 \EmFi@DefineKey{filespec}{\EmFi@file}

File specification string in Unicode.
315 \EmFi@DefineKey{ucfilespec}{}

File system (optional).
316 \EmFi@DefineKey{filesystem}{}

Description (optional).
317 \EmFi@DefineKey{desc}{}


```

AFRelationship (mandatory for PDF/A-3 compliance).

```
318 \EmFi@DefineKey{afrelationship}{}
```

Method for converting text to PDF strings.

```
319 \EmFi@DefineKey{stringmethod}{%
320   \ifx\pdfstringdef\@undefined
321     escape%
322   \else
323     \ifx\pdfstringdef\relax
324       escape%
325     \else
326       psd%
327     \fi
328   \fi
329 }
```

Option id as key for object numbers.

```
330 \kv@define@key{EmFi}{id}{%
331   \def\EmFi@id#1}%
332   \EmFi@idtrue
333 }
```

\EmFi@defobj

```
334 \def\EmFi@defobj#1{%
335   \ifEmFi@id
336     \expandafter\xdef\csname EmFi@#1\EmFi@id\endcsname{%
337       \the\pdflastobj\ltx@space 0 R%
338     }%
339   \fi
340 }
```

\embedfileifobjectexists

```
341 \def\embedfileifobjectexists#1#2{%
342   \expandafter\ifx\csname EmFi@#2@#1\endcsname\relax
343     \expandafter\ltx@secondoftwo
344   \else
345     \expandafter\ltx@firstoftwo
346   \fi
347 }
```

\embedfilegetobject

```
348 \def\embedfilegetobject#1#2{%
349   \embedfileifobjectexists{#1}{#2}{%
350     \csname EmFi@#2@#1\endcsname
351   }{%
352     0 0 R%
353   }%
354 }
```

Initial view of the collection.

```
355 \kv@define@key{EmFi}{view}[]{%
356   \EdefSanitize\EmFi@temp{#1}%
357   \def\EmFi@next{%
358     \global\EmFi@collectiontrue
359   }%
360   \ifx\EmFi@temp\ltx@empty
361     \let\EmFi@view\EmFi@S@details
362   \else\ifx\EmFi@temp\EmFi@S@details
363     \let\EmFi@view\EmFi@S@details
364   \else\ifx\EmFi@temp\EmFi@S@tile
365     \let\EmFi@view\EmFi@S@tile
366   \else\ifx\EmFi@temp\EmFi@S@hidden
367     \let\EmFi@view\EmFi@S@hidden
```

```

368 \else
369 \let\EmFi@next\relax
370 \EmFi@Error{%
371     Unknown value '\EmFi@temp' for key 'view'.\MessageBreak
372     Supported values: 'details', 'tile', 'hidden'.%
373 } \@ehc
374 \fi\fi\fi\fi
375 \EmFi@next
376 }
377 \EmFi@DefineKey{initialfile}{}}

\embedfilesetup
378 \def\embedfilesetup{%
379 \ifEmFi@finished
380 \def\EmFi@next##1{%
381 \EmFi@Error{%
382 \string\embedfilefield\ltx@space after \string\embedfilefinish
383 }{%
384 The list of embedded files is already written.%
385 }%
386 \else
387 \def\EmFi@next{%
388 \kvsetkeys{EmFi}%
389 }%
390 \fi
391 \EmFi@next
392 }

\EmFi@schema
393 \def\EmFi@schema{}

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

\EmFi@@order
395 \let\EmFi@@order\relax

\EmFi@fieldlist
396 \def\EmFi@fieldlist{}

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

\embedfilefield
398 \def\embedfilefield#1#2{%
399 \ifEmFi@finished
400 \EmFi@Error{%
401 \string\embedfilefield\ltx@space after \string\embedfilefinish
402 }{%
403 The list of embedded files is already written.%
404 }%
405 \else
406 \global\EmFi@collectiontrue
407 \EdefSanitize\EmFi@key{#1}%
408 \expandafter\ifx\cname KV@EmFi@\EmFi@key.prefix\endcsname\relax
409 \begingroup
410 \count@=\EmFi@order
411 \advance\count@ 1 %
412 \xdef\EmFi@order{\the\count@}%
413 \let\EmFi@title\EmFi@key
414 \let\EmFi@type\EmFi@S@text

```

```

415 \EmFi@visibletrue
416 \EmFi@editfalse
417 \kvsetkeys{EmFiFi}{#2}%
418 \EmFi@convert\EmFi@title\EmFi@title
419 \xdef\EmFi@schema{%
420 \EmFi@schema
421 /\pdf@escapename{\EmFi@key}<<%
422 /Subtype/%
423 \ifx\EmFi@type\EmFi@S@date D%
424 \else\ifx\EmFi@type\EmFi@S@number N%
425 \else\ifx\EmFi@type\EmFi@S@file F%
426 \else\ifx\EmFi@type\EmFi@S@desc Desc%
427 \else\ifx\EmFi@type\EmFi@S@afrelationship AFRelationship%
428 \else\ifx\EmFi@type\EmFi@S@moddate ModDate%
429 \else\ifx\EmFi@type\EmFi@S@creationdate CreationDate%
430 \else\ifx\EmFi@type\EmFi@S@size Size%
431 \else S%
432 \fi\fi\fi\fi\fi\fi\fi\fi
433 /N(\EmFi@title)%
434 \EmFi@@order{\EmFi@order}%
435 \ifEmFi@visible
436 \else
437 /V false%
438 \fi
439 \ifEmFi@edit
440 /E true%
441 \fi
442 >>%
443 }%
444 \let\do\relax
445 \xdef\EmFi@fieldlist{%
446 \EmFi@fieldlist
447 \do{\EmFi@key}%
448 }%
449 \ifx\EmFi@type\EmFi@S@text
450 \kv@define@key{EmFi}{\EmFi@key.value}{%
451 \EmFi@itemtrue
452 \def\EmFi@temp{##1}%
453 \EmFi@convert\EmFi@temp\EmFi@temp
454 \expandafter\def\csname EmFi@V@#1%
455 \expandafter\endcsname\expandafter{%
456 \expandafter(\EmFi@temp)%
457 }%
458 }%
459 \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
460 \else\ifx\EmFi@type\EmFi@S@date
461 \kv@define@key{EmFi}{\EmFi@key.value}{%
462 \EmFi@itemtrue
463 \def\EmFi@temp{##1}%
464 \EmFi@convert\EmFi@temp\EmFi@temp
465 \expandafter\def\csname EmFi@V@#1%
466 \expandafter\endcsname\expandafter{%
467 \expandafter(\EmFi@temp)%
468 }%
469 }%
470 \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
471 \else\ifx\EmFi@type\EmFi@S@number
472 \kv@define@key{EmFi}{\EmFi@key.value}{%
473 \EmFi@itemtrue
474 \expandafter\edef\csname EmFi@V@#1\endcsname{ ##1}%
475 }%
476 \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%

```

```

477     \fi\fi\fi
478     \kv@define@key{EmFi}{\EmFi@key.prefix}{%
479         \EmFi@itemtrue
480         \expandafter\def\csname EmFi@P@#1\endcsname{##1}%
481     }%
482     \EmFi@GlobalKey{EmFi}{\EmFi@key.prefix}%
483     \kv@define@key{EmFiSo}{\EmFi@key}[ascending]{%
484         \EdefSanitize\EmFi@temp{##1}%
485         \ifx\EmFi@temp\EmFi@S@ascending
486             \def\EmFi@temp{true}%
487         \else\ifx\EmFi@temp\EmFi@S@descending
488             \def\EmFi@temp{false}%
489         \else
490             \def\EmFi@temp{}%
491         \EmFi@Error{%
492             Unknown sort order ‘\EmFi@temp’.\MessageBreak
493             Supported values: ‘\EmFi@S@ascending’, %
494             ‘\EmFi@S@descending
495         }\@ehc
496     \fi\fi
497     \ifx\EmFi@temp\ltx@empty
498     \else
499         \xdef\EmFi@sortkeys{%
500             \EmFi@sortkeys
501             /\pdf@escapename{#1}%
502         }%
503         \ifx\EmFi@sortorders\ltx@empty
504             \global\let\EmFi@sortorders\EmFi@temp
505             \gdef\EmFi@sortcase{1}%
506         \else
507             \xdef\EmFi@sortorders{%
508                 \EmFi@sortorders
509                 \ltx@space
510                 \EmFi@temp
511             }%
512             \xdef\EmFi@sortcase{2}%
513         \fi
514     \fi
515     }%
516     \EmFi@GlobalDefaultKey{EmFiSo}\EmFi@key
517 \endgroup
518 \else
519     \EmFi@Error{%
520         Field ‘\EmFi@key’ is already defined%
521     }\@ehc
522 \fi
523 \fi
524 }

525 \kv@define@key{EmFiFi}{type}{%
526     \EdefSanitize\EmFi@temp{#1}%
527     \ifx\EmFi@temp\EmFi@S@text
528         \let\EmFi@type\EmFi@temp
529     \else\ifx\EmFi@temp\EmFi@S@date
530         \let\EmFi@type\EmFi@temp
531     \else\ifx\EmFi@temp\EmFi@S@number
532         \let\EmFi@type\EmFi@temp
533     \else\ifx\EmFi@temp\EmFi@S@file
534         \let\EmFi@type\EmFi@temp
535     \else\ifx\EmFi@temp\EmFi@S@desc
536         \let\EmFi@type\EmFi@temp
537     \else\ifx\EmFi@temp\EmFi@S@afrelationship
538         \let\EmFi@type\EmFi@temp

```



```

539 \else\ifx\EmFi@temp\EmFi@S@moddate
540 \let\EmFi@type\EmFi@temp
541 \else\ifx\EmFi@temp\EmFi@S@creationdate
542 \let\EmFi@type\EmFi@temp
543 \else\ifx\EmFi@temp\EmFi@S@size
544 \let\EmFi@type\EmFi@temp
545 \else
546 \EmFi@Error{%
547   Unknown type '\EmFi@temp'.\MessageBreak
548   Supported types: 'text', 'date', 'number', 'file',\MessageBreak
549   'desc', 'afrelationship', 'moddate', 'creationdate', 'size'%
550 }%
551 \fi\fi\fi\fi\fi\fi\fi\fi\fi
552 }

553 \kv@define@key{EmFiFi}{title}{%
554 \def\EmFi@title{#1}%
555 }

```

\EmFi@setboolean

```

556 \def\EmFi@setboolean#1#2{%
557 \EdefSanitize\EmFi@temp{#2}%
558 \ifx\EmFi@temp\EmFi@S@true
559 \csname EmFi@#1true\endcsname
560 \else
561 \ifx\EmFi@temp\EmFi@S@false
562 \csname EmFi@#1false\endcsname
563 \else
564 \EmFi@Error{%
565   Unknown value '\EmFi@temp' for key '#1'.\MessageBreak
566   Supported values: 'true', 'false'%
567 } \@ehc
568 \fi
569 \fi
570 }

571 \kv@define@key{EmFiFi}{visible}[true]{%
572 \EmFi@setboolean{visible}{#1}%
573 }

574 \kv@define@key{EmFiFi}{edit}[true]{%
575 \EmFi@setboolean{edit}{#1}%
576 }

```

\EmFi@sortkeys

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

\EmFi@sortorders

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

\embedfilesort

```

579 \def\embedfilesort{%
580 \kvsetkeys{EmFiSo}%
581 }

```

2.8 Embed the file

\embedfile

```

582 \def\embedfile{%
583 \ltx@ifnextchar[\EmFi@embedfile{\EmFi@embedfile[]}%
584 }

```

\EmFi@embedfile

```
585 \def\EmFi@removeslash#1{\if/#1\else#1\fi}%
586 \def\EmFi@embedfile[#1]#2{%
587   \ifEmFi@finished
588     \EmFi@Error{%
589       \string\embedfile\ltx@space after \string\embedfilefinish
590     }{%
591       The list of embedded files is already written.%
592     }%
593   \else
594     \beginngroup
595     \def\EmFi@file{#2}%
596     \kvsetkeys{EmFi}{#1}%
597     \expandafter\expandafter\expandafter
598     \ifx\expandafter\expandafter\expandafter
599       \\pdf@filesize{\EmFi@file}\\%
600     \EmFi@Error{%
601       File '\EmFi@file' not found%
602     }{%
603       The unknown file is not embedded.%
604     }%
605   \else
606     \edef\EmFi@@filespec{%
607       \pdf@escapestring{\EmFi@filespec}%
608     }%
609     \ifx\EmFi@ucfilespec\ltx@empty
610       \EmFi@convert\EmFi@filespec\EmFi@ucfilespec
611     \else
612       \EmFi@convert\EmFi@ucfilespec\EmFi@ucfilespec
613     \fi
614     \ifx\EmFi@desc\ltx@empty
615       \let\EmFi@@desc\ltx@empty
616     \else
617       \EmFi@convert\EmFi@desc\EmFi@@desc
618     \fi
619     \ifx\EmFi@afrelationship\ltx@empty
620       \let\EmFi@@afrelationship\ltx@empty
621     \else
622       \expandafter\edef\expandafter\EmFi@@afrelationship\expandafter
623       {\expandafter\EmFi@removeslash\EmFi@afrelationship}
624     \fi
625     \ifEmFi@item
626       \let\do\EmFi@do
627       \immediate\pdfobjj{%
628         <<%
629           \EmFi@fieldlist
630         >>%
631       }%
632       \edef\EmFi@ci{\the\pdflastobjj}%
633     \fi
634     \immediate\pdfobj stream attr{%
635       /Type/EmbeddedFile%
636       \ifx\EmFi@mimetype\ltx@empty
637       \else
638         /Subtype/\pdf@escapename{\EmFi@mimetype}%
639       \fi
640       /Params<<%
641         /ModDate(\pdf@filemoddate{\EmFi@file})%
642         /Size \pdf@filesize{\EmFi@file}%
643         /Checksum<\pdf@filemdfivesum{\EmFi@file}>%
644       >>%
645     }file{\EmFi@file}\relax
```

```

646     \EmFi@defobj{EmbeddedFile}%
647     \immediate\pdfobj{%
648         <<%
649         /Type/Filespec%
650         \ifx\EmFi@filesystem\ltx@empty
651         \else
652         /FS/\pdf@escapename{\EmFi@filesystem}%
653         \fi
654         /F(\EmFi@filespec)%
655         /UF(\EmFi@ucfilespec)%
656         \ifx\EmFi@desc\ltx@empty
657         \else
658         /Desc(\EmFi@desc)%
659         \fi
660         \ifx\EmFi@afrelationship\ltx@empty
661         \else
662         /AFRelationship/\pdf@escapename{\EmFi@afrelationship}%
663         \fi
664         /EF<<%
665         /F \the\pdflastobj\ltx@space 0 R%
666         >>%
667         \ifEmFi@item
668         /CI \EmFi@ci\ltx@space 0 R%
669         \fi
670     >>%
671 }%
672 \EmFi@defobj{Filespec}%
673 \EmFi@add{%
674     \EmFi@filespec
675 }{\the\pdflastobj\ltx@space 0 R}%
676 \fi
677 \endgroup
678 \fi
679 }

```

\EmFi@do

```

680 \def\EmFi@do#1{%
681     \expandafter\ifx\csname EmFi@P@#1\endcsname\relax
682     \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
683     \else
684     /\pdf@escapename{#1}\csname EmFi@V@#1\endcsname
685     \fi
686     \else
687     /\pdf@escapename{#1}<<%
688     \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
689     \else
690     /D\csname EmFi@V@#1\endcsname
691     \fi
692     /P(\csname EmFi@P@#1\endcsname)%
693     >>%
694     \fi
695 }

```

\EmFi@convert

```

696 \def\EmFi@convert#1#2{%
697     \ifnum\pdf@strcmp{\EmFi@stringmethod}{psd}=0 %
698     \pdfstringdef\EmFi@temp{#1}%
699     \let#2\EmFi@temp
700     \else
701     \edef#2{\pdf@escapestring{#1}}%
702     \fi
703 }

```

```
704 \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, ...).

```
705 \def\EmFi@add#1#2{%
706   \begingroup
707   \ifx\EmFi@list\ltx@empty
708     \xdef\EmFi@list{\noexpand\do{#1}{#2}}%
709   \else
710     \def\do##1##2{%
711       \ifnum\pdf@strcmp{##1}{##2}>0 %
712         \edef\x{%
713           \toks@{%
714             \the\toks@%
715             \noexpand\do{#1}{#2}%
716             \noexpand\do{##1}{##2}%
717           }%
718         }%
719         \x
720       \def\do####1####2{%
721         \toks@\expandafter{\the\toks@\do{####1}{####2}}%
722       }%
723       \def\stop{%
724         \xdef\EmFi@list{\the\toks@}%
725       }%
726     \else
727       \toks@\expandafter{\the\toks@\do{##1}{##2}}%
728     \fi
729   }%
730   \def\stop{%
731     \xdef\EmFi@list{\the\toks@\noexpand\do{#1}{#2}}%
732   }%
733   \toks@{}%
734   \EmFi@list\stop
735 \fi
736 \endgroup
737 }
```

```
\embedfilefinish
```

```
738 \def\embedfilefinish{%
739   \ifEmFi@finished
740     \EmFi@Error{%
741       Too many invocations of \string\embedfilefinish
742     }{%
743       The list of embedded files is already written.%
744     }%
745   \else
746     \ifx\EmFi@list\ltx@empty
747     \else
748       \global\EmFi@finishedtrue
749       \begingroup
750       \def\do##1##2{%
751         (##1)##2%
752       }%
753       \immediate\pdfobj{%
754         <<%
755           /Names[\EmFi@list]%
756         >>%
757       }%
```

```

758     \pdfnames{%
759         /EmbeddedFiles \the\pdflastobj\ltx@space 0 R%
760     }%
761 \endgroup
762 \begingroup
763 \def\do##1##2{%
764     \ltx@space##2%
765 }%
766 \immediate\pdfobj{%
767     [\EmFi@list]%
768 }%
769 \pdfcatalog{%
770     /AF \the\pdflastobj\ltx@space 0 R%
771 }%
772 \endgroup

```

Write collection objects.

```

773 \ifx\EmFi@initialfile\ltx@empty
774 \else
775     \EmFi@collectiontrue
776 \fi
777 \ifEmFi@collection
778     \ifx\EmFi@initialfile\ltx@empty
779         \let\EmFi@@initialfile\ltx@empty
780     \else
781         \edef\EmFi@@initialfile{%
782             \pdf@escapestring{\EmFi@initialfile}%
783         }%
784     \fi

```

Look for initial file among the embedded files.

```

785 \begingroup
786 \let\f=N%
787 \def\do##1##2{%
788     \def\x{##1}%
789     \ifx\x\EmFi@@initialfile
790         \let\f=Y%
791     \let\do\ltx@gobbletwo
792 \fi
793 }%
794 \EmFi@list
795 \expandafter\endgroup
796 \ifx\f Y%
797 \else
798     \@PackageWarningNoLine{embedfile}{%
799         Missing initial file '\EmFi@initialfile'\MessageBreak
800         among the embedded files%
801     }%
802     \let\EmFi@initialfile\ltx@empty
803     \let\EmFi@@initialfile\ltx@empty
804 \fi
805 \ifcase\EmFi@sortcase
806     \def\EmFi@temp{%
807 \or
808     \def\EmFi@temp{%
809         /S\EmFi@sortkeys
810         /A \EmFi@sortorders
811     }%
812 \else
813     \def\EmFi@temp{%
814         /S[\EmFi@sortkeys]%
815         /A[\EmFi@sortorders]%
816     }%
817 \fi

```

```

818 \def\EmFi@@order##1{%
819 \ifnum\EmFi@order>1 %
820 /O ##1%
821 \fi
822 }%
823 \immediate\pdfobj{%
824 <<%
825 \ifx\EmFi@schema\ltx@empty
826 \else
827 /Schema<<\EmFi@schema>>%
828 \fi
829 \ifx\EmFi@initialfile\ltx@empty
830 \else
831 /D(\EmFi@initialfile)%
832 \fi
833 \ifx\EmFi@view\EmFi@S@tile
834 /View/T%
835 \else\ifx\EmFi@view\EmFi@S@hidden
836 /View/H%
837 \fi\fi
838 \ifx\EmFi@temp\ltx@empty
839 \EmFi@temp
840 \else
841 /Sort<<\EmFi@temp>>%
842 \fi
843 >>%
844 }%
845 \pdfcatalog{%
846 /Collection \the\pdflastobj\ltx@space0 R%
847 }%
848 \fi
849 \fi
850 \fi
851 }

852 \begingroup\expandafter\expandafter\expandafter\endgroup
853 \expandafter\ifx\csname AtEndDocument\endcsname\relax
854 \else
855 \AtEndDocument{\embedfilefinish}%
856 \fi
857 \EmFi@AtEnd%
858 </package>

```

3 Installation

3.1 Download

Package. This package is available on CTAN¹:

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

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

Bundle. All the packages of the bundle ‘embedfile’ 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/embedfile.tds.zip](#)

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

¹[CTAN:pkg/embedfile](#)

3.2 Bundle installation

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

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

3.3 Package installation

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

```
tex embedfile.dtx
```

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

```
embedfile.sty           → tex/generic/embedfile/embedfile.sty
dtx-attach.sty         → tex/generic/embedfile/dtx-attach.sty
embedfile.pdf          → doc/latex/embedfile/embedfile.pdf
embedfile-example-plain.tex → doc/latex/embedfile/embedfile-example-plain.tex
embedfile-example-collection.tex → doc/latex/embedfile/embedfile-example-collection.tex
embedfile.dtx          → source/latex/embedfile/embedfile.dtx
```

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

3.4 Refresh file name databases

If your `TEX` distribution (`TEX Live`, `MiKTEX`, ...) relies on file name databases, you must refresh these. For example, `TEX Live` users run `texhash` or `mktextlsr`.

3.5 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
```

4 References

- [1] Scott Pakin: *The attachfile package*; 2005/02/20 v1.2; [CTAN:pkg/attachfile](#).
- [2] Heiko Oberdiek: *The attachfile2 package*; 2006/08/16 v2.2; [CTAN:pkg/attachfile2](#).
- [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/>.

5 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 iniT_EX.
- Package keyval replaced by packages kvsetkeys and kvdefinekeys because of compatibility for iniT_EX.
- 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]

- LuaT_EX compatibility

[2018/11/01 v2.8]

- Remove luatex85 package dependency.

[2019/12/03 v2.9]

- add /AF and /AFrelationship keys (Andreas Karrenbauer)
- add \pdfcatalog emulation for LuaT_EX.
- update to use iftex

[2020-04-14 v2.10]

- Fix issue #4, the value of afrelationship should not be converted but name escaped.

[2020-04-24 v2.11]

- Updated

[2023-01-12 v2.12]

- Allow the error message to be changed for better tex4ht compability (PR#7)
- Adapted the error message to luatex
- The /UF key in the filespec dictionary is now set by default (PR#6)

6 Index

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

Symbols	A
\@PackageError	236
\@PackageWarningNoLine	798
\@ehc	373, 495, 521, 567
\@undefined	168, 239, 320
\ \	599
\advance	411
\aftergroup	139
\AtEndDocument	855
B	
\begin	75

<code>\bye</code>	35	<code>\EmFi@fieldlist</code> ...	396, 445, 446, 629
C		<code>\EmFi@file</code>	314, 595, 599, 601, 641, 642, 643, 645
<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	<code>\EmFi@filespec</code>	607, 610
<code>\count@</code>	410, 411, 412	<code>\EmFi@filesystem</code>	650, 652
<code>\csname</code>	124, 131, 160, 176, 186, 225, 262, 272, 298, 299, 304, 305, 309, 311, 336, 342, 350, 408, 454, 465, 474, 480, 559, 562, 681, 682, 684, 688, 690, 692, 853	<code>\EmFi@finishedtrue</code>	748
D		<code>\EmFi@GlobalDefaultKey</code>	301, 516
<code>\do</code>	444, 447, 626, 708, 710, 715, 716, 720, 721, 727, 731, 750, 763, 787, 791	<code>\EmFi@GlobalKey</code>	297, 302, 459, 470, 476, 482
<code>\documentclass</code>	41	<code>\EmFi@hidden</code>	276
E		<code>\EmFi@id</code>	331, 336
<code>\EdefSanitize</code>	272, 356, 407, 474, 484, 526, 557	<code>\EmFi@idtrue</code>	332
<code>\embedfile</code>	3, 16, 21, 25, 79, 85, 91, 105, 582, 589	<code>\EmFi@initialfile</code>	773, 778, 782, 799, 802
<code>\embedfilefield</code>	4, 50, 54, 58, 62, 66, 382, 398	<code>\EmFi@itemtrue</code>	451, 462, 473, 479
<code>\embedfilefinish</code>	3, 34, 382, 401, 589, 738, 855	<code>\EmFi@key</code>	407, 408, 413, 421, 447, 450, 459, 461, 470, 472, 476, 478, 482, 483, 516, 520
<code>\embedfilegetobject</code>	5, 348	<code>\EmFi@list</code>	704, 707, 708, 724, 731, 734, 746, 755, 767, 794
<code>\embedfileifobjectexists</code> .	5, 341, 349	<code>\EmFi@mimetype</code>	636, 638
<code>\embedfilesetup</code>	3, 4, 11, 46, 378	<code>\EmFi@next</code>	357, 369, 375, 380, 387, 391
<code>\embedfilesort</code>	5, 71, 579	<code>\EmFi@order</code> ...	394, 410, 412, 434, 819
<code>\EmFi@@afrelationship</code>	620, 622, 660, 662	<code>\EmFi@removeslash</code>	585, 623
<code>\EmFi@@desc</code>	615, 617, 656, 658	<code>\EmFi@RequirePackage</code>	224, 232, 246, 257, 258, 259, 260, 270
<code>\EmFi@@filespec</code>	606, 654, 674	<code>\EmFi@S@afrelationship</code>	282, 427, 537
<code>\EmFi@@initialfile</code>	779, 781, 789, 803, 829, 831	<code>\EmFi@S@ascending</code>	286, 485, 493
<code>\EmFi@@order</code>	395, 434, 818	<code>\EmFi@S@creationdate</code> ..	284, 429, 541
<code>\EmFi@@ucfilespec</code>	610, 612, 655	<code>\EmFi@S@date</code>	278, 423, 460, 529
<code>\EmFi@add</code>	673, 705	<code>\EmFi@S@desc</code>	281, 426, 535
<code>\EmFi@afrelationship</code>	619, 623	<code>\EmFi@S@descending</code>	287, 487, 494
<code>\EmFi@AtEnd</code>	205, 206, 223, 255, 268, 857	<code>\EmFi@S@details</code>	361, 362, 363
<code>\EmFi@ci</code>	632, 668	<code>\EmFi@S@false</code>	289, 561
<code>\EmFi@collectiontrue</code> ..	358, 406, 775	<code>\EmFi@S@file</code>	280, 425, 533
<code>\EmFi@convert</code>	418, 453, 464, 610, 612, 617, 696	<code>\EmFi@S@hidden</code>	366, 367, 835
<code>\EmFi@DefineKey</code>	307, 313, 314, 315, 316, 317, 318, 319, 377	<code>\EmFi@S@moddate</code>	283, 428, 539
<code>\EmFi@defobj</code>	334, 646, 672	<code>\EmFi@S@number</code>	279, 424, 471, 531
<code>\EmFi@desc</code>	614, 617	<code>\EmFi@S@size</code>	285, 430, 543
<code>\EmFi@details</code>	274	<code>\EmFi@S@text</code>	277, 414, 449, 527
<code>\EmFi@do</code>	626, 680	<code>\EmFi@S@tile</code>	364, 365, 833
<code>\EmFi@editfalse</code>	416	<code>\EmFi@S@true</code>	288, 558
<code>\EmFi@embedfile</code>	583, 585	<code>\EmFi@schema</code> ..	393, 419, 420, 825, 827
<code>\EmFi@Error</code>	232, 249, 263, 370, 381, 400, 491, 519, 546, 564, 588, 600, 740	<code>\EmFi@setboolean</code>	556, 572, 575
		<code>\EmFi@sortcase</code>	397, 505, 512, 805
		<code>\EmFi@sortkeys</code>	499, 500, 577, 809, 814
		<code>\EmFi@sortorders</code>	503, 504, 507, 508, 578, 810, 815
		<code>\EmFi@stringmethod</code>	697
		<code>\EmFi@temp</code>	271, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 356, 360, 362, 364, 366, 371, 452, 453, 456, 463, 464, 467, 484, 485, 486, 487, 488, 490, 492, 497, 504, 510, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 547, 557, 558, 561, 565, 698, 699, 806, 808, 813, 838, 839, 841
		<code>\EmFi@tile</code>	275

<code>\EmFi@title</code>	413, 418, 433, 554		
<code>\EmFi@type</code>	414, 423, 424, 425, 426, 427, 428, 429, 430, 449, 460, 471, 528, 530, 532, 534, 536, 538, 540, 542, 544		
<code>\EmFi@ucfilespec</code>	609, 612		
<code>\EmFi@view</code>	361, 363, 365, 367, 833, 835		
<code>\EmFi@visibletrue</code>	415		
<code>\empty</code>	127, 128		
<code>\end</code>	97		
<code>\endcsname</code>	124, 131, 160, 176, 186, 225, 233, 262, 272, 298, 299, 304, 305, 309, 311, 336, 342, 350, 408, 455, 466, 474, 480, 559, 562, 681, 682, 684, 688, 690, 692, 853		
<code>\endinput</code>	139, 223		
<code>\endlinechar</code>	114, 145, 181, 187, 199		
F			
<code>\f</code>	786, 790, 796		
G			
<code>\gdef</code>	394, 505		
<code>\Gin@driver</code>	5		
I			
<code>\if</code>	585		
<code>\ifcase</code>	805		
<code>\ifcsname</code>	233		
<code>\ifEmFi@collection</code>	290, 777		
<code>\ifEmFi@edit</code>	293, 439		
<code>\ifEmFi@finished</code>	295, 379, 399, 587, 739		
<code>\ifEmFi@id</code>	296, 335		
<code>\ifEmFi@item</code>	294, 625, 667		
<code>\ifEmFi@sort</code>	291		
<code>\ifEmFi@visible</code>	292, 435		
<code>\ifnum</code>	697, 711, 819		
<code>\ifpdf</code>	247		
<code>\ifx</code>	125, 128, 131, 160, 168, 171, 225, 239, 262, 320, 323, 342, 360, 362, 364, 366, 408, 423, 424, 425, 426, 427, 428, 429, 430, 449, 460, 471, 485, 487, 497, 503, 527, 529, 531, 533, 535, 537, 539, 541, 543, 558, 561, 598, 609, 614, 619, 636, 650, 656, 660, 681, 682, 688, 707, 746, 773, 778, 789, 796, 825, 829, 833, 835, 838, 853		
<code>\immediate</code>	133, 162, 627, 634, 647, 753, 766, 823		
<code>\input</code>	4, 6, 7, 227		
J			
<code>\jobname</code>	95, 103, 108, 109		
K			
<code>\kv@define@key</code>	308, 330, 355, 450, 461, 472, 478, 483, 525, 553, 571, 574		
<code>\kvsetkeys</code>	388, 417, 580, 596		
L			
<code>\ltx@empty</code>	360, 497, 503, 609, 614, 615, 619, 620, 636, 650, 656, 660, 704, 707, 746, 773, 778, 779, 802, 803, 825, 829, 838		
<code>\ltx@firstoftwo</code>	345		
<code>\ltx@gobbletwo</code>	791		
<code>\ltx@ifnextchar</code>	583		
<code>\ltx@newif</code>	290, 291, 292, 293, 294, 295, 296		
<code>\ltx@secondoftwo</code>	343		
<code>\ltx@space</code>	337, 382, 401, 509, 589, 665, 668, 675, 759, 764, 770, 846		
M			
<code>\MessageBreak</code>	371, 492, 547, 548, 565, 799		
N			
<code>\NeedsTeXFormat</code>	40, 101		
<code>\numexpr</code>	240		
O			
<code>\outputmode</code>	244		
P			
<code>\PackageInfo</code>	136		
<code>\pdf@escapename</code>	421, 501, 638, 652, 662, 684, 687		
<code>\pdf@escapestring</code>	607, 701, 782		
<code>\pdf@filemdfivesum</code>	643		
<code>\pdf@filemoddate</code>	641		
<code>\pdf@filesize</code>	599, 642		
<code>\pdf@strcmp</code>	697, 711		
<code>\pdfcatalog</code>	243, 769, 845		
<code>\pdfextension</code>	239, 241, 242, 243		
<code>\pdffeedback</code>	240		
<code>\pdflastobj</code>	240, 337, 632, 665, 675, 759, 770, 846		
<code>\pdfnames</code>	241, 758		
<code>\pdfobj</code>	242, 627, 634, 647, 753, 766, 823		
<code>\pdfoutput</code>	244		
<code>\pdfstringdef</code>	43, 320, 323, 698		
<code>\protected</code>	240, 241, 242, 243		
<code>\ProvidesPackage</code>	102, 129, 177		
R			
<code>\RequirePackage</code>	104, 230		
<code>\resetatcatcode</code>	8		
S			
<code>\stop</code>	723, 730, 734		
T			
<code>\the</code>	187, 188, 189, 190, 191, 192, 193, 194, 207, 337, 412, 632, 665, 675, 714, 721, 724, 727, 731, 759, 770, 846		
<code>\TMP@EnsureCode</code>	204, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222		
<code>\toks@</code>	713, 714, 721, 724, 727, 731, 733		
U			
<code>\usepackage</code>	42, 45		

	W		X
		\x	124, 125,
			128, 132, 136, 138, 161, 166,
\write	133, 162		176, 185, 197, 712, 719, 788, 789