

The `embedfile` package

Heiko Oberdiek*

2019/12/03 v2.9

Abstract

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

Contents

1 Documentation	2
1.1 Introduction	2
1.1.1 Future development	2
1.2 User interface	3
1.3 Collection support (PDF 1.7)	4
1.4 Export of object references	5
1.4.1 Example	6
1.5 Examples	6
1.5.1 plain TeX	6
1.5.2 Collection example	7
1.6 Package <code>dtx-attach</code>	8
2 Implementation	8
2.1 Reload check and package identification	8
2.2 Catcodes	9
2.3 Tools	10
2.4 Check for recent pdfTeX in PDF mode	11
2.5 Strings	11
2.6 Switches	12
2.7 Key value definitions	13
2.8 Embed the file	18
3 Installation	24
3.1 Download	24
3.2 Bundle installation	24
3.3 Package installation	24
3.4 Refresh file name databases	24
3.5 Some details for the interested	25
4 References	25

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

5 History	25
[2006/08/16 v1.0]	25
[2007/04/11 v1.1]	25
[2007/09/09 v1.2]	26
[2007/10/28 v2.0]	26
[2007/10/29 v2.1]	26
[2007/11/11 v2.2]	26
[2007/11/25 v2.3]	26
[2009/09/25 v2.4]	26
[2010/03/01 v2.5]	26
[2011/04/13 v2.6]	26
[2016/05/15 v2.7]	26
[2018/11/01 v2.8]	26
[2019/12/03 v2.9]	27

6 Index	27
----------------	-----------

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 [subsubsection 1.5.1](#) that explains the use with plain T_EX by an example. In L^AT_EX the package is loaded as usually. There are no options.

```
\usepackage{embedfile}
```

```
\embedfile [<options>] {<file>}
```

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

The `<options>` are give as key value pairs. The following keys are supported:

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

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

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

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

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

desc The description for the file.

stringmethod The package must convert the values of the keys `ucfilespec` and `desc` into a PDF string (before version 2.4: `filespec` and `desc`). If `hyperref` is found, then its `\pdfstringdef` will be used, otherwise L^AT_EX'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^AT_EX this is automatically done. The package uses `\AtEndDocument`.

Then the list of all files should be known. However, plain T_EX does not know about `\AtEndDocument`. Thus the user must call `\embedfilefinish` at the end of the document after the last file is embedded.

`\embedfilesetup {<options>}`

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

1.3 Collection support (PDF 1.7)

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

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

Name	Description	Modified	Size
...

Acrobat Reader 10 shows the embedded files in the left panel and adds a new column for the compressed size.

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

Collection support is enabled automatically, if it is used.

`\embedfilesetup {<options>}`

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

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

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

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

`hidden` The collection table is not shown.

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

`\embedfilefield {<key>} {<options>}`

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

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

`text` A text field. Its value is set in `\embedfile` by option `<key>.value`.

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

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

file The file name of the embedded file.

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

moddate The modification date of the embedded file.

size The size of the embedded file.

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

title sets the column title.

visible controls whether the column is presented:

`true` shows the column.

`false` hides the column.

Default: `true`

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

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

`false` disables the feature.

Default: `false`

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

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

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

1.4 Export of object references

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

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

- `EmbeddedFile`
- `Filespec`

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

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

```
\embedfilegetobject {\langle id\rangle} {\langle type\rangle}
```

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

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}[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 L^AT_EX.

```

100 <*dtxattach>
101 \NeedsTeXFormat{LaTeX2e}
102 \ProvidesPackage{dtx-attach}
103   [2019/12/03 v2.9 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 [2019/12/03 v2.9 Embed files into PDF (HO)]%

```

2.2 Catcodes

```

179 \begingroup\catcode61\catcode48\catcode32=10\relax%
180 \catcode13=5 % ^~M
181 \endlinechar=13 %
182 \catcode123=1 % {
183 \catcode125=2 % }
184 \catcode64=11 % @
185 \def\x{\endgroup

```

```

186 \expandafter\edef\csname EmFi@AtEnd\endcsname{%
187   \endlinechar=\the\endlinechar\relax
188   \catcode13=\the\catcode13\relax
189   \catcode32=\the\catcode32\relax
190   \catcode35=\the\catcode35\relax
191   \catcode61=\the\catcode61\relax
192   \catcode64=\the\catcode64\relax
193   \catcode123=\the\catcode123\relax
194   \catcode125=\the\catcode125\relax
195 }%
196 }%
197 \x\catcode61\catcode48\catcode32=10\relax%
198 \catcode13=5 % ^~M
199 \endlinechar=13 %
200 \catcode35=6 % #
201 \catcode64=11 % @
202 \catcode123=1 % {
203 \catcode125=2 % }
204 \def\TMP@EnsureCode#1#2{%
205   \edef\EmFi@AtEnd{%
206     \EmFi@AtEnd
207     \catcode#1=\the\catcode#1\relax
208   }%
209   \catcode#1=#2\relax
210 }%
211 \TMP@EnsureCode{39}{12}%
212 \TMP@EnsureCode{40}{12}%
213 \TMP@EnsureCode{41}{12}%
214 \TMP@EnsureCode{44}{12}%
215 \TMP@EnsureCode{46}{12}%
216 \TMP@EnsureCode{47}{12}%
217 \TMP@EnsureCode{58}{12}%
218 \TMP@EnsureCode{60}{12}%
219 \TMP@EnsureCode{62}{12}%
220 \TMP@EnsureCode{91}{12}%
221 \TMP@EnsureCode{93}{12}%
222 \TMP@EnsureCode{96}{12}%
223 \edef\EmFi@AtEnd{\EmFi@AtEnd\noexpand\endinput}

```

2.3 Tools

```

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

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

```

Luatex compat

```

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

```

2.4 Check for recent pdfTeX in PDF mode

Load package iftex and check mode.

```

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

```

Check version.

```

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

```

2.5 Strings

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

```

267 \EmFi@RequirePackage{pdfescape}[2007/11/11]
268 \def\EmFi@temp#1{%
269   \expandafter\EdefSanitize\csname EmFi@S@#1\endcsname{#1}%
270 }

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

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

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

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

```

```

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

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

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

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

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

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

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

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

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

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

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

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

```

2.6 Switches

```

\ifEmFi@collection
287 \ltx@newif\ifEmFi@collection

\ifEmFi@sort
288 \ltx@newif\ifEmFi@sort

\ifEmFi@visible
289 \ltx@newif\ifEmFi@visible

\ifEmFi@edit
290 \ltx@newif\ifEmFi@edit

\ifEmFi@item
291 \ltx@newif\ifEmFi@item

\ifEmFi@finished
292 \ltx@newif\ifEmFi@finished

\ifEmFi@id
293 \ltx@newif\ifEmFi@id

```

2.7 Key value definitions

```

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

\EmFi@GlobalDefaultKey
298 \def\EmFi@GlobalDefaultKey#1#2{%
299   \EmFi@GlobalKey{#1}{#2}%
300   \global\expandafter\let
301     \csname KV@#1@#2@default\expandafter\endcsname
302     \csname KV@#1@#2@default\endcsname
303 }

\EmFi@DefineKey
304 \def\EmFi@DefineKey#1#2{%
305   \kv@define@key{EmFi}{#1}{%
306     \expandafter\def\csname EmFi@#1\endcsname{##1}%
307   }%
308   \expandafter\def\csname EmFi@#1\endcsname{#2}%
309 }

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

File specification string.
311 \EmFi@DefineKey{filespec}{\EmFi@file}
File specification string in Unicode.
312 \EmFi@DefineKey{ucfilespec}{}
File system (optional).
313 \EmFi@DefineKey{filesystem}{}
Description (optional).
314 \EmFi@DefineKey{desc}{}
AFRelationship (mandatory for PDF/A-3 compliance).
315 \EmFi@DefineKey{afrelationship}{}
Method for converting text to PDF strings.
316 \EmFi@DefineKey{stringmethod}{%
317   \ifx\pdfstringdef\@undefined
318     \escape%
319   \else
320     \ifx\pdfstringdef\relax
321       \escape%
322     \else
323       \psd%
324     \fi
325   \fi
326 }

Option id as key for object numbers.
327 \kv@define@key{EmFi}{id}{%
328   \def\EmFi@id{#1}%
329   \EmFi@idtrue
330 }

```

```

\EmFi@defobj
331 \def\EmFi@defobj#1{%
332   \ifEmFi@id
333     \expandafter\xdef\csname EmFi@#1@\EmFi@id\endcsname{%
334       \the\pdflastobj\ltx@space 0 R%
335     }%
336   \fi
337 }

\embedfileifobjectexists
338 \def\embedfileifobjectexists#1#2{%
339   \expandafter\ifx\csname EmFi@#2@#1\endcsname\relax
340     \expandafter\ltx@secondoftwo
341   \else
342     \expandafter\ltx@firstoftwo
343   \fi
344 }

\embedfilegetobject
345 \def\embedfilegetobject#1#2{%
346   \embedfileifobjectexists{#1}{#2}{%
347     \csname EmFi@#2@#1\endcsname
348   }%
349   O O R%
350 }%
351 }

Initial view of the collection.

352 \kv@define@key{EmFi}{view}[]{%
353   \EdefSanitize\EmFi@temp{#1}%
354   \def\EmFi@next{%
355     \global\EmFi@collectiontrue
356   }%
357   \ifx\EmFi@temp\ltx@empty
358     \let\EmFi@view\EmFi@S@details
359   \else\ifx\EmFi@temp\EmFi@S@details
360     \let\EmFi@view\EmFi@S@details
361   \else\ifx\EmFi@temp\EmFi@S@tile
362     \let\EmFi@view\EmFi@S@tile
363   \else\ifx\EmFi@temp\EmFi@S@hidden
364     \let\EmFi@view\EmFi@S@hidden
365   \else
366     \let\EmFi@next\relax
367     \EmFi@Error{%
368       Unknown value ‘\EmFi@temp’ for key ‘view’. \MessageBreak
369       Supported values: ‘details’, ‘tile’, ‘hidden’.%%
370     }@\ehc
371   \fi\fi\fi\fi
372   \EmFi@next
373 }

374 \EmFi@DefineKey{initialfile}{}

\embedfilesetup
375 \def\embedfilesetup{%
376   \ifEmFi@finished
377     \def\EmFi@next##1{}%
378   \EmFi@Error{%

```

```

379      \string\embedfilefield\ltx@space after \string\embedfilefinish
380    }{%
381      The list of embedded files is already written.%}
382    }%
383  \else
384    \def\EmFi@next{%
385      \kvsetkeys{EmFi}{%
386    }%
387  \fi
388  \EmFi@next
389 }

\EmFi@schema
390 \def\EmFi@schema{}

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

\EmFi@order
392 \let\EmFi@@order\relax

\EmFi@fieldlist
393 \def\EmFi@fieldlist{}

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

\embedfilefield
395 \def\embedfilefield#1#2{%
396   \ifEmFi@finished
397     \EmFi@Error{%
398       \string\embedfilefield\ltx@space after \string\embedfilefinish
399     }{%
400       The list of embedded files is already written.%}
401     }%
402   \else
403     \global\EmFi@collectiontrue
404     \EdefSanitize\EmFi@key{#1}%
405     \expandafter\ifx\csname KV@EmFi@EmFi@key.prefix\endcsname\relax
406       \begingroup
407         \count@\= \EmFi@order
408         \advance\count@ 1 %
409         \xdef\EmFi@order{\the\count@}%
410         \let\EmFi@title\EmFi@key
411         \let\EmFi@type\EmFi@S@text
412         \EmFi@visibletrue
413         \EmFi@editfalse
414         \kvsetkeys{EmFiFi}{#2}%
415         \EmFi@convert\EmFi@title\EmFi@title
416         \xdef\EmFi@schema{%
417           \EmFi@schema
418           /\pdf@escapename{\EmFi@key}<<%
419             /Subtype/%
420             \ifx\EmFi@type\EmFi@S@date D%
421             \else\ifx\EmFi@type\EmFi@S@number N%
422             \else\ifx\EmFi@type\EmFi@S@file F%
423             \else\ifx\EmFi@type\EmFi@S@desc Desc%

```

```

424     \else\ifx\EmFi@type\EmFi@S@afrelationship AFRelationship%
425     \else\ifx\EmFi@type\EmFi@S@moddate ModDate%
426     \else\ifx\EmFi@type\EmFi@S@creationdate CreationDate%
427     \else\ifx\EmFi@type\EmFi@S@size Size%
428     \else S%
429     \fi\fi\fi\fi\fi
430     /N(\EmFi@title)%
431     \EmFi@order{\EmFi@order}%
432     \ifEmFi@visible
433     \else
434         /V false%
435     \fi
436     \ifEmFi@edit
437         /E true%
438     \fi
439     >>%
440   }%
441   \let\do\relax
442   \xdef\EmFi@fieldlist{%
443     \EmFi@fieldlist
444     \do{\EmFi@key}%
445   }%
446   \ifx\EmFi@type\EmFi@S@text
447     \kv@define@key{\EmFi}{\EmFi@key.value}{%
448       \EmFi@itemtrue
449       \def\EmFi@temp{##1}%
450       \EmFi@convert\EmFi@temp\EmFi@temp
451       \expandafter\def\csname EmFi@V@#1%
452       \expandafter\endcsname\expandafter{%
453         \expandafter(\EmFi@temp)%
454       }%
455     }%
456     \EmFi@GlobalKey{\EmFi}{\EmFi@key.value}%
457   \else\ifx\EmFi@type\EmFi@S@date
458     \kv@define@key{\EmFi}{\EmFi@key.value}{%
459       \EmFi@itemtrue
460       \def\EmFi@temp{##1}%
461       \EmFi@convert\EmFi@temp\EmFi@temp
462       \expandafter\def\csname EmFi@V@#1%
463       \expandafter\endcsname\expandafter{%
464         \expandafter(\EmFi@temp)%
465       }%
466     }%
467     \EmFi@GlobalKey{\EmFi}{\EmFi@key.value}%
468   \else\ifx\EmFi@type\EmFi@S@number
469     \kv@define@key{\EmFi}{\EmFi@key.value}{%
470       \EmFi@itemtrue
471       \expandafter\EdefSanitize\csname EmFi@V@#1\endcsname{ ##1}%
472     }%
473     \EmFi@GlobalKey{\EmFi}{\EmFi@key.value}%
474     \fi\fi\fi
475     \kv@define@key{\EmFi}{\EmFi@key.prefix}{%
476       \EmFi@itemtrue
477       \expandafter\def\csname EmFi@P@#1\endcsname{##1}%
478     }%
479     \EmFi@GlobalKey{\EmFi}{\EmFi@key.prefix}%
480     \kv@define@key{\EmFiSo}{\EmFi@key}[ascending]{%
481       \EdefSanitize\EmFi@temp{##1}%

```

```

482          \ifx\EmFi@temp\EmFi@S@ascending
483              \def\EmFi@temp{true}%
484          \else\ifx\EmFi@temp\EmFi@S@descending
485              \def\EmFi@temp{false}%
486          \else
487              \def\EmFi@temp{}%
488          \EmFi@Error{%
489              Unknown sort order '\EmFi@temp'.\MessageBreak
490              Supported values: '\EmFi@S@ascending', %
491                  '\EmFi@S@descending
492          }@\ehc
493      \fi\fi
494      \ifx\EmFi@temp\ltx@empty
495      \else
496          \xdef\EmFi@sortkeys{%
497              \EmFi@sortkeys
498              /\pdf@escapename{\#1}%
499          }%
500          \ifx\EmFi@sortorders\ltx@empty
501              \global\let\EmFi@sortorders\EmFi@temp
502              \gdef\EmFi@sortcase{1}%
503          \else
504              \xdef\EmFi@sortorders{%
505                  \EmFi@sortorders
506                  \ltx@space
507                  \EmFi@temp
508              }%
509              \xdef\EmFi@sortcase{2}%
510          \fi
511      \fi
512  }%
513  \EmFi@GlobalDefaultKey{EmFiSo}\EmFi@key
514  \endgroup
515 \else
516     \EmFi@Error{%
517         Field '\EmFi@key' is already defined%
518     }@\ehc
519     \fi
520 \fi
521 }

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

```

```

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

550 \kv@define@key{EmFiFi}{title}{%
551   \def\EmFi@title{#1}%
552 }

\EmFi@setboolean

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

568 \kv@define@key{EmFiFi}{visible}[true]{%
569   \EmFi@setboolean{visible}{#1}%
570 }

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

\EmFi@sortkeys

574 \def\EmFi@sortkeys{}

\EmFi@sortorders

575 \def\EmFi@sortorders{}

\embedfilesort

576 \def\embedfilesort{%
577   \kvsetkeys{EmFiSo}%
578 }

```

2.8 Embed the file

```

\embedfile

579 \def\embedfile{%
580   \ltx@ifnextchar[\EmFi@embedfile{\EmFi@embedfile[]}]%
581 }

```

```

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

```

```

639      >>%
640      }file{\EmFi@file}\relax
641      \EmFi@defobj{EmbeddedFile}%
642      \immediate\pdfobj{%
643      <<%
644      /Type/Filespec%
645      \ifx\EmFi@filesystem\ltx@empty
646      \else
647      /FS/\pdf@escapename{\EmFi@filesystem}%
648      \fi
649      /F(\EmFi@@filespec)%
650      \ifx\EmFi@@ucfilespec\ltx@empty
651      \else
652      /UF(\EmFi@@ucfilespec)%
653      \fi
654      \ifx\EmFi@@desc\ltx@empty
655      \else
656      /Desc(\EmFi@@desc)%
657      \fi
658      \ifx\EmFi@@afrelationship\ltx@empty
659      \else
660      /AFRelationship\EmFi@@afrelationship%
661      \fi
662      /EF<<%
663      /F \the\pdflastobj\ltx@space 0 R%
664      >>%
665      \ifEmFi@item
666      /CI \EmFi@ci\ltx@space 0 R%
667      \fi
668      >>%
669      }%
670      \EmFi@defobj{Filespec}%
671      \EmFi@add{%
672      \EmFi@@filespec
673      }{\the\pdflastobj\ltx@space 0 R}%
674      \fi
675      \endgroup
676      \fi
677 }

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

```

```

\EmFi@convert
694 \def\EmFi@convert#1#2{%
695   \ifnum\pdfstrcmp{\EmFi@stringmethod}{psd}=0 %
696     \pdfstringdef\EmFi@temp{#1}%
697     \let#2\EmFi@temp
698   \else
699     \edef#2{\pdfescapestring{#1}}%
700   \fi
701 }

702 \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,
            ...).
703 \def\EmFi@add#1#2{%
704   \begingroup
705     \ifx\EmFi@list\ltx@empty
706       \xdef\EmFi@list{\noexpand\do{#1}{#2}}%
707     \else
708       \def\do##1##2{%
709         \ifnum\pdfstrcmp{##1}{#1}>0 %
710           \edef\x{%
711             \toks@{%
712               \the\toks@%
713               \noexpand\do{#1}{#2}%
714               \noexpand\do{##1}{##2}%
715             }%
716           }%
717           \x
718           \def\do####1####2{%
719             \toks@\expandafter{\the\toks@\do{####1}{####2}}%
720           }%
721           \def\stop{%
722             \xdef\EmFi@list{\the\toks@}%
723           }%
724           \else
725             \toks@\expandafter{\the\toks@\do{##1}{##2}}%
726             \fi
727           }%
728           \def\stop{%
729             \xdef\EmFi@list{\the\toks@\noexpand\do{#1}{#2}}%
730           }%
731           \toks@{}%
732           \EmFi@list\stop
733         \fi
734       \endgroup
735 }

\embedfilefinish
736 \def\embedfilefinish{%
737   \ifEmFi@finished
738     \EmFi@Error{%
739       Too many invocations of \string\embedfilefinish
740     }{%
741       The list of embedded files is already written.%
742     }%

```

```

743   \else
744     \ifx\EmFi@list\ltx@empty
745   \else
746     Write /EmbeddedFiles entry.
747       \global\EmFi@finishedtrue
748       \begingroup
749         \def\do##1##2{%
750           (##1)##2%
751         }%
752         \immediate\pdfobj{%
753           <<%
754             /Names[\EmFi@list]%
755           >>%
756         }%
757         \pdfnames{%
758           /EmbeddedFiles \the\pdflastobj\ltx@space 0 R%
759         }%
760       \endgroup
761       \begingroup
762         \def\do##1##2{%
763           \ltx@space##2%
764         }%
765         \immediate\pdfobj{%
766           [\EmFi@list]%
767         }%
768         \pdfcatalog{%
769           /AF \the\pdflastobj\ltx@space 0 R%
770         }%
771       \endgroup
772     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
785       Look for initial file among the embedded files.
786       \begingroup
787         \let\f=N%
788         \def\do##1##2{%
789           \def\x{##1}%
790           \ifx\x\EmFi@initialfile
791             \let\f=Y%
792             \let\do\ltx@gobbletwo
793           \expandafter\endgroup
794           \ifx\f Y%
795           \else
796             \PackageWarningNoLine{embedfile}{%

```

```

797      Missing initial file '\EmFi@initialfile'\MessageBreak
798      among the embedded files%
799      }%
800      \let\EmFi@initialfile\ltx@empty
801      \let\EmFi@@initialfile\ltx@empty
802      \fi
803      \ifcase\EmFi@sortcase
804          \def\EmFi@temp{}%
805      \or
806          \def\EmFi@temp{}%
807          /S\EmFi@sortkeys
808          /A \EmFi@sortorders
809      }%
810      \else
811          \def\EmFi@temp{}%
812          /S[\EmFi@sortkeys]%
813          /A[\EmFi@sortorders]%
814      }%
815      \fi
816      \def\EmFi@@order##1{%
817          \ifnum\EmFi@order>1 %
818              /O ##1%
819          \fi
820      }%
821      \immediate\pdfobj{%
822          <<%
823          \ifx\EmFi@schema\ltx@empty
824          \else
825              /Schema<<\EmFi@schema>>%
826          \fi
827          \ifx\EmFi@@initialfile\ltx@empty
828          \else
829              /D(\EmFi@@initialfile)%
830          \fi
831          \ifx\EmFi@view\EmFi@S@tile
832              /View/T%
833          \else\ifx\EmFi@view\EmFi@S@hidden
834              /View/H%
835          \fi\fi
836          \ifx\EmFi@temp\ltx@empty
837              \EmFi@temp
838          \else
839              /Sort<<\EmFi@temp>>%
840          \fi
841      >>%
842      }%
843      \pdfcatalog{%
844          /Collection \the\pdflastobj\ltx@space0 R%
845      }%
846      \fi
847      \fi
848      \fi
849 }

850 \begingroup\expandafter\expandafter\expandafter\endgroup
851 \expandafter\ifx\csname AtEndDocument\endcsname\relax
852 \else
853     \AtEndDocument{\embedfilefinish}%
854 \fi

```

```
855 \EmFi@AtEnd%
856 </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.

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 T_EX:

```
tex embedfile.dtx
```

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

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

3.4 Refresh file name databases

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

¹[CTAN:pkg/embedfile](#)

3.5 Some details for the interested

Unpacking with L^AT_EX. The .dtx chooses its action depending on the format:

plain T_EX: Run docstrip and extract the files.

L^AT_EX: Generate the documentation.

If you insist on using L^AT_EX for docstrip (really, docstrip does not need L^AT_EX), 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^AT_EX:

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

[2011/04/13 v2.6]

- Docu fixes (thanks Hans-Martin Münch).

[2016/05/15 v2.7]

- LuaTeX compatibility

[2018/11/01 v2.8]

- Remove `luatex85` package dependency.

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

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	
\@PackageError	234
\@PackageWarningNoLine	796
\@ehc	370, 492, 518, 564
\@undefined	168, 236, 317
\`	595
A	
\advance	408
\aftergroup	139
\AtEndDocument	853
B	
\begin	75
\bye	35
C	
\catcode	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
\count@	407, 408, 409
\csname	124, 131, 160, 176, 186, 225, 259, 269, 295, 296, 301, 302, 306, 308, 333, 339, 347, 405, 451, 462, 471, 477, 556, 559, 679, 680, 682, 686, 688, 690, 851
D	
\do	441, 444, 621, 706, 708, 713, 714, 718, 719, 725, 729, 748, 761, 785, 789
\documentclass	41
E	
\EdefSanitize	. 269, 353, 404, 471, 481, 523, 554
\embedfile	3, 16, 21, 25, 79, 85, 91, 105, 579, 585
\embedfilefield	... 4, 50, 54, 58, 62, 66, 379, 395
\embedfilefinish	... 3, 34, 379, 398, 585, <u>736</u> , 853
\embedfilegetobject	... 6, 345
\embedfileifobjectexists	5, <u>338</u> , 346
\embedfilesetup	4, 4, 11, 46, <u>375</u>
\embedfilesort	5, 71, <u>576</u>
\EmFi@afrelationship	616, 618, 658, 660
\EmFi@desc	611, 613, 654, 656
\EmFi@filespec	602, 649, 672
\EmFi@initialfile	... 777, 779, 787, 801, 827, 829
\EmFi@order	392, 431, 816
\EmFi@ucfilespec	606, 608, 650, 652
\EmFi@add	671, <u>703</u>
\EmFi@afrelationship	615, 618
\EmFi@AtEnd	205, 206, 223, 252, 265, 855
\EmFi@ci	627, 666
\EmFi@collectiontrue	355, 403, 773
\EmFi@convert	415, 450, 461, 608, 613, 618, <u>694</u>
\EmFi@DefineKey	304, 310, 311, 312, 313, 314, 315, 316, 374
\EmFi@defobj	331, 641, 670
\EmFi@desc	610, 613
\EmFi@details	271
\EmFi@do	621, <u>678</u>
\EmFi@editfalse	413
\EmFi@embedfile	580, <u>582</u>
\EmFi@Error	232, 246, 260, 367, 378, 397, 488, 516, 543, 561, 584, 596, 738
\EmFi@fieldlist	393, 442, 443, 624
\EmFi@file	311, 591, 595, 597, 636, 637, 638, 640
\EmFi@filespec	603
\EmFi@filesystem	645, 647
\EmFi@finishedtrue	746
\EmFi@GlobalDefaultKey	298, 513
\EmFi@GlobalKey	294, 299, 456, 467, 473, 479
\EmFi@hidden	273
\EmFi@id	328, 333

\EmFi@idtrue	329	\empty	127, 128
\EmFi@initialfile	771, 776, 780, 797, 800	\end	97
\EmFi@itemtrue	448, 459, 470, 476	\endcsname	
\EmFi@key	404, 405, 410, 418, 444, 447, 456, 458, 467, 469, 473, 475, 479, 480, 513, 517	. 124, 131, 160, 176, 186, 225, 259, 269, 295, 296, 301, 302, 306, 308, 333, 339, 347, 405, 452, 463, 471, 477, 556, 559, 679, 680, 682, 686, 688, 690, 851	
\EmFi@list	702, 705, 706, 722, 729, 732, 744, 753, 765, 792	\endinput	139, 223
\EmFi@mimetype	631, 633	\endlinechar	114, 145, 181, 187, 199
\EmFi@next	354, 366, 372, 377, 384, 388		
\EmFi@order	391, 407, 409, 431, 817		
\EmFi@RequirePackage	224, 232, 243, 254, 255, 256, 257, 267		
\EmFi@S@afrelationship	279, 424, 534		
\EmFi@S@ascending	283, 482, 490		
\EmFi@S@creationdate	281, 426, 538		
\EmFi@S@date	275, 420, 457, 526		
\EmFi@S@desc	278, 423, 532		
\EmFi@S@descending	284, 484, 491		
\EmFi@S@details	358, 359, 360		
\EmFi@S@false	286, 558		
\EmFi@S@file	277, 422, 530		
\EmFi@S@hidden	363, 364, 833		
\EmFi@S@moddate	280, 425, 536		
\EmFi@S@number	276, 421, 468, 528		
\EmFi@S@size	282, 427, 540		
\EmFi@S@text	274, 411, 446, 524		
\EmFi@S@tile	361, 362, 831		
\EmFi@S@true	285, 555		
\EmFi@schema	390, 416, 417, 823, 825		
\EmFi@setboolean	553, 569, 572		
\EmFi@sortcase	394, 502, 509, 803		
\EmFi@sortkeys	496, 497, 574, 807, 812		
\EmFi@sortorders 500, 501, 504, 505, 575, 808, 813		
\EmFi@stringmethod	695		
\EmFi@temp	268, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 353, 357, 359, 361, 363, 368, 449, 450, 453, 460, 461, 464, 481, 482, 483, 484, 485, 487, 489, 494, 501, 507, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 544, 554, 555, 558, 562, 696, 697, 804, 806, 811, 836, 837, 839		
\EmFi@tile	272		
\EmFi@title	410, 415, 430, 551		
\EmFi@type	411, 420, 421, 422, 423, 424, 425, 426, 427, 446, 457, 468, 525, 527, 529, 531, 533, 535, 537, 539, 541		
\EmFi@ucflespec	605, 608		
\EmFi@view	358, 360, 362, 364, 831, 833		
\EmFi@visibletrue	412		
\empty	127, 128		
\end	97		
\endcsname			
. 124, 131, 160, 176, 186, 225, 259, 269, 295, 296, 301, 302, 306, 308, 333, 339, 347, 405, 452, 463, 471, 477, 556, 559, 679, 680, 682, 686, 688, 690, 851			
\endinput	139, 223		
\endlinechar	114, 145, 181, 187, 199		
F			
\f	784, 788, 794		
G			
\gdef	391, 502		
\Gin@driver	5		
I			
\ifcase	803		
\ifEmFi@collection	287, 775		
\ifEmFi@edit	290, 436		
\ifEmFi@finished	292, 376, 396, 583, 737		
\ifEmFi@id	293, 332		
\ifEmFi@item	291, 620, 665		
\ifEmFi@sort	288		
\ifEmFi@visible	289, 432		
\ifnum	695, 709, 817		
\ifpdf	244		
\ifx ... 125, 128, 131, 160, 168, 171, 225, 236, 259, 317, 320, 339, 357, 359, 361, 363, 405, 420, 421, 422, 423, 424, 425, 426, 427, 446, 457, 468, 482, 484, 494, 500, 524, 526, 528, 530, 532, 534, 536, 538, 540, 555, 558, 594, 605, 610, 615, 631, 645, 650, 654, 658, 679, 680, 686, 705, 744, 771, 776, 787, 794, 823, 827, 831, 833, 836, 851			
\immediate	133, 162, 622, 629, 642, 751, 764, 821		
\input	4, 6, 7, 227		
J			
\jobname	95, 103, 108, 109		
K			
\kv@define@key			
. 305, 327, 352, 447, 458, 469, 475, 480, 522, 550, 568, 571			
\kvsetkeys	385, 414, 577, 592		
L			
\ltx@empty			
. 357, 494, 500, 605, 606, 610, 611, 615, 616, 631, 645, 650, 654, 658, 702, 705, 744, 771, 776, 777, 800, 801, 823, 827, 836			

\ltx@firstoftwo	342	\pdfnames	238, 756
\ltx@gobbletwo	789	\pdfobj	239, 622, 629, 642, 751, 764, 821
\ltx@ifnextchar	580	\pdfoutput	241
\ltx@newif	. 287, 288, 289, 290, 291, 292, 293	\pdfstringdef	43, 317, 320, 696
\ltx@secondoftwo	340	\protected	237, 238, 239, 240
\ltx@space	334, 379, 398, 506, 585, 663, 666, 673, 757, 762, 768, 844	\ProvidesPackage	102, 129, 177
			R
		\RequirePackage	104, 230
		\resetatcatcode	8
			S
		\stop	721, 728, 732
			T
		\the	187, 188, 189, 190, 191, 192, 193, 194, 207, 334, 409, 627, 663, 673, 712, 719, 722, 725, 729, 757, 768, 844
		\TMP@EnsureCode	. 204, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222
		\toks@	711, 712, 719, 722, 725, 729, 731
			U
		\usepackage	42, 45
			W
		\write	133, 162
			X
		\x	124, 125, 128, 132, 136, 138, 161, 166, 176, 185, 197, 710, 717, 786, 787