The embedfile package

Heiko Oberdiek*

2025-10-27 v2.13

Abstract

This package embeds files to a PDF document. Currently the only supported drivers are pdfTeX >= 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 . . . 3
1.3 Collection support (PDF 1.7) . . . 4
1.4 Export of object references . . . 5
1.4.1 Example . . . 6
1.5 Examples . . . 6
1.5.1 plain TeX . . . 6
1.5.2 Collection example . . . 6
1.6 Package dtx-attach . . . 8

2 Implementation **8**
2.1 Reload check and package identification . . . 8
2.2 Catcodes . . . 9
2.3 Tools . . . 10
2.4 Check for recent pdfTeX in PDF mode . . . 10
2.5 Strings . . . 11
2.6 Switches . . . 12
2.7 Key value definitions . . . 12
2.8 Embed the file . . . 18

3 Installation **23**
3.1 Download . . . 23
3.2 Bundle installation . . . 23
3.3 Package installation . . . 23
3.4 Refresh file name databases . . . 24
3.5 Some details for the interested . . . 24

4 References **24**

5 History **25**
[2006/08/16 v1.0] . . . 25
[2007/04/11 v1.1] . . . 25
[2007/09/09 v1.2] . . . 25
[2007/10/28 v2.0] . . . 25
[2007/10/29 v2.1] . . . 25
[2007/11/11 v2.2] . . . 25

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

[2007/11/25 v2.3] . . . 25
[2009/09/25 v2.4] . . . 25
[2010/03/01 v2.5] . . . 25
[2011/04/13 v2.6] . . . 25
[2016/05/15 v2.7] . . . 25
[2018/11/01 v2.8] . . . 26
[2019/12/03 v2.9] . . . 26
[2020-04-14 v2.10] . . . 26
[2020-04-24 v2.11] . . . 26
[2023-01-12 v2.12] . . . 26
[2025-10-27 v2.13] . . . 26

6 Index **26**

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.

Since version 2.13 the package is compatible with the LaTeX PDF management (loaded with `\RequirePackage{pdfmanagement}` or `\DocumentMetadata`), but it doesn't make full use of its tools yet, the dictionary of the embedded files can *not* be changed as described in the l3pdffile documentation.

1.1.1 Future development

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

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

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

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

1.2 User interface

This package embedfile can be used with both LaTeX and plain TeX. See subsubsection 1.5.1 that explains the use with plain TeX by an example. In LaTeX the package is loaded as usually. There are no options.

```
\usepackage{embedfile}
```

`\embedfile [`⟨*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! If ucfilespec is not set, filespec or the file name are used for the `/UF` key too.

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.

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 pdfTeX's `\pdfescapestring` is used. Value `psd` forces the use of `\pdfstringdef`, value `escape` the use of `\pdfescapestring`.

⟨**key**⟩**.value** Sets the value of a collection item property, see section 1.3.

⟨**key**⟩**.prefix** Sets the prefix of a collection item property, see section 1.3.

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

`\embedfilefinish`

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

`\embedfilesetup {`⟨*options*⟩`}`

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

1.3 Collection support (PDF 1.7)

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

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

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

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

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

Collection support is enabled automatically, if it is used.

`\embedfilesetup {`⟨*options*⟩`}`

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

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

- `details` The full collection table is displayed at the top below the collection bar.
- `tile` The files of the collection are shown in tile mode on the left.
- `hidden` The collection table is not shown.

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

`\embedfilefield {`⟨*key*⟩`} {`⟨*options*⟩`}`

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

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

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

`date` A date field. Its value is set in `\embedfile` by option ⟨*key*⟩.value. A special format is required, see "3.8.3 Dates" [3].

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

`file` The file name of the embedded file.

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

`moddate` The modification date of the embedded file.

`size` The size of the embedded file.

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

title sets the column title.

visible controls whether the column is presented:

`true` shows the column.

`false` hides the column.

Default: `true`

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

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

`false` disables the feature.

Default: `false`

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

`\embedfilesort` {⟨*key-sort-list*⟩}

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

1.4 Export of object references

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

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

- `EmbeddedFile`
- `Filespec`

`\embedfileifobjectexists` {⟨*id*⟩} {⟨*type*⟩} {⟨*then*⟩} {⟨*else*⟩}

Macro `\embedfileifobjectexists` tests whether object of ⟨*type*⟩ is available for the embedded file identified by ⟨*id*⟩.

`\embedfilegetobject` {⟨*id*⟩} {⟨*type*⟩}

Macro `\embedfilegetobject` expands to the full object reference object of ⟨*type*⟩ for the embedded file identified by ⟨*id*⟩.

1.4.1 Example

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

1.5 Examples

1.5.1 plain 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.

```
⟨*exampleplain⟩
%<<END
% Load packages
\input miniltx
% \def\Gin@driver{pdftex.def}
% \input graphicx.sty
\input embedfile.sty
\resetatcatcode

% default setting
\embedfilesetup{
  mimetype=text/plain
}

% Embed files
\embedfile[
  filespec=example.tex,
  desc={Source code (plain-TeX) of this example}
]{embedfile-example-plain.tex}

\embedfile[
  desc={Source of package 'embedfile'}
]{embedfile.dtx}

\embedfile[
  mimetype=application/pdf,
  desc={Documentation of package 'embedfile'}
]{embedfile.pdf}

% Some text
This example document contains three embedded files.

% End of document
\embedfilefinish % don't forget
\bye
%END
⟨/exampleplain⟩
```

1.5.2 Collection example

```
⟨*examplecollection⟩
%<<END
\NeedsTeXFormat{LaTeX2e}
\documentclass{article}
\usepackage[bookmarks=false]{hyperref}
% provides \pdfstringdef that is then used by 'title' and
% other keys.
\usepackage{embedfile}[2019/12/03]
\embedfilesetup{
  view=details,
  initialfile=embedfile.pdf
}
\embedfilefield{file}{
  type=file,
  title={File name}
}
\embedfilefield{description}{
  type=desc,
  title={Description}
}
\embedfilefield{date}{
  type=moddate,
  title={Date}
}
\embedfilefield{size}{
  type=size,
  title={Size}
}
\embedfilefield{type}{
  type=text,
  title={Type},
  visible=false
}
\embedfilesort{
  type,
  date=descending
}
\begin{document}
An example for embedded files as collection.
You need Acrobat Reader 8 or higher.

\embedfile[
  desc={Source file of package 'embedfile'},
  description.prefix={Package: },
  type.value={DTX}
]{embedfile.dtx}

\embedfile[
  desc={Documentation of package 'embedfile'},
  description.prefix={Package: },
  type.value={PDF}
]{embedfile.pdf}

\embedfile[
  desc={The source for this example},
  description.prefix={Example: },
  type.value={TEX}
]{\jobname.tex}

\end{document}
%END
⟨/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.

```
⟨*dtxattach⟩
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{dtx-attach}
  [2025-10-27 v2.13 Embed \string\jobname.dtx (HO)]%
\RequirePackage{embedfile}[2019/12/03]
\embedfile[%
  stringmethod=escape,%
  mimetype=plain/text,%
  desc={LaTeX docstrip source archive for package `\jobname'}%
]{\jobname.dtx}
⟨/dtxattach⟩
```

2 Implementation

```
⟨*package⟩
```

2.1 Reload check and package identification

Reload check, especially if the package is not used with LaTeX.

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode35=6 % #
  \catcode39=12 % '
  \catcode44=12 % ,
  \catcode45=12 % -
  \catcode46=12 % .
  \catcode58=12 % :
  \catcode64=11 % @
  \catcode123=1 % {
  \catcode125=2 % }
  \expandafter\let\expandafter\x\csname ver@embedfile.sty\endcsname
  \ifx\x\relax % plain-TeX, first loading
  \else
    \def\empty{}%
    \ifx\x\empty % LaTeX, first loading,
      % variable is initialized, but \ProvidesPackage not yet seen
    \else
      \expandafter\ifx\csname PackageInfo\endcsname\relax
        \def\x#1#2{%
          \immediate\write-1{Package #1 Info: #2.}%
        }%
      \else
        \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
      \fi
      \x{embedfile}{The package is already loaded}%
      \aftergroup\endinput
    \fi
  \fi
\endgroup%
```

Package identification:

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

```
  \catcode40=12 % (
  \catcode41=12 % )
  \catcode44=12 % ,
  \catcode45=12 % -
  \catcode46=12 % .
  \catcode47=12 % /
  \catcode58=12 % :
  \catcode64=11 % @
  \catcode91=12 % [
  \catcode93=12 % ]
  \catcode123=1 % {
  \catcode125=2 % }
  \expandafter\ifx\csname ProvidesPackage\endcsname\relax
    \def\x#1#2#3[#4]{\endgroup
      \immediate\write-1{Package: #3 #4}%
      \xdef#1{#4}%
    }%
  \else
    \def\x#1#2[#3]{\endgroup
      #2[{#3}]%
      \ifx#1\@undefined
        \xdef#1{#3}%
      \fi
      \ifx#1\relax
        \xdef#1{#3}%
      \fi
    }%
  \fi
\expandafter\x\csname ver@embedfile.sty\endcsname
\ProvidesPackage{embedfile}%
  [2025-10-27 v2.13 Embed files into PDF (HO)]%
```

2.2 Catcodes

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode123=1 % {
  \catcode125=2 % }
  \catcode64=11 % @
  \def\x{\endgroup
    \expandafter\edef\csname EmFi@AtEnd\endcsname{%
      \endlinechar=\the\endlinechar\relax
      \catcode13=\the\catcode13\relax
      \catcode32=\the\catcode32\relax
      \catcode35=\the\catcode35\relax
      \catcode61=\the\catcode61\relax
      \catcode64=\the\catcode64\relax
      \catcode123=\the\catcode123\relax
      \catcode125=\the\catcode125\relax
    }%
  }%
\x\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode64=11 % @
\catcode123=1 % {
\catcode125=2 % }
\def\TMP@EnsureCode#1#2{%
  \edef\EmFi@AtEnd{%
    \EmFi@AtEnd
```

```
    \catcode#1=\the\catcode#1\relax
  }%
  \catcode#1=#2\relax
}
\TMP@EnsureCode{39}{12}% '
\TMP@EnsureCode{40}{12}% (
\TMP@EnsureCode{41}{12}% )
\TMP@EnsureCode{44}{12}% ,
\TMP@EnsureCode{46}{12}% .
\TMP@EnsureCode{47}{12}% /
\TMP@EnsureCode{58}{12}% :
\TMP@EnsureCode{60}{12}% <
\TMP@EnsureCode{62}{12}% >
\TMP@EnsureCode{91}{12}% [
\TMP@EnsureCode{93}{12}% ]
\TMP@EnsureCode{96}{12}% `
\edef\EmFi@AtEnd{\EmFi@AtEnd\noexpand\endinput}
```

2.3 Tools

\EmFi@RequirePackage

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \def\EmFi@RequirePackage#1[#2]{%
    \input #1.sty\relax
  }%
\else
  \let\EmFi@RequirePackage\RequirePackage
\fi
```

\EmFi@Error

```
\EmFi@RequirePackage{infwarerr}[2007/09/09]%
\ifcsname EmFi@Error\endcsname
\else
  \def\EmFi@Error{%
    \@PackageError{embedfile}%
  }
\fi
```

Luatex compat

```
\ifx\pdfextension\@undefined\else
    \protected\def\pdflastobj {\numexpr\pdffeedback lastobj\relax}
    \protected\def\pdfnames   {\pdfextension names }
    \protected\def\pdfobj     {\pdfextension obj }
    \protected\def\pdfcatalog {\pdfextension catalog }
    \let\pdfoutput            \outputmode
\fi
```

2.4 Check for recent pdfTEX in PDF mode

Load package iftex and check mode.

```
\EmFi@RequirePackage{iftex}[2019/11/07]
\ifpdf
\else
  \EmFi@Error{%
    Missing pdfTeX or luaTeX in PDF mode%
  }{%
    Currently other drivers are not supported. %
    Package loading is aborted.%
  }%
  \expandafter\EmFi@AtEnd
\fi%
```

```
\EmFi@RequirePackage{pdftexcmds}[2007/11/11]
\EmFi@RequirePackage{ltxcmds}[2010/03/01]
\EmFi@RequirePackage{kvsetkeys}[2010/03/01]
\EmFi@RequirePackage{kvdefinekeys}[2010/03/01]
```

Check version.

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname pdf@filesize\endcsname\relax
  \EmFi@Error{%
    Unsupported pdfTeX version%
  }{%
    At least version 1.30 is necessary. Package loading is aborted.%
  }%
  \expandafter\EmFi@AtEnd
\fi%
```

2.5 Strings

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

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

`\EmFi@details`

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

`\EmFi@tile`

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

`\EmFi@hidden`

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

`\EmFi@S@text`

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

`\EmFi@S@date`

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

`\EmFi@S@number`

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

`\EmFi@S@file`

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

`\EmFi@S@desc`

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

`\EmFi@S@afrelationship`

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

`\EmFi@S@moddate`

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

`\EmFi@S@creationdate`

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

`\EmFi@S@size`

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

```
\EmFi@S@ascending
                    \EmFi@temp{ascending}
\EmFi@S@descending
                    \EmFi@temp{descending}
      \EmFi@S@true
                    \EmFi@temp{true}
     \EmFi@S@false
                    \EmFi@temp{false}
```

2.6 Switches

```
\ifEmFi@collection
                    \ltx@newif\ifEmFi@collection
       \ifEmFi@sort
                    \ltx@newif\ifEmFi@sort
    \ifEmFi@visible
                    \ltx@newif\ifEmFi@visible
       \ifEmFi@edit
                    \ltx@newif\ifEmFi@edit
       \ifEmFi@item
                    \ltx@newif\ifEmFi@item
  \ifEmFi@finished
                    \ltx@newif\ifEmFi@finished
        \ifEmFi@id
                    \ltx@newif\ifEmFi@id
```

2.7 Key value definitions

```
    \EmFi@GlobalKey
                    \def\EmFi@GlobalKey#1#2{%
                      \global\expandafter\let\csname KV@#1@#2\expandafter\endcsname
                                              \csname KV@#1@#2\endcsname
                    }
\EmFi@GlobalDefaultKey
                    \def\EmFi@GlobalDefaultKey#1#2{%
                      \EmFi@GlobalKey{#1}{#2}%
                      \global\expandafter\let
                          \csname KV@#1@#2@default\expandafter\endcsname
                          \csname KV@#1@#2@default\endcsname
                    }
    \EmFi@DefineKey
                    \def\EmFi@DefineKey#1#2{%
                      \kv@define@key{EmFi}{#1}{%
                        \expandafter\def\csname EmFi@#1\endcsname{##1}%
                      }%
                      \expandafter\def\csname EmFi@#1\endcsname{#2}%
                    }
```

Subtype of the embedded file (optional).

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

File specification string.

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

File specification string in Unicode.

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

File system (optional).

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

Description (optional).

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

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

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

Method for converting text to PDF strings.

```
\EmFi@DefineKey{stringmethod}{%
  \ifx\pdfstringdef\@undefined
    escape%
  \else
    \ifx\pdfstringdef\relax
      escape%
    \else
      psd%
    \fi
  \fi
}
```

Option id as key for object numbers.

```
\kv@define@key{EmFi}{id}{%
  \def\EmFi@id{#1}%
  \EmFi@idtrue
}
```

`\EmFi@defobj`

```
\def\EmFi@defobj#1{%
  \ifEmFi@id
    \expandafter\xdef\csname EmFi@#1@\EmFi@id\endcsname{%
      \emFi@pdfobjectreflast%
    }%
  \fi
}
```

`\embedfileifobjectexists`

```
\def\embedfileifobjectexists#1#2{%
  \expandafter\ifx\csname EmFi@#2@#1\endcsname\relax
    \expandafter\ltx@secondoftwo
  \else
    \expandafter\ltx@firstoftwo
  \fi
}
```

`\embedfilegetobject`

```
\def\embedfilegetobject#1#2{%
  \embedfileifobjectexists{#1}{#2}{%
    \csname EmFi@#2@#1\endcsname
  }{%
    0 0 R%
  }%
}
```

Initial view of the collection.

```
\kv@define@key{EmFi}{view}[]{%
  \EdefSanitize\EmFi@temp{#1}%
  \def\EmFi@next{%
    \global\EmFi@collectiontrue
  }%
  \ifx\EmFi@temp\ltx@empty
    \let\EmFi@view\EmFi@S@details
  \else\ifx\EmFi@temp\EmFi@S@details
    \let\EmFi@view\EmFi@S@details
  \else\ifx\EmFi@temp\EmFi@S@tile
    \let\EmFi@view\EmFi@S@tile
  \else\ifx\EmFi@temp\EmFi@S@hidden
    \let\EmFi@view\EmFi@S@hidden
  \else
    \let\EmFi@next\relax
    \EmFi@Error{%
      Unknown value `\EmFi@temp' for key `view'.\MessageBreak
      Supported values: `details', `tile', `hidden'.%
    }\@ehc
  \fi\fi\fi\fi
  \EmFi@next
}
\EmFi@DefineKey{initialfile}{}
```

`\embedfilesetup`

```
\def\embedfilesetup{%
  \ifEmFi@finished
    \def\EmFi@next##1{}%
    \EmFi@Error{%
      \string\embedfilefield\ltx@space after \string\embedfilefinish
    }{%
      The list of embedded files is already written.%
    }%
  \else
    \def\EmFi@next{%
      \kvsetkeys{EmFi}%
    }%
  \fi
  \EmFi@next
}
```

`\EmFi@schema`

```
\def\EmFi@schema{}
```

`\EmFi@order`

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

`\EmFi@@order`

```
\let\EmFi@@order\relax
```

`\EmFi@fieldlist`

```
\def\EmFi@fieldlist{}
```

`\EmFi@sortcase`

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

`\embedfilefield`

```
\def\embedfilefield#1#2{%
  \ifEmFi@finished
    \EmFi@Error{%
```

```
    \string\embedfilefield\ltx@space after \string\embedfilefinish
  }{%
    The list of embedded files is already written.%
  }%
\else
  \global\EmFi@collectiontrue
  \EdefSanitize\EmFi@key{#1}%
  \expandafter\ifx\csname KV@EmFi@\EmFi@key.prefix\endcsname\relax
    \begingroup
      \count@=\EmFi@order
      \advance\count@ 1 %
      \xdef\EmFi@order{\the\count@}%
      \let\EmFi@title\EmFi@key
      \let\EmFi@type\EmFi@S@text
      \EmFi@visibletrue
      \EmFi@editfalse
      \kvsetkeys{EmFiFi}{#2}%
      \EmFi@convert\EmFi@title\EmFi@title
      \xdef\EmFi@schema{%
        \EmFi@schema
        /\pdf@escapename{\EmFi@key}<<%
          /Subtype/%
          \ifx\EmFi@type\EmFi@S@date D%
          \else\ifx\EmFi@type\EmFi@S@number N%
          \else\ifx\EmFi@type\EmFi@S@file F%
          \else\ifx\EmFi@type\EmFi@S@desc Desc%
          \else\ifx\EmFi@type\EmFi@S@afrelationship AFRelationship%
          \else\ifx\EmFi@type\EmFi@S@moddate ModDate%
          \else\ifx\EmFi@type\EmFi@S@creationdate CreationDate%
          \else\ifx\EmFi@type\EmFi@S@size Size%
          \else S%
          \fi\fi\fi\fi\fi\fi\fi\fi
          /N(\EmFi@title)%
          \EmFi@@order{\EmFi@order}%
          \ifEmFi@visible
          \else
            /V false%
          \fi
          \ifEmFi@edit
            /E true%
          \fi
        >>%
      }%
      \let\do\relax
      \xdef\EmFi@fieldlist{%
        \EmFi@fieldlist
        \do{\EmFi@key}%
      }%
      \ifx\EmFi@type\EmFi@S@text
        \kv@define@key{EmFi}{\EmFi@key.value}{%
          \EmFi@itemtrue
          \def\EmFi@temp{##1}%
          \EmFi@convert\EmFi@temp\EmFi@temp
          \expandafter\def\csname EmFi@V@#1%
          \expandafter\endcsname\expandafter{%
            \expandafter(\EmFi@temp)%
          }%
        }%
        \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
      \else\ifx\EmFi@type\EmFi@S@date
        \kv@define@key{EmFi}{\EmFi@key.value}{%
          \EmFi@itemtrue
```

```
            \def\EmFi@temp{##1}%
            \EmFi@convert\EmFi@temp\EmFi@temp
            \expandafter\def\csname EmFi@V@#1%
            \expandafter\endcsname\expandafter{%
              \expandafter(\EmFi@temp)%
            }%
          }%
          \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
        \else\ifx\EmFi@type\EmFi@S@number
          \kv@define@key{EmFi}{\EmFi@key.value}{%
            \EmFi@itemtrue
            \expandafter\EdefSanitize\csname EmFi@V@#1\endcsname{ ##1}%
          }%
          \EmFi@GlobalKey{EmFi}{\EmFi@key.value}%
        \fi\fi\fi
        \kv@define@key{EmFi}{\EmFi@key.prefix}{%
          \EmFi@itemtrue
          \expandafter\def\csname EmFi@P@#1\endcsname{##1}%
        }%
        \EmFi@GlobalKey{EmFi}{\EmFi@key.prefix}%
        \kv@define@key{EmFiSo}{\EmFi@key}[ascending]{%
          \EdefSanitize\EmFi@temp{##1}%
          \ifx\EmFi@temp\EmFi@S@ascending
            \def\EmFi@temp{true}%
          \else\ifx\EmFi@temp\EmFi@S@descending
            \def\EmFi@temp{false}%
          \else
            \def\EmFi@temp{}%
            \EmFi@Error{%
              Unknown sort order `\EmFi@temp'.\MessageBreak
              Supported values: `\EmFi@S@ascending', %
              `\EmFi@S@descending
            }\@ehc
          \fi\fi
          \ifx\EmFi@temp\ltx@empty
          \else
            \xdef\EmFi@sortkeys{%
              \EmFi@sortkeys
              /\pdf@escapename{#1}%
            }%
            \ifx\EmFi@sortorders\ltx@empty
              \global\let\EmFi@sortorders\EmFi@temp
              \gdef\EmFi@sortcase{1}%
            \else
              \xdef\EmFi@sortorders{%
                \EmFi@sortorders
                \ltx@space
                \EmFi@temp
              }%
              \xdef\EmFi@sortcase{2}%
            \fi
          \fi
        }%
        \EmFi@GlobalDefaultKey{EmFiSo}\EmFi@key
      \endgroup
    \else
      \EmFi@Error{%
        Field `\EmFi@key' is already defined%
      }\@ehc
    \fi
  \fi
}
```

```
\kv@define@key{EmFiFi}{type}{%
  \EdefSanitize\EmFi@temp{#1}%
  \ifx\EmFi@temp\EmFi@S@text
    \let\EmFi@type\EmFi@temp
  \else\ifx\EmFi@temp\EmFi@S@date
    \let\EmFi@type\EmFi@temp
  \else\ifx\EmFi@temp\EmFi@S@number
    \let\EmFi@type\EmFi@temp
  \else\ifx\EmFi@temp\EmFi@S@file
    \let\EmFi@type\EmFi@temp
  \else\ifx\EmFi@temp\EmFi@S@desc
    \let\EmFi@type\EmFi@temp
  \else\ifx\EmFi@temp\EmFi@S@afrelationship
    \let\EmFi@type\EmFi@temp
  \else\ifx\EmFi@temp\EmFi@S@moddate
    \let\EmFi@type\EmFi@temp
  \else\ifx\EmFi@temp\EmFi@S@creationdate
    \let\EmFi@type\EmFi@temp
  \else\ifx\EmFi@temp\EmFi@S@size
    \let\EmFi@type\EmFi@temp
  \else
    \EmFi@Error{%
      Unknown type `\EmFi@temp'.\MessageBreak
      Supported types: `text', `date', `number', `file',\MessageBreak
      `desc', `afrelationship', `moddate', `creationdate', `size'%
    }%
  \fi\fi\fi\fi\fi\fi\fi\fi\fi
}
\kv@define@key{EmFiFi}{title}{%
  \def\EmFi@title{#1}%
}
```

\EmFi@setboolean

```
\def\EmFi@setboolean#1#2{%
  \EdefSanitize\EmFi@temp{#2}%
  \ifx\EmFi@temp\EmFi@S@true
    \csname EmFi@#1true\endcsname
  \else
    \ifx\EmFi@temp\EmFi@S@false
      \csname EmFi@#1false\endcsname
    \else
      \EmFi@Error{%
        Unknown value `\EmFi@temp' for key `#1'.\MessageBreak
        Supported values: `true', `false'%
      }\@ehc
    \fi
  \fi
}
\kv@define@key{EmFiFi}{visible}[true]{%
  \EmFi@setboolean{visible}{#1}%
}
\kv@define@key{EmFiFi}{edit}[true]{%
  \EmFi@setboolean{edit}{#1}%
}
```

\EmFi@sortkeys

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

\EmFi@sortorders

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

\embedfilesort

```
\def\embedfilesort{%
  \kvsetkeys{EmFiSo}%
}
```

2.8 Embed the file

\embedfile

```
\def\embedfile{%
  \ltx@ifnextchar[\EmFi@embedfile{\EmFi@embedfile[]}%
}
```

\EmFi@embedfile

```
\def\EmFi@removeslash#1{\if/#1\else#1\fi}%
\def\EmFi@embedfile[#1]#2{%
  \ifEmFi@finished
    \EmFi@Error{%
      \string\embedfile\ltx@space after \string\embedfilefinish
    }{%
      The list of embedded files is already written.%
    }%
  \else
    \begingroup
      \def\EmFi@file{#2}%
      \kvsetkeys{EmFi}{#1}%
      \expandafter\expandafter\expandafter
      \ifx\expandafter\expandafter\expandafter
          \\\pdf@filesize{\EmFi@file}\\%
        \EmFi@Error{%
          File `\EmFi@file' not found%
        }{%
          The unknown file is not embedded.%
        }%
      \else
        \edef\EmFi@@filespec{%
          \pdf@escapestring{\EmFi@filespec}%
        }%
        \ifx\EmFi@ucfilespec\ltx@empty
          \EmFi@convert\EmFi@filespec\EmFi@@ucfilespec
        \else
          \EmFi@convert\EmFi@ucfilespec\EmFi@@ucfilespec
        \fi
        \ifx\EmFi@desc\ltx@empty
          \let\EmFi@@desc\ltx@empty
        \else
          \EmFi@convert\EmFi@desc\EmFi@@desc
        \fi
        \ifx\EmFi@afrelationship\ltx@empty
          \let\EmFi@@afrelationship\ltx@empty
        \else
          \expandafter\edef\expandafter\EmFi@@afrelationship\expandafter
            {\expandafter\EmFi@removeslash\EmFi@afrelationship}
        \fi
        \ifEmFi@item
          \let\do\EmFi@do
          \emFi@pdfwritedict{\EmFi@fieldlist}%
          \edef\EmFi@ci{\emFi@pdfobjectreflast}%
        \fi
        \emFi@pdfwritefstream
         {%
          /Type/EmbeddedFile%
          \ifx\EmFi@mimetype\ltx@empty
```

```
          \else
            /Subtype/\pdf@escapename{\EmFi@mimetype}%
          \fi
          /Params<<%
            /ModDate(\pdf@filemoddate{\EmFi@file})%
            /Size \pdf@filesize{\EmFi@file}%
            /CheckSum<\pdf@filemdfivesum{\EmFi@file}>%
          >>%
        }{\EmFi@file}\relax
        \EmFi@defobj{EmbeddedFile}%
        \emFi@pdfwritedict{%
            /Type/Filespec%
            \ifx\EmFi@filesystem\ltx@empty
            \else
            /FS/\pdf@escapename{\EmFi@filesystem}%
            \fi
            /F(\EmFi@@filespec)%
            /UF(\EmFi@@ucfilespec)%
            \ifx\EmFi@@desc\ltx@empty
            \else
              /Desc(\EmFi@@desc)%
            \fi
            \ifx\EmFi@@afrelationship\ltx@empty
            \else
              /AFRelationship/\pdf@escapename{\EmFi@@afrelationship}%
            \fi
            /EF<<%
              /F \emFi@pdfobjectreflast%
            >>%
            \ifEmFi@item
              /CI \EmFi@ci%
            \fi
        }%
        \EmFi@defobj{Filespec}%
        \EmFi@add{%
          \EmFi@@filespec
        }{\emFi@pdfobjectreflast}%
      \fi
    \endgroup
  \fi
}
```

\EmFi@do

```
\def\EmFi@do#1{%
  \expandafter\ifx\csname EmFi@P@#1\endcsname\relax
    \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
    \else
      /\pdf@escapename{#1}\csname EmFi@V@#1\endcsname
    \fi
  \else
    /\pdf@escapename{#1}<<%
      \expandafter\ifx\csname EmFi@V@#1\endcsname\relax
      \else
        /D\csname EmFi@V@#1\endcsname
      \fi
      /P(\csname EmFi@P@#1\endcsname)%
    >>%
  \fi
}
```

\EmFi@convert

```
\def\EmFi@convert#1#2{%
  \ifnum\pdf@strcmp{\EmFi@stringmethod}{psd}=0 %
```

```
    \pdfstringdef\EmFi@temp{#1}%
    \let#2\EmFi@temp
  \else
    \edef#2{\pdf@escapestring{#1}}%
  \fi
}

\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, ...).

```
\def\EmFi@add#1#2{%
  \begingroup
    \ifx\EmFi@list\ltx@empty
      \xdef\EmFi@list{\noexpand\do{#1}{#2}}%
    \else
      \def\do##1##2{%
        \ifnum\pdf@strcmp{##1}{#1}>0 %
          \edef\x{%
            \toks@{%
              \the\toks@%
              \noexpand\do{#1}{#2}%
              \noexpand\do{##1}{##2}%
            }%
          }%
          \x
          \def\do####1####2{%
            \toks@\expandafter{\the\toks@\do{####1}{####2}}%
          }%
          \def\stop{%
            \xdef\EmFi@list{\the\toks@}%
          }%
        \else
          \toks@\expandafter{\the\toks@\do{##1}{##2}}%
        \fi
      }%
      \def\stop{%
        \xdef\EmFi@list{\the\toks@\noexpand\do{#1}{#2}}%
      }%
      \toks@{}%
      \EmFi@list\stop
    \fi
  \endgroup
}
```

`\embedfilefinish`

```
\ifcsname ProvidesPackage\endcsname
\ExplSyntaxOn
\sys_ensure_backend:
\IfPDFManagementActiveTF
 {
   \cs_new_eq:NN\emFi@pdfmanagementadd\pdfmanagement_add:nne
 }
 {}
\cs_new_eq:NN\emFi@pdfobjectreflast\pdf_object_ref_last:
\cs_new:Npn\emFi@pdfwritedict#1{\pdf_object_unnamed_write:ne{dict}{#1}}
\cs_new:Npn\emFi@pdfwritearray#1{\pdf_object_unnamed_write:ne{array}{#1}}
\cs_new:Npn\emFi@pdfwritefstream#1#2{\pdf_object_unnamed_write:ne{fstream}{{#1}{#2}}}
\ExplSyntaxOff
\else
```

```
\def\IfPDFManagementActiveTF#1#2{#2}
\def\emFi@pdfobjectreflast{\the\pdflastobj\ltx@space 0 R}
\def\emFi@pdfwritedict#1{\immediate\pdfobj{<<#1>>}}
\def\emFi@pdfwritearray#1{\immediate\pdfobj{[#1]}}
\def\emFi@pdfwritefstream#1#2{\immediate\pdfobj stream attr{#1}file{#2}}
\fi
\def\embedfilefinish{%
  \ifEmFi@finished
    \EmFi@Error{%
      Too many invocations of \string\embedfilefinish
    }{%
      The list of embedded files is already written.%
    }%
  \else
    \ifx\EmFi@list\ltx@empty
    \else
```

Write `/EmbeddedFiles` entry.

```
      \global\EmFi@finishedtrue
      \begingroup
      \IfPDFManagementActiveTF
         {%
          \def\do##1##2{%
           \emFi@pdfmanagementadd{Catalog/Names}
             {EmbeddedFiles}{##2}}%
          \EmFi@list
         }
         {%
          \def\do##1##2{%
           (##1)##2%
           }%
          \emFi@pdfwritedict{/Names[\EmFi@list]}
           \pdfnames{%
          /EmbeddedFiles \emFi@pdfobjectreflast%
         }}%
      \endgroup
      \begingroup
        \IfPDFManagementActiveTF
         {%
          \def\do##1##2{%
           \emFi@pdfmanagementadd{Catalog}{AF}{##2}}%
          \EmFi@list
         }
         {%
           \def\do##1##2{%
            \ltx@space##2%
            }%
           \emFi@pdfwritearray{\EmFi@list}%
           \pdfcatalog{%
            /AF \emFi@pdfobjectreflast%
          }%
         }
      \endgroup
```

Write collection objects.

```
      \ifx\EmFi@initialfile\ltx@empty
      \else
        \EmFi@collectiontrue
      \fi
      \ifEmFi@collection
        \ifx\EmFi@initialfile\ltx@empty
          \let\EmFi@@initialfile\ltx@empty
        \else
          \edef\EmFi@@initialfile{%
```

```
      \pdf@escapestring{\EmFi@initialfile}%
    }%
  \fi
```

Look for initial file among the embedded files.

```
  \begingroup
    \let\f=N%
    \def\do##1##2{%
      \def\x{##1}%
      \ifx\x\EmFi@@initialfile
        \let\f=Y%
        \let\do\ltx@gobbletwo
      \fi
    }%
    \EmFi@list
  \expandafter\endgroup
  \ifx\f Y%
  \else
    \@PackageWarningNoLine{embedfile}{%
      Missing initial file `\EmFi@initialfile'\MessageBreak
      among the embedded files%
    }%
    \let\EmFi@initialfile\ltx@empty
    \let\EmFi@@initialfile\ltx@empty
  \fi
  \ifcase\EmFi@sortcase
    \def\EmFi@temp{}%
  \or
    \def\EmFi@temp{%
      /S\EmFi@sortkeys
      /A \EmFi@sortorders
    }%
  \else
    \def\EmFi@temp{%
      /S[\EmFi@sortkeys]%
      /A[\EmFi@sortorders]%
    }%
  \fi
  \def\EmFi@@order##1{%
    \ifnum\EmFi@order>1 %
      /O ##1%
    \fi
  }%
  \emFi@pdfwritedict{%
      \ifx\EmFi@schema\ltx@empty
      \else
        /Schema<<\EmFi@schema>>%
      \fi
      \ifx\EmFi@@initialfile\ltx@empty
      \else
        /D(\EmFi@@initialfile)%
      \fi
      \ifx\EmFi@view\EmFi@S@tile
        /View/T%
      \else\ifx\EmFi@view\EmFi@S@hidden
        /View/H%
      \fi\fi
      \ifx\EmFi@temp\ltx@empty
        \EmFi@temp
      \else
        /Sort<<\EmFi@temp>>%
      \fi
  }
```

```
        \IfPDFManagementActiveTF
         {%
           \emFi@pdfmanagementadd{Catalog}{Collection}{\emFi@pdfobjectreflast}%
         }%
         {%
           \pdfcatalog{%
            /Collection \emFi@pdfobjectreflast%
         }%
        }%
      \fi
    \fi
  \fi
}

\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname AtEndDocument\endcsname\relax
\else
  \AtEndDocument{\embedfilefinish}%
\fi

\EmFi@AtEnd%
⟨/package⟩
```

3 Installation

3.1 Download

Package. This package is available on CTAN[1]:

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 TEX 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 TEX:

```
tex embedfile.dtx
```

[1] CTAN:pkg/embedfile

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 `mktexlsr`.

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 pdfLATEX:

```
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 iniTeX.
- Package keyval replaced by packages kvsetkeys and kvdefinekeys because of compatibility for iniTeX.
- 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.

[2019/12/03 v2.9]

- add `/AF` and `/AFrelationship` keys (Andreas Karrenbauer)
- add `\pdfcatalog` emulation for LuaTeX.
- 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 compatibility (PR#7)
- Adapted the error message to luatex
- The /UF key in the filespec dictionary is now set by default (PR#6)

[2025-10-27 v2.13]

- Add support for PDF management

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 236
\@PackageWarningNoLine 823
\@ehc 373, 495, 521, 567
\@undefined 168, 239, 320
\\ 599

A

\advance 411
\aftergroup 139
\AtEndDocument 884

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@ 410, 411, 412
\cs 738, 741, 742, 743, 744
\csname 124, 131, 160, 176, 186, 225, 262, 272, 298, 299, 304, 305, 309, 311, 336, 342, 350, 408, 454, 465, 474, 480, 559, 562, 676, 677, 679, 683, 685, 687, 882

D

\do 444, 447, 626, 703, 705, 710, 711, 715, 716, 722, 726, 767, 773, 784, 789, 812, 816
\documentclass 41

E

\EdefSanitize 272, 356, 407, 474, 484, 526, 557

\embedfile 3, 16, 21, 25, 79, 85, 91, 105, 582, 589
\embedfilefield 4, 50, 54, 58, 62, 66, 382, 398
\embedfilefinish 4, 34, 382, 401, 589, 733, 884
\embedfilegetobject 5, 348
\embedfileifobjectexists . 5, 341, 349
\embedfilesetup 4, 4, 11, 46, 378
\embedfilesort 5, 71, 579
\EmFi@@afrelationship 620, 622, 656, 658
\EmFi@@desc 615, 617, 652, 654
\EmFi@@filespec 606, 650, 669
\EmFi@@initialfile 804, 806, 814, 828, 853, 855
\EmFi@@order 395, 434, 843
\EmFi@@ucfilespec 610, 612, 651
\EmFi@add 668, 700
\EmFi@afrelationship 619, 623
\EmFi@AtEnd 205, 206, 223, 255, 268, 886
\EmFi@ci 628, 664
\EmFi@collectiontrue . . 358, 406, 800
\EmFi@convert 418, 453, 464, 610, 612, 617, 691
\EmFi@DefineKey 307, 313, 314, 315, 316, 317, 318, 319, 377
\EmFi@defobj 334, 643, 667
\EmFi@desc 614, 617
\EmFi@details 274
\EmFi@do 626, 675
\EmFi@editfalse 416
\EmFi@embedfile 583, 585
\EmFi@Error 232, 249, 263, 370, 381, 400, 491, 519, 546, 564, 588, 600, 755
\EmFi@fieldlist . . . 396, 445, 446, 627
\EmFi@file 314, 595, 599, 601, 638, 639, 640, 642
\EmFi@filespec 607, 610
\EmFi@filesystem 646, 648
\EmFi@finishedtrue 763
\EmFi@GlobalDefaultKey 301, 516
\EmFi@GlobalKey 297, 302, 459, 470, 476, 482
\EmFi@hidden 276
\EmFi@id 331, 336
\EmFi@idtrue 332
\EmFi@initialfile 798, 803, 807, 824, 827
\EmFi@itemtrue 451, 462, 473, 479
\EmFi@key 407, 408, 413, 421, 447, 450, 459, 461, 470, 472, 476, 478, 482, 483, 516, 520
\EmFi@list 699, 702, 703, 719, 726, 729, 761, 770, 776, 786, 792, 819
\EmFi@mimetype 633, 635
\EmFi@next 357, 369, 375, 380, 387, 391
\EmFi@order . . . 394, 410, 412, 434, 844
\emFi@pdfmanagementadd 738, 768, 785, 870
\emFi@pdfobjectreflast 337, 628, 661, 670, 741, 748, 778, 794, 870, 874
\emFi@pdfwritearray . . . 743, 750, 792
\emFi@pdfwritedict 627, 644, 742, 749, 776, 848
\emFi@pdfwritefstream . 630, 744, 751
\EmFi@removeslash 585, 623
\EmFi@RequirePackage 224, 232, 246, 257, 258, 259, 260, 270
\EmFi@S@afrelationship 282, 427, 537
\EmFi@S@ascending 286, 485, 493
\EmFi@S@creationdate . . 284, 429, 541
\EmFi@S@date 278, 423, 460, 529
\EmFi@S@desc 281, 426, 535
\EmFi@S@descending 287, 487, 494
\EmFi@S@details 361, 362, 363
\EmFi@S@false 289, 561
\EmFi@S@file 280, 425, 533
\EmFi@S@hidden 366, 367, 859
\EmFi@S@moddate 283, 428, 539
\EmFi@S@number 279, 424, 471, 531
\EmFi@S@size 285, 430, 543
\EmFi@S@text 277, 414, 449, 527
\EmFi@S@tile 364, 365, 857
\EmFi@S@true 288, 558
\EmFi@schema . . 393, 419, 420, 849, 851
\EmFi@setboolean 556, 572, 575
\EmFi@sortcase 397, 505, 512, 830
\EmFi@sortkeys 499, 500, 577, 834, 839
\EmFi@sortorders 503, 504, 507, 508, 578, 835, 840
\EmFi@stringmethod 692
\EmFi@temp 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, 693, 694, 831, 833, 838, 862, 863, 865
\EmFi@tile 275
\EmFi@title 413, 418, 433, 554
\EmFi@type 414, 423, 424, 425, 426, 427, 428, 429, 430, 449, 460, 471, 528, 530, 532, 534, 536, 538, 540, 542, 544
\EmFi@ucfilespec 609, 612
\EmFi@view 361, 363, 365, 367, 857, 859
\EmFi@visibletrue 415
\empty 127, 128
\end 97
\endcsname 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, 676, 677, 679, 683, 685, 687, 733, 882

\endinput 139, 223
\endlinechar . . 114, 145, 181, 187, 199
\ExplSyntaxOff 745
\ExplSyntaxOn 734

F

\f 811, 815, 821

G

\gdef 394, 505
\Gin@driver 5

I

\if 585
\ifcase 830
\ifcsname 233, 733
\ifEmFi@collection 290, 802
\ifEmFi@edit 293, 439
\ifEmFi@finished 295, 379, 399, 587, 754
\ifEmFi@id 296, 335
\ifEmFi@item 294, 625, 663
\ifEmFi@sort 291
\ifEmFi@visible 292, 435
\ifnum 692, 706, 844
\ifpdf 247
\IfPDFManagementActiveTF 736, 747, 765, 782, 868
\ifx 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, 633, 646, 652, 656, 676, 677, 683, 702, 761, 798, 803, 814, 821, 849, 853, 857, 859, 862, 882
\immediate . . . 133, 162, 749, 750, 751
\input 4, 6, 7, 227

J

\jobname 95, 103, 108, 109

K

\kv@define@key 308, 330, 355, 450, 461, 472, 478, 483, 525, 553, 571, 574
\kvsetkeys 388, 417, 580, 596

L

\ltx@empty 360, 497, 503, 609, 614, 615, 619, 620, 633, 646, 652, 656, 699, 702, 761, 798, 803, 804, 827, 828, 849, 853, 862
\ltx@firstoftwo 345
\ltx@gobbletwo 816
\ltx@ifnextchar 583
\ltx@newif 290, 291, 292, 293, 294, 295, 296
\ltx@secondoftwo 343
\ltx@space 382, 401, 509, 589, 748, 790

M

\MessageBreak 371, 492, 547, 548, 565, 824

N

\NeedsTeXFormat 40, 101
\numexpr 240

O

\outputmode 244

P

\PackageInfo 136
\pdf 741, 742, 743, 744
\pdf@escapename 421, 501, 635, 648, 658, 679, 682
\pdf@escapestring 607, 696, 807
\pdf@filemdfivesum 640
\pdf@filemoddate 638
\pdf@filesize 599, 639
\pdf@strcmp 692, 706
\pdfcatalog 243, 793, 873
\pdfextension 239, 241, 242, 243
\pdffeedback 240
\pdflastobj 240, 748
\pdfmanagement 738
\pdfnames 241, 777
\pdfobj 242, 749, 750, 751
\pdfoutput 244
\pdfstringdef 43, 320, 323, 693
\protected 240, 241, 242, 243
\ProvidesPackage 102, 129, 177

R

\RequirePackage 104, 230
\resetatcatcode 8

S

\stop 718, 725, 729
\sys 735

T

\the 187, 188, 189, 190, 191, 192, 193, 194, 207, 412, 709, 716, 719, 722, 726, 748
\TMP@EnsureCode 204, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222
\toks@ 708, 709, 716, 719, 722, 726, 728

U

\usepackage 42, 45

W

\write 133, 162

X

\x 124, 125, 128, 132, 136, 138, 161, 166, 176, 185, 197, 707, 714, 813, 814