The embedfile package

Heiko Oberdiek[*]
<heiko.oberdiek at googlemail.com>

2016/05/15 v2.7

Abstract

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

Contents

[*] Please report any issues at https://github.com/ho-tex/oberdiek/issues

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 meantioned 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. 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 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 embeded files in the left panel and adds a new column for the compressed size.

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

Collection support is enabled automatically, if it is used.

`\embedfilesetup {⟨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}[2016/05/15]
\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:macros/latex/contrib/oberdiek/. It also serves as small example for the use of the package with LaTeX.

```
⟨*dtxattach⟩
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{dtx-attach}
  [2016/05/15 v2.7 Embed \string\jobname.dtx (HO)]%
\RequirePackage{embedfile}[2016/05/15]
\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}%
  [2016/05/15 v2.7 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]%
\def\EmFi@Error{%
  \@PackageError{embedfile}%
}
```

Luatex compat

```
\ifx\pdfextension\@undefined\else
  \EmFi@RequirePackage{luatex85}[2016/01/01]
\fi
```

2.4 Check for recent pdfTeX in PDF mode

Load package ifpdf and check mode.

```
\EmFi@RequirePackage{ifpdf}[2007/09/09]
\ifpdf
\else
  \EmFi@Error{%
    Missing pdfTeX 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@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}{}
```

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{%
      \the\pdflastobj\ltx@space 0 R%
    }%
  \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@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
        /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@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', `moddate', `creationdate', `size'%
    }%
  \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@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
        \let\EmFi@@ucfilespec\ltx@empty
      \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
      \ifEmFi@item
        \let\do\EmFi@do
        \immediate\pdfobj{%
          <<%
            \EmFi@fieldlist
          >>%
        }%
        \edef\EmFi@ci{\the\pdflastobj}%
      \fi
      \immediate\pdfobj stream attr{%
        /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}>%
        >>%
      }file{\EmFi@file}\relax
      \EmFi@defobj{EmbeddedFile}%
      \immediate\pdfobj{%
        <<%
          /Type/Filespec%
          \ifx\EmFi@filesystem\ltx@empty
          \else
```

```
            /FS/\pdf@escapename{\EmFi@filesystem}%
            \fi
            /F(\EmFi@@filespec)%
            \ifx\EmFi@@ucfilespec\ltx@empty
            \else
              /UF(\EmFi@@ucfilespec)%
            \fi
            \ifx\EmFi@@desc\ltx@empty
            \else
              /Desc(\EmFi@@desc)%
            \fi
            /EF<<%
              /F \the\pdflastobj\ltx@space 0 R%
            >>%
            \ifEmFi@item
              /CI \EmFi@ci\ltx@space 0 R%
            \fi
          >>%
        }%
        \EmFi@defobj{Filespec}%
        \EmFi@add{%
          \EmFi@@filespec
        }{\the\pdflastobj\ltx@space 0 R}%
      \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

```
\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
        \def\do##1##2{%
          (##1)##2%
        }%
        \immediate\pdfobj{%
          <<%
```

```
       /Names[\EmFi@list]%
      >>%
     }%
     \pdfnames{%
      /EmbeddedFiles \the\pdflastobj\ltx@space 0 R%
     }%
    \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
        }%
        \immediate\pdfobj{%
          <<%
            \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
          >>%
        }%
        \pdfcatalog{%
          /Collection \the\pdflastobj\ltx@space0 R%
        }%
      \fi
    \fi
  \fi
}
```

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

```
\EmFi@AtEnd%
⟨/package⟩
```

3 Test

3.1 Catcode checks for loading

```
⟨*test1⟩
```

```
\catcode`\{=1 %
\catcode`\}=2 %
\catcode`\#=6 %
\catcode`\@=11 %
\expandafter\ifx\csname count@\endcsname\relax
  \countdef\count@=255 %
\fi
\expandafter\ifx\csname @gobble\endcsname\relax
  \long\def\@gobble#1{}%
\fi
\expandafter\ifx\csname @firstofone\endcsname\relax
  \long\def\@firstofone#1{#1}%
```

```
\fi
\expandafter\ifx\csname loop\endcsname\relax
  \expandafter\@firstofone
\else
  \expandafter\@gobble
\fi
{%
  \def\loop#1\repeat{%
    \def\body{#1}%
    \iterate
  }%
  \def\iterate{%
    \body
      \let\next\iterate
    \else
      \let\next\relax
    \fi
    \next
  }%
  \let\repeat=\fi
}%
\def\RestoreCatcodes{}
\count@=0 %
\loop
  \edef\RestoreCatcodes{%
    \RestoreCatcodes
    \catcode\the\count@=\the\catcode\count@\relax
  }%
\ifnum\count@<255 %
  \advance\count@ 1 %
\repeat

\def\RangeCatcodeInvalid#1#2{%
  \count@=#1\relax
  \loop
    \catcode\count@=15 %
  \ifnum\count@<#2\relax
    \advance\count@ 1 %
  \repeat
}
\def\RangeCatcodeCheck#1#2#3{%
  \count@=#1\relax
  \loop
    \ifnum#3=\catcode\count@
    \else
      \errmessage{%
        Character \the\count@\space
        with wrong catcode \the\catcode\count@\space
        instead of \number#3%
      }%
    \fi
  \ifnum\count@<#2\relax
    \advance\count@ 1 %
  \repeat
}
\def\space{ }
\expandafter\ifx\csname LoadCommand\endcsname\relax
  \def\LoadCommand{\input embedfile.sty\relax}%
```

```
\fi
\def\Test{%
  \RangeCatcodeInvalid{0}{47}%
  \RangeCatcodeInvalid{58}{64}%
  \RangeCatcodeInvalid{91}{96}%
  \RangeCatcodeInvalid{123}{255}%
  \catcode`\@=12 %
  \catcode`\\=0 %
  \catcode`\%=14 %
  \LoadCommand
  \RangeCatcodeCheck{0}{36}{15}%
  \RangeCatcodeCheck{37}{37}{14}%
  \RangeCatcodeCheck{38}{47}{15}%
  \RangeCatcodeCheck{48}{57}{12}%
  \RangeCatcodeCheck{58}{63}{15}%
  \RangeCatcodeCheck{64}{64}{12}%
  \RangeCatcodeCheck{65}{90}{11}%
  \RangeCatcodeCheck{91}{91}{15}%
  \RangeCatcodeCheck{92}{92}{0}%
  \RangeCatcodeCheck{93}{96}{15}%
  \RangeCatcodeCheck{97}{122}{11}%
  \RangeCatcodeCheck{123}{255}{15}%
  \RestoreCatcodes
}
\Test
\csname @@end\endcsname
\end
```

⟨/test1⟩

3.2 Simple test

⟨*test2⟩

```
\input embedfile.sty\relax
\embedfile[%
  stringmethod=escape,%
  mimetype=plain/text,%
  desc={LaTeX docstrip source archive for package `embedfile'},%
  id={embedfile.dtx}%
]{embedfile.dtx}
\nopagenumbers
Test (plain-\TeX): {\tt embedfile.dtx} should be embedded.%

\def\Test#1{%
  \par
  \embedfileifobjectexists{embedfile.dtx}{#1}{%
    Object #1 (embedfile.dtx): %
    \embedfilegetobject{embedfile.dtx}{#1}%
  }{%
    \errmessage{Missing object #1 (embedfile.dtx)}%
  }%
}
\Test{EmbeddedFile}
\Test{Filespec}
\embedfilefinish
\bye
```

⟨/test2⟩

⟨*test3⟩

```
\NeedsTeXFormat{LaTeX2e}
```

```
\let\SavedJobname\jobname
\def\jobname{embedfile}
\RequirePackage{dtx-attach}[2016/05/15]
\let\jobname\SavedJobname
\documentclass{minimal}
\begin{document}
  Test (\LaTeX): \texttt{embedfile.dtx} should be embedded.%
\end{document}
⟨/test3⟩
```

3.3 Test for ini-TEX

```
⟨*test4⟩
\catcode`\{=1 %
\catcode`\}=2 %
\input ifluatex.sty %
\ifluatex
  \directlua{%
    tex.enableprimitives('', {%
      'pdflastobj',%
      'pdfnames',%
      'pdfobj',%
      'pdfoutput'%
    })%
  }%
  \ifx\pdfextension\undefined\else
    \protected\def\pdflastobj {\numexpr\pdffeedback lastobj\relax}
    \protected\def\pdfnames   {\pdfextension names }
    \protected\def\pdfobj     {\pdfextension obj }
    \let\pdfoutput            \outputmode
  \fi
\fi
\pdfoutput=1 %
\input embedfile.sty %
\shipout\hbox{}
\embedfile[%
  stringmethod=escape,%
  mimetype=plain/text,%
  desc={iniTeX source},%
]{\jobname.tex}
\embedfilefinish
\end
⟨/test4⟩
```

4 Installation

4.1 Download

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

CTAN:macros/latex/contrib/oberdiek/embedfile.dtx The source file.

CTAN:macros/latex/contrib/oberdiek/embedfile.pdf Documentation.

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

[1] http://ctan.org/pkg/embedfile

CTAN:install/macros/latex/contrib/oberdiek.tds.zip

TDS refers to the standard "A Directory Structure for TEX Files" (CTAN:tds/tds.pdf). Directories with `texmf` in their name are usually organized this way.

4.2 Bundle installation

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

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

Script installation. Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package attachfile2 comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

4.3 Package installation

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

```
tex embedfile.dtx
```

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

embedfile.sty	→ tex/generic/oberdiek/embedfile.sty
dtx-attach.sty	→ tex/generic/oberdiek/dtx-attach.sty
embedfile.pdf	→ doc/latex/oberdiek/embedfile.pdf
embedfile-example-plain.tex	→ doc/latex/oberdiek/embedfile-example-plain.tex
embedfile-example-collection.tex	→ doc/latex/oberdiek/embedfile-example-collection.tex
test/embedfile-test1.tex	→ doc/latex/oberdiek/test/embedfile-test1.tex
test/embedfile-test2.tex	→ doc/latex/oberdiek/test/embedfile-test2.tex
test/embedfile-test3.tex	→ doc/latex/oberdiek/test/embedfile-test3.tex
test/embedfile-test4.tex	→ doc/latex/oberdiek/test/embedfile-test4.tex
embedfile.dtx	→ source/latex/oberdiek/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.

4.4 Refresh file name databases

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

4.5 Some details for the interested

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

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

Unpacking with 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
```

5 Catalogue

The following XML file can be used as source for the TeX Catalogue. The elements caption and description are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is embedfile.xml.

```
⟨*catalogue⟩
<?xml version='1.0' encoding='us-ascii'?>
<!DOCTYPE entry SYSTEM 'catalogue.dtd'>
<entry datestamp='$Date$' modifier='$Author$' id='embedfile'>
  <name>embedfile</name>
  <caption>Embed files into PDF.</caption>
  <authorref id='auth:oberdiek'/>
  <copyright owner='Heiko Oberdiek' year='2006-2011'/>
  <license type='lppl1.3'/>
  <version number='2.7'/>
  <description>
    This package embeds files in a PDF document, using the PDF
    format's embedding operation (note the contrast with the attach
    operation used by the <xref refid='attachfile'>attachfile</xref>
    and <xref refid='attachfile2'>attachfile2</xref> packages).
    Currently only <xref refid='pdftex'>pdfTeX</xref> &gt;=1.30, in
    PDF mode, is supported.

    <p/>
    The package is part of the <xref refid='oberdiek'>oberdiek</xref>
    bundle.
  </description>
  <documentation details='Package documentation'
      href='ctan:/macros/latex/contrib/oberdiek/embedfile.pdf'/>
  <ctan file='true' path='/macros/latex/contrib/oberdiek/embedfile.dtx'/>
  <miktex location='oberdiek'/>
  <texlive location='oberdiek'/>
  <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip'/>
</entry>
⟨/catalogue⟩
```

6 References

[1] Scott Pakin: *The attachfile package*; 2005/02/20 v1.2; CTAN:macros/latex/contrib/attachfile/.

[2] Heiko Oberdiek: *The attachfile2 package*; 2006/08/16 v2.2; CTAN:macros/latex/contrib/oberdiek/attachfile2.pdf.

[3] Adobe Systems Incorporated: *PDF Reference, Sixth Edition, Version 1.7*, Oktober 2006; http://www.adobe.com/devnet/pdf/pdf_reference.html.

[4] Network Working Group: RFC 2046, *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*, November 1996; http://www.rfc-editor.org/.

[5] IANA (Internet Assigned Numbers Authority): *MIME Media Types*, May 2006; http://www.iana.org/assignments/media-types/.

7 History

[2006/08/16 v1.0]

- First public version.

[2007/04/11 v1.1]

- Line ends sanitized.

[2007/09/09 v1.2]

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

[2007/10/28 v2.0]

- Collection support added (PDF 1.7).

[2007/10/29 v2.1]

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

[2007/11/11 v2.2]

- Use of package pdftexcmds for LuaTEX support.

[2007/11/25 v2.3]

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

[2009/09/25 v2.4]

- Bug fix: If hyperref is used with option unicode, the Unicode encoded file name causes trouble. Therefore `\pdfstringdef` is now never used for option filespec, always method `escape` is applied (Peter Cibulka).
- Bug fix for initialfile.
- Bug fix for file names in `/EmbeddedFiles`.
- New option ucfilespec for file name support in Unicode (since PDF 1.7).

[2010/03/01 v2.5]

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

[2011/04/13 v2.6]

- Docu fixes (thanks Hans-Martin Münch).

[2016/05/15 v2.7]

- luaTeX compatibility

8 Index

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