

# The ifpdf package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2007/12/12 v1.6

## Abstract

This package looks for pdfTeX in pdf mode and implements and sets the switch `\ifpdf`. The detection is based on `\pdfoutput` and the package will not change this value. It works with plain or L<sup>A</sup>T<sub>E</sub>X formats.

## Contents

<b>1</b>	<b>Documentation</b>	<b>2</b>
1.1	Introduction . . . . .	2
1.2	Usage . . . . .	2
1.3	Specification . . . . .	3
1.4	Future . . . . .	3
<b>2</b>	<b>Implementation</b>	<b>3</b>
2.1	Reload check and package identification . . . . .	3
2.2	Catcodes . . . . .	4
2.3	Check for previously defined <code>\ifpdf</code> . . . . .	5
2.4	<code>\ifpdf</code> . . . . .	5
2.5	Protocol entry . . . . .	6
<b>3</b>	<b>Test</b>	<b>6</b>
3.1	Catcode checks for loading . . . . .	6
<b>4</b>	<b>Installation</b>	<b>7</b>
4.1	Download . . . . .	7
4.2	Bundle installation . . . . .	8
4.3	Package installation . . . . .	8
4.4	Refresh file name databases . . . . .	8
4.5	Some details for the interested . . . . .	8
<b>5</b>	<b>History</b>	<b>9</b>
	[2001/06/14 v1.0] . . . . .	9
	[2001/07/14 v1.1] . . . . .	9
	[2001/09/26 v1.2] . . . . .	9
	[2005/07/22 v1.3] . . . . .	9
	[2006/02/20 v1.4] . . . . .	9
	[2007/09/09 v1.5] . . . . .	9
	[2007/12/12 v1.6] . . . . .	9
<b>6</b>	<b>Index</b>	<b>10</b>

# 1 Documentation

## 1.1 Introduction

It is commonly known that Hàn Thê Thành's pdfT<sub>E</sub>X generates PDF output directly and many people uses pdfT<sub>E</sub>X for this purpose. However the DVI output was never thrown away. In contrary, he new features for typesetting that works in both PDF and DVI mode.

In the meantime many T<sub>E</sub>X distributions replace the traditional T<sub>E</sub>X binary with pdfT<sub>E</sub>X. Then, for example, called as `latex` pdfT<sub>E</sub>X works in DVI mode with the L<sup>A</sup>T<sub>E</sub>X format preloaded, called as `pdflatex` pdfT<sub>E</sub>X starts in PDF mode.

Often packages or users want to know, whether the current document is typeset by pdfT<sub>E</sub>X in PDF mode, because the different modes have different capabilities (color setting, graphics inclusion, ...). For this purpose pdfT<sub>E</sub>X's `\pdfoutput` can be asked.

As regulary reader of T<sub>E</sub>X newsgroups and mailing lists I could observe many problems with this task. Common errors are:

- pdfT<sub>E</sub>X has *two* modes. Using pdfT<sub>E</sub>X does not mean that the user always want to have PDF mode. For example, the PostScript support is better in DVI mode in conjunction with a PostScript aware DVI driver (e.g. `dvips`). Also the additional typesetting features are mode independent and also available in DVI mode.
- L<sup>A</sup>T<sub>E</sub>X's `\@ifundefined` inherited the side effect from `\csname`. Unknown commands are defined with the meaning of `\relax`. If it is checked, whether `\pdfoutput` is defined, then this should not be forgotten.
- Having `\pdfoutput` does not automatically mean PDF mode. Also the value of `\pdfoutput` must be asked.
- `\pdfoutput` must not be destroyed in some way. Later code and packages are fooled then and will perhaps make wrong decisions. For example they may drop support for PDF mode, because they do not know that pdfT<sub>E</sub>X is running at all.

Robin Fairbairns provides an entry for this topic in his excellent FAQ (<http://www.tex.ac.uk/faq>): **Am I using PDFT<sub>E</sub>X?**

## 1.2 Usage

The package `ifpdf` can be used with both plain-T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X:

**plain-T<sub>E</sub>X:** `\input ifpdf.sty`

**L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>:** `\usepackage{ifpdf}`

`\ifpdf`      The package provides the switch `\ifpdf`:

```
\ifpdf
... do things, if pdfTEX is running in pdf mode ...
\else
... other TEX or pdfTEX in dvi mode ...
\fi
```

Users of the package `ifthen` can use the switch as boolean:

```
\boolean{ifpdf}
```

The package can also be used to set global documentclass options:

```

\RequirePackage{ifpdf}
\ifpdf
\documentclass[pdftex,...]{...}
\else
\documentclass[...]{...}
\fi

```

### 1.3 Specification

The package have the following properties:

- It asks the setting of `\pdfoutput` for detecting pdfTeX in PDF mode.
- It never changes `\pdfoutput`.
- It can be used with many formats including plain-T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X.

The mode detection implements the following algorithm:

```

if undefined(\pdfoutput)
  \ifpdf := \iffalse % pdfTeX is not running
else
  if \pdfoutput ≤ 0
    \ifpdf := \iffalse % pdfTeX in DVI mode
  else
    \ifpdf := \iftrue % pdfTeX in PDF mode
  fi
fi

```

The function `undefined` checks both cases, undefined command and `\relax`.

### 1.4 Future

Currently the package can be fooled, by redefining/undefining `\pdfoutput`. Therefore the package will use the `\primitive` feature that is discussed currently on the pdfTeX developer list (2006), if it hits a stable release. Of course the package will then remain usable with older pdfTeX versions as usual.

## 2 Implementation

```

1 (*package)

```

### 2.1 Reload check and package identification

Reload check, especially if the package is not used with L<sup>A</sup>T<sub>E</sub>X.

```

2 \begingroup
3 \catcode44 12 % ,
4 \catcode45 12 % -
5 \catcode46 12 % .
6 \catcode58 12 % :
7 \catcode64 11 % @
8 \expandafter\let\expandafter\x\csname ver@ifpdf.sty\endcsname
9 \ifcase 0%
10 \ifx\x\relax % plain
11 \else
12 \ifx\x\empty % LaTeX
13 \else
14 1%
15 \fi
16 \fi
17 \else

```

```

18 \catcode35 6 % #
19 \catcode123 1 % {
20 \catcode125 2 % }
21 \expandafter\ifx\csname PackageInfo\endcsname\relax
22 \def\x#1#2{%
23 \immediate\write-1{Package #1 Info: #2.}%
24 }%
25 \else
26 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
27 \fi
28 \x{ifpdf}{The package is already loaded}%
29 \endgroup
30 \expandafter\endinput
31 \fi
32 \endgroup

```

Package identification:

```

33 \begingroup
34 \catcode35 6 % #
35 \catcode40 12 % (
36 \catcode41 12 % )
37 \catcode44 12 % ,
38 \catcode45 12 % -
39 \catcode46 12 % .
40 \catcode47 12 % /
41 \catcode58 12 % :
42 \catcode64 11 % @
43 \catcode123 1 % {
44 \catcode125 2 % }
45 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
46 \def\x#1#2#3[#4]{\endgroup
47 \immediate\write-1{Package: #3 #4}%
48 \xdef#1{#4}%
49 }%
50 \else
51 \def\x#1#2[#3]{\endgroup
52 #2[#{#3}]%
53 \ifx#1\relax
54 \xdef#1{#3}%
55 \fi
56 }%
57 \fi
58 \expandafter\x\csname ver@ifpdf.sty\endcsname
59 \ProvidesPackage{ifpdf}%
60 [2007/12/12 v1.6 Provides the ifpdf switch (H0)]

```

## 2.2 Catcodes

```

61 \begingroup
62 \catcode123 1 % {
63 \catcode125 2 % }
64 \def\x{\endgroup
65 \expandafter\edef\csname ifpdf@AtEnd\endcsname{%
66 \catcode35 \the\catcode35\relax
67 \catcode64 \the\catcode64\relax
68 \catcode123 \the\catcode123\relax
69 \catcode125 \the\catcode125\relax
70 }%
71 }%
72 \x
73 \catcode35 6 % #
74 \catcode64 11 % @
75 \catcode123 1 % {

```

```

76 \catcode125 2 % }
77 \def\TMP@EnsureCode#1#2{%
78   \edef\ifpdf@AtEnd{%
79     \ifpdf@AtEnd
80     \catcode#1 \the\catcode#1\relax
81   }%
82   \catcode#1 #2\relax
83 }
84 \TMP@EnsureCode{10}{12}% ^^J
85 \TMP@EnsureCode{44}{12}% ,
86 \TMP@EnsureCode{45}{12}% -
87 \TMP@EnsureCode{46}{12}% .
88 \TMP@EnsureCode{47}{12}% /
89 \TMP@EnsureCode{58}{12}% :
90 \TMP@EnsureCode{60}{12}% <
91 \TMP@EnsureCode{61}{12}% =
92 \TMP@EnsureCode{94}{7}% ^

```

## 2.3 Check for previously defined \ifpdf

```

93 \begingroup
94   \expandafter\ifx\csname ifpdf\endcsname\relax
95   \else
96     \edef\i/{\expandafter\string\csname ifpdf\endcsname}%
97     \expandafter\ifx\csname PackageError\endcsname\relax
98       \def\x#1#2{%
99         \edef\z{#2}%
100         \expandafter\errhelp\expandafter{\z}%
101         \errmessage{Package ifpdf Error: #1}%
102       }%
103       \def\y{^^J}%
104       \newlinechar=10 %
105     \else
106       \def\x#1#2{%
107         \PackageError{ifpdf}{#1}{#2}%
108       }%
109       \def\y{\MessageBreak}%
110     \fi
111     \x{Name clash, \i/ is already defined}{%
112       Incompatible versions of \i/ can cause problems,\y
113       therefore package loading is aborted.%
114     }%
115   \endgroup
116   \ifpdf@AtEnd
117   \expandafter\endinput
118 \fi
119 \endgroup

```

## 2.4 \ifpdf

`\ifpdf` Create and set the switch. `\newif` initializes the switch with `\iffalse`.

```
120 \newif\ifpdf
```

Test `\pdfoutput`. Is it defined and different from `\relax`? Someone could have used L<sup>A</sup>T<sub>E</sub>X internal `\@ifundefined`, or something else involving. Notice, `\csname` is executed inside a group for the test to cancel the side effect of `\csname`.

```

121 \begingroup\expandafter\expandafter\expandafter\endgroup
122 \expandafter\ifx\csname pdfoutput\endcsname\relax
123 \else
124   \ifnum\pdfoutput<1 %

```

`\pdfoutput=0` or negative, so not generating pdf.

```

125   \else
126     \pdftrue

```

```

127 \fi
128 \fi

```

## 2.5 Protocol entry

Log comment:

```

129 \begingroup
130 \expandafter\ifx\csname PackageInfo\endcsname\relax
131   \def\x#1#2{%
132     \immediate\write-1{Package #1 Info: #2.}%
133   }%
134 \else
135   \let\x\PackageInfo
136 \expandafter\let\csname on@line\endcsname\empty
137 \fi
138 \x{ifpdf}{pdfTeX in pdf mode \ifpdf\else not \fi detected}%
139 \endgroup
140 \ifpdf@AtEnd
141 \</package>

```

## 3 Test

### 3.1 Catcode checks for loading

```

142 \<test1>
143 \catcode'\{=1 %
144 \catcode'\}=2 %
145 \catcode'\#=6 %
146 \catcode'\@=11 %
147 \expandafter\ifx\csname count@\endcsname\relax
148   \countdef\count@=255 %
149 \fi
150 \expandafter\ifx\csname @gobble\endcsname\relax
151   \long\def\@gobble#1{%
152 \fi
153 \expandafter\ifx\csname @firstofone\endcsname\relax
154   \long\def\@firstofone#1{#1}%
155 \fi
156 \expandafter\ifx\csname loop\endcsname\relax
157   \expandafter\@firstofone
158 \else
159   \expandafter\@gobble
160 \fi
161 {%
162   \def\loop#1\repeat{%
163     \def\body{#1}%
164     \iterate
165   }%
166   \def\iterate{%
167     \body
168     \let\next\iterate
169   \else
170     \let\next\relax
171   \fi
172   \next
173 }%
174 \let\repeat=\fi
175 }%
176 \def\RestoreCatcodes{}
177 \count@=0 %

```

```

178 \loop
179   \edef\RestoreCatcodes{%
180     \RestoreCatcodes
181     \catcode\the\count@=\the\catcode\count@\relax
182   }%
183 \ifnum\count@<255 %
184   \advance\count@ 1 %
185 \repeat
186
187 \def\RangeCatcodeInvalid#1#2{%
188   \count@=#1\relax
189   \loop
190     \catcode\count@=15 %
191     \ifnum\count@<#2\relax
192       \advance\count@ 1 %
193     \repeat
194 }
195 \expandafter\ifx\csname LoadCommand\endcsname\relax
196   \def\LoadCommand{\input ifpdf.sty\relax}%
197 \fi
198 \def\Test{%
199   \RangeCatcodeInvalid{0}{47}%
200   \RangeCatcodeInvalid{58}{64}%
201   \RangeCatcodeInvalid{91}{96}%
202   \RangeCatcodeInvalid{123}{255}%
203   \catcode'\@=12 %
204   \catcode'\=0 %
205   \catcode'\{=1 %
206   \catcode'\}=2 %
207   \catcode'\#=6 %
208   \catcode'\[=12 %
209   \catcode'\]=12 %
210   \catcode'\%=14 %
211   \catcode'\ =10 %
212   \catcode13=5 %
213   \LoadCommand
214   \RestoreCatcodes
215 }
216 \Test
217 \csname @@end\endcsname
218 \end
219 </test1>

```

## 4 Installation

### 4.1 Download

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

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

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

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

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

TDS refers to the standard “A Directory Structure for T<sub>E</sub>X Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

---

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

## 4.2 Bundle installation

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

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

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

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

## 4.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain- $\text{\TeX}$ :

```
tex ifpdf.dtx
```

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

<code>ifpdf.sty</code>	$\rightarrow$ <code>tex/generic/oberdiek/ifpdf.sty</code>
<code>ifpdf.pdf</code>	$\rightarrow$ <code>doc/latex/oberdiek/ifpdf.pdf</code>
<code>test/ifpdf-test1.tex</code>	$\rightarrow$ <code>doc/latex/oberdiek/test/ifpdf-test1.tex</code>
<code>ifpdf.dtx</code>	$\rightarrow$ <code>source/latex/oberdiek/ifpdf.dtx</code>

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

## 4.4 Refresh file name databases

If your  $\text{\TeX}$  distribution (`te $\text{\TeX}$` , `mik $\text{\TeX}$` , ...) relies on file name databases, you must refresh these. For example, `te $\text{\TeX}$`  users run `texhash` or `mktextlsr`.

## 4.5 Some details for the interested

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

```
pdftk ifpdf.pdf unpack_files output .
```

**Unpacking with  $\text{\LaTeX}$ .** The `.dtx` chooses its action depending on the format:

**plain- $\text{\TeX}$ :** Run `docstrip` and extract the files.

**$\text{\LaTeX}$ :** Generate the documentation.

If you insist on using  $\text{\LaTeX}$  for `docstrip` (really, `docstrip` does not need  $\text{\LaTeX}$ ), then inform the autodetect routine about your intention:

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

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



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

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

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

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

## 5 History

### [2001/06/14 v1.0]

- First public version.

### [2001/07/14 v1.1]

- Documentation addition: global options

### [2001/09/26 v1.2]

- Documentation typo corrected.
- Version number corrected.
- Line number in log entry removed.

### [2005/07/22 v1.3]

- Some source code comments from Robin Fairbairns added.
- Bug fix for negative values of `\pdfoutput` (Oleg Katsitadze)
- LPPL 1.3
- Installation section with locations added.

### [2006/02/20 v1.4]

- DTX framework.
- More robust check in case of undefined `\pdfoutput`.
- Extended documentation.

### [2007/09/09 v1.5]

- Catcode settings added.

### [2007/12/12 v1.6]

- Minor update.

## 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; numbers in roman refer to the code lines where the entry is used.

Symbols		
\#	145, 207	\ifpdf@AtEnd . . . . . 78, 79, 116, 140
\%	210	\ifx . . . . . 10, 12, 21, 45, 53, 94, 97, 122, 130, 147, 150, 153, 156, 195
\@	146, 203	\immediate . . . . . 23, 47, 132
\@firstofone	154, 157	\input . . . . . 196
\@gobble	151, 159	\iterate . . . . . 164, 166, 168
\[	208	
\]	204	<b>L</b>
\{	143, 205	\LoadCommand . . . . . 196, 213
\}	144, 206	\loop . . . . . 162, 178, 189
\]	209	
		<b>M</b>
\_	211	\MessageBreak . . . . . 109
		<b>N</b>
<b>A</b>		\newif . . . . . 120
\advance	184, 192	\newlinechar . . . . . 104
		\next . . . . . 168, 170, 172
<b>B</b>		
\body	163, 167	<b>P</b>
		\PackageError . . . . . 107
<b>C</b>		\PackageInfo . . . . . 26, 135
\catcode	3, 4, 5, 6, 7, 18, 19, 20, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 62, 63, 66, 67, 68, 69, 73, 74, 75, 76, 80, 82, 143, 144, 145, 146, 181, 190, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212	\pdfoutput . . . . . 124
\count@	148, 177, 181, 183, 184, 188, 190, 191, 192	\pdftrue . . . . . 126
\countdef	148	\ProvidesPackage . . . . . 59
\csname	8, 21, 45, 58, 65, 94, 96, 97, 122, 130, 136, 147, 150, 153, 156, 195, 217	
		<b>R</b>
<b>E</b>		\RangeCatcodeInvalid . . . . . 187, 199, 200, 201, 202
\empty	12, 136	\repeat . . . . . 162, 174, 185, 193
\end	218	\RestoreCatcodes . . 176, 179, 180, 214
\endcsname	8, 21, 45, 58, 65, 94, 96, 97, 122, 130, 136, 147, 150, 153, 156, 195, 217	
\endinput	30, 117	<b>T</b>
\errhelp	100	\Test . . . . . 198, 216
\errmessage	101	\the . . . . . 66, 67, 68, 69, 80, 181
		\TMP@EnsureCode . . . . . 77, 84, 85, 86, 87, 88, 89, 90, 91, 92
		<b>W</b>
<b>I</b>		\write . . . . . 23, 47, 132
\i	96, 111, 112	
\ifcase	9	<b>X</b>
\ifnum	124, 183, 191	\x . . 8, 10, 12, 22, 26, 28, 46, 51, 58, 64, 72, 98, 106, 111, 131, 135, 138
\ifpdf	2, <u>120</u> , 138	
		<b>Y</b>
		\y . . . . . 103, 109, 112
		<b>Z</b>
		\z . . . . . 99, 100