

# The pst-pdf package\*

Rolf Niepraschk<sup>†</sup>      Hubert Gäßlein

2006/07/14

## 1 Introduction

The package `pst-pdf` simplifies the use of graphics from `PSTricks` and other PostScript code in PDF documents. As in building a bibliography with `BIBTEX` additional external programmes are being invoked. In this case they are used to create a PDF file (`\PDFcontainer`) that will contain all this graphics material. In the final document this contents will be inserted instead of the original PostScript code.

## 2 Usage

### 2.1 Package options

**active** Activates the extraction mode (DVI output). An explicit declaration usually is not necessary (default in `LATEX` mode).

**inactive** No special actions; only the packages `pstricks` and `graphicx` are loaded (default in `VTEX`). Can be used to just convert the document with `LATEX` into a DVI file while avoiding the automatic extraction mode.

**pstricks** The package `pstricks` is loaded (default).

**nopstricks** The package `pstricks` does not get loaded. Once it is detected that `pstricks` was loaded however in some other way, the `pspicture` environment is treated as if the option “`pstricks`” was given.

**draft** From the `\PDFcontainer` file included graphics is displayed as frame in `pdfLATEX` mode.

**final** From the `\PDFcontainer` file included graphics is correctly displayed in `pdfLATEX` mode (default).

**tightpage** The graphics’ dimensions in the `\PDFcontainer` file match exactly those of the corresponding `TEX` boxes (default).

**notightpage** The dimensions of the `TEX` box corresponding to its graphics is not always correct, since a PostScript statement can draw outside its box. The option “`notightpage`” makes the graphics in the `\PDFcontainer` file to be at

---

\*This document corresponds to `pst-pdf v1.1n`, dated 2006/07/14. Thanks to Peter Dyballa for the translation.

<sup>†</sup>`Rolf.Niepraschk@ptb.de`

least the size of the whole page. To be able to make use of the graphics' in a later pdfL<sup>A</sup>T<sub>E</sub>X run, the `\PDFcontainer` file needs to be finished in a way that each graphics gets reduced in size to its visible part. For this an external programme like `pdfcrop`<sup>1</sup> can be useful. Its use can save declaring the option “trim” (see also section 2.4).

**displaymath** In PDF mode the mathematical environments `displaymath`, `eqnarray`, and `$$` get also extracted and included as graphics. This way additional PSTricks extensions can easily be added to the contents of these environments. (Question: how do AMSL<sup>A</sup>T<sub>E</sub>X environments behave?)

**<other>** All other options are passed to `psctricks` package.

## 2.2 Program calls

The following table shows the course necessary to create a PDF document containing PostScript graphics<sup>2</sup>. As comparison the analogous course for a bibliography is shown.

PostScript graphics	bibliography
<code>pdflatex document.tex</code>	<code>pdflatex document.tex</code>
<i>auxiliary calls</i>	
<code>latex document.tex</code>	
<code>dvips -o document-pics.ps document.dvi</code>	
<code>ps2pdf document-pics.ps</code>	<code>bibtex document.aux</code>
<code>pdflatex document.tex</code>	<code>pdflatex document.tex</code>

While creating the output only code from inside a `pspicture` or `postscript` environment is considered. PostScript graphics files, which are passed as parameter of an `\includegraphics` statement, too are included into the `\PDFcontainer` file. This file's name is by default `\jobname-pics.pdf`. It can be changed by re-defining the macro `\PDFcontainer`.

## 2.3 User commands

`pspicture`      `\begin{pspicture}[\langle keys \rangle] (\langle x0,x1 \rangle) (\langle y0,y1 \rangle) ... \end{pspicture}`  
 The `pspicture` environment is not available when the option “`nopstricks`” was given. It is to be used the same way as if in PSTricks. In pdfL<sup>A</sup>T<sub>E</sub>X mode this environment's contents is only displayed when the `\PDFcontainer` file was created before.

`postscript`    `\begin{postscript}[\langle keys \rangle] ... \end{postscript}`  
 The `postscript` environment can contain any code except floats. In pdfL<sup>A</sup>T<sub>E</sub>X mode its contents is take too off the `\PDFcontainer` file. Other as in the `pspicture` environment the necessary space is not always preserved when the `\PDFcontainer` file does not exist yet.

`\includegraphics`      `\includegraphics[\langle keys \rangle]{\langle filename \rangle}`

<sup>1</sup>CTAN: support/pdfcrop/

<sup>2</sup>The T<sub>E</sub>X distribution “teT<sub>E</sub>X” contains a UNIX shell script `ps4pdf` which executes all the necessary steps. See: CTAN: macros/latex/contrib/ps4pdf/

To be used as in `graphics/graphicx` defined. In pdfL<sup>A</sup>T<sub>E</sub>X mode it is now additionally feasible to pass the name of an EPS file. Its visible contents too is taken from the `\PDFcontainer` file.

<code>\includegraphics</code>	<code>\includegraphics[<i>keys</i>](<i>pfxadd</i>)&lt;<i>ovpfgd</i>&gt;[<i>ovpbgd</i>]{<i>filename</i>}</code> Wie im Paket <code>psfragx</code> definiert zu verwenden.
<code>\savepicture</code>	<code>\savepicture{<i>name</i>}</code> The last output graphics (result of the <code>pspicture</code> or <code>postscript</code> environments or the <code>\includegraphics</code> statement with an PostScript file as argument) is being saved in a file under the name as given by the parameter.
<code>\usepicture</code>	<code>\usepicture[<i>keys</i>]{<i>name</i>}</code> Die zuvor mit <code>\savepicture</code> gespeicherte Grafik wird ausgegeben. Der optionale Parameter entspricht dem bei der Anweisung <code>\includegraphics</code> möglichen.
<code>pst-pdf-defs</code>	<code>\begin{pst-pdf-defs} ... \end{pst-pdf-defs}</code> Sollen eigene Makros oder Umgebungen definiert werden, die das Zeichen <code>&amp;</code> (andere?) im Ersetzungstext enthalten, so müssen diese Definitionen von der Umgebung <code>pst-pdf-defs</code> umschlossen werden.

## 2.4 Command options

The behaviour of the `\includegraphics` and `\usepicture` statements and the `postscript` environment can be modified with any of the following parameters (key value syntax):

- frame**=`<true|false>` As with the `\fbox` statement a frame is drawn around the graphics. Any change of size due to rotation is taken into account. Drawing happens in pdfL<sup>A</sup>T<sub>E</sub>X mode; before, in creating the `\PDFcontainer` file, it is ignored. Default: `false`.
- innerframe**=`<true|false>` As in “`frame`”, but the frame is drawn around the graphics, not its box.
- ignore**=`<true|false>` If “`true`” no graphics is output. With `\savepicture{name}` the graphics can be used later in a different place via `\usepicture`. Default: `false`.
- showname**=`<true|false>` A caption of minimal font size records the used file’s name. Default: `false`.
- namefont**=`<font commands>` Controls the font used when “`showname=true`” is set. Default: `\ttfamily\tiny`

All parameters can be set globally as in `\setkeys{Gin}{key=value}`.

## 3 Implementation

1 `{*package}`

### 3.1 Package options

2 `\newcommand*\ppf@TeX@mode{-1}`

```

3 \newcommand*\ppf@draft{false}
4 \newif\if@ppf@PST@used\@ppf@PST@usedtrue
5 \newif\if@ppf@tightpage \@ppf@tightpagetrue
6 \DeclareOption{active}{\def\ppf@TeX@mode{0}}
7 \DeclareOption{inactive}{\def\ppf@TeX@mode{9}}
8 \DeclareOption{ignore}{\def\ppf@TeX@mode{999}}
9 \DeclareOption{pstricks}{\@ppf@PST@usedtrue}
10 \DeclareOption{nopstricks}{\@ppf@PST@usedfalse}
11 \DeclareOption{displaymath}{%
12   \PassOptionsToPackage\CurrentOption{preview}}
13 \DeclareOption{draft}{\def\ppf@draft{true}}
14 \DeclareOption{final}{\def\ppf@draft{false}}%
15   \PassOptionsToPackage\CurrentOption{graphicx}}

16 \DeclareOption{notightpage}{\@ppf@tightpagefalse}%
17 \DeclareOption{tightpage}{\@ppf@tightpagetrue}%
18 \DeclareOption*{%
19   \PassOptionsToPackage\CurrentOption{pstricks}}
20 \ProcessOptions\relax
21 \ifnum\ppf@TeX@mode=999\relax\expandafter\endinput\fi

```

### 3.2 Compiler tests

It is tested which  $\TeX$  compiler in which mode of operation is actually used (see ‘graphics.cfg’ in  $\text{te}\TeX/\TeX$  Live). Accordingly the environments `pspicture` and `postscript` gain each a different range of functions. This test is only executed when the options `active` or `inactive` were not given.

```

22 \ifnum\ppf@TeX@mode=-1\relax
23   \begingroup
Default ( $\TeX$  with a dvi-to-ps converter)
24   \chardef\x=0 %
Check pdf $\TeX$ 
25   \@ifundefined{pdfoutput}{}{%
26     \ifcase\pdfoutput\else
27       \chardef\x=1 %
28     \fi
29   }%
Check V $\TeX$ 
30   \@ifundefined{OpMode}{}{\chardef\x=2 }%
31   \expandafter\endgroup
32   \ifcase\x
⇒ DVI mode
33   \def\ppf@TeX@mode{0}%
34   \or
⇒ pdf $\TeX$  is running in PDF mode
35   \def\ppf@TeX@mode{1}%
36   \else
⇒ V $\TeX$  is running
37   \def\ppf@TeX@mode{9}%
38   \fi
39 \fi

```

```

40 \newcommand*\PDFcontainer{}
41 \edef\PDFcontainer{\jobname-pics.pdf}
42 \newcounter{pspicture}
43 \newcommand*\ppf@other@extensions[1]{}
44 \newcommand*\usepicture[2] []{}
45 \newcommand*\savepicture[1]{}

```

pst-pdf-defs

```

46 \newenvironment*{pst-pdf-defs}%
47 {%
48 \endgroup
49 % ??? \@currentvline
50 }{%
51 \begingroup
52 \def\@currentvir{pst-pdf-defs}%
53 }

54 \RequirePackage{graphicx}%
55 \let\ppf@Gininclude@graphics\Gininclude@graphics
56 \let\ppf@Gin@extensions\Gin@extensions
57 \let\ppf@Gin@ii\Gin@ii

58 \newif\if@ppf@pdftex@graphic
59 \newif\if@Gin@frame\Gin@framefalse
60 \newif\if@Gin@innerframe\Gin@innerframefalse
61 \newif\if@Gin@showname\Gin@shownamefalse
62 \newif\if@Gin@ignore\Gin@ignorefalse

```

\ifpr@outer in fact is defined in package preview. We have to do it here too since otherwise T<sub>E</sub>X could “stumble and fall” while parsing the \ifcase structure.

```
63 \newif\ifpr@outer
```

\ppf@is@pdfTeX@graphic

Parameter #1 is the name of a graphics file with or without extension, parameter #2 contains the valid extensions in PDF mode, parameter #3 contains the valid extensions in DVI mode. If it works to process the graphics in PDF mode, then the statements in #4 are executed, otherwise those in #5.

```

64 \newcommand*\ppf@is@pdfTeX@graphic[5]{}
65 \@ppf@pdftex@graphicfalse%
66 \begingroup
67 \edef\pdfTeXext{#2}%

```

Instead of loading the found graphics, only a test on file name extension.

```

68 \def\Gin@setfile##1##2##3{%
69 \edef\@tempb{##2}%
70 \@for\@tempa:=\pdfTeXext\do{%
71 \ifx\@tempa\@tempb\global\@ppf@pdftex@graphictrue\fi}}%

```

File types for both modes need to be determined to prevent a wrong error message “File ‘#1’ not found”.

```
72 \edef\Gin@extensions{#2,#3}%
```

Trial invocation. Output is completely inhibited.

```

73 \pr@outerfalse\ppf@Gininclude@graphics{#1}%
74 \endgroup
75 \if@ppf@pdftex@graphic#4\else#5\fi
76 }

```

```
77 \ifcase\ppf@TeX@mode\relax
```

### 3.3 Extraction mode (DVI output)

The `pspicture` environment retains any definition from `pstricks.tex`. Only the code from the environments `pspicture` and `postscript` as well as `\includegraphics` with PostScript files leads to records into the DVI file. The remainder of the document's code is ignored for output. After conversion of the DVI file via PostScript (“`dvips`”) into PDF (`\PDFcontainer` file) each graphics takes exactly one page in the `\PDFcontainer` file. The  $\TeX$  compiler with DVI output and the package option “`active`” both force this mode.

```
78 \PackageInfo{pst-pdf}{%
79   MODE: \ppf@TeX@mode\space (dvi -- extraction mode)}
80 \nofiles
81 \ifppf@PST@used\RequirePackage{pstricks}\fi
82 \RequirePackage[active,dvips,tightpage]{preview}[2005/01/29]%
83 \newcommand*\ppf@PreviewBbAdjust{}
84 \newcommand*\ppf@RestoreBbAdjust{}
85 \let\PreviewBbAdjust\ppf@PreviewBbAdjust}%
```

The pdf $\LaTeX$  mode compliant graphics file formats are needed too.

```
86 \begingroup
87 \let\AtBeginDocument\@gobble \let\PackageWarningNoLine\@gobbletwo
88 \def\pdftexversion{121}\input{pdftex.def}%
89 \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}
90 }%
91 \x
```

In PDF mode no rules must be defined for its compliant (PNG, JPEG, PDF) graphics file formats (because of for example ‘`dvips`’ extensions). The universal EPS rule is used to at least find these files.

```
92 \AtBeginDocument{%
93   \@for\@tempa:=\ppf@other@extensions\do{%
94     \expandafter\let\csname Gin@rule@\@tempa\endcsname\relax}%
95   \DeclareGraphicsRule{*}{eps}{*}{*}}%
```

No function in this mode.

```
96 \define@key{Gin}{innerframe}[true]{}%
97 \define@key{Gin}{frame}[true]{}%
98 \define@key{Gin}{ignore}[true]{}%
99 \define@key{Gin}{showname}[true]{}%
100 \define@key{Gin}{namefont}{}%

101 \ifppf@tightpage\else
102   \def\PreviewBbAdjust{%
103     -.5\paperwidth -.5\paperheight .5\paperwidth .5\paperheight}%
104   \AtEndDocument{%
105     \PackageWarningNoLine{pst-pdf}{Picture container needs cropping.}}%
106 \fi
```

`postscript` The `postscript` environment utilises the `trim` option in the same manner as does `\includegraphics` (any specification without dimension is interpreted as if given in `bp`).

```
107 \newenvironment{postscript}[1] []%
```

```

108  {%
109    \global\let\ppf@PreviewBbAdjust\PreviewBbAdjust
110    \if@ppf@tightpage
111      \begingroup
112        \setkeys{Gin}{#1}%
113        \xdef\PreviewBbAdjust{%
114          -\Gin@vllx bp -\Gin@vllly bp \Gin@vurx bp \Gin@vury bp}%
115        \endgroup
116      \fi
117      \ignorespaces
118    }%
119  {\aftergroup\ppf@RestoreBbAdjust}%

120  \PreviewEnvironment{postscript}%
121  \AtBeginDocument{%
122    \@ifundefined{PSTricksLoaded}{}%
123    {%

```

pspicture Announce preview original definition.

```

124    \PreviewEnvironment{pspicture}%

```

psmatrix Announce preview original definition.

```

125    \@ifundefined{psmatrix}{}%
126    {%
127      \PreviewEnvironment{psmatrix}%
128      \newcommand*\ppf@set@mode{}%
129      \newcommand*\ppf@test@mmode{%
130        \ifmmode
131          \ifinner
132            \let\ppf@set@mode=$%
133          \else
134            \def\ppf@set@mode{$$}%
135          \fi
136        \else
137          \let\ppf@set@mode=\@empty
138        \fi
139      }%
140      \let\ppf@psmatrix=\psmatrix
141      \expandafter\let\expandafter\ppf@pr@psmatrix%
142      \expandafter=\csname pr@\string\psmatrix\endcsname
143      \let\ppf@endpsmatrix=\endpsmatrix
144      \def\psmatrix{\ppf@test@mmode\ppf@psmatrix}
145      \expandafter\def\csname pr@\string\psmatrix\endcsname{%
146        \ppf@set@mode\ppf@pr@psmatrix}%
147      \def\endpsmatrix{\ppf@endpsmatrix\ppf@set@mode}%
148    }%

```

Announce internal macro `\pst@object` to enable the use of some PSTricks code outside of `pspicture` environments. At the moment invocations of the following kind are feasible:

```

\pst@object {<m>}{*}[<o>]{<o>}{<o>}(<o>)(<o>)(<o>)
(m = necessary, * = optional, o = optional)

```

More than three optional arguments at the call's end, as in `\psline` possible, do not work yet.

```

149     \PreviewMacro[{}*[]%
150         ?\bgroup{##1}{##1}}{}%
151         ?\bgroup{##1}{##1}}{}%
152         ?({#{(##1)}{({##1})}}){}%
153         ?({#{(##1)}{({##1})}}){}%
154         ?({#{(##1)}{({##1})}}){}%
155         ]{\pst@object}}

```

Prevent multiple test-wise setting of table contents by “`tabularx`”.

```

156     \@ifundefined{tabularx}{}{}%
157     \def\tabularx#1#2{\tabular{#2}}%
158     \newcolumntype{X}{c}%
159     \let\endtabularx=\endtabular}%

```

Support of `\includegraphicx` from the package `psfragx`.

```

160     \@ifundefined{pfx@includegraphicx}{}{}%
161     \PreviewMacro[{}{}]{\pfx@includegraphicx}}%
162     }%

```

`\Gininclude@graphics` All graphics content of well known format (for instance EPS files) is treated in a regular way, which in this mode denotes that it is subject to preview functions. Other graphics content (for instance PDF files) is ignored.

```

163     \def\Gininclude@graphics#1{%
164         \ifpr@outer

```

Generally pdf<sub>T</sub>E<sub>X</sub> supported graphics formats are intended to be preferred (inclusion in final pdf<sub>T</sub>E<sub>X</sub> run). If it's a PostScript type graphics, then the original definition is in function again and registration for the `preview` package is necessary in order to convert this PostScript type graphics into PDF.

```

165         \ppf@is@pdfTeX@graphic{#1}{\ppf@other@extensions}{\Gin@extensions}}%

```

Dummy box to prevent a division by zero while scaling or rotating. Otherwise ignored.

```

166         {\rule{10pt}{10pt}}%
167         {\ppf@Gininclude@graphics{#1}}%
168         \else

```

Inside a PostScript environment (`pspicture` etc.) `\includegraphics` has to behave as in its original definition (only DVIPS supported graphics formats are allowed).

```

169         \ppf@Gininclude@graphics{#1}}%
170     \fi
171 }%

```

```

172     \PreviewMacro[{}{}]{\ppf@Gininclude@graphics}}%
173     \let\pdfliteral\gobble%
174 \or

```

### 3.4 pdf<sub>L</sub>A<sub>T</sub>E<sub>X</sub> mode (PDF output)

When the `\PDFcontainer` file (default: `\jobname-pics.pdf`) exists, the contents of the environments `pspicture` and `postscript` is ignored. Instead the corresponding graphics from the `\PDFcontainer` file is used.

```

175 \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (pdfTeX mode)}%
176 \@temptokena{%
177   \let\Gin@PS@file@header@gobble\let\Gin@PS@literal@header@gobble
178   \let\Gin@PS@raw@gobble\let\Gin@PS@restored@gobble
179   \@ifundefined{PSTricksLoaded}{}{}}%

```

Necessary if PSTricks < 2.0.

```

180   \PSTricksOff
181   \@ifundefined{color@to@ps}{\def\color@to@ps#1 #2\@{}}{}}%

```

Prevent pdfTeX's message Non-PDF special ignored!.

```

182 \ifppf@PST@used
183   \let\ppf@temp\AtBeginDvi\let\AtBeginDvi@gobble
184   \RequirePackage{pstricks}\let\AtBeginDvi\ppf@temp
185 \fi

```

PostScript output is now inhibited and later once again.

```

186 \the\@temptokena %%% ????
187 \expandafter\AtBeginDocument\expandafter
188   {\the\@temptokena\@temptokena{}}%
189 \@ifundefined{PSTricksLoaded}{}{
190   {%

```

To parse the arguments of PSTricks' `\pst@object` we load `preview` in active mode, but restore the default definitions of `\output` and `\shipout`. `\pr@startbox` and `\pr@endbox` serve here only to disable `\pst@object` and to load the corresponding graphics from the `\PDFcontainer` file. At present a maximum of three optional parameters in round braces (parenthesis) at the end of `\pst@object` is supported, which is sufficient, but not always enough.

```

191   \newtoks\ppf@temptoken
192   \ppf@temptoken\expandafter{\the\output}%
193   \let\ppf@nofiles\nofiles \let\nofiles\relax
194   \RequirePackage[active]{preview}[2005/01/29]%
195   \let\shipout=\pr@shipout \let\nofiles\ppf@nofiles
196   \output\expandafter{\the\ppf@temptoken}%
197   \ppf@temptoken{}%

```

`\pr@startbox`, `\pr@endbox`: simpler over original definitions.

```

198   \long\def\pr@startbox#1#2{%
199   \ifpr@outer
200     \toks@{#2}%
201     \edef\pr@cleanup{\the\toks@}%
202     \setbox\@tempboxa\vbox\bgroup
203     \everydisplay{}%
204     \pr@outerfalse%
205     \expandafter\@firstofone
206   \else
207     \expandafter@gobble
208   \fi{#1}}%
209   \def\pr@endbox{%
210   \egroup
211   \setbox\@tempboxa\box\voidb@x
212   \ppf@getpicture
213   \pr@cleanup}%

```

(See also the identical definition in DVI mode.)

```

214 \AtBeginDocument{%
215   \ifundefined{pst@object}{-%
216     {%
217       \PreviewMacro[{}*[]%
218       ?\bgroup{##1}{##1}}{-}%
219       ?\bgroup{##1}{##1}}{-}%
220       ?({##1}){({##1})}}{-}%
221       ?({##1}){({##1})}}{-}%
222       ?({##1}){({##1})}}{-}%
223     }{\pst@object}}%
224   }%
225 }%

```

Too the supported file name extensions from DVI mode are needed.

```

226 \begingroup
227   \input{dvips.def}%
228   \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}}%
229   \x

```

Dummy definition for in DVI mode supported file formats.

```

230 \DeclareGraphicsRule*{eps}{*}{*}{%
231   \define@key{Gin}{innerframe}[true]{%
232     \lowercase{\Gin@boolkey{#1}}{innerframe}}%
233   \define@key{Gin}{frame}[true]{%
234     \lowercase{\Gin@boolkey{#1}}{frame}}%
235   \define@key{Gin}{ignore}[true]{%
236     \lowercase{\Gin@boolkey{#1}}{ignore}}%
237   \define@key{Gin}{frame@@}{%

```

(For internal use only!)

```

238   \edef\@tempa{\toks@{\noexpand\frame{the\toks@}}}%
239   \ifcase#1\relax
240     \ifGin@innerframe\else\let\@tempa\relax\fi
241   \or
242     \ifGin@frame\else\let\@tempa\relax\fi
243   \fi
244   \@tempa
245 }%
246 \define@key{Gin}{showname}[true]{%
247   \lowercase{\Gin@boolkey{#1}}{showname}}%
248 \define@key{Gin}{namefont}{%
249   \begingroup
250     \@temptokena\expandafter{\ppf@namefont#1}%
251     \edef\x{\endgroup\def\noexpand\ppf@namefont{the\@temptokena}}%
252     \x
253   }%
254   \newcommand*\ppf@filename{%
255     \newcommand*\ppf@namefont{\tiny\ttfamily}%
256     \newcommand*\ppf@Gin@keys{}%
257     \let\ppf@Gin@setfile\Gin@setfile

```

\Gin@setfile Save real file name and, if applicable, page number for later use.

```

258   \def\Gin@setfile#1#2#3{\ppf@Gin@setfile{#1}{#2}{#3}%
259     \xdef\ppf@filename{%
260       #3\ifx\GPT@page\empty\else(\GPT@page)\fi}}%

```

`\Gin@ii` Examine the options “frame”, “ignore”, etc. as soon as other special cases.

```

261 \def\Gin@ii[#1]#2{%
262 \begingroup

The value of \ifGin@innerframe has to be known before the inner frame is drawn.
The values for \ifGin@showname and \ppf@namefont need to be available after
rendering the graphics too. Thus beforehand and protected inside a group examine
the options.

263 \setkeys{Gin}{#1}%
264 \@temptokena{#1}\def\@tempb{#2}%

Finds empty file name when calling \usepicture.

265 \ifx\@tempb\@empty\else
266 \ppf@is@pdfTeX@graphic{#2}{\Gin@extensions}{\ppf@other@extensions}%

Graphics out of \PDFcontainer are complete – scaled, rotated, etc. Don't apply
these things again and therefore ignore the optional parameters.

267 {%
268 \ifx\@tempb\PDFcontainer
269 \@temptokena{page=\GPT@page}%
270 \fi
271 }%
272 {%
273 \refstepcounter{pspicture}%
274 \@temptokena{page=\the\c@pspicture}\def\@tempb{\PDFcontainer}%
275 }%
276 \fi
277 \ifGin@ignore\else

“frame@@=0” = inner frame, “frame@@=1” = outer frame.

278 \edef\@tempa{\noexpand\ppf@Gin@ii[frame@@=0,\the\@temptokena,
279 frame@@=1]{\@tempb}}%
280 \@tempa
281 \ifGin@showname
282 \ppf@namefont
283 \raisebox{-\ht\strutbox}[0pt][0pt]{\llap{\ppf@filename}}%
284 \gdef\ppf@filename{}}%
285 \fi
286 \fi
287 \endgroup
288 }%

289 \IfFileExists{\PDFcontainer}%
290 {%

```

`\ppf@container@max` The number of pages as contained in `\PDFcontainer` file.

```

291 \pdfximage{\PDFcontainer}%
292 \edef\ppf@container@max{\the\pdflastximagepages}%

293 \AtEndDocument{%
294 \ifnum\c@pspicture>\z@

A warning only makes sense when a graphics is needed at all.

295 \ifnum\c@pspicture=\ppf@container@max\else
296 \PackageWarningNoLine{pst-pdf}{%
297 ‘\PDFcontainer’ contains \ppf@container@max\space pages

```

```

298         \MessageBreak but \the\c@pspicture\space pages are requested:
299         \MessageBreak File '\PDFcontainer' is no more valid!
300         \MessageBreak Recreate it
301     }%
302     \fi
303     \fi
304 }%
305 }%
306 {%
307     \def\ppf@container@max{0}%
308     \AtEndDocument{%
309         \ifnum\c@pspicture>\z@
310             \filename@parse{\PDFcontainer}%
311             \PackageWarningNoLine{pst-pdf}{%
312                 File '\PDFcontainer' not found.\MessageBreak
313                 Use the following commands to create it:\MessageBreak
314                 -----
315                 \MessageBreak
316                 latex \jobname.tex\MessageBreak
317                 dvips -o \filename@base.ps \jobname.dvi\MessageBreak
318                 ps2pdf \filename@base.ps\MessageBreak
319                 -----
320             }%
321         \fi
322     }%
323 }%

```

`\ppf@isnum` If parameter #1 is numeric, the instructions in #2, otherwise those in #3 are executed (see `bibtopic.sty`).

```

324 \newcommand\ppf@isnum[1]{%
325     \if!\ifnum9<1#1!\else_\fi\expandafter\@firstoftwo
326     \else\expandafter\@secondoftwo\fi}%

```

`postscript` Both environments ignore their contents and load instead the corresponding graphics out of the `\PDFcontainer` file. The value of the herein used `pspicture` counter's value can be used in `\label/\ref`.

`psmatrix`

```

327 \newcommand*\ppf@set@mode{%
328 \newcommand*\ppf@test@mmode{%
329 \ifmmode
330     \ifinner
331         \let\ppf@set@mode=$%
332     \else
333         \def\ppf@set@mode{$$}%
334     \fi
335 \else
336     \let\ppf@set@mode=\@empty
337 \fi
338 }
339 \newenvironment{postscript}[1] []
340 {%
341     \ppf@test@mmode
342     \gdef\ppf@Gin@keys{%

```

```

343   \def\@tempa{postscript}\ifx\@tempa\@currentenv\gdef\ppf@Gin@keys{#1}\fi
    Inside this environment parsing of \pst@object's arguments is not necessary, thus
    the original definition is used again.
344   \expandafter\let\expandafter\pst@object
345     \csname pr@\string\pst@object\endcsname
346   \pr@outerfalse
    Needed for \psmatrix.
347   \@makeother\&%
348   \def\Gin@ii[##1]##2{\setbox\@tempboxa=\vbox\bgroup
349     \ppf@set@mode
350   }%
351   {\ppf@set@mode\egroup\aftergroup\ppf@getpicture}%
352   \AtBeginDocument{%
353     \@ifundefined{PSTricksLoaded}{}%
354     {%
355       \iffalse
356         \PreviewEnvironment{pspicture}% Why doesn't it work?
357         \g@addto@macro\pspicture{%
358           %\pr@outerfalse% necessary, or already there anyway?
359           \@makeother\&% necessary?
360           \def\Gin@ii[#1]#2{%
361             }%
362           \g@addto@macro\endpspicture{\ppf@getpicture}%
363         \else
364           \def\pst@@@picture[#1](#2,#3)(#4,#5){\postscript}%
365           \def\endpspicture{\endpostscript\endgroup}%
366         \fi
367         \@ifundefined{psmatrix}{}%
368         {\let\psmatrix=\postscript\let\endpsmatrix=\endpostscript}%
369       }%
370     \@ifundefined{pfx@includegraphics}{}{%
    Die im pdfTeX-Modus unnütze Umdefinition von \includegraphics (Paket
    psfrag) führt zu zweifachem Einfügen des Ergebnisses, weshalb die Originalde-
    finition wiederhergestellt wird.
371     \let\includegraphics=pfx@includegraphics
372     \def\pfx@includegraphics#1#2{\ppf@getpicture}%
373   }%
374 }%

```

**\savepicture** Saves the recent graphics' number in a macro named \ppf@@@#1.

```

375   \def\savepicture#1{%
376     \expandafter\edef\csname pfx@@@#1\endcsname{\the\pdfastximage}}%

```

**\usepicture** Inserts graphics with symbolic name #2. This name has to be declared beforehand in \savepicture{<name>}. Instead of a name a number can be used too, which directly addresses a graphics in the \PDFcontainer file. The optional parameter #1 corresponds to the one in \includegraphics.

```

377   \renewcommand*\usepicture[2] [] {%
378     \@ifundefined{ppf@@@#2}%
379     {%
380       \ppf@isnum{#2}%
381       {\ppf@getpicture{#1}{#2}}%

```

```

382     {\@latex@error{picture ‘#2’ undefined}\@ehc}%
383 }%
384 {%
385   \begingroup
386     \def\Gininclude@graphics##1{%
387       \xdef\ppf@filename{#2}%
388       \setbox\z@\hbox{\pdfrefximage\@nameuse{ppf@@@#2}}%
389       \Gin@nat@height\ht\z@ \Gin@nat@width\wd\z@
390       \def\Gin@llx{0} \let\Gin@lly\Gin@llx
391       \Gin@defaultbp\Gin@urx{\Gin@nat@width}%
392       \Gin@defaultbp\Gin@ury{\Gin@nat@height}%
393       \Gin@bboxtrue\Gin@viewport@code
394       \Gin@nat@height\Gin@ury bp%
395       \advance\Gin@nat@height-\Gin@lly bp%
396       \Gin@nat@width\Gin@urx bp%
397       \advance\Gin@nat@width-\Gin@llx bp%
398       \Gin@req@sizes
399       \ht\z@\Gin@req@height \wd\z@\Gin@req@width
400       \leavevmode\box\z@}%
401     \define@key{Gin}{type}{}%
402     \includegraphics[scale=1,#1]{}%
403   \endgroup
404 }}%

```

`\ppf@getpicture` Inserts the page (graphics) with number #2 from the `\PDFcontainer` file. Parameter #1: any option as in `\includegraphics`.

```

405 \newcommand*\ppf@getpicture[2]{%
406   \@tempcnta=#2\relax%
407   \ifnum\@tempcnta>\ppf@container@max
408     \PackageWarningNoLine{pst-pdf}{%
409       pspicture No. \the\@tempcnta\space undefined}%
410   \else
411     \includegraphics[draft=\ppf@draft,#1,page=\the\@tempcnta]%
412       {\PDFcontainer}%
413   \fi
414   \gdef\ppf@Gin@keys{}}%

```

`\ppf@@getpicture` Inserts next page (graphics) from the `\PDFcontainer` file.

```

415 \newcommand*\ppf@@getpicture{%
416   \ifpr@outer
417     \refstepcounter{pspicture}%
418     \expandafter\ppf@getpicture\expandafter{\ppf@Gin@keys}%
419     {\the\c@pspicture}%
420   \fi}%

```

`pst-pdf-defs` Umgebung, die keine eigene Gruppe aufmacht. Innerhalb der Umgebung bekommt das Zeichen & den Kategoriecode "other". Gedacht für eigene Makros, die z. B. eine `psmatrix` enthalten. (Einen "Hook" verwenden, falls andere Zeichen auch noch benötigt werden!?)

```

421 \renewenvironment*{pst-pdf-defs}%
422 {%
423   \endgroup
424 %   ??? \@currentline

```



v1.0l	General: Options “framesep”, “framerule”, “linewidth” removed, “fname” and “innerframe” added. (RN) . . . . .	1	Supress handling of pdf $\LaTeX$ graphic formats in DVI mode. (RN) . . . . .	6	
v1.0m	General: New package option “notightpage” added. (RN) . . .	1	v1.1d	<b>psmatrix</b> : Support for PSTricks environment “psmatrix”. (RN) .	12
v1.0n	General: Changed marco names ( <code>\savepicture</code> and <code>\usepicture</code> ). (RN) . . . . .	1	v1.1e	General: New option “displaymath” (see preview package). (HjG/RN) . . . . .	3
v1.0o	General: New code for “notightpage”. (RN) . . . . .	6	v1.1f	General: Package option “ignore” reimplemented. Now the compilation of the dtx file in $\LaTeX$ mode is possible. (RN) . . . . .	3
v1.0p	General: Some code and documentation cleaning. (RN) . . . . .	1	v1.1g	<b>psmatrix</b> : “psmatrix” environment (preserve math mode). (RN/HjG) . . . . .	12
v1.0q	<code>\usepicture</code> : Now <code>\usepspicture</code> works for all kind of graphics. (RN) . . . . .	13	<b>pspicture</b> : <code>pspicture</code> environment must still parse its arguments. (RN/HjG) . . . . .	12	
v1.0r	<code>\ppf@is@pdfTeX@graphic</code> : Changed <code>\ppf@is@known@graphic</code> to <code>\ppf@is@pdfTeX@graphic</code> . Now pdf $\TeX$ graphics are preferred. (RN) . . . . .	5	v1.1h	<code>\Gin@graphics</code> : Check if inside of a PS-related environment (correct graphic inclusion). (RN) . . . . .	8
v1.0s	General: Scaling e.g. of PostScript pictures now only in extraction mode. Some code cleaning. (RN) . . . . .	1	v1.1i	General: <code>\ifpr@outer</code> must be predefined. (HjG/RN) . . . . .	5
v1.1a	General: Support for the internal PSTricks macro <code>\pst@object</code> . (HjG/RN) . . . . .	8	Package option “final” also for “graphicx”. (RN) . . . . .	4	
v1.1b	General: Ignore the call of <code>\nofiles</code> inside of <code>preview</code> . (RN) . . . . .	9	<code>\Gin@graphics</code> : Correction of the inside check. (RN/HjG) . . .	8	
v1.1c	General: New package option “tightpage” added. (RN) . . . . .	1	v1.1k	General: New environment <code>pst-pdfdefs</code> : Support for PSTricks environment “psmatrix” inside user definitions. (RN,HjG) . . . . .	1
	Special support for “tabularx”. (RN) . . . . .	8	v1.1l	General: Support for the package “psfragx”. (RN) . . . . .	8
			v1.1m	General: Merge english and german version of the documentation. (RN) . . . . .	1
			v1.1n	General: <code>\nofiles</code> added (suggestion of Torsten Bronger). . . . .	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.

<b>Symbols</b>			
<code>\&amp;</code>	347, 359, 425, 426, 428	environments:postscript	<code>\Gininclude@graphics</code> .
<code>\@currenvir</code>	52, 343, 430	postscript . . . . . 2	. . . . . 55, <u>163</u> , 386
<code>\@currenvline</code>	.. 49, 424	environments:pspicture	<code>\GPT@page</code> . . . . 260, 269
<code>\@ehc</code>	. . . . . 382	pspicture . . . . . 2	<b>I</b>
<code>\@firstofone</code>	. . . . . 205	environments:pst-pdf-	<code>\if@ppf@pdftex@graphic</code>
<code>\@latex@error</code>	. . . . . 382	defs	. . . . . 58, 75
<code>\@makeother</code>	347, 359, 426	pst-pdf-defs . . . . 3	<code>\if@ppf@PST@used</code> . .
<code>\@ppf@PST@usedfalse</code>	10	<code>\everydisplay</code> . . . . 203	. . . . . 4, 81, 182
<code>\@ppf@PST@usedtrue</code>	4, 9	<b>F</b>	<code>\if@ppf@tightpage</code> .
<code>\@ppf@pdftex@graphicfalse</code>	. . . . . 65	<code>\filename@base</code> 317, 318	. . . . . 5, 101, 110
<code>\@ppf@pdftex@graphictrue</code>	. . . . . 71	<code>\filename@parse</code> . . . 310	<code>\ifGin@frame</code> . . . 59, 242
<code>\@ppf@tightpagefalse</code>	16	<code>\frame</code> . . . . . 238	<code>\ifGin@ignore</code> . . 62, 277
<code>\@ppf@tightpagetrue</code>	. . . . . 5, 17	<b>G</b>	<code>\ifGin@innerframe</code> .
<b>A</b>		<code>\Gin@bboxtrue</code> . . . . 393	. . . . . 60, 240
<code>\AtBeginDvi</code> . . . 183, 184		<code>\Gin@boolkey</code> . . . . .	<code>\ifGin@showname</code> 61, 281
<b>C</b>		. 232, 234, 236, 247	<code>\ifinner</code> . . . . . 131, 330
<code>\c@lor@to@ps</code> . . . . . 181		<code>\Gin@defaultbp</code> 391, 392	<code>\ifmmode</code> . . . . . 130, 329
<code>\c@pspicture</code> 274, 294,		<code>\Gin@extensions</code> 56,	<code>\ifpr@outer</code> . . . . .
295, 298, 309, 419		72, 89, 165, 228, 266	. 63, 164, 199, 416
<code>\catcode</code> . . . . . 425, 428		<code>\Gin@framefalse</code> . . . 59	<code>\includegraphics</code> . .
<code>\CurrentOption</code> 12, 15, 19		<code>\Gin@ignorefalse</code> . . 62	. . 2, 371, 402, 411
<b>D</b>		<code>\Gin@ii</code> 57, <u>261</u> , 348, 360	<code>\includegraphicx</code> . . . 3
<code>\DeclareGraphicsRule</code>		<code>\Gin@innerframefalse</code> 60	<b>J</b>
. . . . . 95, 230		<code>\Gin@llx</code> . . . . . 390, 397	<code>\jobname</code> . . 41, 316, 317
<code>\define@key</code> . 96–100,		<code>\Gin@lly</code> . . . . . 390, 395	<b>L</b>
231, 233, 235,		<code>\Gin@nat@height</code> . . .	<code>\leavevmode</code> . . . . . 400
237, 246, 248, 401		. 389, 392, 394, 395	<code>\long</code> . . . . . 198
<b>E</b>		<code>\Gin@nat@width</code> . . . .	<b>N</b>
<code>\endpostscript</code> 365, 368		. 389, 391, 396, 397	<code>\newcolumntype</code> . . . 158
<code>\endpsmatrix</code> . . . . .		<code>\Gin@PS@file@header</code> 177	<code>\nofiles</code> . . 80, 193, 195
. . . . 143, 147, 368		<code>\Gin@PS@literal@header</code>	<b>O</b>
<code>\endpspicture</code> . 362, 365		. . . . . 177	<code>\output</code> . . . . . 192, 196
<code>\endtabular</code> . . . . . 159		<code>\Gin@PS@raw</code> . . . . . 178	<b>P</b>
<code>\endtabularx</code> . . . . . 159		<code>\Gin@PS@restored</code> . . 178	<code>\paperheight</code> . . . . . 103
environments:		<code>\Gin@req@height</code> . . . 399	<code>\paperwidth</code> . . . . . 103
postscript . <u>107</u> , <u>327</u>		<code>\Gin@req@sizes</code> . . . . 398	<code>\PassOptionsToPackage</code>
psmatrix . . <u>125</u> , <u>327</u>		<code>\Gin@req@width</code> . . . . 399	. . . . . 12, 15, 19
pspicture . . <u>124</u> , <u>327</u>		<code>\Gin@setfile</code> 68, 257, <u>258</u>	<code>\PDFcontainer</code> . . . . .
pst-pdf-defs <u>46</u> , <u>421</u>		<code>\Gin@shownamefalse</code> . 61	40, 41, 268, 274,
		<code>\Gin@urx</code> . . . . . 391, 396	289, 291, 297,
		<code>\Gin@ury</code> . . . . . 392, 394	299, 310, 312, 412
		<code>\Gin@viewport@code</code> . 393	<code>\pdflastximage</code> . . . . 376
		<code>\Gin@vllx</code> . . . . . 114	
		<code>\Gin@vlly</code> . . . . . 114	
		<code>\Gin@vurx</code> . . . . . 114	
		<code>\Gin@vury</code> . . . . . 114	

<code>\pdflastximagepages</code>	292	<code>\ppf@namefont</code>	.....	<code>\PreviewMacro</code>	.....
<code>\pdfliteral</code>	..... 173		. 250, 251, 255, 282		. 149, 161, 172, 217
<code>\pdfoutput</code>	..... 26	<code>\ppf@nofiles</code>	.. 193, 195	<code>\psmatrix</code>	.... 140,
<code>\pdfrefximage</code>	..... 388	<code>\ppf@other@extensions</code>	..... 43,		142, 144, 145, 368
<code>\pdfTeXtext</code>	..... 67, 70		89, 93, 165, 228, 266	<code>psmatrix</code>	(environ-
<code>\pdftexversion</code>	.... 88	<code>\ppf@pr@psmatrix</code>	..		ment) ... <a href="#">125</a> , <a href="#">327</a>
<code>\pdfximage</code>	..... 291		..... 141, 146	<code>\pspicture</code>	..... 357
<code>\pfx@includegraphics</code>	..... 371	<code>\ppf@PreviewBbAdjust</code>	..... 83, 85, 109	<code>pspicture</code>	(environ-
<code>\pfx@includegraphicx</code>	..... 161, 372	<code>\ppf@psmatrix</code>	. 140, 144		ment) . <a href="#">2</a> , <a href="#">124</a> , <a href="#">327</a>
<code>\postscript</code>	... 364, 368	<code>\ppf@RestoreBbAdjust</code>	..... 84, 119	<code>pst-pdf-defs</code>	(environ-
<code>postscript</code>	(environ-	<code>\ppf@set@mode</code>	.....		ment) .. <a href="#">3</a> , <a href="#">46</a> , <a href="#">421</a>
	ment) . <a href="#">2</a> , <a href="#">107</a> , <a href="#">327</a>		.... 128, 132,	<code>\pst@@@picture</code>	.... 364
<code>\ppf@getpicture</code>	212,		134, 137, 146,	<code>\pst@object</code>	.....
	351, 362, 372, <a href="#">415</a>		147, 327, 331,		. 155, 223, 344, 345
<code>\ppf@container@max</code>	..... <a href="#">291</a> ,		333, 336, 349, 351	<code>\PSTricksOff</code>	..... 180
	295, 297, 307, 407	<code>\ppf@temp</code>	.....		
<code>\ppf@draft</code>	3, 13, 14, 411		. 183, 184, 425, 428		
<code>\ppf@endpsmatrix</code>	..	<code>\ppf@temptoken</code>	....		
	..... 143, 147		. 191, 192, 196, 197		
<code>\ppf@filename</code>	. 254,	<code>\ppf@test@mmode</code>	...		
	259, 283, 284, 387		. 129, 144, 328, 341		
<code>\ppf@getpicture</code>	...	<code>\ppf@TeX@mode</code>	. <a href="#">2</a> , <a href="#">6</a> –		
	.... <a href="#">381</a> , <a href="#">405</a> , 418		8, 21, 22, 33, 35,		
<code>\ppf@Gin@extensions</code>	56		37, 77, 79, 175, 433		
<code>\ppf@Gin@ii</code>	.... 57, 278	<code>\pr@cleanup</code>	... 201, 213		
<code>\ppf@Gin@keys</code>	. 256,	<code>\pr@endbox</code>	..... 209		
	342, 343, 414, 418	<code>\pr@outerfalse</code>	....		
<code>\ppf@Gin@setfile</code>	..		. 73, 204, 346, 358		
	..... 257, 258	<code>\pr@shipout</code>	..... 195		
<code>\ppf@Gin@include@graphics</code>	..... 55,	<code>\pr@startbox</code>	..... 198		
	73, 167, 169, 172	<code>\PreviewBbAdjust</code>	..		
<code>\ppf@is@pdfTeX@graphic</code>	. <a href="#">64</a> , 165, 266, 435		. 85, 102, 109, 113		
<code>\ppf@isnum</code>	... <a href="#">324</a> , 380	<code>\PreviewEnvironment</code>	. 120, 124, 127, 356		

## R

`\raisebox` ..... 283  
`\refstepcounter` 273, 417  
`\rule` ..... 166

## S

`\savepicture` . [3](#), 45, [375](#)  
`\setkeys` ..... 112, 263  
`\shipout` ..... 195  
`\string` ... 142, 145, 345  
`\strutbox` ..... 283

## T

`\tabular` ..... 157  
`\tabularx` ..... 157

## U

`\usepicture` .. [3](#), 44, [377](#)

## V

`\voidb@x` ..... 211