The pst-pdf package[*]

Rolf Niepraschk[†] Hubert Gäßlein

2008/05/02

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 v1.1r, dated 2008/05/02. Thanks to Peter Dyballa for the translation.

[†] `Rolf.Niepraschk@ptb.de`

least the size of the whole page. To be able to make use of the graphics' in a later pdfLaTeX 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`[1] 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 AMSLaTeX environments behave?)

⟨***other***⟩ All other options are passed to pstricks package.

2.2 Program calls

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

PostScript graphics	**bibliography**
`pdflatex document.tex`	`pdflatex document.tex`
auxiliary calls	
`latex document.tex`	
`dvips -o document-pics.ps document.dvi`	
`ps2pdf document-pics.ps`	`bibtex document.aux`
`pdflatex document.tex`	`pdflatex document.tex`

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}[`⟨*keys*⟩`] (`⟨*x0,x1*⟩`)(`⟨*y0,y1*⟩`) ...\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 pdfLaTeX mode this environment's contents is only displayed when the `\PDFcontainer` file was created before.

`postscript` `\begin{postscript}[`⟨*keys*⟩`] ...\end{postscript}`
The postscript environment can contain any code except floats. In pdfLaTeX 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[`⟨*keys*⟩`]{`⟨*filename*⟩`}`

[1] CTAN: `support/pdfcrop/`

[2] The TeX distribution "teTeX" 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 pdfLaTeX mode it is now additionally feasable to pass the name of an EPS file. Its visible contents too is taken from the `\PDFcontainer` file.

`\includegraphicx` `\includegraphicx[`⟨*keys*⟩`](`⟨*pfxadd*⟩`)<`⟨*ovpfgd*⟩`>[`⟨*ovpbgd*⟩`] {`⟨*filename*⟩`}`
Wie im Paket psfragx definiert zu verwenden.

`\savepicture` `\savepicture{`⟨*name*⟩`}`
The last output graphics (result of the pspicture or postscript environments or the `\includegraphics` statement with an PostScript file as argument) is being saved in a file under the name as given by the parameter.

`\usepicture` `\usepicture[`⟨*keys*⟩`]{`⟨*name*⟩`}`
Die zuvor mit `\savepicture` gespeicherte Grafik wird ausgegeben. Der optionale Parameter entpricht dem bei der Anweisung `\includegraphics` möglichen.

`pst-pdf-defs` `\begin{pst-pdf-defs}` ... `\end{pst-pdf-defs}`
Sollen eigene Makros oder Umgebungen definiert werden, die das Zeichen `&` (andere?) im Ersetzungstext enthalten, so müssen diese Definitionen von der Umgebung `pst-pdf-defs` 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 pdfLaTeX 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

```
⟨*package⟩
```

3.1 Package options

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

```
\newcommand*\ppf@draft{false}
\newif\if@ppf@PST@used\@ppf@PST@usedtrue
\newif\if@ppf@tightpage \@ppf@tightpagetrue
\DeclareOption{active}{\OptionNotUsed}
\DeclareOption{inactive}{\def\ppf@TeX@mode{9}}
\DeclareOption{ignore}{\def\ppf@TeX@mode{999}}
\DeclareOption{pstricks}{\@ppf@PST@usedtrue}
\DeclareOption{nopstricks}{\@ppf@PST@usedfalse}
\DeclareOption{displaymath}{%
  \PassOptionsToPackage\CurrentOption{preview}}
\DeclareOption{draft}{\def\ppf@draft{true}}
\DeclareOption{final}{\def\ppf@draft{false}%
  \PassOptionsToPackage\CurrentOption{graphicx}}
```

```
\DeclareOption{notightpage}{\@ppf@tightpagefalse}%
\DeclareOption{tightpage}{\@ppf@tightpagetrue}%
\DeclareOption*{%
  \PassOptionsToPackage\CurrentOption{pstricks}}
\ProcessOptions\relax
\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 teTeX/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.

```
\ifnum\ppf@TeX@mode=-1\relax
  \begingroup
```

Default (TeX with a dvi-to-ps converter)

```
    \chardef\x=0 %
```

Check pdfTeX

```
    \@ifundefined{pdfoutput}{}{%
      \ifcase\pdfoutput\else
        \chardef\x=1 %
      \fi
    }%
```

Check VTeX

```
    \@ifundefined{OpMode}{}{\chardef\x=2 }%
  \expandafter\endgroup
  \ifcase\x
```

⇒ DVI mode

```
    \def\ppf@TeX@mode{0}%
  \or
```

⇒ pdfTeX is running in PDF mode

```
    \def\ppf@TeX@mode{1}%
  \else
```

⇒ VTeX is running

```
    \def\ppf@TeX@mode{9}%
  \fi
\fi
```

```
\newcommand*\PDFcontainer{}
\edef\PDFcontainer{\jobname-pics.pdf}
\newcounter{pspicture}
\newcommand*\ppf@other@extensions[1]{}
\newcommand*\usepicture[2][]{}
\newcommand*\savepicture[1]{}
```

pst-pdf-defs

```
  \newenvironment*{pst-pdf-defs}%
  {%
    \endgroup
%    ??? \@currenvline
  }{%
    \begingroup
    \def\@currenvir{pst-pdf-defs}%
  }
```

```
\RequirePackage{graphicx}%
\let\ppf@Ginclude@graphics\Ginclude@graphics
\let\ppf@Gin@extensions\Gin@extensions
\let\ppf@Gin@ii\Gin@ii
```

```
\newif\if@ppf@pdftex@graphic
\newif\ifGin@frame\Gin@framefalse
\newif\ifGin@innerframe\Gin@innerframefalse
\newif\ifGin@showname\Gin@shownamefalse
\newif\ifGin@ignore\Gin@ignorefalse
```

`\ifpr@outer` in fact is defined in package preview. We have to do it here too since otherwise TEX could "stumble and fall" while parsing the \ifcase structure.

```
\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`.

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

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

```
    \def\Gin@setfile##1##2##3{%
      \edef\@tempb{##2}%
      \@for\@tempa:=\pdfTeXext\do{%
        \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".

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

Trial invocation. Output is completely inhibited.

```
    \pr@outerfalse\ppf@Ginclude@graphics{#1}%
  \endgroup
  \if@ppf@pdftex@graphic#4\else#5\fi
}
```

```
\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.

```
\PackageInfo{pst-pdf}{%
  MODE: \ppf@TeX@mode\space (dvi -- extraction mode)}
\nofiles
\let\makeindex\@empty \let\makeglossary\@empty
\AtBeginDocument{\overfullrule=\z@}%
\if@ppf@PST@used\RequirePackage{pstricks}\fi
\RequirePackage[active,dvips,tightpage]{preview}[2005/01/29]%
\newcommand*\ppf@PreviewBbAdjust{}
\newcommand*\ppf@RestoreBbAdjust{%
  \let\PreviewBbAdjust\ppf@PreviewBbAdjust}%
```

The pdfLATEX mode compliant graphics file formats are needed too.

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

```
\AtBeginDocument{%
  \@ifpackageloaded{keyval}{%
    \def\KV@errx#1{\PackageInfo{keyval}{#1}}%
    }{}%
  \@ifpackageloaded{xkeyval}{%
    \def\XKV@err#1{\PackageInfo{xkeyval}{#1}}%
  }{}%
```

In this mode undefined keys should not be an error.

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

No function in this mode.

```
\define@key{Gin}{innerframe}[true]{}%
\define@key{Gin}{frame}[true]{}%
\define@key{Gin}{ignore}[true]{}%
\define@key{Gin}{showname}[true]{}%
\define@key{Gin}{namefont}{}%

\if@ppf@tightpage\else
  \def\PreviewBbAdjust{%
```

```
    -600pt -600pt 600pt 600pt}%
  \AtEndDocument{%
    \PackageWarningNoLine{pst-pdf}{Picture container needs cropping.}}%
\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).

```
\newenvironment{postscript}[1][]%
{%
  \global\let\ppf@PreviewBbAdjust\PreviewBbAdjust
  \if@ppf@tightpage
    \begingroup
      \setkeys{Gin}{#1}%
      \xdef\PreviewBbAdjust{%
        -\Gin@vllx bp -\Gin@vlly bp \Gin@vurx bp \Gin@vury bp}%
    \endgroup
  \fi
  \ignorespaces
}%
{\aftergroup\ppf@RestoreBbAdjust}%
```

```
\PreviewEnvironment{postscript}%
\AtBeginDocument{%
  \@ifundefined{PSTricksLoaded}{}%
  {%
```

pspicture

Announce preview original definition.

```
    \PreviewEnvironment{pspicture}%
```

psmatrix

Announce preview original definition.

```
    \@ifundefined{psmatrix}{}%
    {%
      \PreviewEnvironment{psmatrix}%
      \newcommand*\ppf@set@mode{}%
      \newcommand*\ppf@test@mmode{%
      \ifmmode
        \ifinner
          \let\ppf@set@mode=$%
        \else
          \def\ppf@set@mode{$$}%
        \fi
      \else
        \let\ppf@set@mode=\@empty
      \fi
      }%
      \let\ppf@psmatrix=\psmatrix
      \expandafter\let\expandafter\ppf@pr@psmatrix%
        \expandafter=\csname pr@\string\psmatrix\endcsname
      \let\ppf@endpsmatrix=\endpsmatrix
      \def\psmatrix{\ppf@test@mmode\ppf@psmatrix}
      \expandafter\def\csname pr@\string\psmatrix\endcsname{%
        \ppf@set@mode\ppf@pr@psmatrix}%
      \def\endpsmatrix{\ppf@endpsmatrix\ppf@set@mode}%
    }%
```

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.

```
    \PreviewMacro[{{}*[]%
      ?\bgroup{#{#1}{{#1}}}{}%
      ?\bgroup{#{#1}{{#1}}}{}%
      ?({#{(#1)}{({#1})}}{}%
      ?({#{(#1)}{({#1})}}{}%
      ?({#{(#1)}{({#1})}}{}%
      }]{\pst@object}}
```

Prevent multiple test-wise setting of table contents by "tabularx".

```
  \@ifundefined{tabularx}{}{%
    \newcolumntype{X}{c}%
    \expandafter\let\expandafter\tabularx\csname tabular*\endcsname
    \expandafter\let\expandafter\endtabularx\csname endtabular*\endcsname
  }%
```

Support of `\includegraphicx` from the package psfragx.

```
  \@ifundefined{pfx@includegraphicx}{}{%
    \PreviewMacro[{{}{}}]{\pfx@includegraphicx}}%
}%
```

`\Gscale@@box` Disable scaling.

```
\def\Gscale@@box#1#2#3{%
  \toks@{\mbox}%
}
```

`\Ginclude@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.

```
\def\Ginclude@graphics#1{%
  \ifpr@outer
```

Generally pdfTEX supported graphics formats are intended to be preferred (inclusion in final pdfTEX 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.

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

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

```
    {\rule{10pt}{10pt}}%
    {\ppf@Ginclude@graphics{#1}}%
  \else
```

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

```
    \ppf@Ginclude@graphics{#1}%
```

```
  \fi
}%
```

```
\PreviewMacro[{{}}]{\ppf@Ginclude@graphics}%
\let\pdfliteral\@gobble%
\or
```

3.4 pdfLaTeX 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.

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

Necessary if PSTricks < 2.0.

```
    \PSTricksOff
    \@ifundefined{c@lor@to@ps}{\def\c@lor@to@ps#1 #2\@@{}}{}}}%
```

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

```
\if@ppf@PST@used
  \let\ppf@temp\AtBeginDvi\let\AtBeginDvi\@gobble
  \RequirePackage{pstricks}\let\AtBeginDvi\ppf@temp
\fi
```

PostScript output is now inhibited and later once again.

```
\the\@temptokena %%% ????
\expandafter\AtBeginDocument\expandafter
  {\the\@temptokena\@temptokena{}}%
\@ifundefined{PSTricksLoaded}{}
{%
```

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.

```
  \newtoks\ppf@temptoken
  \ppf@temptoken\expandafter{\the\output}%
  \let\output\@gobble
  \let\ppf@nofiles\nofiles \let\nofiles\relax
  \RequirePackage[active]{preview}[2005/01/29]%
  \let\shipout=\pr@shipout \let\nofiles\ppf@nofiles
  \output\expandafter{\the\ppf@temptoken}%
  \ppf@temptoken{}%
```

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

```
  \long\def\pr@startbox#1#2{%
  \ifpr@outer
    \toks@{#2}%
    \edef\pr@cleanup{\the\toks@}%
    \setbox\@tempboxa\vbox\bgroup
    \everydisplay{}%
    \pr@outerfalse%
    \expandafter\@firstofone
  \else
    \expandafter\@gobble
  \fi{#1}}%
  \def\pr@endbox{%
  \egroup
  \setbox\@tempboxa\box\voidb@x
  \ppf@@getpicture
  \pr@cleanup}%
```

(See also the identical definition in DVI mode.)

```
  \AtBeginDocument{%
    \@ifundefined{pst@object}{}%
    {%
      \PreviewMacro[{{}*[]%
        ?\bgroup{#{#1}{{#1}}}{}%
        ?\bgroup{#{#1}{{#1}}}{}%
        ?({#{(#1)}{({#1})}}{}%
        ?({#{(#1)}{({#1})}}{}%
        ?({#{(#1)}{({#1})}}{}%
        }]{\pst@object}}%
    }%
  }%
```

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

```
\begingroup
  \input{dvips.def}%
  \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}}%
\x
```

Dummy definition for in DVI mode supported file formats.

```
\DeclareGraphicsRule{*}{eps}{*}{}%
```

```
\define@key{Gin}{innerframe}[true]{%
  \lowercase{\Gin@boolkey{#1}}{innerframe}}%
\define@key{Gin}{frame}[true]{%
  \lowercase{\Gin@boolkey{#1}}{frame}}%
\define@key{Gin}{ignore}[true]{%
  \lowercase{\Gin@boolkey{#1}}{ignore}}%
\define@key{Gin}{frame@@}{%
```

(For internal use only!)

```
  \edef\@tempa{\toks@{\noexpand\frame{\the\toks@}}}%
  \ifcase#1\relax
    \ifGin@innerframe\else\let\@tempa\relax\fi
  \or
    \ifGin@frame\else\let\@tempa\relax\fi
  \fi
  \@tempa
}%
\define@key{Gin}{showname}[true]{%
  \lowercase{\Gin@boolkey{#1}}{showname}}%
\define@key{Gin}{namefont}{%
```

```
  \begingroup
    \@temptokena\expandafter{\ppf@namefont#1}%
    \edef\x{\endgroup\def\noexpand\ppf@namefont{\the\@temptokena}}%
  \x
}%
\newcommand*\ppf@filename{}%
\newcommand*\ppf@namefont{\tiny\ttfamily}%
\newcommand*\ppf@Gin@keys{}%
\let\ppf@Gin@setfile\Gin@setfile
```

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

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

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

```
\def\Gin@ii[#1]#2{%
  \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.

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

Finds empty file name when calling \usepicture.

```
    \ifx\@tempb\@empty\else
      \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.

```
      {%
        \ifx\@tempb\PDFcontainer
          \@temptokena{page=\GPT@page}%
        \fi
      }%
      {%
        \refstepcounter{pspicture}%
        \@temptokena{page=\the\c@pspicture}\def\@tempb{\PDFcontainer}%
      }%
    \fi
    \ifGin@ignore\else
```

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

```
      \edef\@tempa{\noexpand\ppf@Gin@ii[frame@@=0,\the\@temptokena,
        frame@@=1]{\@tempb}}%
      \@tempa
      \ifGin@showname
        \ppf@namefont
        \raisebox{-\ht\strutbox}[0pt][0pt]{\llap{\ppf@filename}}%
        \gdef\ppf@filename{}%
      \fi
    \fi
  \endgroup
}%
```

```
\IfFileExists{\PDFcontainer}%
{%
```

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

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

```
  \AtEndDocument{%
    \ifnum\c@pspicture>\z@
```

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

```
      \ifnum\c@pspicture=\ppf@container@max\else
        \PackageWarningNoLine{pst-pdf}{%
          `\PDFcontainer' contains \ppf@container@max\space pages
          \MessageBreak but \the\c@pspicture\space pages are requested:
          \MessageBreak File `\PDFcontainer' is no more valid!
          \MessageBreak Recreate it
        }%
      \fi
    \fi
  }%
}%
{%
  \def\ppf@container@max{0}%
  \AtEndDocument{%
    \ifnum\c@pspicture>\z@
      \filename@parse{\PDFcontainer}%
      \PackageWarningNoLine{pst-pdf}{%
        File `\PDFcontainer' not found.\MessageBreak
        Use the following commands to create it:\MessageBreak
        ----------------------------------------------------
        \MessageBreak
        latex \jobname.tex\MessageBreak
        dvips -o \filename@base.ps \jobname.dvi\MessageBreak
        ps2pdf \filename@base.ps\MessageBreak
        ----------------------------------------------------
      }%
    \fi
  }%
}%
```

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

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

`postscript` `pspicture` 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`

```
\newcommand*\ppf@set@mode{}%
\newcommand*\ppf@test@mmode{%
```

```
\ifmmode
  \ifinner
    \let\ppf@set@mode=$%
  \else
    \def\ppf@set@mode{$$}%
  \fi
\else
  \let\ppf@set@mode=\@empty
\fi
}
\newenvironment{postscript}[1][]
{%
  \ppf@test@mmode
  \gdef\ppf@Gin@keys{}%
  \def\@tempa{postscript}\ifx\@tempa\@currenvir\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.

```
  \expandafter\let\expandafter\pst@object
    \csname pr@\string\pst@object\endcsname
  \pr@outerfalse
```

Needed for `\psmatrix`.

```
  \@makeother\&%
  \def\Gin@ii[##1]##2{}\setbox\@tempboxa=\vbox\bgroup
  \ppf@set@mode
}%
{\ppf@set@mode\egroup\aftergroup\ppf@@getpicture}%
\AtBeginDocument{%
  \@ifundefined{PSTricksLoaded}{}%
  {%
    \iffalse
      \PreviewEnvironment{pspicture}% Why doesn't it work?
      \g@addto@macro\pspicture{%
        %%\pr@outerfalse% necessary, or already there anyway?
        \@makeother\&% necessary?
        \def\Gin@ii[#1]#2{}%
      }%
      \g@addto@macro\endpspicture{\ppf@@getpicture}%
    \else
      \def\pst@@@picture[#1](#2,#3)(#4,#5){\postscript}%
      \def\endpspicture{\endpostscript\endgroup}%
    \fi
    \@ifundefined{psmatrix}{}%
    {\let\psmatrix=\postscript\let\endpsmatrix=\endpostscript}%
  }%
  \@ifundefined{pfx@includegraphicx}{}{%
```

Die im pdfTeX-Modus unnütze Umdefinition von `\includegraphics` (Paket psfragx) führt zu zweifachem Einfügen des Ergebnisses, weshalb die Originaldefition wiederhergestellt wird.

```
    \let\includegraphics=\pfx@includegraphics
    \def\pfx@includegraphicx#1#2{\ppf@@getpicture}%
  }%
}%
```

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

```
\def\savepicture#1{%
  \expandafter\xdef\csname ppf@@@#1\endcsname{\the\pdflastximage}}%
```

\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.

```
\renewcommand*\usepicture[2][]{%
  \@ifundefined{ppf@@@#2}%
  {%
    \ppf@isnum{#2}%
    {\ppf@getpicture{#1}{#2}}%
    {\@latex@error{picture ‘#2’ undefined}\@ehc}%
  }%
  {%
    \begingroup
      \def\Ginclude@graphics##1{%
        \xdef\ppf@filename{#2}%
        \setbox\z@\hbox{\pdfrefximage\@nameuse{ppf@@@#2}}%
        \Gin@nat@height\ht\z@ \Gin@nat@width\wd\z@
        \def\Gin@llx{0} \let\Gin@lly\Gin@llx
        \Gin@defaultbp\Gin@urx{\Gin@nat@width}%
        \Gin@defaultbp\Gin@ury{\Gin@nat@height}%
        \Gin@bboxtrue\Gin@viewport@code
        \Gin@nat@height\Gin@ury bp%
        \advance\Gin@nat@height-\Gin@lly bp%
        \Gin@nat@width\Gin@urx bp%
        \advance\Gin@nat@width-\Gin@llx bp%
        \Gin@req@sizes
        \ht\z@\Gin@req@height \wd\z@\Gin@req@width
        \leavevmode\box\z@}%
      \define@key{Gin}{type}{}%
      \includegraphics[scale=1,#1]{}%
    \endgroup
  }}%
```

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

```
\newcommand*\ppf@getpicture[2]{%
  \@tempcnta=#2\relax%
  \ifnum\@tempcnta>\ppf@container@max
    \PackageWarningNoLine{pst-pdf}{%
      pspicture No. \the\@tempcnta\space undefined}%
  \else
    \includegraphics[draft=\ppf@draft,#1,page=\the\@tempcnta]%
      {\PDFcontainer}%
  \fi
  \gdef\ppf@Gin@keys{}}%
```

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

```
\newcommand*\ppf@@getpicture{%
  \ifpr@outer
```

```
     \refstepcounter{pspicture}%
     \expandafter\ppf@getpicture\expandafter{\ppf@Gin@keys}%
     {\the\c@pspicture}%
   \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!?)

```
 \renewenvironment*{pst-pdf-defs}%
 {%
   \endgroup
%    ??? \@currenvline
   \chardef\ppf@temp=\catcode`\&%
   \@makeother\&%
 }{%
   \catcode`\&=\ppf@temp
   \begingroup
   \def\@currenvir{pst-pdf-defs}%
 }
```

```
\else
```

3.5 Inactive Mode

Only the packages pstricks and graphicx are loaded – no further exertion of influence. The package option “`inactive`” as soon as the VTeXcompiler force this mode.

```
 \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (inactive mode)}%
 \newenvironment{postscript}[1][]{\ignorespaces}{}
 \let\ppf@is@pdfTeX@graphic\relax
\fi
```

```
\InputIfFileExists{pst-pdf.cfg}{%
 \PackageInfo{pst-pdf}{Local config file pst-pdf.cfg used}}{}
⟨/package⟩
```

Change History

v1.0a

General: Initial version. 1

v1.0b

General: Some code and documentation cleaning. (RN) 1

v1.0c

General: New options “pstricks”, “nopstricks”, “draft” and “final”. (RN) 3

v1.0d

General: Redefinition of `\includegraphics` in modes 0 und 1. Now using of eps graphics directly in pdfLaTeX is possible. (RN) 1

v1.0e

`postscript`: “trim” option added. (RN) 7

v1.0f

General: Config file loading added. (RN) 15

`\savepicture`: New macro `\savepspicture`. (RN) 14

`\usepicture`: New macro

`\usepspicture`. Useful for putting a PSTricks graphic in a box or something else. (RN) 14

v1.0g

- General: Definition of `\PDFcontainer` now with `\edef`. (RN) 5
- `\usepicture`: Now `\usepspicture` does accept a numerical parameter. (RN) 14

v1.0h

- `postscript`: Based no more on the comment environment from the verbatim package. (RN) 12

v1.0i

- `\ppf@is@pdfTeX@graphic`: No more errors for given files without extensions. (RN) 5

v1.0j

- General: Check AtBeginDocument for package 'pstricks' even if "nopstricks" is given. (RN) 1

v1.0k

- `\Gin@setfile`: Show also the pagenumber if exists. (RN) 11
- `\Ginclude@graphics`: Prevent division by zero. (RN) 8

v1.0l

- General: Options "framesep", "framerule", "linewidth" removed, "fname" and "innerframe" added. (RN) 1

v1.0m

- General: New package option "notightpage" added. (RN) 1

v1.0n

- General: Changed marcro names (`\savepicture` and `\usepicture`). (RN) 1
  - Some code cleaning. (RN) 1

v1.0o

- General: New code for "notightpage". (RN) 6
  - Option "fname" renamed to "showname". (RN) 1

v1.0p

- General: Some code and documentation cleaning. (RN) 1

v1.0q

- `\usepicture`: Now `\usepspicture` works for all kind of graphics. (RN) 14

v1.0r

- `\ppf@is@pdfTeX@graphic`: Changed `\ppf@is@known@graphic` to `\ppf@is@pdfTeX@graphic`. Now pdfTEX graphics are prefered. (RN) 5

v1.0s

- General: Scaling e.g. of PostScript pictures now only in extraction mode. Some code cleaning. (RN) 1
- `\Gin@ii`: Rewritten. (RN) 11

v1.1a

- General: Support for the internal PSTricks macro `\pst@object`. (HjG/RN) 8

v1.1b

- General: Ignore the call of `\nofiles` inside of preview. (RN) 9
  - Some code and documentation cleaning. (RN) 1

v1.1c

- General: New package option "tightpage" added. (RN) 1
  - Special support for "tabularx". (RN) 8
  - Supress handling of pdfLATEX graphic formats in DVI mode. (RN) 6

v1.1d

- `psmatrix`: Support for PSTricks environment "psmatrix". (RN) 12

v1.1e

- General: New option "displaymath" (see preview package). (HjG/RN) 3

v1.1f

- General: Package option "ignore" reimplemented. Now the compilation of the dtx file in LATEX mode is possible. (RN) 3

v1.1g

- `psmatrix`: "psmatrix" environment (preserve math mode). (RN/HjG) 12
- `pspicture`: pspicture environment must still parse its arguments. (RN/HjG) 12

v1.1h

- `\Ginclude@graphics`: Check if inside of a PS-related environment (correct graphic inclusion). (RN) 8

v1.1i

- General: `\ifpr@outer` must be predefined. (HjG/RN) 5

Package option “final” also for “graphicx”. (RN) 4

`\Ginclude@graphics`: Correction of the inside check. (RN/HjG) . . . 8

v1.1k

General: New environment pst-pdf-defs: Support for PSTricks environment “psmatrix” inside user definitions. (RN,HjG) 1

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: `\nofiles` added (suggestion of Torsten Bronger). 6

v1.1o

`\Gscale@@box`: Disable scaling. (RN) 8

v1.1p

General: `\nofiles` makes `\makeindex` and `\makeglossary` to `\relax`. `\@empty` is better because of later `\renewcommand`’s. 6

v1.1p1

General: `\let\output\@gobble` before loading of “preview” added. (RN) 9

v1.1q

General: Problem with “tabularx” and “threeparttabel” solved. (RN) 8

v1.1r

General: Fixed values for `\PreviewBbAdjust` because `\paperwidth` is not allways defined (suggested by Will Robertson). 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.