

# Das Paket `pst-pdf`\*

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

2008/09/16

## 1 Einleitung

Das Paket `pst-pdf` vereinfacht die Verwendung von PSTricks-Grafiken und anderem PostScript-Code in PDF-Dokumenten. Ähnlich wie beim Erstellen des Literaturverzeichnisses mit `bibTeX` werden zusätzlich externe Programme aufgerufen. Sie dienen in diesem Fall dazu, eine PDF-Datei, die sämtliche Grafiken enthält, zu erzeugen. Ihr Inhalt wird im endgültigen Dokument statt des ursprünglichen PostScript-Codes eingefügt.

## 2 Anwendung

### 2.1 Paketoptionen

**active** Aktiviert den Extraktionsmodus (DVI-Ausgabe). Die explizite Angabe ist normalerweise unnötig (Standard im `LATEX`-Modus).

**inactive** Keine besonderen Aktionen; es werden nur die Pakete `pstricks` und `graphicx` geladen (Standard bei Verwendung von `VTEX`). Kann dazu benutzt werden, um das Dokument mit `LATEX` in eine DVI-Datei zu wandeln und dabei die automatische Verwendung des Extraktionsmodus<sup>1</sup> zu vermeiden.

**pstricks** Das Paket `pstricks` wird geladen (Standard).

**nopstricks** Das Paket `pstricks` wird nicht geladen. Wird später festgestellt, dass `pstricks` doch noch anderweitig geladen wurde, wird die Umgebung `pspicture` nachträglich in der Weise behandelt, als wäre die Option “`pstricks`” doch angegeben worden.

**draft** Im `pdfLATEX`-Modus werden aus der Containerdatei eingefügte Grafiken nur als Rahmen dargestellt.

**final** Im `pdfLATEX`-Modus werden aus der Containerdatei eingefügte Grafiken vollständig dargestellt (Standard).

**tightpage** Die Abmessung Grafiken in der Containerdatei entsprechen denen der zugehörigen `TEX`-Boxen (Standard).

---

\*Dieses Dokument bezieht sich auf `pst-pdf` v1.1u vom 2008/09/16.

<sup>†</sup>`Rolf.Niepraschk@gmx.de`

**notightpage** Die Abmessung der zur Grafik gehörenden  $\TeX$ -Box ist manchmal nicht korrekt, da PostScript-Anweisungen auch außerhalb der Box zeichnen können. Die Option “notightpage” führt dazu, dass die Grafiken in der Containerdatei mindestens die Größe des gesamten Blattes einnehmen. Um die Grafiken im späteren pdf $\LaTeX$ -Lauf verwenden zu können, muss die Containerdatei nachbearbeitet werden, so dass die Größe der Grafiken auf die der sichtbaren Bestandteile reduziert ist. Dazu kann z. B. das Programm `pdfcrop`<sup>1</sup> dienen. Die Anwendung dieses Verfahrens kann die Angabe der Option “trim” erübrigen (siehe Abschnitt 2.4).

**displaymath** Es werden zusätzlich die mathematischen Umgebungen `displaymath`, `eqnarray` und `$$` extrahiert und im pdf-Modus als Grafik eingefügt. So können zusätzliche PSTricks-Ergänzungen leicht dem Inhalt dieser Umgebungen zugefügt werden. (Frage: Wie verhalten sich die AMS $\LaTeX$ -Umgebungen?)

**<other>** Alle anderen Optionen werden an das Paket `pstricks` weitergereicht.

## 2.2 Programmaufrufe

Die folgende Tabelle zeigt den Ablauf, der nötig ist, um ein PDF-Dokument mit PostScript-Grafiken zu erzeugen<sup>2</sup>. Im Vergleich dazu ist der analoge Ablauf für Literaturverzeichnisse angegeben.

PostScript-Grafiken	Literaturverzeichnis
<code>pdflatex document.tex</code>	<code>pdflatex document.tex</code>
<i>Hilfsaufrufe</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>

Bei der Erzeugung wird nur Code berücksichtigt, der sich innerhalb der Umgebungen `pspicture` oder `postscript` befindet. Ebenfalls werden PostScript-Grafiken, die als Parameter von `\includegraphics` angegeben wurden, der Containerdatei hinzugefügt. Der Name dieser Datei ist standardmäßig `\jobname-pics.pdf`. Er kann durch Undefinieren des Makros `\PDFcontainer` geändert werden.

## 2.3 Nutzeranweisungen

**pspicture** `\begin{pspicture}[\langle keys \rangle] (\langle x0,x1 \rangle) (\langle y0,y1 \rangle) ... \end{pspicture}`  
Die `pspicture`-Umgebung steht zur Verfügung, wenn nicht die Option “nopstricks” angegeben wurde. Sie wird so wie in PSTricks üblich verwendet. Im pdf $\LaTeX$ -Modus wird ihr Inhalt nur dann dargestellt, wenn vorher die Containerdatei erzeugt wurde.

**postscript** `\begin{postscript}[\langle keys \rangle] ... \end{postscript}`

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

<sup>2</sup>Die Shell-Skripte `ps4pdf` bzw. `ps4pdf.bat` führen alle angegebenen Programmaufrufe automatisch aus.

Die `postscript`-Umgebung kann beliebigen Code mit Ausnahme von Gleitumgebungen aufnehmen. Im pdfL<sup>A</sup>T<sub>E</sub>X-Modus wird ihr Inhalt ebenfalls der Containerdatei entnommen. Ist diese Datei nicht vorhanden, wird – anders als bei der `pspicture`-Umgebung – der später benötigte Platz möglicherweise nicht korrekt frei gehalten.

<code>\includegraphics</code>	<code>\includegraphics[⟨keys⟩]{⟨filename⟩}</code> Wie in <code>graphics/graphicx</code> definiert zu verwenden. Zusätzlich ist es nun möglich, auch im pdfL <sup>A</sup> T <sub>E</sub> X-Modus EPS-Dateien als Argument anzugeben und ihren Inhalt darzustellen. Er wird dazu ebenfalls der Containerdatei entnommen.
<code>\includegraphicsx</code>	<code>\includegraphicsx[⟨keys⟩](⟨pfxadd⟩)&lt;⟨ovpfgd⟩&gt;[⟨ovpbgd⟩]{⟨filename⟩}</code> Wie im Paket <code>psfragx</code> definiert zu verwenden.
<code>\savepicture</code>	<code>\savepicture{⟨name⟩}</code> Die zuletzt ausgegebene Grafik (Ergebnisse der Umgebungen <code>pspicture</code> , <code>postscript</code> und der <code>\includegraphics</code> -Anweisungen mit PostScript-Dateien) wird unter dem als Parameter übergebenen Namen gespeichert.
<code>\usepicture</code>	<code>\usepicture[⟨keys⟩]{⟨name⟩}</code> Die zuvor mit <code>\savepicture</code> gespeicherte Grafik wird ausgegeben. Der optionale Parameter entpricht 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

Das Verhalten der Anweisungen `\includegraphics`, `\usepicture` und der Umgebung `postscript` kann mit den folgenden optionalen Parametern beeinflusst werden (key-value-Syntax):

**frame**=⟨true|false⟩ Es wird – ähnlich wie bei der Anweisung `\fbox` – ein Rahmen um die Grafik gezeichnet. Die durch Rotation geänderte Gesamtgröße wird dabei berücksichtigt. Das Zeichnen geschieht im pdfL<sup>A</sup>T<sub>E</sub>X-Modus; vorher beim Erzeugen der Containerdatei wird dieser Parameter ignoriert. Standard: false.

**innerframe**=⟨true|false⟩ Wie “**frame**” jedoch wird der Rahmen nur um die Grafik selbst, nicht aber um die resultierende Box gezeichnet.

**ignore**=⟨true|false⟩ Bei “**true**” wird die Grafik nicht ausgegeben. Bei Angabe von `\savepicture{⟨name⟩}` kann sie später jedoch an anderer Stelle mit `\usepicture` verwendet werden. Standard: false.

**showname**=⟨true|false⟩ Gibt in kleiner Schrift den tatsächlich verwendeten Dateinamen unter der Grafik aus. Standard: false.

**namefont**=⟨font commands⟩ Beeinflusst die Schriftart, die bei “**showname=true**” benutzt wird. Standard: `\ttfamily\tiny`

Alle Parameter können auch global per `\setkeys{Gin}{⟨key=value⟩}` gesetzt werden.

### 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}{\OptionNotUsed}
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 Compilertests

Es wird getestet, welcher T<sub>E</sub>X compiler in welchem Modus läuft (siehe ‘graphics.cfg’ von t<sub>E</sub>X/L<sub>AT</sub>E<sub>X</sub>Live). Entsprechend dem Ergebnis bekommen die Umgebungen `pspicture` und `postscript` unterschiedliche Funktionalität. Der Test wird nur ausgeführt, wenn nicht die Paketoptionen `active` oder `inactive` angegeben wurden.

```
22 \ifnum\ppf@TeX@mode=-1\relax
23   \begingroup
Default (TEX with a dvi-to-ps converter)
24   \chardef\x=0 %
Check pdfTEX
25   \@ifundefined{pdfoutput}{}{%
26     \ifcase\pdfoutput\else
27       \chardef\x=1 %
28     \fi
29   }%
Check VTEX
30   \@ifundefined{OpMode}{}{\chardef\x=2 }%
31   \expandafter\endgroup
32   \ifcase\x
⇒ DVI mode
33   \def\ppf@TeX@mode{0}%
34   \or
```

```

⇒ pdfTeX is running in PDF mode
35   \def\ppf@TeX@mode{1}%
36   \else
⇒ VTeX 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][1]{}
45 \newcommand*\savepicture[1]{}

```

#### pst-pdf-defs

```

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

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

57 \newif\ifppf@pdftex@graphic
58 \newif\ifGin@frame\Gin@framefalse
59 \newif\ifGin@innerframe\Gin@innerframefalse
60 \newif\ifGin@showname\Gin@shownamefalse
61 \newif\ifGin@ignore\Gin@ignorefalse

```

\ifpr@outer wird eigentlich im Paket preview definiert. Wir müssen es aber bereits hier zusätzlich tun, da sonst T<sub>E</sub>X u. U. beim Parsen der \ifcase-Struktur “außer Tritt” kommt.

```

62 \newif\ifpr@outer

```

**\ppf@is@pdfTeX@graphic** Parameter #1 ist der Name einer Grafikdatei mit oder ohne Endung, Parameter #2 enthält die gültigen Dateieendungen im pdf-Modus, Parameter #3 enthält die gültigen Dateieendungen im dvi-Modus. Ist es möglich, die Grafik im pdf-Modus zu verarbeiten, werden die Anweisungen in #4 ausgeführt, sonst die in #5.

```

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

```

Statt Einladen einer identifizierten Grafik nur Test der Grafikendung.

```

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

```

Es müssen Dateitypen beider Moden gefunden werden, um die Fehlermeldung “File ‘#1’ not found” zu vermeiden.

```

71 \edef\Gin@extensions{#2,#3}%
Testaufruf. Dabei Ausgabe vollständig verhindern.
72 \pr@outerfalse\ppf@Gininclude@graphics{#1}%
73 \endgroup
74 \ifppf@pdftex@graphic#4\else#5\fi
75 }

76 \ifcase\ppf@TeX@mode\relax

```

### 3.3 Extraction mode (dvi output)

Die Umgebung `pspicture` behält die Definition aus `pstricks.tex`. Ausschließlich der Code der Umgebungen `pspicture` und `postscript` sowie `\includegraphics` mit PS-Dateien bewirken Einträge in die DVI-Datei. Der restliche Code des Dokuments wird bei der Ausgabe der DVI-Datei ignoriert. Nach Wandlung der DVI-Datei über PostScript (“dvips”) nach PDF (Datei `\PDFcontainer`) nimmt jede Grafik genau eine Seite der pdf-Datei ein. Der TeX-Compiler mit DVI-Ausgabe sowie die Paketoption “active” erzwingen diesen Modus.

```

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

```

Es werden auch die im pdf<sub>La</sub>TeX-Modus erlaubten Endungen von Grafikdateien benötigt.

```

87 \begingroup
88 \let\AtBeginDocument\@gobble \let\PackageWarningNoLine\@gobbletwo
89 \chardef\pdftexversion=121 %
90 \chardef\pdfoutput=1 %
91 \input{pdftex.def}%
92 \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}
93 }%
94 \x

```

Für die im PDF-Modus möglichen Grafikformate dürfen keine speziellen Regeln definiert sein (z. B. wegen ‘dvips’-Erweiterungen). Für sie wird die universelle EPS-Regel verwendet, damit sie zumindest gefunden werden.

```

95 \AtBeginDocument{%
96   \@ifpackageloaded{keyval}{%
97     \def\KV@errx#1{\PackageInfo{keyval}{#1}}%
98   }{}%
99   \@ifpackageloaded{xkeyval}{%
100     \def\XKV@err#1{\PackageInfo{xkeyval}{#1}}%
101   }{}%

```

In diesem Modus sollten undefinierte keys keinen Fehler bewirken.

```

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

In diesem Modus keine Funktion.

105 \define@key{Gin}{innerframe}[true]{}%
106 \define@key{Gin}{frame}[true]{}%
107 \define@key{Gin}{ignore}[true]{}%
108 \define@key{Gin}{showname}[true]{}%
109 \define@key{Gin}{namefont}{}%
110 \ifundefined{GPT@page}{\define@key{Gin}{page}{}{}}

111 \ifppf@tightpage\else
112 \def\PreviewBbAdjust{%
113 -600pt -600pt 600pt 600pt}%
114 \AtEndDocument{%
115 \PackageWarningNoLine{pst-pdf}{Picture container needs cropping.}}%
116 \fi

```

**postscript** Die Umgebung postscript wertet die trim-Option in derselben Weise wie `\includegraphics` aus (Angaben ohne Maßeinheit werden als bp interpretiert).

```

117 \newenvironment{postscript}[1][]{%
118 {%
119 \global\let\ppf@PreviewBbAdjust\PreviewBbAdjust
120 \ifppf@tightpage
121 \begingroup
122 \setkeys{Gin}{#1}%
123 \xdef\PreviewBbAdjust{%
124 -\Gin@vllx bp -\Gin@vlly bp \Gin@vurx bp \Gin@vury bp}%
125 \endgroup
126 \fi
127 \ignorespaces
128 }%
129 {\aftergroup\ppf@RestoreBbAdjust}%

130 \PreviewEnvironment{postscript}%
131 \AtBeginDocument{%
132 \ifundefined{PSTricksLoaded}{}%
133 {%

```

**pspicture** Originaldefinition preview bekannt machen.

```

134 \PreviewEnvironment{pspicture}%

```

**psmatrix** Originaldefinition preview bekannt machen.

```

135 \ifundefined{psmatrix}{}%
136 {%
137 \PreviewEnvironment{psmatrix}%
138 \newcommand*\ppf@set@mode{}%
139 \newcommand*\ppf@test@mmode{%
140 \ifmmode
141 \ifinner
142 \let\ppf@set@mode=$%
143 \else

```

```

144         \def\ppf@set@mode{$$}%
145         \fi
146     \else
147         \let\ppf@set@mode=\@empty
148     \fi
149 }%
150 \let\ppf@psmatrix=\psmatrix
151 \expandafter\let\expandafter\ppf@pr@psmatrix%
152     \expandafter=\csname pr@\string\psmatrix\endcsname
153 \let\ppf@endpsmatrix=\endpsmatrix
154 \def\psmatrix{\ppf@test@mode\ppf@psmatrix}
155 \expandafter\def\csname pr@\string\psmatrix\endcsname{%
156     \ppf@set@mode\ppf@pr@psmatrix}%
157 \def\endpsmatrix{\ppf@endpsmatrix\ppf@set@mode}%
158 }%

```

Internes Makro `\pst@object` bekanntmachen, um manchen PSTricks-Code außerhalb von `pspicture`-Umgebungen ebenfalls verwenden zu können. Derzeit sind Aufrufe der folgenden Art möglich:

```

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

```

Mehr als drei optionale Argumente am Ende des Aufrufs, wie beispielsweise bei `\psline` denkbar, sind noch nicht möglich.

```

159     \PreviewMacro[{{}*[]}%
160     ?\bgroup{#{#1}{#{1}}}{}%
161     ?\bgroup{#{#1}{#{1}}}{}%
162     ?({#{(1)}({#{1})})}{}%
163     ?({#{(1)}({#{1})})}{}%
164     ?({#{(1)}({#{1})})}{}%
165     }]\pst@object}}

```

Mehrfaches testweises Setzen von Tabelleninhalten durch “`tabularx`” verhindern.

```

166     \@ifundefined{tabularx}{}{%
167         \newcolumntype{X}{c}%
168         \expandafter\let\expandafter\tabularx\csname tabular*\endcsname
169         \expandafter\let\expandafter\endtabularx\csname endtabular*\endcsname
170     }%

```

Unterstützung von `\includegraphicx` aus dem Paket `psfrag`.

```

171     \@ifundefined{pfx@includegraphicx}{}{%
172         \PreviewMacro[{{}}]{\pfx@includegraphicx}}%
173     }%

```

`\Gscale@@box` Skalieren verhindern.

```

174     \def\Gscale@@box#1#2#3{%
175         \toks@{\mbox}%
176     }

```

`\Ginclude@graphics` Alle Grafiken mit bekanntem Format (z. B. EPS-Dateien) werden normal verarbeitet, was in diesem Modus bedeutet, dass sie der Preview-Funktionalität unterliegen. Andere Grafiken (z. B. PDF-Dateien) werden ignoriert.

```

177     \def\Ginclude@graphics#1{%
178         \ifpr@outer

```



Im allgemeinen Fall sollen pdfTeX-Grafiken bevorzugt werden (Einfügen erst im pdfTeX-Modus). Ist nur eine DVIPS-Graphik vorhanden, dann wirkt wieder die Originaldefinition und Registrierung beim preview-Paket muss erfolgen.

```

179     \ppf@is@pdfTeX@graphic{#1}{\ppf@other@extensions}{\Gin@extensions}%
    Dummy-Box, um Division durch Null bei Skalierung/Rotation zu vermeiden. Wird
    ansonsten ignoriert.
180     {\rule{10pt}{10pt}}%
181     {\ppf@Gin@include@graphics{#1}}%
182     \else
    Innerhalb von PS-Umgebungen (pspicture usw.) muss sich \includegraphics
    wie die Originaldefinition verhalten (nur die DVIPS-Graphik-Typen sind gültig).
183     \ppf@Gin@include@graphics{#1}%
184     \fi
185 }%

186 \PreviewMacro[{}]{\ppf@Gin@include@graphics}%
187 \let\pdfliteral\@gobble%
188 \or

```

### 3.4 pdfL<sup>A</sup>T<sub>E</sub>X mode (pdf output)

Ist die Datei \PDFcontainer (default: <jobname>-pics.pdf) vorhanden, so wird der Inhalt der Umgebungen pspicture und postscript ignoriert. Stattdessen wird die zugehörige Grafik aus der Datei \PDFcontainer eingebunden.

```

189 \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (pdfTeX mode)}%
    Verhindert pdfTeXs Warnung Non-PDF special ignored!.
190 \if@ppf@PST@used
191     \let\ppf@temp\AtBeginDvi\let\AtBeginDvi\@gobble
192     \RequirePackage{pstricks}\let\AtBeginDvi\ppf@temp
193 \fi
194 \@temptokena{%
195     \let\Gin@PS@file@header\@gobble\let\Gin@PS@literal@header\@gobble
196     \let\Gin@PS@raw\@gobble\let\Gin@PS@restored\@gobble
197     \@ifundefined{PSTricksLoaded}{-}{%

```

Für PSTricks < 2.0 nötig.

```

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

```

PostScript-Ausgabe jetzt verhindern und später noch einmal.

```

200 \the\@temptokena
201 \expandafter\AtBeginDocument\expandafter
202 {\the\@temptokena\@temptokena{}}%
203 \@ifundefined{PSTricksLoaded}{-}{%

```

Zum Parsen der Argumente von PSTricks' \pst@object laden wir preview im active-Modus, restaurieren aber die standardmäßigen Definitionen von \output und \shipout. \pr@startbox und \pr@endbox dienen hier nur dazu, um \pst@object wirkungslos zu machen und stattdessen die zugehörige Grafik aus der Containerdatei einzuladen. Derzeit werden nur maximal 3 optionale Parameter in runden Klammern am Ende von \pst@object unterstützt, was für viele, aber nicht für alle Fälle ausreichend ist.

```

204 \newtoks\ppf@output
205 \ppf@output\expandafter{\the\output}%
206 \let\ppf@nofiles=\nofiles \let\nofiles=\relax
207 \let\ppf@shipout=\shipout
208 \RequirePackage[active]{preview}[2005/01/29]%
209 \let\shipout=\ppf@shipout \let\ppf@shipout=\relax
210 \let\nofiles=\ppf@nofiles \let\ppf@nofiles=\relax
211 \output\expandafter{\the\ppf@output} \ppf@output}%

```

\pr@startbox, \pr@endbox: Gegenüber Originaldefinition vereinfacht.

```

212 \long\def\pr@startbox#1#2{%
213   \ifpr@outer
214     \toks@{#2}%
215     \edef\pr@cleanup{\the\toks@}%
216     \setbox\@tempboxa\vbox\bgroup
217     \everydisplay{}%
218     \pr@outerfalse%
219     \expandafter\@firstofone
220   \else
221     \expandafter\@gobble
222   \fi{#1}}%
223 \def\pr@endbox{%
224   \egroup
225   \setbox\@tempboxa\box\voidb@x
226   \ppf@@getpicture
227   \pr@cleanup}%

```

(Siehe auch identische Definition im DVI-Modus.)

```

228 \AtBeginDocument{%
229   \ifundefined{pst@object}{}%
230   {%
231     \PreviewMacro[{}*[]%
232     ?\bgroup{##1}{##1}}{}%
233     ?\bgroup{##1}{##1}}{}%
234     ?({##1}){({##1})}}{}%
235     ?({##1}){({##1})}}{}%
236     ?({##1}){({##1})}}{}%
237     }]\pst@object}}%
238   }%
239 }%

```

Es werden auch die im DVI-Modus erlaubten Endungen von Grafikdateien benötigt.

```

240 \begingroup
241   \input{dvips.def}%
242   \edef\x{\endgroup\def\noexpand\ppf@other@extensions{\Gin@extensions}}%
243   \x

```

Dummy-Definition für die im DVI-Modus gültigen Dateitypen.

```

244 \DeclareGraphicsRule{*}{eps}{*}{}%
245 \define@key{Gin}{innerframe}[true]{%
246   \lowercase{\Gin@boolkey{#1}}{innerframe}}%
247 \define@key{Gin}{frame}[true]{%
248   \lowercase{\Gin@boolkey{#1}}{frame}}%
249 \define@key{Gin}{ignore}[true]{%

```

```

250 \lowercase{\Gin@boolkey{#1}}{ignore}}%
251 \define@key{Gin}{frame@}{%
    (Nur intern zu benutzen!)
252 \edef\@tempa{\toks@{\noexpand\frame{the\toks@}}}%
253 \ifcase#1\relax
254 \ifGin@innerframe\else\let\@tempa\relax\fi
255 \or
256 \ifGin@frame\else\let\@tempa\relax\fi
257 \fi
258 \@tempa
259 }%
260 \define@key{Gin}{showname}[true]{%
261 \lowercase{\Gin@boolkey{#1}}{showname}}%
262 \define@key{Gin}{namefont}{%
263 \begingroup
264 \@temptokena\expandafter{\ppf@namefont#1}%
265 \edef\x{\endgroup\def\noexpand\ppf@namefont{the\@temptokena}}%
266 \x
267 }%
268 \newcommand*\ppf@filename{%
269 \newcommand*\ppf@namefont{\tiny\ttfamily}%
270 \newcommand*\ppf@Gin@keys{%
271 \let\ppf@Gin@setfile\Gin@setfile

```

\Gin@setfile Realen Dateinamen und ggf. Seitenzahl zur späteren Verwendung merken.

```

272 \def\Gin@setfile#1#2#3{\ppf@Gin@setfile{#1}{#2}{#3}%
273 \xdef\ppf@filename{%
274 #3\ifx\GPT@page\empty\else(\GPT@page)\fi}}%

```

\Gin@ii Auswertung der Optionen “frame”, “ignore” usw. sowie weiterer Spezialfälle.

```

275 \def\Gin@ii[#1]#2{%
276 \begingroup

```

Der Wert \ifGin@innerframe muss bereits vor Zeichnen des inneren Rahmens bekannt sein. Die Werte für \ifGin@showname und \ppf@namefont müssen auch nach Darstellung der Grafik verfügbar sein. Daher durch eine Gruppe geschützt vorher Auswertung der Optionen.

```

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

```

Leerer Dateiname beim Aufruf von \usepicture aus.

```

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

```

Grafiken aus Containerdatei sind bereits skaliert usw. Nicht noch einmal, daher optionalen Parameter ignorieren.

```

281 {%
282 \ifx\@tempb\PDFcontainer
283 \@temptokena{page=\GPT@page}%
284 \fi
285 }%
286 {%
287 \refstepcounter{pspicture}%
288 \@temptokena{page=the\c@pspicture}\def\@tempb{\PDFcontainer}%

```

```

289     }%
290     \fi
291     \ifGin@ignore\else
        “frame@@=0” = innerer Rahmen, “frame@@=1” = äußerer Rahmen.
292     \edef\@tempa{\noexpand\ppf@Gin@ii[frame@@=0,\the\@temptokena,
293         frame@@=1]{\@tempb}}%
294     \@tempa
295     \ifGin@showname
296     \ppf@namefont
297     \raisebox{-\ht\strutbox}[Opt][Opt]{\llap{\ppf@filename}}%
298     \gdef\ppf@filename{}%
299     \fi
300     \fi
301     \endgroup
302 }%

303 \IfFileExists{\PDFcontainer}%
304 {%

```

\ppf@container@max Die Anzahl der in der Containerdatei enthaltenen Seiten.

```

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

307     \AtEndDocument{%
308         \ifnum\c@pspicture>\z@

```

Warnung ist nur sinnvoll, wenn überhaupt Grafiken benötigt wurden.

```

309         \ifnum\c@pspicture=\ppf@container@max\else
310         \PackageWarningNoLine{pst-pdf}{%
311             ‘\PDFcontainer’ contains \ppf@container@max\space pages
312             \MessageBreak but \the\c@pspicture\space pages are requested:
313             \MessageBreak File ‘\PDFcontainer’ is no more valid!
314             \MessageBreak Recreate it
315         }%
316         \fi
317     \fi
318 }%
319 }%
320 {%
321     \def\ppf@container@max{0}%
322     \AtEndDocument{%
323         \ifnum\c@pspicture>\z@
324         \filename@parse{\PDFcontainer}%
325         \PackageWarningNoLine{pst-pdf}{%
326             File ‘\PDFcontainer’ not found.\MessageBreak
327             Use the following commands to create it:\MessageBreak
328             -----
329             \MessageBreak
330             latex \jobname.tex\MessageBreak
331             dvips -o \filename@base.ps \jobname.dvi\MessageBreak
332             ps2pdf \filename@base.ps\MessageBreak
333             -----
334         }%
335         \fi

```

```

336     }%
337 }%

\ppf@isnum Ist Parameter #1 numerisch, werden Anweisungen in #2 sonst die in #3 ausgeführt
(siehe bibtopic.sty).
338 \newcommand\ppf@isnum[1]{%
339 \if!\ifnum9<1#1!\else_\fi\expandafter\@firstoftwo
340 \else\expandafter\@secondoftwo\fi}%

psmatrix Beide Umgebungen ignorieren ihren Inhalt und laden stattdessen die zugehöri-
pspicture ge Grafik aus der Containerdatei. Auf den Wert des dabei benutzten Zählers
(pspicture) kann per \label/\ref zugegriffen werden.

postscript
341 \newcommand*\ppf@set@mode{%
342 \newcommand*\ppf@test@mmode{%
343 \ifmmode
344 \ifinner
345 \let\ppf@set@mode=%
346 \else
347 \def\ppf@set@mode{${}}%
348 \fi
349 \else
350 \let\ppf@set@mode=\@empty
351 \fi
352 }

353 \RequirePackage{environ}%
354 \newenvironment{postscript}[1][]{%
355 \def\@tempa{postscript}%
356 \ifx\@tempa\@currenvir
357 \def\ppf@Gin@keys{#1}%
358 \else
359 \def\ppf@Gin@keys{}%
360 \fi
361 \ppf@@getpicture
362 \Collect@Body\@gobble}{}%
363 \AtBeginDocument{%
364 \@ifundefined{PSTricksLoaded}{}%
365 \def\pst@@picture[#1](#2,#3)(#4,#5){\postscript}%
366 \def\endpspicture{\endpostscript\endgroup}%
367 \@ifundefined{psmatrix}{}%
368 \let\psmatrix=\postscript
369 \let\endpsmatrix=\endpostscript}%
370 }%
371 \@ifundefined{pfx@includegraphicx}{}%

Die im pdfTeX-Modus unnütze Umdefinition von \includegraphics (Paket
psfragx) führt zu zweifachem Einfügen des Ergebnisses, weshalb die Originalde-
finition wiederhergestellt wird.
372 \let\includegraphics=\pfx@includegraphics
373 \def\pfx@includegraphicx#1#2{\ppf@@getpicture}%
374 }%
375 }%

```

`\savepicture` Speichert die Nummer der aktuellen Grafik in einem Makro mit Namen `\ppf@@@#1`.

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

`\usepicture` Fügt Grafik mit symbolischem Namen #2 ein. Der Name muss vorher mit `\savepicture{(Name)}` vereinbart worden sein. Statt des Namens kann auch eine Zahl angegeben werden, die dann direkt eine Grafik aus der Containerdatei adressiert. Der optionale Parameter #1 entspricht dem bei `\includegraphics`.

```
378 \renewcommand*\usepicture[2][]{%
379 \@ifundefined{ppf@@@#2}%
380 {%
381 \ppf@isnum{#2}%
382 {\ppf@getpicture{#1}{#2}}%
383 {\@latexerror{picture ‘#2’ undefined}\@ehc}%
384 }%
385 {%
386 \begingroup
387 \def\Gininclude@graphics##1{%
388 \xdef\ppf@filename{#2}%
389 \setbox\z@\hbox{\pdfrefximage\@nameuse{ppf@@@#2}}%
390 \Gin@nat@height\ht\z@ \Gin@nat@width\wd\z@
391 \def\Gin@llx{0} \let\Gin@lly\Gin@llx
392 \Gin@defaultbp\Gin@urx{\Gin@nat@width}%
393 \Gin@defaultbp\Gin@ury{\Gin@nat@height}%
394 \Gin@bboxtrue\Gin@viewport@code
395 \Gin@nat@height\Gin@ury bp%
396 \advance\Gin@nat@height-\Gin@lly bp%
397 \Gin@nat@width\Gin@urx bp%
398 \advance\Gin@nat@width-\Gin@llx bp%
399 \Gin@req@sizes
400 \ht\z@\Gin@req@height \wd\z@\Gin@req@width
401 \leavevmode\box\z@}%
402 \define@key{Gin}{type}{}%
403 \includegraphics[scale=1,#1]{}%
404 \endgroup
405 }%}
```

`\ppf@getpicture` Fügt die Seite (Grafik) mit Nummer #2 aus der Containerdatei ein. Parameter #1: Optionen wie bei `\includegraphics`.

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

`\ppf@getpicture` Fügt die nächste Seite (Grafik) aus der Containerdatei ein.

```
416 \newcommand*\ppf@getpicture{%
```

```

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

```

**pst-pdf-defs** Umgebung, die keine eigene Gruppe aufmacht. Innerhalb der Umgebung bekommt das Zeichen & den Kategoriecode „other“. Gedacht für eigene Makrodefinitionen, die z. B. eine psmatrix enthalten.

```

422 \renewenvironment*{pst-pdf-defs}%
423 {%
424 \endgroup
425 % ??? \@currentvline
426 \chardef\ppf@temp=\catcode'\&%
427 \@makeother\&%
428 }{%
429 \catcode'\&=\ppf@temp
430 \begingroup
431 \def\@currentvir{pst-pdf-defs}%
432 }
433 \else

```

### 3.5 Inactiver Modus

Es werden nur die Pakete pstricks und graphicx geladen – keine weitere Einflussnahme. Die Paketoption „inactive“ sowie der V<sub>T</sub><sub>E</sub>X-Compiler erzwingen diesen Modus.

```

434 \PackageInfo{pst-pdf}{MODE: \ppf@TeX@mode\space (inactive mode)}%
435 \newenvironment{postscript}[1][\ignorespaces]{%
436 \let\ppf@is@pdfTeX@graphic\relax
437 \fi
438 \InputIfFileExists{pst-pdf.cfg}{%
439 \PackageInfo{pst-pdf}{Local config file pst-pdf.cfg used}}{%
440 \package}

```

## Change History

v1.0a	General: Initial version. . . . .	1	und 1. Now using of eps graphics directly in pdf <sup>L</sup> A <sub>T</sub> E <sub>X</sub> is possible. (RN) . . . . .	1	
v1.0b	General: Some code and documentation cleaning. (RN) . . . . .	1	v1.0e	postscript: “trim” option added. (RN) . . . . .	7
v1.0c	General: New options “pstricks”, “nopstricks”, “draft” and “final”. (RN) . . . . .	4	v1.0f	General: Config file loading added. (RN) . . . . .	15
v1.0d	General: Redefinition of \includegraphics in modes 0		\savepicture: New macro		
			\savepspicture. (RN) . . . . .	14	
			\usepicture: New macro		

	<code>\usepspicture</code> . Useful for putting a PSTricks graphic in a box or something else. (RN) . . . . .	14		<code>\ppf@is@pdfTeX@graphic</code> . Now pdfTeX graphics are preferred. (RN) . . . . .	5
v1.0g	General: Definition of <code>\PDFcontainer</code> now with <code>\edef</code> . (RN) . . . . .	5	v1.0s	General: Scaling e.g. of PostScript pictures now only in extraction mode. Some code cleaning. (RN) . . . . .	1
	<code>\usepicture</code> : Now <code>\usepspicture</code> does accept a numerical parameter. (RN) . . . . .	14		<code>\Gin@ii</code> : Rewritten. (RN) . . . . .	11
v1.0h	<code>psmatrix</code> : Based no more on the comment environment from the verbatim package. (RN) . . . . .	13	v1.1a	General: Support for the internal PSTricks macro <code>\pst@object</code> . (HjG/RN) . . . . .	8
v1.0i	<code>\ppf@is@pdfTeX@graphic</code> : No more errors for given files without extensions. (RN) . . . . .	5	v1.1b	General: Ignore the call of <code>\nofiles</code> inside of <code>preview</code> . (RN) . . . . .	9
v1.0j	General: Check <code>AtBeginDocument</code> for package ‘ <code>pstricks</code> ’ even if ‘ <code>nopstricks</code> ’ is given. (RN) . . . . .	1		Some code and documentation cleaning. (RN) . . . . .	1
v1.0k	<code>\Gin@setfile</code> : Show also the pagenumber if exists. (RN) . . . . .	11	v1.1c	General: New package option “tightpage” added. (RN) . . . . .	1
	<code>\Gin@include@graphics</code> : Prevent division by zero. (RN) . . . . .	8		Special support for “ <code>tabularx</code> ”. (RN) . . . . .	8
v1.0l	General: Options “framesep”, “framerule”, “linewidth” removed, “fname” and “innerframe” added. (RN) . . . . .	1		Supress handling of pdfLaTeX graphic formats in DVI mode. (RN) . . . . .	6
v1.0m	General: New package option “notightpage” added. (RN) . . . . .	1	v1.1d	<code>postscript</code> : Support for PSTricks environment “ <code>psmatrix</code> ”. (RN) . . . . .	13
v1.0n	General: Changed macro names ( <code>\savepicture</code> and <code>\usepicture</code> ). (RN) . . . . .	1	v1.1e	General: New option “displaymath” (see <code>preview</code> package). (HjG/RN) . . . . .	4
	Some code cleaning. (RN) . . . . .	1	v1.1f	General: Package option “ignore” reimplemented. Now the compilation of the dtx file in LaTeX mode is possible. (RN) . . . . .	4
v1.0o	General: New code for “notightpage”. (RN) . . . . .	7	v1.1g	<code>postscript</code> : “ <code>psmatrix</code> ” environment (preserve math mode). (RN/HjG) . . . . .	13
	Option “fname” renamed to “showname”. (RN) . . . . .	1		<code>pspicture</code> : <code>pspicture</code> environment must still parse its arguments. (RN/HjG) . . . . .	13
v1.0p	General: Some code and documentation cleaning. (RN) . . . . .	1	v1.1h	<code>\Gin@include@graphics</code> : Check if inside of a PS-related environment (correct graphic inclusion). (RN) . . . . .	8
v1.0q	<code>\usepicture</code> : Now <code>\usepspicture</code> works for all kind of graphics. (RN) . . . . .	14	v1.1i	General: <code>\ifpr@outer</code> must be predefined. (HjG/RN) . . . . .	5
v1.0r	<code>\ppf@is@pdfTeX@graphic</code> : Changed <code>\ppf@is@known@graphic</code> to				



Package option “final” also for “graphicx”. (RN) . . . . .	4	v1.1p1	General: <code>\let\output\@gobble</code> before loading of “preview” added. (RN) . . . . .	9
<code>\Ginclude@graphics</code> : Correction of the inside check. (RN/HjG) . . .	8	v1.1q	General: Problem with “tabularx” and “threeparttable” solved. (RN) . . . . .	8
v1.1k		v1.1r	General: Fixed values for <code>\PreviewBbAdjust</code> because <code>\paperwidth</code> is not allways defined (suggested by Will Robertson). . . . .	7
General: New environment <code>pst-pdf-defs</code> : Support for PSTricks environment “psmatrix” inside user definitions. (RN,HjG) . . . . .	1	v1.1s	General: Dummy definition of the page key in DVI mode. . . . .	7
v1.1l		v1.1t	General: Remove the line “ <code>\let\output\@gobble</code> ” because of bad side effects. (RN) .	9
General: Support for the package “psfragx”. (RN) . . . . .	8	<code>postscript</code> : Using <code>environ</code> the environment <code>postscript</code> is now simple and more robust. (RN) . .	13	
v1.1m		v1.1u	General: 1 must be set when loading “pdfTeX.def” in DVI mode. (RN) . . . . .	6
General: Merge english and german version of the documentation. (RN) . . . . .	1			
v1.1n				
General: <code>\nofiles</code> added (suggestion of Torsten Bronger). . . . .	6			
v1.1o				
<code>\Gscale@@box</code> : Disable scaling. (RN) . . . . .	8			
v1.1p				
General: <code>\nofiles</code> makes <code>\makeindex</code> and <code>\makeglossary</code> to <code>\relax</code> . <code>\@empty</code> is better because of later <code>\renewcommand</code> ’s. . .	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>	<b>pst-pdf-defs</b> ...	<b>I</b>
\& ..... 426, 427, 429	..... <i>3</i> , <u>46</u> , <u>422</u>	\if@ppf@pdfTeX@graphic
\@currentenv 51, 356, 431	\everydisplay ..... 217	..... 57, 74
\@currentline .. 48, 425	<b>F</b>	\if@ppf@PST@used ..
\@ehc ..... 383	\filename@base 331, 332	..... 4, 82, 190
\@firstofone ..... 219	\filename@parse ... 324	\if@ppf@tightpage .
\@ifpackageloaded 96, 99	\frame ..... 252	..... 5, 111, 120
\@latex@error ..... 383	<b>G</b>	\ifGin@frame ... 58, 256
\@makeother ..... 427	\Gin@bboxtrue ..... 394	\ifGin@ignore .. 61, 291
\@ppf@PST@usedfalse 10	\Gin@boolkey ..... ..... 246, 248, 250, 261	\ifGin@innerframe .
\@ppf@PST@usedtrue 4, 9	\Gin@defaultbp 392, 393	..... 59, 254
\@ppf@pdfTeX@graphicfalse	\Gin@extensions 55, 71, 92, 179, 242, 280	\ifGin@showname 60, 295
..... 64	\Gin@framefalse ... 58	\ifinner ..... 141, 344
\@ppf@pdfTeX@graphictrue	\Gin@ignorefalse .. 61	\ifmmode ..... 140, 343
..... 70	\Gin@ii ..... 56, <u>275</u>	\ifpr@outer ..... ..... 62, 178, 213, 417
\@ppf@tightpagefalse 16	\Gin@innerframefalse 59	\includegraphics ..
\@ppf@tightpagetrue	\Gin@llx ..... 391, 398	.. <i>3</i> , 372, 403, 412
..... 5, 17	\Gin@lly ..... 391, 396	\includegraphicsx ... <i>3</i>
<b>A</b>	\Gin@nat@height ... ..... 390, 393, 395, 396	<b>J</b>
\AtBeginDvi ... 191, 192	\Gin@nat@width .... ..... 390, 392, 397, 398	\jobname .. 41, 330, 331
<b>C</b>	\Gin@PS@file@header 195	<b>K</b>
\color@to@ps ..... 199	\Gin@PS@literal@header ..... 195	\KV@errx ..... 97
\c@p@picture 288, 308, 309, 312, 323, 420	\Gin@PS@raw ..... 196	<b>L</b>
\catcode ..... 426, 429	\Gin@PS@restored .. 196	\leavevmode ..... 401
\Collect@Body ..... 362	\Gin@req@height ... 400	\long ..... 212
\CurrentOption 12, 15, 19	\Gin@req@sizes .... 399	<b>M</b>
<b>D</b>	\Gin@req@width .... 400	\makeglossary ..... 80
\DeclareGraphicsRule	\Gin@setfile 67, 271, <u>272</u>	\makeindex ..... 80
..... 104, 244	\Gin@shownamefalse . 60	\mbox ..... 175
\define@key 105–110, 245, 247, 249, 251, 260, 262, 402	\Gin@urx ..... 392, 397	<b>N</b>
<b>E</b>	\Gin@ury ..... 393, 395	\newcolumnType .... 167
\endpostscript 366, 369	\Gin@viewport@code . 394	\nofiles .. 79, 206, 210
\endpsmatrix ..... ..... 153, 157, 369	\Gin@vllx ..... 124	<b>O</b>
\endpspicture ..... 366	\Gin@vllly ..... 124	\OptionNotUsed ..... 6
\endtabularx ..... 169	\Gin@vurx ..... 124	\output ..... 205, 211
environments:	\Gin@vury ..... 124	\overfullrule ..... 81
postscript <i>2</i> , <u>117</u> , <u>341</u>	\Gin@vury ..... 124	<b>P</b>
psmatrix .. <u>135</u> , <u>341</u>	\Gin@vury ..... 124	\PassOptionsToPackage
pspicture <i>2</i> , <u>134</u> , <u>341</u>	\Gin@vury ..... 124	..... 12, 15, 19
	\Gin@vury ..... 124	\PDFcontainer ..... 40, 41, 282, 288,

303, 305, 311, 313, 324, 326, 413	\ppf@is@pdfTeX@graphic . 63, 179, 280, 436	\PreviewMacro ..... . 159, 172, 186, 231
\pdflastximage .... 377	\ppf@isnum ... 338, 381	\psmatrix .... 150, 152, 154, 155, 368
\pdflastximagepages 306	\ppf@namefont ..... . 264, 265, 269, 296	psmatrix (environ- ment) ... <u>135</u> , <u>341</u>
\pdfliteral ..... 187	\ppf@nofiles .. 206, 210	pspicture (environ- ment) . 2, <u>134</u> , <u>341</u>
\pdfoutput ..... 26, 90	\ppf@other@extensions ..... 43, 92, 102, 179, 242, 280	pst-pdf-defs (environ- ment) .. 3, <u>46</u> , <u>422</u>
\pdfrefximage ..... 389	\ppf@output 204, 205, 211	\pst@@@picture .... 365
\pdfTeXtext ..... 66, 69	\ppf@pr@psmatrix .. ..... 151, 156	\pst@object ... 165, 237
\pdfTeXversion .... 89	\ppf@PreviewBbAdjust ..... 84, 86, 119	\PSTricksOff ..... 198
\pdfximage ..... 305	\ppf@psmatrix . 150, 154	
\pfx@includegraphics ..... 372	\ppf@RestoreBbAdjust ..... 85, 129	<b>R</b>
\pfx@includegraphicsx ..... 172, 373	\ppf@set@mode ..... . 138, 142, 144, 147, 156, 157, 341, 345, 347, 350	\raisebox ..... 297
\postscript ... 365, 368	\ppf@shipout .. 207, 209	\refstepcounter 287, 418
postscript (environ- ment) . 2, <u>117</u> , <u>341</u>	\ppf@temp ..... . 191, 192, 426, 429	\rule ..... 180
\ppf@getpicture .. . 226, 361, 373, <u>416</u>	\ppf@test@mode ... ..... 139, 154, 342	<b>S</b>
\ppf@container@max . ..... 305, 309, 311, 321, 408	\ppf@TeX@mode . 2, 7, 8, 21, 22, 33, 35, 37, 76, 78, 189, 434	\savepicture . 3, 45, 376
\ppf@draft 3, 13, 14, 412	\pr@cleanup ... 215, 227	\setkeys ..... 122, 277
\ppf@endpsmatrix .. ..... 153, 157	\pr@endbox ..... 223	\shipout ..... 207, 209
\ppf@filename . 268, 273, 297, 298, 388	\pr@outerfalse . 72, 218	\string ..... 152, 155
\ppf@getpicture ... .... 382, <u>406</u> , 419	\pr@startbox ..... 212	\strutbox ..... 297
\ppf@Gin@extensions 55	\PreviewBbAdjust .. . 86, 112, 119, 123	<b>T</b>
\ppf@Gin@ii .... 56, 292	\PreviewEnvironment ..... 130, 134, 137	\tabularx ..... 168
\ppf@Gin@keys . 270, 357, 359, 415, 419		<b>U</b>
\ppf@Gin@setfile .. ..... 271, 272		\usepicture .. 3, 44, <u>378</u>
\ppf@Gin@include@graphics ..... 54, 72, 181, 183, 186		<b>V</b>
		\voidb@x ..... 225
		<b>X</b>
		\XKV@err ..... 100