The pstool package

Zebb Prime & Will Robertson

v0.2 2008/08/03

Contents

Part I
Documentation

1 Processing modes

Default commands:

`\psfig` Process this figure if no PDF, or new EPS;

`\psfig*` Always process this figure; and maybe,

`\psfig!` Never process this figure?

The following package options override the above: `[process=all]`, `[process=none]` (the default is `[process=auto]`).

2 Cropping graphics

Graphics are cropped to the appropriate size with the preview package. Sometimes, however, this will not be good enough when an inserted label protrudes from the natural bounding box of the figure. A good way to solve this problem is to use the pdfcrop program (requires a Perl installation under Windows). This can be activated in pstool with the `[pdfcrop]` package option.

3 Todo

1. Higher commands (`\psfragfig`, `\matlabfig`, `\mathfig`)
2. Automatic detection for PDF older than EPS (Windows; shells other than bash?)
3. Generalise "olding" code for multiple files.
4. mylatex integration

Part II
Implementation

```
\ProvidesPackage{%
      pstool}[2008/08/03 v0.2 Wrapper for processing postscript figures]
```

Initialisations

```
\if@pstool@always@
 \if@pstool@never@
\if@pstool@oldpdf@
\if@pstool@pdfcrop@
\newif\if@pstool@always@
\newif\if@pstool@never@
\newif\if@pstool@oldpdf@
\newif\if@pstool@pdfcrop@
        \pstool@out
\newwrite\pstool@out
```

Package options

```
\RequirePackage{xkeyval}
```

pdfcrop

```
\DeclareOptionX{pdfcrop}{\@pstool@pdfcrop@true}
```

process

```
\define@choicekey *{pstool.sty}{process}[\@tempa\@tempb]{%
      all,none,auto}{%
  \ifcase\@tempb\relax
    \@pstool@always@true
  \or
    \@pstool@never@true
```

```
  \or
  \fi}

\ProcessOptionsX
```

External packages

```
\RequirePackage{%
      catchfile,color,ifpdf,ifplatform,inversepath,graphicx,suffix}
```

These are cute:

\OnlyIfFileExists
\NotIfFileExists

```
\providecommand\OnlyIfFileExists[2]{\IfFileExists{#1}{#2}{}}
\providecommand\NotIfFileExists[2]{\IfFileExists{#1}{}{#2}}
```

Command line abstractions between platforms:

```
\edef\pstool@cmdsep{\ifwindows\string&\else\string;\fi\space}
\edef\pstool@rm{\ifwindows␣del␣\else␣rm␣--␣\fi}
```

\pstool@try@rm

```
\newcommand\pstool@try@rm[1]{%
  \@for\@tempa:=#1\do{%
    \OnlyIfFileExists{\@tempa}{\immediate\write18{%
          \pstool@rm␣"\@tempa"}}}}
```

It is necessary while executing commands on the shell to write the exit status to a temporary file to test for failures in processing.

\pstool@statusfile
\pstool@writestatus

```
\def\pstool@statusfile{status-deleteme.txt}
\def\pstool@writestatus{%
  \ifwindows
    echo␣\pstool@percentsign␣ERRORLEVEL\@percentchar\space␣>␣%
          \pstool@statusfile
  \else
    echo␣$?␣>␣\pstool@statusfile
  \fi}
```

Write more to the file to ensure the buffer is flushed and the file is written to disk properly (allowing it to be read by `\CatchFileEdef`). (Maybe even a `touch` would be enough?)

\pstool@flushstatus

```
\def\pstool@flushstatus{%
  \@tempcnta=0
  \loop
    \advance\@tempcnta␣by␣1
    \immediate\write18{%
```

```
      cd␣"\ip@directpath"\pstool@cmdsep
      echo␣"\@percentchar␣buffer␣text"␣>>␣\pstool@statusfile}%
  \ifnum\@tempcnta<10\repeat}

\pstool@retrievestatus
\def\pstool@retrievestatus#1{%
  \pstool@flushstatus
  \CatchFileEdef{#1}{\ip@directpath\pstool@statusfile}{}%
  \pstool@try@rm{\ip@directpath\pstool@statusfile}% uses \@tempa
        internally
}
```

use `ls` (or `dir`) to detect if the EPS is newer than the PDF:

```
\pstool@datefiles
\def\pstool@datefiles{%
  \edef\pstool@filenames{\ip@lastelement.eps\space␣%
       \ip@lastelement.pdf\space}%
  \immediate\write18{%
    \ifwindows
    \else
      cd␣"\ip@directpath"\pstool@cmdsep
      ls␣-t␣"\ip@lastelement.eps"␣"\ip@lastelement.pdf"␣>␣%
           \pstool@statusfile
    \fi}%
  \pstool@retrievestatus\@tempb
  \ifx\@tempb\pstool@filenames
    \@pstool@oldpdf@true
  \else
    \@pstool@oldpdf@false
  \fi}
```

Generic function to execute a command on the shell and pass its exit status back into LaTeX. Any number of `\pstool@exe` statements can be made consecutively followed by `\pstool@endprocess`, which also takes an argument. If *any* of the shell calls failed, then the execution immediately skips to the end and expands `\pstool@error` instead of the argument to `\pstool@endprocess`.

```
\pstool@exe
\def\pstool@exe#1{%
  \immediate\write18{%
    cd␣"\ip@directpath"\pstool@cmdsep
    #1\pstool@cmdsep
    \pstool@writestatus}%
  \pstool@retrievestatus\@tempb
  \ifnum\@tempb␣>␣0
    \PackageWarning{pstool}{%
         Execution␣failed␣during␣process:^^J␣␣#1^^J}%
    \expandafter\pstool@abort
```

```
  \fi}
```

Edit this definition to print something else when graphic processing fails.

\pstool@error
```
\def\pstool@error#1{\fbox{\color{red}%
      \ttfamily␣An␣error␣occured␣processing␣this␣graphic.}}
```

\pstool@abort
```
\def\pstool@abort#1\pstool@endprocess{\pstool@error}
\let\pstool@endprocess\@firstofone
```

4 Command parsing

User input is \psfig (with optional * or ! suffix) which turns into one of the following three macros depending on the mode.

\pstool@alwaysprocess
```
\newcommand\pstool@alwaysprocess[3][]{%
  \inversepath*{#2}% calculate filename, path & inverse path
  \pstool@process[#1]{#2}{#3}}
```

\pstool@neverprocess
```
\newcommand\pstool@neverprocess[3][]{%
  \includegraphics[#1]{#2}}
```

For regular operation, which processes the figure only if the command is starred, or the PDF doesn't exist.

\pstool@maybeprocess
```
\newcommand\pstool@maybeprocess[3][]{%
  \inversepath*{#2}% calculate filename, path & inverse path
  \IfFileExists{#2.pdf}{%
    \pstool@datefiles
      \if@pstool@oldpdf@\expandafter\@firstoftwo
        \else\expandafter\@secondoftwo\fi{%
          \pstool@process[#1]{#2}{#3}%
        }{%
          \includegraphics[#1]{#2}}%
  }{%
    \pstool@process[#1]{#2}{#3}%
  }}
```

4.1 User commands

Finally, define \psfig as appropriate for the mode:

```
\ifpdf
  \if@pstool@always@
    \let\psfig\pstool@alwaysprocess
```

```
\psfig
\psfig*
      \WithSuffix\def\psfig!{\pstool@alwaysprocess}
      \WithSuffix\def\psfig*{\pstool@alwaysprocess}
    \else\if@pstool@never@
      \let\psfig\pstool@neverprocess
\psfig
      \WithSuffix\def\psfig!{\pstool@neverprocess}
\psfig*
      \WithSuffix\def\psfig*{\pstool@neverprocess}
    \else
      \let\psfig\pstool@maybeprocess
\psfig
      \WithSuffix\def\psfig!{\pstool@neverprocess}
\psfig*
      \WithSuffix\def\psfig*{\pstool@alwaysprocess}
    \fi\fi
  \else
    \let\psfig\pstool@neverprocess
\psfig
    \WithSuffix\def\psfig!{\pstool@neverprocess}
\psfig*
    \WithSuffix\def\psfig*{\pstool@neverprocess}
  \fi
```

5 The figure processing

```
\pstool@process
\newcommand{\pstool@process}[3][]{%
  \pstool@write@processfile{#1}{#2}{#3}%
  \pstool@exe{latex␣-shell-escape␣-output-format=dvi
      -interaction=batchmode␣"\ip@lastelement-process.tex"}%
  \pstool@exe{dvips␣"\ip@lastelement-process.dvi"}%
  \if@pstool@pdfcrop@
    \pstool@exe{ps2pdf␣"\ip@lastelement-process.ps"␣"%
          \ip@lastelement-process.pdf"}%
    \pstool@exe{pdfcrop␣"\ip@lastelement-process.pdf"␣"%
          \ip@lastelement.pdf"}%
  \else
    \pstool@exe{ps2pdf␣"\ip@lastelement-process.ps"␣"%
          \ip@lastelement.pdf"}%
  \fi
  \pstool@endprocess{\includegraphics[#1]{#2}}}
```

The file that is written for processing is set up to read the preamble of the original document and set the graphic on an empty page (cropping to size is done either here with preview or later with pdfcrop).

```
ostool@write@processfile
\def\pstool@write@processfile#1#2#3{%
    \immediate\openout\pstool@out␣#2-process.tex\relax
    \immediate\write\pstool@out{%
      \unexpanded{%
        \pdfoutput=0% force DVI mode if not already
```

Input the main document; redefine the document environment so only the preamble is read:

```
\document
```

```
125           \let\origdocument\document
126           \def\document{\endgroup\endinput}%
127         }\noexpand
128           \input{\ip@inversepath\jobname}%
```

Now the preamble of the process file: (restoring document's original meaning)

```
129         \if@pstool@pdfcrop@\else
130           \noexpand\usepackage[active,tightpage]{preview}
131         \fi
132         \unexpanded{%
133           \let\document\origdocument
134           \pagestyle{empty}% remove the page number
135           \begin{document}
```

And the document body to place the graphic on a page of its own:

```
136           \centering\null\vfill}%
137         \if@pstool@pdfcrop@\else
138           \noexpand\begin{preview}%
139         \fi
140           \unexpanded{#3}% this is the "psfrag" material
141           \noexpand\includegraphics[#1]{\ip@lastelement}%
142         \if@pstool@pdfcrop@\else
143           \noexpand\end{preview}%
144         \fi
145         \unexpanded{%
146           \vfill\end{document}}%
147         }%
148       \immediate\closeout\pstool@out}
```

⟨*eof*⟩