ltximg
export tikz|pstricks environments to image format
v. 1.0

Pablo González Luengo
pablgonz at yahoo dot com

December 4, 2013

Abstract

ltximg[1] is a Perl script that automates the process to export tikzpicture or pspicture environments to image formats (PDF, EPS, PPM, PNG).

Contents

1 Required Software

For the full operation of the script you need the following opensource programs (available for windows and linux), external to CTAN repositories.

- Perl.
- Ghostscript.
- pdftops (optional, for images in EPS format).
- pdftoppm (optional, for images in PPM format).
- ImageMagick (optional, for conversion images).

[1]Thanks to Giuseppe Matarazzo for his kind help on testing the script.

2 Run and options

For TEXLive or MikTEX users the syntax for ltximg script is simple:

```
perl ltximg file.tex -options
```

Table 1: Options for ltximg

name	short	default	*description*
`--help`	`--h`		display help information and exit.
`--version`	`--v`		display version information and exit.
`--license`	`--li`		display license information and exit.
`--imageDir=`		`images`	The dir for the created images.
`--DPI=`	`--d`	`300`	Dots per inch for gs, pdftoppm and mogrify.
`--IMO="..."`			Aditional options for mogrify (need double quotes).
`--clear`	`--c`		Delete all temp files.
`--xetex`	`--xe`		Create all image using xelatex (tikz and pstricks).
`--luatex`	`--lu`		Create all image using lualatex (tikz).
`--latex`	`--la`		Create all image using latex(pstricks).
`--useppm`	`--up`		Create jpg and png using mogrify and ppm
`--usemog`	`--um`		Create jpg and png (transparent) using mogrify and pdf
`--margins=`	`--m`	`0`	Margins for pdfcrop.
`--pdf`			Create .pdf files using gs.
`--ppm`			Create .ppm files (need pdftoppm).
`--eps`			Create .eps files (need pdftops).
`--jpg`			Create .jpg files (deafult use gs).
`--png`			Create .png files (deafult use gs).
`--skip=`	`--s`	`skip`	Name for skip environmet in input file.
`--other=`	`--o`	`other`	Name for other export environmet.
`--all`	`--a`		Create pdf/jpg/png/eps image type.

3 How it works

The script works in two steps, but giving the same result, a new file *without tikzpicture* environments and a folder with the images from these environments.

3.1 Comment and ignore

The first step ltximg script create a image dir calls images and create a copy for in file, processing is as follows, being assumed that our file is `test.tex`:

1. Create a copy file called test-tmp.tex and put the problematic environments (verbatim, verbatim, lstlisting, LTXexample, Verbatim, comment, alltt, minted, tcblisting, xcomment and skip) inside the:

   ```
   \begin{nopreview}
   ...
   \end{nopreview}
   ```

   and:

(a) If the option is latex adds the following lines to the beginning of the test-fig.tex:

```
\AtBeginDocument{
\RequirePackage[active,tightpage]{preview}
\PreviewEnvironment{pspicture}
\PreviewEnvironment{other}}
```

(b) If options its xetex adds the following lines to the beginning of the test-fig.tex:

```
\AtBeginDocument{
\RequirePackage[xetex,active,tightpage]{preview}
\PreviewEnvironment{tikzpicture}
\PreviewEnvironment{pspicture}
\PreviewEnvironment{other}}
```

(c) And if no option is given, adds the following lines at the beginning of the test-fig.tex. This is the default for lualatex and pdflatex.

```
\AtBeginDocument{
\RequirePackage[pdftex,active,tightpage]{preview}
\PreviewEnvironment{tikzpicture}
\PreviewEnvironment{other}}
```

2. Open test-tmp.tex and change the problematic words for verbatin in line or after % symbol:

```
\pspicture          => \TRICKS
\endpspicture       => \ENDTRICKS
\begin{pspicture    => \begin{TRICKS
\end{pspicture      => \end{TRICKS
\begin{postscript} => \begin{POSTRICKS}
\end{postscript}    => \end{POSTRICKS}
\begin{tikzpicture => \begin{TIKZPICTURE
\end{tikzpicture    => \end{TIKZPICTURE
\begin{other        => \begin{OTHER
\end{other          => \end{OTHER
```

and save file called test-fig.tex then runs (pdf/lua/xe)latex in `test-fig.tex` and pdfcrop in `test-fig.pdf`.

3.2 Split and convert

If ltximg called with option -pdf or -eps or -um the file `test-fig.pdf` is splitting in `test-fig-1.pdf, test-fig-2.pdf,...` and puts them into `images` dir. The invoked behind this command is:

```
gs -q -sDEVICE=pdfwrite -dPDFSETTINGS=/prepress -dNOPAUSE -dBATCH -sOutputFile=imageDir/test-fig-%d.pdf \
test-fig.pdf
```

and then processes the remaining options.

For example, if you use the option -pdf -um the command behind this is:

```
mogrify -define png:format=png32 -define png:compression-filter=4 -quality 100 -transparent white \
-density 300 -format png *.pdf
```

And if you use the option -pdf -up the command behind this is:

```
mogrify -quality 100 -define png:format=png32 -define png:compression-filter=4 -density 300 \
-format png *.ppm
```

4 Creating other images format

If you need to create other image formats we first need to generate the PPM format or PDF, then the procedure is simple using the ImageMagick `convert` command, command usage is so:

```
mogrify -format ext *.ppm
```

for TIFF use

```
mogrify -format tiff *.ppm
```