aprendtech.com >> blog >> this post

If you have trouble viewing this, try the pdf of this post. You can download the code used to produce the figures in this post.

Scientific blogging with LyX and eLyXer

In this post, I will describe the methods that I use to create the posts on this blog. My approach is to automate the process as much as is comfortable. This reduces the repetitive work required to put a post online, reduces errors and produces a standard package of files with standard format. The standard format makes it easier for readers to follow the post and allows me to re-use the posts; for example, to create an ebook compendium of the posts. My blog discusses technical topics with a lot of mathematics so the methods that I describe will be geared to my interests but I think the general approaches of automating and creating re-useable content are useful for other subjects. The shell scripts and other code I have written were “hacked” together to do a job and designed only for my use so they lack many niceties. I will present them here as an example you can use to create code for your blog.

Specification

I publish my blog using the wordpress package installed by my website host. The wordpress post entries have an introductory paragraph, which typically does not contain mathematics, and then a link to the html page for the complete post. The requirements will be different if you use a standard blogging site like wordpress.com but I guess you can use a similar approach to the one described here. Put an introductory paragraph in the blog post with a link to the html page. If anyone knows whether this can be done, let me know.

With my setup, each post requires the following files:

1. An html file
2. A pdf file
3. A zip archive containing:
   a) the pdf file
   b) the Matlab code to reproduce the figures and other results
   c) a license to the code
4. Shell scripts to create the files and to ftp the required files to my website
5. A list of the Matlab files required for the zip archive.

To ease the automation, I put all the posts as separate sub-directories under a single main directory. To keep a standardized appearance and format, I created a “template post” with software to create a new sub-directory with the files required for a post. This is done with a shell scripts and a set of template files.

Software tools

The main software tools that I use are the Lyx document processor, the eLyXer LyX to html converter, Bash Unix shell scripts running on Windows 7 under Cygwin. Other tools that I use are 7-zip to create zip archives of the files for each post, lftp to transmit the posts to my website, and the rcs revision control system

LyX

According to their website

> *LyX combines the power and flexibility of TeX/LaTeX with the ease of use of a graphical interface. This results in world-class support for creation of mathematical content (via a fully integrated equation editor) and structured documents like academic articles, theses, and books. In addition, staples of scientific authoring such as reference list and index creation come standard. But you can also use LyX to create a letter or a novel or a theatre play or film script. A broad array of ready, well-designed document layouts are built in.*
>
> *LyX is for people who want their writing to look great, right out of the box. No more endless tinkering with formatting details, "finger painting" font attributes or futzing around with page boundaries. You just write. On screen, LyX looks like any word processor; its printed output — or richly cross-referenced PDF, just as readily produced — looks like nothing else.*

For me, it offers the following advantages:

1. beautifully typeset mathematics with minimum fooling around.
2. output in multiple formats: pdf and html. More formats are possible by exporting TeX and then using LaTeX tools but I do not have experience doing this.
3. supports consolidating sub-documents into a master document by including links in the master document. I use this to create compendiums of my posts.
4. can be run from the command line with shell scripts
5. its file is standard ASCII so I can modify it easily with tools like sed.

I use the same LyX file to create both the pdf and the html file. As noted below, eLyXer likes png image format so I use that when I write the LyX file. Then, in the shell script to create the pdf format output, I use this hack to substitute the eps file for the png

Listing 1: shell script snippet to create pdf with eps graphics

```
#make the pdf
rm $1.pdf
# use the eps graphics files instead of the png for the pdf
sed 's/\.eps/\.eps/g' $1.lyx > $1tmp.lyx
# export the pdf from LyX
/cygdrive/L/Progra~2/LyX20/bin/lyx.exe -e pdf $1tmp.lyx
mv $1tmp.pdf $1.pdf
rm $1tmp.lyx
```

In Listing 1, the variable $1 is the post title. The script first removes the old pdf file if it is present. Then it uses sed to substitute .eps for .eps everywhere in the file creating a temporary LyX file. LyX is then invoked from the command line to create the pdf file and then the script renames the temporary pdf to the post-title.pdf and deletes the temporary LyX file.

I export a figure in Matlab in both eps and png format using the code in Listing 2:

Listing 2: Matlab code to export graphics files

```
%% export the figure as png and eps
figno = sprintf('-f%d',fh);
fname = 'SNRvsProjAngle';
   % export as eps
%exportfig(fh,[dirname fname '.eps'],'width',6,'LockAxes',0,'color','cmyk');
    % export as png
set(fh,'papertype','usletter')
print(figno,'-dpng','-cmyk','-r100',[dirname fname '.eps']) % resolution 100 dpi
```

eLyXer

Although LyX can export html, I prefer to use eLyXer because that guarantees that I use the latest version and it offers the possibility to modify it for my application. eLyXer is a python program that I run from within a shell script. It has many options and I use some of these to put the footnotes at the end of the document and to specify the title of the html page. You can find instructions on the eLyXer website. Listing 3 is the command line code to invoke the eLyXer python program under cygwin.

Listing 3: eLyXer command line code

```
"/cygdrive/c/Program Files (x86)/Lyx20/Python/elyxer.py" --footnotes hover,end --title "$2" $1.
```

Creating the zip archive

I create the archive file using 7-zip with the shell script shown in Listing 4. As an aside, this script uses the bash 'case' command to test that it is invoked with the correct number of arguments. If the arguments are incorrect the script outputs a hint to the user about the correct usage and exits. The script uses a text file with the Matlab files to include that is created with the MakeBlogPostFileLists.m function, which is included with the archive for this post.

Listing 4: Make zip shell script

```
#!/bin/bash
# create zip archive
#
# use 'case' to test for number of arguments
case $# in
1)
echo creating zip file $1.zip
ZIP="/cygdrive/c/Progra~1/7-Zip/7z.exe"
# create zip -- first add the post pdf, license and matlab files
$ZIP  a -tzip "$1".zip ../license.txt "$1".pdf ./code/"$1".m

# add files  from filelist.txt-> made using MakeBlogPostFileLists.m
for FNAME in $(cat filelist.txt)
        do
                echo zipping $FNAME
                $ZIP a $1.zip $FNAME
        done
;;
*)
echo usage: dozip name_root
exit 1
;;
esac
```

ftp files to website

I use lftp to send the files to my website. This uses a command file called ftp2web that I create in several steps using shell script in Listing 5. First it adds the files that are included with every post: the html, pdf, and zip archive. Then, the script scans the LyX file with grep for statements that input a graphics file. The grep output is piped to an Awk command to form the ftp 'mput' statement. The 'grep -A 1' option outputs the first line after the line with a match for 'begin_inset Graphics'. This line has only the LyX statement 'filename' followed by the graphics file name.

Listing 5: Make ftp command file script

```
# transmit to website
   # make ftp command file. start with files always included
   # the html, pdf, and zip archive.
cp ftp2web_root ftp2web
    # add png files to ftp script
# in following grep -A 1 looks for lines containing  'begin_inset␣Graphics␣then␣prints␣that␣lin
#␣then␣filters␣that␣to␣produce␣lines␣of␣mput␣statements␣for␣end␣of␣ftp␣script
grep␣-A␣1␣'begin_inset Graphics' $1.lyx|gawk '/filename/␣{print␣"mput",␣$2}' >> ftp2web
echo quit >> ftp2web
lftp -f ftp2web
```

The overall processing script

The process of making the files and transmitting them to my website is controlled by a single shell script that I call for historical reasons 'ElyxPost', although it does more than just invoke eLyXer. This is shown in Listing 6. I have explained the steps in this code in the previous sections.

Listing 6: overall processing script

```
#!/bin/bash
# create html and pdf for post
# REA 2011–Sep–6 16:20
# make the html
case $# in
2)
# write this out completely so can have html title with more than one word––could not make that
echo $2
"/cygdrive/c/Program Files (x86)/Lyx20/Python/elyxer.py" ––footnotes hover,end ––title "$2" $1.
#make the pdf
rm $1.pdf
# use the eps graphics files instead of the png for the pdf
sed 's/\.eps/\.eps/g' $1.lyx > $1tmp.lyx
/cygdrive/L/Progra~2/LyX20/bin/lyx.exe –e pdf $1tmp.lyx
mv $1tmp.pdf $1.pdf
rm $1tmp.lyx
# create zip file with code and results
./dozip $1
# transmit to website
cp ftp2web_root ftp2web
    # add png files to ftp script
# in following grep –A 1 looks for lines containing  'begin_inset␣Graphics␣then␣prints␣that␣lin
#␣then␣filters␣that␣to␣produce␣lines␣of␣mput␣statements␣for␣end␣of␣ftp␣script
grep␣–A␣1␣'begin_inset Graphics' $1.lyx|gawk '/filename/␣{print␣"mput",␣$2}' >> ftp2web
echo quit >> ftp2web
lftp –f ftp2web
;;
∗)
echo usage: ElyxPost post_title html_title
echo html_title will be displayed at top of browser page
exit 1
;;
esac
```

This file is invoked by the user i.e. me from a script called 'Proc' that was created from the template with the name of the post and other information required for the processing. The 'Proc' file is shown in Listing

Listing 7: Shell script to invoke overall processing

```
./ElyxPost  P37Lyxblog   'Blogging␣with␣Lyx␣and␣Elyxer'
```

cygwin tips

Navigating to the directory for my posts was a major pain before I figured out some workarounds. Here is how I do it. I copy the Windows format path to the directory to the clipboard by, for example, clicking on the path box in Windows explorer and then entering ctrl-C. Unfortunately, the cygwin cd command does not seem to understand the path, which uses a backslash \ instead of the standard Unix forward slash /. However, if you enclose the path in single quotes, cd understands it. To paste text from the clipboard into the cycgwin command line, enter shift-insert on the keyboard.

The template post

I create the files shown in the previous section from templates with a shell script called ’MakeNewFiles’. To create the files for a new blog post, I require three pieces of information

1. The post file name. This is the name of the LYX file and the sub-directory for the files.
2. A link number, which is used to make a link to the blog post
3. A title, which is displayed by the browser when it accesses the post.

I will not go into more details about the template files. Listing 8 shows the shell script commands to make the sub-directory and files for a post.

Listing 8: Script to make new post files from templates

```bash
#!/bin/bash
# MakeNewFiles
# create files for new blog post
# REA 2011-Sep-6 16:36

case $# in
3)
# make directory for new ElyxPost files
mkdir ../$1
# and for the code
mkdir ../$1/code
# copy ElyxPost script
cp ElyxPostTemplate ../$1/ElyxPost
# make new Lyx File
LYXFILE=$1.lyx
echo creating Lyx file
SEDCMD='-e␣s/Postxx/'$1'/g -e s/p=xxx/p='$2'/g'
echo $SEDCMD
sed  $SEDCMD Post.lyx > ../$1/$LYXFILE
# make new zip file --will need to edit to add files to send
cp dozipTemplate ../$1/dozip
# make new ftp2web
sed  $SEDCMD ftp2webTemplate > ../$1/ftp2web_root
# make overall processing script
echo './ElyxPost␣' $1 '␣' "'"$3"'"> ../$1/Proc
# make the code file
echo %  $1.m $3 > ../$1/code/$1.m
echo %  >>  ../$1/code/$1.m
echo %  COPYRIGHT:  © Aprend Technology and Robert E. ALvarez, 2013. All rights reserved. >>
../$1/code/$1.m
echo '␣' >>  ../$1/code/$1.m
echo %%  init vars >>  ../$1/code/$1.m
CODELINE="post_title = '$1';"
echo $CODELINE >>  ../$1/code/$1.m
CODELINE2="dirname = ['G:\Projects\Blog\Posts\' post_title '\'];"
echo $CODELINE2 >>  ../$1/code/$1.m
CODELINE="addpath([dirname '\code\']);"
echo $CODELINE >>  ../$1/code/$1.m

# make the post info file --use comma delim
echo '-information␣for␣␣Post␣' $1>  ../$1/PostInfo.txt
echo "-date,"$(date) >>  ../$1/PostInfo.txt
echo "post_title,"$1>>  ../$1/PostInfo.txt
echo "short_link_number,"$2>>  ../$1/PostInfo.txt
echo "html_title,"$3>>  ../$1/PostInfo.txt
echo "lyx_file_name,"$1.lyx
;;
*)
echo make files for post in new subdir
echo usage: MakeNewFiles post_title short_link_number '"html␣title␣can␣have␣multiple␣words"'
echo put html_title in double quotes
echo e.g. post_title is P15CTprojsim
echo e.g. short_link_number is 144
echo e.g. html_title is "CT projection data simulator"
echo need to create new post on web and get short_link_number before starting here
exit 1
;;
esac
```

Conclusion

Automating the production of the files required for a blog post makes it easier for readers to follow the material, facilitates re-use of the post in other media such as books, and reduces work and errors in producing the post.

—Bob Alvarez

Last edited June 2, 2013

Copyright ©2013 by Robert E. Alvarez
Linking is allowed but reposting or mirroring is expressly forbidden.

References