Pweave Literate Programming

June 3, 2015

Pweave is a python module that is designed for use with literate programming, and has built in capability with matplotlib for creating figures.

Literate programming is a type of programming that focusses on documentation containing code, instead of the typical format where code contains comments occasionally for documentation.

1 Getting Started

In order to use Pweave the module must be installed. This can be done by going to https://pypi.python.org/pypi/Pweave

Once Pweave has been installed, you can begin by writing your code within any text editor and saving the file with a text file extension such as .txt or .Pnw

2 Creating Programs

Pweave has some very simple syntax: Any code chunks must start with a line marked with <<>>= or << *options* >>= and end with line marked with @. For example:

Written by Nolan Driessen.
This is to show that documentation goes above the code chunks that were discussed previously!
<<>>=
```
def simple(a,b):
    if a==b:
        return a*b
```

```
    else:
        return a+b
@
```

Anything written before the first <<>>= or after the closing @ but before the next <<>>= will be the document chunks. This will not be code to be executed but will still appear in the documentation that is generated. Note that anything in these lines will not show up in the generated .py files.

3 Code Chunk Options

Within your <<>>= you can specify many different options for the code chunk you are defining.

- name: If the first option has no name attached it is by default assigned to the name. This can also be done through name = 'Name' or label = 'Name'
- fig: By setting fig = 'False' you are stating whether any figures generated by matplotlib should be included within the file. The default is True.
- include: Similar to fig, if include is set to false figures will be generated but not included giving you more control over where the figures go within the document.
- width: By saying width = '12 cm' for example, you can set the size of any figures to be included. The default size depends on the output format.
- echo: If set to false the code chunk will not be displayed within the document.
- evaluate: If evaluate is set to false, the code chunk will not be executed.
- results: Can be set to 'verbatim' for literal block, or 'hidden' for hidden results.
- caption: A caption for the figure produced within a code chunk. Must be used with fig = true
- term: If term is True the output emulates a terminal session.
- wrap: If true both code and output will have longer lines wrapped to 75 characters. Either can also be specified individually for just that option to be turned on.

This information was all found here: http://mpastell.com/pweave/usage.html#code-chunk-options

4 Weaving Programs

In order to utilize your written text file, open up a python shell and import both the pweave module and os module (os comes with python 2.7). You must then set the path to the program you wish to run using the os.chdir(path) function. Once the path has been set, you can use the pweave.weave(file) function to generate a document containing the output of the text file you generated. Alternatively pweave.tangle(file) will generate a .py file containing the python code which you can then run.

```
>>> import os
>>> import pweave
>>> path = "C:/Users/Nolan/Documents/Work"
>>> os.chdir(path)
>>> pweave.weave('test1.Pnw')
Processing chunk 1 named None from line 9
Pweaved test1.Pnw to test1.rst
>>> pweave.tangle('test1.Pnw')
Tangled code from test1.Pnw to test1.py
>>>
```

Within my section of the repository I have 3 documents uploaded: A file named ma.Pnw which contains the program itself, ma.rst which comes from .weave and finally ma.py from .tangle. The documents are from an example found in the following link and illustrate the use of both weaving and tangling.

http://mpastell.com/pweave/pweave.html

5 LaTeX Documentation

You can also use LaTeX to write your Pweave documents as well. Just write your document like you would any other, when you want to insert some python code just place it within the code chunks defined by <<>>= and @.

In order to weave these files you must use the command line in the format Pweave [options] sourcefile. There is a file called FIR_design_minted.texw within my section of the repository. Using that file as an example, you would run the commands
Pweave -f texminted FIR_design_minted.texw
pdflatex -shell-escape FIR_design_minted.tex

The first command gives a .tex file and the second generates a PDF from that .tex file. Many other examples can be found here: http://mpastell.com/pweave/examples/index.html.