Perl Python & PHP 3 Programming Powerhouses

Coming into this class I had a few pre-conceived notions on some of these languages, having worked a little with Python, and worked and talked with people who know Perl and PHP. I had heard from some Perl haters that it was really cryptic and difficult to read. I had also learned from some Perl programmers that it was powerful and easy to quickly write the code they needed. As I mentioned in my introduction on the discussion board, I was looking for a class to improve my Python skills and this was the only class with Python that UML was offering. What I knew about Python at the beginning of the class was that it seemed fairly intuitive and easy to use. The only thing I knew about PHP, was that it was used for Web Programming. Some of these ideas I had were true, and some were not, but in any case, coming away from this class I have a new respect for all of these languages since they are all very powerful in their own ways.

I am no longer petrified of Perl. Perl was created by Larry Wall in 1987 and is still oversees it’s development. Some of the things I like about Perl are that it’s possible to do a whole lot with a little bit of code as demonstrated in the whirlwind tour example in our textbook “Learning Perl”. It also has a huge collection of modules to add whatever extensions you need. Some of the drawback to this is that the wide selection makes it difficult to choose which one to use, and that some of the shorthand can make the code difficult to read (unless it’s documented well). Another drawback of Perl is that it’s not as platform agnostic as the other two languages. My program ran as strict on the PC, but not on the Mac, I had to remove use strict; for it to run. Also installing the modules on the PC was not so intuitive. My last nitpick about Perl is that variable scalar variable start with $ and list and array variables start with @. However, even with all of the issues, I think writing my program in Perl was the easier than I expected, to my surprise! Probably because Perl has a lot of freedom and also since I was able to find an example of parsing an XML file exactly the way I was envisioning it. Perl uses dynamic typing, so it automatically converts between numbers and strings based on the context in which the variable is being used. Another thing that made it easy to program with Perl was that it’s interpreted, for all intents and purposes. I didn’t need to compile and link it myself. Although it has Object-oriented (OO) features, I did not use it that way. I used it more like a functional language, although some of the modules I used created objects and had methods, which I called, but you don’t need to fully understand OO programming to use it. I did not get to make my program as a web app, with Perl so I cannot say first hand how it is as a web-programming platform. However, I do know that it requires an additional web framework (which there are many to choose from See [Comparison of web application frameworks](http://en.wikipedia.org/wiki/Comparison_of_web_application_frameworks#Perl)) in order to use it for Web Programming. All in all Perl is a very powerful, flexible high-level language capable of handling many programming tasks and not as difficult to use as many people make it out to be. But in my experience, the differences between the Windows fork and the Mac/Linux fork are a mark against it.

I still like Python, and I still want to learn more about it, but I did have some issues converting my moderately complex program from Perl to Python. Mainly because I was trying to use Python the same way I was using Perl, more on that later though. First Python is an interpreted Object-oriented, High level programming language created by Guido Van Rossum in 1991. What I like about Python is its intuitive syntax, readability, access to many modules and strong and dynamic typing. I also like how the commands are terminated with the newline character and blocks are formed by indentation and no $ or @ symbols are needed in front of variable names. (less keyboarding) Also Python seems more platform agnostic than Perl. Installation was painless or already included, and I had no issues between the Windows versions of my programs and Mac/Linux versions. Another thing that’s unique about Python is it’s interactive mode where you can test out commands on the fly or execute a program one line at a time. One last reason I like Python is because it can easily be used to create programs that add additional functionality to, or programs that act as the glue between other applications. Many applications in the creative field like Autodesk’s Maya and The Foundry’s Nuke have Python programming interfaces allowing Python programmers to automate repetitive tasks or pass data between applications. The reason I had difficulty converting my Perl program to Python was because I had a difficult time finding a way to easily parse my XML data file with a couple lines of code like it did with my Perl version of program. Once I remembered that the philosophies between the two languages were different, I gave up trying to make Python act like Perl and programmed the way Python liked it. For instance, I was not able to find an XML parsing module for the Python version of my complex program that worked the same way that the Perl XML module worked. So the fastest fix was to reformat the way the Data was stored in the XML File for Python’s Module. The alternative would have been to spend more time researching 3rd party modules, or writing my own. I also did not get to write my complex Python program as a Web Application, but Python also uses [web frameworks](http://en.wikipedia.org/wiki/Comparison_of_web_application_frameworks#Python) for web programming. Python is another very powerful, flexible, easy to use/read, high-level programming language capable of handling just about any programming task if you don’t mind using a web framework. I know there is much more I need to learn about GUI programming, but from what I’ve seen so far there isn’t much to not like about Python.

PHP = Mind blown! I think my pre-conceived notions of PHP were the most incomplete of the bunch. I was correct in that it’s used for web programming, but it is much deeper than that. Another unexpected tidbit I learned PHP is a general purpose interpreted scripting language that features integrated support for interfacing with web servers and databases. This makes it especially good for creating server-side web applications and dynamic web pages. The installation is pretty involved requiring a web server and a SQL database server all configured to communicate with the PHP interpreter. It’s best to use a MAMP or WAMP or similar bundle to make configuration as painless as possible. I found that using an IDE (like PHPStorm from JetBrains) was also very helpful. PHP is a completely different animal from Perl and Python. Not only is the installation different, but also the whole way of thinking behind building a program is different. When I looked at the way PHP handles variables, the different commands and control structures, it looked very similar to other programming languages. However actually using it to create my complex program as a web application was a completely different experience from creating the program with Perl or Python as a command line tool. The first thing that’s different is that PHP code can get mixed in with HTML, and HTML can be used in PHP output. This added an extra step in making sure my program (and web page) was readable. I ended up using HTML to create a form to get the input from the user and PHP to process the input and output the result to a separate page. PHP also uses loose dynamic typing. Variable don’t need to be declared before you use them. Although it automatically figured out the type of the variables, it sometimes got it wrong and I had to trick it into getting it right, for instance by including in some operation with a literal of the correct type. Although PHP has Object Oriented capabilities, I did not use them, I used it as a procedural language to make my Asian zodiac calculator Server-side web app. All in all PHP is another powerful flexible high-level programming language, capable of handling different programming tasks. However its strengths lie in its ease of integration with html and SQL servers. The drawback is the initial learning curve of how to best combine HTML and PHP code, well worth the trouble for what you get in return.

In conclusion, Perl, Python and PHP are all powerful languages, but their power lies in different areas. For Perl, its brevity makes it great for writing quick scripts for processing files or text in a Linux environment. Python’s ease of use and readability make it a great all around language for creating many kinds of applications, scripts and plug-ins. PHP with its html and SQL integration wins hands down when it comes to server side web applications and dynamic web pages. These three different languages Perl Python and PHP are all very powerful in their respective arenas and I will not hesitate to use any of them when the need arises.