Brian Cullinan

Principles of Languages

Assignment 1 - Critical Language Review

Bjc76

Homework 1 – Review aspects and features of multiple languages to show a greater understanding of the predefined principles of languages.

Anyone who knows me knows that I am not a fan Java, and therefore I will start by criticizing that language. For my first example, Java breaks the orthogonality principle. The exact same job or functionality is controlled by completely different symbols or syntax. My favorite example is in String, Array, and ArrayList. To obtain the length of those three objects each one uses a completely different function call. ArrayList uses the size() function, Array uses getLength(), and string uses the length() function. This makes it particularly difficult on programmers (if you aren’t using eclipse) because the same functionality is controlled by completely different functions.

This also violates regularity because it makes the language harder to implement since the API has to constantly be referenced, and harder to learn and remember. Java has great abstraction support, even the ability to create a class within another class or function even. However there are some classes that are apart of Java that violate the information principle in that it gives the programmer too little information. For example the logic behind any of the swing layout systems, it seems to be completely arbitrary in how it functions, and makes it terribly difficult for the end user programmer.

Java does not allow for jumps and gotos, all movements through code must be done using function calls. I find this annoying because there are instances although rare, when goto calls are essential. For example in cases where a Dialog is presented to the user and unacceptable information is received and the dialog should be redisplayed. In a simple function that opens a dialog and does a short check for validity, it doesn’t make sense to force the programmer into splitting up the code for such a simple function. A goto call could be used instead to step back to displaying the dialog until proper input is received.

Java constantly violates the localized cost principle. Many times objects and classes in the language are used just because it supports 1 function needed to manipulate a piece of data. But at the cost of loading libraries just to perform a single function. Java is very bulky and the framework is not optimized. Another example is the framework is backwards compatible, but at the cost of being very big and very slow. Who still writes applications in java 1.0? I have never used an application from Java 1.0 but it still supports it.

This brings me to my next language, my absolute favorite language ever. VB.Net. It is feature rich, easy to learn, and easy to do complex tasks in minimal constructs. VB.Net is all around in favor of the simplicity principle. For example, in previous versions of the language and any language API calls to the operating system are needed. For performing tasks such as saving information to the file system, or looking up a process ID. VB.Net has abstracted away almost every API call supported by windows. VB.Net gives the programmer the ability to do complex calls that would normally wrap around lines of code, in very simple functions and classes provided by .Net framework. It very much supports the Information hiding principle.

Similar to Java VB.Net supports all types of abstraction, complex object types can be abstracted out of the code by using simple code structures and enumerated types. Ironically that is exactly what they are called in VB.Net, a Struct has the ability to create a basic object based on a simple structure of elements and data types. An Enum in VB.Net allows the programmer to abstract away enumerated types.

VB.Net supports goto’s. It has strong support for control structures. However is does violate the regularity principle. Sadly, there is one case where it is irregular in its scope of variables. When a variable in declared in a ‘For’ statement that variable can be accessed after the loop. Microsoft has claimed it is a feature, the reason being if you break out of a ‘for’ loop you can detect at which point it was broken out of. I believe it is a feature, but it is still irregular.

VB.Net meets the localized cost principle. Not only has it abstracted away the different versions of the framework, but on the built in libraries side there is dedicated functionality for everything. For example, if I needed to manipulate a Char array as if it were a String, .Net makes this very easy. I can use a static class that supports more functions than just an array, but I don’t actually have to convert it into a string (using more resources) to do it.

PHP is also a wonderful language. It meets the simplicity principle, but up until recent versions, classes were not even supported. This breaks the information hiding principle. And still there is hardly any support for scoping; private and public functions are completely arbitrary. There are certainly conventions that some libraries use, but it makes it difficult on the programmer not knowing what should be accessed and what shouldn’t.

PHP also breaks the regularity principle. Inversely to Java having too much cascading in function calls, PHP has no cascading of function calls. There are no classes that contain functions, all the built in functionality has a global scope. It breaks the control principle in that the structure can be linear or completely obfuscated. PHP executes code outside of functions, in this sort of global object, and the placement or organization of code can be completely unrelated to the order in which it is executed. Function calls can exist anywhere in the code, but the actual function could be in completely random locations. This causes a lot of horrible code, and a lot of horrible programmers and often times make it tedious to understand other people’s code.

PHP does support the abstraction principle. Even with the limited support for classes, procedures and recurring patterns can be factored out and organized. There are rare incidents where the label and names of procedures are unrelated to its functionality.

PHP is incredibly efficient in every way. Even the most complex scripts and algorithms can be executed at incredibly fast rates. Because it is a scripting language features of the language are easily recognized. Writing programs comes quickly because many of the features are easily referenced because of the lack of cascading classes and namespaces.