Lesson 03 – Discussion

**Variables and Scope**

How do rules of scope differ in Python (as opposed to other programming languages)?

<http://python-textbok.readthedocs.io/en/latest/Variables_and_Scope.html>

When defined in the main body of the file, it is considered a Global Variable. Variables with a global scope are used rarely and when they are used, it should be with caution. This is because they are known to have wide ranging, negative effects that can have some unwanted problems. In Python it is actually possible to change or create a global variable in a local context with the keyword ‘global’. But, as stated before, global variables are seldom considered to be of good programming practice. In contrast to variables within the global scope, there is also a local scope. Variables defined inside of a function or a class, for example, are local to that function or class. It is often said that when this function or class ends, the variable declared within it then “dies” because it is now “out of scope”. Pythons rules of scope differ slightly from other languages. For example, in Java, a variable is scoped inside its block of curly brackets, But in Python, a variable is scoped to the innermost function – not including control blocks. I found a short and sweet Stack Overflow post visualizing this concept if anyone is interested.

<https://stackoverflow.com/questions/30727711/local-variable-scope-in-java-vs-python>

In statically typed languages, variables have predetermined types, and a variable can only be used to hold values of that type. In Python, we may reuse the same variable to store values of any type.

Not all variables are accessible from all parts of our program, and not all variables exist for the same amount of time. Where a variable is accessible and how long it exists depend on how it is defined. We call the part of a program where a variable is accessible its scope, and the duration for which the variable exists its lifetime.