iThink360's Java Programming Tutorial Series

Modifiers and Scope Cheat Sheet

Scope: Refers to the access level of attributes and methods, including constructors. Can be altered via access modifiers.

Sequencing Scope: What I like to coin the concept of variable inaccessibility prior to its existence. Simply put, variables cannot be accessed prior to their declaration within a sequencing algorithm (i.e. within a method/constructor call).

Block Scope: Refers to the variable accessibility constraint when dealing with code blocks. A code block is a pair of curly braces that contain sequencing code statements. This includes loops, methods/constructors, if-else statements, switch statements, and more. Any variables declared within a block of code may NOT be accessed outside that block. *Note that any variables declared within a signature line, i.e. for loop iterators and method/constructor parameters, count within the scope of the respective block.*

Modifiers: Used to alter the behavior of a variable, method, constructor, or class. There are 2 types of modifiers: access and non access.

Access Modifiers: Alter the access level (in a nutshell, scope) of a variable, method, constructor, or class. Note that some modifiers are not allowed to be used in certain circumstances (for example, a class declared at the root of a .java file cannot be private or protected, but a nested class can be). There are 4 access modifiers:

- *public:* The respective data can be accessed from anywhere within the project.
- *default (no access modifier specified):* The respective data can be accessed from anywhere within the same package as the respective class. (We haven't learned about packages yet.)
- *protected:* The respective data can only be accessed from within the same class and from subclasses. (We will cover this in Module 3.)
- *private:* The respective data can only be accessed from within the same class.

Non-Access Modifiers: Alter other aspects of a variable, method, constructor, or class. Once again, some modifiers may not be allowed in certain circumstances. There are a total of 6 non-access modifiers:

- *final:* Attributes and methods cannot be modified/overridden, and classes cannot be extended
- *static:* Attributes and methods can be accessed directly via the class rather than through an object
- *abstract:* Declares a class as abstract. An abstract method must then be declared within the class (no method body attached). A subclass overrides the abstract method and implements the body. (We will cover this in Module 3.)
- *transient:** Attributes and methods are skipped when serializing the object containing them
- *synchronized:** Methods can only be accessed one thread at a time
- *volatile:** The value of an attribute is not cached thread-locally, and is always read from the "main memory"

** Nor do I have any clue what these modifiers are for nor do I know how they work. We might never cover these in our tutorial series*