Head First Java: Chapter 1 Notes

August 27, 2019

The way Java works:

1. Source code as `.java` file;
2. Compiled using `javac` to `.class` file;
3. Bytecode files (`.class`) can be read platform-independently by *Java Virtual Machines (JVM)*;
4. Application is manifested by *JVM* and perform desired activities.

Here are some sample java code:

`int size = 27;`	Declare an integer variable named 'size' and give it the value 27
`String name = "Fido";`	Declare a string of characters variable named 'name' and give it the value "Fido"
`Dog myDog = new Dog(name, size);`	Declare a new Dog variable 'myDog' and make the new Dog using 'name' and 'size'
`x = size - 5;`	Subtract 5 from 27 (value of 'size') and assign it to a variable named 'x'
`if (x < 15) myDog.bark(8)`	if x (value of 22) is less than 15, tell the dog to bark 8 times
`while (x > 3) {`	Keep looping as long as x is greater than 3
`  myDog.play();`	Tell the dog to play
`}`	End of the loop; everything in {} is done in the loop
`int [] numList = {2,4,6,8};`	Declare a list of integers variable 'numList', and put 2,4,6,8 into the list
`System.out.print("Hello");`	Print out "Hello" (in this case to the command line)
`System.out.print("Dog:"  + name)`	Print out "Dog: Fido" as above
`String num = "8";`	Declare a character string variable 'num' and give it the value of "8"
`int z = Integer.parseInt(num);`	Convert the string of characters "8" into an actual numeric value 8
`try {`	Try the executing code between {}
`  readTheFile("myFile.txt");`	Read a text file named "myFile.txt"
`}`	End of try-block
`catch(FileNotFoundException ex) {`	Declare exceptions to "catch"
`  System.out.print("File not found.");`	If code within try-block filed due to declared exception, print out "File not found."
`}`	End of exception-block

Code structure in Java

- A `.java` source code file must hold *one* **class** definition.
- A **class** is a piece of a Java program.
- Within a **class** there are one or more **methods**.
- **Methods** must be held within **classes**.
- Within each **methods** are **statements** that describe what each **method** should perform.

Java programs are run by the JVM, and must contain *at lest one class* and one *main method*. The *main method* almost always look exactly like this:

```
public class MyFirstApp{
    public static void main (String[] args) {
        (your code here...)
    }
}
```

Note: The `String[] args` assigns an array of strings to the `main` method, naming the argument `args`. When running a Java program, the JVM searches for the *main* method, runs all code within it before stopping. Code outside of the *main* method must be called within the method. Several things to remember for Java syntax:

- Each statement must end with a ';',
- Single line comments begins with `//`,
- Whitespace generally do not matter,
- Variables must be declare with a type (e.g. `int, double, char,` etc.),
- Classes and methods must be enclosed with {}.

To declare a string array (like a list in python, use:)

```
String[] name = {''String1'', ''String2'', ''String3'';}
```

Some notes about string arrays:

- Arrays use o-index (just like Python),
- Use `.length` method to find length of array,
- Use `name[index]` to access items in array,