JavaScript-to-Java Tips

- Learning new languages is inevitable over the course of a computing career—get used to it :)
- Formally, one approaches a new language from two perspectives: (a) how it looks (its syntax) and (b) what it means (its semantics)
- This handout mixes the two approaches, taking a more case-based approach to how the languages differ

Everything is in a Class

- All Java code must reside in a class
- For now, let's leave it at that—we will get into what classes are later
- You define one class per file, and that file's name should be the name of the class with a .java extension
    - If a class is named `HelloWorld`, then its source code should be in a file name HelloWorld.java
    - Class and file names are case-sensitive

Data Types Rule

- In JavaScript, data does have a type, but variables and functions can change them around dynamically…not so in Java
- All Java entities must commit to a type—i.e., if you declare a variable, you must state the data type that this variable will hold, and you cannot change it later
- This may seem restrictive at first, but in exchange, you can detect some errors more readily

- Data types' most visible effect on code is the way they replace the `var` or `function` keywords in JavaScript; data types take their place:

```
int x = 5;           // An integer variable named x.
String s = "Hello";  // A String variable named s.
boolean done = true; // A boolean variable named done.

// An array of ten Strings.
String[] labels = new String[10];

// A method/function that takes zero parameters and
// returns a string.
public String getName()

// A method/function that takes three integer parameters
// and returns an integer.
public int getMedian(int a, int b, int c)
```

- Don't worry—if you don't get a declaration right, you will know it when you compile with javac

Functions ☞ Methods

- Functions in Java are called methods
- Java methods commit to returning a specific data type, and all of their parameters also have designated types
- Methods that return nothing have a special data type: `void` (e.g., the `main` method)
- Methods have many other characteristics not seen in JavaScript functions—we will reveal those as needed (or you can read ahead)

About `;` `==` and `!=`

- Many symbols play very similar roles in Java as they do in JavaScript: `= ( ) [ ] { } + - / *` …etc.
- Notable exceptions include:
  - ◇ Semicolons—They actually do play a very similar role, but they are more strictly required in Java
  - ◇ Equality and inequality—Comparators === and `!==` are JavaScript-only; in Java stay with == and `!=`

main Marks the Spot

- A Java class can only be "run" (i.e., invoked using the java command) if it contains a method whose declaration is:

```
public static void main(String[] args)
```

- Yes, every word above has a distinct meaning; no, you don't need to know all of them in depth (yet)
- The `args` parameter corresponds to information provided by the user at the command line—yes, Java takes care of gathering that up for you

Loops and Conditionals

- By and large, loops and conditionals look the same in Java as they do in JavaScript: `if` statements, `while` loops, and some `for` loops should be recognizable
- One difference with conditions: Java does not have a concept of "truthiness" or "falsiness"—conditions must have a boolean data type
- One difference with `for` loops: Java has a variant of the `for` loop that is tailored to iterating through collections —you'll know it when you see it