Introduction to Java

What is Java?

- Java is an Object-Oriented Programming Language
  - Programming language − a vocabulary and set of grammatical rules for instructing a computer or computing device to perform specific tasks. The specific tasks are known as a <u>Computer Program</u>.

A computer program

- Is a set of instructions that performs a specific task when executed by a computer.

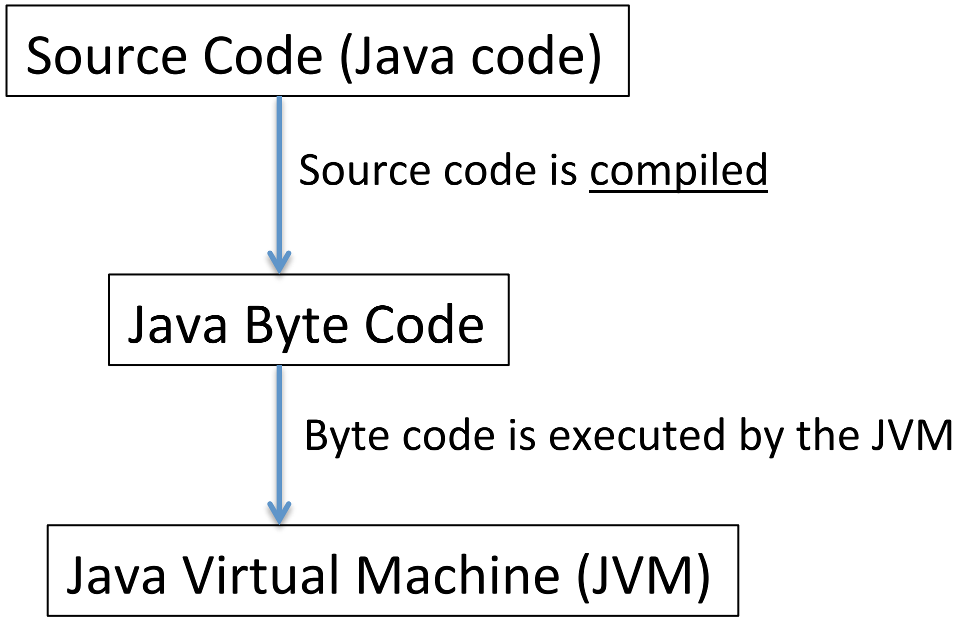
Make and Drink Coffee

```
coffeeMethod = frenchPress;
grindBeans();
boilWater();
pourWaterOverBeans();
wait(5); // wait for 5 minutes
pourCoffeeInCup();
while(coffeeStillInCup()){
    drinkCoffee();
}
```

What can computers understand?

- A computer probably couldn’t follow the instructions we just gave for making coffee.
- Different computers have different basic operations they can perform, like addition, subtraction, print text, open file, etc.
- A compiler converts our programs into the things a computer can understand.

Compiling and Running Programs



Downloading Java

- JDK (Java Development Kit)
  - Allows you to compile and run Java programs
  - Go to http://www.oracle.com/technetwork/java/javase/downloads/index.html for latest JDK

Opening Eclipse

- Eclipse is an <u>IDE</u>
- <u>IDE</u> – Integrated Development Environment – where we write and test code.
- This is where we'll be writing all of our Java programs.
- Download instructions on course webpage

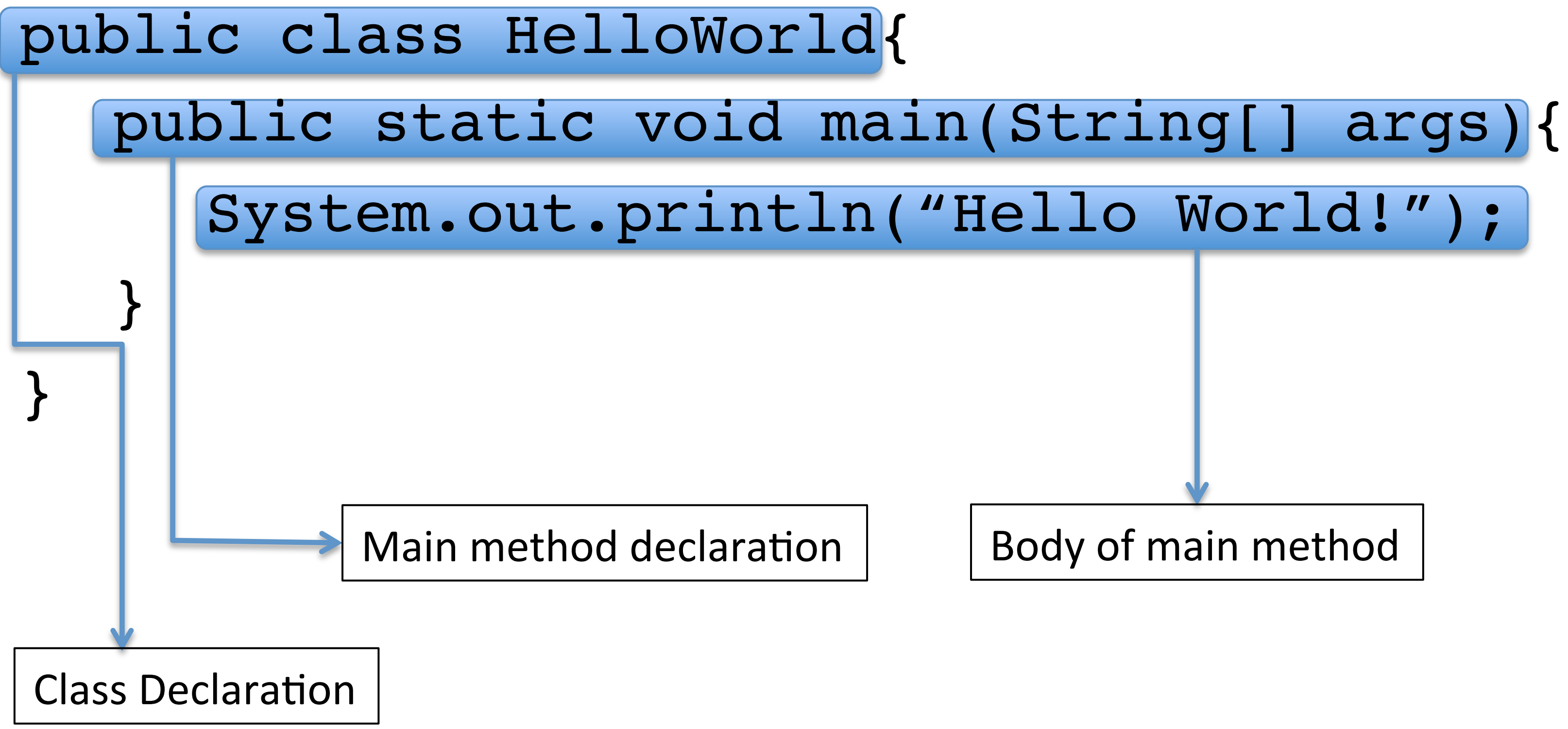
Writing a First Program

```
public class HelloWorld{
   public static void main(String[] args){
      System.out.println(“Hello World!”);
   }
}
```

Output:

```
Hello World!
```

Writing a First Program



```
public class HelloWorld{
   public static void main(String[] args){
      System.out.println(“Hello World!”);
   }
}
```

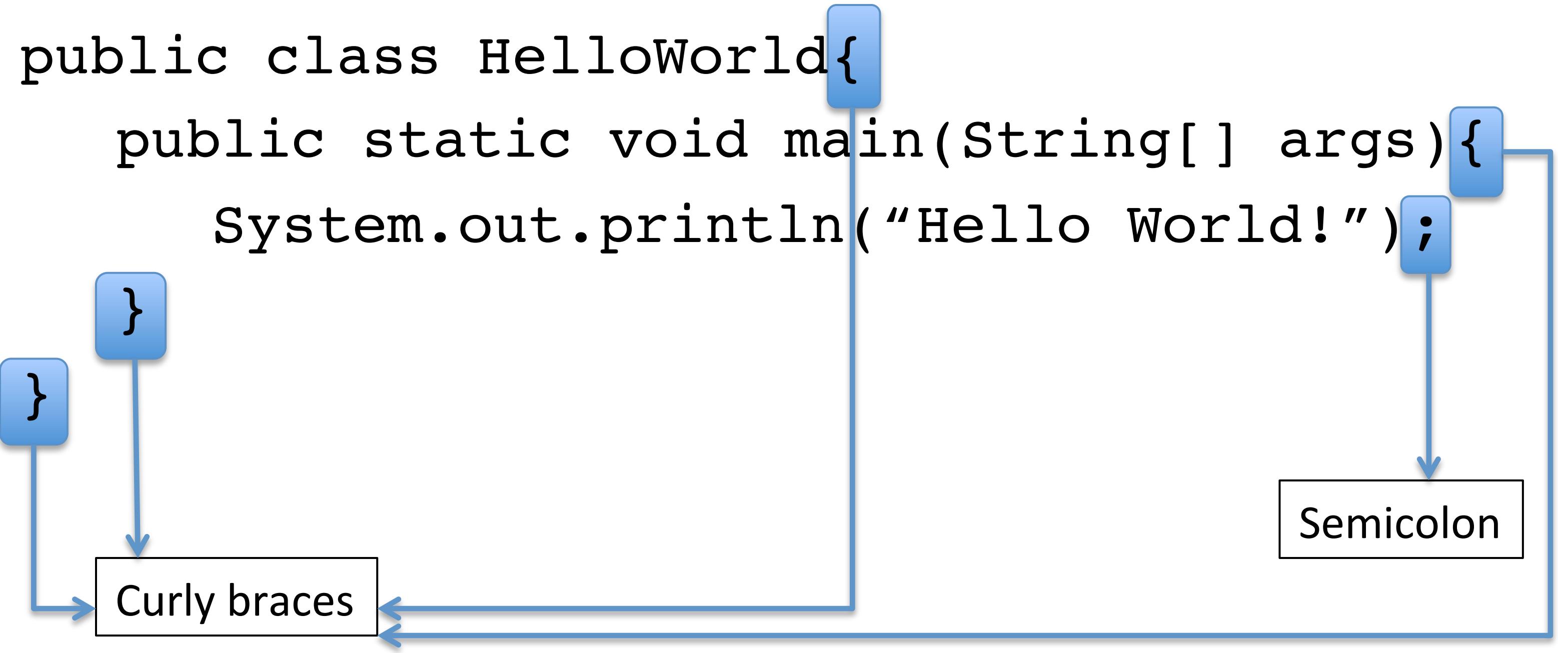
Program Anatomy

- Class declaration
  - `public class HelloWorld`
  - Every java file will need a class declaration. The "public class" part will always need to be the same, but the HelloWorld part will be named by <u>you</u>.
- Main method declaration
  - You will be memorizing this line. It will always be the same.
- Body of main method
  - All the lines of code we're trying to run.

Program Anatomy

- Some other components of our program:
  - Curly Braces − { }
    - Every curly brace that opens, also needs to be closed
    - Curly braces will accompany the class, and methods, (and others we'll get to later)
  - Semi-colon − ;
    - Every line of code must end with a semi-colon

Program Anatomy



```
public class HelloWorld{
   public static void main(String[] args){
      System.out.println("Hello World!");
   }
}
```

Syntax – the set of rules that defines the combinations of symbols that are considered to be a correctly structured document or fragment in that language. **Syntax** refers to the spelling and grammar of a programming language.

Writing a First Program

- `System.out.println();`
  - This line of code is used to write output to the console
  - The “ln” suffix tells us there will be a new line after the output has been printed

```
System.out.println(“First line”);
System.out.println(“Second line”);
System.out.println(“Third line”);
```

Output:

```
First line
Second line
Third line
```

Writing a First Program

- `System.out.print();`
  - Unlike System.out.println(), this line of code will not go to the next line

```
System.out.print(“First”);
System.out.print(“Second”);
System.out.print(“Third”);
```

Output:

```
FirstSecondThird
```

Comments and Whitespace

- A comment is text added to code by a programmer, intended to be read by humans to better understand the code, but ignored by the compiler.
  - Single-line comment: Comment consisting of one line
  - Multi-line comment: Comment consisting of more than one line
- Whitespace is used to put space in between code to make it more readable

Comments

```
public class CommentDemo{
   public static void main(String[] args){
      System.out.println("This program demos two
                              types of comments");
      // This is a single line comment

      /* This comment
         spans multiple
         lines
      */
   }
}
```

Variables

- A variable stores a value; we can change and use the value.
  - For example, we create a variable to store the cost of a computer
    - `int cost = 600;`
    - The variable cost has been <u>declared</u> and <u>initialized</u>
    - The int means this variable can only store integers
  - We change the cost of the computer to 500
    - `cost = 500;`
    - The variable cost has been <u>assigned</u> a new value

Variables

- Variable rules:
  - Must consist of letters (a-z, A-Z, _, $) and digits (0-9)
  - Must start with a letter
  - Can’t be a reserved word
    - Reserved words are listed at the end of section 2.3
  - Make it **<u>descriptive</u>**!

Variables

- Give a descriptive name to your variable.

```
int x = 0;
```

x doesn’t give much description about what type of information is being stored

```
int balance = 0;
```

balance would be more descriptive

Variables

Example Variable Name	Valid?
grade	Yes
grade	Yes
$grade$	Yes (but bad)
=grade	No
grade3	Yes
GRADE	Yes
class	No
Class	Yes (but bad)
a_long_variable_name	Yes
aLongVariableName	Yes
“a long variable name”	No

Primitive Data Types

Data Type	Bits	Description	Example
byte	8	-2^7 to 2^7	238
short	16	-2^{15} to 2^{15}	
int	32	-2^{31} to 2^{31}	
long	64	-2^{63} to 2^{63}	
float	32	coefficient of 23 bits, exponent of 8 bits	3.14159265359
double	64	coefficient of 53 bits, exponent of 11 bits	
char	16	ASCII characters	‘A’, ‘!’, ‘9’
boolean	1	True or false value	True, false

Various Variable Initializations

```
// declare multiple variables
int time, date, year;

// declare and initialize multiple variables
int time=1245, date=1002, year=2017;

// declare some variables, declare and initialize others
int time=1330, date=929, year;
```

String Data Type

- Not quite a primitive data type, but often times used like one.

Constants

```
// constants can’t be changed
final double BOILING_TEMP = 212.0;
final double HUMAN_TEMP = 98.6;
```

Arithmetic Expressions

Arithmetic operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulo

Modulo

- Gives the remainder of division
  - Recall that integer division only results in <u>quotient</u>

4 % 6 = 4

10 % 6 = 4

12 % 6 = 0

7 % 5 = 2

9 % 5 = 4

Type Conversion

- Implicit conversion – happens automatically
  - No loss of precision
- Examples:

int num1	**double num2**	**Expression**	**Result**
5	2.0	num1*num2	10.0
5	2.0	num1/num2	2.5

- As a general rule, lower types are converted to higher types
  - int < long < float < double

Type Casting

- This is forced type conversion

```
double numerator = 5.0;
int denominator = 2;

// the variable expression below evaluates to 2, an int
int expression = (int)numerator/denominator;

//------------------------------------------------------------

int numerator2 = 5;
int denominator2 = 2;

// the variable experssion2 below evaluates to 2.5, a double
double expression2 = (double)numerator2/denominator;
```

Basic Input

```
import java.util.Scanner;
public class ScannerTest {
   public static void main(String[] args){
      Scanner keyboard = new Scanner(System.in);
      System.out.println("Please enter a number
                        between 1 and 10");
      int x = keyboard.nextInt();
      System.out.println(x);
   }
}
```

This program prints a message to the user "Please enter a number between 1 and 10", then takes whatever the user enters and assigns it to the variable x

Conversion Methods with Scanner Class

`next()`	`String`
`nextInt()`	`int`
`nextLong()`	`long`
`nextFloat()`	`float`
`nextDouble()`	`double`
`nextBoolean()`	`boolean`
`nextLine()`	`String`

Getting a single character

- Use charAt method which belongs to the String class

```
String user = “Guest User”;
System.out.println(user.charAt(0));
System.out.println(user.charAt(2));
System.out.println(user.charAt(4));
```

What is a method?

- Method – a named collection of instructions
  - Method calls
  - Method definitions
    - Static methods
      - ClassName.methodName(…arguments…)
    - Instance methods
      - instanceOfClass.methodName(…arguments…)

Math Methods

- No import statement required (java.lang library is included by default)

```
Math.sqrt(x) – evaluates to squareroot of x
Math.pow(x,y) – evaluates to xʸ
Math.round(x) – x rounded to the nearest int
Math.random() – random double between 0 and 1

Math.PI – the value of pi as a double
Math.E – the value of e as a double
```

Errors and warnings

- Code needs to be written *perfectly*, otherwise you will get a <u>syntax error</u>
  - Some Examples:
    - Forgetting semicolon
    - Forgetting curly brace
    - Misspelling reserved words

Errors and warnings

```
// How many errors can you find in this program?

/ This program prints a message about Santa Cruz
public class Errors{
    public static void main(String[] args){
        System.out.print(“Santa Cruz );
        System.out.println(“is the real “)
        System.out.println(“Surf City”);
    }
```

A. 0
B. 1
C. 2
D. 3
E. 4

Running a Program at the Command Line

```
javac HelloWorld.java
java HelloWorld
```

Runs the program by executing the .class file.

Compiles the program and generates a .class file. If you have errors, they will be discovered here.