Java Programming 1

PROFESSOR: CÂI FILIAULT

Topics for this week

- Computers
- Operating Systems
- Programming
- Java
- IDE's
- Hello World
- History

What is a computer?

- A computer is a **general-purpose device**
- Computers are comprised of a series of physical components that are integrated together.
- These components are referred to as **hardware**
- Computer hardware can be instructed to complete tasks through **computer programming**
- When someone writes a computer code, the result is known as a **computer program** or **software**
- The actual lines of instruction that are written are often referred to as **"code"**
- *Think of when someone asks for a hard vs soft copy of something*

The **operating system** is the largest piece of **software** on a computer and is responsible for the following:

- Controlling and monitoring computer hardware
- Allocating and assigning pieces of computer hardware

What is programming?

Programming is the process of writing computer programs or software

Within the world there are hundreds of programming languages that have their own purposes and specialties.

We will be utilizing the Java programming language to build desktop/laptop/server computer programs and mobile applications.

Java is one of the primary language used to develop apps on android devices.

Language	Primary Use
c	The c programming language is primary used for low level operations and developing operating systems
python	A general-purpose programming language used in a variety of fields such as computer science, mathematics
PHP	A server-side programing language used to interact with databases and product dynamic and secure web pages
C++	A general-purpose programming language used to build a variety of different things from desktop applications to games

What is Java?

Java is a general-purpose programming language used on all operating systems including some mobile platforms.

Java was originally invented by a team at Sun Microsystems lead by **James Gosling** where it was originally named **oak** after the tree outside of James office window

It can be used to build:

Type	Example
Terminal	Most of the applications we will be building this semester. *RedAlert, Speedavg*
Desktop	Many of the world's most popular software development programs are written in Java *Eclipse, Intellij*
Web	Java is a capable language in almost all forms of software development and has a large presence in the web. SpringBoot is a common framework used to develop websites and web applications. *Staples, FitBit, AutoTrader, Intuit etc.*
Mobile	Java is one of the leading languages in android mobile application development. Most of the applications you would have installed on your android phone or tablet will have been written in Java.
Games	Java can also be used to write games. The most well-known game to be written in Java is Minecraft. https://libgdx.badlogicgames.com/

What is Java?

Java is a unique language. It is a portable language and is built on the premise “Write once, run anywhere”

- Java like many languages needs to be **compiled** before it can be run. Which means it needs to be sent through a **compiler**.
- Java is what's known as a **high-level programming language** (meaning it is very readable to humans)
- Computers understand very low-level **Machine languages** such as **assembly** and **binary** (not very human readable)
- The purpose of sending your code through a compiler is to convert it from the high-level language that a human can read to the low-level language that a computer can read.
- Unique to Java, the complier translates code into Java **bytecode**.

What is Java?

The JVM:

- Before a computer can run a piece of Java software it needs Java installed.
- The process of installing Java on a computer is to give it a **Java Virtual Machine** (JVM)
- The JVM sits on top of the operating system ready to run Java **bytecode**.
- The JVM will correctly convert the java bytecode into the correct instructions depending on the computer environment it is running on
- Now a programmer can write one piece of code and have it run on any operating system

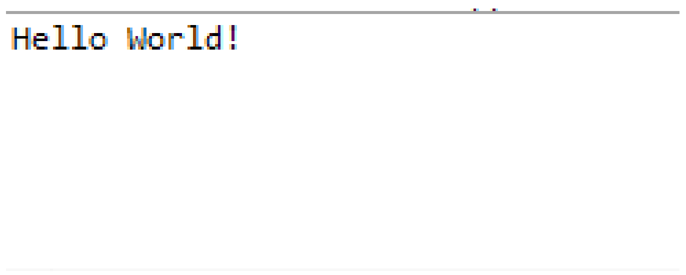
Mac	**Windows**	**Linux**

Hello World

Let's look at the steps needed to build our first java application:

At the end of this mini lecture, you will have a piece of software that looks like the image depicted on the right

It may not look like much but its officially the first program you will write in java



```
Hello World!
```

Step 1: Learn the IDE

An Integrated Development Environment or (IDE) is a software that is used to make programming easy.

Throughout your stay at St. Clair College, you will use many IDE’s:

- Android Studio
- PHPStorm
- Intellij
- Visual Studio

Step 1: Learn the IDE (IntelliJ)

The IDE used within this course is IntelliJ

Features: (More detail will be provided on these features)

- Debugger
- Compiler
- Auto-completion
- Inline error detection
- Easy importing
- Plugin support

Step 1: Learn the IDE (IntelliJ)

Debugger:

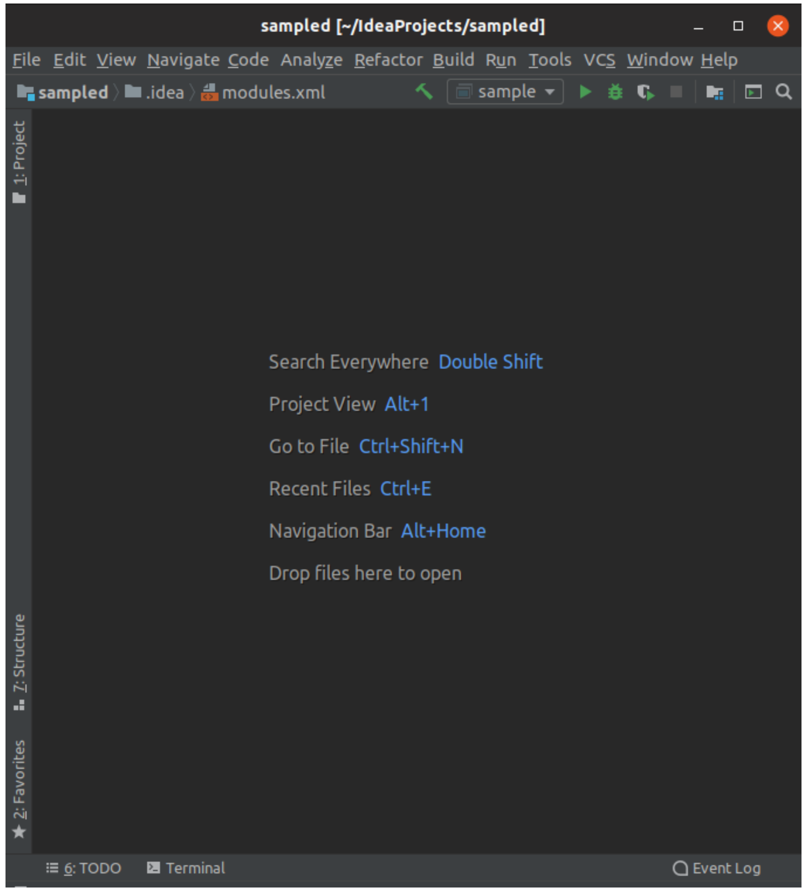
Errors within code are referred to as **bugs**

- A debugger is used to find “bugs” or errors within your code.
- We will go more into debugging in a later lecture

Step 2: Creating a new project

Start by opening IntelliJ and selecting the following:

File> New> Project


sampled [~/IdeaProjects/sampled]
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
sampled .idea modules.xml
sample
1: Project
7: Structure
2: Favorites
6: TODO
Terminal
Event Log

Search Everywhere Double Shift

Project View Alt+1

Go to File Ctrl+Shift+N

Recent Files Ctrl+E

Navigation Bar Alt+Home

Drop files here to open

Step 2: New Project

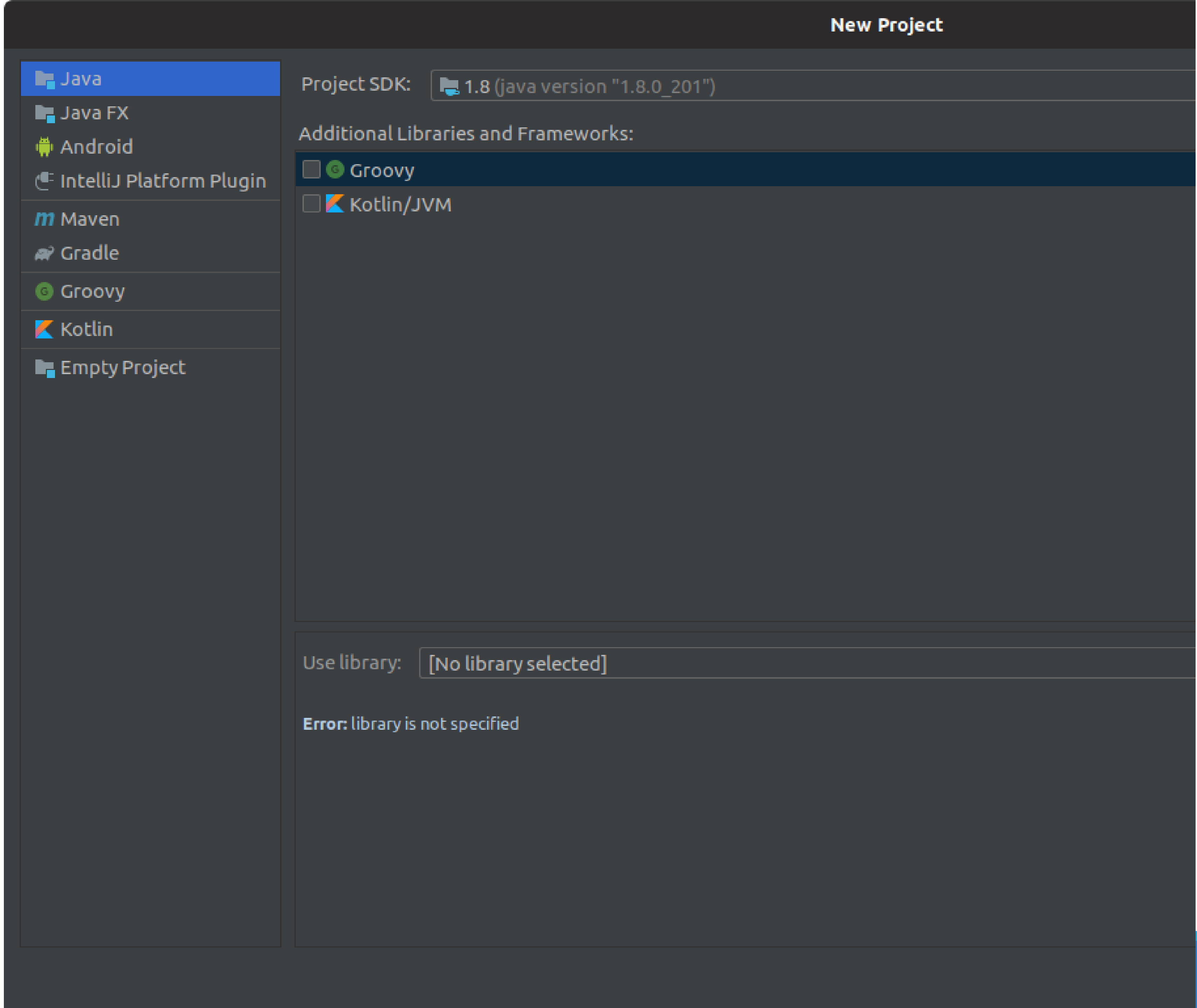
You will now see a dialog box appear *(It may be a little different than the one that I have shown)*

Starting on the left we will select a new Java Project

We will now want to select the project SDK of 15

SDK stands for Software Development Kit, and it dictates which version of Java we are using.

At this point we can hit the next button twice



Project SDK: 1.8 (java version "1.8.0_201")

Additional Libraries and Frameworks:

Use library: [No library selected]

Error: library is not specified

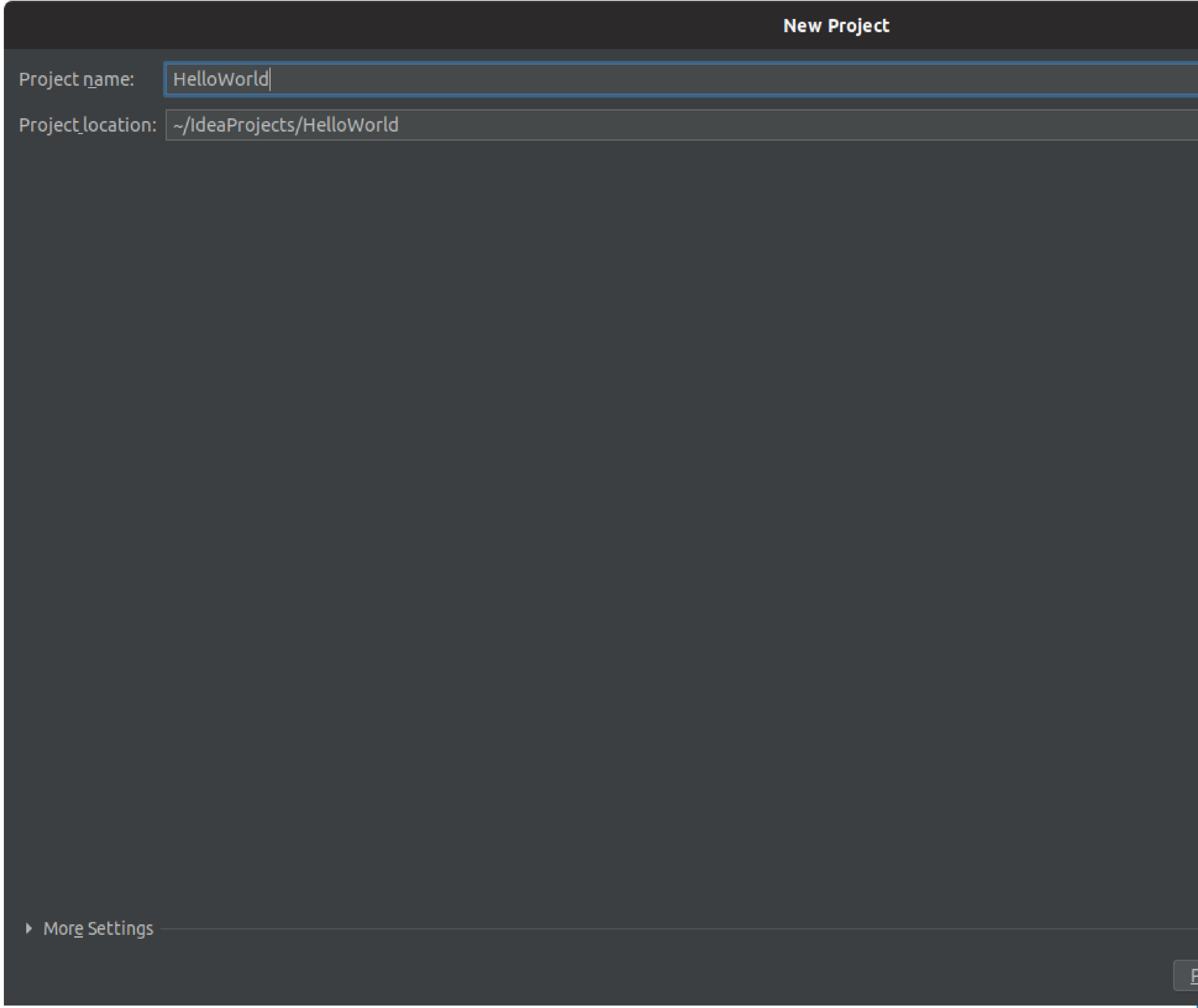
Step 2: New Project

Give the project a name of **HelloWorld**

At this time, you can change the project location to any convenient place you would like

Tip:

Keeping your programming projects on a flash/jump drive is perfectly fine. However, you will want to avoid from directly working on the drive.



Project name: HelloWorld
Project location: ~/IdeaProjects/HelloWorld

Step 2: Creating a new project

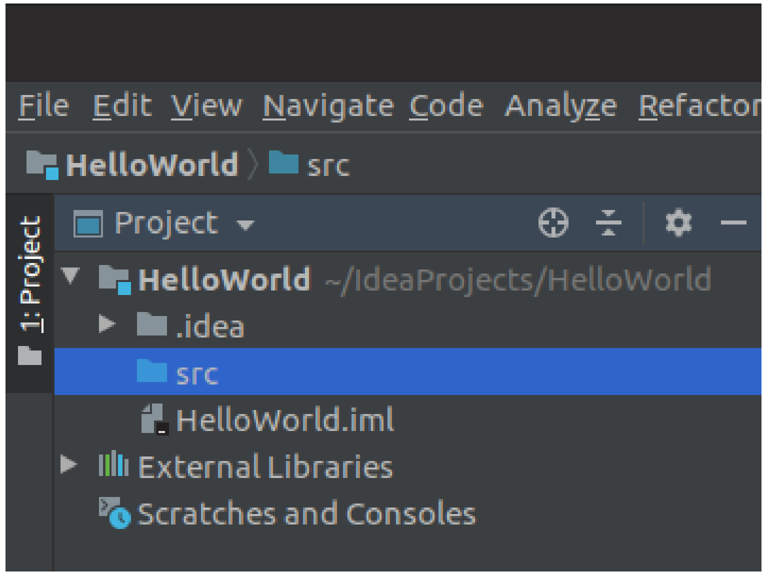
You will now see in the top left-hand corner of your IDE a package explorer.

Within the package explorer you should see the hello world project.

Expand the hello world project to see a directory named **src**.

src is where we store our **sour**ce code.

When working on a Java project it is convention that all your **source code** is stored in the **src** folder.



Step 3: Creating a new class file

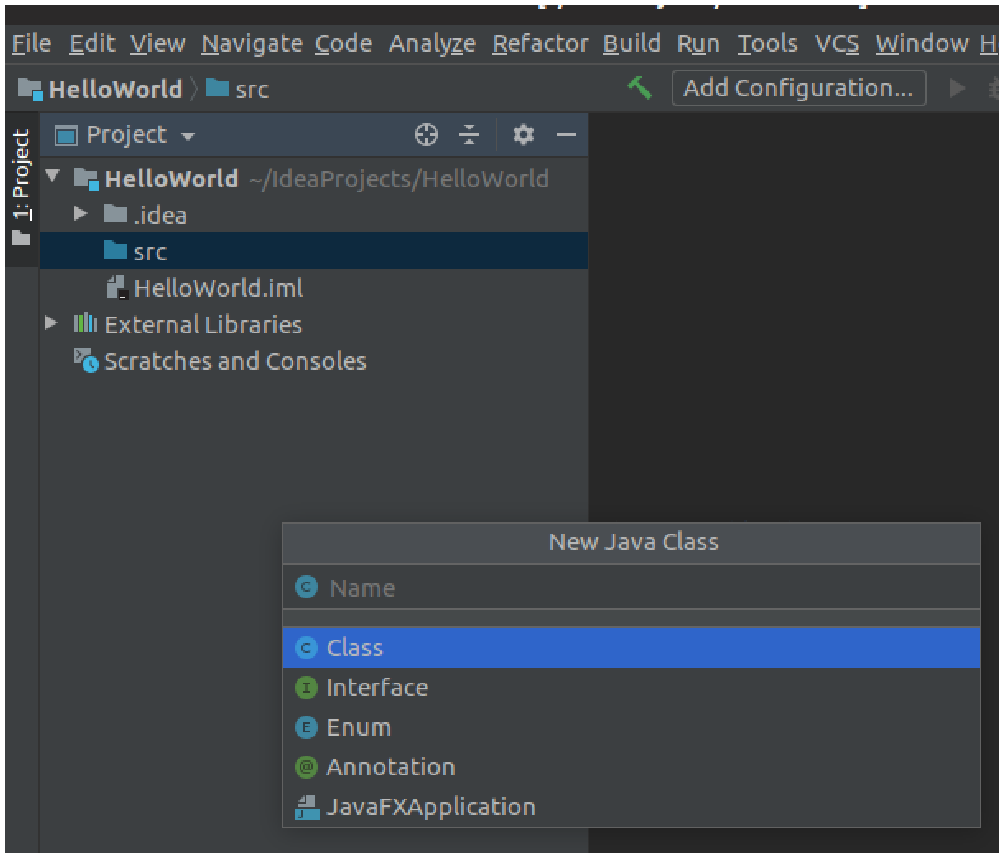
Next, we will need to create a new class file.

Right click on the **src** folder and select:

New> Java Class

Creating a new class is like creating a new program

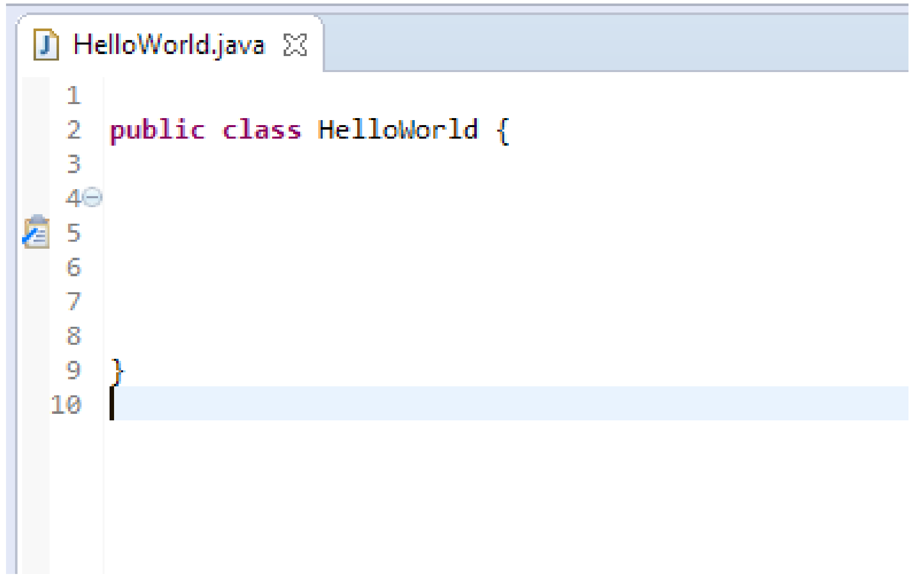
Name the class **HelloWorld** and hit **enter**



Step 4: Writing the program

You should now see a HelloWorld.java file opened on the screen.

- Refer to the image on the right to see what your file should look like
- The first thing we are going to do is type "psvm" and hit the enter key
- This is what's known as **"auto-completion"**

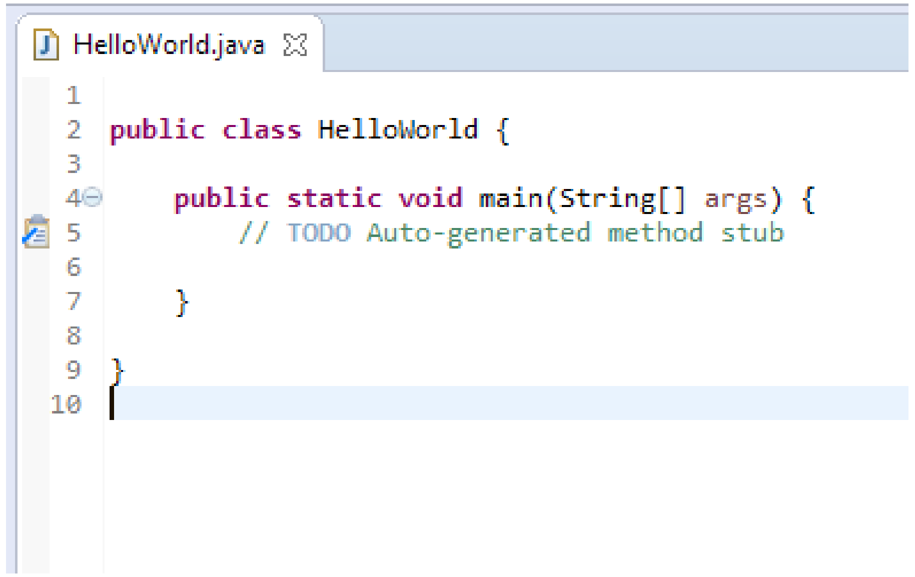


```

public class HelloWorld {






}

```

Step 4: Writing the program

We should now see the project look like the following on the right

- The next few slides will help you understand the code on the right.
- Let's examine what each line means

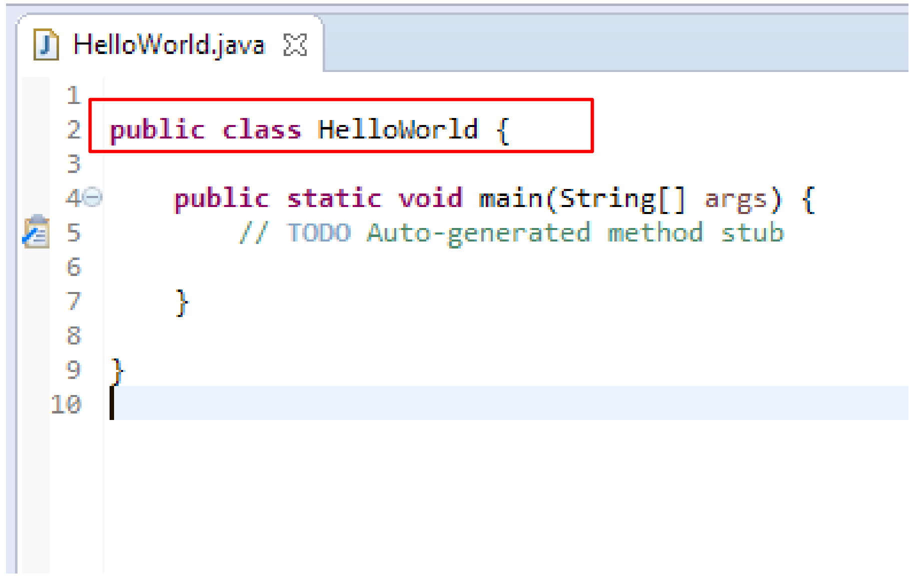


```

public class HelloWorld {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}
```

Step 4: Writing the program

Line number 2 defines a class:

- Every program in Java must have a class with a class name.
- In this case our class was named “HelloWorld”
- This tells the computer we are creating a new program and the programs name is HelloWorld



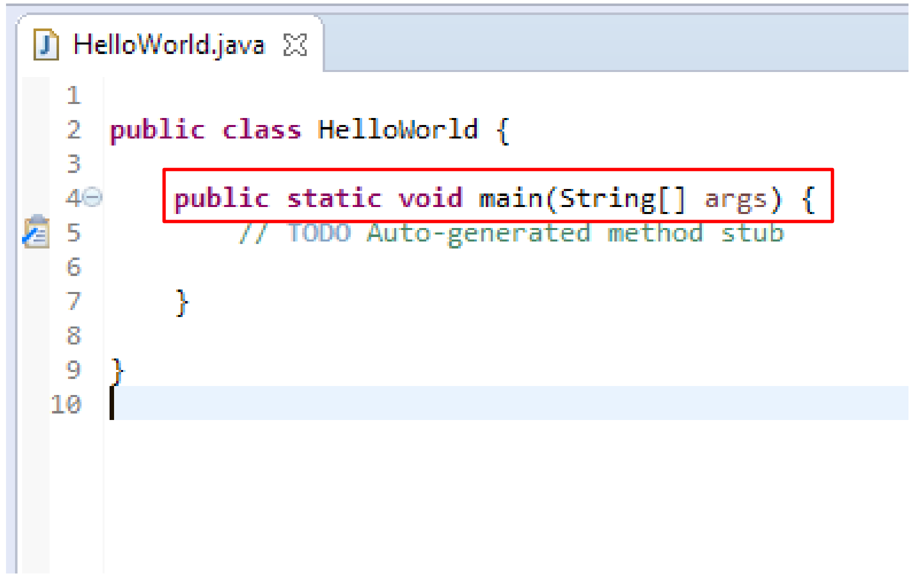
```
 1
 2 public class HelloWorld {
 3
 4     public static void main(String[] args) {
 5         // TODO Auto-generated method stub
 6
 7     }
 8
 9 }
10
```

Good programming practice:
Class names should start with a capital letter

Step 4: Writing the program

Line number 4 defines the main method:

- Although it may seem a weird concept, we can have programs that do not run. *(we will worry about this later)*
- The only concept you need to understand right now is that our program will not run without a main method
- This is where the program begins execution.
- All code we want to execute will be placed after this line

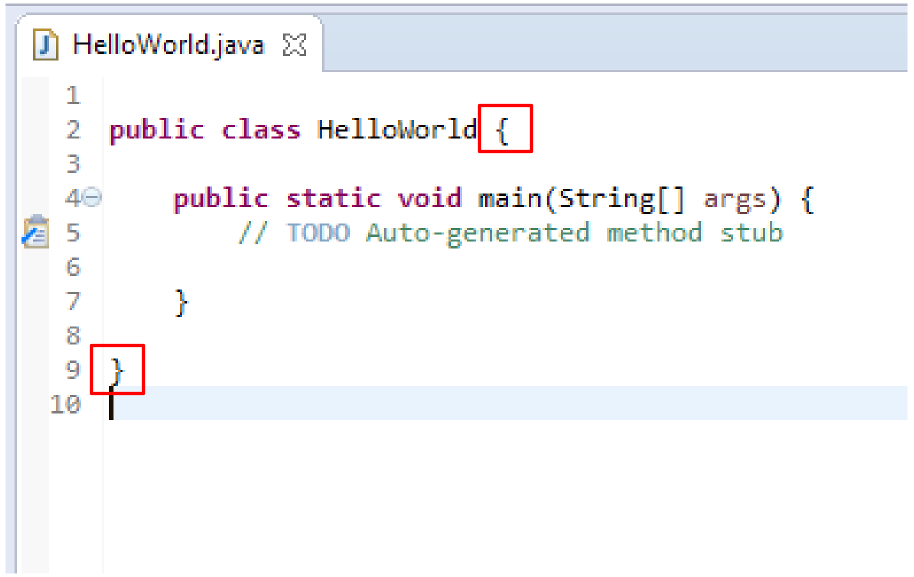


```
1
2  public class HelloWorld {
3
4      public static void main(String[] args) {
5          // TODO Auto-generated method stub
6
7      }
8
9  }
10
```

Step 4: Writing the program

The class block:

- A pair of curly braces { } creates what is called a block.
- Blocks are used to group program components
- Notice how all content is stored within the class block
- Blocks can be nested…

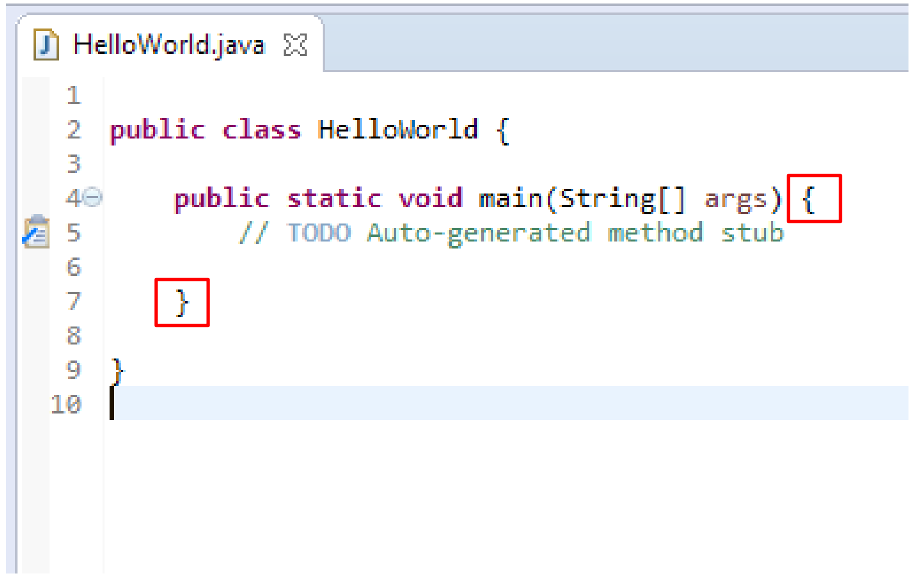


```

public class HelloWorld {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}

```

Step 4: Writing the program

The method block:

- Notice how the main method also has its own block and is nested within the class block.
- Nested meaning a block can be placed inside another block
- Content we want to execute needs to be coded within the main method block

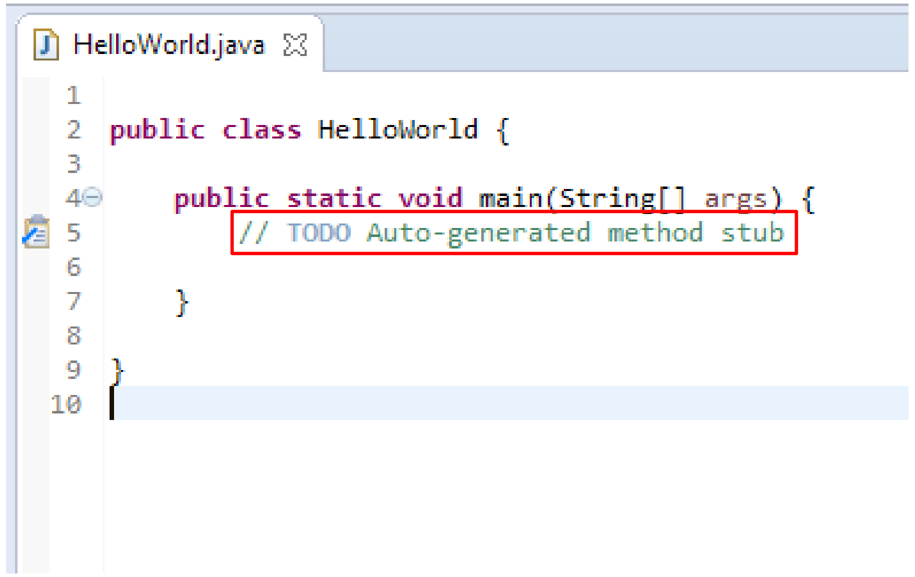


```

public class HelloWorld {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}

```

Step 4: Writing the program

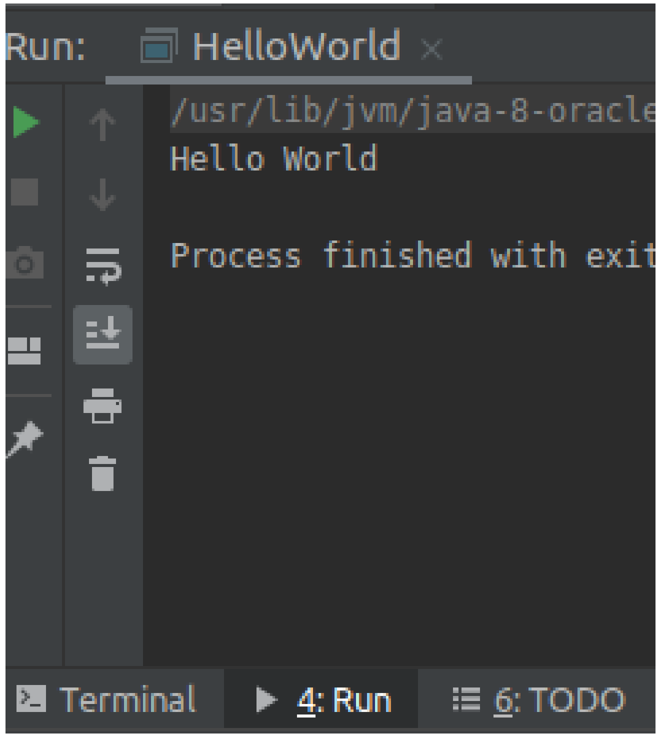
Line number 5 is a comment:

- The comment seen on line 5 was auto-generated by the IDE and can be removed.
- Comments are used to document code within Java.
- Line number 5 is an example of a one-line comment. Meaning if you are on line 5 and hit the enter key, the program will think you are writing code again on line 6


HelloWorld.java

```
 1
 2 public class HelloWorld {
 3
 4     public static void main(String[] args) {
 5         // TODO Auto-generated method stub
 6
 7     }
 8
 9 }
10
```

Step 5: Compile the code



Run: HelloWorld

```
/usr/lib/jvm/java-8-oracle
Hello World

Process finished with exit
```

The next step is to compile our Java program into bytecode and run it.

We can do this by going to the menu and selecting:

Run> Run > HelloWorld

The IDE will then automatically compile our program and run it within the console at the bottom of the screen

Special Characters

When programming java there are some special characters we need to know about:

Character	Name	Description
{ }	Curly Braces	Used to denote a block of code
()	Parentheses	Used to denote a method
[]	Brackets	Used to denote an array
//	Double slashes	Proceed a comment
“”	Quotation marks	Used to mark a string
;	Semicolon	Used at the end of every statement
/* */	Block Comment	Used for a block comment

Examine The Code

Let's examine some more of the code:

System.out.println("Hello World");

We can see that that "Hello World" is a string and that the statement was ended with a semicolon

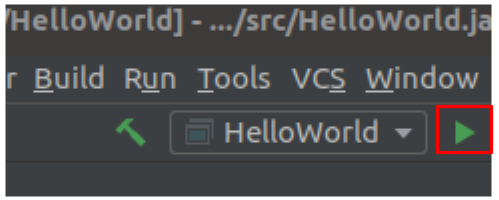
System.out.println() is a method that allows us to display code to the default output device which in this case is the console

Altering The Code

Let's alter the code:

Alter code so that your software displays the messages seen on the right

Compiling and executing can now be done by hitting the run button


HelloWorld] - .../src/HelloWorld.ja
r Build Run Tools VCS Window
HelloWorld

```
My name is "Your Name Here"
This is the altered version
This is my first program
```

Solution provided in class

Creating another class

Let's create another class:

Right click on the src folder and select:

New> Java Class

This time name the class **RoofQuote**

Once the class has been created be sure to include the public static void main method by entering "psvm" and hitting the enter key

RoofQuote

Let's alter the class:

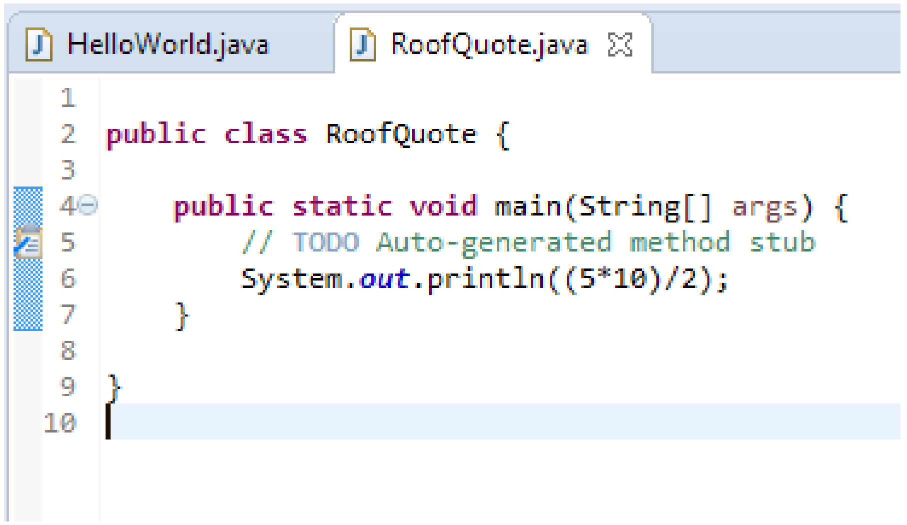
Open the RoofQuote.java file and add the following:

*System.out.println((5*10)/2);*

Compile and **run** your software.

Notice how the program **outputs 25**

Java also supports arithmetic



```

public class RoofQuote {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println((5*10)/2);
    }

}

```

RoofQuote

Arithmetic operators in Java:

Character	Name	Description
*	Multiplication operator	Used to multiply two number together
-	Subtraction operator	Used to subtract one number from another
/	Division operator	Used to divide one number by another
+	Addition operator	Used to add one number to another
%	Modulus operator	Used to calculate remainder

RoofQuote

Java follows the rules of BEDMAS:

Where it will mathematically compute everything in the **B**rackets.

Then all **E**xponents.

Then all **D**ivision and **M**ultiplication in the order they appear.

Then all **A**ddition and **S**ubtraction in the order they appear.

RoofQuote

Java follows the rules of BEDMAS:

Where it will mathematically compute everything in the **B**rackets.

Then all **E**xponents.

Then all **D**ivision and **M**ultiplication in the order they appear.

Then all **A**ddition and **S**ubtraction in the order they appear.

Truss Goodman

OUR FIRST CLIENT

Email

Truss Goodman

Local business owner

Owns a roofing business

Hello,

I am a small business owner and I need a piece of software that I can use to quote the different roofing jobs I have. I currently calculate the cost of the roof by using the following:

- Cost of the shingles
- Size of the roof
- Installation Cost per Square Foot

Truss Goodman

Add the following comments to your RoofQuote.java:

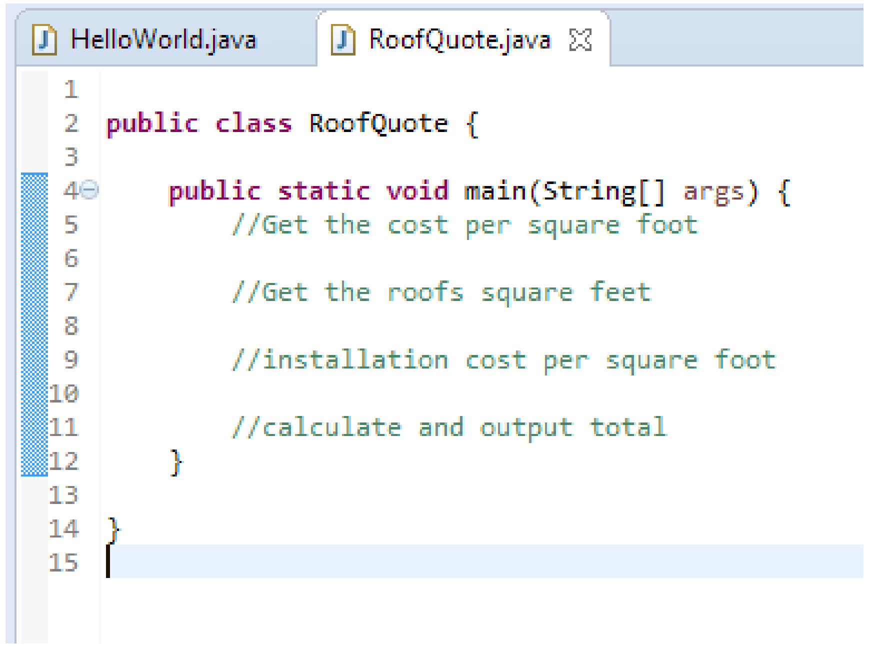
//Get the cost per square foot

//Get the roofs square feet

//Installation cost per square foot

//Calculate and output total

Before we continue past the commenting process, we need to learn another concept



```java

public class RoofQuote {

    public static void main(String[] args) {
        //Get the cost per square foot

        //Get the roofs square feet

        //installation cost per square foot

        //calculate and output total
    }

}
```

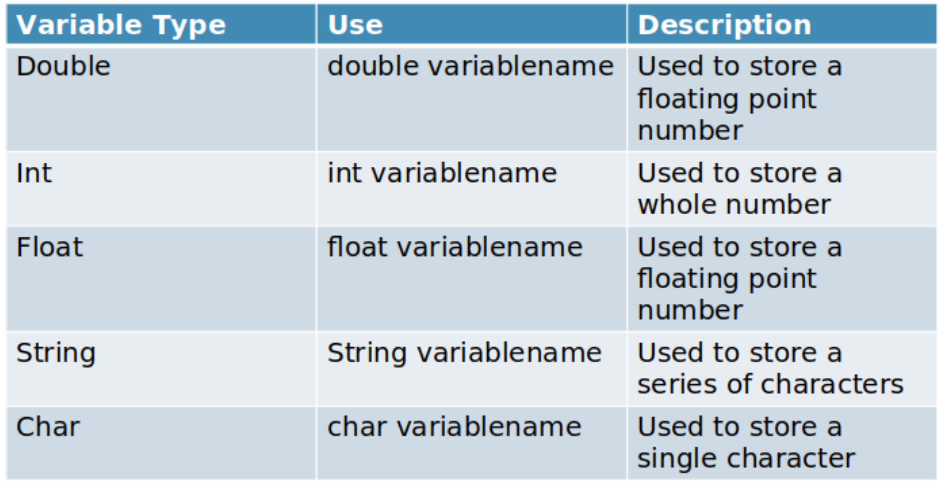
Truss Goodman

Variables allow us to store information for later use.

Variables are data containers.

These data containers can be of different data types.

Java is a strongly typed language, meaning we need to specify the data type any time we declare a variable.



Variable Type	Use	Description
Double	double variablename	Used to store a floating point number
Int	int variablename	Used to store a whole number
Float	float variablename	Used to store a floating point number
String	String variablename	Used to store a series of characters
Char	char variablename	Used to store a single character

Variables

Variables are created in the following format:

int age = 45;

String name = “Truss Goodman”;

double bankAccount = 43000.92;

We first declare the data type then the name of the variable.

We then assign the variable a value.

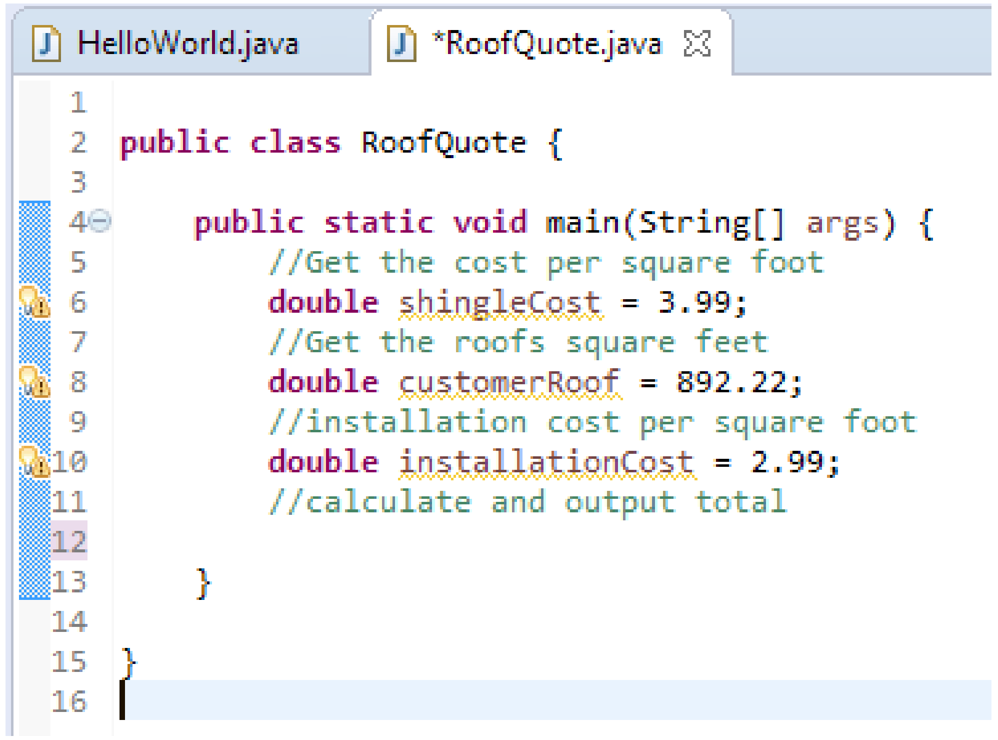
Truss Goodman

Lets create some variables to store the information we will use for the roofing quote software:

double shingleCost = 3.99;

double customerRoof = 892.22;

double installationCost = 2.99;



HelloWorld.java *RoofQuote.java

```java

public class RoofQuote {

    public static void main(String[] args) {
        //Get the cost per square foot
        double shingleCost = 3.99;
        //Get the roofs square feet
        double customerRoof = 892.22;
        //installation cost per square foot
        double installationCost = 2.99;
        //calculate and output total

    }

}

```

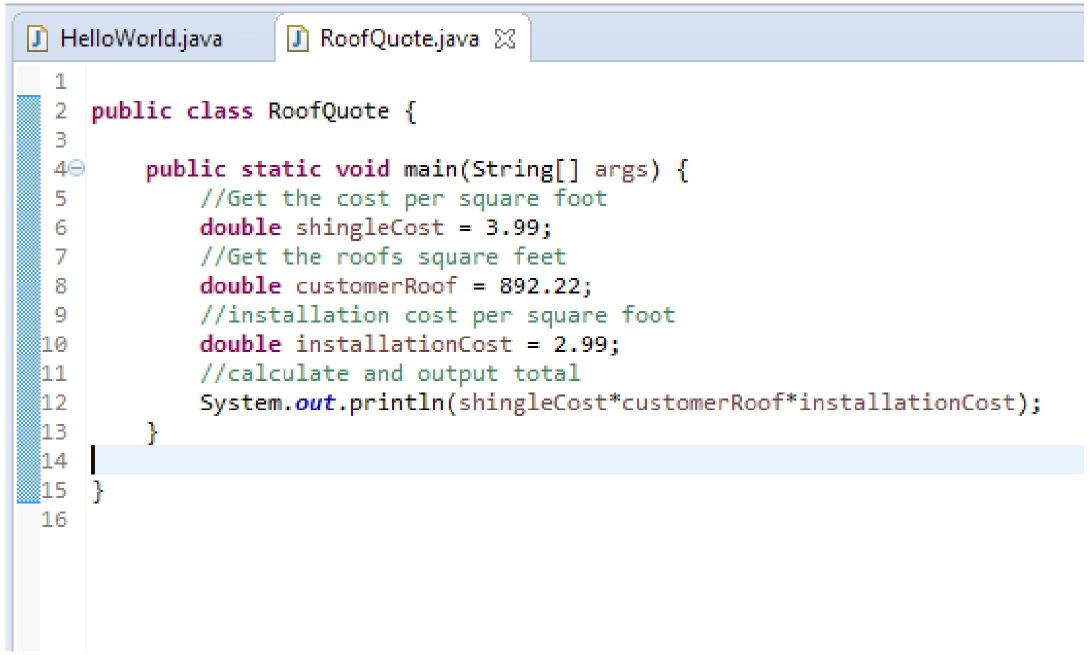
Truss Goodman

Add the final line to output the calculated total to the screen:

*System.out.println(shinleCost*customerRoof *InstallationCost);*

Compile and run the software and see the corresponding output

(there is a deliberate error placed inside this code for future learning purposes)for future learning purposes)



```

public class RoofQuote {

    public static void main(String[] args) {
        //Get the cost per square foot
        double shingleCost = 3.99;
        //Get the roofs square feet
        double customerRoof = 892.22;
        //installation cost per square foot
        double installationCost = 2.99;
        //calculate and output total
        System.out.println(shingleCost*customerRoof*installationCost);
    }

}

```

Exercise

Using what you know take minute to build a software that meets the following requirements:

- two variables (number1, number2)
- two outputs
- The sum
- The product

Compile and run the software and call me over when you are complete

Homework

Read pages 1-36 of your textbook

Next Week

Next week we will discuss the feedback Truss Goodman gave us about the software and some changes that he would like us to be able to make

Topics:

- Variables in depth
- Reading input from the console
- Assignment Statements & Expressions
- Naming constants and conventions
- Data Types
- Operator
- Numeric type conversions