INTRODUCTION TO

Java™

LESSON SUMMARY

- What is Java?
- What is Java used for?
- Who uses Java?
- Opening NetBeans to create a Java file
- Understanding the layout of NetBeans
- Understanding the layout of a Java file
- Writing your first program
- Writing your second program with input and output

- This PDF will be available after class on my Github page for free download github.com/mavnyin88/xaverianIntroToJava

WHAT IS JAVA?

- Java is a popular computer programming language (think of Java as writing the "back-bones" of a program -or- the "behind-the-scenes" functionality of a program) {
  i.e. writing the functionality for clicking a button, or the code for opening a file
  };

- Java is formally known as a "Object-Oriented" (Programming Language) {
  We don't have time to get into the specifics of what "object oriented" means but think of it as a type of programming language that uses objects to model real world objects in our applications (e.g. Classes in Java. An object is an instance of a class)
  };

- Java is famous for the Java Virtual Machine (JVM) {
  Write once run anywhere(portability, hardware independent)// < -- power of the JVM
  };

- Java is NOT JavaScript (a lot of people confuse the two);

- From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere! (Source: https://www.java.com/en/about/)

WHAT IS JAVA USED FOR?

Using Java, developers(programmers) write professional:

- Desktop applications (e.g. NetBeans)
- Mobile applications (Android Apps)
- Enterprise(business) applications (E-commerce)
- Database (OODB)

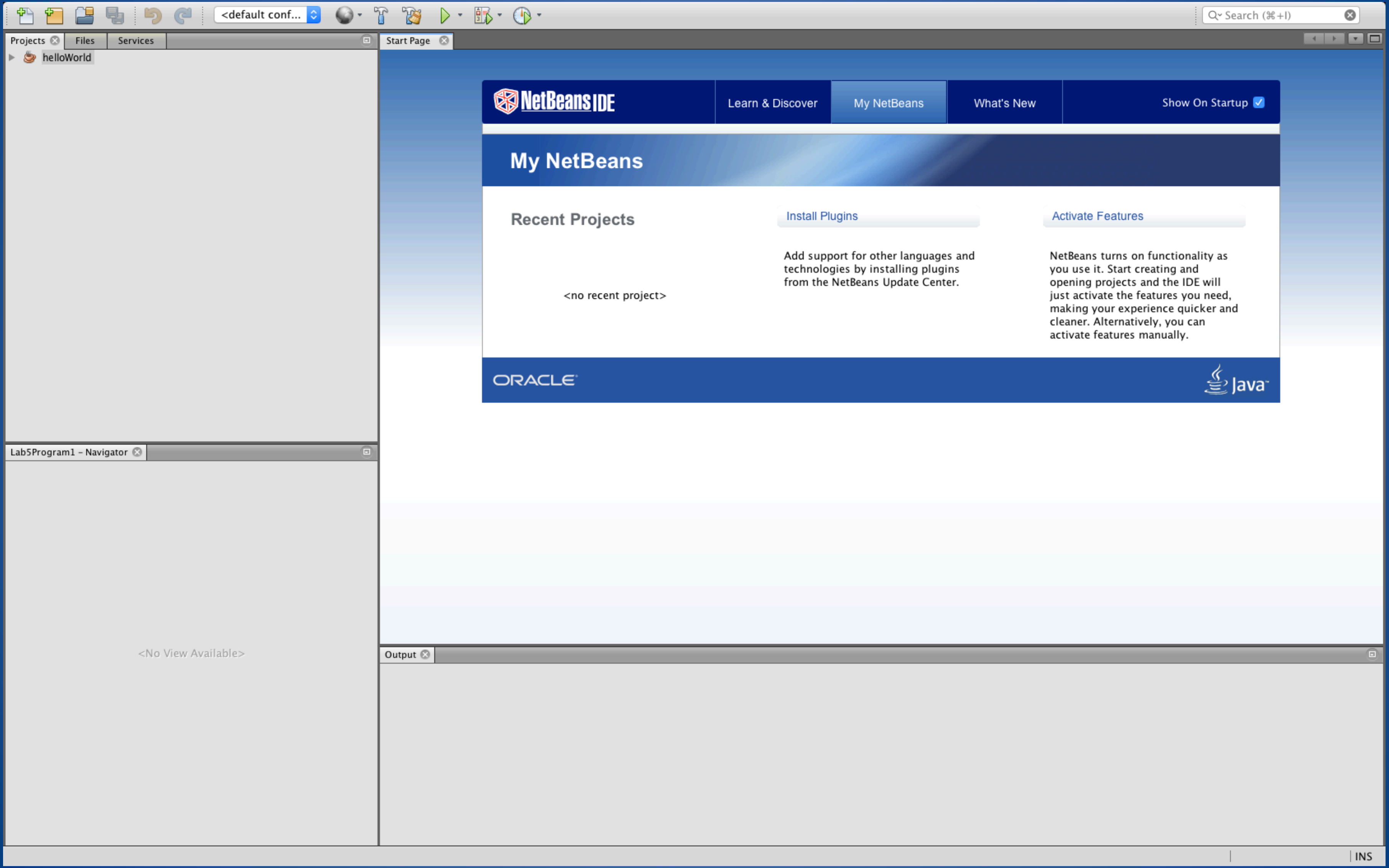
WHO USES JAVA?

Companies that currently/previously used Java:

LETS GET STARTED!

Everyone please open NetBeans
(we will do this together)

Once NetBeans is open you will see the main screen:
(* your screen may look a little different *)



My NetBeans

Recent Projects

<no recent project>

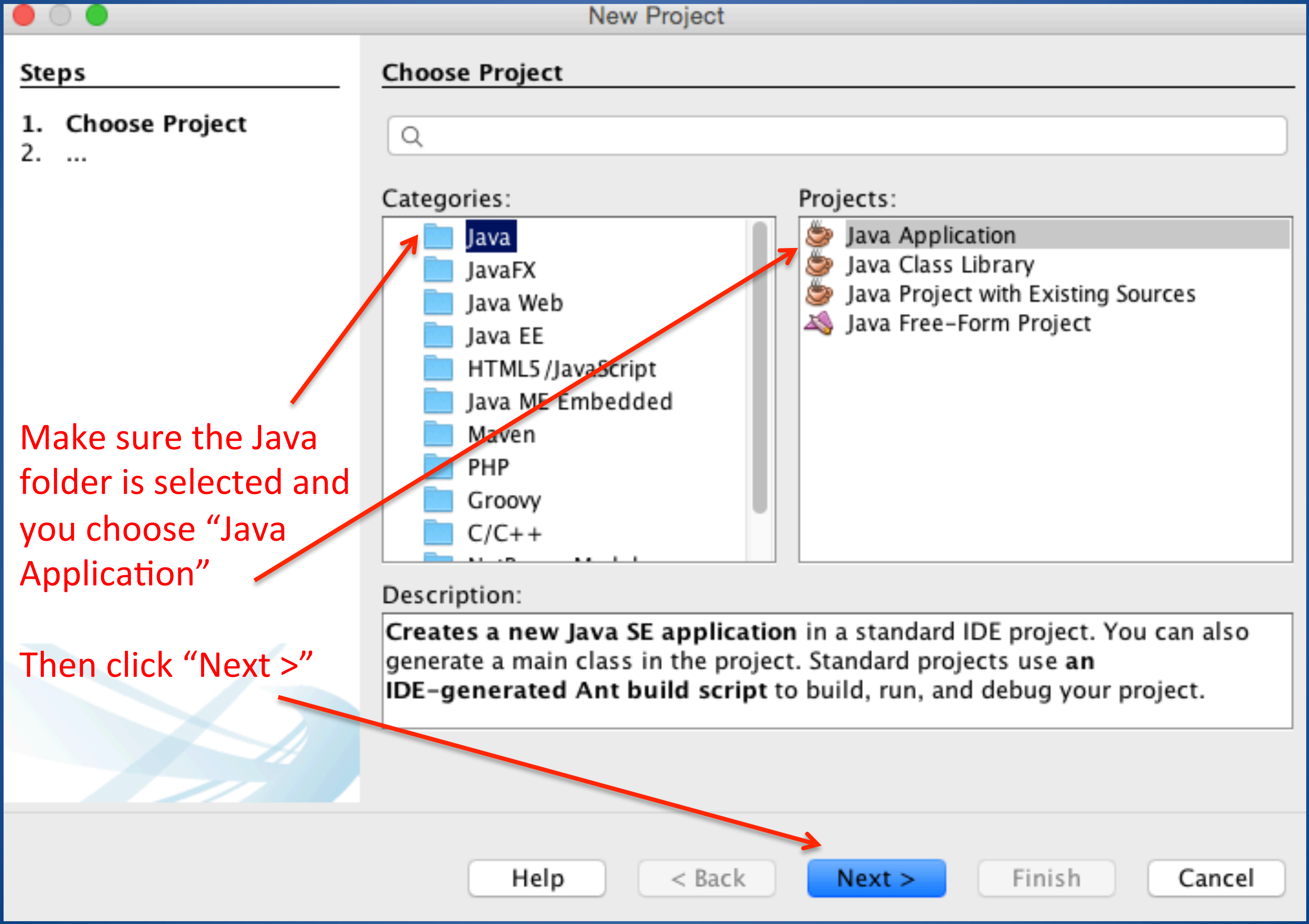
Install Plugins

Add support for other languages and technologies by installing plugins from the NetBeans Update Center.

Activate Features

NetBeans turns on functionality as you use it. Start creating and opening projects and the IDE will just activate the features you need, making your experience quicker and cleaner. Alternatively, you can activate features manually.

Everyone click on the top left corner "File" then "New Project" and you will see the following prompt.

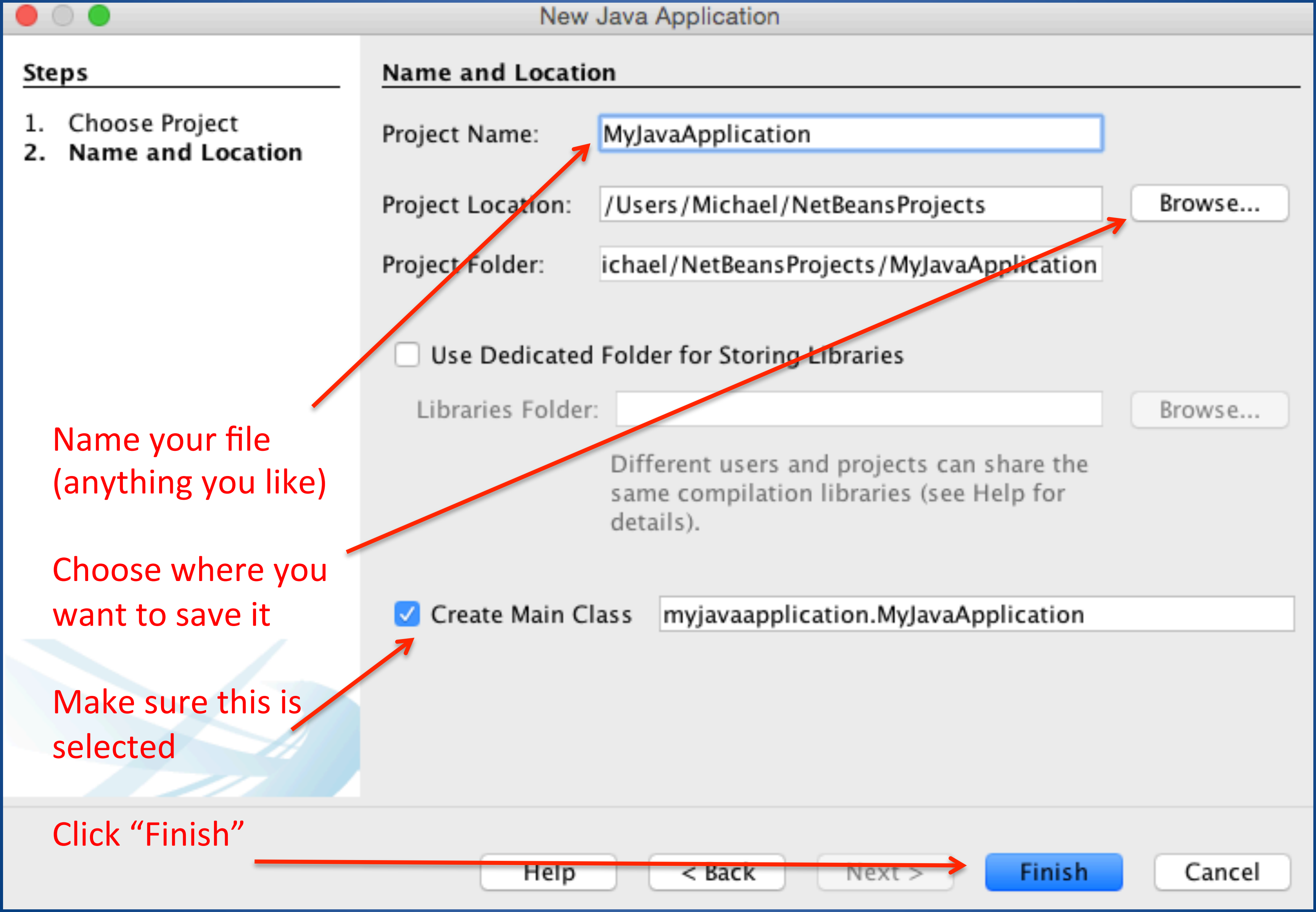


Creates a new Java SE application in a standard IDE project. You can also generate a main class in the project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug your project.

Make sure the Java folder is selected and you choose "Java Application"

Then click "Next >"

You will then be asked to name your Java Project, and save it in any location you wish.



New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name: MyJavaApplication

Project Location: /Users/Michael/NetBeansProjects

Project Folder: ichael/NetBeansProjects/MyJavaApplication

☐ Use Dedicated Folder for Storing Libraries

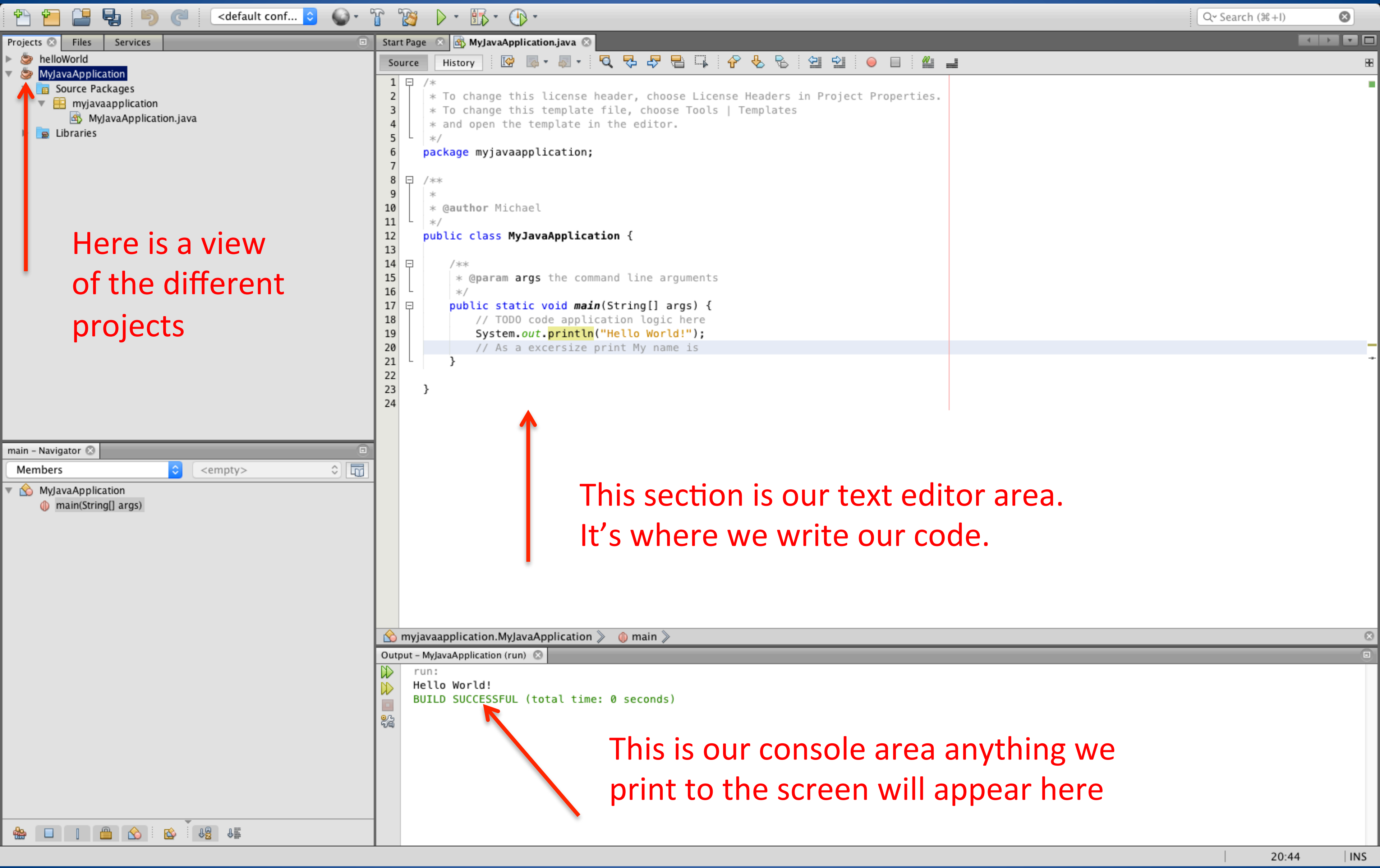
Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

☑ Create Main Class myjavaapplication.MyJavaApplication

Browse...
Browse...
Help
< Back
Next >
Finish
Cancel
Name your file (anything you like)
Choose where you want to save it
Make sure this is selected
Click "Finish"

Today we will focus on three main aspects of the NetBeans IDE:



```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package myjavaapplication;

/**
 *
 * @author Michael
 */
public class MyJavaApplication {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        System.out.println("Hello World!");
        // As a excersize print My name is
    }

}
```

```
run:
Hello World!
BUILD SUCCESSFUL (total time: 0 seconds)
```

Here is a view of the different projects

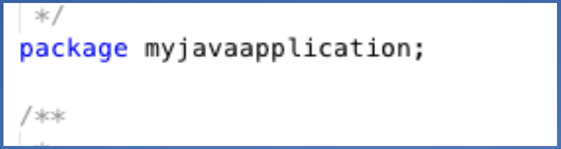
This section is our text editor area. It's where we write our code.

This is our console area anything we print to the screen will appear here

Before we get to writing code its important to understand a few notions of the Java language.

PACKAGE

package HelloWorld



```
 */
package myjavaapplication;

/**
```

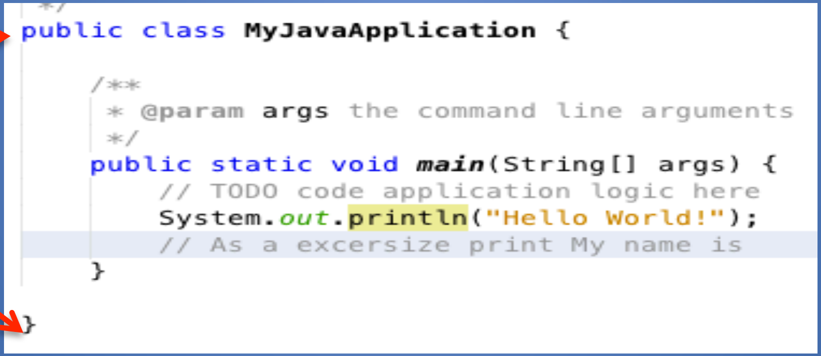
Simply put a package is a way to organize Java files.
As projects become more complex storing similar Java files in packages will help you manage your files. (same concept as directory/folders on a computer)

Class starts here

CLASS

Class ends here

class HelloWorld



```
public class MyJavaApplication {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        System.out.println("Hello World!");
        // As a excersize print My name is
    }

}
```

EVERYTHING IN JAVA IS A CLASS!

Simply put a class is a blueprint(model/plan) that contains a chunk of code contained within the two curly braces { …. } {the code written in between curly braces defines the class}
Notice "the main function" is located within our "class" (Main function executes our code)
Today we will be writing all of our code within the main function.

*NOTE YOUR PACKAGE AND CLASS NAME WILL ALWAYS BE THE SAME AS WHAT YOU NAMED YOUR JAVA PROJECT

EXERCISE 1.1 (Done together)

Print "Hello World to the screen"

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

System is a class, out is variable, println() is a built in Java function

EXERCISE 1.2 (Done by students then together)

Print "My name is …"

```
System.out.println("My name is Michael");
```

Every line of text in between the quotation marks " …. " is considered a String.
In Java a String is an "object" made up of several characters.
(i.e. a string is an object (its an instance of the built in Java class String)).
Simply put a String is a data type that stores text(characters).

EXERCISE 1.3(Done together)

Create a String variable that stores students first name and print it to the screen.

```
String firstName = "Michael";
```

Declaring firstName to be a String

firstName is an "object" and instance(example) of class String with special properties.

The equal sign = initializes our string firstName to store the value Michael. Its important that the value is enclosed within the quotation marks "Michael".

Example like in math (x = 10 …. x + 1 = 10 + 1 = 11)

```
System.out.println(firstName);
```

Now that we understand how to print to the console(screen), lets learn how to print a message to the screen to ask the user for input.

We will introduce three new topics here.

- The Scanner class for input
- Java API(Application Program Interface) to access the Scanner class (java.util.*)
- Concatenation(+)

EXERCISE 2.1 (Done together)

Print to the screen: What is your age?

Have the user enter there age then print to the screen.

```
System.out.print("Enter your age: ");
Scanner myAge = new Scanner(System.in);
String age = myAge.next();
System.out.println("Your age is: "+age);
```

The text we enter on the screen will get stored in this variable(myAge).

Then create a variable age which stores the users input as a string (next() is a method of scanner class that gets user input)

Concatenation (+)
think of it as adding strings together to print one combined string to the screen

EXTRA EXERCISES IF TIME ALLOWS

EXERCISE 2.2 (Done together)

First and last name.
Have the user enter his or her first name and last name.

Print both to the screen.

```
System.out.print("Enter your first name: ");
Scanner myFirstNameScanner = new Scanner(System.in);
String fName = myFirstNameScanner.next();
System.out.print("Enter your last name: ");
Scanner myLastNameScanner = new Scanner(System.in);
String lName = myLastNameScanner.next();
System.out.println("Your name is: "+fName+" "+lName);
```

EXERCISE 2.3 (Done together)

Username and password.
Have the user enter a username and password for a basic login.

Print both to the screen.

```
System.out.print("Create a user-name: ");
Scanner userNameScanner = new Scanner(System.in);
String userName = userNameScanner.next();
System.out.print("Create a password: ");
Scanner passwordScanner = new Scanner(System.in);
String password = passwordScanner.next();
System.out.println("Your username is: "+userName+" and your password is: "+password);
```

CONGRATULATIONS ON LEARNING JAVA AND WRITING YOUR FIRST PROGRAM!