Introduction to Java Programming

Lecture 1 of 8
March 19, 2019
Emerson Family School

Maxwell Young

Overview of Course

Assistant prof at Mississippi State Univ. in Computer Science

- algorithm design & analysis, networks, security

Target audience:

- zero to very little programming experience

Tentative course content:

- variables, data types, operators
- program flow control: if-else, switch, for and while loops
- methods
- data structures
- object oriented programming

Goals:

- give you basic programming knowledge and good practices
- impart an appreciation for programming
- develop problem-solving attitude

Overview of Course

Class structure:

- 40 minutes of lecture + interactive programming
- 80 minutes of lab work

Expectations:

- your attendance in each class is important
- ***perseverance:*** trying and failing is important to learning
- patience: if I don't know the answer, I'll find out and get back to you

Resources:

- lecture slides available after class at `www.maxwellyoung.net/teaching`
- searching for online is useful if you get stuck, lots of examples

Very useful:

Free tutorials by Oracle: `https://docs.oracle.com/javase/tutorial/`

Documentation on classes: `https://docs.oracle.com/javase/7/docs/api/`

Overview of Course

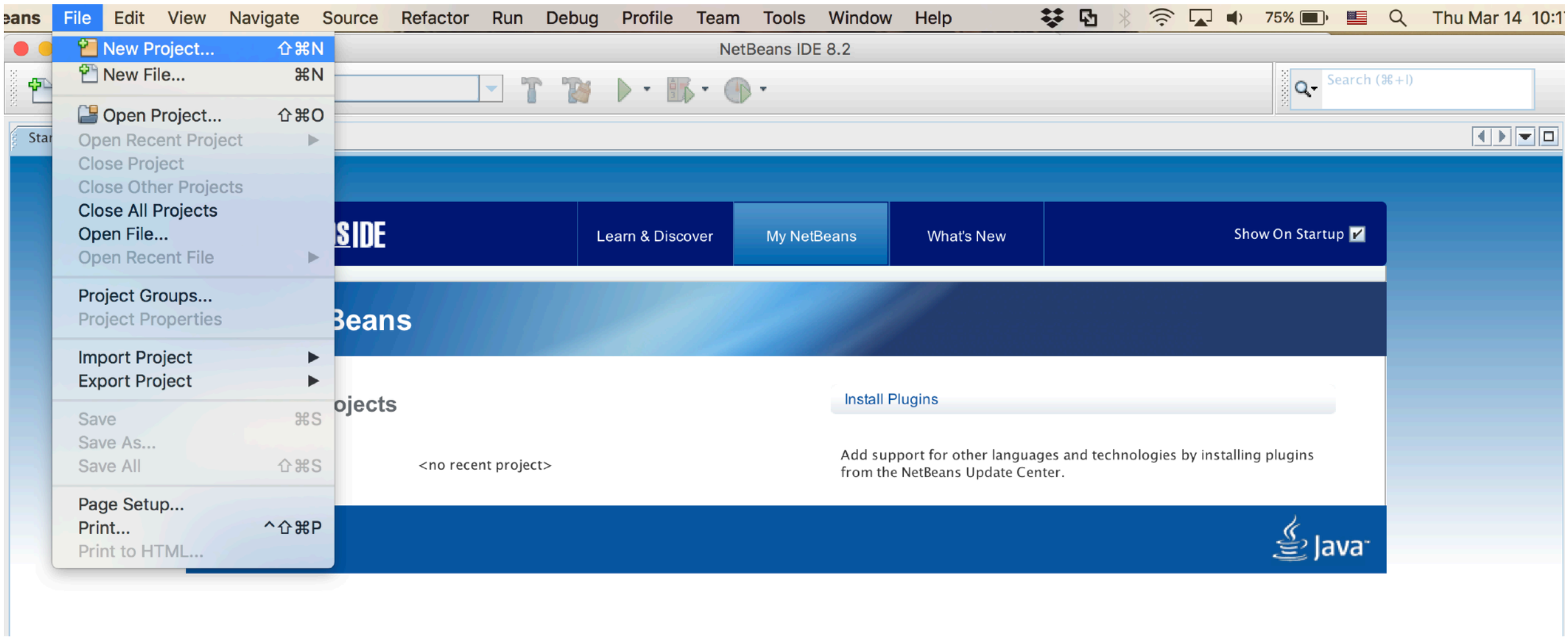
Why learn Java?:

- mindset (logical/algorithmic problem solving) and practices translate to many programming languages
- object-oriented approach carries over to C++ to a large extent
- many large tech companies allow you to do interviews in Java
- hard to quantify, but Java is a widely-used language in industry
- can build whatever you like, be creative, it's fun

Writing a Simple Program

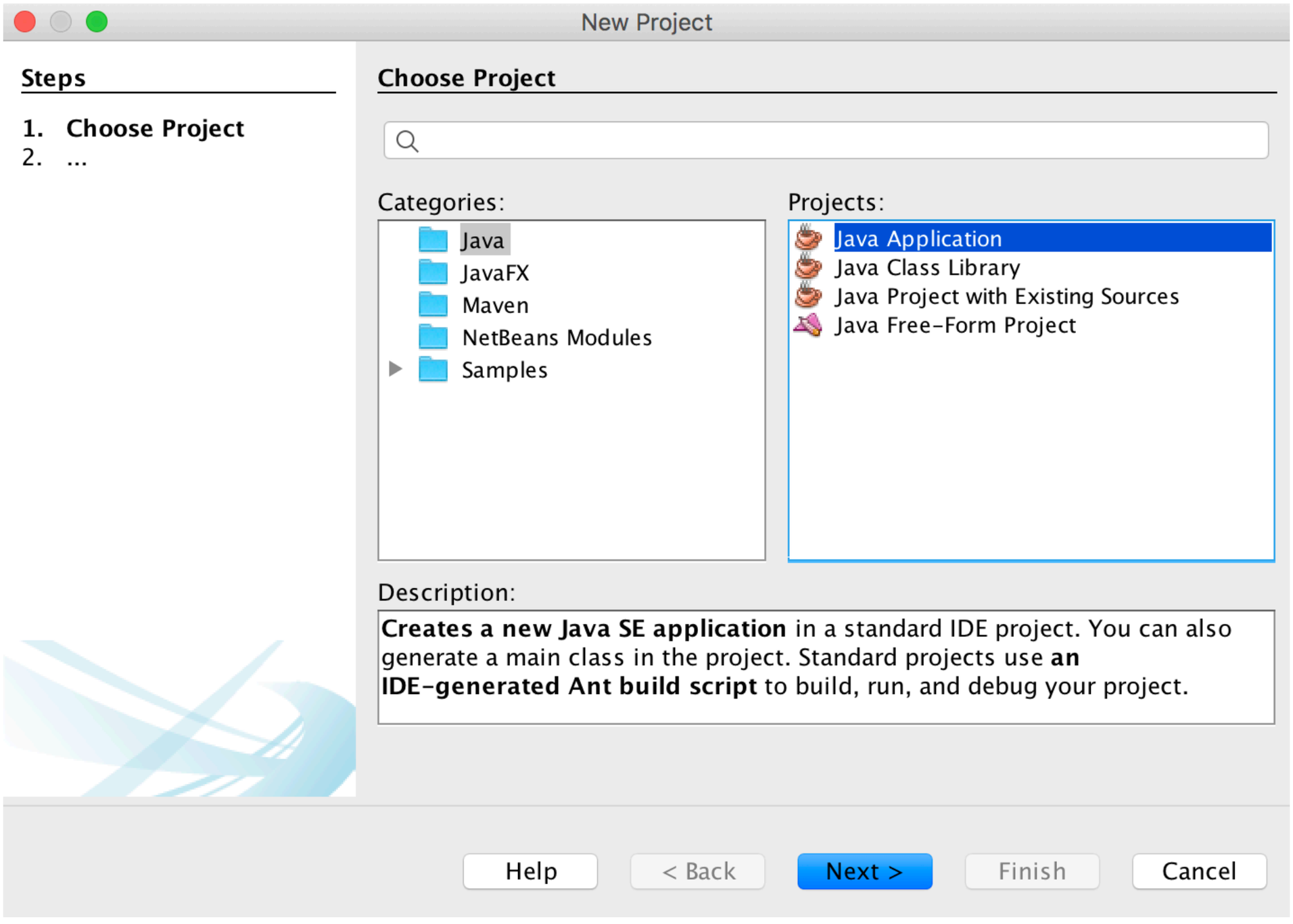
Let's do some coding

Open Netbeans. Set up ``New Project..."



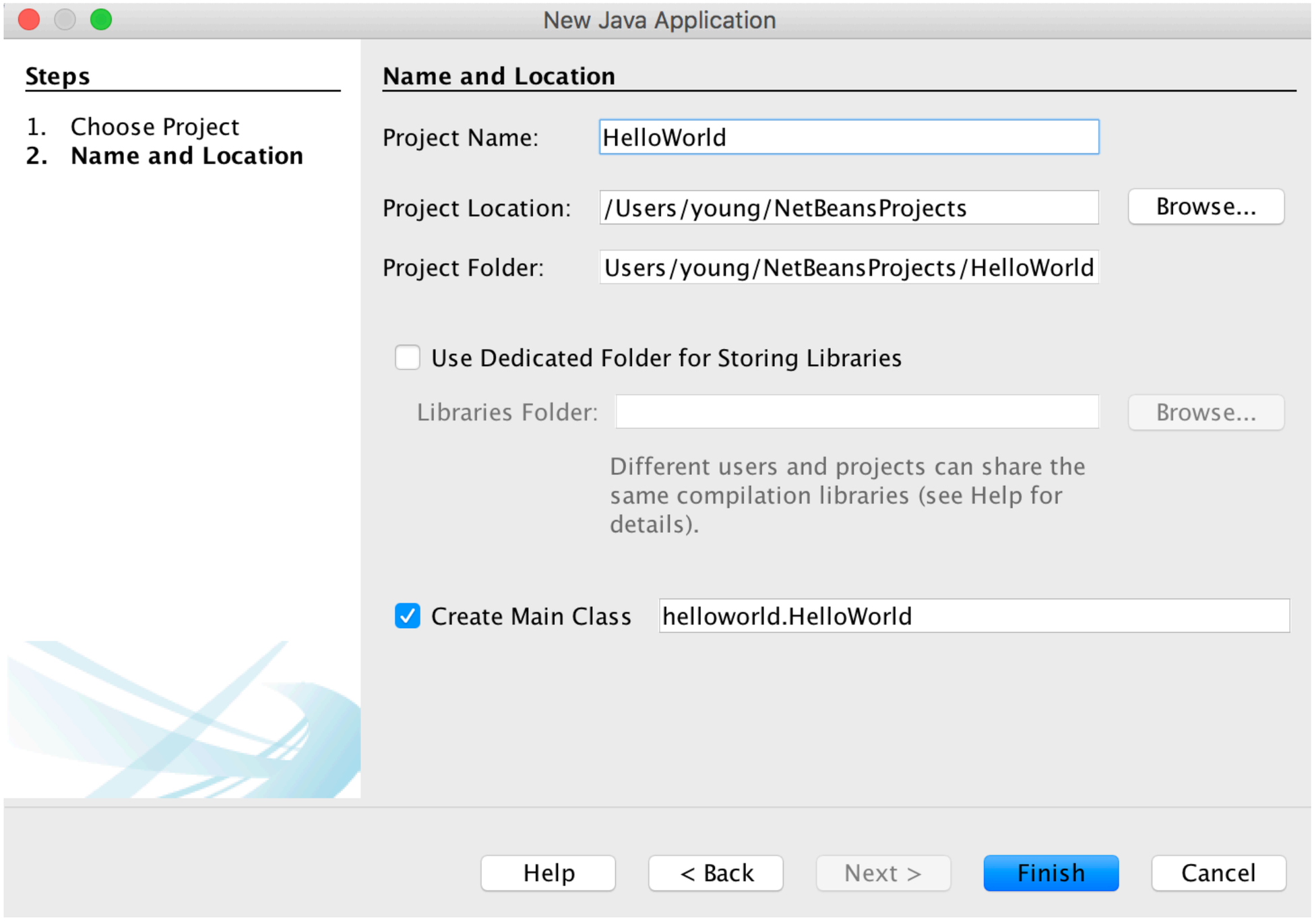
Writing a Simple Program

Select ``Java Application''



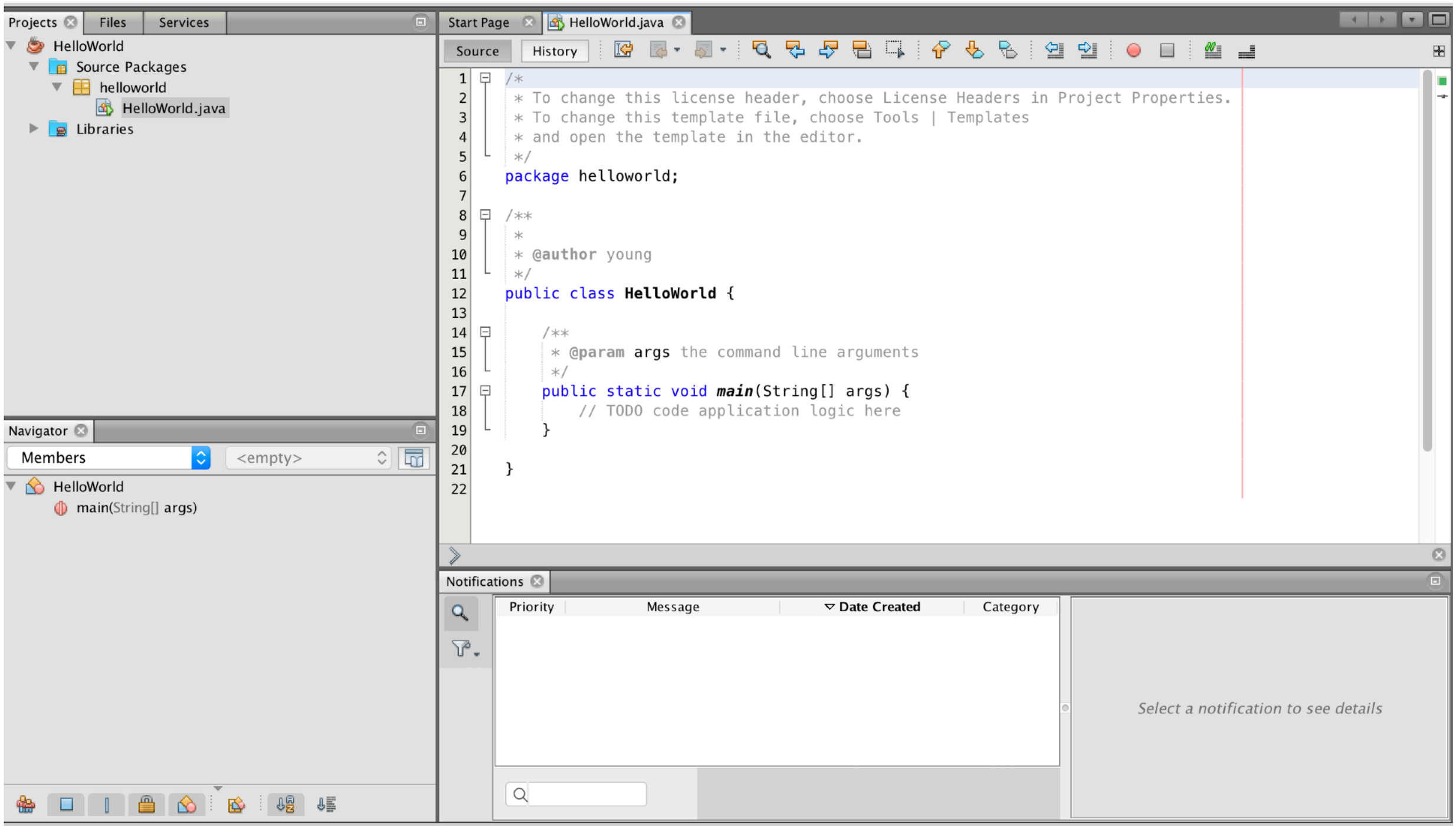
Writing a Simple Program

Give a project name of ``HelloWorld''



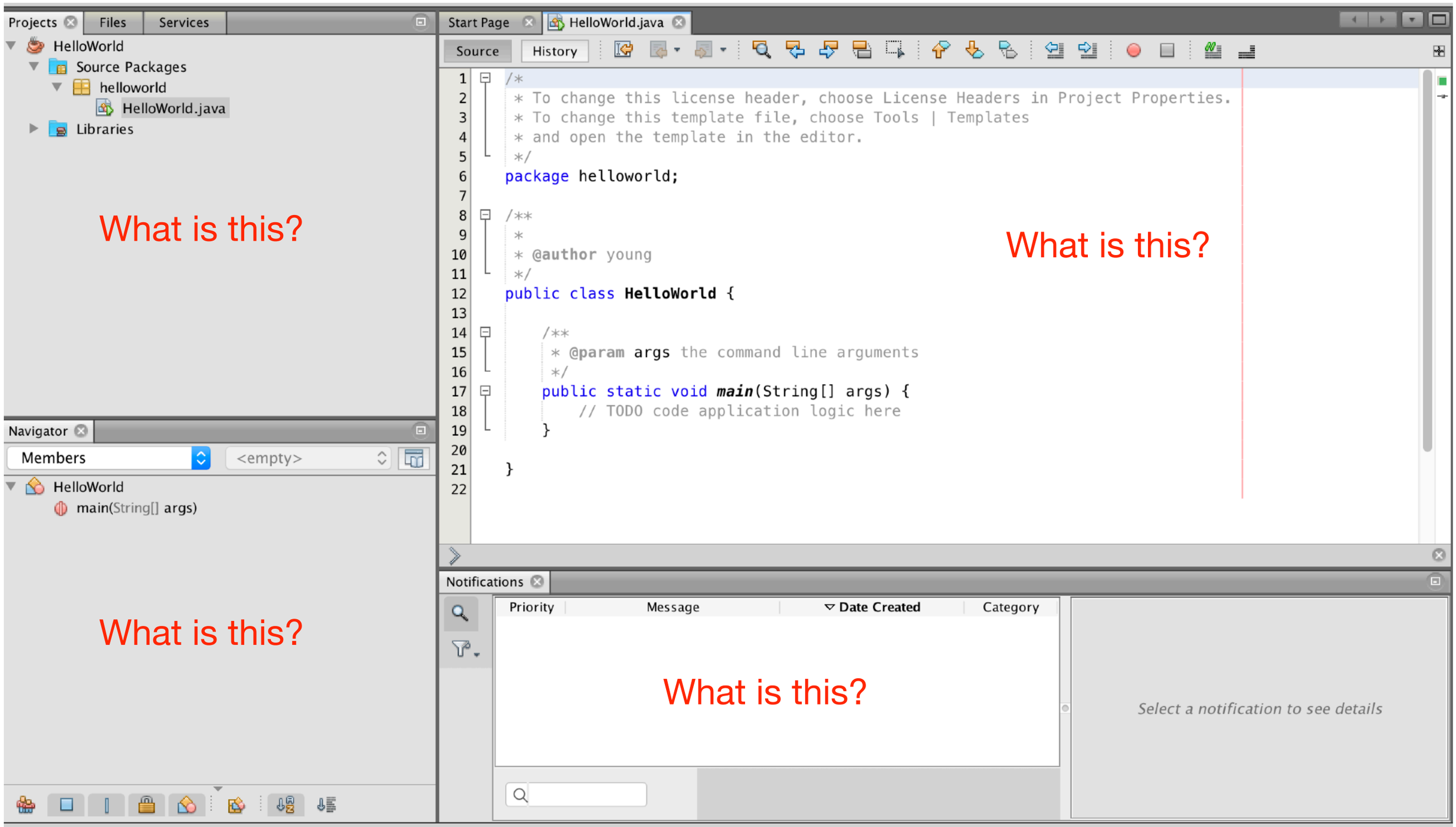
Writing a Simple Program

You should see this in your Integrated Developer Environment (IDE):



Writing a Simple Program

You should see this in your Integrated Developer Environment (IDE):



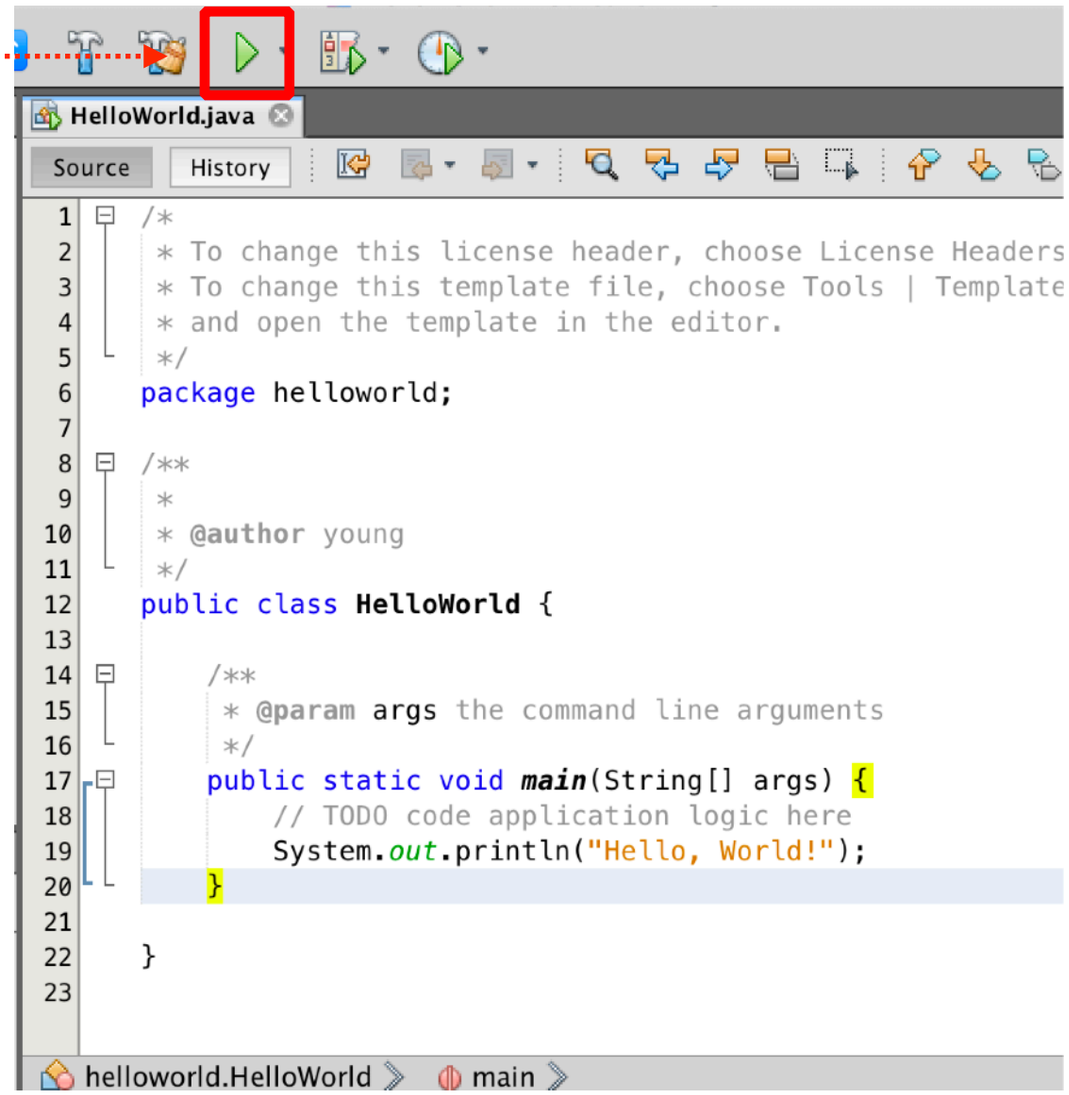
Writing a Simple Program

In Source View pane, add the following line of code (in red)

```
public static void main(String[] args) {
    // TODO code application logic here
        System.out.println("Hello, World!");
}
```

Now, compile your code

- use green button at top of IDE



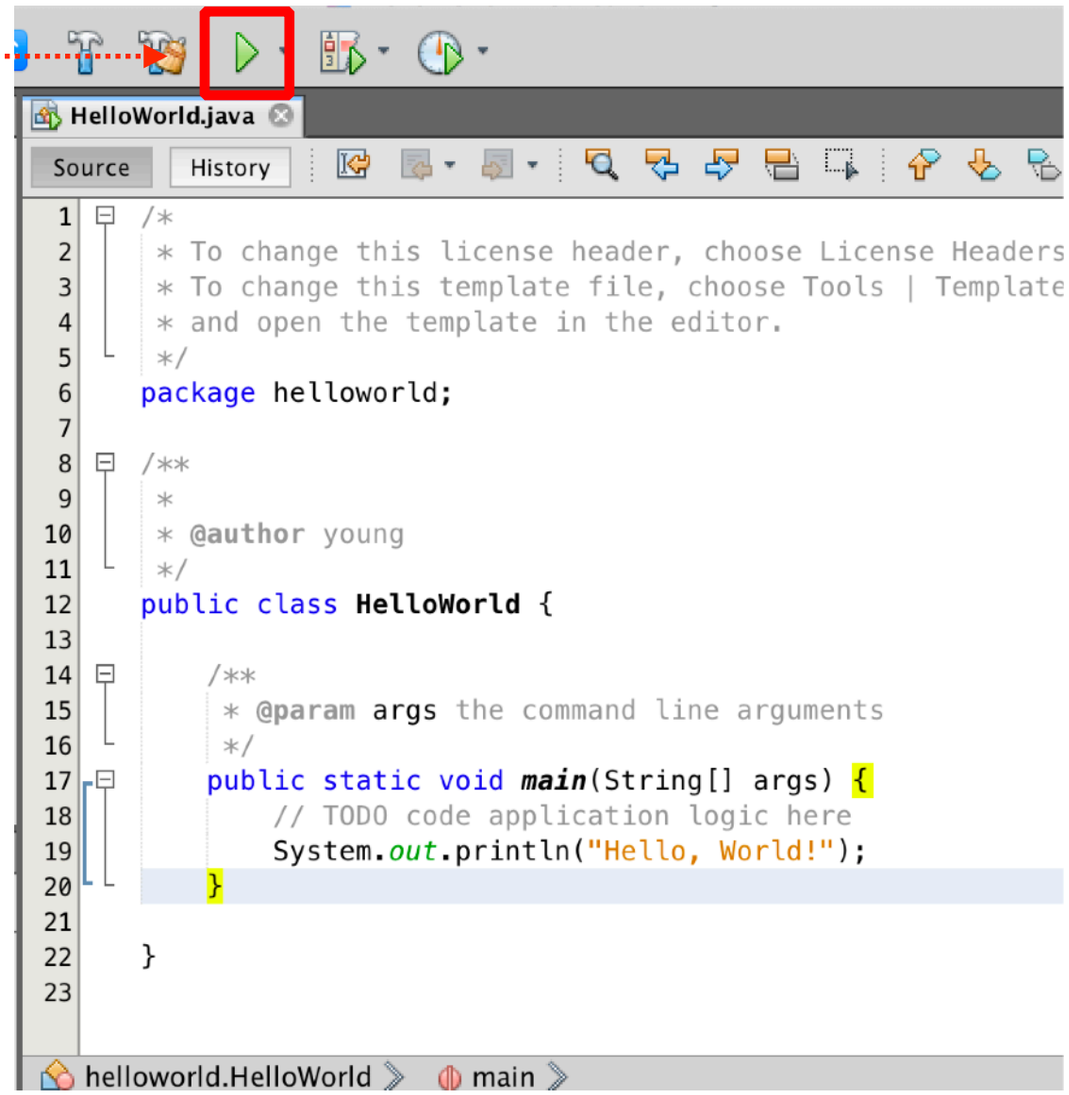
Writing a Simple Program

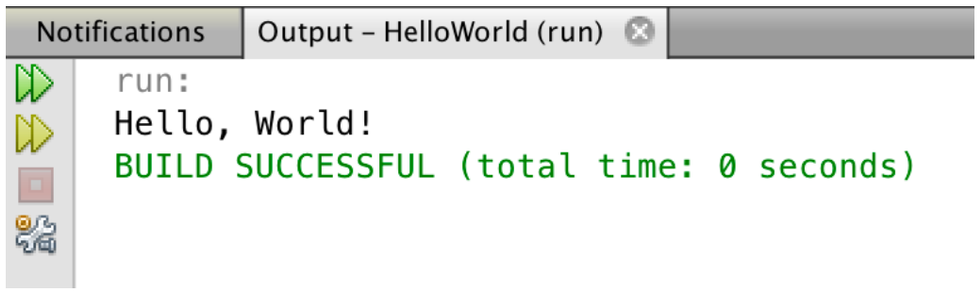
In Source View pane, add the following line of code (in red)

```
public static void main(String[] args) {
    // TODO code application logic here
        System.out.println("Hello, World!");
}
```

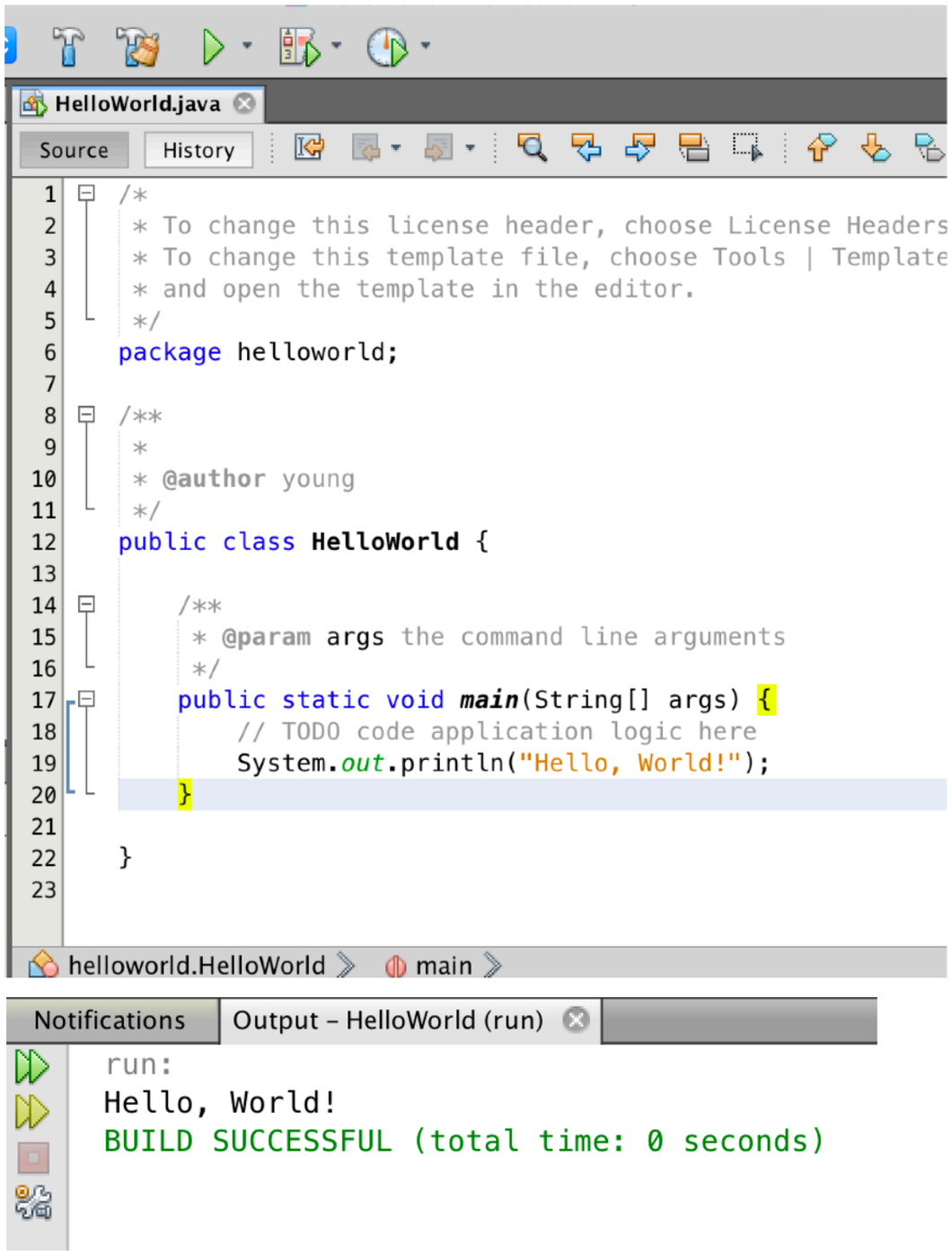
Now, compile your code

- use green button at top of IDE





Writing a Simple Program

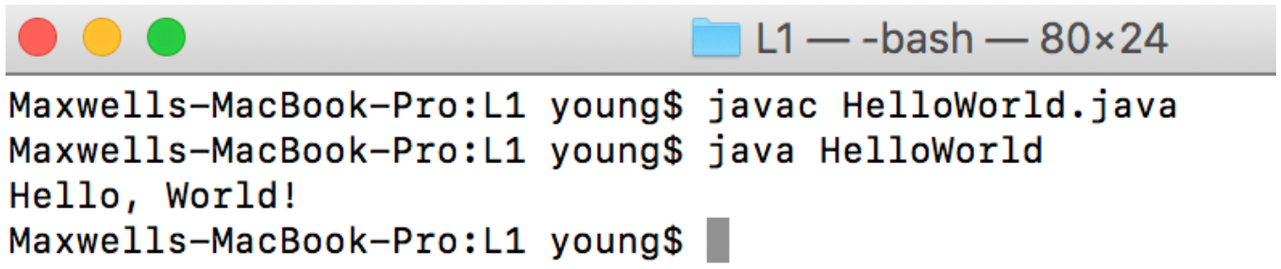


Note helpful things from IDE:

- text highlighting
- bracket matching (yellow highlighting)
- error detection (red underline)
- compile-error summary in output pane

Don't need an IDE, but often used

- can use simple text editor to write code
- need java compiler (`javac`)



Writing a Simple Program

HelloWorld.java

Source | History

```
1  /*
2   * To change this license header, choose License Headers
3   * To change this template file, choose Tools | Template
4   * and open the template in the editor.
5   */
6  package helloworld;
7
8  /**
9   *
10  * @author young
11  */
12 public class HelloWorld {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         // TODO code application logic here
19         System.out.println("Hello, World!");
20     }
21
22 }
23
```

helloworld.HelloWorld > main >

What do this code mean?

Comments start with `/*`, `*` on each line, end with `*/`
Also `// this is a comment`

`package` = (roughly) collection of classes

`class` refers to a ``template'' that defines an ``object''

- has attributes and member functions
- we defined a HelloWorld object

`public` means class is accessible to any other class of objects

- perhaps another object will want to use a HelloWorld object

Class name must be same as filename

Writing a Simple Program

HelloWorld.java

Source | History

```
1  /*
2   * To change this license header, choose License Headers
3   * To change this template file, choose Tools | Template
4   * and open the template in the editor.
5   */
6  package helloworld;
7
8  /**
9   *
10  * @author young
11  */
12 public class HelloWorld {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         // TODO code application logic here
19         System.out.println("Hello, World!");
20     }
21
22 }
23
```

helloworld.HelloWorld > main >

What do this code mean?

`static` defer for now, but roughly refers to things that ``belong'' to class rather than instance of class

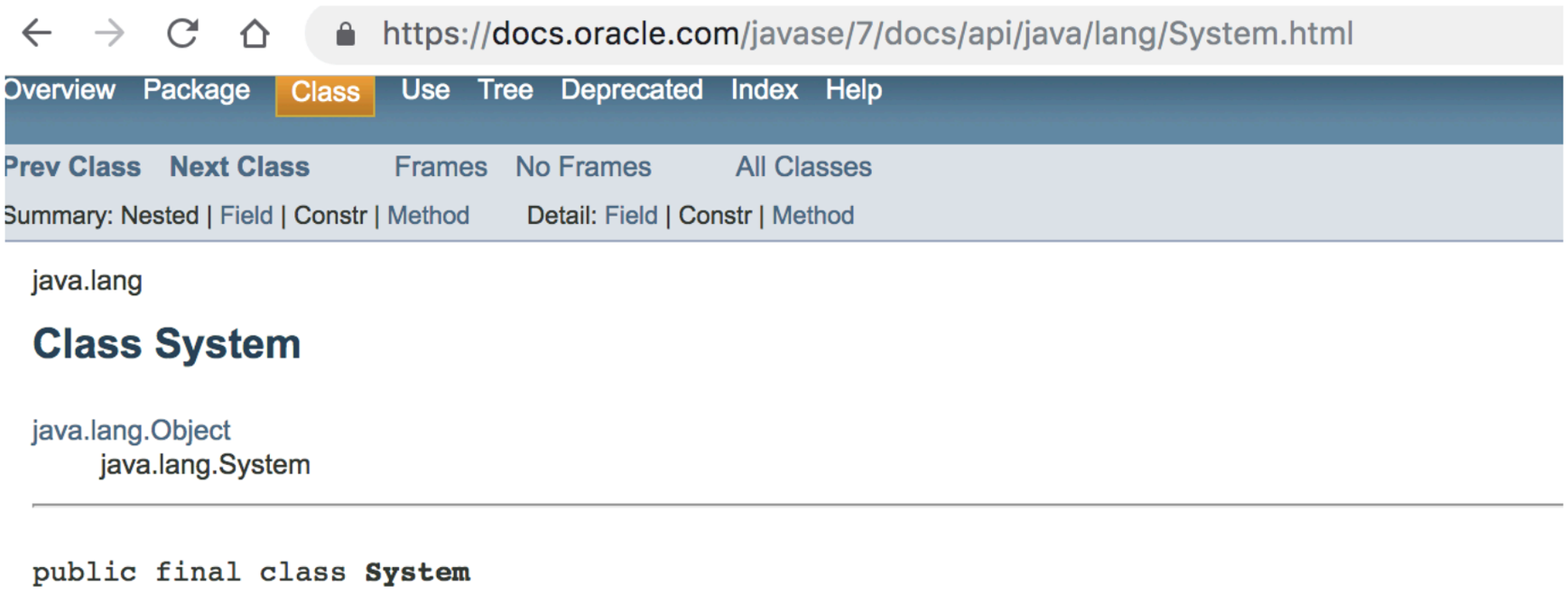
`main` drives execution of your code

- must be public and static

`System.our.println` prints line of text to screen

- `System` is a class
- `out` is a member of System class of type PrintStream
- `println` method of PrintStream class

Writing a Simple Program



Remember you can find all this:

Documentation on classes: https://docs.oracle.com/javase/7/docs/api/

All of this will become clearer and more natural as we code more

Main take away: A Java program is a set of steps written in proper syntax

Basic Data Types

Variables allow you to store state

Variable with name `myInt` of type `int`

```
int myInt = 5
```

This variable will store an integer; that is, hold it in computer's memory

Good practice: use sensible names starting with lowercase letter from alphabet capital letter for each word

Can use numbers, $, and _ characters also (cannot use whitespace)

Basic Data Types

Some other useful data types (not exhaustive)

```
double myDouble = 3.14159;

boolean myBool = true;

char myChar = `a';
```

Above are ***primitive types*** in Java programming language, NOT objects

Another useful ``data type'' (although, technically not a data type)

```
String myString = "Take me to your leader";
```

Is actually an object, but with special support given by " " to create object

Basic Data Types

Some other useful data types (not exhaustive)

```
double myDouble = 3.14159;

boolean myBool = true;

char myChar = `a';
```

} ***literal*** = constant value assigned to variable

Above are ***primitive types*** in Java programming language, NOT objects

Another useful ``data type'' (although, technically not a data type)

```
String myString = "Take me to your leader";
```

Is actually an object, but with special support given by `" "` to create object

Basic Data Types

I write and attempt to compile:

```
double myDouble;
System.out.println(myDouble);
```

What happens?

Basic Data Types

I write and attempt to compile:

```
double myDouble;
System.out.println(myDouble);
```

What happens?

Good practice: Always initialize your variables.

Basic Operators

```
int a = 7;
int b = 5;
boolean myBool = false;
```

Examples (not exhaustive) of useful operators

```
boolean myBool = a>b;

a  =  a + 1;

a++;

a--;

b = a*b;

int c = 20/2;

c = 20/3;
```

Basic Operators

```
int a = 7;
int b = 5;
boolean myBool = false;
```

Examples (not exhaustive) of useful operators

```
boolean myBool = a>b;

a  =  a + 1;

a++;

a--;

b = a*b;

int c = 20/2;

c = 20/3;
```

What values do a, b, and c have after these lines of code are executed?

Basic Operators

```
int a = 7;
int b = 5;
boolean myBool = false;
```

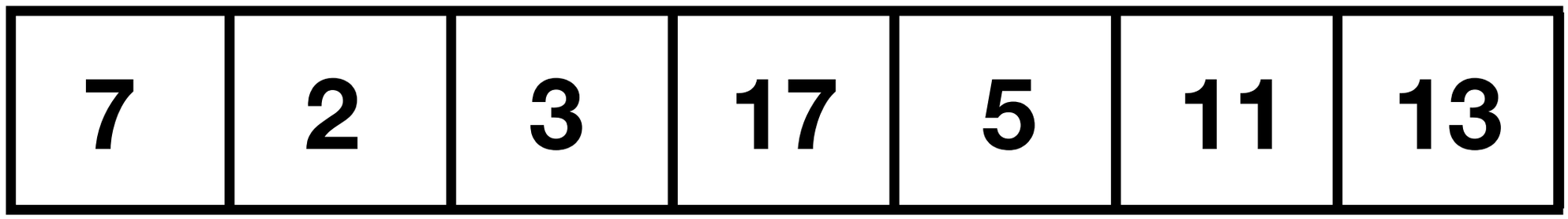
Examples (not exhaustive) of useful operators

```
boolean myBool = a>b;

a  =  a + 1;

a++;

a--;

b = a*b;

int c = 20/2;

c = 20/3;
```

What values do a, b, and c have after these lines of code are executed?

1D Arrays

Useful container is an array




```
int[] myArray = new int[7];
myArray[0] = 7;
myArray[1] = 2;
.
.
.
```

Can store other primitive types or objects

Creating a Class

Calculate the volume of a sphere: $V = (4/3)\ \pi\ r^3$

Will use two classes: Math and Double (not to be confused with double)

Create a new project called **`Sphere`**, and try to write the code where radius ***r*** is hardcoded

Creating a Class

Calculate the volume of a sphere: $V = (4/3)\ \pi\ r^3$

Will use two classes: Math and Double (not to be confused with double)

Create a new project called **`Sphere`**, and try to write the code where radius ***r*** is hardcoded

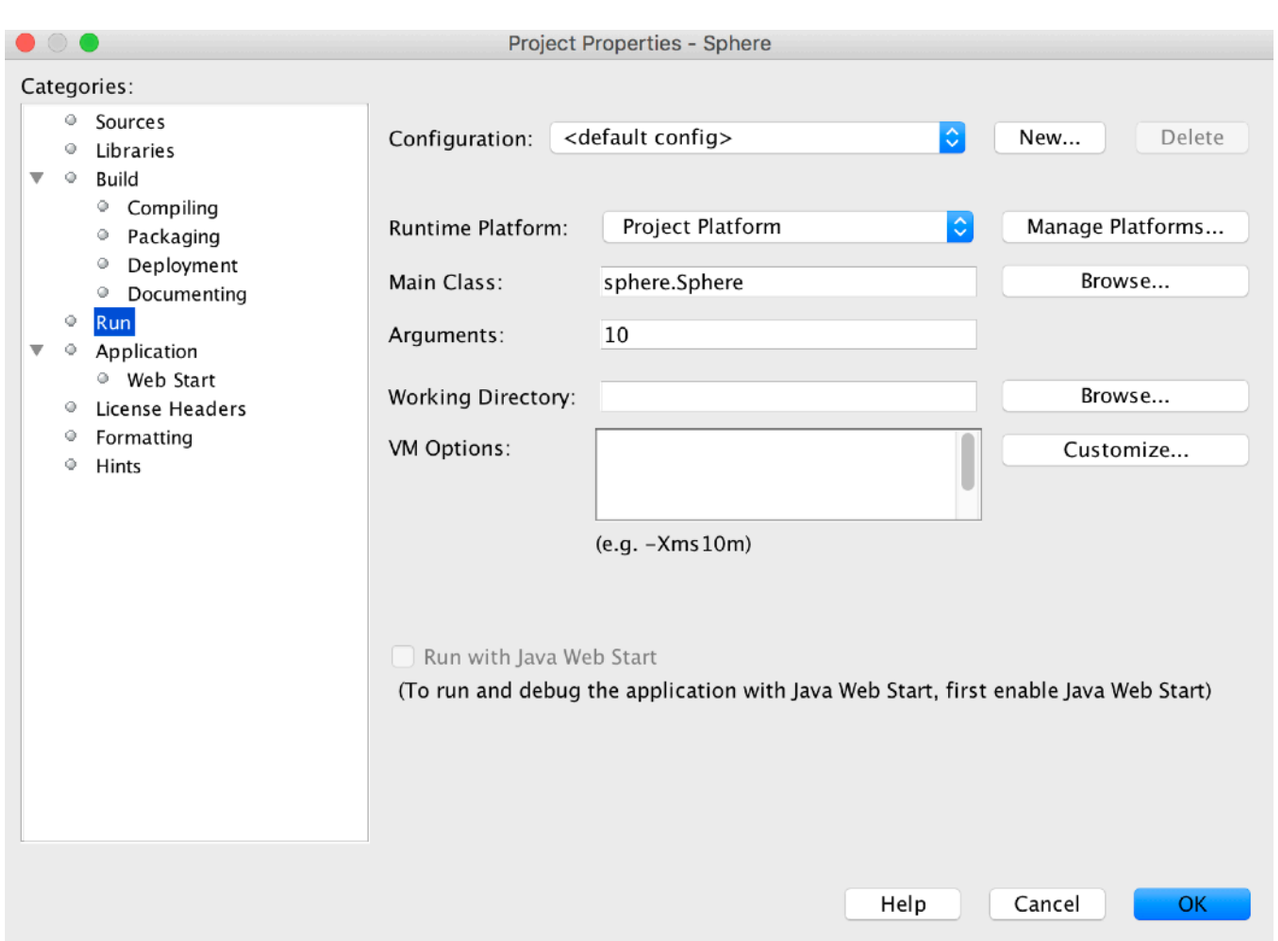
```
public class Sphere {

    public static void main(String[] args) {
        
        double r=10;
        double V = (4.0/3.0)*(Math.PI)*r*r*r;
        System.out.println("Volume of our sphere is " + V);
        // Math.pow(r,3.0);
    }

}
```

Creating a Class

Calculate the volume of a sphere: $V = (4/3)\ \pi\ r^3$

Will use two classes: Math and Double (not to be confused with double)

Create a new project called **`Sphere`**, and try to write the code for this where argument ***r*** is specified in **`args[0]`**



Creating a Class

Calculate the volume of a sphere: $V = (4/3)\ \pi\ r^3$

Will use two classes: Math and Double (not to be confused with double)

Create a new project called **`Sphere`**, and try to write the code for this where argument ***r*** is specified in **`args[0]`**

```
 */
public class Sphere {

    public static void main(String[] args) {

        String radius = args[0];  // take in first argument as radius
        double r = Double.parseDouble(radius);
        double V = (4.0/3.0)*(Math.PI)*r*r*r;
        System.out.println("Volume of our sphere is " + V);
        // Math.pow(r,3.0);
    }

}
```