Java Programming 1

PROFESSOR: CÂI FILIAULT

Topics for this week

Arrays

Purpose

Useage

As parameters

As Return types

Marty A. Theodore

NEW EMAIL

Email

Marty A. Theodore

Grade one teacher

Hello,

I would like you to build me a piece of software that allows me to enter the number of students in my class. Followed by the names of all my students one at a time. When I am finished it will ask me if I want the list in alphabetical or reverse alphabetical order. Then the program will display my class list the desired order.

-Thanks

Plan of Action

Create a data set that will be capable of storing the names of every person in a class.

Create a repetition structure that can store the names of ever user in the data set.

Output the list of users in the desired order.

Sorting and displaying a list of names could potentially be a useful piece of code that we could use again

We should build a function that is able to:

- Process a list of names
- Determine the order we want to sort and display the names

Plan of Action

How will we store everyone's name without knowing how many people are in a class?

How will we be able to store the values of each person in the class in a format that we will be able to access?

How do we sort values that are in each data set?

To answer these questions, we will need to learn a few new concepts:

Array Data Structure

An Array is what's known as a data structure.

- Data structures can be used to store large amounts of data in one convenient place.
- Arrays are structured so that each value is marked with an index number.
- Like the way each person in this room could have a different phone number that would uniquely identify them.
- When we create a new array, we must specify the number of items that we wish to store within the array.
- We also specify the data type that the array will be storing.
- *Note: we will use numeric index locations even if the data stored is another data type*

Array Data Structure

Arrays are very similar to variables:

- Both are used to store information.
- An array is like a variable that can store multiple pieces of information.
- Each piece of information is marked with a unique index number.
- Arrays start with index numbers at 0 and work their way up.
- The information in an array can be overwritten

There are three methods that can be used when creating an array:

Type[] arrayName = new Type[number of values to store]; (Most common method)

Type[] arrayName = {comma-delimited values};

Type arrayName[] = new Type[number of values to stored]; (less common method)

Let's look at how this is applied In a few different data types:

Array Data Structure - Strings

When dealing with an array that contains strings, we use the following format:

```
String[] list = new String[3];
String[] list = {“Good”, “Better”, “Best”};
String list[] = new String[3];
```

All three of these arrays create three memory locations. Recall we said that arrays are like variables that can store multiple pieces of information at the same time.

If we refer to the array create above, then we can see the following:

- List is the name of the memory location.
- Only when accessing list, we can access list[0], list[1], list[2]
- list[0] contains “Good” and list[2] contains “Best”

Array Data Structure - Strings

If we tried to access list[3] or list[4] then we would encounter an error like the following:

- Array out of bounds exception
- Meaning you are trying to access a memory location that does not exist.

Array Data Structure - Integers

When creating an array that will contain integers, we do the following:

int[] grades = new int[6];

int[] grades = {89, 10, 99, 100, 78, 55};

int grades[] = new int[6];

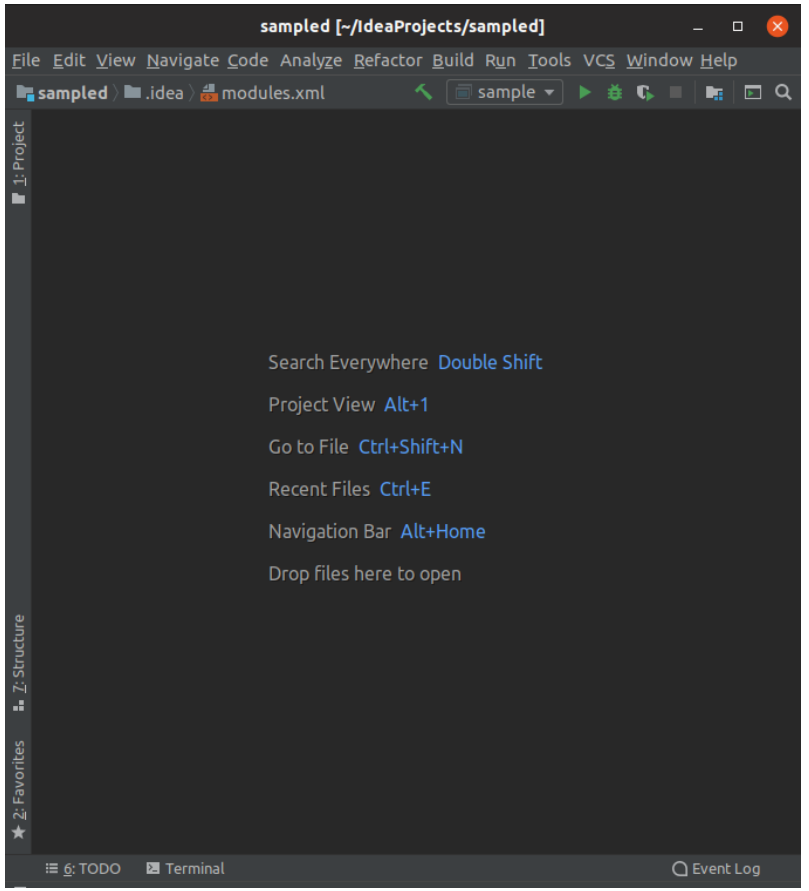
This same concept can be applied for booleans, characters, integers, Strings and every other data type covered within this class.

Let's build a quick example program that we can use to explain arrays and their functionality a little further.

Creating a new project

Start by opening IntelliJ and selecting the following:

File> New> Project



New Project

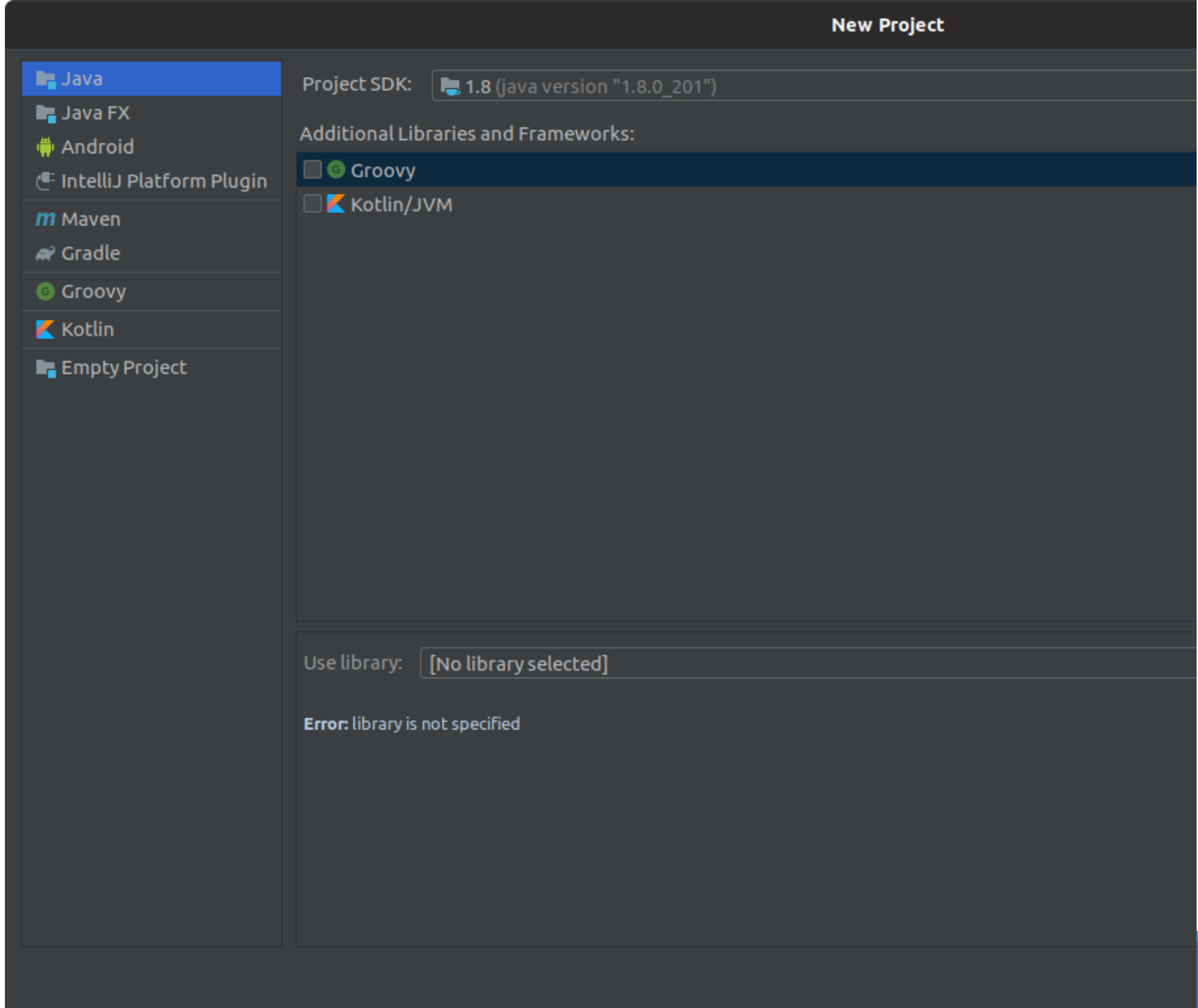
You will now see a dialog box appear *(It may be a little different than the one that I have shown)*

Starting on the left we will select a new Java Project

We will now want to select the project SDK of 1.8

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



New Project

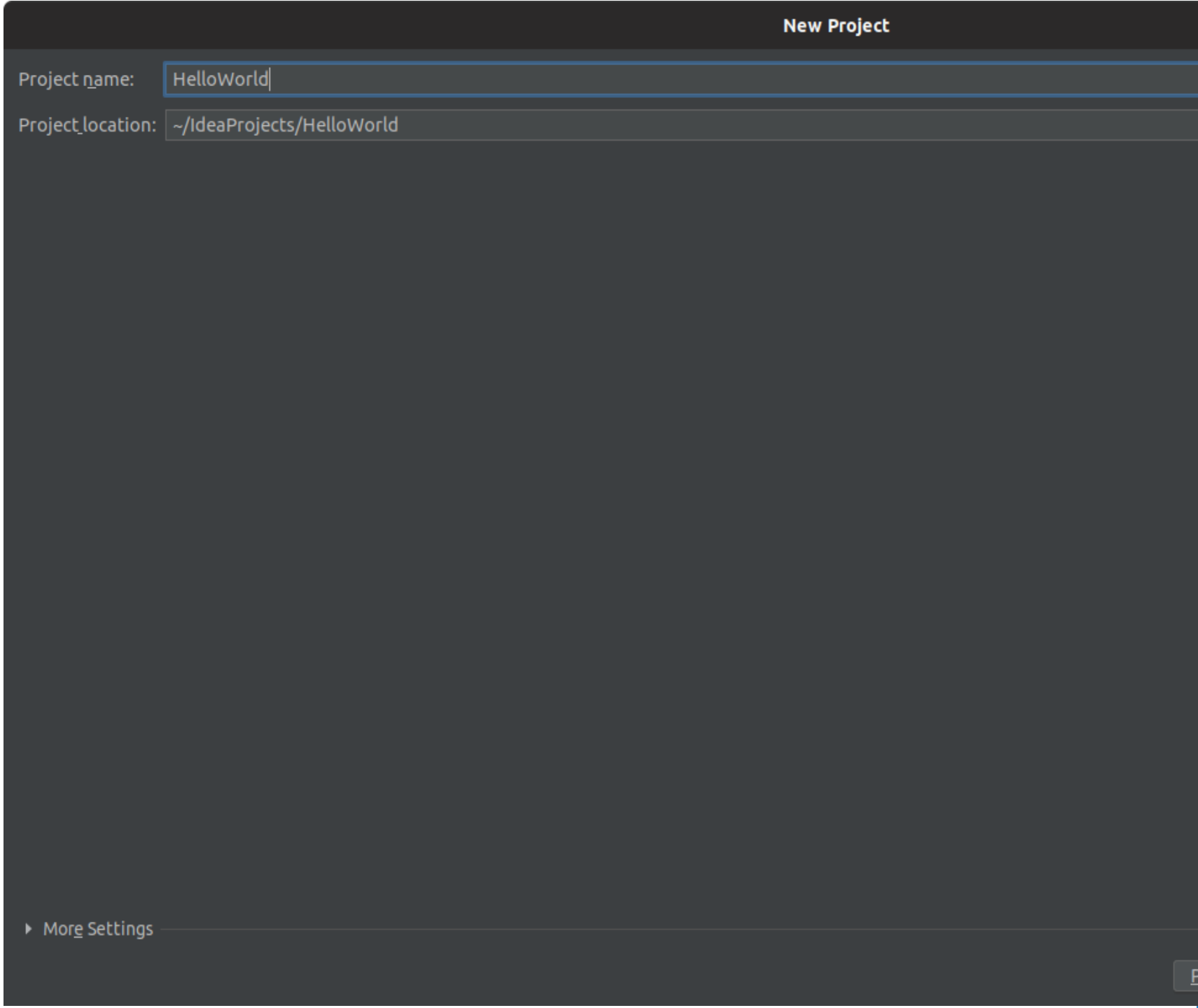
Give the project a name of:

ArrayTest

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.

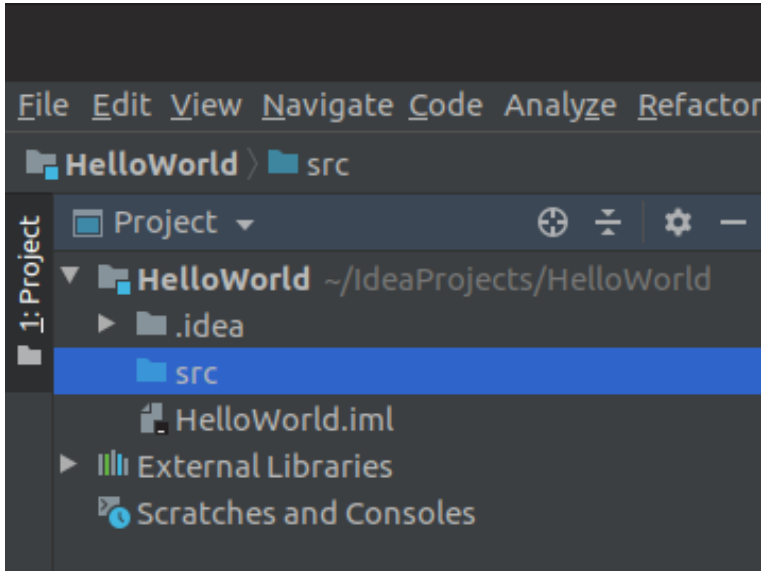


Creating a new project

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

Within the package explorer you should see the **ArrayTest** project.

Expand the **ArrayTest** project to see a folder named **src**.



Creating a new class file

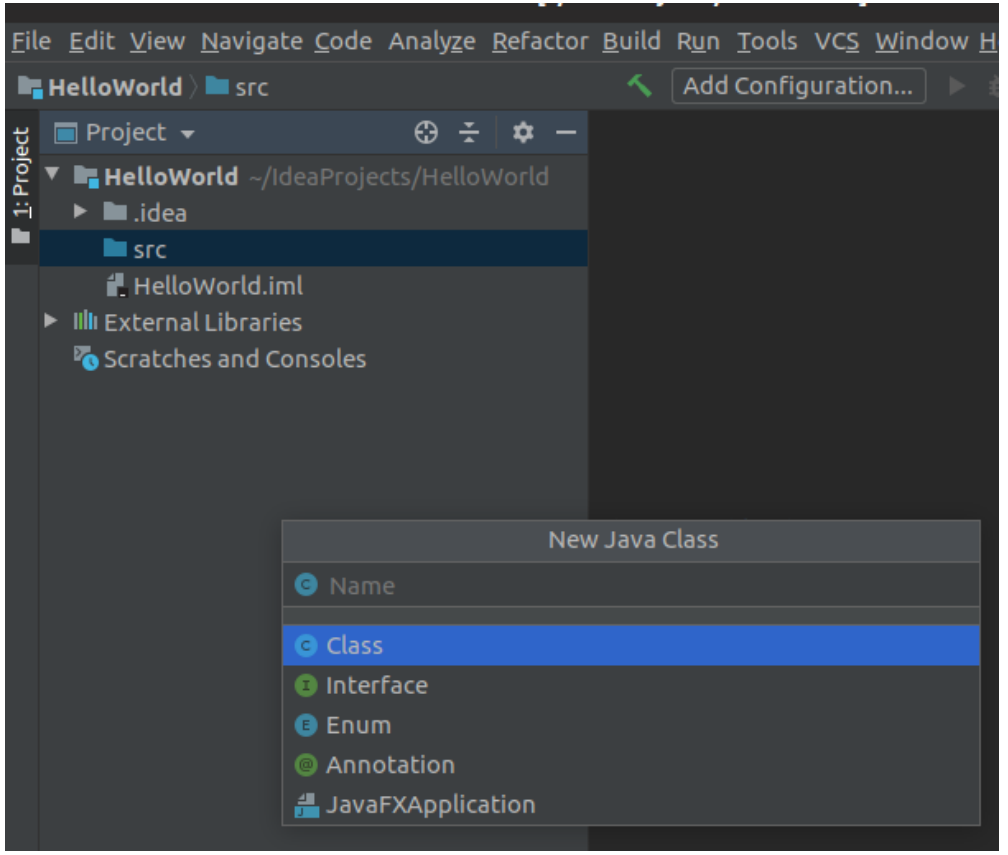
Next, we will need to create a new class file.

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

New> Java Class

Creating a new class is like creating a new program

Name the class **ArrayTest** and hit **enter**



Writing the program

You should now see ArrayTest.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"

Writing the program

```java
import java.util.Scanner;

public class ArrayTestProgram {

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        //when creating an array we have already established the following syntax
        int[] marks = new int[4];
        //this will create the locations marks[0], marks[1], marks[2], marks[3]
        //all of which are like individual variables that can each store a value
        //we can use the following syntax to populate each location in the array
        marks[0] = 99;
        marks[1] = 88;
        marks[2] = 77;
        marks[3] = 66;
        //just like a variable we can access those values at any time
        //lets take a look at how we can print those values
        System.out.println(marks[0]);
        //we can access the size of the array by using the following
        System.out.println("The marks array currently contains " + marks.length + " items");
        //notice how when we created the array we mentioned 4
```

Writing the program

```
//We can also create arrays with no predefined size
String[] words = {"Word", "baseball", "rat", "array", "school",
        "moo", "house", "dinosaur", "turtle"};

words[8] = "";
System.out.println(words[7]);
System.out.println(words.length);
```

Writing the program

```
//finally lets look at how we can output all items within an array
for(int i = 0; i < words.length; i++){
    System.out.print(words[i] + " ");
}

//just to ensure we are on a new line
System.out.println();

//lets create one more example where we populate the array through user input
int numberOfGrades;
System.out.println("How many grades are you about to enter: ");
numberOfGrades = input.nextInt();
//now create an array
int[] grades = new int[numberOfGrades];
//for each item in the array populate its value
```

Writing the program

```
        //for each item in the array populate its value
        for(int i = 0; i < numberOfGrades; i++){
            System.out.println("Enter the value for grades[" + i + "]:");
            grades[i] = input.nextInt();
        }
        //next output all the values
        for(int i = 0; i < numberOfGrades; i++){
            System.out.println(grades[i] + " ");
        }

    }

}
```

Writing the program

```
for(int grade: numberOfGrades) {
    System.out.println(grade + " ");
}
```

Creating a new class file

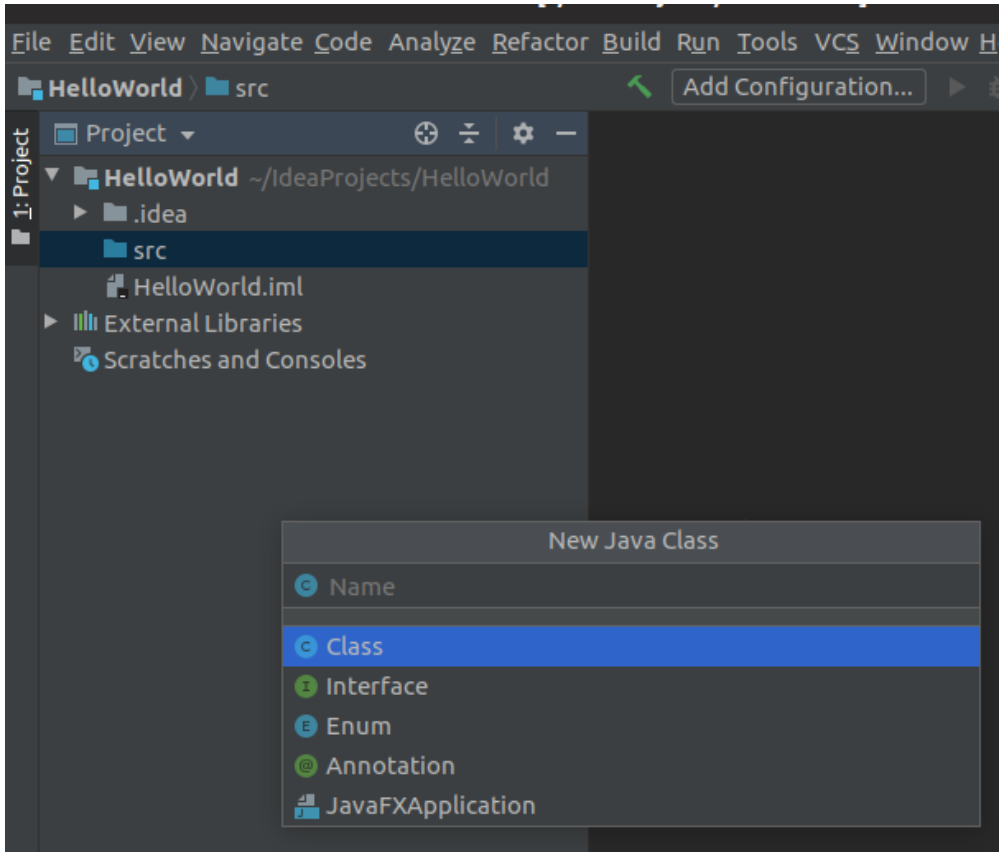
Let's begin working on Marty's Software by creating a new class file

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

New> Java Class

Creating a new class is like creating a new program

Name the class **ClassListGenerator** and hit **enter**



Writing the program

Begin by adding the following comments:

```
3  public class ClassListGenerator {
4
5      public static void main(String[] args) {
6          //Welcome the user to the software (not using the function we created last week)
7          //ask the user to enter the number of students in their class
8          //create an array to store all the names of the students
9          //create a repetition structure that will loop grabbing the users name and storing it in the array
10         //sort the array
11         //ask the user alphabetical or reverse alphabetical order
12         //display the list accordingly
13     }
14
15 }
16
```

Writing the program

- Let's start to create the program by declaring a function within the class declaration.
- The function will take 2 parameters
  - The array
  - The order in which we wish to display the list
- Note that we can have arrays in parameters
- We indicate the parameter is an array by adding [] to the end of the data type

```
3  public class ClassListGenerator {
4
5      public static void printSortedList(String[] list, int order, int size){
6
7      }
8
9      public static void main(String[] args) {
10         //Welcome the user to the software (not using the function we created last week)
```

Writing the program

- The next step was to create a repetition structure that will loop grabbing the student's names and storing them in the array one at a time.
- The number of students in the class is a known quantity.
- Anytime we are dealing with a determinant number of values we need to perform an action on we are building a for loop.
- Like 25 builds a for loop that utilizes the numberOfStudents variable to populate the names array

```
23          //create a repetition structure that will loop grabbing the users name and storing it in the array
24          System.out.println("Please enter each students name followed by the enter key");
25          for(int i = 0; i < numberOfStudents; i++){
26              names[i] = input.nextLine();
27          }
28
```

Writing the program

- The next step is to ask the user which order they would like to display the items.
- We will use a simple 1 or 0 to determine whether they want the values in ascending or descending order.
- Line number 30 asks the user to input the value
- Line number grabs the integer from the user and stores the value within the order variable

```
25          System.out.println("Please enter each students name followed by the enter key");
26          for(int i = 0; i < numberOfStudents; i++){
27              names[i] = input.nextLine();
28          }
29          //ask the user alphabetical or reverse alphabetical order
30          System.out.println("Do you want the list in alphabetical (1) or reverse alphabetical order(0):");
31          int order = input.nextInt();
```

Writing the program

- The next step is to ask the user which order they would like to display the items.
- We will use a simple 1 or 0 to determine whether they want the values in ascending or descending order.
- Line number 30 asks the user to input the value
- Line number grabs the integer from the user and stores the value within the order variable

```
25          System.out.println("Please enter each students name followed by the enter key");
26          for(int i = 0; i < numberOfStudents; i++){
27              names[i] = input.nextLine();
28          }
29          //ask the user alphabetical or reverse alphabetical order
30          System.out.println("Do you want the list in alphabetical (1) or reverse alphabetical order(0):");
31          int order = input.nextInt();
```

Writing the program

- The final step was the create the function so that we can utilize it to display out output
- Line number 9 sorts the array in ascending order.
- We then check if the user entered the value 1 or 0 for the order.
- If the user entered the value 1 then we want to display the contents of the array in ascending order
- Otherwise, we want to display the values in descending order
- Take a moment and try to figure out how we can display the contents in descending order (what would we place inside the else statement)

```
8   public static void printSortedList(String[] list, int order, int size){
9       Arrays.sort(list);
10      if(order == 1){
11          for(int i = 0; i < size; i++){
12              System.out.println(list[i]);
13          }
14      }
15      else{
16
17      }
18  }
```

Writing the program

- Once the values are sorted outputting the values in ascending or descending order is simple
- We can take the original sorted data set and traverse backwards
- If we start at the end (size-1) and then decrement "i" until we reach 0 then we will traverse all values backwards.

```
8   public static void printSortedList(String[] list, int order, int size){
9       Arrays.sort(list);
10      if(order == 1){
11          for(int i = 0; i < size; i++){
12              System.out.println(list[i]);
13          }
14      }
15      else{
16          for(int i = (size-1); i >= 0; i--){
17              System.out.println(list[i]);
18          }
19      }
20  }
```

Writing the program

- Finally at the end of the program run the function passing the array, order in which we wish to display the values and the number of students in the class.
- Line number 45 runs the function we created

```
45          printSortedList(names, order, numberOfStudents);
46          input.close();
47      }
48
49  }
50
```

Writing the program

- Save and compile the program
- What issues do you notice with the program?

Writing the program

- Your software is not running the correct number of times.
- Note that it is running the correct number of times but one of the words you entered were equal to nothing.
- When you asked for the user to enter the number of students input.nextInt()
- They entered something like the following:
- 45 *enter key*
- The input.next() only caught the number 45 and the enter key is still sitting in the keyboard buffer waiting to be read.

```
29          System.out.println("Enter the number of students in the class:");
30          int numberOfStudents = input.nextInt();
31          input.nextLine();
32          //create an array to store all the names of the students
33          String[] names = new String[numberOfStudents];
34
```

Writing the program

- The keyboard must first “escape” or read this enter key before it is able to read another word
- We can circumvent this issue through putting an additional input.nextLine() after grabbing the integer value (as seen on line 31)
- We don’t need to store this value as it is just being discarded.
- This will just stop the enter key from being entered when we prompt for the user to enter a name
- Update your code! Save and compile

```
29          System.out.println("Enter the number of students in the class:");
30          int numberOfStudents = input.nextInt();
31          input.nextLine();
32          //create an array to store all the names of the students
33          String[] names = new String[numberOfStudents];
34
```

Test your software

Take a moment to test your software

Homework

Read Chapter 7 of your textbook

Next Week

&&, ||, TypeCasting