

Introduction to Basic Java Program

OBJECTIVES:-

- Write, compile and run Java programs.
- Declare and define classes and objects
- Declare fields, methods, and constructors
- Read input from commandline

1. Install and set environment variables

In Linux variants like Ubuntu, we do not have to set PATH environment variable explicitly after installing the Java Development Kit (JDK) using the package manager. The package manager usually takes care of adding the necessary paths to the system's environment configuration.

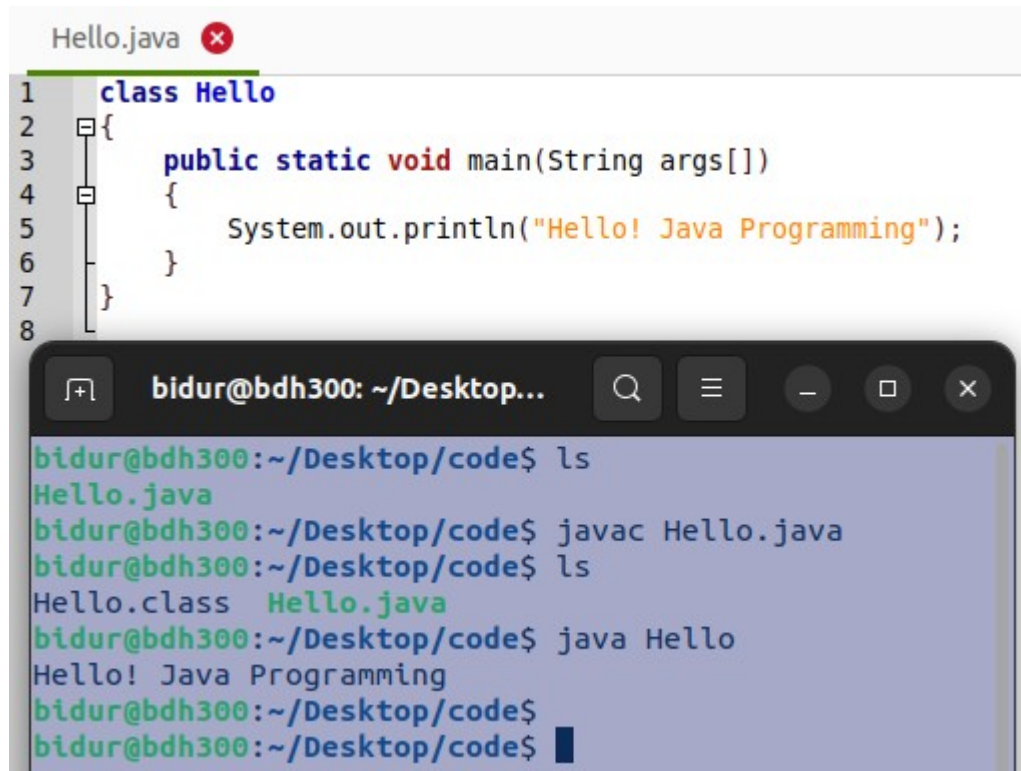
But on windows, it is necessary that we set the PATH environment variable so that we can run Java-related commands and compile Java programs from the command prompt without specifying the full path to the Java executables each time.

<https://www.youtube.com/watch?v=FVxKbAukRxk>

General syntax of java program:

```
access-specifier class ClassName {  
    // fields  
    fieldType fieldName;  
  
    // methods  
    public returnType methodName() {  
        statements;  
    }  
}
```

1. Now, write your code in a text-editor (e.g. sublime text, notepad++):



The screenshot shows a text editor window titled 'Hello.java' with the following code:

```

1  class Hello
2  {
3      public static void main(String args[])
4      {
5          System.out.println("Hello! Java Programming");
6      }
7  }
8

```

Below the editor is a terminal window with the following commands and output:

```

bidur@bdh300: ~/Desktop/code$ ls
Hello.java
bidur@bdh300: ~/Desktop/code$ javac Hello.java
bidur@bdh300: ~/Desktop/code$ ls
Hello.class  Hello.java
bidur@bdh300: ~/Desktop/code$ java Hello
Hello! Java Programming
bidur@bdh300: ~/Desktop/code$
bidur@bdh300: ~/Desktop/code$

```

2. Save the program in Desktop/code/ folder with the name Hello (same as the name of class) and Extension (.java) for e.g. Hello.java

3. In order to compile and run the program, use javac and java commands as shown above.

After you compile the program using javac, it converts the java source code into binary program consisting of byte codes.

javac Hello.java

If the program contains no errors, the compiler generates a *byte code program (.class file)* from your source file. The compiler stores the byte code program in a file with the same name as source file, but with the extension .class.

The Java interpreter inspects the byte code and checks it to ensure that the security restrictions are met and then execute the program within the java virtual machine.

java Hello

If there is no exception in the program the output is printed on the command prompt as shown above.

System.out.print command in java is same as **cout** in C++ . **System.out.println** command is similar to **System.out.print** but also prints a newline as well. Run the following programs and observe the output.

Program#2 : WAP to demonstrate Class called Motorbike. Provide its properties and behavior (method) as well.

```
Motorbike.java
1 public class Motorbike {
2
3     int speed;
4     String model;
5
6     public Motorbike(String model) {
7         this.model = model;
8     }
9
10    public void accelerate() {
11        // add 1 km/hr
12        speed = speed + 1;
13    }
14
15    public void stop() {
16        // set current speed to zero
17        speed = 0;
18    }
19
20    public void printSpeed() {
21        // display the current speed of this car
22        System.out.println("The current speed of " + model + " is " + speed + " mph");
23    }
24
25    public static void main(String[] args) {
26        // create new Honda car
27        Motorbike honda = new Motorbike("Honda");
28        // create new Yamaha car
29        Motorbike yahama = new Motorbike("Yahama");
30
31        // print current speed of Honda
32        honda.printSpeed();
33
34        // call the accelerate method twice on Honda
35        honda.accelerate();
36        honda.accelerate();
37
38        // call the accelerate method once on Yamaha
39        yahama.accelerate();
40
41        // print current speed of Honda
42        honda.printSpeed();
43        // print current speed of Yamaha
44        yahama.printSpeed();
45
46        // now park the Honda car
47        honda.stop();
48
49        // print current speed of Honda
50        honda.printSpeed();
51    }
52 }
```

Program#3

Program to read an integer from console and display it

```

ReadData.java
1  import java.io.*;
2
3  public class ReadData {
4
5
6      public int getInteger(){
7          System.out.println("Write an Integer number:");
8          String line = null;
9          int value = 0;
10         try {
11             BufferedReader is = new BufferedReader(new InputStreamReader(System.in));
12             line = is.readLine();
13             value = Integer.parseInt(line);
14         } catch (Exception e) {
15             System.err.println("Unexpected IO ERROR: " + e);
16         }
17         // System.out.println("I read this number: " + value);
18
19         return value;
20     }
21
22     public static void main(String[] args) {
23
24         ReadData c = new ReadData();
25
26         System.out.println(" Integer: " + c.getInteger());
27
28     }
29
30 }
31

```

Modify this program so that you can input two integers and display the sum.

Program#4

Write a program with a class which saves the data for a bank account.

```

1  import java.util.Scanner;
2  public class BankAccount
3  {
4      String id;
5      float balance;
6      int transactionCount;
7      String name;
8
9      public BankAccount(String id, float balance, String name )
10     {
11         this.id = id;
12         this.balance = balance;
13         this.transactionCount = 0;
14         this.name = name;
15     }
16
17     public void readAccountDetails()
18     {
19         Scanner b = new Scanner (System.in);
20         System.out.println("Enter name: ");
21         name = b.nextLine();
22         System.out.println("Enter id: ");
23         id = b.nextLine();
24         System.out.println("Enter balance: ");
25         balance = b.nextFloat();
26
27     }
28
29     String getAccountDetails()
30     {
31         return("ID: " + id + " Balance: " + balance + "Name: " + name );
32     }
33
34     void deposit(float amount)
35     {
36         balance = balance + amount;
37         transactionCount++;
38     }
39
40     void withdraw(float amount)
41     {
42         balance = balance - amount;
43         transactionCount++;
44     }
45
46     public static void main(String[] args) {
47         BankAccount acc1 = new BankAccount("", 1000 , "" );
48         acc1.readAccountDetails();
49         System.out.println( "Details \n"+ acc1.getAccountDetails());
50         acc1.deposit(50);
51         acc1.getAccountDetails();
52         acc1.withdraw(25);
53         acc1.getAccountDetails();
54     }
55 }
56

```

Program#5

Write a program to count the number of instances of a class using class variable(i.e. static variable).

```
ObjectCountDemo.java x
1
2 class SampleClass {
3
4     // Set count to zero initially.
5     static int count = 1;
6     float data;
7
8
9     public SampleClass() {
10
11         // increment the count on each call to the constructor via any instance
12         count = count + 1;
13
14         System.out.print(" number: " + count);
15     }
16 }
17
18 public class ObjectCountDemo {
19
20     public static void main(String[] args) {
21         SampleClass stuff1 = new SampleClass();
22         stuff1.data = SampleClass.count;
23         SampleClass stuff2 = new SampleClass();
24         stuff2.data = SampleClass.count;
25         stuff2.data += stuff1.data;
26         System.out.println("Final Data: " + stuff2.data + " " + stuff1.data );
27     }
28 }
29
30 }
31
```

Program#6

Write a program that reads two numbers from the command line, the number of hours worked by an employee and their base pay rate. Add warning messages if the pay rate is less than the minimum wage (12.5 an hour) or if the employee worked more than the number of hours in a week.

Compile: <javac Salary.java>

Run: <java Salary 30 50>

```
Salary.java x
1  class Salary {
2
3  public static void main (String[] args) {
4
5      double hours = Double.valueOf(args[0]).doubleValue();
6      double rate = Double.valueOf(args[1]).doubleValue();
7      double pay;
8
9      pay = rate * hours;
10
11     System.out.println("The paycheck is " + pay + " dollars.");
12     if ( rate < 12.5) {
13         System.err.println("This employee is not getting the legally required minimum wage.");
14     }
15     if ( hours > 7*24) {
16         System.err.println("Did this employee really work " + hours + " hours?");
17     }
18
19 }
20
21 }
22
```