**JAVA**

1. Steps to install Java?

Step 1

Goto link. Download click on download JDK.

Step 2

1. Accept License Agreement.
2. Download latest java JDK for your version (32 or 64 bit) of Java for windows.

Step 3

Once the download is complete, run the exe to install JDK. Click Next-Next-Next.

Step 4

Once the installation is complete click close.

And then the Java development kit is installed in your windows. You will see your Java folder contains two folders jdk 8.0\_91 and jre 8.0\_91

The process is not yet over. Before you could compile and run your java programs you need to set PATH and CLASSPATH variables. (Are you wondering what these are? Be patient you will know soon)

Step 5

We start with setting the PATH and then the CLASSPATH variables for JAVA in our windows.

1. Right click on This PC (or my computer) and select properties option.
2. Now click on ADVANCED SYSTEM SETTINGS.
3. In the next window that appears, under the Advanced tab click on Environment Variables.
4. Under System Variable on Environment Variables Window, select Path variable and click on Edit.
5. Now you will have Edit environment variable window, click New.

Once you click New, use browse option to browse to the location to bin folder. Click OK. This sets the PATH variable.

NOTE: if you have some other version of windows you will have to append this location to PATH variable. You can do this by simply putting (;) at the end of the current variable value and then add the location of bin folder to it. Make sure you don’t alter other specified locations.

1. Now to set CLASSPATH variable, under the System Variables click on New.
2. Under New System Variable window, specify Variable name and Variable Values as follows.

myClasses is a folder created by me in jdk folder. Don’t forget to write “;”

1. Click OK-OK-OK and you are done.

2.What is main method will do?

package helloWorld;

public class HelloWorldApp { public static void **main**(String[] args) { System.out.println("Hello World!"); // Display the string.

} }

It is the entry point for any **java** program, it **does** whatever you tell it to **do**. When you execute your class, anything in the **main method** runs. Whatever the statements written within the main are executed.

3.Creating property/data members?

Properties is a subclass of Hashtable. It is used to maintain lists of values in which the key is a String and the value is also a String. The Properties class is used by many other Java classes. For example, it is the type of object returned by System.getProperties( ) when obtaining environmental values.

Example:

**package** Example;

**import** java.util.Iterator;

**import** java.util.Properties;

**import** java.util.Set;

**public** **class** propDemo {

**public** **static** **void** main(String[] args)

{

Properties capitals = **new** Properties();

String str;

Set states;

capitals.put("Illinois", "Springfield");

capitals.put("Missouri", "Jefferson City");

capitals.put("Washington", "Olumpia");

capitals.put("California", "Sacramento");

capitals.put("indiana", "Indianapolis");

//show all states and capitals in hashtable

states = capitals.keySet(); //Get set-view 0f keys

Iterator itr = states.iterator();

**while**(itr.hasNext())

{

str = (String)itr.next();

System.***out***.println("The capital of "+ str +" is "+ capitals.getProperty(str) );

}

System.***out***.println();

//look for the state which is not in the list --specify default

str = capitals.getProperty("Florida"," not found");

System.***out***.println("The capital of Florida is" + str);

}

}

Output:

The capital of Missouri is Jefferson City

The capital of Illinois is Springfield

The capital of California is Sacramento

The capital of Washington is Olumpia

The capital of indiana is Indianapolis

The capital of Florida is not found

3.What is a variable?

A **Java variable** is a piece of memory that can contain a data value. A **variable** thus has a data type. Data types are covered in more detail in the text on **Java** data types. **Variables** are typically used to store information which your **Java** program needs to do its job.

There are three kinds of variables in Java −

* Local variables

Here when the variables are declared these variables are used inside of the same method.

Example:

**package** samplelocalvarprgm;

**public** **class** Test {

//declaring a local variable

**public** **static** **void** pupage()

{

**int** age = 0;

age = age+1;

System.***out***.println("My puppy age is " +age+ ".");

}

**public** **static** **void** main(String[] args) {

//Using the above declared local variable in the main function by calling pupage() through object

Test test = **new** Test();

test.*pupage*();

}

}

Output:

My puppy age is 1.

* Instance variables

These are declared in a class, but outside a method or constructor or any block and are visible for these three. Instance variables are created when an object is created with the use of the keyword 'new'. Instance variables can be accessed directly by calling the variable name inside the class. Access modifiers can be given for instance variables.

Example:

**package** example;

**public** **class** Employee {

//This instance variable is visible inside any child class

**public** String name;

//Salary variable is visible only in Employee class only

**private** **static** **double** *salary*;

**private** **static** **int** *ID*;

//The Name variable is assigned in the constructor

**public** Employee(String empName)

{

name = empName;

}

//The salary variable and ID variable is assigned to value

**public** **static** **void** setDetails(**double** empSal, **int** empID)

{

*salary* = empSal;

*ID* = empID;

}

**public** **void** printEmp()

{

System.***out***.println("Emploee Name is " +name);

System.***out***.println("Emploee salary is " +*salary*);

System.***out***.println("Emploee ID is " +*ID*);

}

**public** **static** **void** main(String[] args)

{

Employee empOne = **new** Employee("Sneha");

empOne.*setDetails*(1000, 1);

empOne.printEmp();

}

}

Output:

Employee Name is Sneha

Employee salary is 1000.0

Employee ID is 1

* Class/Static variables

Class variables also known as static variables are declared with the static keyword in a class, but outside a method, constructor or a block. There would only be one copy of each class variable per class, regardless of how many objects are created from it. Static variables are rarely used other than being declared as constants. Constants are variables that are declared as public/private, final, and static.

Example:

**package** Example;

**public** **class** Employee {

//Declaring variable salary as private static

**private** **static** **double** *salary*;

//Declaring Department variable as public static

**public** **static** **final** String ***Department***= "Development ";

**public** **static** **void** main(String[] args) {

*salary* = 1000;

System.***out***.println(***Department*** + "average salary " +*salary*);

}

}

Output:

Development average salary 1000.0

4.Creating method with void?

Method is a collection of statements that are grouped together to perform an operation. When you call the System.out.**println()** method, for example, the system actually executes several statements in order to display a message on the console.

Example:

**package** Parameters;

**public** **class** Example {

**public** **static** **void** main(String[] args)

{

//passing parameters with method1

*welcome*("Sneha");

*welcome*( "Anvesh");

// parameters with method2

*Add*(20,8);

*Add*(100,200);

//parameters with method3

*add*(10,20,30);

**int** value = *add*(10,20,30);

System.***out***.println("value:" +value);

}

//method1 with type void

**public** **static** **void** welcome(String name)

{

System.***out***.println("name:" +name);

}

//method2 with type void

**public** **static** **void** Add(**int** a, **int** b)

{

**int** c;

c = a+b;

//System.out.println("Value:" +c);

System.***out***.println(a+b);

}

//Method3 with return type

**public** **static** **int** add(**int** a, **int** b, **int** c)

{

**int** d;

d = a+b+c;

**return**(d);

}

}

Output:

name:Sneha

name:Anvesh

28

300

value:60