Core Java

- Java language is used for developing platform independent applications
- Java language is developed by the Software engineer s of Sun Micro Systems and take over by oracle.
- Secured language
- Built-in Exception Handling & Multithreading
- JAVA's INNOVATIVE FEATURES

1. JAVA is Multithreaded
2. JAVA is secured
3. **JAVA is an Object Oriented Language**
4. JAVA is distributed

Core Java Agenda:

1. **Data types**
2. **Operators**
3. **Variables**
4. **Arrays**
5. **Conditions**
6. **Loops**
7. **Branching statements (Break, continue , return)**
8. **Built in methods**
9. **Collections (Dynamic Array)**
10. **Methods & types**
11. **Oops concepts**
12. **File handling(XLS,Properties,DB)**
13. **Regular Expressions**
14. **Exceptions**

Installation Steps:

1. Download Jdk1.8

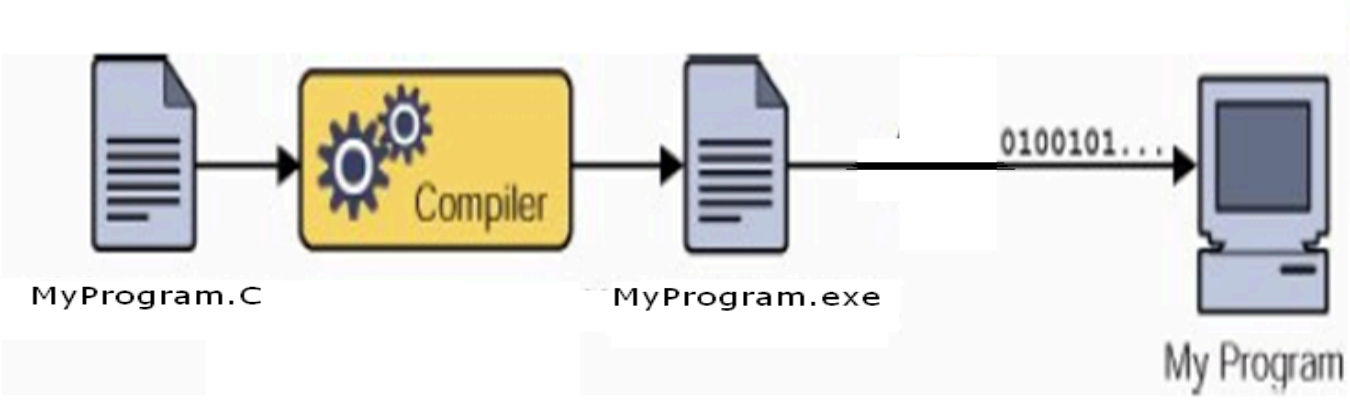
2. Install in local machine

Setup Environment Variables:

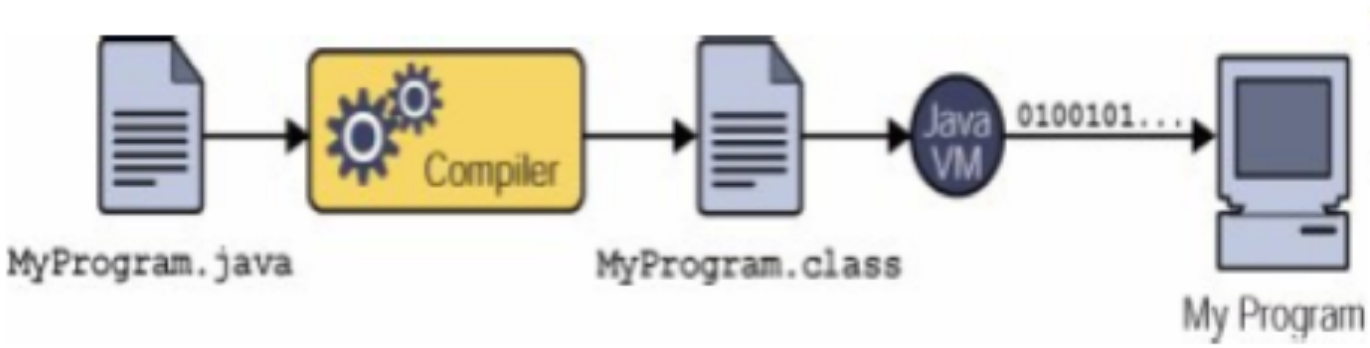
Follow the steps

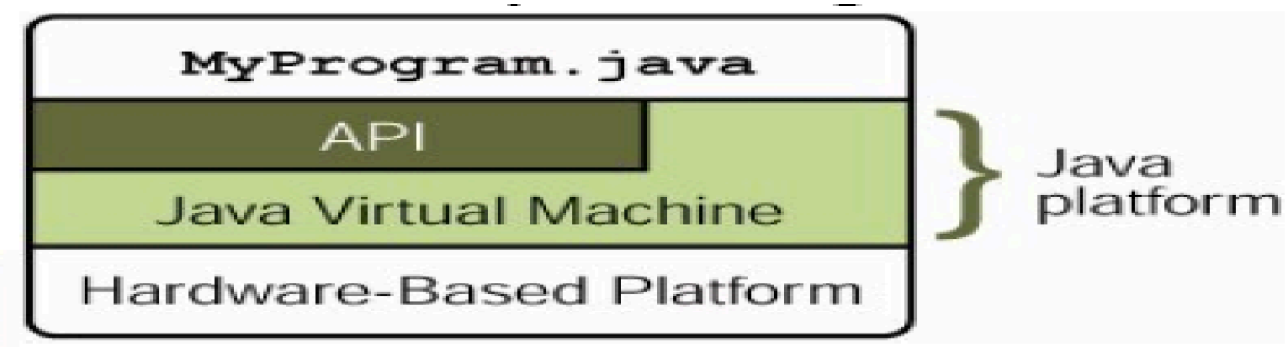
1. Right click on my computer icon of desktop of our computer.
2. Select properties.
3. Select "advanced" tab
4. Click on Environment Variables button.
5. Focus on System variables
6. Focus of Path Variable
7. Click on Edit to create a variable. Here variable is a pair of values name and value.
8. PATH: ;C:\Program Files\Java\jdk1.8.0_14\bin;
9. Click on Apply & OK

IN C Language:

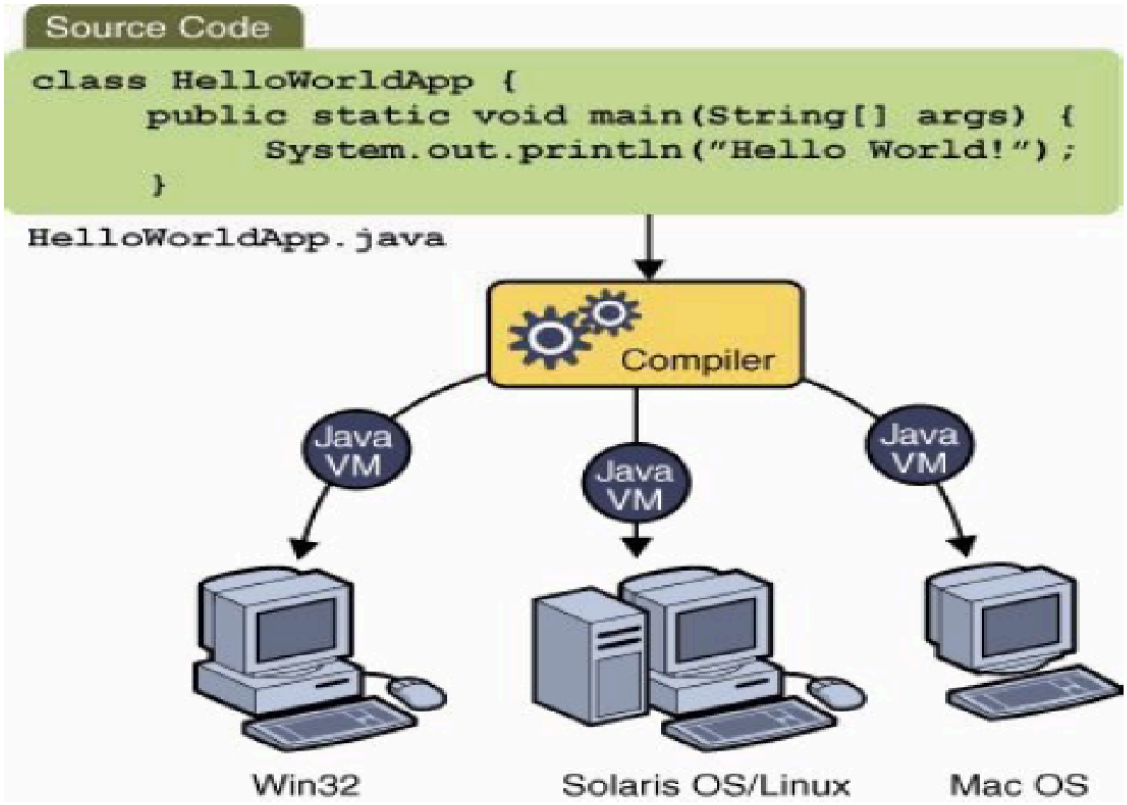


In Java


Compiler
Java VM
0100101...
MyProgram.java
MyProgram.class
My Program


MyProgram.java
API
Java Virtual Machine
Hardware-Based Platform
Java platform

Execution Process of Java in all platforms:



Creating Java Application:

1. creating source code

   EX:- ClassName.java

2. compiling source using : Using command prompt

   javac <sourcefilename>

   EX:- javac ClassName.java

3. Interpretation using java.exe : Using command prompt

   EX:- java <ClassFileName>

Java classname

Class:

A class is a storage area, a class is a way of creating a new data type. Once it is created we can use them for creation of objects.

This will be done using "class" keyword.

Main method:

It is a starting point of execution.

Syntax for main method:

```
Class Ex1
{
        public static void main(String[]  args)
        {
                ------
        }
}
```

Example 1:

```
class  sample
{
        public static void main (String[] args)
        {
                System.out.println("Good Morning");
        }
}
```

Introduction to Eclipse:

Eclipse is an open source IDE (Integrated development environment).

Configuration:

1. Install JDK
2. Download Eclipse
3. Unzip Eclipse zip file
4. Open Eclipse
5. Create workspace
6. Click on ok
7. Click on workbench

Navigation to Create Projects

1. Select Menu item File
2. Select New
3. Click on Project ->Java - > java project
4. Enter project Name
5. Click on finish

How to change Java Run time Environment:

1. right click on java project
2. Click on properties
3. Click on java build path
4. Click on libraries tab
5. Select JRE system library and click on Edit
6. Select alternate JRE radio button
7. Click on installed JRE button
8. Select jre and click on Edit
9. Click on directory
10. Select the JDK folder
11. Click on Finish

Navigation for creating Class File:

1. Right click on src
2. Go to New
3. Select class
4. Enter class name
5. Click on Finish

```
public class Ex
{
   public static void main(String[] args)
   {
        System.out.println("Good morning");
   }
}
```

Data Types:

A data type is a keyword and it is used for performing three things.

They are

1. It is used for allocation of fixed size memory.

2. It is also used for specifying type of information to be stored in that memory.

3. It is also used for giving name (variable) to that memory for further usage throughout the program.

Example:

Student - name - "xyz“ - String x=”Ratnam”

age - 25 - int age = 25;(4bytes)

rollno - 1000 - int r = 1000;

height - 5.4 - float h=5.4;(4bytes)

true/false -boolean b=true;

- A value type is used for the declaration of values directly.

Value types provided by Java are

Boolean	- 1-bit (true/false)
byte	**- 8-bit**
short	- 16 - bit
int	- 32-bit
long	- 64-bit
float	- 4-bytes
double	- 8-bytes
String	- depends up on no. of characters.

Variable Naming Rules: Assign value to the var

1. Var name must be starts with alphabets or _ or $ (Ex: a or _a or $a)
2. Var name should not contain embedded period (spaces, spl char except "_"and "$")

//**********

Operators:

We are using to perform Assignment, Arithmetic, Comparisons and logical operations.

The Simple Assignment Operator:

The simple assignment operator "=".

The Arithmetic Operators:

+ , - , * , / , % (MOD)

The Increment & Decrement Operators

++ Increment operator; increments a value by 1

-- Decrement operator; decrements a value by 1;

```
//-------------------------------------------------------------------------

//Increment Operator (++): increment by 1
/*
 Two ways
 1. Post increment
 2. Pre increment
 */
//-----------------------------------------
int x=10;
System.out.println(x); //10
//Post increment
System.out.println(x++); //10
System.out.println(x); //11
//Pre increment
System.out.println(++x); //12
//*************************************************************
//Decrement Operator (--): decrement by 1
/*
Two ways
1. Post Decrement
2. Pre Decrement
*/
//-----------------------------------------
int x=10;
System.out.println(x); //10
//Post Decrement
System.out.println(x--); //10
System.out.println(x); //9
//Pre Decrement
System.out.println(--x); //8
//*************************************************************
```

The Equality and Relational Operators

==	Equal to
!=	not equal to
>	Greater than
>=	greater than or equal to
<	Less than
<=	less than or equal to

The logical Operators

&& Conjunction - AND

|| Disjunction - OR

! Negation – Not

```
//*****************************************************************
//Conjunction
	System.out.println(true && true); //true
//Disjunction
	System.out.println(false || true); //true
//Negation
	System.out.println(!false); //true
//*********************************************
```

Conditional Operator:

?: Ternary (shorter form of **if-then-else** statement)

```
//*************************************************************************

int x=101;
System.out.println((x%2==0)?"Even":"ODD");

//*************************************************************************
```

Arrays:

Array is collection of similar elements

Array index starts from zero

Declaring a Variable to Refer to an Array

int[ ] a;

Similarly, you can declare arrays of other types:

byte[ ] a;

short[ ] a;

```
long[ ] a;
float[ ] a;
double[ ] a;
boolean[ ] a;
char[ ] a;
String[ ] a;
```

Example1:

```
public class array
{
        public static void main(String[] args)
    {
            int[] anArray;              // declares an array of integers
            anArray = new int[4];       // allocates memory for 4 integers
            anArray[0] = 100;
            anArray[1] = 200;
            anArray[2] = 300;
            anArray[3] = 400;
            System.out.println("Element at index 0: " + anArray[0]);
            System.out.println("Element at index 1: " + anArray[1]);
            System.out.println("Element at index 2: " + anArray[2]);
            System.out.println("Element at index 3: " + anArray[3]);
           System.out.println("Length of Array:="+anArray.length);
```

```
        }

}
```

Example 2:

```
public static void main(String[] args)
     {
          String[] a={"sree","ram","jon","dinu"};
          System.out.println(a[0]);
          System.out.println(a[1]);
          System.out.println(a[2]);

          System.out.println(a.length);

     }
```

Exmaple 3: To add different type of data into an array

```
Object[] a={"sree",10,true,5.4};

          System.out.println(a[0]);
          System.out.println(a[1]);
          System.out.println(a[2]);

          System.out.println(a.length);
```

Multidimensional Arrays:

Examples

```
     public static void main(String[] args)
     {
          String[] a={"Mr.","Ms."};
          String[] b={"sreedhar","sreerama"};
          String[] c={"Usha","latha"};

          String[][] x={a,b,c};

          System.out.println(x[0][1]+x[2][1]);
```

```
        System.out.println(x[0][0]+x[1][1]);
    }
```

Example 2

```
class MultiDimArrayDemo
{
 public static void main(String[] args)
{
 String[][] names = {{"Mr. ", "Mrs. ", "Ms. "}, {"Smith", "Jones"}};
        System.out.println(names[0][0] + names[1][0]); //Mr. Smith
        System.out.println(names[0][2] + names[1][1]); //Ms. Jones
 }
 }
```

Control structure:

1. Conditional statements

2. Looping statements

3. Branching statements

//**********

Conditional statements:

If, if -else, if-else-if, Nested If, switch

//**********

```
// if
            if(Condition)
            {
                  Block code;
            }

// if – else
            if(Condition)
            {
                  Block code;
            }
            Else
            {
                  Else Block code;
            }
```

```
      //Verify given number is even or odd
            int x=101;
            if(x%2==0)
            {
                  System.out.println("Even number");
            }
            else
            {
                  System.out.println("ODD number");
            }
```

```
// if - else - if: To compare more than one single condition

          String x="C";
          if(x=="A")
          {
               System.out.println("QA");
          }
          else if(x=="B")
          {
               System.out.println("Java");
          }
          else if(x=="C")
          {
               System.out.println(".net");
          }
          else
          {
               System.out.println("Invalid");
          }

// Nested If: Within one if condition, we can apply one more if
condition
          int x=10;
          if(x<=10)
          {
               if(x<=5)
               {
                    System.out.println("Less than 5");
               }
               else
               {
                    System.out.println("Less than 10");
               }
          }
          else
          {
               System.out.println("Greater than 10");
          }
```

// Switch : Alternative of if-else if

```
public class switchcondition
{
     public static void main(String[] args)
     {
          String x="B";
          switch (x)
          {
          case "A":
               System.out.println("Selenium");
               break;
          case "B":
               System.out.println("QTP");
               break;
          case "C":
               System.out.println("Selenium");
               break;
          default:
               System.out.println("Inalid");
               break;
          }

     }
}
```

****

```
      int month =2;
     String monthString;
     switch (month)
   {
        case 1:
             monthString = "January";
             break;
```

```
        case 2:
            monthString = "February";
            break;
        default:
            monthString = "Invalid month";
            break;
    }
    System.out.println(monthString);
```

//**********

Looping Conditions:

For, while, do-while, for – each

// for loop statement: whenever we know how many iterations we want to do.

//Whenever we know upper limit.

/* /**********

```
for (initialization; Condition; increment/decrement)
{
Code;
}
```

Ex1: Print values from 1 to 5
Ex2: Print values from 5 to 1
Ex2: Print factorial of a given number
Ex3: Print all values from an array

```
int[] a={10,20,30,40,50};
```

```
/******Print 1 to 5  values ************************************/
for(int i=1; i<=5; i++)
{
	System.out.println("value  is: " + i);
}

//*****************************************************************

//Print factorial for given number

//Way 1:

public static void main(String[] args)
	{
		int x=5;
		int f=1;
		for(int i=1;i<=x;i++)
		{
			f=f*i;
		}
		System.out.println(f);
	}

//Way 2:

int x=5;
		int f=1;
		for(int i=x;i>=1;i--)
		{
			f=f*i;
		}
		System.out.println(f);

//*************************************************************

//Print all values from array, and also verify 6 value in array

		int[] a={10,20,30,6,40,50};
```

```
for(int i=0;i<a.length;i++)
{
     System.out.println(a[i]);
     if(a[i]==6)
     {
          System.out.println(6+" found at index: "+i);
          break;
     }
}
```

//******************While Loop**********************************

It is a condition based loop, it will verify for condition at beginning of loop, if condition is true then it will go for execution else it will stop.

```
While (Condition)

{

Code;

}
```

/*******Print 1 to 5 values ************************************/

```
int i=1;
while(i<=5)
{
     System.out.println(i);
     i++;
}
```

//***

//Print factorial for given number

//Way 1:

```
int x=5;
int i=1;
int f=1;
while(i<=x)
{
     f=f*i;
     i++;
}
System.out.println(f);
```

//Way 2:

```
int i=5;
int f=1;
while(i>=1)
{
     f=f*i;
     i--;
}
System.out.println(f);
```

/*----------

The Java programming language also provides a do-while statement, it is just a alternative of While loop.

```
do {
statement(s)
} while (expression);
```

*----------print values 1 to 5----------

```
int i = 100;
do
{
     System.out.println("value  is:  " + i);
     i++;
} while (i <= 5);
```

//----------

for –each loop :

It is designed for arrays or collections

Using for each loop we can take values from array.

```
int[] a = {1,2,3,4,5,6,7,8,9,10};
for (int i : a)
{
   System.out.println("Value  is: " + i);
}
//---------------------------------------------------------------------
```

Branching statements:

break, continue, return

```
public class Branching {
        public static void main(String[] args)
      {
              // break command
      int[] arrayOfInts = { 32, 87, 3, 589, 12, 1076,2000, 16, 622, 127 };
      int searchfor = 16;
      int i;
```

```
boolean foundIt = false;
for (i = 0; i < arrayOfInts.length; i++)
{
    if (arrayOfInts[i] == searchfor) {
        foundIt = true;
        break;
    }
}
if (foundIt) {
    System.out.println("Found " + searchfor + " at index " + i);
} else {
    System.out.println(searchfor + " not in the array");
}

// Continue
String searchMe = "peter piper picked a peck of pickled peppers";
int max = searchMe.length();
int numPs = 0;

for (int j = 0; j < max; j++)
{
    //interested only in p's
    if (searchMe.charAt(j) != 'p')
```

```
            continue;


        //process p's
        numPs++;
    }
    System.out.println("Found " + numPs + " p's in the string.");

  }

}
```

What is a Method?

Method is a series of statements,
Reusable code we will store under a method and we will call whenever we need that method.

Methods we can use in four ways

Type	Arguments (Input)	Returns (Output)
I	N	N
2	Y	N
3	N	Y
4	Y	Y

if you don't want to return an output, then method type should be void.

Syntax

```
void methodname()
{
        code;
}
```

If you want to return an output, then define method with return type (int|String|boolean|float),
And also add return statement, return statement should be last line of the method.

```
boolean methodname()
{
        code;
        return true;
}
```

In single class we can write n-number of methods, but if you want to execute methods we should create instance/object for class.

access modifiers.

The access modifiers in java specifies accessibility (scope) of a data member, method, constructor or class.

There are 4 types of java access modifiers:

1. private :The private access modifier is accessible only within class.
2. default: If you don't use any modifier, it is treated as **default** bydefault. The default modifier is accessible only within package.
3. protected:The **protected access modifier** is accessible within package and outside the package but through inheritance only.
4. public:The **public access modifier** is accessible everywhere. It has the widest scope among all other modifiers.

What is Package?

1. Package is a folder, under each package we can create collection of classes.
2. To import one package classes into another package, we should use import keyword.

Navigation To Create Package:

1. Right Click on SRC
2. Go TO New ->Select Package
3. Enter Package name and Click on Finish.

What is a Jar?

Jar is nothing but collection of Packages, Each Package contains collection of classes.

Navigation For Adding Jar file to current project in Eclipse:

1. Right click on project
2. Select properties

3. Select java build path
4. Click Libraries
5. Click on Add External Jars
6. Browse and select jar
7. Apply and OK

Thank you