# **Welcome to Java**

# Inner class

In java, inner class means ,Every java program must run within the class of main function, without class no java program is run .In that inner class means the class contains one or more class within itself.

Example:

public class Javaclass {

class newclass

{

void fun(){

System.out.println("HEllo,LogesH");

}

class newclass2

{

void fun2(){

System.out.println("U R the BEST");

}

}

}

public static void main(String[] args) {

Javaclass obj1=new Javaclass();

Javaclass.newclass obj2=obj1.new newclass();

obj2.fun();

// newclass.newclass2 obj3=obj2.new newclass2();

Javaclass.newclass.newclass2 obj3=obj2.new newclass2();

obj3.fun2();

}

}

# Static inner class

Static inner class is used to minimize the object creation for access the inner class, that means in normal inner class we create the object for outer class and create the object for inner class by using the outer class object

By using this static inner class we can call the functions of inner class by one object itself.

Example:

public class Javastaticclass {

static class newclass{

void fun1()

{

System.out.println("U R the BEST");

}

static class newclass2{

void fun2()

{

System.out.println("Hello,LogesH");

}

}

}

public static void main(String[] args) {

// Javastaticclass.newclass obj2=new newclass();

// obj2.fun1();

Javastaticclass.newclass.newclass2 obj=new Javastaticclass.newclass.newclass2();

obj.fun2();

}

}

# Method local inner class

Method local inner class means class in present inside the method ,so the class is local for that method only. The method local inner class is also called by object that object is called with in that method, then only the method can access that inner class.

Example:

public class Methodinnerclass

{

class newclass

{

void fun()

{

class newclass2

{

void innerfun()

{

System.out.println("Hello,LogesH");

}

}

newclass2 refobj=new newclass2();

refobj.innerfun();

}

}

public static void main(String[] args) {

Methodinnerclass obj1=new Methodinnerclass();

Methodinnerclass.newclass obj2=obj1.new newclass();

obj2.fun();

}

}

# Loops

Loops are used for iteration purpose .In Java, there are 3 loops are available

* For
* While
* Do while

But in for loop there are 2 types , one is normal form and another one is for each or Enhanced for loop

Declaration of Enhanced for loop

Data\_type variable : array\_variable

Eg: for(int k : a) //if a is an array int[] a={1,2,3,4,5}

Example:

public class Java2 {

public static void main(String[] args) {

int a[]={1,2,3,4,5};

int x=10;

for(int k :a)

{ System.out.println(k);

k+=1;

if(k==3)break;

}

for(int i=0;i<=10;i++)

{ if(i%2==0)continue;

{

System.out.print(i);

continue;}

}

while(x==10)

{ System.out.println("\n"+"LOgesh");

break;

}

do

{

System.out.println("Logu");

}

while(x<10);

} }

# Inheritance

Inheritance is used to inherit the sub class from super class ,i.e. class is derived from another class

The derived class contains the information about the parent class or super class, but the super class does not contains the information about the derived class .

extends is the keyword that is used to derive the class from one class to another.

Example:

import java.io.\*;

import java.util.\*;

class a

{ int a=5;

}

class b extends a

{ int b=10;

void bvalue()

{

int z=10+ a;

System.out.println(z);}

}

class c extends b

{ int c=a\*b;

void cvalue()

{ c=10+b+a;

System.out.println(c);}

}

public class Java1 {

int a,b;

public static void main(String[] args) {

c obj=new c();

b obj2=new b();

obj2.bvalue();

obj.cvalue();

Java1 obj3=new Java1();

obj3.sum();

obj3.multiply();

}

public void sum()

{ a=5;

b=a+5;

System.out.println(b);

}

public void multiply()

{ int c=b\*a;

System.out.println(c);

}

}

In java multiple inheritance are not directly possible, that means the sub class is created my using two or more super class

|  |
| --- |
| Example  Class sub extends S1,S2  //it is not possible |



Example

c class sub extends s1,s2

To implement this multiple inheritance, the interface implementation is help full to archive the multiple inheritance

Example:

interface parent1

{

int a=10;

}

interface parent2

{

int b=20;

}

public class Interface implements parent1,parent2{

public static void main(String[] args) {

int c=a+b;

System.out.println(c);

}

}

Here interface is a key word to declare the interface class .And implements is a keyword to call the interface class.

The interface class also perform inheritance using the key word extends

Example:

|  |
| --- |
| interface gparent  {  int d=5;  }  interface parent1 extends gparent  {  int a=10+d;  } |