**Inner Classes in Java Programming**

If one class is existing within another class is known as inner class or nested class

**Syntax**

class Outerclass\_name

{

.....

.....

class Innerclass\_name1

{

.....

.....

}

class Innerclass\_name1

{

.....

.....

}

.....

}

**The main purpose of using inner class**

* To provide more security by making those inner class properties specific to only outer class but not for external classes.
* To make more than one property of classes private properties.

Private is a keyword in java language, it is preceded by any variable that property can be access only within the class but not outside of it (provides more security).

If more than one property of class wants to make as private properties than all can capped under private inner class.

**Syntax**

class Outerclass\_name

{

private class Innerclass\_name

{

.....

..... //private properties

}

}

**Note:**No outer class made as private class otherwise this is not available for JVM at the time of execution.

**Rules to access properties of inner classes**

* Inner class properties can be accessed in the outer class with the object reference but not directly.
* Outer class properties can be access directly within the inner class.
* Inner class properties can't be accessed directly or by creating directly object.

**Note:**In special situation inner class property can be accessed in the external class by creating special objects with the reference of its outer class.

**Example**

**class** A //outer class

{

**void** fun1()

{

System.**out**.println("Hello fun1()"); // inner class properties should be access using

//object reference in outer class.

B ob=**new** B();

ob.x=10

System.**out**.println("x= "+ob.x);

ob.fun2();

}

**void** fun3() // outer class fun3()

{

System.**out**.println("Hello fun3()");

}

**class** B // inner class

{

**int** x; // inner class variable

**void** fun2() //inner class fun2()

{

System.**out**.println("Hello fun2()");

fun3(); //outer class properties can be access directly

}

}

}

**class** C // external class

{

**void** fun3()

{

System.**out**.println("Hello fun3()");

}

}

**class** IncDemo

{

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

{

A oa=**new** A();

oa.fun1();

C oc=**new** C();

oc.fun3();

}

}

**Output**

Hello fun1()

X=10

Hello fun2()

Hello fun3()

**Accessing inner class properties in the external class**

1. If inner class in non static the object can be created with the following syntax

**Syntax**

class Outer\_class

{

class Inner\_class

{

.....

.....

}

.....

.....

}

class External\_class

{

Outer\_class.Inner\_Class objectrefernce=new Outer\_Class.External\_Class();

}

2. If inner class is static the object reference can be created with the following syntax

**Syntax**

class Outer\_class

{

static class Inner\_Class

{

.....

.....

}

}

class External\_Class

{

Outer\_class.Inner\_Class objectrefernce=new Outer\_Class.External\_Class();

}

}

**Example**

**class** A //Outer class

{

**class** B // non-static inner class

{

**int** x; //inner class variable

**void** fun1() //inner class fun1()

{

System.**out**.println("Hello fun1()");

}

}

**static** **class** C //static inner class

{

**int** y=20; // inner class variable

**void** fun2()

{

System.**out**.println("Hello fun2()");

}

}

}

**class** IncDemo

{

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

{

A.B ob=**new** A().**new**.B();

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

ob.fun1();

A.C oc=**new** A.C();

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

oc.fun2();

}

}