2.3: STATIC, NESTED and INNER CLASSES

Definition:

An inner class is a class that is defined inside another class.
Inner classes let you make one class a member of another class. Just as classes have member variables and methods, a class can also have member classes.

Benefits:

1. Name control
2. Access control
3. Code becomes more readable and maintainable because it locally group related classes in one place.

Syntax: For declaring Inner classes

```
[modifier] class OuterClassName
{
        ---- Code -----
          [modifier] class InnerClassName
          {
          ---- Code ----
          }
}
```

➢ **Instantiating an Inner Class:**
Two Methods:

1. **Instantiating an Inner class from outside the outer class:**
   To instantiate an instance of an inner class, you must have an instance of the outer class.
   Syntax:
   OuterClass.InnerClass objectName=OuterObj.new InnerClass();

2. **Instantiating an Inner Class from Within Code in the Outer Class:**
   From inside the outer class instance code, use the inner class name in the normal way:

Syntax:

InnerClassName obj=new InnerClassName();

Advantage of java inner classes

There are basically three advantages of inner classes in java. They are as follows:

1) Nested class **can access all the members (data members and methods) of outer class** including private.
2) Nested classes are used **to develop more readable and maintainable code**.
3) **Code Optimization**: It requires less code to write.

Types of Nested classes

There are two types of nested classes non-static and static nested classes. The non-static nested classes are also known as inner classes.

- Non-static nested class (inner class)
    1. Member inner class
    2. Anonymous inner class
    3. Local inner class
- Static nested class

Type	**Description**
Member Inner Class	A class created within class and outside method.
Anonymous Inner Class	A class created for implementing interface or extending class. Its name is decided by the java compiler.
Local Inner Class	A class created within method.
Static Nested Class	A static class created within class.
Nested Interface	An interface created within class or interface.

1. Java Member inner class

A non-static class that is created inside a class but outside a method is called member inner class.

Syntax:

```
class Outer
{
 //code
    class Inner
    {
        //code
    }
}
```

Java Member inner class example

In this example, we are creating msg() method in member inner class that is accessing the private data member of outer class.

```
1.    class TestMemberOuter1
2.    {
 private int data=30;

 class Inner
{
  void msg()
{
System.out.println("data is "+data);
}
 }
 public static void main(String args[])
{
  TestMemberOuter1 obj=new TestMemberOuter1();
  TestMemberOuter1.Inner in=obj.new Inner();
  in.msg();
 }
}
```

Output:

```
data is 30
```

2. Java Anonymous inner class

A class that have no name is known as anonymous inner class in java. It should be used if you have to override method of class or interface. Java Anonymous inner class can be created by two ways:

1. Class (may be abstract or concrete).
2. Interface

Java anonymous inner class example using class

```
abstract class Person
{
  abstract void eat();
}
class TestAnonymousInner
{
 public static void main(String args[])
{
  Person p=new Person()
{
  void eat()
{
    System.out.println("nice fruits");
}
  };
  p.eat();
 }
}
```

Output:

```
nice fruits
```

Java anonymous inner class example using interface

```
interface Eatable
{
 void eat();
 }
class TestAnnonymousInner1
{
 public static void main(String args[])
{
 Eatable e=new Eatable()
  {
  public void eat(){System.out.println("nice fruits");
  }
 };
 e.eat();
 }
}
```

Output:

```
nice fruits
```

3. Java Local inner class

A class i.e. created inside a method is called local inner class in java. If you want to invoke the methods of local inner class, you must instantiate this class inside the method.

Java local inner class example

```
public class localInner1
{
 private int data=30;//instance variable
void display()
{
int value=50;
  class Local
{
 void msg()
   {
    System.out.println(data);
    System.out.println(value);
   }
 }
 Local l=new Local();
 l.msg();
}
public static void main(String args[])
{
 localInner1 obj=new localInner1();
 obj.display();
}
}
```

Output:

```
30
50
```

Rules for Java Local Inner class

1. Local inner class cannot be invoked from outside the method.
2. Local inner class cannot access non-final local variable till JDK 1.7. Since JDK 1.8, it is possible to access the non-final local variable in local inner class.
3. Local variable can't be private, public or protected.

Properties:

1. Completely hidden from the outside world.
2. Cannot access the local variables of the method (in which they are defined), but the local variables has to be declared final to access.

4. Java static nested class

A static class i.e. created inside a class is called static nested class in java. It cannot access non-static data members and methods. It can be accessed by outer class name.

- It can access static data members of outer class including private.
- Static nested class cannot access non-static (instance) data member or method.

Java static nested class example with instance method

```
class TestOuter1
{
    static int data=30;
    static class Inner
  {
     void msg()
     {
       System.out.println("data is "+data);
      }
    }
    public static void main(String args[])
   {
    TestOuter1.Inner obj=new TestOuter1.Inner();
    obj.msg();
    }
   }
```

Output:

```
data is 30
```

Java static nested class example with static method

If you have the static member inside static nested class, you don't need to create instance of static nested class.

```
class TestOuter2{
  static int data=30;
static class Inner
{
   static void msg()
    {
     System.out.println("data is "+data);
    }
  }
  public static void main(String args[])
{
  TestOuter2.Inner.msg();//no need to create the instance of static nested class
  }
}
```

Output:

```
data is 30
```