Programming Language II
CSE-215

Prof. Dr. Mohammad Abu Yousuf
yousuf@juniv.edu

Java Inner Classes

- **Java inner class** or nested class is a class which is declared inside the class or interface.

- We use inner classes to logically group classes and interfaces in one place so that it can be more readable and maintainable.

- Additionally, it can access all the members of outer class including private data members and methods.

Syntax of Inner class

```
class Java_Outer_class{
 //code
 class Java_Inner_class{
  //code
 }
}
```

Advantage of java inner classes

- There are basically three advantages of inner classes in java. They are as follows:
- 1) Nested classes represent a special type of relationship that is **it 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** because it logically group classes and interfaces in one place only.
- 3) **Code Optimization**: It requires less code to write.

Difference between nested class and inner class in Java

- Inner class is a part of nested class. Non-static nested classes are known as inner classes.

Types of Nested classes:

- Non-static nested class (inner class)
  - Member inner class
  - Anonymous inner class (Will learn later on)
  - Local inner class
- Static nested class

Difference between nested class and inner class in Java

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.

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

```
class TestMemberOuter1{
 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

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

Internal working of Java member inner class

- The java compiler creates two class files in case of inner class. The class file name of inner class is "**Outer$Inner**".
- If you want to instantiate inner class, you must have to create the instance of outer class. In such case, instance of inner class is created inside the instance of outer class.

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

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

Output:

```
30
```

Java Local inner class

Rule: Local variable can't be private, public or protected.

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.

Example of local inner class with local variable

```
class localInner2{
 private int data=30;//instance variable
 void display(){
  int value=50;//local variable must be final till jdk 1.7 only
  class Local{
   void msg(){System.out.println(value);}
  }
  Local l=new Local();
  l.msg();
 }
 public static void main(String args[]){
  localInner2 obj=new localInner2();
  obj.display();
 }
}
```

Output:

```
50
```

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();
  }
}
```

In this example, you need to create the instance of static nested class because it has instance method msg().
But you **don't need to create the object** of Outer class because nested class is static and static properties, methods or classes can be accessed without object.

Output:

```
data is 30
```

Java static nested class example with static method

```
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 instanc
e of static nested class
  }
}
```

```
Output:   data is 30
```

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

Thank you