ECAP615

Programming in Java

Harjinder Kaur

Assistant Professor

Learning Outcomes

After this lecture, you will be able to

- learn the basic concept of Nested Classes,
- understand the different types of Nested Classes,
- difference in the working of different types of inner classes,
- difference between inner and static nested classes.

Nested Classes

- The Java programming language allows you to define a class within another class.
- Such a class is called a *nested class.*

Syntax:

```
class OuterClass {

...

class NestedClass {

...

} }
```

Properties of Nested Class

- The scope of a nested class is bounded by the scope of its enclosing class.
- A nested class has access to the members, including private members, of the class in which it is nested.
- However, the reverse is not true i.e., the enclosing class does not have access to the members of the nested class.

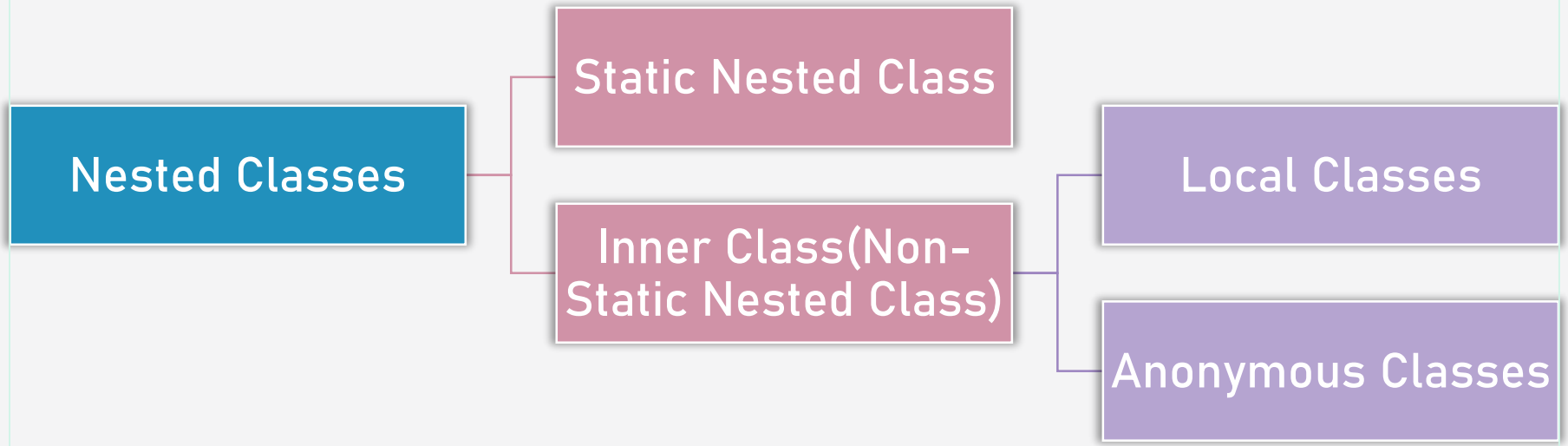
Properties of Nested Class

- A nested class is also a member of its enclosing class.
- As a member of its enclosing class, a nested class can be declared as private, public, protected, or package private.

Why Use Nested Classes?

- Logical grouping of Classes.
- Increased encapsulation.
- More readable, maintainable code.

Categories of Nested Classes



Categories of Nested Classes

Nested classes are divided into two categories:

1. Static
2. Non-static.

- Nested classes that are declared static are called static nested classes.
- Non-static nested classes are called inner classes.

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.

Static Nested Classes

- A static nested class is associated with its outer class similar to class methods and variables.
- A static nested class cannot refer directly to instance variables or methods defined in its enclosing class.
- It can use them only through an object reference.

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();

  }

}
```

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 instance of sta
tic nested class    }

}
```

Inner Class Types

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.

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
 }
}
```

Example

```
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();   }  }
```

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:

- Class (may be abstract or concrete).
- Interface

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();
 }
}
```

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();
 }
}
```

Local inner class

A class 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.

Example

```
public class localInner1{
 private int data=30;//instance variable
 void display(){
  class Local{
   void msg()
{
System.out.println(data);}
  }
```

Example

```
 Local l=new Local();

  l.msg();

 }  public static void main(String args[]){

  localInner1 obj=new localInner1();

  obj.display();

 }

}
```

Rules for Local Inner class

- Local inner class cannot be invoked from outside the method.
- 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.

Advantage of inner classes

The following are the basic advantages of inner classes :

- It can access all the members (data members and methods) of outer class.
- Code Optimization
- To develop more readable and maintainable code.

Advantage of inner classes

The following are the basic advantages of inner classes :

- It can access all the members (data members and methods) of outer class.
- Code Optimization
- To develop more readable and maintainable code.

Advantage of inner classes

The following are the basic advantages of inner classes :

- It can access all the members (data members and methods) of outer class.
- Code Optimization
- To develop more readable and maintainable code.

Comparison between Inner and Static Nested Class

Inner class	Static nested class
The inner class object is always associated with the outer class object.	Static nested class object is not associated with the outer class object.
Inside inner class, static members can't be declared.	Inside static nested class, static members can be declared.
As main() method can't be declared, regular inner class can't be invoked directly from the command prompt.	As main() method can be declared, the static nested class can be invoked directly from the command prompt.
Both static and non static members of outer class can be accessed directly.	Only a static member of outer class can be accessed directly.


That’s all for now…