Inner Class in Java

A class declared inside another class is known as nested classes in java. The scope of a nested class is tied by the scope of its enclosing class (outer class).

Thus, if class B is defined inside class A, then class B cannot exist independently without class A.

Here, class A is outer or enclosing class and class B is nested class.

Inner class in Java

An inner class in java is a class that is declared inside of another class without static modifier. It is also commonly known as a **non-static nested class** in Java. It can access all members (**variables and methods**) of its **outer class**.

An **inner class** cannot have any kind of static members. The members of the java inner class may be:

1. **Instance variables**
2. **Instance methods**
3. **Constructors**
4. **Initializer block**
5. **Inner class**

Syntax of Inner class

```
access modifier class OuterClassName {
        access modifier class InnerClassName  {
            // Members of the inner class.
        }
  // Other members of the outer class.
}
```

For example:

```
public class B {        //  Here, Test is an outer class.
    // Inner class
    public class A { // A is inner class as a member of its outer class.
     }
}
```

Features of Inner class

There are several important features of an inner class that is as follows:

1. An inner class cannot have the same name as the outer class. But, it is possible to use the same names for members of both outer and inner classes.
2. The class that holds the inner class is called enclosing or outer class. The enclosing class is a top-level class whereas the inner class is member of outer class.
3. The scope of inner class is bounded by the scope of its outer class.
4. Without existing an outer class object or instance, there will be no chance of an existing inner class object.
5. An inner class can directly access all the variables and methods of the outer class including private.

Features of Inner class

6. If the variable name of inner class is the same as the variable name of outer class, we can access the outer class variable like this. **OuterClassName.this.VariableName**; here this represents current outer class object.

7. Java inner class is hidden from another class in its enclosing class. Therefore, it provides a safety mechanism in the application program and decreases readability (understanding) of the program.

8. An object of inner class is often created in its outer class and cannot be created from other classes.

9. Both outer class and inner class objects are created in separate memory locations.

Features of Inner class

10. The relationship between outer class and inner class represents Has-A relationship. A **Has-A relationship** establishes the relationship between two classes. It is also known as composition or aggregation in Java.
11. When an inner class is an enclosing class for another inner class, it is called nesting of inner classes. There is no limit to the nesting of inner classes.
12. Inner class was introduced in the **Java 1.1** version without any changes in the JVM used to handle the class files. It is completely implemented with the help of Java compiler.
13. Java compiler generates a separate dot class file for each inner class named **Outer$Inner.class** and for an outer class named **Outer.class**. For example, an inner class A is compiled into a class Test named **Test$A.class**.
14. If any dot class file name contains **"$"** symbol in its middle, it shows an inner class file.
15. A member inner class cannot have a static declaration. It means that we cannot have the main method in the inner class because it is static.

Syntax to Create Object of Inner class in Java

Syntax:

```
OuterClass.InnerClass innerObject = outerObject.new InnerClass();
```

For example:

```
// First create an object of Outer class Test.
   Test  t = new  Test();
// Second create an object of inner class A.
   Test.A   innerObj = t.new   A();
```

Types of Inner class in Java

Based on declaration and behaviors, there are basically four types of inner classes in Java. They are as follows:

1. **Normal or Regular Inner Class**
2. **Method Local Inner Class**
3. **Anonymous Inner Class**
4. **Static Nested Inner Class**

Types of Inner class in Java

1. **Normal or Regular inner class:** A class created inside another class and outside the method without static modifier is called regular or normal inner class.

2. **Method local inner class:** A class declared within a method of the outer class is called method local inner class.

3. **Anonymous inner class:** A class declared without any name at all is called anonymous inner class.

4. **Static nested inner class:** A class declared with static modifier within a class is called static nested inner class. It is also known as a top-level nested class. It can access only the static members of the outer class.

Advantages of Inner class

- Inner class provides a way to group the classes and interfaces in one place that logically belongs together.
- It can directly access all the variables and methods of the outer class including private.
- Inner class is used to develop more readable and maintainable code in Java.
- It needs less code to write.
- It increases encapsulation.

How to instantiate Member Inner class in Java?

The general form of syntax to create an object of the member inner class is as follows:

Syntax:

```
OuterClassReference.new   MemberInnerClassConstructor();
```

Here, **OuterClassReference** is the reference of the outer class followed by a dot which is followed by the **new** operator.

How to instantiate Member Inner class in Java?

Example:

Step 1: To create an object of inner member inner class, we must have to create first an object of its outer class.

```
Outer  o = new  Outer();      // (1)
```

Step 2: Now we need to use the new operator on the 'o' object reference variable to create the object of a member inner class.

```
Outer.Inner  i  =  o.new  Inner();  // (2)
```

Where,

i is an object reference variable to store the member inner class object.

Inner() is a constructor name that is the same as the simple class name for member inner class.

How to instantiate Member Inner class in Java?

When we combine the above **Step 1 and Step 2** lines of code, we can write the following new statement.

```
Outer.Inner i = new Outer.new Inner();   // One line of code.
```

Suppose we are invoking any method like **m1()** using object reference variable **i.**

```
i.m1();          // Calling m1 method.
```

By combining (1), (2), and (3), we will get the following statement.

```
new Outer().new Inner().m1();
```

Just a minute:

How will you create an object of class D for the above lines of code?

```
public class A
{
  public class B
{
     public class C
   {
        public class D
       {
            . . . . . .
       }
    }
  }
}
```

```
A  a  =  new   A();
 A.B  b  =  a.new   B();
  A.B.C  c  =  b.new   C();
   A.B.C.D  d =  c.new   D();
```

Anonymous Inner Class in Java

An anonymous inner class in Java is an inner class which is declared without any class name at all. In other words, a nameless inner class is called anonymous inner class.

Java anonymous inner classes are useful when we need only one object of the class. Since an anonymous inner class does not have a name, it cannot have a constructor because we know a constructor name is the same as the class name.

Static Nested Class in Java

When an inner class is defined with a static modifier inside the body of another class, it is known as a **static nested class** in Java.

A static nested class is also considered a top-level class, nested top-level class, or static member class in java. But it is not considered an inner class.

```
// A top-level class.
 public class Outer
 {
// Static member class.
   public static class Nested
   {
      // Body for class Nested.
   }
 }
```

Can We Declare Main Method within Static Nested Class?

Yes, we can declare the main method inside a static nested class.

A static nested class can have both static and non-static members including the main method.

Can we create object of Static Nested Class outside Outer class?

Yes, we can create an object of the static nested class from outside the outer class. Let's take an example program where we will create an object of static nested class outside the outer class.

FAQs in Interview:

1. Can you declare a class inside a class?
2. Can you declare a class inside an interface?
3. Can you declare an interface inside a class?
4. Can you declare an interface inside an interface?

Answer: If the interviewed person asks the above kind of questions then you tell a simple answer like this. Anything inside anything is possible with respect to class and interface. All four combinations are valid.