Nested Classes in Java

Difficulty Level : Easy • Last Updated : 19 Oct, 2022

In Java, it is possible to define a class within another class, such classes are known as *nested classes*. They enable you to logically group classes that are only used in one place, thus this increases the use of the encapsulation, and creates more readable and maintainable code.

- The scope of a nested class is bounded by the scope of its enclosing class. Thus in below example, class *NestedClass* does not exist independently of class *OuterClass*.
- A nested class has access to the members, including private members, of the class in which it is nested. But the enclosing class does not have access to the members of the 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 *private*, *public*, *protected*, or *package private*(default).
- Nested classes are divided into two categories:
  1. **static nested class** : Nested classes that are declared static are called static nested classes.
  2. **inner class** : An inner class is a non- static nested class.

Syntax:

```
class OuterClass
{
...
    class NestedClass
    {
        ...
    }
}
```

Static nested classes

In the case of normal or regular inner classes, without an outer class object existing, there cannot be an inner class object. i.e., an object of the inner class is always strongly associated with an outer class object. But in the case of static nested class, Without an outer class object existing, there may be a static nested class object. i.e., an object of a static nested class is not strongly associated with the outer class object. As with class methods and variables, a static nested class is associated with its outer class. And like static class methods, 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. They are accessed using the enclosing class name.

```
OuterClass.StaticNestedClass
```

For example, to create an object for the static nested class, use this syntax:

```
OuterClass.StaticNestedClass nestedObject =
    new OuterClass.StaticNestedClass();
```

Java

```
// Java program to demonstrate accessing
// a static nested class

// outer class
class OuterClass
{
    // static member
    static int outer_x = 10;

    // instance(non-static) member
    int outer_y = 20;

    // private static
    private static int outer_private = 30;

    // static nested class
    static class StaticNestedClass
    {
        void display()
        {
            // can access static member of outer class
            System.out.println("outer_x = " + outer_x);

            // can also access private static member of outer class
            System.out.println("outer_private = " + outer_private);

            // can't access non-static members directly since
            // an static nested class cannot directly access outer
            // System.out.println("outer_y = " + outer_y);
        }
    }

    // driver code
    public static void main(String[] args)
    {
        // accessing an inner static class
        OuterClass.StaticNestedClass nestedObject = new OuterClass
        nestedObject.display();
    }
}
```

Output:

```
outer_x = 10
outer_private = 30
```

Inner classes

To instantiate an inner class, you must first instantiate the outer class. Then, create the inner object within the outer object with this syntax:

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

There are two special kinds of inner classes :

1. Local inner classes
2. Anonymous inner classes

Java

```
// Java program to demonstrate accessing
// a inner class

// outer class
class OuterClass
{
    // static member
    static int outer_x = 10;

    // instance(non-static) member
    int outer_y = 20;

    // private static
    private static int outer_private = 30;

    // inner class
    class InnerClass
    {
        void display()
        {
            // can access static member of outer class
            System.out.println("outer_x = " + outer_x);

            // can also access non-static member of outer class
            System.out.println("outer_y = " + outer_y);

            // can also access private static member of outer class
            System.out.println("outer_private = " + outer_private
        }
    }

    // driver code
    public static void main(String[] args)
    {
        OuterClass outerObject = new OuterClass();
        OuterClass.InnerClass innerObject = outerObject.new Inner

        innerObject.display();
    }
}
```

Output:

```
outer_x = 10
outer_y = 20
outer_private = 30
```

Comparison between normal or regular inner class and static nested class

S.NO	Normal/Regular inner class	Static nested class
1.	Without an outer class object existing, there cannot be an inner class object. That is, the inner class object is always associated with the outer class object.	Without an outer class object existing, there may be a static nested class object. That is, static nested class object is not associated with the outer class object.
2.	Inside normal/regular inner class, static members can't be declared.	Inside static nested class, static members can be declared.
3.	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.
4.	Both static and non static members of outer class can be accessed directly.	Only a static member of outer class can be accessed directly.

Like 77

Previous
Anonymous Inner Class in Java

Next
Java.util.Objects class in Java

Related Articles

1. Nested Interface in Java
2. Nested try blocks in Exception Handling in Java
3. Java Nested Loops with Examples
4. Break Any Outer Nested Loop by Referencing Its Name in Java
5. Adding Nested Tables to a PDF using Jova
6. Java Program to Demonstrate the Nested Initialization For Singleton Class
7. Flattening Nested Collections in Java
8. Nested if in Java
9. Difference Between Static and Non Static Nested Class in Java
10. Java Program to List all Files in a Directory and Nested Sub-Directories

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Easy

Easy Normal Medium Hard Expert

Improved By : gauramsingh39, somanshreddy, mayankbdj, bhuvanesh, codewithkunal, tanhaankanjaa

Article Tags : Java-Class and Object, Java

Practice Tags : Java, Java-Class and Object

Improve Article Report Issue

WHAT'S NEW

JAVA Programming Foundation- Self Paced Course
View Details

Complete Interview Preparation- Self Paced Course
View Details

Data Structures & Algorithms- Self Paced Course
View Details

GeeksforGeeks
A-143, 9th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

© geeksforgeeks. All rights reserved