SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

COURSE NAME : 23CSB101- OBJECT ORIENTED PROGRAMMING

I YEAR /II SEMESTER

Unit II – INHERITANCE, PACKAGES AND INTERFACES

Topic : STATIC, NESTED AND INNER CLASSES

Static, Nested and Inner Classes

- Classes can be nested inside other classes. These are called nested classes and can be either static or non-static (inner classes).

Static Nested Class

- A static nested class is a static class inside another class. It:
  - Does not need an instance of the outer class.
  - Can access only static members of the outer class.

Static Nested Class

```
class Outer {
  static String msg = "Hello from Outer Class";
 // Static nested class
    static class Nested {
        void display() {
            System.out.println(msg);
// Can access static members of Outer
        }
    }
}
```

```
public class Main {
    public static void main(String[] args) {
// Creating an object of static nested class
   Outer.Nested obj = new Outer.Nested();
    obj.display();
    }
}
```

Hello from Outer Class

An inner class (non-static nested class)

- **Needs** an **instance** of the outer class to be created.
- Can **access both static and non-static members** of the outer class.

An inner class (non-static nested class)

```
class Outer {
    String msg = "Hello from Outer Class";
 // Inner class (non-static)
    class Inner {
        void display() {
        System.out.println(msg);
// Can access non-static members
        }
    }
}
```

```
public class Main {
    public static void main(String[] args) {
  // Creating outer class object
 Outer outer = new Outer();
// Creating inner class object
Outer.Inner inner = outer.new Inner();
        inner.display();
    }
}
```

Hello from Outer Class

Local inner class

- A local inner class is **defined inside a method**.
- It **can access local variables** (if they are final or effectively final).
- It **cannot be used outside the method**.

Local inner class

```
class Outer {
    void outerMethod() {
        // Local inner class inside a method
        class Local {
            void display() {
System.out.println("Hello from Local Inner Class");
            }
        }
```

```
 Local obj = new Local(); // Creating object
inside the method
        obj.display();
    }
}
public class Main {
    public static void main(String[] args) {
        Outer outer = new Outer();
        outer.outerMethod();
    }
}
```

Hello from Local Inner Class

Anonymous inner class

- An anonymous inner class is a class **without a name**.
- **Used** when you need to **override a method** once.
- Commonly used with **interfaces or abstract classes**.

An inner class (non-static nested class)

```
abstract class Greeting
 {
    abstract void sayHello();
}
public class Main
{
    public static void main(String[] args)
{
```

```
// Anonymous inner class implementing Greeting
        Greeting g = new Greeting() {
            void sayHello() {
System.out.println("Hello from Anonymous Inner Class");
            }
        };
        g.sayHello();
    }
}
```

Hello from Anonymous Inner Class

Differences

Feature	Static Nested Class	Non-Static Inner Class	Local Inner Class	Anonymous Inner Class
Needs Outer Class Instance?	✗ No	✓ Yes	✓ Yes	✓ Yes
Can Access Outer Class Members?	✓ Only static	✓ Both static and non-static	✓ (only final/effectively final local variables)	✓ Can access members
Usage	Grouping helper classes	Access outer class members	Encapsulation within a method	Short-lived classes

Summary

- **Static Nested Class** → Works independently of the outer class.
- **Inner Class** → Requires an instance of the outer class.
- **Local Inner Class** → Defined inside a method.
- **Anonymous Inner Class** → No name, used for quick method overrides.


Thank
you