Local Inner Class in Java

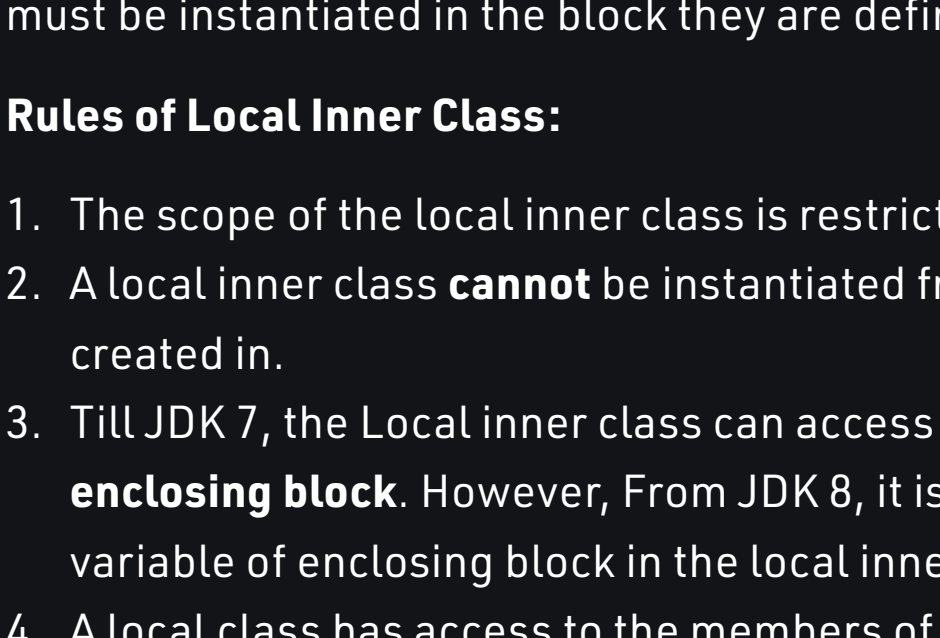
Difficulty Level : Medium • Last Updated : 07 Dec, 2021

Prerequisites: Nested Classes in Java

Local Inner Classes are the inner classes that are defined inside a *block*. Generally, this block is a method body. Sometimes this block can be a for loop or an if clause. Local Inner classes are not a member of any enclosing classes. They belong to the block they are defined within, due to which local inner classes cannot have any access modifiers associated with them. However, they can be marked as final or abstract. This class has access to the fields of the class enclosing it. Local inner class must be instantiated in the block they are defined in.

Rules of Local Inner Class:

1. The scope of the local inner class is restricted to the *block* they are defined in.
2. A local inner class **cannot** be instantiated from outside the *block* where it is created in.
3. Till JDK 7, the Local inner class can access only the final local variable of the **enclosing block**. However, From JDK 8, it is possible to access the non-final local variable of enclosing block in the local inner class.
4. A local class has access to the members of its **enclosing class**.
5. Local inner classes can extend an abstract class or implement an interface.



Declaring a Local Inner class: A local inner class can be declared within a block. This block can be either a method body, initialization block, for loop, or even an if statement.

Accessing Members: A local inner class has access to fields of the class enclosing it as well as the fields of the block that it is defined within. These classes, however, can access the variables or parameters of the block that encloses it only if they are declared as final or are effectively final. A variable whose value is not changed once initialized is called an effectively final variable. A local inner class defined inside a method body has access to its parameters.

What happens at compile time?

When a program containing a local inner class is compiled, the compiler generates two .class files, one for the outer class and the other for the inner class that has the reference to the outer class. The two files are named by the compiler as:

- Outer.class
- Outer$1Inner.class

Declaration within a method body

Java

```java
// Java program to illustrate
// working of local inner classes

public class Outer
{
    private void getValue()
    {
        // Note that local variable(sum) must be final till JDK 7
        // hence this code will work only in JDK 8
        int sum = 20;

        // Local inner Class inside method
        class Inner
        {
            public int divisor;
            public int remainder;

            public Inner()
            {
                divisor = 4;
                remainder = sum%divisor;
            }
            private int getDivisor()
            {
                return divisor;
            }
            private int getRemainder()
            {
                return sum%divisor;
            }
            private int getQuotient()
            {
                System.out.println("Inside inner class");
                return sum / divisor;
            }
        }

        Inner inner = new Inner();
        System.out.println("Divisor = "+ inner.getDivisor());
        System.out.println("Remainder = " + inner.getRemainder())
        System.out.println("Quotient = " + inner.getQuotient());
    }

    public static void main(String[] args)
    {
        Outer outer = new Outer();
        outer.getValue();
    }
}
```

Output

```
Divisor = 4
Remainder = 0
Inside inner class
Quotient = 5
```

Note: *A local class can access local variables and parameters of the enclosing block that are effectively final.*

For example, if you add the highlighted assignment statement in the *Inner* class constructor or any method of *Inner* class in the above example:

```java
public Inner()
{
   sum = 50;
   divisor = 4;
   remainder = sum%divisor;
}
```

Because of this assignment statement, the variable *sum* is not *effectively final* anymore. As a result, the Java compiler generates an error message similar to "local variables referenced from an inner class must be final or effectively final."

Declaration inside an if statement

Java

```java
// Java program to illustrate Declaration of
// local inner classes inside an if statement

public class Outer
{
    public int data = 10;
    public int getData()
    {
        return data;
    }
    public static void main(String[] args)
    {
        Outer outer = new Outer();

        if(outer.getData() < 20)
        {
            // Local inner class inside if clause
            class Inner
            {
                public int getValue()
                {
                    System.out.println("Inside Inner class");
                    return outer.data;
                }
            }

            Inner inner = new Inner();
            System.out.println(inner.getValue());
        }
        else
        {
            System.out.println("Inside Outer class");
        }
    }
}
```

Output

```
Inside Inner class
10
```

Demonstrating Erroneous codes for Inner class

Java

```java
// Java code to demonstrate that inner
// classes cannot be declared as static

public class Outer
{
    private int getValue(int data)
    {
        static class Inner
        {
            private int getData()
            {
                System.out.println("Inside inner class");
                if(data < 10)
                {
                    return 5;
                }
                else
                {
                    return 15;
                }
            }
        }

        Inner inner = new Inner();
        return inner.getData();
    }

    public static void main(String[] args)
    {
        Outer outer = new Outer();
        System.out.println(outer.getValue(10));
    }
}
```

Output

```
Compilation error
```

Explanation: The above program causes compilation error because the inner class cannot be declared static. Inner classes are associated with the block they are defined within and not with the external class(Outer in this case).

Java

```java
// Java code to demonstrate
// the scope of inner class

public class Outer
{
    private void myMethod()
    {
        class Inner
        {
            private void innerMethod()
            {
                System.out.println("Inside inner class");
            }
        }
    }

    public static void main(String[] args)
    {
        Outer outer = new Outer();
        Inner inner = new Inner();
        System.out.println(inner.innerMethod());
    }
}
```

Output

```
prog.java:20: error: cannot find symbol
        Inner inner = new Inner();
        ^
  symbol:   class Inner
  location: class Outer
```

Explanation: The above program causes compilation error because the scope of inner classes is restricted to the block they are defined in.

This article is contributed by **Mayank Kumar**. If you like GeeksforGeeks and would like to contribute, you can also write an article using write.geeksforgeeks.org or mail your article to review-team@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Like 27

Previous
How to prevent Singleton Pattern from Reflection, Serialization and Cloning?

Next
Moving a file from one directory to another using Java

Related Articles

1. Inner Class And Anonymous Inner Class that Implements Runnable | Concurrent Programming Approach 3
2. Java Program to Check if a Given Class is a Local Inner Class
3. Java Program to Check if a Given Class is an Inner Class
4. Java - Inner Class vs Sub Class
5. Inner Class in Java
6. Diamond operator for Anonymous Inner Class with Examples in Java
7. Java Program to illustrates Use of Static Inner Class
8. Anonymous Inner Class in Java
9. Difference between Anonymous Inner Class and Lambda Expression
10. How to Access Inner Classes in Java?

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Medium

Easy Normal Medium Hard Expert

Improved By : nishkarshgandhi

Article Tags : Local Inner Class, Nested Class, Java

Practice Tags : Java

Improve Article Report Issue