Static Keyword in Java

In Java, static keyword is mainly used for memory management. It can be used with variables, methods, blocks and nested classes. It is a keyword which is used to share the same variable or method of a given class. Basically, static is used for a constant variable or a method that is same for every instance of a class. The main method of a class is generally labeled static.

In order to create a static member (block, variable, method, nested class), you need to precede its declaration with the keyword static. When a member of the class is declared as static, it can be accessed before the objects of its class are created, and without any object reference.

In Java programming language, static keyword is a non-access modifier and can be used for the following:

- Static Block
- Static Variable
- Static Method
- Static Classes

// Java program to demonstrate the use of static blocks

```
import java.util.*;
public class StaticBlock
 {
        static int j = 10;
        static int n;
static {
          System.out.println("Static block initialized.");
            n = j * 8;
        }
 public static void main(String[] args)
        {       System.out.println("Inside main method");
            System.out.println("Value of j : "+j);
            System.out.println("Value of n : "+n);
        }
  }
```

Static Block

If you need to do the computation in order to initialize your static variables, you can declare a static block that gets executed exactly once, when the class is first loaded. Take a look at the below Java program to understand the usage of Static Block.

When you execute the above program, static block gets initialized and displays the values of the initialized variables.

Output:

```
Static block initialized
Inside main method
Value of j:10
Value of n : 80
```

Now that you know how static block works, let's move further and see what are static variables and how it is helpful.

Static Variable

When you declare a variable as static, then a single copy of the variable is created and divided among all objects at the class level. Static variables are, essentially, global variables. Basically, all the instances of the class share the same static variable. Static variables can be created at class-level only.

```
// Java program demonstrate execution of static blocks and variables
 import java.util.*;
 public class VariableExample
{
static int j = n();
static {
System.out.println("Inside the static block");
}
 static int n() {
          System.out.println("from n ");
          return 20;
                }
 public static void main(String[] args)
           {
           System.out.println("Value of j : "+j);
           System.out.println("Inside main method");
         }
}
```

When you execute the above program, it will execute static block and the variable in order as defined in the above program.

Output:

from n
Inside the static block
Value of j: 20
Inside main method

Important points for static variables

- We can create static variables at class-level only. See here
- Static block and static variables are executed in order they are present in a program.

Static Methods

When a method is declared with the static keyword, it is known as a static method. The most common example of a static method is the main( ) method. Methods declared as static can have the following restrictions:

They can directly call other static methods only.
They can access static data directly.

Note

- A static method belongs to the class rather than the object of a class.
- A static method can be invoked without creating an instance of a class.
- A static method can access static data member and can change the value of it.

There are two main restrictions for the static method. They are:

- The static method can not use non static data member or call non-static method directly.
- this and super cannot be used in static context.