In Java, the `static` keyword is used to denote class-level behavior and properties, as opposed to instance-level behavior and properties. This means that `static` members belong to the class itself rather than to any specific instance of the class. Here's a breakdown of how `static` is used:

1. Static Variables (Class Variables)

Static variables are shared among all instances of a class. They are initialized only once, at the start of the execution, and all instances of the class share the same copy of this variable.

```
java
Copy code
public class StaticVariableExample {
    static int count = 0; // Static variable

    public StaticVariableExample() {
        count++;
    }

    public static void displayCount() {
        System.out.println("Count: " + count);
    }

    public static void main(String[] args) {
        StaticVariableExample obj1 = new StaticVariableExample();
        StaticVariableExample obj2 = new StaticVariableExample();

        StaticVariableExample.displayCount(); // Outputs: Count: 2
    }
}
```

2. Static Methods

Static methods belong to the class, not instances of the class. They can be called without creating an instance of the class. Static methods can only directly access other static members (variables and methods) of the class.

```
java
Copy code
public class StaticMethodExample {
    static int square(int x) {
        return x * x;
    }

    public static void main(String[] args) {
        int result = StaticMethodExample.square(5);
        System.out.println("Square of 5: " + result); // Outputs: Square of
5: 25
    }
}
```

3. Static Blocks

Static blocks are used for static initializations of a class. This block is executed only once, when the class is first loaded into memory. It's useful for setting up static variables or performing class-level initialization.

```
java
Copy code
public class StaticBlockExample {
    static {
        System.out.println("Static block is executed.");
    }

    public static void main(String[] args) {
        System.out.println("Main method is executed.");
    }
}
```

Output:

```
vbnet
Copy code
Static block is executed.
Main method is executed.
```

4. Static Classes

In Java, only nested classes (classes within other classes) can be declared as `static`. A static nested class can be instantiated without an instance of the outer class, and it cannot access the instance variables or methods of the outer class.

```
java
Copy code
public class OuterClass {
    static class StaticNestedClass {
        void display() {
            System.out.println("Inside static nested class.");
        }
    }

    public static void main(String[] args) {
        OuterClass.StaticNestedClass nested = new
OuterClass.StaticNestedClass();
        nested.display(); // Outputs: Inside static nested class.
    }
}
```

Summary

- **Static Variables**: Shared across all instances of the class. Initialized once.
- **Static Methods**: Can be called without an instance. Can only access other static members.
- **Static Blocks**: Used for static initialization when the class is loaded.

- **Static Nested Classes**: Can be instantiated without an instance of the outer class and cannot access non-static members of the outer class.