

The background is a solid teal color. Scattered across the frame are approximately 12 birds, likely pigeons, in various stages of flight. They are dark in color with lighter, possibly iridescent, wingtips. The birds are positioned at different heights and angles, creating a sense of movement and depth. The central text is white and stands out against the teal background.

Non Access Modifiers

- The non-access modifiers in java are
 - Static
 - Final
 - Abstract
 - Synchronized
 - Transient
 - volatile

Static modifier

- The static modifier can be used in
 - Variable and
 - methods

Static variable

- The static key word is used to create variables that will exist independently of any instances created for the class.
- Only one copy of the static variable exists regardless of the number of instances of the class.
- Static variables are also known as class variables.
- Local variables cannot be declared static.

Static method

- The static key word is used to create methods that will exist independently of any instances created for the class.

Static methods do not use any instance variables of any object of the class they are defined in.

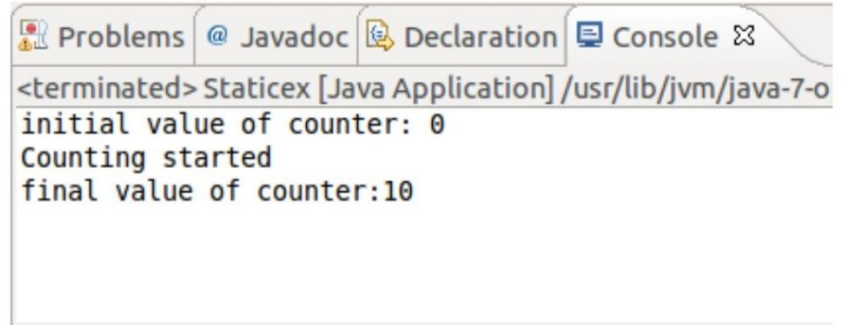
- Static methods take all the data from parameters and compute something from those parameters, with no reference to variables.
- Class variables and methods can be accessed using the class name followed by a dot and the name of the variable or method.

Example of static modifier

```
package day6;

class Staticmodifier
{
    private static int counter = 0;
    static void addcounter()
    {
        counter++;
    }
    static int getCount()
    {
        return counter;
    }
    Staticmodifier()
    {
        Staticmodifier.addcounter();
    }
}

public class Staticex
{
    public static void main(String args[])
    {
        System.out.println("initial value of counter: " + Staticmodifier.getCount());
        System.out.println("Counting started");
        for(int i=0;i<10;i++)
        {
            new Staticmodifier();
        }
        System.out.println("final value of counter:" + Staticmodifier.getCount());
    }
}
```



```
<terminated> Staticex [Java Application] /usr/lib/jvm/java-7-o
initial value of counter: 0
Counting started
final value of counter:10
```

Final modifier

- The final modifier can be given to
 - Variable and
 - Method
 - Class

Final variable

- A final variable can be explicitly initialized only once.
- A reference variable declared final can never be reassigned to refer to an different object.
- The data within the object can be changed. So the state of the object can be changed but not the reference.

Final method

- A final method cannot be overridden by any subclasses.
- The final modifier prevents a method from being modified in a subclass.

Final class

- The main purpose of using a class being declared as final is to prevent the class from being subclassed.
- If a class is marked as final then no class can inherit any feature from the final class.

Final modifier sample code

```
package day6;

class Final1
{
    final int a =10;
    int out()
    {
        return a;
    }
    @Override
    public String toString()
    {
        return a+"";
    }
    Final1()
    {
        out();
    }
}

public class Finalex
{
    public static void main(String[] args)
    {
        System.out.println(new Final1().toString());
        Final1 f = new Final1();
        //f.a =12;
    }
}
```

Problems @ Javadoc Declaration Console

<terminated> Finalex [Java Application] /usr/lib/jvm/java-7

10

Synchronized modifier

- The synchronized key word used to indicate that a method can be accessed by only one thread at a time.
- The synchronized modifier can be applied with any of the four access level modifiers.

Example

```
public synchronized add()  
{.....}
```

Transient modifier


- An instance variable is marked transient to indicate the JVM to skip the particular variable when serializing the object containing it.
- This modifier is included in the statement that creates the variable, preceding the class or data type of the variable.
- Example

`transient int a = 10; // will remain forever`

`int a1 = 20; // will be deallocated after its scope.`

Volatile modifier

- The volatile is used to let the JVM know that a thread accessing the variable must always merge its own private copy of the variable with the master copy in the memory.
- Accessing a volatile variable synchronizes all the cached copied of the variables in the main memory.
- Volatile can only be applied to instance variables, which are of type object or private.
- A volatile object reference can be null.



**“Everything has
beauty, but not
everyone sees it.”**

—CONFUCIUS

