Java - Variables

What is Variables ?

In Java, **variables** are containers used to store data values. Each variable has a specific type that determines the size and layout of the memory, the range of values that can be stored, and the operations that can be performed on it.

Types of Variables in Java

Java variables are categorized into three main types:

1. Local Variables

A variable declared inside the body of the method is called local variable. You can use this variable only within that method and the other methods in the class aren't even aware that the variable exists.

A local variable cannot be defined with "static" keyword.

▼ Summary

- Declared inside a method, constructor, or block.
- Scope is limited to the block in which they are declared.
- Must be initialized before use.

```
Copy codepublic class Example {
    public void display() {
        int localVar = 10; // Local variable
        System.out.println("Local Variable: " + localVar);
    }
}
```

2. Instance Variables

A variable declared inside the class but outside the body of the method, is called an instance variable. It is not declared as `static`

It is called an instance variable because its value is instance-specific and is not shared among instances.

▼ Summary

- Declared inside a class but outside any method, constructor, or block.
- Each object of the class has its own copy of the instance variable.
- Default values are assigned (e.g., `0` for integers, `null` for objects).

```
public class Example {
    int instanceVar; // Instance variable

    public void display() {
        System.out.println("Instance Variable: " + instanceVar);
    }
}
```

3. Static (Class) Variables

A variable that is declared as static is called a static variable. It cannot be local. You can create a single copy of the static variable and share it among all the instances of the class. Memory allocation for static variables happens only once when the class is loaded in the memory.

▼ Summary

- Declared using the `static` keyword inside a class but outside any method, constructor, or block.
- Shared among all objects of the class.
- Memory is allocated only once at the class level.

```
public class Example {
    static int staticVar = 100; // Static variable

    public void display() {
        System.out.println("Static Variable: " + staticVar);
    }
}
```

Variable Declaration and Initialization

To declare a variable in Java, use the following syntax:

```
type variableName = value;
```

Example:

```
int age = 25;          // Integer variable
String name = "John";  // String variable
float salary = 5000.5f; // Float variable
boolean isActive = true; // Boolean variable
```

Rules for Naming Variables

1. Must start with a letter, `_`, or `$`.
2. Cannot start with a digit.
3. Cannot use reserved keywords (e.g., `class`, `int`).
4. Case-sensitive (e.g., `age` and `Age` are different).

Default Values of Variables

Data Type	Default Value
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'A' or 'a'
Boolean	True or False
Object	null

Example Program

```
public class VariableExample {
    int instanceVar = 10; // Instance variable
    static int staticVar = 20; // Static variable

    public void display() {
        int localVar = 30; // Local variable
        System.out.println("Instance Variable: " + instanceVar);
        System.out.println("Static Variable: " + staticVar);
        System.out.println("Local Variable: " + localVar);
    }

    public static void main(String[] args) {
        VariableExample obj = new VariableExample();
        obj.display();
    }
}
```

Output:

```
Instance Variable: 10
Static Variable: 20
Local Variable: 30
```

Key Points

- Local variables must be initialized before use.
- Instance variables are object-specific.
- Static variables are shared across all objects of the class.