Object Oriented Programming

JAVA

Abdulaziz Abdullah Aisari | Spark2Code | 21/07/2025

Introduction

Object-Oriented Programming (OOP) is a key part of Java. It allows developers to build programs using objects that combine data and behavior. Java uses classes and objects to help organize code, making it easier to understand, reuse, and maintain. This report introduces the basic ideas of OOP in Java and explains why it's an important approach in software development[1].

Main Concepts of OOP in Java

Java is built around four main principles of Object-Oriented Programming:

- Encapsulation: Means wrapping data (variables) and code (methods) together into a single unit called a **class**.

```java
public class Person {
    private String name;

    public void setName(String newName) {
        name = newName;
    }

    public String getName() {
        return name;
    }
}
```

- Inheritance: Allows a class to **reuse** the properties and methods of another class. The new class is called a **subclass**, and the original class is the **superclass**.

```java
class Animal {
    void sound() {
        System.out.println("Animal makes a sound");
    }
}

class Dog extends Animal {
    void sound() {
        System.out.println("Dog barks");
    }
}
```

- Polymorphism: Means having many forms. In Java, this allows one method to behave differently based on the object calling it, typically using **method overriding** or **overloading method**.

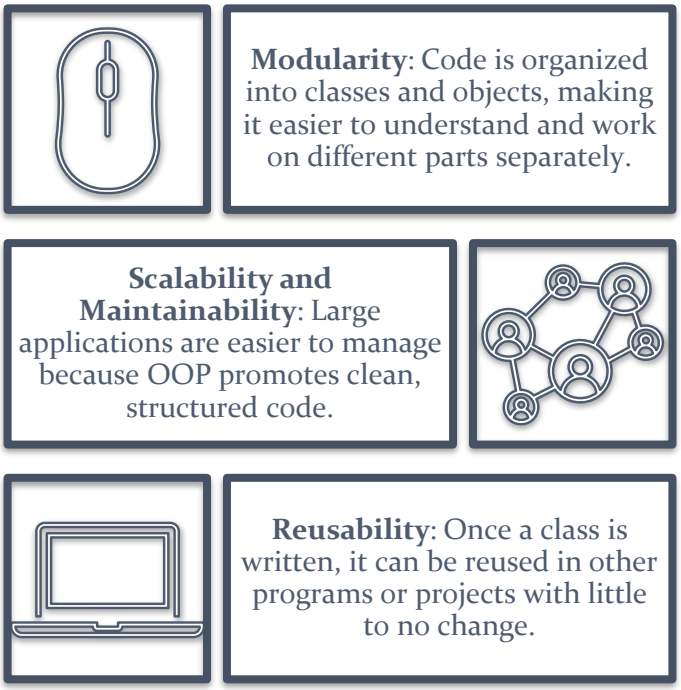
```
Animal a = new Dog();
a.sound();  // Outputs: Dog barks
```

- Abstraction: Is about hiding unnecessary details and showing only the essential parts of the object. In Java, it's done using **abstract classes** or **interfaces**.

```
abstract class Shape {
    abstract void draw();
}

class Circle extends Shape {
    void draw() {
        System.out.println("Drawing Circle");
    }
}
```

Advantages OOP in Java:



Disadvantages OOP in Java:

Complexity for Small Programs: For very small tasks, using OOP can feel like overkill due to extra structure and setup.

Memory and Performance Overhead: Object creation and management can consume more memory and processing time compared to procedural code.

Learning Curve: Beginners may find OOP concepts like inheritance and polymorphism hard to grasp at first.

Reference:

[1] Oracle. (n.d.). *Object-oriented programming concepts*. Retrieved from https://docs.oracle.com/javase/tutorial/java/concepts/