OBJECT ORIENTED PROGRAMMING

OBJECT ORIENTED PROGRAMMING I

Sercan Külcü | Object Oriented Programming I | 10.01.2023

Contents

Introduction

Object-oriented programming (OOP) is a popular programming paradigm that enables developers to design complex software systems by creating objects that interact with each other. Java is a widely used programming language that has gained immense popularity due to its excellent support for object-oriented programming. In this chapter, we will delve into the fundamental concepts of object-oriented programming in Java and how to apply them to develop robust software solutions.

Classes and Objects

In Java, everything is an object. A class is a blueprint for creating objects. It defines the properties and behaviors of objects. To create an object, you must first define its class. For example, if you want to create an object of a car, you must define its class that includes its properties like model, make, and color, and its behaviors like accelerate, brake, and turn.

Encapsulation

Encapsulation is the mechanism of hiding the implementation details of an object from the outside world. In Java, encapsulation is achieved using access modifiers such as private, public, and protected. Private members can only be accessed within the class, whereas public members can be accessed from any class. Protected members can be accessed within the same package and its subclasses.

Inheritance

Inheritance is a mechanism by which a class can inherit the properties and behaviors of another class. The class that inherits the properties and behaviors is called a subclass or derived class, and the class from which it inherits is called the superclass or base class. In Java, inheritance is implemented using the keyword extends. A subclass can inherit all the properties and behaviors of its superclass and can also add new properties and behaviors of its own.

Polymorphism

Polymorphism is the ability of an object to take on many forms. In Java, polymorphism is implemented using method overloading and method overriding. Method overloading allows a class to have two or more methods with the same name but different parameters. Method overriding allows a subclass to provide its own implementation of a method that is already defined in its superclass.

Abstraction

Abstraction is the process of hiding complex implementation details and exposing only the essential features of an object. In Java, abstraction is implemented using abstract classes and interfaces. An abstract class is a class that cannot be instantiated but can be inherited. An interface is a collection of abstract methods that a class can implement.

Conclusion

Java's support for object-oriented programming makes it an ideal language for developing large-scale software systems. In this chapter, we have covered the fundamental concepts of object-oriented programming in Java, including classes and objects, encapsulation, inheritance, polymorphism, and abstraction. Understanding these concepts is crucial for developing robust software solutions.