Day 10: Object-Oriented Programming in Java - Inheritance and Polymorphism!

Yesterday, we explored the basics of classes and objects in Java. Today, we're going to dive deeper into object-oriented programming (OOP) and learn about two fundamental concepts: inheritance and polymorphism!

Inheritance

Inheritance is a fundamental concept in OOP that allows one class to inherit the properties and behavior of another class. In Java, you can use the **extends** keyword to create a subclass that inherits from a superclass.

Here's an example:

```
public class Animal {
    private String name;

    public Animal(String name) {
        this.name = name;
    }

    public void sound() {
        System.out.println("The animal makes a sound.");
    }
}

public class Dog extends Animal {
    public Dog(String name) {
        super(name);
    }

    @Override
    public void sound() {
        System.out.println("The dog barks.");
    }
}
```

Polymorphism

Polymorphism is another key concept in OOP that allows objects of different classes to be treated as objects of a common superclass. In Java, you can use method overriding or method overloading to achieve polymorphism.

Here's an example:

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

public class Dog extends Animal {
    @Override
    public void sound() {
        System.out.println("The dog barks.");
    }
```

```
}

public class Cat extends Animal {
    @Override
    public void sound() {
        System.out.println("The cat meows.");
    }
}

public class Main {
    public static void main(String[] args) {
        Animal myAnimal = new Dog("Fido");
        myAnimal.sound(); // Output: The dog barks.

        myAnimal = new Cat("Whiskers");
        myAnimal.sound(); // Output: The cat meows.
    }
}
```

Challenges

Create a subclass called **Square** that inherits from **Rectangle**, and overrides the **area()** and **perimeter()** methods. Create an array of **Animal** objects, and use polymorphism to call the **sound()** method on each object.

Code

You can find the code for today's challenges in the Day10 folder of this repository.

Join the Conversation

How was your experience with inheritance and polymorphism in Java today? Did you encounter any challenges or have any questions? Share your thoughts in the comments below!