Programación en Java – Clase 7

La clase pasada comenzamos a trabajar sobre la programación orientada a objetos (POO). En esta clase, profundizaremos el trabajo sobre este paradigma a través de dos conceptos: herencia y polimorfismo.

En Java, el polimorfismo es un concepto fundamental de la programación orientada a objetos que permite a los objetos de diferentes clases ser tratados de manera uniforme a través de una referencia común.

El polimorfismo se basa en dos conceptos principales: la herencia y los métodos sobrescritos.

1. **Herencia:** En Java, una clase puede heredar propiedades y comportamientos de otra clase mediante la declaración de una relación de herencia. La clase que hereda se denomina clase derivada o subclase, y la clase de la que se heredan se denomina clase base o superclase. El polimorfismo se logra cuando se utiliza una referencia de la clase base para apuntar a un objeto de la clase derivada.

2. **Métodos sobrescritos:** En el polimorfismo, los métodos de la clase base pueden ser sobrescritos en las clases derivadas para proporcionar una implementación específica. Cuando se llama a un método a través de una referencia de la clase base, el método invocado será el correspondiente a la clase derivada en tiempo de ejecución.

El polimorfismo permite tratar objetos de diferentes clases de manera uniforme a través de una referencia común, lo que ofrece varias ventajas, como la flexibilidad y la extensibilidad del código. Permite escribir código genérico y reutilizable, ya que las clases derivadas pueden compartir una interfaz común y proporcionar su propia implementación específica.

Veamos un ejemplo:

```
class Animal {
   public void hacerSonido() {
      System.out.println("El animal hace un sonido.");
   }
}
```

```
class Perro extends Animal {
    @Override
    public void hacerSonido() {
        System.out.println("El perro ladra.");
    }
}

class Gato extends Animal {
    @Override
    public void hacerSonido() {
        System.out.println("El gato maulla.");
    }
}

public class Main {
    public static void main(String[] args) {
        Animal animal1 = new Perro();
        Animal animal2 = new Gato();

        animal1.hacerSonido(); // Salida: El perro ladra.
        animal2.hacerSonido(); // Salida: El gato maulla.
    }
}
```

En este ejemplo, tenemos una clase base **Animal** con un método **hacerSonido()**. Las clases **Perro** y **Gato** son clases derivadas de **Animal** que sobrescriben el método **hacerSonido()** con su propia implementación. En el método main(), se crean objetos de las clases **Perro** y **Gato** y se asignan a referencias de tipo `Animal`. A pesar de que las referencias son de tipo `Animal`, cuando se llama al método `hacerSonido()`, se ejecuta la implementación correspondiente de la subclase en tiempo de ejecución.

Este es solo un ejemplo básico del polimorfismo en Java, pero muestra cómo las referencias de una clase base pueden apuntar a objetos de diferentes clases derivadas y cómo se invoca la implementación específica en tiempo de ejecución.