



TEMA 10

Interfaces y Polimorfismo

V1.1

Manuel Pereira González

Agenda



- **Interfaces**
 - **Declaración**
 - Implementación
 - Referencias a Interfaces
- Polimorfismo
- Resumen



Interfaces: Declaración



- Una **interfaz** define un conjunto de métodos sin implementación
- Para declarar una interfaz, se utiliza la palabra clave **interface** en lugar de class
- Una interfaz sólo puede tener métodos públicos (todos los métodos de una interfaz tienen que declararse de tipo public).
- Una interfaz sólo puede contener atributos de tipo static y final (constantes de clase)

```
public interface Movable {  
    public void move(int x, int y);  
}
```

Interfaces: Declaración



- Principales diferencias entre una **interfaz** y una clase **abstract**:
 - Una clase abstracta puede tener métodos que no lo son. Ninguno de los métodos de una interfaz puede tener implementación (una interfaz no puede tener código en su declaración).
 - En java no existe la herencia múltiple, es decir, una subclase no puede heredar a la vez de dos clases. Una clase **sí que puede implementar varias interfaces**, no existe limitación en cuanto a implementación múltiple de interfaces

Agenda



- Interfaces
 - Declaración
 - **Implementación**
 - Referencias a Interfaces
- Polimorfismo
- Resumen



Interfaces: Implementación



- Para indicar que una clase implementa los métodos de una interface se utiliza la palabra clave **implements**.
- Muchas clases pueden implementar una interfaz
- Para implementar una interfaz, la clase debe proporcionar una implementación de todos los métodos que define la interfaz.

Interfaces: Implementación



```
public interface Movible {
    public void moverse(int x, int y);
}

public class Vehiculo {
    public int numRuedas;

    public Vehiculo(int numRuedas) {
        this.numRuedas = numRuedas;
    }
}

public class Coche extends Vehiculo implements Movible {
    String color;
    public int posicionX, posicionY;

    public Coche(String color) {
        super(4);
        this.color = color;
    }

    public void moverse(int x, int y) {
        posicionX += x;
        posicionY += y;
    }
}

public abstract class SerVivo {
    public boolean vivo;

    public SerVivo() {
        vivo = true;
    }

    public void morir() {
        vivo = false;
    }

    public abstract void reproducirse();
}

public class Persona extends SerVivo implements Movible {
    public String nombre;
    public int numHijos;
    public int posicionX, posicionY;

    public Persona(String nombre) {
        this.nombre = nombre;
        numHijos = 0;
    }

    public void reproducirse() {
        numHijos++;
    }

    public void moverse(int x, int y) {
        posicionX += x;
        posicionY += y;
    }
}
```

Agenda



- Interfaces
 - Declaración
 - Implementación
 - **Referencias a Interfaces**
- Polimorfismo
- Resumen



Referencias a Interfaces



- Es posible crear referencias a interfaces

```
public class ReferenciasAInterfaces {  
    public static void main(String[] args) {  
        Persona p = new Persona("Juan");  
        // Crea una referencia a objeto movable  
        Movable mv = p;  
        mv.moverse(1, 2);  
    }  
}
```

- Sin embargo, las interfaces no pueden ser instanciadas

```
public class ReferenciasAInterfaces {  
    public static void main(String[] args) {  
        // Error de compilación, no puede instanciarse una interfaz  
        Movable mv = new Movable();  
        mv.moverse(1, 2);  
    }  
}
```

Agenda



- Interfaces
 - Declaración
 - Implementación
 - Referencias a Interfaces
- **Polimorfismo**
- Resumen



Polimorfismo



- Polimorfismo
 - Tratar objeto de una clase más general independientemente de que sea de una clase concreta.
 - Ej: Si tengo un perro, un elefante y una jirafa, puedo tratarlos a todos como animales
 - Ej: Polígono -> Método para calcular el perímetro
 - Círculo: $2 * PI * R$
 - Rectángulo: $Base * Altura$
 - Triángulo: $Base * Altura / 2$
 - Lista de polígonos, cada uno sabe calcular su área pero se tratan de igual manera sin saber de qué tipo de polígono concreto se trata

Polimorfismo



```
public abstract class Figura {
    public abstract float calcularArea();
}

public class Circulo extends Figura {
    public float radio;
    public static final float PI = 3.14f;

    public Circulo(float radio) {
        this.radio = radio;
    }

    public float calcularArea() {
        return PI * radio * radio;
    }
}

public class Triangulo extends Figura {
    public float base, altura;

    public Triangulo(float base,
        float altura) {
        this.base = base;
        this.altura = altura;
    }

    public float calcularArea() {
        return base * altura / 2f;
    }
}

public class EjemploPolimorfismoClases {
    public static void main(String[] args) {
        Triangulo t1 = new Triangulo(3, 5);
        Triangulo t2 = new Triangulo(4, 3);
        Circulo c1 = new Circulo(3);

        Figura[] figuras = new Figura[3];
        figuras[0] = t1;
        figuras[1] = t2;
        figuras[2] = c1;

        imprimeAreas(figuras);
    }

    public static void imprimeAreas(Figura[] figs) {
        for(int i=0; i<figs.length; i++) {
            System.out.print("Área de la figura " + i + ": ");
            System.out.println(figs[i].calcularArea());
        }
    }
}

C:\Practicas\lecciones\classes>java EjemploPolimorfismoClases
Área de la figura 0: 7.5
Área de la figura 1: 6.0
Área de la figura 2: 28.26
C:\Practicas\lecciones\classes>
```

Polimorfismo



```
public class MoverDeObjetos {
    public static void mueveDerecha(Movible m) {
        m.moverse(1, 0);
    }
    public static void mueveIzquierda(Movible m) {
        m.moverse(-1, 0);
    }
    public static void mueveArriba(Movible m) {
        m.moverse(0, -1);
    }
    public static void mueveAbajo(Movible m) {
        m.moverse(0, 1);
    }
}

public class EjemploPolimorfismo {
    public static void main(String[] args) {
        Persona p = new Persona("Juan");
        Coche c = new Coche("Rojo");

        System.out.println("Posición persona x=" + p.posicionX + ", y=" + p.posicionY);
        System.out.println("Posición coche x=" + c.posicionX + ", y=" + c.posicionY);

        MoverDeObjetos.mueveDerecha(p);
        MoverDeObjetos.mueveArriba(c);

        System.out.println("Posición persona x=" + p.posicionX + ", y=" + p.posicionY);
        System.out.println("Posición coche x=" + c.posicionX + ", y=" + c.posicionY);
    }
}
```

cmd Símbolo del sistema

```
C:\Practicas\lecciones\classes>java EjemploPolimorfismo
Posición persona x=0, y=0
Posición coche x=1, y=0
Posición persona x=1, y=0
Posición coche x=0, y=-1
C:\Practicas\lecciones\classes>
```

Resumen: Para más información



- <http://www.arrakis.es/~abelp/ApunteSJava/Interfaces.htm>
- http://pisuerga.inf.ubu.es/lsi/Invest/Java/Tuto/II_9.htm
- <http://java.sun.com/docs/books/tutorial/java/concepts/interface.html>