## **S07.s1 - Implementación de programas usando polimorfismo**

## **¿Qué es el polimorfismo en programación y cuál es su propósito en Java?**

El **polimorfismo** es la capacidad de que un mismo método o acción pueda tener **múltiples formas de implementación** dependiendo del objeto que lo ejecute.

En **Java**, el propósito principal es:

* **Tratar a diferentes objetos a través de una misma referencia común** (superclase o interfaz).
* Permitir que el programa sea más **flexible y extensible**, ya que el comportamiento exacto se decide en tiempo de ejecución.

## **¿Cuál es la diferencia clave entre una clase abstracta y una interfaz en Java en el contexto del polimorfismo?**

| **Aspecto** | **Clase Abstracta** | **Interfaz** |
| --- | --- | --- |
| **Herencia** | Una clase solo puede heredar de **una** clase abstracta. | Una clase puede implementar **múltiples interfaces**. |
| **Contenido** | Puede tener métodos abstractos **y** métodos con implementación. También puede tener atributos. | Solo declara métodos (hasta Java 7). Desde Java 8+, puede tener default y static methods, pero no atributos de instancia. |
| **Propósito en polimorfismo** | Define una **base común** con comportamiento parcial que las subclases heredan y completan. | Define un **contrato** que cualquier clase puede implementar, garantizando que cumpla con ciertos métodos. |

En polimorfismo:

* **Clases abstractas** sirven cuando varias clases comparten cierta lógica común.
* **Interfaces** se usan cuando distintas clases (aunque no estén relacionadas entre sí) deben cumplir un mismo comportamiento.

**¿Por qué es útil utilizar el polimorfismo al diseñar programas en Java?**

* **Reduce la complejidad:** Se puede trabajar con referencias genéricas en lugar de tipos específicos.
* **Aumenta la flexibilidad:** Se pueden añadir nuevas clases (con nuevos comportamientos) sin cambiar el código existente.
* **Facilita la mantenibilidad:** Cambios en una clase no afectan directamente a otras si cumplen con el mismo contrato.
* **Favorece la reutilización del código:** Métodos genéricos funcionan para múltiples tipos de objetos.

**CIERRE**

**¿Qué es el polimorfismo y cómo se relaciona con la programación en Java?**

El **polimorfismo** significa *“muchas formas”*. En programación orientada a objetos es la capacidad de que un mismo método o acción pueda tener **diferentes comportamientos** dependiendo del objeto que lo ejecute.

En **Java**, el polimorfismo permite:

* Usar **referencias genéricas** (superclases o interfaces) para manipular objetos de distintas clases.
* Hacer que el método que realmente se ejecute se decida en **tiempo de ejecución**

**¿Cuál es la ventaja principal de usar el polimorfismo en la programación?**

La ventaja principal es la **flexibilidad**:

* Permite **manejar diferentes tipos de objetos de manera uniforme**.
* Facilita la **extensibilidad** del programa: se pueden agregar nuevas clases sin modificar el código existente.
* Reduce el uso de condicionales (if/switch) para decidir qué acción ejecutar.

## **¿Cómo se implementa el polimorfismo en Java y cuál es su impacto en la reutilización de código?**

### **Implementación en Java:**

1. **Herencia (clases abstractas)**
   * Una superclase define métodos comunes.
   * Las subclases heredan y sobreescriben estos métodos.
2. **Interfaces**
   * Definen un contrato común.
   * Varias clases implementan la interfaz de manera distinta.
3. **Sobrescritura de métodos (@Override)**
   * Permite que una subclase redefina el comportamiento de un método de su padre.