Conceptos de POO

Pedro O. Pérez M., PhD.

Desarrollo de videojuegos con Java
Tecnológico de Monterrey

pperezm@tec.mx

10-2023

Programación Orientada a Objetos

- ¿Qué es la Programación Orientada a Objetos?

Conceptos base

- Modificadores de acceso
- Clase
- Métodos
- Objeto

Características de POO

- Abstracción
- Encapsulación
- Herencia
- Polimorfismo

¿Qué es la Programación Orientada a Objetos?

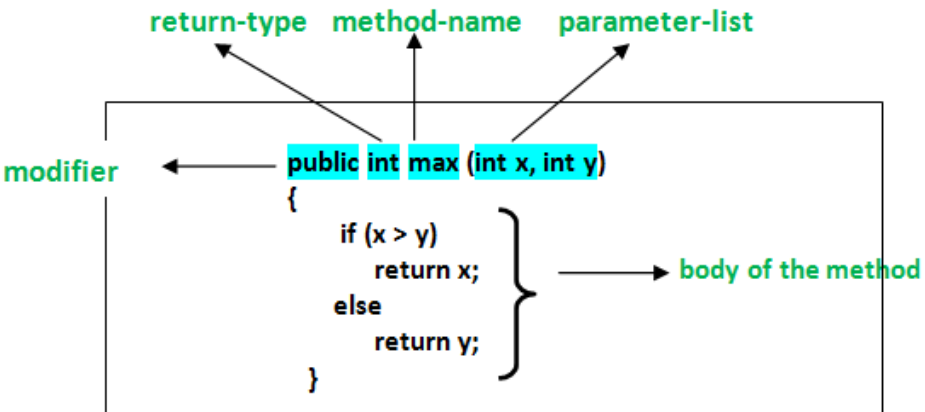
- Este paradigma de programación parte del concepto de "objeto" como base. Estos objetos son elementos autosuficientes de un programa que representa un grupo de características relacionadas entre sí y diseñados para realizar una tarea dada; cada objeto contienen información en forma de campos (a veces también referidos como atributos o propiedades) y código en forma de métodos.
- Los objetos son capaces de interactuar y modificar los valores contenidos en sus campos o atributos (estado) a través de sus métodos (comportamiento).

Clase

- Una clase es un modelo o prototipo por el usuario a partir del cuál se crearán objetos. Representa el conjunto de propiedades o métodos que son comunes a todos los objetos de un tipo. Usando clases, puedes crear múltiples objetos con el mismo comportamiento en lugar de escribir su código varias veces.

Métodos

- El nombre de un método suele ser una sola palabra que debe ser un verbo en minúsculas o varias palabras, que comienza con un verbo en minúsculas seguido de un adjetivo, sustantivo. Después de la primera palabra, la primera letra de cada palabra debe ser en mayúscula.



En Java,

- Los argumentos se pasan a los métodos por valor; se hace una copia del tipo de dato primitivo, como `int`, `float`, etcétera, o la referencia al objeto de la clase o un arreglo, y se pasa al método.
- Es posible sobrecargar métodos, es decir, definir dos o más dentro de la misma clase, que compartan nombre y que las declaraciones de sus parámetros sean diferentes; la sobrecarga es una forma de polimorfismo.
- Los métodos abstractos se definen en clases para imponer funcionalidad a las subclases; una clase con un método abstracto es abstracta; estos métodos se definen por la palabra reservada `abstract` y no tienen cuerpo; su implementación se deja a las subclase.

Una clase en java

Listing 1: Archivo: Example2.java

```
public class Example2 {
        private String text;

        public Example2() {
                text = null;
        }

        public Example2(String text) {
                this.text = text;
        }

        public String getText() {
                return text;
        }

        public void setText(String text) {
                this.text = text;
        }

        public static void main(String args[]) {
                Example2 obj = new Example2("This is an example");

                System.out.println(obj.getText());
        }
}
```

Objeto

- Un objeto es una unidad básica de programación orientada a objetos que representa entidades de la vida real. Un típico programa en Java crea muchos objetos que interactúan invocando métodos.
- Un objeto se compone principalmente de:
  - **Estado:** Está representado por los atributos de un objeto.
  - **Comportamiento:** Está representado por los métodos de un objeto. Un método es una colección de declaraciones que realizan una tarea específica y devuelven un resultado. Un método puede realizar alguna tarea específica sin devolver nada.

Creación de un objeto

- Cuando se crea un objeto de una clase, se dice que la clase tiene una instancia. Todas las instancias comparten los atributos y el comportamiento de la clase.
- En Java, si declaramos una instancia (o variable de referencia), su valor será indeterminado (`null`) hasta que se cree y se le asigne un objeto. La simple declaración de una variable de referencia no crea un objeto.
- El operador `new` crea una instancia de una clase asignando memoria para un nuevo objeto y devolviendo una referencia a esa memoria. El operador `new` también invoca al constructor de la clase.

Abstracción

- La Abstracción de Datos es la propiedad en virtud de la cual sólo se muestran al usuario los detalles esenciales. Las características triviales o no esenciales no se muestran al usuario.
- La Abstracción de Datos también puede definirse como el proceso de identificar sólo las características requeridas de un objeto, ignorando los detalles irrelevantes.
- En Java, la abstracción se logra mediante interfaces y clases abstractas. Podemos lograr un 100 % de abstracción utilizando interfaces.

Encapsulación

- Se define como la agrupación de datos en una sola unidad. Es el mecanismo que une el código y los datos que manipula. Otra forma de pensar en la encapsulación es que es un escudo protector que evita que cualquier entidad fuera de este escudo acceda a los datos.
- Técnicamente, en encapsulación, las variables o los datos de una clase están ocultos de cualquier otra clase y solo se puede acceder a ellos a través de cualquier función miembro de la clase en la que están declarados.
- En la encapsulación, los datos de una clase se ocultan de otras clases, lo cual es similar a lo que hace la ocultación de datos. Por tanto, los términos "encapsulación" y "ocultación de datos" se utilizan indistintamente.
- La encapsulación se puede lograr declarando todas las variables de una clase como privadas y escribiendo métodos públicos en la clase para establecer y obtener los valores de las variables.

Herencia

- La herencia es un elemento importante de la POO. Es el mecanismo mediante el cual se permite a una clase heredar las características (atributos y métodos) de otra clase. En Java, la herencia se realiza mediante el uso de la palabra reservada `extends`. La herencia también se conoce como relación "es-un".
- Analicemos algunas terminologías importantes de uso frecuente:
  - **Superclase:** La clase cuyas características se heredan se conoce como superclase.
  - **Subclase:** La clase que hereda de otra clase se conoce como subclase. La subclase puede agregar sus propios campos y métodos además de los campos y métodos de la superclase.
  - **Reutilización:** Cuando creamos una nueva clase a partir de la clase existente estamos reutilizando los atributos y métodos de la clase existente.

Polimorfismo

- Se refiere a la propiedad por la que es posible enviar mensajes sintácticamente iguales a objetos de tipos distintos. Aunque el mensaje sea el mismo, diferentes objetos pueden responder a él de manera única y específica. Esta característica permite que, sin alterar ni tocar el código existente, se puedan incorporar nuevos comportamientos y funciones (es decir la interfaz sintáctica se mantiene inalterada pero cambia el comportamiento en función de qué objeto estamos usando en cada momento). El único requisito es que los objetos deben ser capaces de responder al mensaje que se les envía, garantizando así una flexibilidad y extensibilidad en el diseño del software.