19/08/25 – BACKEND DE APLICACIONES

POO EN JAVA

Los lenguajes orientados a objetos usan descriptores de entidades llamados clases. Una clase es la descripción de una entidad u objeto de forma de que se pueda usar para crear objetos que respondan a la descripción dada. La clase es única y describe la forma y el comportamiento de los muchos objetos de la clase. Una clase incluye:

* Atributos: variables que se declaran dentro de la clase y sirven para indicar la forma de cada objeto representado por esa clase. Muestran lo que cada objeto es o tiene.
* Métodos: funciones, procedimientos o rutinas declaradas dentro de la clase usados para describir el comportamiento de los objetos. Muestran lo que cada objeto hace.

Cada aplicación típica de java necesita incluir el método especial main() que es el primero que se ejecuta al lanzar la aplicación y desde allí se administrar la participación de cada objeto creado.

La definición de la clase en Java se hace mediante la palabrea reservada class{}. Es común que los atributos de la clase se declaren antes que los métodos. El conjunto de atributos y métodos aparecen como el conjunto de miembros de la clase. Los atributos son variables que pueden ser primitivos o ser objetos de otras clases.

Principio del Ocultamiento: al definir una clase no se debe permitir que los atributos sean accesibles de forma directa desde el exterior de la clase, sino que se deben usar métodos para consultar/modificar sus valores. La idea es que el programador que use una clase predefinida no deba preocuparse por los detalles de la implementación interna de la clase, sino que simplemente use los métodos que esta provee y maneje un nivel de abstracción mas alto. Permite que la clase controle que valores tiene cada atributo y de qie forma esos valores deberían cambiar.

Modificadores de acceso: forma que provee java para que se obligue a respetar el Principio de Ocultamiento. Son palabras reservadas que colocadas delante de la declaración de un atributo o de un método de una clase hacen que ese atributo/método tenga una accesibilidad más o menos amplia:

* Public: un miembro publico es accesible desde el interior (métodos propios) como del exterior (por métodos de otras clases).
* Private: solo accesible desde el interior de la clase con métodos propios.
* Protected: aplicable en contexto de herencia. Hace que un miembro sea publico para sus clases derivadas u para clases del mismo paquete, pero privados para el resto.
* Default: si no se ha marcado con ningún modificador, significa que será publico para las clases del paquete, pero privado para el resto. No es una palabra reservada.

Entre los métodos de la clase encontramos el método constructor, cuyo objetivo básico es inicializar los atributos de un objeto en el momento en que se crea. Se utiliza el operador new y el constructor se invoca automáticamente al crear con new una instancia. El constructor lleva el mismo nombre que la clase que lo contiene. Tanto los constructores como otro método pueden ser sobrecargados, lo que significa que se puede definir varias versiones del mismo método. Si un método tiene varias versiones deben diferir en la cantidad de parámetros, en el tipo de parámetros o en ambas.

Métodos de acceso de lectura y escritura: para los de consulta se usa get y para los modificadores set. Los métodos de consulta son públicos y lo

Toda clase en java se define a partir de la clase Object. Algunos métodos deben ser redefinidos, como el método toString(); el cual si no se modifica retorna una cadena con el nombre de la clase y la dirección de memoria en formato hexadecimal.

Código Repetitivo: a este tipo de código se llama boiler plate y no aparta lógica de negocio, pero si es necesario. Una de las herramientas que sirve para reducir este tipo de código es Lombok.

Lombok es una biblioteca de java que permite eliminar la necesidad de escribir código repetitivo mediante anotaciones. Con solo agregar una anotación sobre una clase (@Getter, @Setter, @ToString, @AllArgsConstructor, etc.) se genera automáticamente el código correspondiente en tiempo de compilación. Es un comportamiento transparente, ya que no aparece en los archivos .java, pero si esta disponible en los archivos .class y puede usarse como si estuviera escrito a mano. Se basa en un mecanismo llamado annotation processing que permite analizar y modificar el código en tiempo de compilación. Para ello:

* Utiliza una librería interna que intercepta el compilador javac y modifica el árbol de sintaxis abstracta (AST) del código.
* Inyecta automáticamente el código necesario según las anotaciones.
* El bytecode generado por el compilador incluirá getters, setters, constructores, etc.

Todo esto ocurre en tiempo de compilación.

Para agregar la dependencia en el proyecto (en pom.xml):

Texto

El contenido generado por IA puede ser incorrecto.

|  |  |  |
| --- | --- | --- |
| lombok | Método | Uso |
| @ToString | toString() | Genera el toString incluyendo todos los atributos no static ni transient. |
| @Getter | get() | Genera el get() |
| @Setter | Set() | Genera el set() |
| @NoArgsConstructor | Public Clase() {} | Constructor Vacio |
| @AllArgsConstructor | Public Clase(e1) { this.algo = x;} | Constructor con todos los argumentos |
| @EqualsAndHashCode | Equals() y hashCode() | Se puede modificar agregando el paramentro (of = “atributo”) a la notación |
| @Data | getNumerador(), getDenominador() setNumerador(), setDenominador() toString(), equals(), hashCode() | Genera automáticamente todos los anteriores. Se encarga de los métodos de acceso, comparación y representación textual. |
| @Value | Equals(), hashCode(), toString(), gettters | Genera un record que es una nueva estructura de clase inmutable, pensada específicamente para modelar objetos portadores de datos (data carriers) de manera concisa, segura, legible. |

Texto

El contenido generado por IA puede ser incorrecto. 

Si iniciamos de cero una aplicación con las últimas versiones de java, los record son, quizás, la alternativa oficial a las clases inmutables típicas. Son más livianos, integrados y expresivos. Si trabajás con versiones anteriores o necesitás estilo JavaBeans (get/set) por herencia de dependencias o implementación de frameworks existentes, @Value sigue siendo una excelente opción, especialmente en proyectos con Lombok y Spring.