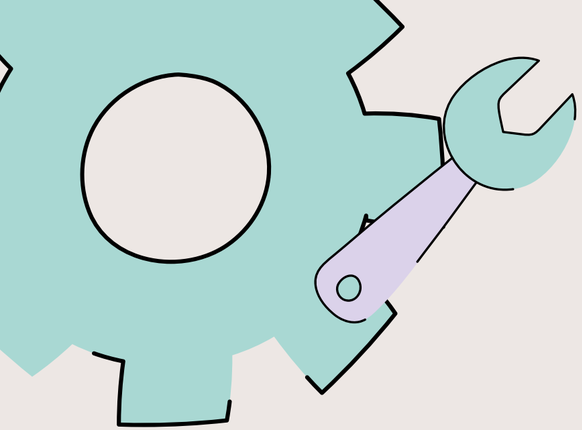
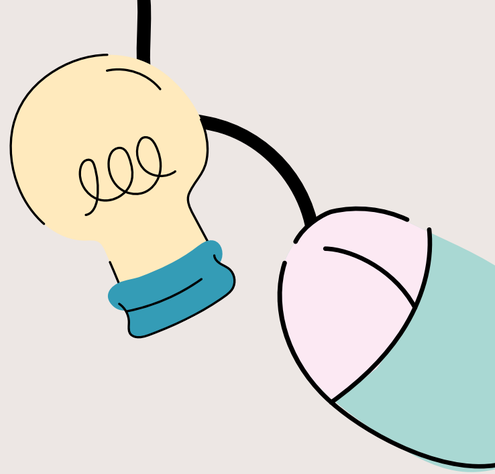
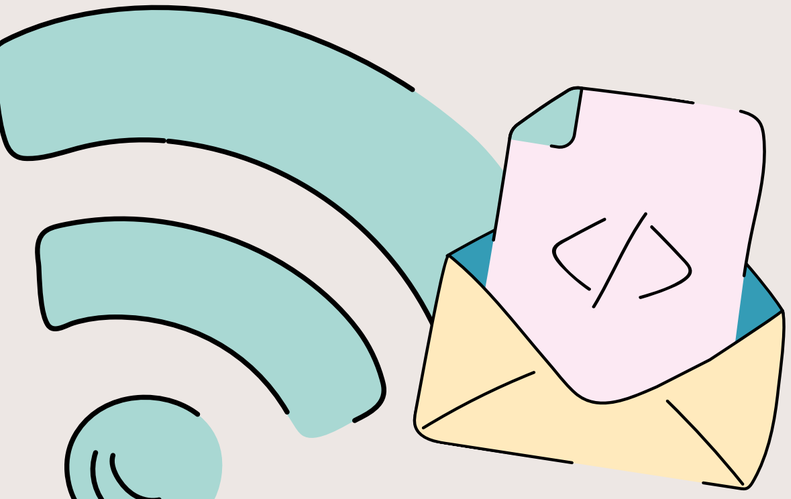
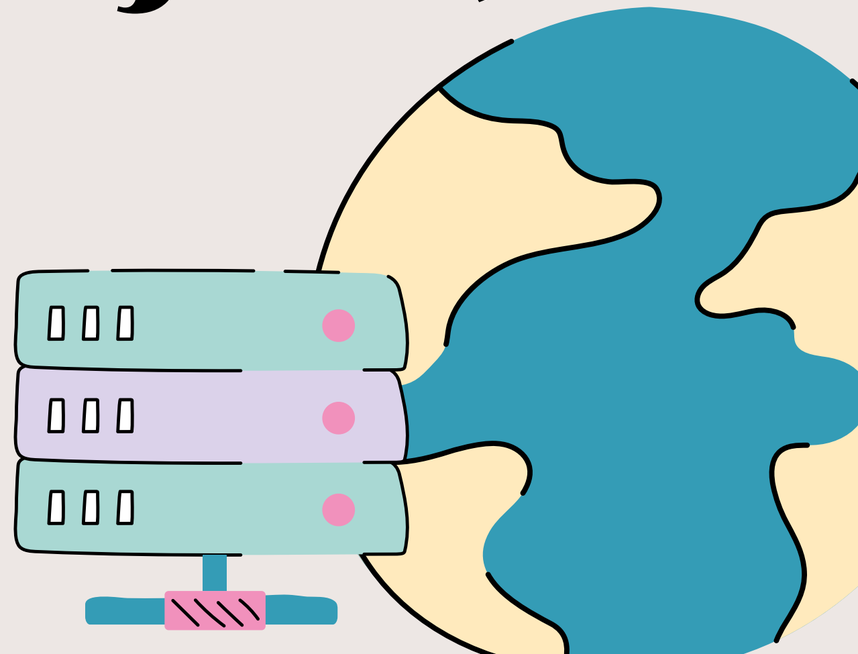




Patrón DAO (Data Access Object)

Josue Isai Carballo Vivas





¿Qué es el Patrón DAO

El Patrón Data Access Object (DAO) es un patrón de diseño estructural que proporciona una capa de abstracción entre la lógica de negocio de una aplicación y la capa de acceso a datos. Su propósito es encapsular todas las operaciones relacionadas con la base de datos en una clase específica para mejorar la mantenibilidad, reutilización y modularidad del código.

Caracteristicas

- **Abstracción del acceso a datos:** Separa la lógica de acceso a la base de datos de la lógica de negocio.
- **Independencia de la base de datos:** Permite cambiar de base de datos sin modificar la lógica de negocio.
- **Uso de interfaces y clases:** Define una interfaz DAO con operaciones estándar y su implementación concreta.
- **Facilidad de prueba:** Se pueden crear implementaciones falsas (mocks) para pruebas unitarias sin una base de datos real.
- **Encapsulación de operaciones CRUD:** Gestiona la creación, lectura, actualización y eliminación de datos de manera centralizada.
- **Facilidad de mantenimiento:** Al estar desacoplado, cualquier cambio en la BD solo afecta la capa DAO.

Estructura

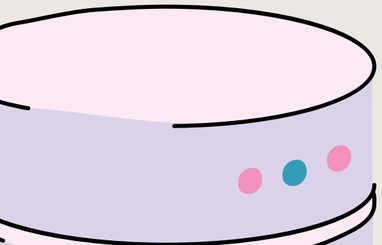
El Patrón DAO sigue una arquitectura estructurada que separa las responsabilidades en diferentes capas:

Entidad o Modelo de Datos: Representa los objetos de la aplicación que serán almacenados en la base de datos (Ejemplo: Usuario).

Interfaz DAO: Define los métodos estándar para acceder y manipular los datos (Ejemplo: obtenerUsuarioPorId(int id)).

Implementación DAO: Contiene la lógica concreta para interactuar con la base de datos usando SQL, JDBC, Hibernate, SQLAlchemy, etc.

Capa de Negocio: Consume los DAOs sin preocuparse por los detalles de la base de datos.



Ventajas de utilizar un Patrón DAO

Desacoplamiento entre lógica de negocio y acceso a datos

- La lógica de negocio no depende de la base de datos específica, lo que facilita modificaciones en la infraestructura.

Facilidad de mantenimiento y escalabilidad

- Si necesitas cambiar de MySQL a PostgreSQL o a una base de datos NoSQL, solo debes modificar la capa DAO sin afectar la lógica de negocio.

Reutilización del código

- Los métodos de acceso a datos pueden ser reutilizados en diferentes partes de la aplicación sin necesidad de repetir código SQL.

Pruebas unitarias más simples

- Al usar DAOs, es más fácil simular el acceso a datos con objetos mock, permitiendo pruebas unitarias sin depender de una base de datos real.

Desventajas de utilizar un Patrón DAO

Aumento de Clases y Archivos

- En aplicaciones grandes, cada entidad tiene su propio DAO, lo que genera muchas clases y archivos

Mantenimiento adicional

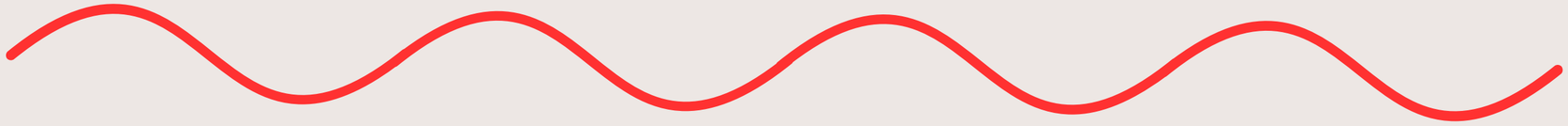
- Si la estructura de la base de datos cambia frecuentemente, mantener la implementación de los DAOs puede ser costoso.

Curva de aprendizaje

- Para desarrolladores sin experiencia en patrones de diseño, la estructura DAO puede resultar difícil de comprender inicialmente.

Rendimiento ligeramente afectado en algunas situaciones

- Agregar una capa extra de abstracción puede generar una leve sobrecarga en el rendimiento, especialmente si no se optimizan las consultas.



Conclusión

El Patrón DAO es una forma inteligente de manejar la base de datos sin mezclarla con el resto del código. Permite organizar mejor el proyecto, facilita los cambios y hace que el sistema sea más fácil de mantener.

Es muy útil en aplicaciones grandes donde se manejan muchos datos, pero puede ser innecesario en proyectos pequeños. Aunque requiere más trabajo al principio, a largo plazo hace que el código sea más limpio, seguro y fácil de actualizar.

¡GRACIAS POR SU ATENCIÓN!