DAO Pizzería **Jesús Villar**

El patrón DAO se emplea para separar la lógica de negocio de la lógica de acceso a datos con ventajas como que solo se tenga que cambiar la lógica de las clases DAO al cambiar de sistema de BBDD.

IConfiguration

Es una interfaz que define los métodos get() que se encargarán de obtener los datos de usuario, la contraseña y la url de la base de datos para ser implementados por la clase Configuration, la cual establecerá la conexión.

Configuration

Implementa los métodos de IConfiguration para establecer la conexión con un patrón Singleton.

Statement

Interfaz funcional con un método **run()**, el cual se encarga de ejecutar las acciones SQL que se quieren llevar a cabo, las cuales se le pasan cómo argumento junto a una entidad, en este caso de tipo Pizza.

IRunnables

Interfaz con 2 métodos: **getSQL()** y **run().**

getSQL() → Captura la transacción SQL

run() → Ejecuta un PreparedStatement o lanza una excepción SQLException.

Runnables

Implementación de la interfaz anterior con 3 atributos:

String sql

<T> Entity

Statement<T> (Interfaz functional)

Básicamente lo que quiero que haga, a quién y cómo.

Esta clase nos sirve para tener un objeto que guarde estos 3 atributos para ser implementado por los métodos del Entity Manager.

IEntityManager

Interfaz con 4 métodos a implementar por el Entity Manager.

save()

addStatement (T Entity, String sql, Statement<T> statement)

addRangeStatement(Iterable<T> iterable, String sql, Statement<T> statement)

select(Class<T> clazz, Resultset<T> resultset)

EntityManager

Consta de 2 atributos: uno de tipo IConfiguration que establecerá la conexión con la BBDD en los métodos save y select y una lista de IRunabbles, donde se almacenarán las acciones que queremos llevar a cabo con la BBDD.

Métodos:

buildConnection: establece la conexión con la BBDD.

addStatement: crea un Objeto Runabble y lo añade a la lista mencionada anteriormente.

addRangeStatement: este método es igual que addStatement, sólo que se vale de un iterable para poder introducir varios Objetos Runabble en lugar de uno sólo.

select: realiza la conexión y ejecuta las transacciones, pudiendo finalizar en commit o rollback y finalmente cierra la conexión. Devuelve un objeto Entity de sólo lectura que se almacena en un ResultSet y finalmente limpia la Lista de Runabbles de la clase.

save: realiza la conexión y ejecuta las transacciones, emplea try catch para no tener que implementarlo en el DAO. Cómo se puede deducir, el método se emplea para escribir en la BBDD. Al igual que select al final limpia la Lista de Runnables de la clase.

IDAO

Interfaz con los métodos a implementar por la clase DAO, en este caso sólo disponemos de save y select.

PizzaDAO

Se vale del Entity Manager para realizar las inserciones y lecturas en la BBDD.