**Eliseo Iglesias González**

**Documentación de la Clase DatabaseManager**

**Descripción**

DatabaseManager es una clase que facilita la conexión y manipulación de una base de datos utilizando JDBC en Java. Implementa eventos para notificar sobre la conexión, la ejecución de consultas y posibles errores. Además, permite almacenar y cargar la configuración de la base de datos desde un archivo JSON.

**Uso**

Esta clase permite:

* Guardar la configuración de conexión a la base de datos.
* Cargar la configuración almacenada.
* Conectarse a la base de datos.
* Ejecutar consultas SELECT y actualizaciones INSERT, UPDATE, DELETE.
* Desconectarse de la base de datos.
* Notificar eventos a través de Consumer<String>.

**Atributos**

**private Connection connection;**

Referencia a la conexión activa con la base de datos.

**private Properties config;**

Objeto que almacena la configuración de la base de datos.

**Eventos (Consumer<String>)**

* onConnected: Notifica cuando se establece la conexión.
* onQueryExecuted: Notifica cuando una consulta se ejecuta con éxito.
* onError: Notifica cuando ocurre un error.

**Constructores**

**DatabaseManager(Consumer<String> onConnected, Consumer<String> onQueryExecuted, Consumer<String> onError)**

Inicializa el DatabaseManager con callbacks para eventos y carga la configuración desde el archivo JSON.

**Parámetros**

* onConnected: Callback cuando la conexión es exitosa.
* onQueryExecuted: Callback cuando una consulta se ejecuta.
* onError: Callback cuando ocurre un error.

**Métodos**

**public static void guardarConfiguracion(String url, String user, String password)**

Guarda la configuración de conexión en un archivo db\_config.json.

**Parámetros**

* url: URL de la base de datos.
* user: Usuario de la base de datos.
* password: Contraseña del usuario.

**Excepciones**

* IOException: Si ocurre un error al escribir el archivo.

**private void loadConfig()**

Carga la configuración de conexión desde el archivo db\_config.json y la almacena en config.

**Excepciones**

* Exception: Si ocurre un error al leer el archivo.

**public void connect()**

Establece una conexión con la base de datos utilizando los datos almacenados en config.

**Excepciones**

* SQLException: Si ocurre un error al conectar.

**public ResultSet executeQuery(String query)**

Ejecuta una consulta SELECT y devuelve un ResultSet con los resultados.

**Parámetros**

* query: Consulta SQL a ejecutar.

**Retorno**

* ResultSet: Resultado de la consulta.

**Excepciones**

* SQLException: Si la consulta falla.

**public void executeUpdate(String query)**

Ejecuta consultas de modificación (INSERT, UPDATE, DELETE).

**Parámetros**

* query: Consulta SQL a ejecutar.

**Excepciones**

* SQLException: Si la consulta falla.

**public void disconnect()**

Cierra la conexión con la base de datos si está abierta.

**Excepciones**

* SQLException: Si ocurre un error al cerrar la conexión.

**Notificación de eventos**

**private void notifyConnected(String message)**

Llama al callback onConnected cuando la conexión es exitosa.

**private void notifyQueryExecuted(String message)**

Llama al callback onQueryExecuted cuando se ejecuta una consulta.

**private void notifyError(String message)**

Llama al callback onError cuando ocurre un error.