



TECNOLOGICO NACIONAL DE MEXICO

INSTITUTO TECNOLÓGICO DE OAXACA

Carrera:

Ingeniería en Sistemas Computacionales

Materia:

Diseño e Implementación de software con patrones

Documentación de patrón singleton

Docente:

Espinoza Pérez Jacob

Equipo:

Ordaz Pacheco Ruudvan

Santos Manuel Julia Marlenny

Vera Acevedo Héctor Aramís

Grupo:

7SB

Fecha de entrega:

07/03/2025

Documentación del Patrón Singleton en el Sistema de Ventas

Introducción

El patrón Singleton es un patrón de diseño creacional que garantiza que una clase tenga solo una instancia y proporciona un punto de acceso global a dicha instancia. En el sistema de ventas, este patrón se implementó en la clase Conexion para administrar la conexión a la base de datos, asegurando que todas las operaciones utilicen la misma instancia de conexión.

Clase Conexion (Singleton)

La clase Conexion fue rediseñada para implementar el patrón Singleton, convirtiéndose en el núcleo de la gestión de conexiones a la base de datos. Sus características principales son:

- Constructor privado para prevenir instanciación externa
- Variable estática que almacena la única instancia
- Método getConnection() sincronizado para acceso thread-safe
- Método cerrarConexion() para liberación controlada de recursos

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class Conexion {
    // 1. Instancia única como variable estática
    private static Connection instancia;

    // 2. Constructor privado para evitar instanciación
    private Conexion() {}

    // 3. Método estático para obtener la instancia (versión thread-safe)
    public static synchronized Connection getConnection() {
        if (instancia == null) {
            try {
                // 4. Crear la conexión solo si no existe
                instancia = DriverManager.getConnection(
                    "jdbc:mysql://localhost/bd_sistema_ventas",
                    "root",
                    "1234");
                System.out.println("Conexión establecida");
            } catch (SQLException e) {
                System.err.println("Error al conectar: " + e.getMessage());
                throw new RuntimeException("Error al establecer conexión", e);
            }
        }
        return instancia;
    }

    // 5. Método para cerrar la conexión
    public static void cerrarConexion() {
        if (instancia != null) {
            try {
                instancia.close();
                instancia = null;
                System.out.println("Conexión cerrada");
            } catch (SQLException e) {
                System.err.println("Error al cerrar conexión: " + e.getMessage());
            }
        }
    }
}
```

Adaptación en Controladores

Todos los controladores del sistema (Ctrl_Usuario, Ctrl_Producto, etc.) fueron modificados para utilizar la nueva implementación Singleton:

```
public boolean guardar(Categoria objeto) {
    boolean respuesta = false;
    Connection cn = Conexion.getConnection();
    try {

        PreparedStatement consulta = cn.prepareStatement("insert into tb_categoria values(?, ?, ?)");
        consulta.setInt(1, 0);
        consulta.setString(2, objeto.getDescripcion());
        consulta.setInt(3, objeto.getEstado());

        if (consulta.executeUpdate() > 0) {
            respuesta = true;
        }

        Conexion.cerrarConexion();
    } catch (SQLException e) {
        System.out.println("Error al guardar categoria: " + e);
    }

    return respuesta;
}
```

Missing @param tag for idCategoria

(Alt-Enter shows hints)

Nota: De igual manera se implementó la conexión en todas las clases que lo necesitaban como lo son: Todos los controladores, en modelo la clase Ventas y en las vistas serían todas a excepción de FrmLogin y FrmMenu.

Ventajas del Patrón Singleton en este Código

1. **Control centralizado:** Todas las partes del sistema utilizan la misma instancia de conexión
2. **Eficiencia en recursos:** Evita la creación múltiple de conexiones a la BD
3. **Consistencia:** Garantiza que todas las operaciones trabajen con el mismo estado de conexión
4. **Facilidad de mantenimiento:** Cambios en la configuración de conexión se realizan en un solo lugar
5. **Thread-safe:** La implementación sincronizada previene problemas en entornos concurrentes

Resultados

Gestionar Productos

Administrar Productos

N°	nombre	cantidad	precio	descripcion	Iva	Categoria	estado
1	Cerveza	80	25.0	Corona	0.0	Vinos y Lico...	1
6	Yoghurt Oik...	0	19.0		0.0	Vinos y Lico...	1
30	Vino tinto	32	12.0	Licor caro	1.44	Vinos y Lico...	1
2	Leche Nutri ...	4	28.0	Leche Desc...	0.0	Lacteos	1
14	sabritas	2	56.0		0.0	Lacteos	1
15	dwed	34	23.4	adwda	0.0	Lacteos	1
18	awdawed	123	21.0	adwd	0.0	Lacteos	1
23	peokd	431	74.0	adwdaw	8.88	Lacteos	1
16	dwad	12	31.1	awdawdsd	0.0	Frutas Y ver...	1
17	dawdaw	123	32.0	adwdwa	0.0	Frutas Y ver...	1
22	dwda	234	34.0	awdawd	0.0	Frutas Y ver...	1
29	Ciruelas	43	29.0	Fruta Seca	3.48	Frutas Y ver...	1
33	avion	2	32.0	vuela	3.84	Frutas Y ver...	1
36	ave	234	11.0	cantora	1.32	Frutas Y ver...	1

Actualizar

Eliminar

Nombre: perros

Precio: 8.00

IVA: No grava iva

Cantidad: 12

Descripcion: Mascotas

Categoria: Tostadas

Gestionar Ventas

Administrar Ventas

N°	Cliente	Total Pagar	Fecha Venta	estado
1	Uri Hernandez	1254.0	2024-10-24	Activo
2	Uri Hernandez	50.16	2024-10-24	Activo
3	Uri Hernandez	50.16	2024-10-25	Activo
4	Uri Hernandez	125.4	2024-10-28	Activo
5	Uri Hernandez	125.4	2024-10-28	Activo
9	Uri Hernandez	168.0	2024-11-07	Activo
10	Uri Hernandez	194.0	2024-11-07	Activo
11	Uri Hernandez	72.0	2024-11-07	Activo
17	Uri Hernandez	76.0	2024-12-02	Activo
19	Uri Hernandez	294.0	2024-12-05	Activo
20	Uri Hernandez	116.0	2024-12-05	Activo
25	Uri Hernandez	56.0	2025-03-22	Activo
30	Uri Hernandez	18.0	2025-03-22	Activo
35	Uri Hernandez	287.0	2025-03-22	Activo

Actualizar

Total Pagar:

Fecha:

Cliente: Seleccione cliente:

Estado: Activo


Nueva Categoría

Nueva Categoría

Descripcion categoria:

Guardar

Message

 Registro Guardado

OK

Gestionar Categorías

Administrar Categorías

idCategoria	descripcion	estado
1	Vinos y Licores	1
2	Lacteos	1
3	Frutas Y verduras	1
4	Sabritas	1
5	Tostadas	1
6	Bebidas Energe...	1
7	Refrescos	1
8	Radiadores	1
9	Autos	1

Actualizar

Eliminar

Descripcion:

Conclusión

La implementación del patrón Singleton en la clase Conexion ha proporcionado una solución robusta y eficiente para la gestión de conexiones a la base de datos en el sistema de ventas. Este enfoque ha demostrado ser particularmente beneficioso para:

- Reducir el consumo de recursos
- Simplificar el código de los controladores
- Mejorar la consistencia en las operaciones con la base de datos
- Facilitar el mantenimiento futuro del sistema