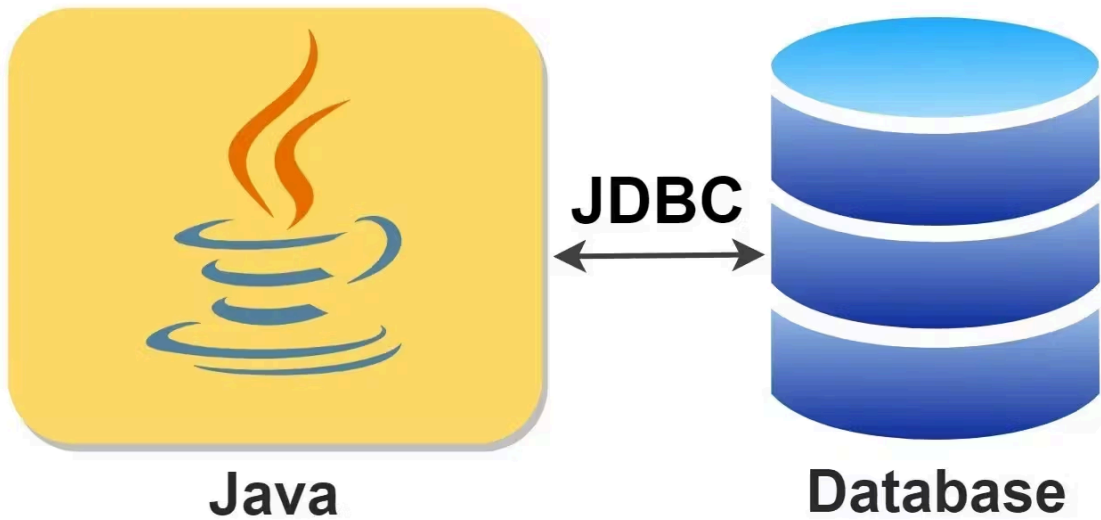


PROJECTE JDBC



DAM1 - MP03 - Programació

UF6. POO. Introducció a la persistència en les BD

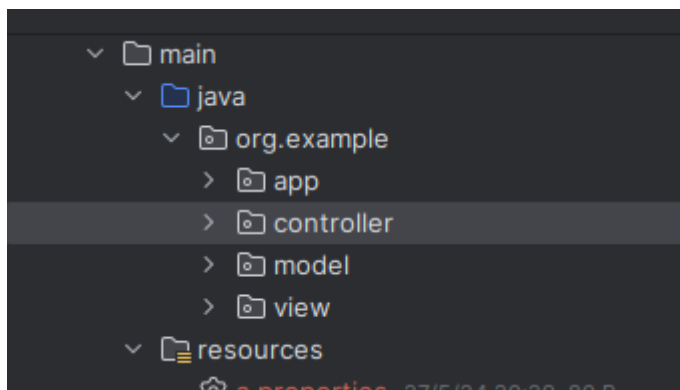
Pietro Scorza Fernandez

EXPLICACIÓ

En aquesta ho UF he fet un projecte al qual consisteix en realitzar una aplicació, la qual fa servir una base de dades d'oracle, el principal repte ha sigut fer la connexió al la el codi en java i la base oracle per ha fer-ho he implementat una interfície DAO i un fitxer properties com ara veurem.

MVC

Com podem veure la estructura del projecte segueix la dinàmica de la uf anterior, ya que separó la lògica de la part visual en molde vista i controlador.



API JDBC

Aqui defineixo la Clase ClientDAOJDBCOracleImpl la qual esta implementat DAO que es la interfície que ens permet la connexió amb la base de dades Oracle, també es veu com estic creant un objecte de la classe PropertiesFitxer, en el qual tinc les dades necessàries per a fer la connexió

```

10
11 public class ClientDAOJDBCOracleImpl implements DAO<Client> {
12
13     PropertiesFixter conexio = new PropertiesFixter();
14
15     @Override
16     public Client get(String nom) throws DAOException {
17         // Declaración de variables del método
18
19         Connection con = null;
20         PreparedStatement st = null;
21         ResultSet rs = null;
22         Client cliente = null;
23
24         // Acceso a la BD usando el API JDBC

```

CRUD

El CRUD (Create, Read, Update, Delete), A la interfície DAO he inserit aquest mètodes per a que sigui obligatori tindre's a l'hora de cridar al DAO. per a fer el CRUD de la base de dades, per a posteriorment sobrescriure i definir les seves funcions

```

1 package org.example.model.daos;
2
3
4 > import ...
5
6
7
8 public interface DAO <T>{
9
10     T get(String id) throws DAOException;
11
12     List<T> getAll() throws DAOException;
13     void clearTable() throws DAOException;
14     //CRUD
15     void update(T obj,String nomAntic) throws DAOException;
16
17     void delete(T obj) throws DAOException;
18
19     void insert(T obj) throws DAOException;
20
21     void save() throws DAOException;
22 }
23

```

```

@Override
public void save() throws DAOException {
//commit
    String sql = "commit";
    try (Connection con = DriverManager.getConnection(
        conexio.getUrl(),
        conexio.getUsername(),
        conexio.getPassword()
    ));
        PreparedStatement st = con.prepareStatement(sql);
    ) {
        st.executeUpdate();
    } catch (SQLException throwables) {
        throw new DAOException(16);
    }
}

```

```

@Override
public void update(Client obj, String nomAntic) throws DAOException {

    String sql = "UPDATE CLIENTES SET INGRES = ?, EDAD = ?, ACTIU = ?, NOM = ?, GENE = ? WHERE ID = ?";
    try (Connection con = DriverManager.getConnection(
        conexio.getUrl(),
        conexio.getUsername(),
        conexio.getPassword()
    ));
        PreparedStatement st = con.prepareStatement(sql);
    ) {
        st.setDouble(1, obj.getIngres());
        st.setInt(2, obj.getEdad());
        st.setBoolean(3, obj.getMatriculat());
        st.setString(4, obj.getNom());
        st.setString(5, obj.getGene());
        st.setString(6, nomAntic);
        st.executeUpdate();
    } catch (SQLException throwables) {
        throw new DAOException(16);
    }
}

```

```

@Override
public void delete(Client obj) throws DAOException {
    String sql = "DELETE FROM CLIENTES WHERE NOM = ?";
    try (Connection con = DriverManager.getConnection(
        conexio.getUrl(),
        conexio.getUsername(),
        conexio.getPassword()
    ));
        PreparedStatement st = con.prepareStatement(sql);
    ) {
        st.setString(1, obj.getNom());
        st.executeUpdate();
    } catch (SQLException throwables) {
        throw new DAOException(14);
    }
}

```

```

@Override
public void insert(Client obj) throws DAOException {
    String sql = "INSERT INTO CLIENTES (NOM, INGRES, EDAD, ACTIU, GENERE) VALUES (?, ?,

    try (Connection con = DriverManager.getConnection(
        conexio.getUrl(),
        conexio.getUsername(),
        conexio.getPassword()
    ));
        PreparedStatement st = con.prepareStatement(sql);
    ) {
        st.setString(1, obj.getNom());
        st.setDouble(2, obj.getIngres());
        st.setInt(3, obj.getEdad());
        st.setBoolean(4, obj.getMatriculat());
        st.setString(5, obj.getGenere());
        st.executeUpdate();
    } catch (SQLException throwables) {
        throw new DAOException(12);
    }
}

```

Interfície DAO i DAO EXCEPTION

A la imatge següent es pot veure els mètodes abstractes.

```

package org.example.model.daos;

> import ...

public interface DAO <T>{

    T get(String id) throws DAOException;

    List<T> getAll() throws DAOException;
    void crearTabla() throws DAOException;
    //CRUD
    void update(T obj,String nomAntic) throws DAOException;

    void delete(T obj) throws DAOException;

    void insert(T obj) throws DAOException;

    void save() throws DAOException;
}

```

Aquí es pot veure la DAO Exception la qual està formada per un mapa amb totes les excepcions del codi, i un número identificador, per a cada.

```

package org.example.model.exceptions;

> import ...

public class DAOException extends Exception{

    private static final Map<Integer, String> missatges = new HashMap<>();
    //num i retorna string, el map
    static {
        missatges.put(k: 0, v: "Error al connectar a la BD!!");
        missatges.put(k: 1, v: "Restricció d'integritat violada - clau primària duplicada");
        missatges.put(k: 10, v: "Camps buits");
        missatges.put(k: 11, v: "Ingres no es un numero valid");
        missatges.put(k: 12, v: "No s'ha pogut inserir el registre");
        missatges.put(k: 13, v: "ninguna fila seleccionada");
        missatges.put(k: 14, v: "No es pot repetir el nom");
        missatges.put(k: 15, v: "No s'ha pogut borrar el registre");
        missatges.put(k: 16, v: "No s'ha pogut actualitzar el registre");
        missatges.put(k: 17, v: "La edad introduida no es valida");
        missatges.put(k: 18, v: "El nom ha de començar per majúscula seguit de minúscules");
        missatges.put(k: 19, v: "L'ingres o la edad no son valids");
        missatges.put(k: 20, v: "Error amb la base de dades");
        missatges.put(k: 21, v: "Error al crear la base de dades");
        missatges.put(k: 23, v: "Selecciona un genere");
        missatges.put(k: 904, v: "Nom de columna no vàlid");
        missatges.put(k: 936, v: "Falta expressió en l'ordre SQL");
        missatges.put(k: 942, v: "La taula o la vista no existeix");
    }
}

```

Botons Etiquetes Camps

He afegit un boto per a inserir, borrar, modificar, i guardar, a més, esta el camp NOM(String), Ingres(Double), Edad(int), Actiu(boolea) i Genere(String).

nom	Ingres	Edad	Actiu	Genere
Pietro	32	32	<input checked="" type="checkbox"/>	Hombre
Pietroas	32	43	<input checked="" type="checkbox"/>	Hombre

Nom
 Edad
 Genere **SELECCIONA** ▼

Ingres
 Client? ☐

Insertar

Modificar

Borrar

Commit

Tractament de les excepcions

He tractat les excepcions utilitzant IFs o TRY junt a mètodes que he fet per a comprovar diferents coses com si els camps estan buits, quan no es compleix una comprovació, faig servir `setExcepcio(new DAOExceptio())`, aquest mètode mostrava un `JOptionPane` amb el missatge del error.


```

ide
void actionPerformed(ActionEvent e) {

    if (metodes.campsBuits(campNom.getText(), campIngres.getText(), campEdad.getText()) == true)
        setExcepcio(new DAOException(10));
    } else {
        if (generBox.getSelectedItem().toString().equals("SELECCIONA")) {
            setExcepcio(new DAOException(23));
        } else {
            if (metodes.primerMayuscula(campNom.getText()) == false) {
                setExcepcio(new DAOException(18));
            } else {
                //Si tots els camps estan plens, creem un nou Client i l'afegim a la BD (i a la t
                try {
                    int edad = 0;
                    double ingres = metodes.passarIngres(campIngres.getText());
                    try {
                        edad = metodes.passarEdad(campEdad.getText());

                        String genere = (String) generBox.getSelectedItem().toString();

                        Client al = new Client(campNom.getText(), ingres, edad, caixaClient.isSel

                        try {
                            dadesClients.insert(al);
                            campNom.requestFocus();

```

```

public void setExcepcio(DAOException excepcio) {
    DAOException valorVell=this.excepcio;
    this.excepcio = excepcio;
    canvis.firePropertyChange(PROP_EXCEPCIO, valorVell,excepcio);
}

PropertyChangeSupport canvis=new PropertyChangeSupport( sourceBean: this);

@Override
public void propertyChange(PropertyChangeEvent evt) {
    DAOException rebuda=(DAOException)evt.getNewValue();

    try {
        throw rebuda;
    } catch (DAOException e) {
        //Aquí farem ele tractament de les excepcions de l'aplicació
        switch (evt.getPropertyName()) {
            case PROP_EXCEPCIO:
                switch (rebuda.getTipo()) {
                    case 0:
                        JOptionPane.showMessageDialog( parentComponent: null, rebuda.getMessage());
                        System.exit( status: 1);
                        break;
                    case 1:
                        JOptionPane.showMessageDialog( parentComponent: null, rebuda.getMessage());
                        break;
                    case 2:
                        JOptionPane.showMessageDialog( parentComponent: null, rebuda.getMessage());
                        //this.view.getCampNom().setText(rebuda.getMissatge());
                        this.view.getCampNom().setSelectionStart(0);
                        this.view.getCampNom().setSelectionEnd(this.view.getCampNom().getText().length());
                        this.view.getCampNom().requestFocus();

                        break;
                    default:
                        JOptionPane.showMessageDialog( parentComponent: null, rebuda.getMessage()

```

Invalid VCS root mapping
The directory /Documents/It

Expresio Regular

Per al campNom he volgut que només es pugue inserir si la primera es majúscula i esta seguit de minuscules,

```
> import ...

public class Metodes extends Exception {

    @ public boolean campsBuits(String nom, String pes, String edat) {
        return nom.isEmpty() || pes.isEmpty() || edat.isEmpty();
    }

    public boolean primeraMayuscula(String input) {
        Pattern pattern = Pattern.compile(regex: "[A-Z][a-z]*$");
        Matcher matcher = pattern.matcher(input);
        if (matcher.matches()) {
            return true;
        } else {
            return false;
        }
    }
}
```

Enumeracions

Per al JComboBox he fet servir un enum el qual guarda les opcions que poden haver a la hora de escollir el gènere i després ho he ficat a les opcions del JComboBox

```
public enum Generes {
    SELECCIONA,
    Hombre,
    Mujer,
    noBinario,
    prefieroNoDecir,
    otro
}
```

```

    for (Generes genere : Generes.values()) {
        genereBox.addItem(genere);
    }
}

```

fitxer Properties

El fitxer propertis es un fitxer on podem guardar les dades de la configuració de la base de dades. com podem veure tinc tres mètodes getUrl(), getUsername(), getPassword() els quals retornen les dades que estan al fitxer.

```

package org.example.model.impls;

import java.io.InputStream;
import java.util.Properties;

public class PropertiesFitxer {
    private Properties properties = new Properties();

    public PropertiesFitxer() {
        try (InputStream input = getClass().getClassLoader().getResourceAsStream("a.properties")) {
            if (input == null) {
                System.out.println("Lo siento, no se pudo encontrar el archivo database.properties");
                return;
            }
            // Cargar el archivo properties
            properties.load(input);
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

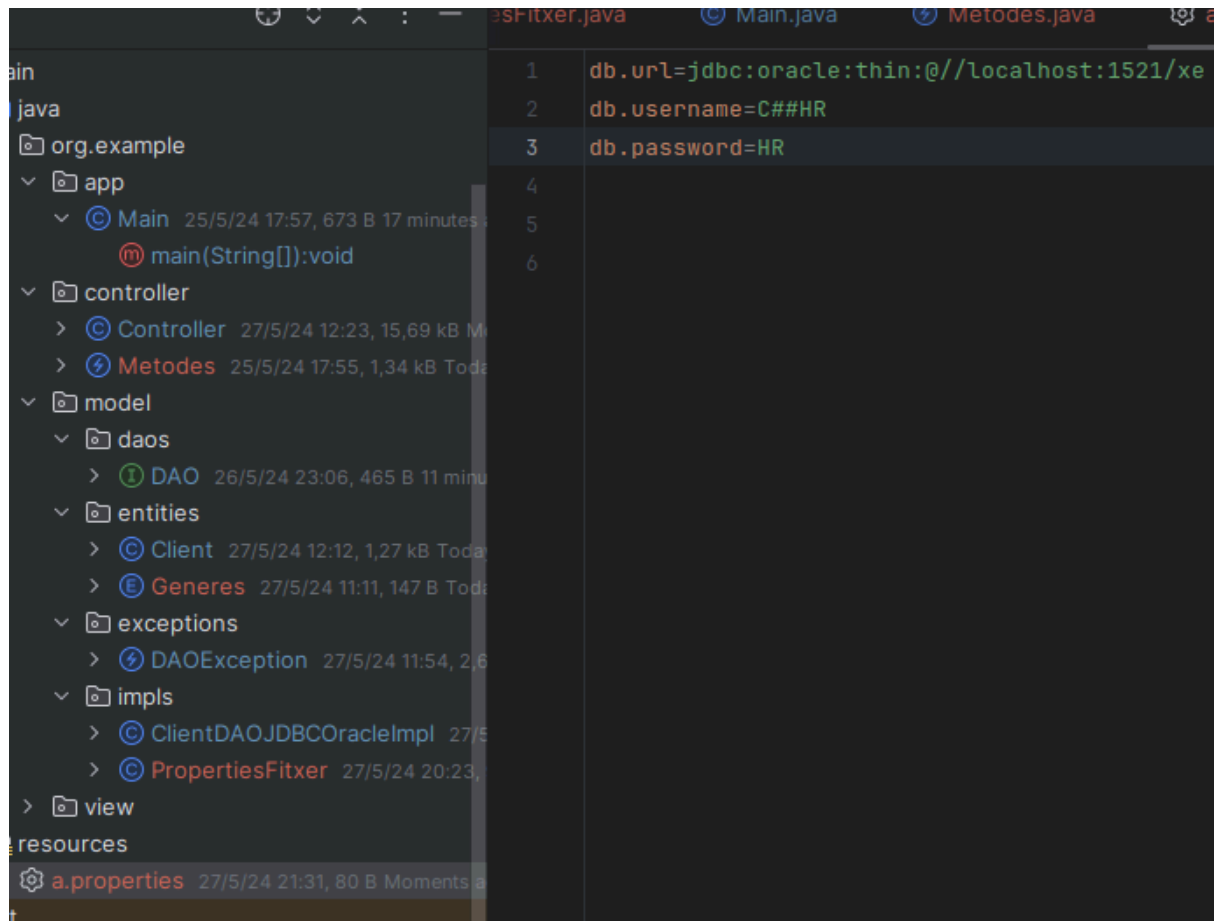
    > public String getUrl() { return properties.getProperty("db.url"); }

    > public String getUsername() { return properties.getProperty("db.username"); }

    > public String getPassword() { return properties.getProperty("db.password"); }
}

```

El fitxer en qüestió.



```
try (Connection con = DriverManager.getConnection(
    conexio.getUrl(),
    conexio.getUsername(),
    conexio.getPassword()
);
```

Prepared Statements

La sentencia sql li he de pasar al `PreparedStatement` i configurar les variables para ejecutar la sentencia

```

@Override
public void insert(Client obj) throws DAOException {
    String sql = "INSERT INTO CLIENTES (NOM, INGRES, EDAD, ACTIU, GENERE) VA

    try (Connection con = DriverManager.getConnection(
        conexio.getUrl(),
        conexio.getUsername(),
        conexio.getPassword()

    );
        PreparedStatement st = con.prepareStatement(sql);
    ) {
        st.setString(1, obj.getNom());
        st.setDouble(2, obj.getIngres());
        st.setInt(3, obj.getEdad());
        st.setBoolean(4, obj.getMatriculat());
        st.setString(5, obj.getGenere());
        st.executeUpdate();
    } catch (SQLException throwables) {
        throw new DAOException(12);
    }
}

```

```

@Override
public void update(Client obj, String nomAntic) throws DAOException {

    String sql = "UPDATE CLIENTES SET INGRES = ?, EDAD = ?, ACTIU = ?, NOM
    try (Connection con = DriverManager.getConnection(
        conexio.getUrl(),
        conexio.getUsername(),
        conexio.getPassword()

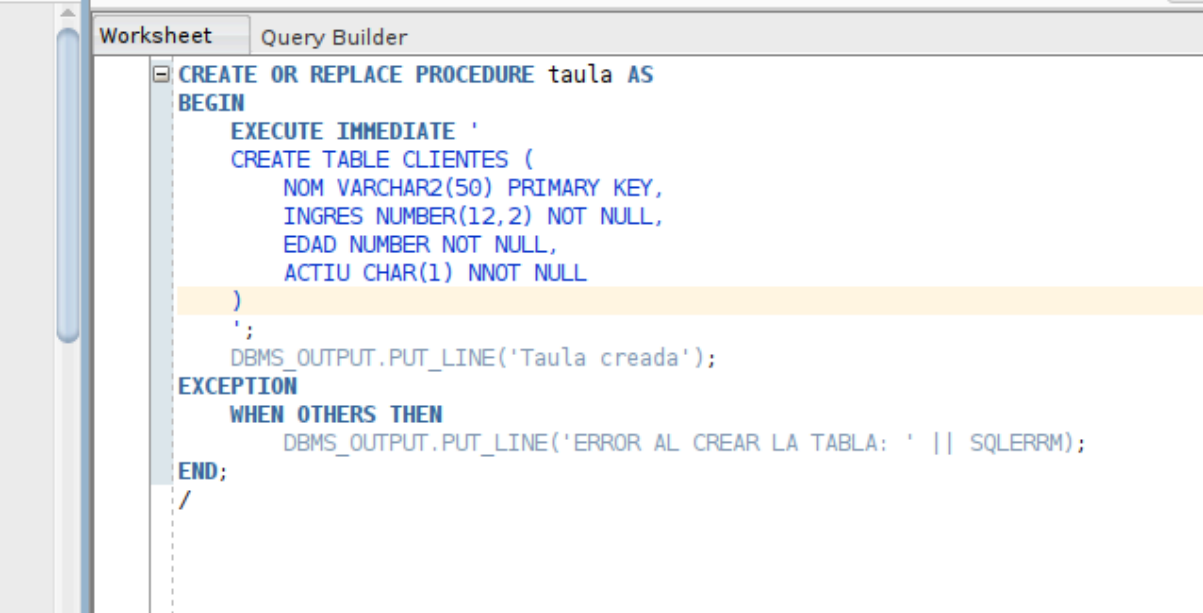
    );
        PreparedStatement st = con.prepareStatement(sql);
    ) {

        st.setDouble(1, obj.getIngres());
        st.setInt(2, obj.getEdad());
        st.setBoolean(3, obj.getMatriculat());
        st.setString(4, obj.getNom());
        st.setString(5, obj.getGenere());
        st.setString(6, nomAntic);
        st.executeUpdate();
    } catch (SQLException throwables) {
        throw new DAOException(16);
    }
}

```

procediment emmagatzemat de l'Oracle

En aquest procediment creo una taula per emmagatzemat clients, aquest la farem servir al codi per quan entres per primera vegada i no tingues taula que la cree automàticament.



```
CREATE OR REPLACE PROCEDURE taula AS
BEGIN
    EXECUTE IMMEDIATE '
    CREATE TABLE CLIENTES (
        NOM VARCHAR2(50) PRIMARY KEY,
        INGRES NUMBER(12,2) NOT NULL,
        EDAD NUMBER NOT NULL,
        ACTIU CHAR(1) NOT NULL
    )
    ';
    DBMS_OUTPUT.PUT_LINE('Taula creada');
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('ERROR AL CREAR LA TABLA: ' || SQLERRM);
END;
```

Crida a una funció emmagatzemada

Primer comprovo si la taula existeix i després crido al mètode que crida la funció.

```

        throw new DAOException(12);
    }
}

@Override
public void crearTabla() throws DAOException {
    String sql = "BEGIN TAULA; END;";
    try (Connection con = DriverManager.getConnection(
        conexio.getUrl(),
        conexio.getUsername(),
        conexio.getPassword()
    );
        PreparedStatement st = con.prepareStatement(sql);
    ) {
        st.executeUpdate();
    } catch (SQLException throwables) {
        throw new DAOException(10);
    }
}
}

```

Metodes

Hi ha diferents mètodes que he utilitzat sobretot per a la comprovació de camps, aquestos com havia de fer la mateixa comprovació mes de una vegada he guardat els mètodes a una classe.

```
import ...

public class Metodes extends Exception {

    public boolean campsBuits(String nom, String pes, String edat) {
        return nom.isEmpty() || pes.isEmpty() || edat.isEmpty();
    }

    public boolean primeraMayuscula(String input) {
        Pattern pattern = Pattern.compile(regex: "[A-Z][a-z]*$");
        Matcher matcher = pattern.matcher(input);
        if (matcher.matches()) {
            return true;
        } else {
            return false;
        }
    }

    public Double passarIngres(String ingres) throws DAOException {
        NumberFormat num = NumberFormat.getNumberInstance(Locale.getDefault()); //C
        try {
            double ingresDouble = num.parse(ingres).doubleValue();
            return ingresDouble;
        }
    }
}
```