



18-5-2021

DOCUMENTACIÓN JAVAFX



Antonio Franco, José Antonio Villarreal, Salvador
Vázquez, Fernando Borrego
GRUPO 5 – 1º CFGS DAM

Contenido

1.- INTRODUCCIÓN:	2
2.- CONTROLADORES:	2
3.- CLASES:.....	3
3.1.- DONACION.JAVA:	3
3.2.- DONANTE.JAVA:	4
3.3.- IOBASEDATOS.JAVA:.....	5
3.4.- IODONANTESDAT.JAVA:	6
3.5.- LEERXML.JAVA:.....	7
4.- VISTAS:	8
5.- LIBRERIAS:	9
6.- JAVADOC:	10
7.- REFACTORIZACIONES:	10
8.- DIAGRAMAS UML:.....	11

1.- INTRODUCCIÓN:

Este proyecto, tiene como objetivo implementar una aplicación que contemple los aspectos que concurren en la gestión de un banco de sangre.

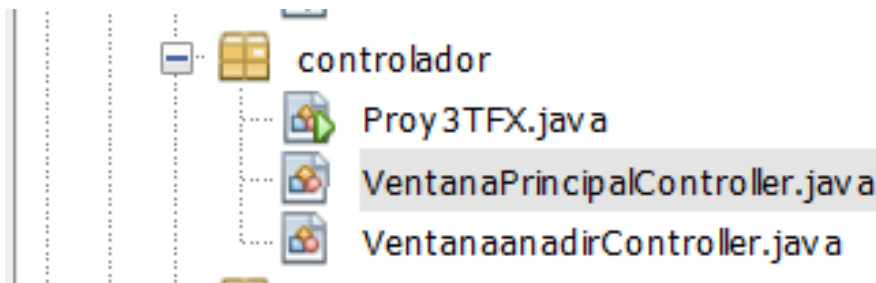
El archivo que se nos facilita para la realización del proyecto es "**sanitarios.xml**", que contiene tres sanitarios que pueden hacer la extracción de sangre a los donantes.

También hemos añadido un **logo**, que aparecerá como fondo en la ventana de nuestra aplicación.

2.- CONTROLADORES:

Como controladores tenemos una **VentanaPrincipalControler**, para añadir y eliminar personas y refrescar la tabla. Con ella, iniciamos la tabla **donante** y **donaciones**.

Con **VentanaAñadirControler**, se pueden crear personas, comprobar si ya existen, modificar objetos tales como los donantes y cerrar la ventana. Todo ello va acompañado de un Main **Proy3TFX**.



3.- CLASES:

Dentro de las clases, tenemos a **Donación**, **Donante**, **IObaseDatos**, **IODonantesDat** y **LeerXML**.

3.1.- DONACION.JAVA:

En Donación, guardamos datos como: el DNI, el código del sanitario, la incidencia, el nombre del sanitario y del centro, el teléfono del sanitario y la fecha de la donación.

```
public class Donacion {

    private String DNI;
    private String codSan;
    private String cant;
    private String inc;
    private String nomSan;
    private String nomCen;
    private String telSan;
    private String fecDon;

    public Donacion(String DNI, String codSan, String cant, String inc, String nomCen, String nomSan, String telSan, String fecDon) {
        this.DNI = DNI;
        this.codSan = codSan;
        this.cant = cant;
        this.inc = inc;
        this.nomCen = nomCen;
        this.nomSan = nomSan;
        this.telSan = telSan;
        this.fecDon = fecDon;
    }

    public String getDNI() {
        return DNI;
    }

    public void setDNI(String DNI) {
        this.DNI = DNI;
    }
}
```

3.2.- DONANTE.JAVA:

En Donante, podemos encontrar: DNI, nombre del donante, dirección, código postal del donante, su localidad, así como su fecha de nacimiento, su correo, teléfono, grupo sanguíneo, factorRH, y su PK.

```
public class Donante {  
  
    private String DNI;  
    private String Nombre;  
    private String Direccion;  
    private String CodPosatal;  
    private String Localidad;  
    private LocalDate FechaNac;  
    private String Correo;  
    private String Telefono;  
    private String GrupoSang;  
    private String FactorRH;  
    private int PK;  
  
    public Donante() {  
    }  
  
    public Donante(String DNI, String Nombre, String Direccion, String CodPosatal, String Localidad, LocalDate FechaNac, String Correo,  
        this.DNI = DNI;  
        this.Nombre = Nombre;  
        this.Direccion = Direccion;  
        this.CodPosatal = CodPosatal;  
        this.Localidad = Localidad;  
        this.FechaNac = FechaNac;  
        this.Corrreo = Correo;  
        this.Telefono = Telefono;  
        this.GrupoSang = GrupoSang;  
        this.FactorRH = FactorRH;  
    }  
  
    public String getDNI() {  
        return DNI;  
    }  
}
```

```
public String getCodSan() {  
    return codSan;  
}  
  
public void setCodSan(String codSan) {  
    this.codSan = codSan;  
}  
  
public String getCant() {  
    return cant;  
}  
  
public void setCant(String cant) {  
    this.cant = cant;  
}  
  
public String getInc() {  
    return inc;  
}  
  
public void setInc(String inc) {  
    this.inc = inc;  
}  
  
public String getNomSan() {  
    return nomSan;  
}  
  
public void setNomSan(String nomSan) {  
    this.nomSan = nomSan;  
}  
}
```

3.3.- IOBASEDATOS.JAVA:

IObaseDatos establece la conexión con la base de datos realizada con anterioridad gracias a Donación y Donante. Establece la conexión a la BBDD mediante el método `getConnection()`, este método contiene la cadena de conexión **urlCon**, también le pasamos el usuario y la contraseña de nuestra BBDD, generando una conexión directa con ella. En esta clase también contamos con los métodos de `executeQuery()`, los cuales nos permite ejecutar la consulta que recibe como argumento. Otro segundo método el cual es `executeUpdate`, se usa para las 3 opciones más habituales, las cuales son "insert", "update" y "delete".

```
public void actualizaRegistros(String actualiza) throws SQLException{
    try {
        Class.forName("org.mariadb.jdbc.Driver");
        String urlCon = "jdbc:mariadb://localhost:3306/Proy3TE5";
        Connection conexBd = DriverManager.getConnection(urlCon, "root", "root");
        Statement encapsulaCons = conexBd.createStatement();

        int filActualizadas = encapsulaCons.executeUpdate(actualiza);
        if(filActualizadas > 0){System.out.print("Hola");}

        encapsulaCons.close();
        conexBd.close();
    } catch (ClassNotFoundException | SQLException cnfe) {
        System.out.println(cnfe.getMessage());
    }
}

public ResultSet introduceRegistros(String consulta) {
    ResultSet resulCons = null;
    try {
        Class.forName("org.mariadb.jdbc.Driver");
        String urlCon = "jdbc:mariadb://localhost:3306/Proy3TE5";
        Connection conexBd = DriverManager.getConnection(urlCon, "root", "root");
        Statement encapsulaCons = conexBd.createStatement();

        // "INSERT INTO DONANTES(DNI, Nombre, Direccion, CodPostal, Localidad, FechaNac, Correo, Telefono, GrupoSang, FactorRH) VAI
        resulCons = encapsulaCons.executeQuery(consulta);

        encapsulaCons.close();
        conexBd.close();
    } catch (ClassNotFoundException | SQLException cnfe) {
```

3.4.- IODONANTESDAT.JAVA:

IODonantesDat, es la clase que controla el "Donantes.dat" para leer e introducir datos. Esta clase, contiene 2 métodos, el primero de ellos introDatos() al que le pasamos como argumento dniDonante, codigoSanitario, fechaDonacion, cantidadMl e Incidencia.

Como indica el nombre de este método lo usamos para introducir los datos en DonantesDat.

```
public void introDatos(String dniDonante, int codigoSanitario, String fechaDonacionWapilla, float cantidadMl, boolean incidencia)
{
    try {
        ficheroSalida = new FileOutputStream("src/Donantes.dat", true);
        datosSalida = new DataOutputStream(ficheroSalida);

        datosSalida.writeUTF(dniDonante);
        datosSalida.writeInt(codigoSanitario);
        datosSalida.writeUTF(fechaDonacionWapilla);
        datosSalida.writeFloat(cantidadMl);
        datosSalida.writeBoolean(incidencia);

        System.out.println("\t\nDatos incorporados al fichero\n");
    } catch (IOException fnfe) {
        System.out.println(fnfe.getMessage());
    } finally {
        if (ficheroSalida != null) {
            ficheroSalida.close();
        }
        if (datosSalida != null) {
            datosSalida.close();
        }
    }
}
```

El segundo método llamado leerDatosDat(), le pasamos como argumento, dniDeseado y como indica el nombre sirve para leer los datos del fichero.

```
public Donacion leerDatosDat(String dniDeseado) throws FileNotFoundException, IOException {
    boolean encontrado = false;

    RandomAccessFile fichero = new RandomAccessFile("src/Donantes.dat", "r");
    fichero.seek(0);
    int contador = 0;
    Donacion Don = null;

    while (true) {
        String dni = fichero.readUTF();
        String codigoSanitarioDat = Integer.toString(fichero.readInt());
        String fechaDonacion = fichero.readUTF();
        String cantidadMl = Float.toString(fichero.readFloat());
        String incidencia = Boolean.toString(fichero.readBoolean());

        if (dni.equals(dniDeseado)) {
            System.out.print("Encontrado");

            Don = new Donacion(dni, codigoSanitarioDat, fechaDonacion, cantidadMl, X.leerDatosXML(codigoSanitarioDat), "laa", "aaaa");
            return Don;
        }
    }
}
```

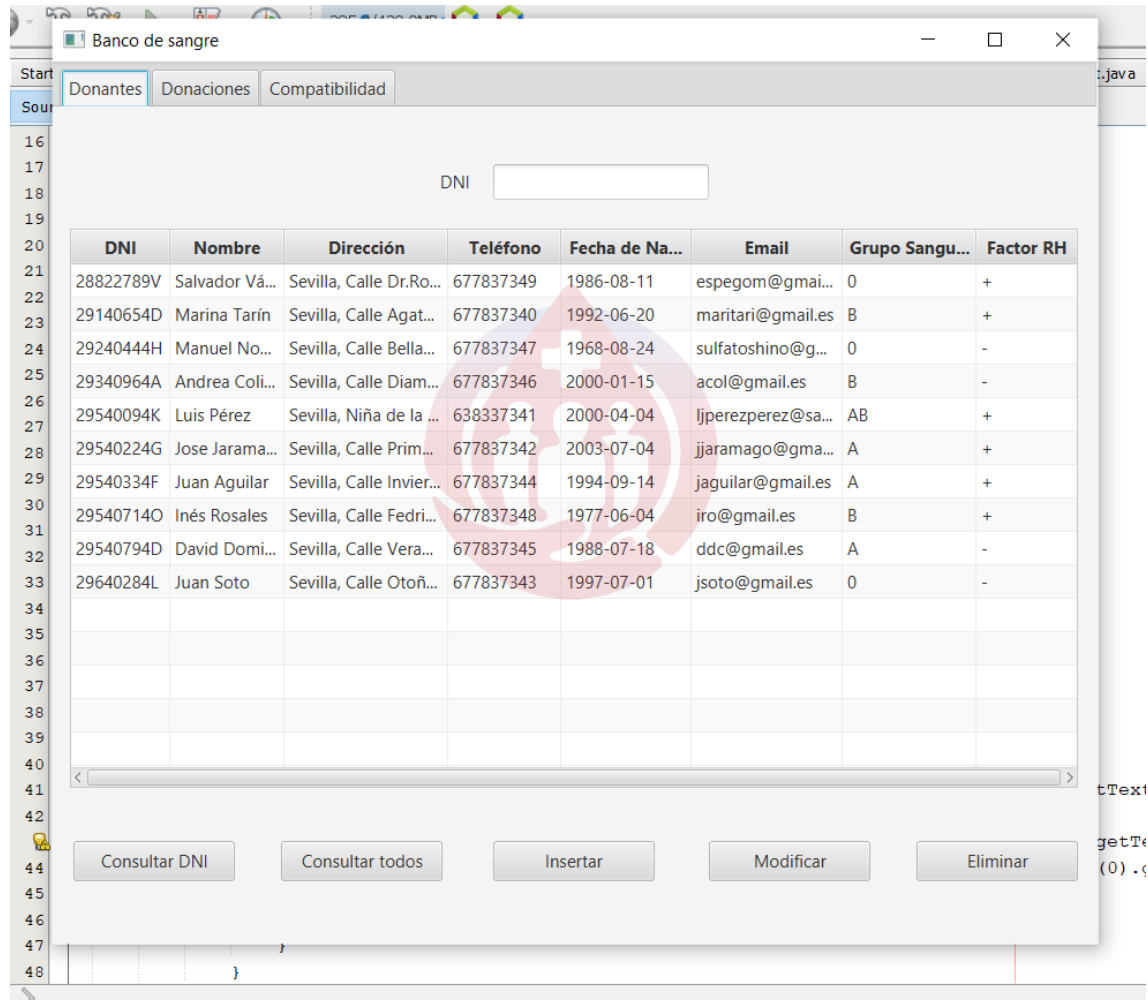
3.5.- LEERXML.JAVA:

En LeerXML, es una clase que contiene un método llamado leerDatosXML(), que le pasamos como argumento CodSan, abre el documento XML, recorre los nodos y saca los datos a pantalla

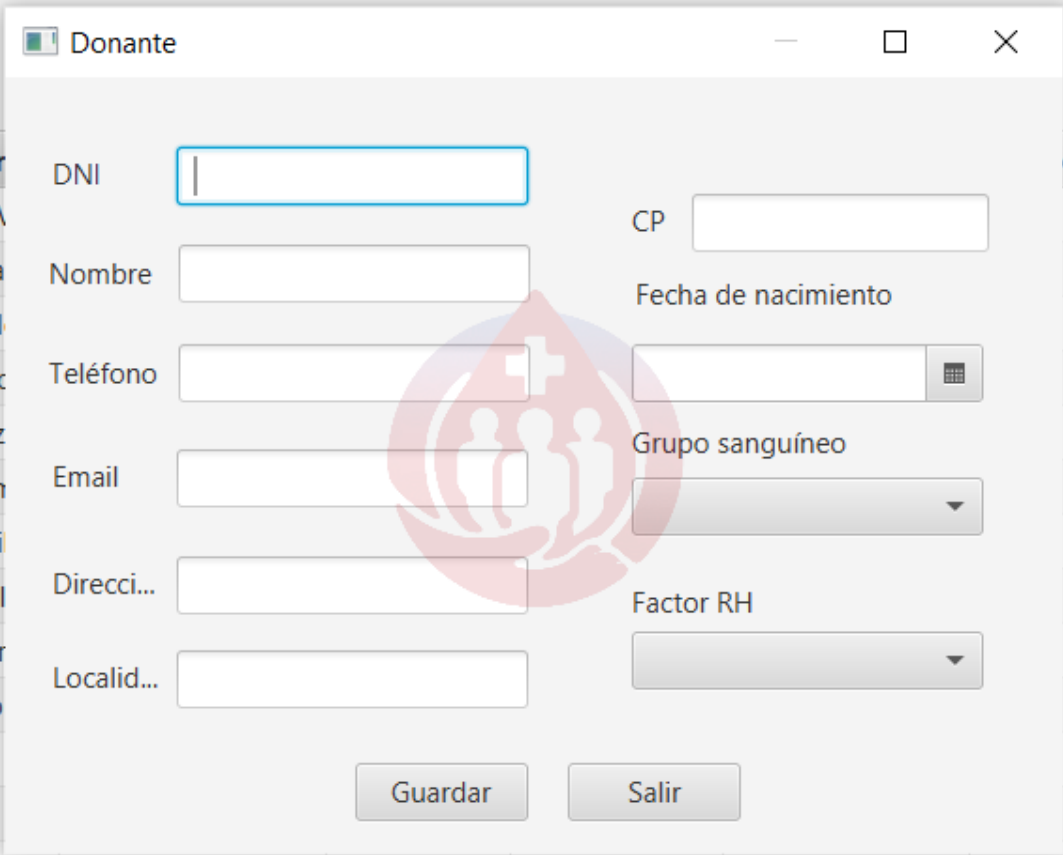
```
public class LeerXML {  
    protected String leerDatosXML(String codSan) {  
        File fichXML = new File("sanitarios.xml");  
        String retorno = null;  
  
        try {  
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();  
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();  
            Document doc = dBuilder.parse(fichXML);  
            doc.getDocumentElement().normalize();  
  
            doc.getDocumentElement();  
  
            NodeList sanitarios = doc.getElementsByTagName("sanitario");  
  
            for (int cont = 0; cont < sanitarios.getLength(); cont++) {  
  
                Node nodo = sanitarios.item(cont);  
  
                if (nodo.getNodeType() == Node.ELEMENT_NODE) {  
  
                    Element element = (Element) nodo;  
  
                    if (codSan.equals(element.getElementsByTagName("codsan").item(0).getTextContent())) {  
  
                        String correo = element.getElementsByTagName("correo").item(0).getTextContent();  
                        String telefono = element.getElementsByTagName("telefono").item(0).getTextContent();  
                        retorno = telefono;  
                    }  
                }  
            }  
        }  
    }  
}
```


4.- VISTAS:

Dentro de las vistas, tenemos dos, por una parte en VentanaPrincipal tenemos 3 paneles, uno de la tabla donantes, otro de donaciones y finalmente otro de compatibilidad.



En la vista de Ventanaanadir, aparecerá una vista que pide los datos del Donante en general, contiene dos botones, uno para salir de la vista y otro para guardar el donante que se desea insertar.



Donante

DNI

CP

Nombre

Fecha de nacimiento

Teléfono

Grupo sanguíneo

Email

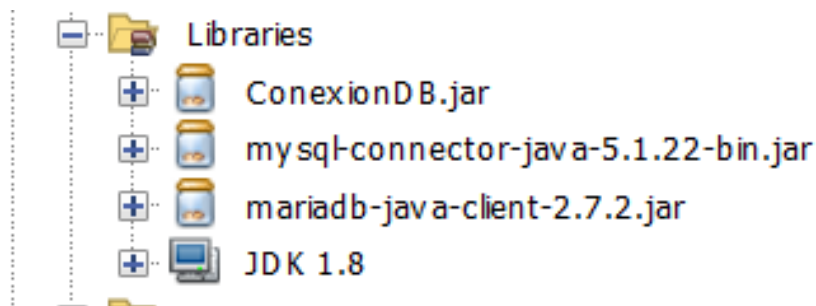
Direcci...

Factor RH

Localid...

5.- LIBRERIAS:

Finalmente, en las librerías hemos añadido los siguientes archivos: conexionDB.jar, mysql-connector-java y JDK 1.8.



6.- JAVADOC:

Ruta de Javadoc:

<file:///C:/Users/josea/Desktop/Proy3TFX/dist/javadoc/index.html?overview-summary.html>

7.- REFACTORIZACIONES:

- Eliminación de 'imports' que no se usan durante el código.
- Eliminación de espacios para hacer más agradable el código a la vista.
- Cambio de nombre de algunas clases y métodos, para hacerlas más intuitivas.

8.- DIAGRAMAS UML:

