PROYECTO DE TRANSFERENCIAS Y PAGOS

Este proyecto está creado para transferir datos, o bien convirtiendo y adaptando archivos de Excel (CSV) en XML o bien de base de datos a XML o de base de datos a CSV. Se creó por la norma **ISO20022**, pero finalmente se acabó adaptando también al **Cuaderno 34** y al **Cuaderno 34_14**.

Cuaderno 34_14

La función principal es la de crear un CSV con los datos de una tabla llamada c34_14 en una base de datos MySQL.

Esta función corresponde al método generarCSV_c34_14(List<Pago> pagos, String ruta) de la clase MensajeDAO.java. Se le pasa una lista de pagos (correspondiente a la clase Pago.java y a la tabla c34_14 de la base de datos) y la ruta de destino.

El archivo se pasa por la clase FrmCreadorXML.java en el btnGenerarActionPerformed.

Con este método *getPagos* se obtienen todos los pagos de la tabla *c34_14* de la base de datos.

```
public List<Pago> getPagos() throws SQLException {
    List<Pago> pagos = new ArrayList<>();
    String query = "SELECT * FROM c34_14";
    try (PreparedStatement statement = cn.prepareStatement(query)) {
        ResultSet resultSet = statement.executeQuery();
        while (resultSet.next()) {
            Pago pago = new Pago();
            pago.setPagos(resultSet.getString("pagos"));
            pago.setNormal(resultSet.getString("normal"));
            pago.setCifOrdenante(resultSet.getString("cifOrdenante"));
            pago.setNumero(resultSet.getString("numero"));
            pago.setFechaOperacion(resultSet.getString("fechaOperacion"));
            pago.setIbanOrdenante(resultSet.getString("ibanOrdenante"));
            pago.setConcepto(resultSet.getString("concepto"));
            pago.setNombreOrdenante(resultSet.getString("nombreOrdenante"));
            pago.setIbanBeneficiario(resultSet.getString("ibanBeneficiario"));
            pago.setImporte(resultSet.getDouble("importe"));
            pago.setNombreBeneficiario(resultSet.getString("nombreBeneficiario"));
            pago.setGastos(resultSet.getString("gastos"));

            pagos.add(pago);
        }
    }
    return pagos;
}
```

Desde el FrmCreadorXML.java:

```
List<Pago> pagos = mensaje.getPagos();
```

Y se le pasa como argumento al método que genera el CSV, que es el *generarCSV_c34_14*:

```
public int generarCSV_c34_14(List<Pago> pagos, String ruta) {
    try {
        Writer writer = new OutputStreamWriter(new FileOutputStream(ruta), StandardCharsets.UTF_8);
    // FileWriter writer = new FileWriter(ruta);

        for (Pago pago : pagos) {
            writer.write(pago.getPagos() + ";");
            writer.write(pago.getNormal() + ";");
            writer.write(pago.getCifOrdenante() + ";");
            writer.write(pago.getNumero() + ";");
            writer.write(";");
            writer.write(pago.getFechaOperacion() + ";");
            writer.write(pago.getIbanOrdenante() + ";");
            writer.write(pago.getConcepto() + ";");
            writer.write(pago.getNombreOrdenante() + ";");
            writer.write(";");
            writer.write(pago.getIbanBeneficiario() + ";");
            writer.write(pago.getImporte() + ";");
            writer.write(pago.getNombreBeneficiario() + ";");
            writer.write(pago.getGastos() + ";");
            writer.write(";");
            writer.write(pago.getConcepto() + "\n");

        }

        writer.close();
        return 1;
    } catch (IOException e) {
        e.printStackTrace();
        return 0;
    }
}
```

El archivo se genera en la ruta especificada en la interfaz:

El archivo se genera con el nombre mensaje seguido de la fecha y hora de creación: *mensaje_ddMMyyyy_HHmmss.csv*

Aquí se especifica:

```
SimpleDateFormat sdf = new SimpleDateFormat("ddMMyyyy_HHmmss");
SimpleDateFormat sdf2 = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
String timestamp = sdf.format(new Date());
String timestamp2 = sdf2.format(new Date());
String nombreArchivo = "mensaje_" + timestamp + ".xml";
String nombreArchivoCSV = "mensaje_" + timestamp + ".csv";
String nombreLog = "mensaje_" + timestamp + ".log";
String ruta2 = rutaDestino + "\\" + nombreArchivoCSV;
rutaDestino += "\\" + nombreArchivo;
```

Y se crea desde el FrmCreador.java en estas líneas:

```
List<Pago> pagos = mensaje.getPagos();
int resultado = mensaje.generarCSV_c34_14(pagos, ruta2);
if(resultado == 1) {
    JOptionPane.showMessageDialog(null, "Excel creado con éxito en " + ruta2, "CSV creado", JOptionPane.INFORMATION_MESSAGE);
    crearLog(nombreArchivoCSV, rutaDestino, timestamp2);
} else {
    JOptionPane.showMessageDialog(null, "Error creando excel", "Error", JOptionPane.ERROR_MESSAGE);
}
```

Formato ISO 20022

Otra de las funciones es la del formato ISO 20022, el más nuevo. Dentro de este apartado hay dos funciones: Convertir un Excel (CSV) a un XML y convertir datos de una tabla de base de datos a XML.

De Excel a XML

En este apartado se leen los registros del Excel:

```
private static List<Cuerpo> leerCuerpos(String origen) {
    List<Cuerpo> cuerpos = null;
    try (BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStream(origen), "UTF-8"))) {
    //  (BufferedReader br = new BufferedReader(new FileReader(origen)))
        cuerpos = br.lines().skip(1) // saltar la cabecera del CSV
                .map(linea -> {
                    String[] datos = linea.split(";");
                    Cuerpo cuerpo = new Cuerpo();
                    // Asignar datos del CSV a los campos de Cuerpo
                    // Suponiendo que los datos están en el mismo orden que los campos en la clase Cuerpo.java
                    cuerpo.setIdentificadorPago(datos[0]);
                    cuerpo.setMedioPago(datos[1]);
                    cuerpo.setIndicadorApunteCuenta(datos[2]);
                    cuerpo.setInformacionTipoPago(datos[3]);
                    cuerpo.setFecha(datos[4]);
                    cuerpo.setNombreOrdenante(datos[5]);
                    cuerpo.setNifOrdenante(datos[6]);
                    cuerpo.setCuentaOrdenante(datos[7]);
                    cuerpo.setNombreBeneficiario(datos[8]);
                    cuerpo.setNifBeneficiario(datos[9]);
                    cuerpo.setCuentaBeneficiario(datos[10]);
                    String controlSumaStr = datos[11].replace(',', '.'); // Reemplazar comas por puntos
                    cuerpo.setControlSuma(Double.parseDouble(controlSumaStr));
                    return cuerpo;
                }).collect(Collectors.toList());
    } catch (IOException e) {
        e.printStackTrace();
    }
    return cuerpos;
}
```

Este método calcula el controlSuma de la cabecera (sumando todos los controlSuma de los pagos):

```
private static Double calcularControlSuma(List<Cuerpo> cuerpos) {
    return cuerpos.stream().mapToDouble(Cuerpo::getControlSuma).sum();
}
```

Con este método se crea la cabecera:

```
private static Cabecera generarCabecera(String origen) {
    Cabecera cabecera = new Cabecera();
    cabecera.setFecha(new SimpleDateFormat("yyyy-MM-dd").format(new Date()));
    cabecera.setHora(new SimpleDateFormat("HH:mm:ss").format(new Date()));
    List<Cuerpo> cuerpos = leerCuerpos(origen);
    cabecera.setNumeroOperaciones(cuerpos.size());
    cabecera.setControlSuma(calcularControlSuma(cuerpos));
    return cabecera;
}
```

Y con este método se crea la estructura y se rellenan los campos del XML:

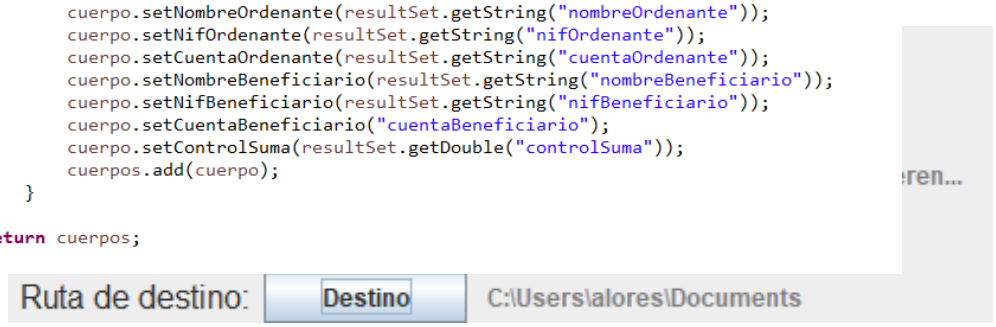
```
private static void generarXML(Cabecera cabecera, List<Cuerpo> cuerpos, String destino) {
    try (Writer writer = new BufferedWriter(new OutputStreamWriter(new FileOutputStream(destino), "UTF-8"))) {
    //  (Writer writer = new FileWriter(destino))
        writer.write("<mensaje>\n");

        // CABECERA
        writer.write("<cabecera>\n");
        writer.write("\t<fecha>" + cabecera.getFecha() + "</fecha>\n");
        writer.write("\t<hora>" + cabecera.getHora() + "</hora>\n");
        writer.write("\t<numeroOperaciones>" + cabecera.getNumeroOperaciones() + "</numeroOperaciones>\n");
        String controlSumaFormateado = String.format("%.2f", cabecera.getControlSuma());
        writer.write("\t<controlSuma>" + controlSumaFormateado + "</controlSuma>\n");
        writer.write("</cabecera>\n");

        // CUERPO
        writer.write("<cuerpo>\n");
        for (Cuerpo cuerpo : cuerpos) {
            writer.write("\t<pago>\n");
            writer.write("\t\t<identificadorPago>" + cuerpo.getIdentificadorPago() + "</identificadorPago>\n");
            writer.write("\t\t\t<medioPago>" + cuerpo.getMedioPago() + "</medioPago>\n");
            writer.write("\t\t\t<indicadorApunteCuenta>" + cuerpo.getIndicadorApunteCuenta() + "</indicadorApunteCuenta>\n");
            writer.write("\t\t\t<informacionTipoPago>" + cuerpo.getInformacionTipoPago() + "</informacionTipoPago>\n");
            writer.write("\t\t\t<fecha>" + cuerpo.getFecha() + "</fecha>\n");
            writer.write("\t\t\t<nombreOrdenante>" + cuerpo.getNombreOrdenante() + "</nombreOrdenante>\n");
            writer.write("\t\t\t<nifOrdenante>" + cuerpo.getNifOrdenante() + "</nifOrdenante>\n");
            writer.write("\t\t\t<cuentaOrdenante>" + cuerpo.getCuentaOrdenante() + "</cuentaOrdenante>\n");
            writer.write("\t\t\t<nombreBeneficiario>" + cuerpo.getNombreBeneficiario() + "</nombreBeneficiario>\n");
            writer.write("\t\t\t<nifBeneficiario>" + cuerpo.getNifBeneficiario() + "</nifBeneficiario>\n");
            writer.write("\t\t\t<cuentaBeneficiario>" + cuerpo.getCuentaBeneficiario() + "</cuentaBeneficiario>\n");
            writer.write("\t\t<controlSuma>" + cuerpo.getControlSuma() + "</controlSuma>\n");
            writer.write("\t</pago>\n");
        }
        writer.write("</cuerpo>");

        writer.write("</mensaje>");
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Finalmente, para crear el archivo XML usamos este método, en el cual se pasan por parámetro el archivo CSV de origen y la ruta de destino:

```
public static void convertirCSVaXML(String origen, String destino) {
    Cabecera cabecera = generarCabecera(origen);
    List<Cuerpo> cuerpos = leerCuerpos(origen);
    generarXML(cabecera, cuerpos, destino);
    System.out.println("Generado");
}
```

```java
public List<Cuerpo> getCuerposByIdMensaje(int idMensaje) throws SQLException {
    List<Cuerpo> cuerpos = new ArrayList<>();
    String query = "SELECT * FROM cuerpo WHERE idMensaje = ?";
    try (PreparedStatement statement = cn.prepareStatement(query)) {
        statement.setInt(1, idMensaje);
        ResultSet resultSet = statement.executeQuery();
        while (resultSet.next()) {
            Cuerpo cuerpo = new Cuerpo();
            cuerpo.setIdentificadorPago(resultSet.getString("identificadorPago"));
            cuerpo.setIdMensaje(resultSet.getInt("idMensaje"));
            cuerpo.setMedioPago(resultSet.getString("medioPago"));
            cuerpo.setIndicadorApunteCuenta(resultSet.getString("indicadorApunteCuenta"));
            cuerpo.setInformacionTipoPago(resultSet.getString("informacionTipoPago"));
            cuerpo.setFecha(resultSet.getString("fecha"));
            cuerpo.setNombreOrdenante(resultSet.getString("nombreOrdenante"));
            cuerpo.setNifOrdenante(resultSet.getString("nifOrdenante"));
            cuerpo.setCuentaOrdenante(resultSet.getString("cuentaOrdenante"));
            cuerpo.setNombreBeneficiario(resultSet.getString("nombreBeneficiario"));
            cuerpo.setNifBeneficiario(resultSet.getString("nifBeneficiario"));
            cuerpo.setCuentaBeneficiario("cuentaBeneficiario");
            cuerpo.setControlSuma(resultSet.getDouble("controlSuma"));
            cuerpos.add(cuerpo);
        }
    }
    return cuerpos;
}
```

Así es como se ve desde la interfaz cuando se usa esta opción:



De Base de Datos a XML

El primer método lo que hace es obtener la cabecera por el id que le pasamos por parámetro:

El siguiente método obtiene los pagos por el ID del mensaje:

```java
public Cabecera getCabeceraById(int idMensaje) throws SQLException {
    Cabecera cabecera = null;
    String query = "SELECT * FROM cabecera WHERE idMensaje = ?";
    try (PreparedStatement statement = cn.prepareStatement(query)) {
        statement.setInt(1, idMensaje);
        ResultSet resultSet = statement.executeQuery();
        if (resultSet.next()) {
            cabecera = new Cabecera();
            cabecera.setIdMensaje(resultSet.getInt("idMensaje"));
            cabecera.setFecha(resultSet.getString("fecha"));
            cabecera.setHora(resultSet.getString("hora"));
            cabecera.setNumeroOperaciones(resultSet.getInt("numeroOperaciones"));
            cabecera.setControlSuma(resultSet.getDouble("controlSuma"));
        }
    }
    return cabecera;
}
```

Y, por último, el método que genera el XML a partir de estos datos obtenidos de la base de datos MySQL:

```
public int generarXML_iso20022(Cabecera cabecera, List<Cuerpo> cuerpos, String ruta) throws IOException {
    try {
        FileWriter writer = new FileWriter(ruta);
        writer.write("<mensaje>\n");
        writer.write("\t<cabecera>\n");

        LocalDateTime fechaHoraActual = LocalDateTime.now();

        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("HH:mm");
        String horaFormateada = fechaHoraActual.format(formatter);
        writer.write("\t\t<idMensaje>" + cabecera.getIdMensaje() + "</idMensaje>\n");
        writer.write("\t\t<fecha>" + fechaHoraActual.toLocalDate() + "</fecha>\n");
        writer.write("\t\t<hora>" + horaFormateada + "</hora>\n");

        int numeroOperaciones = cuerpos.size();
        Double sumaControlSuma = 0.0;
        for (Cuerpo cuerpo : cuerpos) {
            sumaControlSuma += cuerpo.getControlSuma();
        }
        String sumaControlSumaFormateada = String.format("%.2f", sumaControlSuma);

        writer.write("\t\t<numeroOperaciones>" + numeroOperaciones + "</numeroOperaciones>\n");
        writer.write("\t\t<controlSuma>" + sumaControlSumaFormateada + "</controlSuma>\n");
        writer.write("\t</cabecera>\n");

        writer.write("\t<cuerpo>\n");
        for (Cuerpo cuerpo : cuerpos) {
            writer.write("\t\t<pago>\n");
            writer.write("\t\t\t<identificadorPago>" + cuerpo.getIdentificadorPago() + "</identificadorPago>\n");
            writer.write("\t\t\t<medioPago>" + cuerpo.getMedioPago() + "</medioPago>\n");
            writer.write("\t\t\t<indicadorApunteCuenta>" + cuerpo.getIndicadorApunteCuenta() + "</indicadorApunteCuenta>\n");
            writer.write("\t\t\t<informacionTipoPago>" + cuerpo.getInformacionTipoPago() + "</informacionTipoPago>\n");
            writer.write("\t\t\t<fecha>" + cuerpo.getFecha() + "</fecha>\n");
            writer.write("\t\t\t<nombreOrdenante>" + cuerpo.getNombreOrdenante() + "</nombreOrdenante>\n");
            writer.write("\t\t\t<nifOrdenante>" + cuerpo.getNifOrdenante() + "</nifOrdenante>\n");
            writer.write("\t\t\t<cuentaOrdenante>" + cuerpo.getCuentaOrdenante() + "</cuentaOrdenante>\n");
            writer.write("\t\t\t<nombreBeneficiario>" + cuerpo.getNombreBeneficiario() + "</nombreBeneficiario>\n");
            writer.write("\t\t\t<nifBeneficiario>" + cuerpo.getNifBeneficiario() + "</nifBeneficiario>\n");
            writer.write("\t\t\t<cuentaBeneficiario>" + cuerpo.getCuentaBeneficiario() + "</cuentaBeneficiario>\n");
            writer.write("\t\t\t<controlSuma>" + cuerpo.getControlSuma() + "</controlSuma>\n");
            writer.write("\t\t</pago>\n");
        }
        writer.write("\t</cuerpo>\n");
        writer.write("</mensaje>");
        writer.close();

        return 1;
    } catch(IOException e) {
        return 0;
    }
}
```

Así se ve en la interfaz: