Informes con JasperReports (II)

- Para generar un informe en un proyecto, lo primero es añadir las siguientes bibliotecas: *commons-beanutils-1.9.0.jar, commons-codec-1.5.jar, commons-collections-3.2.1.jar, commons-digester-2.1.jar, commons-loggins-1.1.1..jar, itext-2.1.7.js4.jar, Jackson-annotations-2.1.4.jar, Jackson-core-2.1.4.jar, Jackson-databind-2.1.4.jar, jasperreports-6.2.0.jar, jasperreports-fonts-6.2.0.jar, y tolos.jar.*
- Después habrá que seguir los siguientes pasos:
  - Generar una plantilla, fichero .jrxml, en el que se configura cómo queremos el informe.
  - Compilar el fichero .jrxml para obtener un fichero .jasper
  - Rellenar los datos del informe. Esto generará un fichero .jrprint
  - Exportar el fichero .jrprint al formato deseado: pdf, html, xml, etc.

Informes con JasperReports (III)

- Generar el fichero .jrxml
  - Se puede generar a mano o mediante herramientas gráficas como *iReport*, o mejor aún con *Jaspersoft Studio.*
  - Se especifican los parámetros del informe, la sentencia SQL, los datos a visualizar, la cabecera, el detalle y el pie de la página..
- Compilar la plantilla .jrxml

  JasperReport report =
  JasperCompileManager.compileReport("./plantilla/plantilla.jrxml");
- Rellenar el informe con datos
- *JasperPrint MiInforme = JasperFillManager.fillReport(report, params, conexBD);*
- *donde conexBD es la conexión con la base de datos. Los parámetros tienen que crearse y almacenarse en un HashMap. Ejemplo:*

  Map<String, Object> params = new HashMap<>();
  params.put("titulo", "RESUMEN DATOS DE DEPARTAMENTOS.");
  params.put("autor", "J@");
- *params.put("fecha", java.time.LocalDate.now().toString());*

Informes con JasperReports (IV)

- Exportar el fichero JasperPrint, al formato que se desee.
  - Para visualizarlo con un visor en pantalla:
    JasperViewer.viewReport(MiInforme);
  - Si queremos cerrar el visor sin cerrar la aplicación (por ejemplo en una aplicación con ventanas) se añade *false*:
    JasperViewer.viewReport(MiInforme, false);
  - Para generarlo en html:
    JasperExportManager.exportReportToHtmlFile(MiInforme, reportHTML);
  - Para generarlo en pdf:
    JasperExportManager.exportReportToPdfFile(MiInforme, reportPDF);
  - Para generarlo en xml:
    JasperExportManager.exportReportToXmlFile(MiInforme, reportXML, false); //false para indicar que no hay imágenes

Ejemplo proyecto con JasperReports (I)

```
public class EjempJasper {
   public static void main(String[] args) {
            String reportSource = "./plantilla/plantilla.jrxml";
            String reportHTML = "./informes/Informe.html";
            String reportPDF = "./informes/Informe.pdf";
            String reportXML = "./informes/Informe.xml";
            Map<String, Object> params = new HashMap<>();
            params.put("titulo", "RESUMEN DATOS DE DEPARTAMENTOS.");
            params.put("autor", "J@");
            params.put("fecha", java.time.LocalDate.now().toString());
            try {
                        JasperReport jasperReport =
JasperCompileManager.compileReport(reportSource);
                        Class.forName("com.mysql.jdbc.Driver");
                        Connection conn = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost/ejemplo", "ejemplo", "ejemplo");
                        JasperPrint MiInforme = JasperFillManager.fillReport(jasperReport, params, conn);
```

76

Ejemplo proyecto con JasperReports (II)

```
                JasperViewer.viewReport(MiInforme);
                JasperExportManager.exportReportToHtmlFile(MiInforme, reportHTML);
                JasperExportManager.exportReportToPdfFile(MiInforme, reportPDF);
                JasperExportManager.exportReportToXmlFile(MiInforme, reportXML, false);
                System.out.println("ARCHIVOS CREADOS");
        } catch (CommunicationsException c) {
                System.out.println(" Error de comunicación con la BD. No está arrancada.");
        } catch (ClassNotFoundException e) {
                System.out.println(" Error driver. ");
        } catch (SQLException e) {
                System.out.println(" Error al ejecutar sentencia SQL ");
        } catch (JRException ex) {
                System.out.println(" Error Jasper.");
                ex.printStackTrace();
        }
}
}
```

Plantilla .JRXML

Sección	Descripción
title	Su contenido se imprime una vez al principio del informe
pageHeader	Se imprimirá en cada página
columHeader	Se escribe la cabecera que se va a poner para el detalle. Es decir, los nombres de las columnas que se visualizarán en *detail*
detail	Es el cuerpo del documento, donde se coloca la información a desplegar en forma tabular obtenida de la select
columFooter	Aquí se ponen los totales acumulados, y la información para cada una de las columnas de detalle
pageFooter	Es el pie de página que se repite al final de cada página. Por ejemplo el número de página
summary	Se utiliza para concluir el documento, se imprime una sola vez al final del informe

Gestión de errores (I)

- Hasta ahora, cuando se producía un error la secuencia de llamadas al método que ha producido la excepción y la línea de código donde se producía el error se visualizaba con printStackTrace().
- Cuando se produce un error con SQLException podemos acceder a cierta información usando los siguientes métodos.
  - getMessage(). Devuelve una cadena que describe el error.
  - getSQLState(). Es una cadena que contiene un estado definido por el estándar X/OPEN SQL.
  - getErrorCode(). Es un entero que proporciona el código de error del fabricante.

Gestión de errores (II)

```
try
  { // código }
catch (ClassNotFoundException cn) {cn.printStackTrace));}
catch {SQLException e)
{
  System.out.println("HA OCURRIDO UNA EXCEPCIÓN:");
  System.out.println("Mensaje   :"+e.getMessage());
  System.out.println("SQL estado:"+e.getSQLState());
System.out.println("Cod error :"+e.getErrorCode());
}
```