Enfoque adoptado a la resolución de la Tarea 3

Primeras cosas hechas:

Ahora tengo que generar HTML de archivos XML, he indagado un poco y me he encontrado con las siguiente tecnología:

- **XPath** la cual me sirve para navegar a través de los elementos y atributos de un archivo XML para así obtener los datos y procesarlos/transformarlos.

- **XSLT** puede agregar/eliminar elementos/atributos desde/hacia el archivo de salida (en mi caso HTML). También puede reorganizar y ordenar elementos, realizar pruebas y decidir sobre qué elementos ocultar y mostrar, parece bastante potente. **XSLT usa XPath para buscar información en un documento XML.**

Enfoque adoptado:

Después de enfocarlo un poco pensé en usar la librería que me permitió en la tarea 2.2 generar los archivos XML, se llama **org.w3c.dom** digamos que si para generar los XML antes los escribía, ahora tengo que hacer lo contrario (leerlos), así que opté por seguir explorando un poco el tema de los **Node** y los **Nodelist** y como si de JavaScript se tratase podemos hacer uso de los métodos getElementsByClassName() o getElementById(), aquí he usado un método similar que me ha parecido más cómodo (**getElementsByTagName()**) ya que para esta actividad voy a realizar la acción de leer un documento XML y por lo tanto ya tiene su jerarquía DOM montada, de modo que coger un elemento directamente por su etiqueta y almacenarlo en un **NodeList** no me parece mala idea:

```
Document document = null;
try {
    //Cargamos el document del fichero XML existente
    DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
    DocumentBuilder db = dbf.newDocumentBuilder();

    document = db.parse(new File(rutaFichero));
    document.getDocumentElement().normalize();
} catch (ParserConfigurationException e) {
    e.printStackTrace();
}
if (document != null) {
    //Obtenemos todos los campos de las filas
    NodeList listaFilas = document.getDocumentElement().getElementsByTagName("FILA");
    Element fila;
```

Cabe mencionar que esta tarea se trata de la elaboración de una aplicación WEB que nos permita lo que he comentado anteriormente: Leer un fichero XML e imprimirlo con formato tabular por pantalla.

Segunda etapa del enfoque:

Después de hablar de los **Nodes** cambiamos de tema temporalmente para la elaboración de la interfaz web que tenemos que diseñar, he decidido tomar como recurso el **framework Bootstrap 3,** el cual nunca había utilizado por cuenta propia, buscaba sencillez y esto me lo da.

¿Cómo será mi interfaz?:

Cómo busco la belleza de la sencillez en mi interfaz web lo que he pensado es de alguna forma mostrar un **<select></select> dinámico** por pantalla el cual mediante unas líneas de código acceda a la carpeta que previamente le he indicado y me muestre en el select únicamente los ficheros que haya en ese directorio que tenga el formato **.XML. Cabe mencionar** que para esta tarea voy a hacer uso del **lenguaje JSP** el cual combina **Java y HTML** de manera muy potente, también pensé en JavaScript pero después de darle un giro de tuerca vi que JSP es la clave para esto, aunque no son tecnologías incompatibles y podría usarse, de momento como este no es el objetivo principal vamos a darle rienda al JSP.

Os muestro una captura de como he confeccionado el select combinando JAVA y HTML, como extra de interés también podéis apreciar algunos **class de Bootstrap 3**:

```
<div class="container">
    <form class="form" method="post" action="tarea3.jsp">
        <select class="form-control" name="elegir">
            <%
              String rutaDir = "C:/Users/Administrador/Documents/NetBeansProjects/Tarea2_II/resultados/";
              /**
               * Aquí comprobare que lo que he pasado como ruta es realmente un
               * directorio, a parte almacenare en un array de File todos los
               * ficheros
               */
              //Apuntando a directorio.
              File directorio = new File(rutaDir);

              //Se comprueba si la ruta existe
              if (!directorio.exists()) {
                out.println("<h4 class='text-center'>La ruta " + directorio.getAbsolutePath() + " no existe.</h4>");
                return;
              }

              //Se comprueba si es un directorio
              if (!directorio.isDirectory()) {
                out.println("<h4 class='text-center'>La ruta " + directorio.getAbsolutePath() + " no es un directorio</h4>");
                return;
              }
              out.println("<h4 class='text-center'> Actuando sobre: " + directorio.getAbsolutePath() + "</h4>");

              //obtener todo el contenido del directorio
              File[] lista = directorio.listFiles();
              //List<String> ficherosXML = new ArrayList<String>(); //LISTA
              //se recorre el directorio fichero por fichero
              int contador = 1;
              for (File s : lista) {
                if (s.getName().substring(s.getName().length() - 4).equals(".xml")) {

                    out.println("<option title='"+s.getName()+"' value='" + s.getName() + "'>" + contador + " - " + s.getName() + "</option>");
                    contador++;
                }
              }
            %>
        </select><br>
        <button type="submit" value="Visualizar" class="btn btn-info"><span class="glyphicon glyphicon-eye-open"></span> Visualizar</button>
    </form>
```

1 File directorio = new File(rutaDir);

2 //Se comprueba si la ruta existe

3 //Se comprueba si es un directorio

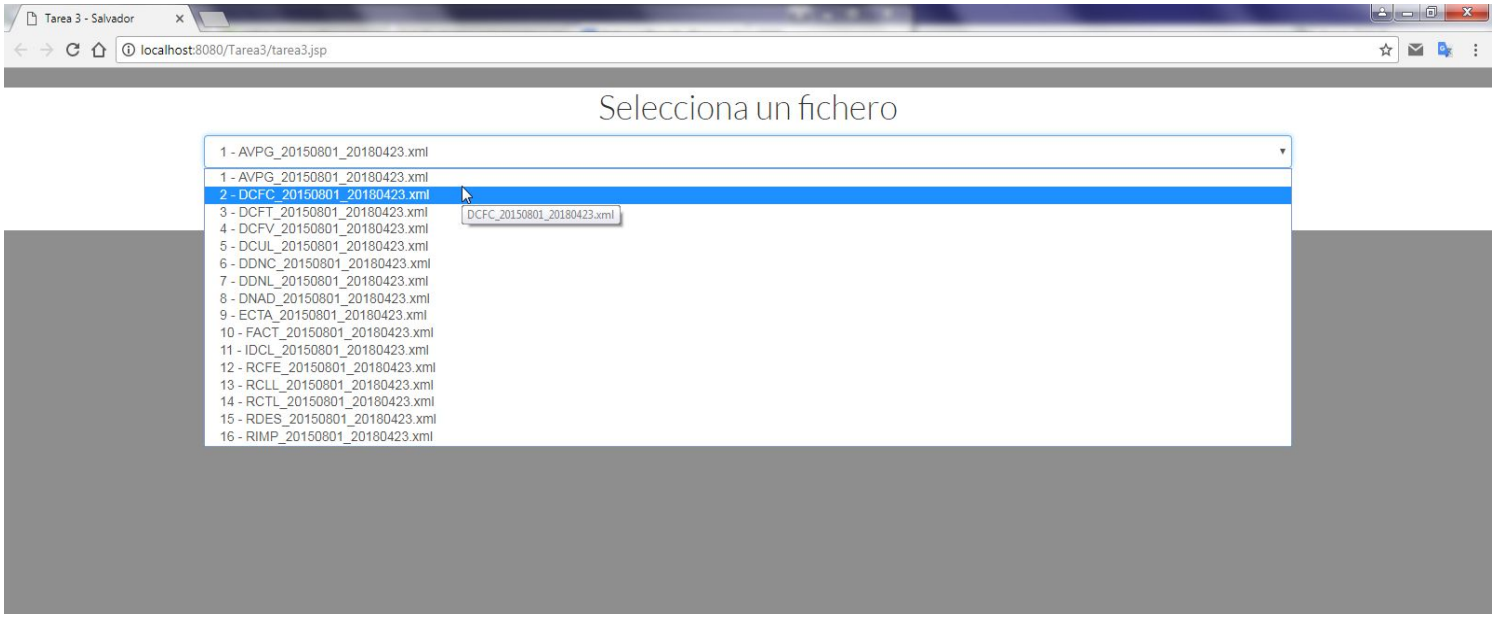
4 //obtener todo el contenido del directorio (Y los almacena en una lista)

Condicional interesante que obtiene el formato del fichero y lo comparar para saber si es XML

Uso de for each para recorrer lista de ficheros

5 Si se da el caso de que es un fichero XML el que pasa por la condicional, hago que se escriba un <option></option> del select con el nombre del fichero

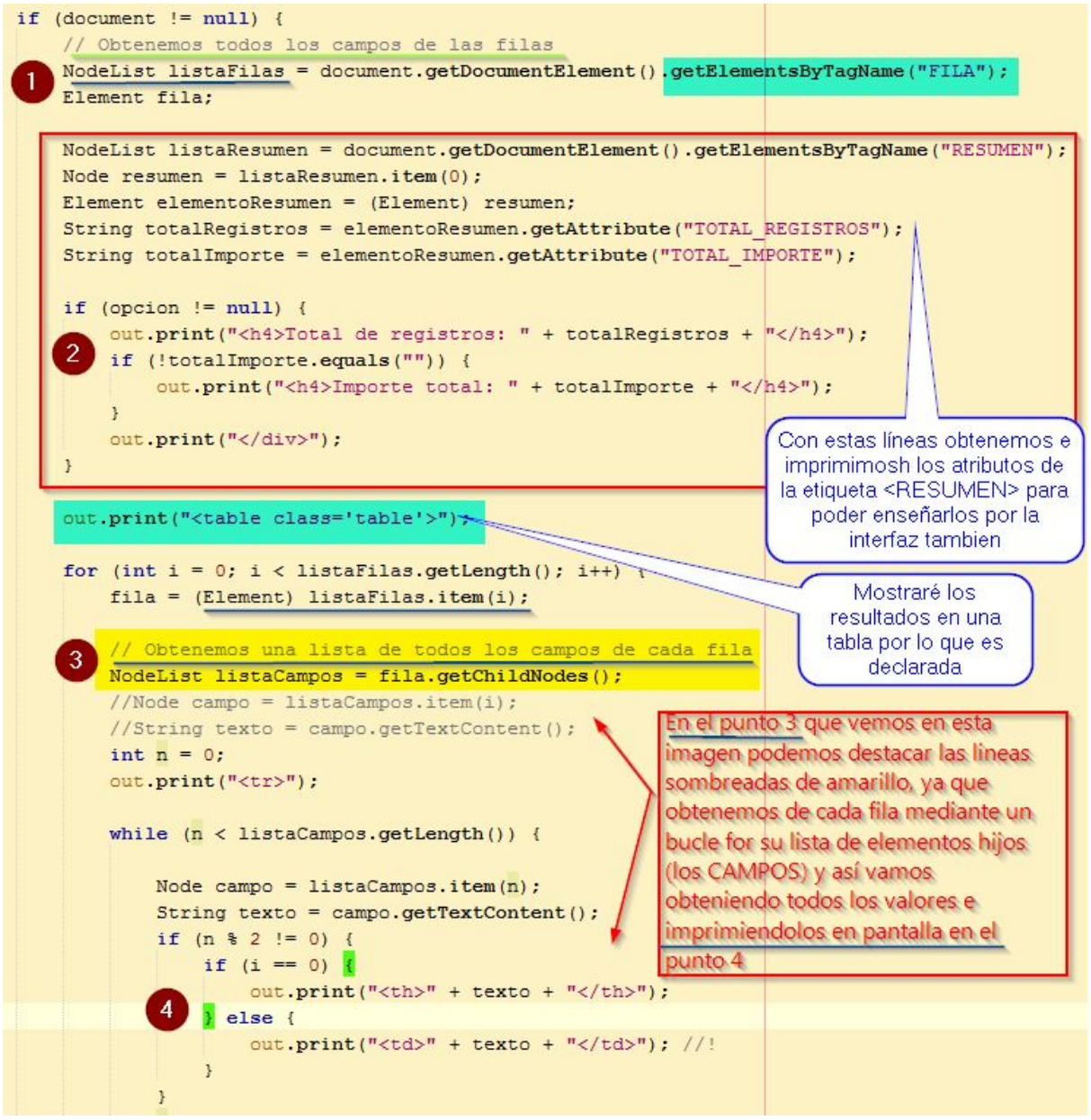
Este es el resultado final de la interfaz web (no funcional, aun no podemos visualizar nuestros XML, pero si seleccionarlos):



Todo va cogiendo forma, ahora tengo que preocuparme de la visualización por pantalla de los XML.

Tercera etapa del enfoque:

Volvemos a hablar de los **Nodes** esta vez más bien de su utilización directamente funcional, así que como una imagen vale más que mil palabras voy a mostrar como he intentado afrontar esta parte final de la actividad:



RESULTADO FINAL: