índice

Enunciado Tarea

Tarea 3.1: Creación, validación y procesamiento de un Fichero XML utilizando DOM

Fecha de entrega No hay fecha de entrega **Puntos** 0

Objetivo: El objetivo de esta tarea es que los alumnos creen un fichero XML que represente información sobre una colección de objetos de su elección (por ejemplo, películas, canciones, productos, etc.). Luego, deben validar el fichero XML utilizando el tipo de acceso DOM y mostrar los datos por pantalla.

1. Elige un tema o categoría para una colección de objetos. Puede ser cualquier cosa que os interese.
2. Diseña una estructura XML que represente la información sobre los objetos de su colección. Asegúrate de incluir al menos tres elementos diferentes y algunos atributos en su estructura XML, así como al menos 3 niveles de anidación.
   Ejemplo: <bibliotecas> <biblioteca> <libro> .....
3. Crea un fichero XML válido. Asegúrate de que el fichero XML cumpla con las reglas y la estructura definida.
4. Implementa un programa .java que utilice el tipo de acceso DOM para cargar y validar el fichero XML creado en el paso 3.
5. Una vez validado el fichero XML, extrae y muestra por pantalla la información de los objetos de su colección.
6. Comenta el código para explicar las diferentes partes y cómo funciona el proceso de validación y extracción de datos.
7. Entrega tanto el fichero XML como el código fuente de su programa junto con cualquier documentación adicional.

XML de Ejemplo:
prueba.xml
parserDOM.java

1. Crear el xml

He creado el xml de canciones de mi playlist más recientes.

Ejemplo con una canción, hay más xml..

```xml
<!-- XML Playlist. -->
<playlist>
    <cancion id="1">
        <titulo>Marta tiene un marcapasos</titulo>
        <artista>
            <nombre>Héroes del Silencio</nombre>
            <nacionalidad>Español</nacionalidad>
        </artista>
        <album>
            <nombre>Senderos de Traición</nombre>
            <anio>1990</anio>
        </album>
        <generos>
            <genero>Rock</genero>
            <genero>Pop</genero>
        </generos>
    </cancion>
```

2. Validar XML con DTD

En la parte superior del documento xml.

```xml
<!-- Validación DTD. -->
<!DOCTYPE playlist [
    <!ELEMENT playlist (cancion+)>
    <!ELEMENT cancion (titulo, artista, album, generos)>
    <!ATTLIST cancion id CDATA #REQUIRED>
    <!ELEMENT titulo (#PCDATA)>
    <!ELEMENT artista (nombre, nacionalidad)>
    <!ELEMENT nombre (#PCDATA)>
    <!ELEMENT nacionalidad (#PCDATA)>
    <!ELEMENT album (nombre, anio)>
    <!ELEMENT anio (#PCDATA)>
    <!ELEMENT generos (genero+)>
    <!ELEMENT genero (#PCDATA)>
]>
```

DOM Y SAX: Son herramientas que nos ofrecen la posibilidad de leer ficheros XML. Estas

herramientas se dedican a verificar si sintácticamente son ficheros válidos. Son los llamados “parsers”

o analizadores. En este ejemplo se puede ver también como podemos hacer consultas xon Xpath

3. Crear parserDOM.java

Creamos un try-catch:

Para manejar los errores

dentro del try-catch

Creamos una instancia de DocumentBuilderFactory:

Actúa como una fábrica (factory) que configura el DocumentBuilder que creemos después

Estos objetos se utilizan para analizar (parsear) documentos XML.

Comprobar que el fichero que se cargue este bien validado y sin espacios en blanco:

setValidating()

setIgnoringElementContentWhitespace()

Creamos el DocumentBuilder con la factoria creada:

```
DocumentBuilder builder = factory.newDocumentBuilder();
```

Especificamos el fichero xml que queremos analizar:

con clase File file y la ruta del archivo

Parsear el archivo XML

Convierte el archivo XML en un objeto document que podemos manipular en nuestro codigo

```
Document doc = builder.parse(file);
```

Hacer consultas con XPath

Creamos instancia de Xpath

```
XPath xPath = XPathFactory.newInstance().newXPath();
```

Definir consulta o expresión

```
String expression = "/playlist/cancion";
```

Evaluar la consulta

Con compile() y evaluate()

y que se almacene en una NodeList

```
NodeList listaNodos = (NodeList) xPath.compile(consulta).evaluate(doc, XPathConstants.NODESET);
```

Luego con los for, he recorrido el NodeList (la lista, cogiendo los atributos y elementos para mostrar la información del xml)

```
for (int i = 0; i < listaNodos.getLength(); i++) {

    Node nNode = listaNodos.item(i);
     System.out.println("\nElemento Actual: " + nNode.getNodeName());

            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) nNode;
                System.out.println("Título: " + eElement.getElementsByTagName("titulo").item(0).getTextContent());

                // Ahora accedemos al nombre del artista correctamente
                Element artistaElement = (Element) eElement.getElementsByTagName("artista").item(0);
                String nombreArtista = artistaElement.getElementsByTagName("nombre").item(0).getTextContent();
                System.out.println("Artista: " + nombreArtista);

                // Si necesitas la nacionalidad
                String nacionalidadArtista = artistaElement.getElementsByTagName("nacionalidad").item(0)
                        .getTextContent();
                System.out.println("Nacionalidad: " + nacionalidadArtista);

                System.out.println("=========="); // Separador
            }
        }
```

```java
public class parserDOM {
    public static void main(String[] args) {
        try {
            // Crear una instancia de DocumentBuilderFactory
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            factory.setValidating(true);
            factory.setIgnoringElementContentWhitespace(true);

            // Crear un objeto DocumentBuilder
            DocumentBuilder builder = factory.newDocumentBuilder();

            // Ruta del XML que queremos analizar
            File file = new File("playlist.xml");

            // Parsear el XML y obtener un objeto Document
            Document doc = builder.parse(file);


            // Crear un objeto XPATH para consultar el documento XML
            XPath xPath = XPathFactory.newInstance().newXPath();
            // Definir la expresión XPath para obtener la lista de nodos
            String consulta = "/playlist/cancion";
            NodeList listaNodos = (NodeList) xPath.compile(consulta).evaluate(doc, XPathConstants.NODESET);

            // Iterar la lista
            for (int i = 0; i < listaNodos.getLength(); i++) {
                Node nNode = listaNodos.item(i);
                System.out.println("\nElemento Actual: " + nNode.getNodeName());

                if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element eElement = (Element) nNode;
                    System.out.println("Título: " + eElement.getElementsByTagName("titulo").item(0).getTextContent());

                    // Ahora accedemos al nombre del artista correctamente
                    Element artistaElement = (Element) eElement.getElementsByTagName("artista").item(0);
                    String nombreArtista = artistaElement.getElementsByTagName("nombre").item(0).getTextContent();
                    System.out.println("Artista: " + nombreArtista);

                    // Si necesitas la nacionalidad
                    String nacionalidadArtista = artistaElement.getElementsByTagName("nacionalidad").item(0)
                            .getTextContent();
                    System.out.println("Nacionalidad: " + nacionalidadArtista);

                    System.out.println("=========="); // Separador
                }
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```