

MANEJO DE FICHEROS XML CON DOM

Javier García-Retamero Redondo

¿QUÉ NECESITAMOS?

- Métodos para cargar documentos desde una fuente de datos.
 - Las clases e interfaces de los paquetes:
 - **org.w3c.dom**
 - **Interfaz:** DOMImplementation
 - **javax.xml.parsers**
 - **Clases:** DocumentBuilderFactory y DocumentBuilder
- Generar un fichero XML a partir de un árbol DOM:
 - **java.xml.transform**

MANEJO DE FICHEROS XML CON DOM

CLASES A IMPORTAR



CLASES A IMPORTAR

- Aquí tenemos los import a realizar para poder trabajar:

```
import org.w3c.dom.*;  
import javax.xml.parsers.*;  
import javax.xml.transform.*;  
import javax.xml.transform.dom.*;  
import javax.xml.transform.stream.*;  
import java.io.*;
```

MANEJO DE FICHEROS XML CON DOM

EL PARSER



EL PARSER

- **Para trabajar con XML debemos utilizar un parser.**

```
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
```

- **Creamos un constructor de documentos:**

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


MANEJO DE FICHEROS XML CON DOM

CONSTRUIR EL DOCUMENTO EN MEMORIA



CONSTRUIR EL DOCUMENTO

- Creamos un documento vacío con un nodo principal:

```
DOMImplementation implementation = builder.getDOMImplementation();  
Document document = implementation.createDocument(null, "Empleados", null);
```

- Asignamos la versión de XML:

```
document.setXmlVersion("1.0");
```

DOCUMENT

Principal

Llevaríamos escrito:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

```
<Empleados>
```


CONSTRUIR EL DOCUMENTO

- Creamos el nodo "Empleado":

```
Element elem = document.createElement("Empleado");
```

- Lo pegamos al documento:

```
document.getDocumentElement().appendChild(elem);
```

Llevaríamos escrito:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

```
<Empleados>
```

```
<Empleado>
```

getDocumentElement


elem

CONSTRUIR EL DOCUMENTO

- Para crear un nodo:
- Creamos el hijo "id" y lo añadimos:
`Element elemText = document.createElement("id");`
`elem.appendChild(elemText);`
- Asignamos un valor al hijo y lo añadimos:
`Text valor = document.createTextNode("1");`
`elemText.appendChild(valor);`

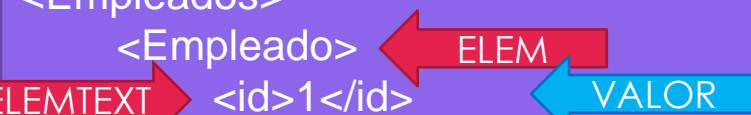
Llevaríamos escrito:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Empleados>
  <Empleado>
    <id>
```



Llevaríamos escrito:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Empleados>
  <Empleado>
    <id>1</id>
```



CONSTRUIR EL DOCUMENTO

- Al ser una tarea muy repetitiva, para crear los nodos construimos un método que será el encargado de llevarlo a cabo.

```
static void CrearNodo( String elemento,  
                      String valorEntrada,  
                      Element raiz,  
                      Document document){  
    Element elemText = document.createElement(elemento); //creamos el hijo  
    raiz.appendChild(elemText); //pegamos el elemento hijo a la raiz  
    Text valor = document.createTextNode(valorEntrada); //damos valor  
    elemText.appendChild(valor); //pegamos el valor al elemento  
}
```

MANEJO DE FICHEROS XML CON DOM

PASAR EL DOCUMENTO DE MEMORIA A DISCO



PASAR EL DOCUMENTO DE MEMORIA A DISCO

- Creamos un documento vacío con un nodo raíz:

```
DOMImplementation implementation = builder.getDOMImplementation();  
Document document = implementation.createDocument(null, "Empleados", null);
```

- Asignamos la versión de XML:

```
document.setXmlVersion("1.0");
```

PASAR EL DOCUMENTO DE MEMORIA A DISCO

- Indicamos el origen a convertir:

```
Source source = new DOMSource(document);
```

- Especificamos el archivo de salida:

```
Result salida = new StreamResult(new File("Empleados.xml"));
```

- Obtenemos un Transformer y lanzamos la conversión:

```
Transformer transformer=TransformerFactory.newInstance().newTransformer();  
transformer.transform(source,salida);
```


MANEJO DE FICHEROS XML CON DOM

MOSTRAR EL DOCUMENTO POR PANTALLA EN LUGAR DE GUARDAR EN
DISCO



MOSTRAR DOCUMENTO POR PANTALLA

- Indicamos el origen a convertir:

```
Source source = new DOMSource(document);
```

- Especificamos el archivo de salida:

```
Result salida = new StreamResult(System.out);
```

- Obtenemos un Transformer y lanzamos la conversión:

```
Transformer transformer=TransformerFactory.newInstance().newTransformer();  
transformer.transform(source,salida);
```

MANEJO DE FICHEROS XML CON DOM

LECTURA DE UN ARCHIVO XML



LECTURA DE UN ARCHIVO XML

- Para trabajar con XML debemos utilizar un parser.

```
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
```

- Creamos un constructor de documentos:

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

LECTURA DE UN ARCHIVO XML

- Cargamos el documento en memoria con el método parse:

```
Document document = builder.parse(new File("Empleados.xml"));
```

- Normalizamos el documento (elimina inconsistencias del archivo xml):

```
document.getDocumentElement().normalize();
```

- Para obtener el elemento raíz:

```
document.getDocumentElement().getNodeName();
```

Ejemplo: `System.out.printf("Elemento raíz: %s %n",`

```
document.getDocumentElement().getNodeName());
```

LECTURA DE UN ARCHIVO XML

- Para obtener una lista de nodos con un determinado nombre:
`NodeList empleados = document.getElementsByTagName("Empleado");`

- Para recorrer la lista anterior haríamos un bucle desde `i=` hasta `empleados.getLength()-1`:
`Node emple = empleados.item(i);`

- Verificamos si lo leído es un nodo element y si es así convertimos el nodo en un objeto Element:

```
if (emple.getNodeType() == Node.ELEMENT_NODE) {  
    Element elemento = (Element) emple;
```


LECTURA DE UN ARCHIVO XML

- Para coger la lista de nodos “id” del nodo empleado (como sólo hay un identificador por empleado cogemos el elemento cero):

```
NodeList nodo = elemento.getElementsByTagName("id").item(0).getChildNodes();
```

- Para pasar el elemento de la lista a un objeto de tipo Nodo (como sólo contiene un valor cogemos el elemento cero):

```
Node valornodo = (Node) nodo.item(0);
```

- Para visualizar el contenido del nodo anterior:

```
System.out.println(valornodo.getNodeValue());
```

LECTURA DE UN ARCHIVO XML

- Otra opción si queremos mostrar el contenido de un nodo sería:
 - Para obtener el texto de un nodo "id" del nodo empleado (como sólo hay un identificador por empleado cogemos el elemento cero):

```
elemento.getElementsByTagName("id").item(0).getTextContent();
```

Ejemplo: `System.out.printf("ID= %s %n",`

```
elemento.getElementsByTagName("id").item(0).getTextContent());
```