Actividad 1.11 Tratamiento de ficheros XML con DOM I

A. Investiga sobre las clases de DOM para el tratamiento de ficheros XML.

Marcos Alcalde

¿Para qué se utiliza DOM en JAVA?

- En **Java, el DOM** (Document Object Model) se utiliza para leer, navegar, modificar y crear documentos XML (y HTML) en la memoria mediante una estructura de árbol de objetos

¿En qué otros ámbitos, además de en programación con JAVA, se utiliza el enfoque DOM?

Se utiliza en cualquier ámbito que requiere estructurar, acceder y manipular documentos de texto marcado (como HTML y XML) como un árbol de objetos.

Lectura de ficheros XML con DOM: ¿Cuáles son las clases principales utilizadas para leer y cargar un fichero XML usando DOM?

En el entorno de programación de Java (el estándar más común para la API DOM de XML), las clases principales utilizadas para leer y cargar un fichero XML son:

- **DocumentBuilderFactory:** Es la clase encargada de crear una fábrica (instancia) que permite configurar y obtener los constructores de documentos XML. Se inicializa habitualmente mediante el método estático .newInstance().

- **DocumentBuilder:** Es el analizador (parser) que procesa el flujo de datos o el archivo XML y lo convierte en un árbol de nodos en memoria. Se obtiene a partir de la fábrica y su método principal es .parse().

- **Document:** Representa el documento XML completo ya cargado en memoria con una estructura de tipo árbol jerárquico (Document Object Model). Permite acceder a los elementos, atributos y nodos hijos.

- **File:** Se utiliza para apuntar y referenciar la ruta física del archivo XML en el sistema de ficheros que se desea leer.

¿Qué clases se usan para acceder a los nodos asociados a las etiquetas del fichero XML? ¿Y para acceder a la información que tienen?

Clases que se usan para acceder a los nodos (etiquetas)

- **Document:** Representa el documento XML completo y actúa como el nodo raíz del árbol. Se usa para empezar a buscar elementos (por ejemplo, con getElementsByTagName()).

- **Node**: Es la interfaz base de la que heredan la mayoría de los nodos del árbol (elementos, textos, atributos). Permite navegar por la jerarquía (getParentNode(), getFirstChild(), getChildNodes()).

- **Element:** Representa específicamente una etiqueta del archivo XML (hereda de Node). Se usa para referenciar las etiquetas y filtrar por su nombre.

- **NodeList:** Es una lista que almacena una colección de nodos (por ejemplo, todas las etiquetas con un nombre específico) para recorrerlas mediante un índice.

Clases e interfaces para acceder a la información que tienen

• Node (métodos de contenido):

- getNodeName(): Devuelve el nombre del nodo (el nombre de la etiqueta).
- getNodeValue(): Devuelve el valor asociado al nodo (en nodos de tipo texto o atributo).
- getTextContent(): Extrae todo el contenido de texto que se encuentra dentro de una etiqueta

Escritura de ficheros XML con DOM:

¿Cuáles son las clases principales utilizadas para escribir un fichero XML?

- **TransformerFactory:** Factoría abstracta que sirve para crear instancias de la clase Transformer.

- **Transformer:** Objeto encargado de realizar la transformación o el volcado de los datos del árbol DOM hacia un flujo de salida.

- **DOMSource:** Actúa como el adaptador de entrada, encapsulando el objeto Document (el árbol DOM en memoria que se desea escribir).

- **StreamResult:** Actúa como el adaptador de salida, indicando el destino del fichero XML (un File, un OutputStream o un Writer)

¿Qué clases se usan para crear una etiqueta, el contenido asociado a una etiqueta o el contenido asociado a un atributo?

Para crear y gestionar estos componentes en el estándar DOM (W3C), se utilizan las siguientes interfaces o clases:

- **Crear una etiqueta (elemento XML):** Se usa la interfaz Element (creada mediante el método createElement("nombre") del objeto Document).

- **Crear el contenido asociado a una etiqueta (texto interno):** Se usa la interfaz Text (creada mediante el método createTextNode("contenido") del objeto Document).

- **Crear el contenido o valor asociado a un atributo:** Se usa la interfaz Attr (creada mediante createAttribute("nombre") del objeto Document o asignada directamente con el método setAttribute() / propiedad value de la clase Element).