Procesamiento de documentos XML

Para procesar (*parse*) la información contenida en documentos XML dispone de diversas técnicas, cada una de las cuales tiene sus propias características. Las dos técnicas (modelos) fundamentales que existen para el procesamiento de documentos XML de manera independiente al lenguaje de programación son las siguientes:

- **DOM** (Document Object Model)
  - ∴ Es un estándar diseñado por W3C (Consorcio World Wide Web)
  - ∴ Basado en la estructura de árbol de XML
  - ∴ Almacena todo el árbol en memoria (no recomendado para grandes documentos)
  - ∴ Acceso directo a cualquier parte del documento
- **SAX** (Simple API for XML)
  - ∴ Con licencia de dominio público, se ha convertido en un estándar "de hecho". Inicialmente diseñado para Java, pero se ha extendido a varios lenguajes.
  - ∴ Se basa en un funcionamiento basado en flujo (stream), de manera que se va procesando según se recibe su contenido. No es necesario cargar todo el documento en memoria para procesarlo.
  - ∴ Su programación está basada en el tratamiento de eventos. La información es empujada (pushing) hacia la aplicación al lanzarse un evento cuando se detecta un determinado tipo de información.
  - ∴ No permite el acceso directo a una determinada parte del documento.
  - ∴ No permite la modificación de su contenido.
- **StAX** (Streaming API for XML)
  - ∴ Diseñado como una técnica intermedia entre las dos anteriores.
  - ∴ La aplicación avanza un cursor trayendo (pulling) la información según se va necesitando, sin posibilidad de hacer acceso directo.
  - ∴ Permite la escritura de datos en el documento.

De ™/®The World Wide Web Consortium (W3C) - Vulphere optimized (cropped) original version from:
https://www.w3.org/Icons/XML/xml.svg
Dominio público:
https://commons.wikimedia.org/w/index.php?curid=102284664

Comparativa de técnicas de procesadores (parsers) de documentos XML

	DOM	**SAX**	**StAX**
Tipo de API	Árbol en memoria	Flujo por eventos (*Streaming, push*)	Flujo por extracción (*Streaming, pull*)
Facilidad de uso	Alta	Media	Alta
Capacidad XPath	Sí	No	No
Eficiencia de memoria y CPU	Varía	Buena	Buena
Acceso directo	Sí	No	No
Lectura XML	Sí	Sí	Sí
Escritura XML	Sí	No	Sí
Crear, leer, actualizar, borrar	Sí	No	No

Además de estas técnicas para el procesamiento de los datos contenidos en documentos XML existen otras técnicas para su tratamiento como las siguientes:

- **XPATH:** Es un lenguaje que permite realizar consultas mediante expresiones que recorren y procesan un documento XML.
- **XSLT:** (XML Style Sheet Language Transformations) Estándar definido por W3C. Presenta una forma de transformar documentos XML en otros e incluso a formatos que no son XML.

La API que incorpora Java de manera predeterminada para manipular documentos XML se denomina **JAXP** (Java API for XML Processing), que permite utilizar todas esas técnicas mencionadas anteriormente. Dispone de diversas clases agrupadas en los siguientes paquetes:

- **javax.xml.parsers:** Proporciona un interfaz común para los diferentes procesadores SAX y DOM.
- **org.w3c.dom:** Define las clases de los componentes del modelo DOM.
- **org.xml.sax:** Define la API básica de SAX.
- **javax.xml.transform:** Define la API de XSLT para transformar los documentos XML en otros formatos.
- **javax.xml.stream:** Proporciona la API para StAX.
- **javax.xml.xpath:** Proporciona la API para XPath.

Nodos en el modelo DOM de XML

En el modelo DOM para XML, se considera un nodo a **cualquier cosa** que se encuentra dentro del documento:

- Todo el documento en su conjunto es un **nodo documento.**
- Cada elemento XML es un **nodo elemento.**
- El texto de los elementos XML son **nodos texto.**
- Cada atributo es un **nodo atributo.**
- Los comentarios son **nodos comentario.**

Para manipular los nodos en Java se dispone de la **clase Node** → API Java 7 – API Java 8 – API Java 9, que se encuentra en el paquete ***org.w3c.dom*** al igual que las clases de los distintos tipos de nodos que se acaban de comentar. Los nombres de estas clases coinciden con las de otras clases de otros paquetes de Java, por lo que puede ser recomendable **importar todas las clases de ese paquete** para evitar confusiones:

```
import org.w3c.dom.*
```

En los ejemplos se trabajará con las siguientes clases:

- **Document.** Es un objeto que equivale a un ejemplar de un documento XML. Permite crear nuevos nodos en el documento. Sólo puede haber 1, es el objeto principal del árbol.La raíz del documento se obtendrá con: document.getDocumentElement().
- **Element.** Representa a cada elemento del documento XML, como son las etiquetas.
- **Node.** Representa a cualquier nodo del documento (puede ser un document, element, un atributo, un texto o un comment.
- **NodeList.** Contiene una lista con los nodos hijos de un nodo.
- **DocumentBuilderFactory**, se utiliza para construir el parser.
- **DOMImplementation**, permite crear un documento con un elemento raíz (un documento bien formado). Un objeto Document se crea a partir de un DOMImplementation

Ejercicios con XML y Java:

1. Crea el fichero miempresa.xml utilizando lo aprendido.

```
<? Xml version="1.0" encoding="UTF-8" standalone="no"?>
<empresa>
	<empleado id="1">
		<nombre>Roberto</nombre>
		<apellidos>Gutiérrez</apellidos>
		<seccion>Desarrollo</seccion>
		<salario>2800</salario>
	</empleado>
	<empleado id="2">
		<nombre>José</nombre>
		<apellidos>Jiménez</apellidos>
		<seccion>Investigación</seccion>
		<salario>2700</salario>
	</empleado>
</empresa>
```

2. Crea el fichero cd.xml utilizando lo aprendido.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<CD_OFERTA>
	<CD>
		<album>Iron Maiden</album>
		<banda>Iron Maiden</banda>
		<pais>UK</pais>
		<discografica>EMI</discografica>
		<precio>11.95</precio>
		<anyo>1980</anyo>
	</CD>
	<CD>
		<album>Killers</album>
		<banda>Iron Maiden</banda>
		<pais>UK</pais>
		<discografica>EMI</discografica>
		<precio>11.95</precio>
		<anyo>1981</anyo>
	</CD>
	<CD>
		<album>The number of the beast</album>
		<banda>Iron Maiden</banda>
		<pais>UK</pais>
		<discografica>EMI</discografica>
		<precio>11.95</precio>
		<anyo>1982</anyo>
	</CD>
```

```
    <CD>
        <album>Piece of mind</album>
        <banda>Iron Maiden</banda>
        <pais>UK</pais>
        <discografica>EMI</discografica>
        <precio>11.95</precio>
        <anyo>1983</anyo>
    </CD>
    <CD>
        <album>Powerslave</album>
        <banda>Iron Maiden</banda>
        <pais>UK</pais>
        <discografica>EMI</discografica>
        <precio>11.95</precio>
        <anyo>1984</anyo>
    </CD>
    <CD>
        <album>Live after death</album>
        <banda>Iron Maiden</banda>
        <pais>UK</pais>
        <discografica>EMI</discografica>
        <precio>11.95</precio>
        <anyo>1985</anyo>
    </CD>
    <CD>
        <album>Somewhere in time</album>
        <banda>Iron Maiden</banda>
        <pais>UK</pais>
        <discografica>EMI</discografica>
        <precio>11.95</precio>
        <anyo>1986</anyo>
    </CD>
    <CD>
        <album>Seventh son of a seventh son</album>
        <banda>Iron Maiden</banda>
        <pais>UK</pais>
        <discografica>EMI</discografica>
        <precio>11.95</precio>
        <anyo>1988</anyo>
    </CD>
</CD_OFERTA>
```

3. Lee el contenido del fichero miempresa.xml utilizando lo aprendido. El resultado en consola debe ser parecido a este:

```
Empleado 1
***************
nombre:Robert
apellidos:Gutiérrez
seccion:desarrollo
salario:3900

Empleado 2
***************
nombre:Susana
apellidos:Fernández
seccion:informática
salario:2500
```

De ™/®The World Wide Web Consortium (W3C) - Vulphere optimized (cropped) original version from:
https://www.w3.org/Icons/XML/xml.svg
Dominio público:
https://commons.wikimedia.org/w/index.php?curid=102284664