PRÁCTICA 3 - PTI

Kais Harim, Omair Iqbal

6.03.2017

INTRODUCCIÓN

El objetivo de esta sesión es entender el funcionamiento de JDOM, ya que nos facilitará crear documentos XML a partir de esta tecnología.

JSON

JSON hoy en día esta sustituyendo a XML, pero XML sigue siendo una herramienta muy amplia y usada que permite realizar funcionalidades que otros lenguajes no permiten. Por eso vamos a estudiar e introducir el caso de JSON.

JSON, acrónimo de JavaScript Object Notation, es un formato de texto ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript aunque hoy, debido a su amplia adopción como alternativa a XML, se considera un formato de lenguaje independiente.

Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos es que es mucho más sencillo escribir un analizador sintáctico (parser) de JSON. En JavaScript, un texto JSON se puede analizar fácilmente usando la función eval(), lo cual ha sido fundamental para que JSON haya sido aceptado por parte de la comunidad de desarrolladores AJAX, debido a la ubicuidad de JavaScript en casi cualquier navegador web.

En la práctica, los argumentos a favor de la facilidad de desarrollo de analizadores o del rendimiento de los mismos son poco relevantes, debido a las cuestiones de seguridad que plantea el uso de eval() y el auge del procesamiento nativo de XML incorporado en los navegadores modernos. Por esa razón, JSON se emplea habitualmente en entornos donde el tamaño del flujo de datos entre cliente y servidor es de vital importancia (de aquí su uso por Yahoo, Google, etc, que atienden a millones de usuarios) cuando la fuente de datos es explícitamente de fiar y donde no es importante el no disponer de procesamiento XSLT para manipular los datos en el cliente.

Si bien es frecuente ver JSON posicionado contra XML, también es frecuente el uso de JSON y XML en la misma aplicación. Por ejemplo, una aplicación de cliente que integra datos de Google Maps con datos meteorológicos en SOAP hacen necesario soportar ambos formatos.

XML

XML, siglas en inglés de eXtensible Markup Language, traducido como "Lenguaje de Marcado Extensible" o "Lenguaje de Marcas Extensible", es un meta-lenguaje que permite definir lenguajes de marcas desarrollado por el World Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible. Proviene del lenguaje SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML) para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones deben comunicarse entre sí o integrar información.1

XML no ha nacido únicamente para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable.

XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

JDOM

JDOM es una biblioteca de código abierto para manipulaciones de datos XML optimizados para Java. A pesar de su similitud con DOM del consorcio World Wide Web (W3C), es una alternativa como documento para modelado de objetos que no está incluido en DOM. La principal diferencia es que mientras que DOM fue creado para ser un lenguaje neutral e inicialmente usado para manipulación de páginas HTML con JavaScript, JDOM se creó específicamente para usarse con Java y por lo tanto beneficiarse de las características de Java, incluyendo sobrecarga de métodos, colecciones, etc. Para los programadores de Java, JDOM es una extensión más natural y correcta. Se asemeja al sistema RMI optimizado para Java (invocación remota de métodos), y se amolda mejor que CORBA (arquitectura de intermediario solicitador de objetos comunes) que es más neutral respecto a los lenguajes.

Como dato curioso, aunque JDOM parezca un acrónimo de Java Document Object Model (Documento de Modelado de Objetos en Java), esto no es así, siendo desmentido por el propio proyecto de JDOM.

RECURSOS

1. Git de la asignatura
2. Java
3. Herramientas para editar documentos java

PROCEDIMIENTO

En primer lugar hemos instalado Java en nuestra máquina siguiendo la guía de la práctica.

Una vez instalado java y puesto todo a punto hemos comenzado a programar la web que pedía el enunciado.
A continuación veremos las funcionalidades que se nos ha pedido implementar en esta práctica:

- Reset: crea un documento XML inicializado.
- New: pide los datos al usuario y añade un alquiler al fichero XML.
- List: muestra por pantalla el fichero XML .
- Xslt: transforma el fichero XML a HTML usando una plantilla xslt.

RESET

Para ello, se llama a la función createDocument(), la cual crea un elemento llamado carrental, y a partir de este se crea el documento. Una vez creado el documento, este se pasa a la función outputDocumentToFile(Document) que usando XMLOtputter, escribe el documento JDOM en el ficher carRental.xml.

Para ejecutar esta funcionalidad, escribiremos en terminal:

- Javac carRental.java . Para compilar el archivo
- Java carRental reset . Para ejecutar la funcionalidad

NEW

La función “addNewRental()” será la encargada de ralizar la tarea de pedir los datos necesarios al usuario para realizar un nuevo alquiler (Fabricante, Modelo, Fecha de inicio del alquiler y fecha de fin del alquiler). Posteriormente, se crea un elemento en el que se guarda la información de cada parámetro del alquiler (Fabricante, Modelo, etc). Estos se guardarán como elementos hijos del elemento “rental”. Una vez asignados todos, se retornará el elemento “rental” a la función “addNewRental()”, que lo añadirá como hijo al elemento root del Document.
Tras esto, se escribirá el Document mediante la función “outputDocumentToFile(Document)”, con lo que se finalizará el proceso de añadir un nuevo alquiler.

El resultado tendrá la siguiente forma:

```
<?xml version="1.0" encoding="UTF-8"?>
<carrental>
    <rental id="44">
      <make>Toyota</make>
      <model>Celica</model>
      <start>2016-09-24-06:00</start>
      <end>2016-09-26-06:00</end>
    </rental>
</carrental>
```

LIST

Esta función simplemente muestra por pantalla el contenido del archivo carrental.xml. Para ello se usa la funcion readDocument para generar un Document a partir de un objeto File de Java. Una vez se ha creado este documento, se pasa a la función outputDocument, que lo imprime por pantalla. Para ello, igual que en el apartado de Reset, se usa XMLOutputter para escribir el documento por el canal de salida correspondiente. En este caso el terminal.

XSLT

Para crear el documento HTML a partir del documento XSLT se usa la clase TransformerFactory que permite, a partir de un fichero xslt y un document crearlo. Para obtener el Document, se llama a a la función readDocument. Una vez leído este y transformado con la función executeXSLT, se escribe por pantalla el resultado.

CONCLUSIONES

Esta práctica nos ha resultado muy útil, ya que nos ha permitido aprender el funcionamiento de un recurso que no conocíamos como es JDOM y ver el gran potencial que tiene. En definitiva una práctica muy interesante acorde con la materia de la especialidad y muy útil para mejorar nuestras capacidades. Sobre todo el hecho de realizar el mismo ejercicio de distintas maneras, nos está haciendo madurar nuestro criterio para resolver los problemas que se nos presenten en el futuro.