Servicios Multimedia Avanzados
XML con Java

Máster I. de Telecomunicación

Bibliografía disponible on-line

- **Para una completa revisión del tema podéis mirar:**
  - **http://www.cafeconleche.org/books/xmljava/**
  - **http://es.scribd.com/doc/15490934/Java-and-XML**

APIs en Java para procesar XML

- Algunas posibilidades:
  - API Java para Procesar XML (**JAXP**) — procesa documentos XML usando varios analizadores.
  - Arquitectura Java para Uniones XML (**JAXB**) — mapea elementos XML a clases del lenguaje Java.
  - API Java para Mensajería XML (**JAXM**) — envía mensajes SOAP sobre Internet de una forma estándar.
  - API Java para Registros XML (**JAXR**) — proporciona una forma estándar para acceder a registros de negocios que comparte información.
  - API Java para RPC basado en XML (**JAX-RPC**) — envía llamadas a métodos SOAP a partes remotas sobre Internet y recibe los resultados.

JAXP - Java API for XML Processing

- **Incluye varios APIs en realidad:**
  - **DOM -** Document Object Model
  - **SAX -** Simple API for XML
  - **StAX -** Streaming API for XML
  - **XSLT -** Extensible Stylesheet Language Transformations

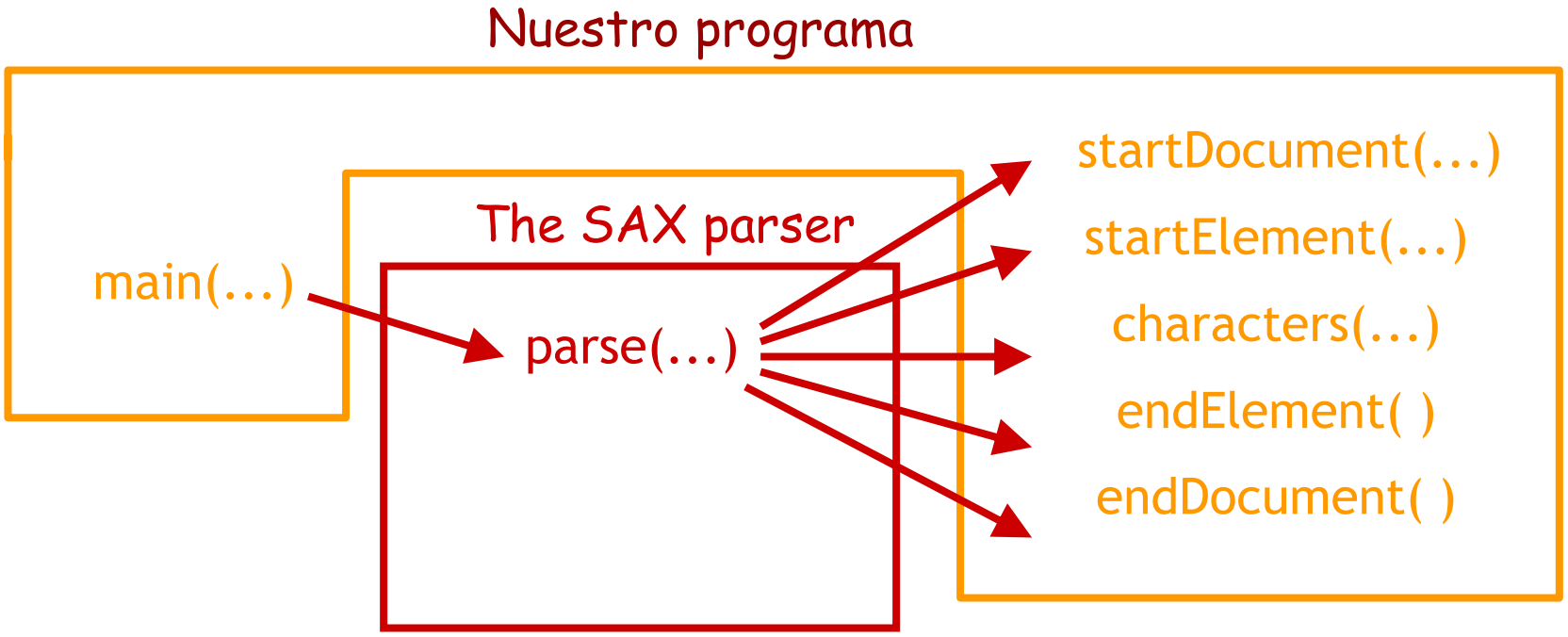
- **Nos centraremos en SAX**

SAX y DOM

- SAX y DOM son estándares para procesar documentos XML
  - DOM es un estándar W3C
  - SAX es un estándar ad-hoc (pero muy popular)
  - SAX fue desarrollado por David Megginson y es código abierto
- Hay varias implementaciones disponibles
- Ambos parte de JAXP (Java API for XML Processing)
- JAXP se incluye como paquete a partir de Java 1.4
  - JAXP disponible de forma separada para Java 1.3

SAX

- **SAX funciona mediante callbacks: nuestro programa llama a un parser, y éste nos llama a nuestros callbacks**



API SAX

- **Se basa en 2 clases principales - un parseador y un manejador:**
  - **SAXParser**
  - **org.xml.sax.helpers.DefaultHandler**
    - ✓ **startDocument()**
    - ✓ **endDocument()**
    - ✓ **startElement(String uri, String lname, String qname, Attributes atts)**
    - ✓ **endElement(String uri, String lname, String qname)**
    - ✓ **characters(char text[], int start, int length)**

API SAX

- **startElement()**
  - **4 parámetros:**
    - ✓ **String uri = El "namespace URI" (Uniform Resource Identifier)**
    - ✓ **String lname = el "local name" del elemento**
    - ✓ **String qname = el "qualified name" del elemento**
    - ✓ **Attributes atts = lista de atributos para este elemento**
- **characters()**
  - **3 parámetros:**
    - ✓ **char text[] = array de caracteres que contiene el documento XML**
    - ✓ **int start = índice inicial de los datos actuales dentro de text[]**
    - ✓ **int length = índice final en text[]**

Armazón aplicación SAX

```
import javax.xml.parsers.*; // for both SAX and DOM
import org.xml.sax.*;
import org.xml.sax.helpers.*;


// For simplicity, we let the operating system handle exceptions
// In "real life" this is poor programming practice
public class Sample {
    public static void main(String args[]) throws Exception {
        // Create a parser factory
        SAXParserFactory factory = SAXParserFactory.newInstance();
        // Tell factory that the parser must understand namespaces
        factory.setNamespaceAware(true);
        // Make the parser
         SAXParser saxParser = factory.newSAXParser();
// Create a handler
         Handler handler = new Handler();
         // Tell the parser to use this handler
         parser.setContentHandler(handler);
         // Finally, read and parse the document
         parser.parse("hello.xml");
    } // end of Sample class
```

Ejemplo SAX

```
import java.io.*;
import org.xml.sax.*;
import org.xml.sax.helpers.*;
import org.apache.xerces.parsers.SAXParser;

public class Flour extends DefaultHandler {

  float amount = 0;

  public void startElement(String namespaceURI, String localName,
                           String qName, Attributes atts) {
    if (namespaceURI.equals("http://recipes.org") && localName.equals("ingredient")) {
       String n = atts.getValue("","name");
       if (n.equals("flour")) {
         String a = atts.getValue("","amount"); // assume 'amount' exists
         amount = amount + Float.valueOf(a).floatValue();
       }
    }
  }

  public static void main(String[] args) {
    Flour f = new Flour();
    SAXParser p = new SAXParser();
    p.setContentHandler(f);
    try { p.parse(args[0]); }
    catch (Exception e) {e.printStackTrace();}
    System.out.println(f.amount);
  }
}
```

Fichero XML a parsear del ejemplo anterior

```
<collection xmlns="http://recipes.org" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://recipes.org recipes.xsd">
<description>Some recipes used in the XML tutorial.</description>
<recipe>
<title>Beef Parmesan with Garlic Angel Hair Pasta</title>
<ingredient name="beef cube steak" amount="1.5" unit="pound"/>
<ingredient name="onion, sliced into thin rings" amount="1"/>
<ingredient name="green bell pepper, sliced in rings" amount="1"/>
<ingredient name="Italian seasoned bread crumbs" amount="1" unit="cup"/>
<ingredient name=“flour" amount=“20"/>
...
```

Codificar un programa para parsear el fichero XML y copiar la información a una instancia de la clase ServiceData

```
<?xml version="1.0"?>
<service>
  <inbound>
     <availableFrom>10</availableFrom>
     <availableUntil>22</availableUntil>
     <bannedUser>sip:jones@domain.org</bannedUser>
     <bannedUser>sip:james@dom.org</bannedUser>
  </inbound>
  <outgoing>
     <availableFrom>10</availableFrom>
     <availableUntil>22</availableUntil>
     <bannedUser>sip:jones@domain.org</bannedUser>
  </outgoing>
</service>
```

```
public class ServiceData {
      private int in-av-from;
      private    int in-av-to;
      private    Vector in-banned;
      private    int out-av-from;
      private    int out-av-to;
      private    Vector out-banned;

      public void ServiceData(int in-av-from, int in-av-to, Vector in-
      banned, int out-av-from, int out-av-to, Vector out-banned) {…};

      public int getIn-av-from() {…};
      public void setIn-av-from(int in-av-from) {…};
      public     int getIn-av-to() {…};
      public     void setIn-av-to(int in-av-to) {…};
      public     Vector getIn-banned() {…};
      public     void setIn-baned(Vector in-banned) {…};
      public     int getOut-av-from() {…};
      public void setOn-av-from(int out-av-from) {…};
      public     int getOut-av-to() {…};
      public     void setOut-av-to(int out-av-to) {…};
      public     Vector getOut-banned() {…};
      public     void setOut-banned(Vector out-banned) {…};

}
```

Codificar un programa para parsear el fichero XML y copiar la información a una instancia de la clase ServiceData

```
public class myParser extends DefaultHandler {

  ServiceData sd = new ServiceData();
  private String elemento;
  private boolean in_out;

  public void startElement(String namespaceURI, String localName,
       String qName, Attribues atts) {

       elemento = localName;

       if (localName.equals("inbound"))
         in_out = true;

       if (localName.equals("outgoing"))
         in_out = false;
  }

public void characters(char ch[], int start, int length)
   throws SAXException {

       String valor;
       Vector v;

  if(elemento.equals("availableFrom")) {
              valor = new String(ch, start, length);
              if (in_out)
                sd.setIn-av-from(Integer.parseInt(valor))
              else
}   sd.setOut-av-from(Integer.parseInt(valor));
  else
    if(elemento.equals("availableUntil")) {
              valor = new String(ch, start, length);
              if (in_out)
                sd.setIn-av-to(Integer.parseInt(valor))
              else
    sd.setOut-av-to(Integer.parseInt(valor));
  }
    else
              if(elemento.equals("bannedUser")) {
              valor = new String(ch, start, length);
              if (in_out) {
                v = sd.getIn-banned();
                v.add(valor);
                sd.setIn-banned(v))
              }
              else {
                  v = sd.getOut-banned();
                  v.add(valor);
      sd.setOut-banned(new Vector(Integer.parseInt(valor)));
              }
  }  }  }
```

Codificar un programa para parsear el fichero XML y copiar la información a una instancia de la clase ServiceData

```
public static void main(String[] args) {

   myParser m = new myParser();
   SAXParser p = new SAXParser();
   p.setContentHandler(m);
   try{
         p.parse(args[0]);

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

}
```