Práctica 3: Parser SAX
Clase de apoyo

Ejemplo funcionalidad P3

catalogo.xsd (I)

```
<xsd:element name="catalog">
     <xsd:complexType>
          <xsd:sequence>
               <xsd:element ref="tns:concepts" />
               <xsd:element ref="tns:datasets" />
               </xsd:sequence>
          </xsd:complexType>
……
</xsd:element>
……
<xsd:element name="concepts">
     <xsd:complexType>
          <xsd:sequence>
               <xsd:element ref="tns:concept" maxOccurs="unbounded"/>
          </xsd:sequence>
     </xsd:complexType>
……
</xsd:element>
……
<xsd:element name="datasets">
     <xsd:complexType>
          <xsd:sequence>
               <xsd:element ref="tns:dataset" maxOccurs= "unbounded"/>
          </xsd:sequence>
     </xsd:complexType>
……
</xsd:element>
```

catalogo.xsd (II)

```
<xsd:complexType name="tConcept">
     <xsd:sequence>
          <xsd:element name="code" type="xsd:string"/>
          <xsd:element name="label" type="xsd:string"/>
          <xsd:element ref="tns:concepts" minOccurs="0"/>
     </xsd:sequence>
     <xsd:attribute name="id" type="xsd:anyURI"/>
</xsd:complexType>
……
<xsd:complexType name="tDataset">
     <xsd:sequence>
          <xsd:element name="title" type="xsd:string" />
          <xsd:element name="description" type="xsd:string" minOccurs="0" />
          <xsd:element name="keyword" type="xsd:string" minOccurs="0"/>
          <xsd:element name="theme" type="xsd:string" minOccurs="0"/>
          <xsd:element name="publisher" type="xsd:string" minOccurs="0"/>
          <xsd:element name="concepts" minOccurs="0">
               <xsd:complexType>
                    <xsd:sequence>
                         <xsd:element name="concept" maxOccurs="unbounded">
                              <xsd:complexType>
                                   <xsd:attribute name="id" type="xsd:anyURI"/>
                              </xsd:complexType>
                         </xsd:element>
                    </xsd:sequence>
               </xsd:complexType>
          </xsd:element>
     </xsd:sequence>
     <xsd:attribute name="id" type ="xsd:anyURI"/>
</xsd:complexType>
```

Programa principal

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

/* Código a completar:
```

- Validar los argumentos recibidos en main()
- Instanciar un objeto ManejadorXML pasando como parámetro el código de la categoría recibido en el segundo argumento de main()
- Instanciar un objeto SAXParser e invocar a su método parse() pasando como parámetro un descriptor de fichero, cuyo nombre se recibió en el primer argumento de main(), y la instancia del objeto ManejadorXML
- Invocar al método getConcepts() del objeto ManejadorXML para obtener un List<String> con las uris de los elementos <concept> cuyo elemento <code> contiene el código de la categoría buscado
- Invocar al método getLabel() del objeto ManejadorXML para obtener el nombre de la categoría buscada
- Invocar al método getDatasets() del objeto ManejadorXML para obtener un mapa con los datasets de la categoría buscada
- Crear el fichero de salida con el nombre recibido en el tercer argumento de main()
- Volcar al fichero de salida los datos en el formato XML especificado por ResultadosBusquedaP3.xsd

```
 */

System.exit(0);
}
```

Interface ParserCatalogo

```
//=========================================================
// Interface ParserCatalogo
//=========================================================
Interface ParserCatalogo {


  String getLabel();  //Devuelve el nombre de la categoría buscada
                        Ej: "Danza y baile"

  List<String> getConcepts(); // Devuelve una lista con las uris
                                 de los concepts de la categoría
                                 buscada


  Map<String, HashMap<String, String>> getDatasets();
                     // Devuelve un mapa con la información de
                       los datasets que están asociados a la
                       categoría buscada
}
```

XMLCatalog Parser

```
public class ManejadorXML extends DefaultHandler
                                  implements ParserCatalogo {
   private String sNombreCategoria;
   private String sCodigoConcepto;
   private List <String> lConcepts;
   private Map <String, HashMap<String,String>> hDatasets;
   private StringBuilder contenidoElemento;
   ……


   public ManejadorXML(String sCodigoConcepto)
            throws SAXException, ParserConfigurationException {
       this.sCodigoConcepto = sCodigoConcepto;


   ……
   }
```

XMLCatalog Parser

```
@Override
public String getLabel(){
  return sNombreCategoria;
}

@Override
public  List <String> getConcepts(){
    return lConcepts;
}

@Override
public HashMap<String, HashMap<String,String> > getDatasets(){
    return hDatasets;
}
……
```

ManejadorXML

```
//=========================================================
// Métodos a implementar de DocumentHandler
//=========================================================
 public final void startDocument() throws SAXException{…}

 public final void endDocument() throws SAXException{…}

 public final void startElement(String ns, String name,
           String qname, Attributes attrs) throws SAXException {…}

 public final void endElement(String ns,String name,
            String qname) throws SAXException {
      …
      contenidoElemento.setLength(0);
 }
```

ManejadorXML

```
public final void characters(char chars[], int start, int len)
 throws SAXException {
       super.characters(chars, start, len);
       contenidoElemento.append(chars,start,len);
}
```

En este método **NO** se debe gestionar la extracción del valor de cadena de los elementos de información a transformar.
No se puede suponer que se invoca una única vez por cada contenido de elemento.

Ejemplo:

<nombre> Juán Castilla </nombre>

Catálogo xml. Extracto con un concept

```
<concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile">
   <code>0016-018</code>
   <label><![CDATA[Danza y baile]]></label>
      <concepts>
         <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/ClasicaEspanola">
            <code>0001-0016FL18</code>
            <label><![CDATA[Clásica Española]]></label>
         </concept>
          <concept id="https://datos.madrid.es/egob/kos/.../ContemporaneaBreakdanceHipHop">
              <code>0002-0016FL18</code>
              <label><![CDATA[Contemporánea Breakdance Hip-Hop]]></label>
          </concept>
          <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/Flamenco">
              <code>0003-0016FL18</code>
              <label><![CDATA[Flamenco]]></label>
          </concept>
          <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/FolcloreEtnica">
              <code>0004-0016FL18</code>
              <label><![CDATA[Folclore Étnica]]></label>
          </concept>
          <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/SalonTango">
              <code>0005-0016FL18</code>
              <label><![CDATA[Salón Tango]]></label>
          </concept>
      </concepts>
</concept>
```

ManejadorXML(). Recoger los id de los concept

Lo que deben hacer los métodos del ContentHandler es:

- **Paso 1:** Detectar cuando llega un evento *startElement* del elemento <concept> para guardar temporalmente el valor del atributo id y anotar que se ha entrado en un <concept>
- **Paso 2:** esperar la llegada de un evento *endElement* del elemento </code>. Si se está dentro de un <concept> y el contenido es el código de la categoría buscada, almacenar el valor del atributo id guardado en el paso 1 en el ArrayList lConcepts. Anotar que se ha encontrado la categoría y es del primer nivel
- **Paso 3:** esperar la llegada del evento *endElement* de </label>. Si se está dentro de un <concept>, se ha encontrado la categoría y se está en el primer nivel, almacenar el contenido del elemento en el atributo sNombreCategoria
- **Paso 4:** para poder recoger todas las subcategorías, mientras está abierto el <concept> correspondiente a la categoría buscada, cuando lleguen eventos *startElement* de <concept> , se deberán obtener los atributo id de los <concept> y almacenarlos también en lConcepts. Además habrá que ir incrementando una variable que indique el nivel y decrementarla cuando se salga del <concept> correspondiente hasta llegar a 0, que indicará que se ha salido del <concept> raíz.

Se deben usar variables *boolenas* para saber en qué punto de la jerarquía del documento se encuentra el elemento que se acaba de abrir o cerrar

ManejadorXML(). Recoger los id de los concept

Catálogo XML. Extracto con los datasets

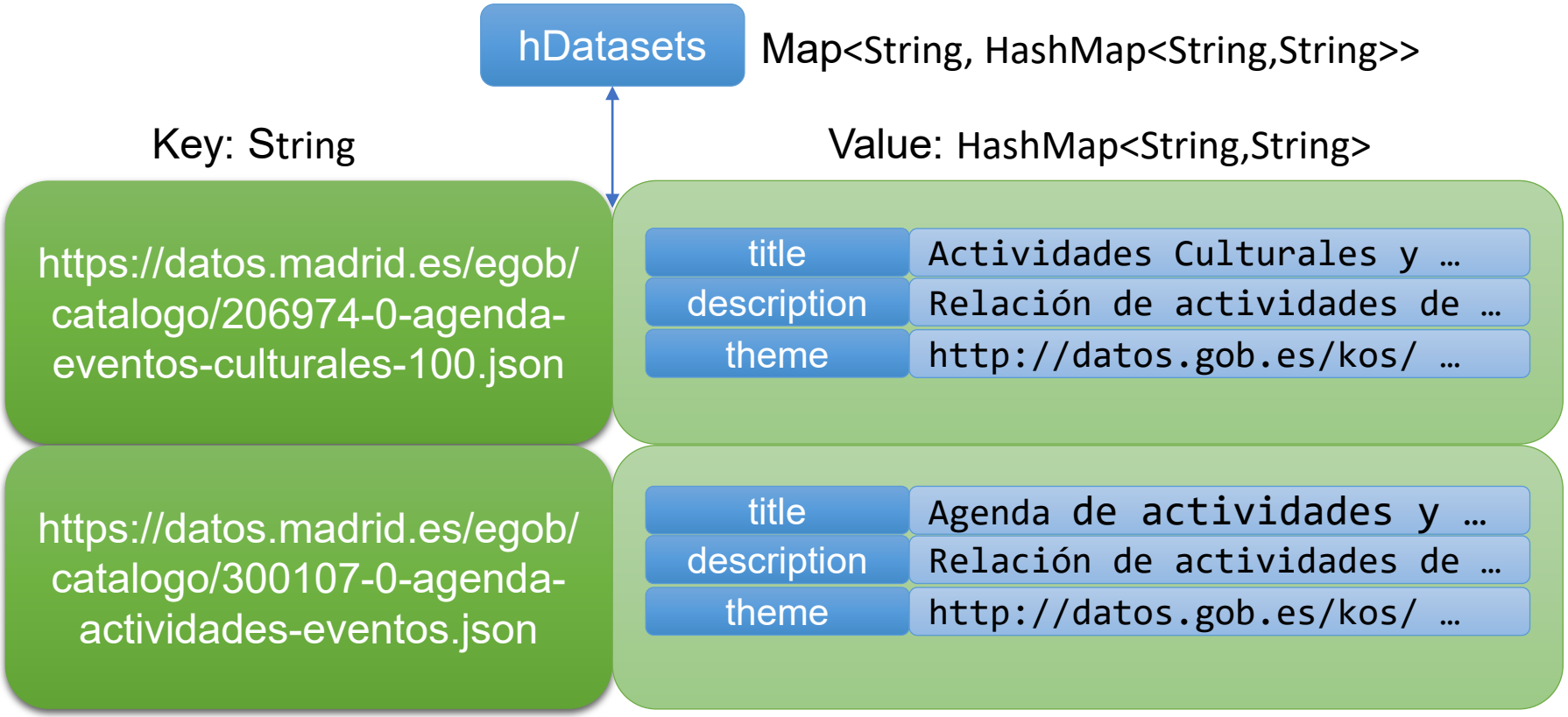
```
<datasets>
   <dataset id="https://datos.madrid.es/egob/catalogo/206974-0-agenda-eventos-culturales-100.json">
      <title><![CDATA[Actividades Culturales y de Ocio Municipal en los próximos 100 días]]></title>
      <description><![CDATA[Relación de actividades de carácter cultural y de ocio …… los datos.]]></description>
      <keyword><![CDATA[Cultura y Ocio]]></keyword>
      <theme><![CDATA[http://datos.gob.es/kos/sector-publico/sector/cultura-ocio]]></theme>
      <publisher><![CDATA[http://datos.gob.es/recurso/sector-publico/org/Organismo/L01280796]]></publisher>
      <concepts>
         <concept id="https://datos.madrid.es/egob/kos/actividades/ConferenciasColoquios" />
         ……
         <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/FolcloreEtnica" />
         <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile" />
         <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/Flamenco" />
         ……
      </concepts>
   </dataset>
   <dataset id="https://datos.madrid.es/egob/catalogo/300107-0-agenda-actividades-eventos.json">
      <title><![CDATA[Agenda de actividades y eventos]]></title>
      <description><![CDATA[Relación de actividades de distinto tipo …… comprender mejor los datos.]]></description>
      <keyword><![CDATA[Cultura y Ocio]]></keyword>
      <theme><![CDATA[http://datos.gob.es/kos/sector-publico/sector/cultura-ocio]]></theme>
      <publisher><![CDATA[http://datos.gob.es/recurso/sector-publico/org/Organismo/L01280796]]></publisher>
      <concepts>
         <concept id="https://datos.madrid.es/egob/kos/actividades/ConferenciasColoquios" />
         ……
         <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/FolcloreEtnica" />
         <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile" />
         <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/Flamenco" />
         ……
      </concepts>
   </dataset>
</datasets>
```

ManejadorXML(). Recoger los dataset

- Cuando llegue un evento *starElement* <datasets>, llegue el evento *startElement* <dataset>, llegue el evento *startElement* <concepts> y llegue el evento *startElement* <concept> donde su atributo id coincida con alguno de las url almacenadas en el ArrayList lConcepts, se almacenará en hDatasets el id del dataset y los elementos <title>, <descrition> y <theme>.

- Hay que tener en cuenta que antes del poder conocer en el <dataset> existen <concept> cuyo id puede ser relevante, habrá que guardar en variables temporales el id del elemento <dataset>, y los elementos <title>, <description> y <theme>

ManejadorXML(). Recoger los dataset



Documento xml de salida

- Una vez almacenada la información pertinente de catalogo.xml en las estructuras de datos del programa, se ha de volcar a un fichero xml cuyo XMLSchema está descrito en searchResultP3.xsd.

- Se recomienda implementar una clase que se encargue de dicha misión.

Documento XML de salida

```
<?xml version="1.0" encoding="UTF-8"?>
<searchResults xmlns="http://www.piat.dte.upm.es/ResultadosBusquedaP3"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.piat.dte.upm.es/ResultadosBusquedaP3 ResultadosBusquedaP3.xsd">
      <summary>
          <query>0016-018</query>
          <numConcepts>6</numConcepts>
          <numDatasets>2</numDatasets>
       </summary>
       <results>
            <concepts>
              <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile"/>
              <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/ClasicaEspanola"/>
              <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/
                           ContemporaneaBreakdabceHipHop"/>
              <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/Flamenco"/>
              <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/FolcloreEtnica"/>
              <concept id="https://datos.madrid.es/egob/kos/actividades/DanzaBaile/SalonTango"/>
            </concepts>
```

Documento xml de salida

```
    <datasets>
      <dataset id="https://datos.madrid.es/egob/catalogo/206974-0-agenda-eventos-culturales
                  -100.json">
        <title>Actividades Culturales y de Ocio Municipal en los próximos 100 días</title>
        <description>Relación de actividades de carácter cultural y de ocio, que …</description>
        <theme>http://datos.gob.es/kos/sector-publico/sector/cultura-ocio</theme>
      </dataset>
      <dataset id="https://datos.madrid.es/egob/catalogo/300107-0-agenda-actividades-eventos.json">
        <title>Agenda de actividades y eventos</title>
        <description>Relación de actividades de distinto tipo que se van a celebrar …. </description>
        <theme>http://datos.gob.es/kos/sector-publico/sector/cultura-ocio</theme>
      </dataset>
    </datasets>
  </results>
</searchResults>
```

Documento xml de salida

```
//Sugerencia de una posible forma de generar el fichero xml la salida

public class GenerarXML {
     ……
     private static final String conceptPattern= "\n\t\t\t<concept id=\"#ID#\"/>" ;
     ……

     private static String conceptsToXML (List <String> lConcepts){
          StringBuilder sbSalida = new StringBuilder();
          sbSalida.append("\n\t\t<concepts>" );

          for (String unConcepto : lConcepts){
               sbSalida.append (conceptPattern.replace("#ID#", unConcepto));
          }

          sbSalida.append("\n\t\t</concepts>");
          return sSalida sbSalida.toString();
     }

}
```