# LABORATORIO DE PROCESAMIENTO DE INFORMACIÓN EN APLICACIONES TELEMÁTICAS

## Práctica 3.

## Curso 2021/2022

Versión 6/04/2022

# Contenido

# 1 Objetivos

El objetivo principal de esta práctica es la familiarización con la tecnología **SAX** y el desarrollo de un analizador de documentos XML basado en dicha tecnología.

Como objetivo secundario se pretende capacitar en el diseño de algoritmos eficientes que permitan la extracción y transformación de información a partir de fuentes de contenidos estructurados.

# 2 Descripción

Se desea desarrollar una aplicación que extraiga cierta información de un fichero XML obtenido a partir de los datos publicados en el Portal de Datos Abiertos del Ayuntamiento de Madrid (http://datos.madrid.es).

La información (organismos, eventos, actividades, …) publicada a través del Portal de Datos Abiertos presenta las siguientes características:

- Los elementos de información, en adelante recursos (`concept`), se identifican mediante una URI.
- Cada recurso se encuentra asociado a una categoría (elemento `code` de `concept`).
- Los recursos se encuentran agrupados en conjuntos de datos (elemento `dataset`) accesibles en formato JSON a través de un URI indicada en el atributo `id` del elemento `dataset`.
- En un `dataset` puede haber información sobre recursos asociados a varias categorías.
- Los recursos de una categoría pueden estar accesibles a través de diferentes `datasets`.
- Para la categorización de los recursos se utiliza un sistema de clasificación jerárquico basando en características temáticas.

Esta información se encuentra descrita en un Catálogo de Datos en formato XML (`catalogo.xml`) válido con respecto al esquema **`catalogo.xsd`**.

La aplicación a desarrollar proporcionará una herramienta de búsqueda que posibilite la recuperación de recursos asociados a un código de categoría generando un documento XML con los resultados.

La Figura 1 muestra un ejemplo de presentación de parte de la estructura jerárquica de los `concepts` del catálogo.

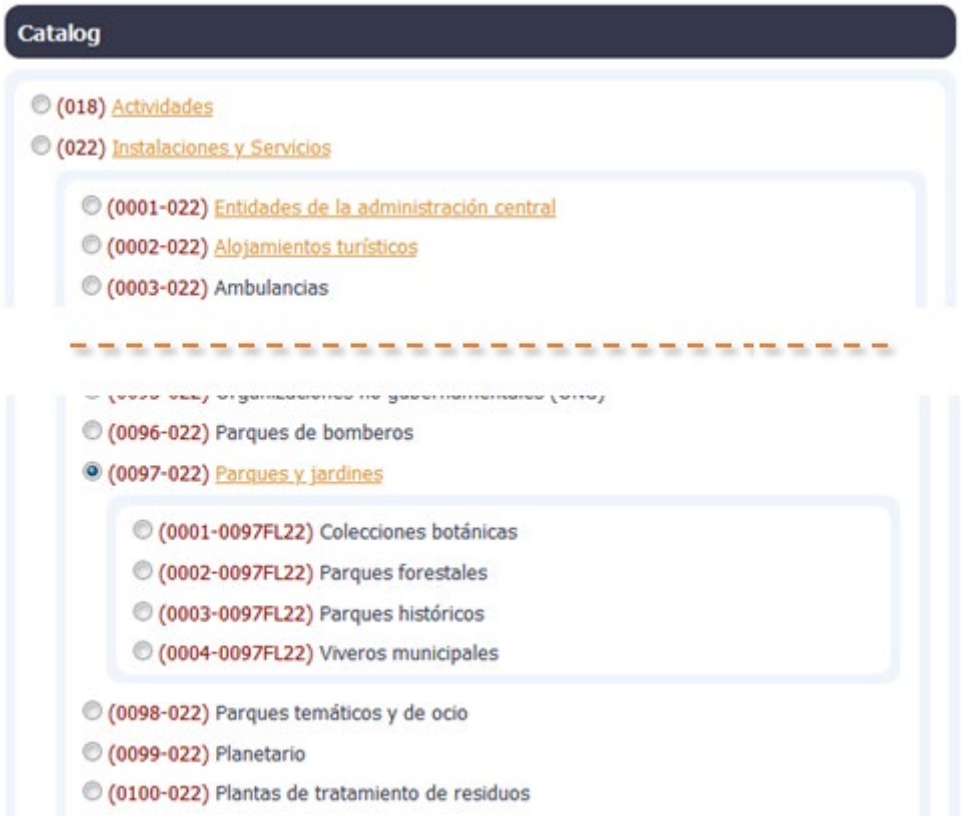


Figura 1.- Representación de la estructura jerárquica de los ***concepts*** del **catálogo de datos**

La aplicación a desarrollar recibirá como argumento el criterio de búsqueda, esto es, el código de la categoría de la que se desea información, y proporcionará información sobre los `concepts` y `datasets` pertinentes, aplicando para ello los siguientes criterios:

- Se considerarán pertinentes el `concept` cuyo código (elemento `code`) coincida con el criterio de búsqueda y todos los `concepts` descendientes del mismo.
- Se considerarán pertinentes los `dataset` que contengan información asociada a alguno de los `concept` pertinentes (contengan un elemento `concept` con el atributo `id` igual al atributo `id` del elemento `concept` pertinente).

La Figura 2 muestra un ejemplo de búsqueda del `concept` con código 0097-022 y los resultados que se obtendrían.


Figura 2.- ***Concepts*** y ***datasets*** pertinentes para el código 0097-022.

# 3 Realización

## 3.1 Especificaciones de la aplicación

La aplicación (`P3_SAX`) deberá recibir los siguientes argumentos: [1]

- `(ARG0) Ruta al documento` **`catalogo.xml`**.
- `(ARG1)` Criterio de búsqueda, expresado por el `código de la categoría` de la que se desea información.
- `(ARG2) Ruta al documento XML de salida` en el que se almacenará el resultado de la búsqueda.

La aplicación deberá realizar las siguientes acciones:

1. Verificación y validación de los argumentos de entrada.
2. Extracción de información del documento XML de entrada.
3. Transformación de información y generación del documento de resultados.

A continuación, se detallan estos pasos.

[1] En el código de la aplicación no puede haber referencias al sistema de ficheros local donde se encuentran los ficheros XML. La aplicación debe por tanto poder ejecutarse en cualquier ordenador pasando los parámetros oportunos que indican el camino (*path*) de los ficheros.

### 3.1.1 Verificación y validación de los argumentos de entrada.

Si el número de argumentos no es correcto, o los argumentos no toman valores válidos con respecto al tipo de información esperada, la aplicación deberá finalizar indicando la causa.

En concreto se deberá verificar:

- Que el número de argumentos es correcto.
- Que los argumentos se corresponden con el tipo de información que se espera de ellos, realizándolo mediante expresiones regulares:
  - Los argumentos `ARG0` y `ARG2` deben finalizar con los caracteres ".xml".
  - El argumento `ARG1` debe empezar por 3 o 4 caracteres numéricos, seguidos opcionalmente de un guion y a continuación de 3 a 8 caracteres, pudiendo ser estos últimos números y/o caracteres alfanuméricos en mayúscula. Ejemplos válidos: "018", "0001-018", "0001-0003FL18".
- Que `ARG0` se corresponde con el `path` de un fichero al que se tiene permiso de acceso de lectura.
- Que `ARG2` se corresponde con el `path` de un fichero del que se tiene permiso de escritura.

No será necesario realizar ningún tipo de validación sobre el contenido del documento `ARG0`.

### 3.1.2 Extracción de información del documento XML de entrada

Mediante un analizador basado en el modelo de objetos SAX XML, implementado en la clase **`ManejadorXML`**, se deberá analizar el documento XML pasado como argumento (`ARG0`) y se deberán obtener los `concepts` y `datasets` pertinentes.

La clase **`ManejadorXML`** deberá implementar la interfaz **`ParserCatalogo`**.

### 3.1.3 Transformación de información y generación del documento de resultados

Se deberá generar un documento XML (con el nombre indicado en `ARG2`) válido con respecto a `ResultadosBusquedaP3.xsd`, a partir de la serialización a XML de la información extraída del documento XML de entrada.

La serialización a XML deberá hacerse directamente mediante la correcta gestión de objetos de tipo **`StringBuilder`**. No está permitido la utilización de APIs auxiliares que posibiliten la serialización mediante la instanciación de objetos, ni de soluciones equivalentes. Puede usar las colecciones que estime oportunas para almacenar la información que va a serializar.

## 3.2 Restricciones de implementación

- La aplicación deberá implementarse mediante, al menos, las siguientes clases `java` pertenecientes al paquete **`piat.opendatasearch`**:
  - `P3_SAX`: clase inicial de la aplicación que contendrá el método estático `main()` que se encargará de:

        - La validación de los argumentos de entrada.
        - Realizar el análisis y extracción de la información pertinente del catálogo.
        - Crear el documento XML de salida.

      Esta clase no se debe instanciar.

    - `ParserCatalogo`: interfaz que debe implementar la clase `ManejadorXML`.
    - `ManejadorXML`: analizador SAX. Debe usar colecciones para almacenar la información pertinente al criterio de búsqueda durante el análisis de cada elemento del documento XML. No serán válidas soluciones basadas en tomas de decisiones una vez analizado el contenido de todo el documento, es decir, no se pueden utilizar colecciones auxiliares para el almacenamiento de los objetos procedentes de los eventos, para posteriormente analizarlos. El resultado del análisis, almacenado en colecciones, estará accesible a través de los métodos públicos definidos en la interfaz `ParserCatalogo` que implementa y que se invocarán desde la clase `P3_SAX`.

- La aplicación no deberá implementar ninguna herramienta que explícitamente determine la validez del documento generado. La validez del documento XML de salida la deberá proporcionar la robustez de los algoritmos diseñados, serializando únicamente información válida con respecto al documento de salida. No obstante, puede validar el documento XML generado mediante una herramienta externa como hizo en la práctica 2.
- Todas las clases java y documentos implicados deberán estar codificados en **`UTF-8`**.
- En todos los ficheros JAVA debe estar escrito, antes de la definición de la clase, el nombre del alumno en formato `Javadoc` (dentro del *tag* `@author`).

# 4 Material disponible

En Moodle encontrará el siguiente material para la realización de la práctica:

- Esqueleto de las clases a codificar: `P3_SAX` y `ManejadorXML`. Su diseño se basa en el uso de las colecciones del documento *Practica 3. Clase de apoyo*.
- Interface `ParserCatalogo` a implementar en `ManejadorXML` y su Javadoc.
- Documento XML `catalogo.xml` con información obtenida del Catálogo de Datos del Portal y su esquema `catalogo.xsd`.
- Esquema del documento XML que debe generar la aplicación como resultado de la búsqueda: `ResultadosBusquedaP3.xsd`.

| | |
|---|---|
| **IMPORTANTE**: | Puesto que las clases java que se le proporcionan están codificadas en **UTF-8**, asegúrese que el proyecto de eclipse lo ha configurado para que use esta codificación antes de importar los ficheros java que se le proporcionan. Observará que lo ha hecho correctamente si puede ver las tildes de los comentarios |

# 5 Entrega

Con anterioridad al 28 de abril a las 10:30 horas, deberá entregarse en Moodle, dentro de `Espacio para la entrega de la práctica 3`, los siguientes ficheros (sin comprimir):

- Los ficheros Java desarrollados con el nombre del alumno en formato Javadoc (dentro del tag @author) antes de la definición de la clase.
- Un fichero `XML` con la salida de la aplicación al ejecutarla con el código de catálogo 018.