

## INTRODUCCIÓ

Fins ara hem fet servir les eines que té eXistDB per fer les nostres consultes, ara toca poder interactuar des d'una aplicació amb Java amb aquest SGBD.

Per poder realitzar aquesta feina disposem de l'API **XMLDB** on bàsicament farem servir els següents 3 paquets:









- **org.xmldb.api**: Interfaces, DatabaseManager
- **org.xmldb.api.base**: Interfaces, Collection, Configurable, Database, Resource, ResourceIterator, ResourceSet, Service, Classes, ErrorCodes, Exceptions, XMLDBException
- **org.xmldb.api.modules**: Interfaces, BinaryResource, CollectionManagementService, TransactionService, XMLResource, XPathQueryService, XUpdateQueryService

Així que, per poder realitzar el nostre projecte i crear col·leccions, pujar documents i fer consultes al nostre SGBD eXistDB, hem de fer els següents passos:

1. Connexió amb eXistDB
2. Manegament de col·leccions i documents xml
3. Realització de consultes o modificacions als nostre xml

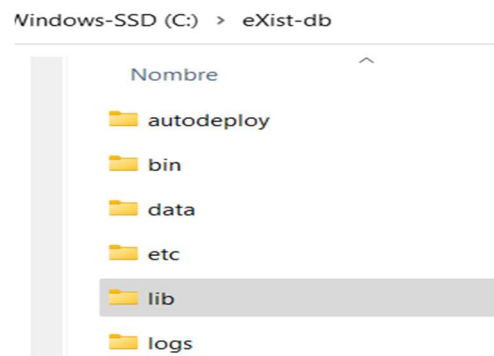
### 1. CONNEXIÓ AMB eXist-DB

- a) Per aconseguir establir des del nostre programa en Java una connexió amb el SGBD eXistDB necessitem les següents llibreries:

-  exist.jar
-  exist-optional.jar
-  log4j-1.2.15.jar
-  ws-commons-util-1.0.2.jar
-  xml-apis-1.3.04.jar
-  xmlldb.jar
-  xmlrpc-client-3.1.1.jar
-  xmlrpc-common-3.1.jar

Les hem d'incloure al nostre projecte java en Eclipse per tal de poder fer servir les classes i mètodes que ens permetran establir la connexió amb eXistDB i poder manegar la nostra BD de documents XML.

També es necessari que aquestes llibreries estiguin copiades en la carpeta de llibreries de la ruta on tenim instal·lat eXistDB.



**b)** Hem de conèixer el driver necessari, que en aquest cas serà:

`"org.exist.xmldb.DatabaseImpl"` o `"org.apache.xindice.client.xmldb.DatabaseImpl"`

**c)** I per últim hem de conèixer la URL de la nostra BD a eXistDB.

## Codi font de connexió a la nostra col·lecció

```
import org.xmldb.api.*;
import org.xmldb.api.base.*;
import org.xmldb.api.modules.*;

public class ConsultaExistDB {

    public static void main(String[] args) throws XMLDBException {

        String driver = "org.exist.xmldb.DatabaseImpl"; //Driver para eXist
        Collection col = null; // Colección
        String URI="xml:db:exist://localhost:8080/exist/xmlrpc/db/La vostra col·lecció"; //URI
        String usu="admin"; //Usuario
        String usuPwd="EL vostre password"; //Clave

        try {
            Class cl = Class.forName(driver); //Cargar del driver
            Database database = (Database) cl.newInstance(); //Instancia de la BD
            DatabaseManager.registerDatabase(database); //Registro del driver

            } catch (Exception e) {
                System.out.println("Error al inicializar la BD eXist");
                e.printStackTrace(); }

        col = DatabaseManager.getCollection(URI, usu, usuPwd); //Carreguem la nostra col·lecció

        if(col == null)
            System.out.println(" *** LA COLECCION NO EXISTE. ***");

        XPathQueryService servicio = (XPathQueryService) col.getService("XPathQueryService", "1.0");

        //Ara farem i executarem la consulta xquery sobre el nostre fitxer
        ResourceSet result = servicio.query ("for $a in //title/text() return $a");

        // recorrer los datos del recurso.
        ResourceIterator i;
        i = result.getIterator();
        if (!i.hasMoreResources())
            System.out.println(" LA CONSULTA NO DEVUELVE NADA.");

        while (i.hasMoreResources()) {
            Resource r = i.nextResource();
            System.out.println((String) r.getContent());
        }
        col.close(); //Cerramos colección
    }
    // FIN
}
```

## 2- MANEGAMENT DE COL·LECCIONS

### OPERACIONS SOBRE COL·LECCIONS I DOCUMENTS

L'API **XMLDB** ens permetrà a més de consultar documents a la BD, crear i eliminar col·leccions, i crear i eliminar documents.

- **Crear una col·lecció:** per crear una col·lecció nova, utilitzarem el mètode **createCollection** del servei **CollectionManagementService**. El següent exemple crea la col·lecció NOVA\_COL·LECCIÓ dins de la col·lecció col:

```
CollectionManagementService mgtService =
    (CollectionManagementService)col.getService("CollectionManagementService","1.0");
mgtService.createCollection("NUEVA_COLECCION");
```

- **Esborrar una col·lecció:** per esborrar utilitzem el mètode **removeCollection**

```
mgtService = (CollectionManagementService)col.getService("CollectionManagementService","1.0");
mgtService.removeCollection("NUEVA_COLECCION");
```

- **Pujar un fitxer a la nostra col·lecció:** Una vegada tenim creada la nostra col·lecció per pujar documents xml haurem de fer:

```
File arxiu= new File("nomfitxer.xml");

if(!arxiu.canRead()) System.out.println("ERROR AL LLEIGIR ARXIU");
else{
    Resource nouRecurs = col.createResource(arxiu.getName(), "XMLResource");
    nouRecurs.setContent(arxiu); //Comprova que es un arxiu
    col.storeResource(nouRecurs); }
```

### 3. REALITZACIÓ DE CONSULTES O MODIFICACIONS

Per tal de poder modificar des de Java els nostres documents XML, farem servir algunes classes com **XPathQueryService**, **ResourceSet**, **ResourceIterator** i mètodes com per exemple **query**.

Una vegada estem connectats a la nostra col·lecció dins la BD i tenim pujat el nostre document ja podem fer operacions sobre ell.

El primer que haurem de fer es definir el nostre servei de consulta, que farem de la següent manera:

```
XPathQueryService servicio = (XPathQueryService) col.getService("XPathQueryService", "1.0");
```

Seguidament farem la nostra consulta:

```
ResourceSet result = servicio.query ("Aquí introduïrem la nostra expressió XQuery");
```

Si la nostra operació ha anat correctament podem mostrar el resultat de la següent manera:

```
// recorrer les dades del recurs.  
ResourceIterator i;  
i = result.getIterator();  
if (!i.hasMoreResources())  
    System.out.println(" LA CONSULTA NO RETORNA RES.");  
  
while (i.hasMoreResources()) {  
    Resource r = i.nextResource();  
    System.out.println((String) r.getContent());  
}
```

A continuació podeu veure alguns exemples de consulta que us poden servir de base per fer les vostres, encara que també podeu buscar informació a internet o manuals.

Els exemples estan fets sobre el document books.xml que ja hem fet servir per fer activitats.

### Consulta els títols dels llibres de la biblioteca

```
XPathQueryService servicio = (XPathQueryService) col.getService("XPathQueryService", "1.0");
//Ara farem i executarem la consulta xquery sobre el nostre fitxer
ResourceSet result = servicio.query ("for $a in //title/text() return $a");
```

Resultat:

```
Everyday Italian
Harry Potter
XQuery Kick Start
Learning XML
```

### Inserta un nou llibre a la biblioteca

```
ResourceSet result = servicio.query ("update insert"
+ "    <book category='FANTASY'>"
+ "    <title lang='es'>El señor de los anillos</title>"
+ "    <author>J.R.R Tolkien</author>"
+ "    <year>1954</year>"
+ "    <price>22</price>"
+ "    </book>"
+ "into doc('books.xml')//bookstore"
);
```

Resultat:

```
<book category="WEB">
  <title lang="en">Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
</book>
<book category="FANTASY">
  <title lang="es">El señor de los anillos</title>
  <author>J.R.R Tolkien</author>
  <year>1954</year>
  <price>22</price>
</book>
```

Mostrar el títol i l'autor dels llibres de l'any 2005 i etiquetar cadascun d'ells amb "lib2005"

```
ResourceSet result = servicio.query ("for $a in //book[year=2005]" +
" return <lib2005> {$a/title/text()} / {$a/author/text()}</lib2005>");
```

Resultat:

```
<lib2005>Everyday Italian / Giada De Laurentiis</lib2005>
<lib2005>Harry Potter / J K. Rowling</lib2005>
```