

La base de données XML eXist

```
<SPEECH>
<SPEAKER>HAMLET</SPEAKER>
<LINE>Rest, rest, perturbed spirit!</LINE>
<STAGEDIR>They enter</STAGEDIR>
<LINE>So, gentle...
<LINE>... all my... do...
<LINE>... and... a man... Hamlet...</LINE>
<LINE>... to... his... and... to you,</LINE>
<LINE>... God... should... not... together;</LINE>
<LINE>... all... your... faces... you... I...</LINE>
<LINE>The time is out of joint: O cursed spite,</LINE>
<LINE>That ever I was born to set it right!</LINE>
<LINE>Nay, come, let's go together.</LINE>
</SPEECH>
```

eXist

Introduction

◆ Qu'est-ce-que eXist?

- C'est une base de donnée native, entièrement écrite en Java
- XML n'est pas une base de données en soi
- Bien qu'il possède quelques caractéristiques
 - DTD, XML Schemas
 - des langages de requêtes -XQuery, XPath,
 - des interfaces de programmation : SAX, DOM, JDOM
- Il ne propose pas :
 - un stockage efficace, les index, la sécurité, les transactions et l'intégrité des données, l'accès multi-utilisateur

Introduction

◆ Les points forts de eXist

- Stockage intelligent de fichiers dans des **collections**
- Possibilités d'administration avancée (triggers, indexation, etc.)
- Peut-être interrogée à distance si elle est mise sur un serveur distant
- Plusieurs langages (Java, Perl, Php, etc.) peuvent utiliser eXist via des API

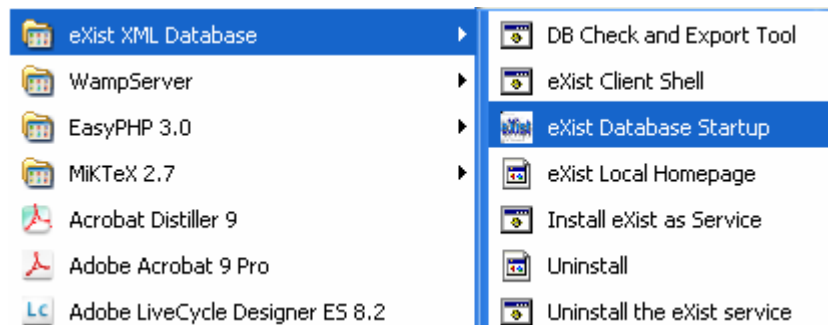
Téléchargement et installation

◆ Étapes

1. Télécharger la dernière version de eXist à cette URL:

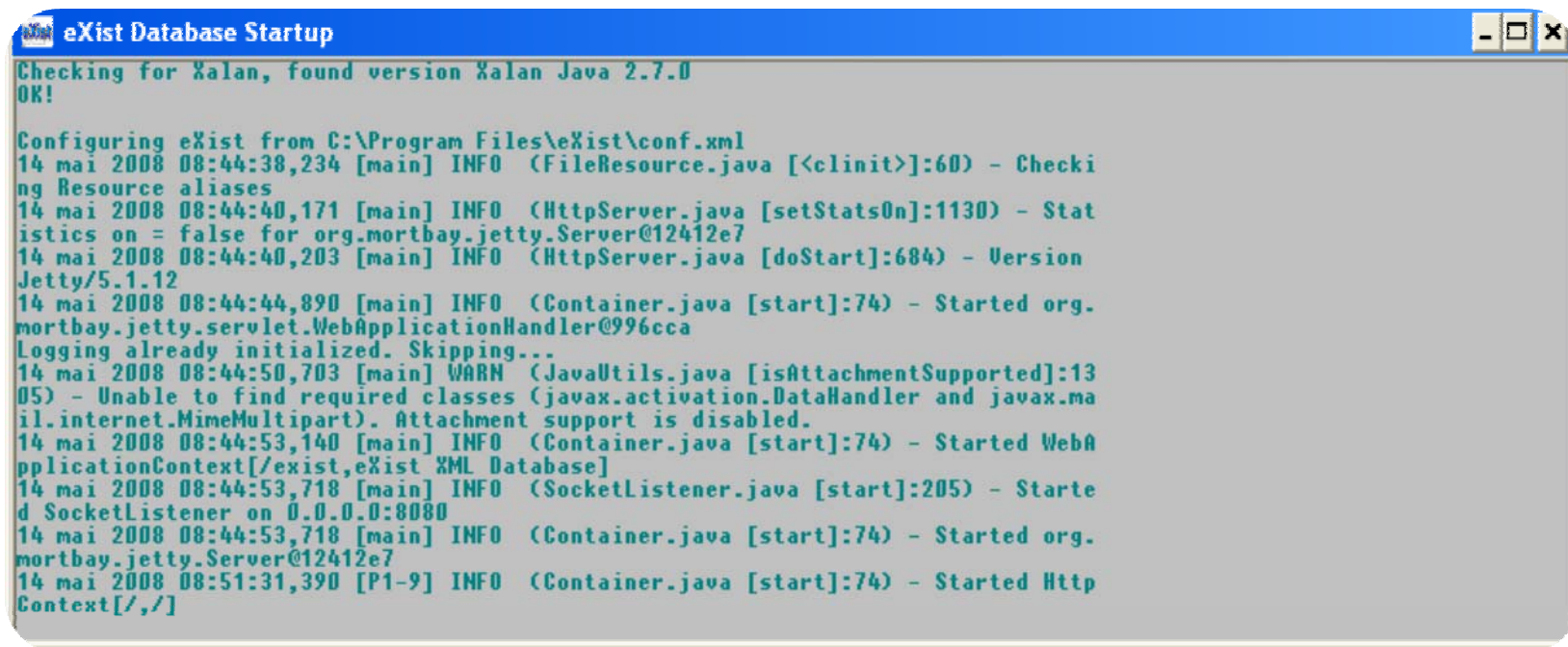
<http://exist-db.org/download.html> (current release)

2. Une fois le fichier eXist-[version]-build-XXXX.jar téléchargé, ouvrez-le, tout en vérifiant que la JDK 1.4.2 minimum est déjà installée sur votre machine
3. Maintenant que tous les outils permettent de faire fonctionner eXist ont été installés, lancez le fichier eXist Database Startup qui se trouve dans le menu démarrer :



Démarrage

- La console suivante doit apparaître, **ne la fermez pas** :



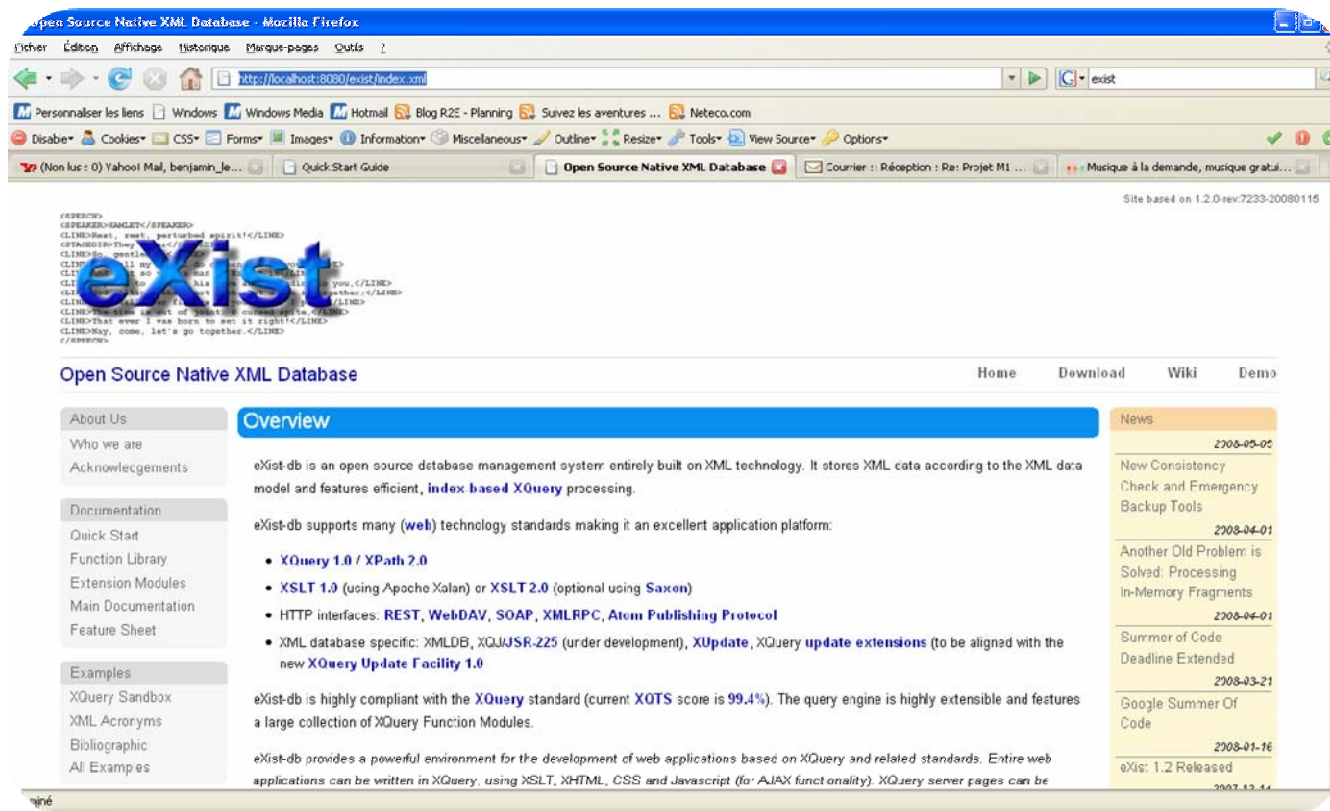
```
eXist Database Startup
Checking for Xalan, found version Xalan Java 2.7.0
OK!

Configuring eXist from C:\Program Files\eXist\conf.xml
14 mai 2008 08:44:38,234 [main] INFO (FileResource.java [<clinit>]:60) - Checki
ng Resource aliases
14 mai 2008 08:44:40,171 [main] INFO (HttpServer.java [setStatsOn]:1130) - Stat
istics on = false for org.mortbay.jetty.Server@12412e7
14 mai 2008 08:44:40,203 [main] INFO (HttpServer.java [doStart]:684) - Version
Jetty/5.1.12
14 mai 2008 08:44:44,890 [main] INFO (Container.java [start]:74) - Started org.
mortbay.jetty.servlet.WebApplicationHandler@996cca
Logging already initialized. Skipping...
14 mai 2008 08:44:50,703 [main] WARN (JavaUtils.java [isAttachmentSupported]:13
05) - Unable to find required classes (javax.activation.DataHandler and javax.ma
il.internet.MimeMultipart). Attachment support is disabled.
14 mai 2008 08:44:53,140 [main] INFO (Container.java [start]:74) - Started WebA
pplicationContext[/exist,eXist XML Database]
14 mai 2008 08:44:53,718 [main] INFO (SocketListener.java [start]:205) - Starte
d SocketListener on 0.0.0.0:8080
14 mai 2008 08:44:53,718 [main] INFO (Container.java [start]:74) - Started org.
mortbay.jetty.Server@12412e7
14 mai 2008 08:51:31,390 [P1-9] INFO (Container.java [start]:74) - Started Http
Context[/,/]
```

Démarrage

◆ Test

- Pour tester que le serveur a bien été lancé, allez à cette URL locale <http://localhost:8080/exist/index.xml>
- La page suivante doit apparaître :



Paramétrage

◆ Étapes

- Au niveau de l'administration, la page <http://localhost:8080/exist/admin/admin.xql> permet de configurer les utilisateurs, les collections et les statuts du système
- A la première ouverture, il vous sera demandé de créer un identifiant et un mot de passe. Il est important de les garder car ils sont utiles pour toute tâche d'administration. (vous pouvez aussi laisser vide le champ « mot de passe » et laisser « **admin** » en nom d'utilisateur, ce qui sera plus simple)

Exemple pour moi : identifiant : **admin** et pssword : **readteam**

- Avant de pouvoir créer des collections, il est intéressant d'en comprendre le fonctionnement grâce à des exemples présents à cette page : <http://localhost:8080/exist/examples.xml>

Ma première collection

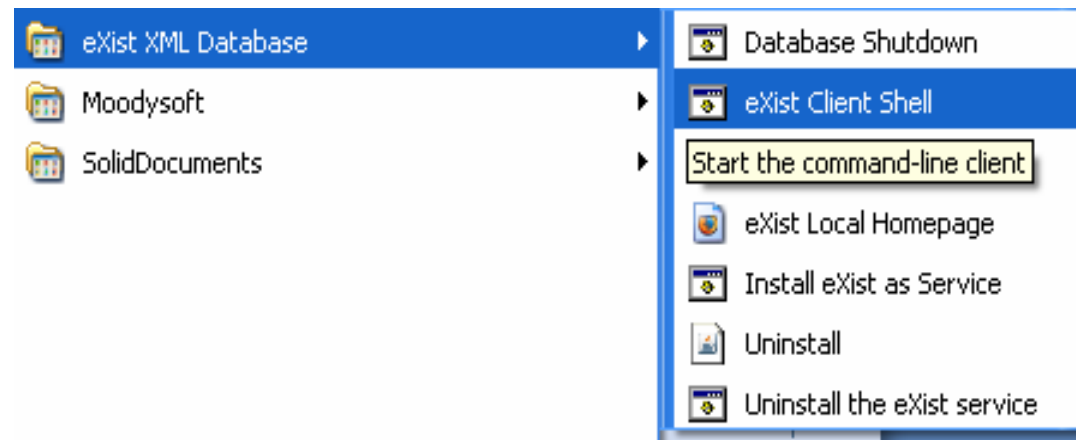
◆ Création

- Nous allons créer des collections de fichiers XML dans eXist, interroger les fichiers XML correspondant à ces collections (interroger la base de données)
- Il faut savoir que beaucoup d'API ont été développées pour accéder à eXist
- Pour les tests qui suivent, concernant l'utilisation d'eXist, nous allons
 - dans un premier temps, regarder le fonctionnement avec le **client eXist** (fenêtre avec interface graphique),
 - puis dans un second temps, nous utiliserons la « **sandbox** », qui se trouve dans les exemples du site de eXist
 - Enfin, nous verrons comment l'interfacer avec un langage de programmation comme **Java**

Ma première collection

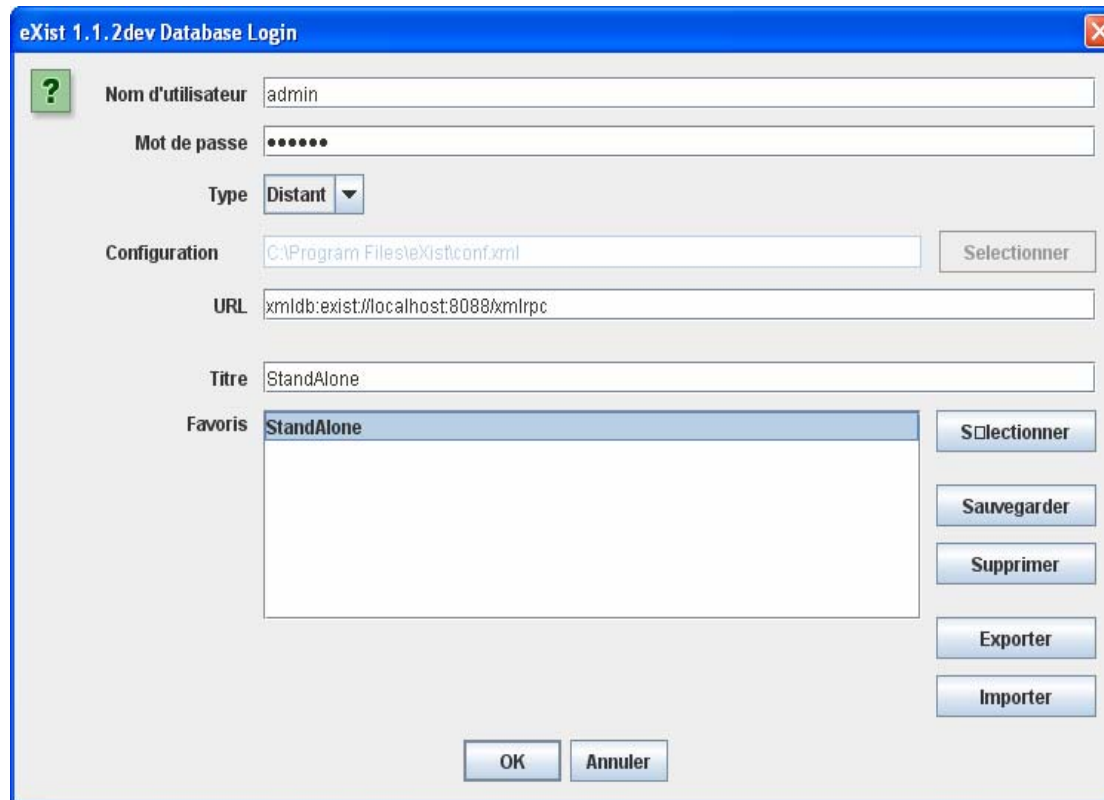
◆ Le client eXist

- Lancez le client *eXist Client Shell*, qui se trouve dans le menu démarrer:



Ma première collection

L'écran suivant apparaît :



The screenshot shows a dialog box titled "eXist 1.1.2dev Database Login". It contains the following fields and controls:

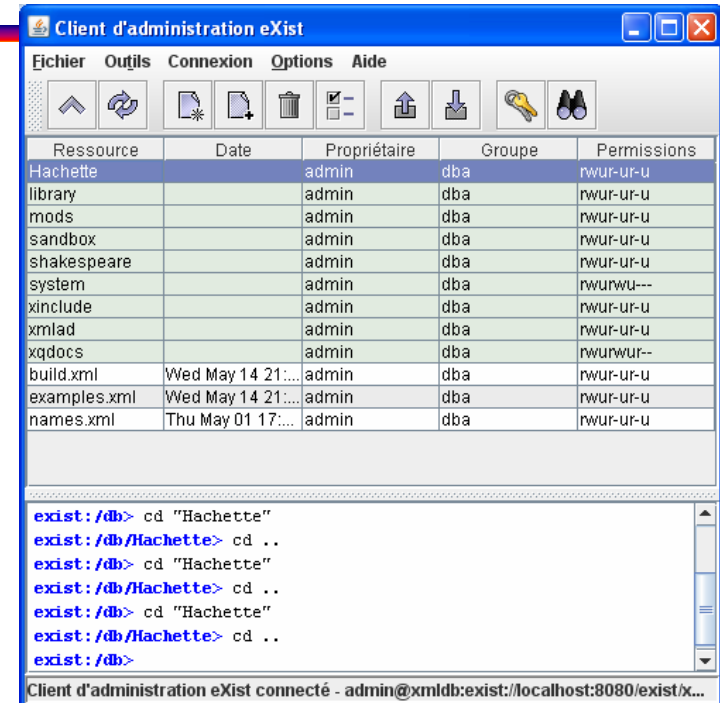
- Nom d'utilisateur:** Text field containing "admin".
- Mot de passe:** Password field with masked characters "••••••".
- Type:** Dropdown menu set to "Distant".
- Configuration:** Text field containing "C:\Program Files\exist\conf.xml" with a "Selectionner" button to the right.
- URL:** Text field containing "xmldb:exist://localhost:8088/xmlrpc".
- Titre:** Text field containing "StandAlone".
- Favoris:** A list box containing "StandAlone" with a "Selectionner" button to its right.
- Buttons:** "OK", "Annuler", "Sauvegarder", "Supprimer", "Exporter", and "Importer" are located on the right side of the dialog.

Saisissez alors le nom d'utilisateur et le mot de passe que vous avez choisi sur la page d'administration

Ma première collection

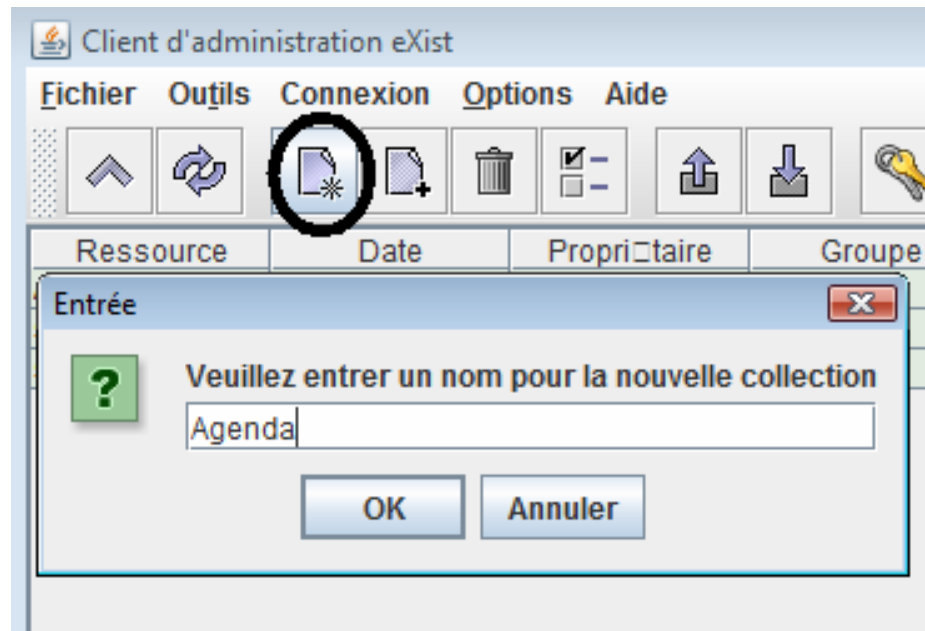
◆ Créer une nouvelle collection

- Sur la fenêtre « Client d'administration eXist », cliquez sur
 - Fichier >> créer une collection
 - Créez une collection du nom de *hachette*
 - Double-cliquez dessus (cf fenêtre de gauche)
 - Faites fichier→charger des fichiers/répertoires
 - Choisissez le fichier **XML-Auteurs.xml**, et faites de même avec le fichier **XML-Collections.xml**, fichiers qui sont donnés en même temps que le présent cours (dans l'archive)



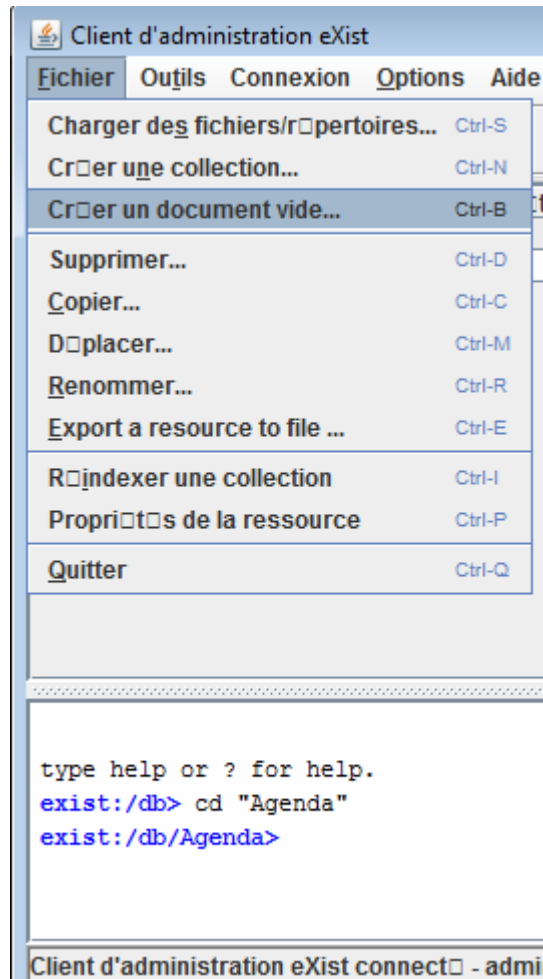
Créer une collection

- ◆ On peut aussi procéder comme suit



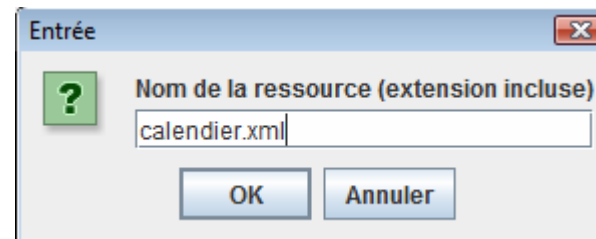
Créer une nouvelle collection (bouton entouré)
Entrez Agenda (attention à la majuscule) et cliquez sur OK

Créer une collection



Une fois la collection Agenda créée, cliquer dessus puis faire : Fichier -> Créer un document vide

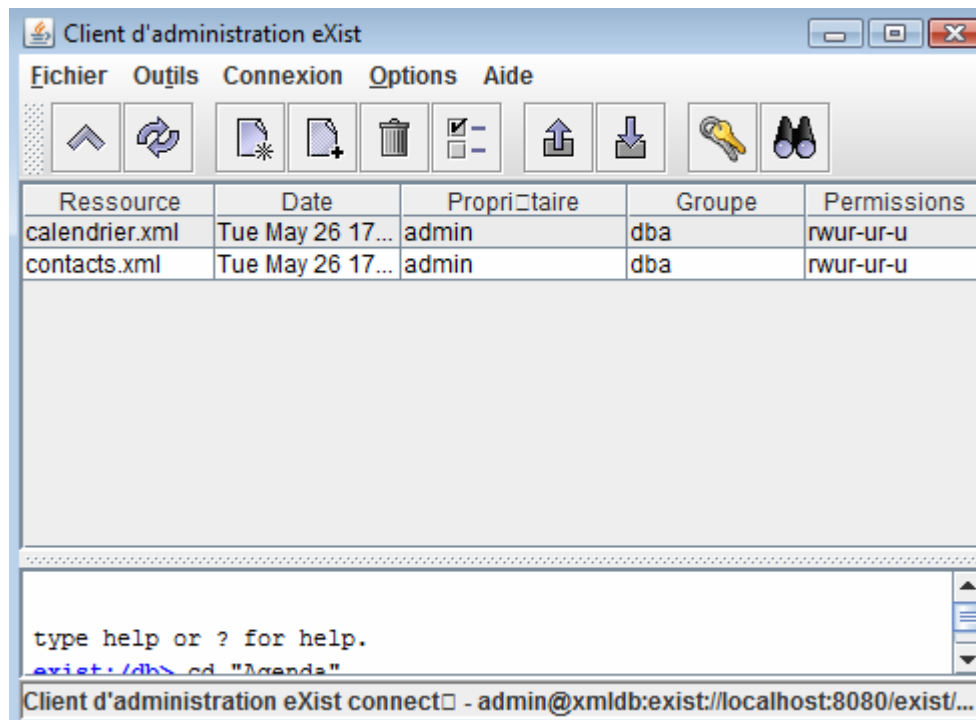
Créer une ressource **calendrier.xml** puis cliquez sur OK :



Refaire la même suite d'opérations et créer une ressource **contacts.xml**

Créer une collection

Vous obtenez :



Créer une collection

Ouvrir le fichier calendrier.xml et mettez dedans les balises <calendrier>
</calendrier> puis sauvegarder (bouton entouré)



Ouvrir le fichier contacts.xml et mettre les balises <contacts>
</contacts> (ou la balise unique <contacts/>) puis sauvegarder



Interrogation de la base de données

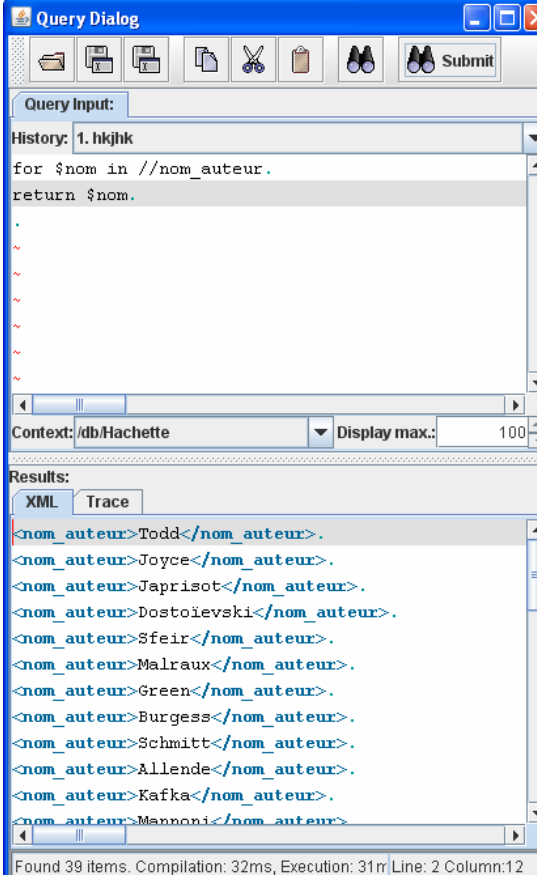
◆ Principe

- Cliquez sur
 - Outils >> interroger la base et la fenêtre Query Dialog s'ouvrira
- Tout est prêt désormais pour interroger la base de données
- Dans la fenêtre *client d'administration eXist* , double-cliquez sur la collection XML-Auteurs.xml pour l'ouvrir , afin de visualiser le fichier tout en faisant vos requêtes depuis la fenêtre *Query Dialog*
 - C'est le langage Xquery qui s'occupe d'interroger la base de données, via Xpath pour trouver les chemins

Interrogation de la base de données

- Écrivons quelques requêtes en XQuery sur la collection XML-Auteurs.xml:
 1. Lister les noms de tous les auteurs :

```
for $nom in
  //nom_aute
ur
return $nom
```



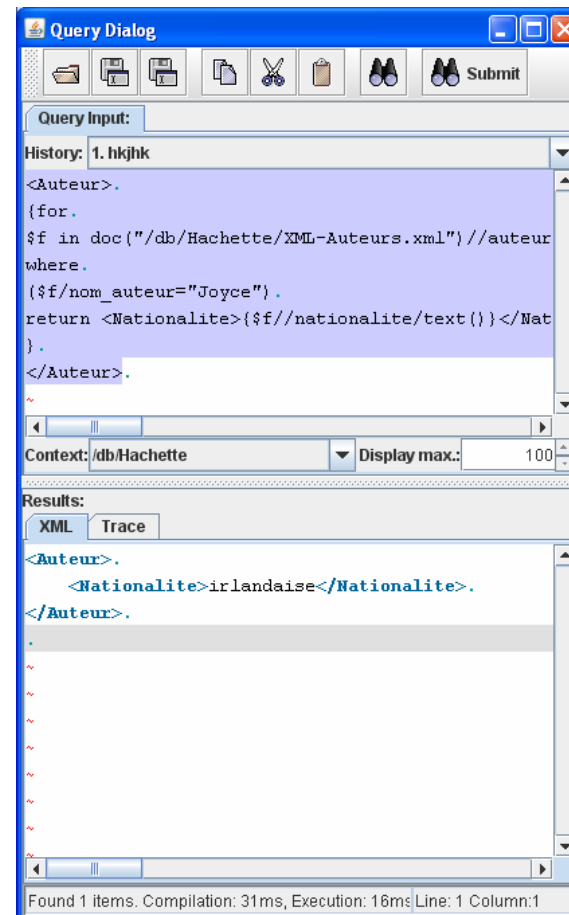
The screenshot shows a 'Query Dialog' window with a toolbar at the top containing icons for file operations and a 'Submit' button. The 'Query Input' section contains the XQuery: `for $nom in //nom_auteur. return $nom.` The 'Context' is set to `/db/Hachette` and 'Display max.' is 100. The 'Results' section is split into 'XML' and 'Trace' tabs, with 'XML' selected. It displays a list of author names in XML format: `<nom_auteur>Todd</nom_auteur>.`, `<nom_auteur>Joyce</nom_auteur>.`, `<nom_auteur>Japrisot</nom_auteur>.`, `<nom_auteur>Dostoievski</nom_auteur>.`, `<nom_auteur>Sfeir</nom_auteur>.`, `<nom_auteur>Malraux</nom_auteur>.`, `<nom_auteur>Green</nom_auteur>.`, `<nom_auteur>Burgess</nom_auteur>.`, `<nom_auteur>Schmitt</nom_auteur>.`, `<nom_auteur>Allende</nom_auteur>.`, `<nom_auteur>Kafka</nom_auteur>.`, and `<nom_auteur>Mannoni</nom_auteur>.` The status bar at the bottom indicates 'Found 39 items. Compilation: 32ms, Execution: 31ms Line: 2 Column:12'.

Résultat

Interrogation de la base de données

2. Lister la nationalité de tous les auteurs dont le nom est « Joyce »

```
<Auteur>
{for
$f in doc("XMLAuteurs.xml")//auteurs
where
($f/nom_auteur="Joyce")
return
  <Nationalite>{$f//nationalite/text()}</National
ite>
}
</Auteur>
```



Interrogation de la base de données

◆ Utilisation de la Sand Box

- Nous allons utiliser une autre API pour effectuer des requêtes sur notre collection
- Pour une première utilisation:
 - Identifiez-vous sur la page <http://localhost:8080/exist/admin/admin.xql>
 - Cliquez ensuite à gauche sur le lien «*Examples Setup* »
 - Cliquez sur *Skip*

Select a Page

- Home
- System Status
- Browse Collections
- User Management
- View Running Jobs
- Examples Setup
- Install Tools
- Backups
- Query Profiling

Examples Setup

eXist ships with a number of XQuery examples. Some to be stored in the database. The XQuery examples al with the distribution which can be downloaded by se

- eXist-db shipped files
- [The XML Acronym Demystifier](#) (approx. 384K)
- [The Mondial Database \(Europe\)](#) (approx. 410K)

Import Files

Skip

Interrogation de la base de données

◆ Suite

- Les exemples ont bien été importés. Le menu *Exemples* sur la page d'accueil permet d'y accéder (sinon revenir et cocher import files)
- Rendez-vous sur : XQuery Sandbox
- La fenêtre suivante doit apparaître:

The screenshot displays the XQuery Sandbox web application. At the top, the title "XQuery Sandbox" is visible, along with navigation links for "Home", "Download", and "Wiki". The main interface is divided into several sections:

- Slots:** A vertical list of 10 slots, each labeled "[1]" through "[10]" and containing the text "-- empty --".
- Paste saved query:** A text input field with a dropdown arrow, currently showing a list of example queries. The visible options include: "Simple fulltext query on Shakespeare plays", "Fulltext query using near", "Show the context of a match", "Group hits by play", "Show table of contents for Macbeth", "Find books by author", "Find a city by name", "Show countries with decreasing population", "Find spanish provinces and their cities", "List all organizations Germany is a member of", "Countries having a common border with Austria", "For each country, list 3 cities with highest population", "Show countries with highest roman catholic population", "Java binding (static)", "Java binding (instance)", "Request module", and "Display system properties".
- Buttons:** Below the "Paste saved query" field are three buttons: "Send", "Clear", and "Check".
- Display:** A dropdown menu set to "20".
- Results:** A large empty area below the buttons, with a status bar at the bottom left indicating "Found 1 in 0.015 seconds." and "Terminé".

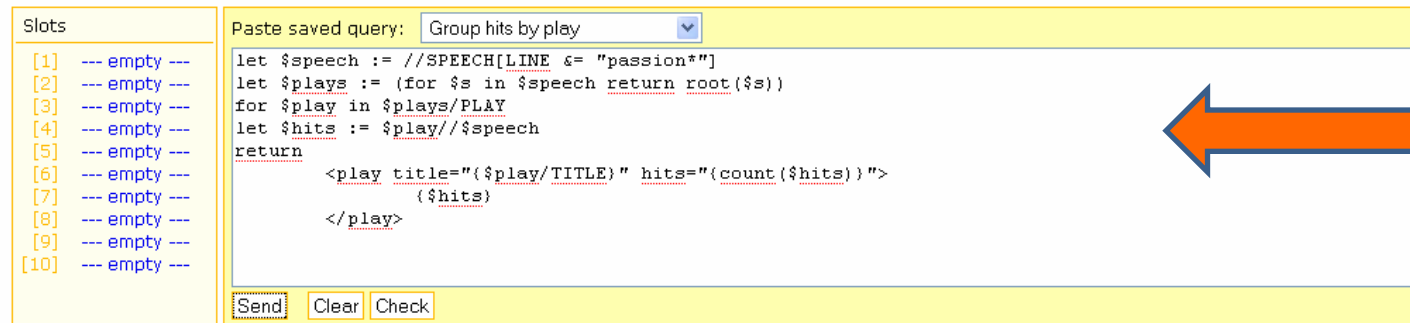
Interrogation de la base de données

◆ Ensuite

- Faites défiler la liste déroulante, comme dans l'écran précédent, et choisissez un exemple de requête (par exemple *Group hit by play*). Cliquez sur *Send*, pour obtenir le résultat suivant:

XQuery Sandbox

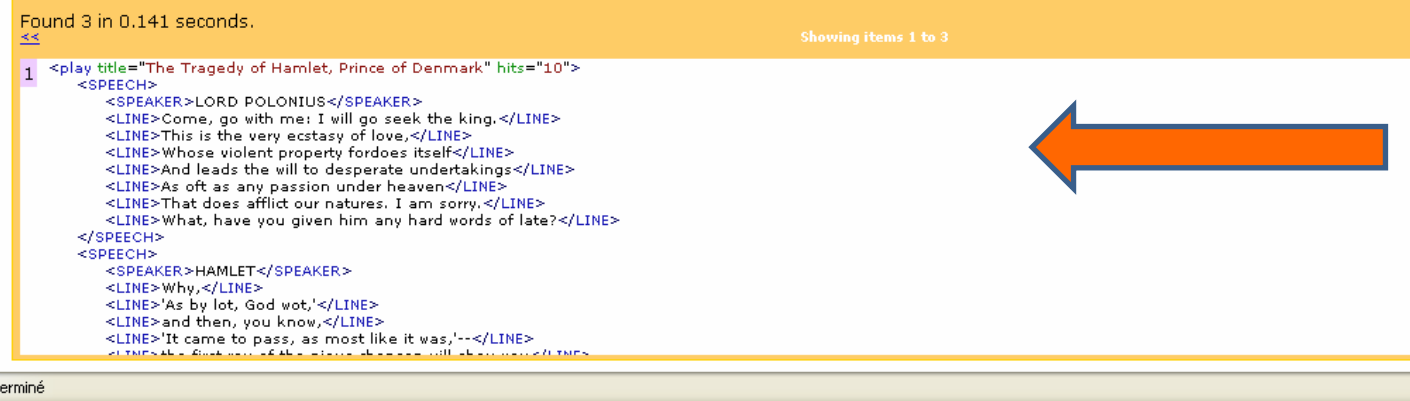
[Home](#)



The screenshot shows the XQuery Sandbox interface. On the left, there is a list of slots numbered [1] to [10], each containing the text "-- empty --". On the right, there is a text area for the XQuery. Above the text area is a dropdown menu labeled "Paste saved query:" with "Group hits by play" selected. Below the text area are three buttons: "Send", "Clear", and "Check".

```
let $speech := //SPEECH[LINE &= "passion*"]
let $plays := (for $s in $speech return root($s))
for $play in $plays/PLAY
let $hits := $play//$speech
return
  <play title="{ $play/TITLE}" hits="{count($hits)}">
    { $hits }
  </play>
```

Requête
XQuery



The screenshot shows the result of the XQuery. At the top, it says "Found 3 in 0.141 seconds." and "Showing items 1 to 3". Below this, there is a list of results. The first result is highlighted and shows the XML output of the query.

```
1 <play title="The Tragedy of Hamlet, Prince of Denmark" hits="10">
  <SPEECH>
    <SPEAKER>LORD POLONIUS</SPEAKER>
    <LINE>Come, go with me: I will go seek the king.</LINE>
    <LINE>This is the very ecstasy of love,</LINE>
    <LINE>Whose violent property fordoes itself</LINE>
    <LINE>And leads the will to desperate undertakings</LINE>
    <LINE>As oft as any passion under heaven</LINE>
    <LINE>That does afflict our natures. I am sorry.</LINE>
    <LINE>What, have you given him any hard words of late?</LINE>
  </SPEECH>
  <SPEECH>
    <SPEAKER>HAMLET</SPEAKER>
    <LINE>Why,</LINE>
    <LINE>As by lot, God wot,</LINE>
    <LINE>and then, you know,</LINE>
    <LINE>'It came to pass, as most like it was,'--</LINE>
  </SPEECH>
```

Résultat de la
requête

Terminé

Interrogation de la base de données

◆ Ensuite

- Testons enfin notre requête sur la collection XML-Auteurs.xml
- Copier-coller la requête dans l'encadré sur l'espace de requête, puis cliquer sur send

```
<Auteur>
{for
$f in doc("/db/Hachette/XML-
Auteurs.xml")//auteurs
where
($f/nom_auteur="Joyce")
return
    <Nationalite>{$f//nationalite/text()}</National
ite>
}
</Auteur>
```

- Le résultat que vous devez obtenir se trouve sur la page suivante

Interrogation de la base de données

XQuery Sandbox

Slots

- [1] --- empty ---
- [2] --- empty ---
- [3] --- empty ---
- [4] --- empty ---
- [5] --- empty ---
- [6] --- empty ---
- [7] --- empty ---
- [8] --- empty ---
- [9] --- empty ---
- [10] --- empty ---

Paste saved query:

```
<Auteur>
{for
$f in doc("/db/Hachette/XML-Auteurs.xml")//auteurs
where
($f/nom_auteur="Joyce")
return <Nationalite>{$f//nationalite/text()}</Nationalite>
}
</Auteur>
```

Found 1 in 0.031 seconds. Showing items 1 to 1

<<

1 <Auteur>
 <Nationalite>irlandaise</Nationalite>
</Auteur>

Terminé

eXist et Java (XML:DB)

- ◆ On utilise l'API XML:DB (<http://exist-db.org/api/>)
 - Cet API permet
 - Gestion de la base et des collections
 - Stockage et récupération des ressources
 - Requêtes sur toute ou une partie de l'arborescence
 - Il existe des extensions déjà implémentées qui permettent entre autre :
 - La gestion des utilisateurs
 - La gestion de plusieurs bases de données en parallèle, avec chacune une configuration propre
 - Pour plus d'informations :
http://exist.sourceforge.net/devguide_xmlldb.html

eXist et XML:DB avec Eclipse

◆ Création de projet sous Eclipse

1. Ouvrez Eclipse et créez un nouveau projet du nom de Agenda
2. Une fois le projet créé, ouvrez les propriétés de votre projet, menu *Java Build Path (clic droit sur le projet → Properties)*
3. Dans l'onglet *libraries*, choisissez *Externals jars*
4. Il va s'agir d'importer les différents Jar de eXist, qui se trouvent dans le répertoire de eXist: (par défaut, C:/Program Files/eXist)
 - Jars qui sont dans le répertoire eXist/lib/core
 - Jars qui sont directement dans le répertoire de eXist
5. Sélectionnez chaque fois tout les fichiers Jars dans le dossier, et importez-les

eXist et XML:DB sans Eclipse

◆ Pour utiliser XML:DB sans passer par Eclipse

- il faut définir la variable d'environnement **CLASSPATH** manuellement, avec une commande comme suit (sous Windows) :

```
SET CLASSPATH= %CLASSPATH%; . ;
```

```
C:/Program Files/eXist/lib/core/xmlldb.jar;
```

```
C:/Program Files/eXist/exist.jar;
```

```
C:/Program Files/eXist/lib/core/log4j-1.2.15.jar;
```

```
C:/Program Files/eXist/lib/core/antlr-2.7.6.jar;
```

```
C:/Program Files/eXist/lib/core/commons-collections-3.1.jar;
```

```
[etc...]
```

Et ce avec [toutes les archives .jar montrées précédemment](#)

XML-DB

◆ Accès à la collection

- Pour accéder à une collection, il faut se connecter au driver de la base de données
- Le driver utilisé dans eXist est `org.exist.xmldb.DatabaseImpl`
 - Il doit être enregistré avec le `DatabaseManager`
- On extrait une collection à partir de la base de données en appelant la méthode statique `DatabaseManager.getCollection ()`
- Pour identifier la collection désirée, il faut lui passer comme paramètre l'URI complet dont le format est :
 - `xmlldb : [DATABASE-ID] :// [HOST-ADDRESS] / db/ collection`
- où `DATABASE-ID` est "eXist", et `HOST-ADDRESS` est localhost:8080 si nous sommes en local et si le port utilisé est 8080

XML-DB

◆ Accès à la collection (suite)

- Du fait que plusieurs drivers peuvent être enregistrés avec le gestionnaire de la base de données, la première partie de l'URI (xmldb : exist) est requise pour **déterminer quelle classe** de driver sera utilisée
- Le **database-id** est utilisé par le gestionnaire de la base de donnée pour sélectionner le bon driver dans la liste des drivers disponibles
- Pour utiliser eXist, cet **ID** doit toujours être égale à exist (sauf si l'application traite avec de multiples instances de bases de données, auquel cas **ID** pourra aussi prendre le nom des autres bases de données).

XML-DB

◆ Exemple

- Créer un package appelé `serv_exist`, ensuite créer une classe `QueryExample` qui contient le code qui se trouve sur la page suivante

- La ligne :

```
Collection col = DatabaseManager.getCollection  
("xmldb:exist://localhost:8080/exist/xmlrpc/db/hachette" );
```

créera un objet collection contenant la collection hachette, créée précédemment

```

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

public class QueryExample {
    public static void main(String args[]) throws Exception {
        String driver = "org.exist.xmldb.DatabaseImpl";
        //déterminer la classe de driver utilisée
        //Pour se connecter à une base de données il est essentiel de charger dans un premier temps le
        //pilote de la base de données à laquelle on désire se connecter grâce à un appel au
        // DriverManager (gestionnaire de pilotes) : Class.forName("nom.de.la.classe");
        // Cette instruction charge le pilote et crée une instance de cette classe.

        Class cl = Class.forName(driver);
        Database database = (Database)cl.newInstance();
        DatabaseManager.registerDatabase(database);

        //Accès à la collection
        Collection col = DatabaseManager.getCollection(
            "xml:db:exist://localhost:8080/exist/xmlrpc/db/hachette" );

        //Appel au service permettant d'exécuter des requêtes avec XPath
        XPathQueryService service =
            (XPathQueryService) col.getService("XPathQueryService", "1.0");
        service.setProperty("indent", "yes");

        //Description de la requête
        ResourceSet result = service.query("//*[name()='nom_auteur']");
        ResourceIterator i = result.getIterator();
        while(i.hasMoreResources()) {
            Resource r = i.nextResource();
            System.out.println((String)r.getContent());
        }
    }
}

```

XML-DB API

◆ Accès à la collection (suite)

- La ligne

```
XPathQueryService service = (XPathQueryService)
col.getService("XPathQueryService", "1.0");
service.setProperty("indent", "yes");
```

- Appelle le service qui va exploiter votre collection : ici, XPathQueryService pour interroger la base

- On crée finalement une requête sur la base :

```
ResourceSet result = service.query("//*[name()='nom_auteur']");
ResourceIterator i = result.getIterator();
while(i.hasMoreResources()) {
Resource r = i.nextResource();
System.out.println((String)r.getContent());
}
```

Tous les tags
/ <nom_auteur>

- En exécutant le main de cette classe, vous devez avoir le résultat suivant

```
<terminated> Main [Java Application] C:\Program Files\Java\jre1.6.0_03\bin\javaw.exe (23 mai 2008 21:18:05)
<nom_auteur>Todd</nom_auteur>
<nom_auteur>Joyce</nom_auteur>
<nom_auteur>Japrisot</nom_auteur>
<nom_auteur>Dostoievski</nom_auteur>
<nom_auteur>Sfeir</nom_auteur>
<nom_auteur>Malraux</nom_auteur>
<nom_auteur>Green</nom_auteur>
<nom_auteur>Burgess</nom_auteur>
<nom_auteur>Schmitt</nom_auteur>
<nom_auteur>Allende</nom_auteur>
<nom_auteur>Kafka</nom_auteur>
<nom_auteur>Mannoni</nom_auteur>
<nom_auteur>Hasek</nom_auteur>
<nom_auteur>Pennac</nom_auteur>
<nom_auteur>Orwell</nom_auteur>
<nom_auteur>Nothomb</nom_auteur>
<nom_auteur>Sijie</nom_auteur>
<nom_auteur>Buzzati</nom_auteur>
<nom_auteur>Christie</nom_auteur>
<nom_auteur>Voltaire</nom_auteur>
<nom_auteur>Grass</nom_auteur>
<nom_auteur>Süskind</nom_auteur>
<nom_auteur>Levi</nom_auteur>
<nom_auteur>Proust</nom_auteur>
<nom_auteur>Adams</nom_auteur>
<nom_auteur>Neruda</nom_auteur>
<nom_auteur>Ceccaldi</nom_auteur>
<nom_auteur>Izzo</nom_auteur>
<nom_auteur>Garcia-Marquez</nom_auteur>
<nom_auteur>Dumas</nom_auteur>
<nom_auteur>Maubourguet</nom_auteur>
<nom_auteur>Pellerin</nom_auteur>
<nom_auteur>Verba</nom_auteur>
<nom_auteur>Verdier</nom_auteur>
<nom_auteur>Prax</nom_auteur>
<nom_auteur>Lona</nom_auteur>
```


XML-DB API

◆ Enregistrement

- Si vous souhaitez enregistrer un nouveau document dans le répertoire, vous devez créer une nouvelle ressource XML, lui affectant le contenu du nouveau document, en appelant la méthode `storeResource` de la classe `Collection`
 - Tout d'abord, une nouvelle ressource est créée par la méthode `Collection.createResource ()` qui attend deux paramètres:
 - l'identifiant et
 - le type de ressources en cours de création
 - Ce que montre la classe suivante, que vous pouvez tester en créant une nouvelle classe `StoreExample` dans Eclipse :

```
public class StoreExample {
    public final static String URI = "xmldb:exist://localhost:8080/exist/xmlrpc";

    public static void main(String args[]) throws Exception {
        if(args.length < 2) {
            System.out.println("usage: StoreExample collection-path document");
            System.exit(1);
        }

```

```
String collection = args[0], file = args[1];
```

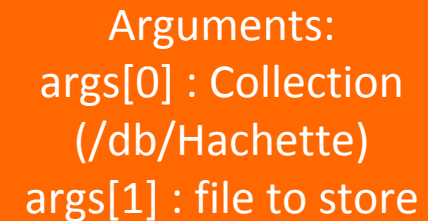
```
// initialisation du driver
```

```
String driver = "org.exist.xmldb.DatabasImpl";
Class cl = Class.forName(driver);
Database database = (Database)cl.newInstance();
DatabaseManager.registerDatabase(database);
```

```
// Accès à la collection
```

```
Collection col =
    DatabaseManager.getCollection(URI + collection);
```

Initialisation du driver
DatabasImpl



Arguments:
args[0] : Collection
(/db/Hachette)
args[1] : file to store

```
if(col == null) {
    // la collection n'existe pas. Créer une root pour une nouvelle collection
    // pour simplifier, on suppose que la nouvelle collection est un fils direct de la root collection,
    // e.g. /db/test. L'exemple échouera sinon
    Collection root = DatabaseManager.getCollection(URI + "/db");
    CollectionManagementService mgtService = (CollectionManagementService)
        root.getService("CollectionManagementService", "1.0");
    col = mgtService.createCollection(collection.substring("/db".length()));
}
// créer une nouvelle XMLResource; un id sera affecté à la nouvelle ressource
XMLResource document = (XMLResource)col.createResource(null, "XMLResource");
File f = new File(file);
if(!f.canRead()) {
    System.out.println("cannot read file " + file);
    return;
}
document.setContent(f);
System.out.print("storing document " + document.getId() + "...");
col.storeResource(document);
System.out.println("ok.");
}
}
```

eXist sur le Web

◆ Les API

- Il existe des API permettant de créer un lien entre eXist et des langages comme PHP ou ASP, mais la manière **la plus simple** et sûrement **la plus efficace** reste **d'utiliser XQuery** directement pour réaliser des pages Web
- Cela passe par la création d'un **Servlet** ou d'un **cocon (Cocoon)** XQuery, qui accède à la base en étant appelé par un script donné
- Un bon exemple peut être consulté à cet endroit :

http://exist.sourceforge.net/devguide_xquery.html

eXist et les autres langages

◆ Les API

- XML-RPC (voir plus haut) fournit tout un protocole basé sur le protocole RPC, afin de permettre la communication avec des bases de données XML comme eXist
- Il existe des API XML-RPC pour beaucoup de langages comme Java, Perl, C, ...
- Un exemple d'une URI XML-RPC en Java :
- **Usage:** `org.exist.http://demo.exist-db.org/examples.xmlrpc.Retrieve "` + "`path-to-document`"
- Un ensemble d'exemples peut être consulté à cet endroit :
- `http://exist.sourceforge.net/devguide_xmlrpc.html`

Au final ?

◆ Les plus :

- Une **bonne alternative** aux SGBD relationnels
- Compatible et accessible avec **beaucoup de langages**
- Traitements **rapides** et **efficaces** des données
- **Hiérarchisation** des données intéressante
- Gestion des utilisateurs


◆ Les moins :

- **Pas encore** de prise en charge de la **DTD** des fichiers
- Une approche différente **à maîtriser**
- Encore en **développement**

Conclusion

◆ Points importants

- eXist n'est pas encore beaucoup utilisé, pourtant cette base de données mérite que l'on se penche dessus
- Simple à l'installation et simple d'utilisation, il est un outil puissant, notamment grâce l'utilisation que l'on peut en faire dans d'autres langages (se reporter à la partie traitant de eXist et les autres langages (Java))
- Pour approfondir ...



<http://peccatte.karefil.com/software/RBourret/xmlBD.htm#isxmldatabase>
<http://localhost:8080/exist/devguide.xml>
<http://xmlfr.org/actualites/tech/040210-0001>