# 1 Quick Start

This chapter describes the fastest way to install and start a database server instance. We show in brief how to store XML documents and explain the most important steps to access the stored XML data with the provided tools.

## 1.1 Introduction

The research project *XML Transaction Coordinator (XTC)* was initiated to explore aspects of native XML data processing. Basically, the project consists of the prototypes *XTCserver* (a native XML database management system), *XTCclp* (a text-based command line processor), and XTCcc (a control center with graphical user interface). Within the XTC project we are evaluating XQuery processing, index structures, XML APIs, and transaction isolation for concurrent document accesses based on node-level locking.

## 1.2 Requirements, Installation, Server Startup

The entire XTC project is implemented in Java and runs on any operating system providing for a Java Virtual Machine version 5.0 and above. Because the Java version 5.0 bytecode is not compatible downwards to Java version 1.4.x or 1.3.x it is an essential requirement to provide for a Java 5.0 runtime environment.

All components of the XTCserver-release are compressed into a ZIP-archive. Just extract the archive into an arbitrary folder of your file system. Because we currently do not use any environment variables of the operating system it is **very important** that you start each component directly at the folder it is stored in.

### 1.2.1 File System Structure

Extracting the XTCserver ZIP-archive into an arbitrary folder of your file system creates the following sub folders:

- BIN
  - XTCcc
  - XTCclp
  - XTCserver
    - cntfiles
    - logfiles
    - tempfiles
- DRIVER
  - DOC
- SAMPLES
  - DOM
  - SAX
  - XQuery

The *BIN* folder contains all executables of the project. These are the graphical control center XTCcc, the text-based command line processor XTCclp, and the XTCserver itself. The actual XML data managed by the XTCserver is stored in so-called container files. They are resided in the *cntfiles* folder. The *logfiles* and *tempfiles* folders contain log and temporary files.

The driver, stored as *XTCdriver.jar* in the *DRIVER* folder, provides for methods to administer the XTCserver and is used for application development to access the stored XML data via the DOM and SAX interface or to execute XQuery statements. The *DOC* folder contains a generated Java documentation of the driver classes and methods.

The *SAMPLE* folder consists of three sub folders containing a collection of sample Java programs for DOM, SAX, and XQuery access to server-side managed XML data.

### 1.2.2 Installing the Server

To install the server just change into the folder *BIN/XTCserver* and execute the script *XTCserver* with the parameter *install*:

```
XTCserver install
```

There are scripts for Windows and Unix environments which invoke the Java Virtual Machine with the *XTCserver.jar* file. The main class of the server *XTCserver.class* is already specified within the manifest file in the Java archive.

The installation procedure creates various container files in the *cntfiles* folder and generates the meta-data required by the XTCserver. The meta data is stored in the so-called *master document* within the server and can be examined with the command line processor.

### 1.2.3 Starting and Stopping the Server

After a successful installation two components can be used to start the XTCserver. The XTCserver itself is started resp. stopped in the folder *BIN/XTCserver* by the commands

```
XTCserver start
XTCserver stop
```

Starting the server opens all container files, allocates the corresponding memory buffers, initializes all required data structures in main memory, and provides applications for RMI ports to connect to the server.

In order to observe the availability of the XTCserver, a further component, called the *XTCguard*, can be started and stopped. Therefor, you execute in the folder *BIN/XTCserver* the commands

```
XTCserver startguard
XTCserver stopguard
```

The XTCguard observes the port of the XTCserver and restarts the server automatically if it is not available (e.g., in case of a crash). If you have started the XTCguard and want to stop the XTCserver, then first stop the guard and then stop the server.

## 1.3 Connecting to the Server

There are several ways for connecting to the server and accessing the stored data. The fastest way is the command line processor XTCclp, a more comfortable way is the control center XTCcc. The XTCserver also provides for an http interface, so you also can access your XML documents with a standard web browser. Applications are using the XTCdriver-methods for establishing a connection to the server. A description of the XTCdriver for application development and the feature of interaction between the DOM, SAX, and XQuery interface within one single transactional context is not given in this quick-start-chapter.

### 1.3.1 The Command Line Processor

The text-based command line processor is started in folder *BIN/XTCclp* by one of the following commands:

```
XTCclp
XTCclp username
XTCclp username password
```

If you start the XTCclp without passing a user name or password the command line processor prompts you a request where you can input the missing parameters.

If you want to connect to a server running on a different machine than your command line processor you directly have to invoke the Java Virtual Machine with the XTCclp Java archive and parameters for the name of the host on which the sever is running, the port, and the user name and password:

```
java -jar XTCclp.jar host port username password
```

The default user name and password is *xtc* and *xtc*. Currently, our prototype does not support any user management facilities, so you always have to use these parameters for authentication. The XTCserver is running by default on port 24203.

After having started the command line processor you can type *?* or *help* at the command line prompt to get a list and description of all available commands.

#### Document Management

The XTCserver provides for a file-system-like view on the stored XML documents. Corresponding to a file system, XML documents are simply stored into folders. During the installation procedure the XTCserver creates the sample documents *sample.xml* and *index.html* (in XHTML format) for your first experiences with the system. All sample programs are based on the *sample.xml* document, too.

The contents of the current working folder can be displayed by the command *dir*. You can create new folders with the command *mkdir* and change into these folders with *chdir:*

```
mkdir testfolder
chdir testfolder
```

### Document Storage

In order to store a document, the command *put* is executed with the *fileName* parameter which specifies the file that has to be stored. The command line processor then transports the file *fileName* to the server and initializes the storage procedure. The file is stored into the XTCserver with its original name. If you want to store the file with a different name you can specify this name as the second parameter of the *put* command.

```
put <fileName> [<destDocName>]
```

The specified file of the *put* command must be a well-formed XML document. If you want to store an arbitrary file (e.g., PDF documents or images for a web presentation via the XTC-server http interface) you can store these files as binary large objects (BLOBs) with the command *putblob:*

```
putblob mylogo.gif
```

### Document Access

Displaying a document stored inside the XTCserver is simply done by typing the corresponding document name at the command line prompt. The specified document is then being reconstructed and sent from the server to the command line processor. For displaying the document *sample.xml* just type:

```
sample.xml
```

You can additionally append some simple XPath expressions to the document name which are evaluated before the document is returned. An example for an expression on the sample document is:

```
sample.xml/bib/book/title
```

The XPath processor only accepts the child axis and some simple value comparison expressions because this processor is only used for internal access to the system's meta-data which is also maintained in an XML document. For more complex queries we offer the XQuery processor.

To evaluate XQuery expressions just input the XQuery statement at the command line processor's prompt. The XQuery statement is sent to the XTCserver, executed by the XQuery processor, and the result is returned and directly displayed by the command line processor. For example, querying all authors of the *sample.xml* document can be performed with the XQuery statement

```
<authors>{doc(„sample.xml“)/bib/book/author}</authors>
```

The command *get* is used to reconstruct a complete document from the XTCserver and to store the document in the local file system of the command line processor. The following command

```
get sample.xml sample2.xml
```

gets the sample document from the server and stores it with name *sample2.xml* in your file system.

### Element Access by ID Attribute

Currently, we are only indexing elements within an XML document by their ID attribute values. Because we do not consider any schemas for the documents all attributes corresponding to the name *xtc:id* are treated as ID attributes. This name can be configured in the server's configuration file.

The command *elementbyid* addresses the element which owns an ID attribute with the specified value, reconstructs the entire subtree of the element to an XML fragment, and returns the constructed fragment to the command line processor. The following command returns the *book* element of the sample document with the *xtc:id* attribute value *book2*:

```
elementbyid sample.xml book2
```

### Transaction Management

Each command typed in the command line processor's prompt is executed within a transactional context. Therefor, for each command, a new transaction is started, the command is executed, and finally, the transaction is committed.

In order to execute several commands within one transaction you can explicitly start, commit, or rollback a transaction with the commands *begin*, *commit*, and *rollback*. If a transaction is currently running each command is executed within the transaction's context. If no transaction is currently running a new transaction is started for the command execution as described above.

With the command *isolation* you can adjust the isolation level with which the next started transaction will run. Currently, the isolation levels *none*, *uncommitted*, *committed*, *repeatable*, and *serializable* are supported.

### XTCserver Meta-Data

All meta-data needed to maintain the folder structures for the XML document storage, the container files, and memory buffer management is stored in the so-called *master document*. For information purposes, this document can be displayed with the command *master*.

## 1.3.2 The Control Center

The XTC control center is a graphical user interface which allows to navigate across stored XML documents via the DOM API, to start SAX parsing, or to execute XQuery statements. The control center is located in the folder *BIN/XTCcc*. Change into this folder and execute the command:

```
XTCcc
```

After the control center has be started, select the *Connect* item in the *Control Center* menu and the login dialog will appear. Type into the corresponding input lines the default user name and password (*xtc* and *xtc*) and click the *connect* button to establish a connection to the server. If the XTCserver is not running on the same machine as your control center select the *Server Settings* tab in the login dialog and specify the host and port of the XTCserver. After establishing the connection to the server, the control center presents the working environment as depicted in Figure 1.

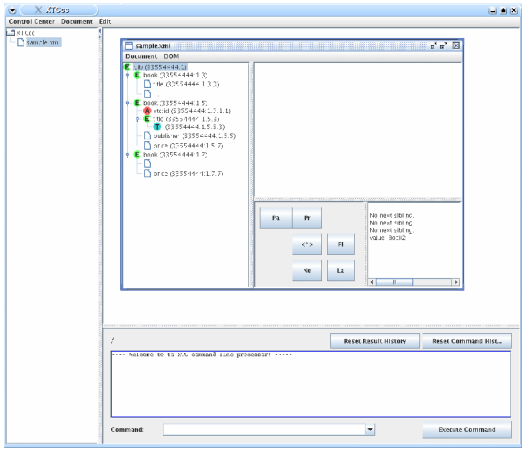

*Figure 1: Control Center*

On the left pane the folder structure and stored documents of the XTCserver are shown. Immediately after the server installation this overview only contains the sample documents *sample.xml* and *index.html*.

To open a document for navigation just highlight the document by clicking it once with your left mouse button and select the *Open* Item in the *Document* menu or the context menu. You can open the context menu by clicking on the document name with your right mouse button.

The XML document will be opened in a separate window in the right upper pane where initially only the document root node is shown. In order to navigate across the document you can use the navigation methods which are listed in the *DOM* menu of the document window or you can use the large navigation buttons. The buttons allow a faster access to the parent (*Pa*), previous (*Pr*) or next sibling (*Ne*), and the first (*Fi*) or last (*La*) child of the selected context node. The button in the middle (<*>) returns the context node's value if it is available.

SAX parsing of the opened document can be performed by selecting the *SAX parse* item in the *Document* menu of the opened document window. The recognized nodes during the parsing procedure are displayed in a separate window.

For executing an XQuery statement you have to select the *Execute XQuery* item of the *Document* menu either in the control center menu bar or the opened document menu bar. If you are executing an XQuery statement via the opened document menu you can reference the opened document itself by the function d*oc(".")* within the query. The following statement executed on the sample document returns all book titles in the *sample.xml* document:

```
<titles>
  {let $d := doc(".")/bib/book/title return $d}
</titles>
```

If you want to execute the same query with the *Execute XQuery* item of the *Document* menu in the control center menu bar you can use the following statement:

```
<titles>
  {let $d := doc("sample.xml")/bib/book/title return $d}
</titles>
```

If an executed XQuery statement returns a well-formed XML data structure the result is opened as a document in a new document window and can again be processed with the DOM or SAX API or a further XQuery statement.

### 1.3.3 The Http Connection

The XTCserver also initializes at startup an http interface on the default port 8080. This port can be adjusted in the server's configuration file. The http interface provides the file-system-like view on XML documents, BLOBs, and folders for common web browsers.

After a successful startup of the XTCserver you can access the sample document *index.html* with your web browser via the URL:

```
http://localhost:8080
```

If no document name is specified in the URL (like in the example above) the XTCserver tries to get the default document *index.html* from the directory of the URL. This default document name can also be adjusted in the server's configuration file. To address directly a document just append its name to the URL as in common web browser:

```
http://localhost:8080/sample.xml
```

As an additional feature you can transform your stored XML documents with XSL style sheets. Assumed, you have already stored an XSL style sheet as a well-formed XML document with name *myStyle.xsl* into the XTCserver, then you can request the *sample.xml* document transformed corresponding to the style sheet with the following URL:

```
http://localhost:8080/sample.xml@myStyle.xsl
```

In general, for requesting a transformed XML document via a web browser, just append the style sheet document name separated by an @-character to the URL.