MarkLogic Server

XCC Developer's Guide

MarkLogic 6
September, 2012

Last Revised: 6.0-5, March, 2014

Copyright © 2014 MarkLogic Corporation. All rights reserved.

Table of Contents

XCC Developer's Guide

1.0 Introduction to XCC 4
- 1.1 Overview of XCC 4
  - 1.1.1 XCC Client Libraries Communicate With an XDBC Server 4
  - 1.1.2 Client-Server Architecture 5
  - 1.1.3 Automatically Pools Connections 5
- 1.2 API and Other Documentation 6
- 1.3 XCC Requirements 6
  - 1.3.1 XCC MarkLogic Server Requirements 6
  - 1.3.2 XML Contentbase Connector for Java (XCC/J) Requirements 6
  - 1.3.3 XML Contentbase Connector for .NET (XCC/.NET) Requirements 6

2.0 Programming in XCC 7
- 2.1 Configuring an XDBC Server 7
- 2.2 XCC Sessions 7
- 2.3 Point-In-Time Queries 8
- 2.4 Automatically Retries Exceptions 8
- 2.5 Coding Basics 8
- 2.6 Accessing SSL-Enabled XDBC App Servers 10
  - 2.6.1 Creating a Trust Manager 10
  - 2.6.2 Accessing a Keystore 12
  - 2.6.3 Managing Client Side Authentication 13
- 2.7 Understanding Result Caching 14
- 2.8 Multi-Statement Transactions 14
  - 2.8.1 Overview 14
  - 2.8.2 Example: Using Multi-Statement Transactions in Java 15
  - 2.8.3 Terminating a Transaction in an Exception Handler 16
  - 2.8.4 Retrying Multi-statement Transactions 17
- 2.9 Participating in XA Transactions 18
  - 2.9.1 Overview 18
  - 2.9.2 Predefined Security Roles for XA Participation 19
  - 2.9.3 Enlisting MarkLogic Server in an XA Transaction 19
  - 2.9.4 Heuristically Completing a Stalled Transaction 20
    - 2.9.4.1 Understanding Heuristic Completion 21
    - 2.9.4.2 Heuristically Completing a MarkLogic Server Transaction 22
    - 2.9.4.3 Cleaning Up After Heuristic Completion 24
  - 2.9.5 Reducing Blocking Caused by Slow XA Transactions 24

3.0 Downloading and Using the XCC API 25

3.1 XCC/J Java Packages 25
3.2 XCC/.NET C# Packages 26

4.0 Using the Sample Applications 27
4.1 Setting Up Your Environment 27
4.1.1 Setting Up Your MarkLogic Server Environment 27
4.1.2 Setting Up Your Java Environment 27
4.1.3 Setting Up Your .NET Environment 28
4.2 Sample Applications 28
4.2.1 ContentFetcher 29
4.2.2 ContentLoader 29
4.2.3 DynamicContentStream 30
4.2.4 HelloSecureWorld 30
4.2.5 HelloWorld 31
4.2.6 ModuleRunner 31
4.2.7 OutputStreamInserter 31
4.2.8 SimpleQueryRunner 32
4.2.9 XA 32

5.0 Technical Support 34

Combined Product Notices 35
COPYRIGHT 35
TRADEMARK NOTICE 47

1.0 Introduction to XCC

The XML Contentbase Connector (XCC) is an interface to communicate with MarkLogic Server from a Java or .NET middleware application layer. This chapter provides background on XCC and includes the following sections:

- Overview of XCC
- API and Other Documentation
- XCC Requirements

1.1 Overview of XCC

The XML Contentbase Connector (XCC) is used to communicate between a Java or .NET application layer and MarkLogic Server. XCC replaces the XDBC libraries, and the XDBC libraries are now deprecated. You can still use XDBC 3.0 to communicate with MarkLogic Server 4.0, but if you want to use any of the functionality newer than 3.1 (for example, point-in-time queries), you must use XCC; XDBC will no longer have any new features added (bug fixes only), so you should use XCC.

This section provides an overview of XCC and includes the following parts:

- XCC Client Libraries Communicate With an XDBC Server
- Client-Server Architecture
- Automatically Pools Connections

1.1.1 XCC Client Libraries Communicate With an XDBC Server

XCC has a set of client libraries that you use to build applications that communicate with MarkLogic Server. There are Java and .NET versions of the client libraries. XCC requires that an XDBC server is configured in MarkLogic Server.

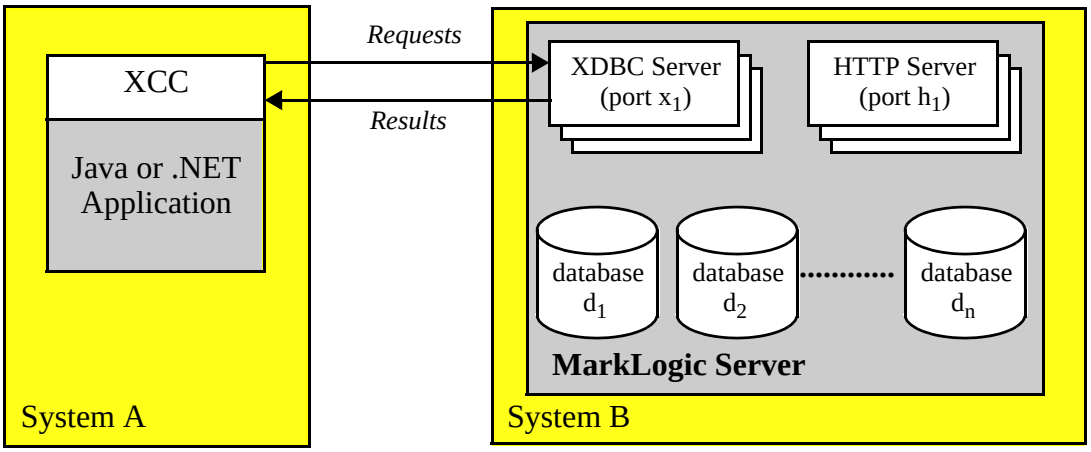
An XDBC server responds to XDBC and XCC requests. XDBC and XCC use the same wire protocol to communicate with MarkLogic Server. You can write applications either as standalone applications or ones that run in an application server environment. Your XCC-enabled application connects to a specified port on a system that is running MarkLogic Server, and communicates with MarkLogic Server by submitting requests (for example, XQuery statements) and processing the results returned by those programs. These XQuery programs can incorporate calls to XQuery functions stored and accessible by MarkLogic Server, and accessible from any XDBC-enabled application. The XQuery programs can perform the full suite of XQuery functionality, including loading, querying, updating and deleting content.

XQuery requests submitted via XCC return results as specified by the XQuery code. These results can include XML and a variety of other datatypes. It is the XCC application's responsibility to parse, process and interpret these results in a manner appropriate to the variety of datatypes available. There are a number of publicly available libraries for assisting with this task, or you

may write your own code. In order to accept connections from XCC-enabled applications, MarkLogic Server must be configured with an XDBC Server listening on the designated port. Each XDBC Server connects by defaut to a specific database within MarkLogic Server, but XCC provides the ability to communicate with any database in the MarkLogic Server cluster to which your application connects (and for which you have the necessary permissions and privileges).

1.1.2 Client-Server Architecture

XCC communicates with MarkLogic Server with a client-server architecture, where the XCC application is the client and MarkLogic Server is the server. The following figure illustrates the high-level architecture:



As shown in the diagram above, the XCC-enabled application can run on the same system as an instance of MarkLogic Server (a host), or it can run on a completely different system, as long as the two systems are networked together.

In the diagram, the XCC application running on System A has opened an XDBC connection to port x_1 on System B. On System B, MarkLogic Server is configured with an XDBC Server listening to port x_1, and that XDBC Server connects to database$_{d1}$. Consequently, the configuration shown in the diagram above allows the XCC application on System A to submit XQuery requests (including query, load, update and delete) for evaluation against database$_{d1}$.

> **Note:** Do not use an HTTP load balancer with an XCC application. XCC communicates with MarkLogic Server via the XDBC protocol, which is not the same as HTTP.

1.1.3 Automatically Pools Connections

XCC automatically does connection pooling, so you do not need to write any connection pooling logic in your application. The XCC `Session` object automatically obtains and releases connections for XCC applications as needed.

1.2 API and Other Documentation

This document provides an introduction to the XCC developer libraries. For detailed API documentation for XCC and for MarkLogic Server, or to learn how to configure XDBC servers in MarkLogic Server, see the appropriate documents:

- Java API documentation (Javadoc available on developer.marklogic.com)
- .NET API documentation (available on developer.marklogic.com)
- MarkLogic Server *Application Developer's Guide*
- MarkLogic Server *Administrator's Guide*
- *MarkLogic XQuery and XSLT Function Reference*

1.3 XCC Requirements

This section lists the requirements for XCC and has the following parts:

- XCC MarkLogic Server Requirements
- XML Contentbase Connector for Java (XCC/J) Requirements
- XML Contentbase Connector for .NET (XCC/.NET) Requirements

1.3.1 XCC MarkLogic Server Requirements

XCC requires MarkLogic Server 3.*x* or later. You must connect to MarkLogic Server 3.1 or later to use any of the 3.1-specific or later features (for example, point-in-time queries).

> **Note:** Do not use an HTTP load balancer with an XCC application. XCC communicates with MarkLogic Server via the XDBC protocol, which is not the same as HTTP.

1.3.2 XML Contentbase Connector for Java (XCC/J) Requirements

XCC/J has the following requirements:

- Java 1.6 or later
- MarkLogic Server 3.1 or later (on any platform)

1.3.3 XML Contentbase Connector for .NET (XCC/.NET) Requirements

XCC/.NET has the following requirements:

- .NET 2.0 Service Pack 1 (SP1), or later
- MarkLogic Server 3.1 or later (on any platform)

2.0 Programming in XCC

XCC allows you to create multi-tier applications that communicate with MarkLogic Server as the underlying content repository. This chapter describes some of the basic programming concepts used in XCC. It includes the following sections:

- Configuring an XDBC Server
- XCC Sessions
- Point-In-Time Queries
- Automatically Retries Exceptions
- Coding Basics
- Accessing SSL-Enabled XDBC App Servers
- Understanding Result Caching
- Multi-Statement Transactions
- Participating in XA Transactions

2.1 Configuring an XDBC Server

Use the Admin Interface to set up an XDBC server, specifying a name, port, a database to access, and other configuration parameters. For detailed instructions how to configure an XDBC Server, see the *Administrator's Guide*. You need an XDBC Server for an XCC program to communicate with MarkLogic Server.

2.2 XCC Sessions

XCC programs use the `Session` interface to set up and control communication with MarkLogic Server. XCC automatically creates and releases connections to MarkLogic Server as needed, and automatically pools the connections so that multiple requests are handled efficiently.

A `Session` handles authentication with MarkLogic Server and holds a dynamic state, but it is a lightweight object. It is OK to create and release Session objects as needed and as makes logical sense for your program. Do not expend effort to pool and reuse them, however, because they are not expensive to create. For example, if your program is doing multiple requests one after another, create a `Session` object at the beginning and close it when the last request is complete.

You set up the connection details with the `ContentSource` object. You can submit the connection details when you invoke the XCC program with a URL that has the following form:

```
xcc://username:password@host:port/database
```

Also, there are discrete arguments to the constructors in the API to set up any or all portions of the connection details.

2.3 Point-In-Time Queries

Point-in-time queries allow you to query older versions of content in a database. In an XCC application, you set up the options for any requests submitted to MarkLogic Server with the `RequestOptions` class. One of the options you can set is the effective point-in-time option. Therefore, to set up a query to run at a different point in time, you just set that option (the `setEffectivePointInTime` method in Java) on the `RequestOptions`. The query will then run at the specified point in time.

There are several things you must set up on MarkLogic Server in order to perform point-in-time queries. For details, see the "Point-In-Time Queries" chapter of the *Application Developer's Guide*.

2.4 Automatically Retries Exceptions

Certain exceptions that MarkLogic Server throws are *retryable*; that is, the exception is thrown because of a condition that is transitory, and applications can try the request again after getting the exception. XCC will automatically retry retryable exceptions in single-statement transactions. You can control the maximum number of retryable exceptions with the `RequestOptions` interface.

Multi-statement transactions cannot automatically be retried by the server. Your application must handle retries explicitly when using multi-statement transactions. For details, see "Retrying Multi-statement Transactions" on page 17.

2.5 Coding Basics

To use XCC, there are several basic things you need to do in your Java or .NET code:

- Import the needed libraries.
- Set up the `ContentSource` object to authenticate against MarkLogic Server.
- Create a new `Session` object.
- Add a `Request` to the session object.
- Submit the request and get back a `ResultSequence` object from MarkLogic Server.
- Do something with the results (print them out, for example).
- Close the session.

The following are Java code samples that illustrate these basic design patterns:

```
package com.marklogic.xcc.examples;

import com.marklogic.xcc.ContentSource;
import com.marklogic.xcc.ContentSourceFactory;
import com.marklogic.xcc.Session;
import com.marklogic.xcc.Request;
import com.marklogic.xcc.ResultSequence;
```

```
URI uri = new URI("xcc://user:pass@localhost:8000/mycontent");
ContentSource contentSource =
          ContentSourceFactory.newContentSource (uri);

Session session = contentSource.newSession();

Request request = session.newAdhocQuery ("\"Hello World\"");

ResultSequence rs = session.submitRequest (request);

System.out.println (rs.asString());

session.close();
```

Note: `Session` objects are not thread safe. A `Session` object should not be used concurrently by multiple threads.

2.6 Accessing SSL-Enabled XDBC App Servers

There are three basic approaches for an XCC application to create a secure connection to an SSL-enabled XDBC App Server, which include:

- Creating a Trust Manager
- Accessing a Keystore
- Managing Client Side Authentication

These approaches are described in this section and demonstrated in the `HelloSecureWorld.java` example distributed with your MarkLogic XCC software distribution.

2.6.1 Creating a Trust Manager

This section describes how to use a simple Trust Manager for X.509-based authentication. The Trust Manager shown here does not validate certificate chains and is therefore unsafe and should not be used for production code. See your Java documentation for details on how to create a more robust Trust Manager for your specific application or how to obtain a Certificate Authority from a keystore.

To enable SSL access using a trust manager, import the following classes in addition to those described in “Coding Basics” on page 8:

```
import javax.net.ssl.SSLContext;
import com.marklogic.xcc.SecurityOptions;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

import java.security.KeyManagementException;
import java.security.cert.X509Certificate;
import java.security.cert.CertificateException;
```

Create a trust manager and pass it to the `SSLContext.init()` method:

```
protected SecurityOptions newTrustOptions()
  throws Exception
{
   TrustManager[] trust = new TrustManager[] { new X509TrustManager() {
    public void checkClientTrusted(
       X509Certificate[] x509Certificates,
       String s)
          throws CertificateException {
     // nothing to do
    }

    public void checkServerTrusted(
       X509Certificate[] x509Certificates,
       String s)
          throws CertificateException {
     // nothing to do
    }

    public X509Certificate[] getAcceptedIssuers() {
     return null;
    }
  }
};

  SSLContext sslContext = SSLContext.getInstance("SSLv3");
  sslContext.init(null, trust, null);
  return new SecurityOptions(sslContext);
}
```

Call `ContentSourceFactory.newContentSource()` with a host name, port, user name, password, and SSL security options defined by `newTrustOptions()`:

```
ContentSource cs =
    ContentSourceFactory.newContentSource (host,
                                           port,
                                           username,
                                           password,
                                           null,
                                           newTrustOptions());
```

Note: If you are passing a URI to `ContentSourceFactory.newContentSource()`, specify a connection scheme of `xccs`, rather than `xcc`., as shown in "Accessing a Keystore" on page 12.

2.6.2 Accessing a Keystore

You can use the Java `keytool` utility to import a MarkLogic certificate into a keystore. See the Java JSSE documentation for details on the use of the `keytool` and your keystore options.

You can explicitly specify a keystore, as shown in this example, or you can specify a null keystore. Specifying a null keystore causes the `TrustManagerFactory` to locate your default keystore, as described in the *Java Secure Socket Extension (JSSE) Reference Guide*.

To enable SSL by accessing certificates in a keystore, import the following classes in addition to those described in "Coding Basics" on page 8:

```
import com.marklogic.xcc.SecurityOptions;
import com.marklogic.xcc.ContentSource;
import com.marklogic.xcc.ContentSourceFactory;

import java.io.FileInputStream;
import java.net.URI;
import javax.net.ssl.KeyManager;
import javax.net.ssl.KeyManagerFactory;
import javax.net.ssl.TrustManager;
import javax.net.ssl.TrustManagerFactory;
import javax.net.ssl.X509TrustManager;
import javax.net.ssl.SSLContext;

import java.security.KeyStore;
import java.security.cert.X509Certificate;
```

Get the signed certificate from a keystore and pass it to the `SSLContext.init()` method:

```
protected SecurityOptions newTrustOptions()
  throws Exception
{

// Load key store with trusted signing authorities.
  KeyStore trustedKeyStore = KeyStore.getInstance("JKS");
  trustedKeyStore.load(
    new FileInputStream("C:/users/myname/.keystore"),
    null);

// Build trust manager to validate server certificates using the
   specified key store.
  TrustManagerFactory trustManagerFactory =
    TrustManagerFactory.getInstance("SunX509");
  trustManagerFactory.init(trustedKeyStore);
  TrustManager[] trust = trustManagerFactory.getTrustManagers();

  SSLContext sslContext = SSLContext.getInstance("SSLv3");
  sslContext.init(null, trust, null);
  return new SecurityOptions(sslContext);
}
```

Call `ContentSourceFactory.newContentSource()` with a URI:

```
ContentSource cs =
    ContentSourceFactory.newContentSource (uri,
                                           newTrustOptions());
```

The URI is passed from the command line in the form of:

```
xccs://username:password@hostname:port
```

2.6.3 Managing Client Side Authentication

You can define a `KeyManager`, if your client application is required to send authentication credentials to the server. The following example adds client authentication to the `newTrustOptions` method shown in “Accessing a Keystore” on page 12:

```
protected SecurityOptions newTrustOptions()
  throws Exception
{

// Load key store with trusted signing authorities.
  KeyStore trustedKeyStore = KeyStore.getInstance("JKS");
  trustedKeyStore.load(
    new FileInputStream("C:/users/myname/.keystore"),
    null);

// Build trust manager to validate server certificates using the
   specified key store.
  TrustManagerFactory trustManagerFactory =
    TrustManagerFactory.getInstance("SunX509");
  trustManagerFactory.init(trustedKeyStore);
  TrustManager[] trust = trustManagerFactory.getTrustManagers();

// Load key store with client certificates.
  KeyStore clientKeyStore = KeyStore.getInstance("JKS");
  clientKeyStore.load(
    new FileInputStream("C:/users/myname/.keystore"),
    null);

// Get key manager to provide client credentials.
  KeyManagerFactory keyManagerFactory =
    KeyManagerFactory.getInstance("SunX509");
  keyManagerFactory.init(clientKeyStore, "passphrase");
  KeyManager[] key = keyManagerFactory.getKeyManagers();

// Initialize the SSL context with key and trust managers.
  SSLContext sslContext = SSLContext.getInstance("SSLv3");
  sslContext.init(key, trust, null);
  return new SecurityOptions(sslContext);
}
```

2.7 Understanding Result Caching

When you submit a request to MarkLogic Server, the results are returned to your application in a `ResultSequence`. By default the `XdmItem` objects in the sequence are cached. That is, all the result items are read and buffered in memory. Cached results do not tie up any connection resources, and are usually preferred.

A non-cached, or streaming, `ResultSequence` may only be accessed sequentially and hold the connection to MarkLogic Server open. Individual results may only be read once and on demand, so the result set consumes less memory, at the cost of less efficient access.

If you are retrieving large results, such as a large binary document, you may disable result caching to conserve memory. You may disable result caching per request by creating a `RequestOption` object with the setting disabled, and associating it with a request, either directly with `Request.setOptions` or passing it as a parameter to a Request creation method such as `Session.newAdhocQuery`. You may disable result caching per session by setting the default request options for the session using `Session.setDefaultRequestOptions`.

For details, see the `ResultSequence` XCC javadoc.

2.8 Multi-Statement Transactions

By default, all transactions run as single-statement, auto-commit transactions. MarkLogic Server also supports multi-statement, explicitly commited transactions in XQuery and XCC/J. Transaction concepts are discussed in detail in the "Understanding Transactions" chapter of the *Application Developer's Guide*. This section covers only related behaviors unique to XCC/J:

- Overview
- Example: Using Multi-Statement Transactions in Java
- Terminating a Transaction in an Exception Handler
- Retrying Multi-statement Transactions

2.8.1 Overview

To use multi-statement, explicitly created transactions with XCC/J:

1. Create a `Session` object in the usual way.

2. Call `Session.setTransactionMode` to set the transaction mode to either `UPDATE` or `QUERY`. All transactions created in the session from then until the transaction modes changes run as a multi-statement transaction of the appropriate type.

3. Call `Session.submitRequest` as usual to operate on your data. All requests run in the same transaction until the transaction is commited or rolled back.

4. Call `Session.commit` or `Session.rollback` to commit or rollback the transaction. If the session ends or times out without explicitly commit or rolling back, the transaction is rolled back.

5. To restore a session to the default, single-statement transaction model, call `Session.setTransactionMode` with a value of `AUTO`.

Multi-statement query transactions allow all the statements in a transaction to share the same point-in-time view of the database, as discussed in "Point-In-Time Queries" on page 8.

In a multi-statement update transaction, updates performed by one statement (or request) are visible to subsequent statements in the same transaction, without being visible to other transactions.

A multi-statement transaction remains open until it is committed or rolled back. Use `Session.commit`. to commit a multi-statement transaction and make the changes visible in the database. Use `Session.rollback` to roll back a multi-statement transaction, discarding any updates. Multi-statement transactions are implicitly rolled back when the containing session ends or the transaction times out. Failure to explicitly commit or rollback a multi-statement update transaction can tie up resources, hold locks unnecessarily, and increase the chances of deadlock.

Note: You may receive a `java.lang.IllegalStateException` if you call `Session.commit` from an exception handler when there are no pending updates in the current transaction. Committing from a handler is not recommended.

For a detailed discussion of multi-statement transactions, see the "Understanding Transactions" chapter of the *Application Developer's Guide.*

Multi-statement transactions impose special re-try semantics on XCC/J applications. For details, see "Retrying Multi-statement Transactions" on page 17.

2.8.2 Example: Using Multi-Statement Transactions in Java

The following example demonstrates using multi-statement transactions in Java. The first multi-statement transaction in the session inserts two documents into the database, calling `Session.commit` to complete the transaction and commit the updates. The second transaction demonstrates the use of `Session.rollback`. The third transaction demonstrates implicitly rolling back updates by closing the session.

```
import java.net.URI;

import com.marklogic.xcc.ContentSource;
import com.marklogic.xcc.ContentSourceFactory;
import com.marklogic.xcc.Session;

public class SimpleMST {
  public static void main(String[] args) throws Exception {
    if (args.length != 1) {
```

```
      System.err.println("usage: xcc://user:password@host:port/
contentbase");
      return;
    }

    // Obtain a ContentSource object for the server at the URI.
    URI uri = new URI(args[0]);
    ContentSource contentSource =
        ContentSourceFactory.newContentSource(uri);

    // Create a Session and set the transaction mode to trigger
    // multi-statement transaction use.
    Session updateSession = contentSource.newSession();
    updateSession.setTransactionMode(Session.TransactionMode.UPDATE);

    // The request starts a new, multi-statement transaction.
    updateSession.submitRequest(updateSession.newAdhocQuery(
    "xdmp:document-insert('/docs/mst1.xml', <data/>)"));

    // This request executes in the same transaction as the previous
    // request and sees the results of the previous update.
    updateSession.submitRequest(updateSession.newAdhocQuery(
    "xdmp:document-insert('/docs/mst2.xml', fn:doc('/docs/
mst1.xml'));"));

    // After commit, updates are visible to other transactions.
    // Commit ends the transaction after current stmt completes.
    updateSession.commit();                  // txn ends, updates kept

    // Rollback discards changes and ends the transaction.
    updateSession.submitRequest(updateSession.newAdhocQuery(
    "xdmp:document-delete('/docs/mst1.xml')"));
    updateSession.rollback();               // txn ends, updates lost

    // Closing session without calling commit causes a rollback.
    updateSession.submitRequest(updateSession.newAdhocQuery(
    "xdmp:document-delete('/docs/mst1.xml')"));
    updateSession.close();                   // txn ends, updates lost
  }
}
```

2.8.3 Terminating a Transaction in an Exception Handler

Calling `Session.commit` from an exception handler that wraps a request participating in a multi-statement transaction may raise `java.lang.IllegalStateException`. You may always safely call `Session.rollback` from such a handler.

Usually, an exception raised during multi-statement transaction processing leaves the `Session` open, allowing you to continue working in the transaction after handling the exception. However, in order to preserve consistency, exceptions occurring under the following circumstances always roll back the transaction:

- After an XQuery statement has finished but before the XCC request is completed
- In the middle of an explicit commit or rollback

If such a rollback occurs, the current transaction is terminated before control reaches your exception handler. Calling `Session.commit` when there is no active transaction raises a `java.lang.IllegalStateException`. Calling `Session.rollback` when there is no active transaction does not raise an exception, so rollback from a handler is always safe.

Therefore, it is usually only safe to call `Session.commit` from an exception handler for specific errors you expect to receive and for which you can predict the state of the transaction.

2.8.4 Retrying Multi-statement Transactions

MarkLogic Server sometimes detects the need to retry a transaction. For example, if the server detects a deadlock, it may cancel one of the deadlocked transactions, allowing the other to complete; the cancelled transaction should be re-tried.

With single-statement, auto-commit transactions, the server can usually retry automatically because it has the entire transaction available at the point of detection. However, the statements in a multi-statement transactions from XCC/J clients may be interleaved with arbitrary application-specific code of which the server has no knowledge.

In such cases, instead of automatically retrying, the server throws a `RetryableXQueryException`. The calling application is then responsible for re-trying all the requests in the transaction up to that point. This exception is more likely to occur when using multi-statement transactions.

The following example demonstrates logic for re-trying a multi-statement transaction. The multi-statement transaction code is wrapped in a retry loop with an exception handler that waits between retry attempts. The number of retries and the time between attempts is up to the application.

```
import java.net.URI;

import com.marklogic.xcc.ContentSource;
import com.marklogic.xcc.ContentSourceFactory;
import com.marklogic.xcc.Session;
import com.marklogic.xcc.exceptions.RetryableXQueryException;

public class TransactionRetry {
  public static final int MAX_RETRY_ATTEMPTS = 5;
  public static final int RETRY_WAIT_TIME = 1;

  public static void main(String[] args) throws Exception {
    if (args.length != 1) {
      System.err.println("usage: xcc://user:password@host:port/
contentbase");
      return;
    }
```

```
        // Obtain a ContentSource object for the server at the URI.
        URI uri = new URI(args[0]);
        ContentSource contentSource =
            ContentSourceFactory.newContentSource(uri);

        // Create a Session and set the transaction mode to trigger
        // multi-statement transaction use.
        Session session = contentSource.newSession();
        session.setTransactionMode(Session.TransactionMode.UPDATE);

        // Re-try logic for a multi-statement transaction
        for (int i = 0; i < MAX_RETRY_ATTEMPTS; i++) {
          try {
            session.submitRequest(session.newAdhocQuery(
              "xdmp:document-insert('/docs/mst1.xml', <data/>)"));
            session.submitRequest(session.newAdhocQuery(
              "xdmp:document-insert('/docs/mst2.xml', fn:doc('/docs/
mst1.xml'));"));
            session.commit();
            break;
          } catch (RetryableXQueryException e) {
            Thread.sleep(RETRY_WAIT_TIME);
          }
        }
        session.close();
      }
    }
```

2.9 Participating in XA Transactions

MarkLogic Server can participate in distributed transactions by acting as a Resource Manager in an XA/JTA transaction. This section covers the following related topics:

- Overview
- Enlisting MarkLogic Server in an XA Transaction
- Heuristically Completing a Stalled Transaction
- Reducing Blocking Caused by Slow XA Transactions

2.9.1 Overview

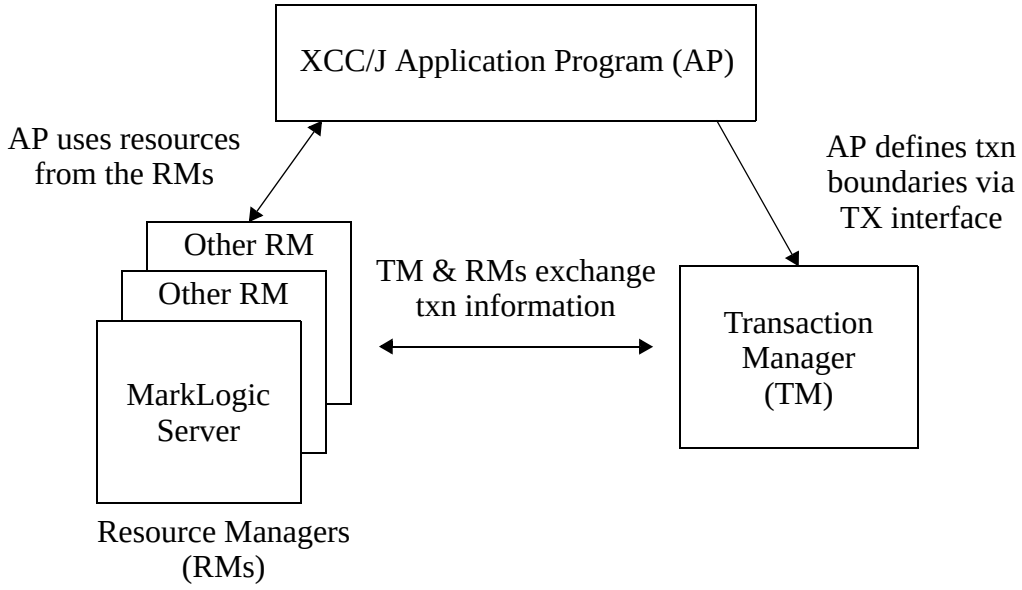
XA is a standard for distributed transaction processing defined by The Open Group. For details about XA, see:

https://www2.opengroup.org/ogsys/jsp/publications/PublicationDetails.jsp?catalogno=c193

JTA is the Java Transaction API, a Java standard which supports distributed transactions across XA resources. For details about JTA, see http://java.sun.com/products/jta/.

The XCC API includes support for registering MarkLogic Server as a resource with an XA Transaction Manager. For details, see “Enlisting MarkLogic Server in an XA Transaction” on page 19.

The basic XA architecture as it applies to MarkLogic Server is shown in the following diagram:



2.9.2 Predefined Security Roles for XA Participation

The following security roles are predefined for participating in and administering XA transactions:

- The `xa` user role allows creation and management of one’s own XA transaction branches in MarkLogic Server.
- The `xa-admin` role allows creation and management of any user’s XA transaction branches in MarkLogic Server.

The `xa` role is required to participate in XA transactions. The `xa-admin` role is intended primarily for Administrators who need to complete or forget XA transactions; see “Heuristically Completing a Stalled Transaction” on page 20.

2.9.3 Enlisting MarkLogic Server in an XA Transaction

To use MarkLogic Server in an XA transaction, use the `Session.getXAResource` method to register your XCC session as a `javax.transaction.xa.XAResource` with the XA Transaction Manager.

The following code snippet shows how to enlist an XCC session in a global XA transaction. Once you enlist the `Session`, any work performed in the session is part of the global transaction. For complete code, see the sample code in `com.marklogic.xcc.examples.XA`.

```
javax.transaction.TransactionManager tm = ...;
Session session = ...;

try {
    // Begin a distributed transaction
    tm.begin();

    // Add the MarkLogic Session to the distributed transaction
    javax.transaction.xa.XAResource xaRes = session.getXAResource();
    tm.getTransaction().enlistResource(xaRes);

    // Perform MarkLogic Server updates under the global transaction
    session.submitRequest(session.newAdhodquery(
        "xdmp:document-insert('a', <a/>)"));

    // Update other databases here

    //Commit all updates together
    tm.commit();
} catch (Exception e) {
    e.printStackTrace();
    if (tm. getTransaction != null) tm.rollback();
} finally {
    session.close();
}
```

When MarkLogic Server acts as an XA transaction Resource Manager, requests submitted to the server are always part of a multi-statement update transaction, with the following important differences:

- The `Session.TransactionMode` setting is ignored. The transaction is always a multi-statement update transaction, even if only a single request is submitted to MarkLogic Server during the global transaction.
- The application should not call `Session.commit` or `xdmp:commit`. The transaction is committed (or rolled back) as part of the global XA transaction. To commit the global transaction, use the Transaction Manager with which the MarkLogic Server `XAResource` is registered.
- The application may call `Session.rollback` or `xdmp:rollback`. Doing so eventually causes rollback of the global XA transaction. Rolling back via the Transaction Manager is usually preferable.

To learn more about multi-statement transactions, see “Multi-Statement Transactions” on page 14.

2.9.4 Heuristically Completing a Stalled Transaction

Under extreme circumstances, a MarkLogic Server administrator may need to heuristically complete (intervene to manually commit or rollback) the MarkLogic Server portion of a prepared XA transaction. This section covers the following topics related to heuristic completion:

- Understanding Heuristic Completion
- Heuristically Completing a MarkLogic Server Transaction
- Cleaning Up After Heuristic Completion

2.9.4.1 Understanding Heuristic Completion

This section provides a brief overview of the concept of heuristically completing XA transactions. For instructions specific to MarkLogic Server, see "Heuristically Completing a MarkLogic Server Transaction" on page 22.

The unit of work managed by a Resource Manager in an XA transaction is a *branch*. Manually intervening to force completion of a prepared XA transaction branch is *making a heuristic decision*, or *heuristically completing* the branch. The branch may be heuristically completed by either committing or rolling back the local transaction.

XA uses Two Phase Commit to commit or rollback global transactions. Normal transaction completion follows the flow:

- The Transaction Manager instructs all participants to prepare to commit.
- Each participant reports responds with whether or not it is ready to commit.
- If all participants report prepared to commit, the Transaction Manager instructs all participants to commit.
- If one or more participants is not prepared to commit, the Transaction Manager instructs all participants to roll back.

If the Transaction Manager goes down due to a failure such as loss of network connectivity or a system crash, the Transaction Manager does not remember the global transaction when it comes back up. In this case, the local transaction times out normally, or may be cancelled with a normal rollback, using `xdmp:transaction-rollback`.

If the Transaction Manager goes down after the transaction is prepared, the Transaction Manager normally recovers and resumes the flow described above. However, it may not always be possible to wait for normal recovery.

For example, if connectivity to the Transaction Manager is lost for a long time, locks may be held unacceptably long on documents in MarkLogic Server. Under such circumstances, the administrator may heuristically complete a branch of the global transaction to release resources.

> **Note:** Heuristic completion bypasses the Transaction Manager and the Two Phase Commit process, so it can lead to data integrity problems. Use heuristic completion only as a last resort.

The XA protocol requires a Resource Manager to remember the outcome of a heuristic decision, allowing the Transaction Manager to determine the status of the branch when it resynchronizes the global transaction.This remembered state is automatically cleaned up if the global transaction eventually completes with the same outcome as the heuristic decision.

Manual intervention may be required after a heuristic decision. If the Transaction Manager recovers and makes a different commit/rollback decision for the global transaction than the local heuristic decision for the branch, then data integrity is lost and must be restored manually.

For example, if the administrator heuristically completes a branch by committing it, but the global transaction later rolls back, then the heuristically completed branch requires manual intervention to roll back the previously committed local transaction and make the resource consistent with the other global transaction participants.

Heuristic completion is not needed in cases where a global transaction stalls prior to being prepared. In this case, the global transaction is lost and the local branches time out or otherwise fall back on normal failure mechanisms.

2.9.4.2 Heuristically Completing a MarkLogic Server Transaction

Use the `xdmp:xa-complete` built-in function to heuristically complete the MarkLogic Server branch of a prepared global XA transaction. When using `xdmp:xa-complete`, you must indicate:

- whether or not to commit the local transaction
- whether or not MarkLogic Server should remember the heuristic decision outcome (commit or rollback)

Usually, you should rollback the local transaction and remember the heuristic decision outcome.

Forgetting the heuristic decision leads to an error and possibly loss of data integrity when the Transaction Manager subsequently attempts to resynchornize the global transaction. If the outcome is remembered, then the Transaction Manager can learn the status of the branch and properly resume the global transaction.

The following examples demonstrate several forms of heuristic completion. The 3rd parameter indicates whether or not to commit. The 4th parameter indicates whether or not to remember the outcome:

```
(: commit and remember the transaction outcome:)
xdmp:xa-complete($forest-id, $txn-id, fn:true(), fn:true())

(: roll back and remember the transaction outcome:)
xdmp:xa-complete($forest-id, $txn-id, fn:false(), fn:true())

(: commit and forget the transaction outcome :)
xdmp:xa-complete($forest-id, $txn-id, fn:true(), fn:false())
```

The forest id parameter of `xdmp:xa-complete` identifies the coordinating forest. Once an XA transaction is prepared, the coordinating forest remembers the state of the MarkLogic Server branch until the global transaction completes. Use the Admin UI or `xdmp:forest-status` to determine the transaction id and coordinating forest id.

For example, the following query retrieves a list of all transaction participants in transactions that are prepared but not yet comitted. The coordinating forest id for each participant is included in the results. For details, see `xdmp:forest-status` in *XQuery and XSLT Reference Guide*.

```
xquery version "1.0-ml";
for $f in xdmp:database-forests(xdmp:database())
return xdmp:forest-status($f)//*:transaction-participants
```

Additional cleanup may be necessary if the Transaction Manager resumes and the global transaction has an outcome that does not match the heuristic decision. For details, see “Cleaning Up After Heuristic Completion” on page 24.

You may also use the Admin Interface to heuristically rollback XA transaction or to forget a heuristic decision. See Rolling Back a Prepared XA Transaction Branch in the *Administrator’s Guide*.

2.9.4.3 Cleaning Up After Heuristic Completion

If a heuristic decision is made for a MarkLogic Server branch of an XA transaction and the Transaction Manager subsequently completes the transaction, there are two possible outcomes:

- The global transaction completes with an outcome that matches the heuristic decision. No further action is required.
- The global transaction completes with an outcome that does not match the heuristic decision. Take the clean up steps listed below.

If the global transaction outcome does not agree with the heuristic decision, you may need to do the following to clean up the heuristic decision:

- Take whatever manual steps are necessary to restore data integrity for the heuristically completed branch.
- If the heuristic decision was remembered by setting the `remember` parameter of `xdmp:xa-complete` to true, call `xdmp:xa-forget` to clean up the remaining transaction state information.

2.9.5 Reducing Blocking Caused by Slow XA Transactions

Since XA transactions may involve multiple participants and non-MarkLogic Server resources, they may take longer than usual. A slow XA transaction may cause other queries on the same App Server to block for an unacceptably long time.

You may set the "multi-version concurrency control" App Server configuration parameter to `nonblocking` to minimize blocking, at the cost of less timely results. For details, see Reducing Blocking with Multi-Version Concurrency Control in the *Application Developer's Guide*.

3.0 Downloading and Using the XCC API

The XCC API is available by downloading the XCC packages from developer.marklogic.com.

This chapter describes the basics of setting up your XCC environment, and includes the following sections:

- XCC/J Java Packages
- XCC/.NET C# Packages

For a description of the sample applications included with XCC, see “Using the Sample Applications” on page 27.

3.1 XCC/J Java Packages

The Java distribution of XCC has the following directory structure:

Document or Directory	Description
`docs/`	Includes the Javadoc for XCC in both expanded HTML and compressed zip format.
`lib/`	Contains the `marklogic-xcc-6.0.x.jar` file, which is the XCC libraries, and the `marklogic-xcc-examples-6.0.x.jar` file, which has the compiled versions of the sample applications. Note that the name of the XCC jar file has the version number encoded.
`src/`	Includes the source code for the sample applications.
`Readme.txt`	Includes the version number and any last-minute updates not included in the documentation.

3.2 XCC/.NET C# Packages

The .NET distribution of XCC has the following directory structure:

Document or Directory	Description
`dll/`	Contains the `MarklogicXcc.dll` file, which is the XCC libraries, as well as some supporting DLLs.
`docs/`	Includes the .NET API documentation for XCC in both expanded HTML and compressed zip format.
`src/`	Includes the source code for the sample applications.
`Readme.txt`	Includes the version number and any last-minute updates not included in the documentation.

4.0 Using the Sample Applications

The XCC packages contain a number of sample applications. Each sample application is provided along with its source code, giving you a starting point for creating your own applications. This chapter describes the sample applications and contains the following sections:

- Setting Up Your Environment
- Sample Applications

4.1 Setting Up Your Environment

Before running the sample applications, be sure to set up the necessary environment to run the application. This section has the following parts:

- Setting Up Your MarkLogic Server Environment
- Setting Up Your Java Environment
- Setting Up Your .NET Environment

4.1.1 Setting Up Your MarkLogic Server Environment

Before you run the sample applications, complete the following steps:

1. Install MarkLogic Server, or have a MarkLogic Server installation to which you can connect. For details on installing MarkLogic Server, see the *Installation Guide*.

2. Create and configure an XDBC Server using the Admin Interface. See the *Administrator's Guide* for details on how to create and configure an XDBC Server.

3. Configure a user for the XDBC Server you created. For example, add a user to the security database with the username as `user` and the password as `pass`. See the *Administrator's Guide* for details on adding a user to the security database.

4.1.2 Setting Up Your Java Environment

If you are using XCC/J, you must have Java installed on your client machine. Additionally, you will need the following set up to run the sample applications:

- Set your `JAVA_HOME` environment variable, if it is not already set. For example, if you are running a Windows machine, set `JAVA_HOME` in a command window as in the following example:

  ```
  set JAVA_HOME=c:\Sun\SDK\jdk
  ```

  Substitute the directory in which Java is installed in your environment.

- Set your `CLASSPATH` environment variable correctly, or use the `-classpath` option to pass the appropriate classpath on the command line. Make sure to use the correct name for the `marklogic-xcc-6.0.x.jar` file in your `CLASSPATH`, as the name corresponds to the service release version number.

4.1.3 Setting Up Your .NET Environment

If you are using XCC/.NET, you must have .NET 2.0 Service Pack 1 (SP1), or later installed on your client machine.

4.2 Sample Applications

The source code and API documentation for the sample applications are includes in the XCC packages.

The Java distribution of XCC includes `marklogic-xcc-jar-6.0.x.jar` and `marklogic-xcc-examples-6.0.x.jar` files. The commands to launch the sample programs in this section assume you have renamed these jar files to `xcc.jar` and `xccexamples.jar`, respectively. The commands to launch the sample programs also assume the XCC Java distrubtion is installed in `XCC_HOME`.

The sample applications are as follows:

Sample	Description
ContentFetcher	This class fetches documents from the conentbase and writes their serialized contents to a provided OutputStream.
ContentLoader	This program accepts a server URI (in the format expected by `ContentSourceFactory.newContentSource(java.net.URI)`) and one or more file pathnames of documents to load.
DynamicContentStream	This program demonstrates inserting unbuffered, chunkable dynamic content into the database without spawning a new thread.
HelloSecureWorld	This simple program prints out the string “Hello World”, using SSL/TLS to connect to MarkLogic Server.

Sample	Description
HelloWorld	This simple program prints out the string `"Hello World"`.
ModuleRunner	This simple program invokes a named XQuery module on the server and return the result.
OutputStreamInserter	This program demonstrates inserting unbuffered dynamic content into the database by spawning a new thread to write the data.
SimpleQueryRunner	This is a very simple class that will submit an XQuery string to the server and return the result.
XA	This program demonstrates using MarkLogic Server in a distributed XA transaction, using JBoss as the Transaction Manager.

4.2.1 ContentFetcher

This program fetches a document from MarkLogic Server and serializes its contents. You can serialize the contents to the standard output (display it on the screen) or to a file using the `-o` option. The following is a sample command to run the `ModuleRunner` class:

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar"
     com.marklogic.xcc.examples.ContentFetcher
     xcc://username:password@localhost:8021
     /mydocs/hello.xml -o myHelloFile.xml
```

This sends the contents of the document at `/mydocs/hello.xml` to the file `myHelloFile.xml` (in the same directory in which the command is run). It connects to the default database of the XDBC Server listening on port 8021 of the local machine, using the credentials `username` and `password` to authenticate the connection.

4.2.2 ContentLoader

This program loads the specified document in the database. It loads the file with a URI equal to the fully-qualified pathname of the file. The following is a sample command to run the `ContentLoader` class:

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar"
     com.marklogic.xcc.examples.ContentLoader
     xcc://username:password@localhost:8021 hello.xml
```

This loads the file at `hello.xml` to a document with the fully-qualified pathname of `hello.xml` (for example, `c:\xcc\examples\hello.xml`). It loads it into the default database of the XDBC Server listening on port 8021 of the local machine, using the credentials `username` and `password` to authenticate the connection.

4.2.3 DynamicContentStream

This program demonstrates inserting dynamic content without spawning a new thread, by using `ContentFactory.newUnBufferedContent`. The following is a sample command to run the `DynamicContentStream` program:

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar"
     com.marklogic.xcc.examples.DynamicContentStream
     xcc://username:password@localhost:8021
     /any/valid/docURI
```

This inserts a new document at `/any/valid/docURI` in the contentbase described by the XDBC App Server URI `xcc://username:password@localhost:8021`. For demonstration purposes, the document contents are generated dynamically by the sample application.

4.2.4 HelloSecureWorld

This program runs a query on MarkLogic Server that returns the string `"Hello World"`, using an SSL/TLS connection to MarkLogic Server.

This example can load a specified key store of trusted signing authorities, use the Java default key store, or use a stub that accepts any server certificate. It can also load client certificates from a specified key store, or connect without a certificate.

The server certificate command line parameter may be any of the following:

- the path to a Java Key Store containing trusted signing authorities
- `DEFAULT` - use the Java default cacerts
- `ANY` - accept any server certificate

Optionally, you may also specify the path to a Java Key Store containing client certificates, along with its passphrase.

The following is a sample command to run the `HelloSecureWorld` class, accepting any certificate and no client authentication (the XDBC App Server needs `ssl require client certificate` set to `false` for this configuration).

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar"
     com.marklogic.xcc.examples.HelloSecureWorld
     xccs://username:password@localhost:8021 ANY
```

Note: When FIPS mode is enabled (which is the default), you need to use strong client ciphers, and you need to download and install the Java Cryptography Extension (JCE) Unlimited Strength package (http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html).

4.2.5 HelloWorld

This program runs a query on MarkLogic Server that returns the string `"Hello World"`. The following is a sample command to run the `HelloWorld` class:

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar"
     com.marklogic.xcc.examples.HelloWorld
     xcc://username:password@localhost:8021
```

4.2.6 ModuleRunner

This program allows you to invoke a module on the server. The module must exist under the XDBC server root, either in the database (when a modules database is configured) or on the filesystem (when the filesystem is configured for modules). The following is a sample command to run the `ModuleRunner` class:

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar"
     com.marklogic.xcc.examples.ModuleRunner
     xcc://username:password@localhost:8021 hello.xqy
```

This invokes the module named `hello.xqy`. The request is submitted to the XDBC Server running on the local machine at port 8021, using the credentials `username` and `password` to authenticate the connection. The module path is resolved relative to the XDBC Server root.

4.2.7 OutputStreamInserter

This program demonstrates inserting dynamic content by spawning a new thread to write data to the receiving end of an input stream. For an alternative method of inserting dynamically generated streamed content, see the example “DynamicContentStream” on page 30.

The following is a sample command to run the `OutputStreamInserter` program:

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar"
     com.marklogic.xcc.examples.OutputStreamInserter
     xcc://username:password@localhost:8021
     /any/valid/docURI
```

This inserts a new document at `/any/valid/docURI` in the contentbase described by the XDBC App Server URI `xcc://username:password@localhost:8021`. For demonstration purposes, the document contents are generated dynamically by the sample application.

4.2.8 SimpleQueryRunner

This program allows you to store XQuery in a file and then submit the XQuery to MarkLogic Server. The following is a sample command to run the `SimpleQueryRunner` class:

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar"
     com.marklogic.xcc.examples.SimpleQueryRunner
     xcc://username:password@localhost:8021 hello.xqy
```

This submits the contents of the `hello.xqy` file to a MarkLogic Server XDBC Server running on the local machine at port 8021, using the credentials `username` and `password` to authenticate the connection.

4.2.9 XA

This program demonstrates using MarkLogic Server in a distributed XA transaction. The sample uses JBoss as a Transaction Manager, with two MarkLogic Server clusters participating in the distributed transaction.

The sample uses libraries from JBossTS. Download and install JBossTS 4.15.0 or later from the following location. The JBoss Application Server is not needed.

http://www.jboss.org/jbosstm

Include the following libraries from the JBossTS package on your classpath. `JBOSS_HOME` is the directory where JBossTS is installed.

- `JBOSSTS_HOME/lib/jbossjta.jar`
- `JBOSSTS_HOME/lib/ext/jboss-transaction-api_1.1_spec.jar` (or other JTA implementation)
- `JBOSSTS_HOME/lib/ext/jboss-logging.jar`

The participating MarkLogic Server clusters may simply be two XDBC App Servers on the same instance, serving different databases.

The following is a sample command to run the `ModuleRunner` class. Change `XCC_HOME`, `JBOSSTS_HOME`, and the two content base URIs to match your installation.

```
java -classpath "XCC_HOME/lib/xcc.jar;XCC_HOME/lib/xccexamples.jar;
     JBOSSTS_HOME/lib/jbossjta.jar;
     JBOSSTS_HOME/lib/ext/jboss-transaction-api_1.1_spec.jar;
     JBOSSTS_HOME/lib/ext/jboss-logging.jar"
     com.marklogic.xcc.examples.XA
     xcc://username1:password1@host1:port1/contentbase1
     xcc://username2:password2@host2:port2/contentbase2
```

The sample program enlists each contentbase as a resource with the JBoss Transaction Manager, and then inserts a document in each contentbase, as part of a single global transaction. The output from xdmp:host-status relevant to each branch's transaction is printed out. This information includes the global transaction id and branch qualifer, as well as the local transaction id:

```
<transaction xmlns="http://marklogic.com/xdmp/status/host">
  <transaction-id>750821115632601886</transaction-id>
  <host-id>8814043795788656336</host-id>
  <server-id>4366002345564888063</server-id>
  <xid format-id="131076" xmlns="http://marklogic.com/xdmp/xa">
    <global-transaction-id>...825D0000000931</global-transaction-id>
    <branch-qualifier>...825D0000000A</branch-qualifier>
  </xid>
  <name/>
  <mode>update</mode>
  <timestamp>0</timestamp>
  <state>active</state>
  <database>2901782035623219290</database>
  <canceled>false</canceled>
  ...
</transaction>
<transaction xmlns="http://marklogic.com/xdmp/status/host">
  <transaction-id>4949312806261581854</transaction-id>
  <host-id>8814043795788656336</host-id>
  <server-id>7579943212553445602</server-id>
  <xid format-id="131076" xmlns="http://marklogic.com/xdmp/xa">
    <global-transaction-id>0...825D0000000931</global-transaction-id>
    <branch-qualifier>...825D0000000D</branch-qualifier>
  </xid>
  <name/>
  <mode>update</mode>
  <timestamp>0</timestamp>
  <state>active</state>
  <database>2852559629722654718</database>
  <canceled>false</canceled>
  ...
</transaction>
```

For details on use XA with MarkLogic Server, see "Multi-Statement Transactions" on page 14.

5.0 Technical Support

MarkLogic provides technical support according to the terms detailed in your Software License Agreement or End User License Agreement. For evaluation licenses, MarkLogic may provide support on an “as possible” basis.

For customers with a support contract, we invite you to visit our support website at http://support.marklogic.com to access information on known and fixed issues.

For complete product documentation, the latest product release downloads, and other useful information for developers, visit our developer site at http://developer.marklogic.com.

If you have questions or comments, you may contact MarkLogic Technical Support at the following email address:

support@marklogic.com

If reporting a query evaluation problem, please be sure to include the sample XQuery code.

Combined Product Notices

MarkLogic Server 6.0 and supporting products.
Last updated: August 19, 2013

COPYRIGHT

Copyright © 2013 MarkLogic Corporation. All rights reserved.
This technology is protected by U.S. Patent No. 7,127,469B2, U.S. Patent No. 7,171,404B2, U.S. Patent No. 7,756,858 B2, and U.S. Patent No 7,962,474 B2.

The MarkLogic software is protected by United States and international copyright laws, and incorporates certain third party libraries and components which are subject to the attributions, terms, conditions and disclaimers set forth below.

1. Contains SAP BusinessObjects Text Analysis XI from SAP AG. Copyright © 1996-2011. All rights reserved.

2. Highslide Software from Highslide Software Torstein Honsi. All Highslide Software is protected by local and international copyright laws. All rights reserved.

3. Icons developed by Yusuke Kamiyamane. Copyright © 2011 Yusuke Kamiyamane. All rights reserved. Icons are licensed subject to http://creativecommons.org/licenses/by/3.0/legalcode.

4. Antenna House OfficeHTML Copyright © 2000-2008 Antenna House, Inc. All rights reserved.

5. Argus Copyright ©1999-2008 Iceni Technology Ltd. All rights reserved.

6. Rosette Linguistics Platform 6.5.2/6.5.3 from Basis Technology Corporation, Copyright © 2004-2008 Basis Technology Corporation. All rights reserved.

7. ISYS Search. Copyright © 2011 ISYS™ Search Software, Inc. All rights reserved.

8. Software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/) Copyright © 1995-1998 Eric Young (eay@cryptsoft.com). All rights reserved. Copyright © 1998-2011 The OpenSSL Project. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)" 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org. 5. Products derived from this

software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project. 6. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)" THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com). Additional terms may apply to the foregoing software as further set forth at http://www.openssl.org/source/license.html

9. ICU v. 4.2.1 available from http://site.icu-project.org// Copyright © 1995-2011 International Business Machines Corporation and others. All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

10. Tidy available at http://www.w3.org/People/Raggett/tidy/. Copyright © 1998-2008 World Wide Web Consortium (Massachusetts Institute of Technology, European Research Consortium for Informatics and Mathematics, Keio University). All Rights Reserved.

11. 'zlib' general purpose compression library v. 1.2.3. Copyright © 1995-2010 Jean-loup Gailly and Mark Adler. http://www.zlib.net/zlib_license.html

12. RSA Data Security, Inc. MD5 Message-Digest Algorithm, copyright © 1991-2, RSA Data Security, Inc. Created 1991. All rights reserved. License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function. License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work. RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind. These notices must be retained in any copies of any part of this documentation and/or software.

13. TRE software, available at http://laurikari.net/tre/. Copyright © 2001-2009 Ville Laurikari <vl@iki.fi>. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: (1) Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. (2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER AND CONTRIBUTORS``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

14. Eigen Library, version 2.0.10, available at http://eigen.tuxfamily.org/index.php?title=Main_Page. Eigen Library is free software: you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. Eigen Library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with Eigen Library. If not, see: http://www.gnu.org/copyleft/lesser.html

15. FunctX XQuery Library v.1.0 available at http://www.xqueryfunctions.com/xq/, Copyright © Datypic (Priscilla Walmsley). You can redistribute such library and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the

hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License as available at http://www.gnu.org/licenses/lgpl-2.1.html#SEC4 for more details.

16. DeployJava.js v. 1.8, Copyright © 2006, 2011, Oracle and/or its affiliates. All rights reserved. ORACLE PROPRIETARY/CONFIDENTIAL. Use of the foregoing library is subject to the following license terms. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of Oracle nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

17. YUI libraries v.2.8 available at http://developer.yahoo.com/yui/. Copyright © 2011 Yahoo! Inc. All rights reserved. Use of the foregoing libraries is subject to the following license terms. Redistribution and use of this software in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of Yahoo! Inc. nor the names of YUI's contributors may be used to endorse or promote products derived from this software without specific prior written permission of Yahoo! Inc. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING

NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. Additional notices related to components included in the foregoing library are also available at the following URL: http://yuilibrary.com/license/

18. XSLTForms libraries available at http://sourceforge.net/projects/xsltforms/ . You can redistribute such libraries and/or modify them under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. These libraries are distributed in the hope that they will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License as available at http://www.gnu.org/licenses/lgpl-2.1.html#SEC4 for more details.

19. jQuery v.1.7 libraries available at http://jquery.com/ and jQuery UI v.1.8.2 libraries available at http://jqueryui.com/. Copyright © 2011 John Resig. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OFMERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE ANDNONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BELIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTIONOF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTIONWITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

20. JSON v. 2 libraries available at http://www.json.org/json2.js. Copyright © 2002 JSON.org. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. The Software shall be used for Good, not Evil. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

21. CodeMirror libraries available at http://codemirror.net/. Copyright © 2011 by Marijn Haverbeke marijnh@gmail.com. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

22. History.js available https://github.com/balupton/History.js/ Copyright © 2011, Benjamin Arthur Lupton. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of Benjamin Arthur Lupton nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

23. Sample content provided for use with the software is copyrighted by the respective authors as contributed by each of them to Wikipedia pursuant to the Creative Commons Attribution-ShareAlike License found at http://creativecommons.org/licenses/by-sa/3.0/.

24. Visual Studio Tools for Office 3.0 (Microsoft VSTO 3.0) licensed under Microsoft Public License Microsoft Public License (Ms-PL) found at http://msdn.microsoft.com/en-us/library/ff647676.aspx.

25. The MarkLogic Connector for Hadoop contains Apache Jakarta Commons Modeler. Copyright 2001-2007 The Apache Software Foundation. Licensed under the Apache License, Version 2.0 (the "License"). You may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

26. jemalloc http://www.canonware.com/jemalloc/index.html, released under the terms of the following BSD-derived licenses: Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice(s), this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice(s), this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER(S) ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER(S) BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. Portions of the software are copyright (C) 2002-2013 Jason Evans, and portions copyright (C) 2009-2013 Facebook, Inc. and copyright (C) 2007-2012 Mozilla Foundation. Neither the name of Facebook, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. All rights reserved.

27. Gziphelper; version: 1.0, Feb, 2003, authored by Gao Dasheng, is distributed subject to the following terms: This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software. Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions: (1.) the origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required; (2) altered source versions must be plainly marked as such, and must not be misrepresented as being the original software; (3) this notice may not be removed or altered from any source distribution.

28. Some features of MarkLogic allow you to develop and use Google Maps. Review Google Map's privacy policy at http://www.google.com/policies/privacy/, legal notices at http://maps.google.com/help/legalnotices_maps.html, and terms of use at http://maps.google.com/help/terms_maps.html.

29. commons-codec, HttpComponents, commons-logging, and Google Maps Utility are used under the Licensed under the Apache License, Version 2.0 (the "License"). You may not use these files except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

30. Mapstraction. Copyright (c) 2012, Mapstraction. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

31. logback-classic and logback-core are provided in object code form only under the Eclipse Public License - v 1.0 available at http://www.eclipse.org/legal/epl-v10.html. Copyright (C) 1999-2012, QOS.ch. All rights reserved. Source code for the Program is available http://logback.qos.ch/.

32. Simple Logging Facade for Java (SLF4J). Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

33. cURL v. 7.22.0 available at http://curl.haxx.se/. Copyright © 1996 - 2011, Daniel Stenberg, <daniel@haxx.se>. All rights reserved. Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

34. heatmap.js distributed subject to the "THE BEER-WARE LICENSE" (Revision 42): <phk@FreeBSD.ORG> wrote this file. As long as you retain this notice you can do whatever you want with this stuff. If we meet some day, and you think this stuff is worth it, you can buy me a beer in return Poul-Henning Kamp.

35. Except for components that are otherwise subject to other open source licenses, the MarkLogic ODBC Driver is released under the GNU Lesser General Public License as published by the Free Software Foundation, version 2.1 of the License.

36. Solr-commons-csv-3.5.0 Copyright 2005-2006 The Apache Software Foundation. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

37. Avro v. 1.7.1. Copyright © The Apache Software Foundation. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

38. Guava v. 11.0.2. Copyright © Google. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

39. Hadoop-auth v. 2.0.0. Copyright © The Apache Software Foundation. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

40. hadoop-hdfs v. 2.0. Copyright © The Apache Software Foundation. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

41. commons-lang v. 2.4, 2.5. Copyright © The Apache Software Foundation. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

42. commons-configuration v. 2.0. Copyright © The Apache Software Foundation. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

43. jackson-core-asl v. 1.8.8. Copyright © FasterXML. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

44. jackson-mapper-asl v. 1.8.8. Copyright © FasterXML. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

45. log4j v. 1.2.16. Copyright © The Apache Software Foundation. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, this file is distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

46. protobuf-java v. 2.4.0a. Copyright © Google. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of Google nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

47. xpp3 v. 1.1.3.3. Copyright © 2002 Extreme! Lab, Indiana University. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Indiana University Extreme! Lab(http://www.extreme.indiana.edu/)." Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear. The names "Indiana Univeristy" and "Indiana Univeristy Extreme! Lab" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact http://www.extreme.indiana.edu/. Products derived from this software may not use "Indiana Univeristy" name nor may "Indiana Univeristy" appear in their name, without prior written permission of the Indiana University. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS, COPYRIGHT HOLDERS OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IFADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

48. xstream v. 1.4.2. Copyright © 2003-2006, Joe Walnes. Copyright © 2006-2009, 2011 XStream Committers. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of XStream nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR

BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

TRADEMARK NOTICE

The MarkLogic name and logo are registered trademarks of MarkLogic Corporation in the United States and/or other countries. Excel and PowerPoint are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Any other names or logos included in the Software, this notice or the documentation are property of the respective trademark owners.