FioranoMQ® 9

C++(JNI) RTL Sample Applications Guide

Content

JNI-Based C++ Runtime Examples

The sample programs illustrating the use of C++ RTL for PubSub and PTP operations.

- PTP
  - RequestReply
  - SendReceive
  - HTTP
- PubSub
  - MTPubsub
  - RequestReply
  - ObjectMessages
  - HTTP
  - MultiThreading
- Unified Domain
  - CreateCF
  - Rollback
  - SendReceive
  - Transacted
- XA
  - Admin
  - XA Sample

These samples are available in %FMQ_DIR%\cpp\jni\samples directory.

PTP Samples

Reqrep

This directory contains two sample programs which illustrate JMS Request-Reply mechanism using the FioranoMQ C++ Runtime Library.

- **Requestor.cpp** - Reads strings from standard input and makes request on the queue "primaryQueue".
- **Replier.cpp** - Implements an asynchronous listener, which listens on the queue "primaryQueue", and replies to each request.

To run these samples using FioranoMQ, do the following:

1. Compile each of the source files. For convenience, compiled version of the sources is included in this directory.
2. The %FMQ_DIR%\cpp\jni\scripts directory contains a script called cppclient-build.bat which compiles the C++ programs
3. Run the Requestor by executing the Requestor.exe executable file.
4. Run the asynchronous Replier by executing the Replier.exe file.

Note: To run any of the C++ samples, please ensure that Environment variable FMQ_DIR points to Fiorano's installation directory (this defaults to Fiorano\FioranoMQ) and MessageAdaptor.dll should be available in System PATH.

This JNI based C++ runtime library uses jvm.dll (of jre 1.3) bundled with the product.

Basic

This directory contains two sample programs which illustrate basic JMS PTP functionality using the FioranoMQ C++ Runtime Library.

- **Sender.cpp** - Reads strings from standard input and sends them on the queue "primaryQueue".
- **Receiver.cpp** - Implements an asynchronous listener, which listens on the queue "primaryQueue", and prints out received messages.

To run these samples using FioranoMQ, do the following:

1. Compile each of the source files. For convenience, compiled version of the sources is included in this directory.

2. The %FMQ_DIR%\cpp\jni\scripts directory contains a script called cppclientbuild which compiles the C++ programs

3. Run the Sender by executing the Sender.exe executable file.

4. Run the asynchronous receiver by executing the Receiver.exe file.

Note: To run any of the C++ samples, please ensure that Environment variable FMQ_DIR points to Fiorano's installation directory (this defaults to \Fiorano\FioranoMQ\fmq).

The C++ runtime library uses the jvm.dll (of jre 1.3) bundled with the product.

HTTP

This directory contains two sample programs which illustrate the use of HTTP protocol for basic JMS ptp functionality using the FioranoMQ C++ JNI Runtime Library.

- **HttpQReceiver.cpp** - Receives messages asynchronously on primaryQueue. This program implements an asynchronous listener to listen for messages published on the queue primaryQueue.

- **HttpQSender.cpp** - Implements a client application publishing user specified data on primaryQueue. This program reads strings from standard input and publishes them on the Queue primaryQueue.

To run this sample using FioranoMQ, do the following:

1. Compile the source file. For convenience, compiled version of the sources is included in this directory. The %FMQ_DIR%\cpp\scripts directory contains a script called cppclientbuild.bat which compiles the C++ program.

2. Run the HttpQReceiver by executing the HttpQReceiver.exe executable file.

3. Run the HttpSender by executing the HttpQSender.exe executable file.

Note: To run any of the C++ samples, please ensure that environment variable FMQ_DIR points to Fiorano installation directory.

PubSub Samples

Basic

This directory contains the following sample programs, which illustrate basic JMS Publish/Subscribe functionality, using the FioranoMQ C++ Runtime Library.

- **Publisher.cpp** - Reads strings from standard input and publishes them on primarytopic.
- **Subscriber.cpp** - Implements an asynchronous listener, which listens on primarytopic, and prints the received messages.
- **PubSub.cpp** - Reads strings from standard input and publishes them on the primarytopic. Implements an asynchronous listener, which listens on primary-topic, and prints the received messages.

To run these samples using FioranoMQ, perform the following steps:

1. Compile each of the source files. For convenience, compiled version of the sources is included in this directory. The %FMQ_DIR%\cpp\jni\scripts directory contains a script called cppclient- build.bat which compiles the C++ programs
2. Run the Publisher by executing the Publisher.exe executable file.
3. Run the asynchronous subscriber by executing the Subscriber.exe file.
4. Run the publisher and the asynchronous subscriber by executing the Pub- Sub.exe file.

To run any of the C++ samples, ensure that the environment variable FMQ_DIR points to Fiorano installation directory (this defaults to \Fiorano\FioranoMQ) and MessageAdaptor.dll is available in System PATH.

MT

This directory contains a single program, which illustrates basic JMS Publish/Sub-scribe functionality using the MultiThreaded FioranoMQ C++ Runtime Library.

- **MtPubSub.cpp** - Reads strings from standard input and publishes them on the topic "primarytopic". The Publisher is created on the user Thread and the subscriber listens for data on the main Thread.

To run these samples using FioranoMQ, perform the following steps:

1. Compile each of the source files. For convenience, compiled version of the sources is included in this directory.
2. The %FMQ_DIR%\cpp\jni\scripts directory contains a script called cppclient-build.bat which compiles the C++ programs.

3. Run the sample by executing the PubSub.exe file.

4. To run any of the C++ samples, please ensure that Environment variable FMQ_DIR points to Fiorano installation directory (this defaults to \Fiorano\FioranoMQ) and MessageAdaptor.dll is available in System PATH.

The C++ runtime library uses the jvm.dll (of jre 1.3) bundled with the product.

ObjectMessages

Basic

This directory contains two programs:

- **Publisher.java** - The publisher publishes Integer objects as ObjectMessages
- **ObjectSubscriber.cpp** - Receives messages published by publisher.

To run these samples using FioranoMQ, do the following:

1. 1. Compile each of the source files. For convenience, compiled version of the sources are included in this directory.
   On Windows based machines, %FMQ_DIR%\cpp\jni\scripts directory contains a script called ccpclientbuild.bat which compiles the C++ programs. On Unix based machines, %FMQ_DIR%\cpp\jni\scripts directory contains a script called ccpclientbuild.sh which compiles the C++ programs.

2. Run the Replier.

3. Run the Publisher.

To run any of the C++ samples, ensure that:

1. Environment variable FMQ_DIR points to Fiorano installation directory (this defaults to \Fiorano\FioranoMQ on Windows based systems).

2. MessageAdaptor.dll is available in System PATH on Windows based systems.

3. MessageAdaptor is available in System LD_LIBRARY_PATH on Solaris based systems.

4. The C++ runtime library uses the jvm.dll (of jre 1.3) bundled with the product on the Windows based systems. On Solaris system, ensure JRE/JDK libraries are available on LD_LIBRARY_PATH

UDT

This directory contains two programs: ObjectSubscriber.cpp and ObjectPub-lisher.cpp.

The publisher publishes Employee objects as ObjectMessages which are received by the C++ subscriber and unwrapped. The C++ Subscriber then prints the data received

To run these samples using FioranoMQ, do the following:

1. Compile each of the source files. For convenience, compiled version of the sources are included in this directory. The %FMQ_DIR%\cpp\jni\scriptsdirec-tory contains a script called ccpclientbuild.bat which compiles the C++ programs

2. Define Environment Variable called USR_DIR to point to the directory where the User Defined Types [UDT] (Here the Employee class) lie. If this is not set then both the Publisher and Subscriber would fail.

3. Run the replier by executing the ObjectSubscriber.exe file.

4. Run the Publisher.

To run any of the C++ samples, ensure that:

1. Environment variable FMQ_DIR points to Fiorano's installation directory. (this defaults to \Fiorano\FioranoMQ on Windows based systems).

2. MessageAdaptor.dll is available in System PATH on Windows based systems.

3. MessageAdaptor is available in System LD_LIBRARY_PATH on Solaris based systems.

4. The C++ runtime library uses the jvm.dll (of jre 1.3) bundled with the product on the Windows based systems. On Solaris system, please ensure JRE/JDK libraries are available on LD_LIBRARY_PATH.

ReqReply

This directory contains two programs:

- **RequestPublisher.cpp -** Implements a C++ client application making requests.

- **RequestReplier.cpp** - Implements a application that listens for requests on a given Topic and replies to each request.

These programs illustrate the request/reply abstraction supplied by the JMS API using a C++ publishing App and replying Application.

To run these samples using FioranoMQ, do the following:

1. Compile each of the source files. For convenience, compiled version of the sources are included in this directory.

2. The %FMQ_DIR%\cpp\jni\scripts directory contains a script called ccpclient-build.bat which compiles the C++ programs

3. Run the replier by executing the RequestReplier.exe file.

4. Run the Publisher by executing the RequestPublisher.exe executable file.

To run any of the C++ samples, ensure that:

1. Environment variable FMQ_DIR points to Fiorano's installation directory (this defaults to \Fiorano\FioranoMQ on Windows based systems)

2. MessageAdaptor.dll is available in System PATH on Windows based systems.

3. MessageAdaptor is available in System LD_LIBRARY_PATH on Solaris based systems.

4. The C++ runtime library uses the jvm.dll (of jre 1.3) bundled with the product on the Windows based systems. On Solaris system, please ensure JRE/JDK libraries are available on LD_LIBRARY_PATH.

HTTP

This directory contains two sample programs which illustrate the use of HTTP protocol for basic JMS pubsub functionality using the FioranoMQ C++ JNI Runtime Library.

- **HttpTopicSubscriber.cpp** - Receives messages asynchronously published on "primaryTopic". This program implements an asynchronous listener to listen for messages published on "primaryTopic".

- **HttpTopicPublisher.cpp** - Implements a client application publishing user specified data on "primaryTopic". This program reads strings from standard input and publishes them on “primaryTopic”.

To run this sample using FioranoMQ, do the following:

1. Compile the source file. For convenience, compiled version of the sources are included in this directory. The %FMQ_DIR%\cpp\scripts directory contains a script called cclientbuild.bat which compiles the C++ program.

2. Run the HttpTopicSubscriber by executing the HttpTopicSubscriber.exe executable file.

3. Run the HttpTopicPublisher by executing the HttpTopicPublisher.exe executable file.

Note: To run any of the C++ samples, please ensure that environment variable FMQ_DIR points to Fiorano installation directory.

XA

Admin

The example in this directory illustrates the use of the FioranoMQ Administration APIs for creation of JMS Administered objects such as XA-enabled-Destinations and XA-ConnectionFactories.

The program "AdmintestXA" uses Administrator APIs to log in as administrator with a password, creates the following:

- XA-enabled Queues
    - RDBMSQueue1
    - RDBMSQueue2
- XA-enabled Topics
    - RDBMSTopic1
    - RDBMSTopic2
- XAQueueConnectionFactories
    - myXAQueueConnectionFactory1
    - myXAQueueConnectionFactory2
- XATopicConnectionFactories
    - myXATopicConnectionFactory1
    - myXATopicConnectionFactory2
- XAConnectionFactories (acc to JMS1.1unified domain)
    - myXAUnifiedConnectionFactory1
    - myXAUnifiedConnectionFactory2

Then this sample looks up the XA Administered objects created using the appropriate APIs.

The administrator uses the password "passwd" to login as the administrator. If you have previously set the administrator password to something other than "passwd", then the test case will have to be modified accordingly.

To run these samples using FioranoMQ, do the following:

1. Compile each of the source files. For convenience, a compiled version of the sources is included in this directory.

2. The %FMQ_DIR%\cpp\scripts directory contains a script called cppclientbuild.bat which compiles the C program.

3. Run the sample by executing the AdminTestXA.exe executable file.

Note: To run any of the C samples, please ensure that environment variable FMQ_DIR points to Fiorano's installation directory. This defaults to \Fiorano\FioranoMQ.

XA Samples

The example in this directory illustrate the XA capabilities of FioranoMQ.

- **XASampleS.cpp** - In this sample, a receiver synchronously receives 10 messages from a queue and publish these messages on a topic demonstrating the xa implementation os FioranoMQ.

- **XASample.cpp** - In this sample, a receiver asynchronously receives 10 messages from a queue and publish these messages on a topic demonstrating the usage of 2 phase protocol of distributed transaction.

- **LocalTransactions.cpp** - This sample demonstrates how a JMS application switches from the local transaction context to global transaction context. In this sample, intially a sender and publisher performs the following functions.

    - Sender sends 10 messages on a queue in a local transaction.

    - Publisher published 10 messages on a topic in a local transaction.

    - Both the local transaction are committed (it can be rolledback also), so that application can switch to global transaction context.

    - Sender sends 10 messages in a global transaction.

    - Publisher publishes 10 messages in a global transaction.

The parameters for connecting to a database can be changed by modifying the file rdbms.cfg.

To run these samples using FioranoMQ, do the following:

1. Compile each of the source files. For convenience, a compiled version of the sources is included in this directory. The %FMQ_DIR%\cpp\scripts directory contains a script called cppclientbuild.bat which compiles the C++ program.

2. Run XASample by executing XASample.exe.

3. Run XASampleS by executing XASampleS.exe.

4. Run LocalTransactions by executing LocalTransactions.exe.

Note: To run any of the C++ samples, please ensure that environment variable FMQ_DIR points to Fiorano's installation directory. This defaults to\Fiorano\FioranoMQ.