Websphere MQ Everyplace JDBC Adapter

Supportpac EP02

Version 1.0

Take Note!
Before using this report be sure to read the general information under "Notices".

First Edition, April 2003
This edition applies to Version 1.0 of Websphere MQ Everyplace JDBC Adapter and to all subsequent releases and modifications unless otherwise indicated in new editions.

Contents

Preface

Websphere MQ Everyplace is a messaging product that extends the reach of the Websphere MQ family to a wider range of distributed and mobile environments.

Written in pure Java it runs on many platforms and uses a message queuing paradigm to provide secure, robust, and assured once-only in-order delivery of messages between applications.

Websphere MQ Everyplace seamlessly interfaces with other members of the Websphere MQ family of products, and is often used to extend the reach of corporate applications and data onto wireless and small footprint devices.

For more information on the Websphere MQ Everyplace base product, please see http://www-306.ibm.com/software/integration/wmqe/ , or try the free download of the trial version of the product available at http://www-306.ibm.com/software/integration/wmqe/downloads/

Websphere MQ Everyplace has a *pluggable* architecture, allowing users to implement published interfaces, and configure the solution to use their code, rather than the standard components provided with the Websphere MQ Everyplace product.

This supportpac contains an implementation of one such interface; namely the storage adapter interface.

The code in this package is a storage adapter that provides support for reading and writing message and configuration data to a set of database tables. The adapter can also be used by bridge transmit queue listeners for their persistent storage needs.

Java database connectivity (JDBC) is used to communicate with the database.

Using this adapter with Websphere MQ Everyplace provides between 2-8 times the performance one would normally expect on a Websphere MQ Everyplace server, where application queues are accessed by remote devices. Typically a 2-3 times message throughput is obtainable.

Summary of changes

Date	Changes
April 2003	Database adapter that supports DB2
October 2005	Added functionality for Cloudscape .Now this support pac is in the form of a generic JDBC adapter

Getting started

This chapter describes how to install the Support Pac and the software you required Websphere MQ Everyplace JDBC adapter.

Prerequisites

This section describes the software you require to use this Support Pac:

- Websphere MQ Everyplace version 2.0.2.0 or above is required in order to use the Supportpac (we recommend you download the very latest version of Websphere MQ Everyplace from the Websphere MQ Everyplace download site)
- A Java virtual machine (JVM) of level 1.4.2 or above and a Java development toolkit (JDK) are needed to modify or extend the examples.
- DB2 UDB or Cloudscape database. It has been tested using DB2 UDB v8.2 and Cloudscape 10

Installation

This support pac utility is provided in a zip file format. Download the zip file and unpack it to a directory of your choosing.

The support pac contains the following files:

Filename	Content
\userguide.pdf	Documentation in adobe acrobat (PDF) format
\license2.txt	The license covering the use of his utility.
\jdbcadapter.jar	Java archive containing the JDBC adapter and example class files.
\examples\...	Tree of directories containing example source code.

Edit your classpath to include the JDBC adapter classes on your machine.
On windows this can be done using the command-line syntax:

Set CLASSPATH=<ep02_supportpac_directory>;<ep02_supportpac_directory>\jdbcadapter.jar;%CLASSPATH%

Where *<ep02_supportpac _directory>* is the folder into which you unzipped the ep02 Support Pac.

The examples directory contains the Java source code for the example programs, described later in this document.

Websphere MQ Everyplace classes also need to be on the classpath, as do JDBC classes used by the adapter. For example, DB2 ships JDBC classes in db2java.zip and cloudscape ships classes in db2jcc.jar (You need to put db2jcc_license_c.jar in your classpath in case of cloudscape).

Data Migration Warning

No tools exist to migrate existing message or registry data to this form.
Tools may or may not be written to migrate message data from the format understood by this adapter to format used by the Websphere MQ Everyplace product in the future.

Reporting problems

This support pac is not formally supported, though any feedback concerning this package can be sent to vkalangu@in.ibm.com. Problems reported via this route may or may not be fixed, depending on time available.

Future Direction

The list of databases tested with is very small. Over time we hope to improve the adapter so it can support a wider range of databases. For example, DB2 Everyplace.

Performance

Performance tests were performed to put a series of messages to a local queue. The queue was first configured using the MQeDiskFieldsAdapter, readings were taken, and then the queue was re-configured to use the MQeJDBCFieldsAdapter with both DB2, Cloudscape databases.

An example performance test is provided in examples.db2.PerformanceTest
Use the examples.db2.Configure tool to first create the queue manager. Then run the performance test.

You will see output similar to that below:

```
Milli-seconds taken for various operations using various adapters
Adaptr,Put unass,Get unass,PutGet unassurd,Put assurrd,Get assurd,Put+Get assured,
CS          20770,    10715,          31485,      21591,     14611,          36202,
DB2         12237,     7291,          19528,      13419,      9694,          23113,
Disk        46147,     3585,          49732,      61839,     29222,          91061,
Memory       1101,      962,           2063,       2023,      1051,           3074,

Messages per second rates for various operations using various adapters
Adaptr,Put unass,Get unass,PutGet unassurd,Put assurd,Get assurd,Put+Get assured,
CS          48.0,      93.0,            31.0,      46.0,      68.0,       27.0,
DB2         81.0,     137.0,            51.0,      74.0,     103.0,       43.0,
Disk        21.0,     278.0,            20.0,      16.0,      34.0,       10.0,
Memory    908.0,    1039.0,          484.0,     494.0,      51.0,       325.0
```

These figures indicate that MQe message rates are improved 2-3 times on the machine the test was run on, when using the DB2 adapter instead of the disk adapter. The only operation where this isn't true is unassured get operations, where the disk adapter is twice as fast as the JDBC adapter.
These results are fairly typical on a windows system running NTFS, though the difference between the JDBC and disk adapters appears to increase if a FAT drive is used, or if drive fragmentation degrades the drive performance. We have observed up to 8 times improvement when switching to using a JDBC adapter in such circumstances. Use of FAT is not recommended as a result on Windows systems.

When testing using multiple client queue managers, all accessing a central server queue manager, we noted a similar 2-3 times multiple in message throughput when using the JDBC adapter.

How it works

Websphere MQ Everyplace classes rely on subclasses of the com.ibm.mqe.MQeAdapter class for their storage requirements.

WMQe code uses primitives such as “write this data to this file”, “read data from a file called X”…etc.

Most WMQe queue managers are configured using the MQeDiskFieldsAdapter. When WMQe tells it to “write this data to this file” it will do just that.

The MQeAdapter class is published, so can be subclassed by user code, and the storage mechanism re-implemented to map data onto different media.

This adapter takes requests such as “write this data to this file” and maps the request to “write this data to a database record whose index is the filename, in a table whose name is derived from the filename’s directory/path/folder information”. Thus, the adapter can store data in a very simple database, and read the data back using various aspects of the “filename” passed to it from WMQe.

Mapping a filename to an entry in a database

The mapping of the filename passed to the adapter from WMQe to a record in a database is achieved by the following steps.

The directory information passed is stripped from the actual file name information passed.
The directory information is converted into a valid database table name, using a series of string substitution commands. For example, all ‘/’ (slash and special characters) characters are converted to “_’ (underscore) character so that the database table name does not contain the invalid character.

Configuration properties are discussed in more detail elsewhere in this document. Once the desired database table has been selected, the filename is then used as the primary key into a database table. Read, write and update operations can then occur freely.

Database table structure

Each database table has the following columns

- FILENAME
  The primary key of the table. VARCHAR(255)
- MSG
  The data associated with the file name. A binary data blob. The maximum size of the blob is defined by a configurable system property, described elsewhere in this document.

One table per queue would normally be created in which messages are stored.

Permissions required by this adapter

The adapter accesses the JDBC database using a number of JDBC sessions. Each session is obtained using a user-id and password. The user-id / password, Driver and read from file name db2.config, discussed elsewhere in this document.
The user-id/account used by the adapter must be granted enough permission to perform several different actions on the database.

Table creation

This adapter will create database tables if they do not already exist in the database, at the point where they are first needed in order to store some data. You might think of this as "lazy" creation of tables. Consequently the adapter needs enough permission to create tables when necessary.

Table drop

Tables are dropped by this adapter only when a queue is deleted.
Tables used to store configuration information and MQ bridge transmit queue listener state are never dropped.

Clean-up when tables are no longer needed is left as a task for the database administrator.

Row Insert/Update/Query/Remote

During normal use, once the tables have been created, records (files) held within the tables will be read, written and updated.

Properties in db.config file used to configure the adapter

Property	**Description**
Driver	Fully qualified JDBC driver name
URL	Connection URL to the database
UserName	User id to login to database
Password	Password for above user id

Samples are shown below

For DB2

```
Driver=COM.ibm.db2.jdbc.net.DB2Driver
URL=jdbc:db2://localhost/MQEDB2
UserName=uid
password=pwd
```

For Cloudscape

```
Driver=com.ibm.db2.jcc.DB2Driver
URL=jdbc:derby:net://localhost:1527/testdb;create=true;
UserName=uid
password=pwd
```

Examples

The examples.db2 package contains the following programs:

- Configure
  Demonstrates how to configure a queue manager such that the JDBC adapter is used to store configuration information, used beneath the standard system queues (SYSTEM.DEFAULT.LOCAL.QUEUE, AdminQ , AdminReplyQ, DeadLetterQ)
  A test queue is also configured, such that it uses the JDBC adapter to store its messages.
  There are no parameters to this program.
- Unconfigure
  Removes the queue manager previously created by the Configure program.
  There are no parameters to this program.
- Run
  Uses the queue manager created using the Configure program above.
  A message is put to, and got from a test queue.
  There are no parameters to this program

Notices

Trademarks and service marks