Message Queue Interface Module

The Heirloom Computing Message Queue Interface Module converts WebSphere MQ™ Message Queue Interface (MQI) calls to either Java™ Message Service (JMS) calls or WebSphere MQ™ based java calls. COBOL programs written to use the WebSphere MQ™ API can now access virtually any Messaging Platform without having to be rewritten.

WebSphere MQ™ **–** The main benefit of using the WebSphere MQ™ based java configuration rather than the JMS configuration is that an almost one-to-one mapping to the WebSphere MQ™ Messaging Platform is maintained and thus more functions are fully supported.

JMS – advantages of using the JMS configuration over the base java configuration include the ability to choose a Messaging Platform, and having greater flexibility and compatibility when running in an environment supporting JMS (i.e. JEE).

Notes: WebSphere MQ and WebSphere are trademarks of IBM, Corp. Java is a trademark of Oracle, Corp.

Configuring Elastic Cobol Message Queue Environment

Basic configuration is performed through a properties file **heirloomcomputing_mqi.properties** containing configuration information. The Elastic Cobol system searches for this properties file in the classpath at runtime.

Some of these properties can be qualified with a queue manager name and/or queue name. Qualified properties are searched before non-qualified (or lesser qualified) properties. For instance, if the module is searching for a queue named Q1, it will look for a value associated with the *com.heirloomcomputing.ecs.mqi.jms.Destination.Q1* property first and if that is not found it will look for a value associated with the *com.heirloomcomputing.ecs.mqi.jms.Destination* property.

Following properties can be used to configure the environment:

Common Properties

1. Connection Manager Implementation to use.
   com.heirloomcomputing.ecs.mqi.api.MQIConnectionManager=<classname of ConnectionManager implementation>
   - *WebSphere java classes (default if property is not specified):*
     com.heirloomcomputing.ecs.mqi.api.MQIConnectionManager=com.heirloomcomputing.ecs.mqi.wsmqbase.WSMQConnectionManager
   - *JMS implementation:*
     com.heirloomcomputing.ecs.mqi.api.MQIConnectionManager=com.heirloomcomputing.ecs.mqi.jms.JMSConnectionManager
2. Trace setting: com.heirloomcomputing.ecs.mqi.wsmqbase.trace_level=<trace level (1-5)>

JMS Only Properties

1. JMS module to use:
   com.heirloomcomputing.ecs.mqi.jms.JMSModule[.<queueManagerName>]=<classname of JMSModule implementation>

- *JMS implementation (default if property is not specified):*
com.heirloomcomputing.ecs.mqi.jms.JMSModule=com.heirloomcomputing.ecs.mqi.jms.module.StandardJMSModule
- *WebSphere MQ JMS specific implementation:*
com.heirloomcomputing.ecs.mqi.jms.JMSModule=com.heirloomcomputing.ecs.mqi.jms.module.WebSphereMQJMSModule

2. Trace setting:
com.heirloomcomputing.ecs.mqi.jms.JMSModule.trace[.<queueManagerName>]=<true | false>
   - *Example:* com.heirloomcomputing.ecs.mqi.jms.JMSModule.trace=true
3. JNDI InitialContext properties:
   - java.naming.factory.initial=com.sun.jndi.fscontext.RefFSContextFactory
   - java.naming.provider.url=file:/c:/JNDI-Directory
4. Connection Factory to use:
   com.heirloomcomputing.ecs.mqi.jms.ConnectionFactory[.<queueManagerName>]=<jms connection factory to use>
   - *Example:* com.heirloomcomputing.ecs.mqi.jms.ConnectionFactory.QCF1=jms\\Samples\\QCF1
5. The destination to use:
   com.heirloomcomputing.ecs.mqi.jms.Destination[.<queueManagerName>.<queueName> | .<queueName>]=<jms destination>
   - *Example:* com.heirloomcomputing.ecs.mqi.jms.Destination.Q1=jms\\Samples\\Q1

COBOL Routines Usage

Pointers in MQI group items are not supported. Offsets should be used instead, and must be set to specific values. Also, the group item passed must have a specific structure, depending on the function:

- **MQCONNX:**

  If not using channel data, simply pass the MQCNO group item. If using channel data, pass a group containing two groups: The MQCNO group followed by the MQCD group. Also set the MQCNO-CLIENTCONNOFFSET to either 144 or 148 (offset from start of CNO structure).

- **MQOPEN:**

  If not a distribution list, simply pass the MQOD group item. If a distribution list, pass a group containing three items: MQOD group followed by the MQOR group table followed by the MQRR group table. Also set the MQOD-OBJECTRECOFFSET to 328 or 336 (offset from start of OD structure). Also set MQOD-RESPONSERECOFFSET to MQOD-OBJECTRECOFFSET + (96* MQOD-RECSPRESENT).

- **MQPUT:**

  If not a distribution list, simply pass the MQPMO group item. If a distribution list, pass a

group containing three items: The MQPMO group followed by the MQPMR group table followed by the MQRR group table. Also set the MQPMO-PUTMSGRECOFFSET to 144 or 152 (offset from start of PMO structure). Also set MQPMO-RESPONSERECOFFSET to: MQPMO-PUTMSGRECOFFSET + (108* MQPMO-RECSPRESENT).

- **MQPUT1:**

  (See the notes for MQOPEN and MQPUT.)

JMS Currently Unsupported Functionality

1. MQINQ and MQSET
2. Message Segments
3. NAMELIST, PROCESS, Q_MGR, and CHANNEL object types
4. Distribution lists
5. MQOO_INPUT_EXCLUSIVE
6. LOCK and UNLOCK
7. Message Tokens and Accounting Tokens
8. Signals
9. SYNCPOINT_IF_PERSISTENT
10. MARK_SKIP_BACKOUT
11. MATCH_OFFSET and MATCH_MSG_TOKEN
12. Browsing JMS Topic objects
13. JMS Message objects other than TextMessage and BytesMessage