JMX

JMX Component

Available as of Camel 2.6

Standard JMX Consumer Configuration

Component allows consumers to subscribe to an mbean's Notifications. The component supports passing the Notification object directly through the Exchange or serializing it to XML according to the schema provided within this project. This is a consumer only component. Exceptions are thrown if you attempt to create a producer for it.

Maven users will need to add the following dependency to their `pom.xml` for this component:

xml <dependency> <groupId>org.apache.camel</groupId> <artifactId>camel-jmx</artifactId> <version>x.x.x</version> <!-- use the same version as your Camel core version --> </dependency>

URI Format

The component can connect to the local platform mbean server with the following URI:

jmx://platform?options

A remote mbean server url can be provided following the initial JMX scheme like so:

jmx:service:jmx:rmi:///jndi/rmi://localhost:1099/jmxrmi?options

You can append query options to the URI in the following format, ?options=value&option2=value&...

URI Options

confluenceTableSmall

Property	Required	Default	Description
format		xml	Format for the message body. Either "xml" or "raw". If xml, the notification is serialized to xml. If raw, then the raw java object is set as the body.
user			Credentials for making a remote connection.
password			Credentials for making a remote connection.
objectDomain	yes		The domain for the mbean you're connecting to.
objectName			The name key for the mbean you're connecting to. This value is mutually exclusive with the object properties that get passed. (see below)
notificationFilter			Reference to a bean that implements the `NotificationFilter`. The #ref syntax should be used to reference the bean via the Registry.
handback			Value to handback to the listener when a notification is received. This value will be put in the message header with the key "jmx.handback"
testConnectionOnStartup		true	**Camel 2.11** If true, the consumer will throw an exception when unable to establish the JMX connection upon startup. If false, the consumer will attempt to establish the JMX connection every 'x' seconds until the connection is made – where 'x' is the configured *reconnectDelay*.
reconnectOnConnectionFailure		false	**Camel 2.11** If true, the consumer will attempt to reconnect to the JMX server when any connection failure occurs. The consumer will attempt to re-establish the JMX connection every 'x' seconds until the connection is made-- where 'x' is the configured *reconnectDelay*.
reconnectDelay		10 seconds	**Camel 2.11** The number of seconds to wait before retrying creation of the initial connection or before reconnecting a lost connection.

ObjectName Construction

The URI must always have the objectDomain property. In addition, the URI must contain either objectName or one or more properties that start with "key."

Domain with Name property

When the objectName property is provided, the following constructor is used to build the ObjectName? for the mbean:

ObjectName(String domain, String key, String value)

The key value in the above will be "name" and the value will be the value of the objectName property.

Domain with Hashtable

ObjectName(String domain, Hashtable<String,String> table)

The Hashtable is constructed by extracting properties that start with "key." The properties will have the "key." prefixed stripped prior to building the Hashtable. This allows the URI to contain a variable number of properties to identify the mbean.

Example

{snippet:id=e1|lang=java|url=camel/trunk/examples/camel-example-jmx/src/main/java/org/apache/camel/example/jmx/MyRouteBuilder.java}

Full example

Monitor Type Consumer

Available as of Camel 2.8
One popular use case for JMX is creating a monitor bean to monitor an attribute on a deployed bean. This requires writing a few lines of Java code to create the JMX monitor and deploy it. As shown below:

java CounterMonitor monitor = new CounterMonitor(); monitor.addObservedObject(makeObjectName("simpleBean")); monitor.setObservedAttribute ("MonitorNumber"); monitor.setNotify(true); monitor.setInitThreshold(1); monitor.setGranularityPeriod(500); registerBean(monitor, makeObjectName ("counter")); monitor.start();

The 2.8 version introduces a new type of consumer that automatically creates and registers a monitor bean for the specified objectName and attribute. Additional endpoint attributes allow the user to specify the attribute to monitor, type of monitor to create, and any other required properties. The code snippet above is condensed into a set of endpoint properties. The consumer uses these properties to create the CounterMonitor, register it, and then subscribe to its changes. All of the JMX monitor types are supported.

Example
java from("jmx:platform?objectDomain=myDomain&objectName=simpleBean&" + "monitorType=counter&observedAttribute=MonitorNumber&initThreshold=1&" + "granularityPeriod=500").to("mock:sink");

The example above will cause a new Monitor Bean to be created and depoyed to the local mbean server that monitors the "MonitorNumber" attribute on the "simpleBean." Additional types of monitor beans and options are detailed below. The newly deployed monitor bean is automatically undeployed when the consumer is stopped.

URI Options for Monitor Type

property	type	applies to	description
monitorType	enum	all	one of counter, guage, string
observedAttribute	string	all	the attribute being observed
granualityPeriod	long	all	granularity period (in millis) for the attribute being observed. As per JMX, default is 10 seconds
initThreshold	number	counter	initial threshold value
offset	number	counter	offset value
modulus	number	counter	modulus value
differenceMode	boolean	counter, gauge	true if difference should be reported, false for actual value
notifyHigh	boolean	gauge	high notification on/off switch
notifyLow	boolean	gauge	low notification on/off switch
highThreshold	number	gauge	threshold for reporting high notification
lowThreshold	number	gauge	threshold for reporting low notificaton
notifyDiffer	boolean	string	true to fire notification when string differs
notifyMatch	boolean	string	true to fire notification when string matches
stringToCompare	string	string	string to compare against the attribute value

The monitor style consumer is only supported for the local mbean server. JMX does not currently support remote deployment of mbeans without either having the classes already remotely deployed or an adapter library on both the client and server to facilitate a proxy deployment.

Endpoint See Also

- Camel JMX