Using External Messaging Platforms from Within Productions

Version 2024.1
2024-05-16

InterSystems Corporation 1 Memorial Drive Cambridge MA 02142 www.intersystems.com

Using External Messaging Platforms from Within Productions
InterSystems IRIS Data Platform Version 2024.1 2024-05-16

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Response Center (WRC)
Tel: +1-617-621-0700
Tel: +44 (0) 844 854 2917
Email: support@InterSystems.com

Table of Contents

1

Retrieving JMS Messages from within a Production

InterSystems IRIS becomes a JMS consumer when an interoperability production includes a business service that uses the JMS inbound adapter implemented using the PEX framework (EnsLib.JMSPEX). This adapter allows the business service to retrieve messages from a JMS queue or topic. You have two options when using a production as a JMS consumer: use a built-in business service that leverages the inbound adapter or build your own business service that uses the adapter.

Important: This page describes how to use the inbound adapter and business service for JMS which InterSystems implements using the PEX framework. The associated classes are part of the EnsLib.JMSPEX package. *Do not* use classes from the legacy EnsLib.JMS package, which may be removed in future releases.

1.1 Built-in Business Service

Rather than building a custom business service that uses the inbound adapter, you can simply add EnsLib.JMSPEX.Service to the production and define the adapter properties using the Management Portal settings. As the business service retrieves messages from your JMS server at the interval determined by the **Basic Settings** > **Call Interval** setting, these messages are forwarded to another business host in the production using asynchronous requests. The business host where these requests are forwarded is determined by the **Basic Settings** > **Target Config Names** setting.

For basic information about adding a business service to a production, see Adding Business Hosts.

1.2 Adapter Details

The JMS inbound adapter is the class EnsLib.JMSPEX.InboundAdapter. It includes the following settings, all of which appear in the Management Portal as settings for the business service that uses the adapter:

- QueueOrTopicName defines the JMS queue or topic from which the consumer is receiving messages.
- ReceiveSettings (optional) a JSON string defining settings for message retrieval. The list of available settings is the same as the list of properties made available by the JMS API JMSReceiveSettings class, with each property name serving as the key.
- URL defines the URL for the JMS server.

- InitialContextFactoryName defines the name of the initial JMS context factory.
- ConnectionFactoryName defines the name of the JMS connection factory.
- ClientID defines a string to identify the production as a JMS client.
- Credentials defines the InterSystems credentials that correspond to the username and password of a JMS client. For details on creating credentials, see Defining Reusable Items for Use in Settings.

For general information about building a custom business service that uses an inbound adapter, see Defining Business Services.

1.3 See Also

- Sending Messages to JMS from a Production
- Using the JMS Messaging API (for use without a production)

2

Sending Messages to JMS from a Production

InterSystems IRIS becomes a JMS producer when an interoperability production includes a business operation which uses the JMS outbound adapter implemented using the PEX framework (EnsLib.JMSPEX). This adapter allows the business operation to send messages to a JMS queue or topic. You have two options when using a production as a JMS producer: use a built-in business operation that leverages the outbound adapter or build your own business operation that uses the adapter.

Important: This page describes how to use the outbound adapter and business operation for JMS which InterSystems implements using the PEX framework. The associated classes are part of the EnsLib.JMSPEX package. *Do not* use classes from the legacy EnsLib.JMS package, which may be removed in future releases.

2.1 Adapter Details

The JMS outbound adapter is the class EnsLib.JMSPEX.OutboundAdapter. It includes the following settings, all of which appear in the Management Portal as settings for the business operation that uses the adapter:

- URL defines the URL for the JMS server.
- InitialContextFactoryName defines the name of the initial JMS context factory.
- ConnectionFactoryName defines the name of the JMS connection factory.
- ClientID defines a string to identify the production as a JMS client.
- Credentials defines the InterSystems credentials that correspond to the username and password of a JMS client. For details on creating credentials, see Defining Reusable Items for Use in Settings.

For general information about building a custom business operation that uses an outbound adapter, see Defining Business Operations.

2.2 Send JMS Messages

The class of the messages sent by the outbound adapter to JMS is EnsLib.JMSPEX.Message. This class contains the following properties for defining the message:

- destination defines the JMS queue or topic where the producer is sending messages.
- type defines the message type (`"Text"` or `"Bytes"`).
- textBody or bytesBody defines the content of the message body. Of the two, set the property corresponding to the message type.

You can also use the properties property to attach metadata to your message. The properties property accepts a %ListOfObjects collection of %External.Messaging.JMSMessageProperty objects. See the documentation about the JMS API for further guidance on creating JMS message property objects. For general information about JMS message properties, refer to the JMS documentation.

The outbound adapter's **SendMessage()** method takes a EnsLib.JMSPEX.Message object as its sole argument, and sends that message to the JMS server. For example, a custom business operation could call the adapter's method by including the following:

```
Do ..Adapter.SendMessage(pRequest)
```

2.3 Built-in Business Operation

Rather than building a custom business operation that uses the outbound adapter, you can simply add EnsLib.JMSPEX.Operation to the production and define the necessary settings using the Management Portal. This business operation calls the adapter's **SendMessage()** method when it receives a request from another business host in the production. This request should contain a JMS message of type EnsLib.JMSPEX.Message.

For basic information about adding a business operation to a production, see Adding Business Hosts.

2.4 See Also

- Retrieving JMS Messages from within a Production
- Using the JMS Messaging API (for use without a production)

3

Retrieving Kafka Messages from within a Production

InterSystems IRIS becomes a Kafka Consumer when an interoperability production includes a business service that uses the Kafka inbound adapter. This adapter allows the business service to retrieve messages (as Kafka message objects) from a Kafka topic. You have two options when using a production as a Kafka Consumer: use a built-in business service that leverages the inbound adapter or build your own business service that uses the adapter.

3.1 Built-in Business Service

Rather than building a custom business service that uses the inbound adapter, you can simply add EnsLib.Kafka.Service to the production and define the adapter properties using the Management Portal settings. As the business service retrieves messages from Kafka at the interval determined by the **Basic Settings** > **Call Interval** setting, these messages are forwarded to another business host in the production using asynchronous requests. The business host where these requests are forwarded is determined by the **Basic Settings** > **Target Config Names** setting.

For basic information about adding a business service to a production, see Adding Business Hosts.

3.2 Adapter Details

The Kafka inbound adapter is the class EnsLib.Kafka.InboundAdapter. It includes the following settings, all of which appear in the Management Portal as settings for the business service that uses the adapter:

- Topic defines the Kafka topic from which the Consumer is retrieving messages.
- GroupID defines the ID of the Consumer's consumer group.
- ReceiveSettings (optional) a JSON string defining settings for message retrieval. The list of available settings is the same as the list of properties made available by the Kafka API KafkaReceiveSettings class, with each property name serving as the key.
- Servers defines a comma-separated list of IP address:port entries that identify servers in the Kafka broker cluster.
- Credentials defines the InterSystems credentials that correspond to the username and password of a Kafka client. For details on creating credentials, see Defining Reusable Items for Use in Settings.

- SecurityProtocol specifies the security protocol which secures connections to your Kafka broker cluster. Currently, this property supports two values:
  - `SASL_PLAINTEXT`, which performs SASL authentication of the client over an unencrypted channel.
  - `SASL_SSL`, which uses the truststore and keystore information you provide to establish an SSL/TLS connection over which SASL authentication takes place.
- SASLMechanism specifies the SASL authentication mechanism used to authenticate the Consumer using the credentials specified by Credentials. Currently, only `PLAIN` is supported.
- TrustStoreLocation (optional) specifies the file system path to the truststore which contains the certificate authority certificates necessary to validate a certificate from your Kafka broker cluster and establish an SSL/TLS connection.
- TrustStoreCredentials (optional) defines the InterSystems credentials which can be used to gain password-protected access to the truststore at the location specified by truststorelocation.
- KeyStoreLocation (optional) specifies the file system path to the keystore which contains the keys necessary to establish an SSL/TLS connection with your Kafka broker cluster.
- KeyStoreCredentials (optional) defines the InterSystems credentials which can be used to gain password-protected access to the keystore at the location specified by keystorelocation.
- KeyCredentials (optional) defines the InterSystems credentials which can be used to gain password-protected access to a private key within the keystore at the location specified by keystorelocation.

For general information about building a custom business service that uses an inbound adapter, see Defining Business Services.

3.3 See Also

- Sending Messages to Kafka from a Production
- Using the Kafka Messaging API (for use without a production)

4

Sending Messages to Kafka from a Production

InterSystems IRIS becomes a Kafka Producer when an interoperability production includes a business operation that uses the Kafka outbound adapter. This adapter allows the business operation to send messages to a Kafka topic. You have two options when using a production as a Kafka Producer: use a built-in business operation that leverages the outbound adapter or build your own business operation that uses the adapter.

4.1 Adapter Details

The Kafka outbound adapter is the class EnsLib.Kafka.OutboundAdapter. It includes the following settings, all of which appear in the Management Portal as settings for the business operation that uses the adapter:

- Servers defines a comma-separated list of IP address:port entries that identify servers in the Kafka broker cluster.
- ClientID defines the Kafka client ID of the Producer.
- Credentials defines the InterSystems credentials that correspond to the username and password of a Kafka client. For details on creating credentials, see Defining Reusable Items for Use in Settings.
- SecurityProtocol specifies the security protocol which secures connections to your Kafka broker cluster. Currently, this property supports two values:
  - `SASL_PLAINTEXT`, which performs SASL authentication of the client over an unencrypted channel.
  - `SASL_SSL`, which uses the truststore and keystore information you provide to establish an SSL/TLS connection over which SASL authentication takes place.
- SASLMechanism specifies the SASL authentication mechanism used to authenticate the Producer using the credentials specified by Credentials. Currently, only `PLAIN` is supported.
- TrustStoreLocation (optional) specifies the file system path to the truststore which contains the certificate authority certificates necessary to validate a certificate from your Kafka broker cluster and establish an SSL/TLS connection.
- TrustStoreCredentials (optional) defines the InterSystems credentials which can be used to gain password-protected access to the truststore at the location specified by truststorelocation.
- KeyStoreLocation (optional) specifies the file system path to the keystore which contains the keys necessary to establish an SSL/TLS connection with your Kafka broker cluster.

- KeyStoreCredentials (optional) defines the InterSystems credentials which can be used to gain password-protected access to the keystore at the location specified by keystorelocation.
- KeyCredentials (optional) defines the InterSystems credentials which can be used to gain password-protected access to a private key within the keystore at the location specified by keystorelocation.

For general information about building a custom business operation that uses an outbound adapter, see Defining Business Operations.

4.2 Send Kafka Messages

The class of the messages sent by the outbound adapter to Kafka is EnsLib.Kafka.Message, which contains the following properties:

- topic defines the Kafka topic where the Producer is sending messages.
- value (a %String) defines the content of the Kafka message. If value is set, you should not set binaryValue.
- binaryValue (a binary stream of arbitrary length) defines the content of the Kafka message when the length of the message exceeds the maximum length of a %String. If binaryValue is set, you should not set value.
- key defines an optional tag for the Kafka message.

The outbound adapter's **SendMessage()** method takes a EnsLib.Kafka.Message object as its sole argument, and sends that message to the Kafka server. For example, a custom business operation could call the adapter's method by including the following:

```
Do ..Adapter.SendMessage(pRequest)
```

4.3 Built-in Business Operation

Rather than building a custom business operation that uses the outbound adapter, you can simply add EnsLib.Kafka.Operation to the production and define the adapter properties using the Management Portal settings. This business operation calls the adapter's **SendMessage()** method when it receives a request from another business host in the production. This request should contain a Kafka message of type EnsLib.Kafka.Message.

For basic information about adding a business operation to a production, see Adding Business Hosts.

4.4 See Also

- Retrieving Kafka Messages from within a Production
- Using the Kafka Messaging API (for use without a production)

5

Retrieving RabbitMQ Messages from within a Production

InterSystems IRIS becomes a RabbitMQ consumer when an interoperability production includes a business service that uses the RabbitMQ inbound adapter. This adapter allows the business service to retrieve messages from a RabbitMQ queue. You have two options when using a production as a RabbitMQ consumer: use a built-in business service that leverages the inbound adapter or build your own business service that uses the adapter.

5.1 Built-in Business Service

Rather than building a custom business service that uses the inbound adapter, you can simply add EnsLib.RabbitMQ.Service to the production and define the adapter properties using the Management Portal settings. As the business service retrieves messages from RabbitMQ at the interval determined by the **Basic Settings** > **Call Interval** setting, these messages are forwarded to another business host in the production using asynchronous requests. The business host where these requests are forwarded is determined by the **Basic Settings** > **Target Config Names** setting.

For basic information about adding a business service to a production, see Adding Business Hosts.

5.2 Adapter Details

The RabbitMQ inbound adapter is the class EnsLib.RabbitMQ.InboundAdapter. It includes the following settings, all of which appear in the Management Portal as settings for the business service that uses the adapter:

- Queue Name defines the RabbitMQ queue from which the consumer is receiving messages.
- ExchangeName (optional) defines the RabbitMQ exchange which routes messages to the queue.

  Note: For more information about how RabbitMQ routes messages, refer to the RabbitMQ documentation.

- BindingKeys (optional) defines the keys which bind the queue your production is receiving messages from to the exchange you named.
- ReceiveSettings (optional) a JSON string defining settings for message retrieval. The list of available settings is the same as the list of properties made available by the RabbitMQ API RabbitMQReceiveSettings class, with each property name serving as the key.

- MQHost defines the hostname or IP address for the RabbitMQ server.
- MQPort defines the port number for communicating with RabbitMQ.
- MQVirtualHost (optional) defines the virtual hostname for RabbitMQ.
- Credentials defines the InterSystems credentials that correspond to the username and password of a RabbitMQ client. For details on creating credentials, see Defining Reusable Items for Use in Settings.

For general information about building a custom business service that uses an inbound adapter, see Defining Business Services.

5.2.1 SSL/TLS Configuration

Both of the RabbitMQ adapters (inbound and outbound) accept additional settings which allow you to configure your production to connect with RabbitMQ using SSL/TLS. These settings also appear in the Management Portal as settings for the business service that uses the adapter.

For a detailed list of these settings, refer to the instructions for configuring a RabbitMQ client using the RabbitMQ API.

5.3 See Also

- Sending Messages to RabbitMQ from a Production
- Using the RabbitMQ Messaging API (for use without a production)

6

Sending Messages to RabbitMQ from a Production

InterSystems IRIS becomes a RabbitMQ publisher when an interoperability production includes a business operation that uses the RabbitMQ outbound adapter. This adapter allows the business operation to send messages to a RabbitMQ exchange. You have two options when using a production as a RabbitMQ publisher: use a built-in business operation that leverages the outbound adapter or build your own business operation that uses the adapter.

6.1 Adapter Details

The RabbitMQ outbound adapter is the class EnsLib.RabbitMQ.OutboundAdapter. It includes the following settings, all of which appear in the Management Portal as settings for the business operation that uses the adapter:

- MQHost defines the hostname or IP address for the RabbitMQ server.
- MQPort defines the port number for communicating with RabbitMQ.
- MQVirtualHost (optional) defines the virtual hostname for RabbitMQ.
- Credentials defines the InterSystems credentials that correspond to the username and password of a Kafka client. For details on creating credentials, see Defining Reusable Items for Use in Settings.

For general information about building a custom business operation that uses an outbound adapter, see Defining Business Operations.

6.1.1 SSL/TLS Configuration

Both of the RabbitMQ adapters (inbound and outbound) accept additional settings which allow you to configure your production to connect with RabbitMQ using SSL/TLS. These settings also appear in the Management Portal as settings for the business service that uses the adapter.

For a detailed list of these settings, refer to the instructions for configuring a RabbitMQ client using the RabbitMQ API.

6.2 Send RabbitMQ Messages

The class of the messages sent by the outbound adapter to RabbitMQ is EnsLib.RabbitMQ.Message. This class contains several properties for defining the message, including the following:

- exchange defines the RabbitMQ exchange where the publisher is sending messages.
- routingKey defines the routing key which the exchange will use to route the message.
- deliveryMode defines whether the message will be treated as persistent (if the value is 2) or transient (if the value is 2).
- contentEncoding defines the encoding of the message content (such as UTF-8).
- encodedContent defines the content of the message, encoded as specified by contentEncoding.

For full descriptions of the message properties this message class makes available, refer to the RabbitMQ documentation.

The outbound adapter's **SendMessage()** method takes a EnsLib.RabbitMQ.Message object as its sole argument, and sends that message to the RabbitMQ server. For example, a custom business operation could call the adapter's method by including the following:

```
Do ..Adapter.SendMessage(pRequest)
```

6.3 Built-in Business Operation

Rather than building a custom business operation that uses the outbound adapter, you can simply add EnsLib.RabbitMQ.Operation to the production and define the adapter properties using the Management Portal settings. This business operation calls the adapter's **SendMessage()** method when it receives a request from another business host in the production. This request should contain a RabbitMQ message of type EnsLib.RabbitMQ.Message.

For basic information about adding a business operation to a production, see Adding Business Hosts.

6.4 See Also

- Retrieving RabbitMQ Messages from within a Production
- Using the RabbitMQ Messaging API (for use without a production)

7

Sending Messages to Amazon SNS from a Production

Amazon SNS is a cloud service that delivers messages from a publisher to a subscriber. You can configure your InterSystems product to be an SNS publisher by creating an interoperability production that includes a business operation that uses the SNS outbound adapter. If you are new to business operations and adapters, see Introduction to Interoperability Productions.

There are three components to a message sent to SNS: a topic, a subject, and the content of the message. Each message sent by a publisher to SNS must be associated with a specific *topic*. SNS pushes messages to subscribers who have subscribed to a particular topic. A message sent to SNS can also include a *subject*, which SNS uses as the Subject line when the message is distributed to subscribers as an email.

7.1 Adapter Details

The SNS outbound adapter is EnsLib.AmazonSNS.OutboundAdapter. If you are using the built-in business operation, you do not need to know about the details of this outbound adapter — the business operation takes care of it. But if you are writing a custom business operation, it calls the adapter's **Publish()** method to send messages to SNS. To call this method from a custom business operation, you could add the following:

ObjectScript

```
Set tSC = ..Adapter.Publish(..ARNTopic, request.Message, ..Subject)
```

where:

- ARNTopic is a property of the business operation that defines the SNS topic. Required.
- request.Message is the property of a production request object that contains the message sent to SNS. Required.
- Subject is a property of the business operation that defines the SNS subject. Optional.

You can look at the source code of **EnsLib.AmazonSNS.BusinessOperation** for an example of how this works. For details about creating a custom business operation, see Defining Business Operations.

Note: The SNS outbound adapter was developed using the InterSystems PEX framework, so the ObjectScript source code for the adapter looks different than other adapters. For example, the adapter methods are actually wrappers for methods written in a Java PEX component.

7.2 Built-in Business Operation

InterSystems provides a built-in business operation that can be used to publish messages to SNS without needing to write custom code. Since it uses the SNS outbound adapter, this built-in business operation is all you need to turn your production into an SNS publisher. Simply add EnsLib.AmazonSNS.BusinessOperation to your production and configure your other business hosts to send EnsLib.AmazonSNS.PublishRequest requests to this business operation. For information about adding a business operation to a production, see Adding Business Hosts.

The EnsLib.AmazonSNS.BusinessOperation business operation includes a property ARNTopic that is used to associate messages with an SNS topic. You can use the **SNS** > **ARNTopic** setting in the Management Portal to specify the topic.

You can use the **SNS** > **Subject** setting to give messages sent by the business operation a subject. Defining a subject is optional.

While the topic and subject of a message are defined in the business operation, the actual content of a message is sent to the business operation in a EnsLib.AmazonSNS.PublishRequest request, so configure your production to send this request object from another business host.

7.3 General AWS Settings

The SNS outbound adapter extends a common adapter class that includes general AWS properties. When you add a business operation that uses the outbound adapter to a production, these AWS properties can be set using the **AWS** settings for the business operation in the Management Portal.

- **CredentialsFile** — If blank, Amazon uses the default credential provider chain to obtain the credentials needed to access SNS. If you prefer to use an AWS credential file, enter its pathname.
- **Region** — Identifies the AWS region that you want to access. For a list of regions, see Amazon Regions, Availability Zones, and Local Zones.

7.4 See Also

- Using the Amazon SNS Messaging API (for use without a production)

8

Retrieving Amazon SQS Messages from within a Production

InterSystems IRIS becomes a RabbitMQ consumer when an interoperability production includes a business service that uses the RabbitMQ inbound adapter. This adapter allows the business service to retrieve messages from a RabbitMQ queue. You have two options when using a production as a RabbitMQ consumer: use a built-in business service that leverages the inbound adapter or build your own business service that uses the adapter.

8.1 Built-in Business Service

Rather than building a custom business service that uses the inbound adapter, you can simply add EnsLib.AmazonSQS.Service to the production and define the adapter properties using the Management Portal settings. As the business service retrieves messages from Amazon SQS at the interval determined by the **Basic Settings** > **Call Interval** setting, these messages are forwarded to another business host in the production using asynchronous requests. The business host where these requests are forwarded is determined by the **Basic Settings** > **Target Config Names** setting.

For basic information about adding a business service to a production, see Adding Business Hosts.

8.2 Adapter Details

The Amazon SQS inbound adapter is the class EnsLib.AmazonSQS.InboundAdapter. It includes the following settings, all of which appear in the Management Portal as settings for the business service that uses the adapter:

- Queue defines the Amazon SQS queue from which the consumer is receiving messages.
- DeleteAfterReceive determines whether the message is deleted from the queue after the production receives it.
- ReceiveSettings (optional) a JSON string defining settings for message retrieval. The list of available settings is the same as the list of properties made available by the Amazon SQS API SQSReceiveSettings class, with each property name serving as the key.

For general information about building a custom business service that uses an inbound adapter, see Defining Business Services.

8.3 General AWS Settings

The Amazon SQS inbound adapter extends a common adapter class that includes general AWS properties. When you add a business service that uses the inbound adapter to a production, these AWS properties can be set using the **AWS** settings for the business service in the Management Portal.

- **CredentialsFile** — If blank, Amazon uses the default credential provider chain to obtain the credentials needed to access SQS. If you prefer to use an AWS credential file, enter its pathname.
- **Region** — Identifies the AWS region that you want to access. For a list of regions, see Amazon Regions, Availability Zones, and Local Zones.

8.4 See Also

- Sending Messages to Amazon SQS from a Production
- Using the Amazon SQS Messaging API (for use without a production)

9

Sending Messages to Amazon SQS from a Production

InterSystems IRIS becomes an Amazon SQS producer when an interoperability production includes a business operation which uses the Amazon SQS outbound adapter. This adapter allows the business operation to send messages to an Amazon SQS queue. You have two options when using a production as a Amazon SQS producer: use a built-in business operation that leverages the outbound adapter or build your own business operation that uses the adapter.

9.1 Adapter Details

The Amazon SQS outbound adapter is the class EnsLib.AmazonSQS.OutboundAdapter.

For general information about building a custom business operation that uses an outbound adapter, see Defining Business Operations.

9.2 Send Amazon SQS Messages

The class of the messages sent by the outbound adapter to Amazon SQS is EnsLib.AmazonSQS.Message. This class contains several properties for defining the message, including the following:

- queue defines the Amazon SQS queue where the producer is sending messages.
- body defines the content of the message

For full descriptions of the message identifiers this class makes available through message object properties, refer to the Amazon SQS documentation.

You can also use the messageAttributes property to specify custom metadata for your message. The messageAttributes property accepts a %ListOfObjects collection of %External.Messaging.SQSMessageAttribute objects. See the documentation for the Amazon SQS API for further guidance on creating SQS message attribute objects. For general information about the use of message attributes to attach custom metadata, refer to the Amazon SQS documentation.

The outbound adapter's **SendMessage()** method takes a EnsLib.AmazonSQS.Message object as its sole argument, and sends that message to the Amazon SQS server. For example, a custom business operation could call the adapter's method by including the following:

```
Do ..Adapter.SendMessage(pRequest)
```

9.3 Built-in Business Operation

Rather than building a custom business operation that uses the outbound adapter, you can simply add EnsLib.AmazonSQS.Operation to the production and define the necessary settings using the Management Portal. This business operation calls the adapter's **SendMessage()** method when it receives a request from another business host in the production. This request should contain an Amazon SQS message of type EnsLib.AmazonSQS.Message.

For basic information about adding a business operation to a production, see Adding Business Hosts.

9.4 General AWS Settings

The SQS outbound adapter extends a common adapter class that includes general AWS properties. When you add a business operation that uses the outbound adapter to a production, these AWS properties can be set using the **AWS** settings for the business operation in the Management Portal.

- **CredentialsFile** — If blank, Amazon uses the default credential provider chain to obtain the credentials needed to access SQS. If you prefer to use an AWS credential file, enter its pathname.
- **Region** — Identifies the AWS region that you want to access. For a list of regions, see Amazon Regions, Availability Zones, and Local Zones.

9.5 See Also

- Retrieving Amazon SQS Messages from within a Production
- Using the Amazon SQS Messaging API (for use without a production)