Progress FUSE

FUSE™ Integration Designer

FUSE™ Integration Designer User Guide

Version 1.2
May 2009

PROGRESS SOFTWARE

FUSE™ Integration Designer User Guide

Progress Software

Version 1.2

Publication date 23 Jul 2009

Legal Notices

Table of Contents

List of Tables

List of Examples

Chapter 1. Getting Started with FUSE Integration Designer

This chapter introduces the FUSE Integration Designer, an Eclipse tooling used for creating and integrating Web services using Enterprise Integration Patterns (EIPs) as a standard method of notation for integration. It also describes the principal functional areas of FUSE Integration Designer and explains how to find your way around the user interface.

Overview of FUSE Integration Designer

FUSE Integration Designer is an Eclipse based development environment that is designed for Enterprise Integration Pattern developers who wish to visualize, create, deploy and debug EIP applications.

It provides tools and wizards for developers who wish to create and integrate web services using Enterprise Integration Patterns (EIPs). With FUSE Integration Designer, you can create EIP diagrams based on EIPs from FUSE Mediation Router, generate an EIP diagram from a FUSE Mediation Router configuration file, export EIP diagrams to FUSE Mediation Router Spring-based XML configuration files and deploy applications on different types of servers, including FUSE ESB Server. You can also create web services based on FUSE Services Framework.

The Designer helps you in defining Enterprise Integration Patterns in Java, using a domain specific language (DSL), and also includes a graphical tool that lets you explore JMS client functions, properties, and behaviors.

Installing FUSE Integration Designer

This section assumes that you already have a working copy of the FUSE Integration Designer. For using all features available in FUSE Integration Designer, you may want the following also installed on your system:

FUSE Mediation Router

FUSE Mediation Router is an open source rule-based routing and process mediation engine based on the Apache Camel project. It enables you to implement EIPs using plain old Java objects (POJOs).

FUSE Services Framework

FUSE Services Framework is an open source pluggable service framework based on the Apache CXF project. It fully implements the Java API for XML Web Services (JAX-WS) 2.0 specification, making it easier for Java developers to expose existing Java code as a Web service or to write new Web services.

FUSE Message Broker

Based on Apache ActiveMQ, FUSE Message Broker is a flexible messaging platform for reliably executing transactions, moving data, efficiently scaling operations, and connecting processes across heterogeneous database and application environments. FUSE Message Broker is high performance middleware that loosely couples services in time and location.

FUSE ESB Server

FUSE ESB is an open source integration platform based on Apache ServiceMix that supports JBI and OSGi for use in enterprise IT organizations. FUSE ESB is robust SOA infrastructure that provides a standardized methodology, server, and tools to integrate components in mission-critical applications.

Starting FUSE Integration Designer

Once the installation is complete, start FUSE Integration Designer in the following way:

On Windows:

1. Go to **<installation_dir>/fuse-integration-designer-1.2/bin**.
2. Double click on `fuse.exe`. FUSE Integration Designer starts.

On Linux:

1. Launch the terminal window.
2. Set your PATH variables using the following syntax: **export PATH=<Java PATH till bin>:$PATH**
3. Start the FUSE Integration Designer from within the terminal using the following commands: **cd <installation_dir>/fuse-integration-designer-1.2/bin** and **./fuse** .

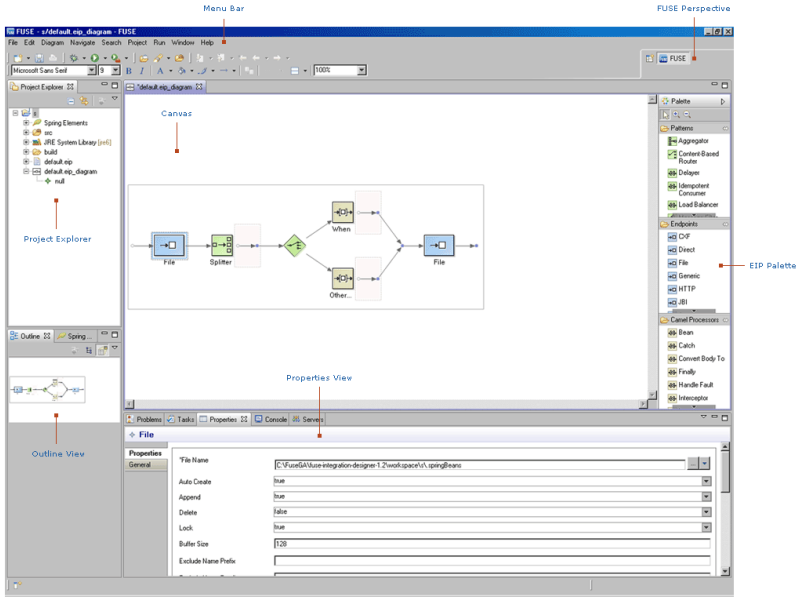
Note

To avoid setting the PATH variable each time, you can also save the PATH variable in a batch file.

When you start FUSE Integration Designer the first time, you are greeted by a Welcome Screen. Use the Show Editor icon to go to the FUSE Integration Designer editor.


FUSE - FUSE
File Edit Navigate Search Project Run Window Help
Welcome
FUSE Integration Designer
Show Editor

Quick Tour of FUSE Integration Designer



Key features

FUSE Integration Designer allows you to:

- Create enterprise integration patterns (EIP) diagrams to describe your integration scenario
- Generate Apache Camel configuration from EIP diagrams
- Generate an EIP diagram from Camel configuration
- Define Enterprise Integration Patterns or routing rules in Java, using a domain specific language (DSL)
- Create Messaging Projects to send and receive JMS messages
- Explore JMS client functions, properties, and behaviors using the FUSE Messaging Tool
- Deploy applications on FUSE ESB server
- Run and debug the EIP file or diagrams

- Run the Camel configuration file
- Create Web services based on Apache CXF (FUSE Services Framework)

Chapter 2. Setting Up FUSE Integration Designer

This chapter helps you start and setup your FUSE Integration Designer and also provides you information about the different types of applications that you can create using FUSE Integration Designer.

Setting Java Preferences

Before you start using FUSE Integration Designer, make sure your installed JRE and Java compiler preferences are correctly set.

FUSE Integration Designer runs on JDK 1.5, so you need to set your Java preferences as follows:

1. Select **Window** → **Preferences**.
2. In the Preferences window, select **Java** → **Installed JREs**.
3. Ensure that JRE 1.5 is selected as the default runtime environment.
4. Select **Java** → **Compiler**.
5. In the **JDK Compliance** section, select **1.5** from the **Compiler compliance level** drop-down list.

Adding Server Runtimes

If you plan to create a Web service for deployment to an application server such as Apache Tomcat, FUSE ESB Server or JBoss, you need to add the server runtime location to Eclipse.

To add the location of an installed runtime:

1. Select **Window** → **Preferences**.
2. In the Preferences window, select **Server** → **Runtime Environments**.
3. In the Installed Server Runtimes panel, click **Add**.
4. In the New Server Runtime dialog, select the type of server that you want to add and click **Next**.
5. Select the location of the server on your machine and click **Finish**.

Switching to the FUSE Perspective

The FUSE perspective includes the Project Explorer, the Palette, Properties View and the Problems View. When you start FUSE Integration Designer for the first time, you are by default in the FUSE perspective.

If you are in a perspective other than FUSE, and want to switch to the FUSE perspective:

- Click **Window** → **Open Perspective** → **FUSE (default)** on the menu bar.

Note

When the perspective opens, the title bar of the window it is in changes to display the name as **FUSE**. Use these icons in the shortcut bar to quickly switch between perspectives.

Creating Projects in FUSE Integration Designer

FUSE Integration Designer enables you to create various types of projects, such as

- **EIP Project:** A FUSE project that allows you to create and integrate Web services using Enterprise Integration Patterns (EIPs) as a standard method of notation for integration. For more on this, read here on page 26.
- **Java DSL Project:** Use this project type to define Enterprise Integration Patterns or routing rules in Java, using a domain specific language (DSL). For more on this, read here on page 126.
- **Messaging Project:** Use this project type to sending and receiving JMS messages. You can also create message producers (QueueSenders and Publishers) and message consumers (QueueReceivers and Subscribers) in this project. For more on this, read here on page 143.
- **Dynamic Web Project:** Use this project type to create a WTP dynamic Web project that complies with the Spring Framework project structures. For more on this, read here on page 114.

.

Chapter 3. Working with EIP Diagrams

FUSE Integration Designer allows you to describe your enterprise integration scenarios using the EIP Diagram Editor.

Creating an EIP Project

To create a FUSE Integration Designer EIP project:

1. Select **File** → **New** → **Project**.
2. In the Select a project panel of the New Project wizard, select **FUSE** → **EIP Project** and click **Next**.
3. In the FUSE Project panel, enter a project name
4. The "Use default location" is enabled by default. You may uncheck this to select a different location (from the workspace).
5. Click **Finish**.

Creating EIP Diagrams

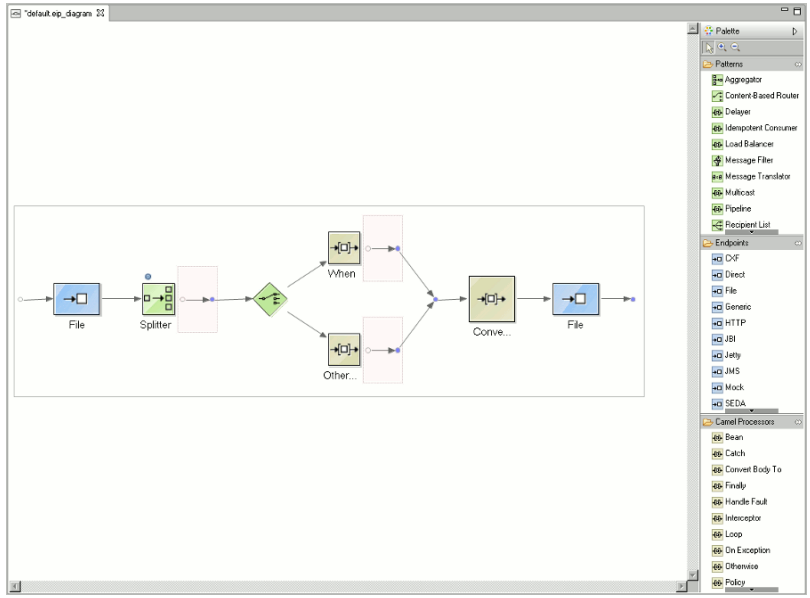
You can create an EIP diagram from within a General project. However, if you want to take advantage of FUSE Integration Designer's integration with Maven and Spring IDE, you should create the diagram from an EIP project.

To create an EIP diagram:

1. Select **File** → **New** → **Other**.
2. In the New wizard, select **FUSE** → **EIP Diagram** and click **Next**.
3. In the Create EIP Diagram panel, select a project folder as the parent folder and enter a unique name in the **File name** field.
4. Click **Next**.
5. In the Create EIP Domain Model Name panel, select a project folder as the parent folder and enter a unique name in the **File name** field.
6. Click **Finish**.

The default EIP diagram that opens comprises one start dummy node, a connector and an end dummy

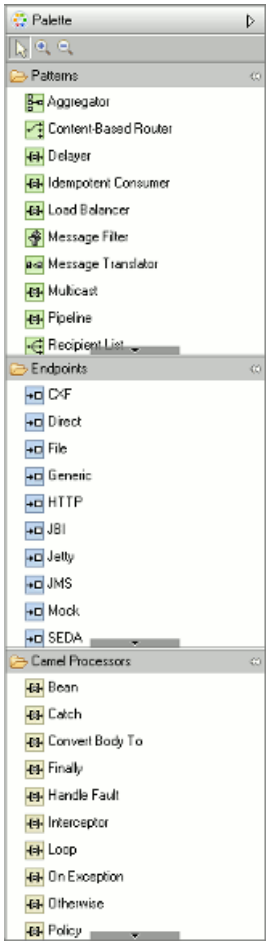
The route created in an EIP Diagram can also be called a Pipeline. In the EIP diagram, click anywhere inside the dotted line on the palette canvas and set the Pipeline properties on page 73.


*default.eip_diagram
Palette
Patterns
Aggregator
Content-Based Router
Delayer
Idempotent Consumer
Load Balancer
Message Filter
Message Translator
Multicast
Pipeline
Recipient List
Endpoints
DXF
Direct
File
Generic
HTTP
JBI
Jetty
JMS
Mock
SEDA
Camel Processors
Bean
Catch
Convert Body To
Finally
Handle Fault
Interceptor
Loop
On Exception
Otherwise
Policy
File
Splitter
When
Other...
Conve...
File

Using the EIP Diagram Editor

The EIP Diagram Editor consists of two sections: the canvas and the palette.

The palette contains a core selection of the patterns, endpoints and processor types supported by Apache Camel.



You build up an EIP diagram by selecting components on the palette and then dragging and dropping them onto the connector icon (where you want to place them) in the canvas. As you drag the mouse across to the palette

and onto the connector, the "cannot drop" icon converts into a "drop" icon. Drop the component on the connector and set properties for each component.

When you drag and drop the component on an existing component, FUSE Integration Designer replaces the exiting component with the new one.

Setting Diagram Preferences

To configure the appearance and behavior of FUSE Integration Designer EIP diagrams:

1. Select **Window** → **Preferences**.
2. In the Preferences window, select **Progress** → **FUSE** → **FUSE EIP Diagram**.

Most of the preferences relate to how your EIP diagrams appear. In most cases, it makes sense to leave these set to their default values. If you make changes that you don't like, use the *Restore Defaults* button to restore the diagram settings to its default values.

Setting FUSE Mediation Router Location

Before deploying your applications, you must configure the FUSE Mediation Router Runtime location:

1. Select **Window** → **Preferences**.
2. In the Preferences window, select **Progress** → **FUSE** → **FUSE EIP Diagram** → **FUSE Mediation Router Location**.
3. Browse for and select the runtime location of FUSE Mediation Router on your machine.

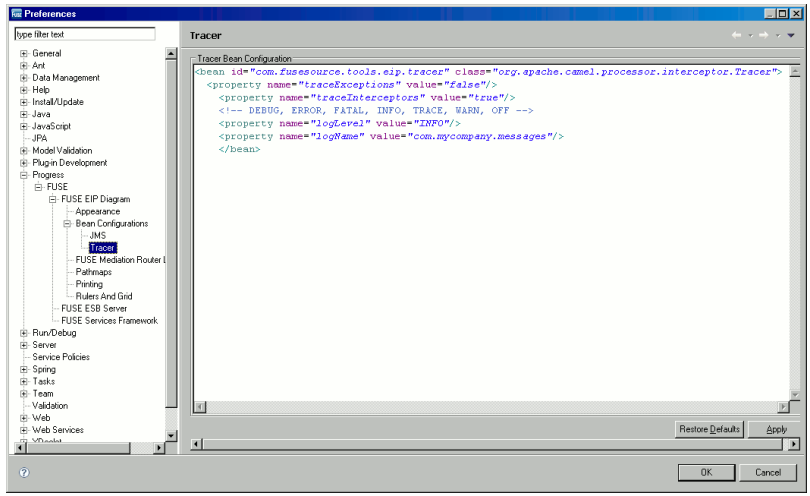
Note

If you have not set the Mediation Router Runtime location here, FUSE Integration Designer does not allow you to run or debug your configurations.

Setting Tracer Preferences

To use the Apache Camel tracer interceptor for logging the route executions at INFO level, you must first configure the Tracer Bean Configuration in the Preferences panel. To be able to view the tracer log in the Console view, you must also enable the tracing on page 106 while configuring your EIPs and Camel Configurations.To set Tracer preferences:

1. Select **Window** → **Preferences**.
2. In the Preferences window, select **Progress** → **FUSE** → **FUSE EIP Diagram** → **Bean Configurations** → **Tracer**.
3. Edit the Tracer Bean Configuration file in the editor.
4. Click **Apply** to apply the changes. To restore the configuration to its default value, click **Restore Defaults**.



Setting JMS Preferences

When using a JMS endpoint in an EIP diagram, you can set a number of preferences for the JMS provider that FUSE Integration Designer supports, namely Apache ActiveMQ (FUSE Message Broker).

To set JMS preferences:

1. Select **Window** → **Preferences**.
2. In the Preferences window, select **Progress** → **FUSE** → **FUSE EIP Diagram** → **Bean Configurations** → **JMS**.

Main preferences

You can set the following properties for connections to an ActiveMQ broker:

Table 3.1. Main ActiveMQ ConnectionFactory Preferences

Property	Description
Broker URL	The URL used to connect to the ActiveMQ broker
Username	Sets the username used for connections created from this JMS endpoint.
Password	Sets the password used for connections created from this JMS endpoint.
Client ID	Sets the JMS client ID to use for the created connection.
Dispatch asynchronously	Configures the broker to dispatch messages to the endpoint asynchronously. This is the best setting for dealing with slow consumers. If you want faster throughput and if the chances of having a slow consumer are low, disable this option.
Copy message on send	Copies the JMS message to a new JMS Message object as part of the send() method. You can disable this if you do not mutate JMS messages after they are sent for a performance boost.
Disable timestamps by default	Select this to achieve a small performance boost by disabling timestamps on messages.
Optimize message dispatch	Select this to use a a larger prefetch limit (only applicable for durable topic subscribers)

Property	Description
Use asynchronous send	Configures the endpoint to send messages to the broker asynchronously. This adds a massive performance boost, but means that the send() method returns immediately whether the message has been sent or not, which could lead to message loss.
Use compression	Compress the message bodies
Use retroactive consumer	Enables retroactive consumers by allowing non-durable topic subscribers to receive messages that were published before the subscriber started.
Watch topic advisories	
Close timeout	Sets the timeout in milliseconds before a close is considered complete.
Always use session asynchronously	If this option is selected a separate thread is not used for dispatching messages for each session in the connection.
Optimize acknowledgement	Enables an optimized acknowledgement mode where messages are acknowledged in batches rather than individually.
Statistics enabled	
Always send synchronously	Requires all messages to be sent synchronously.

Prefetch policy preferences

The maximum number of messages that ActiveMQ pushes to a consumer without the consumer processing a message is set by the prefetch size. You can improve throughput by running ActiveMQ with larger prefetch sizes.

You can use the Prefetch Policy tab of the ActiveMQ preferences page to set prefetch values for properties for different consumer types, including queues, topics, and durable queues and topics.

Redelivery policy preferences

You can set the following redelivery policy properties for ActiveMQ:

Table 3.2. Redelivery Policy Preferences

Property	Description
Maximum redelivery attempts	The maximum number of times a message is redelivered before it is considered a poisoned pill and returned to the broker so it can go to a dead letter queue. Use a value of -1 to define infinite number of redeliveries.
Initial redelivery delay	The initial redelivery delay in milliseconds
Use collision avoidance	Should the redelivery policy use collision avoidance
Use exponential backoff	Should the timeout be increased exponentially
Backoff multiplier	The value by which the timeout should be increased if exponential backoff is enabled
Redelivery collision avoidance percentage	The percentage range of collision avoidance if enabled

BLOB transfer policy

ActiveMQ includes the capability for large files (binary large objects, or BLOBs) to be processed by JMS consumers. BLOBs are sent "out-of-band" using FTP, HTTP, or SCP, and not using ActiveMQ's main transport connections.

You can set the following BLOB transfer policy properties for ActiveMQ:

Table 3.3. BLOB Transfer Preferences

Property	Description
Default upload URL	The default upload URL to use if the broker does not have a configured upload URL
Broker upload URL	Called by the JMS client when a broker advertises its upload URL
Upload URL	Sets the upload URL to use explicitly on the client which will overload the default or the broker's URL
Buffer size	The default buffer size used when uploading or downloading files

Validating Diagrams

FUSE Integration Designer automatically validates your EIP diagrams as you work. Where validation fails on an endpoint or processor, a red X is displayed on the component.

You can then use the Problems view to see details of the validation error.

You can also validate an EIP diagram, by right clicking on the canvas and selecting **Validate** from the menu bar.

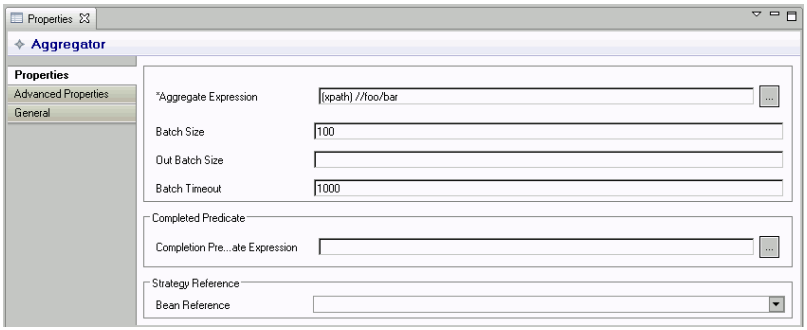
Chapter 4. Properties View

The FUSE Integration Designer Properties view allows you to set properties for the endpoint, patterns and processor in your EIP diagrams.

Introducing the Properties View

Once you have created an EIP diagram, you need to set properties for the endpoints, patterns and data processors in the diagram before you can move to the next step, generating Apache Camel configuration. The Properties view is displayed when you create or open an EIP file, irrespective of the current perspective.

To display properties for a particular endpoint or processor, right-click the object in the EIP diagram and select **Show Properties View**. You can also double click on the particular endpoint or processor to open the Properties view.



The properties view consists of the three tabbed pages. There may be more tabbed pages, according to the type of endpoint or processor that is selected.

The **General** tabbed page contains a display name—as it appears on the EIP diagram—and a description of the endpoint or processor.

The **Properties** tabbed page consists of properties that vary according to the type of endpoint or processor that is selected.

The **Advanced Properties** tabbed page contains a field called Error Handler Bean Reference, where you set an error handler for data processors. The Error Handler Reference attribute must refer to the name of a spring bean, which you have defined in a spring xml configuration on page 41. To set an error handler, select a bean that already exists.

Referencing Spring XML from an EIP

Some EIP processors require you to reference a Spring bean that provides additional routing functionality. For example, content enricher and content filter require additional code in order to augment or filter the content of the message being sent.

To be able to reference a Spring bean from within an EIP processor, you first need to add the Spring XML file to your project, by selecting **File** → **New** → **Spring Bean Definition**.

To reference the Spring XML file from within an EIP diagram:

1. In the EIP diagram, right-click anywhere on the palette canvas and select **Show Properties View**.
2. In the Integration Graph properties view, click **Spring Beans**.
3. Click **Add Beans**. This adds a new row to the Spring Beans section.
4. Click on the row and then click the **...** button to bring up the Select Resource panel. Select the Spring XML file to be added and click **OK**.

Chapter 5. Setting Properties for EIP Endpoints

Use the Properties View to set properties for the EIP endpoints in your EIP diagrams.

Direct Endpoint Properties

A Direct Endpoint allows you to make synchronous calls to another endpoint. It is typically used to connect existing routes or if a client in the same JVM as the Camel router wants to access the routes.

A Direct endpoint takes the following properties:

URI
: Enter a name to uniquely identify the direct endpoint.(Required)

Allow Multiple Consumers
: This is by default set to True. If you set it to false, then a second consumer is started on the endpoint.

HTTP Endpoint Properties

An HTTP endpoint is used for consuming external HTTP resources.

Note

In Apache Camel, you only ever route *to* an HTTP endpoint.

An HTTP endpoint takes the following properties:

Hostname
: The hostname of the machine that the endpoint resides on. (Required)

Port
: The port number that the endpoint is listening on.

Resource URI
: The remainder of the URI to the endpoint, including any options.

  For example, if you entered a hostname of **`www.google.com`**, you could enter **`/search?q=foo`** here to produce a URI of `http://www.google.com/search?q=foo` in the generated Camel configuration.

You can also configure the URI options and Header properties in their respective tabs.

URI Options
: Click the URI Options tab. Click Add Parameter to add the endpoint URI name you wish the message to be routed to. Enter the value of the parameter in the Parameter Value field.

SEDA Endpoint Properties

A SEDA endpoint provides asynchronous SEDA behavior so that messages are exchanged on a BlockingQueue and consumers are invoked in a separate thread to the producer.

Note

In Apache Camel, SEDA does not implement any kind of persistence or recovery if the VM terminates while messages are yet to be processed. If you need persistence, reliability or distributed SEDA then try using either JMS or ActiveMQ.

A SEDA endpoint takes the following properties:

URI
: Enter a string that uniquely identifies the SEDA endpoint within the current CamelContext. (Required)

  For example, if you enter the URI as **`endpointname`**, it would conform to the URI format `seda:endpointName` in the generated Camel configuration.

Size
: The maximum size of the SEDA queue.

Mock Endpoint Properties

A Mock endpoint integrates with the Spring Testing framework to simplify unit testing and integration testing of your FUSE Mediation Router applications.

A Mock endpoint takes the following properties:

URI

Enter a string that uniquely identifies the Mock endpoint. (Required)

For example, if you enter the URI as **`endpointname`**, it would conform to the URI format `mock:endpointName` in the generated Camel configuration.

File Endpoint Properties

In Apache Camel, the File component provides access to file systems, allowing files to be processed by other Camel components and messages from other components to be saved to disk.

A File endpoint takes the following properties:

Table 5.1. Common File Endpoint Properties

Property	Description	Default
File Name	The name of the file or directory where the endpoint is located. Required. Click on the down arrow to select from either the workspace, local file system or directory.	-
Auto Create	If set to true Camel creates the directory to the file if the file path does not exist	True
Append	When writing to the file, should the process append to the end of the file, or overwrite the existing contents?	True
Delete	If true, the file will be deleted once it is processed.	False
Lock	If true, lock the file for the duration of the processing.	True
Buffer Size	Write buffer sized in kilo bytes.	128
Exclude name prefix	Exclude files whose filenames start with the given prefix	-
Exclude name postfix	Exclude files whose filenames end with the given postfix	-
Ignore header	If true, producers ignore the `org.apache.camel.file.name` header and generate a new dynamic filename	False
Expression	Use the given expression to dynamically set the filename.	-
No op	If true, the file is not moved or deleted. This option is good for read only data, or for Extract Transform Load (ETL) scenarios.	False

Property	Description	Default
Move Name Prefix	The string perpended to the filename when moving it. For example to move processed files into the `done` directory, set this value to **`done/`**	-
Move Name Postfix	The string appended to the filename when moving it. For example to rename processed files from `foo` to `foo.old` set this value to **`.old`**	-

Table 5.2. Consumer File Endpoint Properties

Property	Description	Default
Initial Delay	Milliseconds before polling the file or directory starts. Required.	1000
Delay	Milliseconds before the next poll of the file or directory	500
Recursive	If the endpoint is a directory, look for changes in files in all the sub-directories	False
Use Fixed Delay	If true, poll once after the initial delay.	False
Exclusive Read Lock	If true, Camel only polls files on which it has an exclusive read lock. If the file is in the process of being written, Camel waits until the file lock is granted. If false, Camel polls the file even if it is in the process of being written.	True
RegExpression Pattern	The process will only fire for a file that matches the given regular expression pattern.	-
Empty Exchange	If true and there are no files to process, Camel simulates processing a single empty file, so that an exchange is fired.	False
Always consume	The file is consumed even if it hasn't changed since last time it was consumed.	False

JMS Endpoint Properties

The Camel JMS component allows messages to be sent to a JMS queue or topic, or messages to be consumed from a JMS queue or topic.

A JMS endpoint takes the following properties, both of which are required:

Type
: Whether the endpoint is a queue, topic, temp:queue or temp:topic.

Name
: The name of the endpoint.

FUSE Integration Designer uses Apache ActiveMQ as its JMS provider. You can configure ActiveMQ by setting JMS preferences on page 34.

Generic Endpoint Properties

The Generic Endpoint allows you to create a generic endpoint that can take any URL supported by Camel. A Generic URI endpoint takes the following required property:

Endpoint URI
: The String representation of the endpoint URI.

CXF Endpoint Properties

The CXF component enables you to access endpoints using the Apache CXF open services framework (primarily Web services). Because CXF has support for multiple different protocols, you can use a CXF component to access many different kinds of service. A CXF URI endpoint takes the following properties:

URI
: The String representation of the URI that conforms to the format: cxf://Address?QueryOptions, where Address is the physical address of the endpoint, whose format is binding/transport specific. This is a Required field.

WSDL URL
: Location of the WSDL contract file (defaults to the value of the annotation in the service class, if one is specified).

Service Class
: Service Class is the name of the service endpoint interface (SEI) class. This class basically defines the service interface of the web service to be created.If the SEI class is appropriately annotated, it also determines the WSDL location, service name, and port name for the WSDL endpoint. This is a Required field.

  You create an interface class with the necessary methods, and then give the name of the class defined in the Service Class field here. This will create a web service at the location specified in the URI. Ensure that the CXF endpoint is the first endpoint in your EIP diagram for this to be implemented.

Service Name
: The service QName (defaults to the value of the annotation in the service class, if one is specified).

Port Name
: The port QName (defaults to the value of the annotation in the service class, if one is specified).

JBI Endpoint Properties

The JBI component is provided by the ServiceMix Camel module and provides integration with a JBI Normalized Message Router such as provided by Apache ServiceMix. A JBI endpoint takes the following properties:

Type
: Select the type from the drop down list as either endpoint, service or name. Selecting the type here decides the URI format that the JBI will accept. This is a Required field.

URI
: The String representation of the URI that conforms to the format based on the type that you have selected. For example: jbi:service: or jbi:endpoint: followed by the URI you specify. This is a Required field.

MEP
: Allows you to override the MEP being used for interacting with JBI. Select from the values in the drop down: in-only, in-out, robust-in-out and in-optional-out.

Operation
: Specify the JBI operation to be used for the MessageExchange. If no value is supplied, the JBI binding will use the value of the jbi.operation header property.

Jetty Endpoint Properties

In Apache Camel, the Jetty component provides HTTP based endpoints for consuming HTTP requests that arrive at an http endpoint.

A File endpoint takes the following properties:

Table 5.3. Basic Jetty Endpoint Properties

Property	Description	Default
Host Name	The hostname of the machine that the endpoint resides on. Required.	-
Port	The port number that the endpoint is listening on.	-
Resource URI	The remainder of the URI to the endpoint, including any options.	-
HTTP Binding Ref	Reference to a org.apache.camel.component.http.HttpBinding in the Registry. HttpBinding can be used to customize how response should be written.	null
Session Support	If true, it enables the session manager in the server side of Jetty.	False

Table 5.4. Advanced Jetty Endpoint Properties

Property	Description	Default
Connector Type	Enter the value for the type of connector (socket, blocking or select) in use. Enter 2 if you want the connector type as CONNECTOR_SELECT_CHANNEL, and 0 for CONNECTOR_SOCKET.	-
Idle Timeout	Enter the timeout, in units of milliseconds in this field. The field denotes the period in milliseconds a JettyConnection can be idle for before it is closed.	-
Key Manager Password	The password for the key manager file.	-
Key Store Location	The location where Jetty will load the keystore from in order to load the correct SSL certificate.	-

Property	Description	Default
Key Store Password	The password for the keystore file.	-
Max Retries	Enter the maximum number of times to attempt to recreate connections to the server when the connection fails.	-
Max Connections Per Address	The process will only fire for a file that matches the given regular expression pattern.	-
Socket Timeout	Enter the maximum amount of time (in milliseconds) the connector will wait for a response from the server before disconnecting the socket and returning an exception to the client application.	-
Timeout	Enter the Timeout, in units of milliseconds in this field. Timeout specifies the period that an exchange will wait for a response from the server	-
Use Direct Buffers	Sets whether the connector can use Non-blocking (NIO) direct buffers.	True
Trust Store Password	The password for the truststore file.	-
Trust Store Location	The location where Jetty will load the truststore from in order to load the correct SSL certificate.	-

You can also configure the URI options and Header properties in their respective tabs.

URI Options

Click the URI Options tab. Click Add Parameter to add the endpoint URI name you wish the message to be routed to. Enter the value of the parameter in the Parameter Value field.

Timer Endpoint Properties

The Timer component is used to generate message exchanges when a timer fires. You can only consume events from this endpoint. The Timer endpoint takes the following properties:

Name
: The name of the Timer component which is shared across endpoints. If you use the same name for all your timer endpoints then only one Timer object & thread are used. This is a Required field.

Date
: Enter the date when the first event will be generated. The format for the date is **yyyy-MM-dd'T'HH:mm:ss**.

Period
: Enter the Period, in units of milliseconds in this field. Period specifies the number of milliseconds periodic events will be generated each time. The default value is 1000.

Delay
: Enter the delay time, in units of milliseconds in this field. Delay specifies the fixed time that the Timer will wait before the first event is generated. The default value is 0. If you have also specified a Date, the Timer does not consider the Delay value.

Fixed Rate
: Select the Fixed Rate as either True or False. The default value is False. If you select the Fixed Rate as True, events take place at approximately regular intervals, separated by the specified period..

Daemon
: Select the daemon option as either True or False. The default value is true. This specifies whether the thread associated with the timer endpoint must be run as a daemon.

XSLT Endpoint Properties

The XSLT component allows you to process a message using an XSLT template. This is ideal when using Templating to generate responses for requests.The XSLT endpoint takes the following properties:

Template Name
: Select the template to be invoked, either from the workspace or the local directory. The XSLT file must be an XSL file. This is a Required field.

Converter
: Select a converter in this field, in case you want to override the default XmlConverter. The converter you select must be of type **org.apache.camel.converter.jaxp.XmlConverter**.

FTP Endpoint Properties

An FTP endpoint provides access to remote file systems over the FTP and SFTP protocols. The FTP endpoint contains the following properties:

Table 5.5. FTP Endpoint Properties

Property	Description	Default
Protocol	Whether the protocol is FTP, or SFTP. Required.	FTP
User Name	The username used to log in to the remote system. If no username is provided then anonymous login is attempted using no password.	-
Password	The password used to log in to the remote system.	-
Hostname	The hostname of the server. Required.	-
Port Number	The port the that server is listening on.	-
File Name	The name of the file or directory on the server. Can contain nested directories. Required.	-
Directory	Indicates whether the given file name should be interpreted as a directory or file.	True
Binary	Specifies the file transfer mode as BINARY. The default transfer mode is ASCII.	False
FTP Client Config	Reference to a bean in the registry as a org.apache.commons.net.ftp.FTPClientConfig class. Use this option if you need to configure the client according to the FTP Server date format, locale, timezone, platform etc. See the javadoc FTPClientConfig for more documentation.	-
Excluded Name Postfix	Used by FTPConsumer. Is used to exclude files if filename is ending with the given postfix.	-
Timestamp	This option is used for backwards comparability.	False
Expression	Use expression to dynamically set the filename, like dynamic pattern style filenames. The expression options supports both String and Expression types. Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel.	-

Property	Description	Default
Passive Mode	Use this to set whether to use passive mode connections. Default is active. This feature is only for regular FTP, not SFTP.	False
Known Hosts	Use this to set the known_hosts file so that the SFTP endpoint can do host key verification.	-
Private Key File	Use this to set the private key file to that the SFTP endpoint can do private key verification.	-
Private Key File Passphrase	Use this to set the private key file passphrase to that the SFTP endpoint can do private key verification.	-

Table 5.6. Consumer Properties

Property	Description	Default
Recursive	This option is only for backwards comparability.	False
Set Names	-	
Delay	Delay in milliseconds between each poll	500
Initial Delay	Initial delay in milliseconds before polling starts	1000
Regular Expression Pattern	Used by FTPConsumer. Regular expression to use for matching files when consuming.	-
Exclusive Read Lock	Used by FTPConsumer. This option is used to force Camel not to consume files that is currently in the progress of being written. If set to true Camel will only poll the ftp files if it has exclusive read to the file (= the file is not in progress of being written). Camel will wait until it is granted, testing once every second. The test is implemented by Camel will try to rename the file. Setting to false Camel will poll the file even if its in progress of being written..	False
Delete File	Used by FTPConsumer. Flag to set if the consumed file should be deleted after it has been downloaded.	False
Movename Prefix	Used by FTPConsumer. The prefix String perpended to the filename when moving it. For example to move processed files into the done directory, set this value to 'done/'	-

Property	Description	Default
Movename Postfix	Used by FTPConsumer. The postfix String appended to the filename when moving it. For example to rename processed files from foo to foo.old set this value to '.old'	-
Excluded Name Prefix	Used by FTPConsumer. Is used to exclude files if filename is starting with the given prefix.	-
Excluded Name Postfix	Used by FTPConsumer. Is used to exclude files if filename is ending with the given postfix.	-
User Fixed Delay	Use true to use fixed delay between pools, otherwise fixed rate is used.	Flase

Chapter 6. Setting Properties for EIP Patterns

Use the Properties View to set properties for the EIP patterns in your EIP diagrams.

Load Balancer Properties

A Load Balancer Pattern allows you to delegate to one of a number of endpoints using a variety of different load balancing policies.

In FUSE Integration Designer, you can configure the load balancer properties in the Properties view, after adding the load balancer pattern to your EIP diagram.

Sticky Expression

Click on the ... icon to bring up the Expression Editor. Sticky expression is useful only when you select Sticky as a language type. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select the language type from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter an expression to perform load balancing on the message.

Reference

Select from the available list of Spring Beans registered with the EIP diagram. For a bean to be available in the drop-down list, you first need to write a route in Spring XML and reference the Spring bean and method in the Properties view. See Referencing Spring XML from an EIP on page 41 for details on how to add the Spring XML file to your EIP diagram.

Load Balancer Type

Select a load balancer type (also known as the load balancing policy)

- **Round Robin:** The exchanges are selected in a round robin fashion. This spreads the load even.

- **Random:** A random endpoint is selected for each exchange

- **Sticky:** Sticky load balancing using an Expression to calculate a correlation key to perform the sticky load balancing; rather like jsessionid in the web or JMSXGroupID in JMS.

- **Topic:** Topic which sends to all destinations (rather like JMS Topics)

Delayer Properties

A Delayer Pattern allows you to delay the delivery of messages to its specified destination.

In FUSE Integration Designer, you can configure the Delayer properties in the Properties view, after adding the Delayer pattern to your EIP diagram.

Delay
: Enter the delay time, in units of milliseconds in this field. Delay specifies the fixed time that the Dealyer will delay things from the point at which it receives the message. During this time, the delayer collects the messages. For example, to delay things for 2 seconds, enter the value as 2000.

Delayer Expression
: Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel.

  - **Language:** Select a scripting language from the Expression Language drop-down list. The expression language supported helps perform the delay.

    FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

  - **Expression Value:** Enter the correlation expression to determine which messages should be aggregated together.

Recipient List Properties

A Recipient List pattern allows you to route messages to a number of dynamically specified recipients.

In FUSE Integration Designer, you can configure the Recipient List properties and create a dynamic recipient list in the Properties view, after adding the pattern to your EIP diagram.

Recipient List Expression

Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select a scripting language from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter a valid expression.

Routing Slip Properties

A routing slip pattern enables you to route a message consecutively through a series of processing steps, where the sequence of steps is not known at design time and can vary for each message.

In FUSE Integration Designer, you can configure the routing slip properties in the Properties view, after adding the routing slip pattern to your EIP diagram.

Header name
: The name of the header with the list of URI's, delimited by a URI Delimiter. The default value is **routingSlipHeader**.

URI Delimiter
: List of endpoint URIs you wish the message to be routed to. The default value is **,**.

Splitter Properties

A splitter splits an incoming message into a series of outgoing messages, where each of the messages contains a piece of the original message and is processed individually.

In FUSE Integration Designer, you configure the splitter properties in the Properties view.

Splitter Expression
: Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

  - **Language:** Select a scripting language from the Language drop-down list. The expression language supported helps perform the split.

    FUSE Integration Designer allows you to add all Camel supported languages in the Expression Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

  - **Expression Value:** Enter an expression to split the message. This is a Required field.

    For example, to extract bar elements from an incoming message and insert them into separate outgoing messages using an XPath expression, enter `//foo/bar` in the Expression field, if you have selected the language as xpath.

Parallel Processing
: Select from the Boolean values, true or false to specify whether you want to to execute all parts in parallel. The default value is **false**.

Streaming
: Enter any java class type in this field. The output of the splitter is changed to the value specified in this property. For example, `String.class` in the Streaming field converts the incoming message into a camel Message object that contains a String payload.

Message Translator Properties

A message translator modifies the contents of a message, translating it to a different format.

In FUSE Integration Designer, you configure the message translator properties in the Properties view.

Translator Expression

Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select a scripting language from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expression Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter an expression to modify the content of the message.

Content-Based Router Properties

A content-based router examines the message content and routes the message based on the data contained in the message.

If you have added a content-based router pattern to your EIP diagram, you need to set the routing rules in the Properties view.

To set properties for a CBR:

1. In the Properties view, select the When processor in the CBR and click the **Properties** tab.

2. In the **Expression** field, click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

   - **Language:** Select a scripting language from the Language drop-down list.

     FUSE Integration Designer allows you to add all Camel supported languages in the Expression Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

   - **Expression Value:** Enter an expression to modify the content of the message.

3. You do not need to set any properties for the Otherwise processor in the CBR.

   To add more decision branches (When boxes) to the CBR, right click on the Content Based Router box in the EIP diagram and select **Add Decision Branch**.

Aggregator Properties

An Aggregator pattern combines a batch of related messages into a single message.

In FUSE Integration Designer, you configure the aggregator properties in the Properties view. To set properties for an aggregator:

1. In the Properties view, click the **Properties** tab.
2. In the **first section**, enter the following:

Aggregate Expression
: Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

  - **Language:** Select the aggregator type from the Language drop-down list.

    FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

  - **Expression Value:** Enter the correlation expression to determine which messages should be aggregated together.

Batch Size
: Enter an upper limit to the number of messages in a batch. The default value is **100**.

Out Batch Size
: Enter the number of exchanges currently aggregated in the AggregationCollection (used to to store the exchanges that is currently aggregated). When this threshold is reached the batch is completed and sent. By default this option is disabled. The difference to the batchSize options is that this is for outgoing, so setting this size to e.g. 50 ensures that this batch will at maximum contain 50 exchanges when its sent.

Batch Timeout
: Enter the Batch Timeout, in units of milliseconds in this field. Batch Timeout specifies the time interval, during which messages are

collected. The default value is **1000**. If no messages are received during a given time interval, no aggregate message is propagated.

3. In the **Completed Predicate** section, you specify a condition that determines when the current batch is complete. If the predicate resolves to true, the current message becomes the last message of the batch.

Note

When you use the Completion Predicate settings, the aggregator ignores the Batch Size and Batch Timeout settings.

You can specify the Completion Predicate condition using an expression language.

Completion Predicate Expression

Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel.

- **Language:** Select the aggregator type from the Language drop-down list. This is a Required field.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter the correlation expression to determine which messages should be aggregated together. This is a Required field.

4. In the **Strategy Reference** section, you enter a custom aggregation strategy, if you want to apply a different aggregation strategy from the default aggregation strategy. To do this, you must create Spring Bean and attach it to the EIP (using the properties view of the EIP) and then select the id in the Bean Reference field.

Bean Reference

Select a bean id from the Bean Reference drop-down list.

Multicast Properties

A Multicast enables you to monitor the messages passing through a channel by duplicating the message stream: one copy of the stream is forwarded to the main channel and another copy of the stream is forwarded to the tap endpoint, which monitors the stream.

The multicast pattern consists of an input, a destination output, and a tap output. In FUSE Integration Designer, you configure the multicast properties in the Properties view. To set properties for a multicast:

Strategy Reference
: Select the strategy reference responsible for aggregating all of the Out messages into a single reply message.

Parallel Processing
: Enable parallel processing as either True or False. By default, the multicast processor invokes each of the recipient endpoints one after the other (in the order listed in the to() command). In some cases, this might give rise to unacceptably long latency. To avoid such long latency times, you can enable parallel processing in the multicast processor by marking it as True.

Thread Pool Reference
: Refer the thread pool bean to specify the thread pool executor, which helps you customize the size of the thread pool that invokes the buyer endpoints.

Pipeline Properties

A Pipeline enables you to split your processing across multiple independent Endpoint instances which can then be chained together. The opposite of pipeline is multicast; which enables you to fire the same message into each of its outputs.

The Pipeline pattern consists of an input and a destination output. You do not need to configure any additional properties for this pattern type.

Message Filter Properties

You can use a message filter to eliminate undesired messages from a channel based on a set of criteria. The message filter pattern has just one output channel. If the message content matches the criteria specified by the filter, the message is routed to the output channel. If the message content does not match the criteria, the message is discarded.

In FUSE Integration Designer, you configure the message filter properties in the Properties view.

Filter Expressions

Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select a scripting language from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expression Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter an expression to filter the content of the message.

Resequencer Properties

A Resequencer pattern allows you to reorganize messages based on a sequencing expression. Messages that generate a low value for the sequencing expression are moved to the front of the batch and messages that generate a high value are moved to the back.

In FUSE Integration Designer, you configure the Resequencer properties in the Properties view. FUSE Integration Designer supports two resequencing algorithms: Batch and Stream.

The batch resequencing algorithm is enabled by default. If you do not enter values for both algorithms, the system, by default, obtains a batch by collecting all of the incoming messages that arrive in a time interval of 1000 milliseconds (default batch timeout), up to a maximum of 100 messages (default batch size).

To set properties for a resequencer:

1. In the Properties view, click the **Properties** tab.
2. To configure the Stream Resequencing algorithm, enter the following details in the **Stream Config** section.

   Capacity
   : Enter a value for the capacity parameter (maximum number of messages) in this field, used to prevent the resequencer from running out of memory.

   Timeout
   : Enter the Timeout, in units of milliseconds in this field. Timeout specifies the time delay between successive messages (that is, messages with adjacent sequence numbers) in a message streamed.

3. To configure the Batch Resequencing algorithm, enter the following details in the **Batch Config** section.

Note

You can specify either the Batch or Stream Resequencing pattern properties in their respective sections. If you enter values for both algorithms, the system displays an error.

Batch Size
: Enter an upper limit to the number of messages in a batch. The default value is **100**.

Timeout
: Enter the Timeout, in units of milliseconds in this field. Timeout specifies the time interval, during which messages are collected. The default value is **1000**.

4. Click the **Expression** tab. Click the **Add Expression** button to add an expression using a simple expression. You can add multiple expressions to prioritize the resequencing of messages.

   For example, when you add the first expression as `header.JMSPriority` and the second expression as, `header.MyCustomerRating`, the Resequencer sorts first by the header, JMSPriority and then by MyCustomerRating.

Idempotent Consumer Properties

An Idempotent Consumer pattern enables you to filter out duplicate messages. The pattern uses an expression to calculate a unique message ID string for a given message exchange; this ID is then looked up in the MessageIdRepository to see if it has been seen before; if it has the message is consumed; if its not then the message is processed and the ID is added to the repository.

In FUSE Integration Designer, you configure the Idempotent Consumer properties in the Properties view. To set properties for the pattern:

Expression
: Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

  - **Language:** Select a scripting language from the Language drop-down list.

    FUSE Integration Designer allows you to add all Camel supported languages in the Expression Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

  - **Expression Value:** Enter an expression to calculate a unique message ID string for a given message exchange.

Message ID Repository Reference
: Select a bean reference (repository of stored Message IDs) that is used to implement the Idempotent Consumer pattern. The unique ID generated by the Idempotent Consumer pattern can then be looked up in the Message Id Repository Ref to see if it has been seen before; if it has the message is consumed; if its not then the message is processed and the ID is added to the repository.

Throttler Properties

A Throttler pattern enables you to ensure that a specific endpoint does not get overloaded, or that you don't exceed an agreed SLA with some external service.

In FUSE Integration Designer, you configure the Throttler properties in the Properties view. To set properties for the pattern:

Maximum Requests Per Period
: Enter the time period value during which the maximum request count per period can be throttled.

Time Period
: Enter the time period, in units of milliseconds in this field. Time Period specifies the time interval, during which messages are throttled.The Default value is 1000.

Note

For example, if you want to throttle messages from one endpoint to the other, ensuring that a maximum of 3 messages are sent in any 10 second window, enter the value of Maximum Requests Per Period as 3 and the value of Time Period as 10000.

Chapter 7. Setting Properties for Camel Processors

Use the Properties View to set properties for the Camel processors in your EIP diagrams.

Bean Properties

A Bean component uses the Bean Binding to bind Java beans to message exchanges. You can also use the bean component to expose and invoke POJO (Plain Old Java Objects).

In FUSE Integration Designer, you can configure the bean processor properties in the Properties view, after adding the processor to your EIP diagram.

Method
: Enter the bean method name that you want to invoke.

Reference
: Select a bean from the Reference drop-down list. This enables you to bind maps message exchanges to the bean. For a bean to be available in the drop-down list, you first need to write a route in Spring XML and reference the Spring bean and method in the Properties view. See Referencing Spring XML from an EIP on page 41 for details on how to add the Spring XML file to your EIP diagram.

Try, Catch and Finally Properties

Camel supports the Java equivalent of try .. catch and finally directly in the DSL. A few important points to remember while adding these processors are:

- Catch should always be inside the Try processor.
- You can have multiple Catches.
- You can have only one Finally, and it should be inside the Try processor.

In FUSE Integration Designer, you set each of the components' properties in the Properties view, after adding the processors to your EIP diagram.

Catch allows you to define multiple exceptions to catch in a single block:

Exceptions List
: Click the Add Exceptions button and define the exception to be caught in a single block.

Convert Body To Properties

A Convert Body To enables you to convert the payload of the input message to be of the given type.

In FUSE Integration Designer, you can configure the Convert Body processor properties in the Properties view, after adding the processor to your EIP diagram.

Type
: Enter the type you want to convert the body payload into from its existing type. You can enter the type name from the available type conversions, or add your customized type converter.

Thread Properties

A thread is an asynchronous processor that forces subsequent processing in asynchronous thread from a thread pool.

In FUSE Integration Designer you can configure the Thread processor properties in the Properties view, after adding the processor to your EIP diagram.

Name
: Enter the name of the thread pool.

Priority
: Enter the thread pool priority. The default value is 5.

Core Size
: Enter the core size of the thread pool. The default value is 1.

Daemon
: Select the daemon option as either True or False. The default value is true.

Keep Alive Time
: Enter the keep alive time for the thread pool.

Max Size
: Enter the maximum size of the thread pool. The default value is 1.

Stack Size
: Enter the thread pool stack size.

On Exception Properties

An On Exception component enables you to discriminate between different exception types, performing different kinds of processing for different exception types.

In FUSE Integration Designer you can configure the On Exception processor properties in the Properties view, after adding the processor to your EIP diagram.

Expression
: Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

  - **Language:** Select the language type from the Language drop-down list.

    FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

  - **Expression Value:** Enter an expression to perform handling of errors on the message.

Maximum Redeliveries
: Enter the maximum redeliveries value, which specifies how many times redelivery is attempted.

Initial Redelivery Delay
: Enter the delay (in milliseconds) before attempting the first redelivery.

BackOff Multiplier
: Enter the BackOff Multiplier value. If you enable exponential backoff in the next field, let m be the backoff multiplier and let d be the initial delay. The sequence of redelivery attempts are then timed as follows: d, m*d, m*m*d, m*m*m*d, ...

Use Exponential BackOff
: Select exponential backoff as either True or False.

Collision Avoidance Factor
: Enter the collision avoidance factor in this field. If you enable collision avoidance in the next field, let p be the collision avoidance percent. The

collision avoidance policy then tweaks the next delay by a random amount up to plus/minus p% of its current value.

Use Collision Avoidance
: Select Collision Avoidance as either True or False.

Maximum Redelivery Delay
: Enter the delay (in milliseconds) before attempting the final redelivery.

Exceptions List
: Select the Exceptions List tab. Click the Add Exceptions button and define the exceptions to be caught.

Loop Properties

The Loop allows you to process a message a number of times and possibly process them in a different way. Useful mostly for testing.

In FUSE Integration Designer, you can configure the Loop processor properties in the Properties view, after adding the processor to your EIP diagram.

Loop Expression

Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select the language type from the Language drop-down list. The expression language supported helps perform the looping.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter an expression to perform looping on the message.

Interceptor Properties

The Interceptor supports intercepting Exchanges while they are on route.

In FUSE Integration Designer, you can configure the Interceptor processor properties in the Properties view, after adding the processor to your EIP diagram.

Reference
: Select from the available list of Spring Beans registered with the EIP diagram. For a bean to be available in the drop-down list, you first need to write a route in Spring XML and reference the Spring bean and method in the Properties view. See Referencing Spring XML from an EIP on page 41 for details on how to add the Spring XML file to your EIP diagram.

Policy Properties

If you do not want outbound endpoints to enlist in the same transaction as your inbound endpoint, you add a Transaction Policy to the processing route.

In FUSE Integration Designer, you can configure the Policy processor properties in the Properties view, after adding the processor to your EIP diagram.

Reference
: Select from the available list of Spring Beans registered with the EIP diagram. For a bean to be available in the drop-down list, you first need to write a route in Spring XML and reference the Spring bean and method in the Properties view. See Referencing Spring XML from an EIP on page 41 for details on how to add the Spring XML file to your EIP diagram.

Proceed Properties

The Proceed indicates that Camel will continue the normal route path after the interception.

In FUSE Integration Designer, you do not need to configure any properties for the Proceed component.

Process Properties

The Processor interface is used to implement consumers of message exchanges or to implement a Message Translator. In your route you use the process DSL syntax for invoking a processor.

In FUSE Integration Designer, you can configure the Process properties in the Properties view, after adding the processor to your EIP diagram.

Reference
: Select from the available list of Spring Beans registered with the EIP diagram. For a bean to be available in the drop-down list, you first need to write a route in Spring XML and reference the Spring bean and method in the Properties view. See Referencing Spring XML from an EIP on page 41 for details on how to add the Spring XML file to your EIP diagram.

Handle Fault Properties

The Handle Fault interface is used to handle faults within a route.

In FUSE Integration Designer, you can configure the Handle Fault properties in the Properties view, after adding the processor to your EIP diagram.

Reference
: Select from the available list of Spring Beans registered with the EIP diagram. For a bean to be available in the drop-down list, you first need to write a route in Spring XML and reference the Spring bean and method in the Properties view. See Referencing Spring XML from an EIP on page 41 for details on how to add the Spring XML file to your EIP diagram.

Set Header Properties

Use the Set Header to set header values of incoming messages.

In FUSE Integration Designer, you can configure the Set Header properties in the Properties view, after adding the processor to your EIP diagram.

Header Name
: Enter a unique String to identify the header name.

Set Header Expression
: Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select the language type from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter a valid expression.

Set Out Header Properties

Use the Set Out Header to set header values for the outgoing messages.

In FUSE Integration Designer, you can configure the Set Out Header properties in the Properties view, after adding the processor to your EIP diagram.

Header Name
: Enter a unique String to identify the header name.

Set Out Header Expression
: Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select the language type from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter a valid expression.

Set Property Properties

Use the Set Property to set exchange property values of incoming messages.

In FUSE Integration Designer, you can configure the Set Property properties in the Properties view, after adding the processor to your EIP diagram.

Property Name
: Enter a unique String to identify the property name.

Set Property Expression
: Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select the language type from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter a valid expression.

Set Body Properties

Use the Set Body to set body values of incoming messages.

In FUSE Integration Designer, you can configure the Set Body properties in the Properties view, after adding the processor to your EIP diagram.

Set Body Expression

Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select the language type from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expressions Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter a valid expression.

Remove Header Properties

Use the Remove Header to remove the header that you have set previously.

In FUSE Integration Designer, you can configure the Remove Header properties in the Properties view, after adding the processor to your EIP diagram.

Header Name
: Enter the name of the header to be removed. This is a Required field.

Remove Property Properties

Use the Remove Property to remove the property that you have set previously.

In FUSE Integration Designer, you can configure the Remove Property properties in the Properties view, after adding the processor to your EIP diagram.

Property Name
: Enter the name of the property to be removed. This is a Required field.

When and Otherwise Properties

You can drop the When and Otherwise processors only on a Content Based Router in the EIP Diagram.

In FUSE Integration Designer, you can configure these properties in the Properties view, after adding the processors to your EIP diagram.

To configure a When processor

Expression

Click on the ... icon to bring up the Expression Editor. Select the language and enter the expression in this panel. This is a Required field.

- **Language:** Select a scripting language from the Language drop-down list.

  FUSE Integration Designer allows you to add all Camel supported languages in the Expression Language combo field. This field is editable and you can add the language by either selecting from the combo box or typing the language in the field.

- **Expression Value:** Enter an expression to modify the content of the message.

Spring DSL Properties

Use the Spring DSL processor to add any additional types of Camel processors.

In FUSE Integration Designer, you can configure these properties in the Properties view, after adding the processors to your EIP diagram.

To configure the Spring DSL processor

Spring DSL
: Enter the contents of the Spring XML in the space provided. This is a Required field.

  For example, Use the command, **<xmlns="http://activemq.apache.org/camel/schema/spring" ref="myJaxb"/>** to marshal a bean into another format for transmission over some transport via a Camel Component.

Chapter 8. Importing and Exporting Camel Configuration

FUSE Integration Designer allows you to generate Apache Camel Configuration based on EIP diagrams and to generate EIP diagrams from Camel configuration.

Exporting to Camel

Once you have created an EIP diagram and configured all the required properties for each endpoint and processor, you can generate Apache Camel configuration from your diagram.

To generate Camel configuration:

1. Right-click on the EIP editor canvas and select **Export to FUSE Mediation Router...**.
2. In the Save As dialog, select the folder in your workspace where you want to save the Camel configuration file.
3. Edit the given configuration file name if required and click **OK**.

Importing from Camel

FUSE Integration Designer allows you to generate an EIP diagram from a Camel configuration file.

Generating the EIP diagram

To generate an EIP diagram from Camel configuration:

1. In the Package Explorer, right-click on the Camel configuration XML file.

Note

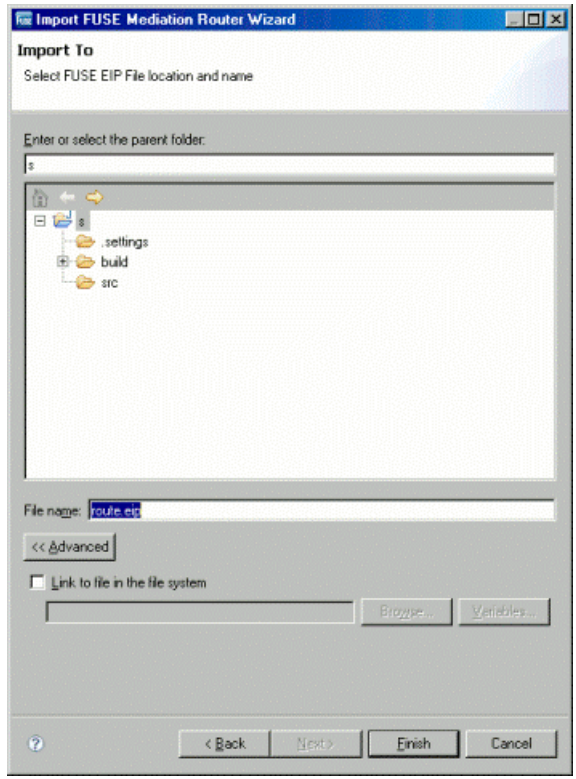
The Import button is disabled for all files other than the Camel Context files.

2. Select **Import**.
3. In the Import wizard, select **FUSE** → **FUSE Mediation Router Configuration** and click **Next**.
4. The input mediation router configuration file to be imported is pre-populated in the File Name field. To select a different file, click **...** or **the down arrow** to select the file from the workspace or local file system.

Note

If the file name does not end with the extension, **.eip**, the Next and Finish buttons are disabled.

5. If the file has unresolved bean references, the next page allows you to add the spring bean xml files to resolve the bean references. Click **Add** → **Workspace** and select the spring file to add. To remove a file that you have already added, click **Remove**.
6. In the next page that comes up, select the folder in which you want to generate the EIP diagram.



7. Enter the default file name (for example, route.eip), and click **Finish**.

The generated EIP diagram opens in the EIP editor. You may need to re-arrange the components and connections to improve the layout of the diagram.

Chapter 9. Running and Debugging

FUSE Integration Designer includes launch configurations that allow you to run and debug EIP diagrams and Camel configuration files.

- *In run mode, the program executes, but you cannot suspend or examine the execution.*
- *In debug mode, you can suspend or resume the execution, inspect the variables, patterns etc and also evaluate the expressions.*

Running EIPs and Camel Configurations

Once you have created an EIP diagram or written or imported a Camel configuration file, you can run the route to ensure that it works.

To run an EIP diagram or a Camel configuration file:

1. Ensure that the EIP editor is not displaying any errors for the EIP that you want to run.
2. Select **Run** → **Run Configurations**.
3. In the Run Configurations dialog, right-click on the node, **FUSE EIP Diagram** or **FUSE Mediation Router Configuration** and select **New**.
4. Type a unique name for the launch configuration in the **Name** field.
5. Click **Browse** to locate the EIP file or Camel Context XML file in your workspace or the local file system.
6. You can also select an input file from the workspace or local file system.
7. Enter a valid endpoint URI. This field gets automatically populated when you select an input file in the previous field.
8. Select the checkbox, **Tracing** to enable the tracer interceptor used for logging the route executions at INFO level.

Note

To configure the Tracer Preferences, go to Setting Tracer Preferences on page 33.

9. Click **Run**.

Note

If you have not set the Mediation Router Runtime location in the Preferences panel, you will not be able to run the configuration. Click on the "Click here to set FUSE Mediation Router Home" link to set the location before trying to run the configuration.

If you are routing messages to and from folders in your workspace, you need to refresh the workspace to see that they have arrived.

Adding External JARs in your Classpath

If your EIP diagram includes any of the following Camel components: JMS endpoints, CXF endpoints, JBI endpoints, HTTP endpoints, and Jetty endpoints, you need to add the required external JARs in the classpath, before you can run the EIP diagram. For example, JMS endpoints require the following JARs: camel-jms-<version>-fuse.jar, camel-spring-<version>-fuse.jar, and activemq-all-<version>-fuse.jar. All these JARs get installed with FUSE Message Broker and are available in the FUSE_MESSAGE_BROKER_HOME/lib folder.

To add external JARs in your classpath:

1. Select **Run** → **Run Configurations**.
2. In the Run Configurations dialog, right-click on the node, **FUSE EIP Diagram** and select **New**.
3. Select the **Classpath** tab.
4. Select **User Entries** and click the **Add External JARs...** button. Browse and select the JAR you want to add.
5. Apply the changes.

Configuring Debug Launch Options

Once you have created an EIP diagram or written or imported a Camel configuration file, you must first configure the debug options before you can debug the route to ensure that it works.

To debug an EIP file or diagram:

1. Ensure that the EIP editor is not displaying any errors for the EIP that you want to debug.
2. Select **Debug** → **Debug Configurations**.
3. In the Debug Configurations dialog, right-click on the node, **FUSE EIP Diagram** and select **New**.
4. Type a unique name for the debug configuration in the **Name** field.
5. Click **Browse** to locate the EIP file in your workspace or local file system.
6. You can also select an input file from the workspace or local file system.
7. Enter a valid endpoint URI. This field gets automatically populated when you select an input file in the previous field.
8. Click **Debug**.

Note

If you have not set the Mediation Router Runtime location in the Preferences panel, you will not be able to debug the configuration. Click on the "Click here to set FUSE Mediation Router Home" link to set the location before trying to debug the configuration.

9. Your program is now launched and the launched process appears in the Debug view.

Working with Breakpoints

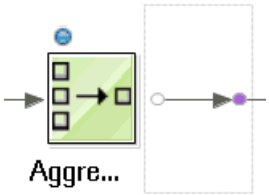
A breakpoint suspends the execution of a program at the location where the breakpoint is set.

Adding breakpoints

To add breakpoints to an EIP diagram:

1. Right-click on an endpoint, pattern or a processor in the EIP editor canvas.
2. Select **Toggle Breakpoint** from the context menu.

A blue circle gets added to the endpoint, pattern or processor, denoting a breakpoint.



Enabling and Disabling breakpoints

Once you have added the breakpoint, you can enable and disable them from the EIP Editor Canvas.

- **Enabling a Breakpoint:** To enable a breakpoint, right click on the location where you have set the breakpoint and select Enable Breakpoint.

  An enabled breakpoint causes a thread to suspend whenever the breakpoint is encountered. Enabled breakpoints are drawn with a blue circle and have a checkmark overlay once successfully set.

- **Disabling a Breakpoint:** To disable a breakpoint, right click on the location where you have set the breakpoint and select Disable Breakpoint.

  A disabled breakpoint will not cause threads to suspend. Disabled breakpoints are drawn with a white circle.

Chapter 10. Creating CXF Services

FUSE Integration Designer allows you to create JAX-WS compliant Web services using FUSE Services Framework (Apache CXF)

Setting FUSE Services Framework (CXF) Preferences

Before you start creating a FUSE Services Framework Web service, you need to set a number of preferences, the most important being the location of your FUSE Services Framework (CXF) home directory.

Setting your FUSE Services Framework home directory

To set your FUSE Services Framework home directory:

1. Select **Window** → **Preferences**.
2. In the Preferences window, select **Progress** → **FUSE**.
3. Select **FUSE Services Framework**.
4. In the FUSE Services Framework Runtime tab, click **Browse** to locate your FUSE Services Framework home directory.

Java-to-Web service settings

Use the Java2WS tab to set default values for when you generate a Web service from a Java service endpoint interface (SEI). These default values are used when you create a Java-first (bottom up) Web service.

You can set the following options:

Default frontend
: Specifies the frontend to use for processing the SEI and generating the support classes. Only JAX-WS is supported in FUSE Integration Designer 1.2.

Default databinding
: Specifies the data binding used for processing the SEI and generating the support classes. The default when using the JAX-WS frontend is Java Architecture for XML Binding (JAXB).

Generate ...
: You can instruct the tool to generate a client, a server, a WSDL file, and wrapper bean and fault beans from the SEI.

Default SOAP binding
: If generating WSDL, both SOAP 1.1 and 1.2 are supported.

Generate separate XSD for the types
: If generating WSDL, you can choose to create a separate XSD file to contain the types.

WSDL-to-Java settings

Use the WSDL2Java tab to set default values for when you generate JAX-WS compliant Java code from a WSDL document. These default values are used when you create a WSDL-first (top-down) Web service.

You can also pass a number of arguments to the xjc tool, which is used in JAXB to convert XML Schema and other schema file types to class representations.

JAX-WS annotation settings

Use the JAX-WS Annotations page to configure which JAX-WS annotations are added to the generated code by default.

The following annotations are available:

`@WebMethod`
: Exposes a method as a Web service operation.

`@Webparam`
: Customizes the mapping of an individual parameter to a Web service message part or XML element.

`@RequestWrapper`
: Specifies the request wrapper bean to be used at runtime.

`@ResponseWrapper`
: Specifies the response wrapper bean to be used at runtime.

`JAX-WS Annotation Processing`
: Enable the checkbox, Enable Annotation Validation Processing to start validating JAX-WS annotations.

Spring configuration settings

Use the Spring Config tab to choose whether CXF configuration should be generated in a `cxf-servlet.xml` file or a Spring application context file.

Creating a Dynamic Web Project

To create a Dynamic Web project:

1. Select **File** → **New** → **Project**.
2. In the Select a project panel of the New Project wizard, select **Web** → **Dynamic Web Project** and click **Next**.
3. In the New Dynamic Web Project panel, enter a project name
4. The "Use default" is enabled by default. You may uncheck this to select a different location (from the workspace).
5. Click **Finish**.
6. If you plan to deploy to an application server, select a target runtime from the drop-down list. See Adding server runtimes on page 21 for details.
7. Select the Dynamic Web Module version as **2.4**.
8. Select the configuration as **CXF 2.x Web service project v2.4**.
9. A new or existing Enterprise Application project (EAR Project) must be associated with your new Web project to facilitate deployment. If you want to override the default settings for the Enterprise Application project, you can do so using the wizard. When your Web project is created at the end of the wizard, the new Enterprise Application project is also created with the name specified in the EAR project field. Note that the default is the name of the web project appended with EAR (unless the ear project was selected when you opened the wizard.)
10. Click **Next**
11. In the Web Module panel, you can accept the default values, or edit them as required. The **Context Root** field refers to the Web application root, which is the top-level directory of your application when it is deployed to the Web server. Click **Finish**.

Creating a Server to Deploy Web Services

The Web service you create must use a server to deploy its services. Example servers that you can use for deploying the service are Apache Jakarta Tomcat, ServiceMix Server or JBoss.

Note

Ensure that your server is started, and you have no exceptions or error messages in your Console view before you deploy your Web services.

Creating an Apache Tomcat Server

If you plan to use Apache Jakarta Tomcat as your server, you must first install and cerate the server before you begin creating your Web service.

Install Apache Jakarta Tomcat from this URL: http://jakarta.apache.org/tomcat

Once you have installed the server, create a Tomcat server in FUSE Integration Designer by doing the following:

1. Select **File** → **New** → **Other** → **Server** → **Server** and click **Next**.
2. Under Apache, select Tomcat v 3.x, 4.x, v5.x, or 6.x as the server type, electing the correct version of Tomcat for your install, and click Next.
3. On the Tomcat server page:
   - Enter a server name.
   - Browse and select where Tomcat is installed.
   - Click Installed JREs, click Add, and enter the appropriate information for the JDK you plan to use for Tomcat. For example, for Tomcat v5.0 the JRE name would be jdk141 and the JRE home would be the home directory for JDK 1.4.1. Click OK twice once you have entered this information. Ensure that the newly added JDK is now selected in the JRE field of the Tomcat server page.

Note

If you point the server at a JRE rather than a JDK the JSPs will not properly compile and will not be able to run on the Tomcat server.

4. Click **Finish**. This creates the server in the workspace.
5. Double click on the tomcat server node in the Servers view. This opens the Overview page, where you can set server properties.
6. Under the Server Locations section, select the checkbox, Use Tomcat Installation to take control over the Tomcat Installation.

Creating a FUSE ESB Server

If you plan to use FUSE ESB Server as your server, you must first install and create the server before you begin creating your Web service.

You can install FUSE ESB Server from this URL:
http://fusesource.com/downloads/

Follow the steps provided here on page 134 to create a new FUSE ESB Server within the FUSE Integration Designer.

Creating a CXF Web service

The CXF Web Service wizard assists you in creating a new Web service, configuring it for deployment, and deploying the Web service to a server. Once your Web service is deployed, the wizard assists you in generating the client proxy and sample application to test the Web service. When you have completed testing, you can publish your Web service to a UDDI Business Registry using the Export wizard.

To create a CXF Web service:

1. Create a FUSE project with the FUSE Services Framework project configuration and the CXF 2.x Web Services project facet selected. See Creating Projects in FUSE Integration Designer on page 23 for details.
2. Select **File** → **New** → **Web Service**.
3. In the Web Services wizard that comes up, you can choose to create either a bottom up (Java first) or a top down (WSDL first) Web service.

Creating a Bottom Up POJO Web Service

When creating a bottom up CXF Web service, you start with a Java class. This approach is also known as "Java first".

Before you create the bottom up Web service, you need to add the Java class that you want to expose as a Web service to your workspace.

To create a bottom up Web Service:

1. Follow the steps described in Creating a CXF Web service on page 117.
2. In the Web Services panel:
   - Select **Bottom up POJO Web service** as the Web service type.
   - Click **Browse** next to the Service implementation field and select the source Java class from your workspace.
   - Use the slider control to select the stages of Web services development that you want to complete.
   - The default server is displayed in the Server field. If you want to deploy your service to a different server, click the link to specify a different server.
   - The default runtime is displayed in the Web Service Runtime field. If you want to deploy your service to a different runtime, click the link to specify a different runtime.
   - The project containing the project selected in your workspace is displayed. To select a different project and EAR, click on the project link. Ensure that the project selected as the Client Web Project is different from the Service Web Project, or the service will be overwritten by the client's generated artifacts.
   - If you want to create a client, select the type of proxy to be generated and repeat the above steps for the client. The client configuration details are enabled when you select the stage of web services development as **Start Service**.
   - **Publish the Web service:** Enable this option to launch the Web Services Explorer to publish your Web service to a UDDI registry.

- **Monitor the Web service:** Enable this option to send the Web service traffic through the TCP/IP Monitor, which allows you to watch the SOAP traffic generated by the Web service and to test this traffic for WS-I compliance. Alternately you can manually set up a TCP/IP monitor as described in Using the TCP/IP Monitor to test Web services.

3. The Java Class Starting Point Configuration panel allows you to generate a service endpoint interface (SEI) from the source Java class. This is the recommended approach to Java first development in CXF. Select the **Use a Service Endpoint Interface** checkbox.

   - **Select SEI from Type Hierarchy:** Select this option and select SEIs from a type hierarchy.
   - **Extract an SEI:** Select this option if you want to extract the SEI from the existing class. Select the members that must be declared in the extracted SEI. You must provide a name for the SEI when using this approach.

   Click **Next**.

4. The Annotate Java Configuration panel allows you to choose which JAX-WS annotations you add to the generated code. The default values are configured via the CXF preferences. Make any required changes and click **Next**.

5. The Java2WS Configuration panel allows you to configure the artifacts that are generated when you complete the wizard. The default values are configured via the CXF preferences. Make any required changes and click **Next**.

6. The Web Service Test page comes up if you selected to test the Web service. Select the test facility for the generated Web service. Click on Launch if you want to open the Web service in the Web Services Explorer and test the web service page right away. Select the operation you want to test, enter the required information, and click Go. The result will display in the Status pane.

   Click **Next**.

7. The next page that comes up is the WSDL2Client Web service configuration page. This page appears only if you have selected to configure the client for the web service to be created.

Select the following options in this page and click **Next** :

Output Directory
: The output directory.

Package Name
: You can change the package name in this field if required.

Specify WSDL Namespace to Package Name Mappings
: Enabling this option brings up a table, where you can review or edit the mappings between packages and namespaces.

Binding Files
: Add or remove the binding files.

8. The next panel that comes up (if you have chosen to Test client option) is Web Service Client Test page: Use this page to select the following options:

   Test the generated proxy
   : Select your test facility as the Web Service Explorer.

   Run test on server
   : Enable this option to start the server for you automatically.

9. The Web Service Publication panel comes up only if you have selected your web service stage as Deploy. You have the option of publishing your service to a UDDI registry or to the Unit Test UDDI Registry.

   Click **Finish**.

The generated Java bean proxy provides a remote procedure call interface to the Web service. The sample Web application demonstrates how to code the proxy file.

Once you have generated your Java client proxy, you may test the methods of the Web service using the Web Services Explorer. Select the operation you want to test, enter the required information, and click Go. The result will display in the Status pane.

Creating a Top Down POJO Web Service

When creating a top down CXF Web service, you start with a WSDL file. This approach is also known as "WSDL first".

Before you create the top down Web service, you need to add the WSDL file from which you want to generate the Web service to your workspace.

To create a top down Web service:

1. Follow the steps described in Creating a CXF Web service on page 117.
2. In the Web Services panel:
   - Select **Top down POJO Web service** as the Web service type.
   - Click **Browse** next to the Service implementation field and select the source WSDL file from your workspace.
   - Use the slider control to select the stages of Web services development that you want to complete.
   - The default server is displayed in the Server field. If you want to deploy your service to a different server, click the link to specify a different server.
   - The default runtime is displayed in the Web Service Runtime field. If you want to deploy your service to a different runtime, click the link to specify a different runtime.
   - The project containing the project selected in your workspace is displayed. To select a different project and EAR, click on the project link. Ensure that the project selected as the Client Web Project is different from the Service Web Project, or the service will be overwritten by the client's generated artifacts.
   - If you want to create a client, select the type of proxy to be generated and repeat the above steps for the client. The client configuration details are enabled when you select the stage of web services development as **Start Service**.
   - **Publish the Web service:** Enable this option to launch the Web Services Explorer to publish your Web service to a UDDI registry.

- **Monitor the Web service:** Enable this option to send the Web service traffic through the TCP/IP Monitor, which allows you to watch the SOAP traffic generated by the Web service and to test this traffic for WS-I compliance. Alternately you can manually set up a TCP/IP monitor as described in Using the TCP/IP Monitor to test Web services.

3. Click **Browse** next to the **Service definition** field and select the source WSDL file from your workspace.

4. Use the slider control to select the stages of Web services development that you want to complete, then click **Next**.

5. In the WSDL2Java Configuration panel, change the output directory and package name if required. You can also fine-tune the mappings of WSDL namespaces to package names.

   Select a WSDL service from the **Service Name** drop-down list and click **Next**.

6. The next WSDL2Java Configuration panel allows you to configure the artifacts that are generated when you complete the wizard. The default values are configured via the CXF preferences. Make any required changes and click **Next**.

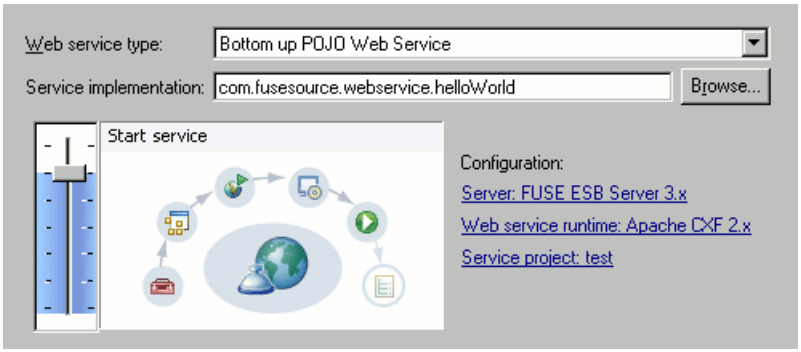
7. The Web Service Publication panel comes up only if you have selected your web service stage as Deploy. You have the option of publishing your service to a UDDI registry. Click **Next**.

8. The Web Service Test page comes up if you selected to test the Web service. Select the test facility for the generated Web service. Click on Launch if you want to open the Web service in the Web Services Explorer and test the web service page right away. Select the operation you want to test, enter the required information, and click Go. The result will display in the Status pane.

   Click **Finish**.

Stages of Web Service Development

When creating either a bottom up or top down Web service, you can use the slider control in the Web services wizard to select the stages of Web service development that you want to complete.



Develop
: Create the service code.

Assemble
: Ensures that the project that will host the Web service or client gets associated to an EAR when required by the target application server.

Deploy
: Creates the deployment code for the service.

Install
: Install the service on the chosen server (for example, Tomcat or JBoss). Selecting this stage enables the Publish the Web service and Monitor the Web service options.

Start
: Start the server once the service has been installed. Selecting this stage enables the client configuration options.

Using the Annotation Properties View

Annotations help specify metadata associated with Web service implementations and also simplify the development of Web services. You can use these annotations to configure bindings, handler chains, set names of portType, service and other WSDL parameters. Annotations are typically used in mapping Java to WSDL and schema, and at runtime to control how the JAX-WS runtime processes and responds to Web service invocations.

The Annotation Properties view in FUSE Integration Designer enables you to specify annotation values for the following types of Java objects:

- types such as a Java class, enum or interface
- methods
- fields representing local instance variables within a Java class
- parameters within a Java method

To work with the Annotation Properties view:

1. Click **Window** → **Show View** → **Other**
2. In the Show View panel that comes up, select **JAX-WS** → **Annotation Properties**. Click OK.
3. This brings up the Annotation Properties view.
4. Open the file you want to define the annotations for in the editor.
5. In the open file, select the java object for which you want to define the annotations.
6. The Annotation Properties view now displays all annotations for the object. You can enable the annotation by checking against the annotation name, and also define its properties (values).

Chapter 11. Working with Java DSL in FUSE Integration Designer

FUSE Integration Designer helps you in defining Enterprise Integration Patterns or routing rules in Java, using a domain specific language (DSL). The routing rules represent the core of a router application and Java DSL is currently the most flexible way to define them.

Some of the benefits of using the Java DSL in FUSE Integration Designer include smart completion of your code, rather than having to mess around with buckets of XML. The Java DSL is also very expressive as you can mix and match your own code within the language for Expression or Predicate evaluations or easily add a custom Processor.

Creating a Java DSL Project

A Java DSL project allows you to define EIPs or routing rules.

To create a FUSE Integration Designer Java DSL project:

1. Select **File** → **New** → **Project**.
2. In the Select a project panel of the New Project wizard, select **FUSE** → **Java DSL Project** and click **Next**.
3. In the New Java DSL Project panel, enter a project name.
4. Fill in the location details if you do not want to store it in the default location and click **Finish**.

The new Java DSL project that is created comprises a src folder, JRE system libraries and the Apache Camel 1.5 JAR files.

Note

By default the JAVA DSL project comprises only the apache-camel jar in its build path. You need to add any other JARs, depending on the routes you want to implement for the project to build. For example, you must add activemq jar if the implemented route consists of a JMS component.

Implementing a RouteBuilder Class

Once you have your JAVA DSL project, you must define routes by implementing a RouteBuilder class. Within the RouteBuilder class that you create, you need to only implement a single method, configure(), which contains a list of routing rules (one Java statement for each rule).The rules themselves are defined using the Domain Specific Language (DSL), which is implemented as a Java API.

To start implementing routes in FUSE Integration Designer, follow the given procedure:

1. Create a new Java class by right clicking on the appropriate location in the Java DSL project in the Package Explorer view and clicking **New Class**.
2. In the New Java Class wizard that comes up, enter all the package and java class details.
3. Click **Finish**.

Note

Switch to the FUSE perspective to view all required panels of the Java DSL project.

4. In the new java class that opens in the editor, you must import the RouteBuilder class using the following command: `import org.apache.camel.builder.RouteBuilder;`

   The org.apache.camel.builder.RouteBuilder class is the base class for implementing your own route builder types.
5. Next, extend the RouteBuilder class.
6. You will now need to implement a single method, configure(), comprising the list of routing rules that you would like to implement.

   A basic Java DSL Syntax looks like this:
   `from("sourceURL").filter(xpath("...").to("targetURL");.`
   Here, from("sourceURL") specifies the source of messages for the routing

rule, and filter() represents the processor to the rule and to("sourceURL") specifies the target for the messages that pass through the rule. .

Note

It is not mandatory to have the filter() processor in your Java DSL Syntax.

7. Save the Java class.

Example 11.1. Java DSL Defining a Route

```
import org.apache.camel.builder.RouteBuilder;

public class MyRouteBuilder extends RouteBuilder {

    public void configure() {
    // Define routing rules here:
    from("activemq:my.queue").split(xpath("//foo/bar")).con
vertBodyTo(String.class).to("file://some/directory")

    // Include more rules, if you like.
    // ...
    }
}
```

In the above example, the message from the queue “my.queue” is split based on the xpath expression, and the output sent to the directory specified. The convertBodyTo(String.class) helps convert the XML DOM object (output provided by the Splitter) into a String.

Deploying a Java DSL Project

Once your Java DSL project and the routes have been defined, you will need to deploy the project into the FUSE ESB container.

To deploy the Java DSL project:

1. Go to the **Servers** view in the Output panel.
2. You will now need to add the Java DSL project to the already added server. To do that, right click on the server node and select **Add and Remove Projects...**.
3. In the Add and Remove Projects panel, select the project that you want to add from the Available projects list and move it to the Configured Projects list.

Note

When you select a project to add to the server, only the projects that are applicable to the type of server will get added to the Configured projects list.

4. Enable the option, if server is started, publish changes immediately. Click **Finish**.
5. Start the server, if it has not been started yet. This will deploy the Java DSL project and also run the application.

Running the Java DSL Application in Standalone Mode

The default way for deploying the Java DSL application is to deploy the project into the FUSE ESB container. To run the Java DSL application in Standalone mode:

1. Implement a RouteBuilder Class. To know more on this, read Implementing a RouteBuilder Class on page 127.

2. Define a Spring file in your application, which includes a Camel context bean and the Package bean containing the route builder.

   Example 11.2. Example of a Spring Definition File

```
                        <?xml version="1.0" encoding="UTF-
8"?>
                      <beans xmlns="http://www.springframe
work.org/schema/beans"
                             xmlns:xsi="ht
tp://www.w3.org/2001/XMLSchema-instance"
                             xsi:schemaLocation=" ht
tp://www.springframework.org/schema/beans http://www.spring
framework.org/schema/beans/spring-beans-2.0.xsd http://act
ivemq.apache.org/camel/schema/spring http://act
ivemq.apache.org/camel/schema/spring/camel-spring-
1.5.0.xsd">
                             <camelContext id="camelroute"
                             xmlns="http://act
ivemq.apache.org/camel/schema/spring">
                          <package>com.progress.javadsl</pack
age>
                             </camelContext>
                             </beans>
```

   In the above example, **com.progress.javadsl** is the package bean that contains the router builder.

3. Add the Spring definition file in the Run Configurations dialog.To know more on this, read Running EIPs and Camel Configurations on page 106.

Note

If you are trying to run the Java DSL application in FUSE Integration Designer (version 1.2), you will also need to manually

add the project (containing the DSL) classpath in the Run Configuration dialog for the configuration file to pick up the packages specified in the project.

.

Chapter 12. Deploying your Applications on FUSE ESB Server

FUSE ESB is an open source integration platform based on Apache ServiceMix that supports JBI and OSGi for use in enterprise IT organizations. FUSE ESB is robust SOA infrastructure that provides a standardized methodology, server, and tools to integrate components in mission-critical applications.

FUSE Integration Designer supports deployment of various types of applications (EIP, Java DSI and CXF) on FUSE ESB Server 3.x and FUSE ESB Server 4.x. This chapter helps you setup your FUSE ESB Server and describes how you can deploy various types of applications on FUSE ESB Server from within FUSE Integration Designer.

Creating a FUSE ESB Server

Create a new FUSE ESB server node to identify the runtime environment that you want to use for deploying your FUSE projects.

To create a server:

1. Ensure to add the Server Runtime using the Preferences panel. To open the Preferences panel, use **Window** → **Preferences** from the main menu. In the panel that opens, select **Server** → **Runtime Environments** and click **Add** to add the server.

2. Right click on the Project Explorer view and select **File** → **New** → **Other...**.

3. In the New wizard that comes up, select **Server** → **Server** and click **Next**.

Note

You can also add your Server from the Servers view. Right-click anywhere on the Server view and select **New** → **Server** .

4. This displays the Define a New Server panel. Enter:

   Server's Host Name
   : Provide the fully qualified DNS name or IP address of the host machine where the server (FUSE ESB Server) is running. By default, this field is pre-filled with the default address: localhost.

   Server Type
   : Select the type of server or test environment where you want to deploy your FUSE projects as either **FUSE** → **FUSE ESB Server 3.x** or **FUSE** → **FUSE ESB Server 4.x**.

   Server Name
   : Enter the name assigned to the server.

   Server Runtime Environment
   : Select the name of the installed server runtime environment which defines the runtime environment of the ESB server for compiling your application.

5. Click **Next**.

6. In the next panel (FUSE ESB) that displays, the connection details are pre-entered. Modify the details if required (for example, you may want to modify the Port Number).

7. Click **Next**.

8. In the Available projects list, select the project that you want to add and click Add. The project appears in the Configured projects list.

Note

When you select a project to add to the server, only the projects that are applicable to the type of server will get added to the Configured projects list.

9. Click **Finish**. A connection to the FUSE ESB Server is created and added as a node in the Servers view.

Starting the FUSE ESB Server

Once you have created a server, a node is added in the Servers view. The added Server's state is **Stopped** by default. You must start the server before trying to deploy your application on the server.

To start FUSE ESB Server:

1. Go to the Servers view in the Output panel.
2. Right click on the server node you want to start and select **Start**.
3. This starts the FUSE ESB Server.
4. To stop a server, right click on the server node and select **Stop**.

Note

Each of the states displays appropriate icons. Based on the connection status, all the deployed projects display either a red or green flag on the project folder icon. Red indicates that there is no connection available for the project, green indicates that the connection is available

Deploying your Projects on FUSE ESB Server

There are three different types of projects that you can deploy on FUSE ESB Server: EIP Project, CXF Project and Java DSL. After you have added a server, you must add the project to the server before deploying the project into the FUSE ESB container.

To deploy the FUSE project:

1. Go to the **Servers** view in the Output panel.
2. Add the FUSE project to the already added server. To do this, right click on the server node and select **Add and Remove Projects...**.
3. In the Add and Remove Projects panel, select the project that you want to add from the Available projects list and move it to the Configured Projects list.

Note

When you select a project to add to the server, only the projects that are applicable to the type of server will get added to the Configured projects list.

4. Click **Finish**.
5. Start the server if not already started.
6. To deploy the FUSE project, right click on the server node and select **Publish**. You can also deploy your projects using the drag-drop feature. Once your project is deployed, the Server state displays as **Synchronized**. If you make changes to the project, the Server state automatically changes to **Republish**.

Note

FUSE ESB Server includes only the camel-core and camel-spring components by default. FUSE Integration Designer supports many more Camel components, and if you are using any of these components, you will need to add these components into your

FUSE ESB Server's SU library. For further reference, read Apache Camel documentation[1] and ServiceMix documentation[2].

[1] http://camel.apache.org/how-to-use-extra-camel-componets-in-servicemix-camel.html
[2] http://servicemix.apache.org/35-using-other-camel-components.html

Defining FUSE ESB Server Preferences

The Servers view displays the current state of all the servers, in addition to the server name and status. Depending on the status of the server and the preference options that are selected for that particular server, FUSE Integration Designer determines the server actions that you may use.

You can modify the server preferences in the Preference page. To view the Server Preference page, double click on the server node in the Servers view. In the Server Preference page that opens, you can modify the following:

1. The first section, General Information, allows you to specify the host name and other common settings:

   - **Server Name:** The name assigned to the server.
   - **Host Name:** The fully qualified DNS name or IP address of the host machine where the server (FUSE ESB Server) is running. By default, this field is pre-filled with the default address: localhost.
   - **Runtime Environment:** The name of the installed server runtime environment which defines the runtime environment of the ESB server for compiling your application.
   - **Open Launch Configuration:** Clicking on this link brings up the Edit Configuration panel, which allows you to configure preferences such as Server runtime, VM and program Arguments, Classpath, Environment and Java launch configurations.

2. The next section, Publishing allows you to modify settings for publishing. Publishing involves copying files (projects, resource files, and server configurations) to the correct location for the server to find and use them.

   Use the following options to either publish your application automatically or manually:

   - **Never Publish Automatically:** Select this option when you want to publish your application manually.

Note

We recommend that you select this option to disable publishing your application every time the resource changes and saving time in the process.

- **Automatically Publish When Resources Change:** This option is checked by default. Enabling this option publishes your application every time the resources change.
- **Publishing Interval (in seconds):** Specifies that all the files should automatically be published at every seconds intervals, where seconds is the number of seconds you have specified in the Publishing interval (in seconds) control. The minimum value for the publishing interval is fifteen seconds. This option is enabled only when you choose the option, Automatically Publish When Resources Change.

3. The third section, Connection Details, allows you to specify the JMX Connection Details, such as Port Number, User Name and Password.
4. The last section, Timeout, allows you to specify the time limit to complete server operations:
   - **Start (in seconds):** Enter the timeout value that the FUSE project will wait for the server to be started. The default is 90 seconds.
   - **Stop (in seconds):** Enter the timeout value that the FUSE project will wait for the server to be stopped. The default is 20 seconds.

Chapter 13. FUSE Messaging Tool

The FUSE Messaging Tool is a graphical tool that lets you explore JMS client functions, properties, and behaviors. You can use the FUSE Messaging Tool to send and receive JMS messages from and to destinations by setting various client properties. This is simply a useful visual messaging testbed that you can launch from within the FID.

The FUSE Messaging Tool helps you create message producers (QueueSenders and Publishers) and message consumers (QueueReceivers and Subscribers); you can also create messages, send the messages to selected queues and topics, and visually inspect the messages after they are delivered.

Switching to the Messaging Perspective

The Messaging perspective includes the Project Explorer, the Servers view and the Problems View. To switch to the Messaging perspective for the first time:

- Click **Window** → **Open Perspective** → **Messaging** on the menu bar.

Note

When the perspective opens, the title bar of the window it is in changes to display the name as **Messaging**. Use these icons in the shortcut bar to quickly switch between perspectives.

Creating a Messaging Project

Your first step in sending and receiving JMS messages is to create a messaging project. After you create the Messaging project, you can view the project in the Project Explorer View. The Messaging Project comprises two nodes: Senders and Listeners.

To create a messaging project:

1. Click **File** → **New** → **Messaging Project**.
2. In the New Messaging Project panel, enter a project name.
3. You will now need to choose a messaging server to identify the runtime environment that you want to use for your messaging projects. To do this, select from the available list of messaging servers in the drop-down list. FUSE Integration Designer currently supports FUSE Messaging Broker. Choosing a messaging server now results in the project getting deployed on the specified server. You can deploy a project only in one messaging server at a time.

   ## Note

   If you have not created a messaging server yet, click the New Server button. See Creating a New Message Server on page 171 for details.

   ## Note

   You can choose a messaging server while adding a destination also.

4. Click **Finish**.

Adding a Destination

A Destination represents a named location to which messages can be sent or received. A Destination must be either a Topic or a Queue (both of which extend the Destination interface). FUSE Integration Designer allows you to add a destination for both the Sender and the Listener.

Adding a Sender to a JMS Destination

JMS senders are grouped by JMS connection under the Senders node (under Messaging Project) in the Project Explorer view. The list of destinations you are sending to is associated with your current workspace and is saved when you exit FUSE Integration Designer and recreated when you restart FUSE Integration Designer.

To add a sender to a JMS destination:

1. In the Project Explorer view, expand the Messaging project. Right-click on the Senders node and select **Add Destination**.
2. In the Add JMS Sender dialog that opens, select the destination type as either Queue or Topic.
3. Enter the name of the queue or topic.
4. Entering a name for the destination enables the Advanced >> button. Click on this button if you want to configure the Sender Properties and ReplyTo Destination.
5. The JMS Sender Properties section has the following parameters, which you can edit:
   - **Time To Live:** Sets the JMS Time-to-live when the JMS message is sent to a FUSE MQ destination. In milliseconds, with 0 indicating no expiration.
   - **Delivery Mode:** Determines the JMS delivery mode used when sending the JMS message to the FUSE MQ Server. This is a required field. Set the values as NON_PERSISTENT, PERSISTENT or Auto-Detect.
   - **Priority:** Sets the JMS Priority when the JMS message is sent to a FUSE MQ destination. You can set the values from 0 to 9.
6. Set the JMSReplyTo destination by clicking on the ... button. In the Choose Destination dialog that comes up, you can choose one of the

destination as ReplyTo or enter a new destination name. A listener will be created automatically under the listeners section when you enter a new destination name.

- **Destination Type:** Select the destination type as either Queue or Topic. Queue is selected by default.
- **Destination Name:** Enter a destination name where the message reply must be sent.

7. Click OK to add the sender to a JMS destination.
8. You can do the following with the added sender:
   - Right-click on the sender in the Project Explorer view and select **Delete** to delete the sender. To prevent accidental deletion of the sender, a Confirm Delete dialog is displayed to confirm whether you really want to delete the sender.
   - Right-click on the sender in the Project Explorer view and select the **Send File...** button. In the dialog that comes up, select the file to be sent. The file selected (if valid) is sent automatically. You can also drag and drop a file from the Project Explorer view onto a sender to perform the Send action.

     The type of messages (files) that you can send are JMS- Text or Standard (default) and XML - Standard or FUSE Message Broker).

Note

If you do not already have a message, use the FUSE Message editor to compose a message.

Adding a Listener to a JMS Destination

JMS listeners are grouped by JMS connection under the Listeners node in the Project Explorer view. The list of destinations you are listening on is associated with your current workspace and is saved when you exit FUSE Integration Designer and recreated when you restart FUSE Integration Designer.

To add a listener to a JMS destination:

1. In the Project Explorer view, , expand the Messaging project. Right-click on the Listeners node and select **Add Destination**

2. In the Add JMS Listener dialog that opens, select the destination type as either Queue or Topic. Queue is selected by default.

3. Enter the name of the queue or topic.

4. Entering a name for the destination enables the Advanced >> button. Click on this button if you want to configure the Listener Properties.

5. The JMS Listener Properties section has the following parameters, which you can edit:

   - **Durable Subscriber:** This option is enabled only for the Topic message type. Enable the checkbox against the Durable Subscriber field to allow creation of durable subscribers. Creating durable subscribers enable the subscribers to receive messages after they recover from offline situations.

   - **Durable Subscriber Name:** Selecting the Durable Subscriber checkbox in the previous field enables the Durable Subscriber Name field. Enter the durable subscription name here.

   - **Message Selector:** Optionally, you can use the Message Selector field to set up a query against header fields and properties to filter the available messages.

6. Click OK. The new listener is added to the list and is available to listen on the JMS destination specified.

7. You can do the following with the added listener:

   - Right-click on the listener in the Project Explorer view and select **Delete** to delete the listener. To prevent accidental deletion of the listener, a Confirm Delete dialog is displayed to confirm whether you really want to delete the listener.

   - Right-click on the listener in the Project Explorer view and select **Unsubscribe and close** to unsubscribe the subscription and remove the destination node from the Project Explorer view. This action is enabled only when the listener is of the Durable Subscriber topic type.

   - Right-click on an active listener in the Project Explorer view and select **Stop listening** to stop listening.

- Right-click on a stopped listener in the Project Explorer view and select **Start listening** to start listening.
- Right-click on the listener in the Project Explorer view and select **Open** to view the messages received by a particular destination. You can also double click on the listener node to view the messages.

Note

If you want to edit the listener properties, you need to delete the destination and re-create it with the required properties.

Sending and Receiving Messages

You can send and listen for messages in the Messaging Project you just created. You can view the messages received in the Message Listener editor. You can do the following:

- Create senders and listeners on JMS topics and queues.
- Drag and drop files from the Project Explorer view onto a sender and the files are automatically sent.
- Edit, save, and reuse received messages

Sending Messages

You can send messages using either of the two procedures described below:

- Use the Send File option in the Project Explorer View.
- Drag and drop a file from the Project Explorer view onto a sender.

To send a message using the Send File option:

1. Right click on the Sender in the Project Explorer view and click the Send File... button
2. In the Open panel that comes up, select the message you want to send.
3. Click Open.

To send a message using the drag-drop option:

- Select a message from the Project Explorer view and drag-drop it on the sender.

Note

When you drag and drop a .txt message, FUSE Messaging Tool sends a text message. When you drag and drop an .xml file.

FUSE Messaging Tool sends a text message with XML content-type.

Viewing the Received Messages

As messages arrive, the listener in the Project Explorer view is displayed in bold and shows the total count of new messages. When you double click on a listener in the Project Explorer view, the Message Listener editor comes up on the right pane and shows the received messages with their tracking IDs and timestamps in the Received Messages list.

Select a specific message in the Message Listener editor to view a tree representing the message headers, properties and body. If there are no properties for a message, the properties node is not displayed; if there is no body for the message, the body node is not displayed.

If you select the following in the tree:

- JMS Headers, you can view the headers as name and value pairs.
- Properties, you can view the properties as name and value pairs.
- Body, depending on the message type, the text and xml messages are displayed in their respective viewers. Selecting a message displays the body of the message by default.

You can perform the following actions on the Message Listener editor:

- Select a message and click **Delete** on the top right corner side to delete the message. To prevent accidental deletion of the message, a Confirm Delete dialog is displayed to confirm whether you really want to delete the message.
- Select a message and click **Save As** to save the message. See Saving Messages on page 151.
- Click **Delete All** to delete all the received messages for the specific listener. To prevent accidental deletion of all messages, a Confirm Delete dialog is displayed to confirm whether you really want to delete all messages.
- By default, 100 messages per listener are stored in the Message Listener editor. To change this value, click **Message History Count** and specify a new count. The value that you set for the history count is globally set to avoid confusion; changing the value for one listener sets it for all listeners.

- Select a message and click **Send JMS Reply** on the top right corner side to send as a reply message. This option is enabled only when you have set a JMSReplyTo property for the selected message.

Saving Messages

You can save messages you view in the Message Listener editor as .message files. You can then edit and resend these saved message files.

To save messages as message files:

1. In the Received Messages table, select the message and click Save As.
2. The Save As dialog box opens.
3. Select a location and enter a file name for the message.

FUSE Integration Designer saves the file as a message file with a .message extension. You can then open the file in the FUSE Message editor.

Chapter 14. Using the FUSE Message Editor

Use the FUSE Message editor to compose message files to use in running, testing, and debugging FUSE applications.

Overview of the FUSE Message Editor

Use the FUSE Message editor to compose messages as text or XML message files for FUSE applications. You can also use these messages to send during running, testing, and debugging.

Once you have created a message, you can use the FUSE Message editor to edit the message headers and parts. You can then save, re-edit, and reuse these messages.

You can create the following types of messages in the FUSE Message Editor:

- Creating Standard messages
- Creating JMS messages
- Creating Standard Text messages
- Creating JMS Text messages
- Creating Standard XML messages
- Creating FUSE Message Broker XML messages

Note

To view the FUSE Message Editor, you must first create a message.

Creating a Standard Message

To create a standard message (message with no body):

1. Select **File** → **New** → **Other...**.
2. In the New wizard that comes up, select **FUSE** → **Message** and click **Next**.
3. In the Create a new Message panel that displays:

   Parent Folder
   : Enter or select a parent folder, where you want to create the message.

   File Name
   : The name of the message to be created. The extension of a message is .message.

4. Click **Next**.
5. In the next panel that displays, select the type of message to create as **Message (Standard)**.
6. Click **Finish**.
7. The new standard message created opens in the Message Editor on the right panel.
8. Click the **Headers** tab to view the Headers section. Use this section add/delete required headers and their values.
9. Click Add to add headers as Name and Value pairs. After you have added a header, you can:

   - Click on a cell to edit it.
   - Click Delete to delete the currently selected property.

10. Select **File** → **Save** to save the standard message file.

Creating a JMS Message

To create a JMS message (JMS message with no body):

1. Select **File** → **New** → **Other...**.
2. In the New wizard that comes up, select **FUSE** → **Message** and click **Next**.
3. In the Create a new Message panel that displays:

   Parent Folder
   : Enter or select a parent folder, where you want to create the message.

   File Name
   : The name of the message to be created. The extension of a message is .message.

4. Click **Next**.
5. In the next panel that displays, select the type of message to create as **Message (JMS)**.
6. Click **Finish**.
7. The new JMS message created opens in the Message Editor on the right panel.
8. Click the **Headers** tab to view the sections: Headers and Properties.
9. Use the **Headers** section to add/delete these header values:

   - **JMSCorrelationID:** Use this header for a client to link messages. Click in the Value field to edit the header-value.
   - **JMSType**: Enter the JMS type value here.

Note

Refer the JMS specification[1] for further details.

[1] http://java.sun.com/j2ee/sdk_1.3/techdocs/api/javax/jms/Message.html

10. Use the **JMSReplyTo** destination value to set where the response should be sent. If you set the JMSReplyTo value when you compose a message, the sender's default JMSReplyTo value is overridden with the new value. If the JMSReplyTo value is set to null for a message, it might be a notification message. You set the value by clicking on the **...** button. In the Choose Destination dialog that comes up, you can choose one of the destination as ReplyTo or enter a new destination name.

    - **Destination Type:** Select the destination type as either Queue or Topic. Queue is selected by default.
    - **Destination Name:** Enter a destination name where the message reply must be sent.

    Use the **Reset** button to clear all the ReplyTo destination values in the message file.

11. Use the **Properties** section to specify any properties that can be set on a JMS message
12. Click Add to add properties as Name and Value pairs. After you have added a header or a property, you can:

    - Click on a cell to edit it.
    - Click Delete to delete the currently selected property.

13. Select **File** → **Save** to save the JMS message file.

Creating a Standard Text Message

A Standard text message has headers and a text/plain body. The body content can be:

- Embedded (or inline)
- A reference to another file

Note

If you reference a file, the input is not static. The referenced file is retrieved each time the message is sent. Therefore, if you modify the referenced file, the message sent is also modified.

To create a text message:

1. Select **File** → **New** → **Other...**.
2. In the New wizard that comes up, select **FUSE** → **Message** and click **Next**.
3. In the Create a new Message panel that displays:

   Parent Folder
   : Enter or select a parent folder, where you want to create the message.

   File Name
   : The name of the message to be created. The extension of a message is .message.

4. Click **Next**.
5. In the next panel that displays, select the type of message to create as **Text (Standard)**.
6. Click **Finish**.
7. The new text message created opens in the Message Editor on the right panel.

8. The **Message** tab displays the body of the message. The new message has no body content. To add content to the body, you can either edit it inline or reference a file.

   From File
   : Enable this checkbox to reference to another file. Click the ... button and select the file to reference. You can either create a new file or select the file from either the local file system or workspace.

   Edit
   : Enable this checkbox and add the text content in the text box provided below.

   Message Type
   : If you want to change the type of message, select from the options available in the drop-down list.

   Click the **Headers** tab to view the sections: Headers. Use this section add/delete required headers and their values.

9. Click Add to add headers as Name and Value pairs. After you have added a header, you can:

   - Click on a cell to edit it.
   - Click Delete to delete the currently selected property.

10. Select **File** → **Save** to save the standard text file.

Creating a JMS Text Message

A JMS text message has headers, properties and a text/plain body. The body content can be:

- Embedded (or inline)
- A reference to another file

Note

If you reference a file, the input is not static. The referenced file is retrieved each time the message is sent. Therefore, if you modify the referenced file, the message sent is also modified.

To create a text message:

1. Select **File** → **New** → **Other...**.
2. In the New wizard that comes up, select **FUSE** → **Message** and click **Next**.
3. In the Create a new Message panel that displays:

   Parent Folder
   : Enter or select a parent folder, where you want to create the message.

   File Name
   : The name of the message to be created. The extension of a message is .message.

4. Click **Next**.
5. In the next panel that displays, select the type of message to create as **Text (JMS)**.
6. Click **Finish**.
7. The new text message created opens in the Message Editor on the right panel.

8. The **Message** tab displays the body of the message. The new message has no body content. To add content to the body, you can either edit it inline or reference a file.

   From File
   : Enable this checkbox to reference to another file. Click the ... button and select the file to reference. You can either create a new file or select the file from either the local file system or workspace.

   Edit
   : Enable this checkbox and add the text content in the text box provided below.

   Message Type
   : If you want to change the type of message, select from the options available in the drop-down list.

   Click the **Headers** tab to view the sections: Headers and Properties.

9. Use the **Headers** section to add/delete these header values:

   - **JMSCorrelationID:** Use this header for a client to link messages. Click in the Value field to edit the header-value.
   - **JMSType**: Enter the JMS type value here.

   ## Note

   Refer the JMS specification[2] for further details.

10. Use the **JMSReplyTo** destination value to set where the response should be sent. If you set the JMSReplyTo value when you compose a message, the sender's default JMSReplyTo value is overridden with the new value. If the JMSReplyTo value is set to null for a message, it might be a notification message. You set the value by clicking on the **...** button. In the Choose Destination dialog that comes up, you can choose one of the destination as ReplyTo or enter a new destination name.

    - **Destination Type:** Select the destination type as either Queue or Topic. Queue is selected by default.

[2] http://java.sun.com/j2ee/sdk_1.3/techdocs/api/javax/jms/Message.html

- **Destination Name:** Enter a destination name where the message reply must be sent.

Use the **Reset** button to clear all the ReplyTo destination values in the message file.

11. Use the **Properties** section to specify any properties that can be set on a JMS message.
12. Click Add to add properties as Name and Value pairs. After you have added a header or a property, you can:
    - Click on a cell to edit it.
    - Click Delete to delete the currently selected property.
13. Select **File** → **Save** to save the JMS text file.

Creating a Standard XML Message

A Standard XML message has headers and a text/xml body. The body content can be:

- Embedded (or inline)
- A reference to another file

Note

If you reference a file, the input is not static. The referenced file is retrieved each time the message is sent. Therefore, if you modify the referenced file, the message sent is also modified.

To create an XML message:

1. Select **File** → **New** → **Other...**.
2. In the New wizard that comes up, select **FUSE** → **Message** and click **Next**.
3. In the Create a new Message panel that displays:

   Parent Folder
   : Enter or select a parent folder, where you want to create the message.

   File Name
   : The name of the message to be created. The extension of a message is .message.

4. Click **Next**.
5. In the next panel that displays, select the type of message to create as **XML (Standard)**.
6. Click **Finish**.
7. The new XML message created opens in the Message Editor on the right panel.

8. The **Message** tab displays the body of the message. The new message has no body content. To add content to the body, you can either edit it inline or reference a file.

   From File
   : Enable this checkbox to reference to another file. Click the ... button to browse and select the file to reference. You can either create a new file or select the file from either the local file system or workspace.

   Edit
   : Enable this checkbox and add the text content in the text box provided below.

   Message Type
   : If you want to change the type of message, select from the options available in the drop-down list.

9. Click the **Headers** tab to view the sections: Headers and Properties. Use this section add/delete required headers and their values.

10. Click Add to add headers as Name and Value pairs. After you have added a header, you can:

    - Click on a cell to edit it.
    - Click Delete to delete the currently selected property.

11. Select **File** → **Save** to save the XML file.

Creating a FUSE Message Broker XML Message

A FUSE Message Broker XML message has headers, properties and a text/xml body. The body content can be:

- Embedded (or inline)
- A reference to another file

Note

If you reference a file, the input is not static. The referenced file is retrieved each time the message is sent. Therefore, if you modify the referenced file, the message sent is also modified.

To create an XML message:

1. Select **File** → **New** → **Other...**.
2. In the New wizard that comes up, select **FUSE** → **Message** and click **Next**.
3. In the Create a new Message panel that displays:

   Parent Folder
   : Enter or select a parent folder, where you want to create the message.

   File Name
   : The name of the message to be created. The extension of a message is .message.

4. Click **Next**.
5. In the next panel that displays, select the type of message to create as **XML (FUSE Message Broker)**.
6. Click **Finish**.
7. The new XML message created opens in the Message Editor on the right panel.

8. The **Message** tab displays the body of the message. The new message has no body content. To add content to the body, you can either edit it inline or reference a file.

   From File
   : Enable this checkbox to reference to another file. Click the ... button to browse and select the file to reference. You can either create a new file or select the file from either the local file system or workspace.

   Edit
   : Enable this checkbox and add the text content in the text box provided below.

   Message Type
   : If you want to change the type of message, select from the options available in the drop-down list.

   Click the **Headers** tab to view the sections: Headers and Properties.

9. Use the **Headers** section to add/delete these header values:

   - **JMSCorrelationID:** Use this header for a client to link messages. Click in the Value field to edit the header-value.
   - **JMSType**: Enter the JMS type value here.

   ## Note

   Refer the JMS specification[3] for further details.

10. Use the **JMSReplyTo** destination value to set where the response should be sent. If you set the JMSReplyTo value when you compose a message, the sender's default JMSReplyTo value is overridden with the new value. If the JMSReplyTo value is set to null for a message, it might be a notification message. You set the value by clicking on the **...** button. In the Choose Destination dialog that comes up, you can choose one of the destination as ReplyTo or enter a new destination name.

    - **Destination Type:** Select the destination type as either Queue or Topic. Queue is selected by default.

[3] http://java.sun.com/j2ee/sdk_1.3/techdocs/api/javax/jms/Message.html

- **Destination Name:** Enter a destination name where the message reply must be sent.

Use the **Reset** button to clear all the ReplyTo destination values in the message file.

11. Use the **Properties** section to specify any properties that can be set on a JMS message
12. Click Add to add properties as Name and Value pairs. After you have added a header or a property, you can:
    - Click on a cell to edit it.
    - Click Delete to delete the currently selected property.
13. Select **File** → **Save** to save the XML file.

Chapter 15. Working with FUSE Message Broker

FUSE Message Broker is an open source JMS message broker and is the JMS platform of choice for scalable, high-performance SOA infrastructure to connect processes across heterogeneous systems.

Starting the FUSE Message Broker

Be sure the FUSE Message Broker is running before executing any of your JMS projects. To start an instance of FUSE Message Broker:

1. In a command prompt or terminal window, change directory to the FUSE Message Broker installation directory (FUSE_MESSAGE_BROKER_HOME).
2. Go to the bin directory (FUSE_MESSAGE_BROKER_HOME/bin).
3. Type the following:
   - **Windows:** activemq.bat
   - **UNIX:** ./activemq

Creating a New Connection to the FUSE Message Server

Create a new connection to the FUSE messaging server to identify the runtime environment that you want to use for your messaging projects.

To create a server:

1. Right click on the Project Explorer view and select **File** → **New** → **Other...**.
2. In the New wizard that comes up, select **Server** → **Server** and click **Next**.

Note

You can also add Servers in the Servers view. Right-click anywhere on the Server view and select **New** → **Server**.

3. This displays the Define a New Server panel. Enter:

   Server's Host Name
   : Provide the fully qualified DNS name or IP address of the host machine where the server (FUSE Message Broker) is running. By default, this field is pre-filled with the default address: localhost.

   Server Type
   : Select the type of server or test environment where you want to publish or test your messaging projects as **FUSE** → **FUSE Message Broker**.

   Server Name
   : Enter the name assigned to the server.

   Server Runtime Environment
   : Select the name of the installed server runtime environment which defines the runtime environment of the message server for compiling your application.

4. Click **Next**.

5. In the next panel (FUSE Message Broker) that displays, the connection details are pre-entered. Modify the details if required. The connection ID must be unique across different connections that you create for the same FUSE Message Broker.

   The checkbox, "Auto Connect if server is running" is checked by default. Keep this if you want to automatically connect to a server (if it is running) while creating the connection to the FUSE Message Server.

6. Click **Next**.

7. Once you have configured a server, you can create a relationship between the projects that contain the files you want to test and your server by can deploying the Messaging Projects on to this server. A warning message informs the user if the selected project has already been deployed in another server.

   In the Available projects list, select the project that you want to add and click Add. The project appears in the Configured projects list.

   ## Note

   When you select a project to add to the server, only the projects that are applicable to the type of server will get added to the Configured projects list.

8. Click **Finish**. A connection to the FUSE message broker is created and added as a node in the Servers view.

Connecting to the FUSE Message Server

Once you have created a server, a node is added in the Servers view. You must connect to the server before starting to work with the messaging application.

Note

If you have enabled the option, "Auto Connect if server is running" while creating the connection to the FUSE Message Server, FUSE Integration Designer automatically connects to the server (if it is running).

To connect to the Messaging Server:

1. Go to the Servers view.
2. Right click on the server node you want to connect to and select **Messaging** → **Connect**.
3. This connects you to the Message Server.
4. To disconnect from a server, right click on the server node and select **Messaging** → **Disconnect**.

Note

When the servers state shows connected, the projects deployed in it can send and receive messages. If the server goes down or the project is removed from the server, the project goes offline and the user cannot send or receive the messages.

Each of the state displays appropriate icons. Based on the connection status, all the deployed projects display either a red or green flag on the project folder icon. Red indicates that there is no connection available for the project, green indicates that the connection is available.