# FME® Server 2013 Reference Manual

SAFE SOFTWARE

E-mail: info@safe.com • Web: www.safe.com

## Document Information

Document Name: FME Server Reference Manual

Version: FME Server 2013

Updated: December 2013

# Contents

# Introduction

This guide provides comprehensive reference information for working with all of the FME Server components. Included in this guide are resources for configuring components using the FME Server web interface, where possible, as well as complete API reference documentation for configuring existing web clients, developing your own custom web clients, and configuring the FME Server Core.

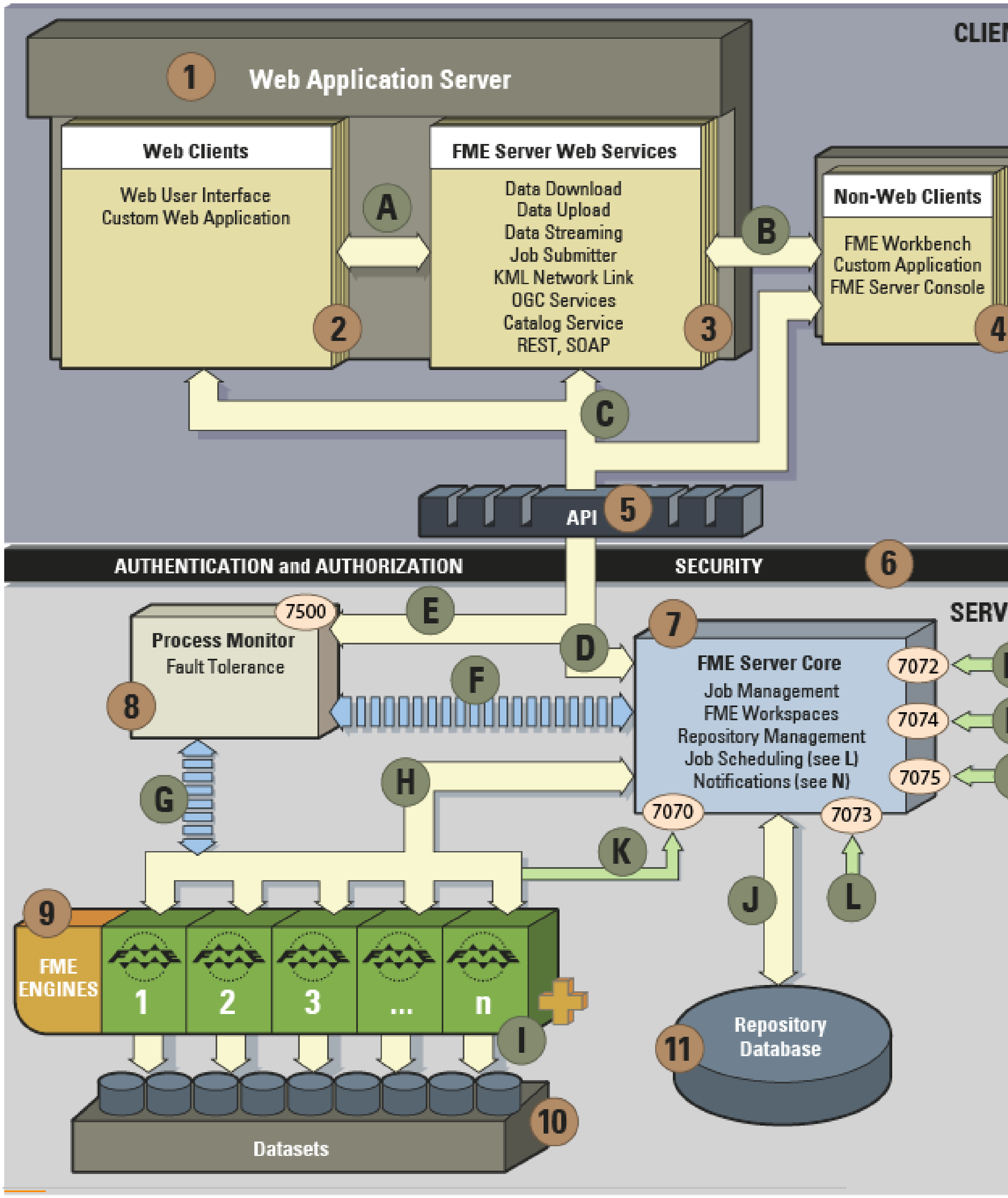
If you are an administrator looking for the most common administration tasks, including installation, refer to the *FME Server Administrator's Guide*. For less common administrative tasks, refer to fmepedia.com. If you are a Workspace Author, refer to the *FME Server Author Tutorial* and the *FME Server Training* course materials.

## FME Server Architecture

FME Server has a number of different components, some of which are considered part of the FME Server core and others that are considered clients of FME Server.

- FME Server Clients, which include:
  - Web Services (for example, FME Server Services)
  - Web Clients of FME Server such as Web User Interface
  - Non-Web Clients of FME Server, which include FME Server Console, FME Workbench, and any application that uses the FME Server API
- Components that are part of the FME Server core include the following:
  - Process Monitor
  - Transformation Manager
  - Repository Manager
  - Repository Database
  - Repository File System
  - FME Engines
  - Scheduling Manager
  - Relay Manager
  - Notification Manager
  - FME Publishers
  - FME Subscribers

Each FME Server component is illustrated in the diagram below and described below.



FME SERVER ARCHITECTURE

**FME Server Architecture**

## FME Server Components

### 1 Web Application Server

A Java Web Application Server is required in order to run the FME Server Web Clients and FME Server Services. Supported Web Application Servers include Tomcat, JBoss and Web Logic.

### 2 Web Clients

The Web User Interface is included with FME Server and can be run in a browser:

Custom web clients can be developed on top of the FME Server REST Service. Clients can also develop against traditional Java, .NET, and C++ APIs.

### 3 FME Server Web Services

FME Server provides predefined services to carry out common tasks. Services provided with FME Server include:

- Data Download
- Data Upload
- Data Streaming
- Job Submitter
- KML Network Link
- OGC Services (WFS and WMS)
- Catalog
- Token Security
- SOAP
- REST
- Notification

## 4 Non-Web Clients

- FME Workbench is the authoring environment for FME Sever but can also be used to submit jobs to FME Server through a number of transformers.
- FME Server Console is a command line interface to FME Server
- Custom clients can be developed on top of the FME Server REST API service, using the FME Server Console or developed against traditional Java, .NET, and C++ APIs.

## 5 API

The low level FME Server API is available in Java, .NET, and C++. All requests are sent to FME Server through the FME Server API. The FME Server REST Service uses the FME Server Java API to send requests to FME Server.

## 6 Security

FME Server provides Authentication and Access Control using the Java Authentication and Authorization Service (JAAS) framework. FME Server also includes a security token service.

## 7 FME Server Core

The FME Server Core manages and distributes job requests (queuing, request routing, scheduling), the repository contents (workspaces, custom formats, custom transformers, data), and notification requests.

## 8 Process Monitor

This component provides fault tolerance functionality ensuring that the FME Server Core and FME Engines remain available to process requests. The Process Monitor also provides a mechanism for managing the FME Server and FME Engine components, including the ability to start, stop, restart and add components.

### 9 FME Engines

FME Engines process job requests by running FME Workspaces. Each FME Engine processes a single request at a time. FME Server processing can be scaled by adding FME Engines to the same computer or to separate computers within a distributed FME Server environment.

In a distributed environment, FME Engines run on a computer or computers that are separate from the FME Server host. Administrators can configure the FME Engines to register with a failover FME Server host, which acts as a backup if the primary FME Server host fails.

### 10 Datasets

Typically, an FME Server job runs a workspace that reads and/or writes data. FME Server administrators must ensure that FME Engines have read access to datasets or databases the workspaces read, and write access to any directories or databases the workspaces write to.

### 11 Repository Database

The FME Server Core uses this database to store job and repository information. The database should never be edited directly, although querying the database for job history and other statistical information is common.

## FME Server Connections

### A

Web Clients use the FME Server Web Services over HTTP. Communication is defined by the Web Services API, SOAP API and REST API.

### B

Non-web Clients use the FME Server Web Services over HTTP. Communication is defined by the Web Services API, SOAP API and REST API.

### C

Web Clients, FME Server Web Services and non-web clients use the FME Server API to communicate over with FME Server over TCP/IP. The requests are sent to the FME Server Core over port 7071. Result messages are returned to the clients over a randomly assigned port created by the FME Server Core.

**D**

The FME Server API can be used to send job and repository requests to the FME Server Core over TCP/IP port 7071.

**E**

The FME Server API can be used to send Deployment Management requests to the Process Monitor over TCP/IP port 7500. The Process Monitor sends shutdown requests to the FME Server Core over TCP/IP port 7071.

**F**

The Process Monitor monitors the FME Server Core processes and restarts them if they halt. The FME Server Core also communicates with the Process Monitor over TCP/IP port 7500.

**G**

The Process Monitor monitors the FME Engine processes and restarts them if they halt.

**H**

Once registered (see K below), the FME Engines communicate with the FME Server Core over TCP/IP ports dynamically determined by the Core.

**I**

The FME Engines read and write data from shared/mounted drives, databases, web services, etc.

**J**

The FME Server Core communicates with the Repository databases via JDBC over TCP/IP.

**K**

The FME Engines perform initial registration with the FME Server Core over TCP/IP port 7070.

**L**

The FME Server API can be used to send job scheduling requests to the FME Server Core over TCP/IP port 7073.

**M**

The FME Server API can be used to send notification requests to the FME Server Core over TCP/IP port 7072.

**N**

The FME Subscribers (Logger, Email, HTTP Push, Apple Push Notification, FTP, Google Cloud Messaging, JMS) perform initial registration with the FME Server Core over TCP/IP port 7074. The FME Subscribers process notifications received by the FME Server Core.

**O**

The FME Publishers (Email, UDP, JMS) perform initial registration with the FME Server Core over TCP/IP port 7075. The FME Publishers relay messages to the FME Server Core.

# API Quick-Find

For developers wishing to access the API documentation contained in this Reference Guide for specific FME Server components, use the following links:

### FME Server Services

### REST Service

- REST API Documentation - See the "FME Server REST API Specifications" document

### FME Server Core

## FME Server Services

FME Server lets your organization address diverse spatial, and non-spatial, data requirements using a single enterprise solution. FME Server provides a Service-Oriented Architecture (SOA) that brings all the capabilities of the FME platform to a server environment, creating a full Spatial Extract, Transform, and Load (ETL) capability.

The capabilities of FME are presented through a number of interfaces, including the Notification Service, web-based services such as the Data Download service, and application interfaces, including the Web User Interface, FME Server Console and FME Workbench. Additionally, programmatic interfaces provide an extra layer of customization.

Through these interfaces, organizations can apply the power of FME at the organization or web level for the first time.

FME Server provides the following services:

- "Data Download Service" on page 15
- "Data Streaming Service" on page 36
- "Job Submitter Service" on page 27
- "KML Network Link Service" on page 45
- "Notification Service" on page 53
- "Catalog Service" on page 147
- "OGC Web Feature Service" on page 102
- "OGC Web Mapping Service" on page 110

The following services act as helper services that provide data or programmatic access to FME Server:

- "Data Upload Service" on page 125
- "Web Connection (SOAP Communication) Service" on page 142
- "REST Service" on page 161
- "Token Service" on page 118

### See Also

For more help in getting started in developing applications with the FME Server services, see the FME Server Developer's Guide document.

## Authentication and Authorization

### Authentication

The FME Server Web Services support two types of authentication:

- HTTP basic authentication (username and password)

  By default, each service is configured to attempt authentication with the FME Server guest user account. Otherwise an HTTP Basic authentication challenge asking for username and password will be made to the requesting client (such as a web browser).

- Authentication with a token provided by the Token service

  A token can be passed to a service using the 'token' query parameter (such as 'token=<value>').

  For example here is a service request with a token included for authentication:

  ```
  http://localhost/fmejobsubmitter/Samples/
  austinDownload.fmw?token=
  c759eaf09e4638954f63ace0ce1b53b40f62ccb7
  ```

### REST

The REST service can also authenticate using a token generated from the token service. The token is passed to the REST service as an HTTP parameter with the name `token`.

### Authorization

Each web service has a client ID that is used to determine which roles and, therefore, which users can access the web service. The client ID is configured in the web application properties file of each service using the parameter `SECURITY_CLIENT_ID`.

# Transformations Services

## Common Request Elements

### *Workspace Published Parameters*

An FME published parameter defined in workspace can be directly passed to a web service as a request parameter. When the web service submits a transformation job to FME Server, the FME Server uses the value of the published parameter to run the workspace.

#### Syntax

The published parameters can be provided to the workspace using the HTTP POST or GET method. Assuming the GET method is used, then the syntax for the published parameter URL parameters is as follows:

```
http://hostname/<service>/<repository>/<workspace>?{Param-
eterName[=ParameterValue]&}
```

#### Example

Here is a URL to invoke the austinDownload.fmw workspace through the data download service (note the URL parameters, such as COORDSYS and it's value LL84):

```
http://hostname/fmedatadownload/Samples/austinDownload.fmw?
COORDSYS=LL84&BBOX_COORDSYS=LL84
 &FORMAT_GENERIC=SHAPE&MINX=-100&MINY=25&MAXX=-90&MAXY=35&TH-
EMES=airports
```

The published parameters from the example above are discussed in the following table. You may also download the austinDownload.fmw workspace from the Samples repository in order to see how the published parameters are configured in the workspace.

| Parameter name | Parameter value | Description |
|---|---|---|
| COORDSYS | LL84 | This is the Writer's *Coordinate System* parameter (i.e. the output coordinate). |

| **Parameter name** | **Parameter value** | **Description** |
|---|---|---|
| BBOX_ COORDSYS | LL84 | This is the Reader's *Search Envelope Coordinate System* parameter for the search envelope provided by the MIN*/MAX* published parameters, which collectively restrict the amount of data that is actually read by FME. Also, this published parameter is used to reproject the search envelope to match the source data's coordinate system in order to perform a hard clip within the workspace using a *Clipper* transformer. |
| FORMAT_ GENERIC | SHAPE | This is the Generic Writer's *Output Format* parameter. |
| MINX | -100 | This is the Reader's *Search Envelope Min X* parameter that will be used to select and clip data. This published parameter is used by both the Reader and a *Clipper* transformer in order to restrict the amount of data that is read and then to hard clip the data respectively. |
| MINY | 25 | This is the Reader's *Search Envelope Min Y* parameter. See MINX for further details. |
| MAXX | -90 | This is the Reader's *Search Envelope Max X* parameter. See MINX for further details. |
| MAXY | 35 | This is the Reader's *Search Envelope Max Y* parameter. See MINX for further details. |
| THEMES | airport | This is the Reader's *Feature Types to Read* parameter. |

***Web Services-Specific Transformation Manager Directives***

Currently the supported TM Directives include the following.

- tm_ttl
- tm_priority
- tm_tag

See "Transformation Manager Directives" on page 213 for details.

***Supported FME Engine Directives***

The following table provides information about the FME Engine directives.

| Directive name | Directive value | Description |
|---|---|---|
| fme_LOG_FILENAME | FME translation log file name. | Specify the FME translation log file name instead of using the default file name. |

## Data Download Service

The Data Download Service provides users with the output from a workspace as a downloadable zip file. Typically the service allows users to specify the data layers, format and coordinate system for the download.

This service is requested using a URL or a form. For example:

```
http://
<host>/fmedatadownload/<repository>/<workspace>.fmw?<param-
eters>
```

A web page opens with a link to a zip file containing the results of the work-space translation. In addition, an email can be sent with a link to the zip file.

***Requirements***

This service works with any workspace that writes single or multiple files. No published parameters are required.

***Note:*** *You may want to publish some parameters from your workspace to control the translation from the URL or form request. For example, feature types to read, output coordinate system, and format (generic writer) are commonly published parameters when using this service.*

**Service Specific Request Parameters**

| Name | Value | Description |
|---|---|---|
| opt_responseformat | xml \| json Default: xml | Defines the language of the response. The text must be all lowercase. |
| opt_geturl | The URL to a dataset | The URL of the source dataset to be used for transformation |
| opt_showresult | true \| false | It indicates whether the XML/JSON responses from these three services include the FME transformation result. The default value is true if this parameter is not present. |
| opt_servicemode | sync \| async | Toggle between synchronous and asynchronous modes of the service. |
| opt_requesteremail | comma-separated e-mail addresses | Addresses to which the notification e-mail messages are sent. |

**Using Remote Data During a Request**

**Using data from HTTP POST body as Reader dataset**

This web service can receive data and override source dataset via HTTP POST. The content of the POST request body will be saved to a temporary file which will be used as the Reader dataset of the workspace. If there is more than one reader in the workspace, then you will need to indicate which reader will use the temporary file. You indicate your choice when publishing the workspace by editing the service registration.

**Using data from HTTP GET as Reader dataset**

This web service can receive data and override Reader dataset via HTTP GET. The dataset should be specified as a URL and passed to service by 'opt_geturl' parameter. Both HTTP and FTP URLs are supported. If there is more than one reader in the workspace, then you will need to indicate which reader will use the temporary file. You indicate your choice when publishing the workspace by editing the service registration.

For example here is a service request that uses a remote KMZ dataset:

```
http://<host>/fmedatadownload/Test/Viewer.fmw?opt_
geturl=http://data.vancouver.ca/download/kml/elementary_
school_boundaries.kmz
```

***Response Elements***

If the response format is specified as XML or JSON, a service response may contain the elements shown in the following table:

| Element | Child Elements | Value | Description |
|---|---|---|---|
| statusInfo | message | message string | service failure message |
| | status | success \| failure | service status |
| | mode | sync \| async | service mode |
| fmeTransformationResult | fmeServerResponse | FME Server response properties | FME Server response |
| | fmeEngineResponse | FME Engine response properties | FME Engine response |
| email | none | Email addresses | Requesters' e-mail addresses |
| url | none | URL string | URL used to download the result dataset (ZIP) |

| Element | Child Elements | Value | Description |
|---|---|---|---|
| jobID | none | job ID | The current job identifier |

### *Web Application Properties*

The following table lists the properties file path and web application properties of the Data Download service.

**Properties file**

```
<WebAppDir>/fmedatadownload/WEB-INF/
conf/propertiesFile.properties
```

| Key Property | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. Can be specified as a network name or an IP.<br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br>Default value = 7071 (the default Transformation Manager listening port) |
| SCHEDULER_PORT | The port on which the Scheduler Manager is listening.<br>Default value = 7073 |

**Properties file**

`<WebAppDir>/fmedatadownload/WEB-INF/conf/propertiesFile.properties`

| **Key Property** | **Description** |
|---|---|
| CONNECTION_ POOL_EXPIRY | The length of time in seconds in which an FME Server connection remains idle before expiring from the connection pool.<br><br>If a value of less than or equal to zero is specified, there is no connection pooling.<br><br>Default value = 300 |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |
| RESULT_DIR | This is the directory where transformation results are stored. |
| MIN_TIME_ BETWEEN_REQS | This is the minimal amount of time that a client must wait before sending another request. If a request is sent before that interval, it will be ignored. The time is specified in milliseconds.<br><br>Default value = 1000 |
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a mes-sageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |

**Properties file**

```
<WebAppDir>/fmedatadownload/WEB-INF/
conf/propertiesFile.properties
```

| Key Property | Description |
|---|---|
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the beginning of each log message. <br><br> Default value = true |
| INCLUDE_THREAD_NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message. <br><br> Default value = false |
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used. <br><br> Default value = true |
| ECHO_LOG_TO_CONSOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log. <br><br> Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one. <br><br> Default value = 3000 |

**Properties file**

```
<WebAppDir>/fmedatadownload/WEB-INF/
conf/propertiesFile.properties
```

| Key Property | Description |
|---|---|
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| ATTACH_TRANSFORMATION_LOG | A boolean value that determines if a transformation log needs to be attached to notification e-mail.<br><br>Default value = true |
| DEFAULT_RESPONSE_FORMAT | The default response format. See "Configuring Service Response Format" on page 1. |
| DEFAULT_USER_ID | The default user ID to log in FME Server. |
| DEFAULT_PASSWORD | The default password for default user ID. |
| SECURE_CLIENT_ADDRESS | A boolean value that determines if token authentication includes the client address. Set it to true if enhanced security is required.<br><br>Default value = false |
| SECURITY_CLIENT_ID | The client ID assigned to this web service. |
| REQUEST_DATA_DIR | The directory which stores request data from HTTP POST body. |
| ENGINE_SUBSECTION | The subsection name used for workspace transformation. |

**Properties file**

```
<WebAppDir>/fmedatadownload/WEB-INF/
conf/propertiesFile.properties
```

| Key Property | Description |
|---|---|
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

```
<WebAppDir>/fmedatadownload/WEB-INF/
conf/propertiesFile.properties
```

| Key Property | Description |
|---|---|
| FAILOVER_SERVER_ NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

**Example 1**

Shows a Data Download service request with two published parameters and specifying XML as response format.

***Request***

http://localhost/fmedatadownload/Samples/austinDownload.fmw?THEMES=airports&FORMAT_GENERIC=SHAPE&opt_responseformat=xml

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>

  <!-- Service Status Info -->
  <statusInfo>
    <status>success</status>
    <mode>sync</mode>
  </statusInfo>

  <!-- Service Return -->
  <fmeTransformationResult>
    <fmeEngineResponse>...</fmeEngineResponse>
    <fmeServerResponse>...</fmeServerResponse>
  </fmeTransformationResult>

  <url>http://localhost/fmedatadownloadresults/FME_3368_
  1245953381697_3368871561115.zip</url>
  <jobID>52</jobID>
</serviceResponse>
```

## Example 2

A Data Download service request with one published parameter, one TM directive and specifying JSON as response format.

***Request***

http:/-
/loca-
lhost/fmedatadownload/Samples/austinDownload.fmw?THEMES=airports&tm_
priority=50&opt_responseformat=json

***Response***

```
{"serviceResponse": {
  "fmeTransformationResult": {
        "fmeEngineResponse": {
        ...
        },
        "fmeServerResponse": {
        ...
      },
      "jobID": 8,
      "statusInfo": {
        "mode": "sync",
        "status": "success"
      },
      "url": "http://localhost/fmedatadownloadresults/FME_5516_
      1258064613849_5516859449883.zip"
    }}
```

## Example 3

A Data Download service request that specifies a requester's e-mail address, service mode as async, and XML as the response format.

***Request***

http://localhost/fmedatadownload/Samples/austinDownload.fmw?opt_serv-icemode=async&opt_requesteremail=joe@safe.com&opt_responseformat=xml

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<serviceResponse>
  <statusInfo>
    <mode>async</mode>
    <status>success</status>
  </statusInfo>
  <email>someone@safe.com</email>
  <jobID>9</jobID>
</serviceResponse>
```

## Example 4

Using the opt_showResult parameter set to false.

***Request***

http://localhost/fmedatadownload/Samples/austinApartments.fmw?opt_show-result=false&opt_responseformat=xml

***Response***

```
<?xmlversion="1.0"encoding="UTF-8"?>
<serviceResponse>
<statusInfo>
<mode>sync</mode>
<status>success</status>
</statusInfo>
<jobID>119</jobID><u-
rl>http://localhost/fmedatadownload/results/FME_7D150E1F_
1317227595851_7460.zip
</url>
</serviceResponse>
```

## Example 5

Using the opt_showResult parameter set to true.

***Request***

http://localhost/fmedatadownload/Samples/austinApartments.fmw?opt_show-result=true&opt_responseformat=json

***Response***

```
{"serviceResponse": {
"statusInfo": {
"mode": "sync",
"status": "success"
},
"fmeTransformationResult": {
"fmeServerResponse": {
"id": "120",
"jobStatus": "SUCCESS",
…
   },
    "fmeEngineResponse": {
    "statusNumber": "0",
      "statusMessage": "Translation Successful",
      …
    }
    },
    "jobID": 120,
    "url": "http://localhost/fmedatadownload/results/FME_
    7D150E1F_1317227687127_7460.zip"
  }}
```

### Job Submitter Service

The Job Submitter service accepts and runs workspace job requests.

This service is requested using a URL or a form. For example,

```
http://
<host>/fmejobsubmitter/<repository>/<workspace>.fmw?<param-
eters>
```

The resulting web page indicates whether the job submission was successful. If the job was submitted successfully, the FME Server transformation results are shown.

#### Service Specific Request Parameters

| Name | Value | Description |
|---|---|---|
| opt_respon-seformat | xml \| json<br>Default: xml | Defines the lan-guage of the response. The text must be all lowercase. |
| opt_geturl | The URL to a dataset | The URL of the source dataset to be used for transformation. |

| Name | Value | Description |
|---|---|---|
| opt_showresult | true \| false | It indicates whether the XML/JSON responses from these three services include the FME transformation result. The default value is true if this parameter is not present. |
| opt_serv-icemode | sync \| async | Toggle between synchronous and asyn-chronous modes of the service. |
| opt_reque-steremail | comma-separated e-mail addresses | Addresses to which the noti-fication e-mail messages are sent. |

**Using Remote Data During a Request**

**Using data from HTTP POST body as Reader dataset**

This web service can receive data and override source dataset via HTTP POST. The content of the POST request body will be saved to a temporary file which will be used as the Reader dataset of the workspace. If there is more than one reader in the workspace, then you will need to indicate which reader will use the temporary file. You indicate your choice when pub-lishing the workspace by editing the service registration.

**Using data from HTTP GET as Reader dataset**

This web service can receive data and override Reader dataset via HTTP GET. The dataset should be specified as a URL and passed to service by 'opt_geturl' parameter. Both HTTP and FTP URLs are supported. If there is more than one reader in the workspace, then you will need to indicate which reader will use the temporary file. You indicate your choice when publishing the workspace by editing the service registration.

For example here is a service request that uses a remote KMZ dataset:

http://localhost/fmejobsubmitter/Test/Viewer.fmw?opt_geturl=http://dat-a.vancouver.ca/download/kml/elementary_school_boundaries.kmz

***Response Elements***

If the response format is specified as XML or JSON, a service response may contain the elements shown in the following table.

| Element | Child Elements | Value | Description |
|---|---|---|---|
| statusInfo | message | message string | service failure message |
| | status | success \| failure | service status |
| | mode | sync \| async | service mode |
| fmeTransformationResult | fmeServerResponse | FME Server response properties | FME Server response |
| | fmeEngineResponse | FME Engine response properties | FME Engine response |
| email | none | E-mail addresses | Requesters' e-mail addresses |
| jobID | none | job ID | The current job identifier |

***Requirements***

The Job Submitter service works with any workspace.

***Web Application Properties***

The following table lists the properties file path and web application properties of the Job Submitter service.

**Properties file**

`<WebAppDir>/fmejobsubmitter/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. Can be specified as a network name or an IP.<br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br>Default value = 7071 (the default Transformation Manager listening port) |
| SCHEDULER_PORT | The port on which the Scheduler Manager is listening.<br>Default value = 7073 |
| CONNECTION_POOL_EXPIRY | The length of time in seconds in which an FME Server connection remains idle before expiring from the connection pool.<br>If a value of less than or equal to zero is specified, there is no connection pooling.<br>Default value = 300 |

**Properties file**

`<WebAppDir>/fmejobsubmitter/WEB-INF/conf/prop-ertiesFile.properties`

| Key Property | Description |
|---|---|
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |
| MIN_TIME_BETWEEN_REQS | This is the minimal amount of time that a client must wait before sending another request. If a request is sent before that interval, it will be ignored. The time is specified in milliseconds.<br><br>Default value = 1000 |
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a messageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message.<br><br>Default value = true |
| INCLUDE_THREAD_NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message.<br><br>Default value = false |

**Properties file**

`<WebAppDir>/fmejobsubmitter/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used.<br><br>Default value = true |
| ECHO_LOG_TO_CONSOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one.<br><br>Default value = 3000 |
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| ATTACH_TRANSFORMATION_LOG | A boolean value that determines if a transformation log needs to be attached to notification e-mail.<br><br>Default value = true |
| DEFAULT_RESPONSE_FORMAT | The default response format. See "Configuring Service Response Format" on page 1. |

**Properties file**

`<WebAppDir>/fmejobsubmitter/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| DEFAULT_USER_ID | The default user ID to log in FME Server. |
| DEFAULT_PASSWORD | The default password for default user ID. |
| SECURE_CLIENT_ADDRESS | A boolean value that determines if token authentication includes the client address. Set it to true if enhanced security is required.<br><br>Default value = false |
| SECURITY_CLIENT_ID | The client ID assigned to this web service. |
| REQUEST_DATA_DIR | The directory which stores request data from HTTP POST body. |
| ENGINE_SUBSECTION | The subsection name used for workspace transformation. |
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

`<WebAppDir>/fmejobsubmitter/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| FAILOVER_SERVER_NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

## Example 1

A Job Submitter service request with one published parameters that specifies XML as the response format.

***Request***

http:/-
/loca-
lhost/fmejobsubmitter/Samples/austinDownload.fmw?THEMES=railroad&opt_
responseformat=xml

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
  <!-- Service Status Info -->
  <statusInfo>
    <status>success</status>
    <mode>sync</mode>
  </statusInfo>

  <!-- Service Return -->
  <fmeTransformationResult>
    <fmeEngineResponse>...</fmeEngineResponse>
    <fmeServerResponse>...</fmeServerResponse>
  </fmeTransformationResult>
  <jobID>53</jobID>
</serviceResponse>
```

## Example 2

A Job Submitter service request that specifies the service mode as async, requester’s e-mail address, and JSON as the response format.

***Request***

http://localhost/fmejobsubmitter/Samples/austinDownload.fmw?opt_serv-
icemode=async&opt_requesteremail=joe@safe.com&opt_responseformat=xml

***Response***

```
{"serviceResponse": {
  "email": "someone@safe.com",
  "jobID": 15,
  "statusInfo": {
    "mode": "async",
    "status": "success"
  }
}}
```

## Data Streaming Service

This service accepts and carries out transformation requests as specified by a workspace, returning the results as a data stream.

> ***Note:*** *You must install this service if you are also installing the KML Network Link service.*

Data Streaming service requests are made as either URL or form requests. For example:

```
http://
<host>/fmedatastreaming/<repository>/<workspace>.fmw?<param-
eters>
```

The resulting dataset is one file only. After the translation has finished, this dataset is streamed with the appropriate content-type (mime-type) over HTTP back to the client making the request. For example, a web browser or Google Earth are clients that could make the request.

### Service Specific Request Parameters

| Name | Value | Description |
|---|---|---|
| opt_responseformat | xml \| json Default: xml | Defines the language of the response. The text must be all lowercase. |
| opt_geturl | The URL to a dataset | The URL of the source dataset to be used for transformation |
| opt_showresult | true \| false | It indicates whether the XML/JSON responses from these three services include the FME transformation result. The default value is true if this parameter is not present. |

## Using Remote Data During a Request

### Using data from HTTP POST body as Reader dataset

This web service can receive data and override source dataset via HTTP POST. The content of the POST request body will be saved to a temporary file which will be used as the Reader dataset of the workspace. If there is more than one reader in the workspace, then you will need to indicate which reader will use the temporary file. You indicate your choice when publishing the workspace by editing the service registration.

### Using data from HTTP GET as Reader dataset

This web service can receive data and override Reader dataset via HTTP GET. The dataset should be specified as a URL and passed to service by 'opt_geturl' parameter. Both HTTP and FTP URLs are supported. If there is more than one reader in the workspace, then you will need to indicate which reader will use the temporary file. You indicate your choice when publishing the workspace by editing the service registration.

For example here is a service request that uses a remote KMZ dataset:

http://localhost/fmedatastreaming/Test/Viewer.fmw?opt_geturl=http://data.vancouver.ca/download/kml/elementary_school_boundaries.kmz

***Response Elements***

If the response format is specified as XML or JSON, a service failure response may contain the elements shown in the following table.

| Element | Child Elements | Value | Description |
|---|---|---|---|
| statusInfo | message | message string | service failure message |
| | status | success \| failure | service status |

| Element | Child Elements | Value | Description |
|---|---|---|---|
| fmeTransformationResult | fmeServerResponse | FME Server response properties | FME Server response |
| | fmeEngineResponse | FME Engine response properties | FME Engine response |

***Requirements***

The streaming service can accept workspaces that write single or multiple files. If the output contains more than one file, the service creates and streams a zip file containing all output files. No published parameters are required.

> ***Note:*** *Output can include workspaces that write HTML, PNG, KML, GeoRSS, GeoJSON, PDF, and others.*

***Web Application Properties***

The following table lists the properties file path and web application properties of the Data Streaming service.

**Properties file**

`<WebAppDir>/fmedatastreaming/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. You can specify this property as a network name or an IP.<br>Default value = localhost |

**Properties file**

`<WebAppDir>/fmedatastreaming/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br><br>Default value = 7071 (the default Transformation Manager listening port) |
| CONNECTION_POOL_EXPIRY | The length of time in seconds in which an FME Server connection remains idle before expiring from the connection pool.<br><br>If a value of less than or equal to zero is specified, there is no connection pooling.<br><br>Default value = 300 |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |
| MIN_TIME_BETWEEN_REQS | This is the minimal amount of time that a client must wait before sending another request. If a request is sent before that interval, it will be ignored. The time is specified in milliseconds.<br><br>Default value = 1000 |

**Properties file**

`<WebAppDir>/fmedatastreaming/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a messageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message.<br><br>Default value = true |
| INCLUDE_THREAD_NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message.<br><br>Default value = false |
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used.<br><br>Default value = true |

**Properties file**

`<WebAppDir>/fmedatastreaming/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| ECHO_LOG_TO_CONSOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one.<br><br>Default value = 3000 |
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| DEFAULT_RESPONSE_FORMAT | The default response format. See "Configuring Service Response Format" on page 1. |
| DEFAULT_USER_ID | The default user ID to log in FME Server. |
| DEFAULT_PASSWORD | The default password for default user ID. |
| SECURE_CLIENT_ADDRESS | A boolean value that determines if token authentication includes the client address. Set it to true if enhanced security is required.<br><br>Default value = false |
| SECURITY_CLIENT_ID | The client ID assigned to this web service. |
| REQUEST_DATA_DIR | The directory which stores request data from HTTP POST body. |

**Properties file**

*<WebAppDir>*`/fmedatastreaming/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| ENGINE_SUBSECTION | The subsection name used for workspace transformation. |
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

`<WebAppDir>/fmedatastreaming/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| FAILOVER_SERVER_ NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

**Example 1**

A Data Streaming service failure response in XML format

***Request***

http://localhost/fmedatastreaming/Samples/austinApartments.fmw?opt_
responseformat=xml

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
  <!-- Service Status Info -->
  <statusInfo>
    <status>failure</status>
    <message> FME Server Transformation Error.</message>
  </statusInfo>
  <fmeTransformationResult>
    <fmeEngineResponse>...</fmeEngineResponse>
    <fmeServerResponse>...</fmeServerResponse>
  </fmeTransformationResult>
</serviceResponse>
```

## Example 2

A Data Streaming service failure response in JSON format

***Request***

http://localhost:/fmedatastreaming/Samples/austinApartments.fmw?opt_
responseformat=json

***Response***

```
{"serviceResponse": {
  "fmeTransformationResult": {
    "fmeEngineResponse": {
    ...
    },
    "fmeServerResponse": {
    ...
    },
    "statusInfo": {
      "message": "FME Server Transformation Error.",
      "status": "failure"
    }
    }}
```

## KML Network Link Service

The KML Network Link service accepts and carries out transformation requests as specified by a workspace, returning the results as a KML Network Link.

> ***Note:*** *If you install this service, you must also install the Data Streaming service.*

This service can be requested using a URL or form. For example,

```
http://<host>/fmekmllink/<repository>/<workspace>.fmw?<param-
eters>
```

The resulting response is a KMZ file that contains a KML Network Link. This link's URL points to the Data Streaming service and references the requested workspace. No translation is performed to provide the KMZ file.

### *Requirements*

This service works with any workspace that writes KML and KMZ files. The following published parameters should exist in the workspace:

- **bboxEast**: Bounding Box East
- **bboxWest**: Bounding Box West
- **bboxNorth**: Bounding Box North
- **bboxSouth**: Bounding Box South

When the KML Network Link layer is enabled, Google Earth passes the extents of its current view window to these parameters. The coordinates are in lat-long. A workspace can use these parameters to read the source data with spatial queries.

### *Response Elements*

If the KML Network Link service successfully processes the request, it returns a KML document enclosed in KMZ format to the client as a data stream. The KML document contains a NetworkLink element where Link/href is set to a URL. This URL invokes the FME Server Data Streaming service with the provided repository and workspace name. For a complete reference of the KML format, refer to: http://code.google.com/apis/kml/documentation/

If the response format is specified as XML or JSON, a service failure response may contain the elements shown in the following table.

| Element | Child Elements | Value | Description |
|---|---|---|---|
| statusInfo | message | message string | service failure message |
| | status | success \| failure | service status |

***Web Application Properties***

The following table lists the properties file path and web application properties of the KML Network Link service.

| **Properties file** | |
|---|---|
| `<WebAppDir>/fmekmllink/WEB-INF/conf/propertiesFile.properties` | |
| **Key Property** | **Description** |
| SERVER_NAME | The computer where the Transformation Manager is running. You can specify this property as a network name or an IP.<br><br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br><br>Default value = 7071 (the default Transformation Manager listening port) |

**Properties file**

`<WebAppDir>/fmekmllink/WEB-INF/conf/prop-ertiesFile.properties`

| Key Property | Description |
|---|---|
| CONNECTION_POOL_ EXPIRY | The length of time in seconds in which an FME Server connection remains idle before expiring from the connection pool.<br><br>If a value of less than or equal to zero is specified, there is no connection pooling.<br><br>Default value = 300 |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |
| MIN_TIME_BETWEEN_ REQS | This is the minimal amount of time that a client must wait before sending another request. If a request is sent before that interval, it will be ignored. The time is specified in milliseconds.<br><br>Default value = 1000 |
| STREAM_SERVICE | The name of FME Data Streaming service.<br><br>Default value = fmedatastreaming |
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a messageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |

**Properties file**

`<WebAppDir>/fmekmllink/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message.<br><br>Default value = true |
| INCLUDE_THREAD_NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message.<br><br>Default value = false |
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used.<br><br>Default value = true |
| ECHO_LOG_TO_CONSOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one.<br><br>Default value = 3000 |

**Properties file**

`<WebAppDir>/fmekmllink/WEB-INF/conf/prop-`
`ertiesFile.properties`

| Key Property | Description |
|---|---|
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| SERVICE_DESCRIPTION | The description of the KML network link service.<br><br>Default value = A KML Link translation processed by FME Server |
| VISIBILITY | Visibility.<br><br>Default value = 0 |
| VIEW_REFRESH_MODE | View refresh mode.<br><br>Default value = onStop |
| VIEW_REFRESH_TIME | View refresh time.<br><br>Default value = 7 |
| REFRESH_MODE | Refresh mode.<br><br>Default value = onInterval |
| REFRESH_INTERVAL | Refresh interval.<br><br>Default value = 60 |
| DEFAULT_RESPONSE_FORMAT | The default response format. See "Configuring Service Response Format" on page 1. |
| DEFAULT_USER_ID | The default user ID to log in FME Server. |
| DEFAULT_PASSWORD | The default password for default user ID. |

| **Properties file** | |
|---|---|
| `<WebAppDir>/fmekmllink/WEB-INF/conf/propertiesFile.properties` | |
| **Key Property** | **Description** |
| SECURE_CLIENT_ADDRESS | A boolean value that determines if token authentication includes the client address. Set it to true if enhanced security is required.<br><br>Default value = false |
| SECURITY_CLIENT_ID | The client ID assigned to this web service. |
| ENGINE_SUBSECTION | The subsection name used for workspace transformation. |
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

`<WebAppDir>/fmekmllink/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| FAILOVER_SERVER_ NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

**Example 1**

A KML Network Link request with a success response in the KML format.

***Request***

http://localhost/fmekmllink/Samples/austinApartments.fmw

***Response***

```
<?xml version="1.0" encoding="UTF-8" ?>
  <kml xmlns="http://earth.google.com/kml/2.1">
    <NetworkLink>
    <name>KML Link Translation: aus-
    tinApartments.fmw</name>
    <visibility>0</visibility>
    <description>A KML Link document processed by FME
    Server</description>
      <Link>
      <viewRefreshMode>onRegion</viewRefreshMode>
      <viewRefreshTime>7</viewRefreshTime>
      <viewFormat>BBOX=[bboxWest],[bboxSouth],[bboxEast],
      [bboxNorth]</viewFormat>
      <href>http://S-
      HOCK-
      WAVE:8080/fmedatastreaming/Samples/austinApartments.fmw</href>
      <refreshMode>onInterval</refreshMode>
      <refreshInterval>60</refreshInterval>
    </Link>
  </NetworkLink>
</kml>
```

## Example 2

A KML Network Link request with a failure response in the XML format:

***Request***

http://localhost/fmekmllink/Samples/austinApartments2.fmw?opt_respon-
seformat=xml

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
  <statusInfo>
    <message>Workspace 'austinApartments2.fmw' does not
    exist.</message>
    <status>failure</status>
  </statusInfo>
```

```
</serviceResponse>
```

### Example 3

A KML Network Link request with a failure response in the JSON format:

***Request***

http://localhost/fmekmllink/Samples/austinApartments2.fmw?opt_responseformat=json

***Response***

```
{"serviceResponse": {"statusInfo": {
  "message": "Workspace 'austinApartments2.fmw' does not
  exist.",
  "status": "failure"
}}}
```

## Notification Service

FME Server notification can be used to create topics about which you want to notify applications (or people), subscribe clients, publish messages, and have messages delivered over clients' protocol of choice (such as HTTP, email, and so on).

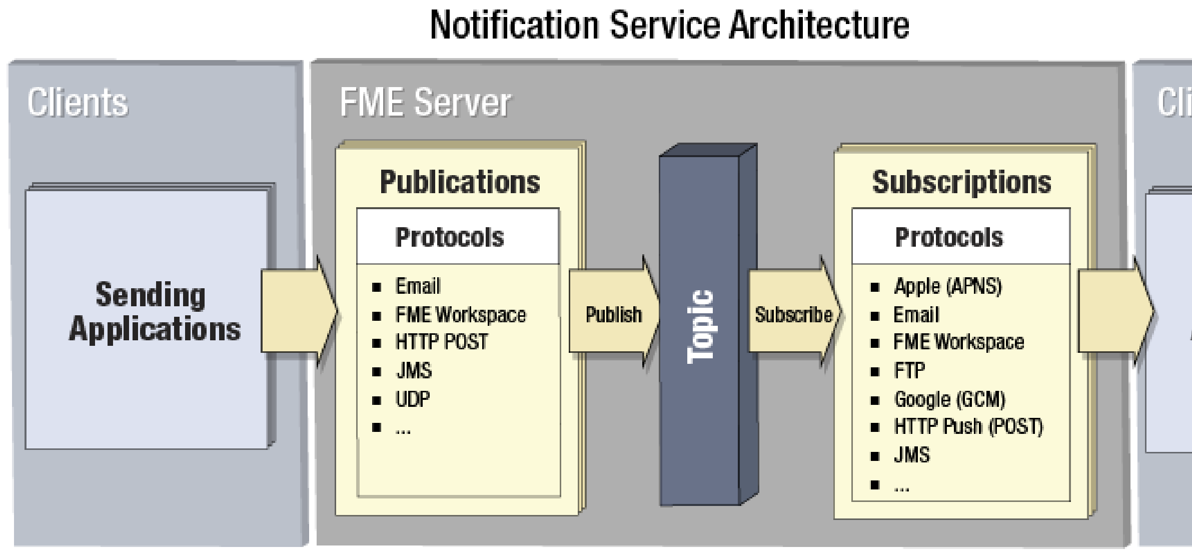
FME Server notification delivers notifications to clients using a *push* mechanism that eliminates the need to periodically check or *pull* for new information and updates. You can use FME Server notification to create event-driven workflows.

Using FME Server notification requires a few simple steps:

1. Create a topic.
2. Add subscribers to a topic.
3. Publish messages or send out notifications.

### Architecture

The FME Server API supports publishing notifications to FME Server. Notification messages consist of a topic and a body.



Topics define how the publisher and subscriber should communicate. At a high level, it is an agreement between the publisher and subscriber. Subscribers can be configured in specific ways to perform specific operations for specific topics. Publishers can publish notification messages with specific content for specific topics. It's important to have Subscribers and Publishers configured appropriately so they both communicate each other in the expected way for a given topic.

For example, the handling of success and failure topics via the Email Subscriber could be different. For success, this might mean the original requester is sent an email that their transformation was successful. For failure, it might mean the original requester is sent an email that their transformation failed and in addition send an FME Server administrator an email with error information for troubleshooting.

## Notification Manager Directives

The following directives are supported by the Notification Manager:

- **jobsuccess_topic** - The name of the topic to be notified when a job succeeds.

- **jobfailure_topic** - The name of the topic to be notified when a job fails.

All other notification manager directives are passed to the Notification Manager and are available in the message received by the Subscriber.

## Default Topics and Subscribers

### Topics

The following topics are shipped with FME Server to support FME Server operations:

- JOB_FAILURE – A notification for this topic is generated when the FME transformation fails.
- JOB_SUCCESS – A notification for this topic is generated when the FME transformation succeeds.
- SAMPLE_TOPIC – This is a sample topic. By default, no notifications are generated for this topic. It's used for testing notifications.

***Note:*** *Currently, only the Data Download Service and Job Submitter Service are configured to publish notifications to the JOB_FAILURE and JOB_SUCCESS topics. However, the E-mail Subscriber needs to be configured before the messages will be sent.*

### Subscribers

The following subscribers are shipped with FME Server:

- Email_JobFailure – This is a subscriber of the JOB_FAILURE topic. It is configured to send emails to the specified recipients when an FME transformation fails. Currently only Data Download Service and Job Subsmitter Service will trigger a notification to this topic for non-blocking jobs that utilize email.
- Email_JobSuccess –This is a subscriber of the JOB_SUCCESS topic. It is configured to send emails to the specified recipients when an FME transformation fails. Currently only Data Download Service and Job

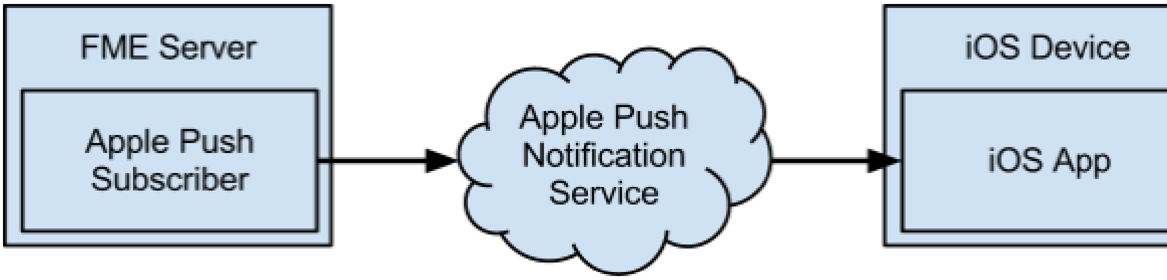
Submitter Service will trigger a notification to this topic for non-blocking jobs that utilize email.

- Logger_Default – This is a subscriber of the JOB_SUCCESS, JOB_FAILURE and SAMPLE_TOPIC topics. It is configured to log out all notifications published to the subscribed topics.

## Subscribers

### *Apple Push Subscriber*

The Apple Push subscriber is a notification subscriber that allows message delivery to an Apple iOS device, such as an iPhone or iPad. The subscriber pushes messages from FME Server to an iOS app installed on the target iOS device by means of a cloud service—the Apple Push Notification Service.



**Note:** *The Apple Push subscriber is one of two subscribers that can push notifications to a mobile device. For notifying Android devices, see "Google Cloud Messaging Subscriber" on page 75.*

This section explains the steps for setting up the Apple Push subscriber:

- "Requirements" on the facing page
- "Configuring the Subscription" on page 59

### Demonstration App

FME Alerts is a demonstration app that showcases the capabilities of FME Server using an Apple Push subscriber. The app is available for free on the Apple App Store.

#### Requirements

When configured, an Apple Push subscriber can send notifications to your target app installed on an iOS device. To accomplish this scenario, the Apple Push subscriber uses the Apple Push Notification Service and two requirements from your target app, both of which require developer access:

1. The iOS app's SSL keystore—Provides the means to establish a secure, authenticated connection to Apple Push Notification Service. For more information, see Provider-to-Service Connection Trust in the Mac Developer Library Local and Push Notification Programming Guide.
2. The iOS device's unique device token—Provides the Apple Push Notification Service with the target device for sending notifications. For more information, see Token Generation and Dispersal.

***Note:*** *The above requirements are unique for each iOS app, and cannot be shared between different apps. When configuring the Apple Push subscriber to work with a new app, a new SSL keystore and new device tokens must be acquired.*

***Note:*** *The Apple Push subscriber requires use of the outbound TCP port 2195 to send notifications. Ensure that the firewall on the FME Server host is properly configured to allow for communication using this port.*

### Acquiring the SSL Keystore

The SSL keystore (.p12 file) contains both a certificate and private key, and is required to establish a connection between the subscriber and the Apple Push Notification Service.

The keystore is associated with the iOS app developer, and generating the keystore is a one-time process.

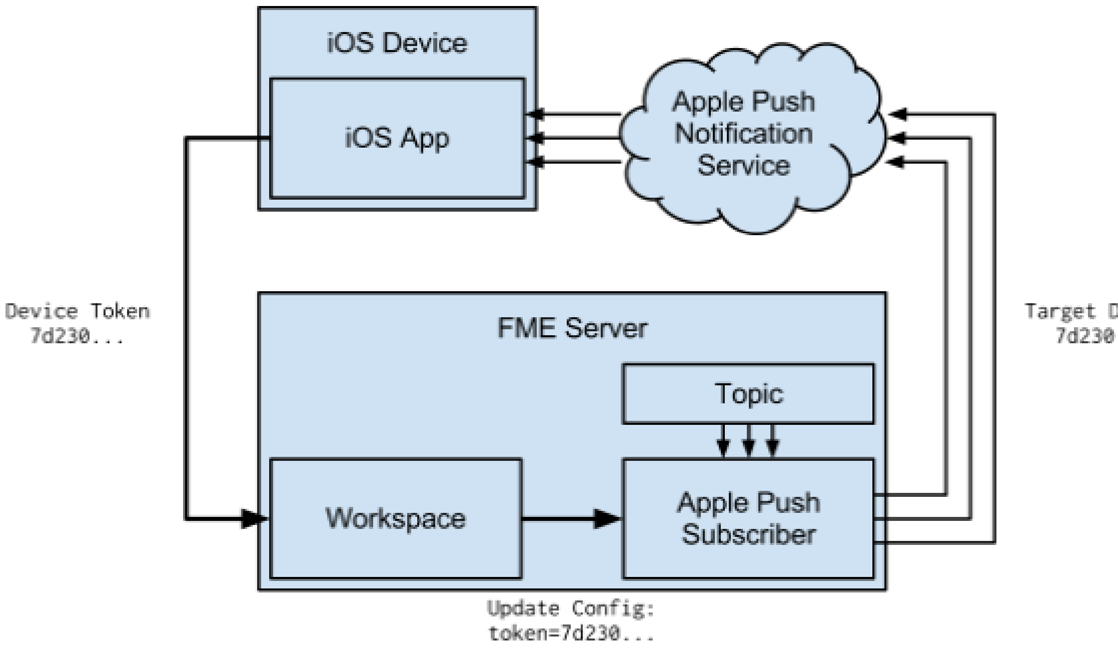
1. Access the iOS Dev Center.
2. Create the SSL certificate and key as described in Creating the SSL Certificate and Keys. When completing the steps, a new SSL certificate and key is added to Keychain Access.
3. In Keychain Access, expand the certificate to reveal the private key.
4. Select both the certificate and private key, and select File > Export Items to export to a keystore (.p12 file). A password will be requested to protect the keystore.

The SSL keystore file and password is required when configuring the subscriber.

### Acquiring the Device Token for the Target Device

Each iOS device can register itself to the Apple Push Notification Service, and when doing so, associates itself with a device token, which is a 64-digit hexadecimal number that uniquely identifies the iOS device.

The iOS app on the device must distribute its device token to FME Server. Typically, the device token is sent to a processing workspace on FME Server that records the token and updates the subscriber. This flow is illustrated below.



An iOS app can register to the Apple Push Notification Service using method `registerForRemoteNotificationTypes:` , and the device token is returned in the app delegate method `appli-cation:didRegisterForRemoteNotificationsWithDeviceToken:`. This process is described in Registering for Remote Notifications.

**What's Next?**

- "Configuring the Subscription" below

**Configuring the Subscription**

**To Create an Apple Push Subscription**

1. In the FME Server Web User Interface, select Notifications on the left. Then, select the Subscriptions tab, and click New.
2. Provide a name for the Subscription.

3. Specify the topics to subscribe to. When the Subscription receives messages from these topics, an Apple Push notification is sent to the target devices.
4. Beside Protocol, specify Apple Push Notification.
5. Specify the Apple Push fields for the Subscription (see below).
6. Click OK to create the Subscription.

| Apple Push Field | Description |
|---|---|
| **Keystore Path** | The path to the SSL keystore .p12 file, used for creating a secure, authenticated connection to the Apple Push Notification Service. |
| **Keystore Password** | The password used to access the keystore. |
| **Service Type** | The keystore environment type (sandbox or production). |
| **iOS Device Tokens** | The list of target iOS devices to receive notifications. Use a comma to separate multiple devices. |
| **Content Template** | The notification content template. |

## Overriding Behavior Using Notification Keywords

By default, an Apple Push Subscriber configuration allows for delivery of notifications to a fixed set of iOS devices, with its message formatted by the prescribed content template. The Apple Push subscriber also allows you to specificy target iOS devices and message format in the incoming notification. The notification keywords that enable this overriding behavior are `apns_token` and `subscriber_content`.

- `apns_token`—Overrides the ‘iOS Device Tokens’ field in an Apple Push subscription. Use a comma to separate multiple devices.
- `subscriber_content`—Overrides the ‘Content Template’ field in an Apple Push subscription.

### Examples

The following Apple Push Subscriber examples are based on this default configuration:

| | |
|---|---|
| **iOS Device Tokens** | `123456,abcdef` |
| **Content Template** | `Message: {msg}` |

| Notification | Target Device(s) | Message Sent to Device(s) |
|---|---|---|
| `{ "msg" : "Hello, world" }` | `123456` `abcdef` | Message: Hello, world |
| `{ "apns_token" : "987654", "msg" : "Hello, 987654" }` | `987654` | Message: Hello, 987654 |
| `{ "subscriber_content" : "Alert", "msg" : "Ignored" }` | `123456` `abcdef` | Alert |

### Demonstration App

FME Alerts is a demonstration app that showcases the capabilities of FME Server using an Apple Push subscriber. The app is available for free on the Apple App Store.

#### *Email Subscriber*

Email subscriber receives notifications from FME Notification Service and delivers emails based on subscribers.

## Keywords

The subject, recipients and attachment of an outgoing email can be configured by the following reserved keywords:

- email subject
- email_to
- email_cc
- email_from
- email_replyto
- email_attachment
- subscriber_content

For example, a notification message is constructed as below:

```
{
  "email_subject": "This is a sample email",
  "email_to": "recipient@example.com",
  "email_attachment": "\\myserver\attachment.txt",
  "BodyContent1": "This is a paragraph",
  "BodyContent2": "This is another paragraph"
}
```

Upon receiving this notification, the email subscriber will send an email message to `recipient@example.com`, with subject “This is a sample email” including an attachment, `attachment.txt`.

If a notification message is sent from FME REST API and it contains unstructured text message, the value of `subscriber_content` keyword contain the content of the text message.

## Subscriber

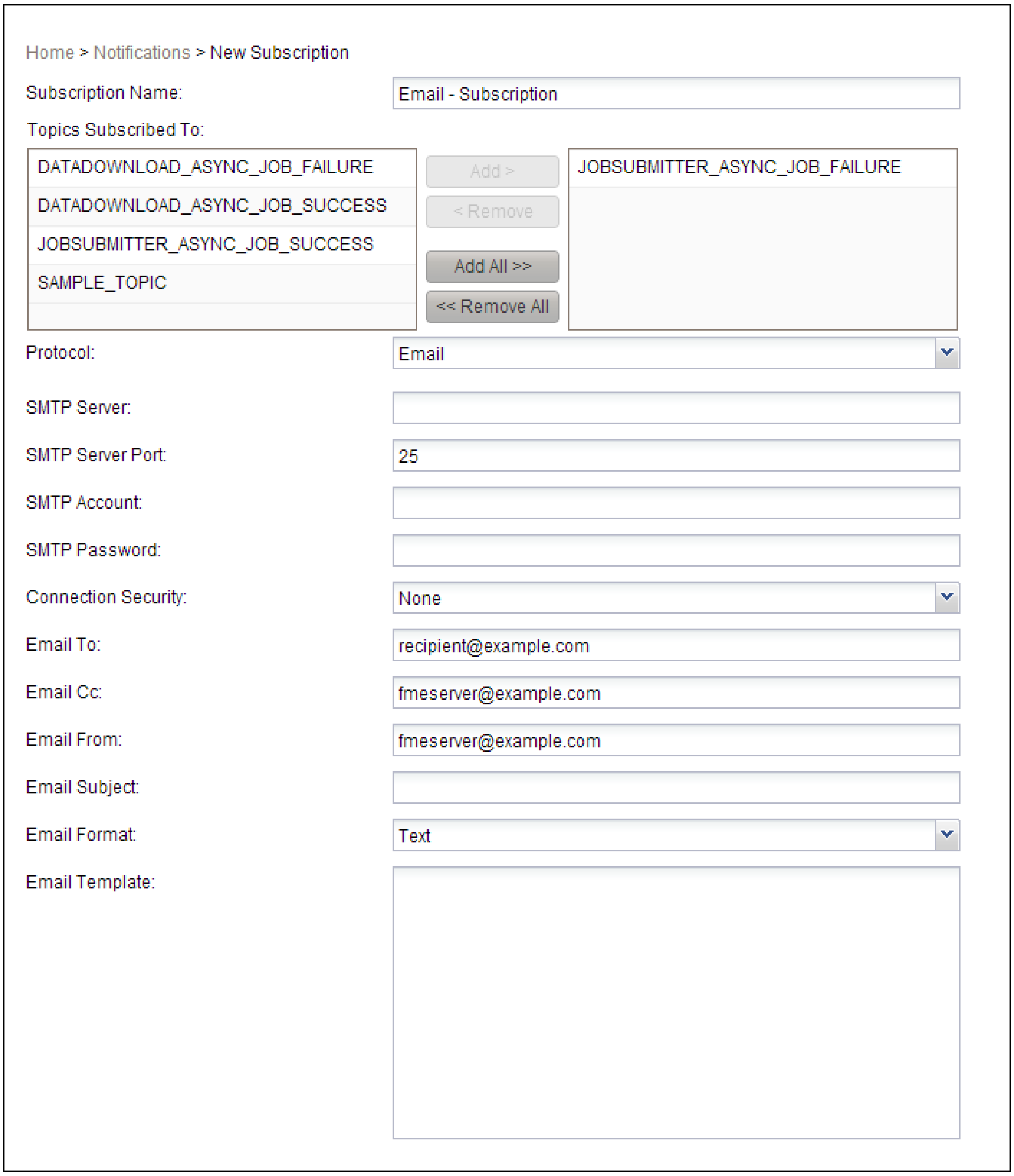
An email subscriber is a configuration required to deliver email messages. The configuration contains information on the email server (SMTP) connection, authentication, default email subject, recipients, email body template and so on. A complete list of email subscriber properties follows:

- **SMTP Server** - Mail exchanger server host name or IP address used for sending email.
- **SMTP Server port** – Mail exchanger TCP port used for sending email.
- **SMTP Account** – If the SMTP server requires authentication, this is the user name.
- **SMTP Password** – If the SMTP server requires authentication, this is the user password.
- **SMTP Connection Security** - The encryption mechanism used for the connnection, if any.
- **Email To** - Corresponds to the To field in standard email messages.
- **Email CC** – Corresponds to the CC field in standard email messages.
- **Email From** – Corresponds to the From field in standard email messages, if users reply to the notification email message it will be directed to this email address.
- **Email Subject** – Corresponds to the Subject field in standard email messages.
- **Email Format** - The format of the email, either plain text or HTML.
- **Email Content Template** – The message body to use for delivery. This is a template that allows the use of various tags that are replaced with dynamic information before delivery. When the message is provided by an FME Server web service, such as the Data Download or Job Submitter service, the following tags are supported:
  - {StatusMessage} – The status message of the transformation job that completed.
  - {id} – The transformation job's ID.
  - {urlPrefix} – Prefix of the URL used to download any job transformation result files.
  - {ResultRootDir} – Root directory fragment part of the URL used to download any job transformation result files.

    - {@getFilename(@zip(OutputLocation))} – Retrieves the filename fragment part of the URL used to download any job transformation result files. The result files are compiled into a zip file.

- **Assigned Topics** - The FME Server topics for which the email notifications will be delivered.

The following dialog allows the FME Server administrator to configure the email subscription using the Web User Interface.



Home > Notifications > New Subscription

| | |
|---|---|
| Subscription Name: | Email - Subscription |
| Topics Subscribed To: | |

| | | |
|---|---|---|
| DATADOWNLOAD_ASYNC_JOB_FAILURE | Add > | JOBSUBMITTER_ASYNC_JOB_FAILURE |
| DATADOWNLOAD_ASYNC_JOB_SUCCESS | < Remove | |
| JOBSUBMITTER_ASYNC_JOB_SUCCESS | Add All >> | |
| SAMPLE_TOPIC | << Remove All | |

| | |
|---|---|
| Protocol: | Email |
| SMTP Server: | |
| SMTP Server Port: | 25 |
| SMTP Account: | |
| SMTP Password: | |
| Connection Security: | None |
| Email To: | recipient@example.com |
| Email Cc: | fmeserver@example.com |
| Email From: | fmeserver@example.com |
| Email Subject: | |
| Email Format: | Text |
| Email Template: | |

If `email_subject` and `email_to` keywords are specified in the notification, they will override the default email subject and recipients configured in subscriber respectively.

## Email Notification Content

Email content is configured via a template on a per subscriber basis, and it defines what is included in the email content and how the content is formatted.

For example, a simple email template may look like:

```
This is a notification email from FME Server.
{BodyContent1}
{BodyContent2}
```

If notification shown in the previous example is received, the subscriber will send email with content shown below:

```
This is a notification email from FME Server.
This is a paragraph.
This is another paragraph.
```

## See Also

- "Email Template Language" below
- "Email Template Keywords" on page 68

### Email Template Language

One of the important features of the new email notifier is the template language. It allows users to configure notification email templates in a flexible way.

## Template Block

A template could contain one or more blocks. A block is defined by an XML style tag `<fmeblock></fmeblock>`. If a template does not contain these starting and ending tags, it will be considered as one required block.

If the type attribute of a block is specified as optional, this block will not be included in the notification email content unless all the macros it contains have been fully resolved.

Sample usage:

```
<fmeblock type="optional">
```

```
FME transformation job {id} succeeded.
</fmeblock>
```

This block is only active if the `{id}` macro is resolved.

### Keyword

A keyword is specified as `{<keyword>}`, and it can be referred anywhere in template blocks for multiple times. When a notification email is generated, keywords will be resolved as their actual values. The keywords and their values are supplied by FME Notification Service via notifications.

The typical keywords contained in a JOB_SUCCESS notification may include:

- id: the job ID
- OutputLocation: the location of output dataset
- StatusMessage: the status message from FME Engine response

Sample usage of keywords:

```
<fmeblock type="optional">
Job {id} succeeded. The Engine status message is {Stat-
usMessage}.
</fmeblock>
```

For more information about supported keywords in email templates, see "Email Template Keywords" on next page.

### Function

A function is a special operator starting with @ which performs certain action on given input parameter. The supported functions are:

- @getFileName: returns the file name only of a full file path
- @zip: compress the given directory or file into zip file and returns its full path

Sample usage:

```
<fmeblock type="optional">
Click here to download the result:
```

```
{urlPrefix}{ResultRootDir}/{@getFileName(@zip(Out-
putLocation))}
</fmeblock>
```

Here the `{@getFileName(@zip(OutputLocation))}` is a chain call of two functions.

It first compresses the folder specified by macro `OutputLocation` and then returns the zip file name.

### HTML Content

A template can be in plain text or HTML. The format is specified by the Email Format property of the subscriber. HTML tags can be defined inside or outside template blocks. Keywords can be used in HTML in the same manner as in plain text.

Sample usage:

```
<fmeblock type="optional">
Job {id} succeeded.<br>
This is a URL: <a href=”www.mycompany.com”>My Company</a>
</fmeblock>
```

#### Email Template Keywords

Email templates can be defined when creating email subscriptions. For more information on Email template syntax, see "Email Template Language" on page 66.

### Job Keywords

Every transformation job with notification enabled automatically supports the following keywords:

| Keyword Placeholder Name | Description |
|---|---|
| {id} | The ID assigned to this transformation. |
| {requestKeyword} | The request keyword string from the client that accompanied the request. |

| Keyword Placeholder Name | Description |
|---|---|
| {timeStarted} | The date when the transformation request started to be processed. |
| {timeFinished} | The date when the transformation result was received from an FME Engine. |

### Client Custom Keywords

All notification manager directives specified by a client are also available as keyword placeholder names. In the case of the Data Download Service and Job Submitter services this means any URL parameters prefixed with "nm_" are available.

For example, a Data Download Service URL like the following will also have keyword placeholder `{title}` with value "`Sample`" and keyword placeholder `{desc}` with value "`Hello World`" available to be used within an email template:

```
http:/-
/localhost/fmedatadownload/Samples/austinDownload.fmw?opt_
servicemode=async&opt_requesteremail=test%40email.com&nm_
title=Sample&nm_desc=Hello%20World
```

In addition to the above Web Services, notification manager directives can also be specified via the REST, Java, .NET and C++ APIs.

### Client-Specific Keywords

Every client that invokes a transformation job specifies a request keyword that corresponds to a specific FME Engine configuration. Clients can also specify additional keywords to customize the job results on job success and job failure. By default, all FME Engine response parameters also available as keyword placeholder names. Additional FME Engine response parameters can also be defined by editing the respective FME Engine configuration.

For the Data Download Service and Job Submitter services, the available keywords are:

| Keyword Place-holder Name | Description |
|---|---|
| {statusMessage} | The descriptive string representation of the job status. |
| {statusNumber} | The status number of the FME transformation. |
| {urlPrefix} | The URL prefix to download transformation results. |
| {ResultRootDir} | The root directory of the transformation results. |
| {OutputLocation} | The location of the transformation results. |
| {NumFeaturesOutput} | The number of features written when job transformation is successful. |
| {LogFileName} | The name of the job transformation log file. |
| {email_to} | The email address to send to. |
| {jobsuccess_topic} | The name of the topic notified on job success. |
| {jobfailure_topic} | The name of the topic notified on job failure. |

***FTP Subscriber***

The FTP subscriber receives notifications and uploads files to specified FTP hosts.

**In this section:**

- "Specifying the FTP Notification" below
- "Creating an FTP Subscription" on page 72

**Specifying the FTP Notification**

### Notification Content Keywords

The following reserved keywords can be used in the notification content to modify FTP subscription settings:

- ftp_host
- ftp_port

- ftp_username
- ftp_password
- ftp_encryption
- ftp_transfertype
- ftp_destpath
- ftp_sourcepath
- ftp_zipsource

**Note:** *File paths must be specified using forward slashes (`/`) or escaped backslashes (`\\`).*

**Note:** *If a notification message is sent from the FME Server REST API and it contains an unstructured text message, the value of the `subscriber_content` keyword contains the content of the text message.*

### Example

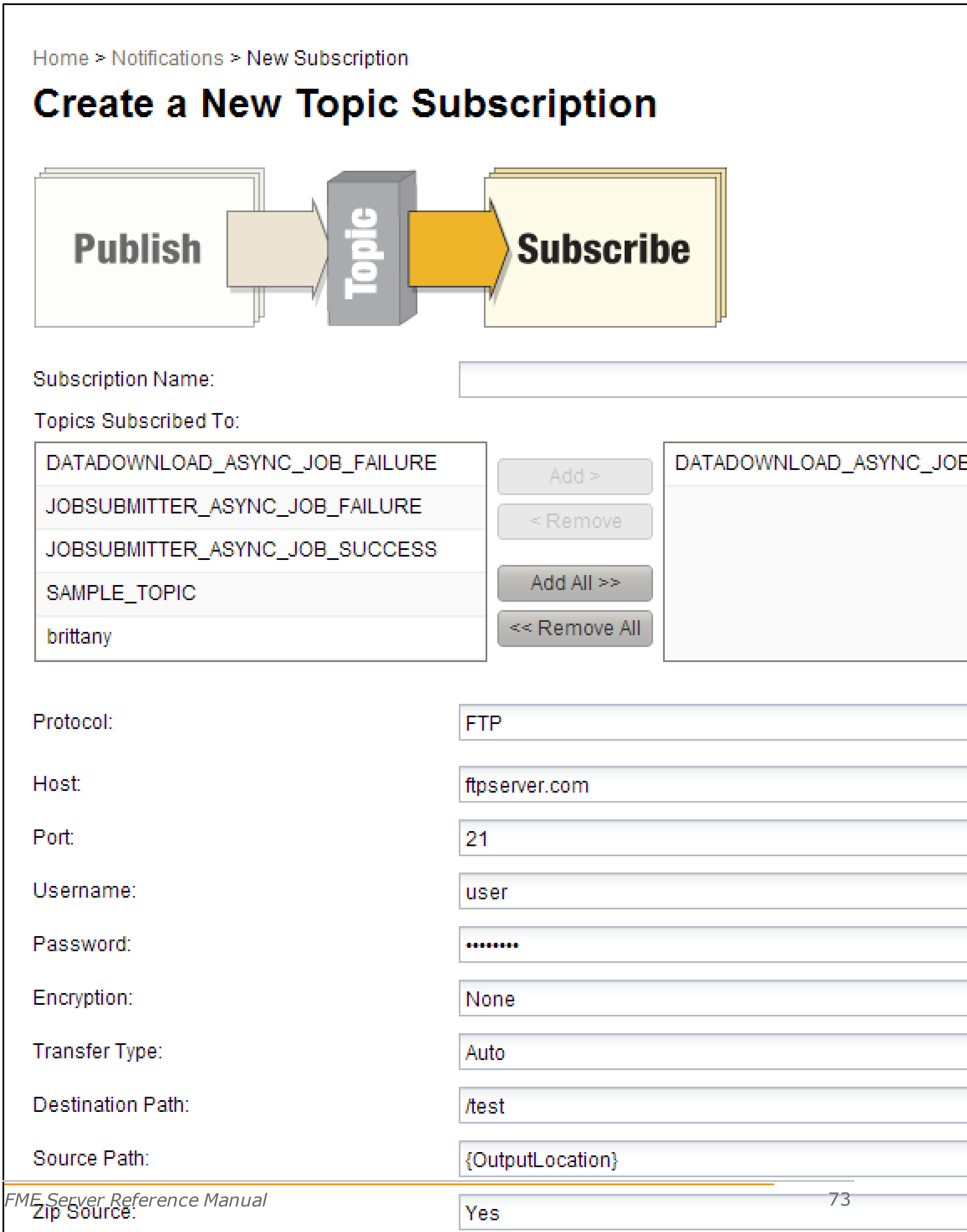
Below is an example of notification content for an FTP subscription:

```
{
  ftp_host : "ftphost",
  ftp_port : "21",
  ftp_username : "ftpuser",
  ftp_password : "ftppassword",
  ftp_encryption : "Plain",
  ftp_transfertype : "Auto",
  ftp_destpath : "/dest/hello/world",
  ftp_sourcepath : "D:/test.jpg",
  ftp_zipsource : "Yes"
}
```

Upon receiving this notification, the FTP subscriber zips and uploads `test.jpg` to the destination path.

## Creating an FTP Subscription

You can use the New Subscription page of the FME Server Web User Interface to create an FTP subscription.



Home > Notifications > New Subscription

Create a New Topic Subscription

Subscription Name:

Topics Subscribed To:

DATADOWNLOAD_ASYNC_JOB_FAILURE
JOBSUBMITTER_ASYNC_JOB_FAILURE
JOBSUBMITTER_ASYNC_JOB_SUCCESS
SAMPLE_TOPIC
brittany

DATADOWNLOAD_ASYNC_JOB

| | |
|---|---|
| Protocol: | FTP |
| Host: | ftpserver.com |
| Port: | 21 |
| Username: | user |
| Password: | •••••••• |
| Encryption: | None |
| Transfer Type: | Auto |
| Destination Path: | /test |
| Source Path: | {OutputLocation} |
| Zip Source: | Yes |

- **Topics Subscribed To** – The topics that this FTP subscription is subscribed to.
- **Host** – The FTP host name.
- **Port** – The FTP port number.
- **Username** – The FTP user name for authentication.
- **Password** – The FTP user password for authentication.
- **Encryption** – The encryption type, depending on the FTP server connection requirements. Options are None, FTPS and FTPES encryption protocols.
- **Transfer Type** – The transfer method. "Auto" means the FTP subscription will guess the file type of the file being transferred. "ASCII" and "Binary" are available options when the file type is known.
- **Destination Path** – The destination path to upload the file. File paths must use forward slashes (`/`) or escaped backslashes (`\\`).
- **Source Path** – The path to the source file to upload. File paths must use forward slashes (`/`) or escaped backslashes (`\\`). Template macros are supported for this field (see below).
- **Zip Source** – "Yes" means the source file should be zipped for delivery. "No" means the source file should not be zipped.

In the example shown above, the source files are all the ones that have been written to `{OutputLocation}`, as specified in the `FILE_DOWNLOAD_SERVICE` section of the FME Engine configuration file. `FILE_DOWNLOAD_SERVICE` is used to configure transformations for the Data Download Service. For more information about the FME Engine configuration file, see "Configuration File Layout" on page 164.

The files are automatically zipped and uploaded to the “test” directory of ftpserver.com.

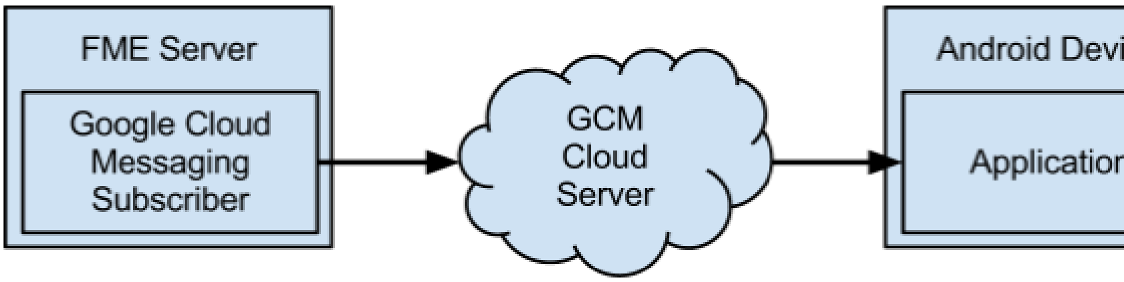
All FME Engine keywords defined for `SUCCESS_RESPONSE` and `FAILURE_RESPONSE` can be used as template macros. For example, the following `SUCCESS_RESPONSE` configuration in the FME Engine configuration file provides

`{OutputLocation}`, `{ResultRootDir}`, `{NumFeaturesOutput}`, and `{Log-FileName}` as available template macros:

```
SUCCESS_RESPONSE 0:Translation Suc-
cessful|OutputLocation=!FME_AUTO_DIR_NAME!|-
ResultRootDir=/fmedatadownload/results|NumFeaturesOutput=!FME_
NUM_FEATURES_OUTPUT!|LogFileName=datadownload/!FME_AUTO_
FILE_NAME_SIMPLE.log! \
```

***Google Cloud Messaging Subscriber***

The Google Cloud Messaging subscriber is a notification subscriber that allows message delivery to a Google Android device, such as a Samsung Galaxy S3. The subscriber pushes messages from FME Server to an Android app installed on the target Android device by means of a cloud service—the Google Cloud Messaging (GCM) service.



**Note:** *The Google Cloud Messaging subscriber is one of two subscribers that can push notifications to a mobile device. For notifying iOS devices, see "Subscribers" on page 56.*

This section explains the steps for setting up the Google Cloud Messaging subscriber:

- "Requirements" on next page
- "Configuring the Subscription" on page 78

**Requirements**

> ***Note:*** *If you do not already have a GCM-enabled Android application, see GCM: Getting Started for an overview on how to write an Android application utilizing Google Cloud Messaging services.*

When configured, a Google Cloud Messaging subscriber can send notifications to your target application installed on an Android device. To accomplish this scenario, the Google Cloud Messaging subscriber uses two requirements from your target application, both of which require developer access to the application:

1. The developer's API key—The authentication mechanism when connecting to GCM cloud servers.
2. The Android device's unique GCM registration ID—Provides the Google Cloud Messaging Service with the target device for sending notifications.

> ***Note:*** *The registration ID is different for each unique Android device/application pair, so it cannot be shared between different applications. When configuring the Google Cloud Messaging subscriber to work with a new application, new registration IDs must be acquired.*

For more information, see Architectural Overview.

**Acquiring the Developer API Key**

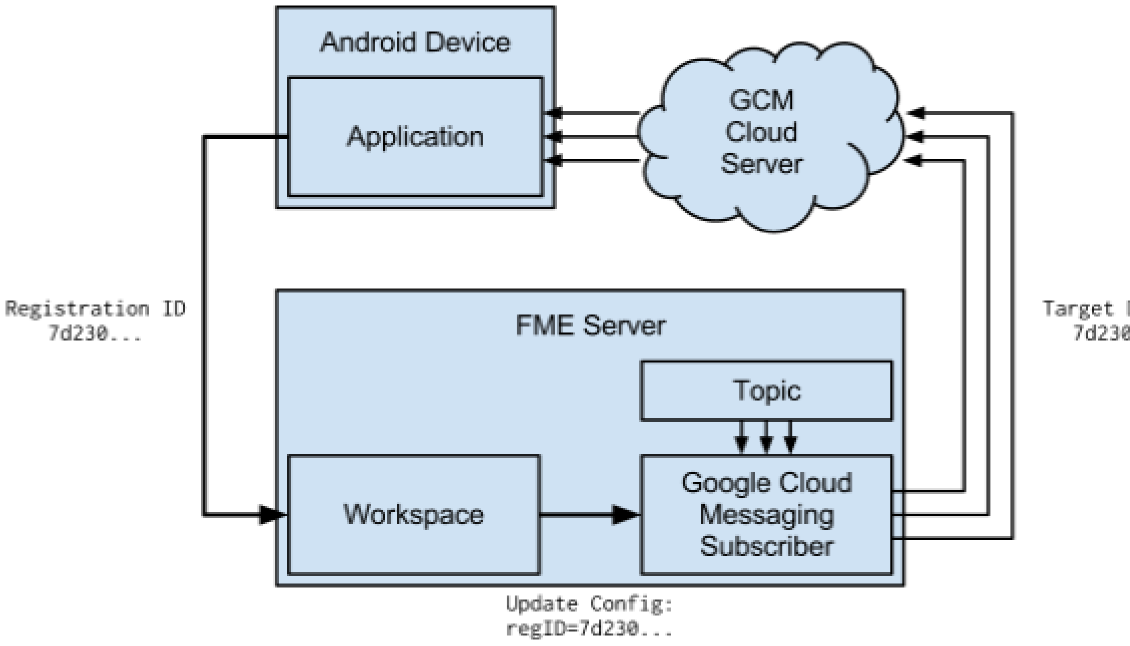
The developer API key authorizes applications (in this case, FME Server) to use the Google Cloud Messaging service to send notifications to Android devices. The API key is associated with the Android application developer, and generating it is a one-time process. To do so, follow the instructions in Obtaining an API Key.

The developer API key will be required when configuring the subscriber.

## Acquiring the GCM Registration ID for the Target Device

Each Android device can register itself to the Google Cloud Messaging service, and when doing so, acquires a registration ID that uniquely identifies the device.

After acquiring the registration ID, the Android application on the device must distribute it to FME Server. Typically, the registration ID is sent to a processing workspace on FME Server that records the token and updates the subscriber. This flow is illustrated below.



An Android application can register to Google Cloud Messaging service by sending the `com.google.android.c2dm.intent.REGISTER` intent, and the registration ID is returned to the application in an intent extended string field `registration_id`. This process is detailed in Writing Android Applications that Use GCM.

**What's Next?**

- "Configuring the Subscription" below

**Configuring the Subscription**

### To Create a Google Cloud Messaging Subscription

1. In the FME Server Web User Interface, select Notifications on the left. Then, select the Subscriptions tab, and click New.
2. Provider a name for the Subscription.
3. Specify the topics to subscribe to. When the Subscription receives messages from these topics, a GCM notification is sent to the target devices.
4. Beside Protocol, specify Google Cloud Messaging.
5. Specify the Google Cloud Messaging fields for the Subscription (see below).
6. Click OK to create the Subscription.

| Google Cloud Messaging Field | Description |
|---|---|
| **API Key** | The developer API key, used for authorizing the subscription to the Google Cloud Messaging service. |
| **Registration IDs** | The list of target Android devices to receive notifications. Use a comma to separate multiple devices. |
| **Content Template** | The notification content template. |

### Overriding Behavior Using Notification Keywords

By default, a Google Cloud Messaging configuration allows for delivery of notifications to a fixed set of Android devices, with its message formatted by the prescribed content template. The Google Cloud Messaging subscriber also allows you to specificy target Android devices and message format in the incoming notification. The notification keywords that enable this overriding behavior are `gcm_token` and `subscriber_content`.

- `gcm_token`—Overrides the 'Registration IDs' field in a Google Cloud Messaging subscription. Use a comma to separate multiple devices.
- `subscriber_content`—Overrides the 'Content Template' field in a Google Cloud Messaging subscription.

## Examples

The following Google Cloud Messaging Subscriber examples are based on this default configuration:

| | |
|---|---|
| **Registration IDs** | `123456,abcdef` |
| **Content Template** | `Message: {msg}` |

| **Notification** | **Target Device(s)** | **Message Sent to Device(s)** |
|---|---|---|
| `{`<br>`   "msg" : "Hello, world"`<br>`}` | `123456`<br>`abcdef` | Message: Hello, world |
| `{`<br>`   "gcm_token" : "987654,fefefe",`<br>`   "msg" : "Hello, device"`<br>`}` | `987654`<br>`fefefe` | Message: Hello, device |
| `{`<br>`   "subscriber_content" : "Alert",`<br>`   "msg" : "Ignored"`<br>`}` | `123456`<br>`abcdef` | Alert |

## Demonstration App

FME Alerts is a demonstration app that showcases the capabilities of FME Server using an Apple Push subscriber. The app is available for free on the Apple App Store.

### *HTTP Push Subscriber*

HTTP push subscriber receives notifications from FME Notification Server and posts HTTP request to the specified URLs in subscribers.

#### Keywords

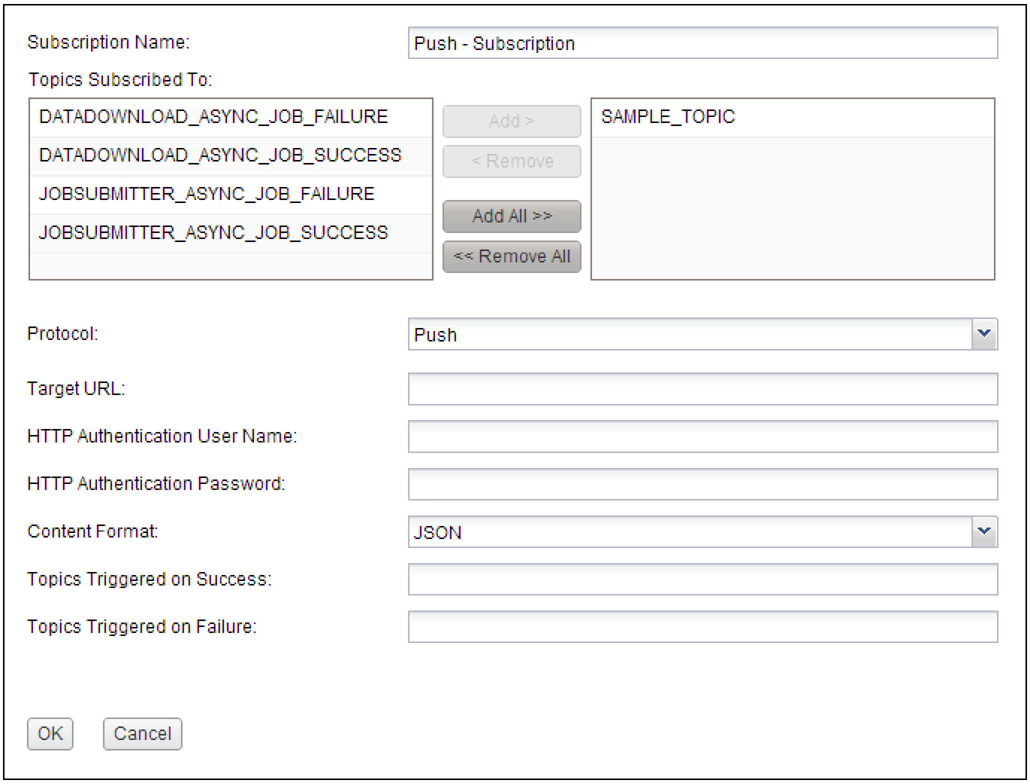
- push_url
- push_content_format
- push_content_encoding
- subscriber_content

#### Subscriber

An HTTP push subscriber is a configuration required to post requests. It contains the following properties:

- Target URL
- HTTP Authentication User
- HTTP Authentication Password
- Content Format: JSON or XML
- Content Encoding: UTF-8

The following is the dialog for creating a new HTTP Push subscription from the Web User Interface.



| | |
|---|---|
| Subscription Name: | Push - Subscription |
| Topics Subscribed To: | |

Available topics: DATADOWNLOAD_ASYNC_JOB_FAILURE, DATADOWNLOAD_ASYNC_JOB_SUCCESS, JOBSUBMITTER_ASYNC_JOB_FAILURE, JOBSUBMITTER_ASYNC_JOB_SUCCESS

Subscribed topics: SAMPLE_TOPIC

| | |
|---|---|
| Protocol: | Push |
| Target URL: | |
| HTTP Authentication User Name: | |
| HTTP Authentication Password: | |
| Content Format: | JSON |
| Topics Triggered on Success: | |
| Topics Triggered on Failure: | |

## Push Notification Content

An FME Server notification stores the message content in keyword/value pairs. If push subscriber receives a notification consisting of only one keyword/value pair, it will send the value as the POST request body. Otherwise, it will format all the keywords and values into JSON or XML based on subscriber.

For example, the notification below will be posted as value only:

```
subscriber_content: "This is my notification message"
```

The POST request body will be "This is my notification message".

Another notification consisting of multiple keyword/value pairs will be posted as JSON or XML:

```
{
        "foo": "value 1",
        "bar": "value 2"
    }
```

Or:

```
    <?xml version="1.0" encoding="UTF-8"?>
    <content>
        <foo>value 1</foo>
        <bar>value 2</bar>
    </content>
```

***JMS Subscriber***

For information about configuring FME Server to communicate using Java Message System (JMS), see "Working with JMS" on page 92.

***Logger Subscriber***

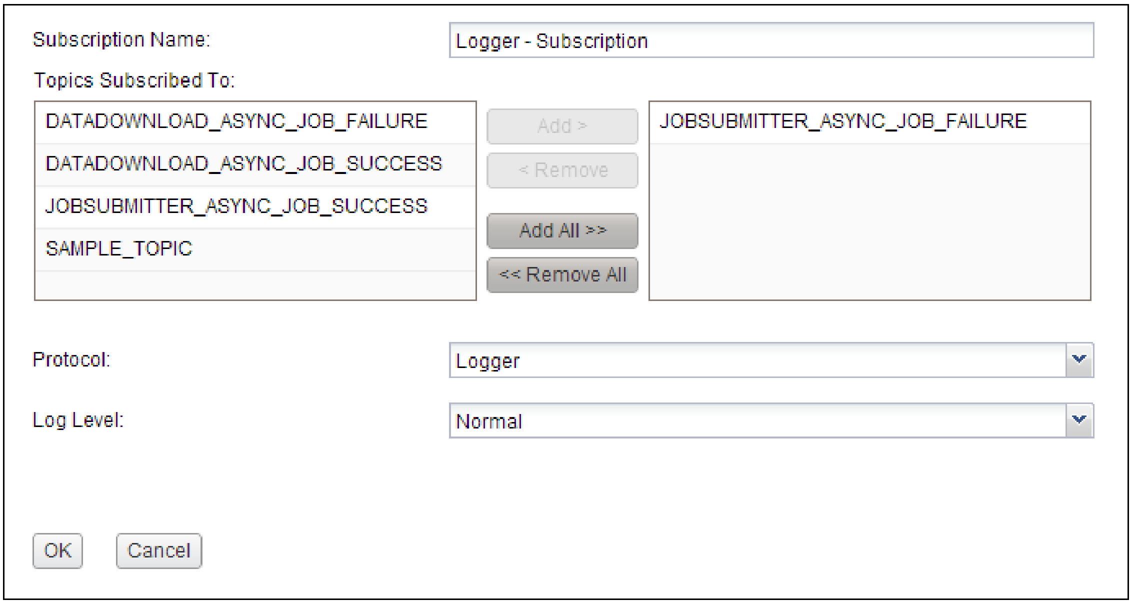
Logger subscriber will log out all notifications of the topics that it is subscribed to.

**Keywords**

None at this time.

**Subscriber**

The level of logging can be controlled. Currently the options are low, normal and high.



| | |
|---|---|
| Subscription Name: | Logger - Subscription |
| Topics Subscribed To: | |
| Available topics | DATADOWNLOAD_ASYNC_JOB_FAILURE, DATADOWNLOAD_ASYNC_JOB_SUCCESS, JOBSUBMITTER_ASYNC_JOB_SUCCESS, SAMPLE_TOPIC |
| Subscribed topics | JOBSUBMITTER_ASYNC_JOB_FAILURE |
| Protocol: | Logger |
| Log Level: | Normal |

## Logger Notification Content

Depending on the log level, the logger subscriber will automatically log out all or parts of the notification message.

The log messages are stored under: `<FMEServerHome>\Server\Logs`

## Publishers

### *Email Publisher*

The Email publisher allows you to configure email addresses to receive messages from SMTP email clients, and publish the email data to FME Server notification topics.

**In this section:**

- "Email Publisher Settings" on next page
- "Example Email Publication" on page 86

### Email Publisher Settings

Most settings are stored in `<FME-Sever>/Utilities/smtprelay/james/apps/james/SAR-INF/config.xml`. For a complete reference on configurations, see http://james.apache.org/.

The following is the configuration section most relevant to the Email publisher. For a description of the parameter tags, see the "Example Email Publication" on page 86

```
<mailet match="All" class="FMENotificationMailet">

  <!-- Relayer server path -->
  <relay-
  ServerPath>D:/Apps/FMEServer/Server</relayServerPath>

  <!-- Email publisher name -->
  <emailPublisherName>email</emailPublisherName>

  <!-- Directory where email attachments should be stored.
  If "system" specified, system temp directory is used. -->

  <emailAttachmentDir>system</emailAttachmentDir>
  <!-- This section defines the keywords used by the noti-
  fication
  messages sent from this mailet -->
  <emailSender>email_relay_from</emailSender>
  <emailRecipients>email_relay_to</emailRecipients>
  <emailSubject>email_relay_subject</emailSubject>
  <emailContent>email_relay_content</emailContent>
  <emailSentDate>email_relay_sent</emailSentDate>
  <emailReceivedDate>email_relay_received</ema-
  ilReceivedDate>
  <emailAttachment>email_relay_attachment</emailAttachment>
</mailet>
```

### Attachment Directory

Specifies the location where the attachment files of incoming emails are stored. By default, `system` is specified, meaning that the system temp directory is used. This value can be changed by modifying the `<emai-lAttachmentDir>` XML tag.

For example:

```
<emai-
lAttachmentDir>D:/temp/myattachments</emailAttachmentDir>
```

## Deploying with a public DNS

The public DNS name for email addresses (host name after @) is for display only and is used by the Protocol Service to show expected email addresses. It does not affect the functionality of the James Server.

If you are deploying a production server to send and receive emails over the Internet, you must change the server name from `localhost` to a public DNS name by changing the `<servername>` XML tag value.

For example, if the public DNS is mycompany.com then change:

```
<servername>localhost</servername>
```

to

```
<servername>mycompany.com</servername>
```

This example allows FME Server to receive emails to <user-name>@mycompany.com

## Configuring secure SMTP and POP3 servers with SSL/TLS

Secure SMTP and POP3 operate through ports 465 and 995, and are disabled by default. To enable them, follow these steps:

1. Obtain an SSL certificate from a certification authority, and deploy it to the following directory:

```
<FMESever>/Utilities/smtprelay/james/apps/james/conf
```

2. Enable the SSL socket factory by un-commenting the following section in the `config.xml` file, and reconfiguring as necessary:

```
<factory name="ssl" class="org.apache-
.av-
alon.cornerstone.blocks.sockets.TLSServerSocketFactory">
```

```
          <ssl-factory>
            <keystore>
              <file>conf/keystore</file> <!-- replace this
              with the relative path of your certificate key-
              store -->
              <password>secret</password> <!-- replace this
              with your keystore password -->
              <key-password>keysecret</key-password> <!--
              replace this with your priviate key password --
              >
              <type>JKS</type>
              <protocol>TLS</protocol>
              <algorithm>SunX509</algorithm>
              <authenticate-client>false</authenticate-
              client>
            </keystore>
          </ssl-factory>
        </factory>
```

3. Change the `<pop3server-tls>` and `<smtpserver-tls>` tags to set attribute enabled="true".

**POP3 and SMTP ports**

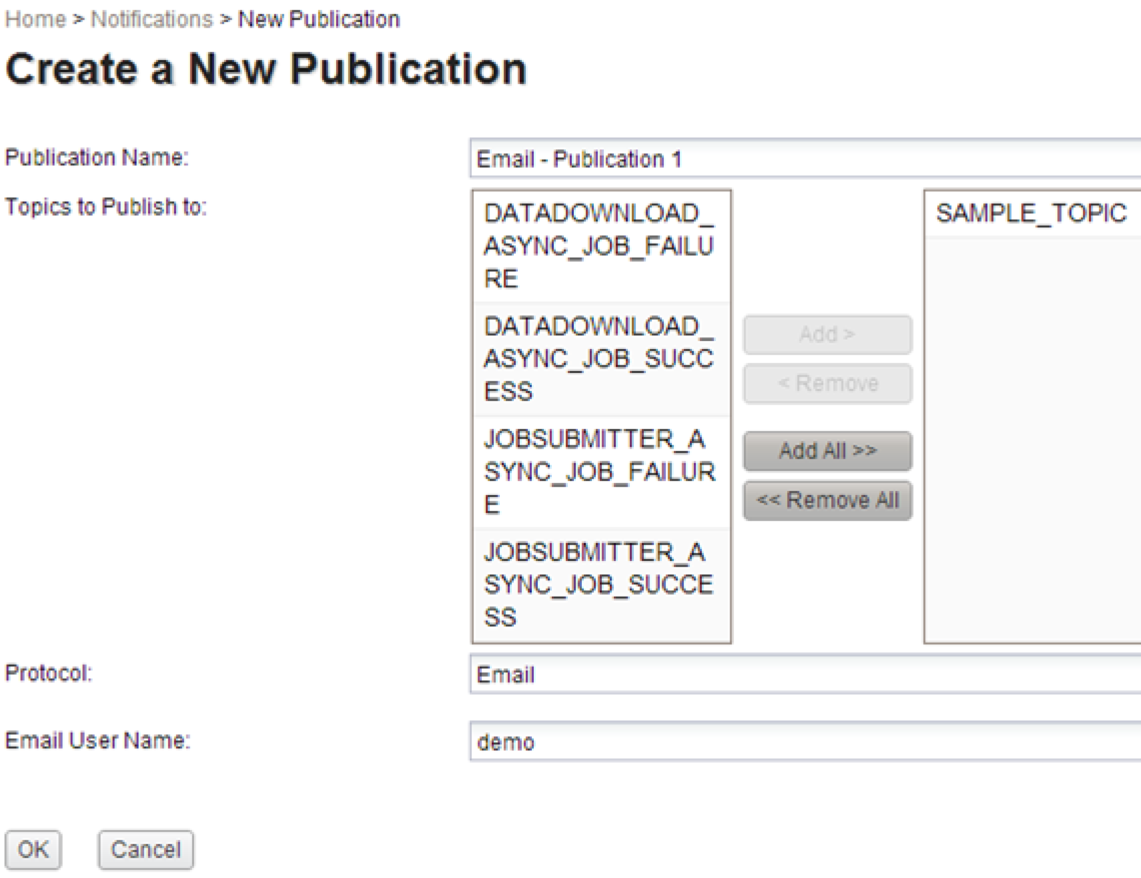
- **Non-secure**: POP3: 110, SMTP: 25
- **Secure with SSL**: POP3: 995, SMTP: 465

The ports listed above are standard for POP3 and SMTP, and cannot be changed if your server wants to communicate with other email servers such as Gmail. We recommend turning off the non-secure SMTP and POP3 ports once you have servers working over SSL. For detailed instructions, see http://wiki.apache.org/james/UsingSSL.

**Example Email Publication**

The following example Email publication receives email messages at demo@myserver, where myserver is the host name, and publishes notification messages to topic SAMPLE_TOPIC.

***Note:*** *Email addresses cannot be duplicated across Email publications. There must be a unique email address for each email publication. Email addresses are case-insensitive.*



Home > Notifications > New Publication

Create a New Publication

| | |
|---|---|
| Publication Name: | Email - Publication 1 |
| Topics to Publish to: | DATADOWNLOAD_ASYNC_JOB_FAILURE, DATADOWNLOAD_ASYNC_JOB_SUCCESS, JOBSUBMITTER_ASYNC_JOB_FAILURE, JOBSUBMITTER_ASYNC_JOB_SUCCESS; selected: SAMPLE_TOPIC |
| Protocol: | Email |
| Email User Name: | demo |

### Notification Content

The notification content that is generated by the email publisher includes the following keywords:

- **fns_type** – The value is “email_publisher” to identify the message is being relayed via the Email publisher.
- **email_publisher_from** – The email address of the email client that sent the mail.
- **email_publisher_to** – The email address that received the email (The same as what is configured in the Email publication).
- **email_publisher_subject** – The email subject
- **email_publisher_content{n}** – The email message content. Multiple "contents" appear in conjunction with multiple "content types," and have a suffix starting at {0}, {1}, {2}, and so on.
- **email_publisher_content_type{n}** – The email message content type. Multiple content types have a suffix starting at {0}, {1}, {2}, and so on. The most common content types are `text/plain` and `text/html`.
- **email_publisher_sent** – The date the email was sent
- **email_publisher_received** – The date the email was received
- **email_publisher_attachment{n}** – The attachment directory. Multiple attachments have a suffix starting at {0}, {1}, {2}, and so on. Forward slashes are supported. Backslashes must be escaped as double blackslashes.

The following is a sample JSON notification message published by the Email publisher upon receiving an email to address 'demo@somehost.com':

```
{
  "fns_type": "email_publisher",
  "email_publisher_to": "demo@somehost.com",
  "email_publisher_subject": "MIME message from sender",
  "email_publisher_content{0}": "Testing Email",
  "email_publisher_content_type{0}": "text/plain",
  "email_publisher_from": "sender@somehost.com",
  "email_publisher_received": "Thu May 17 11:15:46 PDT
  2012",
  "email_publisher_sent": "Thu May 17 11:15:46 PDT 2012",
  "email_publisher_attachment{0}":
  "C:\\Temp\\demo246129673106713_canada.xsd"
}
```

**Note:** *The keywords of the above JSON message can be customized by changing the config.xml file. No code changes are required. For more information, see "Email Publisher Settings" on page 84.*

***JMS Publisher***

For information about configuring FME Server to communicate using Java Message System (JMS), see "Working with JMS" on page 92.

***UDP Publisher***

UDP is a simple datagram protocol that sends and receives data through specific UDP ports. The data delivery is not guaranteed. Many sensor devices and mobile apps use UDP to publish live data.

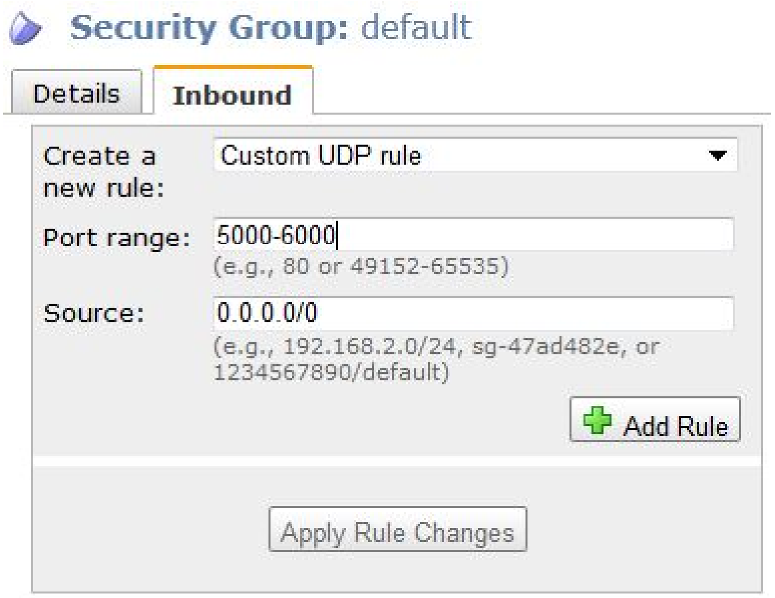
The FME Server Notification Service UDP publisher relays UDP messages and publishes them to notification topics. Using the UDP publisher, you can enable multiple UDP ports to listen for incoming data packets. Each UDP port can be uniquely configured to publish incoming UDP messages to one or more notification topics.

**In this section:**

**Specifying UDP Ports**

UDP ports cannot be duplicated across UDP publications. There must be a unique UDP port for each UDP publication.

When specific ports are desired, the port should be made available and not blocked by firewall settings. In the following example, Amazon Web Services firewall security settings are configured for an instance to reserve ports 5000 - 6000 (incoming) for the UDP Publisher:


Security Group: default

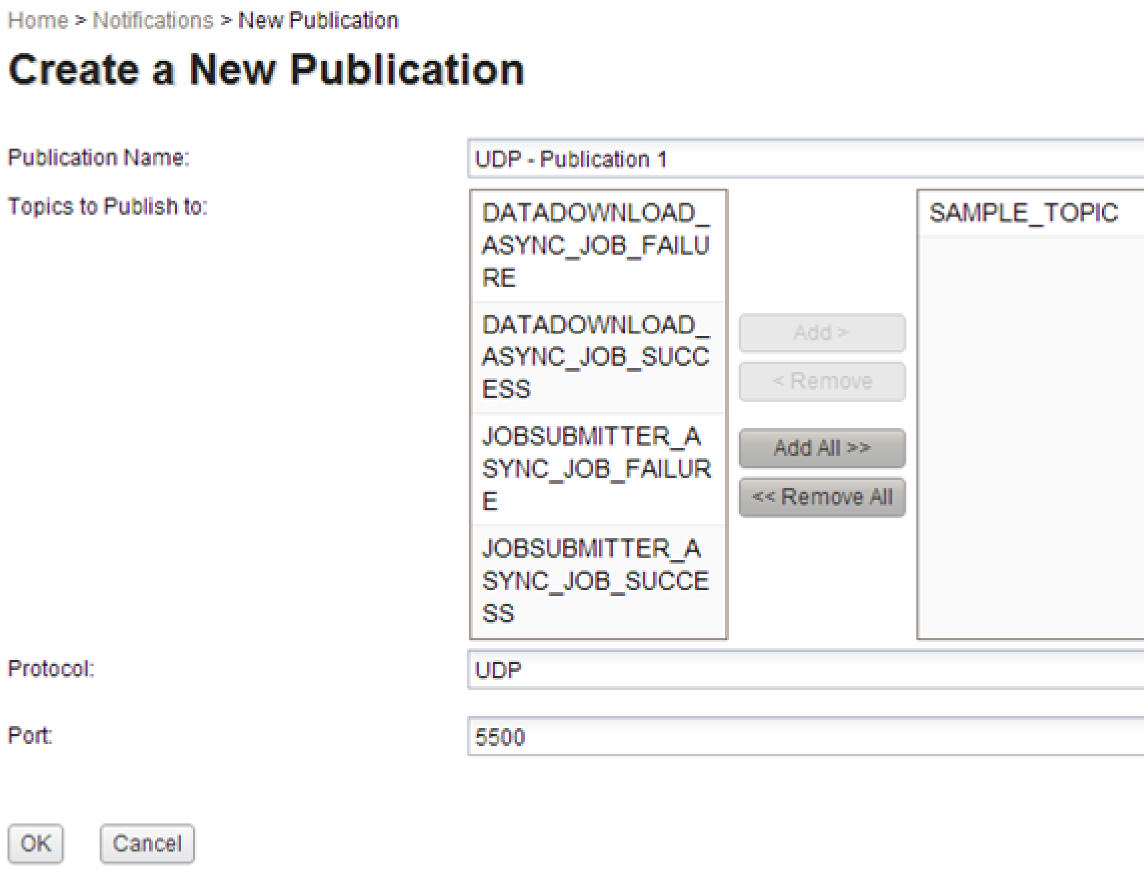
Create a new rule: Custom UDP rule

Port range: 5000-6000
(e.g., 80 or 49152-65535)

Source: 0.0.0.0/0
(e.g., 192.168.2.0/24, sg-47ad482e, or 1234567890/default)

**Example UDP Publication**

The following example UDP publication receives UDP messages on port 5500 and publishes notification messages to topic SAMPLE_TOPIC.


Home > Notifications > New Publication

Create a New Publication

| | |
|---|---|
| Publication Name: | UDP - Publication 1 |
| Topics to Publish to: | DATADOWNLOAD_ASYNC_JOB_FAILURE, DATADOWNLOAD_ASYNC_JOB_SUCCESS, JOBSUBMITTER_ASYNC_JOB_FAILURE, JOBSUBMITTER_ASYNC_JOB_SUCCESS; selected: SAMPLE_TOPIC |
| Protocol: | UDP |
| Port: | 5500 |

## UDP Publisher Notification Content

The notification content sent by the UDP Publisher includes the following keywords:

- **fns_type** – This value is “udp_publisher” to indicate the message is relayed via the UDP publisher.
- **udp_publisher_content** – The content of the UDP message.

The following is an example of UDP Publisher notification content:

```
{
   fns_type : “udp_publisher”,
        udp_publisher_content : "The UDP message to relay"
    }
```

## Working with JMS

Beginning with FME Server 2013, Notification Services can use the Java Message System (JMS) to communicate with JMS-compliant message brokers, such as IBM WebSphere MQ, both as a publisher and subscriber.

FME Server supports communication to and from all message brokers. For specific message brokers, additional configuration may be required. This manual provides instructions for configuring two commonly used message brokers: Apache ActiveMQ and IBM WebSphere MQ 7.

The steps for setting up FME Server to communicate using JMS are:

1. "Installing the JMS Libraries" below
2. "Configuring the JMS Provider" on the facing page
3. "Configuring FME Server as a JMS Client" on page 97
4. "Testing and Troubleshooting" on page 102

### *Installing the JMS Libraries*

An installation of FME Server includes the libraries that are required for using JMS messaging with Apache ActiveMQ 5.6.0. For all other message brokers, vendor-specific JMS libraries must be acquired and installed alongside FME Server.

Typically, vendor-specific JMS libraries are either available in the message broker's installation, or available in a separate message broker client. In the case of IBM WebSphere MQ 7, for example, a separate client installation is available.

The table below provides links to appropriate software. For additional information, consult your vendor's documentation.

**To install the JMS Libraries**

1. Download and install the software containing the vendor-specific JMS implementation library to the machine with the installation of FME

Server.

2. Locate the required libraries. (See the table below, under "Libraries.")
3. Note the directory containing the required libraries. When "Configuring FME Server as a JMS Client" on page 97, this directory is added to the java classpath.

| Message Broker | Libraries | Sample Path (Windows) |
|---|---|---|
| **Apache ActiveMQ** | none (Apache ActiveMQ 5.6.0 libraries bundled with FME Server) | -- |
| **IBM WebSphere 7** | com.ibm.mqjms.jar<br>fscontext.jar | C:\Program Files (x86)\IBM\WebSphere MQ\java\lib |
| **HornetQ** | hornetq-jms.jar<br>hornetq-jms-client.jar<br>jnp-client.jar | C:\hornetq-x.y.z\lib |

### What's Next?

Depending on your message broker, additional configuration may be required to enable JMS. See "Configuring the JMS Provider" below

#### *Configuring the JMS Provider*

Depending on your message broker, additional configuration may be required to enable JMS.

### Configuring JMS for Apache ActiveMQ

JMS support for Apache ActiveMQ is pre-configured in FME Server; no additional configuration is required. For a complete reference, see JNDI Support. (JMS uses the Java Naming and Directory Interface (JNDI) to dynamically request named objects from the message broker.)

When using Apache ActiveMQ with JMS, keep in mind the following:

- FME Server connects to ActiveMQ via the transport connector specified in the message broker's XML configuration file. By default, the URL is `tcp://hostname:61616`.
- FME Server, as a JMS client, connects to ActiveMQ via a connection factory named object, and references ActiveMQ topics or queues via a destination named object. The named objects are typically specified via a JNDI configuration file, `jndi.properties`. For more information, see JNDI Support.
- In the simplest configuration, a `jndi.properties` file is not required. By default, the connection factory name is `ConnectionFactory`. ActiveMQ also provides a means to dynamically connect to queues and topics via the special destination names `dynamicQueues/<queue>` and `dynamicTopics/<topic>`.

## Configuring JMS for IBM WebSphere 7

For IBM WebSphere 7, a set of JMS Administered Objects must be created in order to interface with JMS. The easiest way to accomplish this task is by using MQ Explorer.

***Note:*** *The following steps assume that a message queue is deployed and running on a server, and that a server-connection channel is created for client connections. In our examples, the queue manager name is `qm`, the queue manager server host name is `server` and the server-connection channel name is `c1`.*

1. Install and launch MQ Explorer to the machine with the installation of FME Server.
2. Add a connection to the desired queue manager.
   a. In the navigation on the left, right-click Queue Managers and select Add Remote Queue Manager.
   b. In the wizard, specify the following parameters:

| | |
|---|---|
| **Queue manager name** | `qm` (for example) |
| **Connection method** | Connect directly |
| **Host name or IP address** | `server` (for example) |
| **Port number** | 1414 |
| **Server-connection channel** | SYSTEM.ADMIN.SVRCONN (for example) |

c. Ensure that both the specified server-connection channel (default: `SYSTEM.ADMIN.SVRCONN`) and a listener for the specified TCP port (default: `LISTENER.TCP`) are both enabled.

d. Select Finish to complete the wizard. A new connection to a remote queue manager should appear.

3. Create a JMS initial context.

   a. In the navigation on the left, right-click JMS Administered Objects and select Add Initial Context.

   b. In the wizard, specify the following parameters:

| | |
|---|---|
| **JNDI namespace location** | File system |
| **Directory** | `c:\jndi` (for example) |
| **Provider URL** | `file:/C:/jndi/` (for example) |

   c. Select 'Finish' to complete the wizard. A new initial context should appear.

4. Create a JMS connection factory.

   a. Right-click Connection Factories under the newly-created initial context and select New / Connection Factory.

   b. In the wizard, specify the following parameters:

| | |
|---|---|
| **Name** | `cf1` (for example) |
| **Messaging provider** | WebSphere MQ |

| | |
|---|---|
| **Type** | ConnectionFactory |
| **Transport** | MQ Client |
| **Connection/Base queue manager** | `qm` (for example) |
| **Connection/Connection list** | `server(1414)` (for example) |
| **Channel/Channel** | `c1` (for example) |

c. Ensure that the queue manager and connection parameters match those of the queue manager specified above.

d. Select Finish to complete the wizard. The new connection factory appears.

5. Create a JMS destination for each desired queue or topic.

   a. Select Queues or Topics under the queue manager to bring up the current list of queues or topics.

   b. Right-click the desired queue or topic and select Create JMS Queue or Create JMS Topic.

   c. In the wizard, provide the JMS object with a name `(q1` or `t1`, for example).

   d. Select Finish to complete the wizard. The new destination appears underDestinations for the JMS initial context.

6. Configure security as necessary, using the `setmqaut` tool. For example, to enable access to a queue manager, execute the following command:

```
setmqaut -m qm -t qmgr -g Users +inq +connect
```

   To enable consumer access to a queue or topic to a particular user group Users, execute the following command:

```
setmqaut -m qm -t queue -n q1 -g Users +get +inq
+browse
setmqaut -m qm -t topic -n t1 -g Users +sub
```

To enable producer access to a queue or topic to a particular user group Users, execute the following command:

```
setmqaut -m qm -t queue -n q1 -g Users +put
setmqaut -m qm -t topic -n q1 -g Users +pub
```

### What's Next?

To make FME Server aware of the newly-installed JMS libraries, proceed to "Configuring FME Server as a JMS Client" below.

### *Configuring FME Server as a JMS Client*

FME Server’s notification system comes equipped with both a JMS publisher and a JMS subscriber. The JMS publisher consumes messages from a message broker and publishes them to an FME Server topic. The JMS subscriber does the opposite; it subscribes to an FME Server topic, and provides messages to a message broker.

> ***Note:*** *Sample configurations for different message brokers are provided following these instructions.*

### Provide Access to the JMS Libraries

To ensure the publisher and subscriber both have access to the required JMS libraries, you must add access to the libraries in the java classpaths (see "Installing the JMS Libraries" on page 92).

> ***Note:*** *Skip this section if using JMS with Apache ActiveMQ. the JMS libraries for Apache ActiveMQ are bundled with FME Server, and so java classpaths need not be modified.*

1. In a text editor, open `processMonitorConfig.txt`, located in the `Server` subdirectory of the FME Server installation.
2. Scroll to the section on launching the JMS publisher, with the heading `Start FME Server Publisher Plugin (jms)`.
3. Identify the classpath variable, FMESERVER_CLASSPATH.

4. Append the path containing the vendor-specific JMS library. Ensure that `/*` appears at the end of each path, so all libraries within the path are picked up. For example:

   ```
   ... -FMESERVER_CLASSPATH "...;C:/Program Files (x86)
   /IBM/WebSphere MQ/java/lib/*" ...
   ```

5. Repeats steps 2 to 4 for the JMS subscriber, with the heading `Start FME Server Subscriber Plugin (jms)`.
6. Save the file, and restart FME Server.

***Note:*** *If the classpath variable is incorrectly specified, the following messages (or their variants) may appear in the FME Server Process Monitor log file:*

*"Exception in thread "main" java.lang.-NoClassDefFoundError: javax/jms/JMSException"*

*"Failed to initialize JMS object named "xxx", because of missing class "yyy". Please ensure that the JMS client libraries are available for your JMS provider."*

*For more information, see "Testing and Troubleshooting" on page 102.*

After restarting FME Server, proceed with adding the JMS publisher and/or subscriber.

## Creating a JMS Publisher for FME Server

1. In the FME Server Web User Interface, select Notifications on the left, then select the Publications tab, and click New.
2. Provide a name for the Publication.
3. Specify the topics to publish to. When receiving JMS messages from the message broker, these topics will be notified.
4. Specify 'jms' for the protocol.

5. Specify the JMS-specific fields for the publisher (see the tables below).
6. Click OK.

**Creating a JMS subscriber for FME Server**

1. In the FME Server Web User Interface, select Notifications on the left, then select the Subscriptions tab, and click New.
2. Provide a name for the Subscription.
3. Specify the topics to subscribe to. When receiving messages from these topics, a JMS message is sent to the message broker.
4. Specify ‘jms’ for the protocol.
5. Specify the JMS-specific fields for the subscriber (see the tables below).
6. Click OK.

**JMS-specific Fields**

| | |
|---|---|
| **Provider Type or Context** | The message broker to connect to, or the initial context factory if the broker is not listed. |
| **Provider URL** | The URL of the directory containing connection details for the JMS provider. |
| **Additional Provider Properties** | Any additional provider-specific properties, in the form `key=value`. |
| **Connection Factory** | The name of the connection factory object. |
| **Username** | An authenticating username. |
| **Password** | An authenticating password. |
| **Destination (s)** | The name of the destination object(s). These correspond to queues and/or topics. Multiple destinations can be specified, separated with a comma. |

**Sample Configurations**

### Apache ActiveMQ Sample Configuration

| | |
|---|---|
| **Java Classpath** | No change required. (Apache ActiveMQ 5.6.0 libraries are bundled with FME Server.) |
| **Provider Type or Context** | `ACTIVEMQ` |
| **Provider URL** | `tcp://server:61616` |
| **Additional Provider Properties** | -- |
| **Connection Factory** | `ConnectionFactory` |
| **Username** | -- |
| **Password** | -- |
| **Destination(s)** | `dynamicQueues/MyQueue` |

### IBM WebSphere 7 Sample Configuration

| | |
|---|---|
| **Java Classpath** | `... -FMESERVER_CLASSPATH "...;C:/Program Files (x86) /IBM/WebSphere MQ/java/lib/*" ...` |
| **Provider Type or Context** | `WEBSPHERE70` |
| **Provider URL** | `file:/C:/jndi/` |

| | |
|---|---|
| **Additional Provider Properties** | -- |
| **Connection Factory** | `cf1` |
| **Username** | `admin` |
| **Password** | `admin` |
| **Destination(s)** | `q1,t1` |

**Notification Content**

The notification content sent by the JMS Publisher includes the following keywords:

- **fns_type** – This value is “jms_publisher”.
- **jms_publisher_type**– This value can be one of the following:
  - **text** - TextMessage.
  - **object** - ObjectMessage.
  - **map** - MapMessage
- **jms_publisher_content** - This value can be one of the following:
  - **<*content*>** - TextMessage content.
  - <***object*.toString()**> - ObjectMessage content

<***map_key1> = <map_value1***>, <***map_key2> = <map_value2***>, and so on - MapMessage mapping.

**What's Next?**

It is important to test the system after configuration. For more information, see "Testing and Troubleshooting" below.

***Testing and Troubleshooting***

It is important to test the system after configuration, to ensure that systems connect and communicate smoothly. The log files provide information about the current status of FME Server.

- **Process Monitor log**—Indicates whether the publisher or subscriber has been started correctly. This log file is found at `Logs/processMonitor_xxx.log` of your FME Server installation path.
- **JMS publisher log**—Contain specific information about reading JMS configurations, connecting to JMS providers, and receiving JMS messages from destinations. This log file is found at `Logs/relayers/publishers/jms/jms_xxx.log` of your FME Server installation path.
- **JMS subscriber log**—Contains information similar to the JMS publisher log, as well as information about sending JMS messages to destinations. This log file can be found at `Logs/notifier/subscribers/jms/jms_xxx.log` of your FME Server installation path.

## OGC Services

### OGC Web Feature Service

The Open Geospatial Consortium (OGC) Web Feature Service (WFS) supplies WFS functionality.

Requests can be made using the Standard Web Feature Service (WFS) (http://en.wikipedia.org/wiki/Web_Feature_Service) URL request.

**Base URL Example:**

```
http://<host>/fmeogc/<repository>/<workspace>.fmw
```

**GetCapabilities URL Example:**

```
http://<host>/fmeogc/<repository>/<workspace>.fmw?SERV-
ICE=WFS&REQUEST=GetCapabilities&VERSION=1.1.0
```

The resulting response is a WFS formatted XML file containing the requested information or features.

***Workspace Published Parameters***

This service works with any workspace that writes to a GML dataset and has the following published parameters:

- **bboxEast**: Bounding Box East
- **bboxWest**: Bounding Box West
- **bboxNorth**: Bounding Box North
- **bboxSouth**: Bounding Box South

  ***Note:*** *You must specify the coordinates of the bounding boxes in the coordinate system given by destCoordSystem.*

- **destCoordSystem**: Output coordinate system
- **FEATURE_TYPES**: White space delimited feature type names
- **REQUEST_VERSION**: The workspace should contain two destination datasets using GML2 and GML writers. If the value of this parameter is 1.1.0, only the GML (version 3) writer produces output. If the value of this parameter is 1.0.0, only the GML2 writer produces output. No other values are supported.

  ***Note:*** *If a Coordinate System or Bounding Box is not submitted with the WFS query, default values are used. You can configure these default values within the Web Interface.*

***Web Application Properties***

The following table lists the properties file path and web application properties of the OGC Web Feature service.

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. Can be specified as a network name or an IP.<br><br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br><br>Default value = 7071 (the default Transformation Manager listening port) |
| CONNECTION_POOL_EXPIRY | The length of time in seconds in which an FME Server connection remains idle before expiring from the connection pool.<br><br>If a value of less than or equal to zero is specified, there is no connection pooling.<br><br>Default value = 300 |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a messageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message.<br><br>Default value = true |
| INCLUDE_THREAD_NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message.<br><br>Default value = false |
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used.<br><br>Default value = true |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| ECHO_LOG_TO_CONSOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one.<br><br>Default value = 3000 |
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| PLUGINS_DIR | The directory that the OGC Servlet will scan for plugins. This is the directory where any new jar files will be placed to install new services for the OGC servlet. |
| OSGi_CACHE_DIR | The directory where the OGC Servlet will keep the OSGi information required to maintain the plugin architecture. |
| SCHEMA_PATH | This is the path to the XML schema for the WFS CityGML support. |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| WFS_CAPABIL-ITIESORDER | Defines the order in which the capabilities will be displayed in the GetCapabilities response for the WFS service.<br><br>Use character ; to separate the capabilities. |
| WMS_CAPABIL-ITIESORDER | Defines the order in which the capabilities will be displayed in the GetCapabilities response for the WMS service.<br><br>Use character ; to separate the capabilities. |
| WFS_VERSIONS | Defines the supported WFS versions. |
| WMS_VERSIONS | Defines the supported WMS versions. |
| SECURITY_MODE | The security mode of this web service. |
| DEFAULT_USER_ID | The default user ID to log in FME Server. |
| DEFAULT_PASSWORD | The default password for default user ID. |
| WFS_CLIENT_ID | The client ID assigned to WFS service. |
| WMS_CLIENT_ID | The client ID assigned to WMS service. |
| WFS_ENGINE_SUB-SECTION | The subsection name used for workspace trans-formation (WFS service). |
| WMS_ENGINE_SUB-SECTION | The subsection name used for workspace trans-formation (WMS service). |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| FAILOVER_SERVER_ NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

**Example 1**

Sample GetCapabilities request

http:/-
/loca-
lhost/fmeogc/Samples/austinWFS.fmw?SERVICE=WFS&REQUEST=GetCapabilities&VERSION=1.

### Example 2

Sample DescribeFeatureType request

http:/-
/loca-
lhost/fmeogc/Samples/austinWFS.fmw?SERVICE=WFS&VERSION=1.1.0&REQUEST=DescribeFeatu

### Example 3

Sample GetFeature request

http:/-
/loca-
lhost/fmeogc/Samples/austinWFS.fmw?SERVICE=WFS&VERSION=1.1.0&REQUEST=GetFeature&TY

## OGC Web Mapping Service

The OGC Web Mapping Service (WMS) supplies WMS functionality.

This service is requested using the Standard Web Mapping Service (WMS) (http://en.wikipedia.org/wiki/Web_Map_Service) URL request. This service supports 1.1.1 and 1.3.0 WMS versions.

### Base URL Example:

```
http://<host>/fmeogc/<repository>/<workspace>.fmw
```

### GetCapabilities URL Example:

```
http://<host>/fmeogc/<repository>/<workspace>.fmw?S-
ERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.1.1
```

The result is a PNG, JPEG, GIF, or SVG file containing the requested features.

#### *Requirements*

This service requires a workspace that can write to PNG, JPEG, GIF, and/or SVG datasets, and that has the following published parameters:

- **bboxEast**: Bounding Box East
- **bboxWest**: Bounding Box West
- **bboxNorth**: Bounding Box North

- **bboxSouth**: Bounding Box South

  ***Note:*** *You must specify the bounding box coordinates in Latitude/Longitude.*

- **destCoordSystem**: The coordinate system in which the final output is rendered.
- **BGCOLOR**: The background color of the image specified by comma-delimited RGB values. The range is from (0,0,0) to (1,1,1).
- **FEATURE_TYPES**: White space delimited feature type names
- **OUTPUT_FORMAT**: The value must be one of:
  - image/png
  - image/gif
  - image/jpeg
  - image/svg+xml
- **TRANSPARENT**: Determines whether the image will be transparent. Values are TRUE or FALSE.
- **WIDTH**: Width of the output image (in pixels)
- **HEIGHT**: Height of the output image (in pixels)

  ***Note:*** *If a Coordinate System or Bounding Box is not submitted with the WMS query, default values are used. You can configure these default values within the Web User Interface.*

***Web Application Properties***

The following table lists the properties file path and web application properties of the OGC Web Mapping service.

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. Can be specified as a network name or an IP.<br><br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br><br>Default value = 7071 (the default Transformation Manager listening port) |
| CONNECTION_POOL_EXPIRY | The length of time in seconds in which an FME Server connection remains idle before expiring from the connection pool.<br><br>If a value of less than or equal to zero is specified, there is no connection pooling.<br><br>Default value = 300 |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| **Key Property** | **Description** |
|---|---|
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a messageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message.<br><br>Default value = true |
| INCLUDE_THREAD_NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message.<br><br>Default value = false |
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used.<br><br>Default value = true |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| ECHO_LOG_TO_CONSOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one.<br><br>Default value = 3000 |
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| PLUGINS_DIR | The directory that the OGC Servlet will scan for plugins. This is the directory where any new jar files will be placed to install new services for the OGC servlet. |
| OSGi_CACHE_DIR | The directory where the OGC Servlet will keep the OSGi information required to maintain the plugin architecture. |
| SCHEMA_PATH | This is the path to the XML schema for the WFS CityGML support. |

**Properties file**

*<WebAppDir>*/fmeogc/WEB-INF/conf/propertiesFile.properties

| Key Property | Description |
|---|---|
| WFS_CAPABIL-ITIESORDER | Defines the order in which the capabilities will be displayed in the GetCapabilities response for the WFS service.<br><br>Use character ; to separate the capabilities. |
| WMS_CAPABIL-ITIESORDER | Defines the order in which the capabilities will be displayed in the GetCapabilities response for the WMS service.<br><br>Use character ; to separate the capabilities. |
| WFS_VERSIONS | Defines the supported WFS versions. |
| WMS_VERSIONS | Defines the supported WMS versions. |
| SECURITY_MODE | The security mode of this web service. |
| DEFAULT_USER_ID | The default user ID to log in FME Server. |
| DEFAULT_PASSWORD | The default password for default user ID. |
| WFS_CLIENT_ID | The client ID assigned to WFS service. |
| WMS_CLIENT_ID | The client ID assigned to WMS service. |
| WFS_ENGINE_SUB-SECTION | The subsection name used for workspace trans-formation (WFS service). |
| WMS_ENGINE_SUB-SECTION | The subsection name used for workspace trans-formation (WMS service). |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

`<WebAppDir>/fmeogc/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| FAILOVER_SERVER_NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

**Example 1**

Sample GetCapabilities Request

http:/-
/loca-
lhost/fmeogc/Samples/austinWMS.fmw?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1

**Example 2**

Sample GetMap Request

http:/-
/loca-
lhost/fmeogc/Samples/austinWMS.fmw?SERVICE=WMS&VERSION=1.1.1&REQUEST=GetMap&LAYE
cenart&STYLES=&FORMAT=image/png&SRS=EPSG:4326&BBOX=98.7,29,96.5,
31.5&WIDTH=560&HEIGHT=300

## Utility Services

### Token Service

This service allows users to generate security tokens using a user account name and password.

By default, a token is tied to the client's credentials (username and password) and is valid for a specific period of time. A token can be retrieved manually by logging into FME the Token Service web service in a web browser or programmatically.

Token authentication provides a convenient way for your web applications to invoke FME Server web services and REST API. Instead of embedding pregenerated tokens into an application, your application should request a token from token service dynamically using user name and password via HTTPS. Once the token is obtained, the application can apply it to service requests via HTTP. In this context tokens should have short lifetime to be secure, such as one day.

Token authentication is applicable to the following FME Server web services:

- Data Download Service
- Job Submitter Service
- Data Streaming Service
- KML Network Link Service

- Catalog Service
- REST Service

If you would like to tie a token to the requester's IP address, then you need to set the following parameter to true in the web application property files of the Token service and all other services:

```
SECURE_CLIENT_ADDRESS=true
```

If this property is set to true, every token generated by the FME Token service will be bound to the requester's IP address. When a token gets validated by web services, the client's IP address will also be checked for validity.

## Service URL

```
https://<host>:<port>/fmetoken/service/generate?<parameters>
```

### *Service Request Parameters*

The supported parameters are:

| Parameter | Description |
|---|---|
| opt_user | The user ID |
| opt_password | The user password |
| opt_expiration | The token's expiration time |
| opt_timeunit | The unit of expiration time (optional) |

The available values for **timeunit** are:

- **second**
- **minute**
- **hour**
- **day**

### *Web Application Properties*

The following table lists the properties file path and web application properties of the Token service.

**Properties file**

`<WebAppDir>/fmetoken/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. Can be specified as a network name or an IP.<br><br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br><br>Default value = 7071 (the default Transformation Manager listening port) |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |
| MIN_TIME_BETWEEN_REQS | This is the minimal amount of time that a client must wait before sending another request. If a request is sent before that interval, it will be ignored. The time is specified in milliseconds.<br><br>Default value = 1000 |

**Properties file**

`<WebAppDir>/fmetoken/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log). <br><br> A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a messageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message. <br><br> Default value = true |
| INCLUDE_THREAD_NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message. <br><br> Default value = false |
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used. <br><br> Default value = true |

**Properties file**

*<WebAppDir>*/fmetoken/WEB-INF/conf/prop-
ertiesFile.properties

| Key Property | Description |
|---|---|
| ECHO_LOG_TO_CON-SOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one.<br><br>Default value = 3000 |
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| SECURE_CLIENT_ADDRESS | A boolean value that determines if token authentication includes the client address. Set it to true if enhanced security is required.<br><br>Default value = false |
| SECURITY_CLIENT_ID | The client ID assigned to this web service. |
| MIN_EXPIRATION | The minimum expiration time for token (5 minutes). If the expiration time is not specified, it will default to the minimum expiration.<br><br>Default value = 300 |

**Properties file**

*<WebAppDir>*`/fmetoken/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| MAX_EXPIRATION | The maximum expiration time for token (30 days).<br><br>Default value = 2592000 |
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

| **Properties file** | |
|---|---|
| `<WebAppDir>/fmetoken/WEB-INF/conf/propertiesFile.properties` | |
| **Key Property** | **Description** |
| FAILOVER_SERVER_NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

Here is an example URL request using the FME Token Service API:

```
https://<
host
>:<
```

```
port>/fmetoken/service/generate?user=admin&password=admin&expi-
ration=3&timeunit=day
```

The token service first authenticates the specified user ID and password credentials. It will return an HTTP Not Authorized error code if these credentials are not accepted.

In this example request, the token service will generate a token for user **admin** with password **admin** which will expire after 3 days. The resulting token will be returned to the client as plain text via HTTPS. The default time unit is **second** if not specified.

> ***Note:*** *Although the FME Token Service can be deployed using unsecured HTTP, it's highly recommended that it be deployed as a secured web service using the HTTPS protocol only.*

### Data Upload Service

Users can upload source data or other file-based resources for FME Server Workspaces.

File upload is available on the Configure page for any workspace with a published parameter that represents a file used by the workspace.

Once data is uploaded, users can select the uploaded data for the workspace to use.

Currently, uploaded files are temporary and disappear after a set period of inactivity. Uploaded files are stored in a private location accessible to FME Server but not the user.

***Request Elements***

### Supported Request Methods

This section describes the various methods available for interacting with the Data Upload service. The methods supported for sending files are PUT and POST. The HTTP GET method is supported for browsing files that are already uploaded.

Remember that only volatile storage is supported and that all requests must be associated with an FME Server workspace.

| Request | Request Description | HTTP Method | Target URI |
|---|---|---|---|
| Upload a single file | You can upload a file using the HTTP PUT method. | PUT | http://host[:port]/fmedataupload/<repository>/<workspace>/<filena |
| Upload a single file or multiple files | You can upload a single file or multiple files using a simple form submission that uses the HTTP POST | POST | http://host[:port]/fmedataupload/<repository>/<workspace> |

| Request | Request Description | HTTP Method | Target URI |
|---|---|---|---|
| Browse Uploaded Files | You can acquire a listing of all uploaded files for a session by making a GET request to either the same URI used to upload files originally, or, in the case of archives, append to this an archive file and/or a file path within the archive for a listing of files there.<br><br>You can use the optional path component at the end to show the contents of a sub-folder within an archive file. To report just the root contents of the archive file, omit the optional path part.<br><br>For example, the request URI /fmedataupload/Samples/austinWFS.fmw/myfile.zip/pictures/ shows a listing of all files contained within the pictures subfolder of the archive file named myfile.zip. | GET | http://host[:port]/fmedataupload/<repository>/<workspace>/<ar file name>/[<filename or path within archive>] |

## Service Specific Request Parameters

| Name | Value | Description |
|---|---|---|
| opt_fullpath | true \| false Default: false | When enabled the service response includes the physical or absolute path for each file that the FME Engine can use. |
| opt_extractarchive<br>This parameter is not available in HTTP GET because extracting archives is an operation and not a state flag. | true \| false Default: false | When enabled the service extracts uploaded zip files and returns the file contents inside of zip files.<br>In this case, the file names are logical paths relative to the zip files' container.<br>For example, a SHAPE file named roads.shp inside of a roads.zip file is returned as roads.zip/roads.shp. |
| opt_responseformat | xml \| json Default: xml | Defines the language of the response. The text must be all lowercase. |
| opt_pathlevel | -1 \| 0 \| n Default: 1 | This parameter indicates how many levels of recursion deep to show file and folder information.<br>A value of-1 denotes complete recursion as deep as there is.<br>A value of 0 means do not return any information at all.<br>Any other positive number denotes the number of levels of recursion. For example, a value of 1 would request only the files in the requested path; that is, only children of the requested path and not descendants. |

### Preserving Session State

Java EE lets you explicitly specify the session ID of a request URL. This specification ensures that the correct session state is preserved for a session when cookies do not work properly or are not available. Simply append the following parameter immediately following the path part of the request URI:

```
;jsessionid=<session id>
```

***Response Elements***

| Element | Child Elements | Value | Description |
|---|---|---|---|
| statusInfo | status | success \| failure | The service status |
| file, folder, archive | name | string | The name of the file that was uploaded |
| | size | long integer | The size in bytes of the file uploaded.The size element does not apply to folders and is absent beneath folder elements. |
| | path | string | The absolute or physical path of the file (which is used by an FME Engine) |
| session | none | string | A session identifier that is always included in each data upload response as an HTTP cookie as well as in the body of the response. |

> ***Note:*** *There will always be a folder node with an empty name attribute. This folder node is the placeholder for the directory that contains the uploaded files for the scope defined by the current user session, workspace name and repository name. This placeholder can be used later when requesting full paths to retrieve the full path to the parent directory for all uploaded files in your scope.*

If opt_extractarchive is set to false, the following scenarios results:

- If the requested path inspects the contents of a zip or archive file, then an error response results.
- If the requested path is valid, but the opt_pathlevel is 2 or higher, then a path level of 1 is implied.

### *Workspace Published Parameters*

Any workspace which includes published parameters representing file-based source data or resources.

### *Web Application Properties*

The following table lists the properties file path and web application properties of the Data Upload service.

**Properties file**

`<WebAppDir>/fmedataupload/WEB-INF/conf/propertiesFile.properties`

| KeyProperty | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. Can be specified as a network name or an IP.<br><br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br><br>Default value = 7071 (the default Transformation Manager listening port) |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |
| ENABLE_SECURITY | A boolean value that determines if security is enabled. |
| SECURITY_CLIENT_ID | The client ID assigned to this web service. |

**Properties file**

`<WebAppDir>/fmedataupload/WEB-INF/conf/prop-ertiesFile.properties`

| KeyProperty | Description |
|---|---|
| DEFAULT_USER_ID | The default user ID to log in FME Server. |
| DEFAULT_PASSWORD | The default password for default user ID. |
| UPLOAD_DIR | The upload directory. |
| DEFAULT_FORMAT | The default response format (typically JSON).<br><br>Default value = json |
| LOG_FILE_NAME | The file name that the log files will have in com-mon. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the mes-sageLogger resource bundle, which is a mes-sageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |
| INCLUDE_TIMES-TAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message.<br><br>Default value = true |
| INCLUDE_THREAD_ NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message.<br><br>Default value = false |

**Properties file**

`<WebAppDir>/fmedataupload/WEB-INF/conf/propertiesFile.properties`

| KeyProperty | Description |
|---|---|
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used.<br><br>Default value = true |
| ECHO_LOG_TO_CONSOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one.<br><br>Default value = 3000 |
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |

**Properties file**

`<WebAppDir>/fmedataupload/WEB-INF/conf/propertiesFile.properties`

| KeyProperty | Description |
|---|---|
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

`<WebAppDir>/fmedataupload/WEB-INF/conf/propertiesFile.properties`

| KeyProperty | Description |
|---|---|
| FAILOVER_SERVER_NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

**Example 1**

Queries all files that have been uploaded under Samples/austinWFS.fmw, but does not extract any archives or inspect beneath them.

***Request***

http://localhost/fmedataupload/Samples/austinWFS.fmw?opt_extractarchive=false

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
  <statusInfo>
    <status>success</status>
  </statusInfo>
  <session>2748D67C99FC2EDBA2160B0215339226</session>
  <files>
    <path />
    <folder>
      <name />
    </folder>
    <archive>
      <size>3826063</size>
      <name>bcalberta.zip</name>
    </archive>
    <archive>
      <size>17337905</size>
      <name>Desktop2.zip</name>
    </archive>
    <archive>
      <size>294731</size>
      <name>parcel_M26.zip</name>
    </archive>
    <archive>
      <size>1374228</size>
      <name>pipes.zip</name>
    </archive>
    <file>
      <size>21684</size>
      <name>csv2shape.fmw</name>
    </file>
    <file>
      <size>53</size>
      <name>SDS.csv</name>
    </file>
```

```
        <file>
           <size>231279</size>
           <name>helmethair.png</name>
        </file>
     </files>
  </serviceResponse>
```

### Example 2

Queries all files uploaded under Samples/austinWFS.fmw and, in addition to this, extracts any archive files that were present.

***Request***

http://localhost/fmedataupload/Samples/austinWFS.fmw?opt_extractarchive=true

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
<statusInfo>
<status>success</status>
   </statusInfo>
   <session>2748D67C99FC2EDBA2160B0215339226</session>
   <files>
      <path />
      <folder>
         <name />
      </folder>
      <file>
         <size>21684</size>
         <name>csv2shape.fmw</name>
      </file>
      <file>
         <size>53</size>
         <name>SDS.csv</name>
      </file>
      <file>
         <size>231279</size>
         <name>helmethair.png</name>
      </file>
      <archive>
```

```
         <name>bcalberta.zip</name>
      </archive>
      <archive>
         <name>Desktop2.zip</name>
      </archive>
      <archive>
         <name>parcel_M26.zip</name>
      </archive>
      <archive>
         <name>pipes.zip</name>
      </archive>
   </files>
</serviceResponse>
```

### Example 3

Retrieves the names and full paths of all files uploaded to Samples/austinWFS.fmw, including any archive files and archive file contents to three levels deep.

***Request***

http://localhost/fmedataupload/Samples/austinWFS.fmw?opt_extractarchive=true&opt_path
 level=3&opt_fullpath=true

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
   <statusInfo>
         <status>success</status>
   </statusInfo>
   <session>2748D67C99FC2EDBA2160B0215339226</session>
   <files>
      <path />
      <folder>
            <name />
            <path>$(FME_DATA_REPOSITORY)/Sa-
            mples/aus-
            tin.fmw/AB4F3D23F97885FBD23A32E245/</path>
      </folder>
      <file>
```

```
          <size>70013</size>
          <name>bcalberta.zip/bcalberta/AEROFACP.ffs</name>
          <path>$(FME_DATA_REPOSITORY)/Sa-
          mples/austin.fmw/AB4F3D23F97885FBD23A32E245
   /__fmearch_name_bcalberta.zip/bcalberta/AEROFACP.ffs</path>
     </file>
      <file>
        <size>1010226</size>
        <name>bcalberta.zip/bcalberta/COASTL.ffs</name>
        <path>$(FME_DATA_REPOSITORY)/Sa-
        mples/austin.fmw/AB4F3D23F97885FBD23A32E245
   /__fmearch_name_bcalberta.zip/bcalberta/COASTL.ffs</path>
     </file>
     <file>
        <size>578807</size>
        <name>bcalberta.zip/bcalberta/INWATERA.fsi</name>
        <path>$(FME_DATA_REPOSITORY)/Sa-
        mples/austin.fmw/AB4F3D23F97885FBD23A32E245
   /__fmearch_name_bcalberta.zip/bcalberta/INWATERA.fsi</path>
     </file>
     <archive>
        <name>bcalberta.zip</name>
        <path>$(FME_DATA_REPOSITORY)/Sa-
        mples/austin.fmw/AB4F3D23F97885FBD23A32E245
   /__fmearch_name_bcalberta.zip</path>
     </archive>
     <folder>
        <name>bcalberta.zip/bcalberta</name>
        <path>$(FME_DATA_REPOSITORY)/Sa-
        mples/austin.fmw/AB4F3D23F97885FBD23A32E245
   /__fmearch_name_bcalberta.zip/bcalberta</path>
        </folder>
     </files>
   </serviceResponse>
```

**Example 4**

Queries all files within the uploaded file titled bcalberta.zip at Samples/austinWFS.fmw three levels deeper, including any nested archive files within that archive. Any of these nested archives are also extracted.

***Request***

http://localhost/fmedataupload/Samples/austinWFS.fmw/bcalberta.zip?opt_extractarchive=true&opt_pathlevel=3

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
  <statusInfo>
    <status>success</status>
  </statusInfo>
  <session>2748D67C99FC2EDBA2160B0215339226</session>
  <files>
    <path>/bcalberta.zip</path>
    <folder>
      <name />
    </folder>
    <file>
      <size>70013</size>
      <name>bcalberta/AEROFACP.ffs</name>
    </file>
    <file>
      <size>1010226</size>
      <name>bcalberta/COASTL.ffs</name>
    </file>
    <file>
      <size>578807</size>
      <name>bcalberta/INWATERA.fsi</name>
    </file>
    <folder>
      <name>bcalberta</name>
    </folder>
  </files>
      </serviceResponse>
```

## Example 5

Retrieves the names and full paths of all uploaded files at Samples/austinWFS.fmw including any previously extracted files and folders to

three levels deep.
 It does not extract any archives it encounters, so the term *three levels deep* becomes meaningful only when a request was made previously to extract any associated archives.

***Request***

http://localhost/fmedataupload/Samples/austinWFS.fmw?opt_extractarchive=false&opt_pathlevel=3&opt_fullpath=true

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
  <statusInfo>
    <status>success</status>
  </statusInfo>
  <session>2748D67C99FC2EDBA2160B0215339226</session>
  <files>
    <path />
    <folder>
      <name />
      <path>$(FME_DATA_REPOSITORY)/Sa-
      mples/austin.fmw/AB4F3D23F97885FBD23A32E245/</path>
    </folder>
    <archive>
      <name>bcalberta.zip</name>
      <path>$(FME_DATA_REPOSITORY)/Sa-
      mples/aus-
      tin.fmw/AB4F3D23F97885FBD23A32E245/bcalberta.zip</path>
    </archive>
  </files>
</serviceResponse>
```

## Example 6

Queries all uploaded files at Samples/austinWFS.fmw including any previously extracted files and folders to three levels deep.
 No encountered archives are extracted, so the term three levels deep only becomes meaningful when a request was made previously to extract any associated archives.

### Request

http://localhost/fmedataupload/Samples/austinWFS.fmw?opt_extrac-tarchive=false&opt_pathlevel=3

### Response

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
  <statusInfo>
    <status>success</status>
  </statusInfo>
  <session>2748D67C99FC2EDBA2160B0215339226</session>
  <files>
    <path />
    <folder>
      <name />
    </folder>
    <archive>
      <size>3826063</size>
      <name>bcalberta.zip</name>
    </archive>
    <archive>
      <size>17337905</size>
      <name>Desktop2.zip</name>
    </archive>
    <archive>
      <size>294731</size>
      <name>parcel_M26.zip</name>
    </archive>
    <archive>
      <size>1374228</size>
      <name>pipes.zip</name>
    </archive>
    <file>
      <size>21684</size>
      <name>csv2shape.fmw</name>
    </file>
    <file>
      <size>53</size>
      <name>SDS.csv</name>
    </file>
```

```
      <file>
        <size>231279</size>
        <name>helmethair.png</name>
      </file>
    </files>
  </serviceResponse>
```

### Example 7

Attempts to query the contents of the bcalberta.zip file at Samples/austinWFS.fmw, but fails. The failure is based on the presumption that the bcalberta.zip archive file wasn't previously extracted and that the optional parameter specifies not to extract any archives in this request either.

***Request***

http://localhost/fmedataupload/Samples/austinWFS.fmw/bcalberta.zip?opt_extractarchive=false

***Response***

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<serviceResponse>
  <statusInfo>
    <status>failure</status>
  </statusInfo>
  <session>2748D67C99FC2EDBA2160B0215339226</session>
</serviceResponse>
```

## Web Connection (SOAP Communication) Service

The Web Connection service was previously referred to as the Soap Communication service. The Web Connection service provides an underlying SOAP-based communication mechanism for client-service applications. Therefore, this is not a user-oriented service that delivers translation results. FME Workbench uses the SOAP service to connect to FME Server via http.

This service works with valid SOAP requests based on the following web services description language (WSDL):

```
http://<host>/fmesoap/IFMEServerSoapService?wsdl
```

A SOAP client should send requests to the following URL:

```
http://<host>/fmesoap/IFMEServerSoapService
```

A SOAP response is returned to the client through HTTP.

***Web Application Properties***

The following table lists the properties file path and web application properties of the Web Connection (SOAP Communication) service.

**Properties file**

```
<WebAppDir>/fmesoap/WEB-INF/conf/propertiesFile.properties
```

| Key Property | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. Can be specified as a network name or an IP.<br><br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br><br>Default value = 7071 (the default Transformation Manager listening port) |
| CONNECTION_POOL_EXPIRY | The length of time in seconds in which an FME Server connection remains idle before expiring from the connection pool.<br><br>If a value of less than or equal to zero is specified, there is no connection pooling.<br><br>Default value = 300 |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |

**Properties file**

`<WebAppDir>/fmesoap/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| MIN_TIME_BETWEEN_ REQS | This is the minimal amount of time that a client must wait before sending another request. If a request is sent before that interval, it will be ignored. The time is specified in milliseconds.<br><br>Default value = 1000 |
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a messageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message.<br><br>Default value = true |
| INCLUDE_THREAD_ NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message.<br><br>Default value = false |

**Properties file**

*<WebAppDir>*`/fmesoap/WEB-INF/conf/propertiesFile.properties`

| **Key Property** | **Description** |
|---|---|
| APPEND_TO_EXIST-ING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used.<br><br>Default value = true |
| ECHO_LOG_TO_CON-SOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before cre-ating a new one.<br><br>Default value = 3000 |
| MAX_LOG_FILE_AGE | An integer value that specifies the number of sec-onds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| SECURITY_CLIENT_ID | The client ID assigned to this web service. |

| **Properties file** | |
|---|---|
| `<WebAppDir>/fmesoap/WEB-INF/conf/propertiesFile.properties` | |
| **Key Property** | **Description** |
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

`<WebAppDir>/fmesoap/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| FAILOVER_SERVER_ NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

## Catalog Service

The Catalog service allows us to create web links that users can use to download workspaces, templates, custom formats and custom transformers from FME Server using FME Workbench via `.fmc` files.

The catalog service allows registered workspaces to be browseable via the web. The purpose of this service is to make it easier to download and load various FME resources into FME Workbench from anywhere on the web. One prominent example is automatic download and loading of FME examples and demos from fmepedia.com.

There are three usages for the catalog service. They are described below.

***Generate .fmc File for an FME Workspace***

**Request Elements**

### Request URL

```
http://<host>[:port]/fmecatalog/<repository>/
<workspace>{.format}
```

### Service-Specific Request Parameters

The following table describes the service-specific options.

| Name | Value | Description |
|---|---|---|
| opt_format | xml \| json Default: xml | The format to respond in |
| opt_metadata | wps | If this option is set, the Catalog service will return WPS parameter mappings for the specified workspace. |

### Response Elements

A service response may contain the elements shown in the following table.

| Element | Child Elements | Value | Description |
|---|---|---|---|
| statusInfo | message | message string | service failure message |

***Retrieve the Catalog***

**Request Elements**

### Request URL

```
http://<host>[:port]/fmecatalog/catalog{.format}?
```

```
{name[=value]&}
```

### Service-Specific Request Parameters

The following table describes the service-specific options.

| Name | Value | Description |
|---|---|---|
| opt_format | xml \| json<br>Default: xml | The format to return the response in |
| opt_repfilter | string | Filter by repository. Use '*' at start/end to signal suffix/prefix. Do not use '*' for exact match. eg. FME_* will match repositories such as FME_workspaces. |
| opt_itemtype | WORKSPACE \| CUSTOMFORMAT \| CUSTOMTRANSFORM \| TEMPLATE | Specifies the type of repository items being returned from the catalog. All item types will be returned by default. |

**Response Elements**

A successful service response may contain the elements shown in the following table.

| Element | Child Elements | Value | Description |
|---|---|---|---|
| catalog | category | category elements | list of workspaces grouped by repository |
| category | name | string | name of the repository |
| | description | string | description of the repository |
| | workspace customFormat customTransformer | item elements | information about the workspace |
| workspace customFormat customTransformer | name | string | name of the item |
| | description | string | description of the item |
| | url | url string | url of the item |

***Note:*** *The response may be cached, so it may be necessary to reset the cache for accurate results.*

***Reset the Cached Responses***

It is important to reset the cache for the most up-to-date response.

**Request Elements**

**Request URL**

```
https://<host>[:port]/fmecatalog/resetcache
```

### *Requirements*

There are no requirements on a workspace, custom format or transformer for it to be registered with the Catalog service. However, users wishing to download cataloged items must have FME Workbench installed to perform the download.

### *Web Application Properties*

The following table lists the properties file path and web application properties of the Catalog service.

**Properties file**

`<WebAppDir>/fmecatalog/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| SERVER_NAME | The computer where the Transformation Manager is running. Can be specified as a network name or an IP.<br><br>Default value = localhost |
| SERVER_PORT | The port on which the Transformation Manager is running on the remote machine. This is an integer between 1025 and 65535.<br><br>Default value = 7071 (the default Transformation Manager listening port) |
| CONNECTION_POOL_ EXPIRY | The length of time in seconds in which an FME Server connection remains idle before expiring from the connection pool.<br><br>If a value of less than or equal to zero is specified, there is no connection pooling.<br><br>Default value = 300 |

**Properties file**

`<WebAppDir>/fmecatalog/WEB-INF/conf/propertiesFile.properties`

| **Key Property** | **Description** |
|---|---|
| CONNECTION_TYPE | The type of connection information that will be generated in each response. Permitted values (Case Is Sensitive):<br>▪ fmepedia : Instructs Workbench to pull the resources from FMEPedia<br>▪ direct : Generates FME Server host and port information for connection, not supported<br>▪ web : Generates a web connection URL for SOAP<br>Default value = web |
| FMESERVER_URL | The URL for FME Server web connection (SOAP). |
| RESOURCE_PATH | This is the path to the textual resources of the log. This file stores all the messages that will appear in the log as well as their code. |
| ENABLE_SECURITY | A boolean value that determines if security is enabled. |
| DEFAULT_USER_ID | The default user ID to log in FME Server. |
| DEFAULT_PASSWORD | The default password for default user ID. |
| SECURE_CLIENT_ADDRESS | A boolean value that determines if token authentication includes the client address. Set it to true if enhanced security is required.<br>Default value = false |
| SECURITY_CLIENT_ID | The client ID assigned to this web service. |

**Properties file**

`<WebAppDir>/fmecatalog/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| LOG_FILE_NAME | The file name that the log files will have in common. The log files will have that name with a number before the dot (such as green.log => green1825841623.log).<br><br>A subdirectory can be specified before the file name. The base file path is stored in the messageLogger resource bundle, which is a messageLogger.properties file on the hard drive (under Resources/). The LOG_FILE_NAME content is simply appended to that path. |
| INCLUDE_TIMESTAMP_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the timestamp will appear at the begining of each log message.<br><br>Default value = true |
| INCLUDE_THREAD_NAME_IN_LOG | A boolean value that determines the formatting of the log file. If set to true, the thread name will appear at the begining of each log message.<br><br>Default value = false |
| APPEND_TO_EXISTING_LOG | A boolean value that determines if new instances of the log will use the same log file or create a new file. If set to true only one log file will be used.<br><br>Default value = true |

**Properties file**

`<WebAppDir>/fmecatalog/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| ECHO_LOG_TO_CONSOLE | A boolean value that determines if log messages are passed to the console after being written to the log file. If set to true, messages will appear on the console as well as in the log.<br><br>Default value = false |
| MAX_LOG_FILE_LINES | An integer value that specifies the maximum number of lines allowed in the log file before creating a new one.<br><br>Default value = 3000 |
| MAX_LOG_FILE_AGE | An integer value that specifies the number of seconds the log file stays valid for. After this amount of time, any messages logged will be stored in a new log file.<br><br>Default value = 604800 |
| USE_CACHED_QUERY_BYDEFAULT | A boolean value that determines if the service returns cached responses.<br><br>Default value = true |
| LOCALE | The locale of the service. If no value, the default system locale is used. If specified, the specified locale is used.<br><br>The language, country and variant are separated by underscores. Language is always lower case, and country is always upper case.<br><br>Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX", "fr__MAC" |

**Properties file**

`<WebAppDir>/fmecatalog/WEB-INF/conf/propertiesFile.properties`

| Key Property | Description |
|---|---|
| FAILOVER_SERVER_NAMES | An optional list of space-separated FME Server host names. Each host named in this list must be running an FME Server.<br><br>If this service cannot connect to (or loses its connection to) its primary FME Server, it will attempt to connect to the first FME Server system named in the list. If connection failure is similarly encountered with this system, this service will attempt to connect to the next system in the list.<br><br>This failover sequence is followed until all FME Server systems in the list have been tried once. After this, a subsequent connection failure will trigger a final attempt by this service to connect to the original, primary FME Server system again. If this also fails, this service returns an error message to the client.<br><br>To use FAILOVER_SERVER_NAMES, uncomment the line and replace <namelist> with a space-separated list of one or more FME Server host names. Do not include the angle-brackets in the namelist. For example:<br><br>FAILOVER_SERVER_NAMES=red green blue |

***Examples***

## Example 1

A catalog service failure response in XML format

***Request***

http://localhost/fmecatalog/Samples/doesNotExist.fmw

***Response***

```
<serviceResponse>
   <statusInfo>
      <message>Repository item "doesNotExist.fmw" is not
      found in repository "Samples"</message>
   </statusInfo>
</serviceResponse>
```

## Example 2

A catalog service failure response in JSON format

***Request***

http://localhost/fmecatalog/doesNotExist.fmw.json

***Response***

```
{
   "serviceResponse":{
      "statusInfo":{
         "message":"Repository item \"doesNotExist.fmw\" is
         not found in repository \"Samples\""
      }
   }
}
```

## Example 3

A catalog retrieval request in XML

***Request***

http://localhost/fmecatalog/catalog

***Response***

```
<catalog>
   <repository>
      <name> MyCategory</name>
      <description/>
      <workspaces>
         <workspace>
            <name>D007-kml-update-controller.fmw</name>
```

```
          <description/>
          <title/>
          <url>http://SHOCKWAVE:8080/fmecatalog/FME_TEM-
          PLATES_MyCategory/D007-kml-update-con-
          troller.fmw</url>
        </workspace>
      </workspaces>
    </repository>
    <repository>
      <name>Samples</name>
      <description/>
      <workspaces>
        <workspace>
          <name>D007-kml-update-controller.fmw</name>
          <description/>
          <title/>
          <url>http://S-
          HOCKWAVE:8080/fmecatalog/Samples/D007-kml-update-
          controller.fmw</url>
        </workspace>
      <workspace>
          <name>austinApartments.fmw</name>
          <description>...</description>
          <title>City of Austin: Apartments and other (ACAD
          2 KML)</title>
          <url>http://S-
          HOCK-
          WAVE:8080/fmecatalog/Samples/austinApartments.fmw</url>
        </workspace>
      <workspace>
          <name>austinDownload.fmw</name>
          <description>Designed to be used with the Data
          Download Service.</description>
          <title>City of Austin: Data Download</title>
          <url>http://S-
          HOCK-
          WAVE:8080/fmecatalog/Samples/austinDownload.fmw</url>
        </workspace>
      </workspaces>
      <customFormats>
        <customFormat>
```

```
          <name>ALIROADS.fds</name>
          <description/>
          <title/>
          <url>http://S-
          HOCK-
          WAVE:8080/fmecatalog/Samples/ALIROADS.fds</url>
        </customFormat>
      </customFormats>
      <customTransformers>
        <customTransformer>
          <name>123.fmx</name>
          <description/>
          <title/>
          <url>http://S-
          HOCKWAVE:8080/fmecatalog/Samples/123.fmx</url>
        </customTransformer>
      </customTransformers>
    </repository>
  </catalog>
```

## Example 4

A catalog retrieval request in XML using prefix filter

***Request***

http://localhost/fmecatalog/catalog?opt_repfilter=My*

***Response***

```
  <catalog>
    <repository>
      <name>MyCategory</name>
      <description/>
      <workspaces>
        <workspace>
          <name>D007-kml-update-controller.fmw</name>
          <description/>
          <title/>
          <url>http://SHOCKWAVE:8080/fmecatalog/FME_TEM-
          PLATES_MyCategory/D007-kml-update-con-
          troller.fmw</url>
        </workspace>
        </workspaces>
    </repository>
</catalog>
```

## Example 5

A catalog retrieval request in JSON using suffix filter.

***Request***

http://localhost/fmecatalog/catalog?opt_repfilter=*amples

***Response***

```
{
    "catalog":{
        "repository":{
            "name":"Samples",
            "description":"",
            "workspaces": {
                "workspace":[{
                    "name":"D007-kml-update-controller.fmw",
                    "description":"",
                    "title":"",
                    "url":"http:/-
                    /SHOCKWAVE:8080/fmecatalog/Samples/D007-kml-
                    update-controller.fmw"
                    }, {
                "name":"austinApartments.fmw",
                "description":"",
                "title":"City of Austin: Data Download",
                "url":"http:/-
                /SHOC-
                KWAVE:8080/fmecatalog/Samples/austinApartments.fmw"
                }]
            }
        }
    }
}
```

## REST Service

For REST Service documentation, including web application properties and API Reference, see the FME Server REST API Specifications.

## FME Engines

FME Engines process job requests by running FME Workspaces. Each FME Engine processes a single request at a time. FME Server processing can be scaled by adding additional FME Engines to the same computer or to separate computers within the same distributed FME Server environment.

In a distributed environment FME Engines run on a computer or computers that are separate from the FME Server host. Administrators can configure the FME Engines to register with a failover FME Server host, which acts as a backup if the primary FME Server host fails.

### FME Server Configuration Files

Certain situations require that you edit configuration files. There are different configuration files for separate FME Server components. The files and their locations are listed in the following tables.

Each configuration file contains additional information and help on various parameters or directives.

| **Process Monitor** | |
|---|---|
| Filename | processMonitorConfig.txt |
| Location | `<FMEServerDir>\Server\` |
| Purpose | Configure processMonitor to start FME Server and FME Engines. Use this file to add additional FME Engines or, in a distributed environment, to start only the components you want to start. |
| **FME Server** | |
| Filename | fmeServerConfig.txt |
| Location | `<FMEServerDir>\Server\` |
| Purpose | Configure FME Server, specifically the Transformation and Repository Managers. Use this file to configure the database used by FME Server, Job Routing, and miscellaneous settings for the Transformation and Repository Manager. |
| **FME Engine** | |
| Filename | fmeEngineConfig*.txt |

| | |
|---|---|
| Location | `<FMEServerDir>\Server\` |
| Purpose | Configure each FME Engine. Use one of these files to configure each FME Engine including subsections, pre and post commands, and success and failure responses. More details on FME Engine configuration can be found in Configuring FME Engines. |
| **FME Console** | |
| Filename | FMEServerConsoleConfig.txt |
| Location | `<FMEServerDir>\Clients\FMEServerConsole\` |
| Purpose | Configure FME Server Console. Use this file to set the FME Server host used by the FME Server Console. |
| **FME Server Service** | |
| Filename | propertiesFile.properties |
| Location | `<WebAppDir>/<FMEServerService>/WEB-INF/conf/` For example: `C:/Tomcat/webapps/fmedatastreaming/WEB-INF/conf/propertiesFile.properties` |
| Purpose | Configure properties for FME Server services. Use this file to set the host for the service and various log file settings. |

### Configuration File Layout

An Engine configuration file is comprised of several sections, each of which performs a specific set of tasks. The configuration file can contain two components: a global section and zero or more subsections.

The general layout looks like this:

```
GLOBAL_SECTION
   <GlobalDirectives>
SUB_SECTION <name>
   <SubsectionDirectives>
```

### Global Section

The Global section defines all global directives that apply to a single FME Engine session. A session is defined as the period of time when an FME

Engine is started up, performs some number of translations, and is shut down.

The Global section has the following general form:

```
GLOBAL_SECTION \
[FME_WORKING_DIR <results-dirpath> \]
[FME_MAPPING_DIR <repository-root-dirpath> \]
[FME_RESULT_LIFETIME <results-lifetime-seconds> \]
[FME_PURGE_INTERVAL <results-deletion-interval-seconds> \]
[AUTO_DIR_PREFIX <dir_prefix> \]
[AUTO_FILE_PREFIX <file_prefix> \]
[RECEIVE_TIMEOUT 0 \
[FME_SHARED_RESOURCE_DIR <shared_dir>
[FME_SERVER_LOG_FILE <server-logfilepath>\]
[MACRO_DEF FME_DATA_REPOSITORY <data-repository-dirpath>\]
[FME_ENGINE_MEMORY_REDLINE <percentage> \]
[SUCCESS_RESPONSE <message> \]
[FAILURE_RESPONSE <message> \]
```

The global directives are described in the following table. It’s important to remember that the subsection directives override the global directives.

| GLOBAL_SECTION Directive | Description |
|---|---|
| FME_WORKING_DIR <results-dir-path> | Specifies the directory path into which the FME Server writes all translation results unless the FME_WORKING_DIR is specified in the subsection for a particular job.<br>**Default:** Current Working Directory<br><br>**Warning:** Take care to ensure that the *FME_WORKING_DIR* value is set to the appropriate directory. This location is where the FME Engine writes its translation results. This is important – you can set the FME Engine to periodically and recursively delete all files in this directory that are older than a certain age (specified by *FME_RESULT_LIFETIME*). When the *FME_WORKING_DIR* is set to the wrong directory, the FME Server deletes files in this directory instead of writing the translation results here. |
| FME_MAPPING_DIR <repository-root-dirpath> | Specifies the FME Server's repository root directory path, where workspaces run by the FME Engine during translations are contained.<br>**Default:** Current Working Directory |
| FME_RESULT_LIFETIME <results-lifetime-seconds> | Specifies how long, in seconds, translation results are held. The FME Engine periodically checks for, and recursively deletes, all files older than this amount from the FME_WORKING_DIR as a housekeeping measure. The frequency of this automatic deletion check is set by the value of FME_PURGE_INTERVAL, described next. Whenever automatic file deletion is performed, it occurs after a translation.<br>**Default:** If value is zero or directive is absent, results are never deleted – lifetime is infinite. |

| GLOBAL_SECTION Directive | Description |
|---|---|
| FME_PURGE_INTERVAL <results-deletion-interval-seconds> | Specifies, in seconds, the automatic deletion checking period. The FME Engine checks for files older than FME_RESULT_LIFETIME and deletes any it finds. This check is performed at the FME_PURGE_INTERVAL period. Note that the FME Engine decides whether or not to do the check only after each translation. It does not set a timer to trigger the check. This means that the check can occur at the interval set, but may also occur at longer intervals depending on how frequently translations are made.<br>**Default:** If this directive is absent, the default is 0 – checks after each translation if there are translation results to purge. |

| GLOBAL_SECTION Directive | Description |
|---|---|
| AUTO_DIR_PREFIX <dir_prefix> and AUTO_FILE_PREFIX <file_prefix> | The FME Engine configuration file uses the two pseudo-variables:S<br>`!FME_AUTO_DIR_NAME!`<br>`!FME_AUTO_FILE_NAME!`<br>to represent unique names for translation result directories and files respectively. By default, the names have this form:<br>`FME_<cnnnnnnnnnnnnn>`<br>For example:<br>`FME_a03508997017636`<br>You can change the default FME_ prefix to a site-specific value. Site-specific prefixes are defined using the AUTO_DIR_PREFIX directive to set the prefix of directory names, and the AUTO_FILE_PREFIX directive to set the prefix for file names. The directives have the following syntax:<br>`AUTO_DIR_PREFIX <prefix>`<br>`AUTO_FILE_PREFIX <prefix>`<br>The most useful application of these directives is to set the filename prefix with the AUTO_FILE_PREFIX directive. Doing so allows the names of translation result files to better reflect the originating organization.<br>When either of these directives is present, the specified *<prefix>* value is used instead of FME_. When absent, the prefix defaults to the FME_ default value.<br>For example, if AUTO_FILE_PREFIX Noodle_ was specified, the translation result filenames would take this form:<br>`Noodle_<cnnnnnnnnnnnnn>`<br>Note that a trailing underscore character in the prefix value is not assumed. If you want your prefix to have one of these characters, you must explicitly include it as part of the prefix value.<br>You can specify either or both of these directives in the global section and/or in any subsection of the configuration file.<br>When specified in the global section, the prefix is used by all subsections that don't themselves specify one. Subsections that do specify a prefix use that one instead, and do not use |

| GLOBAL_SECTION Directive | Description |
|---|---|
| | any global prefix that was specified.<br>**Default:** When the directive is absent, a prefix of FME_ is used. |
| RECEIVE_TIMEOUT <timeout_period_ms> | This timeout is measured in milliseconds. When the FME Engine doesn't receive any translation requests within this period of time, the FME Engine shuts itself down. The FME Server system then starts a new FME Engine instance.<br>Under most operating conditions this value is set to its recommended default of zero. This setting means that the FME Engine waits indefinitely for requests and never shuts down due to a lack of incoming requests.<br>However, you can use this directive to specify a finite timeout period in environments where a network monitor shuts down connections that remain inactive beyond a preset period of time. When this is done to the connection on which an FME Engine is listening to receive requests, the FME Engine can no longer respond and enters a hung state.<br>Setting this directive to a non-zero value lets the FME Engine terminate itself after the specified time. The FME Engine effectively breaks out of the hung state, allowing the FME Server system to start a new FME Engine instance that can respond, once again, to requests. |
| FME_SHARED_RESOURCE_DIR <shared_dirs> | Specifies the file paths of one or more root directories. These directories are used by the FME Engine to obtain different types of shared resources, such as custom format, transformer, and coordinate system definitions. Use semicolons to separate multiple shared directory file paths.<br>**Default:**<FMEServerDir>\Server\resources\shared |

| GLOBAL_SECTION Directive | Description |
|---|---|
| FME_SERVER_LOG_FILE <engine-log-filepath> | Specifies the file path to the FME Engine's log file. This log file contains a dump of the Engine's configuration file and other information relating to the overall operation of the FME Engine. Note that this file does not log individual translation request processing, which is logged separately in individual translation log files. For each FME Engine and, therefore, for each fmeEngineConfig.txt, you need to specify a different fmeEngine.log filename. You can use the pseudo variable !FME_INSTANCE_NAME! to create a log file named for the instance name like this: `<FMEServerDir>\Logs\fmeEngine_!FME_INSTANCE_NAME!.log` This directive includes an additional parameter that follows the file path. The parameter's value can be `TRUE` or `FALSE`, indicating whether or not the log file should be appended to the time of each FME Engine startup. `TRUE` means the file is appended to and `FALSE` means it is overwritten. **Default:**`<FMEServerDir>\Logs\fmeEngine.log FALSE` |
| MACRO_DEF FME_DATA_REPOSITORY <data-repository-dir-path> | Specifies the directory used by FME Server's Web User Interface for temporarily uploading data. In a distributed environment, this directory must be accessible to the web server and to the FME Engines. Specifying a UNC path is recommended. |

| GLOBAL_SECTION Directive | Description |
|---|---|
| FME_ENGINE_MEMORY_REDLINE <factor> | The automatic resource manager determines the optimal total memory the FME Engine process should use. It also dynamically allocates this total memory optimally to the algorithms within FME requesting it.<br>The FME_ENGINE_MEMORY_REDLINE directive is a hint to the FME Engine on how aggressive it should be in consuming memory. It takes a value between 0 and 1 (0.5 is the default value). For more aggressive memory usage, a value above 0.5 should be used. For safer memory usage, a value below 0.5 should be used. The risk in being too aggressive is the process running out of memory or the machine thrashing. The risk in being to conservative is that the process will take longer to complete.<br>**Default:** 0.5 |
| **Response Messages**<br>SUCCESS_RESPONSE<br>FAILURE_RESPONSE | The SUCCESS_RESPONSE and FAILURE_RESPONSE directives are specified at the global level. They define the message string that's returned by the FME Engine for successful and failed translations respectively. These directives provide the mechanism by which the FME Engine communicates results back to the client.<br>The content of these response messages is meaningful only to the client. The FME Engine simply passes the message as defined back to the client, which then processes it in whatever manner it chooses. The message can include the predefined pseudo-variables and several other directives. |

## Special-Purpose Directives

In addition to the common Global section directives listed in the preceding table, the following special-purpose directive are also available when required. It's important to remember that the subsection directives override the global directives.

| | |
|---|---|
| Special Purpose Directives | **SDE30_PERM_CONNECT <host> <instance> <database> <userID> <password>** |

| | |
|---|---|
| | Defines a permanent connection to an ESRI SDE Server. This connection is brought up just before the first translation is performed on the FME Engine. The connection is then held by the Engine so subsequent translations need not establish a new SDE connection. Take care when using this directive because SDE connections are a valuable resource and you should use them sparingly. In general, it's good practice to first use the FME Engine without any permanent connection. You may consider a permanent connection later, if you find that connecting and disconnecting to SDE is too expensive. |
| `<host>` | The name of the host computer on which the SDE server is running.<br>**Default:** Site-specific |
| `<instance>` | The SDE instance to which the FME Engine is to connect.<br>**Default:**`port:5151` |
| `<database>` | The database on the instance that is to be connected. When the SDE is on databases such as Oracle the value specified is not used. Although you can specify any value, the convention is to specify the value NOTUSED for the database.<br>**Default:** Site-specific |
| `<userID>` | The user account used to log in to the SDE.<br>**Default:** Site-specific |
| `<password>` | The user password of the user account.<br>**Default:** Site-specific |

## Subsections

Subsections are optional and when present their directives override global directives. When a subsection is present it's named with a keyword by which client applications can reference it. You can define multiple subsections for various purposes.

In an FME Server environment, each service (such as Data Download, Data Streaming, WFS, WMS, and so forth) has its own subsection.

If a job request — including one from an FME Server service — specifies a subsection, the FME Engine runs that job using the directives in the specified subsection. The following table shows the FME Server services and the corresponding subsections used by the FME Engine when running a job request from that service.

> ***Note:*** *Subsection directives take precedence over global directives.*

| FME Server Service | FME Engine SUB_SECTION |
|---|---|
| Data Download Service | FILE_DOWNLOAD_SERVICE |
| Data Streaming Service | STREAM_DOWNLOAD_SERVICE |
| OGC WFS Service | WFS_SERVICE |
| OGC WMS Service | WMS_SERVICE |
| FME Server Console | SERVER_CONSOLE_CLIENT |
| FMEServerJobSubmitter transformer (default subsection) | SERVER_JOB_SUBMITTER |
| Job Submitter Service | JOB_SUBMITTER_SERVICE |
| REST Service | REST_SERVICE |
| FME Server Scheduler | SERVER_SCHEDULER |

Each subsection can specify operations that the FME Engine should perform before and after a translation request is processed. The subsection can also define what translation success and failure response messages get returned to the client, and can define FME workspace macro values as well.

In addition to their own directives, subsections can use a number of directives available to the Global section to override the global values for that particular subsection.

By specifying a subsection by name in the translation request, clients can cause the operations defined by the subsection to be performed by the FME Engine for the translation being requested.

Each subsection has the following general form:

```
SUB_SECTION <keyword> \
[<Global Directives> \]
[FME_TRANSFORMATION_LOG_DIR <logfile-dirpath> \]
```

```
    [MACRO_DEF <macroName> <macroValue> \]*
    [PRE_COMMAND <pre-command> \]*
    [POST_COMMAND <post-command> \]*
    [POST_COMMAND_ALL <post-command> \]*
    [POST_COMMAND_SUCCESS <post-command> \]*
    [POST_COMMAND_FAILURE <post-command> \]*
    [SUCCESS_RESPONSE <message>  \]
    [FAILURE_RESPONSE <message>  \]
```

The subsection directives, which override global directives, are described in the following table.

| **SUB_SECTION Directives** | **Description** |
|---|---|
| SUB_SECTION <key-word> | Identifies the subsection that's being defined. There is no limit to the number of subsections that you can define. Identify each subsection with a unique `<keyword>` – it cannot begin with `FME_`. |
| <Global Directives> | The following global directives are also available to sub-sections where you can redefine them to have different values:<br>`FME_WORKING_DIR`<br>`FME_MAPPING_DIR`<br>`AUTO_DIR_PREFIX`<br>`AUTO_FILE_PREFIX`<br>`SUCCESS_RESPONSE`<br>`FAILURE_RESPONSE`<br>Redefined directives override the global value for the particular subsection for which they are defined. |
| FME_TRANS-FORMATION_LOG_DIR <logfile-dirpath> | Specifies the path to the default directory containing the per-translation log files generated by the FME Engine for that subsection.<br>**Default:** `<FME-ServerDir>\Logs\<subsectionSpecificDir>` |

| SUB_SECTION Directives | Description |
|---|---|
| MACRO_DEF <macroName> <macroValue>> | Defines a macro value that is supplied to the FME workspace when the translation is performed. The first parameter `<macroName>` specifies the name of the macro.<br>All the tokens up to the next directive or the end of the `SUB_SECTION` definition form the value for `<macroValue>`.<br>You can nest macro references.<br>When defined, macros are available for use in pre- and post-commands, and in success and failure response strings, using the pseudo-variable (!macroName!) syntax.<br>Some FME Engine subsections define a specific macro called `FME_SERVER_DEST_DIR` whose value is set as a path to a unique, system-generated directory. The intention is that this directory is used as an output location for translation result files. Workspace authors can reference this value as a published parameter and use it as a well-defined location for destination datasets. |

| SUB_SECTION Directives | Description |
|---|---|

**Pre- and Post-Commands**

The PRE_COMMAND and POST_COMMAND directives define commands placed in an FME Engine configuration file subsection that are performed before or after the translation is run.
There's no limit to the number of pre- or post-commands you can define in a SUB_SECTION. The commands are run in the order specified in the SUB_SECTION. Any command capable of being run from a command line from the operating system on which the FME Engine is running can be specified as a pre- or post-command[1].
These commands may take additional directives to more accurately define how successive commands are handled upon failure.
Available PRE_ and POST_COMMANDs are listed here:

| | |
|---|---|
| PRE_COMMAND | Defines an operating system command that's set to run *before* the FME Engine performs the translation. |
| POST_COMMAND | See POST_COMMAND_SUCCESS. |
| POST_COMMAND_SUCCESS | Defines an operating system command that's to run *after* the FME Engine performs the translation. This command only runs when the FME translation is successful. |
| POST_COMMAND_FAILURE | Defines an operating system command that's set to run *after* the FME Engine performs the translation. This command only runs if the FME translation fails. |

[1]When running FME Server on a Windows computer, some operating system commands may not work. This situation happens because Windows makes a distinction between some of its operating system commands being true OS commands (that is, inherent to the OS) and some being programs accessible to the OS. To ensure that a pre- or post-command works in a Windows environment, use the following syntax:`cmd.exe /c <command>`

For example:

```
POST_COMMAND cmd.exe /c copy file_a file_b
```

| **SUB_SECTION Directives** | **Description** |
|---|---|
| POST_COMMAND_ALL | Defines an operating system command that's set to run *after* the FME Engine performs the translation. This command runs regardless of the FME translation success or failure status. |
| Additionally, both pre- and post-commands may take an optional directive that indicates how the FME Engine behaves when that pre- or post-command fails. The available directives include the following: | |
| !ABORT_WITH_ERROR! | When the pre- or post-command fails, the entire translation is stopped at that point. No further processing is performed, including any subsequent pre- or post-commands. |
| !CONTINUE_WARN! | When the pre- or post-command fails, the translation continues and a warning is given. |
| !CONTINUE_NO_WARN! | When the pre- or post-command fails, the translation continues however no warning is given. |
| **Response Messages** SUCCESS_RESPONSE FAILURE_RESPONSE | The SUCCESS_RESPONSE and FAILURE_RESPONSE directives define the message string that's returned by the FME Engine for successful and failed translations respectively. These directives provide the mechanism by which the FME Engine communicates results back to the client. The content of these response messages is meaningful only to the client. The FME Engine simply passes the message as defined back to the client, which then processes it in whatever manner it chooses. The message can include the predefined pseudo-variables and several other directives. |

## Pseudo Variables

There are a number of *pseudo-variables* you can specify within the GLOBAL_SECTION and SUB_SECTION definitions.

Pseudo-variables act as placeholders that are replaced at translation time with the appropriate values, as described in the following table. Pseudo-variable names are always enclosed within exclamation marks.

Pseudo-variables are used to customize the behavior of the FME Engine for the following purposes:

- to assist with creating files and directories
- to incorporate specific information in response messages

The pseudo-variables that you can specify are described in the following table.

| Pseudo-Variable Name | Replacement Value Description |
|---|---|
| `!FME_AUTO_FILE_NAME!` | `A file path value with an auto-generated filename component whose name is guaranteed to be unique.`[1]<br>The file path represented by this pseudo-variable is the path specified by the `FME_WORKING_DIR` global directive plus the unique filename appended to it. The filename itself is generated before the translation so that multiple references to `!FME_AUTO_FILE_NAME!` will identify the same file throughout a single translation. |
| `!FME_AUTO_FILE_NAME_SIMPLE!` | The unique filename component in the `!FME_AUTO_FILE_NAME!` file path. |

[1]When you include a filename extension with this pseudo-variable, it matters where you place the extension. Different behavior happens depending on where the extension is included – on the inside or on the outside of the exclamation point delimiters of the `!FME_AUTO_FILE_NAME!` pseudo-variable. The default FME Engine configuration uses extensions that are inside the exclamation points, such as: `!FME_AUTO_FILE_NAME.log!` and `!FME_AUTO_FILE_NAME.zip!`. When extensions are inside the exclamation points, different filenames are generated for each occurrence of the same pseudo-variable name. This behavior is by design and guarantees that multiple occurrences of `!FME_AUTO_FILE_NAME.<name>!` (if they existed) produce unique, non-conflicting filenames. When extensions are outside the exclamation points, the same filename is generated for each occurrence of the same pseudo-variable name. This example gives the same name for both the .log and the .zip files: `!FME_AUTO_FILE_NAME!.log` and `!FME_AUTO_FILE_NAME!.zip`

| Pseudo-Variable Name | Replacement Value Description |
|---|---|
| !FME_AUTO_DIR_NAME! | A directory path value with an auto-generated final directory component whose name is guaranteed to be unique.<br><br>The directory path represented by this pseudo-variable is the path specified by the FME_WORKING_DIR global directive plus the final unique directory name appended to it. The final directory name itself is generated before the translation so that multiple references to !FME_AUTO_DIR_NAME! identify the same directory throughout a single translation. |
| !FME_AUTO_DIR_NAME_SIMPLE! | The name of the unique final directory component in the !FME_AUTO_DIR_NAME! directory path. |
| !FME_ERROR_MSG! | The contents of the error message that contains the reason the translation failed.<br>This value isn't available to the PRE_COMMANDS. |
| !FME_ERROR_NUMBER! | The FME Engine internal error number associated with the error message that's returned. |
| !FME_INSTANCE_NAME! | The name of the FME Engine instance as configured with a given fmeEngineConfig.txt file.<br>FME Engine instance names are usually assigned in the Engine startup commands, which are defined in the processMonitorConfig.txt file. The user-definable names typically reflect site-specific usage aspects of the FME Engine that's started. |
| !FME_NUM_FEATURES_OUTPUT! | The number of features in the translation output. |
| !FME_RESULT_LIFETIME! | The value assigned to the FME_RESULT_LIFETIME global directive. |

| Pseudo-Variable Name | Replacement Value Description |
|---|---|
| `!fmeMacroName!` | The value assigned to the named FME macro.<br>A pseudo-variable with the same name as the macro name can access the value of any macro defined to the FME Engine during translation.<br>Macros are defined to the FME Engine for a translation in one of the following three ways:<br>▪ as parameters in the translation request<br>▪ as defined in an FME workspace<br>▪ as defined in the FME Engine configuration file |

## Licensing the FME Engine

If the FME Engine is included in the installation it needs to be licensed. The licenses directory is located here:

```
<FMEServerDir>/Server/fme/licenses
```

This directory contains the following two files:

```
flexlm_config.dat.template
fme_license.dat.template
```

To license the FME Engine, complete the following steps.

1. Rename both of these files by removing the `.template` extension so they both end with `.dat`.
2. Edit the file `fme_license.dat` and replace `your_host_name` with the host running your floating license server. For example:

   ```
   SERVER <host> Any
   USE_SERVER
   ```

3. Save the file.

***Note:*** *If you don't license the FME Engine correctly, the Web User Interface and the FME Server services appear to function—the FME Server jobs will queue up, but no jobs will be completed.*

## FME Server Transformers

The following transformers work with FME Server. Refer to the FME Desktop Readers/Writers Transformer online help for further information.

- FMEServerJobSubmitter: Submits jobs to the FME Server through the web or direct connection.
- FMEServerWorkspaceRunner: Submits jobs to FME Server through the Web Services.
- FMEServerJobWaiter: Waits until all jobs, specified by Job ID, have been processed by FME Server.
- FMEServerLogFileRetriever: Retrieves the FME Translation log for a job specified by Job ID.
- ParameterFetcher: Provides access to published parameters and also a few parameters that are provided by FME Server.

## FME Server Readers/Writers

The following transformers work with FME Server. Refer to the FME Desktop Readers/Writers online help for further information.

- FME Server Repository: Used to import, export, or migrate repository and service metadata.
- FME Server Stream: Used by FME Desktop applications, such as FME Workbench, to exchange features with FME Server.
- FME Server Stream (Attribute): Used by FME Desktop applications, such as FME Workbench, to exchange attributes with FME Server.

## FME Server Core

The FME Server Core manages the job requests (queuing, request routing, scheduling) and the repository contents (workspaces, custom formats, custom transformers, data).

### Configuration File Reference

```
#########################################
# FME Server Config File Parameters.
#########################################
#
# The FME Server program is started from the command line
and takes one argument.
# This argument is the pathname of the configuration param-
eter file.
#
# Configuration parameters are used to set various operating
characteristics
# of the FME Server. Values for these parameters are read in
from this file
# when the FME Server is started. This configuration param-
eter file is an
# ASCII text file containing one parameter assignment on
each line. Each
# assignment consists of a parameter name followed by an
equal-sign followed
# by the parameter's value.
#
# For example, the line:
#
# REQUEST_PORT=7071
#
# assigns the "REQUEST_PORT" parameter a value of "7071".
#
# The FME Server is case-sensitive to parameter names. Blank
lines and lines
```

```
# beginning with the "#" character are treated as comments
and are ignored.
#
# Changes to any parameter value in this file will take
effect only upon
# subsequent restart of the FME Server.
#
# The following parameters must be present in the config
file and named exactly
# as shown:
#
#-----------------------------------------------------------
----------------
# Port and Host Assignments
#-----------------------------------------------------------
----------------
#
# SERVICE_REGISTRATION_PORT - The integer port number on
which to listen for FME Engines
# requesting to be registered as being available.
#
# REQUEST_PORT - The integer port number on which to listen
for FME Server
# requests from clients using the FME Server API.
#
# NOTIFIER_REQUEST_PORT - The integer port number on which
to listen for notification requests
# from clients using the FME Server API.
#
# NOTIFIER_REGISTRATION_PORT - The integer port number on
which to listen for subscriber
# registration requests using the FME Server API.
#
# SCHEDULER_REQUEST_PORT - The integer port number on which
to listen for scheduler requests
# from clients using the FME Server API.
```

```
#
# RELAYER_REQUEST_PORT - The integer port number on which to
listen for protocol relay requests
# from clients using the FME Server API.
#
# RELAYER_REGISTRATION_PORT - The integer port number on
which to listen for relayer
# registration requests using the FME Server API.
#
# PROCESS_MONITOR_ADMIN_PORT - The integer port number on
which the FME Server Process Monitor accepts admin requests.
# The port number specified here must match the value
assigned to the
# ADMIN_PORT parameter defined in the pro-
cessMonitorConfig.txt file used
# to configure the FME Server Process Monitor.
#
# FME_SERVER_PORT_POOL - Range of ports which can be
assigned to FME Engines, Subscribers and Protocols
# when connecting to the FME Server. Each connection uses a
single port available
# from the pool. Once all ports have been allocated, no more
FME Engine
# instances may connect to this Transformation/Repository
Manager.
#
# Using '0' means that any port may be used and there is no
limit on the
# number of connections. If '0' is used in combination with
a range of
# port numbers then it will override that range and make any
port available.
#
# Port pools may be specified as a comma-seperated list of
port numbers and
# port number ranges. For example, the port pool specified
as:
```

```
# 1234, 4567-4570, 7890
# will make all of the following ports available in the
pool:
# 1234, 4567, 4568, 4569, 4570, 7890
#
# FME_SERVER_HOST_NAME - The host name of the FME Server.
#
#-----------------------------------------------------------
----------------
# Connection Limits
#-----------------------------------------------------------
----------------
#
# MAX_PENDING_CONNECT_REQUESTS - Maximum number of pending
connect requests to allow on the
# REQUEST_PORT. Connect requests after this number will be
# rejected, rather than queued up for later processing.
#
# CHANNEL_READ_BUFFER_SIZE - Size of channel buffer used to
read in client requests. The default is
# 8192. A larger buffer size may increase the speed of read-
ing larger
# client requests, however, excessively large buffer sizes
may decrease
# the speed of reading smaller client requests.
#
#-----------------------------------------------------------
----------------
# Environment
#-----------------------------------------------------------
----------------
#
# LOCALE - The locale of the FME Server system.
#
# If no value, then the default system locale is used.
#
```

```
# If specified, the specified locale is used. The language,
country and variant are separated by
# underscores. Language is always lower case, and country is
always upper case.
# Examples: "en", "de_DE", "_GB", "en_US_WIN", "de__POSIX",
"fr__MAC"
#
# ROOT_PATH - The root path under which FME Server system
files can be found.
#
# SERVER_PATH - The root path under which Server files can
be found on FME Server.
#
# RESOURCE_PATH - The root path under which FME Server sys-
tem resources can be found. Resources
# include localisable log message files, system properties,
system state files, etc.
#
#-------------------------------------------------------------
---------------
# Transformation Management
#-------------------------------------------------------------
---------------
#
# TRANSFORMATION_OWNER - The transformation owner.
#
# ENABLE_TRANSACTION_VALIDATION_RETRIES - Can be true or
false. true to enable requeue of jobs that are in invalid
# state, otherwise false to have jobs in invalid states
treated as job failures.
# When true, jobs will be retried up to the limit of
# MAX_FAILED_TRANSACTION_REQUEST_RETRIES.
#
# ENABLE_TRANSACTION_CONNECT_VALIDATION - Can be true or
false. true to validate for invalid jobs during successful
# registration of FME Engines, otherwise false.
#
```

```
# ENABLE_TRANSACTION_REQUEST_VALIDATION - can be true or
false. true to validate for invalid jobs when FME Engines
# are woken up from being idle waiting for a new job, other-
wise false.
#
# SERVICES_USING_FAULT_TOLERATOR - boolean flag indicating
whether or not services are being
# run with the fault tolerator.
#
# A value of "true" requires that ALL services be run WITH
# the fault tolerator. "True" causes the Trans-
formation/Repository Manager to
# shutdown services that reach the MAX_TRANSACTION_RESULT_
SUCCESSES
# or MAX_TRANSACTION_RESULT_FAILURES limits. Service
restart is
# performed by the fault tolerator.
#
# A value of "false" requires that ALL services be run WITH-
OUT
# the fault tolerator. "False" causes the Trans-
formation/Repository Manager to
# shutdown and restart services that reach the MAX_TRANS-
ACTION_RESULT_SUCCESSES
# or MAX_TRANSACTION_RESULT_FAILURES limits.
#
#---------------------------------------
# Limits
#---------------------------------------
#
# MAX_TRANSACTION_RESULT_SUCCESSES - Maximum number of suc-
cessful result transactions
# to accept from the service before shutting down or
# restarting the service.
#
```

```
# MAX_TRANSACTION_RESULT_FAILURES - Maximum number of failed
result transactions
# to accept from the service before shutting down or
# restarting the service.
#
# MAX_FAILED_TRANSACTION_REQUEST_RETRIES - Maximum number of
times a failed transaction request will be automatically
# resubmitted for processing, after which the failed request
is cancelled.
# A value of < 0 means that no maximum limit is imposed, and
failed transaction
# requests will be automatically resubmitted indefinitely. A
value of 0 means that
# automatic resubmission is disabled.
#
# MAX_REGISTRATION_READ_ATTEMPTS - Maximum number of
attempts to retry reading FME Engine registration messages.
The default is 6000.
#
# REGISTRATION_READ_RETRY_WAIT - Amount of time in mil-
liseconds to wait before attempting to retry reading FME
Engine
# registration attempts. The default is 10 ms.
#
# MAINTENANCE_IDLE_WAIT - Amount of time in milliseconds to
wait before attempting to perform maintenance operations.
Currently
# Engines that have already processed jobs and that are idle
for the specified time will undergo
# maintenance which includes recycling connections and mem-
ory used by Engine. The default is 5000 ms.
# A value of 0 or less will disable maintenance.
#
# TRANSACTION_DB_RETRY_WAIT - Amount of time in seconds to
wait before attempting to retry persisting job
# DB transaction information. The default is 20 seconds. A
value less than or equal to zero
```

```
# disables retrying to establish DB connections.
#--------------------------------------
# Communication
#--------------------------------------
#
# ARE_YOU_THERE_MSG - The text string to send to a service
to confirm that it is of the
# desired type before proceeding to use it. For FME
engines, this value
# must be "FME_AreYouThere".
#
# SUCCESSFUL_RESULT_PREFIX - The substring present at the
start of the result string
# returned by a service to indicate a successful result.
For FME
# services, this value must be "0:" (a zero followed by a
colon).
#
#----------------------------------------------------------
----------------
# Schedule Management
#----------------------------------------------------------
----------------
#
# SCHEDULER_OWNER - The scheduler owner.
#
# SCHEDULER_LOG_FILENAME - The pathname of the message log
file for the scheduler.
#
# SCHEDULER_CONNECTIONPOOL_EXPIRY - The length of time in
seconds in which an FME Server connection remains idle
# before expiring from the connection pool.
#
# If a value of less than or equal to zero is specified,
then there is no
# connection pooling.
```

```
#
#------------------------------------------------------------
---------------
# Repository Management
#------------------------------------------------------------
---------------
#
# ENABLE_REPOSITORY - Can be true or false. If true, repos-
itory requests will
# be processed. Otherwise, repository requests will not be
# processed.
#
# REPOSITORY_HOME - The root path under which the repository
resides.
#
# ENABLE_NATIVE_LIBS - Can be true or false. If true, this
enables support for
# native functions such as decrypting encrypted
# workspaces. Otherwise, native functions will not be
# supported.
#
#------------------------------------------------------------
---------------
# Notification Management
#------------------------------------------------------------
---------------
#
# NOTIFIER_LOG_FILENAME - The pathname of the message log
file for the notifier.
#
# NOTIFIER_CONNECTIONPOOL_EXPIRY - The length of time in sec-
onds in which an FME Server connection remains idle
# before expiring from the connection pool.
#
# If a value of less than or equal to zero is specified,
then there is no
# connection pooling.
```

```
#
# NOTIFIER_MAX_PENDING_NOTIFICATIONS - The max number of
notifications allowed to be queued. Once the max number
# has been reached the oldest notification will be dis-
carded.
#
# NOTIFIER_MAX_NOTIFICATION_RESULT_SUCCESSES - The max
number of successful notifications. Once the max number
# has been reached the subscriber will be restarted.
#
# If a value of less than or equal to zero is specified,
then there is no
# restarting of the subscriber.
#
# NOTIFIER_MAX_NOTIFICATION_RESULT_FAILURES - The max
number of failed notifications. Once the max number
# has been reached the subscriber will be restarted.
#
# If a value of less than or equal to zero is specified,
then there is no
# restarting of the subscriber.
#
# NOTIFIER_MAX_SUBSCRIBER_ACCEPT_ATTEMPTS - The max number
of accept attempts. Each accept attempt is 10 ms so the
default is
# for the notifier to wait 5 mins for a Subscriber to com-
plete connecting after a
# successful registration.
#
#-----------------------------------------------------------
----------------
# Relay Management
#-----------------------------------------------------------
----------------
#
```

```
# RELAYER_LOG_FILENAME - The pathname of the message log
file for the relayer.
#
# RELAYER_MAX_PUBLISHER_ACCEPT_ATTEMPTS - The max number of
accept attempts. Each accept attempt is 10 ms so the default
is
# for the relayer to wait 5 mins for a publisher to complete
connecting after a
# successful registration.
#
#------------------------------------------------------------
---------------
# Security Management
#------------------------------------------------------------
---------------
#
# ENABLE_SECURITY - Can be true or false. If true, requests
will be both
# authenticated and authorized. If false, no authentication
# and no authorization of requests will occur.
#
# SECURITY_HOME - The root path under which security modules
reside.
#
# SECURITY_DEBUG - Can be true or false. If true, enables
security debug messages to appear in
# log files. If false, no security debug messages will
appear.
#
#--------------------------------------
# Authentication
#--------------------------------------
# SECURITY_LOGIN_TYPE - Identifies the authentication
(login) method to use:
# database, activedirectory, or custom.
#
#--------------------------------------
```

```
# Authentication - Active Directory
#---------------------------------------
# SECURITY_AD_SERVER_AUTODETECT - Required. Can be true or
false. Controls whether or not FME Server
# should attempt to automatically detect Active Directory
servers on
# the domain. If set to false, Active Directory servers
must be
# specified through the combination of SECURITY_AD_SERVER_
COUNT,
# SECURITY_AD_SERVER_HOSTn, and SECURITY_AD_SERVER_PORTn
parameters.
#
# Technical Note: FME Server automatically detects for
Active
# Directory servers by contacting the domain name system
(DNS)
# server and querying for service (SRV) records pertaining
to
# Lightweight Directory Access Protocol (LDAP) servers, of
which
# Active Directory is one such server. See
# SECURITY_AD_SERVER_DNS_RECORD_NAME for more information.
#
# SECURITY_AD_NT_DOMAIN - Optional. Specifies the default
NT domain name to use when authenticating
# using Active Directory. If not specified, users must
explicitly specify
# the domain name during login (e.g., domain\user vs.
user).
#
# SECURITY_AD_SERVER_DNS_RECORD_NAME - Optional. This param-
eter is only used when
# SECURITY_AD_SERVER_AUTODETECT is set to true.
#
```

```
# When using automatic detection of Active Directory
servers,
# specifies the domain name system (DNS) lookup query. If
this
# parameter is not specified, the lookup query used is
# '_ldap._tcp', omitting the domain name. If you are having
# trouble with the domain name resolution, you may wish to
try
# specifying a lookup query of the form
# '_ldap._tcp.<domain_name>', where <domain_name> is your
# domain name. For example:
#
# DNS record name: _ldap._tcp.mydomain.com
#
# SECURITY_AD_SERVER_COUNT - Optional. This parameter is
only used when
# SECURITY_AD_SERVER_AUTODETECT is set to false.
#
# When not using automatic detection of Active Directory
servers,
# indicates the number of Active Directory servers that will
be manually
# specified. A list of server hosts and ports must be spec-
ified when
# using this parameter. For example, if SECURITY_AD_SERVER_
COUNT is set
# to 3, then FME Server will look for the hosts and ports
specified in
# the parameters:
#
# Server 1: SECURITY_AD_SERVER_HOST1 / SECURITY_AD_SERVER_
PORT1
# Server 2: SECURITY_AD_SERVER_HOST2 / SECURITY_AD_SERVER_
PORT2
# Server 3: SECURITY_AD_SERVER_HOST3 / SECURITY_AD_SERVER_
PORT3
#
```

```
# SECURITY_AD_SERVER_HOSTn - Optional. This parameter is
only used when
# SECURITY_AD_SERVER_AUTODETECT is set to false.
#
# Specifies the nth server host address (n begins with 1).
#
# SECURITY_AD_SERVER_PORTn - Optional. This parameter is
only used when
# SECURITY_AD_SERVER_AUTODETECT is set to false.
#
# Specifies the nth server communication port (n begins
with 1).
# Typically, Active Directory uses port 389.
#
# SECURITY_AD_USE_SSL - Optional. Can be true or false. Con-
trols whether or not to connect to
# Active Directory over secure sockets layer (SSL). If not
specifed, SSL will
# not be used.
#
# SECURITY_AD_NAMING_CONTEXT - Optional. Specifies the base
context in which all Active Directory
# queries begin, in the form of a distinguished name. For
example, if
# your domain name is mydomain.com, the naming context
might be:
#
# DN=mydomain,DN=com
#
# Often, the domain-level object is sufficient as the nam-
ing context.
# However, for efficiency improvements in a larger Active
Directory
# tree, you may consider setting the naming context to a
lower-scoped
```

```
# object containing all relevent user accounts and security
groups for
# the context of FME Server. If this parameter is not spec-
ified, the
# default naming context provided by the Active Directory
will be used.
#
# Technical Note: The default naming context will be
retrieved from the
# domain RootDSE's 'defaultNamingContext' attribute.
#
# SECURITY_AD_PREAUTH_USERNAME - Optional. Specifies the
user name of a pre-authenticated service
# account.
#
# A pre-authenticated service account ensures that all
Active
# Directory queries will have the same read-access rights.
If not
# specified, Active Directory queries will return results
that the
# currently authenticated user has read-access to. Spec-
ifying a pre-
# authenticated service account ensures that regardless of
the
# currently authenticated user, Active Directory queries
will return
# results that the service account has read-access to.
#
# Note that a pre-authenticated service account is required
for token
# authentication.
#
# SECURITY_AD_PREAUTH_PASSWORD - Optional. Specifies the
password of a pre-authenticated service
# account.
#
```

```
# See SECURITY_AD_PREAUTH_USERNAME for a description of a
pre-
# authenticated service account.
#
#---------------------------------------
# Authorization
#---------------------------------------
# SECURITY_AUTHORIZATION_MODULE - The authorization module
to load. If not
# specified, by default the first authorization
# module found is loaded.
#
# SECURITY_PUBPARM - The published parameter used to iden-
tify user roles
# wthin a workspace.
#
# SECURITY_FME_SERVER_HOST - The host name of FME Server.
This is used to
# identify the FME Server host resource for
# authorization.
#
# SECURITY_FME_SERVER_REQUEST_PORT - The client port number
of FME Server. This
# is used to identify the FME Server client
# port resource for authorization.
#-----------------------------------------------------------
----------------
# Database Connection
#-----------------------------------------------------------
----------------
# DB_TYPE - Identifies specific database in use: post-
gresql, sqlserver, oracle.
#
# DB_DRIVER - JDBC driver name used for connecting to the
Repository database.
#
```

```
# DB_JDBC_URL - JDBC url used for connecting to the Repos-
itory database.
#
# DB_USERNAME - Repository database user name
#
# DB_PASSWORD - Repository database user password
#
# DB_CONNECT_EXPIRY - Database connection expiry in seconds
#
# DB_SQLSTMTS_PATH - The path to the SQL statement resource
bundle
#
# Examples:
#
# DB_TYPE=postgresql
# DB_DRIVER=org.postgresql.Driver
# DB_JDBC_URL=jdbc:postgresql://localhost:5432/fmeserver
# DB_USERNAME=fmeserver
# DB_PASSWORD=fmeserver
# DB_CONNECT_EXPIRY=60
# DB_SQLSTMTS_PATH=C:/Apps/FMEServer/Server/database
#
# DB_TYPE=sqlserver
# DB_DRIVER=com.microsoft.sqlserver.jdbc.SQLServerDriver
# DB_JDBC_URL=jdbc:sqlserver://localhost/SQLEXPRESS
# DB_USERNAME=fmeserver
# DB_PASSWORD=$FMEserver
# DB_CONNECT_EXPIRY=60
# DB_SQLSTMTS_PATH=C:/Apps/FMEServer/Server/database
#
# DB_TYPE=oracle
# DB_DRIVER=oracle.jdbc.driver.OracleDriver
# DB_JDBC_URL=jdbc:oracle:thin:@localhost:1521:orcl
# DB_USERNAME=fmeserver
# DB_PASSWORD=fmeserver
# DB_CONNECT_EXPIRY=60
# DB_SQLSTMTS_PATH=C:/Apps/FMEServer/Server/database
#
#------------------------------------------------------------
----------------
# Log Management
#------------------------------------------------------------
----------------
#
# ENABLE_LOGS - Can be true or false. If true, log man-
agement requests will
# be processed. Otherwise, log management requests will not
be
# processed.
#
# LOGS_HOME - The root path under which log files resides.
#
#---------------------------------------
# Log File
#---------------------------------------
#
# LOG_FILENAME - The pathname of the message log file. Note
that for Windows/NT platforms where
# the file-separator character is the backslash "\", the
pathname value
# must specify the backslash as a pair of backslashes. For
example:
#
# C:\\Temp\\Transformation\\Repository Manager.log
#
# INCLUDE_TIMESTAMP_IN_LOG - boolean flag indicating
whether or not to include a timestamp
# in the message log file. A value of "true" causes the
timestamp
# to be included. Any other value causes it to be absent.
#
```

```
# INCLUDE_THREADNAME_IN_LOG - boolean flag indicating
whether or not to include the name of
# the thread that logged the message in the message log
file.
# A value of "true" causes the thread name to be included.
# Any other value causes it to be absent.
#
# APPEND_TO_EXISTING_LOG - boolean flag indicating whether
or not to append to the end of
# an existing message log file (if one exists) or to create
a new
# one, overwriting (destroying) any same-named log file.
# A value of "true" causes messages to be appended. any
other
# value causes a new file to be created.
#
# MAX_LOGFILE_LINES - Maximum number of lines written to cur-
rent logfile, after which it is
# closed, followed by possible deletion of older ones and
creation of a
# new one to continue on with.
# value: <= 0 size limiting is NOT in effect.
# > 0 size-limiting IS in effect.
#
# MAX_LOGFILE_AGE_SECONDS - Maximum allowable age in seconds
of previous versions of logfiles. Any logfiles
# older than this are deleted. Deletion of older logfiles
only occurs
# when the current logfile exceeds MAX_LOGFILE_LINES in size
and is
# closed.
# value: < 0 previous logfiles NEVER deleted.
# = 0 ALL previous logfiles deleted.
# > 0 previous logfiles older than specified value deleted.
#
```

```
# ECHO_LOG_TO_CONSOLE - boolean flag indicating whether or
not to display (echo) all messages sent
# to the message log file to the default system output
device (console) as well.
# A value of "true" causes messages to be echoed to the con-
sole. Any
# other value prevents echoing.
#
# DEBUG_LEVEL - Can be set to NONE (the default), LOW,
MEDIUM, HIGH or SUPER_VERBOSE.
#
#-----------------------------------------------------------
----------------
# Job Routing
#-----------------------------------------------------------
----------------
# Please refer to the FME Server Administrator's Guide for
a full
# description of the Job Routing parameters and their use.
#
# TM_DEFAULT_TAG - This is the default tag name in case
Engines are not configured
# with a tag and if jobs are submitted with a tag that have
no
# associated engines
#
# Example configuration:
#
# TM_DEFAULT_TAG=default
#
# TM_ENGINE_<ConfigNumber>=<EngineName>:<TagName>
#
# This assigns Engines to specific tags. If an Engine is
not assigned to a tag, it
# defaults to using the default tag specified via the TM_
DEFAULT_TAG directive.
```

```
# Both <EngineName> and <TagName> are case sensitive.
#
# Example configuration:
#
# TM_ENGINE_1=Engine1:fast
# TM_ENGINE_2=Engine2:fast
# TM_ENGINE_3=Engine3:utility
#
# This means "Engine1" and "Engine2" are assigned the "fast"
tag and
# "Engine3" on host "panaka" are assigned the "utility" tag.
Any server jobs with the
# "fast" tag will be processed by either "Engine1" or
"Engine2". Any server jobs
# with the "utility" tag will be processed by "Engine3". Any
jobs with any other tag or
# no tag will be processed by any other Engines that have
not been assigned a tag since
# by default it will use the tag assigned to TM_DEFAULT_TAG.
#
# TM_REPOSITORY_<ConfigNumber>=<RepositoryName>:<TagName>
#
# When configured any Server jobs from the configured repos-
itories will utilize the
# specified tag which will override any client specified job
tags.
# Both <RepositoryName> and <TagName> are case sensitive.
#
# Example configuration:
#
# TM_REPOSITORY_1=Samples:fast
# TM_REPOSITORY_2=Utilities:utility
#
# This means any jobs from the "Samples" repository are
assigned the "fast" tag and
```

```
# any jobs from the "Utilities" repository are assigned the
"utility" tag.
#
# Additional Job Routing Notes: <ConfigNumber> should start
at 1 and increment sequentially.
#
#-----------------------------------------------------------
----------------
# System Notifications
#-----------------------------------------------------------
----------------
# These are optional notifications that are generated by
FME Server. The general configuration
# format is NOTIFY_<keyword>=<topic_name>.
#
#
# NOTIFY_FAILOVER - This can be used in a FAILOVER cluster
configuration. It will generate
# notifications when a node starts up, when a node has a
heartbeat failure and when a failover
# operation occurs. The intent of this notification is to
quickly alert users of potential
# system failure. The notification message is contained in
the "msg" keyword so {msg} macro
# can be used in subscription configurations.
#
# Example configuration:
#
# NOTIFY_FAILOVER=FAILOVER_TOPIC
# Note: This means when the above failover events occur
notifications will be sent to FAILOVER_TOPIC.
#
# NOTIFY_JOB_SUCCESS - This will generate a notification to
the specified topic on all successful jobs.
#
```

```
# NOTIFY_JOB_FAILURE - This will generate a notification to
the specified topic on all failed jobs.
#
#-------------------------------------------------------------
---------------
# Cluster Management
#-------------------------------------------------------------
---------------
# Please refer to the FME Server Administrator's Guide for a
full
# description of the cluster management parameters and their
use.
#
# CLUSTER_TYPE - Can be DEFAULT or FAILOVER. This specifies
the type of cluster configuration
# of the FME Server. If FAILOVER is specified, then failover
configuration parameters
# must be specified.
#
# FAILOVER_MONITOR_HOST - This is a required parameter and
specifies the host to monitor. This
# corresponds to the FME_SERVER_HOST_NAME setting of the mon-
itored host. Note: This is
# case sensitive and typically all upper case in a default
installation. It's best to
# confirm by checking the FME_SERVER_HOST_NAME value of the
monitored host.
#
# In a failover cluster configuration there are two nodes in
a cluster. Each node
# is uniquely identified by the host name and are configured
to monitor each other. On
# initial startup, the first node to start up is registered
as the "active" node". The
# second node to startup is registered as the "failover"
node. If an "active" node
```

```
# is shutdown or fails to respond, the "failover" node auto-
matically becomes "active"
# and takes over any unfinished jobs and active schedules
that were handled by the node
# that was detected to have failed. If the failed node is
started back up again, it
# automatically becomes the "failover" node. Any client
requests to a "failover" node
# are rejected and redirected to the "active" node.
#
# The following are optional failover configuration set-
tings:
#
# FAILOVER_SCAN_INTERVAL - The default is 40. This is the
number of seconds in which this node
# waits before checking if the monitor host is still alive.
#
# FAILOVER_MAX_SCAN_FAILURES - The default is 3. This is
the max number of times in which this node
# will retry detecting if the monitor host is still alive.
This setting in conjunction with
# FAILOVER_SCAN_INTERVAL determines how long a host can
appear not alive before a
# failover operation occurs. The max seconds for a host to
appear not alive is equal to
# FAILOVER_SCAN_INTERVAL x FAILOVER_MAX_SCAN_FAILURES. The
default is 40 x 3 = 120 seconds.
#
# FAILOVER_LOG_SCAN - Can be true or false. This logs out
each time the monitor host is scanned by
# this node.
#
# FAILOVER_LOG_FAILURE_DETAILS - Can be true or false. This
logs out more detailed logging
# describing failures.
#
```

```
# The following are only required when default port or owner
names have been changed:
#
# FAILOVER_MONITOR_PORT_SCHEDULER - This is the port of the
scheduler component on the host being monitored.
# This corresponds to the SCHEDULER_REQUEST_PORT setting on
the monitored host.
#
# FAILOVER_MONITOR_PORT_TRANSFORMATION - This is the port of
the core component on the host being monitored.
# This corresponds to the REQUEST_PORT setting on the mon-
itored host.
#
# FAILOVER_SCHEDULER_OWNER - This is the scheduler owner
name of the host to be monitored. By default the
# FAILOVER_MONITOR_HOST value is used which by default cor-
responds with the SCHEDULER_OWNER setitng
# of the monitored host.
#
# FAILOVER_TRANSFORMATION_OWNER - This is the scheduler
owner name of the host to be monitored. By default the
# FAILOVER_MONITOR_HOST value is used which by default cor-
responds with the TRANSFORMATION_OWNER setitng
# of the monitored host.
#------------------------------------------------------------
---------------
# ************** FME SERVER SETTINGS START **************
#------------------------------------------------------------
---------------
# Port and Host Assignments
#------------------------------------------------------------
---------------
SERVICE_REGISTRATION_PORT={SAFE{serviceRegistrationPort}}
REQUEST_PORT={SAFE{fmeserverRequestPort}}
NOTIFIER_REQUEST_PORT={SAFE{notifierRequestPort}}
NOTIFIER_REGISTRATION_PORT={SAFE{notifierRegistrationPort}}
SCHEDULER_REQUEST_PORT={SAFE{schedulerPort}}
```

```
RELAYER_REQUEST_PORT={SAFE{relayerRequestPort}}
RELAYER_REGISTRATION_PORT={SAFE{relayerRegistrationPort}}
PROCESS_MONITOR_ADMIN_PORT={SAFE{processMonitorAdminPort}}
FME_SERVER_PORT_POOL=0
FME_SERVER_HOST_NAME={SAFE{fmeserverHostname}}
#-----------------------------------------------------------
----------------
# Connection Limits
#-----------------------------------------------------------
----------------
MAX_PENDING_CONNECT_REQUESTS=500
CHANNEL_READ_BUFFER_SIZE=8192
#-----------------------------------------------------------
----------------
# Environment
#-----------------------------------------------------------
----------------
LOCALE=
ROOT_PATH={SAFE{rootDir}}
SERVER_PATH={SAFE{serverDir}}
RESOURCE_PATH={SAFE{resourcesDir}}
#-----------------------------------------------------------
----------------
# Transformation Management
#-----------------------------------------------------------
----------------
TRANSFORMATION_OWNER={SAFE{fmeserverHostname}}
SERVICES_USING_FAULT_TOLERATOR=true
#--------------------------------------
# Limits
#--------------------------------------
MAX_TRANSACTION_RESULT_SUCCESSES=100
MAX_TRANSACTION_RESULT_FAILURES=10
MAX_FAILED_TRANSACTION_REQUEST_RETRIES=3
MAX_REGISTRATION_READ_ATTEMPTS=6000
REGISTRATION_READ_RETRY_WAIT=10
```

```
MAINTENANCE_IDLE_WAIT=90000
TRANSACTION_DB_RETRY_WAIT=20
#--------------------------------------
# Communication
#--------------------------------------
ARE_YOU_THERE_MSG=FME_AreYouThere
SUCCESSFUL_RESULT_PREFIX=0:
#------------------------------------------------------------
---------------
# Schedule Management
#------------------------------------------------------------
---------------
SCHEDULER_OWNER={SAFE{fmeserverHostname}}
SCHEDULER_LOG_FILENAME=scheduler/core/fmeScheduler.log
SCHEDULER_CONNECTIONPOOL_EXPIRY=300
#------------------------------------------------------------
---------------
# Repository Management
#------------------------------------------------------------
---------------
ENABLE_REPOSITORY=true
REPOSITORY_HOME={SAFE{repositoryServerRootDir}}/Server/r-
epository
ENABLE_NATIVE_LIBS={SAFE{enableNativeLibs}}
#------------------------------------------------------------
---------------
# Notification Management
#------------------------------------------------------------
---------------
NOTIFIER_LOG_FILENAME=notifier/core/fmeNotifier.log
NOTIFIER_CONNECTIONPOOL_EXPIRY=300
NOTIFIER_MAX_PENDING_NOTIFICATIONS=50000
NOTIFIER_MAX_NOTIFICATION_RESULT_SUCCESSES=10000
NOTIFIER_MAX_NOTIFICATION_RESULT_FAILURES=100
NOTIFIER_MAX_SUBSCRIBER_ACCEPT_ATTEMPTS=30000
```

```
#-------------------------------------------------------------
----------------
# Relay Management
#-------------------------------------------------------------
----------------
RELAYER_LOG_FILENAME=relayer/core/fmeRelayer.log
RELAYER_MAX_PENDING_RELAYS=50000
RELAYER_MAX_PUBLISHER_ACCEPT_ATTEMPTS=30000
#-------------------------------------------------------------
----------------
# Configuration Management
#-------------------------------------------------------------
----------------
CONFIGURATION_MIGRATION_HOME={SAFE{repos-
itoryServerRootDir}}/Server/migration
#-------------------------------------------------------------
----------------
# Security Management
#-------------------------------------------------------------
----------------
ENABLE_SECURITY={SAFE{enableSecurity}}
SECURITY_HOME={SAFE{serverDir}}/security
SECURITY_DEBUG=false
#---------------------------------------
# Authentication
#---------------------------------------
SECURITY_LOGIN_TYPE=database
#SECURITY_LOGIN_TYPE=activedirectory
#SECURITY_AD_SERVER_AUTODETECT=true
#SECURITY_AD_NT_DOMAIN=MYDOMAIN
#---------------------------------------
# Authorization
#---------------------------------------
SECURITY_ROLES_PUBPARM=FME_SECURITY_ROLES
SECURITY_USER_PUBPARM=FME_SECURITY_USER
SECURITY_FME_SERVER_HOST={SAFE{fmeserverHostname}}
```

```
SECURITY_FME_SERVER_CLUSTER=DEFAULT
SECURITY_FME_SERVER_REQUEST_PORT=7071
SECURITY_AUTHORIZATION_MOD-
ULE=COM.safe.-
fmeserver.security.module.FMEDatabaseAuthorizationModule
#------------------------------------------------------------
--------------------
# Database Connection
#------------------------------------------------------------
--------------------
{SAFE{pgsqlComment}}DB_TYPE=postgresql
{SAFE{pgsqlComment}}DB_DRIVER=org.postgresql.Driver
{SAFE{pgsqlComment}}DB_JDBC_URL=jdbc:postgresql://{SAFE
{pgsqlHostname}}:{SAFE{pgsqlPort}}/fmeserver
{SAFE{pgsqlComment}}DB_USERNAME=fmeserver
{SAFE{pgsqlComment}}DB_PASSWORD=fmeserver
{SAFE{pgsqlComment}}DB_CONNECT_EXPIRY=60
{SAFE{pgsqlComment}}DB_SQLSTMTS_PATH={SAFE{serverDir}}/dat-
abase
{SAFE{sqlserverComment}}DB_TYPE=sqlserver
{SAFE{sqlserverComment}}DB_DRIV-
ER=com.microsoft.sqlserver.jdbc.SQLServerDriver
{SAFE{sqlserverComment}}DB_JDBC_URL=jdbc:sqlserver://{SAFE
{sqlserverHostname}}:{SAFE{sqlser-
verPort}};databaseName=fmeserver
{SAFE{sqlserverComment}}DB_USERNAME=fmeserver
{SAFE{sqlserverComment}}DB_PASSWORD=$FMEserver
{SAFE{sqlserverComment}}DB_CONNECT_EXPIRY=60
{SAFE{sqlserverComment}}DB_SQLSTMTS_PATH={SAFE{serverDir}}
/database
{SAFE{oracleComment}}DB_TYPE=oracle
{SAFE{oracleComment}}DB_DRIV-
ER=oracle.jdbc.driver.OracleDriver
{SAFE{oracleComment}}DB_JDBC_URL=jdbc:oracle:thin:@{SAFE{ora-
cleHostname}}:{SAFE{oraclePort}}:{SAFE{oracleSID}}
{SAFE{oracleComment}}DB_USERNAME=fmeserver
{SAFE{oracleComment}}DB_PASSWORD=fmeserver
```

```
{SAFE{oracleComment}}DB_CONNECT_EXPIRY=60
{SAFE{oracleComment}}DB_SQLSTMTS_PATH={SAFE{serverDir}}/dat-
abase
#-----------------------------------------------------------
----------------
# Log Management
#-----------------------------------------------------------
----------------
ENABLE_LOGS=true
LOGS_HOME={SAFE{repositoryServerRootDir}}/Logs
#---------------------------------------
# Log File
#---------------------------------------
LOG_FILENAME=fmeServer.log
INCLUDE_TIMESTAMP_IN_LOG=true
INCLUDE_THREADNAME_IN_LOG=true
APPEND_TO_EXISTING_LOG=true
MAX_LOGFILE_LINES=5000
MAX_LOGFILE_AGE_SECONDS=-1
ECHO_LOG_TO_CONSOLE=false
DEBUG_LEVEL=NONE
#-----------------------------------------------------------
----------------
# Job Routing
#-----------------------------------------------------------
----------------
#-----------------------------------------------------------
----------------
# System Notifications
#-----------------------------------------------------------
----------------
NOTIFY_FAILOVER=
NOTIFY_JOB_SUCCESS=
NOTIFY_JOB_FAILURE=
#------------------------------------------------------------
---------------
# Cluster Management
#------------------------------------------------------------
---------------
#---------------------------------------
# Default Cluster
#---------------------------------------
CLUSTER_TYPE=DEFAULT
#---------------------------------------
# Failover Cluster
#---------------------------------------
#CLUSTER_TYPE=FAILOVER
#FAILOVER_MONITOR_HOST=<FME_SERVER_HOST_NAME>
# ************** FME SERVER SETTINGS END **************
```

## Transformation Manager

The Transformation Manager controls the job queue. The job queue is sorted by priority first and then by date submitted. That mean that jobs with higher priority are executed before jobs with lower priority, even if the jobs with lower priority are submitted first.

### Transformation Manager Directives

Transformation Manager directives allow you to control how jobs are queued and processed. TM directives can be provided through the API, as well as the web services and console, in which case they are prefixed with `tm_`.

#### tag

The job routing tag to distribute jobs between FME engines. Upon receipt of a new job on the FME Server, the Transformation Manager uses this tag, if specified, to redirect the request to an engine that has been configured to accept the given tag. For more information about job routing, see the FME Server Administrator's Guide.

**ttl**

The "time to live" in the job queue in seconds. This is used in cases where jobs are time-sensitive and can become invalid or stale while waiting in the job queue. If a job is queued longer than “time to live”, the Transformation Manager removes it from the queue instead of directing it to an engine.

**priority**

An integer representing the request’s priority level. Lower values mean a higher priority. Priority values must be greater than or equal to zero.

If priority is not set, the default is 100. The range is generally 1 to 100, but there is no restriction other than it being greater than zero. The highest priority is 1. A priority less than or equal to zero defaults to 100.

## Repositories

Repositories are used to store groups of related items that have been published to FME Server (similar to directories). Items such as workspaces, source data, custom formats, custom transformers, and templates can be published in to a repository.

When a custom format or custom transformer is published to a repository, workspaces in the same repository have access to those items. In other words, workspaces can link to custom formats and custom transformers that are in the same repository, whereas workspaces in a different repository cannot do so.

FME Server Security also allows control of who can download, publish, run, or remove items from a specific repository, as well as who can create new repositories.

### Repository Database

The FME Server Core uses the repository database to store job and repository information. The database should never be edited directly, although querying the database for job history and other statistical information is common.

FME Server comes with a default database, and we strongly recommend that you use this database. However, PostgreSQL, MySQL, Oracle, and SQL Server are also supported, if there is a compelling reason to use them.

## Role-Based Access Control

FME Server security controls access to resources with role-based access control. Within an organization, users are grouped into roles. Roles are created for various job functions. Permissions to perform certain operations are assigned to specific roles.

When a user accesses a specific resource, FME Server security determines if any of the associated roles of the user has permissions to perform the requested operation on the specified resource. For example, a request by the user `fmeuser` could be to run a workspace in the Samples repository for Data Download Service. FME Server security will grant access if any of the roles for the `fmeuser` user account has permissions to Run workspaces in the Samples repository and also has access to Data Download Service.

### User Accounts

These are the users of the FME Server system. Permissions to perform particular operations are not assigned directly to users. Instead, users acquire permissions through their roles.

When FME Server is installed for the first time, default user accounts are created.

### About the Trusted User Account

There is a special account referred to as the trusted account that can be used to provided unauthenticated access to any component (for example, web service, workbench publish). By default this trusted account is named guest and assigned to the `fmeguest` role. By default the `fmeguest` role is configured to allow unauthenticated access to the web services. This means it’s possible to invoke a Web Service URL without providing any credentials.

### User Roles

Roles allow an administrator to associate a group of users with a set of resources/permissions. Users can be added and removed from a role and permissions can be added and removed from a role. By default, FME Server has the following roles created:

- `fmeuser` – Provides users access to the Web UI and Web Services
- `fmeauthor` – Provides workspace authors access to FME Server to publish, author and test new workspaces
- `fmeadmin` – Provides full access to FME Server including the Web UI
- `fmeguest` – Provides unauthenticated access to run jobs via Web Service URLs
- `fmesuperuser` – Authorized to access all resources of FME Server, including both existing and newly created resources. By default, FME Server assigns the "admin" account to both the `fmeadmin` and `fmesuperuser` roles.

**Resources**

FME Server resources can be secured to control access depending on the permissions that are assigned. Additional security resources can be added via the Web UI if new resources are created. For example, a new repository could be created or a new custom Web Service. Resources are categorized as:

- **Applications** – These are both desktop and web client applications that access FME Server
- **Components** – These are FME components that access FME Server. Transformers and readers/writers can be used by workspace authors when creating workspaces.
- **Repositories** – These are all the repositories of FME Server. When a new repository is created, a security resource for the repository is automatically created.
- **Services** – These are all the Services of FME Server. When a new service is created, a security resource for the Service is automatically created enabling all access.

### Permissions

Permissions are assigned or un-assigned depending on whether access to a resource should be granted. FME Server provides fine-grained access control for repositories as well as allows control of which users may create new repositories and new Services.

### Securing Client Applications

All FME applications and components that access FME Server require user credentials (such as username and password) for authentication.  To ensure further security control to the administrator, all clients also provide a service client ID that is used for authorization.

#### *Web Service Client Identifier*

Every resource, including web service, has a client identifier that identifies the FME Server client calling context.

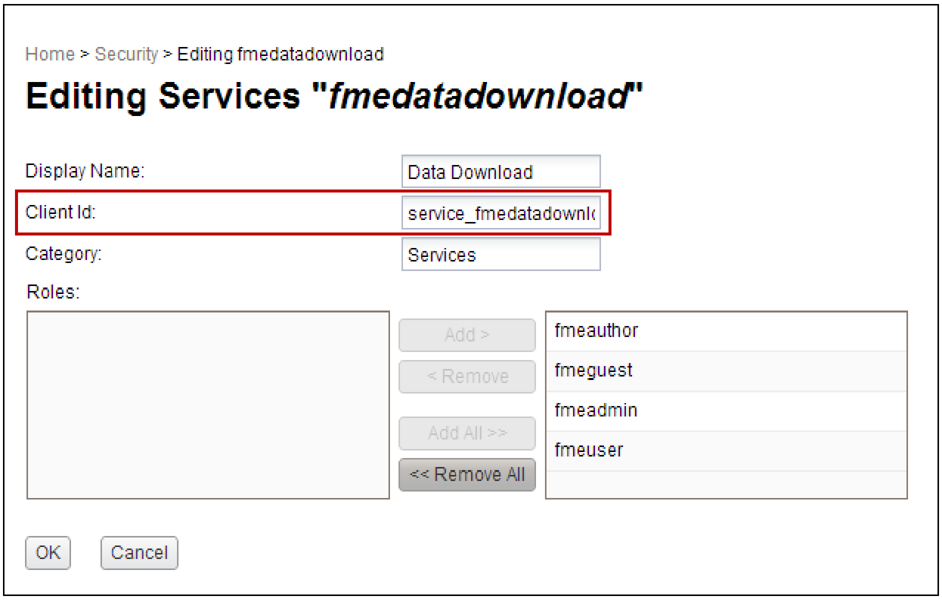
**To view the client ID assigned to a resource:**

1. Navigate to the Resources tab within the Security section of the FME Server Web User interface:

   ```
   http://<host>/fmeserver/security
   ```

2. Click the Resources tab.
3. Click on a resource.
4. The editing page opens, and the client ID for that resource appears in

the Client ID field.



Home > Security > Editing fmedatadownload

Editing Services "*fmedatadownload*"

| | |
|---|---|
| Display Name: | Data Download |
| Client Id: | service_fmedatadownl |
| Category: | Services |

Roles:

fmeauthor
fmeguest
fmeadmin
fmeuser

For added security, you can change the following property in the Web Service properties file:

`SECURITY_CLIENT_ID`

Web service property files are stored in:

`<WebAppDir>\<fmeServiceName>\WEB-INF\conf\propertiesFile.properties`

## Active Directory Configuration Tool

Bundled with your FME Server installation is an Active Directory configuration utility, located at:

`<FMEServerDir>/Utilities/ActiveDirectoryConfigurationConsole.bat`

This utility allows you to test various Active Directory configuration parameters and uses the same API to contact your domain controller and FME Server.

The utility presents the following menu options:

- Connect to an Active Directory server

  This allows you to test connection parameters and verify that a connection to an Active Directory server can be established.

- Browse the Active Directory

  This allows you to verify that all directory browsing options are functioning properly.

- Test NT user authentication

  This allows you to test authentication using standard NT username/password.

- Apply settings for use in FME Server

  This outputs the set of configuration parameters used in this utility. Follow the on-screen instructions to migrate these parameters into the FME Server configuration file.

## APIs

The FME Server API supports the C++, .NET and Java coding environments.

- C++ API Documentation
- .NET API Documentation
- Java API Documentation

## FME Server Bridge

The FME Server Bridge is a library that allows ETL applications to spatially enable their workflows using FME Server. Currently, Informatica PowerCenter and IBM InfoSphere DataStage utilize this library.

Please contact sales@safe.com to discuss the FME Server Bridge.

## FME Server Process Monitor

The Process Monitor is responsible for ensuring that the FME Server Core and the FME Engines are always up and running. The Process Monitor provides fault tolerance by regularly checking all processes and restarting the FME Server Core and any FME Engines when needed. There is no intervention required.

### Configuration File Reference

```
#########################################
# Process Monitor Config File Parameters.
#########################################
#
# The Process Monitor program is started from the command
line and takes one argument.
# This argument is the pathname of the configuration param-
eter file.
#
# Configuration parameters are used to set various operating
characteristics
# of the Process Monitor. Values for these parameters are
read in from this file
# when the Process Monitor is started. This configuration
parameter file is an
# ASCII text file containing one parameter assignment on
each line. Each
# assignment consists of a parameter name followed by an
equal-sign followed
# by the parameter's value.
#
# For example, the line:
#
# ECHO_LOG_TO_CONSOLE=true
#
# assigns the "ECHO_LOG_TO_CONSOLE" parameter a value of
"true".
#
```

```
# The Process Monitor is case-sensitive to parameter names.
Blank lines and lines
# beginning with the "#" character are treated as comments
and are ignored.
#
# Changes to any parameter value in this file will take
effect only upon
# subsequent restart of the Process Monitor.
#
# The following parameters must be present in the config
file and named exactly
# as shown:
#
# RESOURCE_PATH - the root path under which FME Server sys-
tem resources can be found. Resources
# include localisable log message files, system properties,
system state files, etc.
#
# LOG_FILENAME - the pathname of the message logfile.
#
# ECHO_LOG_TO_CONSOLE - boolean flag indicating whether or
not to display (echo) all messages sent
# to the message log file to the default system output
device (console) as well.
# A value of "true" causes messages to be echoed to the con-
sole. Any
# other value prevents echoing.
#
# INCLUDE_TIMESTAMP_IN_LOG - boolean flag indicating
whether or not to include a timestamp
# in the message log file. A value of "true" causes the
timestamp
# to be included. Any other value causes it to be absent.
#
# INCLUDE_THREADNAME_IN_LOG - boolean flag indicating
whether or not to include the name of
```

```
# the thread that logged the message in the message log
file.
# A value of "true" causes the thread name to be included.
# Any other value causes it to be absent.
#
# APPEND_TO_EXISTING_LOG - boolean flag indicating whether
or not to append to the end of
# an existing message log file (if one exists) or to create
a new
# one, overwriting (destroying) any same-named log file.
# A value of "true" causes messages to be appended. any
other
# value causes a new file to be created.
#
# MAX_LOGFILE_LINES - maximum number of lines written to cur-
rent logfile, after which it is
# closed, followed by possible deletion of older ones and
creation of a
# new one to continue on with.
# value: <= 0 size limiting is NOT in effect.
# > 0 size-limiting IS in effect.
#
# MAX_LOGFILE_AGE_SECONDS - maximum allowable age in seconds
of previous versions of logfiles. Any logfiles
# older than this are deleted. Deletion of older logfiles
only occurs
# when the current logfile exceeds MAX_LOGFILE_LINES in size
and is
# closed.
# value: < 0 previous logfiles NEVER deleted.
# = 0 ALL previous logfiles deleted.
# > 0 previous logfiles older than specified value deleted.
#
# The following parameters are optional:
#
```

```
# ADMIN_PORT - TCP/IP port number on which to listen for
admin requests.
#
# REQUEST_PORT - the TCP/IP port number on which the FME
Server core accepts translation requests.
# The port number specified here must match the value
assigned to the REQUEST_PORT
# parameter defined in the fmeServerConfig.txt file used to
configure the FME Server core.
#
# NOTE: Use of this port value is reserved for future use
in cases where the FME Server core
# is used to provide login authentication.
#
# SERVICE_REGISTRATION_PORT - the TCP/IP port number on
which the FME Server core accepts FME Engine registration
# requests. The port number specified here must match the
value assigned to the
# SERVICE_REGISTRATION_PORT parameter defined in the fme-
ServerConfig.txt file used
# to configure the FME Server core.
#
# SHUTDOWN_COMMAND - defines an arbitrary command string
that will be run during full FME Server shutdown
# after all monitored processes have been individually shut
down. This command is run
# after all defined per-process CMDStop_ commands have run.
The latter are described below.
#
# ENABLE_SHUTDOWN_HOOK - boolean flag indicating whether or
not to enable programmatic shutdown of FME Server
# if the latter is interactively terminated using Ctrl-C.
This occurs when FME Server
# has been started from a console command line and then
forced to an abrupt halt by entering
```

```
# Ctrl-C instead of running its shutdown script which would
cause FME Server to shut down
# in a controlled manner. Enabling the shutdown hook causes
a Ctrl-C entry to in effect
# run the same script, causing a controlled shutdown instead
of an abrupt halt.
# This programmatic shutdown is enabled only if this param-
eter is present and assigned a
# value of "true". The default is to enable Ctrl-C pro-
grammatic shutdown.
#
# NODE_NAME - the name used by this FME Server node to iden-
tify itself. If no value is assigned, the node takes
# the name of the system on which it is running.
#
# NODE_DESCRIPTION - the string used by this FME Server node
to describe itself. If no value is assigned, the node's
# description is the empty string.
#
# DEPLOYMENT_NAME - the name of the deployment to which this
FME Server node belongs. If no value is assigned, the node's
# deployment name is set to the name of the system on which
it is running.
#
# ABORT_ON_ANY_STARTUP_FAILURE - Represents a boolean flag
indicating whether or not to abort the entire FME Server
startup
# if any component (Server and/or FME Engines) fails to
start. A value of "false" causes FME
# Server startup to proceed as best it can in spite of
individual component startup failures.
# The absence of this property, or any value other than
"false", causes FME Server startup to
# abort if any component fails to start.
#
# RESTART_WINDOW - Represents the duration of the restart
window as an integer number of
```

```
# milliseconds. If a process uses up its restart attempts
limit within this
# duration, the Process Monitor ceases trying to start that
process. Values <= 0
# cause the restart window duration to be infinite. If this
property is absent, or
# has a non-integer value, a default value of 30000 (30 sec-
onds) is used.
#
# WAIT_FOR_STARTUP_TIMEOUT - Represents the maximum number
of seconds the Process Monitor will wait for a monitored
process
# to signal that it has successfully started. (See the dis-
cussion of the -MONITOR_PORT
# directive below). If this timeout period elapses the Proc-
ess Monitor
# assumes the process failed to start and moves on to the
next process to be started.
# If this property is absent, or has a value <= 0 or has a
non-integer value, a default value
# of 60 seconds is used.
#
# FAILOVER_SERVER_NAMES - optional list of space-separated
FME Server host names. Each host named in this list
# must be running an FME Server. If any FME Engine running
on this system loses its
# connection to its primary FME Server (as defined by the
REGISTER_SOCKET argument
# in the CMDFMEEngine_ commands contained in this file) and
the Engine reaches its
# restart limit using that Server, the Process Monitor will
attempt to restart the Engine
# and connect it to the first FME Server system named in
the list. If connection failure
# and restart attempt exhaustion are similarly encountered
with this system, Engine connection
```

```
# to the next system in the list is attempted. This failure
sequence is followed until all
# FME Server systems in the list have been tried once. After
this, a subsequent connection
# failure will trigger a final attempt to use the original,
primary FME Server system again.
# If this also fails, the Process Monitor stops trying to
restart the Engine.
#
#
# Command Entry Parameters
# ------------------------
#
# The next parameters are those that define the actual com-
mands that the Process Monitor should
# start in a separate process and monitor. Each parameter
represents one command/process. There can be
# one or more command parameters present in the config file.
The format for these command parameters is:
#
# CMD<suffix1>=<command1>[|log][|<noRestartReturnInteger>]
[|attempts=<n>][|count=<n>]
# CMD<suffix2>=<command2>[|log][|<noRestartReturnInteger>]
[|attempts=<n>][|count=<n>]
# CMD<suffix3>=<command3>[|log][|<noRestartReturnInteger>]
[|attempts=<n>][|count=<n>]
# ...
# CMD<suffixN>=<commandN>[|log][|<noRestartReturnInteger>]
[|attempts=<n>][|count=<n>]
#
# where <suffix> has one of the following syntaxes:
#
# FMEServer_<tag> - to start an FME Server core
# FMEEngine_<tag> - to start an FME Engine
# User_<tag> - to start a user process
#
```

```
# <tag> is a user-specified alphanumeric string uniquely
identifying each command.
#
# Here is an example:
#
# CMDFMEEngine_Engine1=/FMEServer/Server/fme/fme.exe REG-
ISTER_SOCKET JIM 7070
#
# When the Process Monitor encounters this parameter, it
starts a new FME Engine process, runs the
# command string in the process and monitors the process
for termination.
# If/when the process terminates, the Process Monitor
creates another process and re-runs the same command
# string again. The Process Monitor continues this behav-
iour until it reaches its restart attempt limit.
#
#
# Monitor Port
#
# Process CMD start commands can include the optional direc-
tive:
#
# -MONITOR_PORT <portNum>
#
# This directive specifies the TCP/IP port on which the
Process Monitor listens for a
# startup completed message from the process. Upon suc-
cessful startup completion the process
# should connect to this port and send the message string
"STARTED". No further communication occurs.
#
# If the directive specifies a <portNum> value of zero then
a free port number is dynamically
# determined by the Process Monitor at runtime. This is the
usual case. If the <portNum> value is specified as
```

```
# non-zero and positive then that literal port number is
used.
#
# If the process does not implement the monitor port capabil-
ity then the
# directive should be omitted from the command string.
#
# See the "WAIT" section below for further details on how
the -MONITOR_PORT directive is used.
#
#
# Suffix Flags
#
# Additionally, a start command string can end with any of
the following
# optional pipe-delimited suffix flags (their order of
appearance is not significant,
# but if present they must appear at the end of the actual
command string, separated
# from it and each other by a pipe character):
#
# "log"
#
# If a start command string includes this suffix flag, proc-
ess output generated by the command
# will be logged to the message log file. If a start command
string does not include this suffix,
# its process output is not logged to the log file. This
mechanism can be used to enable/disable
# process output on a per command basis.
#
# When process output is logged, each output line will be
prefixed by the command's <tag> string.
# This helps to identify which process command generated
each output line.
#
```

```
# The example start command above would have no logging of
its process output. To enable
# process output logging for it, the command would be mod-
ified to be:
#
# CMDFMEEngine_Engine1=/FMEServer/Server/fme/fme REGISTER_
SOCKET JIM 7070|log
#
# The resulting logged output lines from this process would
be prefixed by "Engine1".
#
#
# <noRestartReturnInteger>
#
# This suffix flag consists of an integer number rep-
resenting a status value
# returned to the Process Monitor by the monitored process
when the latter
# terminates. If a start command string specifies a value
for this suffix flag, then
# if the monitored process terminates and returns the spec-
ified status value to
# the Process Monitor, the latter WILL NOT restart the ter-
minated process. If
# this suffix flag value is absent, or it is present but
the terminating process
# returns a status value different from the specified
value, the Process Monitor
# WILL restart the terminated process. If this suffix flag
value is present but specifies an
# invalid integer value, the processMonitor ignores any
return values from the process.
#
# attempts=<n>
# This suffix flag consists of the literal string
"attempts=" followed by an integer representing
```

```
# the number of times the Process Monitor will attempt to
start the process to be monitored. If this maximum
# value is reached the Process Monitor will give up attempt-
ing to start the process. If this suffix flag
# is absent or is present and specifies a value <= 0, the
process will have no limit to the number of
# times it is restarted. If this suffix flag is present and
specifies an invalid integer value then
# the default start attempt limit value of 10 is used.
#
# Here is an example start command line using the noRestart
flag (specifying a return status value of 27),
# the "log" flag and the "attempts=" flag described above:
#
# CMDFMEEngine_Engine1=/FMEServer/Server/fme/fme REGISTER_
SOCKET JIM 7070|log|27|attempts=5
#
#
# count=<n>
#
# This suffix flag consists of the literal string "count="
followed by an integer representing
# the number of separate, simultaneous process instances
that are to be created, each one running
# its own copy of the specified command. When count values
are specified, each process is given its own
# command name consisting of the root command name and a "
(n)" suffix.
#
# For example, if "count=3" were appended to the example com-
mand above, three separate processes would be created,
# each one running an instance of the FME Engine. The cor-
responding process command names would be
# CMDFMEEngine_Engine1(0), CMDFMEEngine_Engine1(1) and
CMDFMEEngine_Engine1(2).
#
#
```

```
# SHUTDOWN
#
# Each process CMD parameter can have an optional SHUTDOWN 
parameter defining a
# custom stop command which will be run to shut down the 
process. The same
# <suffix> value must be used in both parameters for a 
given process. For example:
#
# CMDFMEServer_Core=...
# SHUTDOWNFMEServer_Core=...
#
#
# WAIT
#
# Each process CMD parameter can also have an optional WAIT 
parameter. This
# specifies the number of seconds the Process Monitor will 
wait after starting the
# process before starting the next process, if any. The 
same <suffix> value must
# be used in both parameters for a given process. For exam-
ple:
#
# CMDFMEServer_Core=...
# WAITFMEServer_Core=10
#
# For each process, the Process Monitor interprets the wait 
period specified by
# the WAIT parameter in one of two ways:
#
# 1) Monitor Port Directive is Present
#
# If the -MONITOR_PORT directive is present in the process 
CMD start command line,
# the wait period represents the maximum number of seconds
the Process Monitor
# will wait for the monitored process to signal (via the mon-
itor port) that it has
# successfully started.
#
# If the signal is received before the wait period expires,
the Process Monitor
# immediately proceeds to start the next process.
#
# If the signal has not been received by the time the wait
period expires, the
# Process Monitor stops waiting at that point and proceeds
to start the next
# process.
#
# If the process fails to start after reaching its startup
attempt limit
# (as specified by the "attempts" suffix flag), the Process
Monitor
# immediately proceeds to start the next process, even if
the wait period
# has not yet expired.
#
# To cause the Process Monitor to wait indefinitely for the
start signal, specify
# a WAIT value of zero.
#
#
# 2) Monitor Port Directive is Absent
#
# If the -MONITOR_PORT directive is absent from the process
CMD start command line,
# the wait period represents the number of seconds the Proc-
ess Monitor waits before
# continuing on to start the next process.
#
```

```
# If the process fails to start after reaching its startup
attempt limit
# (as specified by the "attempts" suffix flag), the Process
Monitor
# immediately proceeds to start the next process, even if
the wait period
# has not yet expired.
#
# Specifying a WAIT value of zero causes the Process Mon-
itor to immediately start
# the next process without waiting.
#
#
# In both of the above cases 1) and 2), if no WAIT param-
eter is specified for a
# process, the Process Monitor uses the default time value
(in seconds) as specified
# by the WAIT_FOR_STARTUP_TIMEOUT property which is defined
in this config file.
#-----------------------------------------------------------
----------------
#-----------------------------------------------------------
----------------
# Environment
#-----------------------------------------------------------
----------------
ROOT_PATH={SAFE{installDir}}
RESOURCE_PATH={SAFE{resourcesDir}}
SERVER_EXEC_PATH={SAFE{serverJava}}
ENGINE_EXEC_PATH={SAFE{fmeExec}}
LOG_FILENAME=processMonitor.log
ECHO_LOG_TO_CONSOLE=true
INCLUDE_TIMESTAMP_IN_LOG=true
INCLUDE_THREADNAME_IN_LOG=true
APPEND_TO_EXISTING_LOG=true
```

```
#------------------------------------------------------------
---------------
# Admin
#------------------------------------------------------------
---------------
ADMIN_PORT={SAFE{processMonitorAdminPort}}
REQUEST_PORT={SAFE{fmeserverRequestPort}}
SERVICE_REGISTRATION_PORT={SAFE{serviceRegistrationPort}}
ENABLE_SHUTDOWN_HOOK=true
ABORT_ON_ANY_STARTUP_FAILURE=false
RESTART_WINDOW=30000
WAIT_FOR_STARTUP_TIMEOUT=0
# To use FAILOVER_SERVER_NAMES below, please replace <namel-
ist> with a space-
# separated list of one or more FME Server host names. Do
not include the angle-
# brackets in the namelist. For example:
#
# FAILOVER_SERVER_NAMES=red green blue
#
#FAILOVER_SERVER_NAMES=<namelist>
#SHUTDOWN_COMMAND=<shutdownCommand>
#------------------------------------------------------------
---------------
# Limits.
#------------------------------------------------------------
---------------
MAX_LOGFILE_LINES=3000
MAX_LOGFILE_AGE_SECONDS=604800
#------------------------------------------------------------
---------------
# Node state.
#
# **** Place commented-out commands that must be preserved
ABOVE this line.
```

```
# **** Any appearing below will be deleted whenever this
file is updated.
#
# To prevent a command below from being run, prefix the
associated command line(s)
# with a tilde "~" character (NOT a comment "#" character).
#
# Commands to start & monitor. **** DO NOT MODIFY OR REMOVE
THIS LINE ****
#-------------------------------------------------------------
-----------------
NODE_NAME={SAFE{hostname}}
NODE_DESCRIPTION=FME Server running on {SAFE{hostname}}
DEPLOYMENT_NAME={SAFE{deploymentName}}
# Start FME Server Notifier
{SAFE{serverComment}}CMDUser_Notifier="{SAFE{notifierJava}}
" -FMESERVER_CLASSPATH "{SAFE{classpathNoQuotes}}" -Djava.l-
ibrary.path="{SAFE{fmeUtilPath}}" -Djava.security.manager -
Djava.security.policy="{SAFE{serverDir}}/sec-
urity/fmeserver.policy" -Djava.security.auth.login.config="
{SAFE{serverDir}}/security/fmeserver.config" -Xms32m -
Xmx128m -Xrs COM.safe.fmeserver.FMEServerNotifier "{SAFE
{serverDir}}/fmeServerConfig.txt" -MONITOR_PORT
0|attempts=20
{SAFE{serverComment}}SHUTDOWNUser_Notifier="{SAFE{noti-
fierJava}}" -FMESERVER_CLASSPATH "{SAFE{classpathNoQuotes}}
" -Xrs COM.safe.fmeserver.FMEServerShutdownAgent -fme-
ServerHost {SAFE{fmeserverHostname}} -fmeServerAdminPort
{SAFE{notifierRequestPort}}
{SAFE{serverComment}}WAITUser_Notifier=0
# Start FME Server Core
{SAFE{serverComment}}CMDFMEServer_Core="{SAFE{serverJava}}"
-FMESERVER_CLASSPATH "{SAFE{classpathNoQuotes}}" -Djava.l-
ibrary.path="{SAFE{fmeUtilPath}}" -Djava.security.manager -
Djava.security.policy="{SAFE{serverDir}}/sec-
urity/fmeserver.policy" -Djava.security.auth.login.config="
{SAFE{serverDir}}/security/fmeserver.config" -Xms32m -
```

```
Xmx1024m -Xrs COM.safe.fmeserver.FMEServer "{SAFE{server-
Dir}}/fmeServerConfig.txt" -MONITOR_PORT 0|attempts=20
{SAFE{serverComment}}SHUTDOWNFMEServer_Core="{SAFE{server-
Java}}" -FMESERVER_CLASSPATH "{SAFE{classpathNoQuotes}}" -
Xrs COM.safe.fmeserver.FMEServerShutdownAgent -fmeServerHost
{SAFE{fmeserverHostname}} -fmeServerAdminPort {SAFE{fme-
serverRequestPort}}
{SAFE{serverComment}}WAITFMEServer_Core=0
# Start FME Server Scheduler
{SAFE{serverComment}}CMDUser_Scheduler="{SAFE{sched-
ulerJava}}" -FMESERVER_CLASSPATH "{SAFE{classpathNoQuotes}}"
-Djava.library.path="{SAFE{fmeUtilPath}}" -Djava.s-
ecurity.manager -Djava.security.policy="{SAFE{serverDir}}
/security/fmeserver.policy" -Djava.s-
ecurity.auth.login.config="{SAFE{serverDir}}/sec-
urity/fmeserver.config" -
Dorg.quartz.scheduler.skipUpdateCheck=true -Xms32m -Xmx128m
-Xrs COM.safe.fmeserver.FMEServerScheduler "{SAFE{server-
Dir}}/fmeServerConfig.txt"|attempts=20
{SAFE{serverComment}}SHUTDOWNUser_Scheduler="{SAFE{sched-
ulerJava}}" -FMESERVER_CLASSPATH "{SAFE{classpathNoQuotes}}"
-Xrs COM.safe.fmeserver.FMEServerShutdownAgent -fme-
ServerHost {SAFE{fmeserverHostname}} -fmeServerAdminPort
{SAFE{schedulerPort}}
{SAFE{serverComment}}WAITUser_Scheduler=0
# Start FME Server Subscriber Plugin (email)
{SAFE{serverComment}}CMDUser_Subscriber1="{SAFE{sub-
scriberJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.f-
meserver.notification.plugin.FMESubscriberPluginEmail "{SAFE
{resourcesDir}}/su-
bscrib-
ers/email/subscriberConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Subscriber1=0
# Start FME Server Subscriber Plugin (logger)
{SAFE{serverComment}}CMDUser_Subscriber2="{SAFE{sub-
scriberJava}}" -FMESERVER_CLASSPATH "{SAFE
{classpathNoQuotes}}" -Xms32m -Xmx128m -Xrs COM.safe.f-
meserver.notification.plugin.FMESubscriberPluginLogger "
{SAFE{resourcesDir}}/su-
bscrib-
ers/logger/subscriberConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Subscriber2=0
# Start FME Server Subscriber Plugin (push)
{SAFE{serverComment}}CMDUser_Subscriber3="{SAFE{sub-
scriberJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.f-
meserver.notification.plugin.FMESubscriberPluginPush "{SAFE
{resourcesDir}}/su-
bscrib-
ers/push/subscriberConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Subscriber3=0
# Start FME Server Subscriber Plugin (apns)
{SAFE{serverComment}}CMDUser_Subscriber4="{SAFE{sub-
scriberJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.f-
meserver.notification.plugin.FMESubscriberPluginAPNS "{SAFE
{resourcesDir}}/su-
bscrib-
ers/apns/subscriberConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Subscriber4=0
# Start FME Server Subscriber Plugin (ftp)
{SAFE{serverComment}}CMDUser_Subscriber5="{SAFE{sub-
scriberJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.f-
meserver.notification.plugin.FMESubscriberPluginFTP "{SAFE
{resourcesDir}}/su-
bscrib-
ers/ftp/subscriberConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Subscriber5=0
# Start FME Server Subscriber Plugin (gcm)
```

```
{SAFE{serverComment}}CMDUser_Subscriber6="{SAFE{sub-
scriberJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.f-
meserver.notification.plugin.FMESubscriberPluginGCM "{SAFE
{resourcesDir}}/su-
bscribers/gcm/subscriberConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Subscriber6=0
# Start FME Server Subscriber Plugin (jms)
{SAFE{serverComment}}CMDUser_Subscriber7="{SAFE{sub-
scriberJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.f-
meserver.notification.plugin.FMESubscriberPluginJMS "{SAFE
{resourcesDir}}/su-
bscribers/jms/subscriberConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Subscriber7=0
# Start FME Server Relayer
{SAFE{serverComment}}CMDUser_Relayer="{SAFE{relayerJava}}" -
FMESERVER_CLASSPATH "{SAFE{classpathNoQuotes}}" -Djava.l-
ibrary.path="{SAFE{fmeUtilPath}}" -Djava.security.manager -
Djava.security.policy="{SAFE{serverDir}}/sec-
urity/fmeserver.policy" -Djava.security.auth.login.config="
{SAFE{serverDir}}/security/fmeserver.config" -Xms32m -
Xmx128m -Xrs COM.safe.fmeserver.FMEServerRelayer "{SAFE
{serverDir}}/fmeServerConfig.txt" -MONITOR_PORT
0|attempts=20
{SAFE{serverComment}}SHUTDOWNUser_Relayer="{SAFE{relay-
erJava}}" -FMESERVER_CLASSPATH "{SAFE{classpathNoQuotes}}" -
Xrs COM.safe.fmeserver.FMEServerShutdownAgent -fmeServerHost
{SAFE{fmeserverHostname}} -fmeServerAdminPort {SAFE{relay-
erRequestPort}}
{SAFE{serverComment}}WAITUser_Relayer=0
# Start FME Server Publisher Plugin (email)
{SAFE{serverComment}}CMDUser_Publisher1="{SAFE{pub-
lisherJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.fmeserver.relay.plugin.FMEPublisherPluginEmail "
```

```
{SAFE{resourcesDir}}/pu-
blish-
ers/email/publisherConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Publisher1=0
# Start FME Server Publisher Plugin (udp)
{SAFE{serverComment}}CMDUser_Publisher2="{SAFE{pub-
lisherJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.fmeserver.relay.plugin.FMEPublisherPluginUDP "
{SAFE{resourcesDir}}/pu-
blishers/udp/publisherConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Publisher2=0
# Start FME Server Publisher Plugin (jms)
{SAFE{serverComment}}CMDUser_Publisher3="{SAFE{pub-
lisherJava}}" -FMESERVER_CLASSPATH "{SAFE{clas-
spathNoQuotes}}" -Xms32m -Xmx128m -Xrs
COM.safe.fmeserver.relay.plugin.FMEPublisherPluginJMS "
{SAFE{resourcesDir}}/pu-
blishers/jms/publisherConfiguration.properties"|attempts=5
{SAFE{serverComment}}WAITUser_Publisher3=0
# Start FME Engine 1
{SAFE{engineComment}}CMDFMEEngine_Engine1="{SAFE{fmeExec}}"
REGISTER_SOCKET {SAFE{fmeserverHostname}} {SAFE{serv-
iceRegistrationPort}} "{SAFE{serverDir}}/fmeEngineConfig_
Engine1.txt" |log|attempts=5
{SAFE{engineComment}}WAITFMEEngine_Engine1=0
# Start FME Engine 2
{SAFE{engineComment}}CMDFMEEngine_Engine2="{SAFE{fmeExec}}"
REGISTER_SOCKET {SAFE{fmeserverHostname}} {SAFE{serv-
iceRegistrationPort}} "{SAFE{serverDir}}/fmeEngineConfig_
Engine2.txt" |log|attempts=5
{SAFE{engineComment}}WAITFMEEngine_Engine2=0
# Start FME Engine 3
# CMDFMEEngine_Engine3="{SAFE{fmeExec}}" REGISTER_SOCKET
{SAFE{fmeserverHostname}} {SAFE{serviceRegistrationPort}} "
{SAFE{serverDir}}/fmeEngineConfig_Engine3.txt" |log|-
attempts=5
```

```
# WAITFMEEngine_Engine3=0
# Start FME Engine 4
# CMDFMEEngine_Engine4="{SAFE{fmeExec}}" REGISTER_SOCKET
{SAFE{fmeserverHostname}} {SAFE{serviceRegistrationPort}} "
{SAFE{serverDir}}/fmeEngineConfig_Engine4.txt" |log|-
attempts=5
# WAITFMEEngine_Engine4=0
# Start FME Engine 5
# CMDFMEEngine_Engine5="{SAFE{fmeExec}}" REGISTER_SOCKET
{SAFE{fmeserverHostname}} {SAFE{serviceRegistrationPort}} "
{SAFE{serverDir}}/fmeEngineConfig_Engine5.txt" |log|-
attempts=5
# WAITFMEEngine_Engine5=0
# Start FME Engine 6
# CMDFMEEngine_Engine6="{SAFE{fmeExec}}" REGISTER_SOCKET
{SAFE{fmeserverHostname}} {SAFE{serviceRegistrationPort}} "
{SAFE{serverDir}}/fmeEngineConfig_Engine6.txt" |log|-
attempts=5
# WAITFMEEngine_Engine6=0
# Start FME Engine 7
# CMDFMEEngine_Engine7="{SAFE{fmeExec}}" REGISTER_SOCKET
{SAFE{fmeserverHostname}} {SAFE{serviceRegistrationPort}} "
{SAFE{serverDir}}/fmeEngineConfig_Engine7.txt" |log|-
attempts=5
# WAITFMEEngine_Engine7=0
# Start FME Engine 8
# CMDFMEEngine_Engine8="{SAFE{fmeExec}}" REGISTER_SOCKET
{SAFE{fmeserverHostname}} {SAFE{serviceRegistrationPort}} "
{SAFE{serverDir}}/fmeEngineConfig_Engine8.txt" |log|-
attempts=5
# WAITFMEEngine_Engine8=0
# Start FME Engine 9
# CMDFMEEngine_Engine9="{SAFE{fmeExec}}" REGISTER_SOCKET
{SAFE{fmeserverHostname}} {SAFE{serviceRegistrationPort}} "
{SAFE{serverDir}}/fmeEngineConfig_Engine9.txt" |log|-
attempts=5
# WAITFMEEngine_Engine9=0
# Start FME Engine 10
# CMDFMEEngine_Engine10="{SAFE{fmeExec}}" REGISTER_SOCKET
{SAFE{fmeserverHostname}} {SAFE{serviceRegistrationPort}} "
{SAFE{serverDir}}/fmeEngineConfig_Engine10.txt" |log|-
attempts=5
# WAITFMEEngine_Engine10=0
```

# FME Server Console

The FME Server Console is a command-line utility that you can use to maintain workspaces in the FME Server's repositories and to run workspaces on the FME Server.

You can use this utility to add, list, update, and remove workspaces from the repositories.

How does the FME Server Console differ from the FME Server web user interface?

You can use the FME Server web user interface to perform most maintenance and workflow tasks; however, the FME Server Console also allows you to schedule and run batch files, and to use third-party scheduling tools.

## Starting the FME Server Console

To run the FME Server Console use the following command:

```
<FMEServerDir>\Clients\FMEServerConsole\fmeserverconsole.exe
```

### Configuration File

The FME Server Console reads in a number of configuration parameters from a configuration file when it is started. The configuration file is located at:

```
<FMEServerDir>\Clients\F-
MEServerConsole\fmeserverconsoleconfig.txt
```

During installation, the parameters defined in this file are set to appropriate values and typically do not need modification. Most of the parameters define default values for FME Server Console directive options. Therefore, in most cases, these options do not need to be present on the command line because the configuration file implicitly supplies the appropriate values for these options. Directive options are described further in "Command Line Options" on page 249.

If you do specify an option value on the command line, this value overrides the one assigned to it in the configuration file for the duration of the command.

### Switching from a Direct Connection to a Web Connection

By default, the Server Console will connect to FME Server using a direct connection. To switch to a web connection:

- Open the configuration file in a text editor.
- Change the `fmeServerHost` parameter to specify the FME Server web host URL and web port.

  *For example:*

  ```
  fmeServerHost=http://fmeserver.com
  fmeServerPort=80
  ```

The remaining sections describe how to use the FME Server Console.

## Modes

The FME Server Console can be used in three possible modes:

- "Single-Shot Command-Line Mode" below
- "Interactive Command-Line Mode" on the facing page
- "Batch-File Mode" on the facing page

### Single-Shot Command-Line Mode

You enter the single-shot command-line mode by running `fme-serverconsole.exe` and specifying the desired directive and/or arguments on the same command line. In this mode the FME Server Console performs the command line and then terminates.

| **Examples:** | |
|---|---|
| 1 | `fmeserverconsole.exe municipal/sewers.fmw` |
| | This command specifies that the FME Server should run a work-space called `sewers.fmw` located in the `municipal` repository on the FME Server system. |
| 2 | `fmeserverconsole.exe LISTALL_WORKSPACES` |

**Examples:**

This command uses the `LISTALL_WORKSPACES` command to list all available workspaces in all of the FME Server repositories.

### Interactive Command-Line Mode

The interactive command-line mode can be entered in one of two ways:

- Run `fmeserverconsole.exe` on a DOS command line. Do not specify any arguments.
- Navigate to **Start > All Programs > FME Server > Console**.

The exact navigation path may vary somewhat depending on which operating system you use.

This starts an interactive FME Server Console session, as indicated by the resulting `FMEServerConsole>` prompt. You can enter any number of commands until you use the `QUIT` directive to terminate the session. The following syntax is used in the interactive mode:

```
FMEServerConsole> <command> [options]
```

**Examples:**

```
fmeserverconsole.exe

FMEServerConsole> LISTALL_SERVICES

FMEServerConsole> QUIT
```

### Batch-File Mode

The batch-file mode reads a user-created text file containing one or more directives and runs them one after the other. The file must contain one directive per line.

Directives can span multiple lines by using the "\" line continuation character at the end of an incomplete line.

The "#" character at the beginning of a line indicates a line to be skipped, such as a comment line.

**Examples:**

```
fmeserverconsole.exe -f batch.txt
```

The preceding line runs the FME Server Console in batch mode with `batch.txt` as the input file containing the commands to perform.

Sample batch file:

```
# These 2 first lines won't be read

# but the next line will be.

ADD_WORKSPACE municipal z:\wo-
rkspaces\municipal\eastside\sewers.fmw

# The next command is split across 2 lines

municipal/sewers.fmw \

-description "All municipal sewers"
```

## Directives

The directives described in this section are used with the FME Server Console. Each directive supports a set of option flags that can be used to modify the action of the directive if desired. Additional command line options are described in "Command Line Options" on page 249.

### Running Workspaces

#### "RUN_WORKSPACE <*workspace file*>" [options]

The FME Server Console is directed to run a workspace in synchronous (default) mode by specifying the name of the workspace file as an argument, in the following form:

```
    <repositoryName>/<workspaceName>.fmw
```

The specified workspace must already exist in the specified FME Server repository. Specifying an absolute path to the workspace file is not supported.

Running workspaces in synchronous mode means that the command line prompt does not return until the FME Server finishes the job, at which time the translation results are printed to the screen.

For detailed information on the options, refer to "Command Line Options" on page 249.

**Examples:**

```
RUN_WORKSPACE Samples/parcels.fmw --SourceDataset_MIF
C:\MyData\parcels\*.mif -LOG_FILENAME C:\Logs\parcels.log
TM_priority 50
```

## ABORT_JOB "<*jobID*>" [options]

Removes a job from the Transformation Manager if processing has not started. If the FME Engine has started processing the job, the processing continues.

***Running Workspace in Asynchronous Mode***

## "SUBMIT_JOB <workspace file>" [options]

Using this command is the same as using the `RUN_WORKSPACE` command.

Unlike the `RUN_WORKSPACE` command, which blocks the console until the job has finished, `SUBMIT_JOB` inserts the job into the Transformation Manager's queue and returns a job ID to you immediately.

## GET_JOB_STATUS "<*jobID*>" [options]

Returns the status of a job that was submitted to the Transformation Manager.

***Running a Workspace From an External File***

To run a workspace with parameters defined in an external file, use the XML_PARAMETER_FILE value:

```
run_workspace Samples/austinDownload.fmw -XML_PARAMETER_FILE
C:\Temp\params.xml
```

Structure the XML file as follows:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<TRANSFORMATION_REQUEST>
  <PUBLISHED_PARAMETER name="myParam1">value1</PUBLISHED_
  PARAMETER>
  <PUBLISHED_PARAMETER name="myParam2">value2</PUBLISHED_
  PARAMETER>
  <FME_DIRECTIVE name="fmeDirective">someValue</FME_DIREC-
  TIVE>
</TRANSFORMATION_REQUEST>
```

### Managing Workspaces

For supported command line options, see "Command Line Options" on the facing page.

#### REMOVE_WORKSPACE <*repositoryName*> <*workspaceName*>.fmw [options]

Removes the specified workspace from the specified FME Server repository.

#### LISTALL_WORKSPACES [options]

Lists all workspaces available in all FME Server repositories.

#### LISTWORKSPACES_BY_SERVICE [<*serviceName*>] [options]

Lists all workspaces registered with the specified service. If the `serviceName` argument is absent, all workspaces registered with all services are listed.

### Managing Workspace Resources

#### REMOVE_CUSTOM_FORMAT <*repositoryName*> <*customFormatName*> [options]

Removes the custom format from the specified repository.

#### REMOVE_CUSTOM_TRANSFORMER <*repositoryName*> <*transformerName*> [options]

Removes the custom transformer from the specified repository.

### Managing Repositories

#### ADD_REPOSITORY <*repositoryName*> [description] [options]

Creates a new repository named `repositoryName` with an optional description.

#### REMOVE_REPOSITORY <*repositoryName*> [options]

Removes the repository `repositoryName` from FME Server. All items located in the repository are removed as well.

#### LISTALL_REPOSITORIES

Lists all repositories available in the FME Server database.

### Managing Services

```
ADD_SERVICE <serviceName> <displayName> <URLPattern>
[description [options]]
```

Adds the specified service to FME Server.

```
UPDATE_SERVICE <serviceName> <URLPattern> [description]
[options]
```

Updates an existing service on FME Server.

```
REMOVE_SERVICE <serviceName> [options]
```

Removes the specified service from FME Server.

```
LISTALL_SERVICES [options]
```

Lists all services available on FME Server.

### Command-Line Help

```
HELP [directiveName]
```

Displays usage details about the specified directive. If no directive name is provided, generic usage information is shown instead.

## Command Line Options

The categories of command line options are:

- "FME Workspace Published Parameters" below
- "Console-Specific Transformation Manager Directives" below
- "Console-Specific Options" on the facing page

The first two command line option categories apply to RUN_WORKSPACE and SUBMIT_JOB commands. The last category applies to all commands.

### FME Workspace Published Parameters

These published parameters are created and exposed by the author of an FME workspace. The purpose of these parameters is to let you dynamically customize the translation.

Published parameters are specified as:

```
--<parameter name> <parameter value>
```

For example, you can specify a source dataset or a destination dataset different from the workspace’s default.

**Examples:**

```
--SourceDataset_GML C:\MyData\parcels.gml
```

### Console-Specific Transformation Manager Directives

The Transformation Manager directives allow you to customize some of the current job’s properties before being submitted to the FME Server’s Transformation Manager (TM).

For information about format-based Job Routing, see the *FME Server Administrator’s Guide*.

These directives, also called TM Directives, are specified as:

```
tm_<directive name> <directive value>
```

The "tm" prefix and the directive name must be in lower case.

Currently the supported TM Directives include the following.

- tm_ttl
- tm_priority
- tm_tag

See "Transformation Manager Directives" on page 213 for details.

### Console-Specific Options

These options let you customize some of the server console properties when executing a command. The options are specified as:

```
opt_<option name> <option value>
```

The "opt" prefix and the option name must be in lower case.

The supported options include those listed here.

#### opt_fmeServerHost

A string (without spaces) identifying the name of a host running an FME Server with which the FME Server Console should communicate.

This option is different from the fmeHost option, which is described in "Console-Specific Transformation Manager Directives" on previous page.

#### opt_fmeServerPort

An integer specifying the port number that the FME Server Console should use to communicate with an FME Server running on OPT_fmeServerHost. A valid port value is between `0` and `65535`.

#### opt_subsectionName

A string (without spaces) identifying the name of a subsection in the FME Server's configuration file that the FME Server will use for the requested translation.

#### opt_username

The username that the FME Server Console should use when connecting to the FME Server.

## opt_password

The password that the FME Server Console should use when connecting to the FME Server.