

FME® Server Authoring Training Course



Table of Contents

| | |
|--|--------|
| Welcome | 1.1 |
| Introduction | 1.1.1 |
| Course Overview | 1.1.2 |
| Course Resources | 1.1.3 |
| Introduction to FME Server | 1.2 |
| What is FME Server? | 1.2.1 |
| FME Server Roles | 1.2.2 |
| FME Server Architecture | 1.2.3 |
| Deploying FME Server | 1.2.4 |
| Workbench and FME Server | 1.2.5 |
| Publishing Workspaces | 1.2.6 |
| Exercise: Daily Database Updates: Publishing a Workspace | 1.2.7 |
| Web Interface Basics | 1.2.8 |
| Engines and Licensing | 1.2.9 |
| Running a Workspace | 1.2.10 |
| Jobs | 1.2.11 |
| Exercise: Daily Database Updates: Running a Workspace | 1.2.12 |
| Sharing in FME Server | 1.2.13 |
| FME Server Scheduling | 1.2.14 |
| Exercise: Daily Database Updates: Sharing and Scheduling | 1.2.15 |
| Source Data Management | 1.2.16 |
| Database Connections | 1.2.17 |
| Web Connections | 1.2.18 |
| Publishing Data | 1.2.19 |
| Temporary Uploads | 1.2.20 |
| Exercise: Daily Database Updates: Publishing Data | 1.2.21 |
| Resource Filesystem | 1.2.22 |
| Authoring for Resources | 1.2.23 |
| Exercise: Daily Database Updates: Using Resources | 1.2.24 |
| Running the Job Submitter using a URL | 1.2.25 |

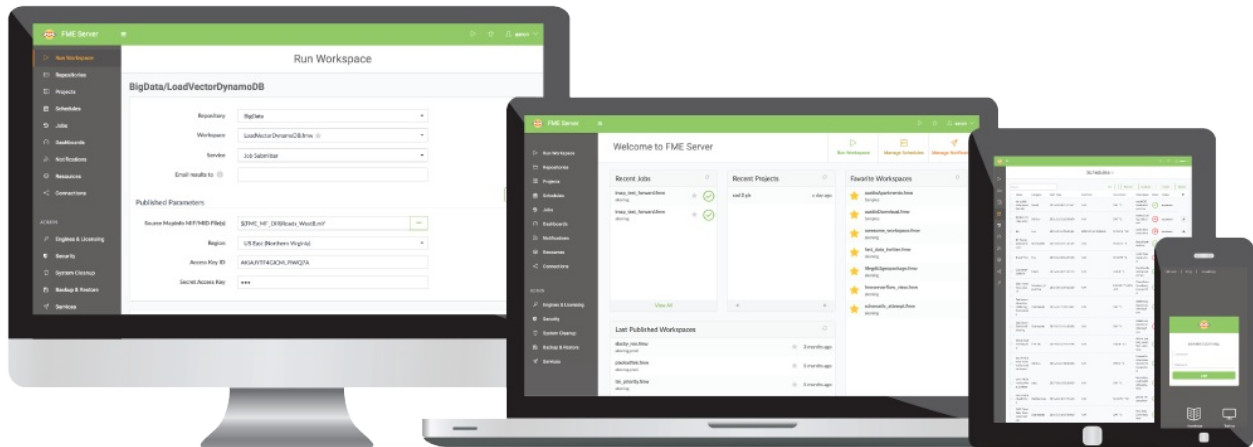
| | |
|---|---------|
| Authoring Job Chains | 1.2.26 |
| Exercise: Authoring Workspace Chains | 1.2.27 |
| Troubleshooting for Administrators | 1.2.28 |
| Module Review | 1.2.29 |
| Q+A Answers | 1.2.30 |
| Self-Serve with FME Server: Part I | 1.3 |
| What is Self-Serve? | 1.3.1 |
| FME Server Services | 1.3.2 |
| Implementing Self Serve | 1.3.3 |
| Exercise: Data Download System: Workspace Creation | 1.3.4 |
| Self-Serve and Parameters | 1.3.5 |
| Published Parameters | 1.3.6 |
| Key Parameter Types | 1.3.7 |
| Exercise: Data Download System: Published Parameters | 1.3.8 |
| Format and Coordinate System Selection | 1.3.9 |
| Exercise: Data Download System: Format and Coordinate Systems | 1.3.10 |
| Layer Selection and Handling | 1.3.11 |
| Exercise: Data Download System: Layer Selection | 1.3.12 |
| Troubleshooting for Administrators | 1.3.13 |
| Module Review | 1.3.14 |
| Q+A Answers | 1.3.15 |
| Self-Serve with FME Server: Part II | 1.4 |
| Geographic Selection | 1.4.1 |
| Geographic Selection by Bounding Box | 1.4.1.1 |
| Geographic Selection by Existing Area | 1.4.1.2 |
| Geographic Selection by Ad Hoc Area | 1.4.1.3 |
| Exercise: Data Download System: Geographic Selection | 1.4.2 |
| Other Self-Serve Services | 1.4.3 |
| Exercise: Data Streaming System | 1.4.4 |
| Troubleshooting for Administrators | 1.4.5 |
| Module Review | 1.4.6 |
| Q+A Answers | 1.4.7 |
| Real-Time with FME Server | 1.5 |
| The Notification Service | 1.5.1 |

| | |
|--|----------|
| Elements of the Notification Service | 1.5.2 |
| Notification Clients | 1.5.3 |
| Publications and Subscriptions | 1.5.4 |
| Notification Topics | 1.5.5 |
| Notification Protocols | 1.5.6 |
| Protocol Details | 1.5.6.1 |
| Exercise: Building Updates Notification Systems: Directory Watch | 1.5.7 |
| Notifications and Workspaces | 1.5.8 |
| Workspaces as Subscribers | 1.5.9 |
| Exercise: Building Updates Notification System: Workspace Subscription | 1.5.10 |
| Incoming Message Handling | 1.5.10.1 |
| Exercise: Building Updates Notification System: Incoming Messages | 1.5.11 |
| Email Notifications | 1.5.12 |
| Email Publications: SMTP | 1.5.12.1 |
| Email Publications: IMAP | 1.5.12.2 |
| Email Subscriptions | 1.5.12.3 |
| Exercise: Building Updates Notification System: Email Publication | 1.5.13 |
| Workspace as Publishers | 1.5.14 |
| Registering Workspaces as Publishers | 1.5.14.1 |
| Workspace Publishers using Transformers | 1.5.14.2 |
| Outgoing Message Handling | 1.5.14.3 |
| Workspace as Both Subscription and Publication | 1.5.15 |
| Exercise: Building Updates Notification System: Outgoing Messages | 1.5.16 |
| Message Streams | 1.5.17 |
| Elements of a Message Stream | 1.5.17.1 |
| Message Streaming Architecture | 1.5.17.2 |
| Exercise: Handling Emergency Phone Call Streams | 1.5.17.3 |
| Troubleshooting for Administrators | 1.5.18 |
| Module Review | 1.5.19 |
| Q+A Answers | 1.5.20 |
| FME Server Projects | 1.6 |
| Creating Projects | 1.6.1 |
| Sharing a Project | 1.6.2 |

| | |
|-------------------------------------|-------|
| Exporting Projects | 1.6.3 |
| Removing Projects | 1.6.4 |
| Importing Projects | 1.6.5 |
| Exercise: FME Server Projects | 1.6.6 |
| Troubleshooting for Administrators | 1.6.7 |
| Module Review | 1.6.8 |
| Q+A Answers | 1.6.9 |
| Course Wrap-Up | 1.7 |
| Product Information and Resources | 1.7.1 |
| Community Information and Resources | 1.7.2 |
| Course Feedback | 1.7.3 |
| Thank You | 1.7.4 |
| Exercise: A Fun Challenge! | 1.7.5 |

Workspace Authoring for FME Server Training Manual

This is the manual for the training course Workspace Authoring for FME Server.



This training provides a framework for authoring workspaces for FME Server. We hope that you will learn all the tools upon which to base your work, and go home with many new FME ideas!

Course Structure

The full course is made up of five main sections. These sections are:

- Introduction to FME Server
- Self-Serve with FME Server: Part1
- Self-Serve with FME Server: Part2
- Real-Time with FME Server
- FME Server Projects

Current Status

The current status of this manual is: **COMPLETE**. This manual **CAN** be used for training.

This manual applies to **FME2017.0** and **FME2017.1**

The status of each chapter is:

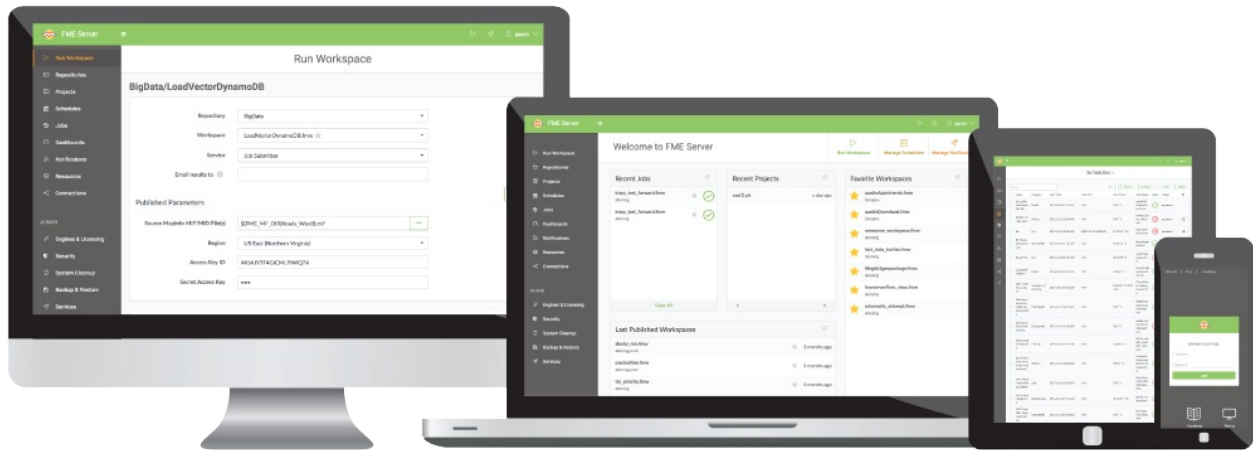
- Chapter 0: Content complete. No exercises.

- Chapter 1: Content and exercises complete.
- Chapter 2: Content and exercises complete.
- Chapter 3: Content and exercises complete.
- Chapter 4: Content and exercises complete.
- Chapter 5: Content and exercises complete.
- Chapter 6: Content complete. No exercises.

NB: *Even for completed content, Safe Software Inc. assumes no responsibility for any errors in this document or their consequences, and reserves the right to make improvements and changes to this document without notice. See the full licensing agreement for further details.*

About This Document

This manual is the introductory-level training course for authoring translations for FME Server.



Look out for residents of the City of Interopolis, who will appear from time-to-time to give you advice and dispense FME-related wisdom. In fact, here comes someone now:

Mr. E.Dict (Attorney of FME Law) says...

On behalf of the City of Interopolis, welcome to this training course. Here is the standard legal information about this training document and the datasets used during the course.

Be sure to read it, particularly if you're thinking about re-using or modifying this content.

Licensing and Warranty

Permission is hereby granted to use, modify and distribute the FME Tutorials and related data and documentation (collectively, the “Tutorials”), subject to the following restrictions:

1. The origin of the Tutorials and any associated FME® software must not be misrepresented.
2. Redistributions in original or modified form must include Safe Software’s copyright notice and any applicable Data Source(s) notices.
3. You may not suggest that any modified version of the Tutorials is endorsed or approved by Safe Software Inc.

4. Redistributions in original or modified form must include a disclaimer similar to that below which: (a) states that the Tutorials are provided “as-is”; (b) disclaims any warranties; and (c) waives any liability claims.

Safe Software Inc. makes no warranty either expressed or implied, including, but not limited to, any implied warranties of merchantability, non-infringement, or fitness for a particular purpose regarding these Tutorials, and makes such Tutorials available solely on an “as-is” basis. In no event shall Safe Software Inc. be liable to anyone for direct, indirect, special, collateral, incidental, or consequential damages in connection with or arising out of the use, modification or distribution of these Tutorials.

This manual describes the functionality and use of the software at the time of publication. The software described herein, and the descriptions themselves, are subject to change without notice.

Data Sources

City of Vancouver

Unless otherwise stated, the data used here originates from open data made available by the [City of Vancouver](#), British Columbia. It contains information licensed under the Open Government License - Vancouver.

Others

Forward Sortation Areas: Statistics Canada, 2011 Census Digital Boundary Files, 2013. Reproduced and distributed on an "as is" basis with the permission of Statistics Canada. © This data includes information copied with permission from Canada Post Corporation.

Digital Elevation Model: GeoBase®

Fire Hall Data: Some attribute data adapted from content © 2013 by [Wikipedia](#), used under a Creative Commons Attribution-ShareAlike license

Stanley Park GPS Trail: Used with kind permission of [VancouverTrails.com](#).

Copyright

© 2005–2017 Safe Software Inc. All rights are reserved.

Revisions

Every effort has been made to ensure the accuracy of this document. Safe Software Inc. regrets any errors and omissions that may occur and would appreciate being informed of any errors found. Safe Software Inc. will correct any such errors and omissions in a subsequent version, as feasible. Please contact us at:

Safe Software Inc.

Phone: 604-501-9985

Fax: 604-501-9965

Email: train@safe.com

Web: www.safe.com

Safe Software Inc. assumes no responsibility for any errors in this document or their consequences, and reserves the right to make improvements and changes to this document without notice.

Trademarks

FME® is a registered trademark of Safe Software Inc. All brand or product names are trademarks or registered trademarks of their respective companies or organizations.

Document Information

Document Name: FME Server Authoring Training Manual 2017

All screenshots relate to FME Desktop and FME Server 2017.0; This manual has been tested with FME Desktop and FME Server 2017.1 Build 17536.

Course Overview

This training material covers how to create FME translation models for use on FME Server.

The training will introduce basic concepts and terminology, help you become an efficient FME Server user, and direct you to resources to help apply the product to your own needs.

Course Structure

The full course is made up of five main sections. These sections are:

- Introduction to FME Server and Running Workspaces
- Self-Serve with FME Server - Part I
- Self-Serve with FME Server - Part II
- Real-Time and Automation with FME Server
- FME Server Projects

The instructor may choose to cover as many of these sections as they feel are required, or possible in the time permitted. They may also cover the course content in a different order and will skip or add new content to better customize the course to your needs.

Therefore the length and content of the course may vary, particularly when delivered online.

Prerequisites

This training material is intended for persons with some prior experience of using FME. It assumes a basic familiarity with the concepts and practices covered of FME Desktop; at the very least to the extent covered by the FME Desktop Tutorial.

In particular it would be helpful to be familiar with:

- Parameters and published parameters
 - Managing Readers, Writers and feature types in a workspace
 - Using transformers similar in complexity to the Clipper and Joiner
-

Ms Analyst says...

A further prerequisite for carrying out some exercises in the Real Time section is an email account that can be accessed through IMAP, for example a Google Gmail account. If you don't have one of these, please set one up before getting to that section of the manual.

About the Manual

The FME Server authoring training manual not only forms the basis for FME Server training – in-person or online – but is also useful reference material for future work you may undertake with FME.

Each chapter has a short appendix with troubleshooting suggestions; general hints to follow in case FME Server causes problems for you.

This training material is designed specifically for use with FME2017. You may not have some of the functionality described if you use an older version of FME.

Course Resources

A number of sample datasets and workspaces will be used in this course.

On Your Training Computer

The following applications may already be installed, licensed, and located on your training computer (real or virtual):

- Java Virtual Machine
- Apache Tomcat
- FME Desktop Version 2017
- FME Server Version 2017
- PostgreSQL / PostGIS

The data used in this training course is based on open data from the City of Vancouver, Canada.

Most exercises ask you to assume the role of a city planner at the fictional city of Interopolis and to solve a particular problem using this data.

Whether it's a local computer or a virtual computer hosted in the cloud, you'll find resources for the examples and exercises in the manual at the following locations:

| Location | Resource |
|---------------------------|--|
| C:\FMEData2017\Data | Datasets used by the City of Interopolis |
| C:\FMEData2017\Resources | Other resources used in the training |
| C:\FMEData2017\Workspaces | Workspaces used in the student exercises |
| C:\FMEData2017\Output | The location in which to write exercise output |
| \FME\Workspaces | The default location to save FME workspaces |

You should also find a digital copy of this manual.

Please alert your instructor if any item is missing from your setup.

You can find the latest version of FME Desktop and FME Server for Windows, Mac, and Linux - together with the latest Beta versions - on the [Safe Software web site](#).

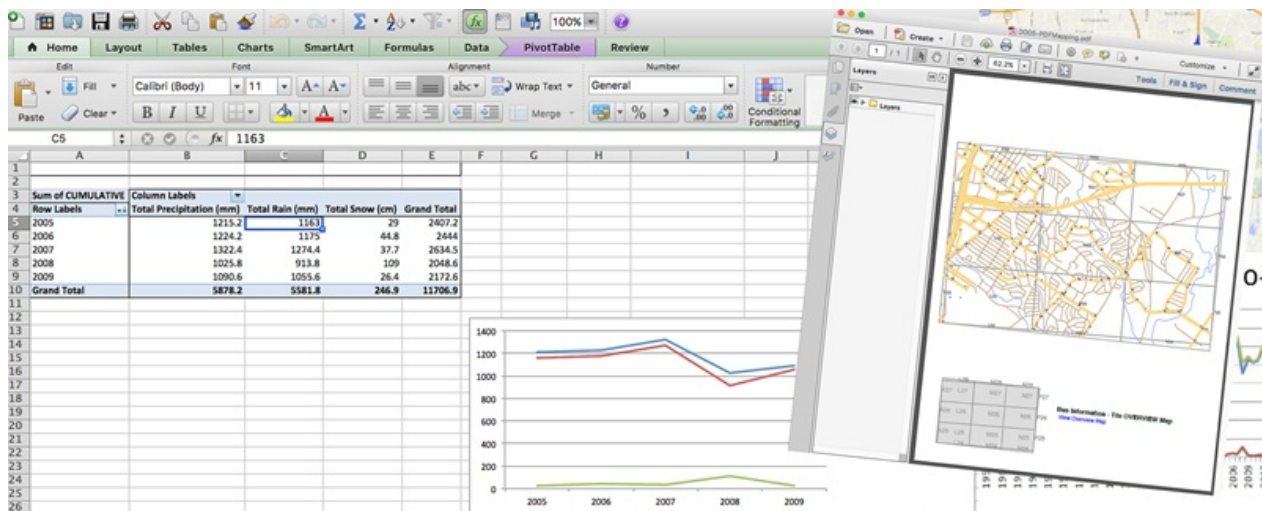
Course Etiquette

For online courses, please consider other students and test your virtual machine connection *before* the course starts. The instructor cannot help debug connection problems during the course!

For live courses, please respect other students' needs by keeping noise to a minimum when using a mobile phone or checking e-mail.

Introduction to FME Server

FME Server is a powerful product that automates the flow of data between application; not to mention any mindless data tasks you're forced to do each day!



This chapter introduces you to FME Server and explains how to run a workspace upon it.

What is FME Server?

FME Server is a powerful product for handling large volumes of data at the enterprise level. It has three core capabilities: Self-Serve, Real-Time, and Automation!

Self-Serve

Self-Serve is the ability for the end-user to select and download the data they require, in the format and structure they require; or to upload data for processing. It eliminates the need for a data manager/analyst to carry out manual data distribution tasks.

Real-Time

Real-Time is the ability to react to real-time events and sensors, to carry out immediate updates, and to deliver instant notifications. It allows subscribers to have the most up-to-date information for their business decision making.

Automation

Automation is the ability to carry out data processing at a specific schedule, and to spontaneously move data through different systems and web services – even onto mobile platforms and devices. It allows data to move from anywhere, to anywhere, without manual intervention.



FME Server vs FME Desktop

FME Server expands on the local processing tools of FME Desktop by providing the same core data translations and transformations, but at an enterprise level, and with the above enterprise-level capabilities.

With these capabilities, FME Server can take advantage of a huge variety of different communication technologies. This means it can provide many different ways to:

- Start a workspace
- Structure a translation
- Deliver the output

FME Server does not include an authoring tool, so FME Desktop is used to create these workspaces.

The focus of this course is how to create these workspaces in ways that will take full advantage of FME Server's enterprise capabilities.

FME Server Roles

When using a product with the range and scope of FME Server, it's no surprise that there are a number of different roles available to different users. Each different role usually has different interests in FME Server.

Author

An author is someone who creates translations (workspaces) using FME Desktop and publishes them to FME Server for use by end-users. This particular role is the focus of these training materials.

Typically an author would work at the analyst level, and would be an experienced FME Desktop user with a good understanding of Readers, Writers, transformers, and other FME Workbench functionality.

User

A user is defined as a person who accesses data using an FME Server Service.

It is not expected that a user in this sense has, or needs to have, any experience of FME and does not even need to be aware of FME Desktop or FME Server.

The user might be someone like a CAD operator checking out spatial data, a business manager looking at corporate data in an FME Server client, or members of the public who want to download data for personal use.

Developer

FME Server can be used as the back-end to power other software applications.

A developer (in FME Server terms) is someone who creates applications that submit jobs to FME Server and then handle the results. This role is somewhat covered by these training materials.

Web developers may choose to include FME Server Web Services within their own web applications or create their own services using the FME Server API.

Administrator

An FME Server administrator is the person responsible for installing and maintaining FME Server and its related services.

The administrator's tasks include:

- Planning system architecture
- Installing prerequisite applications
- Installing and licensing FME Server
- Setting up FME Server services
- Monitoring FME Server services
- Troubleshooting
- Scaling FME Server

Miss Vector says...

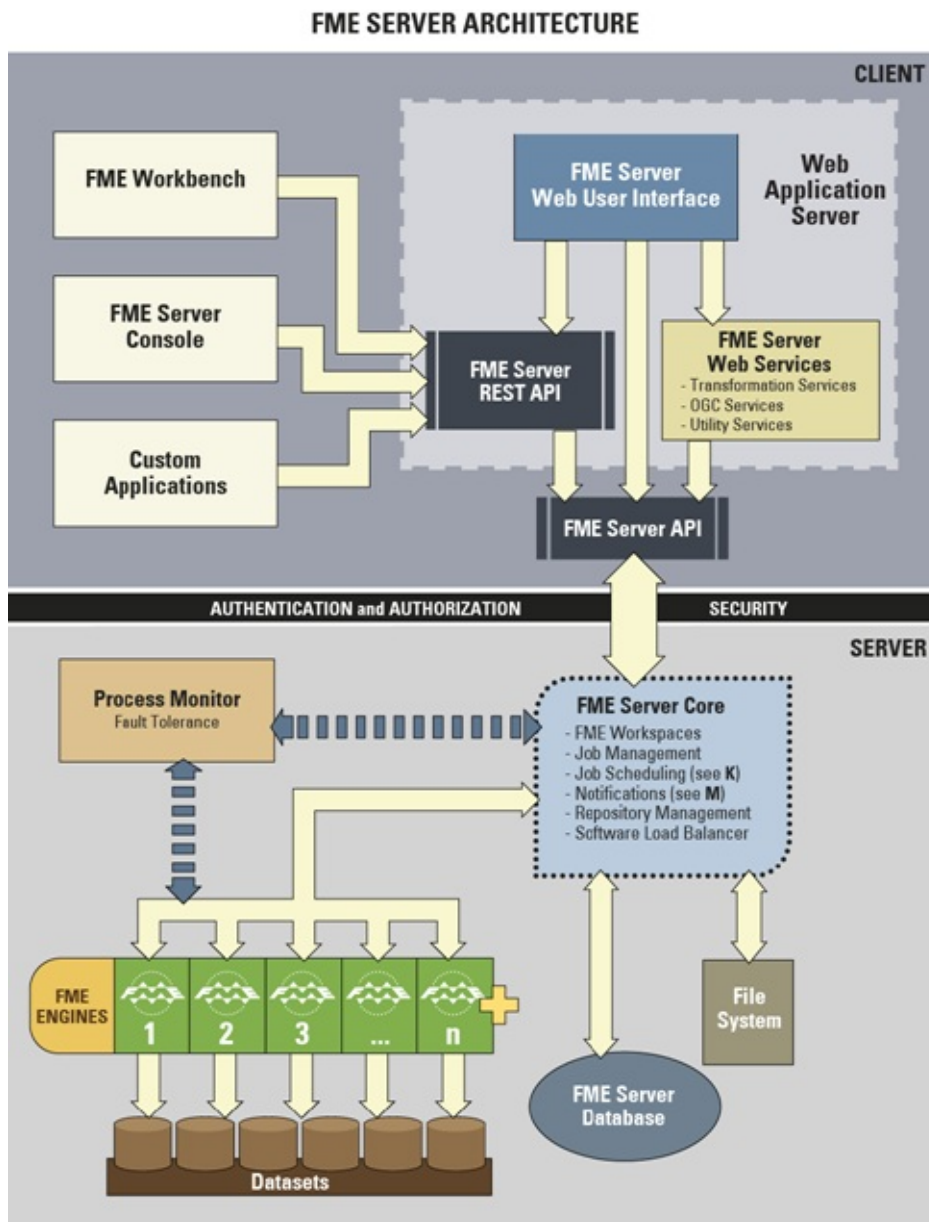
*Hi, I'm Miss Vector, FME schoolteacher. I'll be here now and then to test you on your new FME Server knowledge. For now, answer me this: Which of these is **not** one of the three core capabilities of FME Server?*

1. Automation
2. Real-Time
3. NoSQL Database
4. Self-Serve

FME Server Architecture

FME Server consists of a number of different components.

The architecture of FME Server is a client-server model that looks something like this:



FME Server Components

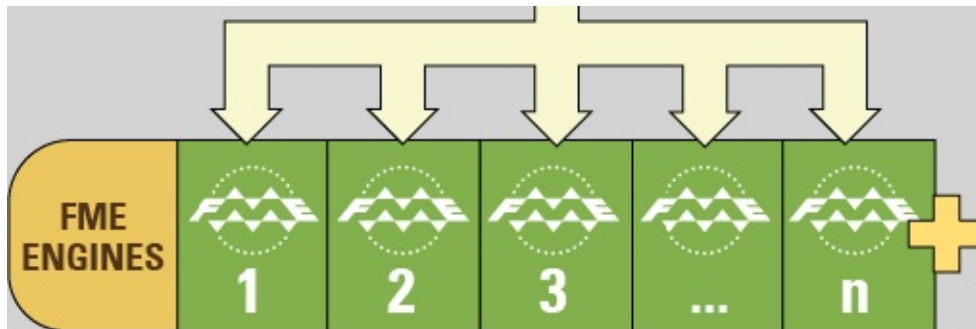
Don't worry about trying to understand the whole structure, just be aware of the main components of FME Server:

- **FME Engines:** To carry out data transformation processing

- **Server Core:** To queue jobs, handle scheduling, and manage load balancing
- **Web Services:** To handle networking capabilities

FME Engines

FME Engines process job requests by running FME Workspaces. This is the same core engine, carrying out the same processing, that is used by FME Desktop. An FME Server installation can possess multiple engines.

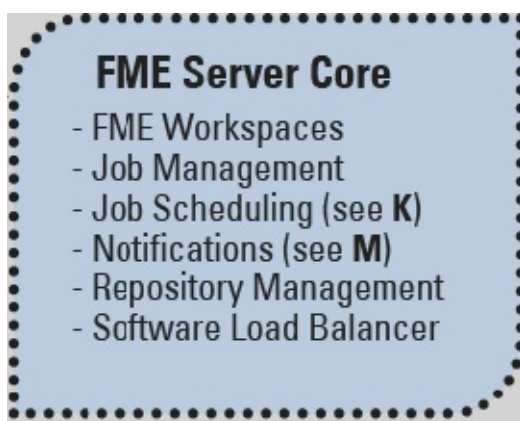


Each FME Engine processes a single request at a time.

FME Server processing can be scaled by adding FME Engines to the same computer or to separate computers within a distributed FME Server environment.

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.



The FME Server Core contains a Software Load Balancer (SLB) that distributes jobs to FME Engines.

Web Services

Much of the FME Server networking capabilities are handled using what we call "Web Services". These Web Services are software whose interface provides communication between server and clients.

FME Server has a number of services:

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

Some services (for example, Data Download) are "transformation" services that carry out data transformation, whereas others (for example, Catalog) are non-transforming "utility" services.



Miss Vector says...

Here's a question to see if you grasp, not the details, but how FME Server works in general. I have an FME Server with two engines. Four users submit jobs at the same time. What happens?

- 1. Two jobs are processed. Two jobs are returned to their authors.*
- 2. Two extra engines will fire up automatically to process all four jobs.*
- 3. The four jobs will be processed simultaneously, sharing the two engines.*
- 4. Two jobs are processed. The other two sit in a queue until an engine becomes free.*

Deploying FME Server

FME Server can be deployed in a number of configurations.

Platforms

FME Server can be deployed on a variety of platforms:

Local Infrastructure (Physical Hardware)

This is the traditional platform where you purchase FME Server and install it on your own hardware systems.

Infrastructure as a Service (Virtual Hardware)

This is where you purchase FME Server and install it on virtual hardware that is provided as service by a company such as Amazon.

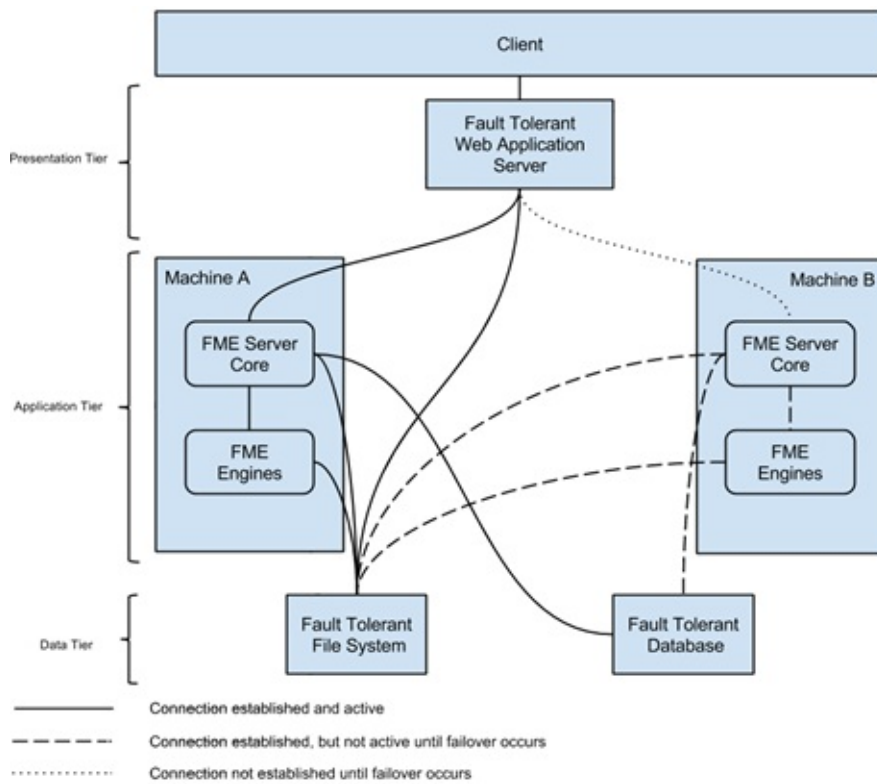
Platform as a Service (FME Cloud)

This is where FME Server is delivered pre-installed on an Amazon virtual computer, with the whole platform provided by Safe Software on a pay-as-you-go basis.

Distributed Systems

A distributed system is when different components of a system are located on separate, networked, computers.

For example, in a distributed environment FME Engines can 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 licensing does not use a separate license server. Instead licensing is stored in the FME Server System Share, which multiple FME Server Cores can point towards.

FME Workspaces and FME Server

FME Server is a model-driven architecture, because its processes are expressed as a model. In FME, these models are better known as **workspaces**.

Workspaces are created – we call it “authored” – using FME Desktop. In particular, the **FME Workbench** application is used. FME Workbench is a client of FME Server, and so they form a client-server pair. However, both share the same core engine and process data the same way.

Police-Chief Webb-Mapp says...

Hello, I'm the police chief and responsible for guiding you through this chapter.

Let's make sure you get the terminology right. The application itself is called FME “Workbench”, but the process defined in the canvas window is called a “Workspace”. The terms are so similar that they are easily confused, but please don't, otherwise I will have to send my grammar squad to arrest you!

Because Workbench is a client of FME Server, it may be used to transfer authored workspaces to and from an FME Server. We call this transfer **publishing**.

FME Workbench has the ability to:

- Author a translation workspace
- Publish a workspace (transfer it to FME Server)
- Republish a workspace (upload a previously published workspace)
- Download a workspace (retrieve it from FME Server)

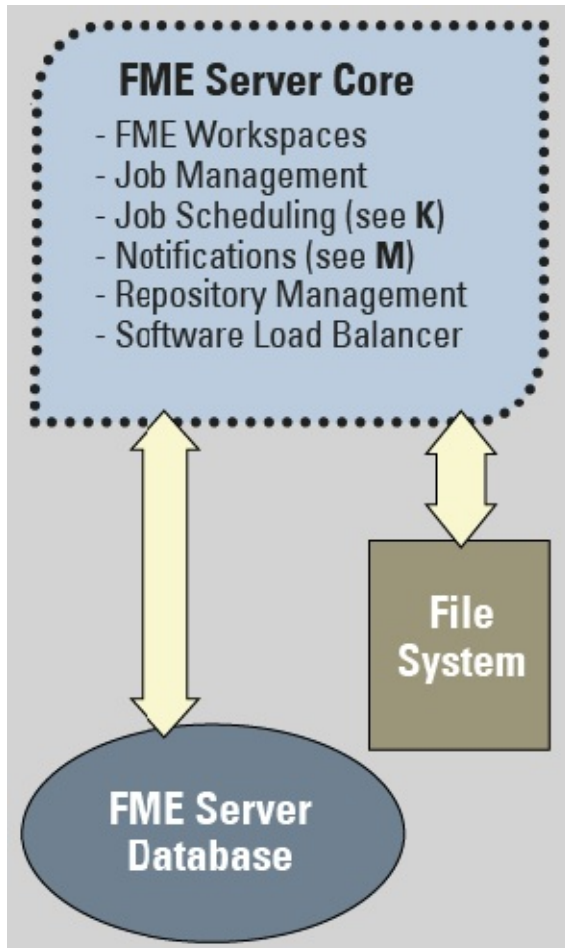
The ability to transfer a workspace back to FME Workbench means workspaces can be downloaded for editing and maintenance, then published back to FME Server.

Repositories

Workspaces are stored on FME Server in devices called **repositories**. Each FME Server may have multiple repositories, but any workspace can only belong to one of them.

A repository consists of two parts. Workspaces published to FME Server are held on a file-based part of the repository.

Metadata related to the workspace is held separately in a repository database. This metadata includes information about the contents of the workspace; for example source and destination datasets, workspace feature types, and published parameters.



Repositories are managed by the FME Core. They can be accessed (by authors and administrators) through the FME Server web interface.

Police Chief Webb-Mapp says...

Security in FME Server is very important, and never more so than for repositories.

You can think of each repository as being like a folder on a file-system, with the same ability to grant access rights to individuals and groups. So, for each repository you create, be sure to check the security settings. If you don't then end-users may not get access to the repository!

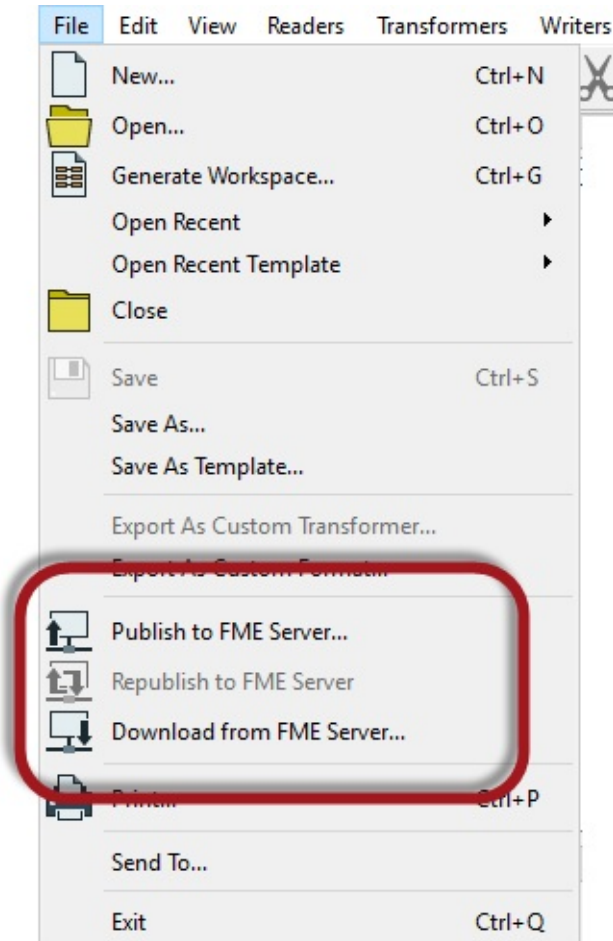
Miss Vector says...

If I wanted to find out about workspaces stored in a repository - for example I'm building a tool to catalogue my workspaces - what is the best way to do it?

- 1. Use the FME Server REST API*
- 2. Scrape the contents of the Server repository page*
- 3. Get a file listing from the repository folder*
- 4. Connect to the FME Server database to query it directly*

Transferring Workspaces

The functionality for publishing or downloading workspaces is accessed in FME Workbench either through the menubar:

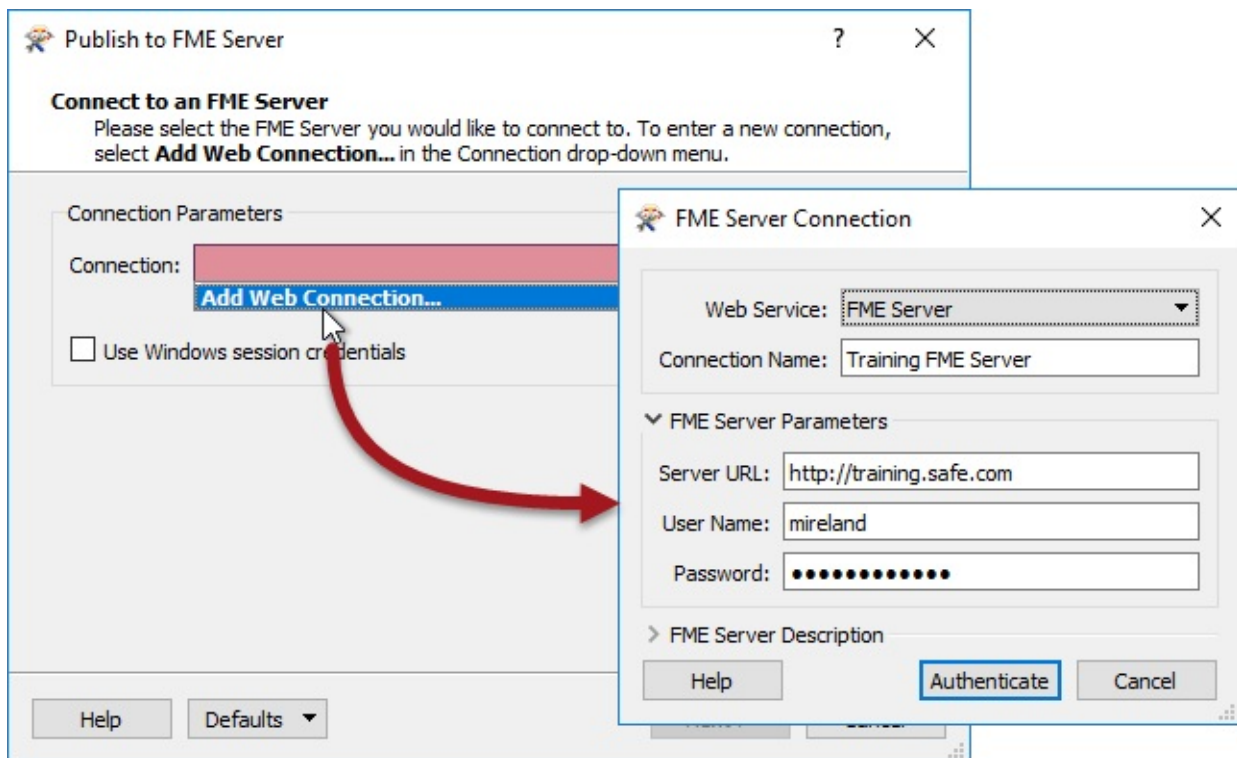


...or the toolbar:

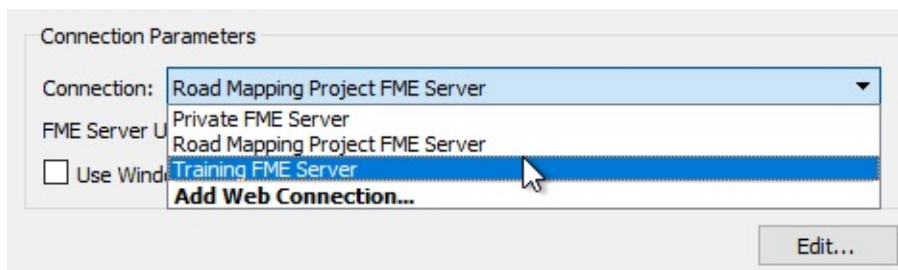


Connecting to Server

The publish tool in Workbench opens a simple wizard interface, the first dialog of which defines a connection to Server.



Adding a web connection opens a dialog with fields in which to define connection credentials. These connection details are saved so that they can be reused in the future simply by picking from the drop-down list:

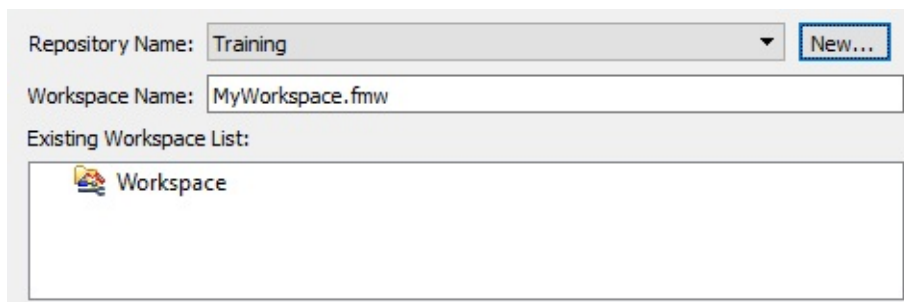


NEW

The ability to save and store multiple Server connections is new for FME2017. For older versions of FME you may save a set of parameters as the default, but one set only.

Repository Selection


The next dialog defines the repository for the workspace:



Repository Name: Training New...

Workspace Name: MyWorkspace.fmw

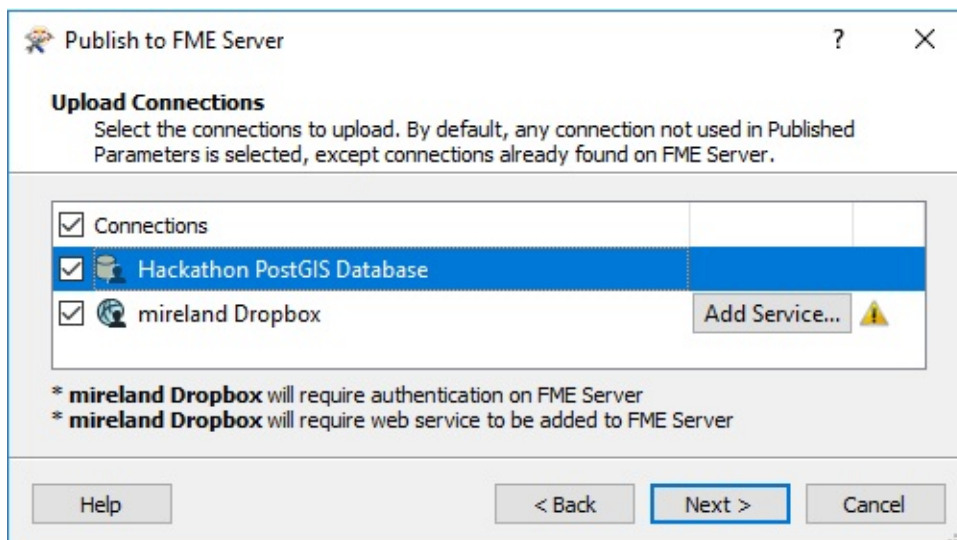
Existing Workspace List:

| |
|---|
|  Workspace |
|---|

Either an existing repository can be used, or a new one created. The workspace name can also be edited, even making it different to what it is saved as locally.




Connections Upload

This dialog only appears when there are database and/or web connections that need to be uploaded with the workspace.



Publish to FME Server

Upload Connections
Select the connections to upload. By default, any connection not used in Published Parameters is selected, except connections already found on FME Server.

| | | |
|-------------------------------------|--|---|
| <input checked="" type="checkbox"/> | Connections | |
| <input checked="" type="checkbox"/> |  Hackathon PostGIS Database | |
| <input checked="" type="checkbox"/> |  mireland Dropbox | Add Service...  |

* mireland Dropbox will require authentication on FME Server
* mireland Dropbox will require web service to be added to FME Server

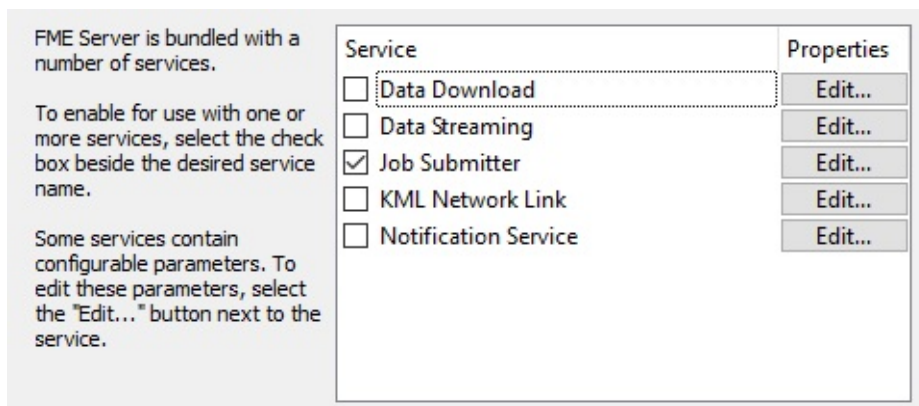
Help < Back Next > Cancel

This workspace contains both a database connection and a web connection that need to be uploaded to function on FME Server. Notice that the *OAuth* web connection needs the service to be added to FME Server, and will also require additional authentication before FME Server will be allowed to use it.

The database connection requires no further authentication, but care must be taken not to accidentally overwrite an existing database connection with the same name that might already be defined on FME Server.

Workspace Registration

The final dialog defines which services the workspace is to be registered against.



The Job Submitter service allows FME Server to run a workspace as-is. This is the closest to running a workspace on FME Desktop. All inputs and outputs are defined in the workspace so data is simply written out and not streamed or delivered in any other manner.

Job submission is ideal for testing workspaces, and for running large-scale and batch translations that make use of the server process queue.

Republishing a Workspace

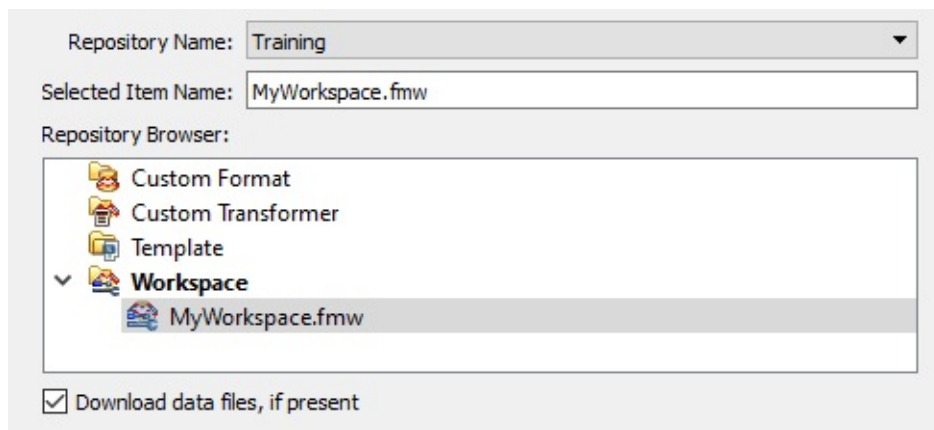
Once a workspace has been published, the republish tool becomes active. Further updates to the workspace (within the same session) can then be uploaded with a single click.

The same parameters are used as before. If changes need to be made to these parameters, then the full publishing wizard should be used.

Downloading a Workspace

Workbench can also “download” a workspace held in an FME Server repository. This is usually done in order to make edits to the workspace. Note that downloaded workspaces are copies of the original, which remains in the FME Server repository.

The downloading wizard begins with the same connection dialog as the publishing wizard. From there, the second – and final – dialog page is a repository and workspace selection tool:



The user is then prompted for a location to save the workspace. The default is <user>/FME/My FME Server Workspaces. The workspace – and resources – are then downloaded.

Once downloaded, the workspace is automatically opened within Workbench for editing.

Ms. Analyst says...

Besides workspaces, it's also possible to publish/download FME custom transformers and custom formats to/from a server repository.

| Exercise 1 Daily Database Updates: Publishing a Workspace | |
|--|---|
| Data | Firehalls (GML) Neighborhoods (KML) |
| Overall Goal | Create a workspace to read and process departmental data and publish it to FME Server |
| Demonstrates | Publishing a workspace to FME Server |
| Start Workspace | None |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\Basics-Ex1-Complete.fmw |

For the exercises in this chapter, you are a technical analyst in the GIS department of your local city. You have plenty of experience using FME Desktop, and your department is now investigating FME Server to evaluate its capabilities.

There are many departments within the city, and one of your tasks is to take the data from each department and merge it together into a single, corporate database.

Because each department produces their datasets in a different format and style, you use FME for this task, and carry it out on a weekly basis.

One of the reasons for purchasing FME Server is to automate this procedure, so let's start implementing that.

Police Chief Webb-Mapp says...

*If you have lots of experience with FME Workbench - **and if your instructor agrees** - simply open the workspace listed in the header above and skip to step 8.*

1) Inspect Source Data

For the sake of simplicity - and because this course is about Server, not Desktop - we'll just use a couple of datasets. These are:

| | |
|-----------------------|---|
| Reader Format | GML (Geography Markup Language) |
| Reader Dataset | C:\FMEData2017\Data\Emergency\FireHalls.gml |

| | |
|-----------------------|---|
| Reader Format | Google KML |
| Reader Dataset | C:\FMEData2017\Data\Boundaries\VancouverNeighborhoods.kml |

So start the FME Data Inspector by selecting it from the Windows start menu. Inspect all of the source data to become familiar with it. The VancouverNeighborhoods has a different coordinate system than the other dataset so be careful and turn on a background map if you want to view all the data together.

The goal of our translation is to convert the Firehalls and Neighborhoods to a database, dividing the firehalls data up into a separate table per neighborhood.

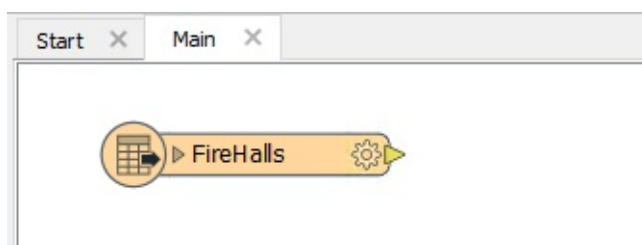
2) Start FME Workbench

Start FME Workbench by selecting it from the Windows start menu. Begin with an empty canvas by closing any existing workspace (if necessary) and clicking on the Main tab.

Now select Readers > Add Reader to start adding a reader to the workspace. When prompted, enter the following details for the Firehalls dataset:

| | |
|-----------------------|---|
| Reader Format | GML (Geography Markup Language) |
| Reader Dataset | C:\FMEData2017\Data\Emergency\FireHalls.gml |

Click OK to add the Reader to the workspace, which will now look like this:

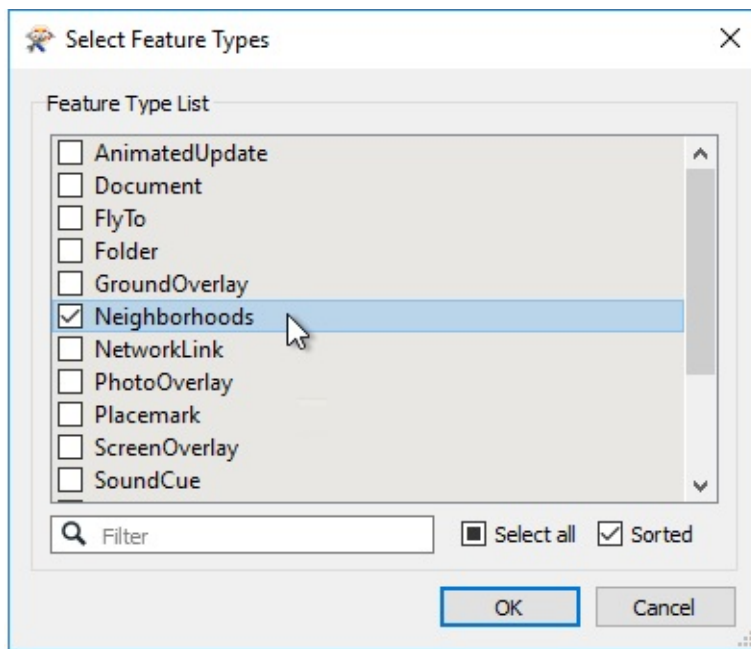


3) Add KML Data

Now repeat the process one more time to add a reader for the KML dataset:

| | |
|-----------------------|---|
| Reader Format | Google KML |
| Reader Dataset | C:\FMEData2017\Data\Boundaries\VancouverNeighborhoods.kml |

While adding the dataset you'll be prompted which feature types (layers) to add to the workspace. The only one we need is called Neighborhoods:

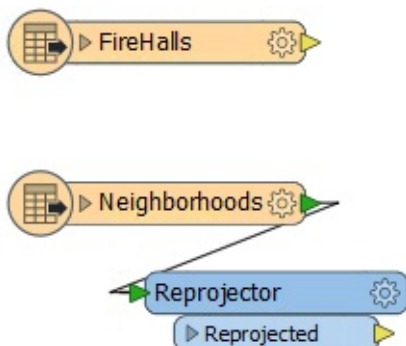


The workspace should now look like this:

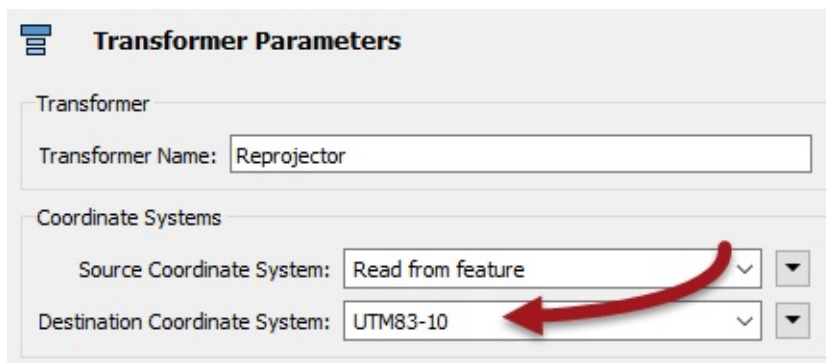


4) Add Reprojector Transformer

Add a Reprojector transformer to the workspace. You can do this by simply clicking on the canvas and starting to type Reprojector. Connect it to the Neighborhoods feature type:



Check the transformer's parameters and set the Destination Coordinate System to UTM83-10:



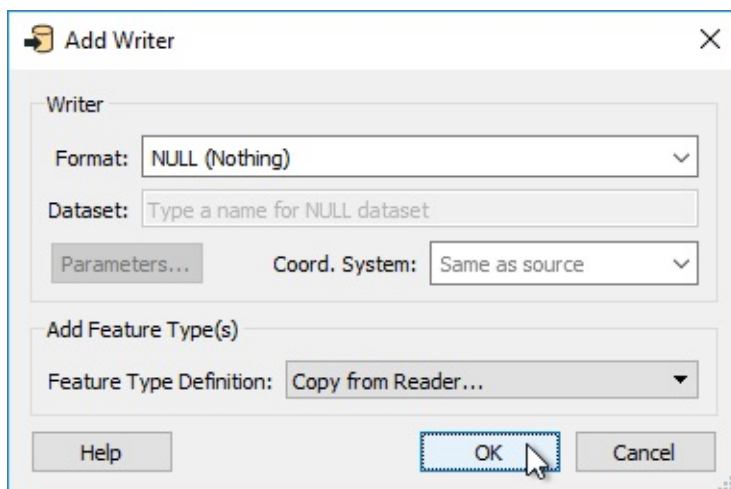
The image shows the 'Transformer Parameters' dialog box. Under the 'Transformer' section, 'Transformer Name' is set to 'Reprojector'. Under the 'Coordinate Systems' section, 'Source Coordinate System' is 'Read from feature' and 'Destination Coordinate System' is 'UTM83-10'. A red arrow points from the 'Read from feature' dropdown to the 'UTM83-10' dropdown.

This will ensure the neighborhoods data is in the same coordinate system as the rest of the data.

5) Add Writer

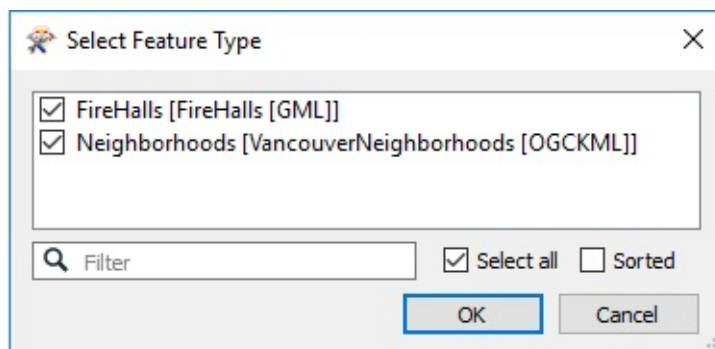
Now we should add a writer to the workspace. For now we'll just set up a dummy writer until we are more familiar with FME Server. So select Writers > Add Writer on the menubar to add a writer and set it up with the following parameters:

| | |
|--|---------------------|
| Writer Format | NULL (Nothing) |
| Feature Class or Table Definition | Copy from Reader... |



The image shows the 'Add Writer' dialog box. Under the 'Writer' section, 'Format' is 'NULL (Nothing)', 'Dataset' is 'Type a name for NULL dataset', 'Parameters...' is a button, and 'Coord. System' is 'Same as source'. Under the 'Add Feature Type(s)' section, 'Feature Type Definition' is 'Copy from Reader...'. At the bottom are 'Help', 'OK', and 'Cancel' buttons. A mouse cursor is pointing at the 'OK' button.

Click OK and OK again. When prompted, select both Firehalls and Neighborhoods as the feature types to add:



The image shows the 'Select Feature Type' dialog box. It has a list of feature types with checkboxes: 'FireHalls [FireHalls [GML]]' and 'Neighborhoods [VancouverNeighborhoods [OGCKML]]'. Both are checked. Below the list is a 'Filter' search box and two checkboxes: 'Select all' (checked) and 'Sorted'. At the bottom are 'OK' and 'Cancel' buttons.

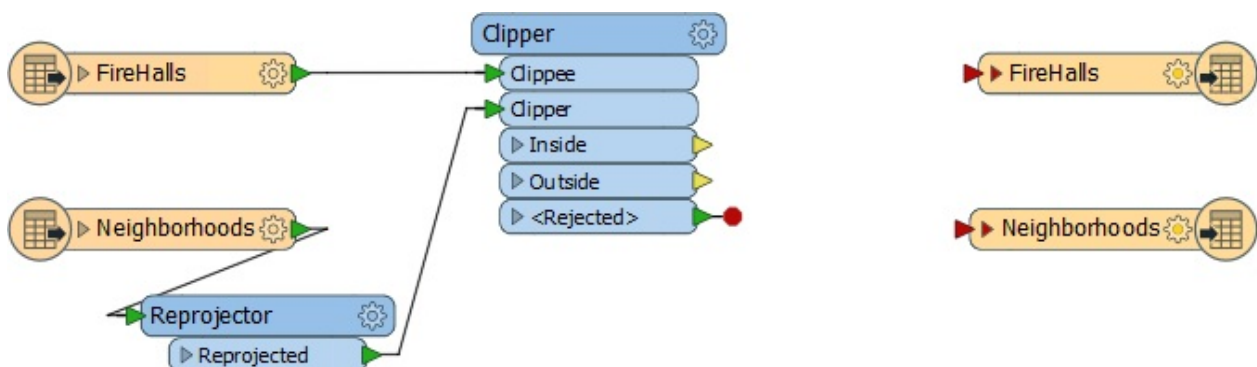
The workspace will now look like this:



6) Add Clipper Transformer

Add a Clipper transformer to the workspace. This will be used to divide the firehall data by neighborhood. Again, you can do this by simply clicking on the canvas and starting to type Clipper.

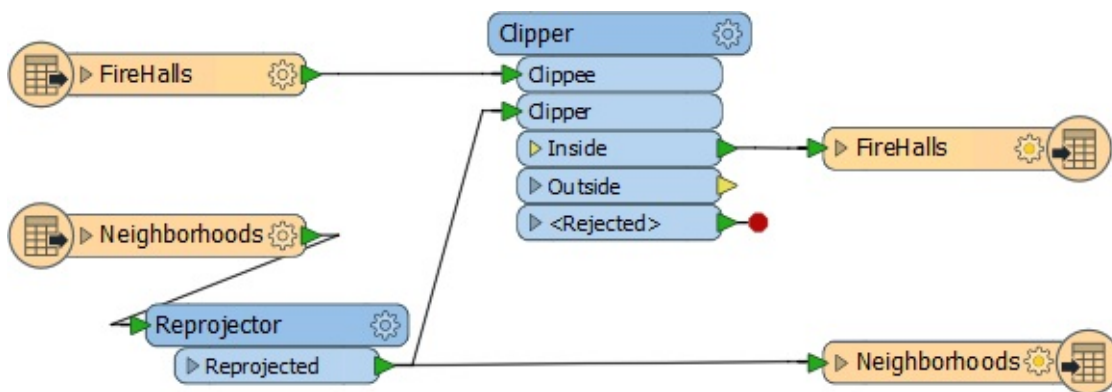
Connect the Firehalls feature type to the Clipper:Clippee port and the Reprojector:Reprojected output to the Clipper:Clipper port. You may wish to rearrange the feature types (or the port order) to avoid overlapping connections:



Check the parameters for the Clipper transformer to ensure the Clipper Type is set to Multiple Clippers. That's because there are multiple neighborhood features to act as a clipper feature.

Also put a check mark in the box labelled Merge Attributes, so that the neighborhood name is copied from the neighborhood features to the firehall features.

Connect the Clipper:Inside port to the Firehalls feature type on the writer. Also make a connection from the Reprojected:Reprojected port to the Neighborhoods feature type:



7) Set Firehall Feature Type Name

Finally, let's set the Feature Type Name for the Firehalls writer feature type.

Inspect its parameters and under Feature Type Name either enter:

```
FireHalls-@Value(NeighborhoodName)
```

...or click the dropdown and use the text editor dialog to enter that value. This will cause firehalls in each different neighborhood to be written to a different table/layer.

Save the workspace.

8) Run Workspace

Here comes the Server part of the process.

The first step, one which is very important, is to run the workspace. If the workspace won't run on FME Desktop then it is not likely to run on FME Server.

Run the workspace. Inspect the log. You should get six tables of firehalls and one of neighborhoods:

```

|=====
|                                     Features Written Summary
|=====
|FireHalls-Downtown                  2
|FireHalls-Fairview                  1
|FireHalls-Kitsilano                 1
|FireHalls-Mount Pleasant            1
|FireHalls-Strathcona                1
|FireHalls-West End                  2
|Neighborhoods                       6
|=====
|Total Features Written              14
|=====

```

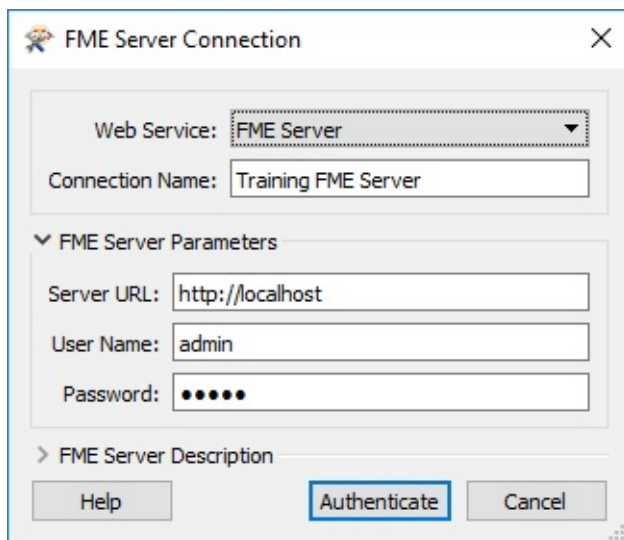
9) Publish to Server: Create Connection

Now we have a workspace and know that it works correctly, let's publish it to FME Server.

In FME Workbench, choose File > Publish to FME Server from the menubar. As this is the first time we've connected to our FME Server we'll need to create a new connection, so select Add Web Connection from the dropdown menu.

In the dialog that opens enter the parameters provided by your training instructor. In most cases the parameters will be as follows:

- **FME Server URL:** <http://localhost>
- **Username:** admin
- **Password:** admin



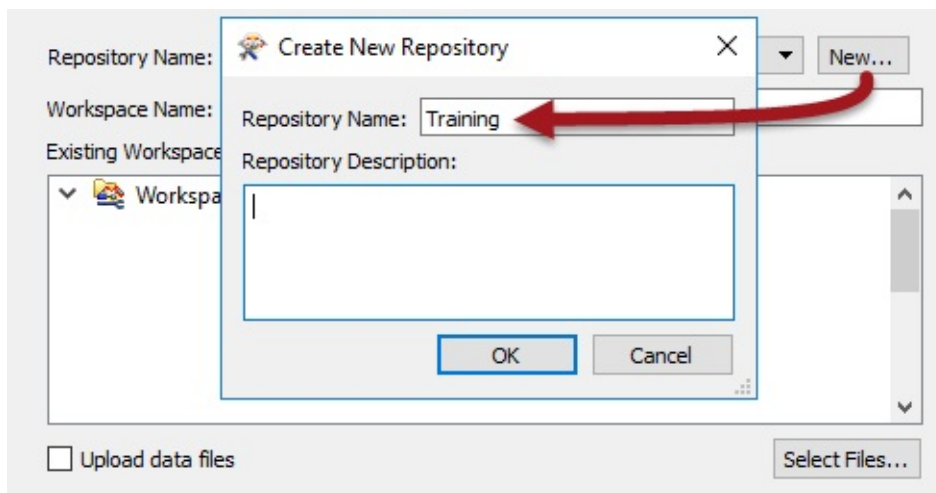
You may or may not (probably not) need to enter a port number with the hostname, depending on how the system is set up.

Click Authenticate to confirm the connection and return to the previous dialog. Make sure the newly defined connection is selected and click Next to continue.

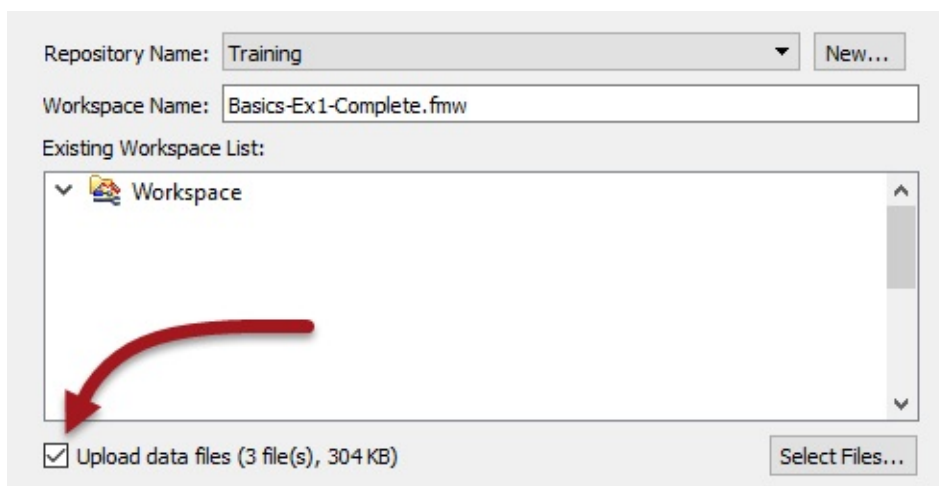
10) Publish to Server: Repository Selection

The next dialog prompts you to choose a repository in which to store the workspace.

For this exercise we'll create a new repository by clicking the New button. When prompted enter the name Training.



Click OK to close the Create New Repository dialog. Enter a name for the workspace if it doesn't already have one. Place a checkmark against the Upload Data Files option:

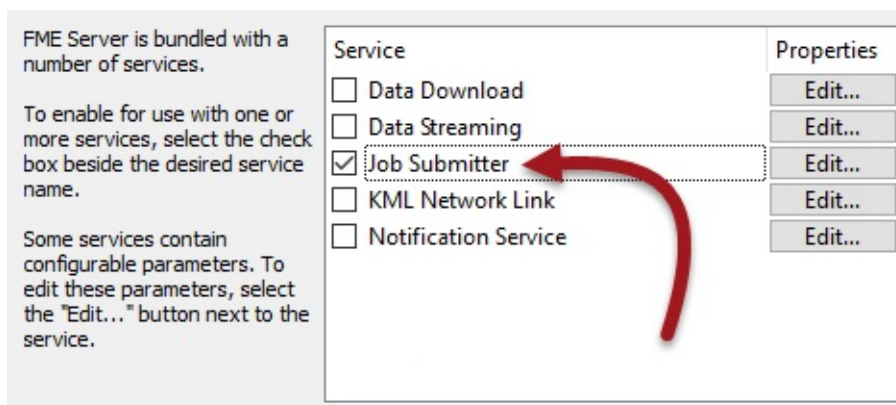


Then click Next to continue the wizard.

11) Publish to Server: Select Service

In the final screen of the wizard we can register the workspace for use with various services.

Select the Job Submitter service as this is the only service we are using for now:



... and click Publish to complete publishing the workspace.

After a workspace is transferred to Server, the log window displays a message reporting which workspace has been published to which repository and for which services. It will look something like this:

```
=====
                          Publish Summary
=====
FME Server URL       : http://training2017
Username             : admin
Repository           : Training
Name                 : Basics-Ex1-Complete.fmw
Direct Link          : http://training2017/fmeserver/#/workspaces/run/Training/Basics-Ex1-Complete.fmw
Uploaded Resources   : C:\FMEData2017\Data\Emergency\FireHalls.xsd
                     : C:\FMEData2017\Data\Boundaries\VancouverNeighborhoods.kml
                     : C:\FMEData2017\Data\Emergency\FireHalls.gml
Registered Services  : Job Submitter
Time                 : Mon Apr 24 10:06:25 2017
=====
```

CONGRATULATIONS

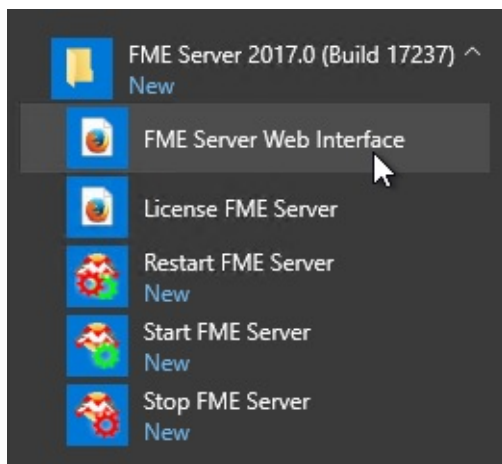
By completing this exercise you have learned how to:

- *Create a workspace converting data to a Null (dummy) format*
- *Use a Clipper to transfer attribute values from one feature to another*
- *Rename output layers according to the value of an attribute*
- *Publish a workspace to FME Server using the Publishing Wizard*
- *Create a repository on FME Server using the Publishing Wizard*
- *Register a workspace with the Job Submitter service using the Publishing Wizard*

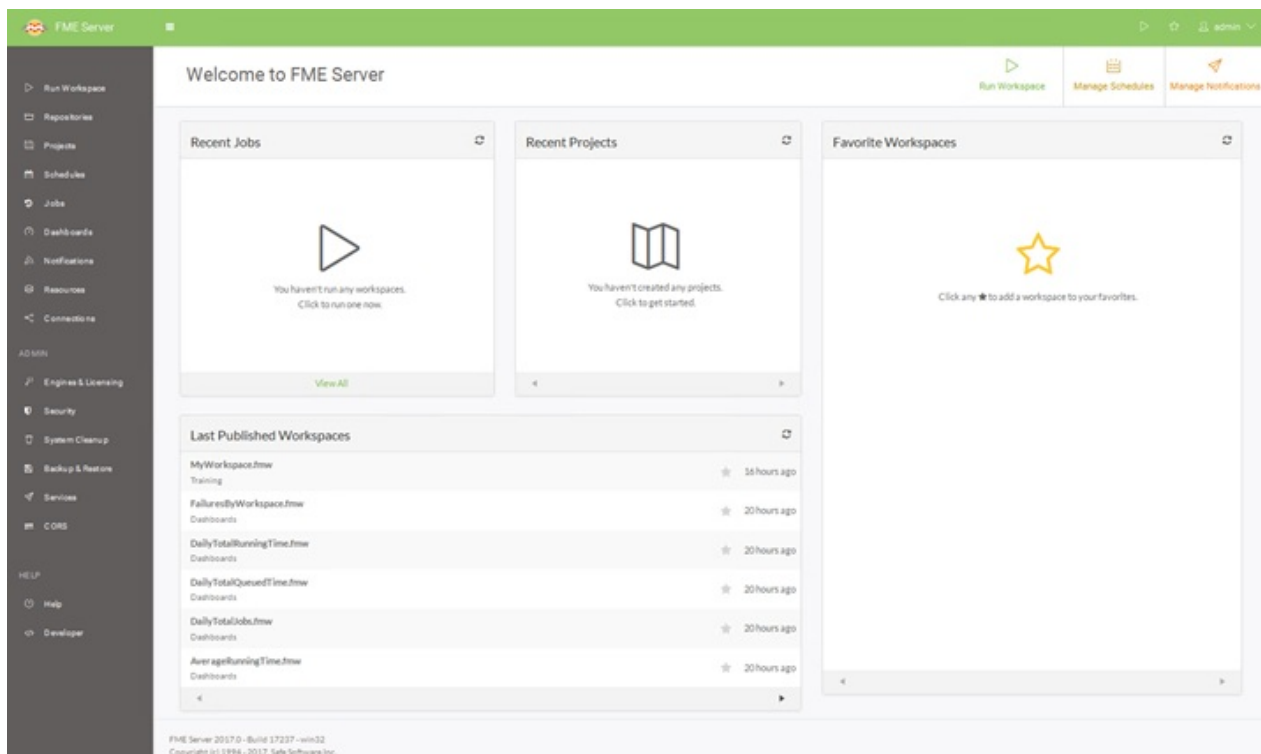
Introduction to the FME Server Web Interface

Although translations are authored in FME Workbench, the core tools of FME Server are accessed through a web-based interface.

The web interface is accessed through the URL **<servername>:<port>/fmeserver** (the port may be optional) or through the start menu:



The web interface for FME Server looks like this:



The main part of the interface displays a page of content that contains information, reports, parameters, and other components. The landing page, for example, has shortcuts to lists of recent jobs, recent projects, and favourite workspace.

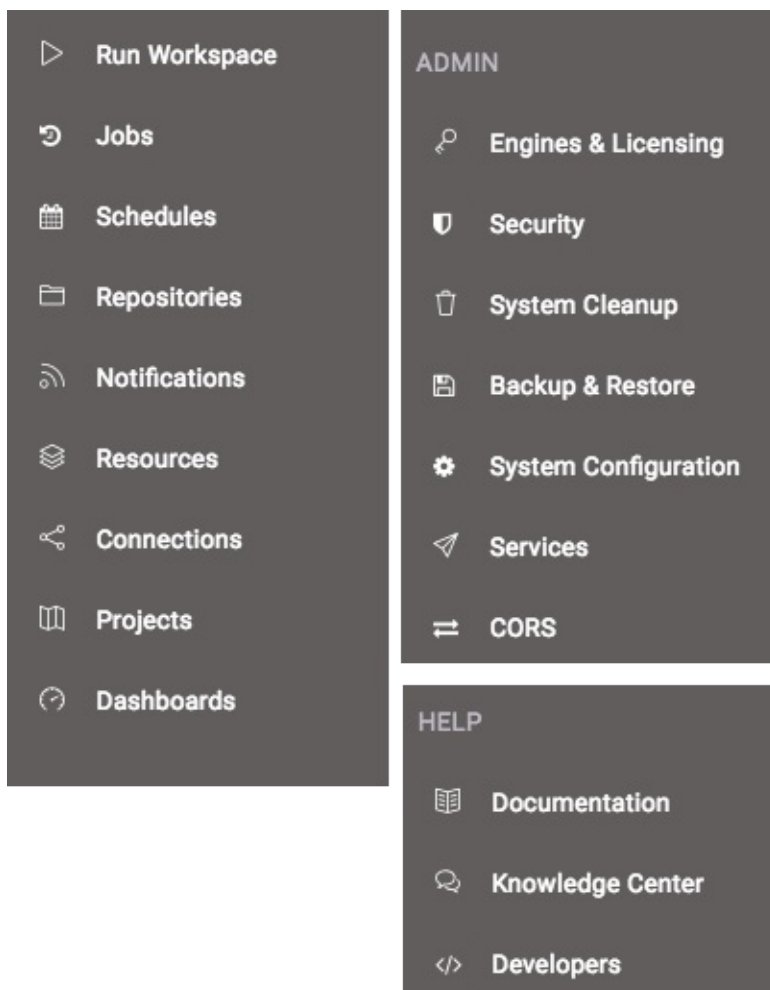
The left hand side of the interface contains a set of menus. Selecting a menu item changes the content of the page to match the menu item chosen.

NEW

The web interface was thoroughly redesigned for 2017, the biggest change perhaps being the menu moving from the top of the display to the left hand side.

Web Interface Menu

In general, FME Server functionality is accessed through the web interface menu. There are three sections of the menu:



The first section of the menu relates to the **use** of FME Server. It has - among others - options for running a workspace, accessing repositories, setting up schedules, and reviewing job history.

The next section of the menu relates to the **administration** of FME Server. It has - among others - options for managing engines, setting up security, and creating system backups.

The final section of the menu provides help tools; for authors, users, administrators, and developers.

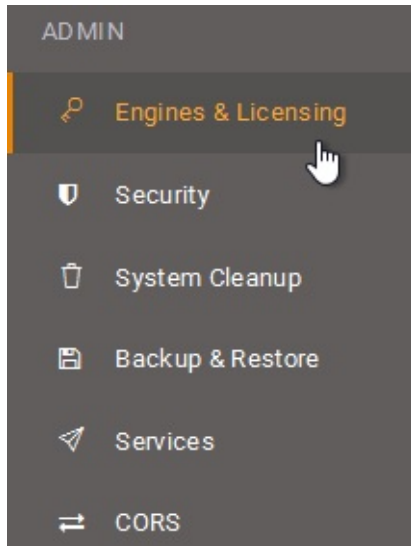
Getting Started

Getting started with the FME Server web interface requires familiarity with three key pages:

- Engines and Licensing
- Run Workspace
- Jobs

Engines and Licensing

The first step in getting started on FME Server is to choose the menu option for Engines and Licensing:



This opens the Engine Management page, where you can ensure FME Server is running correctly, is licensed, and has active engines:

Engine Management

FME Server is Licensed ✓

Licensing Documentation

[Refresh License](#) [Request License](#) [Upload License File](#)

MAX_ENGINES: 2 MACHINE_KEY: 2819312974 EXPIRES: Monday, February 6th 2017 SERIAL_NUMBER: No Serial

Engines

Engines will appear in the table below when ready. The engine list refreshes automatically.

Host: TRAINING2017

| Name | Build | Platform | Current Job |
|----------------------|-------|----------|-------------|
| TRAINING2017_Engine1 | 17237 | WIN32 | |
| TRAINING2017_Engine2 | 17237 | WIN32 | |

Total active engines: 2

Hosts 🟢

| Host | Engines |
|--------------|---|
| TRAINING2017 | 2 <input type="text"/> Change |

Total requested engines: 2

Licensing

The upper section of this page relates to licensing. The labels show how many engines are licensing, when the license expires, and what the machine key and serial number are. Buttons allow you to refresh the license, request a new one, or upload a license file.

FME Server is Licensed

Licensing Documentation

Refresh License

Request License

Upload License File

MAX_ENGINES: 2

MACHINE_KEY: 2819312974

EXPIRES: Monday, February 6th 2017

SERIAL_NUMBER: No Serial

Engines

The middle part shows the engines that are currently started, their FME build, operating system, and what job they are processing (if any). The platform is important because a distributed FME Server setup can have engines running on a variety of operating systems at the same time.

Engines

Engines will appear in the table below when ready. The engine list refreshes automatically.

Host: TRAINING2017

| Name | Build | Platform | Current Job |
|----------------------|-------|----------|-------------|
| TRAINING2017_Engine1 | 17237 | WIN32 | |
| TRAINING2017_Engine2 | 17237 | WIN32 | |

Total active engines: 2

Hosts

The lower part of the page shows which engines are running on which host, and allows you to easily change the number of engines running, up to the maximum provided for by the current license:

Hosts

Host

Engines

TRAINING2017

2

Change

Total requested engines: 2

If your FME Server is licensed, and has engines running that are assigned to the correct host, then you are ready to run a published workspace.

Job Queues

.1 UPDATE

Job Queues are new for FME Server 2017.1! This feature replaces the Job Routing configurations.

This section of the Engines & Licensing page, Job Queues, provides a way to reserve FME Engines for processing jobs from specific repositories. For example – you could have an FME Engine dedicated for processing quick tasks so that slower jobs will not cause a backlog. Similarly, you might reserve an FME Engine that sits on a more powerful machine for processing LiDAR data translations.

Job Queues ⓘ

[Create Queue](#)

Default - System Default Queue

Default

Repositories

Dashboards, Samples, steve, Utilities

Engines

AP-FME64BIT_Engine1, AP-FME64BIT_Engine2

LiDAR Processing

Repositories

No Repositories

Engines

AP-FME64BIT_Engine2

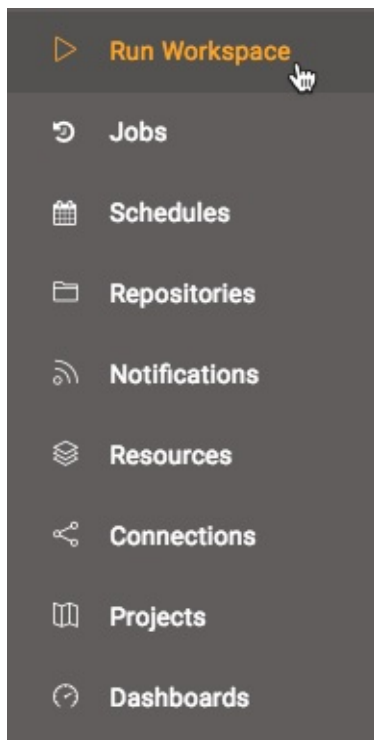
Miss Vector says...

Which of these are good reasons for running engines on multiple operating systems at the same time? Pick all that apply.

- 1. A required format is only available on 32-bit or 64-bit, not both.*
- 2. A required format is only available on Windows, or Linux, not both.*
- 3. FME Desktop users author workspaces using a mix of Windows, Linux, and Mac platforms.*
- 4. You want to process heavy-scale jobs on a more powerful platform.*

Run Workspace

The second step in getting started on FME Server is to choose the menu option for Run Workspace:



This opens the Run Workspace page, where you can pick a repository, workspace, and service, to run a translation:

Run Workspace

Training/Transformation-Ex5-Complete

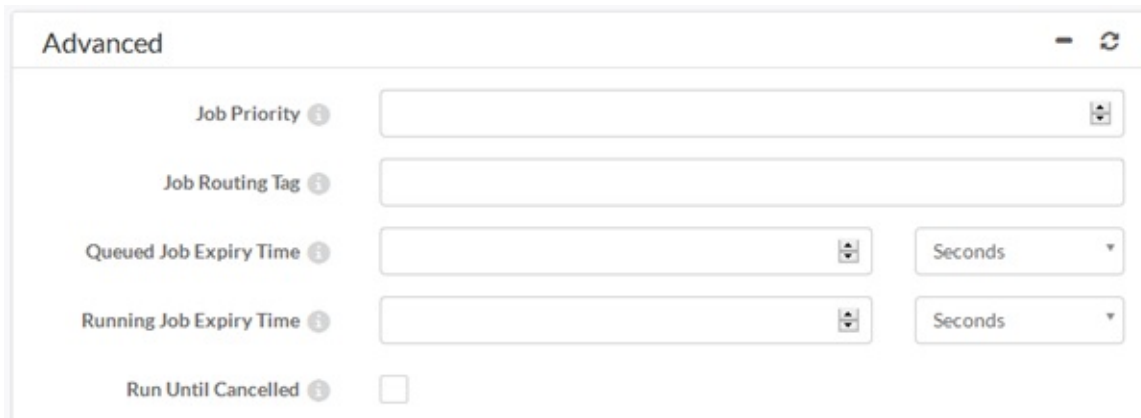
| | |
|--------------------|--|
| Repository | <input type="text" value="Training"/> |
| Workspace | <input type="text" value="Transformation-Ex5-Complete.fmw"/> ★ |
| Service | <input type="text" value="Job Submitter"/> |
| Email results to ⓘ | <input type="text"/> |

Published Parameters Reset

| | |
|------------------------------------|--|
| Source MapInfo TAB (MITAB) File(s) | <input type="text" value="C:\FMEData2017\Data\Parks\Parks.tab"/> ... |
| Destination MapInfo Folder | <input type="text" value="C:\FMEData2017\Output\Training"/> ... |

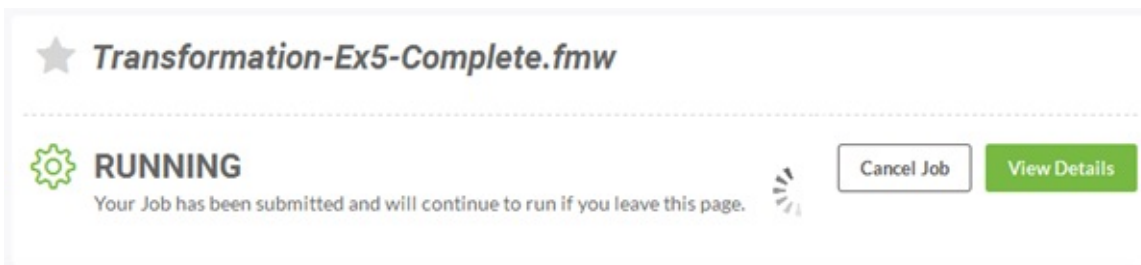
Having selected a workspace all of the published parameters are shown so that they can be set before the translation is run.

An advanced set of parameters gives control over the job that runs the workspace:



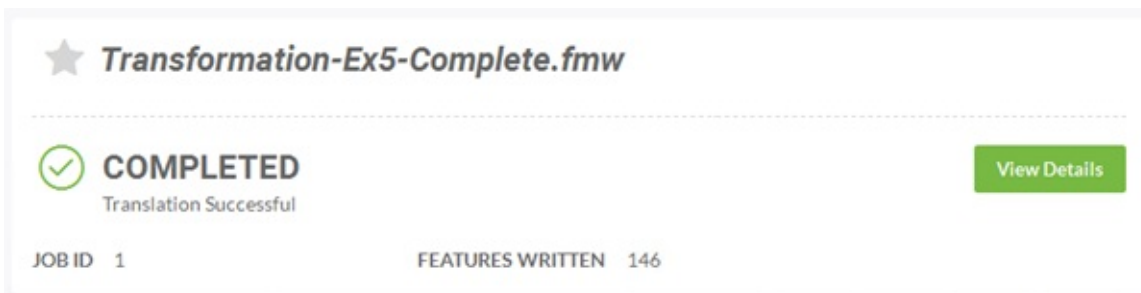
The 'Advanced' settings panel is a light gray box with a title bar containing a minus sign and a refresh icon. It contains five configuration rows. The first row is 'Job Priority' with a dropdown menu. The second row is 'Job Routing Tag' with a text input field. The third row is 'Queued Job Expiry Time' with a dropdown menu and a 'Seconds' unit selector. The fourth row is 'Running Job Expiry Time' with a dropdown menu and a 'Seconds' unit selector. The fifth row is 'Run Until Cancelled' with a checkbox.

Clicking the run button starts the translation:



The status bar shows a star icon followed by the workspace name 'Transformation-Ex5-Complete.fmw'. Below this, a green gear icon is next to the word 'RUNNING'. A message states: 'Your Job has been submitted and will continue to run if you leave this page.' To the right of the message is a loading spinner. Further right are two buttons: 'Cancel Job' and 'View Details'.

Having run, the interface will report the success of the translation:



The status bar shows a star icon followed by the workspace name 'Transformation-Ex5-Complete.fmw'. Below this, a green checkmark icon is next to the word 'COMPLETED'. A message states: 'Translation Successful'. To the right of the message is a 'View Details' button. At the bottom left, it says 'JOB ID 1'. At the bottom right, it says 'FEATURES WRITTEN 146'.

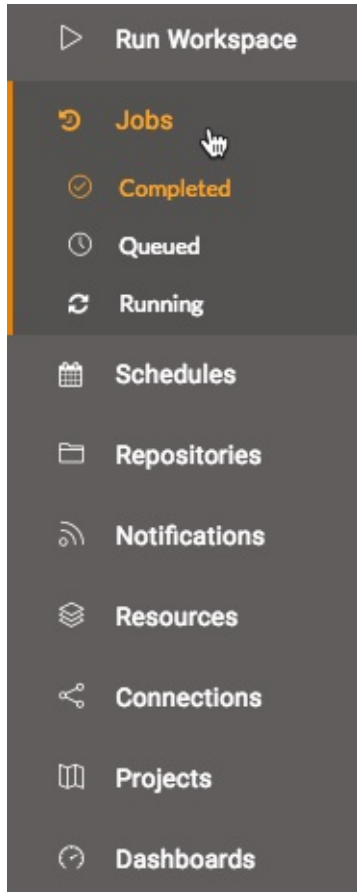
Police Chief Webb-Mapp says...

Note that there are other ways to find and run a workspace. Recently published workspaces and workspaces "starred" as a favourite can be easily found on the interface landing page.

Additionally it's possible to browse for a workspace to run by going through the Repositories page (click Repositories on the menu), which is more like a file browser than a simple selection tool.

Jobs

The final step in getting started on FME Server is to choose the menu option for Jobs:



This opens the Jobs page, where you can see the status of jobs, whether Completed, Queued, or Running:

A screenshot of the FME Server 'Jobs' page. The page title is 'Jobs' with a sub-header 'Completed'. Below the title are tabs for 'Completed', 'Queued', and 'Running'. A dropdown menu shows 'All Users'. There are 'Remove' and 'Remove All' buttons. A table lists jobs with columns: Id, Workspace, Repository, Username, Status, Engine, Finished, Started, and Priority. One job is listed with Id 1, Workspace Transformation-Ex5-Complete.fmw, Repository Training, Username admin, Status (green checkmark), Engine TRAINING2017_Engine2, Finished Today at 09:45:24, Started Today at 09:45:20, and Priority 100. At the bottom, there is a pagination bar showing '1 / 1' and '100*'.

| Id | Workspace | Repository | Username | Status | Engine | Finished | Started | Priority |
|----|---------------------------------|------------|----------|--------|----------------------|-------------------|-------------------|----------|
| 1 | Transformation-Ex5-Complete.fmw | Training | admin | ✓ | TRAINING2017_Engine2 | Today at 09:45:24 | Today at 09:45:20 | 100 |

This allows you to check that the translation you just ran finished successfully. You can also find jobs that are currently running, jobs that are queued to run, or any jobs that have completed running.

TIP

The drop-down menu that allows you to select which user's jobs to display is especially useful when the job history runs to thousands of workspaces from multiple users.

Completed Jobs

Clicking on a completed job opens up a page showing information about that job; such as Job ID, Job Priority, Time Started, and Features Written.

A series of buttons allow you to view the FME log for the translation, download the log, or even resubmit the job with a single click:



This allows you to confirm that the workspace functioned correctly, with the same level of detail as you could find within FME Desktop.

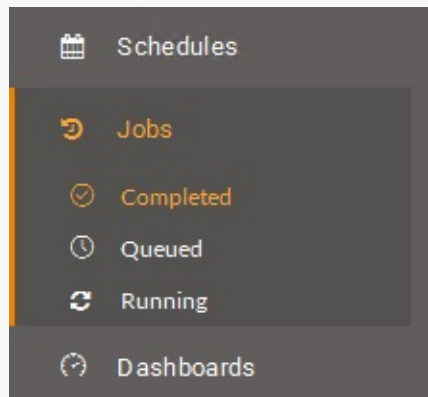
Queued and Running Jobs

Queued and Running jobs can be listed so that you can see what jobs the Server is currently handling. Again you can filter the jobs by a particular username.

One particularly useful feature is that these pages can be used to cancel jobs if they are no longer required.

TIP

The jobs entry - like others on the main menu - expands when clicked to show submenus that can be shortcuts to specific parts of the Jobs page:



| Exercise 2 Daily Database Updates: Running a Workspace | |
|--|---|
| Data | Firehalls (GML) Neighborhoods (KML) |
| Overall Goal | Create a workspace to read and process departmental data and publish it to FME Server |
| Demonstrates | Examining the FME Server interface and running a workspace |
| Start Workspace | None |
| End Workspace | None |

For the exercises in this chapter, you are a technical analyst in the GIS department of your local city. You have plenty of experience using FME Desktop, and your department is now investigating FME Server to evaluate its capabilities.

There are many departments within the city, and one of your tasks is to take the data from each department and merge it together into a single, corporate database.

Because each department produces their datasets in a different format and style, you use FME for this task, and carry it out on a weekly basis.

After creating a workspace to carry out this translation, and publishing it to FME Server, you now wish to log in to Server to run that workspace.

1) Connect to Server

To log in to the server interface either select the Web Interface option from the start menu or - in your web browser - enter the address to your FME Server.

TIP

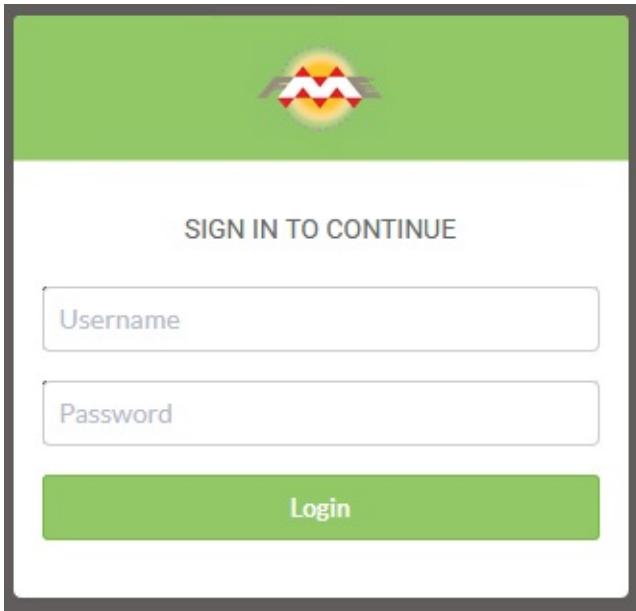
When FME Server is installed on either physical or virtual hardware, the address is `http://<servername>/fmeserver`

If you are using FME Cloud, then the address is: `http://<server name>.fmecloud.com/fmeserver`

This will open the web interface login screen for the FME Server being used. Bookmark this web address, since you will use this link quite often.

2) Log In to Server

In the User Login dialog, enter a username and password for your FME Server account. A common username/password combination for a training installation is admin/admin

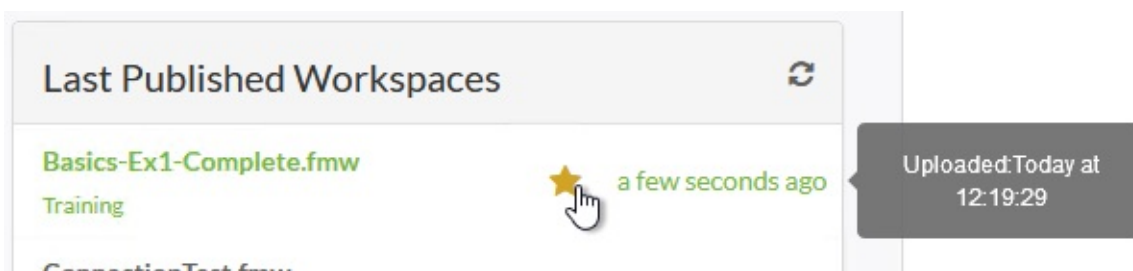
The image shows the FME Server login interface. At the top is a green header with the FME logo. Below the header, the text "SIGN IN TO CONTINUE" is centered. There are two input fields: "Username" and "Password". Below these fields is a green "Login" button.

Click the Login button.

3) Examine the User Interface

This is your primary method for interacting with FME Server.

Notice that one of the windows is labelled as Last Published Workspaces. Here you should be able to find the workspace published in Exercise 1:



Clicking the star icon will set this workspace as a favourite, making it available under the list of favourites panel:



We'll run the workspace shortly, but perhaps first we should make sure FME Server is running correctly (the fact that we could log in is a good sign) and that we are licensed and have engines running.

4) Examine the User Interface

Click Engines & Licensing on the ADMIN part of the interface menu. This will open up the licensing section. You should see a message informing you that FME Server is licensed and a list of the engines available:

FME Server is Licensed

Licensing Documentation

Refresh License

Request License

Upload License File

MAX_ENGINES: 4

MACHINE_KEY: 1633991022

EXPIRES: Never

SERIAL_NUMBER: Training

Engines

Engines will appear in the table below when ready. The engine list refreshes automatically.

Host: TRAINING2017

| Name | Build | Platform | Current Job |
|----------------------|-------|----------|-------------|
| TRAINING2017_Engine1 | 17259 | WIN64 | |
| TRAINING2017_Engine2 | 17259 | WIN64 | |

Total active engines: 2

TIP


If your machine is unlicensed, or is missing engines, then check with your instructor for troubleshooting tips.

5) Run Workspace

Click the FME Server button in the very top-left of the interface. This will return you to the Server interface home page.


Click on the published workspace in the Favourite Workspace panel to open the web page for this workspace.


The workspace page shows a few options, the first of which are for the repository, workspace, and service. These should already be filled in with values:

| | |
|--|--|
| Repository | <input type="text" value="Training"/> |
| Workspace | <input type="text" value="Basics-Ex1-Complete.fmw"/> ★ |
| Service | <input type="text" value="Job Submitter"/> |
| Email results to  | <input type="text"/> |

Because this workspace has a few published parameters, they are also listed; but we can ignore these for now (we'll deal with source datasets and the like shortly).

So, simply click the Run button to run the workspace. The workspace will run to completion and a message to that effect will appear:

 ***Basics-Ex1-Complete.fmw***

 **COMPLETED**
Translation Successful

[View Details](#)

JOB ID 2 **FEATURES WRITTEN** 14

6) Examine Jobs Page

Click Jobs on the main menu. A list of previously run jobs will open, including the one we just ran:

Jobs ⓘ

Completed Queued Running **1**

Completed

Show Jobs For:

All Users **2**

3 **4** [Remove](#) [Remove All](#)

| <input type="checkbox"/> | Id | Workspace | Repository | Username | Status | Engine | Finished | Started | Priority |
|--------------------------|----|-------------------------|------------|----------|--------|----------------------|-------------------|-------------------|----------|
| <input type="checkbox"/> | 2 | Basics-Ex1-Complete.fmw | Training | admin | ✓ | TRAINING2017_Engine1 | Today at 12:38:09 | Today at 12:38:03 | 100 |
| <input type="checkbox"/> | 1 | ConnectionTest.fmw | Training | admin | ! | TRAINING2017_Engine2 | Today at 10:20:56 | Today at 10:20:49 | 100 |

1 / 1 100

Displaying 1 - 2 of 2

Notice some interesting points of the interface:

1. There are links to show Completed jobs (the default), Queued Jobs, and Running Jobs.
2. There is a drop-down list that allows you to filter whose jobs are being shown.
3. Jobs that are successful and which fail are differentiated using a different icon.
4. The jobs are displayed in the chronological order in which they finished (whether successful or not).

Click on your job to inspect the results in more detail. You will be able to see details about the job including the time at which it was submitted, queued, started, finished, and delivered; the exact request made to FME Server; and the full results of the translation. You may also click the View Log button to inspect the FME translation log file.

Police Chief Webb-Mapp says...

Remember, this workspace did not write any data, only sent it to a Null writer. So, for now, to view any results search for the summary in the log file.

Advanced Exercise

If you want to see a job in a different state then we'll have to slow this workspace down some.

Open the workspace in FME Workbench and add a Decelerator transformer (say, before the Reprojector). Set it to delay the workspace by five (5) seconds per feature. Publish the workspace back to FME Server and re-run it.

Now the workspace will take 30+ seconds to run and you should be able to find it under the Running state. Also, if you run it three or four times in quick succession, then you will have more jobs than engines and be able to find some jobs in the Queued state.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Log in to FME Server and check that it is running and licensed*
- *Locate a workspace using the Last Published list*
- *Run a workspace and inspect the job history to confirm it ran correctly*
- *Find and upload resources to FME Server*
- *Check the parameters for cleanup tools*

Sharing

FME Server security is based on whether you own a component or have been given access to it. A component might be a set of functionality or an object like a repository.

When you create something you have full permission for that component. Even if you don't have permission to manage security on your FME Server, you do have the ability to share a component with another user.

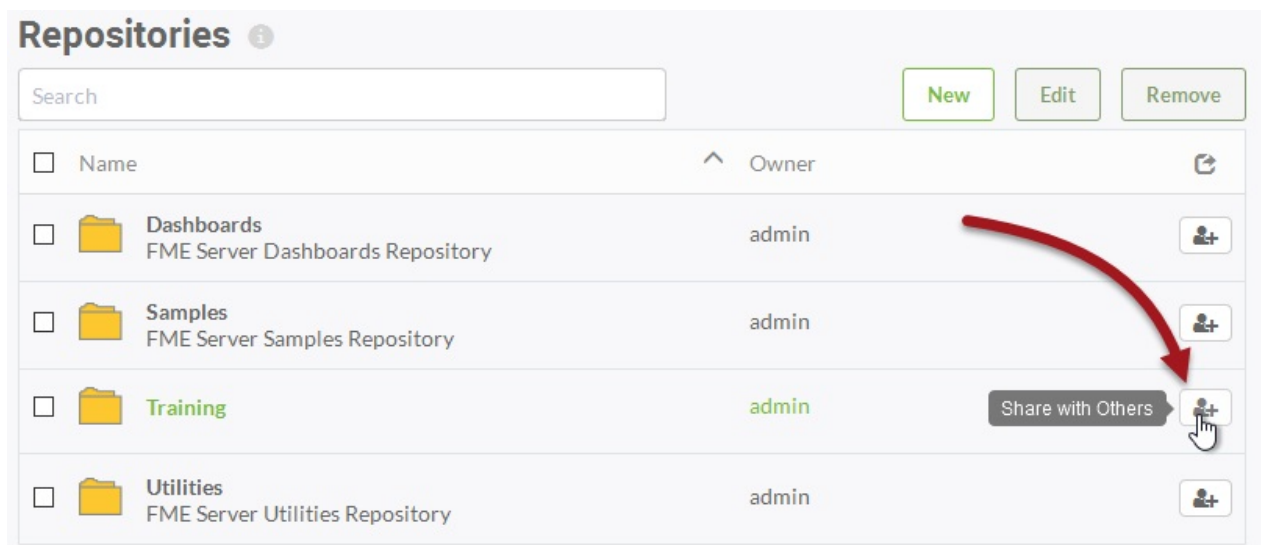
NEW

Component sharing is a new piece of functionality for FME 2017.

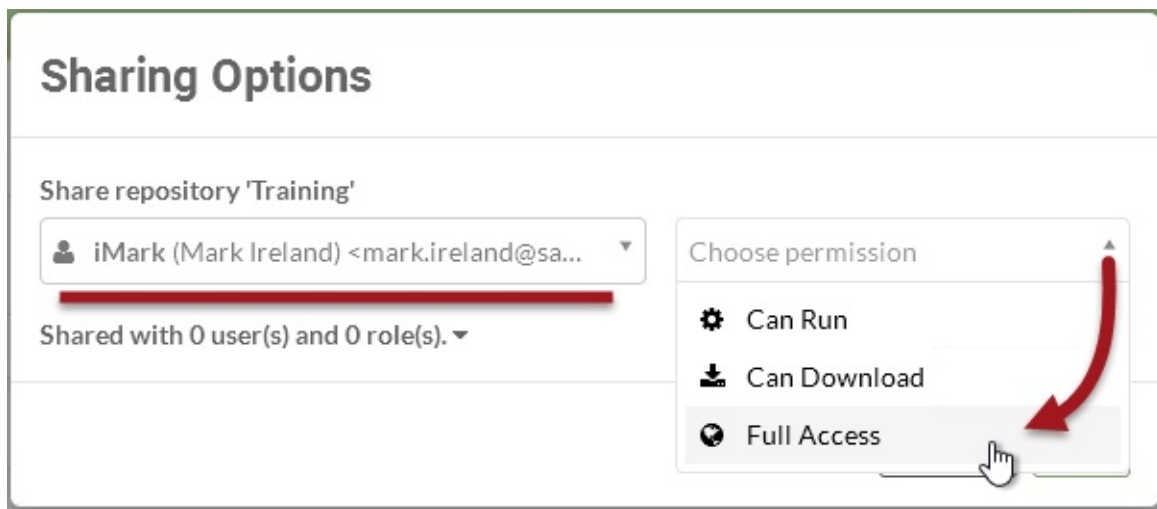
Sharing a Repository

Choose the menu option for Repositories in the Server web interface and you are presented with a list of repositories on the system.

If you are the owner of a repository then you have the ability to click the button to *Share with Others*:



This opens a pop-up dialog in which to select a user and choose the level of permission that you wish to give to them:



FME Security is also based on users and roles. Roles are analogous to a group of users. When sharing a component, the "user" field can be an individual user, or it can be applied to a particular role; for example you can give the ability to run workspaces in a repository to anyone in the *fmeuser* role.

Police Chief Webb-Mapp says...

This is a very important capability. As an author you might publish a workspace intended for use by multiple users inside (and outside) an organization. However, the workspace is of little use if those users don't have access to it.

The Share with Others tool allows you to open up access to your workspace, without you needing the advanced permissions required for full security control.

Besides repositories, other components of FME Server can also be shared with other users. Keep watch within the user interface, and throughout this manual, for other sharing opportunities.

Scheduling

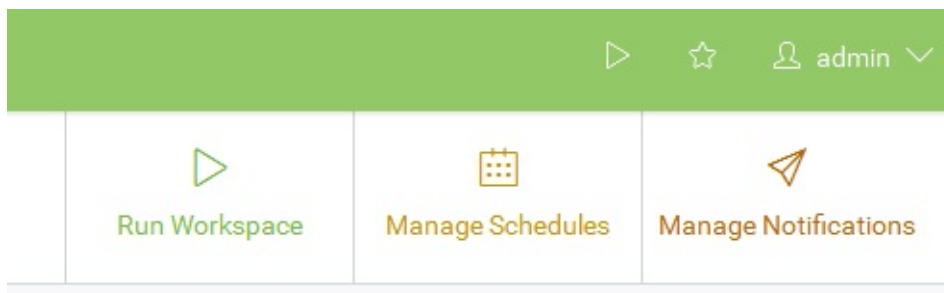
Scheduled Translations are the best way to kick off a workspace at a particular time or date.

What is Scheduling?

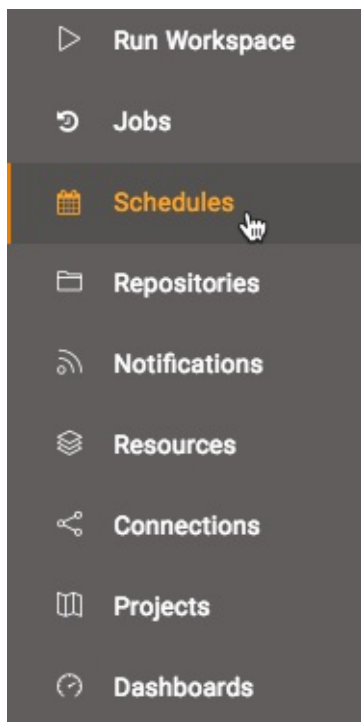
Scheduling is the ability to program FME Server to run a workspace in a repository, at a specific time in the future. The schedule can cause the workspace to run once or on a repeating basis.

Managing Scheduled Tasks

Scheduled tasks are set up in the web interface. Firstly there is a quick link on the landing page to scheduled tasks:



...and secondly there is also a button on the main menu:



The interface supports all the capabilities you would expect, including the ability to create new schedules, remove existing ones, copy existing ones, and to enable and disable existing tasks:

Schedules ⓘ

Search

New

Remove

Duplicate

Enable

Disable

| <input type="checkbox"/> Name | Category | Start Time | End Time | Recurrence | Workspace | Status | Owner | |
|---|------------|---------------------|----------|------------|-----------------------------------|--------|-------|--|
| <input type="checkbox"/> Backup_Configuration | Utilities | 2012-01-01 02:00:00 | N/A | DAY * 1 | backupConfiguration.fmw | | admin | |
| <input type="checkbox"/> DashboardStatisticsGathering | Dashboards | 2016-01-01 00:00:00 | N/A | DAY * 1 | JobHistoryStatisticsGathering.fmw | | admin | |

Creating a Scheduled Task

There are a number of parameters involved in creating a scheduled task.

The first parameters are very simple ones for naming and describing the schedule.

Schedules > New Schedule

General Settings

Name

My Schedule

Category

Training

Description

A scheduled job for FME training

Notice that each schedule can be assigned to a particular category.

The next few parameters concentrate on the workspace to be run.

Once a workspace is selected there will be a short pause while FME retrieves information about the workspace. It will then expose any published parameters that exist in the workspace:

Workspace Settings

Repository

Training

Workspace

MyWorkspace.fmw

Published Parameters

Source Esri Shapefile(s):

☐ Browse resources ☒ Specify a location

Please enter a Location

C:\FMEData2017\Data\Parcels\NewDevelopment.zip

Destination Adobe 3D PDF File:


☐ Browse resources ☒ Specify a location

Please enter a Location


C:\FMEData2017\Output\Training\SurfaceModel.pdf

The key parameters, of course, are for setting the actual schedule. Here the workspace is set to run every week starting on the 30th January at 2:00am

Schedule Settings

Start Date 

☐ Immediately

2017-02-03 

02:00 ▼

Repeats


☒ Interval Based ☐ CRON Expression ☐ Only Occur Once

Repeat Unit


DAY ▼

Interval

1 ▲▼

End Date 

☒ Never



▼

There are also optional parameters for notification topics to trigger on completion of the scheduled task. These could be used to inform an administrator of the success or failure of the translation.

Finally there are options to control job priority, job routing (which engine it should use) and job expiry (for jobs that are time-sensitive and would be no longer useful if held back past a certain time by higher priority tasks).

Once the parameters are set for a scheduled task, it is added to the main Scheduling interface.

| Exercise 3 Daily Database Updates: Sharing and Scheduling | |
|--|---|
| Data | Firehalls (GML) Neighborhoods (KML) |
| Overall Goal | Create a workspace to read and process departmental data and publish it to FME Server |
| Demonstrates | Sharing and scheduling a translation in FME Server |
| Start Workspace | None |
| End Workspace | None |

For the exercises in this chapter, you are a technical analyst in the GIS department of your local city. You have plenty of experience using FME Desktop, and your department is now investigating FME Server to evaluate its capabilities.

There are many departments within the city, and one of your tasks is to take the data from each department and merge it together into a single, corporate database.

Because each department produces their datasets in a different format and style, you use FME for this task, and carry it out on a weekly basis.

You have already (Exercises 1 and 2) created a workspace to carry out this translation, published it to FME Server, and ran it to confirm it works.

As a daily task, you plan to run the translation every day after work. However... what happens if you are not there, or leave early, or someone else stays late. Who will run it then?

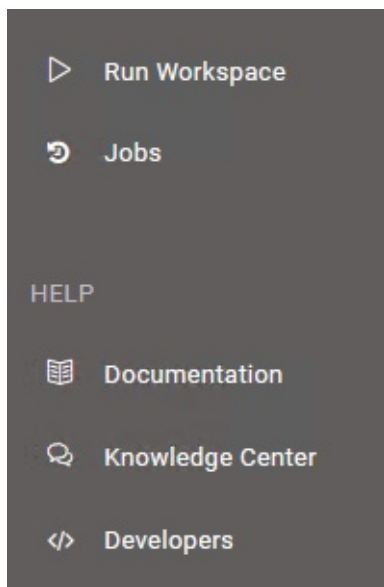
Firstly you should ensure other users have access to the workspace to run it, but you can also set it up to run on an automatic schedule.

1) Connect to Server

Browse to the log in page of the FME Server interface, either starting it through the Web Interface option on the start menu or by logging out if you are already logged in.

This time log in using the generic user account that is a default account on any FME Server installation. The username is **user** and the password is **user** as well!

The first thing you'll notice is that the menu and functionality is much more restricted for this account:

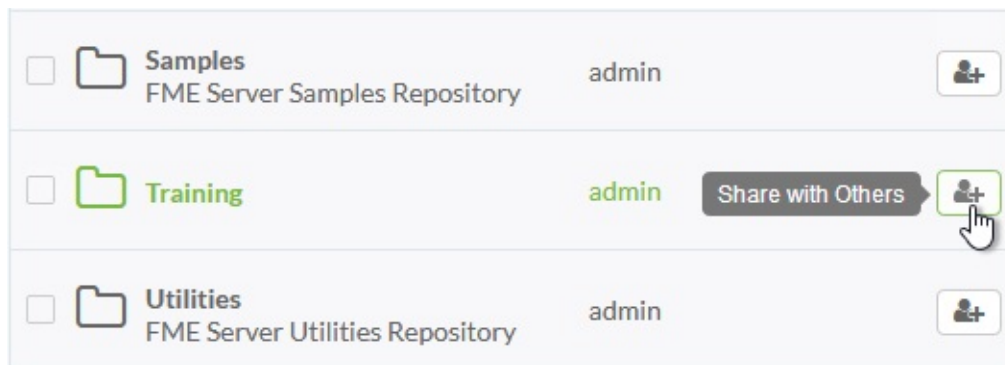


In fact, if you try to run a workspace you'll find that the only repository this account has access to is the Samples repository; not Training where the existing workspace resides.

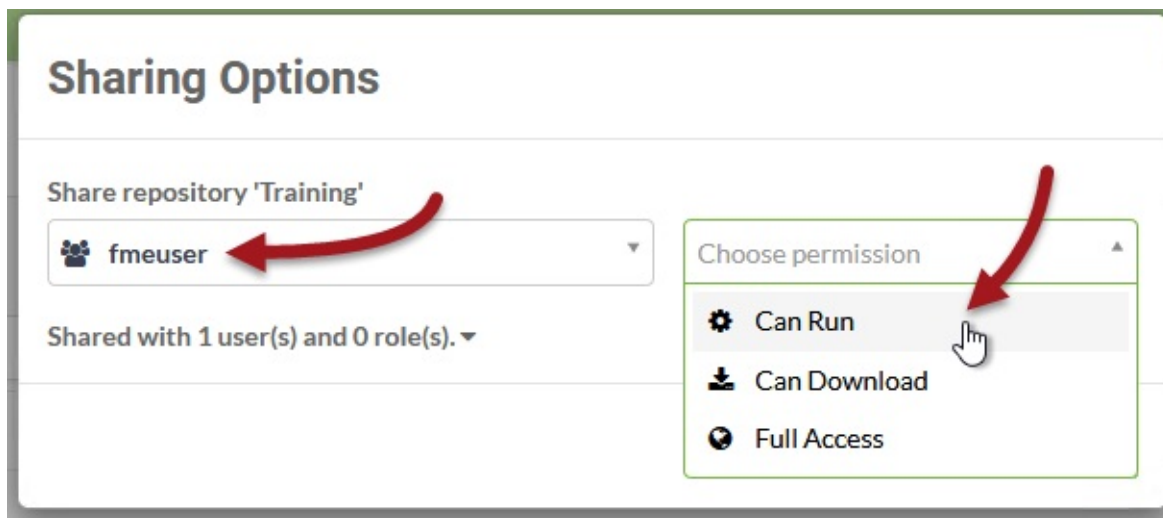
2) Share Repository

Log out of the user account and log back in as an administrator (admin/admin).

Now you have the full set of menu entries, click Repositories on the menu. Under the list of repositories locate the Training repository. Click the Share icon to the right:



In the Sharing Options dialog, select fmeuser as the role to share with, and allow them to run the workspace:



By selecting the *fmeuser* role (rather than the single *user* account) we allow anyone who is tagged as a user to access the workspace; and by allowing them run capability only, we prevent them downloading and making edits to our workspace.

3) Check Sharing

Log out of the administrator account and log back into FME Server with the user account (user/user).

This time you should have access to the Training repository and be able to run the workspace successfully as a general user. Check the Jobs page and you'll see one entry for the workspace, when it was run as the user. There is only one entry because the user does not have the privileges required to view any other users' jobs.

Log out again and log back in as an administrator. Now in the Jobs window you should be able to see both the administrator's jobs and the user's jobs:

| <input type="checkbox"/> | Id | Workspace | Repository | Username | Status | Engine |
|--------------------------|----|-------------------------|------------|----------|--------|----------------------|
| <input type="checkbox"/> | 4 | Basics-Ex1-Complete.fmw | Training | user | ✓ | TRAINING2017_Engine1 |
| <input type="checkbox"/> | 3 | Basics-Ex1-Complete.fmw | Training | admin | ✓ | TRAINING2017_Engine2 |
| <input type="checkbox"/> | 2 | Basics-Ex1-Complete.fmw | Training | admin | ✓ | TRAINING2017_Engine1 |
| <input type="checkbox"/> | 1 | ConnectionTest.fmw | Training | admin | ! | TRAINING2017_Engine2 |

That's because the administrator group does have permission to view all jobs.

4) Create Test Schedule

Now we've allowed other users to run the workspace on demand, but we should also set up the translation to run on a schedule.

Firstly, just to confirm that scheduling does work, let's set up a test schedule. Click Schedules on the menu and in the Schedules window click the New button to start the process.

Set a name of Test Schedule and add it to a Training category by typing Training into the Category field:

The screenshot shows a form with three fields: 'Name' with the value 'Test Schedule', 'Category' with the value 'Training' and a '+' button to the right, and 'Description' with a dropdown menu showing 'Training (new)' in a green bar.

For the time settings, set the schedule to start immediately and run every 30 seconds. Set the end date to be approximately 30 minutes into the future (that way if we forget to cancel the schedule it won't carry on for ever!)

The screenshot shows the time settings section of the form. It includes a green checkmark for 'Enabled', another green checkmark for 'Run Immediately', a dropdown menu for 'Recurrence' set to 'Repeat On Interval', a 'Repeat Every' section with a text box containing '30' and a unit dropdown set to 'Seconds', a checkbox for 'Schedule Does Not Expire' which is unchecked, and an 'End Time' field with the value '2017-4-24 18:00:00' and a calendar icon to its right.

Be aware that the times are given in 24-hour format, so 1:30 means AM and 13:30 means PM. It is also important to note that this time is the local time of the machine FME Server is installed to.

Under Workspace Settings, select the Training repository and within that the workspace previously uploaded (Basics-Ex1-Complete.fmw):

The screenshot shows the 'Repository' dropdown menu set to 'Training' and the 'Workspace' dropdown menu set to 'Basics-Ex1-Complete.fmw' with a star icon next to it.

There are no user parameters we need to change for this workspace, so any can be ignored.

Now click OK to add the new schedule.

5) Examine Jobs Page

Open the Jobs page. A list of previously run jobs will open. You will find (if it was set up correctly) that there will be jobs running to schedule:

| <input type="checkbox"/> | Id | Workspace | Repository | Username | Status | Engine | Finished | Started |
|--------------------------|----|-------------------------|------------|----------|--------|----------------------|-------------------|-------------------|
| <input type="checkbox"/> | 7 | Basics-Ex1-Complete.fmw | Training | admin | | TRAINING2017_Engine2 | Today at 14:49:38 | Today at 14:49:37 |
| <input type="checkbox"/> | 6 | Basics-Ex1-Complete.fmw | Training | admin | | TRAINING2017_Engine2 | Today at 14:49:08 | Today at 14:49:07 |
| <input type="checkbox"/> | 5 | Basics-Ex1-Complete.fmw | Training | admin | | TRAINING2017_Engine2 | Today at 14:48:39 | Today at 14:48:37 |

Notice that the username is set to admin; since that is the user who created the schedule, that is the username under which the job will be run.

6) Create Actual Schedule

Now we are confident that we know how to use the interface, let's set up an actual schedule. We want the workspace to run, say, every day of the week. There should also be no end date.

So, return to the Schedules page. You may now either:

- Click on the Test schedule and edit it to the required values
- Delete the test schedule and create a new one with the required values

Enabled ☒

Run Immediately ☐

Start Time

Recurrence

Repeat Every

Schedule Does Not Expire ☒

This setup will run the workspace at 8:00pm every day. Don't forget to click the OK button!

You may wish to check back periodically during this training to ensure the workspace runs as expected.

CONGRATULATIONS

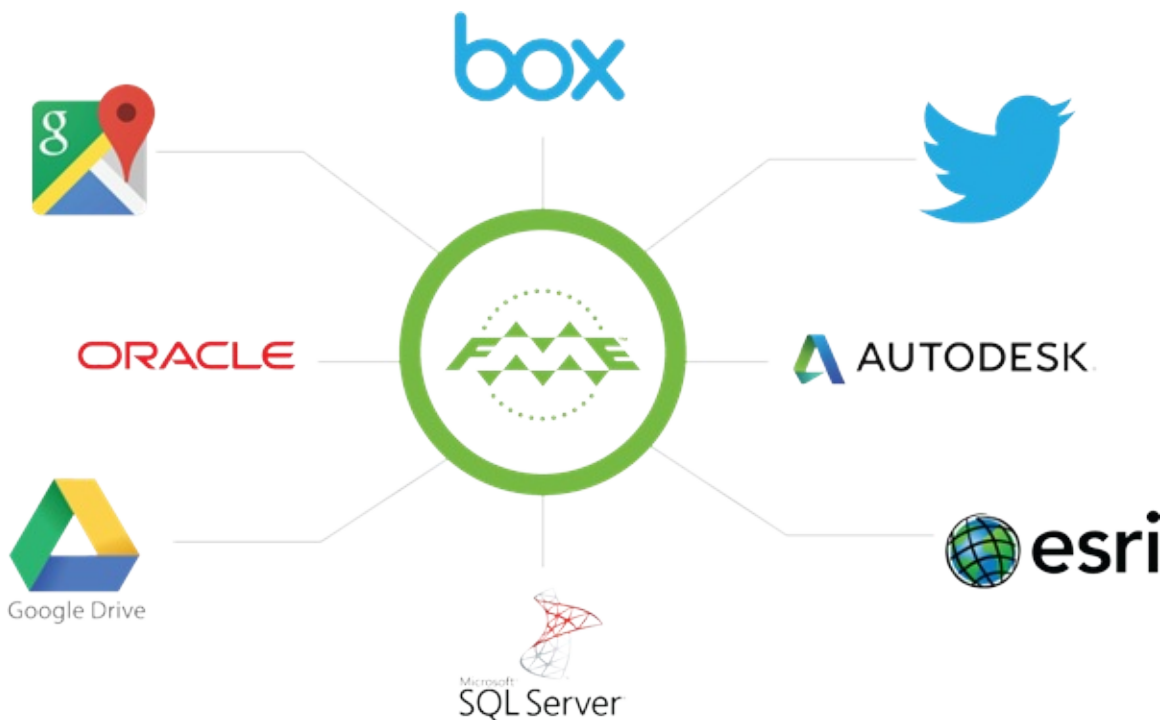
By completing this exercise you have learned how to:

- *Share a repository in FME Server and tested to ensure it is available to the right users*
- *Schedule a translation in FME Server*
- *Check the job history to ensure the scheduled translation took place*

Source Data Management

Nearly every FME workspace starts by reading features from a source dataset.

In some cases that source dataset may be held in a database, or stored on a web service such as Google Drive; it may even be a database running on a web service! On other occasions the data may not be web-based at all, but stored on a file system and shared for others to access.



In general, it's easy to author a workspace for use on FME Server because what works on FME Desktop can be published and run on Server with very little alteration of the source data practices.

However, there are additional methods that can be used to take source data management on FME Server to the next level. These methods include:

- Using Database Connections for data stored in databases
- Using Web Connections for datasets stored on a web service
- Managing file datasets
 - Publishing file datasets to a repository
 - Uploading file datasets at run time
 - Storing file datasets on the 'Resources' file system

Using Database Connections

When the source data for a dataset is a database, FME is capable of storing connection parameters in a secure container. That container can be either published to FME Server or recreated on it.

What is a Database Connection?

Database connections are containers for a set of database connection parameters. These parameters include database server, port number, username, password, and others that vary according to the database type.

The two main advantages of database connections are:

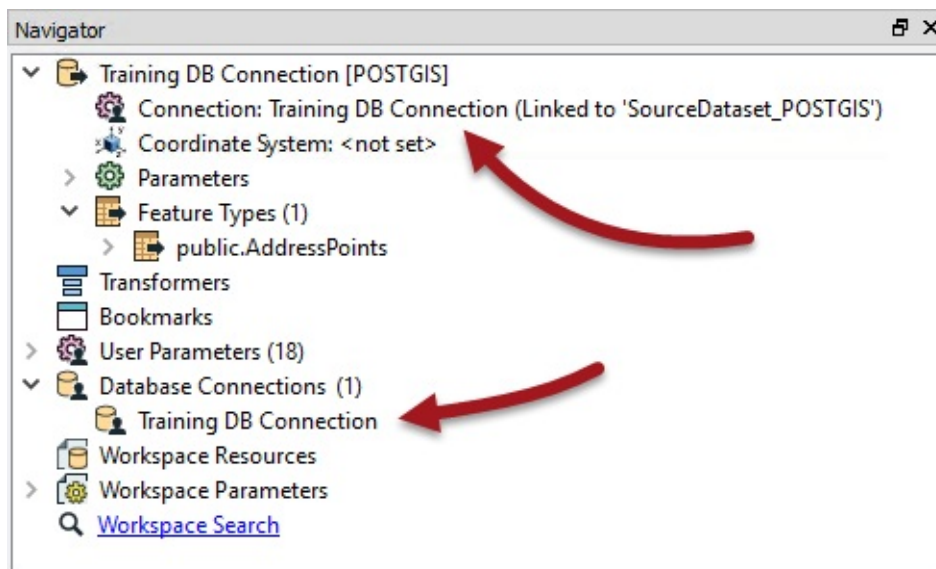
- Connection parameters are no longer embedded in a workspace, meaning less of a security risk
 - For example, your parameters would not be exposed to anyone who downloaded the workspace
- Connection parameters can be reused among multiple workspaces
 - For example, two workspaces that use the same database can use the same connection

Database connections can be published with a workspace from FME Desktop, or they can be added directly within FME Server.

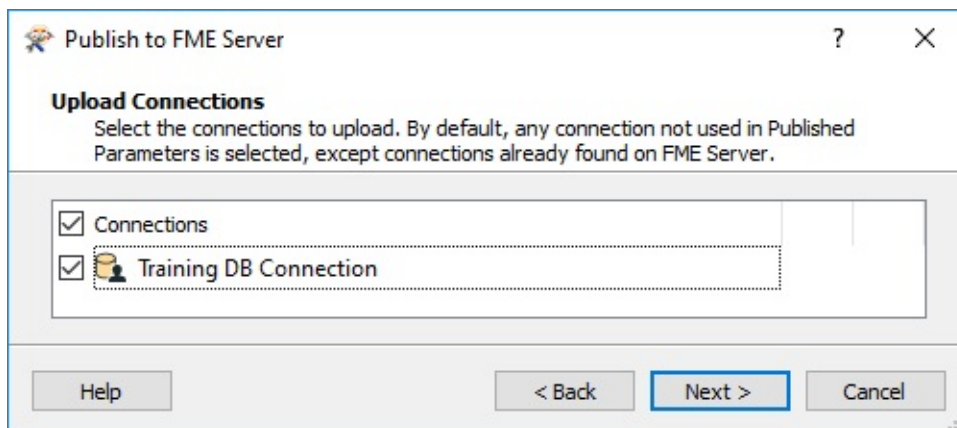
Creating Database Connections

Creating a database connection often starts in FME Desktop. They can be created using Tools > FME Options > Database Connections in the FME Workbench menubar. The defined connection can then be used in a reader, writer, or transformer.

For example, this workspace has a connection for a PostGIS database, as seen in the reader parameters and a list of database connections:



When the workspace is published to FME Server a new dialog asks the author whether to also publish the database connection:



The connection is then added to the connections container on FME Server.

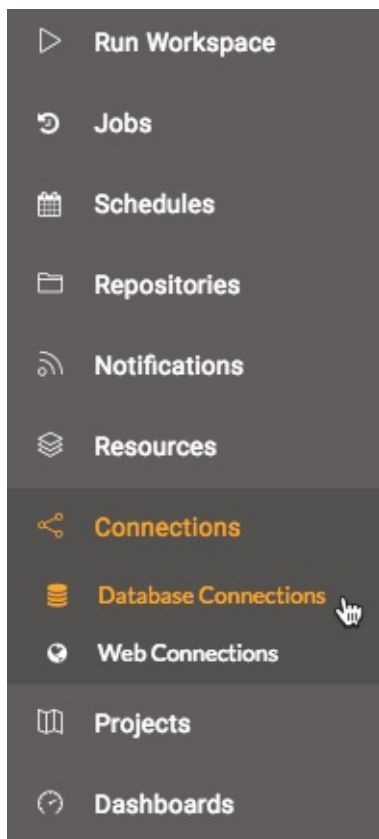
Police Chief Webb-Mapp says...

Note that you don't have to upload the connection with the workspace. If a connection for that database already exists on FME Server you can use that.

If you don't upload the connection, and you don't already have one to use on FME Server, then you will need to use the Database Connections page to create one.

Managing Database Connections

FME Server has a page for managing database connections accessed through the main menu:



This page allows workspace authors - but usually administrators - to create new connections, copy existing connections, delete existing connections, or edit existing connections:

Database Connections > Training DB Connection

Editing "Training DB Connection"

| | |
|----------|-----------------------|
| Type | PostgreSQL |
| Host | postgis.train.safe.ca |
| Port | 5432 |
| Database | fmedata |
| Username | fmedata |
| Password | ••• |

Using Database Connections

When a workspace is run, if it has a database reader (for example) the end-user is prompted with a published parameter and can select the database connection to use:

Training/postgis2none

Repository: Training

Workspace: postgis2none.fmw ★

Service: Job Submitter

Email results to ⓘ [Reset](#)

Published Parameters

Connection: Training DB Connection

The workspace then runs to completion as normal.

Police Chief Webb-Mapp says...

Connections, like other objects on FME Server, have security permissions. Only the owner, someone with whom the connection is shared, or (by default) an administrator, can make use of it. It's not the case that any random user will be provided access to all database connections via the published parameter.

This also means that a workspace can be tested in FME Desktop using the author's connection parameters, but then require the end-user's connection once published to Server; all in a way that is both easy and secure.

Miss Vector says...

If I create a database connection that has superuser permissions then it would bypass any permission checks the database would make for creating and dropping tables. So how do you think I might prevent a user misusing that capability?

- 1. Remove that user's permission to run that workspace on FME Server*
- 2. Remove that user's permissions to access the entire repository that workspace resides in*
- 3. Remove permission to access that particular database connection for that user's role*
- 4. Remove from their role permission to manage database connections*

Using Web Connections

Just as for databases, when the source data for a dataset is a web service, FME is capable of storing connection parameters in a secure container. That container can be either published to FME Server or recreated on it.

What is a Web Connection?

Web connections are containers for a set of web service connection parameters. These parameters include the service, username, password (or authenticated connection), and others that vary according to the service type.

The two main advantages of web connections are:

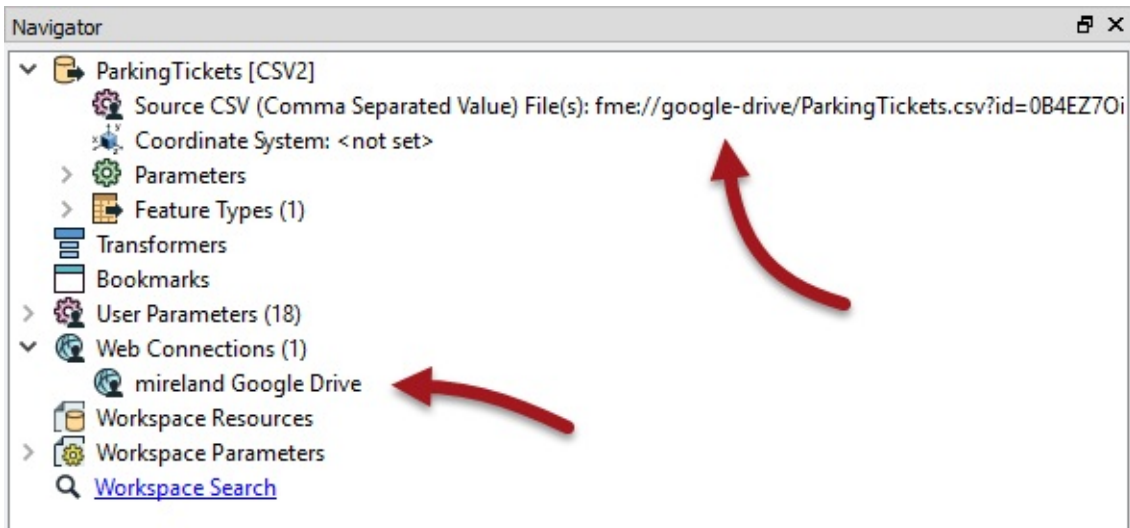
- Connection parameters are no longer embedded in a workspace, meaning less of a security risk
 - For example, your parameters would not be exposed to anyone who downloaded the workspace
- Connection parameters can be reused among multiple workspaces
 - For example, two workspaces that use the same web service can use the same connection

Web connections can be published with a workspace from FME Desktop, or they can be added directly within FME Server.

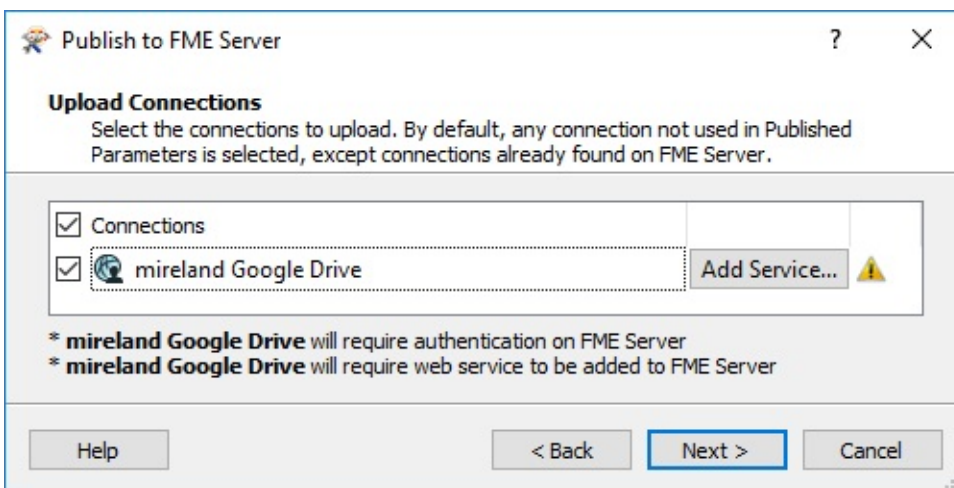
Creating Web Connections

Creating a web connection often starts in FME Desktop. They can be created using Tools > FME Options > Web Connections in the FME Workbench menubar. The defined connection can then be used in a reader, writer, or transformer.

For example, this workspace reads a CSV dataset using a connection to a Google Drive web service, as seen in the reader parameters and a list of web connections:



When the workspace is published to FME Server a new dialog asks the author whether to also publish the web connection:



The connection is then added to the connections container on FME Server.

Police Chief Webb-Mapp says...

Note that - for web services - simply uploading them to FME Server is not enough. Before you use a web connection for the first time from FME Server, you must authorize it, even if it has already been authorized in the workspace from FME Desktop.

Web Connections > Edit

mireland Google Drive

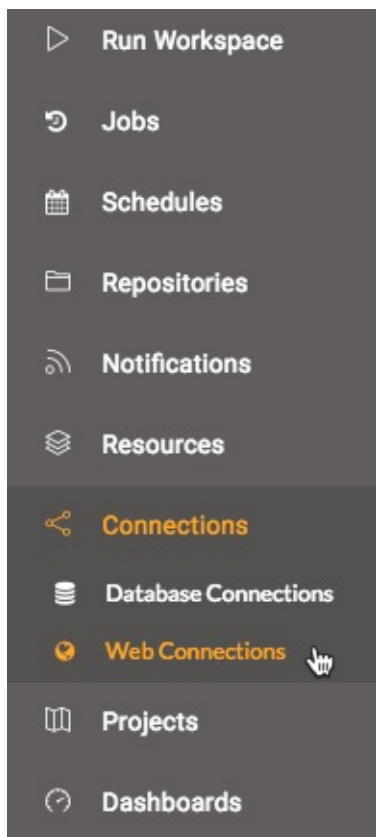
| Type | OAUTH - Google Drive |
|---|----------------------|
| <div>You will have to Authorize this connection before use</div> <div>Authorize</div> | |

*It's **also important** to know that the deployment you are using affects authentication. For example, by default you can't authenticate OAuth2 web services like Google Drive on an FME Server that uses a local host name. Requests for authorization that pass through a web server must come from a public domain.*

Setting up the requirements for OAuth authentication is a task best left to system administrators. See the FME Server Administrator's Guide for more information on how to carry it out.

Managing Web Connections

FME Server has a page for managing web connections, accessed through the main menu:

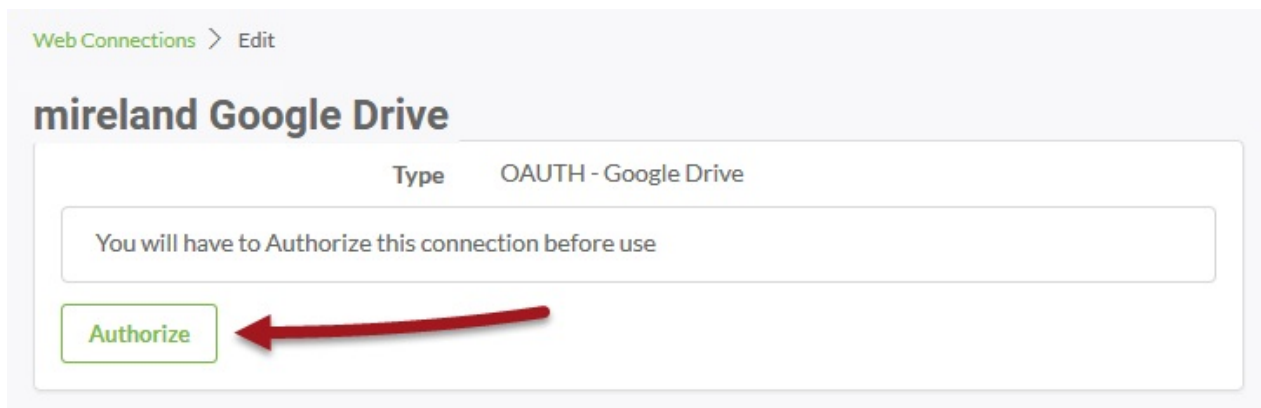


This page allows workspace authors - but usually administrators - to create new connections, copy existing connections, delete existing connections, or edit existing connections.

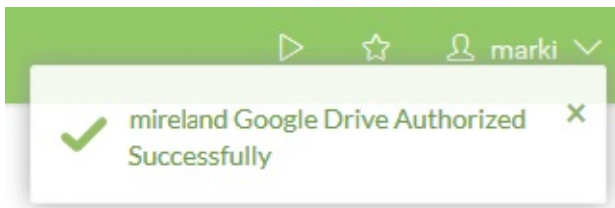
A form titled 'Create Web Connection' with a light grey background. It contains two input fields: 'Name' with the value 'Training Google Drive Connection' and 'Type' which is currently empty. Below the 'Type' field, a dropdown menu is open, showing two options: 'Dropbox (OAUTH)' and 'Google Drive (OAUTH)'. The 'Google Drive (OAUTH)' option is highlighted in blue and has a mouse cursor pointing at it. To the left of the dropdown, there is a text box containing the instruction: 'After saving you should authorize with the API Provider'.

Notice that a web connection of a specific type can only be added when that Web Service has already been added to the FME Server; for example, in the above only Dropbox and Google Drive connections can be created because only those two services already exist on FME Server.

Connections can also be authorized by clicking the Authorize button:



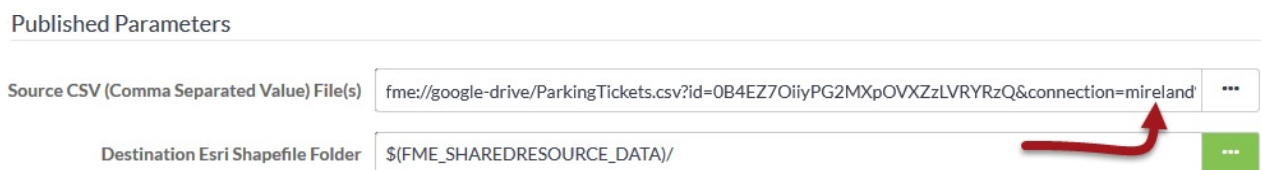
...after which you carry out the authorization using the standard method for that particular service inside a pop-up window.



Using Web Connections

When a workspace is run, if it has a transformer or reader that references a web service then it will run correctly, just as on a Desktop installation.

In the published parameters on Server, the web connection is defined in the source dataset URL:



Sister Intuitive says...

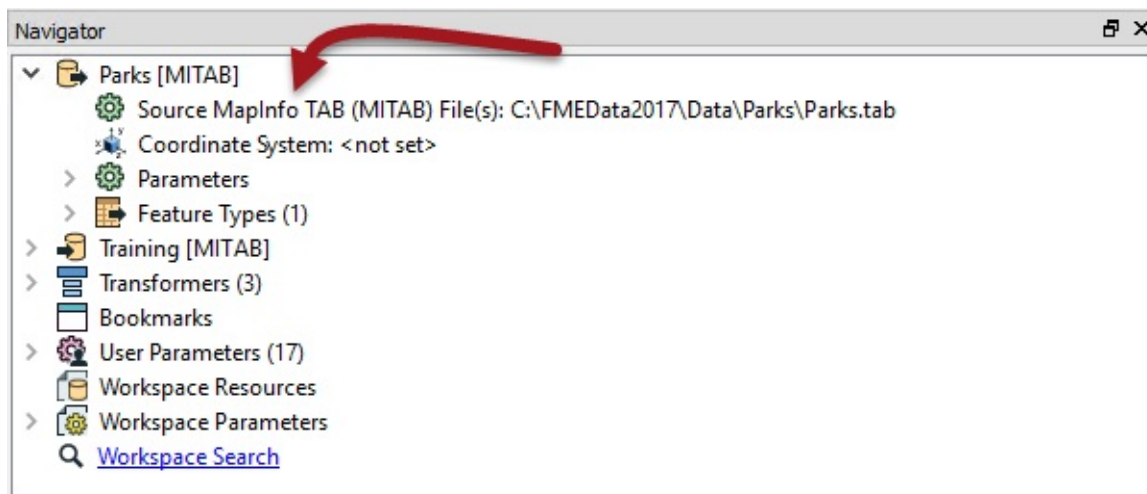
As with database connections, this functionality allows a workspace to be tested in FME Desktop using the author's connection parameters, but then switched to a general account once published to Server; all in a way that is both easy and secure.

Publishing File Datasets to a Repository

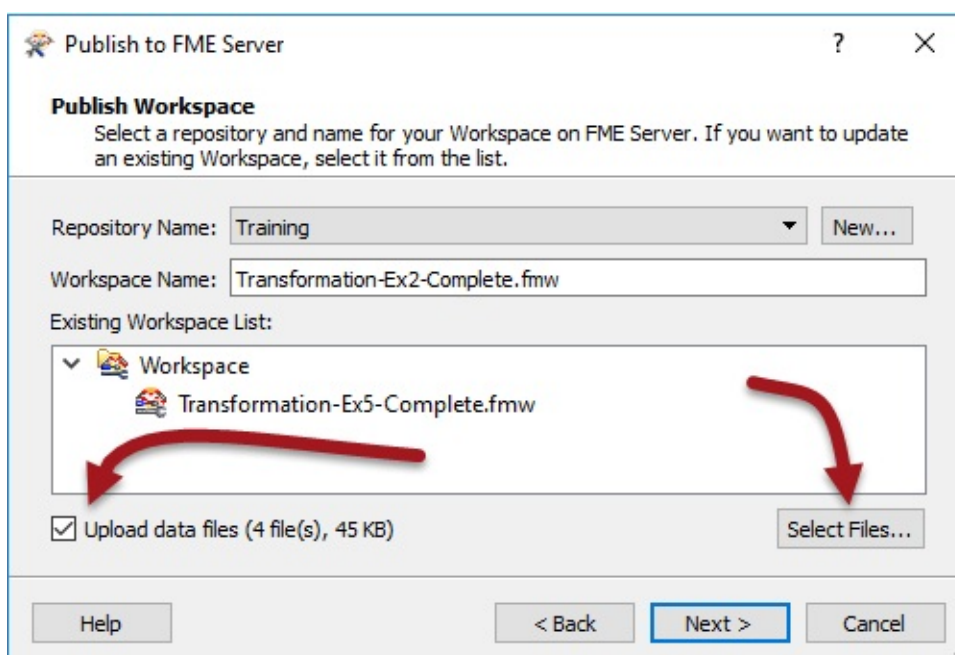
When the source data for a translation is stored as files (rather than a feed or database) it can be published to an FME Server repository along with the workspace.

Publishing Source Data

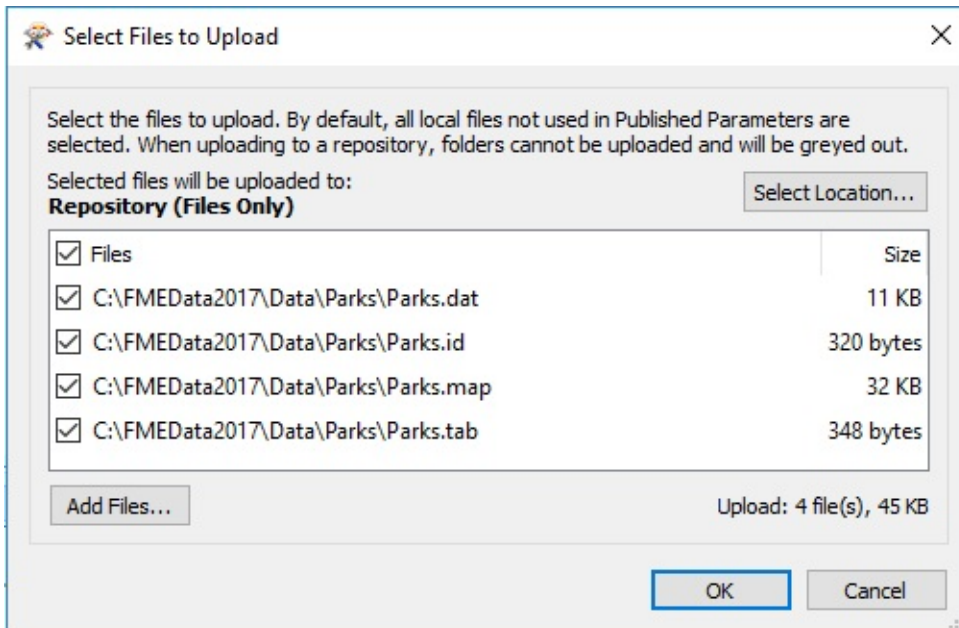
In this workspace the source dataset is MapInfo TAB:



A MapInfo TAB dataset is made up of a series of files (.tab, .dat, .id, .map). When this workspace is published the wizard allows us to publish the data files alongside it by simply checking the box labelled *Upload data files*.



FME automatically selects the files to upload based on what it thinks is necessary to run the translation. If there are other files you wish to upload, or files FME selected that you don't wish to upload, the Select Files button allows you to make changes:



This dialog also allows you to change *where* the files are published to, but for now we'll ignore that setting and go with the default of publishing to the repository.

Once the publishing wizard is complete, those files are uploaded to FME Server and tagged for use with this workspace.

Using Published Source Data

When a workspace published with its data is run on FME Server, the uploaded data automatically is used as the source:

Published Parameters

Source MapInfo TAB (MITAB) File(s)

\$(FME_MF_DIR)Parks.tab



There are no other settings that need to be changed and the workspace will run to completion using the published data as its source.

Miss Vector says...

Although simple, there is a major limitation to publishing data with a workspace. What do you think it is?

- 1. The data is only temporary and will be deleted once the workspace is run*
- 2. The data is hidden within FME Server's system files and limited in its use*
- 3. The data becomes available to anyone regardless of role and security settings*
- 4. The workspace cannot be run using any other data than that published with it*

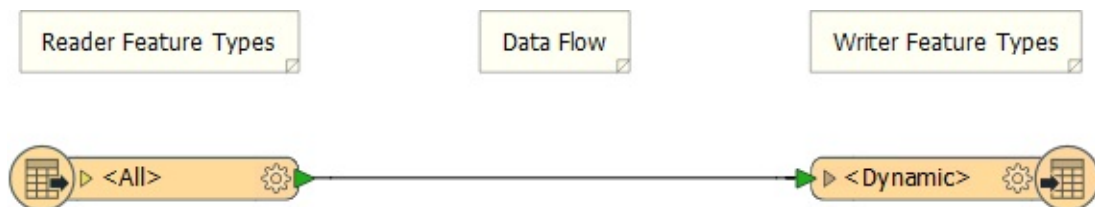
Uploading Datasets at Run Time

Although it's easy for an author to publish data to an FME Server repository along with the workspace, it isn't a method that an end-user has access to.

Therefore, for files (rather than a feed or database), functionality exists to allow the end-user to upload data at run-time.

Uploading Source Data

This workspace was created with a dynamic Reader and Writer. That means it is possible to process any source dataset (of the right format) and have it translated:



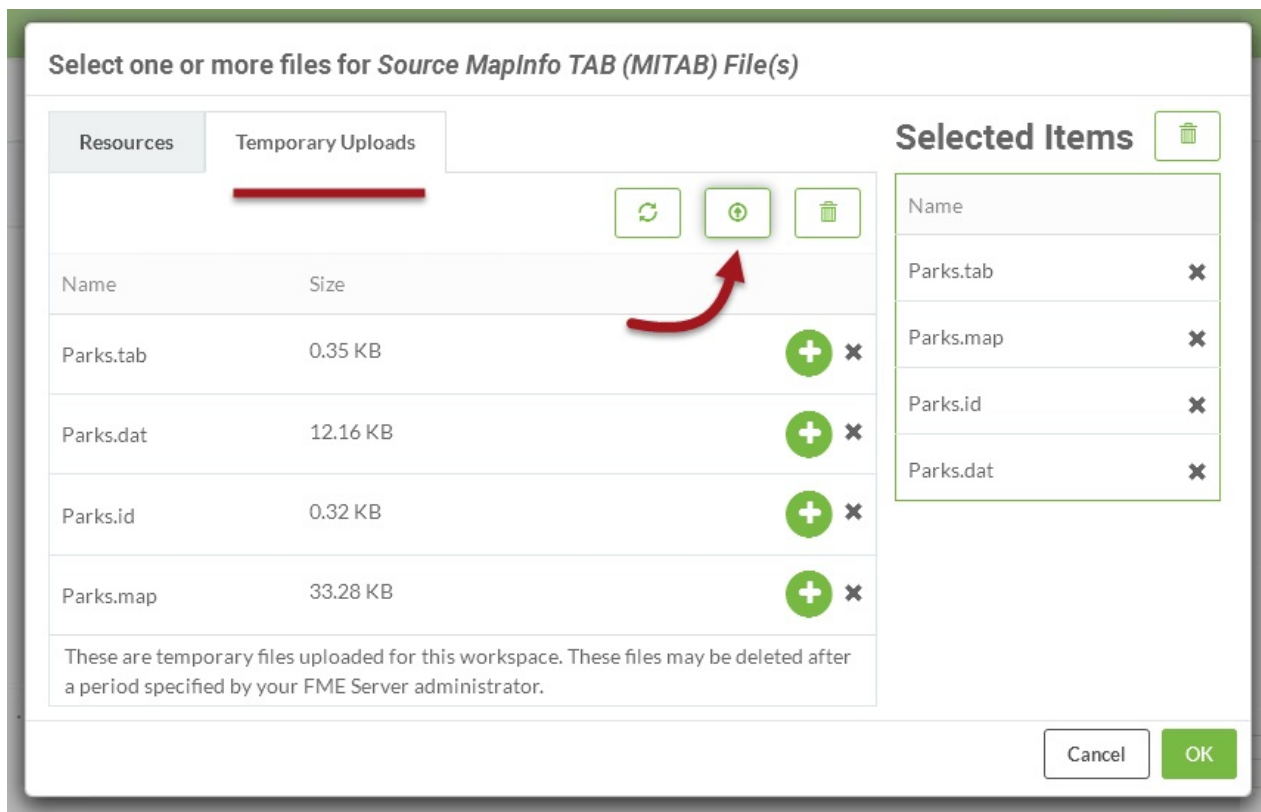
Of course, in this scenario publishing the data with the workspace does not make much sense. It is better if the user uploads data at run time.

Provided the source dataset is a published parameter, they can do this very easily in the Run Workspace page of the FME Server interface by clicking the browse button:

Published Parameters

| | | |
|------------------------------------|-------------------------------------|-----|
| Source MapInfo TAB (MITAB) File(s) | C:\FMEData2017\Data\Parks\Parks.tab | ... |
| Feature Types to Read | | |

This opens a dialog with a button (highlighted) to use for uploading files (here a Parks dataset):

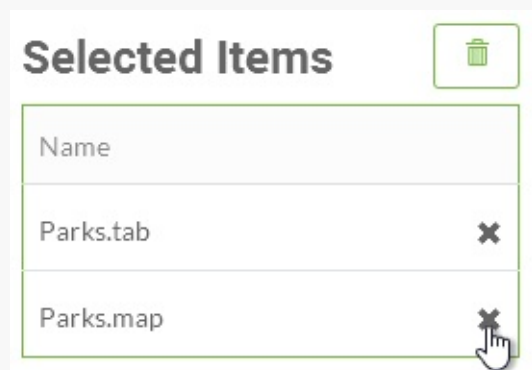


Any file uploaded is automatically *selected* for translation. At that point simply click OK. Now when the workspace runs, the selected user-uploaded data gets translated.

Police Chief Webb-Mapp says...

In the above example, the source dataset is in a format (MapInfo TAB) that consists of several files.

*Although all files need to be uploaded, only the TAB file itself needs to be **selected**.*



The method to use is to deselect files under the section Selected Items; here the .dat, .id, and .map files.

The files are still available, but FME won't treat each of them as a separate dataset, which is what would happen if they all remained selected.

Cautions and Limitations

There are a number of cautions and limitations to be concerned about when data is uploaded for translation by the end-user:

- Giving the user upload ability is risky because their dataset's schema has to match the workspace's schema definition, otherwise the translation will fail with unexpected input. Alternatively - as above - a dynamic (and maybe generic) translation could be used to avoid such issues.
- Data uploaded by the user is only temporarily available. The System Cleanup page shows us that such files are (by default) deleted when they become more than 24 hours old. User uploads are not a long-term solution.
- Data uploaded by the user is, in theory, also accessible through the Resources page (more on that to come). However, in practice, it's in an obscure location where an end user would not be expected to find it. For that reason, temporary data should be considered inaccessible by any other means and unavailable for use by any other workspace.

| Exercise 4 Daily Database Updates: Publishing Data | |
|---|---|
| Data | Neighborhoods (KML) Election Voting (GML) |
| Overall Goal | Create a workspace to read and process departmental data and publish it to FME Server |
| Demonstrates | Publishing source data and uploading temporary datasets |
| Start Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\Basics-Ex4-Begin.fmw |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\Basics-Ex4-Complete.fmw |

For the exercises in this chapter, you are a technical analyst in the GIS department of your local city.

You have already (Exercises 1, 2, and 3) created a workspace to carry out a translation, published it to FME Server, ran it to confirm it works, shared the repository, and set the workspace to run on a schedule.

Now you have a task to create a new workspace. One of the datasets it uses is the same as in the previous exercise, so we will try to have this second workspace use the data belonging to the first.

Sister Intuitive says...

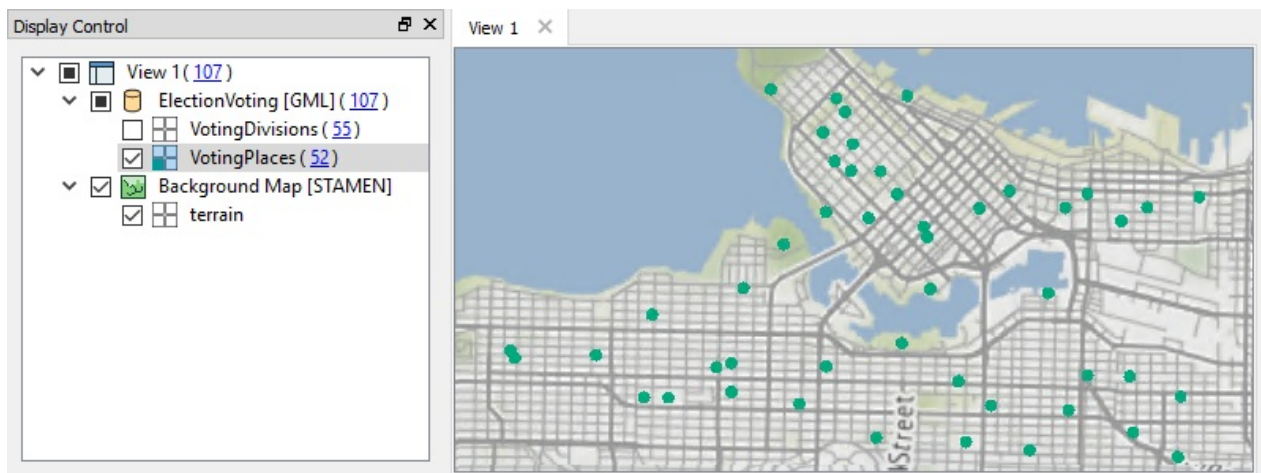
*If you have lots of experience with FME Workbench - **and if your instructor agrees** - simply open the end workspace listed in the header above and skip to step 7*

1) Inspect Source Data

The first task in any new project is to inspect the source data, so let's do that. Use the FME Data Inspector to open these two datasets:

| | |
|-----------------------|--|
| Reader Format | GML (Geography Markup Language) |
| Reader Dataset | C:\FMEData2017\Data\Elections\ElectionVoting.gml |

You can turn off the layer of VotingDivisions. All we are interested in for this exercise are the point features designated as VotingPlaces:



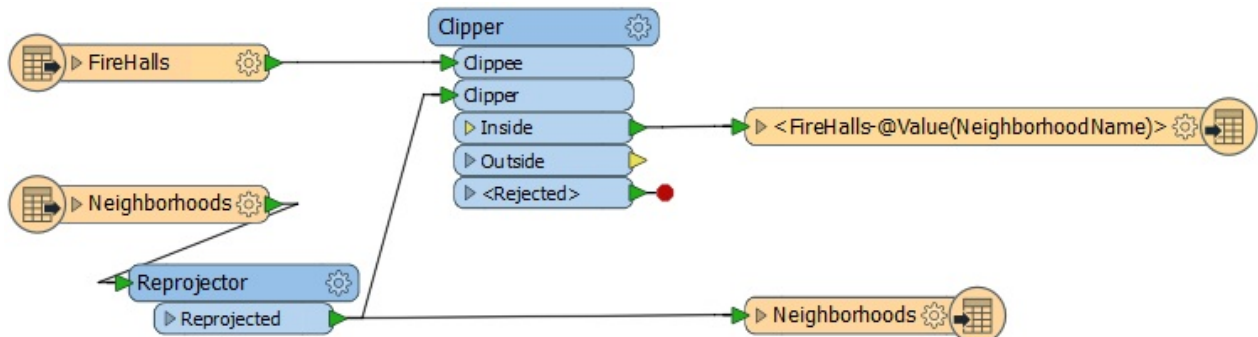
Map tiles by [Stamen Design](#), under [CC-BY-3.0](#). Data by [OpenStreetMap](#), under [CC-BY-SA](#).

2) Create Workspace

Open the starting workspace listed above.

You might notice that it's a copy of our previous project, since the requirements for this workspace are so similar. If you do choose to just carry on working in that workspace, be sure to save it under a different name - otherwise the data we will publish will not work for this exercise.

The workspace looks like this:

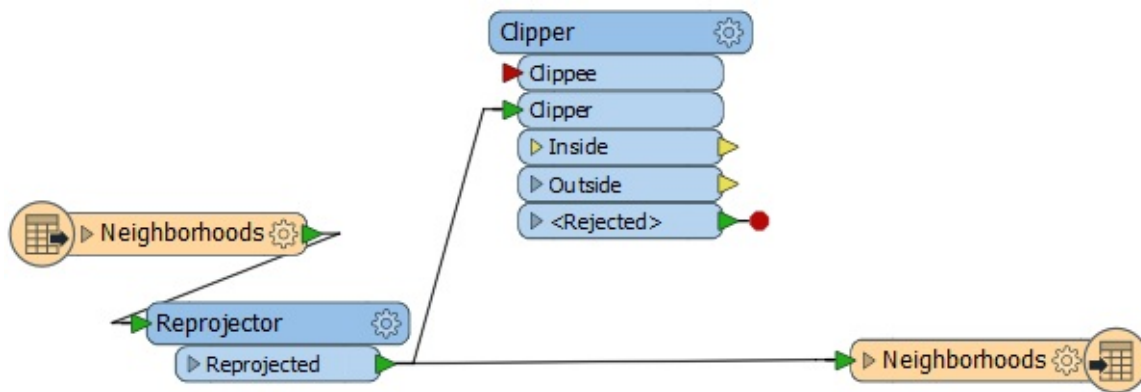


3) Remove Firehalls

For this project we need to process election data instead of firehalls, so firstly delete the writer feature type for the firehalls, and then the reader feature type.

When you delete the reader feature type, you will be asked if you wish to delete the entire reader. We could reuse it but, for the sake of simplicity, click yes.

The workspace now looks like this:

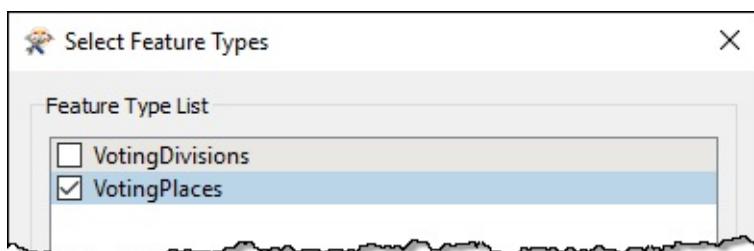


4) Add VotingPlaces

Now select Readers > Add Reader to start adding a reader to the workspace. When prompted, enter the following details for the VotingPlaces data:

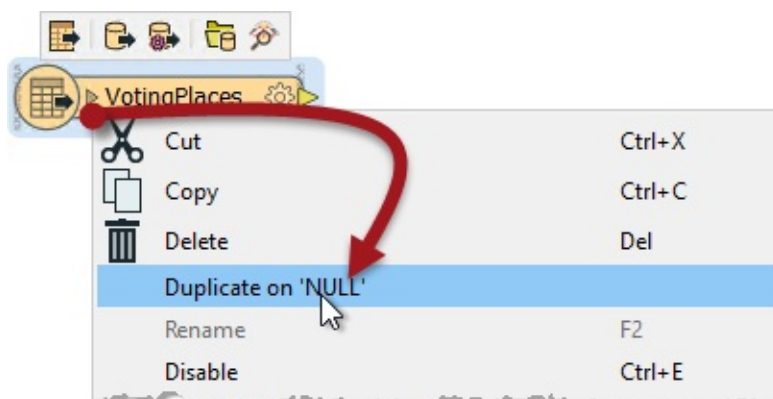
| | |
|-----------------------|---|
| Reader Format | GML (Geography Markup Language) |
| Reader Dataset | C:\FMEDData2017\Data\Elections\ElectionVoting.gml |

Click OK to add the Reader to the workspace. When prompted only select the VotingPlaces feature type, not VotingDivisions:

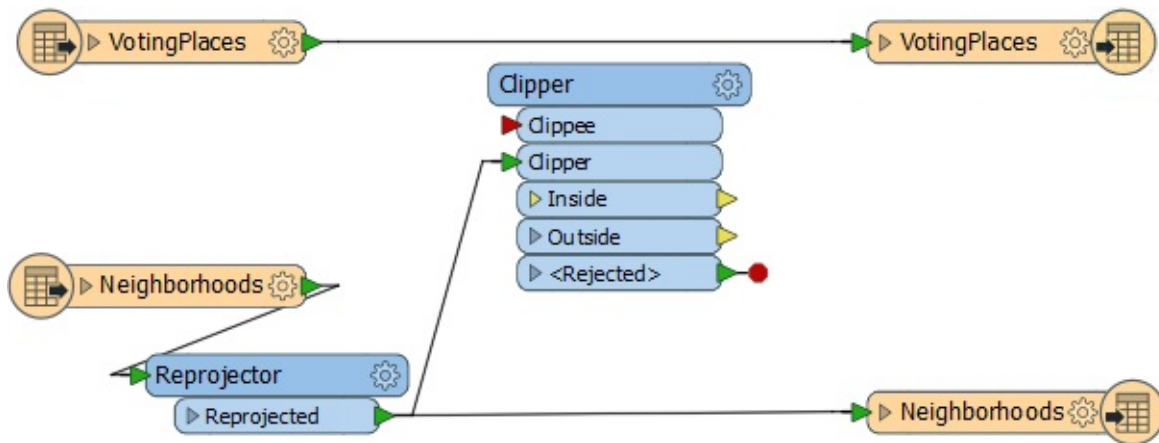


5) Add VotingPlaces to Writer

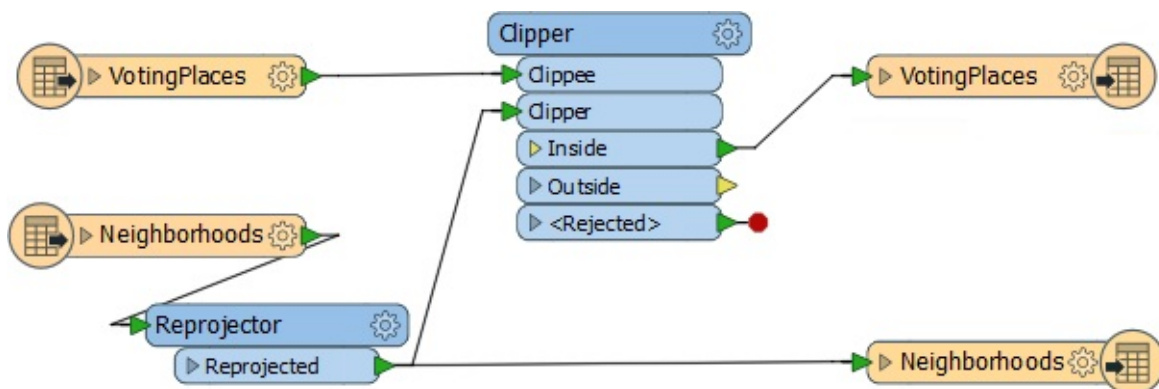
To add VotingPlaces to the writer, right-click the newly placed reader feature type and choose Duplicate on 'NULL':



There will now be a reader and writer feature type for the VotingPlaces dataset:



Change the connections to pass the VotingPlaces data through the Clipper transformer just as the FireHalls used to be:



6) Set VotingPlaces Feature Type Name

Finally, as with the FireHalls, let's set the Feature Type Name for the VotingPlaces writer feature type.

Inspect its parameters and under Feature Type Name either enter:

```
VotingPlaces-@Value(NeighborhoodName)
```

...or click the dropdown and use the text editor dialog to enter that value. This will cause voting places in each different neighborhood to be written to a different table/layer.

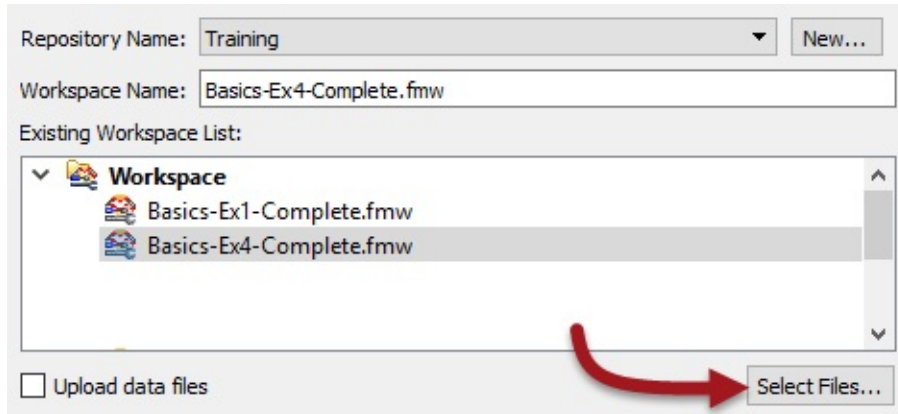
Save the workspace. As already mentioned, make sure it has a different name to the first project.

7) Publish to Server

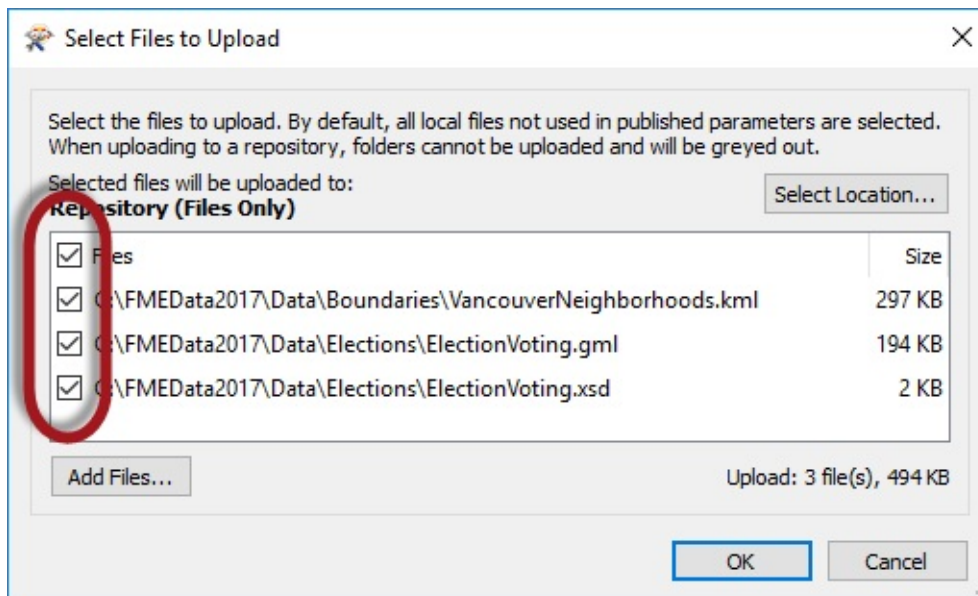
Publish the workspace to FME Server. This time you can simply choose the previously created FME Server connection, rather than having to enter parameters all over again.

For the repository select the previously created Training repository and enter a name for the workspace if it doesn't already have one.

This time, instead of simply checking the box to upload all the data files, click the Select Files button:



This dialog lists the files we are about to publish to the repository with the workspace. Technically the VancouverNeighborhoods dataset was already published to the repository with the previous workspace, but it's not very good practice to try and re-use data this way (even though we could) so place a check mark against all files and click OK:



In the final dialog of the publishing wizard, once again choose the Job Submitter as the web service to register the workspace against.

8) Examine Files

If you have access to the FME Server computer itself, open a file browser and browse to the location that repository data is stored. Here it is C:\ProgramData\Safe Software\FME Server\repositories\Training:



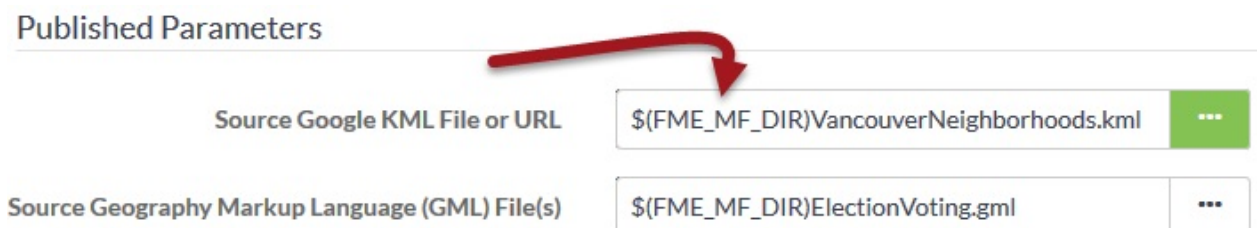
| Name | Date modified | Type | Size |
|---------------------|--------------------|-------------|------|
| Basics-Ex1-Complete | 4/24/2017 10:06 AM | File folder | |
| Basics-Ex4-Complete | 4/25/2017 8:17 AM | File folder | |

You'll see that each workspace is saved to a separate folder. If you inspect the contents of a folder you'll see the uploaded datasets within it.

This is how a workspace has access to files published with it. It can also, with some manual effort, access files stored with another workspace in the same repository.

9) Run Workspace

Locate and run the workspace. In the Run dialog notice that the published parameters denoting the source data include an FME environment variable, FME_MF_DIR:



Published Parameters

| | | |
|--|--|-----|
| Source Google KML File or URL | \$(FME_MF_DIR)VancouverNeighborhoods.kml | ... |
| Source Geography Markup Language (GML) File(s) | \$(FME_MF_DIR)ElectionVoting.gml | ... |

This variable tells FME to look in the same folder as the workspace for the source data files. As you can see, it isn't particularly user friendly to handle data in this way, even though the workspace will run just fine.

9) Upload Temporary Data

Now let's pretend that the layer of VotingPlaces data has changed in some way. You can simulate that by simply opening a file browser and making a copy of the GML file.

For example, rename C:\FMEData2017\Data\Elections\ElectionVoting.gml to NewElectionVoting.gml

***NB:** You don't also have to copy ElectionVoting.xsd - it's fine to use that schema file for the new GML dataset.*

Now, in the FME Server web interface, log out of the admin account and log in as a user (user/user).

So, as a user we wish to run the workspace with the new data. We can't publish the data because the user account doesn't have permission to write to that repository; and in any case, since the workspace hasn't changed in any way, we shouldn't have to go through the publish process.

So, click Run Workspace and select the newly published workspace in the Training repository. However, to use the new dataset, click the browse button to the right of the Source GML prompt:

Published Parameters




Source Google KML File or URL ...

Source Geography Markup Language (GML) File(s) ...



In the dialog that opens, click the Temporary Uploads tab and then on the Upload File button:

Resources **Temporary Uploads**




  





Upload File

| Name | Size |
|--|------|
| These are temporary files uploaded for this workspace. These files may be deleted after a period specified by your FME Server administrator. | |


Select both the files NewElectionVoting.gml and ElectionVoting.xsd and click Open to upload them. Now - back in the prior dialog - click the X button to deselect the xsd file:



Resources **Temporary Uploads**


  

| Name | Size | |
|-----------------------|-----------|---|
| ElectionVoting.xsd | 2.45 KB |   |
| NewElectionVoting.gml | 199.42 KB |   |

These are temporary files uploaded for this workspace. These files may be deleted after a period specified by your FME Server administrator.

Selected Items 

| Name | |
|-----------------------|---|
| NewElectionVoting.gml |  |
| ElectionVoting.xsd |  |



The file needs to exist, but it doesn't need to be selected. Now click OK and then click the Run button.

The workspace will now run to completion using the uploaded dataset.

However - and this is the important part - this was only a temporary upload. The workspace can be re-run immediately and the data will still appear in the temporary upload section, but it is not a permanent solution. The data is likely to be cleaned up automatically within 24 hours.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Update a workspace with a new reader and a new writer feature type*
- *Publish a workspace to FME Server and include source data*
- *Locate source data on the FME Server filesystem*
- *Select a source dataset to upload temporarily at run-time*

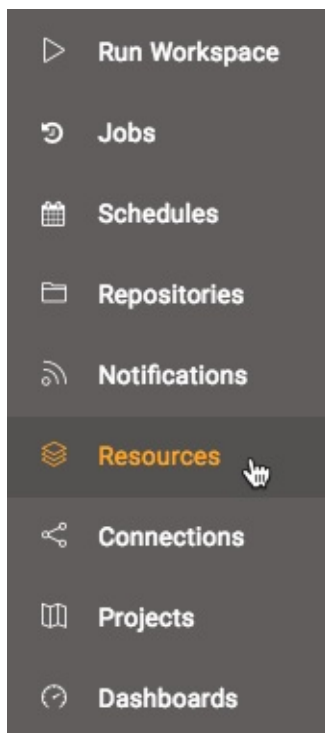
The Resources File System

The final method of managing source data in FME Server is to use the system of tools called Resources.

What are Resources?

"Resources" is an inbuilt file management system that allows data (and other files) to be published to an FME Server instance and used within all Server operations.

FME Server has a page for managing Resources, accessed through the main menu:



The Resources web page looks like this:

Resources

| <input type="button" value="New"/> <input type="button" value="Duplicate"/> <input type="button" value="Edit"/> <input type="button" value="Remove"/> | | | |
|---|--|-------|--|
| <input type="checkbox"/> | Name | owner | |
| <input type="checkbox"/> | Backup This shared resource is the Server backup directory | admin | |
| <input type="checkbox"/> | Dashboards This shared resource is the Server dashboards directory | admin | |
| <input type="checkbox"/> | Data This shared resource is the Server data directory | admin | |
| <input type="checkbox"/> | Engine This shared resource is the Server engine directory | admin | |
| <input type="checkbox"/> | Logs This shared resource is the Server logs directory | admin | |
| <input type="checkbox"/> | System This shared resource is the Server system directory | admin | |
| <input type="checkbox"/> | Temp This shared resource is the Server temp directory | admin | |

As you can see, the Resources filesystem is set up with a number of default folders in which files can be stored. For datasets the most logical to use is the Data folder. Although there are a number of different folders data can be stored in, the Data folder is the most logical location:

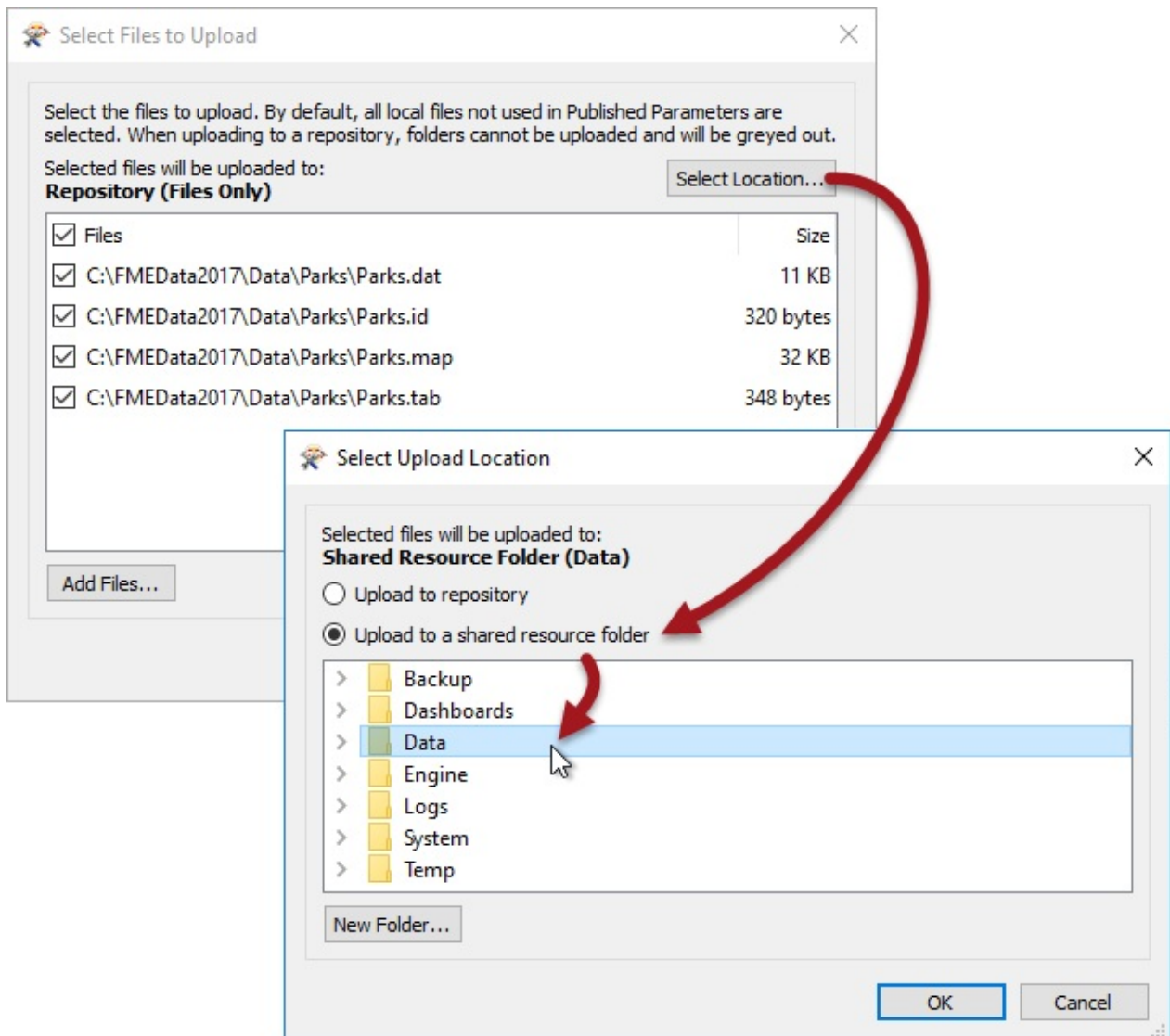
| Resources ⓘ | | | |
|---|------------|-----------|--------------------|
| <input type="button" value="New Folder"/> <input type="button" value="Download"/> <input type="button" value="Delete"/> <input type="button" value="Copy"/> <input type="button" value="Move"/> <input type="button" value="View File"/> <input type="button" value="Properties"/> <input type="button" value="Q"/> <input type="checkbox"/> Overwrite existing files <input type="button" value="Upload"/> | | | |
| Home > Data | | | |
| <input type="checkbox"/> | Name | Size | Date |
| <input type="checkbox"/> | Training | | 2017-1-31 09:54:17 |
| <input type="checkbox"/> | Parks.gml | 211.29 KB | 2017-2-6 08:55:55 |
| <input type="checkbox"/> | Parks.xsd | 1.89 KB | 2017-2-6 08:55:55 |
| <input type="checkbox"/> | readme.txt | 4.08 KB | 2017-1-31 09:54:36 |

Above is the data folder containing several files. Notice the buttons available to carry out actions such as upload, copy, delete, or move files (or folders).

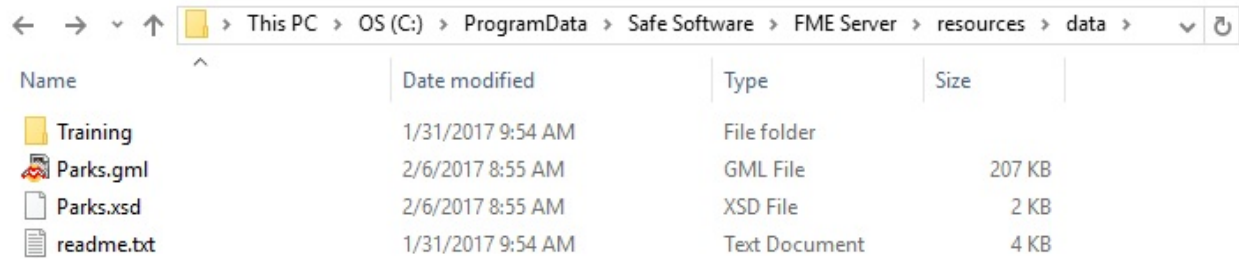
Other Upload Methods

Besides the web interface, there are other ways of getting data into the Resources filesystem.

Firstly, the FME Server publishing wizard in FME Workbench allows this. Where the default method is to select the files and upload them to the same repository as the workspace, it is permitted to change the location to the resources filesystem:

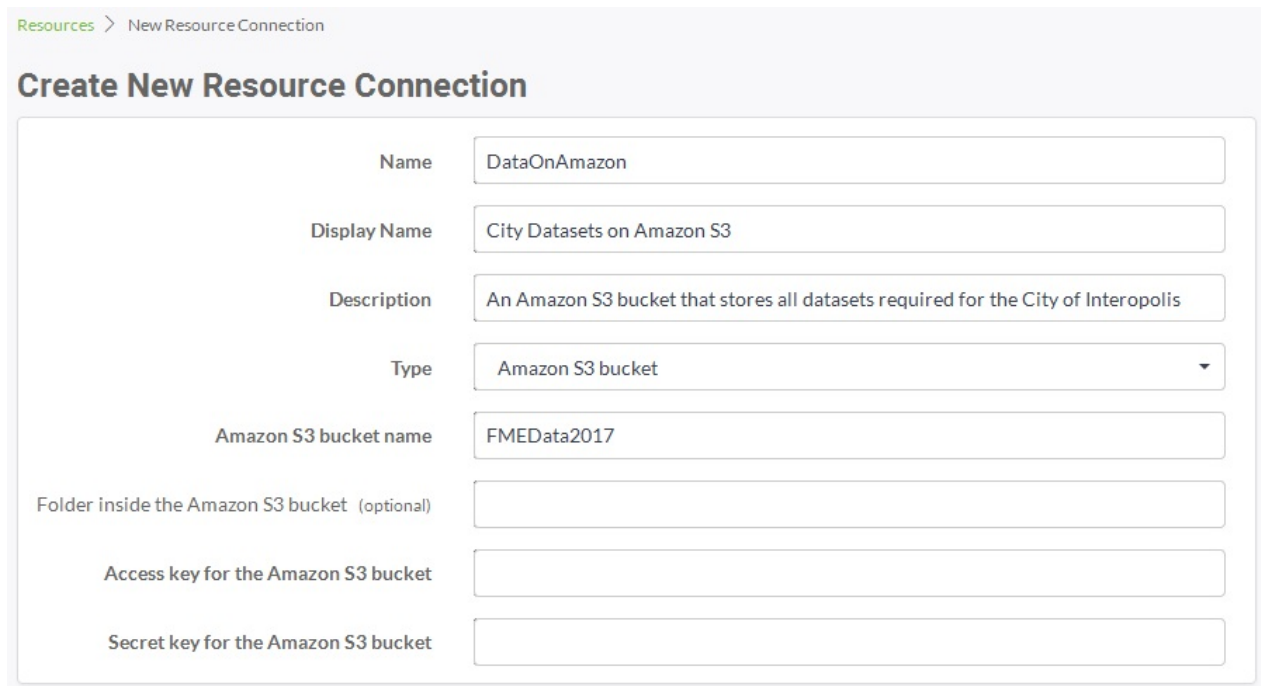


Alternatively, FME Server resources actually exists on the operating system's filesystem, meaning the data can be copied there directly. The default location (on a Windows operating system) is C:\ProgramData\Safe Software\FME Server\resources:



| Name | Date modified | Type | Size |
|------------|-------------------|---------------|--------|
| Training | 1/31/2017 9:54 AM | File folder | |
| Parks.gml | 2/6/2017 8:55 AM | GML File | 207 KB |
| Parks.xsd | 2/6/2017 8:55 AM | XSD File | 2 KB |
| readme.txt | 1/31/2017 9:54 AM | Text Document | 4 KB |

Finally, clicking the New button in the main Resources page allows a connection to be made directly to an Amazon S3 filesystem:



Resources > New Resource Connection

Create New Resource Connection

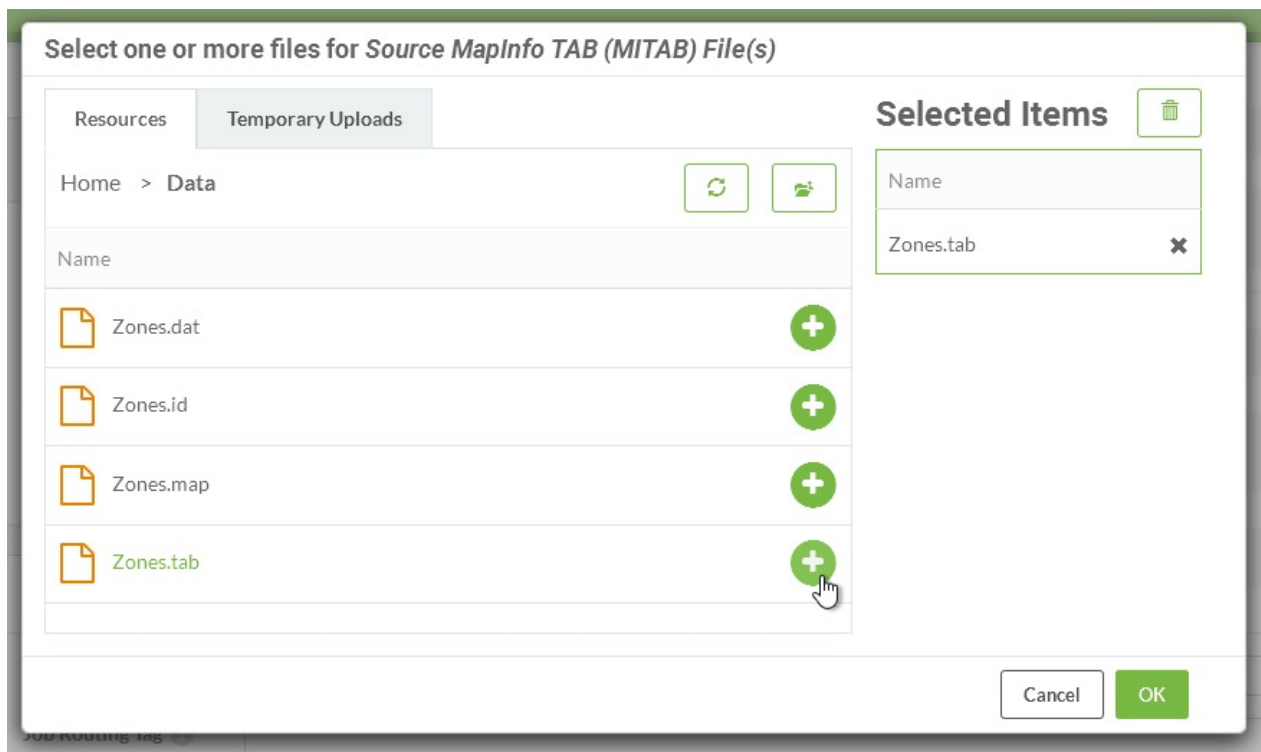
| | |
|---|--|
| Name | <input type="text" value="DataOnAmazon"/> |
| Display Name | <input type="text" value="City Datasets on Amazon S3"/> |
| Description | <input type="text" value="An Amazon S3 bucket that stores all datasets required for the City of Interopolis"/> |
| Type | <input type="text" value="Amazon S3 bucket"/> |
| Amazon S3 bucket name | <input type="text" value="FMEDData2017"/> |
| Folder inside the Amazon S3 bucket (optional) | <input type="text"/> |
| Access key for the Amazon S3 bucket | <input type="text"/> |
| Secret key for the Amazon S3 bucket | <input type="text"/> |

This allows data stored in S3 buckets to be used as the source for a translation on FME Server.

Using Uploaded Data

Using Resources data in a translation is simply a case of selecting it from that folder where prompted. All prompts for data will allow selection of files from a Resources folder.

For example, a user has uploaded a MapInfo Zoning dataset to the Resources data folder. Provided the source dataset is a published parameter, when the workspace is run the user is able to select data from the Resources folders, like so:



In fact, it's even possible to set the output data folder to be a Resources folder too:

Published Parameters

| | | |
|--|---------------------------------------|-----|
| Source MapInfo TAB (MITAB) File(s) | \$(FME_SHAREDRESOURCE_DATA)/Zones.tab | ... |
| Destination Geography Markup Language (GML) File | \$(FME_SHAREDRESOURCE_DATA)/Zones.gml | ... |

Benefits for Data Management

There are several benefits to using the Resources filesystem as a data storage tool:

- Data can be used by any workspace, without having to upload it every time
- Data can be stored locally (to the Server engines), even when access to the operating system's filesystem is restricted
- A Resources folder can be mapped and shared among many users as a physical drive
- A Resources folder is a more permanent solution. Data is not removed by automated system cleaning tools

Miss Vector says...

Not just one, but two questions this time!

Firstly: I copy a workspace into a resources folder using the upload tool. What then?

- 1. I can run it by browsing the resources, selecting the workspace, and clicking run*
- 2. I can run it through the Manage > Workspaces menu tools*
- 3. I can run it by calling it with the FMEServerJobSubmitter transformer in FME Desktop*
- 4. I can't run it because it's not properly published to a repository*

Secondly: Uploading an entire folder of files come with what restriction?

- 1. Folder upload only works on certain web browsers*
- 2. Folder upload requires the folders to be zipped into a single file*
- 3. Folder upload only works on Windows C: drive (not D:, E:, etc)*
- 4. Folder upload requires FME Desktop to be installed on the computer being uploaded from*

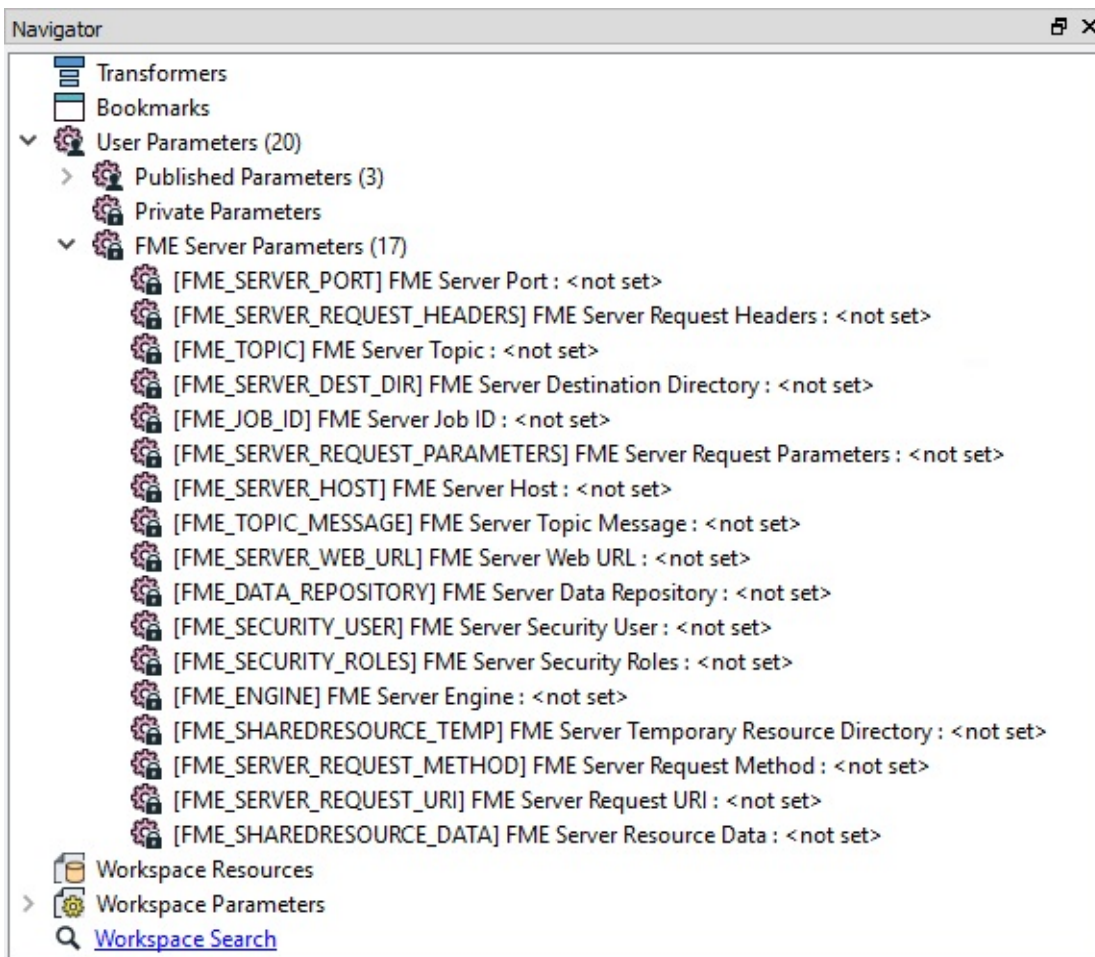
Authoring for the Resources System

Using the FME Server web interface it's simple to select data from the resources folder at run-time. However, in some cases the author will want to read data from a resources folder without the end-user having to select it.

To do this requires the use of an FME parameter to define the data as coming from the resources folders.

FME Parameters for Server

In FME Workbench the Navigator window has a section called user parameters. You might have noticed that one part of this is a list of FME Server-specific parameters:



The uses of these are many and varied; for example FME_SECURITY_USER returns the name of the user running the workspace, and could be used to either write to a custom log or perhaps filter data in different ways in the workspace based on the specific user.

FME_TOPIC would return the name of the notification topic (if any) that invoked the workspace.

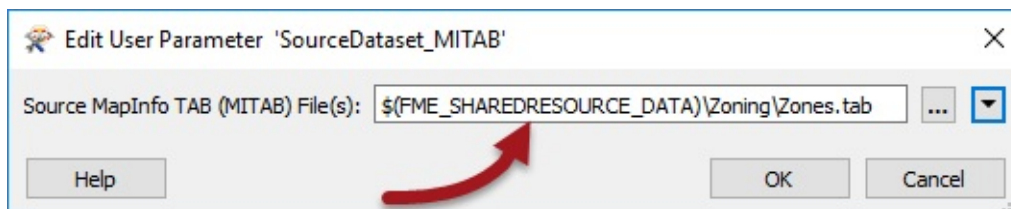
However, when authoring for resources data, the most useful parameter is FME_SHAREDRESOURCE_DATA

WARNING

A common factor to all these parameters is that they only have an effect when the workspace is run on FME Server. If the workspace is run on FME Desktop, it won't return a value. Therefore to test on Desktop a workspace containing such a parameter requires you to provide a dummy value.

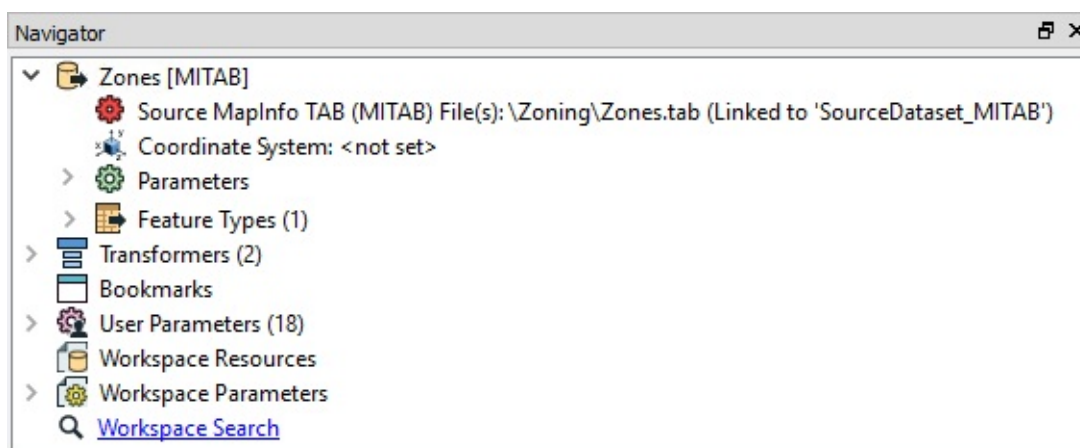
FME_SHAREDRESOURCE_DATA

What the FME_SHAREDRESOURCE_DATA parameter does is return the path of the shared resource data. When authoring a workspace to read data directly from the resource folder, you would normally create the workspace using a local copy of the data, and then update the source dataset field to include the FME Server parameter:



Updating the field can be done directly by typing into the dialog, but is easier to achieve by clicking the drop-down arrow and choosing to use the Text Editor.

The entry in the Navigator window now looks like this:



Although the parameter is colored red in Desktop, when the workspace is run on Server the parameter is replaced by the actual path and the data is read as expected.

WARNING

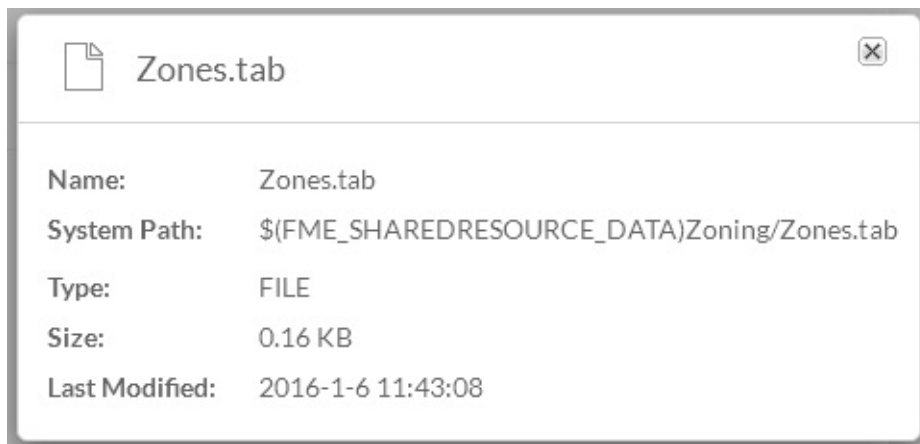
It's important to remember (for example notice in the screenshots above) that the Server parameter `FME_SHAREDRESOURCE_DATA` includes the 'Data' folder in its path.

For example, I use `FME_SHAREDRESOURCE_DATA\Zoning\Zones.tab` not `FME_SHAREDRESOURCE_DATA\Data\Zoning\Zones.tab`

Shortcut to Resource Paths

Rather than setting the path in Workbench before uploading the data, you can also copy the path for an uploaded dataset and then paste that directly into Workbench.

The path for a shared resource is obtained by examining its properties in the FME Server Resources pages:



This path can then be copied and pasted into a workspace in order to reference that dataset directly without manually entering it. Of course, this does require that the data has already been uploaded to FME Server, and isn't going to be uploaded with the workspace when it is published!

Miss Vector says...

So I can make my workspace read specific data from the resources folders - but how do I stop the end-user from being able to change that?

- 1. Remove their security permissions for the Job Submitter service*
- 2. Remove their security permissions for the Resources folders*
- 3. Make the source dataset parameter optional for that Reader*
- 4. Delete the published parameter for that source dataset from the workspace*

| Exercise 5 Daily Database Updates: Using Resources | |
|--|---|
| Data | Neighborhoods (KML) Election Voting (GML) |
| Overall Goal | Create a workspace to read and process departmental data and publish it to FME Server |
| Demonstrates | Uploading data to a resources folder and authoring a workspace to make use of it |
| Start Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\Basics-Ex5-Begin.fmw |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\Basics-Ex5-Complete.fmw |

For the exercises in this chapter, you are a technical analyst in the GIS department of your local city.

You have already (Exercise 4) created a workspace to carry out a translation, and published it to FME Server; both with data and using data uploaded temporarily.

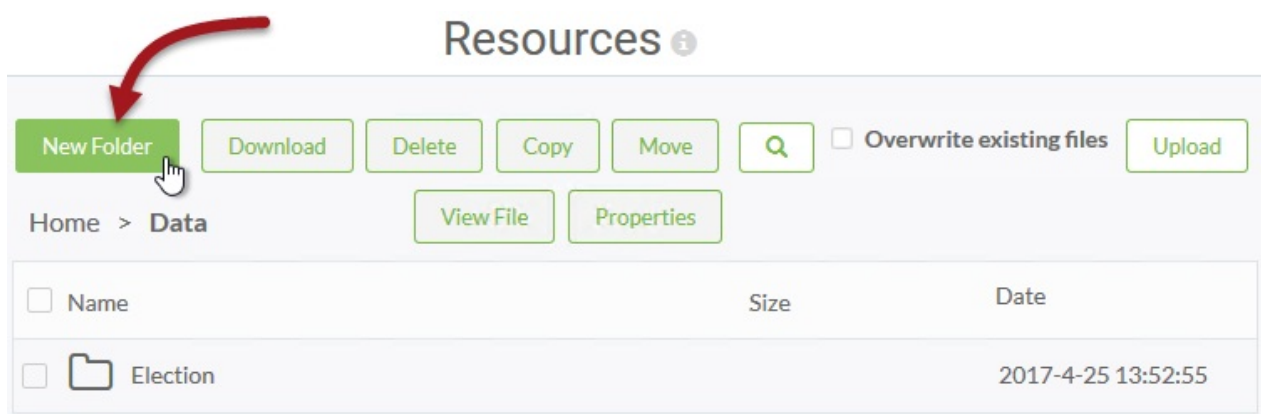
However, such data management tools are not particularly suited to a long term project, so the task here is to upgrade the workspaces to use datasets stored in a Resources folder. There we can store source data and write destination data.

1) Open FME Server Web Interface

Log in to the FME Server web interface using an administrator account (such as admin/admin). Click Resources on the menubar to navigate to the resources management pages.

2) Create Folder

In most cases data should be stored under the Data folder, so click on Data in the Resources dialog to open that folder. To avoid mixing datasets our data should go into its own subfolder. So click on the New Folder button and create a folder called Election:






Next click on the Election folder and within there create subfolders called Input and Output.

3) Upload Source Datasets

Browse to the Input folder and click the upload button. Upload the source datasets for the current translation:

| | |
|------------------------|---|
| Reader Datasets | C:\FMEData2017\Data\Elections\ElectionVoting.gml C:\FMEData2017\Data\Elections\ElectionVoting.xsd C:\FMEData2017\Data\Boundaries\VancouverNeighborhoods.kml |
|------------------------|---|

| Upload(s) Completed | | |
|---|-----------|--------------------|
| <input type="checkbox"/> Name | Size | Date |
| <input type="checkbox"/>  ElectionVoting.gml | 199.42 KB | 2017-4-25 14:00:59 |
| <input type="checkbox"/>  ElectionVoting.xsd | 2.45 KB | 2017-4-25 14:00:59 |
| <input type="checkbox"/>  VancouverNeighborhoods.kml | 304.79 KB | 2017-4-25 14:01:09 |

So we now have both source datasets and a folder to write the output data to.

4) Add Writer

Up until now all of our workspaces have had only a NULL (dummy) writer. Now we know about Resources we can add a proper writer and point its output to the Resources Output folder.

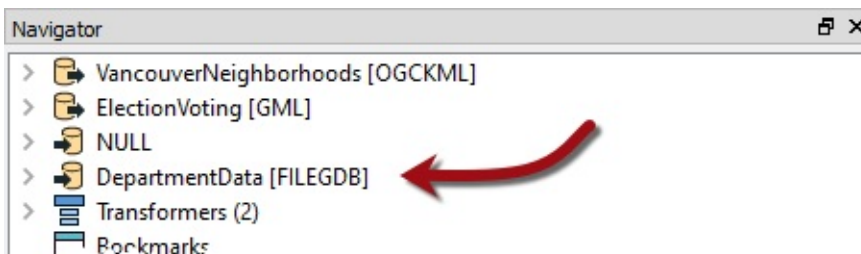
So, open the workspace listed above in FME Workbench and then select Writers > Add Writer on the menubar and set up a new writer with the following parameters:

| | |
|--|--|
| Writer Format | Esri Geodatabase (File Geodb Open API) |
| Writer Dataset | C:\FMEDData2017\Output\Training\DepartmentData.gdb |
| Feature Class or Table Definition | None (Advanced) |

.1 UPDATE

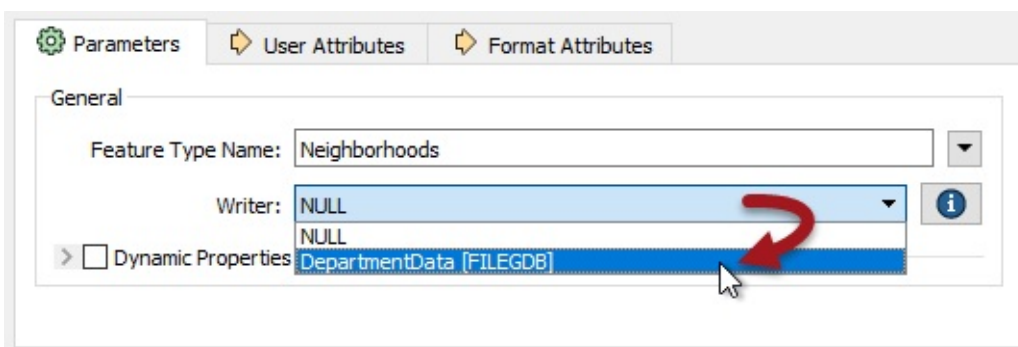
*Note that Esri Geodatabase format was renamed from *File Geodb API* to *File Geodb **Open** API* in FME 2017.1!*

The reason we want to add no feature types is that we can move the existing ones from the NULL writer. So when you click OK the workspace will look no different, but there will be a new writer in the Navigator window:

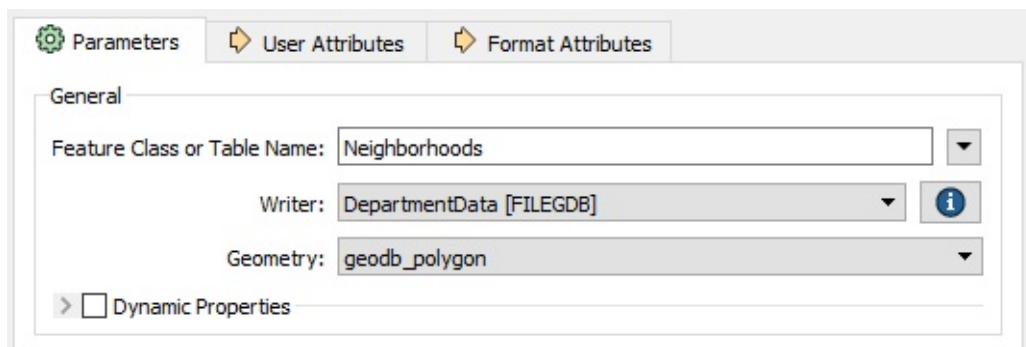


5) Move Feature Types

Inspect the parameters dialog for each writer feature type in turn. For each type move it from the NULL writer to the FILEGDB writer, like so:



This will expose a number of extra parameters. The key one to set is Geometry. For the Neighborhoods they should be set to geodb_polygon:

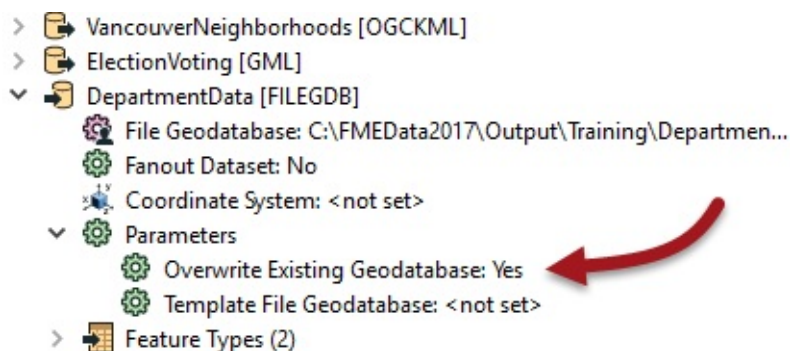


For the VotingPlaces feature type the Geometry parameter should be set to geodb_point.

Now the two feature types belong to the Geodatabase writer, and the NULL writer can be deleted from the Navigator window if you wish.

6) Set Geodatabase Parameter

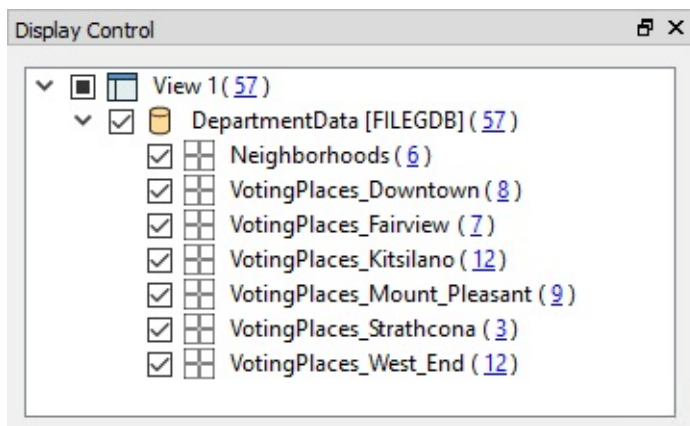
One (very quick) last thing to change: locate the Geodatabase writer in the Navigator window and expand its list of parameters. Double-click the parameter labelled Overwrite Existing Geodatabase and set it to Yes:



This ensures we aren't continually adding data to the same dataset if we run the workspace more than once.

7) Run Workspace

Test run the workspace in FME Desktop. Inspect the output. You should find the output is a Geodatabase containing seven tables (the Neighborhoods table and a separate table for each set of voting places).



8) Publish and Run Workspace

Publish the workspace to FME Server. Be sure not to check the button to upload any data. Register the workspace against the Job Submitter service as usual.

Return to the FME Server web interface. Locate the workspace under the Run Workspace dialog. Notice how the dataset paths are all hard-coded to the original file locations:

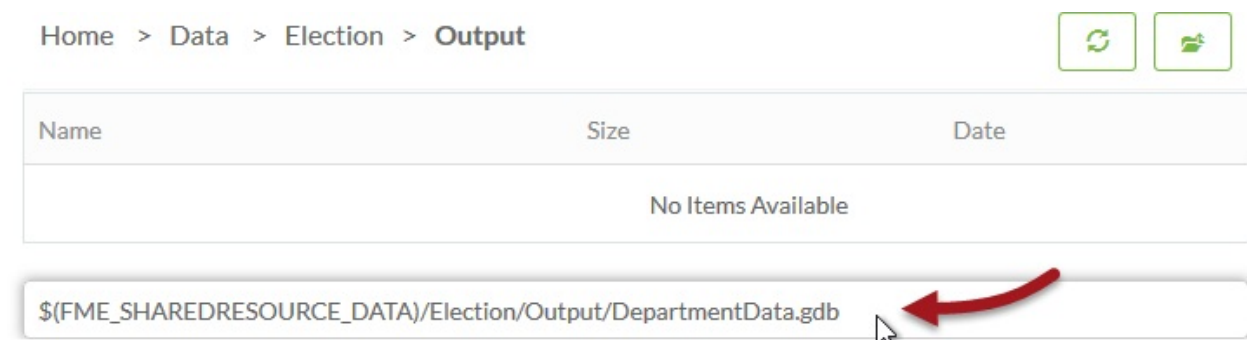
Published Parameters

| | | |
|--|---|-----|
| Source Google KML File or URL | C:\FMEData2017\Data\Boundaries\VancouverNeighborhoods.kml | ... |
| Source Geography Markup Language (GML) File(s) | C:\FMEData2017\Data\Elections\ElectionVoting.gml | ... |
| File Geodatabase | C:\FMEData2017\Output\Training\DepartmentData.gdb | ... |

Obviously this will be of no use if the Server does not have access to those files. However, because we already uploaded them to the Resources folders we can use those files.

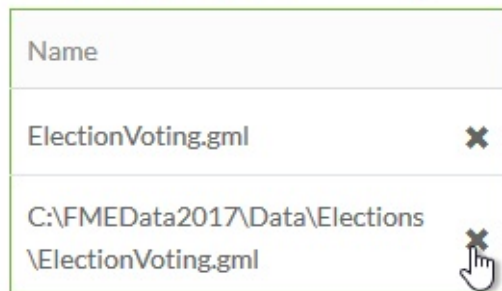
So, for each file, click the browse button, browse to the appropriate subfolder in the Resources folder, and select/set the file location. For the Geodatabase output location you'll need to type the file name manually:

```
$(FME_SHAREDRESOURCE_DATA)/Election/Output/DepartmentData.gdb
```



Remember to remove any existing references to the incorrect files:

Selected Items



| Name | |
|--|---|
| ElectionVoting.gml | X |
| C:\FMEData2017\Data\Elections \ElectionVoting.gml | X |

Now when the workspace is completed a Geodatabase file should appear in the folder Resources\Data\Election\Output:



| Home > Data > Election > Output | | |
|---------------------------------|--------------------|--------------------|
| <input type="checkbox"/> | Name | Date |
| <input type="checkbox"/> | DepartmentData.gdb | 2017-4-25 15:03:22 |

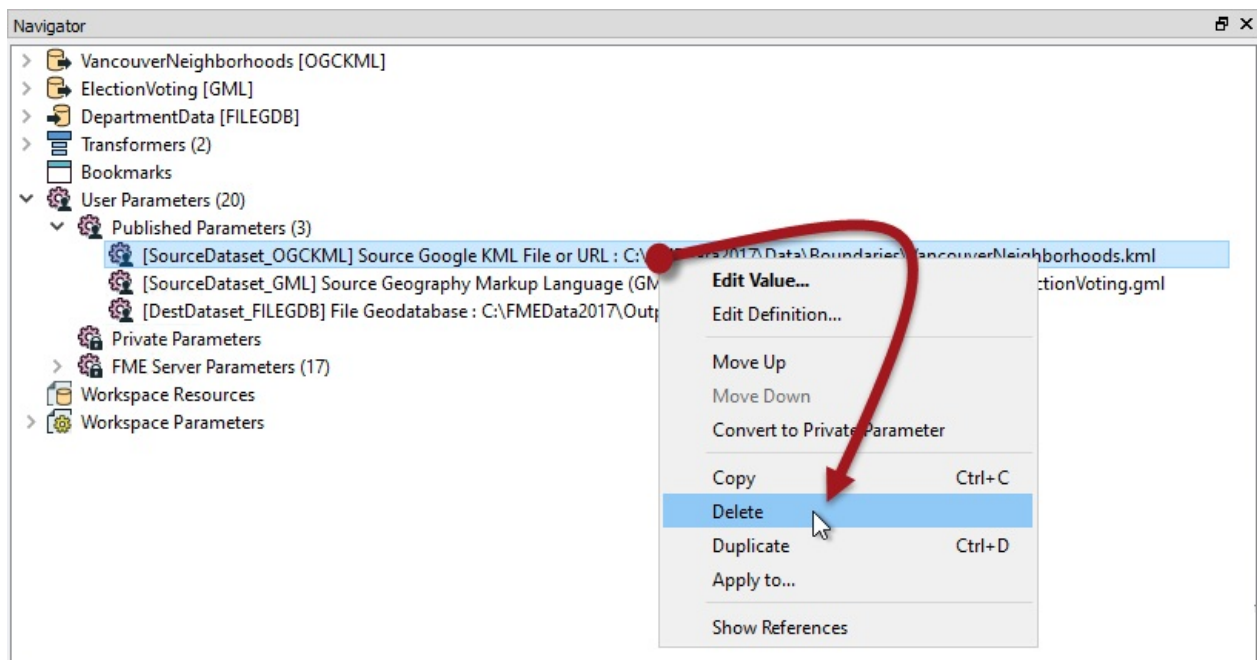
9) Apply FME Server Parameter

Although the workspace ran correctly, and used the data in the resources folder, that's only because we selected that data at run time. It is not a permanent feature of the workspace.

It would be much better if the workspace was programmed to look into the resources folders automatically.

So, return to the workspace in FME Workbench.

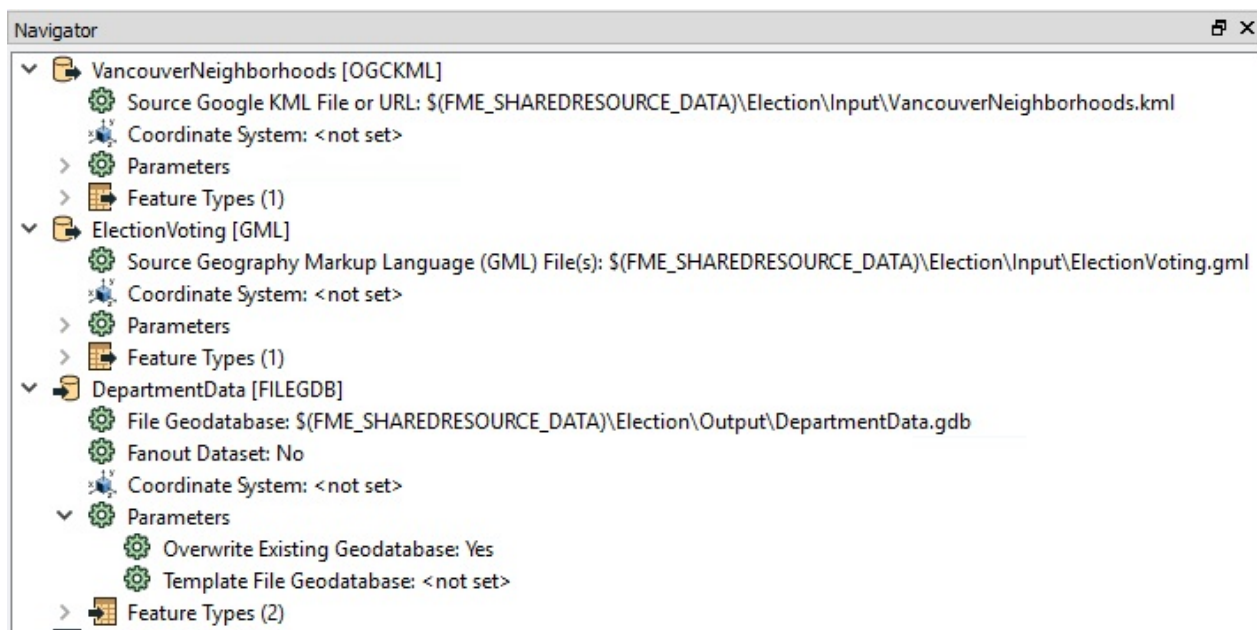
If we do set the workspace to read from the resources folders, we don't want to give users the chance to change that. So in the Navigator window locate the three parameters for source and destination datasets and delete them:



10) Set Source/Destination Parameters

Now, in turn, locate the source and destination dataset parameters for the two readers and one writer. Double-click each in turn and change them to:

| | |
|--------------------|---|
| KML Reader | \$(FME_SHAREDRESOURCE_DATA)\Election\Input\VancouverNeighborhoods.kml |
| GML Reader | \$(FME_SHAREDRESOURCE_DATA)\Election\Input\ElectionVoting.gml |
| Geodatabase Writer | \$(FME_SHAREDRESOURCE_DATA)\Election\Output\DepartmentData.gdb |



Save the workspace and publish it back to FME Server.

Sister Intuitive says...

This time you won't be able to test-run the workspace in FME Workbench, because it won't recognize the shared resource parameter. Only FME Server will return a value for that parameter.

11) Re-Run Workspace

Now run the workspace on FME Server. Be sure to use the Job Submitter service (not Data Download) so the output is written to the required file. This time you will not be prompted with a parameter to select the source (or destination) datasets, but they will be used from the resources folders just the same.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Create resources folders and upload data to them*
- *Add a writer to a workspace and move feature types from another writer*
- *Run a workspace and select data from resources folders*
- *Edit a workspace to permanently use the resources folders*
- *Delete parameters to prevent the end-user changing them*

Running the Job Submitter using a URL

All job requests to an FME Server are a variation on an HTTP request. This makes running a workspace via a URL very simple, provided you know what form the request will take.

The easiest way to find that URL is in the advanced section of the Run Workspace page:

Advanced

Job Priority

Job Routing Tag

Queued Job Expiry Time

Running Job Expiry Time

Run Until Cancelled

Shareable Url

You can use this link to share with others to run this workspace. Parameters will be shared with the published defaults and anyone this link is shared with will require permission to this repository.

<http://training2017:80/fmeserver/#/workspaces/demo/Training/MitabParamTest.fmw/fmejobsubmitter>

Direct Url Example

You can use this link to run this workspace directly with the webservice using the currently configured parameters.

[http://TRAINING2017/fmejobsubmitter/Training/MitabParamTest.fmw?SourceDataset_MITAB=%24\(FME_SHAREDRESOURCE_DATA\)%5CZoning%5CZones.tab&opt_showresult=false&opt_servicemode=sync](http://TRAINING2017/fmejobsubmitter/Training/MitabParamTest.fmw?SourceDataset_MITAB=%24(FME_SHAREDRESOURCE_DATA)%5CZoning%5CZones.tab&opt_showresult=false&opt_servicemode=sync)

Notice that there are two versions of the URL. The descriptions of each are below.

Shareable Url

This URL will simply take the user to the Run Workspace page of the FME Server interface and prompt them to fill in any published parameters.

Direct Url

This URL includes parameters and the command to run the workspace immediately. This URL will run the workspace using the current parameter values without bringing a user to the "Run Workspace" page.

This information is a useful tool for building your own web applications that access FME Server services, because you can copy the HTTP request and embed it on your own website or application. You could also embed the URL or form into an email, or paste the URL directly into a web browser.

TIP

The URLs shown use HTTP GET requests whereas clicking Run on this HTML form uses an HTTP POST request.

There are limits to the amount of data you can send in a GET request because URLs have length restrictions that vary depending on the browser being used. If you anticipate that your request parameters may include very long strings, use a POST request instead.

Authoring Job Chains

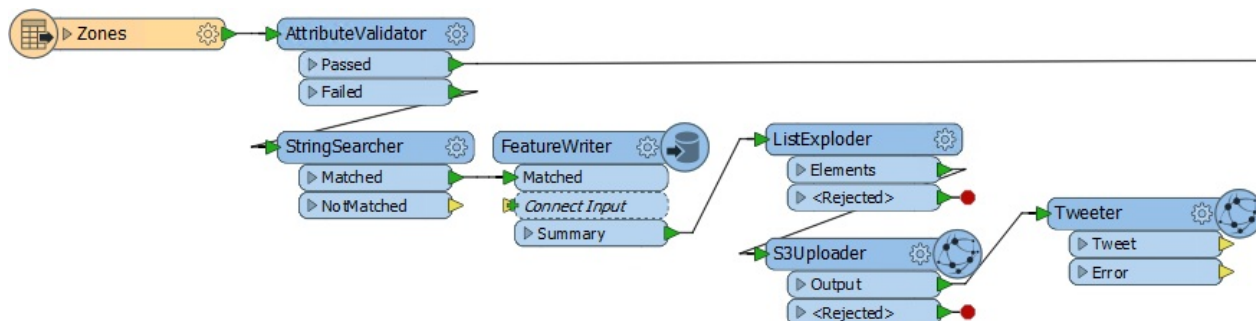
Workflow Management is a technique for controlling workspaces in sequence or branching with in-built logic. Part of this technique is being able to author workspaces that are "chained together" to run one after another.

What are Job Chains?

A chain of jobs is one that is run in sequence one after the other. There are various ways to implement this.

The FeatureWriter Transformer

The easiest way to chain workspaces... is not to! A chain is often necessary because one workspace writes data that the next must then process. However, the FeatureWriter transformer allows data to be written and then further transformation to take place within the same workspace!



In the above workspace, data is validated and validation errors are written out to a series of datasets according to the error type. These datasets are uploaded to Amazon S3 and a tweet sent to alert someone to the problems.

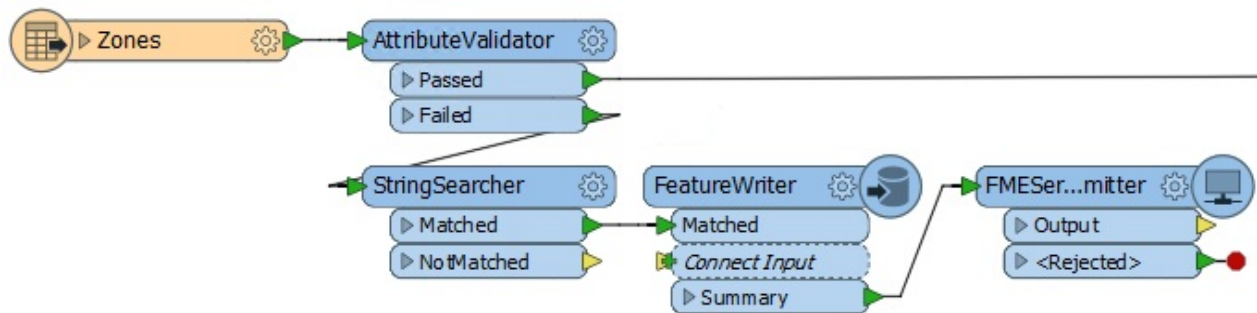
Without the FeatureWriter, such a project might have taken two or maybe three workspaces chained together. Here it only requires one.

A Simple Chain

It's fairly simple to get one workspace to run another; all that is needed is a transformer or shutdown script to send the command. Given that each workspace in FME Server can be run using a URL, and that there is a REST API to do similarly, it's quite simple to run an HTTP command using an HTTPCaller transformer or a shutdown script.

Alternatively, an FMEServerJobSubmitter transformer can be used. This transformer triggers a workspace to run on FME Server. It has various parameters that allow the author to define which workspace on which FME Server is to be run.

By adding this transformer to multiple workspaces a chain of almost any length can be created.



In this workspace the data is sent to a FeatureWriter transformer, and the results of that sent as a single summary feature that triggers the FMEServerJobSubmitter.

However, don't think a FeatureWriter is a necessity for a chain of data. It's equally valid for a step in the chain to not write data, but just carry out an action and trigger the next step.

Sister Intuitive says...

The first workspace in a chain may start with a Reader that reads a source dataset, where each source feature triggers the FMEServerJobSubmitter. For example, the source data may be a list of files that are to be translated.

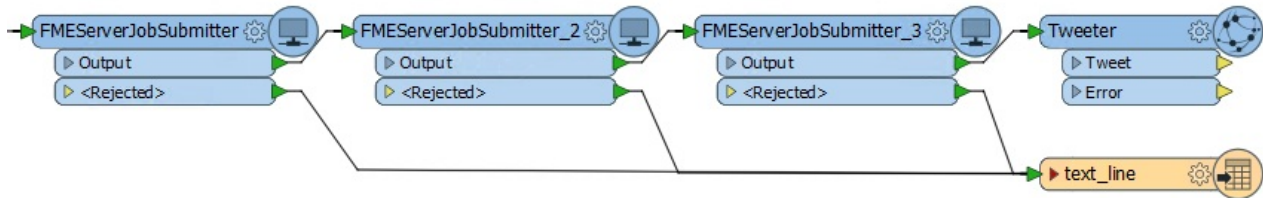
But more often the next job needs to be triggered just once. This requires only a single feature and to produce this a Creator transformer can be used instead of a Reader, or a Sampler transformer used to restrict the flow of features to a single one.

A Parent-Child Approach

Instead of a chain of workspaces where one calls the next, a different approach is to have a control (parent) workspace that runs a series of (child) workspaces in turn.

Like in a simple chain, a master workspace runs other workspaces by using a transformer such as the FMEServerJobSubmitter.

Here a control workspace is using the FMEServerJobSubmitter to run three further FME workspaces. Maybe each workspace is a separate step in a database update process:



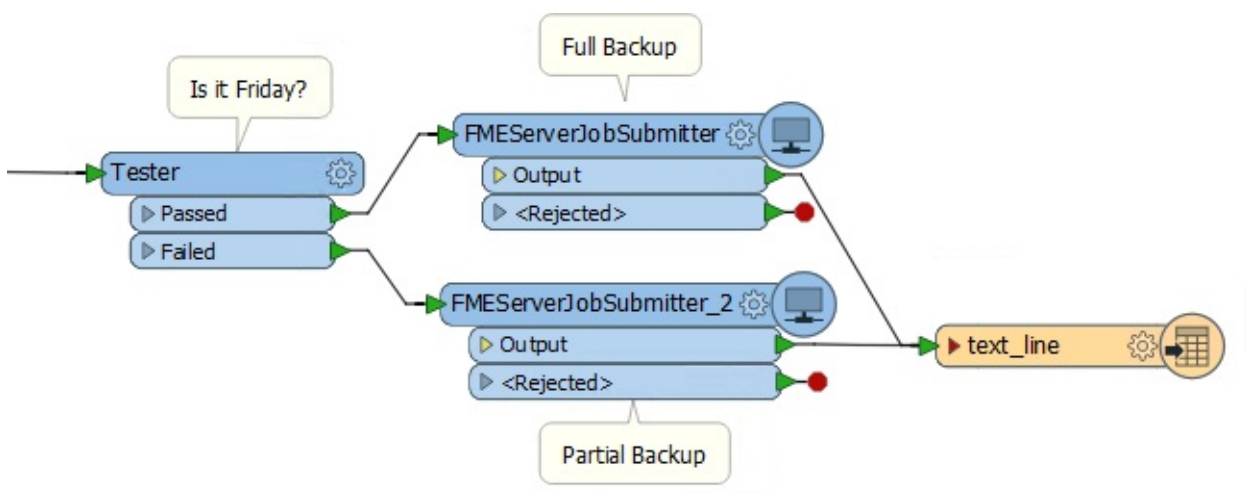
If a particular task fails then the output is routed to a text file Writer – meaning this could be used in a notification system to send an email to an administrator alerting them to the failure. Successfully executing all three results in a tweet being sent.

Instead of "Parent-Child", this setup is sometimes also known as the "Master-Slave" approach.

Conditional Processing

In some scenarios there might be several workspaces, only one of which should be run. To do so the logic for deciding which workspace is executed can be made using a Tester, or other filter transformers.

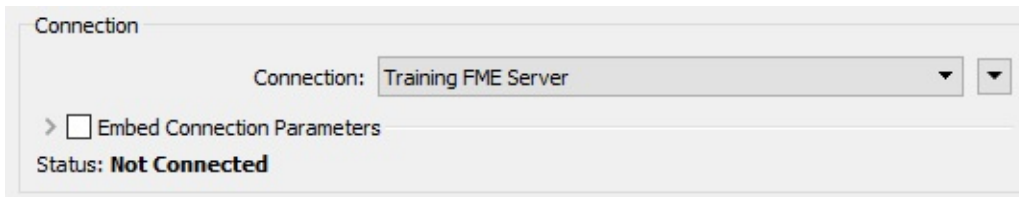
For example, here an organization runs a daily process to upload field updates into a database. Once a week it does the same, but also exports all files to Dropbox (for example):



The test is carried out by a Tester, and the daily/weekly processes are each defined in a separate workspace.

FMEServerJobSubmitter and Portability

The FMEServerJobSubmitter transformer allows the selection of an FME Server connection - in the same way as the Workspace Publishing wizard:



However, many FME Server projects include not one, but two servers: One for development and testing and the other the live system.

If you develop a solution using an FMEServerJobSubmitter that connects to MyFMEServerDev (for example) then, to save having to manually change each transformer MyFMEServerLive, you can make the connection into a user parameter:



Police Chief Webb-Mapp says...

*Another method would be to use the FME Server parameters **FME_SERVER_HOST** and **FME_SERVER_PORT**.*

*You would need to embed the connection parameters (rather than using a predefined connection) and set the FME Server URL to **\$(FME_SERVER_HOST):\$(FME_SERVER_PORT)***

That way the workspace would function depending on which Server it was published to, instead of having to set a published parameter. However, it does mean you couldn't test it on FME Desktop and each child (slave) workspace would need to be on the same FME Server as the parent (master).

WARNING

Interestingly the initial/control/parent workspace can be run on either FME Desktop (e.g. Workbench) or FME Server. The FMEServerJobSubmitter works on both platforms. So you can trigger the entire process to run on Server using a control that is executed on Desktop.

However, there's a difference. On FME Desktop the control workspace runs immediately, but each child job executed by an FMEServerJobSubmitter transformer is submitted to the FME Server queue and may have to wait for an engine.

On FME Server - if you have Wait for Job to Complete = Yes - it's the reverse: the control workspace is submitted to the queue, but each child job executed by an FMEServerJobSubmitter bypasses the queue and runs immediately.

This means that on Desktop the child processes are affected by the FMEServerJobSubmitter Job Priority and Job Tag parameters. But on Server (when Wait for Job = Yes) those parameters are ignored because the child processes are run immediately and not queued.

In short, those FMEServerJobSubmitter parameters only apply when the call comes from FME Desktop, because only then are the jobs queued.

| Exercise 6 Authoring Workspace Chains | |
|---------------------------------------|--|
| Data | Voting Divisions (GML (Geography Markup Language)) Addresses (Esri Geodatabase (File Geodb API)) |
| Overall Goal | Create workspaces to: - process voting divisions - to assign voting divisions to addresses - to chain the previous two translations together |
| Demonstrates | Authoring workspace chains |
| Start Workspace | None |
| End Workspace | C:\FMEDData2017\Workspaces\ServerAuthoring\Basics-Ex6-CompleteA.fmw C:\FMEDData2017\Workspaces\ServerAuthoring\Basics-Ex6-CompleteB.fmw C:\FMEDData2017\Workspaces\ServerAuthoring\Basics-Ex6-CompleteMaster.fmw |

You're a technical analyst in the GIS department of your local city. You have plenty of experience using FME Desktop, and your department has just purchased FME Server.

A municipal election is about to happen and Elections Interopolis have provided a dataset of new voting divisions in GML format. Your first task today is to create a workspace to translate these voting divisions to a Spatialite database format for use within the city, and write the data to a resources folder on FME Server so that everyone can use it.

Coincidentally, the planning department heard of this update and has asked you to assign voting division IDs to each of the records in the city's address database, for use in election planning.

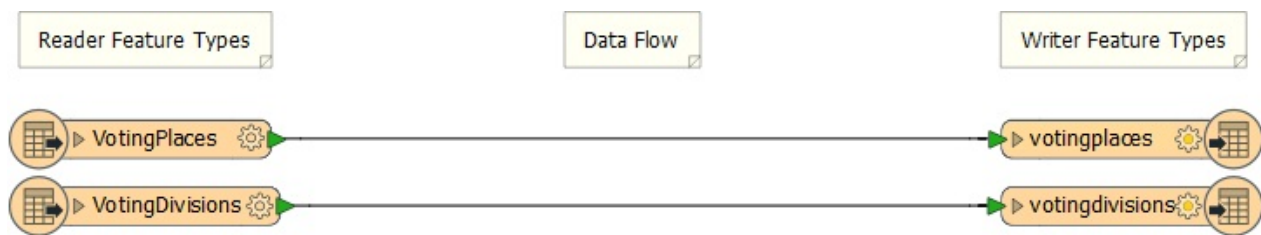
You realize that you can chain these two translations together to execute consecutively under a master workspace. So in all you have three workspaces to create!

1) Start FME Workbench

Start FME Workbench and generate a translation with these parameters:

| | |
|-----------------------|---|
| Reader Format | GML (Geography Markup Language) |
| Reader Dataset | C:\FMEDData2017\Data\Elections\ElectionVoting.gml |
| Writer Format | Spatialite |
| Writer Dataset | |

The Writer dataset can be left empty for now. When prompted, leave both source feature types (layers) selected.



2) Create Resources

We'll handle the input and output of this workspace using the resources folders on FME Server. So, log in to the FME Server web interface and navigate to the Resources page.

If you carried out exercise 5, then you should already have a folder `Resources\Data\Election\Input` containing the source data used in the workspace.

If not, create that set of folders and subfolders. Upload the source GML dataset to the Input folder (you should upload both the .gml and .xsd files):

| Home > Data > Election > Input | | | |
|---|-----------|--------------------|--|
| <input type="checkbox"/> Name | Size | Date | |
| <input type="checkbox"/> ElectionVoting.gml | 199.42 KB | 2017-4-25 14:00:59 | |
| <input type="checkbox"/> ElectionVoting.xsd | 2.45 KB | 2017-4-25 14:00:59 | |
| <input type="checkbox"/> VancouverNeighborhoods.kml | 304.79 KB | 2017-4-25 14:01:09 | |

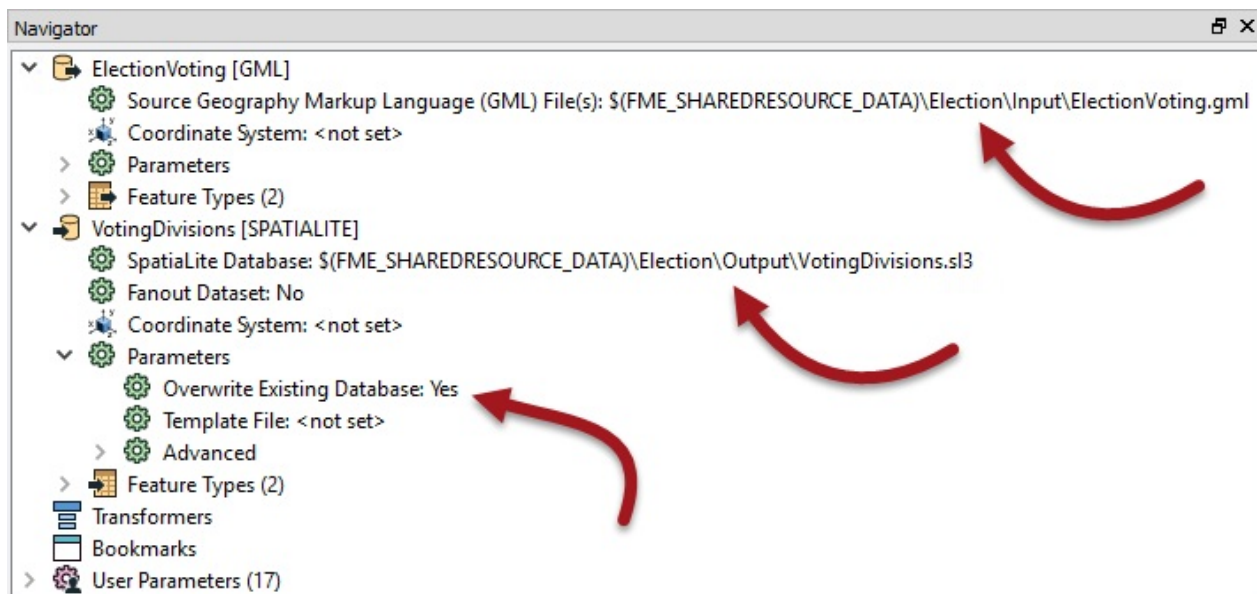
3) Edit Workspace to use Resources

Back in FME Workbench look in the Navigator window under User Parameters for the two existing published parameters called `SourceDataset_GML` and `DestDataset_SPATIALITE`. Click on each in turn and press the delete key to delete them.

Next locate the parameters for the GML source dataset and Spatialite destination dataset. Update the parameters to read as follows:

| | |
|-------------------|---|
| GML Reader | \$(FME_SHAREDRESOURCE_DATA)\Election\Input\ElectionVoting.gml |
| Spatialite Writer | \$(FME_SHAREDRESOURCE_DATA)\Election\Output\VotingDivisions.sl3 |

One final tweak: change the Writer parameter `Overwrite Existing Database` to `Yes`

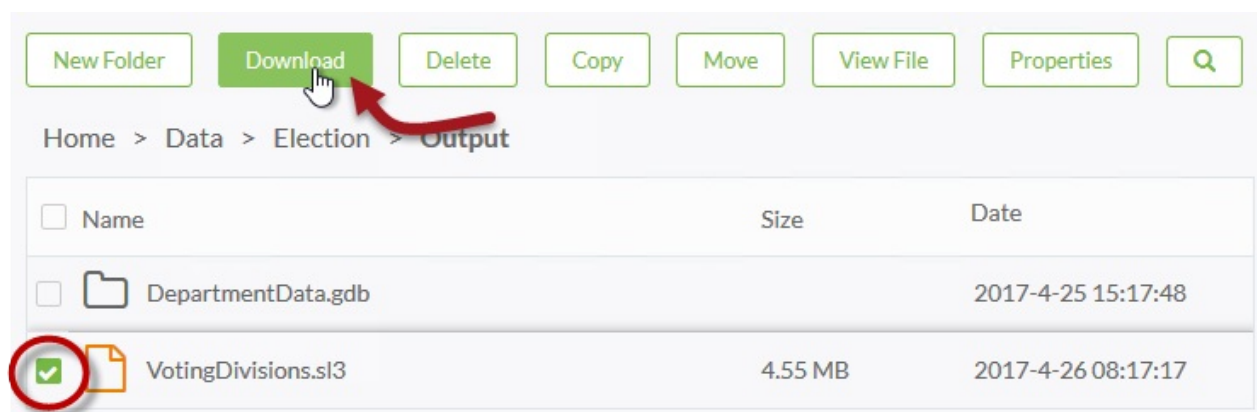


4) Save, Publish, and Run Workspace

Save the workspace (to something like Basics-Ex6-CompleteA.fmw) and remember the filename: it will be important later. Publish the workspace to FME Server. It should be registered with the Job Submitter service.

Locate the workspace in the Server web interface and run it to make sure it runs to completion. The evidence of success will be the log and an sl3 file in the resources folder.

Select the sl3 dataset and click the button to download the file. This is important; we'll need the file to set up our next workspace.



Save the file to the Elections folder, so you will remember where it is; i.e.

C:\FMEData2017\Data\Elections\VotingDivisions.sl3

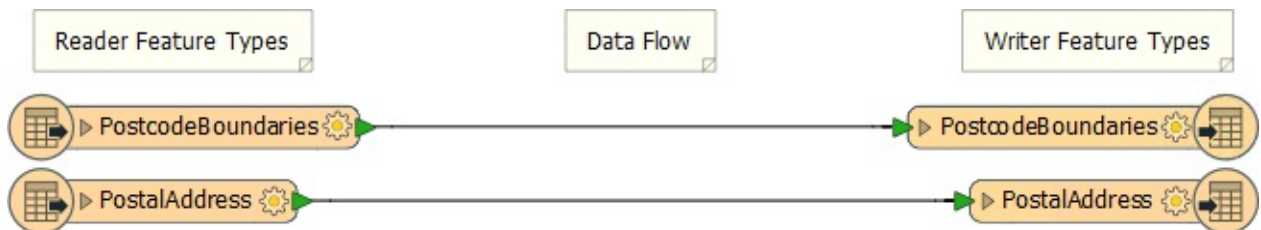
5) Generate Workspace

That was the first workspace in our project. Now for the second.

Open Workbench if necessary and generate a new workspace with these parameters:

| | |
|-----------------------|--|
| Reader Format | Esri Geodatabase (File Geodb Open API) |
| Reader Dataset | C:\FMEDData2017\Data\Addresses\Addresses.gdb |
| Writer Format | Esri Geodatabase (File Geodb Open API) |
| Writer Dataset | C:\FMEDData2017\Output\Training\NewAddresses.gdb |

When prompted, leave both source feature types (tables) selected.



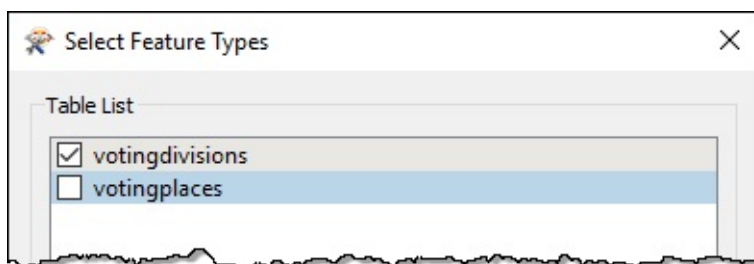
6) Add Reader

To assign voting divisions we need to have that data in our workspace. So, select Readers > Reader from the menubar and add a reader to read the downloaded VotingDivisions Spatialite database:

| | |
|-----------------------|--|
| Reader Format | Spatialite |
| Reader Dataset | C:\FMEDData2017\Data\Elections\VotingDivisions.sl3 |

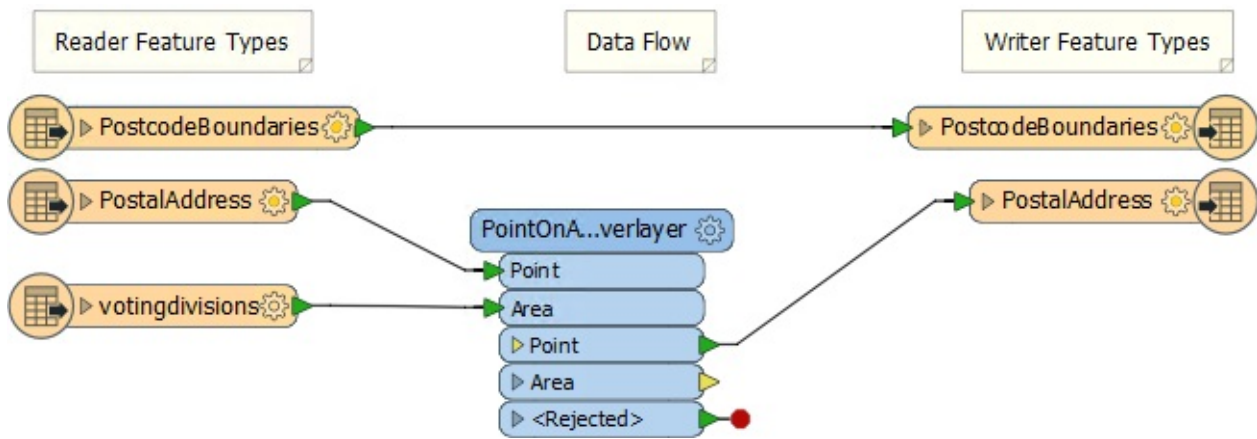
***NB:** If you can't find that sl3 file, go back to step 4 and make sure you downloaded the result of the first workspace.*

When prompted, select only the source feature type (table) *votingdivisions*.



7) Add Transformer

Now let's add a transformer to assign voting divisions to each address. Place a PointOnAreaOverlay transformer into the workspace. Connect it as follows:

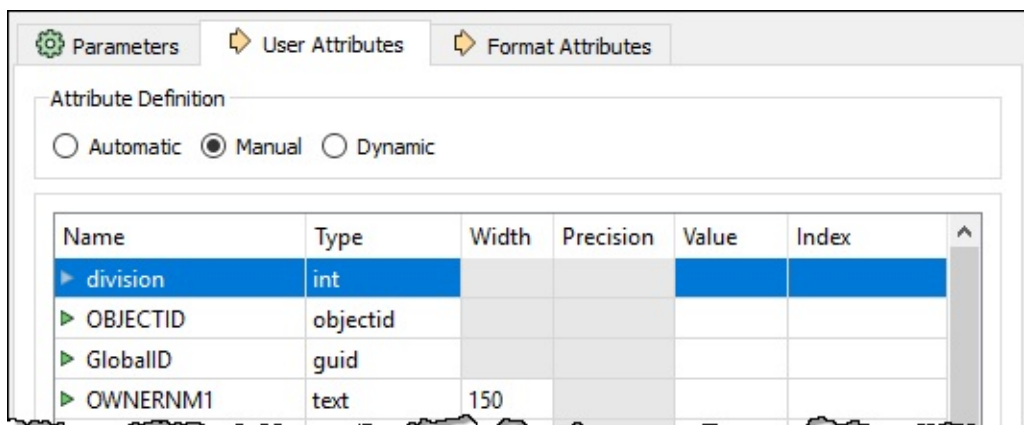


- **Delete Connection:** Geodatabase:PostalAddress > Geodatabase:PostalAddress
- **Add Connection:** Geodatabase:PostalAddress > PointOnAreaOverlayer:Point
- **Add Connection:** Spatialite:votingdivisions > PointOnAreaOverlayer:Area
- **Add Connection:** PointOnAreaOverlayer:Point > Geodatabase:PostalAddress

8) Edit Writer Schema

That transformer will copy the division attribute on to each address, but that attribute won't be written unless we also add it to the output schema.

So, inspect the parameters for the writer feature type PostalAddress. In the User Attributes tab add a new attribute called division (of type int):



division is case-sensitive, since we want it to match what is coming in from the *votingdivisions* table.

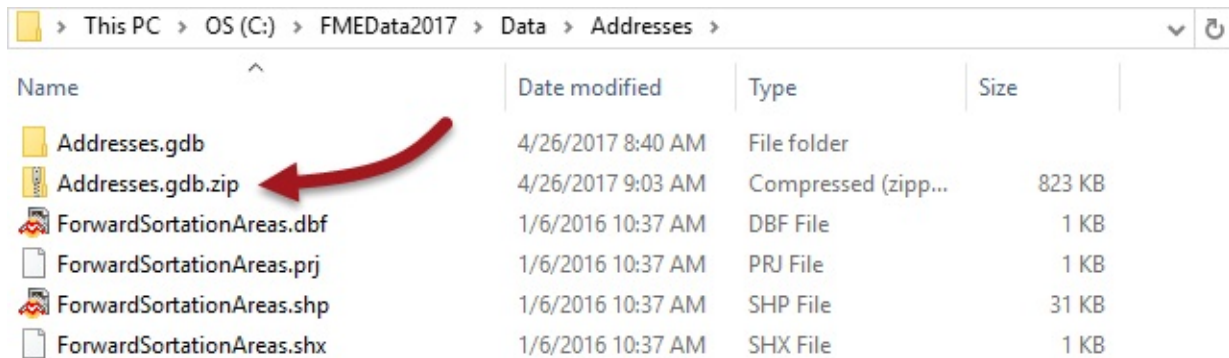
9) Test Run Workspace

Before we start adjusting the dataset paths for use on FME Server, run the workspace to ensure it produces the correct output; i.e. that each address now has a division attribute.

10) Create Resources

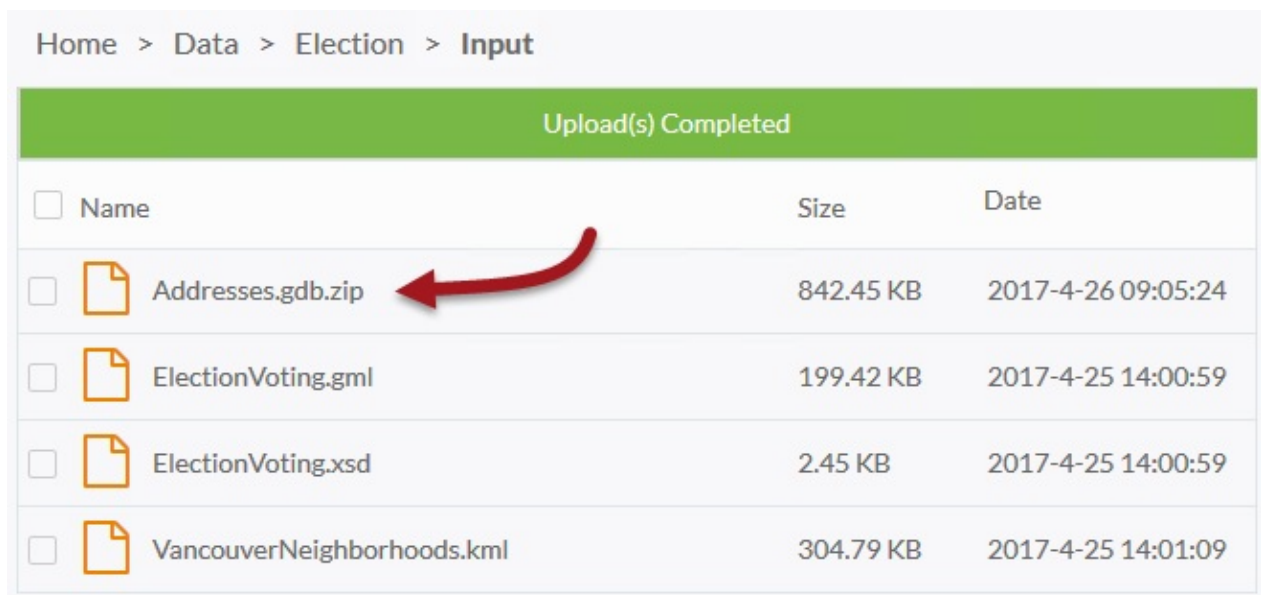
We'll also handle the input and output of this workspace using the resources folders on FME Server.





Firstly, we can upload a File Geodatabase as a folder/file only if we're using the Chrome web browser. Just in case you aren't, locate the source Geodatabase in your file system and compress it into a single zip file:



| Name | Date modified | Type | Size |
|---------------------------|-------------------|---------------------|--------|
| Addresses.gdb | 4/26/2017 8:40 AM | File folder | |
| Addresses.gdb.zip | 4/26/2017 9:03 AM | Compressed (zipp... | 823 KB |
| ForwardSortationAreas.dbf | 1/6/2016 10:37 AM | DBF File | 1 KB |
| ForwardSortationAreas.prj | 1/6/2016 10:37 AM | PRJ File | 1 KB |
| ForwardSortationAreas.shp | 1/6/2016 10:37 AM | SHP File | 31 KB |
| ForwardSortationAreas.shx | 1/6/2016 10:37 AM | SHX File | 1 KB |

Next, upload the zipped address file to the Resources folder on FME Server:



| Upload(s) Completed | | | |
|---|--|-----------|--------------------|
| <input type="checkbox"/> Name | | Size | Date |
| <input type="checkbox"/>  Addresses.gdb.zip | | 842.45 KB | 2017-4-26 09:05:24 |
| <input type="checkbox"/>  ElectionVoting.gml | | 199.42 KB | 2017-4-25 14:00:59 |
| <input type="checkbox"/>  ElectionVoting.xsd | | 2.45 KB | 2017-4-25 14:00:59 |
| <input type="checkbox"/>  VancouverNeighborhoods.kml | | 304.79 KB | 2017-4-25 14:01:09 |

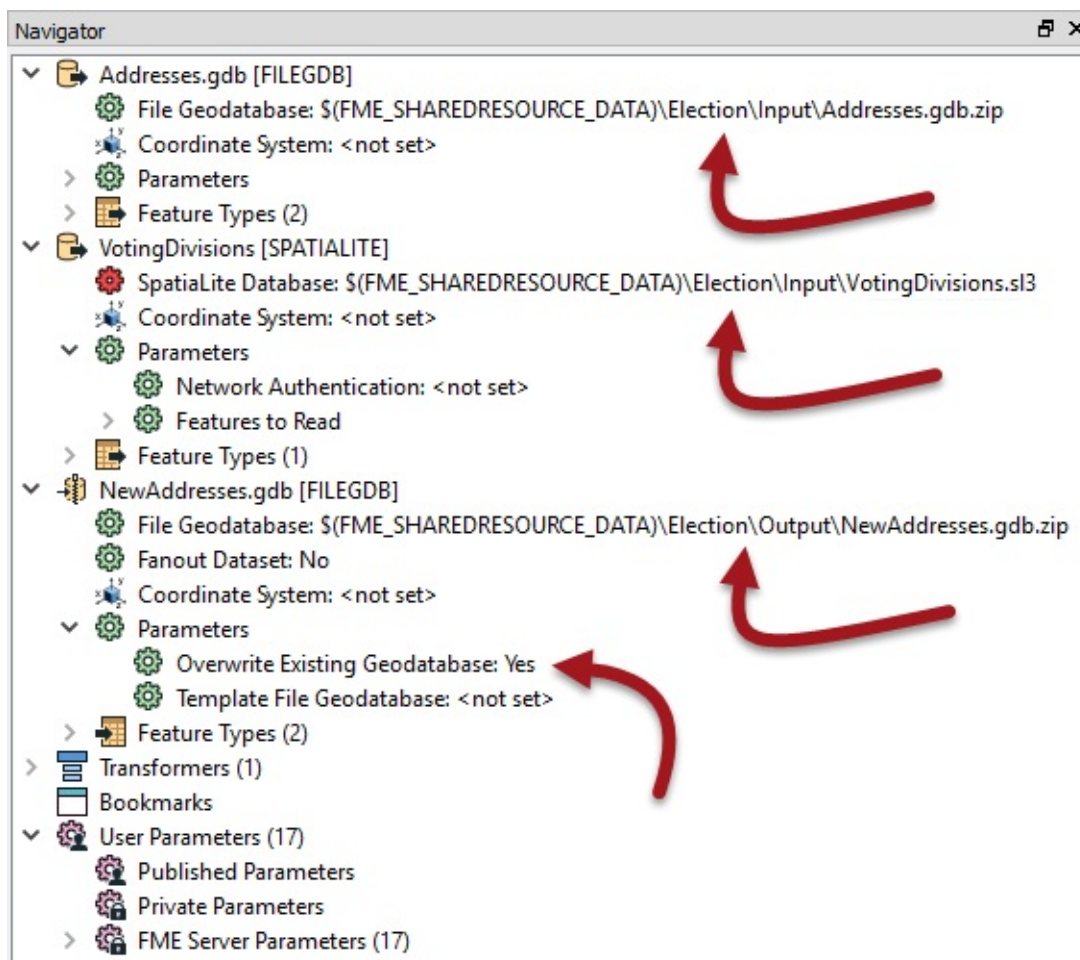
11) Edit Workspace to use Resources

Back in FME Workbench look in the Navigator window under User Parameters for the three existing published parameters called SourceDataset_FILEGDB, SourceDataset_SPATIALITE, and DestDataset_SPATIALITE. Click on each in turn and press the delete key to delete them.

Next locate the parameters for the Geodatabase source dataset, Spatialite source dataset, and Geodatabase destination dataset. Update the parameters to read as follows:

| | |
|--------------------|---|
| Geodatabase Reader | \$(FME_SHAREDRESOURCE_DATA)\Election\Input\Addresses.gdb.zip |
| SpatialLite Reader | \$(FME_SHAREDRESOURCE_DATA)\Election\Output\VotingDivisions.sl3 |
| Geodatabase Writer | \$(FME_SHAREDRESOURCE_DATA)\Election\Output\NewAddresses.gdb |

One final tweak: change the Writer parameter Overwrite Geodatabase to Yes






12) Save, Publish, and Run Workspace

Save the workspace (to something like Basics-Ex6-CompleteB.fmw) and remember the filename: it will be important later. Publish the workspace to FME Server. It should be registered with the Job Submitter service.

Locate the workspace in the Server web interface and run it to make sure it runs to completion. The evidence of success will be the log and a zipped geodatabase file in the resources folder.

Home > Data > Election > Output

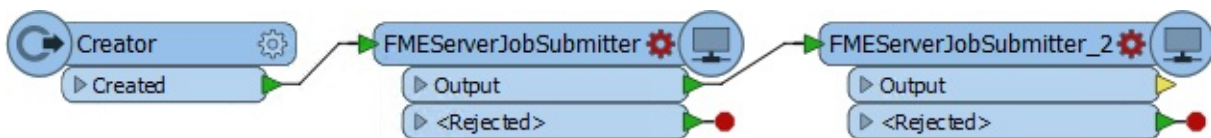
| <input type="checkbox"/> Name | Size | Date |
|---|-----------|--------------------|
| <input type="checkbox"/>  DepartmentData.gdb | | 2017-4-25 15:17:48 |
| <input type="checkbox"/>  NewAddresses.gdb.zip | 846.12 KB | 2017-4-26 09:17:39 |
| <input type="checkbox"/>  VotingDivisions.sl3 | 4.55 MB | 2017-4-26 08:17:17 |

You may wish to download the newly created dataset to inspect it and make sure the output is correct.

13) Create Workspace

That was the second workspace in our project. Now for the third and final workspace. This workspace will be used to chain the previous two workspaces. It is going to be the master, with the two prior workspaces as children.

So, open Workbench and start with an empty canvas. Place a Creator transformer followed by two FMEServerJobSubmitter transformers:

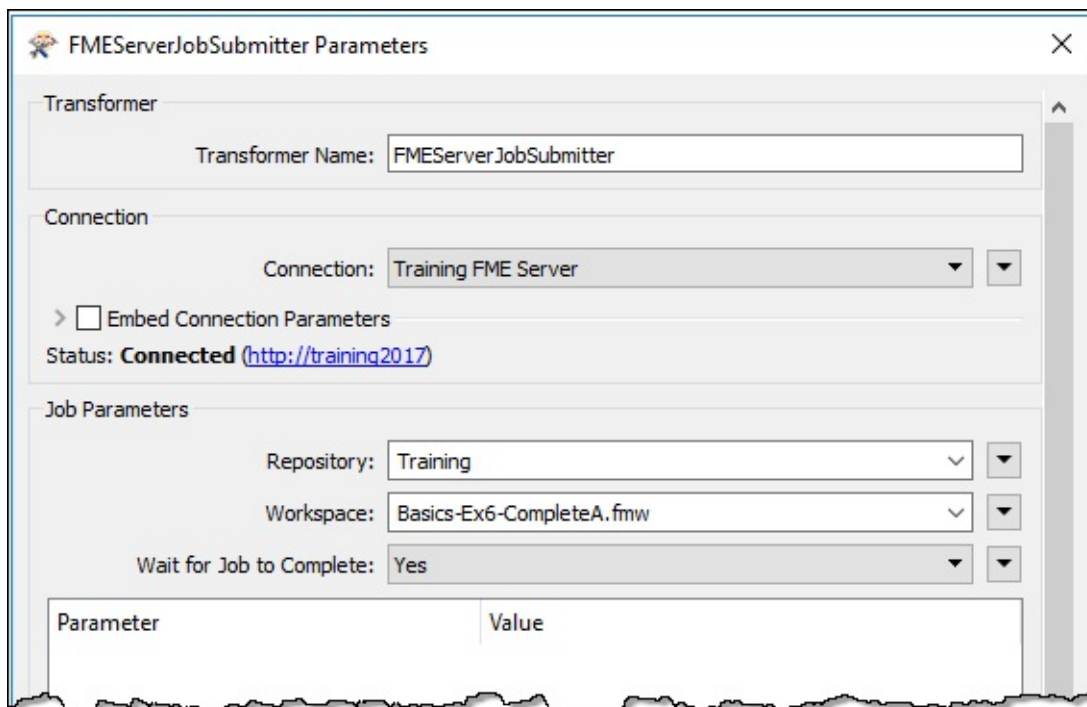


14) Set Parameters

Inspect the parameters for the first of the FMEServerJobSubmitter transformers.

Firstly select your FME Server connection. Then select the Training repository and the first of the two prior workspaces (the one that converted election divisions from GML to Spatialite).

Finally set Wait for Server Job to Complete to Yes. If we didn't do this then the second job submitter transformer would run before the first had finished!



Below is an area where we can set the parameters for the translation. However, since there are no published parameters, we don't need to worry about that.

Click OK to close the dialog.




Now repeat the same process for the second FMEServerJobSubmitter, this time selecting the second workspace (the one that did the overlay of addresses on divisions).

15) Save, Publish, and Run Workspace

Save the workspace and publish it to FME Server. It should be registered with the Job Submitter service.

Locate the workspace in the Server web interface and run it to make sure it runs to completion. It will run each of the two child workspaces in turn.

Don't worry that FME reports zero features written. That only refers to the master workspace (not the child workspaces). Evidence of success will be the log, and new output files (sl3.gdb) in the resources folder:

| Home > Data > Election > Output | | | |
|---------------------------------|--|-----------|--------------------|
| <input type="checkbox"/> | Name | Size | Date |
| <input type="checkbox"/> |  DepartmentData.gdb | | 2017-4-25 15:17:48 |
| <input type="checkbox"/> |  NewAddresses.gdb.zip | 846.12 KB | 2017-4-26 11:05:55 |
| <input type="checkbox"/> |  VotingDivisions.sl3 | 4.55 MB | 2017-4-26 11:05:50 |

Notice that the date/timestamps will be very similar for the two datasets; the VotingDivisions.sl3 file should be created first and then NewAddresses.gdb.zip shortly after.

Sister Intuitive says...

As already mentioned there are many ways to set up chains and this is just one of them. The drawback of writing data to a fixed location (like here) is that someone might change the first workspace to write data to a different location, causing the second workspace to fail.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Create child workspaces that read and write resources datasets*
- *Create a master workspace that runs the child workspaces using the FMEServerJobSubmitter*

Troubleshooting for Administrators

This section shows a few basic troubleshooting techniques in case of emergency.

FME Workbench-FME Server Connection

If you are unable to connect from FME Workbench to FME Server then the following suggestions may be of help.

- Check if there is a firewall running on either your computer or the FME Server. If so, you must open port 80 (or 8080) to use the Web Connection.
- Restart the FME Server Services. On Windows, go to Start > FME Server > Restart FME Server

FME Server Web Interface

If you are unable to access the FME Server web interface then the following suggestions may be of help.

- Confirm that FME Server is running! The easiest way is to restart the FME Server Services (as above).
- Check whether FME Server was installed using an application server port other than 80. For example, if port 80 was already being used the installer might have used a different port; 8080 is most common. To check, try entering the URL with this syntax <http://<host>:<port>/fmeserver> - for example <http://localhost:8080/fmeserver>

Workspaces are Queued but not Run

If a workspace appears in the FME Server queue, but is never executed, then it may be because no engines are running.

- Check the Web Interface (Admin > Engines and Licensing) to confirm engine status. If no engines are available then update licensing as necessary. Once a license is available the engines should restart automatically.

Workspaces Fail when Run

If a workspace fails when it is being run, then the following suggestions may help.

- Run the workspace first on FME Desktop. If it does not work there, it will not work on FME Server.
- Check the FME log file using the Jobs page in the Web Interface. The log may help to explain why there is a problem.
- Data paths can cause problems when moving from a local Desktop machine to a Server environment. Check the dataset parameters (Reader and Writer) to ensure they are not referring to a local path that does not exist on the Server. You may need to change the parameter to use the Resources filesystem and not a file path dialog.
- If you are trying to read data with a UNC path, ensure that the FME Server Windows Service is being run by a user with the proper domain access permissions.
- If you are using an FMEServerJobSubmitter transformer, be sure to check the server connection when porting from a test environment to a live environment.

Scheduling

If a scheduled workspace appears to have not been run at the expected time, then the following suggestions may be of help.

- Ensure an engine is available, and that the scheduled job is not in a queue.
- Check the date and time very carefully to ensure the correct values were entered.
- Check the timezone is correct. The web interface operates on local timezone, which is not necessarily the same timezone as the server is physically located.

Module Review

This module introduced you to FME Server and the Web Interface used to access most of its functionality.

What You Should Have Learned from this Module

The following are key points to be learned from this module.

Theory

- FME Server is a powerful product for handling large volumes of data at the enterprise level. It has three core capabilities: Self-Serve, Real-Time, and Automation.
- FME Server provides the same processing core as FME Desktop, but on the enterprise level. It does not include an authoring tool, for which FME Workbench is required.
- Someone who creates translations for use on FME Server is called an author.
- The three main components of FME Server are FME Engines, the Server Core, and Web Services.
- FME Server can be implemented on Physical hardware, Virtual hardware, or under a product called FME Cloud.
- The FME Server interface includes tools for running a workspace, examining job history, managing resources and data files, and scheduling jobs.
- Databases can have a container of connection information created and stored on FME Server.
- Data (and other related files) can be uploaded and stored on FME Server in various ways; either published data, temporary uploads, or as “Resources”.
- Workspaces can be authored in FME Desktop specifically to use data stored in FME Server resource folders.
- Developer information is provided to help run workspaces via a URL.
- Workspaces can be run in a series of translations using a technique called *chaining*.

FME Skills

- The ability to open the FME Server web interface
- The ability to browse repositories and run a workspace
- The ability to check job status

- The ability to schedule a job to run at a particular time
- The ability to create and use a database connection
- The ability to publish data with a workspace, upload data for a workspace, or manage data in resource folders
- The ability to create a workspace that references data in a resource folder
- The ability to chain together workspaces using a parent-child model

Further Reading

For further reading why not check out [this blog article on deploying FME Server using Docker containers](#) or take a look at [this blog article on the architecture of FME Cloud](#), which is both fascinating and instructive!

Questions

Here are the answers to the questions in this chapter.

Miss Vector says...

*Hi, I'm Miss Vector, FME schoolteacher. I'll be here now and then to test you on your new FME Server knowledge. For now, answer me this: Which of these is **not** one of the three core capabilities of FME Server?*

- 1. Automation*
- 2. Real-Time*
- 3. NoSQL Database**
- 4. Self-Serve*

FME Server is many things - but it is not a NoSQL database!

Miss Vector says...

I have an FME Server with two engines. Four users submit jobs at the same time. What happens?

- 1. Two jobs are processed. Two jobs are returned to their authors.*
- 2. Two extra engines will fire up automatically to process all four jobs.*
- 3. The four jobs will be processed simultaneously, sharing the two engines.*
- 4. Two jobs are processed. The other two sit in a queue until an engine becomes free.**

Yes, the server core keeps a queue of jobs and submits them as engines become available. Extra engines will not fire up, even if you do have spare licenses, and jobs will never be ignored just because no engine is currently available.

Miss Vector says...

If I wanted to find out about workspaces stored in a repository - for example I'm building a tool to catalogue my workspaces - what is the best way to do it?

- 1. Use the FME Server REST API**
- 2. Scrape the contents of the Server repository page*
- 3. Get a file listing from the repository folder*
- 4. Connect to the FME Server database to query it directly*

The REST API is a quick, simple, and official method to query the workspace repositories. Querying the database directly is permissible, however, under no circumstances should you write into or update directly the contents of the database.

Miss Vector says...

Which of these are good reasons for running engines on multiple operating systems at the same time? Pick all that apply.

- 1. A required format is only available on 32-bit or 64-bit, not both.**
- 2. A required format is only available on Windows, or Linux, not both.**
- 3. FME Desktop users author workspaces using a mix of Windows, Linux, and Mac platforms*
- 4. You want to process heavy-scale jobs on a more powerful platform.**

Basically some formats are only available on certain platforms and so you may need to mix of operating systems to cover all your requirements. You may also want to redirect large-scale jobs to a more powerful platform. However, it doesn't matter what platform the workspace author used; their jobs will run on whatever system FME Server is based on.

Miss Vector says...

If I create a database connection that has superuser permissions then it would bypass any permission checks the database would make for creating and dropping tables. So how do you think I might prevent a user misusing that capability?

- 1. Remove that user's permission to run that workspace on FME Server*
- 2. Remove that user's permissions to access the entire repository that workspace resides in*
- 3. Remove permission to access that particular database connection for that user's role**
- 4. Remove from their role permission to manage database connections*

You know it can't be 1 or 2, because the user could simply create a new workspace in a different repository and use that connection. You could remove permission to manage connections (4) but that won't prevent access to ones already created. You should have the system administrator find that connection in the security page and remove from it any roles or users that should not have access.

Miss Vector says...

Although simple, there is a major limitation to publishing data with a workspace. What do you think it is?

- 1. The data is only temporary and will be deleted once the workspace is run*
- 2. The data is hidden within FME Server's system files and limited in its use**
- 3. The data becomes available to anyone regardless of role and security settings*
- 4. The workspace cannot be run using any other data than that published with it*

The limitation is that a dataset published in this way can only be referenced by its own workspace or workspaces run from the same repository. Even then there is no browse capability in the FME Server web interface; the source dataset parameter would need setting manually.

Incidentally, none of the other answers are true: the data won't be deleted, it isn't open to anyone (unless they have specific access to this repository), and the workspace can be run using other data if required.

Miss Vector says...

I copy a workspace into a resources folder using the upload tool. What then?

- 1. I can run it by browsing the resources, selecting the workspace, and clicking run*
- 2. I can run it through the Manage > Workspaces menu tools*
- 3. I can run it by calling it with the FMEServerJobSubmitter transformer in FME Desktop*
- 4. I can't run it because it's not properly published to a repository**

Basically, if you don't publish the workspace properly, you aren't able to have Server run it.

Miss Vector says...

Uploading an entire folder of files come with what restriction?

- 1. Folder upload only works on certain web browsers**
- 2. Folder upload requires the folders to be zipped into a single file*
- 3. Folder upload only works on Windows C: drive (not D:, E:, etc)*
- 4. Folder upload requires FME Desktop to be installed on the computer being uploaded from*

Folder upload works in Chrome, but not in Firefox, Internet Explorer, or any other web browser.

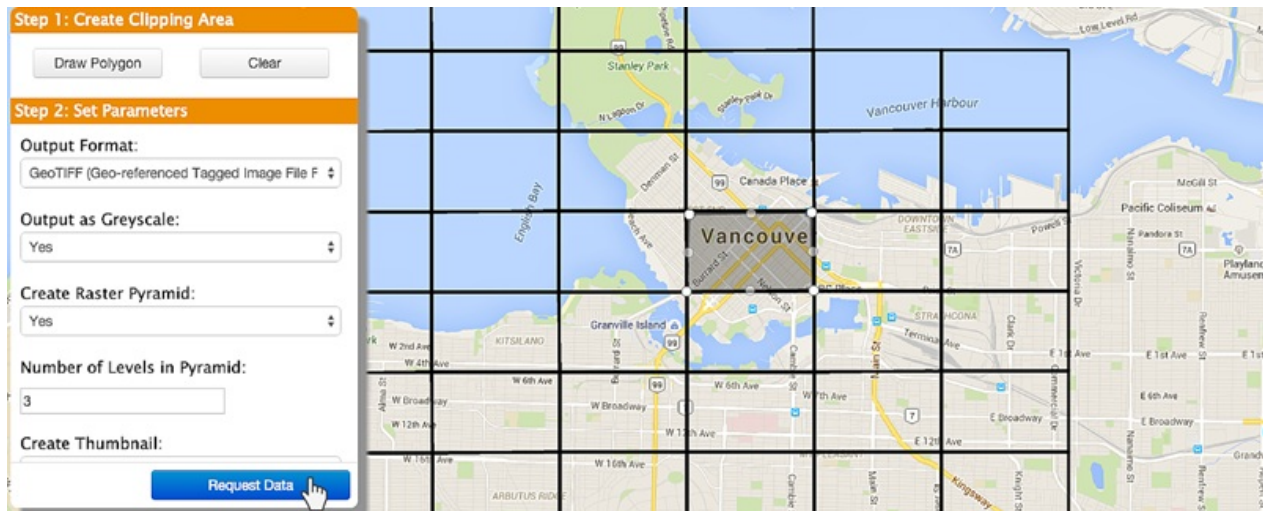
Miss Vector says...

So I can make my workspace read specific data from the resources folders - but how do I stop the end-user from being able to change that?

- 1. Remove their security permissions for the Job Submitter service*
- 2. Remove their security permissions for the Resources folders*
- 3. Make the source dataset parameter optional for that Reader*
- 4. Delete the published parameter for that source dataset from the workspace**

Yes, in the Navigator window look for a published parameter that relates to the source dataset, and remove it. The option to change the dataset will then not be presented to the user.

Self-Serve with FME Server



Self-Serve is the key technique for taking the burden of user requests away from expert staff, to enable them to concentrate on more important things.

This chapter looks at how self-serve is implemented on FME Server, and how to allow the end user to set parameters, select format, choose a coordinate system, and define which layers to download.

Self-Serve with FME Server

Self-Serve is the term for a system set up to allow the end-user to carry out their own data translations and transformations. In this way, routine data management tasks are offloaded from staff to the user, who is empowered to carry out processes at their own convenience.

Usually the system is set up in such a way that the end-user needs no prior FME experience or training to carry out their goals; for example, they can access the functionality through a web interface customized to their particular needs. In fact the user does not even need to know of FME, or that FME is the engine driving their applications!

Self-Serve Types

In general there are two types of self-serve systems.

Data Upload systems are where the user is able to upload their data to be processed on FME Server. A typical application would be a user uploading data to be validated. The data is run against a number of tests in an FME workspace and the results sent back to the user.

Data Download systems are where the user is able to serve themselves with data. A typical application would be an organization that frequently provides data to either staff or customers. With a Data Download system the user can fetch their own data, rather than having to be provided with it in a more manual way. Data can be downloaded as a set of files, or streamed directly into an application.

Data Uploads

A Data Upload system is one where a user provides data to FME Server for it to process.

Data Upload can be used from any FME Server client, including:

- The FME Server web interface
- An FME workspace
- A custom web page or application

Data Upload is often used for submitting data to an organization; for example a property developer submits a planning application containing a DWG dataset to a municipal planning department.

It is also often used for publishing data to be processed on FME Server; for example FME Server can provide a data validation web service and an end-user would upload a set of data to be checked.

It's worth noting that data upload also includes not just data, but other resources that may be required for a translation to run; for example custom transformers or text-file lookup tables may also be uploaded.

Data Downloads

A Data Download system is one where a user selects their own choice of data to download.

Data Download can also be used from any FME Server client, usually:

- The FME Server web interface
- A custom web page or application

Data Download operates in a different way to simply running a workspace.

When you run a workspace (with the Job Submitter) the data is written to the location specified by the workspace; for example a file, directory, or database.

Data Download instead writes the output to a zip file and presents the user with a link to that file. This makes it ideal for self-serve, because the data is delivered directly to the user.

Miss Vector says...

For each of these scenarios, tell me if it is a Data Download project, Data Upload, both, or neither.

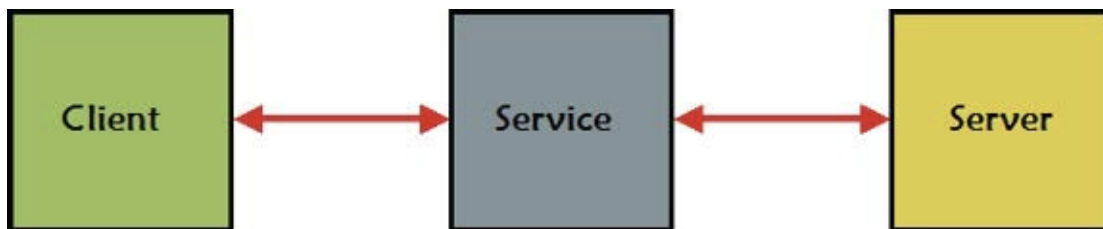
- 1. The user logs on to a web page, draws an area on the map, and is sent a copy of the data within that area*
- 2. The user submits a dataset to a web site that scans the data for errors, and returns a corrected copy*
- 3. The user publishes a workspace that writes data to the user's account in an online PostGIS database*
- 4. The user starts a GIS application, clicks File > Add Data to Map, and pastes in a URL from FME Server*

Self-Serve and Services

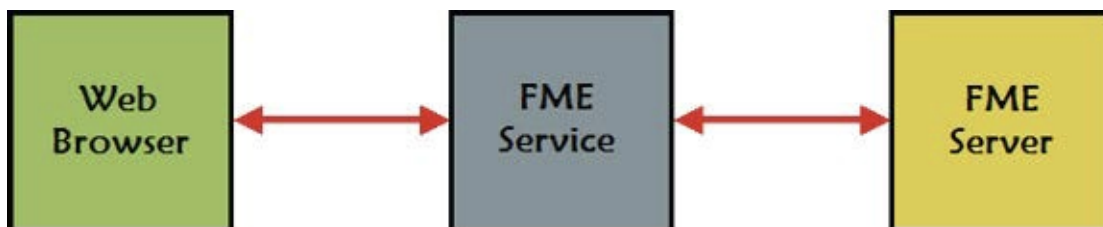
Self-Serve is implemented through a number of Services on FME Server. A Service is a particular method of communication between client and server. FME Server provides a wide range of services to carry out different forms of data self-serve.

What is a Service?

In the simplest of terms, a service is a piece of software that handles communications between a client and a server. In other words, it's a tool that allows users to access complex functionality through a simplified interface.



In terms of FME Server, the client is often—but not always—a web browser that passes requests to FME Server using a service.



In short, a service allows the sending of specific types of requests to FME Server, and allows results to be provided to client applications in a specific way.

For example, instead of just running a workspace, you can have a web page ask for the results of the workspace as a package of data compressed in a zip file.

Professor Spatial F.M.E., E.T.L. says...

Good morning class. I'm here to guide you through this chapter on Self Serve with FME.

Let's start with the idea of services. Although the concept sounds complicated, a service is just a simpler way of communicating requests to FME Server than using the API. Also, FME Server includes a number of predefined services that cover a lot of the functionality you are likely to need.

Available Services

FME Server includes the following services:

| | |
|-------------------------------|-------------------------|
| Data Download Service | Transformation Services |
| Data Streaming Service | |
| Job Submitter Service | |
| KML Network Link Service | |
| Data Upload Service | Utility Services |
| Token Service | |
| Web Connection (SOAP) Service | |
| REST Service | |
| Notification Service | Notification Services |

Remember that services can communicate in both directions. Transformation services – for example Data Download – are primarily Self-Serve tools for Server to deliver data to the end user.

Utility services can be described as “helper” services. They interact with FME Server to assist in menial tasks such as uploading data or providing token security. In most cases these are facilities that an author or developer will be using in a way that’s hidden from the user.

The Notification Service is used for passing short messages into and out of FME Server. Incoming messages notify FME Server to take some action, whereas outgoing messages alert an end-user (or system) that some sort of event has occurred.

TIP

Prior versions of FME had services specifically for OGC services such as WMS and WFS. Safe Software now recommends that customers looking to provide OGC Web Services do so using a combination of a workspace that implements the standard and the data streaming service that serves the output.

This new method of providing OGC Services is more flexible as you have full control over implementation of OGC features. See the FME Server documentation for more information and links to example workspaces.

Workspaces and Services

When a workspace is published to FME Server the last panel of the publishing wizard is for registering it with a particular service:

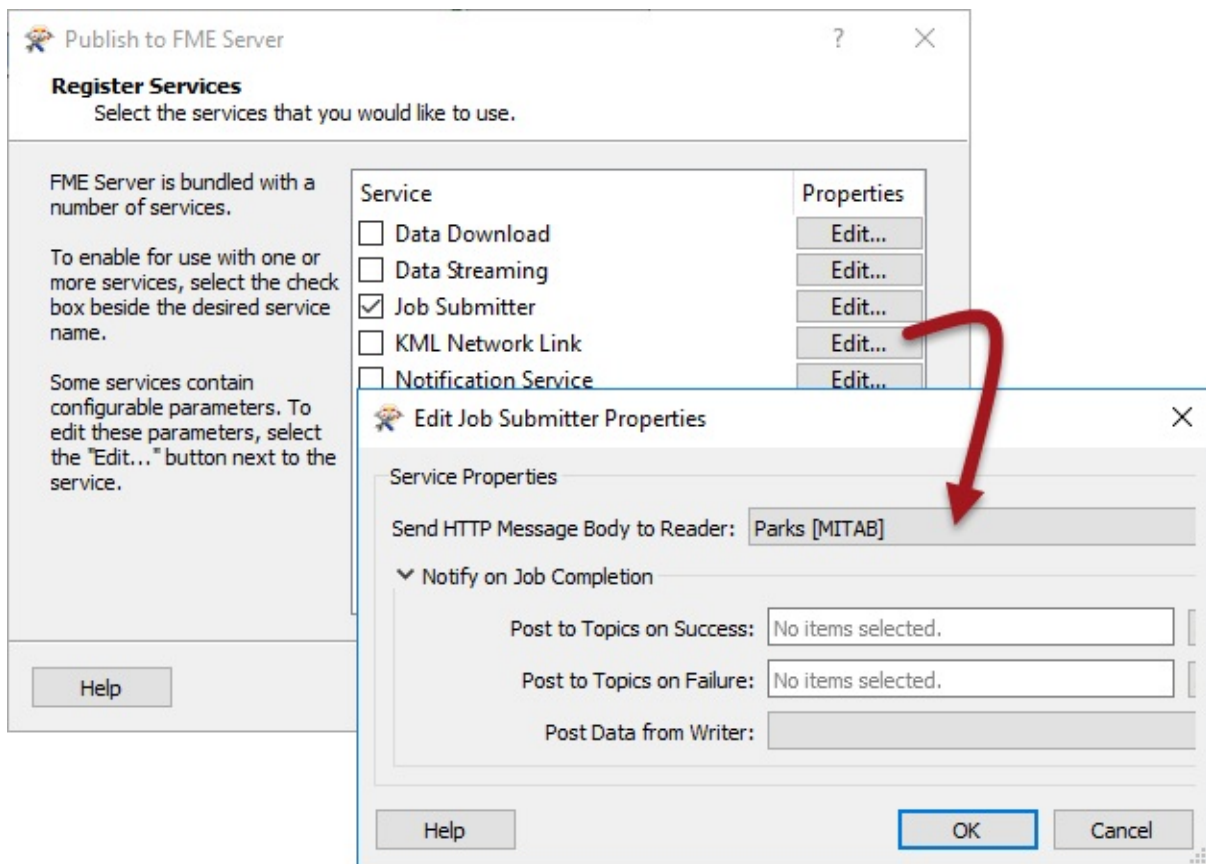
| Service | Properties |
|---|------------|
| <input type="checkbox"/> Data Download | Edit... |
| <input type="checkbox"/> Data Streaming | Edit... |
| <input checked="" type="checkbox"/> Job Submitter | Edit... |
| <input type="checkbox"/> KML Network Link | Edit... |
| <input type="checkbox"/> Notification Service | Edit... |

The Job Submitter service is automatically selected in the FME Server publishing wizard, whenever a workspace is published, but many other services are available too.

Registering a workspace with a service makes the workspace available for use in that service although, as you'll discover, not every workspace is capable of being used by every service.

Be aware of the Edit button to the right of each service.

Every service has a set of parameters available that determine how a workspace will be run with that service:



Notice how these parameters include ones for notification topics to trigger on completion of the workspace.

WARNING

*It's important to understand that a workspace may be registered against one service, many services, **or no services at all!***

Miss Vector says...

When a workspace is not registered against any service, how can you run it? Select all that apply.

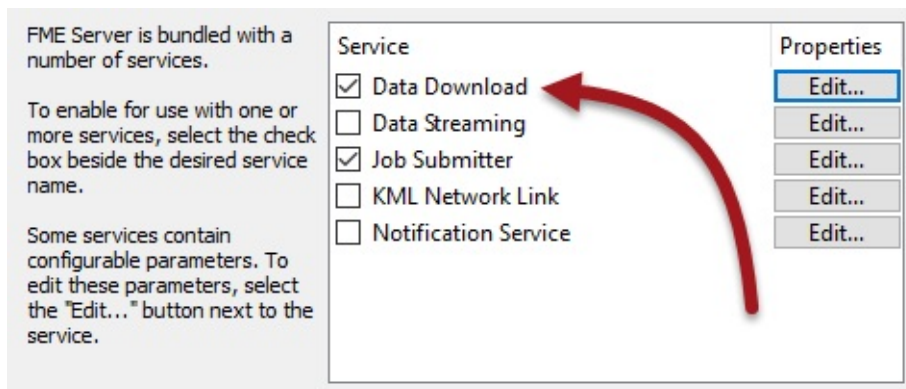
1. With the *FMEServerJobSubmitter* transformer
2. With the run dialog in the web interface
3. With the URL specified under *Developer Information* in the run dialog
4. By setting it to run under a schedule

Implementing Self Serve


Implementing a self-serve system makes use of the appropriate services available on FME Server.

Creating Data Download Services

Creating and using a data download service is very easy. A workspace becomes available for data download use when the author registers it against the Data Download service when publishing it to FME Server:



Once registered this way, Data Download becomes an acceptable way to run this workspace. This might be run in a number of ways, one of which is selecting Data Download as the service in the FME Server Web interface:

| | |
|--|---|
| Repository | <div>Samples ▼</div> <div>FME Server Samples Repository</div> |
| Workspace | <div>austinDownload.fmw ★ ▼</div> |
| Service | <div>Data Download ▼</div> |
| Email results to  | <div></div> |

The results of the workspace are not written to a specific output location; instead they are delivered to the user in the form of a hyperlink to a zipped dataset:

 **austinDownload.fmw**
City of Austin: Data Download

 **COMPLETED**
Translation Successful

View Details

JOB ID 7 FEATURES WRITTEN 929

DATA DOWNLOAD URL http://TRAINING2017/fmedatadownload/results/FME_393E2B08_1486664199867_3464.zip

Creating Data Upload Services

A Data Upload service is different in that a workspace is not registered specifically to this utility service. Instead, publishing source dataset parameters in the workspace allows data upload to take place.

On FME Server there are two ways to upload data, depending on the requirements for the data. There is a specific Data Upload Service and there is also the Resources filesystem.

The Data Upload Service uploads data to a particular workspace in a particular repository. The data is held temporarily for the workspace to be run.

Professor Spatial F.M.E., E.T.L. says...

You will have already met the Data Upload service: It's what is used when data is published to a repository alongside a workspace in the Workbench FME Server Publishing Wizard.

The Resources filesystem allows data to be uploaded to a folder for use by any workspace in any repository. This upload is persistent and the data held there for as long as it required.

When creating a customized solution that involves data upload, the developer can choose whether to use the Data Upload service or whether to use the Resources filesystem instead. The main difference is whether the data needs to reside on the Server permanently or just temporarily.

Miss Vector says...

When a workspace is registered against the Data Download service (and no other), how can you run it? Select all that apply.

- 1. With the FMEServerJobSubmitter transformer*
- 2. With the run dialog in the web interface*
- 3. With the URL specified under Developer Information in the run dialog*
- 4. By setting it to run under a schedule*

| Exercise 1 | Data Download System: Workspace Creation |
|-----------------|---|
| Data | Orthophoto images (GeoTIFF) |
| Overall Goal | Create an FME Server Data Download system for orthophotos |
| Demonstrates | Creating a workspace and running it with Data Download |
| Start Workspace | None |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe1-Ex1-Complete.fmw |

As a technical analyst in the GIS department of your local city you have plenty of experience using FME Desktop, and are just getting started with FME Server.

Other departments frequently ask the GIS team for orthophoto imagery of the city. Their format of choice is usually JPEG. Currently you use FME Desktop to translate the data, adding to your workspace any special requests they have such as a particular resolution, a specific area of interest, or even sets of vector data stamped onto the raster.

However good you are with FME Workbench, it does take time to set up each of these individual requests. It would be far better if the other departments could help themselves to the raster data, with options for all of their special requests built in.

Of course, you very soon realize that a Data Download system implemented on FME Server would be an ideal solution.

Miss Vector says...

*If you have lots of experience with FME Workbench - **and if your instructor agrees** - simply open the workspace listed in the header above and skip to step 6. But make sure you inspect the transformers and their parameters, so you know what you are working with.*

1) Create Workspace

Let's start off by creating a basic FME workspace to translate and transform the source raster data. Start FME Workbench and select Readers > Add Reader. When prompted enter these parameters:

| | |
|--------------------------|---|
| Reader Format | GeoTIFF (Geo-referenced Tagged Image File Format) |
| Reader Dataset | C:\FMEDData2017\Data\Orthophotos\06-07-LM.tif |
| Reader Parameters | Feature Type Name(s): From File Name(s) |
| Workflow Options | Single Merged Feature Type |

It's important to use the Single Merged Feature Type option because there are many source tiles of data, and we may want to read any of them without having to add them as individual feature types.

The Feature Type Name parameter is important because it will help us later allowing the user to select which layers to read.

2) Add Writer

Now we need a Writer. Select Writers > Add Writer from the menubar. When prompted enter these parameters:

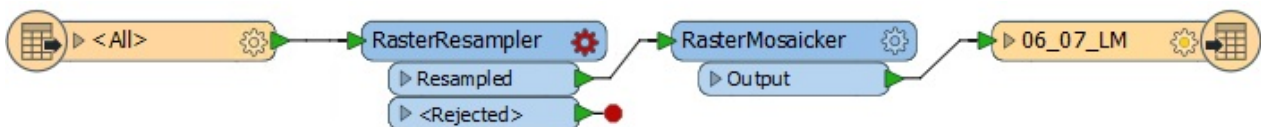
| | |
|--------------------------|---|
| Writer Format | JPEG (Joint Photographic Experts Group) |
| Writer Dataset | C:\FMEDData2017\Output |
| Add Feature Types | Copy From Reader |

Your workspace will now look like this:



3) Add Transformers

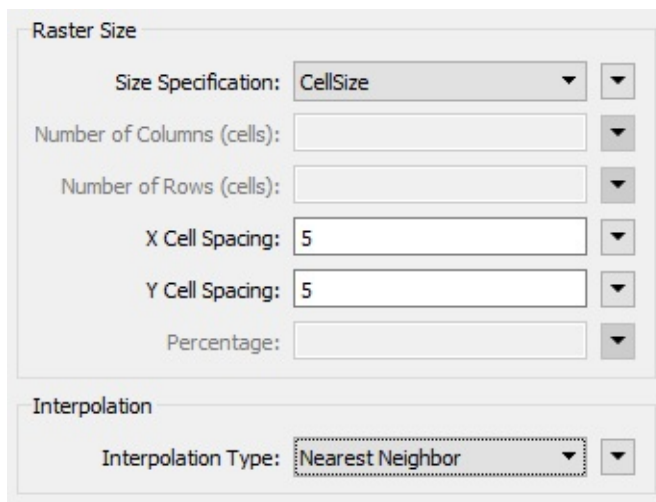
We'll start out with two transformers in our workspace; a RasterResampler and a RasterMosaicker. So place one of each of these and connect up everything in the workspace:



4) Set Transformer Parameters

Inspect the RasterResampler's parameters and set:

| | |
|---------------------------|-----------|
| Size Specification | Cell Size |
| X Cell Spacing | 5 |
| Y Cell Spacing | 5 |



Raster Size

Size Specification:

Number of Columns (cells):

Number of Rows (cells):

X Cell Spacing:

Y Cell Spacing:

Percentage:

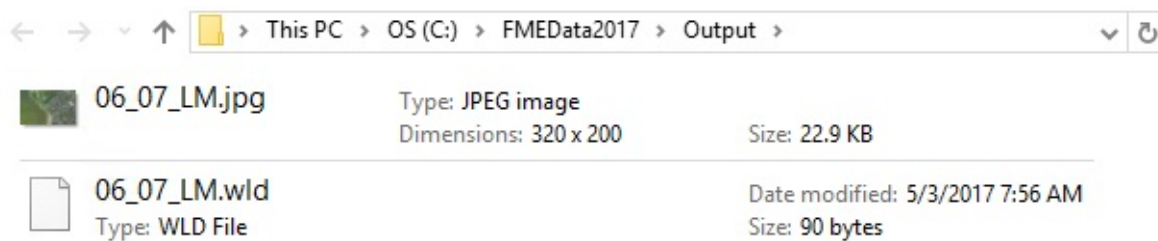
Interpolation

Interpolation Type:

You may inspect the RasterMosaicker's parameters, but there aren't any that need changing at the moment. You may notice the Overlapping Values parameter that is new for FME2017.

5) Save and Run Workspace

Save the workspace and - just to ensure that all is well - run it in FME Workbench. The result should be a JPEG file (06_07_LM.jpg) along with a world file (06_07_LM.wld).



6) Publish Workspace

Now publish the workspace to FME Server. Register it with the Data Download service. If you are working on a remote server, such as an FME Cloud instance, you'll need to either publish the data with the workspace or upload it to an FME Server Resources folder.

7) Run Workspace

Log in to the FME Server web interface, locate the workspace, and run it.

The workspace will run and you will be presented with a hyperlink to a zip file of the output dataset:

★ SelfServe1-Exercise1-Complete.fmw



COMPLETED

Translation Successful

[View Details](#)

JOB ID 56

FEATURES WRITTEN 1

DATA DOWNLOAD URL http://TRAINING2017/fmedatadownload/results/FME_3938_14823589329_3344.zip

CONGRATULATIONS

By completing this exercise you have learned how to:

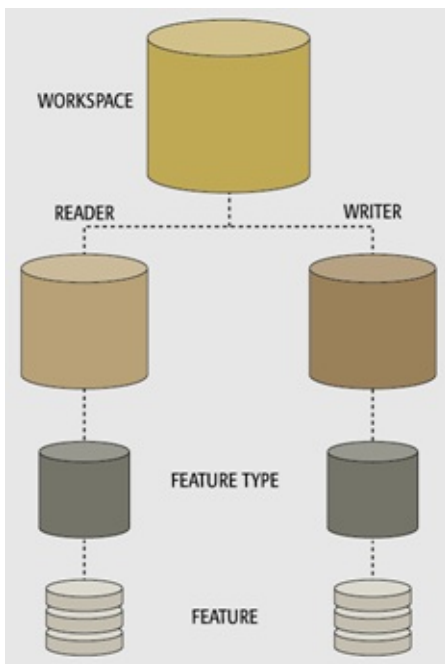
- *Create a workspace to read and write raster data*
- *Publish and run a workspace using the Data Download service*

Self-Serve and Parameters

Workspace parameters are the key to controlling Self-Serve and the Data Download service.

What are Parameters?

Parameters are what control FME translations and transformations. Remember that a translation has a hierarchy of different translation components:



Each different level of the hierarchy has a set of parameters that belong to it. That means there are:

- Workspace Parameters
- Reader Parameters
- Writer Parameters
- Feature Type Parameters

Most of the available parameters are determined by the author of the workspace.

Chef Bimm says...

Order food in my restaurant and I'll decide how long it needs to be cooked, at what temperature, and using what equipment. I'll also decide the amount of seasoning it needs and what plate to present it on. Like an FME author, these are the parameters that control the results, and as the creator of the meal I get to choose how best to set them.

Published Parameters

Although authors set *most* of the parameter, in some cases, the end user needs to be able to set some of them.

Chef Bimm says...

Of course, I will let you have some decision on your meal. For example, I'll let you decide how your steak should be cooked - rare, medium, or well done - and what sauce you wish to add. I'll also let you select the side dishes. These are the parameters in FME that the user gets to wield instead of the author.

To enable users to select the requirements for their translation, FME includes functionality called *User Parameters*. User parameters are methods for getting input into a workspace.

When a user parameter is made available to the end-user it is called a *Published Parameter*. In a self-serve application, published parameters are important to let the end user control how the data is served in terms of style and structure.

Parameter Uses in FME Server

For self-serve systems, published parameters are most commonly used to set:

- What coordinate systems to deliver data in
 - What feature types (layers) to deliver
 - What geographic area (Bounding Box or Area of Interest) to deliver
 - Any other Reader, Writer, or Transformer parameters of use to the user
-

With FME Server, the key to successful workspace authoring is flexibility. Workspaces need to be flexible to allow end-users to make choices without seeing all of the complexity of the workspace or the data behind it. Parameters are one way to accomplish this.

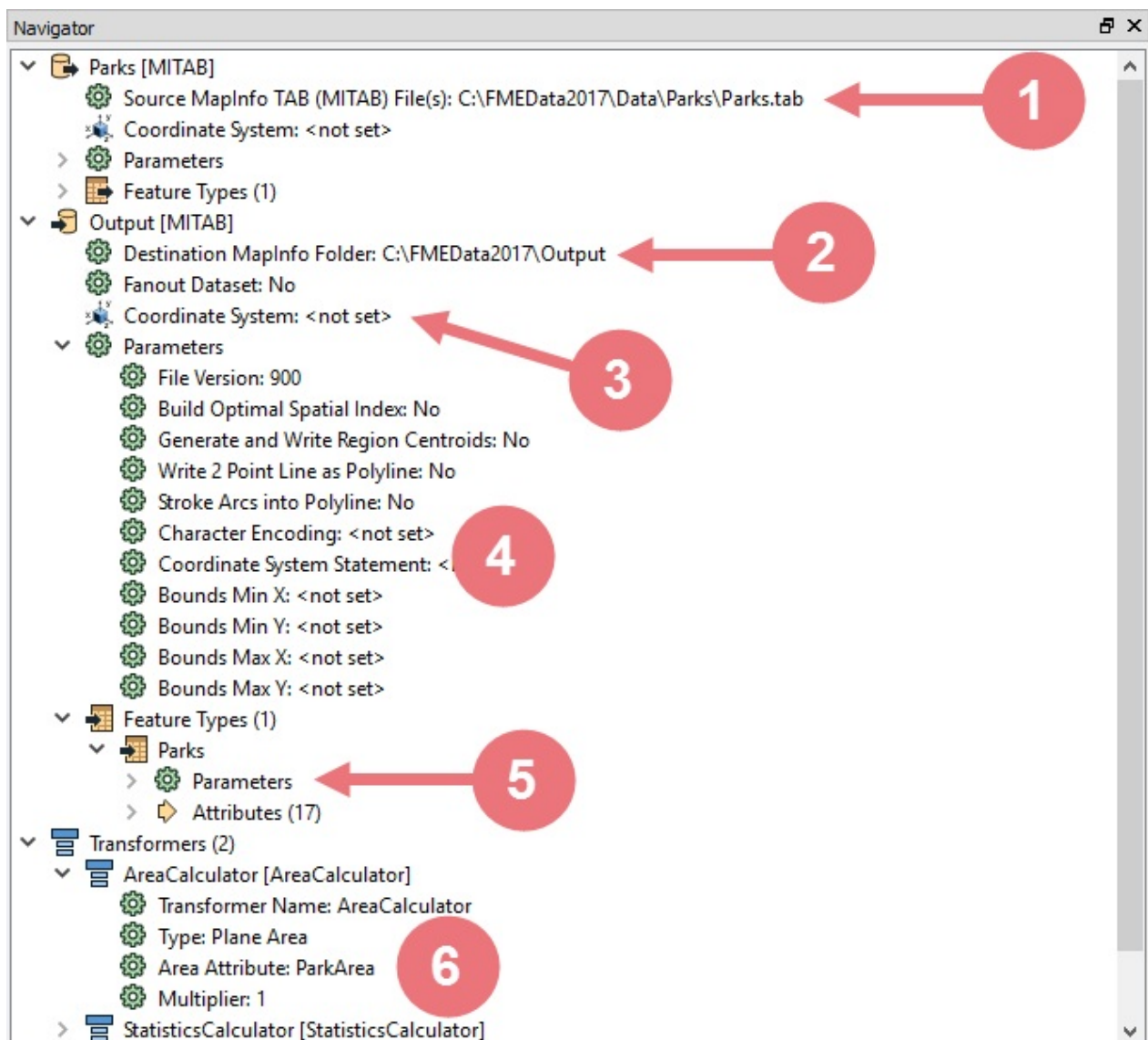
Miscellaneous Published Parameters

Any parameter in FME can be published and presented to the user as a choice to be made when running a workspace. They don't have to be specifically related to the Data Download service.

Parameters are published in FME Desktop (i.e. FME Workbench) for use in FME Server.

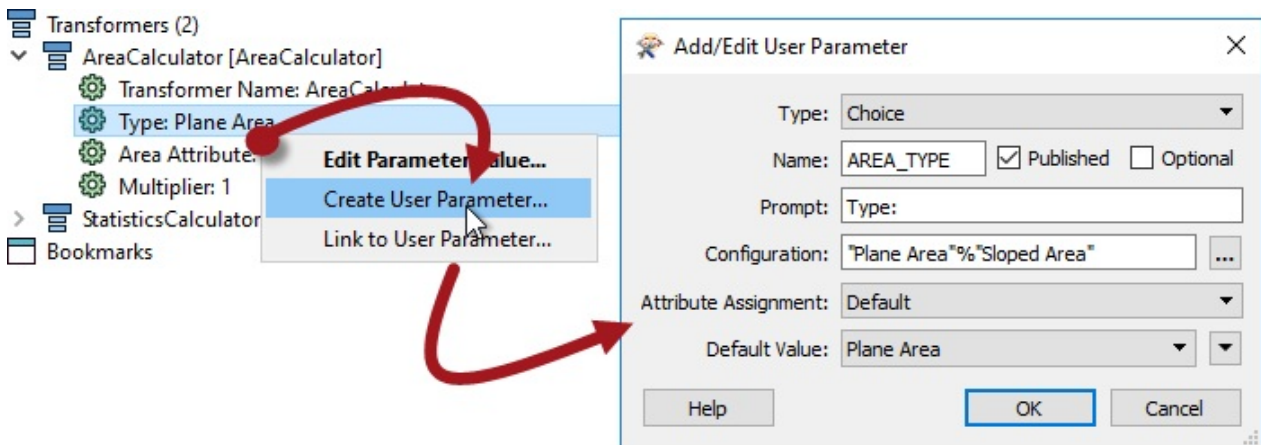
Publishing a Parameter

In Workbench parameters are located in several places, but the FME Workbench Navigator window is the one place where you will find them gathered in a single location:



As you can see, there are parameters for the source data location (1) and the destination dataset (2), which is the one that the Data Download service overrides. There are also parameters for controlling the coordinate system (3), general Reader and Writer parameters (4), parameters for each feature type (5) and parameters for transformers (6).

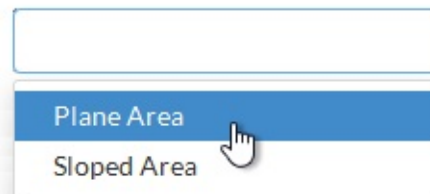
Parameters are published by right-clicking on them and choosing “Create User Parameter”. For example, here a workspace author is publishing a parameter on the LabelPointReplacer transformer:



The author gets to choose the type of parameter (in this case a Choice of different fixed values), the user prompt, and the default value. When the workspace is run the end-user will now be able to decide what value to set the parameter to:

Published Parameters

Type



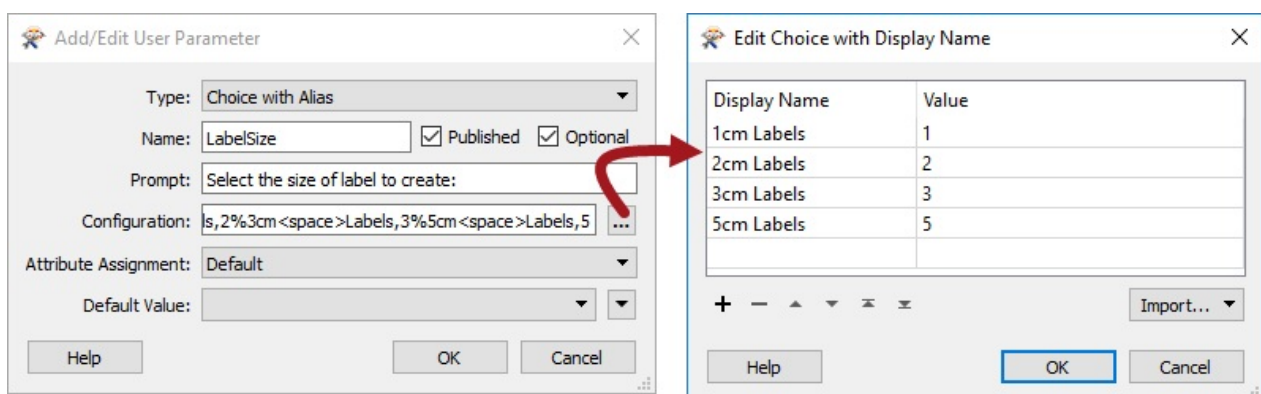
Key Parameter Types

There are multiple types of parameter, mainly related to the data type required; for example integer, float, string. However, two key parameter types for a self-serve setup are "Choice with Alias" and "Scripted".

Choice with Alias

A Choice parameter is where a user is presented with a list of choices and their selected is returned directly to FME. A Choice with Alias parameter is the same thing - the user is presented with a choice - but this time a different value is returned to FME based on a lookup table.

For example, here the author wishes the user to select the height of labels to be created in the output dataset:



The user is presented with the list of (more intuitive) options on the left, but the value returned to FME will be the (more useful) number on the right.

On FME Server this parameter type is very useful for Data Download tasks. That's because many of the values we want to pass to FME - for example format or coordinate system names - have an abbreviated format that wouldn't make sense if presented to the user. A Choice with Alias parameter allows us to provide a user-friendly list of options that hides a more complex response to the server.

Scripted

Scripted parameters are a way to generate a parameter value using a Python or Tcl script.

For example, setting the size of labels in an output dataset might be a bit more complex than just a single number. It may involve calculations related to the type of feature being labelled, the scale of any map being produced, the coordinate system used, and many other factors.

In that case a scripted parameter could be used to calculate the required value in a Python script and return it to FME as a user parameter.

Most importantly, a scripted parameter can include references to other user parameters; for example the scale of the map being produced might be selected by the user in a choice parameter, and label height calculated using that as one of the factors.

On FME Server this is again very useful for Data Downloads. This time it's because some values we want to return - for example a list of feature types - need to be constructed from a number of inputs. A scripted parameter allows us to accept input from the user and incorporate that into a script that constructs the actual, more complex, response to the server.

Miss Vector says...

How well do you know the types of FME published parameters? Decide which of the following are real parameters, and which are fake.

1. *Color*
2. *Double*
3. *Password*
4. *Text (Multiline)*

| Exercise 2 Data Download System: Published Parameters | |
|---|---|
| Data | Orthophoto images (GeoTIFF) |
| Overall Goal | Create an FME Server Data Download system for orthophotos |
| Demonstrates | Creating published parameters for user control in Data Download |
| Start Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe1-Ex2-Begin.fmw |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe1-Ex2-Complete.fmw |

As a technical analyst in the GIS department of a city you have just commenced a project to allow other departments to download orthophoto data, rather than having to ask you to create it for them. Not only will their requests be processed quicker, you will also spend less time on that task.

So far you have created a simple workspace to translate orthophotos to JPEG format, and published it to a Data Download service on FME Server.

Now you need to start customizing the workspace to allow the end-users to have a degree of control over the output.

1) Open Workspace

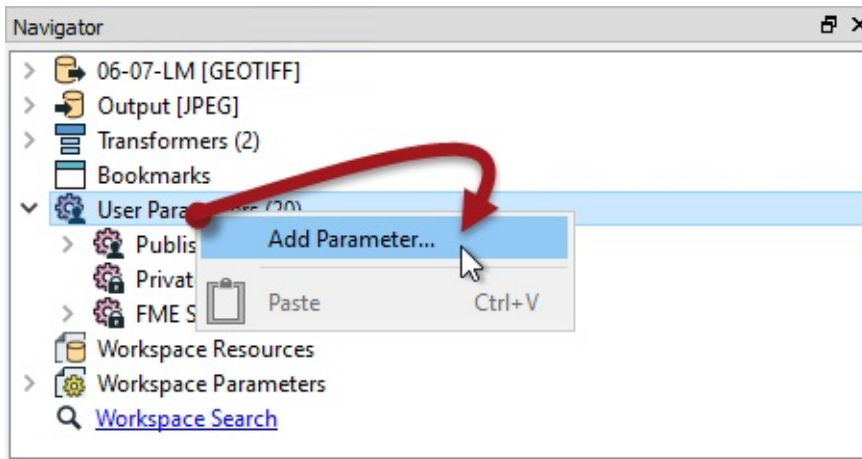
Open the workspace from exercise 1, or the starting workspace listed above. You can see that it consists of a reader, a writer, and two transformers.

In this step we'll give the end-user control over the transformation stages.

2) Create User Parameter

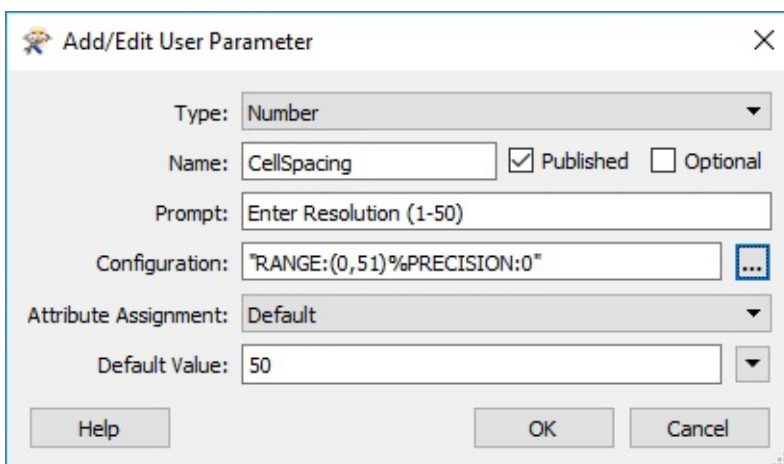
If you look at the parameters for the RasterResampler transformer you'll see parameters for X Cell Spacing and Y Cell Spacing. We should let the end user choose what spacing they want.

So, in the Navigator window of FME Workbench, locate the section marked User Parameters. Right-click on there and choose the option Add Parameter:



The dialog that opens allows us to create a new parameter. Create one using the following parameters:

| | |
|----------------------|--|
| Type | Number |
| Name | CellSpacing |
| Published | Yes |
| Optional | No |
| Prompt | Enter Resolution (1-50) |
| Configuration | Lower Limit: Greater than value: 0 Upper Limit: Less than value: 51 Decimal places of precision: 0 |
| Default Value | 50 |



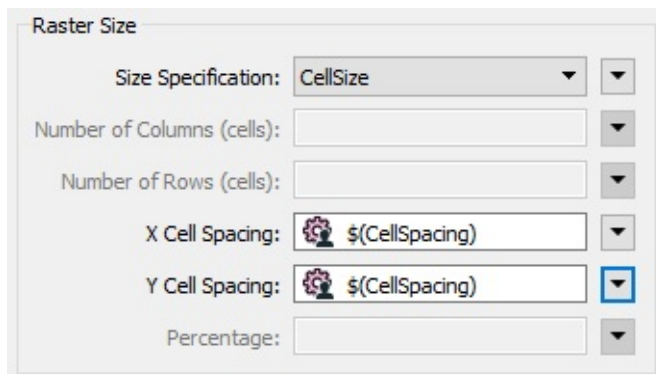
Click OK to close the dialog.

3) Apply User Parameter

Currently we've created a user parameter, but not applied it to anywhere.

Inspect the parameters for the RasterResampler transformer. Click the drop-down arrow to the right of the X Cell Spacing parameter, and choose User Parameter > CellSpacing

Do the same for the Y Cell Spacing parameter. The dialog will now look like this:



Notice that we're using the same values for the X and Y cell sizes. That's OK. Although we could use rectangular (oblong) raster cells, for this exercise we'll stick with square.

4) Create User Parameter

Another setting we might give control of to the user is file compression. This is not defined in a transformer, but in the writer feature type. However, we can still create a published parameter in the same way.

So, right-click on User Parameters in the Navigator window and choose Add Parameter again.

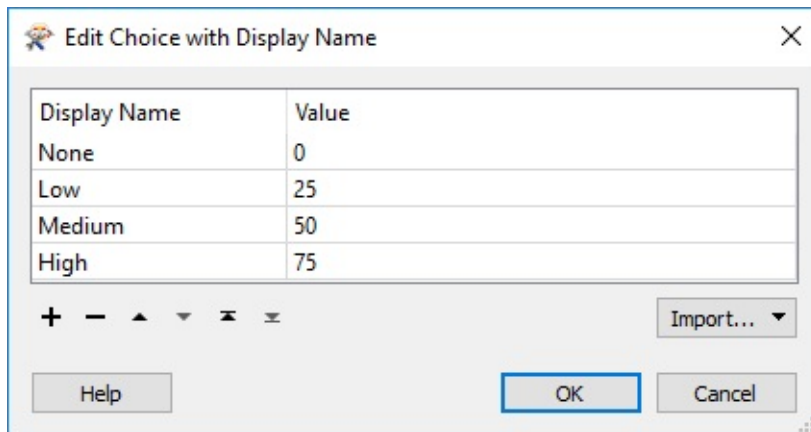
This time we'll do this a little bit differently. Compression can be a value from zero to one hundred, but we'll present the user with the choice of None, Low, Medium, and High.

So create a parameter with the following:

| | |
|------------------|--------------------------|
| Type | Choice with Alias |
| Name | Compression |
| Published | Yes |
| Optional | No |
| Prompt | Select Compression Level |

For the configuration field, click the [...] browse button. In the dialog that opens, set the following:

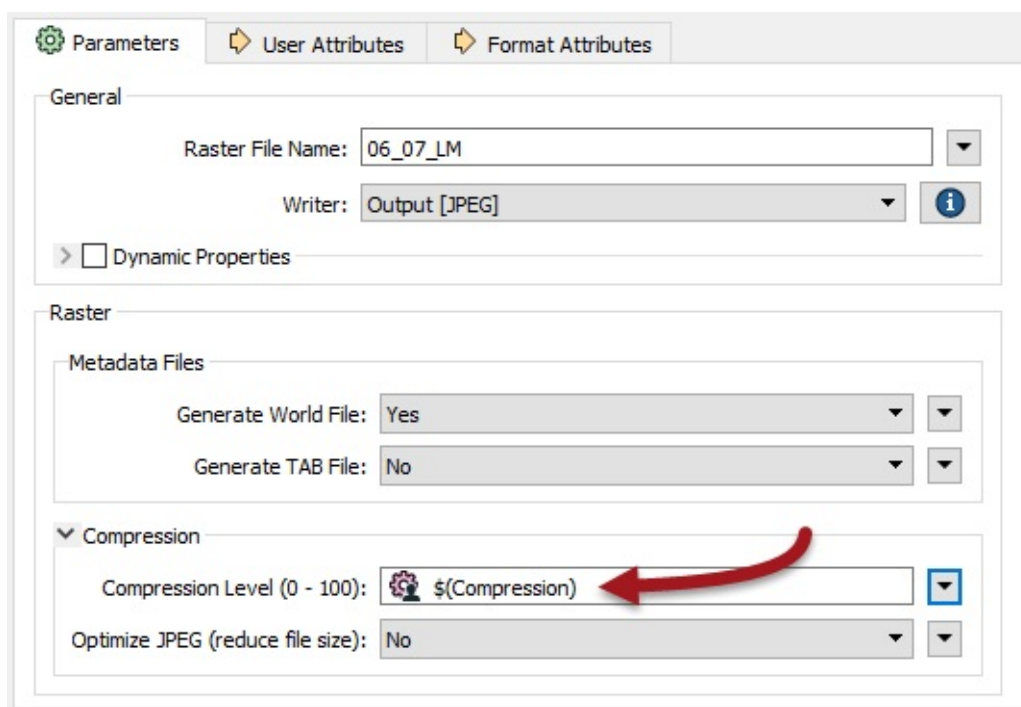
| Display Name | Value |
|--------------|-------|
| None | 0 |
| Low | 25 |
| Medium | 50 |
| High | 75 |



Click OK and OK again to close these dialogs and create the parameter.

5) Apply User Parameter

To apply the parameter, inspect the parameters for the JPEG feature type. Expand the Compression parameters (if necessary) and set the Compression Level parameter to User Parameter > Compression





Click OK to close the dialog. If you press the run button now - with the prompt option set - you'll see that there are now two new prompts for cell size and compression.

6) Publish and Run Workspace

Now publish the workspace to FME Server again. As before, register it with the Data Download service.

Locate the workspace through the FME Server web interface and run it. This time you will be prompted to set the cell size and compression.

Published Parameters

| | |
|--------------------------|---|
| Source GeoTIFF File(s) | <input type="text" value="\$ (FME_MF_DIR)06-07-LM.tif"/> ... |
| Feature Types to Read | <input type="text"/> |
| Enter Resolution (1-50) | <input type="text" value="50"/>  |
| Select Compression Level | <div><div>None</div><div>Low</div><div>Medium</div><div>High</div></div>  |

Run the workspace a few times, varying the cell size and compression, to confirm that the parameters are having an effect. The size of the output file is a good indicator that the process is working correctly.

CONGRATULATIONS

By completing this exercise you have learned how to:

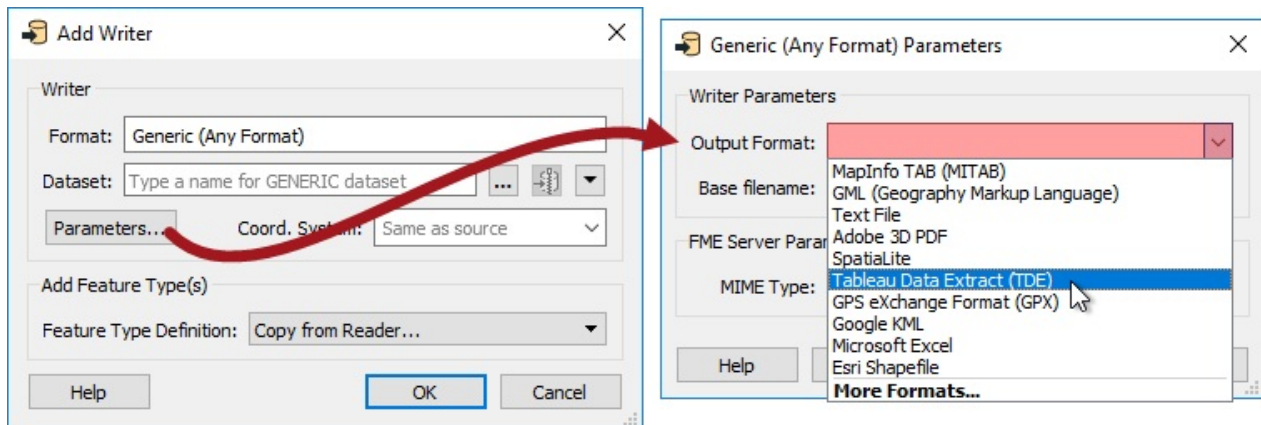
- *Create an integer user parameter and apply it to two transformer parameters*
- *Create a choice user parameter and apply it to a writer feature type parameter*
- *Publish a workspace and use published parameters*

Format and Coordinate System Selection

Self-serve systems allow an end-user to download data in the schema of their choice. This schema includes both data format and data coordinate system.

Format Selection

To control format at run-time requires the use of the Generic reader/writer in FME. The Writer has a parameter that controls which format of data is being written.



Here, for example, the Generic writer is being set to write data in Tableau Data Extract format. However, if the author of the workspace publishes this parameter, the user gets to choose their own output format at run-time:

Published Parameters

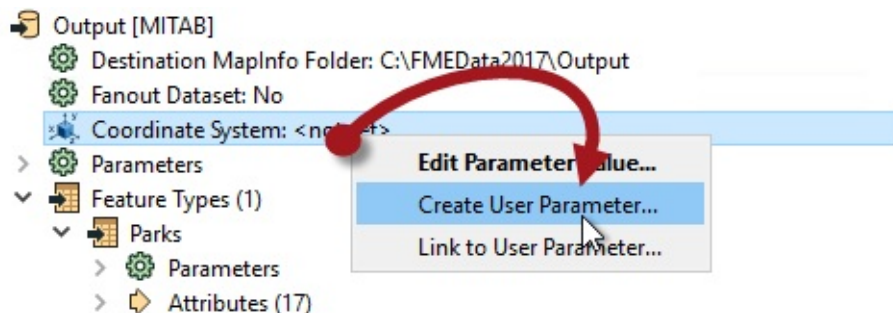
| | |
|---|------------|
| Type | Plane Area |
| Select the size of label to create | |
| Destination Generic (Any Format) Folder | |
| Output Format | TDE |

One point to keep in mind when using the writer is that each writer format has its own specific parameters, and these may still need to be set when a generic writer is used. This can be achieved by adding a writer of the same format and setting the parameters in that

writer. The Generic writer will inherit the parameters of this dummy writer, even if no features are connected to it.

Coordinate System Selection

Regardless of format, each writer in FME has a coordinate system parameter that can be published:



This allows the end-user to receive data in a coordinate system of their choice:

Published Parameters

| | |
|---|------------|
| Type | Plane Area |
| Select the size of label to create | |
| Destination Generic (Any Format) Folder | ... |
| Output Format | TDE |
| Coordinate System | LL84 |

Alternatively a transformer – such as the CsmmapReprojector – can be used, in which case the relevant parameters can be found under the Transformers section of the Navigator window. The obvious advantage to using a transformer is that you have control over other reprojection factors, such as the geographic transformation and grid height.

Using Choice with Alias for Formats/Coordinate Systems

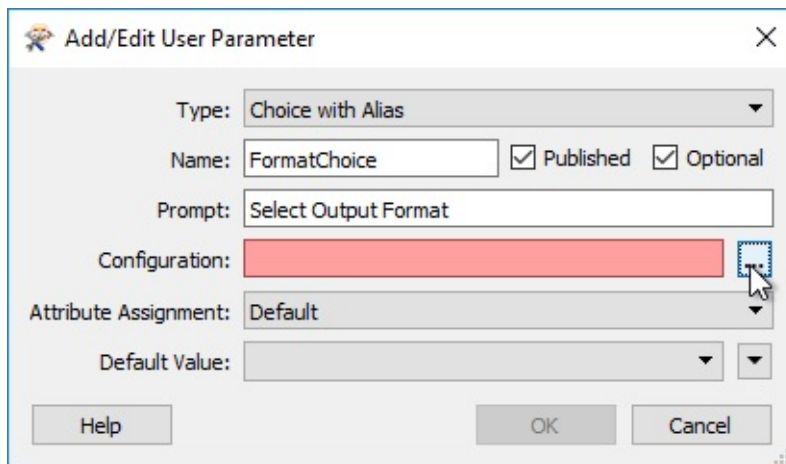
The screenshots above show that, although the format and coordinate system are published, they are plain text-only fields in the FME Server web interface. This would force the end user to manually enter values.

The better solution is to use a Choice With Alias parameter. Not only does that show up as a choice box in the FME Server web interface, but it shows an alias to the user too (so they are presented with Esri Shapefile as the format name instead of SHAPE).

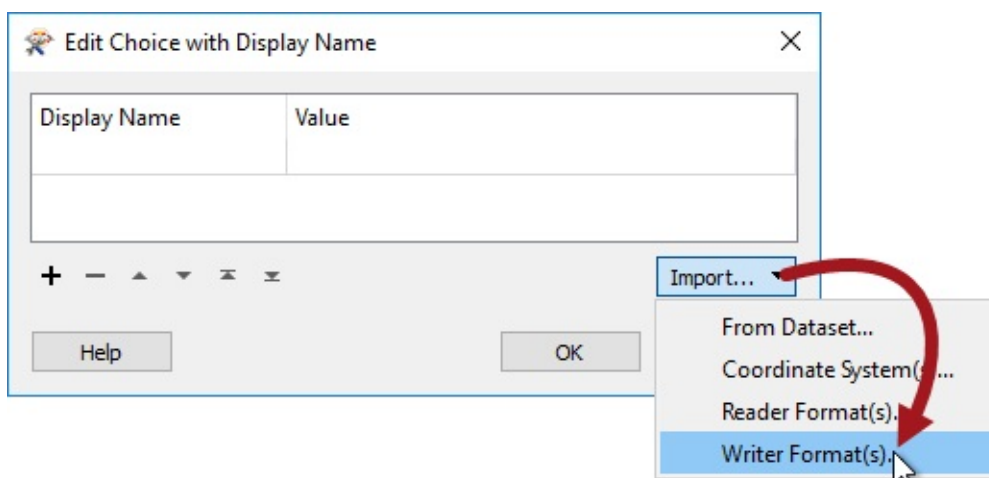
An additional benefit is that the author can reduce the list of formats (or coordinate systems) to a reasonable set of choices, instead of the full list of FME supported formats (or coordinate systems).

The sequence of actions is this:

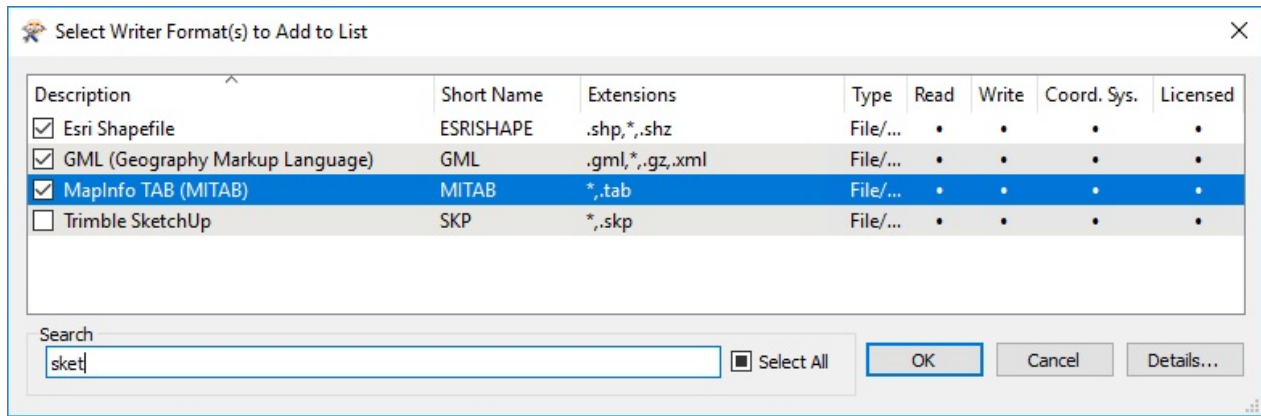
- Use Add Parameter to create a new User Parameter in Workbench. Select the type as Choice with Alias, fill in the other fields, then click the Configuration button:



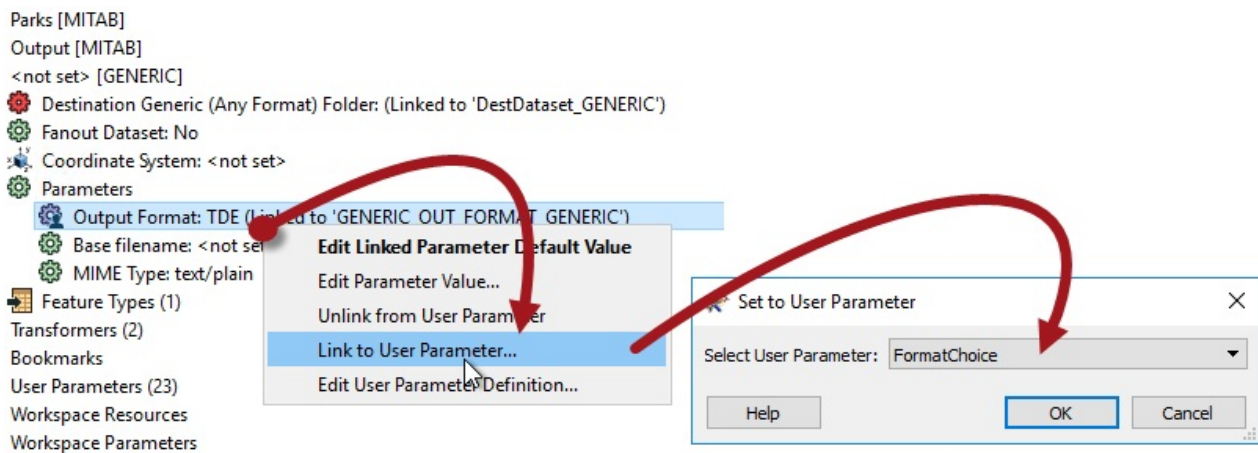
- When creating a Choice with Alias parameter for a format or coordinate system, it's not necessary to manually enter display names and values. Instead choose the import option to select the required formats (or coordinate systems) from a list:



- Then, like this author, search for and select the chosen formats (or coordinate systems):



- Finally you must join the newly-defined Choice with Alias parameter to the appropriate parameter in the FME Navigator:



These selections become part of the parameter definition and allow the user to select their chosen format from within the FME Server web interface:

Published Parameters

Type: Plane Area

Select the size of label to create: [Dropdown]

Destination Generic (Any Format) Folder: [Field] [Menu Icon]

Coordinate System: [Field]

Select Output Format: [Field]

Format Selection List:

- Esri Shapefile
- GML (Geography Markup Language)
- MapInfo TAB (MITAB)

Notice how the list will show only the four selected formats and will pass on the short form of the name, which is what FME expects.

Miss Vector says...

Let me throw an easy question at you! If the Generic Writer parameter is published to determine what format to write data in a data download system, what would the Generic Reader parameter be used for?

- 1. To determine what format of data to read in a Data Download system*
- 2. To determine what format of data to read in a Data Upload system*
- 3. To determine the correct Styler transformer to use in the workspace*
- 4. To determine whether I'm connected to a Data Upload or a Data Download system*

| Exercise 3 Data Download System: Format and Coordinate Systems | |
|--|--|
| Data | Orthophoto images (GeoTIFF) |
| Overall Goal | Create an FME Server Data Download system for orthophotos |
| Demonstrates | Controlling output format and coordinate system in Data Download |
| Start Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe-Ex3-Begin.fmw |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe-Ex3-Complete.fmw |

As a technical analyst in the GIS department of a city you have just commenced a project to allow other departments to download orthophoto data, rather than having to ask you to create it for them. Not only will their requests be processed quicker, you will also spend less time on that task.

So far you have created a simple workspace to translate orthophotos to JPEG format, added published parameters for transformation, and published it to a Data Download service on FME Server.

Now you need to give the end-users control over the output format and output coordinate system.

1) Open Workspace

Open the workspace from exercise 2, or the begin workspace listed above. You can see that it consists of a reader, a writer, and two transformers, plus some published parameters.

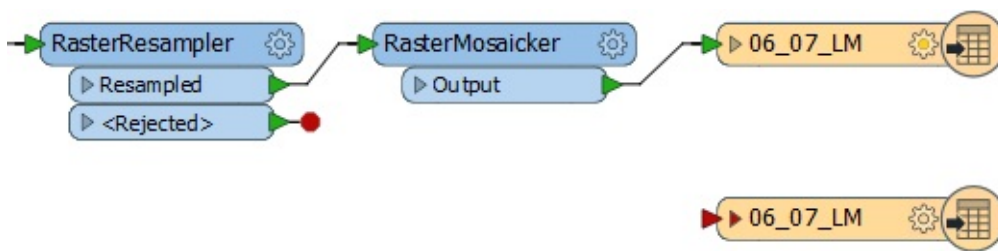
In this step we'll give the end-user control over format and coordinate system.

2) Add Writer

To give control over format you need a Generic format writer. Select Writers > Add Writer from the menubar. When prompted enter these parameters:

| | |
|--------------------------|---|
| Writer Format | Generic (Any Format) |
| Writer Dataset | C:\FMEData2017\Output |
| Writer Parameters | Output Format: JPEG (Joint Photographic Experts Group) MIME Type: img/jpeg |
| Add Feature Types | Copy From Reader |

The MIME type setting doesn't apply for a Data Download service, but we'll set it anyway. It can't hurt. Your workspace will now look like this:



The unconnected feature type belongs to the Generic Writer.

3) Switch Feature Types

We want to write to the Generic Writer, not the JPEG Writer, so switch the connection from the JPEG feature type to the Generic feature type. They are both labelled with the same name, so be sure to inspect their properties to check if you need to. You could (should?) also add an annotation to each, to tell them apart.

Don't delete the JPEG Writer though, or its feature type. We'll need those for reasons to be explained shortly.

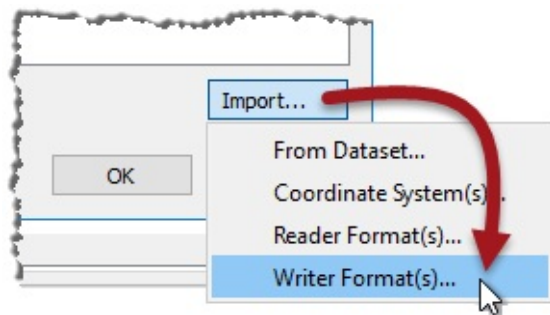
4) Create User Parameter

To give control over format requires a published parameter. So in the Navigator window of FME Workbench, locate the section marked User Parameters. Right-click on there and choose the option Add Parameter.

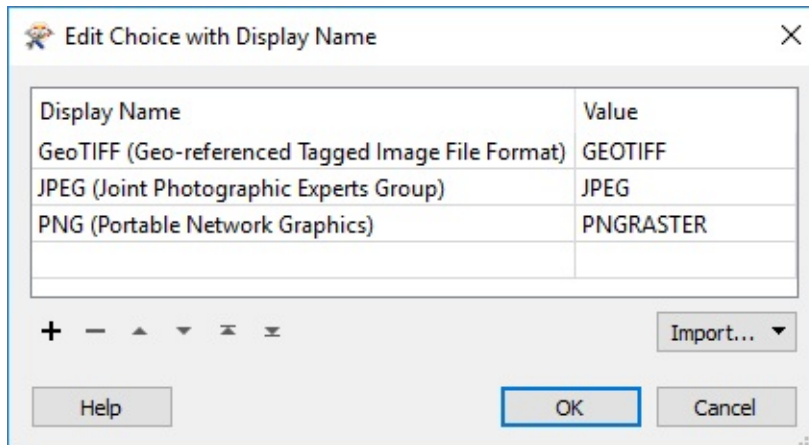
Set the parameter values as follows:

| | |
|------------------|--------------------------|
| Type | Choice with Alias |
| Name | OutputFormat |
| Published | Yes |
| Optional | No |
| Prompt | Select the Output Format |

For the configuration field, click the [...] browse button. In the dialog that opens, click on Import > Writer Format(s):



This will open a list of FME-supported formats. Choose a few simple raster formats such as JPEG, PNG, and GeoTIFF. Click OK to close the dialog and return to the previous one:



WARNING

Be sure to use the PNG format called PNGRASTER. Don't select the format called PNG Rasterizer (PNG) as that is for rasterizing vector data, not writing raster data.

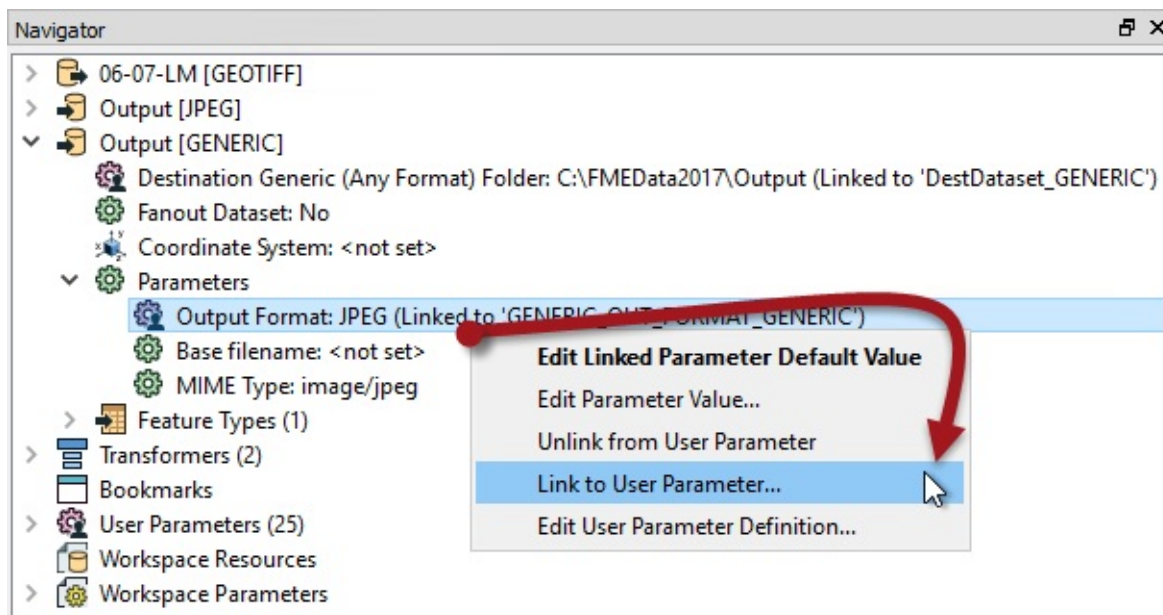
Click OK and OK again to close these dialogs and create the parameter.

5) Apply User Parameter

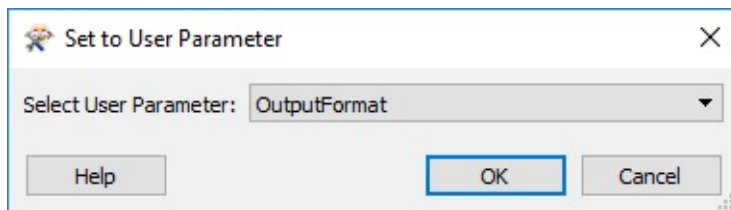
Now that we've created a user parameter, we have to apply it.

Locate the Generic Writer in the Navigator window, expand its parameters and locate the parameter called Output Format. This is already linked to a published parameter that FME created automatically, but we want to ignore that and use our own.

So, right-click on Output Format and choose the option to Link to User Parameter:



When prompted select the OutputFormat parameter that we just created:



The parameter FME created (GENERIC_OUT_FORMAT_GENERIC) will be automatically deleted. FME realizes that we don't need it any more and, since it is used nowhere else, will remove it.

***NB:** If you didn't set a default value for the OutputFormat user parameter, then the Generic writer parameter will turn red (flagged as incomplete). This is nothing to worry about. It will be set at runtime.*

6) Create User Parameter

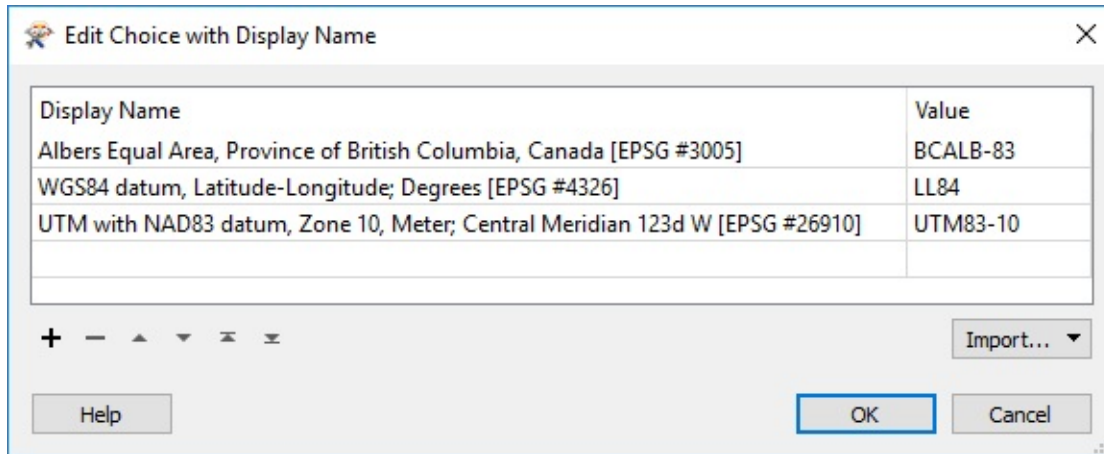
The next parameter required is to give control over output coordinate system. The process is very similar to that of format. In the Navigator window of FME Workbench, locate the section marked User Parameters. Right-click on there and choose the option Add Parameter.

Set the parameter values as follows:

| | |
|------------------|-------------------------------------|
| Type | Choice with Alias |
| Name | OutputCoordSys |
| Published | Yes |
| Optional | No |
| Prompt | Select the Output Coordinate System |

For the configuration field, click the [...] browse button. In the dialog that opens, click on Import > Coordinate System(s). This opens a list of FME coordinate systems. Choose a few simple systems that will apply to this part of Canada, such as LL84, BCALB-83, and UTM83-10.

Click OK to close the dialog and return to the previous one:

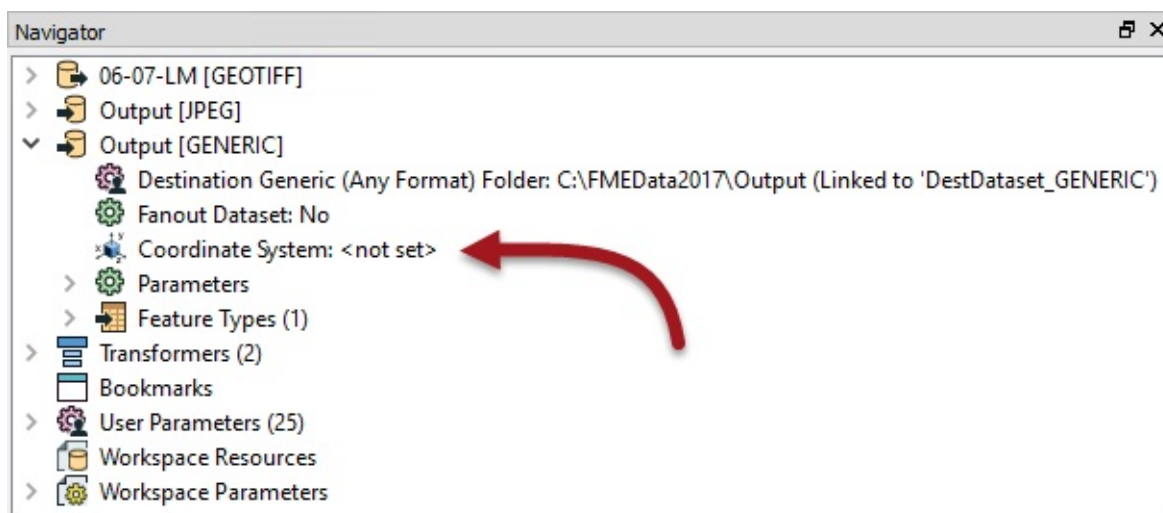


Click OK and OK again to close these dialogs and create the parameter.

7) Apply User Parameter

Once more, now that we've created a user parameter, we have to apply it.

Locate the Generic Writer in the Navigator window, and this time look for the parameter called Coordinate System:

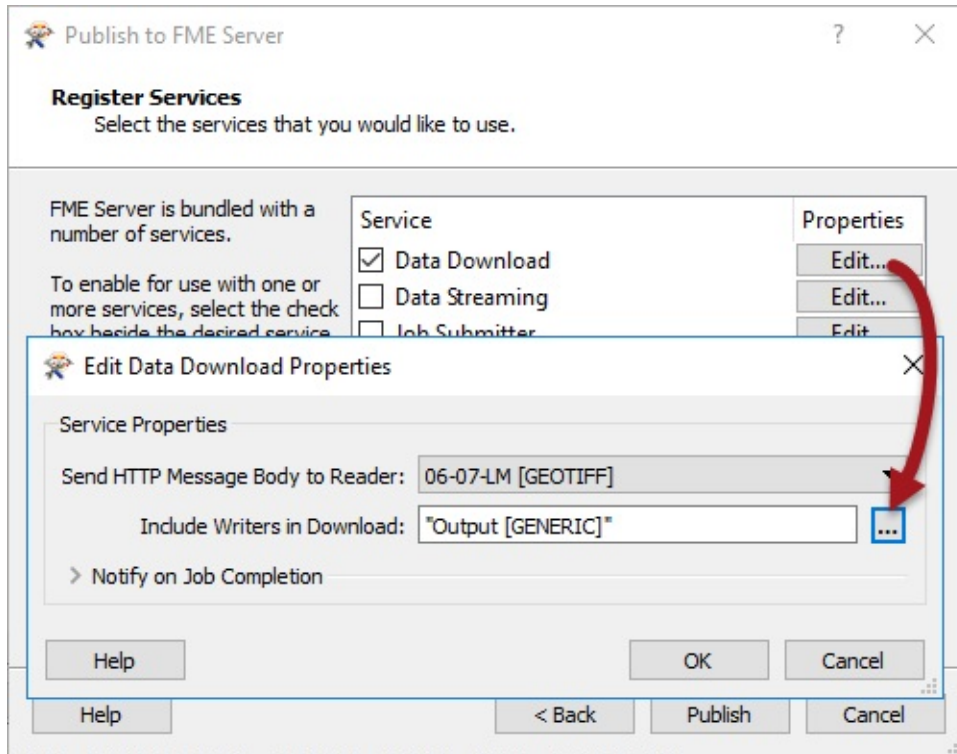


Right-click on this parameter and choose Link to User Parameter. When prompted, select the published parameter called OutputCoordSys that we just created.

If you now use the Run button in Workbench you'll see that both these parameters are now published.

8) Publish to FME Server

Save the workspace and publish it to FME Server. **However!** When you register it with the Data Download service be sure to click the Edit button to edit the service properties. In that dialog you **MUST** set the writer to "Output [GENERIC]" (not "Output [JPEG]").



If you don't do that, then the Data Download will consist of the output of the JPEG Writer. Since that is not connected, there will be no output and so no zip file!

Once published locate the workspace in the FME Server web interface and run it.

Choose different options for output format and coordinate system to see what happens in the output.

Professor Spatial F.M.E., E.T.L. says...

Right now I imagine you have some questions!

Q) Why didn't we delete the original JPEG Writer when we added the Generic Writer?

A) It's because we have a parameter published for JPEG compression. If we deleted the JPEG Writer we would no longer have access to that parameter.

Q) But we're not even using the JPEG Writer anymore, so how would that parameter work?

A) Because the Generic Writer picks up parameters for the format it is writing from any Writer of that format! So you could add a dummy PNG format Writer and the Generic Writer would use the dummy's parameters when writing PNG.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Add a Generic Writer and set up its format and MIME type parameters*
- *Create an output format user parameter and apply it to a Generic Writer*
- *Create an output coordinate system user parameter and apply it to a Generic Writer*
- *Apply a parameter from a dummy Writer to the generic Writer*

Layer Selection and Handling

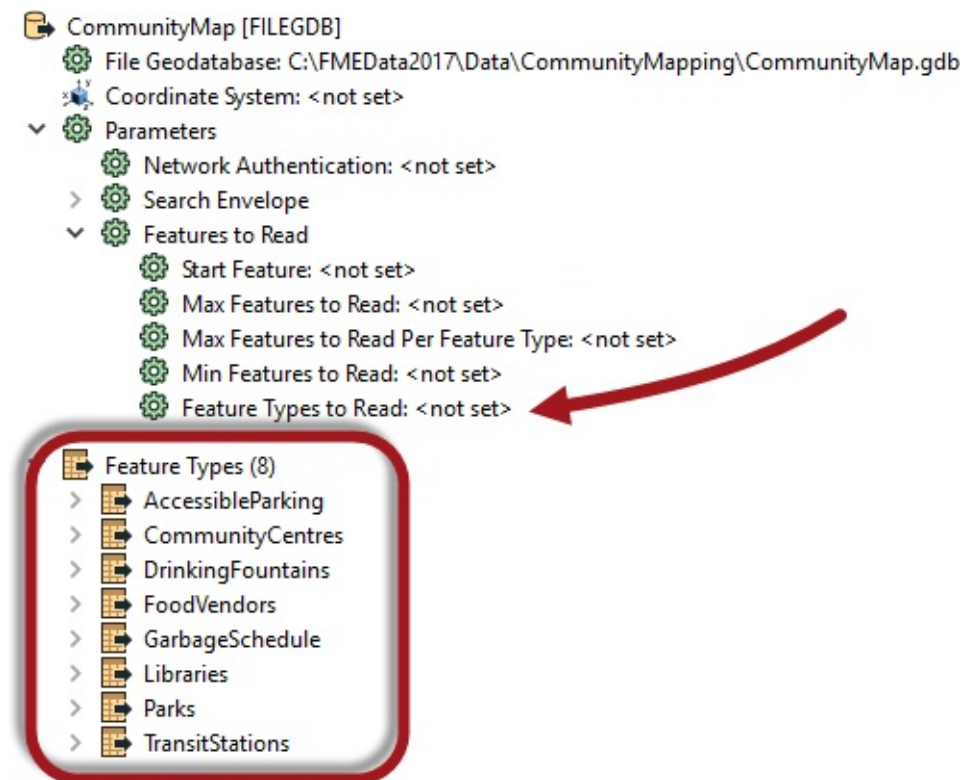
Spatial data is often organized in layers (groups, classes, categories, feature types) and a common way that users will wish to choose data in a self-serve system is on a layer-basis.

In FME, feature type (layer) flexibility can be achieved through the *Feature Types to Read* parameter.

Selecting Layers

Each reader in FME has a parameter called Feature Types to Read. This parameter is used to tell FME which of the feature types in the workspace should be used in the translation.

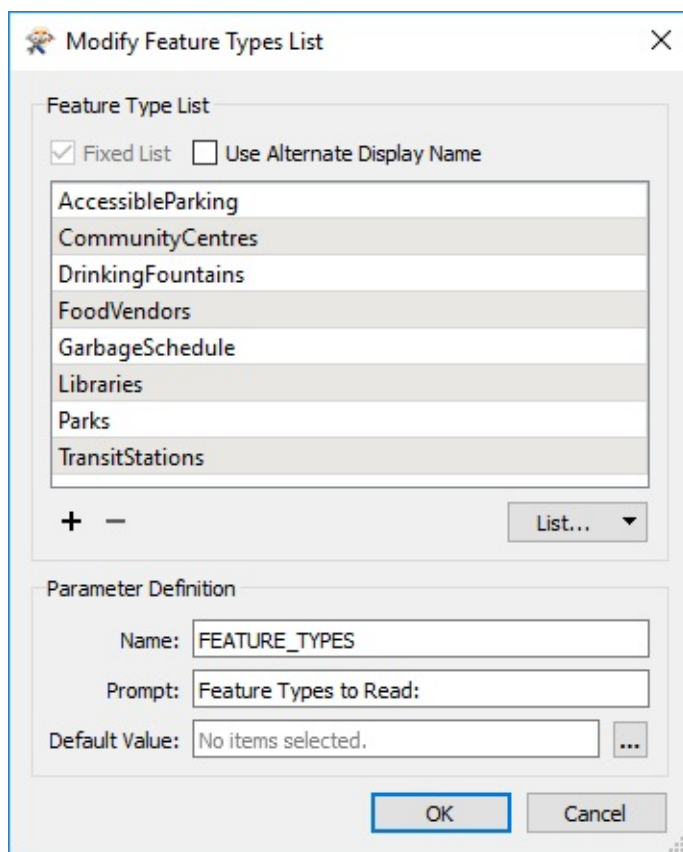
Here a Reader contains eight feature types. The Feature Types to Read parameter has been set up to read just three of these (Libraries, Parks, TransitStations).



The parameter can be set by the workspace author, but in most cases it is published so that the end user can select a list of layers to read.

Publishing Feature Types to Read

Choosing to publish the Feature Types to Read parameter results in a dialog that is significantly different to any other parameter definition:



This dialog is a predefined configuration for the parameter. This is possible because FME already knows what feature types exist in the workspace, whereas for a coordinate system parameter, it has no idea what coordinate systems the author wishes to allow.

If the author wishes to define the parameter with an alias, the checkbox for Use Alternate Display Name allows them to do so.

TIP

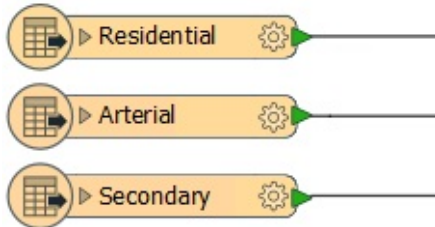
By default the Feature Types to Read list is originally set by the feature types in the workspace, but will automatically update if this is changed. For example, if the Libraries feature type was removed from the workspace then it would also be removed from this parameter.

So the list is fixed to what appears in the workspace.

However, if any of the Reader feature types has a merge filter set, then the list is no longer fixed. It will show all feature types in the source dataset, whether or not they exist in the workspace!

Grouped Layers

On occasion it's better to provide a single choice to the user that represents multiple feature types. For example, in this workspace:



...the author might consider it sufficient to provide an option called "Roads" - that encompasses all three reader feature types - rather than "Residential", "Arterial", and "Secondary" as separate entities. If the user chooses to read "Roads" then all three feature types are read.

The way to do this is by using the Alternate Display Name option, and entering the same value for the Display Name:

Feature Type List

☒ Fixed List ☒ Use Alternate Display Name

| Display Name | Feature Type |
|--------------|--------------|
| Roads | Residential |
| Roads | Secondary |
| Roads | Arterial |

+ - List... ▼

That way the user is presented with groups instead of individual layers:

Published Parameters

Feature Types to Read

Create Labels

Roads X

- Buildings
- Vegetation
- Amenities

Professor Spatial F.M.E., E.T.L. says...

This functionality is useful because of a difference in viewpoint between author and user. The author is asking which layers should be read. But the user is responding to a different question: they are telling us which layers they want to write.

The above example illustrates the case. The user thinks they are choosing to write a layer called Roads, when in reality they are choosing to read three feature types called Residential, Arterial, and Secondary.

| Exercise 4 Data Download System: Layer Selection | |
|--|--|
| Data | Orthophoto images (GeoTIFF) |
| Overall Goal | Create an FME Server Data Download system for orthophotos |
| Demonstrates | Adding vector data. Handling layer selection in Data Download |
| Start Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe-Ex4-Begin.fmw |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe-Ex4-Complete.fmw |

As a technical analyst in the GIS department of a city you have just commenced a project to allow other departments to download orthophoto data, rather than having to ask you to create it for them. Not only will their requests be processed quicker, you will also spend less time on that task.

So far you have created a simple workspace to translate orthophotos to JPEG format. To this you have added published parameters for transformation, format, and coordinate system. The workspace was published to a Data Download service on FME Server.

One of the frequent requests you get when you translate orthophoto data is to add vector data as an overlay on the raster. This is very simple in FME with the VectorOnRasterOverlayer transformer. However, to deploy this on FME Server means you need to give the end-users control over which vector layers are included.

1) Open Workspace

Open the workspace from exercise 3, or the begin workspace listed above. You can see that it consists of a reader, two writers, and two transformers, plus some published parameters.

To add - for example - road features to the raster output first requires a reader for those road features, so that is the first step...

2) Add Reader

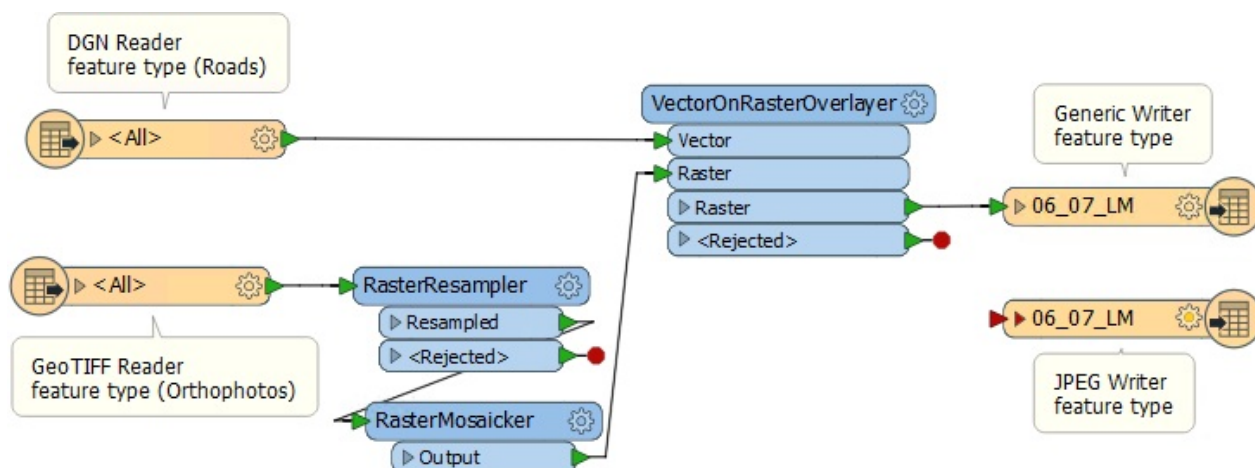
Select Readers > Add Reader and use the following setup:

| | |
|--------------------------|---|
| Reader Format | Bentley MicroStation Design (V8) |
| Reader Dataset | C:\FMEData2017\Data\Transportation\RoadsDGN.dgn |
| Reader Parameters | Group Elements By: Level Names |
| Workflow Options | Single Merged Feature Type |

We'll use the Single Merged Feature Type option here because there are multiple source layers and yet - because they are being overlaid onto the raster as a group - we don't really need to have them divided in any way.

3) Add VectorOnRasterOverlayer Transformer

Add a VectorOnRasterOverlayer transformer. Connect the DGN feature type to the Vector input port, and the output of the RasterMosaicker transformer to the Raster input port:

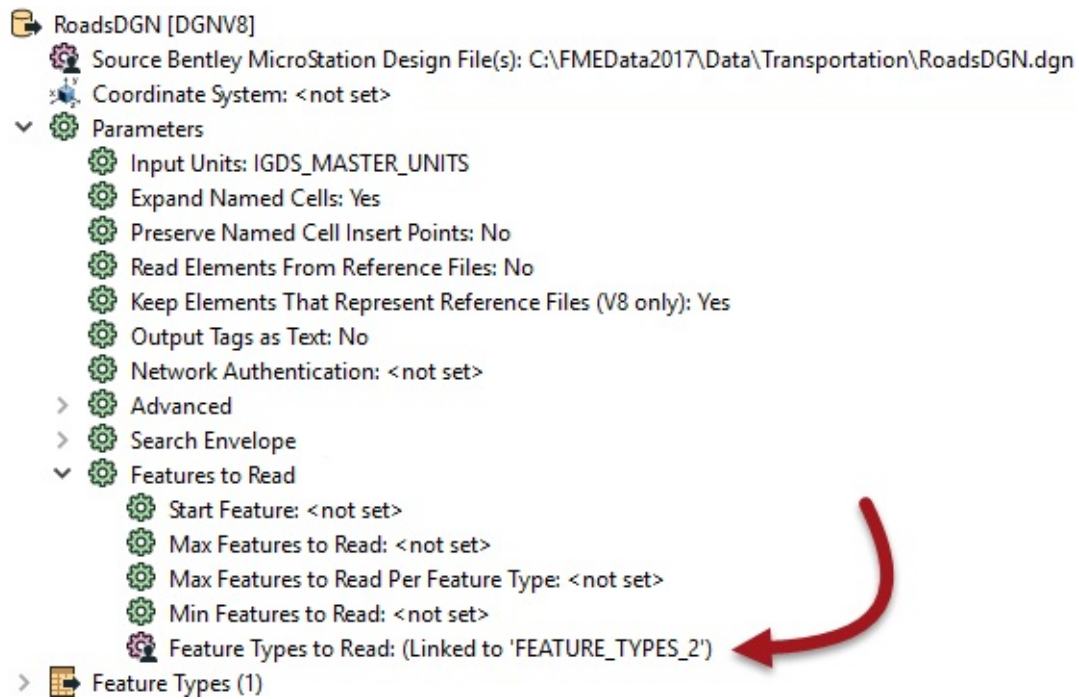


You can check the parameters for this transformer but, for now at least, we'll leave them as they are.

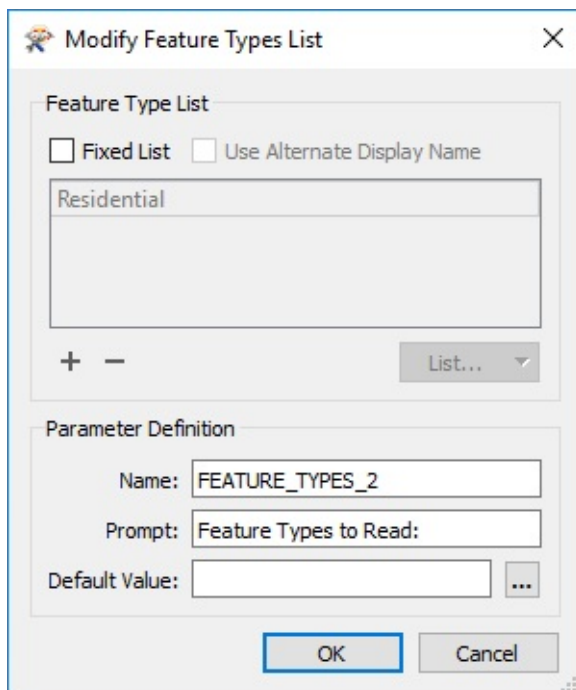
4) Create User Parameter

Now that we have some source data we can create a parameter to control which layers in that data should be read.

In the Navigator window find the DGN Reader's parameters, expand the Features to Read section, and locate the parameter called Feature Types to Read. You will see that it is already published - a result of us using the Single Merged Feature Type:



Right-click on the parameter and choose Edit User Parameter Definition. This will bring up a dialog that looks like this:



Professor Spatial F.M.E., E.T.L. says...

The Feature Types to Read parameter tells FME which layers to read from the source. When you use a Single Merged Feature Type (or manually set a merge filter) it is published automatically. It also is set to update automatically.

This means when the end-user is prompted to select feature types, FME will automatically scan the source dataset for the list. This is particularly useful for databases, where the table list will often change. However, for this project we're going to assume a fixed list of feature types.

5) Edit User Parameter

In the Modify Feature Types List dialog, check the box that is labelled Fixed List, and also the box that is labelled Use Alternate Display Name.

Click List > Add From Current Dataset and - when prompted - select all of the feature types in this dataset. Click OK and the dialog will now look like this:

Modify Feature Types List

Feature Type List

☒ Fixed List ☒ Use Alternate Display Name

| Display Name | Feature Type |
|--------------|--------------|
| | Arterial |
| | Collector |
| | NonCity |
| | Other |
| | Private |
| | Secondary |
| Residential | Residential |

+ - List...

Parameter Definition

Name:

Prompt:

Default Value:

OK Cancel

What we could do is just create a display name for each road type and close the dialog. The end-user will then be able to select any of the individual layers. However, for this project I think we should give them a simpler choice, and we will do that by grouping the layers.

So, under the Display Name, enter values to match as follows:

| Display Name | Feature Type |
|-----------------|--------------|
| Primary Roads | Arterial |
| Primary Roads | Collector |
| Other Roads | NonCity |
| Other Roads | Private |
| Secondary Roads | Residential |
| Secondary Roads | Secondary |
| Other Roads | Other |

The list will sort itself by display name and look like this:

Feature Type List

☒ Fixed List ☒ Use Alternate Display Name

| Display Name | Feature Type |
|-----------------|--------------|
| Other Roads | Other |
| Other Roads | NonCity |
| Other Roads | Private |
| Primary Roads | Collector |
| Primary Roads | Arterial |
| Secondary Roads | Residential |
| Secondary Roads | Secondary |

+ - List...

What this will do is give the user a choice of three options: Primary Roads, Secondary Roads, Other Roads. Whichever they choose will return all of the source layers for that choice.

One final task. In the lower part of the dialog, change the prompt to something like "Vector Roads to Overlay":

Parameter Definition

Name:

Prompt:

Default Value: ...

It's just a small thing but will help with the end user experience.

6) Save and Run Workspace

Save the workspace and then run it in FME Workbench to test it. You should be able to

select any of the three types of roads, or even none of them.

Check that the output includes whatever roads you selected.

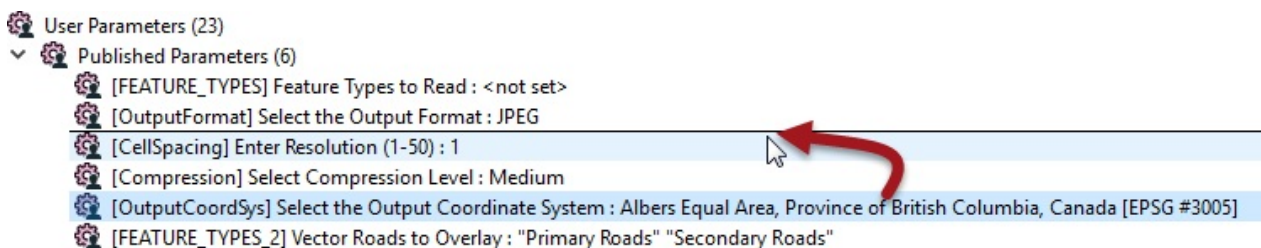
7) Clean User Parameters

If your workspace is like mine, there are a number of extra published parameters we don't really need right now. Plus the order of parameters is not good. Let's take this opportunity to clean it up.

Locate and delete the following published parameters:

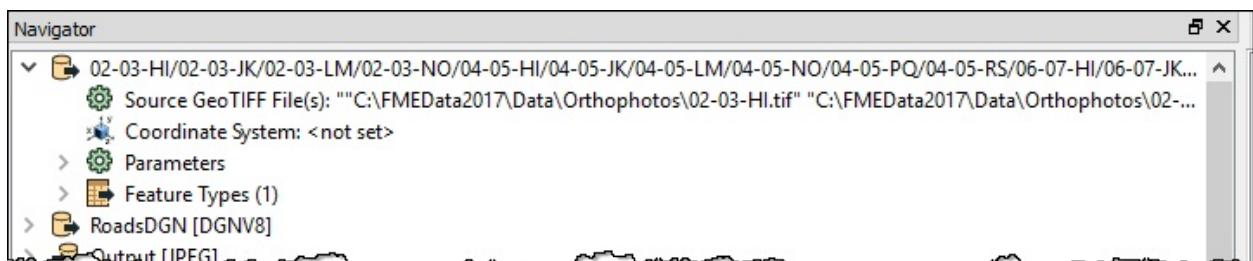
- SourceDataset_GEOTIFF
- SourceDataset_DGNV8
- DestDataset_JPEG
- DestDataset_GENERIC

Finally, let's change the order of parameters. You can do this by dragging one above the other in the Navigator window. So do this and put the parameters in the order that seems best to you:



8) Raster User Parameters

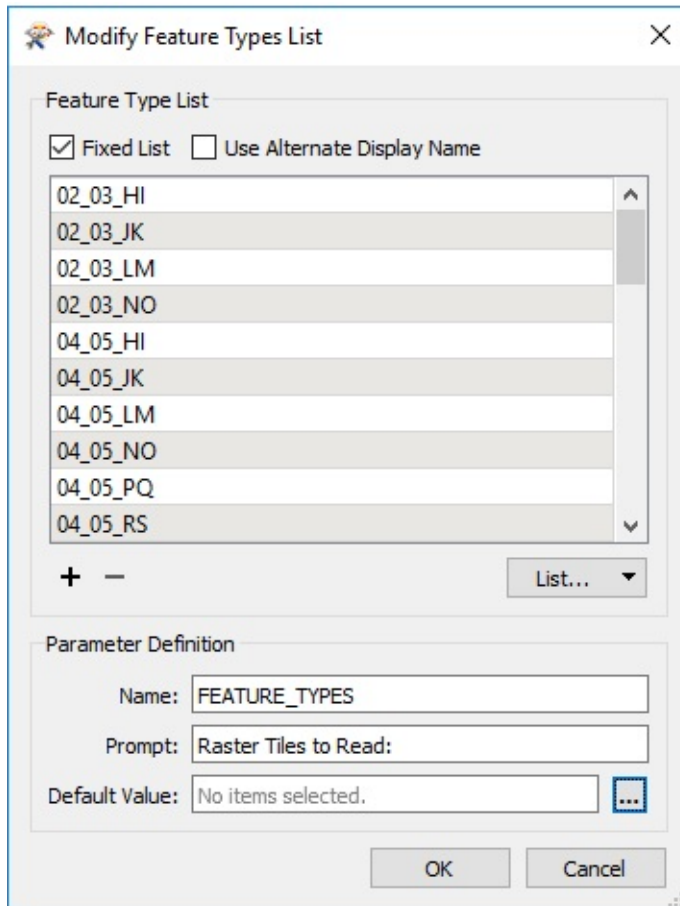
Now let's do something with the source raster. We want the user to be able to select the files to read, without having to upload them. Locate the reader in the Navigator window and double-click the Source GeoTIFF File(s) parameter. When prompted, select all of the GeoTIFF files in the Orthophotos folder.



This would normally mean that all files would get read into the translation; but the Feature Types to Read parameter will let the user actually choose which ones to read. We do, however, need to make some edits.

Open the definition of the GeoTIFF Feature Types to Read parameter. Click the option for a Fixed List. To list all of the available tiles select List > Add From Current Dataset and select all the files.

Finally change the prompt to something sensible like "Raster Tiles to Read".



Now run the workspace again to check on our improved parameters dialog.

WARNING

At this point you might find, on running the workspace, that no filenames appear. Instead all you get is a single option called "Geotiff" in the file prompt.

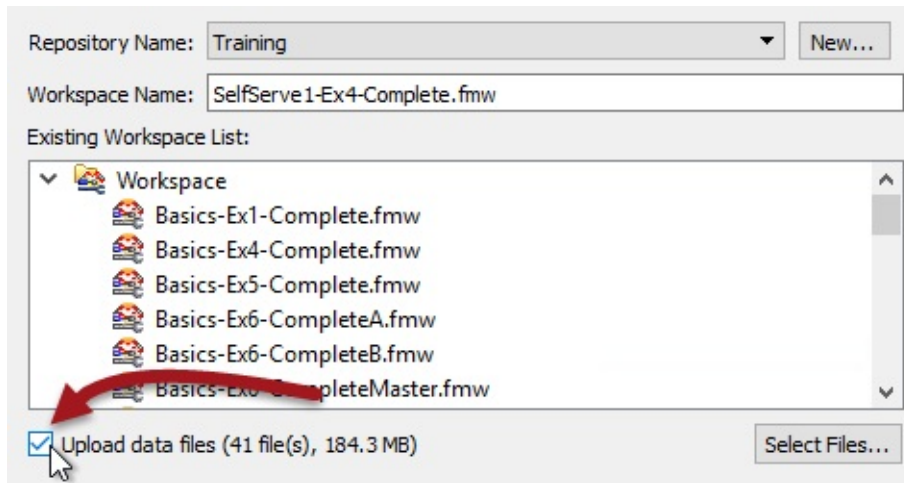
The reason for that is you made a mistake way back in exercise 1, step 1. You missed out setting the parameter "Feature Type Name(s)" to "From file name(s)".

The only solution to this is to delete the GeoTIFF reader, and re-add it, this time being sure to set that parameter! Then you'll also have to clean up the reader's published parameters too.

9) Publish to FME Server

Save the workspace and publish it to FME Server. There are two things to note.

Firstly, because we removed the Source Dataset parameters FME will suggest we upload the data. If your data is on the same computer as FME Server (or on a path otherwise accessible to the Server), then you don't need to do this and can uncheck that box:



If the files aren't accessible, then you will have to upload them all - but at least the end user will never have to.

Secondly, remember to make sure the Data Download service is using the "Output [GENERIC]" writer.

In the FME Server web interface, run the workspace, taking time to admire the new, cleaner set of parameters that are available:

Published Parameters

| | |
|--|--|
| Raster Tiles to Read | 08_09_HI × 08_09_PQ × 08_09_JK × |
| Select the Output Format | JPEG (Joint Photographic Experts Group) ▼ |
| Select the Output Coordinate System | Albers Equal Area, Province of British Columbia, Canada [EPSG #3005] ▼ |
| Enter Resolution (1-50) | 1 ▲▼ |
| Select Compression Level | Medium ▼ |
| Vector Roads to Overlay | Primary Roads × Secondary Roads × |

Outside of a training environment we might want to order the raster tiles into groups, but we'll live with it as-is for now.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Add vector data onto raster*
- *Use the Feature Types to Read parameter in automatic mode*
- *Edit the Feature Types to Read parameter to create a manual list*
- *Edit the Feature Types to Read parameter to create a grouped manual list*
- *Clean up unnecessary user parameters and change the display order*

Troubleshooting for Administrators

This section shows a few basic troubleshooting techniques in case of emergency.

Data Download: No Output

If no zip file is output at the end of a Data Download process then the following suggestions may be of help.

- Run the workspace on FME Desktop to confirm that it does actually write some data. If there is no output data then no zip file will be delivered.
- Check that the FME Server has access to the source data. If the source data is not available to FME Server then there will be no output.
- On FME Server check the log file using Jobs > Completed to confirm that the process did actually try to write some data. If not an error message may help to indicate why.

Data Download: Empty Output

If an empty zip file is output at the end of a Data Download process then the following may be of help.

- Ensure that when you publish the workspace to FME Server you check the Data Download parameters and choose an output dataset to be associated with the Service. If you don't, then FME has no way to determine which Writer will provide the output.

Module Review

This module introduced you to FME Server's Self-Serve Services and related functionality in FME Workbench.

What You Should Have Learned from this Module

The following are key points to be learned from this module.

Theory

- Services handle communication between FME Server and its clients. Each workspace published to FME Server can be registered with any number of services.
- The Data Download service overrides the destination dataset parameter and presents data to the end-user as a link to a zip file.
- Parameters control the different actions of a Data Download system, including the output coordinate system.
- The Generic Writer is a writer whose format is only defined at run time. This can be provided by the user through a published parameter.
- Selection of source data by layer is achieved using the Feature Types to Read parameter

FME Skills

- The ability to publish a workspace and register it to the FME Server Data Download service
- The ability to create and use published parameters
- The ability to give the end-user control over format, coordinate system, and layers

Further Reading

For further reading why not check out [this guide to sharing open data](#) using FME Server, or [this fine blog article](#) that explores how to implement an open-data portal using FME, Amazon S3, Leaflet, JQuery, and a whole lot more!

Questions

Here are the answers to the questions in this chapter.

Miss Vector says...

For each of these scenarios, tell me if it is a Data Download project, Data Upload, both, or neither.

- 1. The user logs on to a web page, draws an area on the map, and is sent a copy of the data within that area: **Data Download***
- 2. The user submits a dataset to a web site that scans the data for errors, and returns a corrected copy: **Both Data Upload and Data Download***
- 3. The user publishes a workspace that writes data to the user's account in an online PostGIS database: **Neither Data Upload or Data Download***
- 4. The user starts a GIS application, clicks File > Add Data to Map, and pastes in a URL from FME Server: **Data Download***

#1 is a simple download of data. #2 is obviously both. #3 is neither. Publishing a workspace is not Data Upload and writing to a database is not Data Download. #4 would be a special type of Data Download. In FME we call it Data Streaming; the workspace runs and the output is sent directly to the application that requested it.

Miss Vector says...

When a workspace is not registered against any service, how can you run it? Select all that apply.

- 1. With the FMEServerJobSubmitter transformer***
- 2. With the run dialog in the web interface*
- 3. With the URL specified under Developer Information in the run dialog*
- 4. By setting it to run under a schedule***

You cannot run the workspace in the web interface (the run button won't work unless services are available) or with a URL (none will be provided).

Miss Vector says...

When a workspace is registered against the Data Download service (and no other), how can you run it? Select all that apply.

1. **With the FMEServerJobSubmitter transformer**
2. **With the run dialog in the web interface**
3. **With the URL specified under Developer Information in the run dialog**
4. **By setting it to run under a schedule**

It will actually run under all of these tools. Of course, only the web dialog and the URL return a zip file for download. The others just output the data to the specified workspace location.

Miss Vector says...

How well do you know the types of FME published parameters? Decide which of the following are real parameters, and which are fake.

1. Color: **Real**
2. Double: **Fake**
3. Password: **Real**
4. Text (Multiline): **Real**

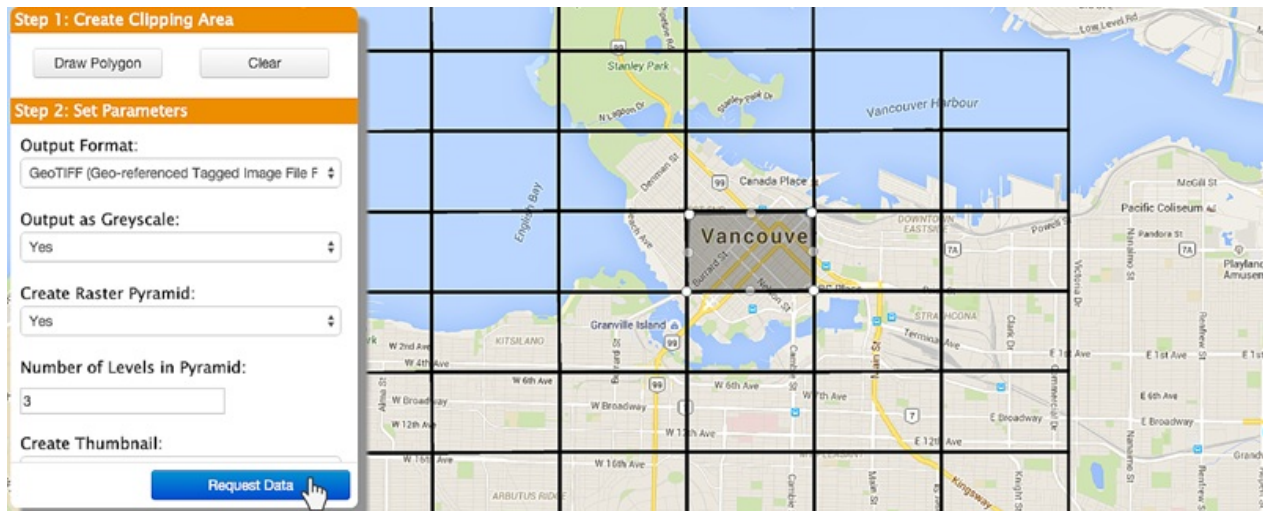
Miss Vector says...

Let me throw an easy question at you! If the Generic Writer parameter is published to determine what format to write data in a data download system, what would the Generic Reader parameter be used for?

1. To determine what format of data to read in a Data Download system
2. **To determine what format of data to read in a Data Upload system**
3. To determine the correct Styler transformer to use in the workspace
4. To determine whether I'm connected to a Data Upload or a Data Download system

Right, I'm uploading some data and I'm going to tell FME Server what format it is. Why would I care (#1) about what format of data is being read in a Data Download system (and how could I tell)? Similarly (#3) it's the output format that determines which Styler to use, not the input format. #4 is just plain nonsense!

Self-Serve with FME Server



Remember, Self-Serve is the key technique for taking the burden of user requests away from expert staff, to enable them to concentrate on more important things.

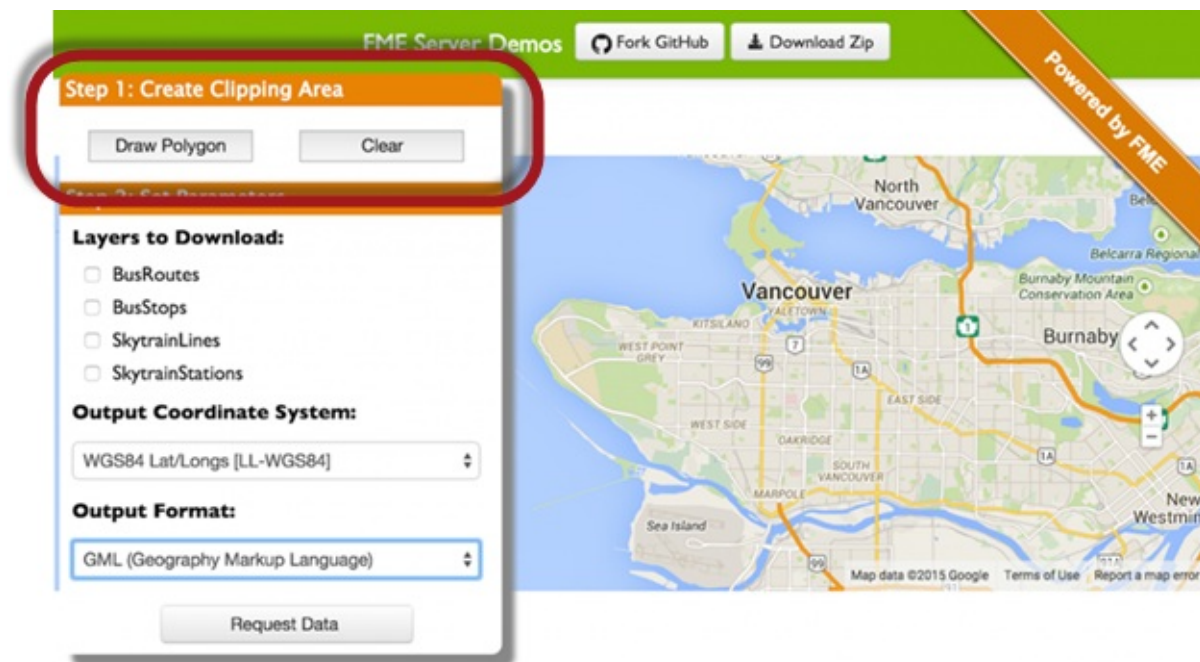
This chapter looks at how to allow the end user to define a geographic area to download, and examines services other than the plain Data Download service.

Geographic Selection

Although most workspaces are designed around FME feature types, this may not be how the end-users see the data, or how they need to select it. An installation that provided selection only by layer (or even a group of layers) would be very limiting indeed.

Besides layers, other methods of data selection revolve around spatially defining an area of interest. There are three obvious ways of doing so:

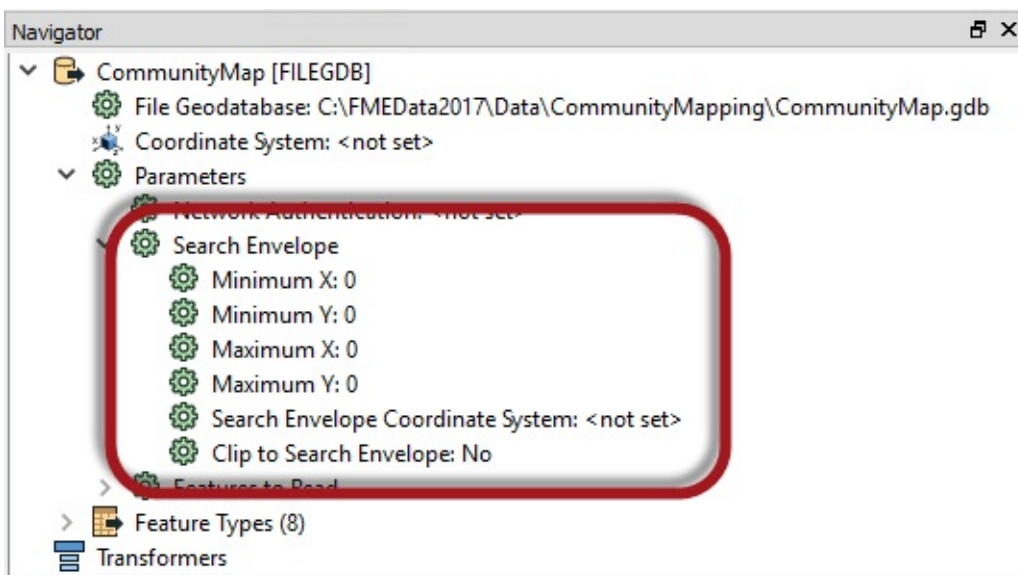
- A simple, rectangular bounding box
- An existing boundary (any area feature such as a regional or municipal boundary)
- An ad-hoc boundary (an area drawn by the user as needed)



Bounding Box

A **bounding box** - also known as a **search envelope** - is simply a rectangular area that defines a geographic area. In FME a bounding box can be defined in several ways, but the easiest is to use a set of parameters.

All FME readers have parameters to define the bounding box (envelope) of data that is being read:



The parameters include not just the coordinates of the bounding box, but also a parameter to define the coordinate system. The coordinate system for the bounding box does not have to be the same as the coordinate system the data is stored in, meaning tools that return lat/long coordinates can be used to clip data stored in UTM (very useful for mobile FME Server solutions).

There is also a parameter specifying whether to clip features to the exact envelope boundary. If set to No then features that overlap the boundary will be included in their full (unclipped) form. For a Data Download service it's more likely that this parameter should be set to Yes.

The workspace author can manually set these parameters, but by publishing them the end-user is able to enter values that define the extent of the data he/she is interested in.

Ms. Analyst says...

Using reader parameters is the most efficient method of selecting an area of interest because it can mean FME does not have to read the entire dataset first and then clip it to size; it can read only what data is necessary. It applies to both vector and raster datasets and is particularly efficient if the source format has a spatial index.

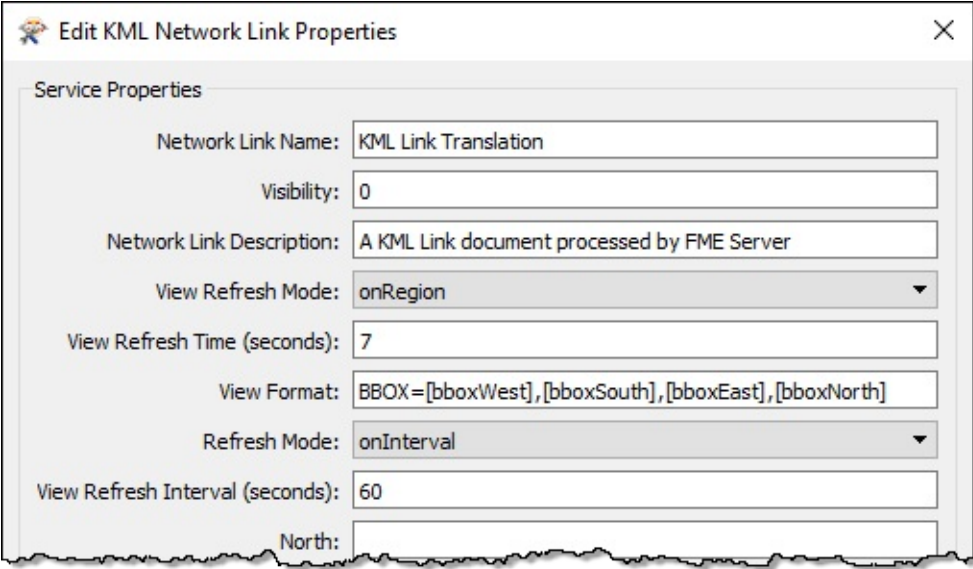
Web Mapping and Bounding Boxes

Bounding box parameters are particularly useful when a web mapping application (for example Google Earth) is requesting data. The application will provide the extents of the current map view and the workspace restricts itself to reading features within those extents.

This, of course, is done using published parameters. The parameters in the workspace need to be given specific names when published, as follows:

- bboxWest
- bboxEast
- bboxNorth
- bboxSouth

A KML Network Link service will automatically make use of these parameters. In fact, if you click the Edit button for KML Network Link properties (in the Publish to FME Server wizard), you will notice how these are used to pass information back to the workspace:



Edit KML Network Link Properties

Service Properties

Network Link Name: KML Link Translation

Visibility: 0

Network Link Description: A KML Link document processed by FME Server

View Refresh Mode: onRegion

View Refresh Time (seconds): 7

View Format: BBOX=[bboxWest],[bboxSouth],[bboxEast],[bboxNorth]

Refresh Mode: onInterval

View Refresh Interval (seconds): 60

North:

Existing Boundaries

By **existing boundary** we mean the perimeter of a non-rectangular geographic area; for example a regional boundary, census area, or similar. In FME these are represented by existing polygon features.

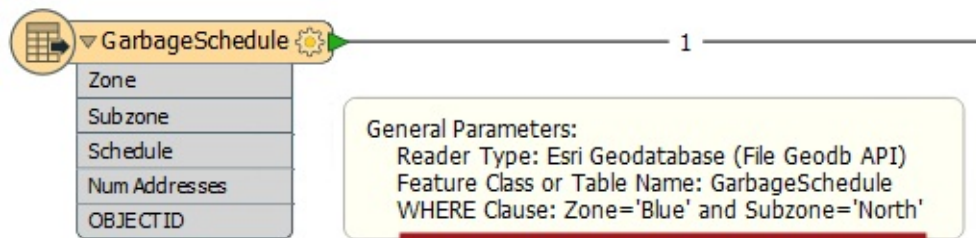
Any polygon feature can be read into a workspace, but its perimeter cannot be used to clip features being read into the workspace via a *reader*. However, it can be used to clip features being read into the workspace via a *transformer*.

So, instead of using a reader and its parameters, an existing boundary is used to clip data either with a FeatureReader transformer or with a Clipper transformer.

Selecting the Existing Area

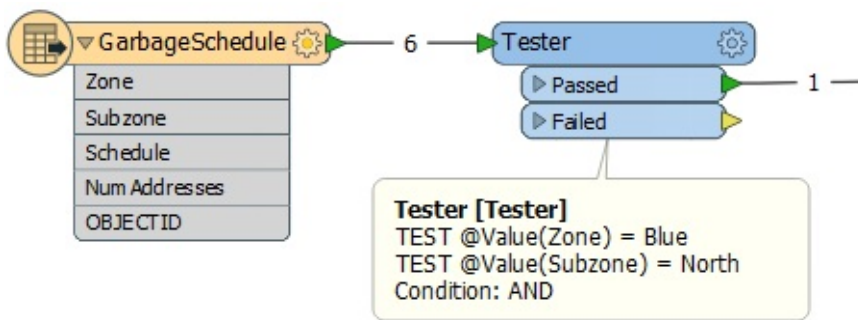
If the existing area feature is a single feature in a dataset, then it can be read into the workspace and used immediately.

If, however, it is part of a larger dataset, then it needs to be filtered from the rest of the data. If the reader has a where clause (and the feature can be identified in that way) then it is the most efficient way of filtering the data:



Here the data from a Geodatabase is filtered down to the required feature using a WHERE clause. The feature count on the connection shows only a single feature passed.

If the Reader has no WHERE clause parameter, then the full dataset can be read and the required feature filtered out using transformers such as the Tester or TestFilter:



However, this technique is not as efficient, because - in the above example - all six features needed to be read instead of just one.

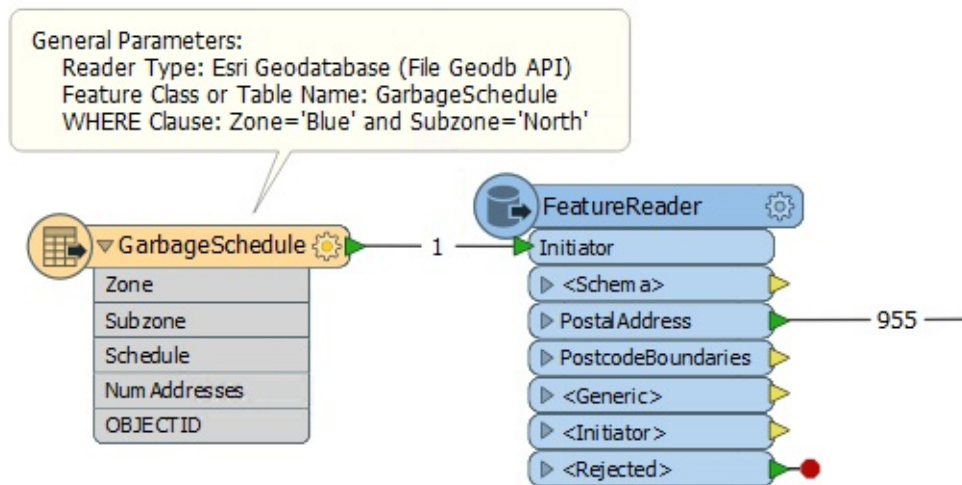
FeatureReader

To use a FeatureReader to filter data by an existing area, the polygon feature is routed into the FeatureReader Initiator port.

Here a single garbage schedule area (Blue zone, north) is passed to the FeatureReader for use as a filter. The idea is to return only addresses that fall inside that garbage collection zone. The parameters are set up like this:

Notice that the FeatureReader is set to read from an address geodatabase. The Spatial Filter parameter tells the transformer to only read features (addresses) inside the incoming polygon feature (garbage collection zone).

These features are output through a port dynamically added to the FeatureReader:

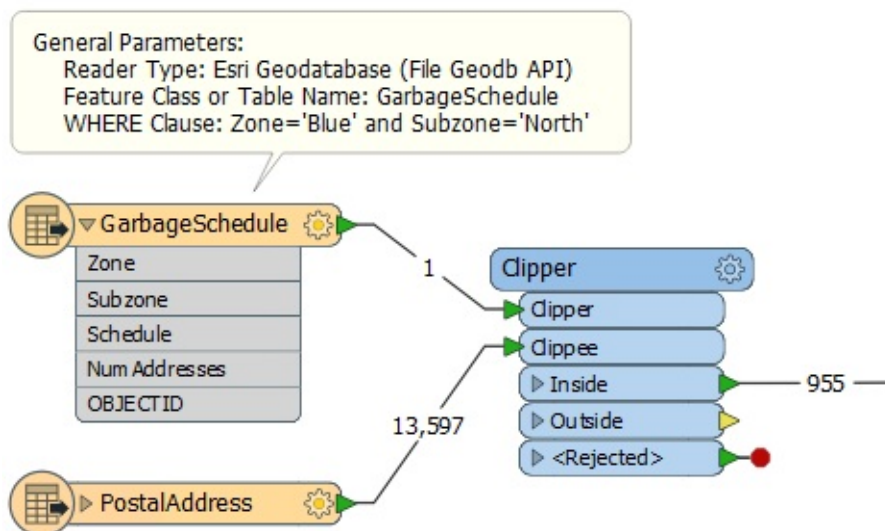


The feature counts here show that there are 955 addresses inside that garbage collection zone. In case you noticed that the GarbageSchedule already has a NumAddresses attribute, yes, it is the same value!

Clipper

The Clipper transformer is a means to spatially filter data that is already read into a workspace (other similar transformers are the PointOnAreaOverlayer, or the SpatialFilter).

Here the address database is read into a workspace and the Clipper used to filter out those addresses that fall inside the chosen garbage collection zone:



Ms Vector says...

In the above examples, the results of the Clipper and FeatureReader are exactly the same. So why is the FeatureReader the preferred option? Pick all the reasons that apply:

- 1. It can be quicker and more resource efficient*
- 2. It allows multiple areas to be used as the existing areas*
- 3. It works with raster data*
- 4. It has more choices for spatial filtering*

Ad Hoc Boundaries

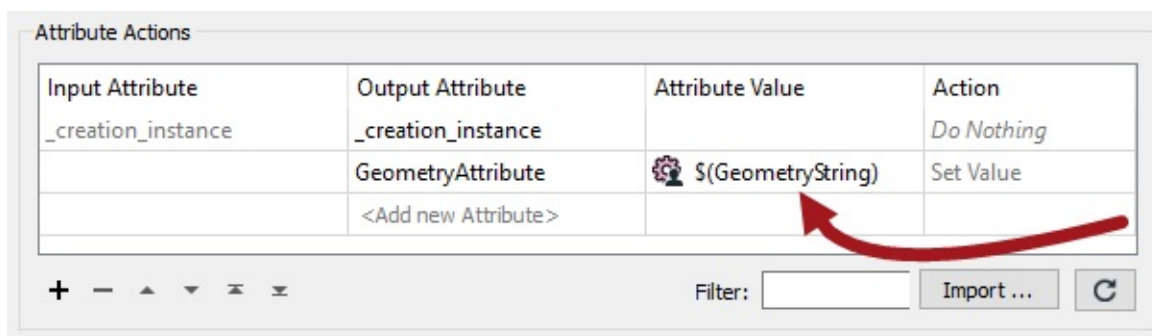
By **ad-hoc boundary** we mean that the geographic area for data download is not a previously known shape. Sometimes it is simply defined by the user on a web map as it is needed.

In this scenario a different technique is needed for your FME workspaces, one that involves a published parameter.

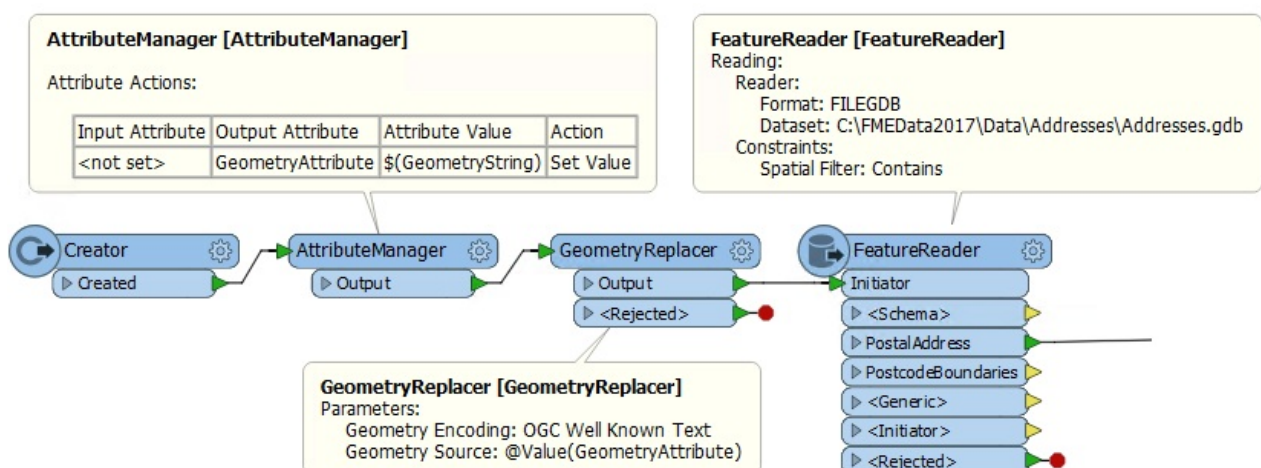
Passing Ad Hoc Boundaries to FME

The way to pass an area boundary to FME is with the geometry defined as a string. The geometry could be, for example, either WKT (Well-Known Text) or XML. The string is passed to FME using a published parameter.

One simple method would be to use an AttributeManager transformer and publish the Value part:



This information could be used in a sequence like this; the translation is triggered with a Creator, the geometry string retrieved with the AttributeCreator, the attribute contents converted into true geometry with a GeometryReplacer, and then the real data read with a FeatureReader:



With these methods, a web mapping interface that allows the user to define their own custom area of interest can pass it to the FME workspace to be used as the area of interest in a Clipper or FeatureReader.

Ms Vector says...

If the incoming geometry string is XML, there is a shortcut to the above example. What is it?

| Exercise 1 | Data Download System: Geographic Selection |
|------------------------|---|
| Data | Orthophoto images (GeoTIFF) |
| Overall Goal | Create an FME Server Data Download system for orthophotos |
| Demonstrates | Handling selection by geographic area |
| Start Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe2-Ex1-Begin.fmw |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe2-Ex1-Complete.fmw |

As a technical analyst in the GIS department of a city you have just commenced a project to allow other departments to download orthophoto data, rather than having to ask you to create it for them. Not only will their requests be processed quicker, you will also spend less time on that task.

You've implemented a lot of different options for transformation, format, coordinate system, and layers to process. However, end-users also often ask for raster data for a particular neighborhood of the city, and that's easy to do using a Clipper transformer.

1) Open Workspace

Open the begin workspace listed above. You can see that it consists of a reader, two writers, three transformers, and various published parameters.

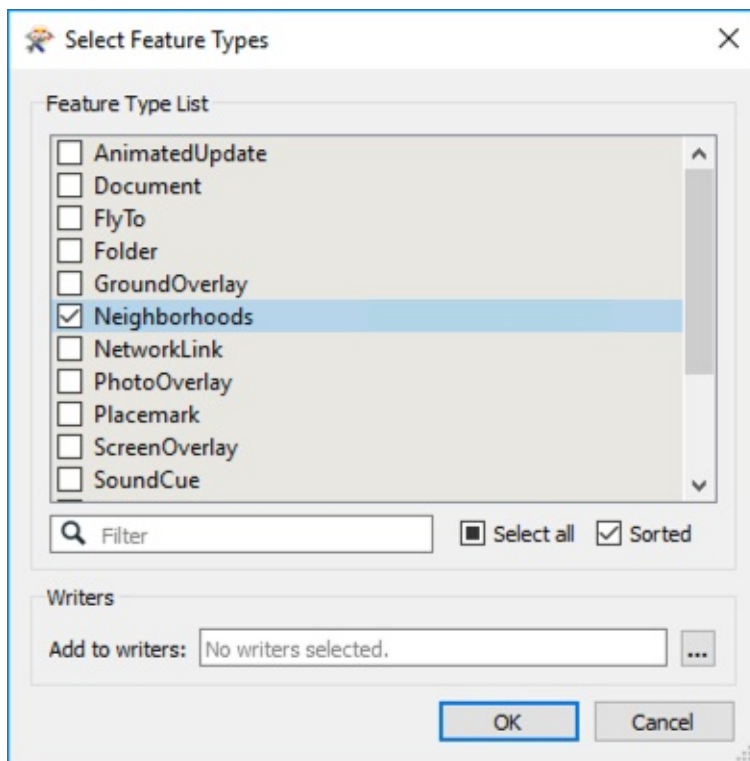
To clip data to a particular neighborhood first requires a reader for those neighborhood features, so that is the first step...

2) Add Reader

Select Readers > Add Reader and use the following setup:

| | |
|-----------------------|---|
| Reader Format | Google KML |
| Reader Dataset | C:\FMEData2017\Data\Boundaries\VancouverNeighborhoods.kml |
| Workflow | Individual Feature Types |

Be sure to set the workflow option if you carried out the previous exercise, as it might default to a different value. Click OK and, when prompted, select only the feature type for Neighborhoods:



Once added, remove the published parameter for SourceDataset_OGCKML. We don't need to prompt the user to select this dataset.

3) Add Published Parameter

Currently the KML Reader will read all of the neighborhoods. However, we need the user to select one of these to clip the data with. To select the neighborhood we'll use a published parameter.

So, add a new parameter. Set the parameter values as follows:

| | |
|------------------|-------------------------|
| Type | Choice |
| Name | Neighborhood |
| Published | Yes |
| Optional | Yes |
| Prompt | Select the Neighborhood |

For the configuration field, click the [...] browse button. In the dialog that opens, enter the names of the neighborhoods of Vancouver. These are:

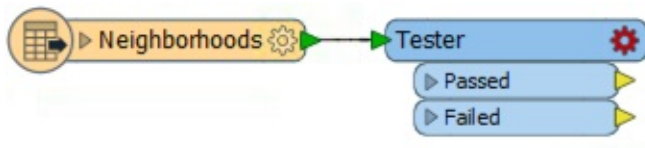
- Downtown
- Fairview
- Kitsilano
- Mount Pleasant
- Strathcona

- West End

Notice that this parameter is optional. The user should not have to select a value if they don't want to. Also, this is a choice field alone; an alias is not needed because the proper values are clear enough.

4) Add Tester

Now we need to filter the neighborhood data by the user's choice. So add a Tester transformer to the workspace, connected to the Neighborhood feature type:



Inspect its parameters and set them up to test where NeighborhoodName = the neighborhood published parameter:

The screenshot shows the 'Test Clauses' configuration window. It contains a table with the following data:

| | Left Value | Operator | Right Value | Negate | Mode |
|---|------------------|----------|------------------|-------------------------------------|-----------|
| 1 | NeighborhoodName | = | \$(Neighborhood) | <input checked="" type="checkbox"/> | Automatic |

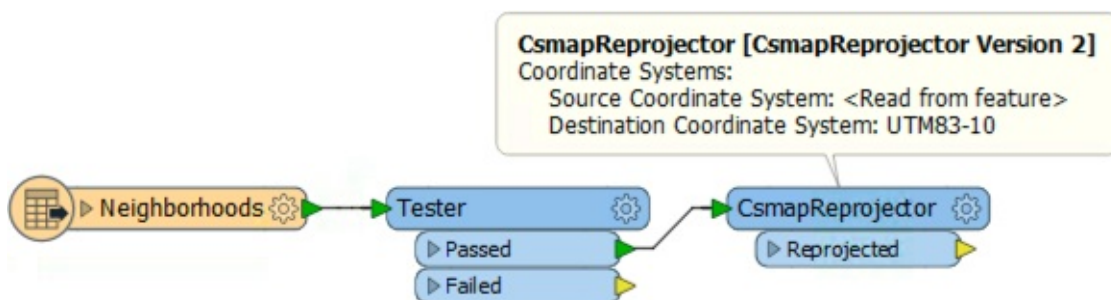
Below the table are several control icons: a plus sign, a minus sign, a left arrow, a right arrow, a double left arrow, and a double right arrow. A 'Duplicate' button is located at the bottom right of the window.

Save the parameter changes.

5) Add CsmmapReprojector

One interesting part of the neighborhood dataset is that it is in a Latitude/Longitude coordinate system, whereas all other data is in UTM83-10. To be able to clip one with the other requires both datasets to be in the same coordinate space.

So, place a CsmmapReprojector transformer after the Tester, connected to the Tester:Passed port. Set it up to reproject to UTM83-10

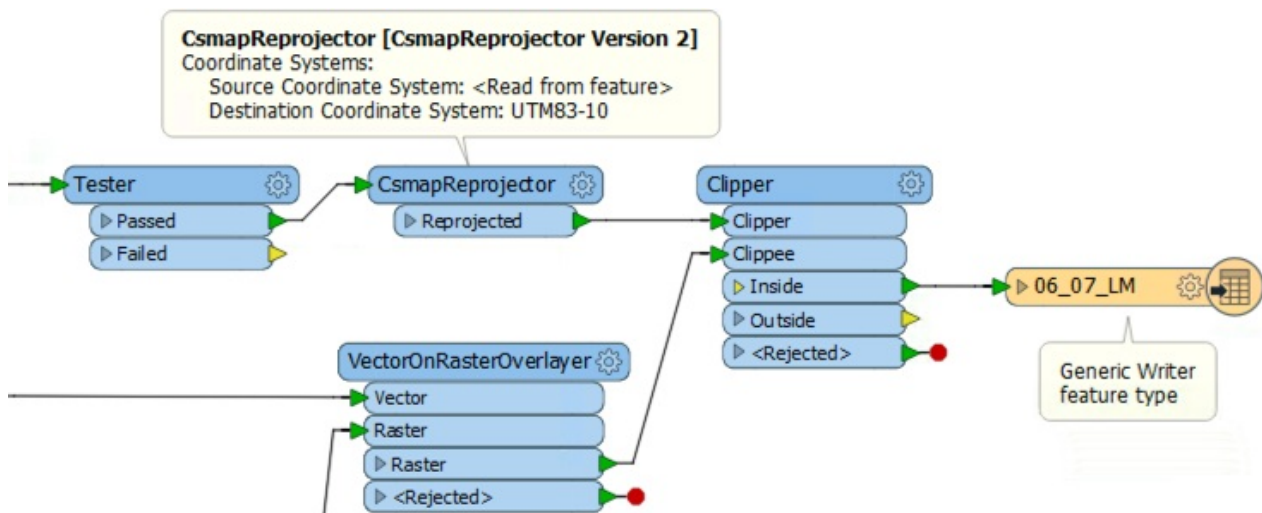


Professor Spatial F.M.E., E.T.L. says...

Why does the CmapReprojector come after the Tester? Because it has less work to do. If the data was reprojected first then we would be reprojecting data that is subsequently filtered out. It might only be a small difference here, but this is the type of detail that really helps workspace performance in larger projects.

6) Add Clipper

Now to clip the raster data. Add a Clipper transformer to the workspace. Connect the CmapReprojector to the Clipper:Clipper port. Connect the output from the VectorOnRasterOverlayer to the Clipper:Clippee port:



Check the parameters. The only parameter to really check is one specifically related to raster data: **Preserve Clippee Extents**. Set this parameter to **No** if it is not already.

7) Publish to FME Server

Save the workspace and publish it to FME Server. Register it with the Data Download service, being sure to click the **Edit** button to edit the service properties. In that dialog set the writer to **"Output [GENERIC]"** (not **"Output [JPEG]"**).

Run the workspace on FME Server. You should now be able to choose all source tiles and clip them to a chosen neighborhood, like so (here, the Downtown neighborhood):



CONGRATULATIONS

By completing this exercise you have learned how to:

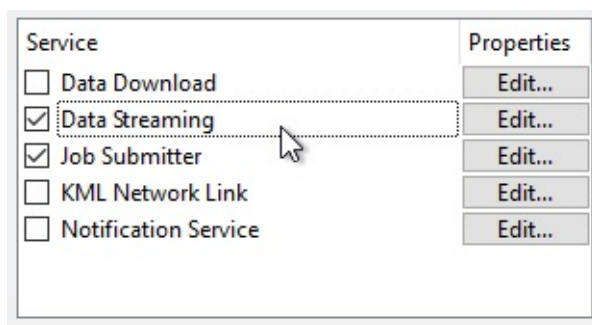
- *Set up a workspace for a user to select a specific area feature*
- *Clip data to a chosen area for use in a Data Download system*

Other Self-Serve Services

Although most people look at self-serve mostly in the context of Data Download, that is not the only Self-Serve Service available.

Data Streaming

Data Streaming is another service that a workspace can be registered against:



Whereas the Job Submitter service writes data, and the Data Download service returns a link to the data, a Data Streaming service returns a file of the data itself, streamed back to the client.

For example, if the Data Streaming URL for a workspace is posted into a web browser, the data will be automatically downloaded and opened in whatever application the browser associates with that file type (some data might open directly in the web browser itself).

Alternatively, the URL can be used directly as the source for a client application, like a GIS tool. When the client actively downloads the contents on a regular basis – as a GeoRSS reader would – then you have a feed, which is significantly different to a regular data download service.

Professor Spatial F.M.E., E.T.L. says...

Data Streaming is a slight misnomer in that a data streaming service does not supply a continuous stream of data; it merely provides a snapshot of the data at a particular point in time.

What Formats can be Streamed?

You can use any workspace with the data streaming service, provided it writes data in a format that is file-based or folder-based (i.e. not to a database or web URL).

If an output dataset is comprised of more than one file, the data streaming service automatically creates a compressed (zip) folder out of the data. For example, AutoCAD DWG format could be streamed, whereas ESRI Shape would be returned in a zipped file.

The most popular formats to stream are those that have a suitable client to read the feed. Some of the main formats that are output using the data streaming services include:

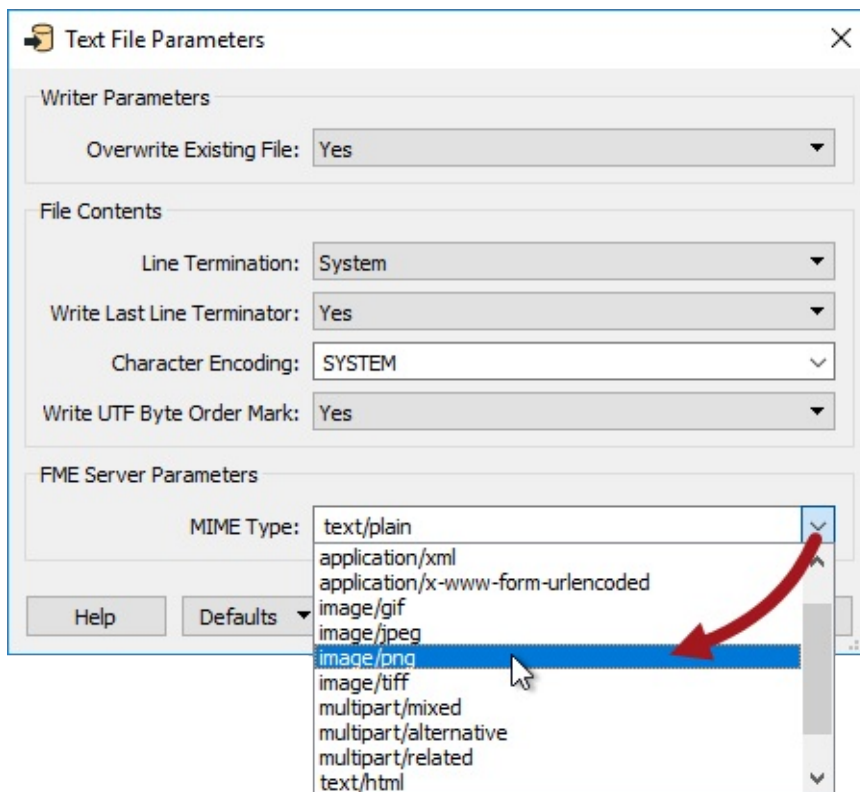
- RSS
- GeoRSS
- GeoJSON
- KML
- HTML
- JSON

MIME Types in Workspaces

A MIME header is a component of a file or e-mail message that is capable of indicating the content type of the file; for example, ***Content-Type: text/plain*** indicates a simple text file.

The application chosen to open a streamed file will depend on the MIME type and file association on the client's system.

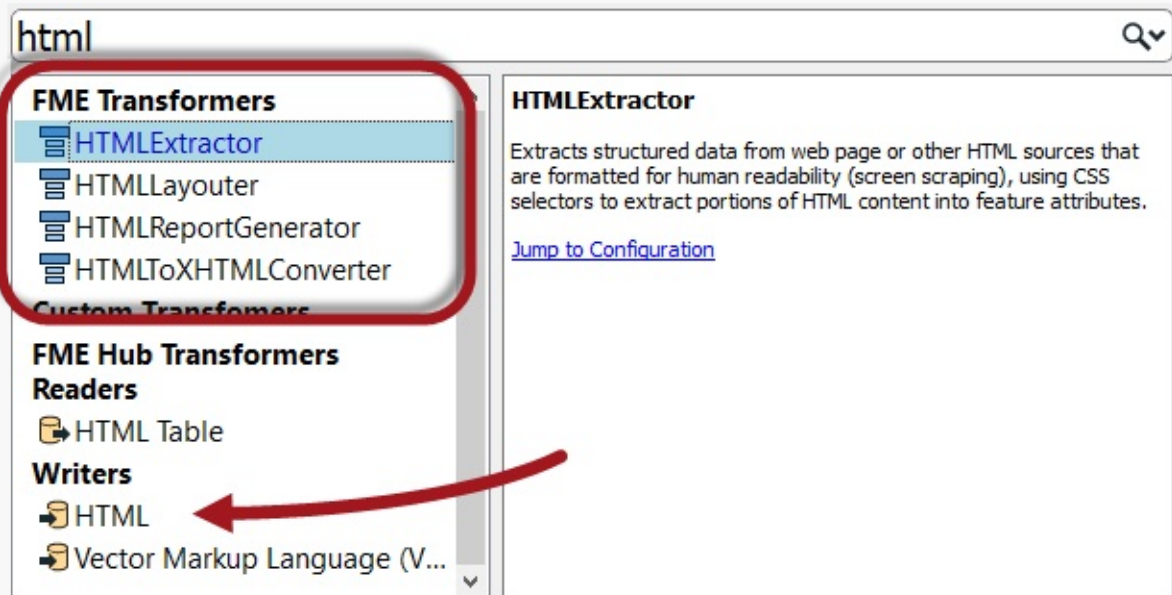
Setting MIME type is most important for FME writers where the content is not specifically defined by the writer. For example, the HTML writer has no MIME type setting because it is obviously providing text/html. The TextFile writer has a MIME type setting because the nature of its contents are ambiguous; it might be writing plain text (text/plain) or XML (text/xml) or it might even write the contents of a blob attribute containing a raster png image (image/png):



Here the author is saying that the content of the text file is valid PNG and should be opened in the default PNG application (possibly a web browser, possibly a graphic editor).

NEW

FME2017 introduces a new writer specifically for HTML format. No longer do you need to write HTML with a TextFile writer and set the MIME type.



Related transformers - the HTMLExtractor, HTMLLayouter, and HTMLReportGenerator - help to generate and format HTML content so that manual manipulation of such data is no longer required.

Also notice the HTML Table reader; not so useful for a Data Streaming setup, but great for scraping source data from web sites!

KML Network Link

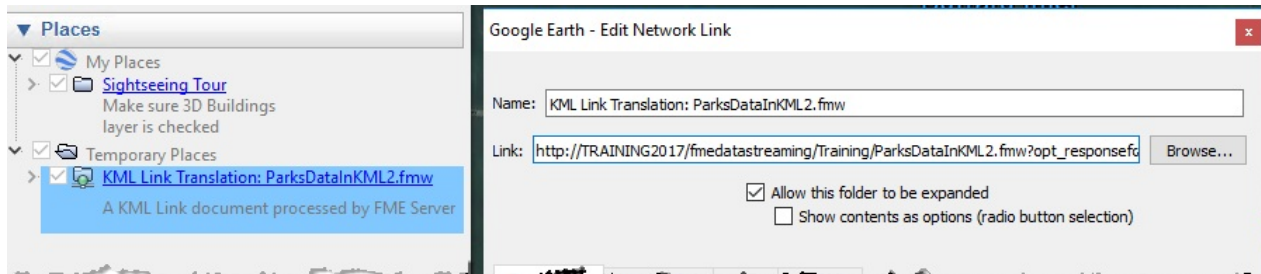
Google [describes a KML Network Link](#) as a type of *bookmark*; a link from Google Earth (or any other KML-reading application) to the true set of data.

In FME you can register any workspace that writes KML as a KML Network Link service, however you must also register it under Data Streaming as well:



Running the workspace using this service merely returns a small KMZ (compressed KML) file that contains this "bookmark". The bookmark is a link back to the workspace whose URL points to the Data Streaming service.

When Google Earth opens a KML Network Link service from FME Server, it receives the link to the workspace:



When the link is followed it triggers FME Server into running the workspace and returning the results as a stream of KML data.

Because Google Earth permits a refresh rate for network links, the translation can be re-run at a user-defined interval. This way the results are always as up-to-date as the chosen interval.

Of course, in this scenario the output is never written to a permanent dataset; the resulting data is simply streamed to the browser, which writes it to a cache.

| Exercise 2 Data Streaming System | |
|----------------------------------|---|
| Data | Orthophoto images (GeoTIFF) |
| Overall Goal | Create an FME Server Data Streaming system for orthophotos |
| Demonstrates | Data streaming |
| Start Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe2-Ex2-Begin.fmw |
| End Workspace | C:\FMEData2017\Workspaces\ServerAuthoring\SelfServe2-Ex2-Complete.fmw |

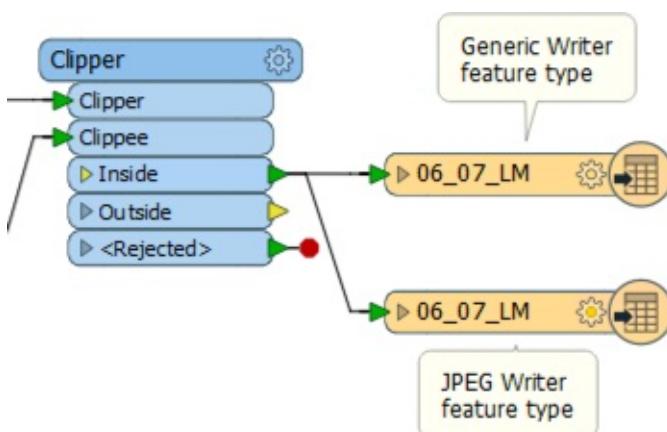
As a technical analyst in the GIS department of a city you have just created a system to allow other departments to download orthophoto data, rather than having to ask you to create it for them.

Sometimes the end-users download data as JPEG just to open it in a browser or image viewer to inspect it. You realize that, in cases like this, they may be able to use a data streaming service, instead of a data download.

1) Open Workspace

Open the workspace from exercise 1, or the begin workspace listed above. You can see that it consists of various readers, writers, transformers, and other functionality. Most importantly there is an unconnected JPEG Writer that could be used to stream data.

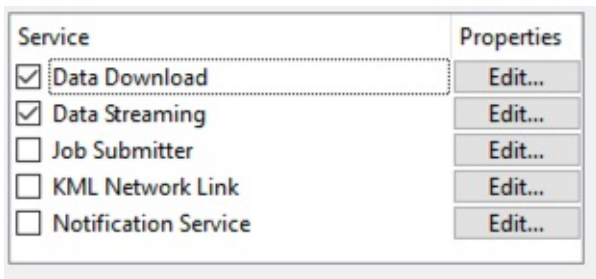
Currently the JPEG Writer is disconnected, so connect it back into the workspace, this time to the Clipper:Inside output port:



2) Publish to FME Server

Now publish the workspace to FME Server.

In the final dialog of the publishing wizard, check the boxes to register the workspace with both Data Download and Data Streaming (but don't click Finish yet):



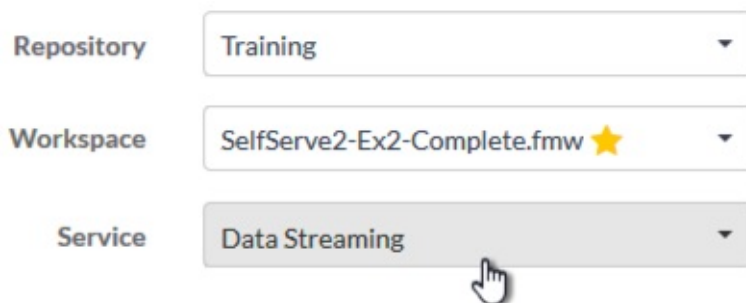
The screenshot shows a dialog box with two columns: 'Service' and 'Properties'. In the 'Service' column, there are five items: 'Data Download' (checked), 'Data Streaming' (checked), 'Job Submitter' (unchecked), 'KML Network Link' (unchecked), and 'Notification Service' (unchecked). In the 'Properties' column, there are five 'Edit...' buttons corresponding to each service. The 'Data Download' service is highlighted with a dashed border.

Now click the Edit button for the Data Download service. Ensure that service is using the output of the Generic Writer.

Next click the Edit button for the Data Streaming service. Ensure that service is using the output of the JPEG Writer (for now we're limiting the streaming of data to JPEG format).

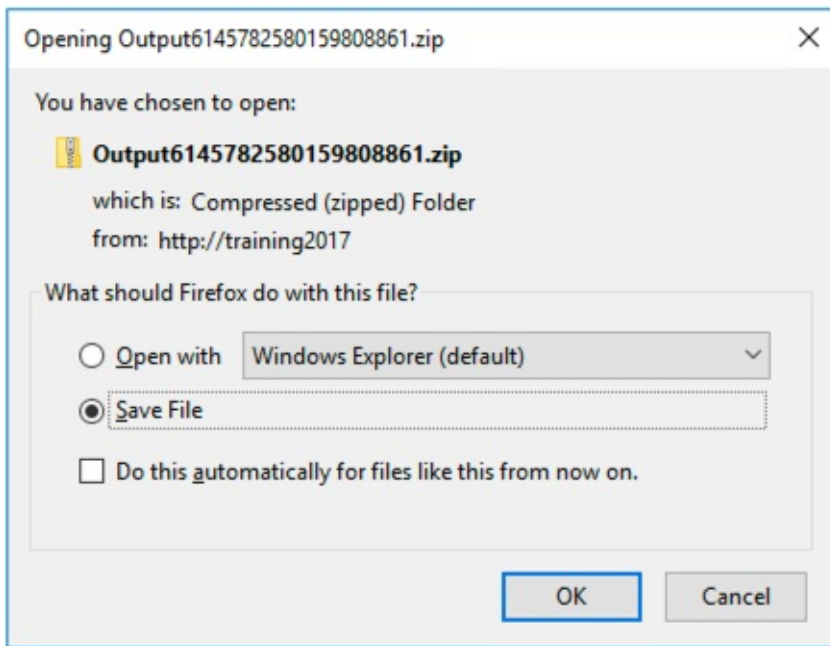
3) Run Workspace

In the FME Server web interface locate the newly published workspace and run it. In the parameters for the workspace be sure to set the web service to Data Streaming instead of Data Download:



The screenshot shows the FME Server web interface with three dropdown menus. The first dropdown is labeled 'Repository' and has 'Training' selected. The second dropdown is labeled 'Workspace' and has 'SelfServe2-Ex2-Complete.fmw' selected, with a yellow star icon next to it. The third dropdown is labeled 'Service' and has 'Data Streaming' selected. A mouse cursor is pointing at the 'Data Streaming' option in the 'Service' dropdown.

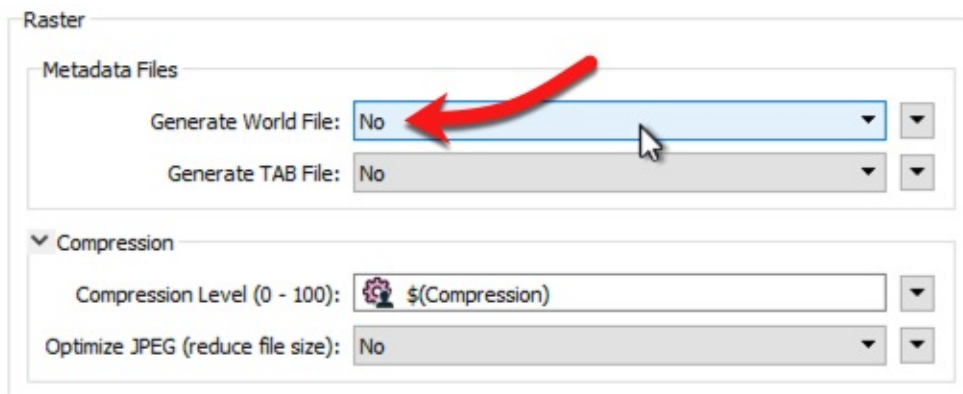
The result of this translation is not a streamed JPEG file. Instead, the translation returns a zip file:



If you open the zip file you'll see that it includes both a JPEG file and a wld (World) file. That's why FME returned a zip file. It will zip the results of a Data Streaming service whenever the result is multiple files.

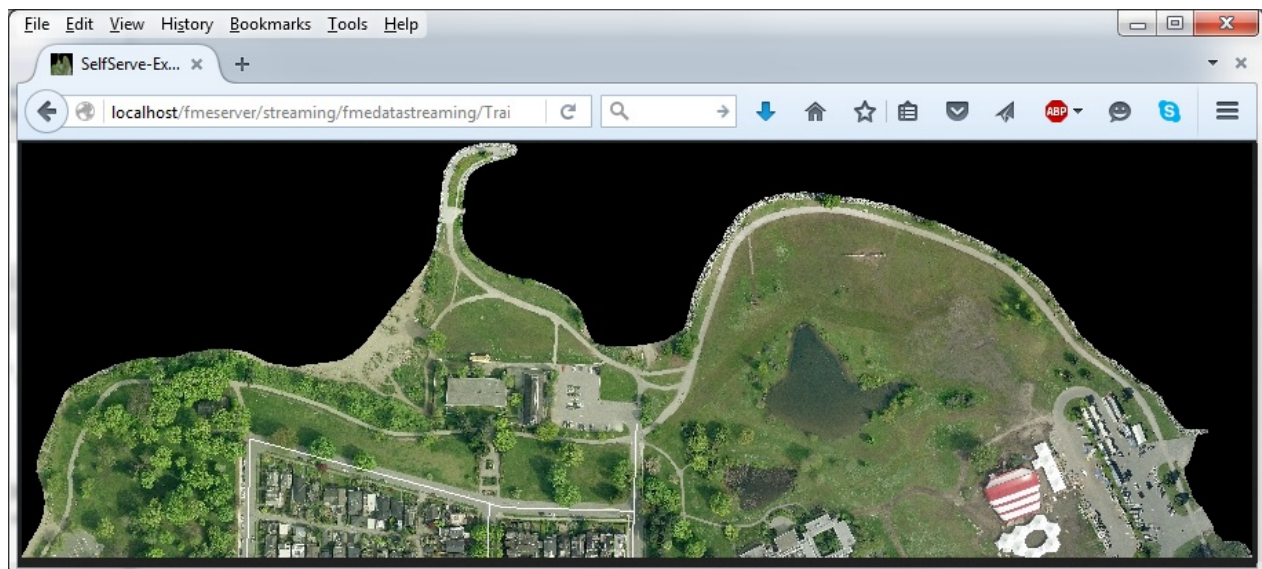
4) Turn off World File Creation

To really stream the data we should turn off the world file creation in the workspace. Check the properties for the JPEG Writer's feature type and set the Generate World File parameter to No:



5) Publish and Run Workspace

Re-publish the workspace and run it on FME Server. You should find that the results of the translation are returned as a streamed JPEG file. Most likely it will open directly in your web browser:



CONGRATULATIONS

By completing this exercise you have learned how to:

- *Set up a workspace for use in a Data Streaming service*
- *Publish a workspace to the Data Streaming service*

Troubleshooting for Administrators

This section shows a few basic troubleshooting techniques in case of emergency.

Data Streaming: Zipped Output

If the Data Streaming service does not provide a dataset, but a zip file, then the following may be of help.

- If the output format is one composed of a number of files – for example Esri Shape – then the Data Streaming service will create a zip of the output.
- Often raster Writers will write an additional World file, and vector Writers a prj file, in order to preserve coordinate system information, and this can unexpectedly cause this issue. In most cases wld/prj file creation can be turned off with either Writer or Feature Type parameters.

Module Review

This module introduced you to FME Server's Self-Serve Services and related functionality in FME Workbench.

What You Should Have Learned from this Module

The following are key points to be learned from this module.

Theory

- Selection of source data by area can be achieved using Bounding Box parameters, or a transformer such as the Clipper or FeatureReader.
- Other FME Services allow data to be streamed live into a suitable application.

FME Skills

- The ability to give the end-user control over area of interest
- The ability to use Data Streaming and KML Network Link services

Further Reading

For further reading why not check out [this guide to sharing open data](#) using FME Server, or [this fine blog article](#) that explores how to implement an open-data portal using FME, Amazon S3, Leaflet, JQuery, and a whole lot more!

Questions

Here are the answers to the questions in this chapter.

Miss Vector says...

Why is the FeatureReader the preferred option [for filtering data in a self-serve system]? Pick all the reasons that apply:

- 1. It can be quicker and more resource efficient**
- 2. It allows multiple areas to be used as the existing areas*
- 3. It works with raster data*
- 4. It has more choices for spatial filtering**

The FeatureReader is more efficient where the source dataset is a format that has a spatial index, because it won't have to read all of the source data. It also has more filtering choices than a Clipper transformer (though not a SpatialFilter)

The FeatureReader does allow multiple areas to be used, and it does support raster data - however so does the Clipper transformer, so there is no difference between the two in those scenarios. Additionally, the two are equivalent in performance where there is no spatial index (i.e. both will need to read the full dataset)

Miss Vector says...

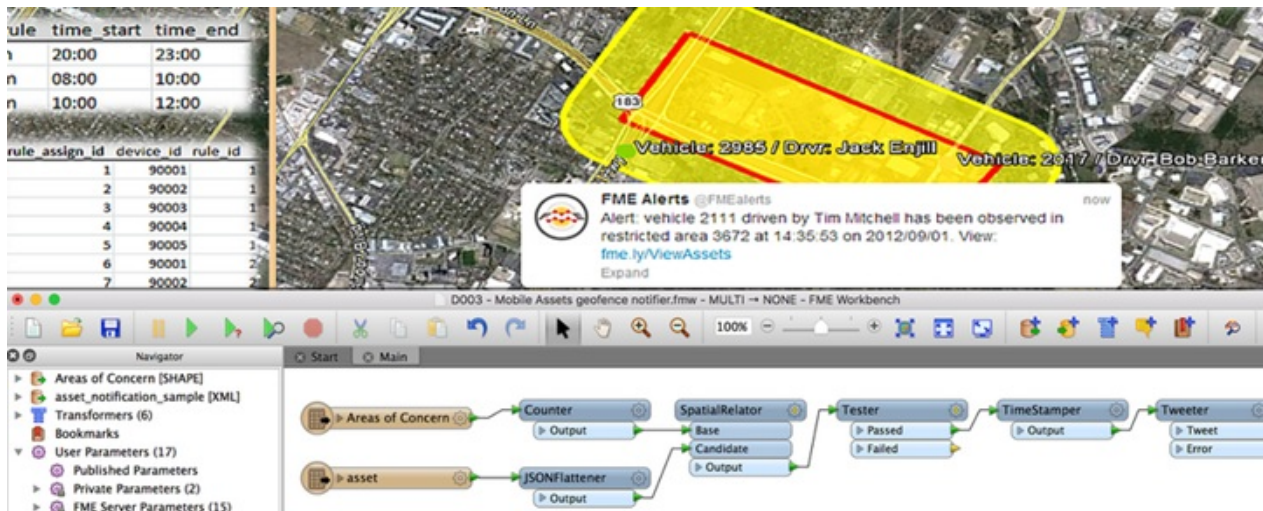
If the incoming geometry string is XML, there is a shortcut [to passing/using it in a workspace]. What is it?

- 1. Use an XML Reader with the source dataset parameter published*
- 2. Publish the source dataset parameter in the FeatureReader to send the XML straight into it*
- 3. Replace the GeometryReplacer with an XMLTemplater*
- 4. Use a Creator transformer with the geometry parameter published**

If you publish the geometry parameter in a Creator transformer you can pass XML directly into it and FME will automatically generate a spatial feature from it. This removes the need for any AttributeManager and GeometryReplacer transformers.

Real-Time with FME Server

Real-time systems are those that act on events as they happen, and send information as it becomes available.



Real-time is often implemented to monitor sensor networks, run event-driven processing, and/or to send alerts to mobile devices such as cell phones. Therefore real-time is usually related to small amounts of information (rather than large datasets) that arrive either as individual events or as a continuous communication stream.

Real-world examples include processing simple location data from a vehicle tracking system, or handling a continuous stream of data from a temperature sensor or lightning detector. A simpler example would be sending an email to a system administrator in response to database table updates.

Real-time data in FME Server is handled in two ways: Notifications and Message Streams.

Notifications

Notifications are how FME Server handles individual messages and alerts.

FME Server has a specific Notification Service that can be set up to listen for an incoming message from outside of FME, and trigger a certain action in response; or that can be set up to send a single alert in response to an event that takes place on FME Server.

Protocols supported include email, websockets, and notifications for Apple and Android devices.

Message Streaming

Whereas Notifications are one-off messages, Message Streaming involves a continuous flow of information

Instead of a workspace being run for each message, the workspace is constantly running and doesn't need to be started each time. Because of this reduced overhead it can process data at a much faster rate than the notification service.

The Notification Service

The Notification Service is a way for data to be pushed to and from FME Server in the form of short messages.

What is a Notification?

A notification is a simple message (sometimes called an “alert”) that informs someone or something that a particular event has happened.

Notifications on FME Server can occur in two different forms: **incoming** and **outgoing**.

Incoming notifications alert FME Server to an event that has taken place elsewhere.

Outgoing notifications alert users to an event that has taken place on FME Server.



In this way, FME Server can take action in response to an event notification, or a user can take action in response to a notification from FME Server.

The **Notification Service** is the part of the FME Server architecture that handles incoming and outgoing notifications.

When to Use Notifications

Notifications should be used when you want to trigger an FME Server response to an event that is reported from outside of FME. The event should not be a continuous series of messages; if there is more than one message per second you should consider using Message Streaming techniques instead.

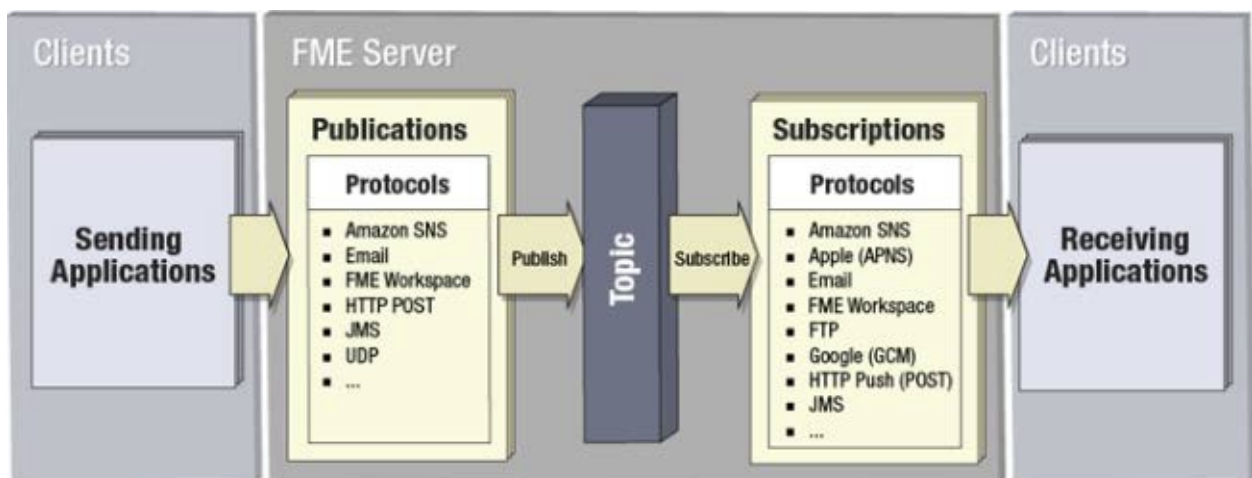
Notifications are also for when you want to send a message about something that happened on FME Server, to an outside subscriber. Again, if there is likely to be more than one message per second you should consider using Message Streaming.

In either case a notification is for sending a brief message, usually in order to trigger an action from the recipient. It's not intended to be used for transmitting large amounts of spatial data. At most you should use it for a single geographic feature such as a point location.

Elements of the Notification Service

The Notification Service includes a number of different components.

- **Clients:** An external user or system that sends or receives a notification
- **Publications:** An event handler that listens for incoming notifications
- **Subscriptions:** An event handler that dispatches outgoing notifications
- **Topics:** A subject that describes what the notification is about
- **Protocols:** Methods by which FME Server can receive or send notifications



Ms Analyst says...

As well as Publications/Subscriptions you'll also hear the terms Publishers/Subscribers. These are alternative names for clients. Publishers are clients that publish to FME Server Publications. Subscribers are clients that subscribe to FME Server Subscriptions. Try not to get the terms muddled!

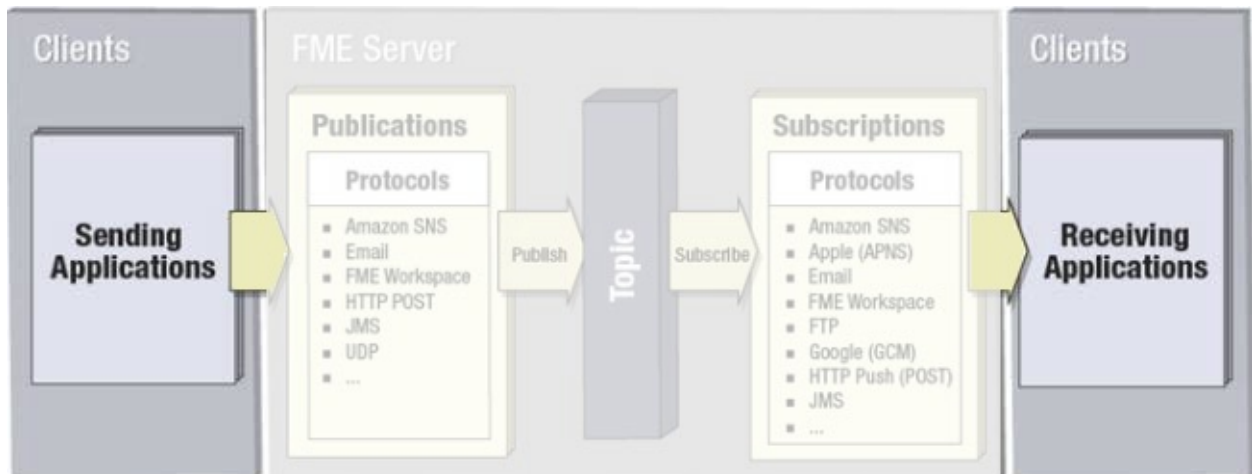
Miss Vector says...

All notification setups must have which of these:

- 1. Incoming components (Publisher, Publication) AND outgoing components (Subscription, Subscriber)*
- 2. Incoming components OR outgoing components OR both*
- 3. Incoming components OR outgoing components but never both*
- 4. None of the above*

Clients

A client is a user or system that sends or receives a notification. The client may be a physical person, or may just be a component in a computer system. Either way, a client is not a core part of FME Server, rather someone or something that interacts with it.



For example, a database update might cause a trigger to send a notification to FME Server, in which case the database system is the client. But a client could also be a person who, for example, triggers a notification by sending an email to FME Server.

Likewise, FME Server can send a notification for another client system to receive.

Alternatively this client can also be a real person, who might receive a notification in the form of an email.

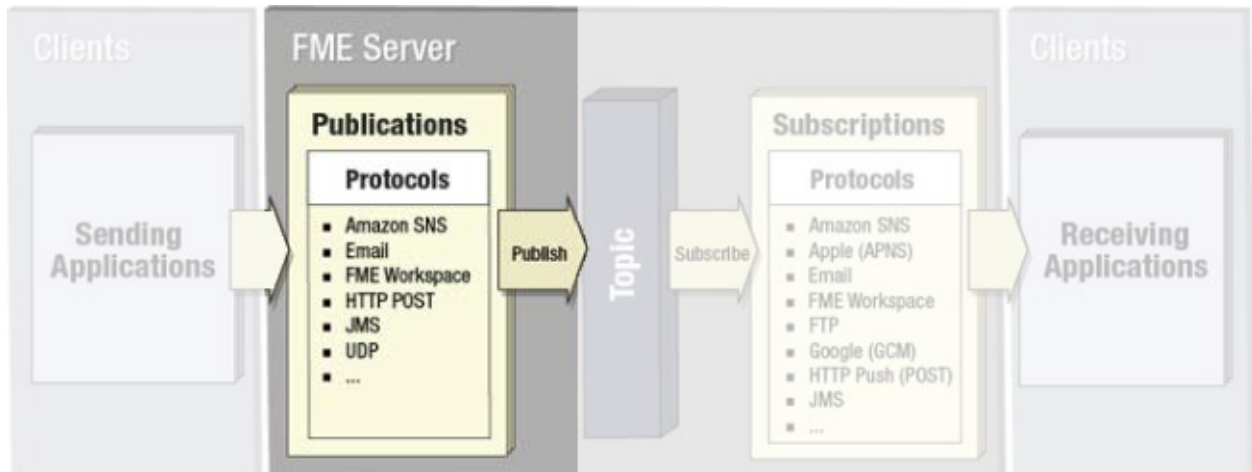
Publishers vs Subscribers

Clients that send a notification to FME Server are called **Publishers**.

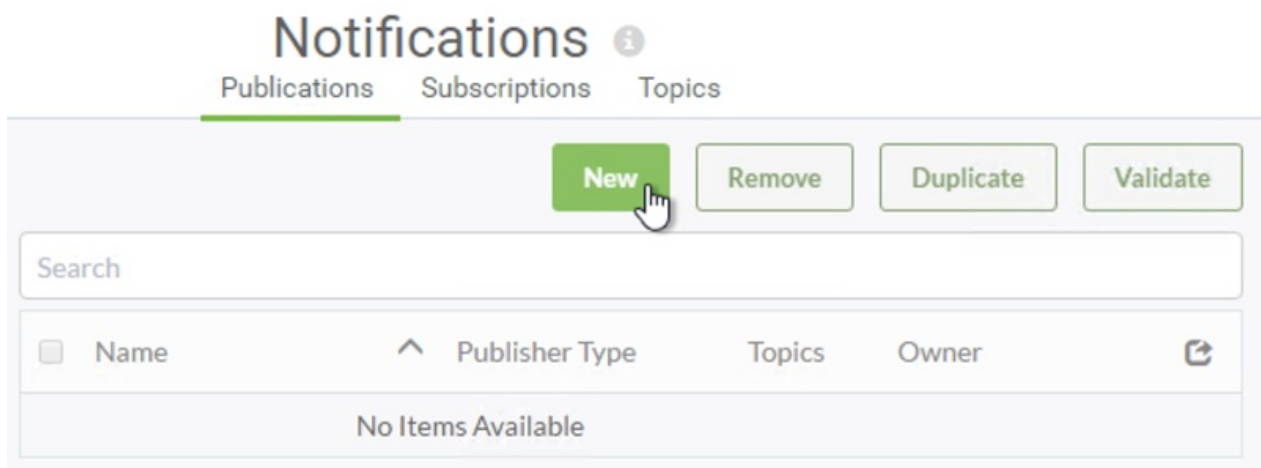
Clients that receive a notification from FME Server are called **Subscribers**.

Publications

A Publication is an FME Server component that receives incoming notifications from a client (publisher).

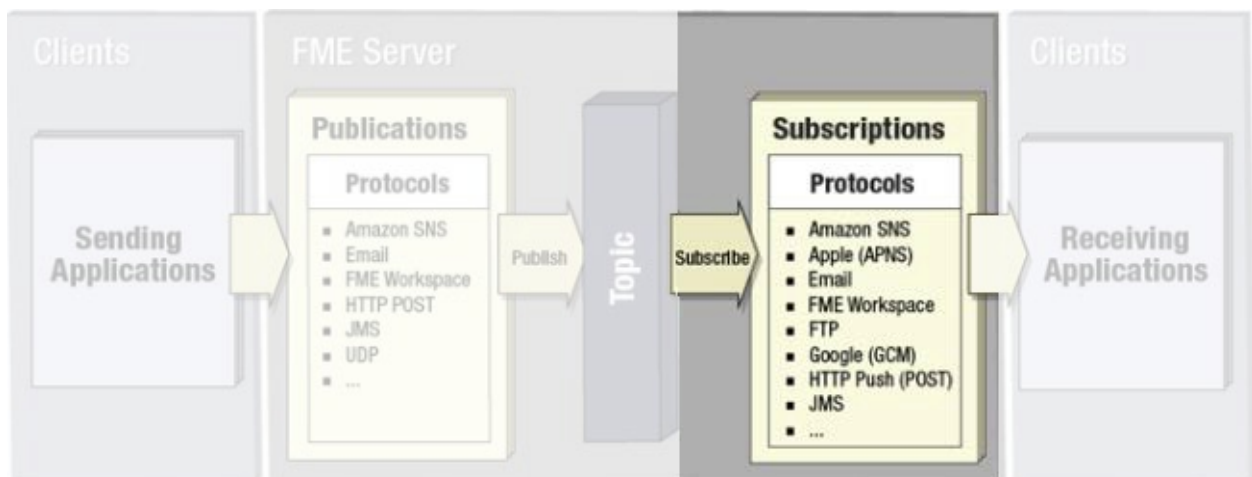


To receive a notification in FME Server a workspace author (or administrator) must create a new Publication. A Publication is created in the FME Server web interface on the Notifications page:

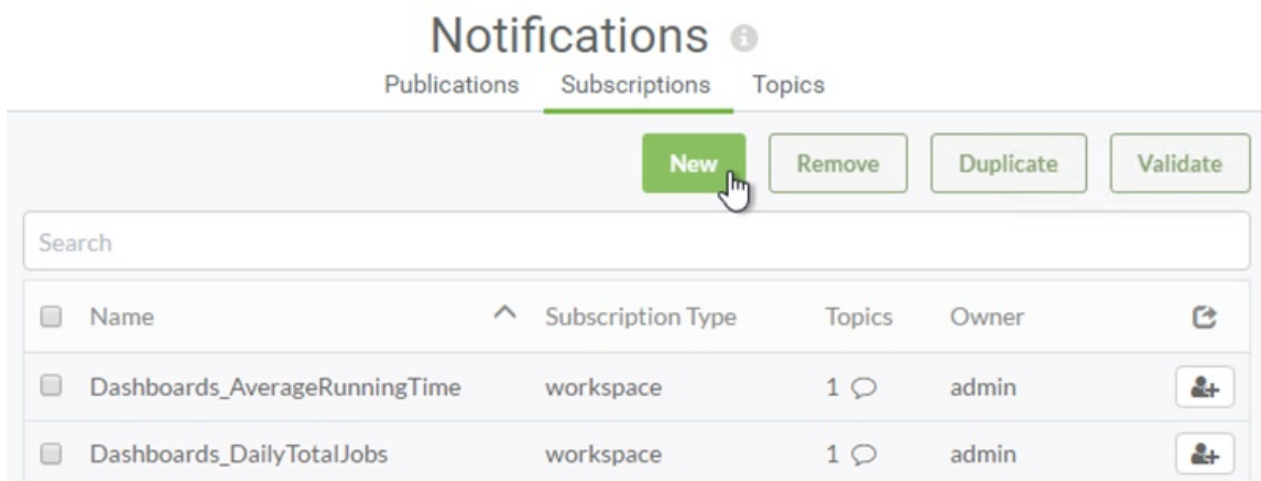


Subscriptions

A Subscription is an FME Server component that sends outgoing notifications to a client (subscriber).



To send a notification in FME Server a workspace author (or administrator) must create a new Subscription. A Subscription is created in the FME Server web interface on the Notifications page:



FME automatically creates some Subscriptions on installation, e.g. to alert clients to the success or failure of jobs that have been run.

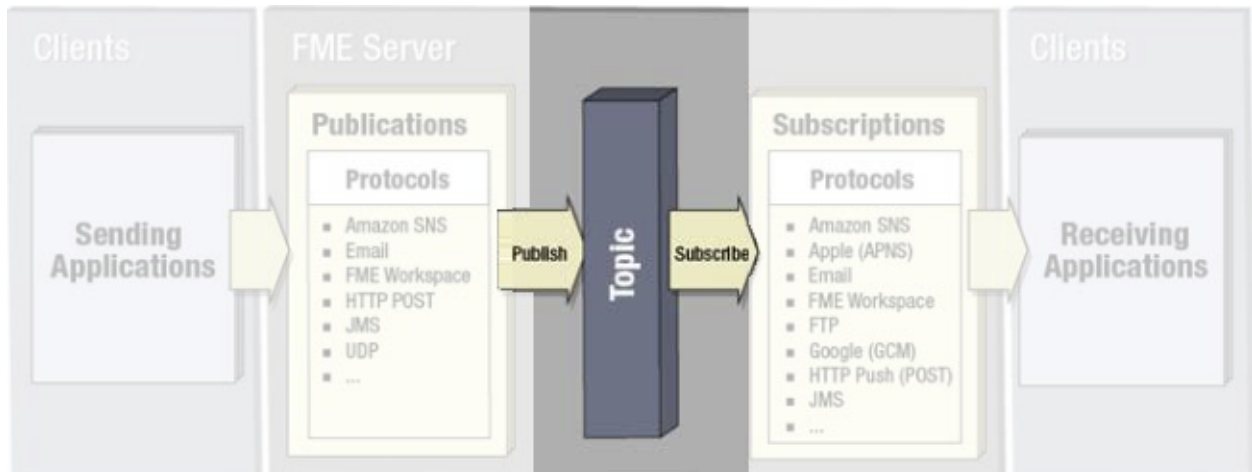
IMPORTANT

Although the actions may seem odd in regard to the name – Publications receive messages, Subscriptions send them – it is correct. The name refers to the Client, not the Server.

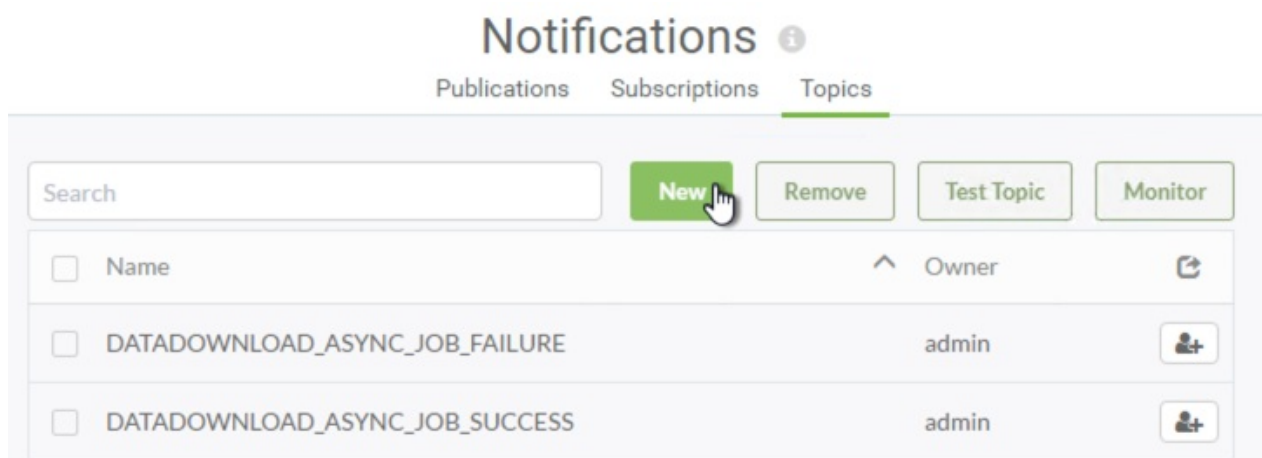
*So FME Server receives a Publication because the client is **publishing** it. Likewise, FME Server sends a Subscription because the client is **subscribing** to it.*

Topics

A Topic is a component that acts as a mediator for messages and defines the message content. Think of it as a mix of subject line for notifications and a trigger for them to occur.



Like Publications, a Topic is created in the FME Server web interface, on the Notifications page:



FME automatically creates some topics on installation, to trigger the subscriptions that it creates.

Publications and Topics

All Publications are linked to a Topic so that when an incoming message is received by a Publication, it is categorized and the related Topic is triggered.

A Publication can be linked to multiple Topics, so each incoming message can trigger multiple actions to occur. Additionally, multiple Publications can trigger the same topic.

For example, a lightning strike sensor might publish to the topics WeatherEvent and AircraftAlerts, whereas a flood sensor might publish to the topics WeatherEvent and RoadConditions.

Daily Interop Reporter, Chad Pugh-Litzer says...

When I write a news article I publish the article to the Daily Interop web site, tagged with a number of topics to describe it. For example, a report about a soccer team's tax return would be filed under both 'Financial' and 'Sports' because it relates to both.

Subscriptions and Topics

All Subscriptions are also linked to a Topic. When a topic is triggered an outgoing message is sent through the Subscription to the Subscriber.

A subscription can be linked to multiple Topics, each topic being triggered causes an outgoing notification. Each Topic can also be subscribed to by multiple Subscriptions.

For example, a police headquarters might subscribe to the RoadConditions topic, to receive notifications on that subject. The local TV weather channel also subscribes to the RoadConditions topics, but in addition subscribes to WeatherEvent to hear about those particular events.

InteropGeek68 says...

I subscribe to articles published on the Daily Interop web site according to their topic. For example, I subscribe to reports with the topic financial, whereas my friend – InteropJock72 – subscribes to sports articles. Because Chad's article has both tags, we'd both receive it.

Miss Vector says...

Please don't get this wrong! Publications and Topics have what relationship?

- 1. One:One (Each Publication has one Topic, each Topic belongs to one Publication)*
- 2. One:Many (Each Publication can have many Topics, each Topic belongs to one Publication)*
- 3. Many:One (Each Publication has one Topic, each Topic can belong to multiple Publications)*
- 4. Many:Many (Each Publication can have many Topics, each Topic can belong to multiple Publications)*

Protocols

A protocol is a system of data exchange between FME Server and a client.

We already know that FME sends and receives notifications. Protocols are the method by which these notifications are sent and received. Each Publication and Subscription is defined using a particular communication protocol.

To trigger an incoming notification by email – for example – you would create an FME Publication using the Email protocol. To send a notification to subscribers with an Apple mobile device, you would create an FME Subscription using the Apple Push protocol.

There are many different protocols available in FME Server; some of them are only for use on Publications, some are only for Subscriptions, and some of them can be used with both notification types.

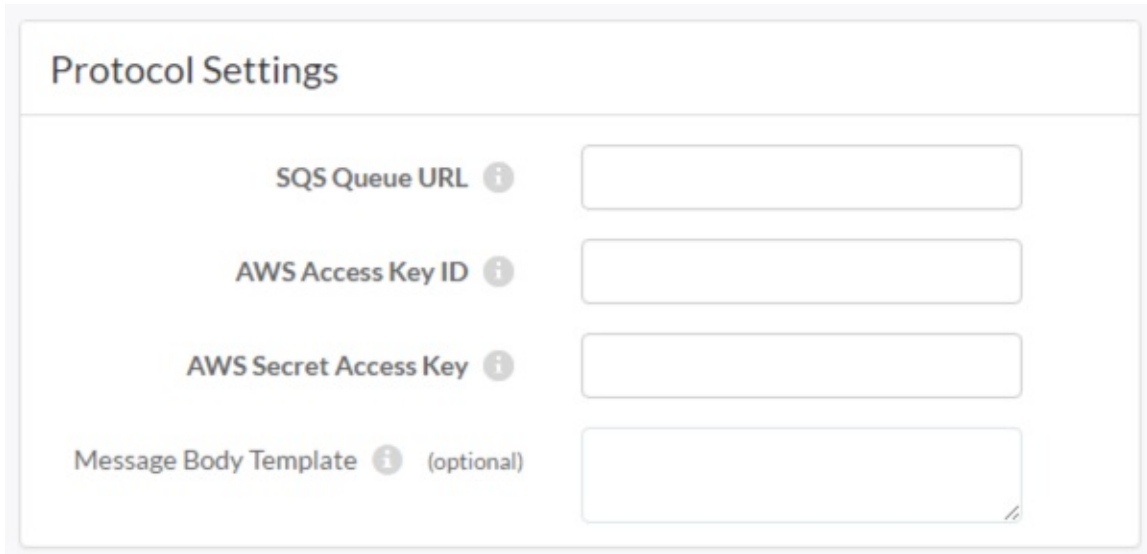
This table lists the different Publication and Subscription protocols and the following pages go into detail on some of the most important.

| Protocol | Description | Publications | Subscriptions |
|-------------------------|--|--------------|---------------|
| Amazon S3 | Communication of a notification (file) to Amazon's Simple Storage Service | | Y |
| Amazon S3 Watch | Monitoring an AWS S3 bucket for activity | Y | |
| Amazon SNS | Communication with Amazon's Simple Notification Service | Y | Y |
| Amazon SQS | Communication with Amazon's Simple Queue Service | Y | Y |
| Apple Push Notification | Communication with Apple mobile devices | | Y |
| Directory Watch | Monitoring a folder for new files as a trigger mechanism | Y | |
| Dropbox | Communication of a notification (file) to the Dropbox web service | | Y |
| Dropbox Watch | Like Directory Watch, but monitoring folders stored in the Dropbox web service | Y | |
| Email (IMAP) | Communication via an email service | Y | |
| Email (SMTP) | Communication via an email server | Y | Y |
| FME Workspace | Communication of notifications to an FME workspace | | Y |
| FTP/SFTP/FTP Watch | Communication with an FTP site | Y | Y |
| Google Cloud Messaging | Communication with Android mobile devices | | Y |
| JMS | Communication with a Java Message Service | Y | Y |
| Logger | Output to a simple log file | | Y |
| Push | Communication via HTTP requests | | Y |
| UDP | Communication via a User Datagram Protocol port | Y | |
| WebSocket | Communication via a WebSocket channel | Y | Y |

Protocols are pre-defined components in the FME Server architecture and do not need to be defined in the web interface.

However, a number of fields are made available to configure them when a Publication or Subscription chooses to make use of that protocol.

For example, here are the parameters for an Amazon SQS Subscription:



Protocol Settings

SQS Queue URL ⓘ

AWS Access Key ID ⓘ

AWS Secret Access Key ⓘ

Message Body Template ⓘ (optional)

These parameters must be set when the Subscription is created as they are needed in order to be able to send out a notification using the protocol.

Miss Vector says...

Tell me, which one of these statements is correct:

- 1. SMTP and IMAP can both be used as either a Subscription and/or a Publication protocol*
- 2. SMTP can be used as both a Subscription and a Publication; IMAP can only be used for a Publication*
- 3. SMTP can only be used for a Publication; IMAP can only be used as both a Subscription and a Publication*
- 4. SMTP can only be used for a Subscription; IMAP can only be used for a Publication*

Available Protocols

Email and FME Workspace protocols are among the most popular for FME Server, and they are covered in detail in the next few sections. Some other commonly used protocols are:

Watch Notifications

There are several watch notification protocols: Directory Watch and FTP Watch are the two most common.

These are Publication-only protocols that monitor folders and FTP sites for new files to be added. When a file appears in that directory then the Topic to which it is tied becomes triggered. The name of the affected file is passed on through the topic message.

A common use is for monitoring a folder in which datasets are stored. When a new dataset is placed into the folder then a workspace automatically runs to process it.

This protocol allows files to be filtered by the action type, so that it is not just the creation of a new file that can be monitored. It's also possible to monitor for file deletions and modifications.

Mobile Notifications

Mobile Notifications are Subscription-only protocols that send notifications to mobile devices. The two protocols available are **Apple Push Notification** and **Google Cloud Messaging**.

Apple Device Notifications

The Apple Push Subscriber allows delivery of messages to an Apple iOS device, like an iPhone. Messages are sent from FME Server to the device via a cloud service called the Apple Push Notification service (APNs).

The FME Server Reference Manual explains how to set up such a notification and what is required in terms of Apple “SSL Keystore” and device tokens.

Android Device Notifications

Google Cloud Messaging is the Android equivalent to Apple notifications. Again, they involve sending content to a mobile device from an FME Server Publication.

Amazon Notifications

Several Amazon protocols are supported by FME Server notifications. SNS and SQS are two types of notification systems. FME Server is capable of pushing messages to both of them and receiving messages from both of them. The difference (if you are interested) is that SNS pushes messages to subscribers immediately, whereas SQS stores the messages in a queue until the subscriber fetches them.

Amazon S3 is an online file storage service. It can be either a Publisher or Subscriber in FME Server; i.e. FME can be notified to read from S3 "buckets" and can issue a notification in the form of writing to an S3 bucket. The S3 Publication is actually a "Watch" type notification. FME Server can watch for files arriving there and trigger a notification in response.

WebSocket Notifications

A WebSocket is a TCP-based line of communication. FME Server can accept and send notifications from/to another WebSocket client. WebSocket protocols are supported by most web browsers, meaning any web application can communicate to FME Server and FME Server can send notifications (even small quantities of data) directly to a web browser.

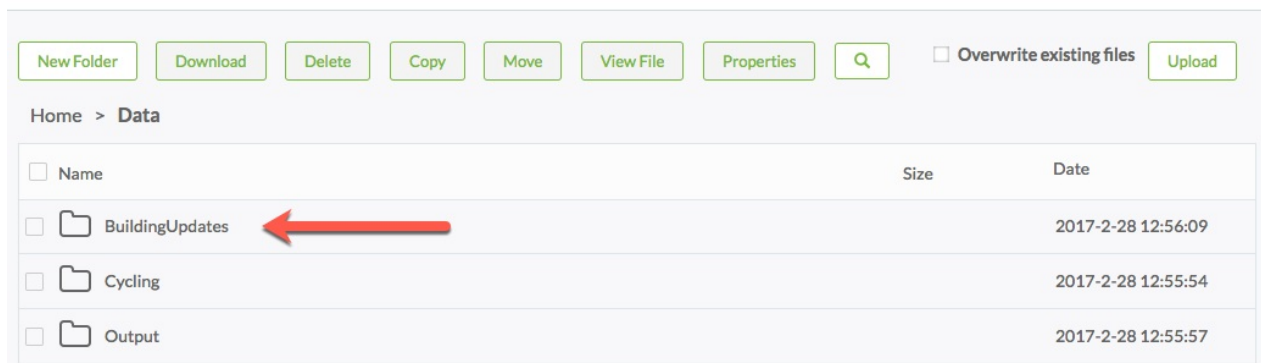
| Exercise 1 Building Updates Notification System | |
|---|--|
| Data | Building footprints (Esri Shapefile) |
| Overall Goal | Trigger notification for new files |
| Demonstrates | Notification topics and Directory Watch publications |
| Start Workspace | N/A |
| End Workspace | N/A |

As a technical analyst in the GIS department you want to start experimenting with notifications in FME Server. The Directory Watch protocol seems like a good place to start, and you already were thinking about a shared folder where users place Shapefile datasets for adding to, or updating, the corporate database.

1) Create Resources Folder

The first step is to create a Resources folder to copy the data to. Open the FME Server web interface and navigate to the Resources page.

Browse to the Data folder and create a new subfolder called BuildingUpdates:



Miss Vector says...

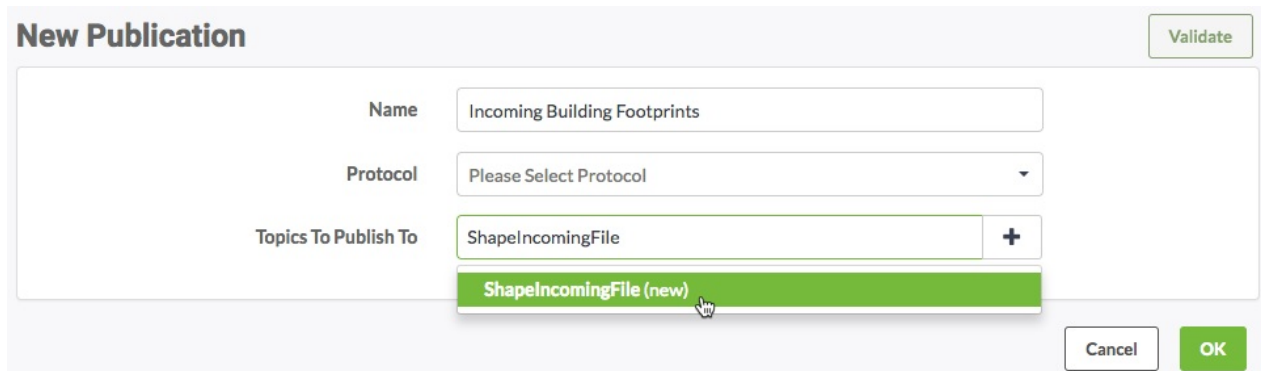
This exercise utilizes the FME Server Resource folders, but there is also native support in FME Server to watch for new resources in Amazon S3 Buckets, Dropbox, and FTP. Using the same concepts described here, you could use one of these protocols instead of Directory Watch.

2) Create Publication

Now to create a publication and topic that will be triggered by a new file. Navigate to the Notifications page, click the Publications tab, and then click the New button.

Enter "Incoming Building Footprints" as the new publication's name.

Next click in the text box besides Topics to Publish To. Type in ShapeIncomingFile and click on the entry with that name that appears in the drop-down list. This will create a new topic and assign it to this publication.

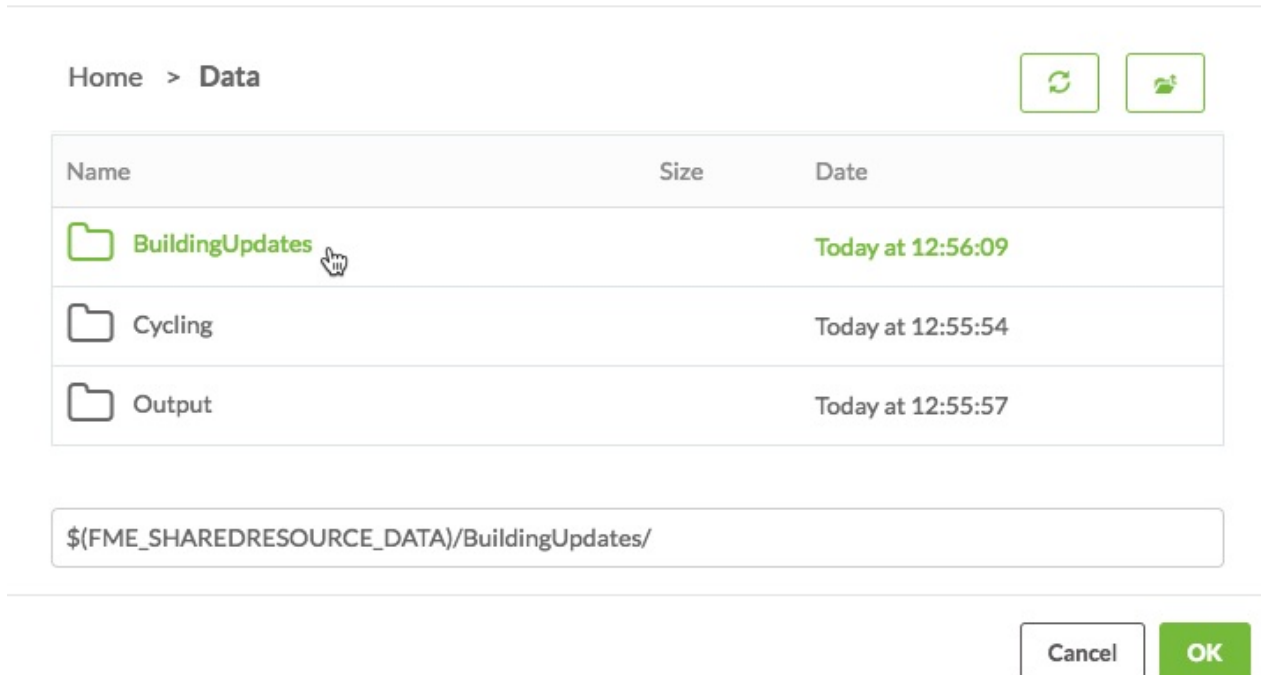


The "New Publication" dialog box is shown. It has a title bar "New Publication" and a "Validate" button in the top right. The form contains three fields: "Name" with the value "Incoming Building Footprints", "Protocol" with a dropdown menu showing "Please Select Protocol", and "Topics To Publish To" with a text input containing "ShapeIncomingFile" and a "+" button. Below the text input, a dropdown list is open, showing "ShapeIncomingFile (new)" highlighted with a mouse cursor. At the bottom right are "Cancel" and "OK" buttons.




3) Set Publication Protocol

Now select Directory Watch from the drop-down list as the protocol for this publication. In the dialog that appears click the browse button for the Directory to Watch parameter and select the newly created resources folder:

Select folder for *Directory to Watch*



The "Select folder for Directory to Watch" dialog box is shown. It has a breadcrumb "Home > Data" and two icons (refresh and search) in the top right. Below is a table with three columns: "Name", "Size", and "Date".

| Name | Size | Date |
|---|------|-------------------|
|  BuildingUpdates | | Today at 12:56:09 |
|  Cycling | | Today at 12:55:54 |
|  Output | | Today at 12:55:57 |

A mouse cursor is pointing at the "BuildingUpdates" folder. Below the table is a text input field containing the path: "\${FME_SHAREDRESOURCE_DATA}/BuildingUpdates/". At the bottom right are "Cancel" and "OK" buttons.

Back in the publication definition, for the Filter parameter remove the MODIFY and DELETE actions. All we really want to monitor are new files arriving, not old ones being removed:

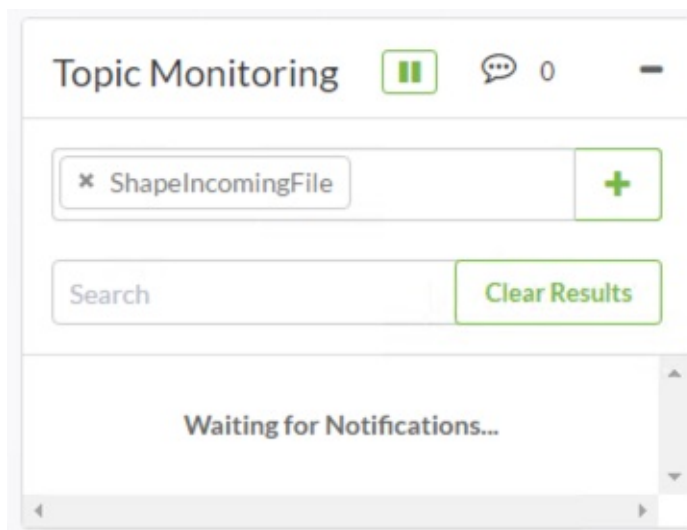


Change the Poll Interval to 1 Minute and click OK to create the new publication.

4) Monitor Topic

Click on the tab for Topics. The ShapeIncomingFile topic should now be listed in the list of topics. There is also a Topic Monitoring dialog to the upper-right of the page.

In the drop-down box for Topic Monitoring, select the ShapeIncomingFile topic and click the green "run" button. This will start Topic Monitoring:




5) Test Topic

Now let's test the topic. Locate the source Shapefile datasets in C:\FMEDData2017\Data\Engineering\BuildingFootprints. Select a set of files (.dbf, .prj, .shp, .shx) for one dataset and create a compressed zip file out of them (right-click > Send to > Compressed (zipped) folder).




Now upload the zip file into the newly created Resources folder. There are two ways to do this.

You can use the file system (by copying the file to C:\ProgramData\Safe Software\FME Server\resources\data\BuildingUpdates) or use the FME Server web interface. If you use the web interface, open a new window or tab, so we can continue to monitor the ShapeIncomingFile topic.

Home > Data > BuildingUpdates

| Upload(s) Completed | | |
|--------------------------|---|---------------------------------|
| <input type="checkbox"/> | Name | Size Date |
| <input type="checkbox"/> |  update001.zip | 9.25 KB 2017-2-28 13:14:12 |

Check back in the Topic Monitoring window and you will see that the topic has been triggered by the new file:

Topic Monitoring   1 

✕ ShapelIncomingFile

+

Search

Clear Results

ShapelIncomingFile Today at 13:15:57

{ "dirwatch_publisher_path":
"C:\\ProgramData\\Safe Software\\FME
Server\\resources\\data\\BuildingUpdates\\update
001.zip", "dirwatch_publisher_content":
"ENTRY_CREATE C:\\ProgramData\\Safe
Software\\FME
Server\\resources\\data\\BuildingUpdates\\update
001.zip", "dirwatch_publisher_action": "CREATE",
"ws_topic": "ShapelIncomingFile", "fns_type":
"dirwatch_publisher" }

Miss Vector says...

Remember, the Publication is set up to check the folder only once per minute - so if the Topic Monitoring doesn't immediately show a result, don't panic! Be patient and it will appear shortly.

Now we know how the Directory Watch notification works! We will see in subsequent exercises how to process this information.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Create a new Publication*
- *Create a new Topic as part of the Create Publication process*
- *Use Directory Watch to trigger Topics and Notifications*
- *Test a Publication and Topic using Topic Monitoring*

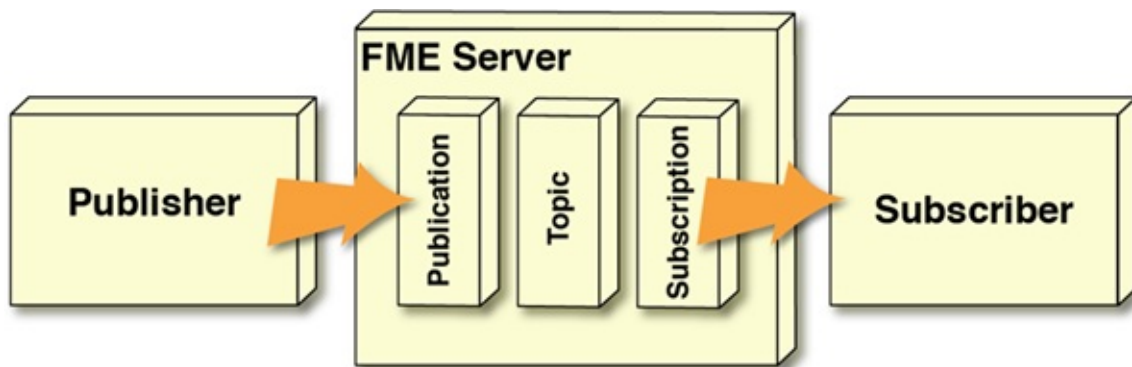
Notifications and Workspaces

Workspaces are what make FME Server the ideal engine for spatial data notifications. That's because the same functionality used to carry out spatial and tabular data transformations is also perfect for creating and transforming notification messages - with the added bonus of spatially based conditional processing!

Message Transformation

A Simple Notification Setup

The simplest setup for the FME Server Notification Service is where a Publisher sends an incoming message to a Publication, which in turn triggers a specific Topic. A Subscription listens to the Topic and passes its information as an outgoing message to any Subscriber:



For example, someone sends an email to FME Server, which triggers an outgoing email in response.

However, that scenario does not include any transformation/restructuring of the message contents. If the message needs to be processed in some way then an FME workspace can be employed within FME Server.

A workspace is the foundation of FME. For notifications it can be used to read an incoming message, extract spatial data from the message (regardless of format), carry out spatial transformations on that data, and then write the results in some way. The workspace can even read in extra data against which the message is to be processed.

It can also generate an outgoing message - possibly in response to some other spatial processing - and pass that on to a subscriber.

This blend of live messaging with Spatial ETL is unique.

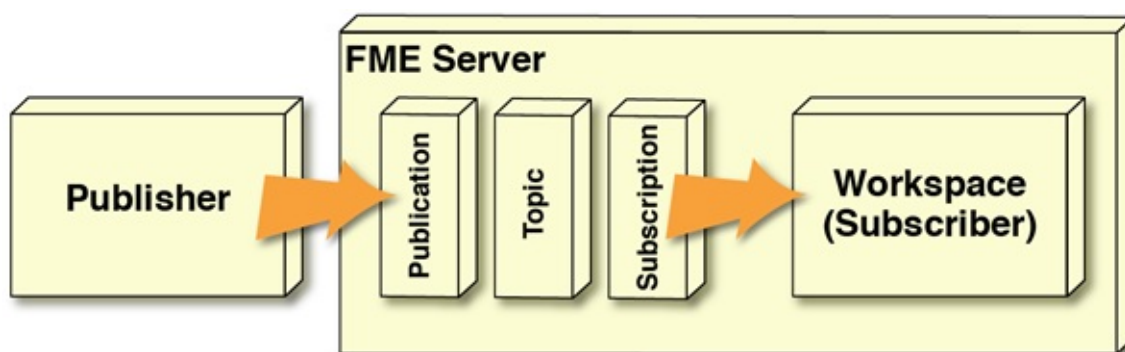
Let's look at two specific scenarios:

- A workspace runs in response to an incoming message
 - A workspace runs and triggers an outgoing message
-

Workspaces Responding to Incoming Message

Let's think about this logically. An incoming message triggers a topic. For a workspace to respond to an incoming message there must be mechanisms for it to listen to that topic and receive the information from it.

We already have terms for these mechanisms: **Subscription and Subscriber!**



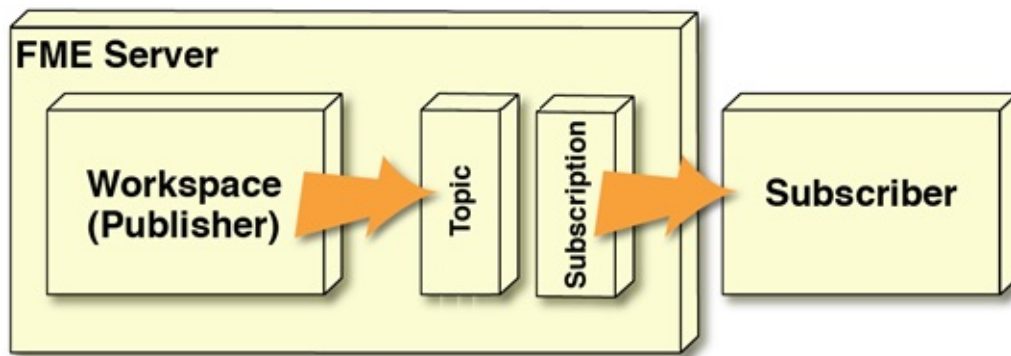
Yes, this scenario is set up by creating a Subscription. The workspace is literally a Subscriber to that Subscription and receives the message content from it. The only difference to the Simple Notification Setup is that the Subscriber now resides inside FME Server, rather than outside of it.

An example here is a Publisher/Publication triggers a workspace that writes the incoming data to a database.

Workspaces Triggering an Outgoing Message

To continue the logic, an outgoing message is activated by a topic. For a workspace to cause that outgoing message there must be mechanisms for it to trigger that topic and send information to it.

Again we already have terms for these mechanisms: **Publication and Publisher!**



Again the workspace is literally a Publisher. Once more that Publisher now resides inside FME Server, rather than outside of it.

However, there is another difference to the Simple Notification Setup from above: this scenario does not need a Publication component. A workspace is able to send information directly to a topic, without a Publication component being defined.

An example here is a workspace started as a scheduled task, that sends a notification email to an administrator once complete.

Full System

Of course, the above diagrams show half-systems; i.e. the workspace either responds to a message (it is a Subscriber) **OR** it causes a message (it is a Publisher).

However, it is just as appropriate to have a workspace that is both a Subscriber (it responds to a message) **AND** a Publisher (it also sends a message).



For example, a publisher sends a message containing an emergency event and the coordinates of the publisher's position. A workspace runs in response to that emergency, processes the coordinates, and sends a new message on to an emergency response unit.

Notice that this setup requires one Publication and two Subscription objects in FME Server. There is an outside publisher (with Publication), a workspace that is both subscriber and publisher (with Subscription), and an outside subscriber (with Subscription).

There are also two different Topic objects. That's important, as we shall see later.

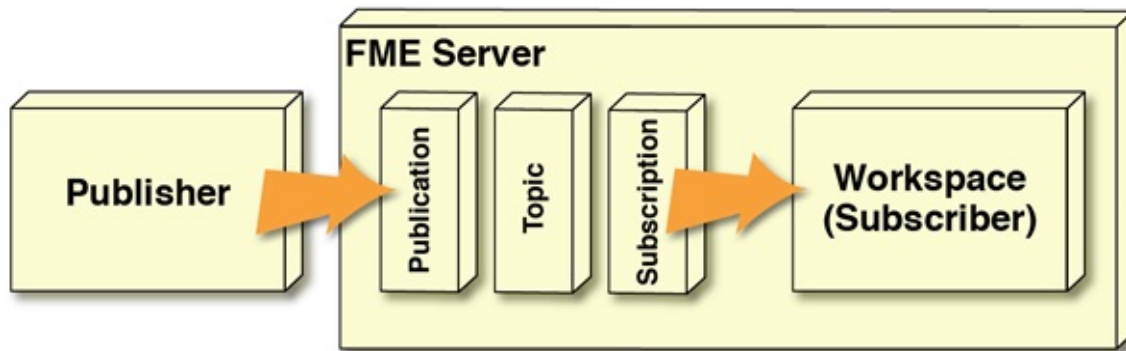
Miss Vector says...

When a workspace is part of a notification system, processing incoming messages, it is a...

- 1. Subscription*
- 2. Publication*
- 3. Protocol*
- 4. Client*

Workspaces as Subscribers

When a workspace needs to react to an incoming notification it is literally a Subscriber, subscribing to a topic. This is set up by connecting the workspace to that topic using a Subscription:



When that topic is triggered (by a Publication or any other means) any workspace connected to that topic through an *FME Workspace* protocol is run.

Registering a Workspace

As the above diagram shows, a workspace that receives notifications requires a Subscription for the topic to be communicated to the workspace.

There are two ways to create this Subscription:

- Through the FME Server Web Interface
- When publishing the workspace with the FME Server Publishing Wizard

Web Interface Workspace Subscriptions

The Subscription to create should have a protocol called (as you might have guessed) *FME Workspace*.

When that protocol is chosen for a new Subscription, the selected workspace is examined and a list of its published parameters provided:

New Subscription Validate

Name

Protocol FME Workspace

Topics Subscribed To × IncomingRoadInfo +

Protocol Settings

Repository Training

Workspace Running-Ex1-Complete.fmw ★

Workspace Published Parameters

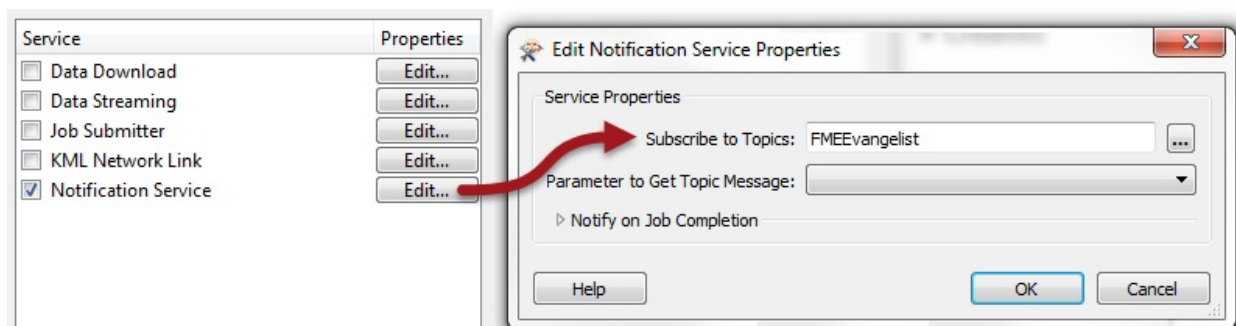
Source CSV (CommaSeparatedValue) File(s) C:\FMEData2016\Data\Engineering... ... ☐ Get Value from Topic Message i

Destination Google KML File C:\FMEData2016\Output\Drinking... ... ☐ Send Result Data in Success Topic Message i










Having the parameters in a dialog like this means it is simple and easy to set up a workspace to run however you want it to in response to a notification. Once created, when that topic is triggered, the workspace will run in response.

Publishing Wizard Workspace Subscriptions

The second way to create an FME Workspace Subscription is when publishing the workspace from Workbench to FME Server. At this point you may choose the Notification service and select a topic to receive information from:



Having done this you don't need to create a new Subscription object; FME will create one for you. It will look like this:

| <input type="checkbox"/> | Name | Subscription Type | Topics | Owner |  |
|---|---------------------------|-------------------|---|-------|---|
| <input type="checkbox"/> | iMark.iMark.DailyDBUpdate | workspace | 1  | iMark |  |
|   <input type="text" value="1"/>  / 1   <input type="text" value="100"/>  | | | | | Displaying 1 of 1 |

The name used for the new Subscription includes the repository name, the owner's name, and the name of the workspace.

WARNING

It's very important to be aware that FME creates this new Subscription. Otherwise there might be consequences in a number of scenarios; for example if you copy components to a new FME Server (in a Project), but don't include this one, then your notifications won't work on that server!

| Exercise 2 Building Updates Notification System | |
|---|--|
| Data | Building footprints (Esri Shapefile) |
| Overall Goal | Real-time updates to databases |
| Demonstrates | Running a workspace in response to a notification |
| Start Workspace | None |
| End Workspace | C:\FMEDData2017\Workspaces\ServerAuthoring\RealTime-Ex2-Complete.fmw |

As a technical analyst in the GIS department you have realized the overhead associated with pushing manual updates to your corporate database. Having read up about notifications in FME Server, you think that it should be possible to set up a system that automates this process.

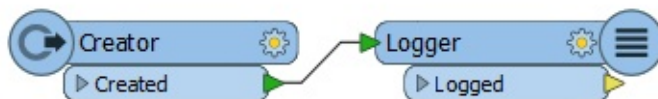
So far you have set up a system for added file notifications to be registered by FME Server. Now you must create a workspace to process these and publish it to FME Server. The workspace must then be triggered by a notification topic.

Miss Vector says...

This exercise continues where Exercise 1 left off. You must have completed Exercise 1 to carry out this exercise.

1) Create Workspace

Start FME Workbench and begin with an empty workspace. Simply add a Creator and Logger transformer:



This will give us a workspace to run in response to new files; albeit one that doesn't do much yet. We're just creating this to check that we can get the setup to work.

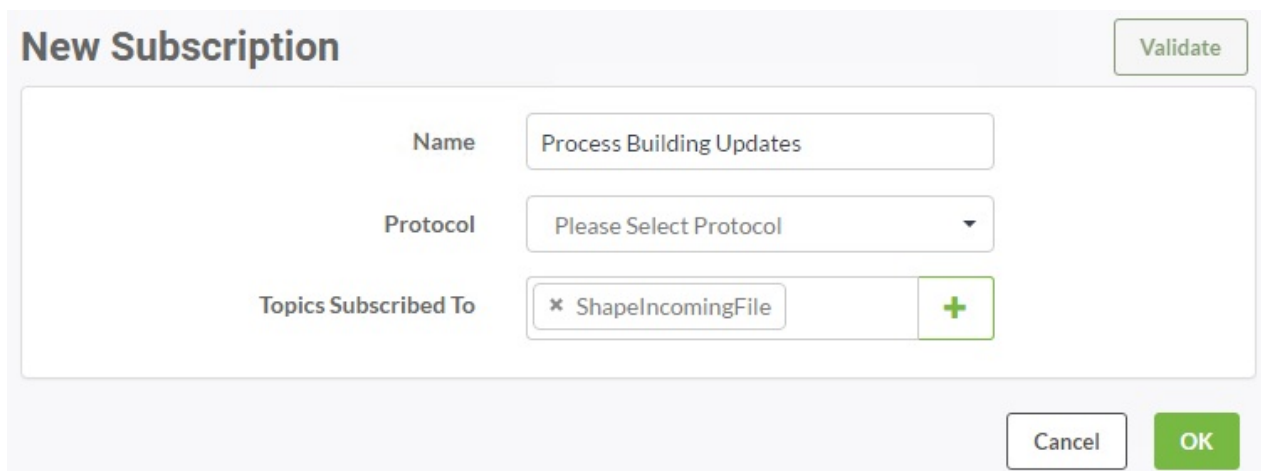
2) Save and Publish Workspace

Save the workspace and publish it to FME Server. We only need it to be run (not do anything special) so register it only with the Job Submitter service.

3) Create Subscription

Return to the FME Server web interface and navigate to the Notifications page. Click on the Subscriptions tab and click New to create a new Subscription.

Call the subscription "Process Building Updates". Subscribe to the topic ShapeIncomingFile:



New Subscription Validate

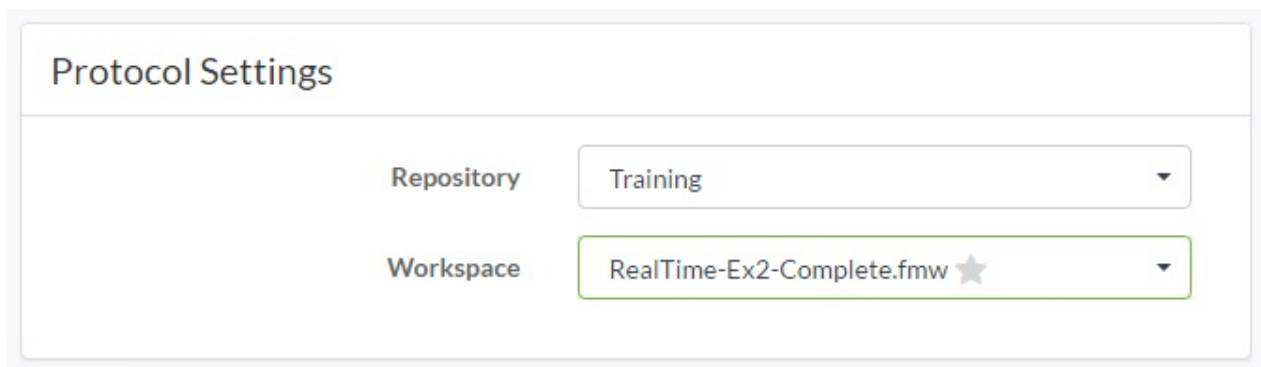
Name

Protocol

Topics Subscribed To +

Cancel OK

Now set the protocol to FME Workspace and select the workspace uploaded in the previous step:



Protocol Settings

Repository

Workspace ★

Click OK to create the subscription. This will cause the workspace to run every time an incoming Shape dataset triggers the ShapeIncomingFile topic.

4) Test Subscription

Test the subscription by uploading another Shapefile dataset.

NB: Since we set the Directory Watch to watch only for new files, the Shapefile uploaded should be a different one - or at least have a different name - to the first.

This time, instead of monitoring the topic (although it will appear there again), check the Jobs page. You should see that the workspace has been run in response to the new file:

Completed

Show Jobs For:

All Users

[Remove](#) [Remove All](#)

| <input type="checkbox"/> | Id | Workspace | Repository | Username | Status | Engine | Finished | Started | Priority |
|--------------------------|----|---------------------------|------------|----------|--------|---------------------|-------------------|-------------------|----------|
| <input type="checkbox"/> | 1 | RealTime-Ex2-Complete.fmw | Training | admin | | AP-FME64BIT_Engine1 | Today at 16:30:53 | Today at 16:30:52 | 100 |

This proves that the workspace has run.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Create a new FME Workspace Subscription*
- *Configure the Subscription to run a workspace in response to a Topic triggering*
- *Test the Notification system by verifying its success on the Completed Jobs page*

Message Content

The ability to interpret and process message content is a key reason for using FME workspaces as a Subscriber. They can receive the message and then transform it in whatever way is required, incorporating spatial data and spatial conditions as necessary.

Passing Messages

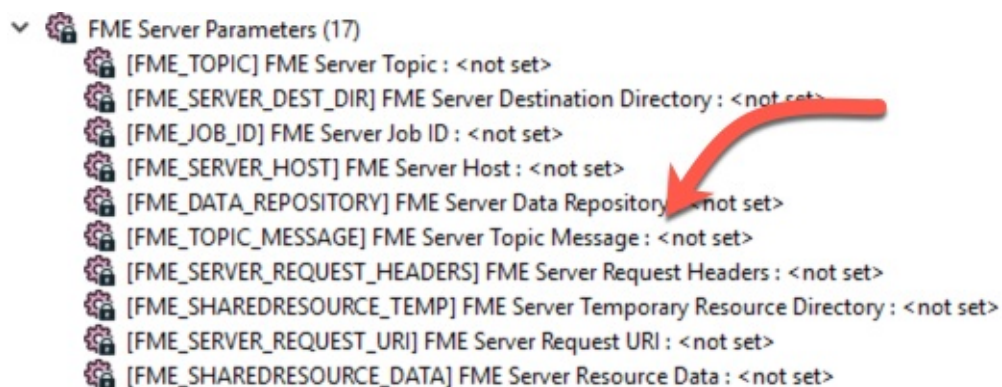
A topic passes on messages to a Subscription. When it does that for this protocol the topic message is written into a temporary JSON file and the name of that file passed to a workspace via a published parameter.

There are two ways to set up a workspace to retrieve the filename information.

The first way to obtain the message file location is to send it to a user parameter of your own choice, using a checkbox when the workspace Subscription is created:



The second method uses an FME Server parameter called FME_TOPIC_MESSAGE. This is one of the fixed FME Server parameters exposed inside FME Workbench:



Remember, this is only passing the name of the file containing the message - it does not include the message itself. For that you have to read the contents of that JSON file.

For instance, you could add a JSON (or Text File) Reader to your workspace and choose its source dataset parameter as the one to receive the name of the JSON file. That way the data is read directly into the workspace.

Alternatively, you could use a transformer such as the AttributeFileReader to read the file contents.

Interpreting the Message

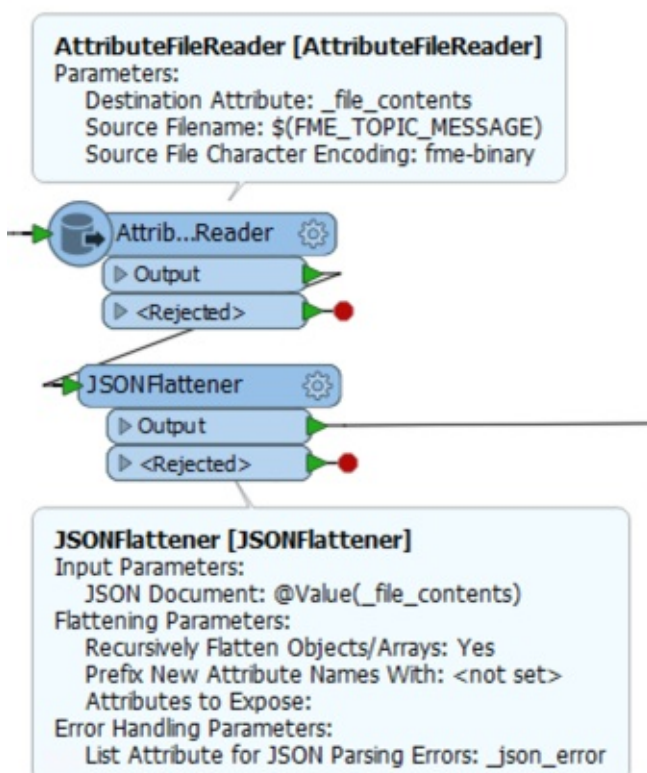
Having read the message into Workbench, it's then necessary to deal with it.

As noted, a Workspace Subscription writes message content to a temporary file and passes the name of that file to the workspace through a published parameter.

An incoming message can be scanned and processed with a number of different transformers. If the messages are in JSON format there are transformers such as the JSONExtractor and JSONFlattener. Similarly there are XMLFlattener and XMLFrangmenter transformers for XML content.

These transformers will convert the message from a JSON (or XML) string and into attributes that FME Workbench is able to process.

Here - for example - an author has added AttributeFileReader and JSONFlattener transformers to their workspace. The AttributeFileReader reads the JSON content (using FME_TOPIC_MESSAGE to identify the file) and the JSONFlattener transformer processes the JSON to extract the message as an attribute called subscriber_content:

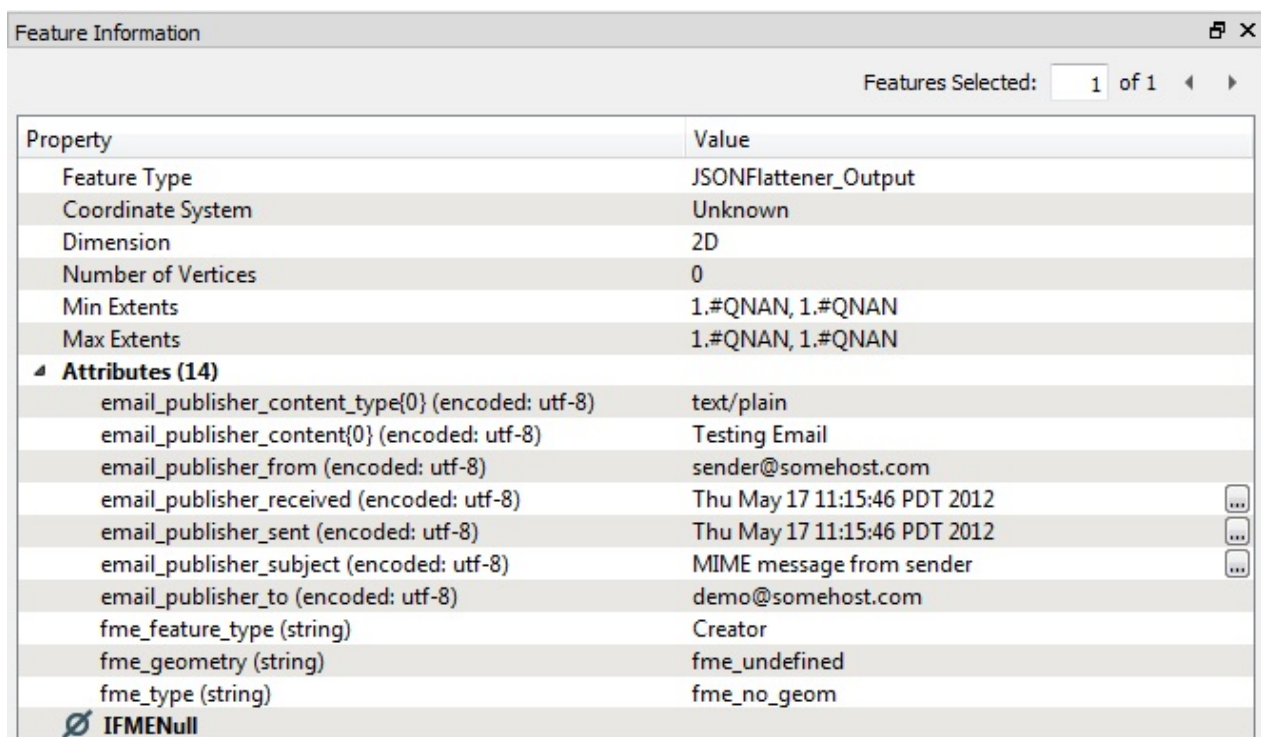



If the incoming message was an email then the JSON content may look like this:

```
{
  "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 18 11:15:46 PDT 2017",
  "email_publisher_sent": "Thu May 18 11:15:46 PDT 2017",
}
```

Notice how it includes the email from and to fields, plus the content itself.

When converted into FME attributes using the JSONFlattener transformer the result – as shown in the FME Data Inspector – will look something like this:



| Property | Value |
|--|------------------------------|
| Feature Type | JSONFlattener_Output |
| Coordinate System | Unknown |
| Dimension | 2D |
| Number of Vertices | 0 |
| Min Extents | 1.#QNAN, 1.#QNAN |
| Max Extents | 1.#QNAN, 1.#QNAN |
| Attributes (14) | |
| email_publisher_content_type{0} (encoded: utf-8) | text/plain |
| email_publisher_content{0} (encoded: utf-8) | Testing Email |
| email_publisher_from (encoded: utf-8) | sender@somehost.com |
| email_publisher_received (encoded: utf-8) | Thu May 17 11:15:46 PDT 2012 |
| email_publisher_sent (encoded: utf-8) | Thu May 17 11:15:46 PDT 2012 |
| email_publisher_subject (encoded: utf-8) | MIME message from sender |
| email_publisher_to (encoded: utf-8) | demo@somehost.com |
| fme_feature_type (string) | Creator |
| fme_geometry (string) | fme_undefined |
| fme_type (string) | fme_no_geom |
|  IFMENull | |

Now the content is available to the workspace as a set of attributes and can be processed as required.

Using the Message

What you do with the message depends on your required setup. If the topic is merely a trigger, and the message is unimportant, it could be ignored. However, in most cases the message content *is* important.

There are almost limitless ways FME could be used to process an incoming message. However, one useful example to consider is when the message contains the name of a dataset that the workspace should read. There are two possible scenarios there.

If an email arrives with a dataset attached, the attachment will be stored on the file system and part of the JSON content will specify the attachment as a path:

```
"email_publisher_attachment{0}":  
"C:\\Temp\\demo246129673106713_canada.dwg"
```

Similarly, the message might just include the name of a dataset that needs to be read (without it being an attachment). In both cases a reader can't be used because we need to extract the name of the dataset before we can read it, so a FeatureReader transformer - with its ability to read data mid-workspace - becomes the perfect solution.

Another interesting scenario is where the message contains an X/Y coordinate - for example the location of a person. Here the X/Y coordinate could be converted into a point feature with the VertexCreator transformer and from there any number of FME transformers could be used to carry out spatial processing such as a geofence.

| Exercise 3 Building Updates Notification System | |
|---|--|
| Data | Building footprints (Esri Shapefile) |
| Overall Goal | Triggering real-time updates to databases |
| Demonstrates | Processing Directory Watch notifications |
| Start Workspace | None |
| End Workspace | C:\FMEDData2017\Workspaces\ServerAuthoring\RealTime-Ex3-Complete.fmw |

Now that you have learned how to run a workspace in response to a notification, it's time to take that basic workspace and adjust it for your overall goal: to provide real-time updates to your corporate database.

The next step towards achieving this is understanding how to extract information from the notifications and configure an FME Workspace to process that incoming data.

Miss Vector says...

This exercise continues where Exercise 2 left off. You must have completed Exercise 2 to carry out this exercise.

1) Create Workspace

Start FME Workbench and begin with an empty workspace.

Select Readers > Add Reader from the menubar. When prompted set the parameters as follows:

| | |
|--------------------------|------------------------------|
| Reader Format | Text File |
| Reader Dataset | C:\FMEDData2017\readme.txt |
| Reader Parameters | Read Whole File at Once: Yes |

It doesn't matter what text file we use as the source right now; setting the source dataset in this step is only to satisfy the text file reader requirements. At run time, the source dataset will be replaced by the content of the incoming message.

2) Add JSONFlattener

Now add a JSONFlattener transformer to the workspace, after the Text File Reader. The incoming message is formatted as JSON, and this transformer will expose attributes on the canvas - making them available to work with.

Inspect the transformer parameters and - under the JSON Document parameter - select the attribute `text_line_data` as the source of the JSON content.

Add a Logger transformer to each output port on the JSONFlattener.

Dr. Workbench says...

Instead of using Text Reader > JSONFlattener we could have used the JSON Reader. Why didn't we? The JSON Reader requires a source file with valid schema. At this stage in the exercise, we do not have a file with this structure yet.

3) Publish to FME Server

Publish the workspace to FME Server, registering it under the Job Submitter service.

4) Update Subscription

Now log in to the FME Server web interface and navigate to the Notifications page.

Click on the Subscriptions tab and select the existing "Process Building Updates" Subscription to edit it.

Change the specified workspace, from the one created in Exercise 2, to the one uploaded in the previous step.

The change of workspace will cause a Source Text File parameter to appear. Here just select the checkbox to the right for *Get Value from Topic Message*.

Workspace Published Parameters

Source Text File(s)

C:\FMEData2017\readme.txt

...



Get Value from Topic Message ⓘ

Click OK to update the Subscription.

5) Test Topic

Once more (as in exercises 1 and 2) locate the source Shapefile datasets in

C:\FMEData2017\Data\Engineering\BuildingFootprints and create a compressed (zip) file from a set of Shapefiles (.dbf, .prj, .shp, .shx).

Be sure to give the zip file a different name to any used previously.

Copy the zip file into the Resources folder data\BuildingUpdates. You can do this through the file system (by copying the file to C:\ProgramData\Safe Software\FME Server\resources\data\BuildingUpdates) or by using the FME Server web interface.

6) Check Results

Open the Jobs page in the web interface. The completed jobs list should include the workspace you updated in the subscription. View or download the log file and look for the logged feature. You should find it has an attribute containing JSON, and a number of attributes extracted from the JSON.

| | |
|----------------------------|---|
| dirwatch_publisher_action | CREATE |
| dirwatch_publisher_content | ENTRY_CREATE C:\ProgramData\Safe Software\FME Server\resources\data\BuildingUpdates\update002.zip |
| dirwatch_publisher_path | C:\ProgramData\Safe Software\FME Server\resources\data\BuildingUpdates\update002.zip |

So now we know what the data looks like and can process it accordingly.

Mr. Flibble says...

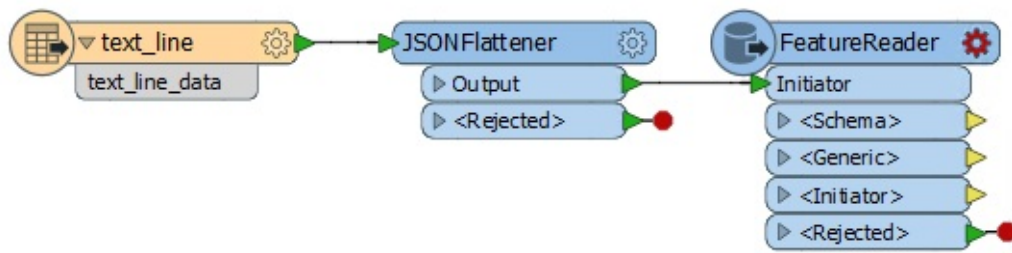
You may recognize these attributes from the Topic Monitoring exercise - indeed you can view the same information there without going through this process of adding Logger transformers!

7) Edit JSONFlattener Transformer

Back in FME Workbench inspect the JSONFlattener transformer parameters once more. Under Attribute to Expose add the attribute *dirwatch_publisher_path* by clicking the browse button and then manually typing its name.

8) Add FeatureReader Transformer

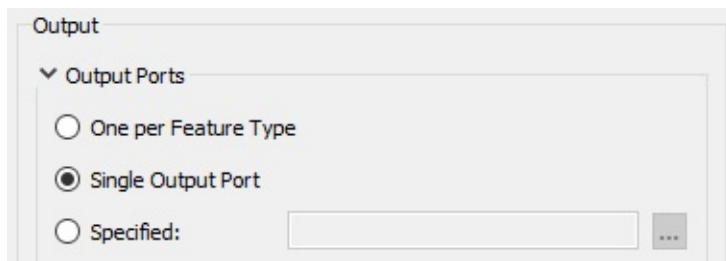
Now remove the Logger transformers and add a FeatureReader transformer to the output of the JSONFlattener:



This is a transformer that will let us read the contents of the dataset into the workflow mid-translation. Inspect the transformer's parameters and set the following values:

| | |
|-----------------------|--|
| Reader Format | Esri Shapefile |
| Reader Dataset | Select Attribute Value > dirwatch_publisher_path |
| Output Port | Single Output Port |

Select to have a Single Output Port:



You may receive a warning message, but it can be safely ignored.

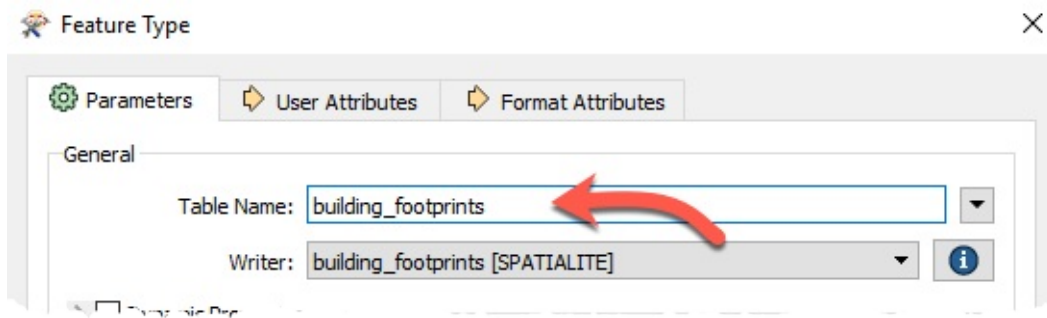
9) Add Writer

Having read the data from a Shapefile, we can now add it to the corporate database.

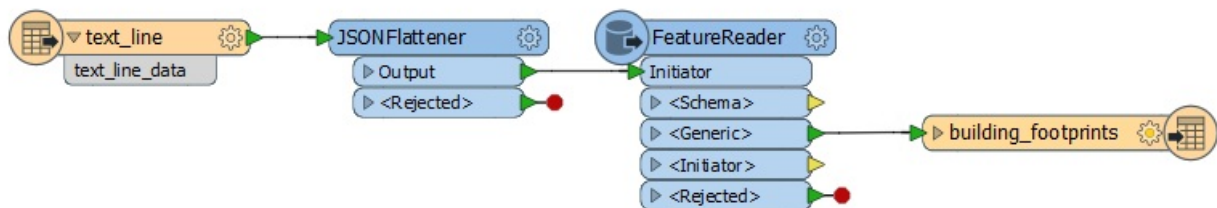
Select Writers > Add Writer from the menubar. When prompted set the parameters as follows:

| | |
|--------------------------|---|
| Writer Format | Spatialite |
| Writer Dataset | C:\FMEDData2017\Data\Engineering\BuildingFootprints\building_footprints.s |
| Writer Parameters | Overwrite Existing Database: No |
| Add Feature Types | Table Definition: Manual |

In the new feature type that is created, change the Table Name parameter to *building_footprints*:



Ensure that the Table Handling is set to "Create if Needed". Click OK to close the dialog and then connect the new feature type to the FeatureReader transformer's <Generic> output port.



10) Republish Workspace

Publish the workspace back to FME Server. If you have the same FME Workbench session open from the start of this exercise, you can use the Republish option on the toolbar or under the File menu.



11) Edit Subscription

Navigate to the Notifications page and open the Process Building Updates Subscription for editing. The parameters should now include one for the output database. Use the browse button to write the database into a Resources > Output folder (create the folder by manually typing the output field as shown below, if it doesn't yet exist):



CONGRATULATIONS

By completing this exercise you have learned how to:

- *Identify JSON attributes on an incoming Topic Message*
- *Use a FeatureReader transformer to read the dataset added to watched folder*

An Introduction to Email Notifications

It's important to cover email notifications in some detail because they are one of the most commonly used type of notifications on FME Server.

“Email” is a protocol that may be used by both Publication and Subscription components. Email Publications receive incoming email from Publishers and Email Subscriptions send outgoing email to Subscribers.

Email Protocols

In actual fact, email is not a single protocol. There are several protocols (methods of email transfer) that exist. FME Server supports two of these email-related protocols: **SMTP** and **IMAP**.

SMTP (Simple Mail Transfer Protocol) is the ability to *directly* send or receive an email through an email server built into FME Server.

IMAP (Internet Message Access Protocol) is an *indirect* process that connects to an email account on a server elsewhere. When that account receives an email the IMAP protocol passes it on to FME Server.

SMTP Publications

SMTP Publications are used when data is published to FME Server via a direct email. FME Server receives an email and triggers a Topic in response.

Such Publications are possible because FME Server includes a built-in email server as one of its components. However, this does require that the host server has a domain name registered with a DNS name server.

Sister Intuitive says ...

The steps to set up the built-in email server for notifications are documented in the FME Server Reference Manual.

However, FME Cloud instances are automatically configured for email notifications, and have a public domain name too, so you don't need to do any additional setup.

Creating an SMTP Publication

Creating an SMTP Publication is done in the Notifications section of the FME Server web interface, by choosing the Email (SMTP) protocol for a new Publication.

New Publication Validate

Name

Protocol

Topics To Publish To +

Protocol Settings

Email User Name

Once the protocol type is selected, the publication must also be given a name and an existing topic chosen to be triggered. The final parameter is the Email User Name; in the above example it is set up to be RoadInfo. Note that you do not need to add *@FMEServer-Hostname.com* to the Email User Name.

Notice the author is prefixing the Publication and Topic names with "Incoming" so that there is no confusion over which topics are being used for incoming notifications and which ones for outgoing.

Now, whenever an email is sent to RoadInfo@FMEServer-Hostname.com – for example someone is sending an email to report snow on the road – the *IncomingRoadInfo* topic is triggered.

IMAP Publications











IMAP (Internet Message Access Protocol) is a variation on email for incoming (Publication) notifications.

Instead of using the built-in email server, an IMAP Publication connects to another email server and monitors it for incoming email. When an email arrives in that account, any Topic tied to that Publication is triggered.

Creating an IMAP Publication

Like the SMTP protocol, IMAP Publications are set up in the Notifications page of the web interface, by creating a new Publication and (in this case) choosing the Email (IMAP) protocol:

Protocol Settings

| | |
|---|---|
| IMAP Server Host  | <input type="text"/> |
| IMAP Server Port  | <input type="text" value="993"/> |
| IMAP Email Account  | <input type="text"/> |
| IMAP Email Password | <input type="password"/> |
| Connection Security  | <input type="text" value="SSL/TLS"/> |
| Verify SSL Certificate  | <input type="text" value="Yes"/> |
| Poll Interval  | <input type="text" value="5"/> <input type="text" value="Minutes"/> |
| Emails to Fetch  | <input type="text" value="New Emails Only"/> |
| Filter Type  | <input type="text" value="Simple"/> |
| Subject Filter  (optional) | <input type="text"/> |
| Download Attachments To | <input type="text" value="Specify Location"/>  |

Notice that most parameters are for defining the IMAP (Email) server connection.

Two important parameters let you decide the interval to check for emails and decide whether to fetch all unread emails or new emails only.

There is also a parameter to select an FME Server resource location in which to store any email attachments.

Monsieur D. Server says ...

Most email servers support IMAP functionality, as do the majority of cloud-based email providers such as Gmail, Outlook.com, Yahoo!, etc; so it's very easy to have FME Server scan a Gmail account (for example) for incoming mail, and then act on its contents.

Email Subscriptions

Email Subscriptions are used to have FME Server send an email in response to a Topic. The built-in email server in FME Server is only for incoming mail, as is the IMAP protocol, and so messages need to be sent via an existing (external) SMTP email server.

Setting up an Email Subscription

Creating an Email Subscription is done in the Notifications page of the web interface, by choosing the Email protocol for a new Subscription.

The Subscription is given a name and an existing topic chosen to be triggered. There are many more parameters for outgoing mail because the full SMTP server connection parameters need to be defined.

Protocol Settings

| | |
|-----------------------------|--------------------------------------|
| SMTP Server ⓘ | <input type="text"/> |
| SMTP Server Port ⓘ | <input type="text" value="25"/> |
| SMTP Account ⓘ (optional) | <input type="text"/> |
| SMTP Password ⓘ (optional) | <input type="password"/> |
| Connection Security ⓘ | <input type="text" value="SSL/TLS"/> |
| Email To ⓘ (optional) | <input type="text"/> |
| Email Cc ⓘ (optional) | <input type="text"/> |
| Email Bcc ⓘ (optional) | <input type="text"/> |
| Email From ⓘ | <input type="text"/> |
| Email Subject | <input type="text"/> |
| Email Format | <input type="text" value="Text"/> |
| Email Template ⓘ (optional) | <div><input type="text"/></div> |

Various fields for the email itself (From, To, Subject, Template) do not need to be hard-coded and can be passed through to the Subscription from a workspace. Another important parameter is the Email Format, which can either be plain text or HTML.

See the following sections on workspaces for information on how to generate content for outgoing emails.

| Exercise 4 | Building Updates Notification System |
|-----------------|--|
| Data | Building footprints (Esri Shapefile) |
| Overall Goal | Provide email-driven notifications for updates |
| Demonstrates | Email publications and Notification service |
| Start Workspace | RealTime-Ex4-Begin.fmw |
| End Workspace | RealTime-Ex4-Complete.fmw |

As a technical analyst in the GIS department you were involved in a recent project to set up a Directory Watch solution for users to automatically update the corporate database.

Having learned that not all users are able to access the internal network where FME Server is hosted, you think that it should be possible to also set up a system that uses email-based automation to handle the same updates.

1) Create Topic

The first step is to create a Topic that will be triggered by the email. Log in to the FME Server web interface and navigate to the Notifications page.

Click the Publications tab and then select New.

Enter "Email Receiver" as the Name. Then click in the text box under Topics to Publish To. Type in *ShapeIncomingEmail* and click on it to add. This will create a new Topic and assign it to this Publication.

New Publication Validate

Name:

Protocol:

Topics To Publish To: +

ShapeIncomingEmail (new)

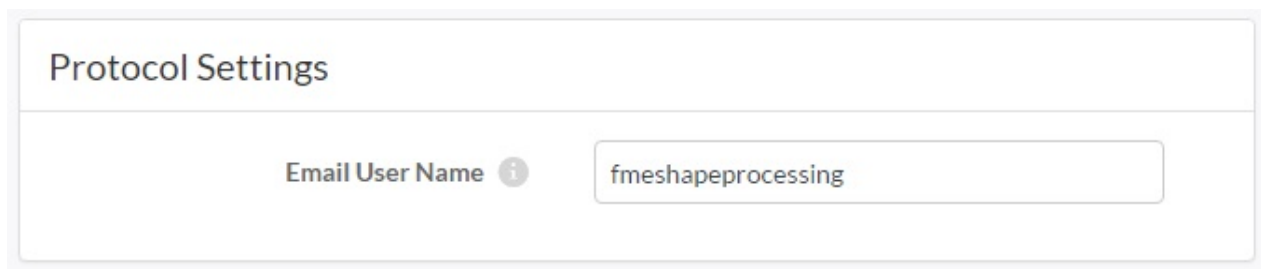
Cancel OK

The new Publication can be created to use either the Email (SMTP) protocol or the Email (IMAP) protocol.

SMTP is easier to set up but FME Server must reside on a server with a proper DNS record (all FME Cloud and Training machines will have this). IMAP is necessary when FME Server resides on an internal network.

Email Protocol

To use the SMTP protocol select Email (SMTP) as the Publication Protocol. This will open the Email User Name parameter. Enter a name for receiving email, for example *fmeshapeprocessing*



The screenshot shows a 'Protocol Settings' window. Inside, there is a label 'Email User Name' followed by an information icon (i) and a text input field. The input field contains the text 'fmeshapeprocessing'.

Clicking OK will create an email address *fmeshapeprocessing@<hostname>* - for example:

| Host | Example Email Address |
|------------|---|
| FME Cloud | fmeshapeprocessing@myfmeserver.fmecloud.com |
| Amazon AWS | fmeshapeprocessing@ec1-23-456-789-012.compute-1.amazonaws.com |

Now all emails sent to that address will trigger the ShapeIncomingEmail topic.

IMAP Protocol

To use the IMAP protocol select Email (IMAP) as the Publication Protocol. This will open a number of other parameters. Enter them according to your email account.

In case it is of use, the server information for Gmail, Outlook, and Yahoo! are as follows:

| IMAP Server Host | imap.gmail.com | imap-mail.outlook.com | imap.mail.yahoo.com |
|-------------------------|----------------|-----------------------|---------------------|
| Server Port | 993 | 993 | 993 |
| Connection Security | SSL/TLS | SSL/TLS | SSL/TLS |
| Verify SSL Certificates | Yes | Yes | Yes |

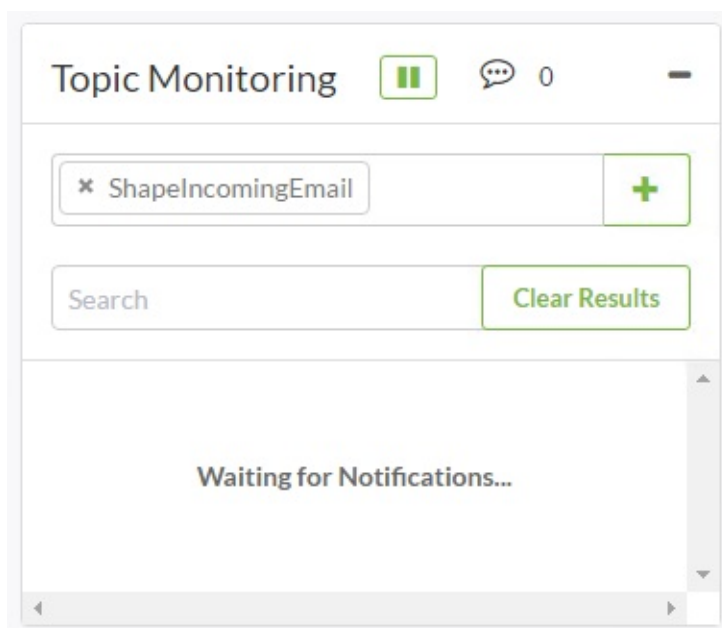
You will also need to check the settings in your email account to make sure IMAP is turned on. Regardless of the email provider, you should set these parameters as follows:

| Parameter | Value |
|-------------------------|----------------------------------|
| Poll Interval | 1 Minute |
| Emails to Fetch | New Emails Only |
| Download Attachments To | A Resource folder of your choice |

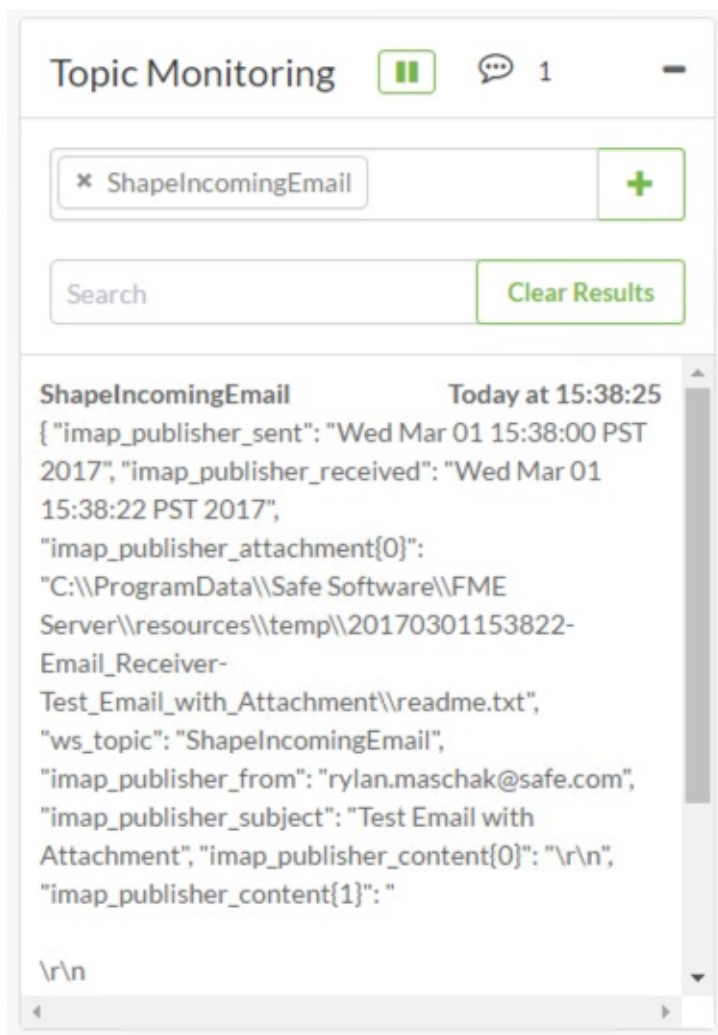
You may select any Resource folder for attachments to be saved to; but (if you have already completed exercise 1-3) don't choose the BuildingUpdates folder, else you'll cause the previous topics to be triggered by each email attachment!

2) Test Publication

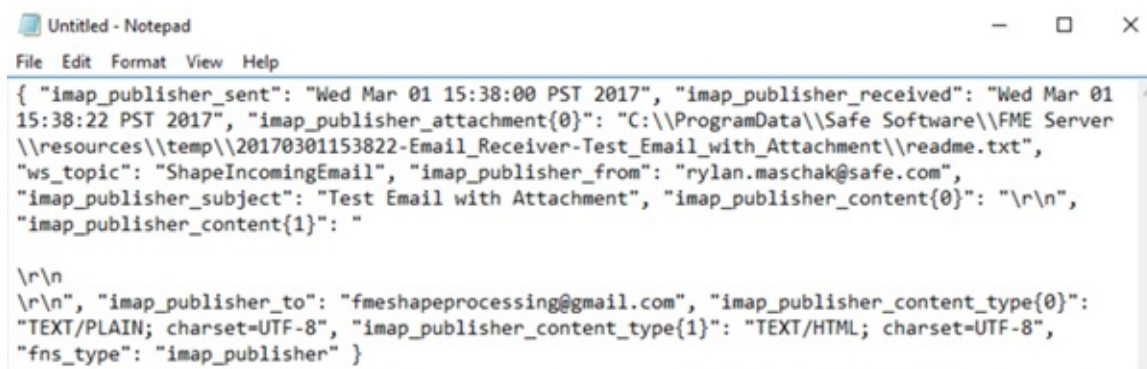
Now let's test the publication. In the Notifications page on FME Server, click the tab marked Topics. Set up Topic Monitoring to watch the topic *ShapeIncomingEmail*:



Now send an email *with an attachment* to the address selected for the new publication. When the email is received by FME Server (SMTP), or FME Server fetches it (IMAP), the topic will be triggered with a message. (Remember that an IMAP publication only checks for an email every 60 seconds, so the result might not be immediate!)



Recall that in the previous exercise you used the Logger Protocol and Logger transformers to record the JSON formatted notification message. The same information is displayed in the Topic Monitoring window. So copy the text from the Topic Monitoring window and paste it into a text editor for use later in this exercise.



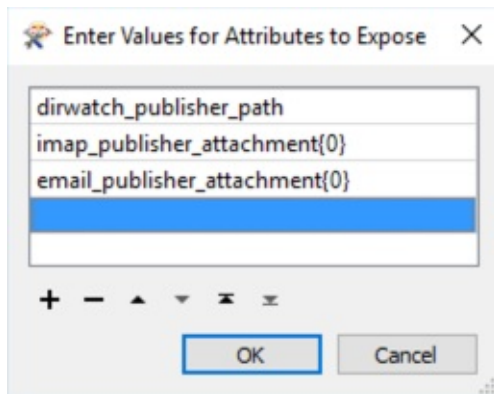
3) Update Workspace

You already have a created a workspace in FME Workbench to handle incoming notifications from Directory Watch. Let's modify the workflow so that it can work with both

Publication protocols. Open the existing workspace

C:\FMEDData2017\Workspaces\ServerAuthoring\RealTime-Ex4-Begin.fmw in FME Workbench.

Open the JSONFlattener parameters, and add *imap_publisher_attachment{0}* and *email_publisher_attachment{0}* under Attributes to Expose:



You can see these are two of the available attributes that are returned by the Topic Message.

Ms Analyst says...

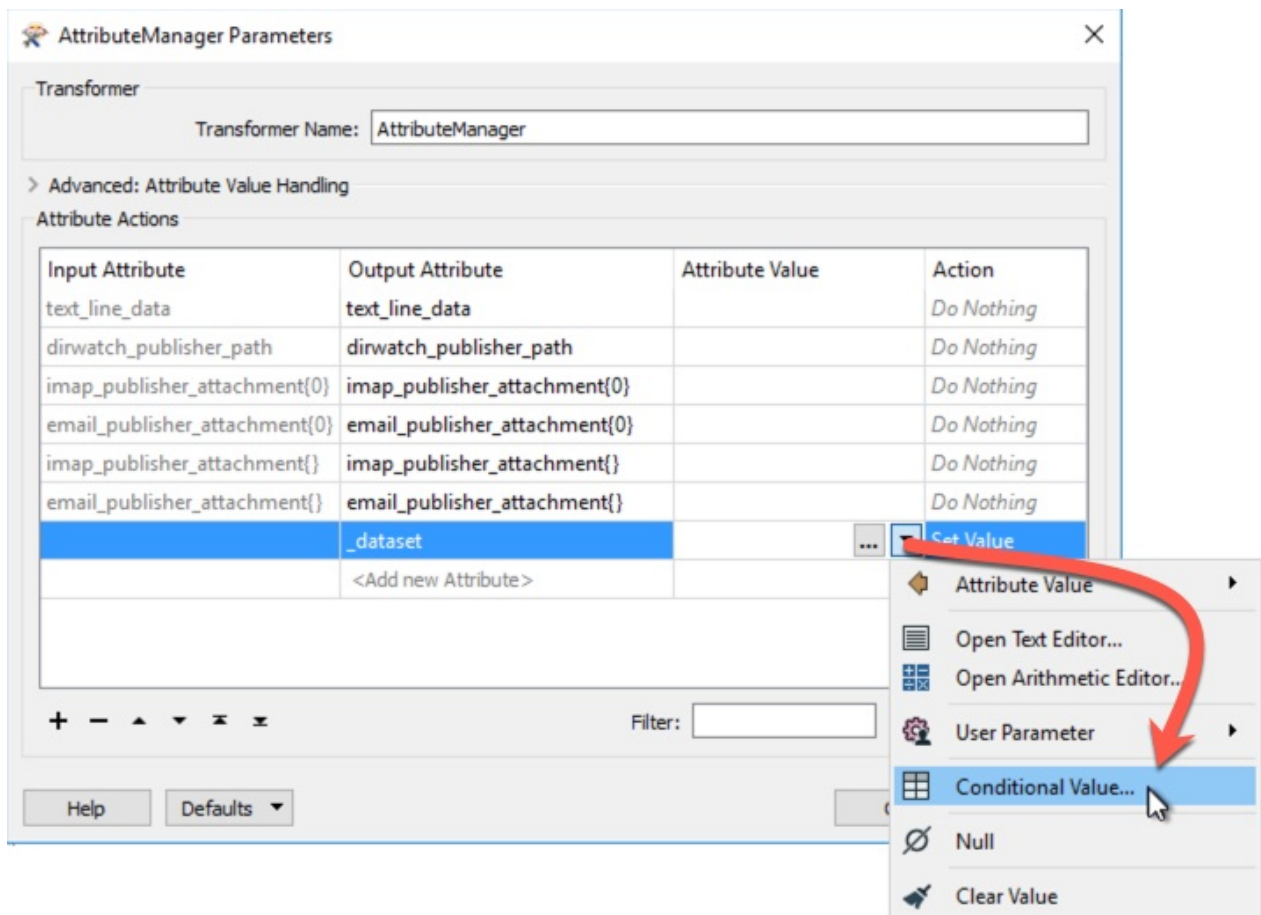
*Adding both *imap_publisher_attachment* and *email_publisher_attachment* modifies this workspace so that it can work with both Email (SMTP) and Email (IMAP) Publications!*

4) Add AttributeManager

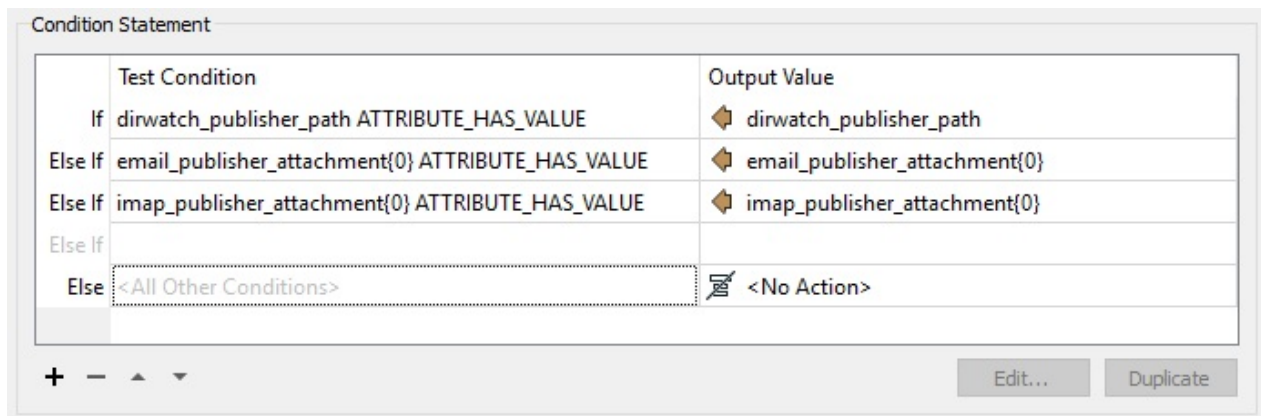
The next step is to insert a transformer that will determine where the data is coming from (Directory Watch or an Email Publication) - this is a task where conditional statements are invaluable.

Add an AttributeManager transformer in between the JSONFlattener and FeatureReader. Open the parameters and add *_dataset* as a new Output Attribute.

Select, from the drop-down menu, the option to set the attribute as a Conditional Value:



Configure the Conditional Value as follows:



| Attribute | Test | Set _dataset To |
|-------------------------------|-----------------------|-------------------------------|
| dirwatch_publisher_path | Attribute has a value | dirwatch_publisher_path |
| email_publisher_attachment{0} | Attribute has a value | email_publisher_attachment{0} |
| imap_publisher_attachment{0} | Attribute has a value | imap_publisher_attachment{0} |

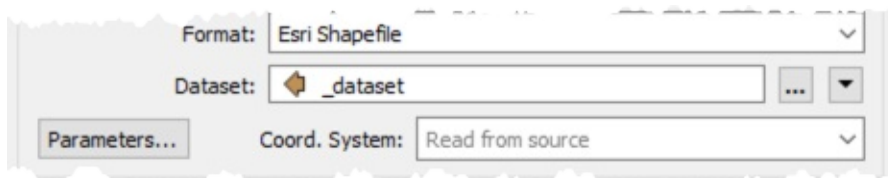
In other words:

- If *dirwatch_publisher_path* has a value, then copy that value into the *_dataset* attribute.
- Else, if *email_publisher_attachment{0}* has a value, then copy that value into the *_dataset* attribute.
- Else, if *imap_publisher_attachment{0}* has a value, then copy that value into the *_dataset* attribute.

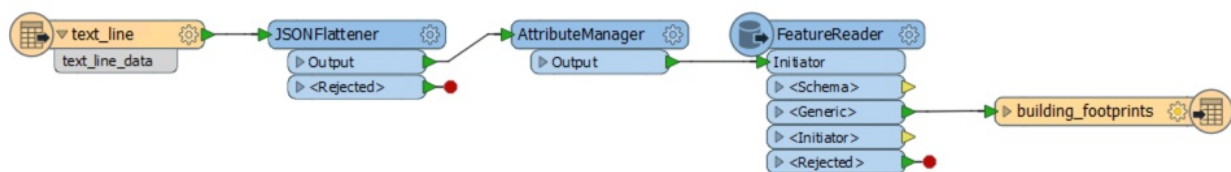
So *_dataset* gets the location of the data to be processed, whether it comes from the directory watch notification, or an email notification of either type.

5) Edit FeatureReader

The final step is to change the Dataset parameter in the FeatureReader transformer. Instead of pointing to *dirwatch_publisher_path*, it should be changed to point at the new *_dataset* attribute:



The workflow should now look like this:

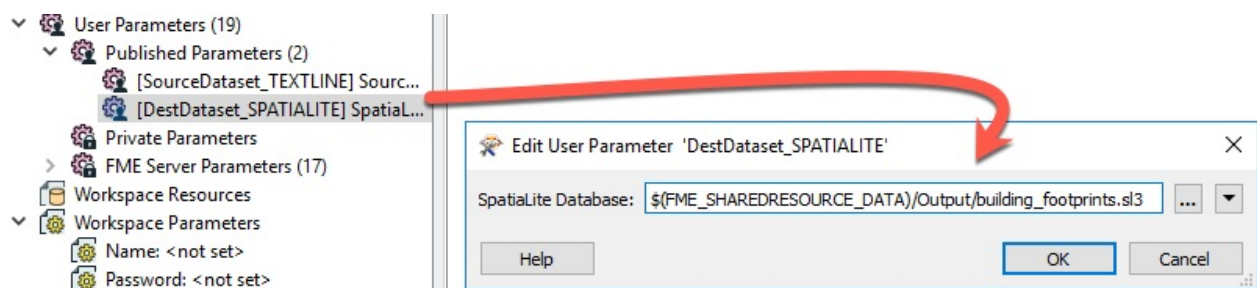


6) Edit User Parameter

As with Exercise 3, specify the output dataset to be written into the FME Server Resources Folder.

Locate the user parameter *DestDataset_SPATIALITE* (under User Parameters > Published Parameters in the Navigator window) and double-click it to open an editor dialog.

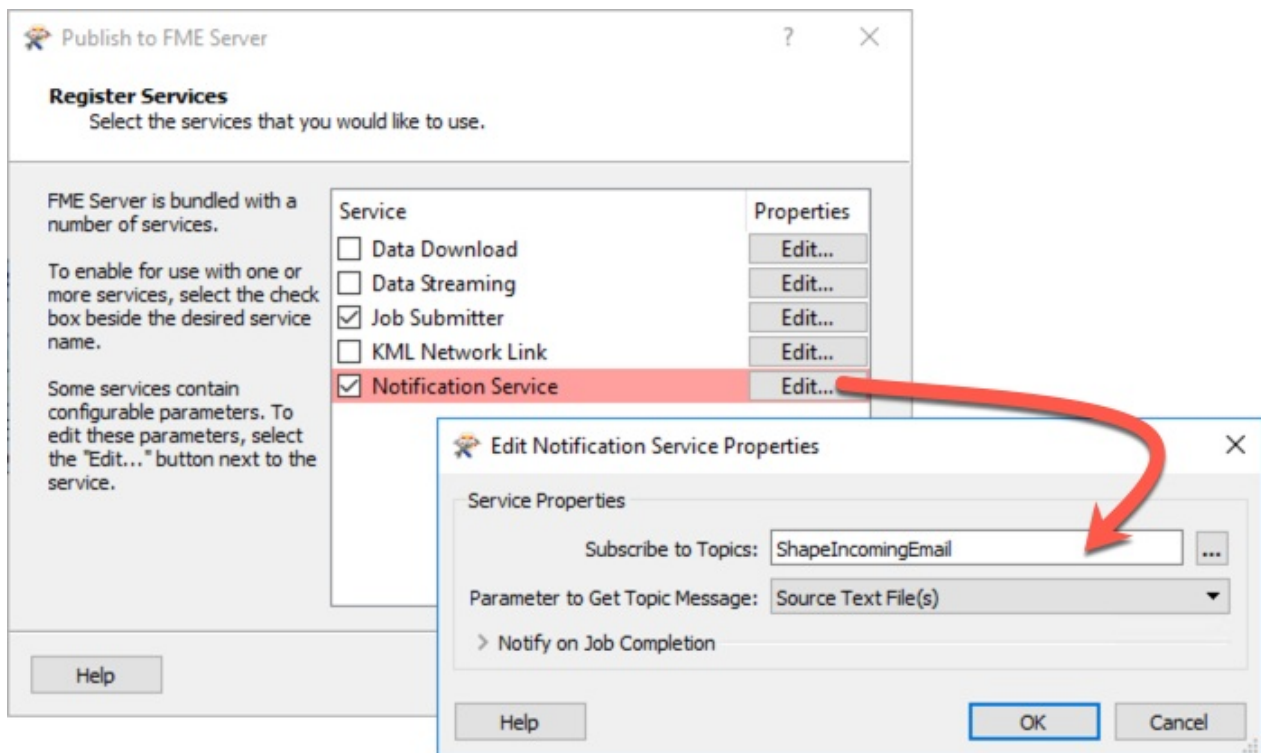
In that dialog enter *\$(FME_SHAREDRESOURCE_DATA)/Output/building_footprints.sl3*



7) Publish Workspace

Publish this workspace to FME Server, registering it under the Notification service. When the Notification service is selected, it is highlighted in red indicating its parameters need to be configured.

Click the "Edit" button and set *ShapeIncomingEmail* for the "Subscribe to Topics" parameter. Set the "Parameter to Get Topic Message" as *Source Text File(s)*:



8) Update Directory Watch Subscription (Optional)

If you have completed Exercise 3, using the FME Server web interface you can set the "Process Building Updates" Subscription to point at this new workspace.

9) Test Workspace

Test the workspace by sending an email to the Publication email address. Be sure to attach a zip file of the Shapefile datasets (.dbf, .prj, .shp, .shx) from C:\FMEData2017\Data\Engineering\BuildingFootprints to the email.

You can verify if the workflow was successful by checking the Completed Jobs page and the timestamp of the Spatialite database in Resources > Data > Output in the FME Server web interface.

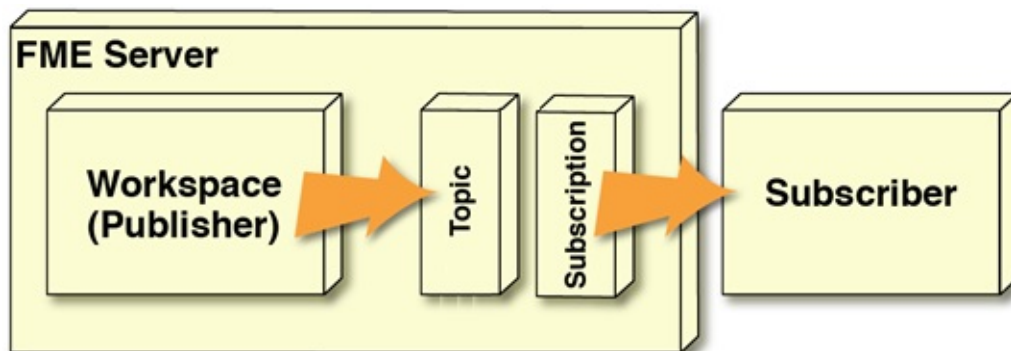
CONGRATULATIONS

By completing this exercise you have learned how to:

- *Create an Email Publication*
- *Create a new FME Workspace Subscription as part of the Publishing process*
- *Use incoming email to trigger Topics/Notifications*
- *Configure a workspace to handle triggers by multiple Publication types*

Workspaces as a Publisher

When a workspace needs to send an outgoing notification it is literally a Publisher, publishing to a topic. This is set up simply by connecting the workspace to that topic.



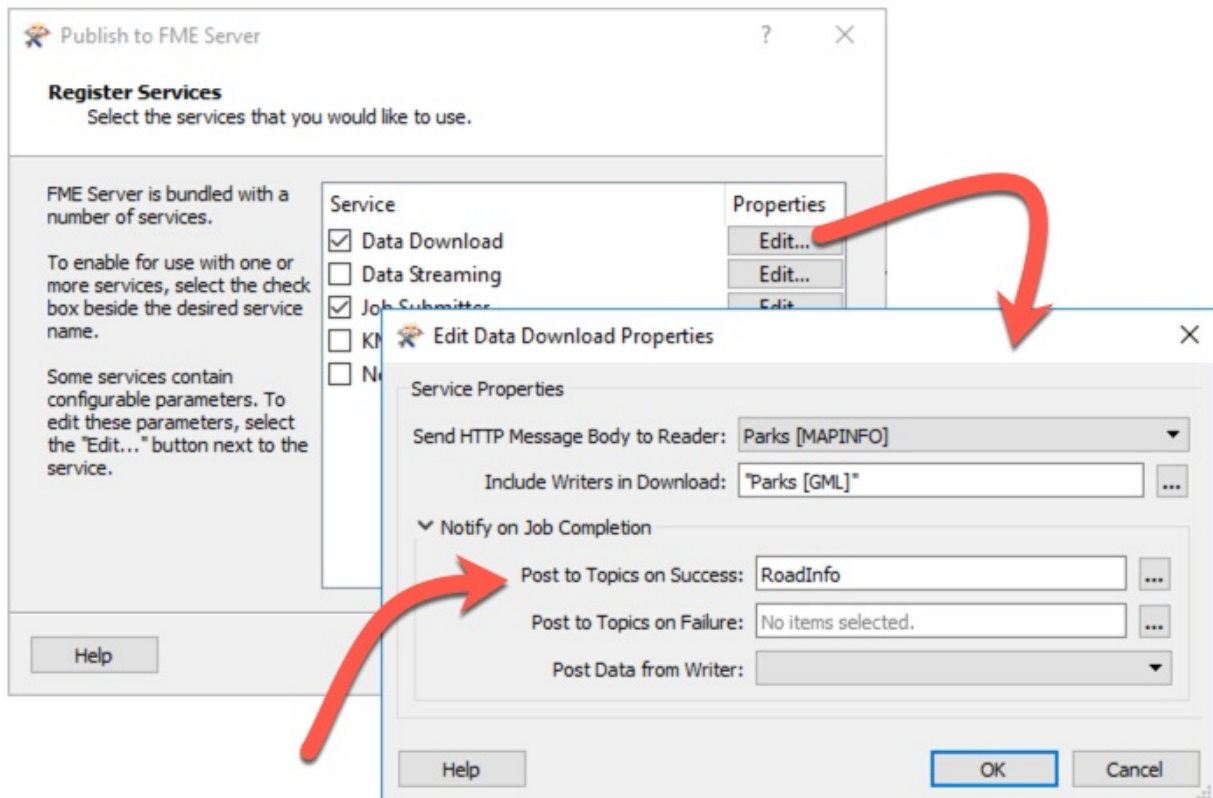
Publishing notification messages from a workspace is useful for both sending messages containing content from the data being processed, and for reporting the status of a translation, such as whether it has run successfully or ended in a failure.

Workspaces can be set up to trigger topics in one of two ways:

- Registering with the topic when published
- Using a transformer

Registering a Workspace as a Publisher

The first way to have a workspace publish a message to a topic is to register it with that topic when the workspace is published to FME Server:



Here, for example, the workspace is registered with the Data Download service. When the workspace is run using that FME Server Web Service, it will post a message to the RoadInfo topic on successful completion.

Notifications are sent once the workspace is complete. The way in which the workspace is run is not important. Notifications can be sent for workspaces run using the Job Submitter Service, or the Data Download Service, or any other service.

Having two Topic parameters is useful because different notifications can be sent depending on whether the workspace succeeds or fails.

A third parameter - Post Data from Writer - defines the content of the message being sent (more on that to come).

TIP

If you remember the diagram for a workspace that publishes notifications, you'll remember that there is no Publication that must be created. The workspace can communicate directly with the topic.

Ms. Analyst says...

Four Topics are pre-installed with FME Server to handle workspace notifications. There are two topics for the Job Submitter Service (one for success, one for failure) and two for the Data Download Service (again, one for success, one for failure). These topics are triggered automatically by FME Server so a system administrator – for example – could subscribe to these topics to receive notifications from all workspaces.

Miss Vector says...

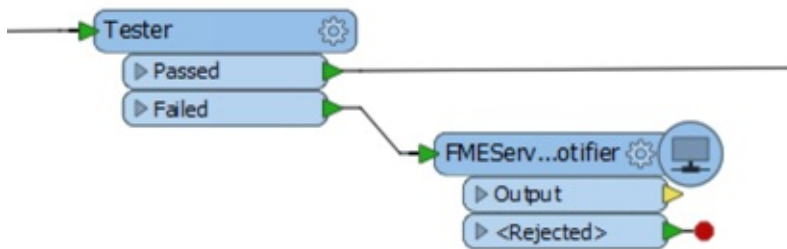
I want my workspace to send me an email when it is run, so I know when people are using it to download data. When I publish it, what should I register it to?

- 1. The Notification Service*
- 2. The Data Download Service*
- 3. The Email (SMTP) Protocol*
- 4. The FME Workspace Subscriber Protocol*

Workspace Publishing with a Transformer

Instead of having a workspace send a notification at its completion, it could instead send a notification through a special transformer called the *FMEServerNotifier*.

Here a workspace author is sending an FME Server notification when a feature fails the conditions of a Tester transformer:



Notification properties are set in the transformer parameters. The parameters include those for connecting to FME Server, one for the topic to post to, and one for the message to be included.

Here the author intends to publish information to a topic called RoadInfo:

Parameters

Topic: RoadInfo

Content: Road Conditions are: @Value(RoadConditions)

There are two advantages to issuing a notification this way, over using the registration method:

- The workspace can issue a notification *during* a translation, rather than at the end of it.
- The workspace does not need to be run on FME Server to generate an FME Server notification. It will produce the same notification when run using FME Desktop.

The disadvantage is that you won't know whether the workspace completed successfully - or not - when the notification is issued.

TIP

Just like triggering a topic at workspace completion, this method needs to Publication to be created. In fact, assuming the topic already exists, this method can be set up without having to open the FME Server web interface at all!

Miss Vector says...

I've got a workspace that reads 50,000 features, transforms them, and writes them out. If I want to send a single notification that the features have been read, which combination of transformers would be of most use?

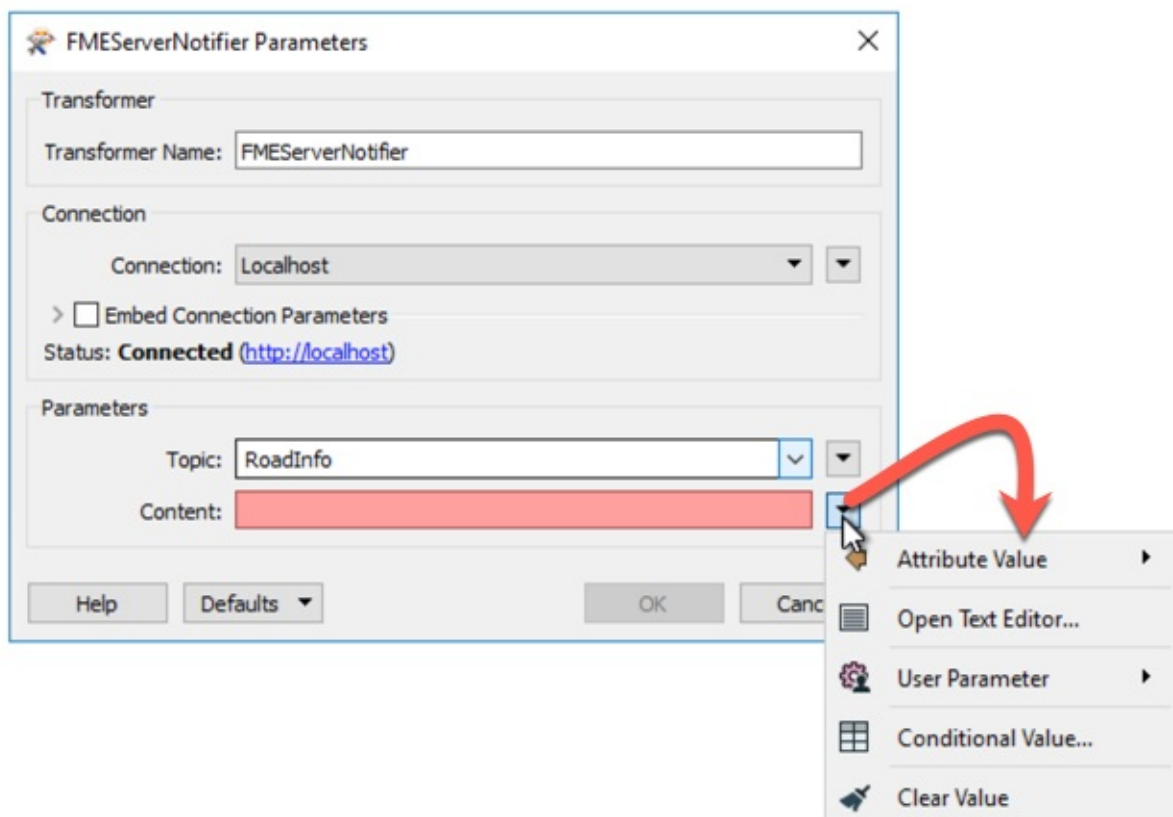
1. *Creator/FeatureWriter/FMEServerNotifier*
2. *Creator/FMEServerJobSubmitter*
3. *Creator/FeatureReader/FMEServerNotifier*
4. *FeatureHolder/Sampler/FMEServerNotifier*

Outgoing Message Content

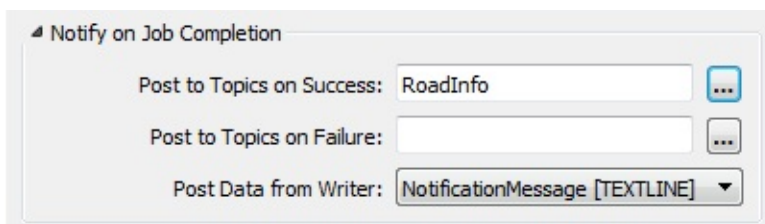
When a workspace triggers a topic it can include (like other Publishers) a message.

The ability to construct message content from a variety of sources – including spatial – is a key reason for using FME workspaces as a Publisher. Workspaces can transform data in multiple ways, use it to construct a message, and then dispatch that message to users as a notification.

When a topic is triggered by the FMEServerNotifier transformer, the message can be defined by an attribute, constructed in a text editor, obtained from a user parameter, or even set up as a conditional value:



When a topic is triggered by registering the workspace with a service topic, the message is sent via a writer:

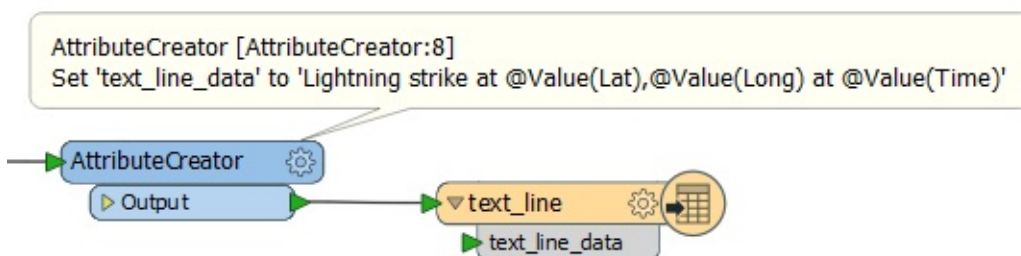


Here the message is being passed through a Text File writer.

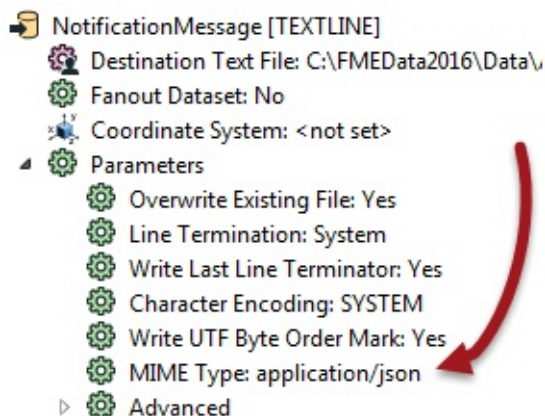
Content Format

For the purposes of FME notifications, the content of a workspace publication can be in any format, maybe even a plain text message. However, for the benefit of a web-based subscriber the outgoing message is often in a JSON or XML format.

For example, here a workspace constructs a plain-text weather (lightning) alert using an AttributeCreator. The message attribute is connected to a Text File Writer in order to provide a means for publishing the outgoing message:



Importantly, the Text File Writer has a MIME type setting that can be applied:



So even if the message were JSON or XML, standard practice is to use a text writer.

Ms Analyst says...

A Text File writer is the commonest method of supplying messages, regardless of format. Even if the message is constructed as a snippet of JSON or XML it would still be passed through the Text File writer.

Email Content in a Workspace

The message being passed out of a workspace may need to be in the form of an email. This, like any other message, can be constructed as an attribute with transformers.

For example, for a JSON-formatted email you could construct an attribute like this:

```
{
  "email_to" : "notifications@enduser.com",
  "email_cc" : "",
  "email_from" : "notifications@fmeserver.com",
  "email_replyto" : "",
  "email_subject" : "Notification Subject",
  "subscriber_content" : "Notification Content"
}
```

This could be created using an AttributeCreator transformer, where the email addresses and content can be substituted with attribute or parameter values as required.

The actual email content could be plain text or, if the Email format is set to HTML in the Notification GUI, it could be constructed as HTML instead.

Ms Analyst says...

The FMEServerEmailGenerator is a custom transformer available in FME Workbench for generating email content in the correct format and structure

Workspaces as a Subscriber AND a Publisher

We've now looked at how to set up a workspace to be a Subscriber (reacting to incoming messages) and how to set up a workspace as a Publisher (sending outgoing messages). A workspace can do each task individually, but when it is set as both a Subscriber and a Publisher, the overall setup looks like this:



In this scenario the same workspace that receives an incoming notification also sends an outgoing notification. It is set up as a Subscriber with a Subscription component in order to receive incoming notifications, and set up as a Publisher by registering it with the notification parameters on a different service.

For example, details of a lightning strike are received via a LightningStrike topic. A Publication watching for this topic starts a workspace that processes the incoming information; for example it determines which state/province/county the strike occurred in. The workspace then creates a new message and dispatches it to subscribers via a WeatherAlert topic.

The important thing to realize is that the scenario must involve at least two different topics.

A major problem occurs if you set up such a scenario but attempt to use the same one topic for incoming and outgoing. That's because the workspace would be set up as a subscriber to each message that it is publishing! It would republish each of its outgoing messages, causing an infinite loop of messages!

Daily Interop Reporter, Chad Pugh-Litzer says ...

Using the same topic for incoming and outgoing notifications is like setting up your email to forward all incoming messages to yourself, or automatically re-tweeting your own tweets. You end up with a loop that continues until you crash the system!

To avoid confusion it's a good idea to clarify which topics are for in and out communication through their names; for example, use LightningStrike_In and LightningStrike_Out to differentiate.

| Exercise 5 Building Updates Notification System | |
|---|--|
| Data | Building footprints (Esri Shapefile) |
| Overall Goal | Provide email-driven notifications for updates |
| Demonstrates | Email subscriptions |
| Start Workspace | RealTime-Ex5-Begin.fmw |
| End Workspace | RealTime-Ex5-Complete.fmw |

After configuring FME Server to process building footprints updates with both the Directory Watch and Email Publications, your supervisor is wondering if they can receive an email whenever the corporate database is updated.

Using an external email server, you think that it is possible to configure another Notification in FME Server to satisfy this requirement.

Miss Vector says...

This exercise continues where Exercise 4 left off. You must have completed Exercise 4 to carry out this exercise.

Access to an SMTP Email Server is required for sending email in this exercise. Gmail, Outlook, and Yahoo! are examples acceptable web-based solutions if you do not have access to an internal email server.

1) Add Subscription

Open the FME Server web interface and navigate to the Notifications page. Click the Subscriptions tab and then click New to create a new Subscription. This will be an email service through which a response will be sent.

Give the subscription a name like *Send Building Update Email* and create a new topic for it such as *BuildingUpdateEmail* (it's important to use a different topic than from the previous exercises).

Set the protocol to Email and set up your SMTP email server parameters.

In case it is of use, the server information for Gmail is as follows:

| | |
|----------------------------|----------------|
| SMTP Server Host | smtp.gmail.com |
| Server Port | 465 |
| Connection Security | SSL/TLS |

Regardless of the email provider, you should set these parameters as follows:

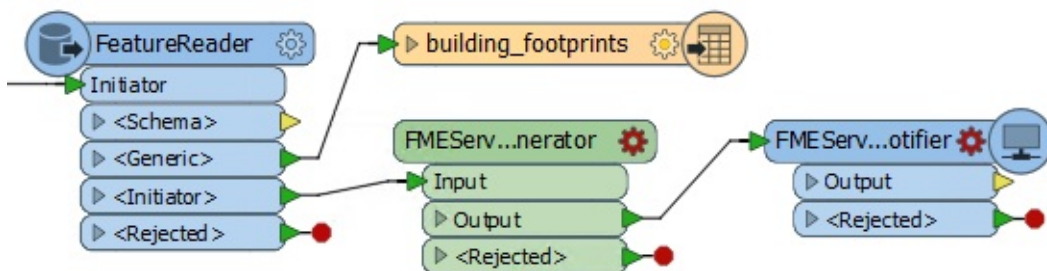
| | |
|----------------------|--|
| Email From | Your account name (for example fmeshapeprocessing@gmail.com) |
| Email Subject | Building Footprints Database Updated |

Most of the general settings (Email To, Email Template, etc.) will be set by the content we are going to provide, so once the above is set, click OK to save the Subscription.

2) Edit Workspace

Open the workspace from Exercise 4 (or the Start Workspace listed above).

Add two new transformers - the [FMEServerEmailGenerator](#) (a custom transformer) and an FMEServerNotifier - as a separate stream of data, connected to the Output Port of the FeatureReader:



NB: It's important to connect these two transformers to the <Initiator> port of the FeatureReader, where only one feature will emerge. If you connect them to the <Generic> output port then you will get an email for every feature in the Shapefile dataset!

3) Edit FMEServerEmailGenerator

Inspect the parameters for the FMEServerEmailGenerator. This transformer can be used to override the configurations in the Email Subscription created in Step 1.

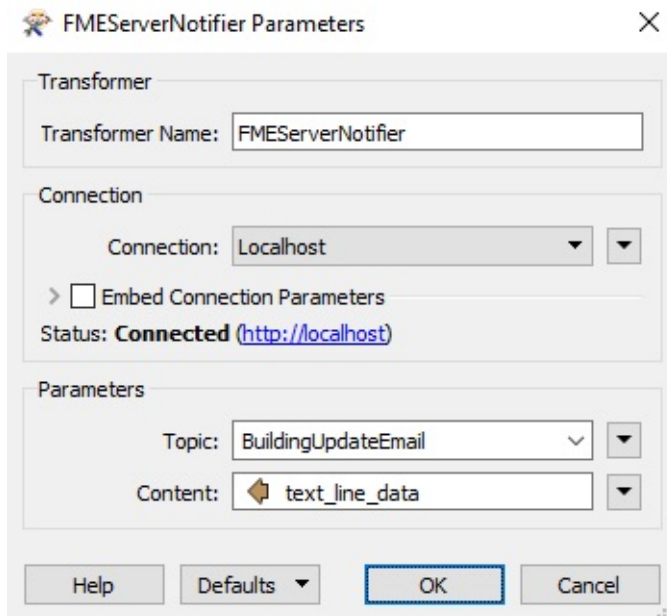
Each field can also accept attributes allowing the email to be dynamically configured. For our purposes in the training course, set the following parameters manually:

| | |
|----------------|--|
| To | (An email you have access to check) |
| Subject | Building Footprints Database Updated |
| Content | The Building Footprints database has been updated! |

4) Edit FMEServerNotifier

Now edit the parameters for the FMEServerNotifier transformer.

Set FME Server Connection parameters, pick the Topic created earlier (BuildingUpdateEmail), and for the Content select the attribute *text_line_data* (this attribute is created by the FMEServerEmailGenerator):



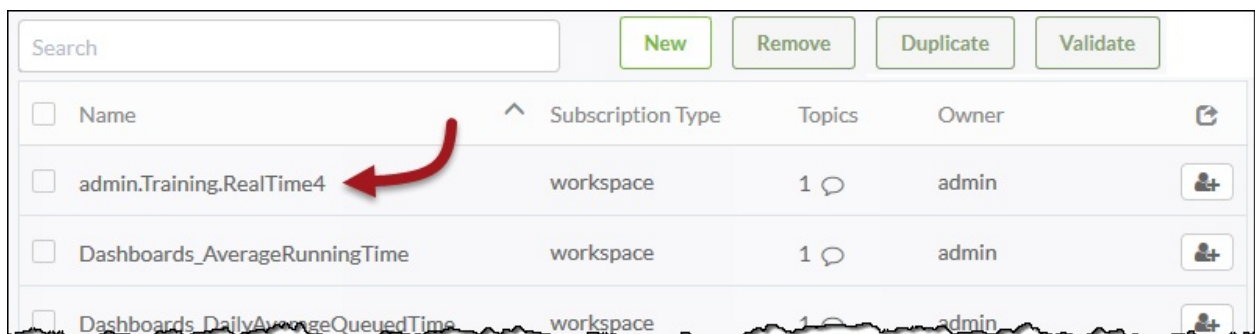
The screenshot shows the 'FMEServerNotifier Parameters' dialog box. It has three main sections: 'Transformer', 'Connection', and 'Parameters'. In the 'Transformer' section, the 'Transformer Name' is 'FMEServerNotifier'. In the 'Connection' section, the 'Connection' is 'Localhost', and the status is 'Connected (http://localhost)'. In the 'Parameters' section, the 'Topic' is 'BuildingUpdateEmail' and the 'Content' is 'text_line_data'. At the bottom, there are buttons for 'Help', 'Defaults', 'OK', and 'Cancel'.

5) Publish Workspace

Save and publish the workspace.

If the workspace name is different to that used in the Exercise 4 workspace, an update will need to be made as follows.

Navigate to the FME Workspace Subscriptions page. Notice that a Subscription will have been automatically created when registering the workspace with the Notification Service in the previous exercise. For example, if the workspace was called RealTime4, the Subscription name will be something like admin.Training.RealTime4:



The screenshot shows the 'FME Workspace Subscriptions' page. It has a search bar and buttons for 'New', 'Remove', 'Duplicate', and 'Validate'. Below is a table with columns: Name, Subscription Type, Topics, Owner, and an icon column. A red arrow points to the 'Name' column header.

| Name | Subscription Type | Topics | Owner | |
|-----------------------------------|-------------------|--------|-------|--|
| admin.Training.RealTime4 | workspace | 1 | admin | |
| Dashboards_AverageRunningTime | workspace | 1 | admin | |
| Dashboards_DailyAverageQueuedTime | workspace | 1 | admin | |

Click on this notification to change its parameters, and set/ensure that the Workspace parameter is pointing to the workspace just published.

6) Test Workspace

Test the workspace by sending an email to the Publication email address. Be sure to attach a zip file of the Shapefile datasets (.dbf, .prj, .shp, .shx) from C:\FMEDData2017\Data\Engineering\BuildingFootprints to the email.

If the workflow was successful, you should receive an email back with a response!

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Set up an outgoing Email Subscription*
- *Trigger an Email Subscription through the FMEServerNotifier transformer*

Message Streaming

Message Streaming is a real-time technique like notifications. However, where notifications are one-off messages, Message Streaming involves a continuous flow of information.

For our purposes, “continuous” means messages arrive at the FME Server at a faster rate than the notification service could handle; say more than one message per second.

Why Use a Message Stream?

When a workspace is involved as a subscriber, processing messages, the notification service operates by starting and running the workspace on demand, in response to an incoming publication.

However, problems occur when the average message interval is less than the time taken for the notification service to start and run a workspace. A message interval of one second or less is the threshold at which the notification service starts to struggle.

In a message stream the workspace used for message processing is constantly running and doesn't need to be started each time. Because of this reduced overhead it can process data at a much faster rate than the notification service.

When used in this way we call it High Capacity Message Streaming, as thousands of messages can be processed every second.

Elements of a Message Stream

Like Notifications, Message Streams can be either into or out of FME, or both:



However, rather than use the Notification service with readers and writers, Message Streams are handled by transformers. A "receiver" transformer acts as a subscriber, listening to a message stream and responding when one is received. A "sender" transformer acts as a publisher, creating a message and sending it to an open message stream.

Transformers and Protocols

A number of transformers can handle message streams, each of which is tied to a specific protocol.

| Transformer | | Protocol |
|-------------------|-----------------|-------------------------------------|
| Receiver | Sender | |
| JMSReceiver | JMSSender | Java Messaging Service (JMS) |
| WebSocketReceiver | WebSocketSender | HTML5 WebSockets |
| TCPIPReceiver | TCPIPSender | Transmission Control Protocol (TCP) |
| SQSReceiver | SQSSender | Amazon Simple Queue Service (SQS) |
| KinesisReceiver | KinesisSender | Amazon Kinesis Streams |
| TweetStreamer | Tweeter | Twitter API |
| PythonCreator | PythonCaller | Others |

Receiver Transformers

All receiver transformers in this list are designed to listen continuously to a message source and emit features only when a message arrives. Even then, the transformers will go on listening and awaiting more messages. Therefore a workspace containing any of these transformers will run continuously and not need to be stopped or started for each message.

Sender Transformers

All sender transformers in this list will emit a message for each feature that enters. They don't, by themselves, keep a workspace running continuously and will shutdown when incoming data is exhausted. However they will keep their connection open if a receiver is still running, therefore a continuous flow of outgoing messages requires both a Receiver *and* Sender.

Python Transformers

The PythonCreator and PythonCaller aren't specifically designed to connect to particular message protocols, but they can be made to do so and are particularly useful for protocols not otherwise supported in FME.

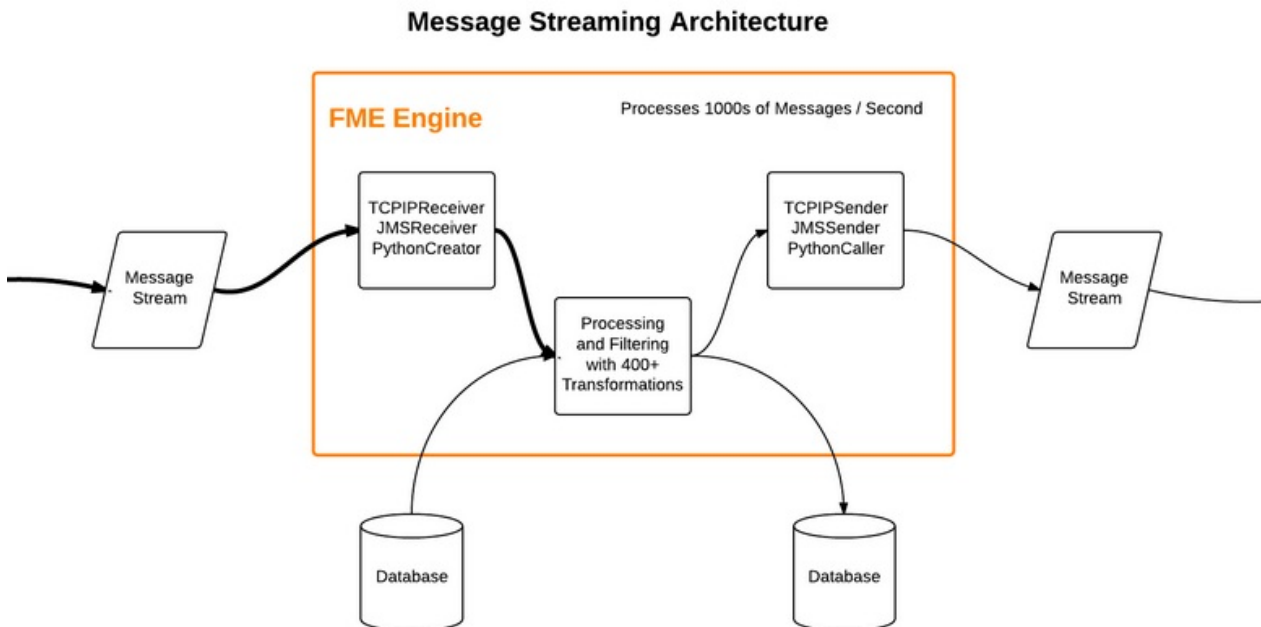
Miss Vector says ...

*Which of the Receiver transformers has a parameter to stop it running continuously?
Select all that apply.*

1. *SQSReceiver*
2. *WebSocketReceiver*
3. *JMSReceiver*
4. *TCPIPReceiver*

Message Streaming Architecture

A Message Streaming architecture that both receives and sends messages looks like this:



- A stream of messages is read into the workspace via one of the available transformers, for example the JMSReceiver
- Each message is processed by any of the available FME transformers, according to the needs of the project
- A stream of messages is sent out of the workspace via one of the available transformers, for example the TCPIPSender

Although the diagram shows a continuous process, it is not necessary for all of these components to be used in a setup. If the system is required to only receive messages, then only a Receiver transformer is needed. Likewise, if the system is intended to only send messages, then only a Sender transformer is required.

If both receiving and sending messages is required, then all components are necessary. However it's still possible to split those actions up across several workspaces.

Databases

The Database components in this diagram are optional, but are very useful. Messages will usually need to be processed against some other datasets (for example overlaid against a geofence) and a database is the quickest solution for reading and writing data.

Data read from a database is intended to be used to process the incoming message. For example perhaps the message represents a point feature (maybe a vehicle location) that is used to filter against database data (maybe traffic conditions).

Data written to the database is usually to record a stream of message information. For example, perhaps each incoming message represents a point feature (a lightning strike) that needs to be written to a database for a historic record.

Miss Vector says ...

Writing to a database in a High Capacity Message Streaming setup requires that the transaction interval is set to what value?

1. <Not Set>
2. Zero (0)
3. One (1)
4. Infinity (∞)

| Exercise 6 | Message Streaming: Handling Emergency Phone Call Streams |
|------------------------|---|
| Data | Event Messages (JSON) |
| Overall Goal | Create a workspace to read, parse, and filter WebSocket messages; keeping only events that affect transit stations. |
| Demonstrates | Creating a workspace to handle a message stream |
| Start Workspace | C:\FMEDData2017\Workspaces\ServerAuthoring\DataStream-Ex1-Begin.fmw |
| End Workspace | C:\FMEDData2017\Workspaces\ServerAuthoring\DataStream-Ex1-Generate-Complete.fmw C:\FMEDData2017\Workspaces\ServerAuthoring\DataStream-Ex1-Process-Complete.fmw |

As a technical analyst in the GIS department you deal with spatial data. Sometimes you need to process that data in real-time and sometimes that data can arrive in great quantities and at great speed.

In one such case, the city has been given access to the monitoring systems of emergency services. That means the ability to access in real-time information about all emergency calls.

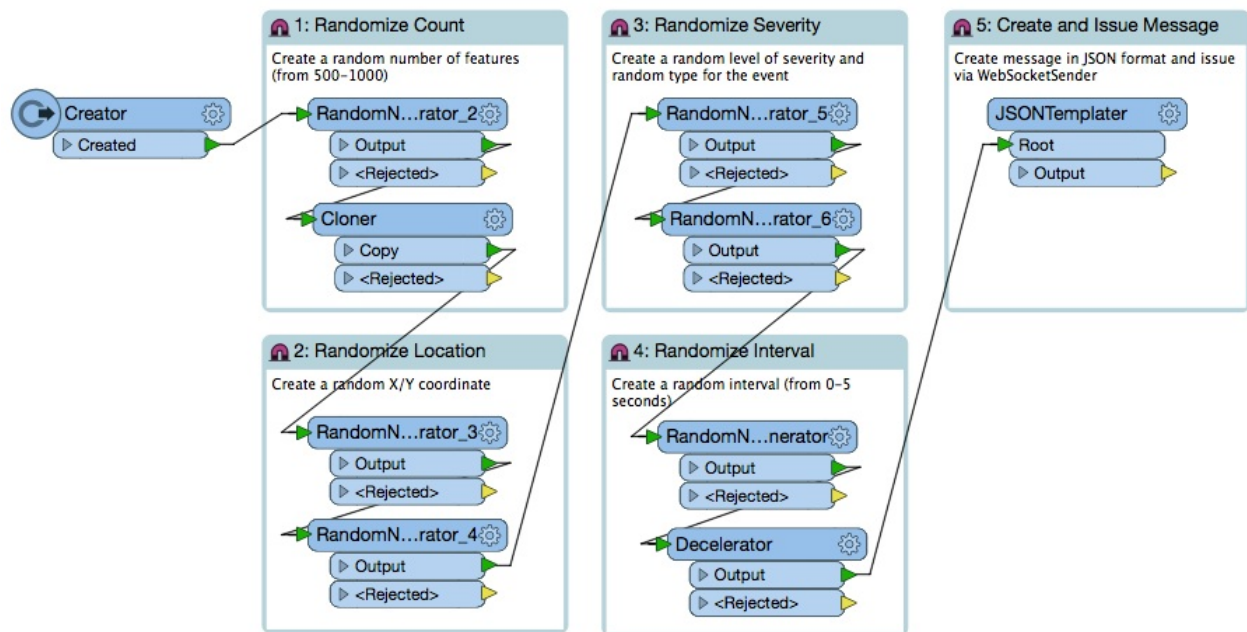
By emergency calls we mean the equivalent of 911 calls in North America, 999 in the UK, 112 in most of Europe, and 000 in Australia.

Of course, these calls can arrive at a tremendous rate, and at unknown intervals. If the city wishes to respond to any of these, and even if they wish to just record a history of the calls, you must implement a message streaming setup in FME Server.

1) Open Workspace

Unfortunately (I'm talking from a training point of view) we don't have access to a real-time stream of emergency phone calls, so we will have to generate our own.

Open the workspace C:\FMEDData2017\Workspaces\ServerAuthoring\DataStream-Ex1-Begin.fmw



Notice that the workspace generates a stream of events. A random number of events are generated, at random times, and at random locations. Additionally random severity and event type attributes are generated.

Each event is wrapped up into a JSON format message. All that we need to do is push that message out as a stream.

Miss Vector says...

This workspace is just generating "events". Those events could be lightning strikes, vehicle locations, traffic accidents, or even UFO sightings! For this exercise, we'll pretend they are emergency phone calls. In real life you would be connecting to an existing stream of data, and wouldn't need to generate one in this way.

2) Add WebSocketSender Transformer

Add a `WebSocketSender` transformer after the `JSONTemplater`. Inspect the parameters and set them as follows:

| | |
|-------------------------|---|
| WebSocket Server URL | ws://localhost:7078 |
| Verify SSL Certificates | No |
| Connection Preamble | <pre>{ ws_op: "open", ws_stream_id: "EmergencyEvents" }</pre> |
| Data To Transmit | <pre>{ ws_op: 'send', ws_msg: '@Value(EventMessage)' }</pre> |

As you can see, these parameters open a WebSocket connection (to an EmergencyEvents stream) and send information (the EventMessage attribute). Save the parameters and then save the workspace.

3) Create Workspace

Now we have the ability to generate a stream of data we will create the workspace that is to process the data. Start Workbench and begin with a blank canvas (don't close the stream generator workspace, as we'll need that as well in a moment).

In the blank canvas add a Creator transformer and follow it with a WebSocketReceiver. Inspect the WebSocketReceiver transformer parameters and set them as follows:

| | |
|-------------------------|---|
| WebSocket Server URL | ws://localhost:7078 |
| Verify SSL Certificates | No |
| Connection Preamble | <pre>{ ws_op: "open", ws_stream_id: "EmergencyEvents" }</pre> |
| Output Attribute | IncomingMessage |

Save the changes and add a Logger transformer after the WebSocketReceiver.

4) Publish Workspaces


Let's test what we have by publishing the workspaces and running them on FME Server.


Publish each workspace in turn. In both cases simply register it with the Job Submitter service. There are no datasets or other parameters we need worry about.

5) Run Workspace

Log in to the FME Server web interface, locate the data stream generator workspace, and run it. The dialog in response will look like this:

Run Workspace

 **DataStream-Ex1-Generate-Complete.fmw**

 **RUNNING**

Your Job has been submitted and will continue to run if you leave this page.

Cancel Job

View Details

The workspace will run for a long time and we can leave it to do so. Leave this page by clicking the Run Workspace button on the main menu and - within the Run Workspace page - locate the processing workspace. Now run that.

Again the response will report that the workspace is running, and will continue to do so.

6) Check Jobs and Cancel

Navigate to the Jobs page and click the tab labelled Running. You will see the two jobs:



Running

Show Jobs For:

All Users

Cancel

Cancel All

| <input type="checkbox"/> | Id | Workspace | Repository | Username | Status | Engine | Started | Priority |
|--------------------------|----|--------------------------------------|------------|----------|---|---------------------|-------------------|----------|
| <input type="checkbox"/> | 10 | DataStream-Ex1-Process.fmw | Training | admin |  | AP-FME64BIT_Engine1 | Today at 12:05:36 | 100 |
| <input type="checkbox"/> | 9 | DataStream-Ex1-Generate-Complete.fmw | Training | admin |  | AP-FME64BIT_Engine2 | Today at 12:05:33 | 100 |

|< < 1 / 1 > >| 100

Displaying 1 - 2 of 2

Let the jobs run for a minute or two. Then choose each of them and click the Cancel button to cancel them:

Running

Show Jobs For:

All Users

Cancel Cancel All

| <input type="checkbox"/> | Id | Workspace | Repository | Username | Status | Engine | Started | Priority |
|-------------------------------------|----|--------------------------------------|------------|----------|--------|---------------------|-------------------|----------|
| <input checked="" type="checkbox"/> | 10 | DataStream-Ex1-Process.fmw | Training | admin | | AP-FME64BIT_Engine1 | Today at 12:05:36 | 100 |
| <input checked="" type="checkbox"/> | 9 | DataStream-Ex1-Generate-Complete.fmw | Training | admin | | AP-FME64BIT_Engine2 | Today at 12:05:33 | 100 |

1 / 1 100

Displaying 1 - 2 of 2

Once cancelled, go to the Completed jobs tab. You'll see the two cancelled jobs:

| <input type="checkbox"/> | Id | Workspace | Repository | Username | Status | Engine | Finished | Started | Priority |
|--------------------------|----|--------------------------------------|------------|----------|--------|---------------------|-------------------|-------------------|----------|
| <input type="checkbox"/> | 9 | DataStream-Ex1-Generate-Complete.fmw | Training | admin | | AP-FME64BIT_Engine2 | Today at 12:07:25 | Today at 12:05:33 | 100 |
| <input type="checkbox"/> | 10 | DataStream-Ex1-Process.fmw | Training | admin | | AP-FME64BIT_Engine1 | Today at 12:07:25 | Today at 12:05:36 | 100 |

Click on the processing workspace job and check the log. You should see messages in the log like this:

```

|=====
=====
INFORM|WebSocketReceiver_Output: Feature is:
INFORM|+++++
+++++
INFORM|Feature Type: `WebSocketReceiver_Output_LOGGED'
INFORM|Attribute(encoded: utf-8): `IncomingMessage' has value `{
"EventID" : 6....
INFORM|Attribute(string)      : `fme_geometry' has value
`fme_undefined'
INFORM|Attribute(encoded: utf-8): `fme_type' has value
`fme_no_geom'
INFORM|Geometry Type: Unknown (0)

```

This proves that the WebSocketReceiver is acting as expected and receiving messages from the message stream.

Miss Vector says...

You've proved that you can create a workspace to process a message stream, which is the important part of this exercise. But if you have the time, let's see what improvements we can add to make the result more realistic.

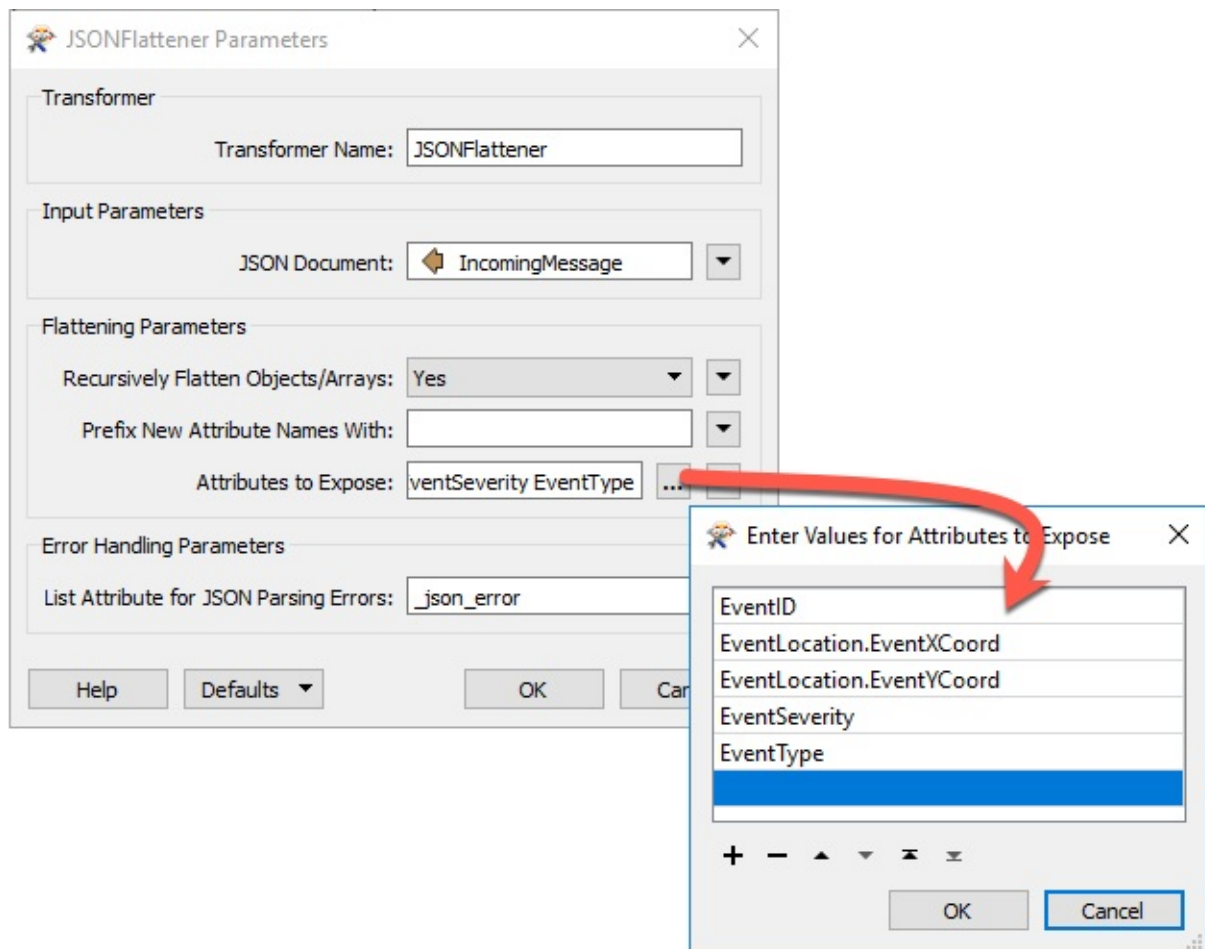
7) Add JSONFlattener

The first thing to do with incoming messages is to extract information as attributes. Because the incoming data is JSON format, add a JSONFlattener transformer to the processing workspace, after the WebSocketReceiver.

Inspect the JSONFlattener's parameters and set the attribute IncomingMessage as the JSON Document to process.

Under Attributes to Expose manually enter:

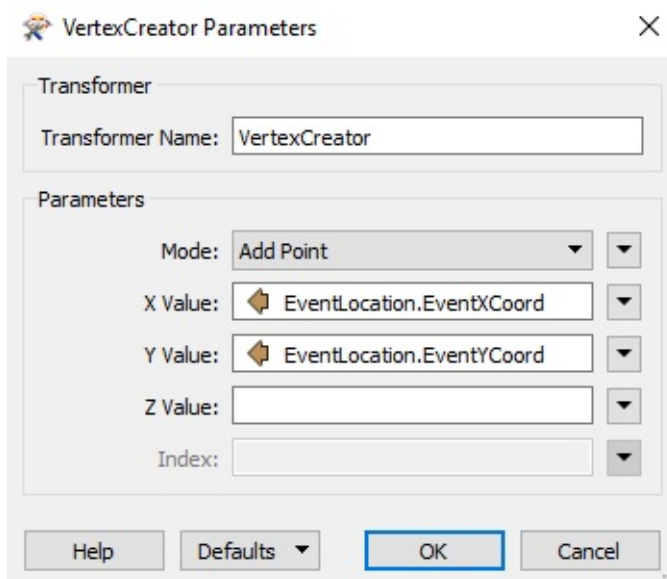
- EventID
- EventLocation.EventXCoord
- EventLocation.EventYCoord
- EventSeverity
- EventType



You will now have the information from the message available as a set of attributes in the workspace.

8) Add VertexCreator

Now add a VertexCreator transformer. Set it up to use the X/Y attributes to create a true point feature:



With this we now have a true geographic feature and can process it as required.

9) Add Reader

The public transportation team within the city has learned you are working with this emergency data. They wish to be alerted immediately if there is an emergency event within 200 metres of a transit station. Let's show them how easy it is to set this up.

Firstly we need the transit station data, so select Readers > Add Reader and add the following:

| | |
|-----------------------|--|
| Reader Format | Esri Geodatabase (File Geodb Open API) |
| Reader Dataset | C:\FMEDData2017\Data\CommunityMapping\CommunityMap.gdb |

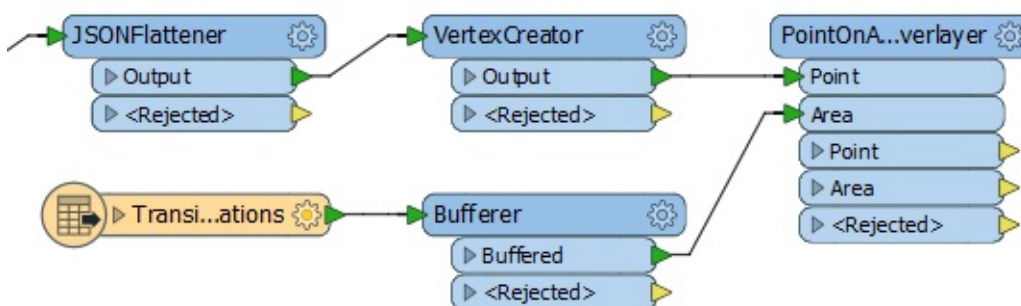
When prompted (or in the parameters dialog) ensure that only the TransitStations table is selected.

10) Filter Data

Now let's filter the emergencies.

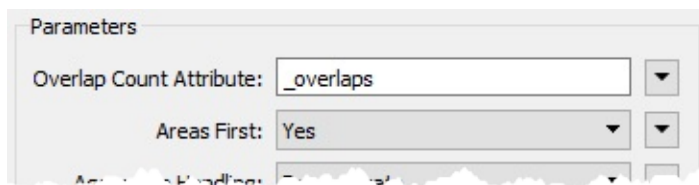
First, add a Bufferer transformer to the TransitStation feature type and buffer the features by 200 metres.

Secondly, add a PointOnAreaOverlayer to assess whether an emergency falls inside one of these buffers. The workspace will now look like this:



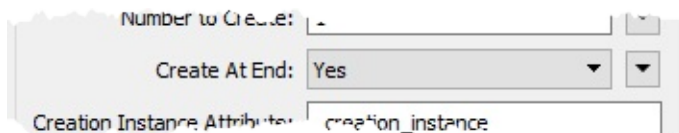
At the moment there is one big problem that stops this from working. The PointOnAreaOverlayer transformer is a Group-Based transformer, sometimes called a "blocker". It will hold on to features until it has finished being fed them, before outputting any data. In our case we want to make it Feature-Based; i.e. it will process each message at once.

So, inspect the PointOnAreaOverlayer parameters and set Areas First to Yes:



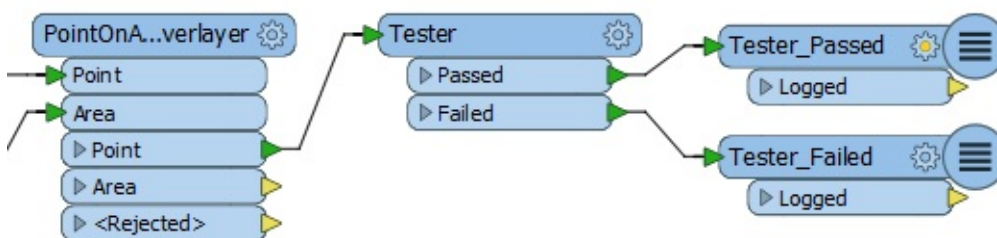
This tells the transformer that all area features (buffered stations) will be first to arrive, therefore any point features (message locations) can be processed immediately.

However, we have to ensure that the transit features will arrive first. Therefore inspect the transformer parameters for the Creator transformer and set Create at End to Yes:



Now, all being well, the transit features will arrive first at the PointOnAreaOverlayer transformer.

Finally, add a Tester transformer after the PointOnAreaOverlayer. Set up the test to check for `_overlaps > 0` (i.e. where the message location falls inside a transit station buffer). Connect some Logger transformers to the Tester output ports:



Note that, if there were other parameters (for example the transit team were only interested in Event Types 7, 8, 9, and 10) you could add them to this Tester as well.

11) Publish Workspaces

Now publish the two workspaces again (you may or may not have to upload the TransitStation Geodatabase along with the workspace) and run them using the same process as before, but leave it for a few minutes longer, as it may take a while for one of the random events to fall inside a transit station buffer.

Once stopped, check the logs and you should see that messages falling within 200 metres of a transit station are logged (with a different header).

Miss Vector says...

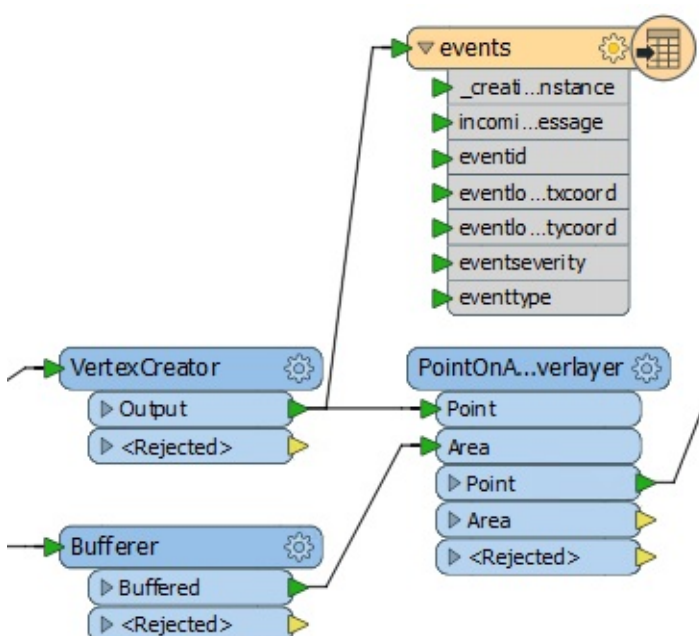
If you want to adjust the settings to get a result quicker, then go ahead. For example, you might set the buffer size to 500 metres instead of 200, or you might reduce the interval time on the message generator. Feel free to make whatever parameter changes you like to test the setup. You could even bypass the Decelerator transformer (in the data-stream creation workspace) to see how fast FME can deal with the incoming messages! But if you do that, be sure to start the processing workspace first, else the generator might finish by the time you do get the processor started!

12) Add Writer

The messages that are being received are not all being used by the transit team, but we should probably keep a record of them. So - back in FME Workbench - select Writers > Add Writer from the menubar. Use the following parameters to add a database Writer to the processing workspace:

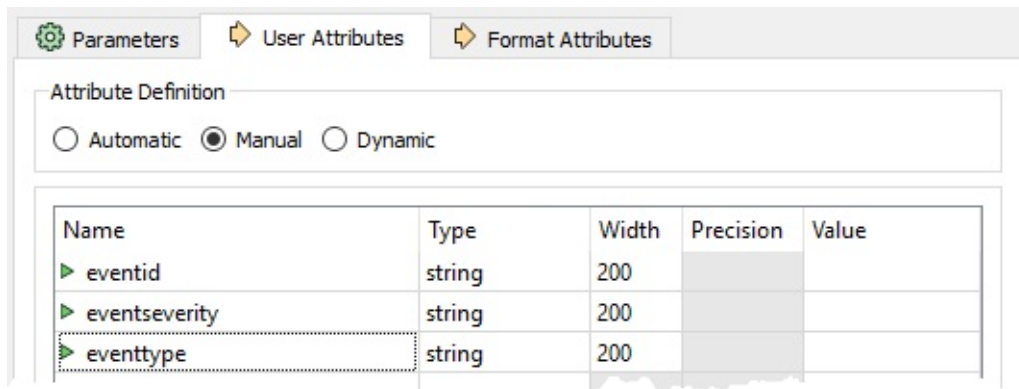
| | |
|----------------------------|---|
| Writer Format | Spatialite |
| Writer Dataset | C:\FMEData2017\Output\EventMessages.sl3 |
| Writer Parameters | Advanced : Features Per Transaction = 1 |
| Add Feature Type(s) | Table Definition: Automatic |

In the newly added feature type, change the name to *events* and close the dialog. Connect the feature type to the VertexCreator output port (i.e. we're recording all events, not just the filtered ones):



The attributes are added automatically, but include a few we don't need. So open up the properties dialog again for the feature type and click the User Attributes tab. Change it from Automatic to Manual and delete the attributes:

- `_creation_instance`
- `incomingmessage`
- `eventlocation_eventxcoord`
- `eventlocation_eventycoord`



Notice that the attributes were automatically renamed (to lower case and removing disallowed characters) to match Spatialite requirements.

If you publish and run the workspace (you may need to set the Spatialite database output to be written to a Resources folder) now you should be able to see - while the workspace is still running - the results being added to the database. You can inspect the file in the FME Data Inspector to prove this.

13) Create Notification

One last task (I promise). The filtered messages are important to the transit team, but at the moment they are going nowhere. We should set up a way to inform them.

We could add another messaging transformer, such as the WebSocketSender, JMSSender, SQSSender, or even a Tweeter. That would make the processing workspace a "pure" messaging workspace.

On the other hand, the outgoing messages are nothing like the same rate as the incoming messages. With the parameters as described in this exercise, there is only a transit message once every minute. So, we can create a "hybrid" solution by setting output messages to be sent via the FME Server Notification Service.

Go to the FME Server web interface and navigate to the Notifications page.

Create a new Topic called EmergencyTransitMessages:

Publications Subscriptions Topics

Search

NewRemoveTest TopicMonitor

| <input type="checkbox"/> Name | ^ Owner | |
|---|---------|--|
| <input type="checkbox"/> DATADOWNLOAD_ASYNC_JOB_FAILURE | admin | |
| <input type="checkbox"/> DATADOWNLOAD_ASYNC_JOB_SUCCESS | admin | |
| <input type="checkbox"/> EmergencyTransitMessages | admin | |
| <input type="checkbox"/> ... | ... | |

Now create a new notification Subscription tied to that topic. There are various protocols we could realistically use for sending a message (email springs to mind) but for the purposes of this exercise use the Logger protocol. Set the Log Level parameter to High:

New Subscription

Validate

Name

EmergencyTransitMessages

Protocol

Logger

Topics Subscribed To

* EmergencyTransitMessages

+

Protocol Settings

Log Level ⓘ

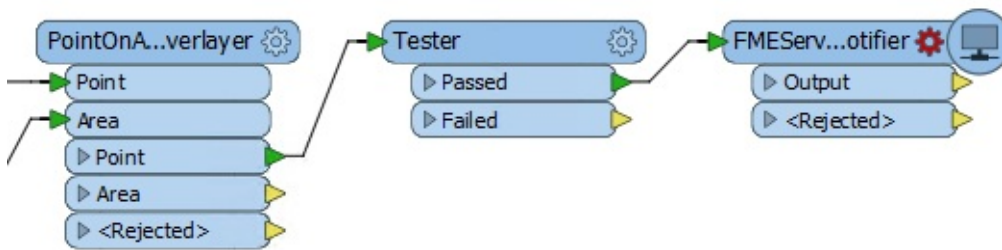
High

Cancel

OK

14) Add FMEServerNotifier Transformer

Back in the processing workspace in Workbench, remove any Logger transformers at the end of the workspace. Add an FMEServerNotifier transformer connected to the Tester:Passed port:

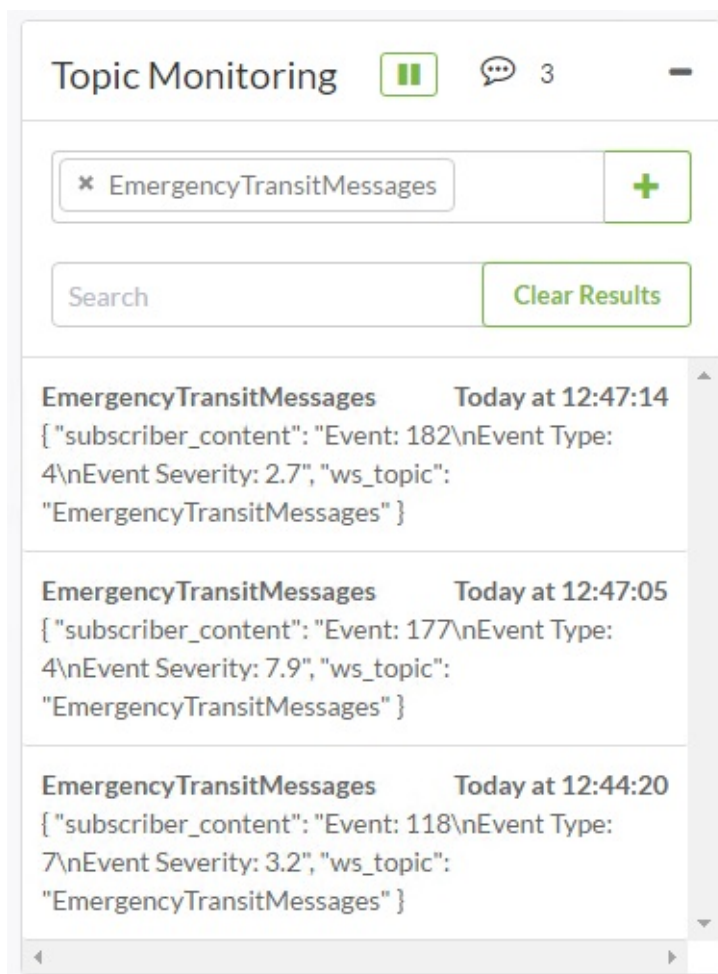


Inspect the transformer parameters and set it up to send a message to the EmergencyTransitMessages topic. Set the message content to be whatever you like. You could use the text editor dialog to create something out of the available attributes (it can be plain text, it doesn't have to be JSON or XML).

15) Publish and Run Workspaces

Re-publish and set the workspaces running again. Navigate to the Notifications page and click on the Topics tab. Enable Topic Monitoring to watch the EmergencyTransitMessages topic for incoming notifications.

In a short while you will start to see emergency messages like this:



Visit Resources > Logs > core > current > subscribers > logger.log to find the results as recorded by the Logger protocol notification.

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Send and receive messages via WebSockets*
- *Publish and run message-streaming workspaces*
- *Cancel message-streaming workspaces and check their log files*
- *Extract attributes from JSON messages*
- *Use transformers to transform and filter a message according to its content*
- *Set up workspaces to handle group-based transformers in a real-time scenario*
- *Record incoming messages into a database*
- *Set up a hybrid system with message streaming **and** notifications*

Troubleshooting for Administrators

This section shows a few basic troubleshooting techniques in case of emergency.

Directory Watch Publication is not Triggered by Adding Files

If you are having troubles triggering the Directory Watch Publication when new files are added then the following may be of help:

- Ensure that the account running the FME Server Windows Services has permissions to access the directory.
 - CREATE notifications are only sent when there is a change in *file size* - this might be an issue if you are overwriting the file, or deleting and re-adding within the set Poll Interval.
-

FME Server Fails to Receive Email

If you are unable to receive email on FME Server then the following suggestions may be of help:

- Ensure you carry out the post-installation configuration steps as noted in the reference manual. Without this only local mail could be delivered.
 - Is the FME Server's email SMTP port open? Typically this will be port 25 or 465.
 - Does your FME Server have a DNS name that is available on an internet name server or your local DNS setup? Email cannot be delivered to it if it cannot be found!
 - For IMAP, is the FME Server's IMAP port open? Typically this will be port 993.
-

FME Server Fails to Send Email

If you are unable to send email on FME Server then the following suggestions may be of help:

- Have you created and properly configured the Email Subscriber? The Data Download and Job Submitter services have pre-defined Subscribers, but they will need to be properly configured.
 - Is the FME Server's email SMTP port open? Typically this will be port 25 or 465.
-

- Does the FME Server have an internet connection?
 - Are you using a Google account with 2-Step Verification? You will need to generate an [app password](#), and replace the SMTP password with this 16 digit app password code.
-

Email Uses Defaults

If email sent by FME Server uses default parameters instead of those you had configured, then the following suggestions may be of help:

- If you are sending an email using the FMEServerNotifier transformer ensure that the Content parameter is set to an attribute that contains the email message information in JSON format (use the custom transformer FMEServerEmailGenerator).
 - If you are sending an email from a completed workspace using the Notification Service and Topics to Notify (Success) ensure the Notification Writer is set to the Text File writer which is writing the email information.
-

Cannot Connect to WebSockets Server

If you cannot connect to a WebSockets server then the following may be of help:

- Ensure the FME Server's WebSockets port (default 7078) is open.
 - Ensure you are using the correct stream_id for sending and receiving between your applications.
-

FME Workspace Subscription is not Triggered by Topic

If a workspace set to subscribe to a Topic is not triggered by it, then the following suggestions may be of help:

- Check the FME Workspace Subscription in the web interface and make sure the required Topic is in the list of Topics subscribed to.
 - Check the FME Workspace Subscription in the web interface and confirm the username and password for the Subscription. If you have changed your password recently the password for the Subscription needs to be changed as well.
-

Topic Monitoring Fails

If Topic Monitoring fails then the following may be of help:

- Topic Monitoring requires the use of WebSockets, so ensure the FME Server's WebSockets port (default 7078) is open.

Module Review

This module introduced you to FME Server's Notification Services.

What You Should Have Learned from this Module

The following are key points to be learned from this module.

Theory

- Real-time systems are message driven and involve small amounts of information.
- The Notification Service handles individual messages. Topics are keywords that define a notification subject.
- Publications are FME Server objects that listen for and respond to notifications from clients
- Subscriptions are FME Server objects that push notifications to clients
- Incoming Notification messages are received from Publishers, outgoing messages are sent to Subscribers
- Protocols are the method of communication (for example, email, FTP, Amazon S3, etc.)
- Message Streams are set up using transformers and involve a workspace that runs continuously receiving/sending messages

FME Skills

- The ability to create Topics, Publications, and Subscriptions
- The ability to use the Directory Watch protocol for Publishing
- The ability to use the Email protocol for Publishing and Subscribing
- The ability to use workspaces to process incoming notification messages
- The ability to create a workspace for handling a message stream

Further Reading

For further reading why not check out...

- [This blog article on the FME Server notification services](#) being used to create reports in response to earthquake alerts.

- [This blog article on JMS message streaming](#) at 125,000 messages per hour!

Questions

Here are the answers to the questions in this chapter.

Miss Vector says...

All notification setups must have which of these:

- 1. Incoming components (Publisher, Publication) AND outgoing components (Subscription, Subscriber)*
- 2. Incoming components OR outgoing components OR both**
- 3. Incoming components OR outgoing components but never both*
- 4. None of the above*

Although the diagram (under Elements of the Notification System) shows a continuous process, it is not necessary for all of these components to be used in a setup. If the system is designed for FME Server to only receive notifications, then only a publisher/publication is needed. Likewise, if the system is intended for FME Server to only send notifications, then only a subscription/subscriber is required. But if both receiving and sending notifications are required then all components apply.

Miss Vector says...

Please don't get this wrong! Publications and Topics have what relationship?

- 1. One:One (Each Publication has one Topic, each Topic belongs to one Publication)*
- 2. One:Many (Each Publication can have many Topics, each Topic belongs to one Publication)*
- 3. Many:One (Each Publication has one Topic, each Topic can belong to multiple Publications)*
- 4. Many:Many (Each Publication can have many Topics, each Topic can belong to multiple Publications)**

You got it wrong? I'm not angry... just terribly, terribly disappointed!

Miss Vector says...

Tell me, which one of these statements is correct:

- 1. SMTP and IMAP can both be used as either a Subscription and/or a Publication protocol*
- 2. SMTP can be used as both a Subscription and a Publication; IMAP can only be used for a Publication**
- 3. SMTP can only be used for a Publication; IMAP can only be used as both a Subscription and a Publication*
- 4. SMTP can only be used for a Subscription; IMAP can only be used for a Publication*

See the table under Notification Protocols for the full list of which protocol can be used for which type of notification.

Miss Vector says...

When a workspace is part of a notification system, processing incoming messages, it is a...

- 1. Subscription*
- 2. Publication*
- 3. Protocol*
- 4. Client**

It is a client! In this case it is a subscriber (it is subscribing to messages produced by a Publication). If it were sending messages it would be a publisher (publishing messages to a Subscription).

It's not a Publication or a Subscription because those are specific components of FME Server.

Miss Vector says...

I want my workspace to send me an email when it is run, so I know when people are using it to download data. When I publish it, what should I register it to?

1. The Notification Service
- 2. The Data Download Service**
3. The Email (SMTP) Protocol
4. The Workspace Subscriber Protocol

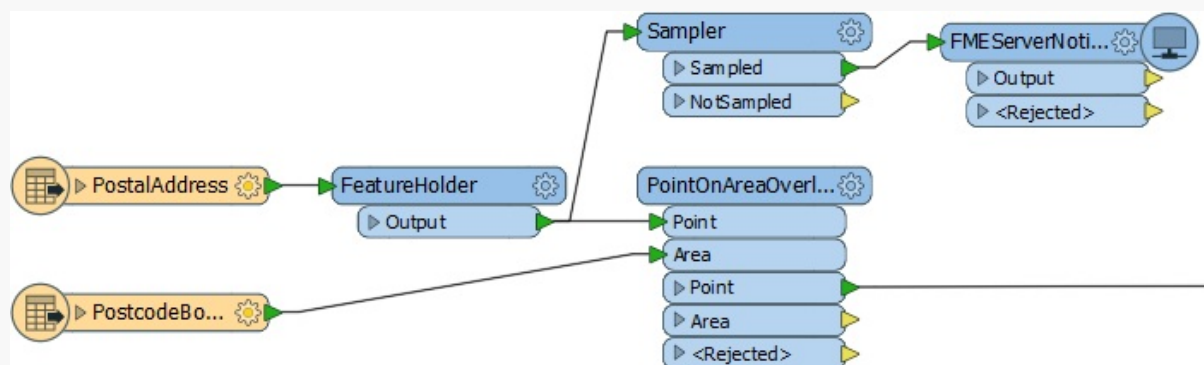
I'm setting it up for people to download data, so I register it as a Data Download service. It's as simple as that. To get a notification I just have to pick a topic to trigger in the Data Download settings. Of course, to get an email I must set up an Email Subscription connected to that topic - but that has nothing at all to do with how I register the workspace!

Miss Vector says...

I've got a workspace that reads 50,000 features, transforms them, and writes them out. If I want to send a single notification that the features have been read, which combination of transformers would be of most use?

1. Creator/FeatureWriter/FMEServerNotifier
2. Creator/FMEServerJobSubmitter
3. Creator/FeatureReader/FMEServerNotifier
- 4. FeatureHolder/Sampler/FMEServerNotifier**

I'm sending a notification (not running a job) so I use the FMEServerNotifier. I already have a reader so I don't need a transformer to read or write data. However I do need a Sampler transformer to reduce the number of features down to one; otherwise I'll send 50,000 notifications. The FeatureHolder ensures the notification is not triggered until all features have been read. It would look like this:



Miss Vector says...

Which of the Receiver transformers has a parameter to stop it running continuously? Select all that apply.

- 1. SQSReceiver**
2. WebSocketReceiver
3. JMSReceiver
- 4. TCPIPReceiver**

The SQSReceiver has the ability to switch to a number of messages to read, and the TCPIPReceiver has the option to close the connection once the publishing client disconnects.

Miss Vector says...

Writing to a database in a High Capacity Message Streaming setup requires that the transaction interval is set to what value?

1. <Not Set>
2. Zero (0)
- 3. One (1)**
4. Infinity (∞)

Setting the transaction interval to one means that each message is committed as it arrives. Any other value (in this list at least) would probably mean the data is never committed until the workspace was terminated.

FME Server Projects

A Project in FME is a way to group FME Server components together in a way that mimics a real-world project.



Grouping components together allows them to be handled together in a single action. For example, a workspace author can back up their work with a single action by grouping together the relevant parts as a project.

Project Components

A project can incorporate most FME Server components, including common ones such as:

- Workspaces
- Custom Transformers
- Repositories
- Schedules
- Notification Components
- Resources
- Web, Database, and Resource Connections
- Users

Project Uses and Advantages

Projects have a number of primary uses:

- Backups
- Migrations
- Sharing

Backups

The Backup tools on FME Server allow the entire FME Server to be backed up by an administrator.

However, a Project allows a workspace author to create backups (without being an administrator) and allows that author to pick which components are included in the backup (it does not need to be the entire server).

Migrations

In a similar way to backups, an administrator can transfer an entire FME Server onto a new server setup, but sometimes a workspace author needs to be able to transfer just their work to a different FME Server instance.

Projects allow an author to migrate their work between machines without the assistance of an administrator.

Sharing

The system administrator has access to security functions on FME Server to control who has permission to use various components and resources. New tools in FME2017 also allow authors to share repositories with other users.

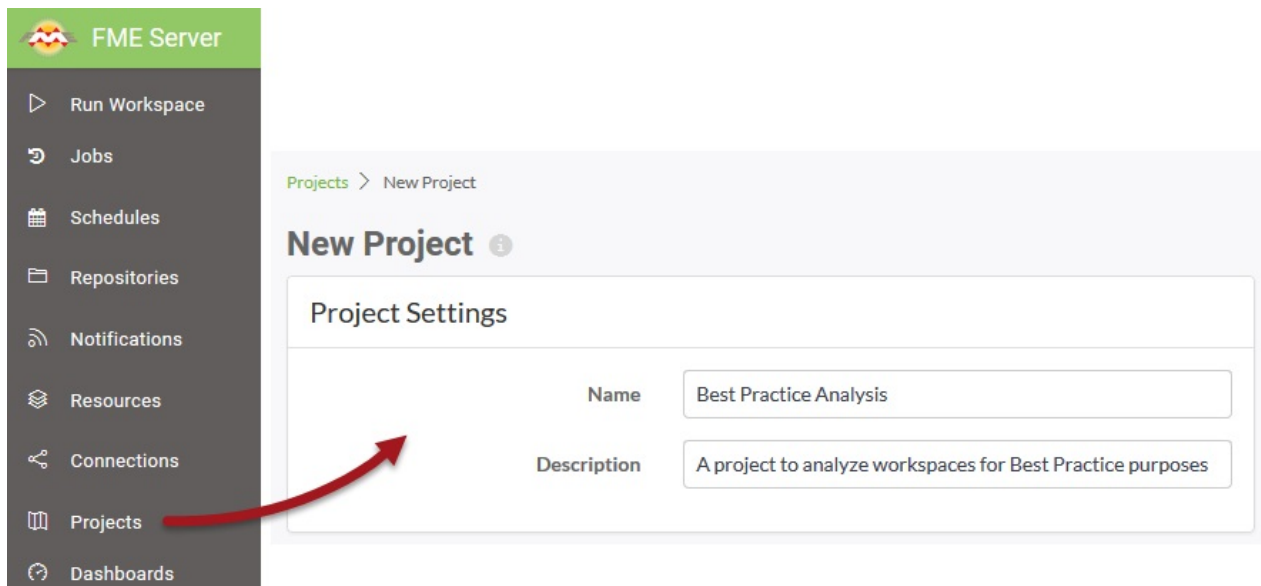
However, a project usually consists of more components than just workspaces, and an FME Project allows an author to share multiple components instead of just a single repository.

Creating a Project

Creating a project involves initiating the creation process and then simply adding the required components to that project.

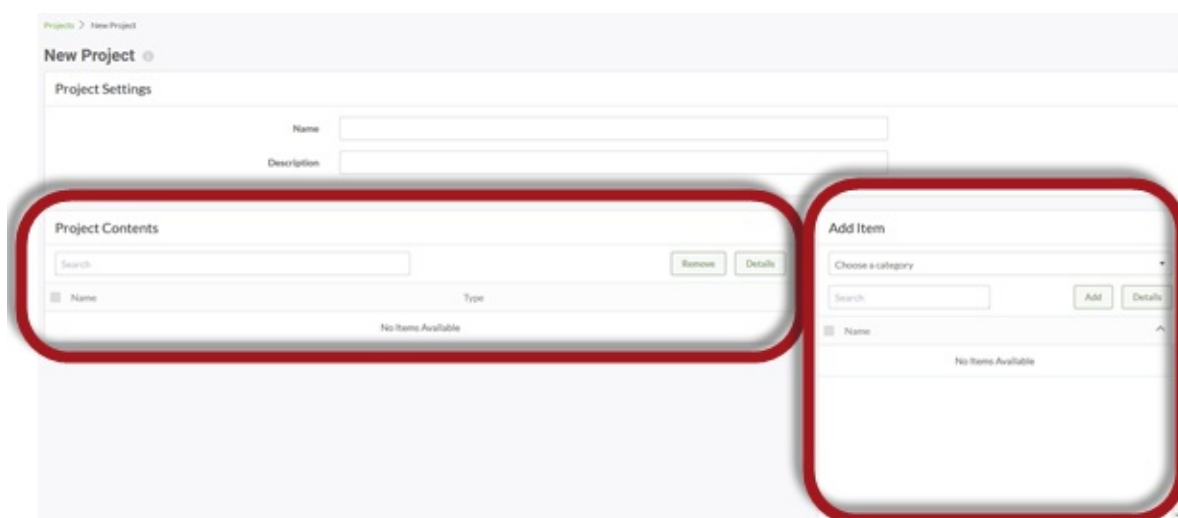
Create Project

Creating a project is carried out on the Projects page (accessed via the main menu) by clicking the New button and entering some basic project settings such as Name and Description:



The screenshot shows the FME Server interface. On the left is a dark sidebar menu with icons and labels: Run Workspace, Jobs, Schedules, Repositories, Notifications, Resources, Connections, Projects, and Dashboards. The 'Projects' item is highlighted with a red arrow pointing to the 'New Project' dialog. The dialog has a breadcrumb 'Projects > New Project' and a title 'New Project' with an information icon. Below the title is the 'Project Settings' section, which contains two input fields: 'Name' with the value 'Best Practice Analysis' and 'Description' with the value 'A project to analyze workspaces for Best Practice purposes'.

Below the Project Settings parameters are two dialogs:

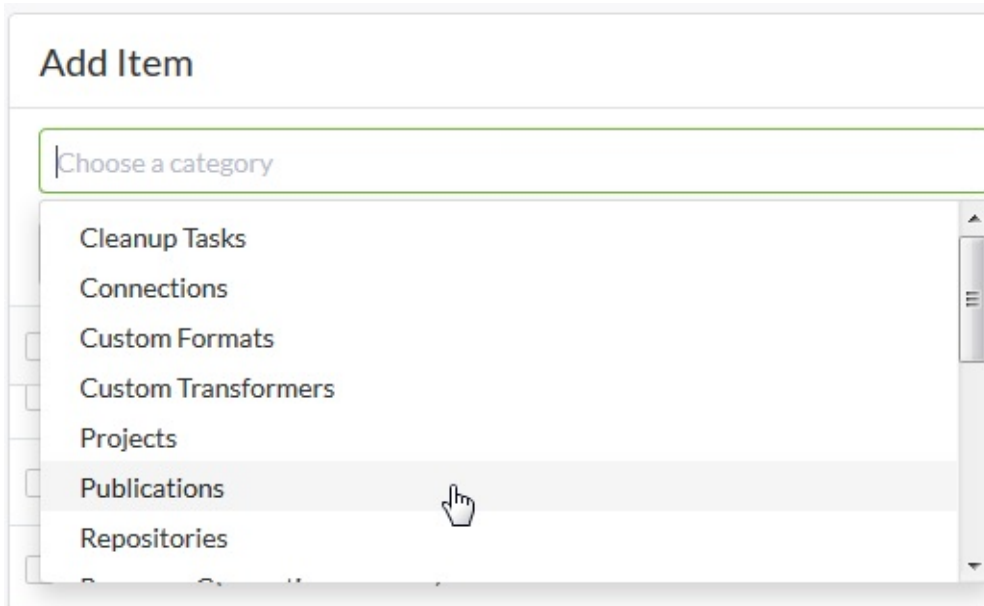


This screenshot shows the 'New Project' dialog with two sub-dialogs overlaid. The 'Project Contents' dialog on the left has a search bar, a table with columns 'Name' and 'Type', and a message 'No Items Available'. The 'Add Item' dialog on the right has a 'Choose a category' dropdown, a search bar, and buttons for 'Add' and 'Details', with a message 'No Items Available' at the bottom. Both sub-dialogs are outlined with red rounded rectangles.

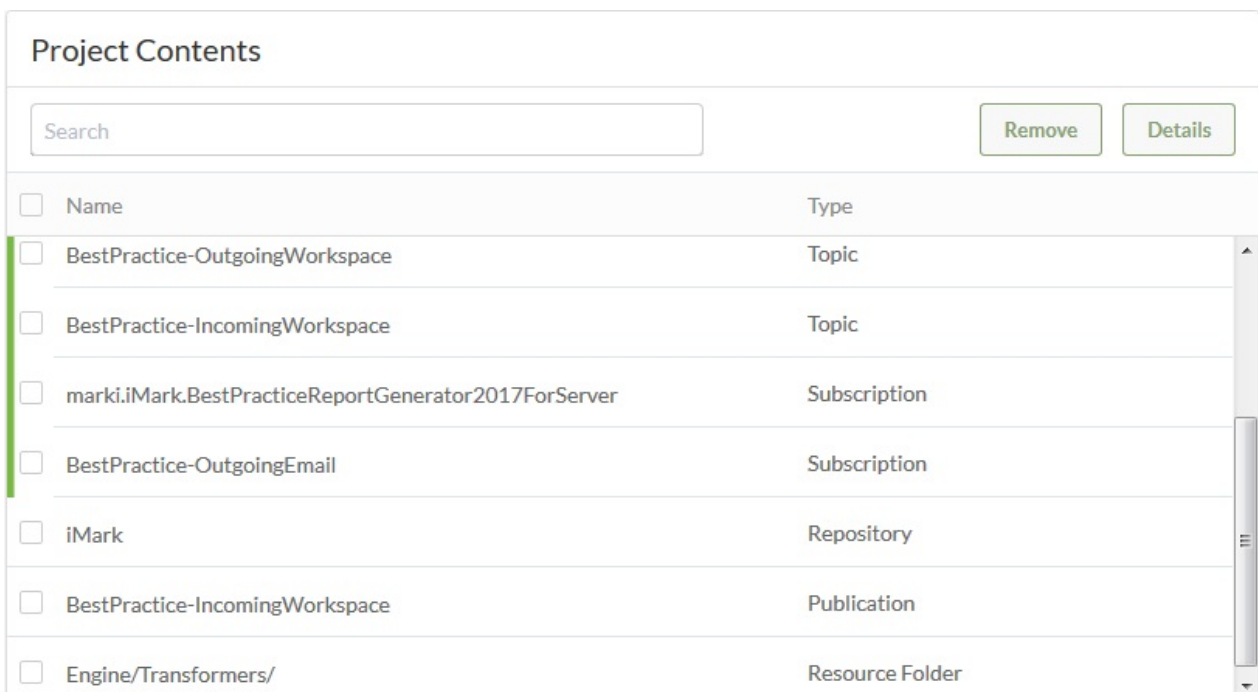
.1 UPDATE

For FME Server 2017.1 and newer, the "Add Item" dialog is not visible until the user selects the "Add" button (located in the same space).

The lower right-hand dialog allows components to be added to the project:



The lower left-hand dialog shows which components have been added to the project:



The above image shows a number of components added to an existing project. Once complete the project is added to the list on the Projects landing page:

Projects ⓘ

Search

NewDuplicateRemoveExportImport

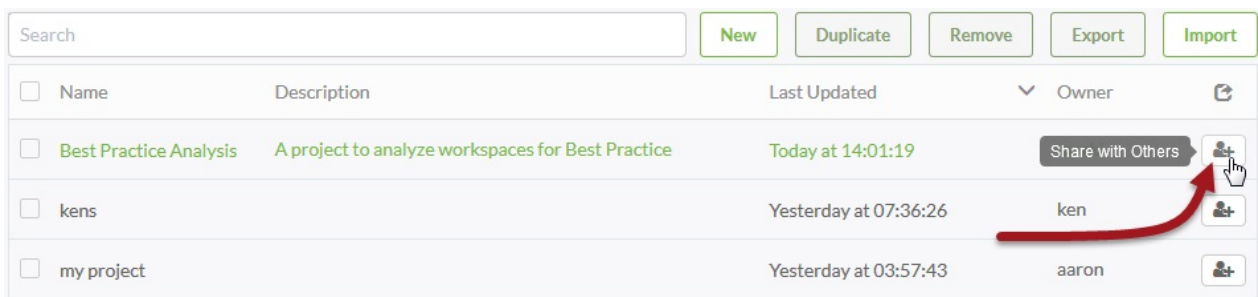
| <input type="checkbox"/> | Name | Description | Last Updated | ▼ | Owner | |
|--------------------------|------------------------|--|--------------------|---|-------|--|
| <input type="checkbox"/> | Best Practice Analysis | A project to analyze workspaces for Best Practice purposes | 2017-4-28 14:01:19 | | marki | |
| <input type="checkbox"/> | MyBerlinProject | | 2017-4-25 06:59:04 | | aaron | |
| <input type="checkbox"/> | TeamOSCAW_DirWatch | Directory Watch for Projects Demo | 2017-3-28 12:53:59 | | rylan | |
| <input type="checkbox"/> | KenFileArrival | Notify me when File Arrives | 2017-3-28 09:09:29 | | ken | |
| <input type="checkbox"/> | FMEWT2017_07_BatchData | All you need to bulk load tiled Vancouver contours into a Geopackage | 2017-1-30 17:31:04 | | dale | |

Sharing a Project

A project is often carried out by a group of FME users and authors, therefore it makes sense that a Project is able to be shared among multiple FME users.

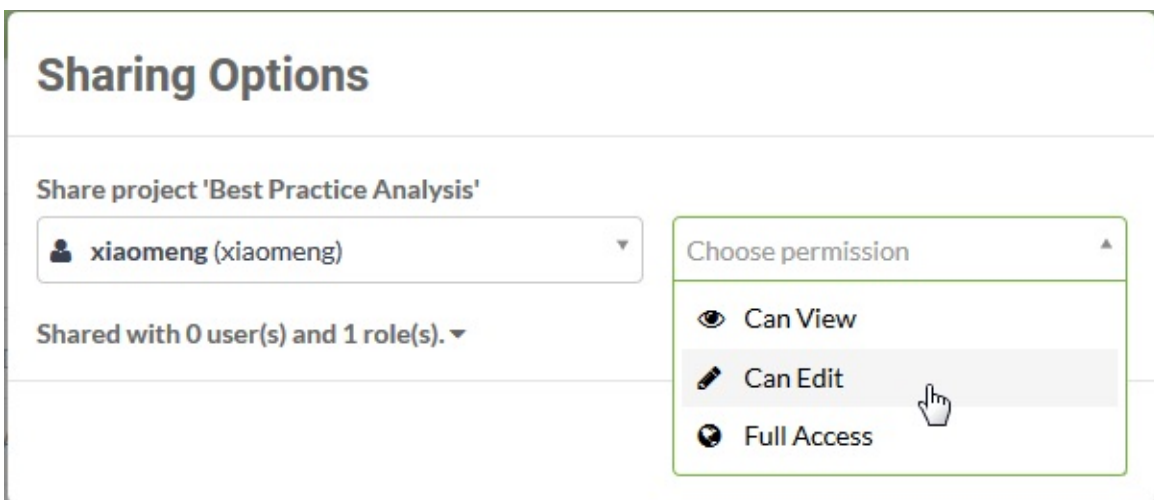
Share Project

Sharing a project is carried out on the Projects page (accessed via the main menu) by simply clicking the Share with Others button for the Project to be shared:



You don't even need to select the Project first.

The Share with Others tool works the same way as the tool for sharing a repository: it opens a pop-up dialog in which to select a user and choose the level of permission that you wish to give to them:



Remember, FME Security is based on users and roles. Roles are analogous to a group of users. When sharing a Project, the "user" field can be an individual user, or it can be applied to a particular role; for example you can give the ability to view your project to anyone in the *fmeuser* role.

Sharing a Project gives the ability to access not just repositories and workspaces, but all of the components related to that Project.

Miss Vector says...

A Project can be shared only in the following circumstances:

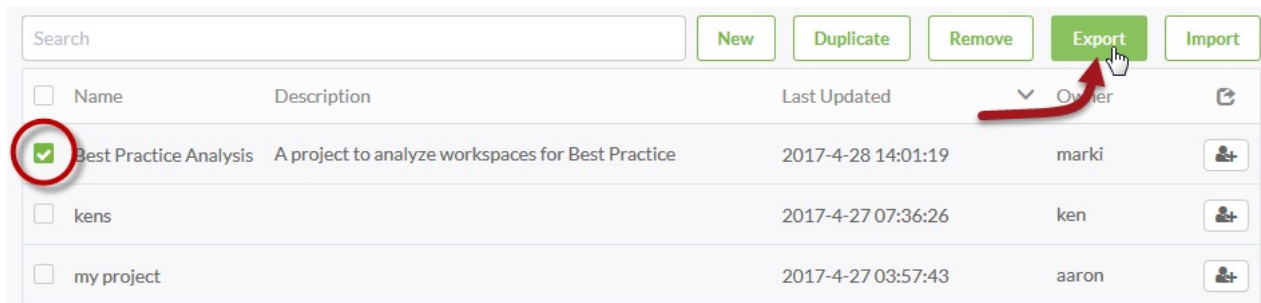
- 1. You must own the Project. Only the Project owner can share it.*
- 2. You must be a user with permission to manage security. Only such a user can share a Project.*
- 3. You can own the Project OR be a user with permission to manage security (i.e. you can be one or the other).*
- 4. You own the Project AND you are a user with permission to manage security (i.e. you must be both).*

Managing Projects

One of the key uses for a Project is for transferring a set of FME Server components from one Server instance to another. This is carried out by using the Export and Import tools. There is also a tool for Removing a Project from a system.

Exporting a Project

Exporting a project is carried out on the Projects page (accessed via the main menu) by selecting the Project to be exported and clicking the Export button:



This opens a dialog in which to configure and carry out the export:

The screenshot shows the 'Configure Export' interface in a web application. At the top, there is a breadcrumb 'Projects > Export' and a 'View Log' button. The 'Configure Export' section has three fields: 'Project(s)' with a dropdown showing 'Best Practice Analysis', 'Filename' with a text input 'Best Practice Analysis_2017-4-28-T140235', and 'Export To' with a dropdown showing 'Download'. Below this, the 'Export Complete' message is displayed with a green checkmark icon and the text 'The export process is now complete.' To the right, a Firefox file opening dialog is shown. The dialog title is 'Opening Best Practice Analysis_2017-4-28-T140324.fsproject'. It states 'You have chosen to open:' followed by a file icon and the name 'Best Practice Analysis_2017-4-28-T140324.fsproject', which is identified as an 'fsproject File' from the URL 'https://fmewt2017-safe-software.fmecloud.com'. The dialog asks 'What should Firefox do with this file?' and offers four options: 'Open with' (with a 'Browse...' button), 'DownThemAll!', 'dTa OneClick!' (with a download icon and a path dropdown set to 'C:\Users\imark\Downloads\'), and 'Save File' (which is selected). There is also a checkbox for 'Do this automatically for files like this from now on.' and 'OK' and 'Cancel' buttons at the bottom.

The export writes to a file with a .fsproject extension. There is an option to either provide the export as a download, or to write it to a Resources folder.

Miss Vector says...

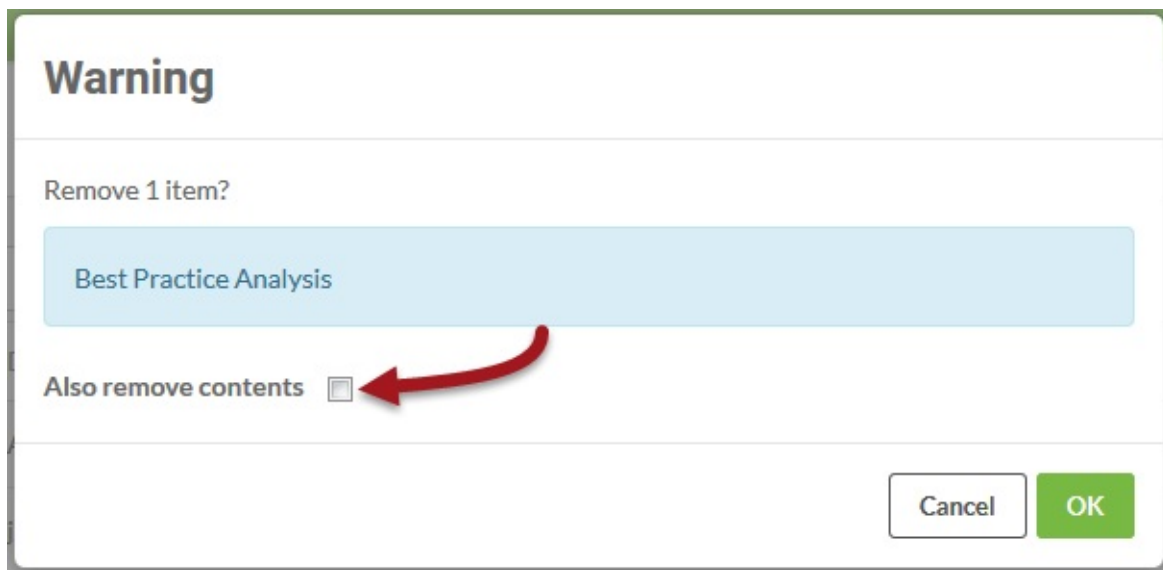
If you choose to export a Project to a Resources folder (rather than download it) then what additional capability do you gain?

1. The ability to trigger a notification topic on completion of the export.
2. The ability to export the FME license for the server.
3. The ability to remove all components of the project as they are exported.
4. The ability to change ownership of the components to your own user account.

Removing a Project

Once a Project has been exported, you may wish to remove it from the Server instance on which it currently resides. That can be done by selecting the Project and clicking the Remove button.

The confirmation dialog that opens allows you to choose whether to also delete all of the components that make up the Project:



Miss Vector says...

Checking the box to remove the contents of a Project removes all of the project components:

1. True
2. False

Importing a Project

Importing a project is - like exporting - carried out on the Projects page; this time by simply clicking the Import button.


The Import page allows an fsproject file to be uploaded or read from a Resources folder. It also includes other parameters to control the import:

Configure Import

Overwrite Existing Items ☐


Pause Notifications System ☒

Import From

Upload 

Choose .fsproject File

The process will start immediately after uploading.


Drop file to upload

Upload File

Miss Vector says...

When you migrate a Project that contains users, and import it on another system, which of these statements apply (there may be more than one)?

- 1. The user receives exactly the same permissions as on the original system.*
- 2. The user receives the same permissions as on the original system, but only for items in the Project.*
- 3. The user receives the same permissions as on the original system, but only when they were granted to the user (and not to a role the user belongs to).*
- 4. The user receives the same permissions as on the original system, but suspended until a moderator approves them.*

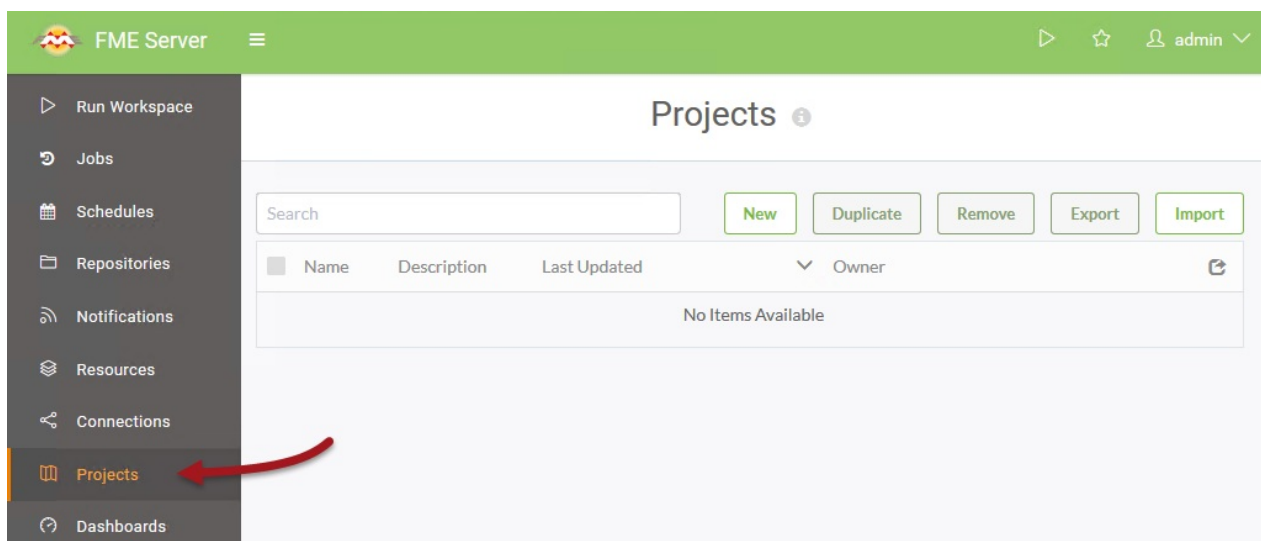
| Exercise 1 Best Practice Workspace Analysis Project | |
|---|------------------------------|
| Data | Workspace Files |
| Overall Goal | Set Up an FME Server Project |
| Demonstrates | FME Server Projects |
| Start Workspace | N/A |
| End Workspace | N/A |

Best Practice is a very important concept for FME workspaces. To encourage colleagues to carry out best practices you wish to install a project that allows workspaces to be analyzed.

1) Browse To Projects

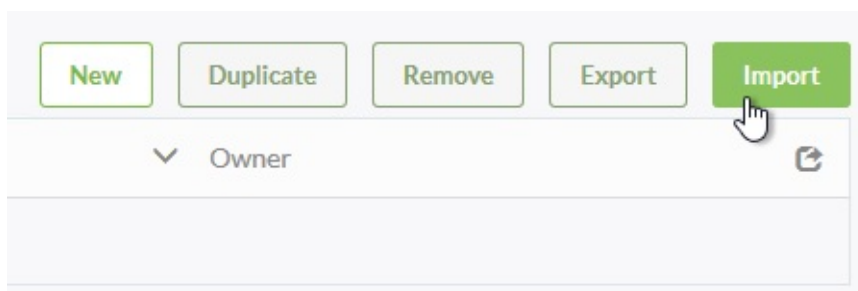
Open the FME Server web interface and log in with an account that has administrator privileges.

Select Projects on the main menu to browse to the Projects page:

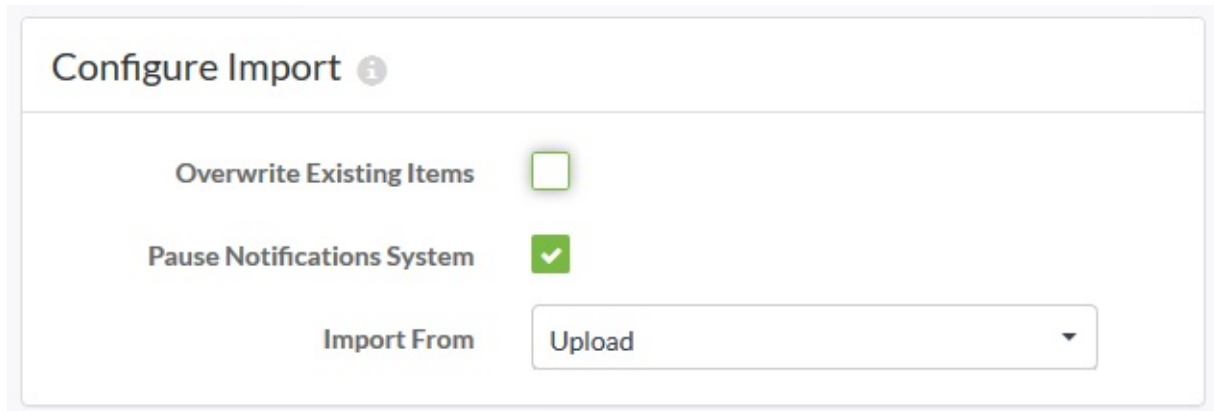


2) Import Project

Click on the Import button to open the Configure Import dialogs:



In the Configure Import section, be sure to set the import to be by an upload:



Configure Import ⓘ

Overwrite Existing Items ☐

Pause Notifications System ☒

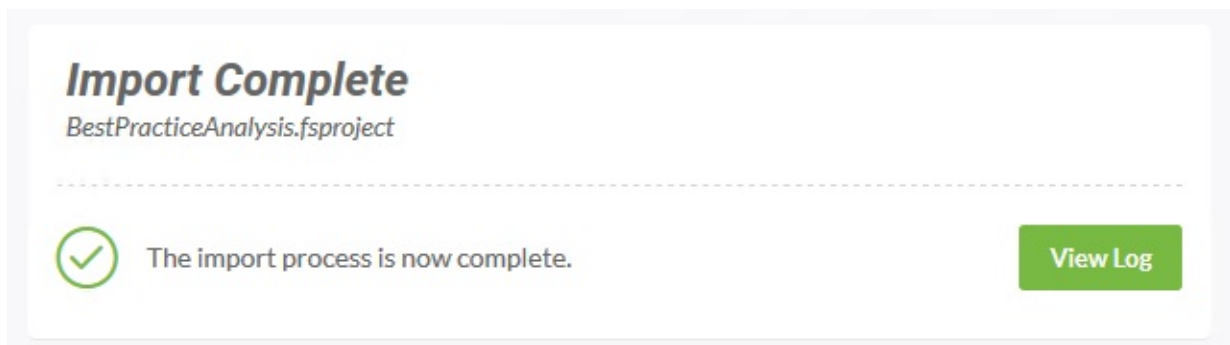
Import From Upload ▼

Overwrite Existing Items is less important because the project should not yet exist for items to need overwriting. Similarly, *Pause Notifications System* is not important because it's very unlikely the notifications in the project will be triggered immediately (they are for handling incoming emails).


Click the Upload File button and browse to/select the file

C:\FMEData2017\Resources\CodeSmellsWorkshop\BestPracticeAnalysis.fsproject

The project will very quickly be imported:



Import Complete
BestPracticeAnalysis.fsproject

 The import process is now complete.

[View Log](#)

3) Check Log

Click the View Log button in order to examine the Backup/Restore log (which is where project imports are documented). A successful import will look something like this (some columns removed for brevity):

```

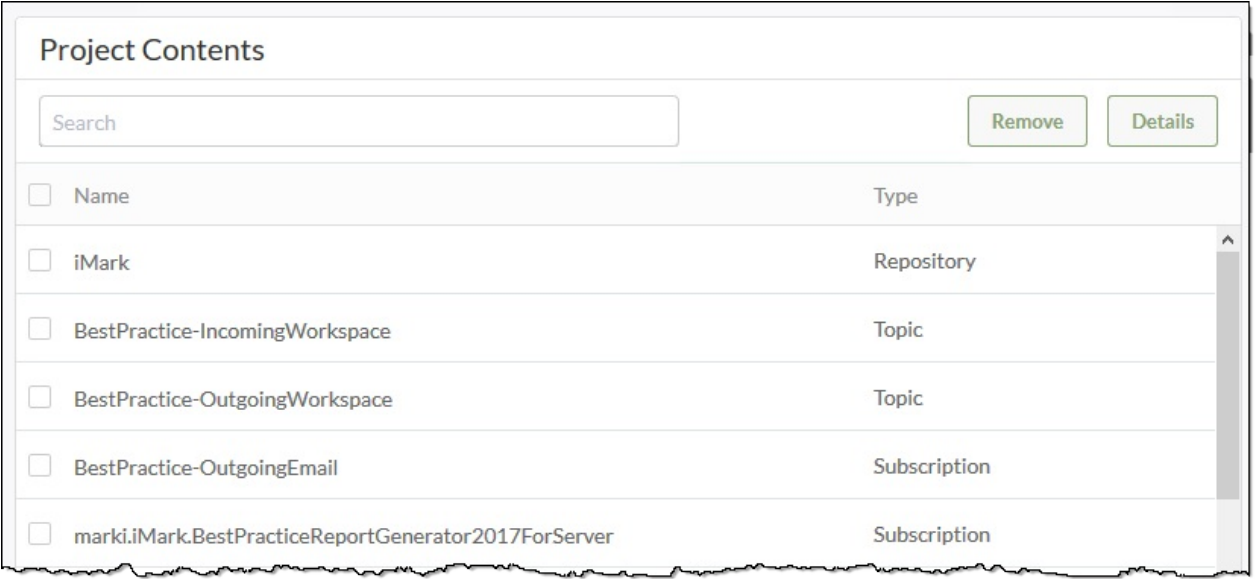
Wed-07-Jun-2017 01:38:28 PM INFORM: (Migration) Received a
configuration package for import.
Wed-07-Jun-2017 01:38:28 PM INFORM: (Migration) Unzipping
configuration package...
Wed-07-Jun-2017 01:38:28 PM INFORM: (Migration) Upgrading
configuration package schema version...
Wed-07-Jun-2017 01:38:29 PM INFORM: (Migration) Importing
configuration package content to server...
Wed-07-Jun-2017 01:38:33 PM INFORM: (Migration) Imported
configuration package successfully.

```

4) Check Components

Now let's check for some of the components that should have been imported.

Click Projects on the menu again, and select the recently imported project. You should now see a list of the imported contents:



The screenshot shows a web interface titled "Project Contents". At the top, there is a search bar with the placeholder text "Search". To the right of the search bar are two buttons: "Remove" and "Details". Below the search bar is a table with two columns: "Name" and "Type". Each row in the table has a checkbox in the "Name" column. The table contains the following entries:

| Name | Type |
|---|--------------|
| <input type="checkbox"/> iMark | Repository |
| <input type="checkbox"/> BestPractice-IncomingWorkspace | Topic |
| <input type="checkbox"/> BestPractice-OutgoingWorkspace | Topic |
| <input type="checkbox"/> BestPractice-OutgoingEmail | Subscription |
| <input type="checkbox"/> marki.iMark.BestPracticeReportGenerator2017ForServer | Subscription |

Use the menu options to check the Repository, Notifications, and Resources pages to ensure that the imported components do really exist.

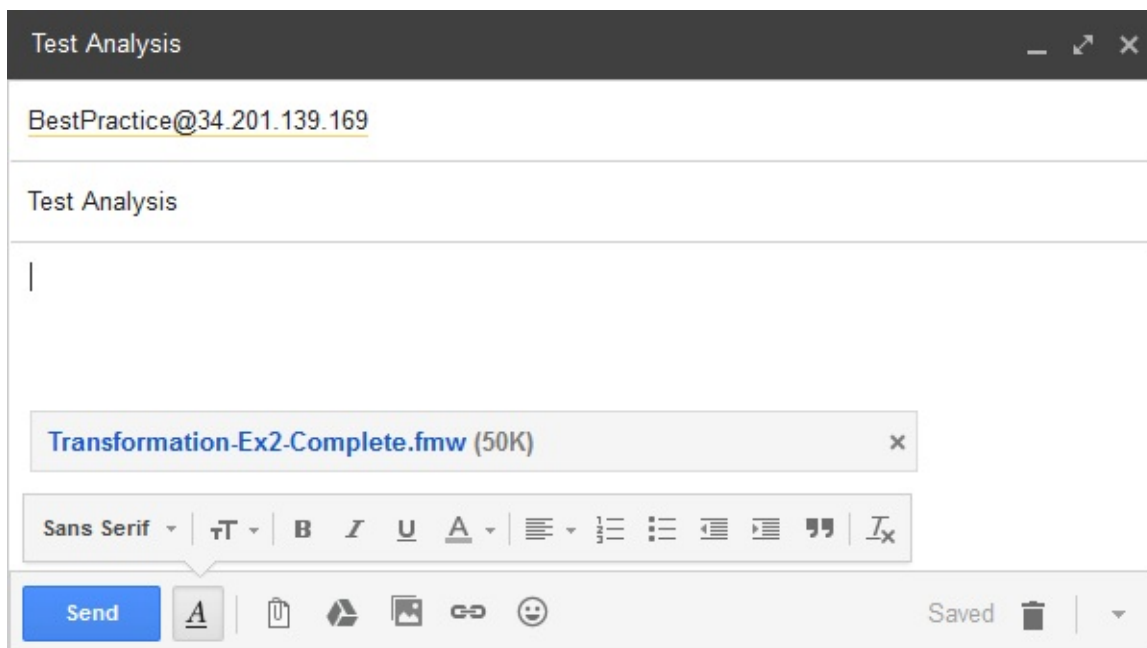
5) Test Project

Now let's send an email to your FME Server to test the project. This assumes that you are using a server that has a public name, domain, or address.

For FME Server on one of Safe's training computers, the public IP address is shown on the top-right of the desktop, or within the readme file obtained when you started the computer:


```
Hostname: FMETraining
Instance ID: i-0ebfa82fc28331c43
Public IP Address: 34.201.139.169
Private IP Address: 172.30.1.77
Instance Size: t2.large
Availability Zone: us-east-1b
Architecture: AMD64
Total Memory: 8 GB
Network Performance: Low to Moderate
```

The email address will be BestPractice@xxxx, where xxxx is the IP address:



Set a subject line and attach a workspace file. Click the Send button. In response (it may take a minute or two) you will receive an email report about the best practices used in that workspace:

FME Best Practice Analysis Results Inbox x

 fmebestpractice@gmail.com 16:28 (1 minute ago) ☆
to me ▾

FME Workspace

Best Practice Report

| | |
|-------------------------|---|
| Report Name | Test Analysis |
| Report Generated | 14:28:58.6253168 UTC 07-06-2017 |
| Workspace Analyzed | C:\ProgramData\Safe Software\FME Server\resources\system\temp\emailattachments\20170607142853-BestPractice@34.201.139.169-Test_Analysis\Transformation-Ex2-Complete.fmw |
| Workspace File Size | 49 kb |
| Workspace Created | 201757142853 |
| Workspace Last Modified | 201757142853 |
| Workspace Last Run | 201757142853 |

This report examines best practices for a workspaces that is about to be put into production.

This demonstrates that the project has been imported and set up correctly.

6) Clean Up Project

One part of the project that is not needed is a user account.

So, return to the project contents, select the iMark account, and remove it.

Project Contents

[Remove](#) [Details](#)

| <input type="checkbox"/> | Name | Type |
|-------------------------------------|------------------------------|-----------------|
| <input type="checkbox"/> | Data/BestPractices/ | Resource Folder |
| <input type="checkbox"/> | Engine/Transformers/ | Resource Folder |
| <input type="checkbox"/> | FMEBestPractice Google Gmail | Connection |
| <input checked="" type="checkbox"/> | marki | User |

Since the project has been imported, the account will also exist on the machine (the above only removed it from the project). So also visit the security pages and remove that user.

Send another email to confirm that the project is still working.

7) Export Project

Now the project has been updated, export it so that it can be imported in its proper form elsewhere.

To do so, browse to the Projects page, select the project (using the checkbox on the left), and click the Export button.

In the dialog that opens you can choose whether to save the project file to a download or a resources folder. Once complete the following message will appear:

Export Complete

Best Practice Analysis_2017-6-7-T144014.fsproject

☒

The export process is now complete.

[View Log](#)

CONGRATULATIONS

By completing this exercise you have learned how to:

- *Import a Project*
- *Check the log and confirm a Project was successfully imported*
- *Edit a Project's contents*
- *Export a Project*

Troubleshooting for Administrators

This section shows a few basic troubleshooting techniques in case of emergency.

Project Import Fails

If you are having troubles with the content being different than expected:

- As you might have guessed, an fsproject file is simply a zip file with a different extension. Thus you can explore the files within, and the XML documents that define their structure, to see if it is what you expect.
-

Module Review

This module introduced you to FME Server's Projects.

What You Should Have Learned from this Module

The following are key points to be learned from this module.

Theory

- Projects are a collection of different FME Server components
- Projects are an efficient way to manage multiple components
- Projects allow administrative-type tasks to be carried out by authors
- Projects can be used to import/export/share FME Server components

FME Skills

- The ability to create an FME Server project and add components to it
- The ability to share an FME Server project with other users
- The ability to export an FME Server project
- The ability to remove an FME Server project
- The ability to import an FME Server project

Further Reading

For further reading why not check out...

- [This blog article on FME Server Projects](#)

Questions

Here are the answers to the questions in this chapter.

Miss Vector says...

A Project can be shared only in the following circumstances:

- 1. You must own the Project. Only the Project owner can share it.*
- 2. You must be a user with permission to manage security. Only such a user can share a Project.*
- 3. You can own the Project OR be a user with permission to manage security (i.e. you can be one or the other).**
- 4. You own the Project AND you are a user with permission to manage security (i.e. you must be both).*

If you own the Project then you can share it. But if you have permission to manage security, then you can share any Project.

Miss Vector says...

If you choose to export a Project to a Resources folder (rather than download it) then what additional capability do you gain?

- 1. The ability to trigger a notification topic on completion of the export.**
 - 2. The ability to export the FME license for the server.*
 - 3. The ability to remove all components of the project as they are exported.*
 - 4. The ability to change ownership of the components to your own user account.*
-

Miss Vector says...

Checking the box to remove the contents of a Project removes all of the project components:

- 1. True*
- 2. False**

A Project can contain users and roles. The remove tool won't remove the user carrying out the removal, nor will it remove any role for which that user is part of. Also it won't remove components you don't have permission to remove.

Miss Vector says...

When you migrate a Project that contains users, and import it on another system, which of these statements apply (there may be more than one)?

- 1. The user receives exactly the same permissions as on the original system.*
- 2. The user receives the same permissions as on the original system, but only for items in the Project.**
- 3. The user receives the same permissions as on the original system, but only when they were granted to the user (and not to a role the user belongs to).**
- 4. The user receives the same permissions as on the original system, but suspended until a moderator approves them.*

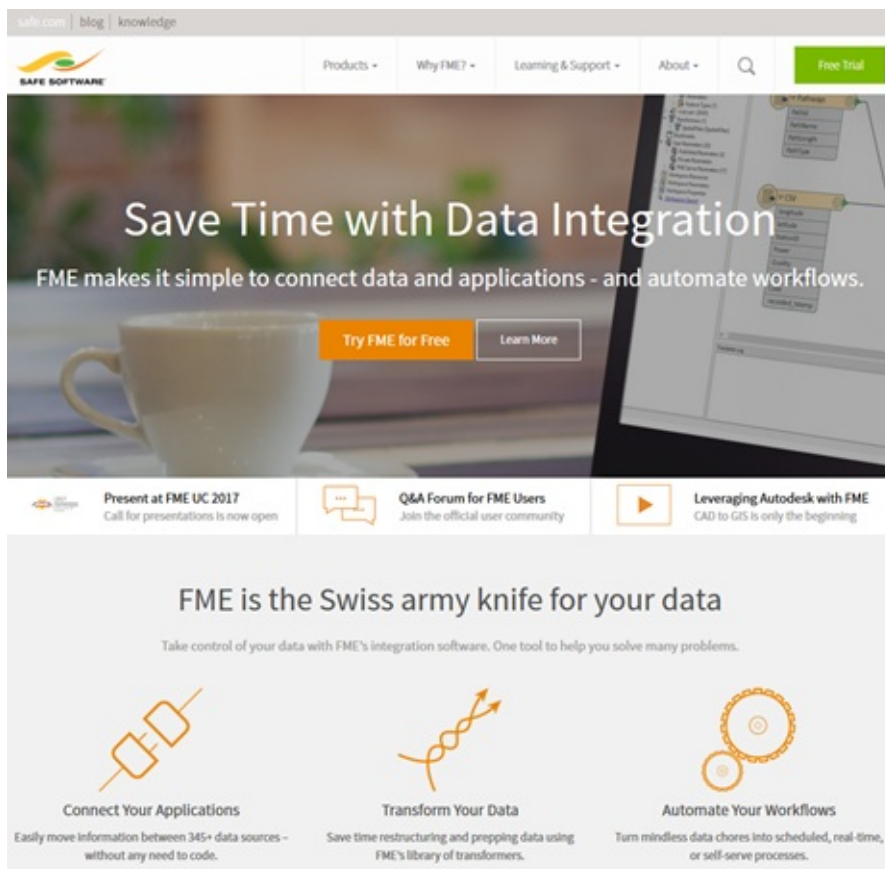
Course Wrap-Up

Although your FME training is now at an end, there is a good supply of expert information available for future assistance.

Product Information and Resources

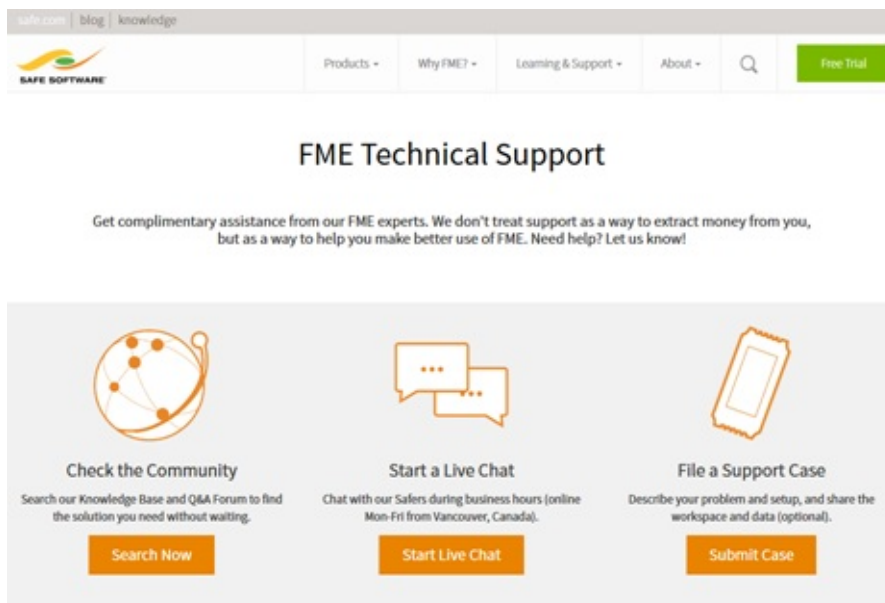
Safe Software Web Site

The [Safe Software web site](#) is the official information source for all things FME. It includes information on FME products, Safe Software services, FME solutions, FME support and Safe Software itself.



Safe Support Team

Behind FME are passionate, fun, and knowledgeable experts, ready to help you succeed, with [a support team](#) philosophy built on the principle of knowledge transfer.



You can request product support through a Support Case (web/email) or using Live Chat.

Your Local Partner

Safe Software has partners and resellers around the world to provide expertise and services in your region and your language.



You can find a list of official partners on the [Safe Software web site](#).




Safe Software Blog

The [Safe Software blog](#) provides technical information about FME, articles about customers' use cases, and general thoughts on spatial data interoperability.

blog.safe The Safe Software Blog

[About Data](#)[About FME](#)[About Our Customers](#)





[About FME](#) | [December 15, 2016](#) | [By Claude Vissier](#)


Improving FME Server Performance with NGINX


All new FME Server 2017.0+ and 2016.1.3 instances are now running behind a NGINX reverse proxy in FME Cloud. As a developer on the FME Cloud team, this is something I have been hoping to achieve for a while and with 2016.1.3+ all the pieces are finally in place. In this blog post, I would [...]

[READ MORE](#)



 Comment

 Share





[About FME](#) | [December 7, 2016](#) | [By Mark Ireland](#)


FME Parallel Processing: Tips and tricks for generating groups


I recently taught the performance chapter of our FME Desktop advanced training course and got into a conversation with a student about creative ways to use parallel processing. Some of the ideas we came up with about generating 'groups' were so interesting I thought I would share them. I hope you find them of use.

[READ MORE](#)



 Comment

 Share





[About Data](#) | [December 5, 2016](#) | [By Tiana Warner](#)


The GeoGeek's Guide to Planet's Satellite Imagery

Small satellite technology is completely changing the world. Here's what Planet is doing to revolutionize Earth Observation data, and how you can leverage satellite trends to get ahead in your industry. Satellites hit a turning point a few years back, when an upsurge in the number being built culminated in 94 launches over the span [...]

[READ MORE](#)

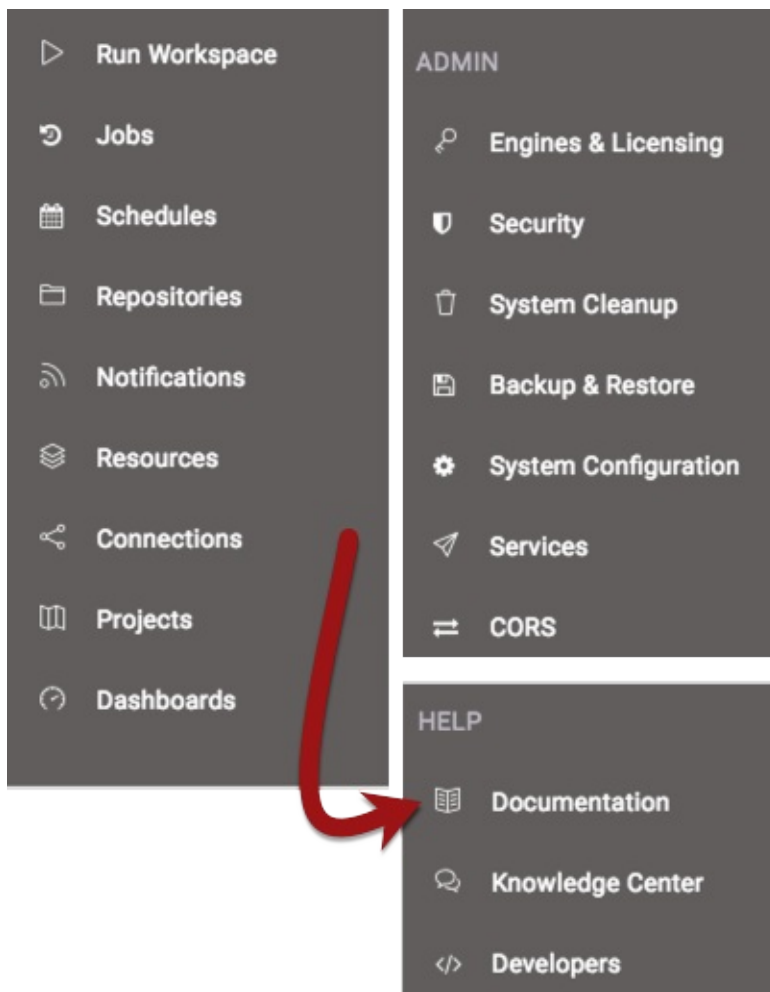


 Comment

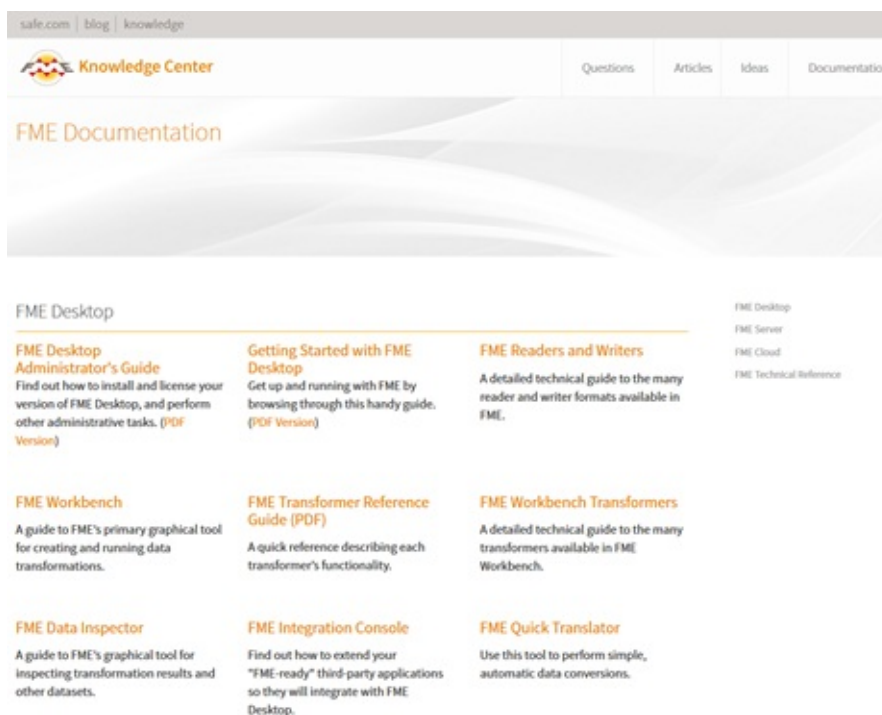
 Share

FME Manuals and Documentation

For FME Server documentation, click the Documentation item in the FME Server web interface menu:



For FME Desktop use the Help function in FME Workbench to access help and other documentation. Alternatively, look on our web site under the [Knowledge Center section](#).

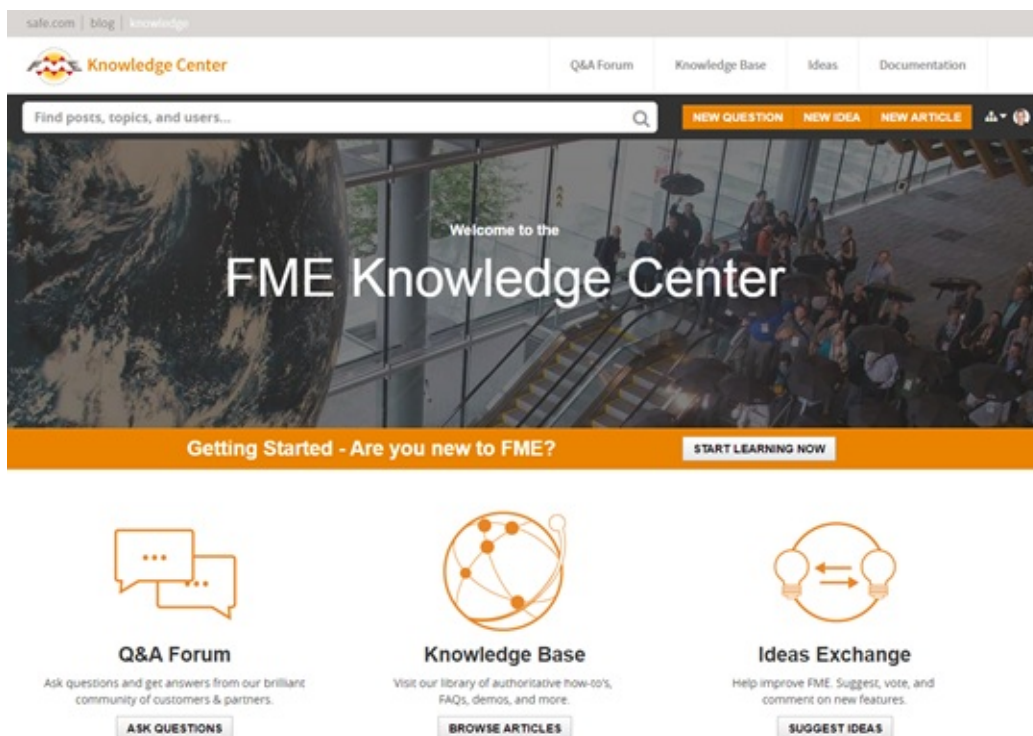


Community Information and Resources

Safe Software actively promotes users of FME to become part of the FME Community.

The FME Knowledge Center

The **FME Knowledge Center** is our community web site - a one-stop shop for all community resources, plus tools for browsing documentation and downloads.



Knowledge Base

The FME Knowledge Base contains a wealth of information; including tips, tricks, examples, and FAQs. There are sections on both FME Desktop and FME Server, with articles on topics from installation and licensing to the most advanced translation and transformation tasks.

Q&A Forum

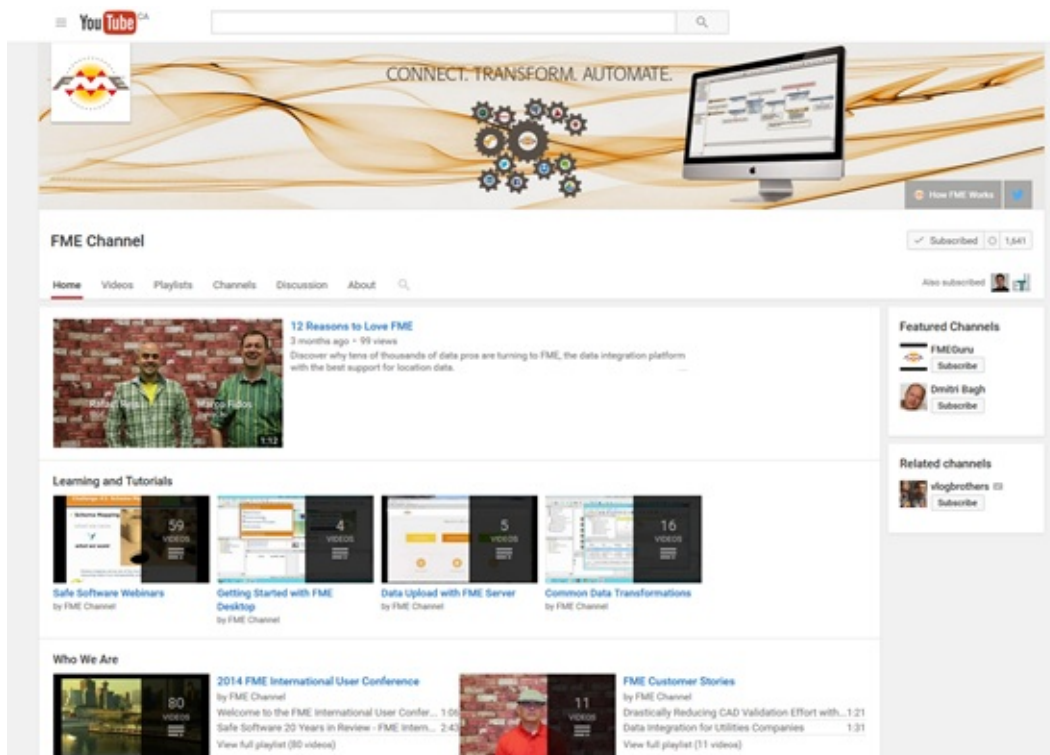
FME community members post FME-related messages, ask questions, and share in answering other users' questions. Members earn "reputation" and "badges" and there is a leaderboard of the top-participating users. Come and see how they can help with your FME projects!

Ideas Exchange

FME development is very much user-driven. The Ideas Exchange gives users the chance to post their ideas for new FME functionality, or improvements to existing functionality, and allows everyone to vote on the proposed ideas. The more votes an idea gets, the more likely it is to be implemented!

The FME Channel

This [FME YouTube channel](#) is for those demos that can only be properly appreciated through a screencast or movie. Besides this there are a host of explanatory and helpful movies, including recordings of most training and tutorials.



Feedback and Certificates

The format of this training course undergoes regular changes prompted by comments and feedback from previous courses.

Course Feedback

Miss Vector says...

There's one final set of questions – and this time you'll be telling me if the answers are correct or not!

Safe Software greatly values feedback from training course attendees and our feedback form is your chance to tell us what you really think about how well we're meeting your training goals.

You can fill in [the feedback form](#) now, but you'll also be reminded by email shortly after your course. Safe Software's partners who carry out training may ask that you fill in a separate form, but you can also use the official Safe Software form if you wish.


Certificates

Mr. E. Dict, (Attorney of FME Law)says...

In order to prove you have taken this training course, a certificate will be emailed automatically to anyone who was logged on for the duration of Safe Software hosted courses.

Thank You

Thank you for attending this FME training course.



All of us at Safe
Software wish you
good luck and lots
of fun as you
conjure up exciting
new FME magic

Congratulations!

As a reward for reading this far, here's a little challenge for you to try out.

Various folk have something to say to you, but can you figure out what it is? Maybe FME can help?

Miss Vector says...

*.6 si ecnetnes siht rof rebmun edoc ehT .rebmun edoc a dnif ot gnidoced
ralimis sdeen ecnetnes hcaE .ylreporp ti tamrof ot EMF esu ot si egnellahc
eht ,yawyna daer ylbaborp dluoc uoy ecnetnes eno si siht hguohtlA
.snoitalutargnoC*

Dr Workbench says...

57656C6C20646F6E652E20596F75207265636F676E697A65642074686973
2061732068657820656E636F64656420746578742E204920686F706520796
F75206465636F64656420697420776974682074686520546578744465636F
646572207472616E73666F726D65722E20427920746865207761792C2074
686520636F6465206E756D6265722066726F6D206D652069732039

Sister Intuitive says...

11114604017115716504014715716404016415014504016016214516615115
71651630401631451561641451561431450401511640401631501571651541
44040150141166145040142145145156040152165163164040141163040145
14116317104016415704014414514315714414504016415015116304015714
31641411540401641451701640401621451601621451631451561641411641
51157156040165163151156147040164150145040124145170164104145143
15714414516204016416214115616314615716215514516205604012415015
1163040143157144145040156165155142145162040151163040061

Police Chief Webb-Mapp says...

Lbh ner tbbq, nera'g lbh. Gur pbqr ahzore sbe guvf fragrapr vf 3. Bs pbhefr, gung ahzore nccrnef va pyrne grkg, ohg bayl lbh jvyy xabj gur cnffjbeq gb gur arkg pyhr vf gur svefg sbhe pbqrf pbagnpgrangrq gbtrgure!

Miss Vector says...

C:\FMEDData2017\Resources\Challenge\FinalChallenge.ffs