

Welcome to ArcGIS Data Interoperability

These help files contain help topics specific to FME Workbench. Note that not all functionality is available in ArcGIS Data Interoperability.

Download Transformer Quick-Reference

Click the link below to access a printable booklet in PDF format:

- FME 2011 Quick-Reference Transformers (PDF format)

Note that you must have Adobe Reader to open the PDF file. You can get it from the FME installation disk, or here for free.

FME Help, Format Information and Technical Reference

Workbench Help contains information on performing translations and transformations using FME Workbench.

Note: Especially for advanced operations, you may have to look in different help files to get the information you need.

Access these reference documents from the Workbench Help menu.

- FME Transformers: Help for transformers is also integrated into the main Transformer Description window.
- FME Readers and Writers: Detailed information on FME-supported formats. Note that most reader and writer directives correspond to Workbench parameters in the Navigation pane.
- FME Functions and Factories: Detailed technical information on functions and factories that correspond to Workbench transformers.
- FME Fundamentals: Detailed reference information on FME fundamentals, that is, how FME processes data. If you want to write or customize your own mapping files, or just want to know how FME works, you can find detailed information on FME architecture, functions, feature factories, interfaces, transformation process, mapping file syntax, and configuration.
- Coordinate System Help: You can also access coordinate system help by pressing the F1 key from the Coordinate System Gallery.

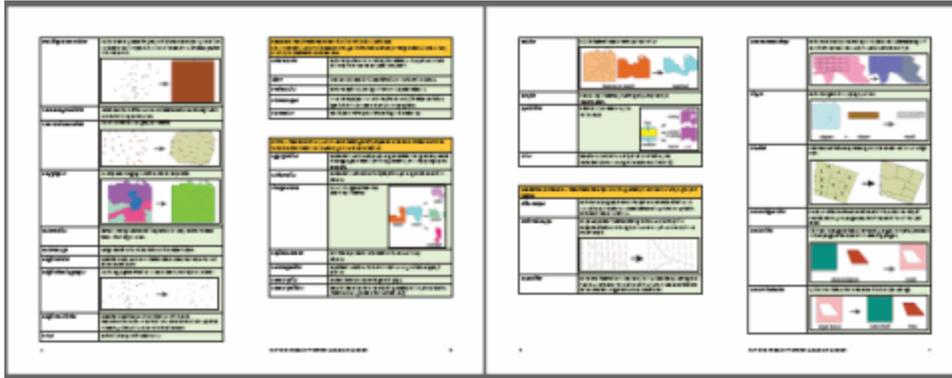
Separate help files exist for FME Universal Translator and FME Universal Viewer.

- FME Universal Viewer Help Files: Start the FME Universal Viewer and click the Help menu.
- FME Quick Translator Help Files: Start the FME Quick Translator and click the Help menu.

FME Workbench Transformers Reference Guide

Click the link below to open the guide (from Safe Software's website). You can also right-click on the link and choose *Save Target As* to save the file directly to your computer or another location.

- [FME 2011 Transformers Reference Guide \(PDF format\)](#)



Note that you must have Adobe Reader to open the PDF file. You can get it from [here](#) for free.

System Requirements

For a complete list of system requirements, including the differences between 32-bit and 64-bit FME, please visit our website.

Windows 32-bit (win32)

- Windows 7
- Windows Vista
- Windows 2008 Server
- Windows 2003 Server
- Windows® XP SP1

Windows 64-bit (win64):

- Windows 7 64-bit
- Windows Vista 64-bit
- Windows 2008 Server 64-bit
- Windows 2003 Server 64-bit

All 64-bit Windows editions listed above also support 32-bit FME. Please visit www.safe.com for information on the differences between 32-bit FME and 64-bit FME.

For more information, please contact support@safe.com.

For FME Premium Editions, FME Smallworld Edition:

- TCP/IP network connection to an ESRI® SDE server (to use the SDE modules)

UNIX (command line only):

- IBM® RS/6000 AIX 6.1
- Sun® SPARC Solaris™ 9, 10

Linux (Desktop Light [without Workbench]):

- Linux® Intel x64 (Red Hat Enterprise Linux AS/ES/WS 5.0)

Note that FME releases after FME 2011 will not be provided on the Linux Intel x86 platform.

Browser

- To view the Help files, you will need Internet Explorer 5.0 or higher (6.0 recommended)
- When viewing Help files on our website, only Google® Chrome does not support the WebHelp format

Specifications

System specifications will vary depending on your FME usage, including the size of your data files. However, at minimum, we recommend:

- Minimum Intel® Pentium® III or 4 processor
- 1 GB of RAM

- 600 MB of available disk space

Note: To run FME on the Citrix Platform, a Floating License is required.

FME Tutorial, Self-Study Modules and Training

FME Tutorial

One of the best ways to get up to speed with FME is to download the FME product tutorial.

Self-Study Modules

Improve your proficiency with FME at a pace that suits your schedule. Access a variety of self-guided modules - based on official FME training courses - for convenient learning from your office.

Improve your proficiency with FME at a pace that suits your schedule. Our FME self-study modules will help you build the FME skills you need to solve your data integration challenges. Work through the complete package of 8 modules one at a time, as your availability allows. Or, if you're already an intermediate-level FME user, dive right into the topics that interest you the most.

 [MORE](#)

Training Courses

For more in-depth, instructor-led training, consider taking one of our FME Training Courses. Both regular and advanced courses are offered, and each include practical exercises designed to help you apply your new FME skills to common data processing tasks.

 [MORE](#)

FME Demos

Please visit the Safe Software website.

Localized Software

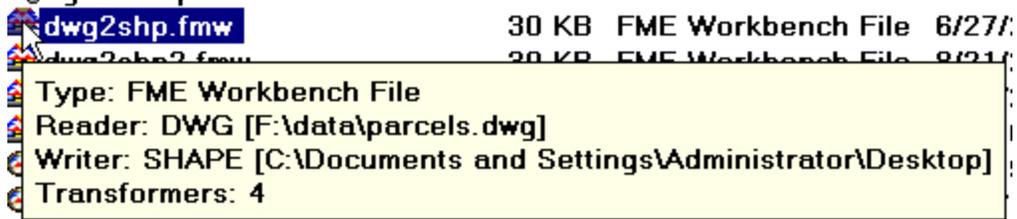
FME localization add-ons are available in other languages. For more information, please see localized FME software and documentation on our website.

FME-Related Filenames

Filename/File Extension	Type	Default Location
.fmw	FME Workbench filename	My Documents\My FME Workspaces
.fme	FME mapping file (FME Quick Translator)	user-defined
.fmv	FME Universal Viewer file	user-defined
.fds	Custom Formats	My Documents\FME\Formats
.fmx	Custom Transformers	My Documents\FME\Transformers
.fmxlist	User-defined transformer list	My Documents\FME\TransformerCategories
.fmc	FME catalog file Contains instructions to direct Workbench to download a specified resource from fmepe-dia.com. Since these files contain only instructions (XML), they are small in size.	
.wbtheme	Workbench theme file	My Documents\FME\Themes
MyCoordSysDefs.fme	custom coordinate system definitions	In the FME installation directory, Reproject
.gdc, .gsb, .las, .los, .mrt, .txt, .csv	Coordinate system and grid data files	Reproject Reproject\GridData
coordsys.db	contains the names and descriptions of all predefined coordinate systems	FME installation directory
LocalCoordSysDefs.fme	This file is automatically loaded and made available to each FME session, to allow sites to add their own coordinate systems. It contains a series of COORDINATE_SYSTEM_DEF, DATUM_DEF, ELLIPSOID_DEF, and UNIT_DEF lines that define additional, site-specific coordinate systems. You can edit these files to add your own definitions.	Reproject subdirectory under the FME installation directory Defining Custom Units

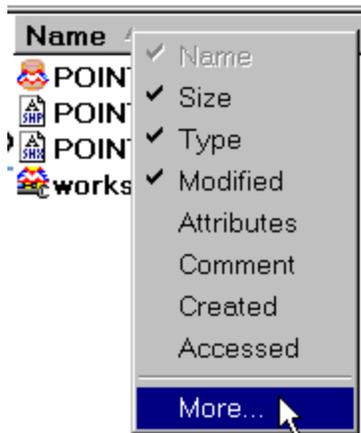
File Management

Through Windows Explorer, you can view summary information about FME-generated files without having to start FME and open the file. For Workbench files, this includes the reader, writer, and number of transformers. Float the cursor over a filename to see the information in a pop-up window.

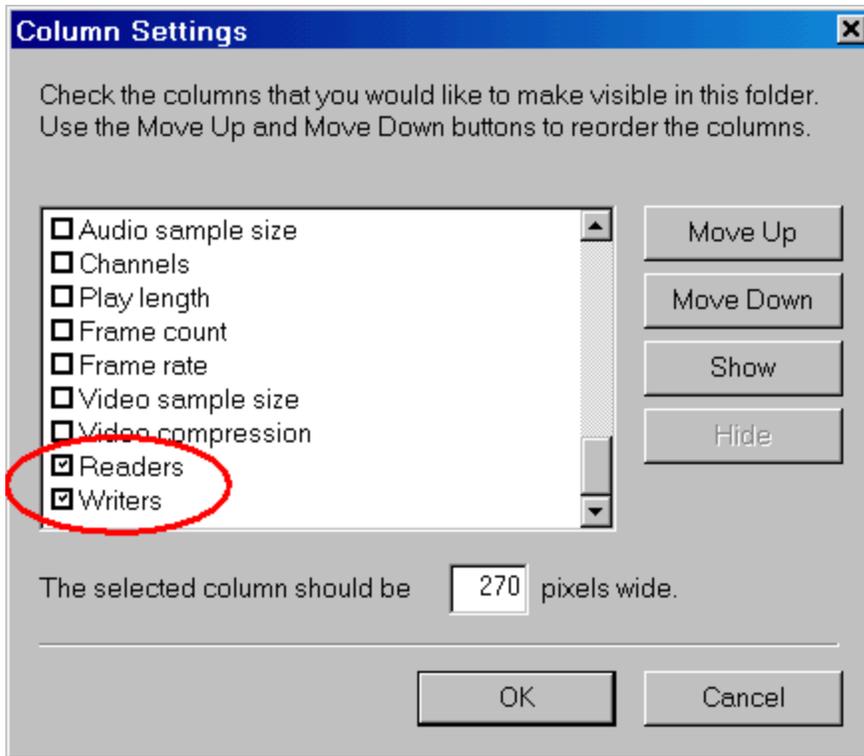


You can also view the source and destination format in two additional columns in Windows Explorer:

1. Open Windows Explorer, and go to the directory that contains your FME files.
2. Right-click on an existing column heading to display the command menu, and select More.



3. Scroll down and check the boxes labelled Readers and Writers.



4. Click OK to display the columns in Windows Explorer.

NOTE: You will have to repeat these steps for each directory that you want to display the Readers and Writers columns.

Daylight Savings Time

FME's performance is not affected in any way by changes in local time zones related to daylight savings, or by extensions to the number of weeks of daylight savings time, as long as your computer's clock has been set to automatically adjust for daylight saving changes.

Users running FME on a Windows operating system can check this option has been selected by opening their Control Panel, selecting Date and Time options, then selecting the Time Zone tab.

Adjusting Memory Resources

One issue that can affect FME performance when you're working with very large datasets is available RAM. If you run a very large dataset through a workspace, you may see a message box that says "Out of Memory. Please free some memory, then choose retry." What this means is that no more memory can be allocated. You can try the following options:

- If you have 4 GB of RAM (available in version FME 2006 GB and above), you can try turning on the 3GB switch.
- You can also refer to the hints in Performance Tuning FME.

Using the /3GB Switch

Note: The /3GB switch is not required for all, or even most, users of FME. The /3GB switch will only benefit those who require translations that will not currently run with 2GB of addressable memory.

Thirty-two-bit versions of the Windows operating system can manage a maximum of 4GB of addressable memory. The 4GB is divided into 2GB for user applications and 2GB for kernel processes. This means that any given application is restricted to 2GB of memory.

Beginning with FME 2006 GB, FME takes advantage of a /3GB switch which causes the operating system to divide the available 4GB of memory into 3GB for user applications and 1GB for kernel processes. Many translations that previously failed due to memory limitations will now run successfully when the /3GB switch is used.

Which operating systems support the 3GB Switch?

The /3GB switch is supported on the following operating systems:

```
Windows XP Professional
Windows Server 2003
Windows Server 2003, Enterprise Edition
Windows Server 2003, Datacenter Edition
Windows 2000 Advanced Server
Windows 2000 Datacenter Server
Windows NT Server 4.0, Enterprise Edition
```

The /3GB switch is not supported on Windows 2000 Server.

Which version of FME do I need?

FME 2006 GB and greater.

How do I turn on the 3GB Switch?

Before FME can use the /3GB switch, you will need to edit your system's *boot.ini* file.

You can access this file by doing the following:

1. Open the System Properties dialog. You can access this dialog by either opening Control Panel and selecting System, or by right-clicking on My Computer and selecting properties.
2. Select the Advanced tab at the top of the System Properties dialog.
3. Click the Settings button under Startup and Recovery.
4. Click the Edit button under System startup. Notepad will open with the boot.ini file.

Under [operating systems], there should be a line that ends with /fastdetect. Add /3GB to the end of this line.

The following is an example of a boot.ini file before the /3GB switch has been added:

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(2)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(2)\WINDOWS="Microsoft Windows XP Professional" /fastdetect
```

When the /3GB switch is added, the above boot.ini file should look like the following:

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(2)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(2)\WINDOWS="Microsoft Windows XP Professional" /fastdetect /3GB
```

You will now need to restart your computer. When your computer has finished rebooting, your translation should be able to finish without running out of memory.

Note: It is recommended that you remove the /3GB switch when you don't require it. Simply remove the /3GB switch from your boot.ini file and restart your computer.

Will this work on my 64-bit machine?

It doesn't need to! For 64-bit versions of windows, FME can take advantage of 4GB of addressable memory. If you are running FME on a 64-bit version of Windows, no further action is required. Translations that previously failed because of limited memory should now work.

For more information, see this documentation from Microsoft (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/memory/base/4gt_ram_tuning.asp).

Where can I find more information?

Both this link (<http://www.microsoft.com/whdc/system/platform/server/PAE/PAEmem.msp>) and this link (<http://support.microsoft.com/default.aspx?scid=kb;en-us;328882&Product=exch2k>) to the Microsoft site have some good information.

Setting the Temporary Directory

When FME runs a large, multi-dataset translation, it often requires a lot of temporary disk space. This is particularly true when running a Dataset Fanout, because there is no guarantee that the features will arrive at the fanout in a single dataset group. Therefore, FME has to write out all of the datasets to temporary storage, and then fan them out afterwards. So the amount of available disk space is important, but on a performance issue you might be more concerned about the speed of all this disk activity.

Where possible, set your temporary directory to point to the fastest disk you have available. **This FAQ** tells you how to use FME_TEMP to set a different temporary directory.

Don't set your temporary directory on the same disk that the operating system uses; FME might be slowed down by the operating system writing to the same disk at the same time.

Try to set the temporary directory to a disk that has a large amount of free space - it won't improve the speed but it may prevent a large translation from failing due to a lack of disk space.

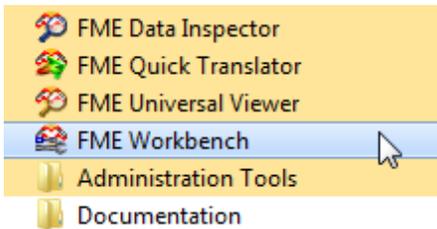
Purging Temporary Files

Sometimes FME creates temporary files that can take up unnecessary disk space (for example, stopping a translation will cause these files to build up).

It's a good idea to periodically purge these files by selecting Tools > Purge Temporary Files.

Starting FME Workbench

Find FME Desktop in the All Programs area of the Windows start menu. Click FME Workbench in the submenu.



Note that, depending on your version of Windows, the Start menu configuration might look slightly different.

When Workbench starts, it displays a Start tab. The information is generated from a live web page so this tab will always display the most up-to-date FME news, downloads, and resources:

A screenshot of the FME Workbench 'Start' tab interface. The interface is divided into several sections:

- Getting Started:** A sidebar on the left with options: 'Blank workspace', 'Create workspace...', 'Generate workspace...', and 'Open workspace...'.
- Recent Files:** A sidebar on the left listing recent workspace files: 'GlobeGenerator.fmw', 'LegoHouse.fmw', 'PolygonBuilding.fmw', and 'Workflow - Example4.fmw'.
- FME User Central:** A main content area with tabs for 'FME User Central', 'FME Support', and 'FME Training'. It features a magnifying glass over a red and white triangle logo, text describing FME User Central as an access point to technical resources, and a list of resources: 'fmepedia', 'FME Downloads', 'FME Evangelist', 'FME Guru', and 'FME Talk'. The URL 'www.FMEUserCentral.com' is displayed below.
- Twitter:** A section with a tweet from '@safesoftware' about architects learning from data-savvy GIS cousins, with the URL 'http://ow.ly/37Chb'. It includes a 'Follow us on Twitter' button.
- What's New:** A list of recent articles with dates and titles, such as 'Nov 10 What Can Architects Learn from their Data-Savvy GIS Cousins?' and 'Nov 3 LiDAR Storage and Analysis: What's Next?'.
- Documentation:** A section with the text 'Download documentation at www.safe.com/Doc' and an image of a book.
- Learning:** A section with the text 'Find self-study resources at www.safe.com/Learning' and an image of a stack of books.

The Getting Started and Recent Files areas provide options for creating new workspaces or opening existing files.

FME Workbench Interface

In Workbench, you'll work in *workspaces* that contain source and destination types (data) and their attributes, as well as transformers that manipulate the data as it moves from source to destination formats.

You can work with multiple workspaces at a time by starting additional Workbench applications, and you can copy and paste between workspaces.

Workspace files have the extension *.fmw*. For a default workspace name, FME suggests a filename based on the formats used in the workspace. For example, if your workspace is set up to read from an ESRI Shapefile and write to both an Access MDB file and AutoCAD drawing, then *shape2mdb_dwg.fmw* would be the suggested default name. You can always change the name through the File > Properties menu.

The View > Windows menu allows you to specify which information panes are displayed (for example, you can choose to hide any of the default panes, as well as display additional information about feature and attribute connections).

Menubar and Toolbar

The menubar and toolbar contain a number of tools: for example tools for navigating around the workspace, controlling administrative tasks and adding or removing source datasets.

Canvas

The Canvas is where you graphically defines your workflow. By default, the workflow reads from left to right; the reader (source data) is on the left, the transformation tools are in the center and the writer (data destination) is on the right. Connections between each item represent the flow of data and may branch in different directions or even lead to a dead-end if required.

Translation Log

The log pane displays a report on translation results. Information includes any warning or error messages, translation status, length of translation and number of features processed.

Navigator

The navigator is an explorer type tool that displays a text definition of source and destination datasets, plus all the settings that apply to these datasets.

Transformer Gallery

The transformer gallery is a tool for the location and selection of FME transformation tools.

Overview Window

The overview window displays a view of the entire workspace, and highlights the outline of the current canvas window display upon it.

Additional viewing options

Additional panes can display lists of feature and attribute connections.

More Information

Creating Workspaces

About the Workbench canvas.

About the Workbench Navigator.

Tell me about the translation log.

Tell me how to include transformers in my workspace.

Tell me how to manage feature type connections.

Tell me how to manage attribute connections.

Workbench Keyboard Shortcuts

General	Shortcut
Cut	Ctrl+X
Copy	Ctrl+C
Delete	Delete key
Open	Ctrl+O
Paste	Ctrl+V
Prompt and Run Translation	Ctrl+R
Redo	Ctrl+Y
Run Translation	F5
Run Translation with Inspection	Shift + F5
Save	Ctrl+S
Select All	Ctrl+A
Undo	Ctrl+Z

Workspace	Shortcut
Close current tab	Ctrl + F4
Change to next and previous tabs, respectively	Ctrl + Tab Ctrl + Shift + Tab
Select Tab number	Ctrl + number_key
Duplicate transformer	Using Quick Add: If you add a transformer and want to add the same transformer again, press the slash "/" key. The Quick Add box will appear showing the last selected transformer. Press Return to include it, then Return again to edit its parameters.
Run Translation	F5
Prompt and Run Translation	Ctrl+R
Run Translation with Inspection	Shift + F5
Save workspace	Ctrl+S

Workspace	Shortcut
Search: Workspace if the focus is in the Workspace or the Navigator; Transformer if the focus is in the Transformer Gallery; Log if the focus is in the Log window	Ctrl + F F3
Toggle Inspection Points	F9
Zoom-in	Ctrl +
Zoom-out	Ctrl -
Zoom 100%	Ctrl 0
Zoom in and out	Ctrl + scroll wheel

Workbench Canvas

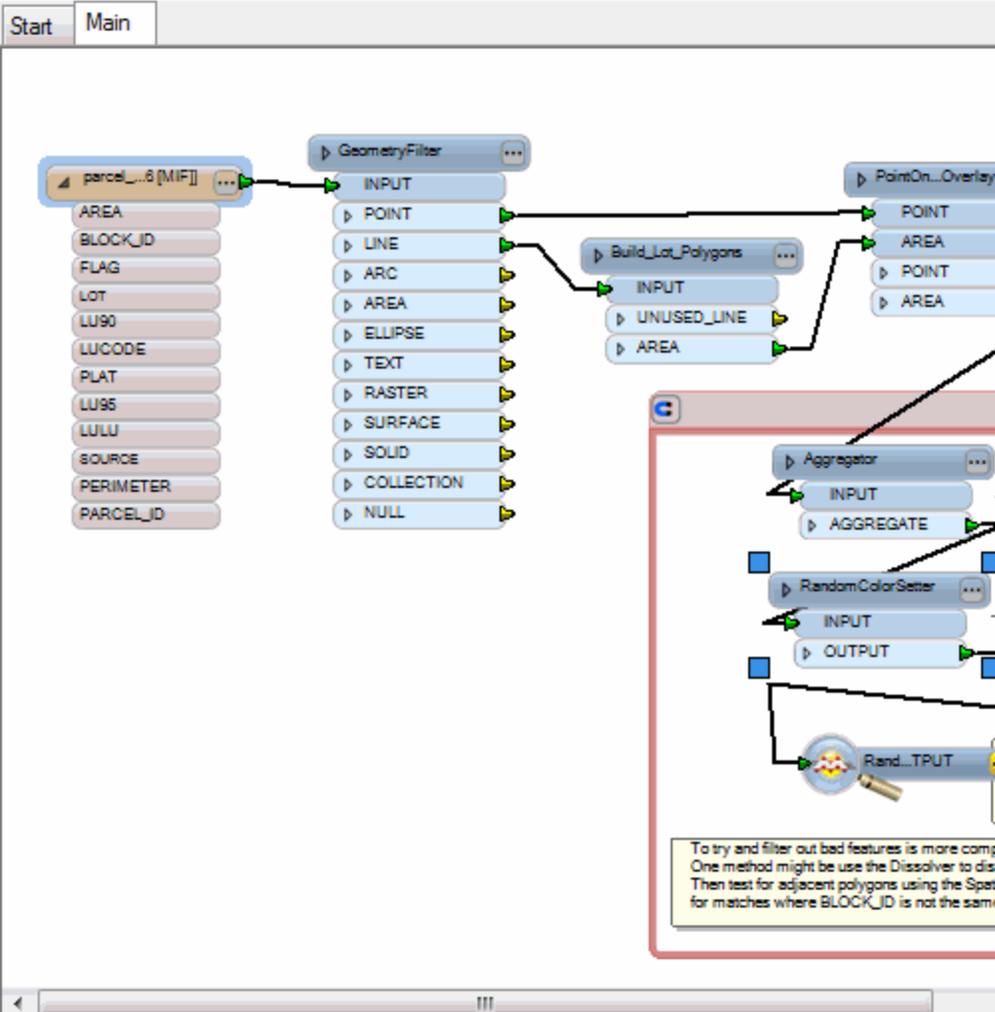
The main Workbench canvas displays the connections between the readers (source data), transformers, and writers (destination data). By default, it is the largest area of the workspace on the right.

After you save what you are working on, the workspace name is displayed in the window's title bar. Here's what an example workspace looks like, with the addition of a few customizations like transformers and annotations:



Navigator

- parcel_L26 [MIF]
- parcels [GEOJSON]
 - Coordinate System: <not set>
 - Parameters
 - Destination GeoJSON File
 - Output Character Set: UTF-8
 - Byte Order Marker: No
 - Write 'null' for attributes
 - Advanced
 - Feature Types
 - parcel_L26
 - Parameters
 - Attributes
 - AREA [float]
 - BLOCK_ID [string]
 - FLAG [float]
 - LOT [string]
 - LU90 [float]
 - LU95 [float]
 - LUCODE [float]
 - LULU [float]
 - PARCEL_ID [string]
 - PERIMETER [float]
 - PLAT [string]
 - SOURCE [string]
 - Transformers
 - Aggregator [Aggregator]
 - Build_Lot_Polygons [AreaBuilder]
 - GeometryFilter [GeometryFilter]
 - PointOnAreaOverlayer [PointOnAreaOverlayer]
 - RandomColorSetter [RandomColorSetter]
 - RandomColorSetter_OUTPUT [RandomColorSetter]
 - Tester [Tester]
 - Bookmarks
 - Advanced Task
 - Parameters
 - Published Parameters
 - Private Parameters
 - Workspace Resources
 - Workspace Properties
 - Workspace Title: <not set>



Transformer Description

RandomColorSetter

Sets a random color for each incoming feature.

Parameters

[Color to Set](#)

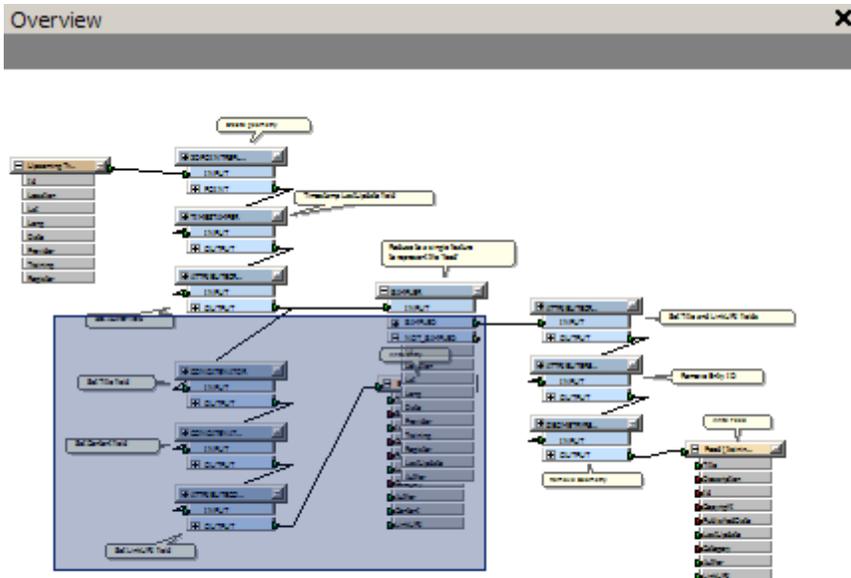
Overview Window

View > Windows > Overview

The Overview window displays a mirrored view of the entire workspace.

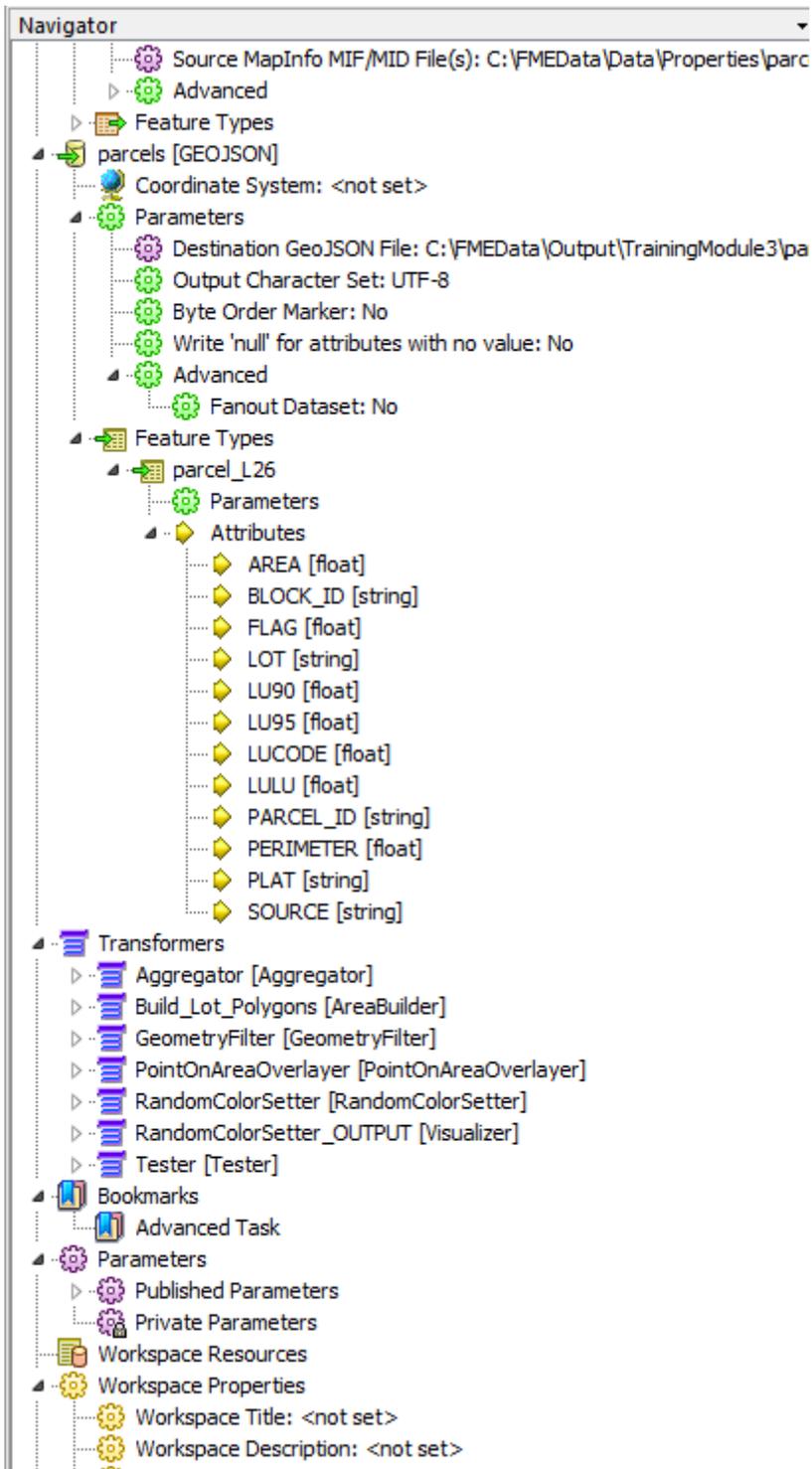
The outline of the current canvas window is highlighted, so if you have a very large workspace, you can quickly get your bearings in relation to the rest of the workspace.

If you zoom or pan the canvas window, the overview window also compensates its view; if you move anything on the canvas, it is mirrored in the Overview window.



Working in the Navigator

The Navigator displays an overview of the source (reader) and destination (writer) information, bookmarks, workspace parameters and transformers. In general, it is a hierarchical view of the information in the graphical pane. You can adjust most dataset, feature type, and attribute parameters from here, as well as directly on the canvas.



One advantage to managing information in this window is that if you have a very large workspace, sometimes it is easier to view all the components in list form. You can:

- collapse and expand information much like in a Windows Explorer environment;
- see at a high level what the workspace contains, at a detailed level when you click to open the parameters;
- tell if any information is missing (for example, an incomplete transformer);
- access functions for all entries in the Navigator by clicking on the parameter, and then right-clicking to display a command menu.

By default, a Workspace Search link will also display in the Navigator.



Workspace Resources

Delete this text and replace it with your own content.

Adding a Reader as a Resource

When you add a reader as a resource, you are inserting a reference to a dataset to be used in the workspace. This reader will not perform any actual data reading, except when up-to-date schema is required at runtime. When required, schema may be requested from that reader.

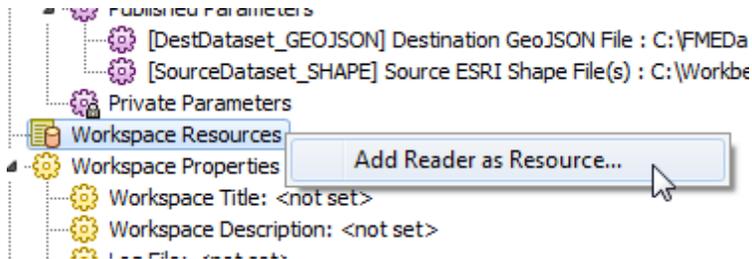
Currently it is only used by the writers that are configured to use Dynamic Parameters.

In the dynamic mode, a writer will extract schema information from one or more of the specified readers or reader resources to use during a translation. This feature is particularly useful when a writer needs to get the schema and features from several different datasets.

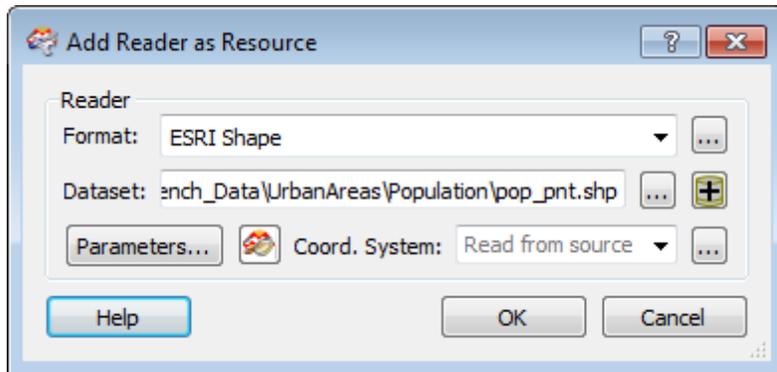
The difference between a Reader and a Reader Resource is that adding a Reader will also add all the associated feature types – a reader resource can be used as a source for schema without actually providing any feature types.

Setting up the Reader Resource

Select Readers > Add Reader as Resource or right-click Workspace Resources in the Navigator:



The function in Workbench is similar to adding a workspace Reader. You will need to specify the format and dataset of the resource you want to add to Workbench, and specify any additional format parameters.



After you click OK, the log file will display the processing statistics associated with the selected dataset, and you will see a Translation SUCCEEDED message, indicating that Workbench has successfully processed the dataset.

You will see an additional Workspace Resources parameter in the Navigator window:

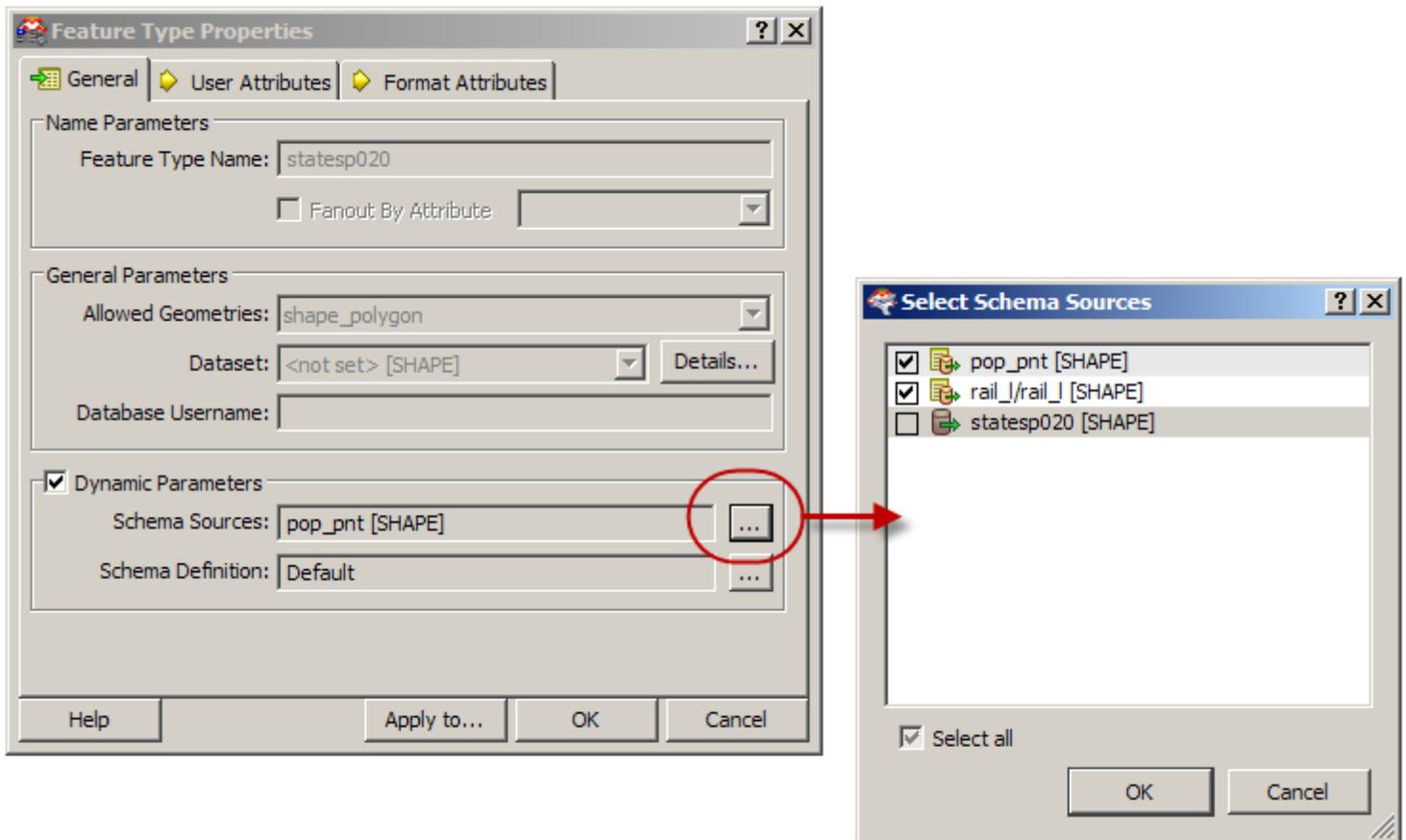


This Reader will remain in the list as a resource of the workspace.

Setting up the Writer to use the Reader Resource

When you want to use the Reader in a translation, open the Writer Feature Type Properties. Clicking Dynamic Parameters enables the Schema Sources and Schema Definition fields.

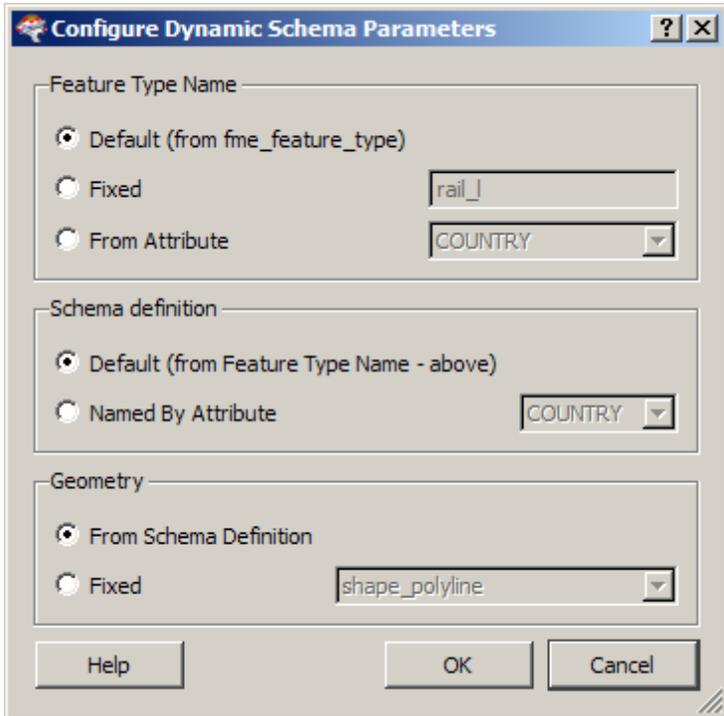
Click the Browse button beside the Schema Sources field to select from a list of existing schema readers. Only the readers that you have selected will be used at runtime.



Changing the Dynamic Schema Parameters (Advanced FME)

In most cases, you can use the default Dynamic Schema parameters.

However, the dynamic writer can operate with a few variations on the schema definition. These are feature type fanouts, more complex schema reader definitions and defining the destination feature geometry.



Feature Type Name

This controls the name of the destination feature type that is written. *Default* uses either the source schema or the schema reader feature type name. *Fixed* uses the name of the feature type in the workspace. *From Attribute* has the same effect as the dataset fanout with a new feature type for each value of the specified attribute.

Schema Definition

If the feature types in the Schema Reader do not match the `fme_feature_type`, then you can set an attribute that defines the name of the schema reader feature type to be used for the schema definitions.

Geometry

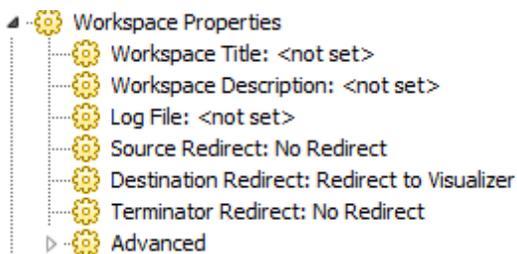
Some formats, e.g. ESRI Shape, have a fixed geometry. This option defines where the definition for the geometry is derived from.

See also

About Dynamic Workspaces

Workspace Properties

Basic Workspace Properties are displayed in the Navigator pane:



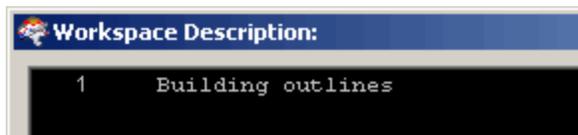
Workspace Title

Double-click to add or edit the workspace name and/or location. When you click OK, the changes will reflect in the Workbench title bar.

Workspace Description

This setting allows you to enter a description of what the workspace does. Where this setting is particularly important is when you are using the workspace via an FME Server – because it provides a mechanism for a web page to describe what the process is intended to carry out.

Double click this parameter to display the settings. When you click the Browse button, a command-line window will appear. Type the description for your workspace. For example:



Then click the OK button at the bottom to dismiss the window.

The description you enter will appear in the Workspace Description with the proper character encodings for spaces, etc. Click OK again to dismiss the window and the description will appear on the parameter line.

Log File

The log file parameter points to the location where a log file will be created when the workspace translation is complete.

Source and Destination Redirect

See Redirecting the Source and Destination.

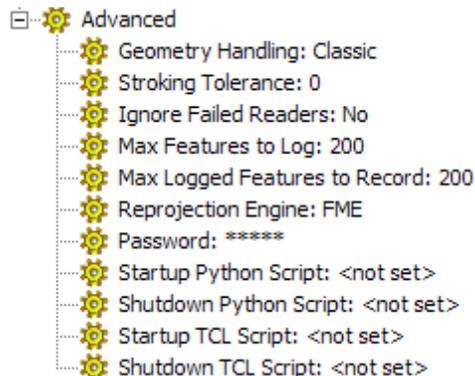
Terminator Redirect

The Terminator transformer is used to detect non-valid situations. When a feature is directed to this transformer, the translation immediately stops and displays an error message (by default, *Translation Terminated*). If you are programming or debugging, you would usually disable the connections to the Terminator and then add a Visualizer to verify if the features are real errors. In production mode, you then have to re-enable these connections and delete the Visualizer.

This option allows you to automatically redirect the features that enter the Terminator to a Visualizer, without having to modify the workspace. When this option is activated, all the features that enter a Terminator are redirected to a Visualizer and the translation continues without stopping. A message is added to the log file to indicate that some features were redirected to a Visualizer.

Advanced Parameters

See the topics under Advanced Parameters.



Geometry Handling

In FME versions previous to FME 2009, *Enhanced Geometry* was referred to as *Rich Geometry*. It has been renamed to more accurately reflect its function in FME.

It is important to understand that, for the most part, the geometry model will be transparent to users. You will not have to change the way you use FME or set up a workspace. The only visible difference should be improved output (if there was room for improvement).

Workspace Navigator > Advanced Settings > Geometry Handling. Valid values are Enhanced and Classic, and the default is Classic.

The addition of enhanced geometry model support allows lines and polygons containing arcs to be maintained, rather than stroked or the geometry split up into multiple segments. It also provides the ability to truly hold measures.

A global directive (in the form of an Advanced setting in Workbench) sets the usage of enhanced geometry in readers/writers and factories/functions/transformers that were created using "classic" geometry. All new readers/writers and factories/functions/transformers use enhanced geometry and are not affected by this new global directive. It also only affects readers that are able to operate in both classic and enhanced geometry modes. If a reader only supports enhanced geometry or only supports classic geometry, then it is not affected by the parameter. In addition, the keyword does not guarantee that all features output by the reader will be enhanced (e.g., SHAPE); however wherever it is beneficial, enhanced geometry will be used. This is also true for some functions and factories as this directive does not tell the factory what type of geometry it should be outputting. Older workspaces/mapping files/FME Objects applications will continue to use classic geometry (for backwards compatibility).

For writers that support enhanced geometry, writing enhanced geometry is handled automatically. Features with either classic or enhanced geometry can be sent to the same writer. As long as the feature sent to the writer contained enhanced geometry, the writer will write the feature using enhanced geometry. In other words, to write enhanced geometry, you must have read enhanced geometry from a reader and routed it to the output. Or, you could have created enhanced geometry in one of the transformers that supports it (say by putting arcs and lines into a PolygonBuilder/PolygonFactory), and routing that output to the writer.

When the Geometry Handling parameter in Workbench is changed between Enhanced and Classic, some Readers and Transformers will change their behavior. If a Reader or Transformer's behavior is affected, this is noted in its documentation, either in the Transformer help, or in the FME Readers and Writers manual.

Additional Information

- See FME Options > Workspace.
- See the fmpedia article on the Geometry Model for more information on geometry handling.
- For more information on the global directive that sets the usage of enhanced geometry in readers and writers (as well as functions, factories and transformers), see the FME Fundamentals on-line help, in FME Configuration > Geometry Handling. (The FME Fundamentals help is available as a link from the help menu.)

Stroking Tolerance

Workspace Navigator > Advanced > Stroking Tolerance

You can set a Stroking Tolerance whenever you want arcs to be stroked into lines, making sure the distance between the resulting line and the true mathematical arc is never more than the value specified. You might want to use this setting when, for example, you are translating from a format that supports arcs to a format that does not support arcs.

This option sets the stroking tolerance for an individual workspace. In FME Workspace Options, you can also set a default to be used for all workspaces. If the two settings ever conflict, the individual workspace setting will take precedence.

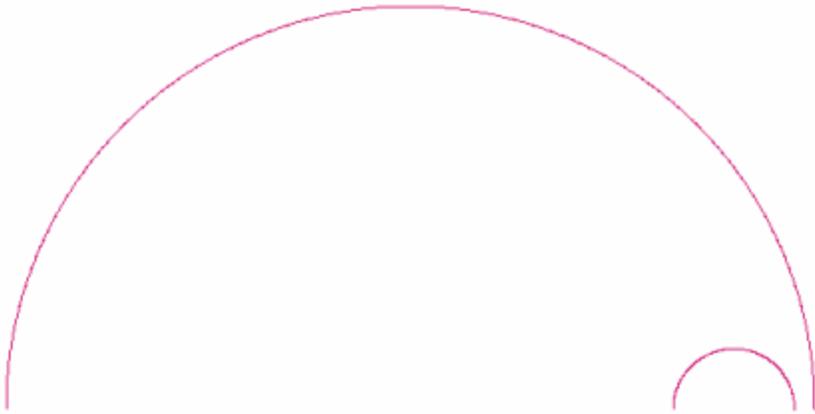
If you choose to leave this setting at the default of 0, you should not notice any difference in your workspace.

Example: Setting the Stroking Tolerance for a Geodatabase to MIF Translation

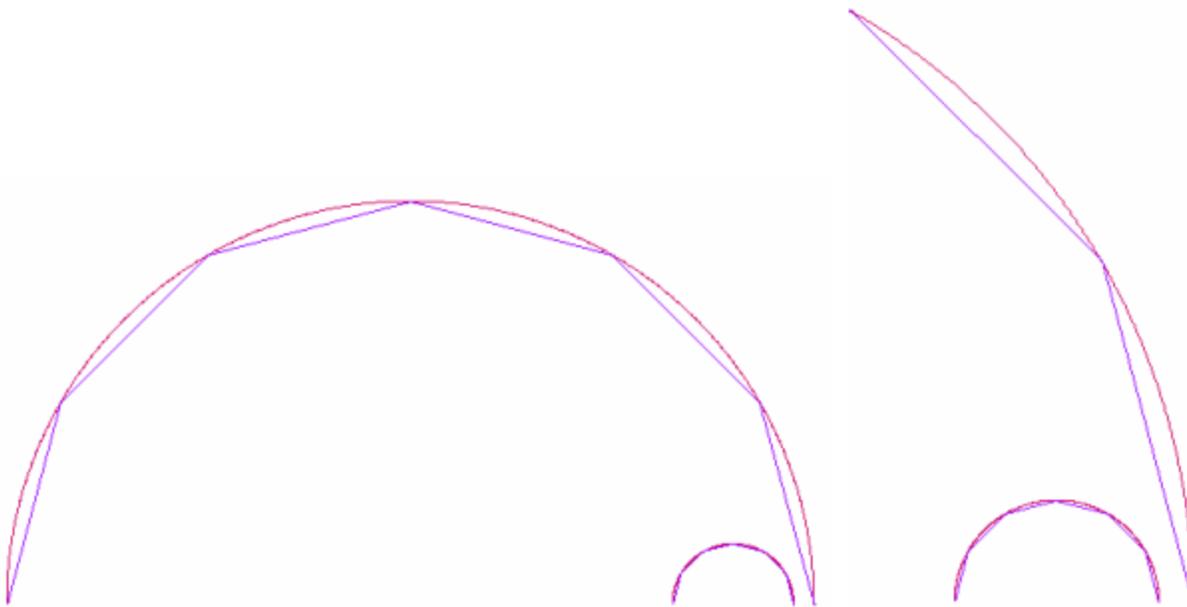
If you want to read an arc feature from a format that supports arcs – such as Geodatabase – and write out this feature to a format that does not support arcs – such as MIF – the arc would need to be vectorized. The value of this setting controls the accuracy of the stroked arc when written out, when compared to the original arc.

If this value is smaller than or equal to 0, arcs are stroked based on the sweep angle of the arcs, and the edges of the resulting lines are spaced out evenly.

Consider two arcs with the same sweep angle of 180 degrees and primary radius of 10 and 1.5 units respectively.

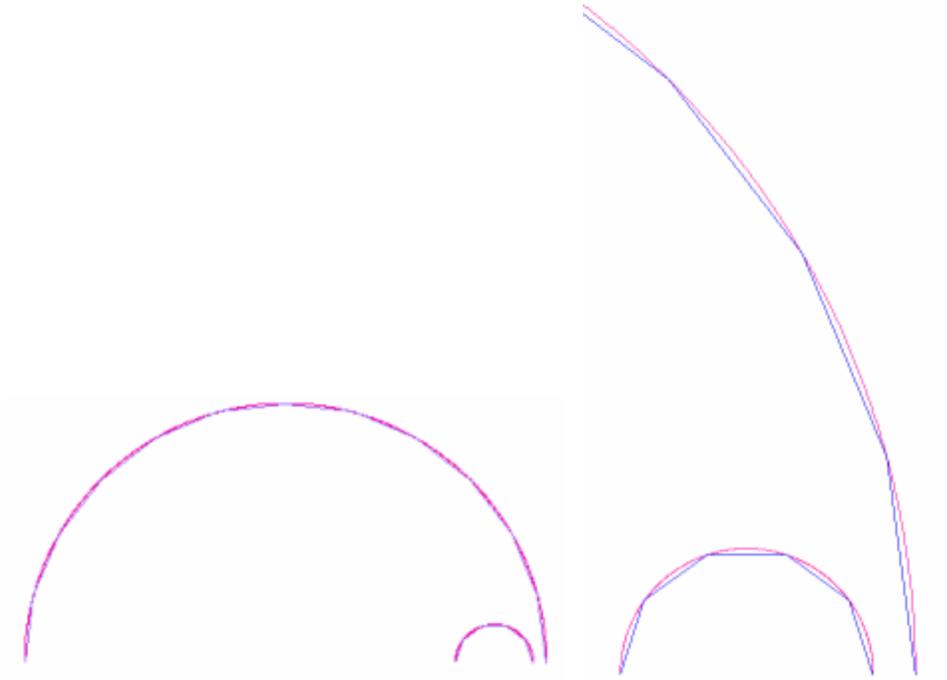


a. Stroking the arcs using 30 degrees per edge will result in the following:



The resulting line from stroking the smaller arc is more accurate than the line from the bigger arc.

b. Stroking the arcs by setting the Stroking Tolerance to 0.1 units will result in the following:



Both resulting lines from the smaller and bigger arcs have the same accuracy, that is, the distance from the mathematical arc is never more than 0.1 units.

Ignore Failed Readers

Workspace Settings > Advanced > Ignore Failed Readers

This setting tells FME whether or not to continue a translation when reading a dataset fails. For example, if the wrong password is entered so that FME cannot read from a database, should the translation continue with any other datasets that FME can read from?

Double-click this parameter to display the dialog. Choose **Yes** if you have specified more than one reader, and you do not want the translation to stop if one of the readers fails or is missing. The default is **No**.

Max Features to Log

Workspace Settings > Advanced > Max Features to Log

To prevent the accidental creation of extremely large log files, FME controls the number of features that may be logged during an FME session.

FME logs features found to be invalid during translation so, for example, if a bad input data file is encountered, or if you inadvertently route lines through a cell feature, FME can literally log thousands of features and a log file several megabytes in size may result.

The value of this setting imposes a limit on the number of features that can be logged during a translation, which by default is 20. This prevents excessively large log files from being inadvertently generated. Any features beyond the set limit are not logged at all. A statistic is output at the end of the log that states the number of suppressed features that were not logged.

Removing the Limit

To remove this limit and log all features, change this setting to -1.

Note: This setting takes precedence over the Maximum Logged Features parameter in the Logger transformer.

Max Logged Features to Record

Workspace Settings > Advanced > Max Logged Features to Record

Any feature that is recorded in the log window (or file) will also get written to an FFS log dataset. This dataset takes the same name as the log file but with “_log” added and an FFS extension.

This setting determines the number of features that will get written to this log dataset.

It’s important to be aware that the log feature maximums apply an overall cap to accumulated log totals. For example, three Logger transformers each have a limit of 10 features to log, giving a theoretical maximum of 30 features. However, the “Max Features to Log” setting will restrict the overall total; e.g. if set to 10, then it will restrict the theoretical maximum from 30 to 10.

You should also note that these settings affect not just manually logged features – using the Logger for example – but also features that are logged for another reason. For example, text features rejected by the AreaBuilder are logged in this way, and their logging affected by these logging maximums.

Reprojection Engine

Workspace Settings > Advanced > Reprojection Engine

Different GIS applications have slightly different algorithms for reprojecting data between different coordinate systems. To ensure that the data FME writes matches exactly to your existing data, this setting allows you to use the reprojection engine from a different application.

Note: To use this feature, you must have a supported GIS application installed on your system (e.g., ESRI ArcGIS®).

Password

You can password-protect workspaces, custom formats and custom transformers. Double-click the Password parameter and enter a case-sensitive password in the dialog that appears. Each time you try to open the workspace, you will be prompted for the password. If you save the workspace under a different name, the password protection will also be duplicated.

WARNING: When you create a password-protected file, make sure you use a password that you can easily remember. If you lose the password, you cannot open or gain access to the file.

Startup and Shutdown Python Scripts

The ability to run a TCL script before or after an FME translation was available in FME 2006 GB. For FME 2007, Python scripting ability has been added and these scripts are exposed as settings in the Workspace Settings dialog.

Potential uses of such scripts are:

- To check a database connection before running the translation.
- To move data prior to or after the translation
- To write the translation results to a custom log or e-mail them to an administrator

For more information, see FME_BEGIN_PYTHON and FME_END_PYTHON in the FME Fundamentals manual. For examples, see fmepedia.

Startup Python Script

This setting allows a Python script file to execute just prior to the start of translation. The script is executed after the mapping file has been completely parsed, and after the logfile has been opened, but before any of the readers or writers have begun to do their processing.

Note: FME will abort the translation if the execution of FME_BEGIN_PYTHON scripts fails. If, for some reason, this behavior is undesirable and you want to continue a translation even if the execution fails, you can use Python's exception handling to trap errors, allowing FME to continue with the translation.

Shutdown Python Script

This setting allows a Python script file to execute just after translation has completed, either successfully or prematurely due to an error being encountered.

If the translation ended due to an error, the script is executed after all cleanup is done, all reader and writers are shut down, and the logfile has been closed. If the translation was successful, the script is executed after all the the readers and writers have finished their work, and the logfile has been closed.

The script has access to a number of Python global variables that contain statistics and other information about the translation.

Startup and Shutdown Tcl Scripts

The ability to run a Tcl script has been added and these scripts exposed as settings in the Workspace Settings dialog.

Potential uses of such scripts are:

- To check a database connection before running the translation.
- To move data prior to or after the translation
- To write the translation results to a custom log or e-mail them to an administrator

For more information, see FME_BEGIN_TCL and FME_END_TCL in the FME Fundamentals manual.

Startup Tcl Script

This setting allows a Tcl script file to execute just prior to the start of translation. The script is executed after the mapping file has been completely parsed, and after the logfile has been opened, but before any of the readers or writers have begun to do their processing.

FME will abort the translation if the execution of FME_BEGIN_TCL scripts fails. If, for some reason, this behavior is undesirable and you want to continue a translation even if the execution fails, you can add a Tcl catch command in your Tcl script – this will catch any errors and FME will continue with the translation.

Shutdown Tcl Script

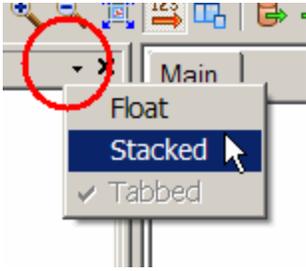
This setting allows a Tcl script file to execute just after translation has completed, either successfully or prematurely due to an error being encountered.

If the translation ended due to an error, the script is executed after all cleanup is done, all reader and writers are shut down, and the logfile has been closed. If the translation was successful, the script is executed after all the the readers and writers have finished their work, and the logfile has been closed.

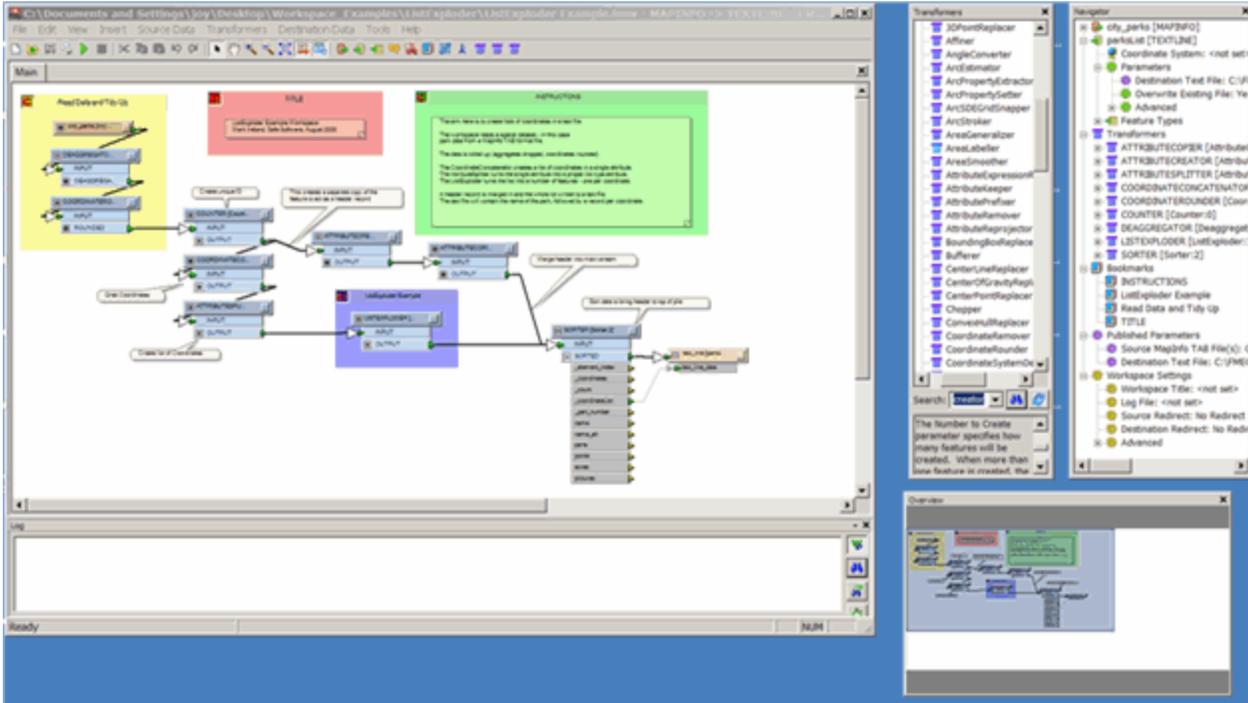
Window Management

You can customize the layout of all areas of the Workbench interface. Click View > Windows to see a list of all the panes that you can enable or disable.

The small arrow (circled below) in the right-hand corner of each pane lets you choose how you want your display (Floating, Stacked or Tabbed). Click the arrow to display the menu.



By default, the Transformer and Navigator panes are tabbed, meaning that they are displayed on top of each other and you can click their tabs to toggle between them. If you choose "Float" you can detach a pane from a fixed position and place it where you want by clicking on the frame of the window and dragging it into a new position. The windows can even float outside of the main Workbench window.



When you have multiple monitors on your system, you can place different panes on a different monitor, leaving the main monitor free to maximize the workspace canvas.

You can re-dock panes by dragging them onto the Workbench window frame. Windows can be docked to either the left, right, upper or lower boundaries of the Workbench frame. When two or more windows are docked in the same location there is the option to arrange them either stacked or tabbed.

Menus and Toolbar

Menu Bar

The pull-down menus on the top of the Workbench window contain commands that will affect the entire workspace. Individual command menus within the graphical interface (displayed by clicking the right mouse button after making a selection) are applicable to separate workspace components.

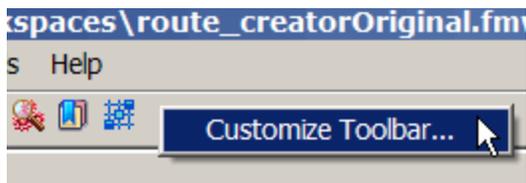
Toolbar

The toolbar gives you quick access to various commands. To see what a toolbar button does without actually selecting it, move your cursor over the button to view a short description of its function.

- <Separator>
-  Action Arrow
-  Add Reader
-  Add Writer
-  Add Writer Feature Type
-  Align Bottom
-  Align Centers Horizontally
-  Align Centers Vertically
-  Align Left
-  Align Right
-  Align Top
-  Attribute Connections
-  Auto Layout
-  Collapse All
-  Copy
-  Cut
-  Display Feature Counts
-  Download from FME Server
-  Enable/Disable Drag and Insert Transformer
-  Expand All
-  Export as .fme
-  Feature Inspector
-  Feature Type Connections
-  Insert Annotation
-  Insert Bookmark
-  Insert Constant
-  Insert Transformer Input
-  Insert Transformer Loop
-  Insert Transformer Output
-  Insert Visualizer
-  Launch Viewer
-  New Workspace
-  Open
-  Overview Window
-  Pan
-  Paste
-  Pause Translation
-  Print
-  Prompt and Run Translation
-  Publish to FME Server
-  Redirect to Visualizer
-  Redo
-  Republish to FME Server
-  Run or Resume Translation
-  Run translation with inspection
-  Save
-  Spread Horizontally
-  Spread Vertically
-  Stop Translation
-  Undo
- Zoom Extents
- Zoom In
- Zoom Out
- Transformers

Customizing the Toolbar

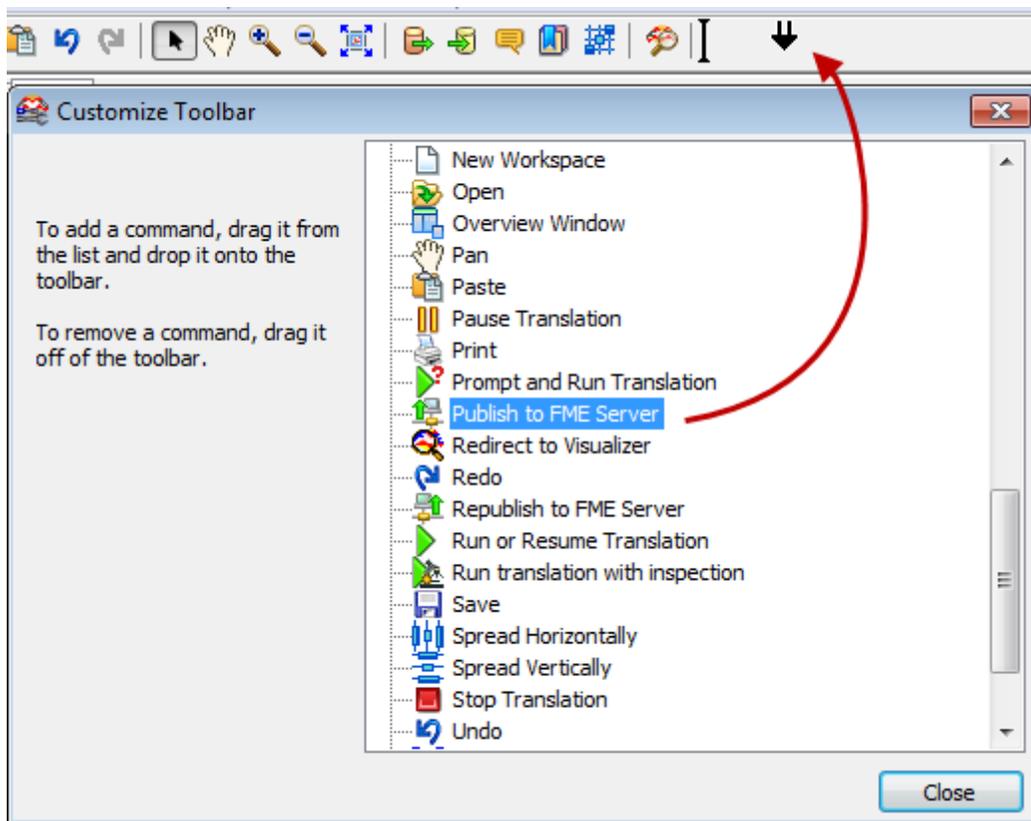
Right-click on the toolbar and click Customize Toolbar.



This displays the Customize Toolbar dialog.

From here, you can customize your toolbar by adding or removing tools and transformer names. Adding a transformer to the toolbar enables one-click workspace placement.

In the Customize Toolbar dialog, click on the tool or transformer name. Drag the icon to the toolbar, and place it. (Note that you can also add transformers to the toolbar by dragging the transformer from the Transformer Gallery to the toolbar.)



To remove a tool, click on it, and drag it off the toolbar. You will see a brief **X** in place of the cursor when the tool is removed.

Status Bar

The area at the bottom left-hand side of the Workbench window displays progress information, as well as short descriptions of selected menu items or toolbar buttons.

Quick Navigation

To quickly move through a workspace, use keyboard navigation.

- The arrow keys will move you to different parts of a workspace, including links, and the Enter key will open properties for editing.
- Press the Esc key to quickly cancel a pan or zoom mode.
- When you are zoomed out in a workspace, you can hover your cursor over constants and annotations to display tooltips that show their text.
- Press and drag the middle mouse button (this is the mouse wheel on most mice) to pan around the workspace.
- Hold down the Ctrl key and roll the mouse wheel to zoom in and out.

Menu Commands

File

Command	Keyboard Shortcut	Function
New		Opens the Create Workspace dialog, which provides different options for creating a new workspace.
Open	Ctrl + O	Opens a file browser so you can pick an .fmw (FME workspace) file.
Close		Closes the current workspace.
Publish/Republish to FME Server		Publishes a workspace to an FME Server. Note that you must have an FME Server configured as part of your system.
Download from FME Server		Downloads a workspace from an FME Server. Note that you must have an FME Server configured as part of your system.
Save	Ctrl + S	Saves the current workspace.
Save As		Saves the current workspace to a different name. You can also save a simple workspace to a mapping file (.fme) for use with the FME Quick Translator.
Save As Template		Saves the current workspace as a template (.fmwt).
Export as Custom Transformer		Exports the workspace as a custom transformer.
Export As Custom Format		Exports the workspace as a custom transformer.
Page Setup		Sets page layout for print.
Print/Print Preview		Prints the workspace.
Run Translation	F5	Runs the workspace with existing parameters.
Prompt and Run Translation	Ctrl + R	Runs the translation and prompts you to change parameters (for example, if you want to use different source data).
Batch Deploy		Runs the translation in batch mode.
Send To		Opens your e-mail client with the current workspace attached to a new e-mail.

Edit

Command	Keyboard Shortcut	Function
Undo	Ctrl + Z	Reverses the most recent action in a workspace.
Redo	Ctrl + Y	Repeats the most recent action that you reversed in a workspace.
Cut	Ctrl + X	Cuts a selection and sends it to the clipboard.
Copy	Ctrl + C	Copies a selection and sends it to the clipboard.
Paste	Ctrl + V	Pastes a selection from the clipboard.
Delete	Del (key)	Deletes a selection.
Bring to Front/Send to Back		Brings objects forward on the canvas or sends them behind other objects.
Select All	Ctrl + A	Selects all content on the canvas.

View

Command	Keyboard Shortcut	Function
Windows		Shows/hides various Workbench windows.
Show Grid		Displays the grid set in grid settings.
Snap to Grid		Snaps object to the grid.
Grid Settings		Sets the grid pattern.
Page Breaks		Pastes a selection from the clipboard.
Expand All/Collapse All		Deletes a selection.
Annotations		Shows/hides annotations.
Summary Annotations		Shows/hides summary annotations, which display a box that contains information about the feature type or transformer.
Bookmarks		Shows/hides bookmarks.
Zoom		Sets default zoom properties.

Insert

Command	Keyboard Shortcut	Function
Constant		Supplies a destination attribute with a constant value.
Parameter Reference		Create a parameter reference that can assign the value of that parameter to an attribute in the workspace.
Annotation		Annotation
Visualizer		Adds a Visualizer transformer to the canvas.
Bookmark		Adds a new bookmark to the canvas.
Parameter		Creates a parameter that is not related to an existing workspace setting. This is the same as right-clicking on Published or Private parameters in the Navigator.
Custom Transformer		Adds a custom transformer to the workspace.
Transformer Input, Output, Loop		Adds options onto Custom Transformers.

Readers

Command	Keyboard Shortcut	Function
Add Reader		Adds a reader to the workspace.
Import Feature Type		Imports a feature type from a different dataset.
Update Feature Types		Updates feature types if the structure of the data has changed outside the workspace.
Enable/Disable Feature Types		Temporarily disables (or enables) feature types in the workspace.
Remove Feature Types		Removes a feature type.
Remove Reader		Removes a reader.
Add Reader as Resource		Inserts a reference to a dataset to be used in the workspace.

Writers

Command	Keyboard Shortcut	Function
Add Writer		Adds a writer to the workspace.
Add Feature Type		Adds a new feature type to the workspace.
Import Feature Type		Imports a feature type from a different dataset.
Update Feature Types		Updates feature types if the structure of the data has changed outside the workspace.
Enable/Disable Feature Types		Temporarily disables (or enables) feature types in the workspace.
Remove Feature Types		Removes a feature type.
Remove Writer		Removes a writer.
Move Feature Types		Moves feature types between datasets.
Redirect to Visualizer		Redirects the output to the FME Universal Viewer or the FME Data Inspector (depending on options set).

Inspection

Command	Keyboard Shortcut	Function
Run Translation with Inspection	Shift + F5	Runs the translation using the Feature Inspector, after you define an inspection point.
Toggle Inspection Point	F9	Turns the inspection point into a regular link, or turns a regular link into an inspection point.
Enable Inspection Point		Enables a disabled inspection point.
Disable Inspection Point		Disables an inspection point.
Remove all Inspection Points		Removes all defined inspection points from the workspace.
Enable all Inspection Points		Enables all disabled inspection points.
Disable all Inspection Points		Disables all inspection points.
Edit Inspection Point		Opens the Inspection Point Parameters.

Tools

Command	Keyboard Shortcut	Function
Remove Unattached		Removes any unattached objects from the workspace.
Edit Parameters		Displays all reader and writer parameters in a dialog. This is useful especially if you have a large workspace or multiple readers/writers.
Edit Header		Allows you to add format directives into the workspace (Advanced user task)
Feature Inspector		Displays the Feature Inspector window.
Purge Temporary Files		Purge temporary Workbench-related files.
Browse Coordinate Systems		Opens the Coordinate System Gallery.
Browse Readers and Writers		Opens the Reader and Writer (supported formats) Gallery.
Auto Layout		Provides layout settings for the canvas.
License Borrowing		Allows floating license holders to temporarily use another floating license.
FME Options		Sets default options.

FME Options



Appearance



Runtime



Workbench



Transformers



Workspace



Default Paths



Network



Coordinate Systems

Appearance

Select **Tools > Options** and click the Appearance icon.



Appearance

Welcome Dialog

Toggle the checkbox to enable/disable the welcome dialog that appears by default when you start Workbench.

Fonts

Canvas and Log font: Set the default font to be used on the workspace canvas, and in the translation log.

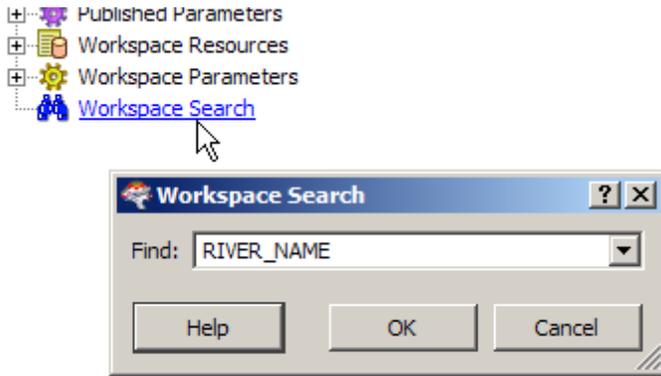
Set Sound Options

You can adjust FME sound events using the Windows Sound and Audio Devices Properties. You can set Workbench to make a distinctive sound whenever an object is cut, pasted, connected, and deleted. It will also tell you when a translation is successful or has failed. You can also set two events in the FME Universal Viewer: when a data load is complete, and when a redraw is performed (but only if the redraw is longer than 3 seconds). Note that by default all the sounds are set to Off. Follow these steps to enable them:

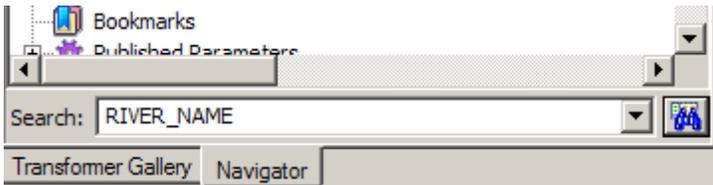
1. Click the Configure Sound Events button.
2. Scroll through the Program Events until you find FME Workbench.
3. Select the sound to associate with the applicable event.
4. Click OK.

Workspace Search Options

By default, the Workspace Search appears as a link in the Navigator. When you click the link, the Workspace Search dialog appears:



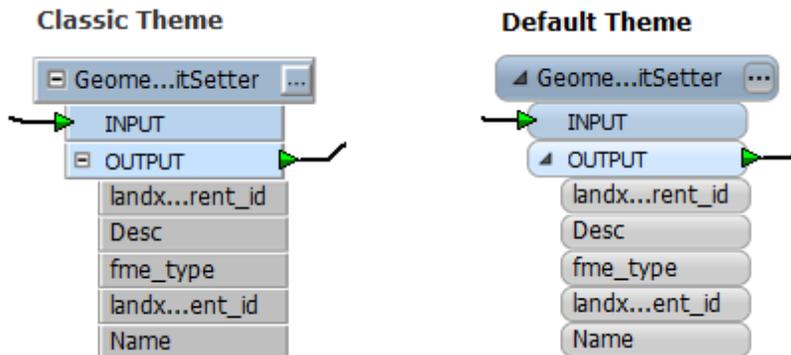
You can change the way the Workspace Search appears in Workbench. Check *Search bar embedded in Navigator window* to change the link in the Navigator to an embedded search bar at the bottom of the Navigator window.



Detailed information on using Workspace Search

Themes

You can change the way objects are displayed on the canvas. Currently, there are two different themes to display: Classic and Default:



If you select a theme from the *Current Theme* pull-down menu, the canvas will immediately reflect the change.

By default, Workbench Theme files (**.wbtheme**) are stored in the My Documents\FME\Themes folder.

Restore Defaults

Click the Restore Defaults button at the bottom of the dialog to revert this page to FME installation defaults. You will be prompted to confirm your selection.

Runtime

Select **Tools > Options** and click the Runtime icon.



Runtime

Translation Priority

Sets the CPU priority for running translations. In most cases, the default **Normal** setting will be adequate. If you regularly run large translations, you can keep the default setting, or change it to **Low**, so the translations won't dominate your CPU (and so you can do other work in the foreground while the translation is running). If you have other CPU-intensive tasks running concurrently, you may want to set the priority to **High** to make sure the translation gets its share of the CPU.

Track Usage Statistics

If you check this box, FME will transmit information about your version of Windows and how you use FME. The information collection process is completely anonymous, and your results will be automatically combined with other users' results. The resulting statistics will help us identify trends and usage patterns (for example, which formats and processing facilities are utilized more than others), which in turn will help us focus our development efforts for future versions of FME. We will not collect your name, address or any other personally identifiable information.

Log File Defaults

- **Save log file:** Save the translation log in the default workspace directory. (When you choose "Save to file" in the log pane, the default location will be the location you choose under Default Workspace Directory, below. The default filename will be *translation_log.txt*.)
- **Append to log file:** Appends log results to the previously generated log, instead of overwriting the file.
- **Log Timestamp and Debugging Information:** You can choose to add these additional details to your log file output.

Log Filter

Choose the type of messages that you want to view in the log file. For example, you might want the log to display only Warnings and Errors.

Workspace Defaults

Select **Tools > Options** and click the Workspace Defaults icon.



Workbench

Set General Defaults

- **Automatically save workspace before running:** Automatically saves your .fmw file after a translation, using a default name based on your source and destination formats.
- **Open up attributes when linking transformers:** By default, feature types are expanded to display attributes whenever a link is established.
- **Draw bookmarks with a filled background:** Bookmarks are shaded instead of transparent.
- **Show unexpected input dialog:** An important part of FME is being able to choose not to read certain source feature types; this can be done by removing them from the workspace. For example, if you don't want to read a certain layer in your data, you would delete the applicable feature type. The Unexpected Input Remover (UIR) is that function that discards such data. It checks the input data against the structure (or schema) defined in the Workbench or mapping file. If the data doesn't have a matching feature type, it is discarded. The results of the UIR are noted in the log pane, in two separate places. This checkbox allows you to optionally display the information in a dialog box. See this [fmpedia](#) article for more information on the UIR.
- **Allow source feature type editing:** This feature, which is unchecked by default, allows you to edit the definition of a source file.

- **Enable quick connect:** To connect elements of a workspace, you can enable this feature to click an output port, release the mouse button, and then click an input port. This provides an easy alternative to the "drag and connect" method.

Set Annotation Defaults

- **Automatically generate header annotations:** New workspaces will appear with the default Source Types, Data Flow, and Destination Types annotations that appear at the top of the workspace.
- **Use transparent annotations:** Gives annotations a transparent background.
- **Automatically create summary annotation:** Summary annotations display detailed information on feature types or transformers.

Set Save Options

- **Save recovery data:** Specify whether (and how often) you want Workbench to write a recovery file for your workspace.

Transformers

Select **Tools > Options** and click the Transformers icon.



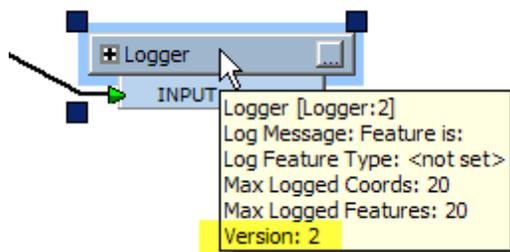
Transformers

Display Options

Show transformer version in tooltip

With new releases of FME, sometimes we also upgrade transformers to include new functionality. If you have a large workspace whose history spans multiple FME versions, it may also span multiple transformer versions.

Previous versions of transformers will work the same way they always did, but you might also find it useful to enable transformer tooltips to show the transformer version (especially if you have different versions of the same transformer). In this example, the transformer version is 2:



Truncate transformer names on: This option allows you to specify text abbreviations from the left, right or center. For example, if your text consists of frequently occurring prefixes, you may want to truncate names from either the center or the left so you can see more text at a glance.

Use Drag-and-Insert Transformers

Enables a feature that allows you to drop a transformer onto an existing connection. The defaults you choose here for the input and output ports will be the default port connections.

Quick Add Options

- Show Quick Add on first keypress: Enables the Quick Add search functionality on the Workbench canvas.
- Auto connect after Quick Add: If you select a Feature Type before initiating the Quick Add search, the transformer will connect automatically to the feature type.
- Quick Add placement follows mouse: The transformer will be placed underneath your cursor.

Workspace Defaults

Select **Tools > Options** and click the Workspace Defaults icon.



Workspace

Stroking

This advanced setting is used by FME whenever arcs need be stroked into lines to make sure the distance between resulting line and the true mathematical arc is never more than the value specified.

You can set a default tolerance here for all workspaces, or you can set it on individual workspaces through Workspace Settings > Advanced > Stroking Tolerance.

Geometry Handling

Change the default setting for Geometry Handling.

Note that if you are editing a pre-existing workspace (a workspace that you have saved at least once) and you change the value for this setting in Workbench Options dialog, the new value will not immediately apply. To affect an existing workspace, you will have to manually change the value in the Advanced Settings of the Workbench Navigator pane.

If you are working in a new workspace that you have not yet saved, or if you start Workbench and edit this setting, your changes will be applied immediately.

This behavior prevents overwriting of any values that may have already been saved with a workspace.

Default Paths

Select **Tools > Options** and click the Default Paths icon.



Default Paths

Default Workspace Folder

Sets the default location to save your workspaces. You can choose to always save to the source data location, a custom location, or a *My FME Workspaces* subfolder in *My Documents*.

When you create a workspace, FME suggests a default workspace name based on the formats used in the workspace. For example, if your workspace is set up to read from an ESRI Shapefile and write to both an Access MDB file and AutoCAD drawing, then **shape2mdb_dwg.fmw** would be the suggested default name.

Shared Resource Directories

A shared resource is any FME file that has been made available for other users to use. These files are:

- workspaces (*.fmw)
- mapping files (*.fme)
- custom transformers (*.fmx)
- custom formats (*.fds)
- custom coordinate systems (*.fme is best, although most file formats will work)
- transformer categories (*.fmxmlist)

This option is especially useful for workgroups. For instance, if an entire workgroup uses just a few custom coordinate system definitions, keeping these definitions in one place means that everyone doesn't have to have a copy. Then, whenever any of the definitions are updated, the entire group automatically has access to the new version.

1. Locate the directory that you want to share, and create subdirectories with these names (or, FME will create them automatically for you during step 5):
 - Transformers
 - Formats
 - Coordinate Systems

NOTE: Your system administrator may have to set up permissions and subdirectories for this folder.

2. From Workbench, select **Options** from the **Tools** menu and click the Default Paths icon.
3. Click the Add  button.
4. Browse to the folder that you want to open for sharing.
5. Click the Open button. If the subdirectories in step 1 do not exist, FME will automatically create them.

The directory name will appear in the list.

Every time you start FME, it will scan the folders for new or updated files.

NOTE: Any locally stored formats, transformers and coordinate systems take precedence over any shared directories. If there is a naming conflict, for example, if you have a transformer with the same name as a transformer in a shared directory, FME will always use the transformer that is stored locally.

FME Server Shared Directories

See FME Server Shared Directories.

Removing Shared Directories

Select the directory name and click the Delete  button.

Coordinate Systems

Select Tools > FME Options and click the Coordinate Systems icon.



Coordinate Systems

How to determine which grid shift files are installed

A number of grid shift files are included when you install FME. For details, see Included Grid Shift Files.

How to maintain and edit grid shift files

This option allows you to maintain and edit grid files. You can manage the grid files by selecting a file and clicking the Edit button (or, simply double-click the file). The dialog that appears lists one or more grid shift files:

- **Add button:** Opens a file browser so you can select a new file and add it to the list.
- **Remove button:** Removes a selected file.

Note: We recommend that you make sure that the grid files you need are in the list, and that you remove any files that you do not need. See *NAD27/NAD83 Datum Shifts in U.S. and Canada* on [fmepedia](#) for information on why it is important to configure FME to use the correct files.

- **Fallback Datum:** Select the datum to use if the selected grid file does not cover the area of your input data. Fallback datums are listed in individual .gdc files, which you can open with any text editor.
- **Move Up and Move Down buttons:** Note that these buttons will rearrange the list, but they do not determine the order of precedence of the grid files.

Click OK to apply changes, and Cancel to discard changes.

WARNING!

If you add a file, you are not copying the file – you are only pointing to the file's location. Thus, if you delete a file from its original location, the entry you create here will point to a nonexistent file. Safe Software recommends that you copy each grid file to FME's Reproject\GridData subdirectory before adding any files to your configuration.

How to add a grid shift file to FME

Follow these steps to install a grid file so that FME will recognize the file:

1. Copy the new file to FME's Reproject\GridData subdirectory.
2. Select Tools > FME Options and click the Coordinate Systems icon.
3. Select the applicable datum shift and click the Edit button.
4. A dialog displays the files already recognized by FME for the applicable datum shift. To include new files, click the Add button, browse to the applicable directory and find the file in Reproject\GridData subdirectory.
5. Select the file and click OK.

More Information

For additional information on grid shift files, see the Coordinate Systems section of this help file, or view these specific topics:

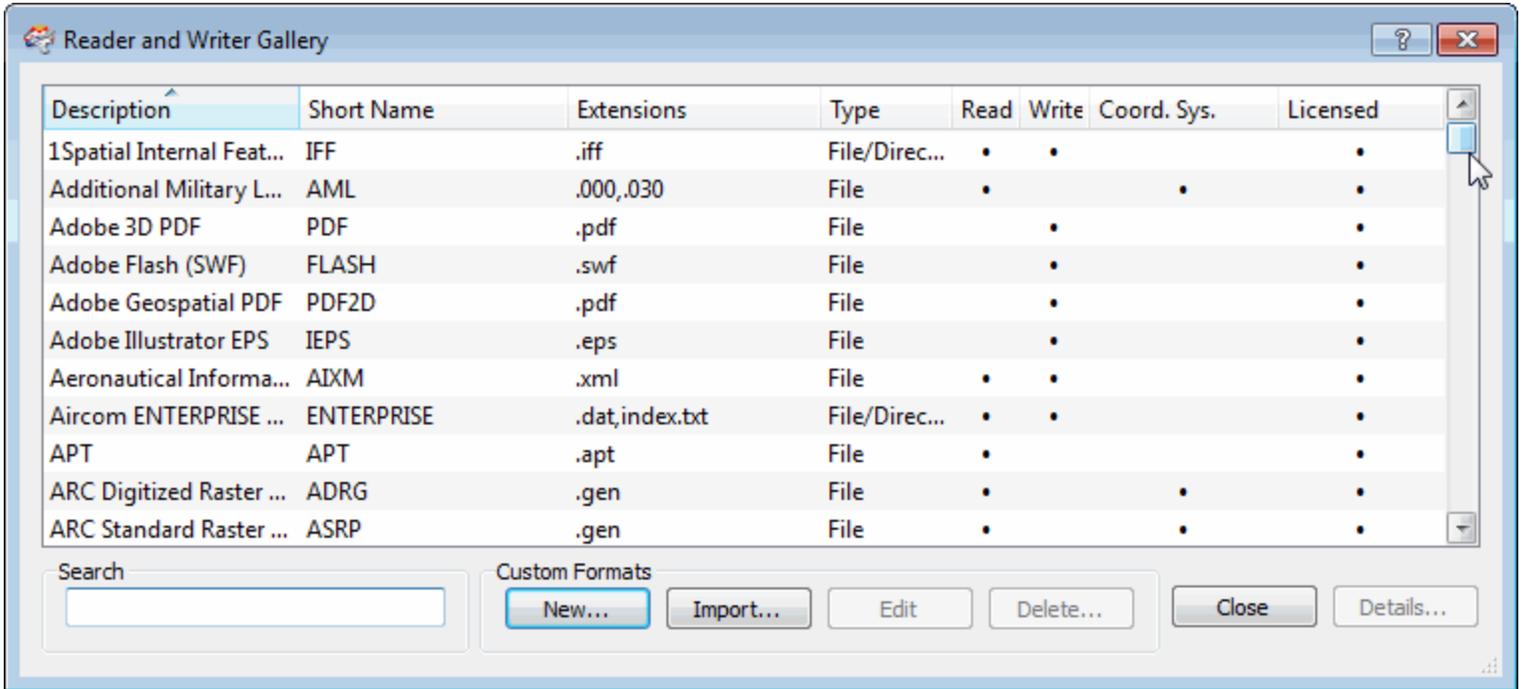
How FME Handles Grid Files

Included Grid Shift Files

About the Reader and Writer Gallery

Tools > Browse Readers and Writers

The Reader and Writer Gallery lists all formats supported by FME.



It includes the following information:

Column	Definition
Description	Full format name
Short Name	Common abbreviation for the format
Extensions	Lists file extension(s) associated with the format
Type	Indicates whether the format is a file- or directory-based format
Read/Write	Indicates whether the format is a reader, a writer, or both
Coordinate System	Indicates whether a custom coordinate system is associated with the format
Licensed	Some formats require that you obtain special licensing, or specific versions of FME. This column indicates whether your FME is licensed to read/write the format
Search	Enter a keyword or a partial text string to filter results.
Custom Formats	<ul style="list-style-type: none"> • Create a new format using the Custom Format wizard. • Import a custom format and add it to the gallery. • Edit an existing custom format.
Close	Close the gallery.
Details	Click on a format, then click Details to display the technical documentation for the format.

Removing Unattached Objects

To quickly remove unattached objects:

- Select Tools > Remove Unattached. A list of unattached objects appears.
- For large lists, use the search filter to display

- Use the checkboxes to delete the selected objects, and click OK.

Edit Parameters

Choose **Edit Parameters** from the Tools menu.

A dialog box will display a complete list of settings from your source and destination datasets. If you have many settings to change at once, this is the most efficient way to do it.

Edit Header

Note: This is for Advanced FME users.

Choose Edit Header from the Tools menu.

Each format has an associated technical chapter in the FME Readers and Writers manual. This manual lists all the directives available for the format. However, Workbench does not always contain an associated parameter for these directives.

To include a directive, you can edit the workspace header by finding the relevant header section for the dataset whose directive you need to add, and inserting it as a macro. For example:

```
DWG_1_SUPPRESS_FONT_WARNINGS NO
```

You can also add the directive as a setting in Workbench by using...

```
DEFAULT_MACRO _SUPP_FONT_DWG_1 NO
GUI_CHOICE _SUPP_FONT_DWG_1 YES%NO Suppress Font Warning:
DWG_1_SUPPRESS_FONT_WARNINGS $( _SUPP_FONT_DWG_1)
```

Choosing Autolayout Parameters

Select **Auto Layout** from the **Tools** menu.

NOTE: If you have existing annotations associated with any parts of your workspace, you should attach them to their source; otherwise, they will move to a different area of the canvas.

Layout Type

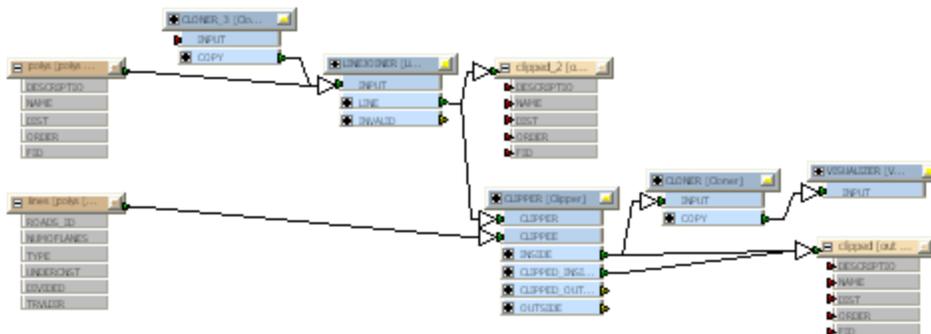
These are preset layout configurations that can be useful if you have a very large workspace. You might have to experiment to see which one works best for your particular workspace. You can choose one of these selections and leave the default values in the fields below, or adjust the values below to fine-tune the layout.

Simple: Chooses a basic layout and greys out horizontal and vertical layout options.

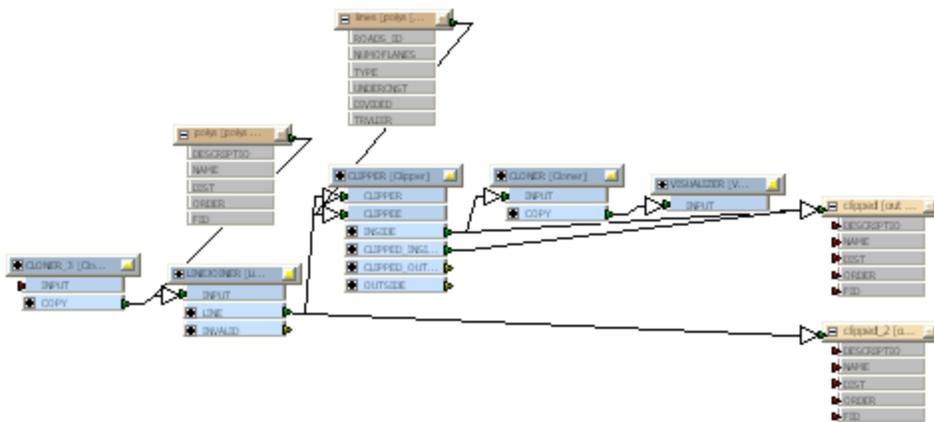
Compact: Compacts the workspace.

Spaced-Out: Spreads out the workspace.

Left-Aligned



Right-Aligned



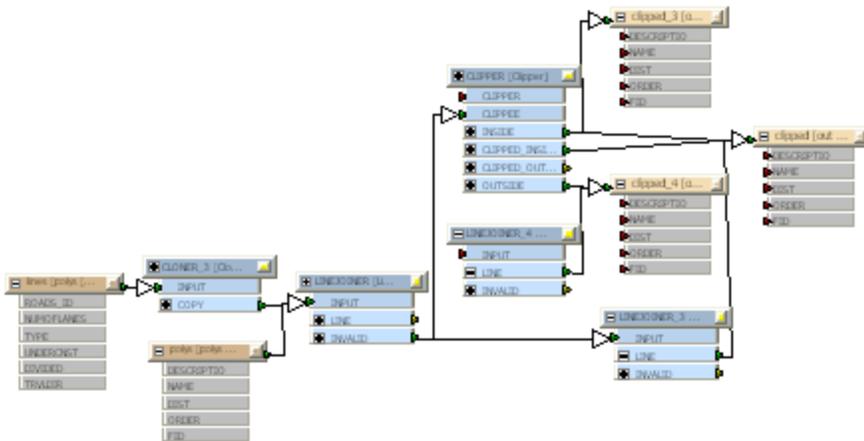
Horizontal Layout

If you choose Squeeze Left or Squeeze right, it will squeeze nodes to either side if there is extra blank space in the view. Choosing Neither means that the nodes will be spread out.

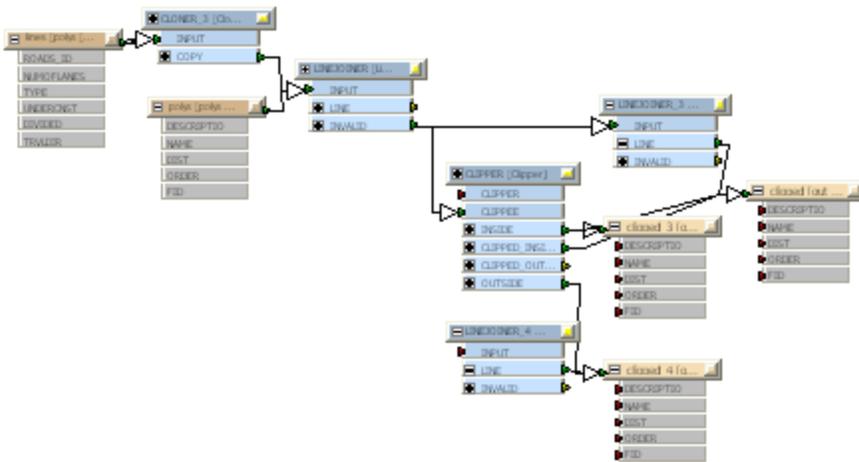
Vertical Layout

If you choose Squeeze Up or Squeeze Down, it will squeeze nodes to the top or bottom if there is extra blank space in the view. Choosing Neither means that the nodes will be spread out.

Squeeze Down



Squeeze Up

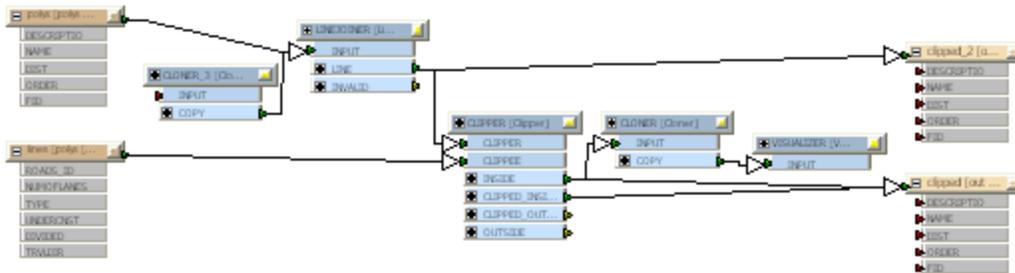


Horizontal and Vertical Spacing

The spacing is the minimum space between adjacent nodes, so even if you have Squeezing enabled, you can specify a lot of space between nodes. The squeeze parameters are really to specify where to place nodes if there is extra blank space in the workspace.

Line up sources and Line up destinations

These checkboxes determine whether the source or destination feature types should be **lined up**, on their own, at the left or right of the diagram. If they aren't set, the source/destination feature types may be moved into the center of the diagram with the transformers.



Layout selected items only

Choose to auto-layout only what you have selected in the workspace.

Insert extra link vertices

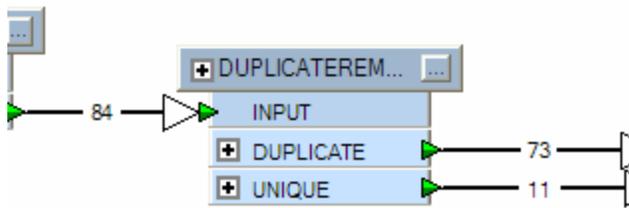
Allows you to route links and place nodes around attached comments using the Insert Vertex on Link command.

Setting the Feature Count Display

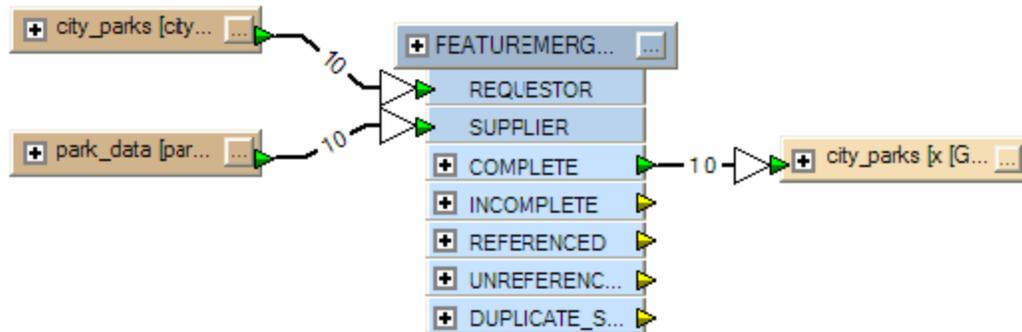
The feature count display shows how many features passed through each link. This function helps analyze the results of a workspace, and provides a reference for debugging if the destination results are not what you expected. Feature counts are also written to the Log file but you can find the references more quickly on the links in a workspace.

To enable feature count display, click the Display Feature Counts tool  on the toolbar. Click the tool again to turn off feature counts (although you will have to run the workspace again to refresh the display).

In this example, you can see that 84 features entered the transformer and that 11 of the features had unique values.



The example below shows the feature counts when spatial and non-spatial data is merged: 10 features from each stream are merged to form 10 output features.



Printing Workspaces

Default Print Options

Set the following default print options by selecting File > Page Setup.

- **Paper Type:** Choose from standard page sizes.
- **Orientation:** Portrait or Landscape
- **Margins:** Set margins in millimeters.
- **Scale:** Choose to always fit the workspace on a page, or always scale to a certain percentage.
- **Decorations:** Check this box to include file-type information, page number and corner borders on the printout.

Print Preview

Select to preview a topic before you print by selecting File > Print Preview. You can print directly from this window.

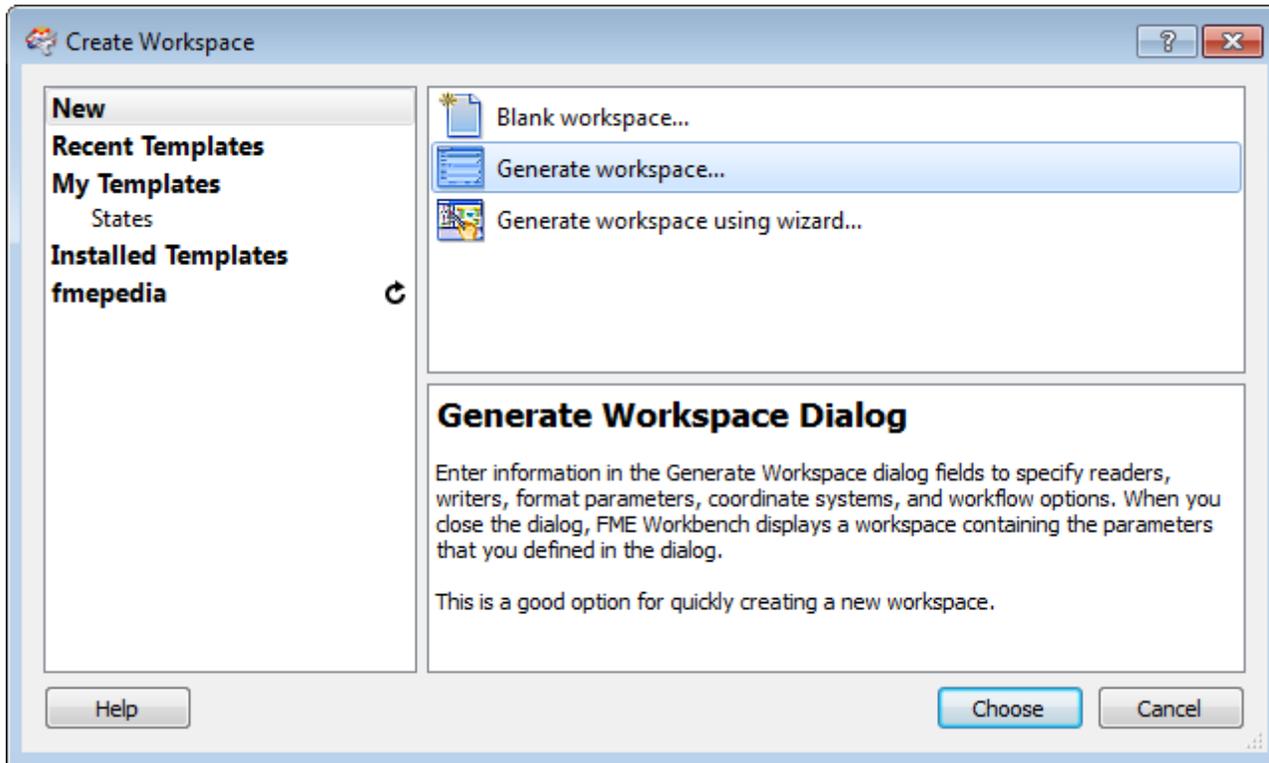
Print

Bypass preview mode and select File > Print.

Creating a New Workspace

There are several options for creating a new workspace.

To display the Create Workspace dialog, select File > New or click the New Workspace tool on the toolbar 



New

- Blank workspace: Opens a blank workspace file. You can build a workspace and save it to a file later.
- Generate workspace to set up a workspace using a traditional dialog.
- Generate workspace using wizard to quickly define source and destination formats and generate an initial workspace.

Templates

Create a workspace from a template.

Using the Workspace Dialog

Location in FME Workbench:

File > New > Generate workspace...

Start tab > Generate workspace

Select Reader Format and Dataset

Select a reader format and enter a dataset name (or you can select a dataset and FME will fill in the format field). These fields will already be filled in if you dragged a dataset directly onto the Workbench window.

Specify Reader Parameters

If the Parameters button is enabled (and this depends on the format you choose), you can specify more detailed information about the reader data. Parameters are optional: if you don't edit them, FME will use default values. For detailed help, press the Help button or F1 key when you open a parameter box.

View Reader Dataset

If you want to inspect the reader dataset before you start the translation, click the FME Universal Viewer button: .

Coord. System

For formats that know their coordinate system, the Coordinate System field in the Source Dataset dialog will display *Read from source* and FME will read the coordinate system from the source dataset. For other formats, the field will display *Unknown*. You do not have to edit these defaults.

Clicking the browse button beside this field will display the Coordinate System Gallery, which contains all the coordinate systems that FME supports.

Select Writer Format and Dataset

Select the format to which the data will be translated, and a name for the output file.

Specify Writer Parameters

If the Parameters button is enabled (and this depends on the format you choose), you can specify more detailed information about the reader data. Parameters are optional: if you don't edit them, FME will use default values. For detailed help, press the Help button or F1 key when you open a parameter box.

Coord. System

For formats that know their coordinate system, the Coordinate System field in the Source Dataset dialog will display *Same as source* and FME will read the coordinate system from the source dataset. For most other input sources, the field will display *<not set>*, which means that FME will use default values. You can accept the defaults in most cases.

Clicking the browse button beside this field will display the Coordinate System Gallery, which contains all the coordinate systems that FME supports. If you want to change the writer coordinate system, see [Changing the Default Coordinate System](#).

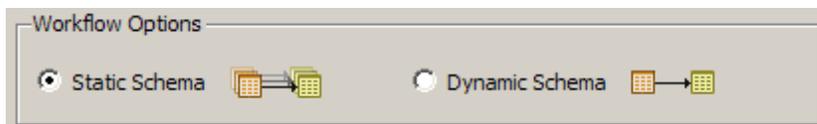
Import Feature Types Definitions (for some database formats)

If you are writing to a database format, you can choose to import feature types with numerous attributes and properties from an existing writer. This eliminates this extra step after you create the workspace.

If, for example, you want to write to an existing Oracle table, you can check Import Feature Type definitions here and import the destination feature type with all of its attributes and data types defined exactly as they are in the existing Oracle table.

Set Workflow Options

You can set workflow options before you create the workspace.



Static Schema is the default workflow, and is suitable for most workspaces. The schema is read from the source data and replicated on the workspace. Each reader feature type will be connected to its corresponding writer feature type.

Dynamic Schema breaks the dependence on the source and destination schema. One merged feature type will be connected to one writer feature type that is configured for dynamic operation. The schema is not replicated on the workspace; therefore, if the source data changes, you will not have to update the workspace – FME will do this automatically.

Read more about Dynamic Workspaces

Generate the new workspace file

Click OK. The translation log displays information related to workspace generation. Note that if you closed the translation log through the View > Windows menu, you can also see a subset of logging information in Workbench's status bar.

Using the Workspace Wizard

Location in FME Workbench:

File > New > Generate workspace using wizard...

If you start Workbench without using shortcuts with any pre-existing data or workspace files, following the steps in the wizard is the easiest way to create a new workspace file:

1. Select the source format from the pull-down list. See Choosing from the Formats List.
2. Click the browse button to locate your source data. Click the Advanced Browser button  to add multiple source datasets.
3. Specify format parameters. If the Parameters button is enabled (and this depends on the format you choose), you can specify more detailed information about the source data. These are optional parameters so if you do not edit them, Workbench will use default settings. Click the Help button in a parameter box for information on parameter fields.
4. Select your destination format, either from predefined formats in the pull-down list, or from the Reader and Writer Gallery.
5. Click Finish to create the workspace.

Once you have created the initial workspace and it is displayed in the graphical view, you can either save the file in its current form, or adjust the data flows by adding transformers or new attributes.

Creating Workspaces from Templates

Templates are one of the best ways to get started with FME Workbench, whether you're new to FME, whether you're setting up a new workflow and you want some hints to get started, or even if you want to implement some best practices in your workspaces.

Templates have the file extension **.fmwt**.

There are several advantages to using templates in Workbench:

- Download and install basic templates from Safe Software's fmepedia site for use with both FME Desktop and FME Server. Downloadable templates will be updated frequently.
- Access training and tutorial workspaces.
- Create your own templates from existing workspaces and save them to a My Templates folder.

fmepedia Templates

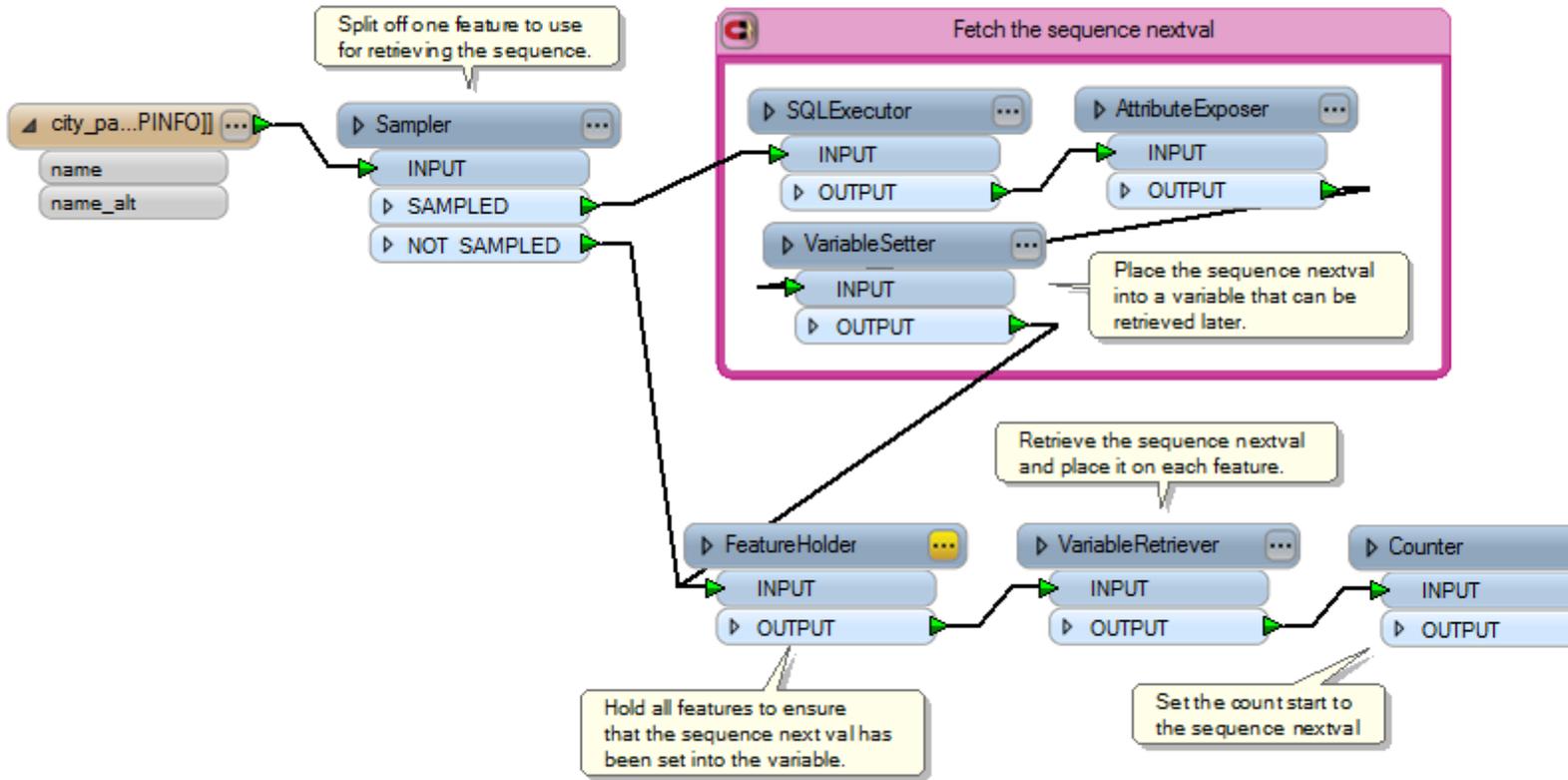
Click the fmepedia heading to view available templates, or click one of the categories to see a subset. The lower-right pane gives a description.

1. To download a template, select it and click Choose, or double-click the name.
2. The Workbench log window shows the status of the download:

```
Requesting catalog details for selected template...
Retrieved catalog details.
Establishing connection to fmepedia...
Successfully connected to fmepedia.
Downloading 1 resource(s)...
```

Downloading 'Read oracle sequence nextval.fmw' (1/1) ...
 Successfully downloaded resource(s).
 The selected template was successfully downloaded.

3. When the template opens in Workbench, it will already contain example input data and transformers, along with explanatory annotations.



Saving a Workspace as a Template

From an open workspace:

1. Select File > Save As Template.
2. You can rename the default title that Workbench assigns to the template.
3. Type a category name as the location to store the template. If you have not previously saved a template, the Category field will not display any other selections.
4. Type a description for the template.
5. Any associated datasets are listed in the Select files to add area. The default is to include the dataset as part of the template. You can also choose to include additional files, including multiple datasets.

The default Save location is <User>\Documents\FME\Templates\<template_name>.fmw.

Creating a Workspace with Multiple Readers

There are a few ways that you can create a workspace with multiple readers:

If readers are the same format and schema

Select multiple files when you create a new workspace:

- If the sources are all the same format and schema, select multiple files when you create a new workspace.
- If the sources are all the same format and schema, double-click the source (reader) dataset parameter in the Navigator pane, and select multiple files.
- For sources of differing format and schema: select Readers > Add Reader on the menu bar. This lets you add new reader datasets to the workspace.
- You can also select Readers > Add Reader if the format and schema are the same, but other dataset settings need to be different.

Using the Workspace Dialog

Location in FME Workbench:

File > New > Generate workspace...

Start tab > Generate workspace

Select Reader Format and Dataset

Select a reader format and enter a dataset name (or you can select a dataset and FME will fill in the format field). These fields will already be filled in if you dragged a dataset directly onto the Workbench window.

Specify Reader Parameters

If the Parameters button is enabled (and this depends on the format you choose), you can specify more detailed information about the reader data. Parameters are optional: if you don't edit them, FME will use default values. For detailed help, press the Help button or F1 key when you open a parameter box.

View Reader Dataset

If you want to inspect the reader dataset before you start the translation, click the FME Universal Viewer button: .

Coord. System

For formats that know their coordinate system, the Coordinate System field in the Source Dataset dialog will display *Read from source* and FME will read the coordinate system from the source dataset. For other formats, the field will display *Unknown*. You do not have to edit these defaults.

Clicking the browse button beside this field will display the Coordinate System Gallery, which contains all the coordinate systems that FME supports.

Select Writer Format and Dataset

Select the format to which the data will be translated, and a name for the output file.

Specify Writer Parameters

If the Parameters button is enabled (and this depends on the format you choose), you can specify more detailed information about the reader data. Parameters are optional: if you don't edit them, FME will use default values. For detailed help, press the Help button or F1 key when you open a parameter box.

Coord. System

For formats that know their coordinate system, the Coordinate System field in the Source Dataset dialog will display *Same as source* and FME will read the coordinate system from the source dataset. For most other input sources, the field will display *<not set>*, which means that FME will use default values. You can accept the defaults in most cases.

Clicking the browse button beside this field will display the Coordinate System Gallery, which contains all the coordinate systems that FME supports. If you want to change the writer coordinate system, see [Changing the Default Coordinate System](#).

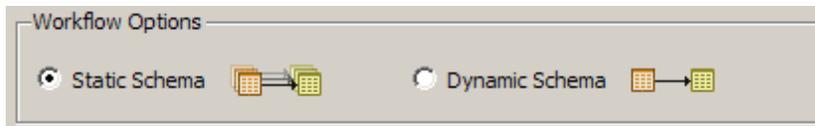
Import Feature Types Definitions (for some database formats)

If you are writing to a database format, you can choose to import feature types with numerous attributes and properties from an existing writer. This eliminates this extra step after you create the workspace.

If, for example, you want to write to an existing Oracle table, you can check Import Feature Type definitions here and import the destination feature type with all of its attributes and data types defined exactly as they are in the existing Oracle table.

Set Workflow Options

You can set workflow options before you create the workspace.



Static Schema is the default workflow, and is suitable for most workspaces. The schema is read from the source data and replicated on the workspace. Each reader feature type will be connected to its corresponding writer feature type.

Dynamic Schema breaks the dependence on the source and destination schema. One merged feature type will be connected to one writer feature type that is configured for dynamic operation. The schema is not replicated on the workspace; therefore, if the source data changes, you will not have to update the workspace – FME will do this automatically.

Read more about [Dynamic Workspaces](#)

Generate the new workspace file

Click OK. The translation log displays information related to workspace generation. Note that if you closed the translation log through the View > Windows menu, you can also see a subset of logging information in Workbench's status bar.

About Dynamic Workspaces

Dynamic workspaces are a way of providing maximum translation flexibility and minimizing the longer-term maintenance of workspaces.

Dynamic means that the workspace will be set up very simply, but with maximum flexibility. You can read any source dataset (of the chosen format) and it will be read and written (to the destination format) correctly. The important part is that you can change the source to a different dataset, and it will still work.

Why Create Dynamic Workspaces?

Traditional FME Workspaces are very tightly bound to the source and destination schemas. In many cases, this is what you will want. However, in certain situations, the workspace needs to be more independent of the schemas.

A dynamic workspace breaks the dependence on the source and destination schema. Common applications for dynamic workspaces are:

- when a relatively simple data transformation is to be applied to all the data irrespective of schema (that is, clipping, coordinate transformation)
- when the source schema is not predictable or well-defined
- ad hoc data transformations
- when the destination format might vary
- long-term maintenance of the workspace is required

While translations are generally very easy to implement using FME Workbench, in the longer term they can require a lot of maintenance to keep up-to-date. If new feature types or attributes are added to the source data, you must also update the workspace. Sometimes you may not even realize that a data owner has changed their data schema.

What can be made Dynamic

There are a number of components that you could make dynamic in an FME workspace:

Schema: Source Feature Types

A workspace that would read any set of feature types from a source dataset

This is supported by the Merge Filter

Schema: Destination Feature Types

A workspace that would read any set of feature types from a source dataset , and would write a corresponding set of feature types

This is supported by the Dynamic mode

Schema: Attributes

A workspace that would read any set of attributes on the source feature types and write them out in the same schema

This is supported by Dynamic mode

Format

A workspace that would write to any format, without the need to add multiple writers

This is supported by the Generic writer

Copying Between Workspaces

You can work with more than one workspace at a time:

1. Start one or more additional Workbench applications and open different workspaces.
2. Drag, copy and paste between workspaces.

Copying Workspaces to Mapping Files

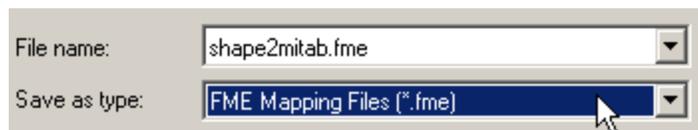
This is a useful feature if you want to start with Workbench and then switch to [FME mapping files](#) for use in the FME Universal Translator.

Mapping Files

An FME translation is controlled by a series of rules specifying the guidelines and transformations of a data translation. The mapping file drives the operation of all FME modules during a data translation.

Although the order of the major sections of a mapping file is not significant, typically the configuration of readers and writers is placed at the top. Next, the factory definitions, if any, are presented. Finally, the definitions of the features in the input and output systems are given with the rules for transforming them from one system to the other.

The easiest way to save a workspace as a mapping file is to choose **File > Save As** and select FME Mapping Files (*.fme) as the type.



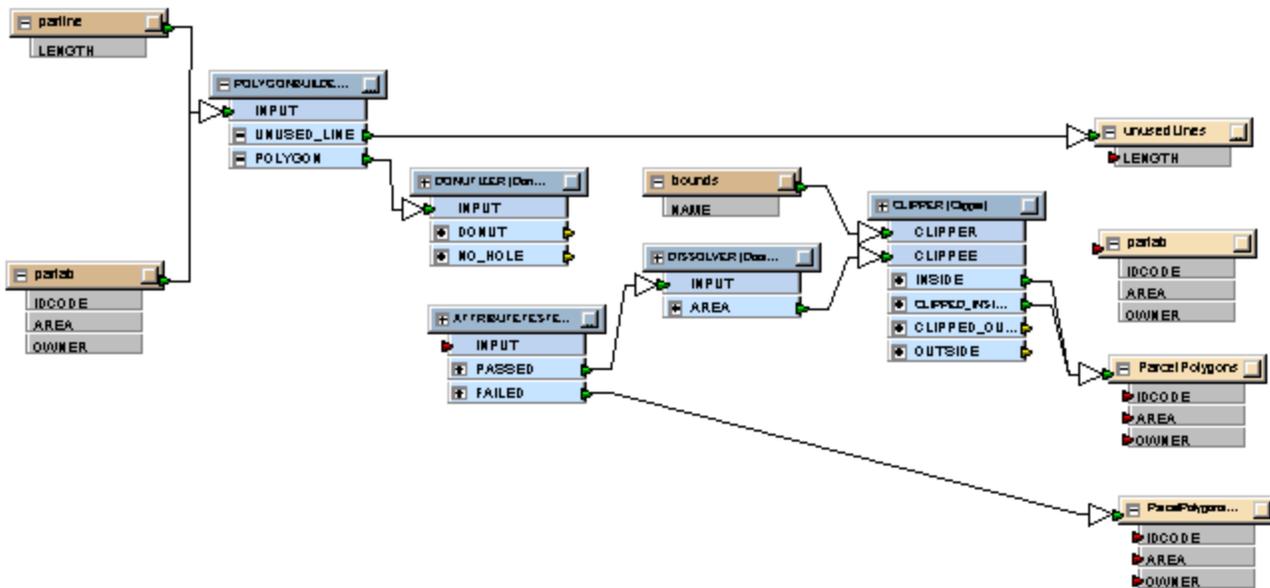
The filename extension will change to .fme so the workspace will be readable in the FME Universal Translator.

Note for Windows Vista users:

When you Save as type FME Mapping Files, you will have to manually change the filename extension in the File name field from <filename>.fmw to <filename>.fme.

Copying Part of a Workspace to a Mapping File

1. Select and then copy (Ctrl+C) all or part of the [contents of the Workbench canvas](#).



2. Open a text editor and [paste the contents](#).

```
23 # -----
24
25
26 FACTORY_DEF * TeeFactory
27     FACTORY_NAME POLYGONBUILDER_Splitter
28     INPUT FEATURE_TYPE parline
29     INPUT FEATURE_TYPE parlab
30     OUTPUT FEATURE_TYPE ___TOPOLY___
31
32 FACTORY_DEF * TeeFactory
33     FACTORY_NAME POLYGONBUILDER_Stroker
34     INPUT FEATURE_TYPE ___TOPOLY___ fme_type fme_ar
35     OUTPUT FEATURE_TYPE *
36         @Arc(&fme_primary_axis,&fme_secondary_axis
37             @SupplyAttributes(fme_type,fme_line)
38
39 FACTORY_DEF * PolygonFactory
40     FACTORY_NAME POLYGONBUILDER
41     INPUT FEATURE_TYPE ___TOPOLY___
42     END_NODDED
43     OUTPUT LINE FEATURE_TYPE POLYGONBUILDER_UNUSED_
44         fme_type fme_line
45     OUTPUT POLYGON FEATURE_TYPE POLYGONBUILDER_POLY
46         fme_type fme_area
47
48
```

3. Save the file with a .fme extension.

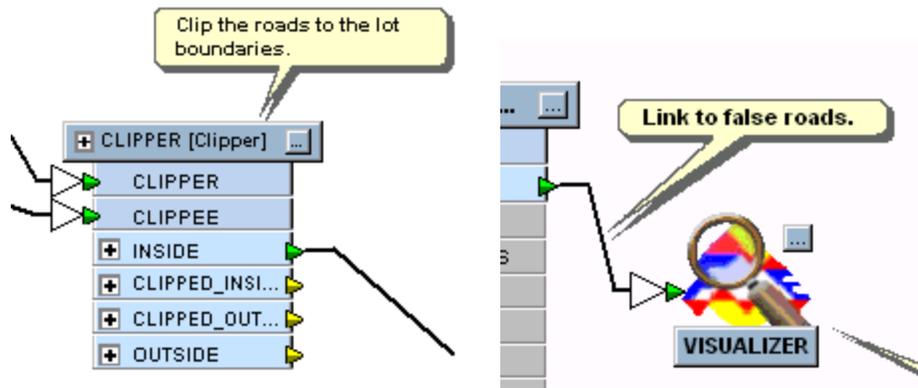
You can now switch to the FME Universal Translator interface and, depending on the information you copied, either run the file or add it to the Mapping File Registry for use in other translations.

Note: You cannot import mapping files to Workbench, and you can not re-import files after you export them as .fme files.

Attaching Annotations

Annotations are useful for documenting the purpose of a feature type, link, or transformer, especially in a large or complex workspace.

1. Select a feature type, link, or transformer, click the right mouse button and then choose **Attach (or Add) Annotation** from the command menu.
2. The default annotation anchors to the selected node. Click twice on the annotation to edit the default text.



If you select exactly one node or link and exactly one annotation, right-click and choose **Attach Annotation**, the free-standing annotation will attach to the selected node or link.

Attaching Default Annotations to Multiple Nodes and Links

The default annotation includes the name of the node. For feature links, it will include both the source and destination node information.

- Select multiple nodes or links.
- Right click, and choose **Attach Annotation**.

Annotations will be added to all the nodes and links. Nodes will show the node name as default string, and links will show <source node name> to <destination node name> as the default string.

Tips:

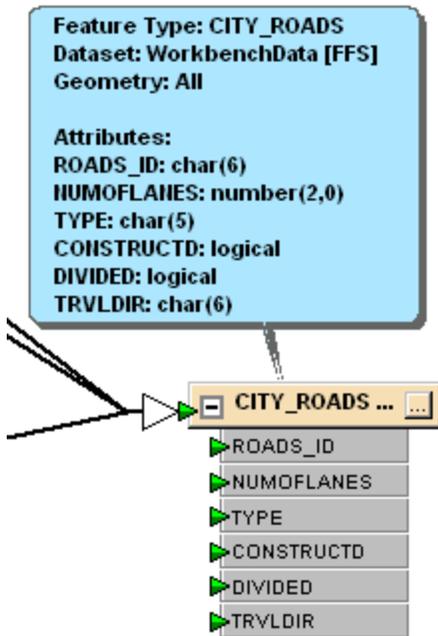
- Right-click on an annotation and select **Follow Attached Node**. If this option is deselected, a comment leader line will stretch to follow a node, but the comment will remain in its original place.
- By default, annotations appear in front of the display and can overlay other parts of the workspace. To move annotations to the back of the display, right-click on an annotation and select **Send to Back**. This also works for feature types and transformers.

You can also insert "floating" annotations in a workspace by selecting **Annotation** from the Insert menu, or by clicking . If you want to connect it later, simply select the annotation, click the right mouse button, and choose **Attach to Feature Type** or **Attach to Transformer**. You can also choose to detach any annotation from its source.

Summary Annotations

Select a feature type or transformer, right-click and choose **Show Summary Annotation** from the command menu. An annotation bubble displays detailed information.

Tip: This feature is useful when printing workspaces.



You can toggle the display by choosing **Summary Annotations** from the View menu.

Disabling Connections

You can disable selected workspace connections.

You might want to do this if you are testing a large workspace and want to isolate or simplify some functionality. It is sometimes easier to disable certain connections than to disable feature types, since a feature type may be linked to several different areas in the workspace. Another advantage, when disabling multiple connections, is that you do not have to disconnect them to temporarily exclude them from the workspace.

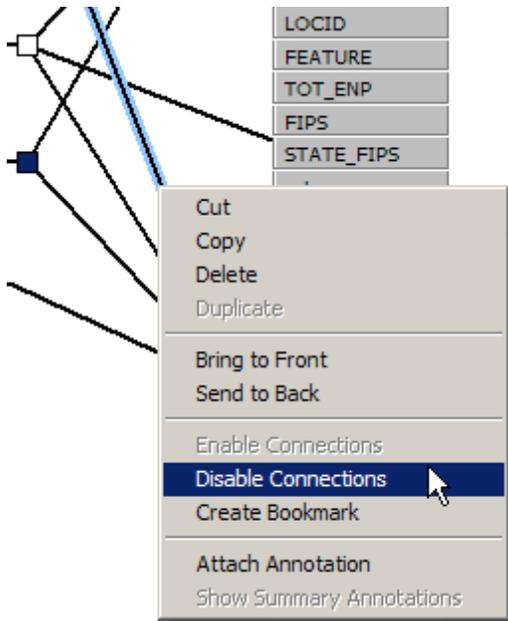
Single Links

When you select one link and right-click, you can choose Disable from the command menu. This will disable and gray out only the selected link.

To enable the link, select the link either in the workspace or in the Navigator pane, right-click and choose Enable.

Multiple Links

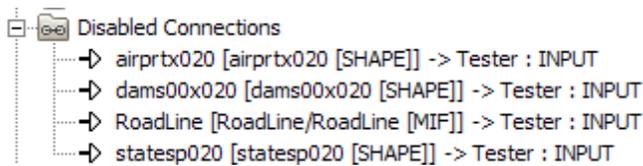
When you select more than one link and right-click, choose Disable Connections from the command menu.



Note: If you select multiple objects that includes links, the Enable Links or Disable Links menu will not appear. The menu that appears will show an Enable/Disable option that applies to applicable selected nodes (e.g., feature type definitions).

Disabled Connections in the Navigator Pane

A Disabled Connections parameter will appear in the Navigator pane. Click to open the node and see the list. Clicking on a link will select and scroll to the link in the main workspace window.



Re-enabling Connections

To re-enable connections, right-click on the disabled connection(s) in the Navigator pane and choose Enable, or right-click on the connection(s) on the canvas and choose Enable Connections.

Running a Translation

To run a translation after you create your workspace, click **OK** in the New Workspace dialog, press Ctrl+R, or click

FME will begin the translation, and you'll see information in the Translation Log.

The amount of time the translation takes depends on different factors that may include: the processing power of the FME host machine, the size of your dataset, and the amount of processing that FME has to do.

In most cases, however, a translation will take a few minutes at most, and upon completion, the Log will display **Translation was SUCCESSFUL**. You can scroll back through the log to check specific statistics.

■ **Do you want to assign more CPU resources to speed up a translation?** Select **Translation Priority** from the Tools | Options

menu, and select **High**.

- **Do you want to make sure you can work on other tasks in the foreground while you're running a large translation?**
Select **Translation Priority** from the Tools | Options menu, and select **Low**.

What Next?

You have completed the steps involved in performing a basic translation.

Unless you selected to save your workspace automatically (through the **Tools > Options** menu), you will be prompted for an output filename. A default filename will be offered, based on your source and destination formats (for example, dwg2dgn.fmw).

What if a Translation Fails?

If you see a message that says **Translation Failed**, the FME log will often be the best source to find out what went wrong. Usually it will be something simple, like a missing file. Sometimes, however, you will have to perform some troubleshooting on either your source data or on the transformer properties. A good way to troubleshoot is to Route your output to an interim FFS file, a Viewer, or a NULL output.

Information and Help

A good source of information and help is [fmpedia](#).

Redirecting the Source and Destination

NOTE: In versions of FME earlier than 2006, this function was accessed from the Tools menu as Route from FFS / Route to FFS.

You can choose to quickly reroute your output without having to change your workspace. You might want to do this if you're having problems in your workspace, and want to examine features or information before writing to your original output file or directory.

After you run your translation, the log window displays a banner as a reminder that you have redirected your source or destination.

Find the Workspace Settings node in the navigator pane. Double-click **Source Redirect** or **Destination Redirect**



Redirect to FFS file is useful for debugging complex workspaces. The output is written to FME Feature Store files (FFS). The FFS files can then be viewed using FME Universal Viewer. Route to FFS can also be used to send database data to Safe Support. Since **Redirect to FFS file** does not use the format writers configured in the workspace, it can be used to isolate problems that may be occurring in a specific FME format writer.

Redirect from FFS file is used for debugging complex workspaces. Output that has been created using **Redirect to FFS file** can be reloaded into the translation pipeline using **Redirect from FFS file**.

Disable Output: Instead of writing many features to a dataset, you might just want to see information or statistics gathered during processing (usually used in conjunction with the Recorder transformer). The actual output will be NULL, or, nothing.

Redirect to Visualizer: View data before writing to the output file or directory.

For detailed information on the FFS format, see the FFS Reader/Writer chapter in the *FME Readers and Writers* manual.

Running a Batch Translation

The FME Workbench interface supports batch execution and batch script creation. For example, you can apply an existing workspace transformation to a large number of input files, and produce separate outputs for each.

Open your existing workspace and select **Batch Deploy** from the **File** menu. The Create Batch Translation Wizard appears.

Note: If the File>Batch Deploy option is disabled, it is because there is no source or destination dataset in the workspace.

Source Type

Select the source type from the drop-down list.

Source Data

Choose the dataset to use for the batch translation.

Destination Location

If you have specified more than one output destination, this pane will appear for each destination.

Batch deploy this writer

Check this box, and enter or browse to the destination location.

Retain Source Dataset Basename in Destination Path

The output dataset name is always taken from the source name, for example: roads.dgn becomes roads.gml. However, this doesn't always make sense when using folder-based datasets. For example:

C:\myData\ShapeFiles\roads.shp

converted to MIF/MID and pointed to a MIFFiles folder would become

C:\myData\MIFFiles\ShapeFiles\roads.mif

This is because the "ShapeFiles" portion is treated by FME as the name of the source dataset, and is therefore included in the output. Because of this, you have an option to drop the source dataset name from the output.

Retain Source Dataset Basename in Destination Path

The **Retain Source Dataset Basename in Destination Path** box is checked by default to ensure backwards compatibility. To drop the source dataset name, uncheck this box. If you uncheck the box, and reference the above example, the destination would now be:

C:\myData\MIFFiles\roads.mif

Note that before using this option, you should be aware of the difference between file- and folder-based datasets.

You can also **Type the optional suffix to append to your destination dataset** (for example, a file-type suffix).

Batch Translation Run Options

Select when you want to run the batch translation:

- **Run Now:** Execute the batch translation immediately, from within Workbench.
- **Save scripts to run outside of Workbench:** Type or browse for the location of the batch translation file. Note that batch files consist of both a .tcl file and a .bat file. Both files are needed to run the batch translation. To run the scripts outside of Workbench, go to the Windows DOS prompt and:
 - type the name of the .bat file, or
 - type **fme tclfilename.tcl**

Run the Translation

Click the Finish button to run the batch translation.

Stopping a Translation

When you're running a translation, the Stop tool  becomes available. Click the  tool to immediately stop the translation.

NOTE: Sometimes stopping a translation causes FME to create temporary files that can take up unnecessary disk space. It's a good idea to select Tools > Purge Temporary Files (or press Alt + C) after you stop a translation.

What Happens After You Stop a Translation?

The log view will display Translation Stopped.

When you stop a translation, the output file is left incomplete and is not usable. If you run the file again, the entire translation will restart, overwriting any partial results from a previously aborted translation.

Using the FME Universal Viewer

Route Output to the Viewer

Quickly route to the Viewer without changing anything in your workspace. This is useful if you want to view the output before writing to the output dataset, and you don't have to add anything to the workspace.

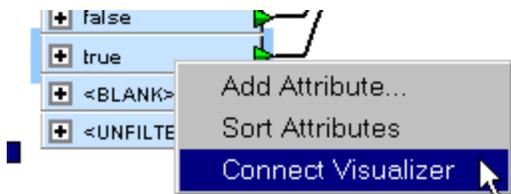
View an Output File

After you finish a translation, you can view the output file with the FME Universal Viewer. If the Viewer isn't already running, start it up and open the output file.

Add Visualizers

Add Visualizer transformers to your workspace to automatically start the Viewer. You might want to add several Visualizers at different points in your workspace, to view different types of data.

Right-click on an output port in a Transformer, and select [Connect Visualizer](#). This connects a Visualizer directly to the output port, and names it according to the output port. Through the Properties button, you can further distinguish between Visualizers by setting unique pen and fill colors. Enable and disable individual Visualizers through the command menu.



Note: For detailed help, open the Help menu in the FME Universal Viewer application.

About Data Transformation

Transformers provide powerful ways to transform data as it moves from the source system to the destination system.

When the output from the source type or a transformer is connected to another transformer, Workbench makes an implicit connection between all attributes that have the same name. If necessary, Workbench automatically assigns attributes to transformers based on both the transformer itself (usually one specific attribute like *count* or *length*, for example) and then appends the attributes contained in the input connection.

What is Data Transformation?

Data Transformation is FME's ability to restructure data. The key feature of this ability is that transformation automatically takes place between the reading (extract) and writing (load) of data, without requiring user intervention.

Data Transformation Types

Data transformation can be subdivided into two distinct operations.

Transforming Structure

This type of transformation might be better called "reorganization". It refers to the ability of FME to channel data from source to destination in an almost infinite number of arrangements. This would include the ability to merge data, divide data, reorder data, and define custom data structures.

Transforming the structure of a dataset is carried out by manipulation of its schema.

Transforming Content

This type of transformation is also known as "restructuring". It refers to the ability to alter the content of a dataset; manipulation of a feature's geometry or attribute values is the best example of how FME is able to transform content.

Relationship to FME Functions and Factories

If you are familiar with FME mapping files, the transformers provided in Workbench encapsulate the FME Functions and Factories. Transformers manipulate your source data to achieve the desired output. Just like functions and factories, some transformers add attributes to features, others erase attributes, and still others will only transform the geometry. Transformers can operate on individual features, one at a time, or on groups of features.

You can create and store your own Custom Transformers.

Workbench transformers are listed in the Transformer gallery.

Understanding Schema

What is a Schema?

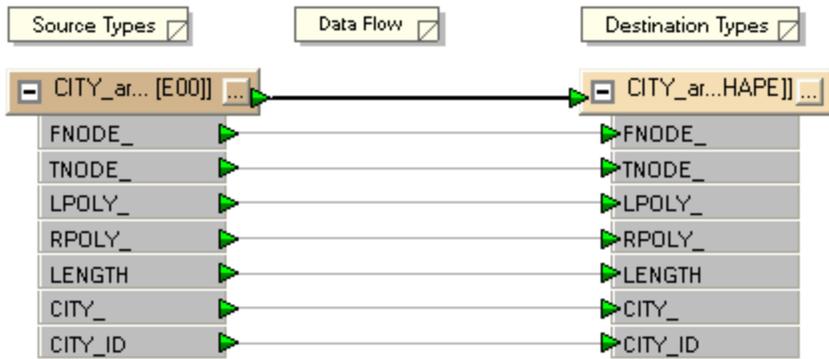
A schema (sometimes known as "data model") can be described as the structure of a dataset, or more accurately a formal definition of a dataset's structure.

Each dataset has its own unique structure (schema) which includes feature types, permitted geometries, user-defined attributes and other rules that define or restrict its content.

How Does FME Handle Schemas?

When you create a new workspace, FME reads the input dataset and creates a workspace definition of the schema. Generally it will also create a writer schema, that is, a definition of the structure of the writer. Writer schemas could be called "logical" schemas since they don't physically exist at that point.

Here is a reader and writer schema as they appear in Workbench. Source data is on the left, and destination data is on the right.



Each item is a separate feature type. Here there is one source and one destination feature type, and each feature type has a set of attributes.

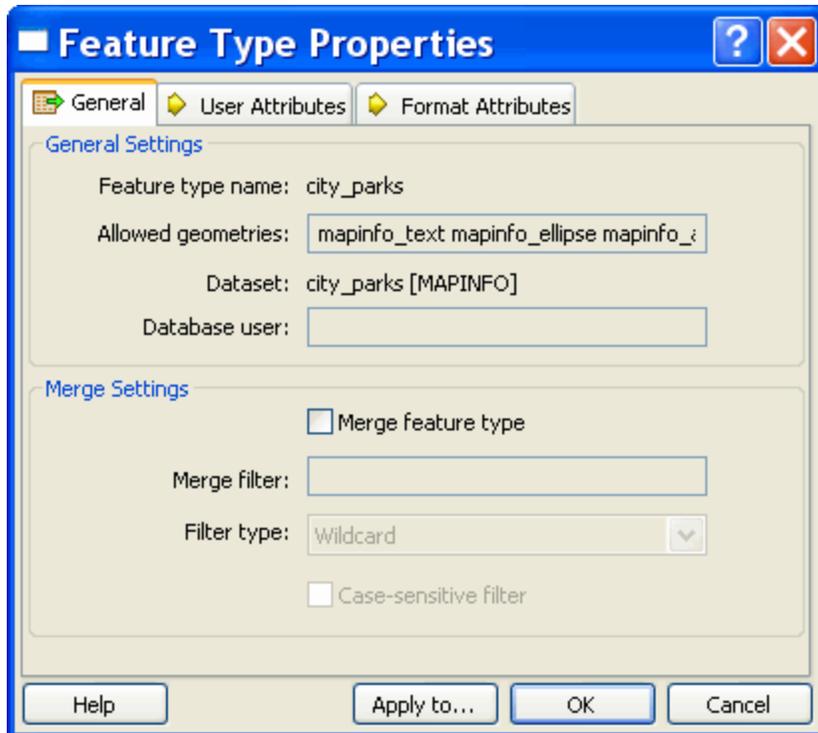
A new workspace will usually have identical source and destination schemas; but this is not always possible – particularly when the source and destination formats are different. In these circumstances, FME will attempt to compensate for any differences between source and destination schema. The workspace can then be edited and the destination schema changed as required; for example attributes can be added, removed or renamed.

One of the real powers of FME is the ability to edit destination schemas and transform data to match during processing.

Viewing the Schema in FME Workbench

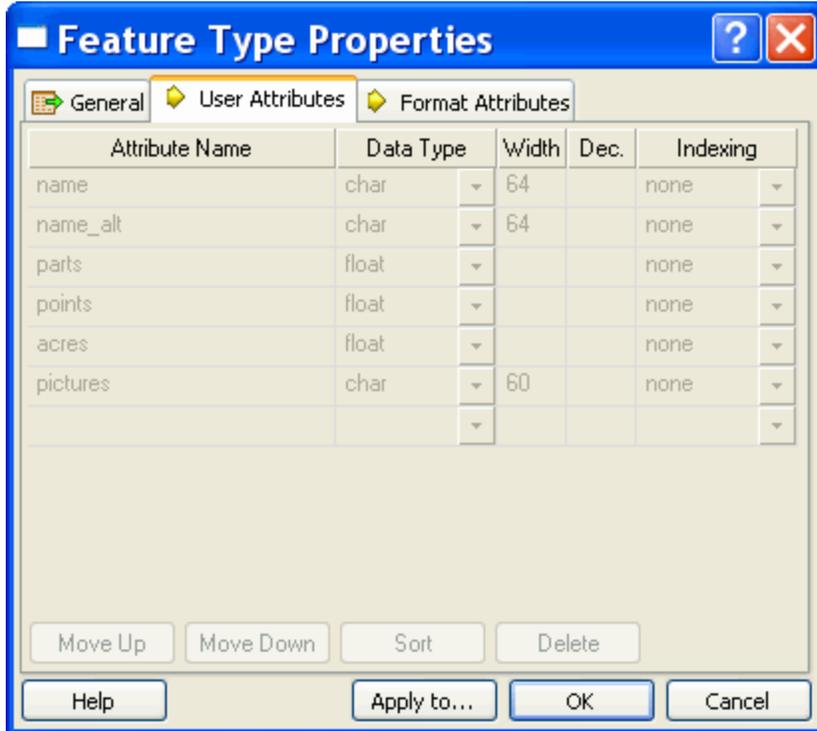
A schema is made up of many components. Some of these relate to a dataset as a whole; for example feature types belonging to a dataset are regarded as part of the overall schema and are depicted in the Workbench canvas window.

However, some parts of the schema relate specifically to a single feature type only. Attributes are one such component. These components are shown in the properties dialog of a feature type.



Above: The Feature Type Properties dialog has a number of tabs to display information.

Under the General tab there is the name of the feature type, in this case city_parks. Permitted geometry types are shown here, as is the name of the parent dataset for the feature type.



Above: The User Attributes tab shows a list of attributes present the feature type. Each attribute is defined by its name, data type, width and number of decimal places.

This example shows a source feature type (you can't edit the attributes). Source attributes cannot be edited because this is a representation of the physical schema of the data; if they were changed the schema would no longer match the source dataset.

The attributes on a destination dataset can be edited, to create the required output.

The Data Type column for an attribute shows only values that match the permitted types for that data format. For example, an Oracle destination schema permits attribute types of varchar or clob.

Schema Editing

The default schema that FME creates is one suitable for a Quick Translation. When there is a need to customize the output schema, edits can be made using Workbench.

What is Schema Editing?

Schema editing is the process of altering the destination schema to customize the structure of the output data. One good example is renaming an attribute field in the output.

What Can be Edited?

You can edit a number of things, including (but not limited to):

- **Rename an Attribute:** Any attribute on the destination schema can be renamed. To rename an attribute open the Feature Type Properties dialog and click the 'User Attributes tab. Click the attribute to be renamed and enter the new name.
- **Change an Attribute Type:** Any attribute on the destination schema can have a change of type, for example changing ID from an integer to a float. To change an attribute type open the Feature Type Properties dialog and click the User Attributes tab. Use the Data Type field to change the type of an attribute.

- **Rename a Feature Type:** To rename a destination feature type (for example rename roads to updated_roads) open the Feature Type Properties dialog. Click the General tab. Click in the Feature Type name field and edit the name as required.
- **Change a Geometry Type:** To change the permitted geometry for a feature type, (for example change the permitted geometries from lines to points) open the Feature Type Properties dialog. Click the General tab. Choose from the list of permitted geometries. This field will be greyed out where the format permits any geometry type.
- **Change a Dataset (Move a Feature Type):** When more than one destination dataset is defined, you can switch a feature type from one destination dataset to another, using the drop-down list under the General tab.

Schema Mapping

Schema Mapping is the means by which a datasets structure can be transformed.

What is Schema Mapping?

In FME Workbench, one side of the workspace shows the source schema (what we have) and the other side shows the destination schema (what we want). Schema mapping is the process of connecting the source schema to the destination schema in a way that ensures the right source features are sent to the right destination feature types and the right source attributes are sent to the right destination attributes.

Feature Mapping

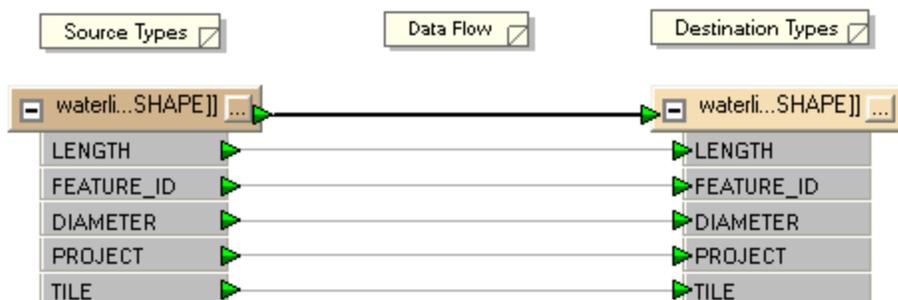
Feature mapping is the process of connecting source feature types to destination feature types.

Attribute Mapping

Attribute Mapping is the process of connecting source attributes to destination attributes.

Below: A source feature type in a Shape dataset (roads) is connected to a destination feature type in a MIF dataset. This is the Feature Mapping.

Each source attribute is also connected to a destination attribute. This is the Attribute Mapping.



In FME Workbench, feature mapping connections (or links) are shown with a thick, black arrow.

Attribute mapping connections are shown with a thinner, grey arrow.

In Workbench, attribute mapping is sometimes implied rather than visualized, and no connecting arrow is shown. The color of the port indicates the connection status. Green indicates a connected attribute. Yellow indicates a source attribute unconnected to a destination, and red indicates a destination attribute that is not connected to a source.

Attributes with the same name in source and destination are automatically connected.

Note: The name is case-sensitive, so ROADS is not the same as roads or Roads.

Schema Mapping in FME Workbench

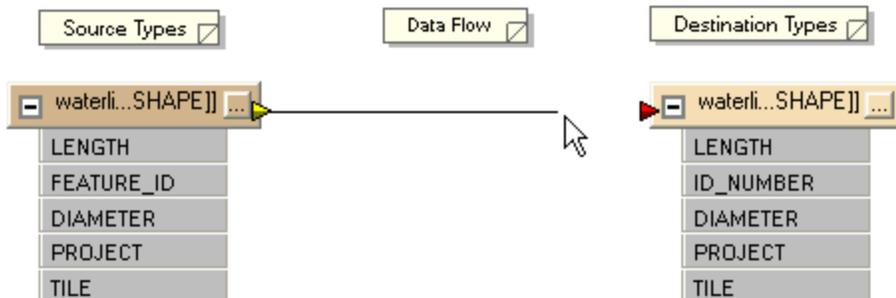
In most cases, FME automatically fills in basic schema mapping in a new workspace. The schema mapping can then be edited as required.

In Workbench's intuitive interface, feature type and attribute connections are made by dragging connecting lines between these parts of the schema.

Feature Mapping in FME Workbench

Feature Mapping is carried out by clicking on the output port of a source feature type, dragging the arrowhead across to the input port of a destination feature type, and releasing the mouse button.

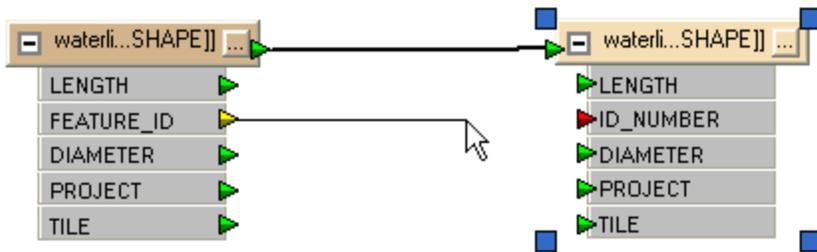
Right: Here a connecting line from source to destination feature type is being created by dragging the arrowhead from source to destination.



Attribute Mapping in FME Workbench

Attribute Mapping is carried out by clicking on the output port of a source attribute, dragging the arrowhead to the input port of a destination attribute, and releasing the mouse button.

Left: Here feature mapping has already been carried out and attribute connections are being made.



A new connection from FEATURE_ID to ID_NUMBER is being made. LENGTH, DIAMETER, PROJECT and TILE have matching destinations so they are connected automatically (implied connection). Note the green, yellow and red color coding showing which attributes have been connected.

Understanding Content Transformation

Content transformations are those that operate on the geometry or attribute content of a dataset.

What is a Feature?

A "feature" in FME is an individual item within the translation. Typically a GIS or cartographic feature consists of a geometric representation, plus a set of related attributes. FME is capable of restructuring either of these components.

A feature in FME is the smallest unit of FME data. Features in FME have a flexible, generic representation; in other words they have a basic "FME" definition that is unrelated to their original format.

Thick-Pipe Translations

The term "thick pipe" translation underscores the content transformation abilities of FME. In other words you can process data during a translation to produce a destination dataset that is greater than the sum of its source components.

Geometric Transformation

Geometric transformation is the term used to describe restructuring of the spatial (or geometric) component of a feature within FME.

What is Geometric Transformation?

Geometric Transformation is the act of restructuring the spatial component of an FME feature. In other words the physical geometry of the feature undergoes some form of change to produce a different output.

Some examples of geometric transformation are...

- Generalization – a cartographic process that restructures data to be more easily visualized at a given map scale.
- Warping – adjustment of the size and shape of a set of features to more closely match a set of reference data.
- Topology Computation – conversion of a set of linear features into a node/line structure

Attribute Transformation

Attribute Transformation is the term used to describe restructuring of the non-spatial (or attribute) component of a feature within FME.

What is Attribute Transformation?

Attribute Transformation is the act of restructuring the non-spatial component of an FME feature. In other words the attributes relating to the physical geometry undergo some form of change to produce a different output.

Some examples of attribute transformation are:

- Concatenation – joining together of two or more attributes.
- Measurement – measuring a feature's length or area to create a new attribute
- ID Creation – creation of a unique ID number for a particular feature

Address1 Suite 2017
Address2 7445-132nd Street
City Surrey
Province British Columbia
PostCode V3W 1J8
Concatenate Address1+", "+Address2+", "+City+", "+Province+", "+PostCode
Output Suite 2017, 7445-132nd Street, Surrey, British Columbia, V3W 1J8

Above: Example of attribute concatenation. Each line of the address is concatenated, along with some constants to get spaces and commas, to return a single line address.

About Data Transformation

Transformers provide powerful ways to transform data as it moves from the source system to the destination system.

When the output from the source type or a transformer is connected to another transformer, Workbench makes an implicit connection between all attributes that have the same name. If necessary, Workbench automatically assigns attributes to transformers based on both the transformer itself (usually one specific attribute like *count* or *length*, for example) and then appends the attributes contained in the input connection.

What is Data Transformation?

Data Transformation is FME's ability to restructure data. The key feature of this ability is that transformation automatically takes place between the reading (extract) and writing (load) of data, without requiring user intervention.

Data Transformation Types

Data transformation can be subdivided into two distinct operations.

Transforming Structure

This type of transformation might be better called "reorganization". It refers to the ability of FME to channel data from source to destination in an almost infinite number of arrangements. This would include the ability to merge data, divide data, reorder data, and define custom data structures.

Transforming the structure of a dataset is carried out by manipulation of its schema.

Transforming Content

This type of transformation is also known as "restructuring". It refers to the ability to alter the content of a dataset; manipulation of a feature's geometry or attribute values is the best example of how FME is able to transform content.

Relationship to FME Functions and Factories

If you are familiar with FME mapping files, the transformers provided in Workbench encapsulate the FME Functions and Factories. Transformers manipulate your source data to achieve the desired output. Just like functions and factories, some transformers add attributes to features, others erase attributes, and still others will only transform the geometry. Transformers can operate on individual features, one at a time, or on groups of features.

You can create and store your own Custom Transformers.

Workbench transformers are listed in the Transformer gallery.

About the Feature Inspector

Using the Feature Inspector can significantly reduce the amount of time it takes to author and test a workspace.

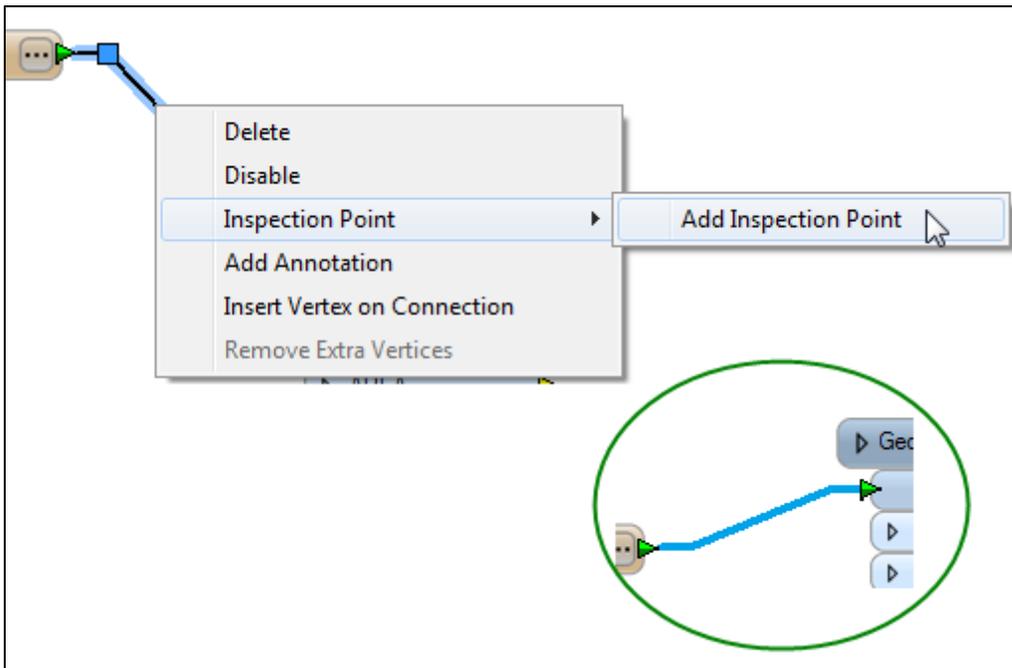
Instead of looking at the log window for information during or after a translation, or routing output to Visualizers so you can view the features, you can analyze features in real time as they move through the workspace.

- You can inspect the properties (attributes and geometry) of an individual feature without having to send the workspace to a Visualizer.
- If you want to add a transformer, especially to a large workspace or a workspace that reads a great deal of input data, you can see the effect it will have on a feature without having to wait for the entire workspace to complete.

How to Enable Feature Inspection

Before you can run a translation with feature inspection, you will have to set at least one inspection point in the workspace.

Select a link until it is highlighted and showing squares on both ends of the link. Right-click to display the command menu and click Inspection Point > Add Inspection Point. You can also set an inspection point by selecting the link and pressing F9. When the inspection point is defined, it will appear in the workspace as a bold, blue line.



There are several ways to use the Feature Inspector in a workspace:

- If you already know the areas of the workspace that you want to inspect, you can set the inspection points. When you run the workspace, the Feature Inspector will stop at each inspection point.
- You can set the workspace to stop at each link in the workspace (even if you have not set it as an inspection point).
- You can run through the workspace, ignoring all inspection points, and pause the processing at any location.
- You can set specific conditions and the Feature Inspector will stop at only the links that meet those conditions.

To run a workspace using the feature inspection capability, click the Run Translation with Inspection tool:



Note that there are three Run buttons. Make sure you click the one that starts the Feature Inspector.

You can also click Inspection > Run Translation with Inspection or use the Shift + F5 keyboard shortcut.

Using the Feature Inspector

How to Use the Feature Inspector

About the Feature Inspector Window

Editing Inspection Point Parameters

Using the Feature Inspector

This Feature Inspector window displays when you run a translation with feature inspection enabled. It contains a viewer, information on the feature's properties, and processing control buttons.

Feature Inspector



Geometry (Polygon)



Contents

Properties

Feature Type	state_OUTPUT
Coordinate System	Unknown
Dimension	2D
Number of Vertices	70238

Bounding Box

Minimum Values	-168.123504638672, 54.7665061950684
Maximum Values	-129.989532470703, 71.3980484008789

Attributes

AREA	267.357
DAY_ADM	3
fme_basename	statesp020
fme_color (utf-8)	0.481408411861122,0.631178149874871,0.2111649469
fme_dataset	C:\Workbench_Data\UrbanAreas\States\statesp020.shp
fme_feature_type	statesp020
fme_fill_color (utf-8)	0.681408411861122,0.831178149874871,0.4111649469
fme_geometry	fme_polygon
fme_type	fme_area
MONTH_ADM	January
multi_reader_full_id	1
multi_reader_id	1
multi_reader_keyword	SHAPE_2
multi_reader_type	SHAPE
ORDER_ADM	49
PERIMETER	374.768
SHAPE_GEOMETRY	shape_polygon
STATE	Alaska
STATE_FIPS	02
STATESP020	2
YEAR_ADM	1959

Geometry

IFMEPolygon

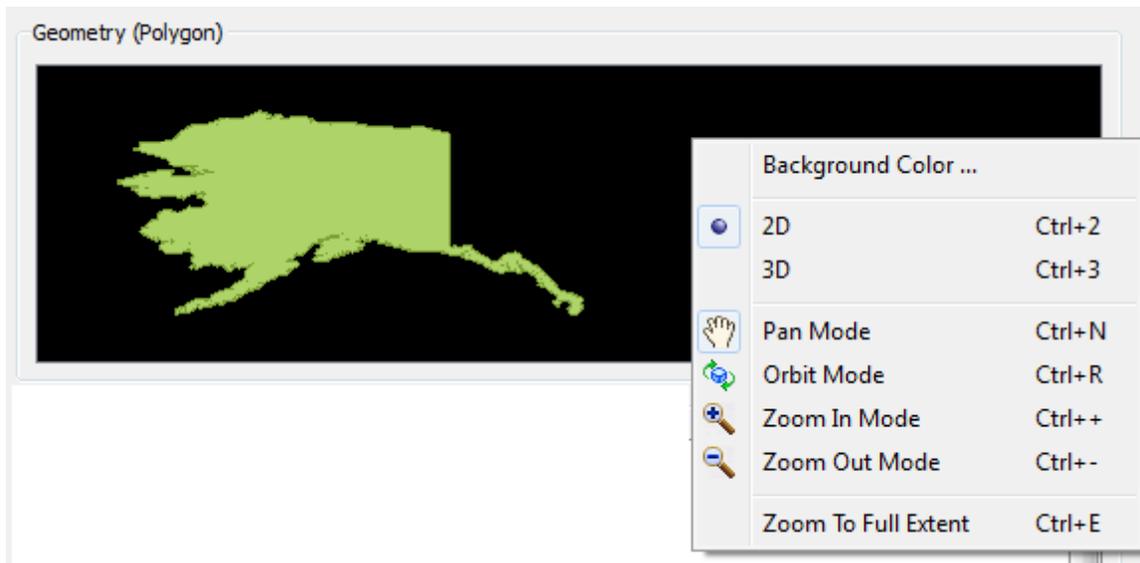
Boundary: IFMELine	70238 Point(s)
--------------------	----------------



Viewer

The viewing area displays features for verification and debugging purposes. When the Feature Inspector stops at an inspection point, the feature will appear in the viewer. You can perform basic zoom-in and zoom-out functions, pan the area, and change the background color. In this example, the background has been changed from the default.

You can also rotate (Orbit Mode) three-dimensional features.



Contents

Property	Value
Properties	<p>The feature that is being analyzed by the Feature Inspector. This area lists the:</p> <ul style="list-style-type: none"> • feature type, • coordinate system (if applicable), • dimension (2D or 3D), • number of vertices, • minimum and maximum bounding box values.
Attributes	<p>This area lists the number of attributes and their values. The types of attributes displayed here are:</p> <ul style="list-style-type: none"> • User Attributes: pieces of user-created information that belong to a feature. • FME attributes: start with fme_ and define the feature within FME. • Format attributes: define the feature within the source format.
Geometry	<p>This section displays all the geometry information for the currently selected feature. You can expand or collapse the arrows next to the items to view or hide the details for each.</p> <p>Information displayed here depends on the feature, but might include the coordinates of the bounding box (or cube in a 3D feature), a full listing of all geometric information for the feature (coordinates for all geometry types, information about measures, transformation matrixes, raster bands, and so on...)</p>

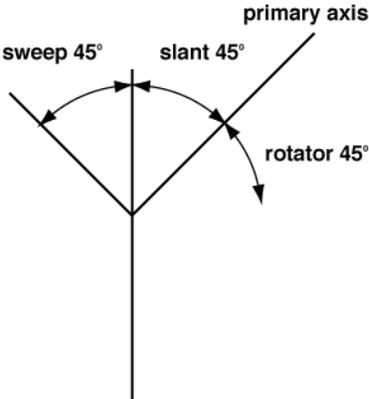
Control Buttons

	Step the translation to the next link, whether or not it is a designated inspection point.
	Step the translation to the next inspection point.
	Continue translation without stopping.
	Stop translation. Pressing this button will clear the Feature Inspection window. You can restart it by pressing the  button.

fme_type

In addition to the **fme_geometry** attribute which indicates what the coordinates of an FME feature are, each FME feature also has an **fme_type** attribute which controls the interpretation of those coordinates. For example, a feature with **fme_geometry** of **fme_point** may be used to represent a point, a text object, an arc, or an ellipse. The value of the **fme_type** attribute is used to indicate which interpretation should be made.

The **fme_type** attribute can have one of a set number of values. Depending on the value of **fme_type**, there may be additional attributes required to fully interpret the geometry. The following table lists the allowed values for **fme_type**, the associated **fme_geometry**, and its additional attributes.

fme_type	fme_geometry	Additional Attributes
fme_arc	fme_point	<p>fme_rotation: The rotation of the primary axis in degrees counterclockwise from the primary axis. If not set, then 0 is assumed.</p>  <p>fme_primary_axis: The length of the primary semi-axis of the defining ellipse measured in ground units.</p> <p>fme_secondary_axis: The length of the secondary semi-axis of the defining ellipse measured in ground units. For circular arcs this value will be equal to the fme_primary_axis.</p> <p>fme_start_angle:</p> <p>Refer to the @Arc (function) in the FME Functions and Factories manual for a detailed definition of start_angle.</p>

fme_type	fme_geometry	Additional Attributes
		fme_sweep_angle: Refer to the @Arc (function) in the FME Functions and Factories manual for a detailed definition of sweep_angle.
fme_area	fme_polygon fme_donut fme_aggregate	None
fme_collection	fme_aggregate	None
fme_ellipse	fme_point	fme_rotation: The rotation of the primary axis in degrees counterclockwise from horizontal. If not set, then 0 is assumed. fme_primary_axis: The length of the primary semi-axis of the ellipse measured in ground units. fme_secondary_axis: The length of the secondary semi-axis of the ellipse measured in ground units. For circles this value will be equal to the fme_primary_axis.
fme_line	fme_line fme_aggregate	None
fme_no_geom	fme_undefined	None
fme_point	fme_point or fme_aggregate	None
fme_raster	fme_aggregate	None
fme_solid	fme_aggregate	None. <div style="border: 1px solid black; padding: 5px;"> fme_solid is the fme_type for the following geometries: IFMEEextrusion, IFMEBox, IFMEBRepSolid, IFMECSGSolid, and IFMECompositeSolid. For a description of FME Surfaces and Solids, see "FME 3D Support" on page 74. </div>
fme_surface	fme_aggregate	None. <div style="border: 1px solid black; padding: 5px;"> fme_surface is the fme_type for the following geometries: IFMEFace, IFMERectangleFace, IFMETriangleStrip, IFMETriangleFan and IFMECompositeSurface. For a description of FME Surfaces and Solids, see "FME 3D Support" on page 74. </div>
fme_text	fme_point	fme_rotation: The rotation of the text in degrees counterclockwise from horizontal. If not set, then 0 is assumed. fme_text_string: The actual annotation string. fme_text_size: The height of the text measured in ground units.
fme_point_cloud	fme_polygon	None.

Any features with an `fme_geometry` value of `fme_aggregate`, and an `fme_type` other than `fme_collection` must be homogeneous aggregates. An `fme_type` of `fme_collection` allows the feature to have heterogeneous aggregate geometry.

FME 3D Support

FME supports translation of 3D geometry from one format to another, provided that each format itself supports 3D. The extent of 3D support is limited by the level of each format's own 3D support. If one format supports solids and another supports only faces, the solids may be converted to faces during the translation. Because of this, each combination of reader and writer will have its own unique characteristics and should be tested. It is also important not to assume a particular transformer will support 3D until you have verified it first.

As with other enhancements to FME, the first step is the extension of core infrastructure and new geometry model, in this case to handle 3D constructs. Over time, this enhanced 3D support will be implemented throughout FME transformers and formats where this provides the most value.

Surfaces and Solids

IFMESurfaces

IFMEFace

A face is a planar area in 3D space. The planar structure can be a polygon, an ellipse, or a donut (IFMEArea).

The planar area has a concept of a surface normal, a vector that points outwards perpendicular from the area. The direction of the surface normal in an IFMEFace is determined by using the right-hand rule - if the fingers of your right hand curl along the order of the vertices, the direction that the thumb points to is the direction of the surface normal.

Contains: IFMEArea (1)

Contained by: IFMECompositeSurface, IFMEMultiSurface

IFMECompositeSurface

IFMEMesh

IFMEMultiSurface

IFMERectangleFace

IFMETriangleFan

IFMETriangleStrip

IFMESolids

IFMEBRepSolid

A solid is a solid volume in 3D space, defined by a collection of connected surfaces. The exterior surface is represented by an IFMECompositeSurface, with the additional requirement that this composite surface must be closed, in order to form a volume.

An IFMEBRepSolid can contain spatial voids. Each spatial void is represented by an IFMECompositeSurface, representing an interior surface. This definition is analogous to the inner boundaries that define donut holes.

In general, an IFMEBRepSolid must contain one exterior surface and zero or more interior surfaces.

Contains: IFMECompositeSurface (0..n)

Contained by: IFMECompositeSolid, IFMEMultiSolid, IFMECSGSolid

IFMEExtrusion

IFMEBox

IFMECSGSolid

IFMECompositeSolid

IFMEMultiSolid

Editing Inspection Point Parameters

By default, the Feature Inspector stops at every inspection point that you define.

You can also set inspection parameters so it will stop only when certain conditions are met.

From the menu bar, select Inspection Point > Edit Inspection Point. You can also right-click on a defined inspection point and select Inspection Point > Edit Inspection Point.

Inspection Point Parameters

Name: statesp020 [statesp020 [SHAPE]] -> GeometryFilter : INPUT

Conditions

Test Clauses:	Left Value	Operator	Right Value
1	value	=	5
2	value	Between	10-20
3			

Pass Criteria: One Test (OR)

Test Expression: 1 OR 2

Comparison Mode: Automatic (compare as numbers if possible)

Stop

Always

When hit count is equal to

Current hit count: 2

Help OK Cancel

Name

This is the default name that the Feature Inspector assigns to the inspection point. You can rename any inspection point.

Conditions

Check this box to define test clauses for the inspection point.

Test Clauses

The **Value** columns may be a literal constant, an attribute name preceded by the value-of operator (&), or an attribute value function. If it is an

attribute value function, the function will be executed on the current feature and the result will be used for the test.

The operands can consist of:

- An ampersand followed by the name of a feature attribute: this takes the "value-of" the attribute and uses that as the operand for the test: `&<attribute name>`

You can pick the attribute name from the pull-down list in the grid, or you can type it with the leading `&`.

- Any FME function: `@FunctionName(argument1, argument2, ...)`

The function must be typed in directly into the grid, and must follow the FME syntax for functions. Choose *Help > FME Functions and Factories* to see a complete list of functions that can be called.

- Any constant value: anything not starting with an ampersand (`&`) or at-sign (`@`) is considered to be a constant: `anyConstant`

You can type the constant in the Test Clause grid.

The grid dialog for the Tester makes it easy to specify any of the above inputs to a test or set of tests.

The Operator column is one of `<`, `>`, `=`, `==`, `!=`, `>=`, or `<=`.

Pass Criteria and Test Expression

The Pass Criteria defines how multiple clauses are interpreted in the final classification of the incoming feature.

You can choose one of three test scenarios:

Scenario	Pass Criteria	Description																
One test is required for the input feature to be classified as PASSED.	One Test (OR)	In this case, as long as one of the test clauses is true, then the feature is PASSED. This is an OR test (test1 OR test2 OR test3). If any one is true, then the result is true.																
All tests are required for the input feature to be classified as PASSED.	All Tests (AND)	This is stricter than One Test (OR) because all tests must pass in order for the result to be true (test1 AND test2 AND test3).																
Create your own test expression. This is useful when you need fine-grained control over when you want the Feature Inspector to stop.	Composite Test	<p>If, for example, you want to check whether the value of an attribute is between 5 and 10, or equals 99, you can set up three test expressions:</p> <p>Clause 1 : $x > 5$</p> <p>Clause 2 : $x < 10$</p> <p>Clause 3 : $x = 99$</p> <p>(where x is the selected attribute in the Left Value field):</p> <table border="1" data-bbox="760 1486 1247 1648"> <thead> <tr> <th></th> <th>Left Value</th> <th>Op.</th> <th>Right Value</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>value</td> <td>></td> <td>5</td> </tr> <tr> <td>2</td> <td>value</td> <td><</td> <td>10</td> </tr> <tr> <td>3</td> <td>value</td> <td>=</td> <td>99</td> </tr> </tbody> </table> <p>To correctly get the desired results, you require that clause 1 AND clause 2 be true (between 5 and 10), OR clause 3 is true (equals 99).</p> <p>In this case, choosing One Test or All Tests modes will not satisfy the test requirement. You can, however,</p>		Left Value	Op.	Right Value	1	value	>	5	2	value	<	10	3	value	=	99
	Left Value	Op.	Right Value															
1	value	>	5															
2	value	<	10															
3	value	=	99															

Scenario	Pass Criteria	Description
		<p>choose Composite Test and enter the following expression in the Test Expression field:</p> <p>((1 AND 2) OR 3)</p> <p>The numbers above correspond to the test clauses defined in the Test Clauses table. When read, the composite expression above states that <i>Clause 1 AND Clause 2 must be satisfied, OR Clause 3 must be satisfied.</i></p>

Comparison Mode

By default, the Comparison Mode is set to *Automatic (compare as numbers if possible)*. This means the Tester will first try to convert the operands to numbers. If it is successful, it will compare them as numbers. If it is still not successful, it will treat operands as strings.

Alphanumeric Strings: Say you have a string that is labeled "4E5". If you choose *Automatic (compare as numbers if possible)*, it is possible that it will be treated as a number. If you want it treated as a text string, set the Comparison Mode to *String (always compare as text strings)* – this ensures that the Tester will always treat the operands as strings, and will compare them as strings.

Stop

- Always
- A *hit* is registered every time a condition is met. Enter a number in the text box, and the Feature Inspector will stop only when the hit count is equal to, less than, greater than, or a multiple of that number.

The current hit count is shown on the right.

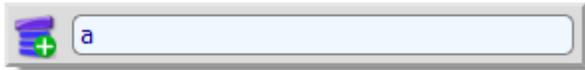
What is Quick Add?

Quick Add is the easiest way to add transformers to the workspace, without having to search the Transformer Gallery.

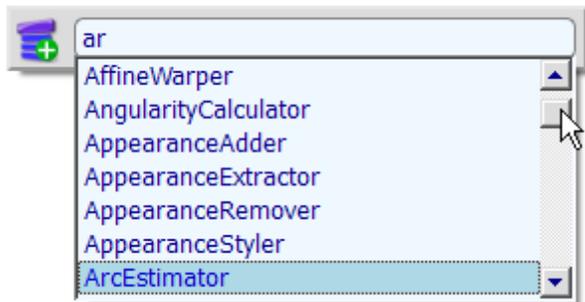
How Does Quick Add Work?

Quick Add is a search dialog that is embedded in the workspace canvas. To try it, click on a blank area on the workspace canvas and press a key on the keyboard.

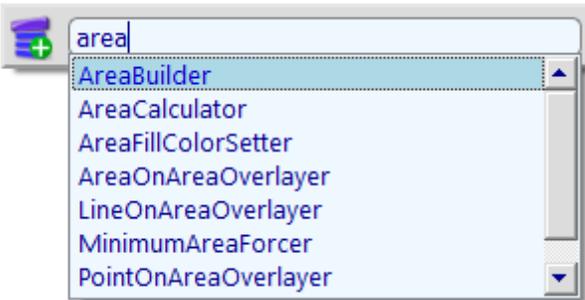
For example, type the letter *a*. When you press the key, a small search bar appears on the canvas:



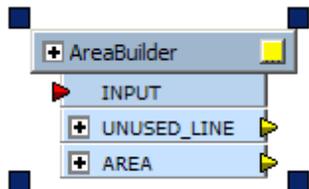
As you continue to type (in this case, the letter *r*) then a list of transformers whose name matches this content starts to appear. The list here contains a lot of transformers, so you can scroll down the list.



To further refine your search, type an entire word – in this case, the word *area* refines the list until only those transformers that contain the word *area* appear in the search box:



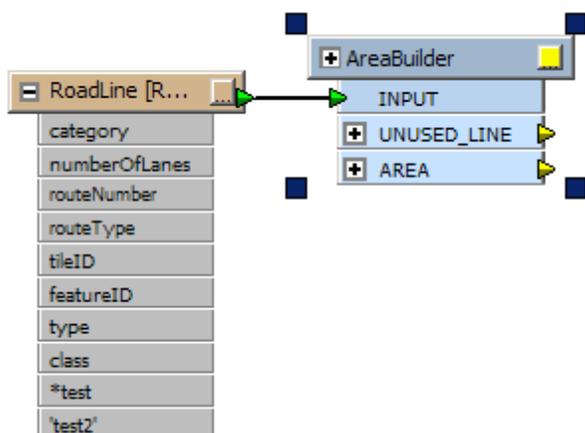
Press Return, and the highlighted transformer appears in the workspace. Press Return again to immediately open the Properties dialog.



Tip: To return even more search results (for example, transformers that may not include the word *area* in the name but may include the word *area* in the description, scroll down to the bottom of the Quick Add list. Click on **{Search for "area"}** and the Transformer Gallery will also be included in the search. The results will display all transformers that have *area* as part of their name or their description.

Auto-Connect After Quick Add

Choose Tools > FME Options, and click the Transformers icon. Make sure that *Auto connect after Quick Add* is checked. Then, if you select a Feature Type before initiating the Quick Add search, the transformer will connect automatically to the feature type. If you choose the Option *Quick Add placement follows mouse*, the transformer will be placed underneath your cursor.



Shortcut: If you add a transformer, then want to add the same transformer, press the slash "/" key on your keyboard. The Quick Add box will appear showing the last selected transformer. Press Return to include it, then Return again to edit its parameters.

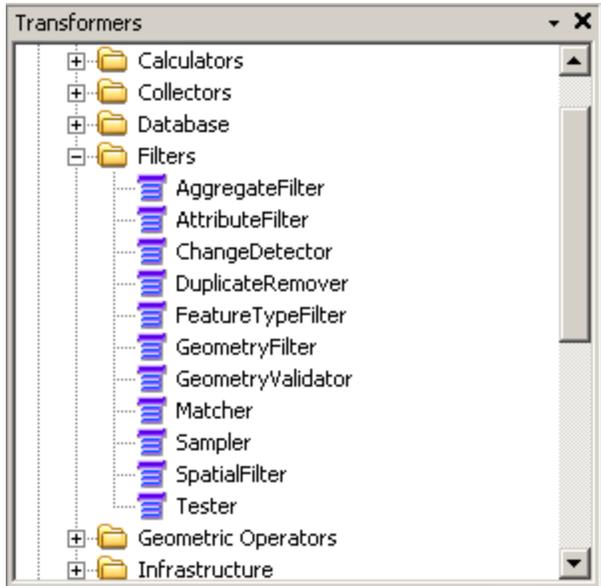
Introducing Transformers

In FME Workbench, source and destination data is represented by objects in the workflow. In the same way, both geometric and attribute restructuring are depicted by objects called Transformers.

What is a Transformer?

A transformer is an FME Workbench object that carries out the restructuring of features. A number of different transformers exist to carry out different types of restructuring.

The layout of windows in Workbench may vary, but by default the Workbench navigation pane will have a tab labelled Transformers. Click this to view FME's transformer list.

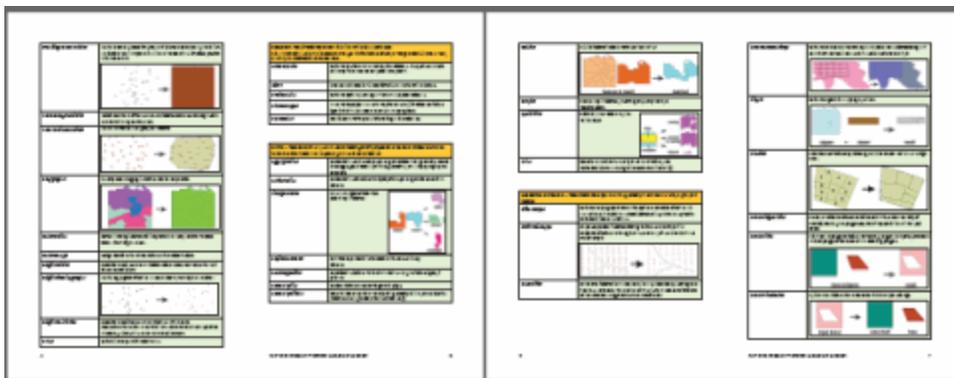


The list is segmented into descriptive category folders, but you can see the entire list by opening the All folder.

FME Workbench Transformers Reference Guide

Click the link below to open the guide (from Safe Software's website). You can also right-click on the link and choose *Save Target As* to save the file directly to your computer or another location.

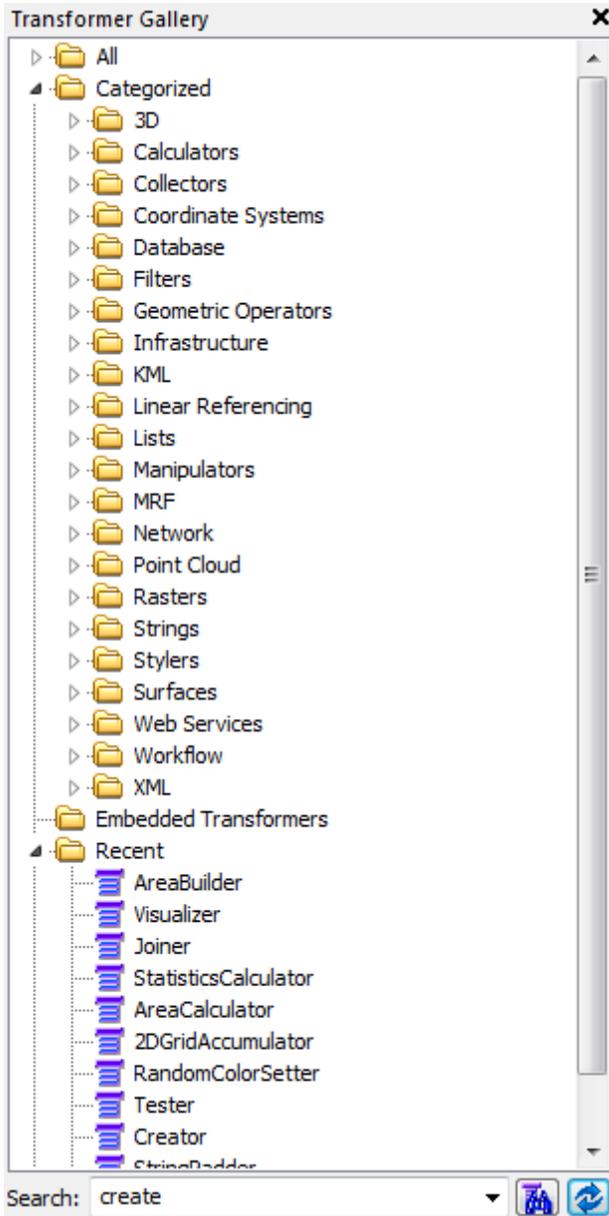
- [FME 2011 Transformers Reference Guide \(PDF format\)](#)



Note that you must have Adobe Reader to open the PDF file. You can get it from [here](#) for free.

Locating Transformers

FME contains hundreds of transformers, and it can sometimes be challenging to navigate the list to locate the ones you need. By default, the Transformer Gallery is a tab beneath the Workbench Navigator pane. Click the Transformer Gallery tab to display the transformer folders.



Transformer Categories

Transformer categories are a good starting point from which to explore the Transformer Gallery. Transformers are grouped in categories applicable to their associated functionality.

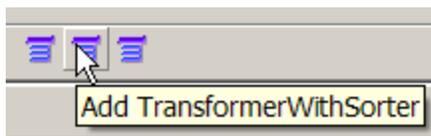
- 3D: Create and modify 3D surface and solid geometries.
- Calculators: Calculate a value and supply it as a new attribute.
- Collectors: Use or create collections or groups of features.

- Database: Interact with external databases.
- Filters: Split and reroute data
- Geometric Operators: Process feature geometry.
- Infrastructure: Provide interaction with the underlying FME translation engine facilities. These include functionality to log features, set feature colors, create individual features and grids of features from nothing, and invoke the FME Universal Viewer on features flowing by.
- KML: Manipulate feature geometry and/or attributes for output using the OGCKML Writer.
- Linear Referencing: Work with linear referencing data structures on FME features.
- Lists: Work with list attributes.
- Manipulators: Operate on individual features (the opposite of Collectors).
- MRF: Repair geometry, particularly during data migration from CAD to GIS. They are built upon the MRFCleanFactory, which is an integration of MRF Geosystems Corporation's cleaning technology into FME. These transformers are all extra-cost.
- Network: Operate on linear features that are connected in a network, performing operations such as priority calculation and orientation correction.
- Point Cloud: Create, use, and output point cloud features. They operate only on data consisting of point clouds.
- Rasters: Create, use, and output rasters. They operate on data consisting of a regularly spaced grid of values.
- Strings: Create, modify and delete string (character) attributes.
- Stylers: Prepare features for output to particular formats by providing a convenient interface for setting color and other display characteristics
- Surfaces: Create, use, and output surfaces.
- Web Services: Access web services via the HTTP protocol.
- Workflow: Run workspaces either locally or on an FME server.
- XML: Work with XML data by mapping XML elements into FME features, using stylesheets to convert XML documents, and query collections of XML data.

You can also customize the Transformer Gallery by creating your own folders and storing your favorite transformers there for quick access. These folders are saved as an external definition, so you can share them with other users.

Including a Transformer in the Toolbar

For one-click access, you can drag a transformer directly from the Transformer Gallery and place it on the toolbar. Float your cursor over a transformer to see its name.



Searching for a Transformer

If you're not sure which transformer you need, you can also search transformer descriptions in the Transformer Gallery.

Enter a keyword in the Search field. The keyword can be a partial Transformer name, or one or more keywords that describe its function. Press Enter or click the  button.

Workbench displays a Search Results folder that contains a list of transformers whose name or descriptions contain the matching keyword. You can also enter the name of an FME Function or Factory if you want to find its corresponding transformer.

Click on one of the transformers to see a description of its functionality. This help text displays in the Transformer Description pane.

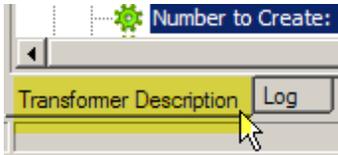
Using Quick Add

Quick Add is the easiest way to add transformers to the workspace.

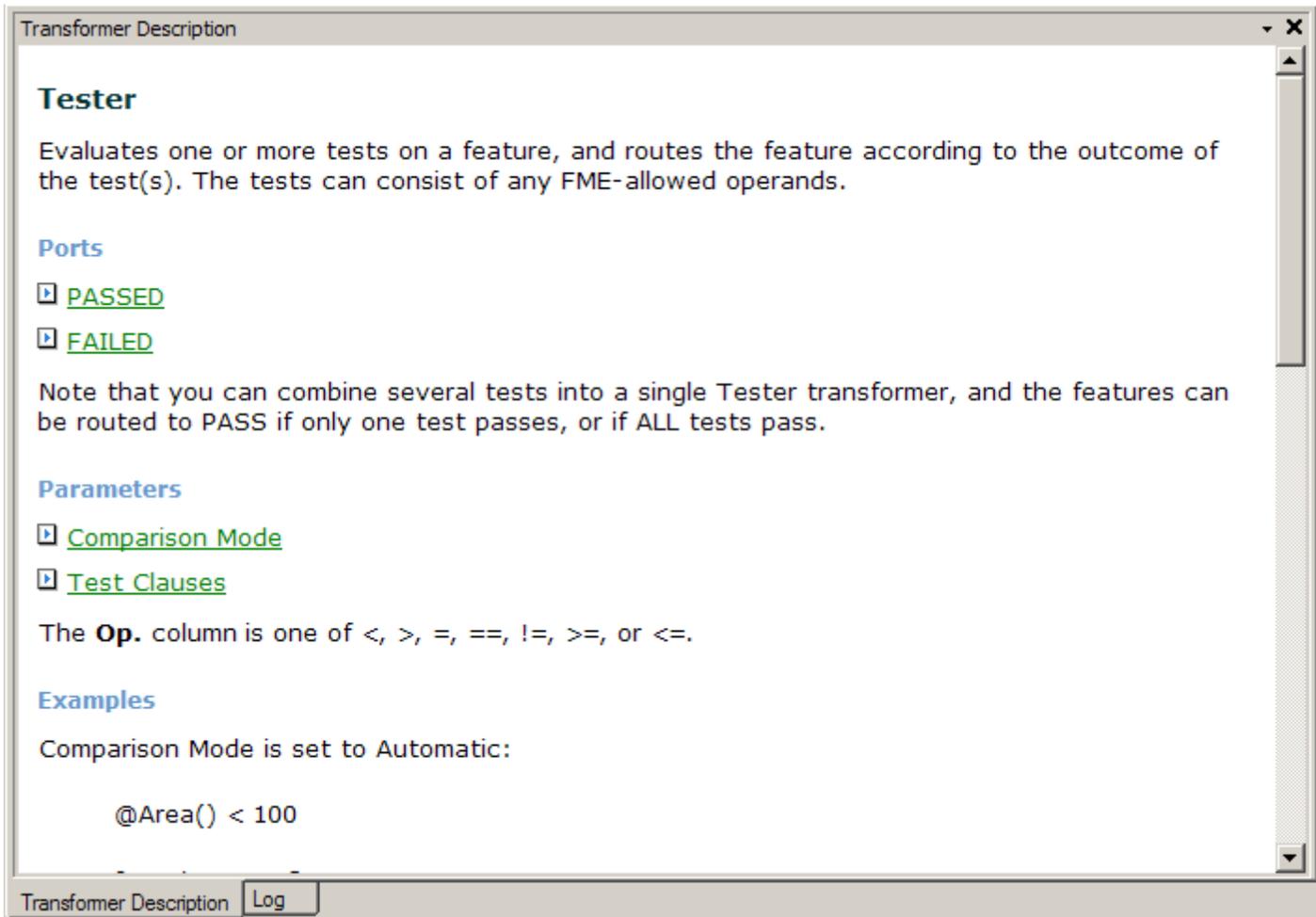
Finding Transformer Help

We have reorganized, rewritten, enhanced and improved the transformer help. We have included information on input and output ports, parameters, licensing, and technical history. And, in many cases, help topics now include direct links to fmepedia examples and related transformers.

To display the Transformer Description window, click the tab next to the Log tab. If the Transformer Description tab is not displayed by default, choose View > Windows > Transformer Description.



If you want the Transformer help visible at all times, you can separate this window from the main Workbench interface. You can also access the same information in a separate help file by choosing Help > FME Transformers.



Transformer Properties

Click the Properties button to the top right of each transformer (or right-click a transformer and select Properties) to display a dialog that contains default properties that Workbench initially set for the transformer.

In most cases, you can use the transformer without changing the default values; sometimes, however, you will need to edit the values before you can use the transformer. See Transformer Defaults for information on saving default values.

Properties Button Colors

The properties button on a transformer is color-coded to reflect the status of the settings.

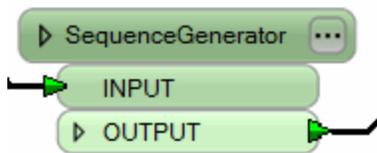
	A blue properties button (or one that matches the color of its transformer) indicates that the default transformer settings have been checked and amended as required, and that the transformer is ready to use.
	A yellow properties button indicates that the default settings have not yet been checked. The transformer can be used in this state, but the results may be unpredictable.
	A red properties button indicates that there is at least one setting for which FME cannot supply a default value. The setting must be provided with a value before the transformer can be used.

Transformer Colors

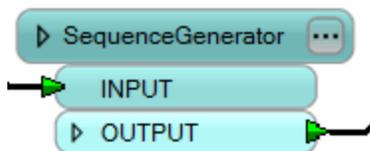
All regular transformers are blue.



Custom transformers (which are covered in later topics) have two different states: embedded or linked. Embedded custom transformers are green:



Linked custom transformers are cyan:



Some transformers, such as the Visualizer, have their own distinctive icon:



Transformer Defaults

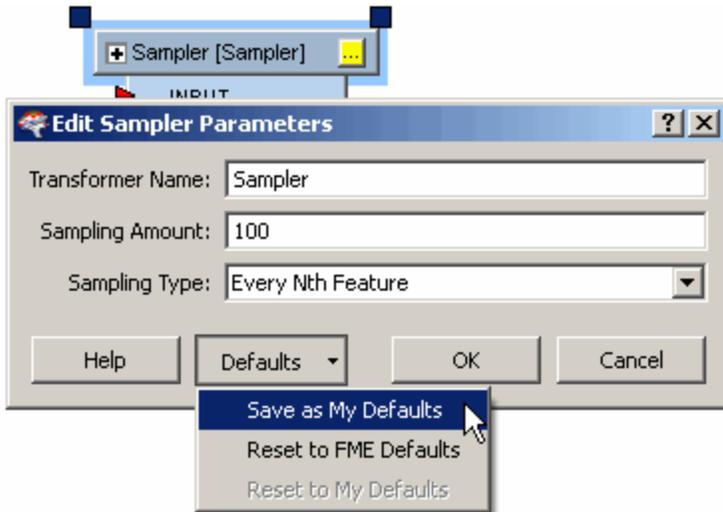
Especially if you are a regular FME user, you will notice that you are always using the same transformers, and the same parameters in these transformers (for example, the same password in a Joiner or the same tolerance in a Snapper). Transformer defaults allow you to override FME defaults and save these parameters in individual transformers.

All transformer dialogs have a Defaults button, with the following options:

- Save as My Defaults: Edit a field, then choose this option to save the parameter.
- Reset to FME Defaults: Changes the field back to the standard (FME) defaults
- Reset to My Defaults: If you manually edit this field, you can reset it back to your own defaults. Note, however, that you cannot restore your own defaults after resetting to FME defaults.

This example sets the default values of the Sampler to sample every 100 features (the usual default is 1).

You can set the sampling amount to 100 and then choose Save as My Defaults.

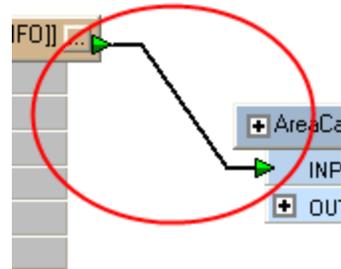
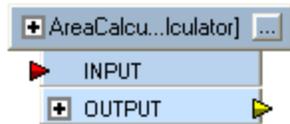


Now whenever you place a Sampler transformer, its parameters will take your own values as defaults.

Including Transformers in a Workspace

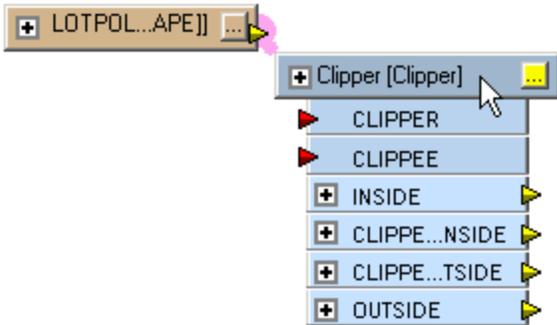
To include a transformer in your workspace, double-click on it with your left mouse button. Or select it and then just drag and drop it onto the canvas.

You can manually connect it to a feature type, attribute or another transformer by dragging the yellow arrow to the desired place.

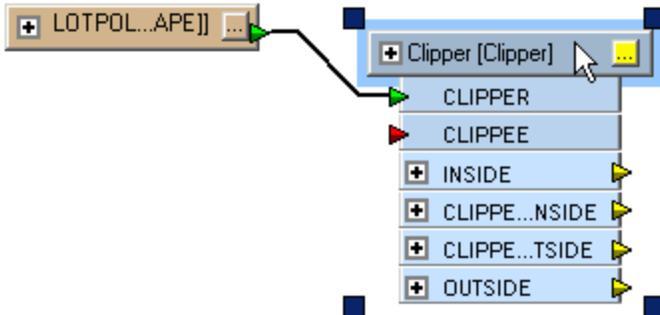


Click on the yellow arrow, and drag the connection to the appropriate red arrow.

You can also connect a transformer is to drag it close to the connection port until you see a highlight.



Release the left mouse button. You may be prompted to select the input port if there is more than one.



You can then manually connect output ports to their destinations, or use the Insert Before function.

Quick Connect

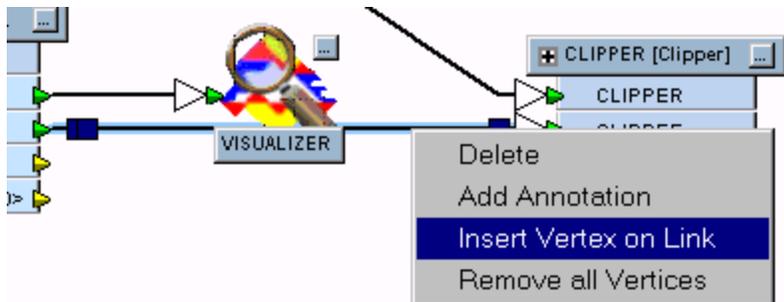
Connect transformers using Quick Connect.

Tips:

- To get help on a transformer once you place it in your workspace, select the transformer and press the F1 key.
- Using annotations throughout your workspace can help you document useful information.
- Showing Summary Annotations can be useful to display information on a printout.
- You can easily insert transformers into an existing workspace without having to manually reconnect the ports.

Adding Vertices to Links

Do you have many connections that you'd like to route around other transformers and feature types? Select the connection, click the right mouse button, and then select Insert Vertex on Link. (Or, you can double-click the connection.) You can drag the vertices anywhere on the link, and use them to route your connection around parts of your workspace.

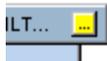


To remove vertices, select the connection, click the right mouse button, and then select Remove all Vertices.

Duplicating Transformers

To copy a transformer to another part of your workspace, select it and then choose **Duplicate** from the command menu. The new transformer will appear in the workspace.

NOTE: The properties button will change back to yellow on the [duplicated transformer](#) to remind you that you may need to edit the new parameters.



Removing Transformers

Select a transformer and press **Delete**. If you have many transformers to remove, choose **Remove Transformers** from the Tools menu and check the box beside each transformer.

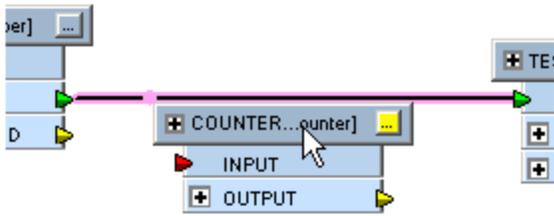
Connecting Transformers

Using the transformer Drag-and-Insert function is the easiest way to insert and connect transformers into an existing workspace. It virtually eliminates the need to manually re-establish connections.

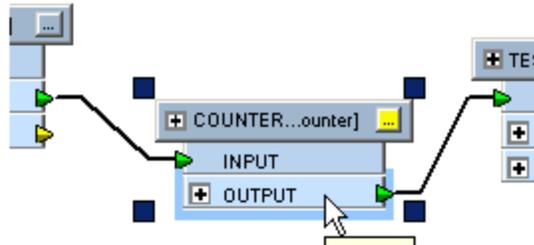
Inserting a Transformer into an Existing Connection

You can "drag and insert" a transformer into an existing connection. When you release the transformer, it is set in place. A wizard sets destination node and output port (if there's more than one). FME then makes feature and attribute connections automatically. If either object has multiple input or output ports, you'll also be prompted to choose which specific port(s) to connect.

Tip: You can change the default port for the automatic input and output connections. See [FME Transformers Options](#).



Drag the transformer over a pipeline until it highlights in pink.

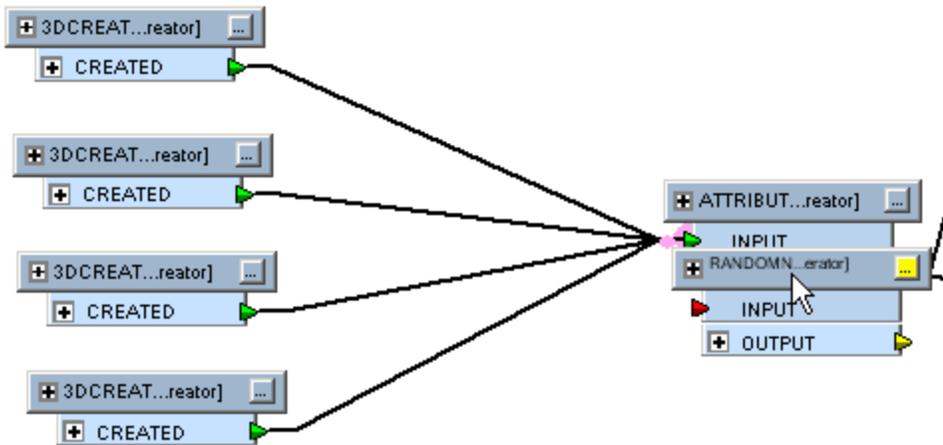


Release the left mouse button and the transformer will be placed and connected.

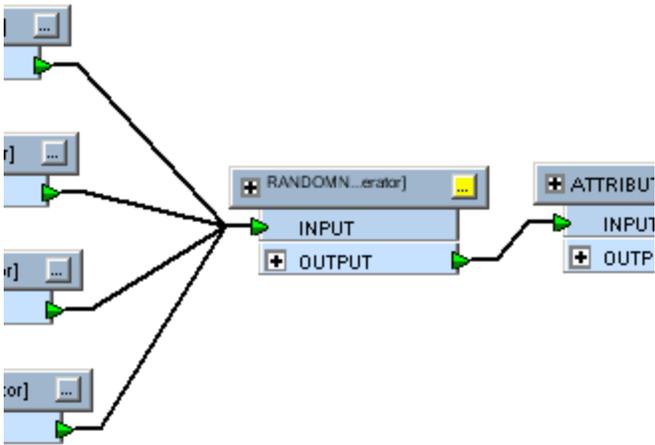
Setting Multiple Connections

You can also insert into multiple pipelines at once by highlighting a transformer input/output port.

This transformer is dragged to the proper input port (highlighted in pink):



When it is released, the transformer is now in position.

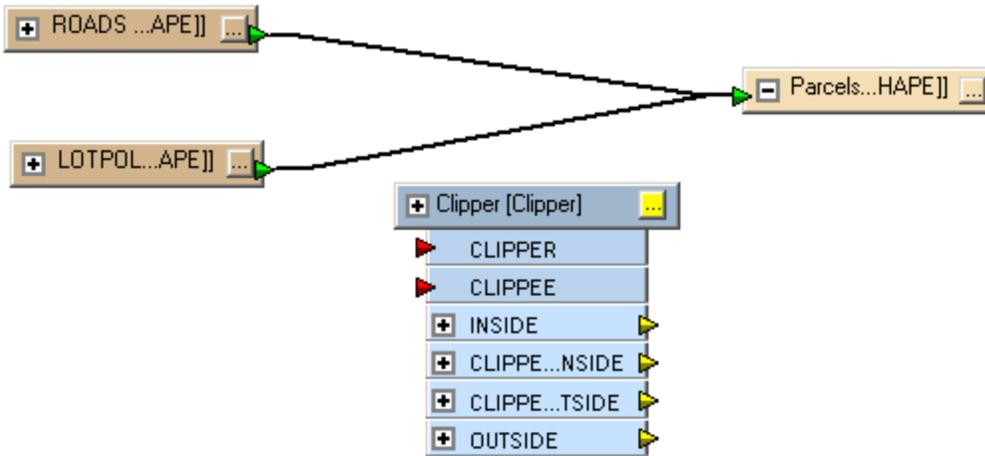


Tip – Reversing the Insertion Point: Sometimes it can be difficult to drop a transformer into the required position because the body of the transformer can hide the connection or port into which it needs to be inserted. You can sometimes work around the problem by pressing the ALT key when you are dragging the transformer. This causes the insertion highlight point to be switched to the opposite corner of the transformer, making it possible to highlight the required port without blocking it from view.

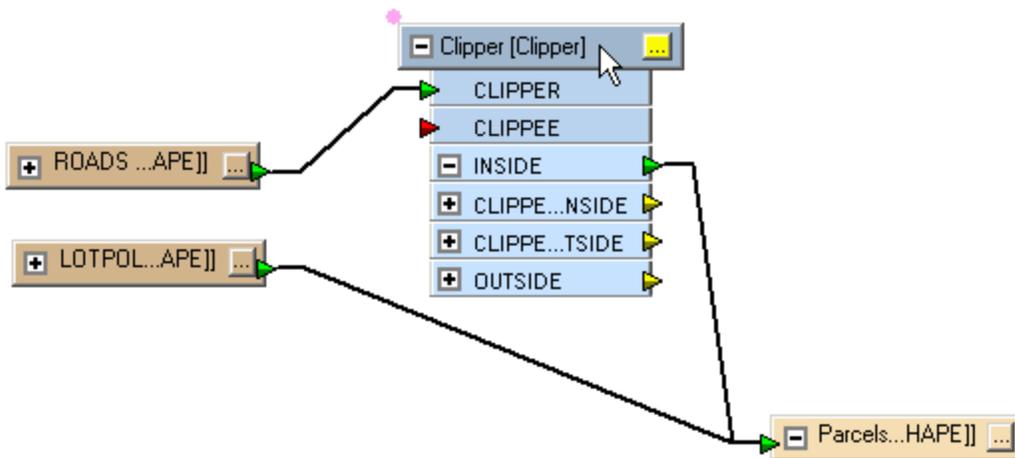
Inserting a Transformer with Separate Connections

You can connect a single transformer into multiple input ports that require separate connections.

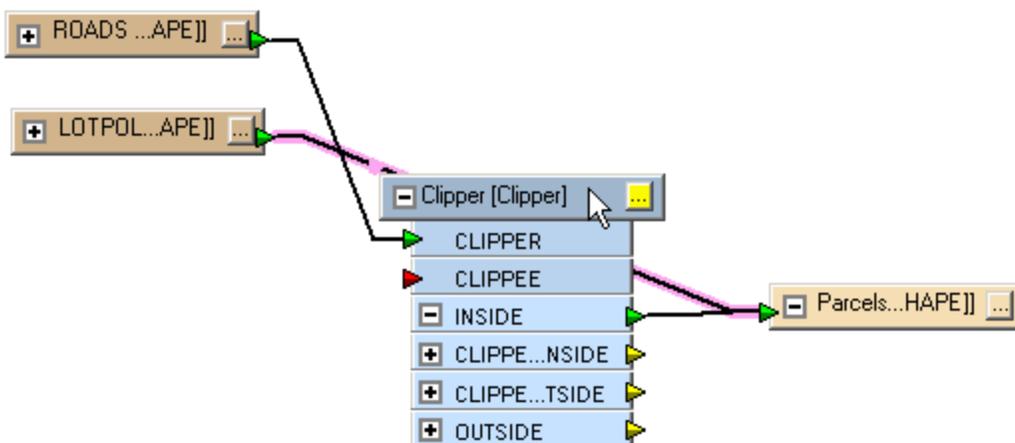
This example shows how to insert a Clipper transformer. In this case, there are multiple input ports requiring separate connections.



The first connection is easy to make with the drag-and-insert function...



Dragging a transformer that is already inserted still causes a highlight point to appear, and connections to be highlighted. You can make a second insertion with the same transformer, but onto the other link...



The Clipper is now correctly inserted.

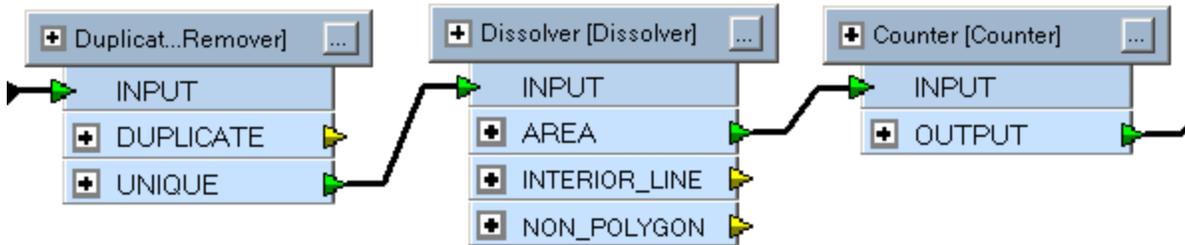
Note: You cannot insert a transformer in a position that would cause a loop in the workflow – such invalid connections will not even be highlighted.

Disabling the Drag-and-Insert Function

This feature is enabled in Workbench by default. To disable it and use manual connections only, click the drag-and-insert tool .

Using a Series of Transformers

Despite the large number of transformers available in FME, you will frequently find that a single transformer does not meet all your requirements. In this situation, you will need to use a combination of transformers. A string of joined transformers graphically represents your overall workflow and is a key concept of FME.



Above: In this example a DuplicateRemover transformer removes duplicate polygon features. A Dissolver transformer merges each remaining (unique) polygon with its neighbour where there is a common boundary. Finally each merged area gains an ID number from the Counter transformer.

Creating Your Own Transformer Folders

By default, Workbench stores Transformers in default categories. However, you can customize the gallery to suit your workflow purposes.

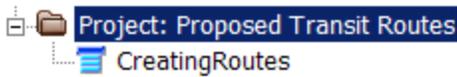
Creating Custom Folders in the Gallery

You can create custom folders in the Transformer Gallery, and drag transformers in and out of folders.

Collapse the folders in the Transformer Gallery, and right-click in the blank area below the folders. Click the New Folder menu that appears:



Type a folder name in the dialog and the new folder will appear in the Gallery. You can drag any transformer in the Gallery to the new folder.



Sharing Transformers from a Custom Folder

Right-click on the custom folder:

- Export... sends the list of transformer it contains to a file (for example, Favorites.fmxlist).
- Send To... opens your default e-mail client and attaches the list to a new e-mail.

Installing an .fmxlist File into the Gallery

An .fmxlist file is a text file that contains a list of transformers, which you can easily install into the Transformer Gallery.

Note that you will have to restart Workbench following this installation.

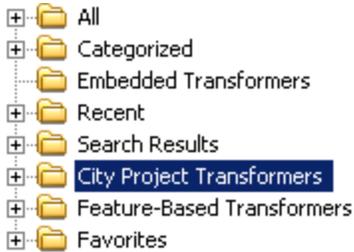
To install the new folder into your Transformer Gallery, you can either drag an .fmxlist file onto the Transformer Gallery window, or double click the file. Note that if you already have a folder with the name specified in the file, you will be prompted to overwrite the name, install it under a new name, or cancel the operation.



You will see this message:



When you click Yes, the list will be installed. Restart Workbench to see the list in the Transformer Gallery.



Transformer Version Numbers

Each transformer contains an independent version number (not related to the FME version). Transformer version numbering runs 0, 1, 2, 3, where a brand-new transformer is version 0.

Whenever a transformer is updated or fixed, the transformer version number increases. That's the only time the version number changes – it doesn't automatically increase with a new FME version. Some transformers might never be changed and still have a version of 0, whereas others might change 3 or 4 times between FME releases.

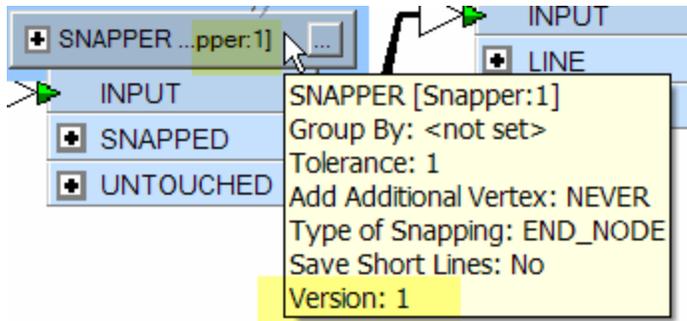
When you place a transformer into a workspace, the version of that transformer is fixed. As long as that transformer remains in the workspace, its version will also remain the same. This applies even if you open an older workspace using a newer version of FME.

Displaying Transformer Versions

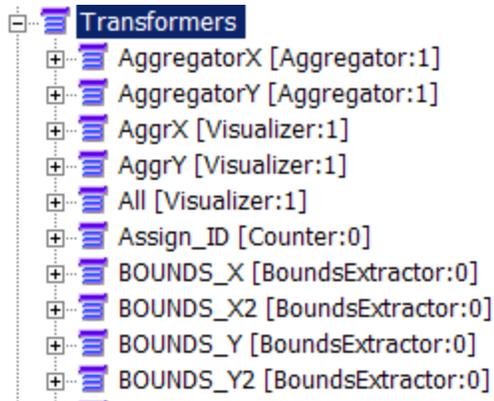
In **FME 2007** (or newer) there is a setting in the **Tools > Options** (Transformers) for enabling the display of transformer versions. By default, the box is unchecked.



If you check the Show Transformer Version box, the workspace will display the transformer version in the transformer name (preceded by a colon), as well as in the tooltip that appears when you float the cursor over the transformer:



The transformer version also appears in the list of transformers in navigation pane:



What if you have an older workspace and transformers with different version numbers?

A transformer's version is the version defined by that FME, so you can have a workspace that contains different versions of the same transformer. For example, if you open a workspace that has a version 2 Bufferer, and you want to place a new Bufferer, the new one may be version 3 (or 4, 5, 6).

Can you open a newer workspace in an older version of FME?

You can't open a workspace when it contains a transformer with a newer version than what is available in the current version of FME. For example, if you open an FME 2007 workspace in FME 2004, there might be a conflict in FME. That's because you may have a transformer version 4 in the workspace, while FME 2004 supports only version 1.

Enabling and Disabling Transformers

You can disable a transformer to temporarily remove it from the translation.

Use Ctrl+click to select multiple transformers, or press the left mouse button and drag the cursor over a specific area. Choose Transformers > Enable/Disable Transformers.

You can also right-click and select Disable Transformer from the command menu.

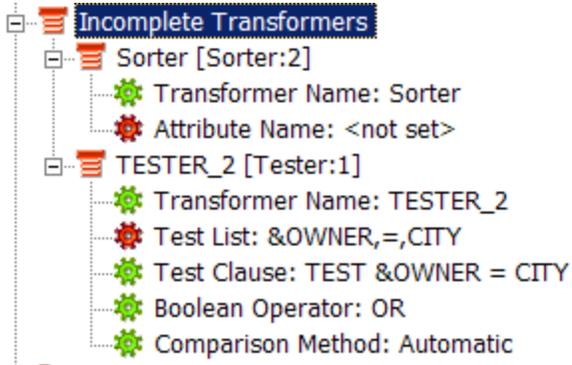
Note: Especially for larger workspaces, it is sometimes better practice to disable connections.

What are incomplete transformers?

Once a workspace becomes a complex series of parallel streams it's important that you are able to test each stream separately, and also be aware of how changing one component can impact other sections further on in the workflow.

If you see an *Incomplete Transformers* node in the Navigator, it means that the workspace contains one or more transformers that need parameters filled in before the transformer will be operational. Any transformer that has a red properties button with an exclamation point  will appear in the list. (Transformers with a yellow properties button will not appear in this list, since they will not prevent the workspace from running.)

The Navigator displays an Incomplete Transformers node, and the problem parameter is shown in red. Most times, you will just need to enter or set a value, the parameter icon will change to green, and the transformer will be removed from the "Incomplete" list.



Interacting Edits

Occasionally, this situation can arise when changes to one transformer cause the settings in another to become invalid.

FME will automatically flag transformers when they are adversely affected by changes elsewhere. It does this by resetting the transformer properties button back to red, to indicate that attention is required.

The individual setting that is causing the problem is also separately highlighted. FME takes changes into consideration for the entire workspace, not just the changed transformer.

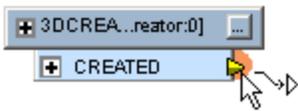
About Quick Connect

Quick Connect is designed to help users author workspaces more easily than ever before. Instead of having to drag connections between different objects in the canvas, you can now connect them by simply clicking on the ports.

Although you can use this feature for all workspace connections, the most useful application for this functionality is when you have a large workspace and you need to make connections from one area to another. Usually you would zoom out until both areas are in view, but this could be difficult if the connection to be made is extremely large, or if the workspace is very dense. This feature lets you choose the source port, locate the area that you want to connect, then choose the destination port to complete the connection. For example, you can initiate Quick Connect, click a bookmark in the Navigator pane, and complete the connection once the workspace centers on the bookmark.

How to use Quick Connect

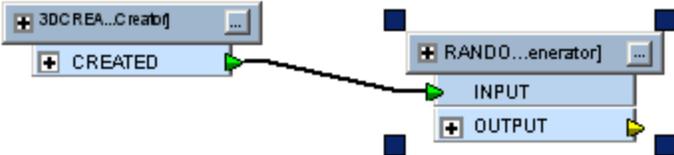
Click on any output port to be connected. An orange circle highlights the port and the cursor changes shape, to  :



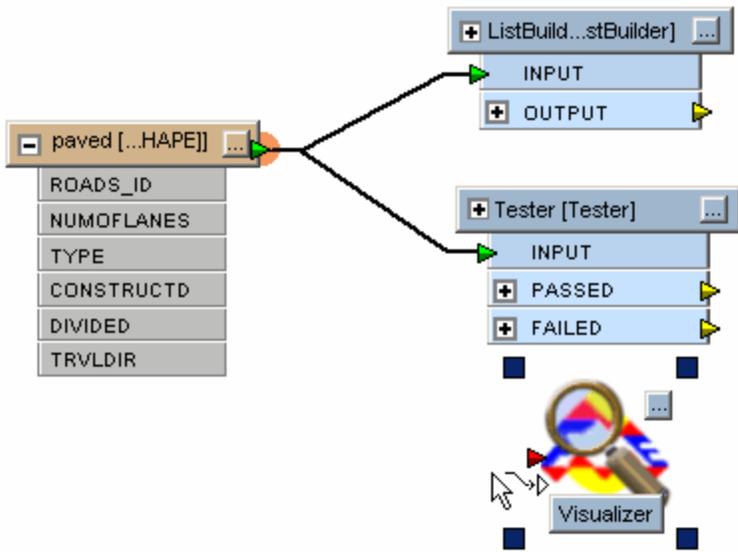
Then hover directly over the input destination port:



And click to complete the connection:



What if you have multiple ports to connect? Just hold down the Ctrl key before you establish the first connection, then click additional input ports:



About Bookmarks

A bookmark is way to define areas of your workspace for easy access. The bookmark shades an area, usually one that is carrying out a specific task, so you can easily move to it in the workspace.

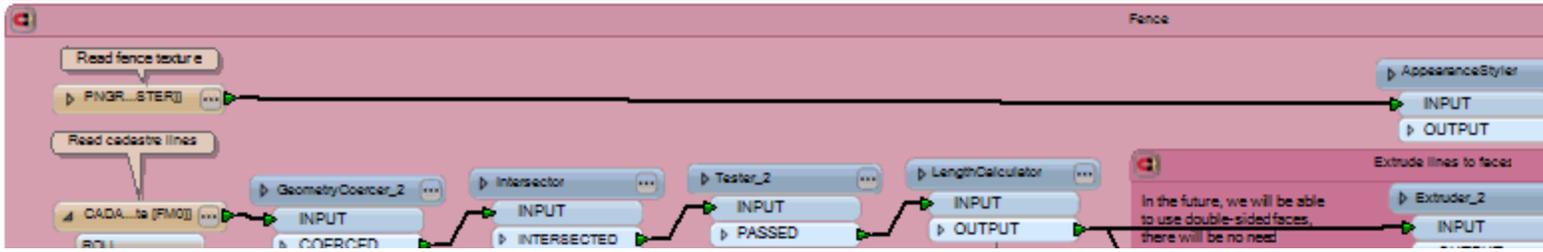
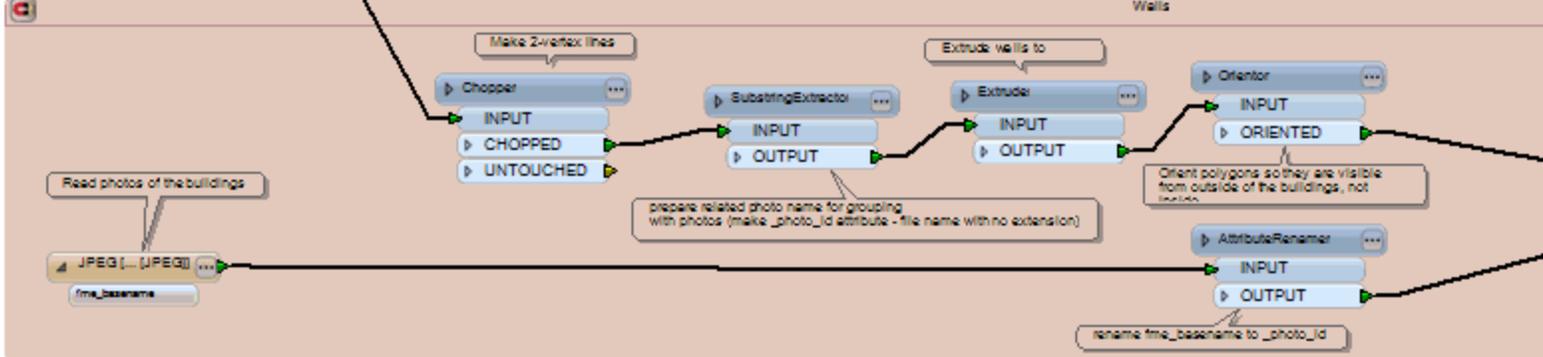
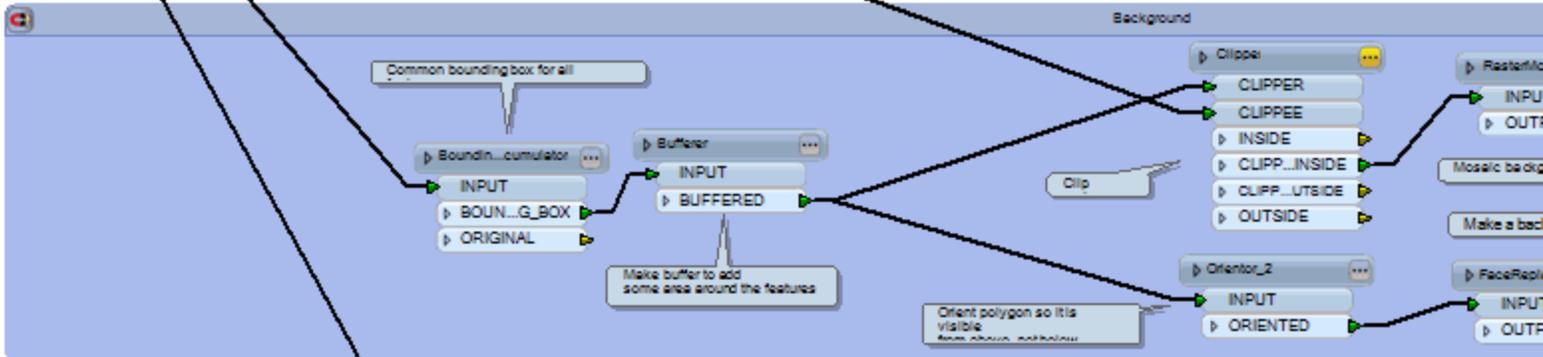
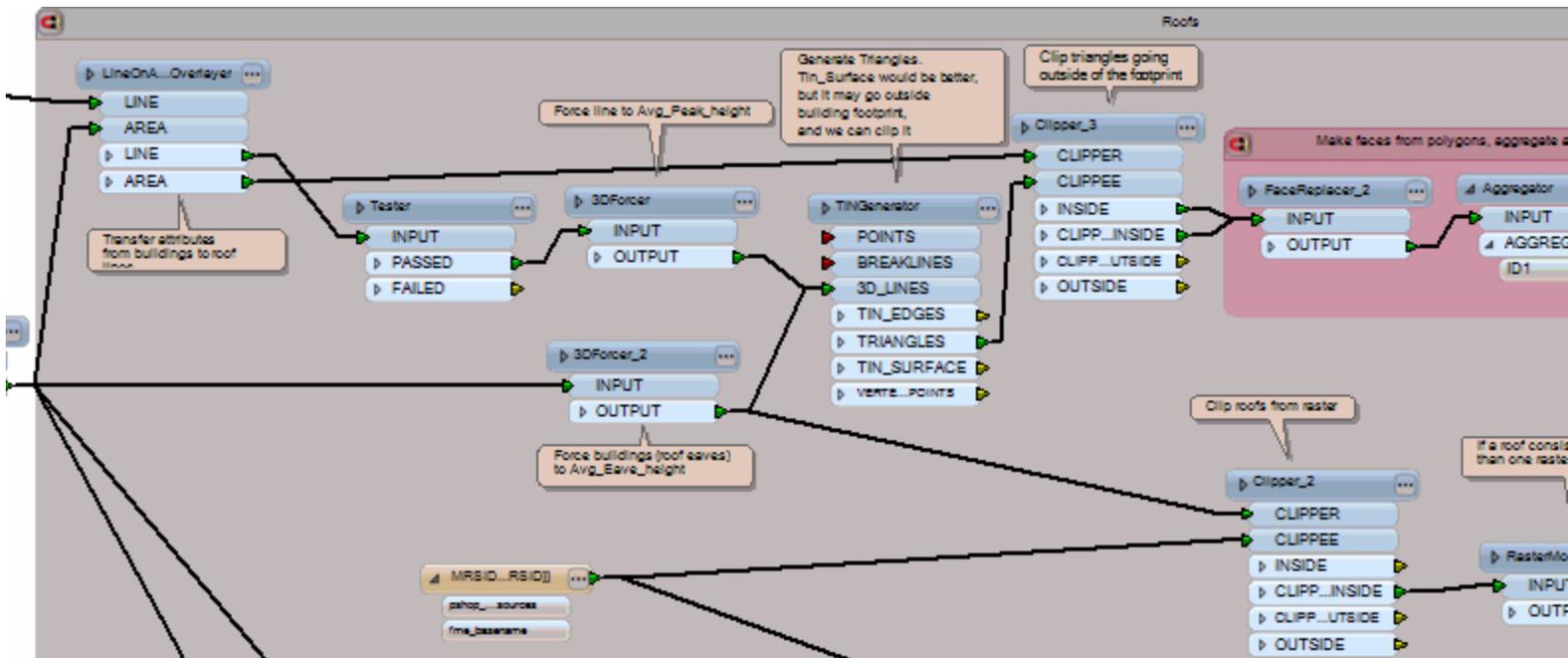
Why use Bookmarks?

Bookmarks play an important part in FME for a number of reasons:

- places a marker for quick access to a certain area
- divides a workspace into different, clearly marked sections
- organizes a workspace so you can move sections of transformers at one time

See Using Bookmarks in a Workspace.

This workspace is a good visual example of how bookmarks can help with the organization of a workspace (mouse over the image to see a full view):



Other than using bookmarks for general organization and ease-of-use, here are some additional reasons:

- Remembering why you created a workspace: Bookmarks can help remind you why you created a workspace, and how it works.
- Deciphering someone else's workspace: Bookmarks can clarify the purpose of a workspace, and provide the necessary context for determining how it works.

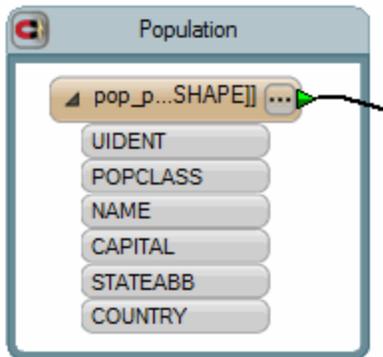
Using Bookmarks in a Workspace

Adding a New Bookmark

Selecting an area of the workspace (for example, a group of transformers) before adding the bookmark ensures that the bookmark appears in the selected area.

Click the bookmark tool  or select Insert > Bookmark.

A bookmark will appear in the workspace. The text field in the bookmark will be highlighted so you can enter a new name as soon as the bookmark appears. (Note that if the characters in your name exceed the width of the bookmark, the name will be truncated).



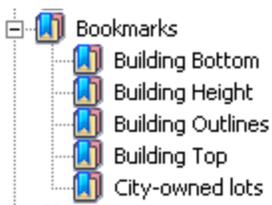
Changing the Background

You can create bookmarks so that they have a filled background. Choose Tools > FME Options > Workbench and check the box *Draw bookmarks with a filled background*.



Viewing Bookmarks in the Navigator

All bookmarks are added as a list in the Navigator:



Resizing a Bookmark

Left-click the bookmark, hold down the mouse button, and drag the bookmark's handles.

Enabling the Bookmark Magnet

By default, the bookmark's contents are attached to the bookmark. When you move it around, the contents follow. This makes it a useful tool for organizing workspaces: you can move entire sections around the canvas until they are in a good layout.

Click the magnet to toggle it on and off.



Magnet is on (contents are attached):



Magnet is off (contents are separate):

Moving a Bookmark

Left-click on the bookmark's border and hold down the mouse button to move the bookmark to another location on the workspace.

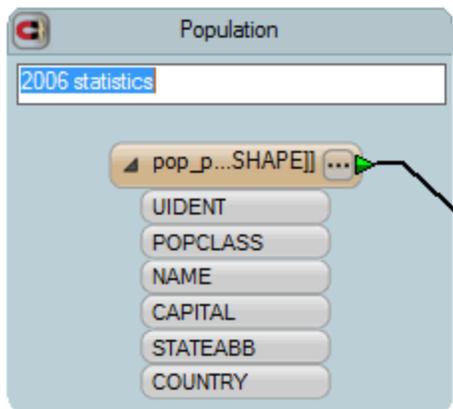
Changing a Bookmark's Properties

Double-click on the colored area of the bookmark's title bar to display the Edit dialog (or select a bookmark, then right-click and choose Properties). The Edit dialog allows you to change the bookmark name and its color properties.

To rename a bookmark, double-click a bookmark name in the workspace.

Adding a Description

1. Select a bookmark and right-click.
2. From the command menu, select Edit Bookmark Description.
3. Enter a description to appear inside the bookmark. You may need to resize the bookmark to make the description more visible.



Zooming to a Bookmark

To zoom to a selected bookmark:

- Right-click to display the Edit menu and choose Zoom to Bookmark, or
- Double-click the bookmark name in the Navigator.

Panning to a Bookmark

To center the workspace on a selected bookmark:

- Click (once) on the bookmark name in the Navigator.

Exporting Bookmarks to a Custom Transformer

1. Right-click to display the Edit menu, and choose *Create Custom Transformer from all Objects in Bookmark*.
2. The Custom Transformer Properties dialog appears. Enter a name and description for the new transformer and click OK.

The objects that were inside the bookmark appear in a custom transformer window, under a new tab. The bookmark that was in the original workspace is replaced by a link to the custom transformer.

[More about custom transformers](#)

About Published and Private Parameters

There may be cases where you want to be able to enter or verify a parameter value every time you run a workspace, or share a single parameter value across multiple parameters. You can do this by defining a parameter. Parameters can include:

- source and destination datasets
- feature types
- certain transformer values
- log files
- coordinate systems

You can also set many parameters to the value of one parameter. In this case, when you want to change the value of all parameters, you only have to change it once, in the main parameter.

There are two parameter types: Published and Private.



Published or Private Parameter?

By defining a Published Parameter, you can be prompted for any information that might change when the workspace is run. This makes the workspace even more portable. All published parameters are also shown in the log file as they would appear on the command line, which allows for simpler migration of workspaces to a command-based environment.

Private Parameters are used internally, mainly to share a parameter value across multiple transformers. For example, multiple SchemaMappers may reference a single parameter for its schema mapping table dataset. You can also create parameters for username/password fields in commonly accessed database tables.

Private Parameters are nearly identical to Published Parameters, but they have the following differences:

- Prompt-and-run  will not prompt for the parameter values.
- When used in custom transformers, Private Parameters will not appear in transformer properties. They will appear in the Navigator as an uneditable parameter.
- The command line, at the top of the translation log and `.fmw` file, will not show the command line argument to set this variable. (However, since they are macros in the mapping file, you can still modify them.)

You can always convert between parameter types by right-clicking on the parameter in the Navigator and selecting Convert to Published Parameter or Convert to Private Parameter.

Using a Parameter

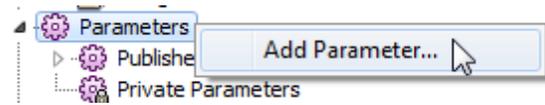
Once you create a parameter, you can use it in several ways:

- Right-click on the parameter and select *Apply To* to apply that parameter to another setting in the workspace.
- Right-click any workspace setting in the Navigator and select *Link to Parameter*. The resulting dialog contains a list of any newly created parameters (Published and Private). Pick one and click OK to set the link.
- Use the ParameterFetcher transformer with the parameter name specified.

Examples

fmepedia includes additional information and examples of published parameters.

Adding a Parameter



Right-click on the Parameters icon in the Navigator and select Add Parameter:

A screenshot of the 'Add/Edit Parameter' dialog box. The dialog has a title bar with a question mark and a close button. It contains several fields: 'Type' is a dropdown menu set to 'Choice'; 'Name' is a text input field with a red background; 'Published' and 'Optional' are checked checkboxes; 'Prompt' is a text input field with a red background; 'Configuration' is a text input field with a red background and a three-dot menu icon to its right; 'Default Value' is a dropdown menu. At the bottom are 'Help', 'OK', and 'Cancel' buttons.

You can create many different parameter types and combinations.

Tip: If you create a parameter based on an existing workspace setting, the Type will be automatically chosen. If you have never worked with Parameters, it might be easier to start with an existing workspace setting.

Type

Choice: Creates a parameter that allows you to choose one of two values.

Choice (Multiple): Creates a parameter that allows you to choose from multiple values.

Choice or Text: Enter a text string or select one from a pick-list

Choice with Alias: Add selected readers, writers, or coordinate systems.

Choice with Alias (Multiple): Similar to Choice with Alias but the display alias for this type is mapped to an internal value, and is returned as the parameter value instead.

Color Picker: Creates a parameter for changing the FME color value.

Coordinate System Name: Creates a parameter for choosing a coordinate system.

Directory: Creates a parameter that allows you to choose the name (and path) of a directory.

Filename (Existing): Creates a parameter for choosing the name and path of a file.

Filename (Output): Creates a parameter for choosing the name and path of the output file.

Float: Creates a parameter for choosing a floating point number.

Integer: Creates a parameter for choosing an integer.

Password: Creates a parameter for entering a password.

Scripted (Python): Creates a parameter from a Python script (Private parameter only).

Scripted (Tcl): Creates a parameter from a Tcl script (Private parameter only).

Slider: Creates a bounded numeric value.

Text: Creates a parameter for entering a text string.

Published

If you uncheck this box, the parameter will be created as a Private parameter.

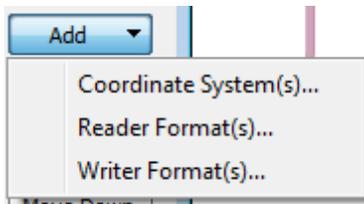
Optional checkbox

It might not be important in every instance to use the value from this particular published parameter. Flagging the parameter as Optional ensures that it can continue with an empty value.

Configuration

The parameters required in this field are dependent on the Parameter Type you choose. Click the browse button beside the parameter to see the applicable configuration choices. For example, you can:

- Populate a list and the contents will be displayed in a pull-down menu when the parameter is published;
- Point to a file or directory;
- Import attribute values from other datasets;
- Add readers, writers and coordinate systems by clicking Add to choose from the galleries:



Name

Enter a name. This will appear under the Published Parameters icon in the Navigator pane.

Prompt

Enter the prompt that will appear when you run the workspace.

Default Value

Choose the default value that will be used.

Usage Notes

- After you create a Published Parameter, run the workspace by clicking Prompt and Run Translation . You will get a good idea how the parameters will affect the workspace just by viewing the prompt dialogs and parameter choices. (Note that this will not work for Private Parameters, since they are not prompted.)
- You can reorder the parameters in the Navigator by dragging them up or down.
- The Reader and Writer dataset fields are published by default.

Modifying a Parameter

Edit

1. Select the parameter, right-click, and select *Edit Value* or *Edit Definition*.
2. Edit the desired fields and click the OK button.

Tip: To quickly change a value only, double-click on a Parameter.

Delete

1. Select a parameter, right-click, and select *Delete*.

Apply to

1. Select the parameter, right-click, and select *Apply to*.
2. In the Apply Published Parameter dialog, click the checkbox next to other workspace parameters.

Note: If you filter the list by keyword, the list will dynamically update as you enter the keyword – clicking OK will close the dialog.

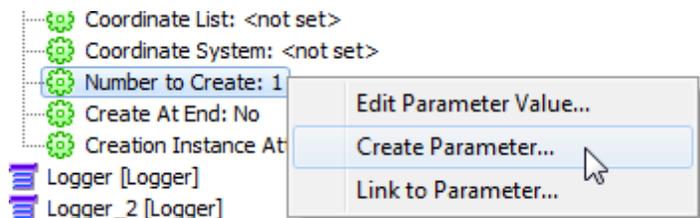
3. Click OK to copy the published parameter to the additional selected parameters.

Converting Between Published and Private Parameters

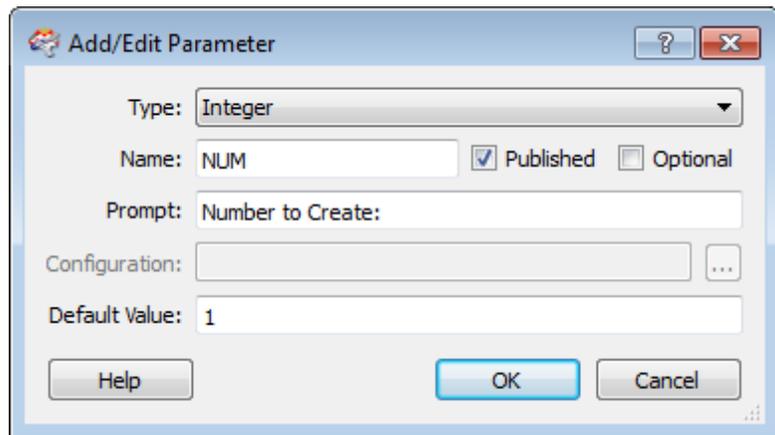
- Right-click on the parameter and select *Convert To Private Parameter* or *Convert to Published Parameter*, or
- Edit the Parameter settings and uncheck the Published checkbox.

Adding a Parameter Based on a Workspace Setting

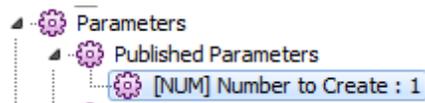
If you right-click on the *Number to Create* parameter shown in this example, you can either edit the existing value, create a parameter, or link to an already existing parameter.



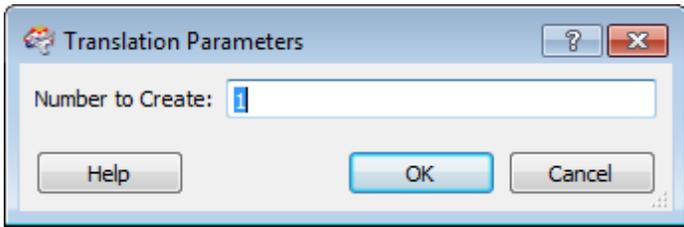
Choose Create Parameter to display the Add/Edit Parameter dialog. The Type field will already be chosen based on the workspace parameter. In this case, the Type is *Integer*.



Click OK to add the Parameter to the Navigator:



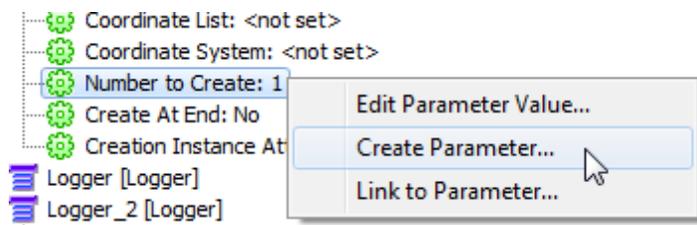
Then, when you click Prompt and Run Translation  to run the workspace, a dialog will prompt you for any changes to the Number to Create parameter:



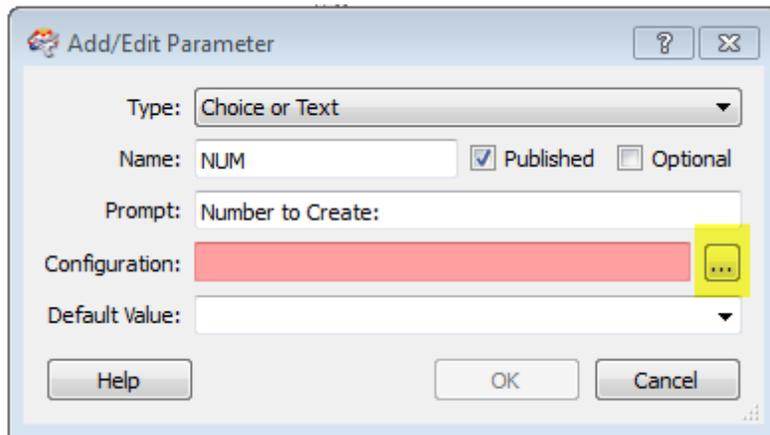
When you click OK, the translation continues.

Creating the Same Parameter with Multiple Choices

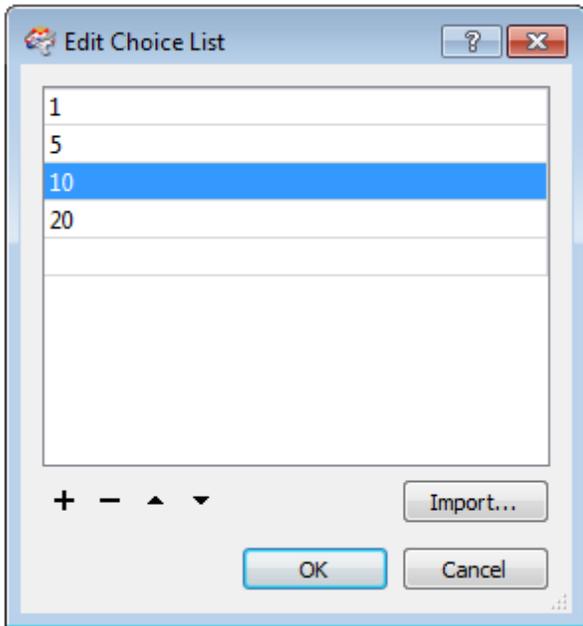
Repeat the first step above:



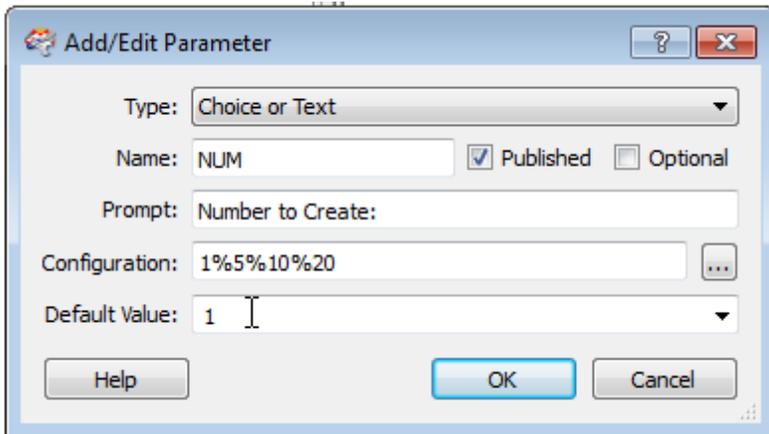
Instead of accepting the default Type, select Choice or Text, and then click the browse button in the Configuration field.



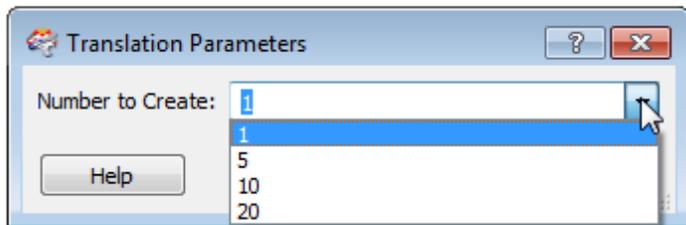
When Edit Choice List appears, enter the choices you want to see, and tab to the next row. Click OK to set the choices.



The choices are now shown in the Configuration field. In the Default Value field, you can enter the default you want to use or leave the field blank.



Click OK to add the Parameter to the Navigator. Then, when you click Prompt and Run Translation  to run the workspace, your choices will be shown in the Number to Create parameter:

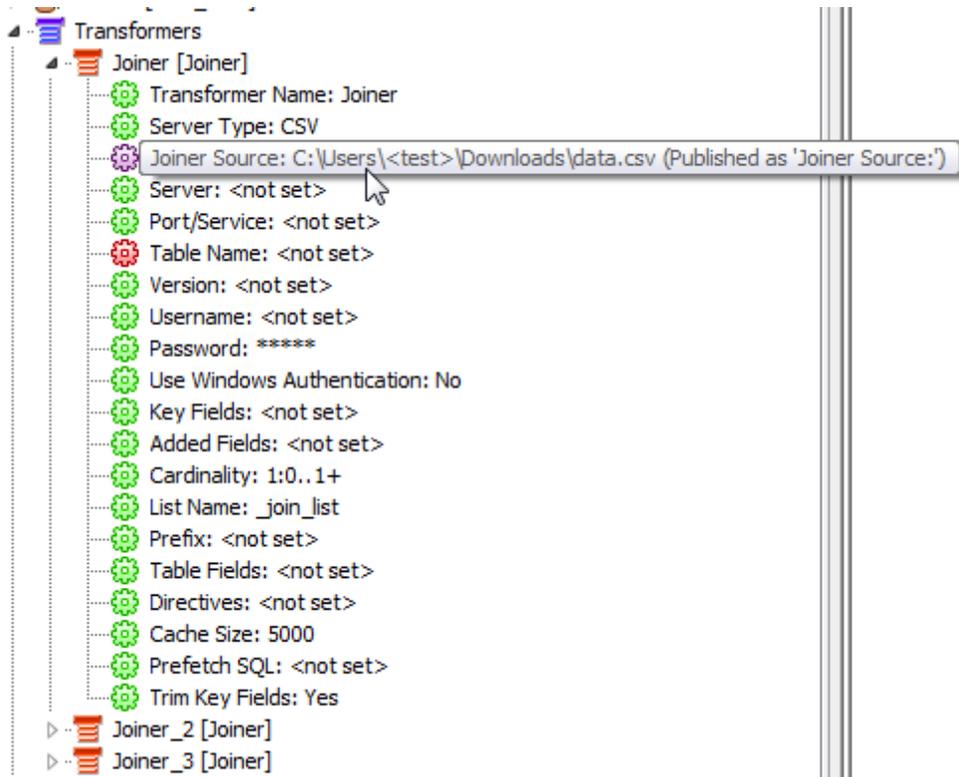


Linking Parameters

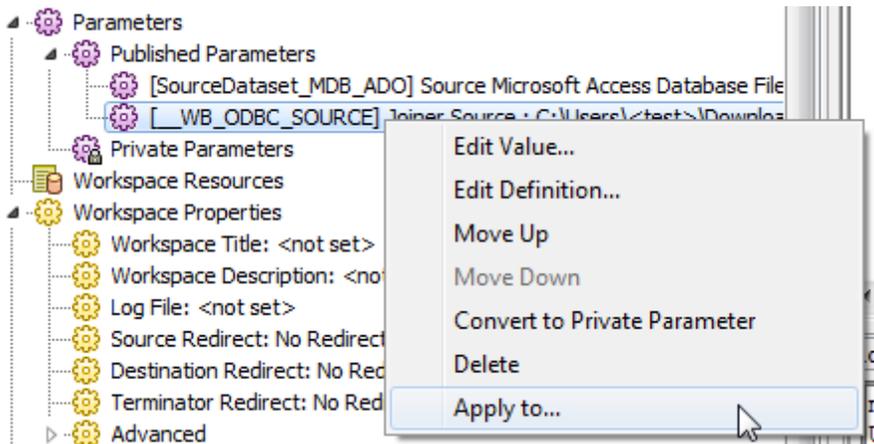
You can link any number of workspace parameters to use the same value as a Published or Private Parameter.

Example

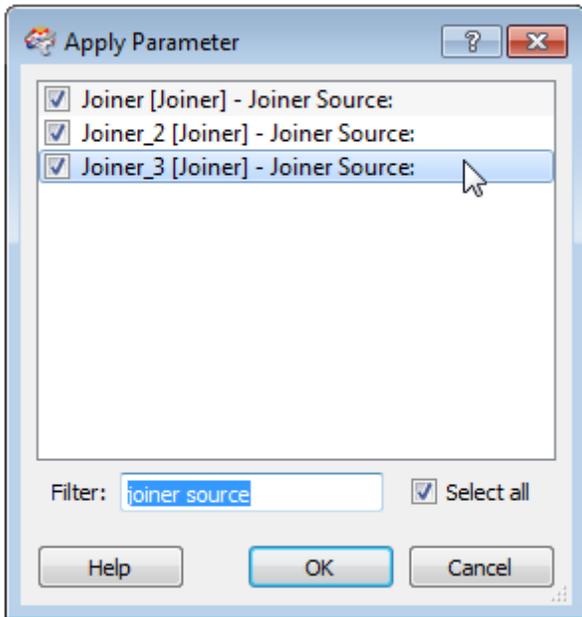
In this example, there are three Joiner transformers, which all join to the same database. Create a parameter from one of the Joiner Source parameters in the Navigator:



Click the Parameter in the Navigator, and select Apply To



Filter the list and click each Joiner Source parameter. Then click OK.



On the first Joiner, the Joiner Source has already been published. When you run this workspace, you will be prompted for the location of the database used in the Joiner. Then the source for Joiner_2 and Joiner_3 are set to use the published parameter value defined in the first Joiner.

When the workspace is run, you are still only prompted for one Joiner source, but the same value will be used for the other Joiners. Regardless of how many other parameters are set to use the first Joiner, changing this one parameter affects all the parameters that depend on it.

Note: The names in the Apply Parameter list will be reflected in the properties of both Published and Private Parameters.

Removing Parameter Links

To remove any parameter from the group, right-click and select Unlink Parameter.

Running a Workspace with Published Parameters

When you run a workspace using Published Parameters, Workbench will pause the translation and display a dialog that prompts you for the Published Parameters (but not Private Parameters). Edit or accept the parameters before continuing with the translation.

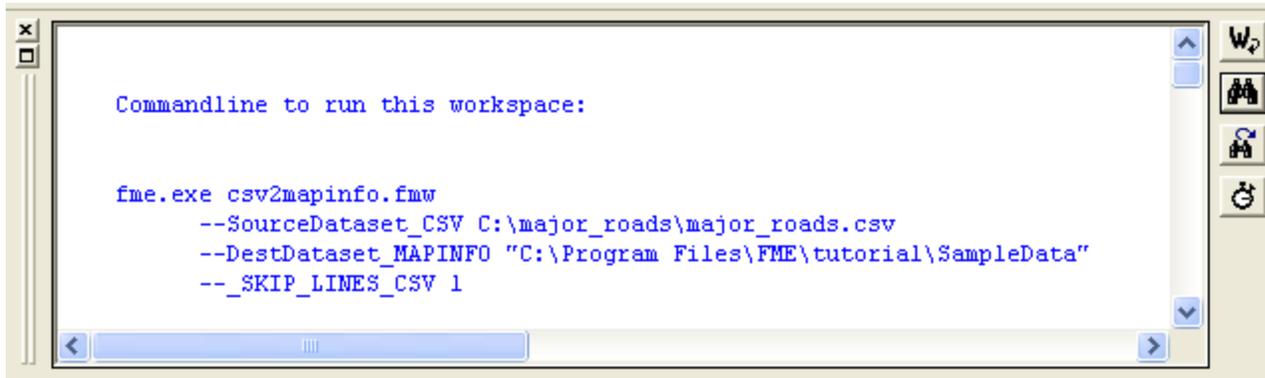
Note: If a parameter is published that can accept either character or numeric input, that parameter will only accept a numeric input when the workspace is run.

From Workbench

1. Select File > Prompt and Run Translation or click . It is important to choose *Prompt and Run Translation*. If you select File > Run Translation, or click the Rerun Translation button , Workbench will not prompt you for parameters. The default values will be used.
2. You will be prompted for values. (Remember, however, that if you defined Private Parameters, you will not be prompted for values.)
3. Click OK to continue with the translation.

From the Command Line

When you click the Run button to run the translation, the command line syntax is generated in the workspace log window. The syntax is visible in the beginning of the log file.

A screenshot of a log window with a light beige background. The window has a title bar with a close button (X) and a scroll bar on the right side. The text inside the window is as follows:

```
Commandline to run this workspace:  
  
fme.exe csv2mapinfo.fmw  
  --SourceDataset_CSV C:\major_roads\major_roads.csv  
  --DestDataset_MAPINFO "C:\Program Files\FME\tutorial\SampleData"  
  --_SKIP_LINES_CSV 1
```

At the bottom of the window, there is a horizontal scroll bar with a blue track and a white slider.

Note: The command-line syntax is displayed in the log window only. It will not appear in the log file. However, you can copy text from the log window and paste it into a text editor.

Viewing the Log

The translation log pane displays statistics and processing information that includes the following:

- FME version
- reader being used
- writer being used
- logging information
- warning messages
- command line, including published parameters

Information Messages

Information messages are displayed until the translation is complete. When the translation is complete, you can use the log window buttons to search for text, copy selected contents of the window directly to another application, toggle word wrap, or save the contents to a text file:



The log file is an important record of a translation and all the stages and processes within it. At first glance it might appear complex, even for a successful translation, but the information in here can be helpful when the output is not what was expected.

ERRORs

An ERROR in the log window signifies a problem that has caused FME to terminate processing. A typical example would be the inability to write the output dataset. It could just be that the user permissions are not correct, but since FME cannot create an output, there is no need to continue.

WARNings

A warning in the log window (displayed as WARN) signifies that there is a problem in the processing. The problem was not sufficient to cause FME to terminate the process, but you'll want to check that the problem has not adversely affected your output. For example, features with an incompatible geometry for the destination Feature Type will not cause the process to fail, since FME can filter out these features. However the output will not be what was expected.

INFORMation

An information message (displayed as INFORM) is a non-error incident which signifies a piece of information that may help you determine whether your translation has been correctly processed. Typical information would be the number of features processed or confirmation of a particular setting for that dataset.

STATisticS

Statistics messages (displayed as STATS) provide information on the number of features read from the source and written to the destination datasets.

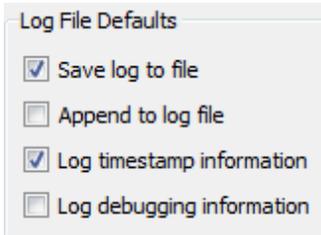
Note: The text in this area contains important information about the translation. If you ever get results that you did not expect in your output data, check the contents of the log.

Filtering Message Types

You can choose which information to display in the log through the Tools > Options > Runtime menu. The browse button  on the right of the log window also displays the Runtime options.

Log Timings

A useful option under Tools > FME Options > Runtime is the ability to turn log timestamp information on or off:



Log timestamps indicate the absolute date and time for each step of the translation process. They also show the time taken by FME to process the previous stage and the cumulative time taken to reach that point in the translation.

You should keep *Log timestamp information* enabled. The messages might add to the amount of content in the window but they can be very useful when trying to analyze a translation.

Performance

In general, FME performance can be slowed by the contents of the log window scrolling past. This is especially true when there are many features that all trigger the same warning. Keep the log window closed or resized to a minimum when running a translation. You can open the window again for inspection after the translation is complete.

Interpreting the Messages

See [Interpreting the Log](#) for information on interpreting the contents of the log.

Note: Because some FME functionality is implemented using a third-party library (for example, GeoDatabase writing uses ESRI ArcObjects), some information in the log window is reported directly from a product.

Saving Log Files by Default

You can choose to always save your log files:

1. Select Tools > Options.
2. Under Log File Defaults, check *Save log to file*. To add information to your log file instead of overwriting it each time you run the same translation, you can also check *Append to log file*.

Log files have the extension `.log` and by default are saved in the same directory as your workspace file.

Interpreting the Log

In the event of an error message or unexpected output, you can analyze the workspace to find which error has been introduced into the workflow definition. The log window is the most important place to look for information when a translation does not complete as you expected.

Check for Warnings

If you closed the log window during the translation, then you will not see any warning messages that may have passed by. The first thing to do is check for the following comment:

```
Translation was SUCCESSFUL with X warning(s)
```

And then (if $X > 0$), use the search option to look for the word WARN.

Sequence

Because FME processes data feature-by-feature instead of transformer-by-transformer, the log file does not show each transformer in sequence. That is only likely to happen when the transformer is group-based and does actually process all features simultaneously.

Output

If the workspace writers have been redirected to an output other than a destination dataset, then this is reported at the very bottom of the log. Watch for such messages. It's amazing the number of times I've forgotten to disable the output and thought there was a writer problem.

Timestamps

FME processing time is the amount of time that FME was actively processing.

The absolute start and end times often differ from FME processing time because non-FME processes, such as a database query, add to the absolute time taken without adding to the FME processing time. The log, therefore, can provide an indication of the efficiency of external processes (for example, database reading). A slow database read would imply database indexing may need to be improved.

Unexpected Input

Unexpected Input is reported through a message at the bottom of the log. However, this message does not mean that there is a definite problem. FME will report this as a warning even if you did not deliberately add unwanted source feature types to the workspace.

Filtering Rejected Input

Occasionally a transformer or writer will reject certain features because they do not match the functionality or parameters of that object. For example, if you send a mixture of point and line features into the AreaBuilder transformer, the log will report that the point features were discarded.

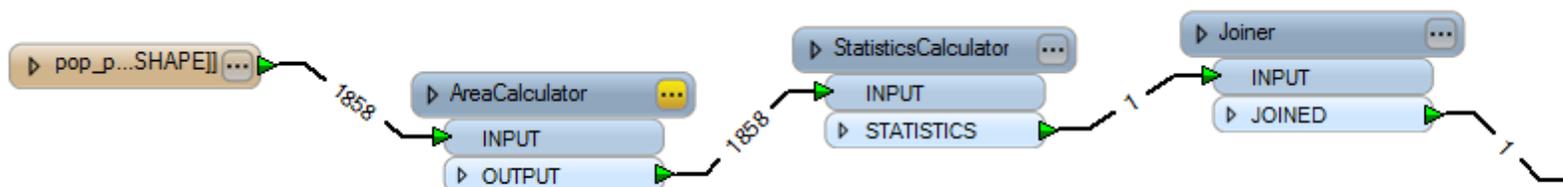
The log will be less verbose – and may even improve performance – if you can remove such features before they become a problem. For example, a GeometryFilter would be enough to filter out points before they even get to the AreaBuilder.

Feature Count Interpretation

When a log file shows a translation that ended in an error, or where the number of output features was not what you were expecting, then the Feature Count values shown on each connection can help diagnose where the error occurred.

However, different interpretations can be placed upon the same numbers.

In this example, when we run the workspace, it fails:



Where does the failure occur?

The feature counts are open to two interpretations:

- 1858 features entered the StatisticsCalculator, but only 1 emerged. Therefore the StatisticsCalculator must be at fault.
- The first feature caused the Joiner to fail. Therefore the Joiner is at fault.

It may not be entirely clear where the problem occurred, but at least you can narrow it down and be fairly certain that the reading of the data and the AreaCalculator are not the locations of the problem.

Useful Analytical Transformers

Some transformers that provide useful information in workspace analysis:

Visualizer Transformer

A Visualizer transformer takes features from the workflow and sends them to FME Universal Viewer, which lets you view partially processed features from the middle of the workflow.

Logger Transformer

A Logger transformer takes features from the workflow and writes information about them to the log window or log file. Limits can be placed on the number of features to be logged in full.

The Logger transformer has a parameter called Log Message: when a feature is logged, this message appears with it. Setting a unique message helps you locate logged features when using the log search function to find the message.

Performance Tuning a Workspace

The following topics contain useful information, especially if you want to cut down on processing time:

- How FME logs processing time
- How FME processes features and how it affects the translation process
- Processing time spent writing to temporary files
- How FME uses temporary disk space
- How FME logs reader information
- How FME logs database information

How FME logs processing time

Different situations affect the way FME logs processing time. For example, your translation log might say that FME took 0.1 seconds to process when it actually took 7.0 minutes. The timing part of the log shows the actual time (e.g. 10.40 am), the time FME has spent in processing, and the time FME spent processing the previous command.

The actual time taken may not match the time FME spends processing because of external calls such as database queries, which can't be logged by FME. See How FME logs database information.

Because FME works by pushing features through the workspace on an individual basis (not a group), it is not possible to give exact timings for every individual transformer. Therefore a lot of time doesn't get logged separately but is grouped together under the next function that does support timing.

How FME processes features and how it affects the translation process

By default, FME processes features on an individual basis; that is, each feature is processed separately within each transformer before being passed on to the next. This is because most transformers work on **Feature-Based Restructuring**, where the processing of one feature does not affect the next. Length measurement is one example of this: measuring the length of a line doesn't affect the length of the next.

However, some transformers work on **Group-Based Restructuring** where the processing of one feature DOES affect the next. Change Detection is one example of this: you can't detect whether a feature has changed unless you compare it against every other feature.

One consequence of group-based processing is that it affects available memory. If the process involves every feature in the dataset, all features must be held in memory while the process is carried out, as opposed to a feature-based process which can read a feature into memory and release it to move onto the next. The Sorter transformer is a good example of the group-processing type of function.

A second consequence is that some transformers will give different results depending on how features are being processed. For example, a VariableSetter/VariableRetriever combination may not give the expected results if you don't consider how features are being processed (the first feature through a VariableSetter will go straight through the VariableRetriever and retrieve the value it set itself - but if you insert a Sorter transformer between to hold up the features then the last feature will pass the VariableSetter before the first passes the VariableRetriever, thus retrieving a different value).

Note: The number of features written really means the number of features that were passed to the writer, which is not necessarily equal to the number of features that can be viewed in the FME Universal Viewer or in the destination format. What the writer does with the features (whether it splits them, or even rejects them, etc.) is dependent on the individual format. There is no reliable way of relaying that information back to the FME core, which outputs the statistics information.

Writing to temporary files

If you notice that FME processing time is spent writing to temporary FFS (FME Feature Store) files, the issue isn't one of actual processing time, but how that time is reported in the log file. FME doesn't log the time taken by each function – often it just reports 0.0. This may be because the

time is less than 0.1 seconds (the precision allowed for), or because the individual function cannot be accurately measured. Therefore, the time used accumulates without being reported.

Therefore, when the "Storing features..." process logs the time taken, it reports all accumulated time, giving the impression that much of the FME process is accounted for by writing FFS files.

In the following example, it doesn't mean that 0.3 seconds were spent writing to FFS, just that 0.3 seconds have been used since the last occasion this number was reported:

```
2010-08-12 10:25:01| 1.1| 0.1|INFORM|Opened DBF File <xxxx> for output
```

```
2010-08-12 10:25:01| 1.2| 0.0|INFORM|Opened Shape File <xxxx> for output
```

```
2010-08-12 10:25:01| 1.2| 0.0|INFORM|Opened DBF File <xxxx> for output
```

```
2010-08-12 10:25:01| 1.5| 0.3|STATS |Storing features to FME feature store file
```

How FME uses temporary disk space

One of the first items of importance in the log file is the temporary directory. You'll see this reported as something like this (timings have been removed for clarity):

```
INFORM|FME Configuration: Temporary directory is 'C:\DOCUME~1\xxxx\LOCALS~1\Temp'
```

You'll also see a line that shows the amount of disk space available in that directory:

```
INFORM|System Status: 37700MB of disk space available in the FME temporary directory
```

When FME runs a large, multi-dataset translation, it often requires a lot of temporary disk space. This is particularly true when running a Dataset Fanout, because there is no guarantee that the features will arrive at the fanout in a single dataset group. Therefore, FME has to write out all of the datasets to temporary storage, and then fan them out afterwards. So the amount of available disk space is important, but on a performance issue you might be more concerned about the speed of all this disk activity.

See Setting the Temporary Directory for tips.

How FME logs reader information

FME works by pushing features through the workspace on an individual basis, and it does this as soon as each feature is read from the source data. Therefore it's sometimes difficult to calculate the time spent reading the data because workspace processing time will be grouped in with it.

For example, 'Emptying factory pipeline' below marks the point at which FME has finished reading data.

```
2006-02-03 11:37:47| 342.7| 0.5|INFORM|Emptying factory pipeline
```

Here it took 342.7 seconds (about 6 minutes) to read the source data. But, this includes time spent processing the features within the workspace. When all the transformers were removed from this workspace, the log read:

```
2006-02-03 14:44:43| 66.5| 0.3|INFORM|Emptying factory pipeline
```

This tells us that 80% of the time was spent processing the data and only 20% reading it. In this case, you should check your workspace to make sure it is as efficient as possible and that there are no unnecessary transformers.

Tip: If you are concerned about the reading performance of a workspace, disable the readers from the transformers in Workbench and run the translation again. Then compare the log files. It may be that a lot of the time you assume was spent reading data is actually used by the workspace transformers and this will show where to concentrate your performance efforts.

How FME logs database information

Databases are an important component of many datasets and the log file will help us determine both how good our database performance is, plus how well FME is interacting with the database.

The following example relates to a prefetch query carried out on an Oracle database:

2004-05-14 17:18:52| 476.1| 0.0|INFORM|Started SQL cache prefetch

2004-05-14 17:25:10| 476.2| 0.1|INFORM|Finished SQL cache

Although the time between when you issue the SQL prefetch and until it is complete is roughly 7 minutes, FME uses only 0.1 seconds of CPU time. The remaining time was spent by Oracle retrieving the data using the query it was given. To reduce that time, you would need to look at how the Oracle database is structured and how the query is written. Perhaps the field searched on isn't indexed? Maybe the query supplied isn't as efficient as it could be?

Check the log carefully to find out how much database-related time is spent outside of FME and see if you need to improve your database efficiency.

Creating a Custom Format

A Custom Format is an FME Workspace that can be used to preprocess data as if it were a data format on its own. Once created, it will show up in the input formats list. From then on, it can be used in any FME component as a separate format, including application extensions.

A Custom Format can only be an input format. When you create a Custom Format, you can, in effect, define your readers, your pipeline, and the format's schema, and you can use it like a standard Source Format. This is useful if you consistently use the same information in a translation, and when you consistently perform the same processing on datasets. After you create and save the format, it will be selectable from the Formats Gallery.

There are two ways to start:

- Export as a Custom Format.
- Use the Custom Format wizard.

For more information, see the [fmepedia](#) page here.

Exporting as a Custom Format

Open an existing workspace and select File > Export as Custom Format. You will be prompted to enter a name and description. A new Workbench window will open, and the custom format information will appear in the title bar.

Using the Custom Format Wizard

To initiate the Custom Format wizard, choose Browse Formats from the Tools menu, and click the New button in the "Custom Formats" area at the bottom of the Gallery window.

The Custom Format wizard will guide you through the following steps:

1. Select the Source Format

Select the format of your source data. You can also click the Browse button to choose from the Reader and Writer Gallery.

2. Locate Source Data

Type the location of your input data. You can also use the buttons to browse for files or add multiple datasets. If the source format has specific default settings that you want to change, you can edit them here.

3. Select Exposed Parameters

The parameters listed here will depend on your source data, and the ones you select will determine which parameters will be shown when you use the format in a translation.

4. Name the Custom Format

Enter a short name and a description for the new format. When the format is saved to the Reader and Writer Gallery, the short name appears in the Short Name column, and the description appears in the Name column.

5. Create the Custom Format

After you click Finish, the Workbench canvas will open and the feature types will appear. Like a regular workspace, you can do the following:

- include transformers
- edit feature types
- merge feature types
- fanout feature types
- add additional readers
- add destination feature types

The only visible difference between a Custom Format and a Workspace is that there is no destination dataset. You can, however, add destination feature type definitions, which become the source schema for the custom format.

6. Save the Format

When you are finished, select File > Save. The custom format will be included in the Reader and Writer Gallery.

FME will automatically assign a default .fds file extension. The new format will be stored in My Documents\My FME Datasources.

Editing a Custom Format

From the Reader and Writer Gallery, choose the format, then click the Custom Formats Edit button.

Deleting a Custom Format

From the Reader and Writer Gallery, choose the format, then click the Custom Formats Delete button.

Maintaining Custom Formats

When you update a version of FME, in most cases, this will not affect how you access custom formats. However, ArcGIS Data Interoperability users may need to perform an additional step.

Procedure for ArcGIS Users

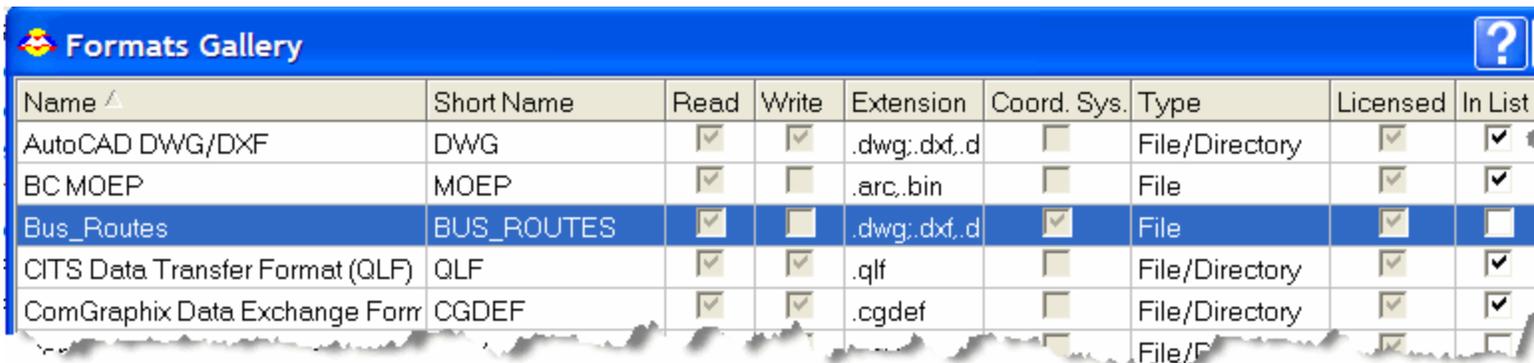
When upgrading to a newer version of ArcGIS (for example, from 9.1 to 9.2), ArcGIS users must use a specific process in order to access custom formats:

- Do not copy the format files (.fds) to the location: C:\<My Documents>\FME\Formats that ArcGIS 9.2 uses.
- You must use the Import option in the Data Interoperability Reader and Writer Gallery and import your formats. Simply copying them over will not be effective since the formats will display in the Reader and Writer Gallery but the "Edit" button will be disabled.

Using a Custom Format in a Translation

To use the Custom Format as a Source Format from the Workspace Wizard or Workspace dialog:

1. Click the Browse button to display the Formats Gallery.
2. Select the Custom Format as a source format.



Name ^	Short Name	Read	Write	Extension	Coord. Sys.	Type	Licensed	In List
AutoCAD DWG/DXF	DWG	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	.dwg;.dxf;.d	<input type="checkbox"/>	File/Directory	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
BC MOEP	MOEP	<input checked="" type="checkbox"/>	<input type="checkbox"/>	.arc;.bin	<input type="checkbox"/>	File	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Bus_Routes	BUS_ROUTES	<input checked="" type="checkbox"/>	<input type="checkbox"/>	.dwg;.dxf;.d	<input checked="" type="checkbox"/>	File	<input checked="" type="checkbox"/>	<input type="checkbox"/>
CITS Data Transfer Format (QLF)	QLF	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	.qlf	<input type="checkbox"/>	File/Directory	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ComGraphix Data Exchange Form	CGDEF	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	.cgdef	<input type="checkbox"/>	File/Directory	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
						File/Directory	<input checked="" type="checkbox"/>	<input type="checkbox"/>

3. In the source dataset field, select the file from which you want the Custom Format to read.
4. Click the **Settings** button.

Here's an example of some [selected parameters](#) from the **Select Parameters to Show** dialog in the Create a Custom Format wizard.

When you use the format as a source format, the selected parameters will appear in a [custom settings box](#). They will also appear as [parameters in the Navigation pane](#).

Select parameters to show

Show	Parameter
<input checked="" type="checkbox"/>	Input Units:
<input checked="" type="checkbox"/>	Expand Cells:
<input type="checkbox"/>	Expand Unnamed Cells:
<input checked="" type="checkbox"/>	Preserve Cell Insert Points:
<input checked="" type="checkbox"/>	Output Tags As Text:
<input checked="" type="checkbox"/>	Drop Complex Chains:
<input checked="" type="checkbox"/>	Propagate Chain Member Linkages:
<input checked="" type="checkbox"/>	Split multi text:
<input checked="" type="checkbox"/>	Preserve Unnamed Cells:
<input type="checkbox"/>	UOR to FME Feature Coordinate Units scale factor:
<input type="checkbox"/>	UCRS REF SURF UNITS:

IGDS to FDS Translation [X]

Input Units:

Expand Cells:

Preserve Cell Insert Points:

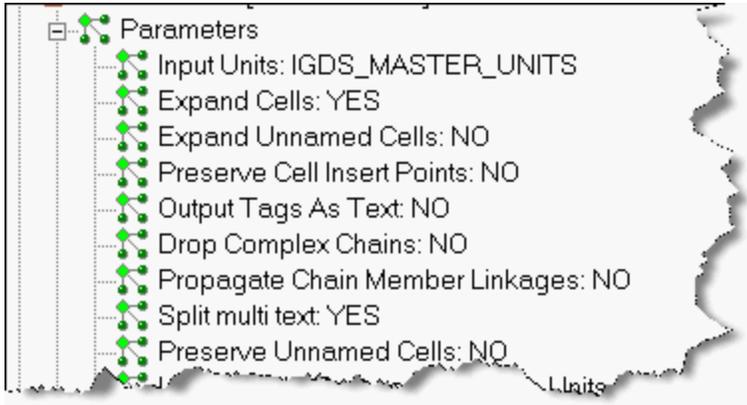
Output Tags As Text:

Drop Complex Chains:

Propagate Chain Member Linkages:

Split multi text:

Preserve Unnamed Cells:



Installing a Custom Format

Double-click on any .fds file and Workbench will automatically install the format.

Sharing a Custom Format

Custom formats can be read from user-defined directories (including network shares).

This option is especially useful for groups who work together. For instance, if an entire workgroup uses just a few custom formats, keeping the definitions in one place means that everyone doesn't have to have a copy. More importantly, whenever any of the definitions are updated, the entire group automatically has access to the new version.

See Shared Resource Directories.

About Custom Transformers

FME Workbench allows you to replace a sequence of existing transformers with a single "custom" transformer. Custom transformers execute the same data flow as the original transformers, and can be edited just like any other transformer. You can use Custom Transformers in two ways:

- create them to store and use locally on your computer
- export them to edit independently or share with other users

How can they be used?

Custom transformers can be used:

- to streamline a large workspace
- to make various data flows within a workspace more understandable and easier to document
- as a replacement for the same chain of transformers multiple times within the same workspace

You can also export a Custom Transformer to share with other users. After they are exported, custom transformers can be:

- edited independently of any workspace
- shared with other users (via e-mail or shared directories)
- embedded or linked in any number of workspaces
- stored as a separate .fmx file

Creating a Custom Transformer

The easiest way to create a custom transformer is by opening an existing workspace and selecting existing transformers:

Creating a Custom Transformer

You can also build a new Custom Transformer from a blank canvas:

Building a Custom Transformer

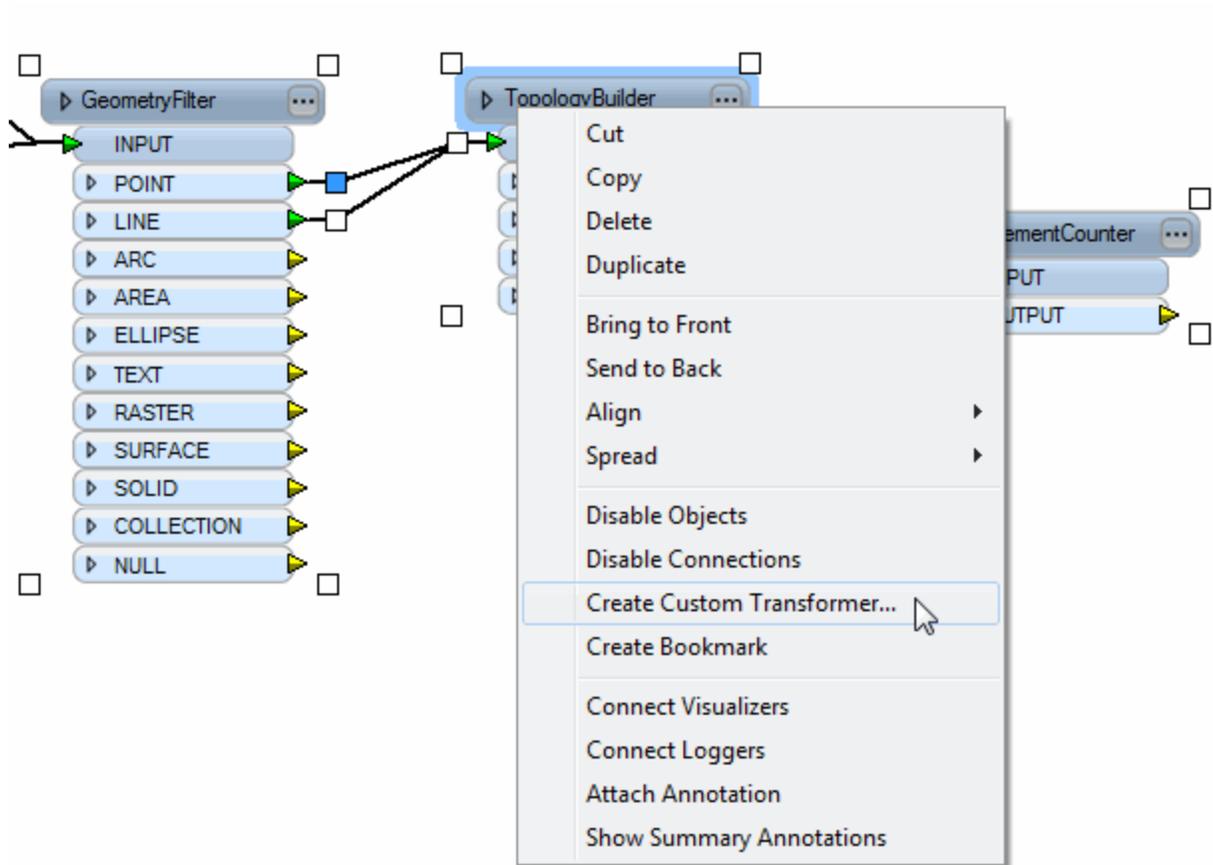
Where they are stored

By default, custom transformers are stored in My Documents\FME\Transformers.

Creating a Custom Transformer

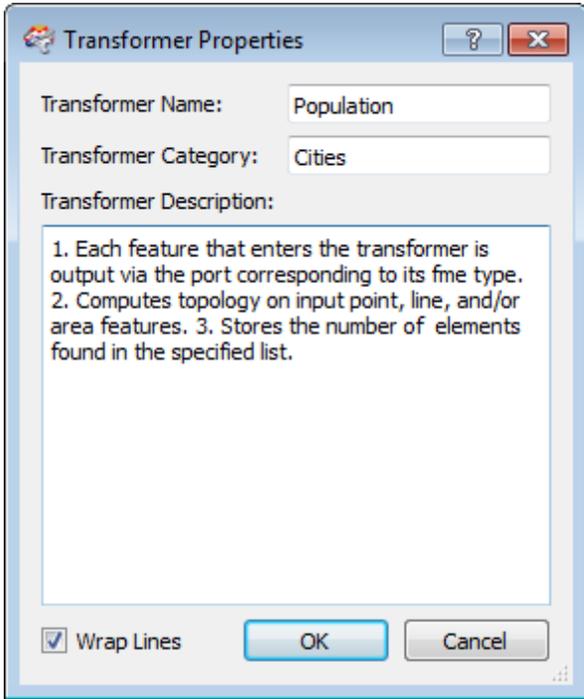
The easiest way to create a custom transformer is from an existing workspace, and by choosing transformers that you are often reusing, or transformers that take up a lot of space on the canvas.

1. Open a workspace.
2. Select a group of transformers: click and hold the left mouse button, and drag a box around the transformers. Release the mouse button.
3. Right-click and select Create Custom Transformer from the command menu.

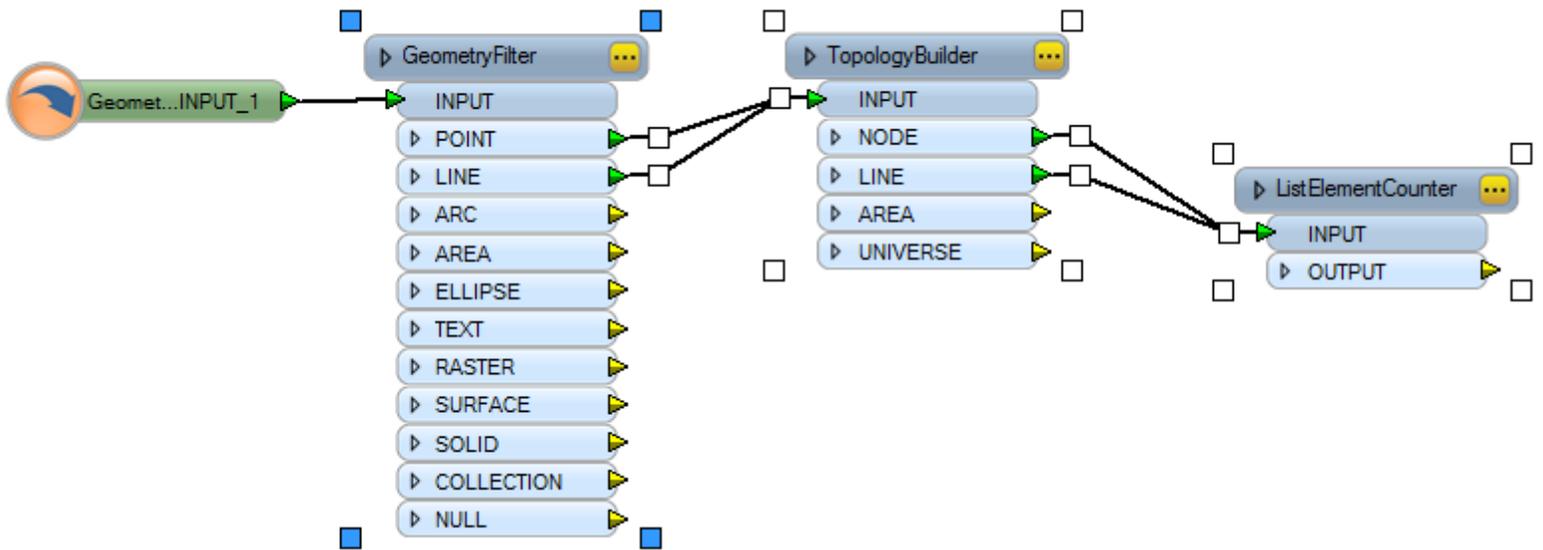


4. Enter a transformer name, category and a description (optional) in the dialog that appears.

Tip: You can enter your own category name in the Transformer Category field. The category will be added to the Transformer Gallery.

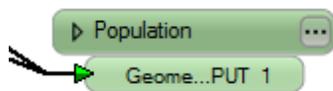


5. Click OK. In this example, Workbench opens a new tab named *Population* and the original group of transformers is shown on the canvas:



The input and output (if included) arrows are for reference only, to reflect the input/output to the custom transformer in the main workspace.

6. The original group of transformers is replaced with the new custom transformer in the main workspace:



Note: You cannot include any reader or writer feature types in a custom transformer.

The Navigator includes details about custom transformers: it lists the transformers (and their attributes) that are part of the custom transformer, and its own parameters are included under a Transformer Properties icon:



In the main workspace, the custom transformers are listed in the Navigator (much like any other transformer), except they appear as green icons.

Other Ways to Create Custom Transformers

- Right-click in your main workspace and choosing Insert Custom Transformer. You can build the transformer workflow separately, and then manually connect it to both the input and output transformer arrows, and in the main workspace.
- Create a bookmark and turn its contents into a custom transformer.

What can you do with a custom transformer?

After you have created a custom transformer, you can:

- edit its definition in separate tabbed window
- duplicate it (copy/paste it) within the same workspace
- select it from the Transformer Gallery, and include it any number of times
- copy/paste it into a new workspace (its elements are also copied)
- share it with other users
- export it as a separate .fmx file (by default, it will be stored in My Documents\FME\Transformers)

Building Custom Transformers

The easiest way to create a custom transformer is from an existing workspace, but you can also build one starting from a blank canvas:

1. Choose Insert > Custom Transformer from the main menu, or right-click anywhere in the blank canvas area and choose Insert Custom Transformer.
2. Enter a name, category and a description (optional) in the dialog that appears.

Tip: If you create your own category, you can easily reference any transformers that you create. The category will be visible in the Transformer Gallery.

Using Custom Transformers

Especially considering the increasing number of very large transformations, using custom transformers can simplify many workspaces. A large workspace that is decomposed into smaller pieces will be easier to manage.

Including a custom transformer in your workspace

Choose the custom transformer from the Transformer Gallery.

Opening a custom transformer

If you exported the custom transformer you can also open it just like you would a regular workspace.

1. Choose File > Open and browse to your FME > Transformers folder.

Editing custom transformer definitions

You can edit an embedded custom transformer (one that you previously exported) by clicking its tab at the top of the workspace window and editing the contents, or right-clicking on the custom transformer and choosing Edit from the command menu.

The changes are automatically applied within the main workspace, wherever the custom transformer is used.

Tip: If you close a custom transformer tab, you can reopen it by right-clicking on the custom transformer in the main workspace and choosing Edit from the command menu.

NOTE: If you delete a custom transformer from the main workspace, you will have to recreate it. As long as its original tab is still open in the workspace, you can use its contents to create a new custom transformer. Choose Insert Custom Transformer from the main workspace, then copy and paste the contents of the original tab to the new tab. You can then close the original tab.

What else can you do with custom transformers?

Publishing Parameters in Custom Transformers

Exporting a Custom Transformer

Sharing a Custom Transformer

Installing a Custom Transformer

Looping a Custom Transformer

Duplicating Custom Transformers

One of the key uses for a custom transformer is to repeat a pattern of transformers throughout a workspace. Reusing a custom transformer can be done the same way as with a normal FME transformer, in one of these ways:

- Locate it within the transformer gallery and place a new instance.
- Select the custom transformer, right-click to display the command menu, and choose Duplicate.
- Copy and paste within the same workspace. If two compound transformers have the same name, Workbench will add a `_2` suffix to the copy.
- Copy and paste from another workspace. The definition is embedded into the current workspace.

You cannot use a custom transformers within itself, but you can use one custom transformer inside a different one.

Working with Copies of Custom Transformers

When a custom transformer is to be used in multiple locations (for example, after you duplicate it), it is important to understand how each instance of that transformer references other parts of a workspace as input. See [Working with Multiple Copies of Custom Transformers](#).

Working with Multiple Copies of Custom Transformers

If a custom transformer will be used in multiple locations (for example, after you duplicate it), it is important to understand how each instance of that transformer references other parts of a workspace as input.

For example, if you have a single instance of a custom transformer that references an attribute name, the first instance of a custom transformer will see the attribute name as being available, because FME can guarantee that the attribute is available within that stream. However, when you reuse the transformer, FME cannot guarantee that the transformer will be connected to a stream containing the same set of attributes.

To avoid the potential problem of referencing a non-existent attribute, when you duplicate and reuse a custom transformer, FME automatically publishes any parameters that are referenced as input. This allows you to individually select different attributes for each instance of the transformer, and keep the custom transformer valid and usable.

Note that custom transformers created before FME 2007 will be automatically updated to include this feature when you open them in FME 2007.

There are two potential workflow scenarios when you reuse custom transformers:

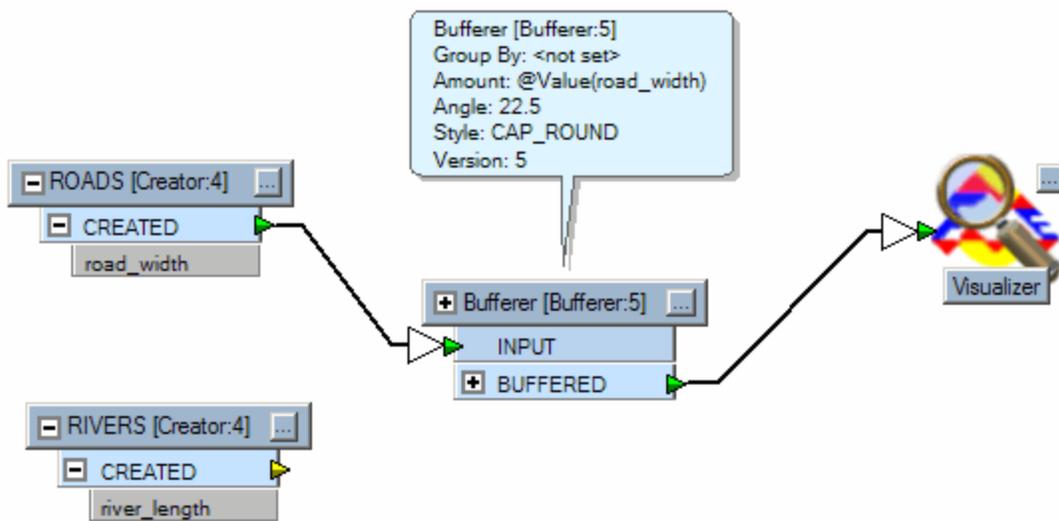
- If you do want to change input attributes, then using published parameters is the best solution. See Reusing custom transformers: changing input attributes, below.
- If you know that you will be using the same attributes, you can introduce an AttributeExposer to replace the published parameter(s). See Other ways of identifying referenced attributes.

Reusing custom transformers: changing input attributes

The simple example below shows what happens when you create a custom transformer that needs to reference different attributes for each instance of the transformer.

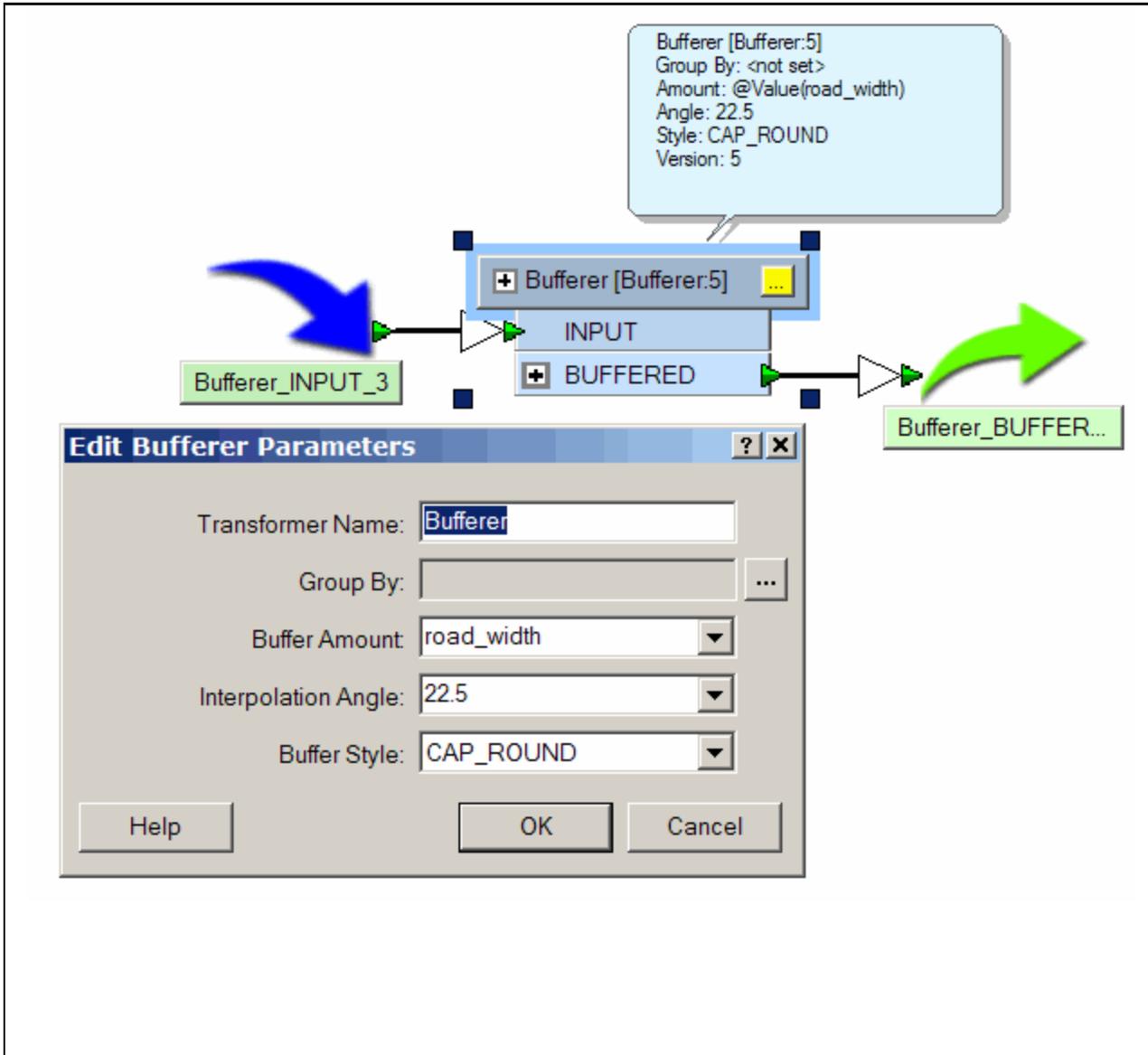
This example shows a set of transformers.

As the Summary Annotation shows, the Buffer Amount field in the Bufferer is linked to the road_width attribute from the Creator.

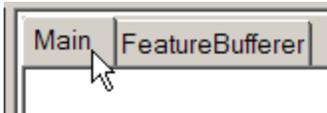


Create a custom transformer from the Bufferer transformer.

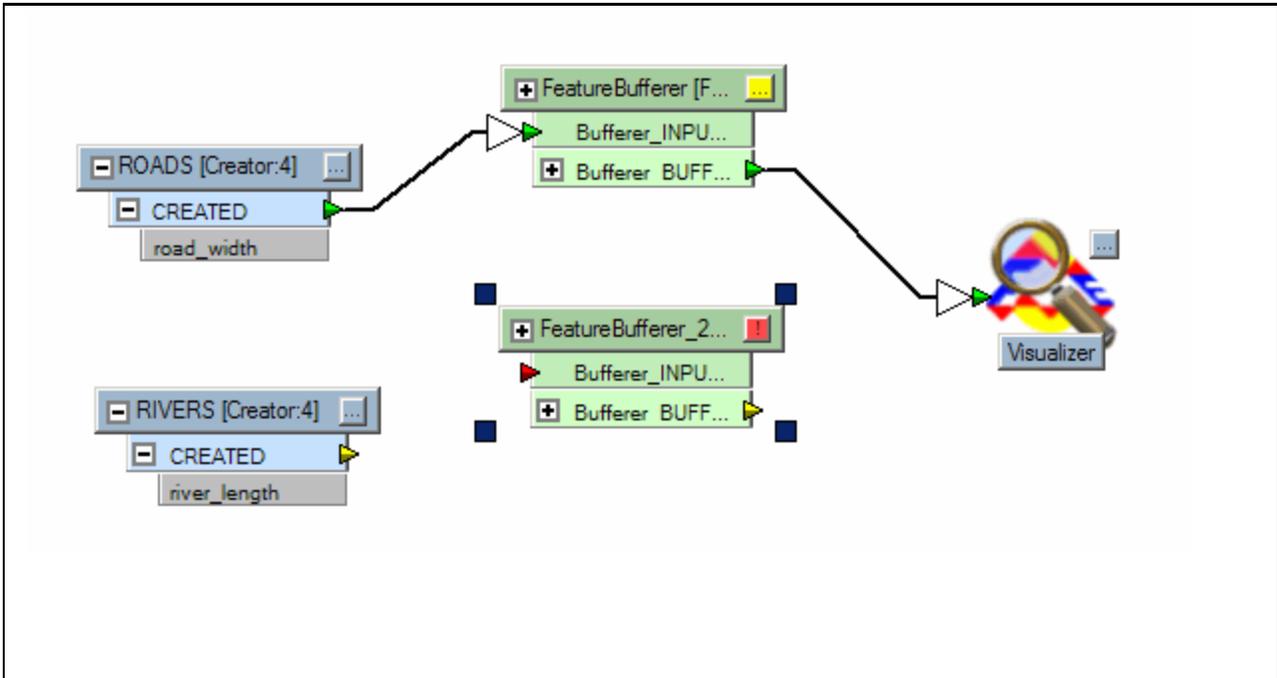
When the custom transformer appears, click the yellow properties button. The Bufferer transformer is still referencing the original attribute from the Creator transformer.



Click the Main tab in the workspace window.



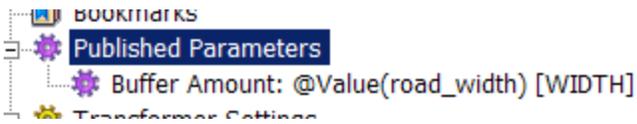
Select the custom transformer and copy/paste it (or choose Duplicate from the command menu). The properties button on the duplicated transformer is red because the transformer is looking for the original road_width attribute.



Click the tab that contains the new custom transformer.

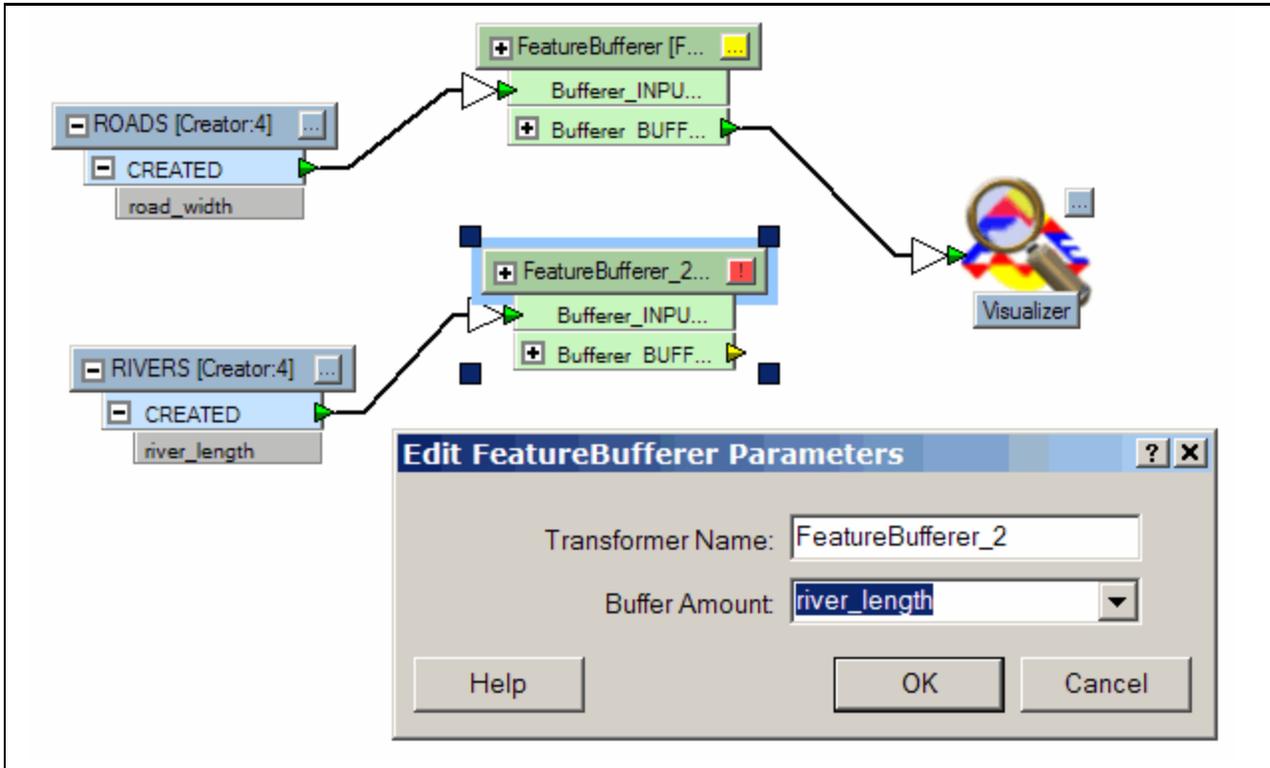


FME has automatically published the Buffer Amount parameter, and the missing attribute is listed under Published Parameters in the Navigator pane.

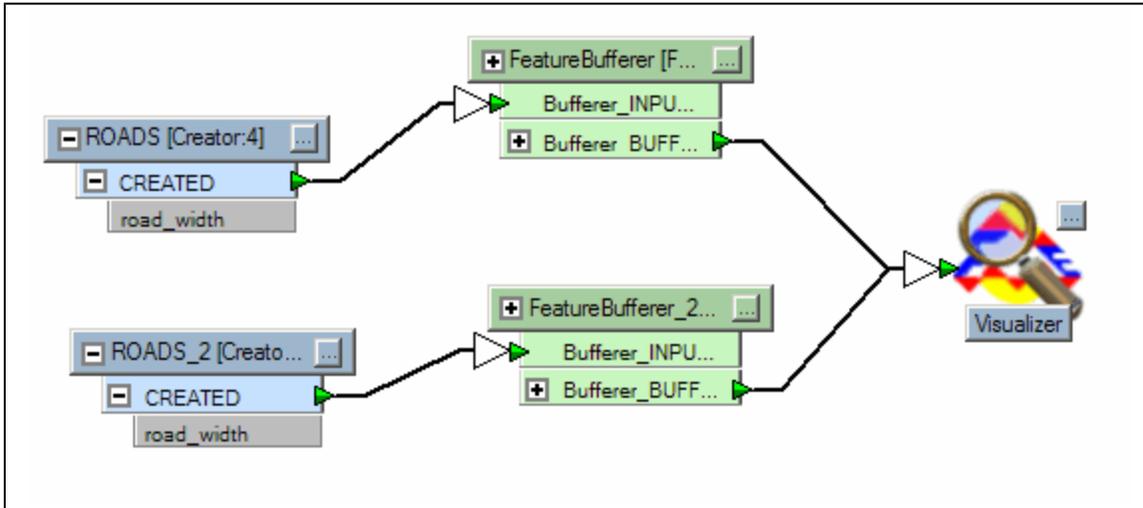


If you delete this published parameter, the properties button of the custom transformer will turn red, since it will still be trying to reference road_width.

Publishing the parameters means that if you now connect the River features to the reused custom transformer, you have the option to choose river_length as the attribute, instead of road_width.



This alternate example shows the situation where both instances do use the same attribute.

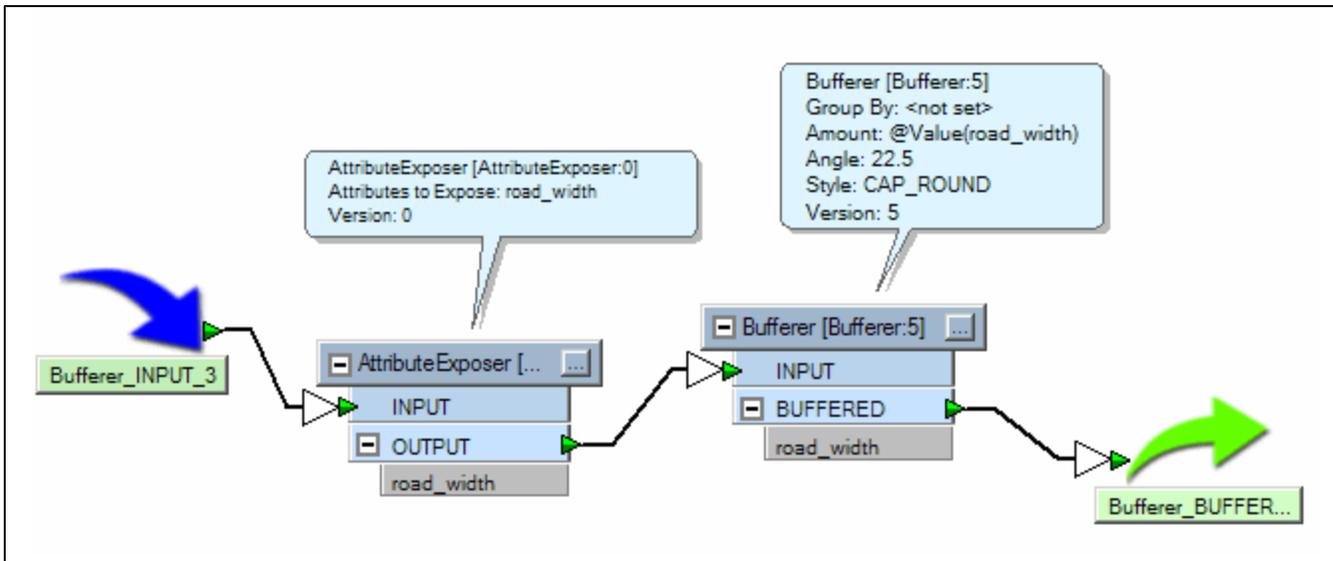


Other ways of identifying referenced attributes

Inserting an AttributeExposer into the custom transformer can identify any hidden referenced attributes. In this scenario, there is no need to select a different attribute when you insert a custom transformer into a workspace.

In this example, we deleted the published parameter that FME created, and replaced it by inserting an AttributeExposer. By manually exposing the road_width attribute, we have made the attribute available within every instance of this custom transformer.

Note, however, that you must be absolutely certain that any workflow that incorporates this custom transformer includes the `road_width` attribute, since it is now "hard-coded" into the transformer.



Looping Custom Transformers

Looping a custom transformer is useful for users who want to leverage existing transformers (Testers, Filters, etc) to determine end conditions.

For example, instead of rerunning a workspace many times and resetting transformer properties, you can set a transformer to "loop" until the conditions are met. This means that any features that enter the transformer will be sent back to its given input if the conditions don't yet match.

These transformers also work well if you want create values for the `fme_color` (or `fme_fill_color`) attribute, which would spread color evenly from dark to light independently on the number of input features.

To set up looping with group-based (or *blocking*) transformers, see [Looping with a Blocking Transformer](#).

Steps

1. Create a Custom Transformer in an existing workspace. You might want to include "Loop" in the name.
2. Include the transformers that will make up the conditions of the loop.
3. Right-click in the workspace and choose Insert Transformer Loop, or select the menu Insert > Transformer Loop.
3. Choose the INPUT to loop to (if there's more than one). The following icon appears on the canvas:



4. Place the transformer in the main workspace and run the workspace. When the looping transformer reaches the preset parameters in the custom transformer properties, the translation will complete.

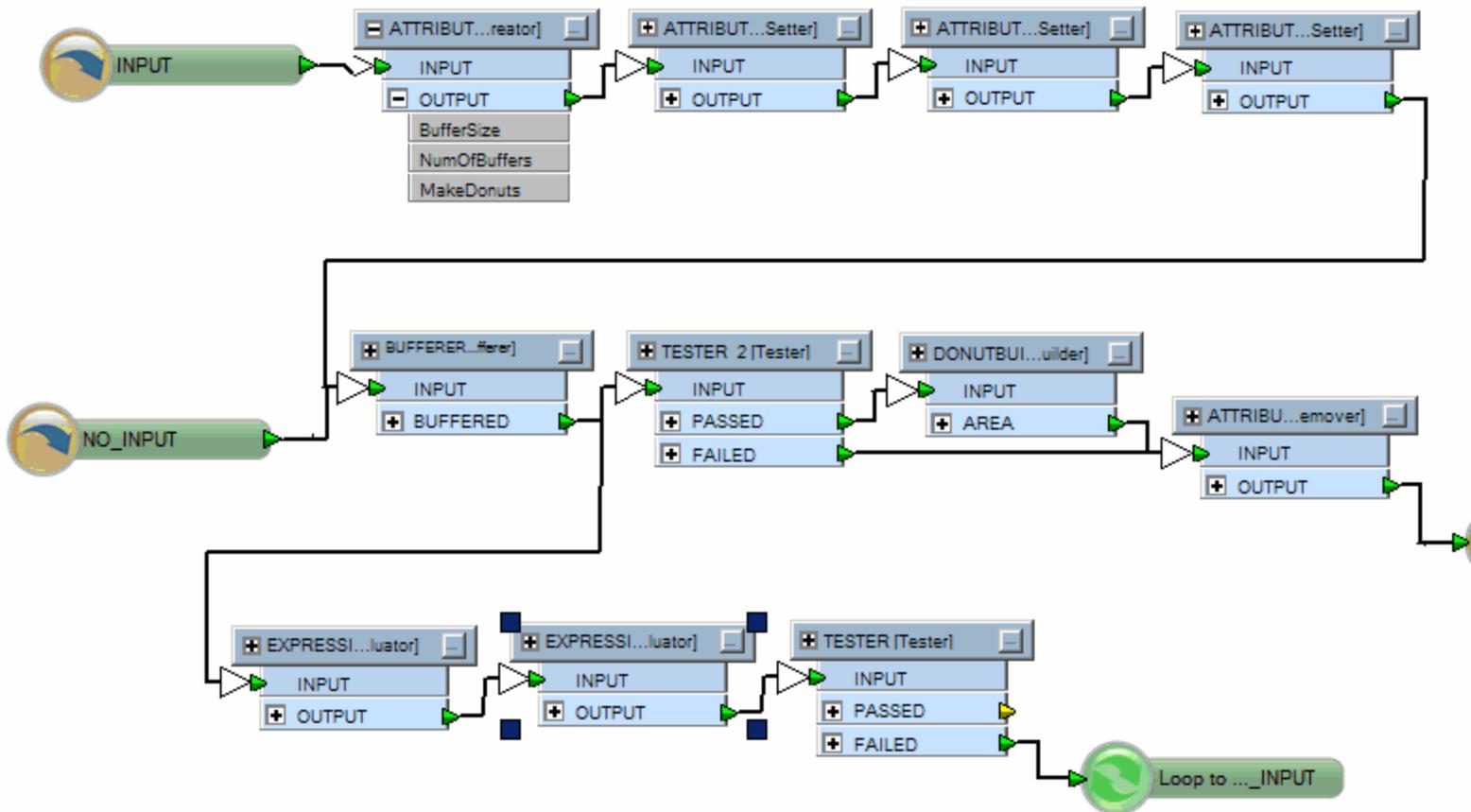
Usage Notes

- Looping transformer inputs are based on transformers specific to the workspace you are in.
- Use a Visualizer to check your results.
- Publish parameters if you want to change the output slightly each time you run the workspace. Make sure you press Ctrl+R to run the translation so that you will be prompted for the parameters.

Disabling Published Input

When you want to set a number of iterations as a transformer's parameter, you might want to connect the Loop to another input port. In this case, this input port should not receive any external input features. To do this, right-click on the applicable port in the Navigator pane, and uncheck the Publish option.

In this example, the NOINPUT input port should not receive any features.



Looping with a Blocking Transformer

FME can now create processing loops around blocking (or group-based) transformers that would usually block the flow of features. This feature is similar to looping functionality but it operates in a very different manner.

Background

In a standard workspace, the feature processing loop processes one feature at a time. FME's usual workflow is to take one feature from the reader, push it through transformers, and then write the result to a writer before moving on to the next input feature. Transformers can divert features along different paths, but FME will push the single feature as far down the workspace as possible before moving on to the next feature.

Workbench provides a way to develop a custom transformer that can bypass FME's natural process of transformation: the transformer loop enables FME to take a feature from one point of processing within the custom transformer and insert it into another point. This is called Looping. Until now, looping was limited to feature-based transformers only.

Some transformers can *block*: that is, they might hold on to some features, very often until FME finishes processing. (This will happen when a transformer's "Group By" functionality is used, for instance, because groups cannot complete until every feature has been considered.) Transformers like this are also often called "Group-Based" transformers. When these transformers hold onto a feature, FME returns to the reader to get the next feature for processing. The blocking transformer cannot generate any output until all of its input has been accumulated. Logically, this means that a blocking transformer cannot be included in a transformer loop because the loop occurs at a time when the transformer is not yet prepared to take on new features.

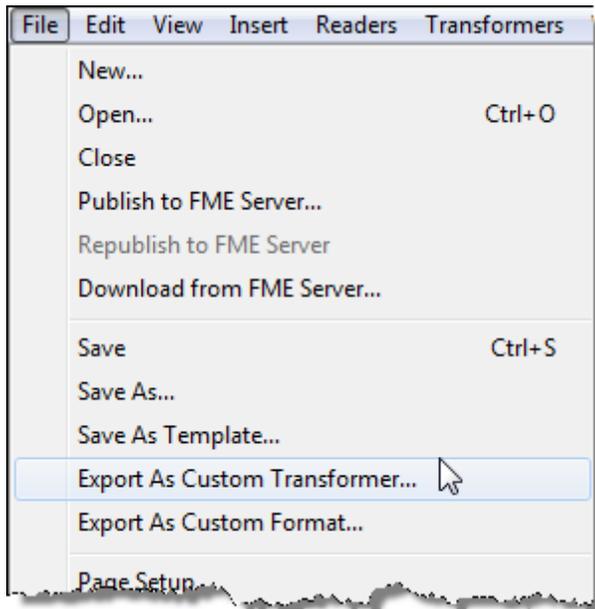
The Enable Blocked Looping feature overcomes this limitation.

Requirements

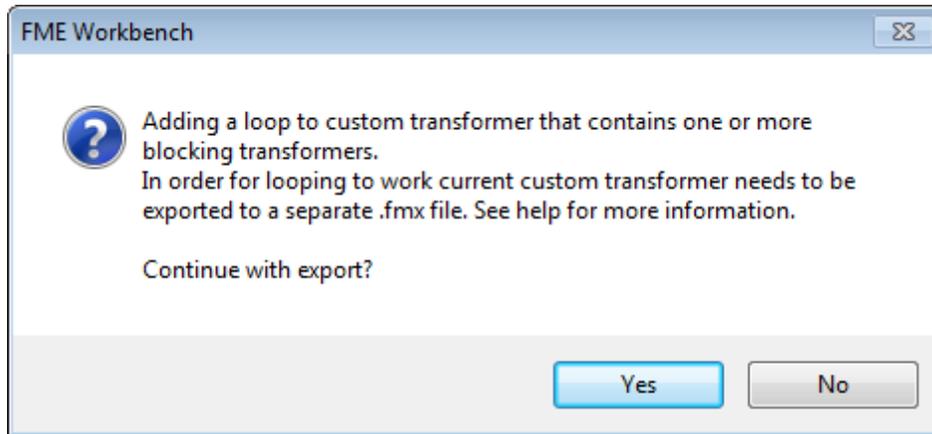
There is only one requirement when using blocking transformers in loops: custom transformers must be external (saved and stored as an `.fmx` file) and always LINKED (not embedded).

Steps

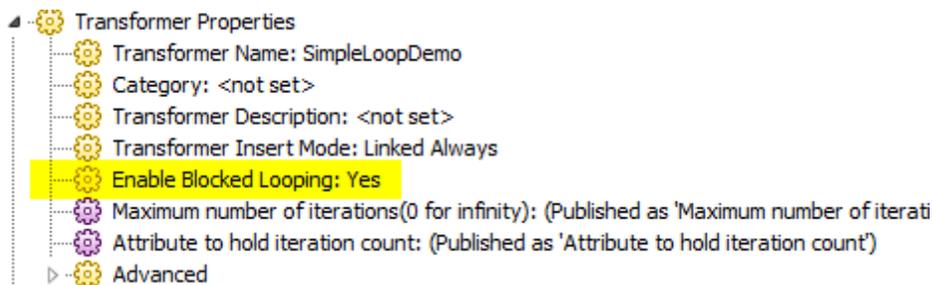
1. Create a custom transformer. If it is an embedded custom transformer, export it as a linked custom transformer.



FME will also prompt you to export the custom transformer: when you place a blocking transformer in a custom transformer with a loop, or add a LOOP port to a custom transformer that contains a blocking transformer, FME displays the following message:

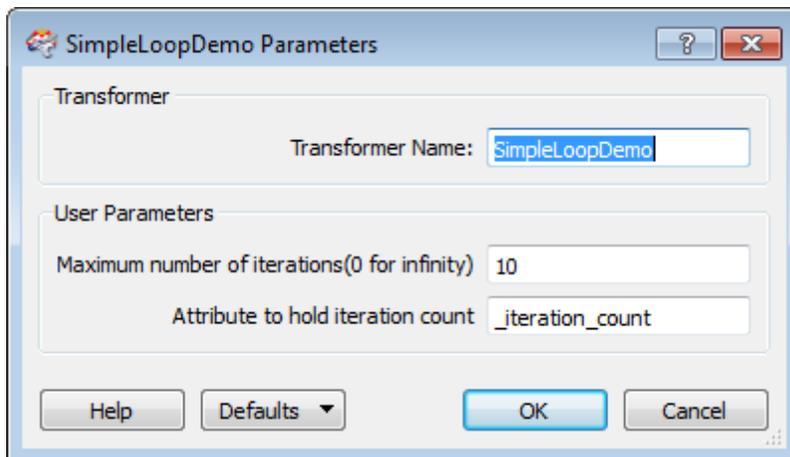


2. Open the linked custom transformer (.fmx file) in Workbench. In the Navigator, under Transformer Properties, set "Enable Blocked Looping" parameter to Yes and save the changes.



Only one LOOP is allowed when "Enable Blocked Looping" is set to Yes.

When "Blocked Looping" is enabled in a linked custom transformer, two additional parameters are exposed on the transformer: *Maximum number of iterations* and *Attribute to hold iteration count*:



These two parameters control how many times the loop is to be executed and help prevent infinite looping.

3. Insert this linked custom transformer into a workspace that requires looping. When you add the transformer to a workspace, it will appear as "Linked". (If it does not, right-click on the inserted transformer and choose *Link*.)

- This inserted linked custom transformer will have one extra output port called INCOMPLETE. You cannot rename this port. It is created automatically to route the features that satisfy the looping condition but cannot be looped back because the loop iteration count has exceeded the value of "Maximum number of iterations".

Note: Any of the INPUT/OUTPUT ports in the linked custom transformer cannot use the name "INCOMPLETE".

Embedding or Linking a Custom Transformer

When you create a custom transformer, its default properties are to embed it into any workspace.

When a transformer is embedded in a workspace, all of its properties are also copied to the workspace. However, you can also choose to link to the transformer (which is useful if you frequently update the transformer and you use it in many different workspaces).

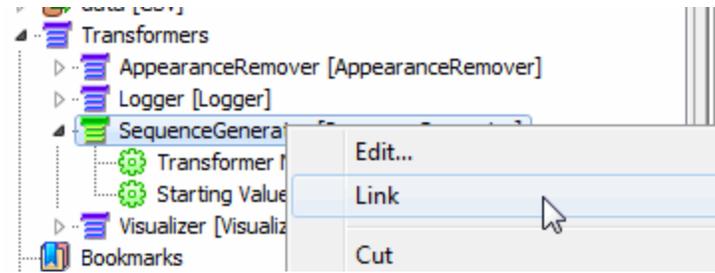
Remember that when you link to a transformer, and you move or share the workspace to which it is linked, you must also move or send the transformer separately. If you open a workspace containing a link to a custom transformer that is not installed on your system, you will see an error. This is not always easy to spot, since custom transformers can be embedded inside one another. Either way you will need to obtain any missing custom transformers to run the workspace properly.

Embedded transformers are colored green in the Navigator:  Creating_Routes

Linked transformers are colored light blue in the Navigator:  Creating_Routes

To change the default setting

Right-click on the transformer name in the Navigator, and select Link. The transformer icon color changes to light-blue to indicate it is now linked instead of embedded.



Publishing Parameters in Custom Transformers

Since attributes aren't otherwise available inside a custom transformer, you might want to publish the parameters of one or more transformers inside its definition.

Publishing a parameter allows it to be set outside of the custom transformer, and then that setting can be reused for any number of parameters.

See About Published and Private Parameters.

Exporting a Custom Transformer

When you export a custom transformer, it is saved to a separate .fmx file. This means that you can open and edit it just like a regular workspace, and you can share it with other users. Custom transformers can be:

- edited independently of any workspace
- shared with other users (via e-mail or shared directories)
- embedded or linked in any number of workspaces

How to export a custom transformer:

1. Open or create a custom transformer.
2. From the File menu, choose Export as Custom Transformer. Another instance of FME Workbench will start, and the transformer will appear on the new canvas. The title bar will display the transformer name (an .fmx file) and the following default location:

C:\Documents and Settings\\My Documents\FME\Transformers



3. Exit Workbench.

Go to your FME\Transformers folder to see the new transformer:



How can you share a custom transformer?

See Sharing Custom Transformers and Installing a Custom Transformer.

Sharing Custom Transformers

A Custom Transformer retains all of its properties, so you can distribute it much like any other file.

Sending a custom transformer via e-mail:

To e-mail a custom transformer directly from Workbench, open the transformer file and choose File > Send To.

This opens a new e-mail window in your default e-mail client, with the transformer attached.

Note: You can also send the transformer by browsing to your default Transformers folder and attaching it directly to an e-mail message.

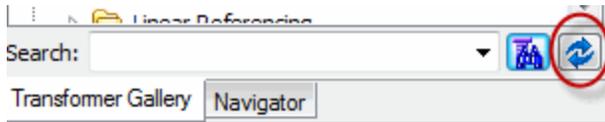
Sharing transformers through a network:

If you have set up a shared directory, any transformers in the Transformers folder are updated each time you start Workbench.

Installing a Custom Transformer

Double-click on any .fmx file in your Windows environment, and Workbench will automatically install the transformer and save it to your default Transformer directory (usually <My Documents>\FME\Transformers).

Start Workbench. If Workbench is already running, click the Refresh button in the Transformer Gallery to see the transformer in the list.



Shared Directories

- Copy the transformer (.fmx) file into any folder you have specified as a shared resource location.
- Start Workbench.
- Click Tools > FME Options and then click the Default Paths icon. See Sharing Custom Transformers.
- Refresh the Transformer Gallery to make the transformer visible.

Adding Readers

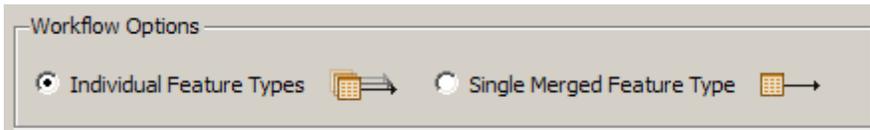
You can add one reader, or specify multiple datasets to include as multiple readers.

Adding a Reader

1. Select Readers > Add Reader.
2. Specify the format and dataset filename.
3. Specify settings and coordinate system (if applicable).

If you drag the file directly onto the Workbench canvas, the Add Reader dialog box will appear with the format and dataset name already filled in.

Workflow Options

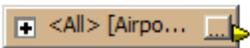


Workflow options determine the number of feature types that will be displayed on the canvas. Selecting either one will not affect the outcome of the workspace: these options determine only how the feature types appear on the canvas, and how they are interpreted within the workspace.

By default, the Individual Feature Types option will be selected. This means that for every feature type in the dataset (assuming there is more than one), Workbench will display a corresponding feature type on the canvas. This has always been the default workflow for Workbench and for most workspaces, this option is sufficient.



If, however, you want to display only one feature type on the canvas, you can select Single Merged Feature Type. All feature types in the source dataset are merged into one single feature type.



Note: If you want to separate the feature types later, you can do this through the Feature Type Properties dialog.

Initiating the Action

After you press OK, the log window will display processing information, and the feature types will appear in the reader area of your workspace.

Adding Multiple Source Datasets

To extend the input of your workspace to include multiple source datasets that are in a different format than your existing source dataset:

1. Select Readers > Add Reader.
2. Specify the source format.

- To select multiple datasets, click the Advanced Browse button . Type directly in the Directory field (you can use wildcards to include all files of a specific format), or click Add Folder button to browse for a specific directory name. Click the Subfolders box to also include all subfolders below that directory. If you know that files are the same schema, check the Identical Schemas box (this can have an effect on overall processing time if you have many datasets).
- Specify parameters and the coordinate system (if applicable).

Adding a Reader as a Resource

When you add a reader as a resource, you are inserting a reference to a dataset to be used in the workspace. This reader will not perform any actual data reading, except when up-to-date schema is required at runtime. When required, schema may be requested from that reader.

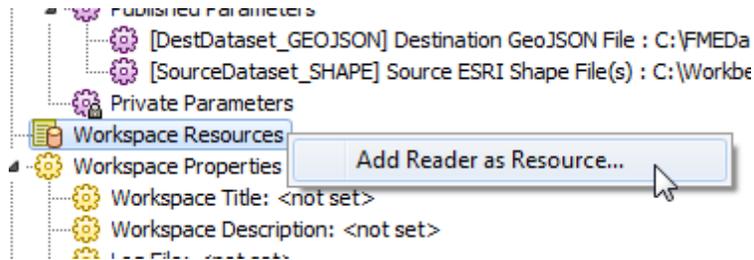
Currently it is only used by the writers that are configured to use Dynamic Parameters.

In the dynamic mode, a writer will extract schema information from one or more of the specified readers or reader resources to use during a translation. This feature is particularly useful when a writer needs to get the schema and features from several different datasets.

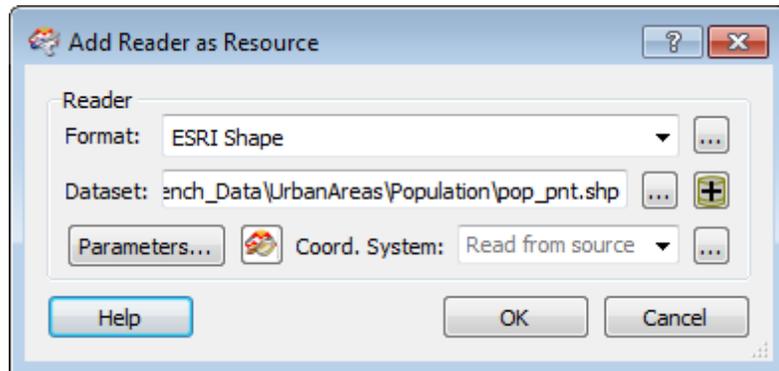
The difference between a Reader and a Reader Resource is that adding a Reader will also add all the associated feature types – a reader resource can be used as a source for schema without actually providing any feature types.

Setting up the Reader Resource

Select Readers > Add Reader as Resource or right-click Workspace Resources in the Navigator:

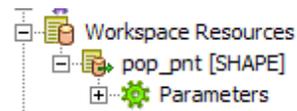


The function in Workbench is similar to adding a workspace Reader. You will need to specify the format and dataset of the resource you want to add to Workbench, and specify any additional format parameters.



After you click OK, the log file will display the processing statistics associated with the selected dataset, and you will see a Translation SUCCEEDED message, indicating that Workbench has successfully processed the dataset.

You will see an additional Workspace Resources parameter in the Navigator window:

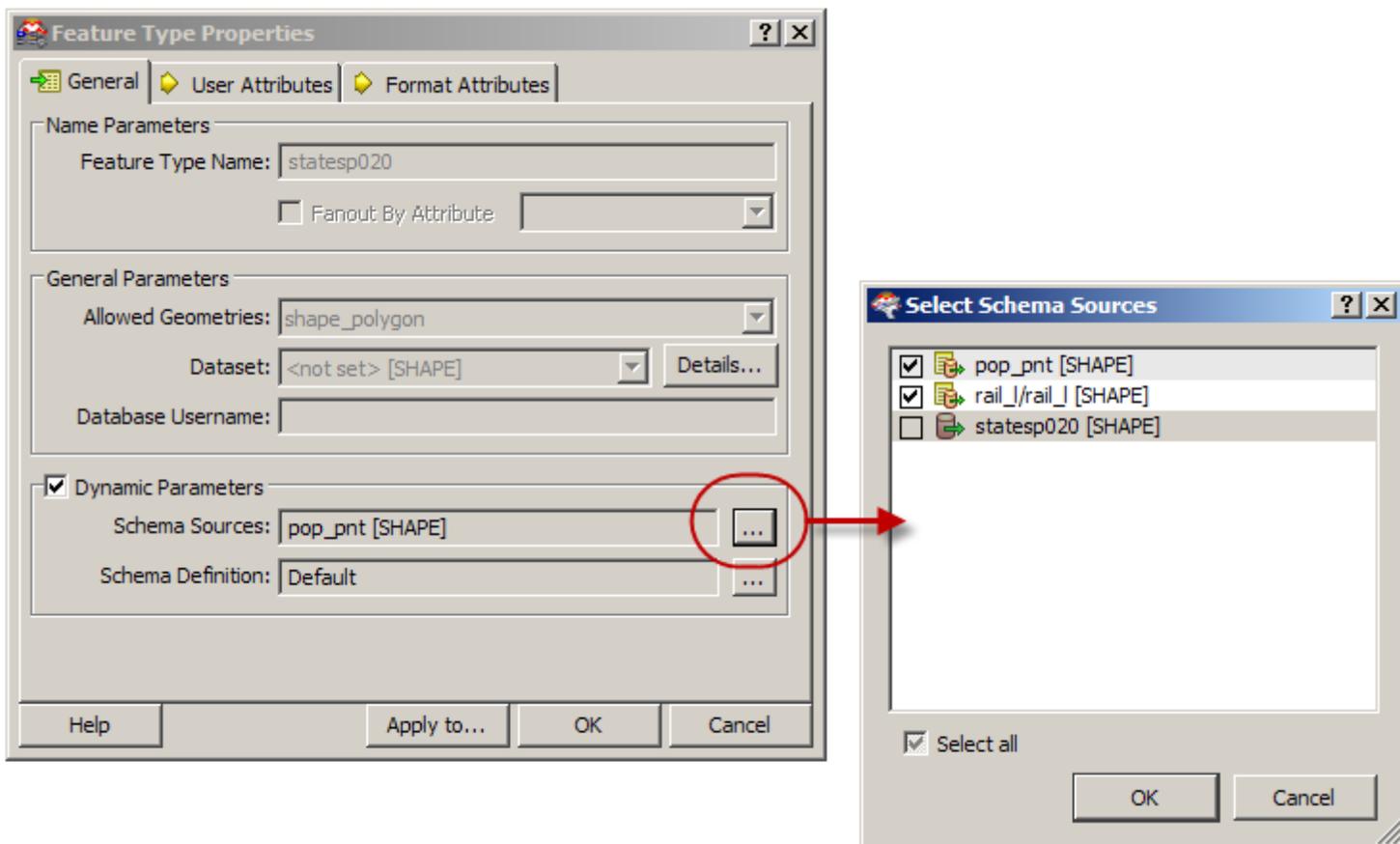


This Reader will remain in the list as a resource of the workspace.

Setting up the Writer to use the Reader Resource

When you want to use the Reader in a translation, open the Writer Feature Type Properties. Clicking Dynamic Parameters enables the Schema Sources and Schema Definition fields.

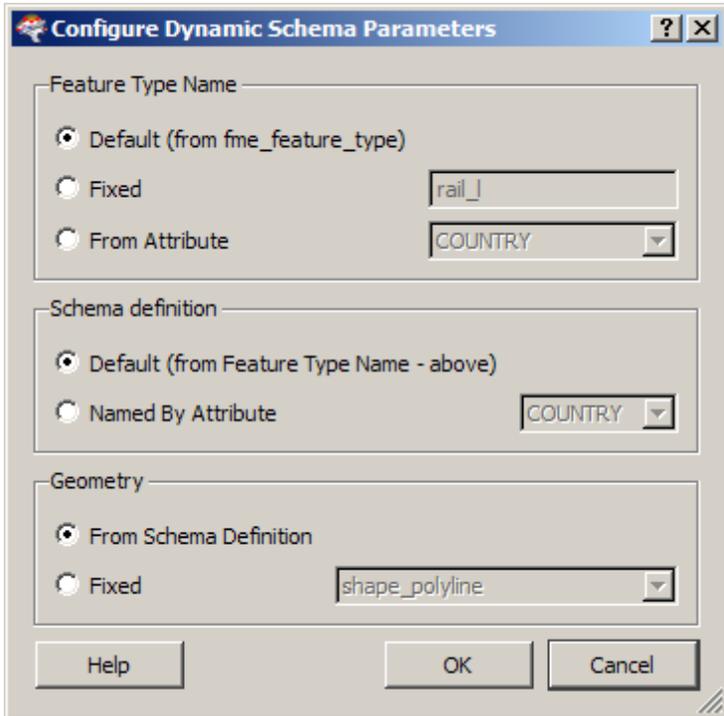
Click the Browse button beside the Schema Sources field to select from a list of existing schema readers. Only the readers that you have selected will be used at runtime.



Changing the Dynamic Schema Parameters (Advanced FME)

In most cases, you can use the default Dynamic Schema parameters.

However, the dynamic writer can operate with a few variations on the schema definition. These are feature type fanouts, more complex schema reader definitions and defining the destination feature geometry.



Feature Type Name

This controls the name of the destination feature type that is written. *Default* uses either the source schema or the schema reader feature type name. *Fixed* uses the name of the feature type in the workspace. *From Attribute* has the same effect as the dataset fanout with a new feature type for each value of the specified attribute.

Schema Definition

If the feature types in the Schema Reader do not match the `fme_feature_type`, then you can set an attribute that defines the name of the schema reader feature type to be used for the schema definitions.

Geometry

Some formats, e.g. ESRI Shape, have a fixed geometry. This option defines where the definition for the geometry is derived from.

See also

About Dynamic Workspaces

Reader Settings

Limiting the Number of Features to Read

You can limit the number of features that you want Workbench to read, and then set the feature number at which to start. This feature is useful if, for example, you have a table that contains many spatial elements and you want to process only a portion of those elements for testing purposes.

Open the Advanced node under the source dataset, then double click **Max features to read** to set the maximum limit, and then **Start feature** to set a feature above 1.



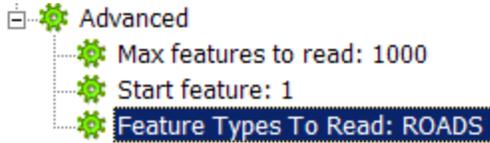
In this example, when you run the Workspace, it will only read the ROADS feature type, starting at feature 1, to a limit of 1000.

Setting the Feature Types to Read

This option is a quick way to restrict the number of feature types to be read, especially if a workspace references dozens of source feature types and not all of them are required during a translation run.

Double click Feature Types To Read and check the applicable feature types in the dialog that appears.

Click OK to accept the changes, and the feature types will appear in the parameter below:



Alternatively, you can enable/disable a source feature type node to achieve the same result.

Since you can publish this parameter, you can also use it to restrict feature types for reading from the command line. Once the parameter is published, a sample command line is shown in the log. For example:

```
fme.exe <workspace>.fmw --FeatureTypesToRead "<roads> <streets>"
```

Merging Similar Datasets

When data sources reside in different locations, you can select them using the Multiple Datasource Selection Tool. This tool is sometimes referred to informally as the "swizzler".

You can add datasets of the same format and with the same schema (data model) to any source dataset you have already defined in your workspace. These datasets will be merged together when you run the translation.

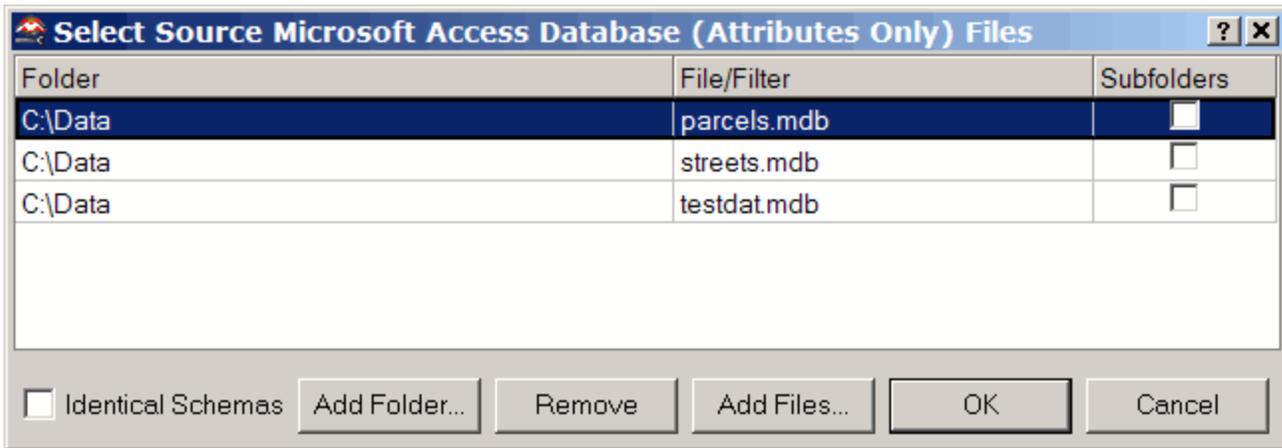
File-Based Formats

- Double-click the dataset name in the [Navigator view](#).



- Click  in the dialog that appears to display the [Select Datasets](#) dialog. You can type directly in the Folder field, and use wildcards to

include all files of a specific format. For example



C:***.dgn

will merge all the .dgn files on your C drive.

- Check the Identical Schemas box **if you know that all files have the same schema.**

Note: This is simply a time-saving function; there will be no difference in the output of the translation. However, if you know the files have the same schema and you check the box, FME will not have to perform an initial scan of all the files to determine their schemas. Instead, FME will take the first file as being representative of the data model.

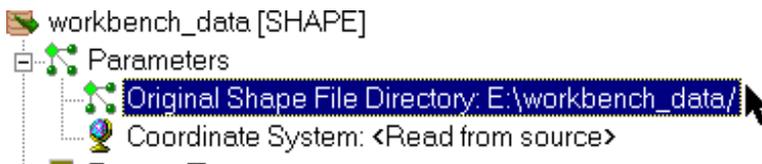
- Click **Add Folder** to browse for a specific directory name, and check the Subfolders box to also include all subfolders. All files that are in the specified format in those folders will be included. You can also:
 - Click **Add Files** to select individual files.
 - **Ctrl+left-click** to select multiple files.
 - Click **Remove** to delete single or multiple files based on your selection.
- Click **OK**. The new datasets will append to the original dataset name. To see these results reflected in the Navigator view, float the cursor over the dataset name.

When you run the translation, it will merge the specified datasets. The log pane will display detailed information during the translation.

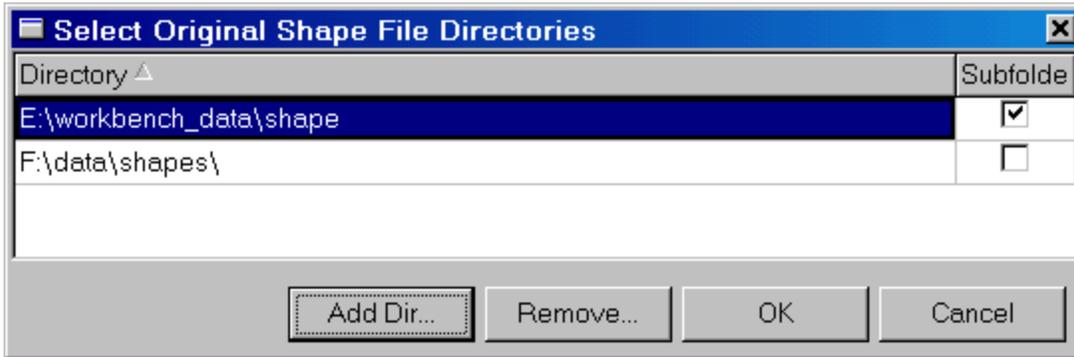
Directory-Based Formats

When you are translating formats that support coverages, you'll have the option to choose folders:

- Double-click the dataset name in the [Navigator view](#).



- Click  in the dialog that appears to display the **Select Directories dialog**.



- Click **Add Folder** to browse for a specific folder name. Check the Subfolders box to also include all subfolders.
- Click **OK**. The new datasets will append to the original dataset name. To see this reflected in the Navigator view, float the cursor over the dataset name.

When you run the translation, it will merge the specified datasets. The log pane will display detailed information during the translation.

Note: You can add more datasets to your workspace at any time.

Adding Multiple Writers

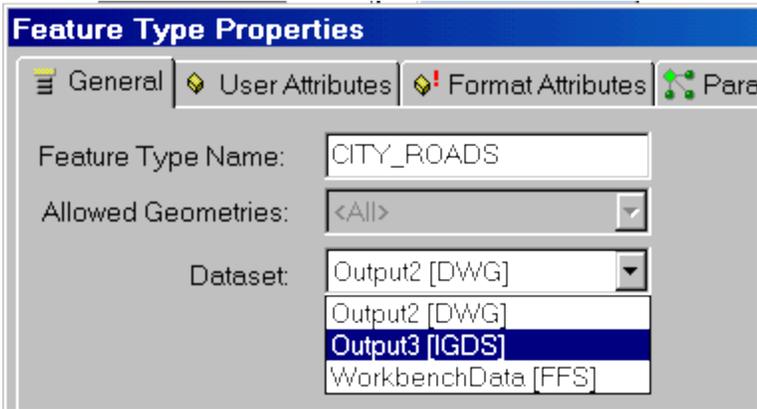
You can write to multiple destination datasets. (You can also set the output coordinate system for each dataset to its own unique coordinate system.)

- Select Add Writer from the Writers menu on the menu bar.
- Specify the format and dataset filename. See Choosing from the Formats List.
- Specify parameters and coordinate system (if applicable), and click OK.

The log window will display processing information, and a new Reader will appear in the navigator view.



- Add new feature types for the new destination dataset using one of the following methods:
 - Select Add Feature Type from the Writers menu. You will be prompted to create a new feature type. You can also right-click on the background of the workspace and select Insert Writer Feature Type. From here, you can set the geometry (for some formats), the feature type name, and user attributes. See Changing Feature Type Properties.
 - Select the Feature Type, right-click and choose Copy Attributes From Feature Type. Choose the destination feature types from an existing destination dataset and they will copy to the new dataset. Now when you display your destination feature type properties, you will see a **drop down list of datasets**, with unique filenames. You can also select an existing destination dataset, copy and then paste it onto your workspace.

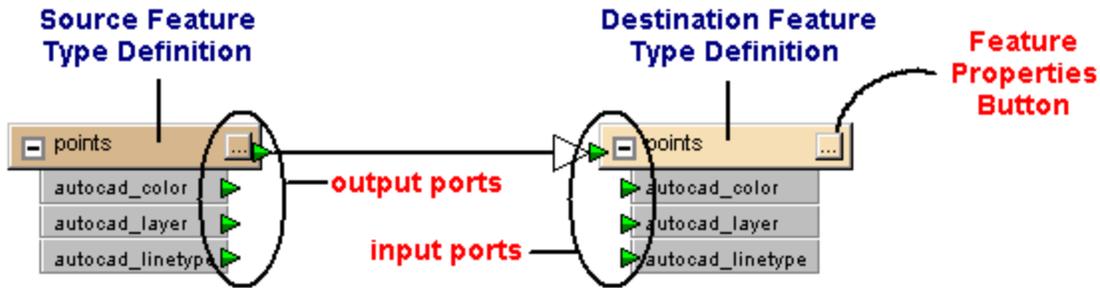


- If you don't want to add an entire dataset you can import another dataset's definitions. Select Import Feature Types from the Writers menu, and then adjust the imported feature types so that they more closely reflect your dataset.

You can change the displayed name of a writer by right-clicking on it in the Navigator pane, and choosing Rename.

Defining Destination Data Characteristics

When FME generates the initial workspace, it reads the feature type definitions from the input data. It initially displays the same feature type definitions and attributes for the output data. So, you will often have something that looks similar to this:



First, you might want to rename the destination feature type so the name more closely reflects what you want to see in the output data. You might also want to change the attributes, by deleting or renaming them, or by editing their properties.

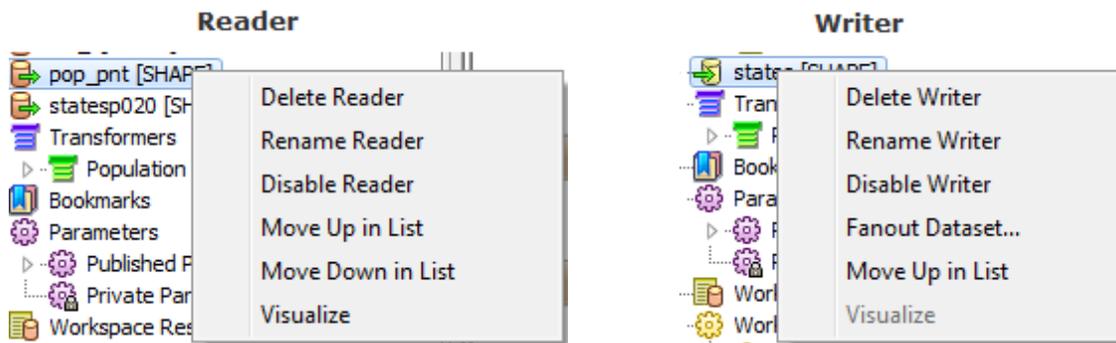
When the output from the source type is connected to the destination type, there is an implicit connection made between all attributes that have the same name. When you rename an attribute, or add a new attribute, you must physically connect the ports to create the desired links.

Reordering, Disabling, and Deleting Readers/Writers

You can reorder or disable readers and writers in a workspace if you want to change the order in which Workbench processes them.

Command Menu

Click on a reader or writer in the Navigator and right-click to display the command menu.



- To reorder: Select *Move Up in List* or *Move Down in List*.
- To disable: Select *Disable Reader* or *Disable Writer*. This will temporarily remove the reader or writer from the workspace without actually deleting it. Disabled objects in the workspace (in this case, feature types) turn gray.
- To remove a reader or writer from the workspace, select *Delete Reader* or *Delete Writer*. You will be prompted to confirm that you want to remove the reader/writer and its feature types.

Menu Bar

Select *Remove Reader* or *Remove Writer* from the Readers or Writers menu.

About Feature Types

In FME, a *Feature Type* refers to the source dataset schema or structure.

Every format used by FME identifies features through a classification scheme. This classification is known in FME as a feature type. Feature types are used to differentiate between different types of features (for example roads, rivers, buildings, contours).

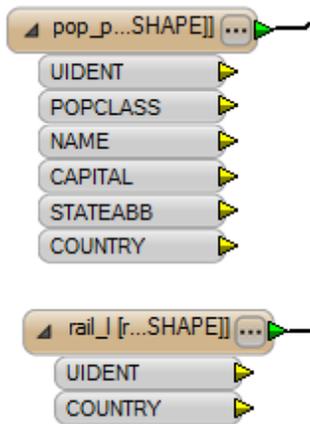
The feature type for any particular format is listed in the *FME Readers and Writers* manual under the "Format Quick Facts" section (see FME Readers and Writers Reference under the Workbench help menu).

Some examples are:

- In MicroStation Design File (DGN) format, each level is a Feature Type.
- In AutoCAD Drawing File (DWG) format, each layer is a Feature Type.
- In Smallworld format, each class is a Feature Type.
- In ESRI Shape format, each file basename (*.shp file) is a Feature Type.

A Feature Type is contained within a dataset.

When you create a workspace using a dataset, the Feature Types present in that dataset are shown on the left-hand side of the workspace canvas:



Managing Feature Types

Importing Feature Types

The Import Feature Types function adds feature types to an existing dataset by importing the schema from any dataset. This second dataset could be the same one as is being read, but this isn't necessary; in fact, it doesn't even need to be in the same format – the schema definition is the important part, and FME can read that from any format.

1. From the Readers or Writers menu, select Import Feature Types. If you have more than one reader, you will be prompted to select the dataset to which you want the schema added.
2. From the Import Feature Types dialog, specify the format and dataset (and parameters and coordinate system, if applicable) and click OK.

The log will display processing information and the feature types will be added to the selected dataset.

You can also choose to import feature types from a selected format in an entire directory by clicking the Advanced Browser button .

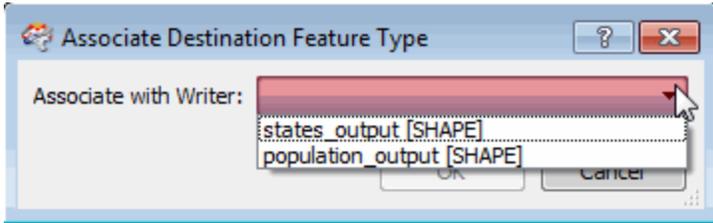
Creating Feature Types

To output certain attributes or results to a different output layer, you can create new feature types for writer datasets.

1. From the Writers menu, choose Add Feature Type.
2. When the Feature Type Properties dialog appears, rename the feature type.
3. To attach attributes to the new feature, click the User Attributes tab.

Duplicating Feature Types

To quickly copy feature types to a writer, right-click on a feature type and select Duplicate (on Writer). If you have one writer, the feature type will automatically be duplicated; if you have more than one writer, you will be prompted to choose the writer to associate with the feature type:



Removing Feature Types

To remove selected feature types, choose Remove Feature Types from the Readers or Writers menu and check the applicable boxes.

You can also select the feature types in the canvas or the Navigator pane and press the Delete key.

Moving Feature Types

Move Feature Type is enabled only when you have more than one destination dataset and at least one destination feature type in the workspace.

It is used to quickly change the dataset for a group of feature types. For example, if you have a MIF dataset with feature types and Oracle dataset without any feature types. You can quickly switch all the MIF feature types to Oracle feature types by changing their associated dataset. All the MIF feature types will be moved over to the Oracle dataset.

You can also do this using one feature type at a time by following these steps:

1. Display the Destination Feature Type Properties dialog.
2. Click on the "Dataset:" drop-down list box and choose from the list of datasets (this changes the feature type's dataset).

Enabling Feature Types

To output only one feature type, right-click on a feature type and select *Enable Only This Feature Type*.

To enable multiple feature types, select the feature types, then right-click and select *Enable Objects*.

Disabling Feature Types

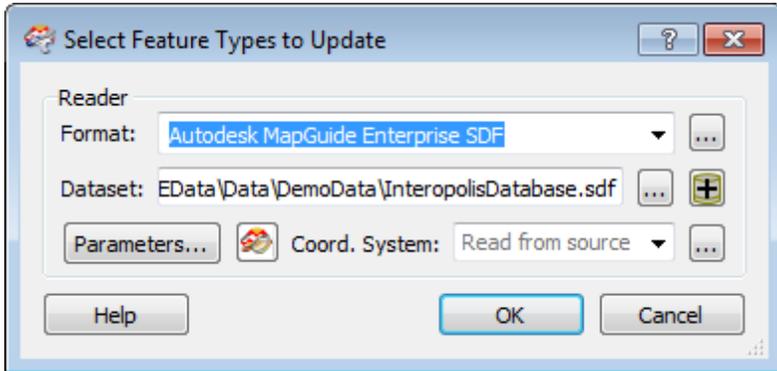
To disable a feature type and all of its links, right-click on a feature type and choose *Disable*.

To disable multiple feature types, select them then right-click and select *Disable Objects*.

Updating Feature Types

If the structure of the data changes in your workspace (for example, the schema changes or an attribute type changes) you will usually want to keep your workspace up-to-date. This is a particularly useful feature if you are working with databases, or if you are sharing data within a workgroup.

Select *Readers > Update Feature Types* or *Writers > Update Feature Types*. You can also access this feature by right-clicking on the feature type and choosing Update from the command menu.



Click OK to reload the feature type.

Understanding Attributes

Workbench supports two types of attributes:

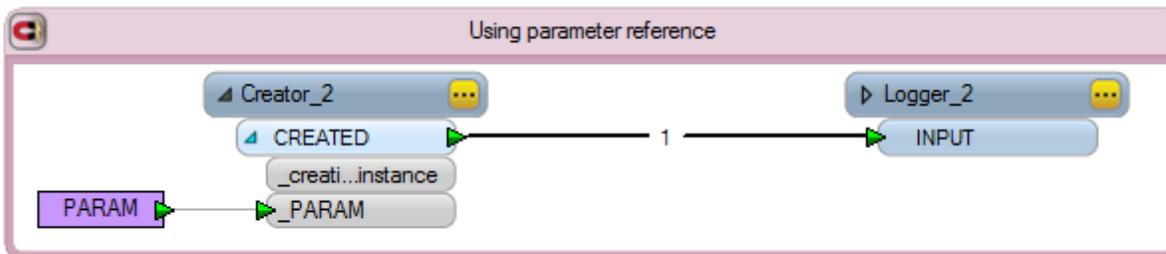
- *User attributes* can have their names and data types adjusted to meet the needs of the user. They correspond to the columns in any attribute tables created during the translation.
- *Format attributes* are fixed by FME. Descriptions of the attributes available for each reader (input format) and writer (output format) are given in the *FME Readers and Writers* manual. (Choose FME Readers and Writers Reference from the Workbench > Help menu.) Format attributes can be *exposed* in a workspace so that they can be set to particular values.

Insert Parameter Reference

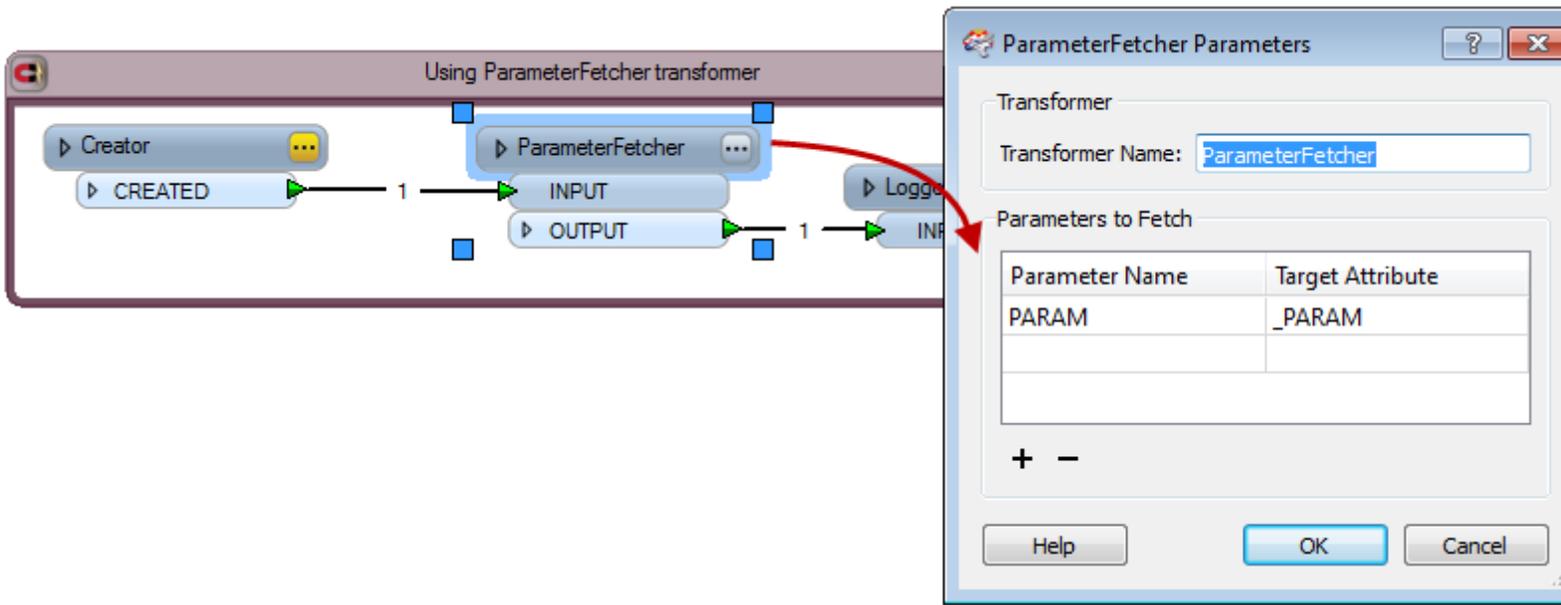
Parameter references are another way of accessing (published or private) parameters in Workbench. They are an alternative to using ParameterFetcher transformers.

Parameter references look like constant nodes, but are purple, instead of gray. Like constant nodes, they can be attached to attribute ports. Doing so will assign the value of that parameter (at runtime) to that attribute.

The greatest advantage to using parameter references over ParameterFetcher transformers is that when the attribute field is visible, the parameter reference is also visible. This makes the association very clear:



If you compare this to the ParameterFetcher transformer, the only way to see the fetched parameter is either by opening the transformer properties (as shown here), or by adding a summary annotation.

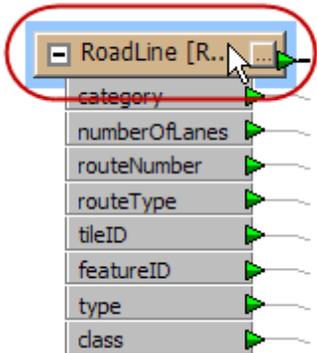


One disadvantage to the parameter reference is that when the attribute field is hidden, so is the parameter reference. For example, in some workspaces there are thousands of exposed attributes, and it may be visually distracting to show all attribute fields. If you hide attribute fields, any parameter references are also hidden.

Feature Type and Attribute Property Menus

Working with Source (Reader) Feature Types

Right-click on a Feature Type, and choose an option from the command menu.



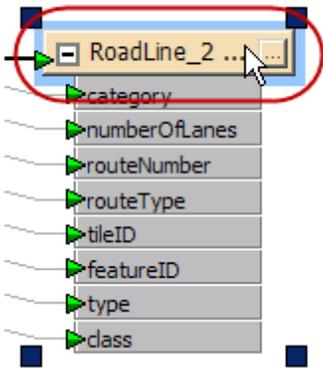
See all menu choices for source (Reader) feature types:

Cut, Copy, Delete	Performs these functions on the selected feature type only.
Duplicate on Writer	Duplicates the feature type on the writer side, and automatically connects all the attributes.
Bring to Front, Send to Back	Useful if you have a large workspace, with many feature types.
Zoom to Feature Type	Zooms and focuses the canvas on the selected feature type.

Update	If you add or rename an attribute, this function allows you to update feature types without having to remove and reload them.
Enable Only This Feature Type	Disables (and grays out) all other feature types
Disable	Disables this feature type. Menu selection changes to Enable when the feature type has been disabled.
Connect Visualizer	Connects a Visualizer transformer.
Connect Logger	Connects a Logger transformer.
Attach Annotation	Allows you to add a comment associated with this feature. The comment, connected by a leader line, will stay joined to its associated feature type.
Show Summary Annotation	Shows hidden object information. This can be useful when printing out a workspace.
Expose Attributes	For advanced FME users: allows you to "expose" or make visible built-in FME attributes.
Pair With	Connects selected reader and writer feature types.
Disconnect	Removes all connections.
Visualize	Displays the data in the FME Universal Viewer, or the FME Data Inspector, if you have changed your default Viewer in the FME Options dialog.
Properties	Rename the feature type, or view or edit the feature type properties. (This is the same as clicking the Properties button on the top right-hand side of the feature type.)

Working with Destination (Writer) Feature Types

Right-click on a Feature Type, and choose an option from the command menu.



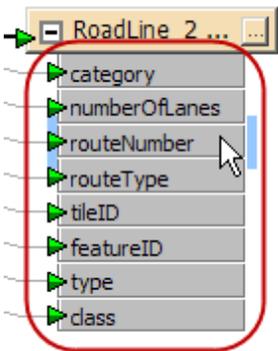
See all menu choices for destination (Writer) feature types:

Update	If you add or rename an attribute, this function allows you to update feature types without having to remove and reload them.
Enable Only This Feature Type	Disables (and grays out) all other feature types

Disable	Disables this feature type. Menu selection changes to Enable when the feature type has been disabled.
Add Attribute	Adds a new attribute. Note that you will have to physically connect this attribute with its source.
Expose Attributes	For advanced FME users: allows you to "expose" or make visible built-in FME attributes.
Copy Attributes from Feature Type	Duplicates the attributes from another feature type.
Copy Attributes from Transformer	Duplicates the attributes from a transformer.
Sort Attributes	Alphabetizes the attribute names.
Rename (F2)	Type a new name and press Enter.
Attach Annotation	Allows you to add a comment associated with this feature. The comment, connected by a leader line, will stay joined to its associated feature type.
Show Summary Annotations	Shows hidden object information. This can be useful when printing out a workspace.
Pair With	Connects selected source and destination types.
Disconnect	Removes all connections.
Visualize	Displays the data in the FME Universal Viewer, or the FME Data Inspector, if you have changed your default Viewer in the FME Options dialog.
Properties	Rename the feature type, or view or edit the feature type properties. (This is the same as clicking the Properties button on the top right-hand side of the feature type.)

Working with Attributes

Right-click on either a reader or writer attribute, and choose an option from the command menu.



See all menu choices for editing attributes

Rename Attribute	Type a new name and press Enter. Note that you will lose any association with any previously connected attribute.
Delete Attribute	Remove attribute from the list.

Disconnect Attribute	Disconnect attribute from its source. This is the same as selecting and deleting the connection.
Set to Constant Value	Assign a fixed value to the attribute (writer attributes only).
Move Attribute up/down	Moves the attribute up or down in the list. You might want to do this so the display looks visually cleaner, or to organize attributes for ease of reference.
Properties	Rename, view or edit the attribute properties. (This is the same as clicking the Properties button on the top right-hand side of the feature type, and clicking the User Attributes tab.)

About Feature Type Properties

Because a destination feature type is simply a definition of what is to be created (in most cases, it does not yet exist), you can edit a destination feature type in a number of ways that aren't permitted for source feature types.

More Information

Editing Feature Type Properties

Editing Reader Feature Type Properties

Editing Writer Feature Type Properties

Editing Feature Type Properties

Feature type properties are dependent on the format and dataset. However, in general, you can edit feature types on both the reader (source) and writer (destination) sides:

Editing Reader Feature Type Properties

Specify merge settings and expose format attributes.

Editing Writer Feature Types Properties

Edit general characteristics, user attributes, format attributes or parameters (for formats that support table creation), specify Dynamic Parameters.

Displaying the Feature Type Properties

To display the Feature Type Properties dialog:

1. Click the Properties button on the upper right of the feature type, or
2. Right-click on a feature type and select Properties from the menu.

Editing Reader (Source) Feature Type Properties

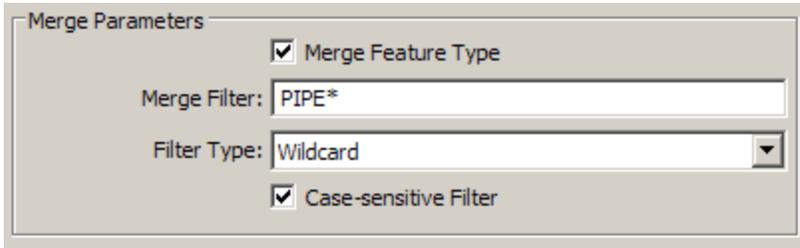
Editing source feature types is an advanced user function and is suitable mainly for testing purposes (it can be useful, for example, if you are setting up a workspace and you do not have access to the original source data).

General – Name Parameters

By default, the General tab is selected. You can rename the feature type by simply clicking in the Feature Type name field and typing a new name.

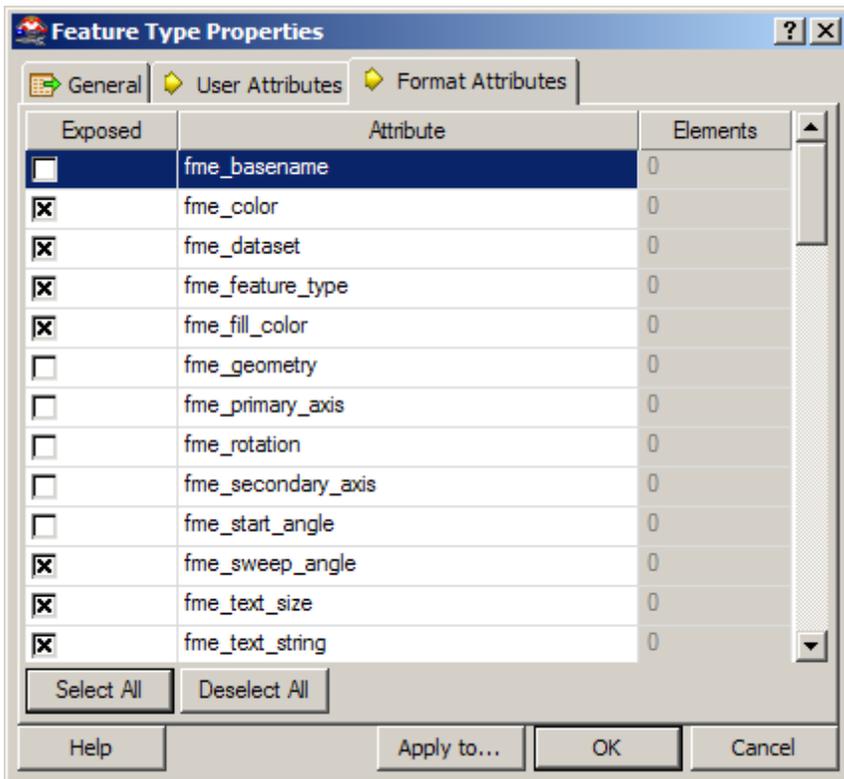
Reader Feature Types: Merge Parameters

For reader feature types, you can specify Merge properties.



Format Attributes

Click the Format Attributes tab.



These are built-in FME attributes that you can "expose" or make visible so that you can set them to particular values and connect them to other format-specific attributes. Apply to... exposes the attributes for multiple, selected feature types.

When you expose an attribute on the reader side, it allows its value to be used in the workspace.

Applying Changes to Multiple Reader Feature Types

1. Open the Feature Type properties dialog.
2. Click the **Apply To** button, and the Change Multiple Feature Types dialog appears.
3. By default, all feature types are listed in the **Feature Types to Change** field. Click the Browse button to check/uncheck feature types from the feature type list that appears (specify a filter to isolate certain feature types). Click OK to close the list box.
4. You can also change merge settings or choose format attributes to expose for the selected feature types.
5. Click OK to close the Change Multiple Feature Types dialog.
6. To confirm the changes to all feature types, including the current one, click OK in the Feature Type Properties dialog. You will be asked to confirm any changes to other feature types. Note that if you click Cancel, none of the changes will be applied.

Editing Writer Feature Type Properties

General – Name Parameters

By default, the General tab is selected. You can rename the feature type by simply clicking in the Feature Type name field and typing a new name.

You can also choose to fanout the feature according to the selected attribute name. See Fanout properties.

Database User

For some formats, such as Oracle, Workbench will display the table name in the Database Username field.

For example, if you specify a database table called CITY.ROADS, the Feature Type is ROADS and the database user is CITY. This field allows you to change the database user (for example, to VANCOUVER) on the output feature type.

General – General Parameters

Changing Geometries

When a destination feature type is automatically created by FME, the destination geometry is assumed to be the same as the source dataset. For example, a source dataset containing line features is assumed to lead to a destination dataset containing line features. However, this is not always the case. Transformers within the workspace may be used to alter feature geometry.

If feature geometry will change during the translation, and the destination format demands a strict definition of allowed geometries (Shape datasets require this), then the geometry type can be selected here from a drop-down list. This is not only a desirable quality, but vital too, since not to make such a change would lead to error messages in the subsequent translation.

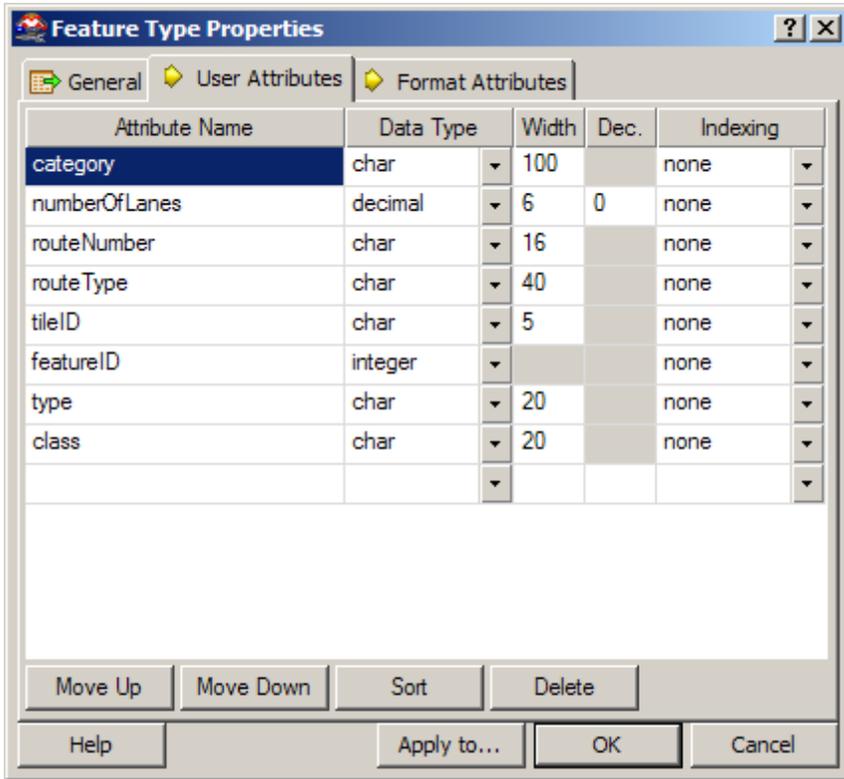
For many formats, all geometries are automatically allowed – in these cases, the Allowed Geometries field is disabled.

General – Dynamic Parameters

Dynamic Parameters are used in conjunction with Adding a Reader as a Resource.

User Attributes

Click the User Attributes tab.



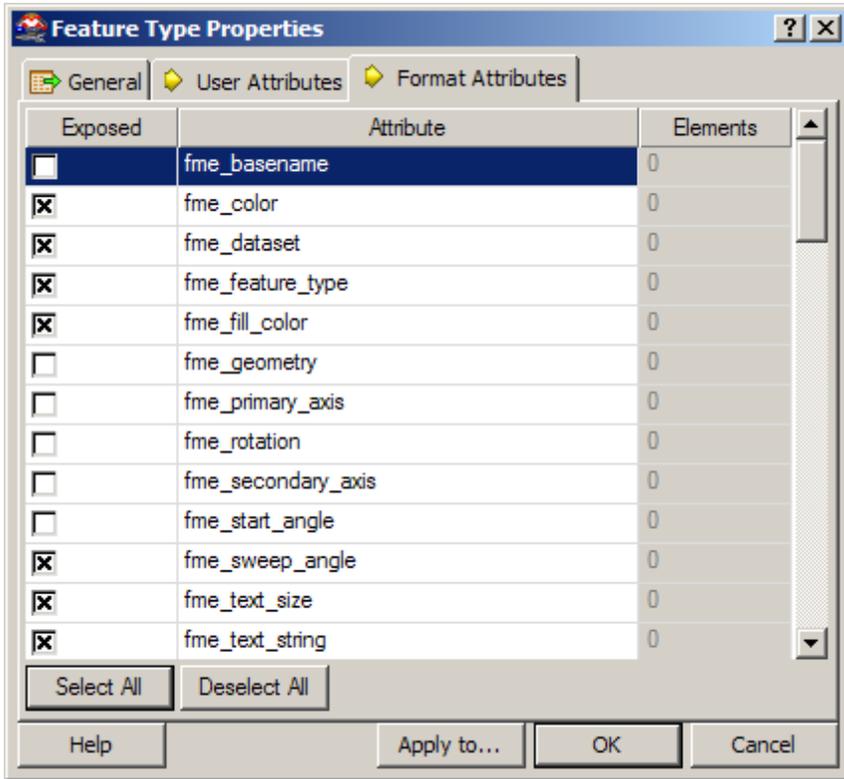
This is a convenient way to alter the schema of a feature type. If a format doesn't allow user attributes, this tab is not present.

On writer feature types, you can easily add, delete, move, rename, sort, and change attributes' data types. For some data types, you can define a width (that is, the width of the field). For other data types, you can specify a number of decimal places. If a column is grayed out, then you can't set a width or number of decimal places.

Tip: To quickly remove several attributes at once: Ctrl + click to select them, and then press Delete.

Format Attributes

Click the Format Attributes tab.



These are built-in FME attributes that you can "expose" or make visible so that you can set them to particular values and connect them to other format-specific attributes. Apply to... exposes the attributes for multiple, selected feature types.

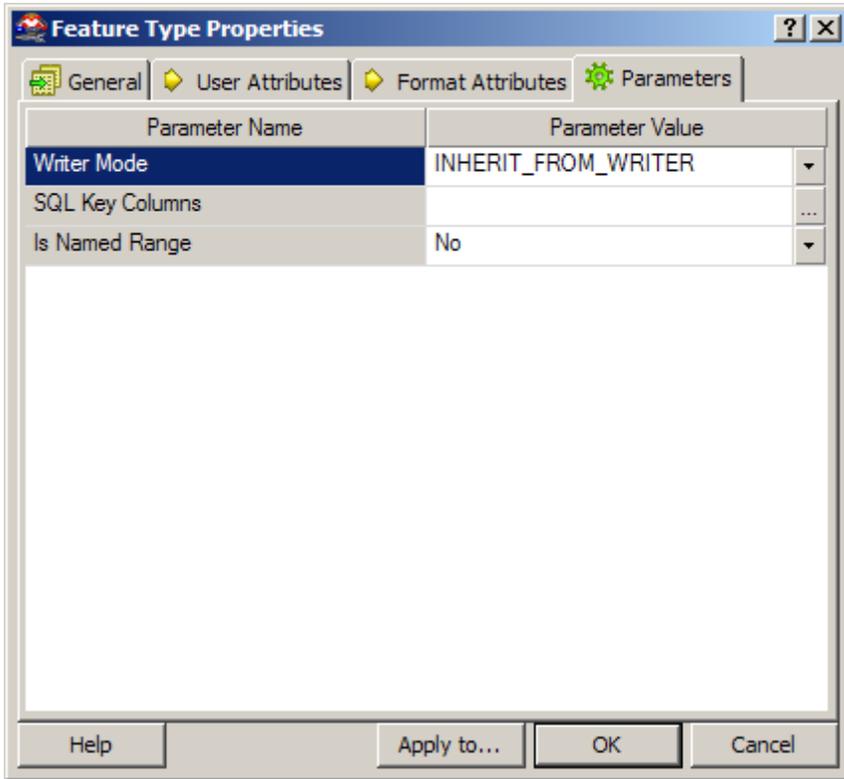
Exposing an attribute on the writer side allows it to be set to a value (for example, you might want to set a *pen_width* or a *line_color* to appear in the destination data).

This is key to fully using FME. The special attributes' purposes are described in the Feature Representation section of each reader (input) and writer (output) in the *FME Readers and Writers* manual (available under the Workbench help menu). These attributes allow a wide variety of special things to be done with formats (like setting line thickness, creating special entities, setting particular bits or bytes).

Parameters

Click the Parameters tab. This tab appears if a format (such as Oracle or SDE) supports table creation.

If a table already exists and you select Yes under the Parameter Value column, then the remaining fields are disabled and Workbench will retrieve the values from the existing table. Otherwise, select No and the table will be created from the parameters that you enter here.



Applying Changes to Multiple Output Feature Types

1. Open the Feature Type properties dialog.
2. Click the **Apply To** button, and the Change Multiple Feature Types dialog appears.
3. By default, all output feature types are listed in the **Feature Types to Change** field. Click the Browse button to check/uncheck feature types from the feature type list that appears (specify a filter to isolate certain feature types). Click OK to close the list box.
4. You can also change the dataset, geometry, fanout settings, or choose format attributes to expose for the selected feature types.
5. Click OK to close the Change Multiple Feature Types dialog.
6. To confirm the changes to all feature types, including the current one, click OK in the Feature Type Properties dialog. You will be asked to confirm any changes to other feature types. Note that if you click Cancel, none of the changes will be applied.

Feature Types to Read

[Reader](#) > [Parameters](#) > [Advanced](#) > [Feature Types to Read](#)

It is quite a common scenario to want to set up an FME translation with many source feature types, but only run the workspace on a subset of these layers.

The most obvious method is to delete or disable the other feature types before running the workspace, but this could involve time-consuming edits to the workspace, and then even more edits to return it to its original state.

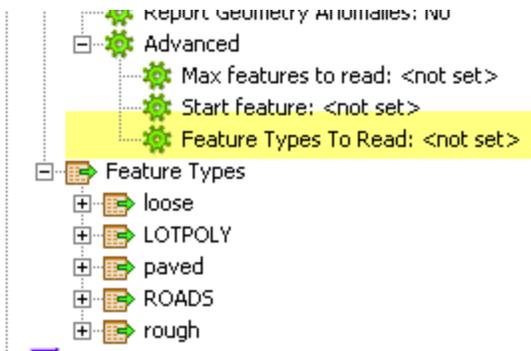
To simplify this task, Workbench includes an Advanced reader parameter to allow the selection of feature types to read.

Because the parameter is publishable, you can set up a generic workspace, and then you can choose which feature types to read at run time.

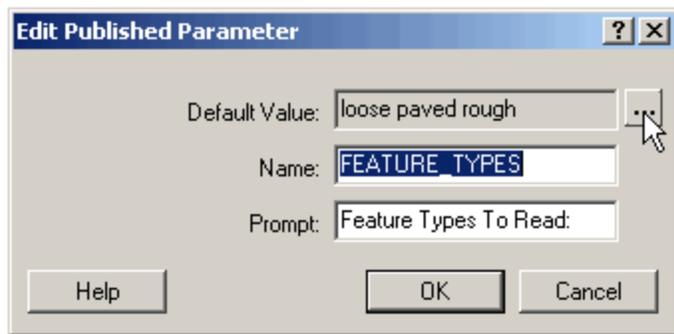
Example

In this example, a workspace contains five different feature types:

loose, LOTPOLY, paved, ROADS, rough



Right-click and choose Publish Parameter. Click the Browse button to select the feature types to read, and then click OK.

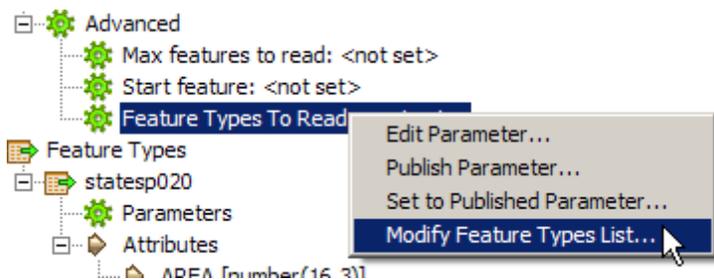


When you run the workspace, you will be prompted for the feature types to include in the translation, and then only the selected feature types will be read.

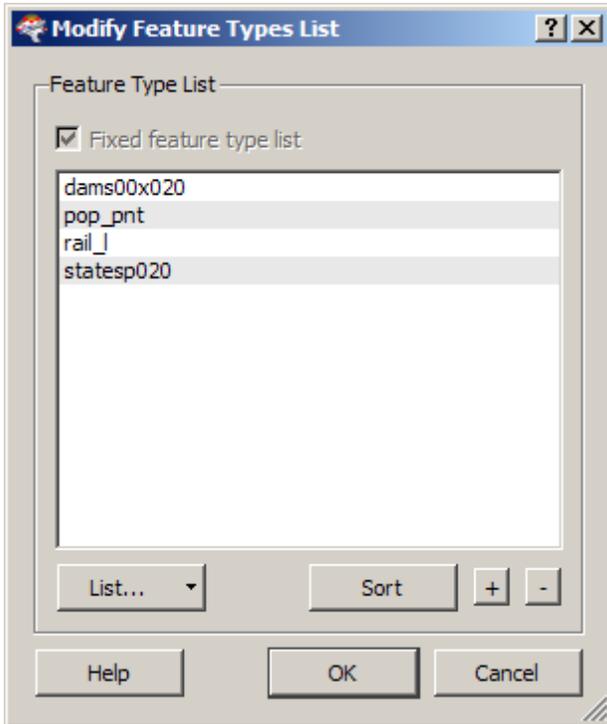
Modify Feature Types to Read

Modify Feature Types To Read is an Advanced source dataset level parameter that allows you to specify exactly which feature types to read from the dataset.

By publishing the parameter, this selection can actually take place at run-time (for example, by a user who runs the workspace via an FME Server web page).

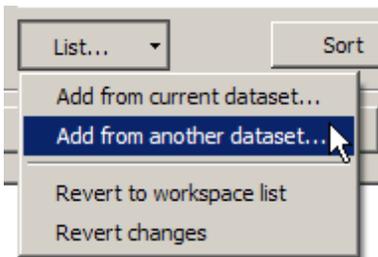


You can choose feature type names from a static list that is built from the feature types that already exist in the workspace for the given dataset. (However, if you have previously chosen to Merge Feature Types, there may be only one feature type in the workspace.)



If value of this parameter is <not set>, then all feature types in the workspace will be read.

From the List menu, you can select the feature type names you want to add to the new list, either from the current dataset or from another dataset. If you choose another dataset, you will be prompted to select it from a Specify Data Source dialog. When you press OK, the feature type(s) will be added to the Feature Type List.



You can also revert recent changes or revert to the original workspace list.

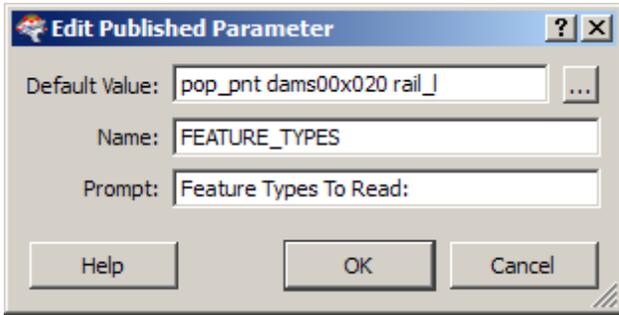
You can continue adding feature type names and sorting the names to build the list.

After you press OK, Workbench will build a new static list of Feature Types To Read.

Publishing the Feature Types to Read Parameter

You can publish the Feature Types to Read Parameter, which allows you to choose the Feature Types to Read at runtime. Right-click on the Feature Types to Read parameter and choose Publish Parameter.

Click the Browse button to select from or modify the list of feature types. You can also edit the name of the published parameter, and the Prompt that will display at runtime.



After the parameter has been published, you can always modify the list by right-clicking on the Published Parameter and choosing selecting Modify Feature Types List.

Editing Geodatabase Domains and Subtypes

If you're writing to Geodatabase, some unique attribute types (*field types*) will be written out as part of the table.

After you generate your initial workspace, click the User Attributes tab on an output feature type. User attributes contain unique data types that you can further define. Most of the attribute types offered are normal attribute types such as char(n), integer, and date; however, domains and subtypes are unique to the geodatabase.

coded_domain

A coded_domain defines a set of values for an attribute in a geodatabase, and consists of a code and its equivalent value. For example, for a road feature class, the numbers 1, 2, and 3 might correspond to three types of road surface: gravel, asphalt, and concrete. Codes are stored in the geodatabase, and corresponding values appear in the attribute table.

1. After you click the Edit button beside the Data Type field, the Edit Coded Domain dialog appears.
2. Enter a domain name (for example, *road_type*)
3. Assuming the domain does not already exist in the database, uncheck the box beside **Domain already exists in dataset**.

Note: If you uncheck the box and the domain does already exist, the Geodatabase writer will validate the definition of the domain and log a message if the definition in Workbench is different than the definition in the geodatabase (but will not overwrite the existing domain).

4. Select a field type (boolean, char(n), date, double, float, integer, or smallint) from the drop-down list; in this case, you might select integer.
5. Enter codes and corresponding descriptions. For example, you might enter 1 for gravel, 2 for asphalt, 3 for concrete, etc.
6. Click OK to set the values.

range_domain

A range_domain defines the range of values for a numeric attribute. For example, the permissible range of values for a lot length might be between 70 and 100 feet.

1. After you click the Edit button beside the Data Type field, the Edit Range Domain dialog appears.
2. Enter a domain name (for example, *house_age*)
3. Assuming the domain does not already exist in the database, uncheck the box beside **Domain already exists in dataset**.

Note: If you uncheck the box and the domain does already exist, the Geodatabase writer will validate the definition of the domain and log a message if the definition in Workbench is different than the definition in the geodatabase (but will not overwrite the existing domain).

4. Select a field type from the drop-down list; in this case, you might select Date.
5. Enter codes and corresponding descriptions. For example, you might enter 19820101 to 1990101.
6. Click OK to set the values.

Note: Date attributes must be of the form YYYYMMDD, YYYYMMDDHHMMSS, or HHMMSS. If the last format is used, then the YYYYMMDD portion of the date will be set to 19000101.

subtypes

Subtypes are a subset of features in a feature class or objects in a table that share the same attributes. For example, the features in a vegetation feature class could be categorized into three subtypes: oak, maple, and birch. Creating subtypes can be more efficient than creating many feature classes or tables in a geodatabase.

In the geodatabase, the field must be an integer type in order to be able to create subtypes on it.

For subtypes, descriptions can be supplied as strings, in which case the codes are generated by the Geodatabase writer. For `subtype_codes`, the input list consists of pairs of codes and corresponding descriptions.

The first code in the list will be used as the default subtype code. If only descriptions are specified, the code created for `value1` will be used as the default subtype code. For instance, if you specify `subtype_codes 1:a:3:b:4:c:5:d`, then 1 (which maps to a) will be used as the default code.

1. After you click the Edit button beside the Data Type field, the Edit Subtypes (or Subtype Codes) dialog appears.
2. Enter a subtype description. For example, if a field is named `house_type`, you might enter descriptions `rancher`, `two-storey`. If you're defining `subtype_codes`, enter a corresponding integer code. Note that the first subtype is always used as the default.
3. Click OK to set the values.

Notes:

- Each table can have only one subtype.
- All the codes have to be unique and valid integers.
- All the code,description pairs have to be unique.
- You cannot add subtypes to an existing table. If you do, it will be ignored and the table will use the existing subtype, if one exists.
- When writing features, the subtype attribute must contain the code (which is stored as an integer by Geodatabase).
- For detailed information, please see the ESRI Geodatabase chapter in the *FME Readers and Writers* manual.

Managing Attribute Connections

Quickly connect attributes to an output feature type – can be faster than dragging connections.

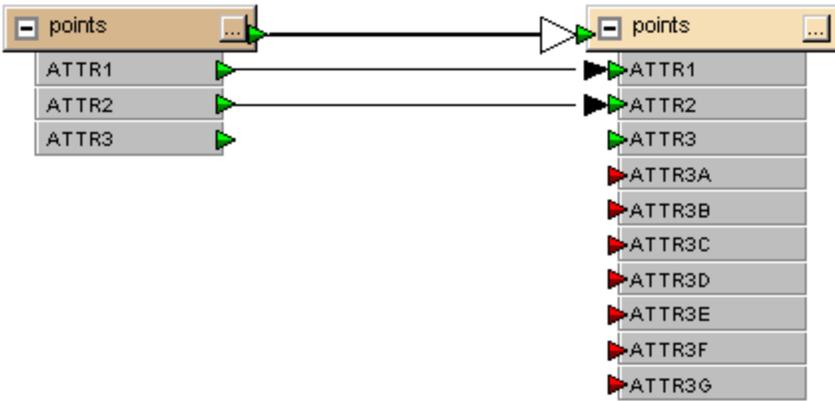
The Attribute Connections window lets you quickly connect one input attribute to many output attributes, without having to manually connect them. You can also easily disconnect numerous connections.

1. From the **View** menu, select **Windows – Attribute Connections**. The Attribute Connections window appears.
2. Follow the instructions outlined in the example below.

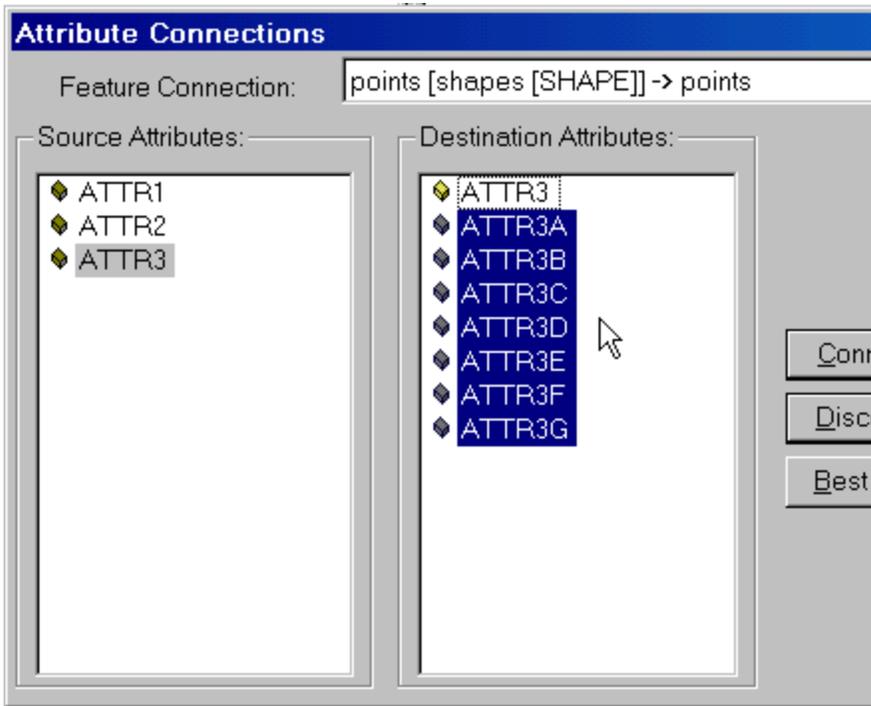
Tip: “Undock” the window by holding down the Ctrl key and dragging the left side of the window out of the docking area (this makes it “float”). If you keep pressing the Ctrl key, then it never will redock no matter where you place it. If you don’t press the Ctrl key while you’re dragging it, it will redock anywhere in the Workbench interface.

Example

You have one input attribute that you want to connect to many destination attributes. Manually connecting each one would take some time.

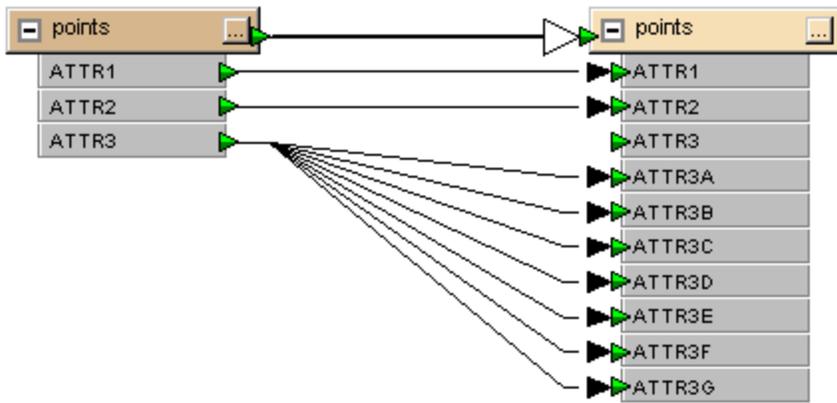


1. Select the input attribute in the Source Area (Shift+click to select multiple lines). Select the destination attributes in the Destinations area.



2. Click the Connect button.

All the attributes are connected in the graphical pane.



You can disconnect in the same way by pressing the Disconnect button.

Do you want to quickly delete connections from the workspace? Click directly on one connection and press the **Delete** key, or hold down the **Ctrl** key and click numerous connections. To quickly select many connections, hold down the left mouse key, drag the cursor in a rubberband around an area, release the mouse button and press Delete.

Adding and Copying Attributes

Creating a New Attribute

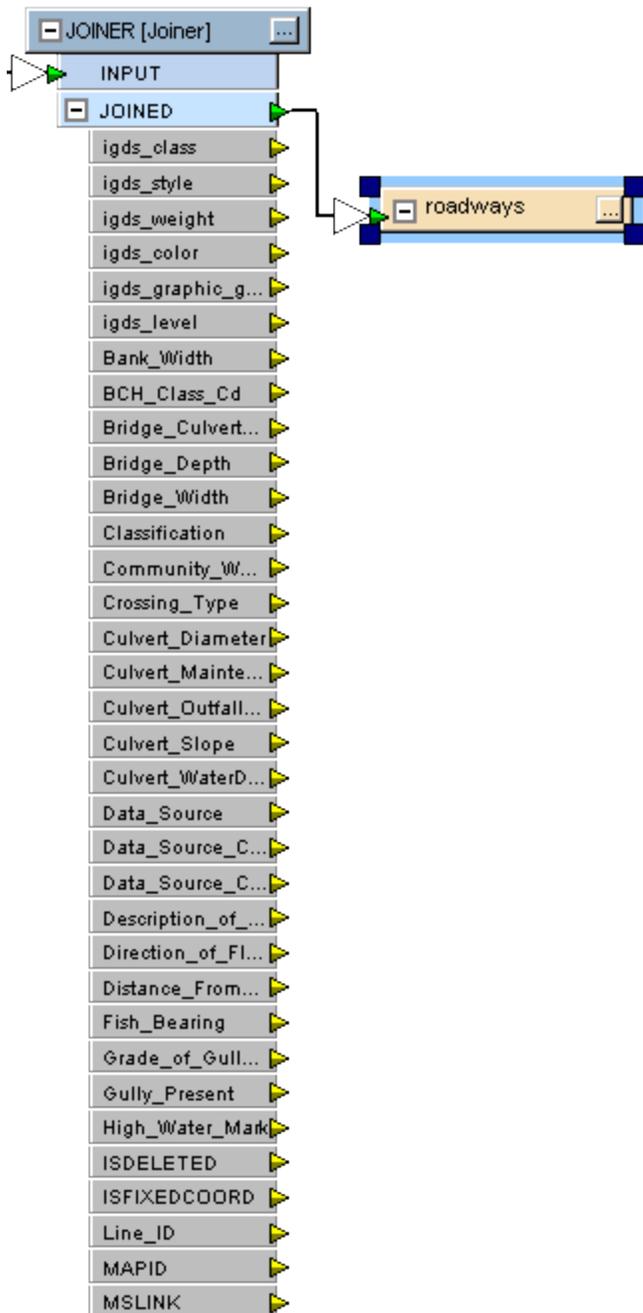
Add a new attribute to the feature type:

1. Select the Destination Feature Type, right-click and select Add Attribute. (You can also click the Properties button and the User Attributes tab.)
2. Enter the name of the attribute and use the tab key to move through the Data Type, Width and Decimal fields. You can enter or change the default values provided in these fields, or continue to enter more attribute names.
3. Click OK.
4. Connect the attribute with its source.

To quickly rename an attribute without reopening the dialog, float the cursor over an attribute name (you will see a highlight), right-click and select Rename.

Copying Attributes

Copy attributes from another feature type or a transformer. This is a useful function, if, for example, you have [many attributes to duplicate](#):



1. Select the Destination Feature Type, right-click and select either Copy Attributes From Feature Type... or Copy Attributes From Transformer.
2. In the dialog that appears, select either a feature type or a transformer from the pull-down list.
3. All the attributes will be copied to the selected feature type.

Tip: Use the Attribute Connection window to quickly connect the attributes. To quickly remove some of the attributes, float the cursor over an attribute name (but name sure you don't click the left mouse button). A background highlight will appear behind the attribute. Press the Delete key. The highlight will move down the list so you can quickly remove many attributes.

Managing Feature Type Connections

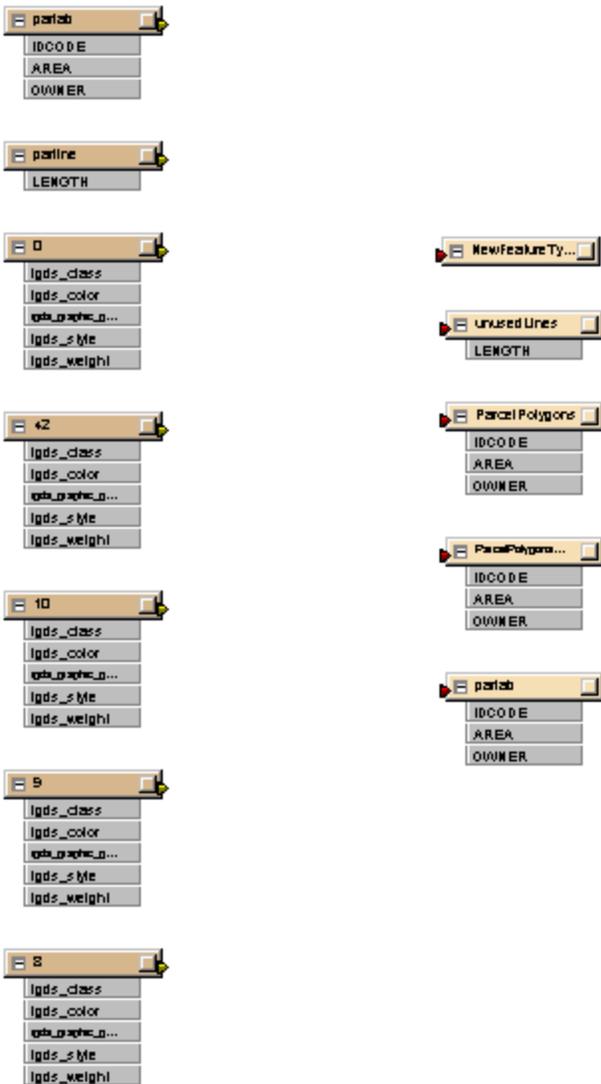
The Feature Connections window lets you quickly connect numerous transformers to a single input, without having to manually connect them. You can also easily disconnect numerous connections. This feature is good for mass connections and disconnections – for example, if you have many input feature types and you want to put them all to a single output feature type

1. Select **View > Windows > Feature Connections**. The Feature Connections window appears.
2. Follow the instructions outlined in the example below.

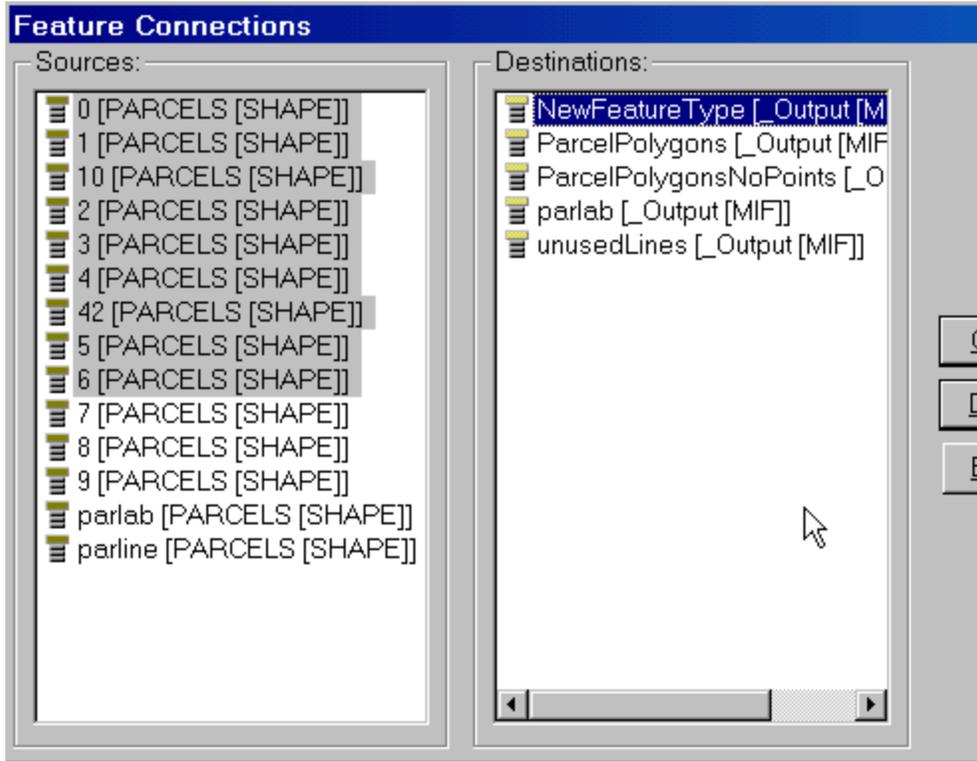
Tip: “Undock” the window by holding down the Ctrl key and dragging the left side of the window out of the docking area (this makes it “float”). If you keep pressing the Ctrl key, then it never will redock no matter where you place it. If you don’t press the Ctrl key while you’re dragging it, it will redock anywhere in the Workbench interface.

Example

You have many input feature types and you want to connect them to one output feature type. Manually connecting each one would take some time:

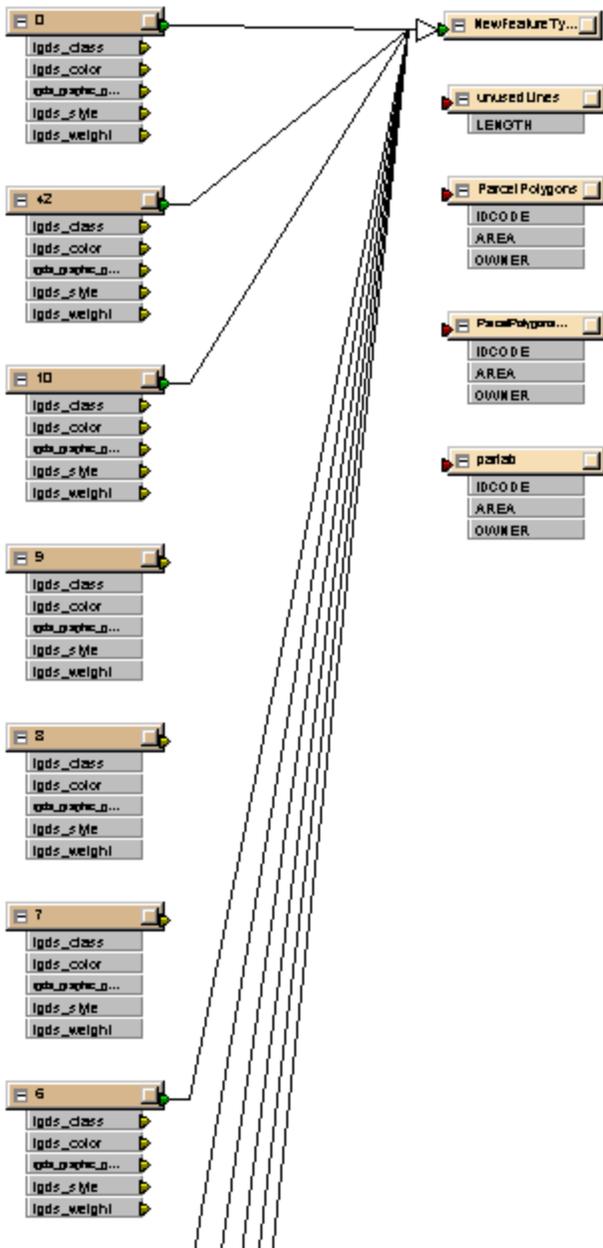


Select the input feature types in the Source Area (Shift+click to select multiple lines). Select the destination feature type in the Destinations area.



Click the Connect button.

All the feature types are connected in the workspace.



You can disconnect in the same way by pressing the Disconnect button.

Do you want to quickly delete connections from the workspace? Click directly on one connection and press the Delete key, or hold down the Ctrl key and click numerous connections. To quickly select many connections, hold down the left mouse key, drag the cursor in a rubberband around an area, release the mouse button and press Delete.

Merging Feature Types

This feature is useful if you have files that contain numerous layers (such as DXF files), you have similar source schemas, and you want to merge all the similar feature types for further processing, based on a matching pattern.

For directory-based formats, you can also use merge a directory of files. For example, if you had a directory full of Shapefiles like PIPE3.SHP, PIPE5.SHP, PIPE2293.SHP, you could merge them all together.

Note: You cannot use this feature with databases (except for file-based databases, e.g., MDB).

Deciding Whether to Import or Merge Feature Types

The two methods of handling multiple feature types have their own advantages and disadvantages.

The Import Feature Type tool has the advantage of keeping each feature type separate within the workspace. It therefore provides individual control over features types and how they are processed. The disadvantage to this method is that all feature types must be known and defined before the translation takes place.

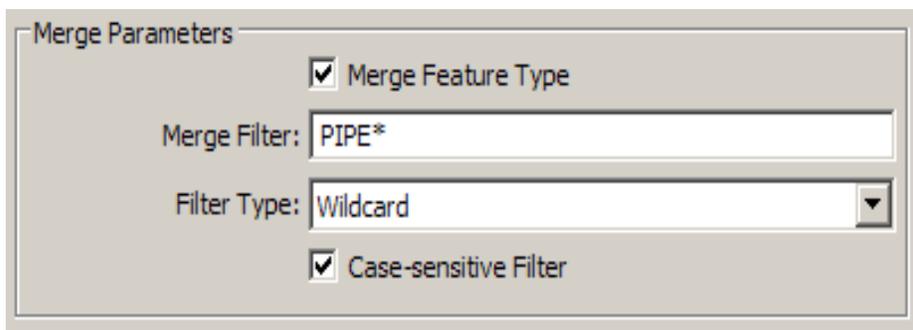
Although it is not a hard and fast rule, generally the Import Feature Types method is used on file-based datasets such as MicroStation DGN or AutoCAD DXF formats.

The Merge Feature Type tool does not need to be aware of all of the feature types beforehand. By setting the filter to '*' all feature types in the source data will be processed, whether or not they were known about. The disadvantage, however, is that routing all data through a single feature type in the workspace causes a loss of individual control over how each feature type is processed.

Again it is not a fixed rule, but Merge Feature Type is generally used on directory-based datasets such as Shape to ensure all feature types in a dataset are processed.

Merge Settings

1. Click Feature Type Properties  on an input feature type.
2. The Feature Type Properties dialog appears.
3. In the General tab under Merge Settings, check Merge feature type.
4. Specify a Merge filter and a Filter type. The filter type can also be an Exact Match or a Regular Expression.



Merge Parameters

Merge Feature Type

Merge Filter: PIPE*

Filter Type: Wildcard

Case-sensitive Filter

5. Click OK to apply the properties to the feature type from which you initiated the Properties dialog.
Click Apply to All to apply these properties to all input feature types (for example, if you add a bunch of attributes to one feature type, and you want other feature types to look the same).

When you run the translation, this pattern will be used to match against any incoming input feature types Matching input feature types will have their feature type "changed" to whatever the "known" feature type is.

Note: If a feature type matches more than one pattern, it will merge according to the first match.

For example, if you have a "PIPE145" feature type in your workspace as an input feature type, and you know there are many other feature types that have some sort of PIPE designation, you would click the Properties button, check the Merge Feature Type checkbox, and then enter a pattern that any incoming feature type would have to match in order to be considered the same as PIPE145. So, you could enter:

PIPE*

as shown in the example above. You can also uncheck the Case-sensitive filter so that even if an incoming feature type is "pipe" or "Pipe", it will be matched.

Regular Expressions

A regular expression describes strings of characters. It's a pattern that matches certain strings and doesn't match others.

Different Types of Regular Expression

There are two types of regular expression (RE), as defined by POSIX:

- extended REs (EREs) are roughly those of the traditional egrep,
- basic REs (BREs) are roughly those of the traditional ed.

This implementation adds a third type – advanced REs (AREs) – which are basically EREs with some significant extensions.

This [topic](#) primarily describes AREs. BREs mostly exist for backward compatibility in some old programs; they will be discussed at the end. POSIX EREs are almost an exact subset of AREs. Features of AREs that are not present in EREs will be indicated.

Tcl regular expressions are implemented using the package written by Henry Spencer, based on the 1003.2 spec and some (not quite all) of the Perl5 extensions. Much of the description of regular expressions is copied verbatim from his manual entry.

Regular Expression Syntax

An ARE is one or more branches, separated by |, matching anything that matches any of the branches.

A branch is zero or more constraints or quantified atoms, concatenated. It matches a match for the first, followed by a match for the second, etc; an empty branch matches the empty string.

A **quantified atom** is an atom possibly followed by a single quantifier. Without a quantifier, it matches a match for the atom. The quantifiers, and what a so-quantified atom matches, are:

*	a sequence of 0 or more matches of the atom
+	a sequence of 1 or more matches of the atom
?	a sequence of 0 or 1 matches of the atom
{m}	a sequence of exactly m matches of the atom
{m,}	a sequence of m or more matches of the atom
{m,n}	a sequence of m through n (inclusive) matches of the atom; m may not exceed n
*? +? ?? {m}? {m,}? {m,n}?	non-greedy quantifiers, which match the same possibilities, but prefer the smallest number rather than the largest number of matches (see Matching)

The forms using { and } are known as bounds. The numbers m and n are unsigned decimal integers with permissible values from 0 to 255 inclusive.

An **atom** is one of:

(re) (where re is any regular expression)	matches a match for re, with the match noted for possible reporting
(?:re)	as previous, but does no reporting (a "non-capturing" set of parentheses)
()	matches an empty string, noted for possible reporting

(?:)	matches an empty string, without reporting
[chars]	a bracket expression, matching any one of the chars (see Bracket Expressions for more detail)
.	matches any single character
\k (where k is a non-alphanumeric character)	matches that character taken as an ordinary character, e.g. \\ matches a backslash character
\c where c is alphanumeric (possibly followed by other characters)	an escape (AREs only), see Escapes
{	when followed by a character other than a digit, matches the left-brace character '{'; when followed by a digit, it is the beginning of a bound (see above)
x	where x is a single character with no other significance, matches that character.

A **constraint** matches an empty string when specific conditions are met. A constraint may not be followed by a quantifier. The simple constraints are as follows; some more constraints are described in Escapes.

^	matches at the beginning of a line
\$	matches at the end of a line
(?=re)	positive lookahead (AREs only), matches at any point where a substring matching re begins
(?!re)	negative lookahead (AREs only), matches at any point where no substring matching re begins

Notes:

- The lookahead constraints may not contain back references, and all parentheses within them are considered non-capturing.
- An RE may not end with \.

Basic Regular Expressions

BREs differ from EREs in several respects. `|', `+', and `?' are ordinary characters and there is no equivalent for their functionality. The delimiters for bounds are `{` and `}`, with `{` and `}` by themselves ordinary characters. The parentheses for nested subexpressions are `(' and `)', with `(' and `)' by themselves ordinary characters. `^' is an ordinary character except at the beginning of the RE or the beginning of a parenthesized subexpression, `\$' is an ordinary character except at the end of the RE or the end of a parenthesized subexpression, and `*' is an ordinary character if it appears at the beginning of the RE or the beginning of a parenthesized subexpression (after a possible leading `^'). Finally, single-digit back references are available, and `<' and `>' are synonyms for `[:<:]' and `[:>:]' respectively; no other escapes are available.

Bracket Expressions

A bracket expression is a list of characters enclosed in `[]`. It normally matches any single character from the list (but see below). If the list begins with `^`, it matches any single character (but see below) not from the rest of the list.

If two characters in the list are separated by `-`, this is shorthand for the full range of characters between those two (inclusive) in the collating sequence, e.g. `[0-9]` in ASCII matches any decimal digit. Two ranges may not share an endpoint, so for example, `a-c-e` is illegal. Ranges are very collating-sequence-dependent, and portable programs should avoid relying on them.

To include a literal `]` or `-` in the list, the simplest method is to enclose it in `[. and .]` to make it a collating element (see below). Alternatively, make it the first character (following a possible `^`), or (AREs only) precede it with `\`. Alternatively, for `-`, make it the last character, or the second endpoint of a range. To use a literal `-` as the first endpoint of a range, make it a collating element or (AREs only) precede it with `\`. With the exception of these, some combinations using `[` (see next paragraphs), and escapes, all other special characters lose their special significance within a bracket expression.

Within a bracket expression, a collating element (a character, a multi-character sequence that collates as if it were a single character, or a collating-sequence name for either) enclosed in `[. and .]` stands for the sequence of characters of that collating element. The sequence is a single element of the bracket expression's list. A bracket expression in a locale that has multi-character collating elements can thus match more than one character. | So (insidiously), a bracket expression that starts with `^ |` can match multi-character collating elements even if none | of them appear in the bracket expression! (Note: Tcl currently has no multi-character collating elements. This information is only for illustration.)

For example, assume the collating sequence includes a `ch |` multi-character collating element. Then the RE `[[.ch.]]*c |` (zero or more `ch`'s followed by `c`) matches the first five | characters of ``chchcc'`. Also, the RE `[^c]b` matches all of | ``chb'` (because `[^c]` matches the multi-character `ch`).

Within a bracket expression, a collating element enclosed in `[= and =]` is an equivalence class, standing for the sequences of characters of all collating elements equivalent to that one, including itself. (If there are no other equivalent collating elements, the treatment is as if the enclosing delimiters were ``[.' and `.]'`.) For example, if `o` and `^` are the members of an equivalence class, then `[[=o=]]`, `[[=^=]]`, and `[o^]` are all synonymous. An equivalence class may not be an endpoint of a range. (Note: Tcl currently implements only the Unicode | locale. It doesn't define any equivalence classes. The | examples above are just illustrations.)

Within a bracket expression, the name of a character class enclosed in `[: and :]` stands for the list of all characters (not all collating elements!) belonging to that class. Standard character classes are:

- alpha A letter – uppercase letter
- lower A – lowercase letter
- digit – decimal digit
- xdigit – hexadecimal digit
- alnum – alphanumeric (letter or digit)
- print – An alphanumeric (same as alnum)
- blank – space or tab character
- space – character producing white space in displayed text
- punct – punctuation character
- graph – character with a visible representation
- cntrl – control character

A locale may provide others. (Note that the current Tcl implementation has only one locale: the Unicode locale.) A character class may not be used as an endpoint of a range.

There are two special cases of bracket expressions: the bracket expressions `[[:<:]]` and `[[:>:]]` are constraints, matching empty strings at the beginning and end of a word respectively. A word is defined as a sequence of word characters that is neither preceded nor followed by word characters. A word character is an alnum character or an underscore (`_`). These special bracket expressions are deprecated; users of AREs should use constraint escapes instead (see Escapes).

Escapes

Escapes (AREs only), which begin with a `\` followed by an alphanumeric character, come in several varieties: character entry, class shorthands, constraint escapes, and back references. A `\` followed by an alphanumeric character but not constituting a valid escape is illegal in AREs. In EREs, there are no escapes: outside a bracket expression, a `\` followed by an alphanumeric character merely stands for that character as an ordinary character, and inside a bracket expression, `\` is an ordinary character. (The latter is the one actual incompatibility between EREs and AREs.)

Character-entry escapes (AREs only) exist to make it easier to specify non-printing and otherwise inconvenient characters in REs:

- `\a` alert (bell) character, as in C
- `\b` backspace, as in C
- `\B` synonym for `\` to help reduce backslash doubling in some applications where there are multiple levels of backslash processing
- `\cX` (where `X` is any character) the character whose low- order 5 bits are the same as those of `X`, and whose other bits are all zero
- `\e` the character whose collating-sequence name is ``ESC'`, or failing that, the character with octal value 033

`\f` formfeed, as in C

`\n` newline, as in C

`\r` carriage return, as in C

`\t` horizontal tab, as in C

`\uwxyz` (where `wxyz` is exactly four hexadecimal digits) the Unicode character U+`wxyz` in the local byte ordering

`\Ustuvwxyz` (where `stuvwxyz` is exactly eight hexadecimal digits) reserved for a somewhat-hypothetical Unicode extension to 32 bits

`\v` vertical tab, as in C are all available.

`\xhhh` (where `hhh` is any sequence of hexadecimal digits) the character whose hexadecimal value is 0xhhh (a single character no matter how many hexadecimal digits are used).

`\0` the character whose value is 0

`\xy` (where `xy` is exactly two octal digits, and is not a back reference (see below)) the character whose octal value is 0xy

`\xyz` (where `xyz` is exactly three octal digits, and is not a back reference (see below)) the character whose octal value is 0xyz

Hexadecimal digits are ``0'-`9'`, ``a'-`f'`, and ``A'-`F'`. Octal digits are ``0'-`7'`.

The character-entry escapes are always taken as ordinary characters. For example, `\135` is `]` in ASCII, but `\135` does not terminate a bracket expression. Beware, however, that some applications (e.g., C compilers) interpret such sequences themselves before the regular-expression package gets to see them, which may require doubling (quadrupling, etc.) the ``\'`.

Class-shorthand escapes (AREs only) provide shorthands for certain commonly-used character classes:

`\d` `[[:digit:]]`

`\s` `[[:space:]]`

`\w` `[[:alnum:]]` (note underscore)

`\D` `[^[:digit:]]`

`\S` `[^[:space:]]`

`\W` `[^[:alnum:]]` (note underscore)

Within bracket expressions, ``\d'`, ``\s'`, and ``\w'` lose their outer brackets, and ``\D'`, ``\S'`, and ``\W'` are illegal. (So, for example, `[a-c\d]` is equivalent to `[a- | c[:digit:]]`. Also, `[a-c\D]`, which is equivalent to `[a- | c[^[:digit:]]]`, is illegal.)

A constraint escape (AREs only) is a constraint, matching the empty string if specific conditions are met, written as an escape:

`\A` matches only at the beginning of the string (see `MATCHING`, below, for how this differs from ``^'`)

`\m` matches only at the beginning of a word

`\M` matches only at the end of a word

`\y` matches only at the beginning or end of a word

`\Y` matches only at a point that is not the beginning or end of a word

`\Z` matches only at the end of the string (see `MATCHING`, below, for how this differs from ``$'`)

`\m` (where `m` is a nonzero digit) a back reference, see below

`\mnn` (where `m` is a nonzero digit, and `nn` is some more digits, and the decimal value `mnn` is not greater than the number of closing capturing parentheses seen so far) a back reference, see below

A word is defined as in the specification of `[[:<:]]` and `[[:>:]]` above. Constraint escapes are illegal within bracket expressions.

A back reference (AREs only) matches the same string matched by the parenthesized subexpression specified by the number, so that (e.g.) `([bc])\1` matches `bb` or `cc` but not ``bc'`. The subexpression must entirely precede the back reference in the RE. Subexpressions are numbered in the order of their leading parentheses. Non-capturing parentheses do not define subexpressions.

There is an inherent historical ambiguity between octal character-entry escapes and back references, which is resolved by heuristics, as hinted at above. A leading zero always indicates an octal escape. A single non-zero digit, not followed by another digit, is always taken as a back reference. A multi-digit sequence not starting with a zero is taken as a back reference if it comes after a suitable subexpression (i.e. the number is in the legal range for a back reference), and otherwise is taken as octal.

Limits and Compatibility

No particular limit is imposed on the length of REs. Programs intended to be highly portable should not employ REs longer than 256 bytes, as a POSIX-compliant implementation can refuse to accept such REs.

The only feature of AREs that is actually incompatible with POSIX EREs is that `\` does not lose its special significance inside bracket expressions. All other ARE features use syntax which is illegal or has undefined or unspecified effects in POSIX EREs; the `***` syntax of directors likewise is outside the POSIX syntax for both BREs and EREs.

Many of the ARE extensions are borrowed from Perl, but some have been changed to clean them up, and a few Perl extensions are not present. Incompatibilities of note include ``\b'`, ``\B'`, the lack of special treatment for a trailing newline, the addition of complemented bracket expressions to the things affected by newline-sensitive matching, the restrictions on parentheses and back references in lookahead constraints, and the longest/shortest-match (rather than first-match) matching semantics.

The matching rules for REs containing both normal and non-greedy quantifiers have changed since early beta-test versions of this package. (The new rules are much simpler and cleaner, but don't work as hard at guessing the user's real intentions.)

Henry Spencer's original 1986 `regexp` package, still in widespread use (e.g., in pre-8.1 releases of Tcl), implemented an early version of today's EREs. There are four incompatibilities between `regexp`'s near-EREs (``RREs'` for short) and AREs. In roughly increasing order of significance:

In AREs, `\` followed by an alphanumeric character is either an escape or an error, while in RREs, it was just another way of writing the alphanumeric. This should not be a problem because there was no reason to write such a sequence in RREs.

`{` followed by a digit in an ARE is the beginning of a bound, while in RREs, `{` was always an ordinary character. Such sequences should be rare, and will often result in an error because following characters will not look like a valid bound.

In AREs, `\` remains a special character within `[]`, so a literal `\` within `[]` must be written `\\`. `\\` also gives a literal `\` within `[]` in RREs, but only truly paranoid programmers routinely doubled the backslash.

AREs report the longest/shortest match for the RE, rather than the first found in a specified search order. This may affect some RREs which were written in the expectation that the first match would be reported. (The careful crafting of RREs to optimize the search order for fast matching is obsolete (AREs examine all possible matches in parallel, and their performance is largely insensitive to their complexity) but cases where the search order was exploited to deliberately find a match which was not the longest/shortest will need rewriting.)

Matching

In the event that an RE could match more than one substring of a given string, the RE matches the one starting earliest in the string. If the RE could match more than one substring starting at that point, its choice is determined by its preference: either the longest substring, or the shortest.

Most atoms, and all constraints, have no preference. A parenthesized RE has the same preference (possibly none) as the RE. A quantified atom with quantifier `{m}` or `{m}?` has the same preference (possibly none) as the atom itself. A quantified atom with other normal quantifiers (including `{m,n}` with `m` equal to `n`) prefers longest match. A quantified atom with other non-greedy quantifiers (including `{m,n}?` with `m` equal to `n`) prefers shortest match. A branch has the same preference as the first quantified atom in it which has a preference. An RE consisting of two or more branches connected by the `|` operator prefers longest match.

Subject to the constraints imposed by the rules for matching the whole RE, subexpressions also match the longest or shortest possible substrings, based on their preferences, with subexpressions starting earlier in the RE taking priority over ones starting later. Note that outer subexpressions thus take priority over their component subexpressions.

Note that the quantifiers `{1,1}` and `{1,1}?` can be used to force longest and shortest preference, respectively, on a subexpression or a whole RE.

Match lengths are measured in characters, not collating elements. An empty string is considered longer than no match at all. For example, `bb*` matches the three middle characters of ``abbbc'`, `(week|wee)(night|knights)` matches all ten characters of ``weeknights'`, when `(.*)*` is matched against `abc` the parenthesized subexpression matches all three characters, and when `(a*)*` is matched against `bc` both the whole RE and the parenthesized subexpression match an empty string.

If case-independent matching is specified, the effect is much as if all case distinctions had vanished from the alphabet. When an alphabetic character that exists in multiple cases appears as an ordinary character outside a bracket expression, it is effectively transformed into a bracket expression containing both cases, so that `x` becomes `[xX]`. When it appears inside a bracket expression, all case counterparts of it are added to the bracket expression, so that `[x]` becomes `[xX]` and `[^x]` becomes `[^xX]`.

If newline-sensitive matching is specified, `.` and bracket expressions using `^` will never match the newline character (so that matches will never cross newlines unless the RE explicitly arranges it) and `^` and `$` will match the empty string after and before a newline respectively, in addition to matching at beginning and end of string respectively. ARE `\A` and `\Z` continue to match beginning or end of string only.

If partial newline-sensitive matching is specified, this affects `.` and bracket expressions as with newline-sensitive matching, but not `^` and `$`.

If inverse partial newline-sensitive matching is specified, this affects `^` and `$` as with newline-sensitive matching, but not `.` and bracket expressions. This isn't very useful but is provided for symmetry.

Metasyntax

In addition to the main syntax described above, there are some special forms and miscellaneous syntactic facilities available.

Normally the flavor of RE being used is specified by application-dependent means. However, this can be overridden by a director. If an RE of any flavor begins with ``***:'`, the rest of the RE is an ARE. If an RE of any flavor begins with ``***='`, the rest of the RE is taken to be a literal string, with all characters considered ordinary characters.

An ARE may begin with embedded options: a sequence `(?xyz)` (where `xyz` is one or more alphabetic characters) specifies options affecting the rest of the RE. These supplement, and can override, any options specified by the application. The available option letters are:

- b rest of RE is a BRE
- c case-sensitive matching (usual default)
- e rest of RE is an ERE
- i case-insensitive matching (see Matching)
- m historical synonym for n
- n newline-sensitive matching (see Matching)
- p partial newline-sensitive matching (see Matching)
- q rest of RE is a literal ("quoted") string, all ordinary characters
- s non-newline-sensitive matching (usual default)
- t tight syntax (usual default; see below)
- w inverse partial newline-sensitive ("weird") matching (see Matching)
- x expanded syntax (see below)

Embedded options take effect at the `)` terminating the sequence. They are available only at the start of an ARE, and may not be used later within it.

In addition to the usual (tight) RE syntax, in which all characters are significant, there is an expanded syntax, available in all flavors of RE with the `-expanded` switch, or in AREs with the embedded `x` option. In the expanded syntax, white-space characters are ignored and all characters between a `#` and the following newline (or the end of the RE) are ignored, permitting paragraphing and commenting a complex RE. There are three exceptions to that basic rule:

- a white-space character or `#'` preceded by ``\`` is retained
- white space or `#'` within a bracket expression is retained
- white space and comments are illegal within multi-character symbols like the ARE ``(?:'` or the BRE ``\(``

Expanded-syntax white-space characters are blank, tab, newline, and any character that belongs to the space character class.

Finally, in an ARE, outside bracket expressions, the sequence ``(?#ttt)'` (where `ttt` is any text not containing a ```) is a comment, completely ignored. Again, this is not allowed between the characters of multi-character symbols like ``(?:'`. Such comments are more a historical artifact than a useful facility, and their use is deprecated; use the expanded syntax instead.

None of these metasyntax extensions is available if the application (or an initial `***=` director) has specified that the user's input be treated as a literal string rather than as an RE.

About Fanout

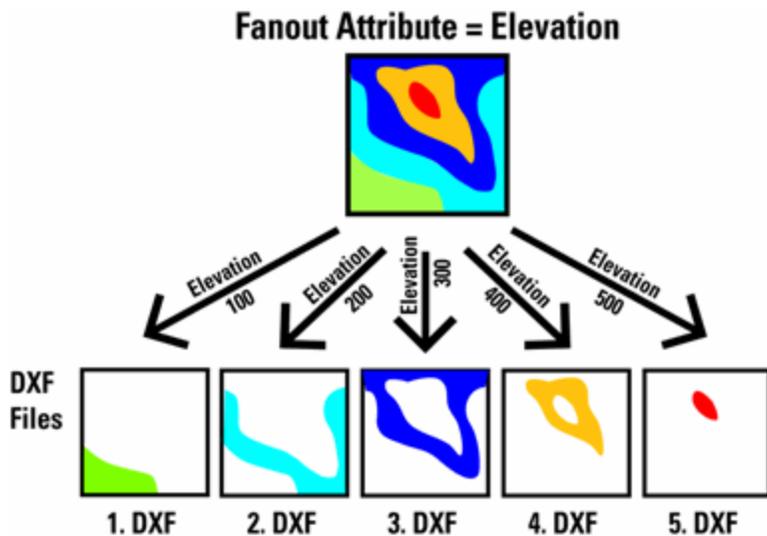
A fanout is a way within FME to split output data based upon the value of an attribute. The data is divided at the time of writing, rather than within the workspace itself.

There are two different kinds of fanout: feature type fanout and dataset fanout. You can also set both a feature type fanout AND a dataset fanout in the same workspace. The dataset fanout will produce a number of output datasets, each of which has a number of layers created by the feature type fanout.

You can set Dataset Fanout from the Navigator pane and you can set Feature Type Fanout from the Feature Type properties dialog.

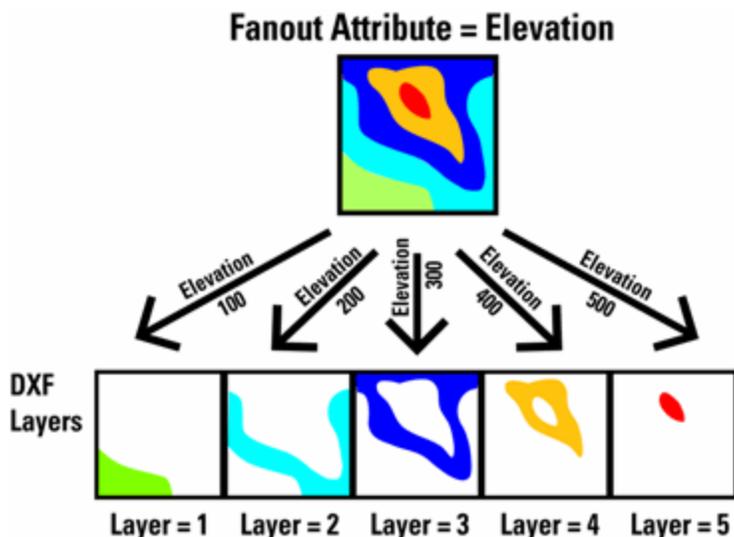
Dataset Fanout

A dataset fanout divides the data and writes a different dataset for each division. In this example, data is divided up into a number of datasets on the basis of an elevation attribute.



Feature Type Fanout

A feature type fanout writes a single dataset but divides the data into a number of layers/themes/feature types/objects/classes in that dataset. In this example, data is divided up into a number of layers on the basis of an elevation attribute.



Example

The difference between a dataset and feature type fanout is best explained through an example.

Given:

- A point dataset (e.g. trees in a city)
- A polygon dataset (e.g. a division of a city into neighborhoods)

The challenge is to produce a DXF file of the points, grouped into layers per neighborhood.

The polygons have an attribute containing the neighborhood code. After doing a simple PointOnAreaOverlay, this attribute has been transferred to the points. On the output feature type, we've set the fanout based upon this attribute. This will group the points by neighborhood and create a separate layer for each group. This is a feature type fanout.

A second challenge is to produce one DXF file per neighborhood. After disabling the feature type fanout, we move to the dataset (in the workspace panel) and again use the neighborhood attribute. This then generates a separate DXF file per neighborhood. This is a dataset fanout.

File Versus Folder Datasets

The behavior of fanouts changes based on whether you are working with a file-based format or a directory-based format. For instance, a DWG writer (file-based) will create new layers with a feature type fanout, and new drawings with a dataset fanout. A Shape writer (directory-based) will create new shapefiles on a feature-type fanout, and new directories on a dataset fanout.

Each format in the *FME Readers and Writers* manual contains a Quick Facts table, which specifies the format's Dataset Type.

Multi-Attribute Fanouts

Each fanout setting only allows a user to select a single attribute to fan out by. However, multi-attribute fanouts can be created by merging attributes together using the Concatenator transformer.

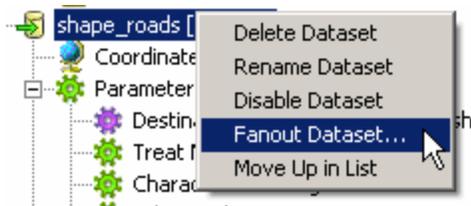
Setting Dataset Fanout Properties

See About Fanout for more information on dataset fanout.

See Setting Feature Type Fanout Properties for more information on setting feature type fanout.

Steps

1. Select the output dataset in the Navigation pane, right-click and choose Fanout Dataset:



2. An Edit Fanout Parameters dialog will appear.

- **Fanout Dataset:** Choose Yes
- **Fanout Directory:** Choose the destination directory for the files.
- **Fanout Prefix:** Enter a common prefix, if desired, for the output files.
- **Attribute to Fanout on:** Choose the attribute from the drop-down list. For example, you want to perform a dataset fanout in Shape format, and three features are output: one has a roadType equal to "gravel", and two have roadType equal to "paved". If you have a prefix "c:\data\", you will end up with feature types in two directories: c:\data\PAVED\ and c:\data\GRAVEL\. The output Shapefile names will be based on the output feature types specified in the workspace.
- **Fanout Suffix:** The suffix identifies the output file extension. If the suffix is blank, this means that nothing will be added to the end of the dataset. For instance, directories will consist of just the prefix and the fanout attribute value.

Combining Feature Type and Dataset Fanouts

You can combine Feature Type and Dataset Fanouts.

For example, if you are working with a directory-based format and you perform a feature type fanout on attribute A and a dataset fanout on attribute B, then you'll end up with multiple directories of files, with the filenames dependent on the values of attribute A.

Dataset Fanout and Temporary Disk Storage

Note that a Dataset Fanout can have a huge impact on temporary disk storage, since there is no guarantee that features arrive at the fanout in a single dataset group. Therefore, FME has to write all of the datasets to temporary storage then fan them out afterwards.

Feature Type Fanout has no similar issues, so you may get away with it if your destination format is directory-based.

Setting Feature Type Fanout Properties

See About Fanout for more information on dataset fanout.

See Setting Dataset Fanout Properties for more information on setting dataset fanout.

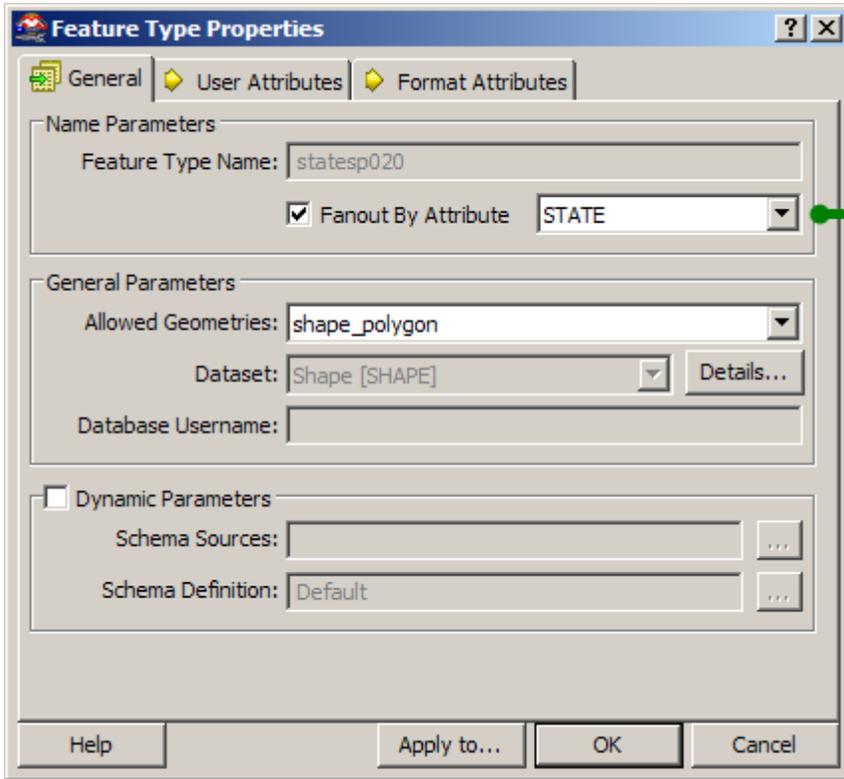
The fanout settings in the Feature Type Properties dialog allow you to use that specific feature type as a template for dynamically created feature types.

Feature type fanout means that many feature types will be created from one template feature type. How this actually appears as output depends entirely on the format. For example:

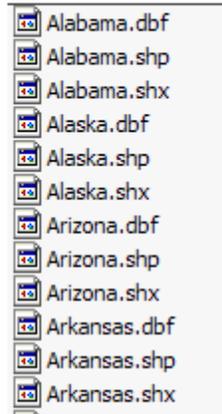
- AutoCAD is a file-based format, so each feature type created will be a different layer in the AutoCAD file.
- Shape is a directory-based format, so each feature type created is a new Shapefile.
- Oracle is a database format, so each new feature type is created as a new table in the database.

Steps

1. Click Feature Type properties button  on an destination feature type.
2. Check Fanout By Attribute.
3. Choose the Fanout Attribute from the drop-down menu. A new feature type will be created for each unique value of the fanout attribute, and all features with that value will be written to the resulting new feature type.
4. Click OK to apply the properties to the feature type from which you initiated the properties dialog. Click Apply to... if you want to apply these properties to all feature types in your output dataset.



Output files based on State name.



Combining Feature Type and Dataset Fanouts

You can combine Feature Type and Dataset Fanouts.

For example, if you are working with a directory-based format and you perform a feature type fanout on attribute A and a dataset fanout on attribute B, then you'll end up with multiple directories of files, with the filenames dependent on the values of attribute A.

Connecting to External Databases

Several transformers provide the ability to attach data from an external database to features as they flow from their source to their destination.

- [Joiner](#)

The Joiner transformer allows you to query a database to retrieve attributes associated with a feature. One or more feature attributes are used as keys to join to one or more columns in the database, and the values from the matching database table row (or rows) are added as feature attributes. 1 to 0 or 1, 1 to 1, and 1 to many relationships are supported.

The external database can be an Access MDB file, an ODBC connection, MySQL, PostGRES, PostGIS, Oracle 7, 8, 8i, or 9i, a dBase III (dbf) file, or a CSV file.

A wizard is used to set the database type, and connection parameters, and key field properties. If a 1 to many relationship is defined, a list name must be specified to hold the attribute data retrieved from the database. Each row retrieved is another entry in the list. Subsequent transformers can perform operations on the resulting list.

An optional prefix can be added to all the attributes retrieved from the database to allow repeated joins to the same database table to store their results into unique attributes.

- [SQLExecutor](#)

The SQL Executor transformer executes an arbitrary SQL statement against a database. If the SQL statement resulted in a row (or rows) being returned, the attributes from the row are added to the feature. This transformer can also be used in conjunction with a Creator transformer to perform database operations such as DROPPing or CREATEing tables at the beginning or end of a translation.

This transformer operates against MySQL, PostGRES, PostGIS, ODBC connections, or directly on Oracle databases.

- [ArcSDEQuerier](#)

The ArcSDEQuerier performs queries on an ArcSDE spatial database. The queries can have both a spatial as well as nonspatial component to them.

One query is issued to the ArcSDE database for each feature that enters the transformer. The results of the query, which could be many, many features, are then output. If the query mode is <delete>, the results of the query are deleted from ArcSDE before they are output from the transformer).

The query feature defines the geometry which will be used to define the spatial component of the query, unless the search method is SDE_NONE. In that case, only an attribute query as defined by the where clause will be executed.

- [OracleQuerier](#)

Performs spatial queries against an Oracle Spatial database. The queries can have both a spatial as well as nonspatial component to them. One query is issued to the Oracle database for each feature that enters the transformer. The results of the query are then output.

The query feature's bounding box is used to define the geometry that will be used to define the spatial component of the query.

- [Database Readers \(Oracle, Access MDB, ODBC, dBase, CSV\)](#)

The database reader can be used to read rows from nonspatial databases, such as Microsoft Access, ODBC, and Oracle.

Database reader feature type properties allow you to specify a complete and arbitrary SQL SELECT statement to be used to determine the features that will be returned. Click the Feature Type Parameters tab. The SELECT statement can involve joins to other tables in addition to filtering data from a single table. As well, the parameters tab allows for a WHERE clause to filter the features that are extracted.

If the attribute data is held in CSV or dBase III files, then the CSV or dBase readers can be used to directly read the data. No filtering via a WHERE clause is available in these cases.

Using a database reader in conjunction with another reader of feature data and the FeatureMerger transformer, can in some situations be more efficient than using the Joiner or SQLExecutor. This is because the Joiner and SQLExecutor make queries of the database for each feature that passes through them. However, if the relevant parts of the database can be identified ahead of time and read in a single query by a

database reader, and the resulting attribute features routed to a FeatureMerger transformer as SUPPLIER features, they can efficiently be paired up to the feature data acting as a REQUESTOR in the FeatureMerger.

Database Connection Issues

Several configuration issues can complicate connecting successfully to an external database.

ODBC Version

Oracle Connectivity

GeoMedia Writing

How FME logs database information

Databases are an important component of many datasets and the log file will help us determine both how good our database performance is, plus how well FME is interacting with the database.

The following example relates to a prefetch query carried out on an Oracle database:

```
2004-05-14 17:18:52| 476.1| 0.0|INFORM|Started SQL cache prefetch
```

```
2004-05-14 17:25:10| 476.2| 0.1|INFORM|Finished SQL cache
```

Although the time between when you issue the SQL prefetch and until it is complete is roughly 7 minutes, FME uses only 0.1 seconds of CPU time. The remaining time was spent by Oracle retrieving the data using the query it was given. To reduce that time, you would need to look at how the Oracle database is structured and how the query is written. Perhaps the field searched on isn't indexed? Maybe the query supplied isn't as efficient as it could be?

Check the log carefully to find out how much database-related time is spent outside of FME and see if you need to improve your database efficiency.

What are lists?

Collections of attributes may be grouped together in a single FME feature using an *attribute list*. An *attribute list element* is just like any other attribute except that it contains an integer index enclosed in braces `{}` as part of its name.

By convention, the first element of a list has an index of 0.

For example, if a feature has these attributes on it:

```
ids{0} 50
ids{1} 45
ids{2} 10
```

We would say that the feature has an *unstructured list* named "ids{" with three elements. In FME Workbench, the list would be shown as part of a transformer output using its *unqualified list name* `ids{}`.

If a feature has these attributes on it:

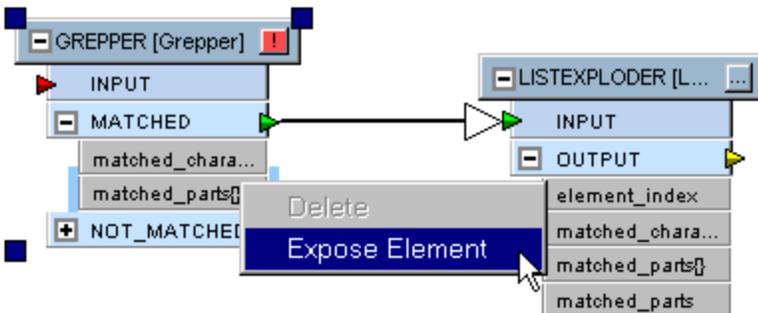
```
overlaps{0}.direction -1
overlaps{0}.line_id 50
overlaps{1}.direction 1
overlaps{1}.line_id 22
overlaps{2}.direction 1
overlaps{2}.line_id 40
overlaps{3}.direction -1
overlaps{3}.line_id 12
```

We would say that the feature has an *structured list* named "overlaps{" with four elements. In FME Workbench, the list would be shown as part of a transformer output by using two unqualified names – `overlaps{}`.direction and `overlaps{}`.line_id.

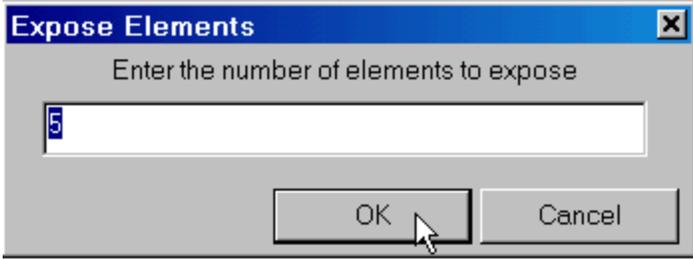
Unqualified list names remind us that there are a number of actual attribute values present on the feature. However, it is not possible to directly access values from an unqualified list name. Instead, these names can be used as parameters to the various transformers that operate on lists. As well, by right-clicking on any unqualified list, a number of its elements may be exposed and from then on be accessed directly. Such attributes are called *qualified list elements*.

See example:

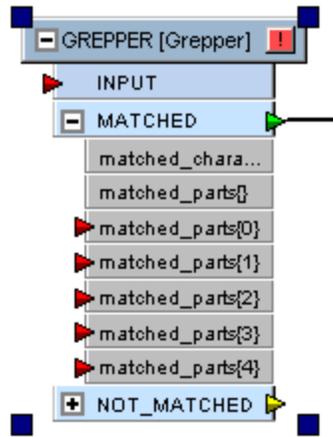
1



2



3



Exposing List Elements and Lists

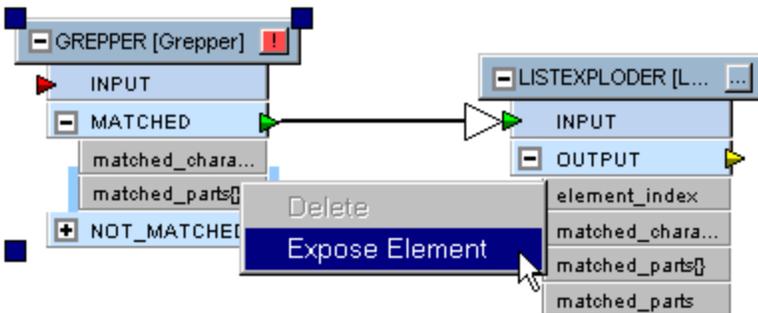
Some transformers and data formats use FME "lists" to hold multiple attribute values. List transformers allow you to work with these lists.

Some transformers and source formats/ destination formats use FME "lists" to hold values that can "repeat". If you want to get at the particular pieces of a list, you can expose some of its elements by using the ListExploder or ListIndexer transformers so that you can work with the values.

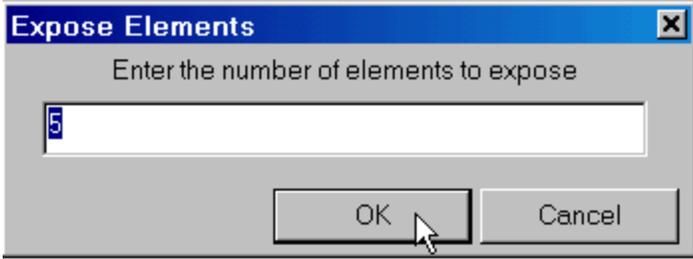
For example, the results of a Grepper transformer can create a list. You may want to work with a certain element of that list, and therefore you would "expose" some elements.

Example of the Grepper transformer returning a list attribute.

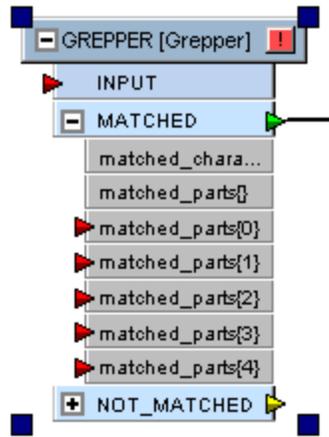
1



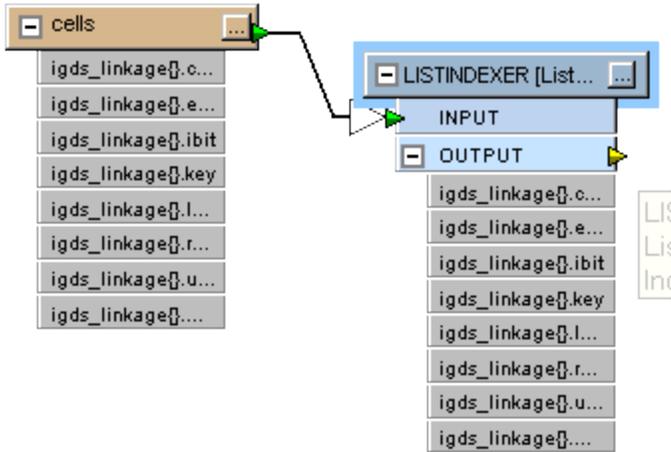
2



3



Example of the ListIndexer: the DGN format returns IGDS linkages in list form. You can use the list processing transformers to process this data.



Where do attribute lists come from?

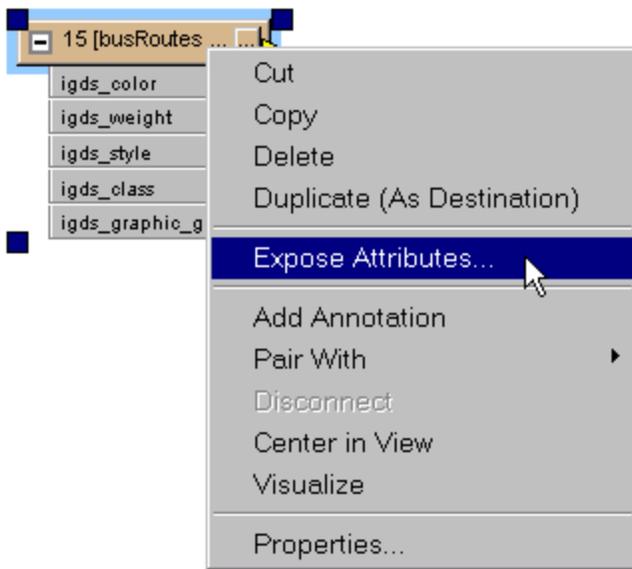
Attribute lists can be created directly by an FME reader, or they can be added to a feature by a transformer. Certain FME readers make use of attribute lists to represent recurring values read from the original data source, particularly when the number of such values is not known ahead of time. In these cases, the unqualified list can be exposed in FME Workbench, and optionally a number of elements can be explicitly qualified and then treated as normal attributes.

See an example.

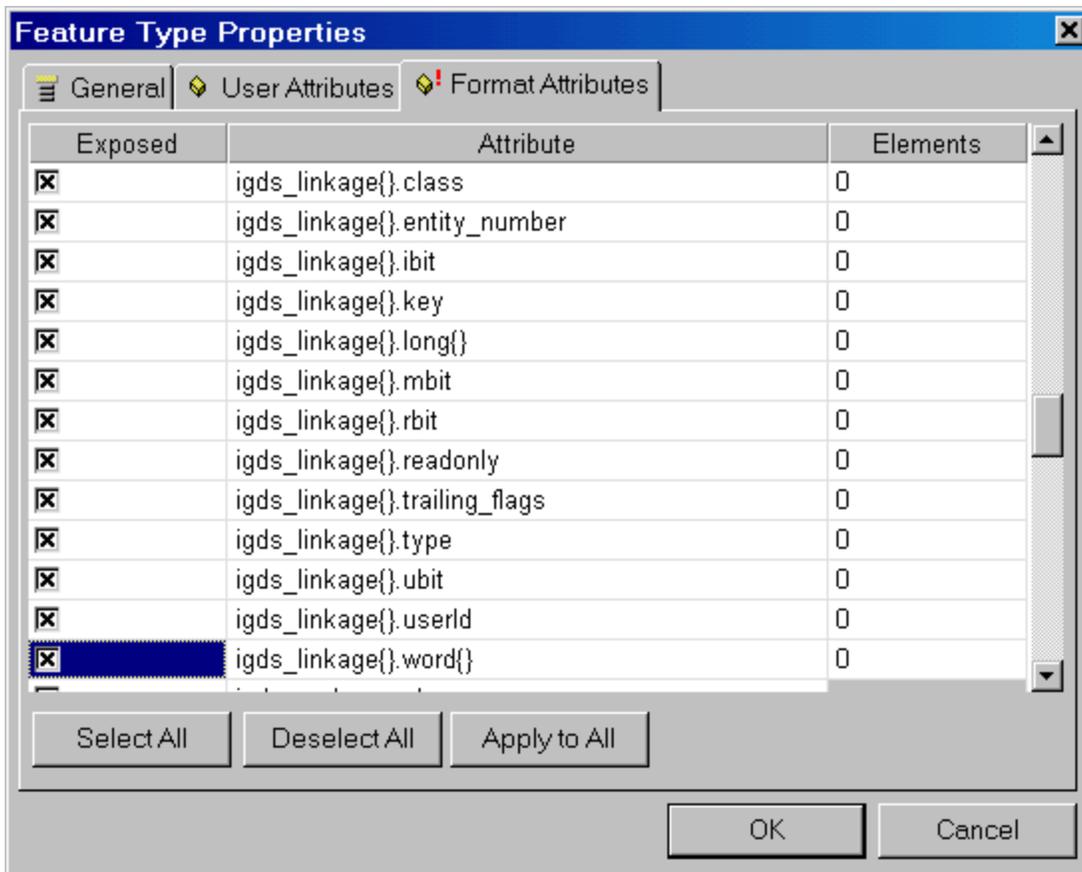
The DGN format returns IGDS linkages in list form. You can use the list processing transformers to process this data.

1





2



3



Many transformers create attribute lists as a result of their processing. For example, when the Joiner is used to fetch records related to a feature from a database, and the relation is 1 to many, each of the rows fetched are added to a structured list on the feature. When the Intersector notices overlapping line segments, it makes a list of all the attributes of the overlapping lines and adds this to the output segment. Overlay transformations can optionally create lists of the attributes of the original features that were involved in creating each resultant feature. Both the Aggregator and ListBuilder may make lists of the attributes of features that they combine. The lists that are created by transformers such as these can contain very important information to be used in later processing.

What can I do with attribute lists?

If you know which element of a list you want, you can expose it directly in FME Workbench and then treat it as you would any other attribute. Alternately, you can use the ListIndexer transformer to demote elements of a structured list into full attributes, and the index can be either a number you provide directly, or one that you extract from another attribute on the feature.

[View Example](#)

For example, if a feature has these attributes on it:

```
overlaps{0}.direction -1
overlaps{0}.line_id 50
overlaps{1}.direction 1
overlaps{1}.line_id 22
overlaps{2}.direction 1
overlaps{2}.line_id 40
overlaps{3}.direction -1
overlaps{3}.line_id 12
```

and it went through a ListIndexer with an index of 2, then the feature would emerge with these attributes:

```
direction 1
line_id 40
overlaps{0}.direction -1
overlaps{0}.line_id 50
overlaps{1}.direction 1
overlaps{1}.line_id 22
overlaps{2}.direction 1
overlaps{2}.line_id 40
overlaps{3}.direction -1
overlaps{3}.line_id 12
```

Note that the list itself was not changed, but the attributes at index 2 were demoted down from the list to become regular attributes.

Other List Operations

Other list operations available include searching a list (ListSearcher) for a particular value, counting the number of elements in a list (ListElementCounter), concatenating all the elements of a list into a single attribute (ListConcatenator), and exploding a list by creating a new feature for each element in the list and adding the demoted attributes to it (ListExploder).

The ListExploder is sometimes used to create individual features for each list element, so that they can be written to a relational database table. In this way, the 1 to many nature of the data model can be preserved by writing the original feature to one table, and the exploded list element features to another table. As long as there is a key field that can be used to rejoin them, the relationship is maintained. Furthermore, there is no requirement that the format used to store the list elements be the same as the format used to store the master feature—very often the exploded list element records are stored in a database table and the master feature is stored in a GIS or CAD system.

Another use of the ListExploder is to make copies of the feature for each list element, and then these copies may be sorted by some criteria (such as length or area), and lastly the elements run through a DuplicateRemover transformer to select only the largest or smallest feature that was originally an element of a list.

View an example problem solving exercise:

Using Lists to Solve Problems

Example Using Intersector and ListConcatenator

You have linear street centerlines and at each intersection point, you'd like to know which streets come together. The output should be a set of points, each with a single string attribute containing a comma-separated set of the street names.

You can solve this by setting up a workspace that routes all the street centerlines into an Intersector. Adjust the parameters of the Intersector to supply a list name, for example, `all_streets`.

Let's assume that the input street lines had an attribute called `NAME`. Now, among other things, the `NODE` output of the Intersector will have an unqualified list on it called `all_streets{}.NAME`. This list will hold the names of all the streets that intersect at each particular point (or `NODE`) that is output.

To turn the list of `NAME`s into a single string, you'd add a ListConcatenator transformer and run the `NODE` features into it. Then you'd set up the ListConcatenator's parameters so that it would put the contents of the `all_streets{}.NAME` list together, separated by commas, into the "result" attribute.

Lastly, you'd route the output of the ListConcatenator to an output file, and ensure that the "result" attribute was routed to an attribute in the output. After running the translation, you'd have the desired result.

Note that you could also access the individual street names by "exposing" some elements of your list (by right clicking on the attribute unqualified list name (in our example, `all_streets{}.NAME`) and saying "Expose Elements", and entering the number of elements to expose. You'd then have to do something with those elements in your translation. (The disadvantage of the this approach is that you need to know ahead of time how many list elements you want to work with -- so if 3 streets intersect at the same node and you only set yourself up to handle two, you'd have to do something special to handle that.)

How FME Supports Linear Referencing

Linear referencing uses measures to associate attributes or events to locations or portions of a linear feature.

FME previously supported writing of measures to Geodatabase, SDE, and Shape files via a special attribute containing a comma-separated list of measure values, corresponding to the coordinates of the feature. However, in order to further support writing of measures, we still had to come up with a set of measures to populate. Rather than pulling the measures from database tables and combining them into a CSV attribute before writing them out (like some applications do) there is now a new Transformer called the **MeasureGenerator**, which populates an attribute with a comma-separated list of the distances from the start of the line to each point. This can then be sent to the "shape_measures" attribute and it will be written out.

Two new "Snipper" transformers have been added:

- The **DistanceSnipper** shortens the geometry of a line feature by snipping off a specified amount from the ends. The amount to snip from the beginning and end of the line can be specified as either a measurement in ground units or a percentage of the line's entire length, starting from the first coordinate.
- The **VertexSnipper** shortens the geometry of a line feature by snipping off vertices from the ends. The vertices from the original line which are to form the first and last vertices of the resulting line are specified as a numeric index, with "0" being the first vertex of the line.

These two transformers can be used to transform data that was stored in an "LRS" way into a segmented view (if you have a table of "whole" geometries, and another event table that says where on a particular line things begin and end, you can do a join, and then a snip of some kind, and get out the segments).

- The **LengthToPointCalculator** calculates the length of a feature from its start until the closest spot to a point, and adds it as a new attribute. This can be used to create an LRS view from a segmented view, when combined in a clever way with the ArcFactory (LineJoiner) and ReferenceFactory (FeatureMerger).