

目录

FME 和数据格式	3
支持的格式.....	3
为什么有这么多种格式?.....	4
非空间格式.....	5
许可证和系统要求.....	5
Format Plug-Ins	5
重要的 FME 术语	6
读模块和写模块.....	6
数据集和要素类别.....	6
数据集的类型.....	7
结构关系.....	9
文件和文件夹形式的数据集.....	9
转换控制.....	11
工作空间设置.....	12
什么是工作空间设置?.....	12
基本的工作空间设置.....	12
高级工作空间设置.....	13
读取和写入参数.....	16
读取和写入参数.....	17
要素类参数.....	19
发布参数.....	21
什么是发布参数.....	21
怎样创建已发布的参数.....	21
怎样使用已发布的参数.....	22
格式化属性.....	23
什么是格式属性?	23
更多有关格式属性的例子.....	23

格式属性与 FME 属性	24
几何模型和格式属性.....	24
用格式化属性过滤.....	25
展开格式属性.....	25
用格式化属性转换.....	27
AttributeSetter 函数	27
AttributeCreator 函数.....	27
常量.....	28
格式属性的缺点.....	28
FME 读写手册	29
格式简介.....	29
Reader Directives	29
Writer Directives	30
要素表示法.....	30
高级格式控制.....	31
The Generic Writer.....	31
要读取的要素类.....	32
语义转换.....	33
FME 支持的几何属性	33
几何属性结构.....	34
FME 支持的属性	35
单元总结.....	37
从这一单元中，你应该学到些什么呢？	37
问与答.....	38

FME 和数据格式

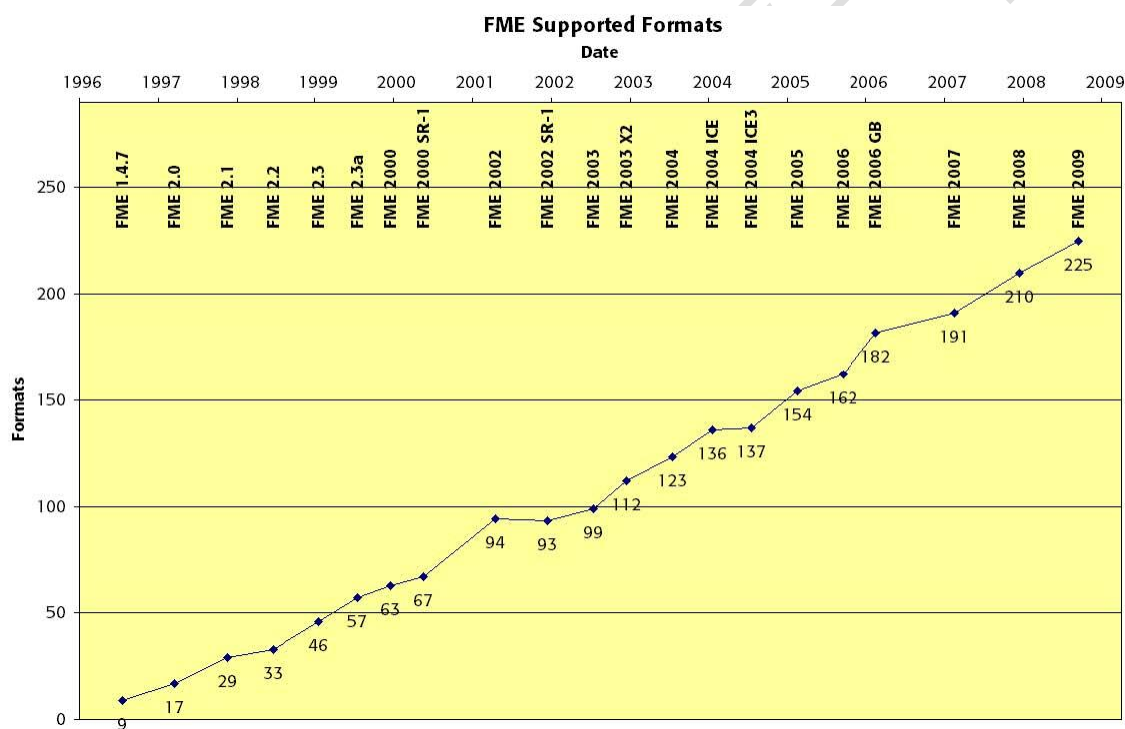


互操作性指的是，交换不同系统和应用程序的信息。FME 通过支持海量空间和非空间数据格式间的转换来实现互操作性。

支持的格式

FME 支持 200 多种格式的数据。

令人惊奇的是，在过去的十年内这种比例呈稳步上升趋势，并没显示出任何下降或停滞不前的趋势。



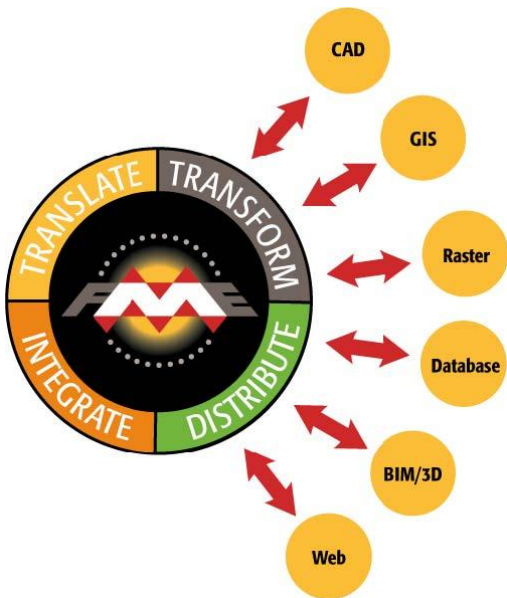
Mr Saif-Investor 说过...

“虽然说过去的表现并不能保证将来，但是在这个图标中，我没有看到任何下滑的趋势。”

FME 能够同时处理多种源格式和目标格式的数据，并且以一种完全语义的方式，我可以用最小的努力，将不同的数据集转化成我想要的格式。

为什么有这么多种格式？

现在有许多领域都要用到空间数据，所以才会有越来越多的格式。在 Safe Software 中，有时我们将这些格式称之为数据的“家族”，这些家族中的每个成员都有不同的数据，这些数据有它们自身的特性，与其它的数据的结构类型区分开来。



Type	Characteristics
CAD	支持对要素的高精度描述，传统上来讲很少或几乎没有底层智能
GIS	现实对象在空间中的表达，通常拥有底层智能
Raster	将空间数据表现为一系列栅格单元——通常应用到GIS或可视化展示中。
Database	将数据记录集中存储起来，通常使用事务的方式进行处理。
BIM/3D	有关建筑的数据，通常以数据模型为基础，在这个模型中，建筑被划分为多个组件
Web	任何形式的空间数据——矢量或栅格，被在线储存，并且通过网络分发这些数据

也可以添加上面提到的家族

Cartographic	对数据进行优化处理，突出某一个空间特性或内容
Transfer	特别处理某种格式，将格式供给变得标准化

当然最大的难处，还是在处理不同类型的数据时，保存这些数据的意思和内容，例如，将地图和数据库的数据结合成 GIS 格式的输出数据，FME 的读模块和写模块就能很好地做到这一点。

读取或写入

值得我们关注的是，并非 FME 支持的每一种格式都能够同时进行读取和写入。一些格式的数据仅仅支持读取（例如，ESRI ArcGIS mxd files），而一些格式仅仅支持写入（例如，SVG）。但是，大多数格式还是同时支持读取和写入的。

非空间格式

FME 支持大量的非空间格式，所以，FME 不仅仅能够处理空间要素的非空间属性，它也能处理非空间要素。

许可证和系统要求

FME 有许多不同的版本，FME 的版本是由许可证决定的，而不是所安装的产品决定的，所以 FME 的 CD-ROM 和下载版本都是相同的。

不同的 FME 版本仅仅在所支持的格式上有所不同，只有 FME Desktop Edition 有不同（更为受限的）的功能。通常，版本的名字是由它们支持的格式所命名，例如，FME Smallworld Edition 就特别支持 GE Smallworld 格式的数据集。

当只有在系统中安装适合的应用程序时，才能支持一些格式，例如，ESRI GeoDatabase: 因为 FME 使用 ArcObjects 组件对象来读取 GeoDatabase 的数据，所以就必须要安装 ArcGIS 并获得有效地许可证，以使 FME 支持该格式。

Format Plug-Ins

FME 需要支持一些非常专业的数据格式，这种情况下，就要花钱购买 plug-in，通常第三方供应商使用 FME Plug-In SDK 来开发特定 plug-in（插件）。



Ms Surveyor 说过...

“登录网站 www.safe.com/formats”，就可以找到 FME 支持格式的完整列表—包括支持它们的 FME 版本。

FME DESKTOP SUPPORTED FORMATS

Filter list by format name(s):
✕

Filter by file type:
☐ Vector
☐ Raster
☐ Non-Spatial
☒ All

Select which editions you would like to compare:
☐ All

☐ FME Base Edition
☒ FME Professional Edition
☒ FME ESRI Edition

☒ FME Intergraph Edition
☐ FME ArcGIS Data Interop Edition
☒ FME Oracle Edition

☒ FME Microsoft SQL Server Edition
☐ FME IBM DB2 Edition
☒ FME Smallworld Edition

R = Read Support
W = Write Support
\$ = Requires Extra-cost Plug-in
+ = Solution Assistance Recommended

Format	FME Professional Edition	FME ESRI Edition	FME Intergraph Edition	FME Microsoft SQL Server Edition	FME Oracle Edition	FME Smallworld Edition
1Spatial Internal Feature Format (IFF)	R / W	R / W	R / W	R / W	R / W	R / W
Additional Military Layers (AML)	R	R	R	R	R	R
Adobe Flash (SWF)	R / W	R / W	R / W	R / W	R / W	R / W
Adobe Illustrator EPS	R / W	R / W	R / W	R / W	R / W	R / W
Adobe PDF	R	W	W	W	W	W

重要的 FME 术语



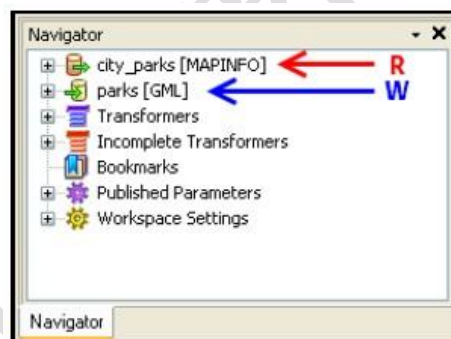
在 FME 会使用到大量特有的术语，我们有必要理解它们的意思。当我们对数据集进行批量处理，或是处理多个数据集时，了解这些属于术语就显得格外重要。

读模块和写模块

写模块或读模块是 FME 的一个术语，表示在转换过程中读取一个源数据集或写入一个目标数据集。

Navigator 窗口中出现的要么是写模块，要么就是读模块，右图例子中出现的就是其中一个。

读模块和写模块的图标如图所示：



但是，为什么读模块/写模块和数据集之间有区别呢？

这是因为许多 FME 转换任务都会处理多个数据集。

但是有时候，源数据集在格式和结构上是完全相同的，所以使用一个单一的读模块能更有效地对它进行处理。

有时候源数据集——虽然在格式上相同，但是可能结构不同，这就要求使用不同的读模块处理它。

数据集和要素类别

数据集

地理信息协会将数据集定义为“具有相同的主题数据的有组织集合”，这是对 FME 中的数据集的一个很好的描述。

在 FME 中，数据集指的是将参与转换的数据的集合。FME 可以从多个数据集中读取数据，或是将数据写入到多个数据集中，并且这些数据可以在相同的位置，或者也可以是分散于不同的位置。

要素类别

要素类别是 FME 的一个术语，描述数据集中可辨别的一个子集，因此，每个数据集可能含有大量不同的要素类别。

这个术语还有其它的常用叫法，例如，“图层 (Layer)”，“要素类 (feature class)”，“对象类 (object class)”

例如，一个空间数据集可能是关于“规划信息的，其中也包括“道路”，“物业”以及“学校”方面的信息。

在 FME 中，‘规划信息’就是数据集的名字，而“道路”，“物业”以及“学校”就是不同的要素类别。



不要将“要素类别”和“几何属性类”型弄混了，后者是有关点，线，多边形的术语。

数据集的类型

源数据集只是许多不同类型中的一种。数据集类型在数据储存和数据结构上进行区分。因为 FME 会用不同的方法处理每一种类型，所以有必要了解这种区别，并且知道你正要处理的数据是什么类型的。

数据集分为四种基本类型，如下：

类型	特征	举例
文件数据集	一个单独的文件	AutoCAD DXF
文件夹数据集	一个文件夹内的一系列文件	ESRI Shape
数据库数据集	一个数据库	Oracle Spatial
网络数据集	一个互联网站	Web Feature Service (WFS)

文件形式的数据集

文件形式的数据集就像你所听到的那样简单，它指的是在一个文件中保存数据集，例如，AutoCAD DXF：每个 DXF 文件都是一个单独的数据集。

文件形式数据集内的要素类

它通常用一些方法将数据进行分类，在 FME 中，这些层次就是不同的要素类。

例如，在 AutoCAD 中，数据就被定义为不同的层，DXF 中的每个层就表示 FME 中的一个要素类。DGN 格式的文件使用“Level”代替层的概念。

文件夹形式的数据集

这种逻辑关系可能让你有些困惑。在这种形式的数据集中，数据集是以文件夹或目录的形式来保存数据。

例如，“C:\FMEDData\Data\Properties”中，数据集的名字就是“Properties”。

ESRI Shape, MapInfo TAB 和 CSV format 都是相关的例子。

文件夹形式数据集内的要素类

数据被保存为一系列的文件。通常，不同名的文件就表示数据集内的一个要素类

例如，在下面的 MapInfo 数据结构中，数据集被叫做“EastA”。要素类则是“BoundaryArea”，“HydrographyLine”和“RoadLine”

C:\FMEDData\Data\DemoData\MultiDataset\EastA\BoundaryArea.mif

C:\FMEDData\Data\DemoData\MultiDataset\EastA\HydrographyLine.mif

C:\FMEDData\Data\DemoData\MultiDataset\EastA\RoadLine.mif



FME 用户通常会将所有的单独文件都当做数据集，这是错误的。当在文件中不能对数据进行再划分时，通常 FME 就会将这个文件当做文件夹形式数据内的一个要素类。

数据库/数据集

正如名字所描述的一样，数据库数据集指的是数据库中储存的一组数据。通常来说，每个不同的数据库就是一个不同的数据集（虽然严格上来说，数据库中的每个不同的用户/模式，应当被认为是一个不同的数据集）。

Oracle 数据库就是一个最显然的例子；数据集用同样的方式处理无论是空间，或是非空间数据。

数据库 数据集中的要素类

数据库中的每个表都被当做一个要素类。

例如，一个名为“资源”的数据库有“用户”，“交通”和“设备”三个表。“资源”就是数据集，“用户”，“交通”和“设备”则是要素类。

网站数据集

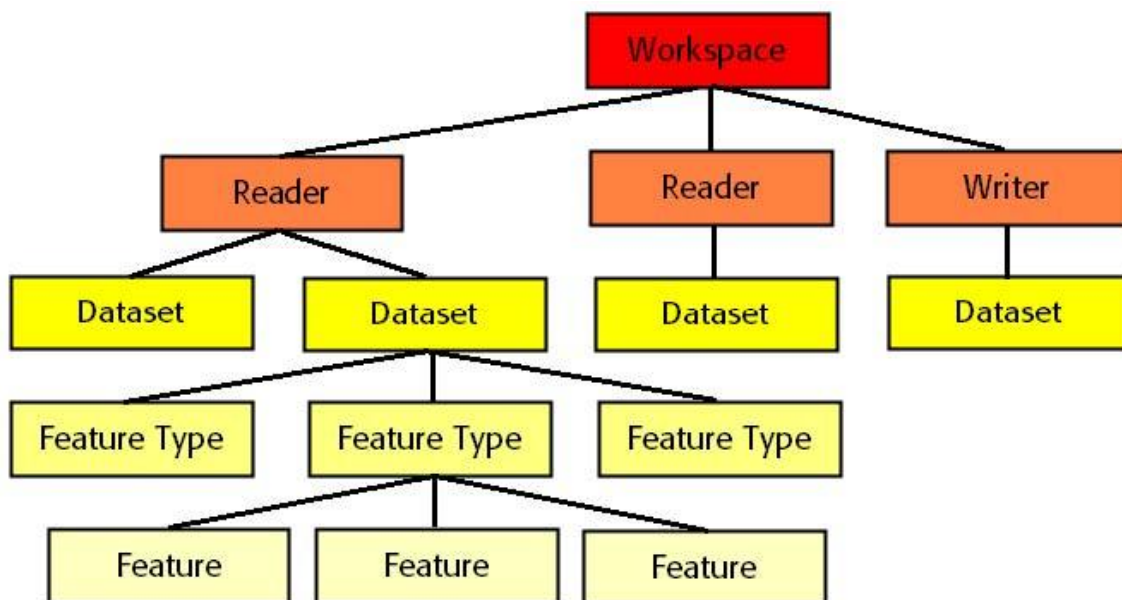
它指的是在一个网站上保存数据，例如，WFS (Web Feature Service)，这种情况下，数据集名和 URL (Universal Resource Locator) 名是一样的。

网站数据集内的要素类

网站数据集通常都会有许多层，而每一层表示一个不同的要素类，例如，WFS 是一个这种类型数据集的例子。

结构关系

下面这个图示很好地显示了工作空间，读模块，写模块，数据集要素类以及要素之间的关系。

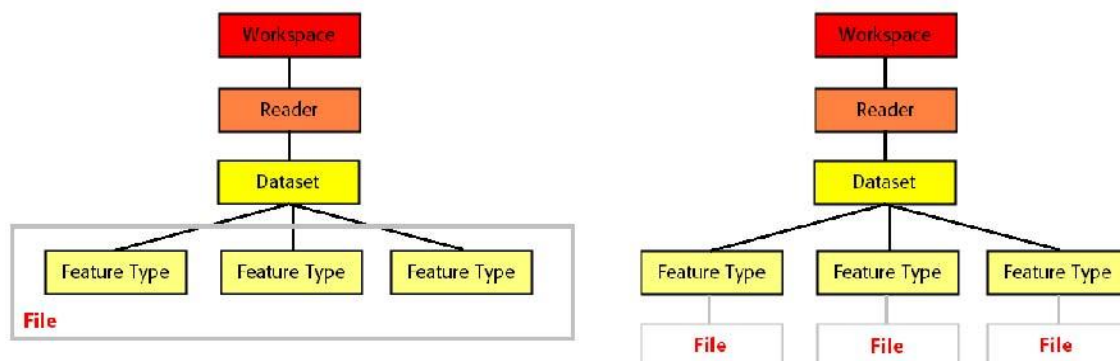


上面：这个图标显示了一个单独的工作空间是怎样包含多个读模块和写模块的。每个读模块包含了许多数据集，而每个数据集包括许多要素类，而每个要素类显然也包含了多个要素。

奇怪的是，每个目标写模块通常都只会写入一个数据集，唯一的例外情况是，当进行 Dataset Fanout 操作时，则会有多个数据集，在第 8 章中会有详细的讲解。

文件和文件夹形式的数据集

因为文件和文件夹形式的数据集是最常见的，也因为它们是最容易弄混的，所以用一个图标来表示它们之间的区别是非常有必要的，如图：



左上图：所有的要素类都被保存在一个文件中，那它一定就是文件形式的数据集。

右上图：所有要素类都有各自的文件，那它一定就是文件夹形式的数据集。

**Example 1: 将 CAD 转换为 GIS 格式**

将 CAD 转换为 GIS 是对空间数据进行的比较常见的处理。

这个例子中，按照要求你要将 CAD 格式的数据集转换为 GIS 格式。

1) 启动 Workbench.

打开一个新的工作空间对话框，使用以下参数创建一个工作空间：

Source Format	Bentley MicroStation Design (V8)
Source Dataset	C:\FMEData\Data\Roads\MajorRoads.dgn
Destination Format	Autodesk MapGuide Enterprise SDF

确定，将源设置“Group Elements By”设置成“Level Names”，按照提示，将 Roads 和 Labels 选择成添加到工作空间的要素类。

Autodesk MapGuide Enterprise SDF 是一个非常有趣的格式，虽然它是文件形式的数据集，但是却按照数据库的方法运行，事实上，它是以 SQLite 数据库为基础的。

2) 运行工作空间

选择工具条中 Destination Data > Redirect to Visualizer，对转换结果进行预览，运行工作空间。

按照提示，选择一个输出文件。

Destination Dataset	C:\FMEData\Output\TrainingModule4\Roads.sdf
---------------------	---

检查输出

观察在交叉点，马路是怎样分叉的，也就是说，一条马路（例如，US HWY 290E）是有多个部分组成的。在下面的例子中，我们会对它进行调整。

保存工作空间，方便以后使用。



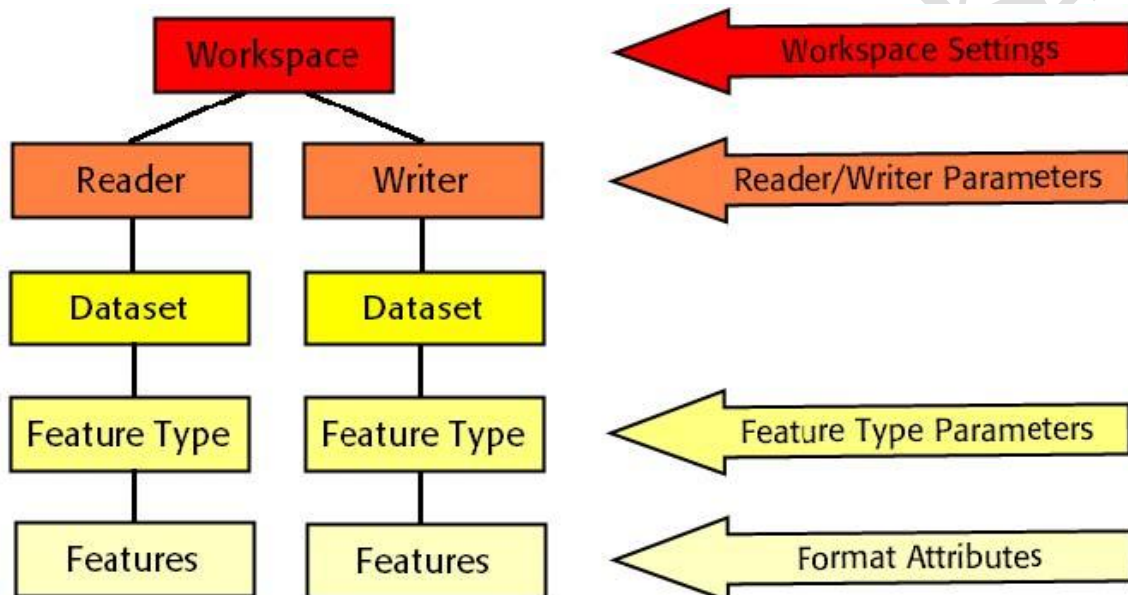
转换控制



不同格式之间的正确转换需要正确使用一些参数，来控制这个转换过程。

转换过程中不同组成部分的分层关系就需要不同的参数来控制相应的层次。

基本上说，高层次的参数会影响低层次的参数。



数据库写入模式就是一个很好的例子，因为它会贯穿整个转换过程。

首先，数据库写入模式（插入，更新或删除）可以在写模块级别被设置，它将应用到所有要素和表格。例如，如果将写模块模式设置成插入，那么就会插入所有的要素。

其次，数据库写入模式也可以被设置成要素类级别，它仅仅使用于要写入到表格中的要素，所以不同的表格就要求不同的写入模式。

最后，数据库写入模式也可以被设置成要素属性级别，它仅仅适用于要素。我们可以对在同一个表中不同的要素中来同时分别进行插入，更新或删除记录。

但是有趣的是，这种情况下，一般低层次模式设置具有优先性。例如，如果将要素属性操作模式被设置成“删除”，那么写模块级别的设置“更新”将不起作用。只要当没有设置较低层的参数时，才会应用到较高层的参数。

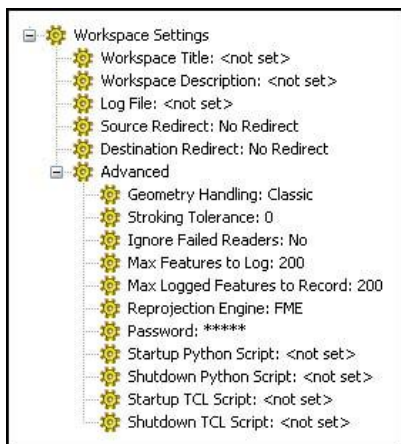
工作空间设置



工作空间的设置指的是与整个工作空间相关的。

什么是工作空间设置？

它指的是整个工作空间的所有参数，这些参数仅仅适用于正在使用的工作空间，并且可能因不同的工作空间而发生变化。



左图：Workbench 导航方框中显示的工作空间设置。

并不是所有的工作空间设置都会影响转换，有一些仅仅是因为操作的方便，例如，工作空间名称的设置和相关描述。其它设置则直接影响到转换方式。

基本的工作空间设置

有许多基本的工作空间设置。

Workspace Title

在 Workbench 的显示条中会出现 workspace title。默认 title 包含有关源格式和目标格式的信息，但是可以根据需求改变这个设置。

下图：使用工作空间参数的一个 title。



工作空间描述

这个设置允许用户输入有关工作空间的描述。当通过 FME Server 来使用工作空间时，这个设置就非常重要，因为它为网页提供了一种机制来描述准备进行的转换。

日志文件

当进行转换时，这个参数指的是创建日志文件的位置。

Source Redirect

source redirect 会取代 FME 从工作空间定义的源数据集读取数据的方式，并且促使 FME 从其它的源数据集中读取数据。唯一的 source redirect 选项就是‘Read from FFS file’。

Destination Redirect

这个参数会取代工作空间所定义的目标数据集，并且促使 FME 发送输出数据到其它位置，而且不会写入数据到目标数据集。为了重新写入输出数据，就要选择 No Redirect 设置来删除这个参数。



destination redirect 选项如下：

Redirect to Visualizer，直接将输出数据发送到 FME Universal Viewer。

Redirect to FFS File：将输出数据发送到 FFS(FME Feature Store)文件

Disable Output：忽略输出数据



上图：工作空间参数 s – Destination Redirect 选项。

你也可以在工具条下的目标数据集菜单中找到 Redirect to Visualizer’

高级工作空间设置

在 Workbench 中也有许多高级的工作空间设置，如下 ...

Geometry Handling

这个设置判断是否要使用基本的或是增强的处理功能来进行几何属性转换。虽然高级功能可能会产生更好的结果，但是这些结果可能与之前的转换结果有所不同，所以这就需要用户自己判断是要使用原来的方法，还是使用新的（增强的）工具。

Ignore Failed Readers

当读取数据集失败时，YES/NO 设置会告诉 FME 是否要继续这个转换过程，例如，如果输入了错误的密码，FME 就不能从数据库中读取数据，那么 FME 是否要从其它数据库中读取数据，然后继续转换过程呢？

Max Features to Log

“Max Features to Log” 限定了最多能写入多少要素到日志文件

Max Logged Features to Record

任何记录在日志窗口（或文件）中要素都会写入到空间日志数据集，这个数据集与日志文件具有相同的名字，但是要添加“_log”，它是 FFS 格式。这个设置决定要写入到日志数据集中的要素数量。



我们一定要清楚，限定日志要素数量能对多个，例如，三个 **Logger** 函数，每个函数都限定只能写入 10 个要素，这样从理论上说，能编辑 30 个要素。但是，“**Max Features to Log**”设置会限定总的数量，例如，如果设置成 10，那么从理论上说，限定的要素数量为 30 到 10。

你也要注意，这些设置不仅仅会影响到手动写入的要素，例如，使用 **Logger** 参数，而且还会影响到要其它的要素，例如，对 **AreaBuilder** 滤去的文本要素进行写入，要素数量是有限定的。

Reprojection Engine

不同的 GIS 应用程序稍微不同的算法来将数据投影到不同的坐标系中。为了确保 FME 写入的数据与已有的数据完全符合，用户就要使用这个设置，从另一个应用程序中获取 **Reprojection Engine**。



左图：安装了 ArcGIS 的用户选择使用页面提供的引擎来重新投射空间数据。

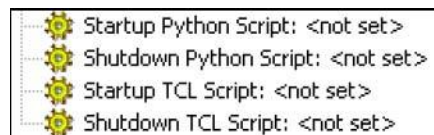
密码

通常会将工作空间传送给用户，这样 FME 用户就能运行这个工作空间了，但是并不希望对其进行编辑。通过设置密码，如果没有密码就不能打开工作空间对 **workbench** 进行编辑，虽然如此，你也可以从 **FME Universal Translator**，或者口令栏中，对工作空间进行编辑。

并且，开发人员或咨询人员更倾向于向 FME 用户传送工作空间，但是不展示空间的内容。用密码来保护工作空间，这样用户就不能使用标准的文本编辑器来读取空间内容。

启动，关闭脚本

使用这些参数就能在 FME 转换之前或之后，运行 **TCL** 或 **Python** 脚本。



右图：工作空间设置对话框中的脚本参数。

可能会用到以下这些脚本：

- 在进行转换之前，检查数据库连接。
- 在转换之前或之后，移动数据
- 写入转换结果到一个自定义日志，或将结果 email 给管理者
- 从其它程序中使用脚本(例如，ESRI ArcObjects Python scripts)



sEnTa cLaus 说过...

“现在好了！作为一个特殊的礼物，FME 现在安装了 FME2009 a Python interpreter，这样就不需要再单独安装一个 Python 了”



Example 2: 继续将 CAD 转换为 GIS 格式

1) 启动 Workbench.

如果有需要, 就启动 **Workbench**, 打开例 1 的工作空间, 或者重新创建一个。你也可以在 `C:\FMEData\Workspaces\TrainingWorkspaces\S4 - Example 2 - Begin.fmw` 中找到一个副本。

2) Clean Up Linework

例 1 中已经提到了, 在交汇点所有的马路要素就会被拆分成多个要素, 而目标要素要求将这些要素合并成一个单一的要素。

将函数 **LineJoiner** 放置到马路数据流中, 要连接的输出端口就应该是 **LINE** (而不是 **INVALID**)。

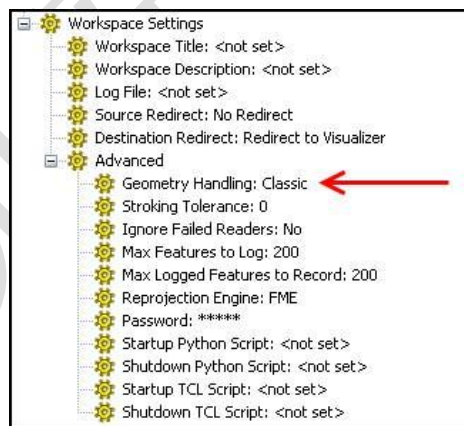
对于这个例子, 就只需要使用默认的函数参数。

3) 检查工作空间设置

在 **Navigator** 方框中, 检查工作空间设置“**Geometry Handling**”, 确保将它设置成 **Classic**。

运行工作空间, 检查数据输出。

对比下面的两幅图, 你会发现, **LineJoiner** 已经将一些线连接在一起, 注意, 左边那幅图中线之间的空白是为了便于更加清楚的查看。



但是, 也要注意, 在左图中, 两条线之间的连接在交汇点就终止了。

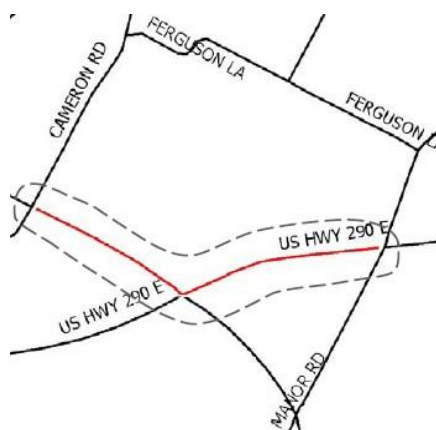
你知道这是为什么嘛?

4) 更改工作空间设置

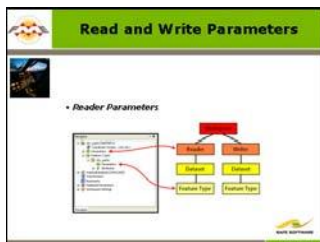
奇怪的是, 在右图中, 在交汇点仍然能够将两条线连接起来。其实很简单, 只需要将 **geometry handling** 改为“**Enhanced**”, 就可以了。

所得到的结果是一样的吗?为什么?

NB: 之后我们会找到问题所在, 并修补 **LineJoiner** 函数, 给出正确的结果。



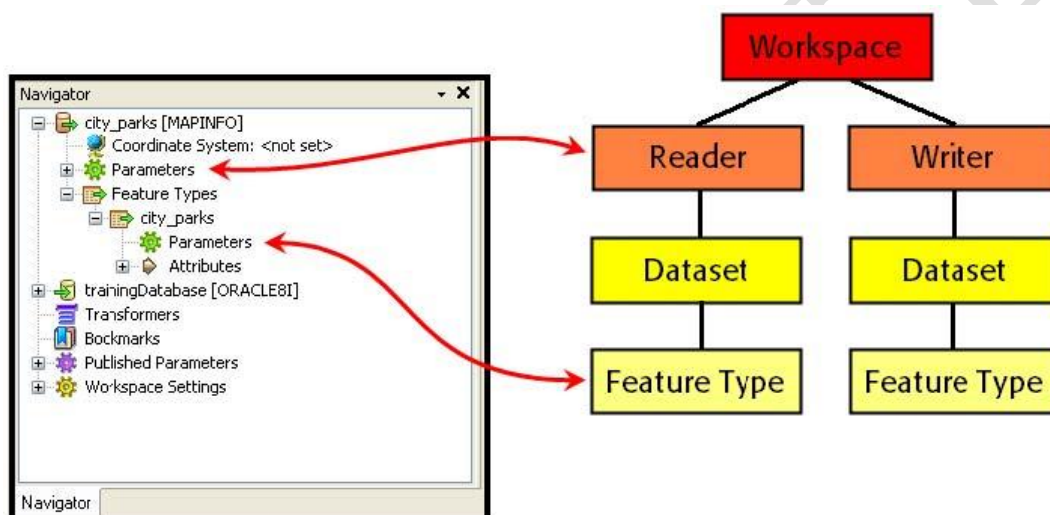
读取和写入参数



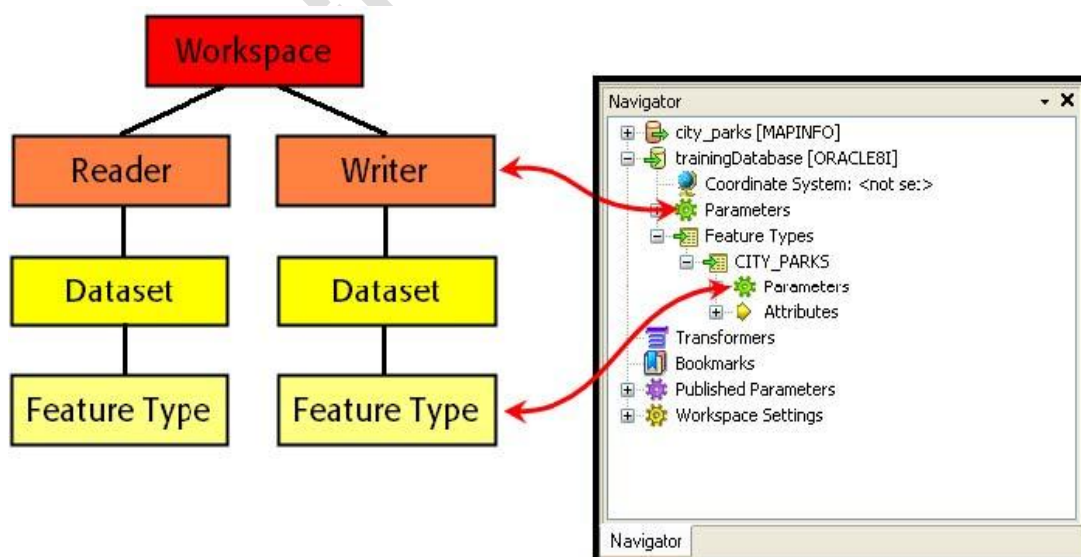
每个写入到工作空间的读模块或写模块都是由大量设置和参数所控制的，而用户可以自己获取这些参数。

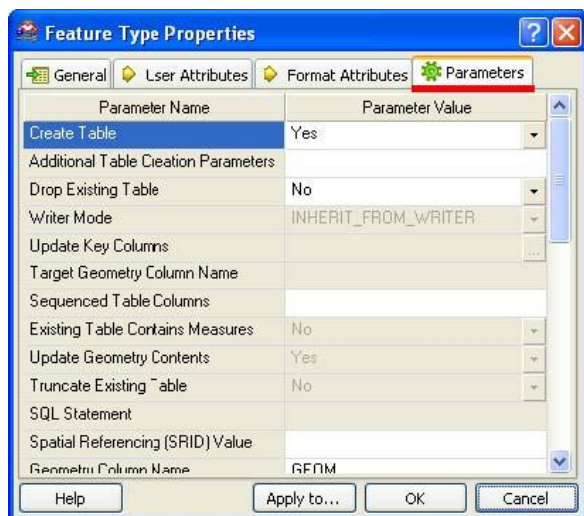
参数用来控制读取或写入数据的方式，并且它适用于所有的读模块/写模块，也适用于某一个要素类。

下图：在 Navigator 中很容易中获取读模块参数。



下图：同样地，也很容易获取写模块参数，同样的目标要素类参数并不是那么明显





左图：在 Feature Type Properties 对话框就能访问到要素类参数，注意“Parameters”方框。

并非所有的要素类都有参数，所以这个标签页并不总是存在。

读取和写入参数

在分层关系结构表中，读模块属于较高层次的，它们的参数适用于其它要素类，例如，要读取所有的数据集，就会包含所有要素类。同样地，写入参数也满足所有的数据集，以及那些要写入的要素类。

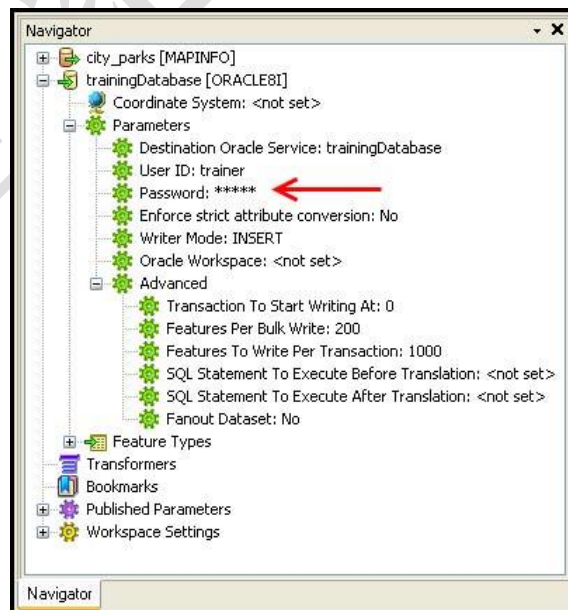
在 Workbench Navigator 方框中就会显示 Reader/Writer 参数。双击参数，就能对它进行编辑，然后就会打开一个对话框，输入参数值。

右图：在 navigator 方框中显示 Oracle Spatial 的写模块参数。

Password（右图中红色标记的）就是有关写模块级别的参数的一个例子。显然，一个密码使用于所有要读取的表格，也就是说，并没有为每个表都设置一个密码。

因为读写模块参数指的是某个特定格式的组成部分和特征，所有没有两种格式会有相同的参数。

并且，可能会需要不同的参数，甚至同一种格式的读写模块可能都会要求不同的参数。



为了方便使用，我们将参数划分为两部分：基本的和高級的。



Example 3: 继续将 GAD 转换为 GIS 格式

1) 启动 Workbench.

如果有需要就启动 **Workbench**，打开例 2 中的工作空间，或重新创建一个。当然，你也可以在 `C:\FMEData\Workspaces\TrainingWorkspaces\S4 - Example 3 - Begin.fmw` 中找到它。

2) 更改输出数据集

我们知道，已经有了一个 **SDF** 数据集，它能够进行一般的转换。因为 **FME** 能够支持 **SDF** 格式，所有我们能够添加道路数据。

在 **Navigator** 方框中找到写模块参数—“Destination SDF 3 File”。

双击它，让后将它改为 `C:\FMEData\Data\Transit\Transit.sdf`。

3) 更改写模块参数

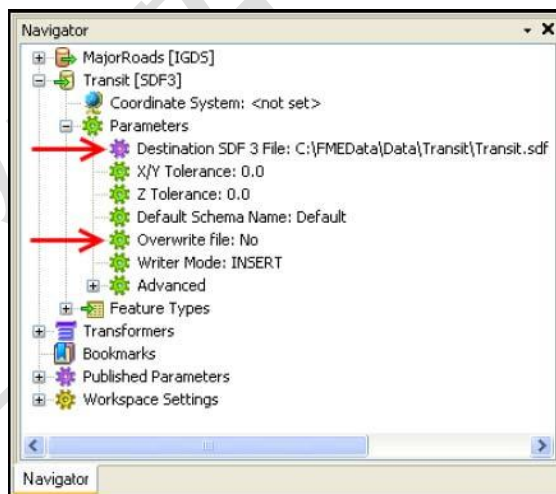
检查已有的 **SDF** 数据集，注意，这个数据集中已经储存了数据。

我们并不打算覆盖这个数据集中的一些数据，所以就必须要将写模块参数设置为附加模式。

在 **Navigator** 方框中，找到写模块参数—“Overwrite File”。

将它设置为 **NO**

右图：新的 **Navigator** 窗口看上去就是这样的。



4) 运行工作空间

关闭 **Destination Redirect**，然后运行工作空间（仅仅运行一次!）。

检查 **SDF** 数据集，确保已经保存了原来的数据，并且成功地添加了道路数据。



左图：SDF 数据(FME Viewer 中显示的) 看起来就是现在这个样子的。

要素类参数

在分层结构图中，要素类属于较低层次。

所以，要素类参数并不影响于整个数据集，仅仅适用于某个要素类。

它们分别对读取和写入时不同的要素类进行控制。

右图：针对 Oracle 目标要素类的参数。

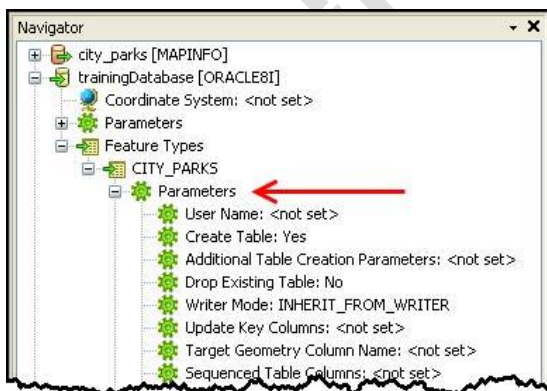
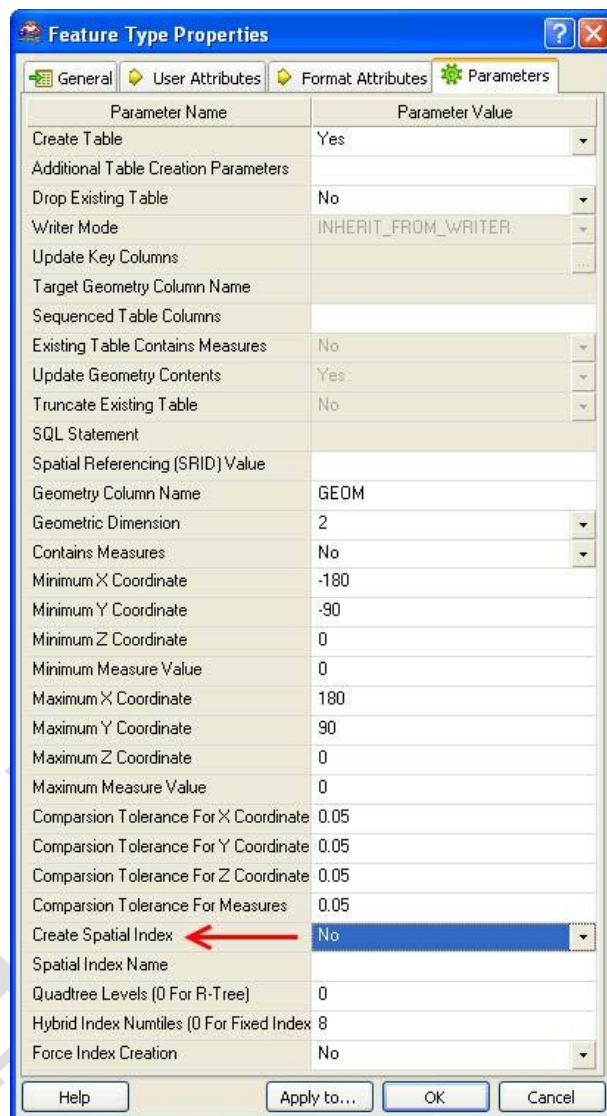
“Create Spatial Index” (红色标记的) 就是一个很好的例子。

我们通过表来决定是否需要使用一个索引；一些表可能需要，而一些可能不需要索引。

因为每个表指的是一个不同的要素类，所以 FME 通过要素类参数来控制这些索引。

这样就使参数变得非常灵活了。如果是一个写模块级别的参数，所有的表都会获取到一个索引，但这个索引并非就是用户想得到的。

相反地，没有将密码参数列出，这是因为它适用于整个数据库，而并非仅仅适用于某个表。



左图：在 Navigator 方框中的要素类参数—Oracle。

并且也只能在 Navigator 方框中发布要素类参数（在下一节中我们会看到）。

因为我们已经在单独的对话框中将所有参数都列了出来，所以在 Feature Type Properties 对话框中更方便做大量的修改。



Example 4: 继续将 CAD 转换为 GIS 格式

1) 启动 Workbench.

如果有需要就启动 Workbench，打开例 3 中的工作空间，或者重新创建一个，你也可以在 C:\FMEData\Workspaces\TrainingWorkspaces\S4 - Example 4 - Begin.fmw 中找到它。

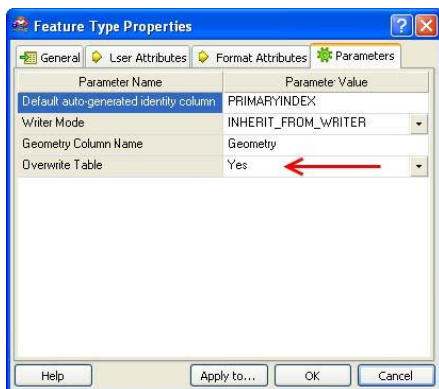
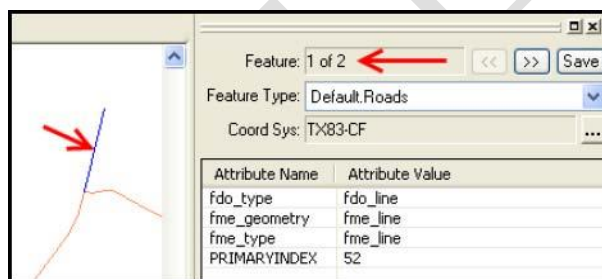
2) 重新运行工作空间

重新运行例 3 中的工作空间

检查输出数据

可能很难看到工作空间，但是我们要将更多的数据写入到同一个文件中，而不是覆盖原有文件，所以我们实际上将马路数据保留了多个副本。

右图：这里给出了提示，某处出现了错误。
查询一个要素会返回它的多个实例。



3) 调整要素类设置

最简单的解决方法就是在写入数据时，清空其中的内容。

我们并不想删除整个数据集，因为我们需要保存一些已有的数据，只是想删除有关表格其中的内容，所以，就需要考虑更改要素类参数。

打开目标 Roads feature type properties 对话框，点击 Parameters，将“Overwrite Table”设置更改为 YES。

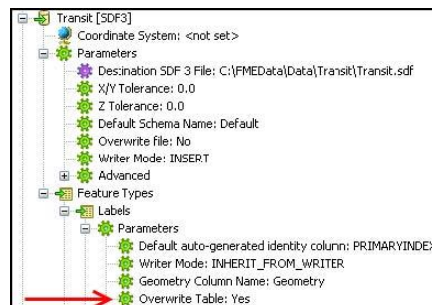
4) 调整要素类设置

对于“table”，进行相同的操作。

使用 Navigator 方框，而不是 properties 对话框，来找到 Labels，将“Overwrite Table”设置更改为 YES。

5) 重新运行工作空间

重新运行工作空间，检查数据，确保只有一个工作空间的副本。



发布参数



发布参数指的是一种方法，在运行的时候提示用户更改读写参数。

什么是发布参数

就像你所知道的，在进行转换时，FME 会提示用户要提供某个参数或设置的值，如果这些参数没有被完整定义的话。发布参数就是这样一种方式来提示用户对任何配置进行设置，而不管是否这个参数已经被正确定义。

在导航方框中出现的任何设置都能够被设置成一个已发布的参数，这种功能不仅仅指在读写模块级别控制参数，也包括要素类设置，函数设置，以及大部分工作空间。

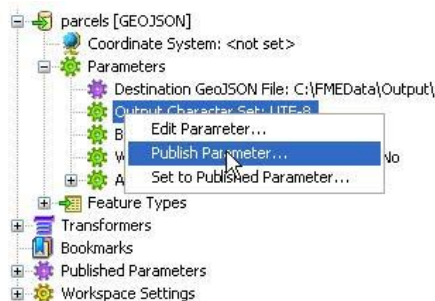
怎样创建已发布的参数

在导航方框中找到这个参数，右击它，然后选择‘Publish Parameter’选项，就能够发布一个参数了。

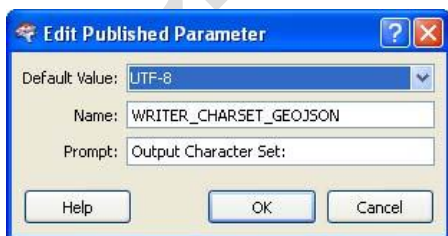
在定义了已发布参数的名字后，它的图标就会变成紫色，表示它已经被发布了。在 Navigator 的一个单独部分会显示所有已发布的参数。

右图：FME Workbench 中的 Navigator 方框。这里，用户打算发布一个写模块设置，这个设置决定了要写入数据的字符编码。

注意写模块的目标数据集参数是怎样转变为紫色的，这就表示这个参数已经被发布了。事实上，所有源数据集和目标数据集设置的默认状态都是将要被发布的状态。

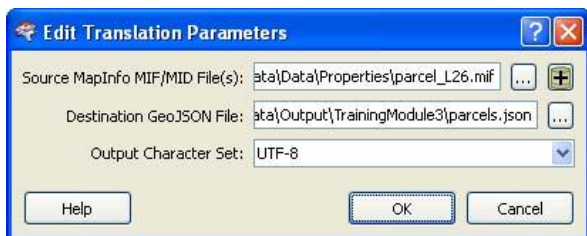


下图：创建发布参数的对话框。我们已经定义了写模块参数的发布过程。观察，用户是如何按照提示输入数值来改变改变这个文本值的？



怎样使用已发布的参数

从 FME Workbench 的工具条选择 File > Prompt and Run Translation，激活已发布的参数，或者使用快捷键<Ctrl+R>。

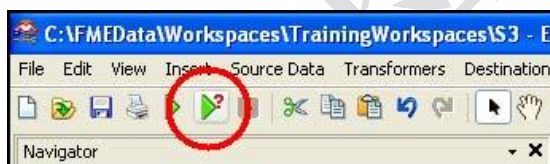


左图：在使用“File > Prompt + Run Translation”来开始转换时，会提示用户输入所有已发布参数的值。

记住，所有的源数据集设置和目标数据集设置都被默认发布了。



Workbench 工具中有一个‘Prompt and Run’按钮（如右图），使用它就能够更简单地进行转换。一般这个按钮是隐藏的，所以你必须使用工具条中的自定义工具来添加它。



Command Line 中已发布参数

你还记得怎样使用 Command Line 来进行转换吗？

使用命令行的规定语句就可以将值发送给已发布的参数

--<Parameter Name> <Parameter Value>

下图：这个日志窗口显示了上面转换中所用到的指令。注意，参数名是与已发布参数的定义值是相匹配的。

```
Windows command-line to run this workspace:
fme.exe "S3 - Exercise - Complete.fmw"
--SourceDataset_MIF C:\FMEDData\Data\Properties\parcel_L26.mif
--DestDataset_GEOJSON C:\FMEDData\Output\TrainingModule3\parcels.json
--WRITER_CHARSET_GEOJSON UTF-8
```

在 FME2009 中，日志窗口中的 command line 被特别更改为 Windows command-line”



之前并不是很清楚，指令将应用于哪个操作系统，因此，就会清空指令，并且将指令设置为 Windows 形式，Windows 和 Unix shells（操作系统）之间的最大区别就是，windows 中的命令要素用空格来隔开，而 Unix 使用引号来隔开。



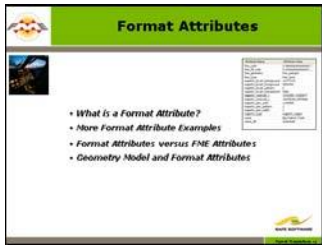
Example 5: 继续将 CAD 转换为 GIS 格式

1) 发布参数

打开例 4 的工作空间，或者重新创建一个。

为 Roads 和 Labels 发布“Overwrite Table”参数，这样在运行时就会提醒用户是否要清空内容。

格式化属性



到现在为止我们仅仅处理了‘user attributes’，这些属性要么是源数据中有的，要么就是工作空间创建的。这部分会引入一组不同的属性，它们是由 FME 创建。

什么是格式属性？

格式属性是系统设置的，FME 会对任何格式，创建一个与结构或图例相关的属性，例如，一个格式属性就记录了要素的颜色。

用户可能会将这些属性当做工作空间的一部分，使用它们来完成某些任务。通常，在 FME Viewer 中查看数据集时，都能够看到这些格式属性。当查询一个要素时，不仅仅能够看到用户属性，也能够看到大部分格式属性。

这种情况下，属性就会以源数据的方式表现数据的属性。

右图：FME Viewer 中的 MapInfo TAB 要素。

在之前的数据集中储存用户自定义属性：“name” and “name_alt”

“mapinfo_pen_color”是一个格式属性，FME 使用它来追踪 MapInfo TAB 格式的要素颜色。

“mapinfo_type”表示的是要素的几何属性类别，由 MapInfo TAB 格式定义。

Attribute Name	Attribute Value
fme_color	0.06666666666666667...
fme_fill_color	0.26666666666666667...
fme_geometry	fme_polygon
fme_type	fme_area
mapinfo_brush_background	16777215
mapinfo_brush_foreground	4504392
mapinfo_brush_pattern	2
mapinfo_brush_transparent	false
mapinfo_centroid_x	3141650.13260477
mapinfo_centroid_y	10078298.6497858
mapinfo_pen_color	1148949
mapinfo_pen_pattern	2
mapinfo_pen_width	1
mapinfo_type	mapinfo_region
name	Big Walnut Creek
name_alt	Greenbelt

更多有关格式属性的例子

一个要素的颜色是有关格式属性最简单的例子，但是，除此之外，还有许多其它的属性，它们随着难易程度而发生改变。下面这些例子就是有关格式属性的：

Rotation: 许多要素都是围绕 Z 轴旋转的，Z 轴就被作为一个格式属性来保存，例如，fm0_rotation（要素以 GeoMedia 格式进行旋转）

Subtype: 对于支持 subtypes 的格式，例如，一些类似于内置的查找表格，它的相关值可能就被设置成格式属性，例如，geodb_subtype_name（针对 ESRI Geodatabase 的 subtype 字段）。

Raster Pyramid: 处理栅格要素的方式不同于矢量要素，其中的一个不同点取决于格式属性。具体来说，Raster Pyramids 通常被保存为设置为格式属性，例如，oracle_raster_pyramid_type（当写入栅格数据到 Oracle 数据库时，就要使用到它）。

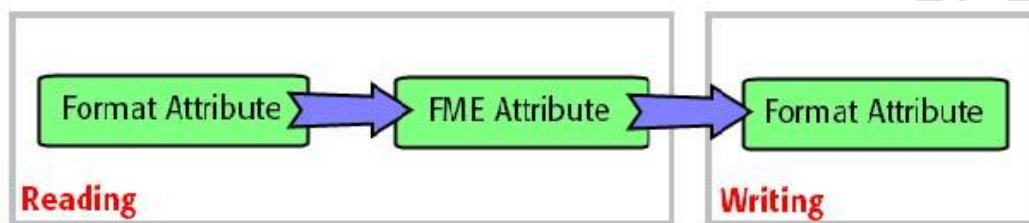
格式属性与 FME 属性

一类特别的格式属性带有前缀 `fme_`，当使用 FME 来查看属性时，它表示的就是数据，所以属性有时候也被叫做 FME Attributes 或 Generic FME Attributes。

当进行转换时：

- FME 读取源数据，并且将属性的相关信息保存为格式属性，这些格式属性反映了数据在源数据格式下的情况。
- FME 将源格式属性转换为 FME Attributes。这些 FME attributes 反映了数据在 FME 内部格式存储下的情况。
- FME 通过创建一组新的格式属性来写入目标数据。这些格式属性反映了数据将被存储于目标数据格式下的情况。

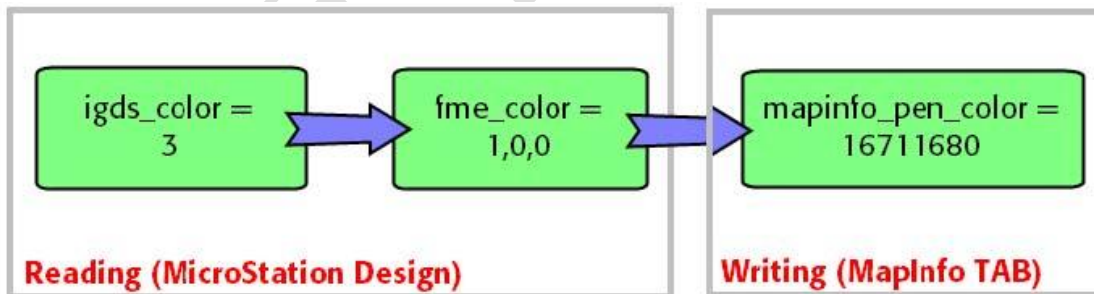
下图：The FME Translation Process and Format Attributes:



这就是为什么，当你检查数据时，你会看到两组属性：前半部分，你会看到 `mapinfo_pen_color` 和 `fme_color`；后半部分的格式属性就是对前面的表达与展现。

使用这种方法，FME 就能够进行格式转换，并且不需要针对每种格式，分别将源格式属性映射到目标格式属性。它仅仅将一切数据转换为 FME 标准中间格式，然后再将中间格式转换为目标格式。

下图：在将 MicroStation DGN 转换为 MapInfo TAB 时，有颜色的格式属性



几何模型和格式属性

随着 FME 几何属性模型的发展，它能够将更多的要素定义保存为几何属性，所以，从长远来看，格式属性也变得不那么重要了。

这包括记录要素几何类别的属性：`fme_type` 和 `fme_geometry`。

我们可以在工作空间中获取 FME Attribute，与获取其他格式属性的方法相同，但是，我们要尽量少用这些属性，因为有关几何类别的信息时保存在几何模型中的，并且并且只能通过其他方式访问。

用格式化属性过滤



源格式属性的主要用途是他提供了在工作空间中过滤数据的一种方式。

从源数据中读取的格式属性，在工作空间中主要使用它来过滤和传导数据。

例如，假设没有将 AutoCAD 数据集内的文本要素划分为不同的层，用户就能够根据文本的大小来选择合适的层，因此就可以根据 `autocad_text_size` 格式属性，使用 FME 来对数据进行分层。但是，在 FME Workbench 中使用格式属性之前，一点要先开放这些格式属性。

展开格式属性

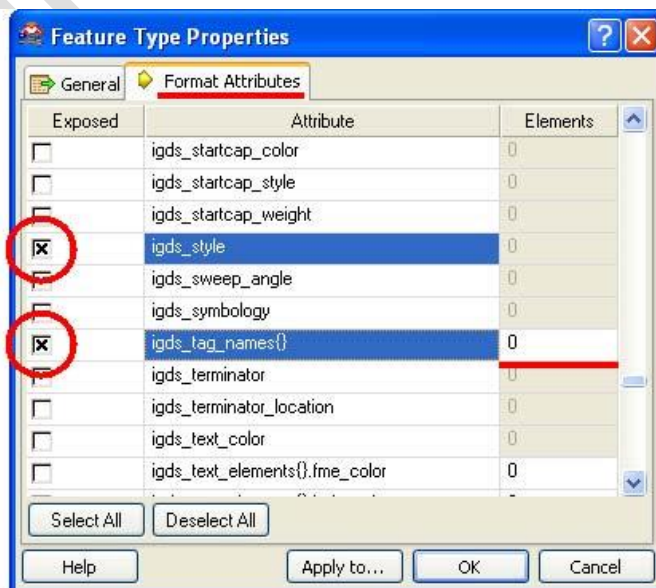
为了避免杂乱，一般在 Workbench 中是看不到格式属性的。所以，要使用它们，就必须首先“开放”它们。

我们使用源或目标 feature type properties 对话框来展开格式属性。点击 Feature Type Properties 按钮，打开对话框，然后点击‘Format Attributes’，找到你需要看到的格式属性，然后在前面的方框中打钩，点击 OK，这样就能在 workbench 中看到格式属性了。

右图：Feature Type Properties. 已经选择了，并且在旁边的方框中打钩了，`igds_style` and `igds_tag_names{}`

例如，`igds_tag_names{}`。使用右边的那一栏，这样用户就不仅可以选择要开放整个属性列表，也能够选择开放部分属性。

虽然我们并不能总是看到格式属性，单丝它们却是始终存在的。所以，有时候你能够使用格式属性，但却不用首先将它们开放。



Firefighter Mapp 说过。。。

“另一种方法就是，使用 `AttributeExposer` 函数来展开格式属性。”

Example 6: 继续将 CAD 转换为 GIS 格式

1) 启动 Workbench.

如果有需要就启动 aWorkbench，打开例 5 中的工作空间，或重新创建一个，或者在 C:\FMEDData\Workspaces\TrainingWorkspaces\S4 - Example 6 - Begin.fmw 也可以找到它的附件。我们也会在这个文件中处理 LineJoiner 的有关问题。

问题就是，如果有多于一种可能的连接，那么线段将不能被连接起来。使用 Group-By 就能解决这个问题了，例如，将具有相同马路 ID 的线类连接起来，这样就能避免不需要的连接，并且让 FME 决定在哪里进行合适的连接。

2) 展开一个格式属性

用户不能使用马路 ID 来组成一组，这是因为，MicroStation 数据并不包含属性，但是它可以与一个属性数据库进行连接。在 FME 中，这种连接可以作为一个格式属性，并且必须要被展开，从而被 Workbench 使用到。

打开马路的源 feature type properties，点击 Format Attributes，在 mslink_0 这一栏打钩，然后点击 OK。

3) Fix LineJoiner Transformer

在 LineJoiner 函数中，设置 group-by，将名为 mslink_0 的要素归为一组。

4) 重新运行工作空间

重新运行工作空间，你会发现，所有含有相同 ID 的马路都被正确地连接起来了。

同样地，在基本和高级设置间切换 Geometry Handling。

选择的不同，会对输出产生什么不同的影响呢？。

你认为为什么 FME 会以这种方式来区分数据处理的方式？



用格式化属性转换



这部分是有关一个常见问题的解决方法，在 FME 转换过程中，怎样重组数据集结构，例如，改变数据集的颜色，线的宽度，线的形式。

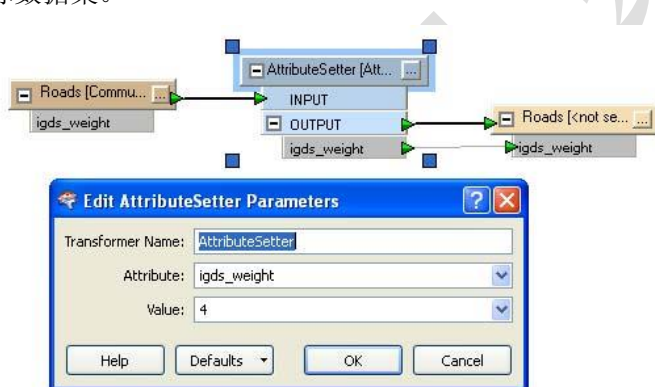
前面已经提到了，FME attributes 会被转化格式属性，它表示要写入的数据，但是，通过对目标格式属性进行重新定义，用户可覆写这个转换过程。

换句话说，通过为目标格式设置一个格式属性，便可以不需要使用一个特定函数，用户就能够进行数据转换了。

AttributeSetter 函数

通过展开源读模块的格式属性，就能够改变 AttributeSetter 的值。

只要从同一种格式的数据集中读取和写入数据，该操作便不会影响到源数据，但是会影响到目标数据集。



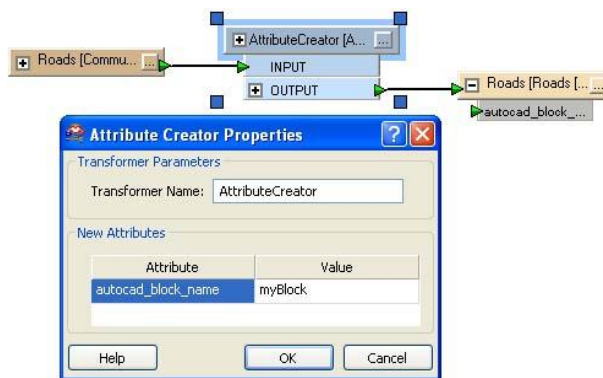
左图：这是一个有关符号转换的例子，展开 `igds_weight`（用来定义线宽度的格式属性），并且使用 `AttributeSetter` 函数，将所有的要素的线类重量设置为 4

因为用户会将数据重新写入到 `MicroStation`，所以 FME 就会应用格式属性来改变线类宽度。

AttributeCreator 函数

当我们写入/读取不同格式的数据时，仅仅在目标数据段展开格式属性并不能使其在整个工作空间中也自动展开，这时，就需要使用 `AttributeCreator` 函数。

右图中，将 `MicroStation` 转换为 `AutoCAD` 格式，`autocad_block_name` 已经展开，但是还没有在整个工作空间中对它进行设置。使用 `AttributeCreator` 函数创建属性，并设置一个属性值。



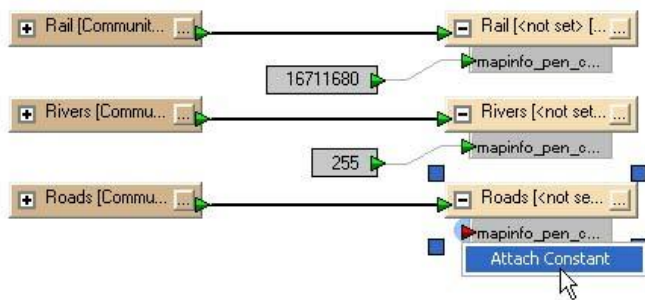
常量

使用常量（而不是一定要使用函数）也能对展开的目标属性进行设置。右击一个属性，选择‘Set to Constant Value’，或者是右击端口，选择‘Attach Constant’，就能够创建一个常量了。

使用相同的方式，就能为要素类格式属性创建一个常量（例如，这个常量适用于所要写入到那个要素类的要素）

右图：将数据写入到三个 MapInfo TAB 文件中。

已经找到了属性 `mapinfo_pen_color`，并且设置了一个常量，红色的表示铁路，蓝色的表示河流。



这个例子显示了在处理过程中将最后的格式属性设置为绿色。

FME 以后的产品，不需要通过要素的格式属性来处理要素，而是可以使用函数来处理要素的几何属性。

例如，`ArcPropertySetter`，用户不需要通过 `fme_primary_axis` 属性来改变弧形要素的半径，而是使用 `ArcPropertySetter` 函数。

格式属性的缺点

尽管格式属性的功能很有用，当使用它来重组数据时，仍然会出现一些问题，特别是在读取和写入同一种格式的数据时。

属性的优先性

如果你同时定义一个格式属性和一个 FME 属性，来对目标数据进行相同的设置，格式属性就更有优势，并且会被优先使用。

例如，如果你要写入 `MicroStation` 格式的数据，将它的值定义为 `igds_color` 和 `fme_color`，当你进行数据输出时，就会优先使用 `igds_color` 属性。

你可能感到很困惑，因为当你同时设置这两个属性值时，它们的区别并不是那么的明显（例如，在读取和写入相同格式时，目标格式属性是已经存在的，或者 FME 属性要求优先使用）。

交织着的属性

FME 特意将有些格式属性交织在一起。当改变了某个交织的属性时，同时就会影响到其它的属性值。

例如，设置一个 `fme_color` 就可能覆盖格式颜色属性（例如，`igds_color`），这是因为这两个交织的属性，改变其中一个，就会将其他的属性值更改为相同的值。

这里你可能感到非常困惑，因为这似乎违背了属性之间的优先性原则，似乎使 FME 属性优先于格式属性，所以为了减少困惑，在必要时就要使用函数来对属性进行重组，避免调整格式属性。

FME 读写手册



FME 用户可以获取许多指导手册，而 FME Readers and Writers Manual 只是其中的一种，它包含许多有关单一格式数据集和要素类的有用信息。

The FME Readers and Writers Manual 是 FME Workbench 帮助文件体系的一部分。

这个培训手册中有一章是介绍 FME 支持的每一种格式的。并且每一章中分为几节，每节都有不同的内容：一节为格式的简介，一节关于格式的 FME 读模块，以及可获取的设置，一节关于 FME 写模块以及相关设置，一节关于怎样表达要素的几何特征。

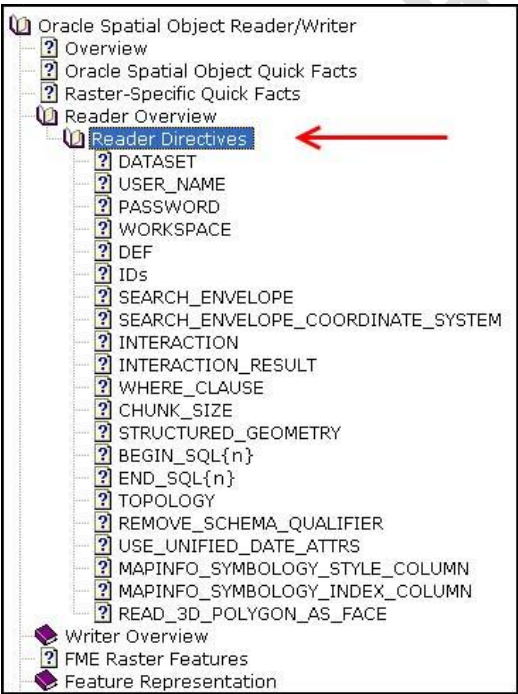
格式简介

这一节讲述的是格式的数据集类型，例如，文件夹，文件，网站或者数据库，以及格式的要素类。

右图： Oracle Spatial Quick Facts 快速浏览页面，数据集类别为数据库，而要素类则是“表”

Oracle Spatial Object Quick Facts

Format Type Identifier	ORACLE8I ORACLERASTER
Reader/Writer	Both
Licensing Level	See Format Notes
Dependencies	See Format Notes
Dataset Type	Database
Feature Type	Table name
Typical File Extensions	N/A
Automated Translation Support	Yes
User-Defined Attributes	Yes
Coordinate System Support	Yes (reading only) or Raster
Generic Color Support	No
Spatial Index	Always
Schema Required	Yes
Transaction Support	Yes
Rich Geometry	Yes
Geometry Type	oracle_type



Reader Directives

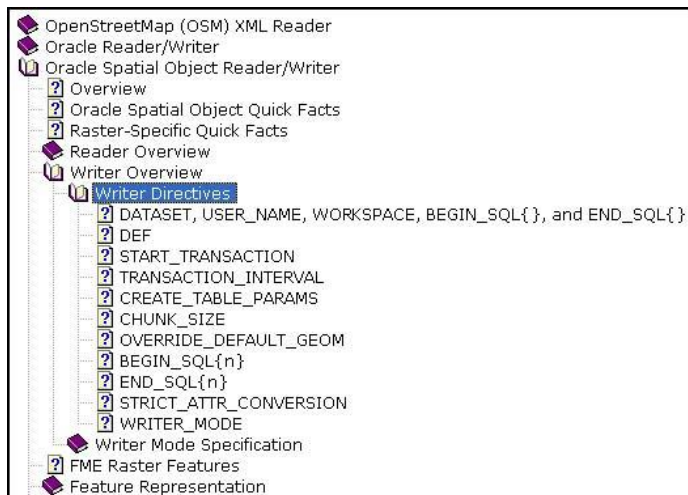
Reader “Directives” 相当于 Navigator 方框中的读模块的参数，但是这个名字更多地是与映射文件有关，而不是工作空间。

左图： Oracle 空间输入的 Reader Directive，注意，列表中的大多数 Reader Directive 是怎样获取一个对应的参数的（例如 USER_NAME 和 PASSWORD）。

Writer Directives

不必感到惊讶, Writer “Directives” 就相当于 Navigator 方框中的写模块参数。

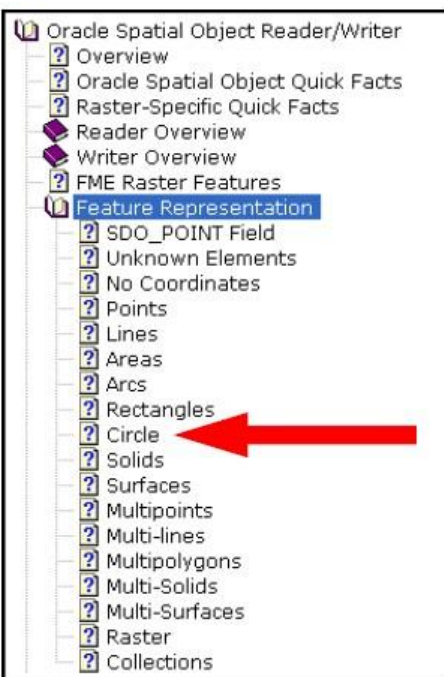
右图: Oracle 空间输入的 Writer Directives



要素表示法

这一节会深入讲解每种格式支持的几何属性类型。

其中最重要的部分就是介绍了各种几何特征的格式属性的列表。



左图: Oracle Spatial 的要素表示部分

下图: Oracle Spatial 里的 Circle 实体的要素表示, 属性格式列表是 oracle_radius 和 oracle_rotation。

Circle

oracle_type: oracle_circle

Circle features are tagged with this value when reading or writing to or from Oracle Spatial (object model). The circle is defined in the FME feature by a center point and an attribute to define the circle's radius:

Attribute Name	Contents
oracle_radius	The length of the circle's semi-major axis, measured in ground units.
oracle_rotation	The rotation of the major axis. The rotation is measured in degrees counterclockwise from horizontal.

高级格式控制



一旦你能够发布写入和读取参数，就可以访问大量的高级控制方法了。

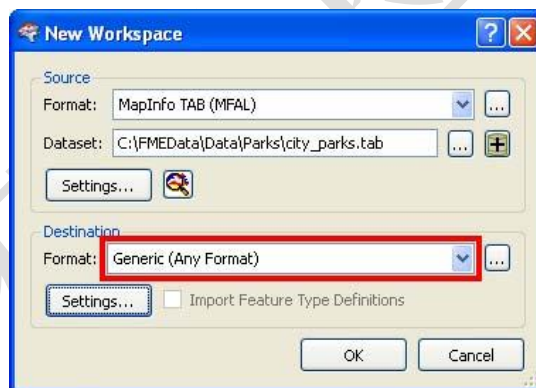
这些方法能够帮助你创建和维护常用的工作空间，特别是当你在 FME Server 下部署工作空间时，它们就变得更加有用了。

The Generic Writer

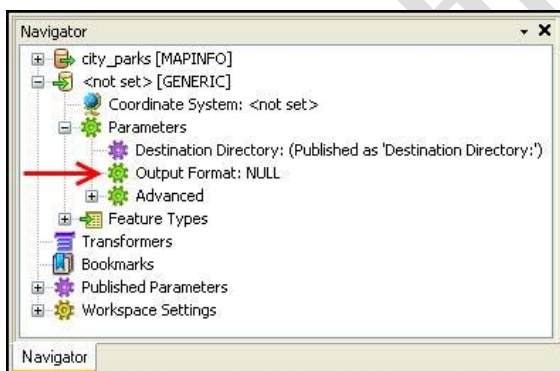
通用写模块的是一个对象，它代表一种没有指定格式的目标数据结果。

当运行这样一个带有通用写模块的工作空间时，写入的数据格式是由在 **Navigator** 方框中设置的参数决定的。

右图：创建一个工作空间，使用通用写模块来转换有关公园的数据。



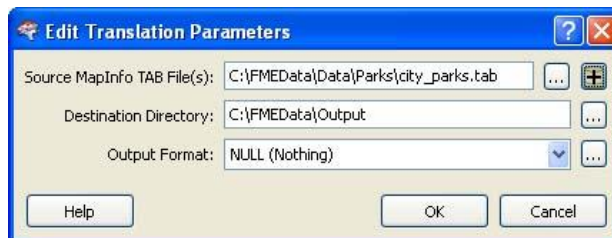
因为格式参数是可以被发布的，这也就是说，可以创建一个通用的工作空间，并且在运行这个空间的时候用户可以选择要写入的格式。



左图：**Navigator** 方框显示的不仅仅是一个目标位置参数，也是目标格式参数。

可以使用与发布目标位置参数相同的方法来发布目标格式参数。

右图：通过发布参数，就会提示用户选择输出位置和输出格式(通常目标格式是一个目录，即使选择的格式是文件形式的)



```
Windows command-line to run this workspace:
fme.exe wb-xlate-122998087194_9600
--SourceDataset_MAPINFO C:\FMEData\Data\Parks\city_parks.tab
--DestDataset_GENERIC C:\FMEData\Output
--GENERIC_OUT_FORMAT_GENERIC GML
```

左图：甚至通过发布参数，你可以从 **command line** 中设置一个常用的写入格式。

要读取的要素类

通常来说，用户希望进行 FME 转换，但是仅仅想在已定义层的子集中运行工作空间。

最常见的方法就是，在运行工作空间之前删除或是移除多余的要素类，但是显然就必须要对工作空间进行大量的编辑，并且还要将工作空间还原为最初的状态。

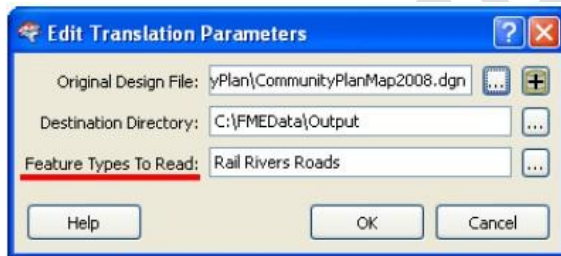
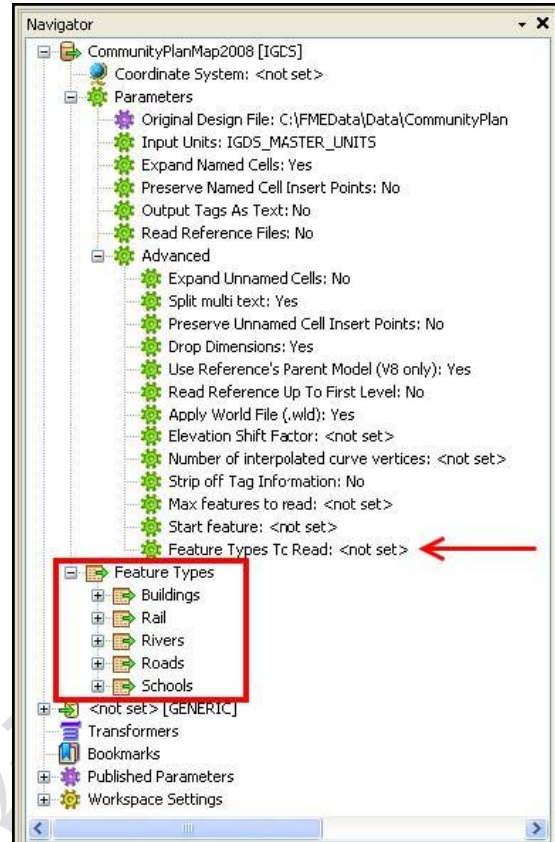
为了不让用户被迫使用这种方法，Workbench 所包含的一个高级 reader 参数就允许用户选择要读取的要素类。

因为这个参数是可发布的，所以就能够创建一个常用工作空间，并且在运行工作空间时，用户可以从这个空间中选择要读取的要素类。

右图：这里，一个工作空间包含五种不同的要素类。

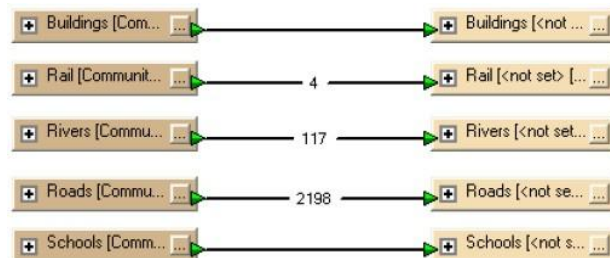
Buildings, Rail, Rivers, Roads, and Schools

用户可以从中进行选择



左图：作为一个可发布参数，也同样能在 command line 中对它进行设置。

右图：要素的数量显示工作空间只读取了选择的要素类

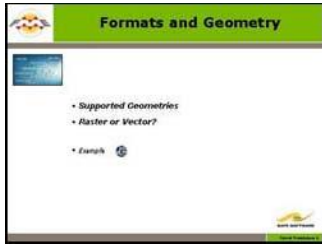


Windows command-line to run this workspace:

```
fme.exe wb-xlata-1229981471546_9600
--SourceDataset_IGDS C:\FMEDData\Data\CommunityPlan\CommunityPlanMap2008.dgn
--DestDataset_GENERIC C:\FMEDData\Output
--FEATURE_TYPES "Rail Rivers Roads"
```

左图：发布参数也可以在命令行中设置。

语义转换



因为 FME 是一个语义转换器， 所以它能根据待转换要素使用的格式来做出转换判断。

FME 是一个语义转换器，这就是说，它会自动进行数据转换，使数据符合目标格式的定义和规则。

换句话说，用户通常是看不到转换过程的。FME 这样做完全是出于为用户考虑在转换过程中尽可能多地保存数据的意义，但是，用户通常并没意识到不同格式之间细微的差异，他们可能会对如此的重构结构感到惊讶。

FME 支持的几何属性

事实上，并非每种格式都支持 FME 所支持的几何属性，例如，ESRI Shape 格式就不支持弧形要素，Oracle spatial 就不支持文本要素。

当转换目标格式不支持的几何属性时，FME 总是尽力将数据转换为目标格式支持的几何属性。例如，FME“打断”弧形（即意味着将弧形转换为线类要素），使用点类要素来取代文本要素，并用一个属性来存储文本内容。



Ms Analyst 写到...

“亲爱的 Aunt Interop 我正在处理我不熟悉的格式，那么我怎样才能确保几何属性类型是 FME 所支持的呢？”



Aunt Interop 写到...

“亲爱的 Ms A. 你会担心这个问题是非常自然的。在 FME Readers and Writers Manual 中的‘Quick Facts’（如右图）这一节中，你会找到 FME 针对所有格式各自支持的几何属性。

FME Quick Facts	
Format Type Identifier	FME
Reader/Writer	Both
Licensing Level	Professional
Dependencies	Writer: GeoMedia installed
Dataset Type	File
Feature Type	Table name
Typical File Extensions	.mdb
Automated Translation Support	Yes
User-Defined Attributes	Yes
Coordinate System Support	Yes
Generic Color Support	No
Spatial Index	Optional
Schema Required	Yes
Transaction Support	No
Enhanced Geometry	Yes
Geometry Type	Geo_Type

Geometry Support			
Geometry	Supported?	Geometry	Supported?
aggregate	yes	point	yes
circles	no	polyline	yes
circular arc	yes	raster	no
donut polygon	yes	solid	no
elliptical arc	no	surface	no
ellipses	no	text	yes
line	yes	z values	yes
none	yes		

千万不要忘了进行数据检查，以确保输出是正确的。”

几何属性结构

有时候，因为几何属性出现错误，或是特定格式要求的结构规则，就要求 FME 来对空间数据进行转换。

对于部分几何属性错误在读取数据或者写入数据时，就能够修复。

例如，Oracle 数据库就规定，一条线或一个多边形上是没有两个顶点完全相同的。FME 会自动检查数据，确保它符合这个规定。

Limitations

我们应该强调，FME 对数据进行处理时，不会改变一个要素的整体形状，面积，长度或尺寸。所以，FME 能自动修复的错误并不属于这一类会影响形状，尺寸的错误。

以下是 FME 可能自动修复的一些错误：

- 在一个线类要素上多余点
- 在一个多边形要素上多余线段
- 多边形定向（顺时针，逆时针）
- 扭曲的，或者是含有八边的多边形
- 面包圈多边形的内部边界与外部边界相交多于一个点。

在这些情况下，唯一可以看到的不同点就是数据顶点数量的变化。

有时候错误太多了，FME 不可能自动将它们全部修复，这时，用户就需要使用函数，例如，SelfIntersector 函数，来寻找并且解决特定的问题。

FME 支持的属性

一些格式既不支持，也不能在数据集中储存属性，这种情况下，FME 目标模式便没有 user attribute，并且在 Feature Type Properties 对话框中不会出现‘user attributes’。

Right:例如， MicroStation Design 格式，你可以看到，在源 Feature Type Properties 中没有自定义属性这一栏。



Mr CAD 说过...

“FME2009 通过自动创建，并写入属性值到 MicroStation Tags 中，从而支持 DGN (V8) 的目标属性。”

Attribute Names

许多格式对属性名的结构和格式都有限制，例如，ESRI Shape 就只能使用大写字母作为属性名字，并且要求属性名不能超过十个字符。

当创建一个工作空间时，如果有需要的话，在目标模式中 FME 会将属性名自动转换为大写字母形式，这种情况下，就不能进行属性连接，所有 FME 会进行模式映射，将源属性和目标属性连接起来。

这只是工作的开始，编辑工作空间可能会打乱 FME 已创建的模式映射，所有你需要创建你自己的属性连接。

Left: 在这个转换过程中，FME 自动创建了使用大写字母为属性名的目标属性，并且将属性映射到源模式。

如果你在源属性和目标属性之间插入一个函数，则可能会失去模式映射。



属性字段的长度

一些格式限定属性的大小，例如，限定字符的最大长度，或是数字的最大值。FME 会自动创建一个目标模式，这个模式就会反映出这种限定。

写入超过限定的数据

如果认为要写入超过限定大小的数据，就需要终止 FME 操作的话，这种想法是不正确的。写模块会决定是否要对数据进行裁剪，或是忽略数据，使它符合限定。当由于目标模式的限制影响到属性数据时，日志文件就会发出错误警告；虽然如此，你不应该盲目相信日志文件，如果你觉得有问题的话，就应该自己检查输出数据

属性类型

不同的格式支持不同的属性类型。即使当两种格式支持同一种属性类型时，属性的名称也是不同的。这种情况下，FME 就需要决定怎样在不同属性间映射数据。例如，在 Oracle 当中，与 MapInfo 'decimal' 表示同一种属性的类型，是哪一个呢？

FME 凭借自身丰富的数据模型支持所有的属性类型，这样就能解决这个问题了。源格式的元文件会指示 FME 将源数据属性类型映射到 FME 模型中，然后第二个元文件指示 FME 将 FME 模型类型映射到目标模式。

例如，对数据 MapInfo 的属性类型进行映射

```
char(width) fme_char(width) date fme_date decimal(width,decimal) fme_decimal(width,decimal)
float fme_real64 integer fme_int32 logical fme_boolean smallint fme_int16
...and here is the attribute type mapping for Oracle data:
char(width) fme_char(width) float fme_real32 number(width,decimal) fme_decimal(width,decimal)
double fme_real64 integer fme_int32 logical fme_boolean smallint fme_int16
```

通过映射，MapInfo 的小数点形式的属性在 FME 数据模式中会转变为 fme_decimal' 形式的属性，然后在将属性写入到 Oracle 中时，将 fme_decimal' 形式转换为“数字”类型的属性。

单元总结



这一单元向你们讲解了 FME 是怎样处理不同的空间数据格式的。

从这一单元中，你应该学到些什么呢？

下面就是你要从这个单元中学到的主要内容：

理论

- FME 支持的格式随着许可证要求，数据类型，几何属性以及属性而改变。
- 一个工作空间中的对象有着不同的等级，Readers 和 Writers 控制着任意数量的数据集，数据集又包含任意数量的要素类。
- 读写模块参数以及格式属性控制着转换过程，这和 Reader-Dataset-Feature Type 的等级结构相似。
- 已发布参数提示用户在进行转换的时候，设置一个参数值。
- 格式属性保存与要素结构和符号相关的信息，使用它就能够过滤源要素，或重组目标要素的结构。
- 有需要时，FME 能够调整输出的几何属性，然后进行语义转换。

FME 技能

- 使用读取和写入参数来控制转换过程
- 在源要素和目标要素中能够调整应用格式属性
- 当转换不同格式时，能够找到影响要素的方法
- 使用 Generic Writer 和 “Feature Types to Read”参数，创建通用工作空间
- 能够从读模块，写模块以及函数中发布参数

问与答



在例 2 中，我们问到：

为什么在线的交汇点不能够使用 LineJoiner 来进行连接？

这是因为 FME 不能够告诉你哪些线是不能够连接的，所以，它就直接不连接，而不是进行错误的连接。

我们可能也会问到：

- 为什么重点标记的交叉点不同呢？
- 为什么 Geometry Handling 模式也是不同的呢？

答案就是，不要看图形的形状，这并不是四个线类要素的交汇地，你需要仔细检查数据，就会发现其中有两个是弧形要素，而不是线类要素。

在一般模型中，FME 是不能够连接弧形和线的，所以，FME 只会连接两条线。

在加强模型中，FME 能够连接弧形和线，所以，我们我们就要再回到之前提到的那个问题——当 FME 不能够决定要连接哪些线的时候，就不会进行任何连接。

在改变几何属性模式时，就会显示不同的类型，有时候，你是很难理解这个改变的。