

## 目录

更多的转换函数.....	2
Adders, Builders 和 Creators: 基本的转换函数.....	2
空间（位置）函数.....	5
函数类型.....	5
函数查找.....	6
函数帮助（Help） .....	6
使用函数.....	7
输入函数设置.....	7
AutoConnect .....	8
高级函数——Drag-and-Insert .....	9
函数与流的相互影响.....	12
Interacting Edits .....	12
断开工作流，对它进行检查.....	12
用户自定义的 Workbench .....	13
自定义 Workbench 的工具条.....	13
自定义 Transformer Gallery.....	14
基本的自定义函数.....	16
什么是自定义函数?.....	16
创建一个自定义函数.....	17
编辑一个自定义函数.....	17
插入或是连接?.....	18
单元复习.....	19
你应该从这单元中学到些什么? .....	19
函数的常用操作.....	20
处理 GPS 数据 (以 CSV 格式).....	20
数据检查.....	20
处理数据/重新编写符合体系 .....	20
疑难解答.....	21

## 更多的转换函数



许多 FME 用户认为有必要对函数进行培训。这单元会介绍更多的函数，以及一些你之前可能不知道的高级函数。

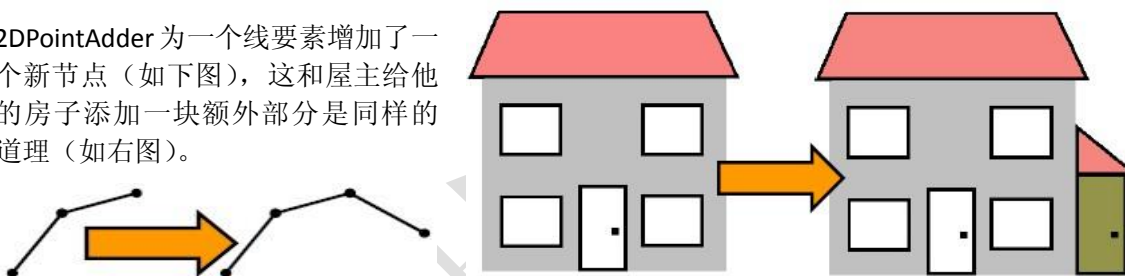
### Adders, Builders 和 Creators: 基本的转换函数

一般，函数根据一些相关的信息来命名，例如运行方式、输出结果等。我们可以通过函数的前缀来寻找线索。

#### Adders 函数

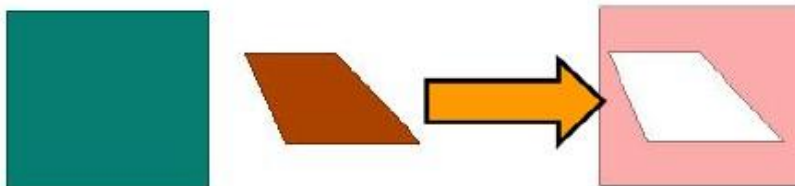
以“Adder”作为后缀的函数，会添加一个新项目到当前要素中。例如，2DPointAdder 函数表示添加一个新的节点（点）到已有的要素中。

2DPointAdder 为一个线要素增加了一个新节点（如下图），这和屋主给他的房子添加一块额外部分是同样的道理（如右图）。



#### Builders 函数

“builder”函数是将一组输入数据重构成一种新的格式，你可以把它看成是原始资料回收站。



输入的面

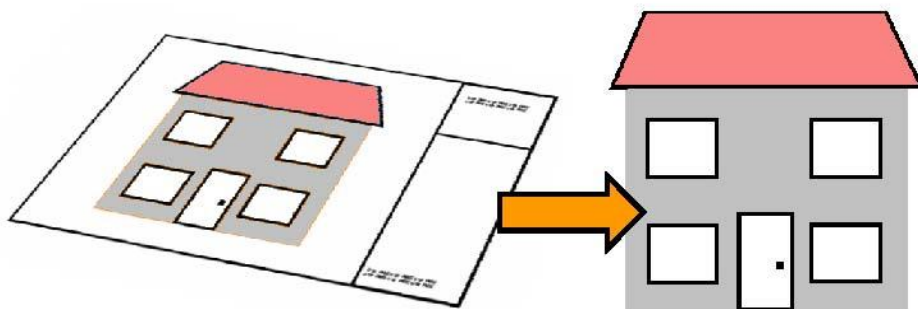
输出结果

这就像是，我们摧毁一个房子，然后重新建造它（如上图）。PolygonBuilder 处理一组多边形，然后输出新的格式（如左图）。

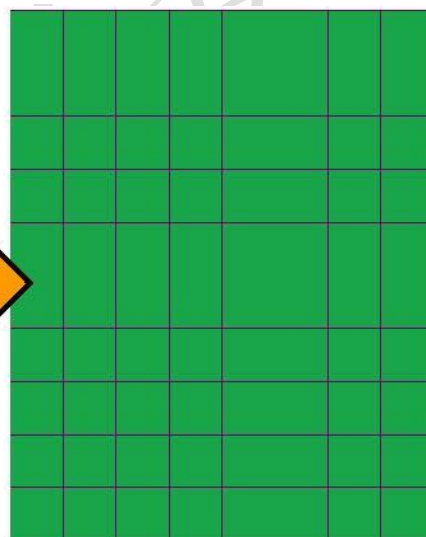
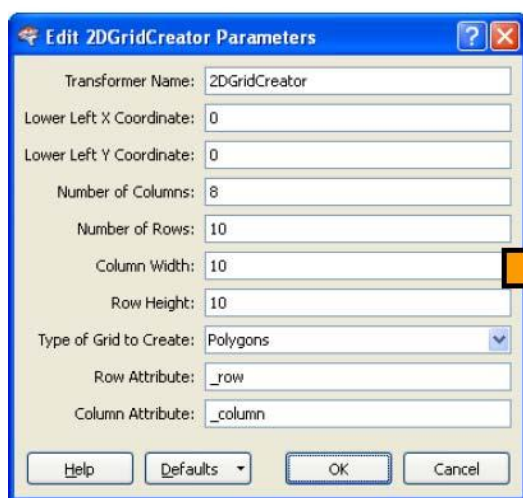
## Creators 函数

“creator” 函数会在工作流中创建一个全新的实体

多数这类函数都不接受输入项，而是通过设置对话框中的信息创建一个全新的实体，例如，2DGridCreator 函数，用户通过输入的一系列参数值，就能够创建一个网格点或面。



这就好比建房子，我们使用原料来建房（如上图）。2DGridCreator 函数则使用一系列用户自定义的参数来创建面网（如下图）。



一些 creator 函数会为已有的要素创建新的属性项，例如 AttributeCreator 函数，任何输入到这个函数中的要素都会添加一系列的新属性，这些属性都是在函数设置对话框中定义的。

其它的 creators 的函数还有：Creator，PythonCreator 和 RasterNumericCreator。

在对函数名知识了解的基础上，推测下 TopologyBuilder 函数是用来做什么的呢？

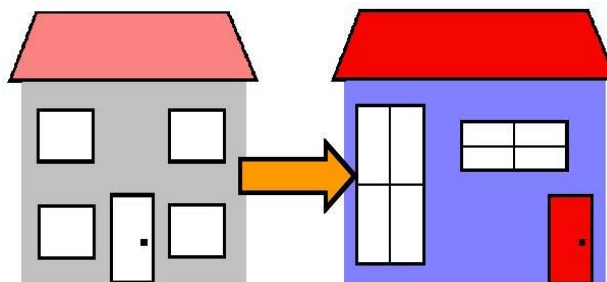


- 1) 在设置对话框中创建参数的拓扑
- 2) 将输入要素转换成拓扑连接的输出要素
- 3) 用拓扑要素代替输入要素
- 4) 将相关拓扑属性插入要素中

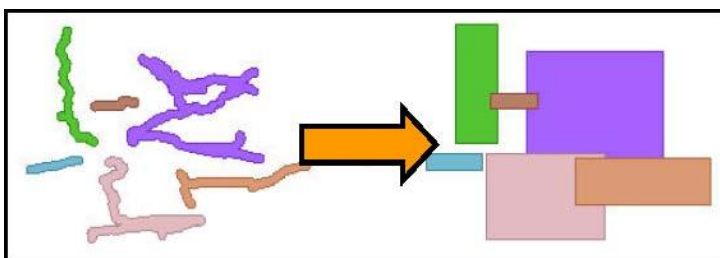
## Replacers 函数

任何以“Replacer”为后缀的函数都会进行以下操作：

输入到该函数中的要素会被一个新要素实体取代；新的要素以第一个要素为基础。



房主决定重建他的房子，但要以一种全新的风格，只有一部分是在原来的基础上（如上图）。



BoundingBoxReplacer 是一个典型的 replacer 函数（如左图）。

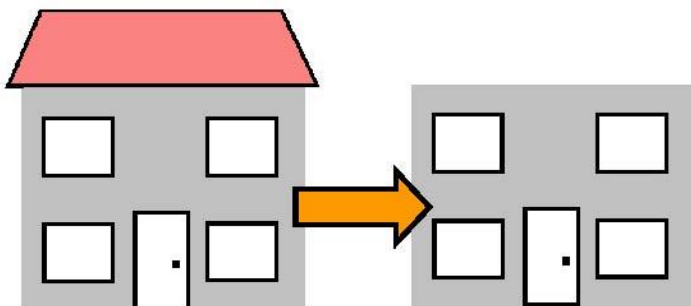
所有的输入要素都被输出要素所取代，这些输出要素表示输入要素的边界框

其它的 Replaces 函数还有：2DPointReplacer, CenterPointReplacer 和 GeometryReplacer。

## Removers 函数

“remover”函数会删除要素的部分属性，输出结果都是修改后的要素。

例如 GeometryRemover 函数，就是删除要素的所有几何图形，输出一个非几何要素，类型是 fme\_no\_geom。



左图：这是房主使用 remover 函数移除了屋顶的效果。

其他的 Remover 函数还有：

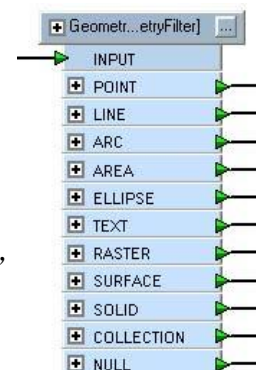
- AttributeRemover
- CoordinateRemover
- CoordinateSystemRemover
- DuplicateRemover
- ListDuplicateRemover

## Filters 函数

“Filter”函数是根据原要素特征进行数据过滤分流，实现结构映射。这些要素特征可以是属性，或者是要素几何类型的某方面。

右图：GeometryFilter 函数根据几何类型将输入要素过滤成多个输出流。

其它的过滤函数还有：AggregateFilter, AttributeFilter, FeatureTypeFilter 和 SpatialFilter(通常,我们容易将 SpatialFilter 与 SpatialRelator 弄混,“filter”强调的是区别)



## 空间（位置）函数



即使是有经验的 FME 使用者看到整个函数列表也会感到很头疼，学了这个单元，你就不再害怕函数，并且还会喜欢上函数。

FME 拥有 300 多个函数，能够实现各种各样的功能。可能有很多你都没有意识到，但这些函数真的是非常有用。这个单元将会帮您寻找需要的函数，即使有些您没有想到会需要它。

### 函数类型

首先，我们来学习函数类型，这是学习函数列表很好的一个起点。我们将同一种类型的函数分为一组，这样方便寻找能够解决问题的函数。

**右图：**图标显示的是函数类型，点击+，展开每个列表，显示一种类型的函数。

以下就是一些类型：

**3D：**特别针对 3D 数据的函数

**Calculators：**计算一个值，并且将它作为一个新的属性

**Database：**与外部数据库一起使用

**Filters：**过滤，重新发送数据

**Geometric Operators：**处理要素几何形

**Lists：**处理属性列表

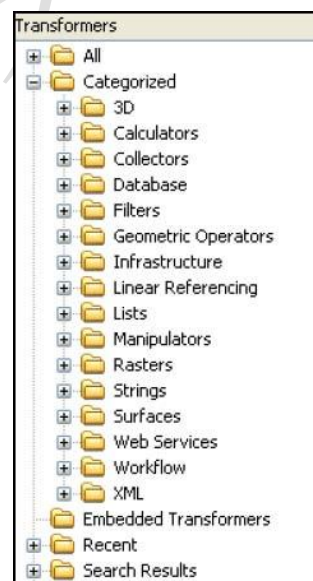
**Linear Referencing：**进行测量，例如，创建测链长度

**Manipulators：**处理单个要素

**Rasters：**处理栅格数据集

**Strings：**创建，调整和删除字符属性

**Surfaces：**处理面类要素，例如，创建一个等高线



**Web Services：**通过 HTTP，分享网站上的信息

**Workflow：**在本地，或者使用 FME Server 来运行工作空间 **New for 2009**

**XML：**将 XML 数据编写到 FME 中

## 函数查找

这个函数会尽力在函数名和函数描述中寻找用户自定义的关键词

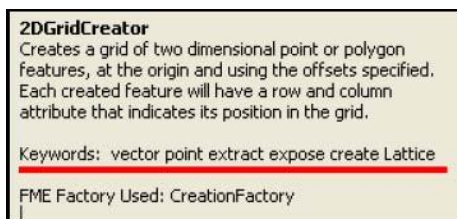
右图：在函数列表中就可以找到 **transformer search** 的窗口（被标记成红色）

简单地输入要术语，或者按<enter>键，或点击 **Search** 图标，就可以使用这个函数了

这个函数也接受含有词语的描述，所以就可以输入‘attrib’，来代替“attribute”

## 关键字（Keyword）查找

函数查找可以接受多个关键字，并且返回所有含有关键字的函数



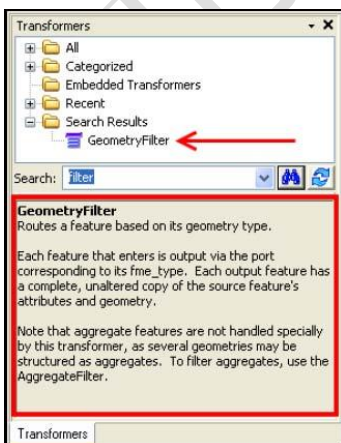
注意，你没有必要一定要输入函数名，或 FME 使用的术语。有关函数的描述包含一般的关键词（如左图），使用这些关键词就能更轻易地寻找到和空间相关的术语。



输入一个函数名的中间部分能更快地寻找到需要的函数，例如，为了快速地寻找函数 **AttributeReplacer**，你可能输入‘Attrib’或‘Replace’，但是含有‘Attrib’的函数有 150 多种，而含有‘Replace’的也有 50 种，如果输入‘uteRep’，就只会出现 2 个函数。可能在理解时有些困难，但是当你操作时，就胡发现它是非常有用的。

## 函数帮助（Help）

**Workbench Help** 会为每个函数提供大量的相关信息。它取决于语境，所有你只需要选择一个函数，然后按 **F1** 键，就可以获取帮助了。



也可以在函数中获取 **Transformer Help**，点击函数列表中的一个函数，就可以获取这个函数的具体信息了。

当输入的寻找关键字与函数名完全符合时，就会默认地选择这个函数，并且会自动显示相关的帮助描述

反之，就会在列表中显示第一个函数的相关帮助描述

左图：使用 **transformer search** 来寻找 **GeometryFilter** 函数，在下面的窗口中显示了对这个函数的描述。

## 使用函数



函数有一些我们不大知道的功能，使用这些功能就能帮助你更有效的工作。

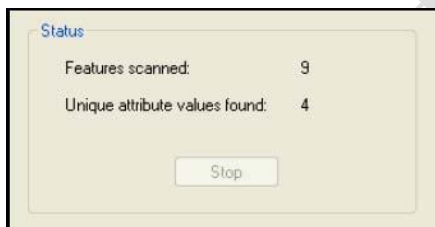
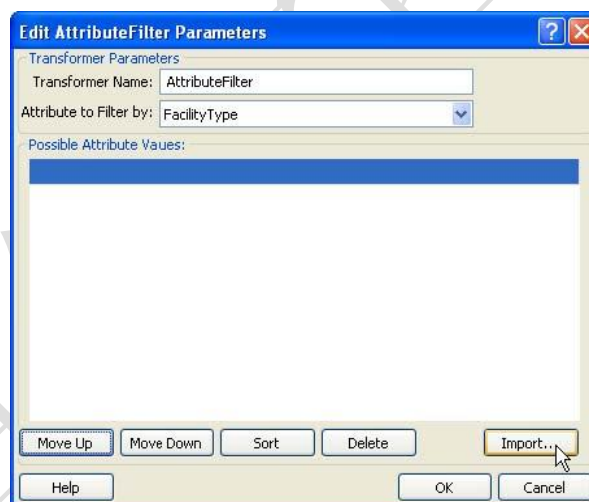
比起简单的忽略函数，或者手动的进行模式映射，这里提供了很多不同的方法，来放置，使用函数。

### 输入函数设置

一些函数需要用户输入大量的内容，这样就可能产生错误，并且使用起来也不是很方便。为了解决这个问题，一些函数就有一个‘Import’选项。

这个输入工具会在已有的数据集中输入一个属性值，并且数据集可能是任何的 FME 格式

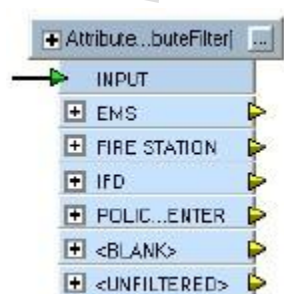
右图:在这个例子中，打开 AttributeFilter 设置对话框，然后点击‘Import’键。



左图：一旦选择了一个数据集和属性，FME 就可以扫描数据集，并且提供一个有关特殊值的列表

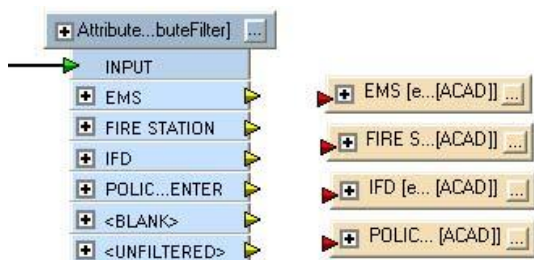
这个例子中，会对 9 个要素进行扫描，找到了 4 个特殊值

右图：??????



## AutoConnect

右击 **Filter** 函数，就会出现一个选项‘AutoConnect’，它能够自动将输出端口连接到正确的目标要素类。虽然我们很少用到这个选项，但是模式映射一个复杂的工作空间时，使用它就非常有用。



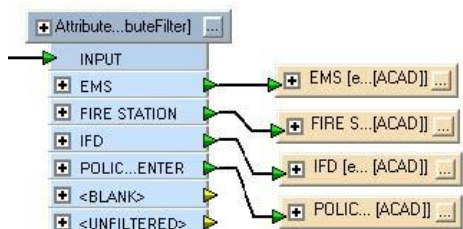
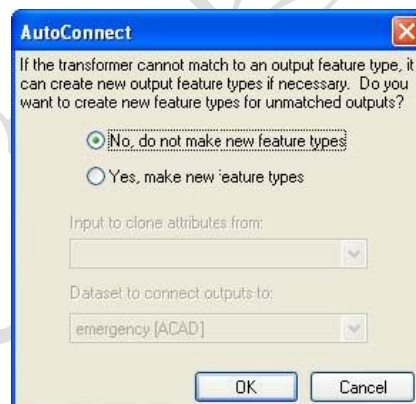
当必须要将过滤函数与大量目标要素类进行连接时，并且要素类名与输出端口名相同，使用 **AutoConnect** 就非常有效。

左图：将 **AttributeFilter** 连接到大量的目标要素类，最理想的方法就是使用 **AutoConnect** 选项。

右图：选择了 **AutoConnect** 后，就会打开这个对话框。

注意，**Auto-connect** 是怎样使用源要素的模式定义来自动创建目标要素类的。

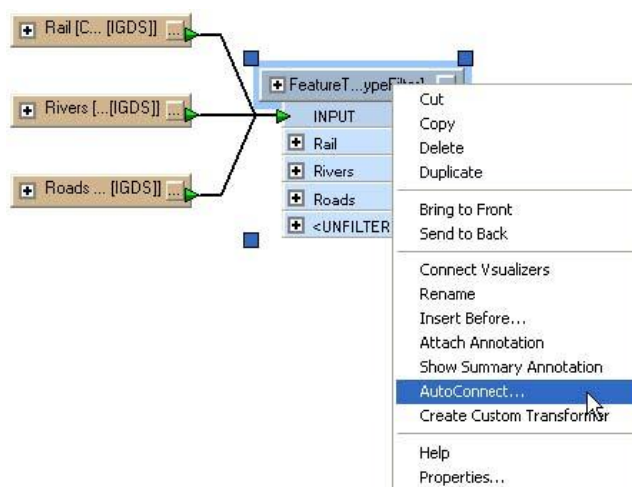
在这个例子中，要素类已经存在了，所以就不需要再创建新的。



左图：AutoConnect 将每个 filter 输出端口连接到相应的要素类。

自动连接函数还有：**AggregateFilter**, **SpatialFilter**, **GeometryFilter** 和 **FeatureTypeFilter**。

如右图：右击 **FeatureTypeFilter**，就能够运行 **AutoConnect** 功能。



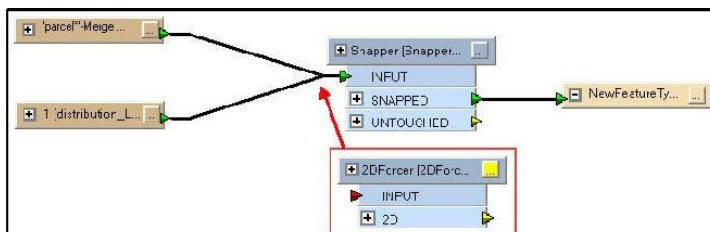
## 高级函数——Drag-and-Insert

这个函数拥有大量的高级功能。

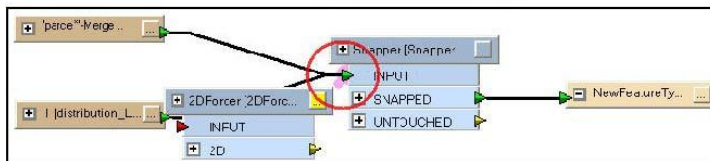
### 多个连接

你可能知道，将函数拖到已有的连接中，然后松开鼠标，就能够将函数插入到一个管道中了，但是你又是否知道，通过标记一个函数输入/输出端口，而不是进行连接，就可以立即将函数插入到多个管道中呢？

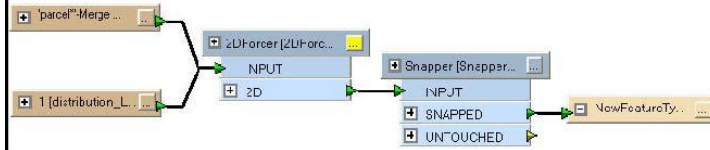
右图：这里，用户希望插入一个 2DForcer 函数，并且将它连接到输入流中...



...他将函数拖到 Snapper 输入端口（被标记为红色）...



... 这样，就插入了这个函数。

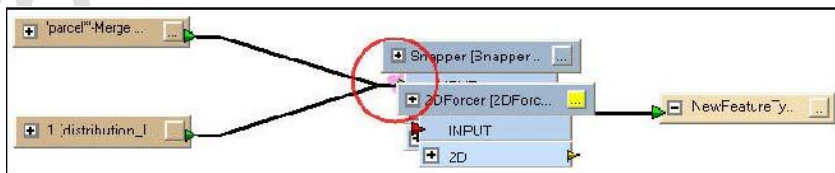


### Reversing the Insertion Point

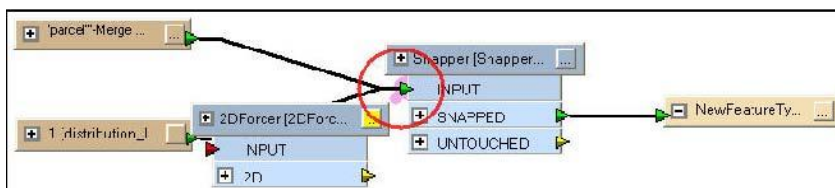
有时候，将函数拖到指定的位置，并不是那么容易的，这是因为函数覆盖了要插入的连接或端口

当你拖动函数时，按 **ALT** 键就可以解决这个问题。通过这个操作，就能将插入醒目点转换到函数的对角点，这样就能明确标记出要输入的端口了。

右图：当醒目点在常用位置时，2DForcer 就会阻碍用户的视野...



...但是通过 **ALT** 键降低点命名，就不会出现上面的情况了。

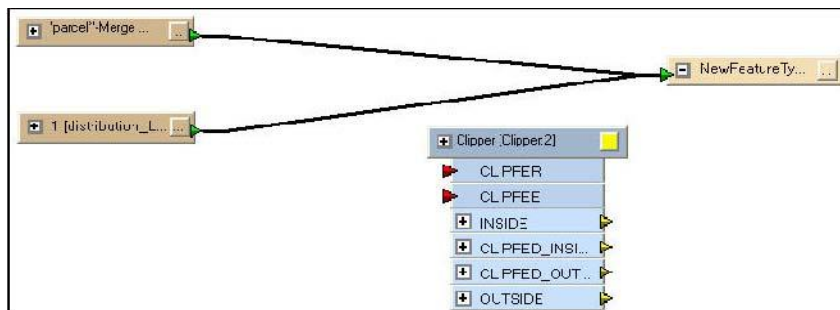


### 插入一个已插入的函数

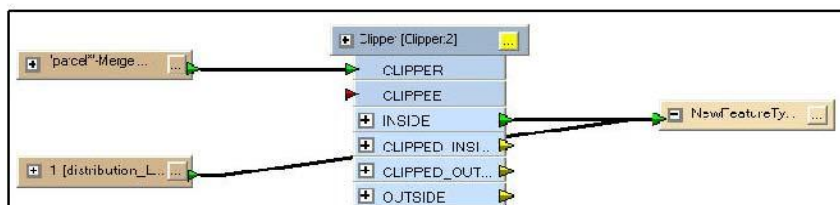
你可能注意到了，拖动一个已经插入的函数仍然会产生醒目点，并明确标记连接，这是因为能够将一个单一的函数连接到多重流，而这种方法能够轻松的帮到你。

右图：用户希望插入一个 Clipper 函数

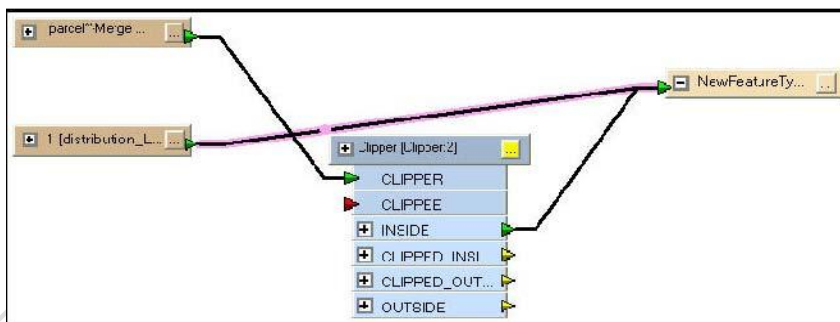
与前面多重连接不同的是，这里有多多个要求进行单独连接的输入端口



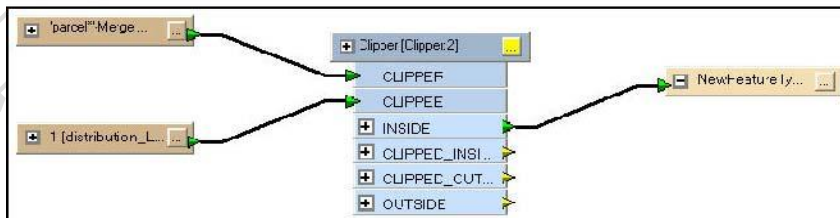
...使用拖动和插入功能，可以很轻易地进行第一个连接...



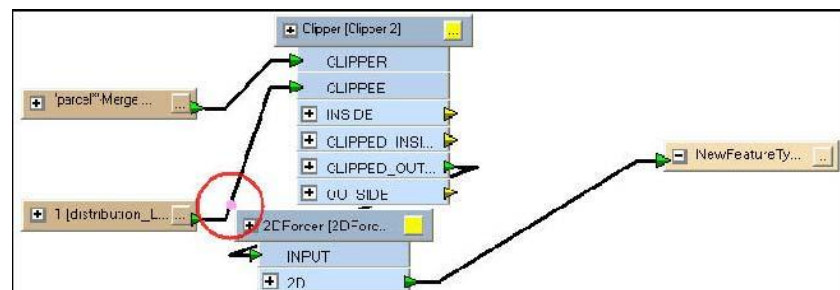
... 现在用户要进行第二个插入了，使用同样的函数，但是通过另一个连接...



... 现在就正确地插入了 Clipper



右图：注意，不能将函数插入到一个可能引起工作流循环运行的位置——甚至不应该标记这种无用的连接

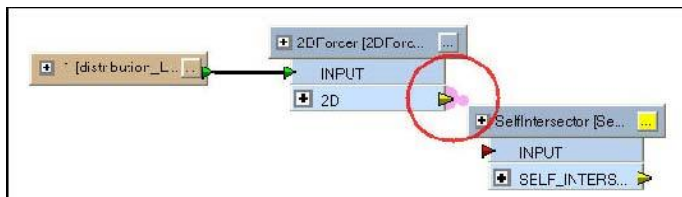


## 连接一个函数

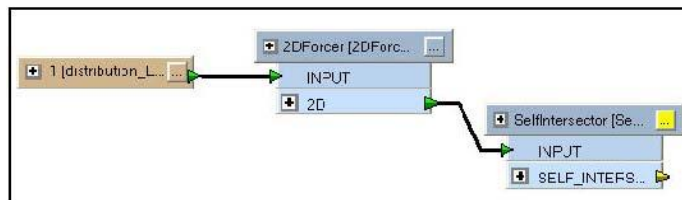
连接一个函数，而不是将它插入到一个已有的连接中，这并不是函数性的一部分，但是这是产品偶然获取的一种功能，当用户使用这种功能时，这种偶然性就变得非常明显了。

我们非常喜欢这个想法，所以有可能在将来的 FME 版本中开发这种功能

右图：在这里，用户希望在现在使用的管道的末端添加一个函数，他将函数拖动到最后一个输出端口...



...放开鼠标，就按要求将这个函数连接到末端了。



### 例 1：使用 Auto-Connect 来创建一个目标模式

Interopolis 城市有一个数据集，它包含了这个城市的所以公园。  
城市希望将数据集分为多个要素类，每个要素类表示一个公园。

启动 Workbench，添加包含城市公园的数据集，并将它作为源数据集，它是 MapInfo TAB 格式。源数据位于 C:\FMEData\Data\Parks\city\_parks.tab

放置一个 AttributeFilter 函数，并将它连接到源要素类。

使用 AttributeFilter 的输出功能。选择公园数据集作为源数据集，并且使用属性“name”进行输入。

观察 AttributeFilter 函数现在是怎样包含这个城市的公园列表的。

添加一个 MapInfo TAB 目标数据集（与源文件夹不同）。但是不要添加一个要素类，右击 AttributeFilter，选择 AutoConnect

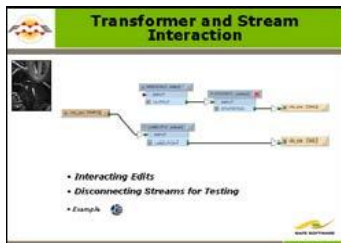
在 AutoConnect 设置中，选择从源数据中复制属性，并且选择创建新要素类的选项。

观察，每个输入是怎样在值后对要素类进行特别命名的。

删除目标数据集以及要素类。

右击 AttributeFilter，选择‘Connect Visualizers’，运行工作空间，确保 AttributeFilter 的每个输入发送到了一个单一的 Visualizer

## 函数与流的相互影响



当平行地使用函数时，工作空间很快就会变为一系列工作流，并且挨个执行不同的任务。我们需要处理和检查不同流之间的相互作用。

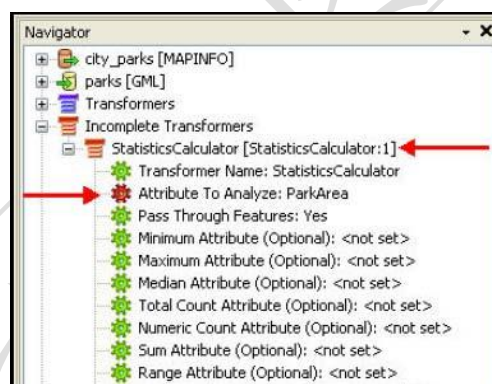
用户能够分别检查工作空间的每个部分，这是很重要的，但是也要知道改变一个部分就会影响到其它部分，甚至是整个工作流

### Interacting Edits

有时候，改变一个函数就可能导致另一个函数的设置错误，这时，函数就会要求用户在使用设置时，对这个设置进行编辑。

当变化对函数起到反作用时，Workbench 就会自动标记这些函数，所以用户就知道出现了错误。通常当函数 properties 按钮出现红色，就说明要引起注意了。

在 Navigator 方框中，同样会出现这种提示。



### 断开工作流，对它进行检查

当一个工作空间是由多个显著的部分组成时，通过断开连接，就能够简单地分裂组成部分，这样就能检查工作空间的特殊组成部分

因为函数之间是相互影响的，所以断开连接可能导致函数在断开的部分变得无效。

通常，FME 会在运行的时候检查工作空间，并且要求为任何不完整的函数提供设置，即便如此，这个功能也能判断是否将这些函数连接到要素流中。

如果不存在连接，就会忽略掉这些函数，并且也不会出现警告。

这个功能允许用户在不需要考虑已断开函数设置的情况下，就能够分裂工作空间的组成部分。

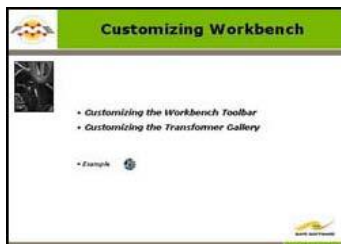


### 例 2：函数和工作流的相互影响（使用例 1 中的工作空间）

删除源要素类和 AttributeFilter 函数之间的连接，你会发现，AttributeFilter 函数现在被标记成“不完整”，为什么呢？

运行工作空间。注意到仍然在进行转换。FME 知道能够安全地忽略这些不完整的函数，这是因为它还没有被连接到源要素流

## 用户自定义的 Workbench



使用大部分 Workbench 的自定义界面就能够更轻易地访问到你最爱的函数

### 自定义 Workbench 的工具条

现在我们就自己定义 Workbench 工具条，这样就能定义一些功能的快捷键，以及常用的函数

右击工具条，然后选择“customize”选项，就会出现 Edit Toolbar 对话框（如右图）



### 添加工具条键

点击工具条的输入列表，然后将它拖到工具条中的指定位置，就能够在工具条中添加一个指令或函数



左图：用户在已有的 Run 和 Stop 按钮之间添加了一个键。

### 移除工具条键

为了移除一个指令或函数，就只需要将它拖到工具条外

### 移动工具条按钮

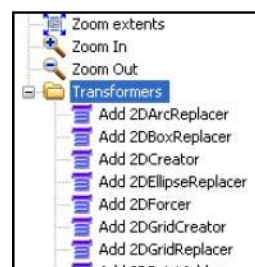
为了在工具条中移动一个指令或函数，就只需要将它拖到一个新的位置

Transformers on the Toolbar 函数出现在 Edit Toolbar 对话框的底端（右图）



左图：已经在这个工具条中添加了大量的函数快捷键，其中一个就是 Clipper

注意，所有快捷键的图标都是相同的，FME 之后的版本可能会提供一组自定义图标来区分它们。



## 自定义 Transformer Gallery

因为 FME Workbench gallery 含有大量不同的函数，所以用户一定要能够合理地安排这些工具，这样才能有效地运用它们。

什么是自定义函数文件夹？

Workbench 函数列表中的函数被合理地保存在大量文件夹中，文件夹数量越多则说明要进行不同类型的操作

FME 能够创建自定义函数文件夹；函数列表中的文件夹包含了任何已有的函数

它的优点就是，现在用户用一种方法来轻易地访问他们最常用的函数，再也不需要浏览这个函数列表或是使用函数寻找工具了。



Doctor Workbench 说过

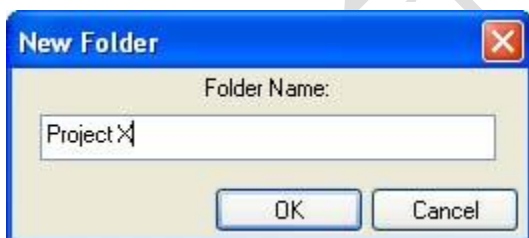
“不要搞混了！自定义函数文件夹指的是保存函数的自定义文件夹，而不是自定义函数的文件夹

创建一个自定义函数文件夹

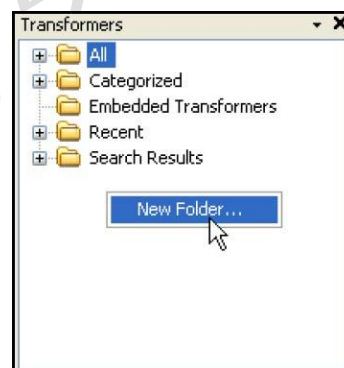
这是创建一个自定义函数文件夹的简单方法

首先，在函数列表中，鼠标右击，然后选择 **New Folder**”选项（如右图）

就会出现 **New Folder** 对话框



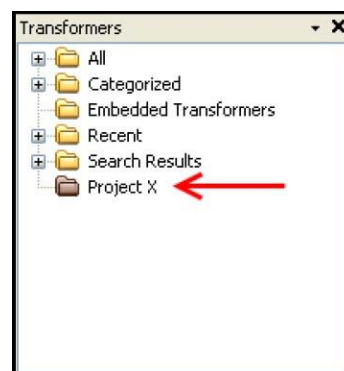
左图：用户创建一个新的文件夹，来保存在 ProjectX 中要用到的函数



右图：Project X 文件夹具惠出现在函数列表中

**重命名或者删除一个自定义函数文件夹**

在函数列表中，右击一个自定义函数文件夹，就会出现选项：重命名或删除

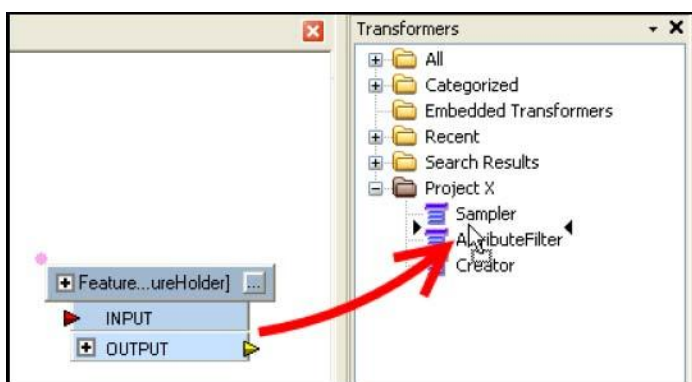


## 添加函数到一个自定义文件夹

进行简单的拖动就能将函数添加到一个自定义文件夹中。可以从函数列表的一个已有入口，或是从 **Workbench** 窗口中，对函数进行拖动。

我们一定要注意，在大多数情况下，在自定义文件夹中的入口只是一个副本。从工作空间或一个已有的文件夹中拖动一个函数并不会删除掉这个函数，但是，从一个自定义文件夹中拖动函数就会出现例外，假如这样的话，就是将函数从一个文件夹中转移到另一个文件夹。

你也要明白一个自定义文件夹并不是独立的版本。无论将函数添加到哪一个版本，任何插入到工作空间的函数都是最新的版本。



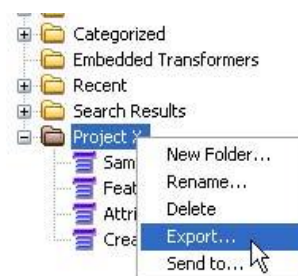
左图：Project 文件夹已经包含了 Sampler, Creator 和 AttributeFilter, 现在用户要添加一个 FeatureHolder 函数

在这个例子中，箭头表示要放置函数的位置，通过这种方式，用户就能够控制函数在文件夹中的次序了

## 分享一个自定义函数文件夹

我们可以使用\*.fmxlist.扩展名来保存自定义函数文件夹。用户可以分享这些文件，这样就能轻易地访问相同的函数，例如，所有为这个工程服务的用户就都能够创建，并且分享一个有关工程的文件夹。

有两种方法来分享一个自定义函数文件夹。右击一个自定义文件夹，就会出现选项“Export...”和“Send To...”，前者简单地编写一个 fmxlist 文件到你要求的位置，后者系统的默认邮件客户端，并且创建一个带有函数列表文件的新邮件，并将邮件作为附录。



### 例 3： Custom Transformer Folder

Create a transformer folder called “For Example 4” Add to it the AreaCalculator and StatisticsCalculator transformer.

## 基本的自定义函数



使用它们的功能，自定义函数是一个非常有用的工具，但是，仅仅是处理一些基本的情况

### 什么是自定义函数？

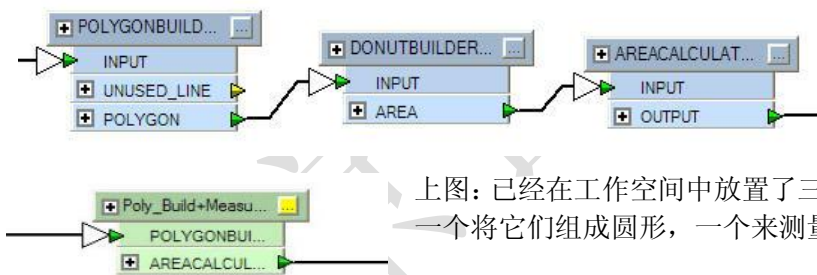
他指的是一组有次序的标准函数，可以压缩成一个单一函数。任何一组函数都能够转换成一个自定义函数

使用自定义函数就能帮助我们：

- 1) 将工作空间中的大量内容转移到下一页，就能够整理工作空间
- 2) 在多个地方，有效地再次使用同一组有次序的函数

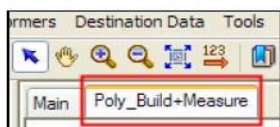
这两种功能都是针对 **Best Practice**.

下图：这个简单的例子显示了，一个自定义函数怎样代替大量的函数，整理工作空间



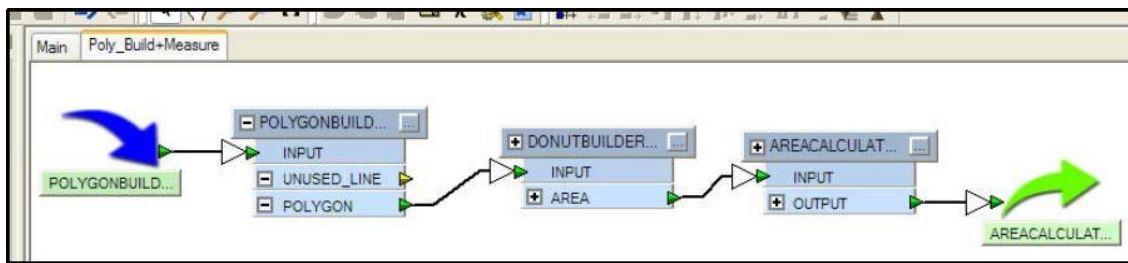
上图：已经在工作空间中放置了三个函数；一个来创建多边形，一个将它们组成圆形，一个来测量已创建的要素

左上：已经将这三个函数压缩为一个单一的自定义函数



左下：FME 在工作空间中创建了次页

下面：在次页中包含了三个之前的函数，并且组成了自定义函数。蓝色和绿色的箭头分别表示它的输入和输出口

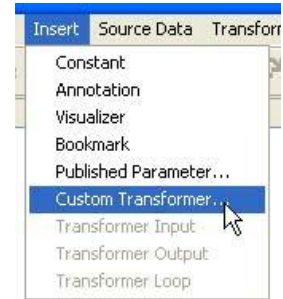


## 创建一个自定义函数

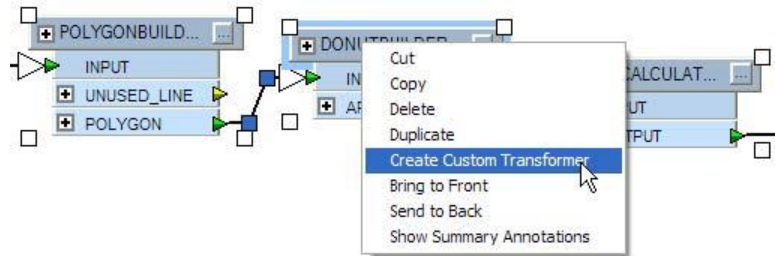
有很多方法可以创建一个自定义函数

使用菜单条中的 **Insert > Custom Transformer** 就可以创建一个空的自定义函数，或者右击空白处，然后选择 **Insert Custom Transformer** 选项

选择一个函数，右击它，然后选择 **Create Custom Transformer** 选项，就可以在这个函数的基础上创建一个自定义函数



右图：从一组已有函数中创建一个自定义函数



左图：All LINKED 会被自动放置在 Transformer Gallery (under ALL), 所以可以像选择常规函数一样，选择它们，然后多次运用在其它地方

正在运行工作空间中的所有自定义函数 EMBEDDED 都会显示在一个单独的 Embedded Transformers 分类中

这是因为两个相同名称的自定义函数可以同时存在— 一个是连接的，一个是插入的

## 编辑一个自定义函数

一旦创建了自定义函数（通过点击相关按钮），就可以根据需求编辑它，就像是编辑一个工作空间。记住，这个方法适用于所有的自定义函数，如果多次使用，这种编辑方法就会适用于任何函数

因为可以使用 **Nested** 自定义函数，所有就能够在另一个函数中创建一个自定义函数。

## 插入或是连接?

自定义函数有两种形式：插入和连接

插入是自定义函数的默认状态，也就是说，函数被保存为工作空间的一部分，所有其他用户不能够使用它

连接函数的定义就是，将它保存在一个外部文件中；要从工作空间中查询函数就需要连接到这个定义，并且如果这个定义改变了，工作空间也发生了变化。使用输出函数就能够与其它用户分享这些函数

使用工具条中的 **File > Export as Custom Transformer** 的选项，就能将已插入函数输出到文件。右击一个连接函数，选择 **Embed.**，就能插入这个函数了



First Officer Transformer 说过...

“一旦插入了一个自定义函数，就不能够再连接它了，也就是说，一旦编辑了一个插入函数，连接选项就会消失了。给你一个建议—当你处理下面这个例子时，就一定要考虑到这一点！”



### 例 4: AverageAreaCalculator Custom Transformer

在之前的例子中，按照要求你已经计算了每个公园的面积，因为这个工具非常有用，所有你可以创建一个有关它的自定义函数。

启动 **Workbench**，创建一个工作空间，将 **city\_parks.tab** 转换为 **GML** 格式

找到并且放置一个 **AreaCalculator** 函数，**StatisticsCalculator** 函数，然后有次序地连接它们到源要素类和目标要素类。

注意：如果你进行了例 3 的操作，已经将这些函数保存在了一个自定义文件中

打开 **transformer properties** 对话框，按照要求调整设置 **StatisticsCalculator** 会将 **'Pass Through Features'** 设置成 **'Yes'**。

选择 **AreaCalculator** 和 **StatisticsCalculator** 函数，右击它们，然后选择 **'Create Custom Transformer'**。在 **Transformer Properties** 对话框中，将函数名设置成 **AverageAreaCalculator.**，按照要求输入类型以及描述

这样就创建了一个自定义函数，点击 **'Main'**，返回到主菜单中，确认自定义函数现在呈绿色

返回自定义函数定义，从菜单中选择 **File > Export as Custom Transformer**，就能够使用 **FME** 新的功能，它包括函数定义。观察怎样将它命名为一个 **.fmx** 文件，如何使用 **Windows Explorer** 来寻找文件

返回到之前的工作空间，观察现在 **AverageAreaCalculator** 以何种方式出现在函数列表中（点击更新键）

在工作空间中寻找自定义函数，它应该仍然显示成绿色，然后右击这个函数，就会出现多个选项。重新操作，将插入改成连接，而不是将连接改成插入。

## 单元复习



这个单元介绍了更多的 FME 函数，以及更加有效地使用这些函数的方法

你应该从这单元中学到些什么？

以下就是你应该学到的主要内容：

### 理论

- 函数名通常显示了希望从中读取的输出类型
- 一个自定义函数指的是，将一组连续函数压缩成一个单一实物，它能够反复使用，能够被分享
- 自定义函数要么是插入的，要么是连接的，采用哪种方式去决定于它们是来自于工作空间，还是一个外部文件

### FME 功能

- 在不需要提前知道函数的情况下，能够寻找一个函数，来进行特别的操作
- 能够自己定义 Workbench 工具条以及 Transformer Gallery
- 能够将一组已存在函数转换为一个自定义函数
- 能够创建一个外部的自定义函数，在插入和连接方式之间进行切换

## 函数的常用操作



我们可以使用函数来进行许多操作，以下是你能进行的一些特殊操作

这个单元允许你自己选择要学习的内容

你可以选择进行一个或所有的操作，或者是在你自己的数据集中使用一个函数

### 处理 GPS 数据 (以 CSV 格式)

许多 GIS 版本含有限定文本形式的空间数据，例如，一个 GPS 接收器中的输出通常就会采取这种方式。你可以使用 FME 的 CSV 阅读器以及一组函数，来将这个数据转换为一个更合理的 GIS 格式

### 数据检查

通常我们会从非电子版本的数据库（例如，文本记录）中获取空间数据以及属性，这样在获取的过程中就会产生许多问题和错误。可以使用 FME 来检查数据，也就是进行数据检查

### 处理数据/重新编写符合体系

重新编写符号体系是一个常见的转换要求。可以使用 FME 来处理一个数据集，产生一个含有相关符号的输出。

## 疑难解答



Based on your knowledge of transformer names what does the TopologyBuilder transformer do?

- 1) Creates topology from parameters in a settings dialog.
- 2) Turns inputs features into topologically connected output.
- 3) Substitutes input features with topology features.
- 4) Assigns topology related attributes to features.

Answer – 2

- 1) No – this would be a TopologyCreator (if it existed).
- 2) Yes – as a Builder this transformer restructures the input data into a new form.
- 3) No – this would be a TopologyReplacer (if it existed).
- 4) No – this would be a TopologyAdder (if it existed).