You should be here
Filters in ParaView

- Filters
- Alphabetical
In preparation for the next section

You should be here

- Delete all objects in the Pipeline Browser
- Select an object in the Pipeline Browser
- Click the Delete button (or right click, then Delete)
- To select multiple objects press and hold the CTRL key while selecting objects
ParaView/Python Scripting
A short introduction to ParaView’s Python Interface
PARAVIEW/PYTHON SCRIPTING

Rich scripting support through Python.

Available

- As part of the ParaView Client (ParaView)
- An MPI-enabled batch application (pvbatch)
- The ParaView python client (pvpython) or
- Any other Python-enabled application

Using Python, users and developers can gain access to the ParaView engine called **Server Manager**
• Library
• Designed to make it easy to build distributed client-server applications
Open Python Shell:
→ Tools
→ Python Shell
Create a Cone Object:

```python
>>> cone = Cone()
```
Create a Cone Object:

```python
>>> cone = Cone()
```
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
```
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
```

This gives you the full list of properties.
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
```

Check what the resolution property is set to:

```python
>>> cone.Resolution
```

```python
6
```
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
>>> cone.Resolution
```

You can increase the resolution:

```python
>>> cone.Resolution = 32
```
Create a Cone Object:
>>> cone = Cone()
>>> help(cone)
>>> cone.Resolution

You can increase the resolution:
>>> cone.Resolution = 32

You could have specified a value for resolution when creating the object
>>> cone = Cone(Resolution=32)
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
```

You can assign values to any number of properties during construction using keyword arguments:

```python
>>> cone.Center
[0.0, 0.0, 0.0]
```

```python
>>> cone.Resolution
6
>>> cone.Resolution = 32
>>> cone.Center
[0.0, 0.0, 0.0]
```
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
>>> cone.Resolution
>>> cone.Center
>>> cone.Center = [1, 2, 3]
```
Create a Cone Object:
>>> cone = Cone()
>>> help(cone)
>>> cone.Resolution
>>> cone.Center
>>> cone.Center = [1, 2, 3]
>>> cone.Center[0:2] = [2, 4]
>>> cone.Center
[2.0, 4.0, 3.0]

Vector properties such as this one support setting and retrieval of individual elements, as well as slices (ranges of elements).
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
>>> cone.Resolution
>>> cone.Center
>>> cone.Center = [1, 2, 3]
>>> cone.Center[0:2] = [2, 4]
>>> cone.Center
[2.0, 4.0, 3.0]
```

Apply a shrink filter to the cone

```python
>>> shrinkFilter = Shrink(cone)
```
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
>>> cone.Resolution
>>> cone.Center
>>> cone.Center = [1, 2, 3]
>>> cone.Center[0:2] = [2, 4]
>>> cone.Center
[2.0, 4.0, 3.0]
```

Apply a shrink filter to the cone

```python
>>> shrinkFilter = Shrink(cone)
>>> shrinkFilter.Input
```
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
>>> cone.Resolution
>>> cone.Center
>>> cone.Center = [1, 2, 3]
>>> cone.Center[0:2] = [2, 4]
>>> cone.Center
[2.0, 4.0, 3.0]
>>> shrinkFilter = Shrink(cone)
>>> shrinkFilter.Input
<paraview.servermanager.Cone object at 0x000000000896EEB8>
```
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)

>>> cone.Resolution

>>> cone.Center

>>> cone.Center = [1, 2, 3]

>>> cone.Center[0:2] = [2, 4]

>>> cone.Center
[2.0, 4.0, 3.0]

>>> shrinkFilter = Shrink(cone)

>>> shrinkFilter.Input
<paraview.servermanager.Cone object at 0x000000000896EEB8>
```

PARAVIEW/PYTHON SCRIPTING

CREATING A PIPELINE
Create a Cone Object:

```python
>>> cone = Cone()
>>> help(cone)
>>> cone.Resolution
>>> cone.Center
>>> cone.Center = [1, 2, 3]
>>> cone.Center[0:2] = [2, 4]
>>> cone.Center
[2.0, 4.0, 3.0]
>>> shrinkFilter = Shrink(cone)
>>> shrinkFilter.Input
<paraview.servermanager.Cone object at 0x000000000896EEB8>
```
Create a Cone Object:

```python
>>> shrinkFilter.UpdatePipeline()

33L

>>> shrinkFilter.GetDataInformation().GetNumberOfCells()

128L

>>> shrinkFilter.GetDataInformation().GetNumberOfPoints()
```
Create Cone Object
Set Cone Resolution
Set Cone Center Properties
Apply Shrink Filter to the Cone
Updated Pipeline
Two objects are needed to render the output

• **A representation** – takes a data object and renders it in a view

• **A view** – responsible for managing a render context and a collection of representations
Type at prompt:

```python
>>> Show(shrinkFilter)
<paraview.servermanager.UnstructuredGridRe
presentation object at 0x000000000BE85B70>

>>> Render()
<paraview.servermanager.RenderView object
at 0x000000000C26D278>

>>>```

PARAVIEW/PYTHON SCRIPTING

RENDERING
Type at prompt:

```python
>>> Show(shrinkFilter)
<paraview.servermanager.UnstructuredGridRepresentation object at 0x000000000BE85B70>

>>> Render()
<paraview.servermanager.RenderView object at 0x000000000C26D278>

>>> 
```
PARAVIEW/PYTHON SCRIPTING

- Should see something similar to this
# Create a cone and assign it as the active object
# Set a property of the active object
# Apply the shrink filter to the active object
# Shrink is now active
# Show shrink
# Render the active view
The value returned by Cone() and Shrink() was assigned to Python variables and used to build the pipeline.

ParaView keeps track of the last pipeline object created by the user. This allows you to accomplish everything that was just done.
PARAVIEW/PYTHON SCRIPTING

CREATING A PIPELINE

```python
>>> from paraview.simple import *
# Create a cone and assign it as the active object
>>> Cone()
<paraview.servermanager.Cone object at 0x2910f0>
# Set a property of the active object
>>> SetProperties(Resolution=32)
# Apply the shrink filter to the active object
# Shrink is now active
>>> Shrink()
<paraview.servermanager.Shrink object at 0xaf64050>
# Show shrink
>>> Show()
<paraview.servermanager.UnstructuredGridRepresentation object at 0xaf57f90>
# Render the active view
>>> Render()
<paraview.servermanager.RenderView object at 0xaf57ff0>
```
RUN FROM SCRIPT

Type the following code in a text editor

Cone()
SetProperties(Resolution=32)
Shrink()
Show()
Render()

Save file as testScript.py

Click RUN SCRIPT from Python Shell
Locate and select script
Click OK

Should see
New objects in Pipeline Browser
Cone rendering in 3D Viewer
ADDITIONAL RESOURCES

http://www.paraview.org

ParaView User’s Guide: Downloaded with ParaView

ParaView Sample Data
http://www.paraview.org/Wiki/The_ParaView_Tutorial

ParaView/Python Scripting – KitwarePublic
http://www.paraview.org/Wiki/ParaView/Python_Scripting

ParaView Server Manager
Vetria L. Byrd
Assistant Professor
Computer Graphics Technology
vlbyrd@purdue.edu

https://polytechnic.purdue.edu/profile/vbyrd
@VByrdPhD, @BPViz, @VisREU

Purdue Polytechnic Institute
polytechnic.purdue.edu

/ TechPurdue