Procedurals in Katana

RenderMan Procedurals give you the opportunity to dynamically create geometry and then pass it to RenderMan while rendering. By passing geometry in to RenderMan at the very last second, you can save the time and memory it would take to generate and display the geometry in Katana, Maya, or other application.

Procedurals are most commonly used in fur/hair rendering, crowd rendering, dynamically grown vegetation, or procedurally created environments. They can be used for anything.

While procedurals have many advantages, one of the disadvantages derives from its strength. Because the geometry is generated only within RenderMan, Katana doesn't know anything about the geometry and can't display it. If the application you are using to create the procedural has some sort of static representation, such as an Alembic file, we recommend that you use the Katana Viewer proxy feature so that you can see something other than a bounding box in the Viewer. See the Katana Help for how to use Viewer proxies.

Setting Up a Procedural in the Node Graph

There are three main steps that you must do in order to tell Katana to properly pass a procedural to RenderMan.

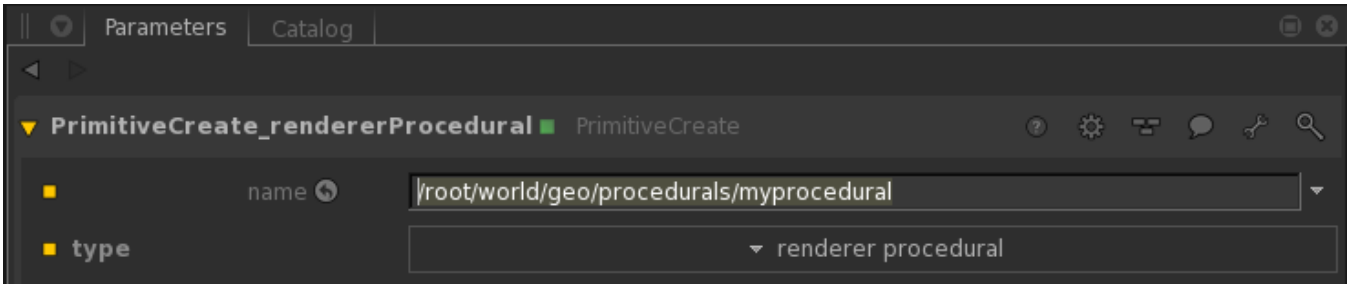
- Set up a Scene Graph location of type "renderer procedural".
- Set the path of the shared library that contains the procedural code that RenderMan will call.
- Set up any additional arguments that the procedural requires in order to do its work.

Below each of the steps is detailed.

Step 1: Set up a Scene Graph location of type "renderer procedural"

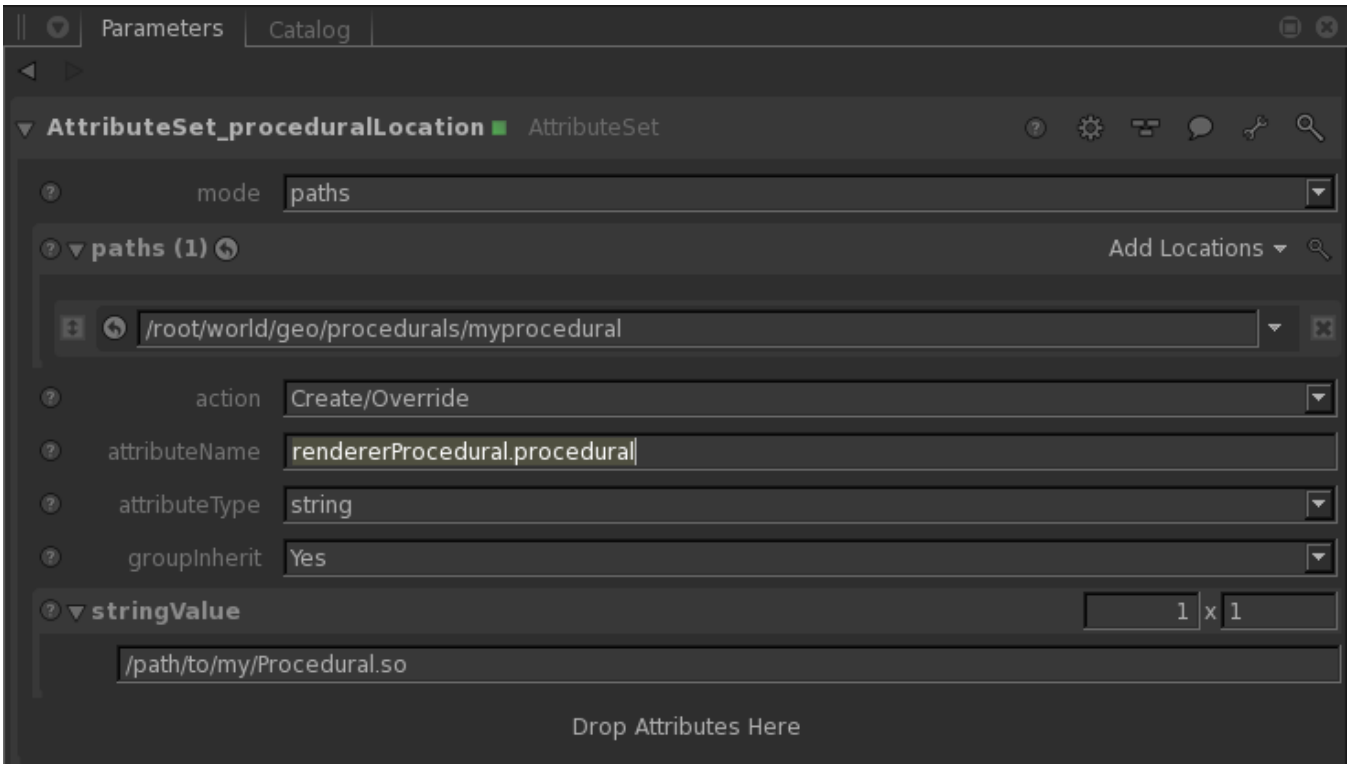
Within the Scene Graph, the "type" of each location controls how the Attributes for that location are passed to RenderMan. For RenderMan to call the Procedural at runtime, you must add a Scene Graph location of type "renderer procedural".

The easiest way for you to do this is to use the *PrimitiveCreate* node and set the type to "renderer procedural". Specify the name as well. In the example below, I have set it to /root/world/geo/procedurals/myprocedural



Step 2: Set the path of the shared library that contains the procedural code that RenderMan will call

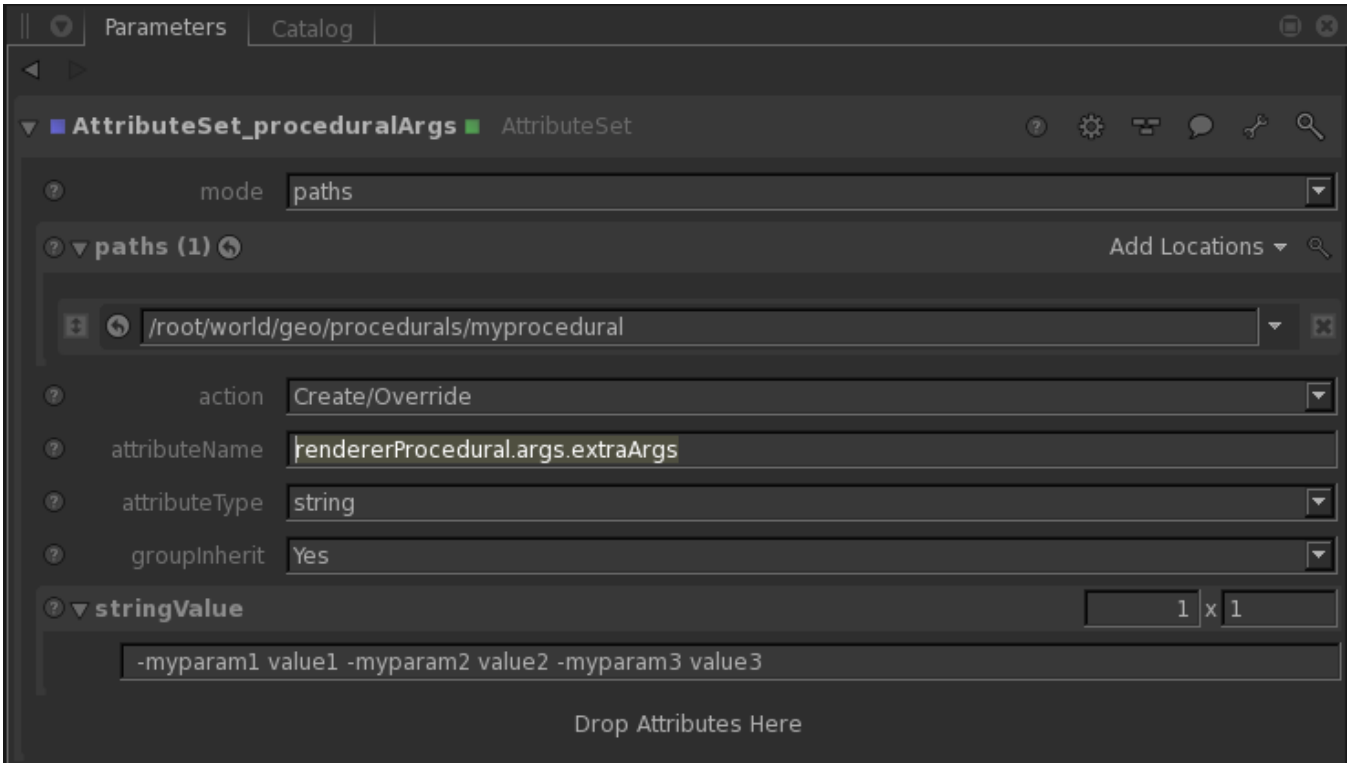
Developers of the application that you created your procedural geometry within will provide you with a RenderMan Procedural library. Consult with the documentation for that application or consult your pipeline team for where you can find this path. Below, I have specified a path to an imaginary library. Use an *AttributeSet* node to do this. The attribute name to use is: rendererProcedural.procedural



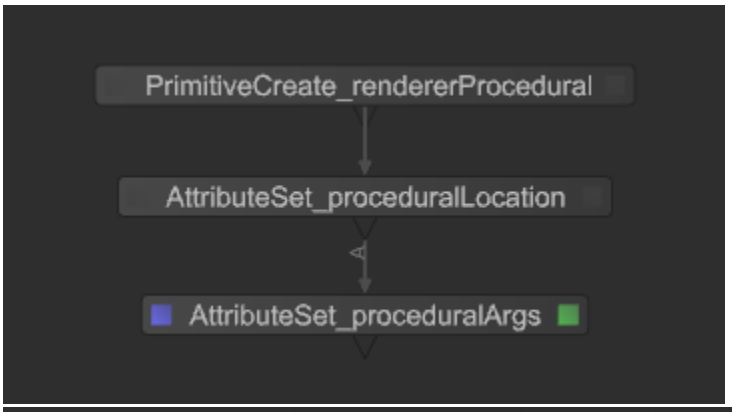
Step 3: **Set up any additional arguments that the procedural requires in order to do its work**

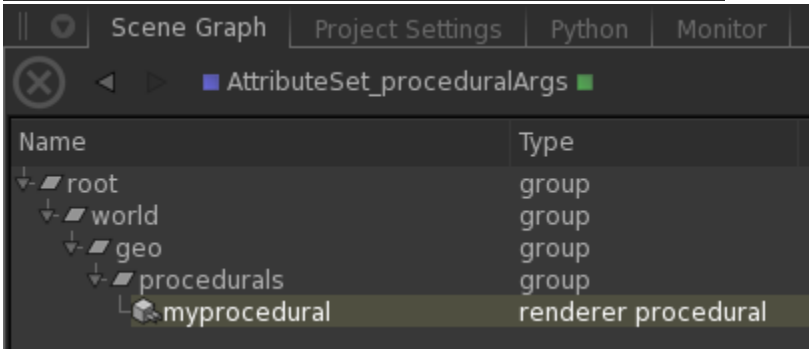
Each procedural will have a unique set of parameters that must be passed to it in order for the procedural to generate the correct geometry for the frame you are rendering. These parameters are likely to be similar to what you would specify to a command-line utility.

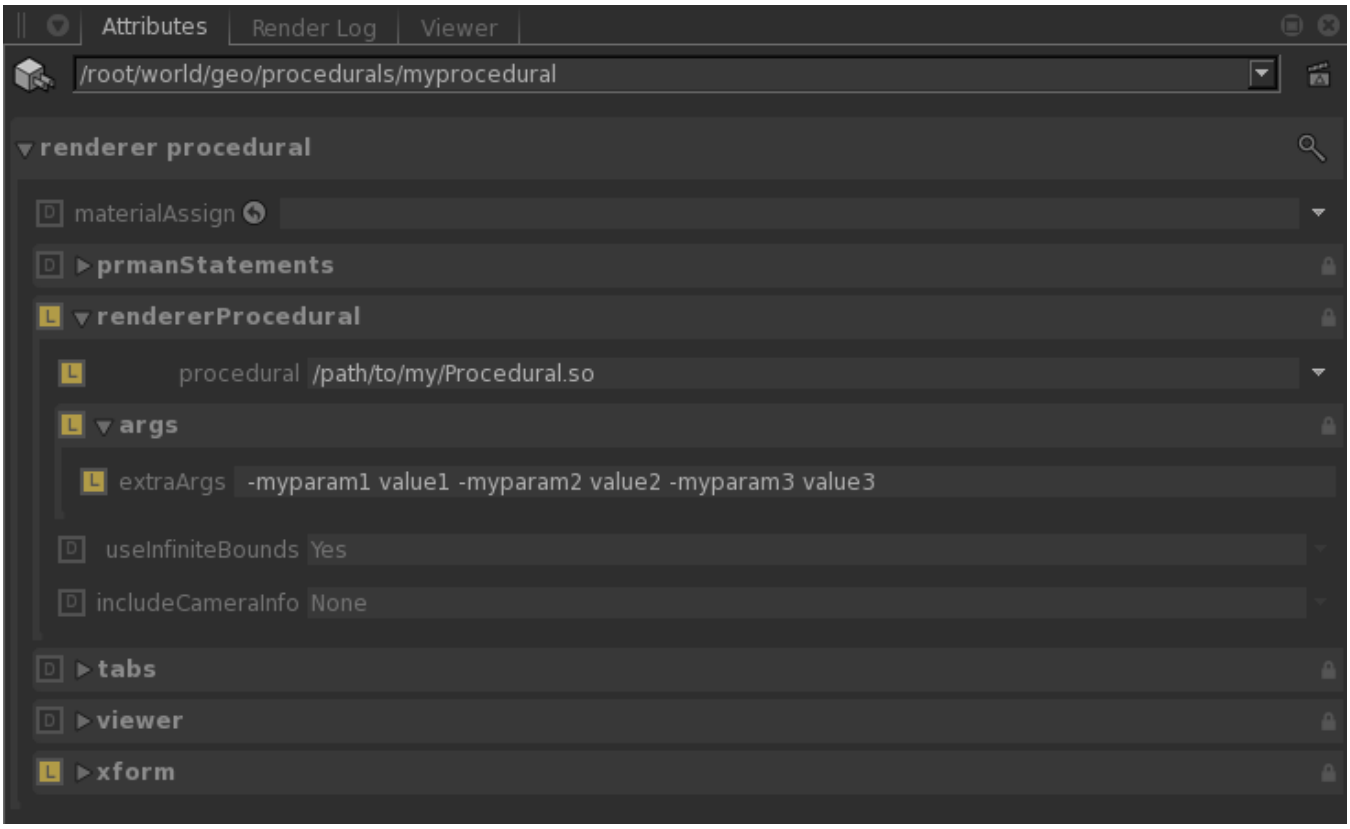
Below I have specified some imaginary attributes using an *AttributeSet* node that the imaginary Procedural.so needs in order to execute properly. The attribute name to use is: rendererProcedural.args.extraArgs



After you complete each of these steps above, you will end up with a Node Graph, Scene Graph, and Attributes similar to the example below.







Going Beyond the Basics

The information below only provides you with the most basic way to insert a RenderMan Procedural into your Katana scene. There are other more advanced options that you or your pipeline team can pursue. For each of the methods below, your mileage may vary. Take a look at the Katana documentation available from the Help menu within Katana for more details on each of the methods below.

- Create + use Args files for renderer procedurals to make it easier to specify the parameters that the procedural needs to do its work.
- Create a custom node that wraps the procedural – the ultimate in user-friendliness, but requires a developer.