Xpath Examples

Simple Examples

All paths:

```
//*
```

All paths that end in "room":

```
//room
```

All paths that end in "room" and start with "renderpass":

```
/renderpass//room
```

All paths that end in "room/chair" and start with "renderpass":

```
/renderpass//room/chair
```

An exact path:

```
/renderpass/street/house/room
```

Starts with renderpass, followed by one path element of anything, and then ends with "house/room":

```
/renderpass/*/house/room
```

Ends in "room/chair":

```
room/chair
```

Any node anywhere under a node called "room":

```
//room//*
```

Odd numbered chairs, but only those inside "room":

```
//*[substring(name(),6) mod 2 = 1 and parent::*[name() = 'room']] //*[substring(name(),6) mod 2 = 1 and parent::*[name() = 'room']]
```

More Complex Examples

The following examples, referring to attributes, are somewhat more complex.

Select objects beginning with the name *paperclip*:

```
//*[starts-with(name(),'paperclip')]
```

Select objects in the, and only in the "wood" set:

```
//*[@sets=',wood,']
```

Select objects beginning with the name pencil_box in the set "wood":

```
//*[starts-with(name(),'pencil_box')]//*[@sets=',wood,']
```

Select desk_1 and desk_2:

```
//desk_1 | //desk_2
```

Select of the objects the belong to the study_area_1 in the set "paintedmetal":

```
//study_area_1//*[contains(@sets, ',paintedmetal,')]
```

Select obects that don't belong to study_area_3:

```
//*[not(contains(name(),'study_area_3'))]
```

Select all object except desk_1, desk_2, blotter_1, and paperclip_2:

```
//*[not(contains(name(),'desk_1')) and not(contains(name(),'desk_2')) and not(contains(name(),'blotter_1')) and not(contains(name(), 'paperclip_1'))]
```

Select all object except *desk_1*`, desk_2, blotter_1, and paperclip_2 in set "shiny" but not set "burnt":

```
//*[not(contains(name(),'desk_1')) and not(contains(name(),'desk_2')) and not(contains(name(),'blotter_1')) and not(contains(name(), 'paperclip_1')) and contains(@sets,',shiny,') and not(contains(@sets,',burnt'))]
```

Select all of the pencil_boxes that are objects with "study" in the name:

```
//*[contains(name(),'study')//*[contains(name(),'pencil_box')]
```

Select the "bolts" in all study_areas except study_area_2:

```
//*[starts-with(name(),'study_area') and not(contains(name(),'3'))]//*[starts-with(name(),'bolt')]
```

Select all of the even numbered study_areas:

```
//*[substring-after(name(),'study_areas') mod 2 = 0]
```

Select all object that are in set "red" and set "chair":

```
//*[contains(@sets,',red,') and contains(@sets,'chair')]
```

Select all object that are in set "red" or set "chair":

```
//*[contains(@sets,',red,') or contains(@sets,'chair')]
```

Select all objects that are in set "red" but not "chair":

```
//*[contains(@sets,',red,') and not(contains(@sets,'chair'))]
```

Select all objects in sets that begin with "paint":

```
//*[starts-with(@sets,',paint')]
```

A Practical Scenario

XPath uses a matching syntax that allows the selection of children based on parents and attributes. It also alows specific selection based on path, index, and ancestry.

Th purpose of this section is to outline a few scenarios and how the XPath expression would be written to select the desired item in order to deliver the appropriate payload.

Say we have a scene with 10 robots, all based on the same model. They are textured the same, and multiple Assemblies have been pre-baked and are inserted into the Maya scene as lightweight Read Archives. All of the lookdev has been done, and explicit connections were made in the Look File associated with the model. Consider the following tasks:

- I need to hide the "front_panel" so that it isn't rendered on robot_3.

  /robot_3//front_panel
- The render is taking to long, so I need to hide all of the bolts on all of the robots, except for the closest one: robot_2.

  /assembly[name!="robot_2"]//shapes/name[contains(text(),'bolt')]text()
- Robot_4 is generally too shiny, so the director has asked for the shinyness (Ks) to be taken down by 1/2./assembly[name="robot_4"]//shape/name[containts(text(),'bolt')]/surface/roughness
- I need to hide all of the objects tagged in the set *expendable*.

  //@expendable

Multipass Binding Rules

Beyond simple path expressions, RLF rules can contain references to the renderpass node, which contains information about the current renderpass. These expressions will include /renderpass[@attrName="..."]/.

For example if you wish to apply a shader to all of the geometry in the scene for all passes, except shadow, where you would apply the default shader you would write the following rules:

Injectable Rule Set

Continue Matching	Rule	Payload
False	/renderpass[@class="shadow"]//*	defaultShader
False	//*	payloadID:a_shader

Rule one will find all geometry and stop matching when the pass_class is shadow, otherwise rule two will match all geometry.

You could also have written the rules in the following manner:

Injectable Rule Set

Continue Matching	Rule	Payload
False	/renderpass[@class !="shadow"]//*	a_shader
False	//*	defaultShader

Here we match all objects that aren't in the shadow pass_class and assign the payload with the ID a_shader, otherwise the default shader payload will be injected since the second rule will match all objects.

Ri Request Edit Rules

XPath attributes are also used to refer to parameters of Ri requests. To match an attribute in XPath, use the @ character before the name of the attribute:

Edit Rule Set

Continue Matching	Rule	Payload
True	/renderpass[@class !="SSRender"]//RiSurface/@*[contains(name(),"SSOutputFile")]	final_SS_payload

This rule will select the parameter of shaders that aren't in the SSRender pass class that have names that contain SSOutputFile and then will apply the payload final_SS_payload, which will replace the original value with an empty string. Since the shader uses the existence of the output file to inform its behavior, we have now achieved the same behavior that was accomplished with unique shader values for each pass.

Appendix

For more information on XPath Expressions:

- A quick reference with examples: http://en.wikipedia.org/wiki/XPath_1.0
- A technical language reference: http://www.w3.org/TR/xpath/