

# JEXL script tester

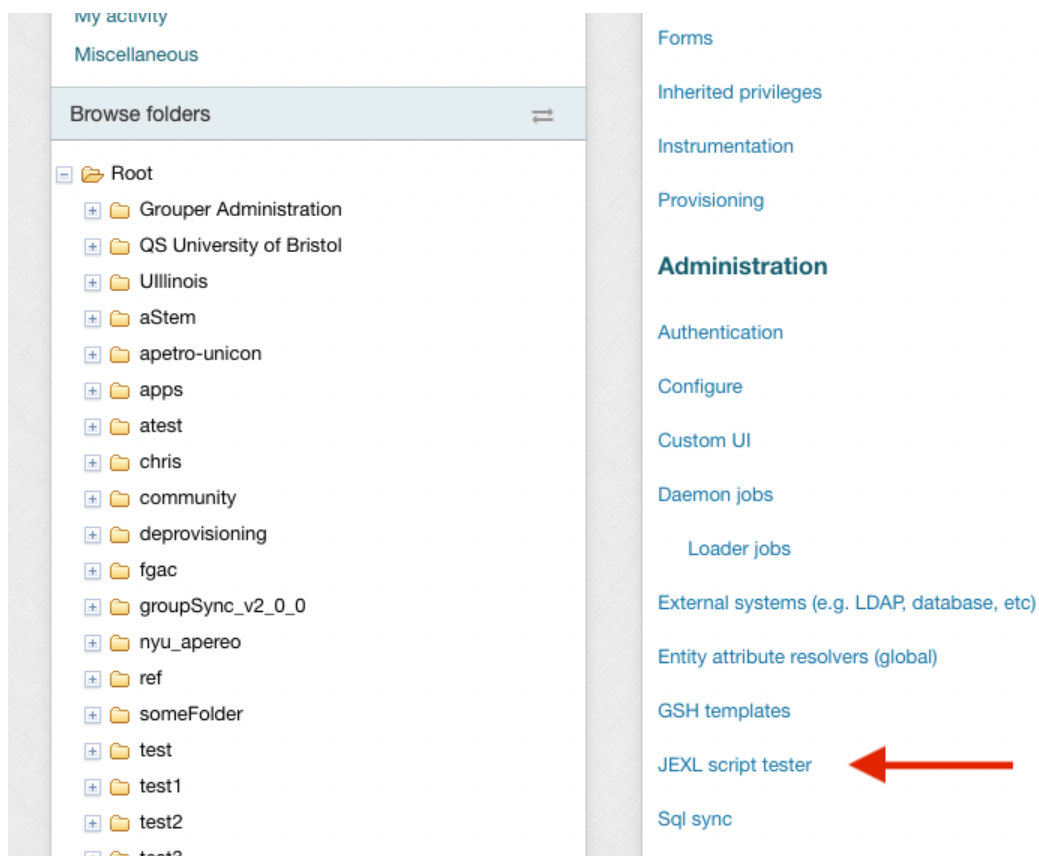
<a href="#">Wiki Home</a>	<a href="#">Grouper Release Announcements</a>	<a href="#">Grouper Guides</a>	<a href="#">Grouper Deployment Guide</a>	<a href="#">Community Contributions</a>	<a href="#">Internal Developer Resources</a>
---------------------------	---	--------------------------------	--	---	--

In Grouper v4.0.3+ there is a UI utility to test JEXL scripts.

The point is to test JEXL scripts, and also to show various example in the places where JEXL scripts are used. If you have a good example of a JEXL script please submit it to the Grouper team to add to the screen

## Use

Go to Misc -> JEXL script tester

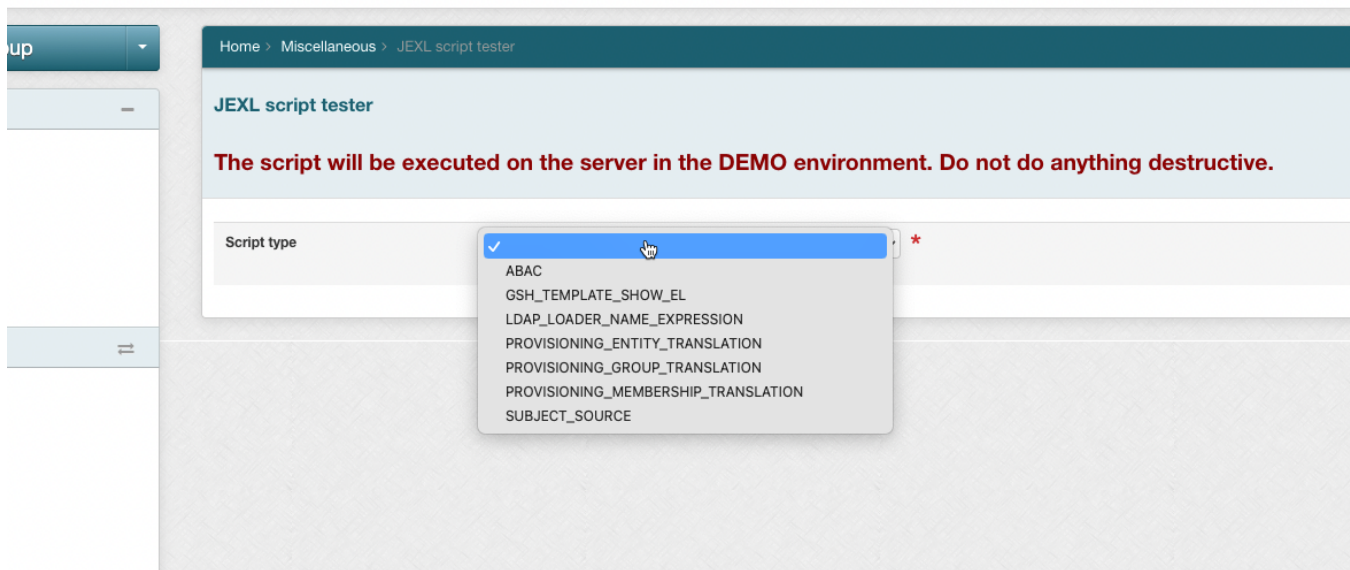


In order to use the JEXL script tester you need to be a Grouper admin and be in the group: etc:jexlScriptTestingGroup

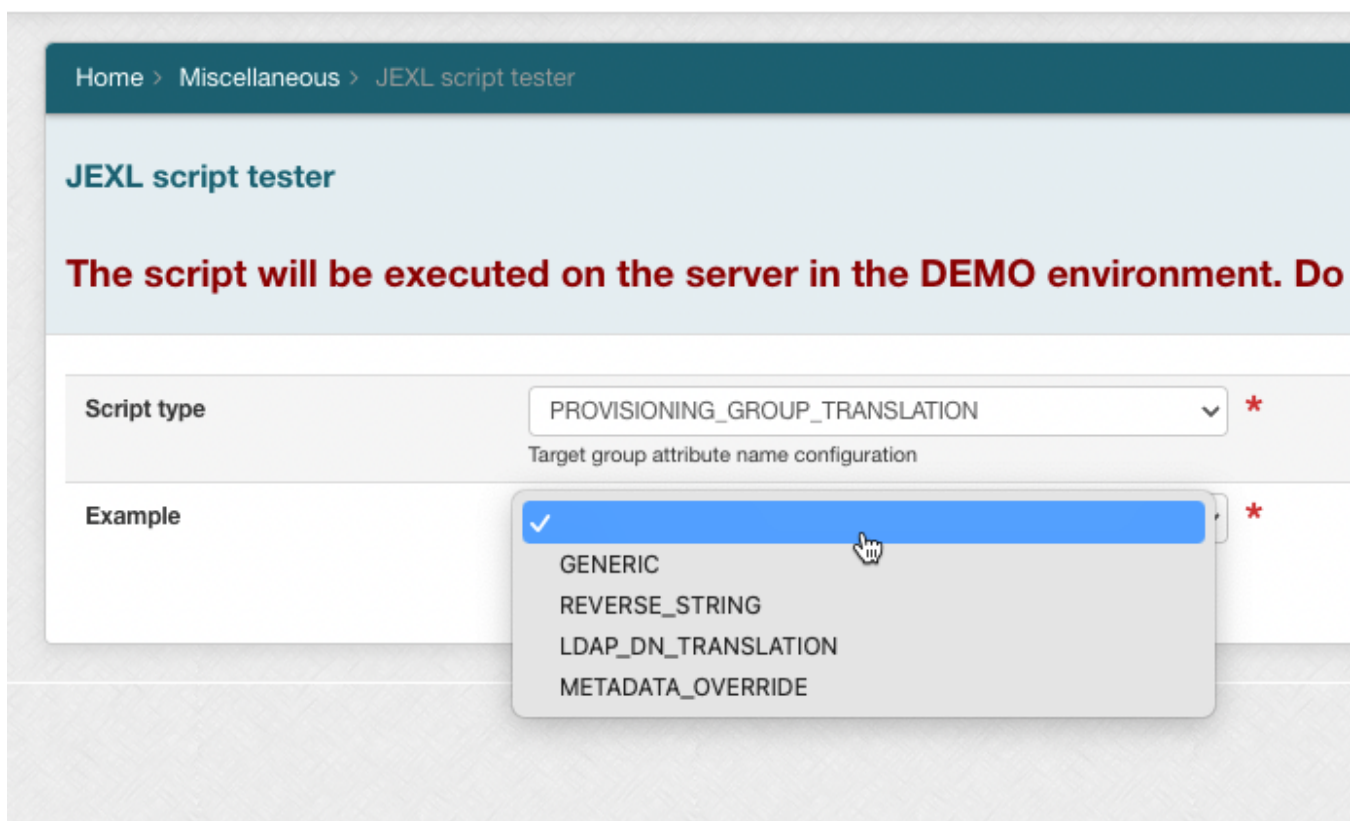
## Warning

These JEXL scripts via GSH will run in the UI in the environment where the UI is running. Do not code destructive GSH / JEXL or bad things will happen.

## Select the type of script to run



Select the example based on the type of script



## Customize 'Available beans'

The 'available beans' section of the screen will pre-populate beans that are in scope for the particular JEXL script. You can customize the data based on what you are testing (in this case, edit the group name)

<b>Example</b>	<div> <div>REVERSE_STRING</div> <div>Reverse the group name, change the delimiter to period, and max length 64</div> </div>
<b>Available beans GSH script</b>	<pre>import edu.internet2.middleware.grouper.app.provisioning.*; import edu.internet2.middleware.grouperClient.jdbc.tableSync.GcGrouperSyncGroup;  ProvisioningGroup grouperProvisioningGroup = new ProvisioningGroup(); grouperProvisioningGroup.setName("applications:departmentOfArtsAndSciences:commonResources:servicesOfSchool:policyGroups:groupsForActiveDirectory:historyDepartmentProfessors"); grouperProvisioningGroup.setId("abc123"); grouperProvisioningGroup.setIdxIndex(1234567L);  ProvisioningGroup grouperTargetGroup = new ProvisioningGroup();  GcGrouperSyncGroup gcGrouperSyncGroup = new GcGrouperSyncGroup();  ProvisioningGroupWrapper provisioningGroupWrapper = new ProvisioningGroupWrapper(); provisioningGroupWrapper.setGrouperProvisioningGroup(grouperProvisioningGroup); provisioningGroupWrapper.setGrouperTargetGroup(grouperTargetGroup); provisioningGroupWrapper.setGcGrouperSyncGroup(gcGrouperSyncGroup);  Map&lt;String, Object&gt; elVariableMap = new HashMap&lt;String, Object&gt;(); elVariableMap.put("grouperProvisioningGroup", grouperProvisioningGroup); elVariableMap.put("provisioningGroupWrapper", provisioningGroupWrapper); elVariableMap.put("grouperTargetGroup", grouperTargetGroup); elVariableMap.put("gcGrouperSyncGroup", gcGrouperSyncGroup);</pre>

## Edit the 'Null checking script'

The problem with JEXL is in strict mode, a misspelled variable name throws an exception, but a null value also throws an exception. So we are adding a sibling script next to each script in Grouper to check for nulls in non-strict mode. If the null check fails, then null will be used for the script and an error will not be thrown. We used to use the ternary operator for this but you should use a null checking script instead.

Here is an example in provisioning

<p>Group attr cn - translation expression</p>	<div> <input type="checkbox"/> EL?         </div> <div> <input type="text" value="\${(groupProvisioningGroup.name.toLowerCase())}"/> </div> <p>This is the translation for this attribute from Grouper to Target format during Group updates and inserts (if not overridden by "insert only" translation). Available variables:</p> <ul style="list-style-type: none"> <li>groupProvisioningGroup (ProvisioningGroup.java): this is the Grouper representation of the group. Fields include: id, idIndex, name, displayName. Attributes include: description. Here is an example of a regex replace all non alphanumeric ids:  <code>\${(groupProvisioningGroup.name.replaceAll('[^a-zA-Z0-9]', '_'))}</code> </li> <li>Retrieve attributes:  <code>\${(groupProvisioningGroup.retrieveAttributeValueString('someAttr'))}</code> </li> <li>Retrieve metadata:  <code>\${(groupProvisioningGroup.retrieveAttributeValueString('md_groupier_allowOnlyMembersToPost'))}</code> </li> <li>gcGrouperSyncGroup (GcGrouperSyncGroup.java): this is the "sync" group object. Fields include: extension, groupAttributeValueCache0, groupAttributeValueCache1, groupAttributeValueCache2, groupAttributeValueCache3, id, idIndex, idIndexString, name</li> <li>provisioningGroupWrapper (ProvisioningGroupWrapper.java): holds references to all group data. Fields include: create, delete, recal, gcGrouperSyncGroup, grouperProvisioningGroup, grouperTargetGroup, targetProvisioningGroup, targetNativeGroup</li> <li>grouperTargetGroup (ProvisioningGroup.java): generally this is not needed since it is being built by the translation process. Attributes depend on the provisioner.  <code>\${(grouperTargetGroup.retrieveAttributeValueString('someAttr'))}</code> </li> </ul>
<p>Group attr cn - check for nulls in script</p>	<div> <input type="checkbox"/> EL?         </div> <div> <input type="radio"/> Default value (False)           <input checked="" type="radio"/> True           <input type="radio"/> False         </div> <p>Any null variables throw an exception in the JEXL translation, so this is a place you can check for null variables and just have it return null if so. Default value is 'false'.</p>
<p>Group attr cn - translation continue condition</p>	<div> <input type="checkbox"/> EL?         </div> <div> <input type="text" value="\${(groupProvisioningGroup != null &amp;&amp; grouperProvisioningGroup.name != null)}"/> </div> <p>Condition that must be true in order to continue translation</p>

Here is an example of the null checking script in the jexl script tester

Available beans GSH script

Null checking JEXL script

```

${grouperProvisioningGroup.name != null}

```

You could put a JEXL script here to check for nulls safely because JEXL will throw exception if any variables are null e.g. false

Script

```

${edu.internet2.middleware.grouper.util.GrouperUtil.stringFormatNameReverseReplaceTruncate(grouperProvisioningGroup.name, ".", 64)}

```

Edit the example JEXL script (or submit as is)

Script

```

${edu.internet2.middleware.grouper.util.GrouperUtil.stringFormatNameReverseReplaceTruncate(grouperProvisioningGroup.name, ".", 64)}

```

Add your script here

See the result

Note the 'type' of data is printed before the string value. In this example the input is:

```
applications:departmentOfArtsAndSciences:commonResources:servicesOfSchool:policyGroups:groupsForActiveDirectory:
historyDepartmentProfessors
```

The method called is: stringFormatNameReverseReplaceTruncate with length of 64 and use dots as separators. You see the result is it reversed with dots and length less than 64.

Add your script here

Submit

Result

```

java.lang.String: historyDepartmentProfessors.groupsForActiveDirectory.policyGroup

```

Gsh script executed

Show/Hide gsh script

## GSH source

If you want to see the GSH source to run this yourself, it is printed below

Gsh script executed

Show/Hide gsh script

```
import edu.internet2.middleware.grouper.app.jexlTester.*;
import edu.internet2.middleware.grouper.app.provisioning.*;
import edu.internet2.middleware.grouperClient.jdbc.tableSync.GcGrouperSyncGroup;

ProvisioningGroup grouperProvisioningGroup = new ProvisioningGroup();
grouperProvisioningGroup.setName("applications:departmentOfArtsAndSciences:commonResources:servicesOfSchool:policyGroups:groupsForActiveDirectory:historyDepartmentProfessors");
grouperProvisioningGroup.setIdx("abc123");
grouperProvisioningGroup.setIdxIndex(1234567L);

ProvisioningGroup grouperTargetGroup = new ProvisioningGroup();

GcGrouperSyncGroup gcGrouperSyncGroup = new GcGrouperSyncGroup();

ProvisioningGroupWrapper provisioningGroupWrapper = new ProvisioningGroupWrapper();
provisioningGroupWrapper.setGrouperProvisioningGroup(grouperProvisioningGroup);
provisioningGroupWrapper.setGrouperTargetGroup(grouperTargetGroup);
provisioningGroupWrapper.setGcGrouperSyncGroup(gcGrouperSyncGroup);

Map<String, Object> elVariableMap = new HashMap<String, Object>();
elVariableMap.put("grouperProvisioningGroup", grouperProvisioningGroup);
elVariableMap.put("provisioningGroupWrapper", provisioningGroupWrapper);
elVariableMap.put("grouperTargetGroup", grouperTargetGroup);
elVariableMap.put("gcGrouperSyncGroup", gcGrouperSyncGroup);

Object result = null;
String nullCheckingJexlScript = "" + '$' + "{grouperProvisioningGroup.name != null}";
boolean shouldContinue = ScriptType.PROVISIONING_GROUP_TRANSLATION.nullCheckingShouldContinue(elVariableMap, nullCheckingJexlScript);
if (shouldContinue) {
    String jexlScript = "" + '$' + "{edu.internet2.middleware.grouper.util.GrouperUtil.stringFormatNameReverseReplaceTruncate(grouperProvisioningGroup.name, '\\.\\', 64)}";
    result = ScriptType.PROVISIONING_GROUP_TRANSLATION.runJexl(elVariableMap, jexlScript);
    JexlScriptTester.registerOutput(result);
} else {
    JexlScriptTester.registerOutput("Null checking script returned false so the main JEXL script did not execute.");
}
```

## Jexl advice

### Jexl java bean property names

It is better to use the getter name than the property name. JEXL will throw an exception if there is a null and referring to it as property name.

For example, this will return null

```
ProvisioningGroup provisioningGroup = new ProvisioningGroup();

Map<String, Object> variableMap = new HashMap<String, Object>();
variableMap.put("provisioningGroup", provisioningGroup);

String result = GrouperUtil.substituteExpressionLanguage("${provisioningGroup.getName()}", variableMap,
true, false, false);

System.out.println(result);
```

But this will throw an error

```
ProvisioningGroup provisioningGroup = new ProvisioningGroup();

Map<String, Object> variableMap = new HashMap<String, Object>();
variableMap.put("provisioningGroup", provisioningGroup);

String result = GrouperUtil.substituteExpressionLanguage("${provisioningGroup.name}", variableMap, true,
false, false);

System.out.println(result);
```

## See Also

[Grouper ABAC Script Analysis](#)