knowledge14

April 27–May 1, 2014 • Moscone Center • San Francisco, CA

Lab Guide

Jelly 301:

Create Advanced Interfaces with Jelly Script

Tyler Jones & Chuck Tomasi

Lab instance: https://**<<name>>**.lab.service-now.com

Login / Passwords:

admin / Knowledge14

itil / Knowledge14

employee / Knowledge14

Lab Goal

The goal is to get a deep understanding of Jelly. You should leave this lab feeling not only like you can read Jelly code, but that you can write it yourself.

The Jelly Tag

We're going to start talking today about the Jelly Tag. This is the base tag at the root of a Jelly script document:

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

</j:jelly>
```

The xmlns:j and the xmlns:j2 are Jelly tags, defined in CoreTagLibrary.java. They include:

- If
- While
- Set

The xmlns:g and xmlns:g2 tags are Glide tags, defined in GlideTagLibrary.java. They include:

- Evaluate
- Breakpoint
- Set_if

®2014 ServiceNow, Inc. All rights reserved.

Two Phase Rendering

One thing that commonly trips people up is two phase rendering. Before I explain why it's important, I'll explain how it works.

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:evaluate var="jvar_records" object="true">

          var records = new GlideRecord('incident');

          records.get(RP.getParameterValue('sysparm_sys_id'));

          records;

    </g:evaluate>

    <j2:if test="${jvar_records.isValidRecord()}">

          We found ${jvar_records.getValue('number')}

    </j2:if>

</j:jelly>
```

When this is executed, the phase 1 content is run first. What does this mean? Every tag starting with g: or j: is run, and all of the JEXL statements that have curly braces {}. Everything outputs as text, so no variables are left when it is fully rendered (this means that variables don't really carry over to phase 2).

©2014 ServiceNow, Inc. All rights reserved.

Before we run phase 2 Jelly, we change the name spaces of the j2 and g2 to jelly:core and glide, meaning that those tags will now run.

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="jelly:core"
xmlns:g2="glide">

    <j2:if test="true">

          We found INC100001

    </j2:if>

</j:jelly>
```

When we run phase two, we run the g2: and j2: tags, as well as the square brackets []. All that is left is "We Found INC100001."

You're probably thinking, "Why do we have two phases? What's the meaning behind that?" The answer is that the *results of the phase 1 render are cached on the server. This makes your system faster*. This also carries with it a cost. You will want to use phase 1 for things like system configuration. For data such as incident data, you will want to use phase 2 scripting. Otherwise, updating records may not cause your page to change content.

©2014 ServiceNow, Inc. All rights reserved.

Tags – Evaluate

This tag helps you to write server-side script as Rhino. You can use all the script APIs that you are familiar with in Business Rules. The variable output of the last line is assigned to the variable indicated.

Attributes:

`var` - The name of the variable that will be set to the result of the script. For best results, start your variable with `jvar_`

`object` - If set to `true`, the result of the expression will be treated as an object (GlideRecord) instead of a primitive variable (string or integer variable values).

`jelly` - If set to `true`, allows Jelly context variables to be referenced in the script.

`expression` - This is an expression to be executed for the value to put in `var`. The expression can be either of two places. First, it can be an attribute on the evaluate tag itself. Otherwise, the content between the beginning tag and ending tag is the expression. The last line of the expression is the actual value passed into `var`.

Sample:

```
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">
    <g:evaluate var="jvar_username">
          var user = gs.getUserName();
          if (user == 'guest')
                user = 'Not logged in';

          user;
    </g:evaluate>
    ${jvar_username}
</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Tags – Evaluate – Use as a library

If you don't specify the var attribute, the variables set inside of a `g:evaluate tag` will be left in the Rhino controller. By leveraging this, we can make small libraries in our Jelly code:

```
<g:evaluate>

    function hello_english(name) {

        return "Hello, " + name;

    }

    function hello_spanish(name) {

        return "Buenas tardes, " + name;

    }

</g:evaluate>

<p>

    English: ${hello_english('Francis')}

</p>

<p>

    Spanish: ${hello_spanish('Pancho')}

</p>
```

©2014 ServiceNow, Inc. All rights reserved.

Tags – If

This tag will perform an if statement. If the test parameter evaluates to false, then the contents of the tag are simply discarded from the output. (Note: there is no else tag)

Attributes:

`test`- The condition for applying the true or false value to the variable. Use a JEXL expression to output the string true or false to this attribute.

Sample:

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <j:if test="${gs.isLoggedIn()}">

          You are logged in as user ${gs.getUserName()}

    </j:if>

    <j:if test="${!gs.isLoggedIn()}">

          You are not logged in!

    </j:if>

</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Tags – Set

This tag sets a variable to a given value.

Attributes:

`var` - The name of the variable that will be set to the result of the script.

`value` – The (string) value to assign to the var

Sample:

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <j:set var="jvar_username" value="${gs.getUserName()}"/>

    ${jvar_username}

</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Tags – Set_If

Simply stated, think of this tag as a combination of the `set` tag and the `if` tag.

Attributes:

`var` - The name of the variable that will be set to the result of the script.

`test`- The condition for applying the true or false value to the variable. Use a JEXL expression to output the string true or false to this attribute.

`true`- The value assigned to `var` if the `test` is true.

`false`- The value assigned to `var` if the `test` is false.

Sample:

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:set_if test="${gs.isLoggedIn()}"

          var="jvar_login_message"

          true="You are logged in as user ${gs.getUserName()}"

          false="You are not logged in!"/>

    ${jvar_login_message}

</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Tags – While

This tag helps to loop over a set of data. If you are familiar with the GlideRecord.next() form of iterating over records, then this will come in handy when iterating over record data.

Attributes:

`test`- The condition for applying the true or false value to the variable. Use a JEXL expression to output the string true or false to this attribute.

Sample:

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:evaluate var="jvar_records" object="true">

          var gr = new GlideRecord('incident');

          gr.addQuery('assigned_to', gs.getUserID());

          gr.query();

          gr;

    </g:evaluate>

    <j:while test="${jvar_records.next()}">

          You own ticket ${jvar_records.getValue('number')}

    </j:while>

</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Tags – Breakpoint

This tag is useful when debugging your script. Sometimes your script doesn't output what you had thought. This tag will help you find the value that you meant to output.

Attributes:

`var` – (optional) – The parameter to show in the logs. If omitted, all variables are output into the log.

Sample:

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:breakpoint var="sysparm_parameter"/>

</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Tags – Choose/when/otherwise

These tags are the closest thing you will find to an if/else syntax. `Choose` starts the clause, `when` is like an `if`, and `otherwise` is like an `else` or base case.

Sample:

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:evaluate var="jvar_incidents" object="true">

          var gr = new GlideRecord('incident');

          gr.addQuery('assigned_to', gs.getUserID());

          gr.query();

          gr;

    </g:evaluate>

    <j:while test="${jvar_incidents.next()}">

        <j:choose>

            <j:when test="${jvar_incidents.getValue('priority') == 1}">

                <b>${jvar_incidents.getDisplayValue()} is a big deal!</b>

            </j:when>

            <j:otherwise>

                ${jvar_incidents.getDisplayValue()} is not important!

            </j:otherwise>

        </j:choose>

        <br/>

    </j:while>

</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Organizing your Code

Organizing your code is important. You can organize your code into smaller components, like fields and rows of tables, and reuse them as necessary. UI Macros make this possible.

Macro Invoke

If you have a UI Macro titled include_me_macro, you could include it on a page in any of the following ways:

1. Use the UI Macro name as a tag

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:include_me_macro/>

</j:jelly>
```

2. Use the macro_invoke tag

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:macro_invoke macro="include_me_macro"/>

</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Dynamic Variables in UI Macros

Sometimes you will want to add dynamic variables into your UI Macros and call them like functions. Let's say that we wrote a UI Macro like the following:

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <label>${jvar_label}

        <input type="${jvar_type}" value="${jvar_value}"/>

    </label>

</j:jelly>
```

There are a two ways that you could call this UI Macro:

1. Declare the variables beforehand

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <j:set var="jvar_label" value="Test Field"/>

    <j:set var="jvar_type" value="text"/>

    <j:set var="jvar_value" value="Test Value"/>

    <g:label_and_field />

</j:jelly>
```

2. Include the parameters when you call the UI Macro:

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:label_and_field label="Test Field" type="text" value="Test Value" />

</j:jelly>
```

©2014 ServiceNow, Inc. All rights reserved.

Macro Tag

You can use the Macro tag to set default values to variables. This allows you to build more robust code as you can plan for the circumstance where not all variables are defined.

Here is a UI Macro titled label_and_field_with_defaults:

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:macro type="text" value="Fake Value"/>

    <label>${jvar_label}

        <input type="${jvar_type}" value="${jvar_value}"/>

    </label>

</j:jelly>
```

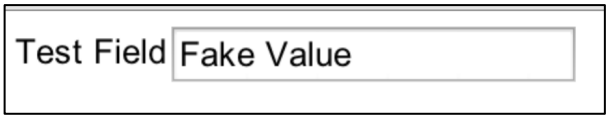
Then you can call this macro with the following UI Page:

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:label_and_field_with_defaults label="Test Field"/>

</j:jelly>
```

Output from UI page:



©2014 ServiceNow, Inc. All rights reserved.

Insert Tag

When a macro is called from another Jelly script, the insert tag will output the contents of the calling tag.

UI Page (caller)

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

     <g:blue_div>

          Message in a blue div

     </g:blue_div>

</j:jelly>
```

UI Macro: blue_div

```
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <div style="background-color:blue; color:white; padding: 20px;">

        <g:insert/>

    </div>

</j:jelly>
```

Output:



©2014 ServiceNow, Inc. All rights reserved.

Security

The following are to aid in making your scripts as secure as possible.

Escaping – HTML

Before	After
&	
<	<
>	>

Look at the following script:

```
<p>
    Hello ${sysparm_name}, and welcome to the site!
</p>
```

This code is insecure and may be susceptible to scripting attacks. This is because if the variable sysparm_name were coming from a URL parameter or user-submitted value, the variable may be output literally and unintended script may be executed. The following is improved:

```
<p>
    Hello ${HTML:sysparm_name}, and welcome to the site!
</p>
```

©2014 ServiceNow, Inc. All rights reserved.

Escaping – JS

Before	After
‘	\’
“	\”
CR	blank
NL	\n (the string “\n”)
<script…>	<script…>
</script>	</script…>
If phase 1	
&	&
<	\\u003C

Take a look at the following script:

```
<span>

    Hello, <span id="the_name_container"></span>!

</span>

<script type="text/javascript">

    $('the_name_container').innerHTML = '${sysparm_name}';

</script>
```

The script above may be susceptible to XSS scripting attacks. The following is improved:

```
<span>

    Hello, <span id="the_name_container"></span>!

</span>

<script type="text/javascript">

    $('the_name_container').innerHTML = '${JS:sysparm_name}';

</script>
```

©2014 ServiceNow, Inc. All rights reserved.

Escaping – No Escape

Look at this script designed to output the text from a KB article:

```
<g:evaluate var="jvar_kb_article">

    var gr = new GlideRecord('kb_knowledge');

    gr.get('number', 'KB0000012');

    gr;

</g:evaluate>

${gr.text}
```

The script above will not output properly because gr.text will output escaped. If gr.text was going to output Text, it would output that text instead of the intended output, **Text**. The below will force the system not to escape the output. Be very careful with this, and make sure that the source of the script is trusted:

```
<g:evaluate var="jvar_kb_article">

    var gr = new GlideRecord('kb_knowledge');

    gr.get('number', 'KB0000012');

    gr;

</g:evaluate>

<g:no_escape>${gr.text}</g:no_escape>
```

©2014 ServiceNow, Inc. All rights reserved.

Client Side JavaScript

Certain characters are appropriate for JavaScript but not for Jelly XML Script. Take, for example, this script. The <= and && characters are not XML compliant and the system will not let you save them.

Result:

```
<span id="script_result"></span>

<script type="text/javascript">

    var result = "";

    for (var i = 1; i <= 100; i++) {

        if (i % 3 == 0)

            result += 'Fizz';

        if (i % 5 == 0)

            result += 'Buzz';

        if (i % 3 != 0 && i % 5 != 0)

            result += i;

        result += '<br/>';

    }

    $('script_result').innerHTML = result;

</script>
```

©2014 ServiceNow, Inc. All rights reserved.

The below script is improved. Instead of &&, use $[AND]. Instead of <=, use !=. If you have to use less than (<), switch the statement and use greater than (>).

```
<span id="script_result"></span>

<script type="text/javascript">

    var result = "";

    for (var i = 1; i != 101; i++) {

        if (i % 3 == 0)

            result += 'Fizz';

        if (i % 5 == 0)

            result += 'Buzz';

        if (i % 3 != 0 $[AND] i % 5 != 0)

            result += i;

        result += '<br/>';

    }

    $('script_result').innerHTML = result;

</script>
```

®2014 ServiceNow, Inc. All rights reserved.

Exercises

DISCLAIMER: *This lab was intended to be done without looking at this section*. We think you'll learn more if you try to look at this section as infrequently as possible. But should you fall behind, this section has you covered. It's the answer key.

Exercise 1: Create a Jelly segment of code to output a message based on if they are an admin.

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <j:if test="${gs.hasRole('admin')}">

          Hello admin, you've logged in!

    </j:if>

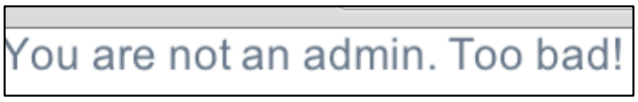
    <j:if test="${!gs.hasRole('admin')}">

          You are not an admin. Too bad!

    </j:if>

</j:jelly>
```

Sample output:



©2014 ServiceNow, Inc. All rights reserved.

Exercise 2: Create a segment that sets a variable with the link of the current user's record

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <j:set var="jvar_user_link" value="sys_user.do?sys_id=${gs.getUserID()}"/>

    <a href="${jvar_user_link}">Click here to view your user record</a>

</j:jelly>
```

Sample output:



©2014 ServiceNow, Inc. All rights reserved.

Exercise 3: Greet admins with the message “Welcome!” and others with a different message

```
<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">

    <g:set_if

          test="${gs.hasRole('admin')}"

          var="jvar_message"

          true="Welcome!"

          false="Welcome, non-admin user!"/>

    ${jvar_message}

</j:jelly>
```

Sample output:



©2014 ServiceNow, Inc. All rights reserved.

Exercise 4: Output a tree of incidents and their parent incidents

```
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
xmlns:g2="null">
    <!-- execute the incident query and pull the record set -->
    <g:evaluate var="jvar_incident" object="true">
          var gr = new GlideRecord('incident');
          gr.addNotNullQuery('parent');
          gr.orderBy('number');
          gr.query();
          gr;
    </g:evaluate>
    <table>
          <thead>
                <!-- column headers -->
                <tr>
                      <th>Number</th>
                      <th>Short Description</th>
                      <th>Parent Number</th>
                      <th>Parent Short Description</th>
                </tr>
          </thead>
          <tbody>
                <!-- display each incident with its parent -->
                <j:while test="${jvar_incident.next()}">
                      <tr>
                            <td>${jvar_incident.getValue('number')}</td>
                            <td>${jvar_incident.getValue('short_description')}</td>
                            <td>${jvar_incident.getValue('parent.number')}</td>

    <td>${jvar_incident.getValue('parent.short_description')}</td>
                      </tr>
                </j:while>
          </tbody>
    </table>
</j:jelly>
```

(Sample output on next page)

©2014 ServiceNow, Inc. All rights reserved.

Sample output:

Number	Short Description	Parent Number	Parent Short Description
INC0000002	Can't get to network file shares	INC0000002	Can't get to network file shares
INC0000014	missing my home directory	INC0000005	The sales department is uncool
INC0000015	I can't launch my game anymore	INC0000005	The sales department is uncool
INC0000017	How do I create a sub-folder	INC0000002	Can't get to network file shares
INC0000019	Can't launch X-Win32	INC0000002	Can't get to network file shares
INC0000025	I need more memory	INC0000005	The sales department is uncool
INC0000029	I can't get my weather report	INC0000002	Can't get to network file shares
INC0000031	EMAIL Server Down	INC0000002	Can't get to network file shares
INC0000040	JavaScript error	INC0000005	The sales department is uncool
INC0000041	My cubical phone does not work	INC0000005	The sales department is uncool
INC0000044	Can't log into SAP again	INC0000002	Can't get to network file shares
INC0000046	Can't access SFA software TEST again	INC0000002	Can't get to network file shares
INC0000047	Issue with email	INC0000005	The sales department is uncool
INC0000048	Having problems with performance on the Sales Tools	INC0000002	Can't get to network file shares
INC0000050	Exchange server appears to be down...lots of users impacted	INC0000002	Can't get to network file shares

Additional Resources

Jelly Tags

https://wiki.servicenow.com/index.php?title=Jelly_Tags

Extensions to Jelly Syntax

http://wiki.servicenow.com/index.php?title=Extensions_to_Jelly_Syntax

How to Escape in Jelly

https://wiki.servicenow.com/index.php?title=How_to_Escape_in_Jelly

©2014 ServiceNow, Inc. All rights reserved.