knowledge15®
April 19–24, 2015 • Mandalay Bay • Las Vegas, NV

Lab Guide

Jelly 201:

Getting Started with Jelly Script for UI Development

Tyler Jones & Will Leingang

Login / Passwords:

admin / Knowledge15

itil / Knowledge15

employee / Knowledge15

This

Page

Intentionally

Left

Blank

Lab Requirements

1. The user can only vote on each question once.
2. If the user has already voted, display the answer with a link to the report to show display results.
3. Results are saved in the **u_poll_response** table.
4. The system can manage multiple polls simultaneously but only display the “active” poll.
5. If more than one poll is active at a time, the most recently modified will be used.

Lab Data Source

1. Data is already provided in the following tables.
   - **u_poll** – Questions.
   - **u_poll_answer** – Possible answers to the questions.
   - **u_poll_response** – Answers users provided to specific answers.
   - **sys_user** – User information.

Wire frame of desired solution

Lab Goal

This lab explains how to create a UI new UI page and display a simple message.

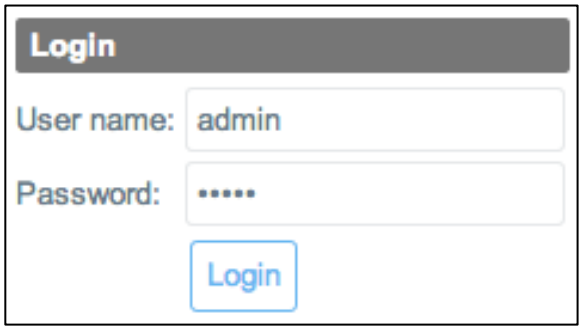
Concept: Become familiar with Jelly format/syntax.

Note: Some of these labs have errors in them to demonstrate common problems people encounter while programming in Jelly.

Lab 1
Create a New UI Page

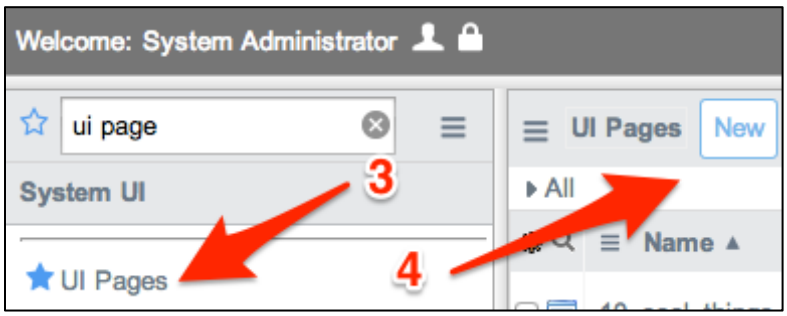
Log in to your instance

1. Your instance URL is on the cover page.
2. Log in using the **admin** credentials provided on the cover.



Create a New UI Page

3. Navigate to **System UI > UI Pages**.
4. Click **New**.



5. Enter the following information in the new record:

   Name: **poll_view**
   Description: **A poll manager UI Page**

The HTML field automatically contains 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">

</j:jelly>
```

6. Between the **<j:jelly> and </j:jelly>** tags, enter the following line:

```
Poll View 1.0<br />
```

Your HTML should now look like the example.

```
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
      xmlns:g2="null">
Poll View 1.0<br />
</j:jelly>
```

7. Click **Try It**.
   (Don't worry – this saves the record first)

8. What can you note from what you have seen?

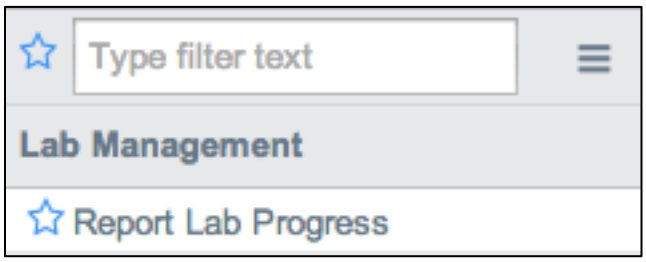
Notes

Verification

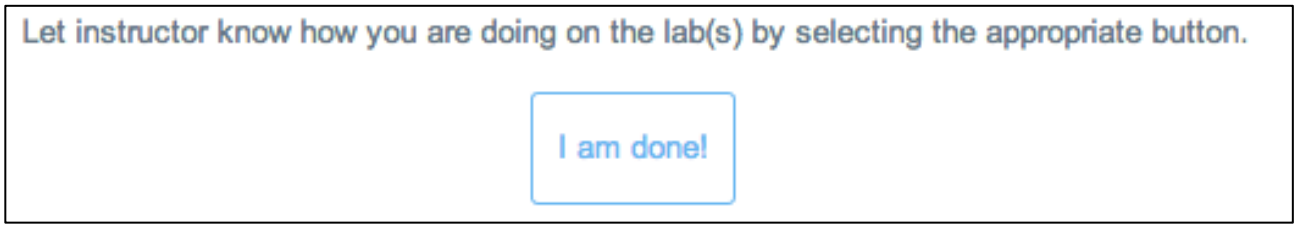
1. Confirm your screen looks like the example.

Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.



2. Click **I am done!**



Lab Goal

This lab explains how to run JavaScript in the UI page and use the result as a Jelly variable. Start simple with a small bit of JavaScript code and test displaying the result on the screen.

Concepts: Use a simple g:evaluate, pass a JavaScript variable to a Jelly variable, and use the Jelly variable in a JEXL expression.

Lab 2
User ID
Test

1. Copy and paste the contents from the box below to your poll_view UI Page immediately after the line
   `Poll View 1.0<br />`

2. Your XML should now look like the example.

```
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null"
      xmlns:g2="null">
Poll View 1.0<br />
  <!--
    Get the current user's ID
    Concepts:
      Simple g:evaluate
      Passing a Javascript value to Jelly variable
      Using the Jelly variable in a JEXL expression
  -->
<g:evaluate var="jvar_userid" expression="gs.getUserID;" />
Welcome user id: $[jvar_userid]<br/>

    <!-- REPLACE THIS LINE WITH LAB 3 -->
</j:jelly>
```

3. Click **Try It**.

4. Did the UI Page display output shown in the example?

```
Poll View 1.0

Welcome user id: 6816f79cc0a8016401c5a33be04be441
```

5. If not, go back and determine the source of the problem.

6. What can you note from what you have seen?

Notes

Verification

1. Confirm the screen appears like the example.

```
Poll View 1.0

Welcome user id: 6816f79cc0a8016401c5a33be04be441
```

Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.

2. Click **I am done!**

Lab Goal

This lab explains how to run JavaScript in the UI page and use the result as a Jelly variable. This time use a bit more complex JavaScript code to retrieve a record, display the results as a test/debug message, and store it for later use.

Concepts: Passing values from JavaScript to Jelly.

Lab 3
Get
Poll ID

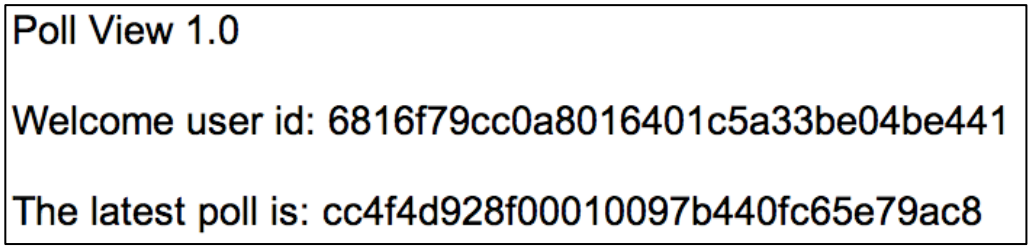
1. Copy and paste the contents from the box below to your poll_view UI page, replacing the line

   <!-- REPLACE THIS LINE WITH LAB 3 -->

```
<!--
  Now get the latest active question
  Concepts:
    Passing values from Javascript to Jelly variables
-->
<g:evaluate var="jvar_poll_id">

  var p1 = new GlideRecord('u_poll');

  p1.addQuery('active', true);
  p1.orderByDesc('sys_updated_on');
  p1.query();

  if (p1.next()) {
    result = p1.getValue('sys_id');
  }

  result;
</g:evaluate>
The latest poll is: ${jvar_pollid}<br/>

 <!-- REPLACE THIS LINE WITH LAB 4 -->
```

2. Click **Try It**.

3. Did the UI Page display output shown in the example?



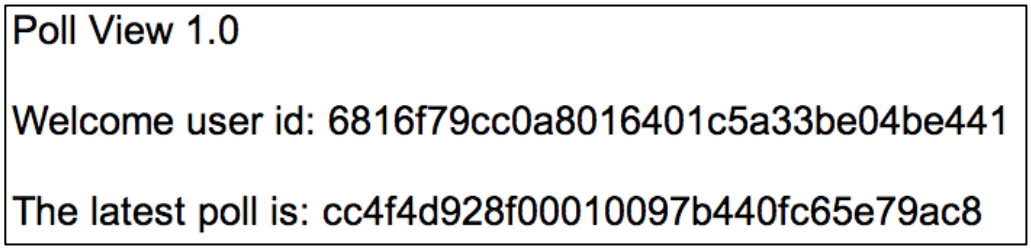
4. If not, go back and determine the source of the problem.

5. What can you note from what you have seen?

Notes

Verification

1. Confirm your screen appears like the example.



Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.

2. Click **I am done**!

Lab Goal

This lab explains how to read a value (the question text) from a database record and store it in a JavaScript variable to print as the poll question.

Concepts: Passing Jelly variables to JavaScript, setting JavaScript variables to be used later, using JavaScript variables in JEXL expressions.

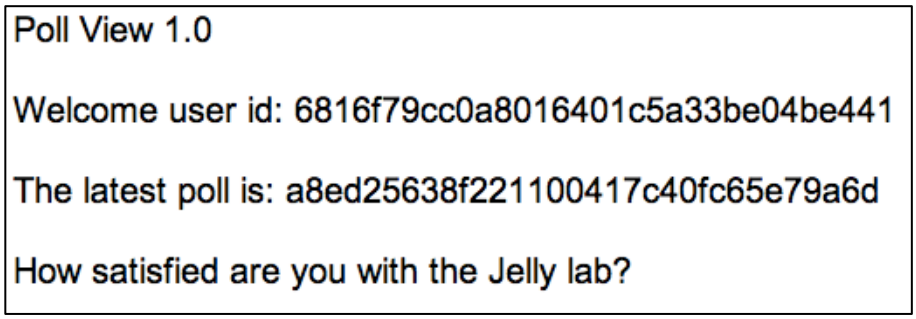
Lab 4 Get Question Text

1. Copy and paste the contents from the box below (or type the following code) to your **poll_view** UI page, replacing the line:

   <!-- REPLACE THIS LINE WITH LAB 4 -->

```
<!--
  Get the question text
  Concepts:
    Passing jelly variables to Javascript
    Setting Javascript variables to be used later
    Using Javascript variables in JEXL expressions
-->
<g:evaluate>

  var p2 = new GlideRecord('u_poll');
  var questionText= '';

  if (p2.get(${jvar_poll_id}) {
    questionText = p2.getValue('u_question');


  } else {
    gs.print('>>>poll_view: Error reading question');
  }

</g:evaluate>
<div class="poll_question">${questionText}</div>

  <!-- REPLACE THIS LINE WITH LAB 5 -->
```

2. Click **Try It**.
3. Did the UI Page display output shown in the following figure?

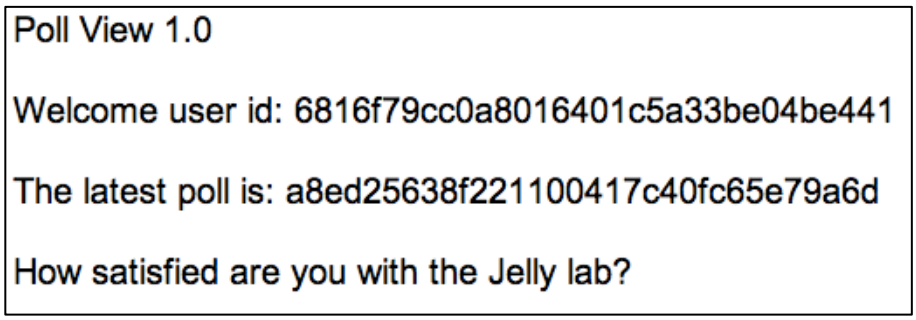


4. If not, go back and determine the source of the problem.
5. What can you note from what you have seen?

Notes

Verification

1. Confirm your screen appears like the example.



Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.
2. Click **I am done!**

Lab Goal

This lab explains Boolean variables in Jelly. In the lab, you attempt to read a record from the database to determine if this user has voted or not and save that status in a Jelly variable.

Concepts: Setting a Boolean Jelly variable, passing Jelly variables to JavaScript.

Lab 5
Get Voter Status

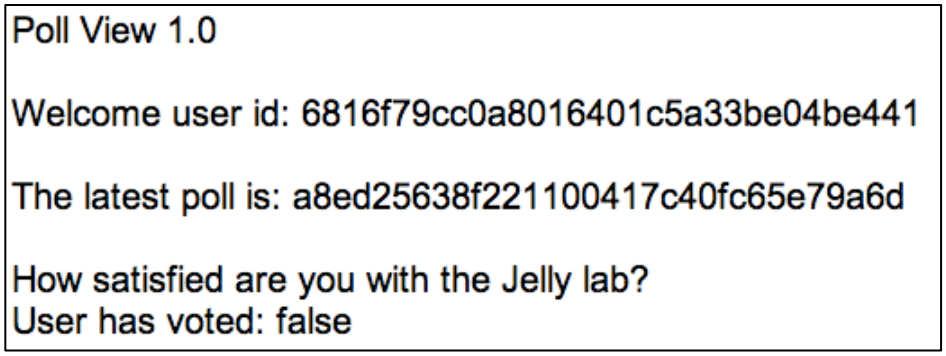
1. Copy and paste the contents from the box below (or type the following code) to your poll_view UI page replacing the line:

   <!-- REPLACE THIS LINE WITH LAB 5 -->

```
<!--
  Determine if the user has voted or not.
  jvar_has_voted will be set to true or false depending if
  an response is found
  Concepts:
    Setting a boolean Jelly variable
    Passing Jelly variables to Javascript
-->
<g:evaluate>
  var response = new GlideRecord('u_poll_response');
  var result;

  response.addQuery('u_user', ${jvar_userid});
  response.addQuery('u_poll', ${jvar_poll_id});
  response.query();
  result = response.next();

  result;
</g:evaluate>

User has voted: ${jvar_has_voted}<br/>

  <!-- REPLACE THIS LINE WITH LAB 6 -->
```

2. Click **Try It**.

3. Did the UI Page display output shown in the example?



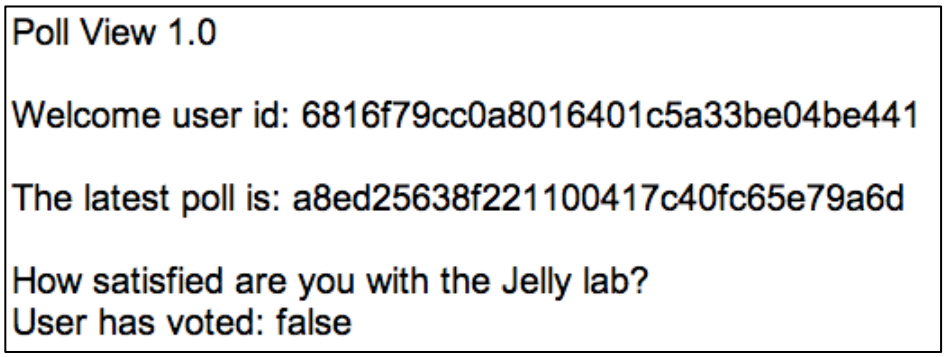
4. If not, go back and determine the source of the problem.

5. What can you note from what you have seen?

Notes

Verification

1. Confirm your screen looks like the example.



Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.
2. Click **I am done!**

Lab Goal

This lab explains Jelly conditional (if) statements. Use the previously saved Jelly variable to determine if this user has voted on this question already.

Concepts: Conditional statements on Boolean Jelly variable.

Lab 6
User Has Voted

1. Copy and paste the contents from the box below (or type the following code) to your poll_view UI page, replacing the line :

   <!-- REPLACE THIS LINE WITH LAB 6 -->

```
<!--
  If the user has already voted...
  Concepts:
    Conditional statement on boolean Jelly variable
-->
<j:if test="${jvar_has_voted eq true}">
Your response was: ${response.u_answer.u_text}<br />
</j:if>

  <!-- REPLACE THIS LINE WITH LAB 7 -->
```

2. Click **Try It**.
3. Did the UI Page display “Your response was: …”?
4. If not, go back and determine the source of the problem.
5. What can you note from what you have seen?

Notes

Verification

1. Confirm the UI page displays "**Your response was: …**"

Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.
2. Click I am Done!

Lab Goal

This lab explains inverse conditional statements. Use the previously saved Jelly variable to retrieve the answers if the user has not voted.

Concepts: Inverse conditional statements, passing Jelly variables to JavaScript, using gs.getMessage().

Lab 7
User Has NOT Voted

1. Copy and paste the contents from the box below (or type the following code) to your poll_view UI page, replacing the line:

   <!-- REPLACE THIS LINE WITH LAB 7 -->

```
<!--
  If the user has NOT voted...
  Concepts:
    Conditional statement on a boolean variable
-->
<j:if test="!${jvar_has_voted}">

<!--
  Retrieve the answers for the current question
  Concepts:
    Passing Jelly variables to Javascript
    Setting a global Javascript variable for later use
    Using gs.getMessage()
-->
  <g:evaluate>

    var ans = new GlideRecord('u_poll_answer');

    ans.addQuery('u_poll', jelly.jvar_poll_id);
    ans.orderBy('order');
    ans.query();

  </g:evaluate>
  We should have an answer object now: ${ans.hasNext()}<br />

  <!-- REPLACE THIS LINE WITH LAB 8 -->

</j:if>
```

2. Click **Try It**.

3. Did the UI Page display output shown in the following figure?

```
Poll View 1.0

Welcome user id: 6816f79cc0a8016401c5a33be04be441

The latest poll is: a8ed25638f221100417c40fc65e79a6d

How satisfied are you with the Jelly lab?
User has voted: false

We should have an answer object now: true
```

4. If not, go back and determine the source of the problem.

5. What can you note from what you have seen?

Notes

Verification

Confirm your screen appears like the example.

```
Poll View 1.0

Welcome user id: 6816f79cc0a8016401c5a33be04be441

The latest poll is: a8ed25638f221100417c40fc65e79a6d

How satisfied are you with the Jelly lab?
User has voted: false

We should have an answer object now: true
```

Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.

2. Click **I am done!**

Lab Goal

This lab explains how to understand and remediate data conditions that can cause a UI page to have issues. Using the question saved earlier, determine if it has answer records associated.

Concepts: Using a JavaScript method in a JEXL expression, **gs.getMessage()**, conditional statements.

Lab 8
Data
Validation

1. Copy and paste the contents from the box below to your poll_view UI Page, replacing the line:

   <!-- REPLACE THIS LINE WITH LAB 8 -->

```
<!--
  Check if this is a question without answers
  Concepts:
    Conditional statement on a Javascript variable/method
    gs.getMessage()

  <j:if test="${!ans.hasNext()}">
    <div class="poll_error">
      ${gs.getMessage('poll_no_answers_found');}
    </div><br/>
  </j:if>

  <!-- REPLACE THIS LINE WITH LAB 9 -->
```

2. Click **Try It**.
3. Did the UI Page display anything?
4. How can we test this?
5. What can you note from what you have seen?

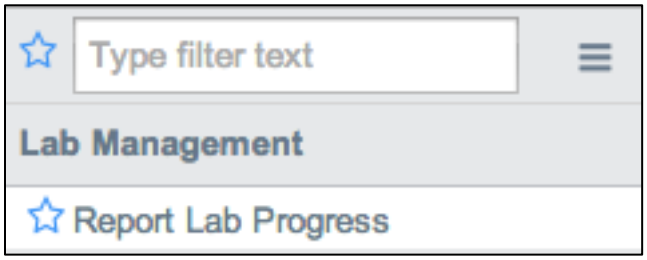
Notes

Verification

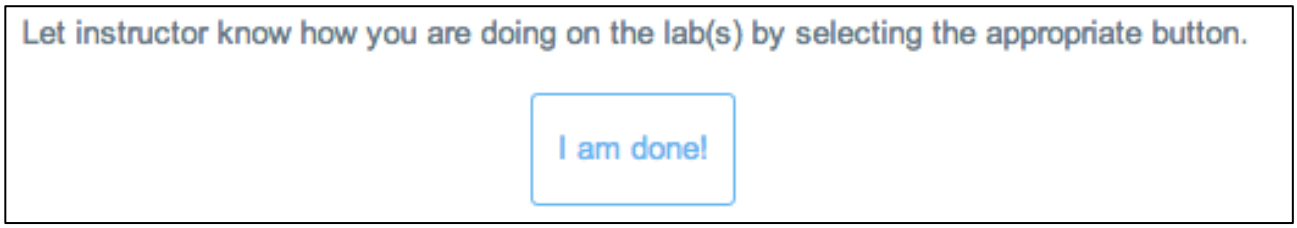
1. Confirm that the page displays the correct message when a question has no answers.

Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.



2. Click **I am done!**



Lab Goal

This lab explains how to integrate HTML in your Jelly script to produce user friendly formatting.

Concepts: Creating an HTML form, using Jelly variables in JEXL expressions in an HTML form, conditional statements, while-loop on a JavaScript variable.

Lab 9
Create The Form

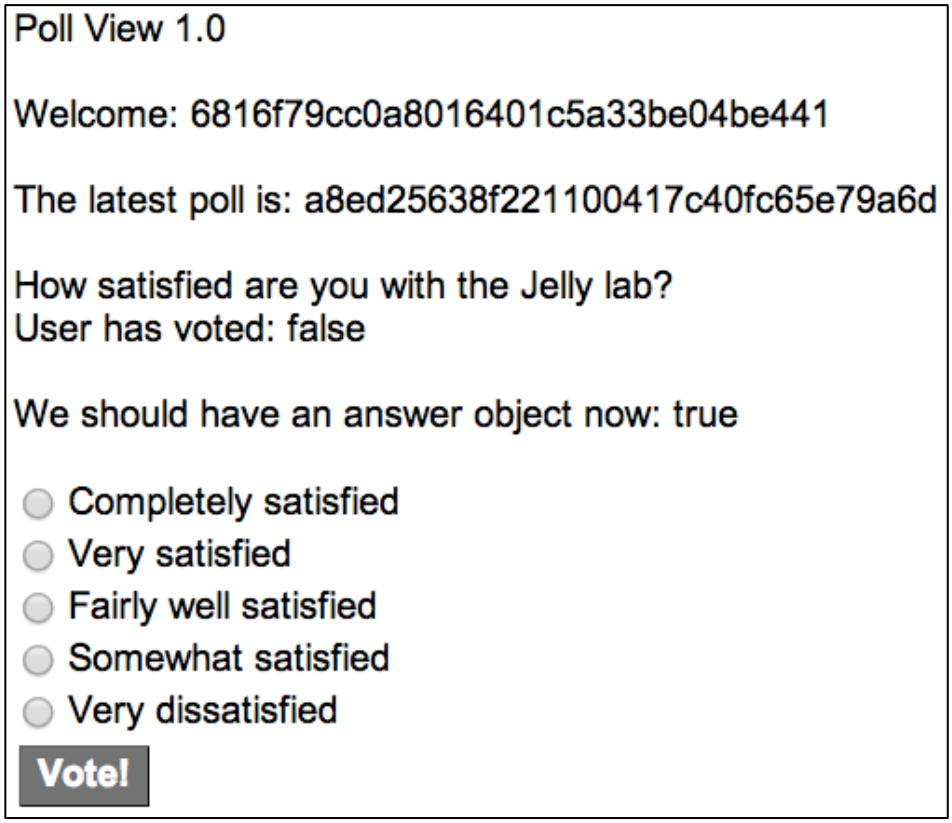
1. Copy and paste the contents from the box below to your poll_view UI Page replacing the line:

   <!-- REPLACE THIS LINE WITH LAB 9 -->

```
<!--
  Create the form
  Concepts:
    Creating a form and passing values to be processed
    Using Jelly variables in JEXL expressions in an HTML form
    Conditional statement on a Javascript variable/method
    While loop on a Javascript variable
-->
<j:if test="${ans.hasNext()}">
  <g:ui_form>
    <input type="hidden" name="poll_id" value="${jvar_poll_id}" />
    <input type="hidden" name="user_id" value="${jvar_userid}" />

    <j:while test="${ans.next()}">
      <div class="poll_answer">
      <g:ui_radio name="poll_answer" value="${ans.sys_id}" text="${ans.u_text}" />
      </div>
    </j:while>
    <g:ui_button action="save">${gs.getMessage('poll_vote_button')}</g:ui_button>

  </g:ui_form>
</j:if>
```

2. Click **Try It**.

3. Did the UI Page display output shown in the example?



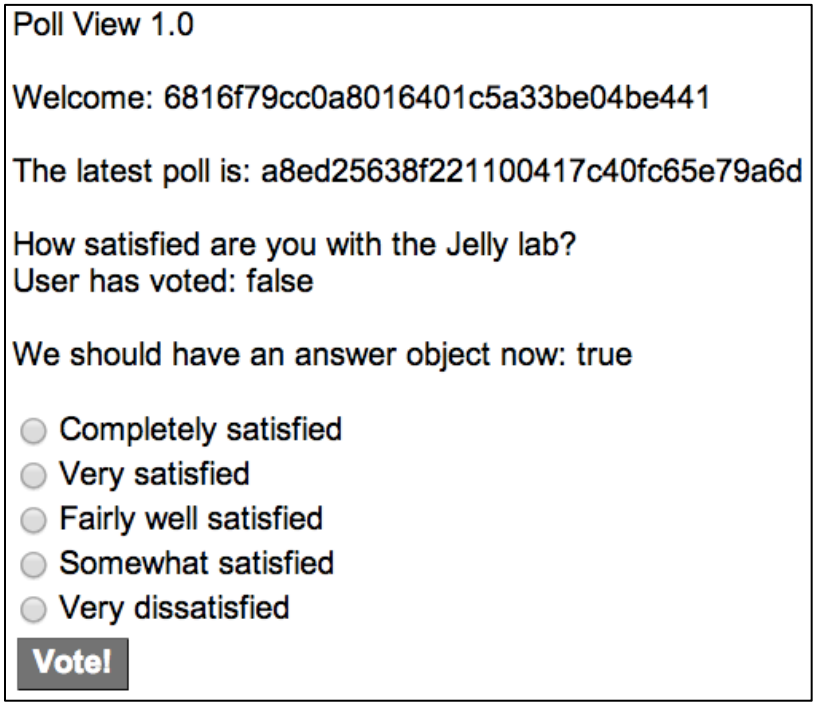
4. If not, go back and determine the source of the problem.

5. What can you note from what you have seen?

Notes

Verification

1. Confirm your screen appears like the example.



Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.
2. Click **I am done!**

Lab Goal

This lab provides a basic understanding of processing form results from a UI page using the Processing Script field on a UI page.

Concepts: Passing form field/value pairs to JavaScript and saving the results in the database.

Lab 10 Process Results

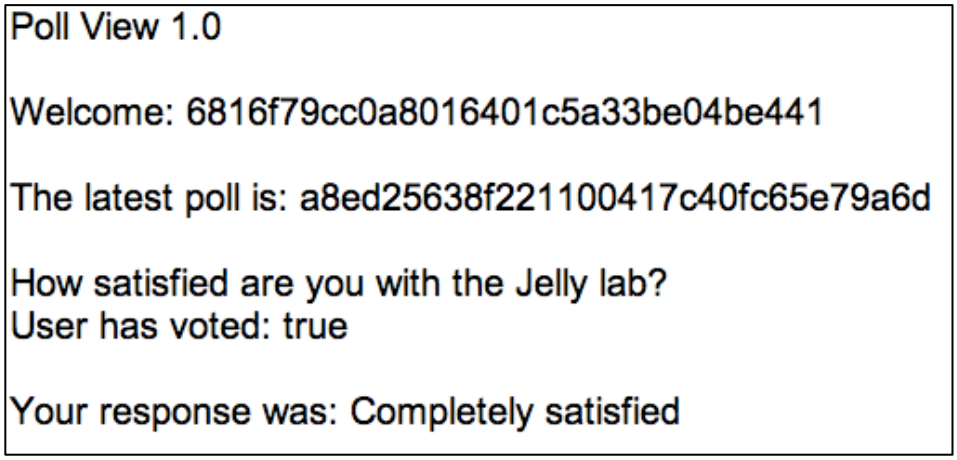
1. Copy and paste the contents from the box below (or type the following code) in the **Processing script** field of your UI page.

```
var pr = new GlideRecord('u_poll_response');

if (!JSUtil.nil(poll_id) && !JSUtil.nil(user_id)) {
   pr.initialize();
   pr.u_poll = poll_id;
   pr.u_user = user_id;
   pr.u_answer = poll_answer;
   pr.update();

}
```

2. Click **Try It**.

3. Choose one of the answers and click **Vote!**

4. Did the UI Page display the response you selected (similar to the example)?

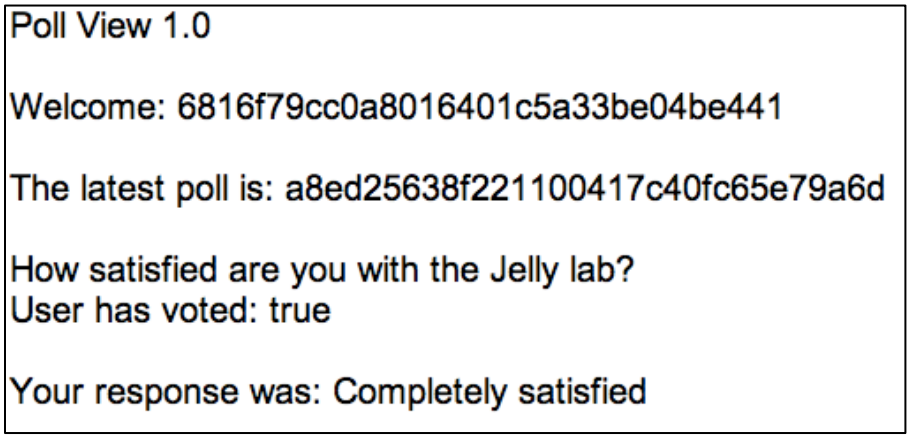


5. If not, go back and determine the source of the problem.
6. What can you note from what you have seen?

Notes

Verification

1. Confirm your screen appears like the one shown below:



Progress Report

1. Navigate to **Lab Management> Report Lab Progress**.
2. Click **I am done!**

Additional Resources

https://wiki.servicenow.com/

https://community.servicenow.com

http://youtube.com/servicenowdemo