Number of Group Members

Lab 8.4 20-25 minutes

Lab objectives

You will perform the following in this lab:

- Create a Script Include which:
  - Extends the AbstractAjaxProcessor
  - Returns the number of members of a group
- Create a Client Script that will:
  - Call a Script Include.
  - Pass a Group name to the Script Include.
  - Parse the response from the Script Include.

A. Create a Script Include

1. Create a new Script Include.

   Name: **AssignmentGroupUtils**
   Client Callable: **Selected (checked)**
   Accessible from: **All application scopes**
   Active: **Selected (checked)**
   Description: **Lab 8.4 extends the AbstractAjaxProcessor class and returns the number of members in a Group.**

2. Examine the pseudo-code for the script:

   - Create a new class called AssignmentGroupUtils.
   - Create an object from the new class with properties inheritable by other objects and which extends the AbstractAjaxProcessor class.
     - Add a method to the new object called countGroupMembers.
       - Create a new variable to store the group name.
       - Store the message that the group has no members in the variable message.
       - Store the sys_id of a group from a Client Script in the variable groupID.
       - Create a new GlideRecord object for the sys_user_grmember table.
       - Query the sys_user_grmember table to return records for all members of the group passed in from a Client Script.

- If there are records returned
  - Store the name of the group in the variable grpName.
- If the value of grpName is not empty
  - Overwrite the current value in the variable message with a string containing the name of the group and number of group members in the variable message.
- Return the variable message to the calling Client Script.

3. Write the script:

```
var AssignmentGroupUtils = Class.create();
AssignmentGroupUtils.prototype = Object.extendsObject(AbstractAjaxProcessor, {
    countGroupMembers: function() {

        var grpName = "";
        var message = "There are no members in this Assignment Group";
        var groupID = this.getParameter('sysparm_group_id');

        var grpMems = new GlideRecord('sys_user_grmember');
        grpMems.addQuery('group.sys_id', groupID);
        grpMems.query();

        if (grpMems.next()) {
            grpName = grpMems.getDisplayValue('group');
        }
        if (grpName != "") {
            message = "There are " + grpMems.getRowCount() + " members in the " + grpName
+ " group";
        }

        return message;
    },
    type: 'AssignmentGroupUtils'
});
```

4. Select **Submit**.

B. Call the Script Include from a Client Script

1. Create a new Client Script.

   Name: **Lab 8.4 Number of Group Members**
   Table: **Incident [incident]**
   Type: **onChange**
   Field name: **Assignment group**
   Active: **Selected (checked)**
   Global: **Selected (checked)**

2. Examine the pseudo-code for the script:
   - Create an instance of the GlideAjax object called AssignmentGroupUtils.
   - Add a param to call the countGroupmembers method.

- Add a param to pass the sys_id of the Assignment group value to the Script Include.
- Use the getXML method and the callback function membersParse to execute the Script Include.
- Pass the response returned from the Script Include to the callback function.
  - Locate the answer variable in the returned XML and store the value in a variable.
  - Generate an alert to display the value of the answer variable.

3. Write the script:

```
function onChange(control, oldValue, newValue, isLoading, isTemplate) {
    if (isLoading || newValue === '') {
        return;
    }

    var membersGA = new GlideAjax('AssignmentGroupUtils');
    membersGA.addParam('sysparm_name', 'countGroupMembers');
    membersGA.addParam('sysparm_group_id', g_form.getValue('assignment_group'));
    membersGA.getXML(membersParse);

    function membersParse(response) {
        var myObj = response.responseXML.documentElement.getAttribute('answer');
        alert(myObj);
    }
}
```

4. Select **Submit**.

C. Test Your Work

1. Navigate to **User Administration > Groups**.
2. Select a group with members assigned to it. Record the group you selected here: _____
3. Open any Incident record.
4. Change the value in the **Assignment group** field to the group you selected in step 2.
5. Did an alert appear? Was it populated correctly? If not, debug and re-test.
6. Change the value in the Assignment group field to **IT Securities**. (The IT Securities group has no members baseline.)
7. Did an alert indicate there are no members in this group? If not, debug and re-test.

D. Reuse the Script Include

1. Can you think of other tables that could use a similar Client Script to call the same Script Include? Write the name of two of those tables here:

   _____

2. Open the **Lab 8.4 Number of Group Members** Client Script.

3. Update the Client Script trigger:

   Name: **Lab 8.4 Number of Group Members (PRB)**
   Table: **Problem [problem]**

4. Select **Insert** on the form's Context menu.

5. Open any Problem record.

6. Change the value in the **Assignment group** field to the group you selected earlier in the lab.

7. Did the alert appear? Was it populated correctly? If not, debug and re-test.

8. Make both the Lab 8.4 Number of Group Members and Lab 8.4 Number of Group Members (PRB) Client Scripts **inactive**.

Lab Completion

You have completed the lab and successfully extended the existing AbstractAjaxProcessor class. The new countGroupMembers() method you created in the AssignmentGroupUtils Script Include uses the passed in Assignment group's sys_id in a GlideRecord query to count the members of that Assignment group and return the information to the client.

You used the GlideAjax class in a Client Script to call the new method and use the value returned from the server in an alert to inform the user of the number of members in their selected group. Time to put these newfound superpowers to work!