Classless Script Includes - logPropertyValues

Lab 8.1 10-15 minutes

Lab objectives

You will perform the following in this lab:

- Create, test and debug a classless Script Include to write logging messages for server-side scripts.

A. Create a Classless Script Include

1. Navigate to **System Definition > Script Includes**.
2. Select **New**.
3. Configure the record:

   Name: **logPropertyValues**
   Accessible from: **All application scopes**
   Description: **Lab 8.1 server-side logging**

4. Delete the Script Include template code stub currently in the *Script* field.
5. Examine the pseudo-code for the script:
   - Create a new function that will be passed a string when it is called.
     - Set the debug property on the object to true.
     - Set the debugPrefix property on the object to <YOUR INITIALS>:
     - If the debug property is true
       - Write a log message containing the debugPrefix property and the passed in string.

6. Write the script:

```
function logPropertyValues(str) {
    this.debug = true;
    this.debugPrefix = "<your_initials>: ";

    if(this.debug){
        gs.info(this.debugPrefix + str);
    }
}
```

7. Select **Submit**.

B. Use the Classless Script Include

1. Create a new Business Rule.

   Name: **Lab 8.1 Script Include Logging**
   Table: **Incident [incident]**
   Active: **Selected (checked)**
   Advanced: **Selected (checked)**
   When: **after**
   Insert: **Selected (checked)**
   Update: **Selected (checked)**

2. Examine the pseudo-code for the script:
   - For the properties in the current object
     - If the property has a value
       - Call the Logging Script Include function with the concatenated string.
       - Set the property and the property value.

3. Write the script:

```
(function executeRule(current, previous /*null when async*/) {

for (var property in current) {
    if (current[property]){
        logPropertyValues(property + ", " + current[property]);
    }
}

})(current, previous);
```

4. Select **Submit**.

C. Test Your Work

1. Create a new Incident and populate the mandatory fields with values of your choice.
2. Select **Submit**.
3. Open **System Logs > System Log > Script Log Statements**. Do the Incident's properties and values appear here? If not, debug and re-test.
4. Modify the Script Include by setting **this.debug = false**.
5. Create a new Incident and populate the fields with values of your choice.
6. **Submit** the Incident.
7. Open **System Logs > System** Log **> Script Log Statements**. Do the new Incident's properties and values appear here? Should they appear? Explain your reasoning:

   _____

   _____

8. Explain why a Script Include to write log data with an "on/off" parameter is useful for debugging scripts:

   _____

9. Make the Lab 8.1 Script Include Logging Business Rule **inactive**.
10. Do you need to make the **logPropertyValues** Script Include inactive? Why or why not? Is there a performance impact if it remains active? Explain your reasoning:

    _____

    _____

Lab Completion

Awesome! You have successfully created a classless Script Include and called it from a Business Rule.