

# Scripting in ServiceNow Fundamentals

---

## Lab Answer Guide

Module 1 - Scripting Overview .....	2
Lab 1.1 Using the Syntax Editor.....	2
Lab 1.2 Syntax Checking.....	2
Lab 1.3 Explore Scripting Resources .....	2
Module 2 - Client Scripts .....	3
Lab 2.1 Two Simple Client Scripts .....	3
Lab 2.2 g_form and g_user .....	4
Lab 2.3 Debugging Client Scripts.....	4
Module 3 - UI Policies.....	6
Lab 3.1 Incident Resolved/Closed UI Policy .....	6
Module 4 - Business Rules.....	5
Lab 4.1 Debugging Business Rules .....	7
Lab 4.2 Current and Previous .....	7
Module 7 - Script Includes .....	8
Lab 7.1 Classless Script Include - logPropertyValues .....	8
Lab 7.1A Classless Script Include - hasRoleExactly .....	8
Lab 7.3 HelloWorld GlideAjax .....	8

**NOTE:** Not all answers are provided in this document (for example, answers to questions asking for your opinion or questions asking you to predict what will happen when an action is performed).

## Module 1 - Scripting Overview

### Lab 1.1 Using the Syntax Editor

**B10 - QUESTION:** Turn off syntax highlighting by selecting the Toggle Syntax Editor button. What differences do you notice in the Syntax Editor?

ANSWER: The Syntax Editor turns into a basic text field. Standard code editing features are no longer available.

**C7a - QUESTION:** Review the description for the 'try' macro. Where did the value come from?

ANSWER: The **Comments** field in the *Editor Macro* record

**C7b - QUESTION:** Review the description for the 'info' macro. Why is it blank?

ANSWER: The **Comments** field in its *Editor Macro* record is empty.

### Lab 1.2 Syntax Checking

**A4/A5 - QUESTION:** Try to save your script. Did it save? How can you tell?

ANSWER: Yes, it saved. The error still appeared after the record was saved.

**A9 - QUESTION:** Did the Syntax Checker find all the errors in the script? Which error(s) were not found?

ANSWER:

- Line 3: Missing quotes around each item in the array. Not "required" but considered best practice.
- Line 3: Extra comma at the end of the array.
- Line 7: Invalid reference to the 'peripheral' array. It should read 'peripherals' to match the variable name declared on line 3.

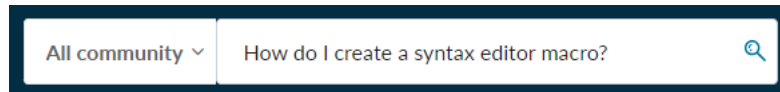
### Lab 1.3 Explore Scripting Resources

**A5 - QUESTION:** Where do you search for existing content in the Community?

ANSWER: Use the Help link at the top-right corner of the page.

**A7 - QUESTION:** How do you ask a question when you cannot find the information you are looking for in the Community?

ANSWER: Go to the Community main page and enter your question in the **All community** form field.

A screenshot of a search bar on a website. On the left, there is a dropdown menu labeled 'All community' with a downward arrow. To the right of the dropdown is a text input field containing the text 'How do I create a syntax editor macro?'. To the right of the text input field is a magnifying glass icon.

**B6 - QUESTION:** Review the *API release notes* article. What information is provided?

ANSWER:

- New scoped classes and additional methods for existing classes
- New global classes and additional methods for existing classes

**C7 - QUESTION:** What Date/Time format does the `GlideDateTime` object use?

ANSWER: Greenwich Mean Time (GMT)

## Module 2 - Client Scripts

### Lab 2.1 Two Simple Client Scripts

**B1 - QUESTION:** Open any Incident record. What happens when the form loads?

ANSWER: An alert window opens with “The form has finished loading and is ready for user input” as its contents.

**B6 - QUESTION:** Does the *Lab 2.1 onLoad Alert Client Script* execute again? Why or why not?

ANSWER: Yes, the Client Script’s trigger is set to execute onLoad, therefore the alert will display every time the form loads.

**B7 - QUESTION:** Create a new Incident. What happens when the form loads for the new incident?

ANSWER: The Client Script’s trigger is set to execute onLoad, therefore when a new Incident opens, the form loads.

**D6 & D11 - QUESTION:** Did the “Resolved” or “Closed” value remain in the *State* field? Why or why not?

ANSWER: No, the script set the value of the **saveAndClose** variable to **false**. This prevented the record from saving any changes when the **callback** function executed. The script will only save the record if the **saveAndClose** variable contains the value **true**.

## Lab 2.2 g\_form and g\_user

**C4 - QUESTION: What happened to the Incident form when you cancelled the submission? Was the Incident submitted?**

ANSWER: Two information messages appear at the top of the form and four field messages appear below the fields identified in the `showFieldMsg()` methods used in the script (category, cmdb\_ci, assignment\_group, short\_description). The JavaScript `confirm()` method returns false when "Cancel" is selected. This stops the submission of the Incident record.

**C5 - QUESTION: Modify the confirmation dialog box in the script so the confirmation message includes the logged in user's first name. Which property do you need to use?**

ANSWER: `g_user.firstName`

**C9 - QUESTION: Create a new Priority-1 Incident and Save (not Submit) the record. Did the confirmation message display? Explain why not.**

ANSWER: No, the script checks first to determine if the logged in user creating the record does NOT have the **major\_inc\_mgr** role. If they have the role, the script does not execute.

## Lab 2.3 Debugging Client Scripts

**D3 - QUESTION and ANSWER: Review the alert message(s).**

- **What is the value currently in the 'incState' variable?** The variable is empty.
- **What should it be?** 1
- **Does the string "LINE 11 EXECUTED" appear as a logged message?** No

**D4 - QUESTION: What can you conclude regarding the debugging output currently in the alert?**

ANSWER: Something is not right with the `incState` variable and because the alert message does not contain the reference to Line 11 executing, the error occurs prior to this line of code in the script.

**D5 - QUESTION: At this point, you know something is wrong with the value in the `incState` variable. Before leaving the Incident form, double-check the State field name by right-clicking the field's label. What is the exact spelling of the State field name?**

ANSWER: `state`

**D7 - QUESTION: Review the State field name in the "`g_form.getValue('incident');`" statement. Does it match the field name you documented in step 7?**

ANSWER: No, the State field's name is not correct. It should read 'state' not 'incident'.

**D15 - QUESTION:** Review the InfoMessage at the top of the form. Are you the only person who can see this script output?

ANSWER: No, everyone can see it.

**D16 - QUESTION:** Form messages are a good debugging strategy as the results are instantly presented at the top of the form you are testing on. Would this type of debugging strategy be best in a development or production instance?

ANSWER: Development, the statement should be removed before moving the script to a production instance.

## Module 3 - UI Policies

### Lab 3.1 Incident Resolved/Closed UI Policy

**B11 - QUESTION:** Does the 'Reverse if false' field need to be selected for the 'Execute if false' script to execute?

ANSWER: Yes, it does. The script in the **Execute if false** field will never execute if the **Reverse if false** field is not selected.

**B12 - QUESTION:** When the 'Reverse if false' field is selected and the 'Execute if false' field is empty, does the reverse of what is scripted in the 'Execute if true' field occur?

ANSWER: No, it does not. If the **Reverse if false** field is selected and the **Execute if false** field is empty, a script will not execute if the UI Policy's conditions are not met.

## Module 4 - Business Rules

### Lab 4.1 Debugging Business Rules

**B4 - QUESTION:** What can you conclude about where breakpoints can be set?

ANSWER: Breakpoints can be set in any scriptable field with the full editor as well the script field in the debugger itself.

**B7 - QUESTION:** Script execution is paused at the first breakpoint. How can you tell?

ANSWER: The number in the gutter and the line of code are highlighted in red.

**B11 - QUESTION:** Explain why the 'myNum' variable value is sometimes *undefined* and other times contains a value?

ANSWER: When the Script Debugger pauses at a breakpoint, it stops the script before the statement it is paused at executes. In this lab, the first breakpoint paused the script before the value of the **myNum** variable was set, therefore it is *undefined*. Once the script continued to the next breakpoint, the value of the variable was set to the value current in the State field.

**C4b, C5b, C6 - QUESTION:** Document the "previous" object's *short\_description* field value and the "current" object's *short\_description* field value. Why are they different?

ANSWER: **current** is an object that stores the current record's fields and values, and **previous** is an object that stores the record's fields and values before any changes were made.

**D11, D14, & D17 - QUESTION: How can you be sure which script you are currently debugging?**

ANSWER: The Code Pane Header reads *Business Rule > Lab 5.1 Business Rule Debugging for D11 and D17 Script Include > SlaTargetNotification for D14*.

**E5 - QUESTION: Which undefined function produced an error and why?**

ANSWER: The *thisFunctionDoesNotExist()* function produced an error because it was scripted inside a **try/catch**. The error handling scripted in the **catch** block uses the global **gs.log()** method to produce the error output.

**E6 - QUESTION: Was a log message produced for the undefined *thisFunctionAlsoDoesNotExist()* function?**

ANSWER: No

**F8 - QUESTION: How do these debugging results compare to the GlideSystem logging method results?**

ANSWER: **Script Tracer** displayed both errors (lines 8 and 9), while **GlideSystem** logging ignored the *thisFunctionAlsoDoesNotExist()* function since it was not inside a **try/catch** statement.

**G2 - Select the Session Log tab and search for the execution of the Lab 5.1 Business Rule Debugging Business Rule by searching for the string ==> Lab 5.1 Business Rule Debugging. Did your test meet the Business Rule's Condition criteria? How can you tell?**

ANSWER: It did not meet the criteria and the debugging output advises it skipped the Business Rule because the condition *current.state != 7* was not satisfied. An excellent strategy to see if your Business Rule executed.

## Lab 4.2 Current and Previous

### **C8 - QUESTION: Does the RCA included field need to remain visible on the form?**

ANSWER: It does not. The Business Rule will take care of ensuring it is populated properly. Manual updates are not needed. The field can then be used to query the table for a list of records with RCA details included.

## **Module 7 - Script Includes**

### Lab 7.1 Classless Script Include - logPropertyValues

#### **C7 - QUESTION: After setting this.debug = false, does the new Incident's properties and values appear in the list of Script Log Statements?**

ANSWER: No, the 'if' statement in the Script Include first checks to determine if the value of this.debug is equal to "true". If yes, the script will write a log message containing the debugPrefix property and the passed in string. If no, (which it is not at this point in the lab as this.debug was just updated to equal false), no further script logic executes.

#### **C8 - QUESTION: Explain why a Script Include to write log data with an "on/off" parameter is useful for debugging scripts.**

ANSWER: It is another quick and easy way to review the values of every field in the record. The on/off parameter makes activation and de-activation of the creation of logs very easy and if you save the Script Log Statements module as a favorite, getting to the log files is quite quick too.

#### **C10 - QUESTION: Do you need to make the logPropertyValues Script Include inactive? Why or why not? Is there a performance impact if it remains active?**

ANSWER: No, you do not need to make the Script Include inactive. It only runs when called, otherwise it does nothing. If the Business Rule is marked inactive, the Script Include will not execute.

### Lab 7.1A Classless Script Include - hasRoleExactly

#### **C7 - QUESTION: Do you need to make the hasRoleExactly Script Include inactive? Why or why not? Is there a performance impact if it remains active?**

ANSWER: No, you do not need to make the Script Include inactive. It only runs when called, otherwise it does nothing. If the Business Rule is marked inactive, the Script Include will not execute.

### Lab 7.3 HelloWorld GlideAjax

**B5 - QUESTION: The first parameter in the script is 'sysparm\_name', which is a reserved parameter name. What information does it pass to the Script Include?**

ANSWER: **sysparm\_name** passes the name of the method/function you want to use.

**B6 - QUESTION: Other than beginning with a “sysparm\_” prefix, do additional parameters have to use reserved parameter names?**

ANSWER: They do not. If the parameter name begins with **sysparm\_** and has no spaces or special characters, you can give the parameter whatever name you like. It is considered best practice for parameter names to describe the data being passed to the Script Include (for example, *sysparm\_start\_date*, *sysparm\_user\_name*, *sysparm\_manufacturer*).