October 28, 2013

Action Builder

If you need the benefits of scripting but don't have the time to master JavaScript, Designer's Action Builder may be the tool you're looking for. Action Builder is a script-creating assistant that you can access by selecting Tools > Action Builder **(Figure 1.1)**. You select form objects, conditions, and results and Action Builder generates the script for you. For instance, you can select a Button object and indicate that you want an action performed when the button is clicked. You can define the result of the action as a message box. Action Builder will generate the script and add it to your Button object's `click` event in the Script Editor. When a user clicks the button at runtime, the pop-up message box will appear.

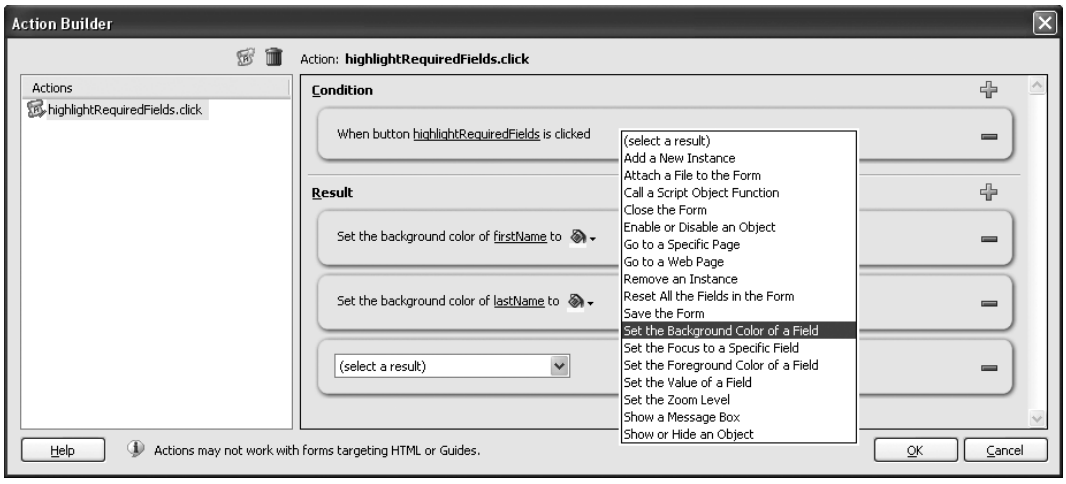

Figure 1.1 The Action Builder dialog box showing a list of functions you can use to easily add automated functionality to your forms.

Follow these steps to use the Action Builder to create a script that will display a message box.

1. Open the changeOfBeneficiaryActions.xdp form from the Samples folder.
2. Select Tools > Action Builder to open the Action Builder dialog.
3. Click the Add a new action button
4. Double click on *(New Action)* in the Actions list and rename the action to *Submit Form*.
5. Click *(object)* in the first Condition.
6. Select the `submitForm` button in the Select an Object dialog and click OK. A condition has been created.
7. Click the green plus sign next to Condition to add a second condition.
8. Click *(object)* in the second Condition.
9. Select the `approved` check box in the Select an Object dialog and click OK. A second condition has been created. A warning has also been added to the two conditions to let you know that you now have multiple trigger conditions for one action.
10. Change *is clicked* to *is checked* in the second condition. By changing this condition you are eliminating the second trigger condition, therefore the warning is removed.
11. Select *Show a Message Box* from the Result drop down.
12. Enter **Your form is ready to be submitted** as the message.
13. Enter **Ready for submission** as the title and click OK. You can also change the message box icon here if you wish.
14. Select Preview PDF to see your script in action.
15. Select the *I approve these updates* check box.
16. Click the Submit Form button. You will see the message box.

Exercise 2

1. Create a showRequiredFields button on the header subform.
2. Select Tools > Action Builder.
3. Click Create a new action called **showRequiredFields**.
4. Create a condition by clicking the object link.
5. Select the showRequiredFields Button object and click OK.
6. Create a result by selecting Set the Background Color of a Field in the dropdown list.
7. Click the object link and select the primaryName object.
8. Select yellow in the drop-down color list.
9. Click OK to complete your action. Designer automatically adds custom JavaScript to your button's click event.
10. Select the showRequiredFields button and review the JavaScript in the button's click event.
11. Select the Preview PDF tab to see the new script in action. When you click the showRequiredFields button, the primaryName field changes to yellow.

12. You can add additional results to the same condition by returning to the Action Builder. Select Design View and click Tools > Action Builder.
13. Select the showRequiredFields action.
14. Click the green plus sign on the right to add a new result. Repeat steps 6–8 for the other fields in the primaryItem subform, and click OK in the Action Builder dialog box when you are done.
15. Select the showRequiredFields button and review the JavaScript in the button's click event.
16. Select the Preview PDF tab to see the new script in action.

Exercise 3

1. Create a changeColor button in the signature subform.
2. Create a checkRequiredFields button in the signature subform.
3. Use the Action Builder to change the background color of the policyNumber field to yellow when the changeColor button is clicked.
4. Use the Action Builder to validate the form for required fields when the checkRequiredFields button is clicked.

Action Builder Scripts are less efficient

Limit the number of times you call the `.resolveNode()` and `.resolveNodes()` methods in your scripts. Call these methods as few times as possible, and store the values you retrieve in a form or local variable. It's more efficient to retrieve a value from a variable than it is to call the methods to reevaluate the expressions again.

Avoid using using the `..` token (double periods) within the `.resolveNode()` and `.resolveNodes()` methods. When you're trying to access descendant nodes, it's better to write out as much of the path that's known before using this token. For instance, the expression `xfa.resolveNode("expenses..subTotal").rawValue` on the page1 node of the SmartDoc Expense Report will iterate through most of form's objects before locating the `subTotal` field. This expression is more efficient: `expenses.total.subTotal.rawValue`.