

Lesson 13. Enable/Disable actions at runtime

One of the exercises of the last lesson asks you to make an action list to be assigned to the – (subtraction) button. We can do the same thing for the * (multiply) and the / (divide) button. It works fine, basically. But it has a defect. Click the / button, then click the + button. Now the calculator will result in infinite number. Click + or some other calculation key, and it displays Infinite or NaN. The problem is caused by the / button. When the / button is clicked, it sets the formula to $\text{Result} / \text{UserInput}$ and sets `UserInput` to 0. When the + button is clicked, it does the calculation $\text{Result} / \text{UserInput}$, and thus perform an action of dividing by 0 because `UserInput` is 0.

It is a programming defect. When the / button is clicked and then the + button is clicked, the user is changing his/her mind, not to do /, but to do + instead. So we should not carry out the division action. The division should be done only when the user clicks some number buttons.

That is, the pending calculation method must be confirmed by a numeric button.

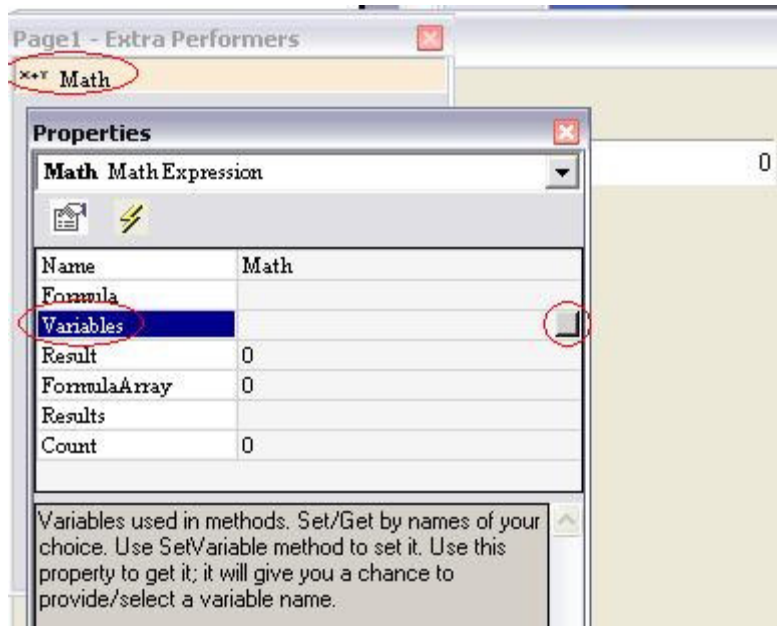
Limnor lets you enable and disable actions at runtime. Using this feature, we can disable the calculation action when it is set, and then enable the action when a number key is clicked. We can use a flag to signal the enabling/disabling of the calculation actions.

13.1. Use variables to enable/disable actions

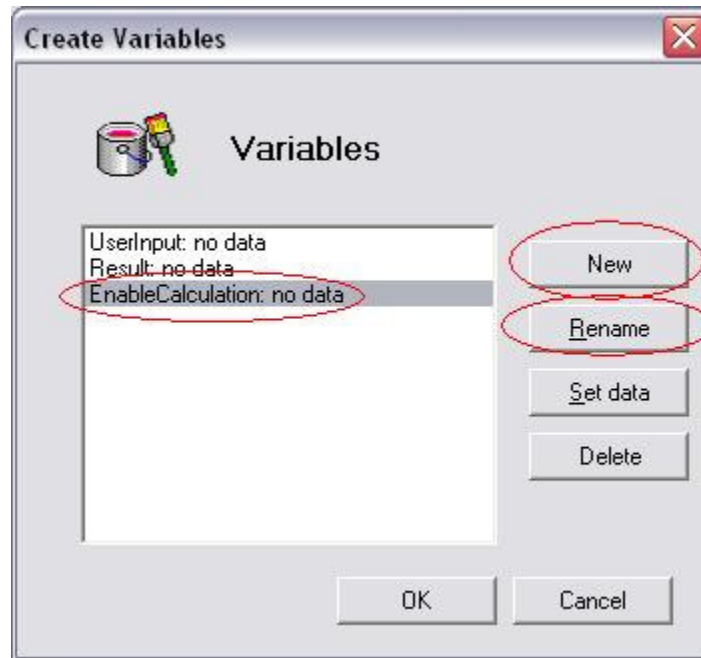
Recall that the `ClickAdd` action list has the following actions:

- i) `Math.Evaluate`
- ii) `LabelDisplay.SetTextResult`
- iii) `Math.SetVariableClearUserInput`
- iv) `Math.SetFormulaAddition`

Let's modify the `Math.Evaluate` action to make it enabled/disabled by a flag. First, let's create a new variable named "EnableCalculation" for this purpose. Open the Properties window for the Math Expression, set its Variables property:



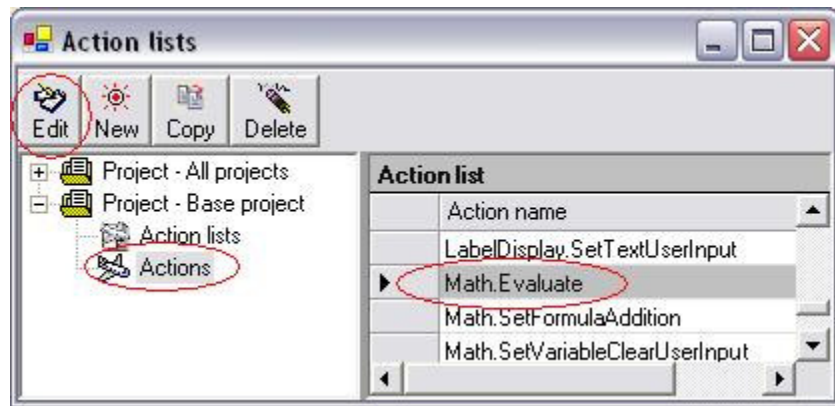
Click Add to create a new variable. Click Rename to name it EnableCalculation:



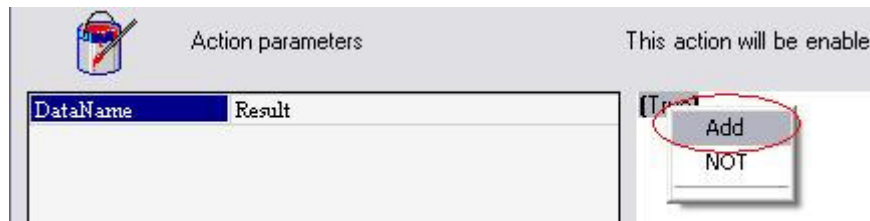
Now we may modify the action Math.Evaluate to make it enabled by variable EnableCalculation.



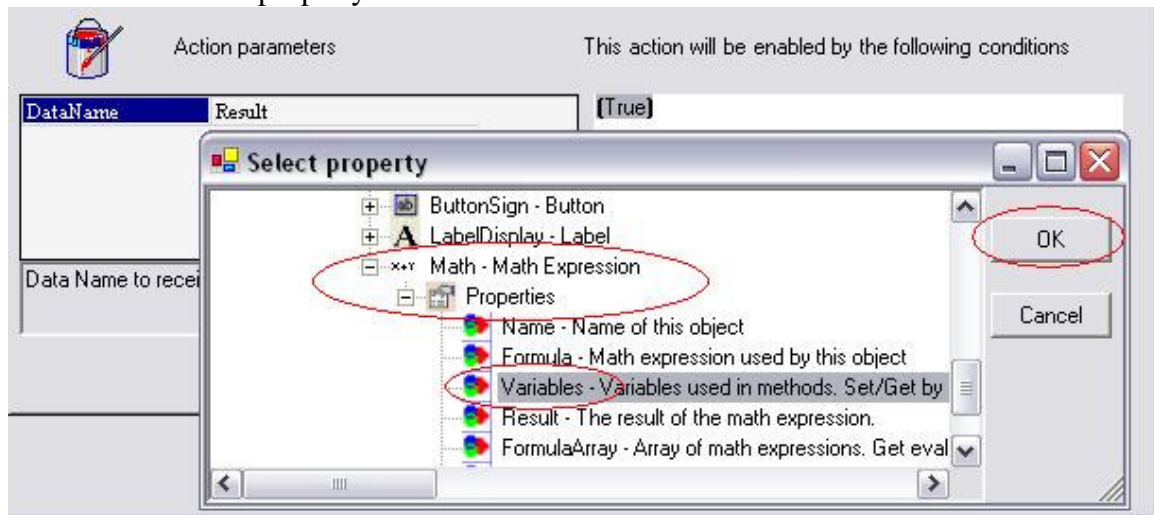
To modify an action, click “Actions” to bring up the Action lists window. Select Actions, pick Math.Evaluate, and click Edit:



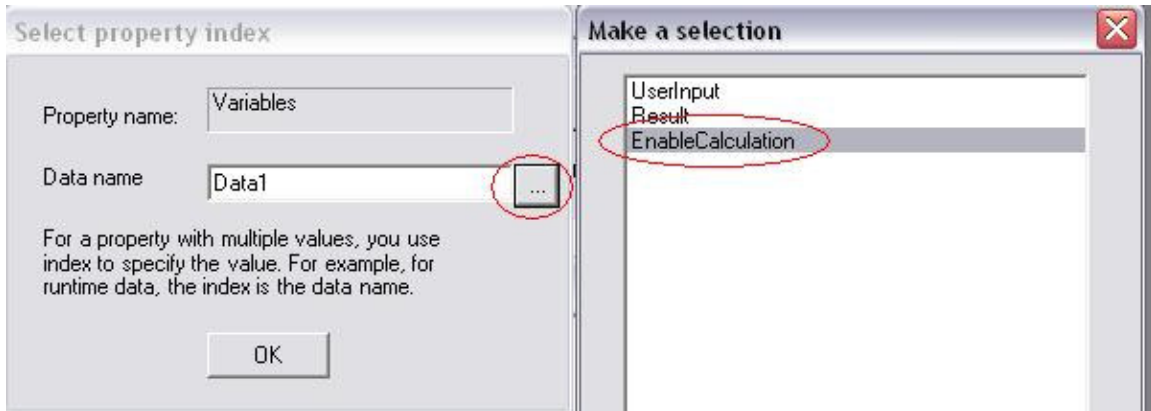
When a dialog box appears, click Next, and right-click on the action condition, select “Add”:



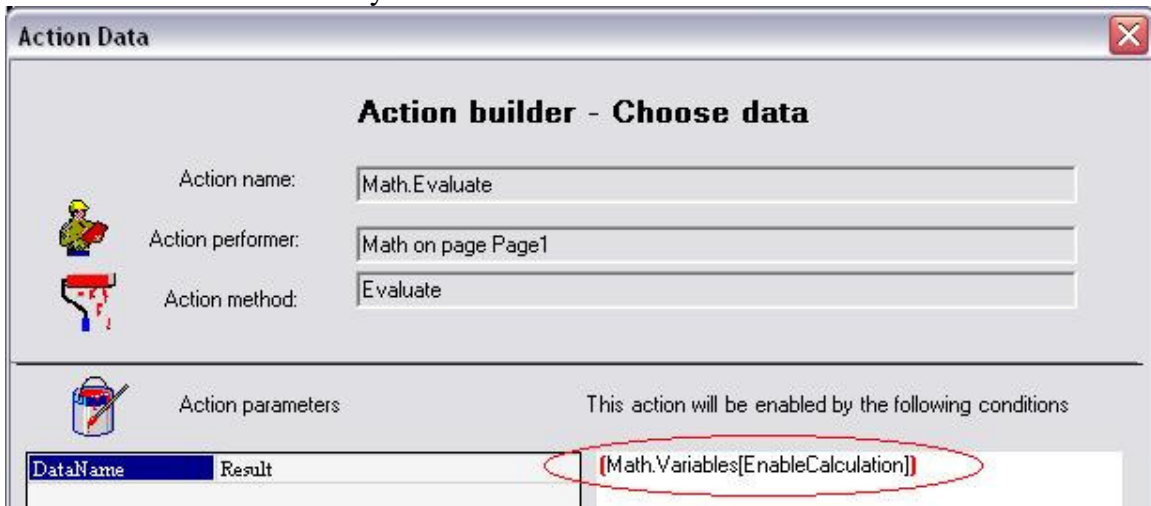
It will let you select a property to be used as the flag to enable/disable this action. We need to select the variable EnableCalculation. Find the Math Expression under Page 1, select its Variables property:



Click  and select EnableCalculation:



Now this action is enabled by this variable:

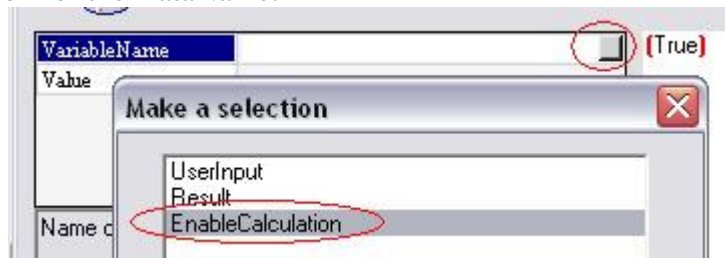


We need to do the same modifications for LabelDisplay.SetTextResult and Math.SetVariableClearUserInput actions.

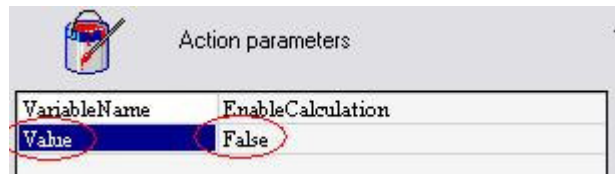
13.2 Create actions to switch the enable/disable flag

Now we need a new action to set “EnableCalculation” to False to disable these first 3 actions in the ClickAdd.

Because EnableCalculation is saved in the Variables property of the Math Expression, we need to use Math Expression’s SetVariable method to make this action. Right-click on the Math Expression; select “Make action” menu, select “SetVariable”, give a name, say, [Math.SetVariableDisableActions](#). Click and select EnableCalculation for the DataName:

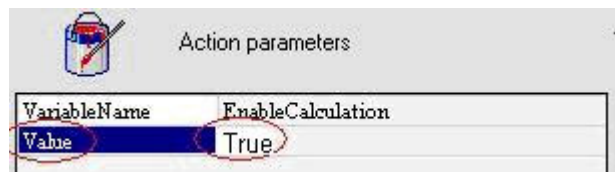


Type “False” for the Value:



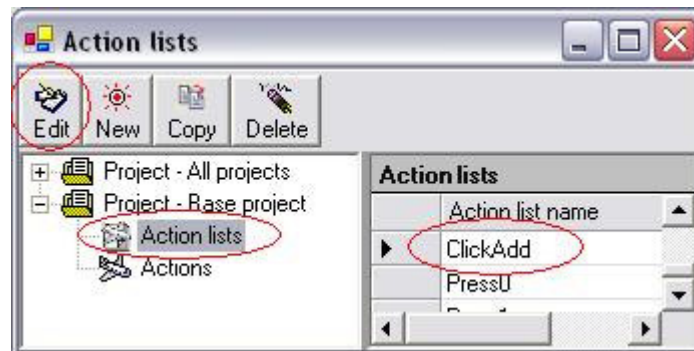
Now we need a new action to set “EnableCalculation” to True to enable these first 3 actions in the ClickAdd.

Right-click on the Math Expression; select “Make action” menu, select “SetVariable” method, give a name, say, [Math.SetVariableEnableActions](#). Select EnableCalculation for the DataName, type “True” for the Value:

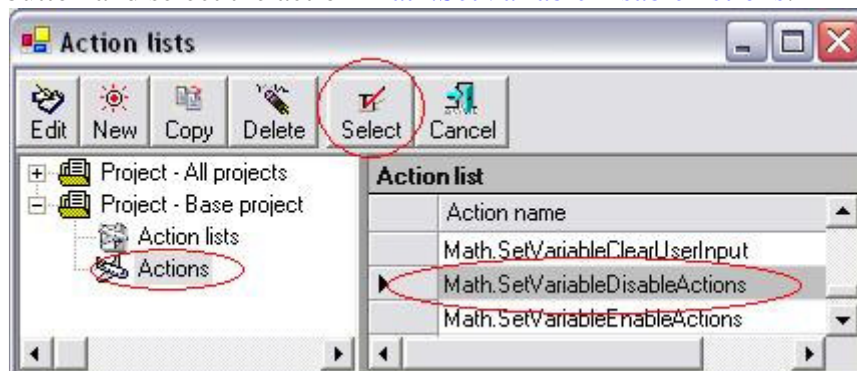


13.3. Enable/disable actions at right moment

When a calculation button is clicked, we need to disable the actions; when a number button is clicked, we need to enable the actions. The following example adds [Math.SetVariableDisableActions](#) action to the action list ClickAdd: Click “Edit” to modify the ClickAdd action list:

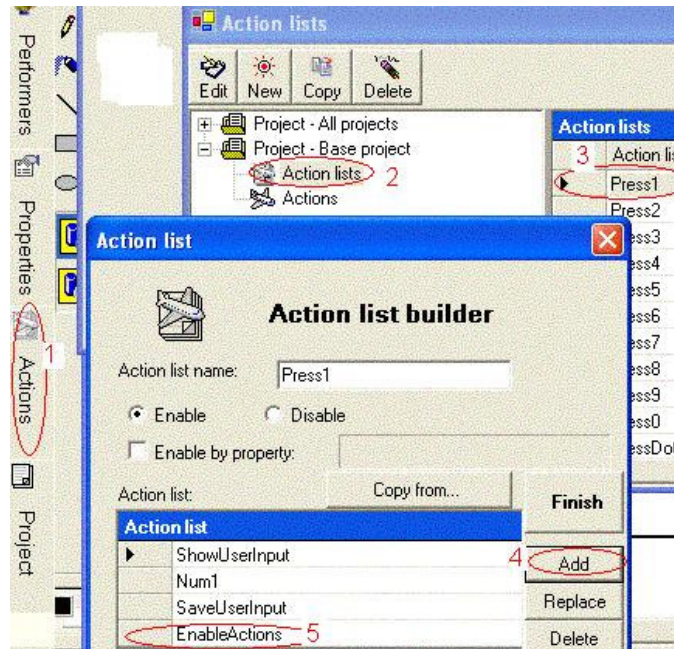


Click Add button and select the action [Math.SetVariableDisableActions](#):



[Math.SetVariableDisableActions](#) should also be added to other action lists assigned to other calculation buttons (-, *, /, =).

The following example adds the [Math.SetVariableEnableActions](#) action to the action list Press1 which is assigned the Click event of button 1:



[Math.SetVariableEnableActions](#) should also be added to the other action lists assigned to other number buttons.

Now Press F2 to try the program.

13.4. What we learned in this lesson

- Every action can be enabled and disabled at runtime. (Every action list can too. This lesson does not involve enabling/disabling action lists.)
- Any performer property can be used as a flag to enable/disable actions. This lesson uses a value in the Data property of the Math Expression as such a flag.
- If the property value is True, then the actions are enabled; if the property value is False, the actions are disabled. (For number properties, if the property value is 0, the actions are disabled, if the value is not 0, the actions are enabled. This lesson does not use number values for the enabling/disabling flag.)
- At runtime, we enable/disable actions by setting the property which is used as the flag. They are the two actions named EnableActions and DisableActions in this lesson. Adding these actions to the proper action lists makes the program enable and disable actions at the right time.

For full description of adding conditions to actions and action lists please see document “ActionConditions.doc”. It is available at

<http://www.limnor.com/downloads/ActionConditions.doc>.

For more examples, see <http://www.limnor.com/downloads/UseMessageBox.doc>

13.5 Exercises

Exercise1.

In this lesson, we showed how to make the action Calculate to be enabled/disabled by the Math Expression’s runtime data named EnableCalculation. Please do exactly the same thing for ShowResult and ClearUserInput actions.

Exercise2.

In this lesson, we added the action DisableActions to the action list ClickAdd which is assigned to the Click event of the + button. Please add DisableActions to other action lists assigned to other calculation buttons (-, *, /, =).

Exercise3.

In this lesson, we added the action EnableActions to the action list Press1 which is assigned to the Click event of the “1” button. Please add EnableActions to other action lists assigned to other number buttons (2,3,4,5,6,7,8,9,0,.,+/-).

Exercise4

If the user using this calculator program tries to divide by 0, the calculator will always show Infinite and cannot do other calculations any more. This is because the Result (saved in the Math Expression’s runtime data named Result) is infinite. Create a CE to clear the Result and UserInput so that the calculator is back to normal again.

Hint: Use Math Expression’s SetData method to create two actions; one to set Result to empty; the other set UserInput to empty (we already have this action, it is named ClearUserInput). We need another new action which sets the math formula to UserInput. Use these 3 actions to make an action list. Assign the action list to the CE button’s Click event.