

Message Box and Conditional Execution



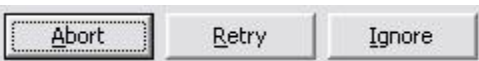


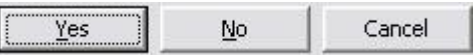
1	Introduction.....	1
2	Use of Message Box	1
3	Conditional Execution	5
4	Questions and Feedbacks.....	10

1 Introduction

This sample application shows how to do conditional execution as IF ... THEN ... ELSE does in computer languages. It uses Message Box to accept user input as the branching condition.

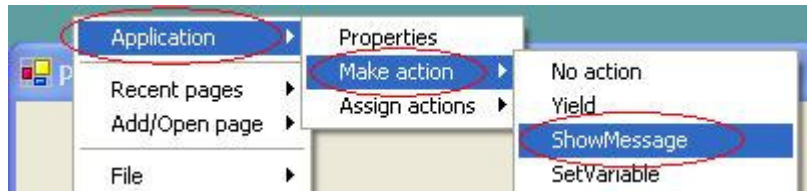
2 Use of Message Box

The Application Performer has a ShowMessage method. It will show a message box. When you create actions using this method, you may specify what buttons you want display on the message box. The user clicks one of a few buttons on the message box to close it. Properties MsgYesRetryOK and MsgIgnore indicate which button the user clicked. The following table shows the values of MsgYesRetryOK and MsgIgnore corresponding to the button the user clicked:

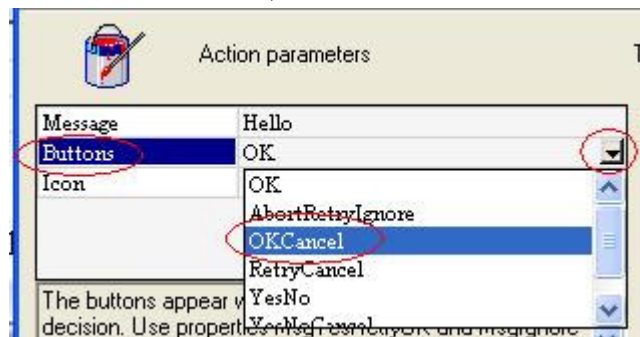
Buttons on box	Button clicked	MsgYesRetryOK	MsgIgnore
	OK	True	False
	OK	True	False
	Cancel	False	False
	Abort	False	False
	Retry	True	False
	Ignore	False	True
	Retry	True	False
	Cancel	False	False
	Yes	True	False
	No	False	False
	Yes	True	False
	No	False	False
	Cancel	False	True

Message Box and Conditional Execution

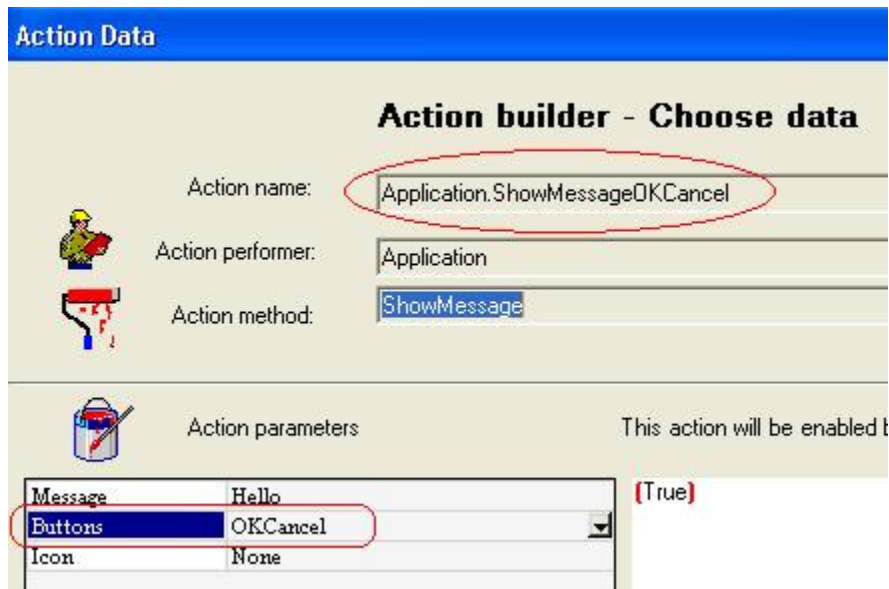
To demonstrate the above table, we create an action for each of the above cases. Let's create an action for showing a message box with two buttons: OK and Cancel. Right-click on a page or background, choose "Application"; choose "Make action"; choose "ShowMessage":



Give an action name, say, Application.ShowMessageOKCancel. The Action Data dialogue box appears. Parameter Message is the message text you want to display on the message box. In this sample, we just say "Hello". Parameter Buttons indicates what buttons to use on the message box. Click to choose the buttons. For this action, we choose "OKCancel":

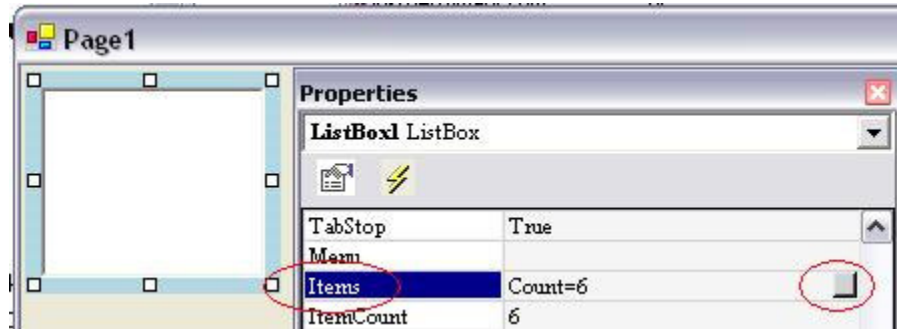


The action looks like:

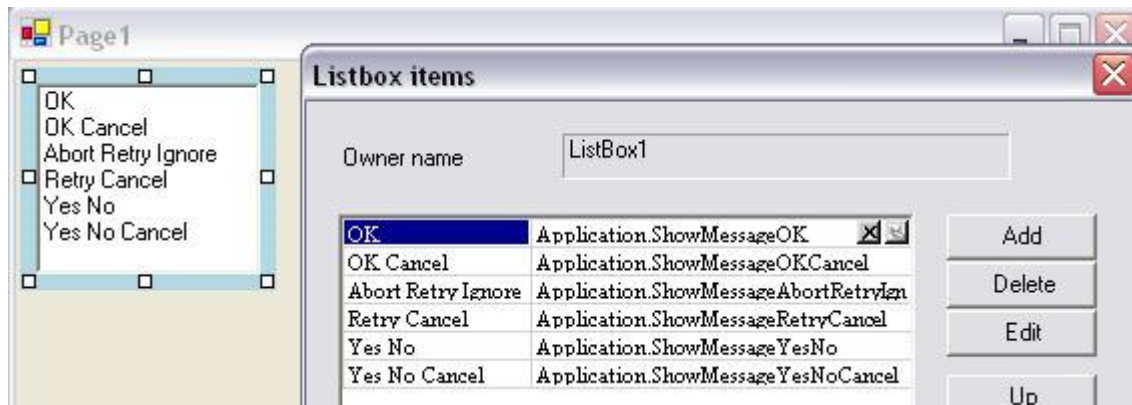


In the same way, we may create 5 other actions using different Buttons parameter. We use a list box to show these actions, by setting the Items property of the list box:

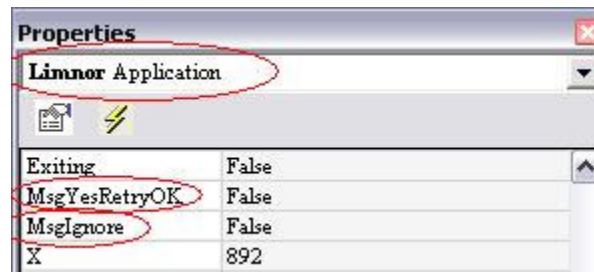
Message Box and Conditional Execution



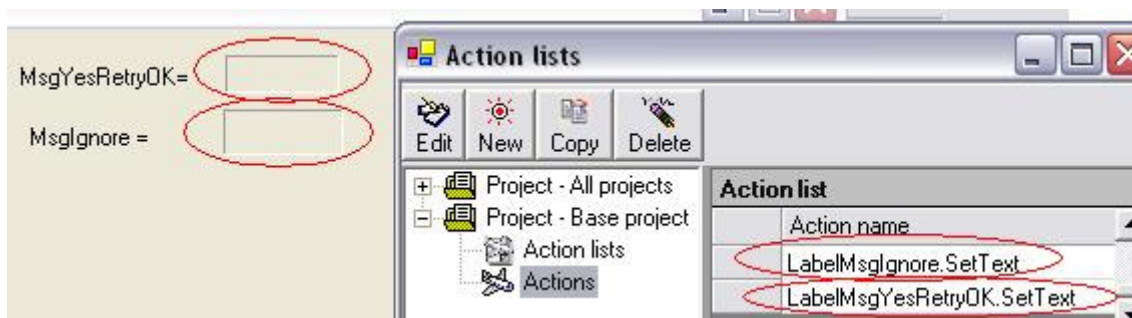
By assigning one action to an item of the list box, the action will be executed when the item is clicked:



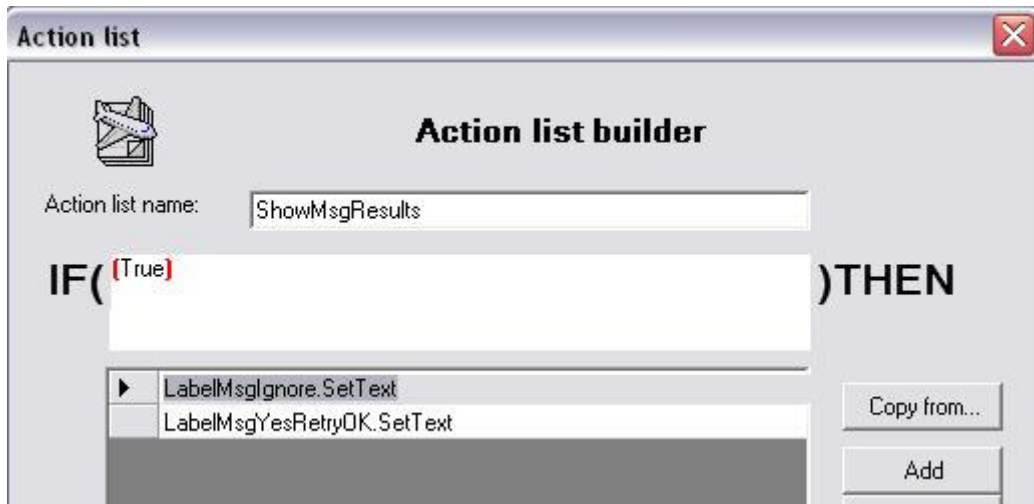
Now when clicking an item of the list box, the corresponding action will be executed, showing a message box with desired buttons. After the message box is closed, property **MsgYesRetryOK** and **MsgIgnore** will indicate which button is used to close the message box:



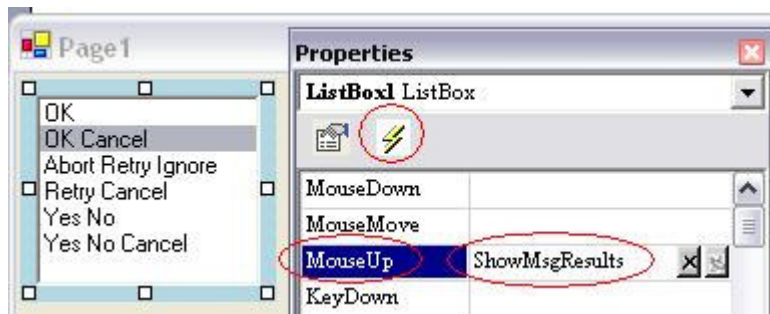
We use two Labels to display these two properties:



We create an action list to include these two actions:



And assign this action list to the MouseUp event of the list box because MouseUp event occurs after Click event for clicking an item:



In this arrangement, when the application runs, clicking on a list item, a message box appears, clicking a button on the message box, the message box disappears and the values of the properties MsgYesRetryOK and MsgIgnore are displayed on two labels.

Let's try it. Click on "Yes No", we get a message box:



Click "Yes" button, the message box disappears; the values of MsgYesRetryOK and MsgIgnore are displayed on two labels:



Now let's click on "Yes No" again, the message box appears again. This time we click on No button, the values of MsgYesRetryOK and MsgIgnore become:



You can see that MsgYesRetryOK becomes True if Yes button is clicked; MsgYesRetryOK becomes False if No button is clicked. You can see for both cases, the value of MsgIgnore is always False. The value of MsgIgnore will become True if Ignore button is clicked or if Cancel button is clicked when Yes/No/Cancel buttons are used.

Now we know that values of MsgYesRetryOK and MsgIgnore indicate which button the user clicked on the message box, we may control what actions to execute based on which button the user clicked. The way to do it is to use the value of MsgYesRetryOK or MsgIgnore to enable or disable actions. We will show it below.

3 Conditional Execution

Many of you are familiar with programming logic IF...THEN...ELSE, as all programming languages use it as a basic way of conditional execution.

How do we do it in a codeless environment like Limnor?

In Limnor, each action can be enabled and disabled by properties. But in many cases the control can be done in the level of action list, not individual action.

Each action list may have two parts and conditions to form such logic:

IF (conditions) THEN

 execute part 1

ELSE

 execute part 2.

The table below shows an example of such logic.

Programming language (pseudo code)	Limnor Programming (Action list)
------------------------------------	----------------------------------

Message Box and Conditional Execution

Show Message Box IF (Condition) THEN code-part A ELSE code-part B	Action: Show Message Box Action List: IF(Condition) THEN Action-list-A ELSE Action-list-B
---	---

In Limnor, the (Condition) to enable/disable actions is a property or properties.

In this sample application, we use the property `MsgYesRetryOK` of the Application Performer as the (Condition).

We use such a programming logic to do the demonstration:

1. Show a message box with Yes and No button.
2. If the user clicks Yes button then do two things:
 - a. show a message box saying “Yes button clicked”;
 - b. display a sentence “Yes button clicked” in a text box.
3. If the user clicks No button then do two things:
 - a. show a message box saying “No button clicked”;
 - b. display a sentence “No button clicked” in a text box.

Use a pseudo code the programming logic can be expressed as

```
IF (User clicks YES button ) THEN
    Show Message Box(“Yes button clicked”)
    TextBox1.Text = “Yes button clicked”
ELSE
    Show Message Box(“No button clicked”)
    TextBox1.Text = “No button clicked”
```

We build an action list named `OnUserClick` to achieve the above code.

The condition (User clicks YES button) in Limnor can be expressed as “The `MsgYesRetryOK` property of Application Performer is True”: `Application.MsgYesRetryOK = True`.

Click the condition to highlight the word “True” and then right-click on it, choose “Add”:

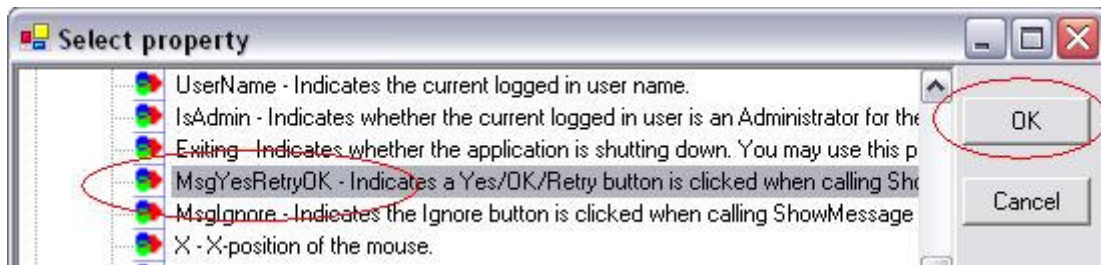


Expend Properties node under Application:

Message Box and Conditional Execution

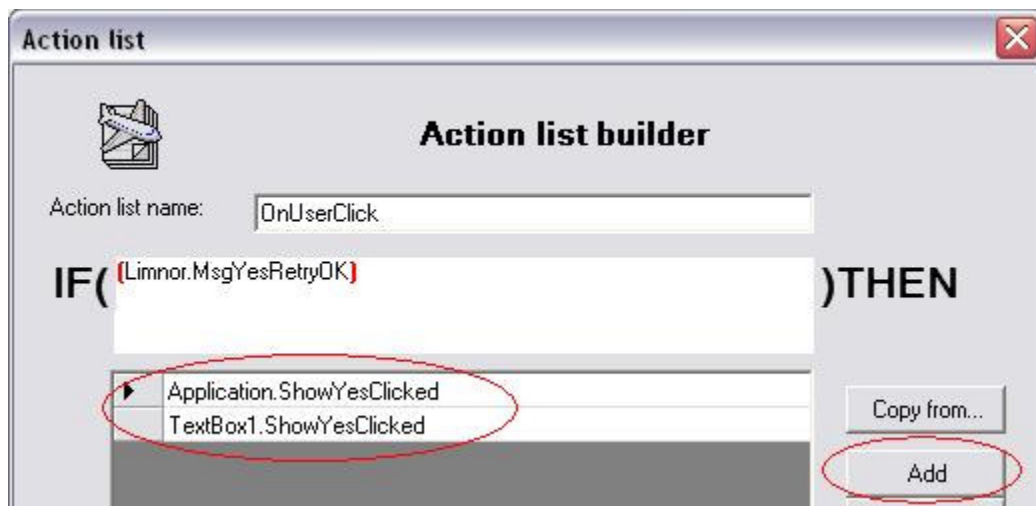



Select property MsgYesRetryOK:



We are done using property MsgYesRetryOK as the Condition.

Click Add button to add two actions for the IF part of the action list:



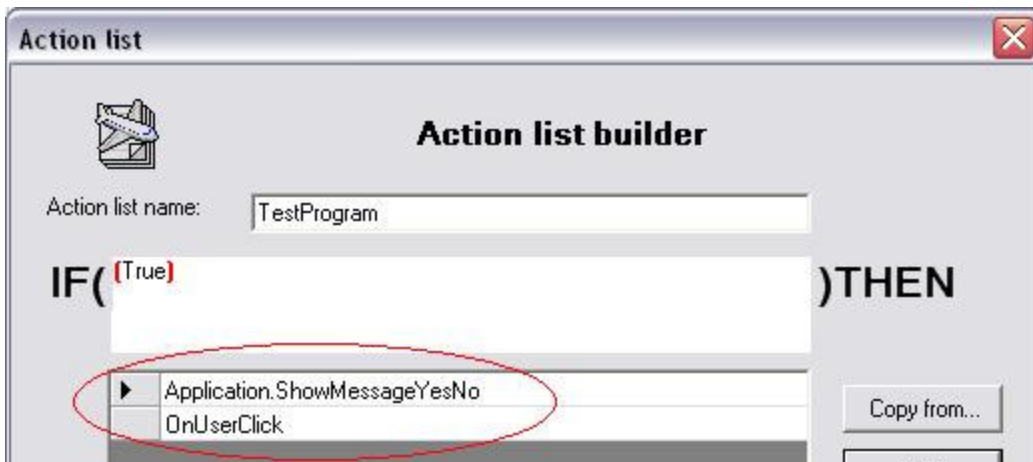
Click  to bring up the action list for ELSE part:



Click Add button to add two actions for the ELSE part of the action list:



Now we put action Application.ShowMessageYesNo and action list OnUserClick together to form a new action list:



We are done. We did not show the creation of the 4 actions: Application.ShowMessageYesClicked, TextBox1.ShowYesClicked, Application.ShowMessageNoClicked, and TextBox1.ShowNoClicked. We assume it is easy for you to do. If you have questions please contact support@limnor.com.

Let's copy the program requirements we stated at the beginning of this section:

Programming language (pseudo code)	Limnor Programming (Action list)
Show Message Box	Action: Show Message Box
IF (Condition) THEN	Action List: IF(Condition) THEN
code-part A	Action-list-A
ELSE	ELSE
code-part B	Action-list-B

The action list, TestProgram, is the implementation for the above logic.

The action Application.ShowMessageYesNo is the implementation of "Show Message Box".

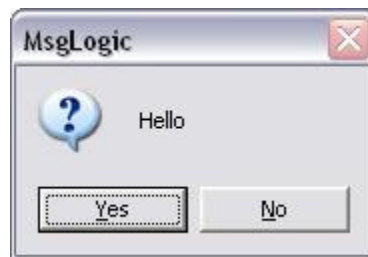
Message Box and Conditional Execution

The action list OnUserClick is the implementation of
IF(Condition) THEN
 Action-list-A
ELSE
 Action-list-B

To test this program, we add a button and assign this action list to the Click event of the button:



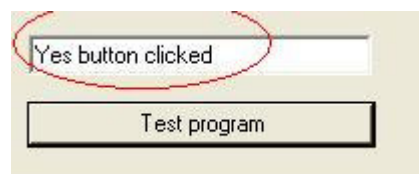
Now we may run the sample and click the button to test it. Click “Test program” button. Action list “TestProgram” is executed. A message box will appear with a Yes button and a No button because action Application.ShowMessageYesNo is executed:



Depending on which button you click, you will see different results. Suppose you click Yes button. The message box disappears. The next action in the action list “TestProgram” will be executed: OnUserClick. This is an action list and controlled by property MsgYesRetryOK. Since you clicked Yes button, the value of MsgYesRetryOK is True. That means the two actions for the “IF” part, Application.ShowYesClicked and TextBox1.ShowYesClicked, are executed. Application.ShowYesClicked is executed first and we see a message box comes up:



Click OK to dismiss this message box, the next action, TextBox1.ShowYesClicked, is executed:

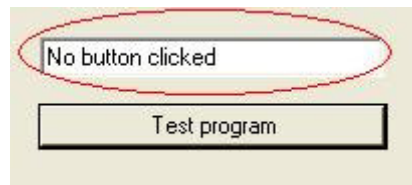


Message Box and Conditional Execution

This finishes executing action list OnUserClick. And thus the action list TestProgram finishes. In this case, actions Application.ShowYesClicked and TextBox1.ShowYesClicked are executed; Application.ShowNoClicked and TextBox1.ShowNoClicked are not executed; because the value of MsgYesRetryOK is True. That is called the conditional executions. Now let's click button "Test program" again, and message box appears again:



This time, we click No button. Now the value of MsgYesRetryOK will be False. Thus actions Application.ShowYesClicked and TextBox1.ShowYesClicked are not executed; Application.ShowNoClicked and TextBox1.ShowNoClicked for the "ELSE" part are executed:



4 Questions and Feedbacks

For questions and feedbacks, please contact support@limnor.com