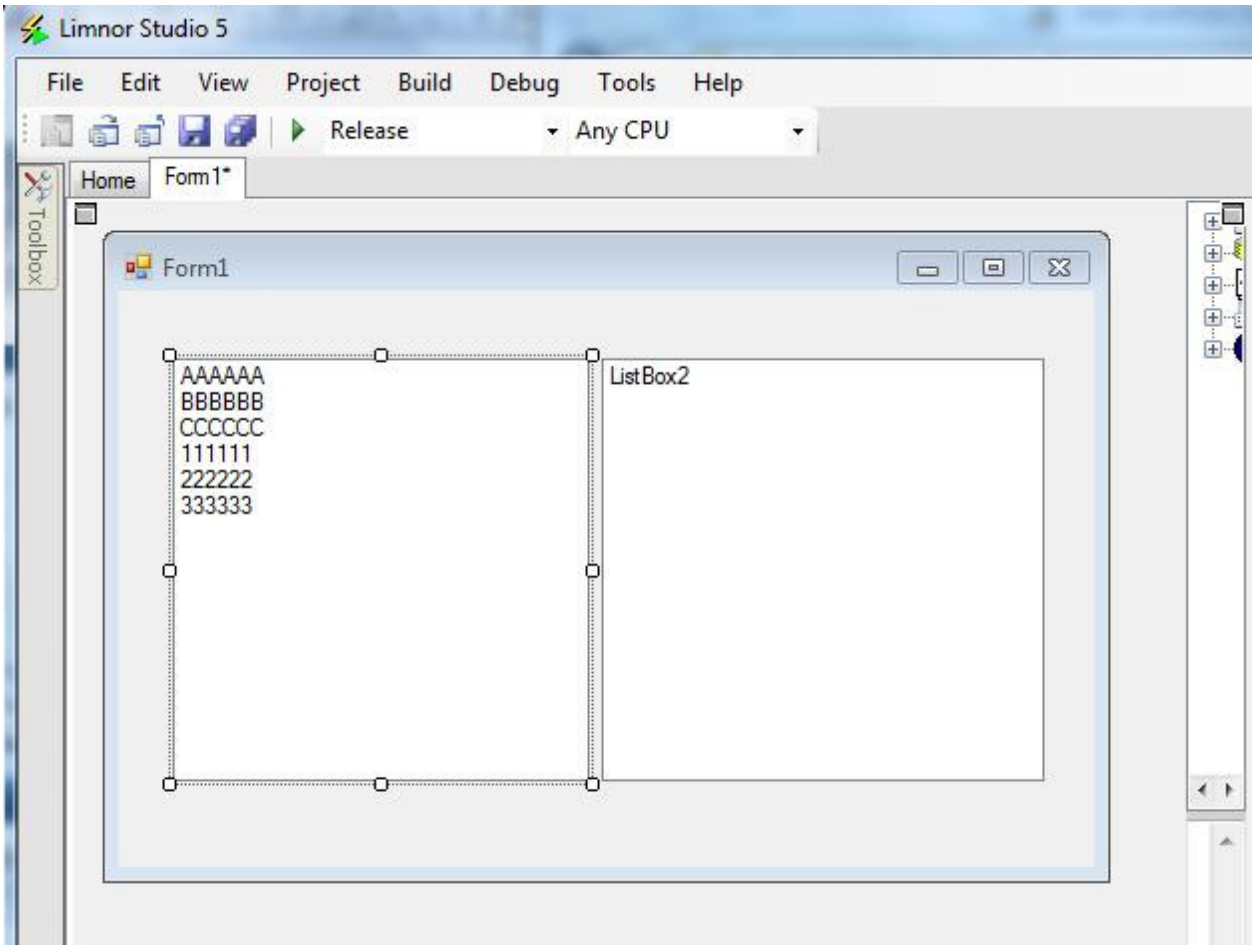
Simple Drag Drop Programming

Contents

Sample Scenario

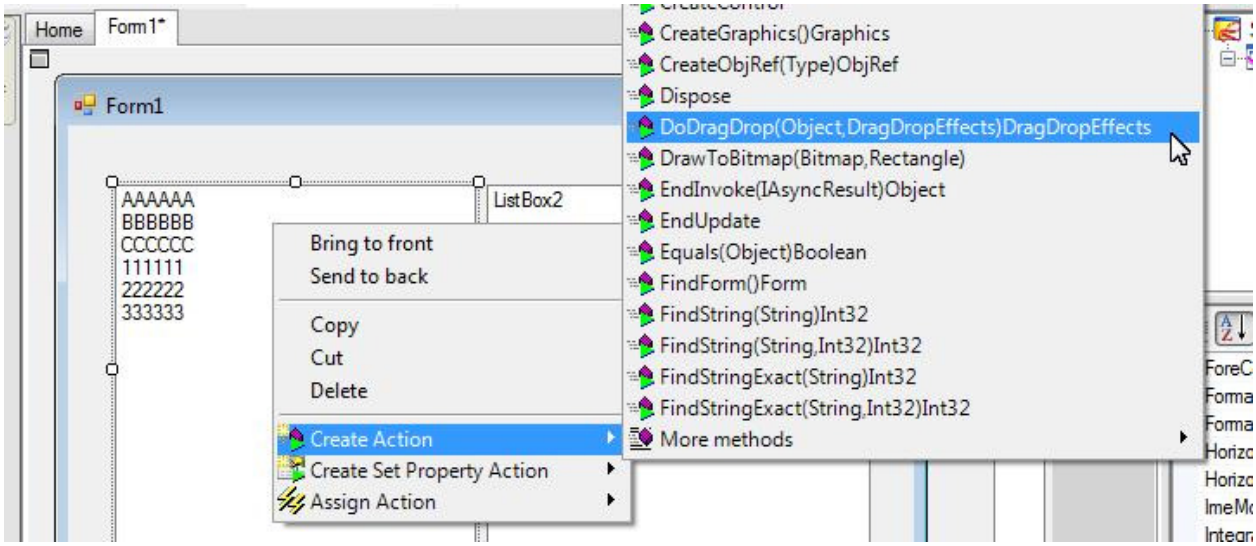
This sample uses two list boxes. The list box items are text. The user may drag selected list item from the first list box and drop it to the second list box.



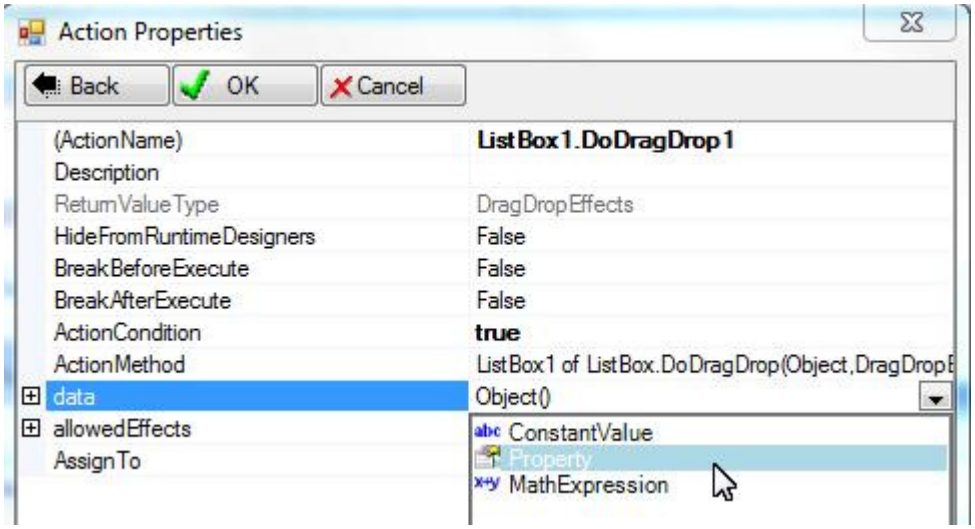
Setup Drag Source

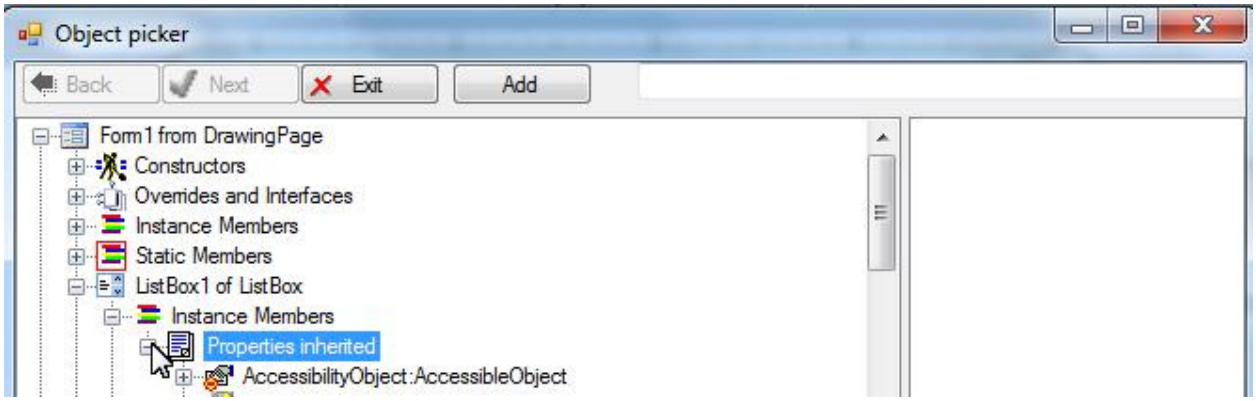
Create DoDragDrop action

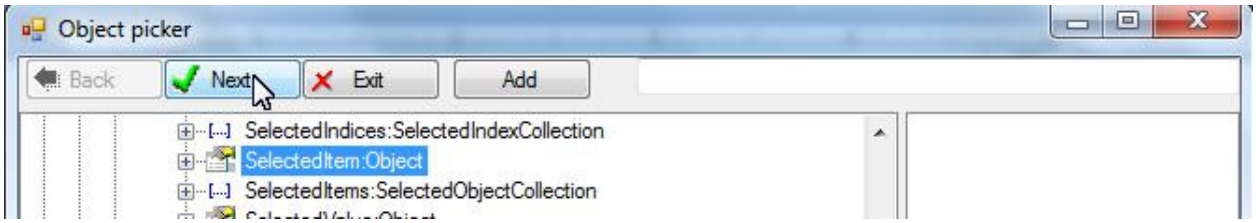
A DoDragDrop action starts a drag-drop operation.



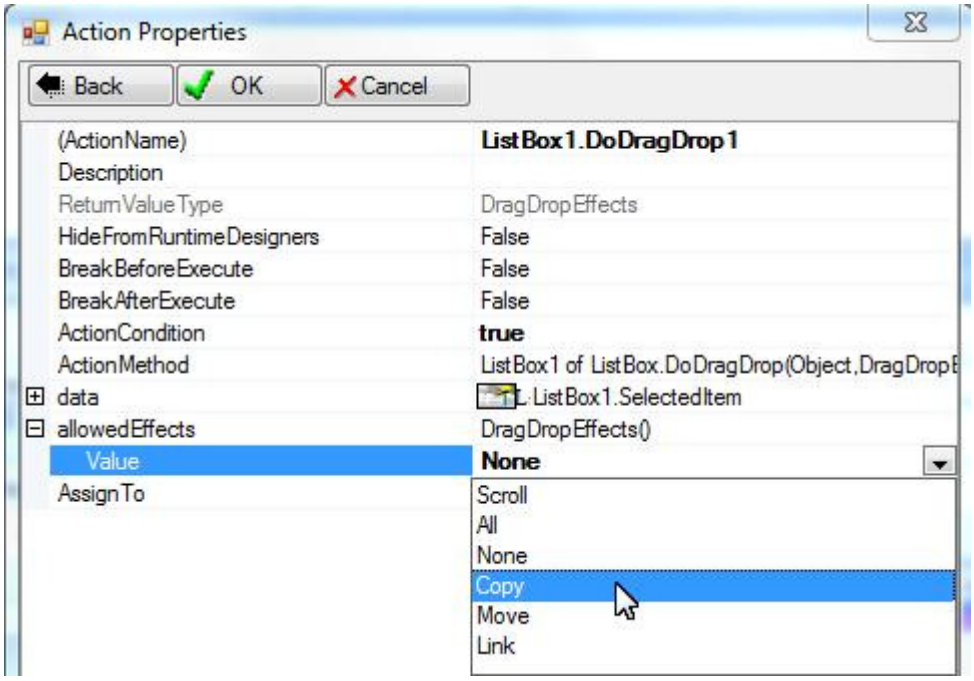
The “data” parameter of the action holds the data to be transferred by drag-drop. In this sample, we want to drag selected item of the list box. Select “Property” to get selected item of the list box:



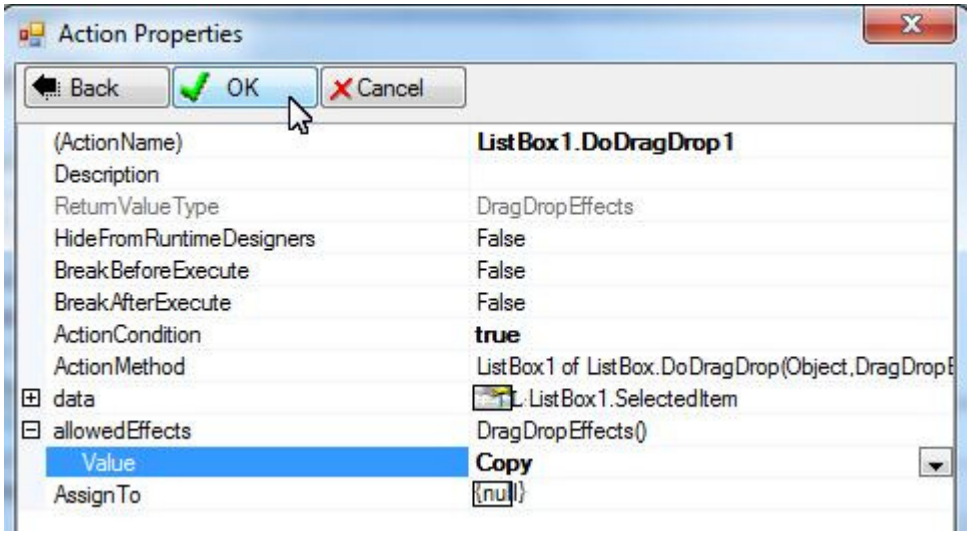




Set “allowedEffects” to Copy:

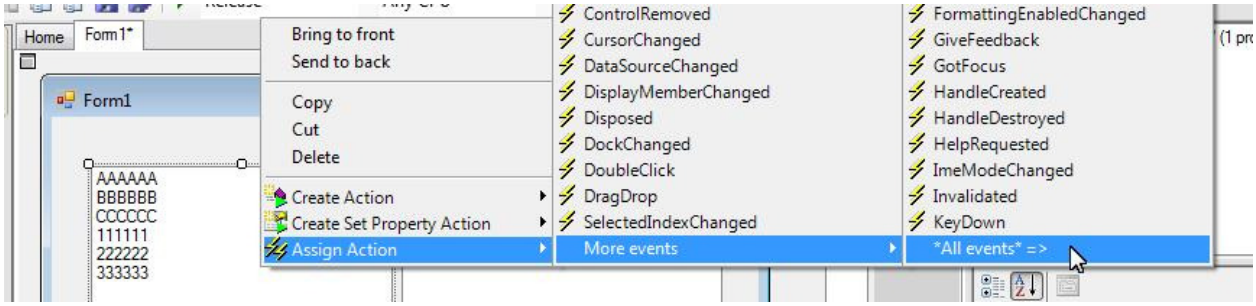


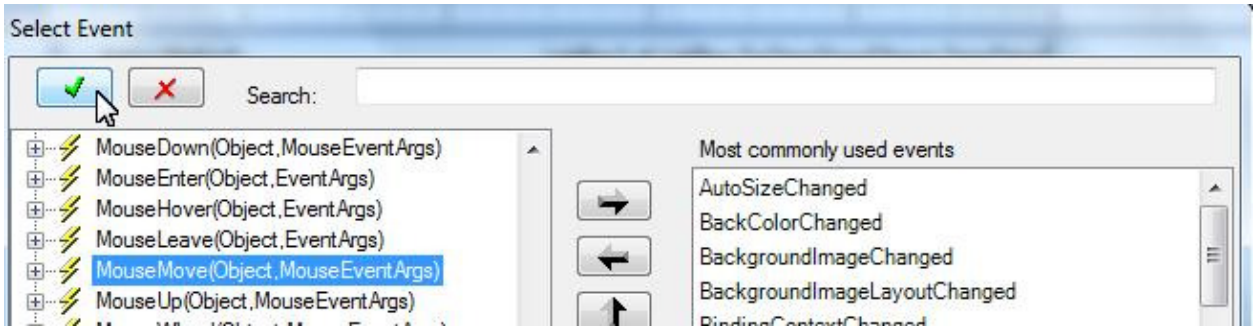
Click OK to finish creating this action:



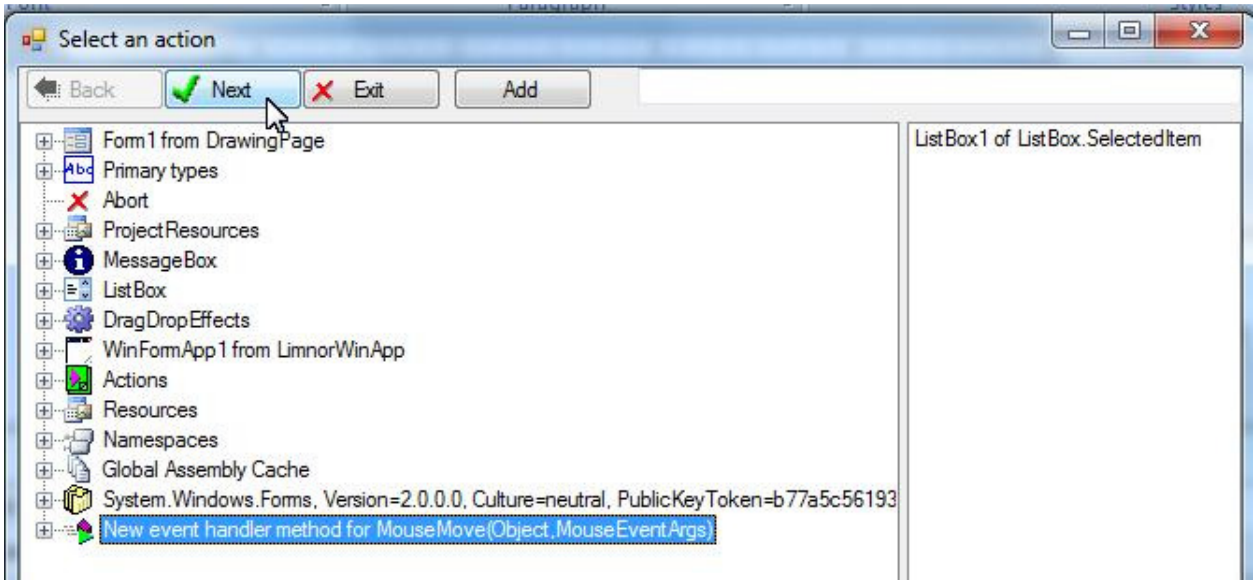
Start dragging

We want to execute the above action when mouse is moving over the first list box and the left mouse button is pressed. We do it by handling MouseMove event.

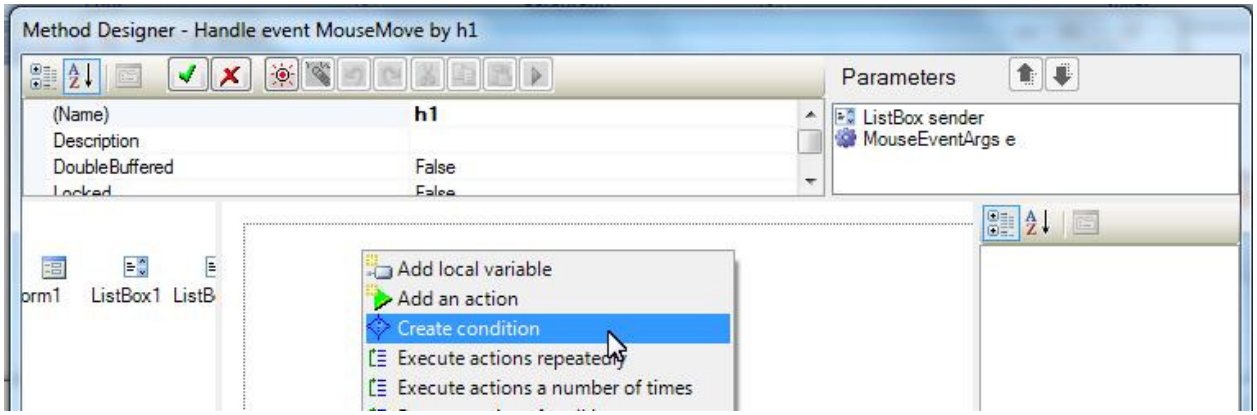




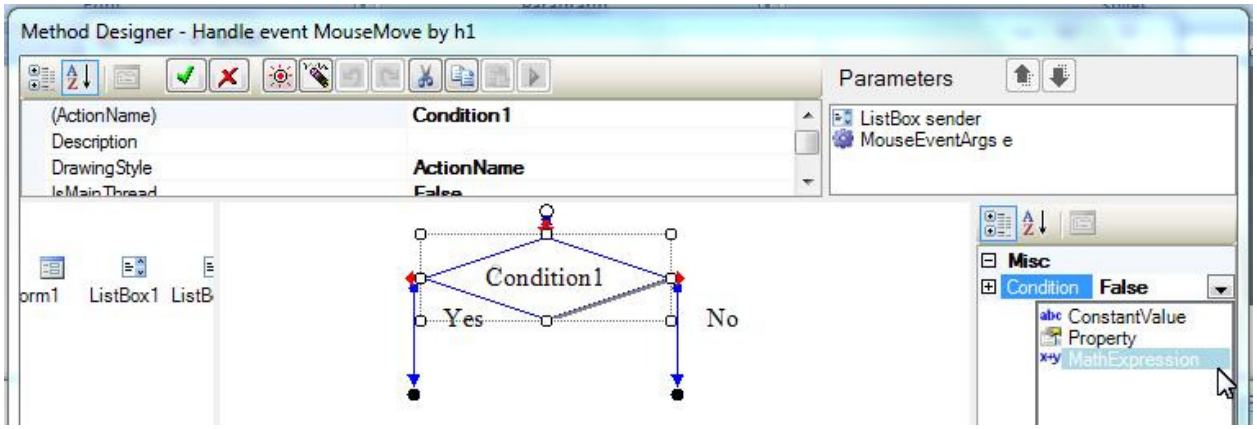
Select “New event handler method for MouseMove” because we need to detect whether the left mouse button is pressed via event parameters.

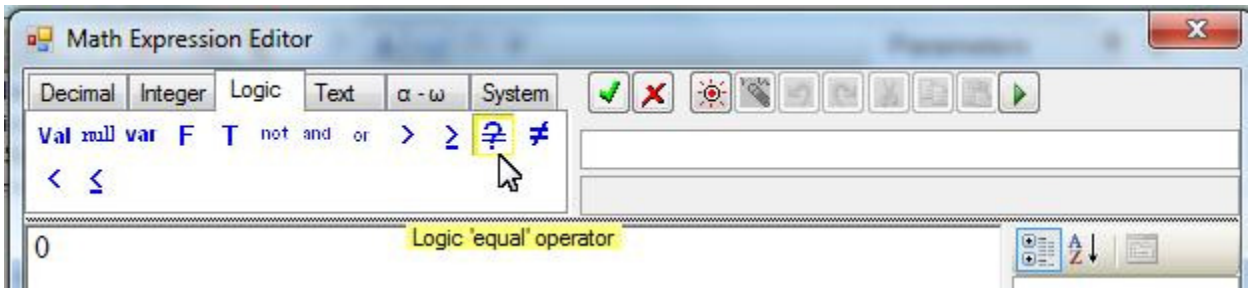


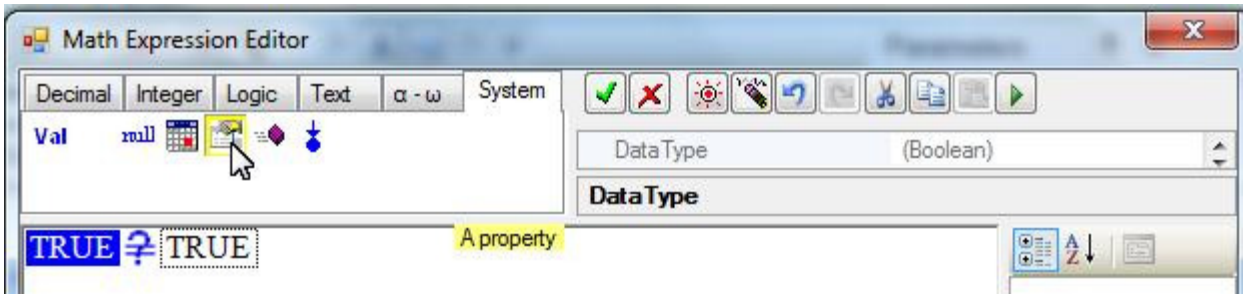
Add a Condition action for detecting mouse button:

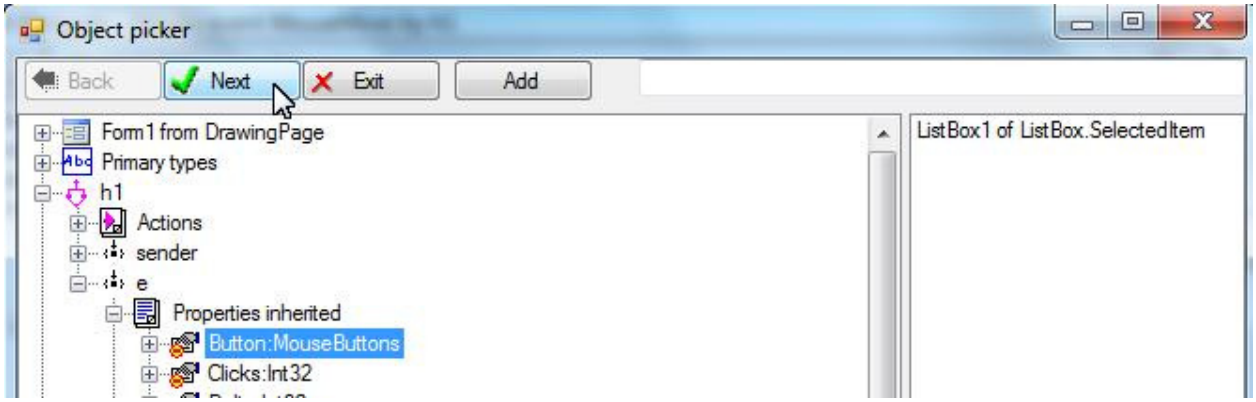


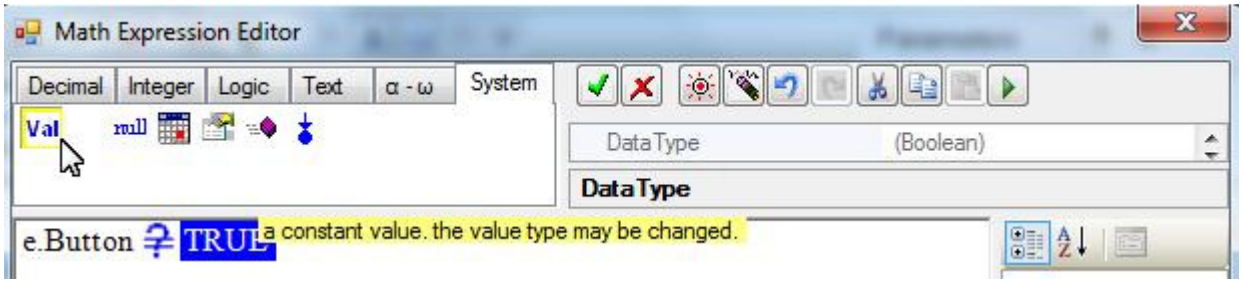
Set the Condition to detect mouse button:

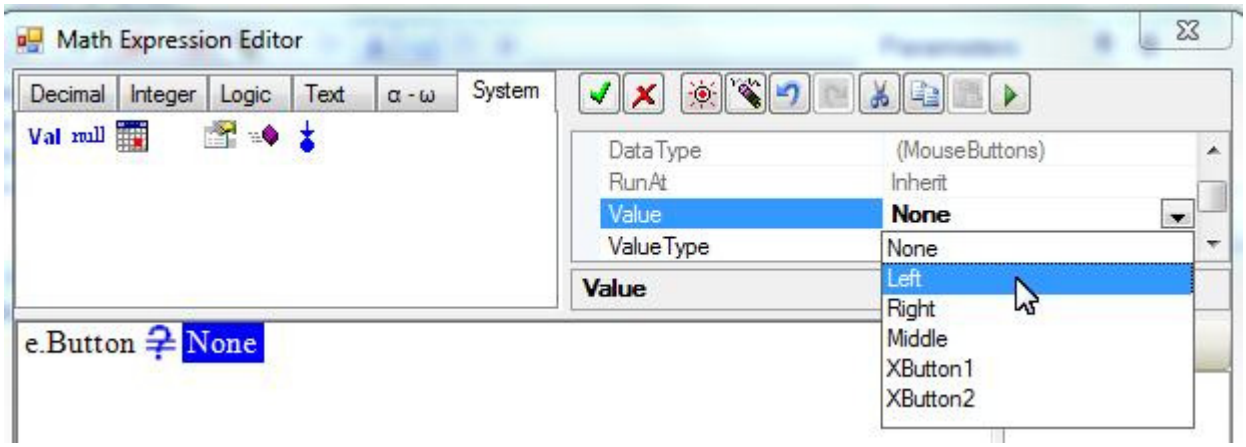




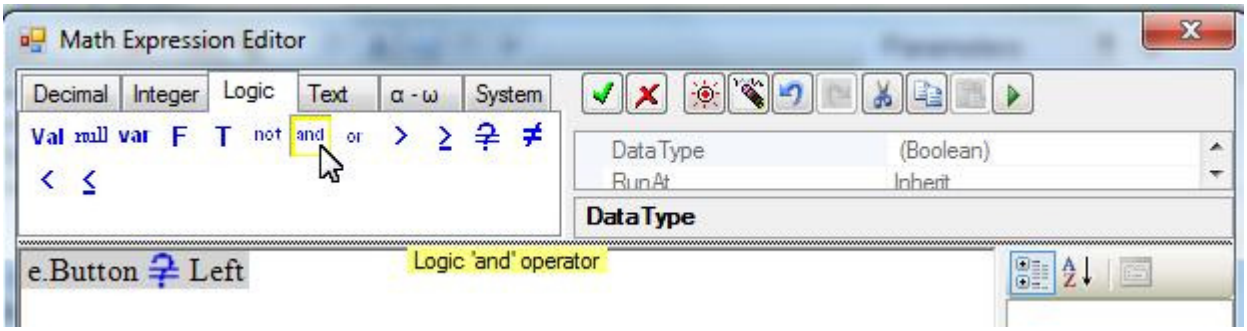




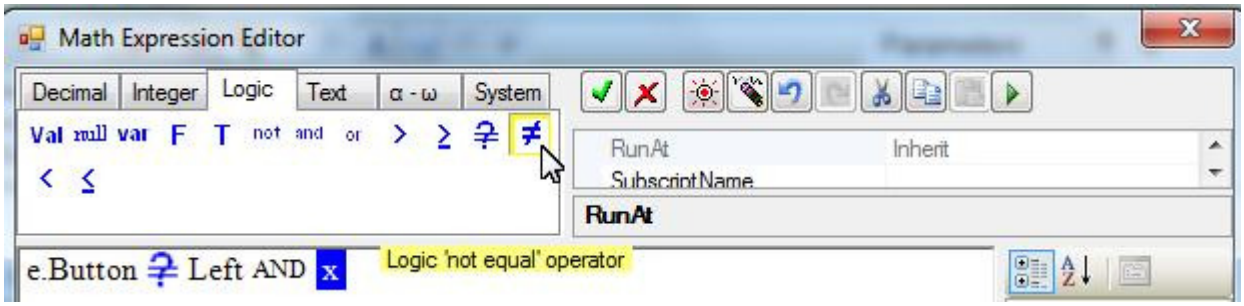




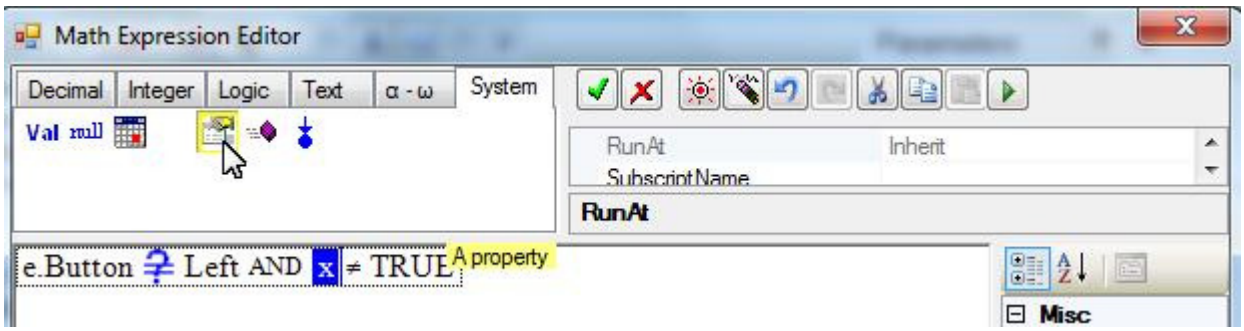
This expression checks whether the left mouse button is pressed. We may also add a checking to see whether an item is selected in the first list box. Click ∓ and click “and”:



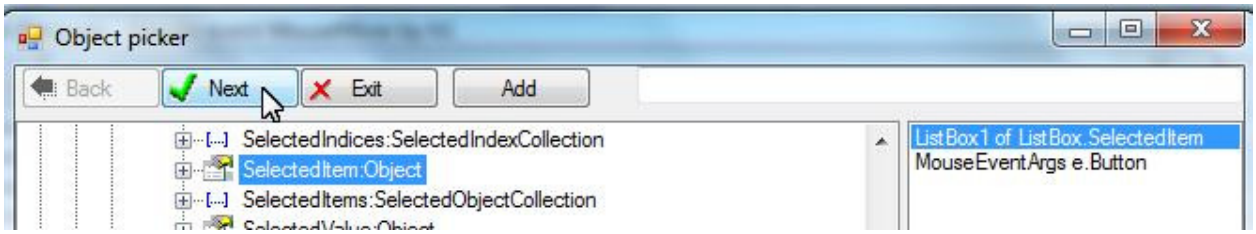
Click “x” and click ≠:



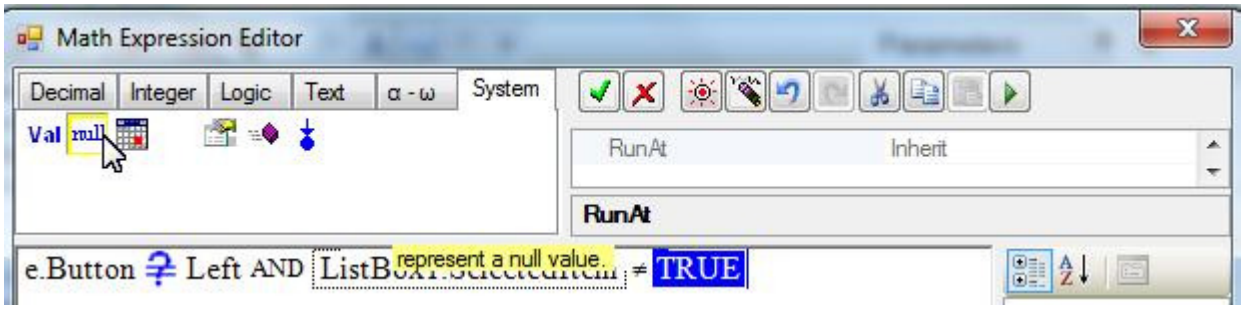
Click “x” and click :



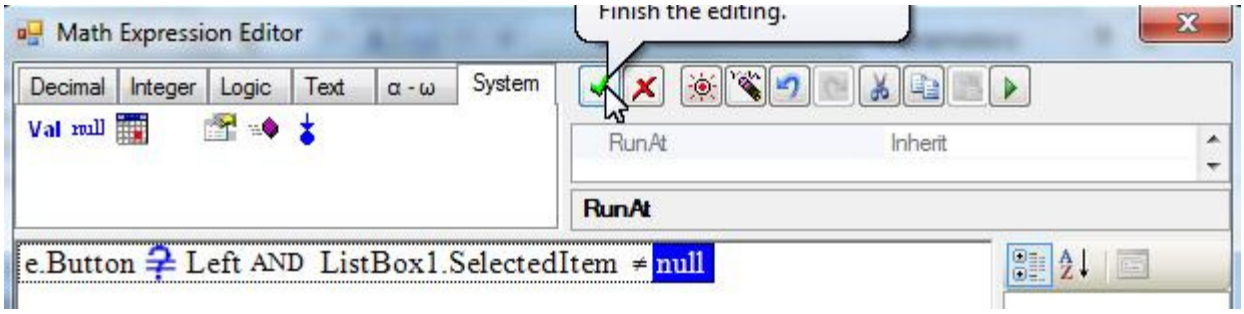
Select SelectedItem from the first list box:



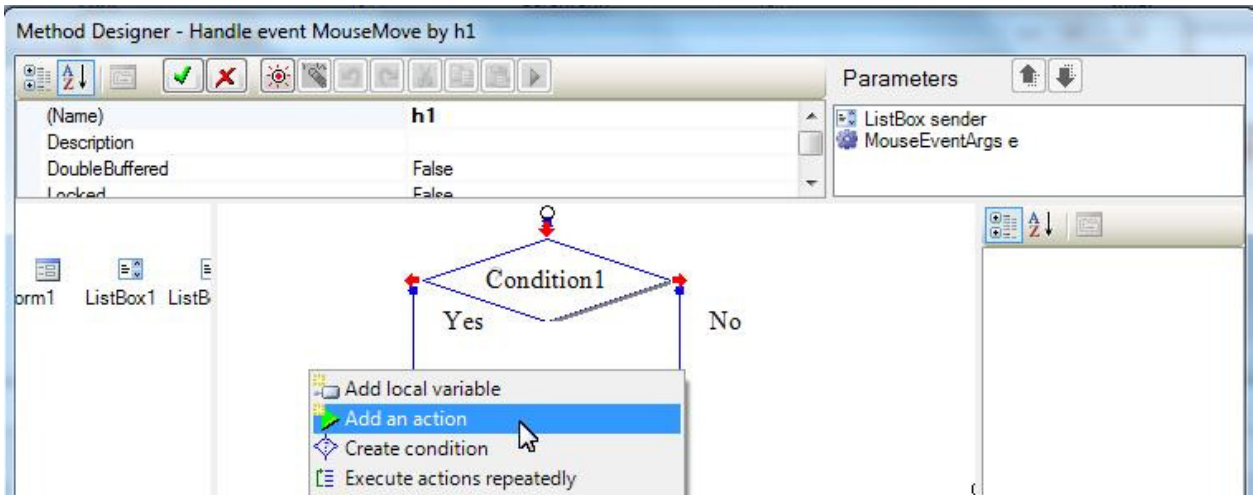
Click “TRUE” and click “null”:

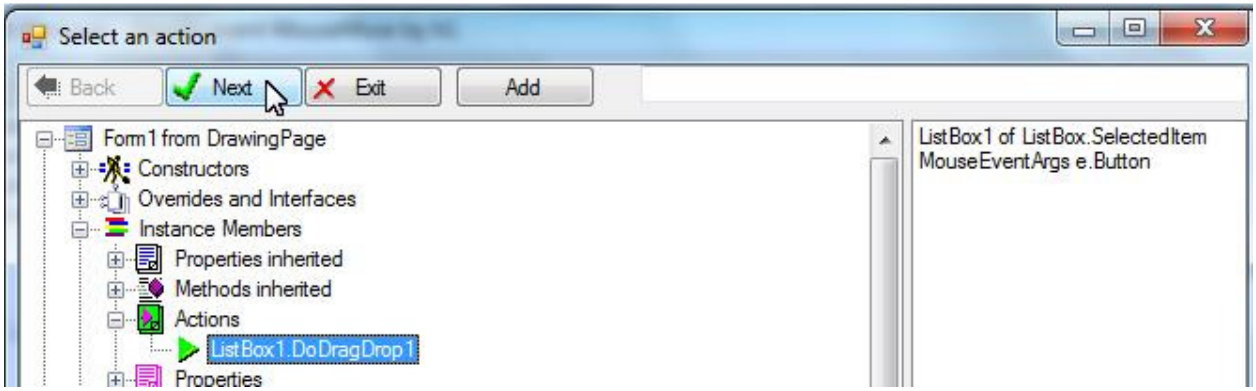


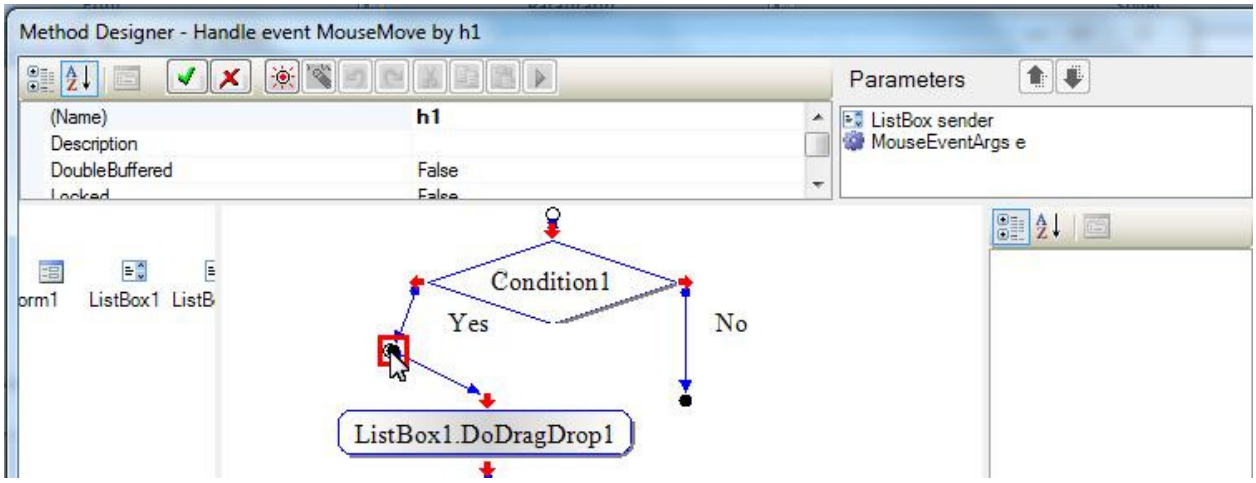
This expression is the condition a drag may start:



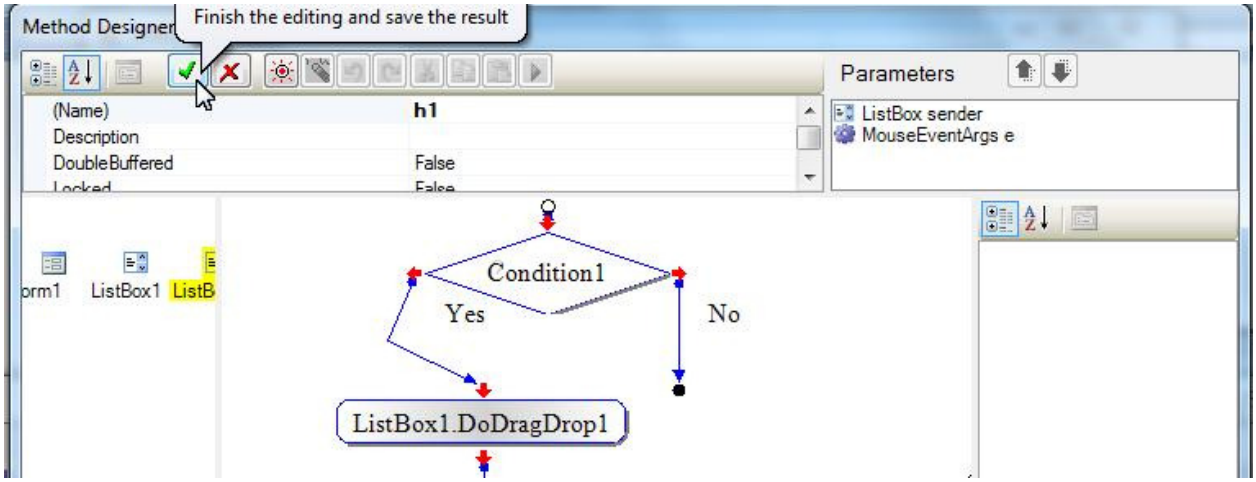
Link DoDragDrop action to the “Yes” port of the condition:







We are done setting up drag source.

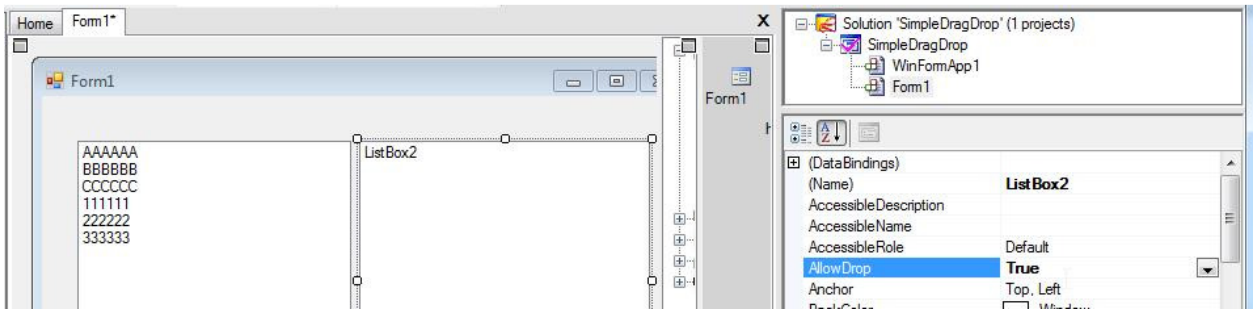


Setup Drag Target

We use the second list box as the drag-drop target.

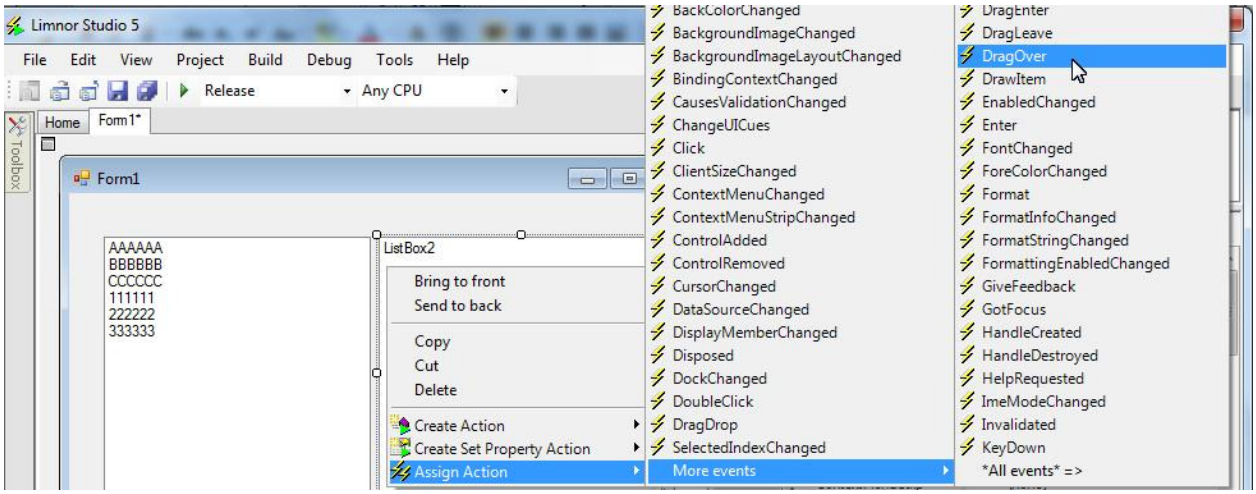
Allow drop

Set AllowDrop of the second list box to true:

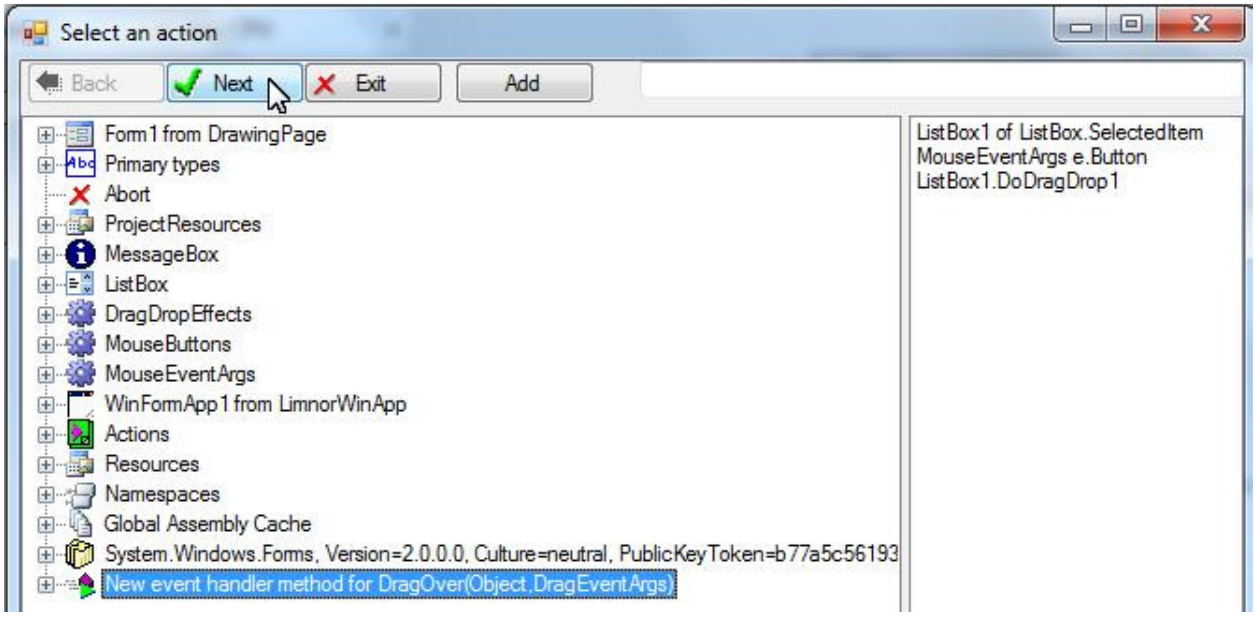


Handle DragOver event

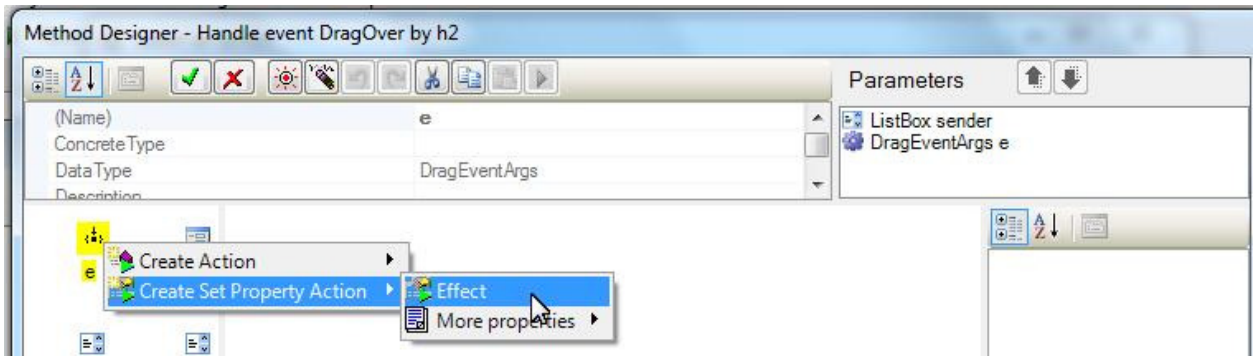
At this event we need to notify the system that dropping is allowed:

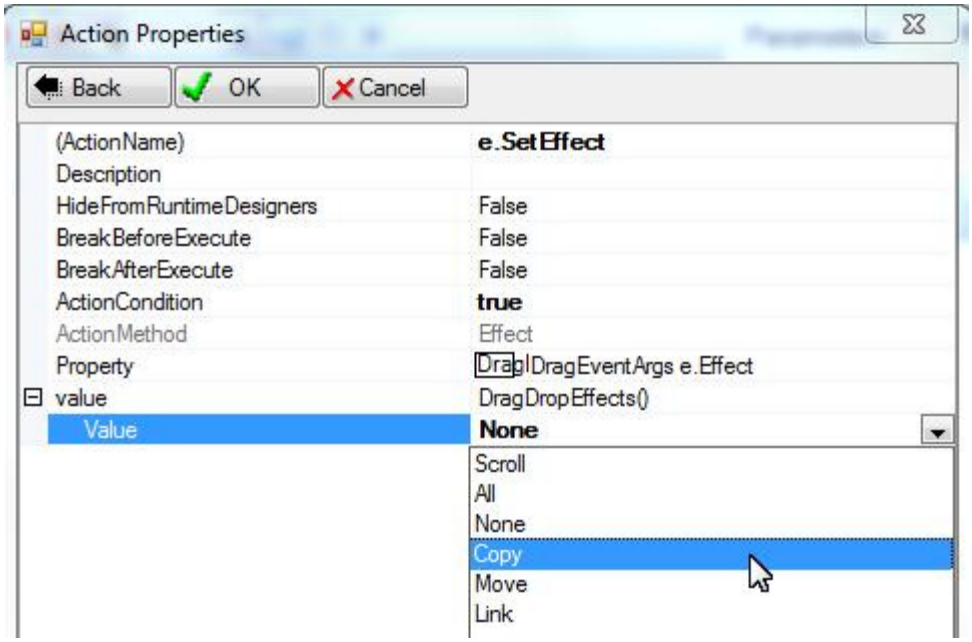


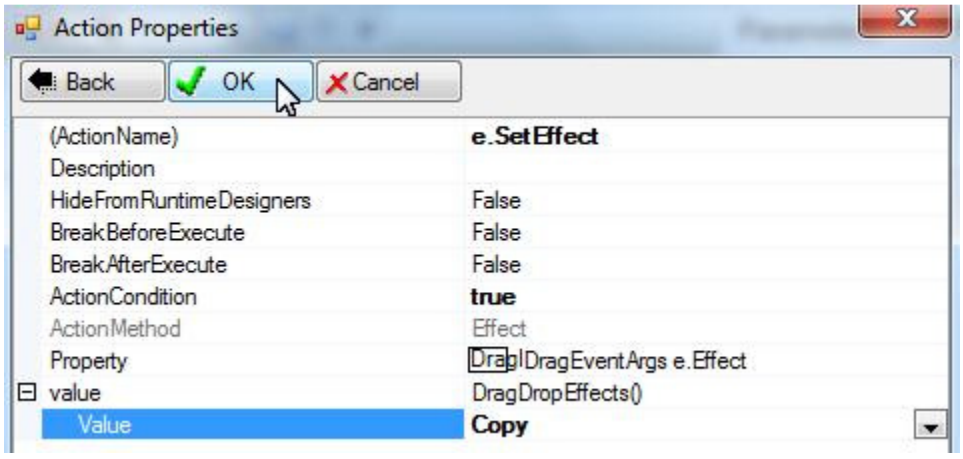
Select “New event handler method for DragOver” because we need to access the event parameters:



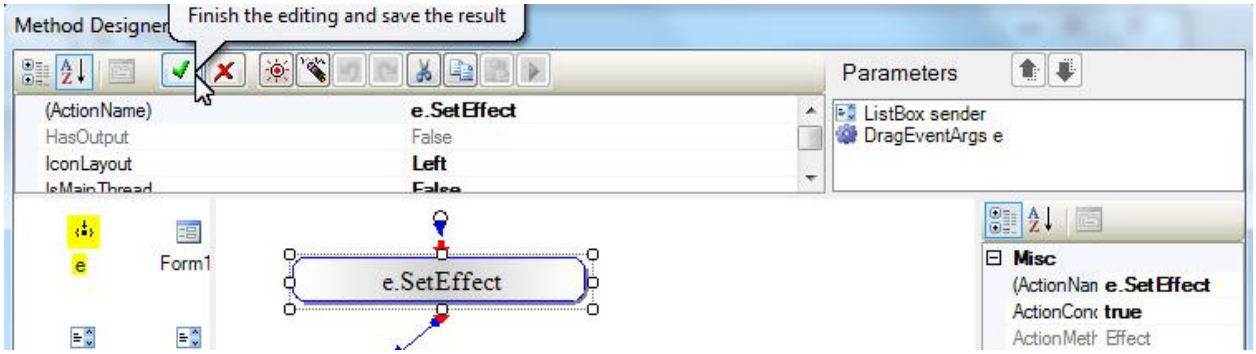
Set “Effect” of the event parameter e to “Copy”:





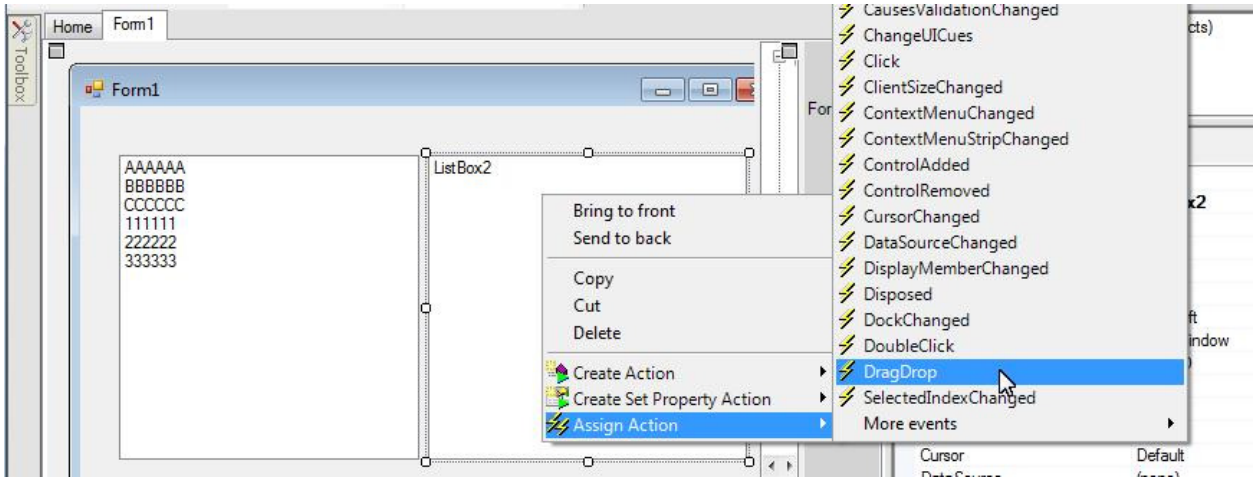


This action is all what we need for handling this event.

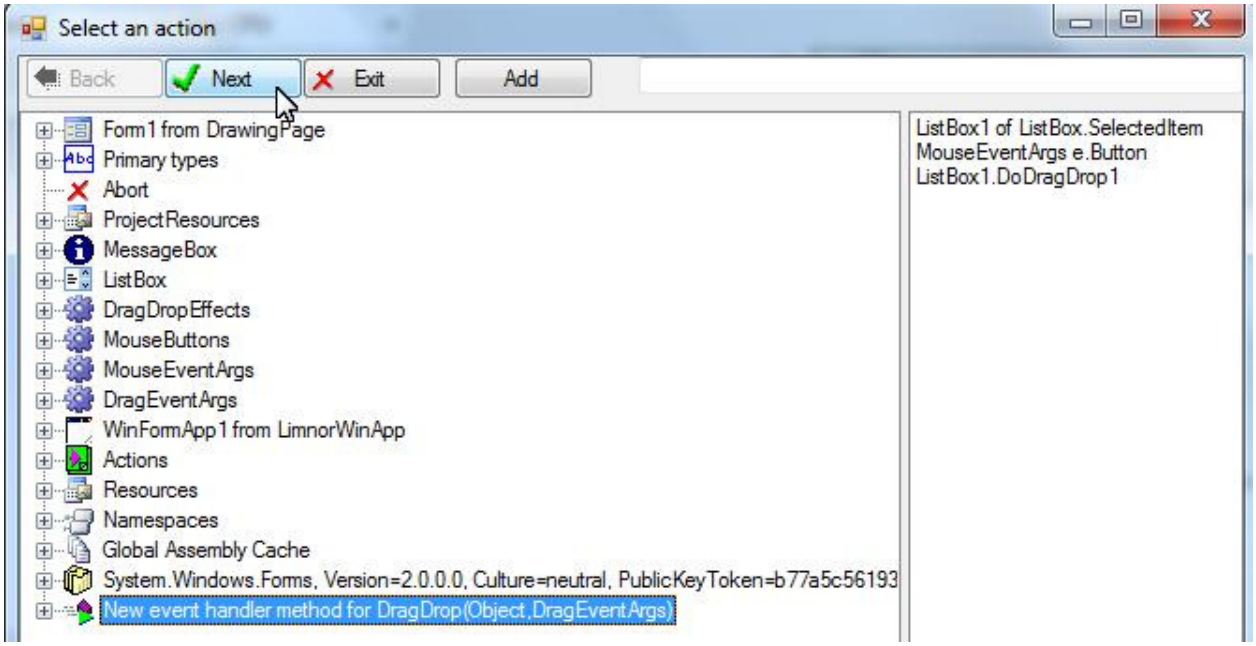


Handling DragDrop event

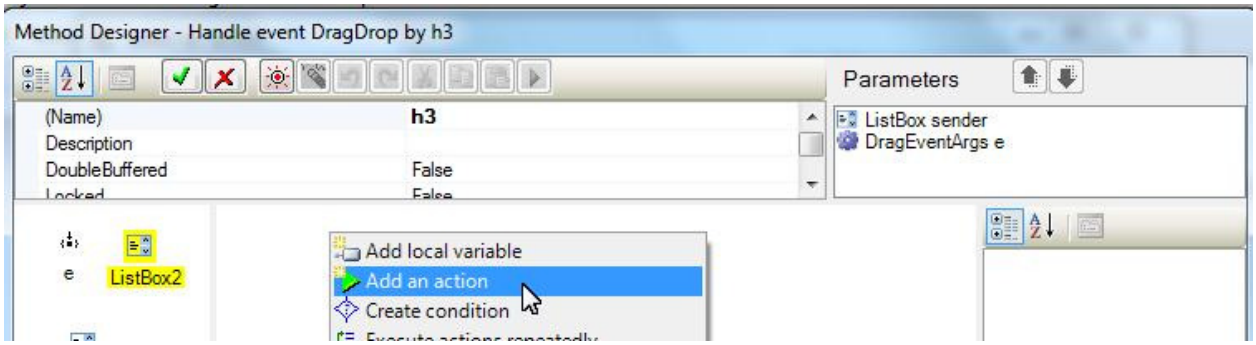
This is the event when drop occurs. In this sample, we want to add the dropped data to the second list box.



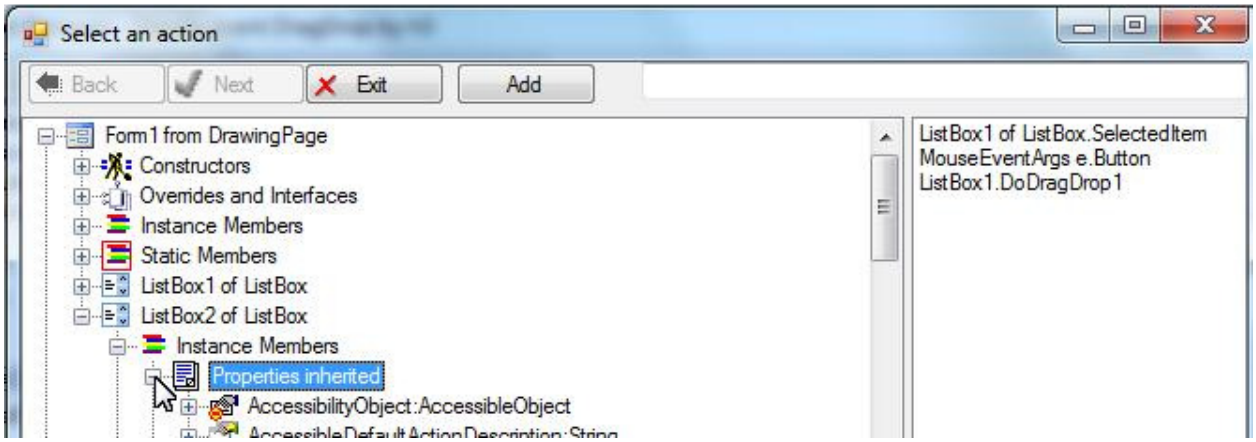
Select “New event handler method for DragDrop” because the dropped data are contained in the event parameters:

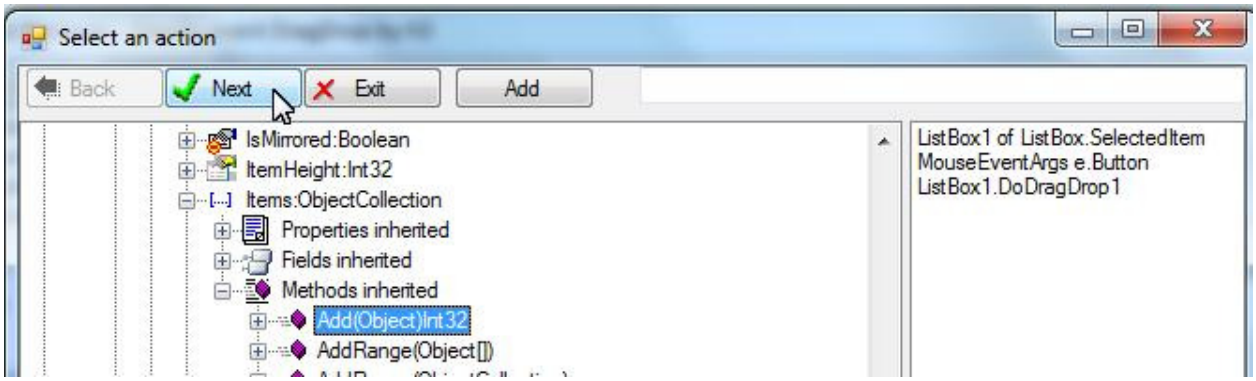


Add an action to add an item to the second list box:

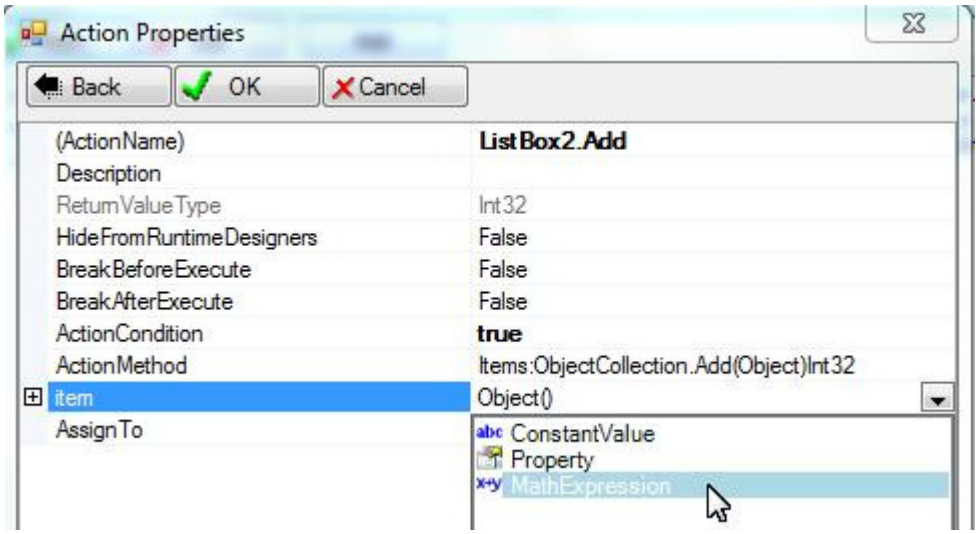


Find the “Items” property of the second list box and select its Add method:

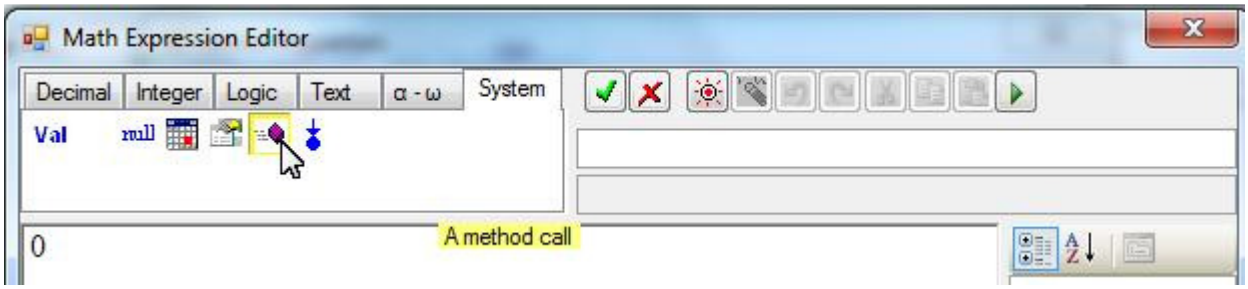




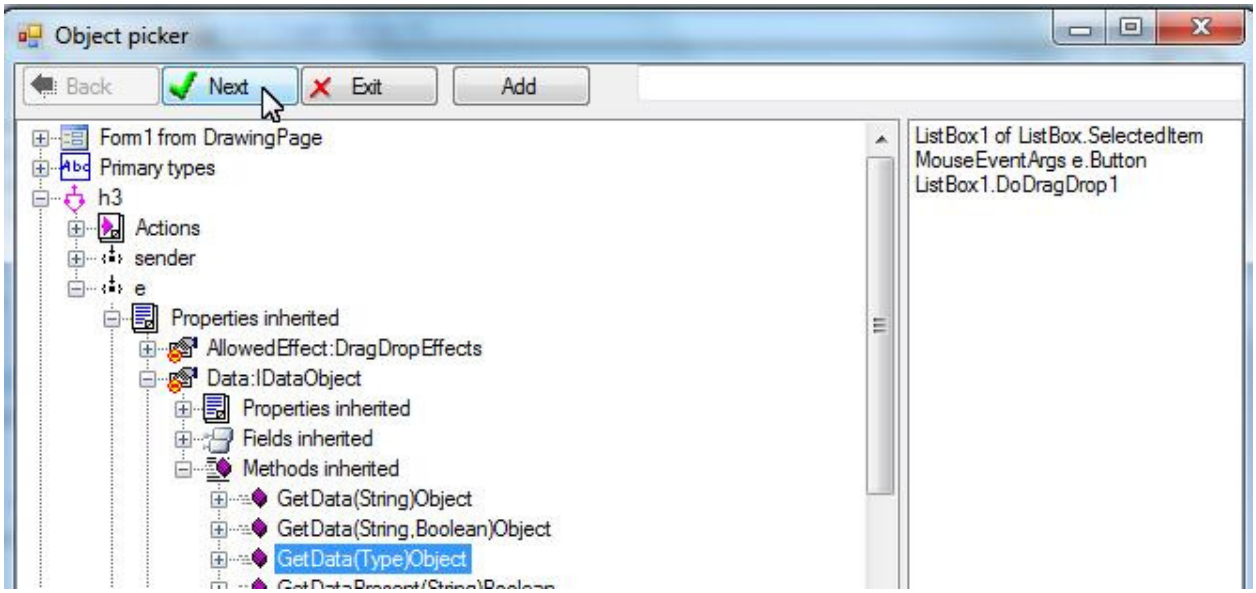
Select “Math Expression” to get dropped data:



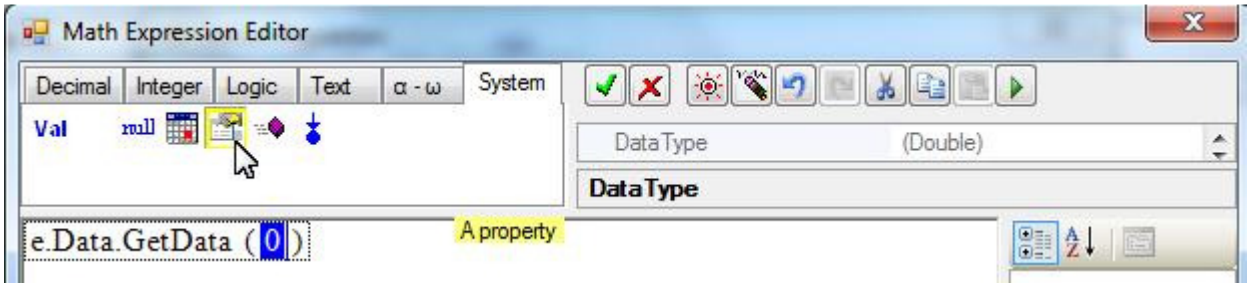
Select method icon  to get dropped data:



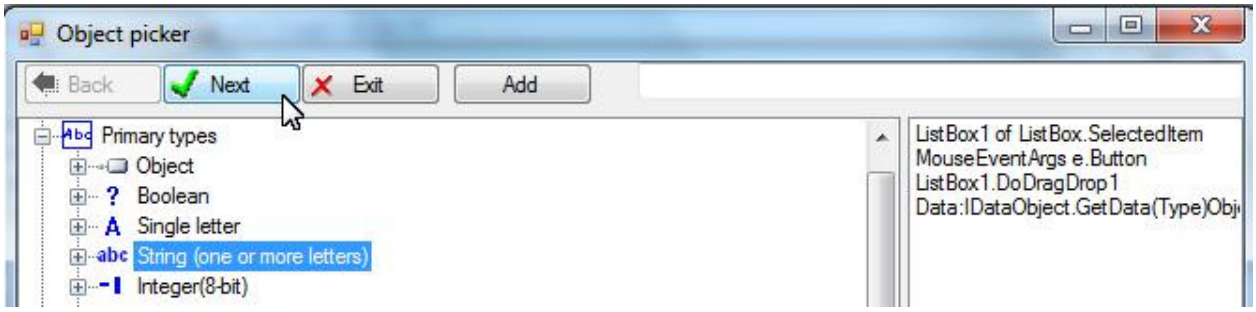
Select GetData(Type) to get dropped data by type because we know that the dropped data is a string:



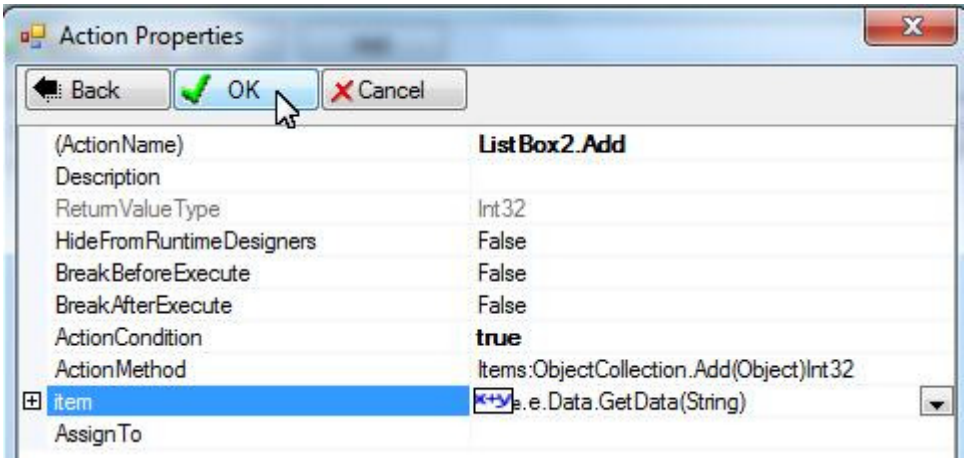
Use  to specify the string type:



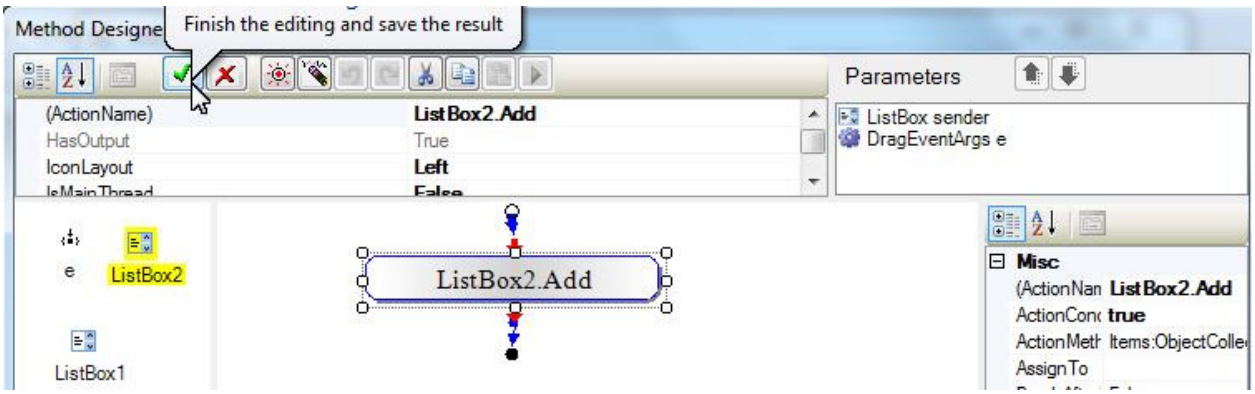
Select “String”:



This action adds the dropped data to the second list box:

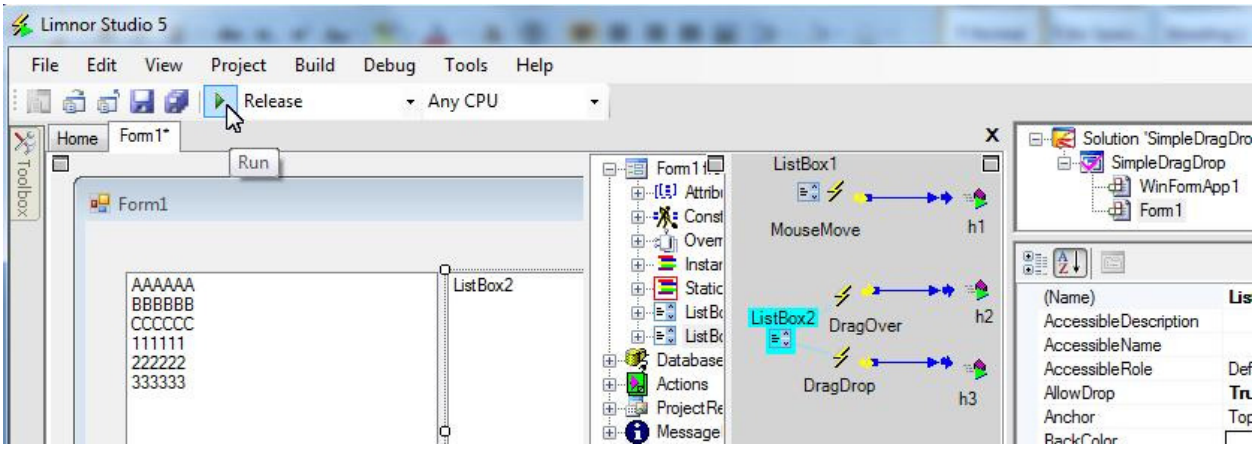


For this sample, this action is all what we need for handling dropping.

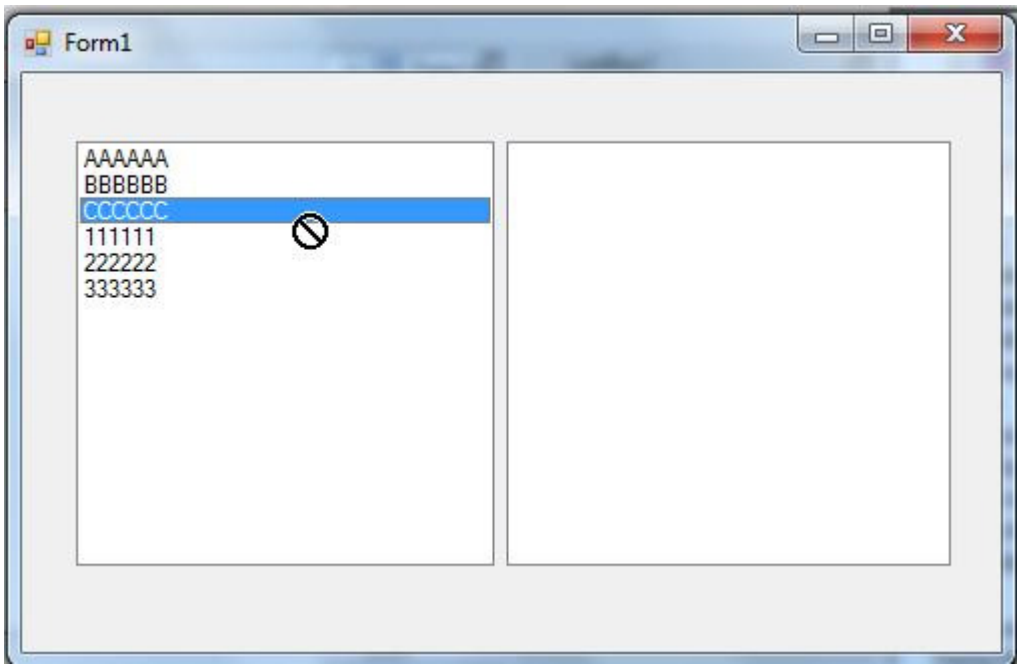


Test

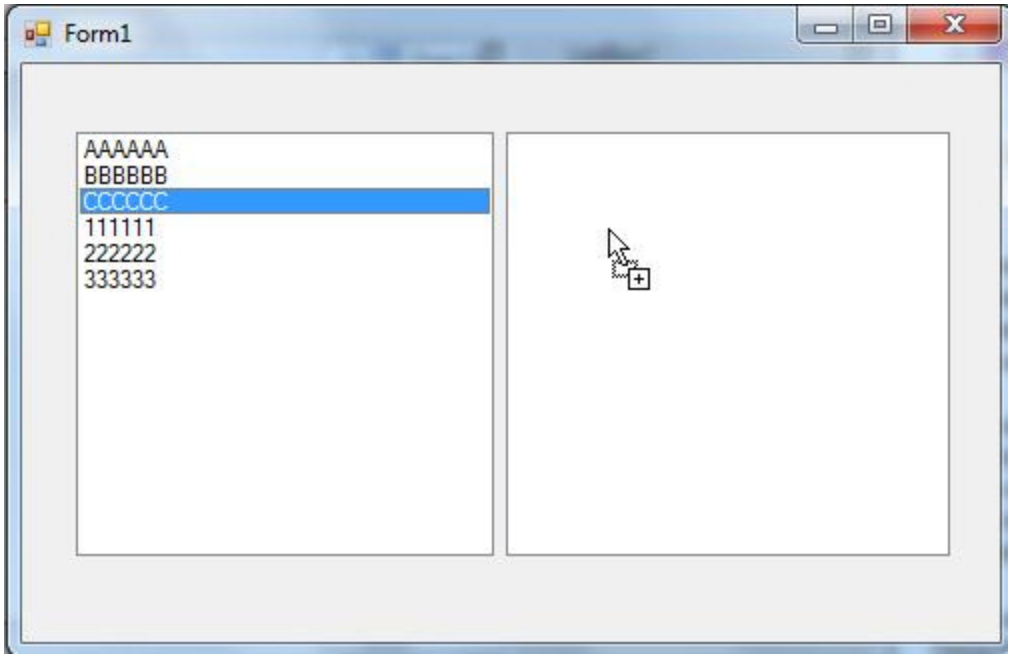
Click the Run button.



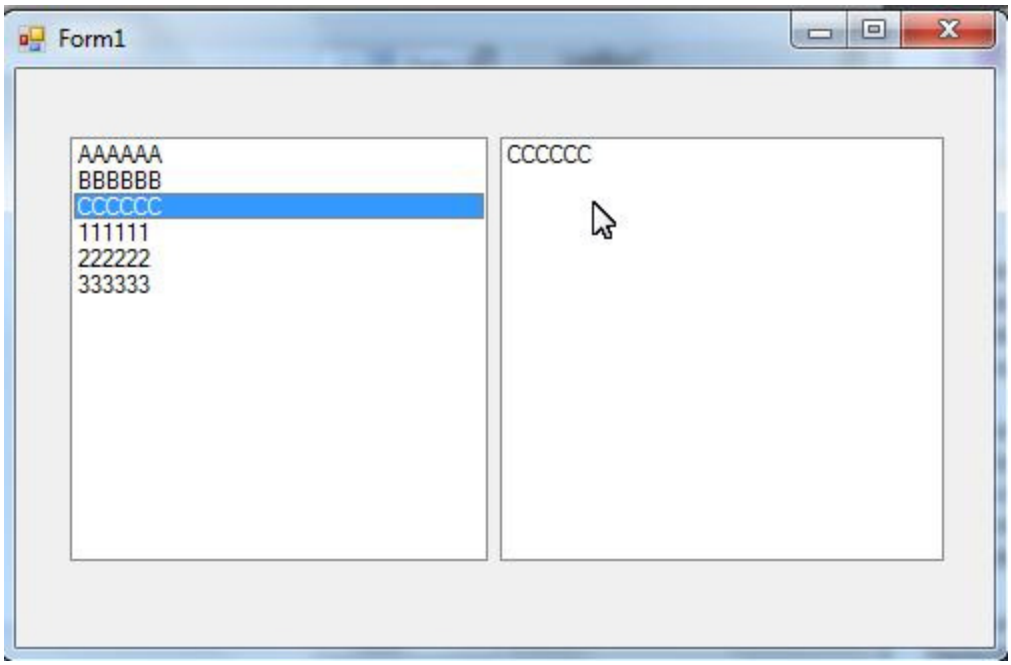
The form appears. Drag the first list box.



When dragging it over the second list box, the dragging icon indicates that it can be dropped there:



Drop it there. The selected item appears in the second list box.



This sample shows dragging text data. You may also drag other types of data if the data can be serialized.