


Lesson 17. Programming Drag and Drop

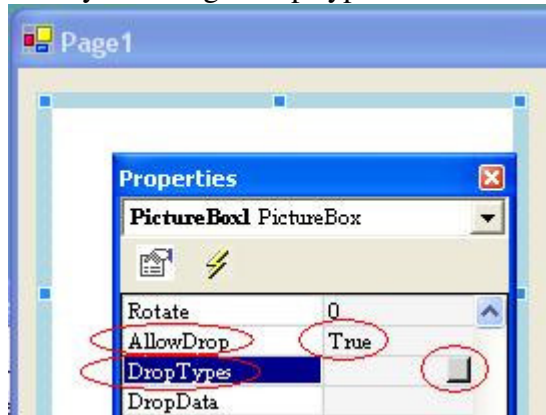
In this lesson, we show how to let performers to accept dropped data and how to let the user to drag files and performers. We use several examples to show you how to do drag and drop programming.

17.1 Drop a file

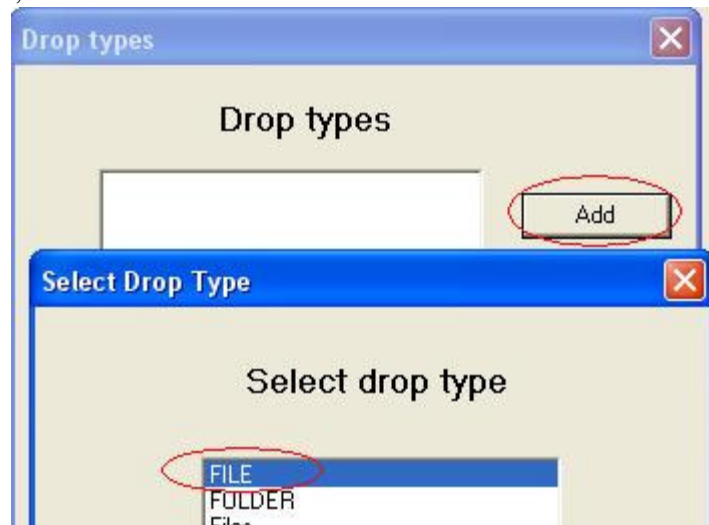
Suppose we want our application to have such a feature: there is a picture box on a page; the user may drag an image from Windows File Explorer and drop to the picture box, the picture box will display that image.

Step 1. Determine the drop acceptor and specify accepted data type.

In this example, the picture box will be the drop acceptor. Any performer with a property named “AllowDrop” can be a drop acceptor. You need to set this property to “True” so that it will accept drop data. You need also to specify which data types are allowed to drop to it. This can be done by selecting “DropTypes” and clicking :



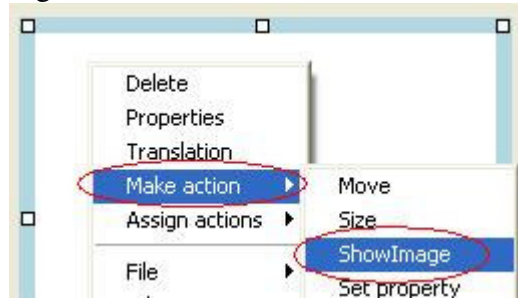
Click Add button, and select “FILE”:



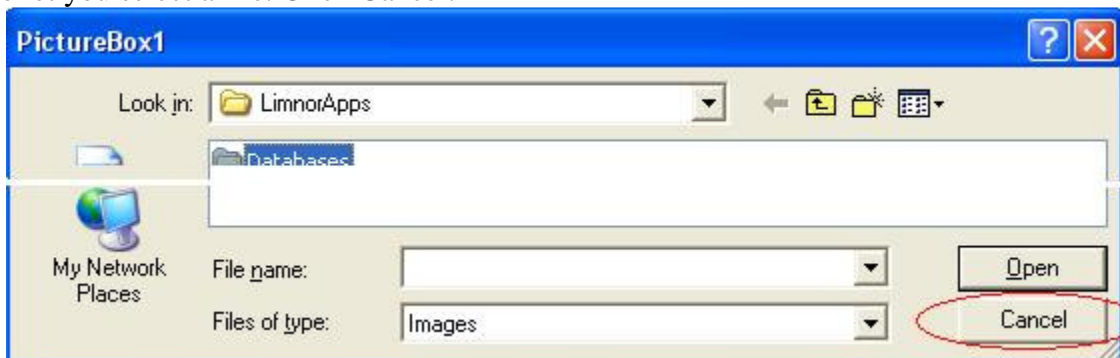
Click OK button to finish it. You may add many accepted data types. For this example, we just accept “FILE”.

Step 2. Make an action to use the dropped data

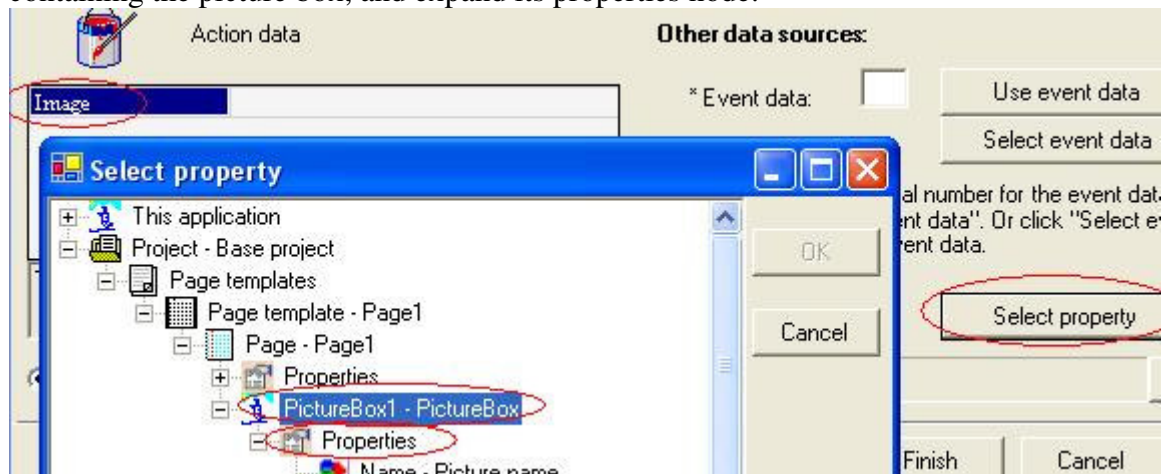
Now this picture box may accept files dropped to it. We need to make an action to display the image file dropped to it. Right-click on the picture box, choose “Make action”, choose “ShowImage”:



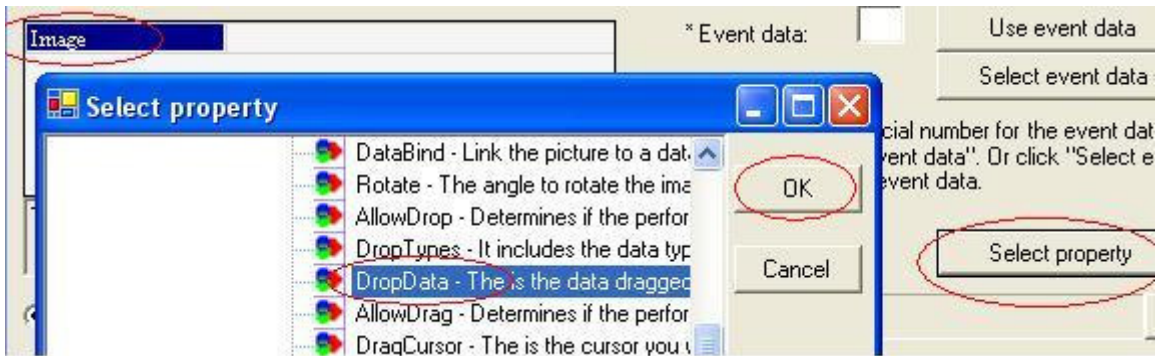
Give an action name, say, ShowDroppedImageFile. It displays file selection dialogue box to let you select a file. Click Cancel:



The Action Data dialogue box appears. Click “Select property” button. Find the page containing the picture box, and expand its properties node:



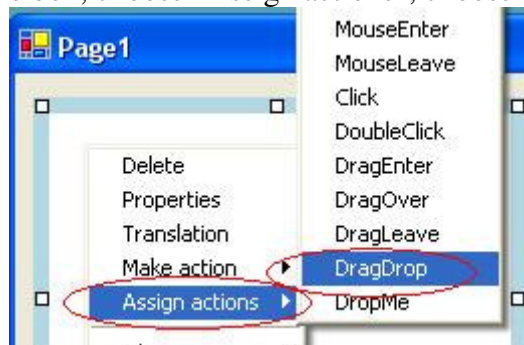
Scroll down and find and select “DropData” property:



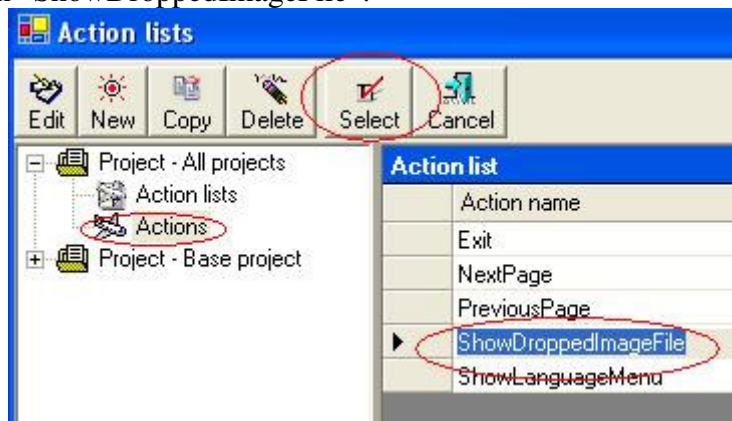
It will ask you the data type for this action. For this example, there is just one type, “FILE”. Select it and click OK.

Step 3. Assign the action to DragDrop event

We want the above action to be executed every time a file is dropped to the picture box. Right-click on the picture box, choose “Assign actions”, choose “DragDrop”:



Select the action “ShowDroppedImageFile”:

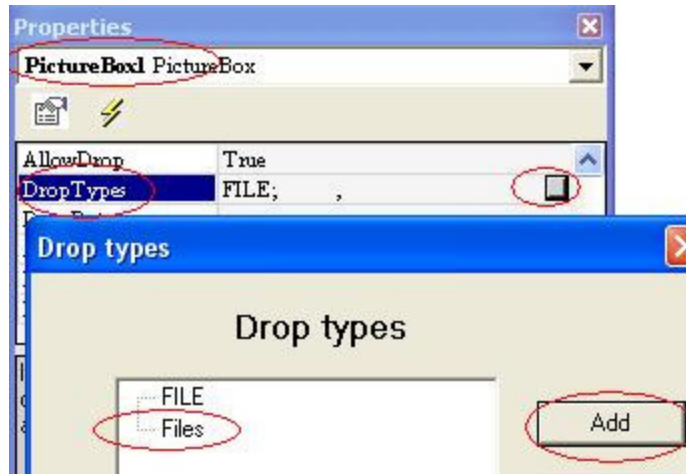


We are done. You may run it and drop image files and drop to the picture box to try it.

17.2 Drop files

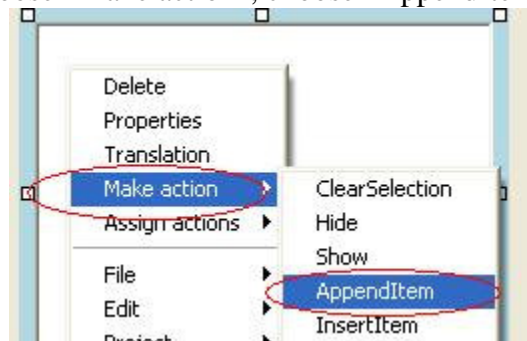
The above example shows how to accept one single dropped file. What if you want to accept more than one file? Still using the above example, but we use a list box to display all the file names dropped to the picture box. You may further process all the files in the list box (this example does not do other file processing).

Step 1. Add “Files” as the accepted drop type:

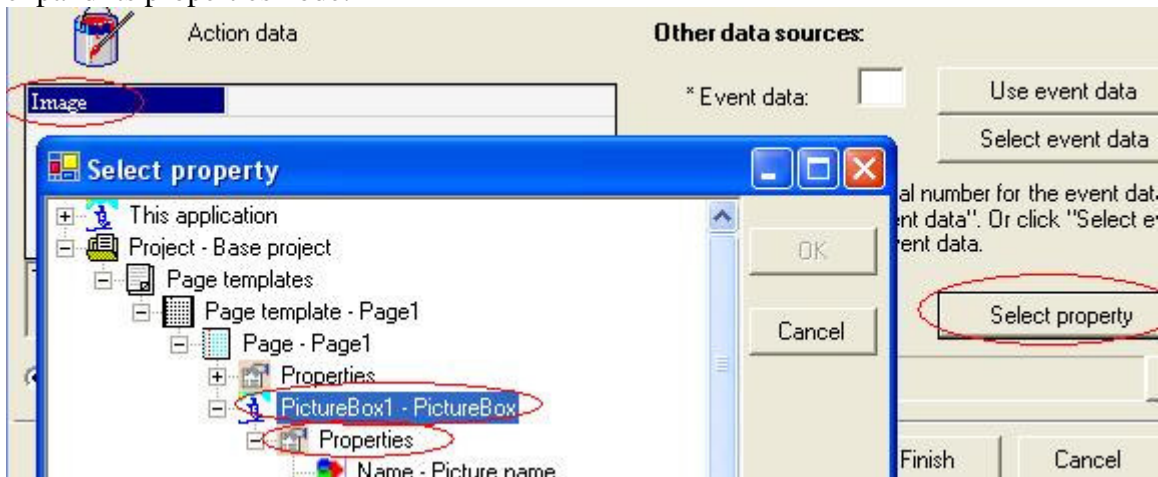


Step 2. Create an action to accept dropped data.

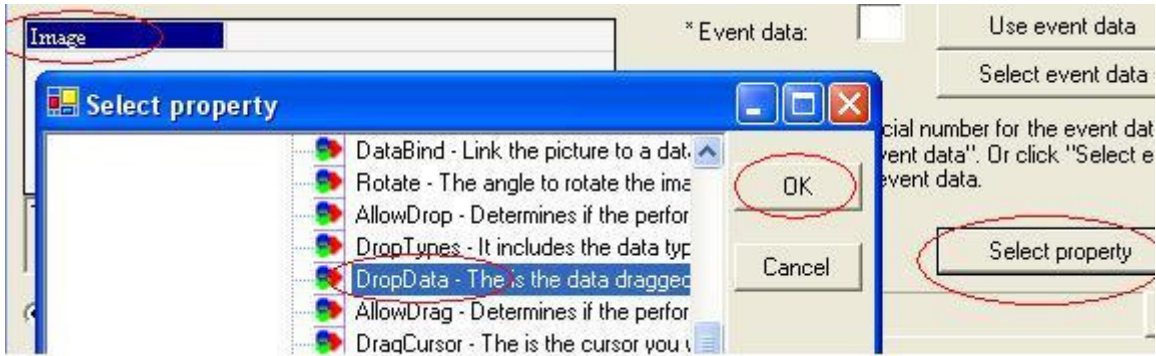
We want to display dropped file names in a list box. Create a list box on the page. Right-click on the list box, choose “Make action”, choose “AppendItem”:



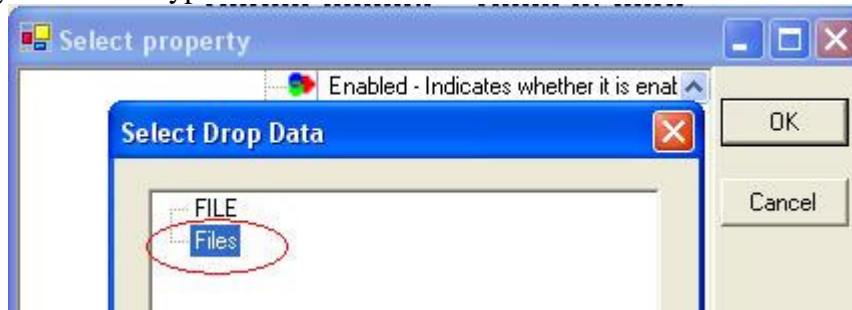
Give an action name, say, ShowDroppedFileNames. The Action Data dialogue box appears. Click “Select property” button. Find the page containing the picture box, and expand its properties node:



Scroll down and find and select “DropData” property:

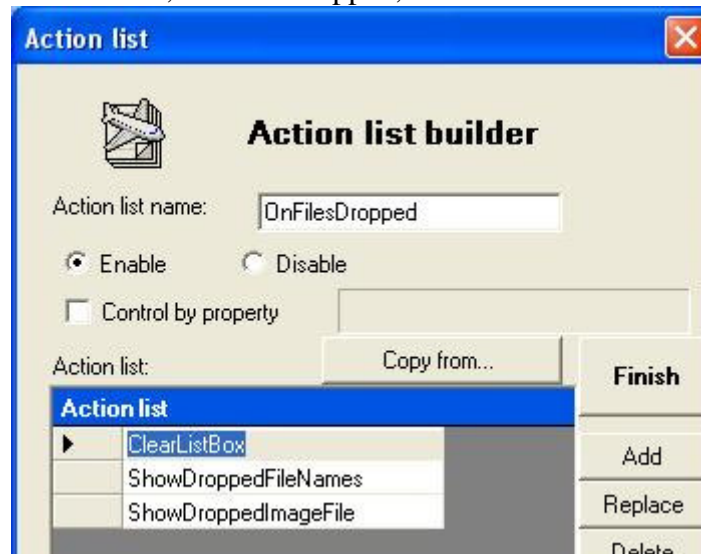


It will ask you the data type for this action. Select “Files”:



Step 3. Assign the action to DragDrop event

We need to assign the above action to the DragDrop event. Because we still want to let the picture box display the image when files are dropped to it, we want both actions, ShowDroppedImageFile and ShowDroppedFileNames, to be assigned to this event. We need to create an action list to include these two actions and assign the action list to the event. Below is the action list, OnFilesDropped, we created:



In the action list, there is a new action, ClearListBox. This is an action created by the list box’s Clear method. It is used to remove all items in the list box. This way, the list box will just show the files newly dropped.

Now, assign this action list, OnFilesDropped, to the picture box’s DragDrop event. We are done.

17.3. Drag and drop performers

You may allow the user to drag one performer and drop it to another performer and do some actions when it happens.

In this example, we allow the user to drag one list box and drop to another list box. When it happens, the selected item will be moved from the dragged list box to the list box it drops to.

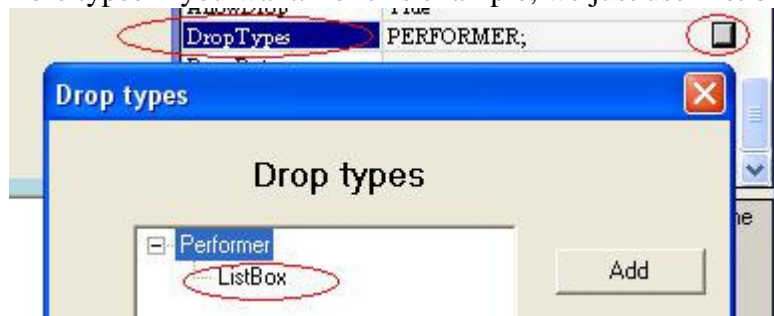
Still use the above example. Now we add the second list box.

Step 1. Add “PERFORMER” as the accepted drop type:

Open the properties of the second list box, set its “AllowDrop” to True, and set its “DropTypes” to include “Performer”. It will ask you to specify which performer types you want to accept. We select “List box”:



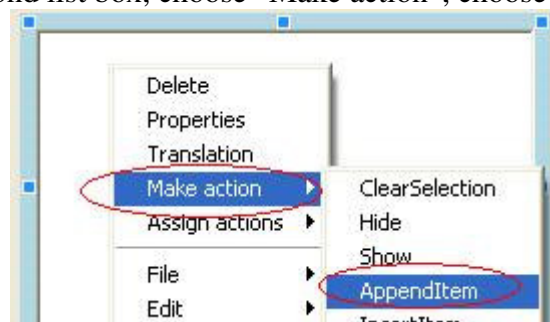
You may add more types if you want. For this example, we just use List box:



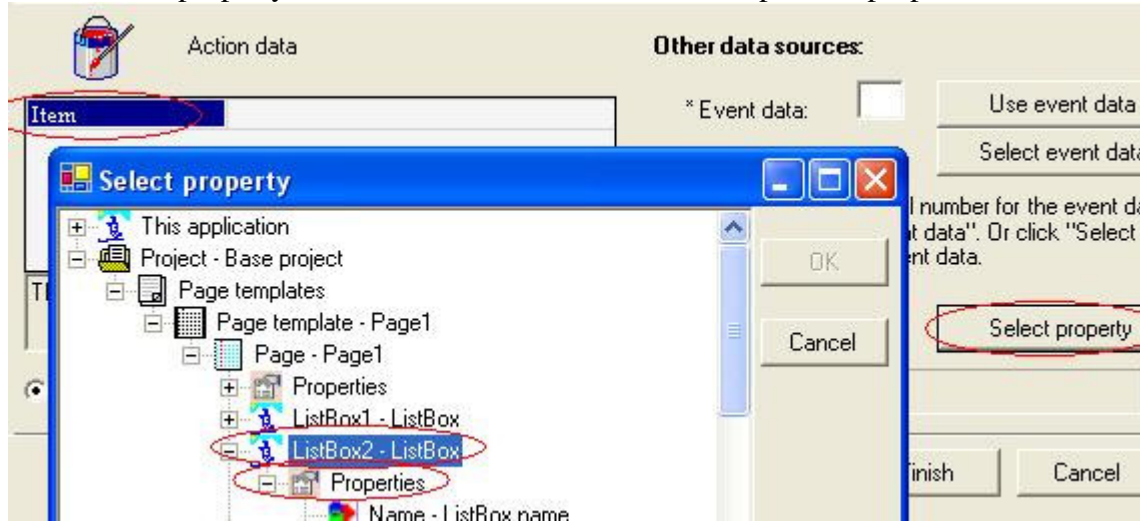
Step 2. Create an action to accept dropped data.

In this example, when the first list box is dropped to the second list box, we want the following two things to be done: 1. append the selected item in the first list box to the second list box; 2. remove the selected item in from the first list box. We will create two actions.

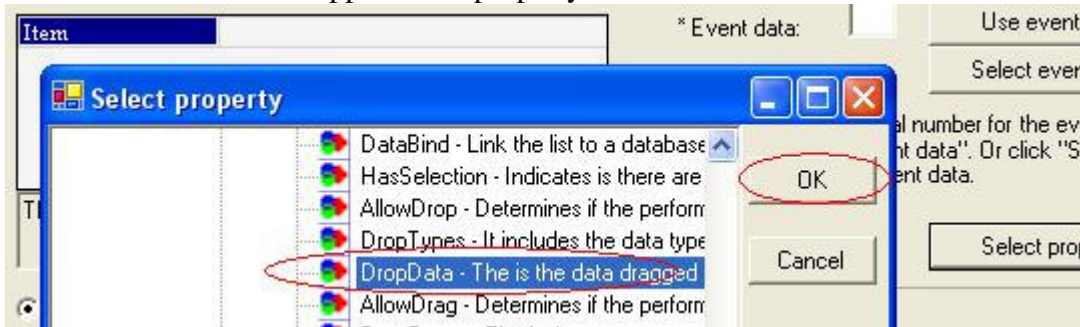
Right-click on the second list box, choose “Make action”, choose “AppendItem”:



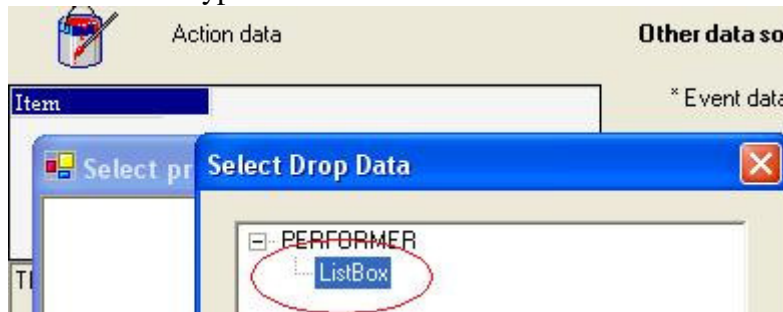
Give an action name, say, AddDroppedItem. The Action Data Dialogue box appears. Click “Select property” button. Find the second list box, expand its properties node:



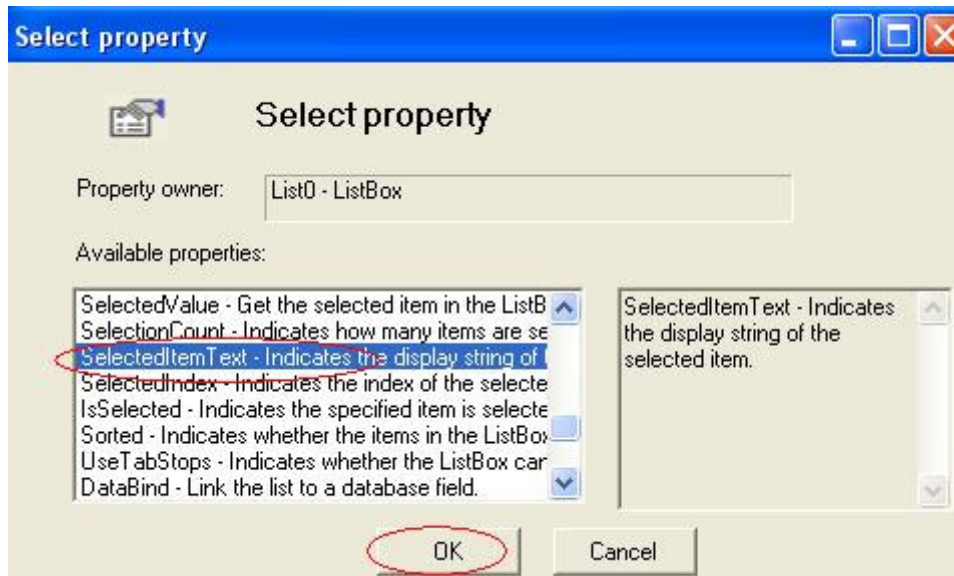
Scroll down and select “DroppedData” property:



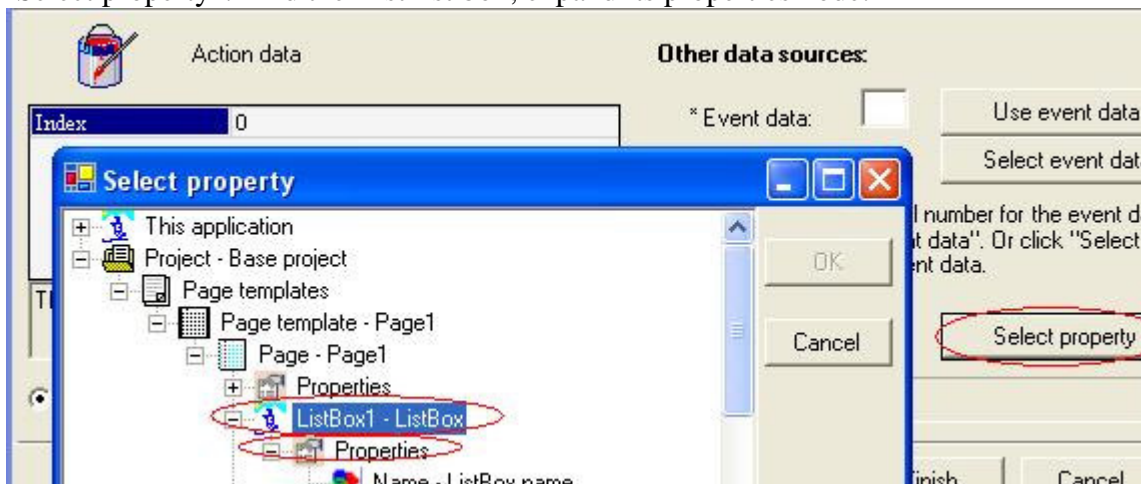
It asks you to select the data type for this action. Select List box:



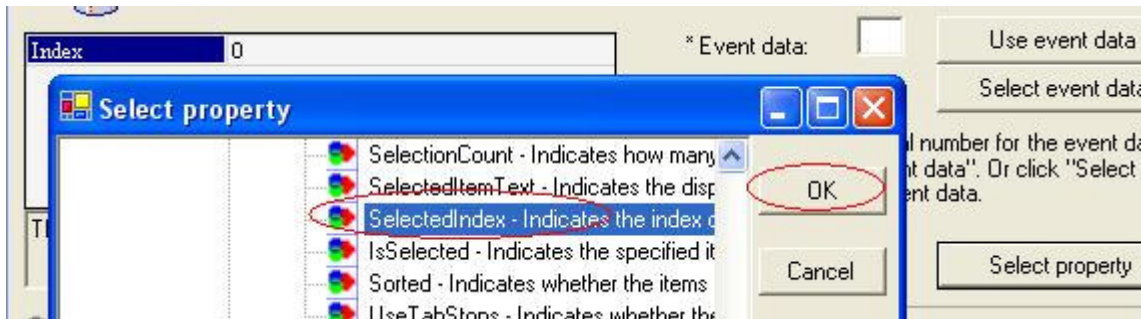
It then lists all the properties of the list box for you to select from. For this example, we may select “SelectedItemText”:



We are done with the first action. Now let's make the second action. Right-click on the first list box, choose "Make action", choose "RemoveItem", give an action name, say, RemoveDroppedItem. The Action Data dialogue box appears. Click "Select property". Find the first list box, expand its properties node:



Scroll down and select "SelectedIndex":



Step 3. Assign the actions to DragDrop and DropMe events

The action, `AddDroppedItem`, should be assigned to the “`DragDrop`” event of the second list box. That is, it is executed whenever a list box is dropped to it.

The action, `RemoveDroppedItem`, should be assigned to the “`DropMe`” event of the first list box. That is, it is executed whenever it is dropped to somewhere.

Step 4. Set “AllowDrag” property to True

This should be done for the first list box because we want it to be dragged.

We are done. You may run it and test it: Drop some files to the picture box so that the first list box will have some file names. Select a file name and drag it and drop it to the second list box, the file name will be moved from the first list box to the second list box.

You may notice that if there is not a file name selected in the first list box, and you still drag the first list box and drop it to the second list box, an empty row will be added to the second list box.

To prevent it, you may create an action to set the “`AllowDrag`” property of the first list box to its “`HasSelection`” property. We name this action `EnableDrag`. We assign this action to `MouseDown` event of the first list box. Now if there is not a file name selected, the user cannot do drag.

You may download this sample application from <http://www.limnor.com>