Honeywell

Lab Exercise – Creating Your First ActiveX Control

57311701L
11/99

TotalPlant

Notices and Trademarks

This module supports **TotalPlant** Solution (TPS) system network.

TPS is the evolution of TDC 3000^X.

Honeywell Inc.
Industrial Automation and Control
Automation College
2820 West Kelton Lane
Phoenix, AZ 85053-3028
1-800 852-3211

Lab Exercise

Introduction

The Visual Basic 5.0 Control Creation Edition makes creating ActiveX controls as easy as creating typical Visual Basic Applications.

Objectives

At the end of this concept lab exercise, you will be able to do the following:

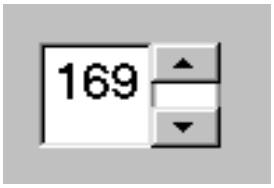
- Create your first ActiveX control using the Visual Basic Application.

Design Criteria

The lab is designed to give you a fast-track overview of the simple process involved in creating ActiveX controls with Visual Basic.

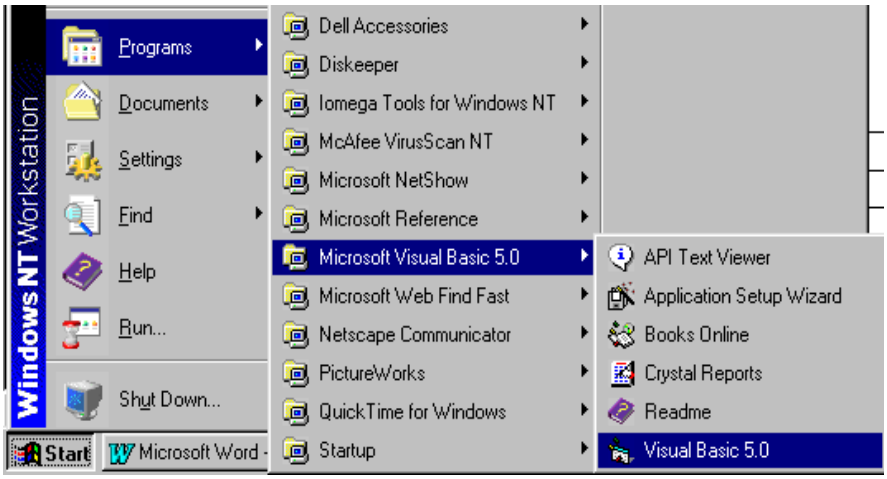
What you will build is called a spinner control. A spinner control is a graphical ActiveX control that allows the user to increment or decrement a value using a mouse.

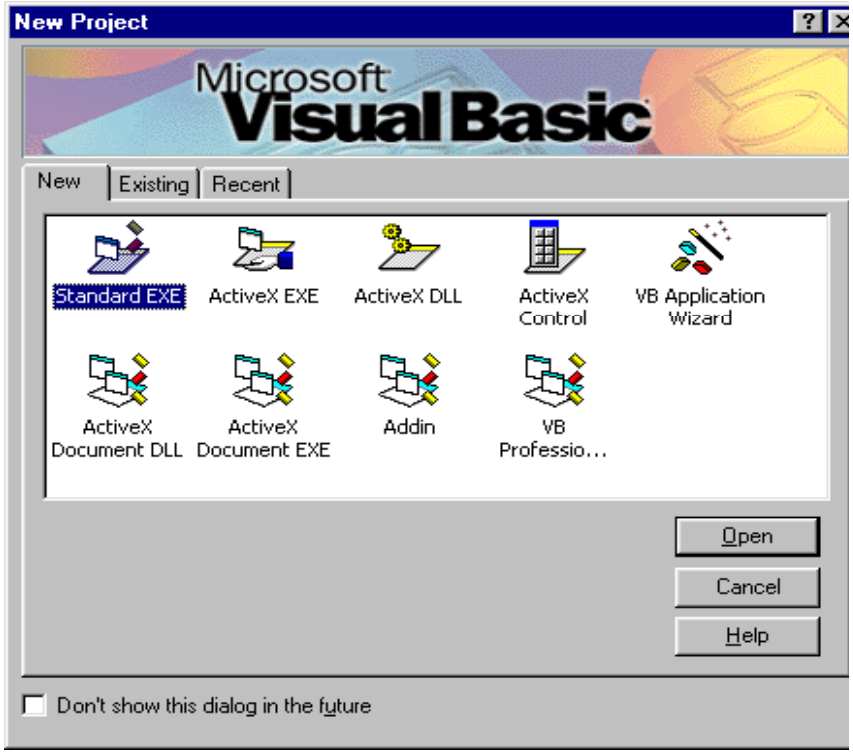
The spinner control display is shown in the following figure.

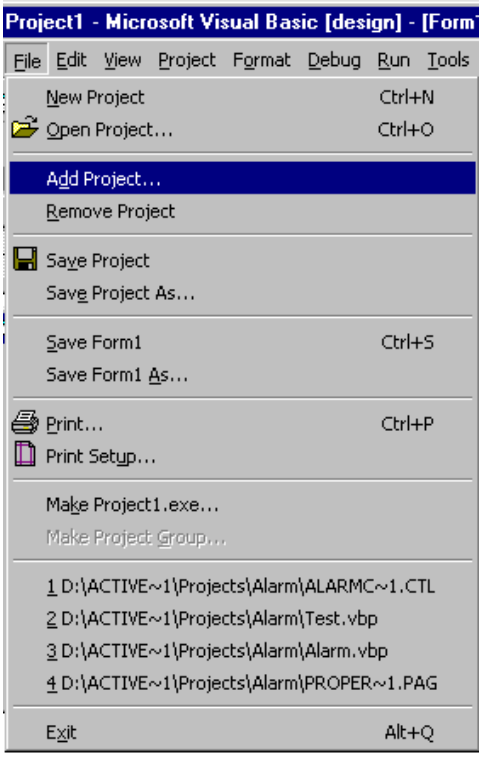


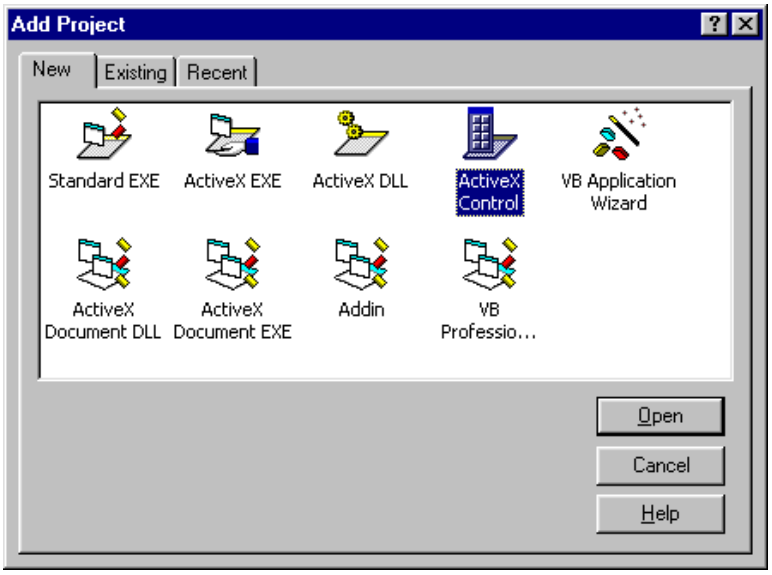
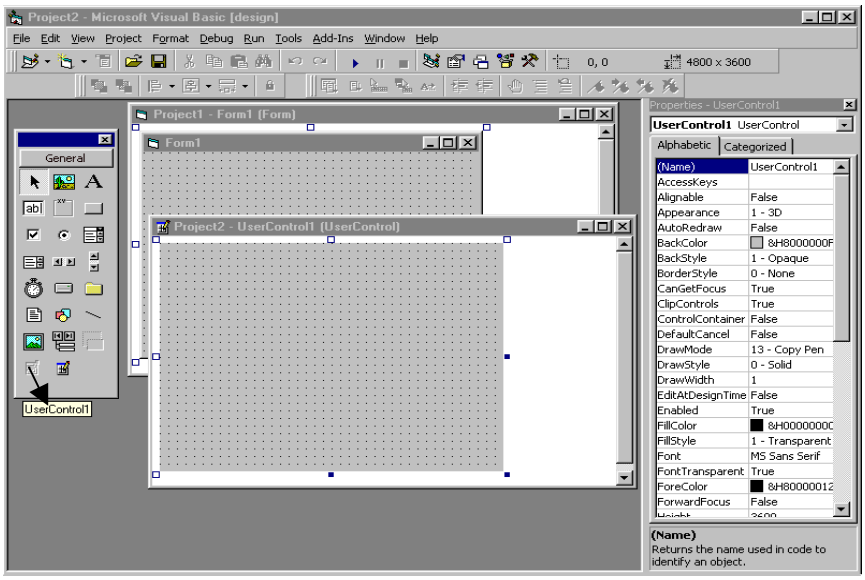
Now let's start building.

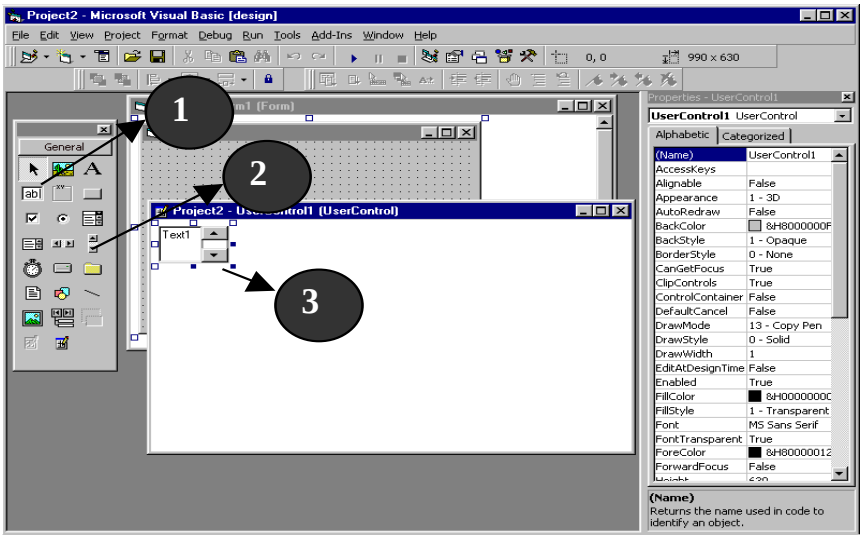
Lab Procedure

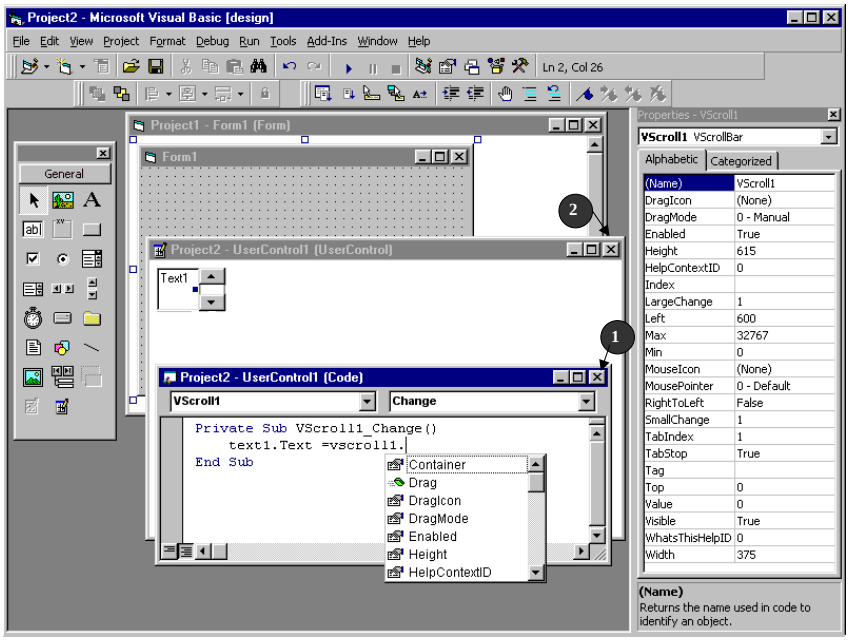
Step	Action
1.	**Create a Test Container** Start your Visual Basic Program. 

Step	Action
2.	The following will appear:  Click on the Standard.exe icon and then click open. This will create the host application. This host will be used as the test container for the spinner control that you are about to create.

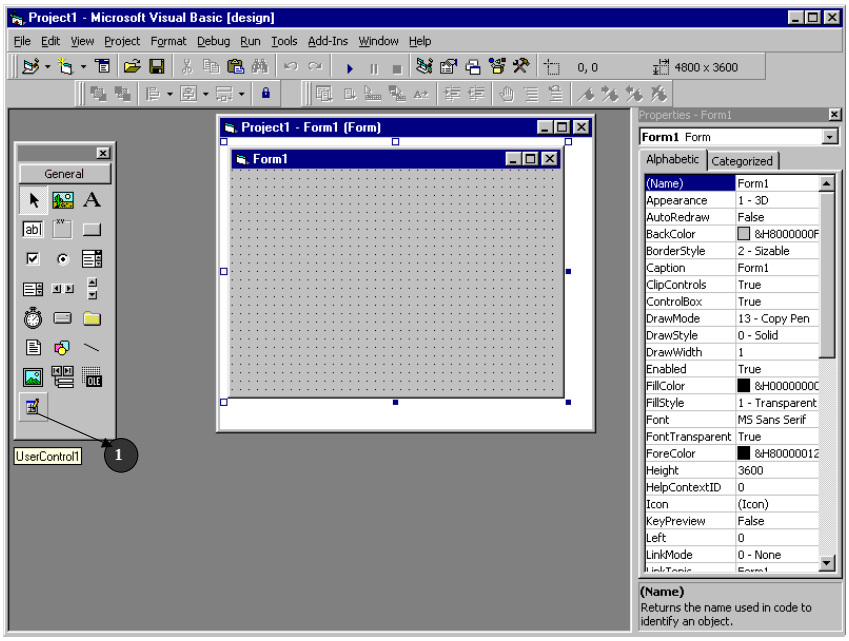
Step	Action
3.	**Add a Blank ActiveX Control Project** From the file menu, select the Add Project command. 

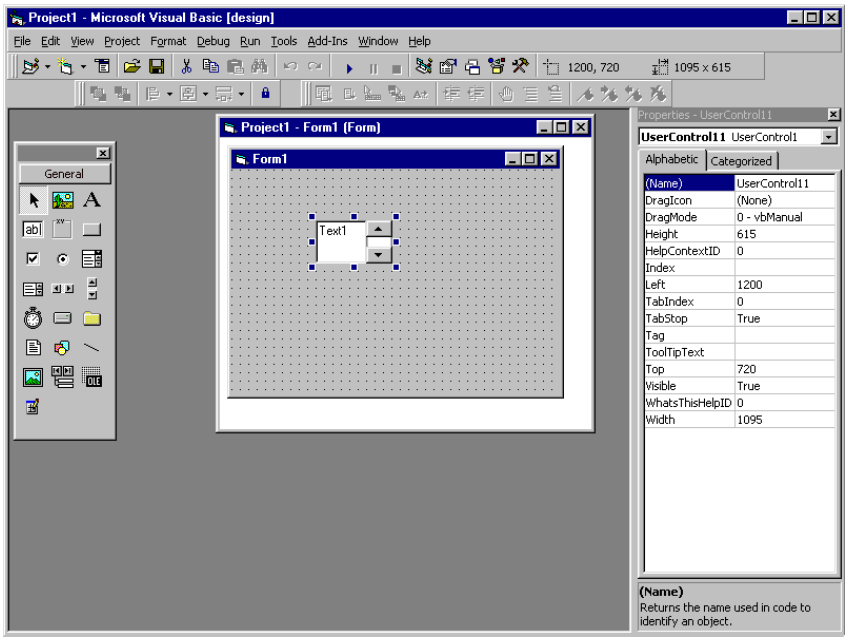
Step	Action
4.	The following will appear.  Select the ActiveX Control and click open.
5.	At this point there should be two projects open: Project1-Form1 and Project2-UserControl1. As you can see in the following diagram, both projects look extremely similar. Note that there is a new control that is now visible in the toolbox (labeled in the diagram as UserControl1). The icon in the Toolbox is gray (disabled) at this point. It will be enabled a few steps from now.  Visual Basic 5.0 uses the same visual metaphor for building ActiveX Controls as it uses in building applications. Using this metaphor, you first "draw" the interface, set some properties, write some event-driven code, and then you are on your way.

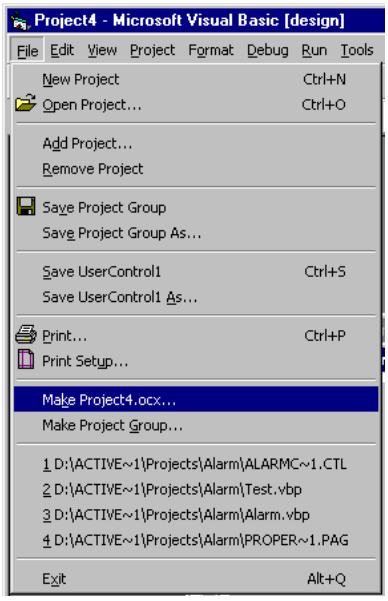
Step	Action
6.	**Draw the Visual Interface for the Control** Get your mouse ready; it is time to create the visual interface for the spinner control. The spinner control can be created using a powerful new feature of the Control Creation Edition - the ability to combine existing controls into new, more specialized controls. To create the spinner control, a standard textbox and a vertical scrollbar will be combined. First, click on the textbox control in the Visual Basic toolbox. The textbox control is identified with the number 1 in the figure below.  Using the mouse, draw a small textbox in the upper left-hand corner of the Project2 window. Second, click on the vertical scrollbar control in the toolbox and draw it just to the right of the textbox control. The vertical scrollbar control is identified with the number 2 in the previous figure. Select your project by clicking on the gray background; control-sizing handles will now appear around your project. Drag the bottom-right sizing handle so that only the 2 newly-added objects remain visible in Project 2 and no gray background is showing. The control-sizing handle is identified with the number 3 in the above figure. Your ActiveX spinner control should now look similar to the figure above.

Step	Action
7.	**Write Event Driven Code** At this point we now have a visual interface for a spinner control. The next step is to write some event-driven code that will place the current value of the vertical scrollbar into the textbox. When the user clicks on the up or down arrow of the vertical scrollbar, the value displayed in the textbox will increment or decrement. To make this happen, some code needs to be written in the Change event of the vertical scroll bar. Double click on the vertical scrollbar that you just drew. This will display the code window. In the code window, type the following line of code: `Text1.text = vscroll1.value` As you typed the line of code, you experienced a small glimpse of the new "intelligence" added to the Visual Basic 5.0 development environment. As soon as you typed the "dot", Visual Basic makes properties automatically available when needed. After the code is entered, close the code window by clicking on the close box. (#1 in the figure below). Finally, close the spinner control form on its close box (#2 in the figure below). 

Step	Action
8.	**Use and Test the Control** At this point, if all has gone well and you closed the spinner form, the new control no longer appears gray in the toolbox and is ready to be used/tested (#1 in the figure below.



Step	Action
9.	To test the new control, click on it in the toolbox and draw it on the form as shown below. Press F5 to run the application. As you click the up and down arrow in the spinner control, the value in the textbox changes, just as you coded it. 

Step	Action
10.	**Prepare your Control for the Display Builder** Reopen your Project2-UserControl1 form by double-clicking it in the Project Group window.  If you are satisfied with your control, select Make project.ocx from the File menu. If the Visual Basic application is on the same GUS station, then your control is automatically registered in the Display Builder. If not, refer back to your class notes for how to register OCX controls. Tip for GUS-compatible ActiveX Controls Using VB5 – Be sure to select the "Binary Compatibility" option for your project. Failure to do so will replace your ActiveX's existing binary ID, by which other applications (e.g. your GUS displays) reference the ActiveX control. Under the Project menu, select "Project2 Properties"; select the Component tab; select the Binary Compatibility option; then enter the control OCX name. To summarize Binary Compatibility, selecting this option ensures that if your ActiveX control is modified or enhanced at a later date, the new version will work (or be compatible with) an application compiled with the previous version. Compatibility will be maintained among projects/displays that have been compiled using your old and enhanced/modified ActiveX control. Launch the Display Builder and insert your control, validate and run the display.

Last Page