18. Understanding of VB Controls

18.1 Types of Controls

There are three basic categories of controls in Visual Basic. They are:

(1) Intrinsic controls, such as the command button and frame controls. Intrinsic controls are always included in the toolbox.

(2) ActiveX controls, these include DataCombo, DataList, etc., which exist as separate files.

(3) Insertable objects, such as MS Excel worksheet object, MS Word document, or others, which can be implemented via Automation.

We just used Data, DBGrid, textbox, command button controls. There are quite a few built-in VB controls. When you first start the VB project, the toolbox shows the default controls (Figure 1). However, other controls can be added into the toolbox from Component menu under Project.

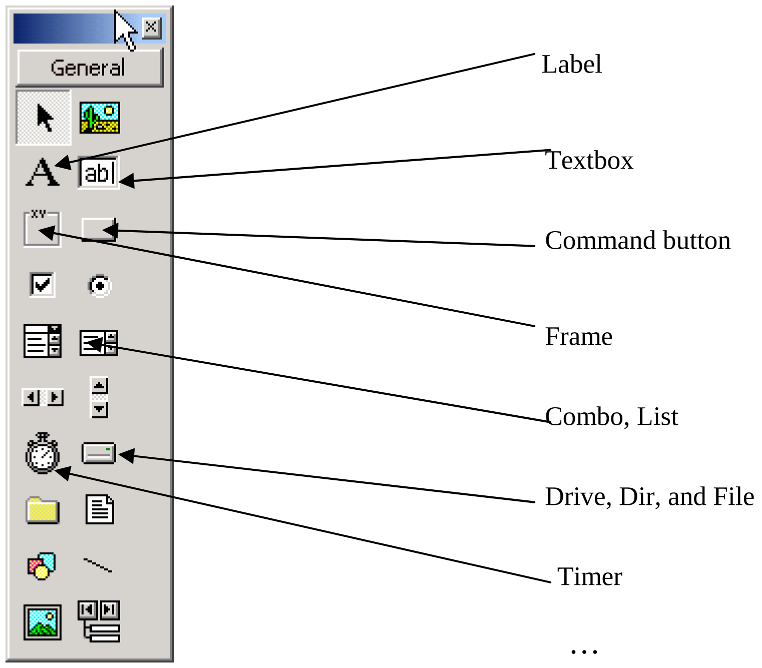


Figure 1. Default toolbox of VB.

Visual Basic forms and controls are objects which expose their own **properties, methods, and events.** Properties can be thought of as an object's attributes, methods as its actions, and events as its responses.

I am going to develop a couple of small projects to demonstrate how to use VB controls.

18.2 Using Timer, Option Button, Frame, and Checkbox Controls

In this example, you can select the harvesting machines by using radio buttons to form a harvesting system, which is displayed in a text box. Meanwhile, the font of the text in a textbox should be allowed to change to be italic, bold, or both.

Creating Interface

Put the following controls on the form:
- a textbox to display the harvesting system
- a label to show the clock
- three frames to hold radio buttons and checkboxes
- a timer control
- a command button to end the application

Table 1 lists the property settings of these controls. The interface of your application will look like what is displayed in Figure 2.

Table 1. Property setting of controls.

Control	Property	Setting
Form1	Name	frmVBControl
	Caption	VB Control Test
Text1	Name	txtDisplay
Label1	Name	lblTime
Frame1	Caption	Felling Machine
Frame2	Caption	Skidding Machine
Frame3	Caption	Display
Radio Button1	Name	optCS
	Caption	Chainsaw
Radio Button2	Name	optFB
	Caption	Feller-buncher
Radio Button3	Name	optHV
	Caption	Harvester
Radio Button4	Name	optCD
	Caption	Cable skidder
Radio Button5	Name	optGD
	Caption	Grapple skidder
Radio Button6	Name	optFD
	Caption	Forwarder
Checkbox1	Name	chkBold
	Caption	Bold
Checkbox2	Name	chkItalic
	Caption	Italic
Timer1	Interval	500 (half a second)
Command1	Name	cmdClose
	Caption	Close

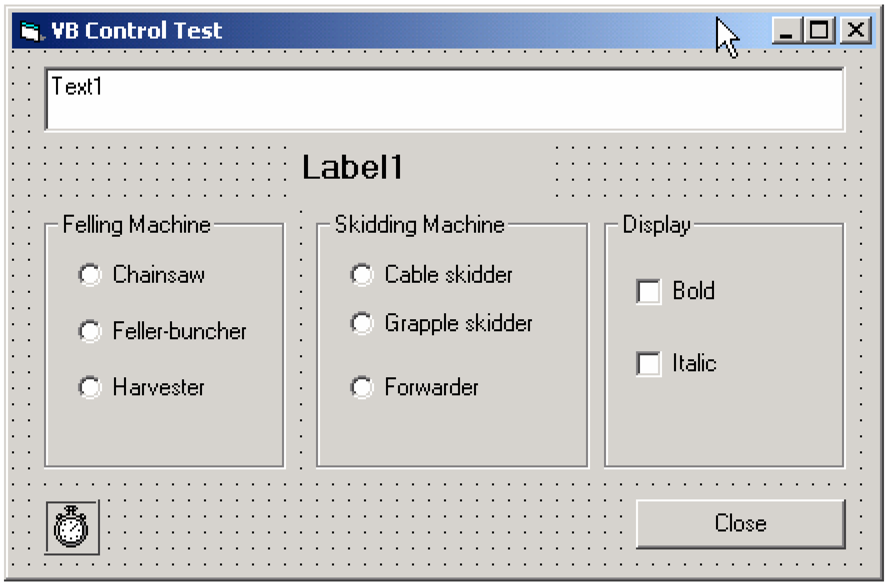


Figure 2. Interface of control test.

Writing Codes

Double-click the related controls and type the event codes there.

‘Add the following code in general declaration section:

```
' set up two string variables to hold the captions
Dim strFeller As String
Dim strSkidder As String

Sub DisplayCaption()
   ' concatenate the caption with the two string
   ' variables.
   txtDisplay.Text = "You selected a " & _
    strFeller & " and " & strSkidder & " harvesting system."
End Sub
```

‘Add the code under command button – cmdClose:

```
Private Sub cmdClose_Click()
   Unload Me   'unload the form
End Sub
```

‘Double-click form1 and the code:

```
Private Sub Form_Load()
   ' invoke a Click event in the default options
   ' to update the label caption
   optCS_Click
   optCD_Click
End Sub
```

‘Double-click each radio button and add the following codes there:

```
Private Sub optHV_Click()
   ' assign a value to the first string variable
   strFeller = "Harvester"
   ' call the subroutine
   Call DisplayCaption
End Sub


Private Sub optCS_Click()
   ' assign a value to the first string variable
   strFeller = "Chainsaw"
   ' call the subroutine
   Call DisplayCaption
End Sub


Private Sub optFB_Click()
   ' assign a value to the first string variable
   strFeller = "Feller-buncher"
   ' call the subroutine
   Call DisplayCaption
End Sub


Private Sub optCD_Click()
   ' assign a value to the second string variable
   strSkidder = "Cable skidder"
   ' call the subroutine
   Call DisplayCaption
End Sub


Private Sub optGD_Click()
   ' assign a value to the second string variable
   strSkidder = "Grapple skidder"
   ' call the subroutine
   Call DisplayCaption
End Sub


Private Sub optFD_Click()
   ' assign a value to the second string variable
   strSkidder = "Forwarder"
   ' call the subroutine
   Call DisplayCaption
End Sub
```

‘Add code associated with checkboxes:

```
Private Sub chkBold_Click()
' The Click event occurs when the check box changes state.
' Value property indicates the new state of the check box.
   If chkBold.Value = 1 Then     ' If checked.
      txtDisplay.FontBold = True
   Else                          ' If not checked.
```

```
        txtDisplay.FontBold = False
    End If
End Sub

Private Sub chkItalic_Click()
' The Click event occurs when the check box changes state.
' Value property indicates the new state of the check box.
    If chkItalic.Value = 1 Then     ' If checked.
        txtDisplay.FontItalic = True
    Else                            ' If not checked.
        txtDisplay.FontItalic = False
    End If
End Sub
```

‘Add the code for timer:

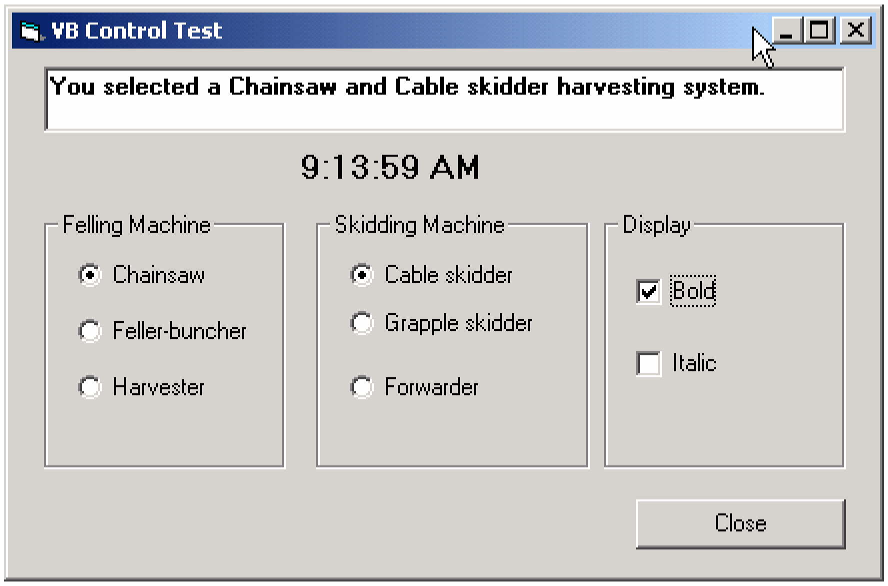
```
Private Sub Timer1_Timer()

    If lblTime.Caption <> CStr(Time) Then
        lblTime.Caption = Time
    End If

End Sub
```

Running the Application

Start the program, check radio button and checkbox, you will see what will happen. The running application looks like Figure 3.



You selected a Chainsaw and Cable skidder harvesting system.

9:13:59 AM

Figure 3. Running the application.

18.3 Using Drive, Dir, File, Combo, List, Frame Controls

This application here shows you how to use the Drive, Directory, and File controls to read a table in a database from the secondary storage. Then, you click a button to populate the data in this table into a Combo box. In order to do that, we need to use Microsoft Data Access Object (DAO). Finally, you need to add the selected item in the Combo box to a list box by clicking another button.

<u>Creating Interface</u>

We need to add the following controls to the form:

- Frame1 control
- Frame2
- Drive1 control
- Dir1 control
- File1 control
- Label1
- Combo1
- Combo2
- List1
- Command button – cmdPopulate
- Command button – cmdAdd
- Command button – cmdClose
- Command button - cmdGetTable

To reference MS DAO:

a. Select References… from the Project menu, then the References dialog box will be displayed.
b. Find Microsoft DAO 3.51 Object Library in the list box and select the check box to its left.
c. Click OK button.

In order to save the time, we will use the default setting for most of the controls. The properties of controls that need to be reset are listed in table 2. The interface should look like what we have in Figure 4.

Table 2. The property settings.

Control	Property	Setting
Form1	Name	frmDrive
	Caption	VB Controls
Command1	Name	cmdPopulate
	Caption	Populate
Command2	Name	cmdAdd
	Caption	Add
Command3	Name	cmdClose
	Caption	Close
Command3	Name	cmdGetTable
	Caption	Get Table

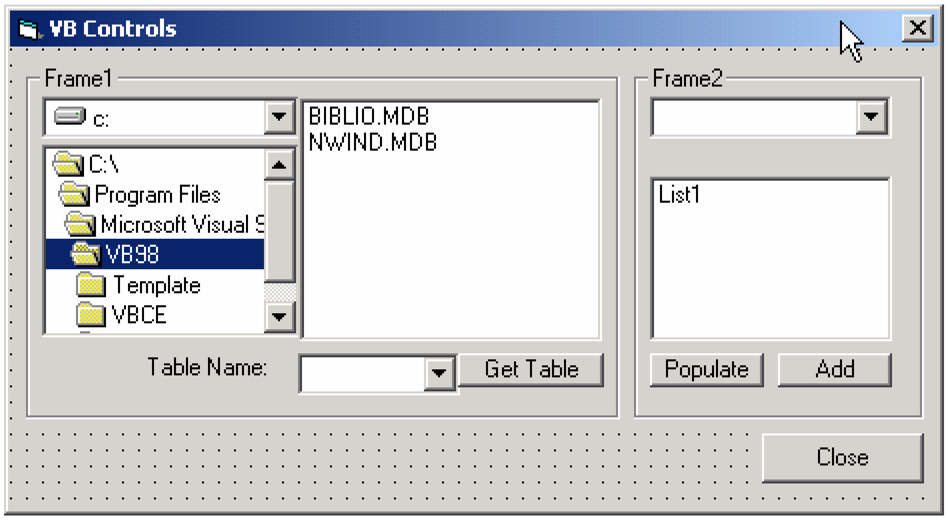


Figure 4. Interface.

Writing Code

```
'Declare the database connection in the General Declaration section
Dim dbInput As Database
Dim rsInput As Recordset


'Add for command button cmdClose
Private Sub cmdClose_Click()

   Unload Me

End Sub
```

```
‘For command button cmdGetTable
Private Sub cmdGetTable_Click()

   Dim filePath As String

   If Right(File1.Path, 1) <> "\" Then
      filePath = File1.Path & "\" & File1.FileName
   Else
      filePath = File1.Path & File1.FileName
   End If

   Set dbInput = OpenDatabase(filePath)
   Combo2.Clear
   For Each tlTableName In dbInput.TableDefs()
      Combo2.AddItem tlTableName.Name
   Next tlTableName
   cmdPopulate.Enabled = True

End Sub

‘For command button cmdPopulate
Private Sub cmdPopulate_Click()


   Set rsInput = dbInput.OpenRecordset(Combo2.Text)

   Combo1.Clear
   If Not (rsInput.EOF And rsInput.BOF) Then
      rsInput.MoveFirst
      While Not rsInput.EOF
         Combo1.AddItem rsInput.Fields(0)          ‘display the first field
         rsInput.MoveNext
      Wend
   End If

   cmdPopulate.Enabled = False

End Sub


End Sub

‘For command button cmdAdd
Private Sub cmdAdd_Click()

   List1.AddItem Combo1.Text

End Sub
```

```
'Double-click the drive, dir, and file controls and add the code there

Private Sub Dir1_Change()
   Dir1.Path = Drive1.Drive
End Sub

Private Sub File1_Click()

   File1.Path = Dir1.Path

End Sub
```

Running the Project

- Start the program
- Find a MS Access database file
- Click "Get Table" button
- Retrieve a table name from the combo box
- Click Populate button
- You should populate the data in the Combo box
- Select an item and add it to the list box

The interface of your running application will look like Figure 5.

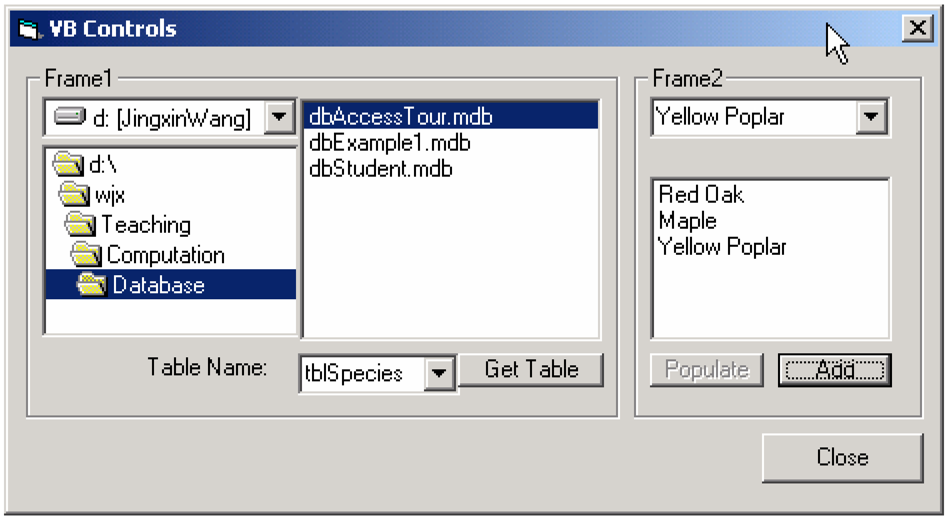


Figure 5. Running the application.