Chapter 3 – Fundamentals of Programming in VB.NET

• Part I
  • VB.NET Controls
  • VB.NET Events

3.1 VB.NET Controls

• Invoking VB.NET
• A Text Box Walkthrough
• A Button Walkthrough
• A Label Walkthrough
• A List Box Walkthrough
• The Name Property
• A Help Walkthrough
• Fonts / Auto Hide
A Text Box Walkthrough

• Drag Text Box from ToolBox
• Sizing
• Delete
• Properties
  • Text, Color, Font, Size, Location, Visible, Enabled

A Button Walkthrough

• Add the button
• Change the Text property
Add an "access key"

A Label Walkthrough

- Add the Label
- Change the Text property
- Resize the control
A List Box Walkthrough

- Add the List Box
- Add data
- Resize the control

The Name Property

- How the programmer refers to a control in code
- Name must begin with a letter
- Must be less than 215 characters long
- May include numbers and the underscore
- Naming convention: use appropriate 3 character naming prefix
  - First three letters identifies the type of control
  - Remaining letters identifies the purpose
  - E.g. a text box to store a social security number would be called **txtSocialSecurity**
Control Name Prefixes

<table>
<thead>
<tr>
<th>Control</th>
<th>Prefix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>button</td>
<td>btn</td>
<td>btnComputeTotal</td>
</tr>
<tr>
<td>label</td>
<td>lbl</td>
<td>lblInstructions</td>
</tr>
<tr>
<td>list box</td>
<td>lst</td>
<td>lstOutput</td>
</tr>
<tr>
<td>text box</td>
<td>txt</td>
<td>txtAddress</td>
</tr>
</tbody>
</table>

Fonts

- Proportional width fonts take up less space for "I" than for "W" – like Microsoft Sans Serif
- Fixed-width fonts take up the same amount of space for each character – like Courier New
- Fixed-width fonts are good for tables
Auto Hide

- Hides tool windows when not in use
- Vertical push pin icon indicates auto hide is disabled
- Click the push pin to make it horizontal and enable auto hide

Viewing the Code

- The GUI Forms Designer generates textual code
  - Prior to VB programmers wrote everything in textual code
- Click on the “Form1.VB” tab to see the code (not the design tab)
3.2 VB.NET Events

• An Event Procedure Walkthrough

An Event Procedure Walkthrough

• An event is an action, such as:
  • The user clicks on a button
  • A form is minimized
  • The mouse enters or exits a control
  • The form is re-drawn
  • Usually, nothing happens until an event occurs
The three steps in creating a VB.NET program:

1. Create the interface; that is, generate, position, and size the objects.
2. Set properties; that is, configure the appearance of the objects.
3. Write the code that executes when events occur.

Changing Properties

- Properties are changed in code with the following:
  \[controlName.property = setting\]
- This is an assignment statement
- Examples:
  \[txtBox.ForeColor = Color.Red\]
  \[txtName.Text = “Hello There”\]
  \[txtName.Visible = False\]
  \[txtName.Location.X = 100\]
Adding Code to an Event

- To add code for an event:
  - In the VB Code Window select the control on the left side menu and the event of interest on the right side menu
  - Or double-click the control in the designer to bring up the most common event for that control

Program Region
Event Procedures

Private Sub objectName_event(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles objectName.event

Shown in the book as:
Private Sub objectName_event(...) Handles objectName.event

Structure of an Event Procedure

Private Sub objectName_event(...) Handles objectName.event

    statements ' Your code goes here

End Sub
IntelliSense

Automatically pops up to give the programmer help.

Code for Walkthrough

```vbnet
Private Sub txtFirst_TextChanged(Handles txtFirst.TextChanged)
    txtFirst.ForeColor = Color.Blue
End Sub

Private Sub btnRed_Click(Handles btnRed.Click)
    txtFirst.ForeColor = Color.Red
End Sub

Private Sub txtFirst_Leave(Handles txtFirst.Leave)
    txtFirst.ForeColor = Color.Black
End Sub
```
Assigning properties in code

• The following won’t work:

  `Form1.Text = "Demonstration"`

• The form is referred to by the keyword `Me`.

  `Me.Text = "Demonstration"

The Declaration Statement of an Event Procedure

• A declaration statement for an event procedure:

  `Private Sub btnOne_Click(...) Handles btnOne.Click`

• The name can be changed at will. For example

  `Private Sub ButtonPushed(...) Handles btnOne.Click`

• Handling more than one event:

  `Private Sub ButtonPushed(...) Handles btnOne.Click, btnTwo.Click`
Changing Control Name

• Careful – if you create events for a control and then change the name of the control to something else, some events may keep the old name
  • And don’t get invoked when the event occurs
  • You the programmer would need to change the name within the code to match the new name
• Easiest to not change the control name!