Visual Basic - Chapter 2

Mohammad Shokoohi

* Adopted from An Introduction to Programming Using Visual Basic 2010, Schneider
Chapter 2 – Visual Basic, Controls, and Events

2.1 An Introduction to Visual Basic
2.2 Visual Basic Controls
2.3 Visual Basic Events
2.1 An Introduction to Visual Basic 2010

- Why Windows and Why Visual Basic
- How You Develop a Visual Basic Application
- The Different Versions of Visual Basic
Visual Basic 2010

• Language used to create Windows applications.
• Provides a Graphical User Interface or GUI.
• The sequence of instructions executed in the program is controlled by events.
Sample Input Screen
How to Develop a Visual Basic Application

• Design the Interface for the user.
• Determine which events the controls on the window should recognize.
• Write the event procedures for those events.
Different Versions of Visual Basic

- Version 1.0 – 1991
- Version 2.0 – 1992
- Version 3.0 – 1993
- Version 4.0 – 1995
- Version 5.0 – 1997
- Version 6.0 – 1998
- Visual Basic.NET – 2002 (NOT BACKWARD COMPATIBLE WITH EARLIER VERSIONS)
- Visual Basic 2005 – November 2005
- Visual Basic 2008 – November 2007
- Visual Basic 2010 – April 2010
2.2 Visual Basic Controls

- Starting a New Visual Basic Program
- Text Box Control
- Button Control
- Label Control
- List Box Control
- Name Property
- Fonts / Auto Hide
- Positioning and Aligning Controls
Start a New Project

Start Page - Microsoft Visual Basic 2010 Express

- New Project... Ctrl+N
- Open Project... Ctrl+O
- Open File...
- Close
- Close Project
- Save Selected Items Ctrl+S
- Save Selected Items As...
- Save All Ctrl+Shift+S
- Export Template...
- Page Setup...
- Print... Ctrl+P
- Recent Files
- Recent Projects and Solutions
- Exit Alt+F4
New Project Dialog Box

Select the Windows Forms Application template and click on the OK button.
Initial Visual Basic Screen
Toolbox
4 Ways to Place a Control from the Toolbox onto the Form Designer

- Double-click
- Drag and Drop
- Click, Point, and Click
- Click, Point, and Drag
Four Controls at Design Time

To select a control, click on it. Sizing handles will appear when a control is selected.
Text Box Control

- Used for input and output
- When used for output, ReadOnly property is set to True
Properties Window

Press F4 to display the Properties window for the selected control.

categorized view  alphabetical view
Properties Window (continued)

- Selected control
- Properties pane
- Description pane
- Settings
Some Often Used Properties

- Text
- Autosize
- Font.Name
- Font.Size
- ForeColor
- BackColor
- ReadOnly
Setting Properties

- Click on property name in left column.
- Enter its setting into right column by typing or selecting from options displayed via a button or ellipses.
Setting the ForeColor Property

1. Click on ForeColor.
2. Click on button at right of settings box.
3. Click on Custom tab to obtain display shown.
4. Click on a color.
Font Property

1. Click on Font in left column.
2. Click on ellipsis at right of settings box to obtain display shown.
3. Make selections.
Button Control

- The caption on the button should indicate the effect of clicking on the button.

![Diagram showing properties of a button control]

**Text property**

Text
The text associated with the control.
Add an Access Key

Properties

Button1  System.Windows.Forms.Button

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TabStop</td>
<td>True</td>
</tr>
<tr>
<td>Tag</td>
<td></td>
</tr>
<tr>
<td>Text</td>
<td>&amp;Calculate Balance</td>
</tr>
<tr>
<td>TextAlign</td>
<td>MiddleCenter</td>
</tr>
</tbody>
</table>

Text
The text associated with the control.
Label Control

- Used to identify the contents of a text box.
- Text property specifies caption.
- By default, label automatically resizes to accommodate caption on one line.
- When the AutoSize property is set to False, label can be resized manually. AutoSize is used primarily to obtain a multi-rowed label.
List Box Control

- Initially used to display several pieces of output.
- In Chapter 4 used to select from a list.
The Name Property

- Used by the programmer to refer to a control in code
- Setting for Name property near top of Properties window
- Use appropriate 3-character naming prefix
- Use descriptive names
## 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>btnCompute</td>
</tr>
<tr>
<td>label</td>
<td>lbl</td>
<td>lblAddress</td>
</tr>
<tr>
<td>text box</td>
<td>txt</td>
<td>txtAddress</td>
</tr>
<tr>
<td>list box</td>
<td>lst</td>
<td>lstOutput</td>
</tr>
</tbody>
</table>


Renaming the Form

- Initial name is Form1
- The Solution Explorer window lists a file named Form1.vb.
- To rename the form, change the name of this file to *newName*.vb
- *newName* should begin with prefix *frm*. 
Fonts

• Proportional width fonts, such as Microsoft Sans Serif, use less space for "I" than for "W"
• Fixed-width fonts take up the same amount of space for each character – like Courier New
• Fixed-width fonts are used for tables.
Auto Hide

• Hides Toolbox 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.
Positioning Controls

![Diagram of a form with a button labeled Button1, with a proximity line indicated]
Aligning Bottoms of Controls

snap line
Aligning Middles of Controls
Tab Order

The tab indices determine the order in which controls receive the focus during tabbing.

The control whose TabIndex property is set to 0 has the focus when the program begins.
2.3 Visual Basic Events

- An Event Procedure Walkthrough
- Properties and Event Procedures of the Form
- The Header of an Event Procedure
Event

- An **event** is an action, such as the user clicking on a button.
- Usually, nothing happens in a Visual Basic program until the user does something and raises an event.
- What happens is determined by statements inside the event procedure.
Sample Statements

- `txtBox.ForeColor = Color.Red`
- `txtBox.Visible = True`
- `txtBox.Text = "Hello World"

General Form:

```
controlName.property = setting
```
Sample Form

[Diagram of a demonstration form with labels: txtFirst, txtSecond, btnRed]
Focus

• When you click on a text box, a cursor appears in the text box, and you can type into the text box.

• Such a text box is said to have the **focus**.

• If you click on another text box, the first text box loses the focus and the second text box receives the focus.
Examples of Events

- btnShow.Click
- txtBox.TextChanged
- txtBox.Leave

General Form:

controlName.event
The Three Steps in Creating a Visual Basic 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.
Code Editor

- Code Editor tab
- Form Designer tab

![Code Editor and Form Designer tabs](image-url)
Display Events for a Control

- Select the control
- Click on the Events button ( events button ) in the Properties window
Structure of an Event Procedure

header

```vbnet
Private Sub objectName_event(...) Handles objectName.event
    statements
End Sub
```

(... is filled automatically with (ByVal sender As System.Object, ByVal e As System.EventArgs))
Create an Outline for an Event
Procedure

• Double-click on a control
  or
• Select a control, click on the Events button in the Properties window, and double-click on an event
  (We nearly always use the first method.)
Double-click on txtFirst to create the outline for the Code Editor
Public Class frmDemo
    Private Sub txtFirst_TextChanged(...) Handles txtFirst.TextChanged
        txtFirst.ForeColor = Color.Blue
    End Sub
End Class
IntelliSense

Automatically pops up to help the programmer.

*txtFirst.*
Code Editor

click tab to return to Form Designer
Double-click on btnRed to return to Code Editor and add the outline of an event procedure
Public Class frmDemo
    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
End Class
Event Procedure txtFirst.Leave

- Select txtFirst on the form
- Click on the Events button in the Properties window
- Double-click on Leave
Private Sub txtFirst_Leave(...) Handles txtFirst.Leave
    txtFirst.ForeColor = Color.Black
End Sub

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
Header of Event Procedure

Private Sub btnRed_Click(...) Handles btnRed.Click
  Name, can be changed.
  Identifies event

Private Sub Button_Press(...) Handles btnRed.Click
Handling Multiple Events

An event procedure can be invoked by two events.

```vbnet
Private Sub Happening(...) Handles btnRed.Click, txtSecond.Leave
    txtFirst.ForeColor = Color.Red
End Sub
```
Altering Properties of the Form

• The following won't work:

```plaintext
frmDemo.Text = "Demonstration"
```

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

```plaintext
Me.Text = "Demonstration"
```
Open and Run an Existing Program

- Click on *Open Project* in the *File* menu.
- Navigate to the program’s folder.
- Double-click on the program’s folder to open it.
- Double-click on the file with extension *sln*.
- In the Solution Explorer double-click on the file with extension *vb*. (The Form Designer will appear.)
- Press F5 to run the program.