Visual Basic - Chapter 2

Mohammad Shokoohi

* Adopted from An Introduction to Programming Using Visual Basic 2010, Schneider

Chapter 2 –Visual Basic, Controls, and Events

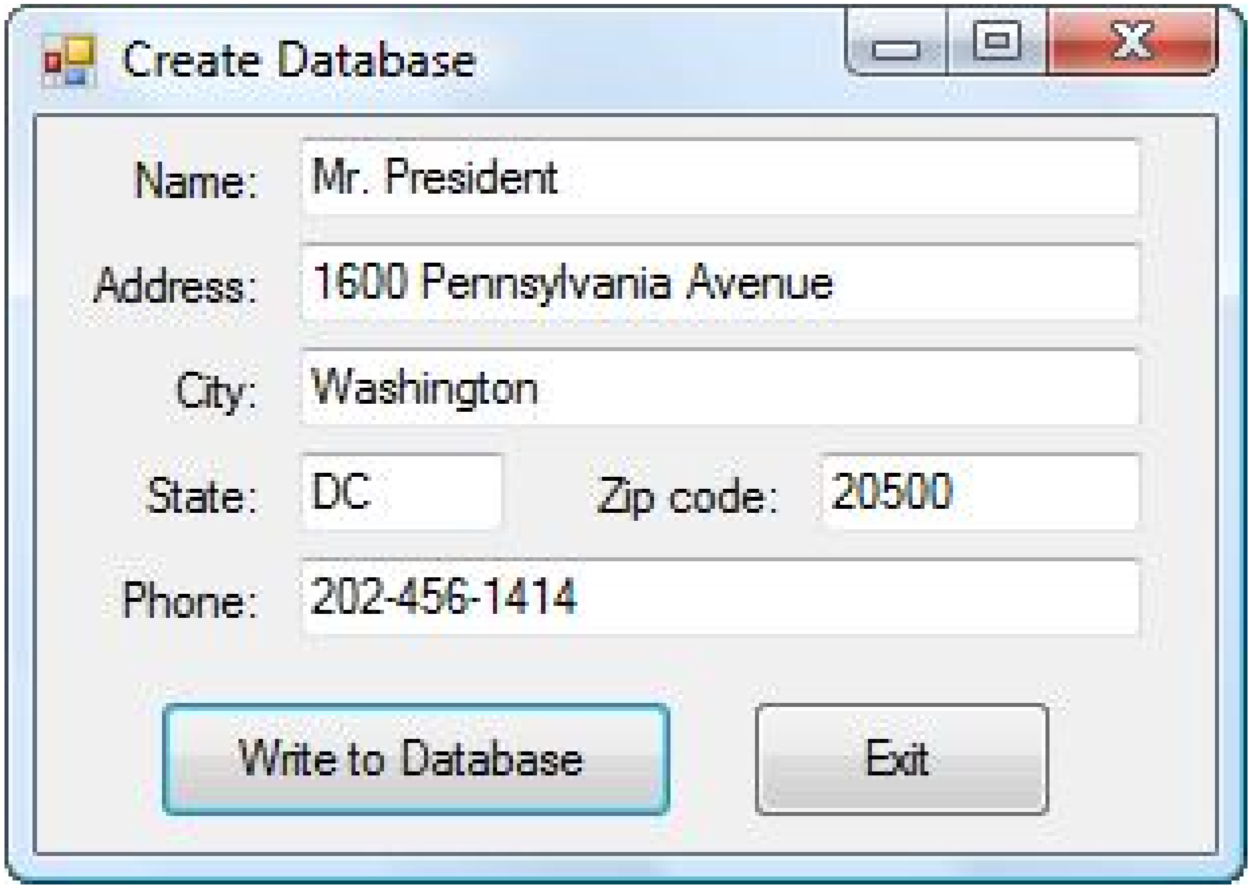
2.1 An Introduction to Visual Basic 2010

- 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 **G**raphical **U**ser **I**nterface or GUI.
- The sequence of instructions executed in the program is controlled by events.

Sample Input Screen



Create Database

Name: Mr. President

Address: 1600 Pennsylvania Avenue

City: Washington

State: DC

Zip code: 20500

Phone: 202-456-1414

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.

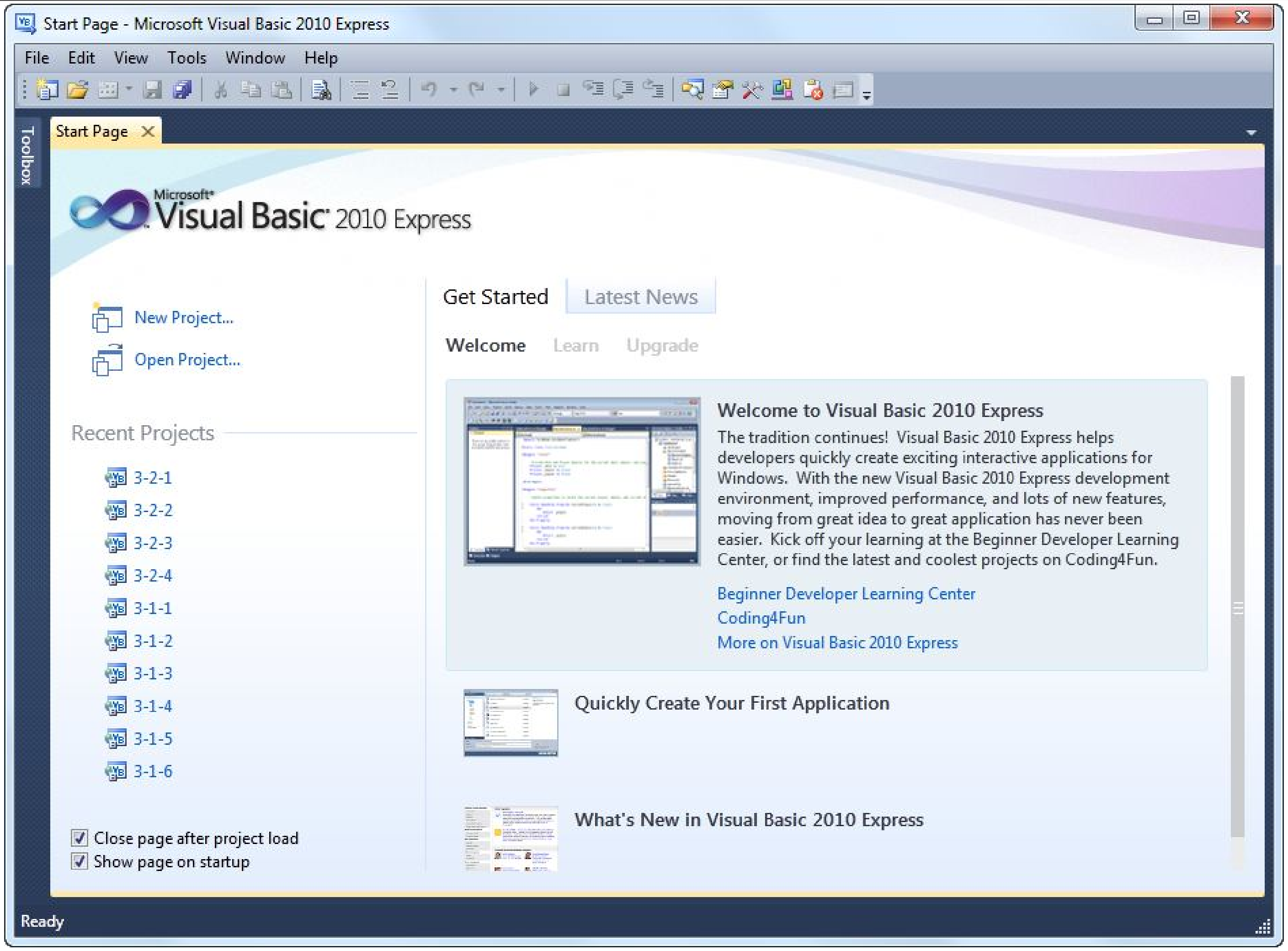
Different Versions of Visual Basic

- 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

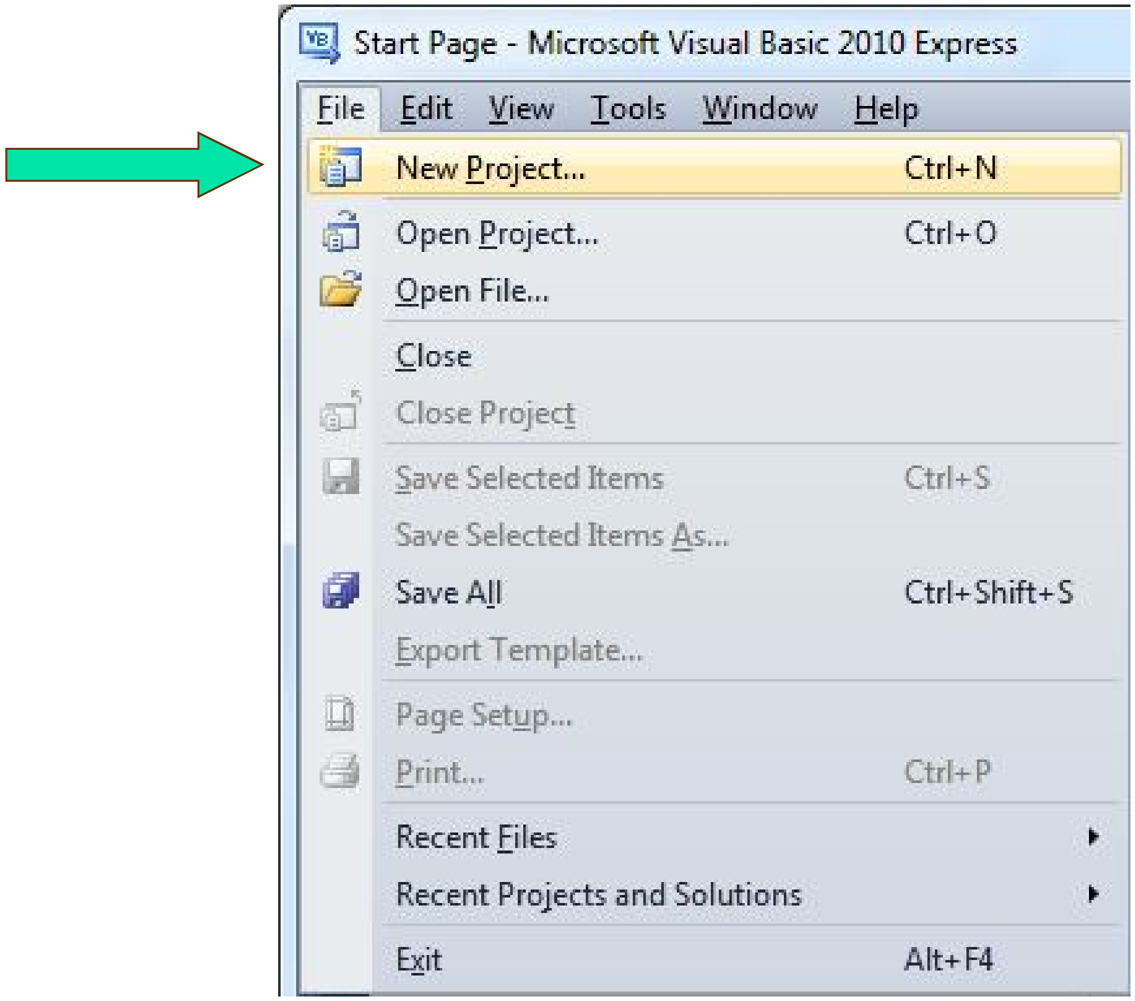
Visual Basic Start Page


Start Page - Microsoft Visual Basic 2010 Express
File Edit View Tools Window Help
Toolbox
Start Page
Microsoft Visual Basic 2010 Express
New Project...
Open Project...
Recent Projects
3-2-1
3-2-2
3-2-3
3-2-4
3-1-1
3-1-2
3-1-3
3-1-4
3-1-5
3-1-6
Close page after project load
Show page on startup
Get Started
Latest News
Welcome
Learn
Upgrade
Beginner Developer Learning Center
Coding4Fun
More on Visual Basic 2010 Express
Quickly Create Your First Application
What's New in Visual Basic 2010 Express
Ready

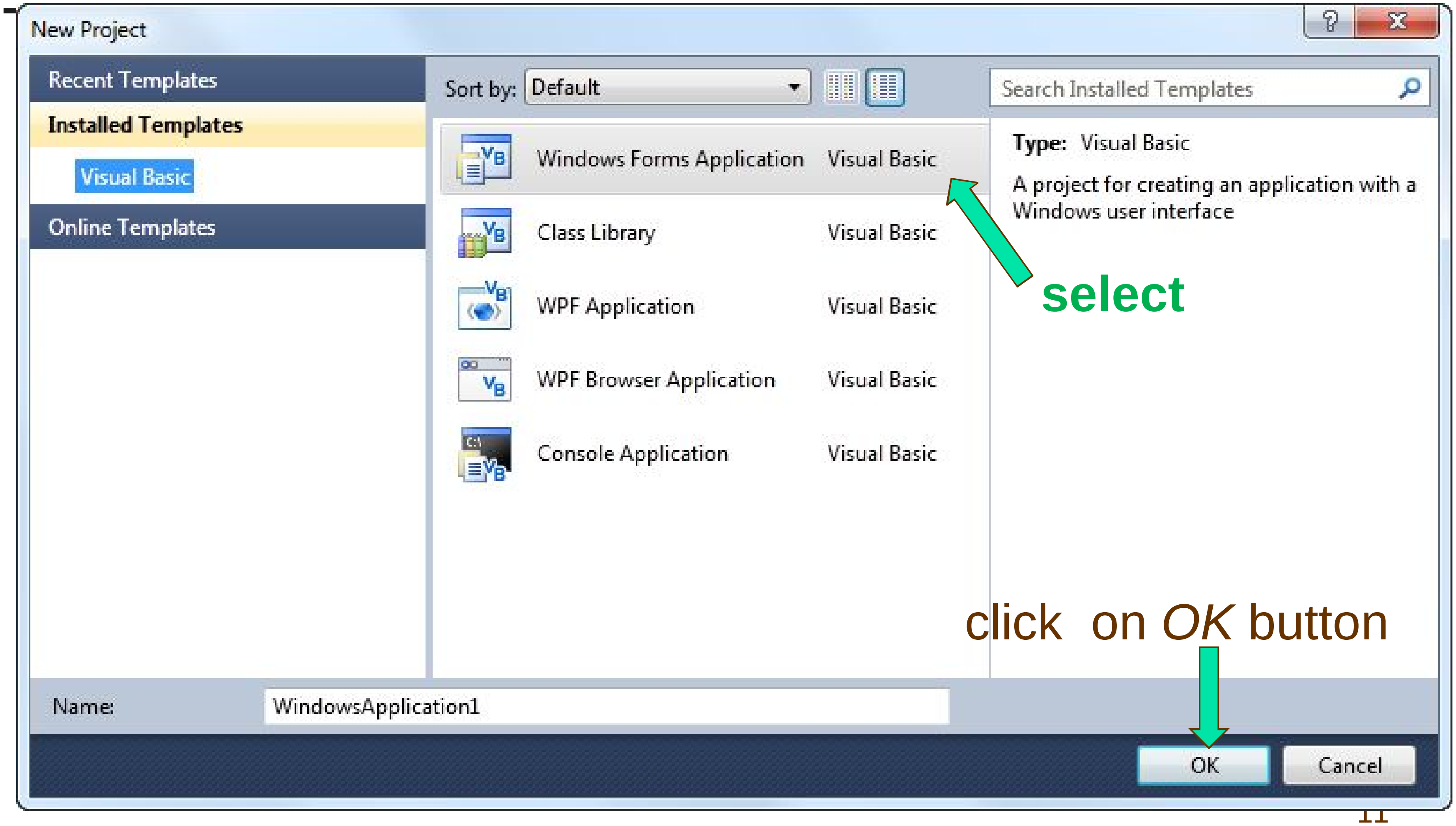
Welcome to Visual Basic 2010 Express

The tradition continues! Visual Basic 2010 Express helps developers quickly create exciting interactive applications for Windows. With the new Visual Basic 2010 Express development environment, improved performance, and lots of new features, moving from great idea to great application has never been easier. Kick off your learning at the Beginner Developer Learning Center, or find the latest and coolest projects on Coding4Fun.

Start a New Project



New Project Dialog Box


New Project

Recent Templates
Installed Templates
Visual Basic
Online Templates

Sort by: Default

Search Installed Templates

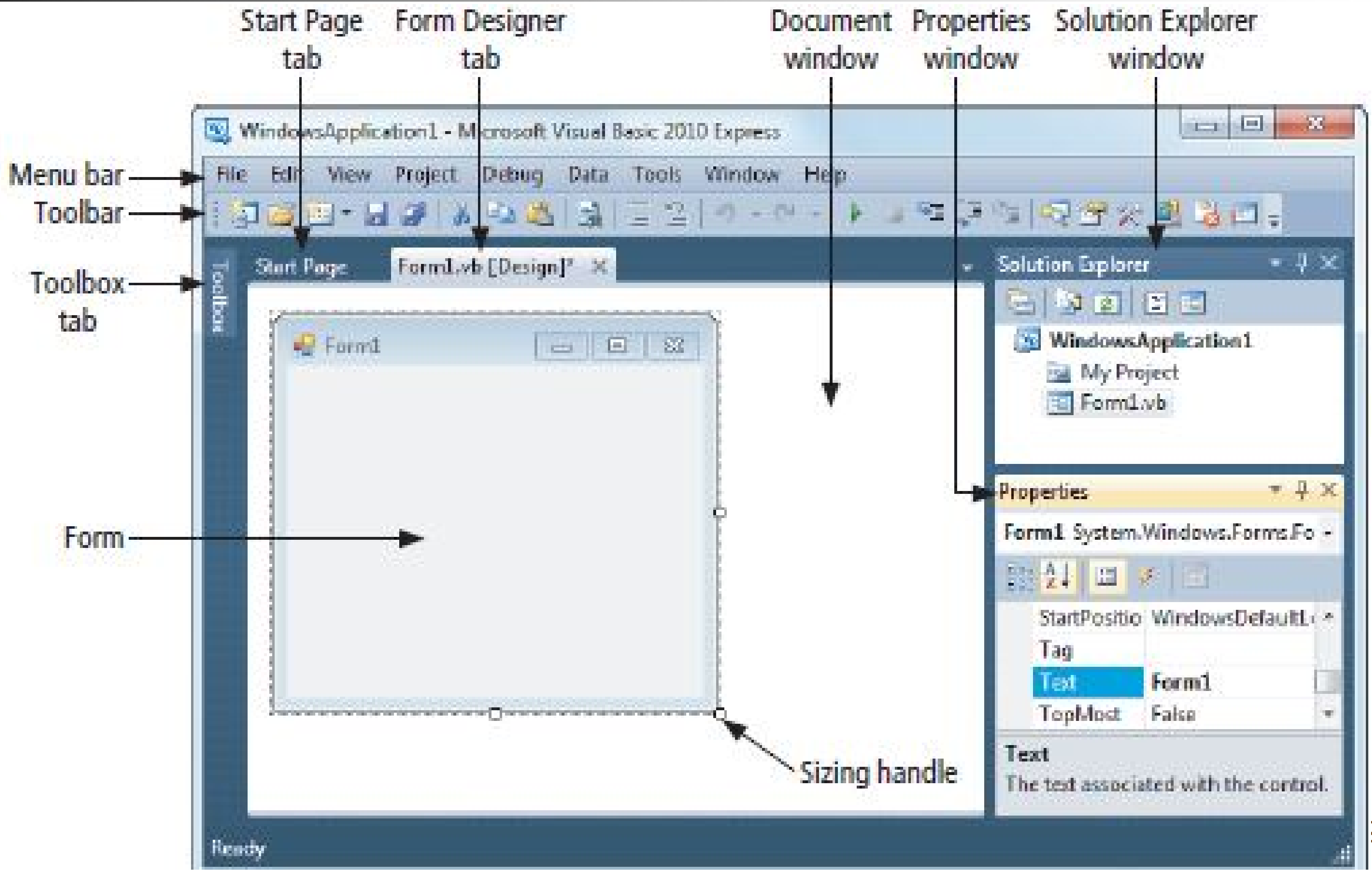
Windows Forms Application	Visual Basic
Class Library	Visual Basic
WPF Application	Visual Basic
WPF Browser Application	Visual Basic
Console Application	Visual Basic

Type: Visual Basic

A project for creating an application with a Windows user interface

Name: WindowsApplication1

Initial Visual Basic Screen

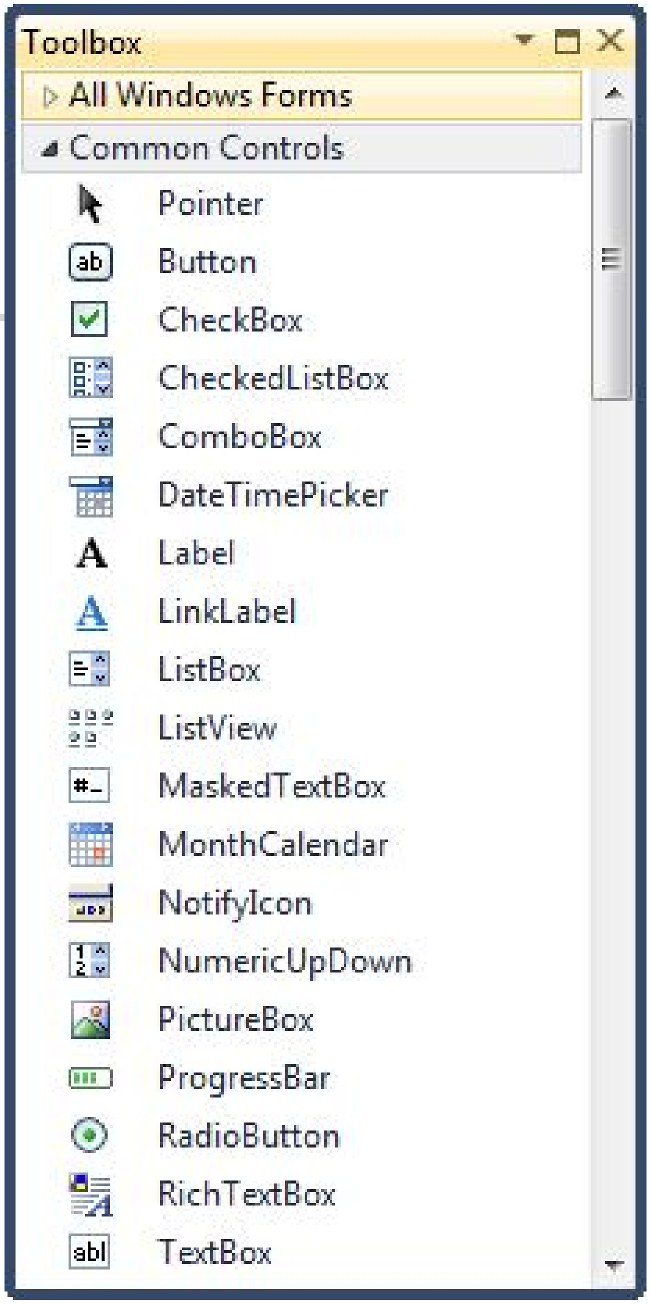


Form1 System.Windows.Forms.Fo

StartPositio	WindowsDefaultL
Tag	
Text	Form1
TopMost	False

Text
The text associated with the control.

Toolbox


Toolbox

All Windows Forms

Common Controls

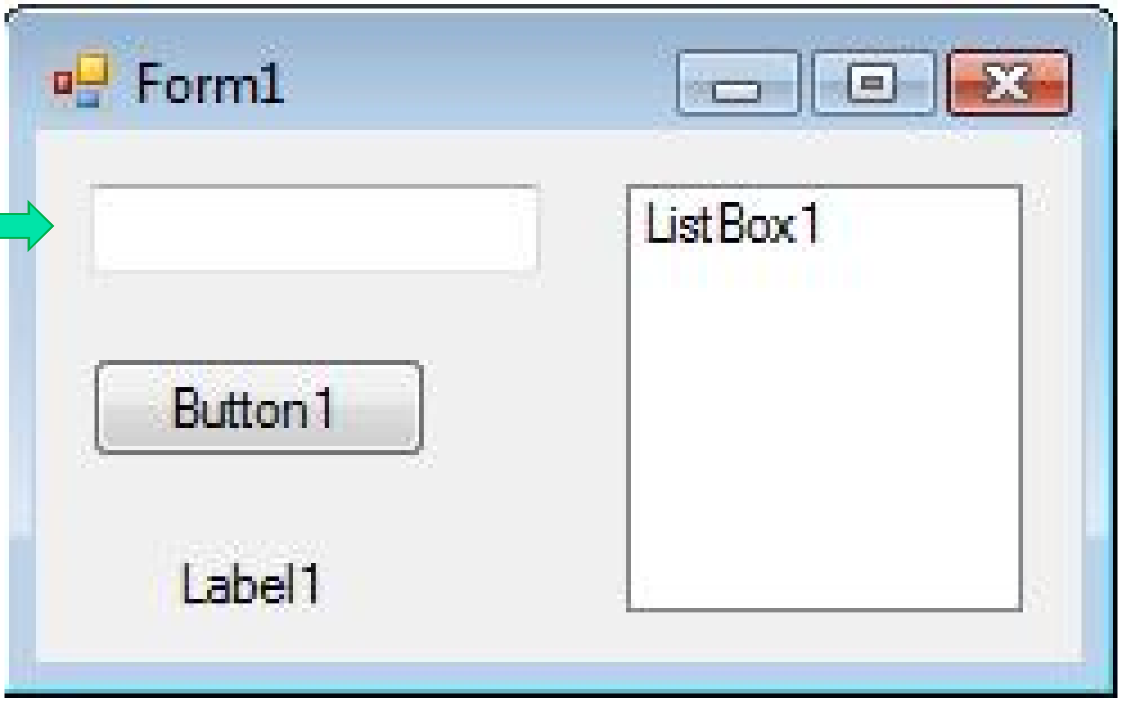
Pointer
Button
CheckBox
CheckedListBox
ComboBox
DateTimePicker
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MonthCalendar
NotifyIcon
NumericUpDown
PictureBox
ProgressBar
RadioButton
RichTextBox
TextBox

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

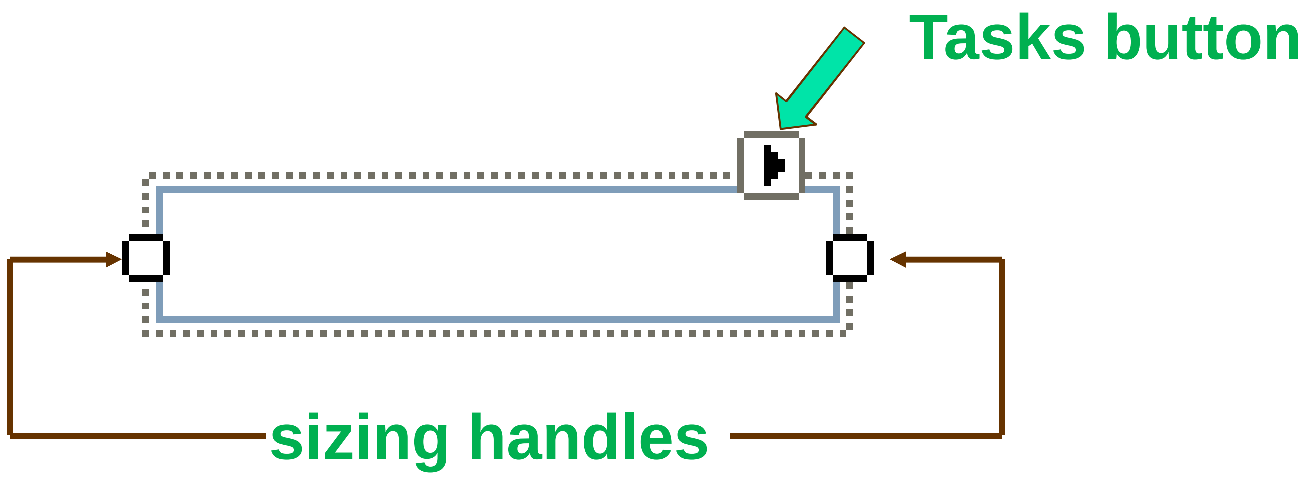
text box



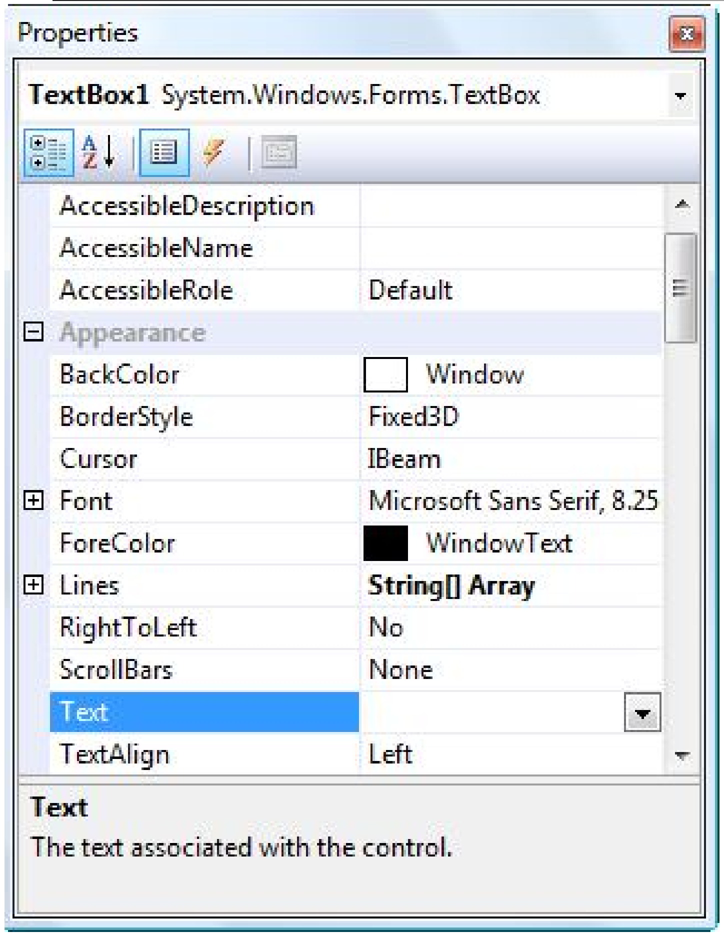
To select a control, click on it. Sizing handles will appear when a control is selected.

Text Box Control

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



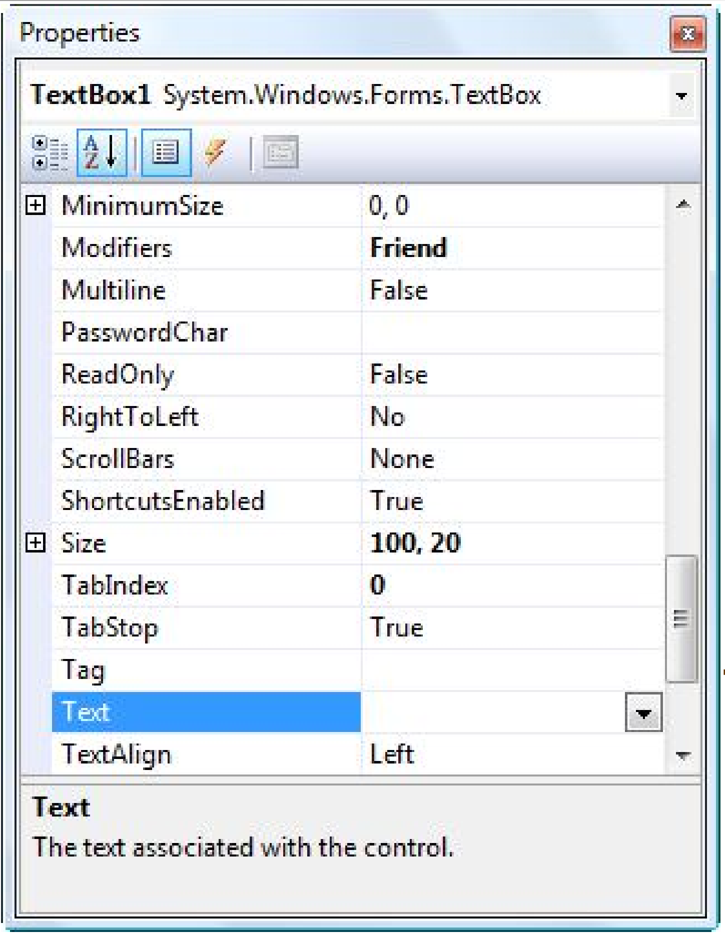
Properties Window



AccessibleDescription	
AccessibleName	
AccessibleRole	Default
⊟ Appearance	
BackColor	⬜ Window
BorderStyle	Fixed3D
Cursor	IBeam
⊞ Font	Microsoft Sans Serif, 8.25
ForeColor	⬛ WindowText
⊞ Lines	**String[] Array**
RightToLeft	No
ScrollBars	None
Text	
TextAlign	Left

Text
The text associated with the control.

categorized view



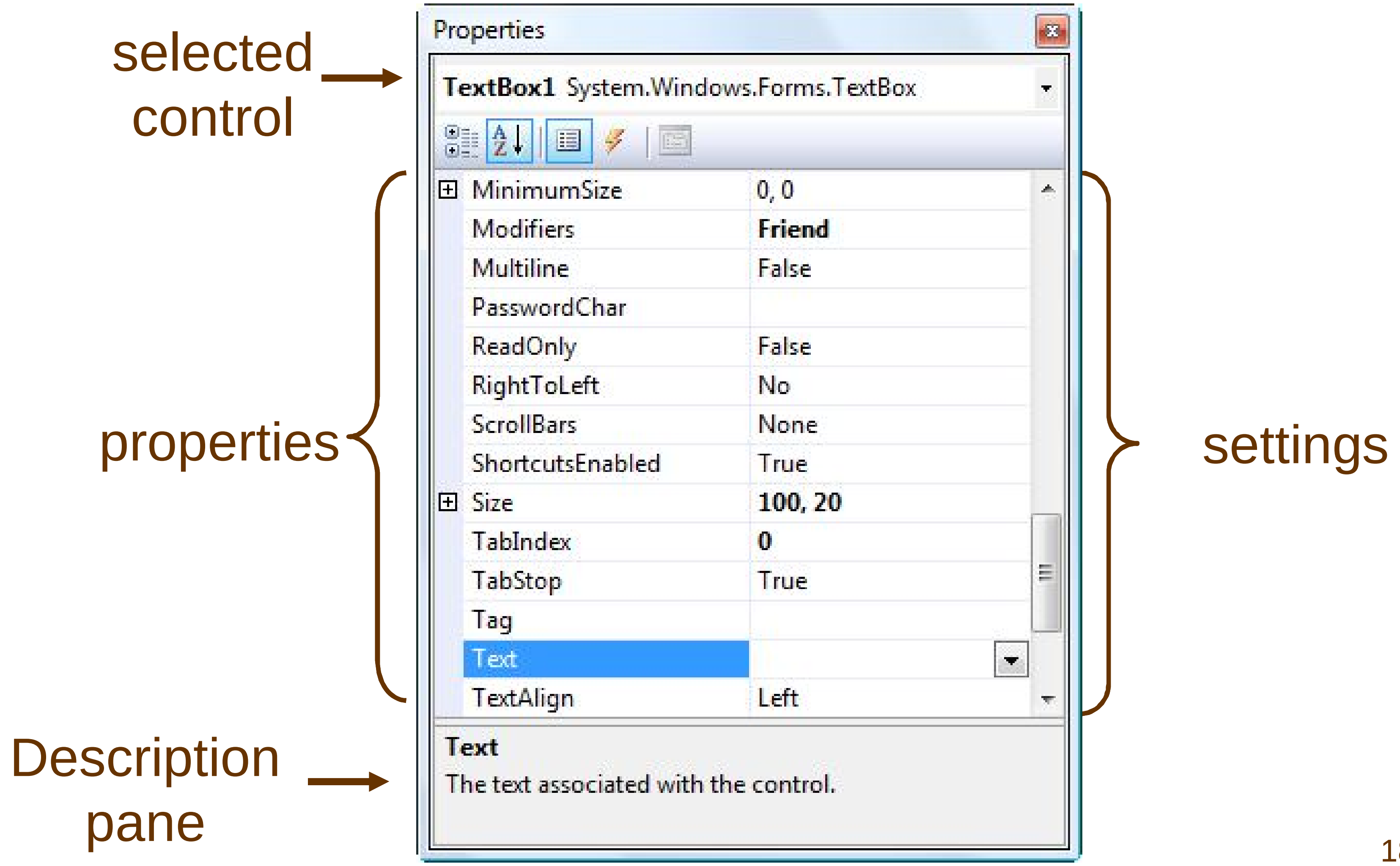
⊞ MinimumSize	0, 0
Modifiers	**Friend**
Multiline	False
PasswordChar	
ReadOnly	False
RightToLeft	No
ScrollBars	None
ShortcutsEnabled	True
⊞ Size	**100, 20**
TabIndex	**0**
TabStop	True
Tag	
Text	
TextAlign	Left

Text
The text associated with the control.

alphabetical view

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

Properties Window (continued)



TextBox1 System.Windows.Forms.TextBox

MinimumSize	0, 0
Modifiers	**Friend**
Multiline	False
PasswordChar	
ReadOnly	False
RightToLeft	No
ScrollBars	None
ShortcutsEnabled	True
Size	**100, 20**
TabIndex	**0**
TabStop	True
Tag	
Text	
TextAlign	Left

Text
The text associated with the control.

Some Often Used Properties

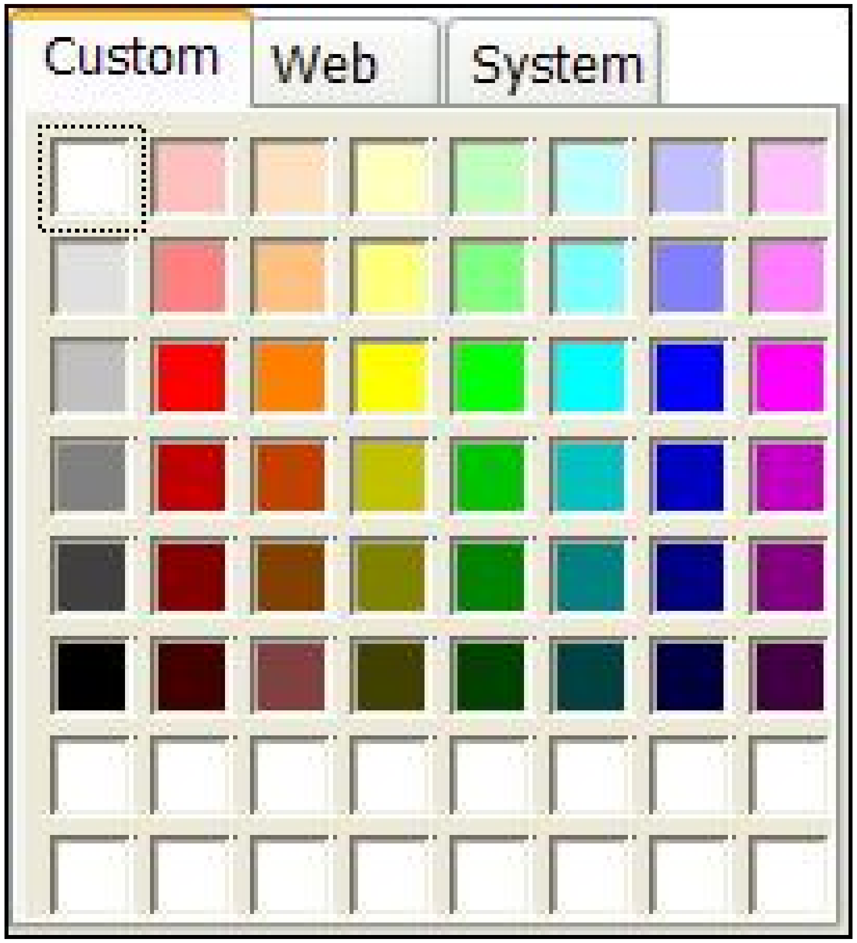
- Text
- Autosize
- Font.Name
- Font.Size
- ForeColor
- BackColor
- ReadOnly

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.

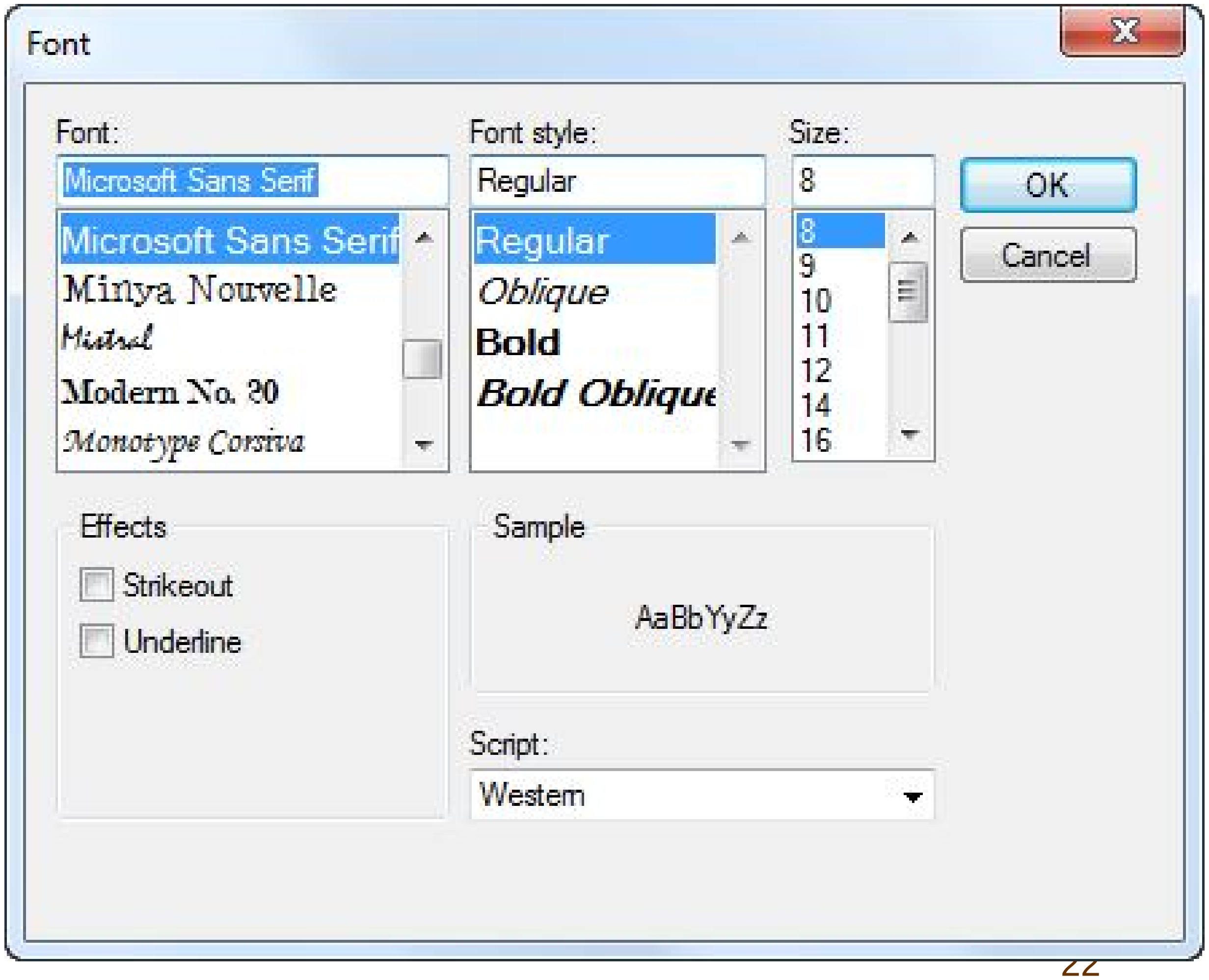
Setting the ForeColor Property

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

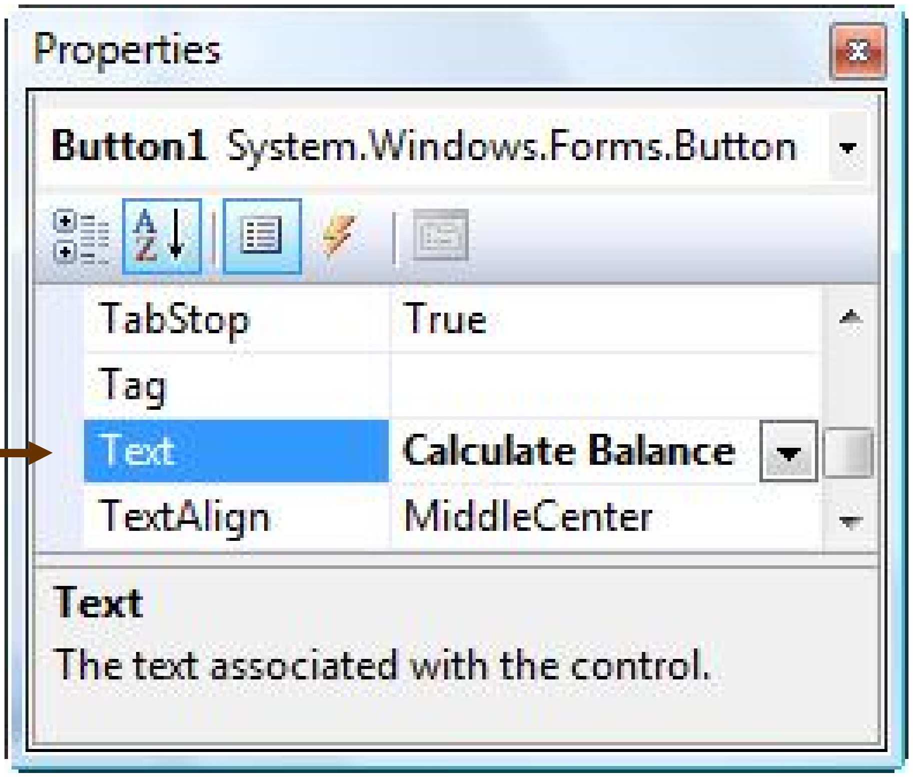
1. Click on Font in left column.
2. Click on ellipsis at right of settings box to obtain display shown.
3. Make selections.



Font: Microsoft Sans Serif
Font style: Regular
Size: 8
Script: Western

Button Control

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


Properties

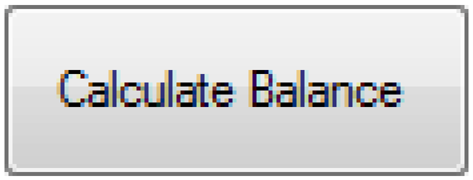
Button1 System.Windows.Forms.Button

TabStop	True
Tag	
Text	**Calculate Balance**
TextAlign	MiddleCenter

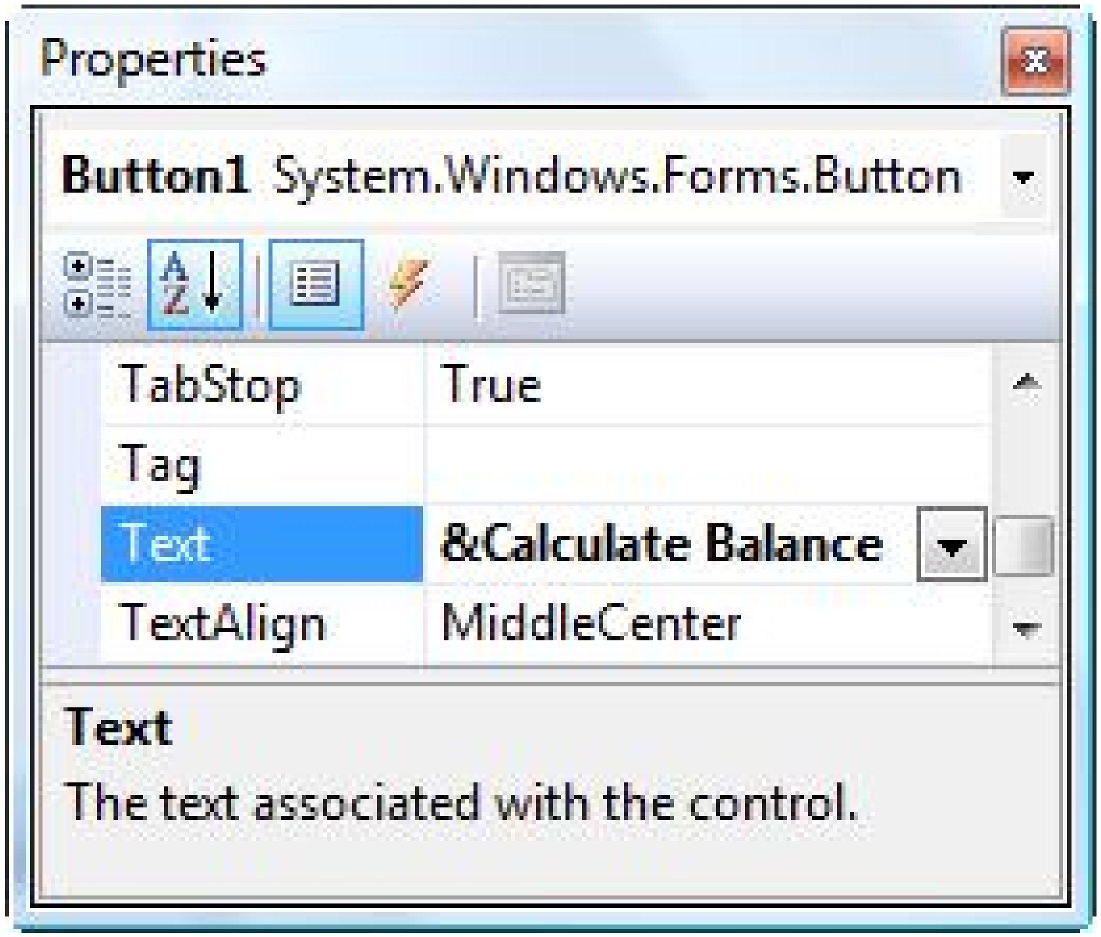
Text

The text associated with the control.

Text property



Add an Access Key



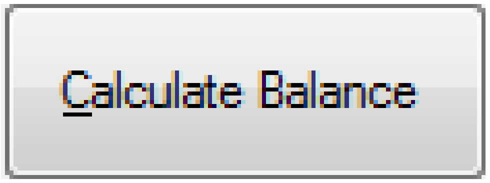
Properties

Button1 System.Windows.Forms.Button

TabStop	True
Tag	
Text	**&Calculate Balance**
TextAlign	MiddleCenter

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

Control	Prefix	Example
button	btn	btnCompute
label	lbl	lblAddress
text box	txt	txtAddress
list box	lst	lstOutput

Renaming the Form

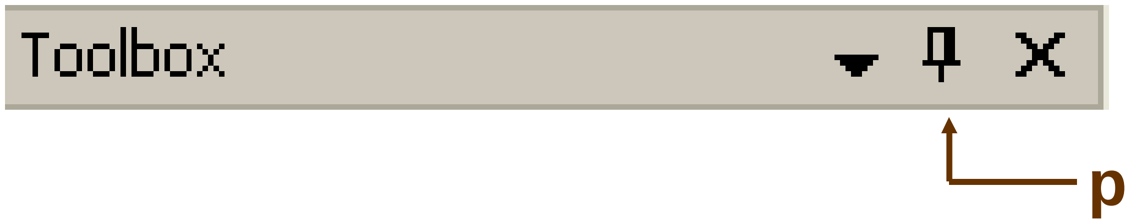
- Initial name is Form1
- The Solution Explorer window lists a file named Form1.vb.
- To rename the form, change the name of this file to *newName*.vb
- *newName* should begin with prefix *frm*.

Fonts

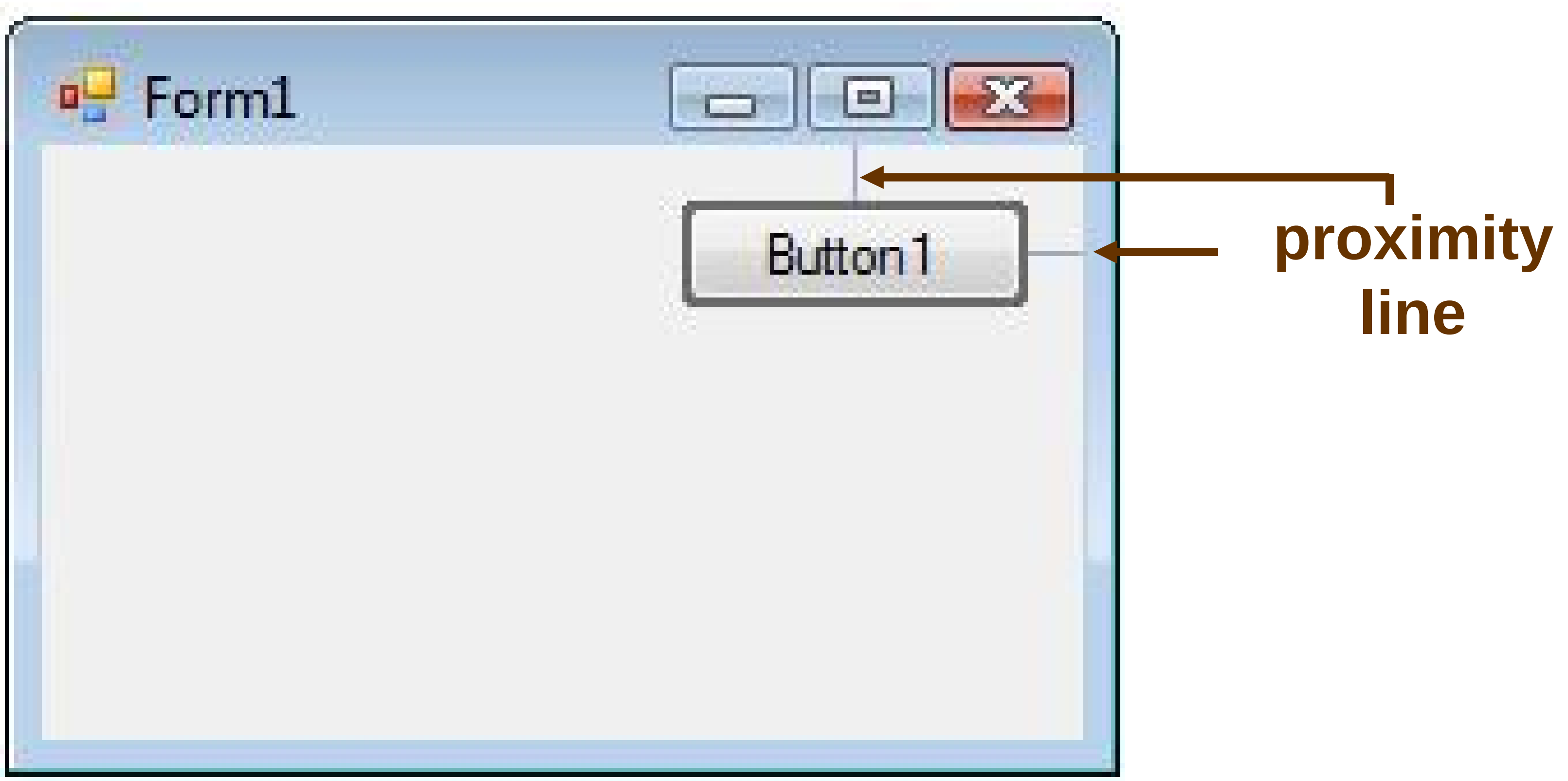
- Proportional width fonts, such as Microsoft Sans Serif, use less space for "I" than for "W"
- Fixed-width fonts take up the same amount of space for each character – like Courier New
- Fixed-width fonts are used for tables.

Auto Hide

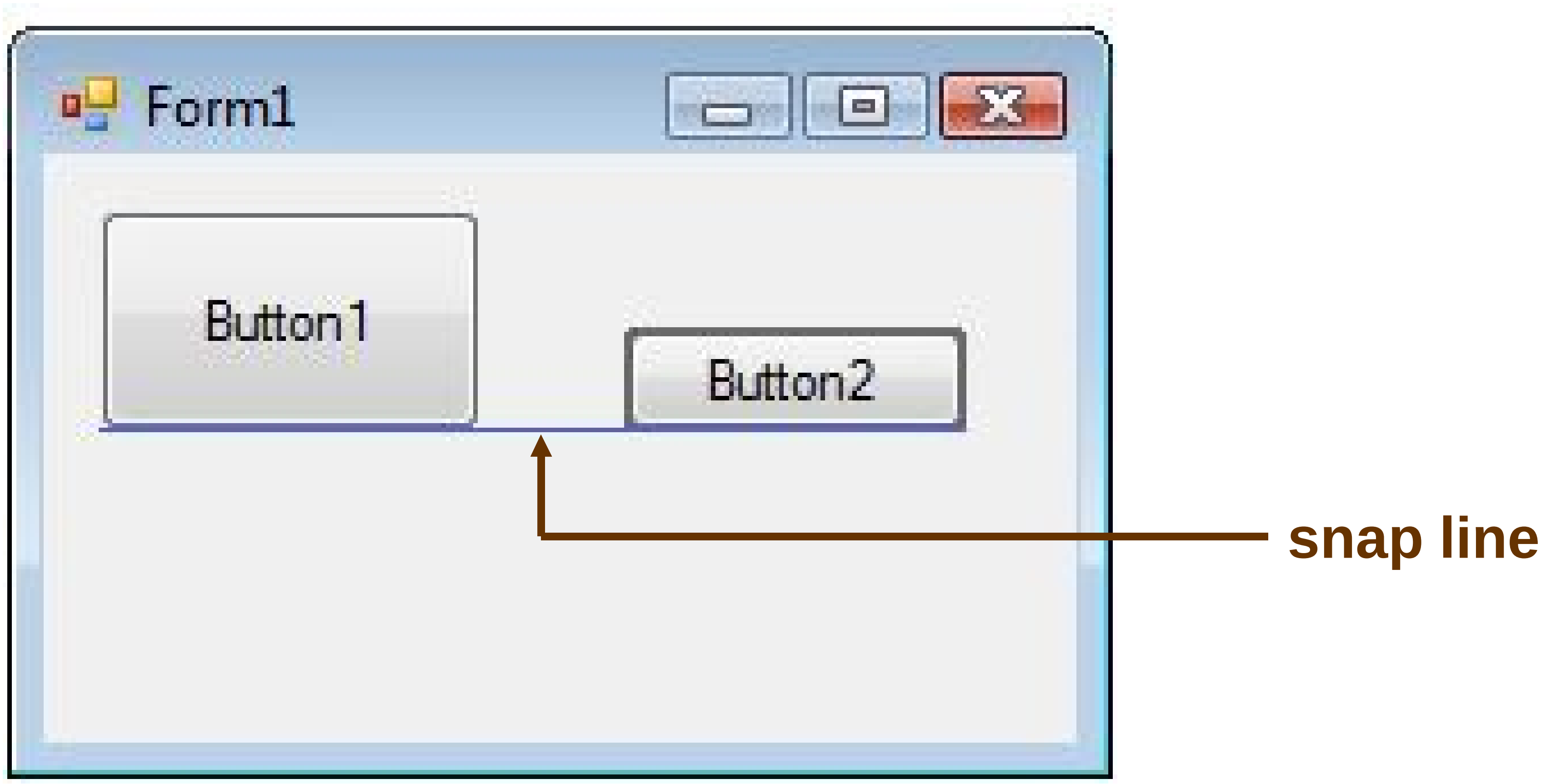
- Hides Toolbox when not in use
- Vertical push pin icon indicates auto hide is disabled.
- Click the push pin to make it horizontal and enable auto hide.



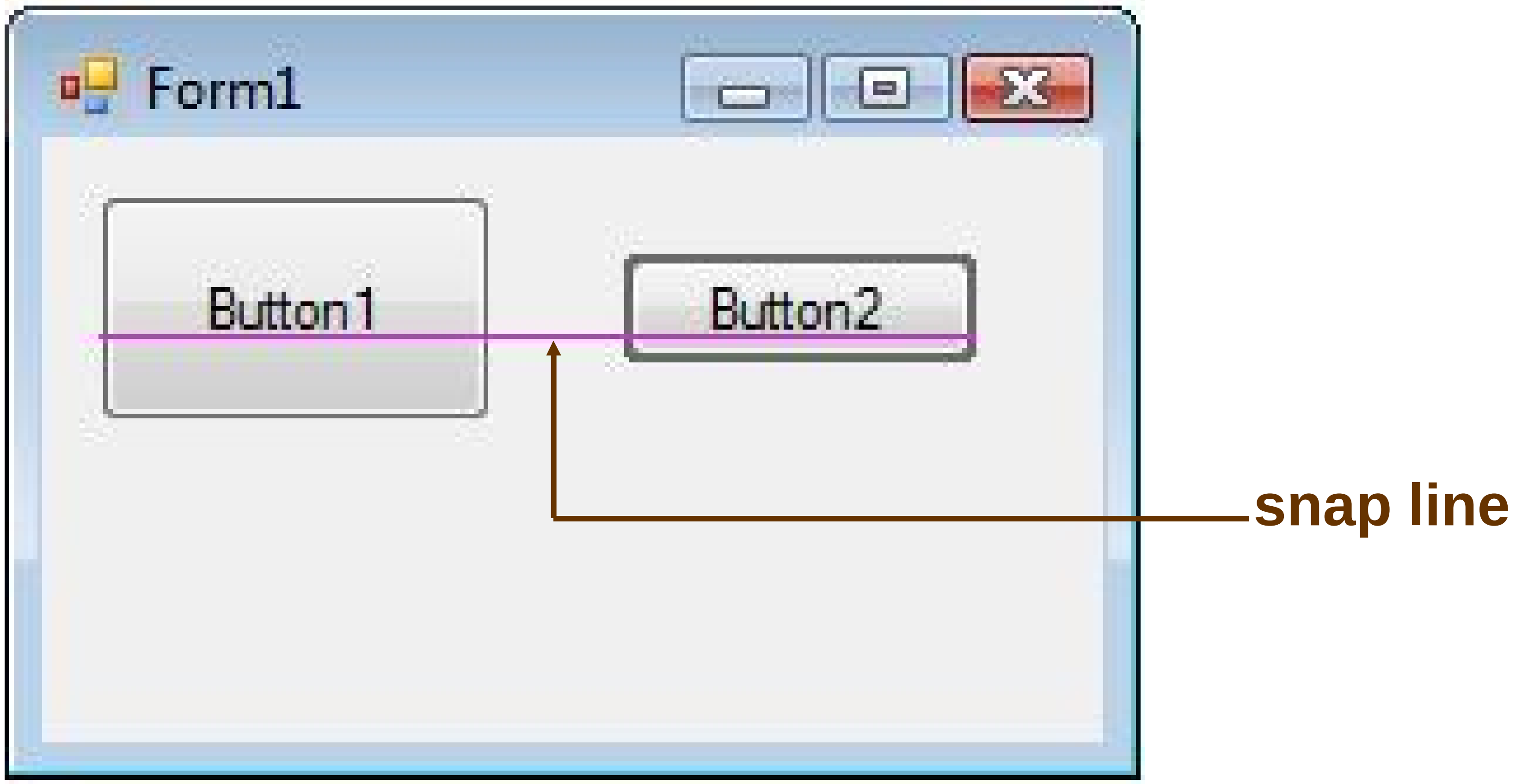
Positioning Controls



Aligning Bottoms of Controls



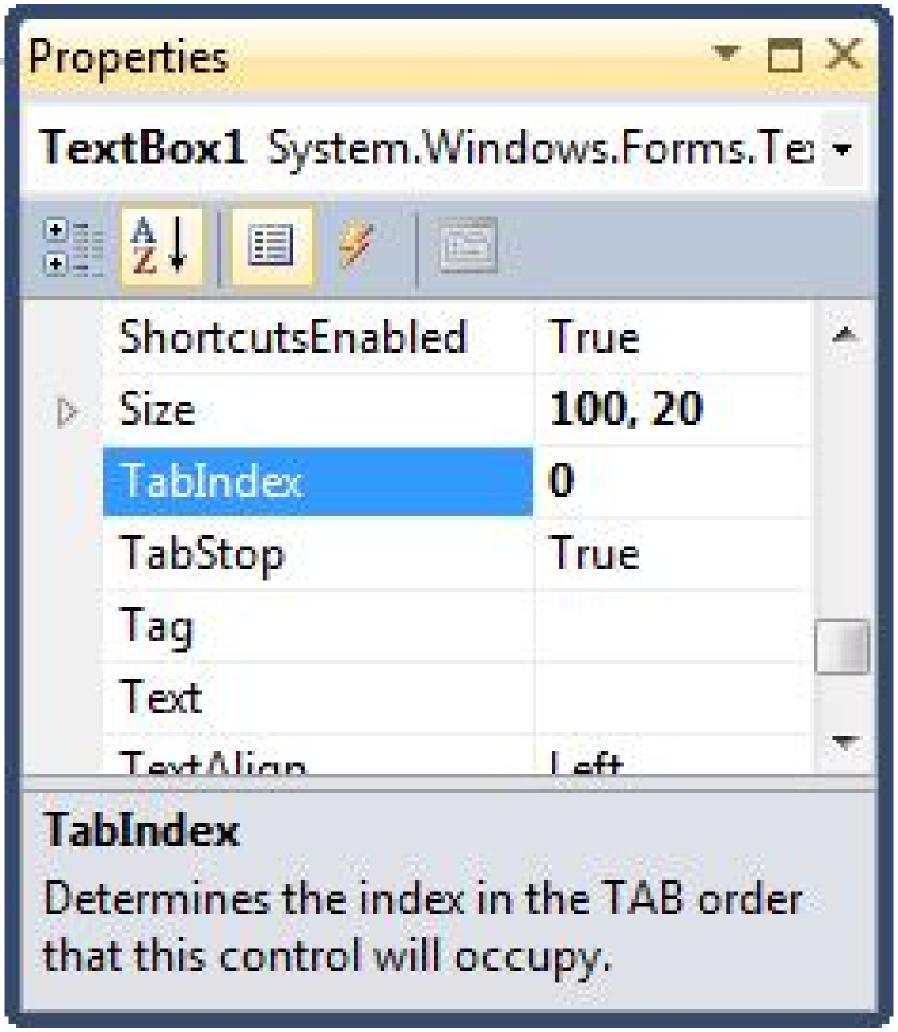
Aligning Middles of Controls



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.


Properties

TextBox1 System.Windows.Forms.Te

ShortcutsEnabled	True
Size	100, 20
TabIndex	0
TabStop	True
Tag	
Text	
TextAlign	Left

TabIndex

Determines the index in the TAB order that this control will occupy.

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.

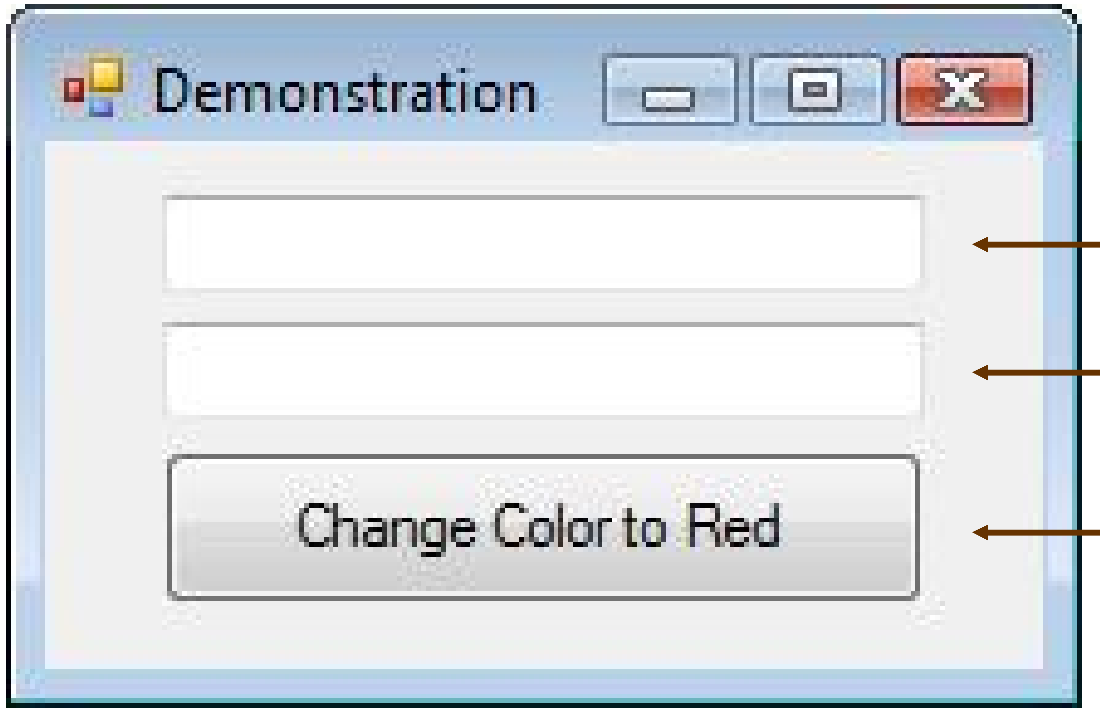
Sample Statements

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

General Form:

controlName.property = setting

Sample Form



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

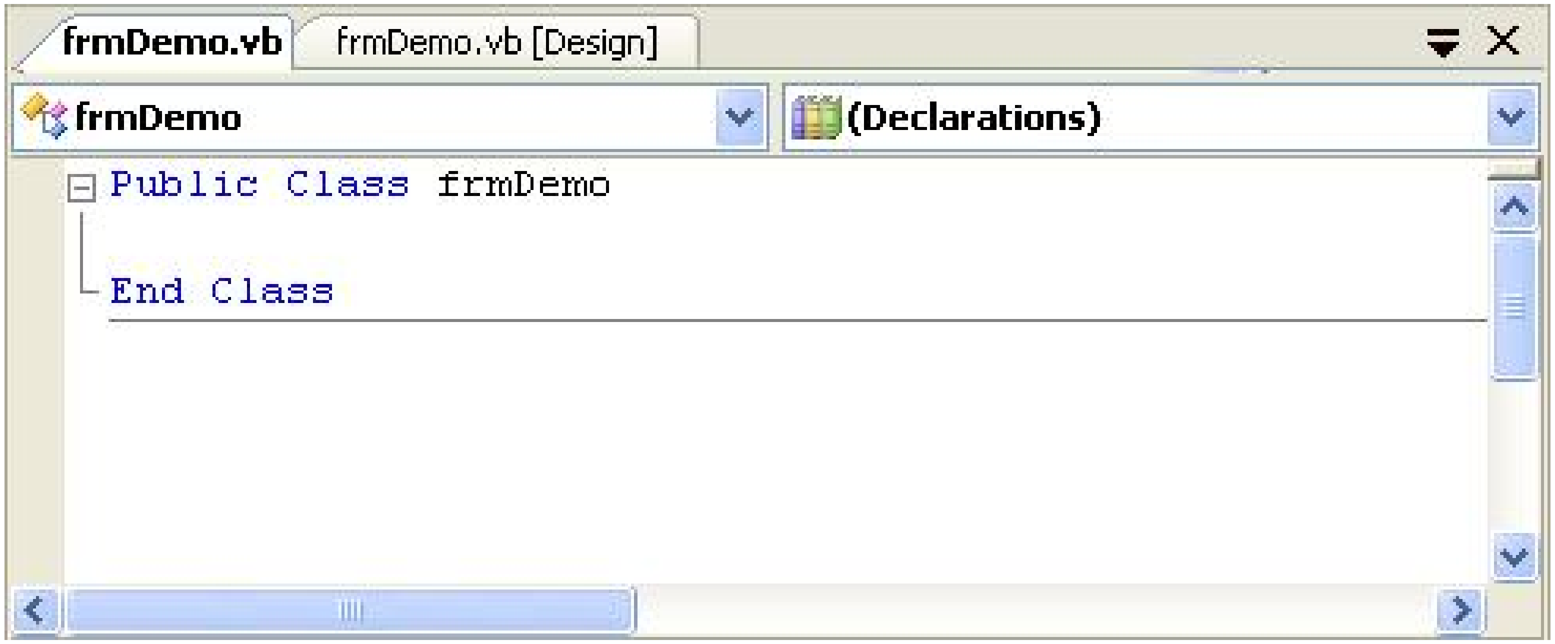
The Three Steps in Creating a Visual Basic Program

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

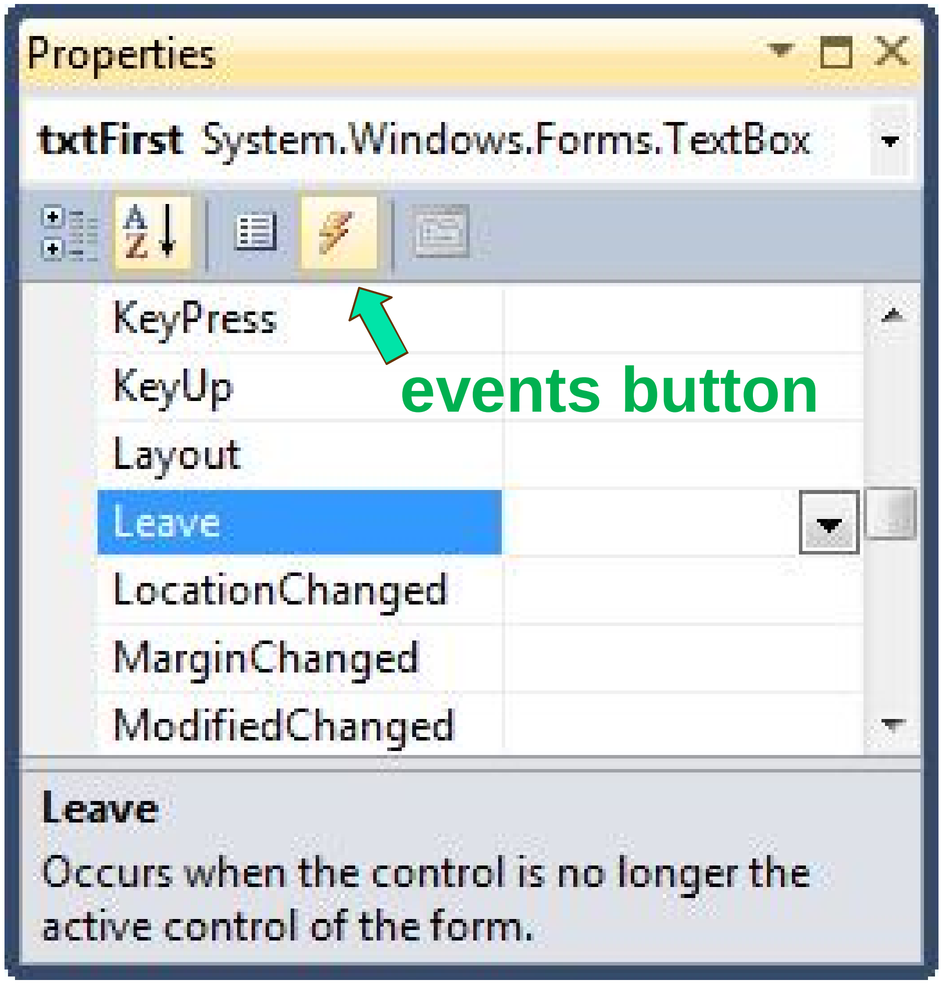


```
Public Class frmDemo

End Class
```

Display Events for a Control

- Select the control
- Click on the Events button (⚡) in the Properties window



Properties

txtFirst System.Windows.Forms.TextBox

KeyPress
KeyUp
Layout
Leave
LocationChanged
MarginChanged
ModifiedChanged

Leave
Occurs when the control is no longer the active control of the form.

Structure of an Event Procedure

```
header {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

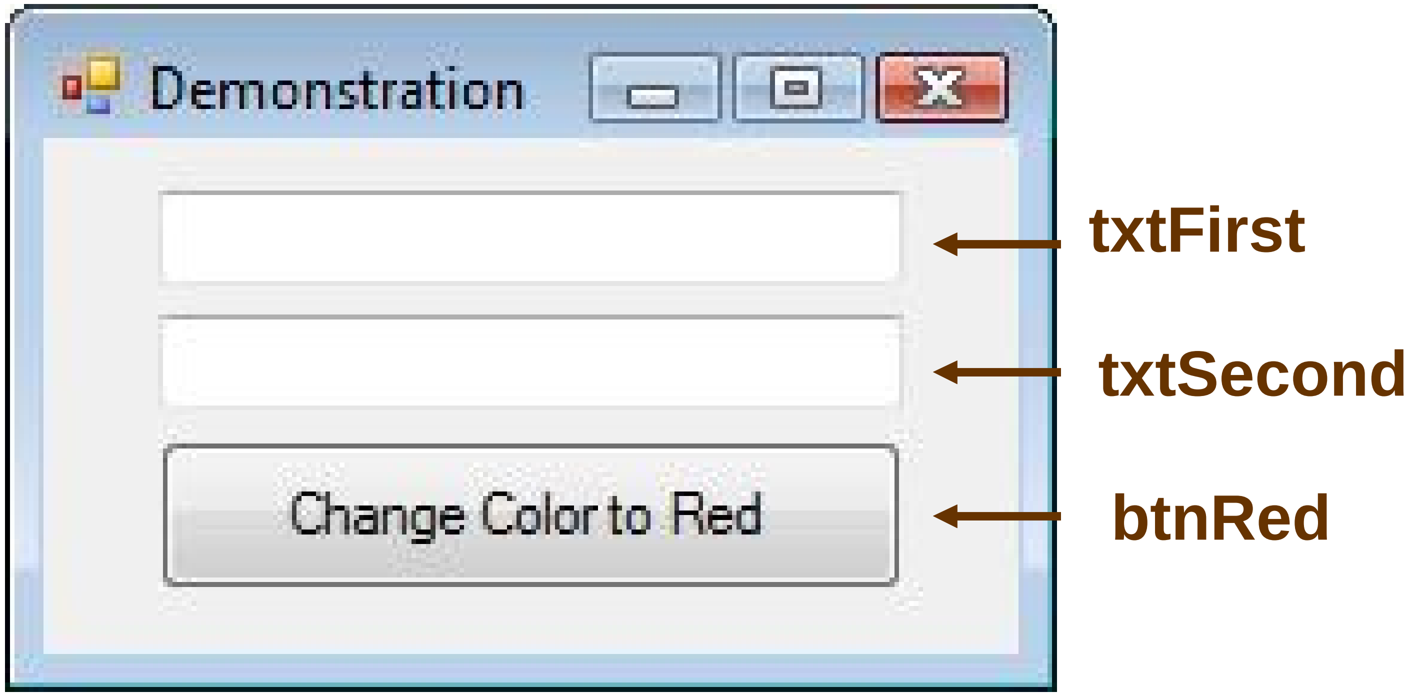
- Double-click on a control

  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.)

Sample Form



Double-click on txtFirst to create the outline for the Code Editor

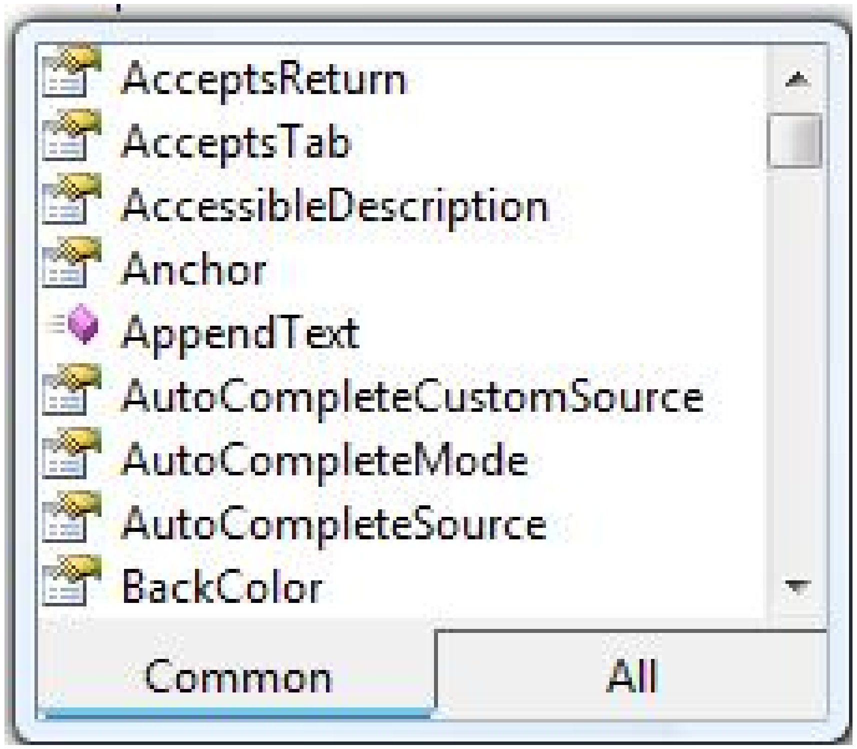
Code for Walkthrough

```
Public Class frmDemo
  Private Sub txtFirst_TextChanged(...)
           Handles txtFirst.TextChanged
    txtFirst.ForeColor = Color.Blue
  End Sub
End Class
```

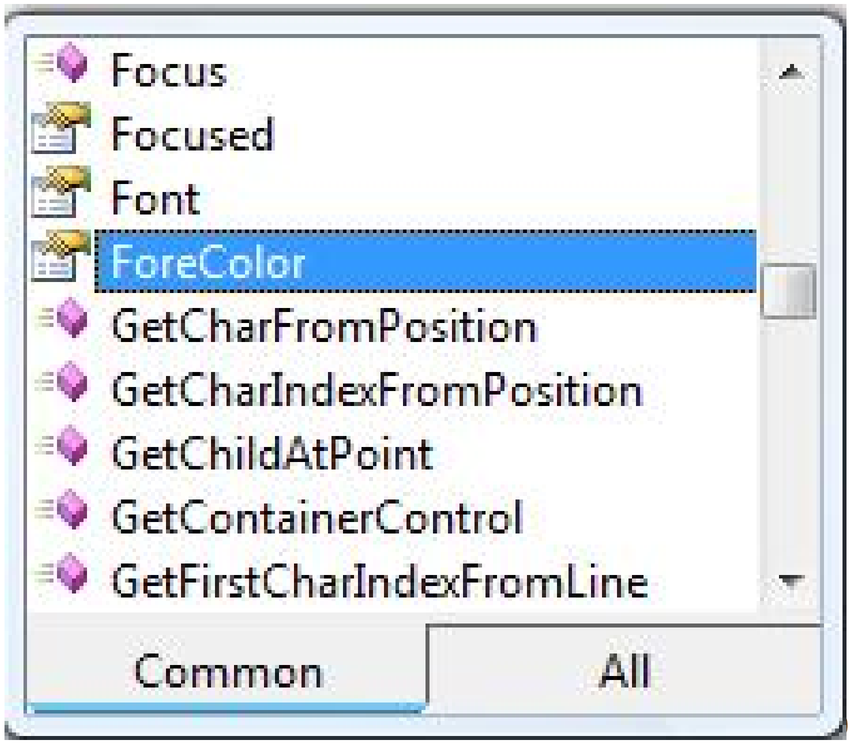
IntelliSense

Automatically pops up to help the programmer.

```
txtFirst.
```



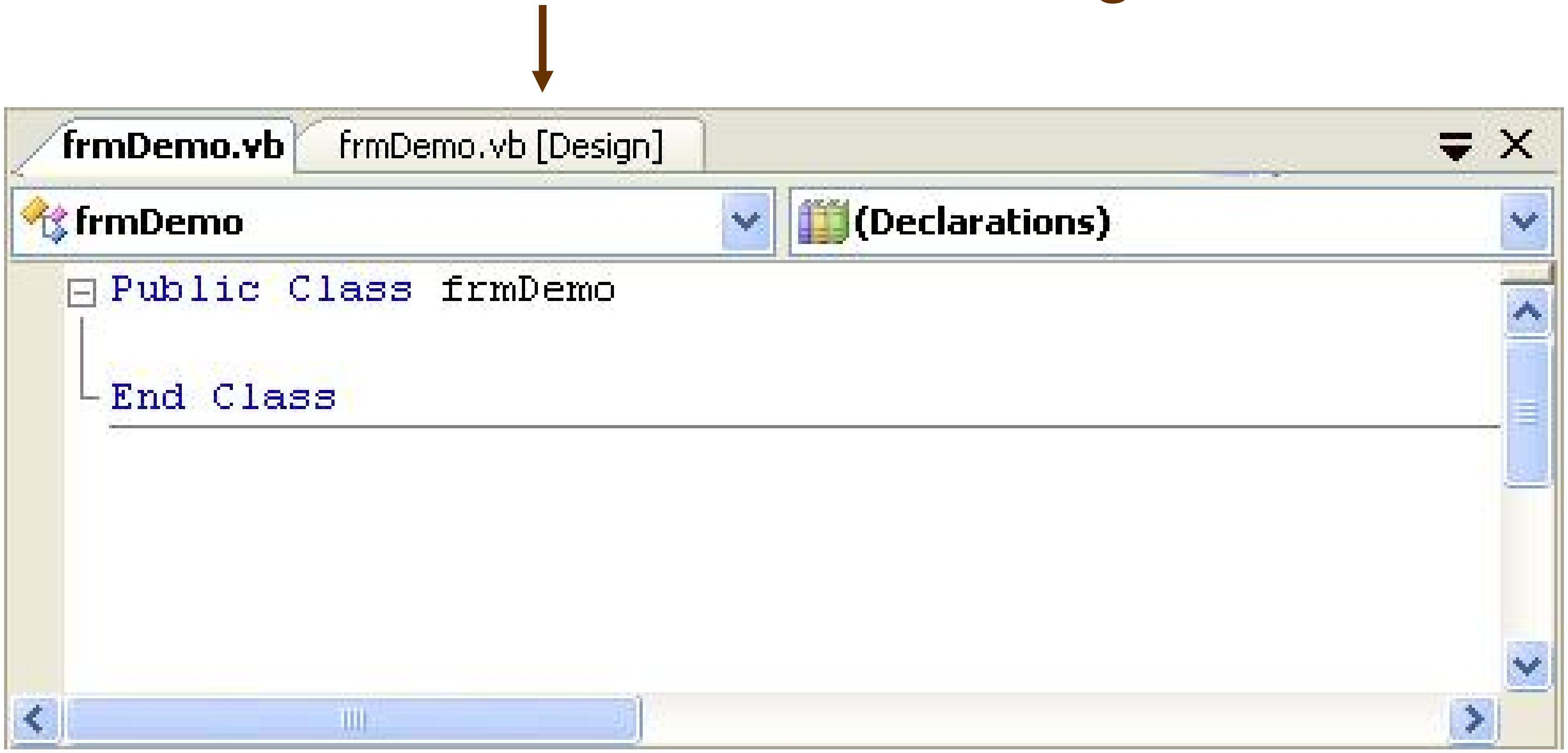
AcceptsReturn
AcceptsTab
AccessibleDescription
Anchor
AppendText
AutoCompleteCustomSource
AutoCompleteMode
AutoCompleteSource
BackColor



Focus
Focused
Font
ForeColor
GetCharFromPosition
GetCharIndexFromPosition
GetChildAtPoint
GetContainerControl
GetFirstCharIndexFromLine

Code Editor

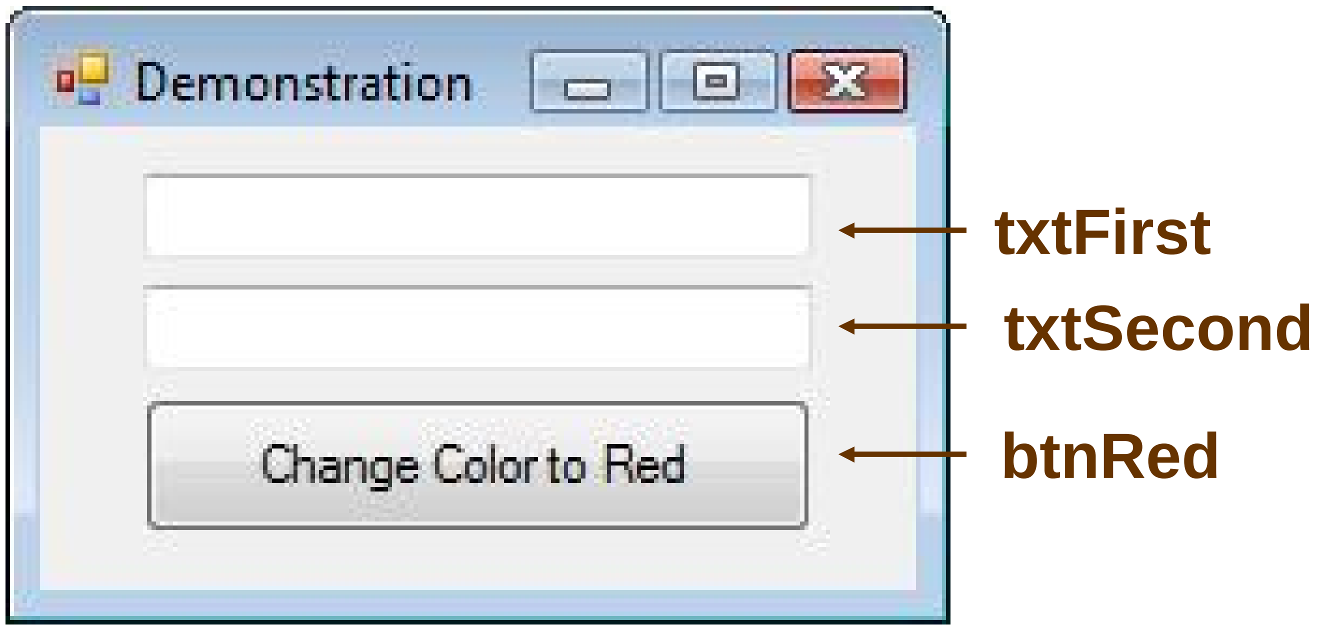
click tab to return to Form Designer



```
Public Class frmDemo

End Class
```

Sample Form



Double-click on btnRed to return to Code Editor and add the outline of an event procedure

Code for Walkthrough

```
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

Code for Walkthrough

```
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

$$\texttt{Private Sub }\underbrace{\texttt{btnRed\_Click}}_{\text{Name, can be changed.}}\texttt{(…) Handles }\underbrace{\texttt{btnRed.Click}}_{\text{Identifies event}}$$

```
Private Sub Button_Press(…) Handles btnRed.Click
```

Handling Multiple Events

An event procedure can be invoked by two events.

```
Private Sub Happening(...)
       Handles btnRed.Click,txtSecond.Leave
  txtFirst.ForeColor = Color.Red
End Sub
```

Altering Properties of the Form

- The following won't work:

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

- The form is referred to by the keyword *Me*.

```
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.