Introduction

- Develop your first application:
  - Display a map and written directions to the Highlander Hotel
  - Use a form with Labels
  - Use a PictureBox control
  - Use Buttons
  - Write an event procedure

Step 1. Define What Program Is To Do

- Purpose: Display a map to the Highlander Hotel
- Input: None
- Process: Display a form
- Output: Display a graphic image showing a map on the form

Step 2. Visualize and Design the User Interface

- Below is a sketch of the form to be shown

In This Section You Create Your First Visual Basic Application: a Window That Displays a Map and Road Directions to a Hotel
Step 3. List the Controls Needed

<table>
<thead>
<tr>
<th>Control Type</th>
<th>Control Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form</td>
<td>frmDirections</td>
<td>A small form that will serve as the window onto which the other controls will be placed</td>
</tr>
<tr>
<td>Label</td>
<td>lblHotelMap</td>
<td>Displays the message &quot;Directions to the Highlander Hotel&quot;</td>
</tr>
<tr>
<td>PictureBox</td>
<td>pbxHotelMap</td>
<td>Displays the graphic image showing the map to the hotel</td>
</tr>
</tbody>
</table>

Step 4. Define Values for each Control’s Relevant Property

- **Form**
  - Name: frmDirections
  - Text: "Directions"

- **Label**
  - Name: lblHotelMap
  - Text: "Directions to the Highlander Hotel"
  - TextAlign: MiddleCenter
  - Font: Microsoft sans serif, bold, 18 point

- **PictureBox**
  - Name: pbxHotelMap
  - Picture: HotelMap.jpg
  -SizeMode: StretchImage

Skipping Steps in Programming Process

- This demo is skipping:
  - Step 5. List methods needed for each control
  - Step 6. Create pseudocode or a flowchart for each method
  - Step 7. Check the code for errors

Step 8. Use VB to Create the Application

- **Tutorial 2-1:**
  - Establish the Form (frmDirections) and set its Text property

- **Tutorial 2-2, 2-3, 2-4:**
  - Add a Label control (lblHotelMap)
    - Position and resize it on the form
    - Set Text, TextAlign, and Font properties

- **Tutorial 2-6:**
  - Add a PictureBox control (pbxHotelMap)
    - Position and resize it on the form
    - Set Image property to display HotelMap.jpg

Steps 10 and 11

- Step 10. Attempt to run application – find syntax errors
- Step 11. Run the application using test data as input

- **Tutorial 2-7:**
  - Run the application
  - Close and save the application
Project Organization on Disk
- User creates a new project in Visual Studio
  - A solution and a folder are created at the same time with the same name as the project
  - The project belongs to the solution
    - Multiple projects can be included in a solution
  - The folder stores files related to the project including:
    - A solution file (.sln)
    - A project file (.vbproj)

Opening an Existing Project
- Use Recent Projects list on Start Page
  - Provided it hasn’t been moved or deleted
- Use Open Project button on Start Page
  - Then browse using Open Project dialog box
- Use Open Project option on File menu
  - Then browse using Open Project dialog box

Properties Window
- Used to view and modify the property values of a given object
- Two views of the properties are available:
  - Alphabetic (across all properties)
  - Categorized (groups properties by logical use)

Augment the Hotel Application
- Now the hotel owner wants to add an option to view written directions
  - Step 2. Visualize application

2.2 Responding to Events
An Application Responds to Events, Such As Mouse Clicks and Keyboard Input, by Executing Code Known As Event Procedures
Write the Event Procedures for the Directions Application

Step 3. Controls to be Added

<table>
<thead>
<tr>
<th>Control Type</th>
<th>Control Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
<td>lblDirections</td>
<td>Displays written directions to the hotel</td>
</tr>
<tr>
<td>Button</td>
<td>btnDisplayDirections</td>
<td>When clicked, causes lblDisplayDirections text to appear on the form</td>
</tr>
<tr>
<td>Button</td>
<td>btnExit</td>
<td>Stops the application when clicked</td>
</tr>
</tbody>
</table>
Step 4. Control Properties

- **Label:**
  - Name: lblDirections
  - Text: "Traveling on I-89, take..." (see page 59)
  - Visible: False

- **Button:**
  - Name: btnDisplayDirections
  - Text: "Display Directions"

- **Button:**
  - Name: btnExit
  - Text: "Exit"

Step 5. List methods needed for each control

- **btnDisplayDirections**
  - Click event procedure

- **btnExit**
  - Click event procedure

Step 6. Create pseudocode or a flowchart for each method

- **btnDisplayDirections** click event procedure
  - Pseudocode
    - Set Visible property for lblDirections control to True.
  - Flowchart

- **Step 7. Check the code for errors.**

Step 7. Check the code for errors.

Step 8. Add these Controls to the Form

- **Label:**
  - Name: lblDirections
  - Text: "Traveling on I-89, take..." (see page 59)
  - Visible: False

- **Button:**
  - Name: btnDisplayDirections
  - Text: "Display Directions"

- **Button:**
  - Name: btnExit
  - Text: "Exit"

Step 9. Use VB to write code for each event procedure

- **btnDisplayDirections** click event procedure
  - Pseudocode
    - Set Visible property for lblDirections control to True.
  - Flowchart

  To add the click event procedure:
  - Double-click on btnDisplayDirections while in form design view (slide 2-26)
  - Enter pseudo-code as a comment (slide 2-26)
  - Enter VB statement(s) for each pseudo-code comment (slides 2-27 thru 2-29)
Method btnDisplayDirections_Click

Private Sub btnDisplayDirections_Click _
    ByVal sender As System.Object, _
    ByVal e As System.EventArgs _
Handles btnDisplayDirections.Click
End Sub

Method btnDisplayDirections_Click

Private Sub btnDisplayDirections_Click _
    ByVal sender As System.Object, _
    ByVal e As System.EventArgs _
Handles btnDisplayDirections.Click
    'Set Visible property for lblDirections control to True
End Sub

Syntax: Referring to a Control's Property

- Specify the control name (lblDirections)
- Then a dot
- Then the PropertyName (Visible)
- Example:
  - lblDirections. Visible
  - Refers to Visible property of lblDirections control
  - Visible property values may only be True or False

Syntax: Assignment Statement

- Specify the item to receive the value
- Then the equal symbol
- Then the value to be assigned
- Example:
  - lblDirections. Visible = True
  - Assigns value True to Visible property of lblDirections control
  - Causes text of lblDirections control to become visible

Method btnDisplayDirections_Click

Private Sub btnDisplayDirections_Click _
    ByVal sender As System.Object, _
    ByVal e As System.EventArgs _
Handles btnDisplayDirections.Click
    'Set Visible property for lblDirections control to True
    Me. lblDirections. Visible = True
End Sub

Step 9. Use VB to write code for each event procedure

-.btnExit click event procedure
  - Pseudocode
    - Close the Form.
  - Flowchart
    - Close the Form
  - To add the click event procedure:
    - Double-click on the btnExit while in form design view
    - Enter pseudo-code as a comment
    - Enter VB statement(s) for each pseudo-code comment
Method btnExit_Click

Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnExit.Click
    ' Close the Form
    Me.Close()
End Sub

Additional Properties

- Control Color properties:
  - BackColor
  - Sets the background (fill) color
  - ForeColor
  - Sets the foreground (text) color
- Form Border style properties:
  - Sizable:
    - (Default) Has min, max, and close buttons; can be resized
  - Fixed3D:
    - Has a 3D look; min, max, and close buttons; cannot be resized
  - FixedSingle:
    - Has single line border; min, max, and close buttons; cannot be resized

Modifying the Text Property With Code

2.3 Quite Often, You Will Need to Change a Control’s Text Property With Code
This Is Done With an Assignment Statement

Modifying the Text Property in Code

Private Sub btnFeet_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnFeet.Click
    ' Display the conversion to feet. Me.lblMessage.Text = "1 Kilometer = 3,281 feet"
End Sub

The AutoSize, BorderStyle, and TextAlign Properties

2.4 The Label Control’s AutoSize Property Allows a Label to Change Size Automatically to Accommodate the Amount of Text in its Text Property

The BorderStyle Property Allows You to Set a Border Around a Label Control
AutoSize Property for Labels

- AutoSize is a Boolean (either True or False) Property of labels
- False (the default) means the box size will not change, regardless of the amount of text assigned to it
- True means the box will automatically resize itself to fit the amount of text assigned to it

BorderStyle Property for Labels

- BorderStyle determines the look of the box
  - None (the default) means no border
  - FixedSingle
    - Results in a border one pixel wide
  - Fixed3D
    - Gives the border a recessed 3-dimensional look

TextAlign Property for Labels

- The value of TextAlign establishes the alignment (or justification) of the text:
  - TopLeft
  - TopCenter
  - TopRight
  - MiddleLeft
  - MiddleCenter
  - MiddleRight
  - BottomLeft
  - BottomCenter
  - BottomRight
- The assignment statement below forces the text of lblTitle to appear in the middle center of the label
  
  lblTitle.TextAlign = ContentAlignment.MiddleCenter

Click Image Controls Other Than Buttons Have Click Event Procedures

PictureBox Controls Can Respond to Mouse Clicks

PictureBox Control

- As we saw earlier the Image Property can be set to a graphic image of some sort
- The flag images in Tutorial 2-16 are clickable
- The click event can be handled by code to take whatever action is desired

PictureBox Click Event code

- When PictureBox picUSA is clicked, the lblMessage text property is set to display United States of America

  Private Sub picUSA_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles picUSA.Click
    ' Display the country name
    lblMessage.Text = "United States of America"
  End Sub

- Do tutorial 2-16 to demonstrate
Dynamic Help

- Dynamic Help provides context sensitive help information relevant to the operation you are currently performing.
- The Properties window and Dynamic Help window may hide each other since they occupy the same area of the screen.
- If one of these windows is hidden, click the tab at the bottom to select which to view.

Help Menu

- The usual categories of Help you are probably accustomed to in Microsoft applications:
  - Contents…
  - Index…
  - Search…
- Are available through this window also.

Types of Errors: Compile Errors

- These are errors in the syntax (form) of your program.
- Visual Basic will inform you of these as soon as they are found.
- The area of the error will be underlined with a jagged blue line.
- A description of the error will be given in the Task List window.
- Display the Task List window by selecting Error List from the View menu option.

Types of Errors: Runtime Errors

- These errors occur as your program runs.
- Runtime errors create incorrect results but do not prevent your program from running.
- Visual Basic will detect some of these and inform you about them.
- Others you must detect yourself.
- Always carefully check the operation of your program to be sure that it operates as required.