Chapter 1

An Introduction to Visual Basic 2015

A Guide to this Instructor's Manual:

We have designed this Instructor's Manual to supplement and enhance your teaching experience through classroom activities and a cohesive chapter summary.

This document is organized chronologically, using the same headings that you see in the textbook. Under the headings you will find: lecture notes that summarize the section, Teaching Tips, Class Discussion Topics, and Additional Projects and Resources. Pay special attention to teaching tips and activities geared towards quizzing your students and enhancing their critical thinking skills.

In addition to this Instructor's Manual, our Instructor's Resources also contain PowerPoint Presentations, Test Banks, and other supplements to aid in your teaching experience.

At a Glance

Instructor's Manual Table of Contents

Lecture Notes

Overview

Chapter 1 introduces students to the Visual Studio 2015 integrated development environment (IDE). Students learn how to create a splash screen for the Crighton Zoo. This chapter contains three lessons:

- Lesson A covers the Visual Studio 2015 IDE and the use of its various windows, setting properties of objects, and saving, opening, and closing applications.
- Lesson B introduces how to add controls to forms and enter code in the Code Editor window, as well as how to run the project.
- Lesson C covers timers, enabling and disabling form objects, and printing code and the interface.

Lesson A Objectives

After studying Lesson A, students should be able to:

- Start and customize Visual Studio 2015
- Create a Visual Basic 2015 Windows application
- Manage the windows in the IDE
- Set the properties of an object
- Restore a property to its default setting
- Save a solution
- Close and open an existing solution

Teaching Tips

Previewing the Splash Screen

1. Introduce the concept of a splash screen. Use Figure 1-1 to introduce the Crighton Zoo screen that will be developed in this chapter.

Teaching Tip	Run the completed program to demonstrate the final result that will be achieved in this chapter. It's important that students know what their goal is. This gives them a better understanding of *why* they are doing the things they are doing.

2. Introduce the concept of an integrated development environment (IDE).

The Splash Screen Application

1. Describe how to start Visual Studio Ultimate 2015.

2. Describe the startup screen for Visual Studio 2015 Ultimate shown in Figure 1-2.

3. Walk through the options in Figure 1-3 and explain which check boxes should and should not be checked.

4. Introduce the concept of a user interface.

5. Introduce the concept of a solution and a project.

6. Walk through the steps involved in creating a new application in Visual Basic 2015.

Teaching Tip	A common problem for new programmers is misunderstanding where the files reside. To avoid overwriting files that have the same name, encourage students to create a new directory for each new project. Also, remind students that unlike other languages they may know, Visual Basic requires students to save the application *before* they start creating screens and writing code.

7. Remind students to remember there they save the solution as shown in Figure 1-4.

8. Use Figure 1-5 to point out the benefits of using the Auto Hide button.

9. Describe the state of the IDE when a new solution has been created, as shown in Figure 1-6.

Teaching Tip	Alert students to the location of the default project on the C drive, as shown in Figure 1-3. Encourage students to save their work to a USB Flash drive.

Teaching Tip	Point out to students that the textbook is written to cover Windows 8 operating systems. The screen shots are from Windows 8, and students' screens will look different under a different OS.

Managing the Windows in the IDE

1. Describe the use of the standard Close button on windows in the IDE.

2. Introduce the Auto Hide button and explain how it works.

Teaching Tip	Demonstrate how to use Auto Hide and how to force windows to stay open. It can be very disconcerting to students when windows appear and disappear seemingly without reason.

3. Describe the Alphabetical button in the Properties window.

Teaching Tip	Encourage students to use the Alphabetical button to quickly locate properties. If they do not use alphabetical order, they must know how Microsoft has grouped the properties.

4. Describe the major components of the IDE shown in Figure 1-6. Point out the location of the Solution Explorer window, the Properties window, and the Windows Form Designer window.

The Windows Form Designer Window

1. Introduce the concept of the graphical user interface, or **GUI**.

2. Describe the **Windows Form Designer window**, which is shown in Figure 1-7.

3. Introduce the concept of a **Windows Form object**, and point out that a **form** is created when a new project is created.

Teaching Tip	Remind students that all objects come from classes. The Windows Form object is no exception.

4. Describe the tab at the top of the Windows Form Designer window that identifies the disk file containing the form object. Explain the usefulness of this tab when you have multiple forms and code windows open.

Teaching Tip	You can demonstrate a form object's built-in capabilities by running the program from the IDE now. Point out the behaviors of the form object (minimize, maximize, resizability, etc.) that students expect from a window.

The Solution Explorer Window

1. Use Figure 1-8 to describe the purpose of the **Solution Explorer window**.

2. Introduce the concept of a **source file** and **code**.

3. Introduce the term **form file**. Point out that a Windows Form object contains code as well as the GUI.

Teaching Tip	Point out the file extensions for the various files in the project and solution. Ensure that students understand that all of these files are required. It is a common mistake for new programmers to believe that only one file is required.

4. Discuss the importance of using a good naming convention for filenames.

The Properties Window

1. Introduce the concept of **properties** and describe the **Properties window** shown in Figure 1-9.

2. Describe the **Object box** and the **Properties list** in the Properties window. Use Figure 1-9 to describe the **Settings box** in the Properties list.

Teaching Tip	Understanding the role that properties play in Windows-based application development is extremely important. Demonstrate the effects of several properties to emphasize their importance.

3. Use Figure 1-10 to show the relationship between the names shown for the form file, form object, and property box entry.

Properties of a Windows Form

1. Remind students that a form is an object and each form has its own set of properties.

2. Using Figure 1-11, point out that the properties can be displayed alphabetically or by categories.

3. Introduce the concept of a **class definition** and a **namespace**.

4. Explain the use of the **dot member access operator** to indicate the hierarchy of namespaces.

The Name Property

1. Introduce the Name property of the Windows Form object.

Teaching Tip	Ensure that students understand the difference between a form object's name and the external filename containing the form. Also distinguish between a form object's name and the text property of a form.

2. Describe the use of Hungarian notation and **camel case** as a standard for a naming convention.

3. Describe the naming convention used in this book.

4. Describe how to change the Name property using the Properties box.

Teaching Tip	Hint: If you change the external filename in the Solution Explorer, it will be automatically changed in the Properties box. Be sure to keep the .vb extension on the name in the Solution Explorer.

The Text Property

1. Introduce the Text property of the form object.

Teaching Tip	Point out that the Text property is common to many objects in Visual Basic. Remind students (again) that the Text property is *not* the form object's name.

The StartPosition Property

1. Introduce the StartPosition property of the form object, and remind students that the splash screen of an application typically appears in the middle of the screen. Explain how the CenterScreen property accomplishes this.

The Font Property

1. Introduce the Font property, and point out that this property is common to many objects in Visual Basic.

2. Explain that Segoe (pronounced SEE-go) UI is the recommended font because it offers improved readability.

Teaching Tip	Now is a good time to give a short lesson covering serif and sans serif fonts, and when to use each. Numerous studies have been conducted on the readability of serif versus sans serif typefaces without reaching a consistent conclusion. Some studies indicate that serif typefaces may be more readable in print, but there is no consensus. Studies of on-screen use are also ambiguous, suggesting that low screen resolutions make serifs more difficult to discern, with a resulting erosion of readability compared to sans serif fonts.

The Size Property

1. Introduce the Size property, and point out that it consists of two properties: Height and Width.

2. Describe the plus box that appears next to the Size property to provide access to the Height and Width properties. Introduce the concept of a pixel.

Teaching Tip	Tell students that they also can size an object by selecting it and then pressing and holding down the Shift key as they press the up, down, right, or left arrow key on the keyboard. Alternatively, they can set the object's Size property.

Setting and Restoring a Property's Value

1. Introduce the BackColor property, and demonstrate how to set and restore a property's value.

2. Use Figure 1-12 to describe the status of the form in the IDE. Point out the asterisk on the tab of an unsaved form.

Saving a Solution

1. Remind students of the importance of periodically saving their solution to avoid losing work in the event of a power or computer failure.

2. Describe how to save all files in the solution.

Closing the Current Solution

1. Describe how to close the currently open solution.

2. Explain the difference between closing a file and closing a solution.

Opening an Existing Solution

1. Describe how to open an existing solution from within Visual Studio.

Teaching Tip	Students who are new to Visual Studio often try to open their solutions directly from Windows Explorer. Because they are more likely to remember the name of the form or project than the solution, they often fail to open the existing solution and instead allow Visual Studio to create a new solution each time they open a form or project. It is a good idea to show students the .sln file and admonish them to use that file to open the solution from Windows Explorer.

Exiting Visual Studio 2015

1. Describe how to exit Visual Studio 2015.

Teaching Tip	Remind students that if they attempt to exit without saving all of their work, Visual Studio will prompt them to save the files first.

Lesson A Summary

1. Review the process of starting Visual Studio 2015.
2. Summarize the steps involved in configuring Visual Studio.
3. Explain the process of creating a Visual Studio 2015 Windows application.
4. Remind students that they can reset the layout of the IDE to its default values.
5. Review how to close and open a window in the IDE.
6. Review the use of the Auto Hide button to hide a window, and to display a window temporarily or permanently.
7. Summarize the process of setting a property value.
8. Review the process of naming objects.
9. Review the form's Text, StartPosition, Font, Size, and BackColor properties.
10. Remind students how to restore a property to its default value.
11. Review how to save and close a solution, and how to open an existing solution.
12. Review how to exit Visual Studio 2015.

Quick Quiz 1

1. A Visual Studio 2015 _____ is a container that stores the projects and files for an entire application.
   Answer: solution

2. True or False: Each object in an object-oriented language has attributes that determine its appearance and behavior.
   Answer: True

3. True or False: Properties in the Properties window can be displayed alphabetically or numerically.
   Answer: False

4. You create (or design) the graphical user interface in the _____ window of Visual Studio 2015.
   Answer: Windows Form Designer

Lesson B Objectives

After studying Lesson B, students should be able to:

- Add a control to a form
- Set the properties of a label, picture box, and button control
- Select multiple controls
- Center controls on the form
- Open the Project Designer window
- Start and end an application
- Enter code in the Code Editor window
- Terminate an application using the `Me.Close()` instruction
- Run the project's executable file

Teaching Tips

The Toolbox Window

1. Remind students how to open the sample application's solution in Visual Studio 2015.

2. Use Figure 1-13 to introduce the **Toolbox window**. Point out that each tool in the **toolbox** represents a class from which an object can be instantiated.

3. Introduce the concept of **controls**.

The Label Tool

1. Introduce the Label tool for creating a **label control**. Point out that the purpose of a label is to display text that cannot be edited by the user at run time.

2. Use Figures 1-14 and 1-15 to describe the process involved in creating label controls.

Teaching Tip	Remind students of the importance of using a naming convention for objects when designing applications. Point out that the naming convention followed in this book is to precede all control names with a three-character ID indicating the type of control, such as "lbl" for label controls.

Setting the Text Property

1. Remind students that the Text property of a label control determines the value that the users see on the form.

2. Describe the process involved in setting the Text property of a label control.

3. Describe the effects of the AutoSize property when the Text property value is changed.

Setting the Location Property

1. Introduce the Location property.

2. Describe how to set the X property and Y property values to specify the location.

Teaching Tip	In most cases, programmers use the drag-and-drop method to position and size controls on the form. Explain that using the up, down, left, and right arrows on the keyboard can make minute adjustments to an object's location. However, it is sometimes necessary to use the numeric property values directly to obtain precise sizing and location.

Changing a Property For Multiple Controls

1. Point out that controls placed on a form inherit the font settings of the form, but can be assigned new settings if desired.

2. Use Figure 1-16 to describe how to select multiple controls on the form in Design view.

3. Describe how to change the Font property settings.

Teaching Tip	Point out that many properties are shared by more than one type of control. For these properties, you can select multiple types of controls and set the property value in the Properties window.

Teaching Tip	Selecting multiple controls provides a way to determine if all of the controls have the same value for a property. After selecting multiple controls, if the value for the property is the same for each of them, it will show in the Properties window. If it is not the same for all of the selected controls, the value will not show in the Properties window but can be set at that time.

Using the Format Menu's Order Option

1. Point out that the Format menu provides an Order option to control the layering of one or more controls on the form.

2. Use Figure 1-17 to describe how to add a control on a layer and place it over another layer with other controls.

3. Use Figure 1-18 to describe how to send a layer that contains controls behind another layer that contains other controls.

The PictureBox Tool

1. Introduce the PictureBox tool for creating a **picture box control** to display an image on a form.

2. Describe the process for adding a picture box control to the form.

3. Use Figure 1-19 to describe how to use the task list contained in the task box on the picture box control to view and perform common tasks associated with the control.

4. Use Figure 1-20 to explain how to import an image into the solution and select it for use in the picture box control.

5. Use Figure 1-21 to describe the effects of the StretchImage option of the picture box control.

Teaching Tip	Tell students that they should minimize the use of graphics in the user interface. Guidelines for GUI design are covered in more detail in Chapter 2.

Using the Format Menu to Align and Size

1. Point out that the Format menu provides options to manipulate the controls on the form.

2. Introduce the term **reference control**.

3. Introduce the Align, Make Same Size, and Center in Form options, and describe the effects of each option.

4. Describe how to center a control horizontally on the form.

The Button Tool

1. Introduce the Button tool for creating a **button control**.

2. Describe the process for adding a button control to the sample application's form.

Teaching Tip	Remind students to give all objects meaningful names that follow the naming standards. Although there are no rules for naming objects, following a naming standard and sticking with it makes the entire process of creating an application easier.

Starting and Ending an Application

1. Introduce the concept of a **startup form**.

2. Use Figure 1-22 to describe how to verify the startup form by using the project properties.

3. Introduce the concept of an **executable file**, and point out that this is what is normally given to the user.

4. Use Figure 1-23 to describe how to start and exit the application.

The Code Editor Window

1. Introduce the concept of **events** and **event procedures** in a Windows application.

Teaching Tip	Remind students that Windows is an event-driven operating system. Therefore, a Windows application must be written to respond to events that are initiated by user actions.

2. Describe how to open the Code Editor window in Visual Studio.

3. Use Figure 1-24 to describe the code that will be placed in the Code Editor window by default.

Teaching Tip	Point out that students should name objects before writing code so that the event procedure names will match the object names.

4. Use Figure 1-25 to explain how to expand and collapse a region of code.

5. Use Figure 1-26 to introduce the concept of a code template for event procedures.

6. Introduce the term **syntax**, and describe the syntax of the **procedure header** and **procedure footer**.

7. Introduce the concept of a **keyword**, and explain the effects of the keyword `Private`.

8. Introduce the concept of a **sub procedure**, and describe the required syntax elements of a sub procedure.

Teaching Tip	Spend some time discussing the color coding of text that appears in the Code Editor window. Ensure that students understand that this provides another way for them to validate that they have used the correct keywords and syntax.

The `Me.Close()` Instruction

1. Introduce the term **method**.

2. Introduce the `Me.Close()` method for closing the current form.

Teaching Tip	Ensure that students understand the distinction between closing a form and exiting the application.

3. Use Figures 1-27 and 1-28 to demonstrate the use of IntelliSense to code the `Me.Close()` instruction.

4. Stress the importance of always testing a program after writing code.

Lesson B Summary

1. Review the process of adding a control to a form.

2. Remind students that label controls are used to provide non-editable text at run time.

3. Summarize the methods that can be used to move a control to a different location on the form.

4. Review the use of the Font property to control the type, style, and size of the font.

5. Review the methods that can be used to select multiple controls and cancel the selection of one or more controls.

6. Review the methods for centering and aligning controls on a form, and for making two or more controls the same size.

7. Summarize the steps involved in adding an image to the user interface.

8. Review the use of the button control.

9. Remind students about the Project Designer window for controlling project settings such as the startup form and the name of the executable file.

10. Summarize the methods to start and stop an application.

11. Remind students how to open the Code Editor window.

12. Review how to display an event procedure in the Code Editor window.

13. Review the use of the `Me.Close()` instruction for closing the current form.

14. Remind students that they can test to see if the project works correctly by running the project's executable file.

Quick Quiz 2

1. True or False: The `Me.Exit()` method can be used to close the current form.
   Answer: False

2. A(n) _____ control should be used to display text that the user cannot edit at run time.
   Answer: label

3. The rules of a programming language are called its _____.
   Answer: syntax

4. True or False: The Text property is only available for the label control.
   Answer: False

Lesson C Objectives

After studying Lesson C, students should be able to:

- Set the properties of a timer control
- Delete a control from the form
- Delete code from the Code Editor window
- Code a timer control's Tick event procedure
- Prevent the user from sizing a form
- Remove and/or disable a form's Minimize, Maximize, and Close buttons
- Print an application's code and interface

Teaching Tips

Using the Timer Tool

1. Introduce the Timer tool for creating a **timer control**.

2. Introduce the concept of a millisecond and the Interval property.

3. Introduce the timer control's Enabled property and its Tick event.

Teaching Tip	Remind students that the default value for a timer's enabled property is set to false; that is, timers are set *not* to run and students must enable the property in order for the timer to work.

4. Use Figure 1-31 to introduce the concept of the **component tray** and how certain controls do not appear on the form during **run time**.

5. Step through the process of adding a timer control to the sample application.

Teaching Tip	Caution students about selecting the Interval value for the timer. If it is too short, users will not have time to read the information on the screen. If it is too long, users will be annoyed by having to wait for the screen to close.

6. Point out that the Exit button will no longer be required on this form. Use Figure 1-32 to describe how to delete the Exit button's Click event procedure code.

Setting the FormBorderStyle Property

1. Introduce the FormBorderStyle property and discuss the possible values for this property.

2. Describe how to set the FormBorderStyle property.

Teaching Tip	Explain that there are situations when you do not want to allow the user to change the size of a form. When you do allow sizing, you should also use anchoring to ensure that your form layout will be preserved if the form's size is changed.

The MinimizeBox, MaximizeBox, and ControlBox Properties

1. Introduce the MinimizeBox and MaximizeBox properties of a form, and describe how to enable or disable these properties.

2. Introduce the ControlBox property. Point out that disabling this property removes the title bar from the form, including the Minimize, Maximize, and Close buttons.

3. Use Figure 1-33 to describe the finished application.

Teaching Tip	Normally, only special-purpose forms will not have a title bar. If you remove the title bar, ensure that there is a way for the form to close, either through a timer or using a button on the form.

Printing the Application's Code and Interface

1. Discuss the reasons for printing an application's code.

2. Using Figure 1-34, describe how to indicate whether collapsed regions and/or line numbers should be printed.

Teaching Tip	Using comments liberally in the code improves the value of the printed application code.

Lesson C Summary

1. Summarize the use of the timer control and how to set its Interval property.

2. Review the steps involved in deleting a control and its code from a form.

3. Review the form's FormBorderStyle property and its effects.

4. Summarize the use of the MinimizeBox and MaximizeBox properties for controlling whether the user can minimize and/or maximize a form at run time.

5. Summarize the use of the ControlBox property for determining if a form will have a title bar.

6. Review the steps involved in printing an application's user interface and code.

Quick Quiz 3

1. Which property of a form object will remove the entire title bar?
   Answer: ControlBox

2. When using a timer control, the event named _____ will determine when the code is executed.
   Answer: Tick

3. True or False: You cannot use line numbers with Visual Basic code.
   Answer: False

4. True or False: The BorderStyle property of a form controls the appearance of the form's border.
   Answer: False

Class Discussion Topics

1. Ask students to describe what they consider to be good and bad splash screens. What did they like? What did they dislike? Can they identify common attributes that made the splash screens enjoyable?

2. Ask students to name other situations for which a timer control might be useful in an application. When is it preferable to use a timer control and not a button clicked by the user to cause the application to proceed?

Additional Projects

1. Ask students to design a splash screen for an application that will provide a picture gallery of their family photos.

2. Show students a screen that contains many objects (labels, check boxes, text boxes, etc.) and have them tell how many objects they can count. Even after telling them that *everything* they see is an object, they most likely will not count correctly.

3. Ask students to experiment with the Interval property of the timer control by setting it to both smaller and larger values. Have them run the program for friends or family and determine an appropriate length of time for users to read all of the information on the screen.

Additional Resources

1. Gallery of splash screens:
   www.guidebookgallery.org/splashes

2. Tools for Visual Studio 2015:
   https://msdn.microsoft.com/en-us/library/dd831853(v=vs.140).aspx

3. Microsoft Consulting Services Naming Conventions for Visual Basic:
   http://support.microsoft.com/kb/110264

4. Visual Studio 2015 Getting Started Tutorials:
   https://msdn.microsoft.com/en-us/library/dd492171(v=vs.140).aspx

5. What's New for Visual Basic in Visual Studio 2015:
   https://msdn.microsoft.com/en-us/library/we86c8x2(v=vs.140).aspx

Key Terms

- **Button control**—the control commonly used to perform an immediate action when clicked
- **Camel case**—used when entering object names in Hungarian notation; the practice of entering the object's ID characters in lowercase and then capitalizing the first letter of each subsequent word in the name
- **Class definition**—a block of code that specifies (or defines) an object's appearance and behavior
- **Code**—program instructions
- **Component tray**—a special area in the IDE; stores controls that do not appear in the interface during run time
- **Controls**—objects (such as a label, picture box, or button) added to a form
- **Dot member access operator**—a period; used to indicate a hierarchy
- **Event procedure**—a set of Visual Basic instructions that tell an object how to respond to an event

- **Events**—actions to which an object can respond; examples include clicking and double-clicking
- **Executable file**—a file that can be run outside of the Visual Studio IDE, such as from the Run dialog box in Windows; the file has an .exe extension on its filename
- **Form**—the foundation for the user interface in a Windows application; also called a Windows Form object
- **Form file**—a file that contains the code associated with a Windows form
- **GUI**—the acronym for graphical user interface
- **Keyword**—a word that has a special meaning in a programming language
- **Label control**—the control used to display text that the user is not allowed to edit while an application is running
- **Method**—a predefined Visual Basic procedure that you can call (invoke) when needed
- **Namespace**—a block of memory cells inside the computer; contains the code that defines a group of related classes
- **Object box**—the section of the Properties window that contains the name of the selected object
- **OOP**—the acronym for object-oriented programming
- **Picture box control**—the control used to display an image on a form
- **Point**—used to measure font size; 1/72 of an inch
- **Procedure footer**—the last line in a procedure
- **Procedure header**—the first line in a procedure
- **Properties**—the attributes that control an object's appearance and behavior
- **Properties list**—the section of the Properties window that lists the names of the properties associated with the selected object, as well as each property's value
- **Properties window**—the window that lists an object's attributes (properties)
- **Reference control**—the first control selected in a group of controls; this is the control whose size and/or location you want the other selected controls to match
- **Run time**—the state of an application while it is running
- **Settings box**—the right column of the Properties list; displays each property's current value (setting)
- **Solution Explorer window**—the window that displays a list of the projects contained in the current solution and the items contained in each project
- **Source file**—a file that contains code
- **Startup form**—the form that appears automatically when an application is started
- **Sub procedure**—a block of code that performs a specific task
- **Syntax**—the rules of a programming language
- **Timer control**—the control used to process code at one or more regular intervals
- **Toolbox**—refers to the Toolbox window
- **Toolbox window**—the window that contains the tools used when creating an interface; each tool represents a class; referred to more simply as the toolbox
- **Windows Form Designer window**—the window in which you create an application's GUI
- **Windows Form object**—the foundation for the user interface in a Windows application; referred to more simply as a form