Computer Programming I

Essential Standard 6.00 Apply tools to obtain and validate user input.

Indicator 6.02 Apply Procedures to Develop Message, Input, and Dialog Boxes. (3%)

Message, Input, and Dialog Boxes

* Message, input and dialog boxes increase the interactivity of your application.
* They allows the user to interface with your application.
* In this presentation we will look at how to add these new control objects to your application.

The MessageBox Control

* The MessageBox is a predefined dialog box that displays information to the user.
* The following code demonstrates how to use the Show method of the MessageBox class to display a message to the user.

Private Sub btnMsgBox\_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnMsgBox.Click

 **MessageBox.Show("This is the text in the MessageBox")**

End Sub

* You can display a message box to request information.
* There are options available with the MessageBoxButtons that allow you to offer to the user more options.

MessageBoxButtons.AbortRetryIgnore

MessageBoxButtons.OK

MessageBoxButtons.OKCancel

MessageBoxButtons.RetryCancel

MessageBoxButtons.YesNo

MessageBoxButtons.YesNoCancel

* You can compare the user’s choice with the DialogResult.



The InputBox Function

* The input box is a function that allows a user to input information.
* It displays a prompt in a dialog box, waits for the user to either hit the enter key or click a button, then returns the information from the textbox as a string.
* There is one required parameter, the Prompt. You will need to add a prompt that will be displayed at the top of the dialog box.
* The Title parameter, while set up with a default of nothing, should be set. The title is displayed in the title bar at the top of the dialog box.

Prompt

Title

* Note that there are additional optional parameters.
	+ XPos, YPos
		- These set the position where the input box will be displayed.
* This example sets the result of the input box to the string variable strAnswer.


* Don’t forget that the input box returns a STRING. If you want to turn this value into a numeric value you will need to use the Val() function.
* Creating a Dialog Box
* Dialog boxes are used to get information from the user.
* You can create your own dialog box by adding a new Windows Form to your application.
* When you create a dialog box, you need to know it’s “result”.
	+ For example, did the user click “OK” or “Close”.
* From the form that displays the dialog box, you can use the DialogResult property to determine which button was clicked.
* You can also set the DialogResult property if no buttons are used.

Creating a Dialog Box

* Steps to add a programmer defined dialog box
1. From the Solution Explorer, right-click on your project, select add, then Windows Form.
Or you can go to the Project menu, select Add Windows Form, then Windows Form.
2. Name the Form (otherwise it will be Form2 if it will be the second form, otherwise it will be Form3, etc).
3. Your new form will be displayed in the Design Window.
4. In the Properties window,
	1. Change the FormBorderStyle property to FixedDialog.
	2. Set the ControlBox, MinimizeBox, and MaximizeBox to false
5. Customize your form.

Using a Dialog Box

* You can use the show and hide your new dialog box using the following commands.
* To show the dialog box
 *formName*.Show()
* To hide the dialog box
 *formName.*Hide()
* To close the dialog box
 *formName*.Close()

For more information on this topic

* [http://msdn.microsoft.com/en-us/library/2chz8edb(v=VS.80).aspx](http://msdn.microsoft.com/en-us/library/2chz8edb%28v%3DVS.80%29.aspx)

For more information on Dialog Boxes

* <http://msdn.microsoft.com/en-us/library/2chz8edb.aspx>