

CMPE1666

Intermediate Programming

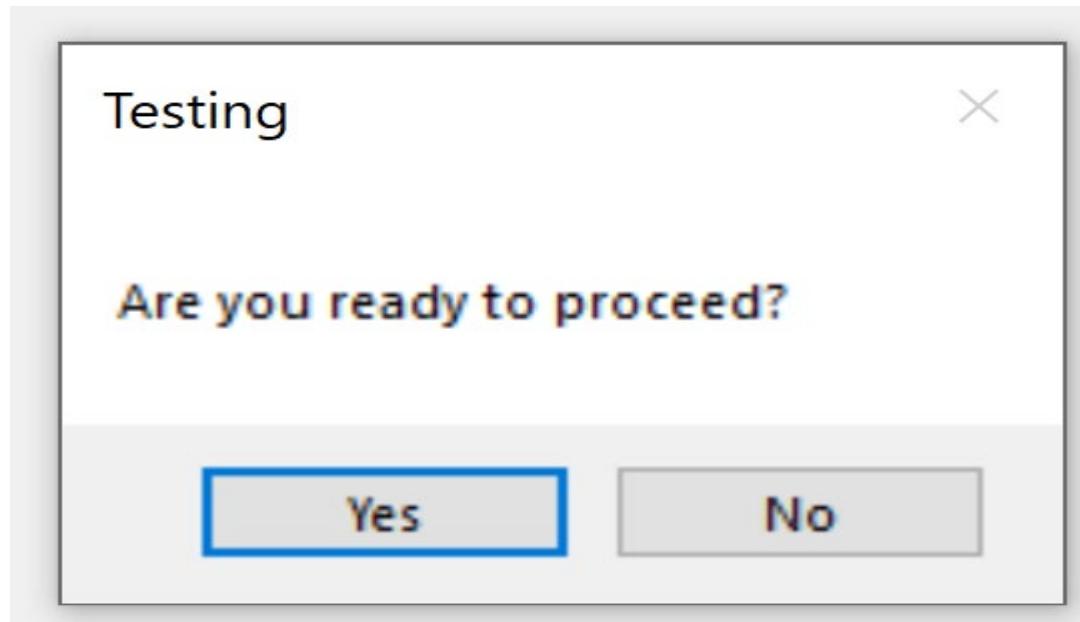
Lecture 8-Modal Dialogs

Acknowledgements To JD Silver

Dialogs

- ▶ Dialogs are commonly used to gather data from the user.
- ▶ There are two types of dialogs: modal and modeless.
- ▶ A modal dialog steals the focus from the main form until it is dismissed.
- ▶ A modeless dialog stays on the screen and allows the main form to retain the focus.

An Example of Modal Dialog- The Message Box



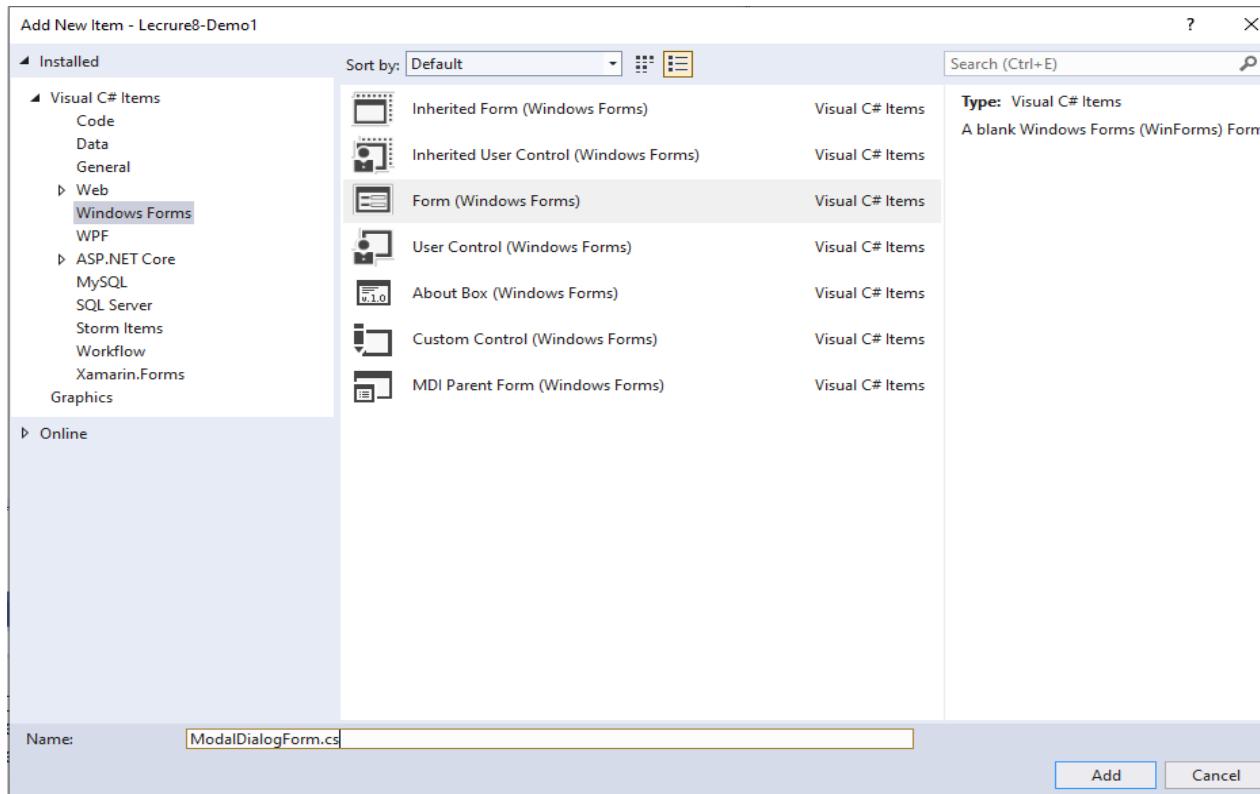
We note that as long as the message box is active we cannot work on the main form

Dialog Project Example

- ▶ We'll study the Modal Dialog concept through an example.
- ▶ Create a Form-based project- Call it Lecture8demo1
 - We'll work with it to illustrate the creation of Modal Dialog

Creating a New Modal Dialog

- ▶ In the Project menu, choose the “Add New Item” button, shown below, then select “Windows Form” and “Forms (Windows Forms)”.
 - ▶ Provide the name **ModalDialogForm** for the form then click “Add”
 - ▶ The form will be created and a class generated for it.

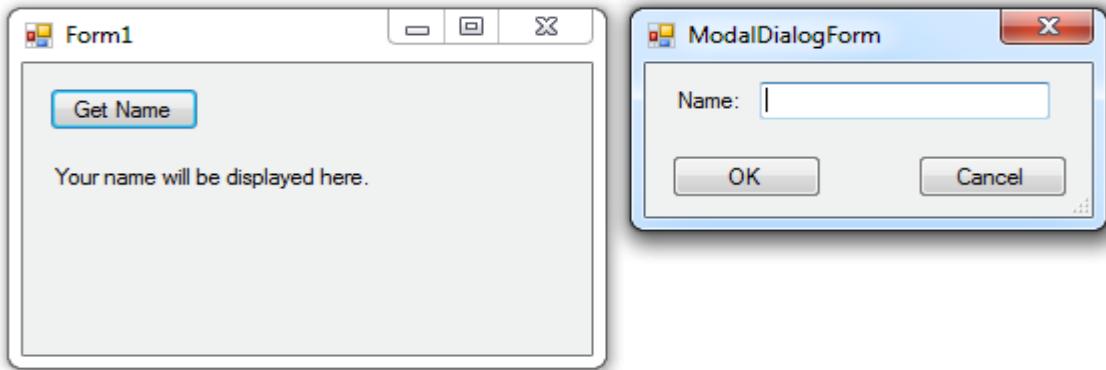


Creating a New Dialog

- ▶ Visual Studio will generate the class and file for the dialog.
- ▶ The form will originally be created using a `FormBorderStyle` of `resizable`; this should be changed to `FixedDialog`.
- ▶ You should also remove the `MinimizeBox`, `MaximizeBox`, and the `ControlBox`.
 - These are under the `Window Style` group of properties.
 - Set them to false

Dialog Example

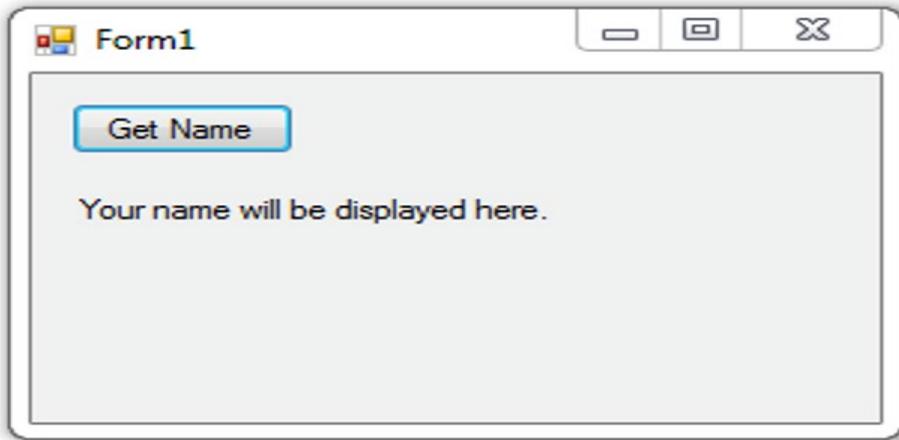
- In this example, we'll create A modal dialog to obtain a name from the user.



- The dialog will be shown when a button is pressed.
- If the user presses the OK button in the dialog, the name will be displayed on the main form.
- If the user presses the Cancel button in the dialog, any text entered will not be displayed.
- We'll build the application stepwise

Dialog Example

- ▶ Step 1: We add the button and the label to the main form (Form1)
- ▶ Name the button `UI_GetName.Btn` and the label `UI_ResultLbl`



- ▶ We run the program, we see that Form1 appears, but so far there is no way to obtain the other form.
- ▶ We note that Program.cs only creates a new form of class Form1

Dialog Example

- ▶ Step 2: add an event handler for btnGetName, as below

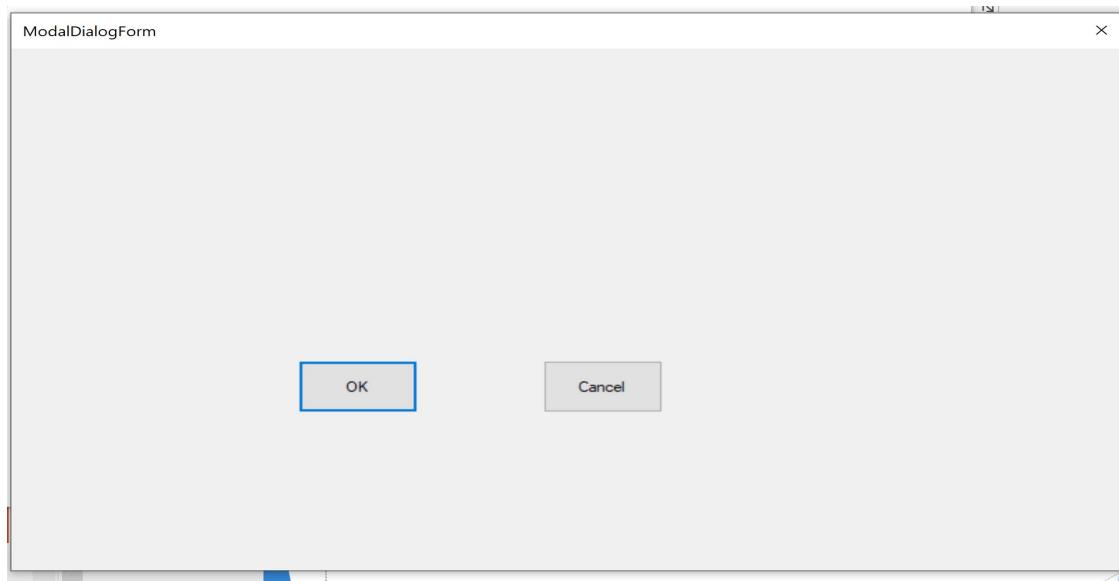
```
private void UI_GetName_Btn_Click(object sender, EventArgs e)
{
    ModalDialogForm dialog = new ModalDialogForm();

    dialog.ShowDialog();
}
```

- ▶ We run the program again. We note that now when we click the button, the second form appears, but doesn't do much

Dialog Example

- ▶ Step 3: We'll add an OK and a Cancel Button to the modal dialog form (These are simply buttons with Text “OK” and “Cancel” respectively)
- ▶ We name the buttons as UI_OK_Btn and UI_Cancel_Btn respectively.



Dialog Example

- ▶ Step 4: We add a handler for the OK button as follows:

```
private void UI_OK_btn_Click(object sender, EventArgs e)
{
    DialogResult = DialogResult.OK;
}
```

- ▶ This will cause the form to close and return the value DialogResult.OK to the caller.
- ▶ The caller can pick the returned value

Dialog Example

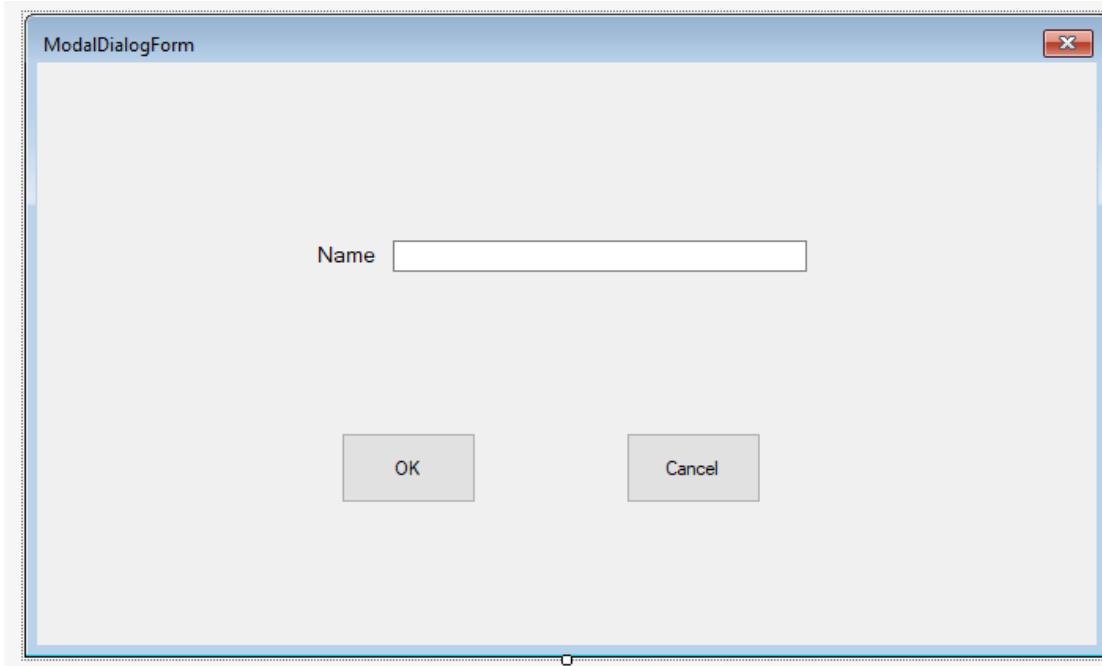
- ▶ Step 5
- ▶ Before proceeding any further, we'll test if our Modal form is working and the button is giving the desired result.
- ▶ Include the following event handler for the “Get Name” Button in Form1.

```
private void UI_GetName_Btn_Click(object sender, EventArgs e)
{
    ModalDialogForm dialog = new ModalDialogForm();
    if (dialog.ShowDialog() == DialogResult.OK)
        UI_Result_Lbl.Text = $"The Result was: OK";
}
```

- ▶ We then run the program, click on the Get Name button, when the Modal dialog appears, we click on OK and observe the output.

Dialog Example

- ▶ Step 6 - Now we add the text box and the corresponding label to the Modal dialog.
- ▶ Name the text box UI_Input_Tbx



Dialog Example

```
private void UI.GetName.Btn_Click(object sender, EventArgs e)
{
    ModalDialogForm dialog = new ModalDialogForm();
    if (dialog.ShowDialog() == DialogResult.OK)
    {
        //Place code here to handle the result from the dialog
    }
}
```

Dialog Example

- ▶ In this example, the dialog will allow the user to enter a name in a textbox, which should be returned to the main form.
- ▶ Note that the text box belongs to the `ModalDialogForm` object. It's thus not directly accessible to the event handlers on the Main form
- ▶ There needs to be mechanism to allow communication from the dialog to the main form.
- ▶ We can create **Public Methods** in the dialog, which can then be read from the dialog in the main form.

Dialog Example

- ▶ **Public Methods** allow access to the inner workings of another class.
- ▶ The modal dialog is another class that is separate from the main form (also a class).
- ▶ We will use a public method to set the value of a member variable of the dialog and another method to get the value of the variable.

Dialog Example

- ▶ The method `getUserName()` is used to return the value of the `inputValue`

```
public string getUserName()
{
    return UI_Input_Tbx.Text;
}
```

Dialog Example

- ▶ The public method can be used in the main form to fetch the text entered by the user.

```
if (dialog.ShowDialog() == DialogResult.OK)
{
    //Use the property to obtain the text from the dialog
    UI_Result_Lbl.Text = dialog.getUserName();
}
```

Dialog Example

- ▶ A public method can also be used to set the value of the textbox when the dialog is shown.

```
public void setUsername(string name)
{
    UI_Input_Tbx.Text=name;
}
```

Dialog Example

- ▶ The method is then used to pass a value from the main form to the dialog when the dialog is created.

```
private void UI_GetName_Btn_Click(object sender, EventArgs e)
{
    ModalDialogForm dialog = new ModalDialogForm();
    dialog.setUserName( "default");

    if (dialog.ShowDialog() == DialogResult.OK)
    {
        //Use the property to obtain the text from the dialog
        UI_Result_Lbl.Text = dialog.getUserName();
    }
}
```

Closing a Dialog From Within

- ▶ A modal dialog can be closed by assigning a result to the DialogResult property.
- ▶ In the following example, when the OK button is pressed the DialogResult property is assigned the value DialogResult.OK.
- ▶ When the Cancel button is pressed the DialogResult property is assigned the value DialogResult.Cancel.

Closing a Dialog From Within

1 reference

```
private void UI_OKBtn_Click(object sender, EventArgs e)
{
    DialogResult = DialogResult.OK;
}
```

1 reference

```
private void UI_CancelBtn_Click(object sender, EventArgs e)
{
    DialogResult=DialogResult.Cancel;
}
```

Checking Exit Condition

- ▶ Generally, if the user closes the dialog with the OK button, some action is required.
- ▶ If the Cancel or X button was pressed, you usually want to ignore any changes made in the dialog.
- ▶ The dialog reports how it was closed with the return value from the ShowDialog() method.

Checking Exit Condition

```
private void UI_GetName_Btn_Click(object sender, EventArgs e)
{
    ModalDialogForm dialog = new ModalDialogForm();
    dialog.setUserName( "default");

    if (dialog.ShowDialog() == DialogResult.OK)
    {
        //Use the property to obtain the text from the dialog
        UI_Result_Lbl.Text = dialog.getUserName();
    }
    else
    {
        //dialog was cancelled using Cancel or X button
        MessageBox.Show("Dialog was closed with Cancel or X button ");
    }
}
```

Lecture 8 -Exercise1

- ▶ Write a program that has 2 forms with the controls shown below. The left form is the main form that runs when the program starts.
- ▶ The form on the right is a modal dialog. It must be non-resizable and must not contain the minimize box, maximize box and control buttons.
- ▶ Clicking on Main Form button must cause the Modal Dialog on the right to appear.
- ▶ The user will input 2 values. When the user clicks on OK in the Modal form, the values must be picked up by the main form, displayed in the 2 read-only textboxes and then the sum must be calculated and displayed.

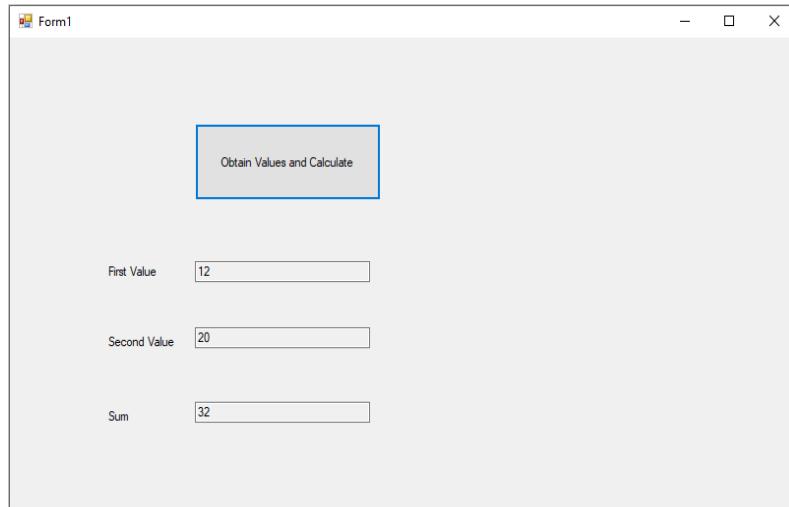
Form1

Obtain Values and Calculate

First Value

Second Value

Sum



ModalDialog

Input1

Input2

OK Cancel

