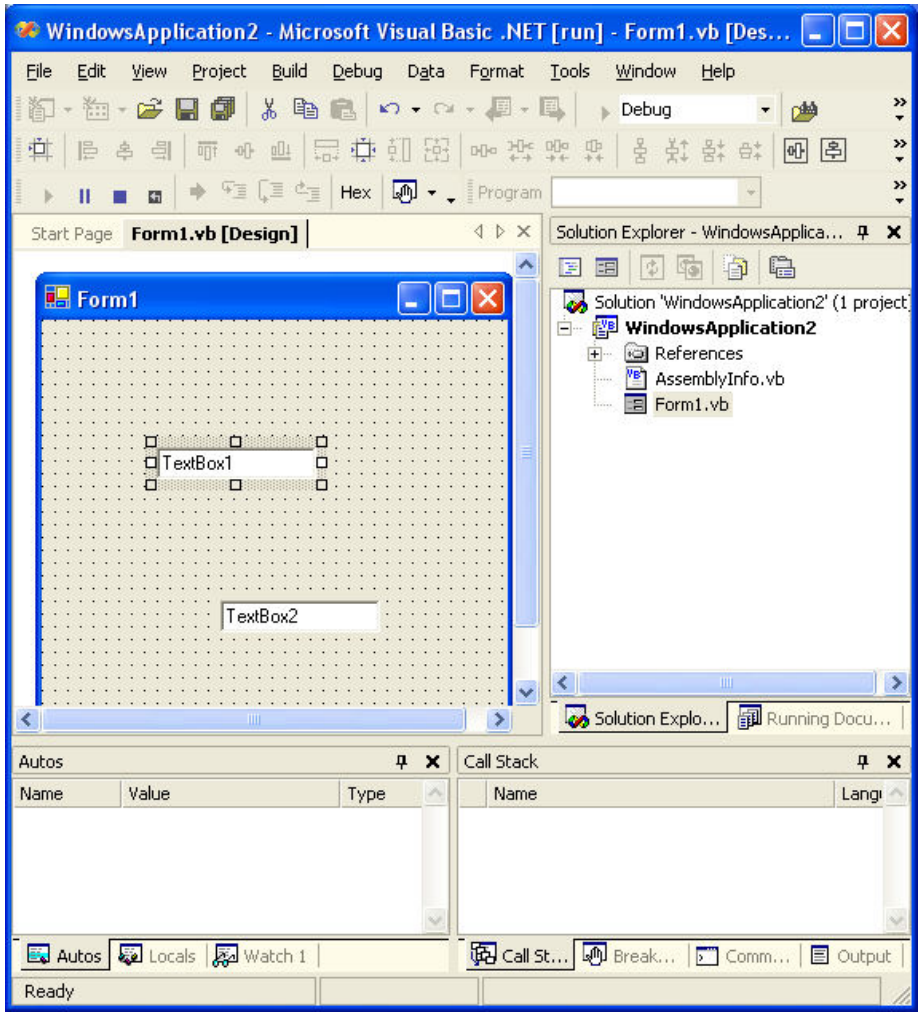
VISUAL BASIC

PRESENTATION BY KHUSHWAN SINGH

Plopping components on your form

- Either double clicking a component in the toolbox menu, or clicking the component in the toolbox then clicking on your form, will put a control on your form.
- Once there, you can "select" it, resize it, align it, or drag it to where you want it to go, or add other properties to it like tab order or a tooltip.

Here's a form with a couple of textboxes plopped on it



Some popular components

- Textboxes, buttons, and labels are the most popular components. Textboxes hold text - often user input.
- Labels are for labeling other components, usually.
- Buttons can be "pressed" to fire an event.
- Picture boxes can "hold" images
- comboboxes allow multiple choices.
- Listboxes are like multi-line textboxes, (Textboxes can also be set to be multiline).
- Radiobuttons and checkboxes display available choices: the user can select/deselect them

More on controls

- Controls can be grouped into groupboxes to help rationalize a complicated display.
- There are other types of controls as well – we won't learn how to use them all this semester.
- You are already familiar with many controls as a user of window applications.

Running your VB application

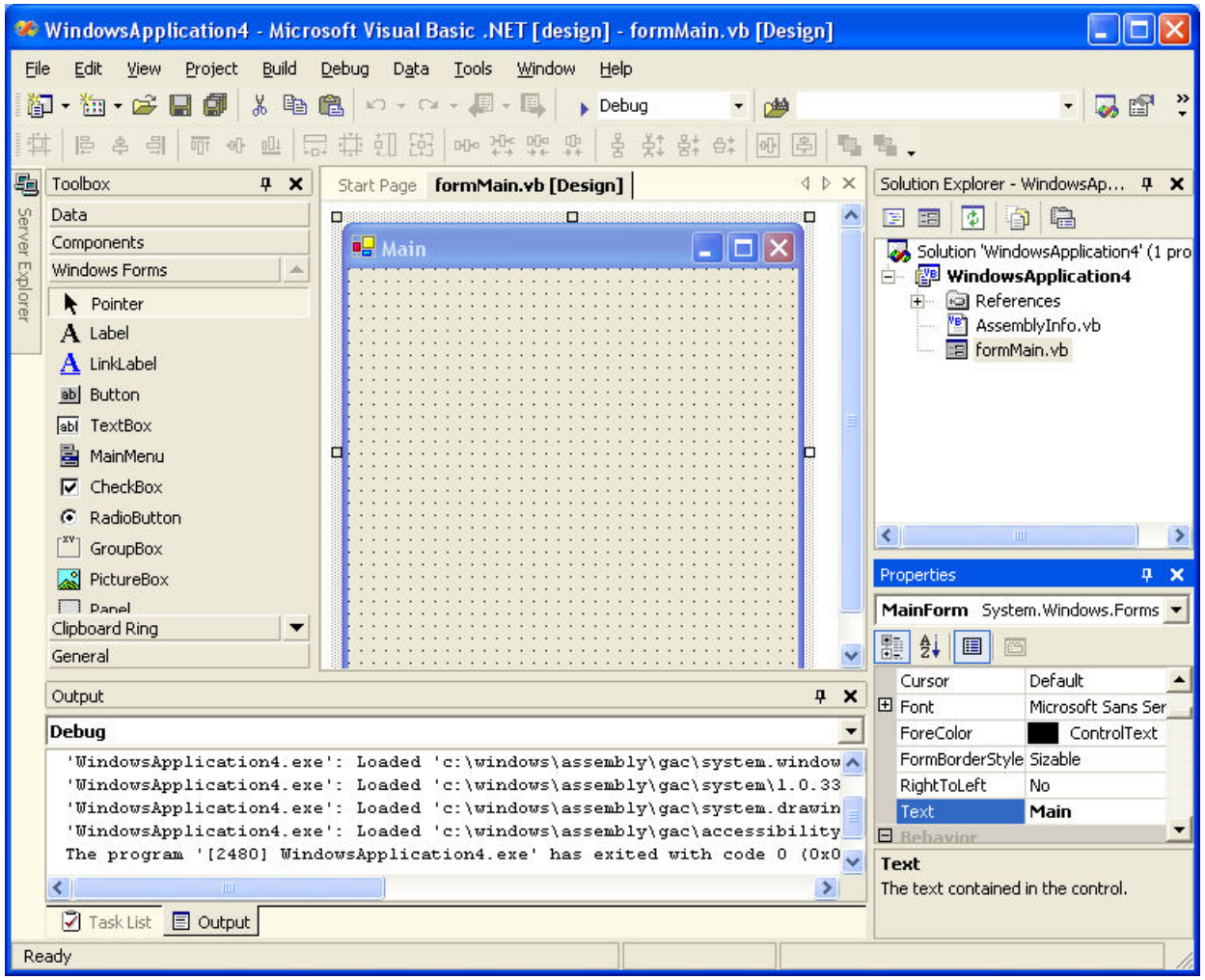
- At any time during development, as long as you have no errors, you can "run" your application.
- To check for errors: Select **build** from the menubar and then select **build solution (or rebuild)**
- To run or check for errors: Select **Debug** on the menu bar, and then pick **start** or press F5.

Running an application with two textboxes (there's no functionality)

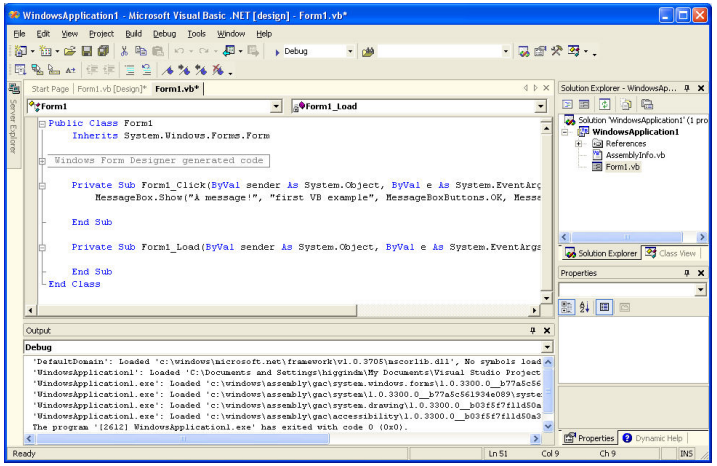
Note

- **You need to close your running application (window) before continuing development on it.** Just click the X in the upper right corner of the running form's window or, in the debug menu, select "**stop debugging"**.
- It is useful to "build" or "debug" periodically to make sure you have what you want and what you have works.

Selecting the form and editing the text property in the properties window allows you to change the text displayed on the form, its “name”, when the form comes up



More “basic” development: Let’s add functionality to a form



- Clicking on the blank form in the development window will cause a “code window” to pop up. You can provide code specifying the action to be taken when the form is clicked.

Events

- VB, VC++ and Java are event-driven languages.
- This means mouse-clicks or letters typed at the keyboard may "fire" (start, initiate, cause) events.
- When events are fired, the programmer can specify what is supposed to happen.

Subroutines

- Subroutines are the Basic program language name for programmer-specified functionality.
- They are referred to as "sub" in the VB code.
- VB helps you to write subs by providing stubs for any event-fired subroutine.
- This saves memorizing some things. It also saves typing and time.
- BUT: You must be careful: make sure the event sub which is stubbed in is the one you want.
- Cutting and pasting stubbed subs can be dangerous since some stubbed values may still need editing.

Our first vb sub

- Let's open a little message window when the user clicks anywhere on the form.
- In VB, messagebox is the name of the little message window component.
- Double-clicking on the form in development will switch us to a code window where a sub for this event-handler has been stubbed for us.

Form Click sub stub

- Below is the stub for form click.
- Be careful, as VB may stub in a sub for form load.

- In any case, you can edit the stub to look like this:

```
Private Sub Form1_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles MyBase.Click

End Sub
```

Our sub

```
Private Sub Form1_Click(ByVal sender As System.Object, ByVal e
    As System.EventArgs) Handles MyBase.Click
        MessageBox.Show("A message!", "first VB example",
    MessageBoxButtons.OK,
    MessageBoxIcon.Exclamation)
'comment…bold text is what you type
    End Sub
```

Remarks about this sub

- **Fit code on one line or use the space-then-underscore to continue a VB statement onto the next line.**
- **Important note: Most of these slides show code spilling onto multiple lines, which won't work.**
- **What you should type into the stubbed sub on one line is:**

```
MessageBox.Show("A message!", "first VB example",
    MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
```

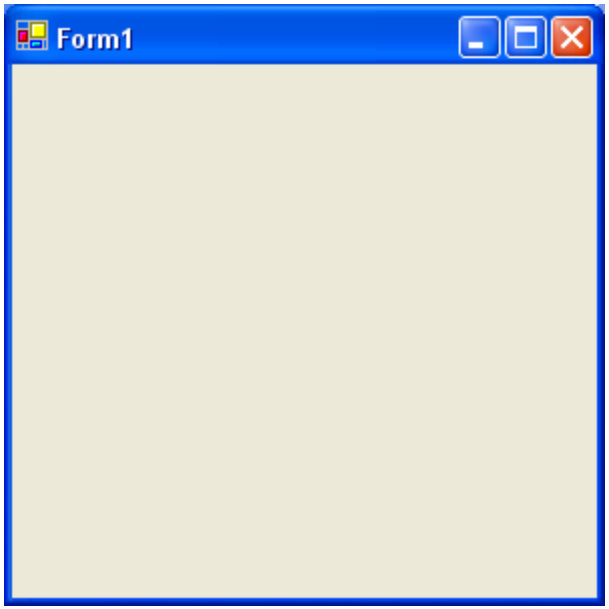
More remarks on this subroutine

```
Private Sub Form1_Click(ByVal sender As System.Object, ByVal e
    As System.EventArgs) Handles MyBase.Click
```

- Notice the name: Form1_Click. This is generated automatically, and would have specified a different name for the method if we had given our form a different name.
- In general, ComponentName_Click is the name of the subroutine handling mouseclicks on a control component named ComponentName.
- MyBase.Click is the event for clicking on the form.
- We'll learn more about the parenthetical arguments and the "Handles…" another time.

Now, run the example (remember: select **debug**, then click **start**)

An empty form appears, but…click on it



A message box pops up

Pull down the View options on the menubar

- Use the **toolbox** to select "tools" (called "controls" in VB) for your project
- Use the **properties** window(s) to set properties for your form and its control components.
- Use the **solution explorer** window to view the different elements of your solution.