Creating a VB2022 Windows Forms App

Follow these steps to create a simple application and use the basic settings of the Visual Studio 2022 IDE:

1. Open Visual Studio. If you are using it for the first time, you will be asked to specify how you will mostly use this program. Select Visual Basic. You will only need to do this once.

   From now on Visual Studio will start on the start page:

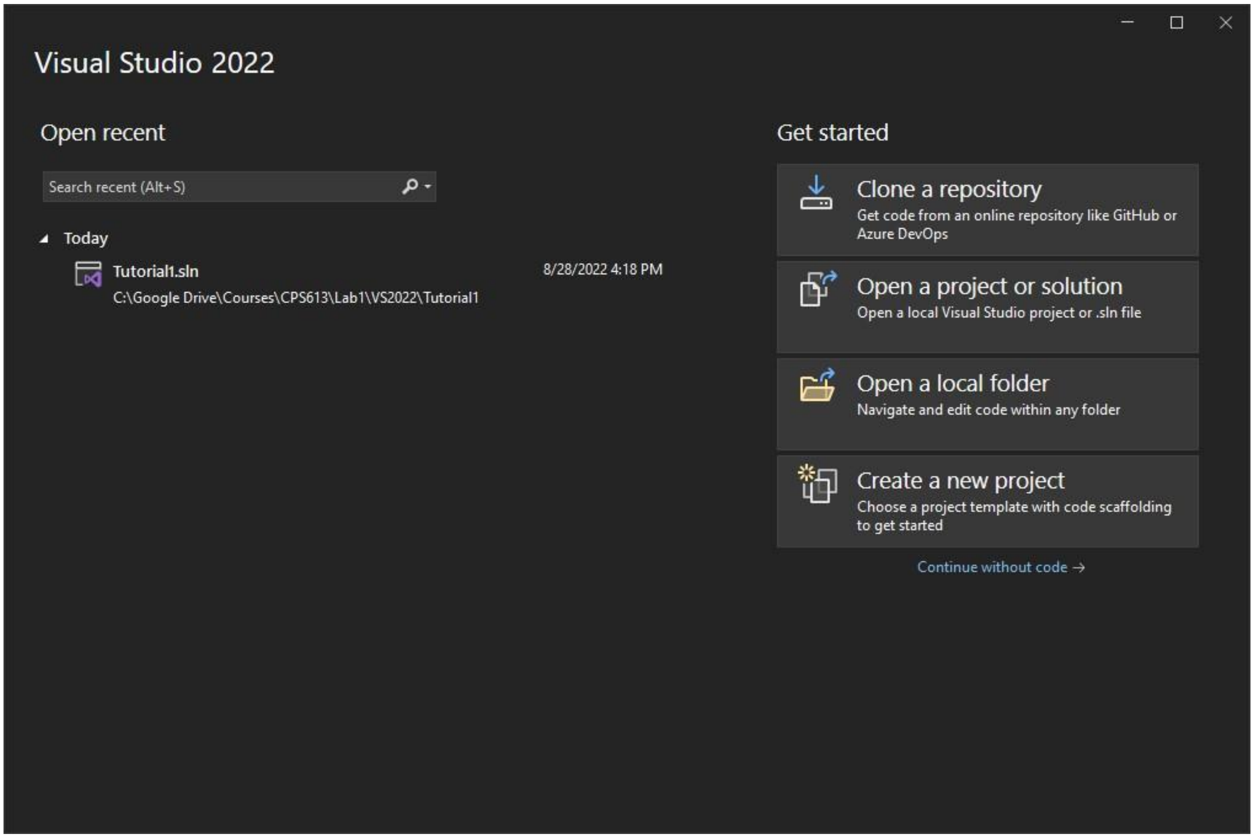
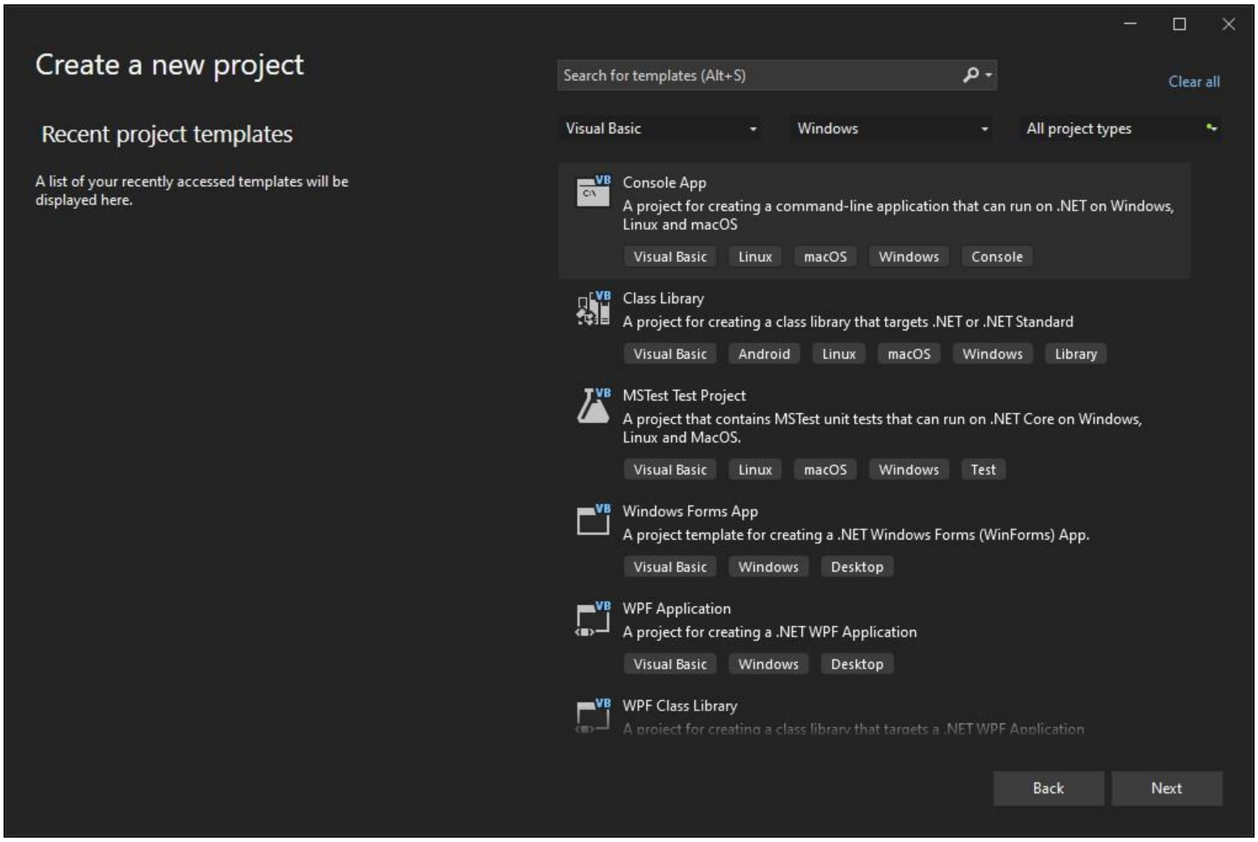

Visual Studio 2022
Open recent
Search recent (Alt+S)
Today
Tutorial1.sln
C:\Google Drive\Courses\CPS613\Lab1\VS2022\Tutorial1
8/28/2022 4:18 PM
Get started
Clone a repository
Get code from an online repository like GitHub or Azure DevOps
Open a project or solution
Open a local Visual Studio project or .sln file
Open a local folder
Navigate and edit code within any folder
Create a new project
Choose a project template with code scaffolding to get started
Continue without code →

FIGURE 1 The Visual Studio 2022 IDE.

2. Choose Create a new project. The Create a new project window will be displayed (Figure 2). In this lab, we will only be working in the Visual Basic language on the Windows platform. Select this language and that platform.



Create a new project

Recent project templates

A list of your recently accessed templates will be displayed here.

Search for templates (Alt+S)

Visual Basic | Windows | All project types

Console App
A project for creating a command-line application that can run on .NET on Windows, Linux and macOS
Visual Basic · Linux · macOS · Windows · Console

Class Library
A project for creating a class library that targets .NET or .NET Standard
Visual Basic · Android · Linux · macOS · Windows · Library

MSTest Test Project
A project that contains MSTest unit tests that can run on .NET Core on Windows, Linux and MacOS.
Visual Basic · Linux · macOS · Windows · Test

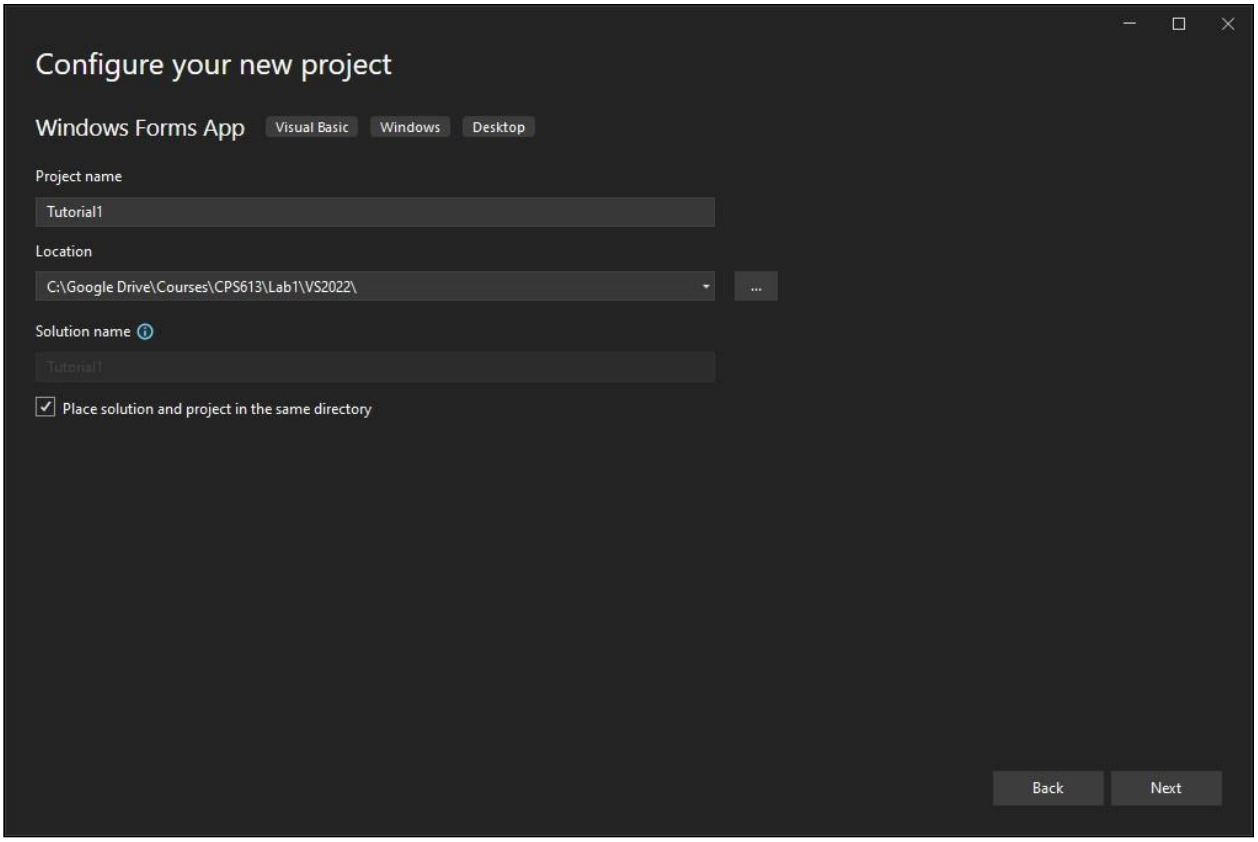
Windows Forms App
A project template for creating a .NET Windows Forms (WinForms) App.
Visual Basic · Windows · Desktop

WPF Application
A project for creating a .NET WPF Application
Visual Basic · Windows · Desktop

WPF Class Library
A project for creating a class library that targets a .NET WPF Application

FIGURE 2 Create a new project window in the IDE with Visual Basic language and Windows platform selected

3. This page displays a variety of standard application templates that can be used in Visual Studio .NET. Select Windows Forms App. This is the template that we will use for this course. The Configure your own project windows will be displayed (Figure 3).



Configure your new project

Windows Forms App

Project name: Tutorial1

Location: C:\Google Drive\Courses\CPS613\Lab1\VS2022\

Solution name: Tutorial1

☑ Place solution and project in the same directory

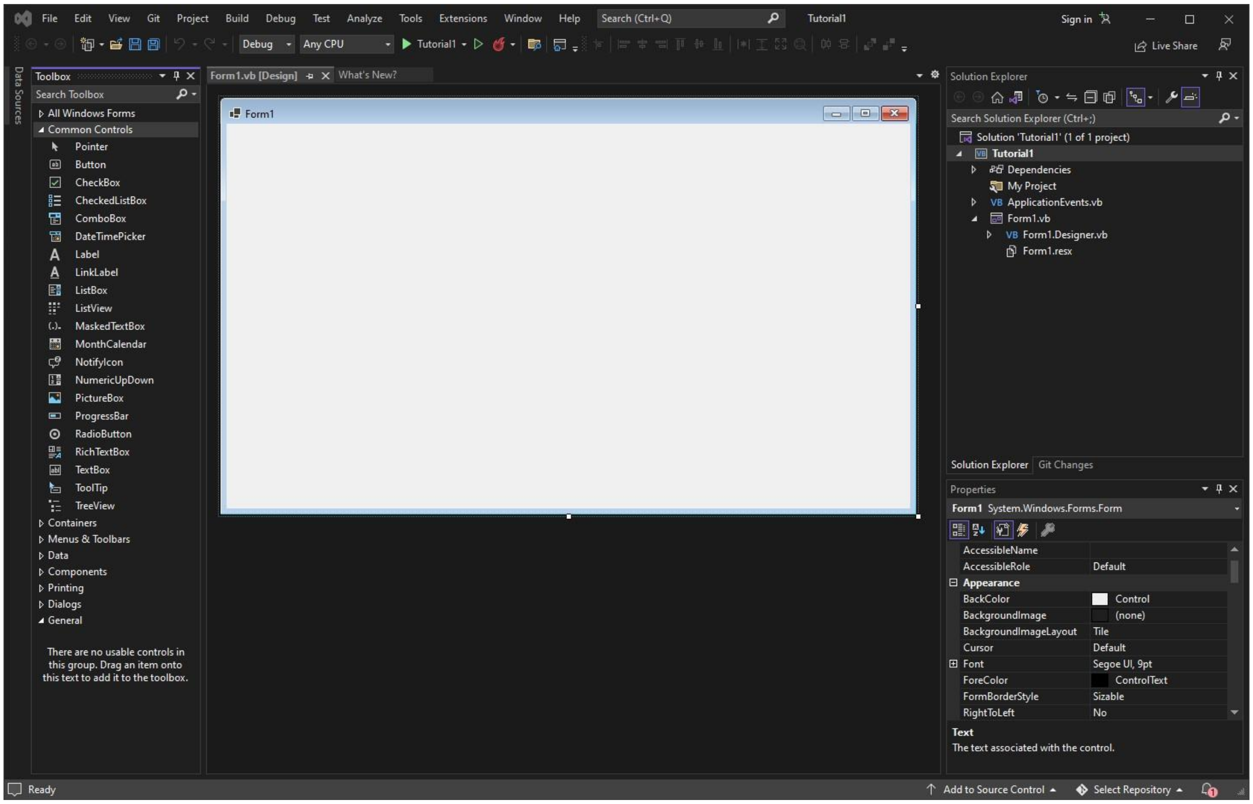
FIGURE 3 Configure your new project windows

4. Select a name for your project, for example "Tutorial1”.

5. Select the location where it will be stored. Choose your name and location carefully because Visual Studio does not have a “Save as” feature. If you want to move your solution to another directory this must be done from outside of the program.

6. Place the solution and the project in the same directory. (A solution can contain more than one project, but we will not be working with multiple project solutions in this course, so you can always check this option in this course.)

7. On the next page you will be asked for the target framework. Select .NET 6.0, then click the Create button. This will create a new project with a starting form (window) called “Form1” and you will now be in the Visual Studio IDE with one tab open showing the designer view of Form1. You will find it useful to pin the Toolbox (on the left) open and open its Common Controls menu (Figure 4)



Form1 System.Windows.Forms.Form

AccessibleName	
AccessibleRole	Default
Appearance	
BackColor	Control
BackgroundImage	(none)
BackgroundImageLayout	Tile
Cursor	Default
Font	Segoe UI, 9pt
ForeColor	ControlText
FormBorderStyle	Sizable
RightToLeft	No

Text
The text associated with the control.

FIGURE 4 New project Tutorial1 with pinned Toolbox and Common Controls menu open

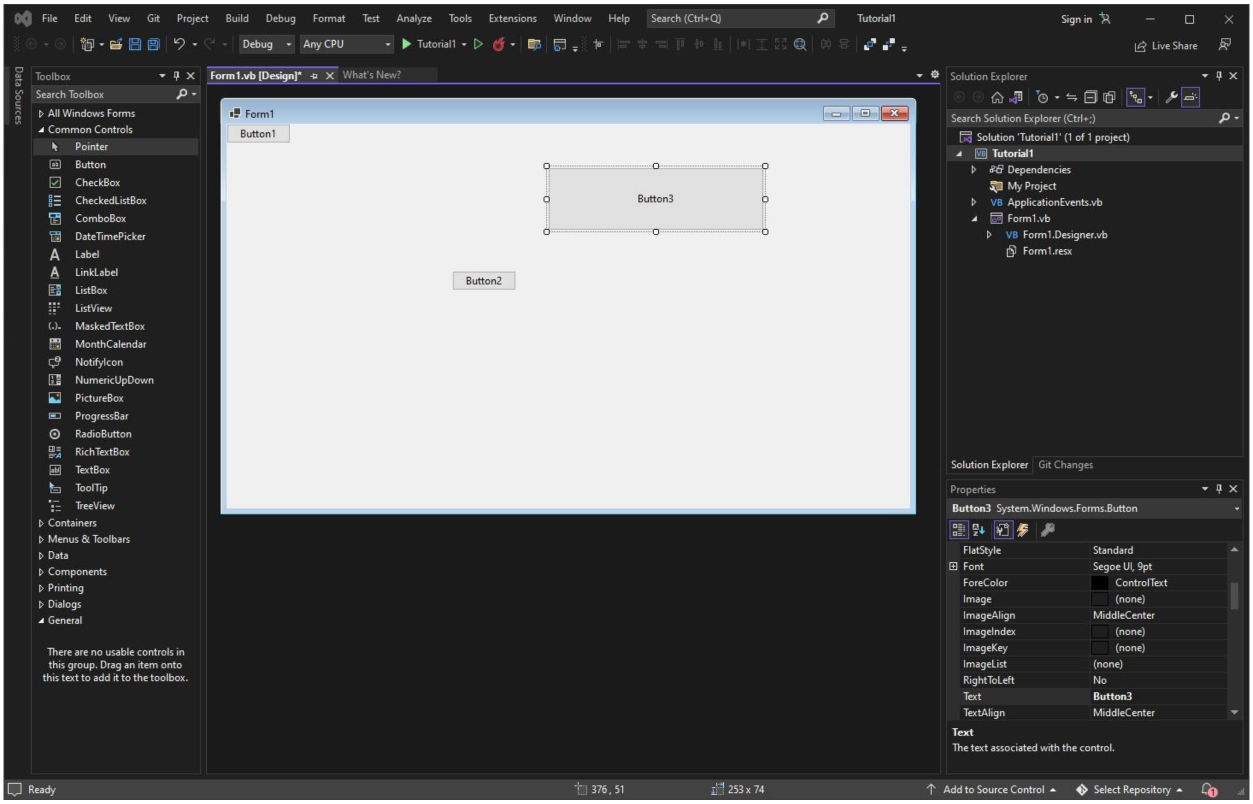
Adding Controls to a Form

Place Button controls from the Common Controls in the Toolbox onto Form1. There are 3 ways to do this:

1. Double-click the control; this will place it in the top-left corner of the form.

2. Drag the control from the toolbox onto the form.

3. Select the control in the Toolbox (single-click), and draw it on the form:
    - Move the mouse pointer to the Form window where the cursor changes to a crosshair.
    - Place the crosshair at the upper-left corner of where you want to position the control.
    - Press the left mouse button and hold it down while dragging the cursor toward the lower-right corner. Release the mouse button once the control has the right shape.



Button3 System.Windows.Forms.Button

FlatStyle	Standard
Font	Segoe UI, 9pt
ForeColor	ControlText
Image	(none)
ImageAlign	MiddleCenter
ImageIndex	(none)
ImageKey	(none)
ImageList	(none)
RightToLeft	No
Text	**Button3**
TextAlign	MiddleCenter

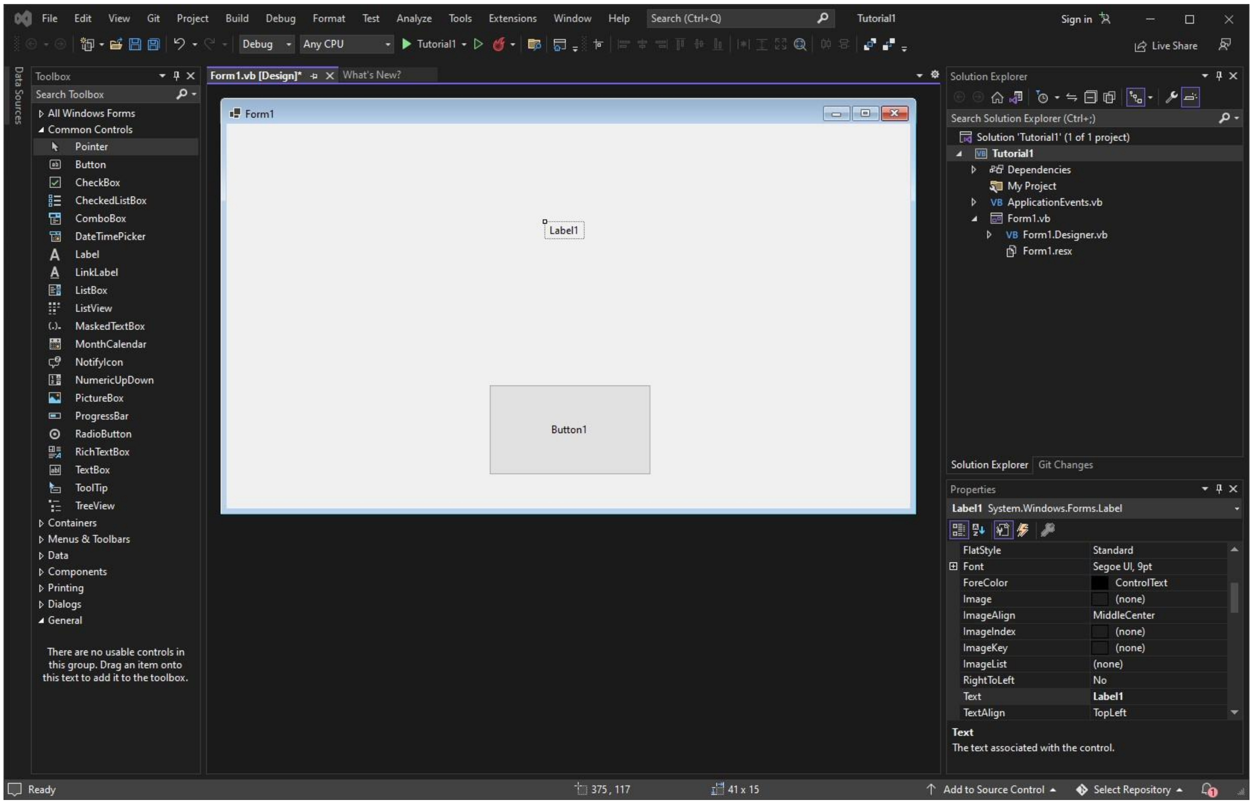
Text
The text associated with the control.

FIGURE 5   Placing Button controls on Form1

You can move and resize controls at any time during the development process. To move a control, click on it in the Form window and drag it to its new location. To resize a control, first click on it to select it. Then the sizing handles appear (see Button3 in Figure 5). These handles can then be clicked and dragged to resize the object.

Making the Application do Something

1. Keep the Button1 control and also place A Label control on Form1 as shown below::



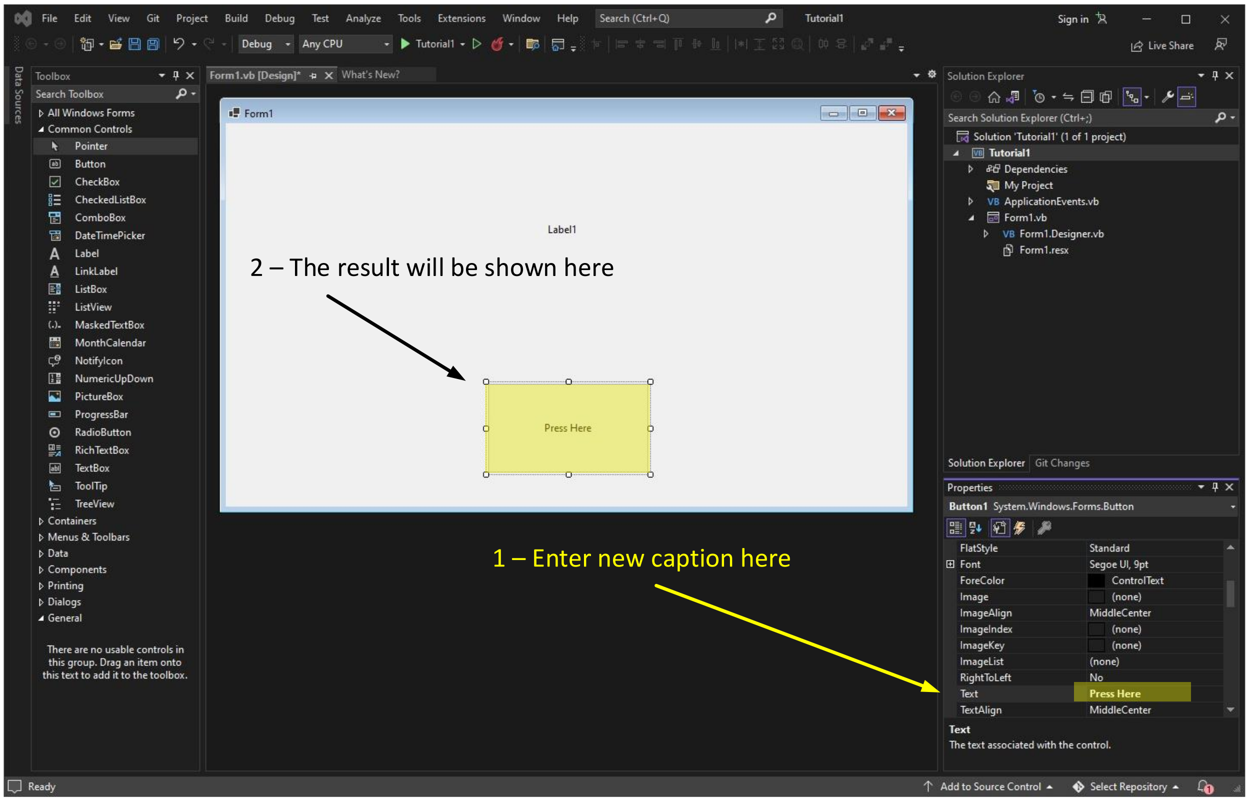
Label1 System.Windows.Forms.Label

FlatStyle Standard
Font Segoe UI, 9pt
ForeColor ControlText
Image (none)
ImageAlign MiddleCenter
ImageIndex (none)
ImageKey (none)
ImageList (none)
RightToLeft No
Text **Label1**
TextAlign TopLeft

Text
The text associated with the control.

FIGURE 6   GUI with label and button

2. In the Properties window (bottom right corner), change the Text property of the button to ***Press Here***.



2 – The result will be shown here

1 – Enter new caption here

Button1 System.Windows.Forms.Button

FlatStyle	Standard
Font	Segoe UI, 9pt
ForeColor	ControlText
Image	(none)
ImageAlign	MiddleCenter
ImageIndex	(none)
ImageKey	(none)
ImageList	(none)
RightToLeft	No
Text	Press Here
TextAlign	MiddleCenter

Text
The text associated with the control.

FIGURE 7 Changing the button caption to "Press Here"

3. Double-click the "Press Here" button. This will open a tab showing the code view of Form1 (Figure 8).

   In that window you will notice that Visual Studio has inserted a code stub for the event handler which responds to a click on Button1. The entire declaration is:

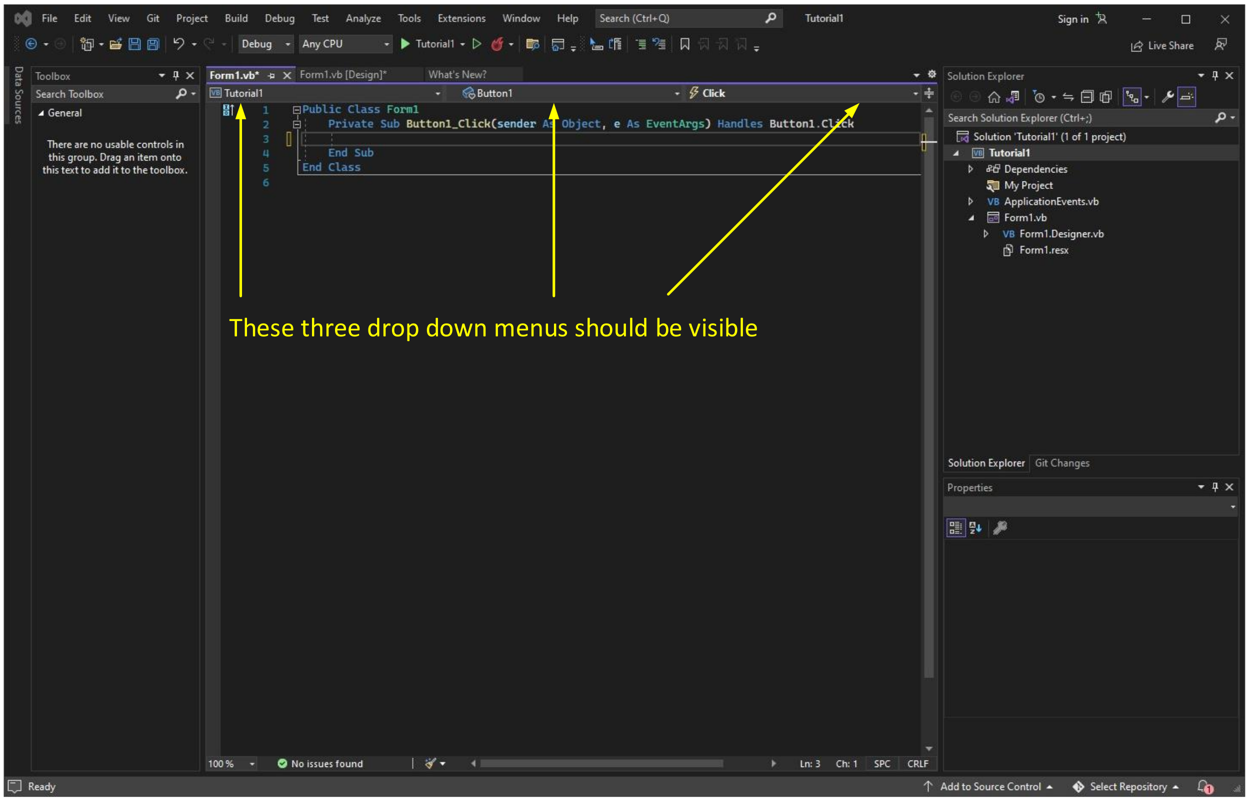
```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click

End Sub
```

   This event handler has been conveniently named "Button1_Click" by Visual Studio, but the part of the declaration which indicates that this routine is an event handler is the "Handles Button1.click" phrase.

4. Your cursor should be flashing beneath the *Private Sub Button1_Click(…)* line. Modify Button1_Click to add a function call to MsgBox with your own name as the argument:

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        MsgBox("Put your own name here")
End Sub
```



```
Public Class Form1
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click

    End Sub
End Class
```

These three drop down menus should be visible

FIGURE 8 Double-clicking on a VB object opens an event-driver for that object in the code window

5. Notice how on Figure 8 above there is a line with 3 drop down menus right between the tab label “Form1.vb” and the first line of code `Public Class Form1.` If you are missing these three drop-down menus, you can reset your IDE to show them by selecting Tools > Options… >Text Editor >All Languages > General and making sure that there is a check mark in the “Navigation bar” box.

These drop down menus are used to create specific event drivers for objects in projects

- The first (left) is used to select the project
- The second (middle) is used to select the object in this project to which an event will be applied
- The third (right) is used to select the event applied to that object.

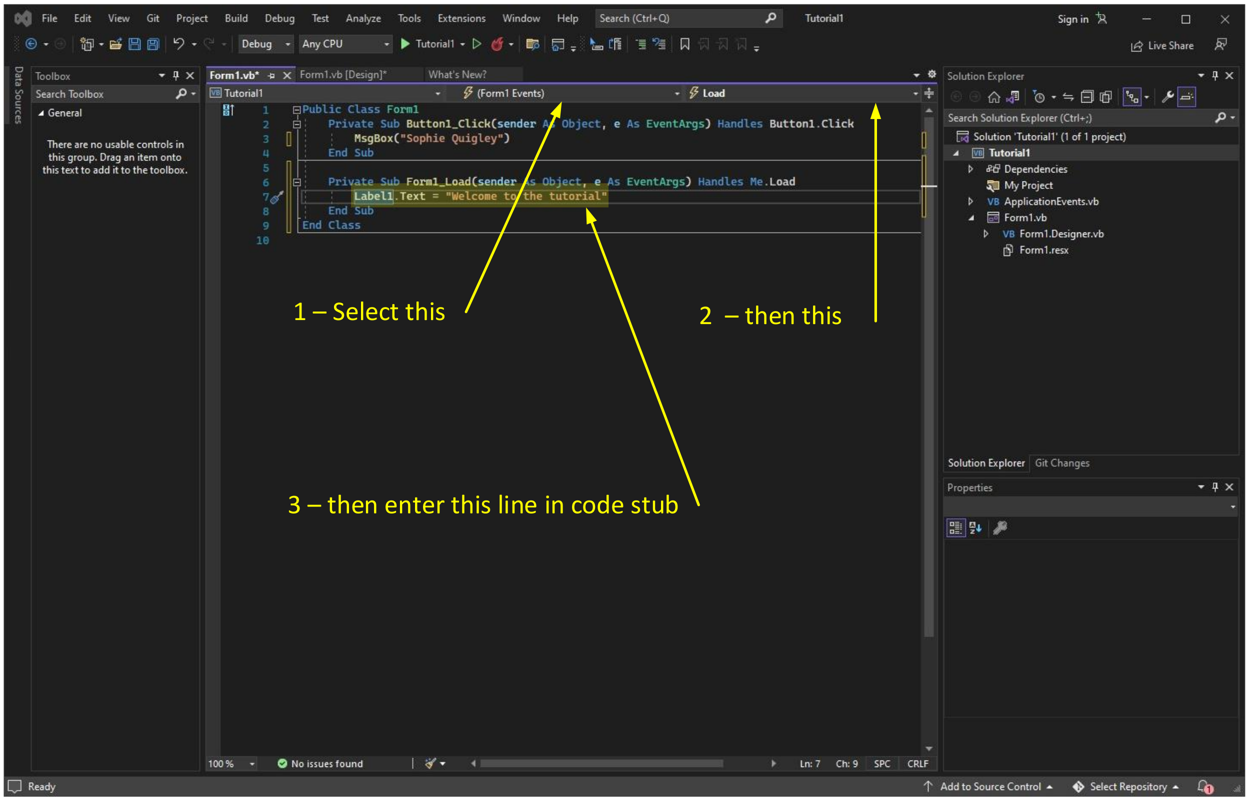
You should always use these menus to create event drivers because they will provide all valid events on all valid objects with their correct method signatures.

6. Select
   - The (Form1 Events) entry in the second drop-down to select the entire form
   - The Load event entry in the third drop-down to select the Load event which is the event invoked when Form1 is first run.

   This should now have created a code stub for the event driver called Form1_load that handles the Load event on the Form1 form (window).

7. Modify Form1_Load to change the text of the Label control when form 1 is loaded:

```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles Me.Load
        Label1.Text = "Welcome to the tutorial"
End Sub
```



```
Public Class Form1
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        MsgBox("Sophie Quigley")
    End Sub

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles Me.Load
        Label1.Text = "Welcome to the tutorial"
    End Sub
End Class
```

FIGURE 9 Creating and modifying Form1_Load

Running/Debugging the Program

You can execute the program from within Visual Studio either in debug mode or run mode. Both are available in the Debug menu and on the Toolbar underneath:

▶ starts the project in debug mode

▷ starts the project without debugging

Then, your project should run and you will see a window that looks like Figure 10

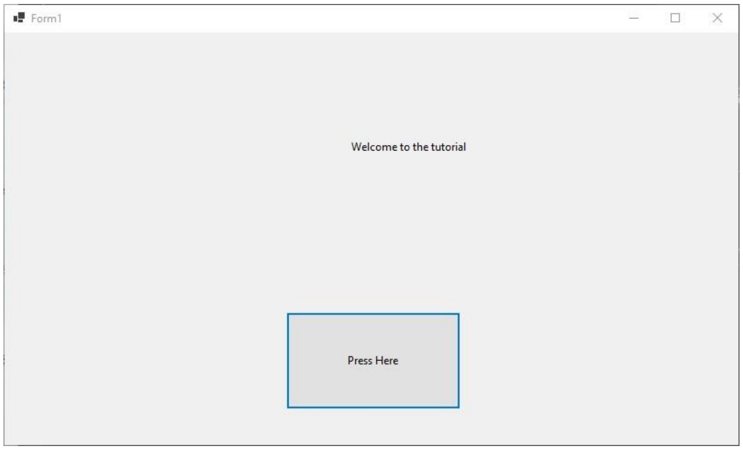


Figure 10 Your first program is running inside the IDE

Click on the Press Here button to see what happens.

You can close it like any Windows program or by selecting the Stop button ■ from within the Visual Basic IDE.

You've created your first Windows Forms program.

Now what?

If you want to save your project, be sure to do a File > Save all because your solution/project is a folder containing multiple files and subfolders.

You do not need to submit this tutorial. You are now ready to work on Lab1, which you will need to submit.