

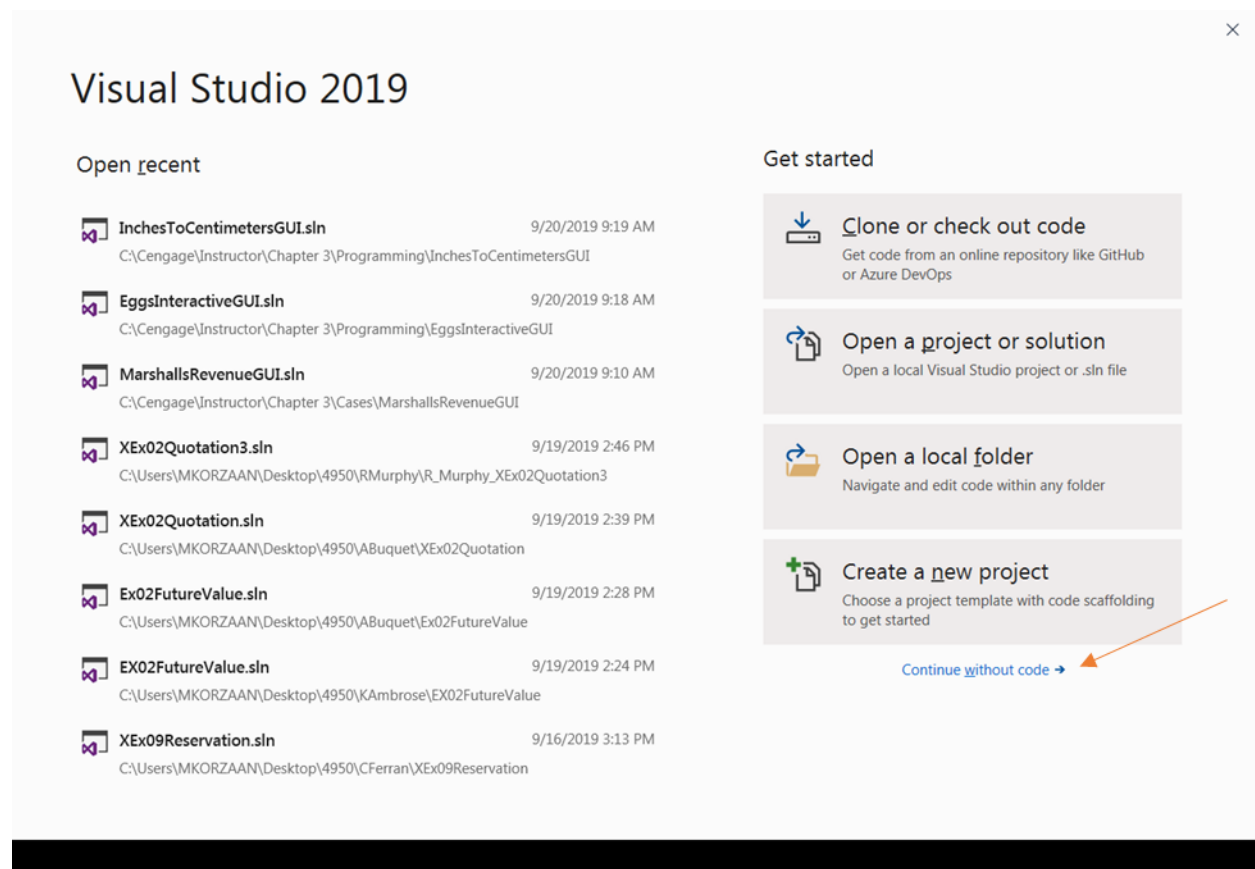
INFS 2600 Visual Studio Assignment 1

Working with the Visual Studio IDE

In the next steps, you use the IDE to create a `Form` with a `Button` and a `Label`. Your first console application in Chapter 1's "You Do It" section displayed "Hello, world!" at the command prompt. This application displays "Hello, Visual World!" on a `Form`.

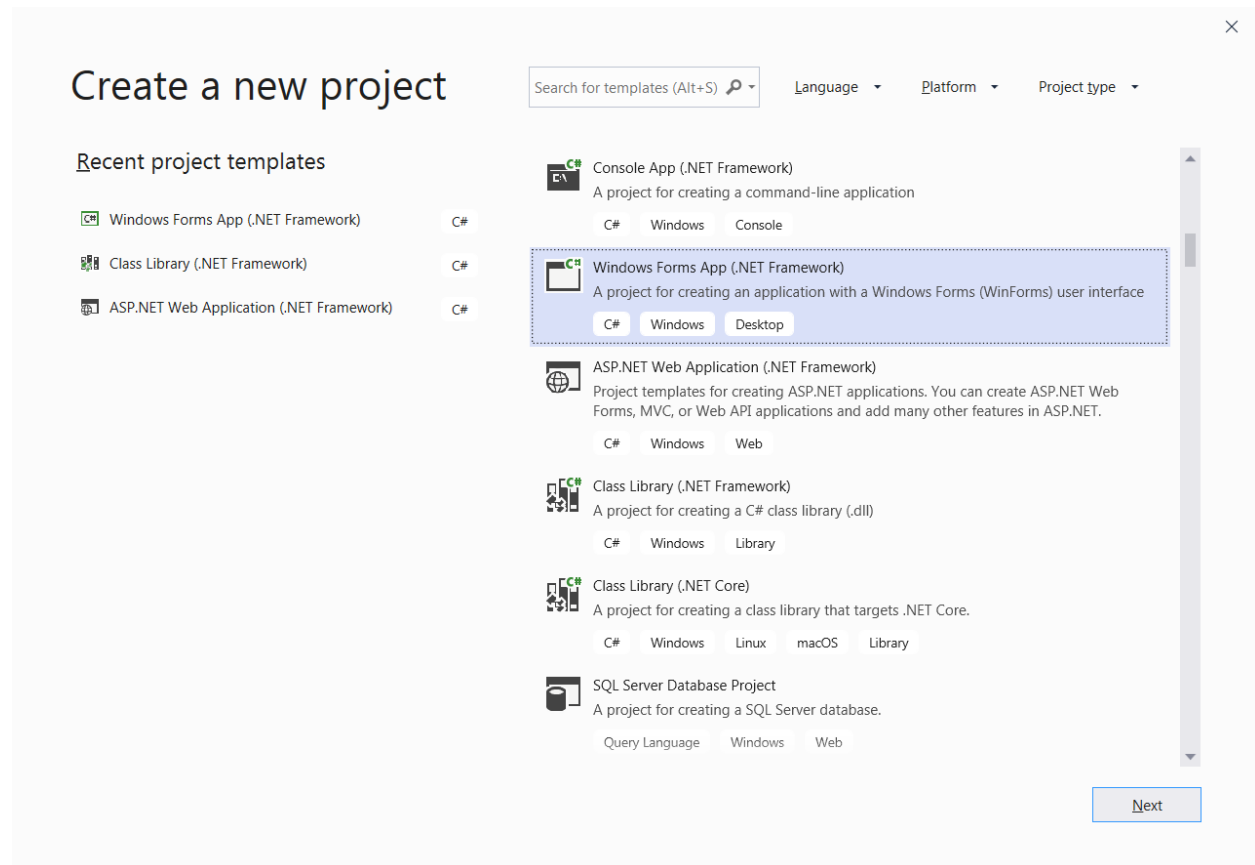
1.

Open Microsoft Visual Studio. You might have a desktop shortcut you can double-click, or in Windows 10 you can type the first few letters of **Visual Studio** in the "Ask me anything" search box, and then select it from the list of choices. Your steps might differ slightly from the ones listed here if you are using a different version of Visual Studio or a different operating system. If you are using a school network, you might be able to select **Visual Studio** from the school's computing menu. Once you open Visual Studio, if the initial display looks like the following image, then select the link in the lower right corner that says "Continue without code".



2.

Within Visual Studio, select **File** from the main menu, then **New** and **Project**. Then, in the New Project window, scroll through the options along the right side of the screen until you find **Windows Forms App (.NET Framework)**. Double check that the programming language is C# and not Visual Basic or F# (See the image below for a reference). Then click **Next**.



Instead of using the default project name, use **HelloVisualWorld**, and select a location to store the project on your computer or thumb drive (Notice the path specified in the Location on the screen print below as “C:\MTSU Classes\INFS 2600\”; in other words, in this example the HelloVisualWorld app will be stored in a folder named “INFS 2600” located within the “MTSU Classes” folder on the C:\ drive of the computer). Then click **Create**.

×

Configure your new project

Windows Forms App (.NET Framework) C# Windows Desktop

Project name

Location

...

Solution name ⓘ

☐ Place solution and project in the same directory

Framework

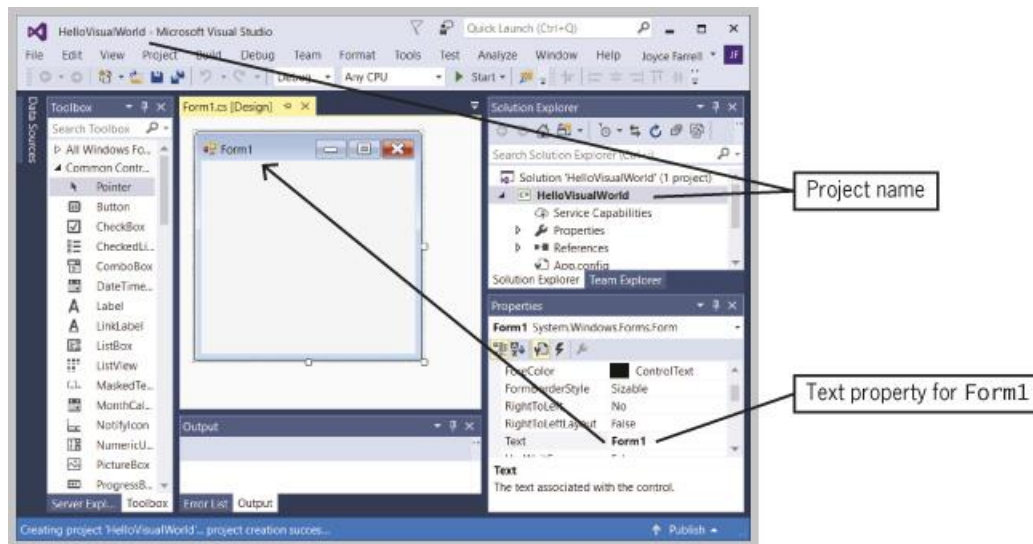
BackCreate

3.

The HelloVisualWorld development environment opens, as shown in Figure 3-16 in your textbook. The text in the title bar of the blank `Form` contains the default text `Form1`. If you click the `Form`, its Properties window appears in the lower-right portion of the screen, and you can see that the `Text` property for the `Form` is set to `Form1`. Take a moment to scroll through the list in the Properties window, examining the values of other properties of the `Form`. For example, the value of the `Size` property is 300, 300 by default.

Figure 3-16

The HelloVisualWorld project development environment



If you do not see the Properties window in the lower-right corner of your screen, click the title bar on the `Form`. Alternatively, press **F4** or click **View** in the menu bar and then click **Properties Window**. If you do not see the Toolbox shown in Figure 3-16, click the **Toolbox** tab at the left side of the screen, and then click the pushpin near the top to pin the Toolbox to the screen. Alternatively, click **View** from the menu bar and then click **Toolbox**. If you do not see the error list at the bottom of the screen, as in Figure 3-16, you can select **View** from the menu bar and then select **Error List**. You might have to drag up the divider that separates the error list from the window above it.

4.

In the Properties window, change the Name of the `Form` to **HelloForm**.

(The Name property is under Design if you choose to have the list categorized; if you choose to display it alphabetically, then Name is near the top.) Then change the Text of the `Form` to **Hello Visual World**. Press **Enter**; the title of the `Form` in the center of the screen changes to *Hello Visual World*.

5.

Examine the Toolbox on the left side of the screen. Select **Common Controls** if it is not already selected. In the Toolbox, click and hold **Button**. As you move your mouse off the Toolbox and onto the `Form`, the mouse pointer changes so that it appears to carry a Button. Position your mouse anywhere on the form, then

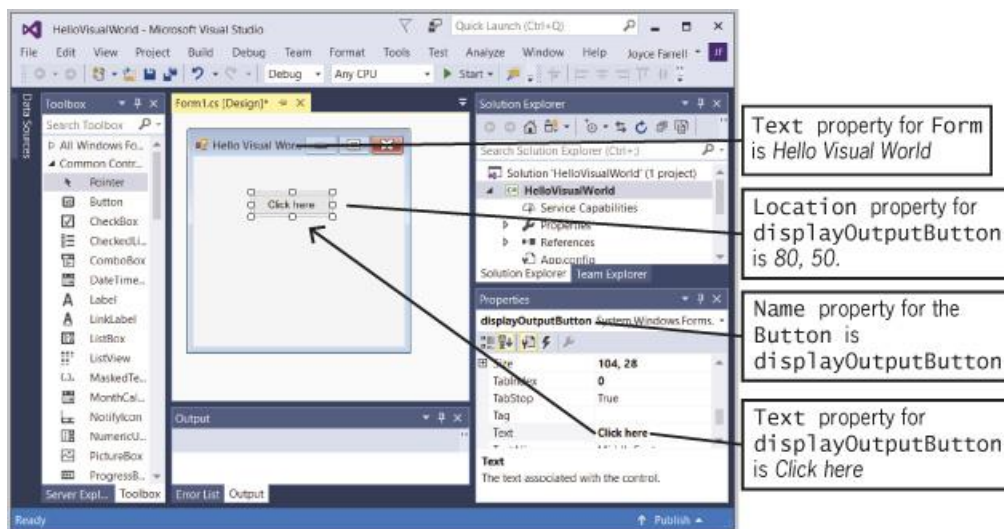
release the mouse button. The `Button` appears on the `Form` and contains the text `button1`. When you click the `Button`, handles appear that you can drag to resize it. When you click off the `Button` on the `Form`, the handles disappear. Click the `Button` to display the properties for `button1`. Change its Name to **displayOutputButton**, and change its Text property to **Click here**. When you press **Enter** on the keyboard or click anywhere on the `Form`, the text of the `Button` on the `Form` changes to *Click here*. You can adjust the size of the `Button` so you see its full text by dragging its handles.

6.

Scroll through the other `displayOutputButton` properties. For example, examine the `Location` property. The first value listed for `Location` indicates horizontal position, and the second value indicates vertical position. Drag the `Button` across the `Form` to a new position. Each time you release your mouse button, the value of the `Form` `Button`'s `Location` property is updated to reflect the new location. Try to drag the `Button` to `Location` 80, 50. Alternatively, delete the contents of the `Location` property field and type **80, 50**. The `Button` moves to the requested location on the `Form`. See Figure 3-17.

Figure 3-17

Changes to the `HelloVisualWorld` project



7.
Save your form by clicking **File** on the menu bar, then clicking **Save All**. Alternatively, you can click the **Save All** button on the toolbar; its icon is two overlapping disks.
8.
Although your `Form` doesn't do much yet, you can execute the program anyway. Click **Debug** on the menu bar, and then click **Start Without Debugging**. The `Form` appears. You can drag, minimize, and restore it, and you can click its `Button`. The `Button` has not yet been programmed to do anything, but it appears to be pressed when you click it. Click the `Form`'s **Close** button to dismiss the `Form`.
9.
You can close a project at any time and come back to it later. Exit Visual Studio now by clicking the **Close** button in the upper-right corner of the screen, or by clicking **File** on the menu bar and then clicking **Exit**. If you have made more changes since the last time you saved, you will be prompted to save again. When you choose **Yes**, the program closes.

Providing Functionality for a Button

In the next steps, you make the `Button` on the `HelloVisualWorld` `Form` functional; it will display a message when the user clicks it.

1.
Start Visual Studio. Select **File** from the menu, click **Open**, click **Project/Solution**, and browse for the **HelloVisualWorld** folder. Double-click it, and select the **HelloVisualWorld Microsoft Visual Studio Solution**. Alternatively, select **File** from the menu, click **Recent Projects and Solutions**, and select the project from the pop-up list.
2.
When the `Form` appears, drag a `Label` from the Toolbox to the `Form`. Change its Name to **helloLabel** and its `Text` property to **Hello, Visual World!**. Change the `Label`'s `Location` property to **80, 140**.
3.
Scroll through the Properties list for `helloLabel`, and change its `Visible` property from `True` to `False` by clicking to the right of *True*, clicking the down arrow, and selecting **False**. You can still see `helloLabel` in the Designer, but the `Label` will be invisible when you execute the program.

4. Double-click the `Button` on the `Form`. A new window that contains program code is displayed, revealing a newly created `displayOutputButton_Click()` method with no statements. Between the curly braces, type the following statement that causes the `Label` to appear when the user clicks the `Button` on the `Form`:

```
helloLabel.Visible = true;
```

5. Change the name of the `displayOutputButton_Click()` method to use conventional Pascal casing by right-clicking the method name, selecting **Rename**, and changing the highlighted name to start with an uppercase **D**. Click **Apply**.
6. Save the file, then run the program by clicking **Debug** on the menu bar and clicking **Start Without Debugging**, or by pressing **Ctrl+F5**. When the `Form` appears, click the **Click here** button to reveal the greeting. If you want, experiment with values for some of the `Label` properties, such as `Font`, `BackColor`, and `ForeColor`.

Adding a Second Button to a Form

In the next steps, you add a second `Button` to the `Form` in the `HelloVisualWorld` project.

1. In the IDE, switch to Design View. Drag a second button onto the `Form`. Make its `Location` property **80, 195**. Change the new `Button`'s `Name` property to **changeOutputButton** and change its `Text` to **Click me last**. Drag the button's handles to adjust the size so all the text is visible.
2. Double-click the `changeOutputButton` to expose the `changeOutputButton_Click()` method. Rename the method to use Pascal casing.
3. Between the curly braces of the method, add the following code:

```
helloLabel.Text = "Goodbye";
```

- 4.

Save the project, then execute the program. Click the `displayOutputButton` to reveal *Hello, Visual World!*. Then click the `changeOutputButton` to reveal *Goodbye*.

5. Execute the program again. This time, click the second `Button` first. No message appears because the `Label` with the message has not been made visible. When you click the `displayOutputButton`, the `Label` becomes visible, but it displays *Goodbye* because the `changeOutputButton`'s `Click()` method has already changed the `Label`'s `Text`.

6. Return to the `Form1.cs` [Design] tab. Click the `changeOutputButton`, and change its `Enabled` property to `False`. Now, when the program executes, the user will not be immediately able to click the second `Button`.

7. Double-click the `displayOutputButton`. Add the following statement to the `Click()` method so that the `changeOutputButton` becomes enabled when the `displayOutputButton` is clicked:

```
changeOutputButton.Enabled = true;
```

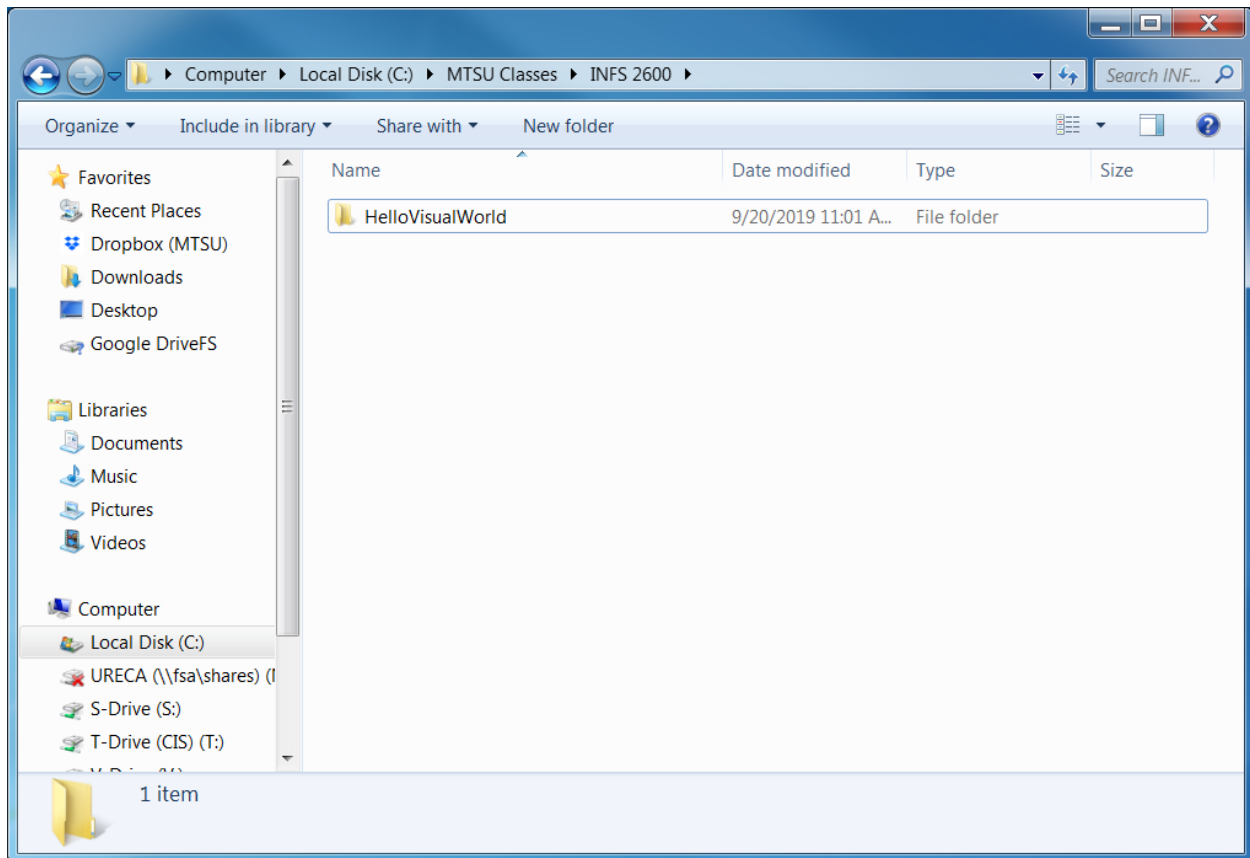
8. Save the project, then execute the program. When the `Form` appears, the `changeOutputButton` is dimmed and not clickable because it is not enabled. Click the enabled `displayOutputButton` to reveal the *Hello, Visual World!* message and to enable the second `Button`. Then click the `changeOutputButton` to expose the *Goodbye* message.

9. Dismiss the `Form`, and close Visual Studio.

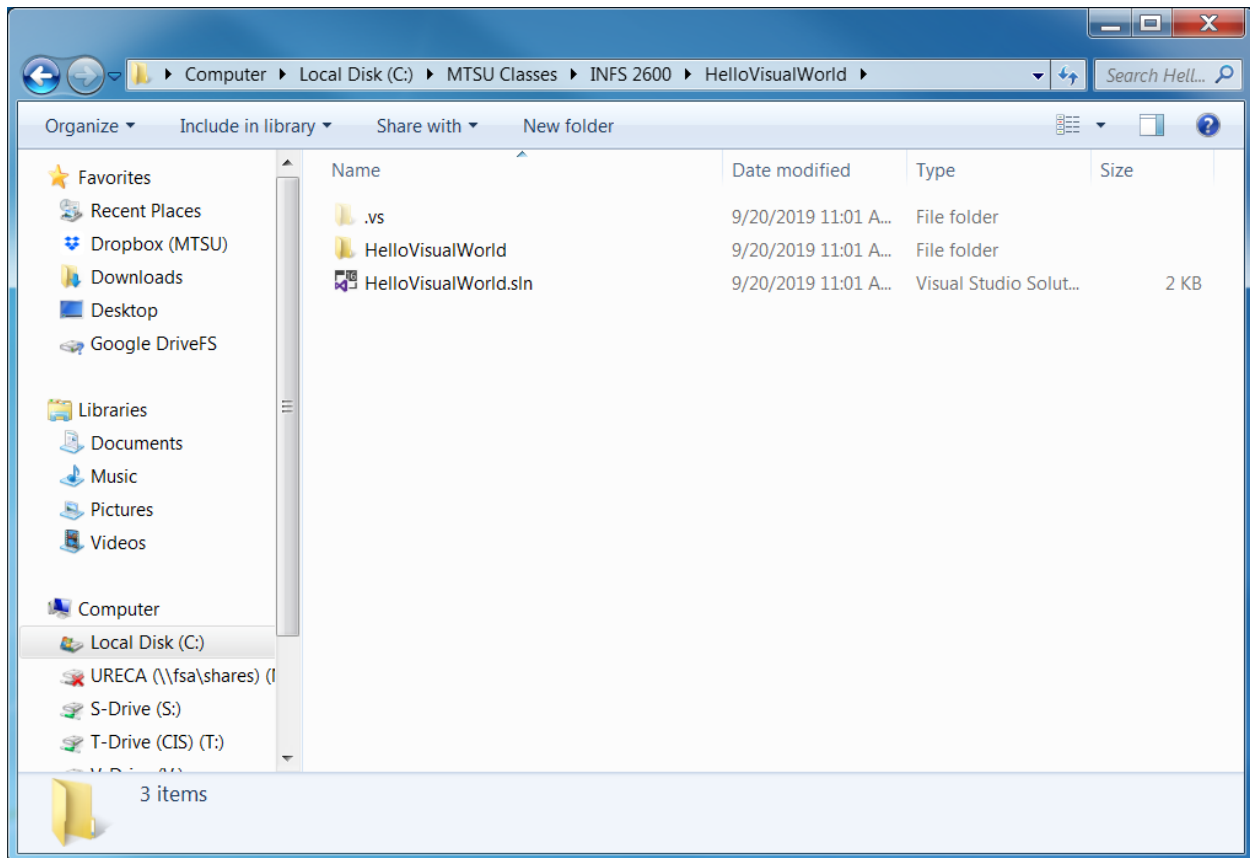
Turn in your app to [D2L Dropbox](#) (compress your app at the root folder level)

You will need to go to the File Explorer (or Windows Explorer) and locate where your application is saved on your computer (note: you specified this location back at the beginning in Step #2 when you were creating the application). In this example, I saved the application in the "INFS 2600" folder located in the "MTSU Classes" folder on the C:\ drive of my computer. In other words, the app is located in the following path C:\MTSU Classes\INFS 2600\. In the screen print

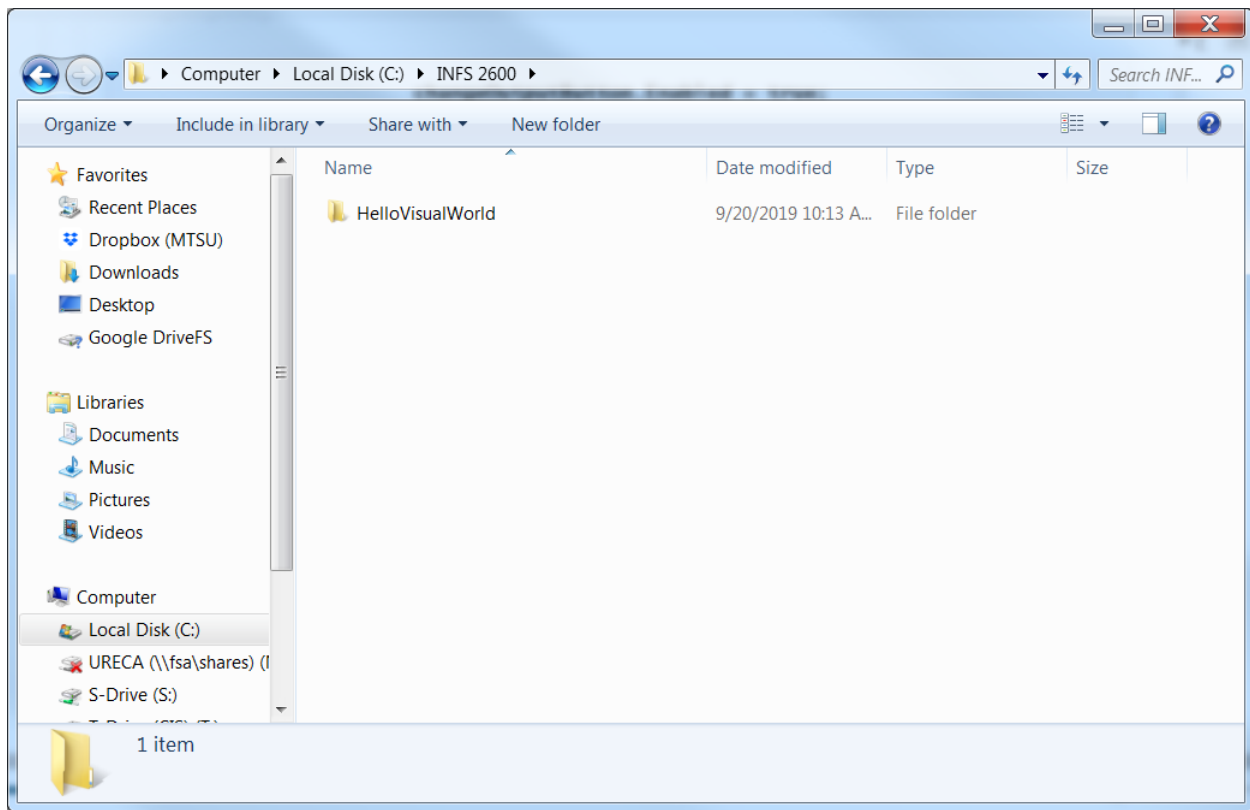
below, you can see the name of the application located in the INFS 2600 folder. The image below shows the root level of the application.



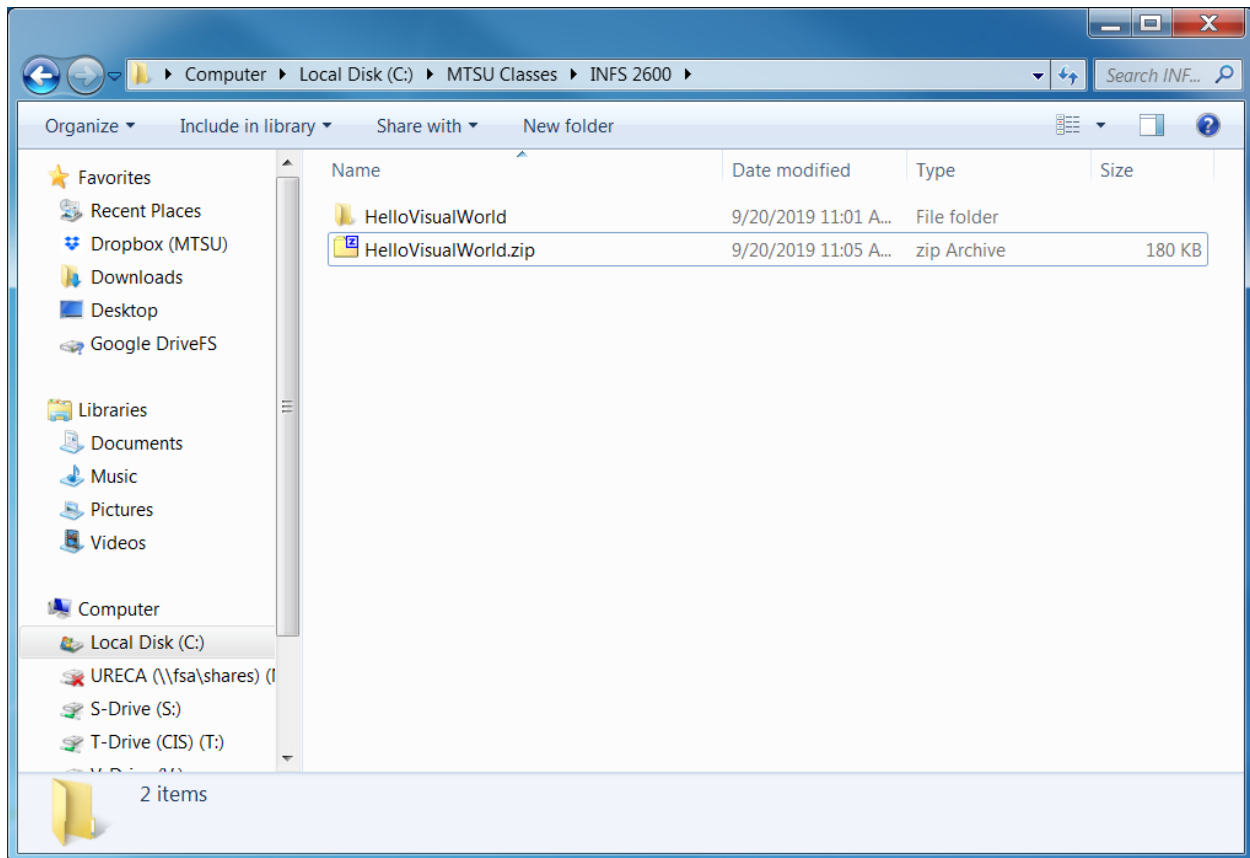
If you double click on the HelloVisualWorld root level folder, it will show you the subfolders and files as shown below in the following image. You can see there is a subfolder for HelloVisualWorld and a file named HelloVisualWorld.sln. You may also see a .vs subfolder (this is a hidden folder, so depending on the settings of your computer the .vs subfolder may or may not be visible). It is not a problem if the .vs subfolder is not visible.



Now that you have viewed the contents, go back up to the root level by returning to C:\MTSU Classes\INFS 2600\. The root level is shown again in the image below.



Next, right click on HelloVisualWorld here at the root level and select “Send to” then “Compressed (zipped) folder”. Your screen should now look like the image below. It will include the original HelloVisualWorld folder along with the compressed HelloVisualWorld.zip folder.



The last step before uploading your assignment to the D2L dropbox is to right click on the HelloVisualWorld.zip file and rename it to include your FirstInitial_LastName_ in front of HelloVisualWorld. In other words, the compressed file should be named as follows:

YourFirstInitial_YourLastName_HelloVisualWorld.zip

For me, I would rename the file to match my own first initial and last name such as:

M_Korzaan_HelloVisualWorld.zip

Then, upload this compressed folder to the appropriate Dropbox folder in D2L.

