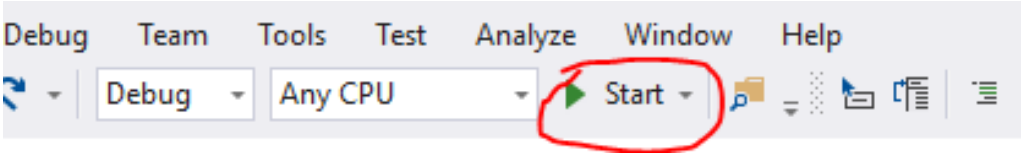
Orthoscope/Arthroscope Software Development Kit – Getting Started

Installation

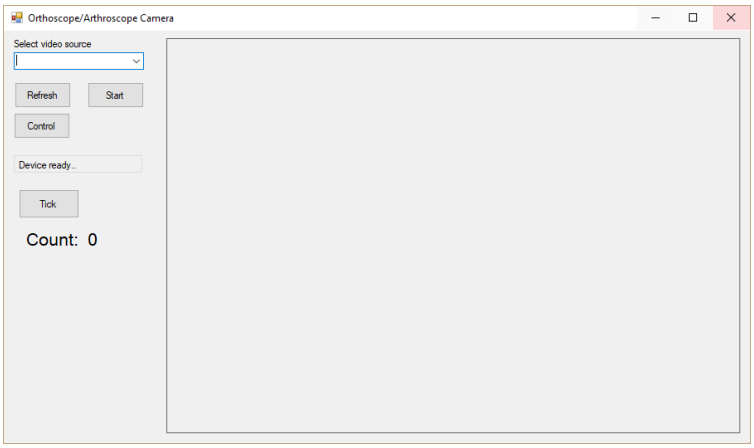
1. Download the `elec371_proj.zip` file
2. Unzip `elec371_proj.zip` into a destination folder.
3. Open the `cam_aforge1` folder and double click the `scopeGUI.sln` file to open Visual Studio 2015 or 2017.

Start the Application

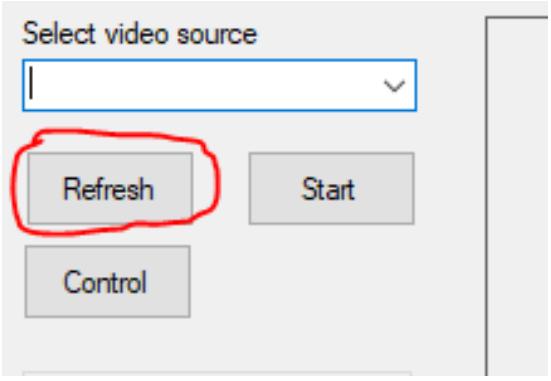
1. Plug the Camera into a USB port.
2. On the upper side of the Visual Studio window, click the Start button.



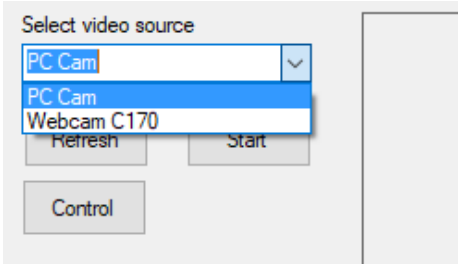
3. The application will compile and run.



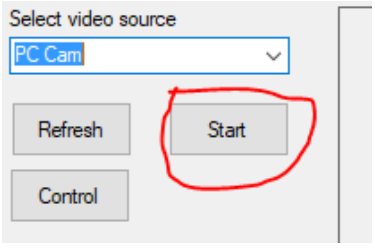
4. Click refresh to update the video sources.



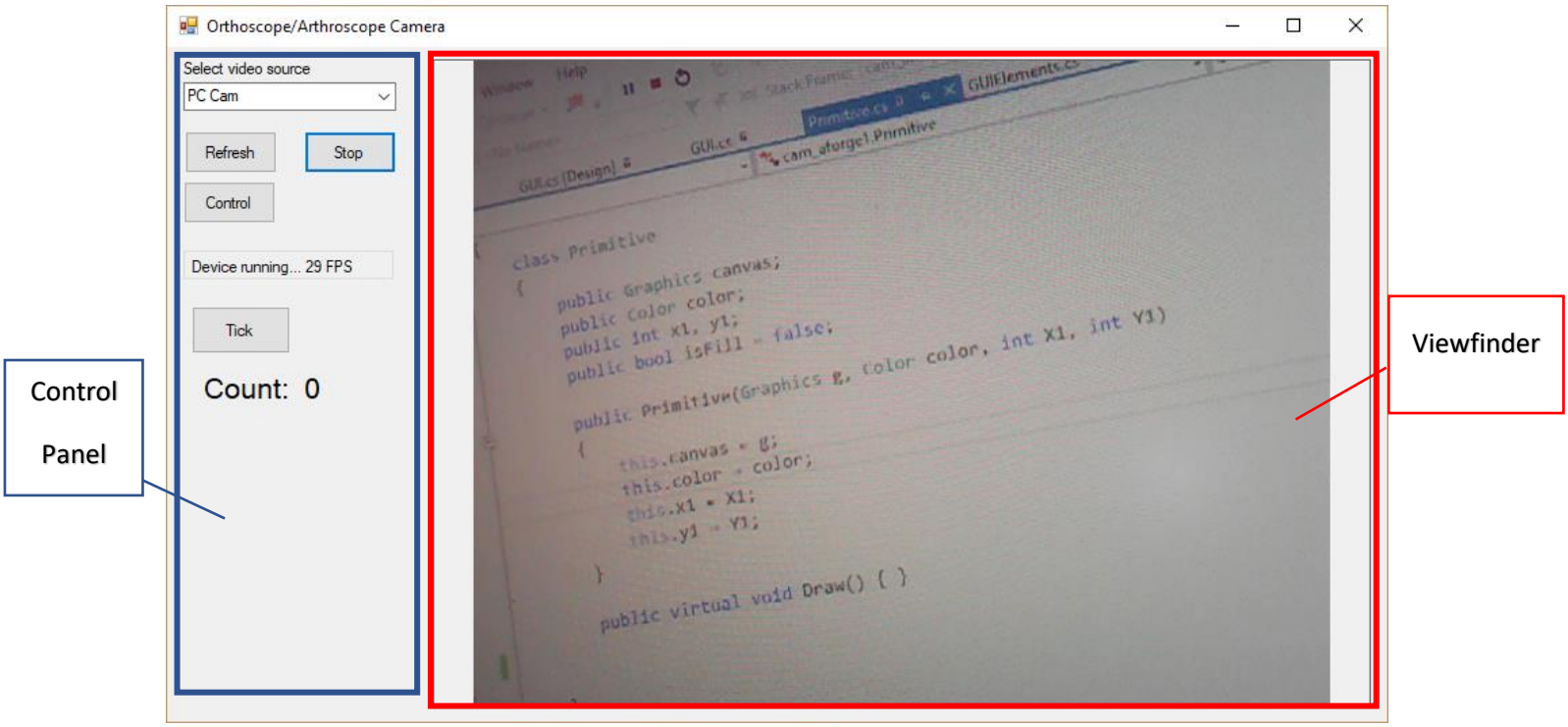
5. Select a source from the dropdown menu. Choose “USB 2.0 Camera” or “PC Cam”. Note that your system may list the Camera under a different name.



6. Click Start



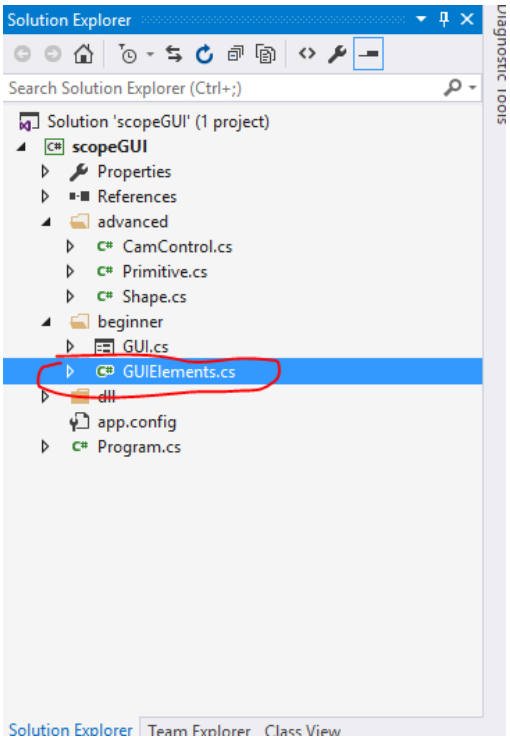
7. The camera feed is displayed.



8. This verifies that everything is working. Close the Orthoscope/Arthroscope Camera window.

Adding Elements to the Viewfinder

1. On the right side of the screen, under the Solution Explorer tab, double click beginner>GUIElements.cs



2. The GUIElements class has been populated with sample code that is commented out. Follow Steps 0-6 for a walkthrough on how to add Primitives, animate them, and group them into Shapes. To uncomment code, delete the double slashes (//) before a line, or delete the pair of slash-asterisks (/* and */) enclosing a code block.

```
GUIElements.cs
scopeGUI                                   cam_aforge1.GUIElements
11      {
12          GUI gui;
13          public Graphics g;
14
15          //This is where you can declare variables that you will be changing as the Run()
16          //method executes.
17
18          //Step 0: To try out the sample code, uncomment all the variables from line 19-25
19          //int circleX1 = 50;
20          //int circleY1 = 50;
21          //int squareX1 = 0;
22          //int squareY1 = 0;
23          //int speed = 5;
24          //bool cirDir = true;
25          //bool sqrDir = true;
26
27          int counter = 0;
28
29          //End Variable declaration.
30          public GUIElements(GUI _gui)
31          {
32              this.gui = _gui;
33          }
34
35          public void Run()
36          {
37              //Step 1: Let's draw a basic Square. Uncomment Lines 38-39 to draw a blue square.
38              //Square sqr = new Square(g, Color.Blue, 4, squareX1, squareY1, 100);
39              //sqr.Draw();
40
41              //Step 2: Now let's draw a filled circle. Uncomment Lines 42-45 to draw a purple
42              //circle in the centre of the square we drew in step 1
```

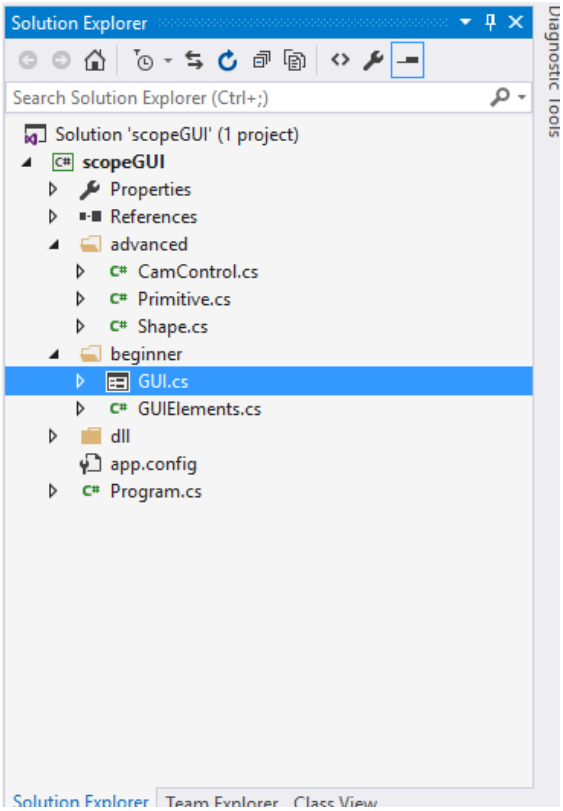
3. During the process of initializing a primitive, a help message box that describes each essential parameter will be displayed. Consult the Cheat Sheet for a complete list of constructors.

```
Square sqr = new Square()
//sqr.Draw();
//Step 2: Now
//circle in the centre of the square we drew in step 1
```

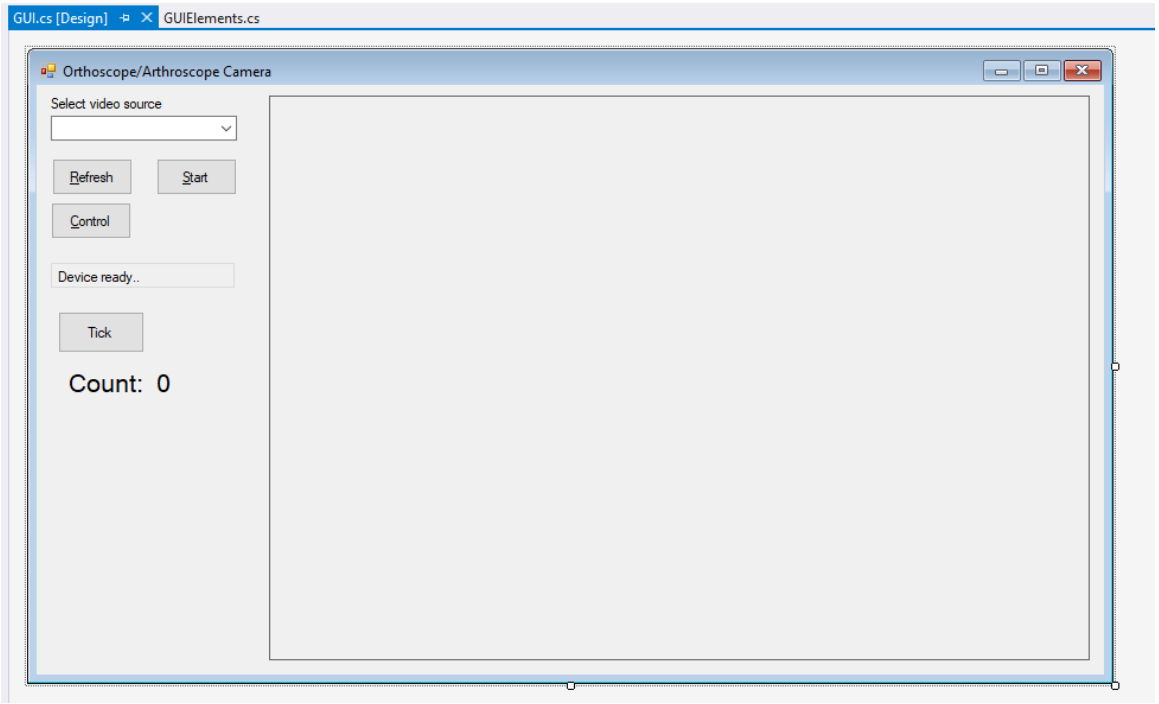
Square(Graphics g, Color sqrColor, int sqrThickness, int sqrX1, int sqrY1, int sqrSize)
Initializes a Square Object
g: The graphics object to be drawn on. ALWAYS PASS ON g IN THIS PARAMETER

Adding Buttons to the Control Panel

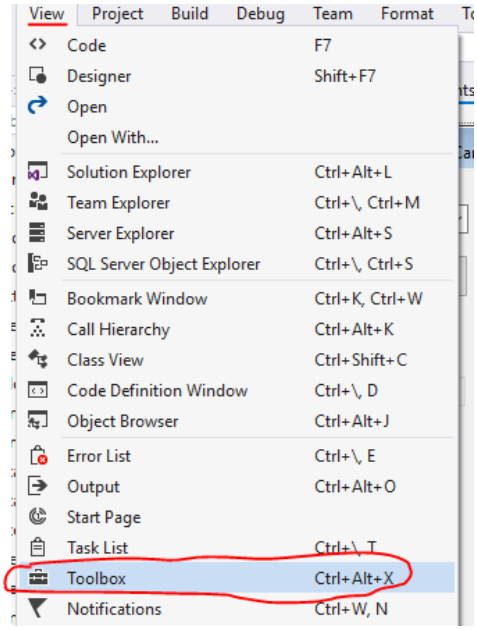
1. On the right side of the screen, under the Solution Explorer tab, double click beginner>GUI.cs



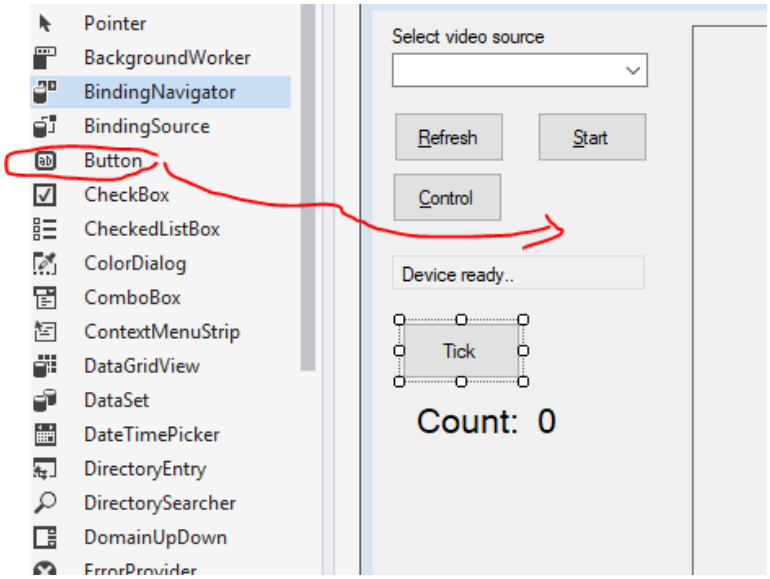
2. The Visual Studio Form Editor opens.



3. Open the Visual Studio Toolbox either by pressing Ctrl+Alt+X or by clicking View>Toolbox



4. To add a button, drag and drop the Button component from the toolbox into the Form Editor.



5. To add functionality to a button, double click the Tick button in Control Panel area in the form editor. This will open a GUI.cs tab in your editor (note that this is different from the GUI.cs [Design] tab). You will be taken directly to the corresponding method call, which is button3_Click in this case.

GUI.cs | GUI.cs [Design] | GUIElements.cs

scopeGUI | cam_aforge1.GUI

```
127             CamControl.show_Controls();
128         }
129
130         private void button3_Click(object sender, EventArgs e)
131         {
132             //Step 7: Let's activate this button. Uncomment lines 142-143 to add functi
133             //to the button.
134             //tickCount++;
135             //countDisp.Text = tickCount.ToString();
136
137             //Step 8: The buttonX_Click method can be used to call a method that you wr
138             //on the GUIElements Class. Uncomment line 148 to call the ButtonWasClicked
139             //from GUIElements.
140             //myCanvas.ButtonWasClicked();
141
142             //Note that the syntax for calling methods in the GUIElements class is alwa
143             //`myCanvas.NameOfMethod();`.
144         }
145     }
146 }
```

6. The button3_Click method in GUI.cs has been populated with sample code that demonstrates the adding of interactivity for this application. Follow Steps 7 and 8 to add functionality to the Tick button.