

Activity 5 Part 2

Ian M. McConihay

College of Science, Engineering and Technology, Grand Canyon University

CST-150: C# Programming I

Mark Smithers

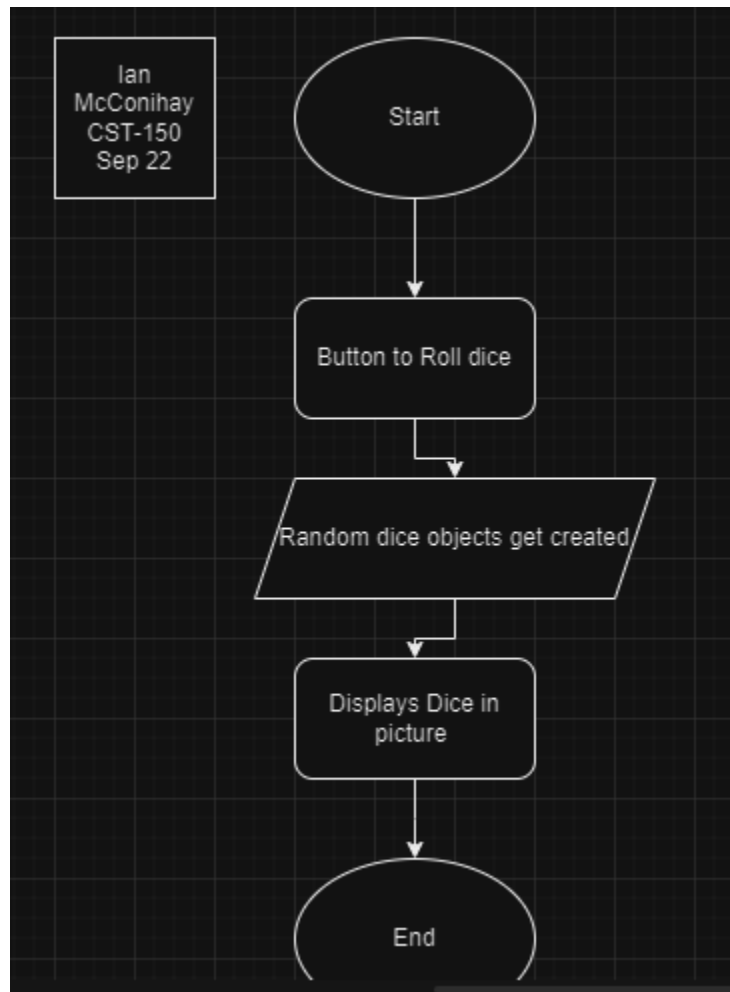
September 22, 2024

Video Link:

<https://www.loom.com/share/deef317f5a0144fbabe0d8410c7ce026?sid=c09d5bf9-4157-4a6d-8ad6-a56a04f0bed7>

Github: <https://github.com/Ian-McConihay/CST-150>

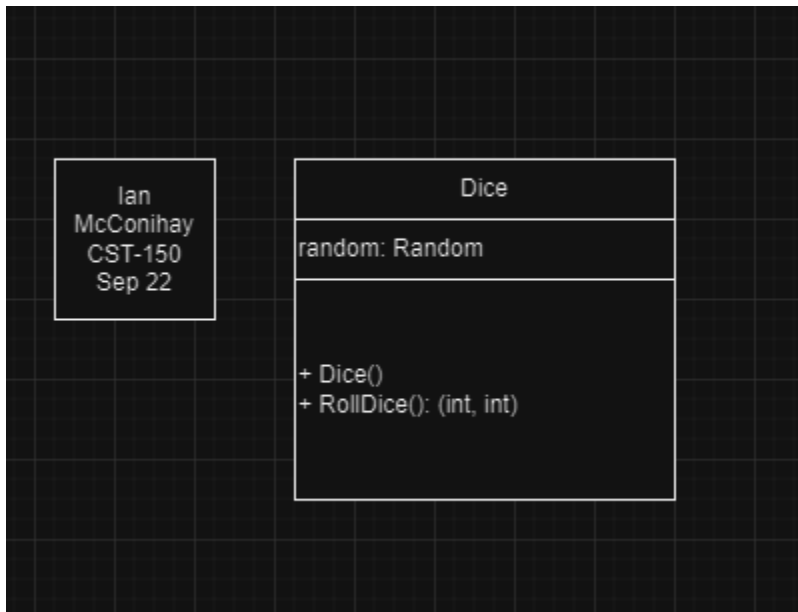
Flowchart



The flow chart for the Activity 5 Part 2 application. This application allows the user to click a button to roll a pair of dice. Once the button is clicked, the user will have two dice appear with the amount rolled.

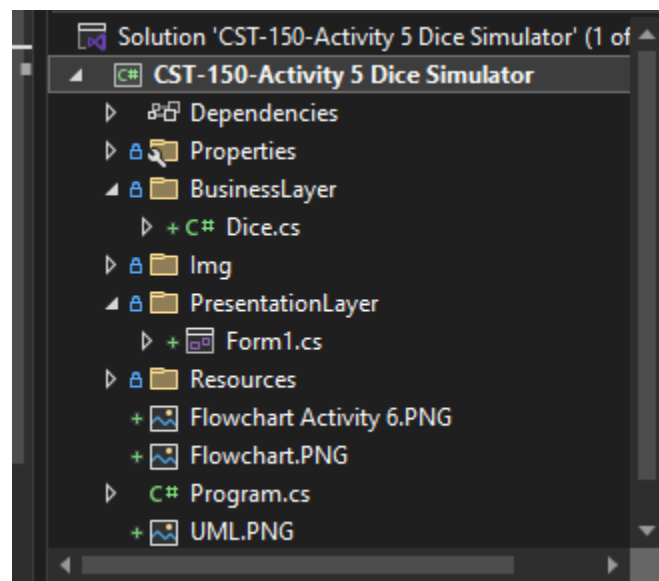
UML

Figure 1: Dice



The UML Dice object is the main object of the application. One property for setting a random var for Random. Also, a constructor and a RollDice method. This is in the businesslayer.

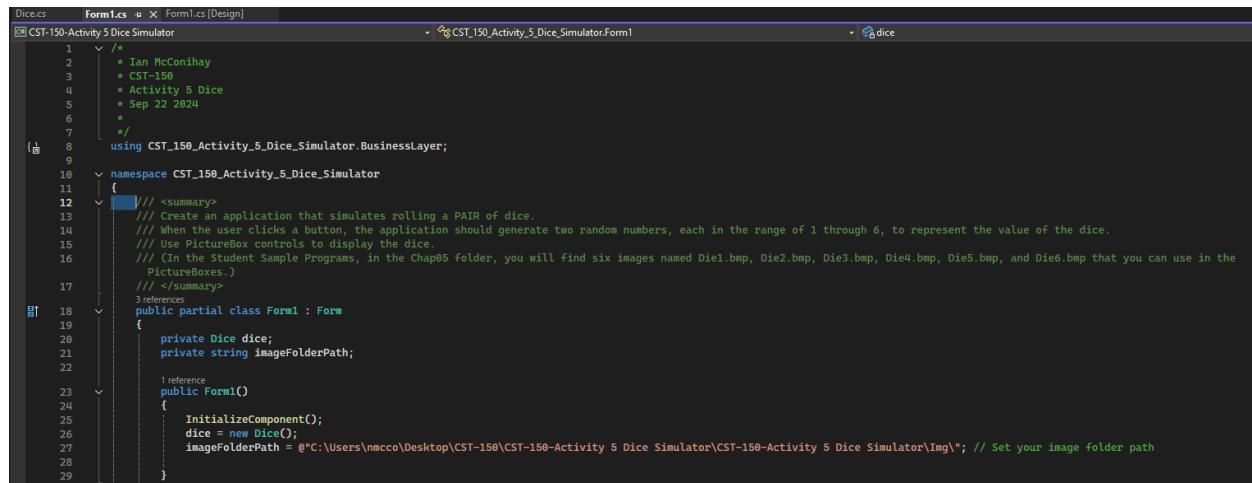
N-Layer



Here is a screenshot of the file structure for the application. N-layer was required and demonstrated in the assignment. Dice has been moved to the BusinessLayer and The PresentationLayer contains the Main form for design.

Application Screenshots

Figure 1: Code



```

1  /*
2   * Ian McConihay
3   * CST-150
4   * Activity 5 Dice
5   * Sep 22 2024
6   */
7
8  using CST_150_Activity_5_Dice_Simulator.BusinessLayer;
9
10 namespace CST_150_Activity_5_Dice_Simulator
11 {
12     /// <summary>
13     /// Create an application that simulates rolling a PAIR of dice.
14     /// When the user clicks a button, the application should generate two random numbers, each in the range of 1 through 6, to represent the value of the dice.
15     /// Use PictureBox controls to display the dice.
16     /// (In the Student Sample Programs, in the Chap85 folder, you will find six images named Die1.bmp, Die2.bmp, Die3.bmp, Die4.bmp, Die5.bmp, and Die6.bmp that you can use in the
17     /// PictureBoxes.)
18     /// </summary>
19     ///
20     References:
21     public partial class Form1 : Form
22     {
23         private Dice dice;
24         private string imageFolderPath;
25
26         1 reference
27         public Form1()
28         {
29             InitializeComponent();
30             dice = new Dice();
31             imageFolderPath = @"C:\Users\mmcco\Desktop\CST-150\CST-150-Activity 5 Dice Simulator\CST-150-Activity 5 Dice Simulator\Img\*"; // Set your image folder path
32         }
33     }
34 }

```

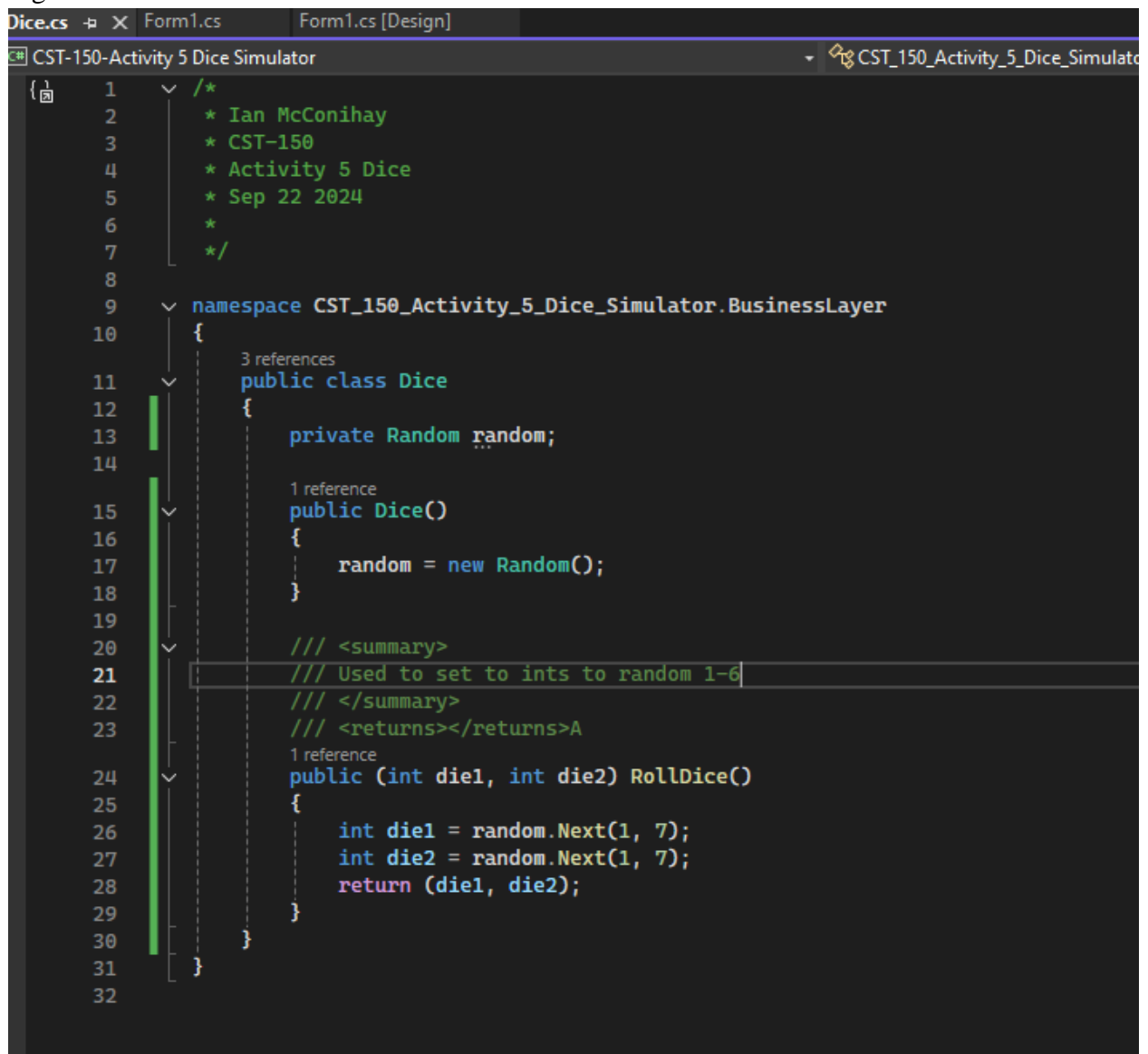
In this screenshot we can see the citation. After that we set a Dice object to be used for click event. We also have the file path used for grabbing our images.

Figure 2: Code

```
31 1 reference
32 public void btnRollDice_Click(object sender, EventArgs e)
33 {
34     // Roll the dice using the Dice Class
35     var (die1, die2) = dice.RollDice();
36
37     string die1ImagePath = $"{imageFolderPath}die{die1}.png";
38     string die2ImagePath = $"{imageFolderPath}die{die2}.png";
39
40     //Check if images are null
41     if (die1ImagePath == null || die2ImagePath == null)
42     {
43         MessageBox.Show("One of the images is not found.");
44         return;
45     }
46
47     // Update PictureBox controls with the corresponding images
48     picBoxDiceOne.Image = Image.FromFile(die1ImagePath);
49     picBoxDiceTwo.Image = Image.FromFile(die2ImagePath);
50     // Display the result
51     lblResult.Text = $"You rolled a {die1} and a {die2}!";
52     lblResult.Visible = true;
53 }
54 }
55 }
56 }
```

In this screenshot we have the btn click event. This btn uses the RollDice method to set the dice path parameter. Once that is set, we check if the images are null. If they are good, then the picture box will persist the image to the corresponding roll.

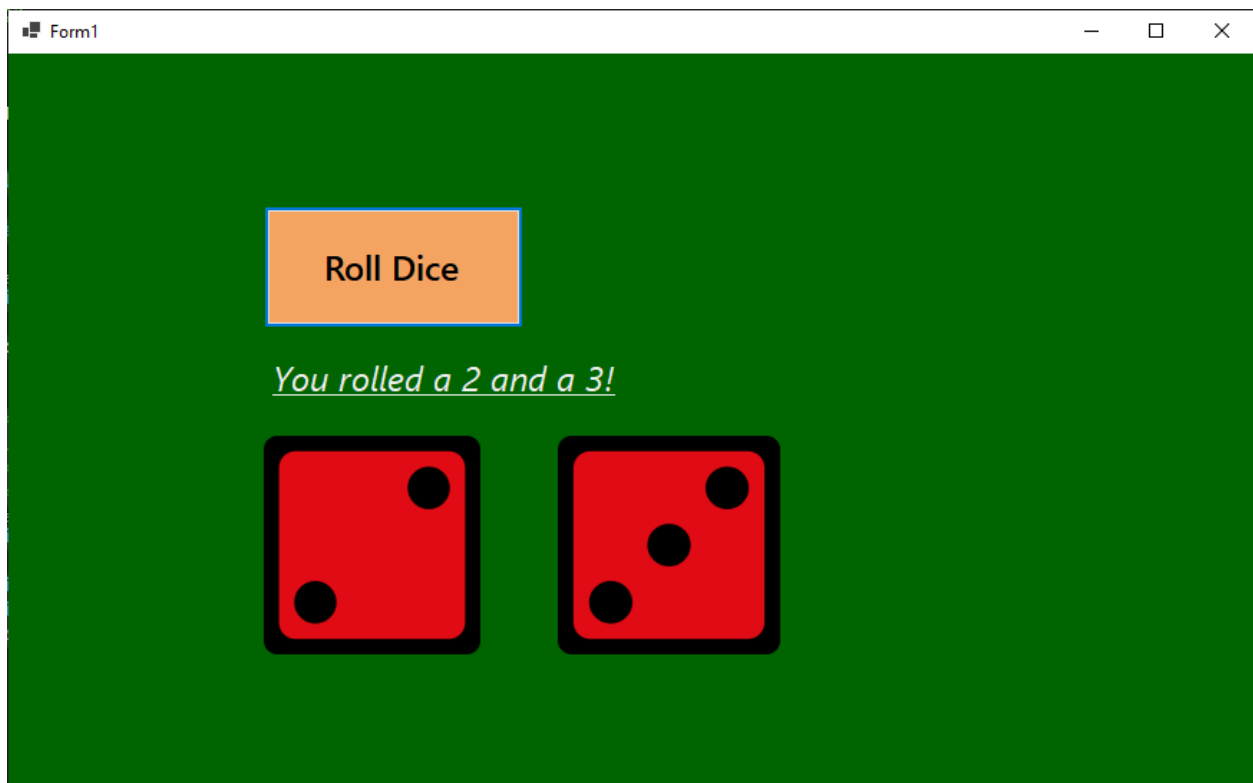
Figure 3: Dice Code



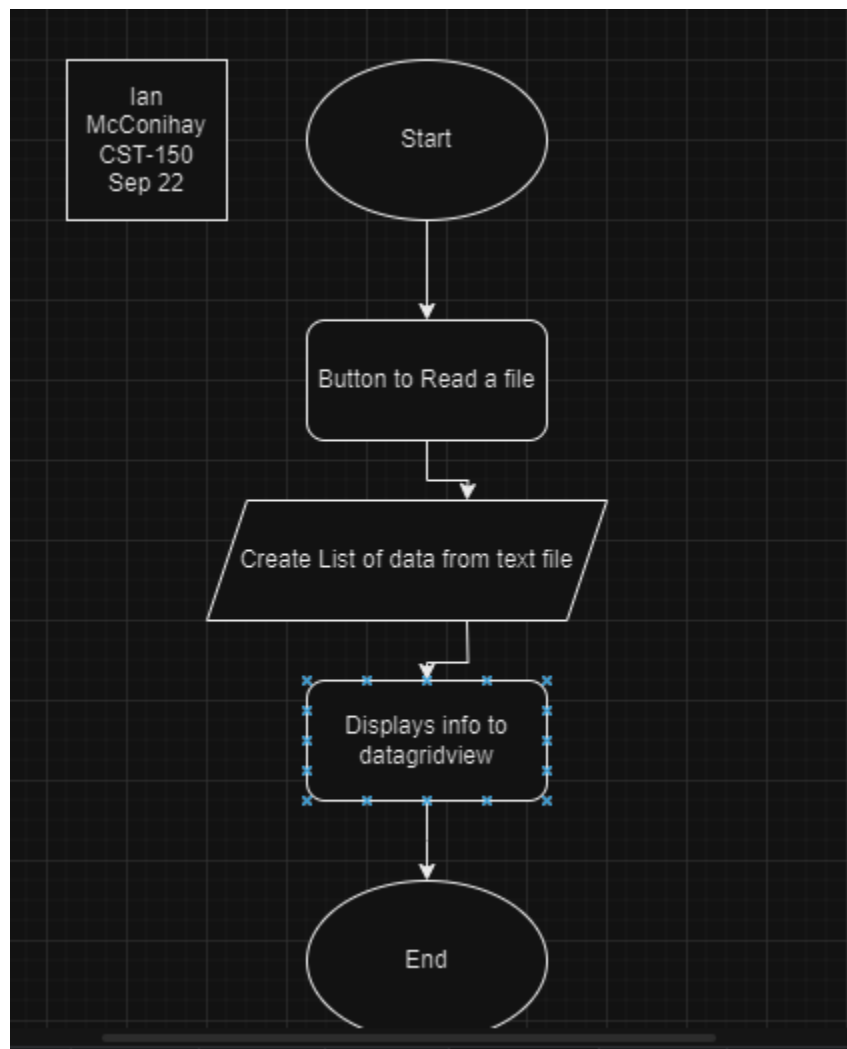
```
1  /*
2   * Ian McConihay
3   * CST-150
4   * Activity 5 Dice
5   * Sep 22 2024
6   *
7   */
8
9  namespace CST_150_Activity_5_Dice_Simulator.BusinessLayer
10 {
11     3 references
12     public class Dice
13     {
14         private Random random;
15
16         1 reference
17         public Dice()
18         {
19             random = new Random();
20         }
21
22         /// <summary>
23         /// Used to set to ints to random 1-6
24         /// </summary>
25         /// <returns></returns>A
26         1 reference
27         public (int die1, int die2) RollDice()
28         {
29             int die1 = random.Next(1, 7);
30             int die2 = random.Next(1, 7);
31             return (die1, die2);
32         }
33     }
34 }
```

Dice class has a random attribute for the single method. The method RollDice takes in two ints and assigns them a random number between 1 and 6. This is the core functionality of the dice class.

Figure 4: Application Running



The application running shows the button clicked. There is a text description displaying what the user rolled. There are also the picture boxes showing the corresponding numbers.

Part 2 of Activity 5**Flowchart**

Activity 5 part 2 required a flowchart for Activity 6. This application allows the user to read from a text file. Then it will display the text file persisted through a List into a datagridview.

What was challenging?

Getting the image paths was a challenge. I first tried to use the Properties folder using the Resource file but that wasn't working.

What did you learn?

I learned about file path manipulation.

How would you improve on the project?

I added a dice with a fire background and I would use that as a default image before clicking the roll dice button.

How can you use what you learned on the job?

File path manipulation can be used for countless tasks and could even be used with a switch statement.