Lab 6 Due next lab period – week of May 30 – June 2

Purpose

1. Create a simple X and Os game using C#
2. Create a class in C#
3. Use event handlers in C#

What to submit

1. Demo will be done in lab, in your lab section.
2. Upload your project compressed into a .zip file to the Dropbox on D2L.

Evaluation (15 marks):

1. Be prepared to demonstrate your solution at the start of next lab. If you are not ready at the start of lab or are not present, a mark of 0 will be given.
2. Your solution will be assessed based on (3 marks each):
    - Proper form layout, adhering to principles of good design
    - Correctly implementing the game logic (win/draw state), and informing user of result
    - All 9 X/O buttons work correctly
    - VB.NET code style – variables named appropriately, comments used in the code where needed, code is readable and understandable.
    - Correctly answering questions about your solution.
3. **All work must be individual.** Any plagiarism will result in a mark of 0 and possibly additional penalties.

Description

In this lab you will create a simple Tic Tac Toe (XO) game in Visual Studio, using C#. To see how the game should look (although you are welcome to improve upon this of course), see the video posted on D2L.

For the rules of the game, see: https://en.wikipedia.org/wiki/Tic-tac-toe

Steps

REVIEW THIS SECTION COMPLETELY BEFORE STARTING.

1. Preliminaries:
    a. Launch Visual Studio, and create a new C# Windows Form Application and call it XOGame. (You are welcome to use a WPF if you like)
    b. Rename Form1.cs to XOForm.cs
2. Add a new Class to your project. Name it Turn.cs
    a. Add the code in Turn.txt on D2L to this class. Overwrite any existing code.
    b. This class handles how turns are taken. To start there are 9 turns (total number of squares that can be clicked), and player X starts
    c. The Flip() method switched players by:
        i. Flipping to next player
        ii. Decrementing the number of turns available
        iii. Triggering the Flipped event

3. Set up your Form:
    a. Give your Form an appropriate name (not Form1)
    b. Add 9 buttons in a 3x3 grid. For each one set the Text property to the empty string
    c. Name them as follows:

_00Button	_01Button	_02Button
_10Button	_11Button	_12Button
_20Button	_21Button	_22Button

    d. Set the font size to Microsoft Sans Serif 28pt
    e. Make each button square (75x75 works)
4. Add a label to display the current turn:
    a. Set its Dock property to Bottom
    b. Set its Autosize property to False
5. Edit the XOForm code
    a. Add a private class variable of type Turn, called _turn
    b. Add the following private class method:

```
private void OnFlipped(object sender, EventArgs e)
 {
   _turnLabel.Text = string.Format("{0} Go!", _turn.WhoseTurn);
 }
```

    c. In the Form Load event do the following:
        i. Create an instance of a Turn object and assign it to the turn variable
        ii. Attach the OnFlipped method to our Turn object's Flipped event using the following code

```
1. _turn.Flipped += OnFlipped;
```

        iii. Call the OnFlipped method by using the following code:

```
1. OnFlipped(null, EventArgs.Empty);
```

6. Add a private class variable called grid, of type 3x3 char array
7. Write the Button handlers:
    a. You can start by writing the code for just the _00Button:
        i. Set the button's text the current player's char (X or O)
        ii. Set grid[0,0] to the current player's char
        iii. Flip the turn
        iv. Disable the button by setting its Enabled property to False
    b. Once you have one button working, write equivalent methods for the other button events
    c. Optional (Bonus mark) Reduce all the Button_Click methods to one method, and have all the Button controls call this method when their Click event is triggered.
8. Add the logic to determine when the game is won or drawn. Use a MessageBox to display if the game is won or is a draw.
9. Feel free to make the game awesomer.