Anna Phan

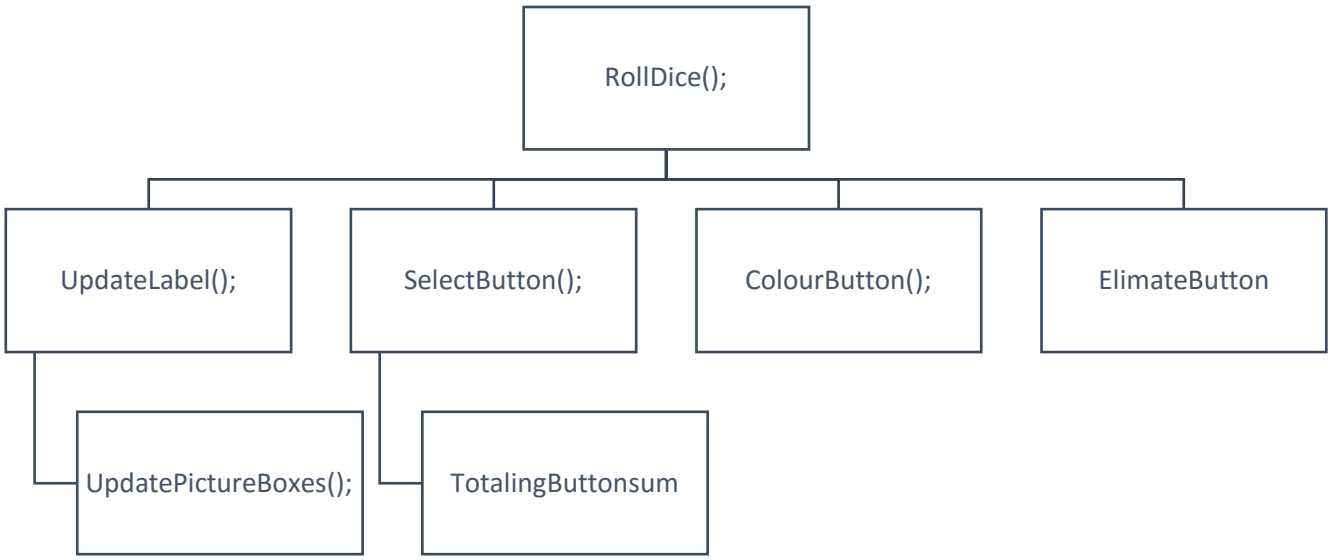
100303872

INFO 1112 – S11

Final Project

Ali Madooei

Structure Chart



Implementation

I started out with referencing with the previous assignment because the dice game before was similar. I have brought over the first three methods and used them in the project. I divided the project by having method for the dices which included the generating the random numbers, adding the sum into the label and updating the picture images to match with the dice. Next, I added out a message box that displayed the button that I have chosen. That was removed afterwards, when the button clicked was displayed correctly. I used the button's number in an array to store which button was pressed. Lastly, I check if the button that I picked could change colours by using the numbers in the stored array. I made that when I was able to get the buttons to dark green I could be able to disable them.

Below I have listed my methods:

RollDice();	I started out with a RolDice(); method to simply give random numbers for the two dices. This method and the two below it was taken from the previous assignment because the idea of rolling dices was the same.
UpdateLabel();	This method was to update the labels with the sum of the dices. I added this into the RollDice(); method because I wanted to call on it once in the main method by using the RollDice button.

Anna Phan

100303872

INFO 1112 – S11

Final Project

Ali Madooei

UpdatepictureBoxes()	This method was to update the pictures. This was added into the RollDice(); method because I can call on it once in the main method.
SelectButton();	This method was to determine what button was pressed and which one were not. Here I made an event handler of which buttons were pressed. Then I converted it as a variable to I can send it though the Totaling ButtonSum(); method. I have also made it into a text and shown it in the lbButtonSelected label to debug if the current button pressed was correct.
TotalingButtonsum();	This method was to total up the button 's sum. What I have done was had all the bottoms pressed into an array. I created a switch in case, if the user had changed his/her mind with the button chosen. The numbers that are in the array is put into a for loop that loops and counted the sum. I have displayed the total in the lbUsersum for debugging purposes.
ColourChangeButton();	This method was a toggle to change back and forth the button selected by the user. This was shown in the class's demo. But this is a switch to see if the button was pressed or not. If the button was clicked, the button changes the bool to true which make the button to light green. Else, it will go back to it regular colour of light grey.
ElimateButton()	This was to eliminate any button that the user had chosen. This also disable the button so that the user is unable to click the buttons again.

In the main method, I had an if-else statement to check if the user's sum matches with the die's sum. Because the die's and user's sum was zero, it rolled the dices right away. I have added the ElminateButton(); method that when the sum matches it will disable the button that was located in the array. If the sums did not match, it went in the else statement which revert everything back to system systems.

Lastly, the New Game button cleared everything and the game can be played once again.

References: C# tutorial: Detect which button is clicked

Thanks for a great semester! ☺