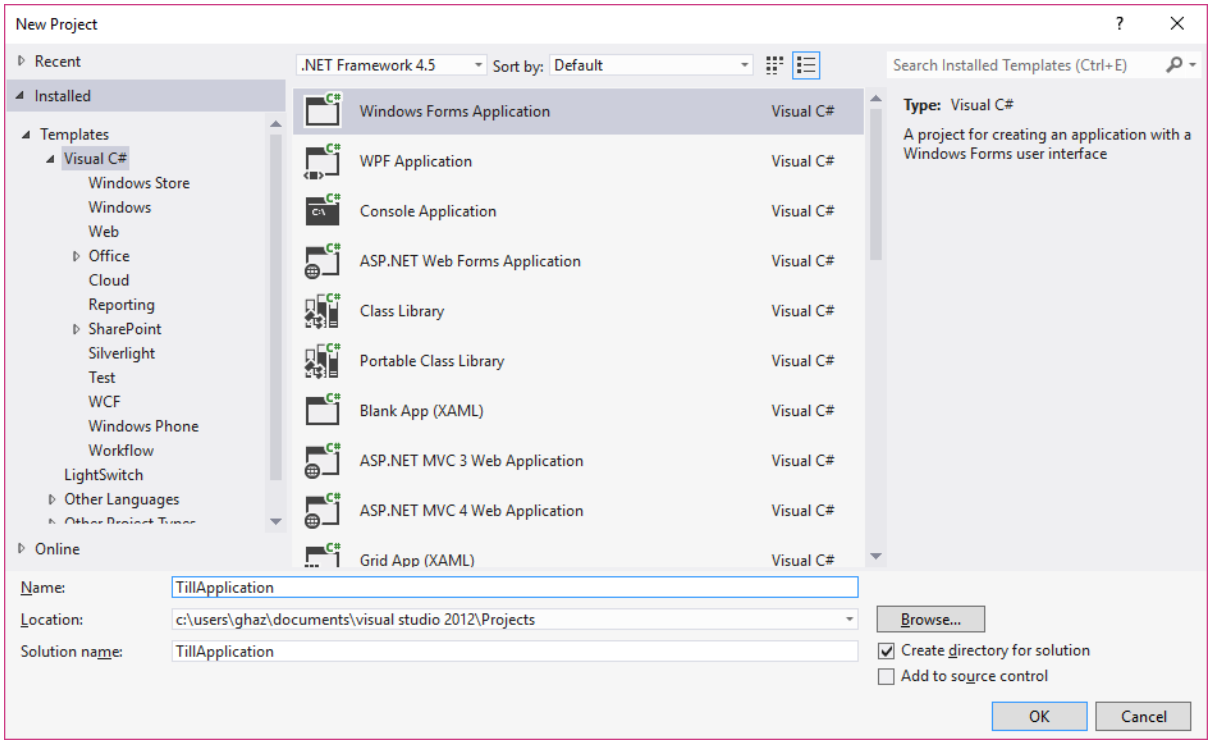
Creating a till system in C#

We will be using C# to code a till system with different food types and prices. We will create this using a Windows Form Application in Visual Studio.

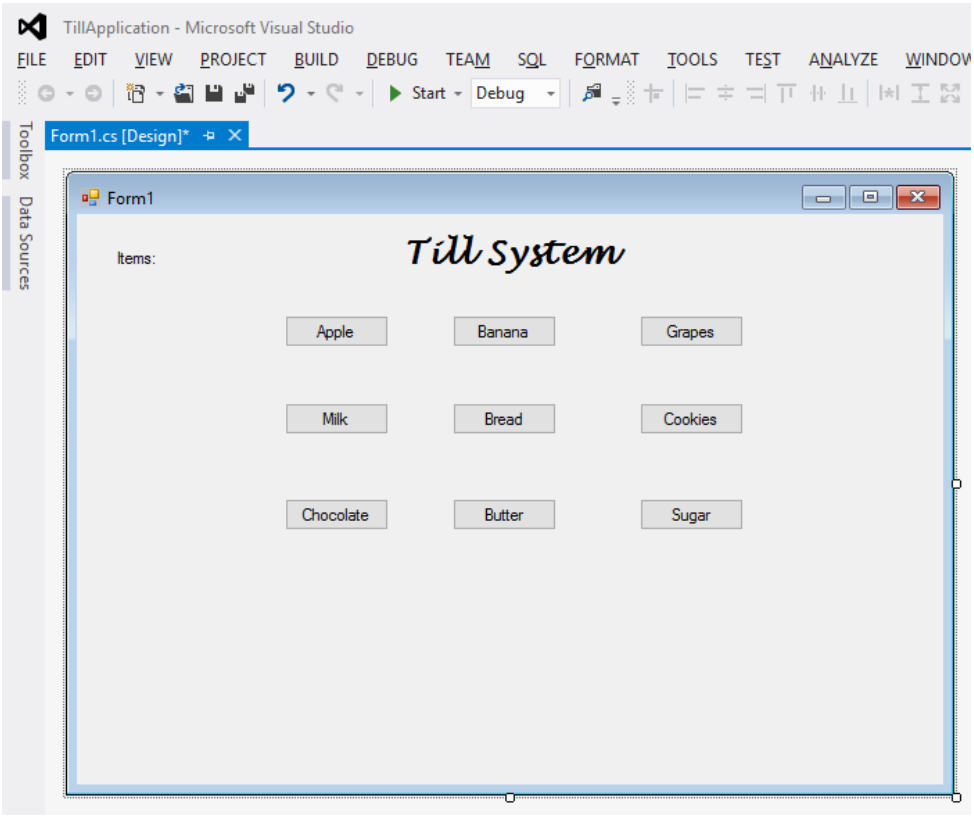
1. Create a new Windows Form project with a suitable name



2. Insert a label and 9 buttons to the form. Follow the details below:

Control	Name	Text
Label	lblItems	Items:
Button1	btnApple	Apple
Button2	btnBanana	Banana
Button3	btnGrapes	Grapes
Button4	btnMilk	Milk
Button5	btnBread	Bread
Button6	btnCookies	Cookies
Button7	btnChocolate	Chocolate
Button8	btnButter	Butter
Button9	btnSugar	Sugar

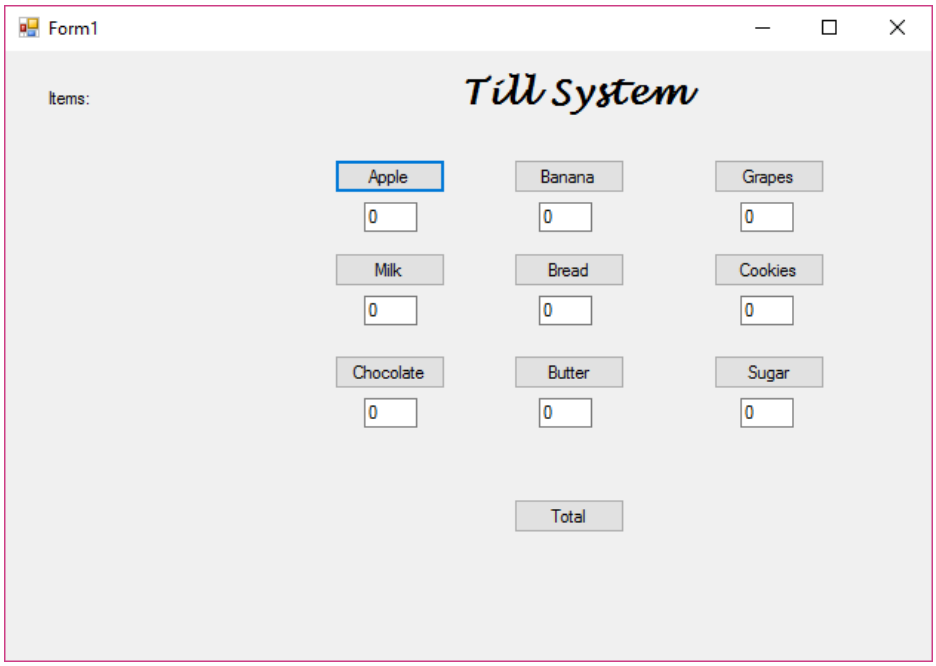
3. You can also include an additional label for the heading. Adjust the font as required:



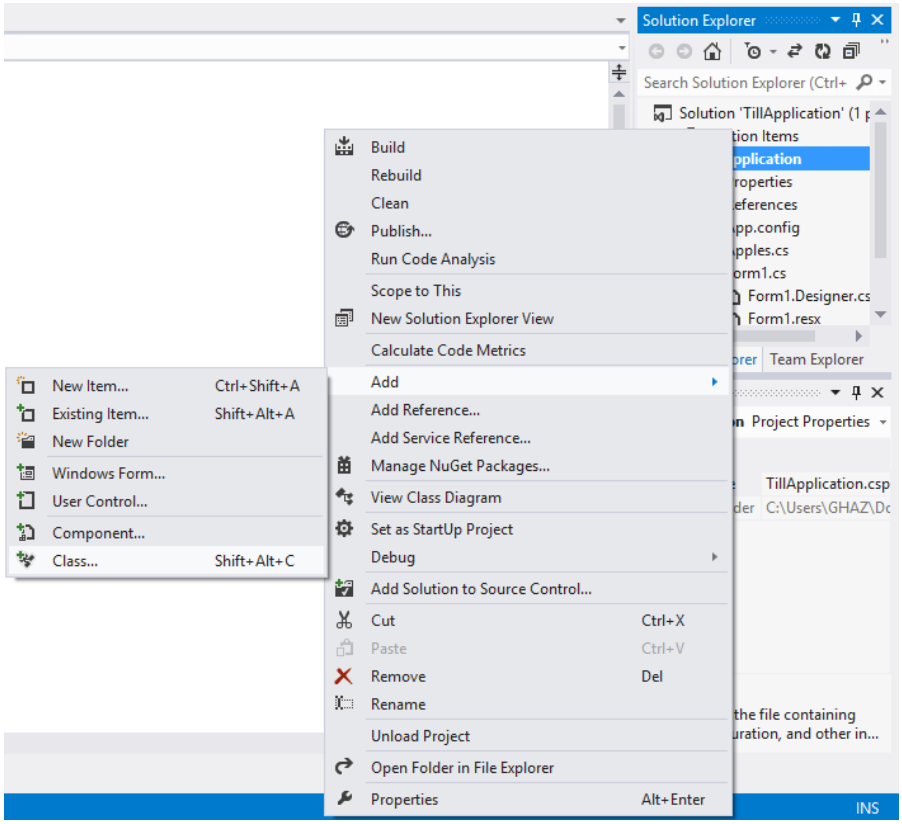
4. We now need to insert textboxes for quantities of each item. Insert a **TextBox** beneath each button with the details below:

Control	Name	Text
TextBox1	tbxApple	0
TextBox2	tbxBanana	0
TextBox3	tbxGrapes	0
TextBox4	tbxMilk	0
TextBox5	tbxBread	0
TextBox6	tbxCookies	0
TextBox7	tbxChocolate	0
TextBox8	tbxButter	0
TextBox9	tbxSugar	0

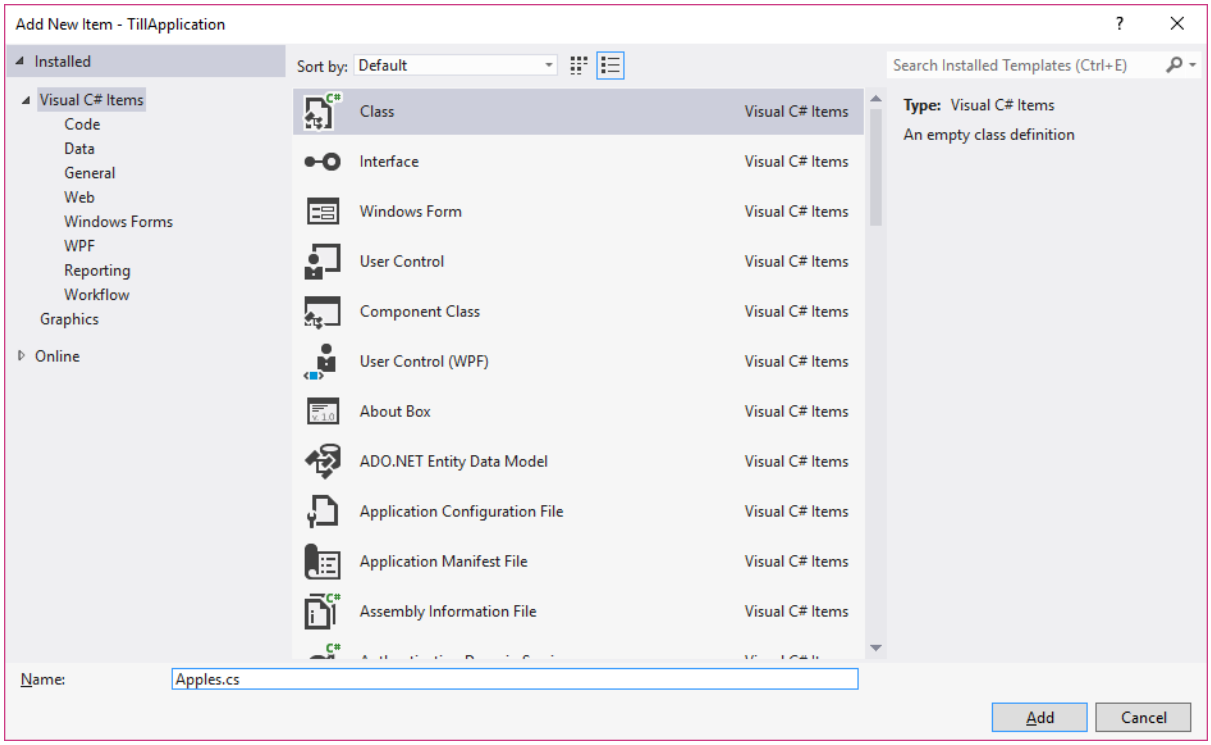
5. Once you have created a class for each till object, add a total button to the form. Call this **btnTotal**. Add a label to the form and call this **lblTotal** with no text.



6. With the basic setup complete we will now code the first **class** to create an **Apple** object. To do this right click on your project in the **Solution Explorer** and select **Add** -> **Class…**



7. Make sure a **Visual C# Class** is selected and give it a name of **Apples.cs,** then click **Add**.



8. We will code the Apples class to be used to create an Apple object. We will have two key parts of the constructor, one for the price and one for the name. First create two global variables for each item:

public	float	price
public	String	name

9. Now we will create the constructor. This means that when we create an Apples object we will need to provide details about the price and name of the object.

```
public Apples(float price, String name)
{
    this.price = price;
    this.name = name;
}
```

10. We will also create a method that will work out the total price of the apples after they have been bought. This will take the quantity from the textbox on the form and multiply this by the price. The price will need to be cast to an integer type to do this.

```
public int Total(int quantity)
{
    return (int)price * quantity;
}
```

11. Now that the class is ready we need a way of creating a new Apples object from the form. Double click on the Apple button to bring up the coding block for the click action. We want a few things to happen here:
    a. Check that the value entered in the quantity text box is a number
    b. Check that the value entered in the quantity text box is not null or zero
    c. Only create an apple object of a quantity exists
    d. Work out the total cost of the apples
    e. Work out the total cost of the entire shopping bill
    f. Display this information in a label

12. We will start by checking if the quantity entered in the textbox is a number:

```
private void btnApple_Click(object sender, EventArgs e)
{
    //check if the text in the quantity box is numeric
    int number;
    bool appleIsNumeric = int.TryParse(tbxApple.Text, out number);
```

13. Then we will check the areas above with an if statement:

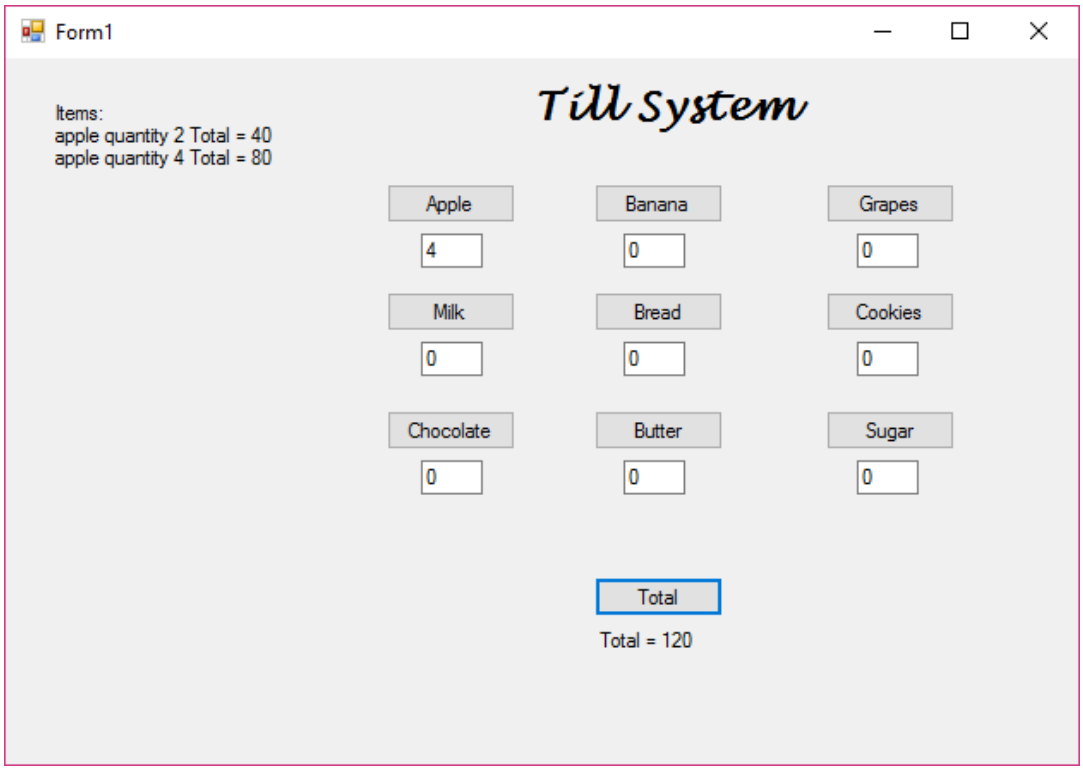
```
//if the text in the apple quantity box is null or zero
if (tbxApple.Text == "" || tbxApple.Text == "0")
{
    //show a message
    MessageBox.Show("Cannot be empty or null");
}

//if the text in the apple quantity box is not null or zero
else
{
    //check if the text is a number using the boolean value of appleIsNumeric
    if (appleIsNumeric == true)
    {
        //if the quantity is a number create an apple object
        numOfApples = int.Parse(tbxApple.Text);
        Apples apple = new Apples(20, "apple");

        //display the details of the object in the Items label
        lblItems.Text += "\n" + apple.name + " quantity " + numOfApples + " Total = " + apple.Total(numOfApples);

        //update the total value with the new total + old total
        total += apple.Total(numOfApples);
    }
```

14. If you now run the application you can see that entering a quantity and clicking on the apple button will add the apple item to the list on the left. If you click the total button it will update the entire total of the shopping.



15. Continue creating classes for the other objects required for the till.