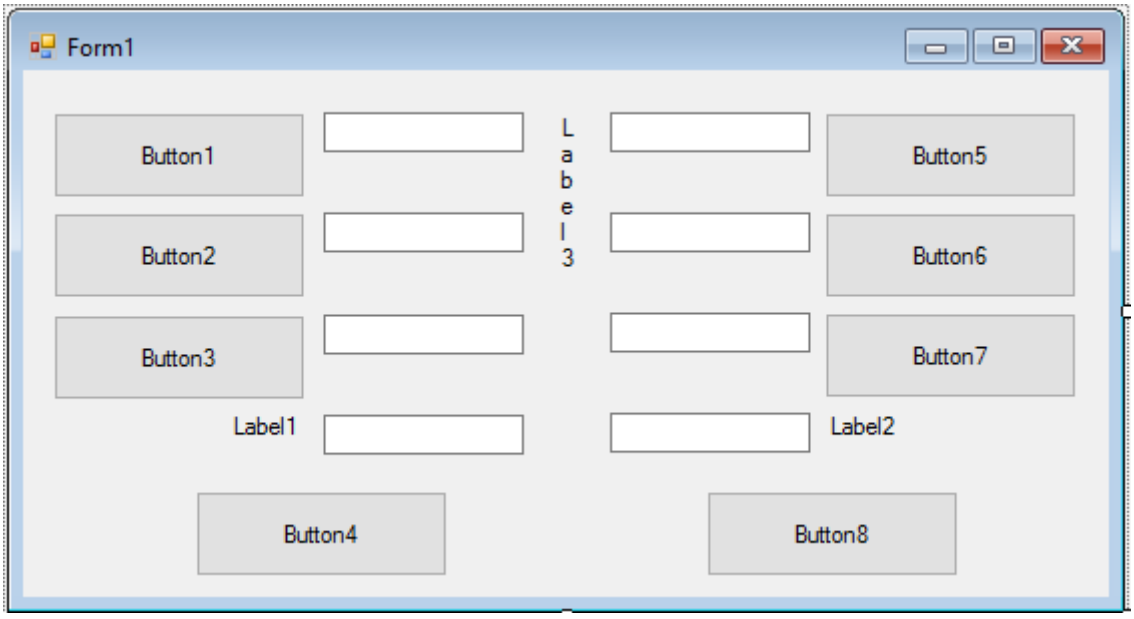
Project 10 - Checkbook Balancer

Project Design

This project will help you do that dreaded monthly task of balancing your checkbook. By entering requested information, you can find out just how much money you really have in your account. The project you are about to build is saved as **Checkbook** in the project folder (**\BeginVCS\BVCS Projects**).

Place Controls on Form

Start a new project in Visual C#. Place three label controls (on one, set **AutoSize** to **False** and make it tall and skinny to use as a dividing line), eight text boxes, and eight buttons on the form. When done, your form should look something like this:



Set Control Properties

Set the control properties using the properties window:

Form1 Form:

Property Name	Property Value
Text	Checkbook Balancer
FormBorderStyle	FixedSingle
StartPosition	CenterScreen

label1 Label:

Property Name	Property Value
AutoSize	False
Text	Adjusted Statement Balance
TextAlign	TopRight

label2 Label:

Property Name	Property Value
AutoSize	False
Text	Adjusted Checkbook Balance

label3 Label:

Property Name	Property Value
AutoSize	False
BackColor	Black
Text	[Blank it out]

textBox1 Text Box:

Property Name	Property Value
Name	txtStmtBalance
Text	0
TextAlign	Right

textBox2 Text Box:

Property Name	Property Value
Name	txtStmtDeposit
Text	0
TextAlign	Right
BackColor	White
ReadOnly	True
TabStop	False

textBox3 Text Box:

Property Name	Property Value
Name	txtStmtCheck
Text	0
TextAlign	Right
BackColor	White
ReadOnly	True
TabStop	False

textBox4 Text Box:

Property Name	Property Value
Name	txtAdjStmtBalance
Text	0
TextAlign	Right
BackColor	White
ReadOnly	True
TabStop	False

textBox5 Text Box:

Property Name	Property Value
Name	txtChkBalance
Text	0
TextAlign	Right

textBox6 Text Box:

Property Name	Property Value
Name	txtChkDeposit
Text	0
TextAlign	Right
BackColor	White
ReadOnly	True
TabStop	False

textBox7 Text Box:

Property Name	Property Value
Name	txtChkCharge
Text	0
TextAlign	Right
BackColor	White
ReadOnly	True
TabStop	False

textBox8 Text Box:

Property Name	Property Value
Name	txtAdjChkBalance
Text	0
TextAlign	Right
BackColor	White
ReadOnly	True
TabStop	False

button1 Button:

Property Name	Property Value
Name	btnStmtBalance
Text	Enter Statement Balance

button2 Button:

Property Name	Property Value
Name	btnStmtDeposit
Text	Add Uncredited Deposit
Enabled	False

button3 Button:

Property Name	Property Value
Name	btnStmtCheck
Text	Subtract Outstanding Check
Enabled	False

button4 Button:

Property Name	Property Value
Name	btnStmtReset
Text	Reset Statement Values

button5 Button:

Property Name	Property Value
Name	btnChkBalance
Text	Enter Checkbook Balance

button6 Button:

Property Name	Property Value
Name	btnChkDeposit
Text	Add Unrecorded Deposit
Enabled	False

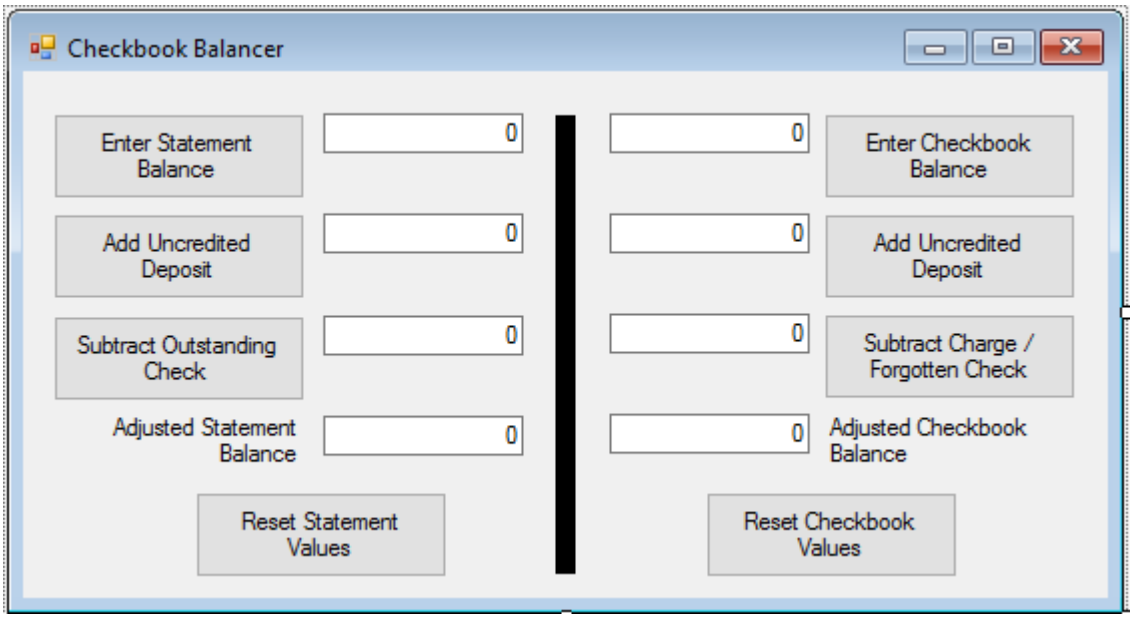
button7 Button:

Property Name	Property Value
Name	btnChkCharge
Text	Subtract Service Charge
Enabled	False

button8 Button:

Property Name	Property Value
Name	btnChkReset
Text	Reset Checkbook Values

When done setting properties, my form looks like:



Write Event Methods

Each of the eight command buttons requires a **Click** event. With each click, appropriate adjustments are made to the corresponding account balance.

Add this code to the **general declarations** area:

```
double adjStmtBalance; //  adjusted statement balance
double adjChkBalance; // adjusted checkbook balance
```

The **btnStmtBalance_Click** event method:

```
private void btnStmtBalance_Click(object sender, EventArgs e)
{
    // Read entered statement balance
    adjStmtBalance = Convert.ToDouble(txtStmtBalance.Text);
    // Disable balance, enable deposit and check
    btnStmtBalance.Enabled = false;
    btnStmtDeposit.Enabled = true;
    btnStmtCheck.Enabled = true;
    txtStmtBalance.ReadOnly = true;
    txtStmtDeposit.ReadOnly = false;
    txtStmtCheck.ReadOnly = false;
    btnStmtDeposit.Focus();
}
```

The **btnStmtDeposit_Click** event method:

```
private void btnStmtDeposit_Click(object sender, EventArgs e)
{
    // Account for uncredited deposit
    adjStmtBalance = adjStmtBalance +
Convert.ToDouble(txtStmtDeposit.Text);
    txtAdjStmtBalance.Text = "$" +
Convert.ToString(String.Format("{0:f2}", adjStmtBalance));
}
```

The **btnStmtCheck_Click** event method:

```
private void btnStmtCheck_Click(object sender, EventArgs e)
{
    // Account for outstanding check
    adjStmtBalance = adjStmtBalance -
Convert.ToDouble(txtStmtCheck.Text);
    txtAdjStmtBalance.Text = "$" +
Convert.ToString(String.Format("{0:f2}", adjStmtBalance));
}
```

The **btnStmtReset_Click** event method:

```
private void btnStmtReset_Click(object sender, EventArgs e)
{
    // Reset statement values to defaults
    adjStmtBalance = 0;
    txtStmtBalance.Text = "0";
    txtStmtDeposit.Text = "0";
    txtStmtCheck.Text = "0";
    txtAdjStmtBalance.Text = "0";
    btnStmtBalance.Enabled = true;
    btnStmtDeposit.Enabled = false;
    btnStmtCheck.Enabled = false;
    txtStmtBalance.ReadOnly = false;
    txtStmtDeposit.ReadOnly = true;
    txtStmtCheck.ReadOnly = true;
    btnStmtBalance.Focus();
}
```

The **btnChkBalance_Click** event method:

```
private void btnChkBalance_Click(object sender, EventArgs e)
{
    // Read entered checkbook balance
    adjChkBalance = Convert.ToDouble(txtChkBalance.Text);
    // Disable balance, enabled deposit and charge
    btnChkBalance.Enabled = false;
    btnChkDeposit.Enabled = true;
    btnChkCharge.Enabled = true;
    txtChkBalance.ReadOnly = true;
    txtChkDeposit.ReadOnly = false;
    txtChkCharge.ReadOnly = false;
    btnChkDeposit.Focus();
}
```

The **btnChkDeposit_Click** event method:

```
private void btnChkDeposit_Click(object sender, EventArgs e)
{
    // Account for unrecorded deposit
    adjChkBalance = adjChkBalance +
Convert.ToDouble(txtChkDeposit.Text);
    txtAdjChkBalance.Text = "$" +
Convert.ToString(String.Format("{0:f2}", adjChkBalance));
}
```

The **btnChkCharge_Click** event method:

```
private void btnChkCharge_Click(object sender, EventArgs e)
{
    // Account for service charge
    adjChkBalance = adjChkBalance -
Convert.ToDouble(txtChkCharge.Text);
    txtAdjChkBalance.Text = "$" +
Convert.ToString(String.Format("{0:f2}", adjChkBalance));
}
```

The **btnChkReset_Click** event method:

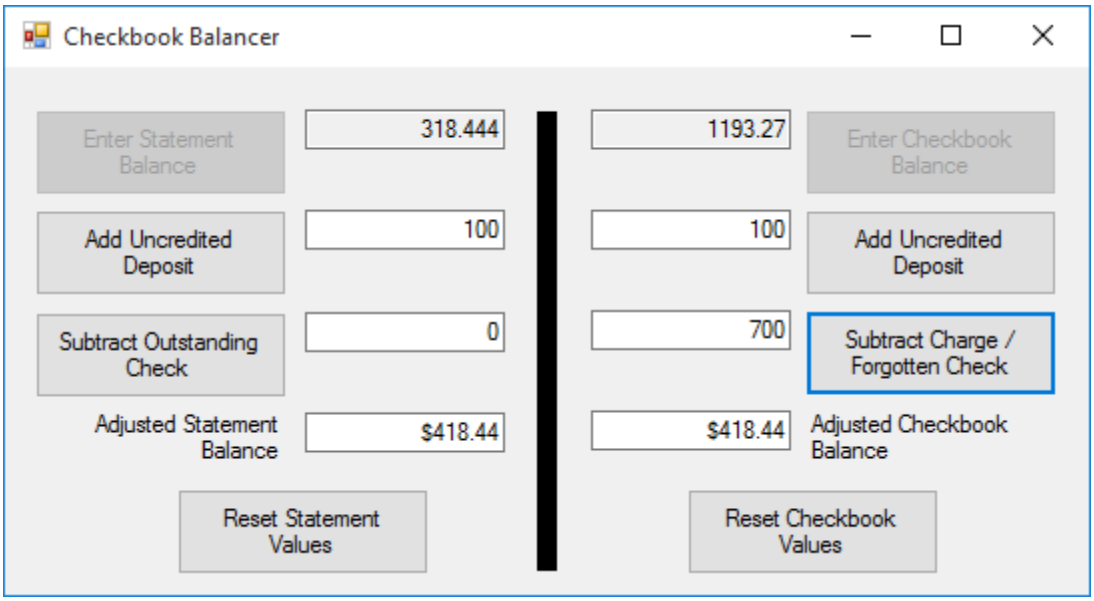
```
private void btnChkReset_Click(object sender, EventArgs e)
{
    // Reset all checkbook values to defaults
    adjChkBalance = 0;
    txtChkBalance.Text = "0";
    txtChkDeposit.Text = "0";
    txtChkCharge.Text = "0";
    txtAdjChkBalance.Text = "0";
    btnChkBalance.Enabled = true;
    btnChkDeposit.Enabled = false;
    btnChkCharge.Enabled = false;
    txtChkBalance.ReadOnly = false;
    txtChkDeposit.ReadOnly = true;
    txtChkCharge.ReadOnly = true;
    btnChkBalance.Focus();
}
```

Run the Project

Save your work. Run the project. Try balancing your latest bank statement with your checkbook - here's the procedure. Start on the left side of the form. Fill in your statement balance and click **Enter Statement Balance**. Next, enter each deposit you have made that is not recorded on the bank statement. After each entry, click **Add Uncredited Deposit**. Next, enter each check you have written that is not listed on the statement. After each check, click **Subtract Outstanding Check**. When done, your **Adjusted Statement Balance** is shown. Clicking **Reset Statement Values** will set everything on the left side back to default values.

Now to the right side of the form. Fill in your checkbook balance and click **Enter Checkbook Balance**. Next, enter each deposit shown on your bank statement that you forgot to enter in your checkbook. After each entry, click **Add Unrecorded Deposit**. Next, enter any service charge the bank may have charged or any check you that you haven't recorded in your checkbook. After each charge, click **Subtract Charge / Forgotten Check**. When done, your **Adjusted Checkbook Balance** is shown. Clicking **Reset Checkbook Values** will set everything on the left side back to default values.

At this point, the adjusted balances at the bottom of the form should be the same. If not, you need to dig deeper into your checks, deposits, and service charges to see what's missing or perhaps accounted for more than once. Here's a run on my account – what do you know? It balanced!!



Other Things to Try

An interesting and useful modification to this project requires learning about a new control - the **Combo Box**. In this control, you can build up a list of entered information and edit it as you see fit. It would be useful to use combo boxes to store up the uncredited and unrecorded deposits, the outstanding checks, and any service charges. With complete lists, you could edit them as you see fit. This would make the checkbook balancing act an easier task. In time, you can even learn to add printing options to the project.