

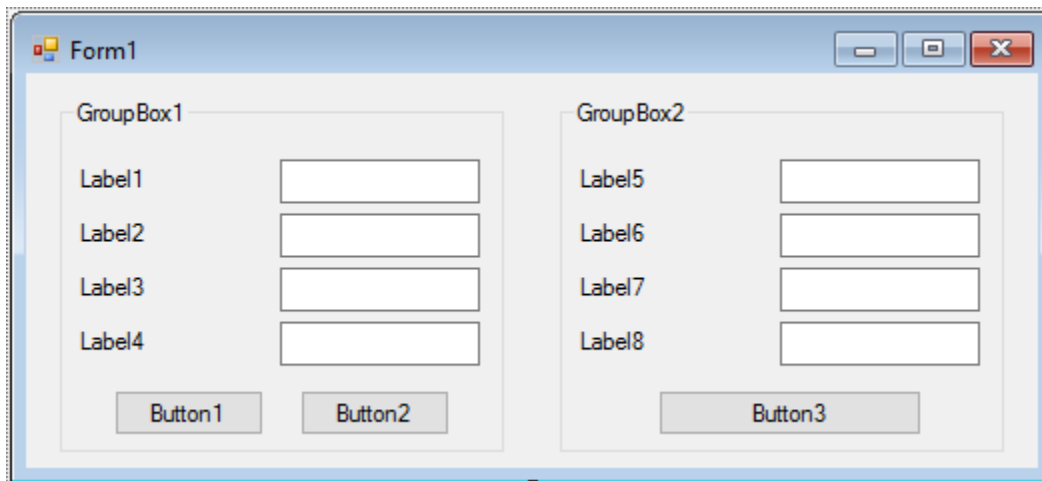
Project 11 – Portfolio Manager

Project Design

In this project, we will build a tool that lets you determine the current value of your stock holdings. You store when you bought a particular stock, how many shares you bought and how much you paid. Then, whenever you want, you enter current values to determine your gain (or possible losses). The project you are about to build is saved as **Portfolio** in the project folder (\BeginVCS\BVCS Projects).

Place Controls on Form

Start a new project in Visual C#. Place two group box controls on the form. In the first group box, place four labels, four text boxes and two buttons. In the second group box, place four labels, four text box controls and a button. When done, your form should look something like this:



The screenshot shows a Windows Form titled "Form1" with a standard Windows XP-style title bar (minimize, maximize, close buttons). The form contains two group boxes, "GroupBox1" and "GroupBox2", arranged side-by-side. GroupBox1 contains four labels (Label1, Label2, Label3, Label4) stacked vertically, each followed by a text box. Below these are two buttons, "Button1" and "Button2". GroupBox2 contains four labels (Label5, Label6, Label7, Label8) stacked vertically, each followed by a text box. Below these is a single button, "Button3".

Set Control Properties

Set the control properties using the properties window:

Form1 Form:

Property Name	Property Value
Text	Portfolio Manager
FormBorderStyle	FixedSingle
StartPosition	CenterScreen

groupBox1 Group Box:

Property Name	Property Value
Name	grpStock
Text	This Stock
Font Size	12
Font Style	Bold

label1 Label:

Property Name	Property Value
Text	Date Purchased
Font Size	8
Font Style	Regular

label2 Label:

Property Name	Property Value
Text	Price/Share
Font Size	8
Font Style	Regular

label3 Label:

Property Name	Property Value
Text	Number of Shares
Font Size	8
Font Style	Regular

label4 Label:

Property Name	Property Value
Text	Price Paid
Font Size	8
Font Style	Regular

textBox1 Text Box:

Property Name	Property Value
Name	txtDate
TextAlign	Right
BackColor	White
Font Size	10
Font Style	Regular
ReadOnly	True
TabStop	False

textBox2 Text Box:

Property Name	Property Value
Name	txtPrice
TextAlign	Right
BackColor	White
Font Size	10
Font Style	Regular
ReadOnly	True
TabStop	False

textBox3 Text Box:

Property Name	Property Value
Name	txtShares
TextAlign	Right
BackColor	White
Font Size	10
Font Style	Regular
ReadOnly	True
TabStop	False

textBox4 Text Box:

Property Name	Property Value
Name	txtPaid
TextAlign	Right
BackColor	White
Font Size	10
Font Style	Regular
ReadOnly	True
TabStop	False

button1 Button:

Property Name	Property Value
Name	btnPrevious
Text	Previous
Font Size	8
Font Style	Regular

button2 Button:

Property Name	Property Value
Name	btnNext
Text	Next
Font Size	8
Font Style	Regular

groupBox2 Group Box:

Property Name	Property Value
Name	grpValue
Text	Current Value
Font Size	12
FontBold	True

label5 Label:

Property Name	Property Value
Text	Today's Date
Font Size	8
Font Style	Regular

label6 Label:

Property Name	Property Value
Text	Today's Price
Font Size	8
Font Style	Regular

label7 Label:

Property Name	Property Value
Text	Yearly Return
Font Size	8
Font Style	Regular

label8 Label:

Property Name	Property Value
Text	Today's Value
Font Size	8
Font Style	Regular

textBox1 Text Box:

Property Name	Property Value
Name	txtTodayDate
TextAlign	Right
BackColor	White
Font Size	10
Font Style	Regular
ReadOnly	True
TabStop	False

textBox2 Text Box:

Property Name	Property Value
Name	txtToday
TextAlign	Right
Font Size	10
Font Style	Regular

textBox3 Text Box:

Property Name	Property Value
Name	txtReturn
TextAlign	Right
BackColor	White
Font Size	10
Font Style	Regular
ReadOnly	True
TabStop	False

textBox4 Text Box:

Property Name	Property Value
Name	txtValue
TextAlign	Right
BackColor	White
Font Size	10
Font Style	Regular
ReadOnly	True
TabStop	False

button3 Button:

Property Name	Property Value
Name	btnReturn
Text	Compute Return

When done setting properties, my form looks like this:

The screenshot shows a Windows application window titled "Portfolio". The window has a standard Windows XP-style title bar with minimize, maximize, and close buttons. The main content area is divided into two panels. The left panel, titled "This Stock", contains four text input fields labeled "Date Purchased", "Price/Share", "Number of Shares", and "Price Paid". Below these fields are two buttons: "Previous" and "Next". The right panel, titled "Current Value", contains four text input fields labeled "Today's Date", "Today's Price", "Yearly Return", and "Today's Value". Below these fields is a single button labeled "Compute Return".

Write Event Methods

In this program, you need to store information about your stocks (date purchased, purchase price and shares owned) in data arrays (the form **Load** method). Then, you use the **Previous** and **Next** buttons to view each stock. For the displayed stock, if you type in the current price (**Today's Price**) and click **Compute Return**, you will be shown the current value and yearly return for that stock.

In this project, we introduce an idea that's used all the time in computer programming. Whenever, there is a certain segment of code that needs to be repeated and used in various parts of a project, we put the corresponding code in something called a **general method**. This saves us from having to repeat code in different locations – a maintenance headache. A general method is identical in use to an event method, with the only difference being it is not invoked by some control event. We control invocation of a general method by **calling** it. In this project, we will use a general method to display the stock information after pressing the **Previous** or **Next** button. Look for the code (method is named **ShowStock**) and see how easy it is to use.

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

```
int numberStocks;  
int currentStock;  
DateTime[] stockDate = new DateTime[25];  
string[] stockName = new string[25];  
double[] stockPrice = new double[25];  
int[] stockShares = new int[25];
```


The **Form1_Load** event method:

```
private void Form1_Load(object sender, EventArgs e)
{
    // Load stock Information
    numberStocks = 6;
    stockDate[0] = new DateTime(2002, 2, 10); stockName[0] =
    "Only Go Up";
    stockPrice[0] = 10; stockShares[0] = 50;
    stockDate[1] = new DateTime(2001, 2, 1); stockName[1] =
    "Big Deal";
    stockPrice[1] = 10; stockShares[1] = 100;
    stockDate[2] = new DateTime(2001, 3, 1); stockName[2] =
    "Web Winner";
    stockPrice[2] = 20; stockShares[2] = 300;
    stockDate[3] = new DateTime(2003, 4, 10); stockName[3] =
    "Little Blue";
    stockPrice[3] = 15; stockShares[3] = 200;
    stockDate[4] = new DateTime(1999, 5, 21); stockName[4] =
    "My Company";
    stockPrice[4] = 40; stockShares[4] = 400;
    stockDate[5] = new DateTime(2000, 11, 1); stockName[5] =
    "Your Company";
    stockPrice[5] = 30; stockShares[5] = 200;
    txtTodayDate.Text = DateTime.Today.ToShortDateString();
    currentStock = 0;
    this.Show();
    ShowStock();
}
```

The **btnPrevious_Click** event method:

```
private void btnPrevious_Click(object sender, EventArgs e)
{
    // display previous stock
    if (currentStock != 0)
    {
        currentStock--;
        ShowStock();
    }
}
```

The **btnNext_Click** event method:

```
private void btnNext_Click(object sender, EventArgs e)
{
    // display next stock
    if (currentStock != numberStocks - 1)
    {
        currentStock++;
        ShowStock();
    }
}
```

Next, we give the code for the **general method** called **ShowStock**. To type this in the code window, go to any line after an existing method. Type the framework for the method:

```
private void ShowStock()
{

```



Type code here

```
}
```

Type the code between the two curly braces following the header you typed.

The complete method is:

```
private void ShowStock()
{
    // Change displayed stock
    grpstock.Text = stockName[currentStock];
    txtDate.Text =
stockDate[currentStock].ToShortDateString();
    txtPrice.Text =
Convert.ToString(stockPrice[currentStock]);
    txtShares.Text =
Convert.ToString(stockShares[currentStock]);
    txtPaid.Text = String.Format("{0:f2}",
stockPrice[currentStock] * stockShares[currentStock]);
    // Allow computation of return
    txtToday.Text = "";
    txtValue.Text = "0.00";
    txtReturn.Text = "0.00%";
    txtToday.Focus();
}
```

The `txtToday_KeyPress` event method:

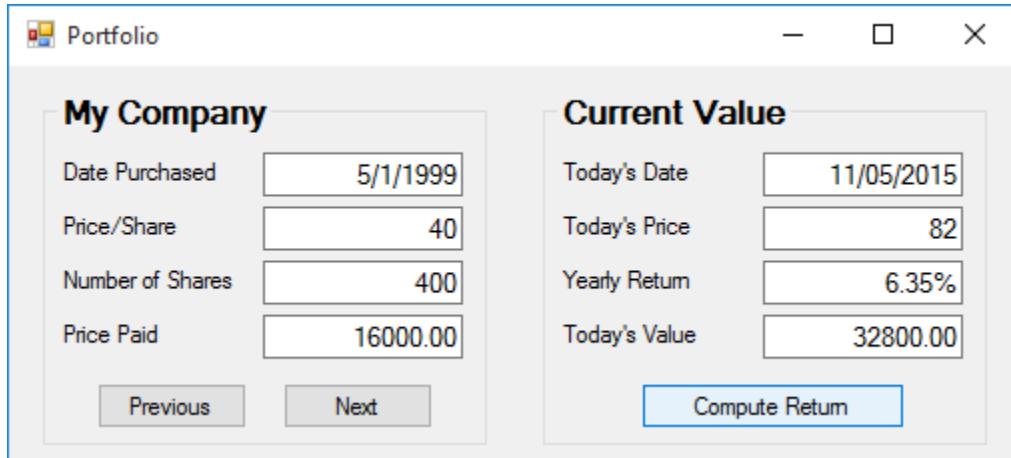
```
private void txtToday_KeyPress(object sender,
KeyPressEventArgs e)
{
    // Only allow numbers, decimal backspace
    if ((int) e.KeyChar == 13)
    {
        // Return key pressed
        btnReturn.PerformClick();
    }
    else if ((e.KeyChar >= '0' && e.KeyChar <= '9') ||
e.KeyChar == '.' || (int) e.KeyChar == 8)
    {
        e.Handled = false;
    }
    else
    {
        e.Handled = true;
    }
}
```

The **btnReturn_Click** event method:

```
private void btnReturn_Click(object sender, EventArgs e)
{
    // compute todays value and percent return
    double p, v, r;
    p = Convert.ToDouble(txtToday.Text);
    v = p * stockShares[currentStock];
    txtValue.Text = String.Format("{0:f2}", v);
    // Daily increase
    TimeSpan diff = DateTime.Today - stockDate[currentStock];
    r = (v / Convert.ToDouble(txtPaid.Text) - 1) / diff.Days;
    // Yearly return
    r = 100 * (365 * r);
    txtReturn.Text = String.Format("{0:f2}", r) + "%";
}
```

Run the Project

Save your work. Run the project. Click the **Previous** and **Next** buttons to view the five stocks stored in the program (you can edit this information in the **Form1_Load** method to reflect your holdings). Make sure you understand the use of the general method (**ShowStock**). For a particular stock, type the current selling price in the displayed text box and click **Compute Return**. The yearly return percentage and current value of that particular stock to your portfolio is displayed. Here's a run I made:



The screenshot shows a Windows application window titled "Portfolio". The window is divided into two main sections: "My Company" and "Current Value".

My Company

Date Purchased	5/1/1999
Price/Share	40
Number of Shares	400
Price Paid	16000.00

Below the table are two buttons: "Previous" and "Next".

Current Value

Today's Date	11/05/2015
Today's Price	82
Yearly Return	6.35%
Today's Value	32800.00

Below the table is a button: "Compute Return".

Other Things to Try

It's a hassle to have to store your holdings in the various arrays. You have to change the code every time you buy new stock or sell old stock. It would be nice to be able to save your holding information on a disk file. Then, it could be read in each time you run the program and any changes saved back to disk. With such saving capabilities, you could also modify the program to allow changing the number of shares you hold of a particular stock, allow addition of new stocks and deletion of old stocks. Accessing files on disk is an advanced topic you might like to study.

As written, the program gives returns on individual stocks. Try to write a summary function that computes the overall return on all the stocks in your current portfolio. I'm sure you can think of other changes to this program. Try them out.