A201/A505 Laboratory Session #16

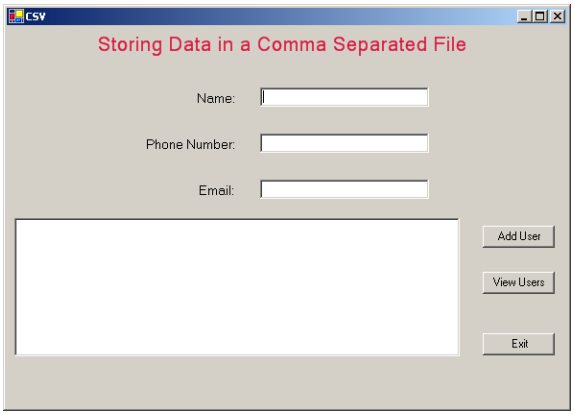
LAB GOALS

To learn about files:
Storing and retrieving data to/from a comma separated value (CSV) file.

Step 1: Create a new Windows project named “CSV”. Add the following lines at the beginning of your program:

```
Option Explicit On
Option Strict On
Imports System.IO  'Import the IO package so that we can do file I/O
```

Step 2: Create the Following GUI:



3 Buttons
- Add Contact
- View Contacts
- Exit

4 Labels
- Name
- Phone Number
- Email
- and a label for the heading....

3 Textboxes
- To correspond with the name, phone and email labels.

1 Listbox
- To display the contacts after they are read from the file.

Step 3: Double Click the “Exit” Button : Create a proper event handler to exist the application:

Step 4: Compile and run the program. Make sure the program compiles and runs at this point.

Step 5: Double Click the “Add User” Button: Type the following code in the event handler.

```
        If txtName.Text = "" Then
            MessageBox.Show("You must enter a name!", "Enter Name")
            txtName.Focus()
        ElseIf txtPhone.Text = "" Then
            MessageBox.Show("You must enter a Phone Number!", "Enter Phone")
            txtPhone.Focus()
        ElseIf txtEmail.Text = "" Then
            MessageBox.Show("You must enter a Email!", "Enter Email")
            txtEmail.Focus()
        Else
            Dim strNewLine As String

            'Concatenate the content of the three string together and put it in a new string
            strNewLine = txtName.Text.Trim & "," & txtPhone.Text.Trim & "," & Me.txtEmail.Text.Trim

            Const Append As Boolean = True

            'Open the file for writing (specifically for “Append” which means we are writing to the end of the file)
            Dim OutputFile As New StreamWriter("Contacts.txt", Append)
            OutputFile.WriteLine(strNewLine)
            OutputFile.Close()

            txtName.Clear()                 'Clear the Textboxes
            txtPhone.Clear()
            txtEmail.Clear()

            txtName.Focus()                 'Put the focus (the cursor) in the Name Textboxs
        End If
```

The above event handler takes the information provided by the user and will write that information to a CSV file. Note that the above event handler must first make sure that the contact information (name, phone number and Email) have been provided in the textboxes before proceeding to add the information to the contact files (See yellow highlight). Once the textboxes are verified, our code will simply concatenate the text in each of the 3 text boxes (including commas between each textbox value) and places the result in a string. (See green highlight). Next we open the "contacts.txt" file in Aappend@ mode and write the string to the file, and close the file. (See gray highlight). Finally, we clear the textboxes to prepare the program for the next contact to be added. (See pink highlight)

Step 6: Compile and run the program: When the program executes, first click the "Add Contact" button without entering any information in the textboxes. What is your observation? After you are convinced that your input validation is working properly, enter a name, phone number and email into the appropriate textboxes and press the "Add Contact" button. At this point the data should be stored in the "contacts.txt" file.

Step 7: Verify that the contact information is properly stored: After pressing the "Add Contact" and adding the information to the file, you should manually verify the "contact.txt" file to see if it is properly created and if it has the proper information in it. (This file should be created under the "Bin/Debug" directory of your project). You can open this file using the windows "notepad" or some other text editor program. Once you verified that the data is in the file, close the notepad.

Step 8: Add a few more contacts by typing the contact information in the textboxes and then clicking the "Add Contact" button.

Step 9: Double Click the "View Contacts" Button: The event handler for this button must read the "contacts.txt" file and display the contact information in the list box. Type the following code in the event handler.

```
Private Sub btnView_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnView.Click
    Try

        'Open the contacts.txt file for reading

        Dim InputFile As StreamReader = New StreamReader("Contacts.txt")
        lstContacts.Items.Clear()

        Dim strNewLine As String

        Do Until InputFile.Peek() = -1
            strNewLine = InputFile.ReadLine()            'Read a line from the contacts.txt file

            lstContacts.Items.Add(strNewLine)            'Add the line to the ListBox
        Loop

        InputFile.Close()

    Catch
        MessageBox.Show("File could not be open/found!", "I/O Error")
    End Try
End Sub
```

Step 10: Modify the above code to produce a formatted output: Add the highlights below and observe the difference.

```
Private Sub btnView_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnView.Click
    Try
        Dim InputFile As StreamReader = New StreamReader("Contacts.txt")

        Dim fmtStr As String = "{0, -20}{1, 15}{2, 25}"

        lstContacts.Items.Clear()
        lstContacts.Items.Add(String.Format(fmtStr, "Name", "Phone     ", "Email    "))

        Dim strNewLine As String
        Dim strValues() As String          ' array of strings

        Do Until InputFile.Peek() = -1
            strNewLine = InputFile.ReadLine()

            'Split the input line into multiple strings
            'using the "," as the delimiter character
            strValues = Split(strNewLine, ",")

            lstContacts.Items.Add(String.Format(fmtStr, strValues))
        Loop

        InputFile.Close()

    Catch
        MessageBox.Show("File could not be open/found!", "I/O Error")
    End Try
End Sub
```

NOTE:
In the code above, in first highlighted line we are creating a format string: "fmtStr"

```
Dim fmtStr As String = "{0, -20}{1, 15}{2, 25}"
```

The format string will take its first argument and left justify its content within 20 spaces: {0, -20}
The second argument is right justified within 15 spaces: {1, 15}
Finally, the third argument is right justified within 25 spaces: {2, 25}

The second highlighted line:

```
lstContacts.Items.Add(String.Format(fmtStr, "Name", "Phone     ", "Email    "))
```

The third highlighted line:

```
Dim strValues() As String                ' Create an array of strings
```

The fourth highlighted line:

```
strValues = Split(strNewLine, ",")       ' Split the strNewLine into multiple strings using the "," as the delimiter
                                         character.  Place each string in separate elements of the strValues array.
```

The fifth highlighted line:

```
lstContacts.Items.Add(String.Format(fmtStr, strValues))
```

Creates a formatted string which includes the Name, Phone, and Email (extracted from the strValues array) and adds the string to listbox.