Input/Output

Introduction

• Formatting Numbers
• Input From Files
• Output To Files
• Input Boxes
Formatting Numbers

• Display numbers in familiar formats
• Examples:
  
  FormatNumber(12345.678, 1) → 12,345.6
  Display “n” rounded to “r” decimal places and display with commas
  FormatCurrency(12345.678, 2) → $12,345.68
  Display dollar sign then “n” to “r” decimal places and display with commas
  FormatPercent(0.185, 2) → 18.5%
  Display “n” as a percent rounded to to “r” decimal places and percent sign

How Do I Use That?

```vbnet
Public Class Form1

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    ListBox1.Items.Add(12345.678)
    ListBox1.Items.Add(FormatNumber(12345.678, 1))
    ListBox1.Items.Add(FormatCurrency(12345.678, 2))
    ListBox1.Items.Add(FormatPercent(0.185, 2))
End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
    End
End Sub
End Class
```
Input From Files

• Three ways:
  – Assign values to variables in program
  – Read values from a file
  – Enter values in text box

Files and Folders

• File: holds programs or data. Its name usually consists of letters, digits, and spaces.

• Filenames:
  – Up to 156 characters (letters, digits, etc.)
  – Extension (.txt, .doc, etc)

• Pathnames:
  – Where does a file live?
  – Path to the folder containing a file: C:\temp\
Key Terms: Folders and Files

<table>
<thead>
<tr>
<th>Term</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk</td>
<td>Hard disk, flash drive, DVD</td>
</tr>
<tr>
<td>File name</td>
<td>Payroll</td>
</tr>
<tr>
<td>Extension</td>
<td>.txt</td>
</tr>
<tr>
<td>Filename</td>
<td>Payroll.txt</td>
</tr>
<tr>
<td>Path</td>
<td>TextFiles\Payroll.txt</td>
</tr>
<tr>
<td>Filespec</td>
<td>C:\TextFiles\Payroll.txt</td>
</tr>
</tbody>
</table>

CSV File Format

- Comma Separated Values (CSV)
- Records are stored on one line with a comma between each field
- Example: USStates.txt
  
  Delaware,DE,1954,759000  
  Pennsylvania,PA,44817,12296000  
  New Jersey,NJ,7417,8135000  
  Georgia,GA,57906,7637000  

(name of state, abbreviation, area, population)
Sample CSV File

First two lines of USStates.txt

Delaware,DE,1954,759000
Pennsylvania,PA,44817,12296000

ame  abbreviation  area  population

Example w/o Files

This is painful!
Open & Read Files

- Data stored in a text file can be read one line at a time with a StreamReader object. Declare a variable that can read from files:

  ```vbnet
  Dim sr As IO.StreamReader = IO.File.OpenText(filespec)
  ```
  - `filespec` is the file to be read
  - A pointer is set to the first line of the file.

- Read data from the file

  ```vbnet
  str = sr.ReadLine
  ```
  - Reads the line pointed to, assigns the line to the string variable `str`, and moves the pointer to the next line of the file.
  - The value of `str.EndOfStream` will be `True` after the entire file has been read.
  - The statement `str.Close()` terminates communication with the file.

Split Operator

- The Split operator provides a way to recover the values from a CSV file.

  ```vbnet
  text = line.Split(“,“)
  ```

- This “splits” the string variable “line” into pieces. The splits are made where a comma appears. The pieces are assigned to the text variable “text()”.

- `text(0)` will have the text preceding the 1st comma, `text(1)` will have the text between the 1st and 2nd commas, etc.
Sample CSV File

First line of USStates.txt

\[
\text{line} = \text{“Delaware,DE,1954,759000”}
\]

\[
\text{text} = \text{line.Split(“,”)}
\]

<table>
<thead>
<tr>
<th>Delaware</th>
<th>DE</th>
<th>1954</th>
<th>759000</th>
</tr>
</thead>
</table>

Example w/Files

```vba
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Dim name As String, abbreviation As String
    Dim area As Integer, population As Integer

    Dim line As String, text() As String
    Dim sr As IO.StreamReader
    sr = IO.File.OpenText("C:\temp\CE311K\USStates.txt")

    line = sr.ReadLine
    text = line.Split(“,”)
    name = text(0)
    abbreviation = text(1)
    area = CInt(text(2))
    population = CInt(text(3))

    ListBox1.Items.Add(name)
    ListBox1.Items.Add(abbreviation)
    ListBox1.Items.Add(area)
    ListBox1.Items.Add(population)

    sr.Close()
    sr = Nothing
End Sub
```
Getting Input from an Input Dialog Box

```vba
stringVar = InputBox(prompt, title)
fullName = InputBox("Enter your full name.", "Name")
```

**Example w/Files**

```vba
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Dim name As String, abbreviation As String
    Dim area As Integer, population As Integer

    Dim filename As String, prompt As String, title As String
    title = "Name of file:"
    filename = InputBox(prompt, title)

    Dim line As String, text() As String
    Dim sr As StreamReader
    sr = IO.File.OpenText(filename)

    line = sr.ReadLine
    text = line.Split(”,”)
    name = text(0)
    abbreviation = text(1)
    area = CInt(text(2))
    population = CInt(text(3))
```

This is painful!
The **OpenFileDialog** Control

- Implements File Open dialog box
- Found in *Dialogs* group of Toolbox
- Icon will appear below Document window.

Using the **OpenFileDialog** Control

- To display the control:
  ```csharp
  OpenFileDialog1.ShowDialog()
  ```

- After the *Open* button has been pressed, the file name selected and its complete filespec will be contained in the property:

  ```csharp
  OpenFileDialog1.FileName
  ```
Example w/Files

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Dim name As String, abbreviation As String, area As Integer, population As Integer
    Dim filename As String = OpenFileDialog1.FileName

    Dim line As String, text As String
    Dim sr As IO.StreamReader
    sr = IO.File.OpenText(filename)

    line = sr.ReadLine
    text = line.Split(CrLf)

    name = text(0)
    abbreviation = text(1)
    area = CInt(text(2))
    population = CInt(text(3))

    ListBox1.Items.Add(name)
    ListBox1.Items.Add(abbreviation)
    ListBox1.Items.Add(area)
    ListBox1.Items.Add(population)
End Sub

Using a Message Dialog Box for Output

MessageBox.Show(prompt, title)

MessageBox.Show("Nice try, but no cigar.", "Consolation")
Open & Write a File

- Data can be placed in a text file one line at a time with a `StreamWriter` object.
  
  ```vba
  Dim sw As IO.StreamWriter = IO.File.CreateText(filespec)
  -- filespec is the file to be created and written
  -- Establishes link between computer and disk drive for writing
  ``

- Write data to the file
  
  ```vba
  sw.WriteLine(info)
  -- initially places the information (info) into the first line of the file
  -- Subsequent statements of that form place information into lines at the end of the file
  ``

- The statement `sw.Close()` terminates communication with the file.

Write to a File

```vba
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Dim name As String, abbreviation As String
Dim area As Integer, population As Integer

Dim filename As String, prompt As String, title As String
filename = InputBox(prompt, title)
Dim sw As IO.StreamWriter
sw = IO.File.CreateText(filename)
sw.WriteLine(name)
sw.WriteLine(abbreviation)
sw.WriteLine(area)
sw.WriteLine(population)
sw.Close()
End Sub
```
Write to a File

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
  Dim name As String, abbreviation As String
  Dim area As Integer, population As Integer
  Dim filename As String
  Dim line As String, text() As String

  OpenFileDialog.ShowDialog()
  filename = OpenFileDialog1.FileName
  Dim sr As IO.StreamReader = IO.File.OpenText(filename)

  line = sr.ReadLine
  text = line.Split(","c)
  name = text(0)
  abbreviation = text(1)
  area = CInt(text(2))
  population = CInt(text(3))

  OpenFileDialog.ShowDialog()
  filename = OpenFileDialog1.FileName
  Dim sw As IO.StreamWriter = IO.File.CreateText(filename)

  sw.WriteLine(name)
  sw.WriteLine(abbreviation)
  sw.WriteLine(area)
  sw.WriteLine(population)
  sw.Close()
End Sub

Using Text Boxes for I/O

• The content of a text box is always a string
• We can use the contents of a text box to assign values to variables
• We can use the contents of a variable to assign values to text boxes
• Numbers typed into text boxes are stored as strings and should be converted to numbers before using in calculations
Example

Private Sub Button1_Click(ByVal sender As System.Object,
    ByVal e As System.EventArgs) Handles Button1.Click

    num1 = CDbl(TextBox1.Text)
    num2 = CDbl(TextBox2.Text)
    sum = num1 + num2
    TextBox3.Text = CStr(sum)

End Sub

Summary

• Formatting Numbers
• Input From Files
• Output To Files
• Input Boxes