

Fundamentals of Programming in Visual Basic – Part 2

Robert Burtch
Surveying Engineering Department
Ferris State University

VARIABLES AND STRINGS

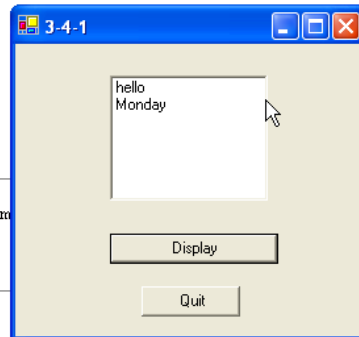
- String variable – name used to refer to a string
 - Allowable names same as for number
- Declared in the following form
`Dim varName As String`
- To assign string literal to string variable
`strVar = "xy...z"`
- To display string in a list box:
`lstBox.Items.Add ("xy...z")`
- Or
`lstBox.Items.Add (strVar)`

VARIABLES AND STRINGS

- Example:

```
Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Dim today As String
    today = "Monday"
    With lstOutput.Items
        .Clear()
        .Add("hello")
        .Add(today)
    End With
End Sub

Private Sub btnQuit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    End
End Sub
Class
```



VARIABLES AND STRINGS

- To assign value from one variable to another

strVar2 = strVar

- String literals used in assignment or `lstBox.Items.Add` statements must be surrounded by quotation marks
 - String variables never surrounded by quotation marks

USING TEXT BOXES FOR INPUT/OUTPUT

- Contents of text box always a string
- To assign contents of text box to string variable strVar:

```
strVar = txtBox.Text
```

- To assign contents of string variable to text box:

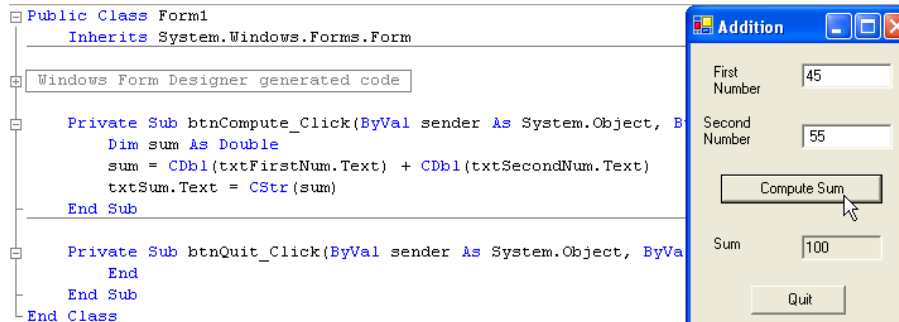
```
txtBox.Text = strVar
```

EXAMPLE

- Create form with the following objects and settings

OBJECT	PROPERTY	SETTING
Form1	Text	Addition
lblFirstNum	Text	First Number:
txtFirstNum	Text	(blank)
lblSecondNum	Text	Second Number
txtSecondNum	Text	(blank)
btnCompute	Text	Compute Sum
lblSum	Text	Sum:
txtSum	ReadOnly	True
btnQuit	Text	Quit

USING TEXT BOXES FOR INPUT/OUTPUT EXAMPLE



USING TEXT BOXES FOR INPUT/OUTPUT

- Numbers typed in text boxes stored as strings
 - Must be converted to numeric format
 - Data-conversion or type-casting functions
 - Cdbl – converts string to Double
 - CInt – converts string to Integer
 - CStr – converts number into string representation
 - Example:

`numVar = Cdbl(txtBox.Text)`

Assigns contents of text box to Double variable numVar

`txtBox.Text = CStr(numVar)`

Assigns contents of numVar to string representation
txtBox

CONCATENATION

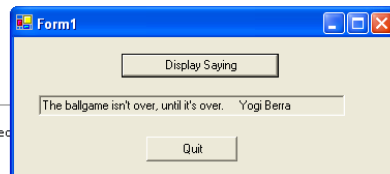
- Operation of joining/combining two strings to form a new string consisting of the strings joined together
- Represented by ampersand (&)
- String expression – combination of strings and ampersand that can be evaluated to form a string

CONCATENATION - EXAMPLE

- Example of simple concatenation:

```
Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnD:
    Dim quote1, quote2, quote As String
    quote1 = "The ballgame isn't over, "
    quote2 = "until it's over."
    quote = quote1 & quote2
    txtOutput.Text = quote & "    Yogi Berra"
End Sub

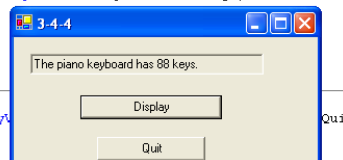
Private Sub btnQuit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnQ:
    End
End Sub
```



- Example concatenation of string with

```
Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnD:
    Dim str As String, numOfKeys As Double
    str = "The piano keyboard has "
    numOfKeys = 88
    txtOutput.Text = str & numOfKeys & " keys."
End Sub

Private Sub btnQuit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnQ:
    End
End Sub
```



ANSI CHARACTER SET

- Assigns characters to numbers from 32 – 255
- If n is number between 32 – 255, **Chr(n)** is a string consisting of characters with ANSI value n
- If str is any string, **ASC(str)** is ANSI value for the first character of str
- Example

```
txtBox.Text = Chr(65)
```

displays letter A in text box

```
lstBox.Items.Add (ASC("Apple"))
```

displays number 65 in list box

ANSI CHARACTER SET

- Portion of standard ANSI/ASCII code:

0	32 [space]	64 @	96 `
1	33 !	65 A	97 a
2	34 "	66 B	98 b
3	35 #	67 C	99 c
4	36 \$	68 D	100 d
5	37 %	69 E	101 e
6	38 &	70 F	102 f
7	39 '	71 G	103 g
8 **	40 (72 H	104 h
9 **	41)	73 I	105 i
10 **	42 *	74 J	106 j
11	43 +	75 K	107 k
12	44 ,	76 L	108 l
13 **	45 -	77 M	109 m
14	46 .	78 N	110 n
15	47 /	79 o	111 o
16	48 0	80 P	112 p
17	49 1	81 Q	113 q
18	50 2	82 R	114 r
19	51 3	83 S	115 s
20	52 4	84 T	116 t
21	53 5	85 U	117 u
22	54 6	86 V	118 v
23	55 7	87 W	119 w
24	56 8	88 X	120 x
25	57 9	89 Y	121 y
26	58 :	90 Z	122 z

ANSI CHARACTER SET

- Extended ANSI code

128	160 [space]	192 À	224 à
129	161 ¡	193 Á	225 á
130	162 ¢	194 Â	226 â
131	163 £	195 Ã	227 ã
132	164 ¤	196 Ä	228 ä
133	165 ¥	197 Å	229 å
134	166 ¦	198 Æ	230 æ
135	167 §	199 Ç	231 ç
136	168 ¨	200 È	232 è
137	169 ©	201 É	233 é
138	170 ª	202 Ê	234 ê
139	171 «	203 Ë	235 ë
140	172 ¬	204 Ì	236 ì
141	173 ®	205 Í	237 í
142	174 ¯	206 Î	238 î
143	175 °	207 Ï	239 ï
144	176 ±	208 Ð	240 ð
145	177 º	209 Ñ	241 ñ
146	178 ²	210 Ò	242 ò
147	179 ³	211 Ó	243 ó
148	180 ´	212 Ô	244 ô
149	181 µ	213 Õ	245 õ
150	182 ¶	214 Ö	246 ö
151	183 ·	215 ×	247 ×
152	184 ¸	216 Ø	248 ø
153	185 ¹	217 Ù	249 ù

ANSI CHARACTER SET

- Can concatenate with Chr

```
txtBox.Text = "32" & Chr(176) & "
    Fahrenheit"
```

displays 32° Fahrenheit in Text box

- Example of incorporating quotation mark in a string of text output

```
txtBox.Text = "George" & Chr(34) & "Babe" & Chr(34) & "
    Ruth"
```

when executed, text box displays:

George "Babe" Ruth

EMPTY STRING

- String containing no characters
- "" is not the same as " "
 - Latter form has space
- listBox.Items.Add("") skips line in list box
- Contents of text box can be cleared by
txtBox.Clear()

or

```
txtBox.Text = ""
```

INITIAL VALUE OF STRING

- When declared with Dim statement, string variable has default keyword *Nothing* as default value
- To declare different initial value
Dim today As String = "Tuesday"
- When performing operation with value *Nothing* – error occurs
 - Need to initialize

PROGRAM DOCUMENTATION

- Helpful to add comments to the program
- Documents
 - Intent of program
 - Purpose of variables
 - Tasks performed by individual portions of program
- Comment statement – begin line with apostrophe
 - Statement ignored when program executed
- Line of code can be documented by adding apostrophe followed by desired information after the end of the line
- Comments appear as green on screen

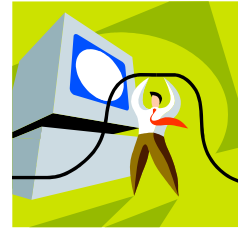
EXAMPLE PROGRAM DOCUMENTATION

```
Private Sub btnAnalyze_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnAn
    'Determine a person's first name and the length of the second name
    Dim fullName, firstName, lastName As String
    Dim n As Integer 'location of the space separating the two names
    fullName = txtName.Text
    n = fullName.IndexOf(" ")
    firstName = fullName.Substring(0, n)
    lastName = fullName.Substring(n + 1)
    'Display the desired information in a list box
    With lstResults.Items
        .Clear()
        .Add("First name: " & firstName)
        .Add("Your last name has " & lastName.Length & " letters.")
    End With
End Sub
```

PROGRAM DOCUMENTATION

- Benefits

- Other people can easily understand program
- You can understand program when you read it later
- Long programs easier to read because the purposes of individual pieces can be determined at a glance



LINE-CONTINUATION CHARACTER

- Long statements can be split across two or more lines by ending each line (except the last) with underscore (`_`) preceded by space
- Example:
msg = "640K ought to be enough for anybody. (Bill Gates, 1981)"
can be written as:
msg = "640K ought to be enough for " & _
"anybody. (Bill Gates, 1981)"
- Make sure underscore does not appear inside quotation marks
- Does not work with comment statements

FORMATTING OUTPUT WITH FORMAT FUNCTIONS

- Example Format functions

FUNCTION	STRING VALUE
FormatNumber(12345.628, 1)	12,345.6
FormatCurrency(12345.628, 2)	\$12,345.63
FormatPercent(0.185, 2)	18.50%

FORMATTING OUTPUT WITH FORMAT FUNCTIONS

- FormatNumber (n,r) – string containing number n rounded to r decimal places
 - Displayed with thousands separators
- FormatCurrency (n,r) – consists of dollar sign followed by value
 - Uses accountant’s convention – negative values surrounded by parenthesis
- FormatPercent (n,r) consists of number n displayed as percent rounded to r decimal places
- Examples:

FormatNumber (1 + Math.Sqrt(2),3)	2.414
FormatCurrency (-1000) (\$1,000.00)	
FormatPercent (“.05”)	5.00%

FORMATTING OUTPUT WITH ZONES

- Display data in tabular form
 - 1) Used fixed-width font like Courier New
 - 2) Divides characters into zones with format string
- Format string form:
Dim fmtStr As String = "{0,15}{1,10}{2,8}"
 - List box divided into zones of 15, 10 and 8 character width
 - Left most zone referred to as zone 0, etc.

FORMATTING OUTPUT WITH ZONES



- If data0, data1 & data2 are strings or numbers, statement prints out as

```
lstOutput.Items.Add (String.Format (fmtStr, data0, data1,  
data2))
```

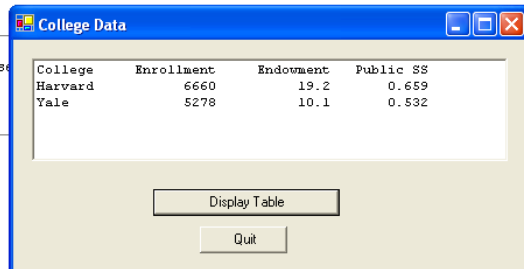
- If any number preceded with minus sign – data left justified
- No limit on number of zone and each can have different widths

FORMATTING OUTPUT WITH ZONES

- Example:

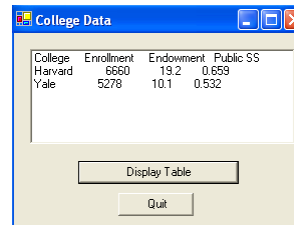
```
Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) |
    Dim fmtstr As String = "(0,-10)(1,12)(2,14)(3,12)"
    With lstColleges.Items
        .Clear()
        .Add(String.Format(fmtstr, "College", "Enrollment", _
            "Endowment", "Public SS"))
        .Add(String.Format(fmtstr, "Harvard", 6660, 19.2, 0.659))
        .Add(String.Format(fmtstr, "Yale", 5278, 10.1, 0.532))
    End With
End Sub

Private Sub btnQuit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) |
    End
End Sub
End Class
```



FORMATTING OUTPUT WITH ZONES

- Example with Sans Serif format



- Placing colon & formatting symbol after width – specific format of numeric data (N or number, C for currency, P for percent) followed by object specifying number of decimal places

FORMATTING OUTPUT WITH ZONES

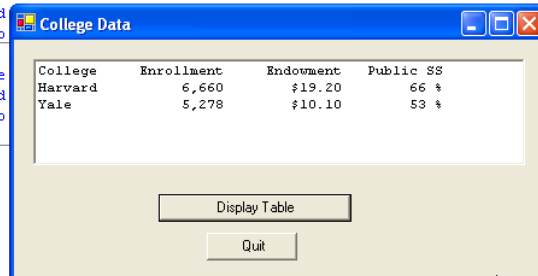
ZONE FORMAT TERM	NUMBER TO BE FORMATTED	NUMBER DISPLAYED
{1,12:N3}	1234.5679	1,234.568
{1,12:N0}	34.6	34
{1,12:C1}	1234.567	\$1,234.6
{1,12:P}	0.569	56.90%

Space between curly brackets – corresponding zones in output separated by those number of spaces

FORMATTING OUTPUT WITH ZONES

- Example of zone format term

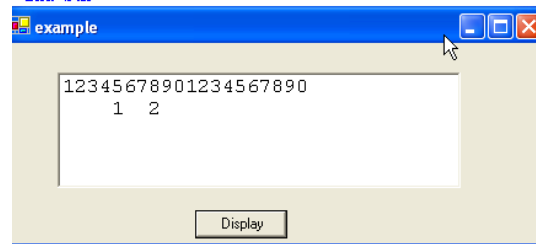
```
Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e
    Dim fmtStr As String = "{0,-10}{1,12:N0}{2,14:C}{3,12:PO}"
    With lstColleges.Items
        .Clear()
        .Add(String.Format(fmtStr, "College", "Enrollment", _
            "Endowment", "Public SS"))
        .Add(String.Format(fmtStr, "Harvard", 6660, 19.2, 0.659))
        .Add(String.Format(fmtStr, "Yale", 5278, 10.1, 0.532))
    End With
End Sub
```



FORMATTING OUTPUT WITH ZONES

- Example with spaces between curly

```
{ Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnDisplay.Click
    Dim fmtStr As String = "{0,5} {1,-5}" 'Two spaces after the first right curly bracket
    With lstOutput.Items
        .Add("12345678901234567890")
        .Add(String.Format(fmtStr, 1, 2))
    End With
End Sub
```



READING DATA FROM FILES

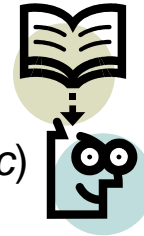
- Data can be stored in files and accessed by user with a StreamReader object or supplied by user with input dialog box
- Data read sequentially and assigned to variable with following steps

1) Execute statement

```
Dim readerVar As IO.StreamReader
```

- StreamReader – object from Input/Output class that can read stream of characters from disk or over internet
- Dim statement – declares variable readerVar to be type StreamReader

READING DATA FROM FILES



2) Execute statement

```
readerVar = IO.File.OpenText(filespec)
```

- *filespec* identifies file to be read
- Statement establishes communications link between computers and disk drive for reading data
- Statement can be combined in Dim statement

```
Dim readerVar As IO.StreamReader =  
    IO.File.OpenText(filespec)
```

READING DATA FROM FILES

3) Read items in data file one at a time with ReadLine method

```
strVar = readVar.ReadLine
```

- Program looks for next unread line of data and assigns it to variable strVar
- Data can be assigned to numeric variable – first need conversion like

```
numVar = Cdbl (readerVar.ReadLine)
```

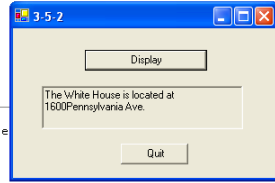
4) After all data read, terminate communications link

```
readerVar.Close()
```

READING DATA FROM FILES

```
Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnD:
    Dim houseNumber As Double
    Dim street, address As String
    houseNumber = 1600
    street = "Pennsylvania Ave."
    address = houseNumber & " " & street
    txtAddr.Text = "The White House is located at " & address
End Sub

Private Sub btnQuit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnQuit:
    End
End Sub
End Class
```

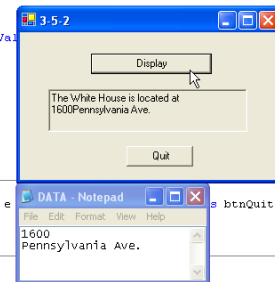


- Example: data assigned v. data read in

Windows Form Designer generated code

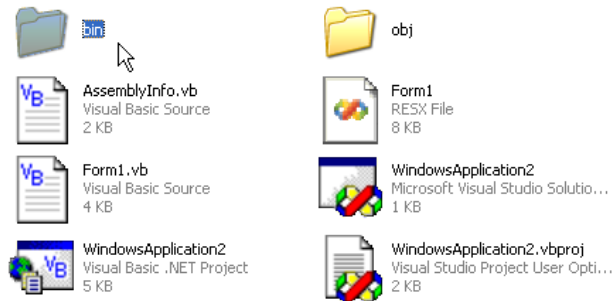
```
Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnD:
    Dim sr As IO.StreamReader = IO.File.OpenText("A:DATA.TXT")
    Dim houseNumber As Double
    Dim street, address As String
    houseNumber = CDb1(sr.ReadLine)
    street = sr.ReadLine
    sr.Close()
    address = houseNumber & " " & street
    txtAddr.Text = "The White House is located at " & address
End Sub

Private Sub btnQuit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnQuit:
    End
End Sub
End Class
```



READING DATA FROM FILES

- Need path provided
 - If blank, VB.NET will look in bin subdirectory
- VB.NET file structure for one procedure



READING DATA FROM FILES

- Example reading in data file

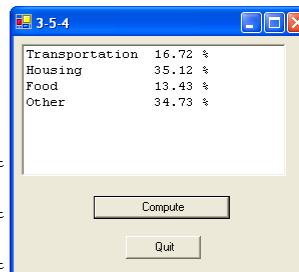
```
Private Sub btnCompute_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btn
    Dim sr As IO.StreamReader = IO.File.OpenText("PAYROLL.TXT")
    'The file PAYROLL.TXT is in the bin subfolder
    Dim name As String
    Dim hourlyWage, hoursWorked, salary As Double
    name = sr.ReadLine
    hourlyWage = CDb1(sr.ReadLine)
    hoursWorked = CDb1(sr.ReadLine)
    salary = hourlyWage * hoursWorked
    lstPayroll.Items.Add(name & " " & FormatCurrency(salary))
    name = sr.ReadLine
    hourlyWage = CDb1(sr.ReadLine)
    hoursWorked = CDb1(sr.ReadLine)
    sr.Close()
    salary = hourlyWage * hoursWorked
    lstPayroll.Items.Add(name & " " & FormatCurrency(salary))
End Sub
```



READING DATA FROM FILES

- Example including incrementing variable

```
Private Sub btnCompute_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnC
    Dim sr As IO.StreamReader = IO.File.OpenText("COSTS.TXT")
    Dim total As Double
    Dim category As String, amount As Double
    Dim fmtStr As String = "(0,-15)(1,8:F)"
    category = sr.ReadLine
    total += CDb1(sr.ReadLine)
    'associated with the category
    category = sr.ReadLine
    total += CDb1(sr.ReadLine)
    category = sr.ReadLine
    total += CDb1(sr.ReadLine)
    category = sr.ReadLine
    total += CDb1(sr.ReadLine)
    sr.Close()
    sr = IO.File.OpenText("COSTS.TXT")
    category = sr.ReadLine
    amount = CDb1(sr.ReadLine)
    lstPercent.Items.Add(String.Format(fmtStr, category, amount))
    category = sr.ReadLine
    amount = CDb1(sr.ReadLine)
    lstPercent.Items.Add(String.Format(fmtStr, category, amount))
    category = sr.ReadLine
    amount = CDb1(sr.ReadLine)
    lstPercent.Items.Add(String.Format(fmtStr, category, amount))
    category = sr.ReadLine
    amount = CDb1(sr.ReadLine)
    lstPercent.Items.Add(String.Format(fmtStr, category, amount / total))
    sr.Close()
End Sub
```



USING MESSAGE DIALOG BOX FOR OUTPUT

- Sometimes nice to grab user's attention with brief message

- Done with message dialog box

- Format:

MsgBox(prompt, , title)

- Example:

MsgBox("Nice try, but no cigar.", , "Consolation")



GETTING INPUT FROM INPUT DIALOG BOX

- Input dialog box form:

stringVar = InputBox(prompt, title)

- Example

```
Private Sub btnDisplay_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnD
    Dim sr As IO.StreamReader
    Dim fileName, prompt, title, street As String
    Dim houseNumber As Double
    prompt = "Enter the name of the file containing the information."
    title = "Name of File"
    fileName = InputBox(prompt, title)
    'We assume the file is located in the subfolder bin
    sr = IO.File.OpenText(fileName)
    houseNumber = CDBl(sr.ReadLine)
    street = sr.ReadLine
    sr.Close()
    txtAddr.Text = "The White House is located at " & _
        houseNumber & " " & street & "."
End Sub

Private Sub btnQuit
End
End Sub
End Class
```

