Visual Basic Reference Guide

# Quick Tutorial

* **NOTE:** Most references to functions can be found below, where you will find more info and examples.
* Use the **DisplayLine()** function to display a string to the program window.
  + This displays the output AND a newline character, i.e. the cursor is positioned on the next line.
  + Use the **Display()** function when you do NOT want a NewLine character output, i.e. when you want the cursor to stay on the same line as the output.
* Use the *string concatenation* character (&) when displaying multiple strings at once:
  + **DisplayLine("Name: " & sName)**
* Use the **ReadString()** function to allow the user to enter a string using the keyboard:
  + **sName = ReadString()**
  + Note that ReadString() must **always** be on the **right** side of an equal sign. On the left side, *you must have a variable* to store (capture) whatever string the user entered.
  + Use **ReadChar()** instead of ReadString() when you want your program to accept a single character ***immediately*** instead of waiting for the user to press Enter.
* Similarly, use the **ReadInteger()** function to allow the user to enter an *integer*:
  + **sAge = ReadInteger()**
  + Once again, note that ReadInteger() must **always** be on the **right** side of an equal sign. On the left side, *you must have a variable* to store (capture) whatever the user entered.
* Use the **ReadDecimal()** function to allow the user to enter a decimal number:
  + **sPrice = ReadDecimal()**
* Using the various **Formatting** functions, explained in detail in the References section of this document, you can change font color, size, bold, etc.

# References

## SetTitle()

Sets the form caption and the program title field at the top of the application.

**Parameters**

* String – the text entered by the user.

**Returns**

* Nothing

**Example**

SetTitle("Calculator")

Sets the caption and program title field to "Calculator".

## DisplayLine()

Displays any text passed into it including NewLine characters to display multiple lines. Appends a newline character to the end of text.

**Parameters**

* Optional String – the text to display. If not present, just displays a blank line.

**Returns**

* Nothing

**Example**

DisplayLine("Happy Birthday!")

Displays the text "Happy Birthday!" and moves the cursor to the next line.

**Example**

DisplayLine("Happy Birthday!" & Enter & "Have a great day!")

Displays the text "Happy Birthday!" on one line, then displays "Have a great day!" on the next line, and moves the cursor to the next line.

**Notes**

* **Enter** (as shown in the example above) is a pre-defined variable that simply means “go to the next line”. In other words, it inserts a line break. The variable **NL** (NewLine) will do the same thing.

**Example**

DisplayLine()

Displays a blank line.

## Display()

Displays any text passed into it including NewLine characters to display multiple lines. Unlike **DisplayLine()**, does NOT append a newline character to the end of text before displaying it.

**Parameters**

* String – the text to display.

**Returns**

* Nothing

**Example**

Display("Happy ")

Display("Birthday!")

Displays the text "Happy Birthday!" all on the same line, and keeps the cursor positioned at the end of the text.

## ReadString()

Reads a string of text input from the keyboard as soon as the user presses the Enter key.

**Parameters**

* None

**Returns**

* String – the text entered by the user.

**Notes**

* ReadString() must always be on the right side of a variable assignment, i.e. you need something to put the string ***into***, so that you can work with it.

**Example**

Display("Enter first name: ")

sFirstName = ReadString()

Displays the text "Enter first name: ", then waits for the user to enter a string and press Enter. The resulting text is assigned to the **sFirstName** variable.

## ReadChar()

Reads a single character from the keyboard as soon as the user enters the character. Does NOT wait for the Enter key to be pressed.

**Parameters**

* None

**Returns**

* String – the character entered by the user.

**Notes**

* ReadChar() must always be on the right side of a variable assignment, i.e. you need something to put the character ***into***, so that you can work with it.

**Example**

Display("Continue (Y/N): ")

sResponse = ReadChar()

Displays the text "Continue (Y/N): **"**, then waits for the user to enter a character. The resulting character is assigned to the **sResponse** variable.

## ReadInteger()

Reads an integer from the keyboard as soon as the user presses the Enter key.

**Parameters**

* None

**Returns**

* Integer – the number entered by the user.

**Notes**

* Accepts positive and negative integers (no decimals).
* ReadInteger() must always be on the right side of a variable assignment, i.e. you need something to put the number ***into***, so that you can work with it. The variable must be of type **Integer**.

**Example**

Dim iAmount as Integer

Display("Enter item amount: ")

iAmount = ReadInteger()

Displays the text "Enter item amount: ", then waits for the user to enter a number and press Enter. The resulting number is assigned to the **iAmount** variable.

## ReadDecimal()

Reads a decimal number from the keyboard as soon as the user presses the Enter key.

**Parameters**

* None

**Returns**

* Double – the number entered by the user.

**Notes**

* Accepts decimal numbers.
* ReadDecimal() must always be on the right side of a variable assignment, i.e. you need something to put the number ***into***, so that you can work with it.
* **NOTE:** The variable must be of type **Double**, **NOT** **Decimal**.

**Example**

Dim dAmount as Double

Display("Enter item amount: ")

dAmount = ReadDecimal()

Displays the text "Enter item amount: ", then waits for the user to enter a number and press Enter. The resulting number is assigned to the **dAmount** variable.

## FormatCurrency()

Formats a number for output, with specified decimal places, and comma separators, and the currency symbol ($ in the U.S.)

**Parameters**

* Number – the number to be formatted.
* Optional Integer – the number of digits after the decimal. If omitted, then the number 2 is assumed.

**Returns**

* String – the number formatted as a string

**Example**

Dim iTestNumber As Integer = 45600

Dim sTestString As String = FormatCurrency(iTestNumber)

'sTestString will contain "$45,600.00"

**Example**

Dim iTestNumber As Integer = 45600

Dim sTestString As String = FormatCurrency(iTestNumber, 1)

'sTestString will contain "$45,600.0"

## FormatNumber()

Formats a number for output, with specified decimal places, and comma separators

**Parameters**

* Number – the number to be formatted.
* Optional Integer – the number of digits after the decimal. If omitted, then the number 2 is assumed.

**Returns**

* String – the number formatted as a string

**Example**

Dim iTestNumber As Integer = 45600

Dim sTestString As String = FormatNumber(iTestNumber)

'sTestString will contain "45,600.00"

**Example**

Dim iTestNumber As Integer = 45600

DisplayLine("My number is: " & FormatNumber(iTestNumber))

'will display "My number is: 45,600.00"

**Example**

Dim iTestNumber As Integer = 45600

Dim sTestString As String = FormatNumber(iTestNumber, 1)

'sTestString will contain "45,600.0"

## GetRandomNumber()

Get s a random number.

**Parameters**

* Number – the highest possible number.
* Optional Number – the lowest possible number. If omitted, then the number 1 is assumed.

**Returns**

* Number (integer) – a random number

**Example**

Dim iMyRandomNumber as Integer

iMyRandomNumber = GetRandomNumber(100)

Gets a random number between 1 and 100.

**Example**

iMyRandomNumber = GetRandomNumber(10, 5)

Gets a random number between 5 and 10.

## Formatting Functions

## SetForegroundColor()

Sets the text foreground color.

**Parameters**

* Color – a specified color of type Color. See <http://msdn.microsoft.com/en-us/library/aa358802.aspx> for a visual list of all possible colors.

**Returns**

* Nothing

**Example**

SetForegroundColor(Color.Yellow)

Sets the foreground text color to yellow.

## SetBackgroundColor()

Sets the text background color.

**Parameters**

* Color – a specified color of type Color. See <http://msdn.microsoft.com/en-us/library/aa358802.aspx> for a visual list of all possible colors.

**Returns**

* Nothing

**Example**

SetBackgroundColor(Color.Red)

Sets the background text color to red.

## ResetForegroundColor()

Resets the text foreground color to its original setting.

**Parameters**

* None

**Returns**

* Nothing

**Example**

ResetForegroundColor()

## ResetBackgroundColor()

Resets the text background color to its original setting.

**Parameters**

* None

**Returns**

* Nothing

**Example**

ResetBackgroundColor()

## ResetColors()

Resets both the text foreground and background colors to their original settings.

**Parameters**

* None

**Returns**

* Nothing

**Example**

ResetColors()

## SetFontFamily()

Sets the font family for the application's text.

**Parameters**

* String – the name of the font family.

**Returns**

* Nothing

**Example**

SetFontFamily("Arial")

Sets the font family to Arial.

## SetFontSize()

Sets the font size (in points) for the application's text.

**Parameters**

* Number – the font size in points.

**Returns**

* Nothing

**Example**

SetFontSize(12)

Sets the font size to 12 points.

## ResetFontSize()

Resets the font size to its original setting.

**Parameters**

* None

**Returns**

* Nothing

**Example**

ResetFontSize()

## SetFontBold()

Sets the font to bold for the application's text.

**Parameters**

* None

**Returns**

* Nothing

**Example**

SetFontBold()

## SetFontItalic()

Sets the font to italic for the application's text.

**Parameters**

* None

**Returns**

* Nothing

**Example**

SetFontItalic()

## SetFontBoldItalic()

Sets the font to bold and italic for the application's text.

**Parameters**

* None

**Returns**

* Nothing

**Example**

SetFontBoldItalic()

## SetFontNormal()

Sets the font to normal (no bold or italic) for the application's text.

**Parameters**

* None

**Returns**

* Nothing

**Example**

SetFontNormal()