VB.NET Reference Sheet

Basics

Namespace
Imports Namespace

Data Types
Boolean,Byte,Char,DateTime,Decimal,Double,Int16,Int32,Int64,Integer, Long, Object, Short,Single,String,IntPtr,UInteger,UintPtr

Variable Declaration
Dim | Public | Private <variable_name> As <type>

Type Declaration
Dim | Public <variable><suffix>

Suffixes
% -Integer. $ - String, @ - Decimal, & - Long, # - Double, ! - Single

Arrays
Dim <arrayname>(<MaxIndexVal>) As <type>
Dim <arrayname>(<LowerBound> To <UpperBound>) As <type>

Initialize Array
Dim <arrayname>() As <type> = {<value1>, <value2>, ... , <valueN>}

Change Size of Array
ReDim <arrayname>(<MaxIndexVal>)

Comments
'Comment text
'No multi-line comments at this time

XML Comments
Press the ' (apostrophe) key thrice.

Line Continuation
strtext = "To break a long string across multiple lines, " & _
"end the string, add the line continuation character " & _
"and continue the string on the next line."

Arithmetic Operators
*+ (Addition), - (Subtraction), * (Multiplication), / (Division), % (Modulus)*

String Concatenation
+, &

Relational Operators
< (Less Than), <= (Less Than or Equal To),> (Greater Than),>= (Greater Than or Equal To),= (Equal To),<> (Not Equal To)

Logical Operators
OR, NOT, AND, AndAlso, OrElse, Xor

String Manipulation
.Substring(<start>,[<length>])
.Trim() <trims from beginning & end of string>
.TrimEnd([<char array>])
.TrimStart([char array])
.ToLower() <to lower case>
.ToUpper() <to upper case>
.Replace(<find>,<replace>)
.Equals(<expression>) <6 available overloads>
.Contains(<string>)
.Join(<seperator>,<value>,[<count>])
.Compare(<string1>,<string2>,[<ignore case>]) <7 overloads available>
.Copy(<string>)

If Else
```
If(<expression>) Then
    <statement 1>
Else
    <statement 2>
End If
```

Inline If
variable = IIf(condition, value_if_false, value_if_true)

For Loop
```
For <initialize> (Relational Operator) <condition>
    <statement>
Next
```

For Each Loop
```
For Each <variable> In <object>
[Exit For]
    <statements>
[Continue For]
    <statements>
Next
```

While Loop
```
While <expression>
    <statement>
End While
```

Do-While Loop
```
Do
    <statement>
Loop While <expression>
```

Select Case Statement
```
Select Case <expression>
Case <expression1>:
    <statement sequence 1>
Case <expression2>
    <statement sequence 2>
Case <expressionN>
    <statement sequence N-1>
Case Else
    <statement sequence N>
End Select
```

Function Structure
```
<Private, Public> <Function_Name>([Parameters])
    body of the function
End Function
```

Class Structure
```
Public Class <Class_Name>
    body of class
End Class
```

Public
```
'method_prototypes
'data_attributes
Private
    'method_prototypes
    'data_attributes
Friend
    'method_prototypes
Shared
    'method_prototypes
    'data_attributes
```

Error Handling
```
Try
    <statements that may cause an error>
Catch
    <statements to use when an error occurs>
Finally
    <statements to use no matter what happens>
End Try
```


Edited By: born2c0de, Louisda16th, PsychoCoder, TophCoder
Published: August 18, 2007