onecompiler.com /cheatsheets/vb

Visual Basic.net(VB.net) Cheatsheet

Basics

Sample Program

```
Public Module Program
        Public Sub Main(args() As string)
                Console.WriteLine("Hello, World!")
        End Sub
End Module
```

- **Public Module** : Every program contains a module which has the data and procedures that your program uses. Here we are declaring module named `Program` with public visibility.
- **Main** : Beginning of your program
- **Console.WriteLine** : Console is a class of the System namespace and WriteLine() is a method in it which is used to print text to the console.
- VB.net is not a Case-sensitive language
- `'` : Single line Comment

Data types

Data type	Description	Range	Memory Size
byte	used to store unsigned integer	0 to 255	1 byte
sbyte	used to store signed integer	-128 to 127	1 byte
ushort	used to store unsigned integers	0 to 65,535	2 bytes
integer	used to store signed integers	-2,147,483,648 to 2,147,483,647	4 bytes
long	used to store signed integers	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	8 bytes
double	used to store fractional numbers	15 decimal digits	8 bytes
char	used to store a single character enclosed in single quote	one character	2 bytes

Data type	Description	Range	Memory Size
boolean	Boolean data type	Stores either true or false	1 bit
date	Used to store date values	0:00:00 January 1, 0001 to 23:59:59 December 31, 9999	8 bytes
decimal	used to store decimal values	0 to +/-7.9228162514264337593543950335 with 28 places to the right of the decimal	16 bytes
String	Stores a sequence of characters enclosed in double quotes	Sequence of Characters	2 bytes per character
object	Object can be represented as base type of all other types.		4 bytes on 32-bit platform and 8 bytes on 64-bit platform

Variables

```
Dim VariableName As Datatype 'variable Declaration
VariableName = value 'Variable Initialization
```

Example

```
Dim byteVar As Byte
   Dim intVar As Integer = 100
   Dim doubleVar As Double
   Dim dateVar As Date
   Dim charVar As Char = "A"
   Dim strVar As String = "OneCompiler"
   Dim boolVar As Boolean = TRUE
```

Constants

```
[ < attributeList > ] [ accessModifier ] [ Shadows ] Const constantName [
As datatype ] = value
```

- ***attributeList*** − attributeList is optional where you can specify the list of attributes applied to the constants.
- ***accessModifier*** − accessModifier is optional where you specify the accessibility of the constants like Public, Protected, Private, Friend or Protected Friend.
- ***Shadows*** − shadows is also optional which makes the constant hide a programming element of identical name.

- ***constantName*** − constantName specifies the name of the constant
- ***datatype*** − datatype specifies the data type of the constant
- ***value*** − value specifies the value assigned to the constant

```
Const TOTAL As Integer = 100
Public Const NAME As String = "One Compiler"
```

Operators

Operator type	Description
Arithmetic Operator	+ , - , * , / , , MOD, ^
Relational Operator	< , > , <= , >= , =
BitWise Operator	AND, OR, XOR, NOT, AndAlso, OrElse, isTrue, isFalse
BitWise Shift Operator	AND, OR, XOR, NOT, << , >>
Logical Operator	&& , ||, !
Assignment Operator	= , += , -= , *= , /=, = , %= ,<<=,>>=, &=, ^=
Miscellaneous Operator	AddressOf, Await, GetType

Strings

```
Dim variableName As String
```

```
Dim Name As String = "OneCompiler"
```

Conditional Statements

1. If

```
If condition-expression Then
    'code
End If
```

2. If-else

```
If(conditional-expression)Then
   'code if the conditional-expression is true
Else
  'code if the conditional-expression is false
End If
```

3. If-else-if ladder

```
If(conditional-expression)Then
   'code if the above conditional-expression is true
Else If(conditional-expression) Then
        'code if the above conditional-expression is true
    Else
        'code if the above conditional-expression is false
End If
```

4. Nested-If

```
If(conditional-expression)Then
   'code if the above conditional-expression is true
   If(conditional-expression)Then
         'code if the above conditional-expression is true
   End If
End If
```

5. Select Case

```
Select [ Case ] expression
   [ Case expressionlist
      'code ]
   [ Case Else
      'code ]
End Select
```

Loops

1. For..Next

```
For counter [ As datatype ] = begin To end [ Step step ]
   'code
   [ Continue For ]
   'code
   [ Exit For ]
   'code
Next [ counter ]
```

2. For..Each

```
For Each element [ As datatype ] In group
    'code
    [ Continue For ]
    'code
    [ Exit For ]
    'code
Next [ element ]
```

3. While

```
While conditional-expression
    'Code
    [ Continue While ]
    'Code
    [ Exit While ]
    'Code
End While
```

4. Do-while

```
Do { While | Until } conditional-expression
    'Code
    [ Continue Do ]
    'Code
    [ Exit Do ]
    'Code
Loop
```

```
Do
    'Code
    [ Continue Do ]
    'Code
    [ Exit Do ]
    'Code
Loop { While | Until } conditional-expression
```

Functions

Functions consists of a set of statements which perform a sepcific functionality and they return a value when they are called.

```
[accessModifiers] Function functionName [(parameterList)] As returnType
   'code
End Function
```

Sub-Procedures

Sub-procedures are similar to functions but they don't return any value.

```
Sub ProcedureName (parameterList)
'Code
End Sub
```