Visual Basic .NET Cheat Sheet

A concise reference for Visual Basic .NET (VB.NET) syntax, concepts, and common tasks.

Language Basics

Data Types

`Boolean`	True or False value.
`Byte`	8-bit unsigned integer (0 to 255).
`Short`	16-bit signed integer (-32,768 to 32,767).
`Integer`	32-bit signed integer (-2,147,483,648 to 2,147,483,647).
`Long`	64-bit signed integer (-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807).
`Single`	32-bit floating-point number.
`Double`	64-bit floating-point number.
`Decimal`	128-bit data type suitable for financial and monetary calculations.
`String`	Sequence of Unicode characters.
`Date`	Represents a date and time value.

Variable Declaration

`Dim variableName As DataType`

Example:

```
Dim age As Integer
Dim name As String = "John Doe"
```

`Const constantName As DataType = value`

Example:

```
Const PI As Double = 3.14159
```

Operators

Arithmetic	`+`, `-`, `*`, `/`, `\` (integer division), `Mod` (modulus), `^` (exponentiation)
Comparison	`=`, `<>`, `<`, `>`, `<=`, `>=`
Logical	`And`, `Or`, `Not`, `Xor`, `AndAlso` (short-circuiting And), `OrElse` (short-circuiting Or)
Concatenation	`&` (string concatenation)
Assignment	`=`, `+=`, `-=`, `*=`, `/=`, `&=`

Control Flow

Conditional Statements

`If...Then...Else` Statement

```
If condition Then
    ' Code to execute if condition is true
ElseIf condition2 Then
    ' Code to execute if condition2 is true
Else
    ' Code to execute if all conditions are
false
End If
```

`Select Case` Statement

```
Select Case variable
    Case value1
        ' Code to execute if variable = value1
    Case value2
        ' Code to execute if variable = value2
    Case Else
        ' Code to execute if variable doesn't
match any case
End Select
```

Looping Structures

`For...Next` Loop

```
For i As Integer = start To end [Step
stepValue]
    ' Code to execute
Next [i]
```

`While...End While` Loop

```
While condition
    ' Code to execute while condition is true
End While
```

`Do...Loop` Loop

```
Do [{While | Until} condition]
    ' Code to execute
Loop

Do
    ' Code to execute
Loop [{While | Until} condition]
```

`For Each...Next` Loop

```
For Each element As DataType In collection
    ' Code to execute for each element
Next
```

Exception Handling

`Try...Catch...Finally` Block

```
Try
    ' Code that might throw an exception
Catch ex As ExceptionType
    ' Code to handle the exception
Finally
    ' Code that always executes, regardless of
exceptions
End Try
```

`Throw` Statement

```
Throw New Exception("An error occurred.")
```

Procedures and Functions

Sub Procedures (Void)

`Sub ProcedureName([parameterList]) ' Code to execute End Sub`

Example:

```
Sub Greet(name As String)
    Console.WriteLine("Hello, " & name & "!")
End Sub
```

Function Procedures (Return Value)

`Function FunctionName([parameterList]) As ReturnType ' Code to execute Return returnValue End Function`

Example:

```
Function Add(x As Integer, y As Integer) As
Integer
    Return x + y
End Function
```

Parameters

`ByVal`	Passes a variable by value (a copy is passed).
`ByRef`	Passes a variable by reference (the original variable is passed, allowing modification).
`Optional`	Specifies that a parameter is optional. Must have a default value. `Optional param As DataType = defaultValue`
`ParamArray`	Allows a procedure to accept an arbitrary number of arguments of a specified type. `ParamArray paramName() As DataType`

Object-Oriented Programming

Classes

`Class ClassName ' Class members (fields, properties, methods) End Class`

Example:

```
Class Dog
    Public Name As String
    Public Sub Bark()
        Console.WriteLine("Woof!")
    End Sub
End Class
```

Properties

`Property PropertyName As DataType Get ' Code to return the property value End Get Set(value As DataType) ' Code to set the property value End Set End Property`

Example:

```
Private _age As Integer
Public Property Age As Integer
    Get
        Return _age
    End Get
    Set(value As Integer)
        If value >= 0 Then
            _age = value
        End If
    End Set
End Property
```

Inheritance

`Class DerivedClass Inherits BaseClass ' Derived class members End Class`

Example:

```
Class Animal
    Public Name As String
End Class

Class Dog
    Inherits Animal
    Public Sub Bark()
        Console.WriteLine("Woof!")
    End Sub
End Class
```

Interfaces

`Interface InterfaceName ' Interface members (method signatures, property signatures) End Interface`

`Implements InterfaceName`

Example:

```
Interface IAnimal
    Sub MakeSound()
End Interface

Class Dog
    Implements IAnimal
    Public Sub MakeSound()
        Console.WriteLine("Woof!")
    End Sub
End Class
```