VB.Net Basic

**Variables**

Variables are used to store data. A variable has a name to which refers data type. In vb.net, variables are declared with ‘Dim’ keywords.

Syntax

Dim *VariableName* as variableType

Example:

Declaration of variables:

Dim x as Integer

Dim Name as String

**Data Types:-**

Following are data types available in Vb.Net

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Size(Bytes)** | **Description** |
| Integer | 4 | 32-bit Signed Integer |
| Double | 8 | 64-bit floating point Variables |
| Long | 8 | 64-bit signed integer |
| Short | 2 | 16-bit signed integer |
| String | Varies | 2 bytes each character |
| Etc. |  |  |

**Constants:**

Constant variable is used for using only one value for entire program. It is declared in one location of program. Then refer to the constant by name throughout the code instead of hard writing numeric values in the code. When we have to change value of constant variable, we can just change in one well-defined part of the code.

Syntax:

Const *Constantvariable*

Example

Module Module1

Sub main()

Const pi=3.14

Dim Radius, Area As Single

Radius=2

Area= pi\*Radius\*Radius

Console.writeLine(“ Area=”& Area)

End Sub

End Module

**# Forcing Variable Declaration in Vb.Net**

In any solution of project, there is option of Explicit in a compile property. So option Explicit statement ensures whether the compiler requires all variables to be explicitly declared or not before it uses in the program.

The option Explicit has two modes, ON and Off Mode. If option Explicit mode is ON, you have to declare all the variable before you use it in the program. If not, it will generate a compile-time error, whenever a variable that has not been declared encountered. If the option Explicit Mode is OFF, Vb.net automatically create variables whenever it sees a variable without declaration.

**# String**

String is a collection of character. Also called array of characters. In vb.net string variable is created by using string keyword as datatype. This keyword belongs to System.String

Syntax

Dim VariableName as *String*

Example

Dim FirstName, LastName, Address As String

String object can be created by following methods

1. By assigning a string literal to a String variable
2. By using a the string concatenation operator(+)
3. By retrieving a property or calling a method that returns a string

**Methods of String**

1. Concat

Syntax: *String.Concat(string1, string2)*

It concatenates two or more than two string variable in one variable.

1. Contains

This method determines if a substring exists in a source string. It returns Boolean that indicates whether the argument string is located in the instance string.

Syntax: *Sorucestring.Contains(‘substring’)*

1. Length

It returns total number of character present in string.

Syntax: *inputString.Length*

1. ToLower

This method converts upper case character to lower case

Syntax:*InputString.ToLower()*

1. Mid

Syntax: *Mid(InputString,5,3)*

It returns a substring.

1. Clone

It creates copy of any earlier string variable value.

Syntax: *Earliervariable.Clone()*

Example:

Dim Name As String

Dim CloneName As String

Name=”Niraj”

CloneName=Name.Clone()

**# Scope and Life Time of Variable**

* The scope of a variable is the section of application that you can see and manipulate the variable

Example : if a variable is declared with in procedure , only the code in the specific procedure has access to that variable.

* Life Time of variable :it is the period for which they retain their value. Variables declared as Public exist for the lifetime of the application. Local Variables , within the procedures or block of code with the Dim or private statement , live as the long as the procedure.
* In Vb.net , where you declare an element determines its scope. And an variables can have scope at one of the following levels:

1. Block Level: Available only within the code block in which it is declared.

When any variable is declared within the block code, then after it can not be used outside of block.

Eg.

If age < 10 Then

Dim Status As String=”Child”

End If

1. Procedure Scope/Local Scope

-Available only within the procedure in which it is declared.

Example:

Sub Info()

Dim Name As String

Name=”Prakash”

Console.WriteLine(Name)

End Sub

Sub Student()

End Sub

In this program Variable Name is only accessible within Info() Procedure. It is not accessible to Procedure Student()

1. Module Scope

Available to all code within the module, class in which it is declared.

Example:

Module VbOnline

Dim Name As String=”Prakash”

Sub Info()

Name=”Prakash”

Console.WrileLine(Name)

End Sub

Sub Student()

Name=”Niraj”

Console.WriteLine(Name)

End Sub  
 Sub Main()

Info()

Student()

Console.ReadKey()

End Sub()

In this program Variable *Name* is accessible to all procedure of *VbOnline* Module.

1. Name Space Scope

Available to all code in the name space.

Inside these levels of scope, you can also specify the scope of an element when you declare it, which are:

1. Public
2. Private
3. Protected

**Data Type Conversion**

The process of changing a value from one data type to another type is called conversion. Conversion may be implicit or explicit.

* An implicit conversion does not require any special syntax in the source code.

Example:

Dim number1 as Integer

Dim number2 as Double

number1=32

number2=number1

* An explicit conversion uses a type conversion keyword. Visual Basic provides several such keywords, which forces to convert data to the desired output. These keywords act like functions, but the compiler generates the code inline, so execution is slightly faster than with a function call.

Example:

Dim i As integer

Dim j As integer

j=Math.sqrt(i)

Some conversion functions are:

1. Cchar(expression)

Converts the expression to char data type

Example

Dim Name as string=”Akash”

Console.WriteLine(“String converted to character data type as : ”&Cchar(Name))

Output: A

1. CInt(Expression)

Converts the expression to integer data type.

Example:

Dim i As Integer

I=CInt(8.78)

Result-> 9

1. CLng(expression)

Converts the expression to Long Data Type

1. CStr(Expression)

Converts the expression to String Data Type

**Operator**

**Do yourself**

**Array**

An array is a sequential collection of same data type of data. It consists of contiguous memory location. The lowest address corresponds to the first element and highest address to the last element.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N(0) | N(1) | N(2) |  |  | N(5) |

Where N(0) is first element and N(5) is last element of array.

* To declare an array in vb.net Dim statement is used.

*Syntax: Dim VariableName(20) As DataType*

*Example: Dim RollNo(20) As Integer*

The data array now has 20 elements, starting from 0 to 19.

**Type of Array**

1. Standard Array

You usually use the Dim Statement to declare a standard array. Here are few example of standard array

*Dim Name(20) As String*

*Dim Number(25) As Integer*

For populating elements array

*For i=0 To 19*

*Number(i)=i+1*

*Next i*

* *Displaying Array Value*

*For i=0 To 19*

*Console.WriteLine(Number(i)*

*Next i*

* You can also initialize the data in an array if you don’t give an array explicit size. Here is the syntax to use

*Dim Varaible\_Name() As DataType={Values}*

Example

*Dim RollNo() As Integer={1,2,3,}*

*Dim Name() as Stirng={“Amit”,”Astha”,”Suraj”}*

* Two dimensional Array

*Syntax : Dim TwoDimensional\_Array(i,j) as integer*

*Example:*

*Dim Number(4,5) As Integer*

1. Dynamic Array

You can use Dim Statement to declare an array with empty parenthesis to declare a dynamic array. Dynamic arrays can be re-dimensioned as you need them with *redim* Statement.

Example:

Dim DynamicArray() As String

ReDim DynamicArray(10)

DynamicArray(0)=”Pokhara”

Redim DynamicArray(20)

DynamicArray(18)=”Lamachaur”

**Enumerations**

Enumerations, which is related constants, manages number of constants.

Syntax: *<attributes> Access modifier*

*Enum EnumerationName[As Datatype]*

*Memberlist*

*End Enum*

An enumerated type is declared using the Enum statement. The Enum statement declares an enumeration and defines values of its members. The Enum Statement can be used at the module, class, structure, procedure etc.

Example:

Module ConstatnEnum

Enum Colors

Red=1

Orange=2

Yellow=3

Green=4

Azure=5

Blue=6

Violet=7

End Enum

Sub Main()

Console.WriteLine(“The color red id is :”&Colors.Red)

End Sub

End Module