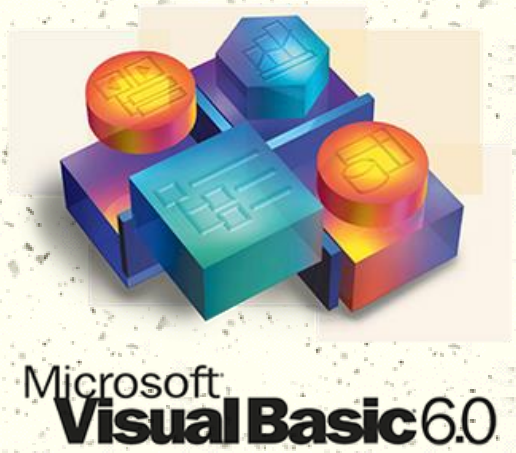
MM PG College Fatehabad

Introduction to VB.NET



Class B.Sc. (CS)

2nd Year/4th Sem.

Agenda

- Why VB.NET
- What is new in VB.NET
- Update to VB.NET?
- VB.NET Language Essential

Why VB.NET (from technical standpoint)

- The world of applications is changing:
  - The move to Web
  - The need for reusability, centralization and scalability
  - MTS, COM+, and Component Services cannot be fully taken advantage of by VB.
  - SOAP: features can be implemented more completely with .NET.

Why VB.NET (cont.)

- To get the benefit of .NET framework and its core execution engine: CLR.
  - Garbage collection
  - OO mechanism
  - Standard security services
  - Integrated debugging tools

Why VB.NET (cont.)

- Why not *C#*
  - VB.NET----"The most productive tool for building .NET-connected applications. "----*Microsoft Corporation*
  - *Root in Basic, the most pure-flavor language product from MS.*
  - *Easier for VB programmers: a number of unique features.*
    - *E.g.: Only VB.NET has background compilation, dropdown list of the code window.*

What is New in VB.NET ----For Experienced VB Programmers

- IDE changes
- Project Changes
- Web Changes
- WebClass Changes
- Data Changes
- Component Authoring Changes
- UserControl Changes
- Forms Changes
- Debugging Changes
- Setup and Deployment Changes
- International Changes
- Windows API Changes
- Registry Access Changes
- Constant Changes
- Namespace Changes
- Run-Time Changes

Overview of Big Changes in VB.Net

- Everything is object-oriented: abstraction, inheritance, overloading, encapsulation and polymorphism.(Note: no multiple inheritance, but interfaces supported.)
- Multithreaded applications are possible.
- Language syntax changes

……

Changes in VB Language

- All data are objects, based on the class: ***System.Object***.
  - E.g. class supports Windows forms: ***System.Windows.Forms.Form***.
- The built-in VB functionality is encapsulated in a namespace called ***System***.
  - E.g ***Collection*** has be replaced by ***System.Collections***.
- Old control are gone, and new ones have appeared.

Changes in VB Language (cont.)

- Many keywords are renamed or gone, while some new added.
  - E.g. ***Gosub*** removed
- Strict data typing is now enforced
  - Variable must be declared before used by default.
  - Cannot assign one data type to another, but can use ***Ctype*** to convert between types.
  - The same as in VC++ and C#.
- Structured exception handling: ***Try…Catch…Finally***.

Changes in VB Language (cont.)

- When calling procedures, must use parentheses.
- Parameters are by default passed by value, instead of by reference.
- Supports constructors and destructors for use when initializing an object of a class.
- ***If…Then*** statements are now short-circuited.

Changes in VB Language (cont.)

- A number of new compound operators
  - E.g. x+=2
- The ***And***, ***Or***, **Not** and ***Xor*** operators have changed from bitwise to boolean operators. Meanwhile, the bitwise versions are ***BitAnd***, ***BitOr***, **BitNot**, and ***BitXor***.
- No default property supported
  - E.g. VB6: TextBox1="Hello"

    VB.Net: TextBox1.Text="Hello"

Changes in VB Language (cont.)

- Three new data types
  - ***Char***: unsigned 16-bit
  - ***Short***: signed 16-bit
  - ***Decimal***: signed 96-bit (replaces ***Variant***)

Integer Type	VB 6.0	VB.NET
8 bit	Byte	Byte
16 bit	Integer	Short
32 bit	Long	Integer
64 bit	Not Applicable	Long

Changes in Data Handling

- A new data-handling model: ADO.NET.
  - Facilitates Web application.
  - Uses XML to exchange data.
- COM/DCOM technologies have been replaced by .NET framework.
- Datasets (not record sets now) are based on XML schema, so they are strongly typed.
- Many new tools are provided to handle data.
- But can still work with ADO using ***COM interoperability*** in the .NET framework.

Changes in Web Development

- Two major types of Web application:
  - Web forms: web-based applications with GUI.
    - Based on ASP.NET
    - Can use standard HTML control, or new Server control handled by the Web server.
    - Controls can be bound on a Web form by setting the codes in the properties.
  - Web services: to process data using HTTP and XML files on the Internet.

Update to VB.NET ?

- "Visual Basic .NET represents a major departure form previous versions of Visual Basic in several ways."

  ----*Microsoft Corporation*
- Plenty changes in VB.NET will take lots of effort of even the experienced VB developers.
- Old but running fine systems, fund, experienced developers…

Update to VB.NET ? (cont.)

- Consideration
  - Unsupported features
    - OLE Container Control
    - Dynamic Data Exchange
    - DAO or RDO Data Binding
    - VB5 Controls
    - DHTML Applications
    - ActiveX Documents
    - Property Pages

Update to VB.NET ? (cont.)

- Carefully reworked
  - Single-tier Database Applications
  - VB Add-ins
  - Games
  - Graphics
  - Drag and Drop Functionality
  - Variants
  - Windows APIs

Update to VB.NET ? (cont.)

- Visual Basic Upgrade Wizard
  - Automatically invoked when open a VB6 project.
  - Results are not satisfactory due to the big different.
- Recoding by hand.

VB.NET Language Essential ---- For Non-VB Programmers

- Projects Types
  - Three most commonly used:
    - Windows Forms
    - Web Forms
    - Console Applications
- Statements

Statement: If…Else

```
Module Module1
  Sub Main()
      Dim intInput As Integer
      System.Console.WriteLine("Enter an interger…")
      intInput=Val(System.Console.ReadLine())
      If intInput=1 Then
         System.Console.WriteLine("Thank you!")
      ElseIf intInput=2 Then
         System.Console.WriteLine("That's good!")
      Else
          System.Console.WriteLine("Not a right number!")
      End If
  End Sub
End Module
```

Statement: Select Case

```
Module Module1
  Sub Main()
      Dim intInput As Integer
      System.Console.WriteLine("Enter an interger…")
      intInput=Val(System.Console.ReadLine())
      Select Case intInput
           Case 1
                System.Console.WriteLine("Thank you!")
           Case 2
                System.Console.WriteLine("That's good!")
           Case 3 To 7
                System.Console.WriteLine("OK")
           Case Is> 7
               System.Console.WriteLine("Too Big")
           Case Else
               System.Console.WriteLine("Not a right number!")
       End Select
  End Sub
End Module
```

Functions: Switch and Choose

- Switch Function
  - Syntax
    - Switch(expr1, value1[, expr2, value2…[,exprn, valuen]])
  - E.g.
    - intAbsValue=Switch(intValue<0, -1 * intValue, intValue>=0, intValue)
- Choose Function
  - Syntax
    - Choose(index, choice1[, choice2,…[,choicen]])
    - Note: unlike array index, choose index from 1 to n
  - E.g.
    - Str=Choose(intValue, “Thank you!”, “That is good!”)

Loop Statement: Do

- Syntax:

```
Do [While|Until] condition]
    [statements]
    [Exit Do]
    [statements]
Loop
```

- E.g.

```
Module Module1
  Sub Main()
     Dim strInput As String
     Do Until Ucase(strInput)="Stop"
        System.Console.WriteLine("What should I do?")
        strInput=System.Console.ReadLine()
     Loop
  End Sub
 End Module
```

Loop Statement: For

- Syntax:

```
For index=start To end [Step step]
     [statements]
     [Exit For]
     [statements]
 Next [index]
```

- E.g.

```
Module Module1
  Sub Main()
      Dim loopIndex As Integer
      For loopIndex=0 to 3
          System.Console.WriteLine("Hello!")
       Next loopIndex
   End Sub
End Module
```

Loop Statement: While

- Syntax:

```
While condition
    [statements]
End While
```

- E.g.

```
Sub CheckWhile()
    Dim intCounter As Integer =0
    Dim intNumber As Integer =10
    While intNumer>6
        intNumber-=1
        intCounter+=1
    End While
    MsgBox("The loop ran " & intCounter & " times.")
End Sub
```

Loop Statement: For Each…Next

- Syntax:

```
For Each element In group
     [statements]
     [Exit For]
     [statements]
 Next element
```

- E.g.

```
Sub Main()
      Dim intArray(2), intItem As Integer
      intArray(0)=0
      intArray(1)=1
      intArray(2)=2
      For Each intItem In intArray
         System.Console.WriteLine(intArray)
      Next intItem
 End Sub
```

Like a Loop: With

- Syntax:

```
With object
      [statements]
 End With
```

- E.g.

```
With TextBox1
      ,Height = 1000
      .Width = 3000
      .Text = "Welcome, World!"
End With
```

Like With: Enumerations

- E.g.

```
Module Module
    Enum Days
        Sunday=1
        Monday=2
        Tuesday=3
        Wednesday=4
    End Enum

    Sub Main()
        System.Console.WriteLine("Monday is day " & Days.Monday)
    End Sub
End Module
```

Option Statement

- **Option Explicit**: On/Off.
  - "On":requires declaration of all variables before used.
- **Option Compare**: Binary/Text.
  - Specifies strings are compared using binary or text comparison operations.
- **Option Strict**: On/Off.
  - "On":used when assigning a value of one type to a variable of another type, indicates any possibility of data loss.

Example for Option Strict

```
Option Strict On
Module Module1
   Sub Main()
         Dim dbData As Double
         Dim intData As Integer
         dbData=3.14159
         intData=Cint(dbData)  'Not   intData=dbData
         System.Console.WriteLine("intData:"&_
         Str(intData))
   End Sub
End Module
```

Imports Statement

- To import a namespace .
- E.g.

```
Option Strict Off
Imports System.Console
Module Module1
   Sub Main()
      WriteLine("Hello!")
   End Sub
End Module
```

Thank you!