

Chapter 2 VB Programming

- ✓ Basic Programming (VB.Net 2008)
 - ✓ Variables, Constants, and Data Type
 - ✓ Common control (TextBox, ComboBox, CheckBox, MaskedTextBox, ListView, DataGridView,..)
 - ✓ Subroutine and Function
 - ✓ Program Control Statements
 - ✓ Error Handling
 - ✓ Array
 - ✓ Creating Class Library (.dll File)
- ✓ Database Programming (VB.Net 2008)
 - ✓ Using OLEDB to connect MS-Access 2003, Microsoft SQL Server 2000 Database
 - ✓ Using DataAdapter, DataSet and DataTable to Retrieve Data from Database
 - ✓ Using OleDbCommand, SqlCommand to access data in Database
 - ✓ Using DataGridView Control to View Data from Database
 - ✓ Using ComboBox with Data from Database
- ✓ Bonus
 - ✓ Export Data from DataGridView to MS-Excel
 - ✓ Save Image to Database in MS-Access Using OleDb
- ✓ Assignments

Variables, Constants, and Data Type

- Common Data Type in VB.Net 2008

Data Type	Value Range
Boolean	True or False
Byte	0 to 255 (unsigned)
Sbyte	-128 to 127
Integer	-2,147,483,648 through 2,147,483,647 (signed)
Long	-9,223,372,036,854,775,808 through 9,223,372,036,854,775,807 (9.2...E+18 [†]) (signed)
String	0 to approx. 2 billion unicode characters.
Date	0:00:00 (midnight) on January 1, 0001 through 11:59:59 PM on December 31, 9999
Single	-3.4028235E+38 through -1.401298E-45 [†] for negative values; 1.401298E-45 through 3.4028235E+38 [†] for positive values
...

Variables, Constants, and Data Type

- Declare Variables
 - Global
 - Can use in a whole project
 - Declare at the top of module as **Public** or **Friend**

```
Public var_name As Data Type [= Initial value]
Friend var_name As Data Type [= Initial value]
```
 - Local
 - Declare in Module or Form (Declare at the top)


```
Private var_name As Data Type [= Initial value]
```
 - Declare in Sub Procedure or Function


```
Dim var_name As Data Type [= Initial value]
```
 - Constant: variable that must has initial value and cannot change that value.


```
Private Const var_name as Data Type = value
```

Variables, Constants, and Data Type

```
Module myModule
    Public pubAppUser As String = "Admin"
    Public pubDBUser As String = "sa"
    Public pubServer As String = "."
    Public Const pubEnglishKeyboard As String= _
        "United Stated-International"
End Module

Private Function GenerateID() As Integer
    Dim myID As Integer = 0
    .. ..
    Return myID
End Sub
```

Variables, Constants, and Data Type

- Assign value to Variable
 - Initial value
`Dim x As Integer = 20`
 - Assign value
`Dim a As Integer`
`Dim b As Integer`
`Dim c As Double`
`a = 245`
`b = 3`
`c = a/b`

Sub and Function Procedure

- Sub Procedure
 - Series of visual basic statement enclosed by **Sub** and **End Sub** statement.
 - The Sub procedure performs a task and then returns control to the calling code, but it does not return a value to the calling code.
 - Syntax
`[Private|Public] Sub sub_name([parameters])`
`'...here is visual basic statements`
`End Sub`

Sub and Function Procedure

- Sub Procedure Example

```

Public Sub Sum2Num(ByVal x As Integer, _
                    ByVal y As Integer )
    Dim result As Integer
    result = x + y
    MsgBox "Result is: " & result
End Sub

Public Sub GetSchoolToComboBox(ByVal cbo_name _
                               As ComboBox)
    Dim dt as New DataTable
    . . . . .
End Sub

```

Sub and Function Procedure

- Function Procedure

- A Function procedure is a series of Visual Basic statements enclosed by the Function and End Function statements.
- The Function procedure performs a task and then returns control to the calling code, and also return a value.
- Syntax

```

[Private|Public] Function fun_n ([parameters]) _
                               As Data Type
    Dim re_value As Data Type
    '....here is visual basic statements
    Return re_value
End Function

```

Sub and Function Procedure

- Function Procedure Example

```
Public Function Sum2Num(ByVal x As Integer, _
                        ByVal y As Integer) As Integer
    Dim result As Integer
    result = x + y
    Return result
End Sub

Public Sub CallSum()
    Dim mySum As Integer
    mySum = Sum2Num(4,120)
    MsgBox "Result is: " & mySum
End Sub
```

Program Control Statements

- If Else If Else End If

```
If Condition Then
    .. .. .
End If

Or

If Condition Then
    .. .. .
ElseIf Condition Then
    .. .. .
ElseIf Condition Then
    .. .. .
Else
    .. .. .
End If
```

Program Control Statements

- Example
 - Students average score is a criteria to determine mention.
 - If average score below 50 then mention F.
 - if average score less than 65 and greater than or equal to 50 then mention E.
 - if average score less than 80 and greater than or equal to 65 then mention D.
 - If average score less than 90 and greater than or equal to 80 then mention C.
 - If average score less than 95 and greater than or equal to 90 then mention B.
 - If average score less than or equal to 100 and greater than or equal to 95 then mention A.

Program Control Statements

```
Friend Function Mention(ByVal score As Double) As String
  If score<50 Then
    Return "F"
  ElseIf score<65 Then
    Return "E"
  ElseIf score<80 Then
    Return "D"
  ElseIf score<90 Then
    Return "C"
  ElseIf score<95 Then
    Return "B"
  ElseIf score<=100 Then
    Return "A"
  Else
    Return ""
  End If
End Function
```

Program Control Statements

- Loops
 - For Next


```
For counter=start_counter to end_counter
    Statement
Next
Or
For each Object in Objects Collection
    Statement
Next
```
 - Do Until


```
Do Until end_loop_condition
    Statement
Loop
Or
Do
    Statement
Loop Until end_loop_condition
```
 - While loop


```
While true_condition
    Statement
End While
```

Program Control Statements

- Calculate n!
 - Using For ... Next


```
Dim result As Long=1
Dim i as Integer
For i=1 to n
    result*=i
Next
```
 - Using Do Loop


```
Dim result As Long=1
Dim i as Integer=0
Do Until i=n
    i+=1
    result*=i
Loop
```

Array

- Declare Array

- One Dimension Array

- Declare array with specific number of array

```
Dim a(10) As Integer
```

```
`There are 11 items we can use (a(0) to a(10))
```

- Declare array with non-specific number of array

```
Dim a() As Integer
```

```
`We cannot use this array yet
```

```
Redim a(20)
```

```
`There are 21 array items we can use
```

```
Redim Preserve a(30)
```

```
`We can add more items without losing the old values  
of array
```

Array

- Multiple Dimension Array Declaration

- Declare array with specific number of array

```
Dim a(10, 10) As Integer
```

```
`There are 11 rows and 11 columns
```

```
Dim b(10,5,3) As Integer
```

```
`There are  $11*6*4 = 264$  array items we can use
```

- Declare array with non-specific number of array

```
Dim a(,) As Integer `Two dimension array
```

```
Dim b(,,) As Integer `Three dimension array
```

```
Redim a(10,10)
```

```
Redim b(5,5)
```

```
Redim Preserve b(5,7)
```

```
`Can preserve only the right most index of array
```

Error Handling

- On Error GoTo

```
On Error GoTo ErrHandler
```

```
... ..
```

```
ErrHandler:
```

```
MsgBox Err.Description
```

- Try.....Catch Exception

```
Try
```

```
... ..
```

```
Catch ex As Exception
```

```
MsgBox ex.Message
```

```
End Try
```

Create Class Library (.Dll File) in VB 2008

- Open Visual Basic .Net 2008
- Create New Project
- Select **Class Library** Project Type
- Enter the name of Project
- Create Function and Sub Routine within any Classes you create
- Press F5 for first debugging
- Build Project Again after Modification
- You can find file **.dll file** in **Project\Debut\Bin**

Common Controls in VB.Net 2008

- **TextBox**: use to entry data
- **ComboBox**: use to entry data or selection
- **CheckBox**: use for choices
- **MaskedTextBox**: use to entry data follow the defining mask such as Date
- **ListView**: use to view data
- **DataGridView**: use to view data and entry data
- **Button**: use for a command
- **Form**: use as a container contain other controls
- **Label**: use to show title of data
- **MenuStrip (menu bar)**: use for functions navigator
- **ToolStrip (tool bar)**: use for functions navigator

Common Controls in VB.Net 2008

- **TextBox**

Properties	Methods	Events
Text, TabIndex TabStop, ReadOnly PasswordChar MultiLine MaxLength	Focus Cut, Copy, Paste	TextChanged KeyDown KeyPress KeyUp

- **CheckBox**

Properties	Methods	Events
Checked Text	Focus	CheckedChanged

Common Controls in VB.Net 2008

- **ComboBox**

Properties	Methods	Events
Text, AutoCompleteMode, AutoCompleteSource, DropDownStyle, Items	Focus	SelectedIndexChanged

- **MaskedTextBox**

Properties	Methods	Events
Text, Mask	Focus	TextChanged

Common Controls in VB.Net 2008

- **ListView**

Properties	Methods	Events
Columns, FullRowSelect, HideSelection, Items, MultiSelect	Focus	ItemChecked, ItemSelectionChanged, SelectedIndexChanged

- **DataGridView**

Properties	Methods	Events
Columns, AllowUserToAddRows, AllowUserToDeleteRows, EditMode, MultiSelect, SelectionMode,	Focus, Sort,	CellBeginEdit, CellClick, CellEndEdit, CellEnter, CellValueChanged, CellValidating,

Common Controls in VB.Net 2008

- Button

Properties	Methods	Events
Text, Image, TextImageRelation	Focus, PerformClick	Click, MouseDown, MouseUp

- Form

Properties	Methods	Events
Text, IsMdiContainer, StartPosition, WindowState, ShowIcon, ShowInTaskBar, KeyPreview, FormBorderStyle	Close, Hide, Show, ShowDialog	Load, KeyDown, FormClosed, Shown, Resize,

Common Controls in VB.Net 2008

- MenuStrip

Properties	Methods	Events
Items,	PerformClick, ShowDropDown,	Click, CheckedException,

- ToolStrip

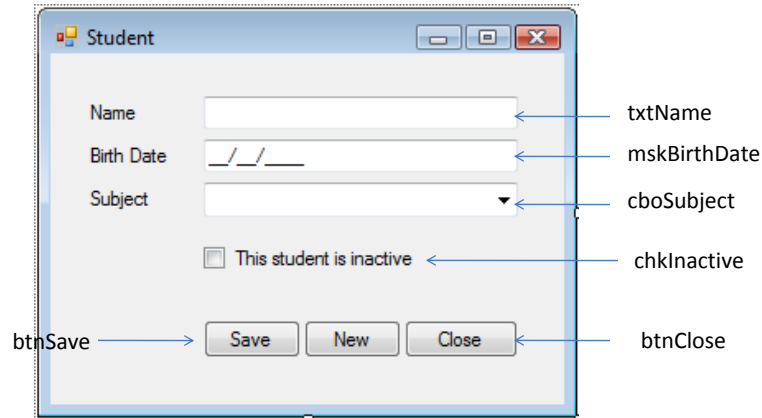
Properties	Methods	Events
Items,	PerformClick,	Click, CheckedException,

- Label

Properties	Methods	Events
Text, TabIndex,	Click,

Common Controls in VB.Net 2008

- Using Common Controls



Common Controls in VB.Net 2008

- Naming Common Controls

Controls	Shortcut Name
TextBox	txt
ComboBox	cbo
CheckBox	chk
Button	btn
ListView	lvw
Form	frm
DataGridView	dgv
MaskedTextBox	msk
MenuStrip	ms
ToolStrip	ts
.....

Problems

- Write VB 2008 Program to find root of:
 $ax^2+bx+c = 0$
- Write VB 2008 Program to find root of:

$$\begin{cases} ax + by = c \\ a'x + b'y = c' \end{cases}$$

- Write VB 2008 Program to find root of:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots \quad \dots \quad \dots \quad \dots \\ \dots \quad \dots \quad \dots \quad \dots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n \end{cases}$$

Problems

- Write VB 2008 Program to Apply Binary Search Method to Search a Word in a Collection of Words.
 - Use DataGridView or ListView to Store a Collections of Words
- Write VB 2008 Program to Convert Number to Word (eg. 12 is “Twelve”, 101 is “One Hundred and One”)
 - Create .dll file contains function Converting Number to Word

Database Connection

- Connect to Microsoft Access 2003 Using OleDb

```
Dim cn As New OleDb.OleDbConnection
cn.ConnectionString = _
    "Provider=Microsoft.Jet.OleDb.4.0; Data Source=" & _
    "File_Name.mdb"
cn.Open()
... ..
cn.Dispose()
```

- Or Connect with Password in MS-Access 2003

```
Dim cn As New OleDb.OleDbConnection
cn.ConnectionString = _
    "Provider=Microsoft.Jet.OleDb.4.0; Data Source=" & _
    "File_Name.mdb; Jet ODBC:Database Password=123"
cn.Open()
... ..
cn.Dispose()
```

Database Connection

- Connect SQL Server 2000 Using OLEDB

```
Dim cn As New OleDb.OleDbConnection
cn.ConnectionString = _
    "Provider=Microsoft.Jet.OleDb.4.0; Data Source=" & _
    "File_Name.mdb"
cn.Open()
... ..
cn.Dispose()
```

- Connect SQL Server 2000 Using SqlClient

```
Dim cn As New SqlClient.SqlConnection
cn.ConnectionString="Data Source=server_name; " & _
    "Database=db_name; User ID=sa; Password=123;" & _
cn.Open()
.....
cn.Dispose()
```

Retrieving Data

- OleDbDataAdapter serves as a bridge between DataSet or DataTable and Data Source (Database Management System – DBMS).
- DataSet is an in-memory cache of data retrieved from a data source. DataSet contains one or more DataTable.
- DataTable is an in-memory cache of data that compose from rows and columns.
- Similarity, using SqlDataAdapter for SQL Server Database
- Note: All classes we are using here is in System.Data Name Space.

Retrieving Data

- Using OleDbDataAdapter and DataTable
 - Below is an example of selecting data from Table Student Using Connection **cn**

```

.....
Dim da As New OleDbDataAdapter("SELECT * FROM" & _
    " Student" , cn)
Dim dt As New DataTable
da.Fill(dt)
For each dr As DataRow in dt.Rows
    MsgBox dr(0) & " " & dr(1)
Next
dt.Dispose(): da.Dispose()
.....

```

Retrieving Data

- Using OleDbDataAdapter and DataSet
 - Below is an example of selecting data from Table Student Using Connection **cn**

```

.....
Dim da As New OleDbDataAdapter("SELECT * FROM" & _
    " Student" , cn)
Dim ds As New DataSet
da.Fill(ds, "std")
For each dr As DataRow in ds.Tables("std").Rows
    MsgBox dr(0) & " " & dr(1)
Next
ds.Dispose(): da.Dispose()
.....

```

Data Manipulation

- Using OleDbCommand or SqlCommand to Execute INSERT, DELETE and UPDATE data in Data Source.
 - Below is an example of inserting one row to table student(id,name) using connection **cn**

```

.....
Dim cmd As New OleDbCommand
cmd.Connection = cn
cmd.CommandText = "INSERT INTO student([id], " &
    "[Name]) VALUES(1, 'Data')"
cmd.ExecuteNonQuery()
.....

```

Data Manipulation

- Deleting one row in table student

```
Dim cmd As New OleDbCommand
cmd.Connection = cn
cmd.CommandText = "DELETE FROM student WHERE [id]=1"
cmd.ExecuteNonQuery()
.....
```

- Updating one row in table student

```
Dim cmd As New OleDbCommand
cmd.Connection = cn
cmd.CommandText = "UPDATE student " &
    "SET [name]='Thida' " & _
    "WHERE [id]=1"
cmd.ExecuteNonQuery()
.....
```

Retrieve Data to ComboBox

```
Public Sub LoadStudent(cbo_name As ComboBox)
    Dim da As New OleDbDataAdapter("SELECT " & _
        & "[id], [name] FROM student",cn)
    Dim dt As New DataTable
    da.Fill(dt)

    cbo_name.DataSource=dt
    cbo_name.DisplayMember = "name"
    cbo_name.ValueMember="id"
End Sub
```

Retrieve Data to DataGridView

```
Private Sub LoadData()  
    Dim da As New OleDbDataAdapter("SELECT [id] " & _  
        ", [name] FROM student",cn)  
    Dim dt As New DataTable  
    da.Fill(dt)  
  
    DataGridViewName.DataSource=dt  
    With DataGridViewName  
        .Columns("id").HeaderText = "Student ID"  
        .Columns("name").HeaderText = "Student Name"  
    End With  
  
    dt.Dispose(): da.Dispose()  
End Sub
```

Assignments

- Create a Database in MS-Access 2003 as follow:
 - Student(stdid, stdname, gender, address, phone)
 - Subject(subid, subname)
 - Score(stdid, subid, semester, score)
- Create VB 2008 Project to:
 - Maintain student list, maintain subject list
 - Record student score for every subject in each semester