

- *VB.NET Syntax I
- 1. Elements of code
- 2. Declaration and statements
- 3. Naming convention
- 4. Types and user defined types
- 5. Enumerations and variables

GEOG 4/590: GIS Programming

Computer Terminologies

- Bit (0, 1)
- Byte (8 bits: 0 - 255)

- Numeral systems
 - Binary (bin) – 0,1
 - Decimal (dec) – 0, 1, 2, 3,...
 - Hexadecimal (hex) – 0, 1, 2, 3,...

Components of a programming language

- Structure
 - Modules, methods (procedures), lines of code
 - (OOP) Classes, properties, methods, events, lines of code
- Syntax
 - Rules for combining the language elements
- Elements
 - Statements, declarations, methods, operators, keywords

Statements

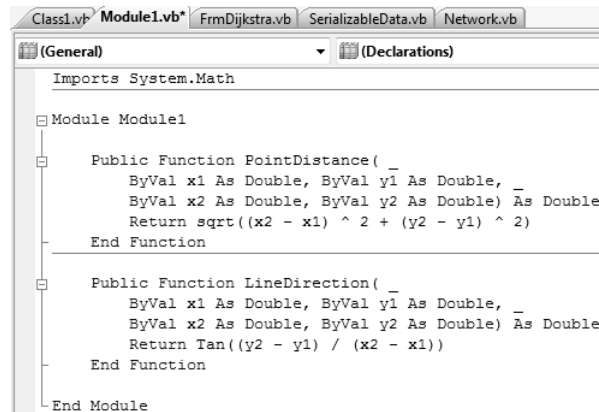
```
Dim a, b As Integer, c As Integer
Dim d As Integer, e As Boolean

a = 3          'a is 3
b = 2 + 5     'b is 7
c = a + b     'c is 10
d = d + 1     'd was 0 and now is 1
e = Not e     'e was false and now is true

If a > b Then c = 3
MsgBox("Value of C is " & c)
```

Modules

- A set of procedures (subroutines or functions)
 - The part of VB .NET that's not OOP



```

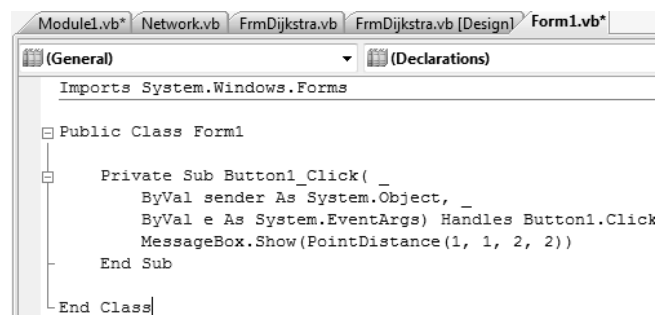
Class1.vb* Module1.vb* FrmDijkstra.vb SerializableData.vb Network.vb
(General) (Declarations)
Imports System.Math

Module Module1
    Public Function PointDistance( _
        ByVal x1 As Double, ByVal y1 As Double, _
        ByVal x2 As Double, ByVal y2 As Double) As Double
        Return sqrt((x2 - x1) ^ 2 + (y2 - y1) ^ 2)
    End Function

    Public Function LineDirection( _
        ByVal x1 As Double, ByVal y1 As Double, _
        ByVal x2 As Double, ByVal y2 As Double) As Double
        Return Tan((y2 - y1) / (x2 - x1))
    End Function
End Module
  
```

Classes

- A set of properties, methods, or events
 - Properties: object's attributes
 - Methods: object's actions
 - Events: object's responses to events

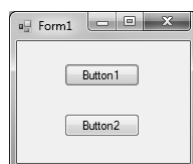


```

Module1.vb* Network.vb FrmDijkstra.vb FrmDijkstra.vb [Design] Form1.vb*
(General) (Declarations)
Imports System.Windows.Forms

Public Class Form1
    Private Sub Button1_Click( _
        ByVal sender As System.Object, _
        ByVal e As System.EventArgs) Handles Button1.Click
        MessageBox.Show(PointDistance(1, 1, 2, 2))
    End Sub
End Class
  
```

Class vs Modules



```

Start Page Object Browser Module1.vb Class1.vb Form1.vb Form1.vb [Design]
(General) (Declarations)
Imports System.Math

Public Class PointsCalculation
    Public Function PointDistance( _
        ByVal x1 As Double, ByVal y1 As Double, _
        ByVal x2 As Double, ByVal y2 As Double) As Double
        Return Sqrt((x2 - x1) ^ 2 + (y2 - y1) ^ 2)
    End Function
End Class

```

```

Start Page Object Browser Module1.vb Class1.vb Form1.vb* Form1.vb [Design]*
(General) (Declarations)
Public Class Form1
    Private Sub Button1_Click( _
        ByVal sender As System.Object, _
        ByVal e As System.EventArgs) Handles Button1.Click
        MessageBox.Show(PointDistance(1, 1, 2, 2))
    End Sub

    Private Sub Button2_Click( _
        ByVal sender As System.Object, _
        ByVal e As System.EventArgs) Handles Button2.Click
        Dim pcal As New PointsCalculation
        MessageBox.Show(pcal.PointDistance(1, 1, 2, 2))
    End Sub
End Class

```

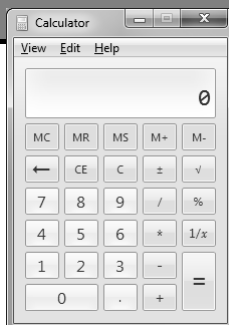
Methods

- A self-contained block of code that does something (aka **procedure**)
- Breaks up a program into manageable modules
- Promotes code reuse
- Some methods have input parameters, some have return values (i.e., **functions**) and some don't (i.e., **subroutines**)

Syntax

- Statements
- Declarations (of variables, objects, arrays, etc)
- Keywords
- Methods
- Operators

Computer Memories



- Memory address
- Pointer to address
- Name of pointer

Resource	Device	Status
0xFF97F800-0xFF97FFFF	Standard AHCI 1.0 Serial ATA Controller	OK
0xFE8D9C00-0xFE8D9F...	Intel(R) ICH9 Family USB2 Enhanced Host Contr...	OK
0xF0000000-0xFEC00000	PCI bus	OK
0xBFF00000-0xDFFFFFFF	PCI bus	OK
0xFF980800-0xFF9808FF	PCI bus	OK
0xFF980800-0xFF9808FF	Intel(R) ICH9 Family USB2 Enhanced Host Contr...	OK
0xFF97C000-0xFF97FFFF	PCI bus	OK
0xFED20000-0xFED9FFFF	PCI bus	OK
0xFEDAD800-0xFEDAD...	PCI bus	OK
0xFE8D9800-0xFE8D9B...	Intel(R) ICH9 Family SMBus Controller - 2930	OK
0xFE8DC000-0xFE8DFFFF	High Definition Audio Controller	OK
0xFE800000-0xFE8FFFFF	Intel(R) ICH9 Family PCI Express Root Port 1 - 2...	OK
0xD0000000-0xDFFFFFFF	ATI Radeon HD 2400 Pro	OK
0xD0000000-0xDFFFFFFF	Intel(R) Q35 Express Chipset PCI Express Root P...	OK
0xFE9F0000-0xFE9FFFFF	ATI Radeon HD 2400 Pro	OK
0xFE900000-0xFEAFFFFF	Intel(R) Q35 Express Chipset PCI Express Root P...	OK
0xFE8E0000-0xFE8FFFFF	Intel(R) 82566DM-2 Gigabit Network Connection	OK
0xFE8B0000-0xFE8BFFFF	Intel(R) 82566DM-2 Gigabit Network Connection	OK
0xFEDAD000-0xFEDAD...	PCI Simple Communications Controller	OK
0xFED00000-0xFED003...	High precision event timer	OK
0xFE8DA000-0xFE8DAF...	PCI Serial Port	OK
0x00000000-0x000000...	PCI bus	OK

Declarations – Dim Statement

When **Option Explicit** is on, VB requires a declaration of a variable or an object before it can be used (Don't turn Option Explicit off!).

```
Option Explicit On
Dim numberOfStudents As Integer
Dim courseNumber As String = "GEOG490" 'initializer
Dim a, b, c As Double, d As Integer

Dim pEnvelopeA As IEnvelope
pEnvelopeA = New CType(pActiveView.Extent, IEnvelope)

Dim pEnvelopeB As New IEnvelope = pEnvelopeA
Dim anObject As Object
```

Declarations – Variables Types

Visual Basic type	CLR type	Nominal storage allocation	Value range
Boolean	Boolean	Platform dependent	True or False
Byte	Byte	1 byte	0 through 255 (unsigned)
Char (single character)	Char	2 bytes	0 through 65535 (unsigned)
Date	DateTime	8 bytes	0:00:00 (midnight) on January 1, 0001 through 11:59:59 PM on December 31, 9999
Decimal	Decimal	16 bytes	0 through +/-79,228,162,514,264,337,593,543,959,335 (+/-7.9...E+28) * with no decimal point; 0 through +/-7.9228162514264337593543950335 with 28 places to the right of the decimal; smallest nonzero number is +/-0.000000000000000000000000000000000001 (+/-1E-28) *
Double (double-precision floating-point)	Double	8 bytes	-1.79769313486231570E+308 through -4.94065645841246544E-324 * for negative values; 4.94065645841246544E-324 through 1.79769313486231570E+308 * for positive values
Integer	Int32	4 bytes	-2,147,483,648 through 2,147,483,647 (signed)
Long (long integer)	Int64	8 bytes	-9,223,372,036,854,775,808 through 9,223,372,036,854,775,807 (9.2...E+18 *) (signed)
Object	Object (class)	4 bytes on 32-bit platform 8 bytes on 64-bit platform	Any type can be stored in a variable of type Object
SByte	SByte	1 byte	-128 through 127 (signed)
Short (short integer)	Int16	2 bytes	-32,768 through 32,767 (signed)
Single (single-precision floating-point)	Single	4 bytes	-3.4028235E+38 through -1.401298E-45 * for negative values; 1.401298E-45 through 3.4028235E+38 * for positive values
String (variable-length)	String (class)	Platform dependent	0 to approximately 2 billion Unicode characters
UInteger	UInt32	4 bytes	0 through 4,294,967,295 (unsigned)
ULong	UInt64	8 bytes	0 through 18,446,744,073,709,551,615 (1.8...E+19 *) (unsigned)
User-Defined (structure)	(inherits from ValueType)	Platform dependent	Each member of the structure has a range determined by its data type and independent of the ranges of the other members
UShort	UInt16	2 bytes	0 through 65,535 (unsigned)

Declaration – Option Explicit & Option Strict

- Option Explicit – prevents confusion in code or incorrect typing

```
a = 5
```

- Option Strict – prevents narrowing conversion

```
Dim a As Integer
```

```
a = 1.25
```

Scope in Visual Basic

Scope	Description	Declaration
Block	Available only within the code block in which it is declared	Dim
Procedure	Available to all code within the procedure in which it is declared	Dim
Module/ Class	Available to all code within the module, class, or structure in which it is declared	Public/ Private
Namespace (project)	Available to all code in the namespace (project) in which it is declared	Friend

Scope Example

```

Module Module1
    Public variable1 As Double
    Private variable2 As Double
    -----
    Public Sub Calculate()
        Dim variable3 As Integer = 2
        variable1 = 2 * variable3
        If variable3 < 5 Then
            Dim variable4 As Integer
            variable2 = Square(variable3)
            variable4 = Square(variable1)
        End If
    End Sub
    -----
    Private Function Square(ByVal x As Double) As Double
        Return x * x
    End Function
End Module

```

Declaration - Modifiers

- Public, Private - module/class level
 - Public maximumAllowed As Double
 - Private salary As Decimal
- Protected - class level only
 - Protected currentUser As String
- Friend - namespace/assembly level (cannot be used in procedures)
 - Friend projectName As String

Declaration - Static

- Specifies that one or more declared local variables are to remain in existence and retain their latest values after termination of the procedure in which they are declared.

```
Function updateSales(ByVal thisSale _  
As Decimal) As Decimal  
    Static totalSales As Decimal = 0  
    totalSales += thisSale  
    Return totalSales  
End Function
```

Declaration – Constant

- Declares and defines one or more constants
- Const can be used only at module or procedure level

```
Const maximum As Long = 459  
Public Const helpString As String = "HELP"  
Private Const startValue As Integer = 5
```

What is Enumeration?

- Enumerations offer an easy way to work with sets of related constants and to associate constant values with names
- Member name [= initializer]

```

Module Module2
  Public Enum flavorEnum
    salty
    sweet
    sour
    bitter
  End Enum

  Public Sub TestMethod()
    MsgBox("My favorite is " & flavorEnum.salty)
  End Sub
End Module

```



Enumeration Data Type

- With integer initializer...

```

Module Module2
  Public Enum flavorEnum
    salty = 1
    sweet = 2
    sour = 3
    bitter = 4
  End Enum

  Public Sub TestMethod()
    MsgBox("My favorite is " & flavorEnum.salty)
  End Sub
End Module

```

Access all members in Enum

- Enum.GetNames and Enum.GetValues methods

```

Module Module2
    Public Enum flavorEnum
        salty = 1
        sweet = 2
        sour = 3
        bitter = 4
    End Enum

    Public Sub TestMethod()
        MsgBox("The strings in the flavorEnum are:")
        Dim i As String
        For Each i In [Enum].GetNames(GetType(flavorEnum))
            MsgBox(i)
        Next
        MsgBox("My favorite is " & flavorEnum.salty)
    End Sub
End Module

```

Enumeration

- Enumerations make for clearer and more readable code, particularly when meaningful names are used.
- Benefits :
 - Reduces errors caused by transposing or mistyping numbers
 - Makes it easy to change values in the future
 - Makes code easier to read, which means it is less likely that errors will creep into it
 - Ensures forward compatibility

Intrinsic Constants and Enumerations in VB .NET

MsgBoxResult Enumeration

Collapse All

Indicates which button was pressed on a message box, returned by the **MsgBox** function.

Remarks

Members

Member	Constant	Description
OK	vbOK	OK button was pressed.
Cancel	vbCancel	Cancel button was pressed.
Abort	vbAbort	Abort button was pressed.
Retry	vbRetry	Retry button was pressed.
Ignore	vbIgnore	Ignore button was pressed.
Yes	vbYes	Yes button was pressed.
No	vbNo	No button was pressed.

- See [http://msdn.microsoft.com/en-us/library/t9o4xoth\(VS.8o\).aspx](http://msdn.microsoft.com/en-us/library/t9o4xoth(VS.8o).aspx)

Declaration Context Levels

Declared element	Namespace	Module/Class	Procedure
Variable (Dim Statement)	Not allowed	Private (Public in Structure)	Public
Constant (Const Statement)	Not allowed	Private (Public in Structure)	Public
Enumeration (Enum Statement)	Friend	Public	Not allowed

Naming Convention

- VB .NET is NOT case sensitive, however...
- Names
 - A list of identifiers concatenated to form a name
- Pascal case
 - ErrorLevel
- Camel case
 - totalNetworkDistance
- Uppercase
 - System.IO

Naming Convention - Principles

- Be consistent!
- Don't use "reserved" terms (e.g., `Double`, `False`)
- Use Pascal casing for namespaces, classes, members, methods, and constants
 - `Const MapUnits As String = "meters"`
- Use Camel casing for variables and parameters
 - `Dim isVisible As Boolean`
- Do not use underscores, hyphens, or any other nonalphanumeric characters

Visual Basic Language Keywords			
URL: http://msdn.microsoft.com/en-us/library/ksh7h19t(VS.71).aspx			
AddHandler	AddressOf	Alias	And
AndAlso	Ansi	As	Assembly
Auto	Boolean	ByRef	Byte
ByVal	Call	Case	Catch
CBool	CByte	CChar	CDate
CDec	Cdbl	Char	CInt
Class	CLng	CObj	Const
CShort	CSng	CStr	CType
Date	Decimal	Declare	Default
Delegate	Dim	DirectCast	Do
Double	Each	Else	Elseif
End	Enum	Erase	Error
Event	Exit	False	Finally
For	Friend	Function	Get
GetType	GoSub	GoTo	Handles
If	Implements	Imports	In
Inherits	Integer	Interface	Is
Let	Lib	Like	Long
; Loop	Me	Mod	Module
MustInherit	MustOverride	MyBase	MyClass
Namespace	New	Next	Not
Nothing	NotInheritable	NotOverridable	Object
On	Option	Optional	Or
OrElse	Overloads	Overridable	Overrides
ParamArray	Preserve	Private	Property
Protected	Public	RaiseEvent	ReadOnly
ReDim	REM	RemoveHandler	Resume
Return	Select	Set	Shadows
Shared	Short	Single	Static
Step	Stop	String	Structure
Sub	SyncLock	Then	Throw
To	True	Try	TypeOf
Unicode	Until	Variant	When
While	With	WithEvents	WriteOnly
Xor	#Const	#ExternalSource	#If...Then...#Else
#Region	-	&	&=

String Class

```
Dim lastName As String
lastName = "brown"

Msgbox(lastName.ToUpper) 'return "BROWN"

Msgbox(String.Compare(lastName.ToUpper, _
    "WHITE") 'return -1

Msgbox(String.Compare(lastName.ToUpper, _
    "BROWN") 'return 0
```

Structures

```
Public Structure SystemInfo
    Public CPU As String
    Public Memory As Long
    Public PurchaseDate As Date
End Structure

Public Class Class1
    Public Function MyFunction
        Dim MySystem, YourSystem As SystemInfo

        MySystem.CPU = "486"
        Dim TooOld As Boolean
        If YourSystem.PurchaseDate < #1/1/1992# Then TooOld =
            True

        YourSystem = MySystem
    End Function
End Class
```

Data Structures

- Array
- Arraylist (queue and stack)
- Hashtable
- Binary (search) tree
- Graph