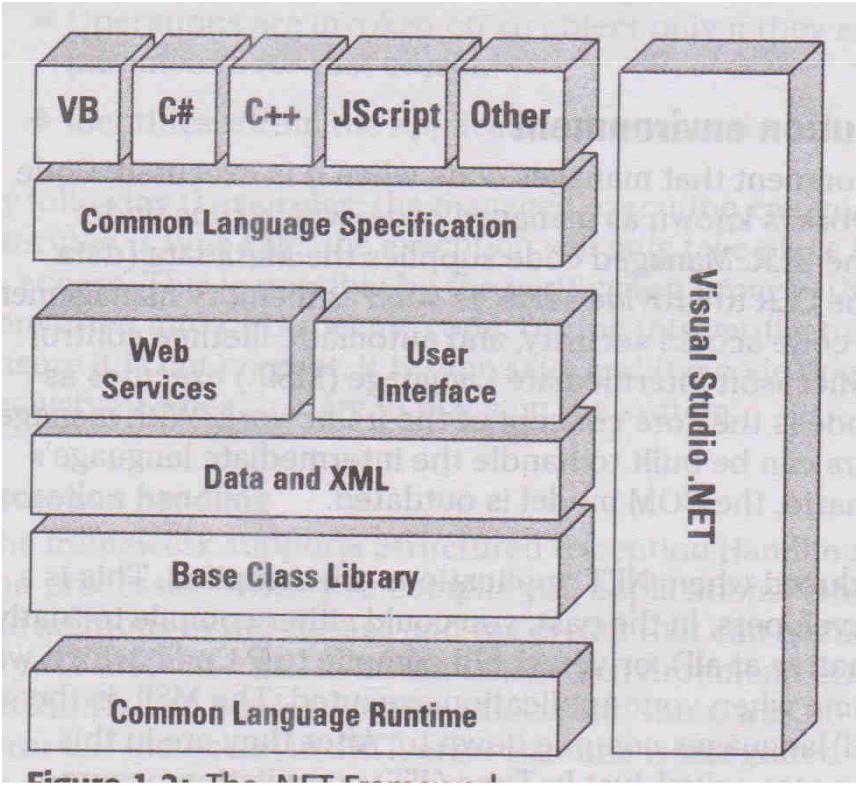
VB.NET

Dr. Binu P Chacko
Associate Professor
Department of Computer Science
Prajyoti Niketan College, Pudukad

.NET framework

- .NET is a platform for building XML web services
- A unified environment for developing client and server based applications and services



Common Language Runtime

Goals

- Secure and robust execution environment: framework supports structured exception handling (SEH) across languages and processes
- Object lifetime is managed through a process called garbage collection
- Simplified development: OO simplifies reuse and interoperability between components
- Tools in VS.NET are useful to develop large-scale, distributed applications
- Multilanguage support: 18 languages
- Simplified deployment and management

VB.NET

Components

- Language innovations: OO language – inheritance, polymorphism, encapsulation, overloading, overriding
- Error handling with SEH
- RAD features: improved designers, server explorers, data designers, XML designers
- Auto-complete and auto-list members are used to verify the code
- Web forms: to develop web based applications
- Web services: create a Web Service callable method by prefixing a method with <webmethod>
- Windows forms

VB.NET IDE

Windows

- Solution Explorer contains list of files. Right-click to view project options
- Class View contains classes and object. Double-click to get the details
- Server Explorer manages all servers and served based services. To view server resources, data connections, e-mail configuration and Web Service management
- Properties window: to modify the properties of the object
- Toolbox contains objects and controls that can be added to Forms
- Macro Explorer: to write macros
- Object Browser: to view objects and their properties and methods. Object Pane, Members Pane, Description Pane
- Task list gives error details and helps to document the project
- Command window is used for executing commands
- Output window displays build errors, libraries loaded, compiler specific information
- Debugging windows: Breakpoints, Autos, Locals, Watch, Call Stack

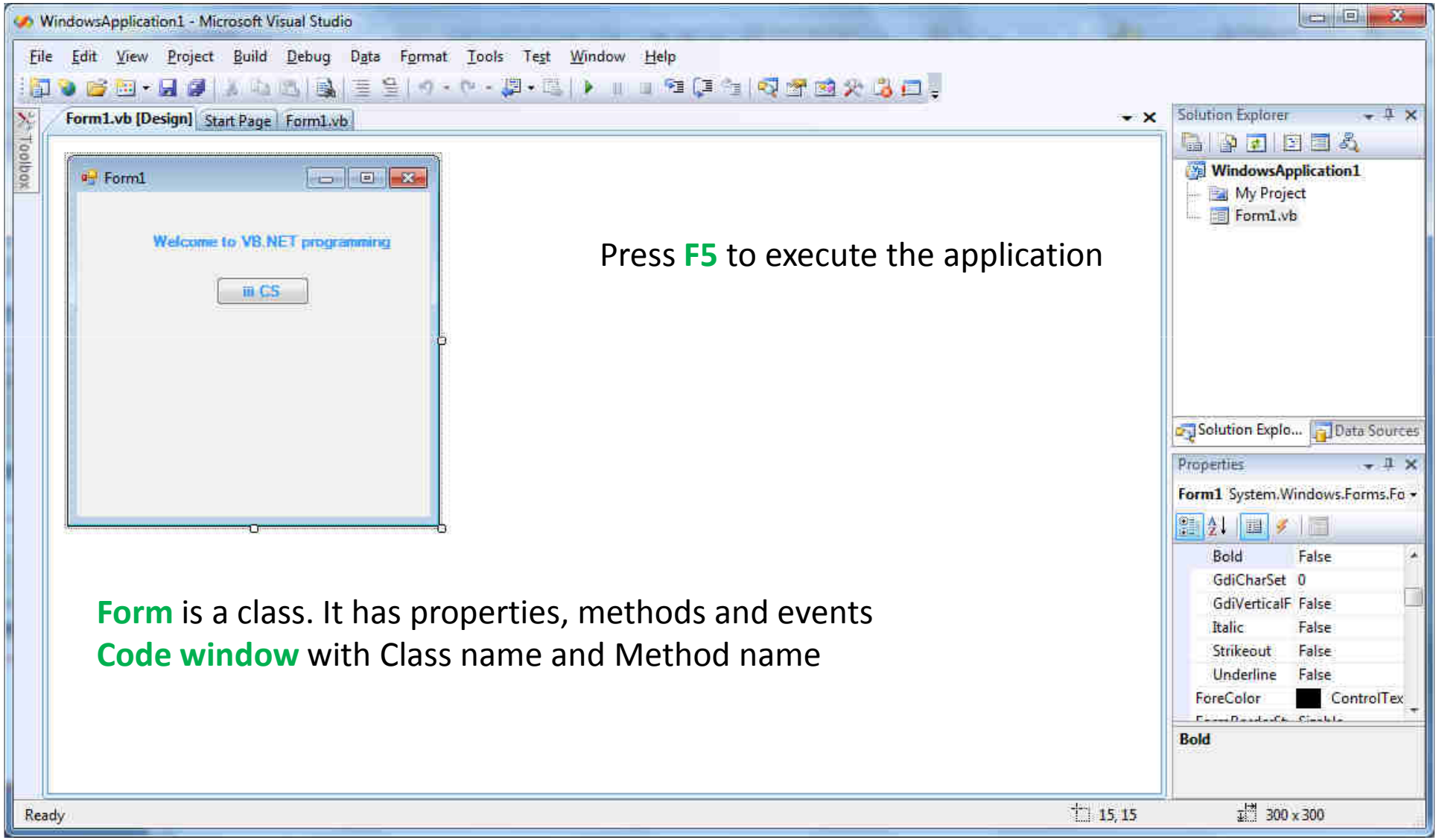
Cont…

- Code editor: Auto List Members, Auto Complete, spell checking

Designers

- Windows Forms designer is used to create forms-based applications
- Web Forms designer for ASP.NET application
- XML designer to create and edit XML and XSD files
- Database designer – database creation tool
- Query designer creates complex queries
- Component designer creates non-graphical middle-tier components
- User Control designer for visual creation of controls to be used in ASP.NET applications

Windows Forms



Controls

- Input controls: Button, RadioButton, ListBox, ListView, HScrollBar, VScrollBar, TextBox, RichTextBox, CheckBox, MenuStrip, CheckedListBox, ComboBox, ContextMenu, DataGridView, PropertyGrid, DateTimePicker, MonthCalendar, ToolStrip, TrackBar, TreeView, DomainUpDown, NumericUpDown
- Display controls: Form, PictureBox, GroupBox, Panel, Label, LinkLabel, ProgressBar, StatusBar, Splitter, TabControl, ToolTip
- Dialog controls: ColorDialog, FontDialog, PrintDialog, OpenFileDialog, PageSetupDialog, SaveFileDialog
- ImageList

Data Types

Type	Bytes	Value	literal	identifier
Boolean	4	True/False		
Byte	1	0 to 255		
Char	2	0 to 65,535		
Date	8	1—1-1 to 31-12-9999		
Decimal	12		@	D
Double	8		#	R
Integer	4		%	I
Long	8		&	L
Short	2	-32,768 to 32,767		S
Object	4	Any type of variables		
Single	4		!	F
String	2*length	2 billion Unicode characters	$	C
structure		Sum of size of its members		

Cont…

- Reference type: String, Arrays, Classes
- Value type: Numeric, Boolean, Char, Date, Structures, Enumerations
- Private x As Single = 123.23
- Dim a, b as integer
- Dim a as integer, b as integer
- Public p as integer
- Default values – numbers: 0, boolean: False, String: nothing, object: nothing, date: 12:00:00 AM
- Option Explicit On variable declaration is compulsory
- Dim I%, AM#, LAZY$
- Dim Price as Object = 120.45D Forces decimal type
- Constants: Private Const str As String = “Try Again”

Scope of variable/constant

- Private, Shared, Public, Protected, Static, Friend
- Private can’t be used to declare local variables in a function
- Private Shared x As Integer
- Shared members are shared by all instances of a class
- Public Class College
- Static sum as Integer = 0
- Retain the values within a method or class
- Protected area as Double
- Available in the class and its derived classes
- Friend item as String
- Accessible from any class or module

Data Type Conversion

- Conversion functions: CBool, CByte, CChar, CDate, CDbl, CDec, CInt, CLng, CObj, CShort, CSng, CStr, CType
- Dim a as single = 23.89
- Dim b as Integer
- MessageBox.Show(CInt(a))
- b = Fix(a) or b=Int(a) No rounding off
- System.Convert members: ToBoolean, ToByte, ToChar, ToDateTime, ToDecimal, ToDouble, ToInt16, ToInt32, ToInt64, ToSByte, ToSingle, ToString, ToUInt16, ToUInt32, ToUlint64
- Option Strict On allows widening conversion; but complier error for narrowing conversion

Operators

- Arithmetic operators: +, -, *, /, \, Mod, ^
- Concatenation operator: &
- Assignment operators: =, +=, -=, *=, /=, \=, ^=, &=
- Comparison operators: <, >, <=, >=, =, <>, Is, Like
- Logical (AndAlso, OrElse)/Bitwise (on numeric expressions) operators: And, Or, Not, Xor
- Operator precedence: ^, - (negation), (*, /), \, Mod, (+, -), bitwise operators, &, comparison operators, Not, And, Or, Xor

Conditional Logic

```
If Quantity > 20 Then
    Discount = 20
ElseIf Quantity > 10 Then
    Discount = 10
Else
    Discount = 5
End If
```

- Nested If

Cont…

```
Select Case Month
    Case 1
        MsgBox("January")
    Case 2
        MsgBox("February")
    Case Else
        MsgBox("Next Year")
End Select
Select Case Order
    Case Is <=10
        Discount = 10
    Case Is <= 20
        Discount = 15
    Case Else
        Discount = 20
End Select
```

```
Select Case Num
    Case 5, 7, 11
        MsgBox("Prime number")
    Case 4, 6, 8
        MsgBox("Even number")
    Case Else
        MsgBox("Incorrect")
End Select
```

```
Select Case Order
    Case 1 To 10
        Discount = 10
    Case 11 To 20
        Discount = 15
    Case Is > 20
        Discount = 20
End Select
```

Looping

```
Do While n < 10
    n = n + 1
Loop
While n < 10
    n = n + 1
End While
For c = 1 To 10
    MsgBox("VB.NET")
Next c
For Each arrayelement in array
    MsgBox(arrayelement)
Next
```

MessageBox Class

- InputBox (Prompt, *Title, DafultValue, x, y*)
- MsgBox (Message, *Constant, Title*)
- MessageBox.Show (Text, Caption, Buttons, Icon, DefaultButton, Options)
- MessageBox.Show (Owner, Text, Caption, Buttons, Icon, DefaultButton, Options)
- MessageBoxButton. OK, OKCancel, YesNo, RetryCancel, YesNoCancel, AbortRetryIgnore
- MessageBoxIcon. Error, Question, Exclamation, Information
- MessageBoxDefaultButton. Button1, Button2, Button3
- MessageBoxOptions. RightAlign, RtlReading

System.String class

Clone()	Makes an exact copy of an object	Compare(), CompareOrdinal()	Compares to strings
CompareTo()	Compares a string to this string object	Concat()	Joins one string to another
Copy()	Copies one string object to another string object	CopyTo()	Copies characters into an array
EndsWith(), StartsWith()	Tests the beginning and ends of a string	Equals()	Tests the equality of one object to another
Format()	Formats numeric strings	GetHashCode()	Gets the hash code of the string
IndexOf(), LastIndexOf()	Gets the index location of characters in a string	Insert()	Inserts substring into a string
Intern(), IsInterned()	Obtains a reference to a string	Join(), Split()	Joins or split strings based on a parameter
PadLeft(), PadRight()	Pads a string with additional characters	SubString()	Isolate a substring from a string
Remove()	Removes characters from a string	Replace()	Replace characters in a string
Trim(), TrimEnd(), TrimStart()		Trims characters from the beginning and ends of strings	

Arrays

- Dim s1(10) As String
- Dim s2() As String = {“B.A”, “B.Sc”, “B.Com”}
- Dim Appln() as String — Dynamic array
- ReDim Appln(10) as String — Changes the size of the array
- ReDim Preserve Appln(10) As String — Preserves the existing values
- Dim n(5,10) As Integer
- Erase s1, s2
- a = n.GetUpperBound(0)
- a = n.GetLowerBound(1)
- a = n.GetLength(0)
- n.SetValue(100, 0, 3)
- Dim x()() as integer = New Integer(2)() {}
- x(0) = New Integer (3) {1, 2, 3, 4}
- x(1) = New Integer (2) {21, 22, 23}
- MessageBox.Show(x(1)(2))
- Dim y(2) as Object
- Dim name(5) as string
- Dim age(5) as Integer
- y(0) = name
- y(1) = age
- MessageBox.Show(y(0)(3))

Procedure

- Types of procedures: Sub (don't return a value), Function (return a value)
- Sub Main()

  [statements]

  End Sub
- [<attrlist>] [{Overloads|Overrides|Overridable|NotOverridable|MustOverride| Shadows|Shared}] [{Public|Protected| Friend|Protected Friend|Private}]

  Sub name [(arglist)]

  [statements]

  [Exit Sub]

  [statements]

  End Sub
- [Call] <ProcedureName> ([Argument list])

Function

- [<attrlist>] [{Overloads|Overrides| Overridable| NotOverridable|MustOverride| Shadows|Shared}] [{Public|Protected| Friend|Protected Friend|Private}] Function name [(arglist)] [As type]

  [statements]

  [Exit Function]

  [statements]

  End Function
- Function returns a value: Function_name = return_value or Return return_value
- Function call: Variable = Function_name([Arguments list])

Built-in Functions

Microsoft.VisualBasic.Conversion

- MessageBox.Show(ErrorToString(13))
- MessageBox.Show(Fix(x))
- MessageBox.Show(Int(y))
- MessageBox.Show(Hex(11))
- MessageBox.Show(Oct(12))
- MessageBox.Show(Str(10))
- MessageBox.Show(Val("23rd June"))

Microsoft.VisualBasic.DateAndTime

- DateVar = DateAdd(DateInterval, Number, DateValue)
- LongVar = DateDiff(DateInterval, Date1, Date2, *FirstDayOfWeek, FirstWeekOfYear*)
- IntVar = DatePart(DateInterval, DateValue, *FirstDayOfWeek, FirstWeekOfYear*)
- DateVar = DateSerial(Year, Month, Day)
- DateString()
- DateVar = DateValue("June 28, 2017")
- IntVar = Day("June 28 2017")
- DateVar = #06/28/2017 2:01:15 PM#
- IntVar = Month(DateVar)
- IntVar = Year(DateVar)
- IntVar = Hour(DateVar)
- IntVar = Minute(DateVar)
- IntVar = Second(DateVar)

Cont…

- StrVar = MonthName(6, *True*)
- Console.Write(Now)
- Console.WriteLine(Today)
- Console.WriteLine(TimeOfDay)
  Console.WriteLine(TimeString)
- Console.Write(Timer)
- DateVar = TimeSerial(Hour, Minute, Second)
- DateVar = TimeValue(Now)
- IntVar = Weekday(DateValue, *FirstDayOfWeek*)
- Console.Write(WeekdayName(4,*False*,*FirstDayOfWeek*))

Microsoft.VisualBasic.Strings

- Messagebox.Show(Asc(“A”))
- Messagebox.Show Chr(65)
- Dim s(2) as String
- s(0) = “We”
- s(1) = “are”
- s(2) = “Indians”
- Dim s1 as String = Filter(s, “are”, *True*, *CompareMethod.Text*)
- Dim s2 As String = “ Programming ”
- Console.Write(GetChar(s1, 4))
- FormatNumber(Number, *NumOfDecimalPlace, IncludingLeadingDigit, ParenthesesForNegativeNumbers, GroupDelimiter*)
- FormatCurrency(), FormatPercent()
- FormatDateTime(DateTime, *DateFormat*)

Cont…

- InStr(*StartNo*, String1, String2, *CompareMethod*)
- InStrRev(StringCheck, StringMatch, *StartNo, CompareMethod*)
- s2 = Join(s, " ")
- Console.Write(LCase("COMPUTER"))
- MessageBox.Show UCase("science")
- s1 = Left(s2, 4)
- Console.Write(Len(s2))
- Console.Write(LSet(s1, 10))
- Console.Write(RSet(s1, 10))
- Console.Write(LTrim(s2))
- Console.Write(RTrim(s2))
- Console.Write(Trim(s2))

Cont…

- Console.Write(Mid(StrVar, StartNo, *Length*))
- Replace(StrVar, FindStr, RepStr, *StartNo, Count, CompareMethod*)
- Spc(10)
- StrArr = Split(StrVar, *Delimiter, LimitNo, CompareMethod*)
- StrComp(str1, str2, *CompareMethod*)
- StrConv(StrVar, ConversionType, *LocaleID*)
- StrReverse(StrVar)

Windows Form

Hierarchy

- Control class: foundation of all controls
- It handles creation and destruction of windows handles, background and foreground colours and text
- It provides functionality for input, pointing devices and bounding
- Scrollable Control: derived from Control
- It adds vertical and horizontal scroll, and auto scrolling capability to controls, and place scroll bars
- ContrainerControl: derived from ScrollableControl
- It adds the ability to place other controls along with other functions (including focus management and tabbing)
- Form: capability to have caption bar, irregular windowing, menus, and default controls
- Introduces the concepts of modality and MDI
- UserControl: gives the ability to create controls

Features

- Windows Form is a fully extensible and inheritable class with visual capabilities
- It has many properties and methods
- Dockable toolbar
- It is opaque
- Capability to capture mouse wheel
- Object oriented approach for graphics
- Border styles are available via FormBorderStyle property
- Autoscroll
- Resizable window – grounds the controls in their respective positions in relation to their container
- Everything measured in pixels, so very consistent interface development

Controls

ButtonBase	
Member name	**Description**
FlatStyle	Values: Flat, PopUp, Standard, System
Image	Specifies the image
ImageAlign	Specifies the alignment of the image
ImageIndex	Specifies the index of the image in image list displayed on Button
ImageList	Specifies the ImageList
TextAlign	Specifies alignment of the text
IsDefault	Specifies whether or not a Button is default button
ResetFlagsAndPaint	Reset control flags to default values and redraw the control
Button	
DialogResult	Specifies Ok, Cancel, Retry or Abort on button click
PerformClick	Generates a click event

Cont…

CheckBox (inherited from ButtonBase)	
Appearance	Specifies the appearance (button or check box) of CheckBox
AutoCheck	Specifies whether Checked and CheckState properties are updated
CheckAlign	Specifies vertical and horizontal alignment
Checked	Specifies whether checkbox is checked
CheckState	Checked, Unchecked, Indeterminate
ThreeState	Specifies whether three states are allowed
AppearanceChanged	Occurs when the value of Appearance changes
CheckStateChanged	Occurs when the value of CheckState changes

ListBox

DefaultItemHeight	Specifies default item height
NoMatches	Specifies that no matching items were found during a search
ColumnWidth	Specifies the width of columns in a multicolumn list box
DrawMode	Specifies whether OS handles drawing of the control or code handles the drawing
HorizontalExtent	Specifies the width by which horizontal scrollbar can scroll
HorizontalScrollBar	Specifies whether horizontal scrollbar is displayed on the control
IntegralHeight	Specifies whether the control should resize to avoid showing partial values
ItemHeight	Returns height of an item
Items	Returns all items in the list box
MultiColumn	Specifies whether the ListBox supports multiple columns
PreferredHeight	Returns the combined height of all items
ScrollAlwaysVisible	Specifies whether the vertical scrollbar is always visible
SelectedIndices	Returns the indices of all selected items
SelectedItem	Currently selected item
SelectedIndexChanged	Occurs when the index of the selected item has changed
CreateItemCollection	Creates a new instance of the items collection

SelectedItems	Returns currently selected items
SelectionMode	None, One, MultiSimple, MultiExtend
Sorted/Sort	Specifies whether the items in the list are sorted alphabetically
TopIndex	Specifies the index of the top item in the list
UseTabStops	Specifies whether the list box can recognize Tab characters
BeginUpdate	Suspends the painting of the control while items are being added to items collection
ClearSelected	Deselect all selected items
EndUpdate	Resumes painting of the control after all items are added
FindString	Finds the specified text within the items
FindStringExact	Finds only exact matches of the specified string
GetItemHeight	Returns the height of the specified item
GetItemRectangle	Returns the bounding rectangle for an item
GetSelected	Returns a value indicating whether item is selected or not
IndexFromPoint	Returns the index of the item at specified coordinates
SetSelected	Toggles the selection for the specified item
DrawItem	Occurs when a visual aspect of the control changes
MeasureItem	Occurs when list box is created and sizes of list items are determined

CheckedListBox

CheckedIndices	Returns the indices of all checked items
CheckedItems	Returns a collection of all checked items
CheckOnClick	Specifies whether an item should be checked or not when it is selected
Items	Returns all items in this control
ThreeDCheckBoxes	Check boxes appear as 3D
GetItemChecked	Returns a value corr: to the item checked
GetItemCheckState	Returns the CheckState of the current item
SetItemChecked	Sets the CheckState of the specified item
SetItemCheckState	Sets the CheckState of the item at the specified index
ItemCheck	Occurs when CheckState of an item changes

- HScrollBar and VScrollBar controls are inherited from ScrollBar class

RadioButton

Appearance	Specifies the current appearance of the control
AutoCheck	Specifies whether the value and appearance of the control automatically change
CheckAlign	Specifies the alignment
Checked	Specifies whether the control is checked
PerformClick	Click on the control as if it is done by the user
AppearenceChanged	Occurs when the appearance of the control is changed
CheckedChanged	Occurs when the checked value of the control is changed

TextBox

AcceptsReturn	Creates newline or activates default button when Enter key is pressed
CharacterCasing	Control modifies the case of the text
PasswordChar	Character to be displayed while typing password
Scollbars	To use scrollbars in the text box
TextAlign	Specifies text alignment
TextAlignChanged	Occurs when alignment is changed

RichTextBox

AutoWordSelection	Whether the selection of a part of word selects the entire word
BulletIndent	Whether an indentation is used when a bullet style is applied
CanRedo	Returns a value indicating Redo
DetectURLs	Automatically format URLs
RedoActionName	Returns name of the action that is reapplied
RightMargin	Specifies right margin
RTF	Rich Text Formatting codes
ScrollBars	Specifies the type of scrollbar
SelectedRTF	Currently selected RTF text
SelectionAlignment	Alignment of current selection or insertion point
SelectionBullet	To apply bullet style to current selection or insertion point
SelectionCharOffSet	To appear the text as normal, superscript or subscript
SelectionColor	Colour of the selected text
SelectionFont	Font of the selected text

SelectionHangingIndent	Distance of indentation
SelectionIndent	Distance between left edge of the control and that of selected text
SelectionProtected	Selected text cannot be edited
SelectionRightIndent	Distance between right edge of the control and that of selected text
SelectionTabs	Specifies Tab stop positions
SelectionType	Returns selection type of the control
ShowSelectionMargin	Specifies whether the selection margin is displayed within the control
Redo	Reapplied the last undone operation
UndoActionName	Returns the name of action when Undo method is called
ZoomFactor	Specifies the zoom factor
CanPaste	Returns a value indicating the status of Paste operation
Find	Returns the location of the sought text
GetCharFromPosition	Returns a character at the specified location
GetCharIndexFromPosition	Returns the index of the character at the specified position
IMEChange	Occurs when IME device has changed

Cont…

LoadFile	Loads the contents of the specified file
SaveFile	Saves the contents of the control to a file
ContentsResized	Occurs when contents of the control have been resized
HScroll	Occurs when user clicks horizontal scrollbar
VScroll	Occurs when user clicks vertical scrollbar
LinkClicked	Occurs when a link in the text is clicked
Protected	Occurs when a user attempts to modify the protected text
SelectionChanged	Occurs when selection within the control has changed
CreateRichEditOLECallBack	This method is useful for COM based communication

•Useful for timed, interactive processes. Raises an event at regular intervals. Inherited from Component

Timer

Enabled	Timer is enabled
Interval	Specifies the time interval in milliseconds
Start	Initiates the timer
Stop	Ceases the timer
Tick	Occurs at specified intervals

ComboBox

DrawMode	Specifies whether OS or code handles the drawing of the control
DropDownStyle	DropDown, DropDownList, Simple
DropDownWidth	Specifies the width of drop down portion
DroppedDown	Specifies whether the control is displaying drop down portion
IntegralHeight	Specifies whether the control should resize to avoid showing partial values
ItemHeight	Returns the height of the item
Items	Returns a collection containing all items
MaxDropDownItems	Specifies the number of items to be shown in drop down portion
PreferredHeight	Returns the preferred height of an item
SelectedItem	Specifies currently selected item
SelectedText	Specifies the selected text within the editable portion
SelectionLength	Specifies the number of characters selected within the editable portion
SelectionStart	Specifies the starting position of the selected text in the editable portion

Sorted	Specifies whether the items are sorted
BeginUpdate	Suspends painting of the control while items are being added
EndUpdate	Resumes painting of the control after all items are added
FindString	Finds the specified text within the items
FindStringExact	Finds exact matches of the specified string
GetItemHeight	Returns height of the specified item
Select	Selects specified range of text within the editable portion
SelectAll	Selects all text within the editable portion
DrawItem	Occurs when a visual aspect of the control changes
DropDown	Occurs when list portion of the control is displayed
DropDownStyleChanged	Occurs when style of the control is changed
MeasureItem	Occurs when size of the items are determined
SelectedIndexChanged	Occurs when index of the selected item has changed
SelectionChangeCommitted	Occurs when change has been committed for the new selection
AddItemCore	Used to add a collection of items to the control

AutoSize	Automatically resizes to display all contents
BorderStyle	FixedStyle, Fixed3D, None
FlatStyle	Popup, Standard, Flat, System
Image	Image to be displayed on the label
ImageAlign	Alignment of the image
ImageList	Specifies the ImageList object
ImageIndex	Specifies the index value in the image list
IMEmode	IME mode supported by this control
PreferredHeight	Preferred height of the control based on the font
PreferredWidth	Preferred width of the control based on the font
TabStop	Specifies whether the user can tab to the label
UseMnemonic	To use an ampersand in front of a character as access key
AutoSizeChanged	Occurs when AutoSize property changes
TextAlignChanged	Occurs when text alignment changes
RenderTransparent	Specifies whether the background of the label should be the container control background
CalcImageRenderBounds	Determines the size and location of an image within the control
DrawImage	Draws the image within the specified bounds

Label

ToolTip

- Inherited from Component class

Active	Specifies whether the control is currently active
AutomaticDelay	Time required to appear the tool tip
AutoPopDelay	Amount of time tool tip remains visible
InitialDelay	Specifies initial delay
ReshowDelay	Amount of time being showings of the ToolTip
ShowAlways	Show always even if the control is not active
GetToolTip	Retrieves the text of the ToolTip
RemoveAll	Removes all tool tips
SetToolTip	Associates tool tip to the specified control

PictureBox

BorderStyle	None, FixedSingle, Fixed3D
Image	Specifies the image to be displayed
IMEMode	Specifies IME mode
SizeMode	Normal, StretchImage, AutoSize, CenterImage, Zoom
SizeModeChanged	Occurs when SizeMode property changes

Inherited from Control class

GroupBox	
FlatStyle	Flat, Standard, Popup, System

[<attrlist>][Public|Private|Protected|Friend| Protected Friend]
[Shadows][MustInherit| NotInheritable]

Class name

[Implements interfacename]

[statements]

End Class

Classes & Objects

Public	Unrestricted access to entities within the class
Private	Members are accessible only within their declaration context
Protected	Members are accessible within the class and in the derived class
Friend	Members are accessible only within the program
Protected Friend	Combination of Protected and Friend
Shadows	Class shadows an identically named programming element within the class
MustInherit	Methods in the class must be implemented by the derived class
NotInheritable	Does not allow inheritance
Interface	Name of the interface implemented by the class
Statements	Events, properties and fields

Inside the Class

- Methods

```
Class Class1
    Public Function Set_Rank(ByVal strRank As
    String) As Boolean
        ‘ function code
    End Function
End Class
```

Cont…

- Properties: used to store variables in a class – getting and setting variables

```
Class Class1
    Private strRank as String
    Public Property Rank() as String
        Get
            return strRank
        End Get
        Set (ByVal value as String)
            strRank = value
        End Set
    End Property
End Class
```

```
Class Class1
  Private Rank as String
  Public Function Set_rank(strRank as String)
     as Boolean
    Rank = strRank
  End Function
  Public Function Get_rank() as String
    Return Rank
  End Function
End Class
```

Cont…

- Fields: variables declared in public scope. Used in place of properties

```
Shared Class Class1
    Public Rank as String
End Class
```

Cont…

- Events: notifications that cause something to happen or occur in response to something happening

```
Public Class Class1
    Private strRank as String
    Event Promoted(ByVal newRank as String)
    Public Property Rank() as String
        Get
            Return strRank
        End Get
        Set(ByVal value as String)
            strRank = value
            If value = “captain” Then
                RaiseEvent Promoted(strRank)
            End If
        End Set
    End Property
End Class
```

- Encapsulation: hide the implementation details, and modify the code without affecting the front-end application

```
Public Class Person
    Private Name As String
    Public Property StudentName() As String
        Get
            Return Name
        End Get
        Set(ByVal Value As String)
            Name = Value
        End Set
    End Property
    Public ReadOnly Property NameUC() As String
        Get
            Return Name.ToUpper
        End Get
    End Property
    Public Overridable Function Save() As String
        Return "Person Saved"
    End Function
End Class
```

OOP

```
Dim p As New Person()
p.StudentName = "Amstrong"
Console.WriteLine(p.NameUC)
```

Property or method to be overridden in a derived class

Inheritance

```
Public Class Customer
    Inherits Person
    Private Ctg As String
    Public Property Category() As String
        Get
            Return Ctg
        End Get
        Set(ByVal Value As String)
            Ctg = Value
        End Set
    End Property
    Public Overrides Function Save() As String
        Return "Customer Saved"
    End Function
End Class
```

```
Dim cust As New Customer()
cust.Category = "Buyer"
cust.StudentName = "Sampras"
Console.WriteLine(cust.NameUC)
Console.WriteLine(cust.Category)
```

Overrides an Overridable property or method defined in the base class

Polymorphism

- Methods and properties in different classes have same name and used interchangeably

```
Public Class Employee
    Inherits Person
    Private Slr As Double
    Public Property Salary() As Double
        Get
            Return Slr
        End Get
        Set(ByVal Value As Double)
            Slr = Value
        End Set
    End Property
    Public Overrides Function Save() As String
        Return “Employee Saved”
    End Function
End Class
```

Cont…

```
Dim emp As New Employee()
emp.StudentName = “Rodricks”
emp.Salary = 13000.00
MessageBox.Show(emp.Save())
Console.WriteLine(emp.NameUC)
Console.WriteLine(emp.Salary)
Dim cust As New Customer()
cust.StudentName = “Juliet”
Cust.Category = “Seller”
Console.WriteLine(cust.NameUC)
Console.WriteLine(cust.Category)
MessageBox.Show(cust.Save())
```

Exception Handling

- Types of Errors: syntax error, logical error, runtime error
- Option Explicit On – forces you to declare a variable before it is referenced
- Option Strict On – notify you against illegal data conversion
- Try…End Try

```
Private Sub Button_Click(…)
    GetFileName(TextBox1.Text)
End Sub
Private Function GetFileName(ByVal strName As String) As String
    Try
        If strName <> “ ” Then
            Return “OK”
        End If
    Catch
        Return “Enter valid file name”
    Finally
        Enter some code
    End Try
End Function
```

Exceptions are objects raised at runtime to abnormal behaviour in the application

Zero or more Catch blocks

Zero or more Finally blocks. Code will execute whether there is an exception or not

Exception Class properties	
HelpLink	Gets or sets location of the help file
HResult	A coded numerical value assigned to an exception
InnerException	Gets a reference to inner exception
Message	Gets a string representation of the error message

Cont...

Properties and methods of Err object	
Clear	Clear the Err object
GetException	Gets the exception that represents the error
GetType	Returns the type of current instance
Raise	Raises an error
Description	Description for the current error number
Erl	Returns the line number of the last executed statement
HelpContext	An integer value that represents the Context ID in the help file
HelpFile	Sets the path to the help file
LastDLLError	Returns a system error code produced by a call to an extended DLL
Source	Application or object name from the error occurred
Number	Returns the numeric value of the error

On Error Goto

```
Private Sub ErrorHandlerTest()
    On Error Goto errHandler
        some code
    Exit Sub
errHandler:
    MsgBox("An error occurred")
End Sub
```

```
On Error Goto -1
    Disable any exception
```

```
Private Sub DisableHandler()
    On Error Goto errHandler
    Dim cn As SqlConnection
    cn.Open()
        Connection is open with no error
    On Error Goto 0
        Disable any further error trap
    Exit Sub
errHandler:
    Handle the error
End Sub
```

```
Private Sub DeleteAndCopyFile()
    On Error Resume Next
    If Dir("C:\file.txt") <> " " Then
        Kill("C:\file.txt")
        If Err.Number <> 0 Then
            Err.Clear()
        End If
        FileCopy("E:\File.txt", "C:\File.txt")
        If Err.Number <> 0 Then
            MsgBox("Can't copy the file")
        End If
End Sub
```

Variations of Resume

```
Private Sub CopyFile()
    On Error Goto errHandler
    Dim strSource, strDest As String
GetFileName:
    strSource = InputBox("Enter source file name")
    strDest = InputBox("Enter destination file
    name")
    If strSource <> "" And strDest <> "" Then
            FileCopy(strSource, strDest)
            MsgBox("File copied")
    Else
            Err.Raise(53)
    End If
    Exit Sub
```

```
errHandler:
    Select Case Err.Number
        Case 53
            MsgBox("File doesn't exist")
            Err.Clear()
            Resume GetFileName
        Case 71
          If MsgBox("Insert disk in drive",
             MsgBoxStyle.RetryCancel) =
               MsgBoxResult.Retry Then
                  Resume
          End If
        Case Else
            MsgBox(Err.Numer & "-" &
                          Err.Description)
            Resume Next
        End Select
End Sub
```