CLR(Common Language Runtime) :

Features of CLR(Common Language Runtime) :

- Exception Handling
- Garbage Collection
- Thread Management
- Type Safety
- Security
- Performance Optimization

NameSpace:- Software projects consist of several pieces of code such as classes, declarations, procedures and functions etc., known as the component or identifiers of the software project. In large projects the number of these components can be very large. These components can be grouped into smaller subcategories. This logical grouping construct is known as a "Namespace" or **we can say that the group of code having a specific name is a "Namespace".** In a Namespace the groups of components are somehow related to each other. Namespaces are similar in concept to a folder in a computer file system.

- we can access a member of a Namespace by using a dot(.) operator

```
1. Ex. Imports System
```

FCL(Framework Class Library) :- .NET Framework Class Library is the collection of classes, namespaces, interfaces and value types that are used for .NET applications. This is also called as Base Class Library and it is common for all types of applications i.e. the way you access the Library Classes and Methods in VB.NET will be the same in C#, and it is common for all other languages in .NET.

Ex. System.Data.OleDb

The following are different types of applications that can make use of .net class library.

1. Windows Application.
2. Console Application
3. Web Application.
4. XML Web Services.
5. Windows Services.

Common Type System (CTS) :- Common Type System (CTS) describes a set of data types, it ensure that objects written in different .Net supported languages can interact with each

other. **For example**, C# has int Data Type and VB.Net has Integer Data Type. Hence a variable declared as int in C# or Integer in vb.net, finally after compilation, use the same structure Int32 from CTS.

Common Language Specification (CLS) It is a sub set of CTS and it specifies a set of rules that needs to be satisfied by all language compilers targeting CLR. It helps in cross language inheritance and cross language debugging.

Garbage Collection : Garbage collection is the mechanisam to releasing memory from unused objects and components of the application. Language such as C++ do not have any garbage collection system, therefor developers have to manually clean the memory. Visual Basic provides automatically Garbage collection system.

The Option Statement : =

Option Explicit statement When Option Explicit statement is set to ON, you must explicitly declare all variables using the Dim or ReDim statements. By default it in ON.

- Option Explicit statement Must be first line in the code

```
Option Explicit Off
Module VBModule
   Sub Main()
      'Dim x As Integer → if Option Explicit On then we must have to declare it
      x = 100  'if  Option Explicit Off then we can directly declare variable like this
      Console.WriteLine("Hello World" & x)
   End Sub
End Module
```

Option Strict Statement If Option Strict Statement is set to ON, Visual Basic prevents (disallows) implicit narrowing data type conversion.

```
Option Strict On
Module VBModule
   Sub Main()
      Dim x As Double
      Dim y As Integer
      y = 651.72
      ' x = y   '→ throw complete time error if  Option Strict On
     x = CDbl(y) ' Explicitly converting y to Double
      Console.WriteLine("Hello World" & x)
   End Sub
End Module
```

Boxing and UnBoxing :

Boxing is the process of converting a value type (such as an integer or a structure) to a reference type (such as an object). When you box a value type, a new object is

created on the heap, and the value of the value type is copied into that object. This allows you to treat the value type as an object.

```
Dim intValue As Integer = 10
Dim obj As Object = intValue
```

Unboxing is the reverse process of boxing. It involves extracting the value stored in a boxed object and converting it back to its original value type.

```
Dim obj As Object = 10 ' Boxed integer value
Dim intValue As Integer = DirectCast(obj, Integer) ' Unboxing occurs here
```

- The **constants** refer to fixed values that the program may not alter during its execution. These fixed values are also called literals.

  Const constname As datatype = value

Enumeration:
Enumeration is a user-defined data type used to define a related set of constants as a list Enumerations provide a way to define a collection of related symbolic values that can be easily referenced and used throughout your code.

```
Enum Months
   January = 1
   February = 2
   March = 3
   ' ...
End Enum

Dim today As DaysOfWeek
today = DaysOfWeek.Monday
```

• The **ByVal keyword** indicates that an argument is passed in such a way that the called procedure or property cannot change the value of a variable passed as the argument in the calling code.
• The **ByRef keyword** indicates that an argument is passed in such a way that the called procedure can change the value of a variable passed as the argument in the calling code.

```
Sub Example2(ByRef test AsInteger) test = 10 EndSub
```

Keywords :

Me keyword ⇒ The Me keyword provides a way to refer to the specific instance of a class or structure in which the code is currently executing. Me behave like either an object variable or a structure variable referring to the current instance.
MyBase Keyword ⇒ You can use the MyBase keyword to access methods in a base class when overriding methods in aderived class

MyClass keyword ⇒ The MyClass keyword behaves like an object variable referring to the current instance of a class as originally implemented.
⇒ MyClass is similar to Me, but all method calls on it are treated as if the method were NotOverridable.

Statement in VB :

a statement is a syntactic unit of code that performs a specific action. Statements are used to instruct the computer to perform tasks, manipulate data, control program flow, and interact with the environment.

- Assignment Statements
- Control Flow Statements
- Looping Statements
- Exit Statements
- Error Handling Statements
- Declaration Statements

Procedure :

a procedure is a named block of code that performs a specific task or operation. Procedures are also known as subroutines or functions, depending on whether they return a value. They are essential for organizing and structuring code, promoting code reuse, and improving maintainability.

1. **Sub Procedures**(Doesn't Return Value)
   eg. Sub DisplayMessage()
   Console.WriteLine("Hello, World!")

   End Sub
2. **Function Procedures** (Return value)
   Eg. Function AddNumbers(num1 As Integer) As Integer
   Return num1
   End Function

Default Argument :

```
Sub Greet(name As String, Optional greeting As String = "Hello")
    Console.WriteLine($"{greeting}, {name}!")
End Sub
```

Arrays and Collections :

Array :

An **array** is a linear data structure that is a collection of data elements of the same type stored on a **contiguous memory** location. Each data item is called an element of the array. It is a fixed size of sequentially arranged elements in computer memory with the first element being at **index 0** and the last element at index **n - 1,** where **n** represents the total number of elements in the array.

Syntax:

```
Dim numbers(4) As Integer
```

Collections :

Collections are dynamic data structures that can grow or shrink in size at runtime. Unlike arrays, collections can store elements of different data types and provide additional functionality for manipulation, such as adding, removing, and searching for elements.

```
Dim arrayList As New ArrayList()
arrayList.Add("Apple")
arrayList.Add("Banana")
arrayList.Add("Orange")
```

Control Flow Statements :

Control flow statements in Visual Basic (VB.NET) are used to control the flow of program execution based on certain conditions or loops. They allow you to make decisions, iterate over collections, and execute blocks of code repeatedly.

Conditional Statements :

Conditional statements allow you to execute different blocks of code based on specified conditions

Looping Statements :

For...Next Statement:

```
For variable As [Data Type] = start To end
// Statements to Execute
Next
```

For I As Integer = 0 To 10

Next

For Each...Next Statement:

```
For Each element As ElementType In collection
    ' Code block to execute
Next
```

While...End While Statement:

```
While condition
    ' Code block to execute
End While
```

Do...Loop Statement:

```
Do
   ' Code block to execute
Loop While condition
```

MsgBox Function:

The MsgBox function displays a message in a dialog box and returns a value indicating which button the user clicked.

MsgBox(prompt [, buttons] [, title] [, helpfile, context])

result = MsgBox("Do you want to continue?", MsgBoxStyle.YesNo, "Confirmation")

InputBox Function:

The InputBox function prompts the user to enter a value into a dialog box and returns the value entered by the user.

InputBox(prompt [, title] [, defaultResponse] [, xpos] [, ypos])

userInput = InputBox("Enter your name:", "Name Entry", "John Doe")

String Functions And Properties :

- Len() // Len("Hello") → 5
- Trim() // Trim(" Hello ") → Hello
- UCase // LUase("Hello") → HELLO
- LCase
- Left() // Left("Hello", 3) → Hel
- Right()
- Mid() // Mid(Hello,2,3) → all
- Replace() // Replace("Hello World", "World", "Universe") → Hello Universe

Properties : str.Length, str.Chars(0), str.ToUpper(), str.ToLower()

Modal Dialogs:

- A **modal dialog** is one that **blocks user interaction** with other parts of the application until it is closed.
- When a modal dialog is open, you **cannot interact** with anything else within your program until you deal with and close the modal dialog.

Eg, **File-Save As** dialogs.

Modeless Dialogs:

- A **modeless dialog** behaves like a **normal window**.
- You can **interact freely** with other parts of your application while a modeless dialog is open

The default value for a Date variable in VB is `#12:00:00 AM#` on January 1, 0001.

The default value for a Boolean variable in VB is `False`.

ReDim keyWord

the **ReDim** keyword is used to dynamically resize the dimensions of an array at runtime. This allows you to change the size of an array after it has been declared, either increasing or decreasing its size. The ReDim statement can be used with or without the Preserve keyword.

With Preserve : preserve the datamember and change its size

Without Preserve : When you use ReDim without the Preserve keyword, it creates a new array with the specified dimensions and copies the existing elements to the new

UnManaged Code:

Unmanaged code refers to programming code that is executed outside the scope of the Common Language Runtime (CLR) environment in the .NET Framework. In other words, unmanaged code is code that is not managed by the CLR and does not adhere to the rules and benefits of managed code execution.

ParamArray

you can use the ParamArray keyword to define a parameter array in a subroutine or function, allowing you to pass a variable number of arguments. Here's how you can use ParamArray to pass a variable number of arguments in a subroutine:

```
Sub PrintValues(ParamArray values() As Integer)
    For Each value As Integer In values
        Console.WriteLine(value)
    Next
End Sub
```

Syntax of Switch case :

```
Select case variable

  case val1
        statements
  case val2
        statements
  case val3
        statements
……….
  case else
        statements
End select
```

¬ **There are two main styles of user interface:**
1. Single-document interface (SDI)
2. Multiple-document interface (MDI).

The **multiple-document interface (MDI)** allows you to create an application that maintains multiple forms within a single container form. An MDI application allows us to display or open multiple documents at the same time.
IsMDIContainer = true

Label Control

In VB.NET, a label control is used to display descriptive text for the form in control. It does not participate in user input or keyboard or mouse events. Also, we cannot rename labels at runtime. The labels are defined in the class System.Windows.Forms namespace.

Button Control

Button control is used to perform a click event in Windows Forms, and it can be clicked by a mouse or by pressing Enter keys. It is used to submit all queries of the form by clicking the submit button or transfer control to the next form. However, we can set the buttons on the form by using drag and drop operation.
Properties : BackColor, ForeColor, BackgroundImage, Image, Text
Methods : click

TextBox Control:

TextBoxes are used to accept input from the useror used to display text. By default we can enter up to 2048 characters ina TextBox but if the Multiline property is set to True we can enter up to32KB of text.
Properties : Multiline, PasswordChar, TextAlign, WordWrap(the multiline Textbox control automatically wraps words to the beginning of the next line when necessary)

RichTextBox :

It's an advance version of TextBox. It provides more advanced text formatting option. It can load RTF ,TXT format files for reading or editing. The wordpad save it‘s file in RTF format. The full form of RTF is Rich Text Format. RichTextBox allows formatting the text, say adding colors, displaying particular font types,selected text effect, bullets, alignment,indents and so on. By default RichTextBox is multiline. By default maxlength of it 2147483647

Properties:

1) SelectionAlignment
2) SelectionBackColor
3) SelectionBullet
4) SelectionFont
5) SelectionColor
6) SelectionLength
7) AcceptsTab

Methods :

1) Clear()
2) Cut()
3) Copy()
4) Undo()
5) Redo()

Events :

1) TextChanged
2) SelectionChanged
3) LinkClicked
4) KeyPress
5) MouseClick

Features :

- Formatted Text Editing
- Clipboard Operation
- Undo and Redo
- HyperLinks
- Event Handling
- Insertion of Images and Objects
- Find and Replace
- Spell Checking

Combobox Control :

The ComboBox control is used to display more than one item in a drop-down list. It is a combination of Listbox and Textbox in which the user can input only one item.Furthermore, it also allows a user to select an item from a drop-down list.

Properties:

- Items
- SelectedIndex
- SelectedItem
- SelectedText
- Sorted (True/False)
- Text (to set Text programmically)
- MaxDropDownItems

Methods:

- FindString()
- Items.Add()
- Items.Clear()
- Items.Remove()
- Items.RemoveAt(index)

Events :

- DropDown
- TextChanged
- SelectedIndexChanged

Message Box

The show method of MessageBox is used to display User Specific message in a Dialog Box and waits for the user to click a button.It returns an integer value indicating which button is click by user.

SYNTAX :

MessageBox.show(Text As String(Text), Text As String(Title), Button As from.windows, Icons As windows.form)

Possible Buttons :
MessageBoxButtons.OKOnly, OKCancel, AbortRetryIgnore, YesNoCancel, YesNo, CancelRetry
Possible Icons : MessageBoxIcon.Critical, Question, Exclamation, Information

InputBox

If user clicks on OK button then it will return content of textbox in the form of string. If user clicks on Cancel Button then it will return a blank value
SYNTAX
InputBox (Prompt As String,[Title As String=""], [DefaultResponse As String=""], [XPos As Integer = -1], [YPos As integer = -1])

TreeView()

The TreeView control is used to display a hierarchical representation of the same data in a tree structure. The top-level in the tree view is the root node with one or more child nodes. In addition, the root node can be contracted or expanded by clicking on the plus sign (+) button. It is also useful to provide the full path of the root node to the child node.

```
1. Dim nd As String
2.         nd = InputBox("Enter Node name")  ' takes input from the user
3.         If TreeView1.SelectedNode Is Nothing Then  ' insert a new node.
4.             TreeView1.Nodes.Add(nd, nd)
5.         Else
6.             TreeView1.SelectedNode.Nodes.Add(nd, nd)   ' insert into the sub node or chi
ld node
7.         End If
```

Timer Control:

The Timer control in Visual Basic is used to execute code at specified intervals, allowing you to create time-based events or perform periodic tasks.

- The Timer control allows you to trigger events or execute code at regular intervals, such as updating the UI, performing background tasks, or creating animation effects.

Properties:

- Interval
- Enabled

- Tag

Methods:

- Start()
- Stop()

Events : Tick

Error Provider :

The ErrorProvider control in Visual Basic (VB) is used to provide visual feedback about errors or validation failures associated with other controls on a form. It helps users identify and correct errors in form input by displaying an icon or tooltip near the control with an error message.

Properties :

- ErrorIcon (ErrorProvider1.ErrorIcon = My.Resources.ErrorIcon)
- IconPadding (ErrorProvider1.IconPadding = 5)
- IconAlignment (ErrorProvider1.IconAlignment = ErrorIconAlignment.MiddleRight)
- ContainerControl (ErrorProvider1.ContainerControl = Me)

Methods :

- Clear() eg. ErrorProvider1.Clear()
- setError() eg. ErrorProvider1.SetError(TextBox1, "Please enter a valid value")

Help Provider :

The HelpProvider control is used to provide contextual help to the user. It associates Help text with individual controls. The help provider to the help text when the user press F1 key.

Properties:

- SetHelpString()
- GetHelpString
-

```
Sub Example()
  On Error GoTo ErrorHandler
```

```
    ' Code that may potentially throw an exception
    Dim result As Integer = 10 / 0 ' Attempting to divide by zero
    Exit Sub ' Exit the subroutine if no error occurs
ErrorHandler:
    ' Handling the exception
    Console.WriteLine("An error occurred: " & Err.Description)
End Sub
```

Dialog box

Create an Instance : Dim saveFileDialog1 As New SaveFileDialog()
Set Properties :

```
saveFileDialog1.InitialDirectory = "C:\"

saveFileDialog1.FileName = "example.txt"

saveFileDialog1.Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*"
```

Display the Dialog Box : Dim result As DialogResult = saveFileDialog1.ShowDialog()

Process User Input :

```
If result = DialogResult.OK Then

    Dim filePath As String = saveFileDialog1.FileName

    ' Perform save operation using the selected file path

End If
```

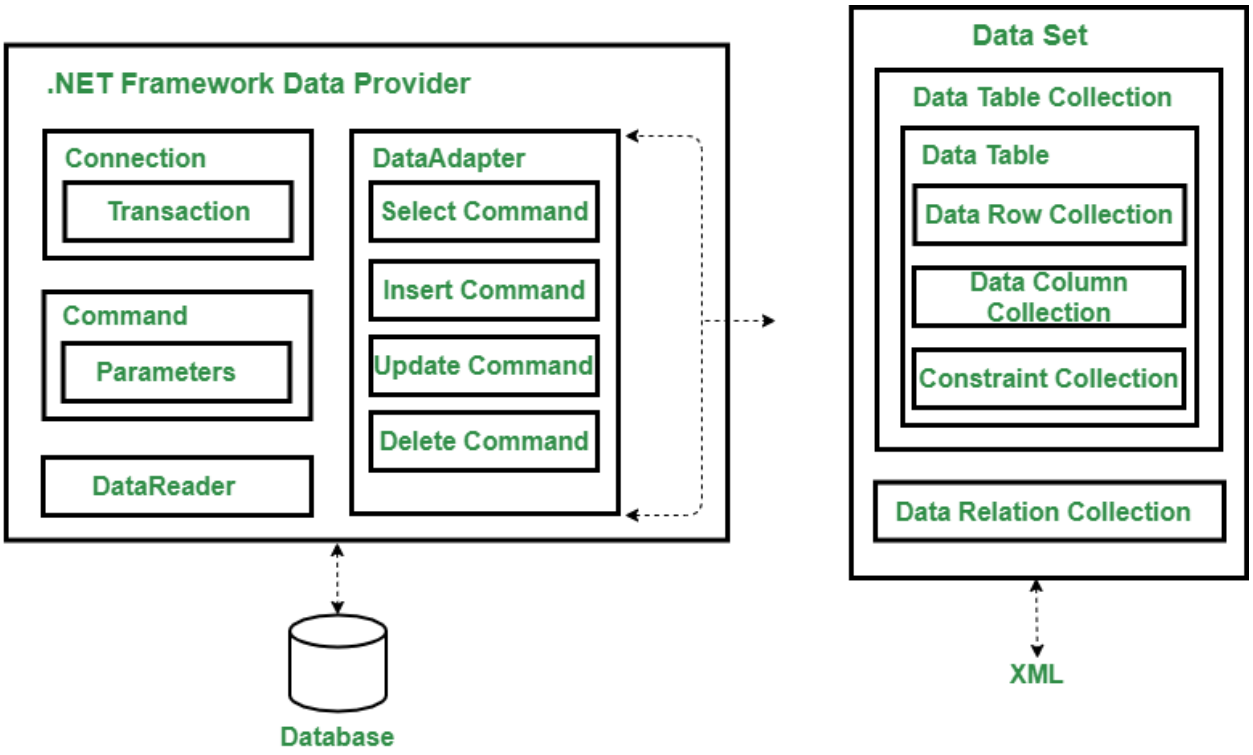
ADO.NET (Active Data Objects for .NET)

ADO.NET is an object-oriented set of libraries that allows you to interact with data sources. Commonly, the data source is a database, but it could also be a text file, an Excel spread-sheet, or an XML file. There are many different types of databases available such as Microsoft SQL Server, Microsoft Access, Oracle, Borland Interbase, IBM DB2, etc.

Connected & Disconnected Data (Architecture)

The data access with ADO.NET consists of two parts:

(1) Data Provider (2) DataSet



The Connection Object

The Connection object creates the connection to the database. Microsoft Visual Studio .NET provides two types of Connection classes: the SqlConnection object, which is designed specifically to connect to Microsoft SQL Server 7.0 or later, and the OleDbConnection object, which can provide connections to a wide range of database types like Microsoft Access and Oracle. The Connection object contains all of the information required to open a connection to the database.

Connection class exist for ODBC (OdbcConnection), OLE DB (OledbConnection), SQL Server (SqlConnection) & Oracle (OracleConnection)

Property :

- ConnectionString // holds the string used to open a database
- ConnectionTimeout
- DataSource // This property gets the name of the server providing the data.
- State
- ServerVersion
- Provider

Methods :

- Open(), Close(), BeginTransaction()

For Details Click here

The Command Object

The Command object is represented by two corresponding classes: SqlCommand and OleDbCommand. Command objects are used to execute commands to a database across a data connection. The Command objects can be used to execute stored procedures on the database, SQL commands, or return complete tables directly.

Command objects provide three methods that are used to execute commands on the database:

ExecuteNonQuery: Executes commands that have no return values such as INSERT, UPDATE or DELETE

ExecuteScalar: Returns a single value from a database query

ExecuteReader: Returns a result set by way of a DataReader object

Properties :

- CommandText
- CommandType
- Connection
- Parameters

For Details Click here

The DataAdapter Object

The DataAdapter is the class at the core of ADO .NET's disconnected data access. It is essentially the middleman facilitating all communication between the database and a DataSet. The DataAdapter is used either to fill a DataTable or DataSet with data from the database with it's Fill method. After the memory-resident data has been manipulated, the DataAdapter can commit the changes to the database by calling

the Update method.

Methods :

- Fill
- FillSchema
- Update
- FillAsync
- ResetFillLoadOption

Properties :

- SelectCommand
- InsertCommand
- UpdateCommand
- DeleteCommand

DataSet

The dataset is a disconnected, in-memory representation of data. It can be considered as a local copy of the relevant portions of the database. The DataSet is persisted in memory and the data in it can be manipulated and updated independent of the database. When the use of this DataSet is finished, changes can be made back to the central database for updating.

Property :

(1) Relations : It is the collection of DataRelation objects, which defines the relationship of the DataTables within the dataset.
(2) Tables : It is the collection of DataTables contained in the dataset.

Method :

- AcceptChanges //Commits all the pending changes made to the dataset.
- Clear
- Clone
- Copy
- GetChanges
- Reset
- RejectChanges // Rolls back the changes made to the dataset since it was created or since the AcceptChanges method was called

DataReader:

A DataReader is a forward-only, read-only cursor used to retrieve data from a data source.It provides a fast, efficient way to read data from a database one record at a time. DataReader retrieves data on demand from the data source.

```
datareader = cmd.ExecuteReader()
While datareader.Read
    MsgBox(datareader(Name))
End While
```

Properties:

- Item[index or name]
- FieldCount //Get the number of columns in the current row
- IsClosed // Check if the reader is closed
- HasRows // Check if the reader has rows
- RecordsAffected //returns the number of rows affected by the last SQL statement

Methods:

- Read()
- Close()
- GetName(Index)
- GetValue(index of column)
- IsDBNull(index)
- Dispose()

Data Provider :

A DataProvider is a set of related components that work together to provide data in an efficient and performance driven manner. The .NET Framework currently comes with Four DataProviders:

- SQL Data Provider,
- ODBC Data Provider,
- OLEDB Data Provider (object linking and embedding database)
- Oracle Data Provider

Component :

- Connection
- Command
- DataReader
- DataAdapter