CSA 7112T VB.NET

Syllabus & Course Outline

- Introduction to .NET, .NET Framework features & architecture, CLR, Common Type System, MSIL, Assemblies and class libraries.
- Introduction to visual studio, Project basics, types of project in .Net, IDE of VB.NET (Menu bar, Toolbar, Solution Explorer, Toolbox, Properties Window, Form Designer, Output Window, Object Browser).
- Event driven Programming – Methods and events.
- The VB.NET Language: Variables (Declaring, Data Types, Scope & lifetime), Constants, Arrays (types, control array), Collections.
- Subroutines, Functions (Passing variable/optional arguments, Returning values).
- Control flow statements: Conditional statements, Loop statements. Msgbox & Inputbox.
- Working with Forms: Loading, showing and hiding forms, controlling One form within another.
- GUI Programming with Windows Form: Textbox, Label, Button, Listbox, Combobox, Checkbox, PictureBox, RadioButton (Properties, Methods, events).
- Dialogs: OpenFileDialog, SaveFileDialog, FontDialog, ColorDialog, PrintDialog. Designing menus (ContextMenu).
- Database programming with ADO.NET – Overview of ADO, from ADO to ADO.NET, Accessing Data using Server Explorer. Creating Connection, Command, Data Adapter and Data Set with OLEDB and SQLDB. Display Data on data bound controls/grid.

HandNotes
by
Kamal Kishor

Unit 1: Introduction to .NET and VB.NET

1. Introduction to .NET

What is .NET?

.NET is a software development framework developed by Microsoft that provides a common platform for building and running applications such as:

- Windows Applications
- Web Applications
- Console Applications
- Web Services

.NET supports multiple programming languages like: VB.NET, C#, F#, C++/CLI

All these languages use the same runtime and class libraries.

2. .NET Framework Architecture

The .NET Framework mainly consists of three layers:

1. Common Language Runtime (CLR)
2. Framework Class Library (FCL)
3. Application Layer

2.1 Common Language Runtime (CLR)

CLR is the heart of the .NET Framework. It manages the execution of .NET programs.

Functions of CLR:

- Memory management
- Garbage collection
- Exception handling
- Security enforcement
- Thread management
- Code execution

Any .NET program must run inside CLR.

Unit 1: Framework Components (FCL, CTS, MSIL)

2.2 Framework Class Library (FCL)

FCL is a huge collection of predefined classes used for:

- File handling
- Database access
- GUI development
- Networking
- Security

Examples:

```
System.IO
System.Data
System.Windows.Forms
System.Net
```

These classes reduce development time and improve reliability.

3. Common Type System (CTS)

CTS defines how data types are declared, used, and managed in .NET.

Purpose of CTS:

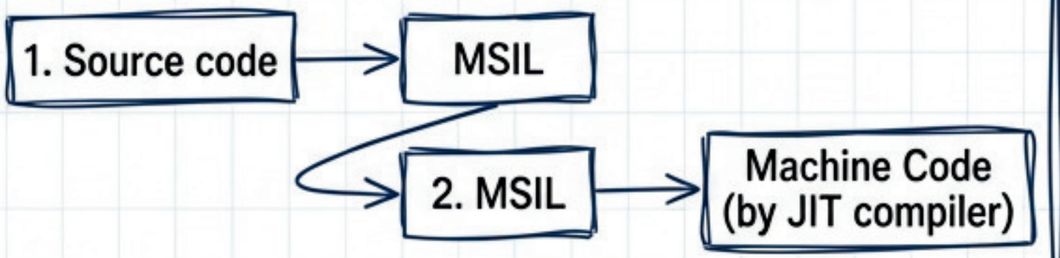
- Ensures language interoperability
- Provides a common data type system

Examples: Integer, Boolean, String, Object

VB.NET and C# share the same data types internally.

4. MSIL (Microsoft Intermediate Language)

When a VB.NET program is compiled:



MSIL is CPU independent.

5. Assemblies

An assembly is the compiled output of a .NET program.

Types of Assemblies:

1. Private Assembly – Used by one application
2. Shared Assembly – Used by multiple applications (stored in GAC)

Assembly contains:

- MSIL code
- Metadata
- Resources

IDE Overview & Unit 2: VB.NET Basics

6. Introduction to Visual Studio IDE

Visual Studio is an Integrated Development Environment (IDE).

Components:
Menu Bar, Toolbar, Solution Explorer, Properties Window, Form Designer, Toolbox, Output Window

UNIT 2: VB.NET Language Basics

1. Variables

What is a Variable? A variable is a named memory location used to store data.
Declaring Variables:

```
Dim a As Integer
Dim name As String
```

2. Data Types

Data Type	Description
Integer	Stores integer values
Double	Stores decimal values
Boolean	True / False
String	Text
Char	Single character
Object	Universal type

Premium Digital Student Notes

Unit 2: Scope, Constants & Arrays

3. Scope and Lifetime of Variables

Scope:

- Local – Inside a procedure
- Class-level – Inside a class
- Global – Throughout application

Lifetime:

- Exists as long as scope is active

4. Constants

Constants store fixed values.

```
Const PI As Double = 3.14
```

Value cannot be changed during execution.

5. Arrays

What is an Array?
An array stores multiple values of same type.

Declaring Array:

```
Dim marks(4) As Integer
```

Types of Arrays:

- Single dimensional
- Multi-dimensional
- Jagged arrays

Unit 2: Control Structures & Procedures

6. Control Structures

Conditional Statements:

If...Then

```
If a > b Then
    MsgBox("A is greater")
End If
```

If...Else

```
If a > b Then
    MsgBox("A is greater")
Else
    MsgBox("B is greater")
End If
```

Looping Statements:

For Loop

```
For i = 1 To 10
    MsgBox(i)
Next
```

While Loop

```
While i <= 5
    i += 1
End While
```

7. Functions and Subroutines

Sub Procedure:

Does not return value.

```
Sub Display()
    MsgBox("Hello")
End Sub
```

Function:

Returns value.

```
Function Add(a As Integer, b As Integer) As Integer
    Return a + b
End Function
```

Interaction & Unit 3: Windows Forms

8. MessageBox and InputBox

MsgBox:

```
MsgBox("Welcome to VB.NET")
```

InputBox:

```
Dim name As String
name = InputBox("Enter Name")
```

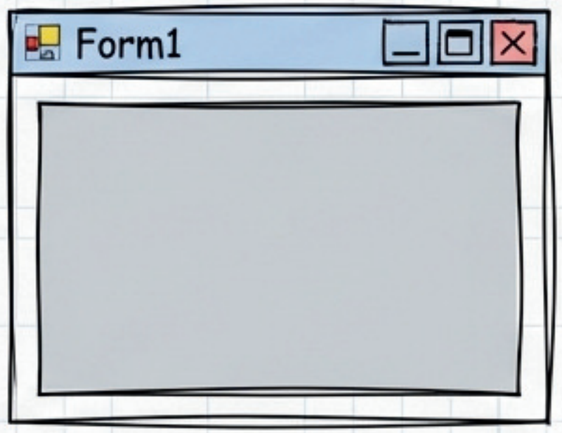
UNIT 3: Windows Forms and GUI Programming

1. Windows Forms

Windows Forms are used to create graphical user interface (GUI) applications.

Form Properties:

- Name
- Text
- BackColor
- Font



Unit 3: Common Controls & Dialogs

2. Common Controls

TextBox: Used to input text.

```
TextBox1.Text
```

A **Label**: Displays text.

Button: Executes action on click.

```
Private Sub Button1_Click()
    MsgBox("Clicked")
End Sub
```

CheckBox: Allows multiple selections.

RadioButton: Allows single selection.

ComboBox: Drop-down list.

ListBox: Displays list of items.

PictureBox: Displays images.

ComboBox: Drop-down list.

ListBox: Displays list of items.

PictureBox: Displays images.

---

3. Dialog Boxes

- OpenFileDialog
- SaveFileDialog
- FontDialog
- ColorDialog
- PrintDialog

---

4. Menu Controls

****ContextMenu**: Provides right-click options.

What is Event Driven Programming?

Execution depends on **user actions** like:

- Click
- Key press
- Mouse movement

Event Example:

Event Handler

```
Private Sub Button1_Click(sender As Object, e As EventArgs)
    MsgBox("Button Clicked")
End Sub
```

Unit 5: Database Programming with ADO.NET

1. Introduction to ADO.NET

ADO.NET is used to connect and interact with databases.

2. Components of ADO.NET

- **Connection**: Establishes connection to database.

```
Dim con As New SqlConnection("connection_string")
```

- **Command**: Executes SQL queries.

```
Dim cmd As New SqlCommand("SELECT * FROM Student", con)
```

- **DataAdapter**: Transfers data between database and dataset.
- **DataSet**: Stores data in memory (disconnected mode).
- **DataGridView**: Displays data in tabular form.

3. Connected vs Disconnected Architecture

Connected	Disconnected
Requires constant connection	Connection not required
Faster for small data	Better for large data

4. Displaying Data in DataGridView

```
DataGridView1.DataSource = dataset.Tables(0)
```

Summary & Conclusion

VB.NET is:

- Simple
- Powerful
- Object-Oriented
- Event-Driven
- Secure
- Memory managed
- Database friendly
- Ideal for Windows application development

It is widely used for:

- Desktop applications
- Database systems
- Enterprise software

Integration:

It integrates smoothly with databases using **ADO.NET** and provides strong support for GUI design.

Premium Digital Student Notes - Deep Dive

VB.NET – DETAILED DEPTH NOTES (BCA)

UNIT 1: .NET Framework & VB.NET – INTERNAL UNDERSTANDING

1. Why .NET Was Introduced (Background)

Before .NET: Different languages had different runtimes; Code reuse was poor; Memory leaks common; Security weak; Applications were platform dependent.

Microsoft introduced .NET to: Provide a common runtime; Allow multiple languages on one platform; Improve security, memory, and performance; Enable language interoperability.

So, .NET is not a language, it is a platform + runtime + libraries.

2. .NET Framework Architecture (Deep Explanation)

.NET works in layered architecture, where each layer has a clear responsibility.

Layer 1: Application Layer - This is where VB.NET, C#, ASP.NET applications exist. Developers write code here.

Layer 2: Framework Class Library (FCL) - This layer provides ready-made classes so that programmers don't need to write everything from scratch. (Examples: System.IO, System.Data, System.Windows.Forms). FCL increases developer productivity and code reliability.

Layer 3: Common Language Runtime (CLR) - CLR is the execution engine of .NET.

1. CLR loads the program -> 2. Checks security -> 3. Manages memory -> 4. Converts code to machine language -> 5. Handles errors and garbage collection.

Note: Without CLR, .NET code cannot run.

3. Common Language Runtime (CLR) – INTERNAL WORKING

CLR provides runtime services.

a) Memory Management

- Allocates memory automatically
- Frees unused memory via Garbage Collector
- Prevents memory leaks

Programmer does not use malloc() or free() like C.

b) Garbage Collection

- Automatically destroys unused objects
- Improves application stability
- Runs in background

This is why VB.NET programs are less prone to crashes.

c) Exception Handling

CLR provides a structured error handling system.

- Runtime errors are handled safely
- Program does not crash abruptly

d) Security- CLR enforces:

- Code access security
- Type safety
- Role-based security

4. Common Type System (CTS) – DEEP VIEW

CTS defines: • What data types exist • How they behave • How they interact

Why CTS is needed? Because .NET supports multiple languages.

Example: Integer in VB.NET and int in C# internally both map to System.Int32.

CTS ensures language interoperability.

5. MSIL (Microsoft Intermediate Language)

VB.NET code is not directly converted into machine code.

Compilation Process:

1. VB.NET source code -> 2. Compiled into MSIL (Stored in assembly) -> 3. JIT compiler converts MSIL -> machine code

Advantages of MSIL:

- Platform independent
- Secure
- Optimized execution

6. Assemblies – COMPLETE EXPLANATION

Assembly is the fundamental building block of .NET applications.

Assembly Contains:

- Metadata (info about classes, methods)
- Security information
- Resources (images, icons)
- MSIL code
- Resources (images, icons)

Types:

1. Private Assembly: Used by one application; Stored in application folder.
2. Shared Assembly: Used by multiple applications; Stored in Global Assembly Cache (GAC).

Assemblies help in: Version control, Deployment, Security.

UNIT 2: VB.NET LANGUAGE – IN DEPTH

1. Variables – CONCEPTUAL VIEW

Variable is not just a name, it is: A memory reference + With a data type + Having scope and lifetime.

Why Variables are Needed? Programs work on data, and variables allow programs to store, process, and manipulate data.

Declaration:

2. Data Types – INTERNAL SIGNIFICANCE

Data types define: Memory size,Range of values, Operations allowed.

Integer -> arithmetic operations. Boolean -> logical decisions.

Using correct data type: Improves performance, Reduces memory usage, Avoids runtime errors.

3. Scope and Lifetime – PRACTICAL IMPORTANCE

Scope: Defines where a variable can be accessed.

Lifetime: Defines how long it exists in memory.

- **Local variables:** Created when function starts, Destroyed when function ends.
- **Class-level variables:** Exist as long as object exists.

Premium Digital Student Notes – Deep Dive

6. Control Structures – LOGIC BUILDING

Control structures define flow of execution.

Conditional Statements: Used for decision making.

If-Else allows program to:

- Compare values
- Take different actions.

Looping Statements: Used for repetition.

Loops reduce:

- Code size
- Human error
- Development time.

7. Functions and Subroutines – DESIGN CONCEPT

Subroutine: Performs task,

- Does not return value.

Function: Performs computation, Returns value.

Why use them?

- Code reuse
- Modularity
- Easy debugging.

4. Constants – WHY AND HOW

Constants represent fixed values.

Why use constants?

- Prevent accidental changes
- Improve readability
- Easy maintenance.

```
Const TAX_RATE As Double = 0.18
```

5. Arrays – REAL UNDERSTANDING

Array stores multiple related values under one name.

Why arrays?

- Avoid multiple variables,
- Enable indexed access
- Efficient data storage.

```
marks(0) = 90
```

Arrays are widely used in:

- Student records
- Data processing
- Database results.

UNIT 3 & 4: FORMS & EVENTS – INTERNAL MECHANISM

1. Windows Forms Architecture

Windows Forms follow: **Event Driven Model + Object Oriented Design.**

Each control: Is an object; Has properties, methods, events.

2. Controls – DEEP VIEW

- **TextBox:** Used for input. Internally stores data as string.
- **Button:** Triggers an event. Acts as action initiator.
- **ComboBox:** Combines TextBox + ListBox (Allows selection + typing).
- **PictureBox:** Handles Image rendering, Scaling, Memory optimization.

UNIT 4: EVENT DRIVEN PROGRAMMING

What is Event Driven Programming?

Program does not run line by line. It reacts to **user actions**.

Click, Key press, Mouse move.

Event Handling Mechanism:

1. User performs action
2. Event is raised
3. Event handler executes

This makes GUI applications **interactive and responsive.**

UNIT 5: ADO.NET – COMPLETE DEPTH

1. Why ADO.NET?

Traditional database systems: Required constant connection; Slow for large applications.
ADO.NET provides: Disconnected architecture; High performance; Scalability.

2. ADO.NET Architecture

Connection Object: Manages connection lifecycle.
Command Object: Executes SQL queries.
DataAdapter: Acts as bridge between Database and DataSet.
DataSet: Stores data in memory. Works even when database is disconnected.
DataGridView: Displays data visually.

3. Connected vs Disconnected Model

Connected: Fast, Resource heavy.
Disconnected: Scalable, Network efficient.