VISUAL BASIC UNIT I

TOPIC OUTLINE

1. What is Visual Basic?
2. History and Definition of Visual Basic
3. Why is it called Visual Basic?
4. Why we use visual basic
5. What does Visual Basic do?
6. Is there a need to study Visual Basic?
7. User Interface (UI)
8. Comparison of visual basic with competing visual programming environment
9. Limitations of Visual Basic

1. What is Visual Basic?

High level programming language developed by Microsoft that makes it easy to create applications using a simple, English-like syntax, especially windows desktop apps. An event-driven programming language and environment that allows you to build software with a graphical user interface by dragging / dropping components like buttons, textboxes and labels.

Levels of Programming Language

Programming languages are grouped into levels based on how close they are to machine / how close they are to human language:

a) Low-level (Hard for humans):

- Machine language: 0101.
- Assembly language: uses mnemonics.

b) Middle-level: Used for system and app programming.

- C
- C++

c) High-level: Close to human languages, easy to learn, used for most apps today (e.g., Python, Java, C#, VISUAL BASIC, PHP, Javascript)

History of Visual Basic

The Pre-Visual Basic Era

Before the advent of the modern graphical environments we use today, Microsoft utilized early programming languages such as Quick BASIC and BASICA, which were primarily designed for the MS-DOS operating system. During this era, Quick BASIC functioned strictly as a text-only programming language. Developers faced significant hurdles because there were no drag-and-drop functionalities; every element of a program, from the logic to the layout, had to be typed

manually. While powerful for its time, this manual approach was far from ideal for building complex applications for the emerging Windows environment.

The Birth of "Visual" Programming

The transition toward a more intuitive system began when Cooper Software developed a revolutionary rapid UI (User Interface) design tool named Tripod. The primary innovation of Tripod was that it allowed users to create and arrange interfaces visually rather than through lines of text code. Recognizing the potential of this technology, Microsoft combined the visual design capabilities of Tripod with the established logic of Quick Basic to create the first version of Visual Basic. This merger allowed programmers to design the "look" of an application with a mouse while using familiar BASIC commands for the "logic."

Evolution and the Shift to .NET

Following its successful launch, Microsoft continued to refine the language, introducing several iterative versions including **1.0, 2, 3, 4, 5, and 6.0**. While these versions were widely popular, the architecture eventually reached its limits. To address modern software needs, Microsoft developed **VB.NET**, which serves as the official **successor to VB6**. Unlike previous updates, VB.NET was not just a simple upgrade; it was **completely rewritten** from the ground up to be a modern, **object-oriented** programming language. Built on the **.NET Framework**, it allows developers to create a wide variety of sophisticated software, including web, mobile, and desktop applications, using contemporary programming standards.

Why is it called Visual Basic?

Visual means: You create programs visually in a drag and drop environment. You create windows (forms) visually by dragging buttons, textboxes, and labels onto a form and positioning them with your mouse. The system automatically generates the backend code for the interface.

BASIC means: Beginner's All-purpose Symbolic Instruction Code:

- Beginner's: Simple rules, easy-to-read commands, simple English-like words.
- All-purpose: Can be used for many types of programming tasks.

- Symbolic: Use of symbols and words to represent operations instead of machine code.
- Instruction Code: Simple instructions that the computer can follow step by step.

Why we Use Visual Basic?

a) Easy to Learn and Use

Visual Basic is widely recognized for being easy to learn and use, making it an ideal starting point for new programmers. This accessibility stems from its use of English-like syntax, which makes the code highly readable and intuitive. For example, instead of using complex symbols, VB uses clear commands like PRINT to display text, IF...THEN...ELSE for making decisions, and INPUT to capture data from the user. Beyond the language itself, the environment allows for drag/drop visual design through its Integrated Development Environment (IDE).

Example: Instead of writing dozens of lines of code to define the size, color, and position of a button, a developer can simply drag a "Command Button" from the Toolbox and drop it onto a Form. They can then use the Properties Window to change its "Caption" to "Submit" without writing a single line of code.

b) Event-Driven Programming

Unlike traditional programs that follow a strict top-to-bottom sequence, Visual Basic is Event-Driven. This means that programs respond to user actions (events) as they happen in real-time. The program sits in a "waiting" state until the user triggers an event, such as clicking a button, moving a mouse, or typing on the keyboard. Each object on the screen can have its own specific "Event Procedure" that contains the code to be executed when that action occurs.

Example: Mouse Move (triggered when the cursor passes over an object) and KeyPress (triggered when a user hits a key while a textbox is active).

c) Rapid Application Development (RAD)

Visual Basic is a premier tool for Rapid Application Development (RAD), a methodology where developers can quickly design, test, and deploy programs. Because much of the user interface is created visually and many common tasks are automated, the time between a "concept" and a "working prototype" is significantly reduced. This is particularly valuable for businesses that need custom software solutions in a short timeframe to respond to market changes.

Example: In a RAD environment like Visual Basic, a developer can build a fully functional "Inventory Management" prototype in a single afternoon. By using pre-built components for menus, dialog boxes, and file handling, they can focus on the business logic rather than the underlying Windows architecture.

d) Integration with Windows and Databases

One of the core strengths of Visual Basic is its deep integration with Windows. It works well with Windows controls (like scroll bars, checkboxes, and list views) to ensure that applications look and feel like native parts of the operating system. Furthermore, it has built-in support for database connectivity, particularly with SQL Server using technologies like ADO.NET. This allows applications to seamlessly store, retrieve, and manage large amounts of corporate data.

Example: A developer can use a Sql Connection object to link their app to a SQL Server database. With just a few lines of code and a Sql Data Adapter, they can pull thousands of customer records into a Data Grid View control, allowing the user to sort and edit the data directly within the Windows interface.

e) Reusable Code and Components.

To improve efficiency and reduce errors, Visual Basic encourages the use of reusable code and components. You can create functions and modules to be used across multiple projects, which prevents you from having to "reinvent the wheel" every time you start a new application. By organizing code into logical units, you make your programs more maintainable and scalable, meaning they are easier to fix and grow over time.

Example: If you write a complex function that calculates "Sales Tax" based on different regions, you can save that code in a Standard Module (.bas or .vb file). Whenever you build a new billing or point-of-sale app, you simply "Import" that module into the new project. This ensures the tax calculation is identical and accurate across every piece of software your company owns.

What does Visual Basic do?

a) Creates Graphical User Interface (GUI) Applications

Visual Basic is primarily used to build Graphical User Interface (GUI) applications. In the past, users had to interact with computers by typing specific text commands into a dark screen (Command Line). VB changed this by allowing developers to create programs that users can interact with visually. This is done through a "Drag and Drop" interface where you place elements like windows, menus, icons, and buttons onto a canvas.

Details: The "Form" acts as the main container for your app. By using the Toolbox, you can place "Controls" (like checkboxes or radio buttons) that make the software intuitive for non-technical users.

b) Responds to User Events

Unlike older programming styles that run in a straight line from start to finish, VB responds to User Events. This means the program is "active" but does nothing until an event is triggered by the user or the system. Common events include clicking a button, opening a form (Load event), or even resizing a window.

Details: Each object has a set of "Event Handlers." For example, the Form_Load event can be used to set up the program's initial settings as soon as it opens, while a Command1_Click event contains the specific code that runs only when that button is pressed.

c) Connects to Databases

A major strength of Visual Basic is its ability to connect to databases. VB can store, retrieve, and manipulate data in external systems like SQL Server, Microsoft Access, or MySQL. This

capability is essential for building business applications, such as inventory systems (to track stock levels) or school payroll systems (to manage teacher salaries and student records).

Details: Using technologies like ADO (ActiveX Data Objects) or ADO.NET, VB acts as a bridge. It sends "Queries" to the database to ask for information and then displays that information in a format the user can understand, such as a table or a report.

d) Performs Calculations and Logic

At its core, Visual Basic is a powerful engine that **performs complex calculations and logic**. It processes data based on mathematical formulas and "If-Then" decision-making rules. This allows the program to take raw input (like hours worked) and turn it into useful output (like a calculated paycheck after taxes).

Details: VB uses variables to hold data in memory and uses logical operators (like AND, OR, NOT) to decide which path the program should take. For instance, it can check if a student's grade is above 50; if "True," it executes the code to display "Pass."

e) Develops Standalone Apps

Visual Basic is used to develop standalone apps—these are programs that are compiled into an executable file (an .exe file). These applications run on Windows computers without needing a web browser or an internet connection. This makes them faster and more reliable for desktop tasks compared to web-based software.

Details: Once you "Build" or "Compile" your project, it becomes an independent package. You can copy this file to another Windows computer, and it will run as a native application, utilizing the computer's CPU and RAM directly for high performance.

Is there a need to study Visual Basic?

a) Active Use in Enterprise Desktop Applications

It is specifically used for creating CRUD (Create, Read, Update, Delete) applications. These are tools used by employees to enter data into a database, generate shipping labels, or manage inventory. Since it integrates perfectly with Windows security (Active Directory), it is often the fastest way for a company to build a secure, private tool for its staff.

b) Usefulness in Office Automation (VBA)

Visual Basic for Applications (VBA) is a specialized version of the language built directly into Microsoft Office. It is the "secret engine" behind advanced **Excel, Word, and Outlook** automation. Businesses use it to turn a simple spreadsheet into a powerful automated system.

- **Calculations & Data Cleaning:** Instead of manually formatting thousands of rows of data, a VBA "Macro" can clean the data, remove duplicates, and perform complex math in seconds.
- **Dashboards:** VBA allows you to create interactive dashboards in Excel with buttons and sliders that update charts instantly, providing real-time business insights.
- **Workflow Automation:** In Outlook, VBA can be used to automatically sort incoming emails, save attachments to specific folders, or send automated replies based on the email's content.

c) Foundation for Event-Driven Programming

Studying Visual Basic provides a clear mental map for **Event-Driven programming**, a concept used in almost every modern language like **C#, Java, and Python**. In sequential programming, the computer follows a list of instructions from top to bottom. In Event-Driven programming, the code stays "asleep" until a specific action (an event) wakes it up.

Details: By learning how a button click in VB triggers a specific sub-procedure, you are learning the same logic used in JavaScript for web buttons, Java Swing for desktop apps, or Python’s Tkinter library.

Transferable Skills: Once you understand the "Trigger → Listener → Action" flow in the simple environment of Visual Basic, transitioning to more complex languages like C# (which is also

part of the .NET family) becomes much easier because the underlying architectural ideas are identical.

Limitations of Visual Basic

a) Limited Platform Support

Visual Basic was developed by Microsoft as a tool specifically for the Windows ecosystem. Consequently, it has limited platform support, meaning it is designed mainly for Windows apps. Unlike modern languages that are "cross-platform," software written in VB cannot build native apps for Mac, Linux, or mobile platforms like Android and iOS.

VB relies heavily on specific Windows libraries (DLLs) and the Windows Registry, the code does not "translate" to other operating systems. If a company wants to reach users on iPhones or Mac Books, they generally have to rewrite the entire application in a different language.

b) Suitability for Modern App Types

While Visual Basic is excellent for traditional desktop software, it is not great for modern app types. The current technology landscape has shifted toward web apps, mobile apps, and cloud-based services, and VB was not originally built to handle these architectures efficiently.

Modern applications often need to run in a web browser or scale across global cloud servers (like AWS or Azure). VB’s structure is "monolithic," meaning it’s designed to run as a single block of code on one local computer. While VB.NET has some web capabilities via ASP.NET, most developers prefer languages like JavaScript or Python for these modern tasks.

c) Declining Popularity

Visual Basic is experiencing declining popularity when compared to modern powerhouses like C#, Python, or Java. In the 1990s and early 2000s, VB was one of the most used languages in the world, but today, most new projects and professional developers have migrated to other languages.

Details: Most universities and bootcamps now teach Python for data science, Java for enterprise systems, or C# for game development (Unity). As the community of VB developers shrinks, it becomes harder to find updated tutorials, third-party libraries, and job opportunities specifically for Visual Basic.

d) Weak OOP (Specifically in VB6)

Older versions of the language, particularly VB6, are often criticized for having weak Object-Oriented Programming (OOP) support. It does not fully support essential features like Inheritance or Polymorphism, which are the building blocks of modern, complex software.

- **Inheritance:** The ability for one class to "inherit" traits from another (e.g., a "Car" class inheriting from a "Vehicle" class). VB6 does not allow this.
- **Polymorphism:** The ability for different objects to be treated as instances of a common parent class.
- **Details:** Without these features, programmers often have to write "messy" code or repeat themselves (breaking the "DRY" - Don't Repeat Yourself principle), making large-scale projects difficult to manage. (Note: VB.NET fixed most of these issues, but the reputation remains a limitation of the legacy language)

e) Slower Performance

Visual Basic generally offers slower performance compared to modern languages like C++ or even optimized versions of C#. It is not optimized for heavy processing tasks, such as 3D rendering, complex scientific simulations, or high-frequency data analysis.

Details: VB is designed for "ease of use" and "safety" rather than raw speed. It uses an abstraction layer that sits between the code and the computer hardware. While this makes it harder for a programmer to crash the whole system, it adds "overhead" that slows down the execution of the program. For a simple office tool, this doesn't matter, but for a high-performance game or a data-heavy application, the lag becomes noticeable.

User Interface (UI)

UI is the interaction point between the user and the computer. It is everything you see, touch, or click while using an app.

Types of UI:

1. GUI (Graphical User Interface)

The **GUI** is the most common type of interface today. It allows users to interact with the computer using visual elements like windows, buttons, icons, and menus. Instead of memorizing complex commands, you simply look for a picture (icon) or a label and click on it.

- Example: Microsoft Windows or macOS. When you want to open a file, you click on a folder icon with your mouse.
- Details: GUIs are designed to be intuitive. They use the "WIMP" model (Windows, Icons, Menus, Pointers). Because they are visual, they are very easy for beginners to learn.

2. CMD (Command Line Interface)

The **CMD** (or CLI) is a text-based interface where the user interacts with the computer by typing commands into a terminal or console. There are no pictures or buttons; the computer only responds to specific text instructions.

- **Example:** The Windows Command Prompt or the Terminal in Linux. To see all the files in a folder, a user types dir (in Windows) or ls (in Linux) and presses Enter.
- **Details:** While CMD is harder for beginners because you must memorize commands, it is preferred by advanced users and system administrators. It is much faster for performing repetitive tasks or managing servers.

3. Menu-Driven Interface

A Menu-driven interface provides the user with a specific list of options to choose from. The user navigates through these lists to find the function they need. This limits the user's choices, which makes the system very simple and prevents them from making mistakes.

- Example: An ATM (Automated Teller Machine) or a microwave's digital panel. When you use an ATM, you are presented with a list: "Withdraw," "Balance Inquiry," or "Transfer." You select one to move to the next set of options.
- Details: These are often used in public kiosks or specialized devices where a full keyboard isn't necessary. They guide the user through a specific process step-by-step.

4. VUI (Voice User Interface)

A **VUI** allows users to interact with a device through spoken commands. The system uses artificial intelligence and speech recognition to understand what the user says and then performs the requested action.

- Example: Apple’s Siri, Amazon’s Alexa, or Google Assistant. You can say, "Set an alarm for 7:00 AM," and the device processes your voice to set the timer.
- Details: VUI is excellent for accessibility and "hands-free" situations, such as when you are driving or cooking. It is becoming increasingly popular in "Smart Home" devices.

5. Touch User Interface

The Touch User Interface allows users to interact with a device by directly touching the screen with their fingers or a stylus. It replaces the mouse and keyboard with "gestures" like tapping, swiping, and pinching.

- **Example:** Smartphones (iPhone/Android), Tablets, and touch-screen laptops. You "swipe" your finger to scroll through social media or "pinch" two fingers together to zoom out of a map.
- **Details:** Touch interfaces are highly interactive and feel more "natural" than using a mouse. They support "multi-touch," meaning the screen can detect more than one finger at a time, allowing for complex gestures.

How to create a Simple User Interface

Creating a simple User Interface (UI) in Visual Basic is a visual process that relies on a "drag-and-drop" environment rather than manual coding for the layout.

Step 1: Use the Visual Design Window (The Form) Visual Basic provides a visual environment where you create windows (called "Forms") that act as the canvas for your application.

Step 2: Select Components from the Toolbox You choose the graphical elements you want to include in your interface, such as buttons, textboxes, and labels.

Step 3: Drag and Drop Instead of typing code to define the size or location of an element, you simply drag these components from the toolbox and drop them onto the form.

Step 4: Position and Layout with the Mouse You use your mouse to move the buttons or textboxes into the desired positions on the window. This "Rapid Application Development" (RAD) approach allows you to see exactly what the user will see while you are designing it.

Step 5: Let the System Generate Backend Code As you design the interface visually, Visual Basic automatically generates the backend code required to make that interface exist.

Step 6: Add Functionality via Events to make the UI "do something," you add code to specific events. For example, you can write a small instruction that tells the program what to do when a user clicks a button or types in a textbox.

Comparison of visual basic with competing visual programming environment

Overview of Visual Programming Environments

A visual programming environment allows you to build software by interacting with graphical elements (like dragging buttons onto a form) rather than writing every line of layout code by hand. While Visual Basic was a pioneer in this field, other languages have developed their own "Visual" tools.

1. Visual Basic vs. Visual C++

Strengths of Visual Basic Visual Basic is designed for **Rapid Application Development (RAD)**. Its primary strength is the speed with which a developer can go from an idea to a working Windows application. The "Drag and Drop" interface is much more intuitive, and the syntax uses simple English words, making it much easier for beginners to learn compared to the complex symbols used in C++.

Weaknesses and Performance Comparison the main weakness of Visual Basic compared to C++ is **execution speed**. C++ is far more powerful and is the industry standard for high-performance software like game engines (e.g., Unreal Engine) and operating systems.

- **Why is Visual Basic slower?**
    1. **High-Level Abstraction:** VB is a "high-level" language, meaning it is designed to be easy for humans to read. It hides complex operations from the programmer. Every time you run a command in VB, the computer has to do extra "translation" work behind the scenes to understand it.
    2. **Lack of Hardware Control:** C++ allows "low-level" access, meaning the programmer can talk directly to the computer's memory (RAM) and processor (CPU). VB prohibits this direct access to prevent beginners from accidentally crashing the whole system. This "safety barrier" adds a layer of software called a **runtime environment** that sits between your code and the hardware, slowing down the final speed.
    3. **Compilation:** C++ compiles directly into **native machine code** (binary), which the CPU can execute instantly. VB code often requires an extra step of translation by the "Common Language Runtime" while the program is running.

Conclusion: Use **Visual Basic** for building internal business tools and simple desktop apps quickly. Use **C++** when you need maximum speed, such as for 3D graphics or heavy data processing.

2. Visual Basic vs. Java (NetBeans/Swing)

Strengths of Visual Basic When it comes to the **Windows operating system**, VB is often "smoother." Because it is built by Microsoft, it integrates perfectly with Windows buttons, menus, and security. In VB, creating a user interface requires much less code; for example, "Event Handling" (making a button do something when clicked) is often a single step, whereas, in Java, you might have to write multiple lines to "listen" for that click.

Weaknesses of Java Comparison The biggest weakness of VB here is **Platform Support**. VB is almost entirely locked to Windows. Java's famous motto is "Write Once, Run Anywhere." A visual app built in Java can run on a Mac or a Linux computer without changing the code.

Conclusion: Visual Basic is the winner for specialized Windows desktop development. **Java** is the winner for enterprise-level applications that need to work across different types of computers or on the web.

3. Visual Basic vs. C# (Visual Studio)

Strengths of Visual Basic VB is often considered "friendlier" than C#. It is not case-sensitive (meaning MyVariable and myvariable are the same), which prevents many common bugs for students. It also includes the **"My" namespace**, a shortcut that allows you to perform complex tasks (like checking if a file exists) with a single line of code that would take several lines in C#.

Weaknesses of C# Comparison C# is more **modern and popular**. Most professional developers prefer C# because its syntax is similar to Java and C++, making it easier to switch between languages. C# also has a much larger community, meaning it is easier to find help, tutorials, and pre-written libraries online.

Conclusion: Visual Basic is excellent for people coming from an office/Excel background or for very quick "one-off" tools. **C#** is the better choice for someone looking to start a long-term career in professional software engineering.

4. Visual Basic vs. Python

Strengths of Visual Basic Visual Basic's visual editor is much more powerful and "true" than Python's. While Python has libraries like Tkinter to build GUIs, they often require you to write

code to position every button. In VB, you see the exact window you are building 4as you drag elements around, which is much faster for UI design.

Weaknesses of Python Comparison Python is far more versatile. While VB is mostly for desktop apps, Python is the world leader in Artificial Intelligence, Data Science, and Web **Development**. Python is also cross-platform and has a much simpler syntax for handling complex data.

Conclusion: If your goal is to build a strictly visual Windows tool, Visual Basic is superior. If you want to work with Data Science or modern AI, Python is the better environment.