جامعة المستقبل
الكلية التقنية الهندسية
قسم هندسة تقنيات الاتصالات

Second Stage

Visual Basic - UOMU0207036

Lecture 5 - Build Modern Windows Application

Presented By:

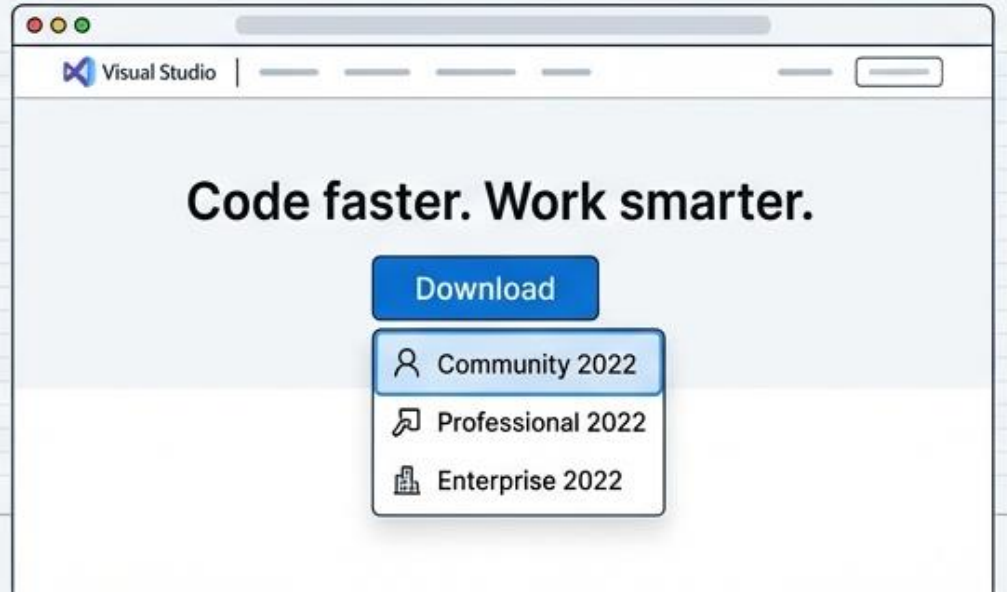
Dr. Mohammed Fadhil

Email: mohammed.fadhil1@uomus.edu.iq

Laying the Groundwork: Your Development Environment

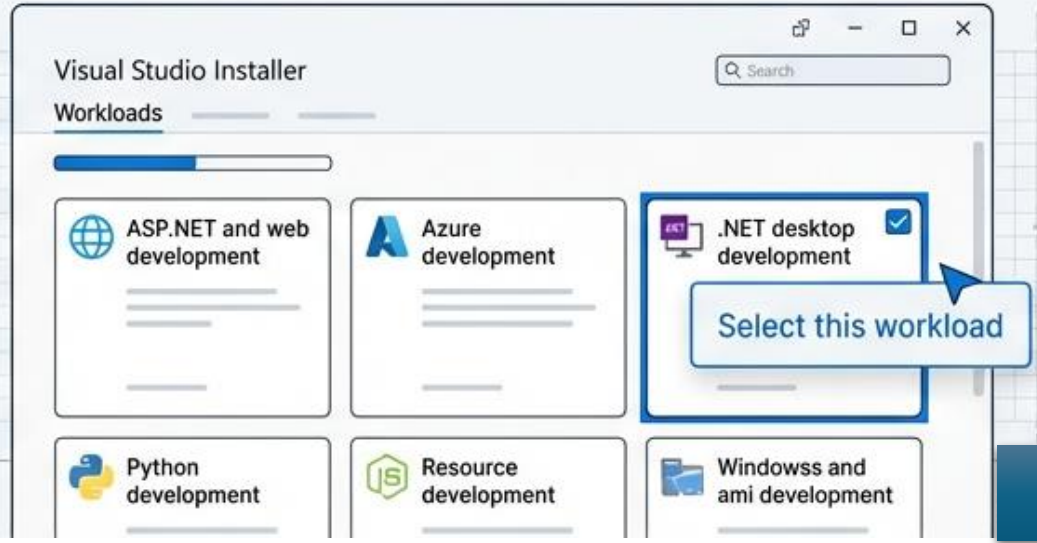
What is Visual Basic 2022?

- A modern, third-generation, event-driven programming language from Microsoft.
- Fully object-oriented, implemented on the .NET Framework.
- Part of the Visual Studio 2022 IDE, which also supports C#, C++, Python, and more.



Installing the Tools

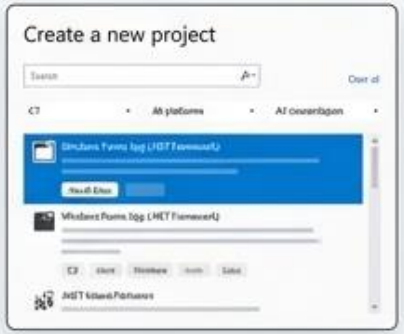
- Visual Basic 2022 is developed within Visual Studio 2022.
- The free "Community 2022" edition provides a full-featured IDE for students, open-source projects, and individuals.
- During installation, select the ".NET desktop development" workload to get all necessary components for building Windows Forms apps.



From Zero to 'Hello, World!': Creating Your First Application

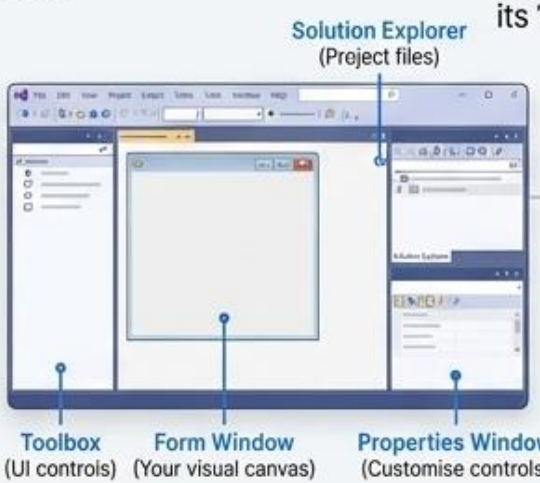
Create Project:

Start by selecting 'Create a new project' and choosing the 'Windows Forms App (.NET Framework)' template for Visual Basic.



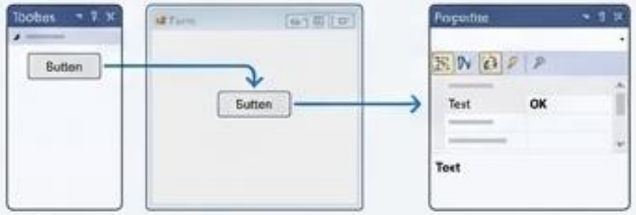
Explore the IDE:

The IDE appears, with four key components for building your app.



Add a Control:

Drag a Button from the Toolbox onto the Form. In the Properties window, change its 'Text' property to 'OK'.



Write the Code:

Double-click the button. In the `Button1_Click` event, add one line of code.

Write the Code: Double-click the button.

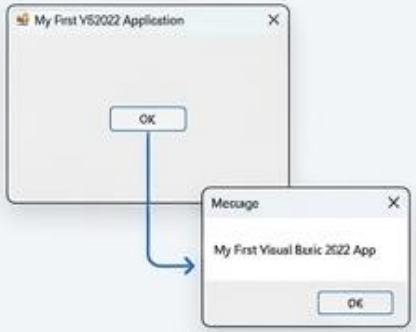
the `Button1_Click` event, add one line of code.

```
MsgBox("My First Visual Basic 2022 App")
```

`MsgBox()` is a built-in function to display a dialog box. This is event-driven programming: the code runs in response to a user's click.

Run and See the Result:

Press F5 to run. Clicking the button displays your message.



The Blueprint: Designing the User Interface

The **Form** is the canvas of your application. Its properties—and the properties of every control on it—can be set in two ways.

Design-Time Customisation

Use the **Properties Window** to change appearance and behaviour before running the app.

Example Properties for a Form:

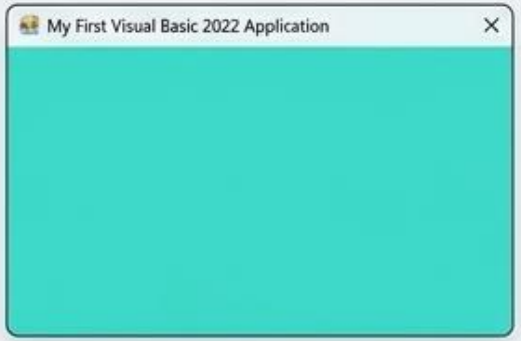
- **Name**: The identifier for the form in your code.
- **Text**: The title displayed in the window's title bar.
- **BackColor / ForeColor**: Sets the background and text colours.
- **MaximizeBox**: Controls whether the user can maximise the window.

Runtime Customisation

Modify properties programmatically using code. This allows for dynamic UIs that respond to user actions or data.

The **Me** keyword refers to the current form instance.

```
' This code runs when the form first loads
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    Me.Text = "My First Visual Basic 2822 Application"
    Me.BackColor = Color.Turquoise
    Me.MaximizeBox = False
End Sub
```



The Builder's Components: Working with Core Controls

Controls are the interactive objects placed on a form. We'll look at four fundamental types.

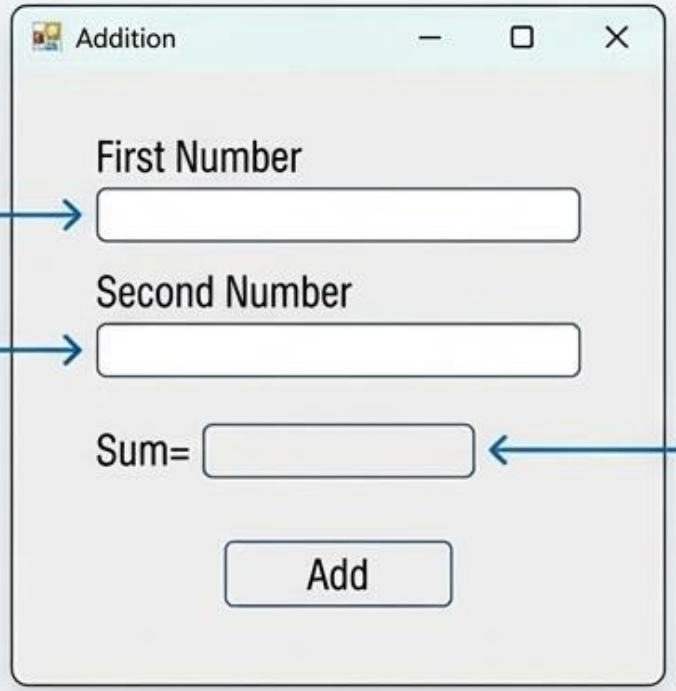
---

1. TextBox

Purpose: Accepts text or numeric input from the user.

Key Property: `TextBox1.Text` gets or sets the string value inside the box.

Key Function: `Val(TextBox1.Text)` converts the text to a numeric value for calculations.



2. Label

Purpose: Displays read-only text, such as instructions or output.

Key Property: `Label1.Text` sets the display text.

3. ListBox

Purpose: Displays a scrollable list of items for user selection.

Key Methods:

```
MyListBox.Items.Add("New Item"),
MyListBox.Items.Remove("Item to Remove"),
MyListBox.Items.Clear()
```

4. ComboBox

Purpose: A drop-down list that combines a text box with a list box. It's space-efficient.

Methods: Similar to ListBox (`.Items.Add`, `.Items.Remove`, etc.).

```
Dim num1 As Double
Dim num2 As Double
Dim result As Double

' Convert input text to numbers
num1 = Val(TextBox1.Text)
num2 = Val(TextBox2.Text)

' Calculate sum
result = num1 + num2

' Display result
TextBox3.Text = result.ToString()
```

The Raw Materials: Managing Data, Variables, and Constants

Visual Basic Data Types

VB classifies data into two main categories to optimise storage and processing.

Numeric Types

For mathematical calculations. Examples include:

- **Integer**: Whole numbers (e.g., -32,768 to 32,767).
- **Long**: Larger whole numbers.
- **Single**, **Double**: For high-precision floating-point numbers.
- **Decimal**: For currency and high-precision financial calculations.

Non-Numeric Types

- **String**: Text and characters (e.g., 'Hello World').
- **Date**: Stores date and time values.
- **Boolean**: Stores `True` or `False` values.

Storing Data

Variables

Named storage locations whose values can change. Declared with the `Dim` keyword.

```
' Syntax: Dim VariableName As DataType
Dim customerName As String
Dim quantity As Integer
Dim price As Decimal = 19.99
```

Constants

Named values that remain the same throughout the program. Declared with the `Const` keyword.

```
' Syntax: Const ConstantName As DataType = Value
Const Pi As Double = 3.14159
```

The Logic Engine: Making Decisions in Code

Use conditional logic to execute different blocks of code based on whether a condition is True or False.
This is powered by **Conditional Operators** (=, >, <, <>) and **Logical Operators** (**And**, **Or**).

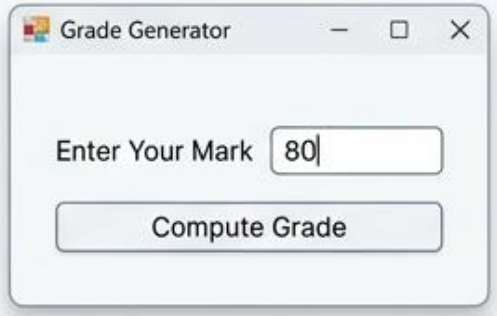
Structure 1: If...Then...ElseIf

Best for evaluating one or more different conditions.

```
Dim Mark As Integer = Val(TxtMark.Text)
Dim Grade As String

If Mark >= 80 Then
    Grade = "A"
ElseIf Mark >= 60 Then
    Grade = "B"
Else
    Grade = "D"
End If

MsgBox("Your grade is " & Grade)
```

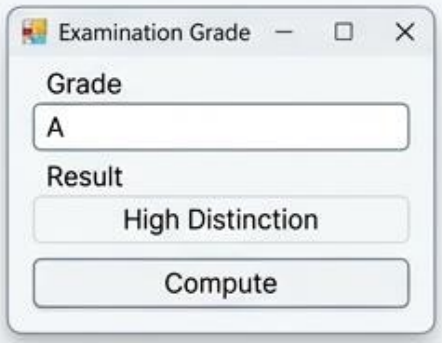


Structure 2: Select Case

Ideal when comparing a single expression against multiple possible values. It's often cleaner and more readable than a long If...Then...ElseIf chain.

```
Dim grade As String = txtGrade.Text

Select Case grade
    Case "A"
        LblResult.Text = "High Distinction"
    Case "B"
        LblResult.Text = "Credit"
    Case Else
        LblResult.Text = "Fail"
End Select
```



The Power of Repetition: Automating Tasks with Loops

Loops allow a block of code to be executed repeatedly until a certain **condition** is met.

Loop Type 1: For...Next Loop

Use Case: Repeats code a specific number of times. Perfect when you know the number of iterations in advance.

```
' This loop runs 10 times, with counter going from 1 to 10.
For counter As Integer = 1 To 10
    ListBox1.Items.Add(counter)
Next
```

Loop Type 2: Do...Loop

Use Case: Repeats code *while* a condition is true or *until* a condition becomes true. Ideal when the number of iterations is unknown.

```
Dim n As Integer = 0
Dim sum As Integer = 0

Do Until n = 10
    n += 1
    sum += n
Loop
' At the end, n=10 and sum=55
```

Crafting Reusable Tools: Functions and Sub Procedures

Instead of writing the same logic repeatedly, encapsulate it into a **procedure** or **function** that can be **called** whenever needed. This promotes clean, organised, and maintainable code (Don't Repeat Yourself - DRY principle).

Sub Procedures

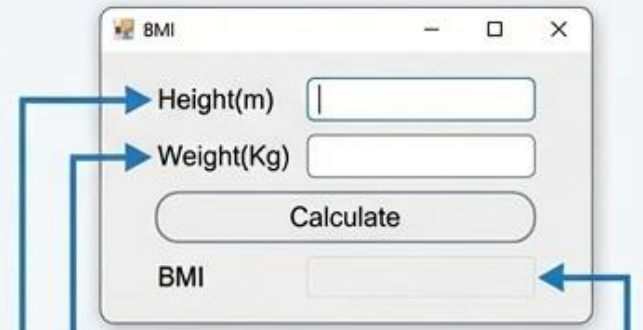
- Performs a specific task.
- Does **not** return a value.
- **Example:** A sub procedure to clear all text boxes on a form.

Functions

- Performs a task and **returns**.
- Performs a task and **returns a value**.
- Ideal for calculations where you need a result back.

Practical Example: The BMI Calculator Function

We can create a dedicated `BMI` function that takes height and weight as inputs and returns the calculated Body Mass Index. This logic can then be reused anywhere in the application.




```
Private Function BMI(Height As Single, weeight As Single) As Double
    BMI = weight / Height ^ 2
End Function
```

```
' In a button's click event:
Dim h As Single = Val(TxtHeight.Text)
Dim w As Single = Val(TxtWeight.Text)
LblBMI.Text = Format(BMI(h, w), "0.00")
```

BMI (Body Mass Index) Calculator

```
1 reference
Public Class Form1
    ' ===== BMI FUNCTION =====
    2 references
    Private Function BMI(Height As Single, Weight As Single) As Double
        BMI = Weight / Height ^ 2
    End Function
    ' ===== BUTTON CLICK EVENT =====
    0 references
    Private Sub BtnCalculate_Click(sender As Object, e As EventArgs) Handles Button1.Click
        ' Read values from textboxes
        Dim h As Single = Val(TextBox1.Text)
        Dim w As Single = Val(TextBox2.Text)

        ' Display BMI result
        TextBox3.Text = Format(BMI(h, w), "0.00")
    End Sub
End Class
```

جامعة المستقبل

Question And Answer Session

Thank You For Your Attention

هندسة تقنيات الاتصالات