المادة الدراسية : البرمجة المرئية

المرحلة الثالثة / قسم علوم الحاسوب

مدرس المادة :   د. شيماء احمد رزوقي

سنان عادل مولود

Third Class

Visual Programming

Visual Studio 2022

Syllabus

- Visual programming
- Visual programming vs. traditional programming
- Examples of visual programming languages
- Applications of Visual Programming language
- Install Visual Studio
- Introduction to Visual Studio
- Visual Studio 2022
- Create a New Project in Visual Studio Community
- Visual Basic .NET Forms
- Adding Controls Using the Toolbox
- Designing the User Interface (UI)
- VB .NET Properties
- Adding a Splash of Color to a VB Form
- Adding a Font effects to a VB Form
- Writing the Code in VB Project
- The Concept of Event-Driven Programming
- Data Types
- Dealing with Variables and Constants
- Scope of Declaration
- Mathematical Operations
- The Message Box In VB .Net
- Working with PictureBox
- Conditional Logic - If Statements
- Add a Combo Box to a VB .NET form
- Check Boxes and Radio Buttons in VB .NET
- An Introduction to Loops, and Array
- Adding Menus to a Visual Basic .NET Form
- Text Files in VB.Net
- An Introduction to Functions and Subs
- Standard Modules in VB .NET
- VB.NET Classes and Objects
- Database Application in VB.Net
- Steps How to Connect Access Database in VB.Net
- .NET Framework data providers
- VB.Net 2022 Connects to SQL Server Database

Visual programming

Visual programming is a type of programming language that lets humans describe processes using illustration. Whereas a typical text-based programming language makes the programmer think like a computer, a visual programming language lets the programmer describe the process in terms that make sense to humans.

What is visual programming?

Visual programming offers a code-free way to create software applications using graphics to represent components and their interactions. This intuitive approach enables developers to build complex programs effortlessly and quickly, often using flowcharts, diagrams, or other visual representations to depict the structure and behavior of the software.

- A visual programming language (VPL) is a programming language that uses graphical elements and figures to develop a program.
- A VPL employs techniques to design a software program in two or more dimensions, and includes graphical elements, text, symbols and icons within its programming context.
- A visual programming language is also known as an executable graphics language.

The reasons for of implementing Visual Basic program are listed as follows:

- ❖ It uses Integrated Development Environment (IDE) which is easier for the user to minimize code writing.
- ❖ All visual programs follow the same concepts, therefore the user will become more familiar with visual approach for other visual languages.
- ❖ It provides Input box and Output box as an interactive windows with user.
- ❖ It is able to connect to Internet, and to call Explorer.

Widely applicable in web and mobile app development, visual programming offers numerous benefits, including:

- Usability and accessibility: Visual programming environments typically have intuitive and user-friendly interfaces.
- Rapid development and prototyping: Because visual programming enables rapid prototype and iteration, users can quickly assemble and test components.

It helps developers understand the impact of their development changes by providing real-time feedback and visualizing program execution.

- Improved debugging: Visual programming language debuggers highlight errors directly in the visual representation, making it easier to identify and resolve issues.

Visual programming vs. traditional programming

Visual development and traditional development are two different approaches to creating software. Traditional development typically involves writing code in a text-based programming language, like as C++, Java, or Python. This code-based app development approach requires the developer to have a strong understanding of the language and its syntax, as well as the concepts of programming, such as data structures and algorithms.

Visual development, on the other hand, uses visual elements, such as diagrams and flowcharts, to represent programming constructs. **Visual development environments** typically have drag-and-drop interfaces.

Depending on the visual development tool and who it targets, it allows professional developers to accelerate their productivity, or users with no programming background to create software.

Overall, the main difference is that traditional development focuses on the actual coding and writing of the code, while visual development focuses on the design and overall structure of the software.

Examples of visual programming languages

Some visual programming examples include:

- Scratch: Primarily used in education, This language enables users, particularly children, to build interactive stories, games, and animations by snapping code blocks together.
- LabVIEW: This National Instruments system design platform and development environment is commonly used in engineering, data acquisition, and industrial automation applications.

- Visual Basic is a programming language. Microsoft developed this object-oriented programming language in the 1990s. With Visual Basic, developers can create graphical user interfaces (GUIs) and applications for Windows.

Visual Basic comprises windows. It also consists of a toolbox which contains many useful controls that allow a programmer to develop his or her VB programs.

- ❖ Form Designer: it is a window for each form to customize the designed interface of the application. Using the form designer, the user can add controls, graphics, and text to create the desired form appearance.
- ❖ Properties Window: it is a List of properties settings for a selected form or a control. These properties are characteristics (such as size, visible, or color) of the selected object it provides an easy way to set properties.
- ❖ Tool-Box: it contains a collection of tools that are needed for project design.

What is visual coding?

In visual coding, programmers create software applications using visual elements such as diagrams, flowcharts, and symbols instead of traditional lines of code. In the same way as visual programming, a programmer selects and arranges prebuilt blocks of code or logic elements to create a program.

Applications of Visual Programming language:

VPL can be used in multiple domains like multimedia, educational purpose, video games, automation:

- **Multimedia:-** VPL helps users create multimedia without worrying about the real code or other complex features. It narrows down to specific functions and with the help of those functions, multimedia is created.
- **Educational Purpose:-**Scratch VPL, etc are used to help students in their projects and make them familiar with the coding.
- **Video Games:-**VPL helps to create the videogames without writing lines of codes Ex-Scratch VPL is used to make videogames.

Advantages of visual programming language:

- Easy to convert ideas into reality for example you don’t know how to code so you can start with VPL(Visual Programming Language). and then switch to actual coding.
- Visuals are easy to grasp i.e. to develop something in visual programming language requires less efforts.
- It includes a variety of built in objects that are required while creating something using VPL.
- It is a beginner-friendly also anyone will be able to derive the logic without worrying about writing lines of code.
- Adding a user-specific code is also available and simple as it allows to create of blocks as per the convenience of the user.

Disadvantages of visual programming language:

- These languages require more memory as they use graphics, as a result, their execution is also slow and a large amount of memory is occupied by them.
- They can only work in an operating system like windows, mac, or any other operating system which supports graphics.
- As the inbuilt functions are not sufficient so you have to add your custom code as a result it is cumbersome.
- Only limited functions are present in these languages.
- Adding our custom code as a block requires coding knowledge or else you have to work with limited functions which are provided with the language.
- As a computer engineer, it is not a good idea to use VPL as most of the tech giants like FAANG or other tech companies work on textual languages like JAVA, HTML, etc, rather than VPL.
- For the long run VPL might not be that much useful as in a regular language you can explore more in it but in VPL at one point you will get bored by using the same it.

Install Visual Studio

The software you need is free from Microsoft and is called Visual Studio Community.

Make sure your computer is ready for Visual Studio

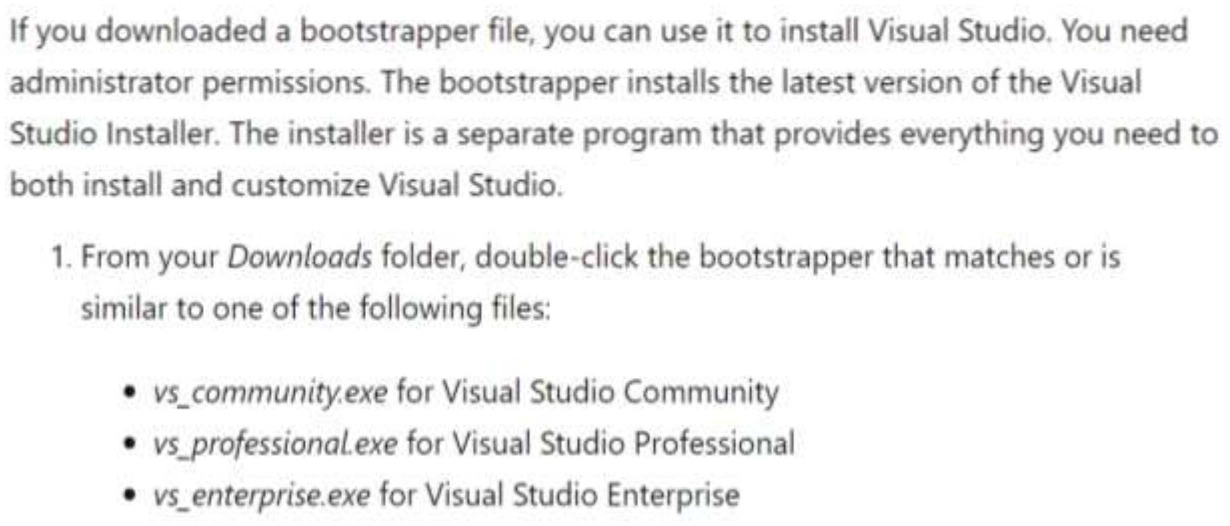
Before you begin installing Visual Studio:

- Check the system requirements. These requirements help you know whether your computer supports Visual Studio 2019.
- Make sure that the user who performs the initial installation has administrator permissions on the machine.
- Apply the latest Windows updates. These updates ensure that your computer has both the latest security updates and the required system components for Visual Studio.
- Restart. Restarting ensures that any pending installs or updates don't hinder your Visual Studio install.
- Free up space. Remove unneeded files and applications from your system drive by, for example, running the Disk Cleanup app.

Determine which version and edition of Visual Studio to install

Decide which version and edition of Visual Studio to install. The latest release of Visual Studio 2019--2022 that is hosted on Microsoft servers.

Initiate the installation

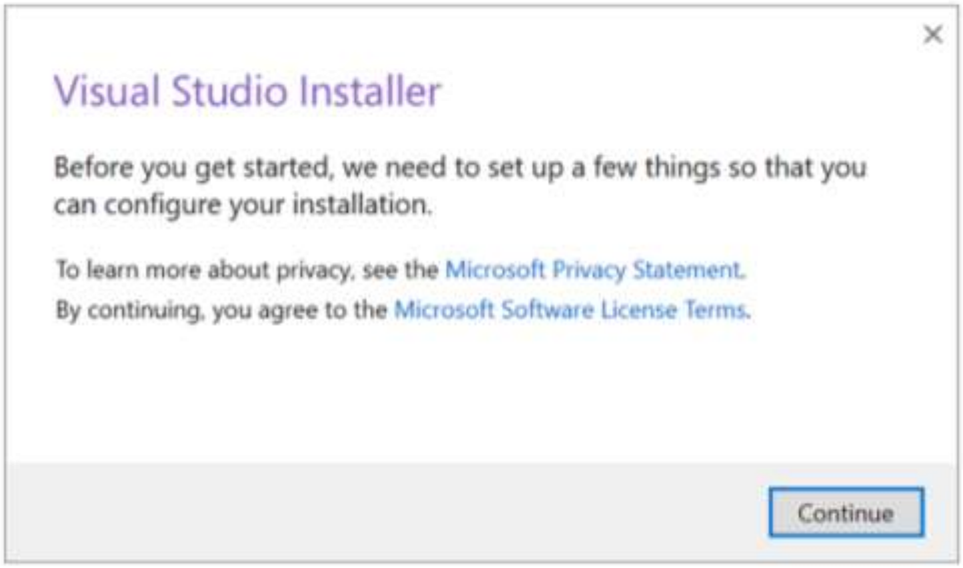


If you downloaded a bootstrapper file, you can use it to install Visual Studio. You need administrator permissions. The bootstrapper installs the latest version of the Visual Studio Installer. The installer is a separate program that provides everything you need to both install and customize Visual Studio.

1. From your *Downloads* folder, double-click the bootstrapper that matches or is similar to one of the following files:

   - *vs_community.exe* for Visual Studio Community
   - *vs_professional.exe* for Visual Studio Professional
   - *vs_enterprise.exe* for Visual Studio Enterprise

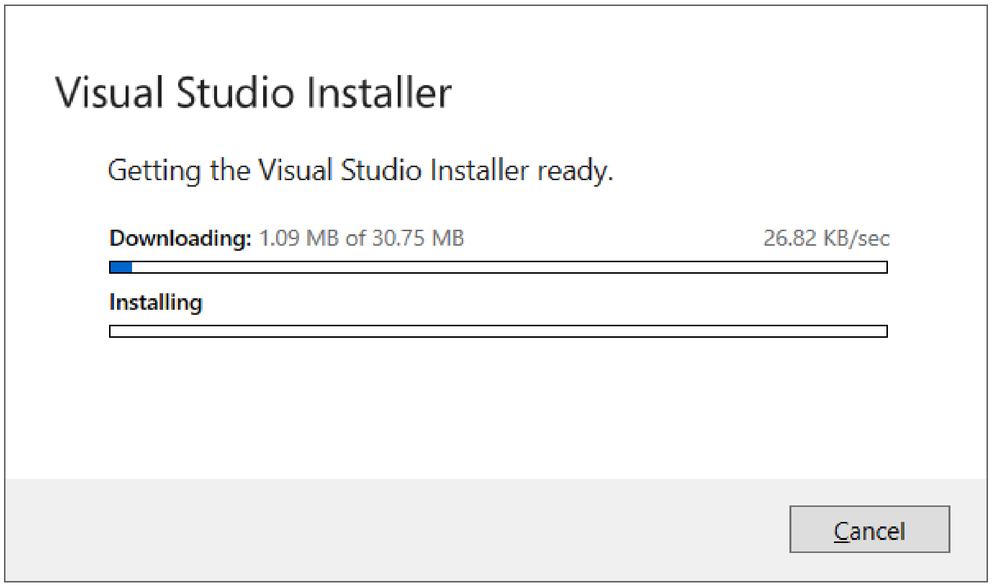
2. If you receive a User Account Control notice, choose **Yes**. The dialog box asks you to acknowledge the Microsoft License Terms and the Microsoft Privacy Statement. Choose **Continue**.



Visual Studio Installer

Before you get started, we need to set up a few things so that you can configure your installation.

To learn more about privacy, see the Microsoft Privacy Statement.
By continuing, you agree to the Microsoft Software License Terms.



Visual Studio Installer

Getting the Visual Studio Installer ready.

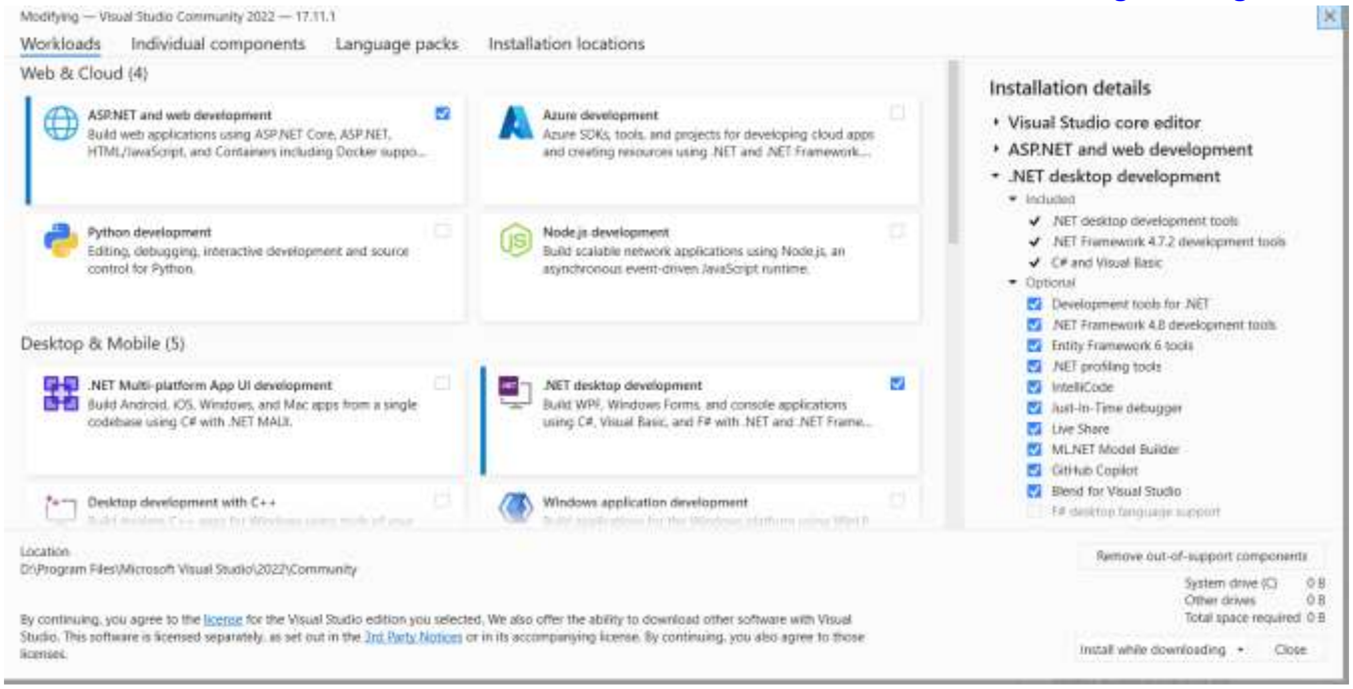
Downloading: 1.09 MB of 30.75 MB 26.82 KB/sec

Installing

Choose workloads

After you install the Visual Studio Installer, you can use it to customize your installation by selecting the feature sets, or *workloads*, that you want. Here's how.

1. Find the workload that you want in the **Visual Studio Installer**.



Modifying — Visual Studio Community 2022 — 17.11.1

Workloads | Individual components | Language packs | Installation locations

Web & Cloud (4)

- ✓ **ASP.NET and web development**
  Build web applications using ASP.NET Core, ASP.NET, HTML/JavaScript, and Containers including Docker suppo...
- **Azure development**
  Azure SDKs, tools, and projects for developing cloud apps and creating resources using .NET and .NET Framework...
- **Python development**
  Editing, debugging, interactive development and source control for Python.
- **Node.js development**
  Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

Desktop & Mobile (5)

- **.NET Multi-platform App UI development**
  Build Android, iOS, Windows, and Mac apps from a single codebase using C# with .NET MAUI.
- ✓ **.NET desktop development**
  Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET and .NET Frame...
- **Desktop development with C++**
- **Windows application development**

Installation details

- Visual Studio core editor
- ASP.NET and web development
- .NET desktop development
  - Included
    - ✓ .NET desktop development tools
    - ✓ .NET Framework 4.7.2 development tools
    - ✓ C# and Visual Basic
  - Optional
    - ✓ Development tools for .NET
    - ✓ .NET Framework 4.8 development tools
    - ✓ Entity Framework 6 tools
    - ✓ .NET profiling tools
    - ✓ IntelliCode
    - ✓ Just-In-Time debugger
    - ✓ Live Share
    - ✓ ML.NET Model Builder
    - ✓ GitHub Copilot
    - ✓ Blend for Visual Studio
    - F# desktop language support

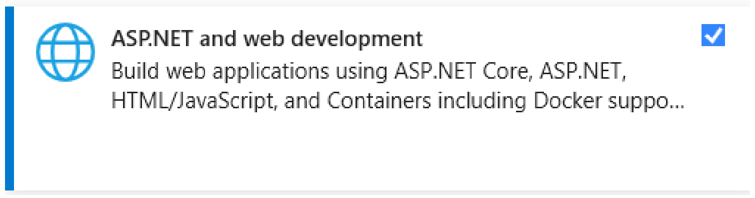
Location
D:\Program Files\Microsoft Visual Studio\2022\Community

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Remove out-of-support components

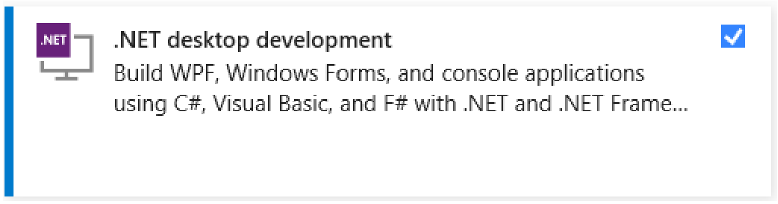
System drive (C)	0 B
Other drives	0 B
Total space required	0 B

1. For example, choose the **ASP.NET and web development** workload. It comes with the default core editor. That editor includes basic code editing support for over 20 languages, the ability to open and edit code from any folder without requiring a project, and integrated source code control.



✓ **ASP.NET and web development**
Build web applications using ASP.NET Core, ASP.NET, HTML/JavaScript, and Containers including Docker suppo...

2. Choose .NET desktop development workloads, then select Install.



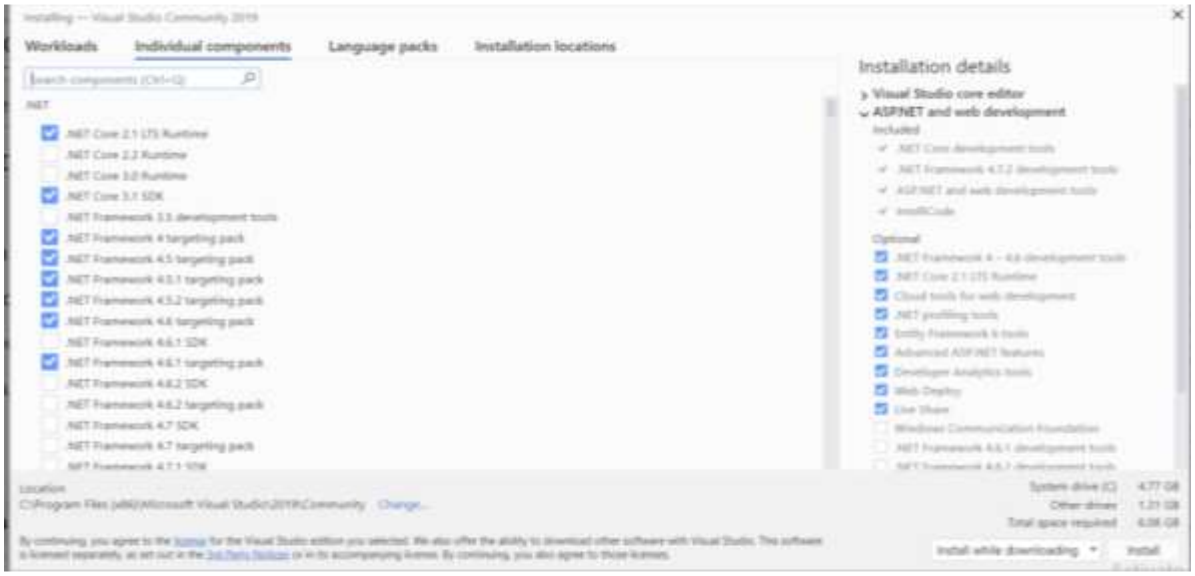
✓ **.NET desktop development**
Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET and .NET Frame...

Tip

At any time after installation, you can install workloads or components that you didn't install initially. If you have Visual Studio open, go to **Tools** > **Get Tools and Features**, which opens the Visual Studio Installer. Or, open the **Visual Studio Installer** from the **Start** menu. From there, you can choose the workloads or components that you wish to install. Then, choose **Modify**.

Choose individual components (optional)

If you don't want to use the Workloads feature to customize your Visual Studio installation, or you want to add more components than a workload installs, you can install or add individual components from the **Individual components** tab. Choose what you want, and then follow the prompts.


Installing — Visual Studio Community 2019

Workloads | Individual components | Language packs | Installation locations

Search components (Ctrl+Q)

.NET

- [✓] .NET Core 2.1 LTS Runtime
- [ ] .NET Core 2.2 Runtime
- [ ] .NET Core 3.0 Runtime
- [✓] .NET Core 3.1 SDK
- [ ] .NET Framework 3.5 development tools
- [✓] .NET Framework 4 targeting pack
- [✓] .NET Framework 4.5 targeting pack
- [✓] .NET Framework 4.5.1 targeting pack
- [✓] .NET Framework 4.5.2 targeting pack
- [✓] .NET Framework 4.6 targeting pack
- [ ] .NET Framework 4.6.1 SDK
- [✓] .NET Framework 4.6.1 targeting pack
- [ ] .NET Framework 4.6.2 SDK
- [ ] .NET Framework 4.6.2 targeting pack
- [ ] .NET Framework 4.7 SDK
- [ ] .NET Framework 4.7 targeting pack
- [ ] .NET Framework 4.7.1 SDK

Installation details

> Visual Studio core editor

ASP.NET and web development

Included
- ✓ .NET Core development tools
- ✓ .NET Framework 4.7.2 development tools
- ✓ ASP.NET and web development tools
- ✓ IntelliCode

Optional
- [✓] .NET Framework 4 – 4.6 development tools
- [✓] .NET Core 2.1 LTS Runtime
- [✓] Cloud tools for web development
- [✓] .NET profiling tools
- [✓] Entity Framework 6 tools
- [✓] Advanced ASP.NET features
- [✓] Developer Analytics tools
- [✓] Web Deploy
- [✓] Live Share
- [ ] Windows Communication Foundation
- [ ] .NET Framework 4.6.1 development tools
- [ ] .NET Framework 4.6.2 development tools

Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community Change...

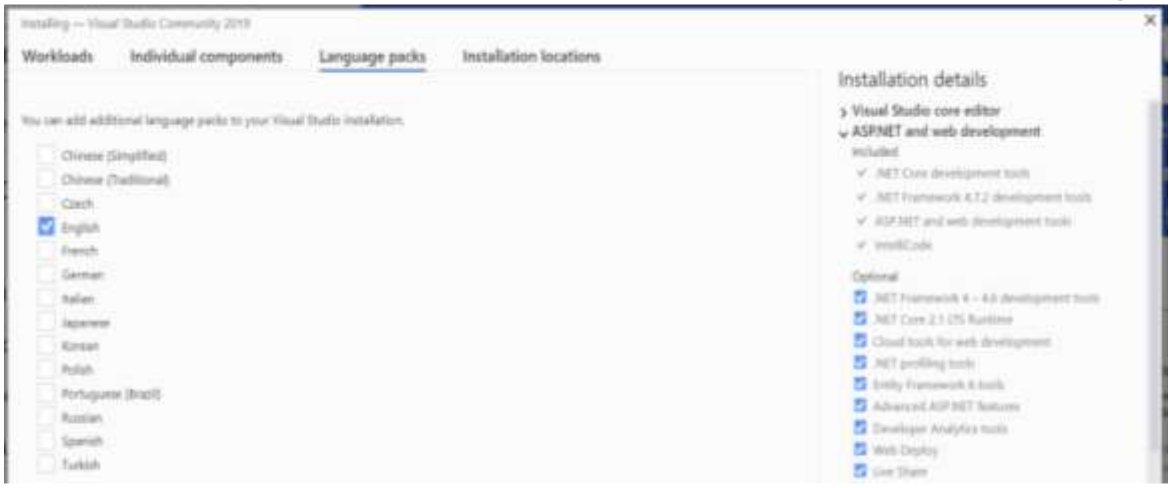
By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

System drive (C)	4.77 GB
Other drives	1.21 GB
Total space required	6.08 GB

Install while downloading | Install

Install language packs (optional)

By default, the installer program tries to match the language of the operating system when it runs for the first time. To install Visual Studio in a language of your choosing, choose the **Language packs** tab from the Visual Studio Installer, and then follow the prompts.



Installing — Visual Studio Community 2019

Workloads | Individual components | Language packs | Installation locations

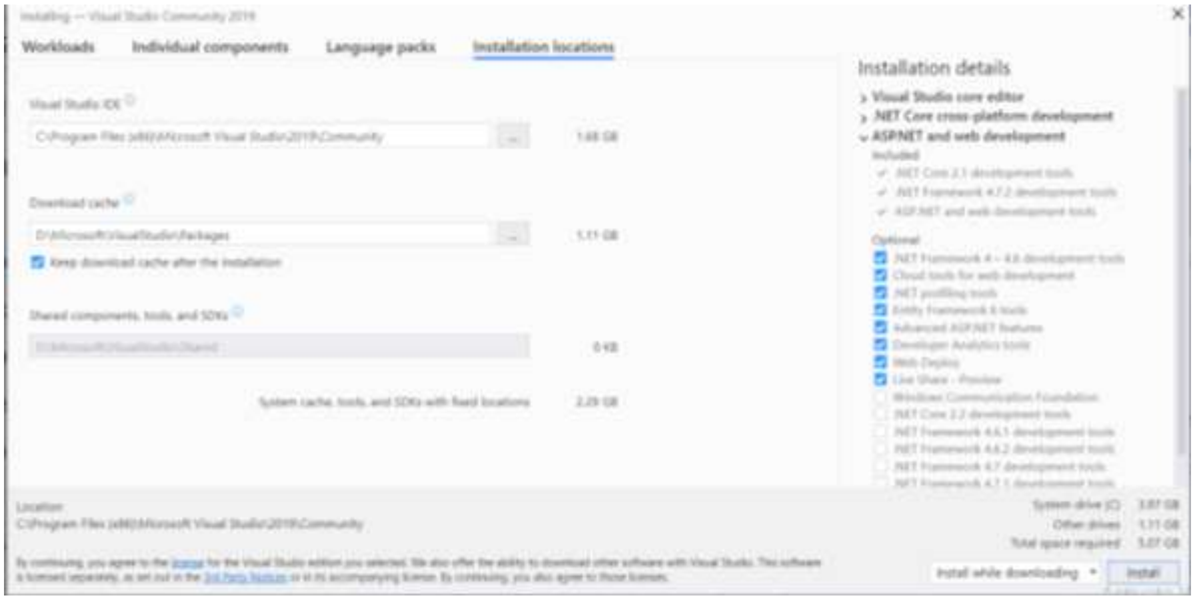
You can add additional language packs to your Visual Studio installation.

- [ ] Chinese (Simplified)
- [ ] Chinese (Traditional)
- [ ] Czech
- [x] English
- [ ] French
- [ ] German
- [ ] Italian
- [ ] Japanese
- [ ] Korean
- [ ] Polish
- [ ] Portuguese (Brazil)
- [ ] Russian
- [ ] Spanish
- [ ] Turkish

Installation details

› Visual Studio core editor
⌄ ASP.NET and web development
Included
✓ .NET Core development tools
✓ .NET Framework 4.7.2 development tools
✓ ASP.NET and web development tools
✓ IntelliCode
Optional
- [x] .NET Framework 4 – 4.6 development tools
- [x] .NET Core 2.1 LTS Runtime
- [x] Cloud tools for web development
- [x] .NET profiling tools
- [x] Entity Framework 6 tools
- [x] Advanced ASP.NET features
- [x] Developer Analytics tools
- [x] Web Deploy
- [x] Live Share

Select the installation location (optional)



Installing — Visual Studio Community 2019

Workloads | Individual components | Language packs | Installation locations

Visual Studio IDE
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community — 1.68 GB

Download cache
D:\Microsoft\VisualStudio\Packages — 1.11 GB
- [x] Keep download cache after the installation

Shared components, tools, and SDKs
D:\Microsoft\VisualStudio\Shared — 0 KB

System cache, tools, and SDKs with fixed locations — 2.29 GB

Installation details

› Visual Studio core editor
› .NET Core cross-platform development
⌄ ASP.NET and web development
Included
✓ .NET Core 2.1 development tools
✓ .NET Framework 4.7.2 development tools
✓ ASP.NET and web development tools
Optional
- [x] .NET Framework 4 – 4.6 development tools
- [x] Cloud tools for web development
- [x] .NET profiling tools
- [x] Entity Framework 6 tools
- [x] Advanced ASP.NET features
- [x] Developer Analytics tools
- [x] Web Deploy
- [x] Live Share - Preview
- [ ] Windows Communication Foundation
- [ ] .NET Core 2.2 development tools
- [ ] .NET Framework 4.6.1 development tools
- [ ] .NET Framework 4.6.2 development tools
- [ ] .NET Framework 4.7 development tools
- [ ] .NET Framework 4.7.1 development tools

Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community

System drive (C) 3.87 GB
Other drives 1.11 GB
Total space required 5.07 GB

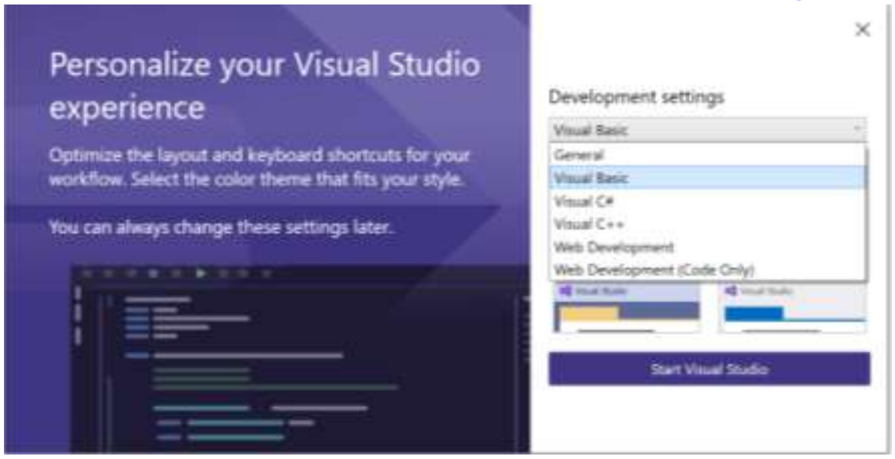
By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Install while downloading | Install

ⓘ **Important**

You can select a different drive for **Visual Studio IDE** or **Download cache** only when you first install Visual Studio. If you already installed it and want to change drives, you must uninstall Visual Studio and then reinstall it.

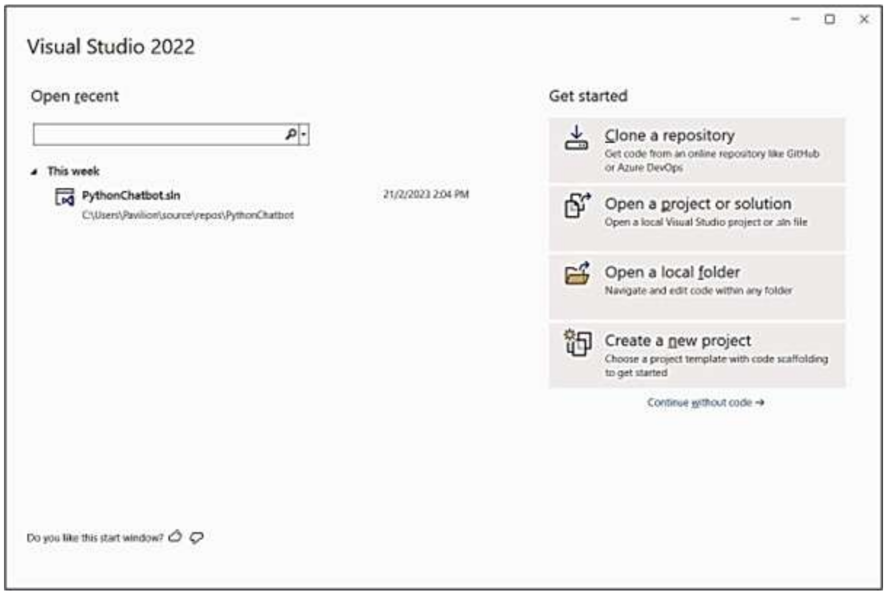
If you installed Visual Studio on your computer before, you won't be able to change the **Shared components, tools, and SDKs** path. It appears greyed out. This location is shared by all installations of Visual Studio.


Personalize your Visual Studio experience
Optimize the layout and keyboard shortcuts for your workflow. Select the color theme that fits your style.
You can always change these settings later.
Development settings
Visual Basic
General
Visual Basic
Visual C#
Visual C++
Web Development
Web Development (Code Only)
Start Visual Studio

Start developing

After installation is complete, you can get started developing with Visual Studio.

- Select the Launch button.
- On the start window, choose Create a new project.
- In the search box, enter the type of app you want to create to see a list of available templates. The list of templates depends on the workloads that you chose during installation. To see different templates, choose different workloads.
- Select Next. Provide other information in the following dialog boxes, and then select **Create**.


Visual Studio 2022
Open recent
This week
PythonChatbot.sln
C:\Users\Pavilion\source\repos\PythonChatbot
21/2/2023 2:04 PM
Get started
Clone a repository
Get code from an online repository like GitHub or Azure DevOps
Open a project or solution
Open a local Visual Studio project or .sln file
Open a local folder
Navigate and edit code within any folder
Create a new project
Choose a project template with code scaffolding to get started
Continue without code →
Do you like this start window?

Introduction to Visual Studio

Microsoft released Studio allows you to code in different programming languages for different platforms, Visual Basic is one of them. The other Programming languages are C# C++, F#, JavaScript, Java and Python. Visual Basic 2022 is the latest version VB.NET programming language released by Microsoft.

What is Visual Studio?

Visual Studio is a development kit (also known as an Integrated Development Environment or IDE) that is mainly for use by software developers to create products, internet, and tools. Microsoft first released Visual Studio in 1998.

Visual Studio is an integrated development environment (IDE) that includes Visual Basic, Visual C++, Visual C#, and other programming languages.

What is a Visual Basic?

Visual Basic is a festival language of programming, similar to fundamental (coding) but with sliders, texts, and other visuals programmed to do an operation.

What is VB.NET ?

VB.NET, also known as Visual Basic.NET, is an object-oriented programming language introduced by Microsoft in 2002. It serves as the successor to Visual Basic 6.0 and is built on the Microsoft .NET Framework.

As a simple and high-level language, VB.NET enables developers to create fully object-oriented applications, comparable to those developed using languages like C++, Java, or C#. With its intuitive syntax and extensive framework support, VB.NET empowers developers to build robust and feature-rich software solutions.

Download the free version of Visual Studio 2022 from the following link:

https://visualstudio.microsoft.com/vs/

After downloading and installing Visual Studio 2022, be ready to launch Visual Studio 2022 and start programming in Visual Basic 2022.

Visual Basic vs VB.NET

Visual Basic	**Visual Basic .NET**
Event-driven programming language	Object-oriented programming language with support for the event-driven paradigm
Relatively simple and easier to learn	More robust syntax with improved features and capabilities
Relies on the Visual Basic 6 runtime and VB runtime libraries	Relies on the Common Language Runtime (CLR). Utilizes the .NET Framework
Limited interoperability with other languages and platforms	Supports interoperability with other .NET languages and platforms
Relies on implicit memory management (automatic garbage collection)	Utilizes the CLR's garbage collector for memory management
Primarily Windows platform	Cross-platform compatibility through .NET Framework and .NET Core

Visual Studio 2022 Start Page

The Visual Studio 2022 start page is quite different from Visual Studio 2017. When first launch Visual Studio 2022, the following start Page appears, as shown in Figure 1.1.

- Open recent: open recently opened projects,open a project or solution, open a local folder
- Create a new project.

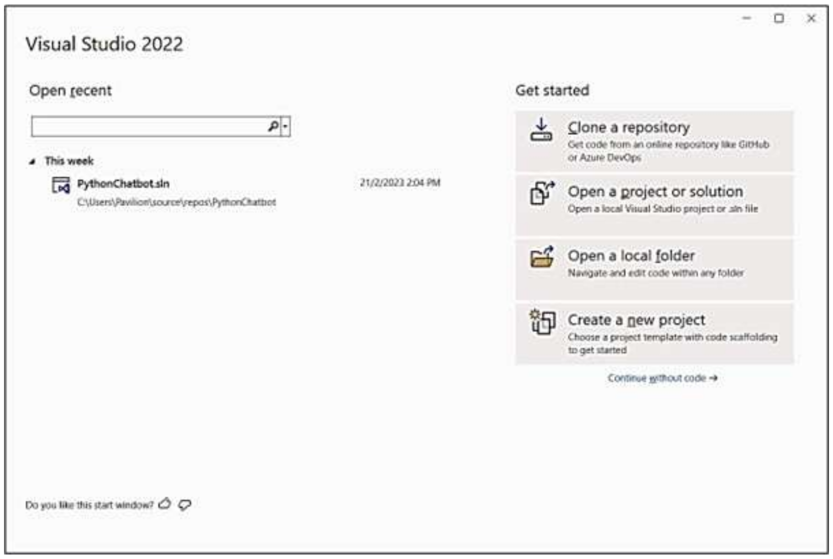

Visual Studio 2022
Open recent
This week
PythonChatbot.sln
C:\Users\Pavilion\source\repos\PythonChatbot
21/2/2023 2:04 PM
Get started
Clone a repository
Get code from an online repository like GitHub or Azure DevOps
Open a project or solution
Open a local Visual Studio project or .sln file
Open a local folder
Navigate and edit code within any folder
Create a new project
Choose a project template with code scaffolding to get started
Continue without code →
Do you like this start window?

Figure 1.1 Visual Studio 2022 Start Page

Create a new project option to Create a new project page, as shown in Figure 1.2. (select the Visual Basic language). Create a new project by clicking the Create a new project option.
The create a new project template page, as shown in Figure 1.2. has a dozen templates to choose from, scroll down to select application.

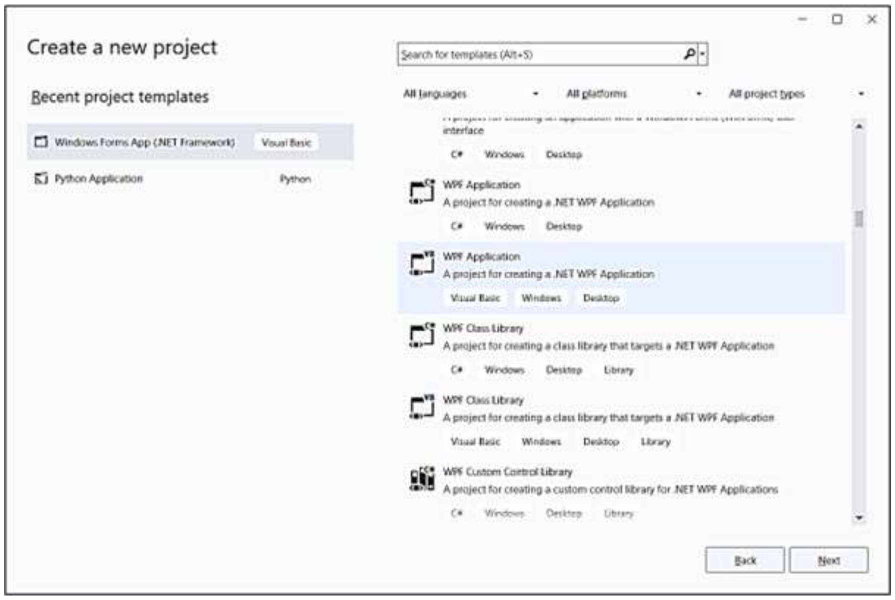
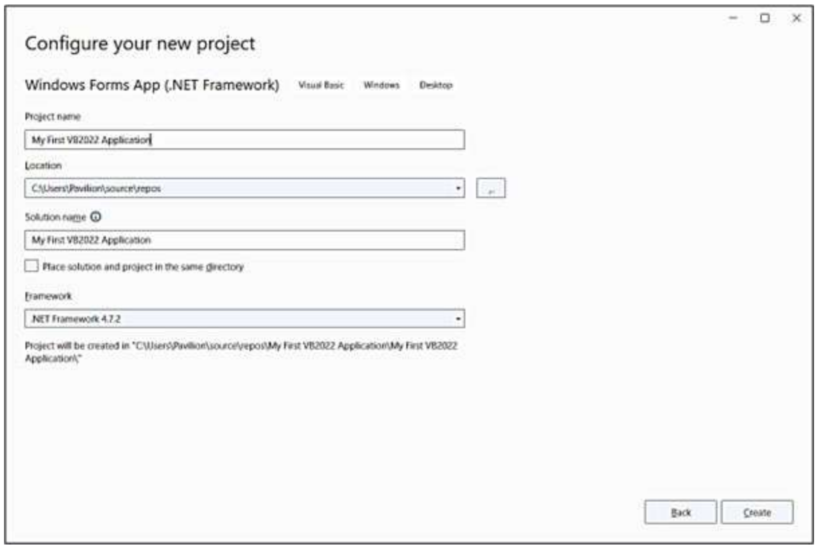

Create a new project
Recent project templates
Windows Forms App (.NET Framework) Visual Basic
Python Application Python
Search for templates (Alt+S)
All languages
All platforms
All project types
interface
C# Windows Desktop
WPF Application
A project for creating a .NET WPF Application
C# Windows Desktop
WPF Application
A project for creating a .NET WPF Application
Visual Basic Windows Desktop
WPF Class Library
A project for creating a class library that targets a .NET WPF Application
C# Windows Desktop Library
WPF Class Library
A project for creating a class library that targets a .NET WPF Application
Visual Basic Windows Desktop Library
WPF Custom Control Library
A project for creating a custom control library for .NET WPF Applications
C# Windows Desktop Library
Back
Next

Figure 1.2 New Project Page

Upon clicking the selected project template, the project configuration page appears, as shown in Figure 1.3. to configure project by typing the project name and select a few other options.


Configure your new project

Windows Forms App (.NET Framework) Visual Basic Windows Desktop

Project name: My First VB2022 Application

Location: C:\Users\Pavilion\source\repos

Solution name: My First VB2022 Application

☐ Place solution and project in the same directory

Framework: .NET Framework 4.7.2

Project will be created in "C:\Users\Pavilion\source\repos\My First VB2022 Application\My First VB2022 Application\"

Figure 1.3 Configuring Project

Visual Basic 2022 IDE

Clicking the Create button will launch the Visual Basic 2022 IDE, as shown in Figure 1.4. the name of the project entered earlier appears on the top right corner of the IDE.

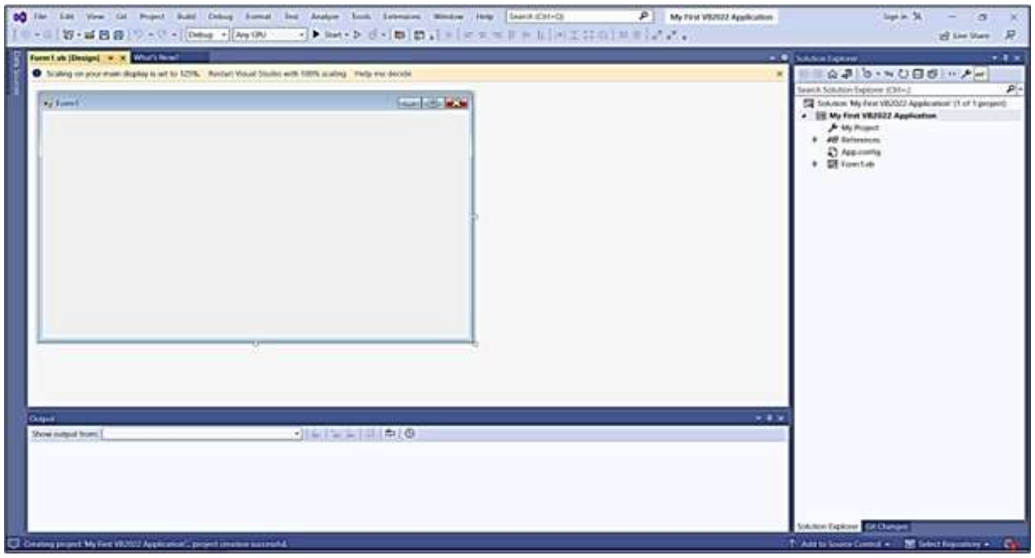


Figure 1.4 Visual Basic 2022 IDE

VB2022 IDE comprises a few windows:

- the Form window
- the Solution Explorer window
- the Properties window.
- The Toolbox

The Toolbox is not shown until click the Toolbox tab. When click the Toolbox tab or use the shortcut keys Ctrl+Alt+x, the common controls Toolbox will appear, as shown in Figure 1.5. drag and move toolbox around and dock it to the right, left, top or bottom of the IDE.

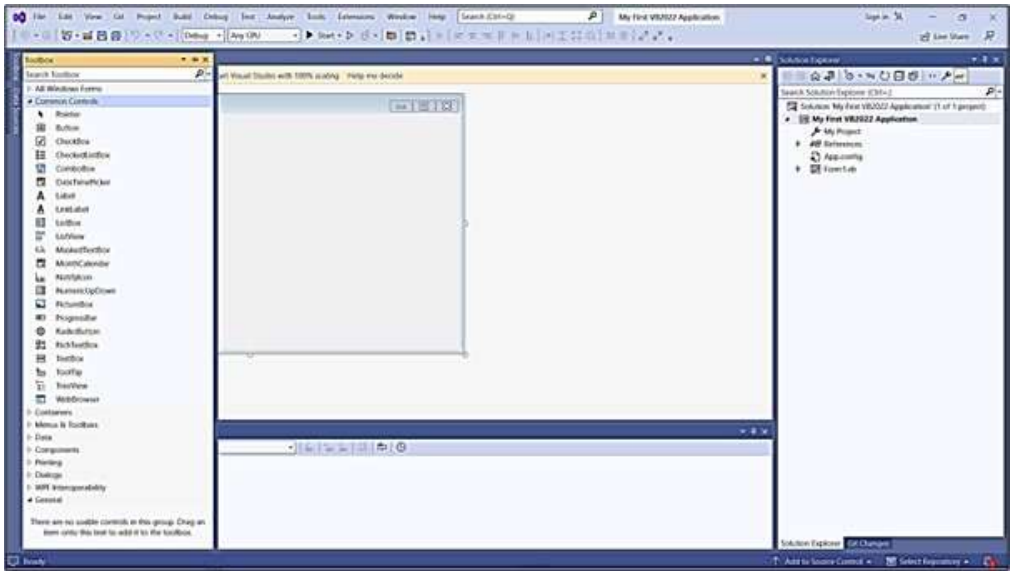


Figure 1.5 The Design Interface

Next, we shall proceed to show you how to create your first VB2022 application. First, change the text of the form to ‘My First VB 2022 Application’ in the properties window; it will appear as the title of the application. Next, insert a button and change its text to OK. The design interface is shown in Figure 1.6

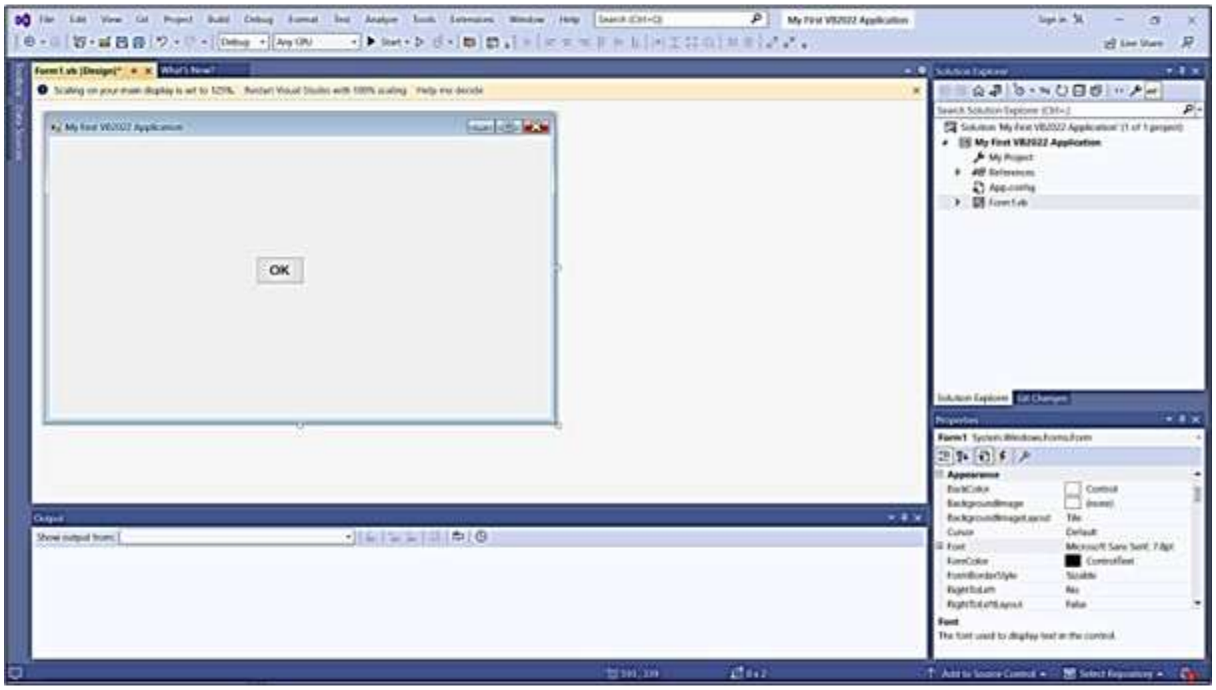


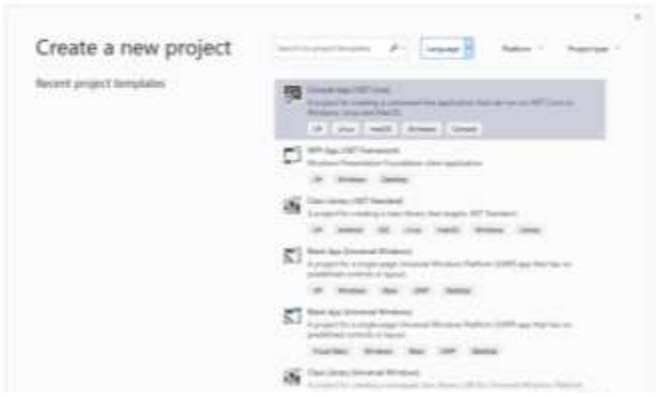
Figure 1.6 The Design Interface

Now click on the OK button to bring up the code window and enter “END” statement between Private Sub and End Sub procedure.

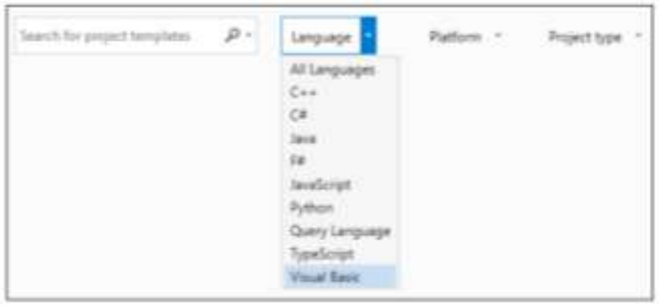
Practical Notes : Create a New Project in Visual Studio Community

When start the software, this shown on the **Create a new project** dialogue box :

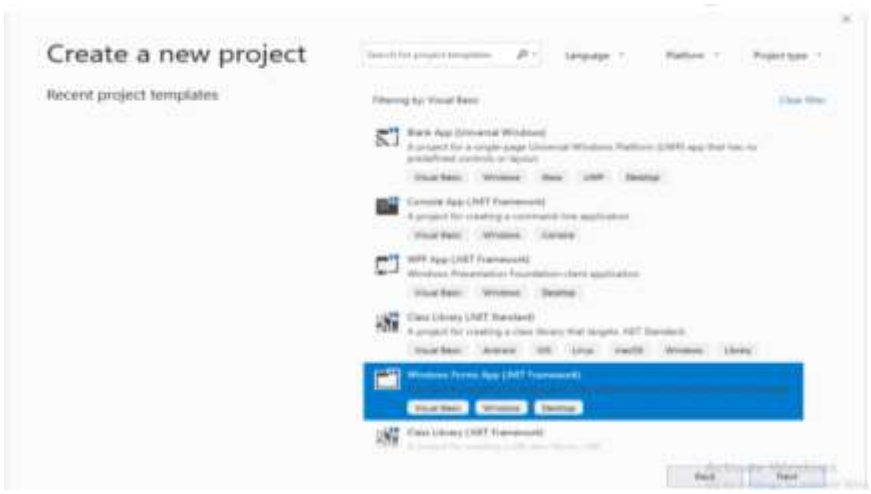
- Any projects create will appear on the left, under **Open recent**,
- **Create a new project**, on the right in the image.



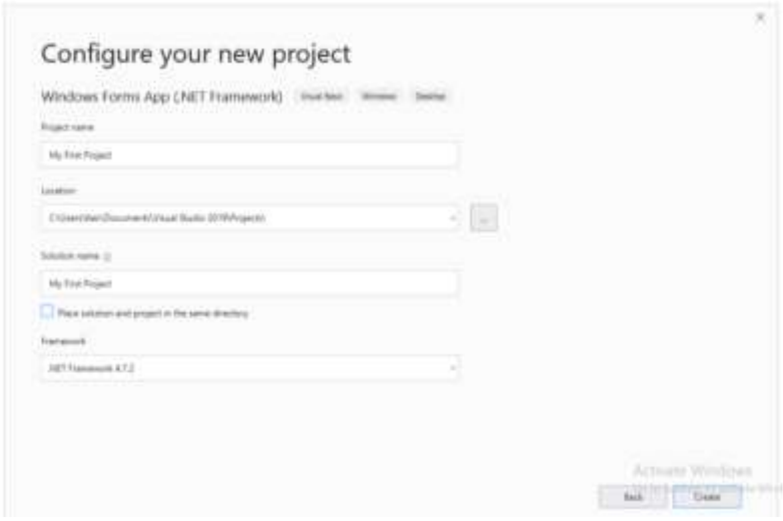
At the top, click the dropdown list for Language. Then select the language (The image below shows Visual Basic selected).



select the option for **Windows Form App** (.NET Framework).the Visual Basic list of project templates.


Create a new project
Recent project templates
Search for project templates
Language
Platform
Project type
Filtering by: Visual Basic
Blank App (Universal Windows)
Console App (.NET Framework)
WPF App (.NET Framework)
Class Library (.NET Standard)
Windows Forms App (.NET Framework)
Class Library (.NET Framework)
Back
Next

When click Next, **Configure your new project** screen shown:



Configure your new project

Windows Forms App (.NET Framework)

Project name

My First Project

Location

C:\Users\[illegible]\Documents\Visual Studio 2019\Projects\

Solution name

My First Project

Place solution and project in the same directory

Framework

.NET Framework 4.7.2

In the **Project name** box at the top, type **My First Project**. created a folder called Projects. The **Solution name** box will get filled in for you. So just click the **Create** button.

You can choose a location to save your projects

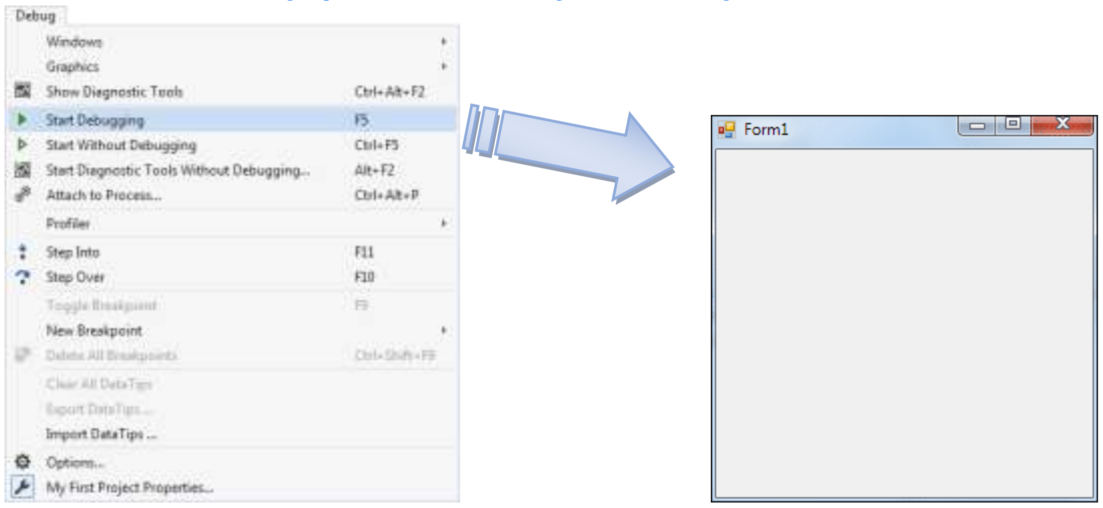
Visual Basic .NET Forms

In the Visual Basic NET design time environment, the first thing to concentrate on is that strange, big square in the top left. That's called a form.

Running your VB Forms

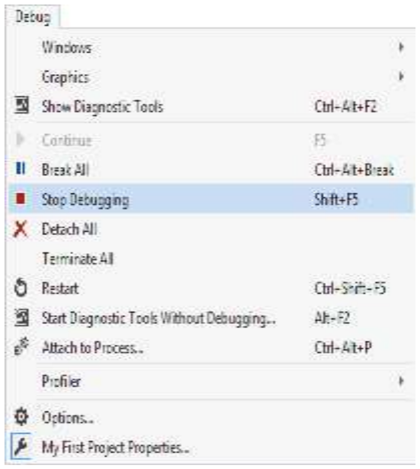
To run the form, try this:

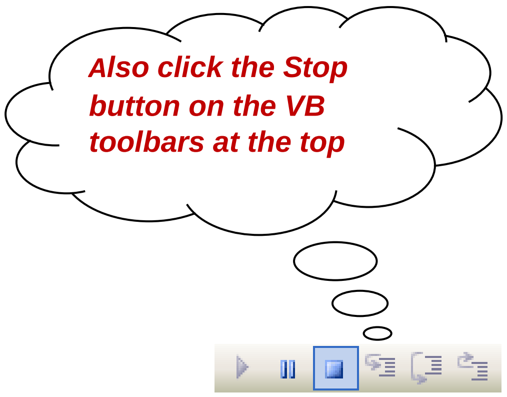
- ***From the menu bar, click Debug***
- ***From the drop down menu, click Start***
- ***Alternatively, press the F5 key on the keyboard***


Debug
Windows
Graphics
Show Diagnostic Tools Ctrl+Alt+F2
Start Debugging F5
Start Without Debugging Ctrl+F5
Start Diagnostic Tools Without Debugging... Alt+F2
Attach to Process... Ctrl+Alt+P
Profiler
Step Into F11
Step Over F10
Toggle Breakpoint F9
New Breakpoint
Delete All Breakpoints Ctrl+Shift+F9
Clear All DataTips
Export DataTips...
Import DataTips...
Options...
My First Project Properties...
Form1

To stop the form running, do one of the following:

- ***Click the Red X at the top right of the Form***
- ***Click Debug > Stop Debugging from the menu bar***
- ***Press Shift + F5 on the keyboard***


Debug
Windows
Graphics
Show Diagnostic Tools Ctrl+Alt+F2
Continue F5
Break All Ctrl+Alt+Break
Stop Debugging Shift+F5
Detach All
Terminate All
Restart Ctrl+Shift+F5
Start Diagnostic Tools Without Debugging... Alt+F2
Attach to Process... Ctrl+Alt+P
Profiler
Options...
My First Project Properties...



Adding Controls Using the Toolbox

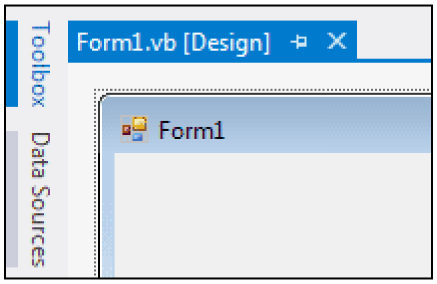
all things that can added to Forms known as Controls. Controls are objects that consist of three elements:

- Properties
- Methods
- Events

The most common controls are the button, label, text box, list box, combo box, PictureBox, checkbox, radio and more. It can be make the controls visible or invisible at runtime. However, some controls will only run in the background and invisible at runtime, one such control is the timer.

To add a control to the Form, just drag the control from the toolbox and drop it onto the Form. The ***Toolbox can be found on the left of the screen*** (The Toolbox is usually is hidden when you start Visual Basic 2022)

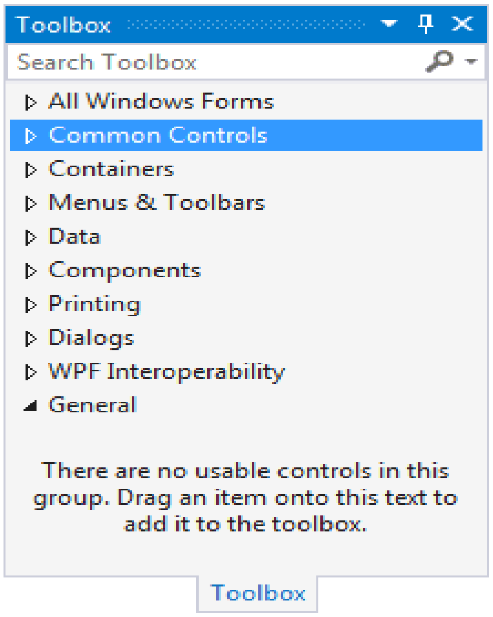
Also , ***click View on the menu bar and then select Toolbox*** to reveal the toolbox. Besides that, use ***shortcut keys Ctrl+Alt+x*** to bring out the toolbox. The picture below show the toolbox icon next to Form1

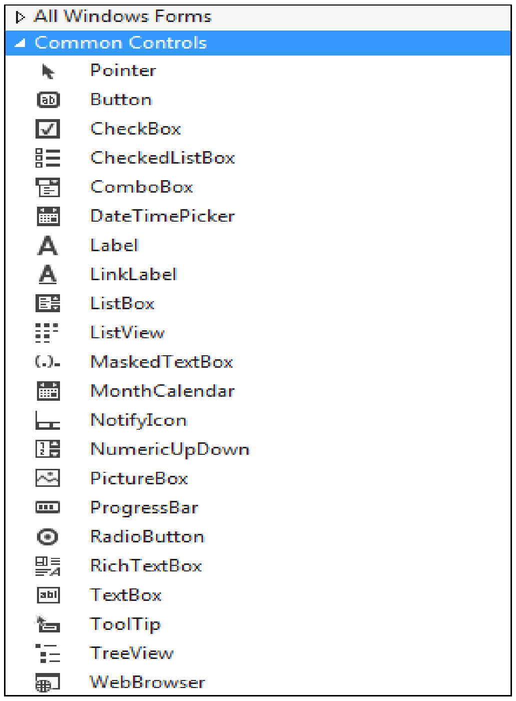


To display all the tools, click on the Toolbox tab. There are lots of categories of tools available. The first toolbox working with is the **Common Controls** toolbox. To see the tools, click on the plus or arrow symbol next to **Common Controls**. There are an awful lot of tools to choose from. For this first section, can using the **Button**, the **TextBox** and the **Label**.

Note:

- *To keep the toolbox displayed, click the **Pin icon** next to the X.*
- *To close the toolbox, simply move mouse away. Or click the word **Toolbox** again.*





Adding a Tool (Control) to the Form

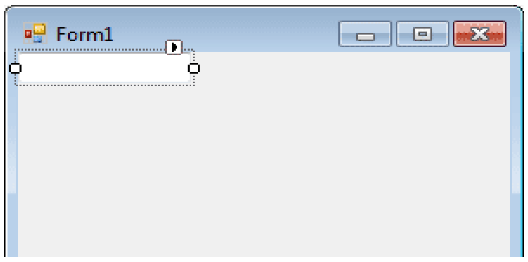
Add a Textbox to VB Form

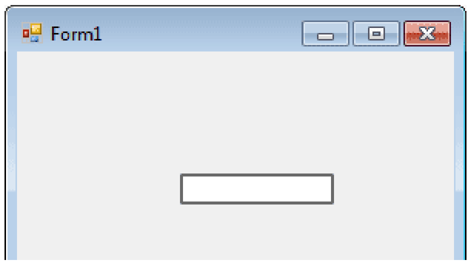
TextBox is for accepting input from the user as well as to display the output. It can handle string and numeric data but not images or pictures. String in a text box can be converted to a numeric data by using the function Val(text).

Start by adding a textbox to form. With the **Common Controls** displayed, do the following:

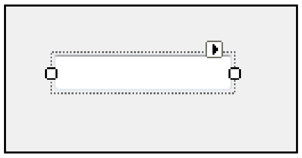
1. **Locate the TextBox tool**
2. **Double click the icon**
3. **A textbox is added to your form:**

The textbox gets added to the top left position of the form(see left image). To move it down, hold the mouse over the textbox and drag to a new position(see right image)

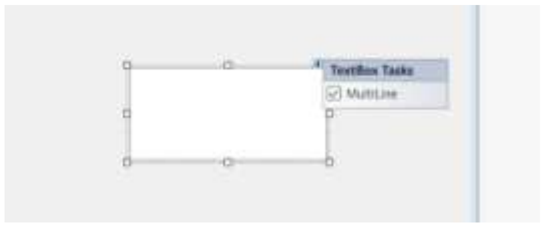




Notice the small squares around the textbox when you let go of the left mouse button:

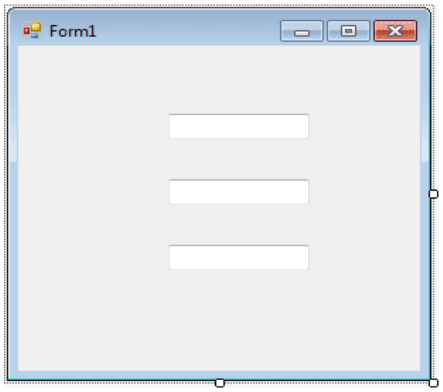


The small squares are sizing handles. Move the mouse over one of them. The mouse pointer turns into an extended line with arrowheads. Hold left mouse button down and drag outwards. The textbox is resized.



One thing will notice is that “can't make the size any higher, but you can make it wider”. because the default action of a textbox is to have it contain only a single line of text. If it's only going to contain one line of text. A textbox can only be made higher if it's set to contain multiple lines of text. to do this :

1. **Create two more textboxes by double clicking on the textbox icon in the toolbar (Or Right-click on the selected textbox and choose Copy. Then Right-click on the Form and choose Paste.)**
2. **Resize them to the same size as your first one**
3. **Line them up one below the other with space in between**
4. **Try to create something that looks like the one below**

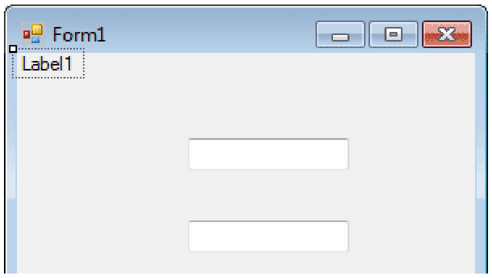


Add a Label to VB Form

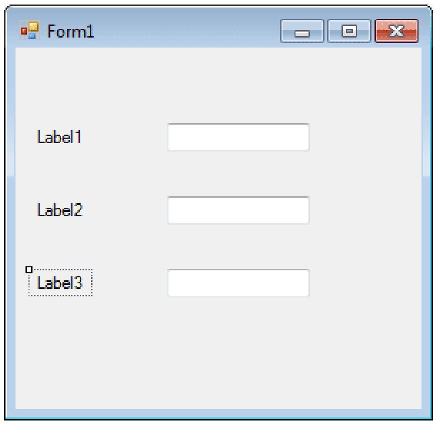
Label is used for multiple purposes like providing instructions and guides to the users, displaying outputs and more. It is different from the TextBox because it is read only, which means the user cannot edit its content at runtime.

Using the syntax Label.Text, it can display string as well as numeric data. You can change its text property in the properties window or at runtime by writing an appropriate code.
Add labels near the textboxes so that users will know what they are for.

1. **Locate the label control in the toolbox**
2. **Double click the label icon**
3. **A new label is added to the form**
4. **It should look like the one below**



Click on the label to select it. Now hold left mouse button down on the label. Keep it held down and drag it to the left of the textbox.
Create two more labels, and position them to the left of the textboxes. a form should like this one:



Now a form with textboxes and labels, something that looks like a form people can fill in. But those labels are not exactly descriptive, and our textboxes have the default text in them.

So how can we enter our own text for the labels, and get rid of that default text for the textboxes?

To do those things, we need to discuss something called a Property.

Designing the User Interface(UI)

Before embarking on the creation of Visual Basic 2022 projects, it's essential to conceptualize the type of this projects. Whether it's a desktop game, a multimedia app, a financial tool, or a database management application, having a clear vision is crucial.

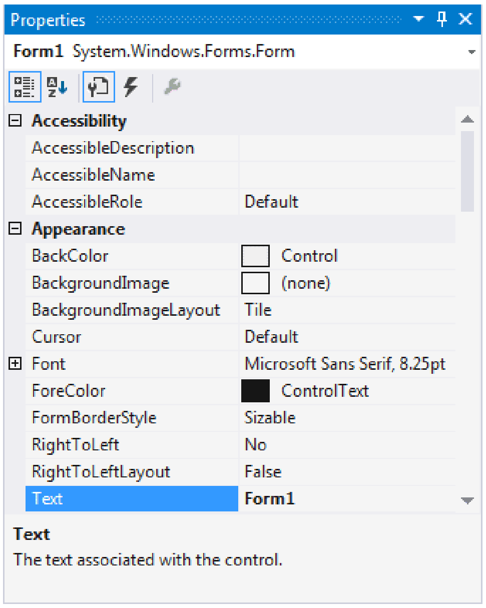
Once determined the nature of the application, the initial step involves designing the User Interface (UI). Starting with a sketch on paper to visualize and refine the design before delving into the VB2022 IDE. This thoughtful planning process ensures a more streamlined and effective development journey.

To design a graphical user interface, it may customize the default form by changing its properties at design phase and at runtime, including its name, title, background color and so forth. After customizing the default form, add controls from the toolbox to the form and then define their properties

VB .NET Properties

The area to the right of the design environment , That's the Properties box:

Click anywhere on the form that (somewhere on the form's grey areas). The form should have the little sizing handles now, indicating that the form is selected. On the right of the design environment there should be the following Properties box:



Form1 System.Windows.Forms.Form

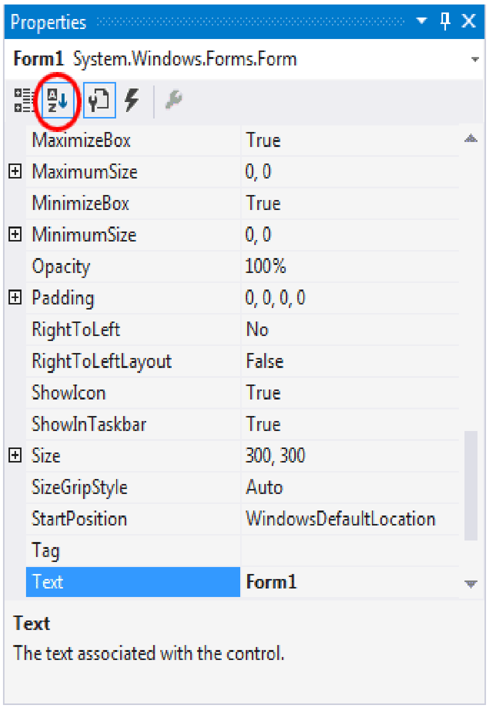
Accessibility	
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
Appearance	
BackColor	☐ Control
BackgroundImage	☐ (none)
BackgroundImageLayout	Tile
Cursor	Default
Font	Microsoft Sans Serif, 8.25pt
ForeColor	■ ControlText
FormBorderStyle	Sizable
RightToLeft	No
RightToLeftLayout	False
Text	**Form1**

Text
The text associated with the control.

The properties window comprises an object drop-down list, a list of properties and the bottom section that shows a description of the selected property. As the only object on the IDE is the default form by the name of Form1(when create new project) , the properties window displays all properties related to Form1 and their corresponding attributes or values.

A list of the properties that a form has: Name , BackColor, Font, Image, Text, etc.
Just to the right of these properties are the values for them. These values are the default values, and can be changed.

First, to display the list of Properties in a more accessible form (display the list properties alphabetically) click the Alphabetic icon at the top of the Properties box, as in the image:


Properties

Form1 System.Windows.Forms.Form

MaximizeBox	True
⊞ MaximumSize	0, 0
MinimizeBox	True
⊞ MinimumSize	0, 0
Opacity	100%
⊞ Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
⊞ Size	300, 300
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**Form1**

Text
The text associated with the control.

Properties can be changed by :

- typing a value
- select a value from a drop-down list
- For the color setting, need to select a color rectangle or from a color palette.

Another method of setting the colors is to manually type the RGB color code or the hex color code. The values of R, G and B ranges from 0 to 255, therefore, by varying the values of the RGB we can obtain different colors.

What is a Property?

Those controls added to the form (textboxes and labels), and the form itself, are called control objects. controls objects (as things), something solid that can pick up and move about. Controls (things) have properties.

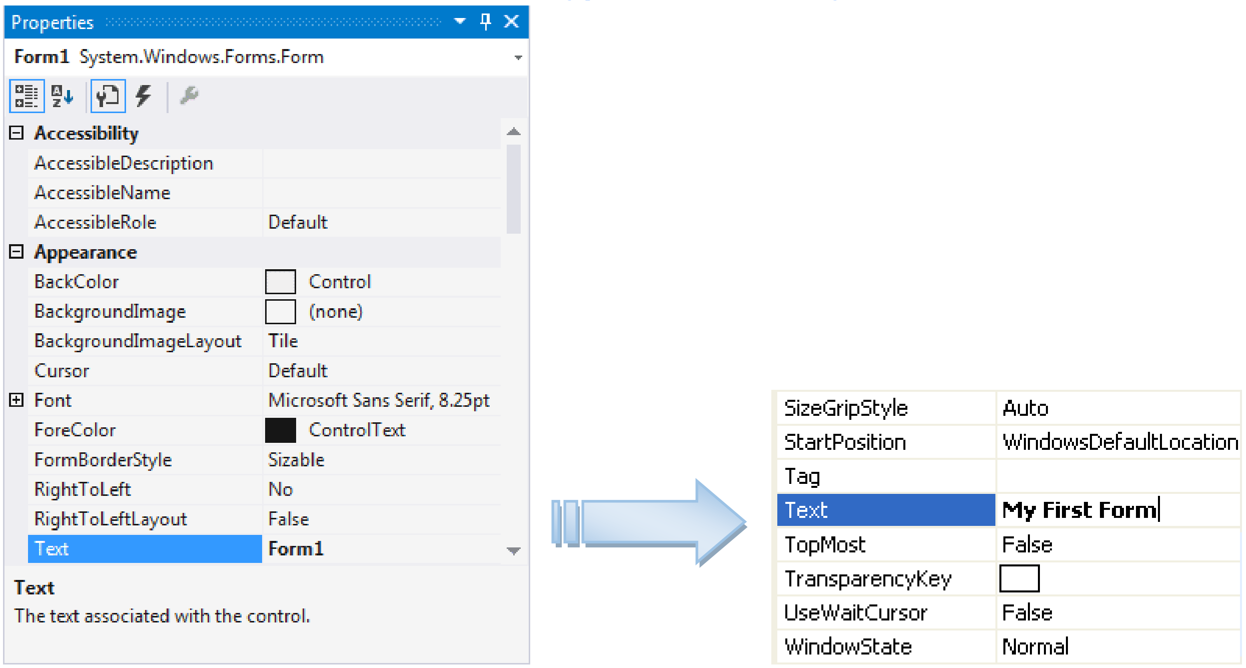
If your phone were a control, it too would have properties: a code property, a color property, a volume property, and a ... well, what other properties would your phone have? Think about it.

The properties of phone will have values. The code would have just one value. The volume property could have a range of values, from zero to ten, for example.
In VB.NET, can change a property of a control from the Properties Box. (can also change a property using code)
To change only one value - the value of the Text property . So, locate the word "Text" in the Property box, as in the image below.

"Text" is a Property of Form1. Don't be confused by the word "Form1" next to the word "Text". All this means is that the current value of the Text property is set to the word "Form1". This is the default.

To change this to something of your own, do this:

- ***Click inside the area next to "Text", and delete the word "Form1" by hitting the backspace key on your keyboard***
- ***When "Form1" has been deleted, type the words "My First Form"***



Properties

Form1 System.Windows.Forms.Form

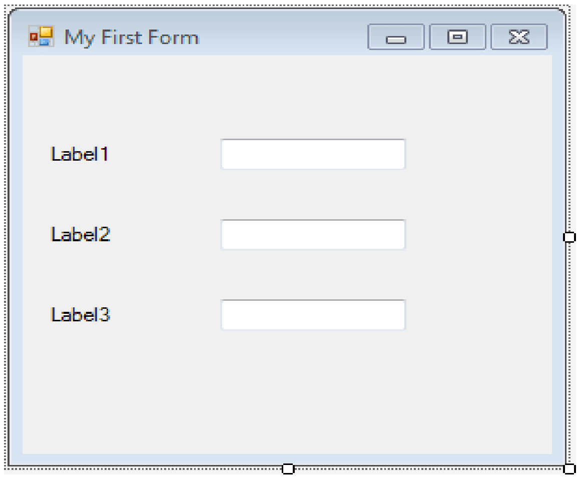
⊟ **Accessibility**	
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
⊟ **Appearance**	
BackColor	☐ Control
BackgroundImage	☐ (none)
BackgroundImageLayout	Tile
Cursor	Default
⊞ Font	Microsoft Sans Serif, 8.25pt
ForeColor	■ ControlText
FormBorderStyle	Sizable
RightToLeft	No
RightToLeftLayout	False
Text	**Form1**

Text
The text associated with the control.

SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**My First Form**
TopMost	False
TransparencyKey	☐
UseWaitCursor	False
WindowState	Normal

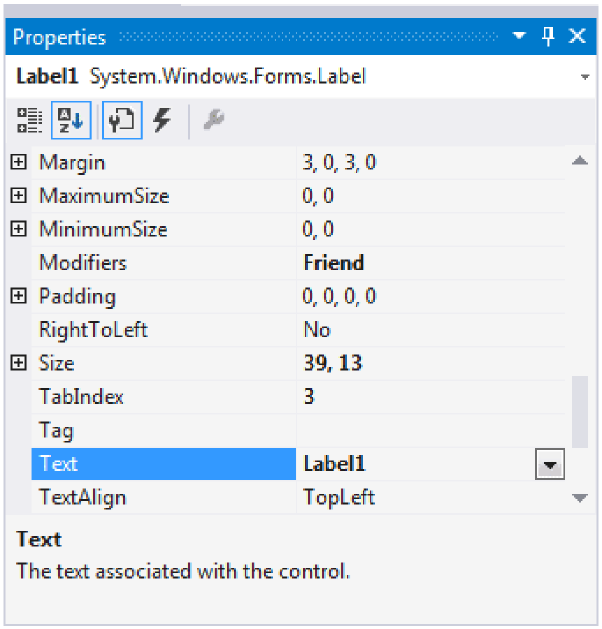
- **Click back on the form itself (the one with the labels and textboxes), or hit the return key on your keyboard**
- **The words "My First Form" will appear as white text on a blue background at the top of the form**

When you've correctly changed the Text property, your Form will then look like this one:



Change the Text properties of the labels, and the Text properties of the Textboxes.

Click on Label1 so that it has the sizing handles, and is therefore selected. Examine the Property box for the Label:



Label1 System.Windows.Forms.Label

Margin	3, 0, 3, 0
MaximumSize	0, 0
MinimumSize	0, 0
Modifiers	**Friend**
Padding	0, 0, 0, 0
RightToLeft	No
Size	**39, 13**
TabIndex	**3**
Tag	
Text	**Label1**
TextAlign	TopLeft

Text
The text associated with the control.

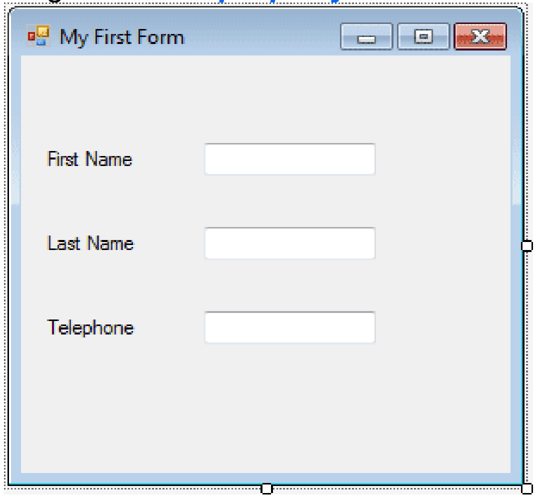
The Label control has quite a few different properties to the Form control. But the Label has a lot of properties that are the same. adds text to label steps:

1. ***With label1 selected, click inside the area next to "Text", and delete the word "Label1" by hitting the backspace key on your keyboard***
2. ***Type in the words "First Name"***
3. ***Click back onto the grey form, or hit the return key on your keyboard***
4. ***Label1 has now changed its text caption to read "First Name"***

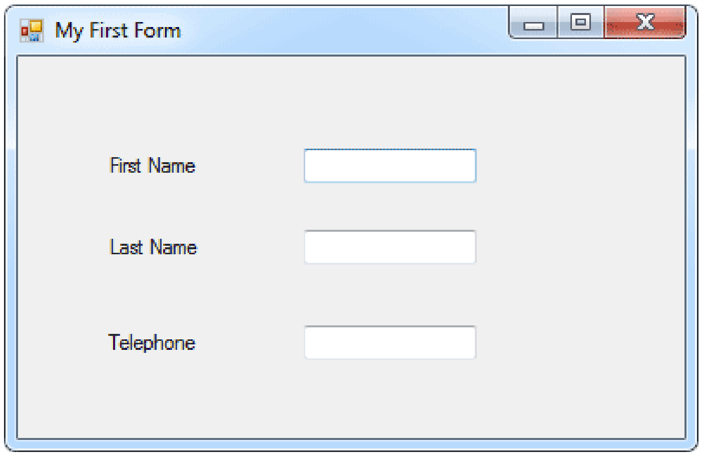
Now, change the Text property of the other two labels. Change them to these values:

Label2: Last Name
Label3: Telephone Number

With the three Textboxes , change the ***Text property to ""***. The form should look like below:



The Form can be resized just like the Label and the textboxes.
Form's edges has friends the sizing handles. To make the form bigger, just stretch.
Reposition and resize the textboxes and labels so that things don't look too squashed. Your form might look like this one:



Adding a Splash of Color to a VB Form

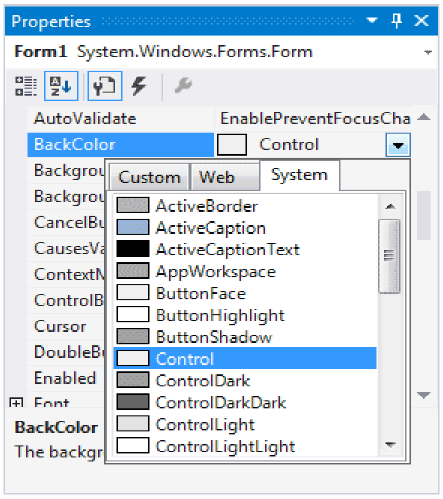
At the moment, the form looks a little bland.
Changing the color of the Form means changing one of its properties - the BackColor property.
So click anywhere on the form that is not a textbox or a label.

The Property Box on the right will read "Form1 or any name entered", and that indicates that you have indeed selected the form. When the Form is selected you can change its properties.

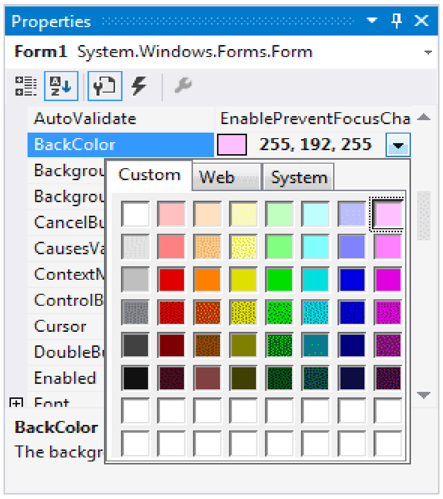
To change the color of the Form, click the word "BackColor" in the Property Box. Next, click the black down-pointing arrow to the right. A drop-down box will appear.

The default color is the one selected - Control. This is on the System Tab (left image). The System colors are to set whatever color scheme the user has opted for when setting up their computers.

To choose a color that is not a System color, click the Custom Tab(right image).



```
Properties
Form1  System.Windows.Forms.Form
AutoValidate      EnablePreventFocusCha
BackColor         Control
Backgrou   Custom | Web | System
Backgrou   ActiveBorder
CancelBu   ActiveCaption
CausesVa   ActiveCaptionText
ContextN   AppWorkspace
ControlB   ButtonFace
Cursor     ButtonHighlight
DoubleBu   ButtonShadow
Enabled    Control
Font       ControlDark
BackColor  ControlDarkDark
The backgr ControlLight
           ControlLightLight
```



```
Properties
Form1  System.Windows.Forms.Form
AutoValidate      EnablePreventFocusCha
BackColor         255, 192, 255
Backgrou   Custom | Web | System
Backgrou
CancelBu
CausesVa
ContextN
ControlB
Cursor
DoubleBu
Enabled
Font
BackColor
The backgr
```

select the Web Tab. to see a list of Web-Safe colors to choose from. A Web-Safe color is one that displays correctly in a web browser, regardless of which computer being used. (might want to use a Web-Safe color if designing a project for the internet)

To change the color of the labels, click on a label to select it. Look in the Property box to see if it reads Label. change the BackColor property of the Label in exactly the same way that we changed the BackColor property for the Form.

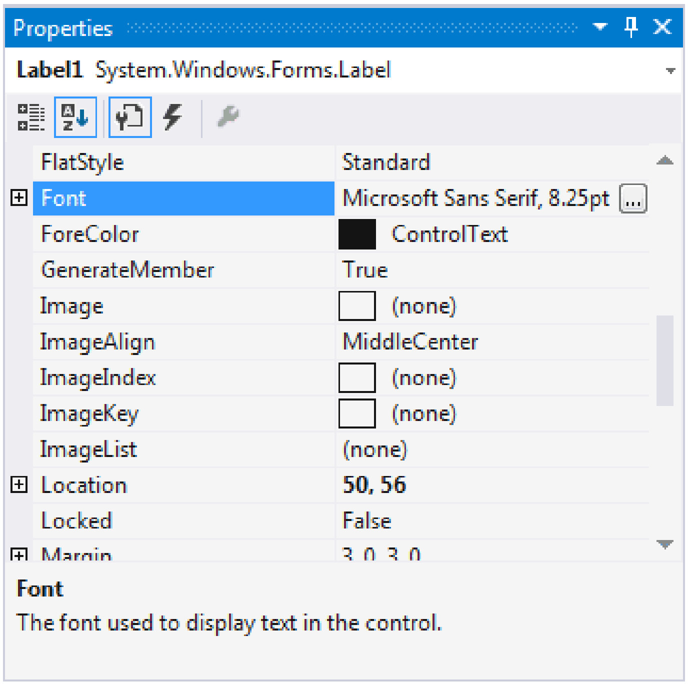
To change the color of more than one Label at a time, click on one Label to select it. Now, hold down the "Ctrl" key on the keyboard and click another Label. the two Labels now have sizing handles around them. Click the third Label with the "Ctrl" key held down, and all three Labels will be selected. change the BackColor property of all three at once.

Adding a Font effects to a VB Form

to change the Font size of the Labels and Textboxes, select a control. Let's start with Label1.

1. ***So click on Label 1***
2. ***Scroll down the Property Box until you see Font***
3. ***Click on the word "Font" to highlight it***
4. ***MS Sans Serif is the default Font***

Notice that the Font property has a cross next to it. This indicates that the property is expandable. Click the cross to see the following:



Properties

Label1 System.Windows.Forms.Label

FlatStyle	Standard
⊞ Font	Microsoft Sans Serif, 8.25pt
ForeColor	■ ControlText
GenerateMember	True
Image	□ (none)
ImageAlign	MiddleCenter
ImageIndex	□ (none)
ImageKey	□ (none)
ImageList	(none)
⊞ Location	**50, 56**
Locked	False
⊞ Margin	3, 0, 3, 0

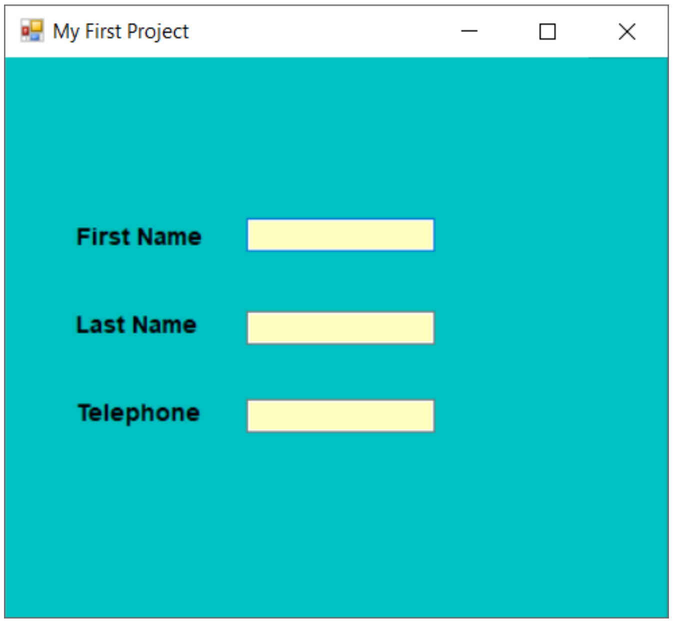
Font
The font used to display text in the control.

Change a lot of Font properties from here: the Name of the font, its Size, whether is should be Bold or not, etc.
Also click the square box with the three dots in it. This brings up a dialogue box where can change the font properties in the same place.

Make the following changes to the three labels:

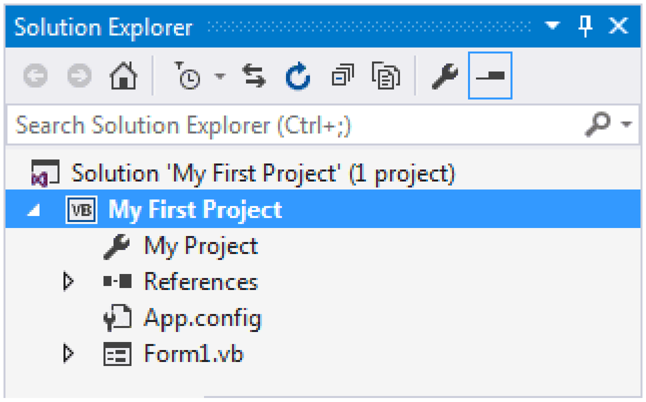
- ***Font: Arial***
- ***Font Style: Bold***
- ***Font Size: 10***

Change the Font of the three Textboxes so that they are the same as the Labels.



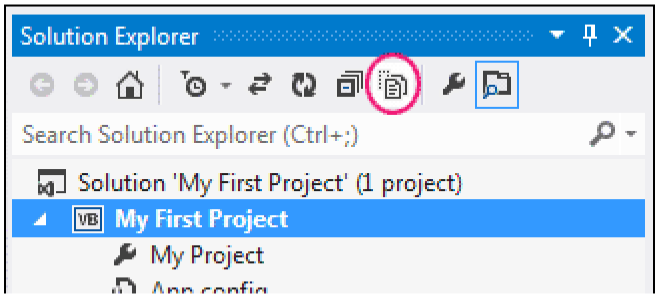
Save VB .NET Projects

The Solution Explorer shown in the top right of the Design Environment,. (If it is not shown, click View > Solution Explorer from the menu at the top.)

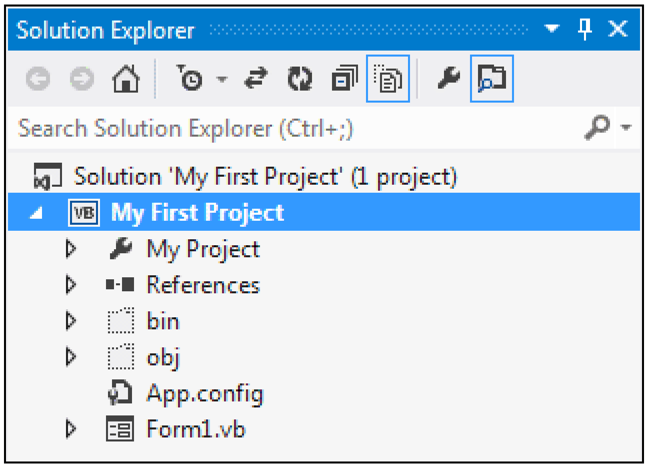

Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'My First Project' (1 project)
My First Project
My Project
References
App.config
Form1.vb

The Solution Explorer shows all the files in the project (Notice that the name of the project is at the top of the tree - My First Project).

At first glance, it looks as though there are not many files in the project. But click the Show All Files icon, circled below:


Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'My First Project' (1 project)
My First Project
My Project

Click Show All Files, the Solution Explorer will look something like this:



Save project, saving all these files.

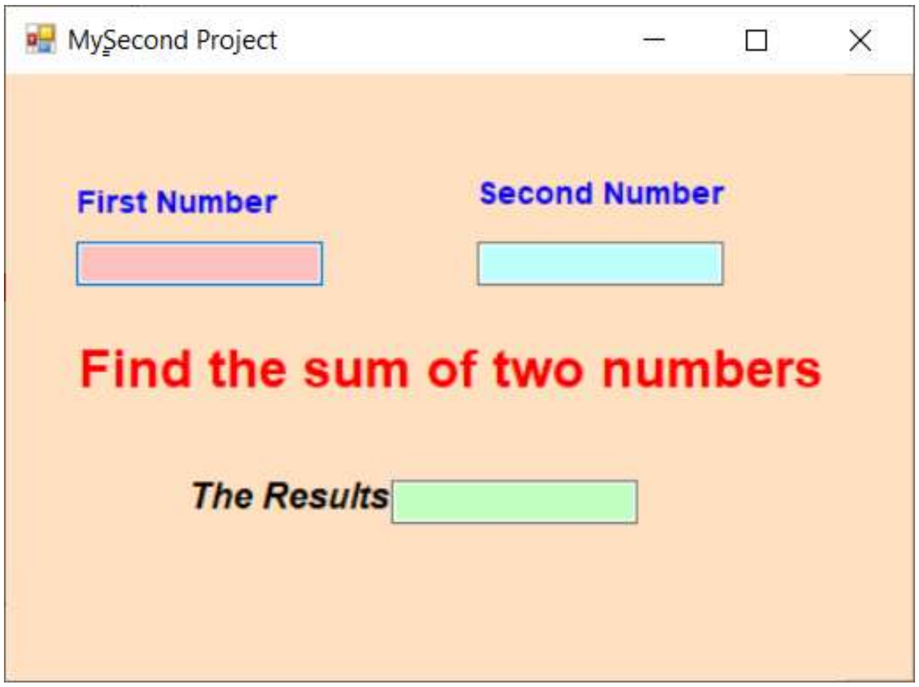
To save work,

- click File > Save All.
- press Ctrl + Shift + S on your keyboard
- click the icon in the Toolbar (the stack of floppy disks)

The files are then saved in the Document folder on the computer, inside of a folder called Visual Studio 2022 (or whatever version you have).

Practical activity

Create a Visual Basic project that includes the following Form:



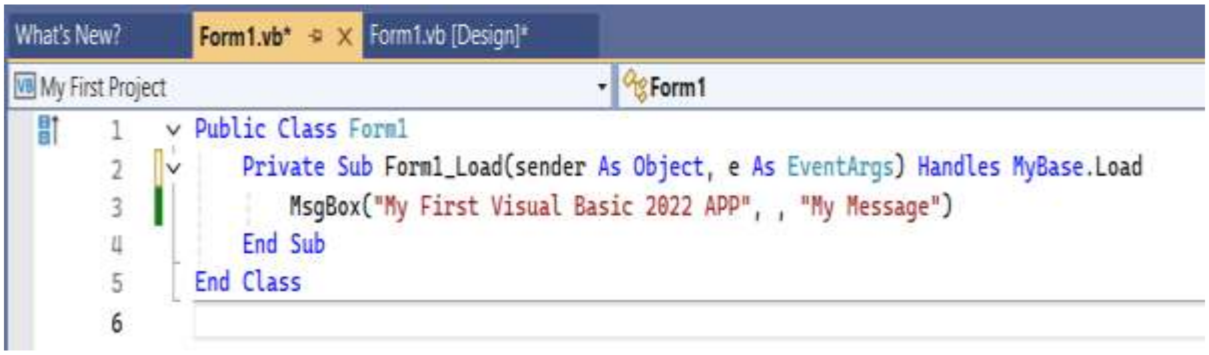
Writing the Code in VB Project

Writing a code for the controls enabling them to interact with the events triggered by the users. To write Visual Basic 2022 code, must delve into the concept of event-driven programming.

To write code, click on any part of the form to enter the code window. This is the structure of an event procedure. In this case, the event procedure is loading Form1. It starts with ***Private Sub*** and ends with ***End Sub***.

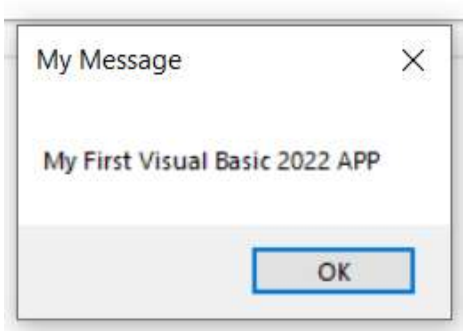
This procedure includes the **Form1 class** and the **event Load**, and they are bound together with an underscore, i.e. **Form_Load**. It does nothing other than loading an empty form.

To make the load event does something (message to user) , write the following code:



```
Public Class Form1
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        MsgBox("My First Visual Basic 2022 APP", , "My Message")
    End Sub
End Class
```

When the program was run, a message box that displays the text “My First Visual Basic 2022 Program” will appear



```
My First Visual Basic 2022 APP
```

MsgBox is a built-in function in Visual Basic that displays a message in a pop-up message box.

Notice that above Private Sub structure there is a preceding keyword Public Class Form1. This is the concept of an object-oriented programming language. When we start a windows application in Visual Basic 2022, we will see a default form with the name Form1 appears in the IDE, it is actually the Form1 Class that inherits from the Form class System.Windows.Forms.Form. A class has events as it creates an instance of a class or an object.

The Concept of Event-Driven Programming

Visual Basic 2022 is an event-driven programming language, meaning that the code is executed in response to events triggered by the user like clicking the mouse or pressing a key on the keyboard. Some other events are selecting an item from a drop-down list, typing some words into a text box and more. It may also be an event that responds to some other events.

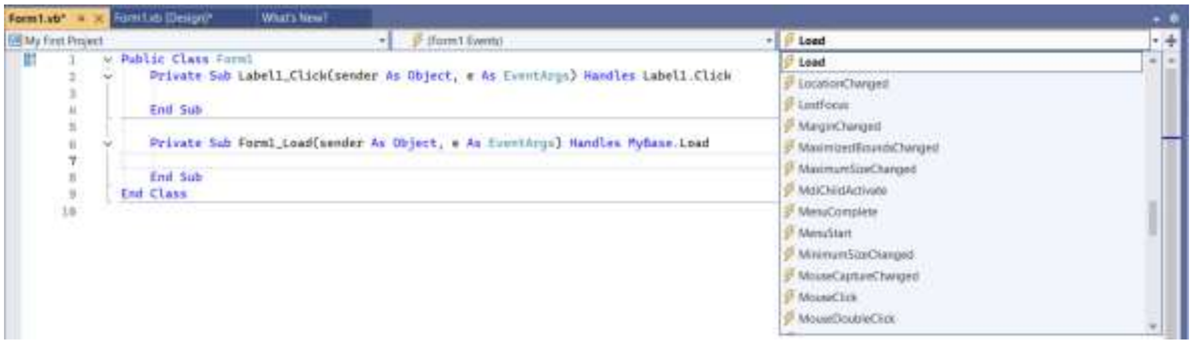
Some of the common events in Visual Basic 2022 are load, click, double-click, drag-drop, keypress and more.

Every control placed on the form has a set of events associate with it.

To view the events, double-click the control on the form to enter the code window. The default event will appear at the top right side of the code window.

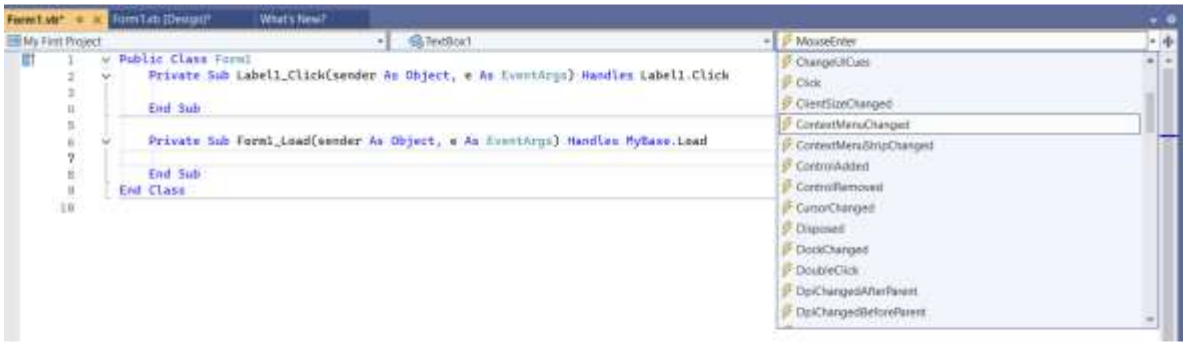
It must be click the default event to view other events associated with the control.

- the event procedure associated with the Form



```
Public Class Form1
    Private Sub Label1_Click(sender As Object, e As EventArgs) Handles Label1.Click

    End Sub

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load

    End Sub
End Class
```

- the events associated with the TextBox.



```
Public Class Form1
    Private Sub Label1_Click(sender As Object, e As EventArgs) Handles Label1.Click

    End Sub

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load

    End Sub
End Class
```

Example : Arithmetic Calculations

Write a code that perform arithmetic calculation

```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
 MsgBox("2" & "+" & "5" & "=" & 2 + 5)
End Sub
```

The symbol & (ampersand) is to perform string concatenation.

Run the code, it will perform the calculation and display the result in a message box

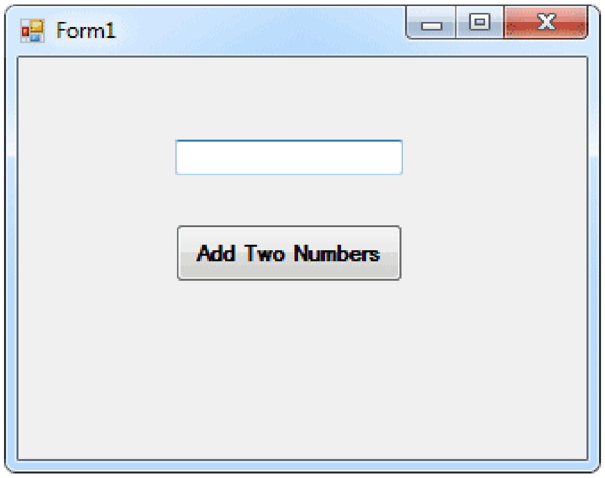
Adding a Button to a Form

Double clicking the Button tool in the toolbox to add the control to the form. The Button control, just like all the other controls we've seen so far, has a list of properties. One of these properties is the Text property. At the moment, the button will say "Button 1". To change that to anything:

- Click on the Button to highlight it
- Click on Text in the Property Box
- Click in the box next to the word "Text"
- Delete the word "Button 1", and Type "Add two numbers"
- Click back on the Form

Now add a Textbox to form using one of the methods outlined (either double-click, or draw).

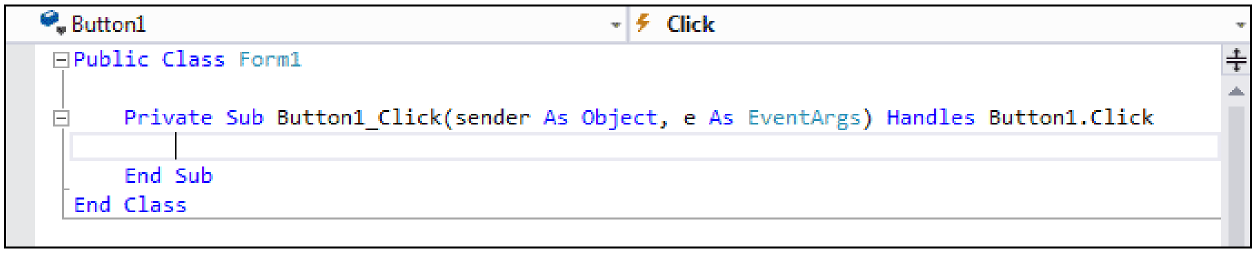
Your Form should now look something like this:



The Font property of the Button has also been changed. The Text for the Textbox control has had its default Text (Textbox 1) deleted.

VB Button Code

To get look at the code window, double click the Button control. The code window will appear, and will look something like this:



```
Public Class Form1

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click

    End Sub
End Class
```

The part to concentrate on for the moment is where cursor is flashing on and off. Because the Button control was double-clicked, the cursor will be flashing between the lines Private Sub … and End Sub.

```
Private Sub Button1_Click(sender As Object, e As EventArgs) _Handles Button1.Click
Textbox1.Text = 2+5
End Sub
```

The part of the code interested in is highlighted in red in the code above. Notice, too, that the underscore character (_) has been used to spread the code over more than one line if it becomes too long:

Private : Private means that no other part of the program can see this code except for our button

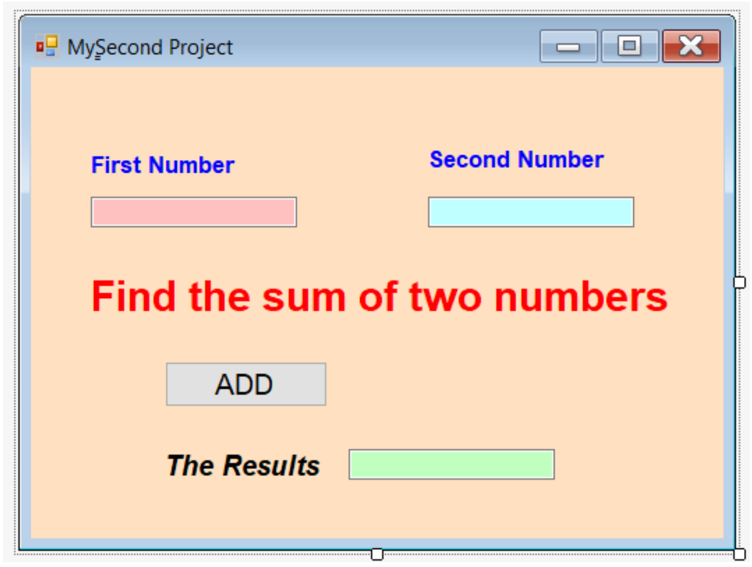
Sub : Short for Subroutine. The "Sub" word tells VB that some code follows, and that it needs to be executed

Button1 : This is the name of the button (the Name property of the control is the important one). If the Name property was changed, VB will change this button name for (when changed the Text property just erased the word "Button1" in caption not the name property)

_Click ( ): This is an Event. In other words, when the button is clicked, the Click Event will fire, and the code will be executed

End Sub: The subroutine ends right here. This signifies the end of the code

Example : ***Create a Visual Basic project that includes the following Form Then write the addition code for the ADD Button***



In this form, add three TextBox controls.

- The current value of the Name property are (Textbox1, Textbox2. Textbox3), Change the Name property to (First_num, Second_num, Result_num)
- The Text property is Textbox1, Textbox2. Textbox3. Scroll down and locate the Text property. Delete the default text, and just leave it blank.

Add Four Labels : change the Text as in the Form (First Number, Second Number, Fined the sum of two numbers , The Results)

Add a button to the Form. Change the Text of the button to ADD. Next, click on the button and enter the following code:

```
Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
 Result_num.Text = Val(First_num.Text) + Val(Second_num.Text)
End Sub
```

This program will add the value in the two TextBoxes and displays the sum in the third TextBox .

Practical activity

Replace TextBox3 with a new Label to show the result in the form design and in the Button1 code.

Data Types

In our daily life, we have to deal with all kinds of information and data. For example, we need to deal with data like names, money, phone number, addresses, date, stock quotes and more.

Similarly, in Visual Basic 2022, we have to deal with all sorts of data, some of them can be mathematically calculated while some are in the form of text or other non-numeric forms.

In Visual Basic 2022, data can be stored as variables, constants or arrays. The values of data stored as variables always change, just like the contents of a mailbox or the storage bin while the value of a constant remains the same throughout.

Visual Basic 2022 classifies information into two major data types, the numeric data types, and the non-numeric data type

Numeric Data Types

In Visual Basic 2022, numeric data types are types of data comprises numbers that can be calculated using various standard arithmetic operators.

Examples of numeric data types are examination marks, height, body weight, the number of students in a class, share values, the price of goods, monthly bills, fees, bus fares and more.

In Visual Basic, numeric data are classified into seven types based on the range of values they can store. Numeric data that involve round figures are classified as Integer or Long integer.

Data that require high precision calculation are classified as single and double precision data types, they are also called floating point numbers.

Numeric data that involve money are classified as currency data types. Lastly, data that require more precision and involve many decimal points are classified as decimal data types.

Non-numeric Data Types

In Visual Basic 2022, non-numeric data types are data that cannot be calculated using standard arithmetic operators. The non-numeric data comprises text or string data types, the Date data types, the Boolean data types that store only two values (true or false), Object data type and Variant data type.

Numeric Data Types	
Type	**Storage**
Byte	1 byte
Integer	2 bytes
Long	4 bytes
Single	4 bytes
Double	8 bytes
Currency	8 bytes
Decimal	12 bytes

Non-numeric Data Types

Type	**Storage**	**Range**
String(fixed length)	Length of string	1 to 65,400 characters
String(variable length)	Length + 10 bytes	0 to 2 billion characters
Date	8 bytes	January 1, 100 to December 31, 9999
Boolean	2 bytes	True or False
Object	4 bytes	Any embedded object
Variant(numeric)	16 bytes	Any value as large as Double
Variant(text)	Length+22 bytes	Same as variable-length string

Dealing with Variables and Constants

With Visual Basic, and most programming languages, what you are doing is storing things in the computer's memory, and manipulating this store.

If want to add two numbers together, put the numbers into storage areas and "tell" Visual Basic to add them up. But it can't do this without variables. So a variable is a storage area of the computer's memory.

Think of it like this: *a variable is an empty* ***cardboard box****. Each empty cardboard box is a single variable. To add two numbers together, write the first number on a piece of paper and put the piece of paper into an empty box. Write the second number on a piece of paper and put this second piece of paper in a different cardboard box.*

Now, out of all your thousands of empty cardboard boxes two of them contain pieces of paper with numbers on them. To help you remember which of the thousands of boxes hold your numbers, put a sticky label on each of the two boxes. Write "number1" on the first sticky label, and "number2" on the second label.

To declare the data and how to store them. In Visual Basic 2022, data can be stored as variables or constants. Variables are like mailboxes in the post office. The content of the variable changes every now and then, just like the mailboxes. In Visual Basic 2022, variables are the specific areas allocated by the computer memory to store data.

Variable Names

Like the mailboxes, you must assign a name to the variable by following a set of rules as follows:

- It must be less than 255 characters
- No spacing is allowed
- It must not begin with a number
- Period is not permitted

Some examples of valid and invalid variable names are listed in Table

Valid Names	Invalid Name
My_Name	My.Name
vb2022	2022vb
Long_Name_Can_beUSE	LongName&Canbe&Use *& is not acceptable

Declaring Variables in VB .NET

In Visual Basic, the variables was declared before use them. To declare a variable, assign a name to the variable and state its data type. If fail to do so, the program will run into an error.

Variables are usually declared in the ***general section*** of the code windows using the **Dim** keyword. The syntax to declare a variable in VB is as follows:

```
Dim VariableName As DataType
```

To declare more variables:

- Declare them in separate lines

```
Dim VariableName1 As DataType1
Dim VariableName2 As DataType2
Dim VariableName3 As DataType3
```

- Combine them in one line, separating each variable with a comma, as follows:

```
Dim VariableName1 As DataType1, VariableName2 As
DataType2, VariableName3 As DataType3
```

Now examine this:

```
Dim number1 As Integer

number1 = 3
```

It's VB's way of setting up (or declaring) variables. Here's a breakdown of the variable Declaration:

Dim: Short for Dimension. It's a type of variable. You declare (or "tell" Visual Basic) that you are setting up a variable with this word. (other types of variables are declare later).

number1: This is a variable (storage area), After the Dim word, Visual Basic is looking for the name of the variable. the variable name is anything but there are a few reserved words that VB won't allow. It's good practice to give the variables a name appropriate to what is going in the variable.(*This is the cardboard box and the sticky label all in one*)

As Integer: telling Visual Basic that the variable is going to be a number (integer).

number1 = 3: The = sign means assign a value. In other words, putting something in the variable. telling Visual Basic to assign a value of 3 to the variable called number1.

Example1 : Creating a New VB Project

Click **File > New Project** from the menu bars. (2015 and 2017 users will get this dialog box):

In 2019 and greater users will see this instead:

Then click Next to get to the configuration screen:

Type **Variables** as the Project Name. (can change Location to save the project elsewhere). When click the OK button, a new form will appear.

look at the Solution Explorer at the top, see the name of the project has changed to the Name gave it (**Variables)**.

The name of the Project is now Variables - the same name as the folder that is created for you to hold all your project files.

Design form contain Button1 to add two numbers together(3 , 6) and displayed the result by using a TextBox1, the form is as:

- First, started with the Dim word, indicating to Visual Basic to set up a variable
- gave the variable a name (number1)
- told VB that what is going inside the variable is a number (As Integer)
- Two more variable were set up in the same way, number2 and answer

After setting up the three variables Told Visual Basic that:

- **what is going into the first variable was the number 3,**
- **what is going into the second variable was the number 6.**

To put something into a variable, use the equals = sign (an assignment operator)

```
number1 = 3
number2 = 6
```

The next part to add two numbers together, store the result in that other variable which is called answer

```
answer = number1 + number2
```

The final part of the program used Visual Basic's TextBox to show the sum answer

```
        Textbox1.Text = answer
```

Type the word **Textbox1**, then type a full stop, should see a drop-down box appear. This is a list of the Properties and Methods that the Textbox can use.

Scroll down until see the word "Text". Double click the Text property and the drop-down box will disappear.

The Text property have chosen is the same Text property that set from the Properties Window earlier. Here, setting the property with code; before, set it at design time.

The code window should then look like this:

Exercise1 : **Add two TextBox to the form, Delete the values replace them with the text values of TextBox**

Exercise2 : **Delete the plus sign in between number1 and number2, and replace them with each of the following in turn:**

- **(the minus sign)** -
- **(the multiplication sign in VB is the asterisk sign)** *
- **(the divide sign in VB is the forward slash)** /

Example2

This example declares two variables of the string type and concatenates them using the ampersand sign &. Create a project and insert a Button and name as ButtonConcat and change its text to Show Message. You can assign a value to the string using the = sign. Now enter the following code:

```
Private Sub ButtonConcat_Click(sender As Object, e As
EventArgs) Handles ButtonConcat.Click
    Dim YourName As String = " Gasim"
    Dim MyMsg As String
    MyMsg = "Happy Birthday!"
    MsgBox(MyMsg & YourName)
End Sub
```

Running the code and clicking the show message button produces the following message:

Example3

In this example, create a simple calculator where the user enter his name, two numbers and click the Calculate button to compute the addition of the two numbers. It also display the date.

There are three data types here, String, Integer and Date. We also use the Format function to display the current date of computation.

```
Private Sub ButtonCal_Click(sender As Object, e As
EventArgs) Handles ButtonCal.Click
    Dim YourName As String
    Dim Num1 As Integer
    Dim Num2 As Single
    Dim Sum As Integer
    Dim RunDate As Date
      YourName = TxtName.Text
      Num1 = Val(TxtNum1.Text)
      Num2 = Val(TxtNum2.Text)
      Sum = Num1 + Num2
      LabelSum.Text = Str(Sum)
      RunDate = Format(Now, "General Date")
      LabelDate.Text = RunDate
End Sub
```

Running the program and clicking on the Calculate button produces the following output as shown in Figure

Assigning Values to Variables Examples

The variable can be a declared variable or a control property value. The expression could be a mathematical expression, a number, a string, a Boolean value (true or false) and more, as illustrated in the following examples:

```
1. firstNumber = 100

2. secondNumber = firstNumber - 99

3. username = ″John Lyan″

4. userpass.Text = password

5. Label1.Visible = True

6. Command1.Visible = False

7. Label4.text = textbox1.Text
```

```
8. ThirdNumber = Val(usernum1.Text)

9. total = firstNumber + (secondNumber * ThirdNumber)

10.   X = sqr(16)

11.   TrimString = Ltrim (" Visual Basic", 4)

12.   Num = Int(Rnd * 6)+ 1
```

An error occurs when try to assign a value to a variable of incompatible data type. For example, if declared a variable as an integer but assigned a string value to it, an error occurred, the following error messages will appear in a dialog box:

Scope of Declaration

In Visual Basic usually use the **Dim** keyword to declare the data. However, can also use other keywords to declare the data. Three other keywords are **Private, Static** and **Public,** The keywords indicate the scope of the declaration. The forms are as shown below:

```
Private VariableName as Datatype
Static VariableName as Datatype
Public VariableName as Datatype
```

Private : declares a local variable or a variable that is local to a procedure or module. However, Private is rarely used, normally use Dim to declare a local variable.

Static : declares a variable that is being used multiple times, even after a procedure has been terminated. Most variables created inside a procedure are discarded by Visual Basic when the procedure is terminated. Static keyword preserves the value of a variable even after the procedure is terminated.

Public : declares a global variable, which means it can be used by all the procedures and modules of the whole VB program.

Declaring Constants

Constants are different from variables in the sense that their values do not change during the running of the program. The syntax to declare a constant in is

```
Const Constant Name As Data Type = Value
```

Example4

Create a VB Button1 code to calculate the area of a circle using variables and Pi constant values. Show the area results using MsgBox

```
Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
Const Pi As Single = 3.142
Dim R As Single = 10
Dim AreaCircle As Single
AreaCircle = Pi * R ^ 2
MsgBox("Area of circle with " & "radius" & R & "=" &
AreaCircle)
End Sub
```

Exercise3: **Design VB form with code to enter first and last name of user and get the full name as in the following figure**

Exercise4: **Design VB form as :**

Type the following code for your Button

```
Dim testNumber As Short

testNumber = Val( txtNumbers.Text )

MessageBox.Show( testNumber )
```

Type of variable declared - As Short. This means "Short Integer". Run the program. While it's running, do the following:

1. **Enter the number 1 into the textbox, and click the Answers button**
2. **Enter the number 12 to the textbox and click the Button**
3. **Enter the number 123 to the textbox and click the Button**
4. **Keeping adding numbers one at a time, then clicking the button**

How many numbers did you get in the textbox before the following error message was displayed? (Click Break to get rid of it.)

Or this style of error message in later versions of Visual Studio:

Then change the Single Type to As Double. Use Answers button, if error message shown or not?

Mathematical Operations

The list of Visual Basic arithmetic operators are shown in Table below:

Operator	Mathematical Function	Example
+	Addition	1+2=3
-	Subtraction	10-4=6
^	Exponential	3^2=9
*	Multiplication	5*6=30
/	Division	21/7=3
Mod	Modulus(returns the remainder of an integer division)	15 Mod 4=3
\	Integer Division(discards the decimal places)	19/4=4

String Manipulation

- Using the & sign and the + sign, both perform the string concatenation which means combining two or more smaller strings into larger strings
```
str = str1 + str2 + str3
str = str1 & str2 & str3
```

- The Len function returns an integer value which is the length of a phrase or a sentence, including the empty spaces. The syntax is
```
Len (Visual Basic 2022) = 17
```

- The Right function extracts the right portion of a phrase. The syntax is
```
Dim MyText As String
MyText = "Visual Basic"
TextBox1 = Microsoft.VisualBasic.Right(MyText, 4)
```

returns four rightmost characters of the phrase entered into the textbox : asic

- The Left function extracts the left portion of a phrase. The syntax is
```
TextBox1 = Microsoft.VisualBasic.Left(MyText, 4)
```

- The Mid function is used to extract a portion of text from a given phrase. The syntax of the Mid Function is

```
LblExtract.Text = Mid(myPhrase, position,n)
```

- The Trim function trims the empty spaces on both sides of the phrase. The syntax is

```
Trim (" Visual Basic ") = Visual basic
```

  The Ltrim function trims the empty spaces of the left portion of the phrase.

  The Rtrim function trims the empty spaces of the right portion of the phrase.

- The InStr function looks for a phrase that is embedded within the original phrase and returns the starting position of the embedded phrase. The syntax is

```
Instr(1, "Visual Basic 2022 ","Basic")=8
```

- The Ucase function converts all the characters of a string to capital letters. On the other hand, the Lcase function converts all the characters of a string to small letters. The syntaxes are

```
Microsoft.VisualBasic.Ucase("Visual Basic") =VISUAL BASIC
Microsoft.VisualBasic.Lcase("Visual Basic") =visual basic
```

- The Chr function returns the string that corresponds to an ASCII code while the Asc function converts an ASCII character or symbol to the corresponding ASCII code. ASCII stands for "American Standard Code for Information Interchange". Altogether there are 255 ASCII codes and as many ASCII characters. The syntax of the Chr function is

```
Chr(charcode)
```

  and the syntax of the Asc function is

```
Asc(Character)
          Chr(65)=A,     Chr(122)=z,      Chr(37)=% ,
          Asc("B")=66,   Asc("&")=38
```

Example: A Calculator Project in VB NET

So create a new project, call it **Calculator,** design the form first. What does a calculator need?

- Numbers10 button for the numbers 0 to 9.
- A display area for the result.
- A plus sign button,
- An equals sign button,
- A clear the display button.

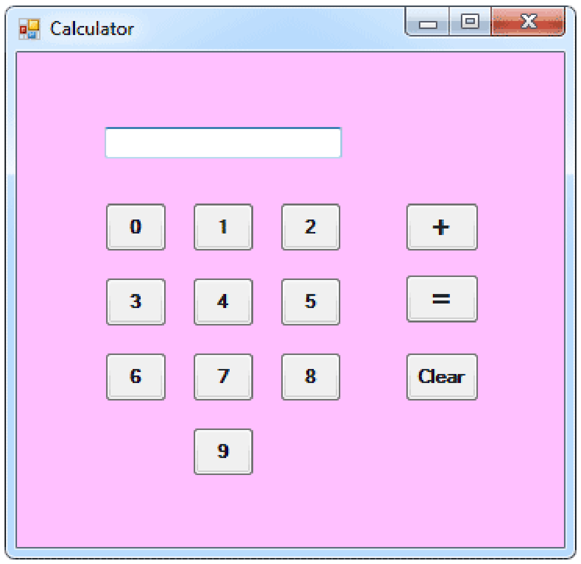
First add ten Buttons (You can add one, then copy and paste the rest). All transferring the Text Properties for the Buttons to the 0 to 9. The number buttons don't do anything else of add text property to a Textbox1.
Three more buttons need to be added

Plus Button
Name: btnPlus
Font: MS Sans Serif, Bold, 14
Text: +

Equals Button
Name: btnEquals
Font MS Sans Serif, Bold, 14
Text: =

Clear Button
Name: btnClear
Font MS Sans Serif, Bold, 14
Text: Clear

When your form design is finished, it might look something like this:



set up two Integer variables there, total1 and total2:

```
Public Class Form1
    Dim total1 As Single
    Dim total2 As Single

```

All the controls on your form can see these two variables. Those Buttons you set up can put something in them, and every button has the ability to see what's inside them.

The Buttons with the Text 0 to 9 only need to do one thing when the button is : clicked - have their Text Properties transferred to the Textbox.

```
Private Sub buttonZero_Click(sender As Object, e As
EventArgs) Handles buttonZero.Click
     txtDisplay.Text = txtDisplay.Text & buttonZero.Text
End Sub
```

When the programme is running, click the 0 button to see that it does indeed transfer the Text on the Button to the textbox

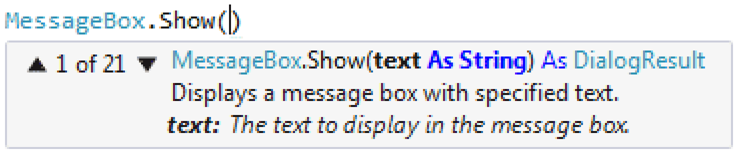
```
Private Sub btnPlus_Click(sender As Object, e As EventArgs)
Handles btnPlus.Click
     total1 = total1 + Val(txtDisplay.Text)
     txtDisplay.Clear()
End Sub

Private Sub btnEquals_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnEquals.Click
     total2 = total1 + Val(txtDisplay.Text)
     txtDisplay.Text = total2
     total1 = 0
End Sub

Private Sub btnClear_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnClear.Click
      txtDisplay.Clear()
End Sub
```

Exercise: Add another operation to the calculator form.

The Message Box In VB .Net

Whenever you have used the Message Box, you have done so like this:

```
MessageBox.Show( "Your Message Here" )
```



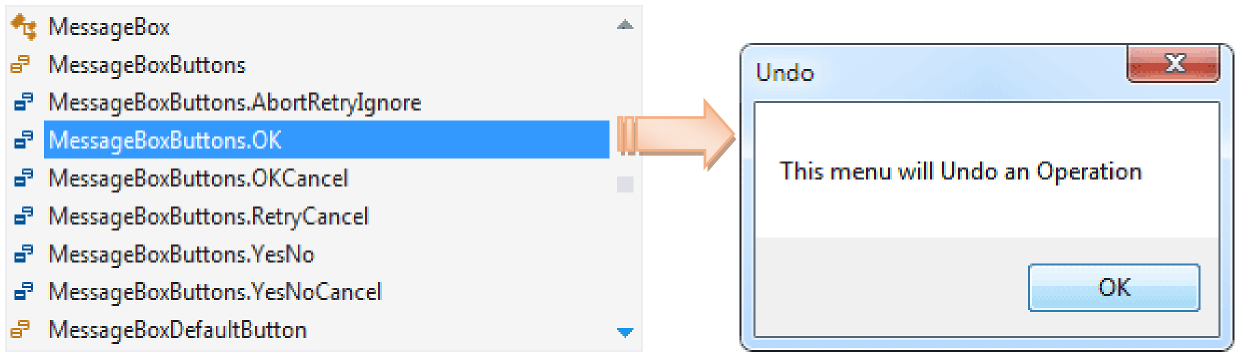
```
MessageBox.Show(|)
▲ 1 of 21 ▼ MessageBox.Show(text As String) As DialogResult
Displays a message box with specified text.
text: The text to display in the message box.
```

This is telling that there are 21 different ways to use the MessageBox function. This currently viewing the first of these (1 of 21). Click on the up and down arrows to scroll through then all.

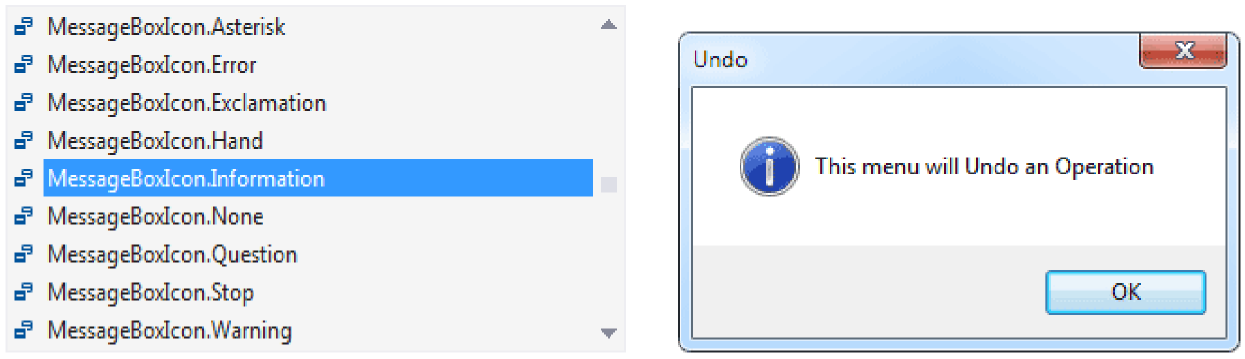
text As String: The text in question is the text that will appear for your message

If only need the OK button on your message boxes:

```
MessageBox.Show( "This menu will Undo an Operation", "Undo" )
```


MessageBox
MessageBoxButtons
MessageBoxButtons.AbortRetryIgnore
MessageBoxButtons.OK
MessageBoxButtons.OKCancel
MessageBoxButtons.RetryCancel
MessageBoxButtons.YesNo
MessageBoxButtons.YesNoCancel
MessageBoxDefaultButton
Undo
This menu will Undo an Operation
OK

Each option for message box is separated by a comma. Type a comma after the "Undo", and then a space, get another pop-up menu. On this menu, can specify which buttons want on message box:



Working with PictureBox

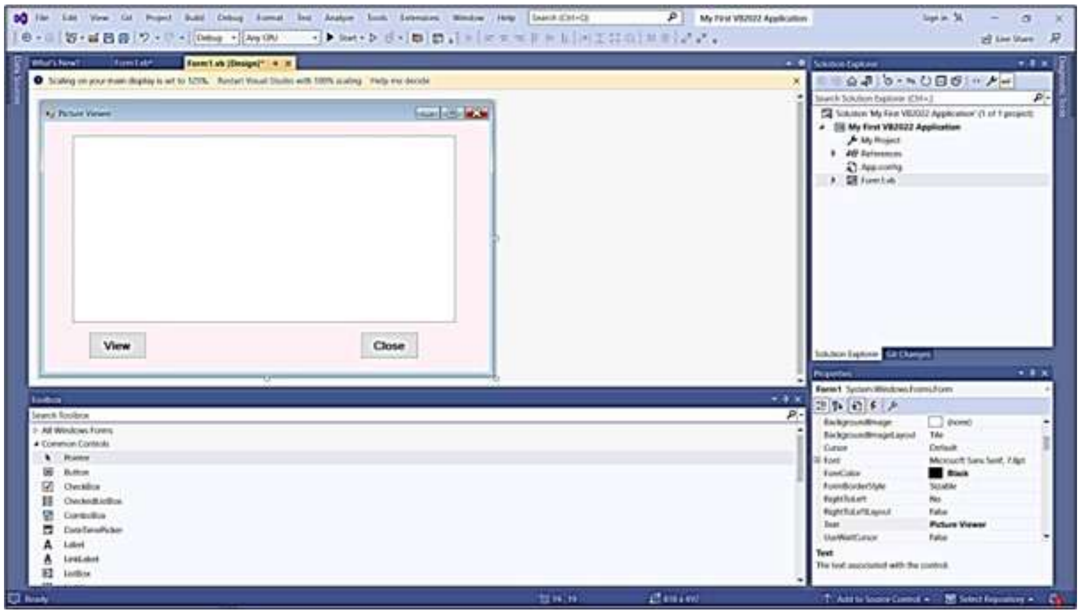
PictureBox is a control in Visual Basic that is used to display images.

Example : Designing the UI of a Picture Viewer

To demonstrate how to add the PictureBox controls and then change their properties, design a picture viewer.

- First, change the title of the default form to Picture Viewer in its properties window.
- Next, insert a picture box on the form change
    - ➢ its text property to Picture Viewer
    - ➢ its background color to white. To do this, right click the picture box and select properties in the popup menu, then change the background color using BackColor Property.
    - ➢ its border property to FixedSingle
    - ➢ the size mode of the image to stretchImage so that the image can fit in the PictureBox
- Finally, add two buttons to the form and change the text to View and Close in their respective properties' windows.

The picture viewer is not functional yet until code was wrote for responding to events triggered by the user.



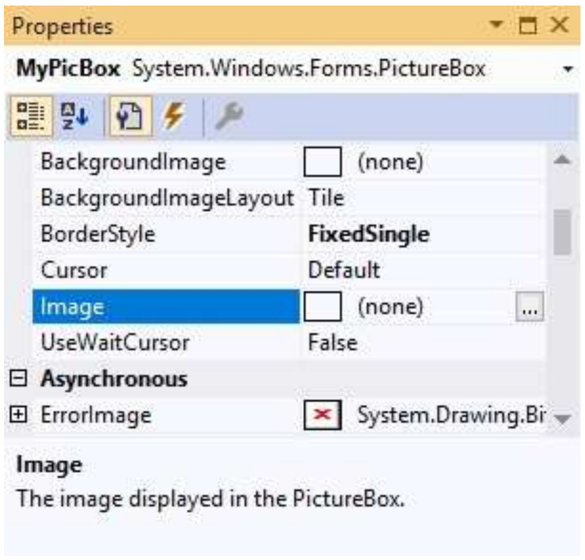
Form1 System.Windows.Forms.Form

BackgroundImage	(none)
BackgroundImageLayout	Tile
Cursor	Default
Font	Microsoft Sans Serif, 7.8pt
ForeColor	Black
FormBorderStyle	Sizable
RightToLeft	No
RightToLeftLayout	False
Text	Picture Viewer
UseWaitCursor	False

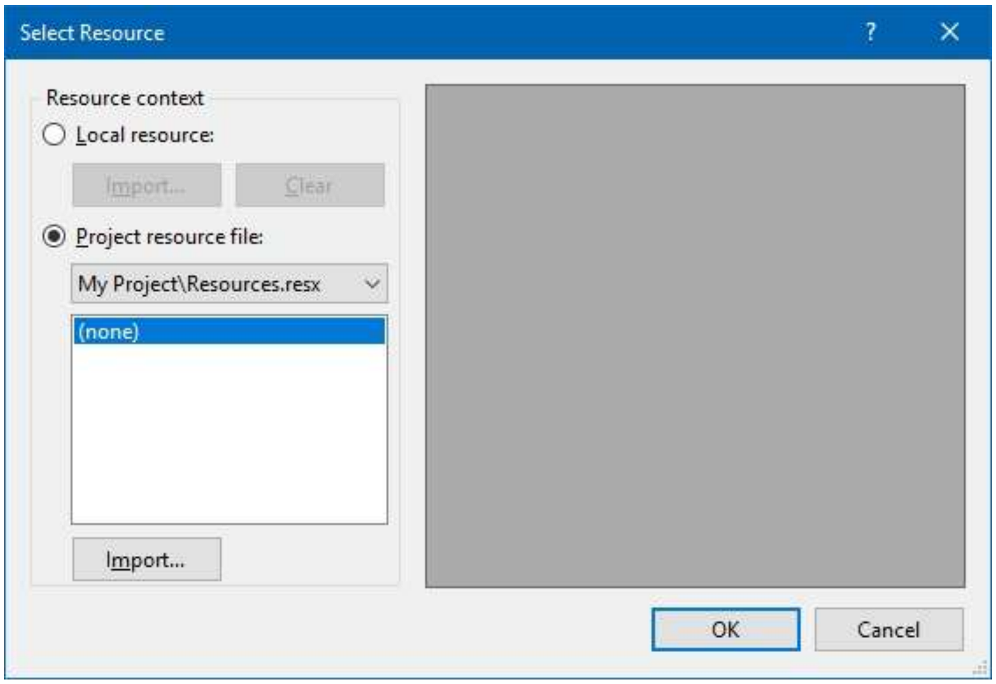
Text
The text associated with the control.

❖ **<u>Loading an Image at Design Time</u>**

After insert a PictureBox on the form find Image property, as shown in Figure


Properties
MyPicBox System.Windows.Forms.PictureBox
BackgroundImage (none)
BackgroundImageLayout Tile
BorderStyle FixedSingle
Cursor Default
Image (none) ...
UseWaitCursor False
Asynchronous
ErrorImage System.Drawing.Bi
Image
The image displayed in the PictureBox.

click on the grey button on the right of Image (…) to bring out the "Select Source" dialog , as shown in Figure



Now select local source and click on the Import button to bring up the Open dialog and view the available image files in local drives. Finally, select the image and then click the open button, the image will be displayed in the PictureBox.

❖ <u>Loading an Image at Runtime</u>

In Visual Basic an image can also be loaded at runtime using the **FromFile** method of the Image control, as shown in the following example.

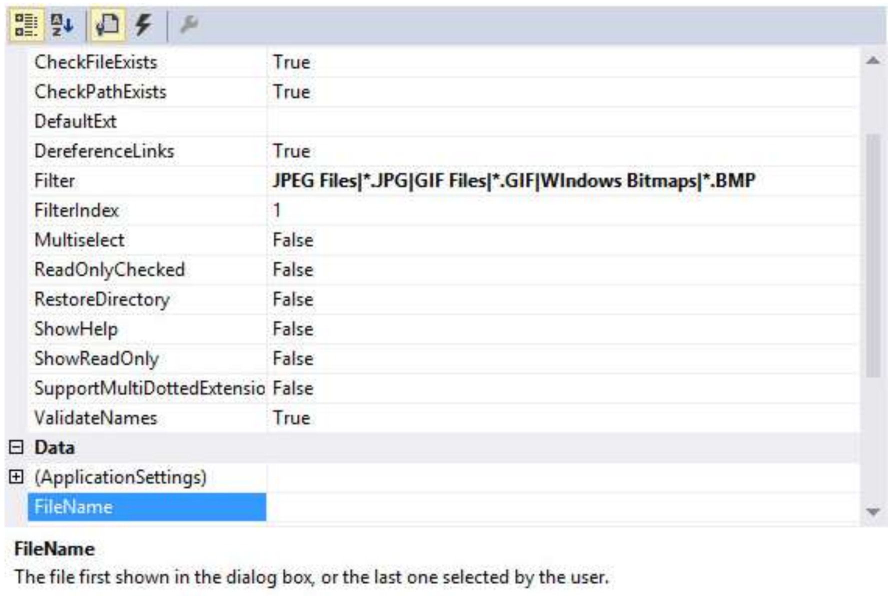
```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
MyPicBox.Image = Image.FromFile("C:\Users\Documents\image1.jpg")
End Sub
```

Loading an Image in a PictureBox using Open File Dialog Control

Write code so that the user can browse for the image files in his or her local drives then select a particular image and display it on the PictureBox at runtime.

- First, add a button and change its text to View and its name to ButtonView.
- Next, add the ***OpenFileDialog*** control on the form. This control will be invisible during runtime but it facilitates the process of launching a dialog box and let the user browse his or her local drives and then select and open a file.
- In order for the **OpenFileDialog** to display all types of image files, specify the types of image files under the **Filter property**.
- Besides that, delete the **default Filename property**.
- Next, right-click on the **OpenFileDialog** control to access its properties window. Beside the Filter property, specify the image files using the format:
  `JPEG Files| *.JPG|GIF Files|*.GIF|WIndows Bitmaps|*.BMP`

as shown in Figure



CheckFileExists	True
CheckPathExists	True
DefaultExt	
DereferenceLinks	True
Filter	**JPEG Files|*.JPG|GIF Files|*.GIF|WIndows Bitmaps|*.BMP**
FilterIndex	1
Multiselect	False
ReadOnlyChecked	False
RestoreDirectory	False
ShowHelp	False
ShowReadOnly	False
SupportMultiDottedExtensio	False
ValidateNames	True
Data	
(ApplicationSettings)	
FileName	

FileName
The file first shown in the dialog box, or the last one selected by the user.

Next, double-click on the **View** button and enter the following code:

```
Private Sub ButtonView_Click(sender As Object, e As EventArgs)
Handles ButtonView.Click

  If OpenFileDialog1.ShowDialog = DialogResult.OK Then
      MyPicBox.Image = Image.FromFile(OpenFileDialog1.FileName)
  End If
End Sub
```

Conditional Logic - If Statements

```
If condition Is True Then run Action
```

You're using conditional logic with those two words. Conditional logic is all about that little **If** word. You can even add Else to it.

```
If condition Is True Then run action1
    Else run action2
End If
```

Also we can use nested If as:

```
If condition1 Is True Then run action1
Else If condition2 Is True Then run action2 …
```

Start a new project. Give it any name you like. In the design environment, add a TextBox and Button to the new form. Double click the button and add the following code to it:

```
Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
    Dim firstname As String
    firstname = TextBox1.Text
    If firstname = "" Then
        MessageBox.Show("FirstName is null")
    Else
        MessageBox.Show("FirstName is" & firstname)
           End If
End Sub
```

Run the programme and see what happens.

Select Case Statements

The Select Case statement is another way to test what is inside of a variable. You can use it when you know there is only a limited number of things that could be in the variable.

Example1: Create a Form with a button, label and a Textbox on it. List the class of student when enter the class value in upper or lower case:

```
Private Sub Button1_Click(sender As Object, e As EventArgs) _
Handles Button1.Click
   Dim st_class As String
   Dim c As String
   c = TextBox1.Text
   Select Case c
```

```
            Case "a"
                st_class = "Class A"
            Case "A"
                st_class = "Class A"
            Case "b"
                st_class = "Class B"
            Case "B"
                st_class = "Class B"
            Case Else
                MessageBox.Show("enter class..")
        End Select
        Label1.Text = st_class
        End Sub
```

Example2: using a range of values in case condition, Determine Student grade estimates

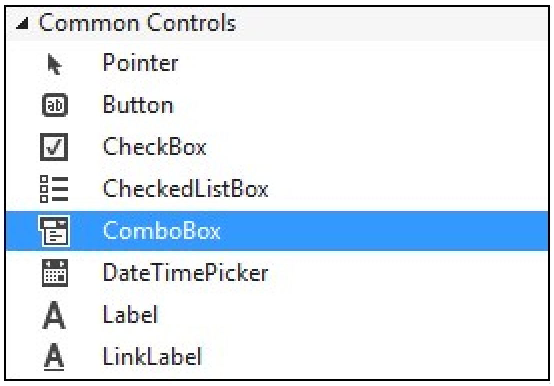
```
        Select Case agerange
            Case < 50
                MessageBox.Show(" راسب")
            Case 50 To 59
                MessageBox.Show("مقبول")
            Case 60 To 69
```

Add a new button to your form. Write a programme that tests if a student degree rating is :
a) Good b) Very Good c) Excellent d) none of the above.

```
        Private Sub Button1_Click(sender As Object, e As EventArgs)
        Handles Button1.Click
            Dim agerange As Integer
            agerange = TextBox1.Text
            Select Case agerange
                Case < 50
                    MessageBox.Show(" راسب")
                Case 50 To 59
                    MessageBox.Show("مقبول")
                Case 60 To 69
                    MessageBox.Show("متوسط")
                Case 70 To 79
                    MessageBox.Show("جيد")
                Case 80 To 89
                    MessageBox.Show("جيد جدا")
                Case 90 To 100
                    MessageBox.Show("امتياز")
                Case Else
                    MessageBox.Show("خاطئ ادخال")
            End Select
        End Sub
```

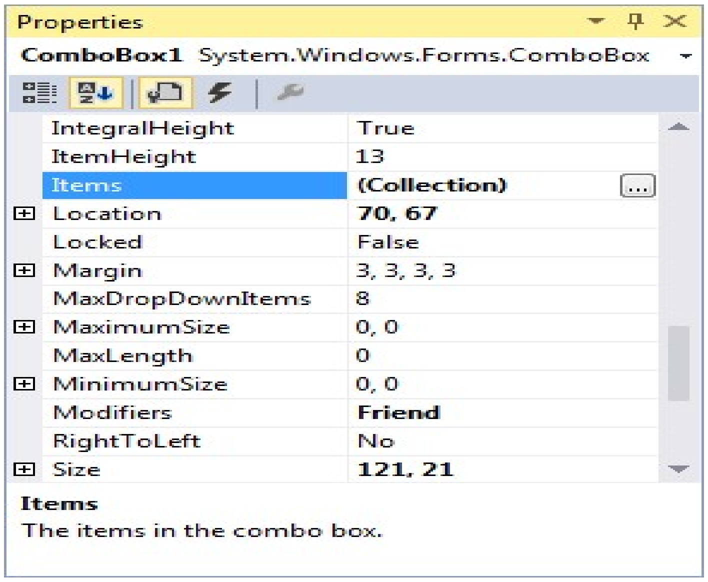
Add a Combo Box to a VB .NET form

Create a new project for this section. Add a button to your new form. Then, locate the Combo Box on the Visual Basic .NET toolbar. It looks like this:



A combo box is a way to limit the choices your user will have. When a black down-pointing arrow is clicked, a drop down list of items appears. The user can then select one of these options. So let's set it up to do that.

- **Click on your Combo Box to select it. Then locate the Item property from the Properties Box:**



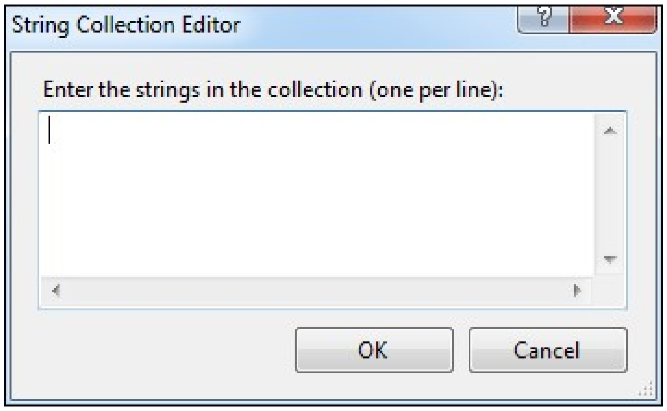
Properties

ComboBox1 System.Windows.Forms.ComboBox

IntegralHeight	True
ItemHeight	13
Items	(Collection)
Location	70, 67
Locked	False
Margin	3, 3, 3, 3
MaxDropDownItems	8
MaximumSize	0, 0
MaxLength	0
MinimumSize	0, 0
Modifiers	Friend
RightToLeft	No
Size	121, 21

Items
The items in the combo box.

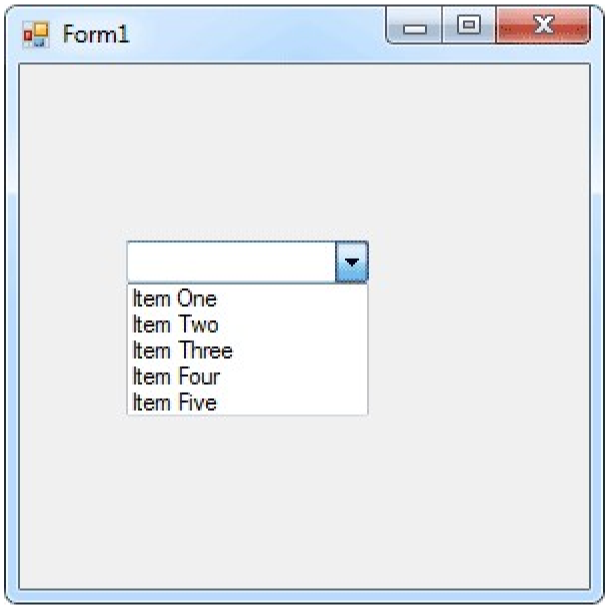
- **Click the grey button, as above. The one with the three dots in it. When you do, you'll get the following box popping up:**



String Collection Editor

Enter the strings in the collection (one per line):

- **To use the String Collection Editor, type an item and press Return (it's just like a normal textbox. Each item will be one item in your drop-down box.)**



Get a Value from a VB NET combo box

Getting a value from a Combo Box is fairly straightforward, because it acts just like a Textbox. The code is like this:

MyVariable = Combobox1.Text

Now we are transferring whatever is selected from the Combo Box to the variable called **MyVariable**. Let's try it. Double click the button you added to your form. This will open the code window. Then enter the following code for the button:

```
Dim MyVariable As String
MyVariable = Combobox1.Text
MessageBox.Show( MyVariable )
```

Finally, the Combo Box has a **DropDownStyle** property. Locate this property and you'll notice its value has a drop down box. The box contains three different Combo Box styles to choose from. Experiment with all three and see how they differ.
Example4 :run example1 using ComboBox1 instead of TextBox1

Conditional Operators

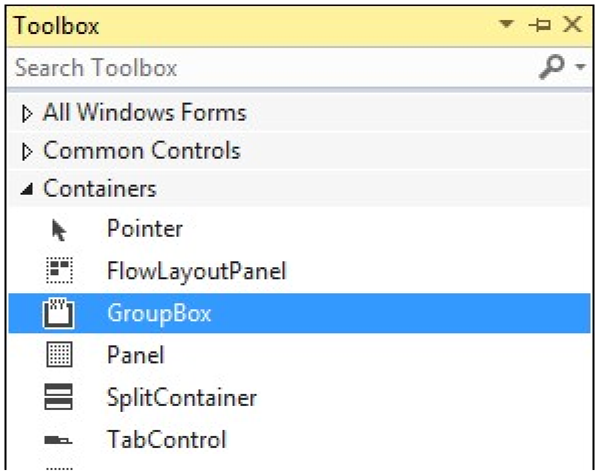
The Conditional Operators allow you to refine what you are testing for. Instead of saying "If X is equal to Y", you can specify whether it's greater than, less than, and a whole lot more. Examine the list of Operators:

Operator	Meaning
>	This symbol means Is Greater Than and is used like this: **If number > 10 Then** **MessageBox.Show("The Number was Greater Than 10")** **End If**
<	This symbol means Is Less Than and is used like this: **If number < 10 Then** **MessageBox.Show("The Number was Less Than 10")** **End If**
>=	This symbol means Is Greater Than or Equal to, and is used like this: **If number >= 10 Then** **MessageBox.Show("The Number was 10 or Greater")** **End If**
<=	This symbol means Is Less Than or Equal to, and is used like this: **If number <= 10 Then** **MessageBox.Show("The Number was 10 or Less")** **End If**
And	You can combine the logical operators with the word And. Like this: **If number > 5 And number < 15 Then** **MessageBox.Show("Greater than 5 And Less than 15")** **End If**
Or	You can combine the logical operators with the word Or. Like this: **If number > 5 Or number < 15 Then** **MessageBox.Show("Greater than 5 Or < Less than 15")** **End If**
< >	A less than symbol followed by a greater than symbol means "Is Not Equal to". It is used like this: **If number1 <> number2 Then** **MessageBox.Show("Is Not Equal to")** **End If**

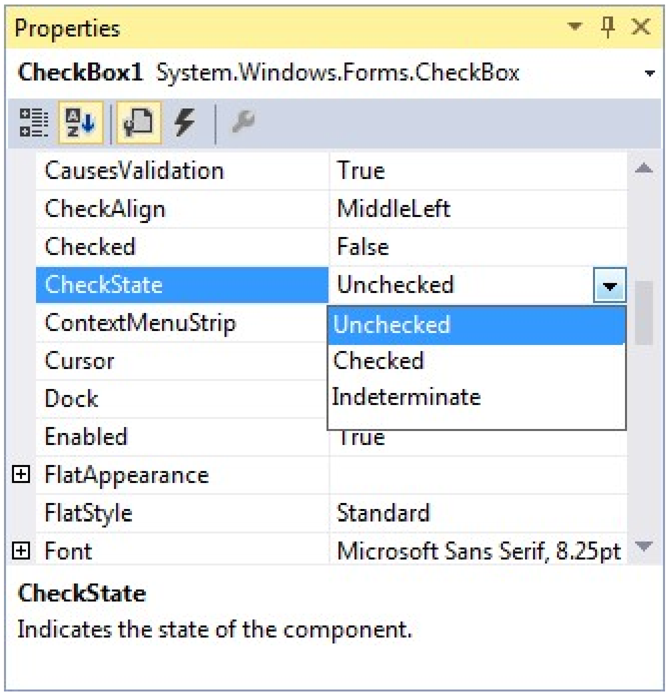
Check Boxes in VB .NET

Two more useful controls in the Visual Basic toolbox are the Check box and the Option Button. You use these when you want to give your users a choice of options.

There is a way to group all your Check Boxes together by using a **Group Box**.



If you click on any one of your Checkboxes and examine its Properties in the Property box, you'll notice that it has a **CheckState** Property.


Properties

CheckBox1 System.Windows.Forms.CheckBox

CausesValidation	True
CheckAlign	MiddleLeft
Checked	False
CheckState	Unchecked
ContextMenuStrip	
Cursor	
Dock	
Enabled	True
FlatAppearance	
FlatStyle	Standard
Font	Microsoft Sans Serif, 8.25pt

CheckState
Indicates the state of the component.

If a checkbox has been selected, the value for the CheckState property will be 1; if it hasn't been selected, the value is zero. (The value for the Indeterminate option is also zero, but we won't be using this.)

```
If CheckBox1.CheckState = 1 Then
    MessageBox.Show("Checked")
End If
```

So the above code is the same as this:

```
If CheckBox1.CheckState = CheckState.Checked Then
    MessageBox.Show("Checked")
End If
```

Modify your code to this:

```
If CheckBox1.CheckState = CheckState.Checked Then
    MessageBox.Show("Checked")
Else
```

```
            MessageBox.Show("Not Checked")
        End If
```

An alternative to Else is ElseIf. It works like this:

```
        If CheckBox1.CheckState = 1 Then
            MessageBox.Show("Checked")
        ElseIf CheckBox1.CheckState = 0 Then
            MessageBox.Show("Unchecked")
        End If
```

When using the ElseIf clause, you need to put what you are testing for on the same line, just after ElseIf. You put the word Then right at the end of the line. You can have as many ElseIf clauses as you want.

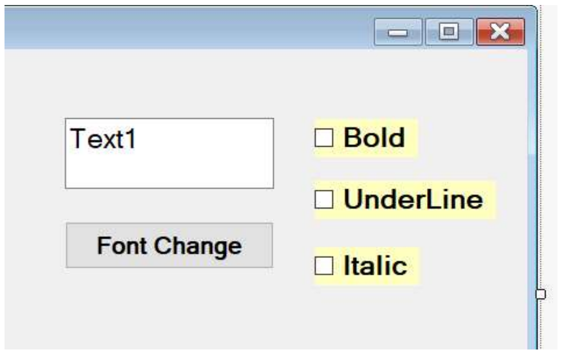
We need to remember what is inside the message variable, so we can just use this:

```
        Message1 = CheckBox1.Text & vbNewLine
        MessageBox.Show("You have chosen " & Message1)
```

What we can also do is count how many Check Boxes were ticked. We can then use a Select Case Statement if we have more Check Boxes.

Exercise

Add a Select Case Statement to the Button1 code to test whatever is inside the three Check Boxes and change the TextBox1 properties



Add Radio Buttons to a VB .NET form

Radio Buttons, sometimes called Option Buttons, are used when you want to restrict a user's choice to one, Male/Female, for example. You want to force your users to pick only one from your list of options.

Adding Radio Buttons to a Form is exactly the same process as adding a Checkbox. Again, we'll add some Radio Buttons to a Group Box, and write code to extract what the user has chosen.

You can place two sets of radio buttons in a first and second group boxes, and these would work independently.

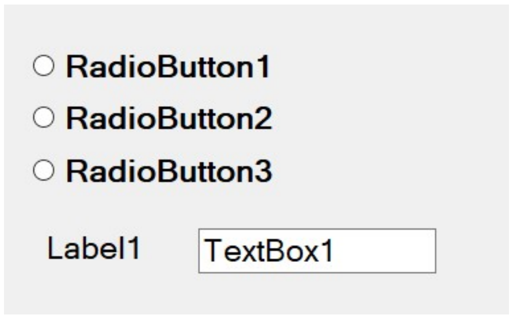
To test which Sit Com was chosen, you can use an If … Elseif Statement. You can do this because only one of the radio buttons will be True if selected: all the others will then have a value of False. (Notice that the Property is now Checked, and not CheckState):

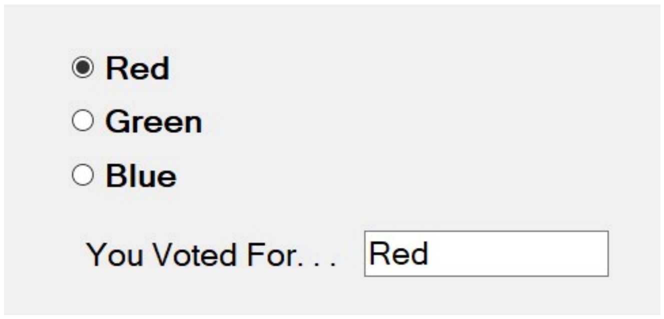
```
Dim Chosen As String
If RadioButton1.Checked = True Then
    Chosen = RadioButton1.Text
ElseIf RadioButton2.Checked = True Then
    Chosen = RadioButton2.Text
     ….
 End If
MessageBox.Show("You voted for " & Chosen)
```

The Text property from the chosen radio button is then placed in a String variable called ChosenSitCom. At the end, we then display the selected radio button in a message box.

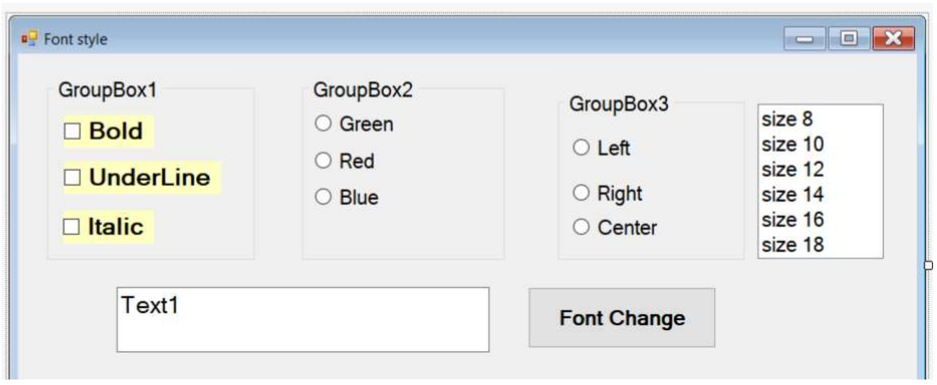
Example

Add a Textbox to your Form. Write code to transfer a chosen option from three options to the Textbox when the button is clicked. Add a label next to the Textbox with the Caption "You Voted For. . . "





Example:
Design the VB form with code to control the Font style of the TextBox as in the figure :



An Introduction to Loops

There are three types of loop for us to cover with VB.NET: 1) For loop, 2) Do loop, 3) While … End While.

For Loops in VB .NET

This first type of loop is called a For Loop. A For loop needs a start position and an end position, and all on the same line. A For loop also needs a way to get the next number in the loop

```
For index = startNumber To endNumber
     . . .
Next
```

Print numbers 1 to 4, create a new Project. Add a button to Form. Double click button and type the following code for it:

```
Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
    Dim startNumber = 1, endNumber = 4, index As Integer
    For index = startNumber To endNumber
        MessageBox.Show(index)
    Next

End Sub
```

Example:

Put two textboxes on your form. The first box asks users to enter a start position for a For Loop; the second textbox asks user to enter an end position for the For loop. When a button is clicked, the program will add up the number s between the start position and the end position. Display the answer in a message box.

Amend your code to check that the user has entered numbers in the textboxes.

For this exercise, you will be passing whatever is in the textboxes to integer variables. It is these variables you are checking with your If Statement. Because numbers will be entered into the textboxes, remember to convert the text to a value with Val( ).

But the Text property will return a zero if the box is empty. So your If statement will need to check the variables for a value of zero. If it finds a zero, Then you can use the Exit Sub code. The If statement should come first, before the For Loop code.

```
Dim startNumber , endNumber , i , answer As Integer
startNumber = Val(TextBox1.Text)
endNumber = Val(TextBox2.Text)
```

```
If startNumber = 0 Or endNumber = 0 Then
    MessageBox.Show("Empty textbox")
    Exit Sub
End If

For i = startNumber To endNumber
    answer = answer + i
Next i
MessageBox.Show(answer)
```

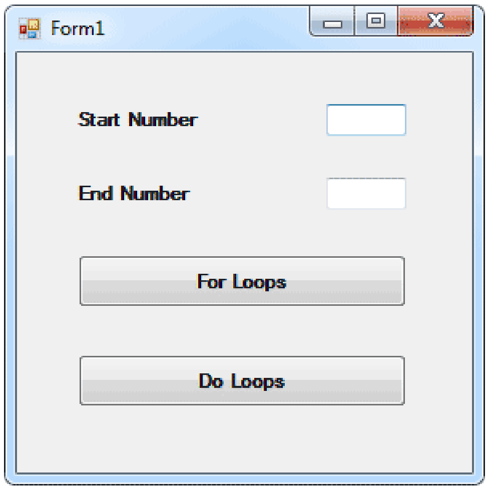
Do Loops in VB .NET

For Loop a specific number of times to loop was coded into it. But what if we don't know how many times around the loop we want to go? A For Loop would not be very efficient in this case.

Do Loop would be use word s like "While" and "Until".

```
Dim number As Integer
number = 1
Do While number < 5
    MessageBox.Show(number)
    number = number + 1

Loop
```

Example:
Load the form you created for the last exercise, the one that has two textboxes and a Button and tested your understanding of For loops. Add another button to the Form. Your form might look something like this:



```
Dim startNumber, endNumber, i, answer As Integer
startNumber = Val(TextBox1.Text)
endNumber = Val(TextBox2.Text)
If startNumber = 0 Or endNumber = 0 Then
    MessageBox.Show("Empty textbox")
    Exit Sub
End If
i = startNumber
Do While i < endNumber
    answer = answer + i
    i = i + 1
Loop
MessageBox.Show(answer)
```

VB NET Do ... Until

Another choice for Do Loops - **Do ... Until**.

There's not much difference between the two, but a Do ... Until works like this. Change your Loop code to the following:

```
Dim number As Integer
number = 1
Do Until number < 5
    MessageBox.Show(number)
    number = number + 1
Loop
```

What is an Array?

One variable was holding one piece of information. An array is a variable that can hold more than one piece of information at a time.

```
Dim MyNumbers(4) As Integer
MyNumbers(0) = 1
MyNumbers(1) = 2
MyNumbers(2) = 3
MyNumbers(3) = 4
MyNumbers(4) = 5
```

In the example above :

- Set up an Integer array with 5 items in it.
- Put number 1 into array position 0, put number 2 into array position 1, put number 3 into array position 2, and so on.
- The array was set to the number 4 - **MyNumbers(4)** - but always remember that an array starts counting at zero, and the first position in your array will be zero.

So that's what an array is - **a variable that can hold more than one piece of data at a time** - but how do they work?

A programming example might help to clear things up.

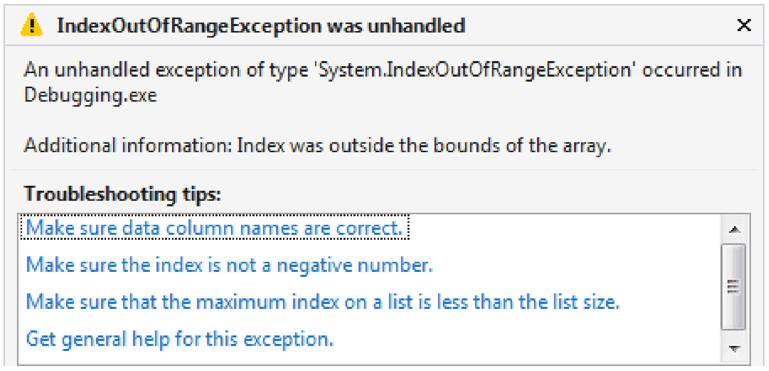
1. **Start a new VB project.**
2. **Add a Button to your Form.**
3. **Set the Text property of the Button to "Integer Array"**
4. **Put the following code behind your button:**

```
Dim MyNumbers(4) As Integer
MyNumbers(0) = 1
MyNumbers(1) = 2
MyNumbers(2) = 3
MyNumbers(3) = 4
MyNumbers(4) = 5
MessageBox.Show("First Number is: " & MyNumbers(0))
MessageBox.Show("Second Number is: " & MyNumbers(1))
MessageBox.Show("Third Number is: " & MyNumbers(2))
MessageBox.Show("Fourth Number is: " & MyNumbers(3))
MessageBox.Show("Fifth Number is: " & MyNumbers(4))
```

The highest Index number is therefore **4**. But we tried to display something at index number **5**:

MyNumbers(5)

Visual Basic said "Wait a minute, the idiot hasn't got a position number 5!" So it stopped the program and gave you an error message.



IndexOutOfRangeException was unhandled

An unhandled exception of type 'System.IndexOutOfRangeException' occurred in Debugging.exe

Additional information: Index was outside the bounds of the array.

Troubleshooting tips:

Make sure data column names are correct.

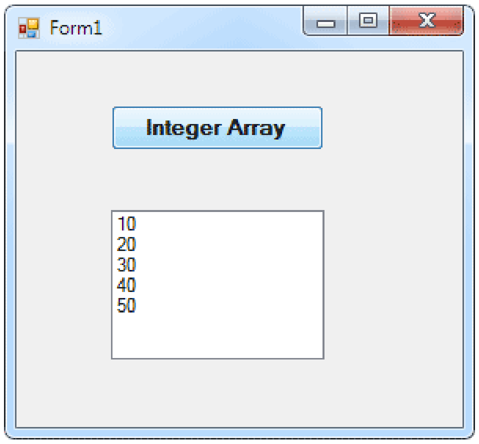
Make sure the index is not a negative number.

Make sure that the maximum index on a list is less than the list size.

Get general help for this exception.

A very handy way to get at the information in array is by accessing its Index number in a For Loop. Add a List Box to form. Make it fairly wide, and just leave it on the default **Name** of **ListBox1**. Then change your code to the following (the new code is in Bold, blue text):

```
Dim MyNumbers(4) As Integer
Dim i As Integer
MyNumbers(0) = 10
MyNumbers(1) = 20
MyNumbers(2) = 30
MyNumbers(3) = 40
MyNumbers(4) = 50
For i = 0 To 4
    ListBox1.Items.Add(MyNumbers(i))
Next i
```



```
10
20
30
40
50
```

Arrays can hold other types of data, too. They can hold Strings of text.

1. Put another Button on your Form
2. Set the Text property to "String Array"
3. Put the following code behind your Button

```
Dim MyText(4) As String
Dim i As Integer
MyText(0) = "This"
MyText(1) = "is"
MyText(2) = "a"
MyText(3) = "String"
MyText(4) = "Array"
For i = 0 To 4
    ListBox1.Items.Add(MyText(i))
Next i
```

There are a number of way you can put data into each position of an array. You can assign values straight from a Textbox into the position of your array. Like this:

MyNumbers(0) = Val(Textbox1.Text)
MyNumbers(1) = Val(Textbox2.Text)
etc

Arrays where the Boundaries are not known

First set up an array with empty brackets

Dim numbers( ) As Integer

Then use the values to reset the array. You reset an array by using the ReDim word. You then specify the new values. Like this:

endAt = Val(Textbox3.Text)
ReDim numbers(endAt)

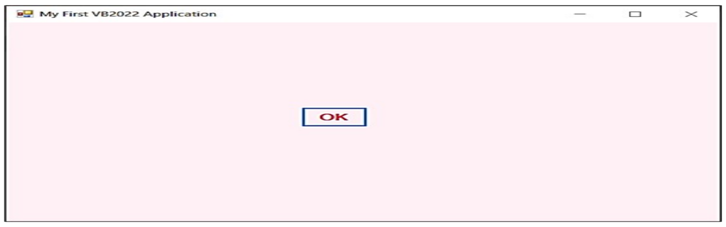
Start a new project, create three textboxes and labels. And add a new Button. Then add the following code:

```
Dim numbers() As Integer
Dim startAt, endAt , times , StoreAnswer , i As Integer
times = Val(TextBox1.Text)
startAt = Val(TextBox2.Text)
endAt = Val(TextBox3.Text)
ReDim numbers(endAt)
For i = startAt To endAt
    StoreAnswer = i * times
    numbers(i) = StoreAnswer
    ListBox1.Items.Add(times & " x " & i & " = " & numbers(i))
Next i
```

VB Tools and Properties Examples

Changing the Properties of the Default-Form at Run-Time

change the properties of the form at run-time by writing the relevant codes. The default form is an object and an instant of the form can be denoted by the name **Me.**



```
Private Sub Form1_Load(sender As Object, e As EventArgs)
Handles MyBase.Load

  Me.Text = "My First vb2022 Program"

  Me.BackColor = Color.GreenYellow

  Me.ForeColor = Color.Blue

  Me.MaximizeBox = False

  Me.MinimizeBox = True

  'Specifying the form size to 400pixel x 400pixel

  Me.Size = New Size(400, 400)

End Sub
```

Changing Font Properties in VB.NET

- **Set font Bold:**

```
Label3.Font = New Font(Label3.Font, Label3.Font.Style Or
FontStyle.Bold)
```

- **Set font not Bold:**

```
Label3.Font = New Font(Label3.Font, Label3.Font.Style And
Not FontStyle.Bold)
```

- **Set font Italic:**

```
Label3.Font = New Font(Label3.Font, Label3.Font.Style Or
FontStyle.Italic)
```

- **Set font not Italic:**

```
Label3.Font = New Font(Label3.Font, Label3.Font.Style And
Not FontStyle.Italic)
```

- **Set font Underline:**

```
Label3.Font = New Font(Label3.Font, Label3.Font.Style Or
FontStyle. Underline)
```

- **Set font not Underline:**

```
Label3.Font = New Font(Label3.Font, Label3.Font.Style And
Not FontStyle. Underline)
```

- **Set font Size:**

```
NewSize=16
Label3.Font = New Font(Label3.Font.Size, NewSize)
```

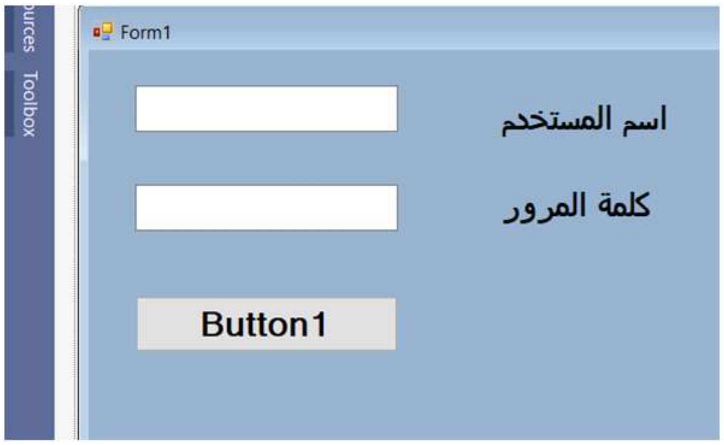
- **Change Text Align:**

```
TextBox1.TextAlign = 0 /1 / 2
```

- **Change text Color:**

```
TextBox1.ForeColor = Color.Blue
```

Design UserName / Password form in VB.NET



1. Label1 (text=”اسم المستخدم”) ,
2. Label2 (text=”كلمة المرور”) ,
3. TextBox1 (Text=””) ,
4. TextBox2 (Text=”” , PasswordChar=”*”)
5. Button1 Code:

```
Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
          Dim name1, pass1 As String
          name1 = "Ahmed"
          pass1 = "123"
          If TextBox1.Text = "" Or TextBox2.Text = "" Then
              MsgBox("enter name or password",, "error")
          ElseIf TextBox1.Text = name1 And TextBox2.Text =
  pass1 Then
              MsgBox("hello  .. " & TextBox1.Text,, "Next
  Form")
          Else
              MsgBox("name or password not correct",, "error")
              Me.BackColor = Color.Red
    Button1.Enabled = False
          End If
  End Sub
```

<u>Input Data from user at run time (InputBox):</u>

```
Variable=  InputBox(“Enter your item ”)
```

Example : Find the product of the two numbers (x ,y) in z ?

```
Dim x, y, z As Integer
x = InputBox("enter x")
y = InputBox("enter y")
z = x * y
TextBox1.Text = z
```

How you can improve Password form by using InputBox to enter ?

Radio / CheckBox /ListBox / ComboBox Events:

- `Private Sub RadioButton1_CheckedChanged(sender As Object, e As EventArgs) Handles RadioButton1.CheckedChanged`

- `Private Sub CheckBox3_CheckedChanged(sender As Object, e As EventArgs) Handles CheckBox3.CheckedChanged`

- `Private Sub ComboBox1_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ComboBox1.SelectedIndexChanged`

How to code for the two controls: ListBox, and ComboBox?

Adding Items to a ListBox , ComboBox

- click on **collection** of the Item property, you will be presented with String Collection Editor whereby you can enter the items one by one by typing the text and press the Enter key

- Items can also be added at runtime using the **Add( ) method**.

```
ListBox.Items.Add("Text")

ComboBox.Items.Add("Text")
```

Example:

```
Dim myitem
myitem = InputBox("Enter your Item")
ListBox1.Items.Add(myitem)
```

Deleting Items from a List Box, ComboBox

- To delete items at design time, simply open the String Collection Editor and delete the items
- To delete an item at runtime, you can use the **Remove** method in the following syntax:

```
ListBox.Items.Remove(“text”)

ComboBox.Items.Remove(“text”)
```

Example:

```
      Dim myitem
      myitem = InputBox("Enter your Item for Deletion")
      MyListBox.Items.Remove(myitem)
```

To clear all the items at once, use the clear method

```
        ListBox.Items.Clear()

        ComboBox.Items.Clear()
```

To count all the items in list , use :

```
        count = ListBox1.Items.Count

        count = ComboBox1.Items.Count
```

To get item value :

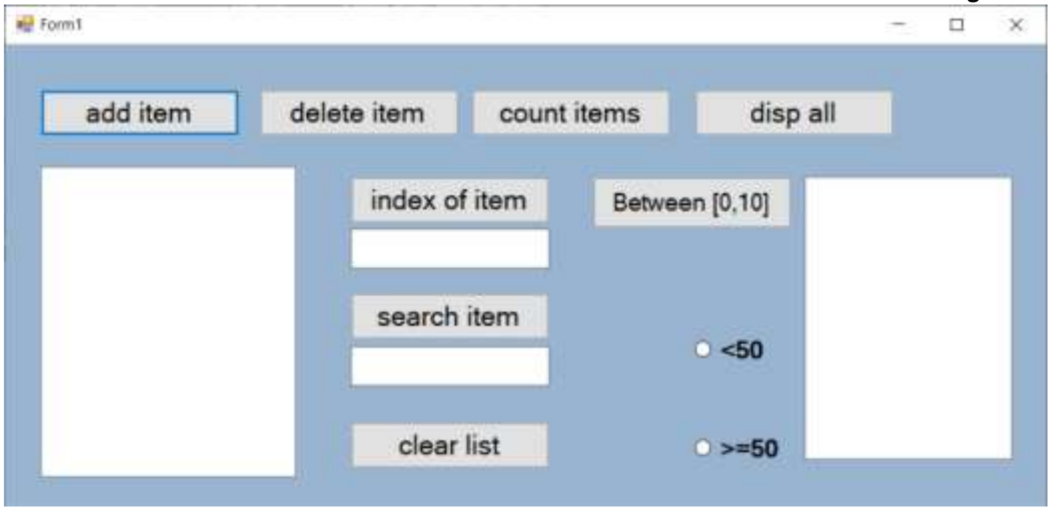
```
        ComboBox1.Items.Item(index)
```

Example : To display all the items use :

```
      Dim i As Integer

      For i = 0 To (ComboBox1.Items.Count) - 1

          MsgBox(i & ComboBox1.Items.Item(i)).

      Next
```

To get the selected item value, item index use:

```
      Item = ComboBox1.SelectedItem

      Index = ComboBox1.SelectedIndex
```

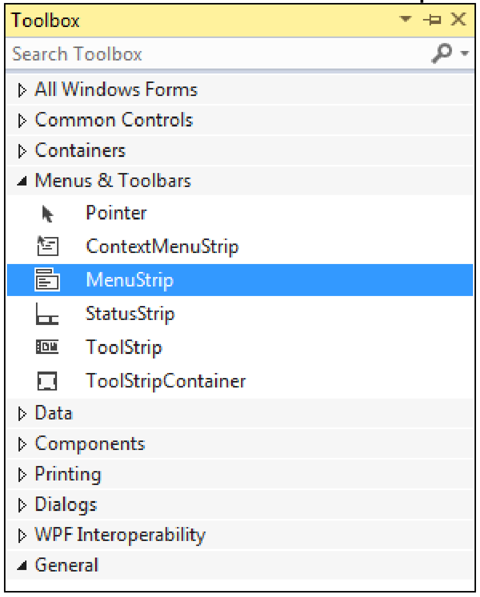


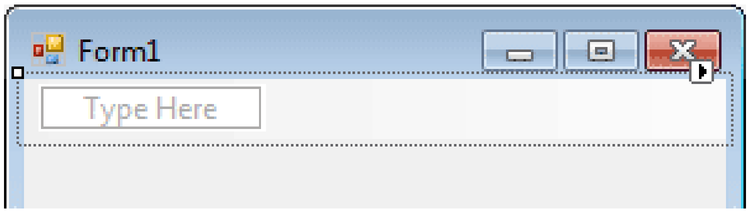
```
Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click
        'add item Button
        'Add one item to list..
        Dim myitem
        myitem = InputBox("Enter your Item ")
        ListBox1.Items.Add(myitem)
    End Sub

Private Sub Button2_Click_1(sender As Object, e As EventArgs)
Handles Button2.Click
        'count items Button
        'show list count..
        Dim count1
        count1 = ListBox1.Items.Count
        MsgBox("list count is " & count1)
    End Sub

Private Sub Button3_Click(sender As Object, e As EventArgs)
Handles Button3.Click

        'delete item Button
        'Delete item from list..
        Dim myitem
        myitem = InputBox("Enter your Item for Deletion")
        ListBox1.Items.Remove(myitem)
    End Sub
Private Sub Button4_Click(sender As Object, e As EventArgs)
Handles Button4.Click
        'index of item Button
        'show index and value of selected item from list ..
        Dim item_index As Integer
        item_index = ListBox1.SelectedIndex
        ' save index in TextBox1
        TextBox1.Text = item_index
        ' show value by MsgBox
        MsgBox("the item value is  " & ListBox1.SelectedItem)
    End Sub

Private Sub Button5_Click(sender As Object, e As EventArgs)
Handles Button5.Click
        'search item Button
        ' Search for text (TextBox2) in list..
        Dim s, r, k
        Dim m As Integer
        k = ""
        ' read text from TextBox2 and convert to Lcase
        s = Microsoft.VisualBasic.LCase(TextBox2.Text)
        For i = 0 To ListBox1.Items.Count - 1
            'read item value from list and covert to Lcase
           r =Microsoft.VisualBasic.LCase(ListBox1.Items.Item(i))
            If InStr(r, s) <> 0 Then
                'collect index of items where s in r
                k = k & " / " & i
                ' add item to list2 where s in r
                ListBox2.Items.Add(ListBox1.Items.Item(i))
                m = m + 1    ' count no. of s in list
            End If
        Next
        If m = 0 Then
            MsgBox("item not in list")
        Else
            MsgBox(k & "   count = " & m)
        End If
    End Sub

Private Sub Button6_Click(sender As Object, e As EventArgs)
Handles Button6.Click
     'clear item Button
        'remove all item from list..
        ListBox1.Items.Clear()
    End Sub

Private Sub Button7_Click(sender As Object, e As EventArgs)
Handles Button7.Click
        'disp all Button
        ' show all item on list one by one ..
        Dim i As Integer
        For i = 0 To (ListBox1.Items.Count) - 1
            MsgBox("index : " & i & " / value is " &
ListBox1.Items.Item(i))
        Next
    End Sub

Private Sub Button8_Click(sender As Object, e As EventArgs)
Handles Button8.Click
        'Between [0,10] Button
        ' add to list2 where item between [0,10]
        Dim i As Integer
        Dim k
        'remove old items from list2
        ListBox2.Items.Clear()

        For i = 0 To ListBox1.Items.Count - 1
            k = Val(ListBox1.Items.Item(i))
            If k > 0 And k < 10 Then ListBox2.Items.Add(k)
        Next
    End Sub
```

```
Private Sub RadioButton1_CheckedChanged(sender As Object, e As
EventArgs) Handles RadioButton1.CheckedChanged
        '<50 RadioButton
        'copy item from list1 to list2 if no. < 50 .
        Dim i As Integer
        Dim k
        ListBox2.Items.Clear()
        'if RadioButton1 (item <50 ) is selected
        If RadioButton1.Checked Then
            For i = 0 To ListBox1.Items.Count - 1
                k = Val(ListBox1.Items.Item(i))
                If k < 50 Then ListBox2.Items.Add(k)
            Next
        End If
    End Sub


Private Sub RadioButton2_CheckedChanged(sender As Object, e As
EventArgs) Handles RadioButton2.CheckedChanged
        '>=50 RadioButton
        'copy item from list1 to list2 if no. >= 50 .
        Dim i As Integer
        Dim k
        ListBox2.Items.Clear()
        'if RadioButton2 (item >=50 ) is selected
        If RadioButton2.Checked Then
            For i = 0 To ListBox1.Items.Count - 1
                k = Val(ListBox1.Items.Item(i))
                If k >= 50 Then ListBox2.Items.Add(k)
            Next
        End If
    End Sub
```

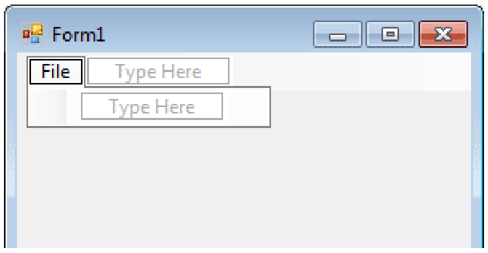
Adding Menus to a Visual Basic .NET Form

Menus are very common to Windows program. Visual Basic itself has many of these drop down menus - File, Edit, View, Project, Format, etc. And they're very easy to add. Start a new project, use the toolbox to add a MenuStrip control:

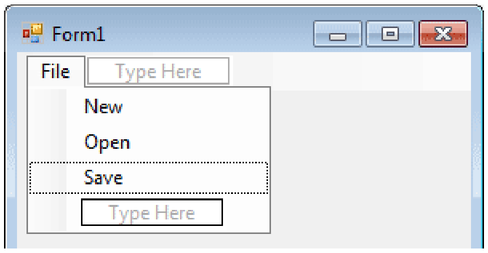




- If clicking on this (it's highlighted above), see that the Properties box on the right changes. There are many properties for the control. But there are lots of properties for the MenuItem object.
- The MenuItem object is the one at the top of the form - The one that says ***Type Here***.
- To start building menu, click inside the area that says "***Type Here***". Type the word ***File***. Now press the enter key, menu will look like this:



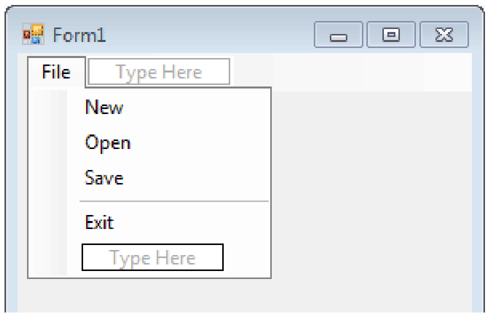
- To create items on File menu, click inside the ***Type Here*** box. Enter the word ***New***, and press the enter key. Add an "Open" and a "Save" item to menu in the same way, menu will then look like this:



- The final item is an "Exit" item.

How can add a separator line between the items in menu?

To add a separator, click inside the "Type Here" box. Instead of typing a letter, type the minus character "-". When you hit your return key, you'll see the separator appear:
Click inside the "Type Here" area, and add an Exit (or Quit) item. Now a File menu like this one:



Adding code to a VB.NET menu

Click File in Design Time to see drop down menu, double click an item to open up the code window.

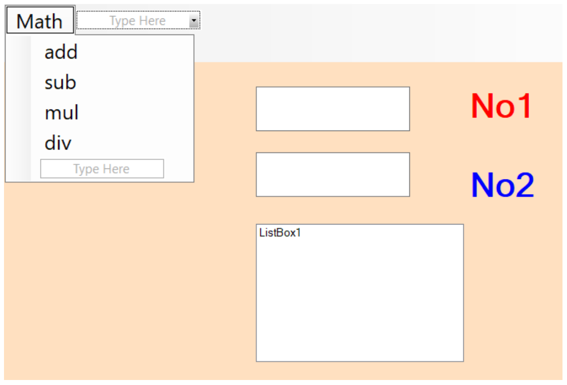
- Click Exit item…. Nothing will happen!
- Select Click event from the list of The Event …. taken straight into the code for that event. It should be like this one:

```
Private Sub ExitToolStripMenuItem_Click_1(sender As Object, e As EventArgs) Handles
ExitToolStripMenuItem.Click

End Sub
```

- Exit menu item actually does something…. form should close down, and returned to the design environment. The word "Me" refers to the form, then :

```
Private Sub ExitToolStripMenuItem_Click_1(sender As Object, e As EventArgs) Handles
ExitToolStripMenuItem.Click

        Me.Close()

End Sub
```

Example: Create VB.Net Form with Math Menu as in figures bellow, the form apply math operation on the text values of the two TextBox, then add results to ListBox1:



- Math:
  - Add: No1+No2
  - Sub: No1-No2
  - Mul:No1*No2
  - Div:No1/No2

```
Private Sub AddToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles
AddToolStripMenuItem.Click
   ListBox1.Items.Add("No1+No2 = " & Val(TextBox1.Text) + Val(TextBox2.Text))
End Sub
-----------------------------------------------------------------------
```

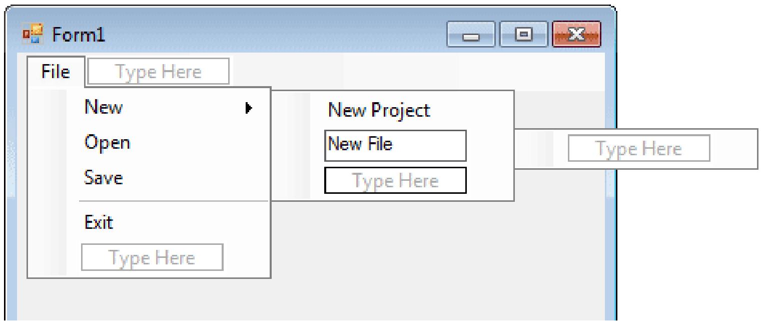
```
Private Sub SubToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles
SubToolStripMenuItem.Click
   ListBox1.Items.Add("No1-No2 = " & TextBox1.Text - TextBox2.Text)
End Sub
----------------------------------------------------------------------

Private Sub MulToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles
MulToolStripMenuItem.Click
   ListBox1.Items.Add("No1xNo2 = " & Val(TextBox1.Text) * Val(TextBox2.Text))
End Sub
----------------------------------------------------------------------

Private Sub DivToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles
DivToolStripMenuItem.Click
   ListBox1.Items.Add("No1/No2= " & Val(TextBox1.Text) / Val(TextBox2.Text))

End Sub
```

Add a Sub Menu to your VB.NET Form

A sub menu is one that branches of a menu item. They usually have an arrow to indicate that there's an extra menu available. Click on menu to see all its branches.

- Select the item, careful where you click. Click once on the left edge of the item to create new menu.
- Return to the Form view, Click on your ***File*** menu,

- Select the ***New*** item,
  1. Click on the "***Type Here***" just to the right of ***New***,
  2. Type ***New Project***, and then hit the return key on your keyboard
  3. Type in ***New File*** and then click away from the menu, somewhere on the form
  4. Then a menu like this one:

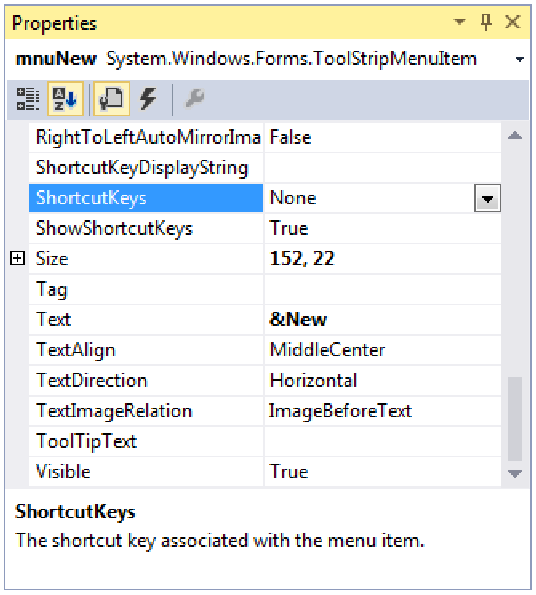


Add Shortcuts to your Menu Items

A key combination shortcut is one that appears at the end of a menu item (**for example, Ctrl + X**), easily add this option to programs. So try this:

1. In Design time, select the Exit item on menu …. **For example Exit**
2. Look at the properties box on the right

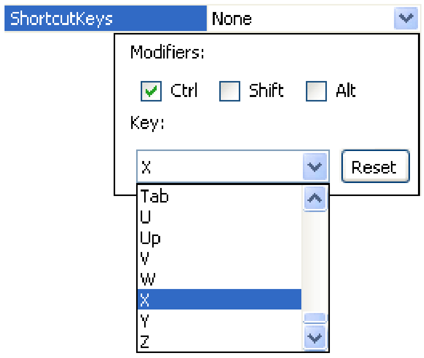
3. Locate the ShortcutKeys item, Click the down arrow to reveal the following: The Modifier is the key you press with your shortcut. For example, the **CTRL key** then **the "X" key** on your keyboard.



mnuNew System.Windows.Forms.ToolStripMenuItem

RightToLeftAutoMirrorIma	False
ShortcutKeyDisplayString	
ShortcutKeys	None
ShowShortcutKeys	True
⊞ Size	152, 22
Tag	
Text	&New
TextAlign	MiddleCenter
TextDirection	Horizontal
TextImageRelation	ImageBeforeText
ToolTipText	
Visible	True

ShortcutKeys
The shortcut key associated with the menu item.



ShortcutKeys None

Modifiers:

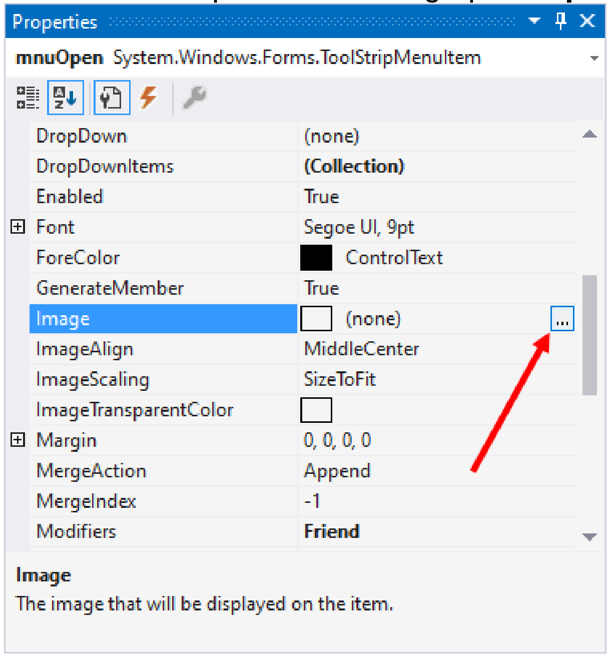
[✓ Ctrl] Shift Alt

Key: X

Add Image Icons to Menu Items

Add image icons to VB Net menus need some icons, Microsoft has released an icon Image Library download for free. The link is here: Microsoft Image Library. The file size is only 11 megabytes.

- To add an icon to a menu item, in Design View, first select an item on menu… **for example Open** item.
- With the item selected, go to the Properties, scroll down and locate the **Image Property**.
- Click the little square with the **three dots** on it to bring up a dialog box. Select Local Resource at the top. This will bring up an **Open File dialog box**.

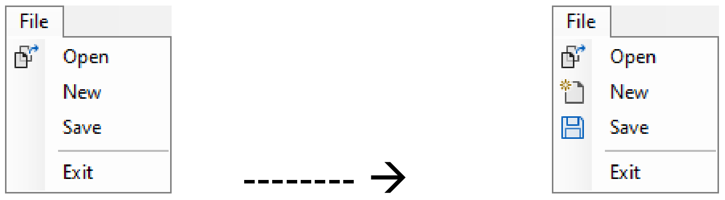


mnuOpen System.Windows.Forms.ToolStripMenuItem

DropDown	(none)
DropDownItems	(Collection)
Enabled	True
⊞ Font	Segoe UI, 9pt
ForeColor	■ ControlText
GenerateMember	True
Image	□ (none)
ImageAlign	MiddleCenter
ImageScaling	SizeToFit
ImageTransparentColor	□
⊞ Margin	0, 0, 0, 0
MergeAction	Append
MergeIndex	-1
Modifiers	Friend

Image
The image that will be displayed on the item.

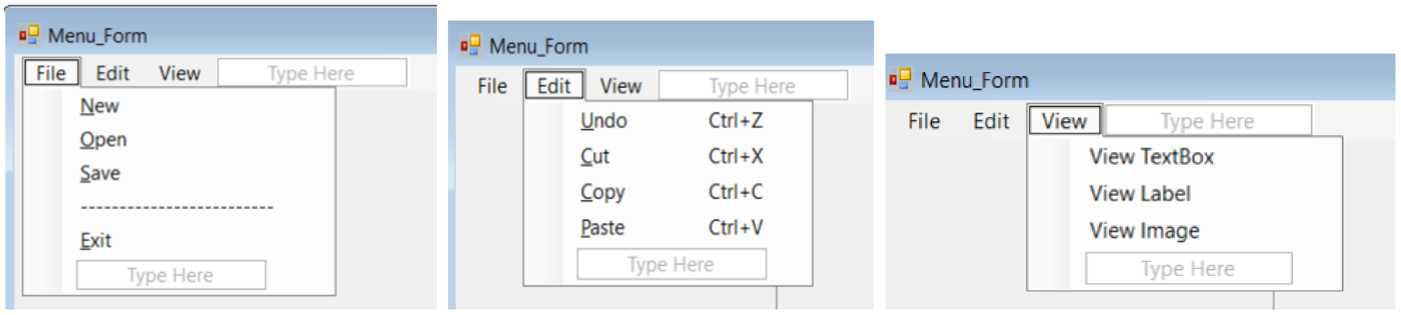
The image is a bit tiny, as it's only 16 x 16 pixels. the icon appear next to a menu item text:



Example : A VB .NET Menu Project

Create VB.Net project, add right-click menus to project as in figures:

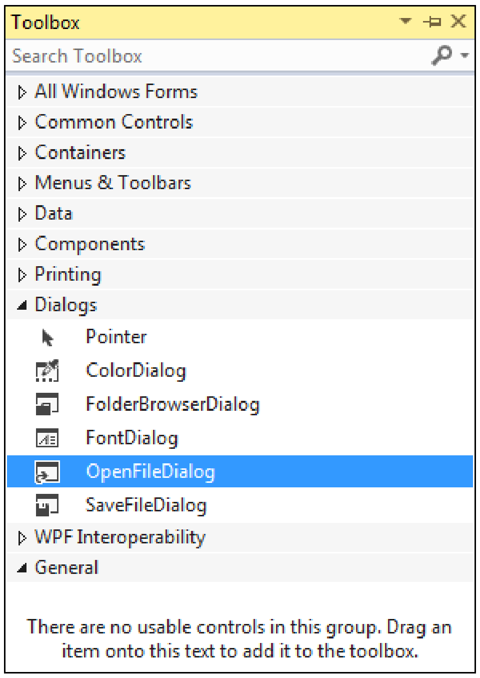
1. Add an underline shortcut for ALL menu item
2. Add a least one key combination shortcuts per drop down menu
3. Write code to display a message box whenever a menu item is clicked, or its shortcut used. The message box should explain what the menu item will do when it's fully implemented.

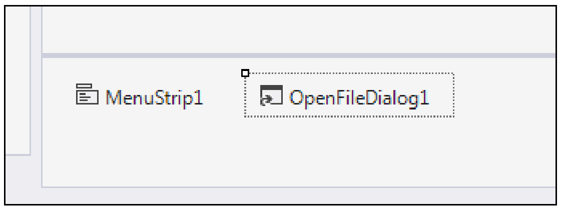


The Open File Dialogue Box

In most programs, if you click the **File menu**, and select the **Open item**, a dialogue box is displayed. From the dialogue box, click on a **File button** to select it, then click the **Open button**. The file clicked on is then opened up.

From toolbox, locate the control called "**OpenFileDialog**". But notice that the control doesn't get added to a form. It gets added to the area at the bottom, manipulate the properties of control:

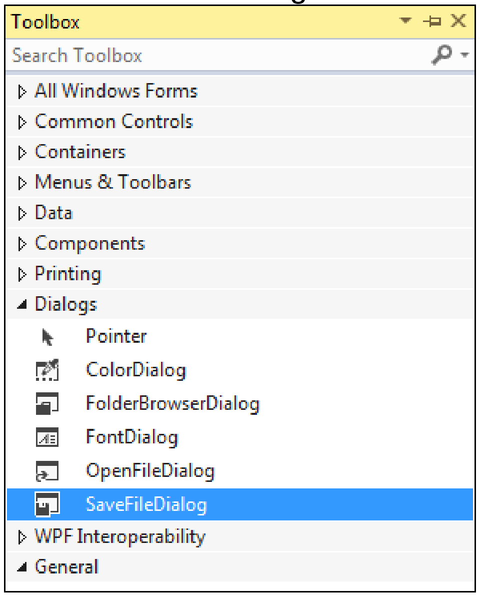


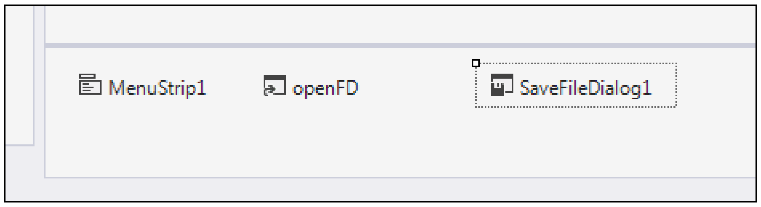


1. **Initial Directory:** the directory the dialogue box should display when it appears. Instead of it displaying the contents of the "My Documents" folder, for example, display the contents of any folder. This done with the Initial Directory property.
2. **Title Property:** By default, the dialogue box will display the word "Open" as a caption at the top of dialogue box, change this with the Title property.
3. **Filter Property:** In most dialogue boxes, can display a list of specific files that can be opened. These are displayed in the "Files of Type" drop down list. To do this in VB.NET, access the Filter property.
4. **FileName Property:** The Open Dialogue box has a property that returns the file name that was selected (as string value).
5. **The location (the path)** of the file is also displayed.
6. **DialogResult.Cancel :** if click the Cancel button, the FileName will be blank. If the cancel button is clicked, the result of the action is stored by VB.NET in this property.

The SaveFileDialog Control

The save dialogue box works in the same way as the Open dialogue box. However, control called SaveFileDialog:





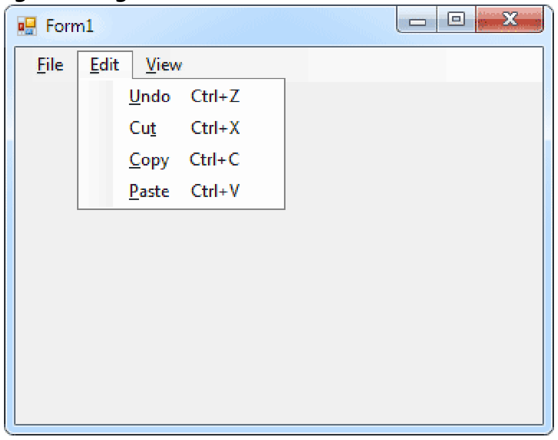
Just like the Open control, can use the properties of the Save control on dialogue boxes. Try changing these properties:

Initial Directory, Title, Filter, FileName

There's another useful property use with the Save control - the **Overwrite prompt**. When set this property, a message box pops up warning that the file will be overwritten.

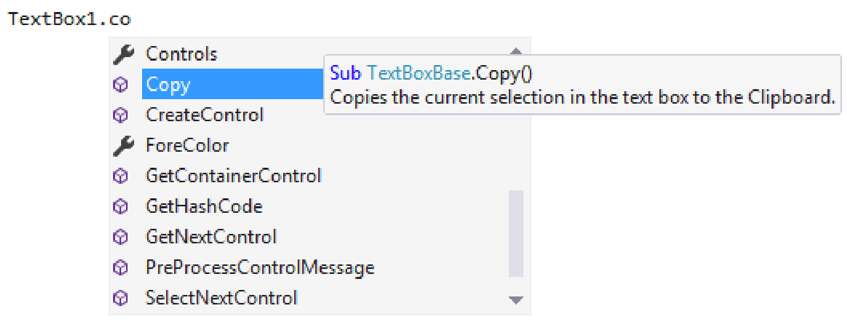
Cut, Copy, Paste and Undo Text in VB .NET

Example … continue : Add two textboxes to form and set their MultiLine property to True. To get the menu items to work: Undo, Cut, Copy and Paste.



Implement a Copy Menu in VB NET

- Type ***Textbox1*** in code window, then a full stop, a list of properties and methods available to the textbox. Scroll up to the top and locate the Copy method:



```
TextBox1.co
```

Sub TextBoxBase.Copy()
Copies the current selection in the text box to the Clipboard.

Copies the current selection to **the Clipboard**. The Clipboard is a temporary storage area available to most Windows applications. When invoke the Copy method of the textbox, any selected text is place in this temporary storage area.

Implement a Paste Menu in VB NET

- Whatever is on the Clipboard was pasted into the selected TextBox.

TextBox2.Paste() **….** "Whatever is on the Clipboard, paste it into Textbox2".

Implement a Cut Menu in VB NET

- Select the text in textbox to cut it, the text should disappear (it's on the clipboard, though). Click inside textbox two, and click Paste. The text should be pasted over.

TextBox1.Cut()

Implement a Undo Menu in VB NET

- Select the text in textbox. Click **Edit > Cut** and the text disappears.
- Now click **Edit > Undo**. The text reappears.

TextBox1.Undo()

How to Show and Hide Controls

The items on our view menu are:

- **View Textboxes**
- **View Labels**
- **View Images**

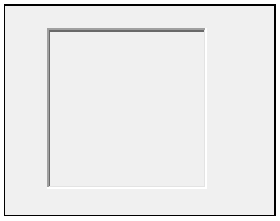
To hide a control, simply set it's Visible property to False. If you want to get it back, show a control by setting the Visible property to True.

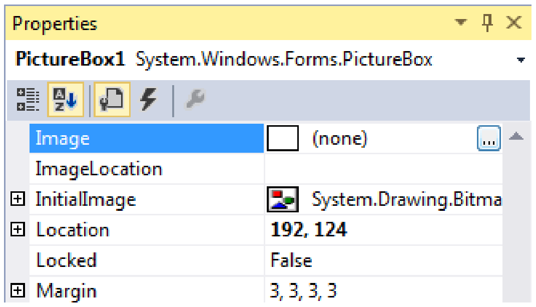
Exercise

Add two labels to your form. Write code to toggle the labels on and off. The two labels should disappear with the textboxes. And they should reappear when the menu item is toggled to the on position

View Images menu Item

- To insert an image, locate the **PictureBox** control in the toolbox (under Common Controls).
- Change the **Height** and **Width** properties of the Picture Box to 100, 100.
- To make it stand out more, locate the **BorderStyle** property. Change the value to **Fixed3D**
- The **SizeMode** property. Set this to **AutoSize** and picture box will resize to the size of the image.
- To add a picture at design time, Locate the **Image property** in the properties box:

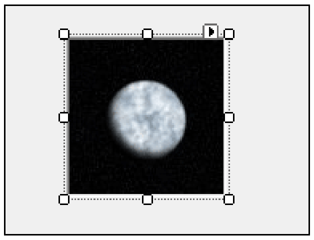




Properties

PictureBox1 System.Windows.Forms.PictureBox

Image	(none)
ImageLocation	
⊞ InitialImage	System.Drawing.Bitma
⊞ Location	192, 124
Locked	False
⊞ Margin	3, 3, 3, 3



Insert an Image with your View Menu

To add Image to the form, **PictureBox** used from **ToolBox** and then select the image file from computer to view in this **PictureBox**. From the View Images menu item, open file dialogue box to specify an image for to select. The code is same as Open File, there's only two lines haven't met yet:

```
Dim strFileName As String
OpenFD.InitialDirectory = "C:\"
OpenFD.Title = "Open an Image"
OpenFD.Filter = "jpegs|*.jpg|gifs|*.gif|Bitmaps|*.bmp"
Dim DidWork As Integer = OpenFD.ShowDialog()
If DidWork <> DialogResult.Cancel Then
        strFileName = openFD.FileName
        PictureBox1.Image = Image.FromFile(strFileName)     '1..
        OpenFD.Reset()                                      '2..
End If
```

1. Previously, loading the image into the Image property of **PictureBox1** directly from the Properties Box (by clicking the grey button with the three dots in it) in design time. At run time, loading an image into the **Image property** using code. The way is with the **FromFile method** of the Image Class. In between round brackets, you type the name and path of the file you're trying to load. Since file name has been placed inside of the **strFileName variable**, then assign this to the Image property of a Picture Box.
2. The last line, **openFD.Reset()**, will reset the initial directory of the open file dialogue box.

Text Files and VB .NET

There is a very useful object in VB.NET called **System.IO** (the IO stands for **Input** and **Output**).can use this object to read and write to text files.

What is a Text File?

The files on computer all end in a three letter extension. Microsoft Word files will have a different three letter extension from Microsoft Excel files. The extension is used to identify one file type from another. That way, Excel won't try to open Word files, or vice versa. Text files have an extension that ends in **.txt**. The Windows operating system gives a basic Text Editor in Notepad. The Notepad program allows save files with the **.txt** extension. In other words, as Text Files. These Text Files can then be opened by a wide variety of programs. A simple text file like this is called a **Sequential File**.

How to Open a Text File in VB .NET

The ability to open up a text file and read its contents can be very useful.

- **Cretae a StreamReader :**
  - To open up a text file, create something called a "**StreamReader**". This reads streams of text.
  - The StreamReader is an object available to System.IO.

    ```
    Dim FILE_NAME As String = "C:\Users\Owner\Documents\test.txt"
    Dim objReader As New System.IO.StreamReader(FILE_NAME)
    ```
  - StreamReader needs the name of a file to Read.
  - Now that **objReader** is an object variable, it has its own **properties** and **methods** available for use (in the same way that the textbox has a Text property).
- **Read To End :** One of the Methods available to StreamReader variable is the ReadToEnd method. This will read the whole of text, right to the end.
- Must have to close the stream objects after using them, otherwise it get errors messages.
- Unless already having a file called test.txt at the location specified, this'll get error message popping up
- **ReadLine :**The ReadLine method to StreamReader variable, to reads text of one line at a time. In order to do this, need to use a loop.
- **Peek** : method takes a peek at the incoming text characters. It's looking ahead one character at a time. If it doesn't see any more characters, it will return a value of minus one (-1). This will signify the end of the text file.
- **vbNewLine :** adding a new line character

Example : Create VB.Net form (design the form and write the vb code) to read data from the file "Users.txt" saved in "D:\" for example, suppose the line of text coming in from the text file was this:

"UserName1, Password1, Password2"

The design: TextBox1 multiline= True,

Button1 (Read Data) to read file and save data to TextBox1

The vb code for Button1(Read File):

```
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Dim FILE_NAME As String = "D:\Users.txt"
        Dim TextLine As String
        If System.IO.File.Exists(FILE_NAME) = True Then
            Dim objReader As New System.IO.StreamReader(FILE_NAME)
            Do While objReader.Peek() <> -1
                TextLine = TextLine & objReader.ReadLine() & vbNewLine
            Loop
            TextBox1.Text = TextLine
        Else
            MessageBox.Show("File Does Not Exist")
        End If
    End Sub
```

How to Write to a Text File in VB .NET

Writing to a text file is similar to reading a text file. Again we use System.IO.

- The **StreamWriter** is used to write a stream of text to a file.
- Unless you have a file to write on it, should see the message box display: "File Does Not Exist."
- But notice that if open up the text file, any text had previously will be gone - it has been overwritten, rather than appended to.
- Created a new variable name, **objWriter**, and using **StreamWriter** instead of **StreamReader**.
- The "**Write ()**" method of **StreamWriter** was chosen to write to file. In between round brackets, put what to write to text file.
- To write line by line. In which case, select **WriteLine** from the available properties and methods of **StreamWriter**.

Example (continue…) : *Add another Button to the been working on. Set the **Text** property of the button to "**Write to File**".*

Double click coding of Button2 (Write to File) :

```
    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
    Button2.Click
        Dim FILE_NAME As String = "D:\Users2.txt"
        If System.IO.File.Exists(FILE_NAME) = True Then
            Dim objWriter As New System.IO.StreamWriter(FILE_NAME)
            objWriter.Write(TextBox1.Text)
            objWriter.Close()
            MessageBox.Show("Text written to file")
        Else
            MessageBox.Show("File Does Not Exist")
        End If
    End Sub
```

Homework Example : write VB code to save the content of ListBox1 to a new text file

Appending Text to a File in VB .NET

- To add text to what you currently have. In which case you need to **Append**.
- Set up the object variable for the StreamWriter, just typed the name and path of the file.

- To append text to a file, you type a comma after your file name then type the word True:

Dim objWriter As New System.IO.**StreamWriter(** FILE_NAME, **True)**

- If you want to add some text to the file, you need that True value. If you leave out the True or False, a new file is not created.

Homework Example : write VB code to append the content of TextBox1 to a text file

Creating a file if it doesn't exist

- To create a file if one doesn't exist, the process is again quite simple:

*Dim objWriter As New System.IO.StreamWriter(FILE_NAME, **False**)*

- This time, just added the word "False" to the end of FILE_NAME. This will ensure that a new text file is created if one doesn't exist.

How to Copy a File in VB .NET

- To copy a file created. use **System.IO.File** object.
- File has it's own properties and methods. One of these is **Copy(,)**
- To use the **Copy** method of **System.IO.File**. In between the round brackets, first type the name of the file you want to copy. After a comma, type the name of the new file and its new location:

```
Dim FileToCopy As String
Dim NewCopy As String
FileToCopy = "d:\test.txt"
NewCopy = "d:\NewTest.txt"
If System.IO.File.Exists(FileToCopy) = True Then
    System.IO.File.Copy(FileToCopy, NewCopy)
    MessageBox.Show("File Copied")
End If
```

How to Move a File in VB .NET

- Move a file in a similar manner as to Copying a File - specify a source file and a new destination for it. This time, using the **Move** method of **System.IO.File**.

System.IO.File.Move(FileToMove, MoveLocation **)**

How to Delete a File in VB .NET

- Deleting a file is quite simple - ***but dangerous!*** So be very careful when trying out this code.

- Make sure the file going to delete is not needed - ***won't be able to restore it from the recycle bin!***
- To delete a file from computer, use the **Delete** method of **System.IO**:

System.IO.File.Delete(FileToDelete **)**

Example1: Create vb.net project to read text from text file “vb_text1.txt” which save in the location “E drive” and load the text to TextBox1 as multiline

```
Dim FILE_NAME As String = "E:\vb_text1.txt"
Dim TextLine As String
If System.IO.File.Exists(FILE_NAME) = True Then
        Dim objReader As New System.IO.StreamReader(FILE_NAME)
    Do While objReader.Peek() <> -1
        TextLine = TextLine & objReader.ReadLine() & vbNewLine
    Loop
    TextBox1.Text = TextLine
    Else
        MessageBox.Show("File Does Not Exist")
End If
```

Example2: Create vb.net project to read students names from text file “D:\vb_students.txt” and load the names to ListBox1, then find number of the students.

```
Dim FILE_NAME As String = "D:\vb_students.txt"
Dim TextLine As String
If System.IO.File.Exists(FILE_NAME) = True Then
        Dim objReader As New System.IO.StreamReader(FILE_NAME)
    Do While objReader.Peek() <> -1
        TextLine = objReader.ReadLine()
  ListBox1.Items.Add(TextLine)
    Loop
Else
        MessageBox.Show("File Does Not Exist")
End If
MessageBox.Show("Number of Students is : ", ListBox1.Items.Count)
```

Example3: Create vb.net project to read the whole data from text file using open dialog and display that data in TextBox1 .

```
Dim FILE_NAME As String
OpenFD.InitialDirectory = "E:\"
OpenFD.Title = "Open Text File"
Dim DidWork As Integer = OpenFD.ShowDialog()
If DidWork <> DialogResult.Cancel Then
    FILE_NAME = OpenFD.FileName
    Dim All_Text As String
    If System.IO.File.Exists(FILE_NAME) = True Then
        Dim objReader As New System.IO.StreamReader(FILE_NAME)
        All_Text = objReader.ReadToEnd()
        TextBox1.Text = All_Text
    Else
        MessageBox.Show("File Does Not Exist")
    End If
End If
```

للحصول على اسم ومسار الملف الذي تم اختياره من قبل المستخدم

Example4: Create vb.net project to write the whole data from TextBox1 to text file "E:\text2.txt".

```
Dim FILE_NAME As String = "E:\text2.txt"
    If System.IO.File.Exists(FILE_NAME) = True Then
        Dim objWriter As New System.IO.StreamWriter(FILE_NAME)
        objWriter.Write(TextBox1.Text)
        objWriter.Close()
        MessageBox.Show("Text written to file")
    Else
        MessageBox.Show("File Does Not Exist")
    End If
```

Example5: Create vb.net project to write the List Items from ListBox1 to text file "E:\text2.txt".

```
Dim FILE_NAME As String = "E:\text2.txt"
If System.IO.File.Exists(FILE_NAME) = True Then
    Dim objWriter As New System.IO.StreamWriter(FILE_NAME)
    Dim text_line As String
    For i = 0 To (ListBox1.Items.Count) - 1
        text_line = ListBox1.Items.Item(i)
        objWriter.WriteLine(text_line)
    Next
    objWriter.Close()
    MessageBox.Show("Text written to file")
Else
    MessageBox.Show("File Does Not Exist")
End If
```

Example6: Create vb.net project to write the List Items from ListBox1 to text file use save dailog.

```
SaveFD.InitialDirectory = "E:\"
SaveFD.Title = "Save Text File"
SaveFD.Filter = "text|*.txt"
Dim sav As Integer = SaveFD.ShowDialog()
If sav <> DialogResult.Cancel Then
    FILE_NAME = SaveFD.FileName
    If System.IO.File.Exists(FILE_NAME) = True Then
        Dim objWriter As New System.IO.StreamWriter(FILE_NAME)
        Dim text_line As String
        For i = 0 To (ListBox1.Items.Count) - 1
            text_line = ListBox1.Items.Item(i)
            objWriter.WriteLine(text_line)
        Next
        objWriter.Close()
    Else
        MessageBox.Show("File Does Not Exist")
    End If
End If
```

An Introduction to Functions and Subs

- So far, the code have been writing has mostly been put together under one button. The problem with this approach is that, the code can get quite long and complex, making it difficult to read, and difficult to put right if something goes wrong.

- Another approach is to separate some of this code into its own routine. This is where functions and subs come in. The two terms refer to segments of code that are separate from main code.

- User can write new Function or Sub once. Then when if want to use it, just refer to it by name, and the code will get executed.

The difference between Functions and Subs

- Functions return a value, and Subs don't.
- A Sub is some code or job that want VB to get on with and execute. An example is closing a form down and unloading it with Me.Close(). don't need to return a value, here; just want VB to close your form down.

Create your Own Subs in VB .NET

- To write a Sub, the cursor needs to be outside of the button code, and on a new line before the "End Class". So, the Sub code written on a new line outside the buttons code .

Example1 : create new VB project, set up a TextBox on a form. The text box will be a First Name text box, and write a Sub code to checking that the user has actually entered something into TextBox.

Sol. do this:

1. **Start a new project, and put a Text Box on the Form.**
2. **Add the following code :**

```
Private Sub ErrorCheck()
     Dim TextBoxData As String
          TextBoxData = Trim(TextBox1.Text)
     If TextBoxData = "" Then
          MessageBox.Show("Please Enter your First Name")
     End If
End Sub
```

The name "ErrorCheck" is entirely the own, and could have called it almost anything liked. But it's better to stick to something descriptive.

How to use your new Sub

- But the Sub is not doing much good where it is. To use the new Sub, need to tell Visual Basic to execute the code by referring to the Sub by name.
- So click inside the button code, just before the End Sub of the button. Type the following:

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
    Call ErrorCheck()
End Sub
```

- ❖ Can use this segment of code whenever in project just by referring to it by name.

Of course, it can have code check more than one Textbox. For example can include the Sub code wrote to check for a correct email address, as well. But all that error checking code is used for TextBox1 only!!.. This is a problem.

Using Parameters in your Subs

- ❖ A Parameter is a value that we want to pass from one code section to another.
- ❖ The parameters are what we want to hand to our Subroutine.
- ❖ They are called **Arguments** when you pass them, and **Parameters** when they are received.

Because this is somewhat confusing, we'll stick to calling them Parameters

Example2: Create a form with two textboxes contain names (FirstName, SecondName), add third textbox for full name, write a Sub code to check the text in the two text box before concatenation the two names into full name. When the button is clicked, a Message Box will pop up revealing the answer of the Full Name.

Adding three labels to the form then do the following:

1. **Set the Name property of the Label1 to FirstName**
2. **Set the Name property of the Label2 to SecondName**
3. **Set the Name property of the Label3 to FullName**
4. **Adding a new button to the Form and set the Text property to "Get Full Name"**

```
Private Sub ErrorCheck2(Tb As String)
    Dim TextBoxData As String
    TextBoxData = Trim(Tb)
    If TextBoxData = "" Then
        MessageBox.Show("Please Enter your Name")
        TextBox3.Enabled = False
    End If
End Sub

Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
    Dim first, second, answer As String
    ErrorCheck2(TextBox1.Text)
    ErrorCheck2(TextBox2.Text)
    first = TextBox1.Text
    second = TextBox2.Text
    answer = first & " " & second
    TextBox3.Text = answer
End Sub
```

Exercise : Create a Sub to check a Textbox for a valid email address of University of Mosul employees or students. Pass whatever is entered in the Textbox to a variable called "email". Pass the value from this variable a new Sub by using a Parameter. When a button is clicked, a message box should pop up telling the user if the email address was wrong.

How to Create a Function in VB .NET

- A **function** is more or less the same thing as a Sub - a segment of code you create yourself, and that can be used whenever you want it.
- The **difference** is that a **Function returns a value, while a Sub doesn't**.
- A Function has a value, will be equal to something, and have to assign a value to it.
- Create a Function in the same way did a Sub, but this time the code will be like this:

```
Private Function NewName() As Boolean
End Function
```

- Added "**As**" something, in this case "**As Boolean**".. Use one of the Types : "**As Integer**", "**As Long**", "**As Double**", "**As String**", or any of the variable types.
- Adding some Parameters to the Function is in exactly the same way as did for a Sub.
- **Return keyword** can use to return the value of function

Example3 : Add a new button and a textbox to the form. Create a function to check if textbox has a value or it is empty as Boolean value, then call the function by the Button.

```
Private Function ErrorCheck3() As Boolean
    Dim TextBoxData As String
    TextBoxData = Trim(Text2.Text)
    If TextBoxData = "" Then
        MessageBox.Show("Blank Text Box detected")
        ErrorCheck3 = True
    Else
        ErrorCheck3 = False
    End If
End Function

Private Sub ButtonFun_Click(sender As Object, e As EventArgs) Handles
ButtonFun.Click
    Dim IsError As Boolean
    IsError = ErrorCheck3()
    If IsError = True Then
        Exit Sub
    Else
        MessageBox.Show("Text2 is : " & text2.Text)
    End If
End Sub
```

Example4: Create Function **AddTwoNumbers**, and set it up to return an **Integer** value. The two parameters passing in are also Integers. The code inside the Function simply adds up whatever is inside the variables **first** and **second**. The result is passed to another variable, **answer**, then pass whatever is inside **answer** to the name of our Function. Use AddTwoNumbers function to add the value of the two TextBox by Button click.

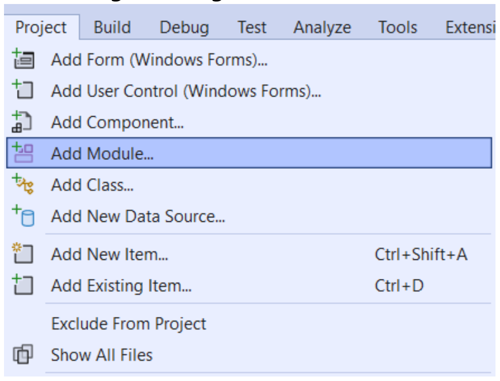
```
Private Function AddTwoNumbers(first As Integer, second As Integer) As
Integer
    Dim answer As Integer
    answer = first + second
    'AddTwoNumbers = answer
    Return answer
End Function

Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
    Dim first As Integer
    Dim second As Integer
    Dim result As Integer
    first = Val(TextBox1.Text)
    second = Val(TextBox2.Text)
    result = AddTwoNumbers(first, second)
    If result = 0 Then
        MessageBox.Show("Please try again ")
    Else
        MessageBox.Show("The answer is " & result)
    End If
End Sub
```

Example5: Create more Functions for: fact, power ,….

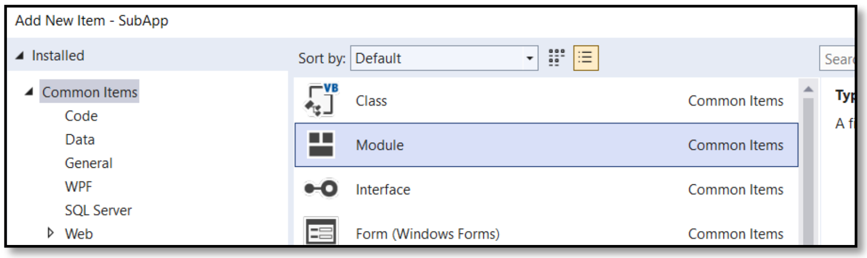
Standard Modules in VB .NET

- ❖ It's better to separate all your Functions and Subs and put them somewhere else - in something called a **Module**.
- ❖ Exploring the **Standard Module**, and move new Functions and Subs outside of Form. Then can use them in other projects.
- ❖ Start a new project. Add a button to a form. To add a Module to project, click **Project** from the menu bar. From the menu, click on **Add Module**, then write the code:

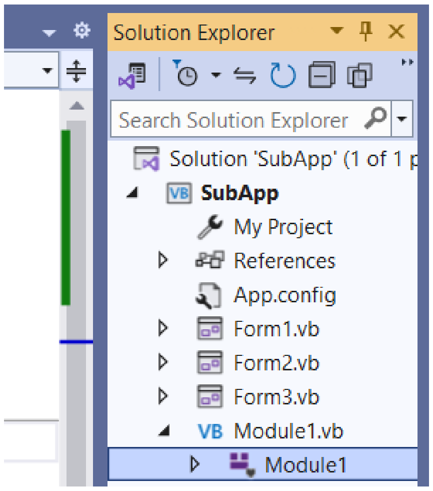
```
Module Module1
    Function AddTwoNumbers(ByVal first As Integer, ByVal second As
Integer) As Integer
        Dim answer As Integer
        answer = first + second
        Return answer
    End Function
End Module
```



→



at the **Solutions Explorer** on the right, a new module is listed:



Sub and Function Examples

```
Private Sub Button3_Click(sender As Object, e As EventArgs) Handles Button3.Click
        Dim x1, y1, z1 As Integer
        x1 = 2
        y1 = 3
        z1 = x2_py(x1, y1)
        MsgBox(z1)
    End Sub
```

```
Function x2_py(x As Integer, y As Integer) As Integer
        Dim z, i As Integer
        z = 1
        For i = 1 To y
            z = z * x
        Next
        Return z
    End Function
```

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Dim s As String
        s = InputBox("enter ")
        ListBox1.Items.Add(s)

    End Sub

 Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
        list_avg()

    End Sub
```

```
Sub list_avg()
        Dim c, avg As Integer
        For c = 0 To (Form1.ListBox1.Items.Count()) - 1
            avg = avg + Int(Form1.ListBox1.Items.Item(c))
        Next
        MsgBox("list avg is " & avg / Form1.ListBox1.Items.Count())
    End Sub
    Private Sub Button4_Click(sender As Object, e As EventArgs) Handles
Button4.Click
        list_avg1(ListBox2)
    End Sub
```

```
Sub list_avg1(l As Object)
        Dim c, avg As Integer
        For c = 0 To (l.Items.Count()) - 1
            avg = avg + Int(l.Items.Item(c))
        Next
        MsgBox("list avg is " & avg / l.Items.Count())
    End Sub
```

```vb
Private Sub Button5_Click(sender As Object, e As EventArgs) Handles Button5.Click
        Dim h As String
        h = text_copy(TextBox3, TextBox2)
        MsgBox(h)
    End Sub
```

```vb
Function text_copy(ch As Object, ch2 As Object) As String
        If ch.text <> "" Then
            ch.copy
            ch2.paste
            text_copy = " copy selected text"
        Else
            text_copy = "no text selected"
        End If
    End Function
```

```
Private Sub Button6_Click(sender As Object, e As EventArgs) Handles Button5.Click
        Dim g As Object
        g = radio_select(GroupBox1)
        MsgBox(g.name)
    End Sub
```

```
Function radio_select(gbox As Object) As Object
        Dim i As Integer
        For i = 0 To (gbox.controls.count()) - 1
            If gbox.controls.item(i).checked Then
                Return gbox.controls.item(i)
                Exit For
            End If
        Next
    End Function
End Module
```

The Click Event

What is an Event?

An event is something that happens. An event in programming terminology is when something special happens. These events are so special that they are built in to the programming language.

The Click Event

Buttons have the ability to be clicked on. When a button is clicked, the event that is fired is the Click Event.

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click

End Sub
```

1. This is a Private Subroutine.
2. The name of the Sub is Button1_Click.
3. The Event itself is at the end: Button1.Click.
4. The Handles word means that this Subroutine can Handle the Click Event of Button1.
5. This sets up two variable: one called sender and one called e.
6. Sender variable type is System.Object. This stores a reference to a control (which button was clicked, for example).
7. The e variable, this is holding an object, too (information about the event), this information might be where the mouse pointer was on the screen.

But because this is the Click Event, there's not much more information available: either the button was clicked or it wasn't.

The MouseDown Event

The **MouseDown event** is available to many controls on the form. A Form can detect when the mouse was held down on it; a textbox can detect when the mouse was held down inside of it and a Button can detect which mouse button was held down to do the clicking.

In between the round brackets of the Subroutine, still have **sender As Object**. But have a new argument now:

e As MouseEventArgs

The name of the variable is still **e**. But the type of Object being stored inside of the **e** variable is different. This stands for Mouse Events Arguments. What is being stored inside of the **e** variable is information about the Mouse Event.

Which Button was Clicked?

Inside of the Button1_MouseDown Subroutine, type the following code, Left and Right are the ones use most often.

```
Private Sub Button1_MouseDown(sender As Object, e As MouseEventArgs)
Handles Button1.MouseDown
    If e.Button = MouseButtons.Right Then
        MsgBox(">R")
    ElseIf e.Button = MouseButtons.Left Then
        MsgBox("L<")
    End If
End Sub
```

MouseDown and the Form

This time, a Private Subroutine called **Form1_MouseDown**. The two arguments are exactly the same as before.

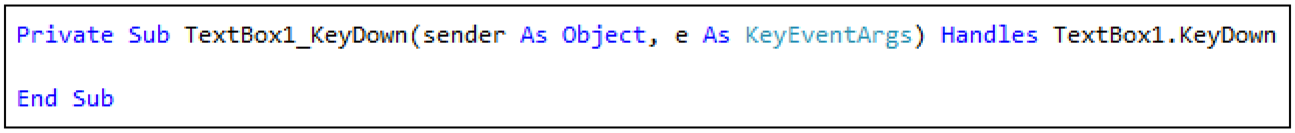
The way to detect when a mouse button was clicked on the form itself.
A code for Form1_MouseDown. Change it to this:

```
   Private Sub Form1_MouseDown(sender As Object, e As MouseEventArgs)
Handles Me.MouseDown
        Dim xPos As Integer
        Dim yPos As Integer
        If e.Button = MouseButtons.Right Then
            xPos = e.X
            yPos = e.Y
            MessageBox.Show("The X Position is " & xPos & " The Y
Position is " & yPos)
        End If
End Sub
```

The **X** property returns how far across, from left to right, the mouse is; the **Y** property returns how far down, from top to bottom, the mouse is.

The KeyDown Event

As its name suggest, this allows to detect when a key on the keyboard was held down. This is useful for things like validating text in a textbox.



```
Private Sub TextBox1_KeyDown(sender As Object, e As KeyEventArgs) Handles TextBox1.KeyDown

End Sub
```

The event that is being Handled is the **KeyDown** event of TextBox1. Notice, though, that there is a slightly different argument in round brackets:

e As KeyEventArgs

This means that the variable **e** will hold information about the Key on the keyboard that you're trying to detect.

Example: to check if F1 is clicked write this in event :

```
    Private Sub TextBox1_KeyDown(sender As Object, e As KeyEventArgs)
Handles TextBox1.KeyDown
        If e.KeyCode = Keys.F1 Then
            TextBox1.Clear()
            MessageBox.Show("Help!!!")
        End If
    End Sub
```

The Form Load Event

For example, set the Enabled property of a control to False when a form loads. Or maybe blank out an item on a menu. this can do all from the Form Load event.

VB.NET Classes and Objects

VB.NET is an Object Oriented programming language. The Objects referred to are created from something called a Class.

Classes and Objects

In VB.NET, a class is that chunk of code mentioned earlier. The Form started out with is a Class. If look right at the top of the code window for a Form :

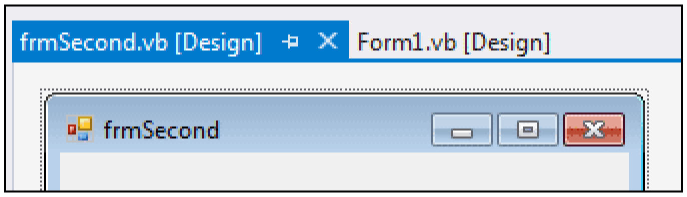
Public Class Form1

- The word "Public" means that other code can see it. Form1 is the name of the Class
- End Class signifying the end of the code for the Class.
- A basic difference between a Class and an Object: **A Class is the code itself; the code becomes an Object when you start using it.**

Creating Multiple Forms in VB .NET

It's a rare program that only has one form in it. Most programs will have other forms. These other forms can be used for things like Find and Replace searches, extra formatting capabilities, to set Options for the program, and a whole lot more besides.

- From the VB.NET design environment, click the **Project** menu.
- From the drop down menu, click **Add Windows Form**.
- The Add New Item dialogue box appears.
- Select **Windows Form** under Templates.
- Then click inside the Name textbox at the bottom.
- Change the Name of the form to **frmSecond.vb**.
- Then click Add.



- To switch between forms, click the tabs. In the image, two tabs are displayed: Form1 (the original and first form), and a new form **frmSecond**.

- So click the tab for Form1, and add a button to this form. Change the **Name** property of the button to **btnShowSecond**.
- In order to display the second form, add this code to your button

```
SecondForm.Show()
```

Example: Create project with two forms(Form1,Form2). Add one TextBox and one Button to each of the forms.

In Form1 Button add the code to copy the Form1 TextBox data into Form2 TextBox before show Form2

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button2.Click
    Form2.TextBox1.Text = Me.TextBox1.Text
    Form2.Show()
End Sub
```

In Form2 Button Add the code to return to Form1 and close Form2

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
    Form1.Show()
    Me.Close()
End Sub
```

Q : what is results if the code is:

```
Form1.Show()
Me.Hide()
```

Modal and Non Modal Forms

A modal from is one that has to be dealt with before a user can continue. An example is the Change Case dialogue box in Microsoft Word. Until click either the Cancel or OK buttons, the program won't let click anywhere else. A Modal form is sometimes called a dialogue box.

The second form is called a **Modeless form**. These are forms than can be hidden or sent to the taskbar, then return to the main form or program and do things with it.

Example cont.: Add a second button to a Form1. Change the **Name** property of the new button to **btnDialogueBox**. Event of click the new Button and add the following code:

```
Dim frmDialogue As New frmSecond

frmDialogue.ShowDialog()
```

To display a form as a Modal dialogue box, use the **ShowDialog** method. When the form is a Modal dialogue box, create OK and Cancel buttons for it. VB.NET then has a trick up its sleeve for these types of buttons.

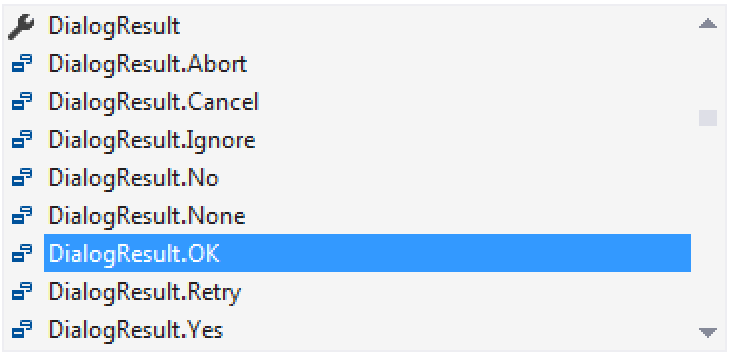
OK and Cancel Buttons

In the design environment, add Form3 then Click the Tab for a **Form3**. When the **Form3** is displayed in the design window, add two buttons to it. Change the **Name** property of the first button to **btnOK**, and the **Name** property of the second to **btnCancel**.

Event click of Form3 btnOK button and add the following code to it:

```
Me.DialogResult = DialogResult.OK
```

- The **Me** keyword refers to the current form.
- Select **DialogResult** from the pop up list that appears.
- DialogResult is a property of the Form. It can accept a range of values.



- When the button is clicked, VB.NET will return a result of OK for this button.
- When using ShowDialog , there is no need to close the dialogform .

Event code for Form3 btnCancel button the following line:

```
Me.DialogResult = DialogResult.Cancel
```

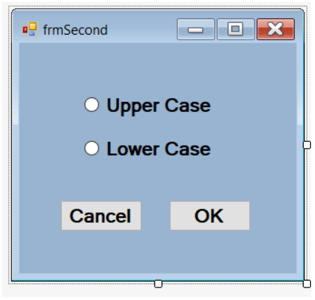
When the button is clicked, VB.NET will return a result of Cancel for this button. **Form1 code**, the lines that display the second form should be these:

```
Private Sub Button3_Click(sender As Object, e As EventArgs) Handles
Button3.Click
    Form3.ShowDialog()
    If Form3.ShowDialog() = DialogResult.OK Then
        MessageBox.Show("OK Button Clicked")
    End If
    If Form3.ShowDialog() = DialogResult.Cancel Then
        MessageBox.Show("Cancel Button Clicked")
    End If
End Sub
```

Getting at Values on other Forms

The form with OK and Cancel buttons on it is not doing much good. Let's turn the form into a Change Case dialogue box.

Example : Design a Form like the one in the following image (this is **frmSecond**):



- In Form1 and add a Textbox to it. When the button on Form1 is clicked, the dialogue box above will display.
- Then select an option button to change the case to Upper, or Lower case.
- This will happen when the OK button is clicked. Whatever text is in Texbox1 on Form1 will be changed accordingly.
- Double click the OK button on **frmSecond** to access the code is the following:

Me.DialogResult = DialogResult.OK

If you want to refer to Texbox1 on Form1, you can just do this:

Form1.Textbox1.Text

- In the code for the OK button on frmSecond. Add the following two lines at the top:
- The two Radio buttons on a form were called **optUpper**, and **optLower** . In the code.
- The one that was chosen will have its **Checked** property set to **True**. We then store into the variable **ChangeCase** the converted text from the textbox.
- The final line puts the converted text back into Textbox1 on Form1.

```
Dim ChangeCase As String
ChangeCase = Form1.TextBox1.Text
If optUpper.Checked Then
     ChangeCase = ChangeCase.ToUpper
ElseIf optLower.Checked Then
     ChangeCase = ChangeCase.ToLower
End If
Form1.TextBox1.Text = ChangeCase
```

Creating a Database Application

A database management system typically deals with storing, modifying, and extracting information from a database. It can also add, edit and delete records from the database. However, a DBMS can be very difficult to handle by ordinary people or businessmen who have no technological backgrounds. Fortunately, we can create user-friendly database applications to handle the aforementioned jobs with the DBMS running in the background. One of the best programs that can create such a database application is Visual Basic.Net 2022. Visual Basic 2022 uses ADO.NET to handle databases. ADO.NET is Microsoft's latest database technology which can work with many other advanced database management systems such as Microsoft SQL server

ADO.NET Data provider

VB.Net allows you many ways to connect to a database or a data source. The technology used to interact with a database or data source is called ADO.NET. The ADO parts stands for Active Data Objects.

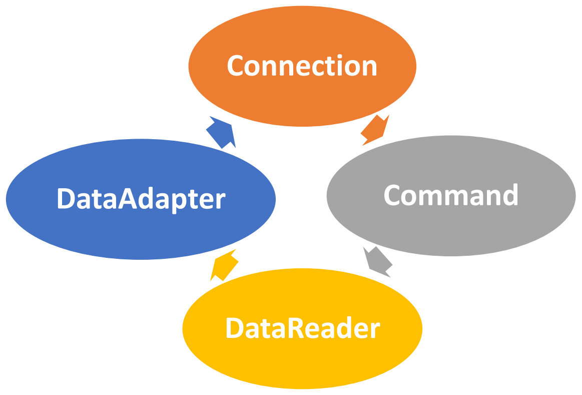
Applications talk to a **database** in two ways:

1. to get the **data stored** there and show it in a way that's easy to understand.
2. to **update the database** by ***adding***, ***Updating***, or ***removing data***.

Microsoft ActiveX Data **Objects.Net (ADO.Net)** is a model that is part of the **.Net framework**.

What is a Data provider?

The data provider is used to connect to a database. Executing commands and retrieving data, storing it in a Dataset, reading the retrieved data, and updating the database. The following four things make up the **ADO.Net data provider**:



What is a Database Connection?

This component is used to set up a connection with a data source

What is Command?

A command is a SQL statement or a stored procedure used to retrieve, insert, delete or modify data in a data source.

What is Data Reader?

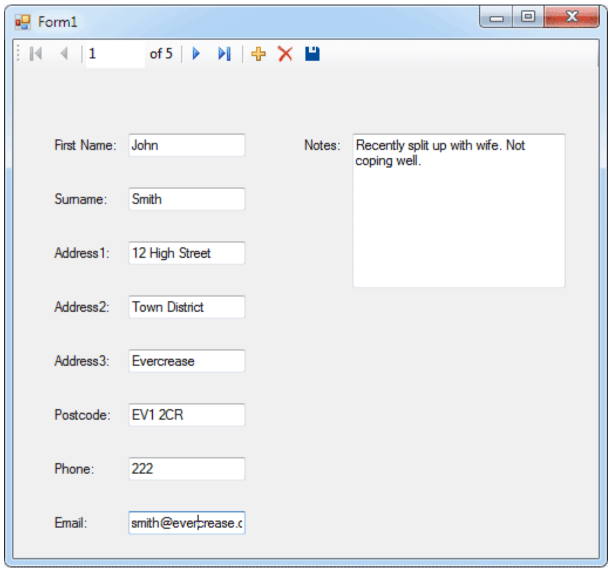
Data reader is used to retrieve data from a data source in a read-only and forward-only mode.

What is DataAdapter?

This is integral to the working of ADO.Net since data is transferred to and from a database through a data adapter. It retrieves data from a database into a dataset and updates the database. When changes are made to the dataset, the changes in the database are actually done by the data adapter.

The Database Wizard in VB NET

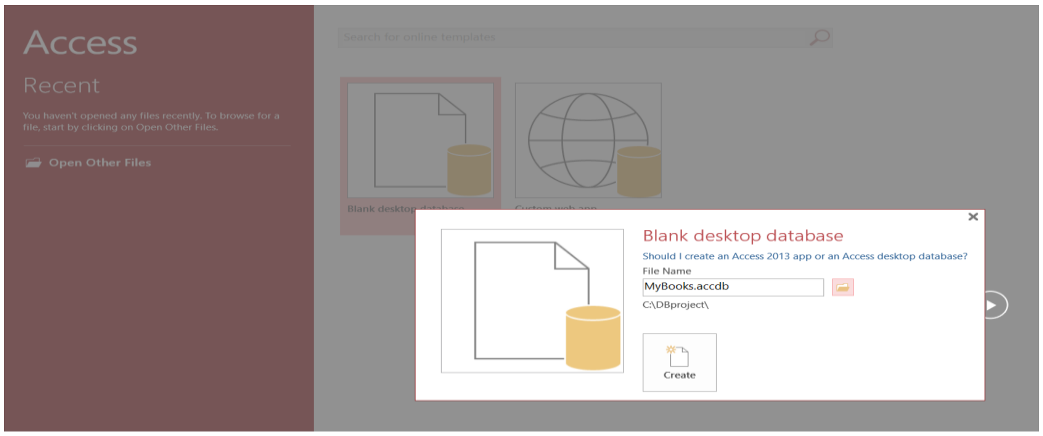
- Use a Wizard to create a program that reads the database and allows us to scroll through it.
- The wizard will do most of the work for us, and create the controls that allow users to move through the database. For Example : The Form we create will look like this when it's finished:



First Name:	John
Surname:	Smith
Address1:	12 High Street
Address2:	Town District
Address3:	Evercrease
Postcode:	EV1 2CR
Phone:	222
Email:	smith@evercrease.c
Notes:	Recently split up with wife. Not coping well.

Steps How to Connect Access Database in VB.Net

Step 1: Create an MS Access Database: Open an **MS Access Database** on your Computer and Create a **Blank Database** and Save it as ***"MyBooks.accdb"***.


Access
Recent
You haven't opened any files recently. To browse for a file, start by clicking on Open Other Files.
Open Other Files
Search for online templates
Blank desktop database
Custom web app
Create

Blank desktop database

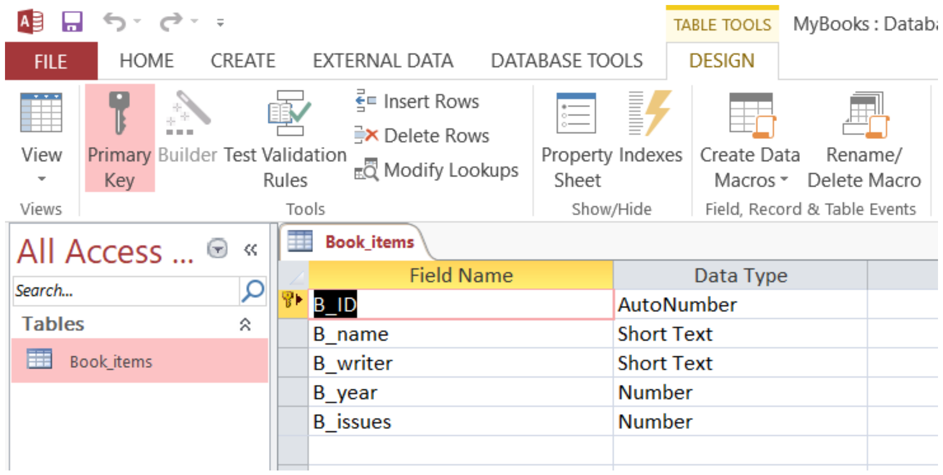
Should I create an Access 2013 app or an Access desktop database?

File Name

MyBooks.accdb

C:\DBproject\

Step 2: Create a Database Table: To create a table, follow the image below and save it as ***"Book_items"***.

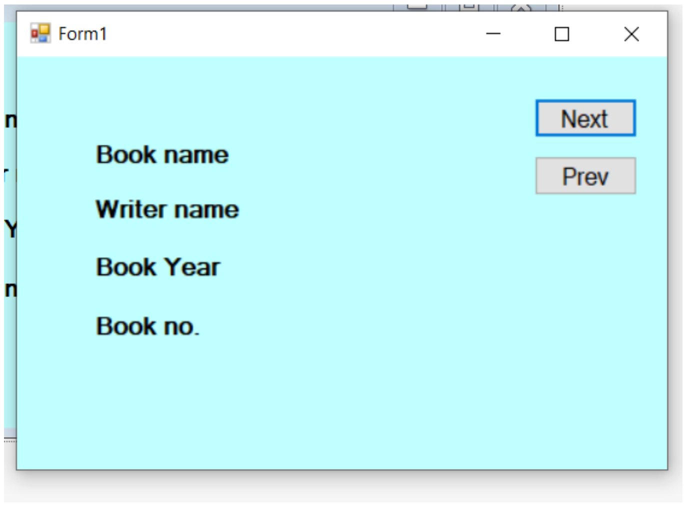


Field Name	Data Type
B_ID	AutoNumber
B_name	Short Text
B_writer	Short Text
B_year	Number
B_issues	Number

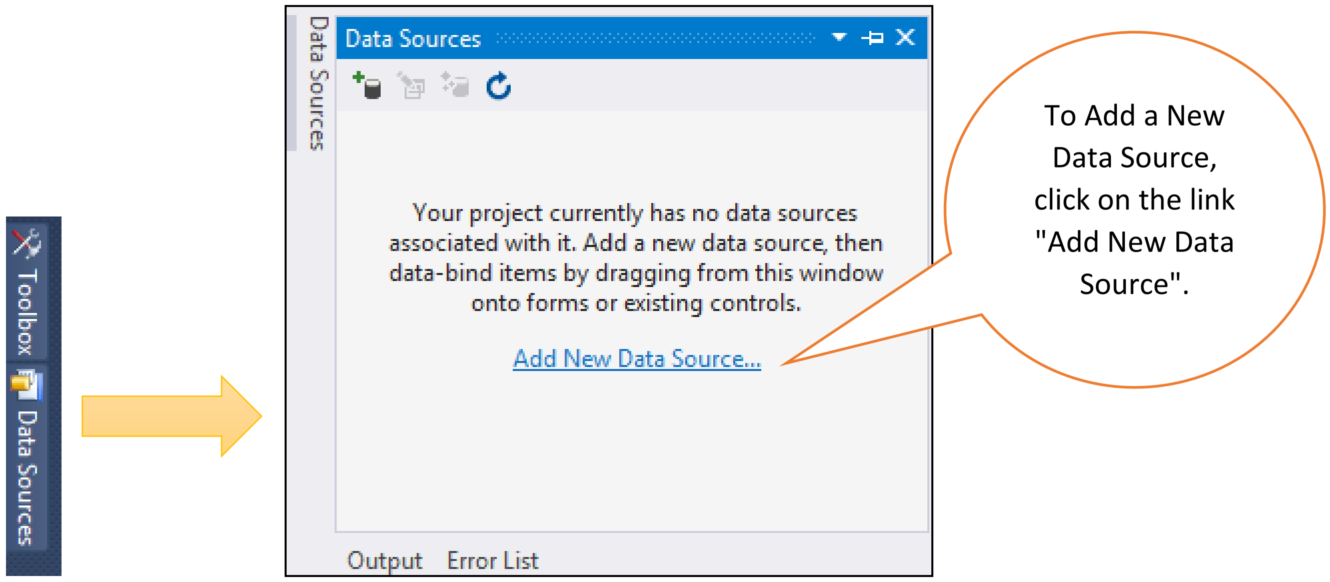
Step 3: Populate the table: Add sample records in the table.

Step 4: Create a VB.Net Application: Open Visual Studio and Create a Visual Basic Application project and Save it as ***"DBaccess"***.

Step 5: Design the user interface: To design the form, follow the image below.



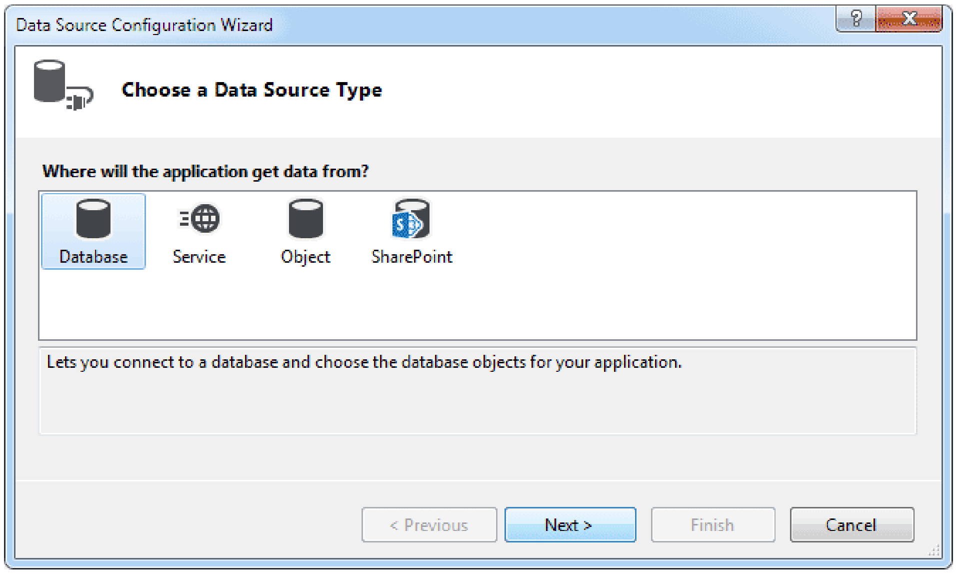
Step 6: Add New Data Source: click on Data Sources on the left of the Toolbox (If you can't see the tab, click **View > Other Windows > Data Sources**):


Toolbox
Data Sources
Data Sources
Data Sources
Output
Error List
To Add a New Data Source, click on the link "Add New Data Source".

Your project currently has no data sources associated with it. Add a new data source, then data-bind items by dragging from this window onto forms or existing controls.

Add New Data Source...

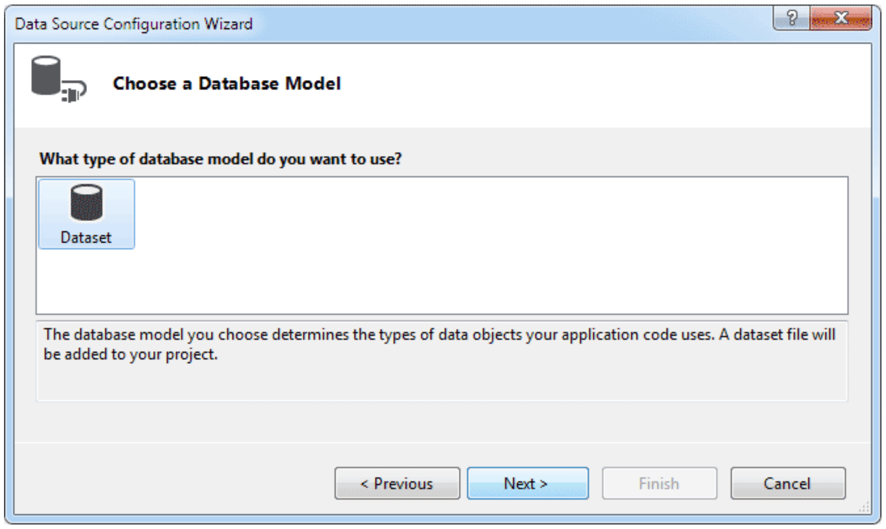
Step 7: Choose a Data Source Type: Select **Database** and **click Next.**



Where will the application get data from?

Lets you connect to a database and choose the database objects for your application.

Select DataSet and click Next.



What type of database model do you want to use?

The database model you choose determines the types of data objects your application code uses. A dataset file will be added to your project.

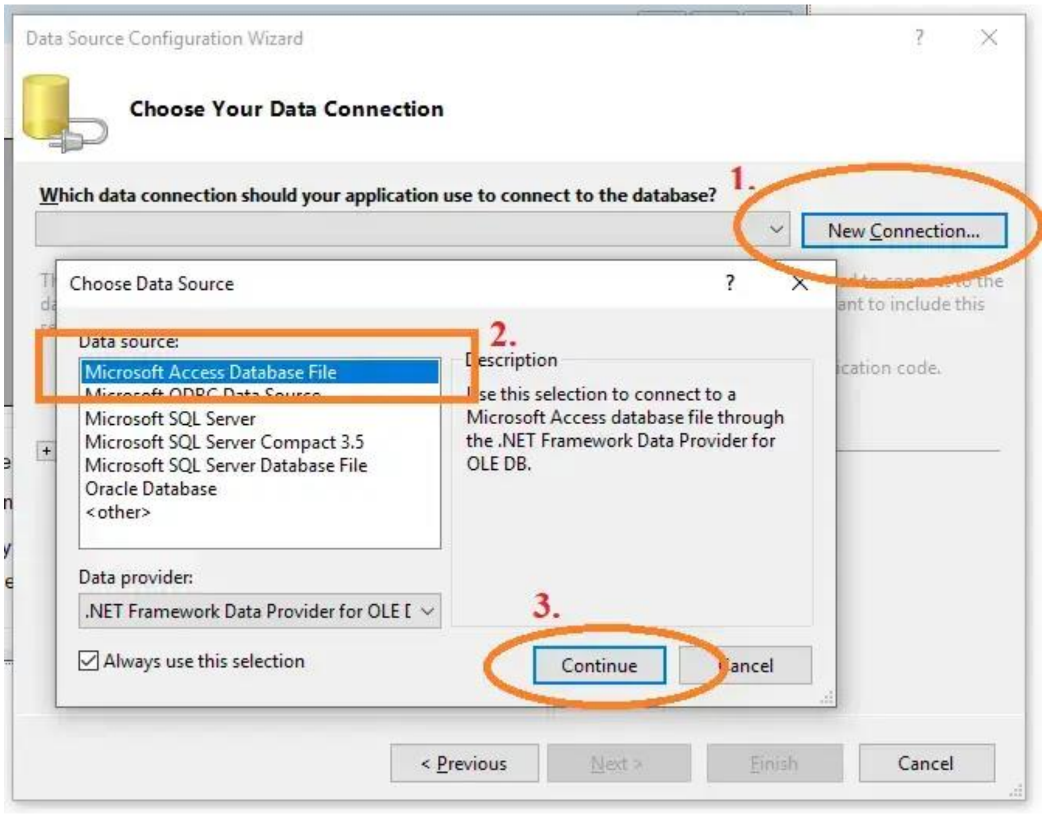
What is DataSet?

- ***The dataset is an in-memory representation of data. It is a disconnected, cached set of records that are retrieved from a database. When a connection is established with the database, the data adapter creates a dataset and stores data in it.***

What is DataTable?

- ***Datatable consists of the DataRow and DataColumn objects. The DataTable objects are case-sensitive.***

Step 8: Choose a Data Source: Click **new Connection** then, Select the **Microsoft AccessDatabase file** and, Click **Continue**. *Follow the image below.*


Data Source Configuration Wizard
Choose Your Data Connection
Which data connection should your application use to connect to the database?
1.
New Connection...
Choose Data Source
Data source:
2.
Microsoft Access Database File
Microsoft ODBC Data Source
Microsoft SQL Server
Microsoft SQL Server Compact 3.5
Microsoft SQL Server Database File
Oracle Database
<other>
Description
Use this selection to connect to a Microsoft Access database file through the .NET Framework Data Provider for OLE DB.
Data provider:
.NET Framework Data Provider for OLE D
Always use this selection
3.
Continue
Cancel
< Previous
Next >
Finish
Cancel

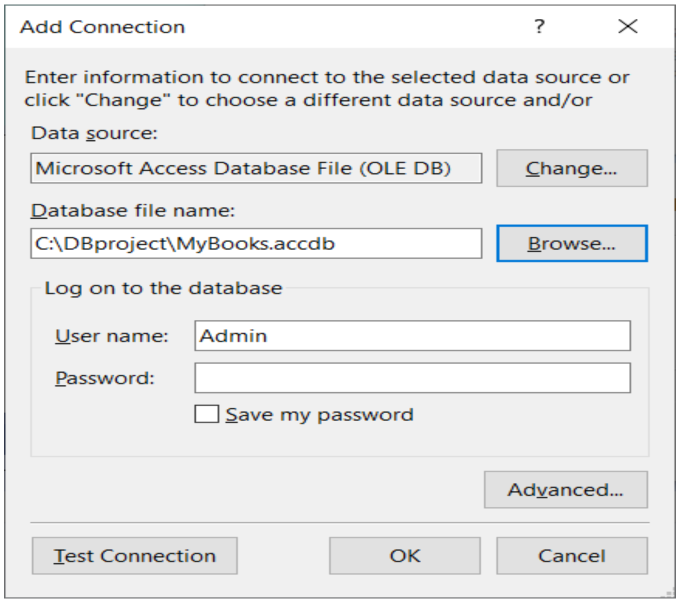
What is OledbConnection?

- ***OleDbConnection is designed for connecting to a wide range of databases, like Microsoft Access and Oracle***.

What is SqlConnection?

- ***SqlConnection is designed for connecting to Microsoft SQL Server***

Step 9: Add Connection : Click the **New Connection** button and another dialogue box pops up - - the Add Connection dialogue box. First, Click the Browse button and navigate to where on computer for Access Database then, Select “**MyBooks.accdb”**, Lastly, Click **Open**.



Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or

Data source: Microsoft Access Database File (OLE DB)

Database file name: C:\DBproject\MyBooks.accdb

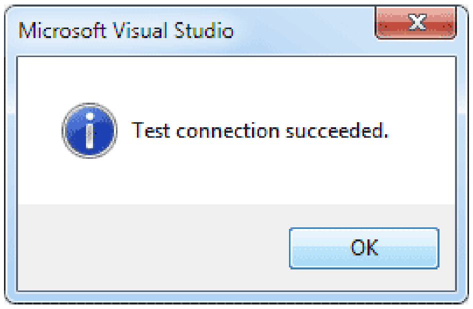
Log on to the database

User name: Admin

Password:

☐ Save my password

Step 10: Test Connection : To test the connection, click the **"Test Connection"** button, and finally, click the **"OK"** button at the side of the **"Cancel"** button.

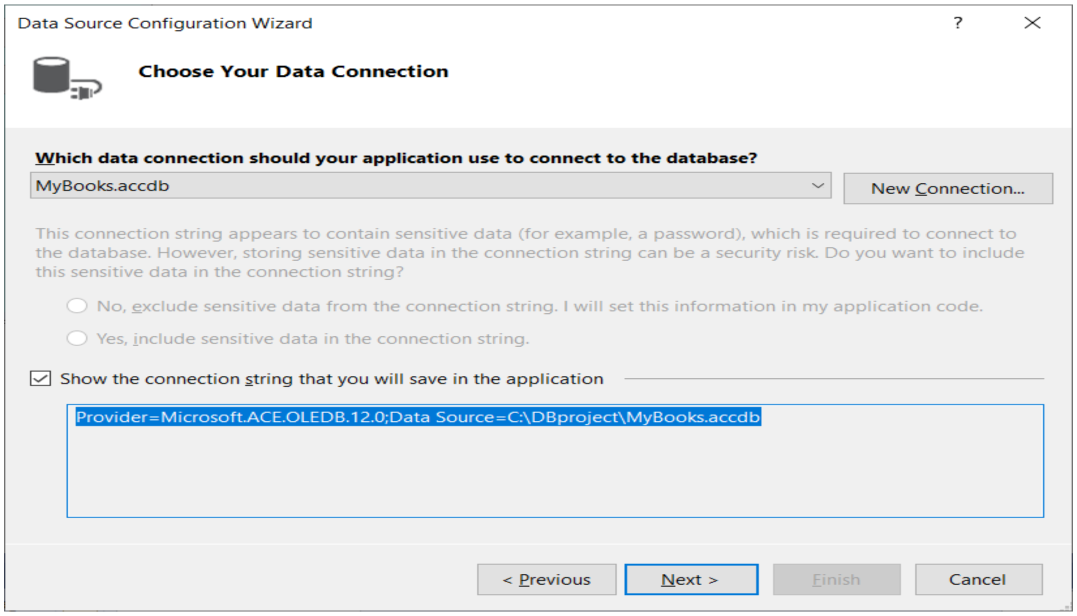


Microsoft Visual Studio

Test connection succeeded.

Click the OK button, then click the OK button on the Add Connection dialogue box as well.

Step 11: Copy the Connection String: Copy the connection string so that we can use this in our next step.



Data Source Configuration Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

MyBooks.accdb

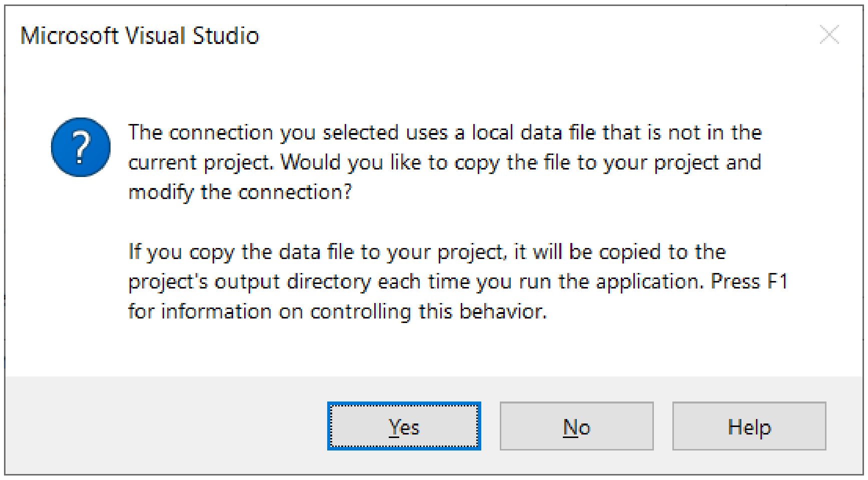
This connection string appears to contain sensitive data (for example, a password), which is required to connect to the database. However, storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

○ No, exclude sensitive data from the connection string. I will set this information in my application code.

○ Yes, include sensitive data in the connection string.

☑ Show the connection string that you will save in the application

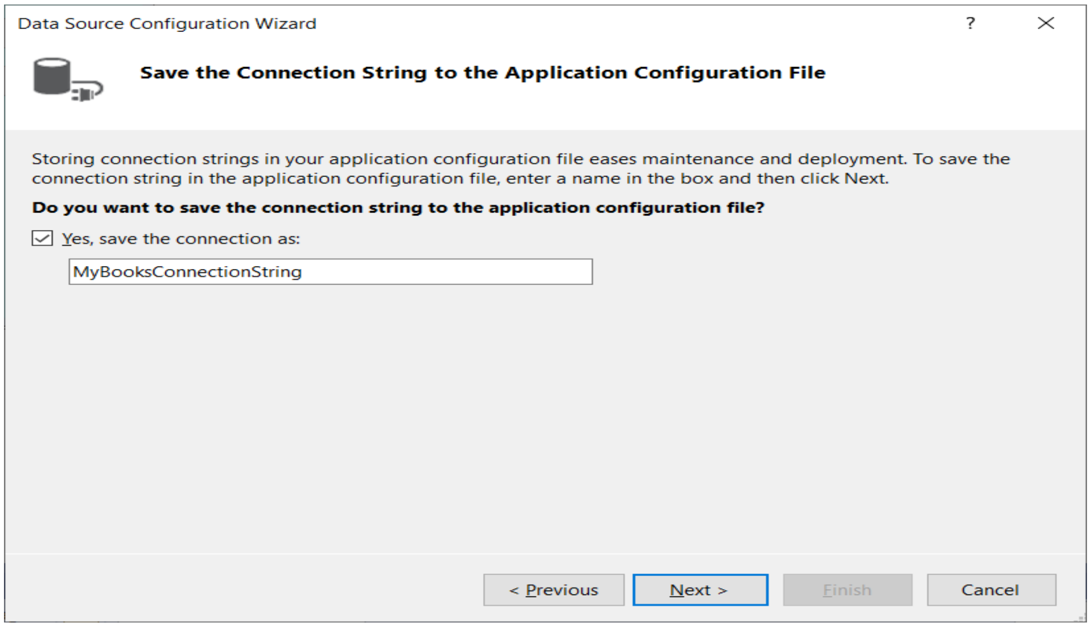
```
Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\DBproject\MyBooks.accdb
```


Microsoft Visual Studio

The connection you selected uses a local data file that is not in the current project. Would you like to copy the file to your project and modify the connection?

If you copy the data file to your project, it will be copied to the project's output directory each time you run the application. Press F1 for information on controlling this behavior.

Make sure there's a tick in the box for "Save the connection", and then click Next.


Data Source Configuration Wizard

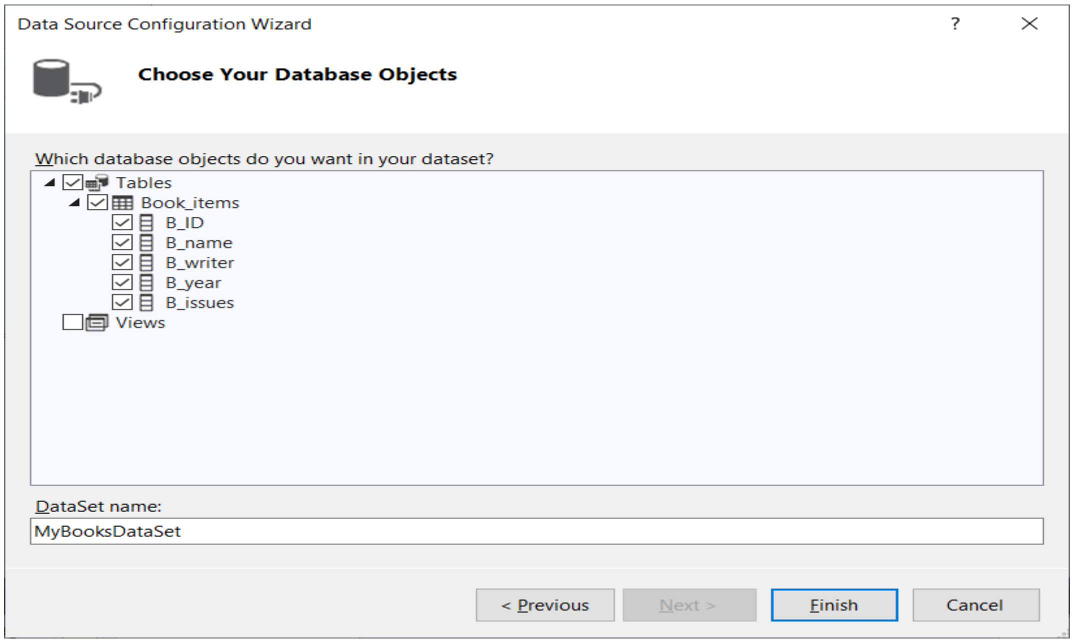
Save the Connection String to the Application Configuration File

Storing connection strings in your application configuration file eases maintenance and deployment. To save the connection string in the application configuration file, enter a name in the box and then click Next.

Do you want to save the connection string to the application configuration file?

[✓] Yes, save the connection as:

MyBooksConnectionString


Data Source Configuration Wizard

Choose Your Database Objects

Which database objects do you want in your dataset?

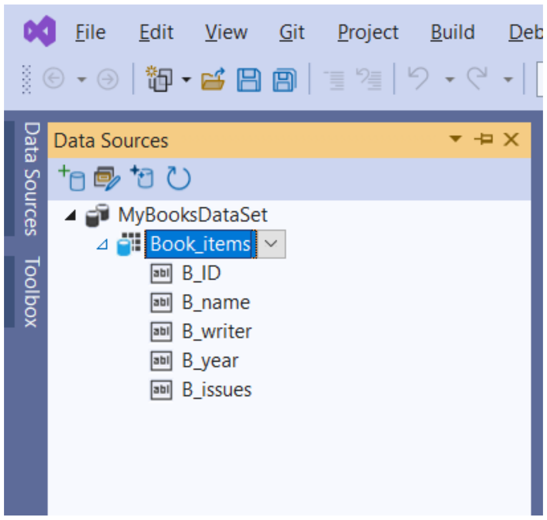
- [✓] Tables
  - [✓] Book_items
    - [✓] B_ID
    - [✓] B_name
    - [✓] B_writer
    - [✓] B_year
    - [✓] B_issues
- [ ] Views

DataSet name:

MyBooksDataSet

Step 12: Start Coding: In this final step, we will start adding functionality to our vb.net program by adding some functional codes.

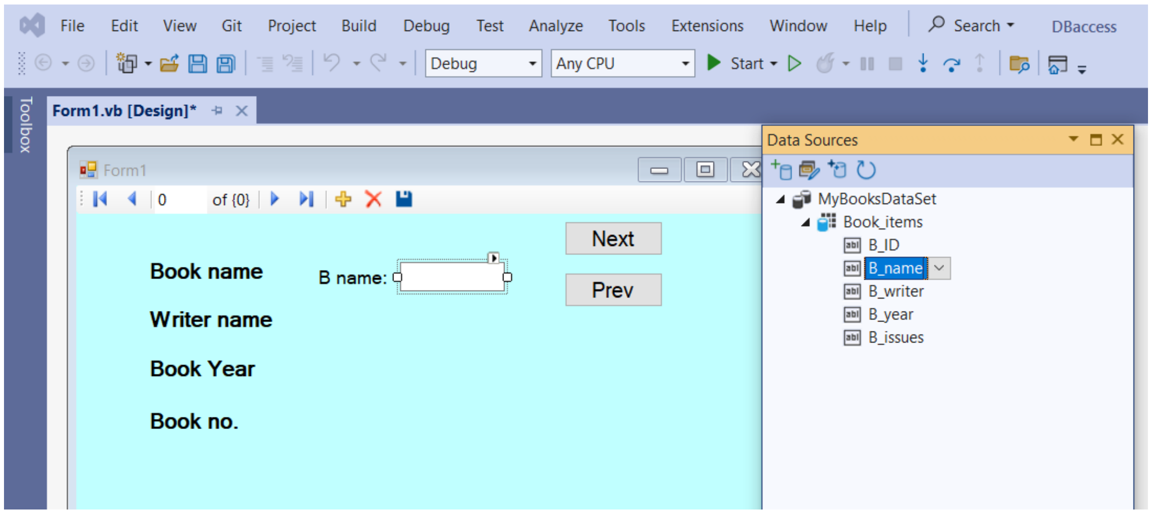
The Data Sources area now displays information about the database. Click the arrow symbol next to **MyBooksDataSet**:



All the Fields in the Book_items table are now showing.

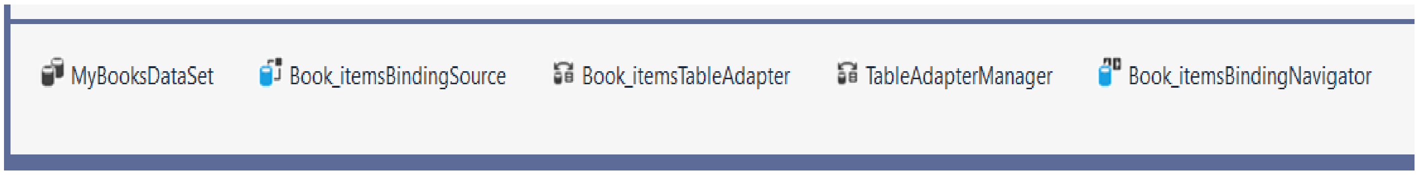
To add a Field to Form:

- click on one in the list. Hold down left mouse button, and drag it over to form:



- In the image above, the B_name field is being dragged on the Form. Your mouse cursor will change shape. When your Field is over the Form, let go of your left mouse button. A textbox and a label will be added:

There are two other things to notice: a navigation bar appears at the top of the form, and a lot of strange objects have appeared in the object area at the bottom:



Note

If using Visual Studio to connect to databases using OLEDB or ODBC data providers, note that versions of Visual Studio prior to Visual Studio 2022 are all 32-bit processes. This means some of the data tools in Visual Studio will only be able to connect to OLEDB or ODBC databases using 32-bit data providers. This includes the Microsoft Access 32-bit OLEDB data provider.

If using Visual Studio 2022 to connect to databases,Visual Studio 2022 is a 64-bit process. This means some of the data tools in Visual Studio will not be able to connect to OLEDB or ODBC databases using 32-bit data providers. Then use an earlier version of Visual Studio that is still a 32-bit process for 32-bit database. The last version of Visual Studio that was a 32-bit process was Visual Studio 2019.

If plan on converting the project to be a 64-bit process, it's recommended to use the 64-bit Microsoft Access database Engine, also called Access Connectivity Engine (ACE). See **OLE DB Provider for Jet and ODBC driver are 32-bit versions only** for more information.

.NET Framework data providers

A .NET Framework data providers are lightweight, creating a minimal layer between the data source and code, increasing performance without sacrificing functionality.

A .NET Framework data provider is used for connecting to a database, executing commands, and retrieving results. Those results are either:

- processed directly, placed in a DataSet in order to be exposed to the user as needed,
- combined with data from multiple sources,
- or remoted between tiers.

The data providers that are included in .NET Framework.

1. **.NET Framework Data Provider for OLE DB**

- For data sources exposed by using OLE DB. The .NET Framework Data Provider for OLE DB supports both local and distributed transactions.
- The .NET Framework Data Provider for OLE DB classes are located in the System.Data.OleDb namespace.
- It recommended for single-tier applications that use Microsoft Access databases. Use of an Access database for a middle-tier application is not recommended.

2. .NET Framework Data Provider for ODBC

- Uses the System.Data.Odbc namespace. The .NET Framework Data Provider for ODBC (Odbc) uses the native ODBC Driver Manager (DM) to enable data access.
- The ODBC data provider supports both local and distributed transactions.

3. .NET Framework Data Provider for SQL Server

- Provides data access for Microsoft SQL Server. Uses the System.Data.SqlClient namespace.
- The .NET Framework Data Provider for SQL Server (SqlClient) uses its own protocol to communicate with SQL Server.
- It is lightweight and performs well because it is optimized to access a SQL Server directly without adding an OLE DB or Open Database Connectivity (ODBC) layer.
- The .NET Framework Data Provider for SQL Server supports both local and distributed transactions.
- Recommended for middle-tier applications that use Microsoft SQL Server, and for single-tier applications that use Microsoft Database Engine (MSDE) or SQL Server
- Recommended over use of the OLE DB provider for SQL Server (SQLOLEDB) with the .NET Framework Data Provider for OLE DB.

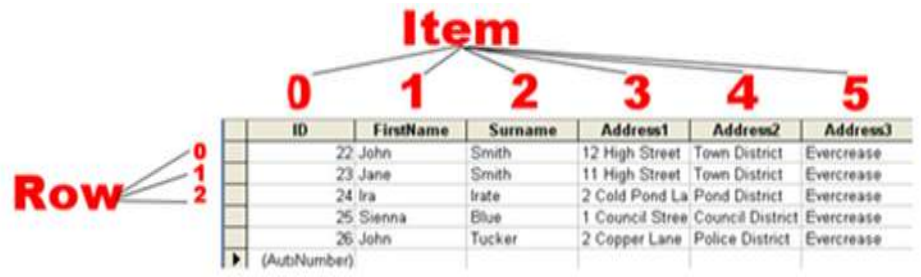
4. .NET Framework Data Provider for Oracle

- For Oracle data sources and uses the System.Data.OracleClient namespace.
- The .NET Framework Data Provider for Oracle supports Oracle client software version 8.1.7 and later.
- The data provider supports both local and distributed transactions.

Steps for execute Select using OLE DB

OLE stands for Object Linking and Embedding, and its basically a lot of objects (COM objects) bundled together that allow you to connect to data sources in general

1- Setting a Connection String (pass two things to it Provider; and Data Source, *If database was password and user name protected, add these two parameters also*)
2- Select a data providers to connect to a modern Access database
3- Adding a path and a database file name to the Data Source.
4- Use the Open method of the Connection Object
5- Dim a DataSet to hold a copy of the information from the database(Each imaginary row of the DataSet represents a Row of information in your Access database. And each imaginary column represents a Column of information in your Access database)
6- Setup Data Adapter (The Data Adapter contacts Connection Object, and then executes a query . The results of that query are then stored in the DataSet.)
7- Filling the DataSet
8- Use the Close method of the Connection Object
9- Displaying the Data in the DataSet



ID	FirstName	Surname	Address1	Address2	Address3
22	John	Smith	12 High Street	Town District	Evercrease
23	Jane	Smith	11 High Street	Town District	Evercrease
24	Ira	Irate	2 Cold Pond La	Pond District	Evercrease
25	Sienna	Blue	1 Council Stree	Council District	Evercrease
26	John	Tucker	2 Copper Lane	Police District	Evercrease
(AutoNumber)					

The image shows which are the Rows and which are the Items in the Access database Table. So the Items go down and the Rows go across.

For Insert, Update or Delete use Command from data providers

1- Create sql command as string
2- Setup OleDbCommand (contacts Connection Object, and then executes a query

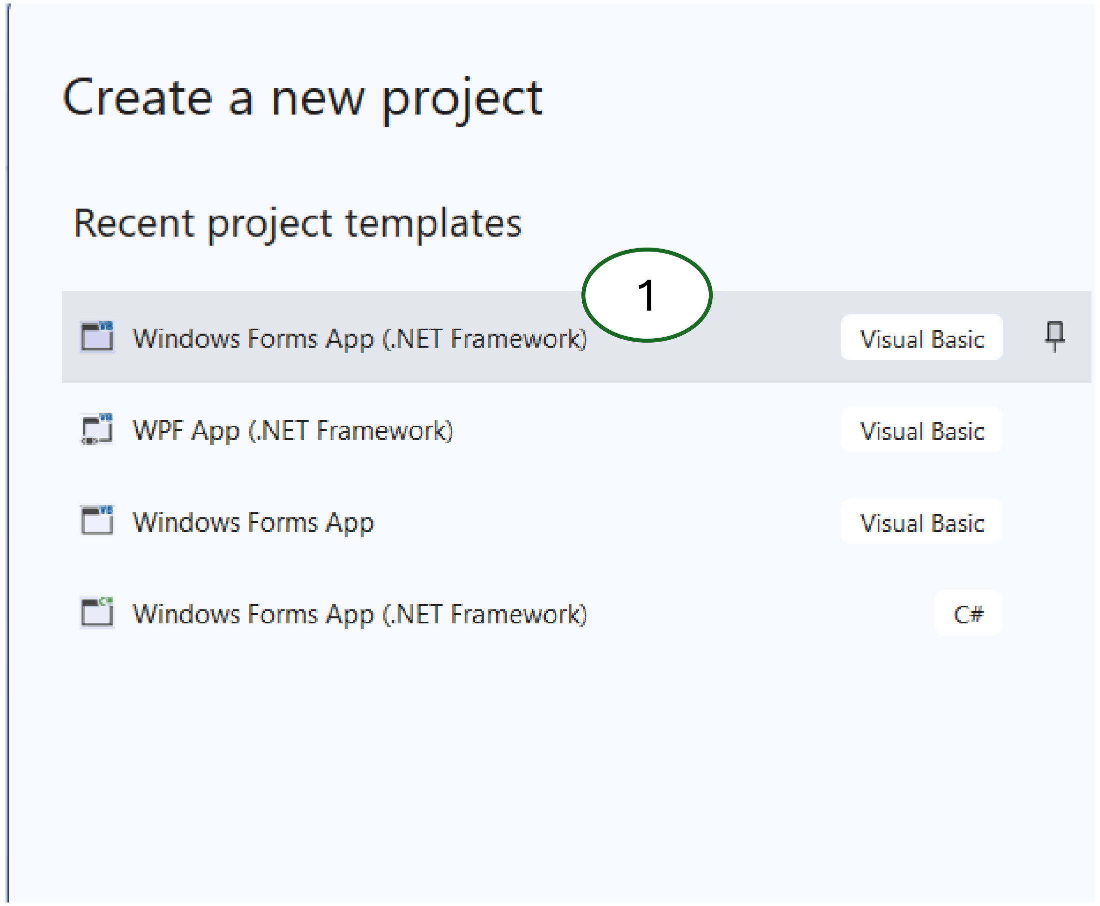
VB.Net 2022
Connect to SQL Server Database

Steps to create Project to connect to sql-server database

Using SQLClient Data Provider

1-Create new VB.Net Project

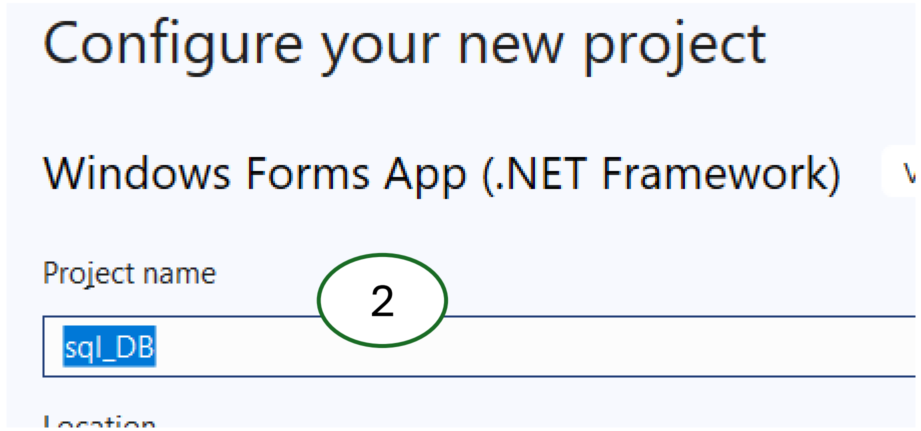
- انشاء مشروع جديد



Create a new project

Recent project templates

Windows Forms App (.NET Framework)	Visual Basic
WPF App (.NET Framework)	Visual Basic
Windows Forms App	Visual Basic
Windows Forms App (.NET Framework)	C#



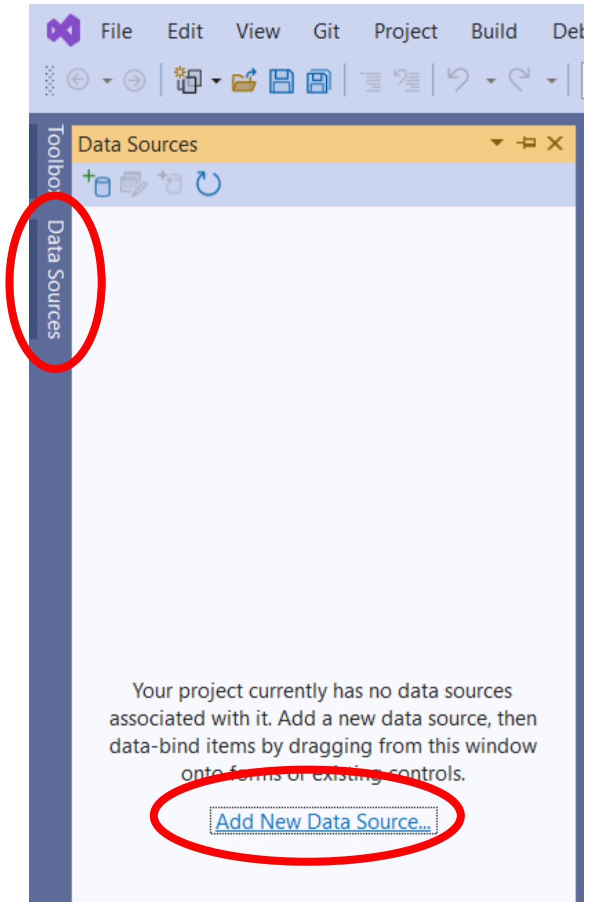
Configure your new project

Windows Forms App (.NET Framework)

Project name

sql_DB

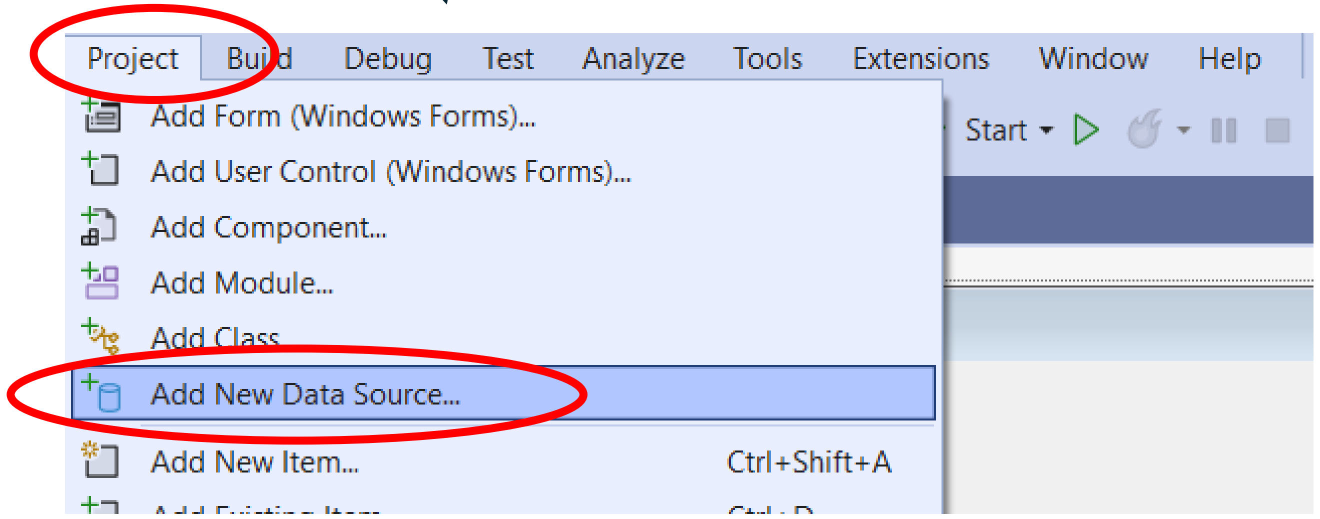
2-Add new Data Source



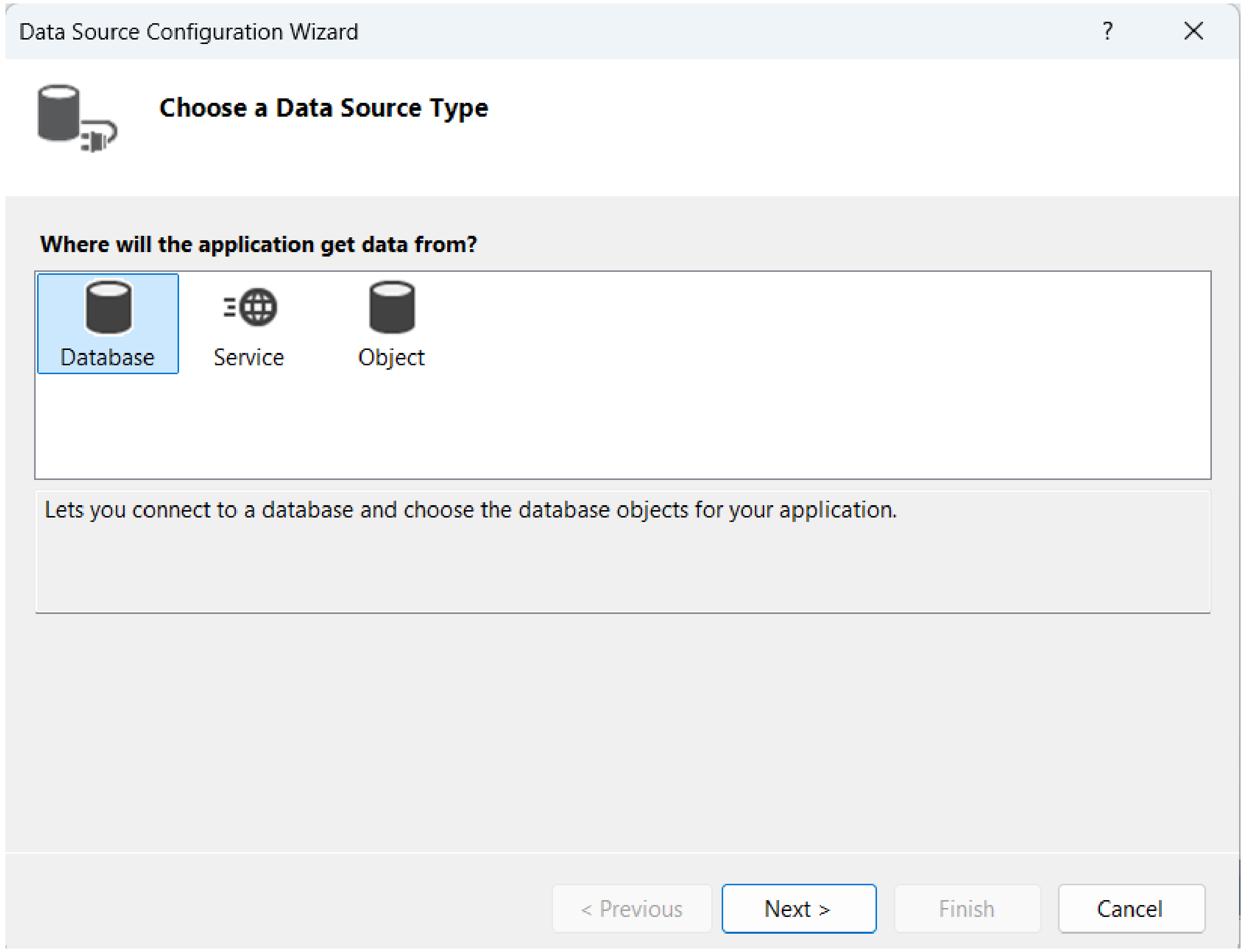
Your project currently has no data sources associated with it. Add a new data source, then data-bind items by dragging from this window onto forms or existing controls.

Add New Data Source...

- اضافة مصدر البيانات مستخدم واحدة من الطريقتين كما في الصور ادناه



3-Choose a Data Source Type



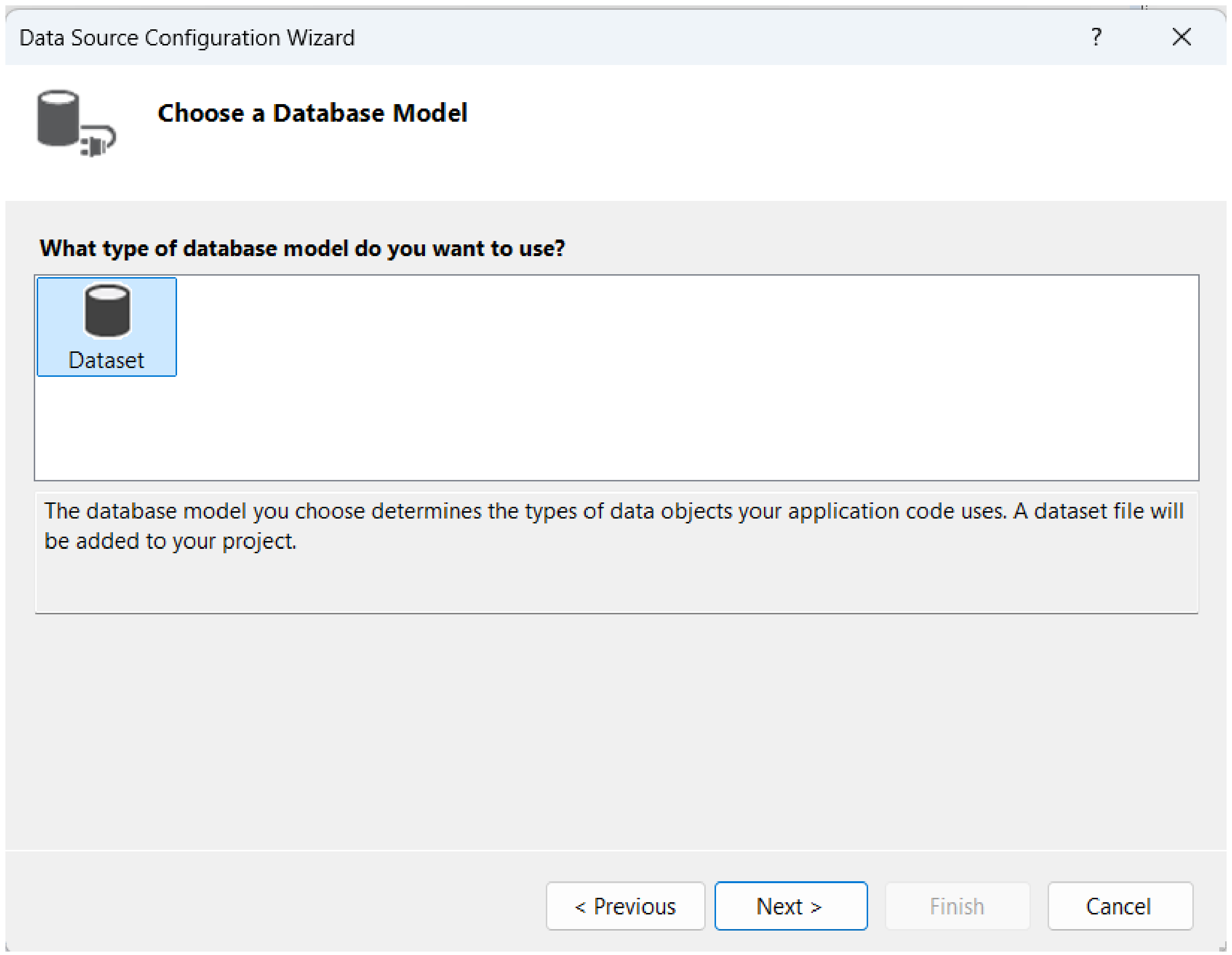
Choose a Data Source Type

Where will the application get data from?

Lets you connect to a database and choose the database objects for your application.

- اختيارنوع مصدر البيانات
- نختار قاعدة البيانات database
- ثم نضغط على Next

4-Choose a Database Model


Data Source Configuration Wizard

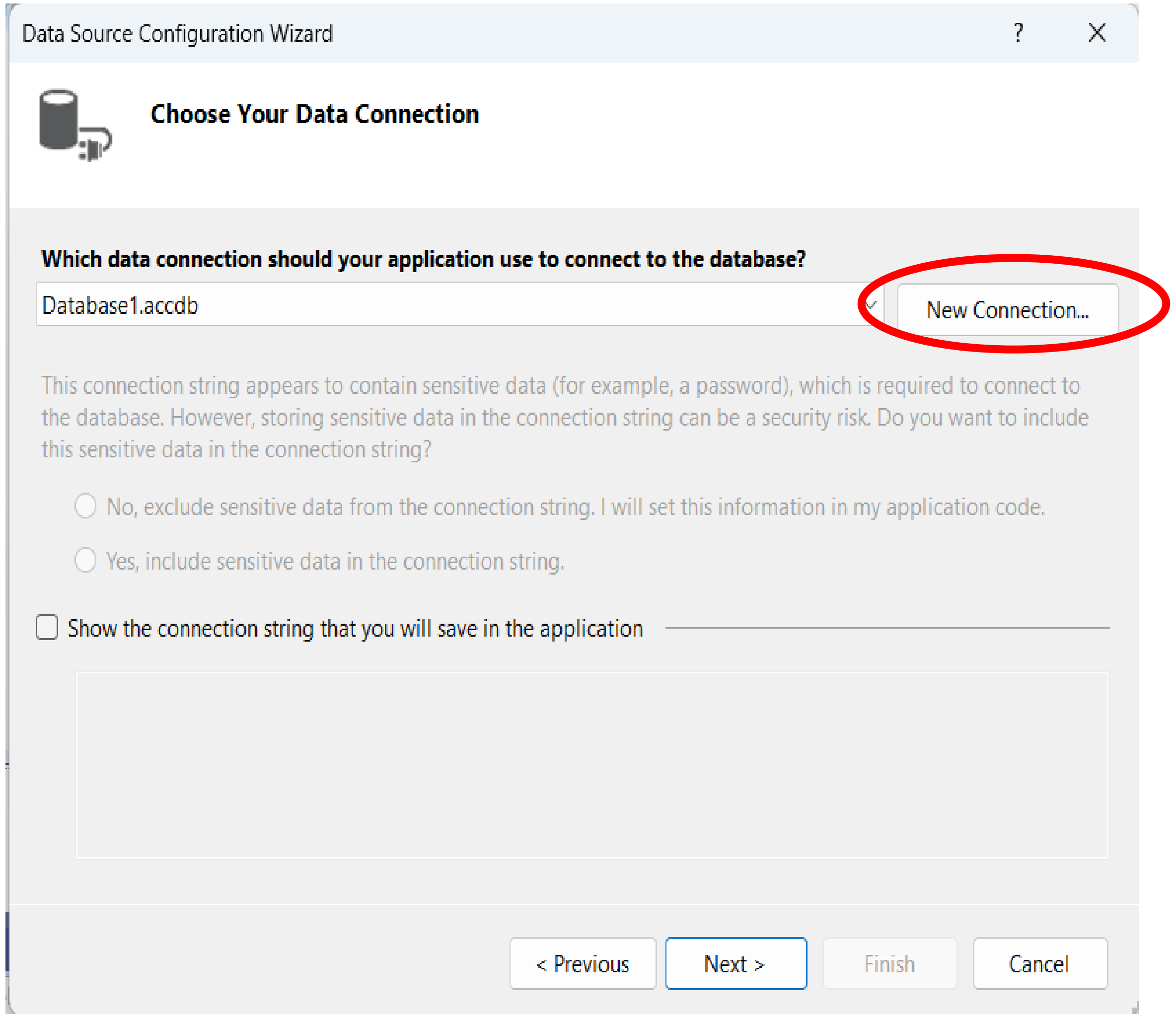
Choose a Database Model

What type of database model do you want to use?

The database model you choose determines the types of data objects your application code uses. A dataset file will be added to your project.

- نختار طريقة التعامل مع قاعدة البيانات
- سوف نستخدم نظام dataset
- نضغط Next

5-Choose Data Connection Type



Data Source Configuration Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

Database1.accdb

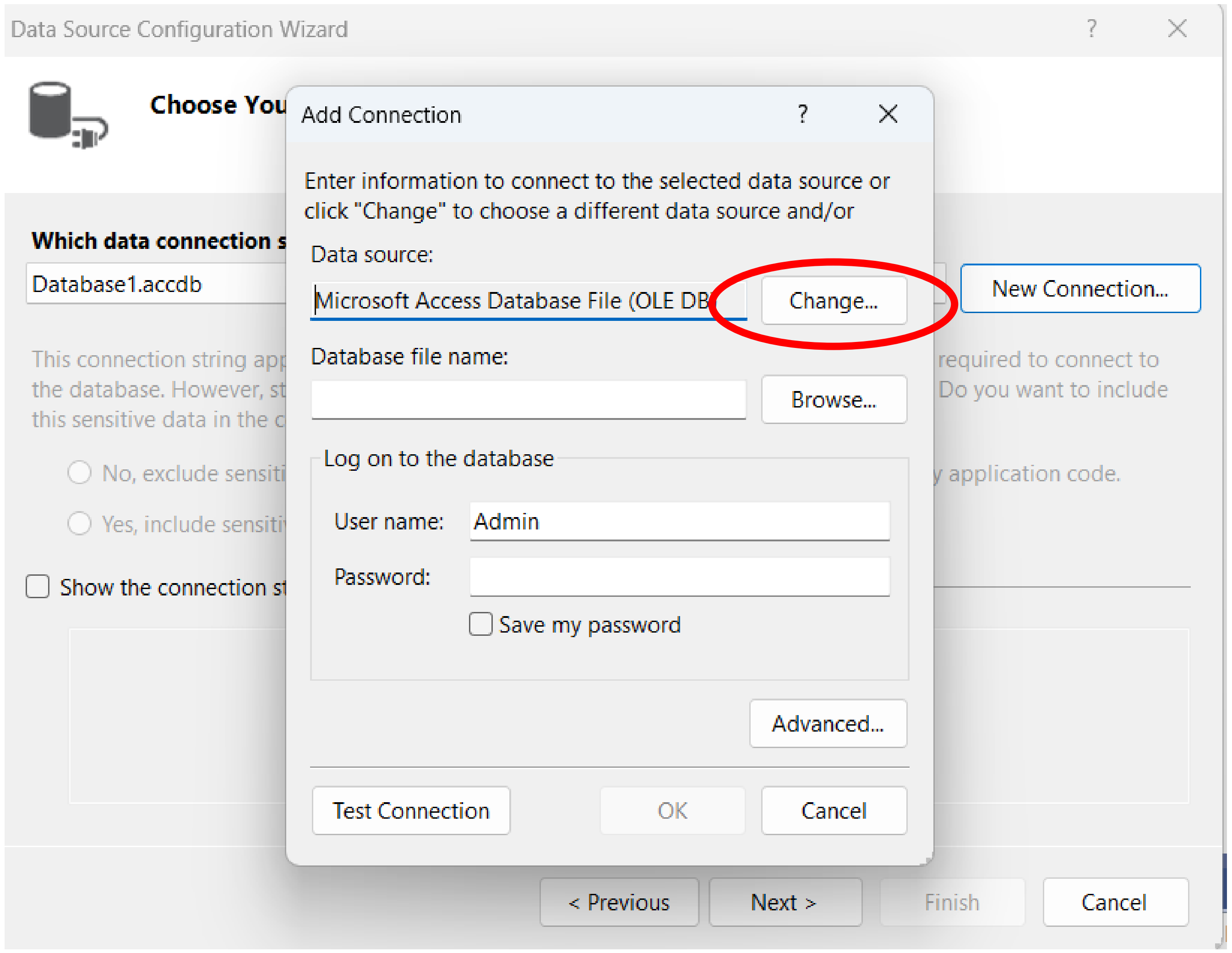
New Connection...

This connection string appears to contain sensitive data (for example, a password), which is required to connect to the database. However, storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

- No, exclude sensitive data from the connection string. I will set this information in my application code.
- Yes, include sensitive data in the connection string.

☐ Show the connection string that you will save in the application

- نضغط على ايكونة New Connection
- من اجل بناء قناة الاتصال بقاعدة البيانات
- وهي هنا تخزن في Connection String


Data Source Configuration Wizard

Choose You

Which data connection s

Database1.accdb

New Connection...

This connection string app ... required to connect to the database. However, st ... Do you want to include this sensitive data in the c

No, exclude sensiti ... y application code.

Yes, include sensiti

Show the connection st

Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or

Data source:

Microsoft Access Database File (OLE DB

Change...

Database file name:

Browse...

Log on to the database

User name: Admin

Password:

Save my password

Advanced...

Test Connection

OK

Cancel

< Previous

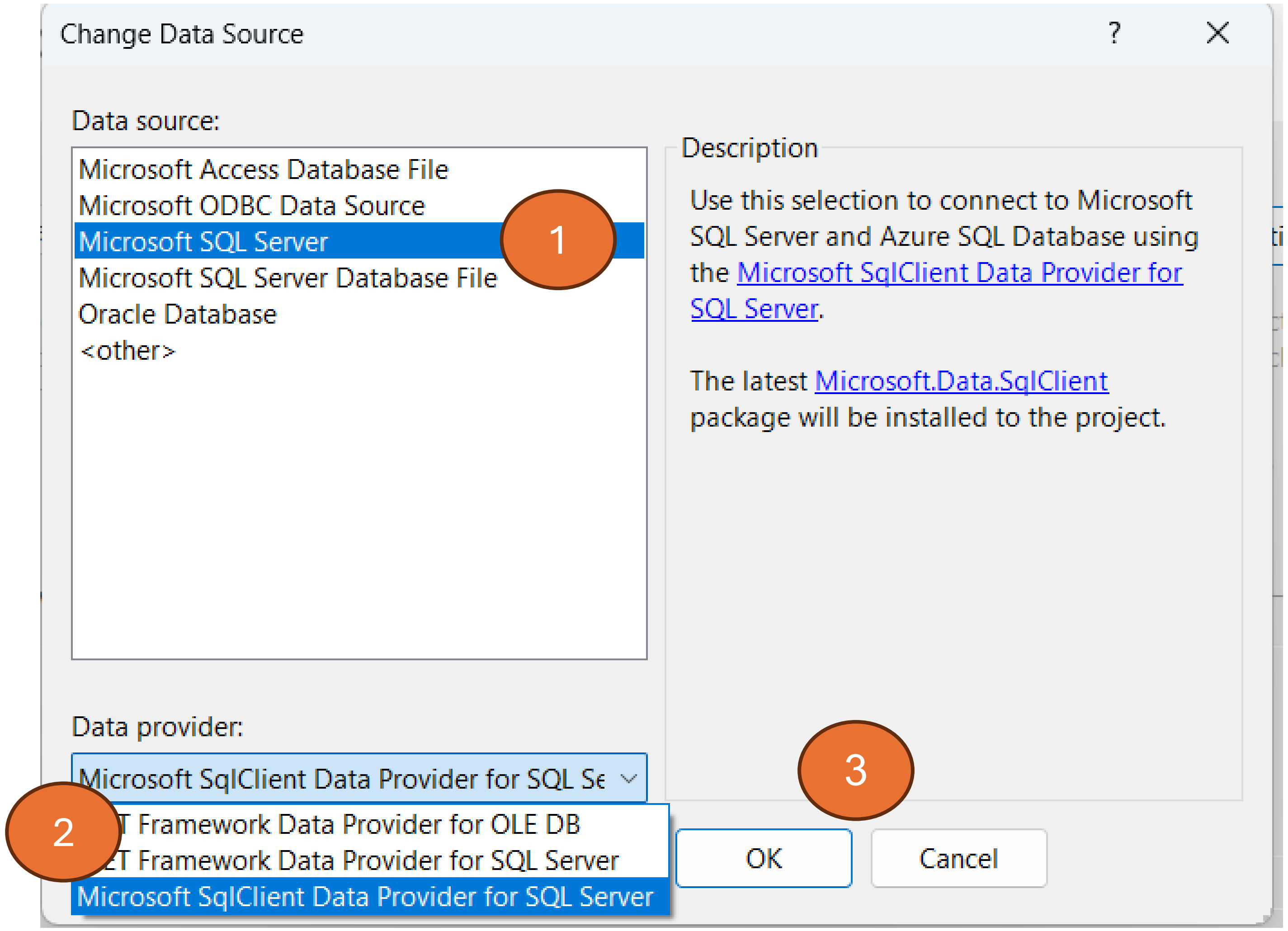
Next >

Finish

Cancel

- تظهر نافذة اضافة اتصال
- نضغط على ايكونة Change

6-Select Data Source and Data Provider



Data source:

Microsoft Access Database File
Microsoft ODBC Data Source
Microsoft SQL Server
Microsoft SQL Server Database File
Oracle Database
<other>

Description

Use this selection to connect to Microsoft SQL Server and Azure SQL Database using the Microsoft SqlClient Data Provider for SQL Server.

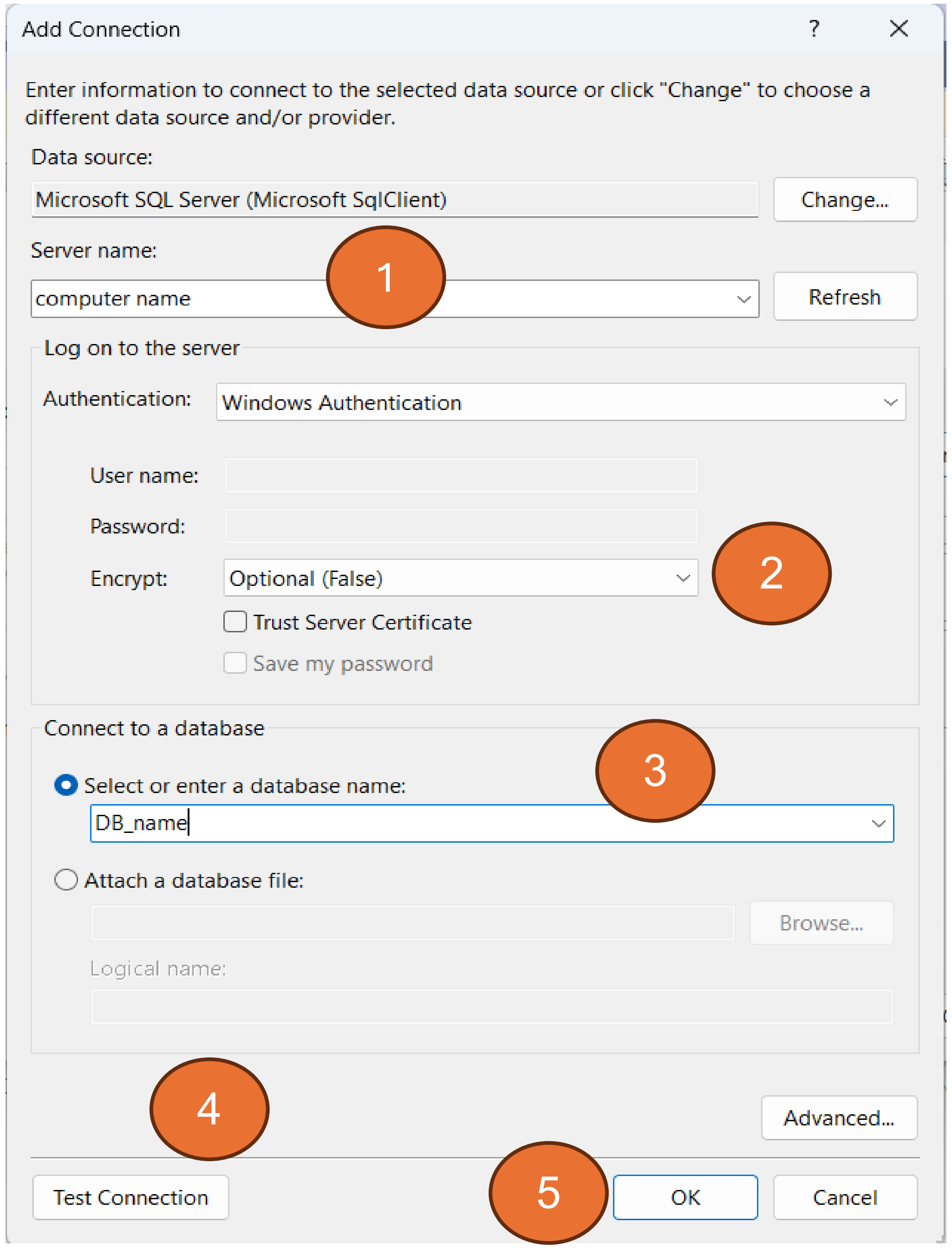
The latest Microsoft.Data.SqlClient package will be installed to the project.

Data provider:

Microsoft SqlClient Data Provider for SQL Se

T Framework Data Provider for OLE DB
ET Framework Data Provider for SQL Server
Microsoft SqlClient Data Provider for SQL Server

1. نختار مصدر البيانات هنا هو SQl Server
2. ثم نختار من القائمة اسفل النافذة نوع المجهز وسوف نستخدم SQLClient Data Provider
3. ثم نضغط Ok


Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: Microsoft SQL Server (Microsoft SqlClient)

Server name: computer name

Log on to the server

Authentication: Windows Authentication

User name:

Password:

Encrypt: Optional (False)

☐ Trust Server Certificate

☐ Save my password

Connect to a database

● Select or enter a database name: DB_name

○ Attach a database file:

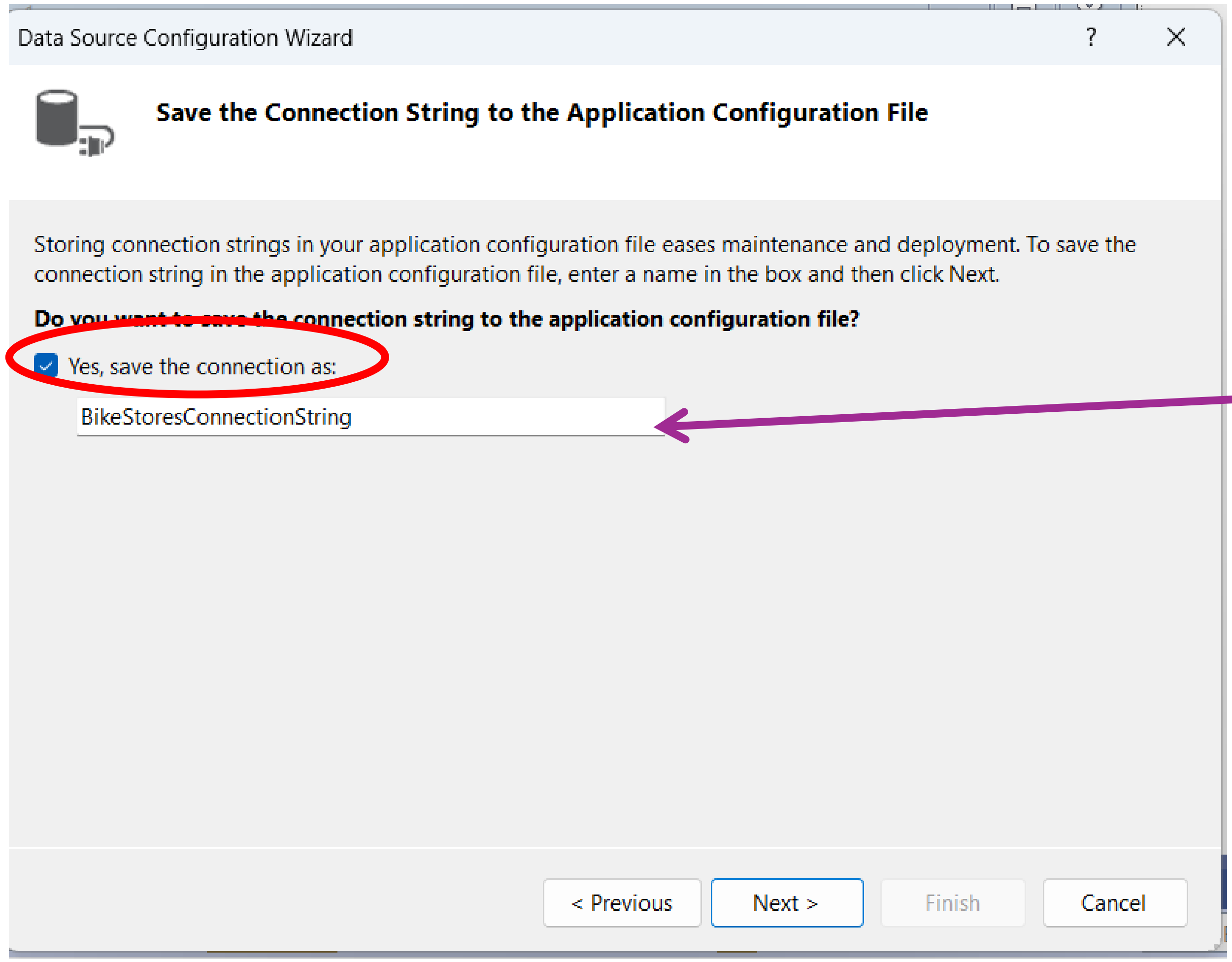
Logical name:

7-Create the Connection

- اضافة تفاصيل الاتصال

1. ادخال اسم جهاز الحاسوب او عنوان IP الخاص بك
1. تحديد نوع التشفير ليكون اختياري وغير مشفر
2. اختيار اسم قاعدة البيانات SQL server
3. فحص الاتصال عن طريق Test Connection
4. نضغط OK
5. ثم نضغط على Next الى ان نصل النافذة في الشريحة التالية

8-Save the Connection String



Data Source Configuration Wizard

Save the Connection String to the Application Configuration File

Storing connection strings in your application configuration file eases maintenance and deployment. To save the connection string in the application configuration file, enter a name in the box and then click Next.

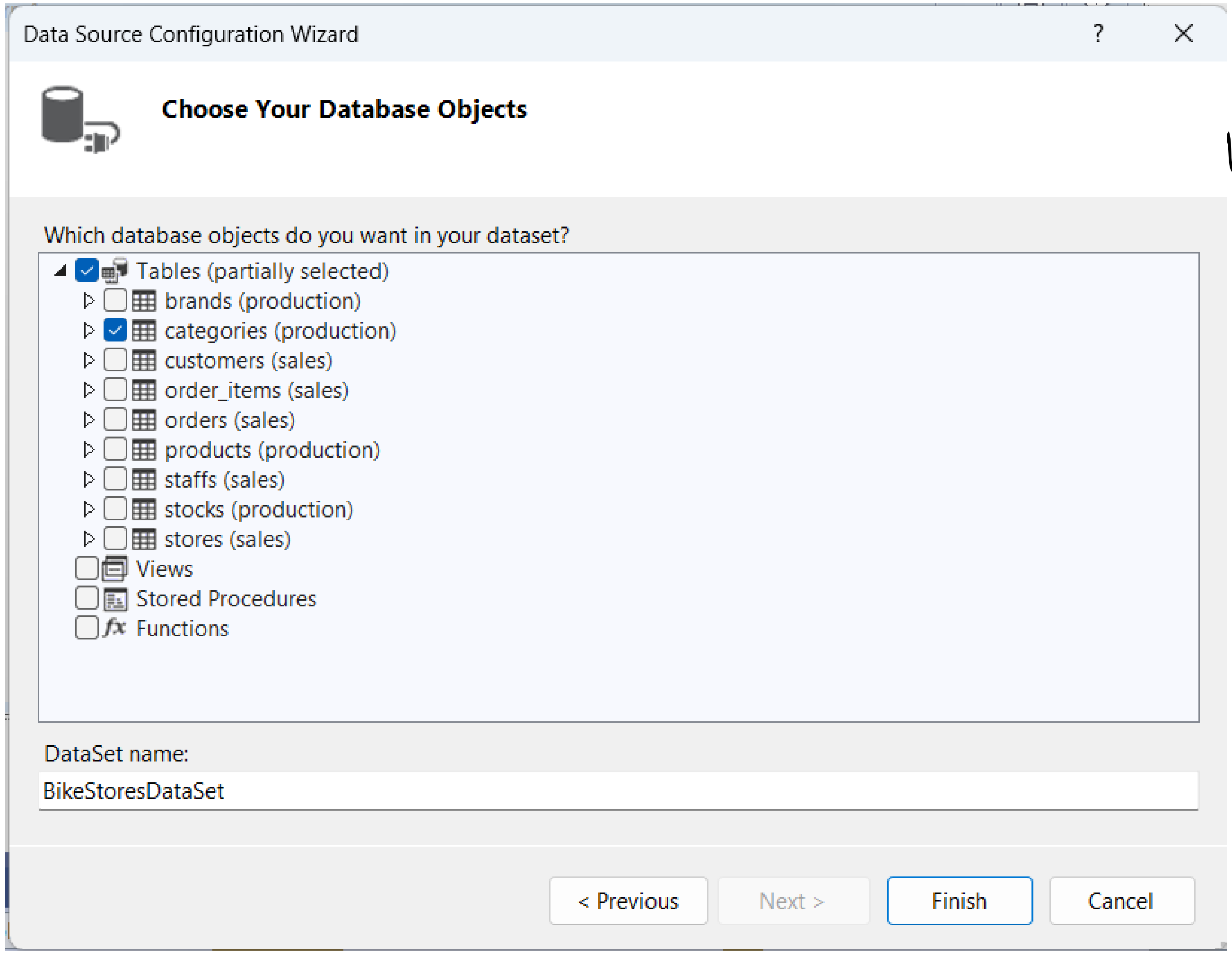
Do you want to save the connection string to the application configuration file?

☑ Yes, save the connection as:

BikeStoresConnectionString

- نؤشر على ,Yes
- نلاحظ ان اسم قناة الاتصال يتكون من قاعدة البيانات + ConnectionString
- ثم نضغط على Next

9-Choose Database Objects (Tables and columns)


Data Source Configuration Wizard
?
Choose Your Database Objects

Which database objects do you want in your dataset?

- [✓] Tables (partially selected)
  - [ ] brands (production)
  - [✓] categories (production)
  - [ ] customers (sales)
  - [ ] order_items (sales)
  - [ ] orders (sales)
  - [ ] products (production)
  - [ ] staffs (sales)
  - [ ] stocks (production)
  - [ ] stores (sales)
- [ ] Views
- [ ] Stored Procedures
- [ ] Functions

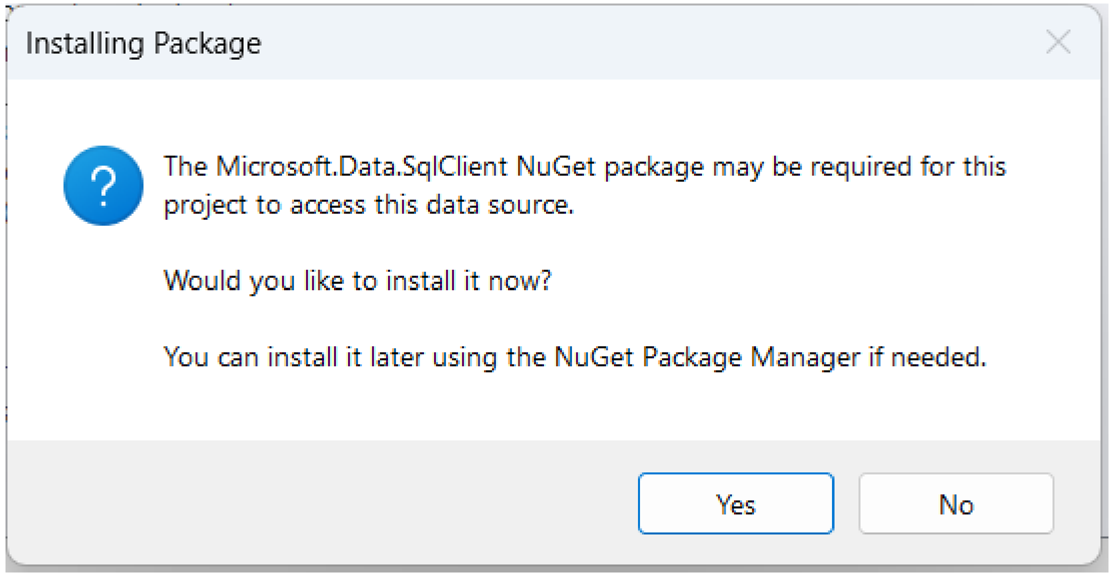
DataSet name: BikeStoresDataSet

< Previous
Next >
Finish
Cancel

- نختار اسم الجدول او الجداول التي
- نريد الاتصال بها ونحدد الاعمدة ايضا
- نضغط على Finish

يجب ان يكون جهاز الحاسوب متصل بالانترنت في هذه الخطوة لاكمال تنصيب الحزم المطلوبة في الخطوة التالية

10-Installing Database Package


Installing Package

The Microsoft.Data.SqlClient NuGet package may be required for this project to access this data source.

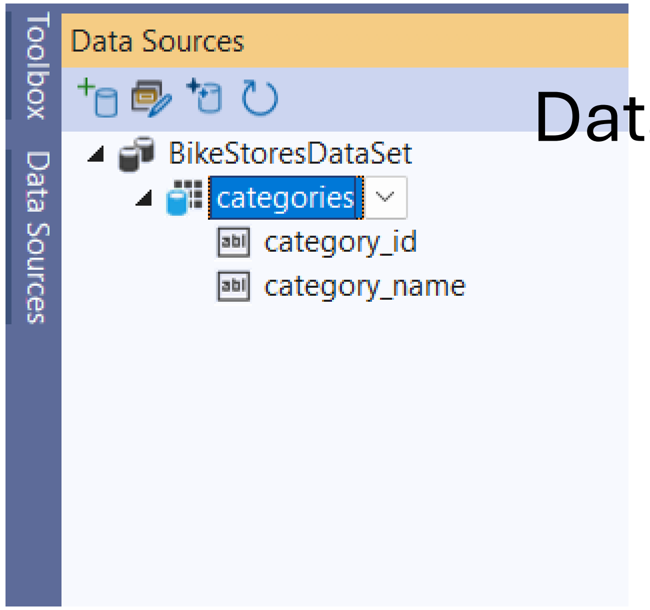
Would you like to install it now?

You can install it later using the NuGet Package Manager if needed.

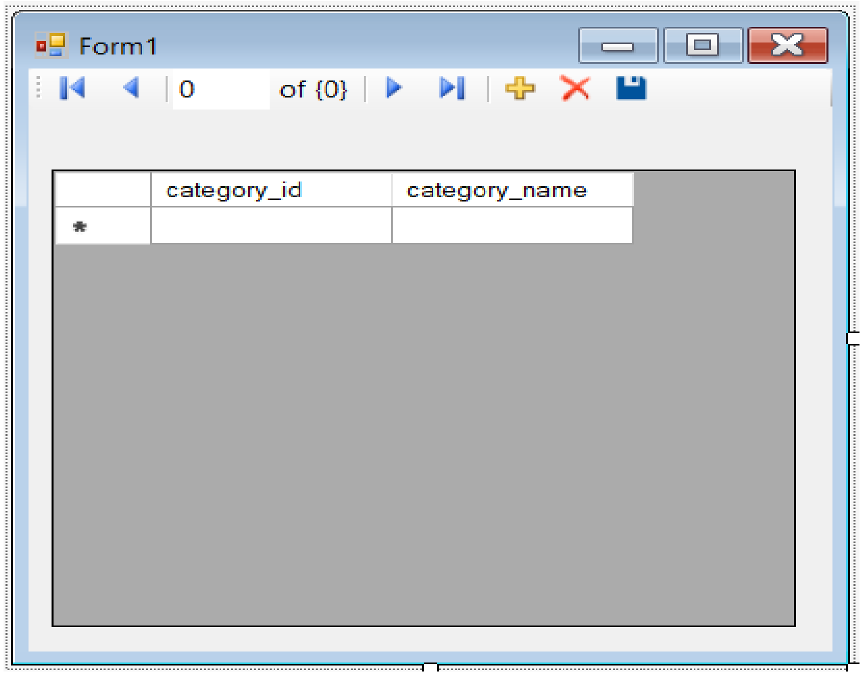
نحتاج الى تنصيب مجموعة من حزم الكود للتعامل مع قواعد البيانات التي حددناها عندما تظهر هذه الرسالة نختار Yes و سوف يقوم البرنامج بتنصيب الحزم التي يحتاجها لذلك سوف نحتاج الى اتصال انترنت في هذه الخطوة

يجب ان يكون جهاز الحاسوب متصل بالانترنت قبل هذه الخطوة لاكمال تنصيب الحزم المطلوبة

11-View Data to the Form

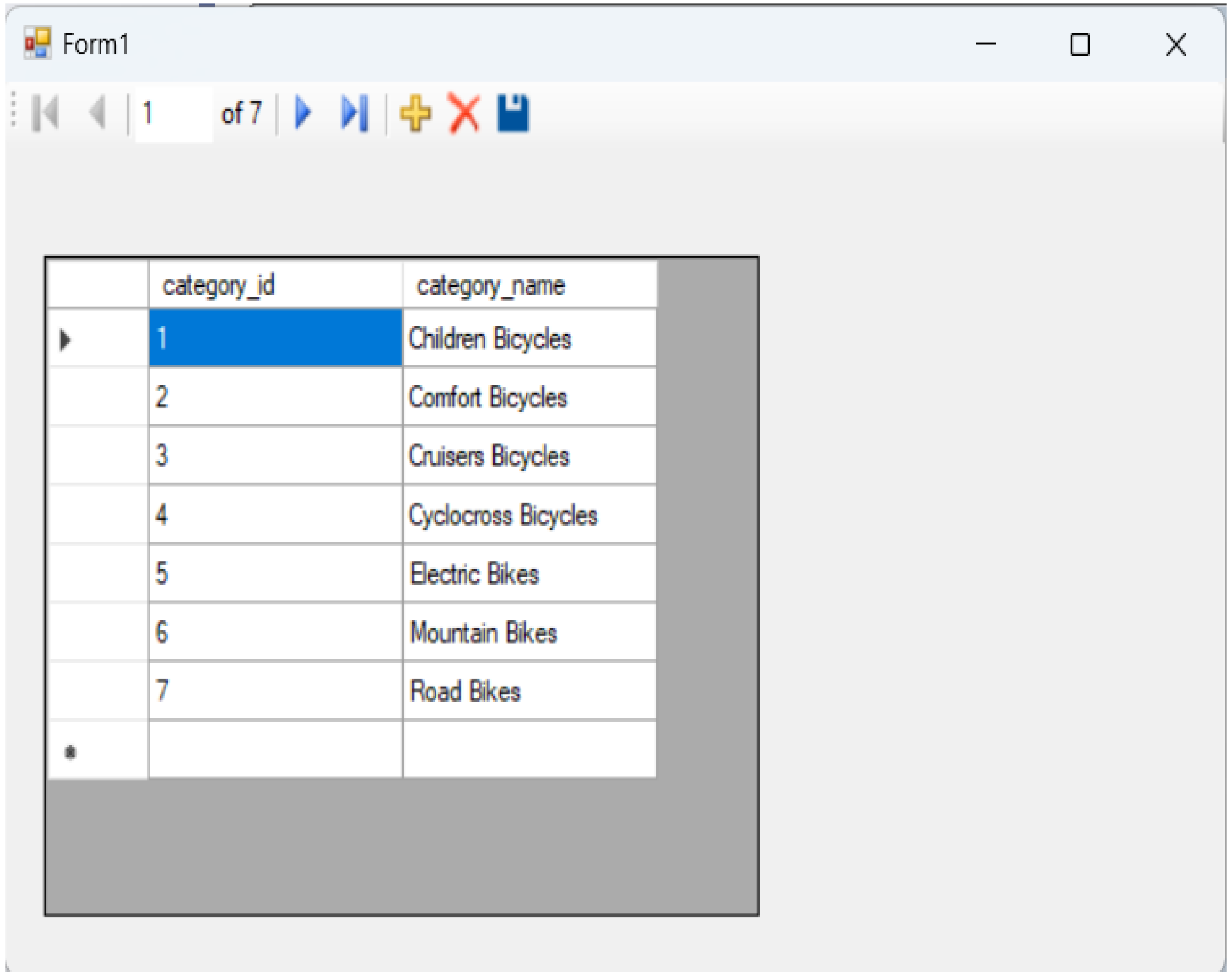


- بعد انتهاء عملية انشاء قناة الاتصال نقوم بالانتقال الى Data Source
- نختار الجدول المطلوب عرضه على نافذة Form
- نضيف الحقول عن طريق السحب والافلات على سطح الفوررم
- كما في الشكل التالي



بعد تنصيب الحزم المطلوبة للاتصال في الخطوة السابقة
حاليا في هذه الخطوات لا نحتاج انترنت

Run the form



	category_id	category_name
▶	1	Children Bicycles
	2	Comfort Bicycles
	3	Cruisers Bicycles
	4	Cyclocross Bicycles
	5	Electric Bikes
	6	Mountain Bikes
	7	Road Bikes
*		

- ملاحظة : اذا ظهرت اي مشكلة او خطأ اثناء تتبع الخطوات صور شاشة الخطأ وارسله الى الكلاسروم

تطبيق عملي : انشاء مشروع للاتصال بقاعدة بيانات SQL server واجراء الاستعلامات عليها

الجزء الاول من التنفيذ العملي يكون في برنامج SQL Server Management Studio

1- انشاء قاعدة بيانات جديدة باسم MYbooks

```
create database MYbooks
use MYbooks
```

2- انشاء جدول داخل قاعدة البيانات MYbooks واسم الجدول هو books ويتكون من ثلاث حقول (رمز الكتاب bid / اسم الكتاب bname / سنة نشر الكتابbyear)

```
create table books (bid integer, bname varchar(10),byear varchar(10))
```

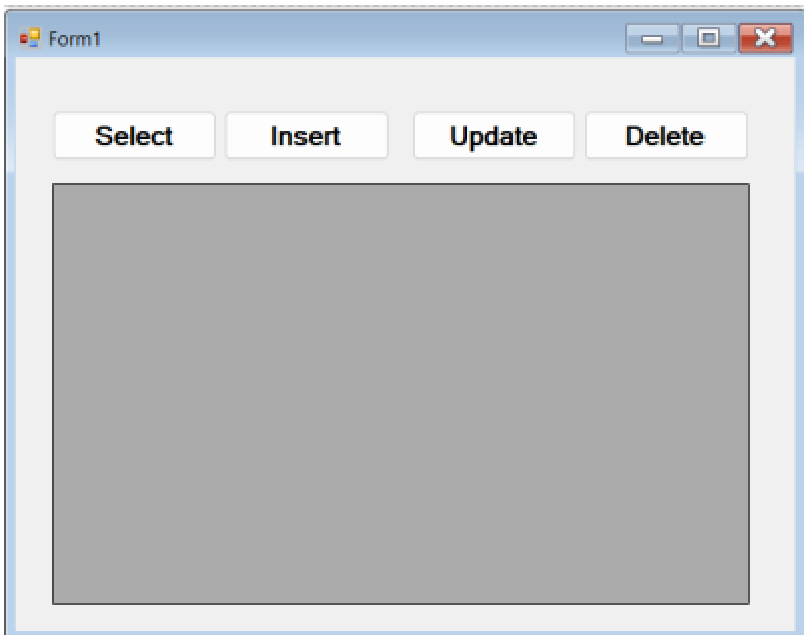
3- اضافة بيانات الى الجدول books (خمس سجلات او اكثر)

```
insert into books1 values(1,'vb.net2015','x2015'),
                         (2,'C++','x2000'),
                         (3,'java','x2006'),
                         (4,'python','x2010'),
                         (5,'matlab20b','x2020')
```

الجزء الثاني انشاء مشروع فجوال

4- اضافة Data Sources وانشاء اتصال مع SQL server وحسب الخطوات الموضحة في المحاضرة السابقة

5- انشاء Form واضافة اربعة من Buttons واداة DataGridView



6- كتابة الكود البرمجي التالي

```
Imports System.Data.SqlClient
Imports Microsoft.Data
```

استدعاء المكتبات المطلوبة للتعامل مع قواعد البيانات

```
Public Class Form1
    Dim con As New SqlConnection
    Dim sqlcom As New SqlCommand
    Dim sqlad As SqlDataAdapter
    Dim ds As DataSet
```

تعريف المتغيرات المطلوبة لاستخدام **SqlClient Provider**

لتفيذ استعلام (Select)في قاعدة البيانات Sql server للجدول الذي تم الاتصال به تستخدم الكود التالي

```
Private Sub Button1_Click(sender As Object, e As EventArgs)
Handles Button1.Click

con = New SqlConnection("Data Source=desktop3;Initial
Catalog=MYbooks;Integrated Security=True;Encrypt=False")
```

تهيئة قناة الاتصال او جملة الاتصال استخدم معلومات الاتصال الخاصة بك (اسم SQL server واسم قاعدة البيانات كما موجود في حاسوبك الشخصي)

```
If con.State = ConnectionState.Open Then con.Close()
con.Open()
```

للتاكد من ان قناة الاتصال غير مفتوحة او مستخدمة من قبل

```
Dim x As String
x = InputBox("enter book id")
sqlcom = New SqlCommand("select * from books where bid=" & x, con)
```

لتهيئة جملة الاستعلام Command Select وهي هنا استرجاع معلومات الكتاب حسب حقل bid

```
sqlad = New SqlDataAdapter(sqlcom)
```

لترجمة البيانات بين Sql server و برنامج VB.net

```
ds = New DataSet
sqlad.Fill(ds)
```

لخزن السجلات المسترجعة من قاعدة البيانات في برنامج فجوال تحديدا في اداة DataSet

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

لعرض السجلات للمستخدم على سطح النموذج باستخدام اداة DataGridView

```
con.Close()
```

لغلق قناة الاتصال وحفظ التغيرات في قاعدة البيانات

```
End Sub
```

لتنفيذ جملة اضافة Insert الى جدول قاعدة بيانات Sql server

```
Private Sub Button2_Click(sender As Object, e As EventArgs)
Handles Button2.Click

con = New SqlConnection("Data Source=desktop3;Initial
Catalog=MYbooks;Integrated Security=True;Encrypt=False")
```

تهيئة قناة الاتصال

```
If con.State = ConnectionState.Open Then con.Close()
        con.Open()
sqlcom = New SqlCommand("insert into books (bid,bname,byear)
values(6,'SQL99','x1997')", con)
sqlcom.ExecuteNonQuery()
```

تهيئة جملة الاستعلام (اضافة سجل من البيانات الى الجدول وتحديد قناة الاتصال ومن ثم تنفيذ عملية الاضافة)

بقية الكود تم شرحه سابقا ويستخدم للاستعلام وعرض نتيجة الجدول بعد اضافة السجل الجديد وغلق قناة الاتصال

```
sqlcom = New SqlCommand("select * from books", con)
sqlad = New SqlDataAdapter(sqlcom)
ds = New DataSet
sqlad.Fill(ds)
DataGridView1.DataSource = ds.Tables("books")
con.Close()
End Sub
```

لتنفيذ جملة تعديل سجل Update في جدول قاعدة بيانات Sql server

```
Private Sub Button3_Click(sender As Object, e As EventArgs)
Handles Button3.Click
con = New SqlConnection("Data Source=desktop3;Initial
Catalog=MYbooks;Integrated Security=True;Encrypt=False")
```

تهيئة قناة الاتصال

```
If con.State = ConnectionState.Open Then con.Close()
con.Open()

sqlcom = New SqlCommand("update books set bname='VB.Net 2022'
where bid=1", con)
sqlcom.ExecuteNonQuery()
```

تهيئة جملة الاستعلام تعديل اسم الكتاب الى VB.Net 2022 للسجل الذي يكون فيه حقل bid=1 ، ثم تنفيذ الاستعلام

بقية الكود تم شرحه سابقا ويستخدم للاستعلام وعرض نتيجة الجدول بعد تعديل اسم الكتاب في السجل 1 وغلق قناة الاتصال

```
sqlcom = New SqlCommand("select * from books", con)
sqlad = New SqlDataAdapter(sqlcom)
ds = New DataSet
sqlad.Fill(ds)
DataGridView1.DataSource = ds.Tables("books")
con.Close()
End Sub
```

لتنفيذ جملة حذف سجل Delete في جدول قاعدة بيانات Sql server

```
Private Sub Button4_Click(sender As Object, e As EventArgs)
Handles Button4.Click

con = New SqlConnection("Data Source=desktop3;Initial
Catalog=MYbooks;Integrated Security=True;Encrypt=False")
If con.State = ConnectionState.Open Then con.Close()
con.Open()
```

تهيئة قناة الاتصال والتاكد من فتح الاتصال

```
sqlcom = New SqlCommand("delete from books where bid=5", con)
sqlcom.ExecuteNonQuery()
```

تهيئة جملة الاستعلام وهي هنا حذف السجل الذي يكون فيه حقل bid=5 ، ثم تنفيذ الاستعلام

بقية الكود تم شرحه سابقا ويستخدم للاستعلام وعرض نتيجة الجدول بعد حذف السجل 5 وغلق قناة الاتصال

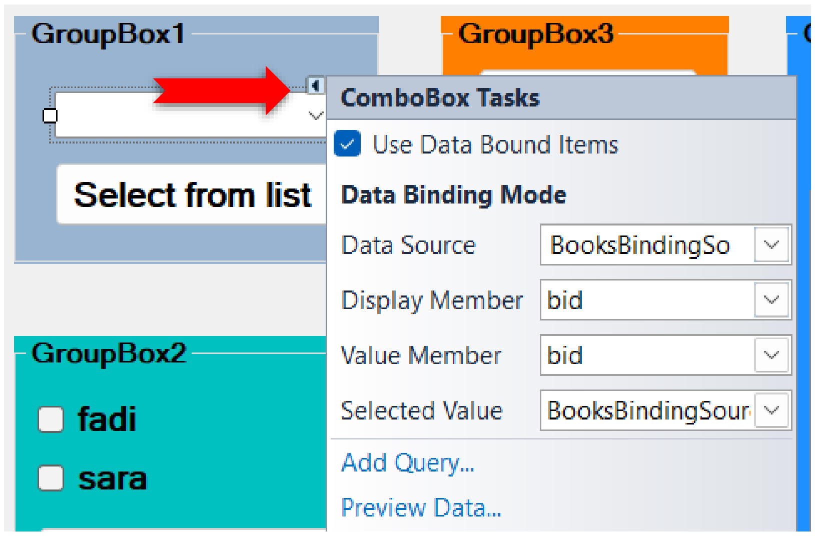
```
sqlcom = New SqlCommand("select * from books", con)
sqlad = New SqlDataAdapter(sqlcom)
ds = New DataSet
sqlad.Fill(ds)
DataGridView1.DataSource = ds.Tables("books")
con.Close()
End Sub
```

VB.Net and SQL server Example (cont.)

```
Imports System.Data.SqlClient
Imports System.Numerics
Imports System.Security.Cryptography.Xml
Imports Microsoft.Data

Public Class Form1
    Dim con As New SqlConnection
    Dim sqlcom As New SqlCommand
    Dim sqlad As SqlDataAdapter
    Dim ds As DataSet
    Dim inc As Integer
```

Group1 : Use ComboBox1 to control the sql command with field(bid) and view rows in the DataGridView1 as bid selected from ComboBox1

at design time load data from table (bid column) to combobox1,



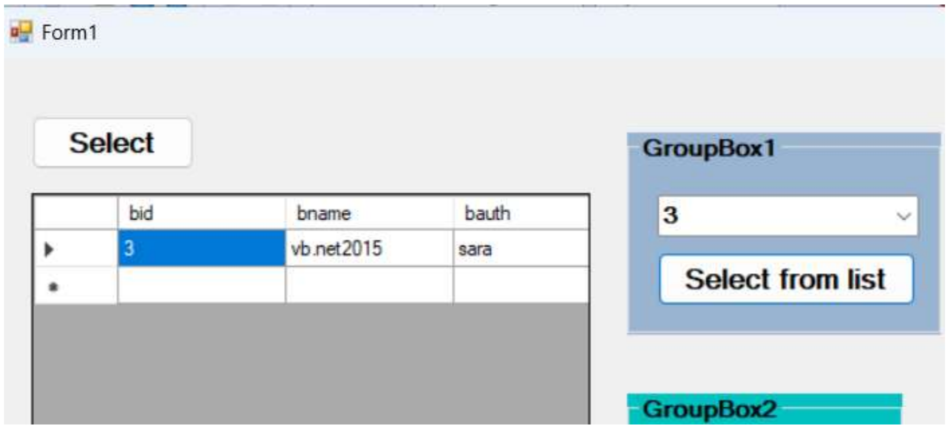
ComboBox Tasks
- ☑ Use Data Bound Items

Data Binding Mode

Data Source	BooksBindingSo
Display Member	bid
Value Member	bid
Selected Value	BooksBindingSour

Add Query...
Preview Data...

then use button (Select from list) to create SqlCommand

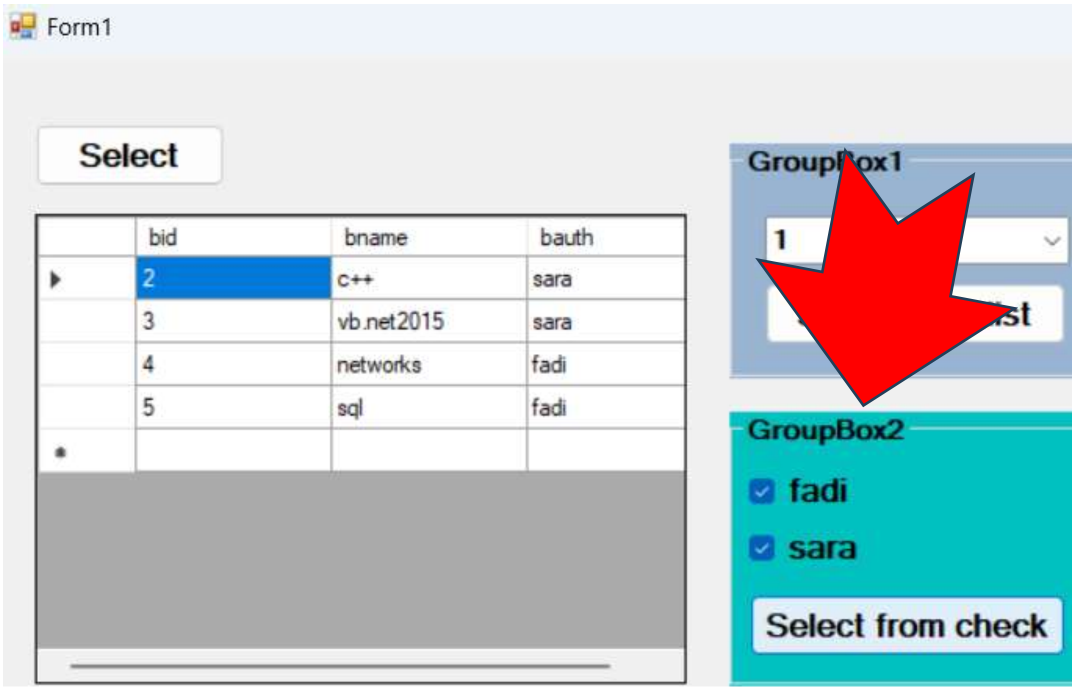


	bid	bname	bauth
▶	3	vb.net2015	sara
*			

VB.Net and SQL server Example (cont.)

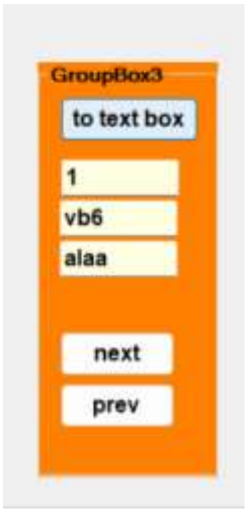
```
Private Sub Button5_Click(sender As Object, e As EventArgs)
Handles Button5.Click
    con = New SqlConnection("Data Source=desktop3;Initial
Catalog=MYbooks;Integrated Security=True;Encrypt=False")
    If con.State = ConnectionState.Open Then con.Close()
    con.Open()
    Dim x1 As String
    x1 = ComboBox1.Text
    sqlcom = New SqlCommand("select * from books where bid=" & x1,
con)
    sqlad = New SqlDataAdapter(sqlcom)
    ds = New DataSet
    sqlad.Fill(ds)
    DataGridView1.DataSource = ds.Tables(0)
    con.Close()
End Sub
```

Group2 : Use CheckBox1 and CheckBox2 to control the sql command with field(bauth) and view rows in the DataGridView1 as selected from Checked using button(Select from Check)



bid	bname	bauth
2	c++	sara
3	vb.net2015	sara
4	networks	fadi
5	sql	fadi

```
Private Sub Button6_Click(sender As Object, e As EventArgs)
Handles Button6.Click
    Dim x1, x2, x_sql As String
    x1 = ""
    x2 = ""

    con = New SqlConnection("Data Source=desktop;Initial
Catalog=MYbooks;Integrated Security=True;Encrypt=False")
    If con.State = ConnectionState.Open Then con.Close()
    con.Open()

If CheckBox1.Checked = True And CheckBox2.Checked = True Then
        x1 = CheckBox1.Text
        x2 = CheckBox2.Text
     x_sql = "select * from books where bauth= '" & x1 & "' or
bauth= '" & x2 & "'"
    End If

    If CheckBox1.Checked = True And CheckBox2.Checked = False Then
        x1 = CheckBox1.Text
        x2 = CheckBox2.Text
        x_sql = "select * from books where bauth= '" & x1 & "'"
    End If

    If CheckBox1.Checked = False And CheckBox2.Checked = True Then
        x1 = CheckBox1.Text
        x2 = CheckBox2.Text
        x_sql = "select * from books where bauth= '" & x2 & "'"
    End If

    sqlcom = New SqlCommand(x_sql, con)

    sqlad = New SqlDataAdapter(sqlcom)
    ds = New DataSet
    sqlad.Fill(ds)
    DataGridView1.DataSource = ds.Tables(0)
    con.Close()
End Sub
```



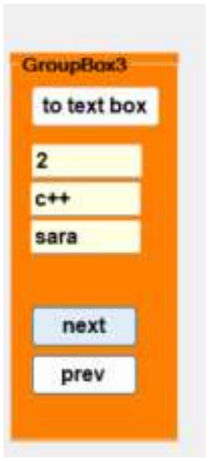
Group3: Use button (to text box) to load the first row from table to the three text boxes

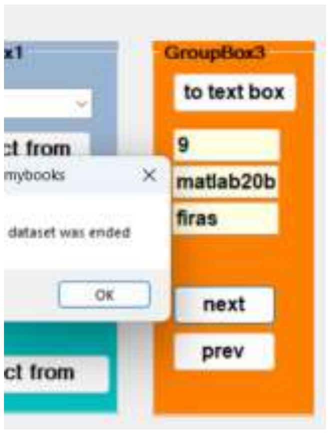
Use button (next) to go to the next row

Use button (prev) to go to the previous row

(to text button)

```
Private Sub Button7_Click(sender As Object, e As EventArgs)
Handles Button7.Click
    con = New SqlConnection("Data Source=deskto3;Initial
Catalog=MYbooks;Integrated Security=True;Encrypt=False")
    If con.State = ConnectionState.Open Then con.Close()
    con.Open()
    sqlcom = New SqlCommand("select * from books ", con)
    sqlad = New SqlDataAdapter(sqlcom)
    ds = New DataSet
    sqlad.Fill(ds)
    inc = 0
    MsgBox("no. of rows=" & ds.Tables(0).Rows.Count)
    TextBox1.Text = ds.Tables(0).Rows(inc).Item(0)
    TextBox2.Text = ds.Tables(0).Rows(inc).Item(1)
    TextBox3.Text = ds.Tables(0).Rows(inc).Item(2)
    con.Close()
End Sub
```

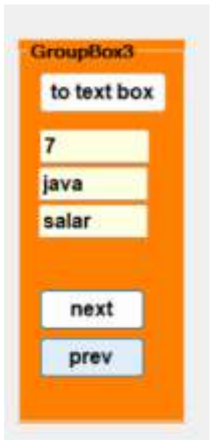


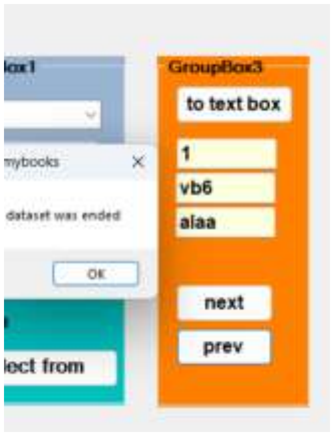


dataset was ended

(next button)

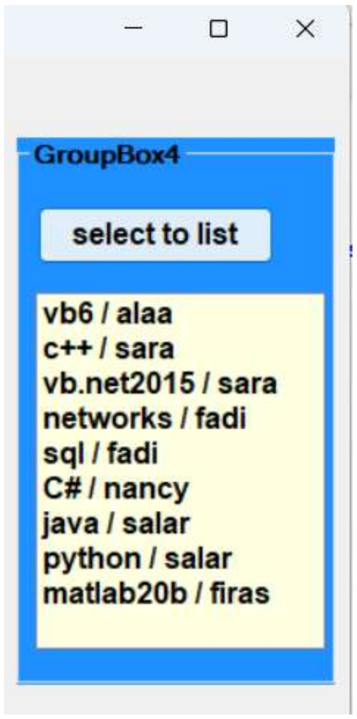
```
Private Sub Button8_Click(sender As Object, e As EventArgs)
Handles Button8.Click
    inc = inc + 1
    If inc <> ds.Tables(0).Rows.Count Then
        TextBox1.Text = ds.Tables(0).Rows(inc).Item(0)
        TextBox2.Text = ds.Tables(0).Rows(inc).Item(1)
        TextBox3.Text = ds.Tables(0).Rows(inc).Item(2)
    Else
        MsgBox("dataset was ended")
    End If
End Sub
```





(prev button)

```
Private Sub Button9_Click(sender As Object, e As EventArgs)
Handles Button9.Click
    inc = inc - 1
    If inc <> -1 Then
        TextBox1.Text = ds.Tables(0).Rows(inc).Item(0)
        TextBox2.Text = ds.Tables(0).Rows(inc).Item(1)
        TextBox3.Text = ds.Tables(0).Rows(inc).Item(2)
    Else
        MsgBox("dataset was ended")
    End If
End Sub
```



vb6 / alaa
c++ / sara
vb.net2015 / sara
networks / fadi
sql / fadi
C# / nancy
java / salar
python / salar
matlab20b / firas

Group4: Use button (select to list) to load the all rows (bname & bauth columns only) from table to the ListBox1 as the style below:

bname / bauth

(select to list button)

```
Private Sub Button10_Click(sender As Object, e As EventArgs)
Handles Button10.Click
     Dim i As Integer
     Dim list_item As String
     For i = 0 To ds.Tables(0).Rows.Count - 1
         list_item = ds.Tables(0).Rows(i).Item(1) & " / " &
ds.Tables(0).Rows(i).Item(2)
         ListBox1.Items.Add(list_item)
     Next
 End Sub
```