

Chương 3: WINDOWS FORM VÀ CÁC ĐIỀU KHIỂN

Mục tiêu bài học



1. Nắm được các khái niệm về đối tượng điều khiển
2. Sử dụng các điều khiển để thiết kế giao diện ứng dụng

Nội dung chương 2



1. Giới thiệu Windows Form
2. Các điều khiển dữ liệu
3. Lập trình ứng dụng cho bài toán mẫu

1. Giới thiệu Windows Form



1.1. Môi trường làm việc

1.2. Tạo Windows Form Project

1.3. Đối tượng điều khiển: thuộc tính, phương thức, sự kiện

Giới thiệu chung



- ☐ Windows Form là một phần của .NET Framework
- ☐ Cung cấp các lớp điều khiển (Control) hỗ trợ lập trình và xử lý dữ liệu trực quan
- ☐ Namespace System.Windows.Forms

Môi trường làm việc



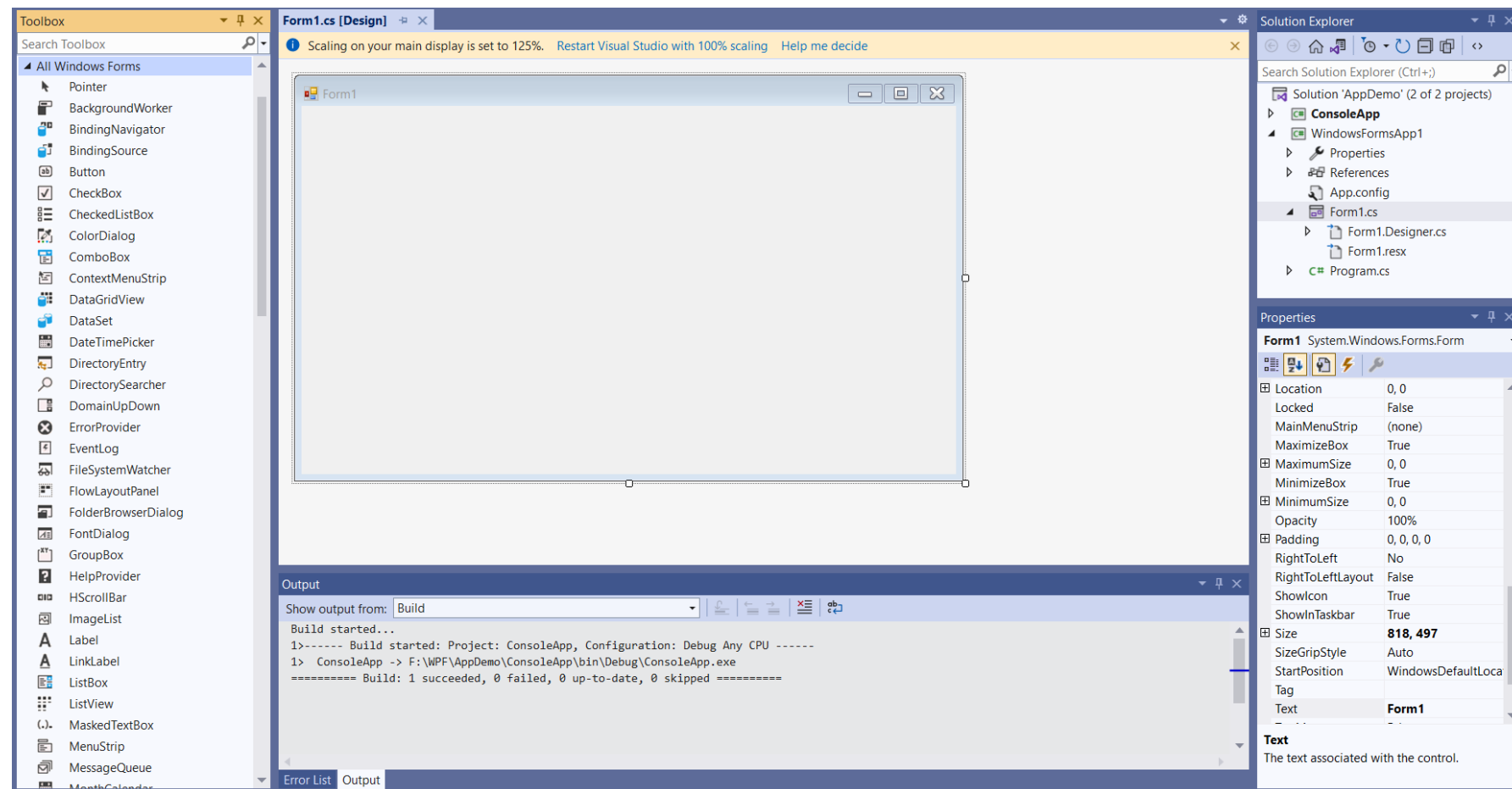
- ☐ VS.Net có thể sử dụng bất kì ngôn ngữ nào mà .Net hỗ trợ.
- ☐ Cung cấp môi trường thiết kế trực quan giúp người dùng có thể thiết kế giao diện dễ dàng
- ☐ Hỗ trợ các công cụ thao tác với database thông qua ADO.NET
- ☐ Windows Form là ứng dụng giao diện người dùng đồ họa dưới dạng các cửa sổ trong Windows.
- ☐ Trong các cửa sổ có các điều khiển (Controls) giúp người dùng tương tác với chương trình.

Môi trường làm việc



Các thành phần bên trong VS.NET:

- Solution Explorer
- Toolbox
- Server Explorer
- Class View
- Properties



Tạo Windows Form Project



- ❖ New Project
- ❖ Chọn ngôn ngữ C#
- ❖ Chọn Windows Form Project

Add a new project

Recent project templates

- Windows Forms App (.NET Framework) C#
- Console App (.NET Framework) C#
- Windows Forms App C#
- WPF App (.NET Framework) C#
- WPF Application C#

Search for templates (Alt+S) 🔍

[Clear all](#)

C#

Windows

Desktop



Windows Forms App

A project template for creating a .NET Windows Forms (WinForms) App.

C#

Windows

Desktop



Windows Forms App (.NET Framework)

A project for creating an application with a Windows Forms (WinForms) user interface

C#

Windows

Desktop



WPF Application

A project for creating a .NET WPF Application

C#

Windows

Desktop



WPF Class Library

A project for creating a class library that targets a .NET WPF Application

C#

Windows

Desktop

Library



WPF Custom Control Library

A project for creating a custom control library for .NET WPF Applications

C#

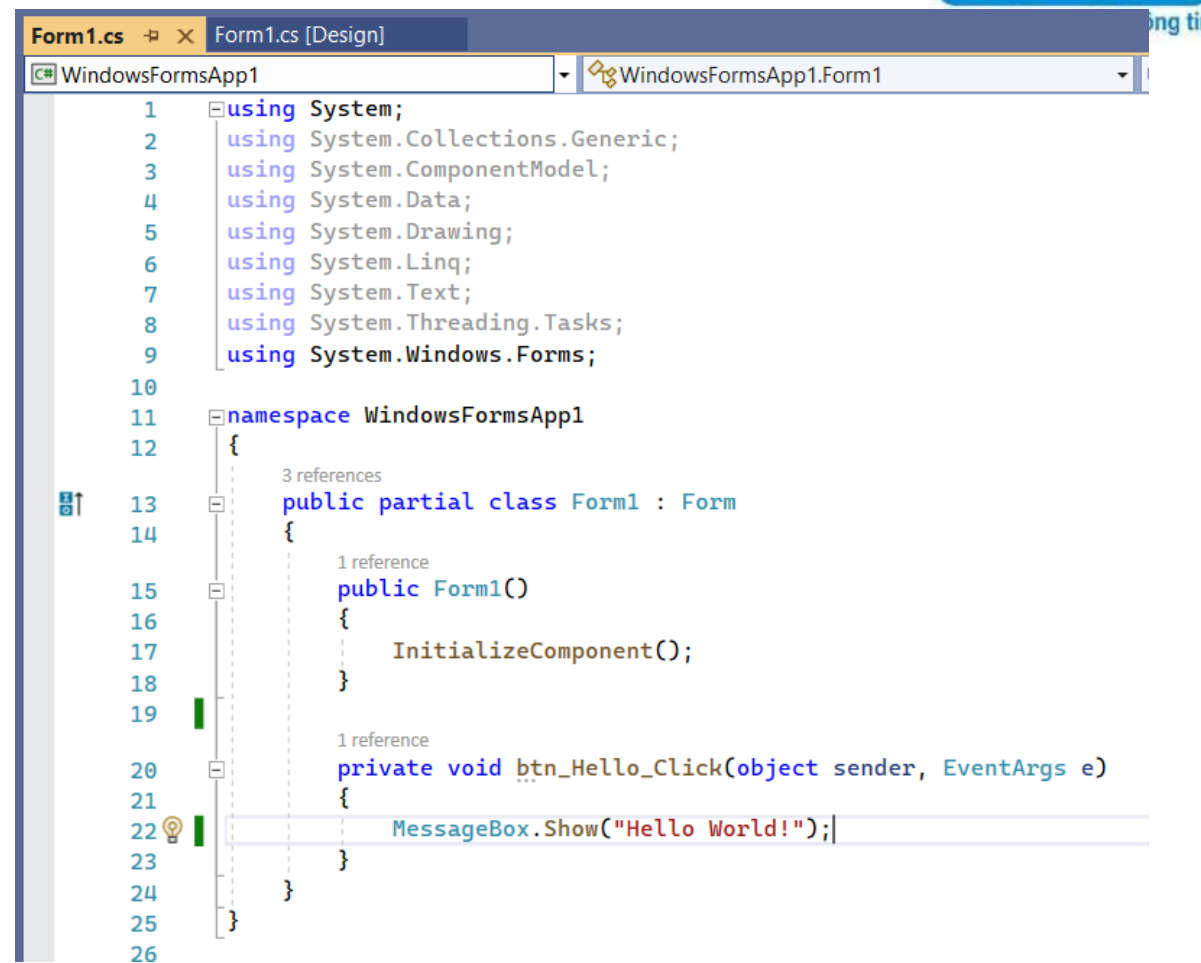
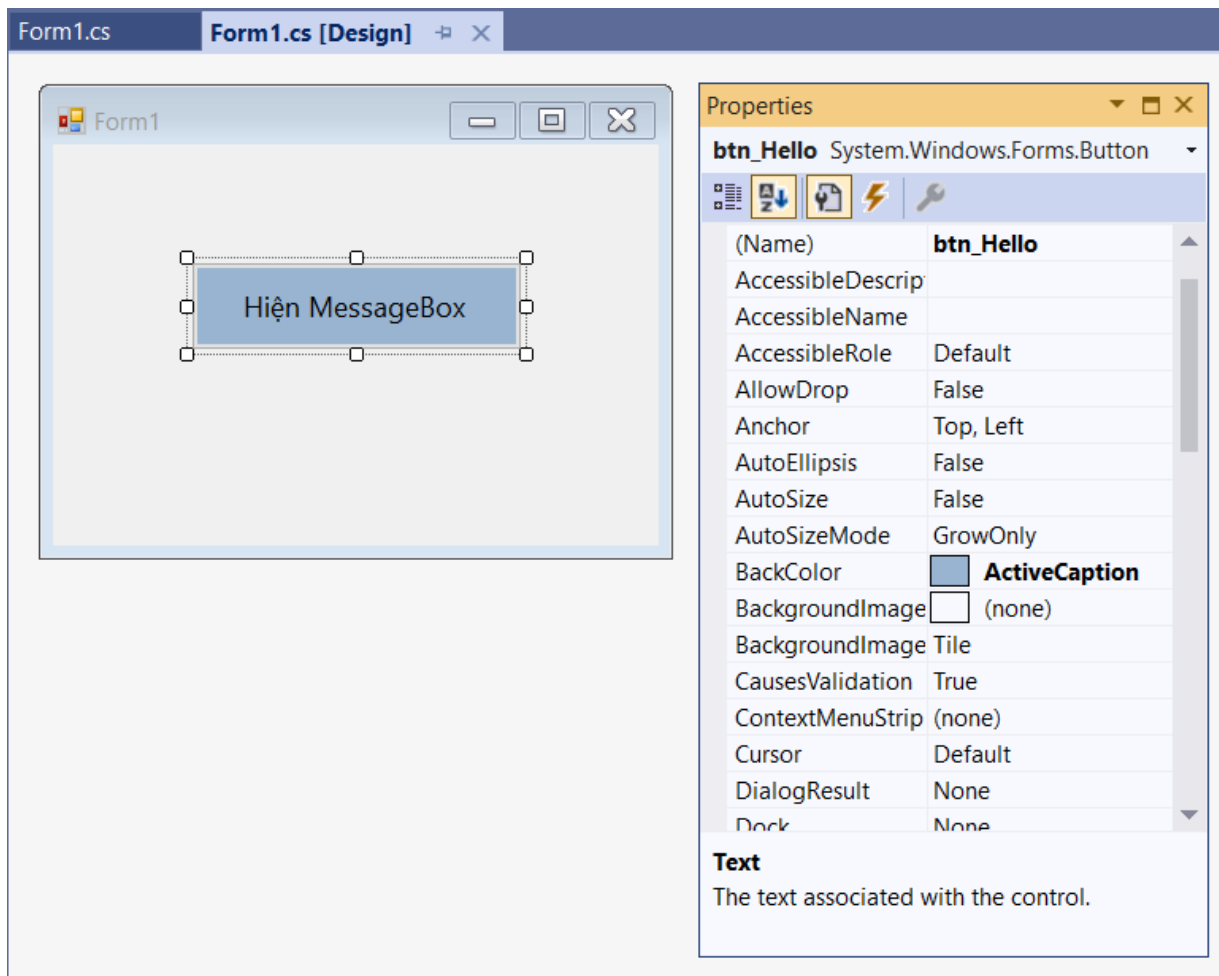
Windows

Desktop

Library

Next

Hello World WF



Chạy chương trình

- ❖ Ctrl + F5: chạy không Debugging
- ❖ Debug:
 - Breakpoint
 - F5
- ❖ Hàm `Main()` trong `Program.cs`

```
namespace WindowsFormsApp1
{
    0 references
    internal static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        0 references
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

- ❖ Đối tượng điều khiển dạng cửa sổ
- ❖ Trên Form, có thể chứa các đối tượng điều khiển khác tạo nên giao diện người dùng như nút lệnh (Button), hộp nhập văn bản (TextBox),...
- ❖ Form chia 2 loại:
 - SDI Form: Single Document Interface: NotePad, Paint
 - MDI Form: Multi Document Interface: Office, VS.NET,...
- ❖ Thêm Form vào ứng dụng
 - Add New Item
 - Form

Thuộc tính của Form



- **Name**: tên form phải khác nhau
- **AcceptButton**: Giá trị mà thuộc tính này nhận là tên của một Button trên Form, nhấn Enter – Click Button
- **AutoSizeMode**: cho phép/ không cho phép thay đổi kích thước Form (GrowOnly/GrowAndShrink)
- **BackColor**: màu nền của Form
- **BackgroundImage**: Chọn ảnh làm nền cho Form
- **CancelButton**: Giá trị mà thuộc tính này nhận là tên của một Button trên Form, nhấn Esc – Click Button
- **IsMdiContainer**: True/False – MDI Form
- **MdiParent**: Child Form – thiết lập runtime
- **StartPosition**: vị trí bắt đầu xuất hiện khi form được thực thi
- **WindowState**: trạng thái xuất hiện của form (minimized, maximized, hoặc normal)
-

Phương thức của Form



- **Active()**
- **Close()**
- **Show()**
- **ShowDialog()**
- ...

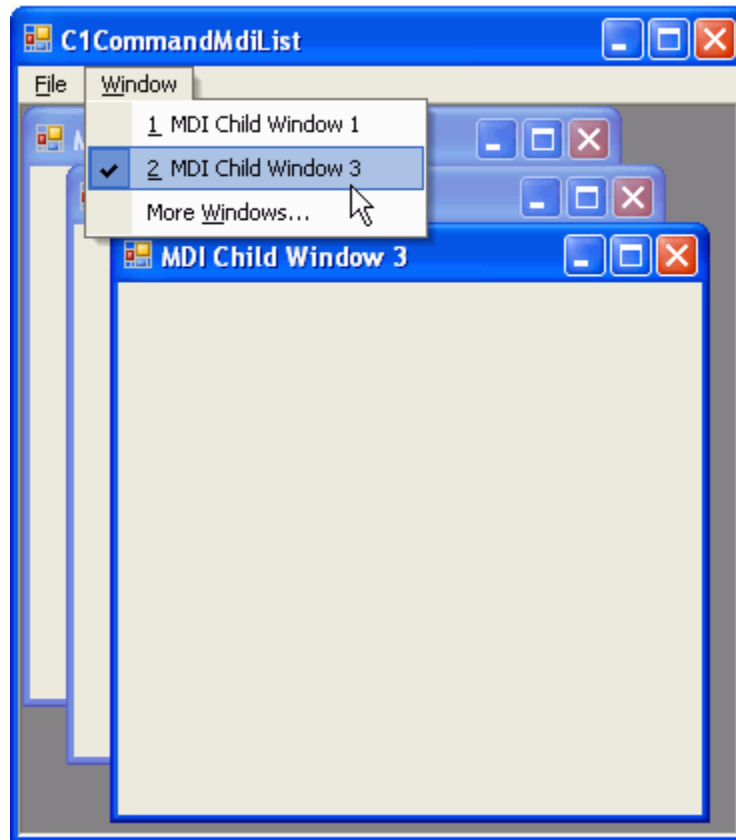
Sự kiện của Form



- **Move**
- **Load**
- **VisibleChanged**
- **Activated**
- **Shown**
- **Paint**
- **Deactivate**
- **FormClosing**
- **FormClosed**
- ...

MDI Form

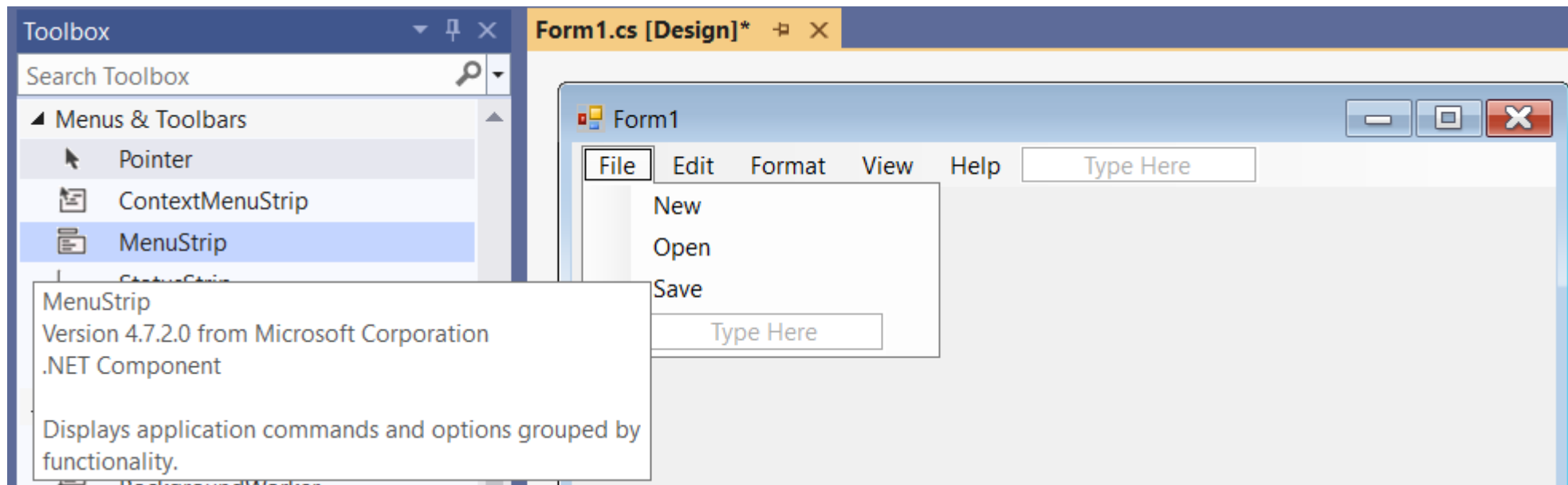
- MDI Form (Multiple Document Interface) là 1 dạng Form chứa nhiều Form con. Một ứng dụng đa cửa sổ MDI có thể hiển thị nhiều cửa sổ con bên trong chúng.



Menu/Toolbar

❖ MenuStrip – MenuItem

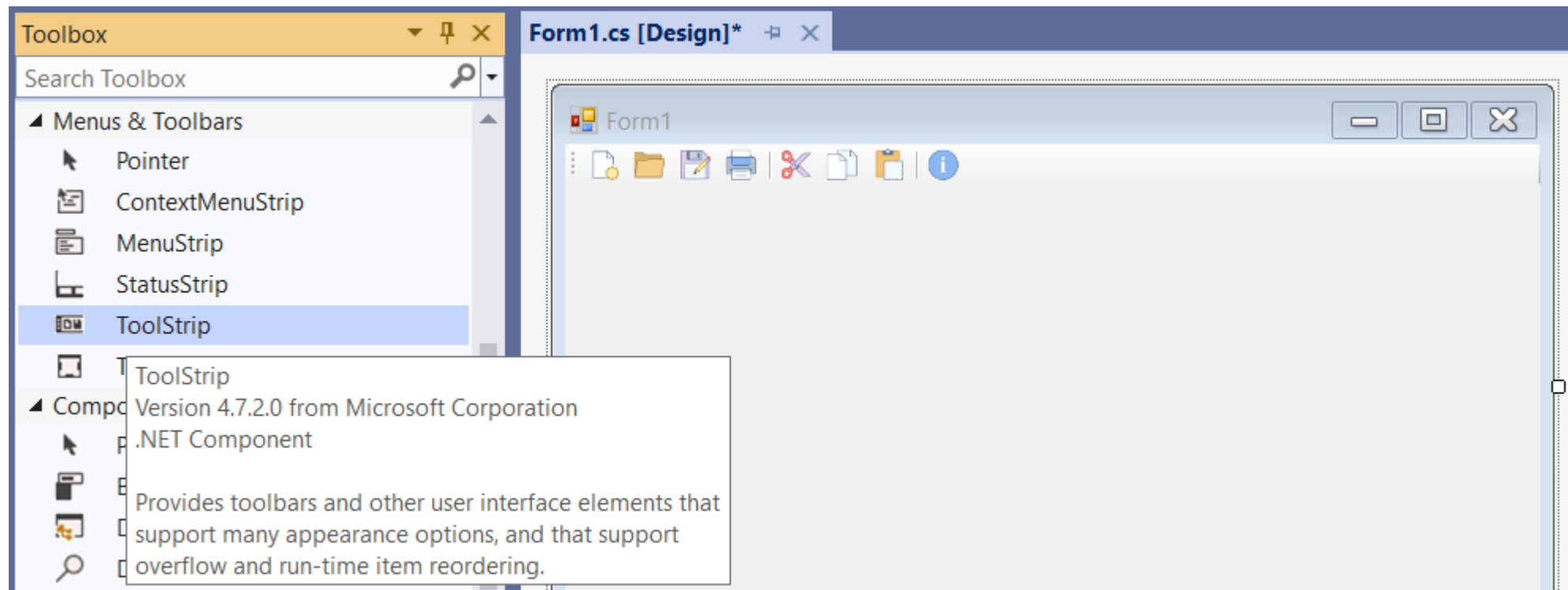
- MenuStrip: Cho phép thiết kế hệ thống menu lựa chọn chức năng cho ứng dụng
- Thao tác: chọn mục MenuItem trên menu – Mỗi một MenuStrip thì chạy 1 chức năng



Menu/Toolbar

❖ ToolStrip

- Tạo thanh công cụ cho các chức năng gắn trên Form.
- Thao tác: Đưa ToolStrip vào Form > Tạo các ToolStripItem > Thiết lập icon, Xử lý sự kiện Click



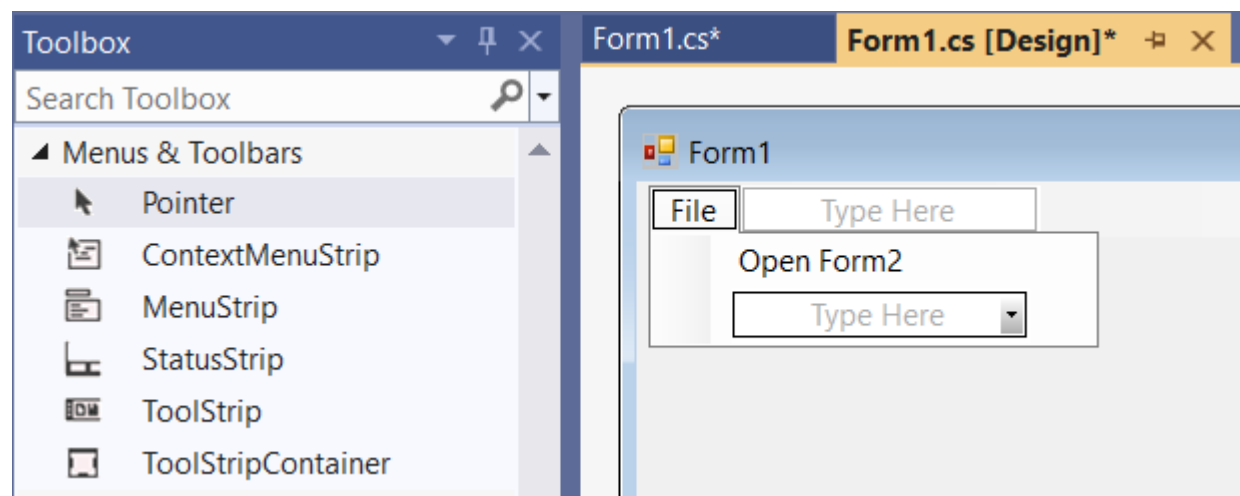
Tạo MDI Form & Menu/Toolbar

■ Cách tạo MDI Form:

- Bước 1: Tạo 1 project Windows Forms mới trong Visual Studio. Thiết lập thuộc tính IsMdiContainer là true cho Form chính được chỉ định là một MDI chứa các Form con

```
1 reference
private void Form1_Load(object sender, EventArgs e)
{
    this.IsMdiContainer = true;
}
```

- Bước 2: Từ Toolbox kéo điều khiển MenuShip vào Form1



Tạo MDI Form & Menu/Toolbar

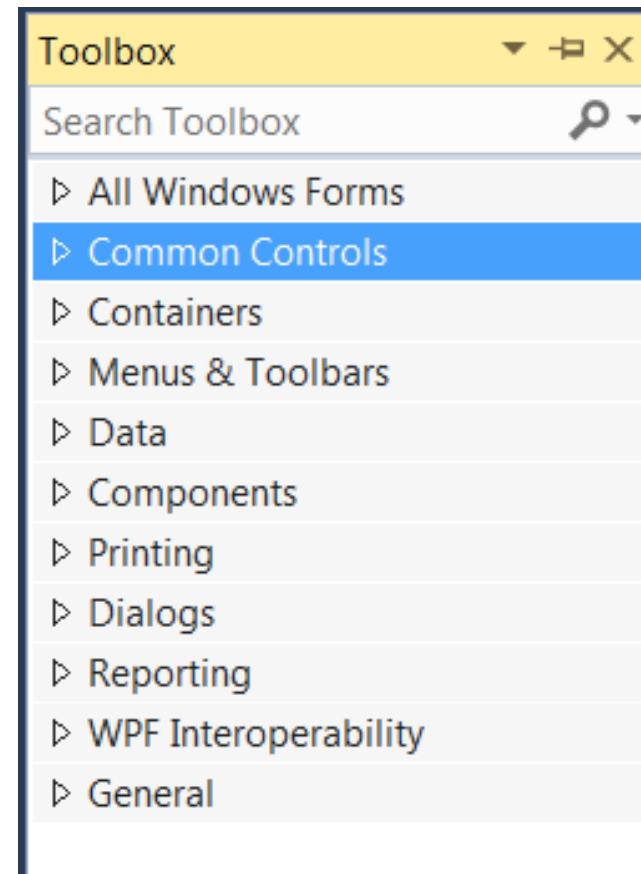
- **Cách tạo MDI Form:**

- Bước 3: Chuột phải vào Project, chọn Add -> New item để thêm mới 1 form đặt tên là Form2
- Bước 4: Tạo sự kiện Click cho menu “Open Form2”, và chèn đoạn mã để mở form con như sau:

```
1 reference
private void openForm2ToolStripMenuItem_Click(object sender, EventArgs e)
{
    Form2 form2 = new Form2();
    form2.MdiParent = this;
    form2.Show();
}
```

Đối tượng điều khiển trong VS.NET

- Visual Studio cung cấp nhiều đối tượng điều khiển giúp người lập trình dễ thiết kế giao diện chương trình
- Một số đối tượng điều khiển:
 - Các điều khiển vật chứa (Containers)
 - Các điều khiển dữ liệu (Data)
 - Điều khiển thành phần (Components)
 - Điều khiển hộp thoại (Dialogs)
 - Điều khiển báo cáo (Reporting)



Đối tượng điều khiển trong VS.NET



- Điều khiển nhập liệu:

- TextBox
- RichTextBox
- MaskedTextBox

- Điều khiển hiển thị:

- Label
- LinkLabel

- Điều khiển kết gán dữ liệu:

- Combobox
- ListBox
- ListView
- DataGridView

- Điều khiển mục chọn:

- CheckBox
- RadioButton
- TrackBar

Label

- ❖ Dùng để hiển thị các thông báo, lời nhắc trên Form.
- ❖ Một số thuộc tính:
 - AutoSize
 - BorderStyle
 - Font
 - Text
 - TextAlign



TextBox

❖ Cho phép nhập và hiển thị dữ liệu

❖ Thuộc tính:

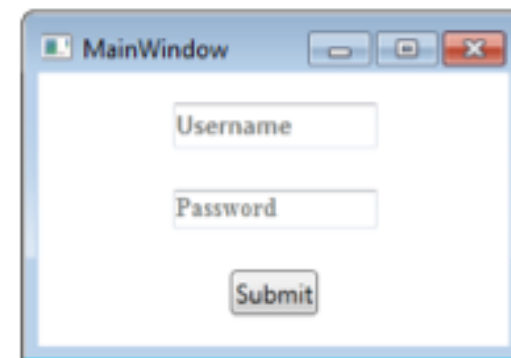
- MaxLength
- Multiline: True/False
- PasswordChar: (*,#,..)
- ScrollBar

❖ Sự kiện:

- TextChanged
- Validating
- KeyPress
- MouseClick

❖ Phương thức:

- AppendText: Nối một chuỗi (string) vào giá trị Text hiện có của Textbox
- Copy, Cut, Paste: Xử lý phần nội dung bôi đen (như Word).
- Focus: Chuyển Focus (con trỏ) vào TextBox
- Clear: Xóa toàn bộ nội dung trong Textbox
- Undo: Xử lý Undo như Word.



MaskedTextBox

❖ Dùng để quy định dạng thức nhập dữ liệu

❖ Thuộc tính:

- Mask: Thiết lập mặt nạ nhập liệu:

- 0, 9, #
- L, ?, &, C
- A, a
- ,
- .
- :
- /
- \$

❖ Ví dụ:

00/00/0000

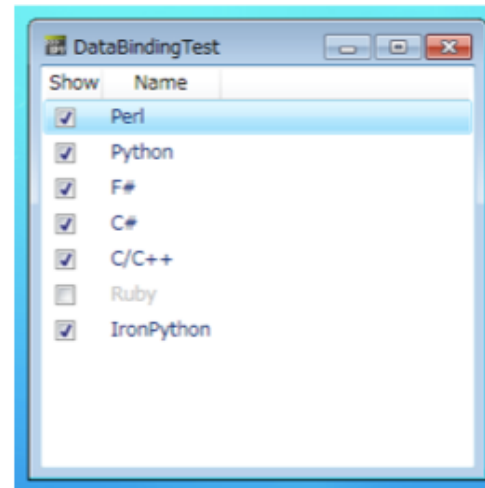
(999)-000-0000

Button

- ❖ Là các nút lệnh được dùng xác nhận thực thi một chức năng
- ❖ Thuộc tính:
 - Name
 - Size
 - Text
 - ...
- ❖ Phương thức
 - Click()
 - ...

CheckBox

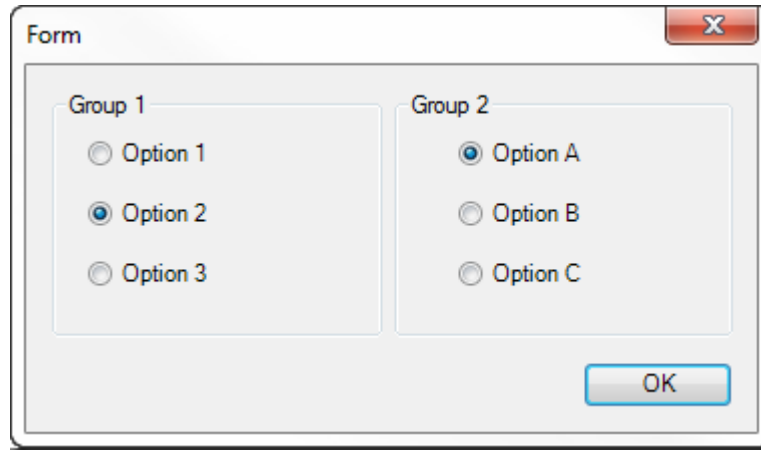
- ❖ Hiển thị danh sách rời rạc, cho phép chọn một hoặc nhiều giá trị (Yes/No hoặc True/False)
- ❖ Thuộc tính:
 - Text
 - CheckAlign
 - Checked: T/F
 - CheckState: Checked/Unchecked/Indeterminate
 - ...



RadioButton

- ❖ Cho phép thay đổi trạng thái biểu hiện – chỉ cho phép chọn 1 lựa chọn
- ❖ Nếu có nhiều lựa chọn – gom nhóm GroupBox
- ❖ Thuộc tính

- Checked: true/false
- Text
- ...



The screenshot shows a standard Windows application window titled "Form". Inside the window, there are two distinct groups of radio buttons, labeled "Group 1" and "Group 2". In "Group 1", the radio button for "Option 2" is selected. In "Group 2", the radio button for "Option A" is selected. At the bottom right of the form, there is an "OK" button. This visualizes how multiple radio buttons can be organized into separate groups to allow for multiple selections across different categories.

ListBox

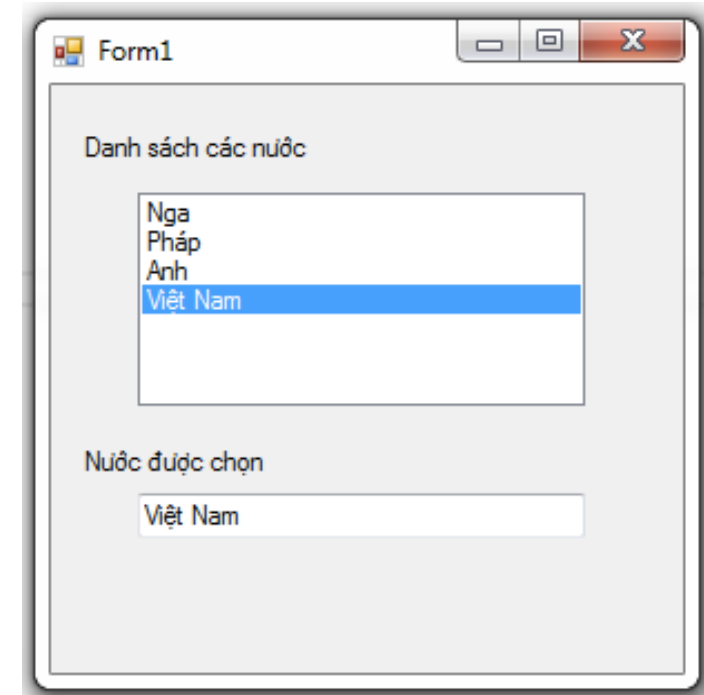
- ❖ Lưu các mục dữ liệu dạng danh sách – chỉ cho chọn
- ❖ Thuộc tính:
 - Items
 - Sorting: None/Ascending/Descending
 - MultiSelect: True/False
 - SelectedIndex, SelectedItem,...
- ❖ Phương thức:
 - Add()
- ❖ Sự kiện:
 - SelectedIndexChanged

ListBox

❖ Thêm Item vào ListBox có 2 cách:

- Design time: Chọn ListBox – cửa sổ Properties – chọn thuộc tính Items – chọn nút ... và nhập các Items của ListBox – OK
- Runtime: bằng phương thức Add()

```
private void Form1_Load(object sender, EventArgs e)
{
    lstNuoc.Items.Add("Nga");
    lstNuoc.Items.Add("Pháp");
    lstNuoc.Items.Add("Anh");
    lstNuoc.Items.Add("Việt Nam");
}
```



❖ Xử lý sự kiện khi chọn một mục thì hiện mục đã chọn trong textBoxNuoc

```
private void lstNuoc_SelectedIndexChanged(object sender, EventArgs e)
{
    txtNuoc.Text = lstNuoc.Items[lstNuoc.SelectedIndex].ToString();
}
```

ComboBox

❖ Lưu các mục dữ liệu dạng danh sách thả xuống, kết hợp của TextBox và ListBox

❖ Thuộc tính:

- Items
- Text
- Sorted: True/False
- SelectedIndex
- DropDownStyle: Simple/ DropDown/ DropDownList
- ...

❖ Phương thức:

- Add
- AddRange

❖ Sự kiện:

- SelectedIndexChanged

Bài tập: Tạo Form chức năng Thêm dữ liệu vào CSDL



ADMIN.TT88_ver2 - dbo.tbSINHVIEN			
	Column Name	Data Type	Allow Nulls
PK	iMaSV	int	<input type="checkbox"/>
	sHoTen	nvarchar(50)	<input checked="" type="checkbox"/>
	dNgaySinh	date	<input checked="" type="checkbox"/>
	sDiaChi	nvarchar(255)	<input checked="" type="checkbox"/>
	sSoDienThoai	varchar(30)	<input checked="" type="checkbox"/>
	bGioiTinh	bit	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Tạo 1 form như hình:

- 6 Label
- 4 TextBox
- 1 MaskedTextBox
- 2 Radio Button
- 5 Button

Yêu cầu:

- TextBox** mã và tên sinh viên không được để trống
- TextBox** Số điện thoại phải là 1 dãy số
- Đặt **MaskedTextBox** theo định dạng short date như hình
- Button** Thêm mới chỉ Enable khi các control đã hợp lệ theo quy định của thiết kế.

Điều khiển kết gán dữ liệu

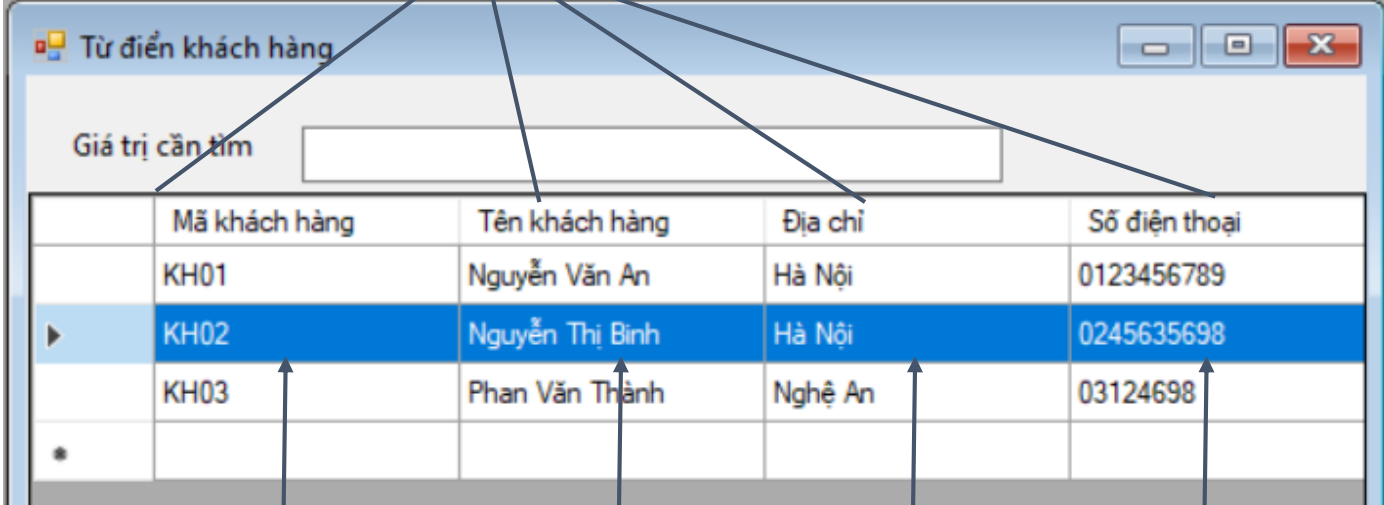


- ❖ Kết gán dữ liệu là quá trình đưa dữ liệu từ bộ nhớ hiển thị lên điều khiển .
- ❖ Các điều khiển kết gán có cung cấp các tính năng hỗ trợ việc tương tác dữ liệu: duyệt, cập nhật, sắp xếp,..
- ❖ Nhóm điều khiển cơ bản gồm: ComboBox, ListBox, CheckListBox, ListView, DataGridView....
- ❖ Thuộc tính kết gán:
 - **DataSource**: nguồn dữ liệu
 - **DisplayMember**: tên cột hiển thị
 - **ValueMember**: tên cột liên kết
 - **SelectedValue**: giá trị liên kết ứng với giá trị hiển thị

DataGridView

- Hiện danh sách dưới dạng lưới
- Các cột có thể thiết lập ở chế độ
 - Design time: Add/Edit Column
 - Runtime: lập trình can thiệp vào tập hợp Columns
- Có thể hiển thị dữ liệu nhiều cấp

Cột - column



Ô - cell

Cell[0]

Cell[1]

Cell[2]

Cell[3]

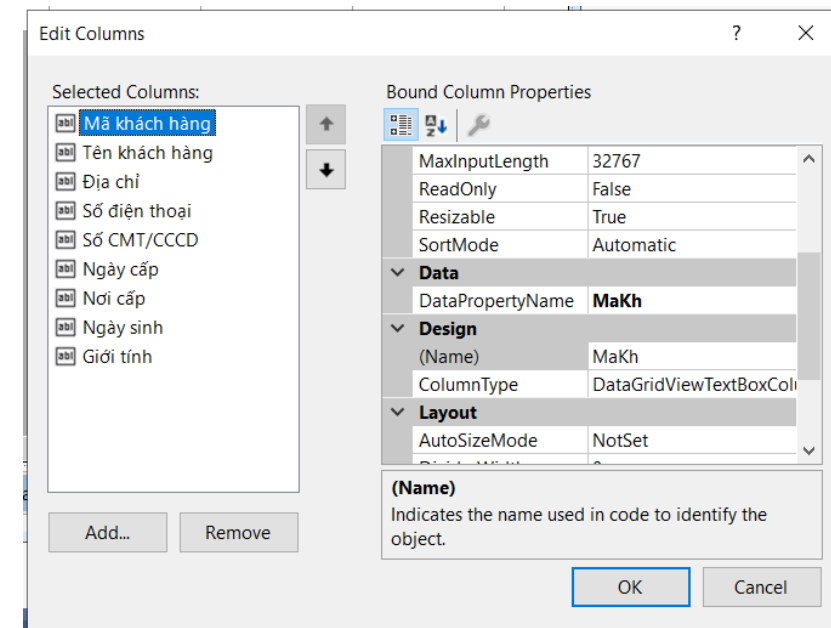
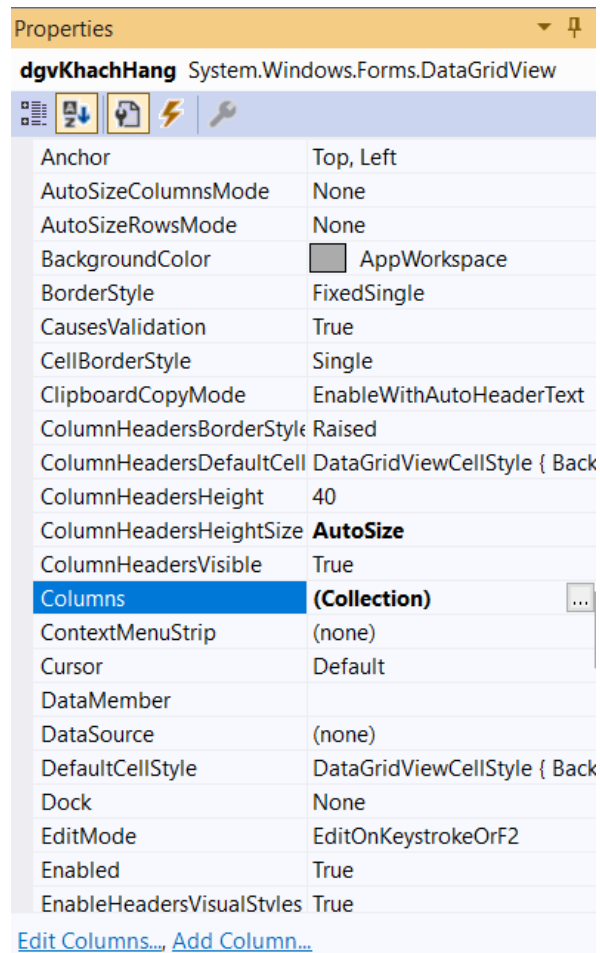
Hàng - row

	Mã khách hàng	Tên khách hàng	Địa chỉ	Số điện thoại
	KH01	Nguyễn Văn An	Hà Nội	0123456789
▶	KH02	Nguyễn Thị Bình	Hà Nội	0245635698
	KH03	Phan Văn Thành	Nghệ An	03124698
•				

DataGridView

- **Một số thuộc tính của DataGridView**

- AutoGenerateColumns
- Columns
- DataSource
- DataMember
- MultiSelect: True/False
- ReadOnly: True/False
- ...



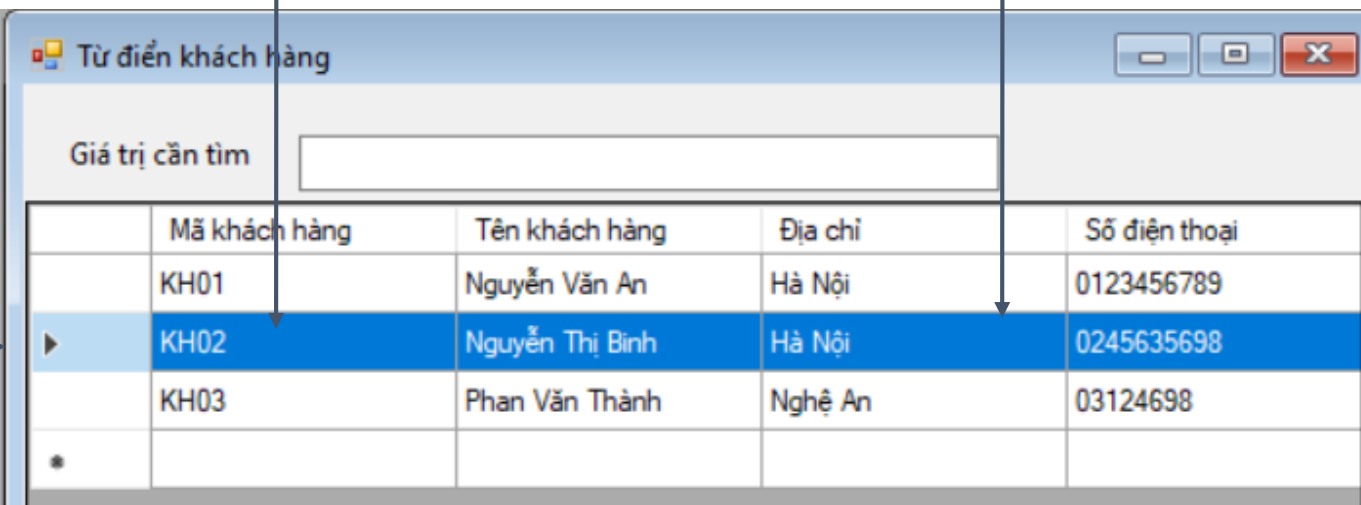
DataGridView

- **Thuộc tính thao tác dữ liệu hàng, cột, ô**
 - CurrentRow, CurrentCell
 - RowCount, ColumnCount
 - Row[i].Cells[j].Value: lấy dữ liệu đang được hiển thị trên ô [i][j]

CurrentCell

Rows[1].Cells[2].Value

CurrentRow



	Mã khách hàng	Tên khách hàng	Địa chỉ	Số điện thoại
	KH01	Nguyễn Văn An	Hà Nội	0123456789
►	KH02	Nguyễn Thị Bình	Hà Nội	0245635698
	KH03	Phan Văn Thành	Nghệ An	03124698
*				

DataGridView

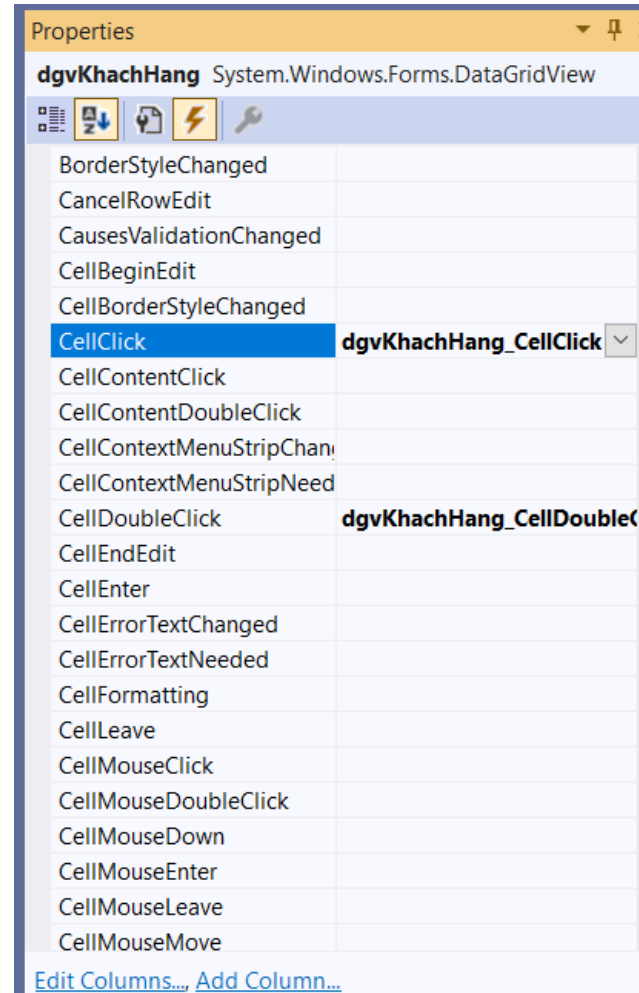
- Một số phương thức DataGridView

Tên phương thức	Mô tả
<u>AutoSizeColumns()</u>	Tự động căn chỉnh độ rộng cột theo tùy chọn
<u>AutoSizeRows()</u>	Tự động căn chỉnh độ rộng hàng theo tùy chọn
<u>Sort()</u>	Sắp xếp nội dung của DataGridView
<u>SelectAll()</u>	Chọn tất cả các ô trong DataGridView
<u>ClearSelection()</u>	Bỏ chọn tất cả các ô đã được người dùng chọn

DataGridView

- Sự kiện trong DataGridView

- SelectionChanged
- Click/DoubleClick
- CellClick
- RowEnter
- RowLeave
- RowAdded
-



Bài tập: Kết gán dữ liệu từ 1 bảng trong CSDL lên DataGridView



Thiết kế 1 form như hình:

Yêu cầu:

- Đặt tên cho các ColumnHeader
- Đặt thuộc tính DataPropertyName có trong DataGridView tương ứng với các trường dữ liệu có trong bảng tblSINHVIEN trong CSDL

The application window titled "Thông tin sinh viên" (Student Information) contains two main sections:

Thông tin sinh viên (Student Information): This section includes input fields for the following data:

- Mã sinh viên (Student ID)
- Họ tên sinh viên (Student Name)
- Số điện thoại (Phone Number)
- Địa chỉ (Address)
- Ngày sinh (Date of Birth) - with a date picker
- Giới tính (Gender) - with radio buttons for Nam (Male) and Nữ (Female)

On the right side of this section, there are five action buttons: "Thêm mới" (Add new), "Bỏ qua" (Skip), "Chỉnh sửa" (Edit), "Xóa bỏ" (Delete), and "Tìm kiếm" (Search).

Danh sách sinh viên (Student List): This section contains a DataGridView with the following columns:

	Mã sinh viên	Họ tên sinh viên	Ngày Sinh	Giới tính
*				

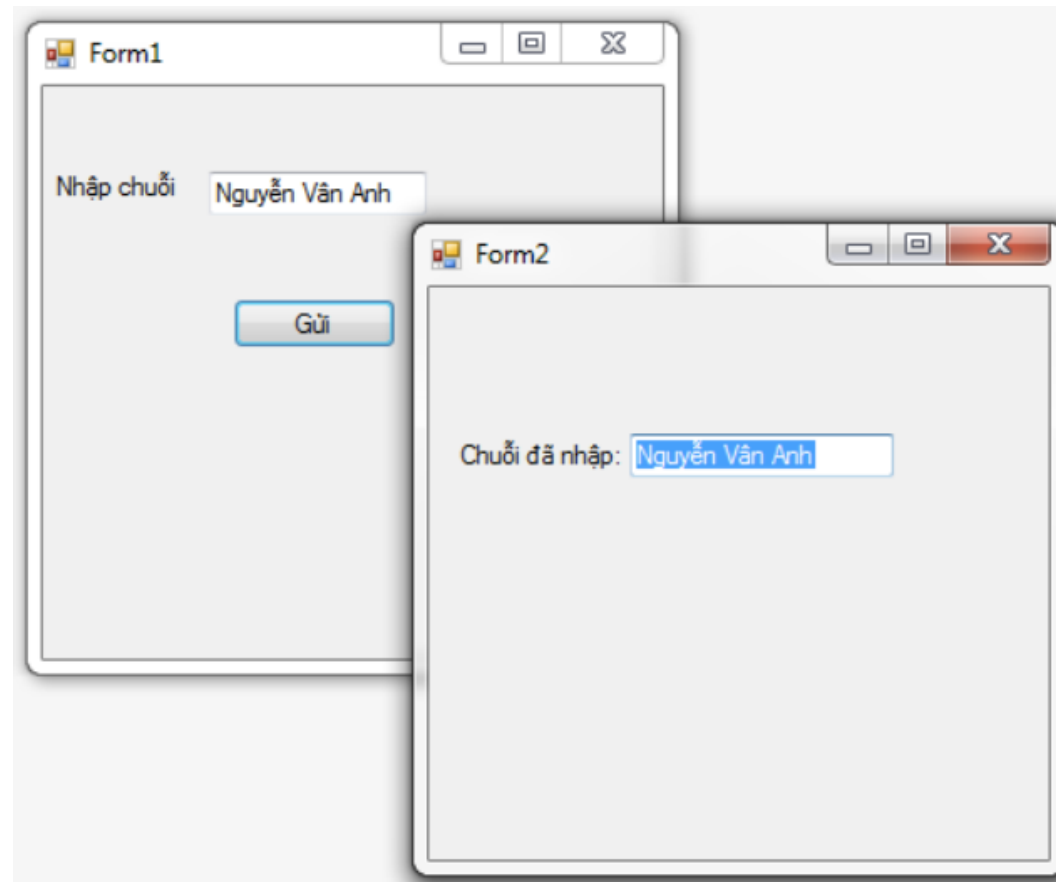
The body of the DataGridView is currently empty, showing only the header row and one placeholder row marked with an asterisk.

Trao đổi thông tin giữa các Form

Form A	Form B
Form A muốn gửi nội dung X cho Form B	
	1. Cần 1 phương thức Xuly(x) có phạm vi public
2. Truy nhập đến biến Form B trong Form A	
3. Gọi <biến Form B>.Xuly(X)	

Trao đổi thông tin giữa các Form

- ❖ Ví dụ: nhập một chuỗi ở Form1 khi nhấn nút Gửi thì nội dung chuỗi được nhận ở Form 2



Trao đổi thông tin giữa các Form

❖ Giải quyết: Dùng Constructor

Bước 1: Form2 có phương thức lấy chuỗi ở dạng public

```
public partial class Form2 : Form
{
    0 references
    public Form2()
    {
        InitializeComponent();
    }

    1 reference
    public Form2(string chuỗiGuiTuForm1)
    {
        InitializeComponent();
        tb_ChuoaiNhanDuoc.Text = chuỗiGuiTuForm1;
    }
}
```

Bước 2: Form1 gọi khởi tạo Form2 và truyền vào tham số cần chuyển đi

```
private void btn_Gui_Click(object sender, EventArgs e)
{
    Form2 form2 = new Form2(tb_ChuoaiCanGuiDi.Text);
    form2.Show();
}
```

Trao đổi thông tin giữa các Form

- ❖ Vi phạm OOP: Public các đối tượng nhận dữ liệu ở Form2 và truyền trực tiếp dữ liệu từ Form1 sang

Bước 1: Đặt thuộc tính Modify của textbox nhận chuỗi ở Form2 thành public

Design	
(Name)	tb_ChuoaiNhanDuoc
GenerateMember	True
Locked	False
Modifiers	Public

Bước 2: Form1 gọi khởi tạo Form2 và truyền trực tiếp chuỗi đã nhập vào textbox của Form2

```
private void btn_Gui_Click(object sender, EventArgs e)
{
    Form2 form2 = new Form2();
    form2.tb_ChuoaiNhanDuoc.Text = tb_ChuoaiCanGuiDi.Text;
    form2.Show();
}
```

**T
H
E N D**