BÀI 2: LẬP TRÌNH WINDOWS FORM VỚI CONTROLS CƠ BẢN

2.1 MỤC TIÊU

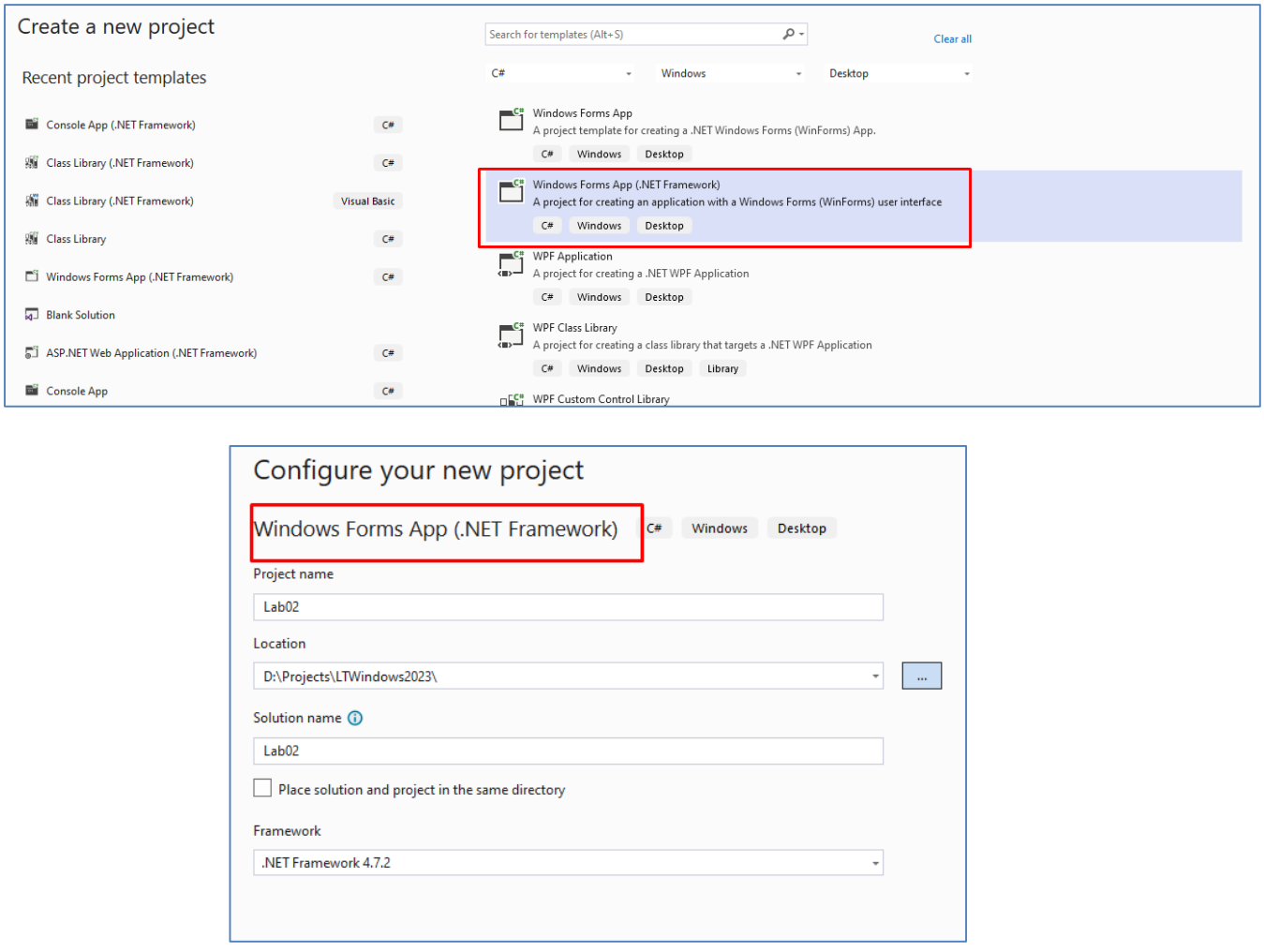
- Sử dụng Visual Studio .NET 2022, tạo ứng dụng dạng Windows Forms App (.NET Framework), ngôn ngữ C# .
- Làm quen với việc sử dụng các control thông dụng trên Forms như:
    - Label: Hiển thị các thông tin chỉ dẫn
    - TextBox: Hộp nhập liệu thông tin
    - Button: Cho phép user click chọn để thực hiện chức năng
    - CheckBox: Cho phép user chọn một hoặc nhiều option
    - Radio button: Cho phép user chọn duy nhất một option
    - MessageBox: Hiển thị thông tin đến user
    - DataGridView: Hiển thị danh sách thông tin trên bảng
    - ListView: Hiển thị một danh sách các item với các biểu tượng
    - ComboBox: Hộp chọn 1 giá trị trong danh sách giá trị
    - ListBox: Danh sách các mục chọn, cho phép chọn 1 hoặc nhiều mục
    - GroupBox: Nhóm các đối tượng về cùng nhóm
    - Panel: Nhóm các đối tượng vào cùng 1 khung
- Tìm hiểu các thuộc tính trên control (Visible, Enable, Name, Text …) và các phương thức là Event (Click, Text_Change, Text_Press...).
- Binding dữ liệu vào controls (Combobox, DatagridView, ListView)

2.2 HƯỚNG DẪN LÀM QUEN VỚI WINDOWS FORM

✓ Tạo Project Application, Giao diện màn hình thiết kế form

- Từ màn hình khởi động Microsoft Studio chọn Menu File - New – Project
- Language : Visual C#
- Loại ứng dụng: Windows Forms Application (.NET Framework)
- Name: Tên Project – ví dụ Lab02

Location: Đường dẫn lưu Project

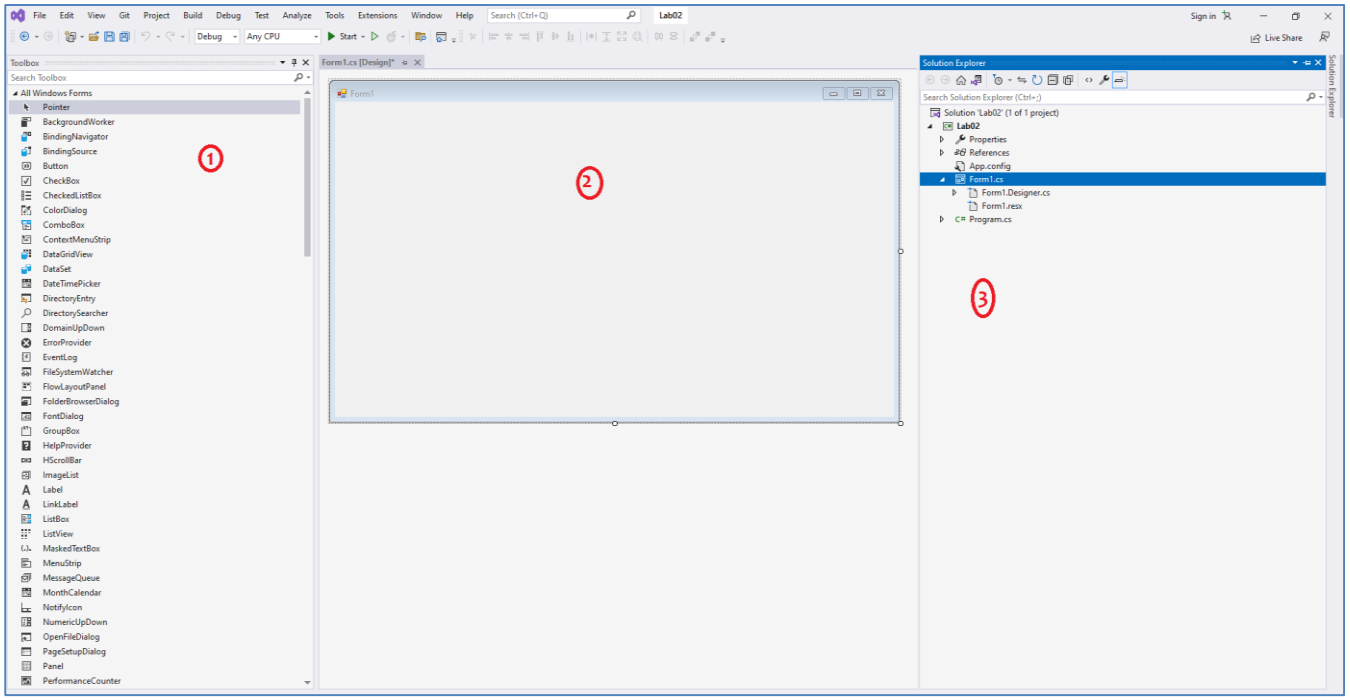


Hình 2.1: Tạo mới project Windows Form Application

Kết quả màn hình VS.NET cho ứng dụng Windows Form bao gồm các phần cơ bản

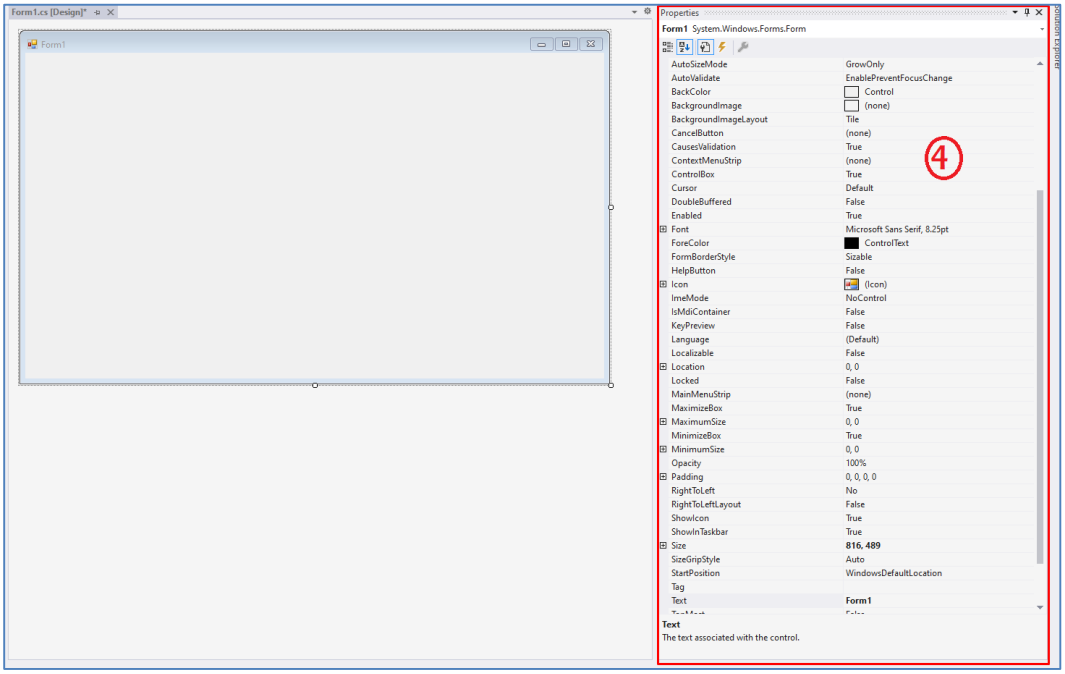
- (1): Toolbox: Chứa các control cho phép kéo thả vào Form

- (2): Chứa thiết kế giao diện Form, có thể chuyển sang View Code…
- (3): Cửa sổ Solution Explorer: Cho phép người lập trình có thể quản lý các thành phần trong project, hỗ trợ định vị nhanh chóng đến các file mã nguồn.



Hình 2: Màn hình VS. NET phục vụ cho việc tạo project Windows Form

- (4): Cửa sổ property: cho phép user có thể custom lại các thành phần control trên form như:



- Thiết lập các thuộc tính (Property): Trong cửa sổ "Properties" (Thuộc tính), sẽ thấy danh sách các thuộc tính của điều khiển được chọn. 1 số thuộc tính thường hay gặp như:

  + Text: Thuộc tính này cho phép thiết lập hoặc lấy văn bản hiển thị trên một Control. Ví dụ khi chọn Text của 1 Button là "Click me" tương đương ở code button1.**Text** = "Click me";

  + Name: Đặt tên cho một Control. Nó được sử dụng để tham chiếu đến Control

  + BackColor: Thiết lập màu nền cho một Control.

  + **ForeColor**: Thiết lập màu chữ cho một Control.

  + **Enabled**: Bật hoặc Tắt một Control. Khi Enabled là False, điều khiển sẽ bị vô hiệu hóa và không thể tương tác được.

  + **Visible**: Ẩn hoặc hiển thị một điều khiển. Khi Visible là False, điều khiển sẽ được ẩn đi và không được hiển thị trên giao diện.

  + **Size**: Thiết lập kích thước của một điều khiển.

  + **Location**: Thiết lập vị trí của một điều khiển trên giao diện

- Khai báo đăng ký xử lý sự kiện: Các điều khiển (controls) có thể kích hoạt các sự kiện (events) khi tương tác xảy ra, chẳng hạn như khi người dùng nhấp chuột, nhập liệu hoặc thay đổi trạng thái của điều khiển. 1 số event phổ biến

  + **Click**: Sự kiện Click xảy ra khi người dùng nhấp chuột vào một điều khiển, chẳng hạn như Button.

  Ví dụ: button1.Click += new EventHandler(button1_Click);

  + **TextChanged**: Khi thay đổi giá trị Text

  + **SelectedIndexChanged**: Khi mục được chọn thay đổi

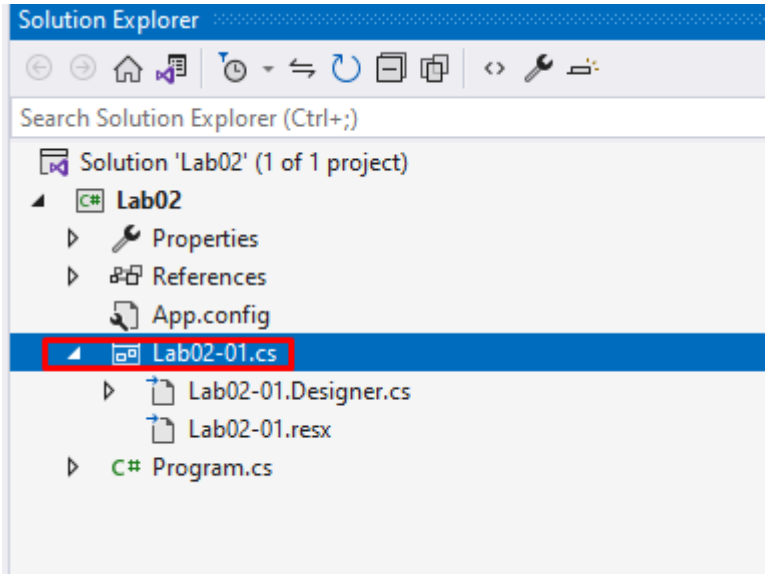
  + **CheckedChanged**: Khi trạng thái được kiểm tra (checked) của Control (như CheckBox hoặc RadioButton thay đổi).

  + MouseClick: Nhấp chuột lên một điều khiển.

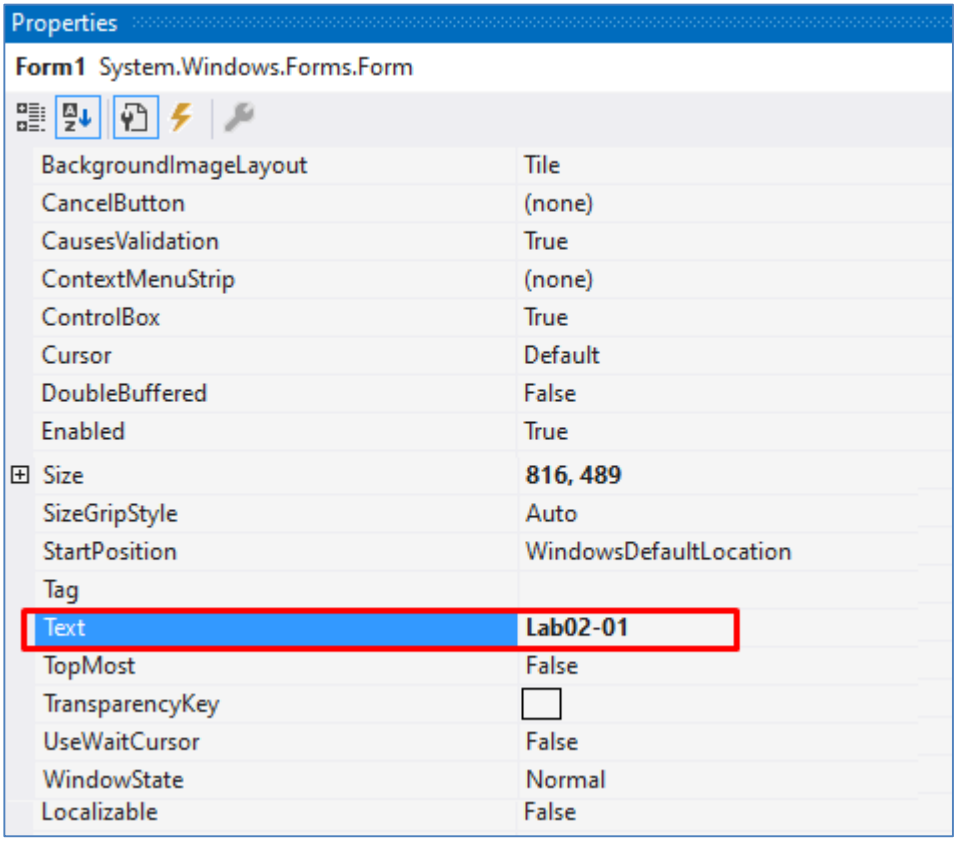
  + **KeyPress**: Nhấn một phím trong khi một Control (như TextBox) đang có trạng thái nhận nhập liệu.

  + **FormClosing**: Khi người dùng đóng một Form hoặc ứng dụng.

- Chú ý cửa sổ Toolbox chứa các công cụ để thiết kế: Nếu không thấy cửa sổ này, ta chọn menu View / **Toolbox**.
- Đổi tên form: Click lên Form1 ở cửa sổ Design, quan sát trên cửa sổ Properties, ta thấy có thuộc tính Text, giá trị mặc định là Form1, ta đổi thành Lab02-01

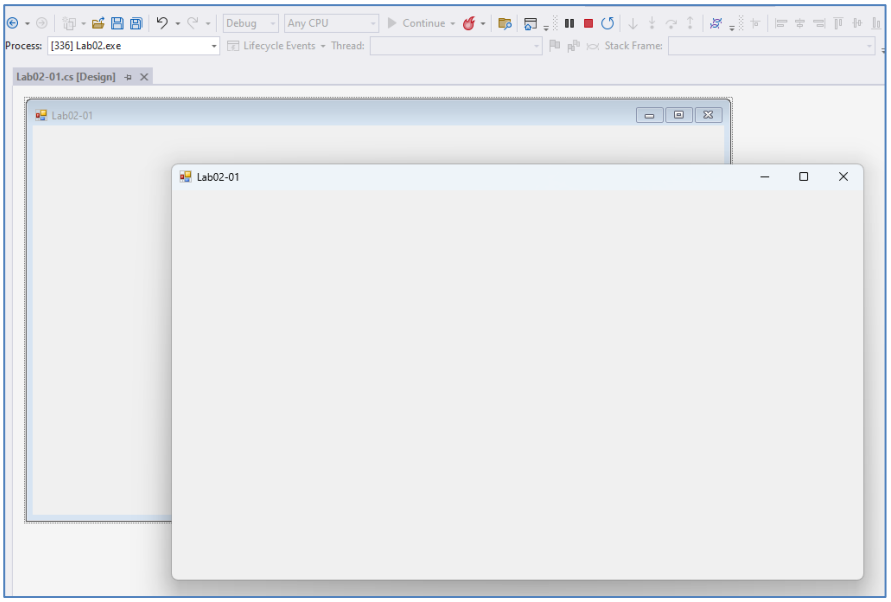


Thay đổi Property Text của Form



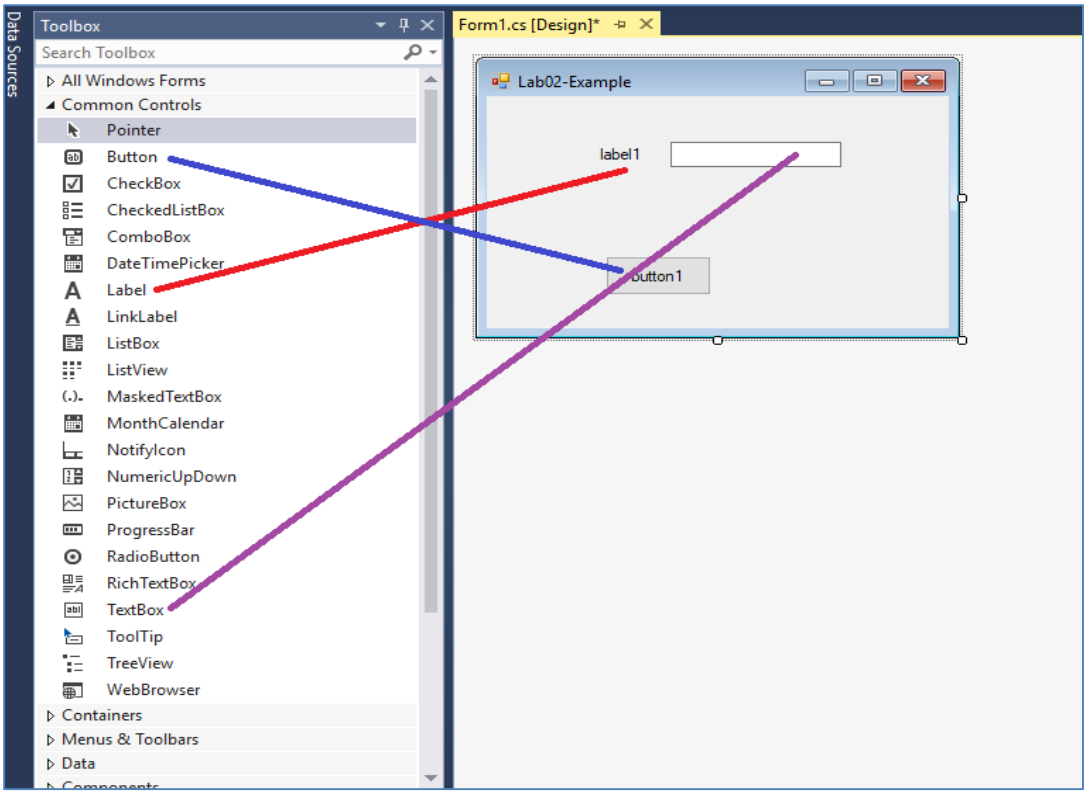
Hình 3: Property của đối tượng Form

- Chạy thử chương trình F5

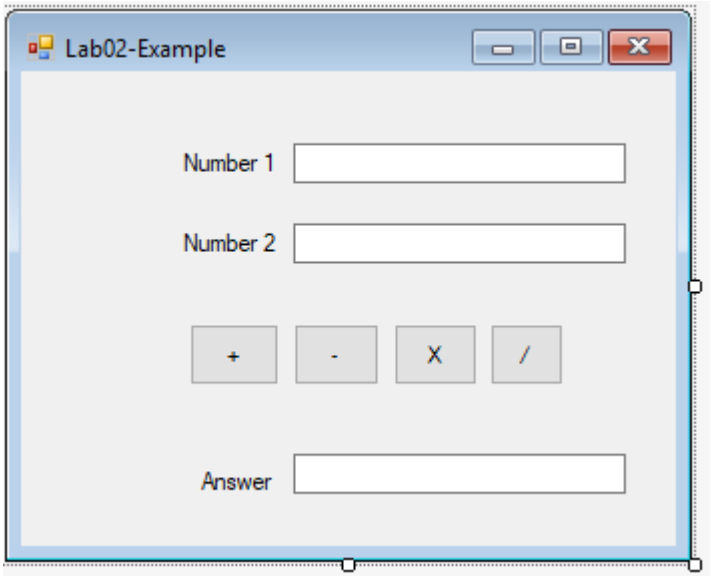


Hình 4: Chạy giao diện Forms đầu tiên

- Kéo thả các control từ **Toolbox** vào Forms: (chương trình tính toán + - * /)



- Thiết kế lại Form như hình dưới đây

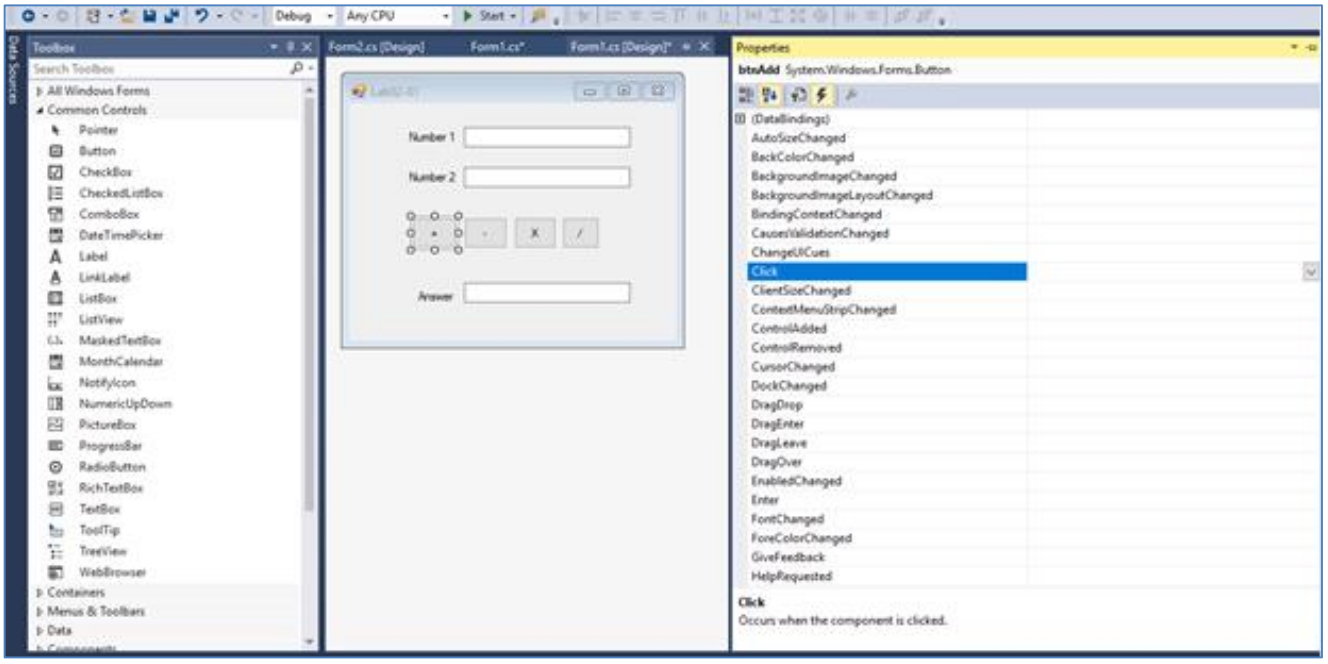


Hình 6: Giao diện sau khi thiết kế, chỉnh sửa Property

Đặt tên các **Controls** (thuộc tính **Name** trong **Property)** để dễ dàng quản lý**.** Đặt tên như sau:

- Các **Label** ở trong form có tên: lblNumber1, lblNumber2, lblAnswer
- Các **TextBox** ở trong form có tên: txtNumber1, txtNumber2, txtAnswer
- Các **Button** ở trong form có tên: btnAdd, btnSub, btnMul, btnDiv

- Xử lý sự kiện trên Control: Vào **Property** chọn biểu tượng **Events**



Hình 7: Các Events trong buttonAdd

✓ Chọn tên Event cần xử lý và double click vào đó.

Đối với các button để chọn nhanh sự kiện click ta có thể double click trực tiếp vào button đó.

Double click vào Button **btnAdd** để tiến hành viết code cho sự kiện Mouse click.

VS .NET tự động sinh ra phương thức ***btnAdd_Click.*** Sau đó tiến hành viết code tính tổng 2 số vừa nhập liệu

```
private void btnAdd_Click(object sender, EventArgs e)
{
        // Viết xử lí tính tổng trong sự kiện Add_click
        float number1 = float.Parse(txtNumber1.Text);
        float number2 = float.Parse(txtNumber2.Text);
        float result = number1 + number2;
        txtAnswer.Text = result.ToString();

 }
```

2.3 BÀI TẬP

Bài tập 1:

✓ Viết các sự kiện hoàn thành chương trình ở ví dụ trên (các phép toán +,-,*,/)

✓ Sử dụng **MessageBox** để đưa thông tin thêm cho người dùng khi gặp tất cả những lỗi ngoài mong muốn

✓ Quản lý lỗi bằng cách sử dụng: try… catch….finally

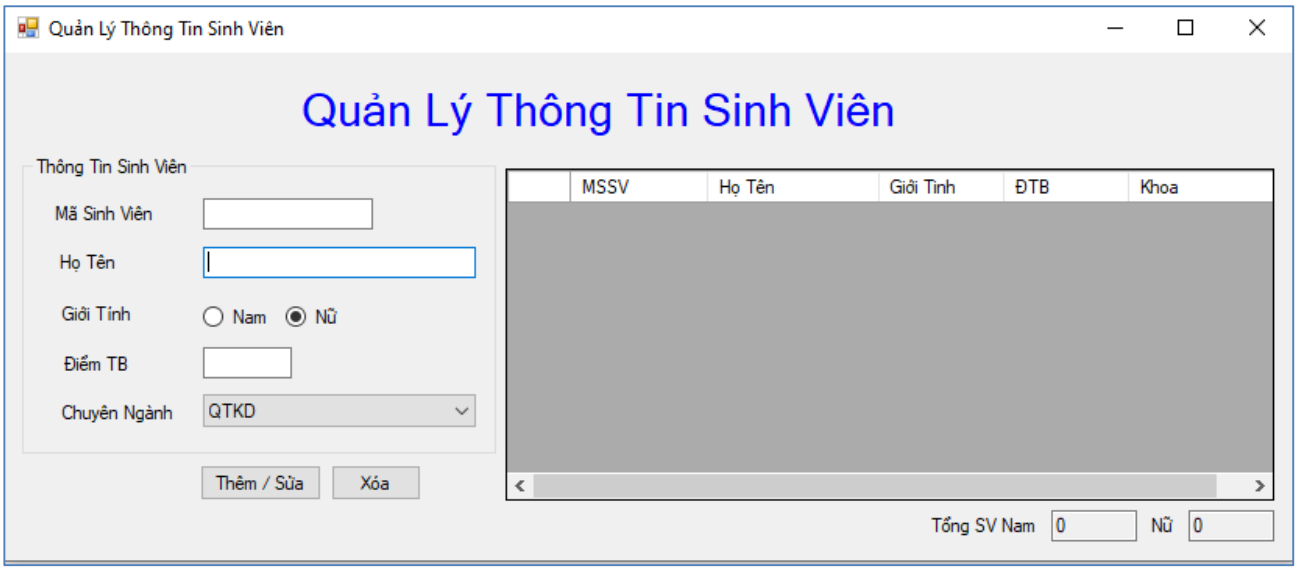
Hướng Dẫn: Sử dụng try…catch ở sự kiện, xử lý các sự kiện trong try

```
private void btnAdd_Click(object sender, EventArgs e)
{
   try
   {
      //Xử lí sự kiện cộng 2 số
   }
   catch(Exception ex)  //Khi gặp bất kì lỗi nào sẽ vào catch
   {
         MessageBox.Show(ex.Message);
   }
}
```

BÀI 2: LẬP TRÌNH WINDOWS FORM VỚI CONTROLS CƠ BẢN

Bài tập 2: Tạo thêm Project "**Lab02-02**" trong cùng Solution Lab02 (Sử dụng CheckBox, ComboBox, DataGridView)

Quản lý thông tin sinh viên cần lưu trữ các thông tin sau: Mã số sinh viên, Họ Tên, Giới Tính, Điểm Trung Bình và Tên Khoa. Thiết kế chương trình quản lý thông tin sinh viên tương tự như sau:



Có 3 khoa được đưa vào comboBox (QTKD, CNTT, NNA).

2.1 Khi mới Load Form

- Khoa được chọn mặc định là QTKD. Giới tính Nữ mặc định được checked. Tổng số sinh viên Nam/Nữ đều là 0.

2.2 Khi nhấn vào nút "Thêm/Sửa"

- Kiểm tra các thông tin bắt buộc phải nhập liệu cho sinh viên như mã sinh viên, tên, và điểm trung bình. Nếu để trống sẽ xuất hiện thông báo lỗi "Vui lòng nhập đầy đủ thông tin!".
- Nếu mã số sinh viên vừa nhập chưa có ở trong DataGridView (bên phải) thì Thêm mới dữ liệu sinh viên vừa nhập vào DataGridView, và thông báo "Thêm mới dữ liệu thành công!"

Nếu đã có MSSV trong DataGridView thì Cập nhật dữ liệu sinh viên vào DataGridView, và thông báo "Cập nhật dữ liệu thành công!"

2.3 Khi nhấn vào nút "Xóa"

- Kiểm tra nếu MSSV cần xóa không tồn tại trong DataGridView thì thông báo lỗi "Không tìm thấy MSSV cần xóa!".

- Ngược lại thì xuất hiện cảnh báo YES/NO. Nhấn YES sẽ thực hiện xóa dòng dữ liệu sinh viên trong DataGridView và thông báo "Xóa sinh viên thành công!".

2.4 Viết code cho sự kiện ở DataGridView, người dùng chọn 1 dòng thì thể hiện ngược lại thông tin của các sinh viên đã chọn ở phần nhập liệu (bên trái).

2.5 Tính toán lại số sinh viên Nam, Nữ phù hợp với các ngữ cảnh khi thay đổi dữ liệu

Hướng Dẫn

- Thiết kế giao diện tương tự như trên. Nên đặt các tên các Control dễ nhớ như: txtStudentID, txtFullName, optMale, optFemale, txtAverageScore, cmbFaculty, dgvStudent, btnUpdate, btnDelete…
- Ở ComboBox Khoa, để sẵn giá trị vào bằng cách vào Properties / Items rồi nhập các dòng (QTKD, CNTT, NNA)
- Thiết kế DataGridView có 5 cột (Columns) như yêu cầu. Đặt các tên column dễ nhớ. Điều chỉnh lại kích thước width cho phù hợp. Properties có SelectionMode = FullRowSelect
- Ở các sự kiện event, nên sử dụng try…catch để bắt lỗi.
- Trong sự kiện Form_Load (double click vào Form để sinh ra). Để chọn mặc định khoa ban đầu

```
private void frmQuanLySinhVien_Load(object sender, EventArgs e)
{
    cmbKhoa.SelectedIndex = 0;
}
```

- Viết sự kiện Thêm/ Sửa. Double click vào button để tạo event btnUpdate_Click

```
private int GetSelectedRow(string studentID)
{
    for (int i = 0; i < dgvStudent.Rows.Count; i++)
    {
        if( dgvStudent.Rows[i].Cells[0].Value.ToString() == studentID)
        {
            return i;
        }
    }
    return -1;
}
private void InsertUpdate(int selectedRow)
```

```
        {
            dgvStudent.Rows[selectedRow].Cells[0].Value = txtStudentID.Text;
            dgvStudent.Rows[selectedRow].Cells[1].Value = txtFullName.Text;
            dgvStudent.Rows[selectedRow].Cells[2].Value = optFemale.Checked ? "Nữ" : "Nam";
            dgvStudent.Rows[selectedRow].Cells[3].Value =
float.Parse(txtAverageScore.Text).ToString();
            dgvStudent.Rows[selectedRow].Cells[4].Value = cmbFaculty.Text;
           }

    private void btnUpdate_Click(object sender, EventArgs e)
    {
        try
        {
            if (txtStudentID.Text == "" || txtFullName.Text == "" || txtAverageScore.Text == "")
                throw new Exception("Vui lòng nhập đầy đủ thông tin sinh viên!");

            int selectedRow = GetSelectedRow(txtStudentID.Text);
            if(selectedRow == -1)   //TH Thêm mới
            {
                selectedRow = dgvStudent.Rows.Add();
                InsertUpdate(selectedRow);
                MessageBox.Show("Thêm mới dữ liệu thành công!", "Thông Báo", MessageBoxButtons.OK);
            }
            else //TH cập nhật
            {
                InsertUpdate(selectedRow);
                MessageBox.Show("Cập nhật dữ liệu thành công!", "Thông Báo", MessageBoxButtons.OK);
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
```

- Viết sự kiện Xóa

```
private void btnDelete_Click(object sender, EventArgs e)
{
    try
    {
        int selectedRow = GetSelectedRow(txtStudentID.Text);
        if (selectedRow == -1)
        {
            throw new Exception("Không tìm thấy MSSV cần xóa!");
        }
        else
        {
            DialogResult dr = MessageBox.Show("Bạn có muốn xóa ?", "YES/NO", MessageBoxButtons.YesNo);
            if (dr == DialogResult.Yes)
            {
                dgvStudent.Rows.RemoveAt(selectedRow);     //xóa thông tin sinh viên tại dòng tìm thấy
                MessageBox.Show("Xóa sinh viên thành công!", "Thông Báo", MessageBoxButtons.OK);
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Sinh viên tự thực hiện yêu cầu còn lại 2.4 và 2.5

Bài tập 3: Tạo thêm Project "**Lab02-03**" trong cùng Solution Lab02

Chương trình bán vé rạp chiếu phim

Rạp có 4 hàng ghế, mỗi hàng có 5 ghế, các ghế được đánh số từ 1 đến 20 và được phân thành 4 dãy như (hình trên):

Dãy 1 (ghế 1-5), Giá vé 30000đ

Dãy 2 (ghế 6-10), Giá vé 40000đ

Dãy 3 (ghế 11-15), Giá vé 50000đ

Dãy 4 (ghế 16-20), Giá vé 80000đ

Trên Form trình bày một sơ đồ các chổ ngồi để người sử dụng chọn vị trí muốn mua. Trên sơ đồ này cũng thể hiện những vị trí đã bán vé và những vị trí chưa bán vé bằng cách thể hiện màu khác nhau (ghế chưa bán vé màu trắng, ghế đã bán vé màu vàng).

Khi người sử dụng click chuột tại một vị trí trên sơ đồ thì:

- ✓ Nếu đây là vị trí chưa bán vé thì đổi màu của vị trí này sang màu xanh để cho biết đây là vị trí đang chọn.
- ✓ Nếu đây là vị trí đang chọn (có màu xanh) thì đổi màu của vị trí này trở về màu trắng
- ✓ Nếu đây là một vị trí đã bán vé (có màu vàng) thì xuất hiện một Message box thông báo cho người sử dụng biết vé ở vị trí này đã được bán. Sau khi đã chọn các vị trí người sử sụng có thể click chuột vào nút **CHỌN** hoặc **HỦY BỎ**

Nếu click vào nút **CHỌN** thì:

- ▪ Đổi màu các vị trí đã chọn (màu xanh) trên sơ đồ sang màu vàng (cho biết vị trí đã bán vé)
- ▪ Xuất lên một Label tổng số tiền phải trả cho số vé đã mua (phụ thuộc vào các vị trí đã chọn)

Nếu click vào nút **HỦY BỎ** thì:

- ▪ Đổi màu các vị trí đã chọn (màu xanh) trên sơ đồ sang màu trắng trở lại
- ▪ Xuất lên Label giá trị 0

Hướng Dẫn:

- Sử dụng GroupBox để gom nhóm (dễ dàng cho việc tìm kiếm, đổi màu, tính tiền…). Mặc định ban đầu các hàng ghế là trống `BackColor= White;`

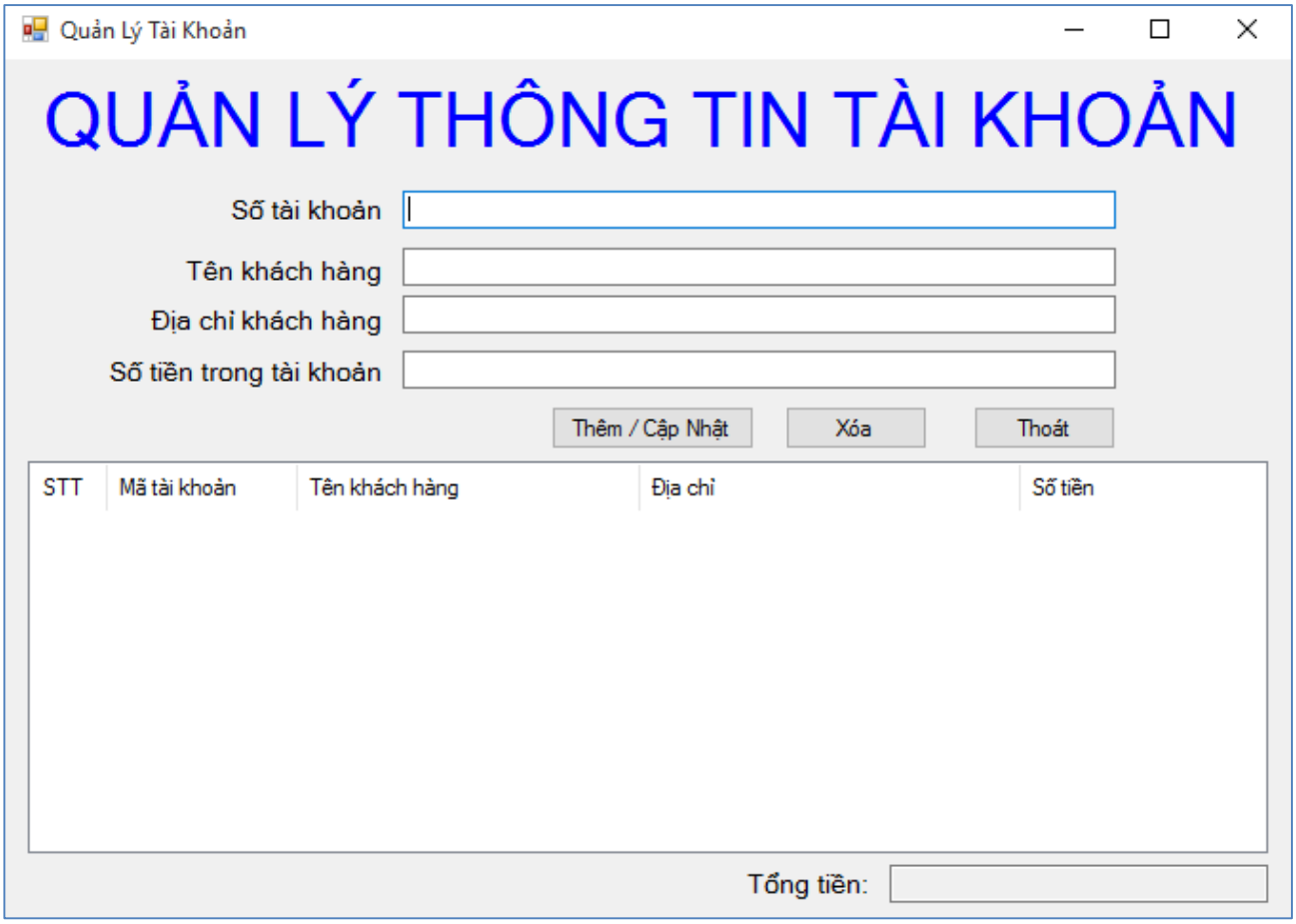
- Nên Viết 1 sự kiện dùng chung cho tất cả các nút bấm (1 – 20)

```
private void btnChooseASeat(object sender, EventArgs e)
{
      Button btn = sender as Button;
      if (btn.BackColor == Color.White)
            btn.BackColor = Color.Blue;
      else if (btn.BackColor == Color.Blue)
            btn.BackColor = Color.White;
      else if (btn.BackColor == Color.Yellow)
           MessageBox.Show("Ghế đã được bán!!");
}
```

- Có thể tạo giao diện ban đầu bằng Code (thay vì từ kéo thả)

Bài tập 4: Tạo thêm Project "**Lab02-04**" trong cùng Solution Lab02 (Sử dụng ListView)

Quản lý thông tin tài khoản cần lưu trữ các thông tin sau: số tài khoản, tên khách hàng, địa chỉ khách hàng và số tiền trong tài khoản. Thiết kế chương trình quản lý thông tin tài khoản cho một ngân hàng tương tự như sau:



4.1 Khi nhấn vào nút "Thêm/Cập Nhật"

- Kiểm tra các thông tin bắt buộc phải nhập liệu cho số tài khoản, tên, địa chỉ và số tiền. Xuất hiện thông báo lỗi "Vui lòng nhập đầy đủ thông tin!".
- Nếu chưa có dữ liệu ***số tài khoản*** trong ListView thì Thêm mới dữ liệu nhập vào ListView, tính lại tổng tiền và thông báo "Thêm mới dữ liệu thành công!"

Nếu đã tồn tại số tài khoản trong ListView thì Cập nhật dữ liệu vào ListView và tính lại tổng tiền và thông báo "Cập nhật dữ liệu thành công!"

4.2 Khi nhấn vào nút "Xóa"

- Kiểm tra nếu số tài khoản cần xóa tồn tại trong ListView , thì xuất hiện cảnh báo YES/NO

Nhấn YES sẽ thực hiện xóa dòng dữ liệu tài khoản trong ListView và thông báo "Xóa tài khoản thành công!".

- Nếu số tài khoản cần xóa không tồn tại thì thông báo lỗi "Không tìm thấy số tài khoản cần xóa!".

4.3 Viết code cho sự kiện ở ListView khi người dùng chọn 1 dòng thì thể hiện ngược lại ở phần nhập liệu đúng thông tin trên.