

Chương 6

Xây dựng giao diện ứng dụng bằng Visual Studio

6.0 Dẫn nhập

6.1 Tổng quát về xây dựng ứng dụng bằng VS .Net

6.2 Một số đối tượng giao diện thường dùng

6.3 Hiệu chỉnh thuộc tính các đối tượng giao diện

6.4 Sự kiện - Hàm xử lý sự kiện

6.5 Qui trình điển hình viết 1 ứng dụng bằng VC#

6.6 Thí dụ viết ứng dụng giải phương trình bậc 2

6.7 Kết chương



6.0 Dẫn nhập

- ❑ Chương này giới thiệu các đối tượng giao diện phổ dụng, qui trình tạo/xóa/hiệu chỉnh thuộc tính của đối tượng cũng như tạo hàm xử lý sự kiện cho 1 số sự kiện quan tâm trên đối tượng giao diện.
- ❑ Chương này cũng giới thiệu qui trình điển hình để xây dựng chương trình có giao diện đồ họa được thiết kế trực quan (thay vì phải viết code khó khăn).



6.1 Tổng quát về xây dựng ứng dụng bằng VS .Net

Một trong các yêu cầu quan trọng của các ứng dụng hiện nay là phải có tính thân thiện cao, gần gũi với người dùng. Để thỏa mãn yêu cầu này, ứng dụng thường sẽ hoạt động ở chế độ đồ họa trực quan.

Các class cấu thành chương trình dùng giao diện đồ họa được chia làm 2 nhóm chính :

- Các class miêu tả các đối tượng giao diện với người dùng như Form, Button, TextBox, Checkbox,... Nhiệm vụ của các đối tượng này là giúp người dùng có thể tương tác dễ dàng, trực quan với chương trình để nhập/xuất dữ liệu, để điều khiển/giám sát hoạt động của chương trình. Các đối tượng này còn che dấu mọi chi tiết về thuật giải và dữ liệu bên trong chương trình, người dùng không cần quan tâm đến chúng.
- Các class miêu tả các chức năng cần thực hiện của chương trình.



6.1 Tổng quát về xây dựng ứng dụng bằng VS .Net

- ❑ Viết code tường minh để đặc tả các đối tượng giao diện là 1 công việc rất khó khăn và tốn nhiều công sức, thời gian.
- ❑ Để giảm nhẹ công sức đặc tả các đối tượng giao diện, các môi trường lập trình trực quan (như Visual Studio .Net) đã viết sẵn 1 số đối tượng giao diện thường dùng và cung cấp công cụ để người lập trình thiết kế trực quan giao diện của ứng dụng bằng cách tích hợp các đối tượng giao diện có sẵn này : người lập trình đóng vai trò họa sĩ để vẽ/hiệu chỉnh kích thước, di chuyển vị trí các phần tử giao diện cần cho ứng dụng.
- ❑ Ngoài ra môi trường trực quan còn cho phép người lập trình tự tạo các đối tượng giao diện mới (User Control) để dùng trong các ứng dụng được viết sau đó (chương 9).



6.1 Tổng quát về xây dựng ứng dụng bằng VS .Net

- ❑ Qui trình viết ứng dụng theo cơ chế này được gọi là viết ứng dụng bằng cách lắp ghép các linh kiện phần mềm, nó giống như việc lắp máy tính từ các linh kiện phần cứng như CPU, RAM, disk, keyboard, monitor,... rất dễ dàng và nhanh chóng.
- ❑ Mọi phần tử giao diện, dù nhỏ hay lớn, dù đơn giản hay phức tạp, đều là cửa sổ (window). HĐH Windows sẽ quản lý các cửa sổ làm việc theo thời gian. Một ứng dụng có thể dùng nhiều cửa sổ trong quá trình hoạt động, nhưng từng thời điểm chỉ có 1 số ít cửa sổ được chương trình hiển thị để làm việc với người dùng.
- ❑ Chúng ta sẽ làm quen 1 số đối tượng giao diện, nắm được tính chất và khả năng của từng đối tượng để khi lập trình ứng dụng nào đó, ta sẽ chủ động chọn lựa và dùng chúng cho phù hợp với từng ngữ cảnh sử dụng.



6.2 Một số đối tượng giao diện thường dùng

Control buttons

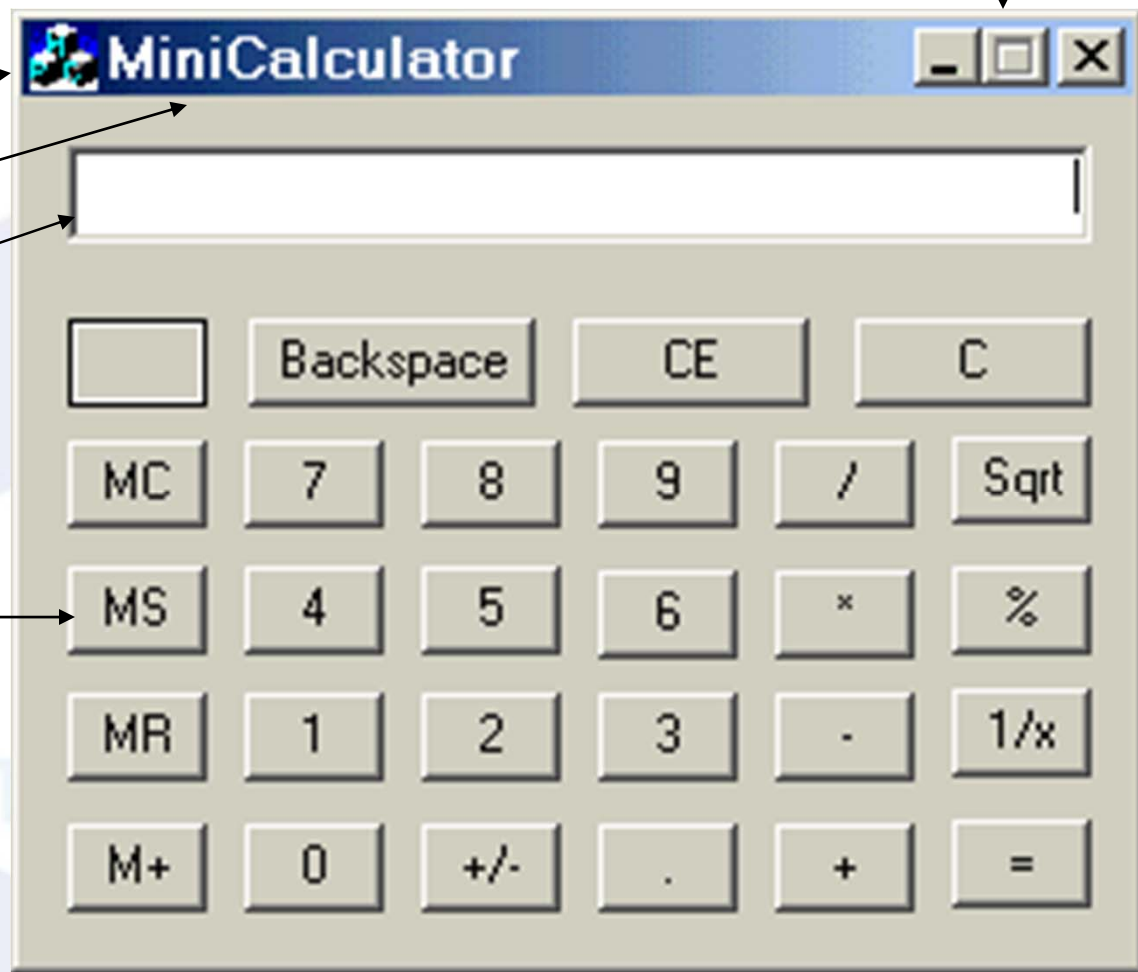
Window \equiv Form,

Dialogbox

Title bar

Textbox

Button



6.2 Một số đối tượng giao diện thường dùng

Label

DriveListBox

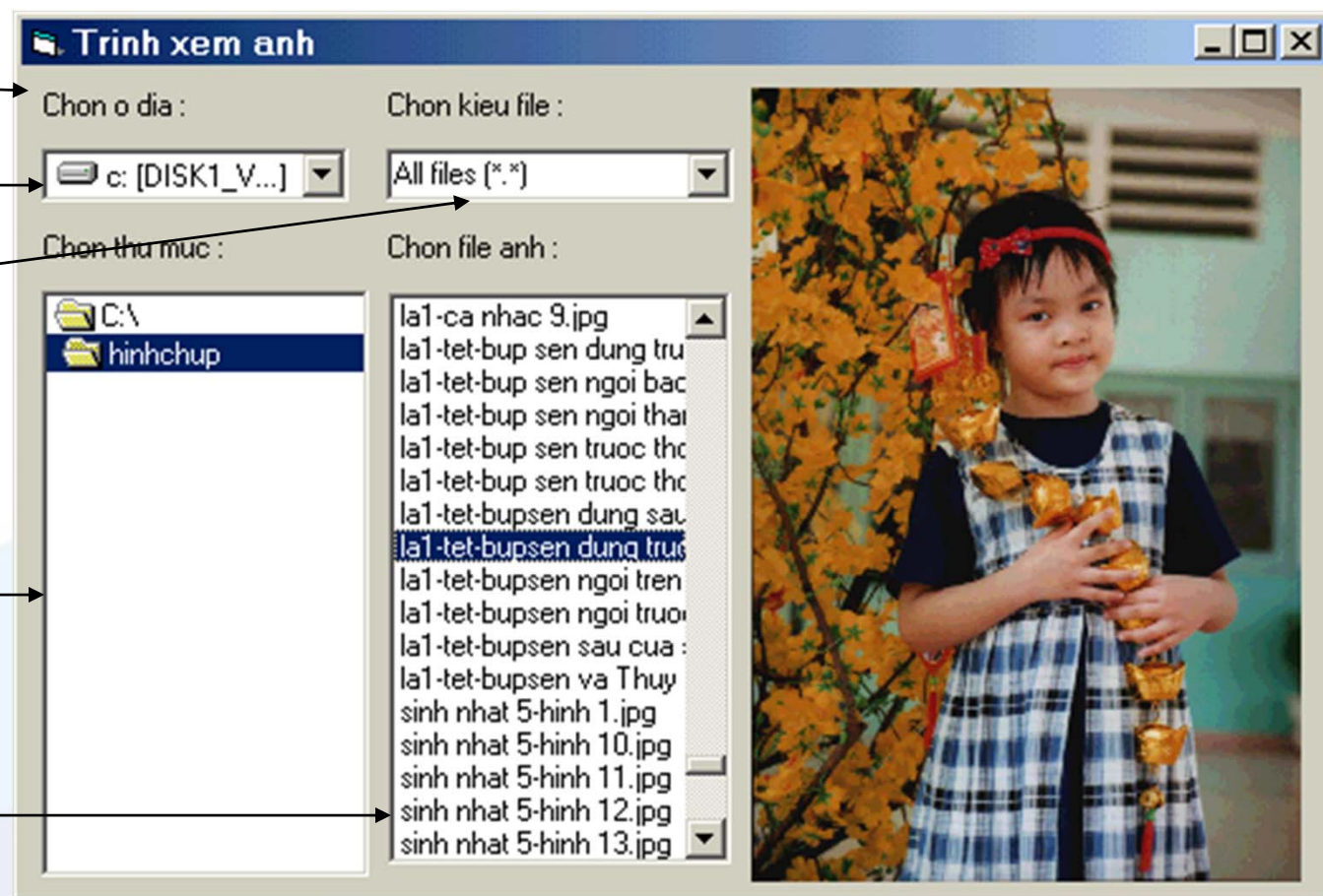
ComboBox \equiv

Textbox + ListBox

DirListBox

FileListBox \cong ListBox

PictureBox

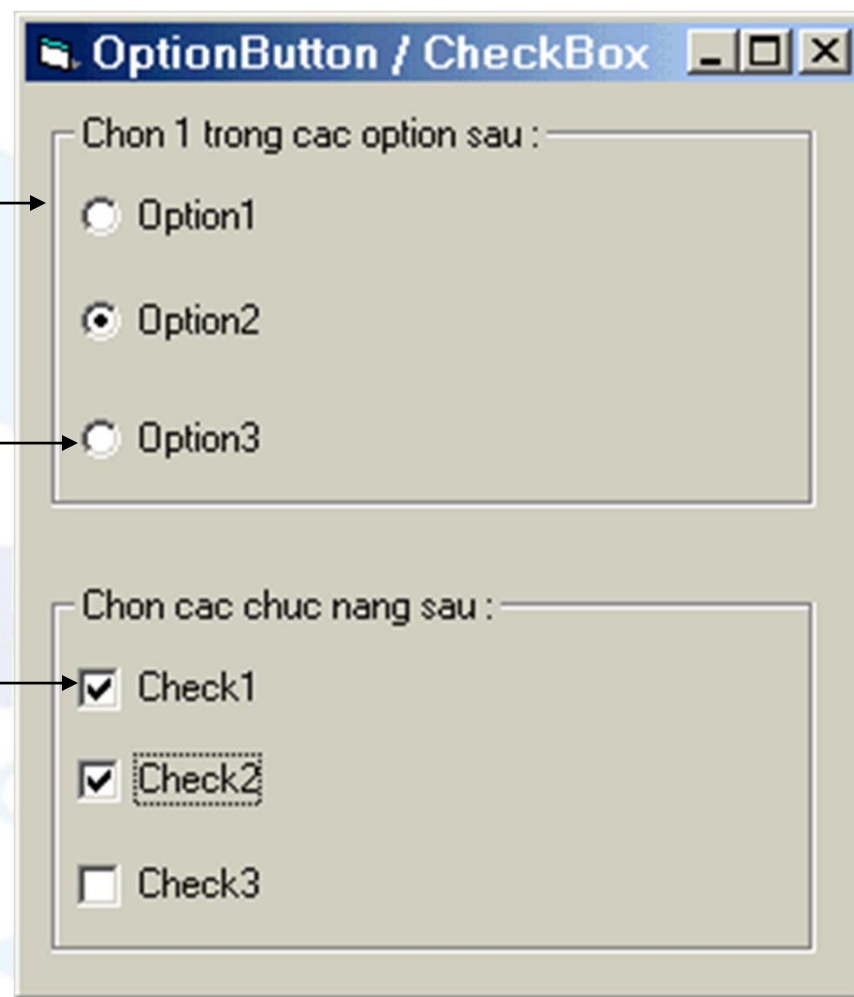


6.2 Một số đối tượng giao diện thường dùng

GroupBox

RadioButton

Checkbox



6.2 Một số đối tượng giao diện thường dùng

MenuTrip

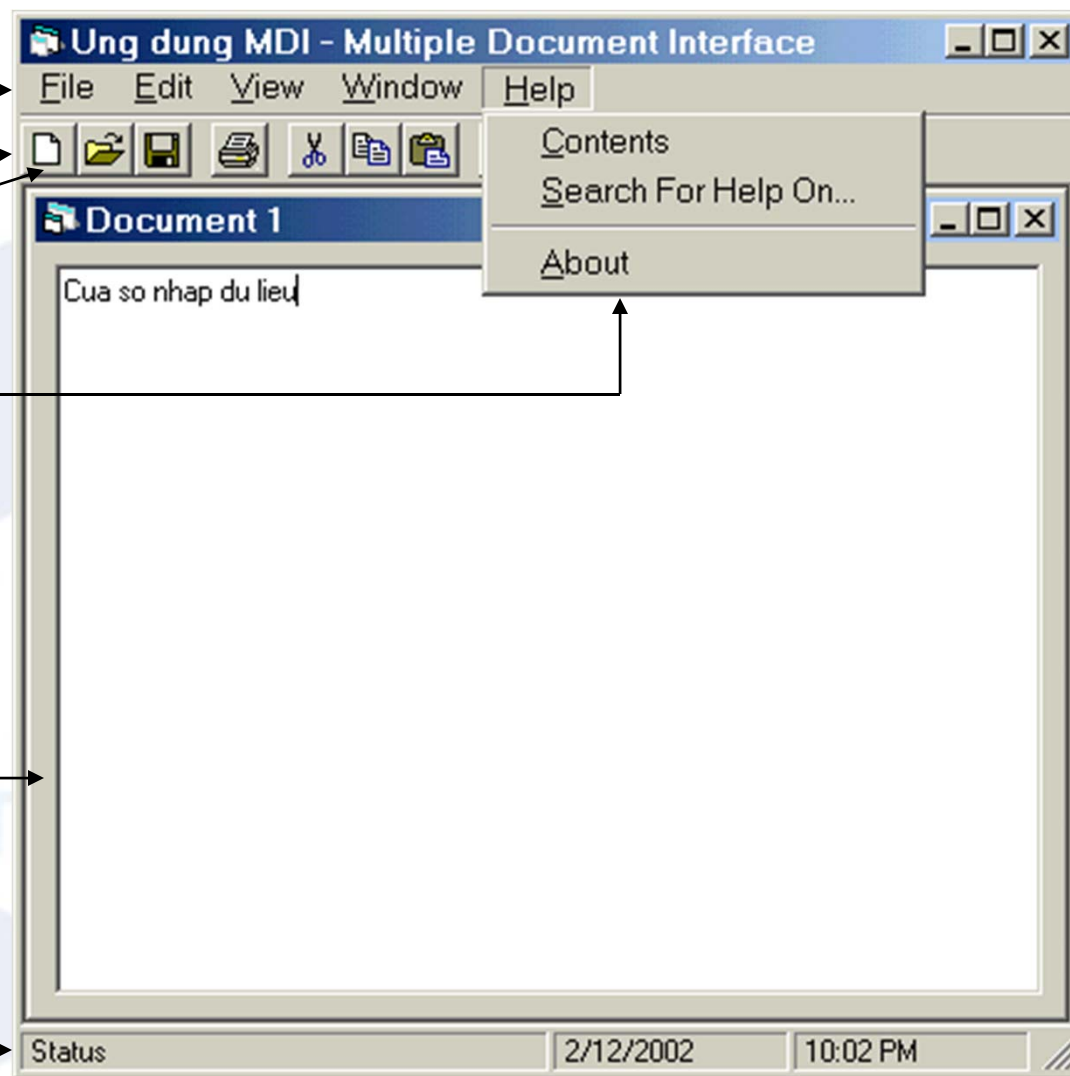
ToolTrip

Button

Pop-up Menu

1 window chứa 1
document của ứng
dụng

StatusTrip



Các tính chất chung của các đối tượng giao diện

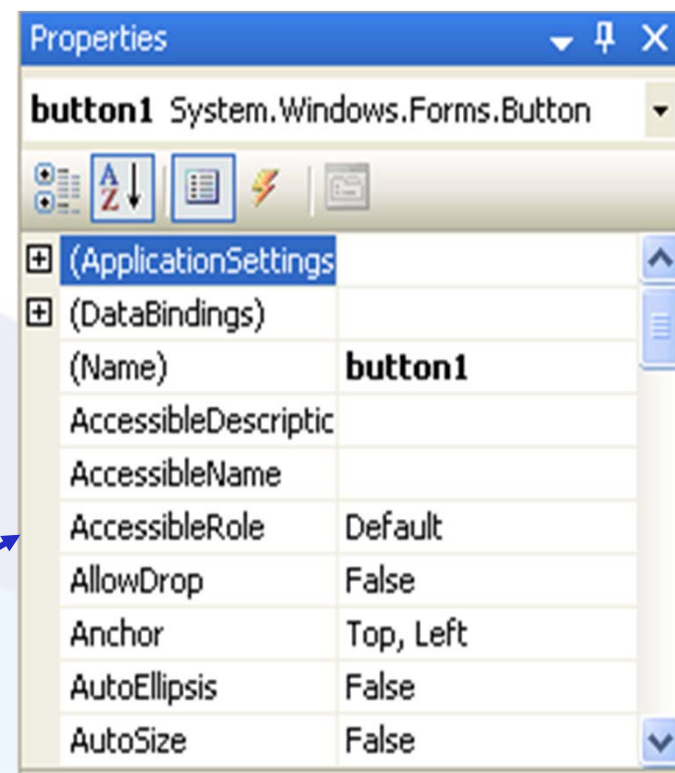
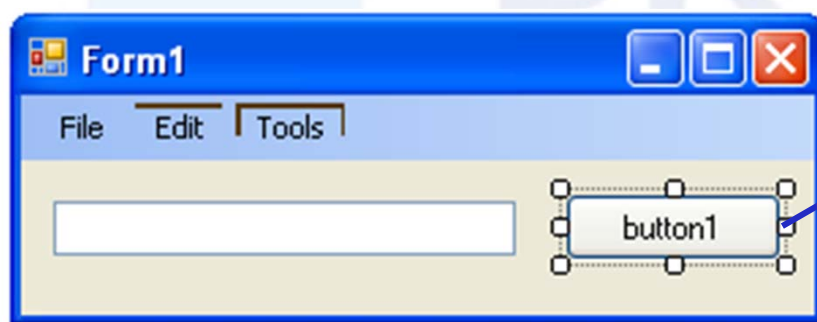
- ❑ Đối tượng giao diện có những tính chất giống như đối tượng bình thường, nó cũng được cấu thành từ các loại thành phần : thuộc tính, tác vụ, event, delegate...
- ❑ Mỗi đối tượng giao diện chứa khá nhiều thuộc tính liên quan đến nhiều loại trạng thái khác nhau :
 - (Name) : đây là thuộc tính đặc biệt, xác định tên nhận dạng của đối tượng, giá trị của thuộc tính này sẽ trở thành biến tham khảo đến đối tượng, code của ứng dụng sẽ dùng biến này để truy xuất đối tượng.
 - các thuộc tính xác định vị trí và kích thước (Layout) : Location, Size, Margin...
 - các thuộc tính xác định tính chất hiển thị : Text, Font, ForeColor, BackColor,...
 - các thuộc tính xác định hành vi (Behavoir) : Enable, Visible...
 - các thuộc tính liên kết dữ liệu database : DataBindings,...




6.3 Hiệu chỉnh thuộc tính các đối tượng giao diện

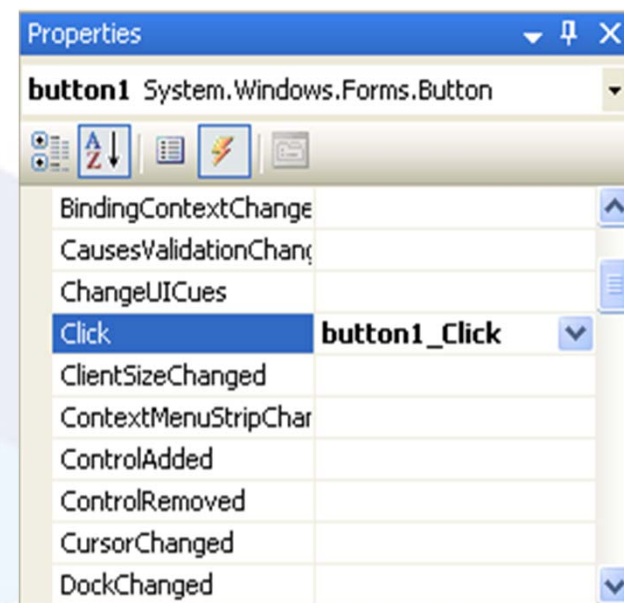
Khi tạo trực quan 1 đối tượng giao diện, môi trường đã gán giá trị đầu mặc định cho các thuộc tính, thường ta chỉ cần thay đổi 1 vài thuộc tính là đáp ứng được yêu cầu riêng. Có 2 cách để hiệu chỉnh giá trị 1 thuộc tính :

- trực quan thông qua cửa sổ thuộc tính của đối tượng giao diện.
- lập trình truy xuất thuộc tính của đối tượng giao diện.



6.4 Sự kiện - Hàm xử lý sự kiện

- ❑ Mỗi đối tượng giao diện có khá nhiều sự kiện để người dùng kích hoạt. Người lập trình có thể định nghĩa hàm xử lý kết hợp với sự kiện cần xử lý. Khi ứng dụng chạy, lúc người dùng kích hoạt sự kiện, hàm xử lý sự kiện tương ứng (nếu có) sẽ chạy.
- ❑ Thí dụ khi user ấn chuột vào button tên "button1", hệ thống tạo ra sự kiện "Click" để kích khởi hàm button1_Click() chạy.
- ❑ Muốn tạo hàm xử lý sự kiện trên đối tượng giao diện, ta chọn đối tượng, cửa sổ thuộc tính của nó sẽ hiển thị, click icon  để hiển thị danh sách các sự kiện của đối tượng, duyệt tìm sự kiện cần xử lý, nhập tên hàm xử lý vào combobox bên phải sự kiện (hay ấn kép chuột vào comboBox để máy tạo tự động hàm xử lý).



6.5 Qui trình điển hình viết 1 ứng dụng bằng VC#

1. Trước hết phải nắm bắt yêu cầu phần mềm để xác định các chức năng mà ứng dụng phải cung cấp cho người dùng.
2. Phân tích sơ lược từng chức năng và tìm ra các class phân tích cấu thành chức năng tương ứng.
3. Thiết kế chi tiết các class phân tích : xác định các thuộc tính và các tác vụ cũng như phác họa giải thuật của từng tác vụ. Phân loại các class phần mềm thành 2 nhóm : nhóm các đối tượng giao diện (các form giao diện) và nhóm các class miêu tả thuật giải các chức năng bên trong của ứng dụng. Trong các ứng dụng nhỏ dùng thuật giải đơn giản, ta thường đặt các thuật giải chức năng trực tiếp trong các hàm xử lý sự kiện của các đối tượng giao diện.



6.5 Qui trình điển hình viết 1 ứng dụng bằng VC#

4. Hiện thực phần mềm bằng VC# gồm 2 công việc chính :

- thiết kế trực quan các form giao diện người dùng : mỗi form chứa nhiều phần tử giao diện, các phần tử giao diện thường đã có sẵn, nếu không ta phải tạo thêm 1 số đối tượng giao diện mới (User Control). Ứng với mỗi phần tử giao diện vừa tạo ra, nên thiết lập giá trị đầu cho thuộc tính "Name" và 1 vài thuộc tính cần thiết.
- tạo hàm xử lý sự kiện cho các sự kiện cần thiết trên các phần tử giao diện rồi viết code cho từng hàm xử lý sự kiện vừa tạo ra.




6.6 Thí dụ viết ứng dụng giải phương trình bậc 2

1. Chạy VS .Net, chọn menu File.New.Project để hiển thị cửa sổ New Project.
2. Mở rộng mục Visual C# trong TreeView "Project Types", chọn mục Window, chọn icon "Windows Application" trong listbox "Templates" bên phải, thiết lập thư mục chức Project trong listbox "Location", nhập tên Project vào textbox "Name:", click button OK để tạo Project theo các thông số đã khai báo.
3. Form đầu tiên của ứng dụng đã hiển thị trong cửa sổ thiết kế, việc thiết kế form là quá trình lập 4 thao tác tạo mới/xóa/hiệu chỉnh thuộc tính/tạo hàm xử lý sự kiện cho từng đối tượng cần dùng trong form.



6.6 Thí dụ viết ứng dụng giải phương trình bậc 2

4. Nếu cửa sổ Toolbox chưa hiển thị chi tiết, chọn menu View.Toolbox để hiển thị nó (thường nằm ở bên trái màn hình). Click chuột vào button  (Auto Hide) nằm ở góc trên phải cửa sổ Toolbox để chuyển nó về chế độ hiển thị thường trực.
5. Duyệt tìm phần tử Label (trong nhóm Common Controls), chọn nó, dờ chuột về vị trí thích hợp trong form và vẽ nó với kích thước mong muốn. Hiệu chỉnh thuộc tính Text = "Nhập a :". Nếu cần, hãy thay đổi vị trí và kích thước của Label và của Form.
6. Duyệt tìm phần tử TextBox (trong nhóm Common Controls), chọn nó, dờ chuột về vị trí bên phải Label vừa vẽ và vẽ nó với kích thước mong muốn. Hiệu chỉnh thuộc tính (Name) = txtA. Nếu cần, hãy thay đổi vị trí và kích thước của TextBox.



6.6 Thí dụ viết ứng dụng giải phương trình bậc 2

7. Lặp lại các bước 4, 5 để vẽ 2 Label "Nhập b :", "Nhập c :", 2 TextBox có (Name) = txtB, txtC, 1 button "Bắt đầu giải" có (Name) = btnStart, 3 Label có (Name) lần lượt là lblKetqua, lblX1, lblX2.

- ❑ Đối với các đối tượng giống nhau, ta có thể dùng kỹ thuật Copy-Paste để nhân bản vô tính chúng cho dễ dàng.
- ❑ Sau khi thiết kế xong, Form có dạng sau :



The screenshot shows a Windows Form titled "Form1" with a standard Windows XP-style title bar. The form contains three text input fields stacked vertically, each preceded by a label: "Nhập a:", "Nhập b:", and "Nhập c:". Below these fields is a button labeled "Bắt đầu giải". At the bottom of the form, there are three placeholder labels, each labeled "label4". The first two are plain text, while the third one has a small square icon to its left, indicating it might be a checkbox or a similar interactive element.

6.6 Thí dụ viết ứng dụng giải phương trình bậc 2

8. Dời chuột về button "Bắt đầu giải", ấn kép chuột vào nó để tạo hàm xử lý sự kiện Click chuột cho button, cửa sổ mã nguồn sẽ hiển thị để ta bắt đầu viết code cho hàm. Lưu ý rằng để tạo hàm xử lý sự kiện bất kỳ cho đối tượng 1 cách chính quy, ta phải hiển thị cửa sổ thuộc tính của đối tượng, rồi hiển thị danh sách các sự kiện rồi mới định nghĩa hàm xử lý sự kiện mong muốn.

9. Viết code cho hàm btnStart_Click() như sau :

```
private void btnStart_Click(object sender, EventArgs e) {
```

```
    //định nghĩa các biến cần dùng
```

```
    double a, b, c;
```

```
    double delta;
```

```
    double x1, x2;
```



6.6 Thí dụ viết ứng dụng giải phương trình bậc 2

```
//mã hóa chuỗi nhập thành giá trị thực a,b,c
a = Double.Parse(txtA.Text);
b = Double.Parse(txtB.Text);
c = Double.Parse(txtC.Text);
//tính biệt số delta của phương trình
delta = b * b - 4 * a * c;
if (delta >= 0) { //nếu có nghiệm thực
    x1 = (-b + Math.Sqrt(delta)) / 2 / a;
    x2 = (-b - Math.Sqrt(delta)) / 2 / a;
    lblKetqua.Text = "Phương trình có 2 nghiệm thực :";
    lblX1.Text = "X1 = " + x1;
    lblX2.Text = "X2 = " + x2;
}
```



6.6 Thí dụ viết ứng dụng giải phương trình bậc 2

```
else { //nếu vô nghiệm
    lblKetqua.Text = "Phương trình vô nghiệm";
    lblX1.Text = lblX2.Text = "";
}
}
```

10. Hiệu chỉnh hàm khởi tạo form như sau :

```
public Form1() {
    InitializeComponent();
    //xóa nội dung ban đầu của các Label kết quả
    lblKetqua.Text = lblX1.Text = lblX2.Text = "";
}
```

11. Chọn menu Debug.Start Debugging để dịch và chạy ứng dụng. Hãy thử nhập từng bộ ba (a,b,c) của phương trình bậc 2 rồi ấn button "Bắt đầu giải" để giải và xem kết quả.



6.0 Kết chương

- ❑ Chương này đã giới thiệu các đối tượng giao diện phổ dụng, qui trình tạo/xóa/hiệu chỉnh thuộc tính của đối tượng cũng như tạo hàm xử lý sự kiện cho 1 số sự kiện quan tâm trên đối tượng giao diện.
- ❑ Chương này cũng đã giới thiệu qui trình điển hình để xây dựng chương trình có giao diện đồ họa được thiết kế trực quan (thay vì phải viết code khó khăn).

