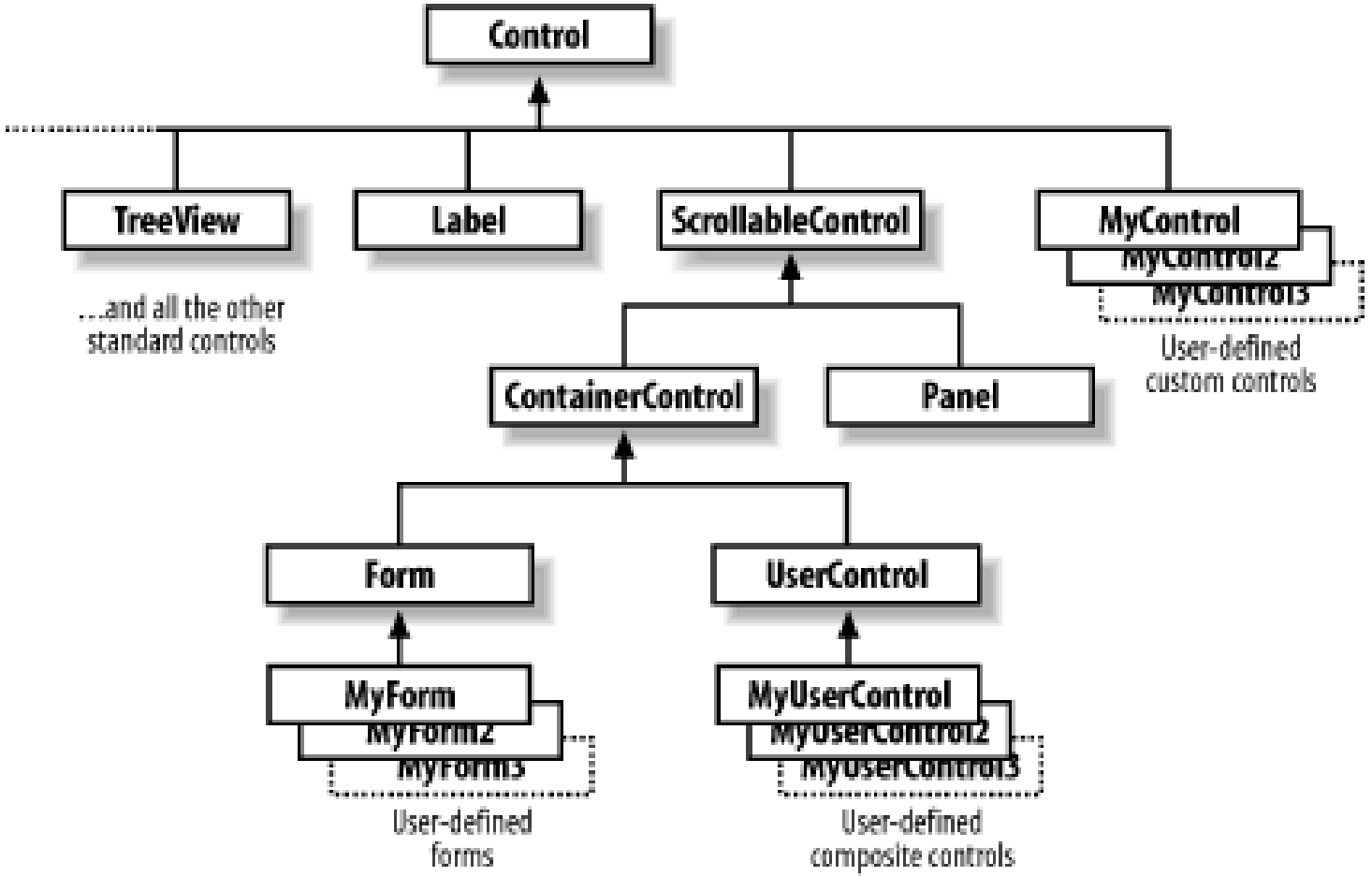
Lập trình GUI

Lập trình GUI

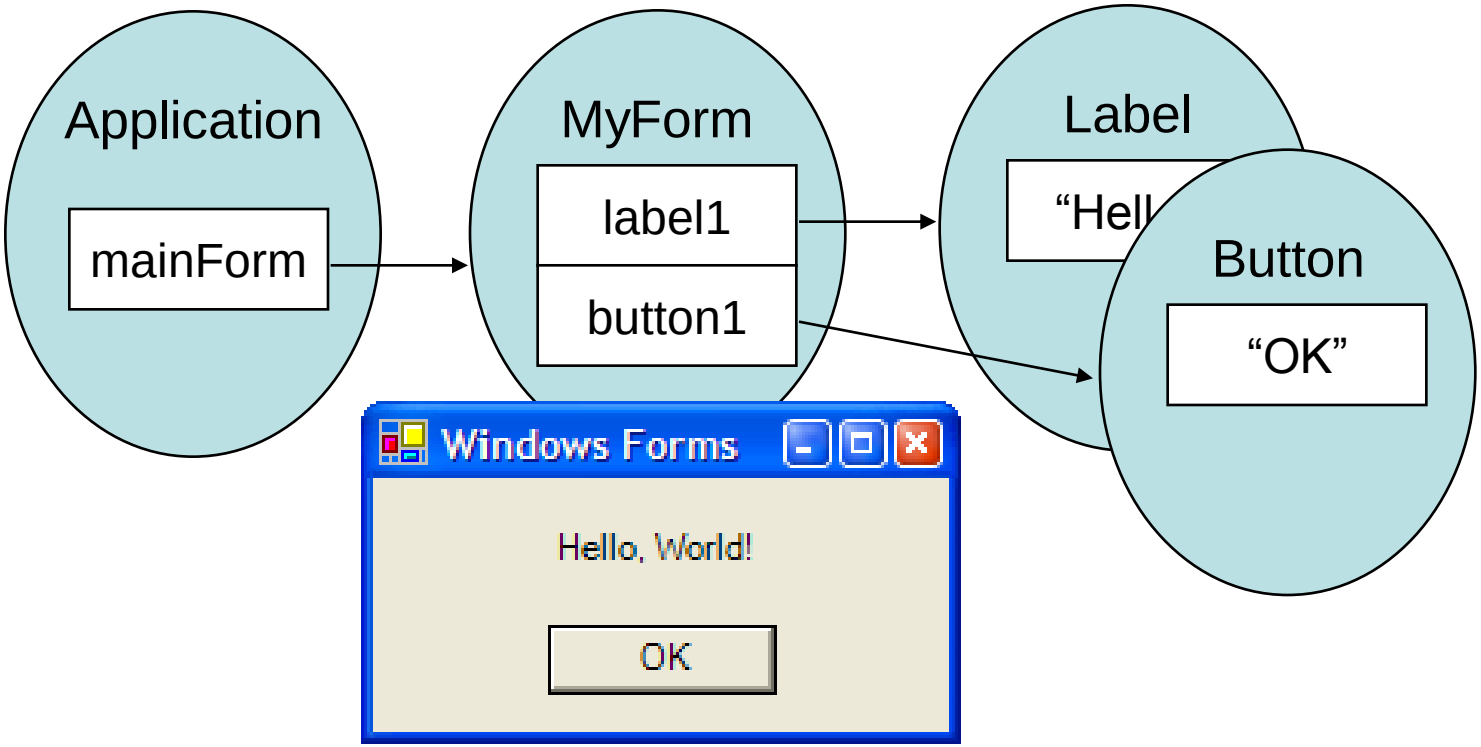
- User interface modeling
- User interface architecture
- User interface coding

The Control class hierarchy



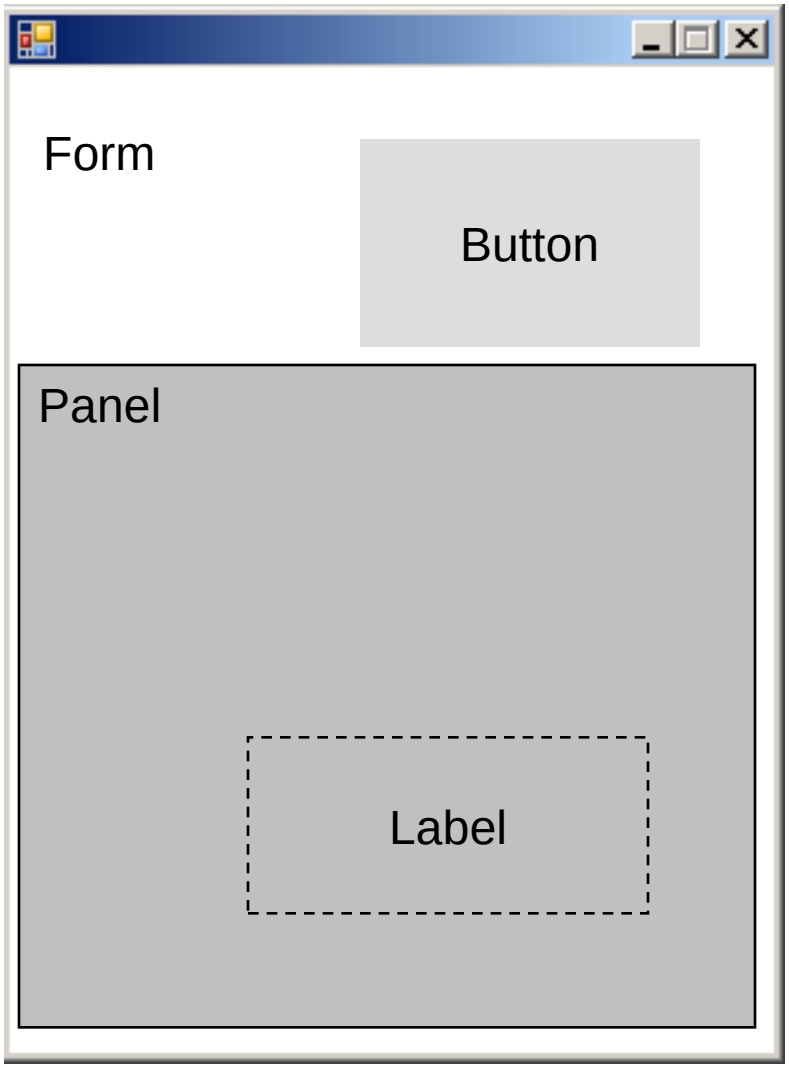
Windows Forms Application Structure

- A Windows Forms application has three pieces
  - the application itself
  - forms in the application
  - controls on the form

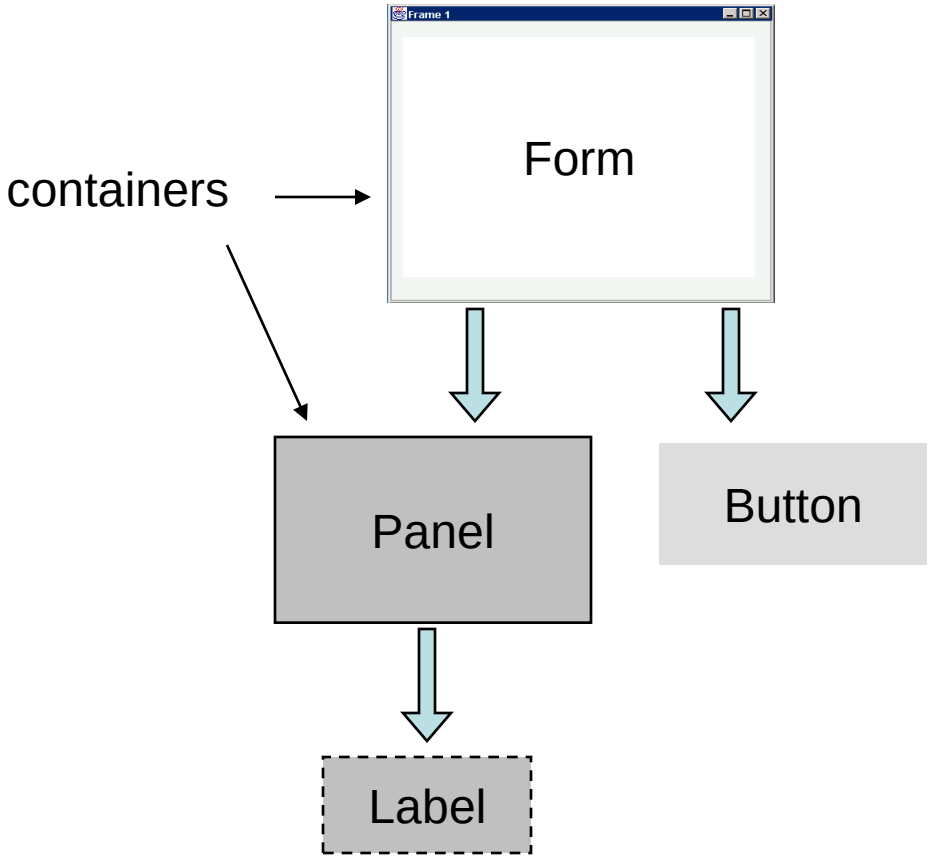


GUI Tree Structure

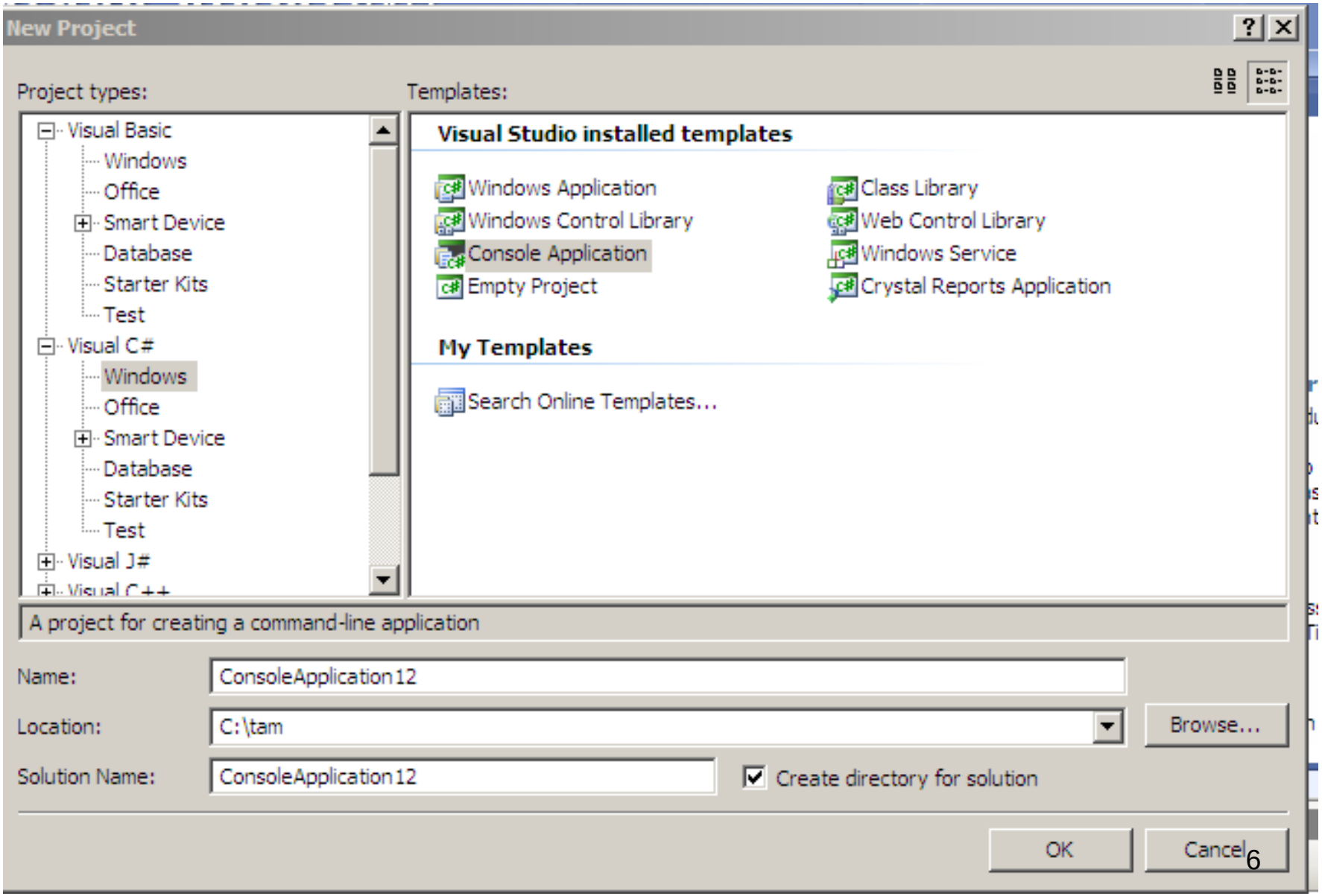
GUI



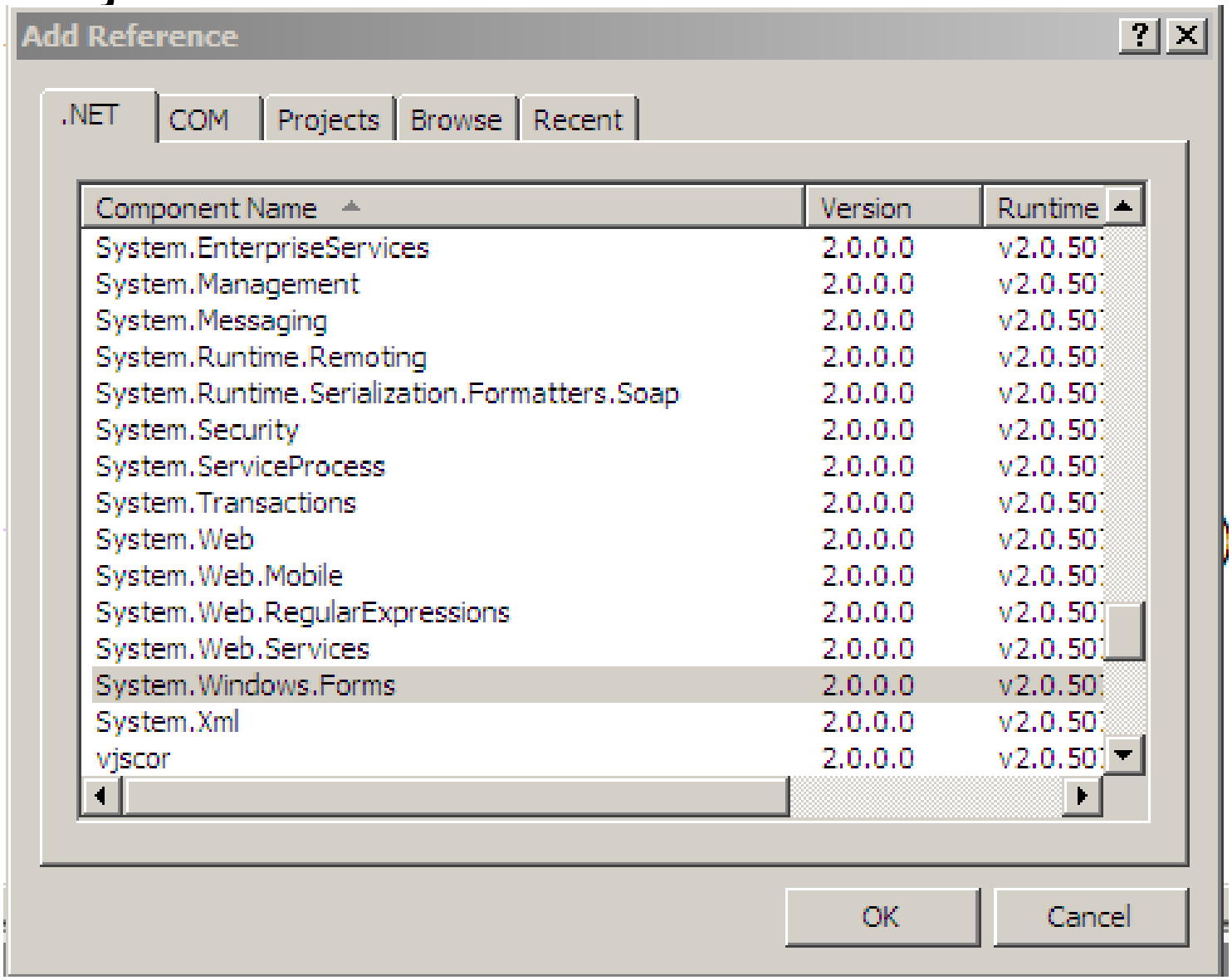
Internal structure



Cách tạo WinForm bằng Console Application



- Project → Add Reference

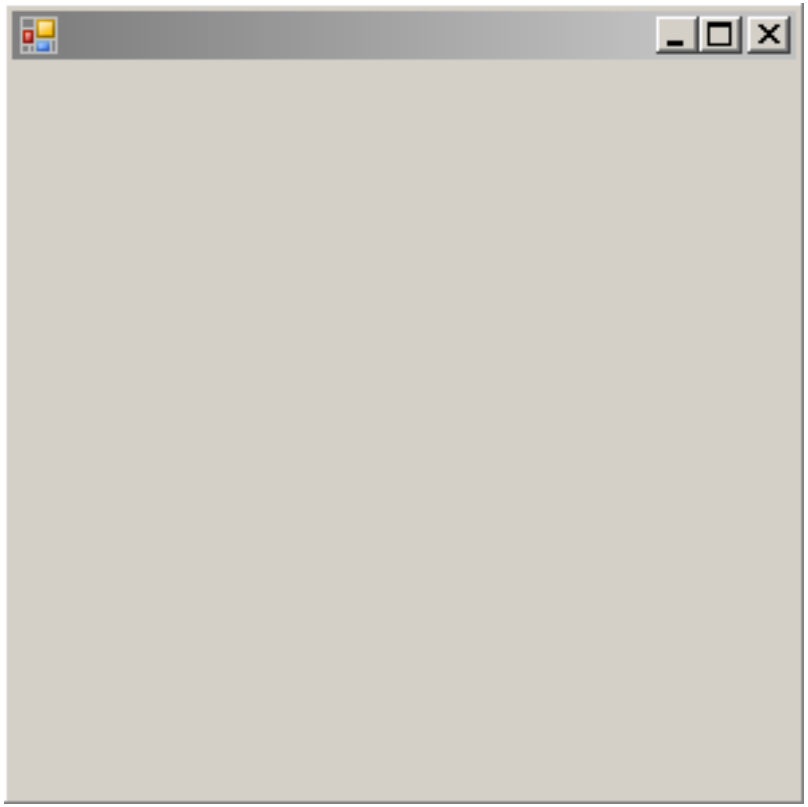


Form

- Một « form » là một cửa sổ màn hình - một đơn vị giao diện người dùng do Microsoft đưa ra kể từ Windows 1.0
- Một ứng dụng Windows Forms (WinForms) phải có ít nhất một cửa sổ « main form » (cửa sổ chính
- Form có thể chứa các component
- Form có thể có các file resource

Ví dụ 1

```
class Program
    {
        static void Main(string[] args)
        {
            Form f = new Form();
            Application.Run(f);
        }
    }
```



Ví dụ 2

```
class Program
   {
      static void Main(string[] args)
      {
         MessageBox.Show("Hello World");
      }
   }
```

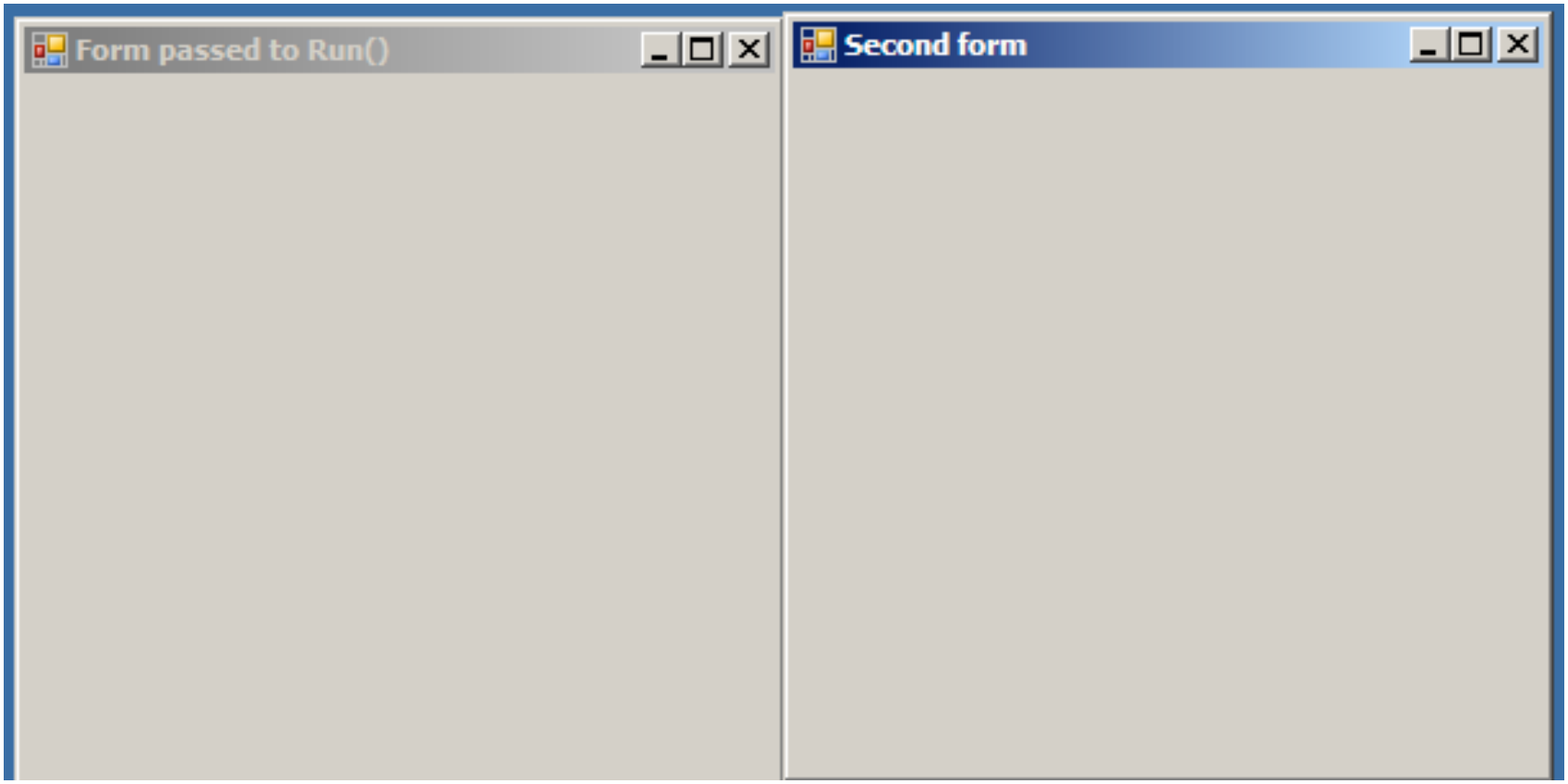
Application class

Exit	Stops all running message loops and closes all windows in the application. Note that this may not force the application to exit
Run	Starts a standard message loop on the current thread. If a Form is given, also makes that form visible.
DoEvents	Processes any Windows messages currently in the message queue.

Ví dụ 3

```
public static void Main()
        {
            Form form1 = new Form();
            Form form2 = new Form();

            form1.Text = "Form passed to Run()";
            form2.Text = "Second form";
            form2.Show();

            Application.Run(form1);

            MessageBox.Show("Application.Run() has returned control
                                        back to Main. Bye, bye!",
                                         "TwoForms");
        }
```

Ví dụ 3



Form Properties

Thuộc tính	Kiểu	Mô tả
FormBorderStyle	FormBorderStyle: FixedDialog, Fixed3D…	Kiểu đường viền
ControlBox	bool	Có system menu box?
MaximizeBox	bool	
MinimizeBox	bool	
Icon	Icon	
ShowInTaskBar	bool	
StartPosition	FormStartPosition	

Form Properties

Thuộc tính	Kiểu	Mô tả
SizeGripStyle	SizeGripStyle: Show, Hide…	
WindowState	FormWindowState: Normal, Maximized, Minimized	
TopMost	bool	
Text	string	
Size	Point	
ForeColor	color	
Font	font	
Location	Point	

Form Properties

Thuộc tính	Kiểu	Mô tả
AcceptButton		
CancelButton		

StartPosition - FormBorderStyle

- **CentreParent** cho modal dialogs
- **CentreScreen** cho main form hay splash screen
- **WindowsDefaultLocation**

- **FixedDialog** : modal dialog boxes
- **FixedSingle** : main form
- **None** : splash screen
- **Sizable**

Ví dụ 4

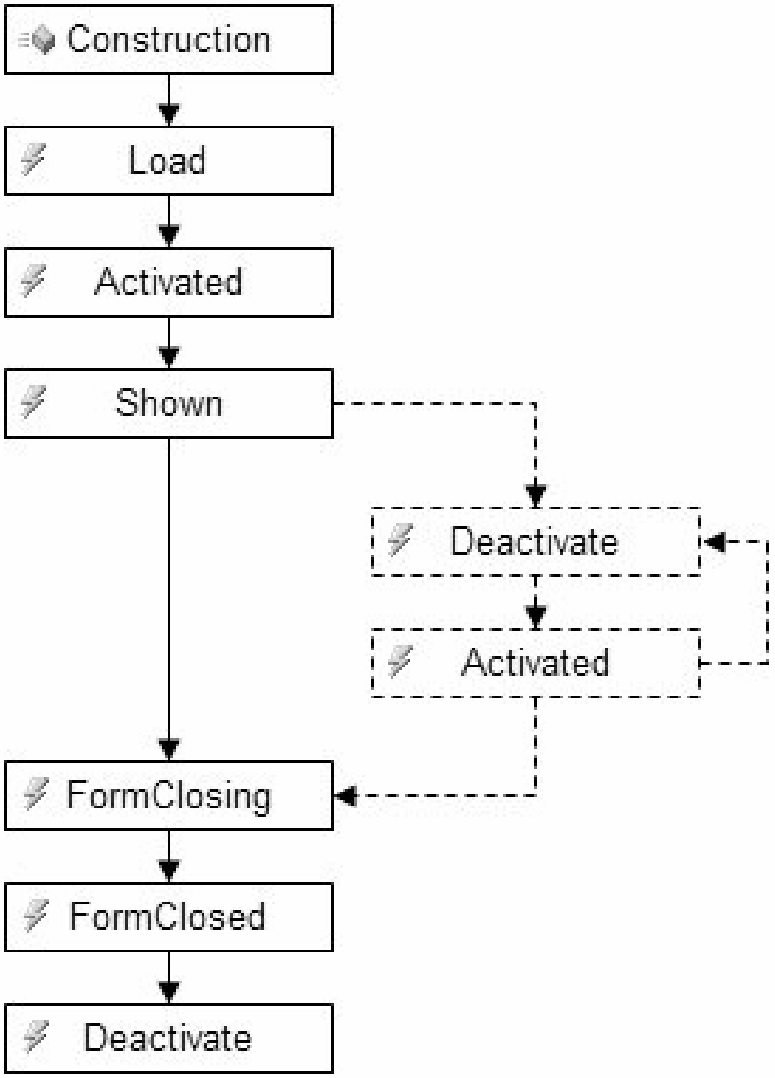
Ví dụ 4

```
static void Main(string[] args)
        {
            Form form = new Form();

            form.Text = "Form Properties";
            form.BackColor = Color.BlanchedAlmond;
            form.Width *= 2;
            form.Height /= 2;
            form.FormBorderStyle = FormBorderStyle.FixedSingle;
            form.MaximizeBox = false;
            form.Cursor = Cursors.Hand;
            form.StartPosition = FormStartPosition.CenterScreen;

            Application.Run(form);
        }
```

Form Method

- Show()
- ShowDialog();
- Hide();
- Close();

Form Lifetime Event Sequence



Form Event

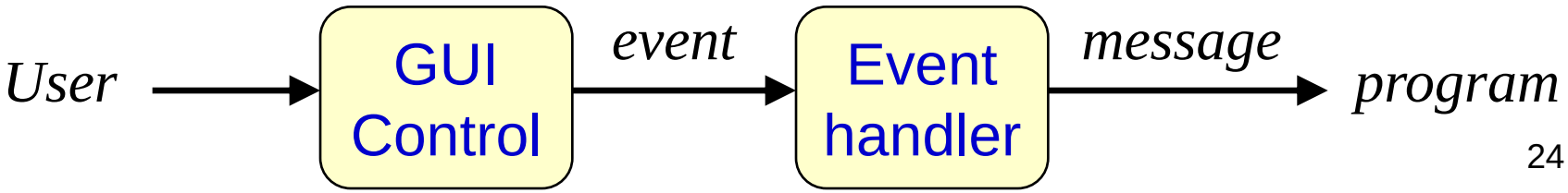
- Click
- DoubleClick
- KeyDown
- MouseHover
- Paint
- Resize
- ……

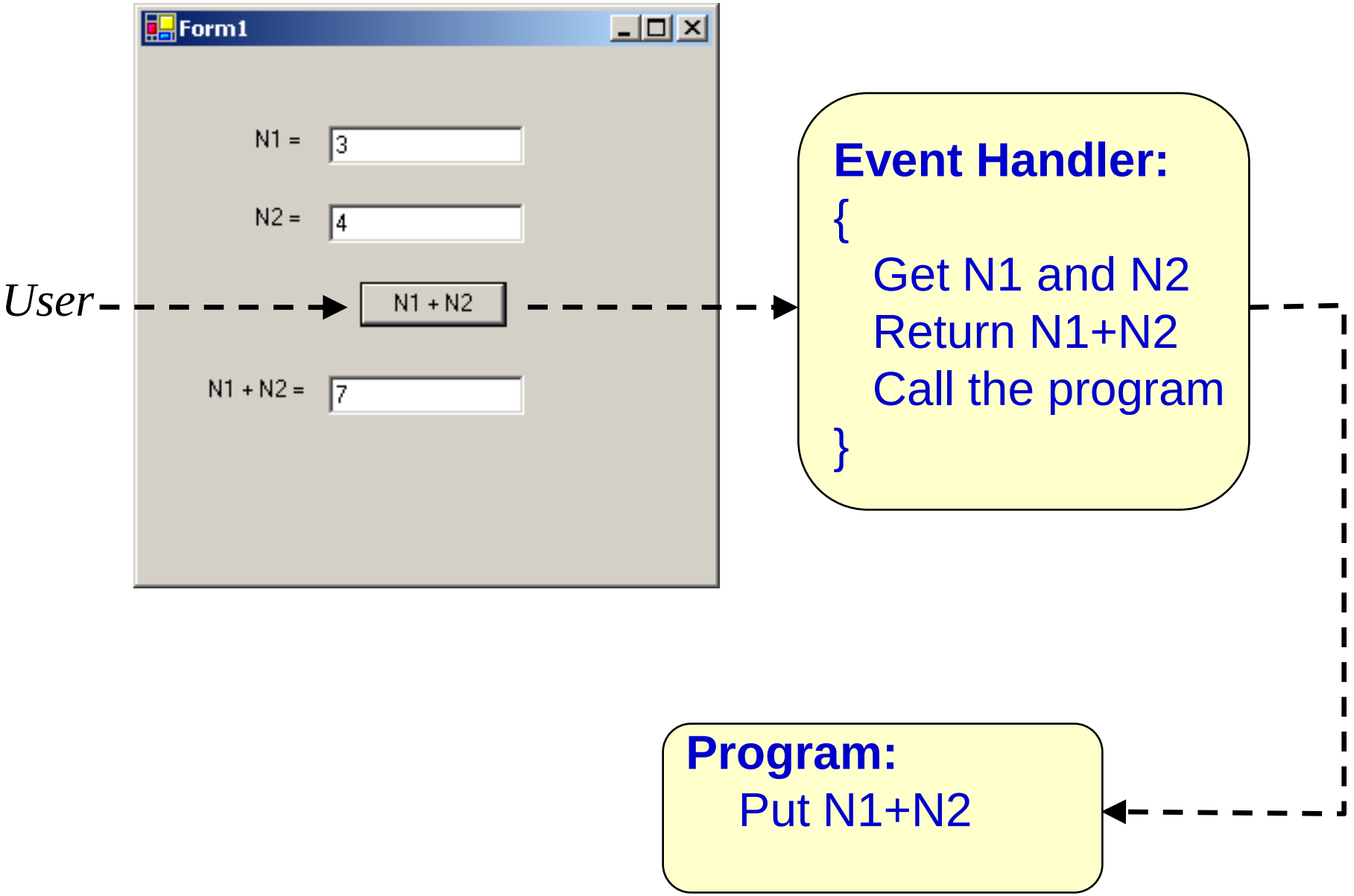
Sự kiên form Load

```
class Program
   {
            static void Main(string[] args)
           {
                Form f = new Form();
                f.Load += new EventHandler(f_Load);
                Application.Run(f);
         }
       private static void f_Load(object sender, EventArgs e)
       {
           MessageBox.Show("Hello ");
       }
      }
```

Events

- Một *event* là một đối tượng biểu diễn một hành động
- Ví dụ:
  - The mouse is moved or button clicked
  - The mouse is dragged
  - A graphical button is clicked
  - A keyboard key is pressed
  - A timer expires
- Sự kiện thường tương ứng với thao tác của người dùng
- Có thể viết các bộ đáp ứng sự kiện




Form1
N1 =
3
N2 =
4
N1 + N2
N1 + N2 =
7
User
Event Handler:
{
Get N1 and N2
Return N1+N2
Call the program
}
Program:
Put N1+N2

Button

Call me when you are pressed

Event Handler

Button

I have been pressed

Event Handler

GUI Control

Handler 1

Handler 2

Handler 3

GUI Control

GUI Control

Handler

Object A raises event E

calls

Delegate for event E

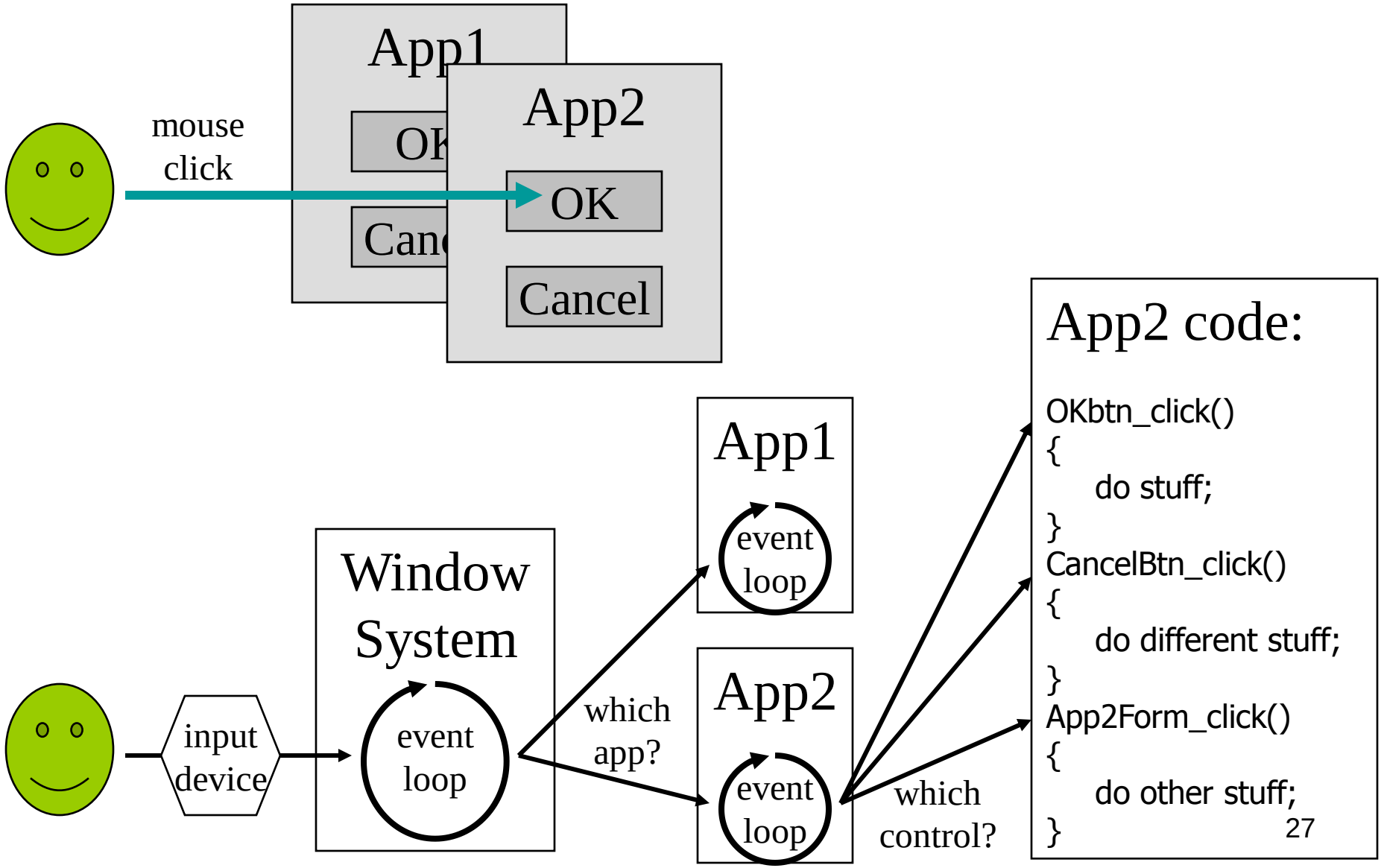
calls

Handler 1 for event E

Handler 2 for event E

Handler 3 for event E

GUI Events


App1
App2
mouse
click
OK
Cancel
Window
System
input
device
event
loop
which
app?
App1
event
loop
App2
event
loop
which
control?
App2 code:
OKbtn_click()
{
do stuff;
}
CancelBtn_click()
{
do different stuff;
}
App2Form_click()
{
do other stuff;
}

GUI program

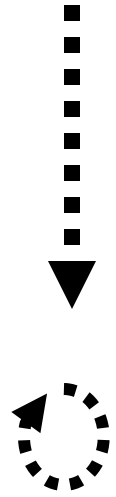
- User input commands
- Non-linear execution
- Unpredictable order
- Much idle time
- Event callback procs

GUI program:

```
main()
{
      decl data storage;
      initialization code;

      create GUI;
      register callbacks;

      main event loop;
}

Callback1()    //button1 press
{     code;
}
Callback2()    //button2 press
{     code;
}
...
```

C# WinApp

- “delegates” = callbacks
- Function pointers
- Listeners

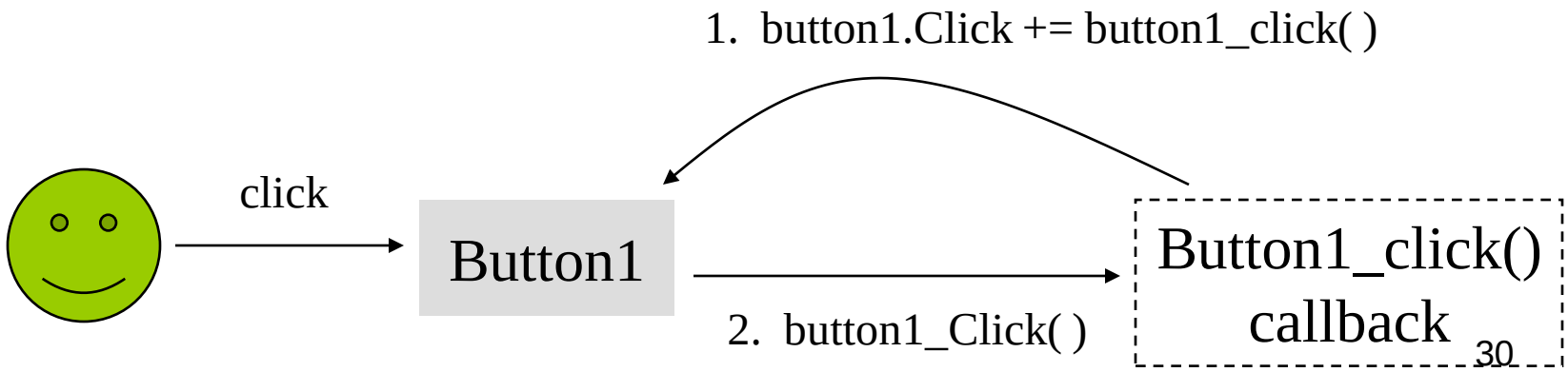


C# WinApp:

```
Class{
  decl data storage;

  constructor(){
      initialization code;
      create GUI controls;
      register callbacks;
  }
  main(){
      Run(new )
  }
  callback1(){
      do stuff;
  }
  callback2(){
      do stuff;
  }
...
```

Delegates

1. Đăng ký control để nhận events
   - Gắn vào Control một function pointer để gọi callback function
   - **F.Load += new EventHandler(MyLoadHandler);**
2. Nhận events từ control
   - Control sẽ gọi function pointer
   - **private void button1_Click(object sender, EventArgs e){**



Event Handler

- Thông điệp gửi đi bằng cách chuyển giao.
- Bộ xử lý sự kiện(Event Handler) sẽ được gọi khi sự kiện tương ứng phát sinh

void EventMethodName(Object sender, EventArgs e)

Form - Click

```
static void Main(string[] args)
        {
            Form f = new Form();
            f.Click+=new EventHandler(f_Click);
            Application.Run(f);

        }
        private static void f_Click(object sender, EventArgs e)
        {
            MessageBox.Show("Bạn đã click chuột");
        }
```

Paint Event

```
class Program
   {
       static void Main(string[] args)
       {
           Form f = new Form();
          f.Click+=new EventHandler(f_Click);
           Application.Run(f);

       }
       static void f_Click(Object sender, EventArgs e) {
           Form f = (Form)sender;
           Graphics gx = f.CreateGraphics();
           gx.DrawString("Form 1 \n Form 1\n", f.Font, Brushes.Black, 30, 30);
       }
 }
```

Paint Event

```
class Program
  {
      static void Main(string[] args)
      {
          Form f = new Form();
          f.Paint+=new PaintEventHandler(f_Paint);
          Application.Run(f);

      }
      static void f_Paint(Object sender, PaintEventArgs e)
      {

          Form f = (Form)sender;
          Graphics gx = f.CreateGraphics();
          gx.DrawString("Form 1 \n Form 1\n", f.Font, Brushes.Black, 30, 30);
      }
 }
```

```
static void Main(string[] args)
        {
            Form f1 = new Form();
            f1.Text = "2 Paint Event";
            f1.BackColor=Color.White;
            f1.Paint += new PaintEventHandler(f1_Paint1);
            f1.Paint += new PaintEventHandler(f1_Paint2);
            Application.Run(f1);
        }

        static void f1_Paint1(Object sender, PaintEventArgs pea)
        {
            Form f = (Form)sender;
            Graphics g = pea.Graphics;
            g.DrawString("Paint 1 Event ", f.Font, Brushes.Black, 0, 0);
        }
        static void f1_Paint2(Object sender, PaintEventArgs pea)
        {
            Form f = (Form)sender;
            Graphics g = pea.Graphics;
            g.DrawString("Paint 2 Event ", f.Font, Brushes.Black, 0, 100);
        }
```

Kế thừa Form

```
class Myform:System.Windows.Forms.Form
    {

        public Myform()
        {
            Text = "My Form Class";
        }
    }
```

```
static void Main(string[] args)
        {
            Myform f=new Myform();
            Application.Run(f);
        }
```

Kế thừa Form

```
class Myform:System.Windows.Forms.Form
   {

      public Myform()
      {
         Text = "My Form Class";
      }
      protected override void OnPaint(PaintEventArgs pea)
      {
         Graphics g=pea.Graphics;
         g.DrawString("Hello World", Font, Brushes.Red,20,20);
      }
   }
```

Kế thừa Form

```
class Program
   {
      static void Main(string[] args)
      {
         Myform f=new Myform();
         f.Text = "Ke thua tu " + f.Text;
         f.Paint+=new PaintEventHandler(f_Paint);
         Application.Run(f);
      }
      static void f_Paint(Object sender, PaintEventArgs pea)
      {
         Form form = (Form)sender;
         Graphics g = pea.Graphics;
         g.DrawString("New Hello World", form.Font, Brushes.Red, 50, 50);

      }
   }
```

Kế thừa Form

```
class Myform:System.Windows.Forms.Form
   {

      public Myform()
      {
         Text = "My Form Class";
      }
      protected override void OnPaint(PaintEventArgs pea)
      {
         base.OnPaint(pea);
         Graphics g=pea.Graphics;
         g.DrawString("Hello World", Font, Brushes.Red,20,20);
     }
   }
```

MessageBox.Show

MessageBox.Show (String strText)
MessageBox.Show (String strText, String strCaption)
MessageBox.Show (String strTex, String strCaption, MessageBoxButtons mbb)
MessageBox.Show (String strTex, String strCaption, MessageBoxButtons mbb, MessageBoxIcon mbi)
MessageBox.Show (String strTex, String strCaption, MessageBoxButtons mbb, MessageBoxIcon mbi, MessageBoxDefaultButton mbdb)
MessageBox.Show (String strTex, String strCaption, MessageBoxButtons mbb, MessageBoxIcon mbi, MessageBoxDefaultButton mbdb, MessageBoxOptions mbo)

MessageBox Buttons

Member	Value
Ok	0
OkCancel	1
AbortRetryIgnore	2
YesNoCancel	3
YesNo	4
RetryCancel	5

MessageBox Icon

Member	Value
None	0x00
Hand	0x10
Stop	0x10
Error	0x10
Question	0x20
Exclamation	0x30
Warning	0x30
Asterisk	0x40
Information	0x40

Form Controls

- Là đơn vị cơ sở để tạo nên giao diện người dùng trong lập trình WinForm.
- Là bất kỳ đối tượng nào nằm trong vùng chứa của Container có khả năng tương tác với người sử dụng.
- Là đối tượng dùng để nhận dữ liệu được nhập vào hoặc xuất dữ liệu trên window form
- Các control có các đặc điểm, các phương thức và các sự kiện riêng cho control đó

Thuộc tính chung

Properties
BackColor
CanFocus
Enabled
ForeColor
Name
Text
Visible

Các lớp cơ sở

- ***System.Windows.Forms.Control*** -chứa chức năng cơ bản của thao tác xử lý bàn phím và nhập từ chuột và xử lý tin nhắn window.
- ***System.Windows.Forms.ButtonBase*** - Lớp này hỗ trợ chức năng cơ bản của một nút
- ***System.Windows.Forms.TextBoxBase*** - cung cấp chức năng và thuộc tính thông thường cho các lớp thừa hưởng. Cả hai lớp TextBox và RichTextBox sử dụng chức năng cung cấp bởi TextBoxBase.
- ***System.Windows.Forms.ScrollableControl*** - quản lý sự phát sinh và hiển thị của các thanh cuộn đến người dùng để truy cập đến gốc của một hiển thị.
- ***System.Windows.Forms.ContainerControl*** - Lớp này quản lý chức năng yêu cầu cho một control để hành động
- ***System.Windows.Forms.Panel*** - có thể chứa các control thêm vào, nhưng khác với lớp *ContainerControl*, nó phân loại các control một cách đơn giản.
- ***System.Windows.Forms.Form*** - Tạo bất kỳ loại cửa sổ nào: standard, toolbox, borderless, modal dialog boxes và multi-document interfaces.
- ***System.Windows.Forms.UserControl*** - tạo một custom control đến việc được dùng trong một nơi phức tạp trong một ứng dụng hay tổ chức

STANDARD CONTROL

- Một đối tượng control kế thừa trực tiếp/ gián tiếp từ System.Windows.Forms.Control
- Có các loại:
  - Action control: Button, Toolbar, MenuBar, ContextMenu
  - Value control: Label, TextBox, PictureBox
  - List control: ListBox, ComboBox, DataGrid, TreeView,
  - Container control: GroupBox, Panel, ImageList, …
  - Dialogs:OpenFileDialog, SaveFileDialog, PrintDialog, etc

Buttons

Control	Mô tả
Button	Normal button for actions (e.g., OK or Cancel)
CheckBox	Yes/no selection button
RadioButton	Single selection from a range of choices

Time and date

Control	Mô tả
DateTimePicker	UI for specifying a date or time
MonthCalendar	UI showing a single calendar month

Labels and pictures

Control	Mô tả
GroupBox	Visual grouping for sets of related controls
Label	Text label, usually providing a name or description for some other control (e.g., a text box)
PictureBox	A picture: supports various bitmap formats (BMP, ICO, JPEG, TIFF, and PNG) and Windows metafiles
LinkLabel	Hyperlink, e.g., a URL; this effectively combines label-like and button-like behavior

Text editing

Control	Mô tả
TextBox	An editable text field (plain text only)
RichTextBox	An editable text fields supporting text with formatting (based on RTF—the Rich Text Format)
NumericUpDown	A text box containing a number, and an associated pair of up/down buttons (often known as a spin control)
DomainUpDown	Similar to a NumericUpDown, only the text box can contain any string; the up and down buttons move through a list of strings

Lists and data

Control	Mô tả
ListBox	A vertical list of selectable text items (items may also have images)
ComboBox	An editable text field with an associated drop-down list of selectable items
ListView	A list of selectable items similar to the contents of a Windows Explorer window; supports Large Icon, Small Icon, List and Details views
TreeView	A hierarchical display, similar to that used in the Folders pane of Windows Explorer
PropertyGrid	A UI for editing properties on some object; very similar to the Properties panels in Visual Studio .NET
DataGrid	A grid control showing the contents of a DataSet

Position and progress bars

Control	Mô tả
HScrollBar	A horizontal Windows scrollbar
VScrollBar	A vertical Windows scrollbar
TrackBar	A UI for selecting from a linear range of values (useful for continuous ranges such as percentages)
ProgressBar	A bar indicating what proportion of a long-running task has completed

Layout

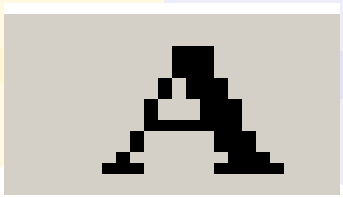
Control	Mô tả
TabControl	Allows multiple similarly sized dialogs to share a single window, with card index style tabs selecting between them—similar to those used on Properties pages in Windows Explorer
Splitter	A bar dividing two parts of a window either vertically or horizontally, allowing the proportion of space given to the two parts to be modified—similar to the divider between the Folders pane and the main pane of a Windows Explorer window
StatusBar	A bar along the bottom of the window providing textual information appropriate to the application, and a window resizing grip (most Windows applications have these)
ToolBar	A bar containing shortcut buttons to frequently used UI operations (most Windows applications have these)

Label

PROPERTIES

Image

TabStop



Methods

Hide

Show

Update

Events

Paint

TextBox

PROPERTIES

AcceptReturn

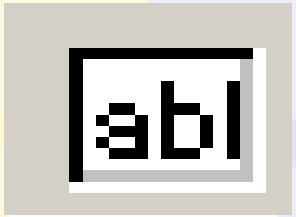
ReadOnly

Passwordchar

MaxLength

Multiline

ScrollBars



Methods

AppendText

Clear

Paste

Cut

Copy

Events

TextChanged

MultilineChanged

Button

PROPERTIES

DialogResult

TextAlign

Methods

PerformClick

Events

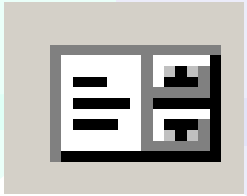
Click

ListBox control

- **ListBox control được dùng để hiển thị danh sách các phần tử.**
- **Người dùng có thể chọn một hay nhiều phần tử từ list.**
- **Bạn có thể thêm phần tử mới vào list thông qua cửa sổ property editor hoặc là thông qua mã chương trình lúc chạy.**
- **Các thuộc tính thường gặp:**
  - SelectionMode
  - Sorted
  - SelectedIndex
  - SelectedItem

ListBox [1]

PROPERTIES



Items

MultiColumn

SelectedIndex

SelectedItem

SelectedItems

Sorted

SelectedValue

Text

ListBox [2]

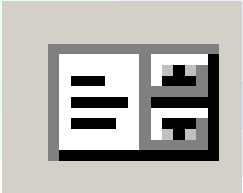
Methods

ClearSelected

GetSelected

FindString

SetSelected



Events

SelectedIndexChanged

SelectedValueChanged

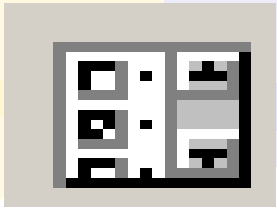
CheckedListBox

PROPERTIES

CheckedIndices

ThreeDCheckBoxes

CheckedItems



Methods

SetItemChecked

GetItemChecked

GetItemCheckState

SetItemCheckState

Events

ItemCheck

ComboBox control

- **Dùng để hiển thị danh sách các phần tử, tuy nhiên ComboBox hiển thị các danh sách này theo kiểu drop – down.**
- **ComboBox có cho phép người dùng nhập dữ liệu vào.**
- **Các phần tử trong ComboBox có thể được thêm vào thông qua property editor hoặc mã chương trình lúc chạy.**
- **Một số các thuộc tính thông dụng:**
  - Text
  - Sorted
  - SelectedIndex
  - SelectedItem

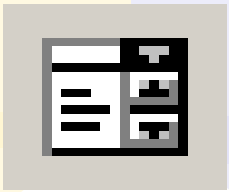
ComboBox

PROPERTIES

DropDownStyle

Focused

MaxDropDownItems



Methods

Select

SelectAll

Events

DropDown

CheckBox control

- **CheckBox control dùng để hiển thị Yes/No hay đúng/sai.**
- **Các thuộc tính thường dùng:**
  - Text
  - Checked
- **CheckBox control cho phép người dùng chọn nhiều hơn 1 lựa chọn**

RadioButton control

- **Dùng để cho người dùng chọn một lựa chọn.**
- **Trong một nhóm, chỉ có một RadioButton được chọn.**
- **Các thuộc tính thường được sử dụng:**
  - Text
  - Checked

Thêm control

```
static void Main(string[] args)
        {
            Form f1 = new Form();
            Button b = new Button();
            b.Text = "OK";
            b.Click+=new EventHandler(b_Click);
            b.Location = new Point(10, 10);
            Button b1 = new Button();
            b1.Text = "Exit";
            b1.Click += new EventHandler(b1_Click);
            b1.Location= new Point(b.Left, b.Height + b.Top + 10);
            f1.Controls.Add(b);
            f1.Controls.Add(b1);
            f1.Text = "2 Paint Event";
            f1.BackColor=Color.White;
            f1.AcceptButton = b;
            f1.CancelButton = b1;
            Application.Run(f1);
        }
```

Thêm control

```
static void b_Click(Object sender, EventArgs e)
    {
        MessageBox.Show("Hello World");
    }
   static void b1_Click(Object sender, EventArgs e)
   {
      Application.Exit();
   }
```