

Chương 15: Quản lý Windows Forms

-----oOo-----

Nội dung thảo luận:

- Thêm vào một form mới cho chương trình
- Thay đổi vị trí của form trên màn hình Windows Desktop
- Thêm một điều khiển vào form khi chương trình đang chạy
- Thay đổi canh lề của các đối tượng trên form
- Chỉ định đối tượng khởi động chương trình

Trong chương này chúng ta sẽ học cách thêm nhiều form vào dự án để xử lý nhập, xuất và hiển thị các thông điệp đặc biệt. Ta cũng sử dụng thuộc tính `DesktopBounds` để định vị trí thay đổi kích thước form, thêm vào các thành phần điều khiển khi chương trình đang chạy...

Chú ý:

- Trong VB.NET muốn đặt thuộc tính cho form thứ hai trong dự án cần phải có tham chiếu đến thể hiện (instance) của biến form đó
- Có thể đặt và định lại vị trí, kích thước form lúc chương trình đang chạy bằng cách sử dụng cửa sổ `Form Layout`. Tuy nhiên bạn cũng có thể sử dụng thuộc tính `DesktopBound` mới do VB.NET không còn hỗ trợ `Form Layout`.
- Thuộc tính mới `Anchor` cho phép xác định kích thước giới hạn tối đa và tối thiểu mà người dùng được phép thay đổi lên form. Thuộc tính `Dock` cho phép Form hay đối tượng có thể neo vào một cạnh cửa sổ hay form khác.
- Trong VB.NET form MDI cha chỉ là một form bình thường có thuộc tính `IsMdiContainer` đặt là `TRUE`. Các form con có thuộc tính `MdiParent` trỏ đến tên của form MDI cha.

1. Thêm một Form mới vào chương trình

Ta có thể thêm rất nhiều form vào chương trình VB.NET. Mỗi form thêm vào được coi là một đối tượng kế thừa từ lớp `System.Windows.Forms.Form`. Các form thêm vào có thứ tự lần lượt và tên tương ứng là `Form1.vb`, `Form2.vb`,...Bạn có thể thay đổi tên mặc định bằng cách chỉ định tên lúc `Add NewItem` hay tại cửa sổ `Solution Explorer`.

Cách sử dụng form:

Bạn có thể cho tất cả các form trong chương trình hiển thị cùng lúc hay chỉ hiển thị khi cần thiết. Khi cho hiển thị lớn hơn một form thì bạn có thể kiểm soát thứ tự form hay cho người dùng hoán chuyển giữa các form.

2. Làm việc với các dự án có nhiều form

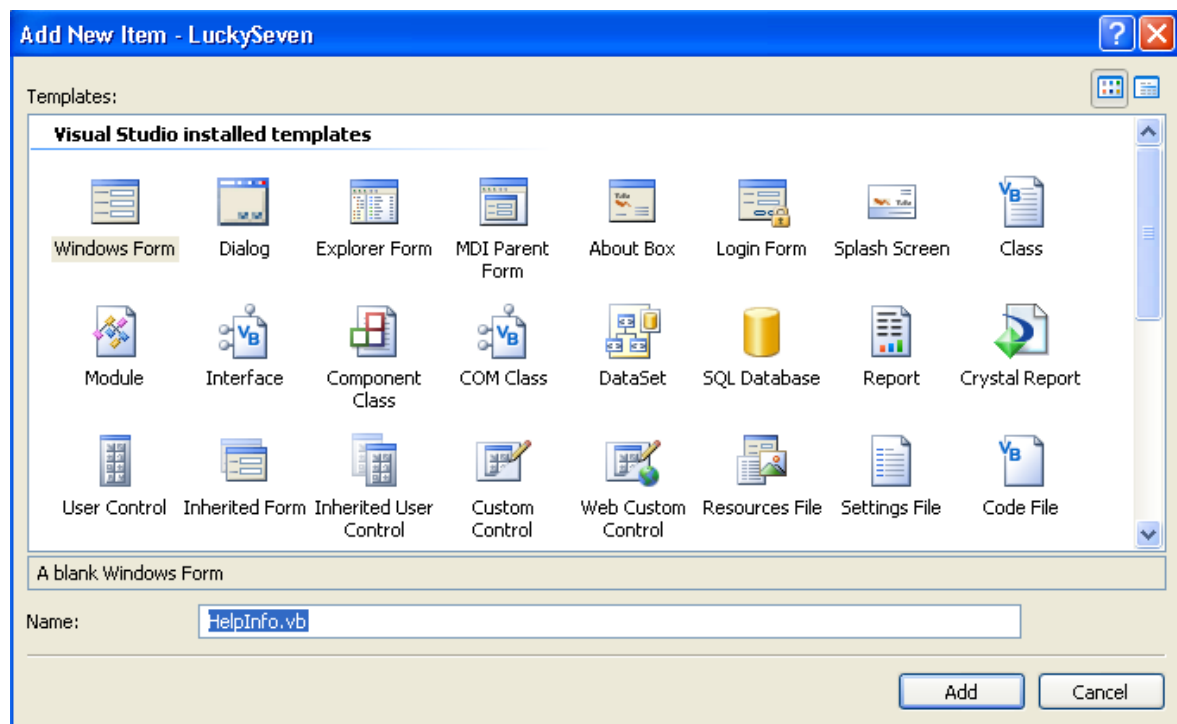
Bài tập sau đây chúng ta sẽ sử dụng một dự án với hai form. Bạn sao chép bài Luckyseven trong chương 10 vào thư mục bài tập của chương 15. Chúng ta sẽ thêm một form thể hiện trợ giúp cho chương trình.

2.1. Thêm form vào dự án

Bạn khởi động giải pháp Luckyseven ta vừa sao chép.

Nhấp đôi vào form1.vb trong cửa sổ Solution Explorer để hiển thị form chính.

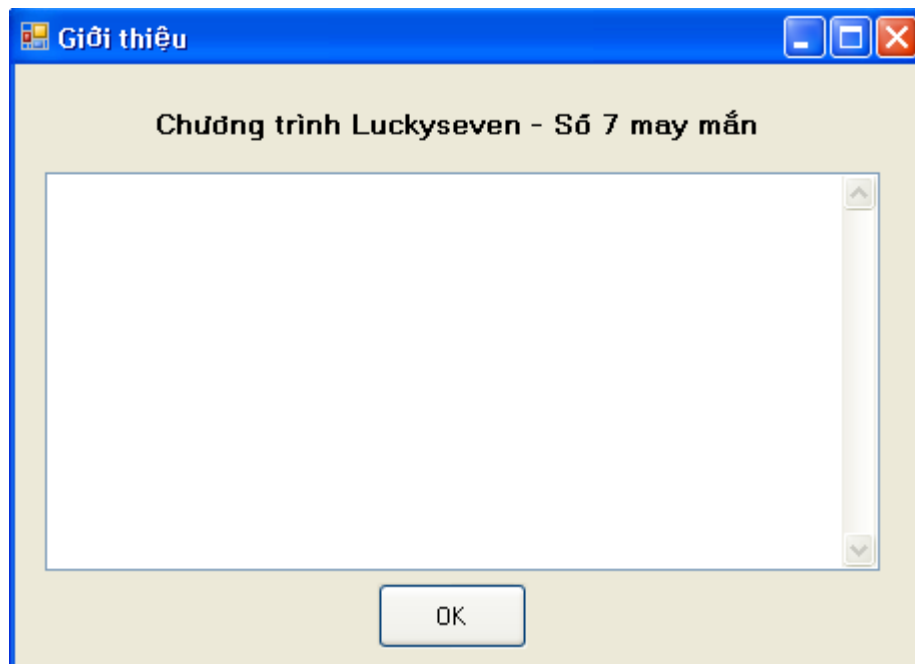
Bạn R-Click vào dự án Luckyseven và chọn Add | New Item. Bạn chọn thêm vào một Windows Form và gõ tên là HelpInfo.vb:



Thêm một số điều khiển vào trong form mới này:

- Thêm vào một nhãn đặt thuộc tính text là “Chương trình Luckyseven – Số 7 may mắn”
- Thêm một TextBox ngay dưới nhãn, thuộc tính MultiLine là True, Scrollbar là Both
- Thêm một nút Button1, thuộc tính Text là OK

Giao diện như hình:



Viết mã:


Form này ta sẽ thể hiện nội dung file Readme.txt trong chương trước chúng ta đã biết. Trước hết ta dùng lớp StreamReader để đọc thông tin của file text và gán cho thuộc tính Text của TextBox1.

Trước hết, khai báo sử dụng lớp này ở đầu form:

```
Imports System.IO
```

Sau đó tạo sự kiện form HelpInfo_Load bằng cách nhấp đôi chuột vào form hay chọn từ danh sách thả xuống như đã biết. Chúng ta nhập đoạn mã sau:

```
Dim StreamToDisplay As StreamReader
StreamToDisplay = New StreamReader _
("D:\Data\Studying\VS.Net 05\Tung buoc lap trinh vb.net\" & _
"Tung buoc lap trinh vb.net\15_Chapter15\Bai tap\LuckySeven\" & _
"LuckySeven\Readme.txt")
TextBox1.Text = StreamToDisplay.ReadToEnd
StreamToDisplay.Close()
```

Việc dùng lớp StreamReader để điền nội dung một file văn bản vào textbox chúng ta đã biết trong chương học về xử lý file text và chuỗi. Ở đây thay vì gõ đường dẫn của file Readme.txt chúng ta có thể kéo thả nó từ trong dự án của mình. Để kéo thả thì file đó phải hiện lên trong cửa sổ Solution Explorer. Muốn nó hiện lên thì bạn có thể chép nó vào thư mục chứa dự án, trở về cửa sổ Solution Explorer nhấp vào nút Refresh  hay copy trực tiếp vào cửa sổ Solution Explorer.

Tạo thủ tục Button1_Click để người dùng click vào nút OK thì đóng form trợ giúp:

```
Me.DialogResult = Windows.Forms.DialogResult.OK
```

Bây giờ làm thế nào để hiển thị form thứ hai này vì dự án của chúng ta có tới hai form?

2.2. Hiển thị Form thứ hai sử dụng thủ tục sự kiện

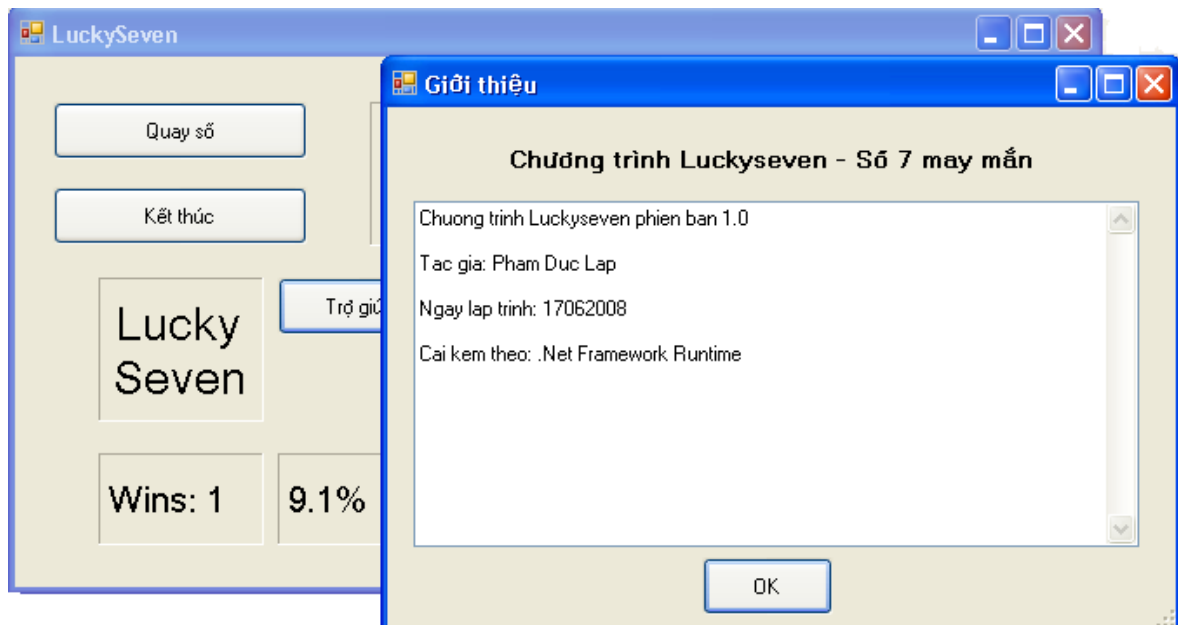
Để làm được điều này ta sẽ thêm một nút ở form thứ nhất Form1.vb và thêm thủ tục triệu gọi form thứ hai.

Bạn mở form1.vb và thêm vào nút nhấn đặt thuộc tính Text cho nó là “Trợ giúp”, thuộc tính name là btnHelp.

Tạo thủ tục btnHelp_Click bằng cách double click vào nút Trợ giúp và nhập đoạn mã sau:

```
Dim frmtrugiup As New HelpInfo()  
frmtrugiup.ShowDialog()
```

Hai phát biểu này cho phép triệu gọi form thứ hai. Như đã nói trước, để tham chiếu đến form thứ hai bạn cần tường minh form đó. Ở đây chúng ta khai báo biến *frmtrugiup* có kiểu *HelpInfo* nhờ phát biểu *New HelpInfo()*. Sau khi đã khởi tạo chúng ta có thể hiển thị form bằng cách gọi đến phương thức *ShowDialog()*. Nếu ở đây bạn gọi form này bằng phương thức *Show()* thì trong thủ tục *Button1_Click* của form *HelpInfo* bạn cần gọi phương thức *Me.Close* thay cho phương thức *DialogResult.OK* chúng ta đã dùng. Bạn chạy chương trình bằng cách ấn F5 và ấn nút Trợ giúp để hiển thị form thứ hai:



3. Định vị form trên màn hình Desktop

Bạn có thể định vị form trên màn hình desktop khi nó xuất hiện bằng thuộc tính DesktopBounds. Nó cho phép định vị trí của form với góc phải dưới và góc trái trên. Đơn vị tính là pixel.

Ngoài ra bạn còn có thể sử dụng thuộc tính StartPosition với các đặc điểm: Manual – bằng tay, CenterScreen – giữa màn hình, WindowsDefaultLocation – vị trí mặc định, WindowsDefaultBound – kích thước mặc định.

3.1. Sử dụng thuộc tính StartPosition

Bây giờ chúng ta sẽ dùng thuộc tính StartPosition và DestopBounds để định vị trí form qua bài tập *MyDesktopBound* sau đây.

Bạn tạo mới giải pháp và thêm vào một dự án cùng tên *MyDesktopBound* và làm như sau:

- Mở properties của form1.vb.
- Thay thuộc tính StartPosition thành CenterScreen và chạy thử. Form sẽ xuất hiện ở chính giữa màn hình.
- Đóng chương trình, đặt thuộc tính StarPosition thành Manual. Với thuộc tính này bạn cần đặt lại thuộc tính Location, ta đặt thuộc tính này là 100, 50.
- Chạy thử chương trình. Form sẽ hiển thị theo tọa độ ta đã đặt.

3.2. Sử dụng thuộc tính DestopBounds

Đặt thêm nút nhấn lên form1, đặt text là “Tạo form mới”.

Tạo thủ tục Button1_Click và nhập mã như sau:

```
'Tạo form thứ hai có tên Form2
Dim form2 As New Form()
'Định nghĩa thuộc tính Text và đường viền cho form
form2.Text = "Form mới"
form2.FormBorderStyle = Windows.Forms.FormBorderStyle.FixedDialog
'Chỉ định vị trí của form được đặt thủ công
form2.StartPosition = FormStartPosition.Manual
'Khởi tạo cấu trúc Rectangle nắm giữ kích thước mới
'Góc trái trên (200,100)
'Chiều dài và cao (300,250)
Dim rectangle_form2 As New Rectangle(200, 100, 300, 250)
'Định kích thước của form sử dụng đối tượng rectangle trên
form2.DesktopBounds = rectangle_form2
'Hiển thị form
form2.ShowDialog()
```

Bạn chạy chương trình này bằng cách ấn F5. Nhấn vào nút “tạo form mới” để tạo form thứ hai. Form này có vị trí như ta đã định. Form này không cho phép bạn kéo lại kích

thước như các form trước đây do ta đã đặt thuộc tính `FormBorderStyle` của form thành `FixedDialog`.

3.3. Phóng to, thu nhỏ và khôi phục lại cửa sổ chương trình

Ngoài ra bạn cũng có thể phóng to, thu nhỏ hay khôi phục lại vị trí mặc định của form. Bạn có thể thực hiện điều này khi thiết kế hay khi chương trình đang chạy.

Để làm điều này trước hết bạn cần cho hiện hai nút `Maximize` và `minimize` ở góc phải trên chương trình bằng hai thuộc tính:

```
MaximizeBox = True
MinimizeBox = True
```

Tiếp đến trong mã chương trình hay trong cửa sổ thuộc tính bạn đặt thuộc tính `WindowState` như sau:

```
WindowState = FormWindowState.Minimized
```

Nếu bạn muốn kiểm soát kích thước phóng to, thu nhỏ cho phép của form bạn đặt thuộc tính `MinimumSize`, `MaximumSize`. Hai thuộc tính này có kiểu cấu trúc `Size` giống như cấu trúc `Rectangle`, ví dụ:

```
Dim FormSize As New Size(400, 300)
MaximumSize = FormSize
```

4. Thêm vào các điều khiển lúc form đang chạy

Ta thường đưa các điều khiển trên `Toolbox` khi thiết kế form. Bạn cũng có thể đưa chúng vào trong form khi chương trình đang chạy – tạo điều khiển động. Quy trình để đưa như sau:

Khai báo biến đối tượng có kiểu lớp của phần tử giao diện mà bạn muốn đưa vào, ví dụ:

```
Dim btnOK As New Button()
```

Thiết lập thuộc tính cho các nút nhấn sau khi đã khai báo như trên:

```
'Đặt thuộc tính cho nút nhấn
btnOK.Text = "OK"
btnOK.Location = New Point(110, 100)
```

Đưa đối tượng vào form. Để thực hiện điều này, bạn đưa các đối tượng vào tập hợp `Controls` của form bằng phương thức `Add`:

```
form2.Controls.Add(btnOK)
```

Bài tập *MyAddControls* sau đây sẽ minh họa cụ thể hơn:

Bạn tạo một giải pháp mới và thêm vào một dự án có cùng tên như trên. Thiết kế form1 có một nút nhấn với thuộc tính `text` là “Hiển thị ngày”. Khi người dùng click vào đây thì

một form mới sẽ được tạo ra. Khi form này tạo ra thì đồng thời mã chương trình sẽ tạo hai điều khiển là nhãn lblNgày ghi ngày hiện hành và nút nhấn btnOK để đóng form thứ hai này lại.

Bạn tạo thủ tục Button1_click và nhập mã như sau:

```
'Khai báo form và các đối tượng điều khiển
Dim form2 As New Form()
Dim lblNgày As New Label()
Dim btnOK As New Button()

'Đặt thuộc tính nhãn
lblNgày.Text = "Hôm nay là: " & DateTime.Now.ToString("dd-MM-yyyy")
lblNgày.Size = New Size(150, 50)
lblNgày.Location = New Point(80, 50)

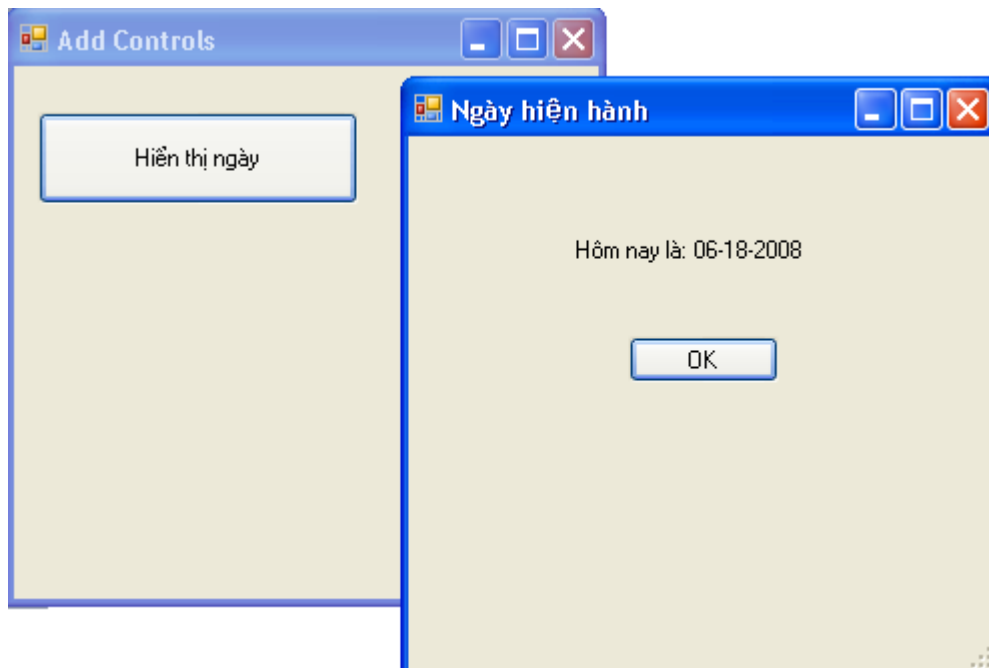
'Đặt thuộc tính cho nút nhấn
btnOK.Text = "OK"
btnOK.Location = New Point(110, 100)

'Đặt thuộc tính cho form mới
form2.Text = "Ngày hiện hành"
form2.CancelButton = btnOK
form2.StartPosition = FormStartPosition.CenterScreen

'Đưa các đối tượng mới vào tập hợp Controls
form2.Controls.Add(lblNgày)
form2.Controls.Add(btnOK)

'Gọi hiển thị form2
form2.ShowDialog()
```

Chạy chương trình và chúng ta sẽ thấy hiệu quả.



5. Tổ chức sắp xếp các điều khiển trên form

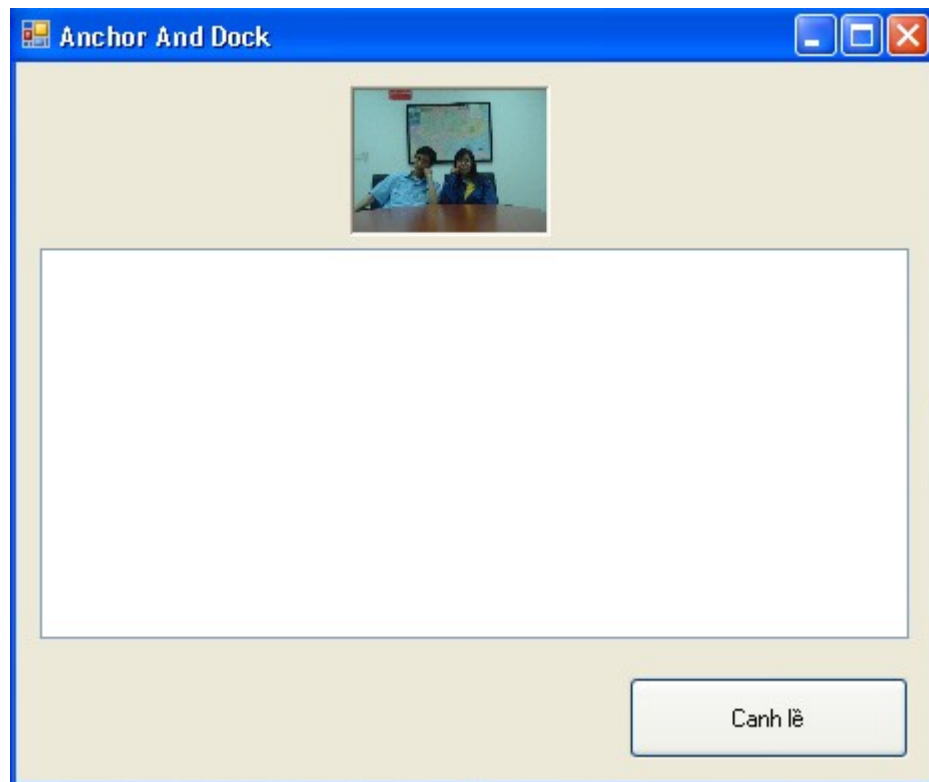
Việc thêm các điều khiển bằng mã chương trình gặp khó khăn trong việc căn chỉnh vị trí các đối tượng do không có công cụ nhìn trực quan. Chúng ta chỉ có thể định kích thước và vị trí thông qua hai thuộc tính là Size và Location. Để khắc phục điều này, VB.NET cung cấp một số thuộc tính mới như **Anchor** – định phạm vi ràng buộc tương đối giữa các đối tượng, **Dock** – neo dính đối tượng này vào cạnh một đối tượng khác. Chúng ta sẽ làm quen với hai thuộc tính này trong bài tập **MyAnchorAndDock** sau đây:

Tìm hiểu chương trình:

Chương trình gồm một PictureBox cho hiển thị một ảnh, một TextBox và một nút nhấn. Khi người dùng click vào nút này thì tiến hành định vị các điều khiển trong form.

Thiết kế giao diện:

Giao diện chính của form như hình:



Thuộc tính của các đối tượng:

- PictureBox1: Image – các bạn có thể cho một ảnh bất kỳ nào (dung lượng nhỏ thôi) để hiển thị; sizemode – StretchImage.
- Button1: Text – “Canh lề”.

Viết mã:

Bạn tạo thủ tục Button1_Click bằng cách double click vào nút “Canh lề” và nhập đoạn mã sau:

```
PictureBox1.Dock = DockStyle.Top
TextBox1.Anchor = AnchorStyles.Bottom Or _
AnchorStyles.Left Or _
AnchorStyles.Right Or AnchorStyles.Top
Button1.Anchor = AnchorStyles.Bottom Or AnchorStyles.Right
```

Chạy chương trình:

Ấn F5 để chạy chương trình. Khi ấn canh lề thì ảnh sẽ được canh lề theo mép trên của form. Bạn có thể kéo form rộng ra theo ý muốn và quan sát. Phóng to form ra thì thấy vị trí các đối tượng trên form cũng không thay đổi vị trí.

6. Chỉ định thủ tục hay đối tượng thực thi chương trình khởi động

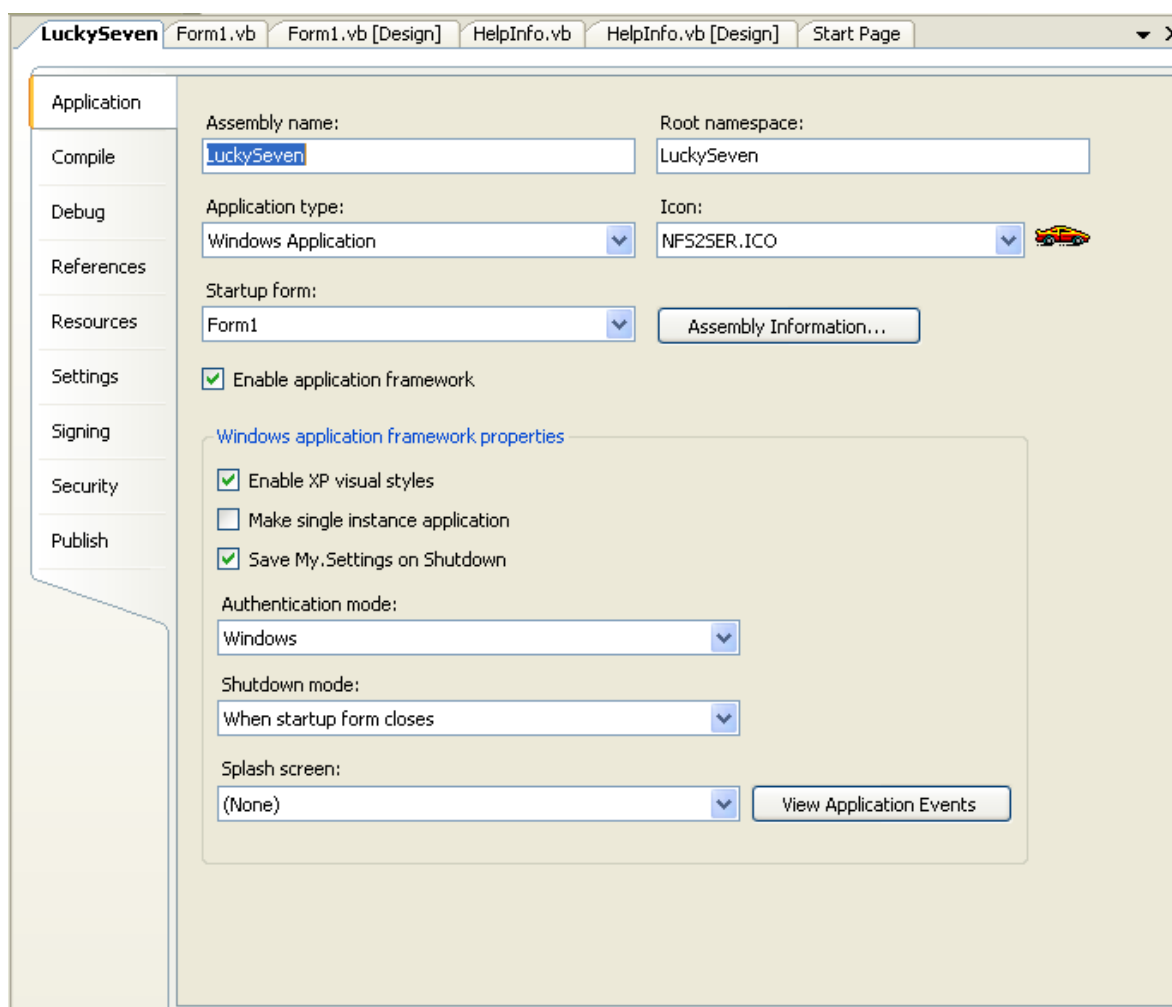
Khi dự án có nhiều form bạn sẽ phải chỉ định xem form nào sẽ khởi động trước form nào.

Bạn có thể làm điều này nhờ hộp thoại Properties của dự án hay yêu cầu VB thực thi thủ tục mang tên Sub Main, trong thủ tục này bạn có trách nhiệm tạo và hiển thị form khác.

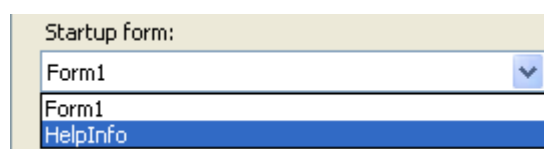
6.1. Thay đổi form khởi động

Bạn mở lại Solution Luckyseven của chương này chúng ta vừa thao tác. Ta thấy dự án Luckyseven có hai form là Form1.vb và HelpInfo.vb. Bây giờ chúng ta sẽ chỉ định xem form nào sẽ khởi động trước.

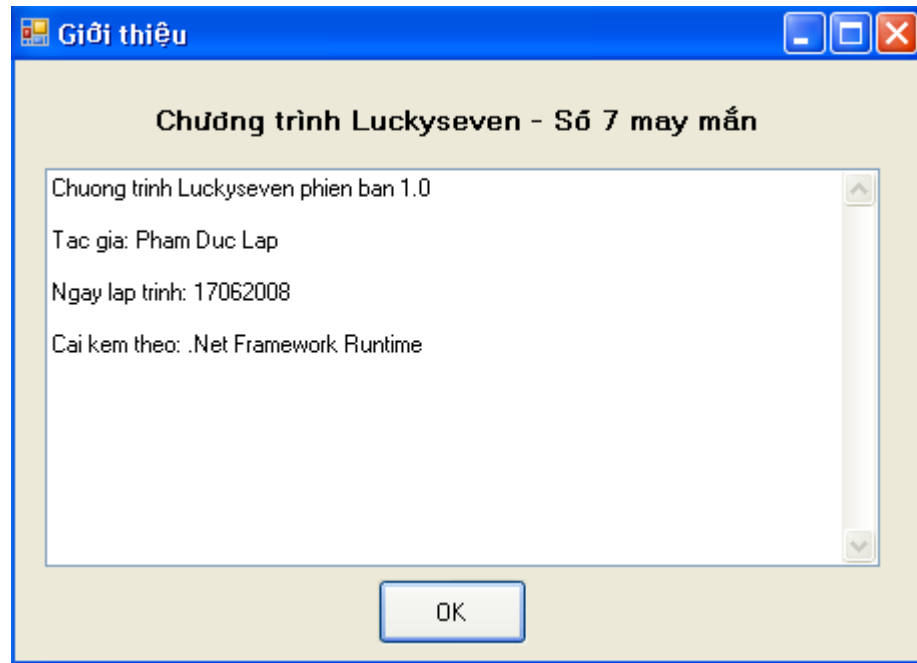
Bạn R-Click vào tên dự án Luckyseven và chọn Properties. Cửa sổ thuộc tính Properties của dự án xuất hiện như hình:



Hộp thoại này cho phép bạn tùy chỉnh lại một số thiết lập cho dự án. Để thay đổi thứ tự form khởi động, bạn dùng combobox Startup Form trong tab Application như hình. Bạn hãy chọn form HelpInfo thay vì form1 và chạy thử chương trình.



Lúc này form khởi động không phải là form1 mà là form HelpInfo:



Đóng chương trình và chọn lại form1 trong danh sách Startup Form và chạy lại chương trình một lần nữa. Khi đó form1 sẽ khởi động trước tiên.

6.2. Thực thi chương trình từ thủ tục Sub Main

Bây giờ thay vì yêu cầu chương trình hiển thị form HelpInfo trước chúng, ta sẽ yêu cầu chương trình thực thi thủ tục Sub Main. Thủ tục này thường được khai báo trong Module.

Bạn R-Click vào dự án LuckySeven và chọn Add | New Item và thêm vào một module có tên SubMainModule.

Bạn nhập vào khai báo như sau:

```
Public MyForm1 As New Form1()
Public MyForm2 As New HelpInfo()

Public Sub Main()
    MsgBox("Đây là Sub Main")
    'Có thể đặt thêm các mã khởi tạo tại đây
    'trước khi hiển thị Form chính
    ...

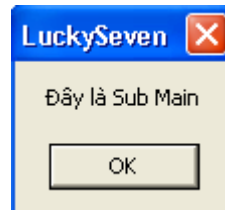
    'Hiển thị Form chính
    MyForm1.ShowDialog()
End Sub
```

Khi bạn thêm thủ tục Sub Main vào dự án, bạn phải đặt nó trong module và khai báo thủ tục này có phạm vi toàn cục Public.

Bây giờ chúng ta cần cho chương trình gọi đến thủ tục này trước tiên. Bạn cũng mở trang Properties của dự án LuckySeven như đã làm. Muốn để sub Main khởi động thì

bạn lại phải chọn lại kiểu của ứng dụng trong danh sách Application Type. Ta có thể chọn là Console Application hay Windows Service. Trong trường hợp này là **Windows Service**, sau đó chọn Sub Main trong danh sách Start Object.

Bây giờ bạn chạy chương trình và sẽ thấy thủ tục Sub Main được triệu gọi trước tiên. Nó đưa ra thông báo “Đây là thủ tục Sub Main” và gọi đến form1 sau khi người dùng ấn OK của hộp thoại:



7. Tổng kết chương 15

Chúng ta đã hoàn thành chương 15 – chương viết về form và quản lý form. Như các chương trước các bạn tự mình tổng kết những gì đã học.