



## CHƯƠNG 4

---

# Menu ngữ cảnh và dải trạng thái

- 4.1 Dải menu ngữ cảnh 93
- 4.2 Sự kiện thả xuống và đối số sự kiện 99
- 4.3 Dải trạng thái 108 4.4 Tóm tắt 115

Chúng ta đã bắt đầu chương 3 với một chuyến tham quan nhanh qua một số lớp Windows Forms và thảo luận về các đối tượng menu nói chung, đồng thời xem các menu là một phần của chức năng ToolStrip do .NET cung cấp như thế nào. Trong chương này, chúng tôi mở rộng cuộc thảo luận này để đề cập đến menu ngữ cảnh, cũng như cách chia sẻ menu giữa menu ngữ cảnh và thanh menu.

Lớp StatusStrip là một loại dải công cụ khác, thường xuất hiện ở cuối biểu mẫu để hiển thị các phản hồi khác nhau cho người dùng. Chương này cũng xem xét các thanh trạng thái trong Windows Forms.

Hình 4.1 cho thấy ứng dụng của chúng ta khi nó xuất hiện ở phần cuối của cuộc thảo luận này. Ngoài các menu của chúng tôi, bạn có thể thấy rằng thanh trạng thái chứa hai khu vực, được gọi là bảng hoặc nhãn. Bạn có thể đặt bất kỳ số lượng bảng nào trên thanh trạng thái và hiển thị cả thông tin văn bản và thông tin đồ họa trong mỗi bảng.

Tiếp tục cách tiếp cận hướng dẫn của chúng tôi, chương này giả định rằng bạn có sẵn giải pháp MyPhotos từ phần 3.4 làm điểm bắt đầu cho cuộc thảo luận của chúng ta. Bạn cũng có thể tải xuống mã này từ trang web của cuốn sách.



Hình 4.1

Thanh trạng thái của chúng tôi bao gồm đồ họa tay cầm định cỡ tùy chọn ở phía dưới bên phải của điều khiển, cho phép người dùng thay đổi kích thước biểu mẫu.

Trước khi chuyển sang các dải trạng thái, chúng ta cần kết thúc cuộc thảo luận về menu của mình. Chúng tôi bắt đầu với lớp ContextMenuStrip .

## 4.1 DẢI MENU LIÊN QUAN

Mặc dù việc tạo các menu ngữ cảnh đòi hỏi một số nỗ lực bổ sung của người lập trình, nhưng các menu như vậy cải thiện đáng kể khả năng sử dụng của một giao diện và cần được xem xét nghiêm túc trong bất kỳ ứng dụng nào. Khả năng người dùng bấm chuột phải vào một điều khiển và xem ngay danh sách các lệnh là một cơ chế mạnh mẽ mà người dùng có kinh nghiệm đặc biệt đánh giá cao. Menu ngữ cảnh thường được liên kết với một điều khiển đồ họa cụ thể, nhưng cũng có thể được hiển thị theo chương trình. Do đó, menu ngữ cảnh cung cấp khả năng truy cập nhanh vào các lệnh liên quan ngay đến những gì người dùng hiện đang cố gắng thực hiện hoặc hiểu.

Hầu hết các điều khiển trong không gian tên System.Windows.Forms đều hỗ trợ thuộc tính ContextMenuStrip được kế thừa từ lớp Điều khiển để chỉ định một đối tượng MenuStrip ngữ cảnh để liên kết với điều khiển.<sup>1</sup> Cài đặt này có thể được thay đổi động để cho phép các menu ngữ cảnh khác nhau hiển thị tùy thuộc vào trạng thái của kiểm soát.

Trong phần này, chúng ta thảo luận chung về các menu ngữ cảnh và thêm một menu vào điều khiển Picture Box của chúng ta . Như thể hiện trong hình 4.2, điều này tạo lối tắt cho người dùng muốn thay đổi cách hiển thị hình ảnh. Chúng tôi bắt đầu bằng cách thêm một menu ngữ cảnh vào ứng dụng của mình và điền nội dung của nó.

---


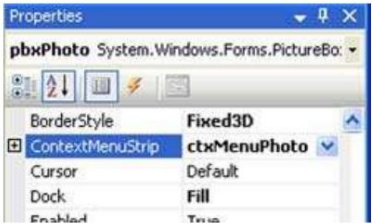
<sup>1</sup> Tất cả các điều khiển cũng hỗ trợ thuộc tính ContextMenu liên kết trong lớp ContextMenu dựa trên Win32 . Chúng được hỗ trợ đầy đủ, nhưng không xuất hiện trong Visual Studio theo mặc định để không khuyến khích sử dụng các lớp dựa trên Win32.



Hình 4.2 Ứng dụng của chúng ta vô hiệu hóa menu con Hình ảnh khi không có hình ảnh nào được tải và đánh dấu chế độ hiển thị hiện tại bất cứ khi nào menu con được hiển thị.

4.1.1 Tạo menu ngữ cảnh Chúng ta

bắt đầu bằng cách thêm một menu ngữ cảnh mới vào ứng dụng của mình và liên kết nó với điều khiển pbxPhoto . Các lớp đằng sau menu ngữ cảnh trong .NET được thảo luận sau những thay đổi này. Phần tiếp theo giải thích cách điền menu này với một số mục menu.

THÊM MENU BỐI CẢNH		
	Hoạt động	Kết quả
1	<p>Thêm đối tượng ContextMenuStrip đã gọi ctxMenuPhoto vào biểu mẫu trong cửa sổ MainForm.cs [Design].</p> <p>Cách kéo</p> <p>mục ContextMenuStrip từ nhóm Menus &amp; Toolbars trong cửa sổ Hộp công cụ vào biểu mẫu và đặt (Tên) thành "ctxMenuPhoto".</p>	<div></div> <p>Trong phương thức InitializeComponent đã tạo, menu ngữ cảnh mới được xác định và khởi tạo.</p> <pre>riêng System.Windows.Forms. ContextMenuStrip ctxMenuPhoto;</pre>
2	<p>Liên kết menu ngữ cảnh mới này với điều khiển PictureBox.</p> <p>Làm thế nào để</p> <p>một. Hiển thị các thuộc tính cho điều khiển pbxPhoto .</p> <p>b. Bấm vào bên phải của Mục nhập ContextMenuStrip.</p> <p>c. Nhấp vào mũi tên xuống.</p> <p>d. Chọn mục ctxMenuPhoto từ danh sách.</p>	<div></div> <p>Trong phương thức InitializeComponent đã tạo , menu ngữ cảnh đã chọn được gán cho điều khiển hộp hình ảnh.</p> <pre>không trống riêng từ InitializeComponent() {     . . .     this.pbxPhoto.ContextMenuStrip =         this.ctxMenuPhoto;</pre>

Lớp ContextMenuStrip thực chất là một thùng chứa các đối tượng ToolStripItem xuất hiện trong trình đơn. Như chúng ta đã thấy trong hình 3.5, lớp này bắt nguồn từ lớp ToolStripDropDownMenu , lớp này lại dựa trên lớp ToolStrip DropDown . Mặc dù chức năng cốt lõi cho tất cả các điều khiển này đến từ các lớp Control và ToolStrip , nhưng tất nhiên, phần lớn chức năng thả xuống được xác định bởi lớp ToolStripDropDown . Các thành viên chính của lớp này được hiển thị trong .NET Bảng 4.1.

.NET Bảng 4.1 Lớp ToolStripDropDown

Tính năng mới trong 2.0 Lớp ToolStripDropDown là một dải công cụ hiển thị một tập hợp các mục từ một mục dải công cụ khác-ví dụ: danh sách thả xuống xuất hiện khi nhấp vào Nút ToolStrip DropDown . Lớp này là một phần của không gian tên System.Windows.Forms và kế thừa từ lớp ToolStrip . Xem .NET Table 16.1 để biết các thành viên kế thừa từ lớp ToolStrip .


Lớp ToolStripDropDownMenu kế thừa từ ToolStripDropDown và là lớp cơ sở cho lớp ContextMenuStrip .

Công cộng Đặc tính	Tự động đóng	Nhận hoặc đặt xem trình đơn thả xuống có tự động đóng khi mất tiêu điểm hay không.
	CanOverflow (được ghi đè từ ToolStrip)	Nhận hoặc đặt xem các mục có bị tràn hay không. Giá trị mặc định cho dải thả xuống là sai.
	Chủ sở hữuItem	Nhận hoặc đặt ToolStripItem sở hữu dải thả xuống này.
Công cộng phương pháp	Đóng	Đóng dải thả xuống, tùy ý cung cấp lý do dưới dạng giá trị ToolStripDropDownClose Reason .
	Trình diễn	Hiển thị dải thả xuống ở tọa độ đã chỉ định.
Công cộng Sự kiện	đã đóng	Xảy ra sau khi dải thả xuống đã đóng.
	Đóng cửa	Xảy ra ngay trước khi dải thả xuống đóng lại.
	đã mở	Xảy ra sau khi dải thả xuống được hiển thị.
	Khai mạc	Xảy ra trước khi dải thả xuống được hiển thị.

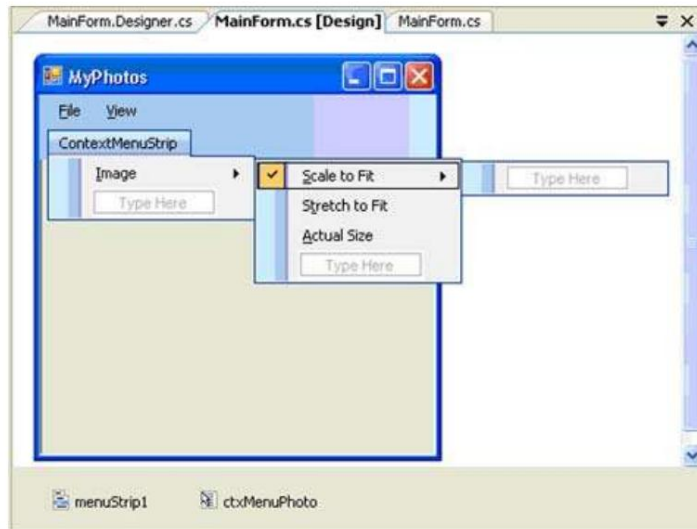
4.1.2 Thêm các mục vào menu ngữ cảnh Bây giờ chúng ta

đã sẵn sàng để thêm các mục vào menu ngữ cảnh của mình, như minh họa trong hình 4.3. Trình đơn ngữ cảnh của chúng tôi sẽ xác định ba cài đặt hiển thị khác nhau cho một hình ảnh, tương ứng với ba giá trị liệt kê PictureBoxSizeMode khác nhau . Người dùng có thể sử dụng menu này để thay đổi cách hiển thị hình ảnh trong điều khiển PictureBox . Việc tạo các mục menu ngữ cảnh trong Visual Studio cũng giống như những gì chúng ta đã thấy đối với các dải menu trong chương 3: nhấp vào đối tượng ctxMenuView trong cửa sổ trình thiết kế để hiển thị thông báo “Type Here” và nhập các menu mới. Vì điều này đã quen thuộc với chúng tôi, bảng sau đây chỉ tóm tắt các thay đổi cần thiết để thêm một số mục menu vào dải menu ngữ cảnh của chúng tôi.

PHÂN TÍCH DẢI MENU CONTEXT

	Hoạt động	Kết quả												
1	<p>Thêm một mục menu Hình ảnh được gọi là "menuImage" vào menu ngữ cảnh.</p>	<div></div> <p>Một đối tượng ToolStripMenuItem mới được gọi là menuImage được thêm vào mã nguồn MainForm.Designer.cs và được định nghĩa là một mục thả xuống cho</p> <p>ctxMenuMenu Ảnh .</p> <pre>. . . this.ctxMenuPhoto.Items.AddRange( new System.Windows.Forms .ToolStripItem[] { this.menuImage } );</pre>												
2	<p>Thêm mục menu con "Scale to Fit" vào Menu hình ảnh và gán các thuộc tính của nó.</p> <p>Làm thế nào để</p> <p>Nhập mục này vào bên phải Hình ảnh thực đơn.</p> <table><tr><th colspan="2">Cài đặt</th></tr><tr><th>Tài sản</th><th>Giá trị</th></tr><tr><td>(Tên)</td><td>menuImageScale</td></tr><tr><td>Đã kiểm tra</td><td>ĐÚNG VẬY</td></tr><tr><td>Chữ</td><td>&amp;Mở rộng để phù hợp với</td></tr></table>	Cài đặt		Tài sản	Giá trị	(Tên)	menuImageScale	Đã kiểm tra	ĐÚNG VẬY	Chữ	&Mở rộng để phù hợp với	<p>Menu mới xuất hiện trong Visual Studio như trong hình 4.3. Mục menuImageScale được tạo và khởi tạo trong mã nguồn được tạo và được xác định là mục thả xuống cho menuImageScale .</p> <pre>// // menuImage //  this.menuImage.DropDownItems.AddRange( new System.Windows.Forms. ToolStripItem[] { this.menuImageScale}); . . .</pre>		
Cài đặt														
Tài sản	Giá trị													
(Tên)	menuImageScale													
Đã kiểm tra	ĐÚNG VẬY													
Chữ	&Mở rộng để phù hợp với													
3	<p>Tương tự, thêm phần "Kéo dài để vừa vặn" và "Thực tế Kích thước" menu dưới dạng menu con của Hình ảnh thực đơn.</p> <table><tr><th colspan="3">Cài đặt</th></tr><tr><th>Thuộc tính</th><th>thực đơn</th><th>Giá trị</th></tr><tr><td>Kéo dài để phù hợp</td><td>(Tên) menuImageKéo dài Chữ</td><td>&amp;Co giãn cho vừa vặn</td></tr><tr><td>Thực sự kích thước</td><td>(Tên) menuHình ảnhThực tế Chữ</td><td>&amp;Kích thước thực sự</td></tr></table>	Cài đặt			Thuộc tính	thực đơn	Giá trị	Kéo dài để phù hợp	(Tên) menuImageKéo dài Chữ	&Co giãn cho vừa vặn	Thực sự kích thước	(Tên) menuHình ảnhThực tế Chữ	&Kích thước thực sự	<p>Các menu mới xuất hiện trong Visual Studio và được phản ánh trong mã nguồn được tạo.</p> <p>Cụ thể, một mảng các đối tượng ToolStripItem được tạo và định nghĩa là các mục thả xuống cho menuImage menu.</p> <pre>this.menuImage.DropDownItems.AddRange( new System.Windows.Forms. ToolStripItem[] { this.menuImageScale, this.menuImageStretch, this.menuImageActual});</pre>
Cài đặt														
Thuộc tính	thực đơn	Giá trị												
Kéo dài để phù hợp	(Tên) menuImageKéo dài Chữ	&Co giãn cho vừa vặn												
Thực sự kích thước	(Tên) menuHình ảnhThực tế Chữ	&Kích thước thực sự												

Mã được tạo trong MainForm.Designer.cs cho menu ngữ cảnh của chúng tôi tương tự như mã mà chúng tôi đã kiểm tra trong chương 3, vì vậy chúng tôi không xem xét lại mã ở đây. Nhận ra rằng các menu ở đây đều có cùng một loại điều khiển—một đối tượng ToolStripMenuItem—bất kể chúng xuất hiện ở đâu trong giao diện. Mục menu Hình ảnh, cũng như ba mục menu con, đều là các đối tượng thuộc loại ToolStripMenuItem.



Hình 4.3 “Type Here” trong trình chỉnh sửa menu cho biết nơi có thể thêm các mục mới và khu vực phía trước mỗi mục chứa một hình ảnh hoặc dấu kiểm tùy chọn.

Trước khi chúng tôi thực sự xử lý menu ngữ cảnh của mình, sẽ rất hữu ích nếu các mục menu ngữ cảnh có thể truy cập được từ dải menu trong ứng dụng của chúng tôi. Phần tiếp theo cho biết cách chia sẻ các mục này để chúng xuất hiện trong cả menu Chế độ xem mới và dải menu ngữ cảnh của chúng tôi.

#### 4.1.3

Chia sẻ menu ngữ cảnh Mặc dù

chắc chắn có thể tạo lại menu Hình ảnh trong dải menu của chúng tôi, nhưng đây dường như không phải là giải pháp tinh tế nhất. Có menu này ở hai nơi sẽ yêu cầu chúng tôi cập nhật một menu khi chúng tôi thực hiện bất kỳ thay đổi nào đối với menu kia. Lý tưởng nhất là chúng tôi muốn có một menu duy nhất và tìm cách chia sẻ menu này giữa hai đối tượng dải.

Giải pháp là triển khai các menu trong menu ngữ cảnh của chúng tôi, sau đó sử dụng lại các menu này làm trình đơn thả xuống cho menu Chế độ xem mới trong ứng dụng. Điều này có thể thực hiện được vì cả menu ngữ cảnh và các lớp mục menu đều có thể chứa một bộ sưu tập menu con. Lớp ContextMenuStrip dựa trên lớp ToolStripDropDown và xác định thuộc tính Mục để chứa một tập hợp các phiên bản ToolStripItem ; đối tượng ToolStripMenuItem dựa trên lớp ToolStripDropDownItem , lớp này xác định một thuộc tính DropDown chứa một thể hiện ToolStripDropDown .

Như chúng ta sẽ thấy, điều này cho phép chúng ta chỉ định một menu ngữ cảnh làm trình đơn thả xuống cho một mục menu. Đoạn mã sau minh họa khả năng này:

```
// Tạo một dải menu với một mục cấp cao nhất MenuStrip ms = new MenuStrip();
ToolStripMenuItem menuTop = new ToolStripMenuItem("Top"); ms.Items.Add(menuTop);
```

```
// Tạo menu ngữ cảnh với ba mục menu
```

```
ContextMenuStrip ctxMenu= new ContextMenuStrip();
ctxMenu.Items.Add("Mục 1");
ctxMenu.Items.Add("Mục 2");
ctxMenu.Items.Add("Mục 3");

// Gán menu ngữ cảnh dưới dạng menu thả xuống cho menu mục cấp cao nhấtTop.DropDown = ctxMenu;
```

Tổng quan về lớp ToolStripDropDownItem được hiển thị trong .NET Bảng 4.2. Như chúng ta đã thấy trong chương 3 với menu Tập, các mục menu sử dụng thuộc tính DropDownItems để xác định các mục riêng lẻ trong menu. Thuộc tính DropDown được sử dụng trong đoạn mã trước cho phép một đối tượng ToolStripDropDown hiện có được chỉ định làm danh sách thả xuống.

.NET Bảng 4.2 Lớp ToolStripDropDownItem

Tính năng mới trong 2.0 Lớp ToolStripDropDownItem là một mục đại công cụ hiển thị một tập hợp các mục thả xuống khi được nhấp. Lớp này là cơ sở cho mục nút thả xuống, mục menu và các mục đại công cụ khác có chức năng thả xuống.

Lớp ToolStripDropDownItem là một phần của không gian tên System.Windows.Forms và kế thừa từ lớp ToolStripItem . Các thành viên kế thừa từ ToolStripItem được hiển thị trong .NET Bảng 3.4.

Công cộng Đặc tính	Thả xuống	Nhận hoặc đặt đối tượng ToolStripDropDown được hiển thị khi mục được nhấp vào
	DropDownCác mặt hàng	Nhận bộ sưu tập các mục trong Lớp ToolStripDropDown được liên kết với mục thả xuống này
	ThảXuốngHướng	Nhận hoặc đặt giá trị liệt kê ToolStripDropDownDirection cho biết nơi điều khiển thả xuống được hiển thị so với điều khiển gốc của nó
	ĐãThả XuốngCác Mặt Hàng	Nhận xem danh sách thả xuống có chứa bất kỳ mục nào không
Công cộng phương pháp	ẨnDropDown	Ẩn danh sách thả xuống được liên kết với mục
	Hiển thịThả xuống	Hiển thị điều khiển ToolStripDropDown được liên kết với mục này
Công cộng Sự kiện	Thả XuốngĐóng	Xảy ra khi điều khiển ToolStripDropDown được liên kết với mục này đã đóng
	DropDownItemClicked	Xảy ra khi một mục trong liên kết Điều khiển ToolStripDropDown đã được nhấp
	Thả XuốngMở	Xảy ra khi điều khiển ToolStripDropDown được liên kết với mục này đã mở
	ThảXuốngMở	Xảy ra khi điều khiển ToolStripDropDown được liên kết với mục này sắp mở

Lớp ToolStripDropDownItem cũng cung cấp các thuộc tính để cho biết danh sách thả xuống được hiển thị ở đâu và liệu nó có chứa bất kỳ mục nào hay không. Nó cũng định nghĩa các phương thức để hiển thị hoặc ẩn danh sách và các sự kiện kích hoạt khi danh sách được mở và đóng. Trong ứng dụng của chúng ta, chúng ta cần tạo menu View để chứa nội dung của menu con text. Như vừa thảo luận, chúng ta có thể gán thuộc tính DropDown của menu này cho menu ngữ cảnh của mình. Các bước sau đây minh họa quy trình này.

GÁN DÃI MENU BỐI CẢNH LÀM MENU XEM THẢ XUỐNG										
	Hoạt động	Kết quả								
1	<p>Trong cửa sổ MainForm.cs [Thiết kế], hãy thêm menu Chế độ xem cấp cao nhất ở bên phải menu Tập hiện có của chúng tôi.</p> <table><tr><th colspan="2">Cài đặt</th></tr><tr><td>Tài sản</td><td>Giá trị</td></tr><tr><td>(Tên)</td><td>thực đơnXem</td></tr><tr><td>Chữ</td><td>&amp;Lượt xem</td></tr></table>	Cài đặt		Tài sản	Giá trị	(Tên)	thực đơnXem	Chữ	&Lượt xem	<pre>private void InitializeComponent() {      . . .     this.menuStrip1.Items.AddRange(         Hệ thống mới...ToolStripItem[] { this.menuFile,         this.menuView});      . . .      // //     menuView //     this.menuView.Name = "menuView";     . . . }      . . .     riêng System.Windows.         Forms.ToolStripMenuItem menuView;</pre>
Cài đặt										
Tài sản	Giá trị									
(Tên)	thực đơnXem									
Chữ	&Lượt xem									
2	<p>Trong hàm tạo MainForm, hãy thêm một dòng gán thuộc tính DropDown cho menu Chế độ xem cho menu ngữ cảnh ctxMenuPhoto .</p>	<pre>công khai MainForm() {     Khởi tạo Thành phần();     SetTitleBar(); menuView.DropDown     = ctxMenuPhoto; }</pre>								

Thay đổi này cho phép cả bấm chuột phải vào hộp ảnh và bấm vào menu Chế độ xem mới để hiển thị cùng một menu. Biên dịch và chạy mã này để xem điều này đang hoạt động.

Với các menu của chúng tôi được xác định, chúng tôi đã sẵn sàng cho một số trình xử lý sự kiện cho Hình ảnh menu để thay đổi cách hình ảnh hiển thị dựa trên cài đặt đã chọn.

4.2 SỰ KIẾN THẢ XUỐNG VÀ BIỆN LUẬN SỰ KIẾN

Các mục menu Tải và Thoát trong menu Tập của chúng tôi khá đơn giản khi các menu di chuyển. Mỗi mục sẽ tạo ra một sự kiện Nhấp chuột khi được chọn và trình xử lý sự kiện được liên kết sẽ thực hiện hành động thích hợp. Trong menu Hình ảnh của chúng tôi, chúng tôi cần thay đổi cài đặt PictureBox.SizeMode bất cứ khi nào một mục menu con Hình ảnh được chọn. Một giải pháp rõ ràng là xử lý sự kiện Nhấp chuột được liên kết với từng mục menu con và giải thích rõ ràng là thay đổi thuộc tính SizeMode thành cài đặt thích hợp.

Mặc dù điều này sẽ hoạt động tốt, nhưng đó không phải là cách tiếp cận của chúng tôi. Thay vào đó, chúng tôi muốn xử lý cả ba mục menu con trong một trình xử lý sự kiện. Có một vài lý do cho việc này. Đầu tiên, tất nhiên, nó cung cấp một ví dụ thay thế cho cuốn sách hay này



bạn đang đọc. Quan trọng hơn, nó gói gọn logic cho các menu này trong một trình xử lý đơn lẻ. Nếu sau này chúng ta cần thực hiện thay đổi hoặc thêm một mục menu con mới, chúng ta chỉ cần sửa đổi một trình xử lý này.

Một tính năng khác mà chúng tôi muốn thêm vào là khả năng kiểm tra mục menu con được liên kết với cài đặt hiển thị hiện tại. Sự kiện Nhấp chuột cho menu Hình ảnh vẫn được nâng lên, vì vậy chúng tôi có thể xử lý thao tác nhấp chuột vào menu Hình ảnh khi menu con được hiển thị và cập nhật các menu này cho phù hợp.<sup>2</sup> Về mặt khái niệm, người dùng đang tìm cách hiển thị nội dung menu con chứ không phải nhấp chuột menu chính, vì vậy sự kiện Nhấp chuột không phải là mô hình phù hợp cho mục đích này.

Menu Hình ảnh của chúng tôi là menu chính cho các mục menu Tỷ lệ để vừa, Kéo dài để vừa và Kích thước thực tế. Khi người dùng nhấp vào menu chính như vậy, họ sẽ thấy menu con được liên kết với menu. Lớp `ToolStripMenuItem` hỗ trợ các sự kiện thả xuống khác nhau trước, trong và sau khi menu con như vậy được hiển thị. Điều này cho phép trình xử lý sự kiện sửa đổi nội dung hoặc hình thức của menu con theo yêu cầu của ứng dụng. Một ví dụ về khái niệm này có thể được tìm thấy trong hệ điều hành Windows. Mở cửa sổ `My Computer` và hiển thị menu `File`. Nội dung của menu này thay đổi tùy thuộc vào loại tệp hiện được chọn.

Đối với dải menu `Windows Forms`, các sự kiện thả xuống mục menu được kế thừa từ lớp `ToolStripDropDownItem`, được hiển thị trong `.NET Table 4.2`. Các sự kiện thả xuống được kích hoạt khi người dùng nhấp vào menu để hiển thị bộ sưu tập các mục trong đối tượng thả xuống được liên kết.

Trong phần này, chúng tôi minh họa các sự kiện `DropDownItemClicked` và `DropDownOpen` để thay đổi hình thức và hành vi của các mục menu con Hình ảnh. Các sự kiện thả xuống khác được xử lý theo cách tương tự.

#### 4.2.1 Xử lý nhấp chuột vào mục menu con Menu

con cho mục menu Hình ảnh bật lên bất cứ khi nào nhấp vào menu Hình ảnh.

Khi người dùng nhấp vào một mục menu con Hình ảnh, chúng tôi muốn hiển thị hình ảnh thay đổi tương ứng. Để thực hiện việc này, chúng tôi sẽ gán thuộc tính `SizeMode` của điều khiển `PictureBox` tùy thuộc vào menu nào được chọn. Các giá trị `SizeMode` được lấy từ phép liệt kê `PictureBoxSizeMode`, như trong `.NET Table 4.3`.

Như đã đề cập trong phần giới thiệu, chúng tôi sử dụng các sự kiện thả xuống để quản lý hành vi của menu con Hình ảnh. Các sự kiện thả xuống hoạt động tốt trong những trường hợp như thế này khi menu con chứa một tập hợp các giá trị hoặc các mục liên quan khác được áp dụng theo cách tương tự. Các trình xử lý này tận dụng thực tế là menu Xem và menu ngữ cảnh của chúng tôi có chung các mục menu.

Để tạo điều kiện thuận lợi cho hành vi tuyệt vời này, chúng tôi bắt đầu bằng cách sử dụng thuộc tính `Thẻ` trong các menu của mình để ghi lại giá trị `SizeMode` mong muốn cho mỗi menu.

---

<sup>2</sup> Đây là một thay đổi trong hành vi từ các menu dựa trên Win32 được gói gọn trong các lớp `Menu` và `MenuItem`. Trong các lớp này, sự kiện `Click` chỉ được kích hoạt nếu menu không chứa menu con.

XÁC ĐỊNH CÀI ĐẶT SIZEMODE CHO MENU CON HÌNH ẢNH												
	Hoạt động	Kết quả										
1 Từ	<div>MainForm.cs [Thiết kế] giành chiến thắng dow, chỉ định thuộc tính Thẻ cho mỗi mục trong menu con Hình ảnh để chứa giá trị liệt kê PictureBoxSizeMode mong muốn .</div> <div><table><tr><th colspan="2">Cài đặt</th></tr><tr><th>Mục menu</th><th>Thuộc tính thẻ</th></tr><tr><td>Mở rộng để phù hợp với</td><td>Phóng</td></tr><tr><td>Kéo dài để phù hợp</td><td>Kéo dài hình ảnh</td></tr><tr><td>Kích thước thực sự</td><td>Bình thường</td></tr></table></div>	Cài đặt		Mục menu	Thuộc tính thẻ	Mở rộng để phù hợp với	Phóng	Kéo dài để phù hợp	Kéo dài hình ảnh	Kích thước thực sự	Bình thường	<div>Các cài đặt được chỉ định trong phương thức InitializeComponent của tệp thiết kế đã tạo.</div> <div><pre>this.menuImageScale.Tag = "Thu phóng"; . . . this.menuImageStretch.Tag =     "StretchImage"; . . . this.menuImageActual.Tag = "Bình thường";</pre></div>
Cài đặt												
Mục menu	Thuộc tính thẻ											
Mở rộng để phù hợp với	Phóng											
Kéo dài để phù hợp	Kéo dài hình ảnh											
Kích thước thực sự	Bình thường											

.NET Bảng 4.3 liệt kê PictureBoxSizeMode

Phép liệt kê PictureBoxSizeMode chỉ định các chế độ hiển thị có thể có cho điều khiển PictureBox và được sử dụng với thuộc tính PictureBox.SizeMode . Bảng liệt kê này là một phần của không gian tên System.Windows.Forms .

liệt kê giá trị	Kích thước tự động	Kích thước của điều khiển PictureBox tự động điều chỉnh theo kích thước của hình ảnh chứa trong đó.
	Hình ảnh trung tâm	Hình ảnh được căn giữa trong điều khiển PictureBox , cắt các cạnh bên ngoài của hình ảnh nếu cần.
	Bình thường	Hình ảnh được đặt ở góc trên bên trái của điều khiển PictureBox , cắt bớt bên phải và dưới cùng của hình ảnh nếu cần.
	Kéo dài hình ảnh	Hình ảnh được kéo dài hoặc thu nhỏ để vừa với điều khiển PictureBox .
	Phóng	Hình ảnh được chia tỷ lệ để vừa với điều khiển PictureBox , sao cho tỷ lệ khung hình của hình ảnh được giữ nguyên.

Thuộc tính Thẻ có sẵn trong hầu hết các thành phần Windows Forms, bao gồm các đối tượng ToolStripItem và cho phép hầu hết mọi thứ được liên kết với thành phần đó. Lớp Control xác định thuộc tính này cho tất cả các điều khiển Windows Forms. Thuộc tính thuộc loại đối tượng, vì vậy bất kỳ lớp, cấu trúc hoặc giá trị nào khác đều có thể được gán. Trong mã của chúng tôi, chúng tôi chỉ định giá trị chuỗi của cài đặt liệt kê mà chúng tôi muốn liên kết với từng mục.

Với các giá trị này được đặt, chúng tôi đã sẵn sàng để xử lý các sự kiện. Sự kiện DropDownItem Clicked xảy ra khi một mục con được nhấp vào. Như đã đề cập trong chương 1, tất cả các trình xử lý sự kiện .NET đều sử dụng một bộ đối số chung. Tham số đầu tiên là đối tượng gửi sự kiện, trong khi tham số thứ hai là đối tượng đối số sự kiện.

Đối với các sự kiện cơ bản như Nhấp chuột, như chúng ta đã thấy, lớp EventArgs là trình giữ chỗ cho tham số thứ hai này.

Đối với các sự kiện phức tạp hơn, bao gồm sự kiện DropDownItemClicked , cần có thêm thông tin chi tiết để xử lý sự kiện đúng cách. Trong những trường hợp như vậy, một lớp mới được lấy từ lớp EventArgs cơ sở để giữ thông tin này. Hãy xem điều này hoạt động như thế nào bằng cách tiếp tục các thay đổi của chúng ta.

THÊM CÁC TRÌNH XỬ LÝ SỰ KIỆN THẢ XUỐNG CHO MENU CON HÌNH ẢNH		
	Hoạt động	Kết quả
2 Thêm một	Trình xử lý sự kiện DropDownItemClicked vào Hình ảnh thực đơn.  Làm thế nào để  Hiển thị các sự kiện cho mục menu Hình ảnh và nhấp đúp vào sự kiện DropDown ItemClicked .	Một phương thức mới được đăng ký cho đối tượng menuImage trong tệp thiết kế đã tạo.  this.menuImage.DropDownItemClicked += new System.Windows.Forms .ToolStripItemClickedEventHandler(  this.menuImage_DropDownItemClicked);  Cửa sổ mã MainForm.cs được hiển thị với con trỏ ở đầu phương thức mới này.  private void menuImage_DropDownItemClicked( người gửi đối tượng,  ToolStripItemClickedEventArgs e) {
3	Trong trình xử lý này, hãy gọi phương thức ProcessImageClick với các đối số đã cho.	ProcessImageClick(e); }
4 Triển khai phương pháp	ProcessImageClick để sửa đổi thuộc tính SizeMode cho điều khiển hộp ảnh dựa trên mục đã chọn.	khoảng trống riêng tư ProcessImageClick( ToolStripItemClickedEventArgs e) { ToolStripItem item = e.ClickedItem; chuỗi enumVal = item.Tag dưới dạng chuỗi; nếu (enumVal != null) {  pbxPhoto.SizeMode = (PictureBoxSizeMode) Enum.Parse(typeof(PictureBoxSizeMode), enumVal);  } }

Có rất nhiều điều đang diễn ra ở đây, vì vậy hãy chia nhỏ điều này để xem chính xác điều gì đang xảy ra. Trước tiên, hãy lưu ý ở bước 2 cách trình xử lý sự kiện được chỉ định cho menu Hình ảnh của chúng tôi. Đối với sự kiện Click , lớp EventHandler cơ sở được sử dụng để định nghĩa trình xử lý này. Các sự kiện xác định một đối số sự kiện khác với lớp EventArgs yêu cầu một lớp trình xử lý sự kiện thay thế. Theo quy ước, trình xử lý sự kiện và lớp đối số sự kiện cho các sự kiện như vậy sử dụng cùng một tên với các hậu tố khác nhau. Đối với sự kiện DropDownItem Clicked , đối số sự kiện là đối tượng ToolStripItemClickedEventArgs và trình xử lý sự kiện là đối tượng ToolStripItemClickedEventHandler .

Liệt kê 4.1 Phương thức ProcessItemClick

```
private void ProcessItemClick(ToolStripItemClickedEventArgs e) {
    ToolStripItem item = e.ClickedItem; chuỗi enumVal =
    item.Tag dưới dạng chuỗi; nếu (enumVal != null) {
        pbxPhoto.SizeMode = (PictureBoxSizeMode)
        Enum.Parse(typeof(PictureBoxSizeMode), enumVal);
    }
}
```

Nhận đối số sự kiện  
ItemClicked

Chuyển đổi thẻ  
Giá trị C thành chuỗi

Chuyển đổi  
chuỗi thành  
Giá trị  
SizeMode

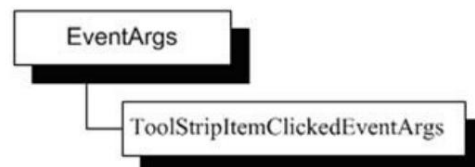
Liệt kê 4.1 hiển thị mã cho phương thức hỗ trợ trình xử lý sự kiện mới của chúng ta. Hãy xem xét các điểm được đánh số trong danh sách này chi tiết hơn:

B Trình xử lý sự kiện DropDownItemClicked nhận tham số người gửi, giống như trình xử lý sự kiện Nhấp chuột của chúng tôi trước đó trong chương.

Nó cũng nhận một số dữ liệu sự kiện dưới dạng một thẻ hiện ToolStripItemClickedEventArgs , bắt nguồn từ lớp EventArgs như trong hình 4.4.

Lớp đối số sự kiện này cung cấp thuộc tính ClickedItem truy xuất đối tượng ToolStripItem được liên kết do người dùng nhấp vào. Mã của

chúng tôi sử dụng thuộc tính này để thay đổi hành vi của nó tùy thuộc vào mục thả xuống nào được nhấp vào.



Hình 4.4 Tất cả các tham số đối số sự kiện kế thừa từ lớp EventArgs cơ sở.

C Khi chúng tôi xác định mục được người dùng nhấp vào, chúng tôi sử dụng thuộc tính Thẻ chứa giá trị chuỗi SizeMode mong muốn. Do thuộc tính Tag là một đối tượng nên chúng ta phải chuyển đổi giá trị của nó thành chuỗi để lấy giá trị được gán. Từ khóa C# as chuyển đổi một biến thành một kiểu tham chiếu nhất định, trong trường hợp này là một đối tượng chuỗi. Nếu biến không thể được chuyển đổi thành loại đã cho, thì null được trả về.<sup>3</sup> Trong ví dụ của chúng ta, chúng ta biết rằng thuộc tính Tag luôn chứa một giá trị chuỗi. Tuy nhiên, cách tốt nhất là kiểm tra null vì bạn không bao giờ biết chương trình có thể phát triển như thế nào theo thời gian.

D .NET Framework chứa một cấu trúc Enum đặc biệt là lớp cơ sở ngầm định cho tất cả các kiểu liệt kê . Lớp này chứa các phương thức tĩnh khác nhau để thao tác các giá trị liệt kê và các chuỗi tương ứng của chúng và được tóm tắt trong .NET

<sup>3</sup> Bạn cũng có thể truyền giá trị trực tiếp vào một chuỗi, sử dụng mã "string enumVal = (string)item.Tag;". Mã này đưa ra một đối tượng InvalidCastException nếu không thể thực hiện truyền, trái ngược với từ khóa as chỉ đơn giản gán giá trị null .

Bảng 4.4. Mã ở đây bao gồm một vài khái niệm mới, vì vậy hãy chia nhỏ dòng này xuống một chút nữa để xem điều gì đang xảy ra.

.NET Bảng 4.4 Cấu trúc enum		
Cấu trúc Enum là lớp cơ sở ngầm định cho tất cả các kiểu liệt kê. Cấu trúc này là một phần của không gian tên Hệ thống và triển khai các giao diện IComparable, IFormattable và IConvertible .		
Công cộng phương pháp	So với	Trả về liệu một đối tượng đã cho có nhỏ hơn, bằng hoặc lớn hơn giá trị liệt kê hay không
	GetTypeCode	Trả về giá trị liệt kê TypeCode đại diện cho Loại giá trị cơ bản cho phiên bản này
	ToString (được ghi đề từ Sự vật)	Trả về biểu diễn chuỗi của giá trị hiện tại
công tính phương pháp	Định dạng	Chuyển đổi một giá trị nhất định cho một kiểu liệt kê nhất định thành một chuỗi, dựa trên định dạng được cung cấp
	Nhận tên	Trả về mảng các hằng số chuỗi cho một kiểu liệt kê đã cho
	Nhận giá trị	Trả về mảng giá trị cho các hằng số trong một kiểu liệt kê đã cho
	Được định nghĩa	Trả về liệu một hằng số với một giá trị nhất định có được xác định trong kiểu liệt kê đã cho hay không
	phân tích cú pháp	Chuyển đổi một hoặc nhiều hằng số chuỗi cho một kiểu liệt kê nhất định thành giá trị liệt kê thích hợp

Chúng tôi sử dụng phương thức Enum.Parse để chuyển đổi giá trị chuỗi thành giá trị liệt kê PictureBoxSizeMode thích hợp . Phương thức Parse được sử dụng ở đây nhận một kiểu liệt kê và một chuỗi, đồng thời trả về một đối tượng đại diện cho giá trị liệt kê. Nó ném ra một đối tượng ArgumentException nếu một chuỗi không hợp lệ được cung cấp.

Kiểu liệt kê thu được bằng cách sử dụng từ khóa typeof . Từ khóa này trả về một đối tượng Loại đại diện cho loại đã cho, trong trường hợp này là kiểu liệt kê PictureBoxSizeMode .

Vì phương thức Parse trả về một giá trị đối tượng nên nó phải được chuyển đổi thành giá trị liệt kê thích hợp. Chuyển đổi này là "xuống" hệ thống phân cấp lớp từ loại đối tượng chung sang kiểu liệt kê PictureBoxSizeMode cụ thể hơn . Trong C++, các hoạt động như vậy rất nguy hiểm vì ngôn ngữ này không cung cấp hỗ trợ rõ ràng cho việc hạ thấp như vậy, như nó được gọi. Trong C#, downcasting được cho phép rõ ràng ted, và một cast không hợp lệ ném ra một ngoại lệ của loại InvalidCastException.

Việc sử dụng sự kiện `DropDownItemClicked` đảm bảo rằng `PictureBox` của chúng tôi được cập nhật với hình vi hiển thị phù hợp, bất kể mục nào được chọn.  
Biên dịch và chạy mã này để xem nó hoạt động.

THỬ NÓ! Phép liệt kê `PictureBoxSizeMode` không chỉ chứa ba cài đặt được sử dụng ở đây. Thêm một mục menu vào menu Hình ảnh có tên là `menuImageCenter`, với nội dung “Hình ảnh Trung tâm,” để xử lý giá trị Trung tâm Hình ảnh. Đặt thuộc tính Thẻ một cách thích hợp để xem mã hiện có của chúng tôi tự động xử lý menu mới như thế nào.

4.2.2 Thay đổi menu con trước khi nó xuất hiện Người

dùng đánh giá cao phản hồi từ một ứng dụng. Giao diện hiện tại của chúng tôi chưa làm được điều này. Người dùng phải hiểu các chế độ hiển thị có thể có để biết những gì hiện đang được chọn, sau đó chọn một cài đặt khác. Một giao diện trực quan hơn sẽ làm nổi bật lựa chọn hiện tại trong menu phụ `menuImage`. Điều này sẽ ngay lập tức cho biết chế độ nào hiện đang được hiển thị và giúp người dùng của chúng tôi đưa ra lựa chọn sáng suốt hơn.

Lớp `ToolStripMenuItem` cung cấp thuộc tính `Checked` đã kiểm tra, khi đúng, sẽ hiển thị dấu kiểm bên cạnh menu. Chúng tôi có thể đặt thuộc tính này một cách rõ ràng khi lựa chọn được sửa đổi, vì vậy người dùng của chúng tôi sẽ thấy phản hồi thích hợp. Tất nhiên, khi chương trình của chúng ta thay đổi, có thể có các lệnh khác hoặc tương tác của người dùng làm thay đổi chế độ hiển thị của hình ảnh, vì vậy phương pháp này có thể trở nên phức tạp. Một giải pháp thay thế có thể đảm bảo rằng các chế độ hiển thị được chọn hoặc bỏ chọn khi chúng được hiển thị cho người dùng. Cách tiếp cận này mạnh mẽ hơn khi đối mặt với những thay đổi trong tương lai, tạo ra một ứng dụng mà người dùng, người lập tài liệu và người thử nghiệm sẽ đánh giá cao trong nhiều năm tới.

Sự kiện `DropDownOpening` được thiết kế cho mục đích này. Sự kiện này xảy ra ngay trước khi danh sách thả xuống được liên kết với một mục được hiển thị, cho phép sửa đổi giao diện hoặc nội dung của mục đó rồi hiển thị ngay cho người dùng.

Hãy xem làm thế nào điều này hoạt động.

THỰC HIỆN MỘT TRÌNH XỬ LÝ DROPDOWNOPENING CHO MENU HÌNH ẢNH		
	Hoạt động	Kết quả
1	Trong cửa sổ <code>MainForm.cs</code> [Thiết kế], hãy thêm trình xử lý sự kiện <code>Mở DropDown</code> cho menu Hình ảnh.	<pre>menu void riêngImage_DropDownOpening(     người gửi đối tượng, EventArgs e) {</pre>
2	Trong trình xử lý này, hãy gọi phương thức <code>ProcessImageOpening</code> , với mục thả xuống đã cho làm đối số.	<pre>    ProcessImageOpening( người         gửi dưới dạng ToolStripDropDownItem); }</pre>

THỰC HIỆN MỘT TRÌNH XỬ LÝ DROPDOWNOPENING CHO MENU HÌNH ẢNH (TIẾP THEO)		
	Hoạt động	Kết quả
3	Triển khai phương thức ProcessImageOpening để gán thuộc tính Đã bật và Đã kiểm tra cho từng mục menu trong danh sách thả xuống đã cho.	<pre>private void ProcessImageOpening(     ToolStripDropDownItem me) {     if (cha != null) {          chuỗi enumVal = pbxPhoto.SizeMode.ToString(); foreach (mục             ToolStripMenuItem trong parent.DropDownItems)              {                 item.Enabled = (pbxPhoto.Image != null); item.Checked =                     item.Tag.Equals(enumVal);             }         }     } }</pre>

Trình xử lý cho sự kiện DropDownOpening tương tự như những gì chúng ta đã thấy đối với sự kiện Nhấp chuột , với một đối tượng làm đối số đầu tiên và một đối tượng EventArgs làm tham số thứ hai. Việc triển khai của chúng tôi gọi phương thức ProcessImageOpening để thực hiện các hành động mong muốn. Chúng ta có thể tránh phương thức này và triển khai logic trực tiếp trong trình xử lý. Sử dụng một phương thức hỗ trợ như chúng ta làm ở đây rất hữu ích nếu bạn muốn gọi chức năng tương tự trong một phần khác của chương trình.

Phương thức ProcessImageOpening đảm bảo mục đã cho không rỗng và sử dụng phương thức ToString để lấy biểu diễn chuỗi của giá trị liệt kê Chế độ kích thước hiện tại . Nó cũng sử dụng từ khóa C# foreach , cung cấp một cách dễ dàng để liệt kê các mục trong một bộ sưu tập. Vòng lặp foreach rất giống vòng lặp for , ngoại trừ mỗi lần lặp lại của vòng lặp xử lý mục tiếp theo trong bộ sưu tập thay vì giá trị dựa trên số nguyên tiếp theo. Chúng ta thảo luận khái niệm này chi tiết hơn trong chương 5.

Đối với mỗi mục menu con trong menu Hình ảnh, mã của chúng tôi đặt thuộc tính Đã bật dựa trên việc điều khiển PictureBox hiện có chứa hình ảnh hay không và thuộc tính Đã kiểm tra dựa trên so sánh chuỗi Chế độ kích thước với giá trị Thẻ của mục con . Các phương thức ToString và Equals được sử dụng ở đây được kế thừa từ lớp đối tượng cơ sở , mà chúng ta cũng sẽ thảo luận trong chương 5.

Lưu ý rằng không có gì trong phương thức ProcessImageOpening để cho biết liệu các mục menu này có phải là một phần của cấu trúc menu cụ thể hay không. Mã này hoạt động đồng nhất cho dù được hiển thị từ menu Xem hay là một phần của menu ngữ cảnh. Biên dịch và chạy ứng dụng để xác minh rằng các menu



Hình 4.5 Chế độ hiển thị Kích thước thực của chúng tôi hiển thị mọi điểm ảnh trong ảnh bên trong cửa sổ, bắt đầu từ góc trên bên trái.

làm việc chính xác. Hình 4.5 hiển thị ứng dụng với hình ảnh được hiển thị ở chế độ Kích thước Thực tế.

Thật không may, con số này cho thấy một vấn đề với điều khiển PictureBox của chúng tôi. Trong hình, hình ảnh lớn hơn vùng hiển thị, nhưng không có cách nào để xem phần còn lại của hình ảnh mà không thay đổi kích thước cửa sổ. Mặc dù điều này có thể xảy ra khi hình ảnh nhỏ, nhưng hình ảnh có độ phân giải cao có thể chứa nhiều pixel hơn màn hình của chúng tôi. Lý tưởng nhất là ứng dụng sẽ hiển thị các thanh cuộn ở đây. Vì điều khiển PictureBox không hỗ trợ các thanh cuộn, nên chúng tôi gặp một chút khó khăn trừ khi chúng tôi hiển thị ảnh ở chế độ hiển thị thay thế.

Bạn có thể thắc mắc về một cuốn sách dạy bạn cách xây dựng một ứng dụng không hoạt động tốt, và bạn nên làm như vậy. Chúng ta thảo luận cách giải quyết vấn đề này trong chương 13 bằng cách sử dụng bảng điều khiển có thể cuộn hoặc trang tab để chứa hộp ảnh và một lần nữa trong chương 19, nơi chúng tôi xây dựng hộp ảnh tùy chỉnh để giải

THỬ NÓ! quyết vấn đề này một cách trực tiếp.\ Được rồi, tôi thừa nhận điều này không liên quan gì đến làm với ứng dụng của chúng tôi. Tuy nhiên, nếu bạn muốn tận hưởng sự kiện DropDownOpening, hãy thêm một menu mới, menu Counter, ở cuối menu ngữ cảnh ctxMenuPhoto với dòng chữ "Counter" và chèn một menu đơn có dòng chữ "DropDown" vào menu con của nó. Xác định một sự kiện DropDownOpening cho menuCounter menu mà Visual Studio sẽ đặt tên là menuCounter\_DropDownOpening. Trong trình xử lý này, tạo động một đối tượng ToolStripMenuItem mới và thêm nó vào cuối menuCounter menu con. Đặt thuộc tính Văn bản cho menu mới của bạn thành "Count #", trong đó # là số lần thả xuống đã xảy ra trên menu mới của bạn. Sử dụng số nguyên tĩnh dropdownCount trong lớp MainForm để theo dõi số lần xuất hiện thả xuống. Các dòng để tự động tạo menu mới trong trình xử lý DropDownOpening của bạn sẽ giống như sau: ToolStripMenuItem

```
mi = new ToolStripMenuItem(); mi.Text = "Count " +
dropdownCount.ToString(); menuCounter.DropDownItems.Add(mi);
```

Ví dụ này minh họa cách dễ dàng tạo menu nhanh chóng với .NET Framework và cách menu chính có thể thay đổi nội dung của menu con khi nó được hiển thị. Điều này có thể được sử dụng, ví dụ, để hiển thị danh sách các tệp được mở gần đây nhất bởi một ứng dụng.

Nếu tất cả những điều này không có ý nghĩa gì với bạn, hãy tải xuống mã này HÃY THỬ ! từ trang web của cuốn sách. Hãy xem menuCounter\_DropDown Trình xử lý mở để xem mã được yêu cầu.

Điều này kết thúc cuộc thảo luận của chúng tôi về các menu bây giờ. Trước khi kết thúc chương này, chúng ta hãy xem xét một loại dải công cụ khác: dải trạng thái.



### 4.3 DẢI TÌNH TRẠNG

Hầu hết các ứng dụng cung cấp rất nhiều thông tin cho người dùng. Thường có một tập hợp con cốt lõi mà hầu hết người dùng sẽ đánh giá cao khi có sẵn. Thanh trạng thái ở dưới cùng của cửa sổ là một vị trí tốt cho loại dữ liệu này, vì nó cung cấp phản hồi nhanh liên quan đến tác vụ hiện tại hoặc vị trí con trỏ. Ví dụ, trình xử lý văn bản của tôi cho biết số trang hiện tại, tổng số trang, vị trí cột và dòng của con trỏ, liệu phím Chèn đã được nhấn hay chưa (dường như tôi nhấn liên tục khi nhắm tới phím Xuống trang), và các thông tin khác mà tôi có thể muốn biết trong nháy mắt. Điều này giúp tôi theo dõi cuốn sách này được hình thành như thế nào, khi nào phím Chèn được nhấn và vị trí của những từ bạn đang đọc này xuất hiện trên trang.

Thanh trạng thái cũng có thể chứa thông tin đồ họa chẳng hạn như trạng thái của yêu cầu in, liệu ứng dụng có được kết nối với Internet hay không và gần như bất kỳ thứ gì khác mà bạn có thể vẽ hoặc tạo hiệu ứng.

Trong .NET, các thanh trạng thái được đại diện bởi lớp `StatusStrip`, đây là một phần của hệ thống phân cấp lớp `ToolStrip` mà chúng ta đã thảo luận. Như một cách để kết thúc cuộc thảo luận của chúng ta và kiểm tra một loại dải công cụ khác, phần này sẽ xem xét nhanh chức năng này. Chúng ta thảo luận về dải trạng thái và bảng trạng thái hoặc nhãn và thêm thanh trạng thái mà chúng ta đã thấy trong hình 4.1 vào ứng dụng `MyPhotos` của mình.

Chúng ta nên lưu ý rằng các thanh trạng thái Win32 cũng được hỗ trợ bởi các lớp `StatusBar` và `StatusBarPanel`. Các lớp này được hỗ trợ cho các ứng dụng hiện có và ứng dụng mới, nhưng được ẩn trong Visual Studio để khuyến khích sử dụng các lớp dải công cụ mới.

4.3.1 Tạo một dải trạng thái Điều khiển `StatusStrip` kế thừa trực tiếp từ lớp `ToolStrip` cơ sở. Một bản tóm tắt của điều khiển này xuất hiện trong .NET Bảng 4.5. Chúng tôi sẽ không thảo luận về các chi tiết của


.NET Bảng 4.5 Lớp `StatusStrip`

Tính năng mới trong 2.0 Lớp `StatusStrip` là một dải công cụ được sử dụng để hiển thị thanh trạng thái trên một biểu mẫu. Lớp này có thể hiển thị một tập hợp các nhãn dưới dạng các đối tượng `ToolStripStatusLabel`. Lớp `StatusStrip` là một phần của không gian tên `System.Windows.Forms` và kế thừa từ lớp `ToolStrip`. Xem .NET Table 16.1 để biết danh sách các thành viên kế thừa và .NET Table 4.7 để biết thông tin về lớp `ToolStripStatusLabel`.

Công cộng Đặc tính	Dock (được ghi đè từ <code>ToolStrip</code> )	Nhận hoặc đặt kiểu lắp ghép cho điều khiển. Mặc định cho dải trạng thái là Dưới cùng.
	GripStyle (được ghi đè từ <code>ToolStrip</code> )	Nhận hoặc đặt xem tay cầm di chuyển bị ẩn hay hiển thị. Mặc định cho dải trạng thái là Ẩn.
	ShowItemToolTips (được ghi đè từ <code>ToolStrip</code> )	Nhận hoặc đặt xem các mục có hiển thị chú giải công cụ của chúng hay không. Giá trị mặc định cho dải trạng thái là sai.
	SizingGrip	Nhận hoặc đặt xem nút điều chỉnh kích thước để thay đổi kích thước biểu mẫu có xuất hiện trên thanh trạng thái hay không.
	Kéo dài (ghi đè từ <code>ToolStrip</code> )	Lấy hoặc đặt xem dải có mở rộng để lấp đầy chiều rộng của vùng chứa của dải chính hay không. Giá trị mặc định cho thanh trạng thái là true.

ToolStrip cho đến chương 16, nhưng một chi tiết cần rút ra từ bảng của chúng ta ở đây là lớp StatusStrip về cơ bản là một ToolStrip ghi đề chức năng cơ sở để đạt được sự xuất hiện của thanh trạng thái kiểu Win32. Điều này bao gồm việc gắn dải ở dưới cùng của biểu mẫu, ẩn tay cầm di chuyển, không hiển thị bất kỳ mẹo công cụ nào và kéo dài dải trên toàn bộ chiều rộng của biểu mẫu.

Về mặt kỹ thuật, một dải trạng thái có thể xuất hiện trong bất kỳ vùng chứa chính nào và được gắn vào bất kỳ cạnh nào của vùng chứa này, do đó, theo truyền thống, điều khiển này chỉ được đặt ở cơ sở của các đối tượng Biểu mẫu . Bước sau đây sẽ thêm một dải trạng thái vào ứng dụng MyPhotos của chúng tôi.

THÊM DẢI TÌNH TRẠNG VÀO ỨNG DỤNG CỦA CHÚNG TÔI		
	Hoạt động	Kết quả
1	Trong cửa sổ MainForm.cs [Thiết kế], hãy kéo điều khiển StatusStrip từ cửa sổ Hộp công cụ lên biểu mẫu.	Dải trạng thái mới xuất hiện ở dưới cùng của biểu mẫu trong cửa sổ trình thiết kế và trong khay thành phần bên dưới biểu mẫu. <div></div>

Lưu ý cách giao diện để điền vào dải trạng thái khá khác so với giao diện được sử dụng cho các menu. Biểu tượng nhỏ, chủ yếu là màu trắng ở bên trái thêm đối tượng ToolStripStatusLabel , trong khi mũi tên thả xuống bên cạnh biểu tượng này hiển thị menu ngữ cảnh gồm các đối tượng ToolStripItem khác nhau có thể được thêm vào dải.

Mã được tạo trong tệp MainForm.Designer.cs giống như mã mà chúng ta đã thấy khi thêm các điều khiển khác vào biểu mẫu của mình. Trường statusStrip1 được tạo ở đầu phương thức InitializeComponent , các giá trị không mặc định của nó được gán ở giữa và được thêm vào Biểu mẫu ở cuối phương thức. Trường được xác định ở cuối tệp được tạo, như được hiển thị ở đây:

```
riêng System.Windows.Forms.StatusStrip statusStrip1;
```

Hầu hết các dải trạng thái chứa một hoặc nhiều bảng trạng thái hoặc nhãn. Đây là chủ đề tiếp theo của chúng tôi.

4.3.2 Thêm nhãn dải trạng thái Trong các lớp dựa

trên Win32, các thanh trạng thái chứa các bảng điều khiển: lớp Thanh trạng thái của Windows Forms đại diện cho một thanh trạng thái và chứa một hoặc nhiều thành phần StatusBarPanel . Trong thế giới dải công cụ, thuật ngữ hơi khác một chút. Lớp StatusStrip đại diện cho một thanh trạng thái và chứa một hoặc nhiều đối tượng ToolStripStatusLabel . Chúng tôi sẽ sử dụng thuật ngữ nhãn trạng thái trong cuốn sách này để phù hợp với tên lớp mới.

Lớp ToolStripStatusLabel cho các nhãn trạng thái dựa trên lớp Tool StripLabel . Một bản tóm tắt của lớp này xuất hiện trong .NET Bảng 4.6. Lớp ToolStripStatusLabel thêm một số chức năng bổ sung dành riêng cho thanh trạng thái và được tóm tắt trong Bảng 4.7 của .NET .

.NET Bảng 4.6 Lớp ToolStripLabel

Tính năng mới trong 2.0 Lớp ToolStripLabel là một mục dải công cụ hiển thị văn bản và hình ảnh không thể chọn cũng như các mục siêu liên kết. Lớp ToolStripLabel là một phần của không gian tên System.Windows.Forms và kế thừa từ lớp ToolStripItem . Xem .NET Table 3.4 để biết danh sách các thành viên kế thừa.

Công cộng Đặc tính	CanSelect (được ghi đè từ ToolStripItem)	Nhận xem nội dung của mục có thể được chọn hay không.  Luôn sai đối với các mục nhãn.
	IsLink	Nhận hoặc đặt xem nhãn có phải là siêu kết nối hay không.
	LinkBehavior	Nhận hoặc đặt giá trị liệt kê LinkBehavior đại diện cho sự xuất hiện của một liên kết.
	LinkColor	Nhận hoặc đặt màu để sử dụng cho một liên kết bình thường. Các thuộc tính ActiveLinkColor và VisitedLinkColor tương ứng đại diện cho màu được sử dụng cho một liên kết được truy cập và đang hoạt động.
	Visited	Nhận hoặc đặt liệu liên kết có hiển thị như thể nó đã được truy cập hay không.

Như bạn có thể thấy trong Bảng .NET 4.6, lớp ToolStripLabel ghi đè thuộc tính CanSelect từ lớp ToolStripItem để đảm bảo rằng các nhãn như vậy không thể được chọn. Các thuộc tính khác liên quan đến việc xử lý nhãn dưới dạng siêu liên kết. Đặc biệt, thuộc tính IsLink xác định xem nhãn có xuất hiện dưới dạng siêu kết nối hay không. Trình xử lý sự kiện Nhấp chuột có thể được sử dụng để xử lý liên kết khi nó được nhấp vào.

Trong .NET Table 4.7, lớp ToolStripLabel được mở rộng để hoạt động trong dải trạng thái. Như bạn có thể thấy, ba thuộc tính mới được thêm vào: hai thuộc tính để xác định đường viền cho nhãn và thuộc tính Spring cho phép nhãn phát triển hoặc co lại khi dải trạng thái được thay đổi kích thước.

.NET Bảng 4.7 Lớp ToolStripStatusLabel

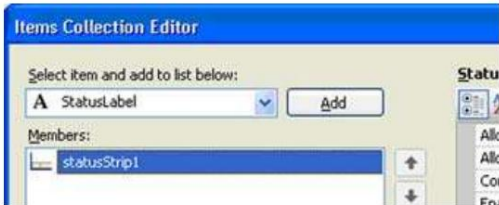
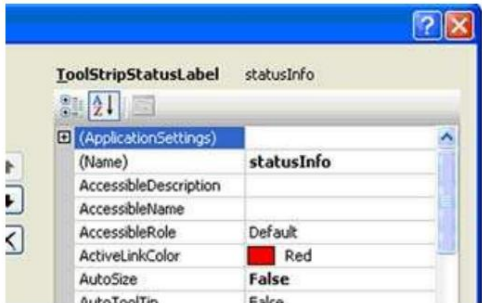
Tính năng mới trong 2.0 Lớp ToolStripStatusLabel là một mục nhãn để sử dụng trong điều khiển StatusStrip . Lớp ToolStripStatusLabel là một phần của không gian tên System.Windows.Forms và kế thừa từ lớp ToolStripLabel .

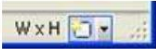
Công cộng Đặc tính	Border	Nhận hoặc đặt các cạnh của nhãn trạng thái sẽ hiển thị đường viền
	BorderStyle	Nhận hoặc đặt giá trị liệt kê Border3DStyle cho cách đường viền của nhãn sẽ xuất hiện
	Stretch	Nhận hoặc đặt xem bảng điều khiển sẽ mở rộng hay thu nhỏ để phù hợp với không gian có sẵn khi dải trạng thái được thay đổi kích thước

Quay trở lại ứng dụng MyPhotos của chúng tôi, chúng tôi đã có một đối tượng StatusStrip trên biểu mẫu của mình, vì vậy nhiệm vụ tiếp theo là xác định một bộ nhân hợp lý. Chúng tôi bắt đầu với ba nhân được đề cập ở đầu chương.

Khi chúng tôi thêm một dải trạng thái vào biểu mẫu của mình, chúng tôi đã đề cập đến giao diện được cung cấp để thêm nhân và các mục khác vào dải. Mặc dù chúng tôi có thể sử dụng giao diện này ở đây, thay vào đó, chúng tôi sử dụng cửa sổ Trình chỉnh sửa Bộ sưu tập Mục làm phương pháp thay thế cho tác vụ này.

Trình chỉnh sửa này cũng có sẵn cho các dải công cụ khác, bao gồm cả các dải menu.

THÊM NHÂN TRẠNG THÁI VÀO DẢI TRẠNG THÁI																
	Hoạt động	Kết quả														
1	<p>Trong cửa sổ MainForm.cs [Thiết kế], hiển thị các mục cho dải trạng thái.</p> <p>Làm thế nào để Nhấp chuột phải vào dải trạng thái và chọn mục Chỉnh sửa Mục.</p> <p>Luân phiên Nhấp vào thẻ thông minh được liên kết với dải trạng thái và nhấp vào Chỉnh sửa Mục trong cửa sổ Nhiệm vụ StatusStrip.</p>	<p>Cửa sổ Trình chỉnh sửa Bộ sưu tập Mục xuất hiện. Trình chỉnh sửa này cho phép một tập hợp các mục, trong trường hợp này là các mục dải công cụ trên dải trạng thái, được định cấu hình cho kiểm soát chính. Một phần của cửa sổ này được hiển thị ở đây.</p> 														
2	<p>Bấm vào nút Thêm hai lần để thêm hai nhân trạng thái vào điều khiển.</p>	<p>Các đối tượng ToolStripStatusLabel mới được hiển thị trong trình chỉnh sửa.</p>														
3	<p>Sửa đổi các thuộc tính của nhân đầu tiên như sau.</p> <table><tr><th colspan="2">Cài đặt</th></tr><tr><td>Tài sản</td><td>Giá trị</td></tr><tr><td>(Tên)</td><td>statusInfo</td></tr><tr><td>Kích thước tự động</td><td>Sai</td></tr><tr><td>Màu xuân</td><td>ĐÚNG VẬY</td></tr><tr><td>Chữ</td><td>giảm xuống</td></tr><tr><td>Căn chỉnh văn bản</td><td>Trung tâm bên trái</td></tr></table>	Cài đặt		Tài sản	Giá trị	(Tên)	statusInfo	Kích thước tự động	Sai	Màu xuân	ĐÚNG VẬY	Chữ	giảm xuống	Căn chỉnh văn bản	Trung tâm bên trái	
Cài đặt																
Tài sản	Giá trị															
(Tên)	statusInfo															
Kích thước tự động	Sai															
Màu xuân	ĐÚNG VẬY															
Chữ	giảm xuống															
Căn chỉnh văn bản	Trung tâm bên trái															
4	<p>Đóng Trình chỉnh sửa bộ sưu tập vật phẩm cửa sổ bằng cách nhấp vào nút OK.</p>	<p>Hai nhân trạng thái được hiển thị trong cửa sổ thiết kế.</p>														

THÊM NHÃN TRẠNG THÁI VÀO DẢI TRẠNG THÁI (TIẾP THEO)														
	Hoạt động	Kết quả												
5	<p>Nhấp vào nhãn trạng thái thứ hai và sửa đổi các thuộc tính của nó như sau.</p> <table><tr><th colspan="2">Cài đặt</th></tr><tr><th>Tài sản</th><th>Giá trị</th></tr><tr><td>(Tên)</td><td>trạng tháiHình ảnhKích thước</td></tr><tr><td>Biên Giới</td><td>trên và dưới</td></tr><tr><td>Kiểu viền</td><td>ChìmBên trong</td></tr><tr><td>Chữ</td><td>Rộng x Cao</td></tr></table>	Cài đặt		Tài sản	Giá trị	(Tên)	trạng tháiHình ảnhKích thước	Biên Giới	trên và dưới	Kiểu viền	ChìmBên trong	Chữ	Rộng x Cao	<p>Các thuộc tính được chỉ định trong tệp MainForm.Designer.cs đã tạo.</p> <pre>// // statusImageSize //  this.statusImageSize.BorderSides = ((System.Windows.Forms. ToolStripStatusLabelBorderSides) ((((System.Windows.Forms.ToolStrip- StatusLabelBorderSides.Left   System.Windows.Forms.ToolStrip- StatusLabelBorderSides.Top)   System.Windows.Forms.ToolStrip- StatusLabelBorderSides.Right)   System.Windows.Forms.ToolStrip- StatusLabelBorderSides.Bottom) )); this.statusImageSize.BorderStyle = System.Windows.Forms. Border3DStyle.SunkenInner; this.statusImageSize.Name = "statusImageSize"; this.statusImageSize.Size = new System.Drawing.Size(40, 17); this.statusImageSize.Text = "W x H";</pre>
Cài đặt														
Tài sản	Giá trị													
(Tên)	trạng tháiHình ảnhKích thước													
Biên Giới	trên và dưới													
Kiểu viền	ChìmBên trong													
Chữ	Rộng x Cao													
6	<p>Thêm nhãn trạng thái thứ ba vào dải trạng thái.</p> <p>Cách thực hiện Nhấp vào biểu tượng nhỏ màu trắng ở bên phải của dải.</p>  <table><tr><th colspan="2">Cài đặt</th></tr><tr><th>Tài sản</th><th>Giá trị</th></tr><tr><td>(Tên)</td><td>trạng tháiAlbumPos</td></tr><tr><td>Biên Giới</td><td>trên và dưới</td></tr><tr><td>Kiểu viền</td><td>ChìmBên trong</td></tr><tr><td>Chữ</td><td>1/1</td></tr></table>	Cài đặt		Tài sản	Giá trị	(Tên)	trạng tháiAlbumPos	Biên Giới	trên và dưới	Kiểu viền	ChìmBên trong	Chữ	1/1	<p>Nhãn thứ ba xuất hiện trong cửa sổ trình thiết kế và được khởi tạo trong mã được tạo. Ba nhãn trạng thái được thêm vào thuộc tính Mục của đối tượng StatusStrip bằng phương thức AddRange .</p> <pre>this.statusStrip1.Items.AddRange(mới System.Windows.Forms.ToolStripItem[] { this.statusInfo, this.statusImageSize, this.statusAlbumPos});</pre>
Cài đặt														
Tài sản	Giá trị													
(Tên)	trạng tháiAlbumPos													
Biên Giới	trên và dưới													
Kiểu viền	ChìmBên trong													
Chữ	1/1													

Ba nhãn này sẽ hiển thị mô tả về hình ảnh hiện tại, kích thước của hình ảnh và vị trí hiện tại của hình ảnh trong album. Tất nhiên, nhãn vị trí album là không cần thiết trong chương này. Chúng tôi thêm nó vào đây để tiết kiệm thời gian và không gian trong chương 6, khi ứng dụng của chúng tôi hiển thị một album thay vì một hình ảnh đơn lẻ.

Trong cài đặt thuộc tính của chúng tôi, hãy lưu ý cách các liên kết thích hợp BorderSides và BorderStyle phối hợp với nhau để xác định cách nhãn xuất hiện trong dải trạng thái. trong chúng tôi

Ví dụ: chúng tôi đặt nhãn kích thước và vị trí của mình để có đường viền ở cả bốn phía bằng cách sử dụng giá trị `SunkenInner` từ phép liệt kê `BorderStyle` .

Đối với hai nhãn còn lại, mô tả và kích thước, chúng ta nên cập nhật chúng bất cứ khi nào chúng ta tải một hình ảnh để chứa tên tệp làm mô tả và chiều rộng và chiều cao tính bằng pixel làm kích thước. Thuộc tính `Text` được kế thừa từ lớp `ToolStripItem` xác định văn bản sẽ hiển thị trên nhãn. Bạn cũng có thể hiển thị một hình ảnh bằng cách gán thuộc tính `Image` , nhưng chúng tôi không sử dụng thuộc tính này ở đây.

Một cách tiếp cận để cập nhật các nhãn này có thể là gán trực tiếp văn bản của từng nhãn trong trình xử lý sự kiện Nhấp chuột của menu Tải của chúng tôi. Đây là một ví dụ khác mà cách tiếp cận rõ ràng nhất có thể không phải là giải pháp lâu dài tốt nhất. Vì dải trạng thái của chúng tôi có khả năng thay đổi, nên chúng tôi gói gọn logic này trong một phương thức riêng tư. Chúng tôi tiếp tục các bước trước đó để thực hiện những thay đổi này.

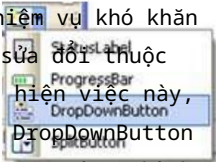
Gán VĂN BẢN CHO CÁC NHÃN TRẠNG THÁI		
	Hoạt động	Kết quả
7	Trong cửa sổ mã <code>MainForm.cs</code> , hãy xác định một phương thức riêng tư mới có tên là <code>SetStatusStrip</code> , phương thức này chấp nhận một đường dẫn tệp làm đối số duy nhất.	<pre>private void SetStatusStrip(đường dẫn chuỗi) {</pre>
8	Nếu một hình ảnh được gán cho hộp hình ảnh, hãy đặt nhãn <code>statusInfo</code> cho đường dẫn đã cho.	<pre>    nếu (pbxPhoto.Image != null) {          statusInfo.Text = đường dẫn;</pre>
9	Đồng thời đặt nhãn <code>statusImageSize</code> thành kích thước của hình ảnh.  Cách sử dụng thuộc tính <code>Chiều rộng</code> và <code>Chiều cao</code> trong lớp <code>Hình ảnh</code> .	<pre>        statusImageSize.Text =             String.Format("{0:##}x{1:##}",                 pbxPhoto.Image.Width,                 pbxPhoto.Image.Height); // statusAlbumPos         được đặt trong ch. 6     }</pre>
10	Nếu không có hình ảnh nào được gán cho điều khiển hộp hình ảnh, hãy đặt các nhãn trạng thái thành trống.  Làm thế nào để đặt từng thuộc tính Văn bản thành <code>null</code> .	<pre>    khác     {         statusInfo.Text = null;         statusImageSize.Text = null;         statusAlbumPos.Text = null;     } }</pre>
11	Cập nhật thanh trạng thái cuối trình xử lý sự kiện <code>menuFileLoad_Click</code> .	<pre>menu void riêngFileLoad_Click(đối tượng người     gửi, EventArgs e) {     . . .     if (dlg.ShowDialog() == DialogResult.OK) {         . . .         SetStatusStrip(dlg.FileName);     } dlg.Dispose(); }</pre>

Gán VĂN BẢN VÀO NHÃN TRẠNG THÁI (TIẾP THEO)		
	Hoạt động	Kết quả
12	Cũng gọi SetStatusStrip trong hàm tạo của biểu mẫu để khởi tạo cài đặt trạng thái.	<pre>công khai MainForm() {      Khởi tạo Thành phần();     SetTitleBar();     SetStatusStrip(null);      menuView.DropDown = ctxMenuPhoto; }</pre>

Biên dịch và chạy chương trình để xem cập nhật dải trạng thái. Đảm bảo rằng bạn đã kiểm tra tải tệp hợp lệ và tệp không hợp lệ để thấy rằng phương thức SetStatusStrip xử lý cả hai trường hợp.

THỬ NÓ! Có một số thay đổi bạn có thể thực hiện ở đây để thử nghiệm các thuộc tính và cài đặt khác nhau được hỗ trợ bởi các lớp StatusStrip và ToolStripStatusLabel . Dưới đây là một vài gợi ý:

- Có một vấn đề nhỏ với nhãn đầu tiên của chúng tôi, vì nếu đường dẫn tệp quá dài thì người dùng không thể đọc được. Chỉ định thuộc tính ToolTipText cho nhãn statusInfo để người dùng có thể xem đường dẫn đầy đủ trong chú giải công cụ. Bạn cũng sẽ cần sửa đổi thuộc tính ShowItemToolTips cho dải trạng thái vì thuộc tính này được đặt thành false theo mặc định.
- Đối với một nhiệm vụ khó khăn hơn, hãy tạo một mục mới trong dải trạng thái để sửa đổi thuộc tính Kiểu viền của nhãn statusImageSize . Để thực hiện việc này, hãy thêm nút thả xuống vào dải bằng cách chọn mục DropDownButton từ trình đơn thả xuống, như thể hiện trong hình. Đặt (Tên) thành "statusBorder" và Văn bản thành "Size Border". Tra cứu kiểu liệt kê Border3DStyle trong tài liệu .NET để tìm mười giá trị có thể có cho kiểu liệt kê này.



Sử dụng các giá trị này để sao chép logic từ trình xử lý menu Hình ảnh của chúng tôi ở phần trước của chương. Mục mới là một đối tượng ToolStrip DropDownButton và bạn có thể thêm các mục thả xuống cho từng giá trị liệt kê vào nút, đặt thuộc tính Thẻ trên mỗi mục và xử lý sự kiện DropDownItemClicked để sửa đổi thuộc tính statusImageSize.BorderStyle . Chạy ứng dụng để tự động thay đổi kiểu viền để bạn có thể xem tác dụng của từng tùy chọn.

## 4.4 TÓM TẮT

Chương này đã thảo luận về dải menu ngữ cảnh và dải trạng thái. Chúng ta bắt đầu với lớp `ContextMenuStrip` và những điểm tương đồng của nó với điều khiển `MenuStrip`. Một ví dụ trong ứng dụng `MyPhotos` của chúng tôi đã tạo menu Hình ảnh để thay đổi cách hiển thị hình ảnh trong điều khiển `PictureBox` của chúng tôi. Menu Chế độ xem mới trong lớp `MenuStrip` của chúng tôi được liên kết khéo léo với menu ngữ cảnh của chúng tôi để cả hai menu sẽ hiển thị cùng nội dung và hành vi. Logic này cho menu Hình ảnh yêu cầu sử dụng cấu trúc Enum để chuyển đổi các giá trị liệt kê `PictureBoxSizeMode` sang và từ các đối tượng chuỗi.

Chúng tôi kết thúc chương này với một cuộc thảo luận về dải trạng thái. Chúng tôi đã chỉ ra cách các điều khiển `StatusStrip` chứa các mục `ToolStripStatusLabel` và tạo một dải trạng thái với một số nhãn trong ứng dụng của chúng tôi.

Một số từ khóa C# mới đã được kiểm tra trong suốt quá trình, cụ thể là từ khóa `foreach` và `as`, và một số phương thức đối tượng phổ biến như `Equals` và `ToString` đã được sử dụng. Chúng tôi đã xem xét cửa sổ Thuộc tính trong Visual Studio chi tiết hơn và sử dụng cửa sổ này để thêm các sự kiện khác nhau vào chương trình của chúng tôi.

Chương này hoàn thành cuộc thảo luận ban đầu của chúng ta về các dải công cụ và các mục của dải công cụ, được thực hiện ở đây trong ngữ cảnh của các menu và dải trạng thái. Các chương trong tương lai dựa vào kiến thức này để thực hiện các thay đổi bổ sung về menu hoặc dải trạng thái khi chúng ta đọc hết cuốn sách. Chúng ta cũng thảo luận chi tiết hơn về các lớp `ToolStrip` và `ToolStripItem` trong chương 16.

Chương tiếp theo đưa chúng ta ra khỏi không gian tên Windows Forms một thời gian ngắn để kiểm tra các thư viện có thể tái sử dụng. Điều này đặt nền tảng cho các cuộc thảo luận của chúng ta về các điều khiển Windows Forms khác nhau trong các chương tiếp theo.