Visual C# .Net using framework 4.5

Eng. Mahmoud Ouf

Lecture 07

Menu

In windows applications, menus are used to provide a list of commands available for user to execute in the application. Example the File menu, which contain New, Open, Close, …

The menu that sits between a form's title bar and the client area is referred as main menu.

Many application support Shortcut menus, context menus, which are menus that appear at the mouse cursor position when you right click the mouse.

A main menu is associated with a form, while context menu is associated with a particular control, that is clicking different controls often causes different context menus to be invoked.

A menu (main or context) contains menu items, such as File, Open, Save,..

Menu

You will find that the File and Edit menu items are different from the Open, Save items. File, Edit are located at the visible part of the program main menu, while other are not.

The items at the visible part are called “Top Level Item”. Selecting any Top Level Item will invoke a menu was called “PopUp menu” or “Drop Down Menu” but now is called SubMenu or ChildMenu.

From the Windows Forms program, The File menu item contains a collection of other menu items contains (Open, Save,…) and the Edit menu item contains a collection of menu items contains (Cut, Copy,…).

One step back, the Main menu itself is a collection of menu items contains (File, Edit,…). Each menu item in the Main menu is associated with its own array of menu items. And in some case some of these menu items have their own array of menu items.

Menu

Similarly, a context menu is an array of menu items.

To Deal With menus, we have 3 classes:

- MenuStrip (the main menu)
- ToolStripMenuItem (sub menus)
- ContextMenuStrip (context menu)

Menu

Creating the Main Menu

1) Create an object from the MenuStrip class
2) Create the Top Level Menu item (object from ToolStripMenuItem class)
3) Add this object (from ToolStripMenuItem) to the object (from MenuStrip) use “items collection”
4) Now, make the submenu item (Object from ToolStripMenuItem)
5) Add it to the Top Level Menu (use DropDownItems collections)
6) Repeat steps 4, 5 to add other submenu
7) Repeat Steps 2 to 6 to add other top level Menu items
8) Attach MainMenu to the Form

Menu

Creating the Main Menu

Step 1:

```
MenuStrip menuStrip1;   //define the object
menuStrip1 = new MenuStrip();          //object creation
```

Step 2:

```
ToolStripMenuItem fileToolStripMenuItem =
new ToolStripMenuItem("&File");
```

Step 3:

```
menuStrip1.Items.Add(this.fileToolStripMenuItem);
```

Menu

Creating the Main Menu

Step 4:

```
ToolStripMenuItem exitToolStripMenuItem =
        new ToolStripMenuItem (“E&xit”,
        new EventHandler(this.exitToolStripMenuItem_Click));
exitToolStripMenuItem.ShortcutKeys = (Keys)(Keys.Control | Keys.X);
```

Step 5:

```
fileToolStripMenuItem.DropDownItems.Add(this.exitToolStripMenuItem);
```

Step 8:

```
this.Controls.Add(this.menuStrip1);
this.MainMenuStrip = this.menuStrip1;
```

Menu

Creating the Main Menu

Step 4:

```
ToolStripMenuItem exitToolStripMenuItem =
        new ToolStripMenuItem (“E&xit”,
        new EventHandler(this.exitToolStripMenuItem_Click));
exitToolStripMenuItem.ShortcutKeys = (Keys)(Keys.Control | Keys.X);
```

Step 5:

```
fileToolStripMenuItem.DropDownItems.Add(this.exitToolStripMenuItem);
```

Step 8:

```
this.Controls.Add(this.menuStrip1);
this.MainMenuStrip = this.menuStrip1;
```

Menu

Creating the Main Menu

The & in the menu name string, tell which variable will be associated with the Alt key to open the menu

Also, each command may have a shortcut to execute this command with no need to open the menu

Context Menu

Creating the Context Menu

1) Create an object from the ContextMenuStrip class
2) Now, make the submenu item (object from ToolStripMenuItem)
3) Add it to the ContextMenuStrip (using Items collections)
4) Repeat steps 2, 3 to add other submenu
5) Attach ContextMenuStrip to the Form (through contextMenuStrip property of the form)

Toolbar

Toolbars provides an alternate means to activate a given menu item. Most applications use toolbar images that are 16 pixels square.

This is done by ToolStrip class

It can Contains:

- ToolStripButton,
- ToolStripLabel,
- ToolStripDropDownButton,
- ToolStripComboBox,
- ToolStripTextBox

Toolbar

Creating and using Toolbar

1) Create object From ToolStrip class
2) Create objects From ToolStripButton, ToolStripLabel, … class
3) Configure each object
4) Configure toolbar and Add buttons (using Items collection)
5) Add the new bar to the control (using Controls collection)

Status bar

Status bars usually conveys textual information to the user.

This is done by StatusStrip class

It can Contains:

ToolStripStatusLabel,

ToolStripDropDownButton,

ToolStripProgressBar

Creating and using Status bar

1) Create object From StatusStrip class
2) Create objects From ToolStripStatusLabel, ToolStripDropDownButton, … class
3) Configure each object
4) Configure status bar and Add buttons (using Items collection)
5) Add the new bar to the control (using Controls collection)

Dialog Box

The DialogBox is a kind of form that is used to set some properties.
There is a lot of defined dialogBox “Common Dialog Box” such as “Open, Save, …)
Also, the user can make his own DialogBox.
There are two types of DialogBox:

- ***Modal DialogBox :***

it is the most common, it changes the mode of i/p from the main application to the dialog box. The user can’t switch between DialogBox and the application until he end explicitly the DialogBox (Ok/Cancel).

- ***Modeless DialogBox :***

here the user can switch between the DialogBox and the Application.

Dialog Box

•*Modal DialogBox :*

it is the most common, it changes the mode of i/p from the main application to the dialog box. The user can't switch between DialogBox and the application

Programmers often use Modal Dialog Box when program needs to obtain information from a user beyond what can be easily managed in Menu.

To Create Modal DialogBox:

1. Create class for DialogBox (Has 2 buttons Ok/Cancel)
2. Create class for Application Create object from Dialog Class
3. Call ShowDialog (Invoke the Modal)

There is a property of the dialogbox form named DialogResult of type DialogResult which is enumeration.

Dialog Box

•*Modal DialogBox :*

In either case select Ok/Cancel, the dialog box is closed “disappear from the screen” and return to the point it is called from.

When closing, the ShowDialog() return the value of the DialogResult property

Although the dialogbox has been terminated and no longer visible, the dialogbox object is still valid. This means that we can access all member of the dialogBox.

•*Modeless DialogBox:*

There is a property named owner, set its value to the Application name. Before calling Show() method

Note: Don’t forget to set the value of the DialogResult of the DialogBox to DialogBox.Ok / DialogBox.Cancel in the handel of Ok/Cancel buttons.

Common Dialog Box

C# contains a set of common dialog boxes, which are ready made dialog box (Color, Open, Save, …).

All common dialog boxes are “Modal dialog box”, and are treated as any modal dialog box.

The dealing with common dialog box, is like other dialog box except you are not going to create the dialog box.

I.E.

In the event that will display the dialog box, you have to:

1) Create an object from the dialog box
2) Call the ShowDialog() method
3) Handle the value which return from ShowDialog(), even Ok or Cancel

Tab Control

The TabControl control creates tabbed windows. This allows the programmer to design user interfaces that fit a large number of controls or a large amount of data without using up valuable screen “real estate.”

TabControls contain TabPage objects which can contain controls.

1) add the TabPages to the TabControl.
2) add controls to the TabPage objects

Only one TabPage is displayed at a time. Programmers can add TabControls visually by dragging and dropping them onto a form in design mode.

To add TabPages in the Visual Studio .NET designer, right-click the TabControl, and select Add Tab.

Alternatively, click the TabPages collection in the Properties window, and add tabs in the dialog that appears.

Tab Control

TabControl properties:

ImageList:	Specifies images to be displayed on a tab
ItemSize:	Specifies tab size.
MultiLine:	Indicates whether multiple rows of tabs can be displayed
SelectedIndex:	Indicates index of TabPage that is currently selected
SelectedTab:	Indicates the TabPage that is currently selected
TabCount:	Returns the number of tabs.
TabPages:	Gets the collection of TabPages within our TabControl