

# WINDOWS FORMS CONTROLS

## Topics in This Chapter

- *Introduction:* A class hierarchy diagram offers a natural way to group Windows Forms controls by their functionality.
- *Button Controls:* The `Button`, `CheckBox`, and `RadioButton` controls are designed to permit users to make one or more selections on a form.
- *PictureBox and TextBox Controls:* The `PictureBox` control is used to display and scale images; the `TextBox` control can be used to easily display and edit single or multiple lines of text.
- *List Controls:* The `ListBox`, `ComboBox`, and `CheckListBox` offer different interfaces for displaying and manipulating data in a list format.
- *ListView and TreeView Controls:* The `ListView` offers multiple views for displaying data items and their associated icons. The `TreeView` presents hierarchical information in an easy-to-navigate tree structure.
- *Timer and Progress Bar Controls:* A timer can be used to control when an event is invoked, a `ProgressBar` to visually monitor the progress of an operation.
- *Building a User Control:* When no control meets an application's needs, a custom one can be crafted by combining multiple controls or adding features to an existing one.
- *Moving Data Between Controls:* Drag and drop provides an easy way for users to copy or move an item from one control to another. .NET offers a variety of classes and events required to implement this feature.
- *Using Resources:* Resources required by a program, such as title, descriptive labels, and images, can be embedded within an application's assembly or stored in a *satellite* assembly. This is particularly useful for developing international applications.

# Chapter

# 7

The previous chapter introduced the `Control` class and the methods, properties, and events it defines for all controls. This chapter moves beyond that to examine the specific features of individual controls. It begins with a survey of the more important .NET controls, before taking an in-depth look at how to implement controls such as the `TextBox`, `ListBox`, `TreeView`, and `ListView`. Also included is a discussion of the .NET drag-and-drop features that are used to move or copy data from one control to another.

Windows Forms (WinForms) are not restricted to using the standard built-in controls. Custom GUI controls can be created by extending an existing control, building a totally new control, or fashioning a user control from a set of related widgets. Examples illustrate how to extend a control and construct a user control. The chapter concludes with a look at resource files and how they are used to create GUI applications that support users from multiple countries and cultures.

## 7.1 A Survey of .NET Windows Forms Controls

The `System.Windows.Forms` namespace contains a large family of controls that add both form and function to a Windows-based user interface. Each control inherits a common set of members from the `Control` class. To these, it adds the methods, properties, and events that give the control its own distinctive behavior and appearance.

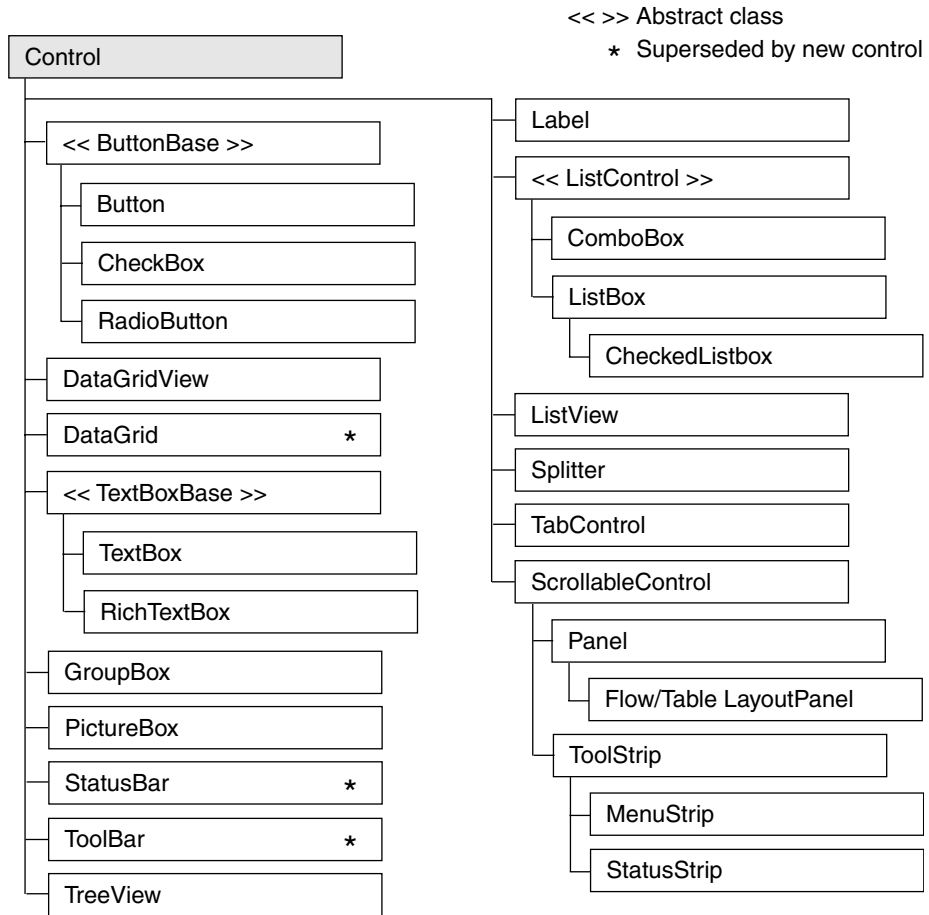


Figure 7-1 Windows Forms control hierarchy

Figure 7-1 shows the inheritance hierarchy of the Windows Forms controls. The controls marked by an asterisk (\*) exist primarily to provide backward compatibility between .NET 2.0 and .NET 1.x. Specifically, the DataGrid has been superseded by the DataGridView, the StatusBar by the StatusStrip, and the ToolBar by the ToolStrip. Table 7-1 provides a summary of the more frequently used controls in this hierarchy.

**Table 7-1** Selected Windows Forms Controls

Control	Use	Description
Button	Fires an event when a mouse click occurs or the Enter or Esc key is pressed.	Represents a button on a form. Its <code>text</code> property determines the caption displayed on the button's surface.
CheckBox	Permits a user to select one or more options.	Consists of a check box with text or an image beside it. The check box can also be represented as a button by setting: <code>checkBox1.Appearance = Appearance.Button</code>
CheckedListBox	Displays list of items.	<code>ListBox</code> with checkbox preceding each item in list.
ComboBox	Provides <code>TextBox</code> and <code>ListBox</code> functionality.	Hybrid control that consists of a text-box and a drop-down list. It combines properties from both the <code>TextBox</code> and the <code>ListBox</code> .
<code>DataGridView</code> <code>GridView</code>	Manipulates data in a grid format.	The <code>DataGridView</code> is the foremost control to represent relational data. It supports binding to a database. The <code>DataGridView</code> was introduced in .NET 2.0 and supersedes the <code>DataGrid</code> .
GroupBox	Groups controls.	Use primarily to group radio buttons; it places a border around the controls it contains.
ImageList	Manages a collection of images.	Container control that holds a collection of images used by other controls such as the <code>ToolStrip</code> , <code>ListView</code> , and <code>TreeView</code> .
Label	Adds descriptive information to a form.	Text that describes the contents of a control or instructions for using a control or form.
ListBox	Displays a list of items— one or more of which may be selected.	May contain simple text or objects. Its methods, properties, and events allow items to be selected, modified, added, and sorted.















































































































