C#控件查询手册

龙马工作室搜集整理制作

索引

图形显示控件

图形存储控件

值的设置控件

数据的设置控件

对话框控件

菜单控件

命令控件

用户帮助控件

分组控件

音频控件

数据显示控件

DataGridView 控件

DataGridView 控件提供用来显示数据的可自定义表。使用 DataGridView 类，可以自定义单元格、行、列和边框。

在可自定义的网格中显示数据。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class DataGridView : Control, ISupportInitialize
```

备注

DataGridView 控件提供用来显示数据的可自定义表。使用 DataGridView 类，可以通过使用 DefaultCellStyle、ColumnHeadersDefaultCellStyle、CellBorderStyle 和 GridColor 等属性对单元格、行、列和边框进行自定义。有关更多信息，请参见 Windows 窗体 DataGridView 控件中的基本格式设置和样式设置。

可以使用 DataGridView 控件来显示有基础数据源或没有基础数据源的数据。如果没有指定数据源，可以创建包含数据的列和行，并将它们直接添加到 DataGridView。或者，可以设置 DataSource 和 DataMember 属性，以便将 DataGridView 绑定到数据源，并自动用数据填充该控件。有关更多信息，请参见在 Windows 窗体 DataGridView 控件中显示数据。

在处理大量数据时，可以将 VirtualMode 属性设置为 true，以便显示可用数据的子集。虚拟模式要求实现用来填充 DataGridView 控件的数据缓存。有关更多信息，请参见 Windows 窗体 DataGridView 控件中的数据显示模式。

有关 DataGridView 控件中可用功能的其他信息，请参见 DataGridView 控件（Windows 窗体）。

虽然 DataGridView 控件替代了以前版本的 DataGrid 控件并增加了功能，但是为了实现向后兼容并考虑到将来的使用（如果您选择的话），仍然保留了 DataGrid 控件。

数据绑定和定位控件

BindingSource 组件

通过提供当前项管理、更改通知和其他服务，来简化将窗体上的控件绑定到数据的过程。封装窗体的数据源。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
Public class BindingSource : Component, IBindingListView, IBindingList, IList,ICollection,IEnumerable,ITypedList,ICancelAddNew,ISupportInitializeNotification, ISupportInitialize, ICurrencyManagerProvider
```

备注

BindingSource 组件有两种用途。首先，它通过提供一个间接寻址层、当前项管理、更改通知和其他服务简化了窗体中控件到数据的绑定。这是通过将 BindingSource 组件附加到数据源然后将窗体中的控件绑定到 BindingSource 组件来实现的。与数据的所有进一步交互，包括定位、排序、筛选和更新，都通过调用 BindingSource 组件实现。

第二，BindingSource 组件可以作为一个强类型的数据源。通常，基础数据源的类型通过以下机制之一固定：

使用 Add 方法可将某项添加到 BindingSource 组件中。

将 DataSource 属性设置为一个列表、单个对象或类型。

这两种机制都创建一个强类型列表。BindingSource 支持由其 DataSource 和 DataMember 属性指示的简单数据绑定和复杂数据绑定。

BindingSource 提供了用于访问基础数据的成员。通过 Current 属性可以检索当前项，通过 List 属性可以检索整个列表。通过 Current、RemoveCurrent、EndEdit、CancelEdit、Add 和 AddNew 方法可支持对当前项的编辑操作。尽管对于所有基础数据源类型会自动处理当前项管理，但该类公开了许多允许自定义的事件，例如 CurrentItemChanged 和 DataSourceChanged。

绑定到 BindingSource 组件的数据源也可以使用 BindingNavigator 类定位和管理，该类提供像 VCR 一样的用户界面 (UI) 用于定位列表中的项。尽管 BindingNavigator 可以绑定到任何数据源，但它被设计为通过其 BindingNavigator.BindingSource 属性与 BindingSource 组件集成。

BindingSource 类的默认属性为 DataSource。默认事件为 CurrentChanged。

BindingNavigator 控件

提供工具栏式的界面来定位和操作窗体上的数据。

表示窗体上绑定到数据的控件的导航和操作用户界面 (UI)。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class BindingNavigator : ToolStrip, ISupportInitialize
```

备注

BindingNavigator 控件表示在窗体上定位和操作数据的标准化方法。多数情况下，BindingNavigator 与 BindingSource 控件成对出现，用于浏览窗体上的数据记录，并与它们交互。在这些情况下，BindingSource 属性被设置为作为数据源的关联 System.Windows.Forms.BindingSource 组件。

默认情况下，BindingNavigator 控件的用户界面 (UI) 由一系列 ToolStrip 按钮、文本框和静态文本元素组成，用于进行大多数常见的数据相关操作（如添加数据、删除数据和在数据中导航）。每个控件都可以通过 BindingNavigator 控件的关联成员进行检索或设置。类似地，还与以编程方式执行相同功能的 BindingSource 类的成员存在一一对应关系，如下表所示。

UI 控件	BindingNavigator 成员	BindingSource 成员
移到最前	MoveFirstItem	MoveFirst
前移一步	MovePreviousItem	MovePrevious
当前位置	PositionItem	Current
统计	CountItem	Count
移到下一条记录	MoveNextItem	MoveNext
移到最后	MoveLastItem	MoveLast
新添	AddNewItem	AddNew
删除	DeleteItem	RemoveCurrent

将 BindingNavigator 控件添加到窗体并绑定到数据源（例如 BindingSource）时，将自

动在此表中建立关系。

BindingNavigator 的所有构造函数都调用 AddStandardItems 方法以将标准的 UI 控件集与导航工具栏关联起来。可使用以下技术之一自定义此工具栏：

创建带有 BindingNavigator(Boolean) 构造函数的 BindingNavigator，此构造函数接受 Boolean 型的 addStandardItems 参数，并将此参数设置为 false。然后将需要的 ToolStripItem 对象添加到 Items 集合。

如果需要进行大量的自定义设置，或者将重复使用自定义设计，应从 BindingNavigator 派生一个类并重写 AddStandardItems 方法以定义附加标准项或替换标准项。

文本编辑控件

TextBox 控件

显示设计时输入的文本，它可由用户在运行时编辑或以编程方式更改。
表示 Windows 文本框控件。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class TextBox : TextBoxBase
```

备注

使用 TextBox 控件，用户可以在应用程序中输入文本。此控件具有标准 Windows 文本框控件所没有的附加功能，包括多行编辑和密码字符屏蔽。

通常，TextBox 控件用于显示单行文本或将单行文本作为输入来接受。可以使用 Multiline 和 ScrollBars 属性，从而能够显示或输入多行文本。通过将 AcceptsTab 和 AcceptsReturn 属性设置为 true，可在多行 TextBox 控件中更加灵活地操作文本。

通过将 MaxLength 属性设置为一个特定的字符数，可以限制输入到 TextBox 控件中的文本数量。TextBox 控件还可用于接受密码和其他敏感信息。可以使用 PasswordChar 属性屏蔽在控件的单行版本中输入的字符。使用 CharacterCasing 属性可使用户在 TextBox 控件中只能输入大写字符、只能输入小写字符，或者输入大小写字符的组合。

若要限制某些文本不被输入到 TextBox 控件，可以为 KeyDown 事件创建一个事件处理程序，以便验证在控件中输入的每个字符。也可以通过将 ReadOnly 属性设置为 true 来限制 TextBox 控件中的所有数据项输入。

Windows Mobile for Pocket PC, Windows Mobile for Smartphone, Windows CE 平台说明：在 Pocket PC 应用程序中，单行文本框中的选项卡显示为括号，但当 Multiline 设置为 true 时正常显示。

RichTextBox 控件

使文本能够以纯文本或 RTF 格式显示。

表示 Windows 多格式文本框控件。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class RichTextBox : TextBoxBase
```

备注

用户可以通过 RichTextBox 控件输入和编辑文本。该控件还提供比标准 TextBox 控件更高级的格式设置功能。可以将文本直接分配给该控件，或者从 RTF 格式文件或纯文本文件加载文本。可以为控件内的文本分配字符和段落格式设置。

RichTextBox 控件提供许多可对控件内任何文本部分应用格式设置的属性。若要更改文本的格式设置，必须首先选定此文本。只能为选定的文本分配字符和段落格式设置。对选定的文本内容进行设置后，在选定内容后输入的所有文本也用相同的设置进行格式设置，直到更改设置或选定控件文档的不同部分为止。SelectionFont 属性使您得以将文本以粗体或斜体显示。还可以使用此属性更改文本的大小和字样。SelectionColor 属性使您得以更改文本的颜色。若要创建项目符号列表，可以使用 SelectionBullet 属性。还可以通过设置 SelectionIndent、SelectionRightIndent 和 SelectionHangingIndent 属性调整段落格式设置。

RichTextBox 控件提供具有打开和保存文件的功能的方法。LoadFile 方法使您得以将现有的 RTF 或 ASCII 文本文件加载到控件中。还可以从已打开的数据流加载数据。SaveFile 使您得以将文件保存到 RTF 或 ASCII 文本中。与 LoadFile 方法相似，还可以使用 SaveFile 方法保存到开放式数据流。RichTextBox 控件还提供查找文本字符串的功能。Find 方法被重载，可以同时查找控件文本内的文本字符串以及特定字符。

也可以将 RichTextBox 控件初始化为内存中存储的数据。例如，可以将 Rtf 属性初始化为包含要显示文本的字符串，包括确定如何设置该文本格式的 RTF 代码。

如果控件内的文本包含链接（如到网站的链接），则可以使用 DetectUrls 属性适当地显示控件文本中的链接。然后可以处理 LinkClicked 事件以执行与该链接关联的任务。SelectionProtected 属性使您得以保护控件内的文本不被用户操作。当控件中有受保护的文本时，可以处理 Protected 事件以确定用户何时曾试图修改受保护的文本，并提醒用户该文本是受保护的，或向用户提供标准方式供其操作受保护的文本。

已使用 TextBox 控件的应用程序很容易就可以调整为使用 RichTextBox 控件。但是，RichTextBox 控件没有与 TextBox 控件相同的 64K 字符容量限制。与字处理应用程序（如 Microsoft Word）类似，RichTextBox 通常用于提供文本操作和显示功能。

MaskedTextBox 控件

约束用户输入的格式

使用掩码区分正确和不正确的用户输入。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class MaskedTextBox : TextBoxBase
```

备注

MaskedTextBox 类是一个增强型的 TextBox 控件，它支持用于接受或拒绝用户输入的声明性语法。通过使用 Mask 属性，无需在应用程序中编写任何自定义验证逻辑，即可指定下列输入：

必需的输入字符。

可选的输入字符。

掩码中的给定位置所需的输入类型；例如，只允许数字、只允许字母或者允许字母和数字。

掩码的原义字符，或者应直接出现在 MaskedTextBox 中的字符；例如，电话号码中的连字符 (-)，或者价格中的货币符号。

输入字符的特殊处理；例如，将字母字符转换为大写字母。

当 MaskedTextBox 控件在运行时显示时，会将掩码表示为一系列提示字符和可选的原义字符。表示一个必需或可选输入的每个可编辑掩码位置都显示为单个提示字符。例如，数字符号 (#) 通常用作数字字符输入的占位符。可以使用 PromptChar 属性来指定自定义提示字符。HidePromptOnLeave 属性决定当控件失去输入焦点时用户能否看到提示字符。

当用户在掩码文本框中键入内容时，有效的输入字符将按顺序替换其各自的提示字符。如果用户键入无效的字符，将不会发生替换。在这种情况下，如果 BeepOnError 属性设置为 true，将发出警告声，并引发 MaskInputRejected 事件。可以通过处理此事件来提供您自己的自定义错误处理逻辑。

如果当前插入点位于原义字符处，用户将有多种选择：

如果键入提示字符以外的字符，将自动跳过该原义字符，输入字符将应用于下一个提示字符所表示的下一个可编辑位置。

如果键入提示字符，并且 AllowPromptAsInput 属性为 true，输入将覆盖提示字符，插入点将移至掩码中的下一个位置。

始终可以使用箭头键来定位到上一个或下一个位置。

可以使用 MaskFull 属性来验证用户是否输入了所有必需的输入内容。Text 属性将始终

检索按照掩码和 TextMaskFormat 属性设置格式的用户输入。

实际上，MaskedTextBox 控件将所有掩码处理工作交给由 MaskedTextProvider 属性指定的 System.ComponentModel.MaskedTextProvider 类来完成。此标准提供程序支持除代理项和纵向组合字符以外的所有 Unicode 字符；但是，可以使用 AsciiOnly 属性将输入限定为字符集 a-z、A-Z 和 0-9 内的字符。

掩码不能保证用户输入一定会表示给定类型的有效值，例如，输入的年龄值可能为 -9。通过将值的类型的实例赋给 ValidatingType 属性，可以确保用户输入表示一个有效值。通过监视 TypeValidationCompleted 事件，可以检测当 MaskedTextBox 包含无效值时，用户是否将焦点从该控件移开。如果键入验证成功，可以通过 TypeValidationEventArgs 参数的 ReturnValue 属性使用表示该值的对象。

与 TextBox 控件一样，几个通用键盘快捷键不能用于 MaskedTextBox。尤其是 Ctrl-R（右对齐文本）、Ctrl-L（左对齐文本）和 Ctrl-E（文本居中）都无效。

信息显示控件

Label 控件

显示用户无法直接编辑的文本。

表示标准 Windows 标签。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class Label : Control
```

备注

Label 控件通常用于提供控件的描述性文字。例如，可使用 Label 为 TextBox 控件添加描述性文字，以便将控件中所需的数据类型通知用户。Label 控件还可用于为 Form 添加描述性文字，以便为用户提供有帮助作用的信息。例如，可将 Label 添加到 Form 的顶部，为用户提供关于如何将数据输入窗体上的控件中的说明。Label 控件还可用于显示有关应用程序状态的运行时信息。例如，可将 Label 控件添加到窗体，以便在处理一列文件时显示每个文件的状态。

Label 参与窗体的 Tab 键顺序，但不接收焦点（Tab 键顺序中的下一个控件接收焦点）。例如，如果 UseMnemonic 属性设置为 true，并且在控件的 Text 属性中指定助记键字符（“and”符 (&) 之后的第一个字符），则当用户按下 Alt+ 助记键时，焦点移动到 Tab 键顺序中的下一个控件。该功能为窗体提供键盘导航。除了显示文本外，Label 控件还可使用 Image 属性显示图像，或使用 ImageIndex 和 ImageList 属性组合显示图像。

LinkLabel 控件

将文本显示为 Web 样式的链接，并在用户单击该特殊文本时触发事件。该文本通常是到另一个窗口或网站的链接。

表示可显示超链接的 Windows 标签控件。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class LinkLabel : Label, IButtonControl
```

备注

ToolStripLabel 控件替换并向 Label 和 LinkLabel 控件添加功能。但是，Label 和 LinkLabel 控件均被保留下来，以实现向后兼容性和供以后使用（如果选用的话）。

LinkLabel 控件除了可显示超链接以外，它与 Label 控件类似。在控件的文本中可以指定多个超链接。每个超链接可在应用程序内执行不同的任务。例如，可使用超链接在 Microsoft Internet Explorer 中显示网站或加载与应用程序关联的日志文件。

在 LinkLabel 控件中显示的每个超链接都是 LinkLabel.Link 类的一个实例。LinkLabel.Link 类定义超链接的显示信息、状态和位置。另外，LinkLabel.Link 类的 LinkData 属性使您得以将信息（如要显示的 URL）与超链接关联。当用户单击控件内的超链接时，引发 LinkClicked 事件，表示所单击的超链接的 LinkLabel.Link 对象作为 LinkLabelLinkClickedEventArgs 对象（该对象作为参数传递）的一部分传递给事件处理程序。可以使用此对象获取与用户所单击的超链接关联的 LinkLabel.Link 对象。LinkLabel 控件内包含的所有超链接都存储在控件的 LinkLabel.LinkCollection 类实例中。

有两种方法可将超链接添加到 LinkLabel 控件中。最快捷的方法是指定 LinkArea，并将其分配给 LinkArea 属性。这使您得以在控件的文本内指定单个超链接。若要添加多个超链接，可使用 LinkLabel.LinkCollection 类的 Add 方法，用户可以通过 Links 属性访问该集合来使用此方法。

创建 LinkLabel 控件时，将向 LinkLabel.LinkCollection 添加包含 LinkLabel 控件内的全部文本的默认超链接。可以使用 LinkArea 属性指定新的链接区域来重写此默认链接；或者使用 LinkLabel.LinkCollection 的 Add 方法指定链接。也可使用 LinkLabel.LinkCollection 类的 Remove 方法来移除默认超链接。

LinkLabel 提供许多属性，这些属性使您得以定义控件中超链接的显示外观。ActiveLinkColor、DisabledLinkColor、LinkColor 和 VisitedLinkColor 属性定义在各种状态下显示超链接时所使用的颜色。LinkBehavior 属性定义与超链接关联的下划线的显示方式。

StatusStrip 控件

通常在父窗体的底部使用有框架的区域显示有关应用程序的当前状态的信息。

表示 Windows 状态栏控件。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class StatusStrip : ToolStrip
```

备注

虽然 StatusStrip 对以前版本的 StatusBar 控件进行替换和扩展，但是考虑到向后兼容性和将来的使用（如果您选择），仍然保留了 StatusBar。

StatusStrip 控件显示关于正在 Form 上查看的对象或该对象的组件的信息，或显示与该对象在应用程序中操作相关的上下文信息。通常 StatusStrip 控件由 ToolStripStatusLabel 对象组成，其中每个对象都显示文本和/或图标。StatusStrip 还可包含 ToolStripDropDownButton、ToolStripSplitButton 和 ToolStripProgressBar 控件。

默认的 StatusStrip 没有面板。若要将面板添加到 StatusStrip，请使用 ToolStripItemCollection.AddRange 方法，或使用 StatusStrip 项集合编辑器在设计时添加、移除或重新排序项并修改属性。使用 StatusStrip 任务对话框在设计时运行常用命令。

ProgressBar 控件

向用户显示操作的当前进度。

表示 Windows 进度栏控件。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class ProgressBar : Control
```

备注

ProgressBar 控件以三种样式中的一种指示较长操作的进度：

从左向右分步递增的分段块。

从左向右填充的连续栏。

以字幕方式在 ProgressBar 中滚动的块。

Style 属性确定显示的 ProgressBar 的样式。注意，ProgressBar 控件只能是水平方向的。有关如何创建垂直方向的 ProgressBar 的示例，请参见 ProgressBarRenderer 类。ProgressBar 控件通常在应用程序执行诸如复制文件或打印文档等任务时使用。如果没有视觉提示，应用程序的用户可能会认为应用程序不响应。通过在应用程序中使用 ProgressBar，可以警告用户应用程序正在执行冗长的任务且应用程序仍在响应。

Maximum 和 Minimum 属性定义了两个值的范围用以表现任务的进度。Minimum 属性通常设置为值 0，Maximum 属性通常设置为指示任务完成的值。例如，若要正确显示复制一组文件时的进度，Maximum 属性应设置成要复制的文件的总数。

Value 属性表示应用程序在完成操作的过程中的进度。ProgressBar 显示的值仅仅是近似于 Value 属性的当前值。根据 ProgressBar 的大小，Value 属性确定何时显示下一个块或增加栏大小。

除了直接更改 Value 属性之外还有许多方式可以修改由 ProgressBar 显示的值。可以使用 Step 属性指定一个特定值用以逐次递增 Value 属性的值，然后调用 PerformStep 方法来使该值递增。若要更改增量值，可以使用 Increment 方法并指定一个用来递增 Value 属性的值。

网页显示控件

WebBrowser 控件

使用户可以在窗体内导航网页。
使用户可以在窗体中导航网页。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class WebBrowser : WebBrowserBase
```

备注

使用 WebBrowser 控件可以在 Windows 窗体应用程序中承载网页以及支持浏览器的其他文档。例如，可以使用 WebBrowser 控件在应用程序中提供基于 HTML 的集成用户帮助或 Web 浏览功能。此外，还可以使用 WebBrowser 控件向 Windows 窗体客户端应用程序添加基于 Web 的现有控件。

WebBrowser 控件不能由部分受信任的代码使用。有关更多信息，请参见通过部分受信任的代码使用库。

WebBrowser 控件具有多个与导航相关的属性、方法和事件。使用下面的成员可以将控件导航到特定 URL、在导航历史记录列表中向后和向前移动，还可以加载当前用户的主页和搜索页：

Url
Navigate
GoBack
GoForward
GoHome
GoSearch

如果导航不成功，则显示一页指示出现的问题。使用这些成员中的任何一个进行导航都会导致在导航的不同阶段发生 Navigating、Navigated 和 DocumentCompleted 事件。

使用这些成员和其他成员（如 Stop 和 Refresh 方法）可以在应用程序中实现与 Internet Explorer 中的用户界面控件类似的用户界面控件。即使不希望在窗体上显示 WebBrowser 控件，某些成员也十分有用。例如，可以使用 Print 方法打印网页的最新版本，而不向用户显示该页。

使用 WebBrowser 控件还可以显示在应用程序中创建的内容或从数据库或资源文件检索的内容。使用 DocumentText 或 DocumentStream 属性，以字符串或数据流的形式获取或设置当前文档的内容。

还可以通过 Document 属性操作网页的内容，该属性包含一个 HtmlDocument 对象，向当前页提供对 HTML 文档对象模型 (DOM) 的托管访问。该属性与 ObjectForScripting 属性组合使用时，对在应用程序代码与网页中的动态 HTML (DHTML) 代码之间实现双向通信十分有用，使用它可以在单个用户界面中组合基于 Web 的控件和 Windows 窗体控件。在应用程序中可以使用 Document 属性调用脚本代码方法。脚本代码可以通过 window.external 对象访问应用程序，该对象是用于主机访问的内置 DOM 对象，它映射到为 ObjectForScripting 属性指定的对象。

Windows Mobile for Pocket PC, Windows Mobile for Smartphone, Windows CE 平台说明：要实现 .NET Compact Framework 应用程序中的 WebBrowser 的完整功能，需要用于 Pocket PC 和 Smartphone 的 Windows Mobile 5.0 版软件。有关更多信息，请参见如何：在 .NET Compact Framework 中使用 WebBrowser 控件。

从列表中选择控件

CheckedListBox 控件

显示一个可滚动的项列表，每项旁边都有一个复选框。

显示一个 ListBox，其中在每项的左边显示一个复选框。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class CheckedListBox : ListBox
```

备注

该控件提供一个项列表，用户可以使用键盘或控件右侧的滚动条定位该列表。用户可以在一项或多项旁边放置选中标记，并且可以通过 CheckedListBox.CheckedItemCollection 和 CheckedListBox.CheckedIndexCollection 浏览选中项。

若要在运行时向列表添加对象，请用 AddRange 方法分配一个对象引用数组。然后，列表显示每个对象的默认字符串值。可以用 Add 方法向列表添加单个项。

CheckedListBox 对象通过 CheckState 枚举支持三种状态：Checked、Indeterminate 和 Unchecked。必须在代码中设置 Indeterminate 状态，因为 CheckedListBox 的用户界面未提供这样操作的机制。

如果 UseTabStops 为 true，CheckedListBox 将在某项的文本中识别并扩展制表符，从而创建列。但是，制表位已预设，无法进行更改。

CheckedListBox 类支持以下三种索引集合：

集合	封装类
CheckedListBox 控件中包含的所有项。	CheckedListBox.ObjectCollection
选中项（包括处于不确定状态的项），它是 CheckedListBox 控件中所包含项的子集。	CheckedListBox.CheckedItemCollection
选中的索引，它是项集合中索引的子集。这些索引指定处于选中状态或不确定状态的项。	CheckedListBox.CheckedIndexCollection

下面三个表是 CheckedListBox 类支持的三个索引集合的示例。

第一个表提供控件中项（控件中包含的所有项）的索引集合的示例。

索引	项	复选状态
0	对象 1	Unchecked
1	对象 2	Checked
2	对象 3	Unchecked
3	对象 4	Indeterminate
4	对象 5	Checked

第二个表提供选中项的索引集合的示例。

索引	项
0	对象 2
1	对象 4
2	对象 5

第三个表提供选中项的索引的索引集合示例。

索引	项的索引
0	1
1	3
2	4

注意 不能将数据绑定到 CheckedListBox。请改用 ComboBox 或 ListBox 绑定数据。有关更多信息，请参见如何：将 Windows 窗体 ComboBox 控件或 ListBox 控件绑定到数据。

ComboBox 控件

显示一个下拉式项列表。

表示 Windows 组合框控件。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class ComboBox : ListControl
```

备注

ComboBox 显示与一个 ListBox 组合的文本框编辑字段，使用户可以从列表中选择项，也可以输入新文本。ComboBox 的默认行为是显示一个编辑字段，该字段具有一个隐藏的下拉列表。DropDownStyle 属性确定要显示的组合框的样式。可以输入一个值，该值提供以下功能：简单的下拉列表（始终显示列表）、下拉列表框（文本部分不可编辑，并且必须选择一个箭头才能查看下拉列表框）或默认下拉列表框（文本部分可编辑，并且用户必须按箭头键才能查看列表）。若要显示用户不能编辑的列表，请使用 ListBox 控件。

若要在运行时向列表添加对象，请用 AddRange 方法分配一个对象引用数组。然后，列表显示每个对象的默认字符串值。可以用 Add 方法添加单个对象。

除了显示和选择功能外，ComboBox 还提供一些功能，使您得以有效地将项添加到 ComboBox 中以及在列表的项内查找文本。使用 BeginUpdate 和 EndUpdate 方法，可以将大量项添加到 ComboBox 中，而无需在每次将一个项添加到列表中时都重新绘制该控件。FindString 和 FindStringExact 方法使您得以在列表中搜索包含特定搜索字符串的项。

可以使用这些属性管理列表中当前选定的项，使用 Text 属性指定编辑字段中显示的字符串，使用 SelectedIndex 属性获取或设置当前项，以及使用 SelectedItem 属性获取或设置对对象的引用。

DomainUpDown 控件

显示用户可用向上和向下按钮滚动的文本项列表。

表示显示字符串值的 Windows 数字显示框（也称为 up-down 控件）。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class DomainUpDown : UpDownBase
```

备注

DomainUpDown 控件显示单个字符串值，该值是通过单击控件的向上或向下按钮从 Object 集合中选定的。除非 ReadOnly 属性设置为 true，否则用户也可以在控件中输入文本

（键入的字符串必须匹配集合中的某项才能被接受）。选中某项时，该对象将转换为一个字符串值，以便可显示在数字显示框中。

若要创建要在 DomainUpDown 控件中显示的对象的集合，可以通过使用 Add 和 Remove 方法分别添加或移除这些项。这可在事件处理程序（如按钮的 Click 事件）中调用。通过 Sorted 属性设置为 true，可按字母顺序对对象集合进行排序。当 Wrap 属性设置为 true 时，如果滚动超过了集合中最后一个或第一个对象，列表将分别从第一个或最后一个对象重新开始，但看起来是在连续的列表中滚动。

在代码中或通过单击向上或向下按钮调用 UpButton 或 DownButton 方法时，将调用 UpdateEditText 来用新字符串更新该控件。如果 UserEdit 设置为 true，则在更新该控件的文本显示之前，该字符串要与集合中的一个值匹配。

ListBox 控件

显示一个文本项和图形项（图标）列表。

表示用于显示项列表的 Windows 控件。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class ListBox : ListControl
```

备注

ListBox 控件使您得以向用户显示一列项，用户可通过单击选择这些项。ListBox 控件可使用 SelectionMode 属性提供单项选择或多重选择。ListBox 还提供 MultiColumn 属性，以启用按多列显示项而不是项的垂直列表。这样，控件便可以显示更多可见项，用户不再需要滚动到某项进行查看。

通常，Windows 处理绘制在 ListBox 中显示的项的任务。您可以使用 DrawMode 属性并处理 MeasureItem 和 DrawItem 事件，以重写 Windows 所提供的自动绘制，自己对项进行绘制。可以使用所有者描述的 ListBox 控件显示高度可变的项、图像或者为列表中每个项的文本显示不同的颜色或字体。HorizontalExtent 属性、GetItemHeight 和 GetItemRectangle 也可以帮助您绘制自己的项。

除了显示和选择功能外，ListBox 还提供一些功能，使您得以有效地将项添加到 ListBox 中以及在列表的项内查找文本。BeginUpdate 和 EndUpdate 方法使您得以将大量项添加到 ListBox 中，而不必每次将一个项添加到列表中时都重新绘制该控件。FindString 和 FindStringExact 方法使您得以在列表中搜索包含特定搜索字符串的项。

Items、SelectedItems 和 SelectedIndices 属性提供对 ListBox 所使用的三个集合的访问。下

表概述 ListBox 使用的三个集合及其在控件内的用途。

集合类	在 ListBox 内使用
ListBox.ObjectCollection	包括 ListBox 控件中包含的所有项。
ListBox.SelectedObjectCollection	包含选定项的集合，该集合是包含在 ListBox 控件中的项的子集。
ListBox.SelectedIndexCollection	包含选定索引的集合，该集合是 ListBox.ObjectCollection 的索引的子集。这些索引指定选定的项。

下面的三个示例阐释 ListBox 类支持的三个索引集合。

下表提供了一个示例，演示 ListBox.ObjectCollection 如何存储 ListBox 的项以及它们在示例 ListBox 控件中的选择状态。

索引	项	ListBox 中的选择状态
0	object1	未选定
1	object2	已选定
2	object3	未选定
3	object4	已选定
4	object5	已选定

根据上表中显示的 ListBox.ObjectCollection，此表显示 ListBox.SelectedObjectCollection 的显示方式。

索引	项
0	object2
1	object4
2	object5

根据上表中显示的 ListBox.ObjectCollection，此表显示 ListBox.SelectedIndexCollection 的显示方式。

索引	项的索引
0	1

1	3
2	4

ListBox.ObjectCollection 类的 Add 方法使您得以将项添加到 ListBox 中。当向 ListBox 添加成员时，Add 方法可接受任何对象。当向 ListBox 中添加对象时，该控件使用在该对象的 ToString 方法中定义的文本，除非在 DisplayMember 属性中指定了该对象内的成员名。除了使用 ListBox.ObjectCollection 类的 Add 方法添加项外，还可以使用 ListControl 类的 DataSource 属性添加项。

ListView 控件

在四个不同视图之一中显示项。这些视图包括纯文本视图、带有小图标的文本视图、带有大图标的文本视图和详细信息视图。

表示 Windows 列表视图控件，该控件显示可用四种不同视图之一显示的项集合。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class ListView : Control
```

备注

ListView 控件允许您显示项列表，这些项带有项文本和图标（可选）来标识项的类型。例如，Windows 资源管理器的文件列表就与 ListView 控件的外观相似。它显示树中当前选定的文件和文件夹的列表。每个文件和文件夹都显示一个与之相关的图标，以帮助标识文件或文件夹的类型。ListViewItem 类表示 ListView 控件中的项。列表中显示的项可以用五种不同视图之一显示。这些项可以显示为大图标、小图标，也可以在垂直列表中显示为小图标。这些项也可以具有子项，子项包含与父项相关的信息。详细信息视图允许您在网格中显示项及其子项，并通过列标头标识要在子项中显示的信息。平铺视图的应用范围有一定的限制（如下所述），它允许以类似贴瓷砖的方式显示项及其子项，每块“瓷砖”（平铺单元）包含一个大图标以及图标旁的文字信息。ListView 支持单选和多选。多重选择功能使用户可以按照与 ListBox 控件相似的方式从项列表中进行选择。另外，用户还可以激活选定项来执行任务。例如，可以使用 ListView 控件显示应用程序可以打开并使用的文件的列表。用户可以选择要打开的文件，然后双击它们来激活项，并在应用程序中打开文件。ListView 也可以使用 CheckBoxes 属性显示复选框，以使用户可以选中要对其执行操作的项。可以用各种方式来使用 ListView 控件。控件可用于显示来自应用程序、数据库或文本文件的信息。ListView 也

可用于获取来自用户的信息，例如选择一组要处理的文件。

ListView 提供了大量可灵活设置外观和行为的属性。View 属性允许您更改项的显示方式。LargeImageList、SmallImageList 和 StateImageList 属性允许您指定包含为项显示的图像的 ImageList 对象；并且，就 StateImageList 属性而言，当 CheckBoxes 属性设置为 true 时，它允许您指定所显示的复选框。要确定选中了哪些项，可使用 CheckedItems 属性来访问 ListView.CheckedListViewItemCollection 集合。Columns 属性允许访问 ListView.ColumnHeaderCollection，它存储了当控件的 View 属性设置为 Details 时显示的列标头。通过 Items 属性，可以在 ListView 中添加和移除项。Items 属性允许您访问控件的 ListView.ListViewItemCollection，它提供在控件中操作项的方法。如果需要允许用户编辑项的文本，可使用 LabelEdit 属性。当控件包含大量的项时，用户在经过排序的列表中查看这些项通常会更加容易。您可以使用 Sorting 属性按字母顺序对项进行排序。您也可以对 ListView 控件的外观进行全面的自定义。为实现此目的，可将 OwnerDraw 属性设置为 true，并处理以下的一个或多个事件：DrawItem、DrawSubItem 和 DrawColumnHeader。

当 ListView 控件的 View 属性设置为 Details 时，将使用该控件的许多属性。AllowColumnReorder 属性允许 ListView 控件的用户在运行时重新配置列的顺序。FullRowSelect 属性允许选择项及其子项（而不仅仅是项）。要在详细资料视图中显示网格线以标识 ListView 中的项及其子项的边界，可使用 GridLines 属性。HeaderStyle 属性允许您指定要显示的列标头的类型。

除了可用于 ListView 控件的许多属性外，应用程序还可以使用方法和事件来为 ListView 提供附加功能。通过 BeginUpdate 和 EndUpdate 方法，可在每次添加项时防止控件进行重新绘制，从而在向 ListView 添加多个项时改善性能。如果 ListView 控件显示的是项和子项，您也许会需要提供用户用鼠标右键单击子项时的功能。要确定其子项被单击的项，可使用 GetItemAt 方法。在用户编辑项后对它们执行验证时，您也许需要向用户显示要更改的特定项。可以调用 EnsureVisible 方法来确保特定项位于控件的可视区域中。

如果 LabelEdit 属性设置为 true，则可以执行如下的类似任务：通过为 BeforeLabelEdit 和 AfterLabelEdit 事件创建事件处理程序，在文本更改前后对所编辑的文本进行验证。要执行打开文件或显示对话框来编辑 ListView 中显示的项这样的任务，可以为 ItemActivate 事件创建事件处理程序。如果允许在用户单击列标头时对 ListView 中的项进行排序，则可以为 ColumnClick 事件创建事件处理程序以执行排序操作。当 CheckBoxes 属性设置为 true 时，您可以通过处理 ItemCheck 事件来确定项的选中状态何时发生更改。

还可以使用 BackgroundImage 属性设置 ListView 的背景图像。为了正确显示 ListView 控件的背景图像，您的应用程序必须对其 Main 方法应用 STAThreadAttribute。此外，如果带有背景图像的 ListView 控件寄宿在 Internet Explorer 中，请在应用程序的清单文件中将 comctl32.dll 6.0 版指定为依赖程序集，确保背景图像能够正确显示。

Windows XP 和 Windows Server 2003 提供了三种功能，以便在您的应用程序调用 Application.EnableVisualStyles 方法时增强 ListView 控件：平铺视图、分组和插入标记。

平铺视图通过在大图标旁边显示项及子项的文本，同时兼顾了图像和文字信息。通过将 View 属性设置为 View.Tile 可启用此行为。

分组功能允许您以可视化形式将项分组到相关类别之中。若要启用此功能，可使用

Groups 属性将 ListViewGroup 对象添加到 ListView 控件。若要临时禁用此功能，请将 ShowGroups 属性设置为 false。

插入标记功能通过指示放置位置，为利用拖放操作调整项的位置提供了视觉反馈信息。使用通过 InsertionMark 属性检索到的 ListViewInsertionMark 对象显示插入标记。

这些功能仅能在 Windows XP 和 Windows Server 2003 下使用。对于较早期的平台，与这些功能相关的代码不会产生任何作用，平铺视图将显示为大图标视图，而插入标记和组则不会显示。在某些情况下，您可能需要编写代码来确定是否能够使用这些功能，并在它们不可用的情况下提供相应的替代功能。提供这些功能的库与提供操作系统主题功能的库为同一个库。若要检查此库的可用性，请调用 FeatureSupport.IsPresent(Object) 方法重载并传入 OSFeature.Themes 值。

下表显示 ListView 的某些成员以及它们可用于的视图。

ListView 成员	**视图**
Alignment 属性	SmallIcon 或 LargeIcon
AutoArrange 属性	SmallIcon 或 LargeIcon
AutoResizeColumn 方法	Details
Columns 属性	Details 或 Tile
DrawSubItem 事件	Details
FindItemWithText 方法	Details、List 或 Tile
FindNearestItem 方法	SmallIcon 或 LargeIcon
GetItemAt 方法	Details 或 Tile
Groups 属性	除 List 之外的所有视图
HeaderStyle 属性	Details
InsertionMark 属性	LargeIcon、SmallIcon 或 Tile

NumericUpDown 控件

显示用户可用向上和向下按钮滚动的数字列表。

表示显示数值的 Windows 数字显示框（也称作 up-down 控件）。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class NumericUpDown : UpDownBase, ISupportInitialize
```

备注

一个包含单个数值的 NumericUpDown 控件，通过单击该控件的向上或向下按钮可使该数值递增或递减。如果未将 ReadOnly 属性设置为 true，则用户也可输入一个值。

可通过设置 DecimalPlaces、Hexadecimal 或 ThousandsSeparator 属性来设置数字的显示格式。若要在控件中显示十六进制值，请将 Hexadecimal 属性设置为 true。若要在适当的时候显示十进制数的千分隔符，请将 ThousandsSeparator 属性设置为 true。若要指定小数点后显示的位数，请将 DecimalPlaces 属性设置为要显示的小数位数。

若要指定控件允许值的范围，请设置 Minimum 和 Maximum 属性。设置 Increment 值，以指定在用户单击向上或向下箭头按钮时将向 Value 属性递增或从其递减的值。您可以通过设置 Accelerations 属性，提高用户连续按向上或向下箭头时该控件在数字间移动的速度。

当调用 UpButton 或 DownButton 方法时，不论是用代码还是通过单击向上或向下箭头键，都会对新值进行验证，而且会以适当的格式用新值对控件进行更新。具体而言，如果 UserEdit 属性设置为 true，则在验证或更新该值之前，将调用 ParseEditText 方法。然后，验证该值是否在 Minimum 和 Maximum 两个值之间，并调用 UpdateEditText 方法。

TreeView 控件

显示一个节点对象的分层集合，这些节点对象由带有可选复选框或图标的文本组成。

显示标记项的分层集合，每个标记项用一个 TreeNode 来表示。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class TreeView : Control
```

备注

Nodes 集合包含分配给 TreeView 控件的所有 TreeNode 对象。此集合中的树节点称作根树节点。随后添加到根树节点上的任何树节点称作子节点。由于每个 TreeNode 都可以包含其

他 TreeNode 对象的集合，您可能会在循环访问集合时觉得很难确定自己在树结构中的位置。您可以使用 PathSeparator 字符串值来分析 TreeNode.FullPath 字符串，以确定 TreeNode 标签的起始和终止位置。

可以在树节点旁显示图像，方法是将一个 ImageList 分配给 ImageList 属性，然后通过引用 Image 在 ImageList 中的索引值来分配该 Image。使用下面的属性分配图像：

将 ImageIndex 属性设置为当树节点未选定时所显示的 Image 的索引值。

将 SelectedImageIndex 属性设置为当树节点被选定时要显示的 Image 的索引值。

ImageIndex 和 SelectedImageIndex 属性值所引用的图像是所有分配给 Nodes 集合的树节点显示的默认图像。每个树节点都可以通过设置 TreeNode.ImageIndex 和 TreeNode.SelectedImageIndex 属性来取代默认的图像。

树节点可以展开，以显示下一级子树节点。用户可以在 TreeNode 旁显示加号 (+) 按钮时通过单击加号 (+) 按钮来展开 TreeNode，或者可以通过调用 TreeNode.Expand 方法来展开 TreeNode。若要展开 Nodes 集合中的所有子树节点级别，请调用 ExpandAll 方法。若要折叠子 TreeNode 级别，可以调用 TreeNode.Collapse 方法，也可以在 TreeNode 旁显示减号 (-) 按钮时按减号 (-) 按钮。还可以通过调用 TreeNode.Toggle 方法在展开和折叠状态之间切换。

树节点可以选择性地显示复选框。若要显示复选框，请将 TreeView 的 CheckBoxes 属性设置为 true。对于处于选中状态的树节点，Checked 属性设置为 true。

通过设置 TreeView 控件的一些显示和样式属性，可以更改此控件的外观。如果将 ShowPlusMinus 设置为 true，则会分别在每个可展开或折叠的 TreeNode 旁显示加号或减号按钮。如果将 ShowRootLines 属性设置为 true，TreeView 则会显示联接所有根树节点之间的连线。通过将 ShowLines 属性设置为 true，可以显示子树节点与其根节点之间的连线。如果将 HotTracking 属性设置为 true，那么当鼠标指针移过树节点标签时，树节点标签的外观将发生变化。如果启用热跟踪，树节点标签将具有超链接的外观。也可以完全自定义 TreeView 控件的外观。若要执行此操作，请将 DrawMode 属性设置为 TreeViewDrawMode.Normal 以外的值并处理 DrawNode 事件。

图形显示控件

PictureBox 控件

在一个框架中显示图形文件（如位图和图标）。
表示用于显示图像的 Windows 图片框控件。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class PictureBox : Control, ISupportInitialize
```

备注

通常使用 PictureBox 来显示位图、元文件、图标、JPEG、GIF 或 PNG 文件中的图形。

在设计时或运行时将 Image 属性设置为要显示的 Image。也可以通过设置 ImageLocation 属性指定图像，然后使用 Load 方法同步加载图像或使用 LoadAsync 方法异步加载图像。SizeMode 属性（设置为 PictureBoxSizeMode 枚举中的值）控制图像在显示区域中的剪裁和定位。可以在运行时使用 ClientSize 属性来更改显示区域的大小。

默认情况下，PictureBox 控件在显示时没有任何边框。即使图片框不包含任何图像，仍可以使用 BorderStyle 属性提供一个标准或三维的边框，以便使图片框与窗体的其余部分区分。PictureBox 不是可选择的控件，这意味着该控件不能接收输入焦点。

图形存储控件

ImageList 控件

充当图像储存库。ImageList 控件和及其包含的图像可以在不同的应用程序中重用。
提供管理 Image 对象集合的方法。无法继承此类。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public sealed class ImageList : Component
```

备注

ImageList 通常由其他控件使用，如 ListView、TreeView 或 ToolBar。可以将位图、图标添加到 ImageList 中，且其他控件能够在需要时使用这些图像。

ImageList 使用句柄管理图像列表。直到在图像列表上执行某些操作（包括获取 Count、获取 Handle 和调用 Draw）时才创建 Handle。

值的设置控件

CheckBox 控件

显示一个复选框和一个文本标签。通常用来设置选项。

表示 Windows CheckBox。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class CheckBox : ButtonBase
```

备注

使用 CheckBox 可为用户提供一项选择，如“真/假”或“是/否”。该 CheckBox 控件可以显示一个图像或文本，或两者都显示。

CheckBox 和 RadioButton 控件拥有一个相似的功能：允许用户从选项列表中进行选择。CheckBox 控件允许用户选择一组选项。与之相反，RadioButton 控件允许用户从互相排斥的选项中进行选择。

Appearance 属性确定 CheckBox 显示为常见的 CheckBox 还是显示为按钮。

ThreeState 属性确定该控件是支持两种状态还是三种状态。使用 Checked 属性可以获取或设置具有两种状态的 CheckBox 控件的值，而使用 CheckState 属性可以获取或设置具有三种状态的 CheckBox 控件的值。

FlatStyle 属性确定控件的样式和外观。如果 FlatStyle 属性设置为 FlatStyle.System，则控件的外观由用户的操作系统确定。

下面描述一种不确定状态：有一个用于确定 RichTextBox 中选定的文本是否为粗体的 CheckBox。选择文本时，您可以单击 CheckBox 以将选定文本变成粗体。同样，选择一些文本时，CheckBox 将显示选定的文本是否为粗体。如果选定的文本包含粗体和常规文本，则 CheckBox 将处于不确定状态。

CheckedListBox 控件

显示一个可滚动的项列表，每项旁边都有一个复选框。

RadioButton 控件

显示一个可打开或关闭的按钮。

当与其他 RadioButton 控件成对出现时，使用户能够从一组选项中选择一个选项。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class RadioButton : ButtonBase
```

备注

RadioButton 控件可以显示文本、Image 或同时显示两者。

当用户选择一个组内的一个选项按钮（也称作单选按钮）时，其他选项按钮自动清除。给定容器（如 Form）中的所有 RadioButton 控件构成一个组。若要在一个窗体上创建多个组，请将每个组放在它自己的容器（例如 GroupBox 或 Panel 控件）中。

RadioButton 和 CheckBox 控件的功能相似：它们提供用户可以选择或清除的选项。不同之处在于，可以同时选定多个 CheckBox 控件，而选项按钮却是互相排斥的。

使用 Checked 属性可以获取或设置 RadioButton 的状态。通过设置 Appearance 属性，可以将选项按钮的外观显示为切换式按钮或标准选项按钮。

TrackBar 控件

允许用户通过沿标尺移动“滚动块”来设置标尺上的值。

表示一个标准的 Windows 跟踪条。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class TrackBar : Control, ISupportInitialize
```

备注

TrackBar 是类似于 ScrollBar 控件的可滚动控件。可以通过以下方式配置跟踪条的 Value 属性值滚动的范围：通过设置 Minimum 属性指定该范围的下限，设置 Maximum 属

性指定该范围的上限。

LargeChange 属性定义在滚动框的任一侧单击鼠标时对 Value 属性进行增减的量。跟踪条可以水平显示或垂直显示。

可以使用此控件输入通过 Value 属性获取的数值型数据。可以在一个控件中显示此数值型数据，或者在代码中使用此数据。

数据的设置控件

DateTimePicker 控件

显示一个图形日历以允许用户选择日期或时间。

表示一个 Windows 控件，该控件用来让用户选择日期和时间并以指定的格式显示此日期和时间。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class DateTimePicker : Control
```

备注

DateTimePicker 控件用来让用户选择日期和时间，并以指定的格式显示此日期和时间。通过设置 MinDate 和 MaxDate 属性，可以限制可选择的日期和时间。

通过设置 CalendarForeColor、CalendarFont、CalendarTitleBackColor、CalendarTitleForeColor、CalendarTrailingForeColor 和 CalendarMonthBackground 属性，可以更改控件日历部分的外观。

Format 属性设置控件的 DateTimePickerFormat。默认日期 Format 为 DateTimePickerFormat.Long。如果 Format 属性设置为 DateTimePickerFormat.Custom，可以通过设置 CustomFormat 属性并生成自定义格式字符串来创建自己的格式化样式。自定义格式字符串可以是自定义字段字符和其他字符的组合。例如，通过将 CustomFormat 属性设置为“MMMM dd, yyyy - dddd”，可以将日期显示为“June 01, 2001 - Friday”。有关更多信息，请参见日期与时间格式字符串。

若要使用数值调节钮控件（也称为 up-down 控件）调整日期/时间值，请将 ShowUpDown 属性设置为 true。日历控件被选定后将不下拉。可以通过分别选择各元素并使用向上和向下按钮更改值来调整日期和时间。

如果需要自定义日期格式设置（例如，选择限制为只使用一个日期），您可以考虑使用 DateTimePicker 控件，而不是 MonthCalendar。使用 DateTimePicker 可限制对日期/时间值进行大量数据验证的需要。

MonthCalendar 控件

显示一个图形日历以允许用户选择日期范围。

表示一个 Windows 控件，该控件使用户能够使用可视月历显示来选择日期。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class MonthCalendar : Control
```

备注

MonthCalendar 控件允许用户使用可视显示来选择日期。通过设置 MinDate 和 MaxDate 属性，可以限制可选择的日期和时间。

通过设置 ForeColor、Font、TitleBackColor、TitleForeColor、TrailingForeColor 和 BackColor 属性，可以更改控件日历部分的外观。

MonthCalendar 控件由操作系统绘制，因此决不会引发 Paint 事件。如果需要为 MonthCalendar 控件提供自定义的外观，则应重写 OnPrint 方法，调用 OnPrint 的基实现，然后再执行自定义绘图。

如果需要将自定义日期格式和选择限制为只用于一个日期，您可以考虑使用 DateTimePicker 控件，而不是 MonthCalendar。使用 DateTimePicker 可避免许多需要验证日期 / 时间值的情况。

对话框控件

ColorDialog 控件

显示允许用户设置界面元素的颜色的颜色选择器对话框。

表示一个通用对话框，该对话框显示可用的颜色以及允许用户定义自定义颜色的控件。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public class ColorDialog : CommonDialog
```

备注

必须调用继承的成员 ShowDialog 才能创建此特定的通用对话框。可重写 HookProc 以实现特定的对话框挂钩功能。使用 Color 可检索用户选定的颜色。

在创建 ColorDialog 的实例时，一些读/写属性被设置为初始值。有关这些值的列表，请参见 ColorDialog 构造函数。

FontDialog 控件

显示允许用户设置字体及其属性的对话框。

提示用户从本地计算机上安装的字体中选择一种字体。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public class FontDialog : CommonDialog
```

备注

必须调用继承的成员 ShowDialog 才能创建此特定的通用对话框。可重写 HookProc 以实现特定的对话框挂钩功能。

在创建 FontDialog 的实例时，一些读/写属性被设置为初始值。

OpenFileDialog 控件

显示允许用户定位文件和选择文件的对话框。提示用户打开文件。无法继承此类。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public sealed class OpenFileDialog : FileDialog
```

备注

使用此类可检查某个文件是否存在并打开该文件。ShowReadOnly 属性确定是否在对话框中显示只读复选框。ReadOnlyChecked 属性指示是否选中只读复选框。

此类的大多数功能都可以在 FileDialog 类中找到。

如果要使用户能够选择文件夹而不是文件，请改用 FolderBrowserDialog。

PrintDialog 控件

显示允许用户选择打印机并设置其属性的对话框。允许用户选择一台打印机并选择文档中要打印的部分。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public sealed class PrintDialog : CommonDialog
```

备注

创建 PrintDialog 的实例时，读/写属性将被设置为初始值。有关这些值的列表，请参见 PrintDialog 构造函数。

PrintPreviewDialog 控件

显示一个对话框，该对话框显示 PrintDocument 组件在打印出来后的外观。

表示包含 PrintPreviewControl 的对话框窗体。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class PrintPreviewDialog : Form
```

备注

在创建 PrintPreviewDialog 类的实例时，一些读 / 写属性被设置为初始值。有关这些值的列表，请参见 PrintPreviewDialog 构造函数。

FolderBrowserDialog 控件

显示用来浏览、创建以及最终选择文件夹的对话框，提示用户选择文件夹。无法继承此类。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public sealed class FolderBrowserDialog : CommonDialog
```

备注

此类提供一种方法，它提示用户浏览、创建并最终选择一个文件夹。如果只允许用户选择文件夹而非文件，则可使用此类。文件夹的浏览通过树控件完成。只能选择文件系统中的文件夹；不能选择虚拟文件夹。

通常在创建新 FolderBrowserDialog 后，将 RootFolder 设置为开始浏览的位置。或者，可将 SelectedPath 设置为最初选定的 RootFolder 子文件夹的绝对路径。也可以选择设置 Description 属性为用户提供附加说明。最后，调用 ShowDialog 方法将对话框显示给用户。如果该对话框关闭并且 ShowDialog 显示的对话框为 DialogResult.OK，SelectedPath 则是一个包含选定文件夹路径的字符串。

如果用户可通过“新建文件夹”按钮创建新文件夹，则可对控件使用 ShowNewFolderButton 属性。

FolderBrowserDialog 是有模式对话框，因此，在显示时，它会阻止应用程序剩余部分的运行，直到用户选定了文件夹。当对话框有模式显示时，不能进行任何输入（通过键盘或鼠标单击），对对话框上的对象的输入除外。在对调用程序进行输入之前，程序必须隐藏或关闭对话框（通常是响应某一用户操作）。

SaveFileDialog 控件

显示允许用户保存文件的对话框。

提示用户选择文件的保存位置。无法继承此类。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public sealed class SaveFileDialog : FileDialog
```

备注

此类可以打开和改写现有文件，也可以创建新文件。

此类的大多数功能都可以在 FileDialog 类中找到。

Windows Mobile for Pocket PC, Windows Mobile for Smartphone, Windows CE 平台说明：在 Pocket PC 上，如果不指定文件的扩展名，控件会附加对话框中选定类型的扩展名。在 Windows CE 上，控件不附加扩展名。所有平台都支持 FilterIndex 属性，该属性返回选定扩展名筛选器的索引。

菜单控件

MenuStrip 控件

创建自定义菜单
提供窗体的菜单系统。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class MenuStrip : ToolStrip
```

备注

MenuStrip 控件表示窗体菜单结构的容器。您可以将 ToolStripMenuItem 对象添加到表示菜单结构中各菜单命令的 MenuStrip 中。每个 ToolStripMenuItem 可以成为应用程序的命令或其他子菜单项的父菜单。

MenuStrip 是 ToolStripMenuItem 、 ToolStripComboBox 、 ToolStripSeparator 和 ToolStripTextBox 对象的容器。

虽然 MenuStrip 对以前版本的 MainMenu 控件的功能进行了替换和添加，但是考虑到向后兼容性和将来的使用（如果您选择），仍然保留了 MainMenu。

ContextMenuStrip 控件

创建自定义上下文菜单。
表示快捷菜单。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class ContextMenuStrip : ToolStripDropDownMenu
```

备注

ContextMenuStrip 类表示快捷菜单，这些快捷菜单在用户在窗体中的控件或特定区域上单击鼠标右键时显示。快捷菜单通常用于组合来自窗体的一个 MenuStrip 的不同菜单项，便于用户在给定应用程序上下文中使用。例如，可以使用分配给 TextBox 控件的快捷菜单提供菜单项，以便更改文本字体，在控件中查找文本或实现复制和粘贴文本的剪贴版功能。还可以在快捷菜单中显示不位于 MenuStrip 中的新的 ToolStripMenuItem 对象，从而提供与特定情况有关且不适合在 MenuStrip 中显示的命令。

当用户在控件或窗体本身上单击鼠标右键时，通常会显示快捷菜单。许多可视控件（以及 Form 本身）都有一个 Control.ContextMenuStrip 属性，该属性将 ContextMenuStrip 类绑定到显示快捷菜单的控件。多个控件可使用一个 ContextMenuStrip。

将 ToolStripDropDownMenu.ShowCheckMargin 属性设置为 true 可向 ToolStripMenuItem 的左侧添加用以容纳选中标记的空间，选中标记显示是否启用或选择了该菜单项。ToolStripDropDownMenu.ShowImageMargin 属性默认被设置为 true。使用 ToolStripMenuItem 左侧的此空间可以为菜单项显示一个图像。

ContextMenuStrip 是 ToolStripMenuItem、ToolStripComboBox、ToolStripSeparator 和 ToolStripTextBox 对象的容器。

虽然 ContextMenuStrip 对以前版本的 ContextMenu 控件的功能进行了替换和添加，但是考虑到向后兼容性和将来的使用（如果的确需要），仍然保留了 ContextMenu。

命令控件

Button 控件

启动、停止或中断进程。
表示 Windows 按钮控件。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class Button : ButtonBase, IButtonControl
```

备注

如果某个 Button 具有焦点，则可以使用鼠标、Enter 键或空格键单击该按钮。

设置 Form 的 AcceptButton 或 CancelButton 属性，使用户能够通过按 Enter 或 Esc 键来单击按钮（即使该按钮没有焦点）。这使该窗体具有对话框的行为。

当使用 ShowDialog 方法显示一个窗体时，可以使用按钮的 DialogResult 属性指定 ShowDialog 的返回值。

可以更改按钮的外观。例如，要使它显示为 Web 风格的平面外观，请将 FlatStyle 属性设置为 FlatStyle.Flat。FlatStyle 属性还可设置为 FlatStyle.Popup；当鼠标指针经过该按钮时，它不再是平面外观，而是将呈现标准的 Windows 按钮外观。

LinkLabel 控件

将文本显示为 Web 样式的链接，并在用户单击该特殊文本时触发事件。该文本通常是到另一个窗口或网站的链接。

NotifyIcon 控件

在表示正在后台运行的应用程序的任务栏的状态通知区域中显示一个图标。
指定在通知区域中创建图标的组件。无法继承此类。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public sealed class NotifyIcon : Component
```

备注

通知区域中的图标是一些进程的快捷方式，这些进程在计算机后台运行，如防病毒程序或音量控制。这些进程不会具有自己的用户界面。NotifyIcon 类提供了编写此功能的方法。Icon 属性定义显示在通知区域中的图标。图标的弹出菜单由 ContextMenu 属性确定。Text 属性分配工具提示文本。要在通知区域中显示图标，必须将 Visible 属性设置为 true。

ToolStrip 控件

创建工具栏，这些工具栏可以具有与 Microsoft Windows XP、Microsoft Office 或 Microsoft Internet Explorer 类似的外观，也可以具有自定义外观，可以有主题，也可以没有主题，并支持溢出和运行时项重新排序。

为 Windows 工具栏对象提供容器。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class ToolStrip : ScrollableControl, IComponent, IDisposable
```

备注

在新的 Windows 窗体应用程序中可使用 ToolStrip 及其相关的类来创建工具栏，这些工具栏具有 Windows XP、Office、Internet Explorer 或自定义的外观和行为（它们均既可以使用主题又可以不使用主题，均支持溢出和在运行时对项重新排序）。ToolStrip 控件也提供丰富的设计时体验，包括就地激活和编辑、自定义布局以及共享指定的 ToolStripContainer 内的水平空间或垂直空间。

ToolStrip 是 ToolStripButton、ToolStripComboBox、ToolStripSplitButton、ToolStripLabel、ToolStripSeparator、ToolStripDropDownButton、ToolStripProgressBar 和 ToolStripTextBox 对象的容器。

尽管 ToolStrip 类提供了许多可管理绘制、鼠标和键盘输入以及拖放功能的成员，但是您可以使用 ToolStripManager 类在指定的 ToolStripContainer 内联接 ToolStrip 控件，以及将 ToolStrip 控件相互合并。通过将 ToolStripRenderer 类和 ToolStripManager 类结合使用，可以获得对绘制样式和布局样式的更好控制和更多的自定义功能。

使用 ToolStripControlHost 类可以在 ToolStrip 中承载任何其他 Windows 窗体控件。

虽然 ToolStrip 对以前版本的 ToolBar 控件的功能进行了替换和增补，但是考虑到向后兼容性和将来的使用（如果您选择），仍然保留了 ToolBar。

用户帮助控件

HelpProvider 组件

为控件提供弹出式帮助或联机帮助。

提供控件的弹出或联机帮助。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
ublic class HelpProvider : Component, IExtenderProvider
```

备注

每个 HelpProvider 实例均维护一个对关联控件的引用的集合。若要使帮助文件与 HelpProvider 关联，请设置 HelpNamespace 属性。通过调用 SetHelpNavigator 方法并提供指定控件的 HelpNavigator 值来指定提供的帮助类型。通过调用 SetHelpKeyword 方法为帮助提供关键字或主题。若要打开特定主题的帮助，则应以 topicName.htm 的形式传入关键字。

若要使特定的帮助字符串与控件关联，请使用 SetHelpString 方法。如果用户在控件包含焦点时按下 F1 键，使用此方法与控件关联的字符串将显示在弹出窗口中。

如果尚未设置 HelpNamespace 属性，则必须使用 SetHelpString 方法提供帮助文本。如果同时设置了 HelpNamespace 和帮助字符串，则基于 HelpNamespace 的帮助信息优先。

HelpProvider 在 Help 类上调用方法来提供帮助功能。

ToolTip 组件

当用户将指针停留在控件上时，提供一个弹出式窗口来显示该控件的用途的简短说明。

表示一个长方形的小弹出窗口，该窗口在用户将指针悬停在一个控件上时显示有关该控件用途的简短说明。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
public class ToolTip : Component, IExtenderProvider
```

备注

使用 ToolTip 类，您可在用户将指针放置在控件上时为用户显示提示信息。ToolTip 类通常用来向用户提示控件的预期用途。例如，可以为接受名称的 TextBox 控件指定工具提示文本，同时指定要键入到控件中的名称的格式。除了提供提示外，还可使用 ToolTip 类提供运行时状态信息。例如，当用户将指针移动到显示 Internet 连接状态的 PictureBox 控件上时，可以使用 ToolTip 类显示连接速度和线路质量数据。

ToolTip 类可在任何容器内使用。若要显式指定容器，请使用 ToolTip(IContainer) 构造函数。单个 ToolTip 组件通常用于为单个窗体上的多个控件创建工具提示。在创建 ToolTip 之后，使用不同的 SetToolTip 方法调用可将工具提示显示文本与每个控件关联。然后，当用户将指针移到控件上时，即会显示工具提示及其文本。对于同一个控件可多次调用 SetToolTip 以更改与控件关联的文本。若要获取与控件关联的文本，请使用 GetToolTip 方法。若要移除 ToolTip 类的某个实例的所有工具提示文本关联，请使用 RemoveAll 方法。

ToolTip 类提供了以下属性和方法以修改工具提示的默认行为和外观。

类别	关联成员
手动显示	Active, Show, Hide, ShowAlways, Popup, StopTimer
工具提示计时	AutoPopDelay, InitialDelay, ReshowDelay, AutomaticDelay, StopTimer
内容	SetToolTip, GetToolTip, StripAmpersands, ToolTipIcon, ToolTipTitle, RemoveAll
外观	BackColor, ForeColor, IsBalloon, OwnerDraw, UseAnimation, UseFading

如果要禁用所有工具提示文本以便不在应用程序中显示它，可以使用 Active 属性。工具提示通常由操作系统绘制，但若要自定义 ToolTip 的外观，可将 OwnerDraw 属性设置为 true 并处理 Draw 事件。

ToolTipTitle 类实现 System.ComponentModel.IExtenderProvider 接口，该接口只有一个 CanExtend 方法。工具提示在设计时扩展同一窗体上的控件，添加一个 ToolTip 属性。有关扩展程序提供程序的更多信息，请参见扩展程序提供程序。

分组控件

Panel 控件

将一组控件分组到未标记、可滚动的框架中。
用于对控件集合进行分组。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class Panel : ScrollableControl
```

备注

Panel 是一个包含其他控件的控件。可以使用 Panel 来组合控件的集合，例如一组 RadioButton 控件。与其他容器控件（如 GroupBox 控件）一样，如果 Panel 控件的 Enabled 属性设置为 false，则也会禁用包含在 Panel 中的控件。

默认情况下，Panel 控件在显示时没有任何边框。可以用 BorderStyle 属性提供标准或三维的边框，将窗面板区与窗体上的其他区域区分开。因为 Panel 控件派生于 ScrollableControl 类，所以可以用 AutoScroll 属性来启用 Panel 控件中的滚动条。当 AutoScroll 属性设置为 true 时，使用所提供的滚动条可以滚动显示 Panel 中（但不在其可视区域内）的所有控件。

Panel 控件不显示标题。如果需要与 Panel 类似可显示标题的控件，请参见 GroupBox 控件。

GroupBox 控件

将一组控件（如单选按钮 (RadioButton)）分组到带标记、不可滚动的框架中。
表示一个 Windows 控件，该控件显示围绕一组具有可选标题的控件的框架。
命名空间：System.Windows.Forms
程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
[ComVisibleAttribute(true)]
public class GroupBox : Control
```

备注

GroupBox 显示围绕一组控件的框架（带或不带标题）。使用 GroupBox 对窗体上的控件集合进行逻辑分组。组框是可用于定义控件组的容器控件。

组框的典型用途是包含 RadioButton 控件的逻辑组。如果有两个分组框，每个分组框都包含多个选项按钮（也称为单选按钮），每组按钮都互相排斥，则每组设置一个选项值。

通过使用 Controls 属性的 Add 方法，可将控件添加到 GroupBox。

GroupBox 不能显示滚动条。如果需要可包含滚动条的类似于 GroupBox 的控件，请参见 Panel 控件。

TabControl 控件

提供一个选项卡式页面以有效地组织和访问已分组对象。

管理相关的选项卡页集。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class TabControl : Control
```

备注

TabControl 包含选项卡页，这些选项卡页由通过 TabPages 属性添加的 TabPage 对象表示。此集合中的选项卡页的顺序反映了选项卡在控件中出现的顺序。

用户可以通过单击控件中的某一选项卡来更改当前的 TabPage。您也可以通过使用下面的 TabControl 属性之一，以编程的方式更改当前的 TabPage：

SelectedIndex

SelectedTab

在 Microsoft .NET Framework 2.0 版 中，也可以使用下面的方法之一：

SelectTab

DeselectTab

在 .NET Framework 2.0 中，可以通过处理下面的事件之一，在当前选项卡发生更改时进行响应：

Deselecting

Deselected
Selecting
Selected

TabControl 中的选项卡是 TabControl 的一部分，但不是各个 TabPage 控件的一部分。TabPage 类的成员（例如 ForeColor 属性）只影响选项卡页的矩形工作区，而不影响选项卡。此外，TabPage 的 Hide 方法不会隐藏选项卡。若要隐藏选项卡，必须从 TabControl.TabPages 集合中移除 TabPage 控件。

SplitContainer 控件

提供用可移动拆分条分隔的两个面板。

表示一个由可移动条组成的控件，该可移动条将容器的显示区域分成两个大小可调的面板。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class SplitContainer : ContainerControl
```

备注

您可以将控件添加到两个大小可调的面板中，并将其他 SplitContainer 控件添加到现有 SplitContainer 面板中，以创建多个大小可调的显示区域。

使用 SplitContainer 控件可分隔容器（如 Form）的显示区域，并使用户可以调整已添加到 SplitContainer 面板中的控件的大小。当用户将鼠标指针移到拆分器上时，光标将发生变化，以指示可以调整 SplitContainer 控件内部的控件的大小。

SplitContainer 也使设计时的控件放置更容易。例如，若要创建一个与 Windows 资源管理器类似的窗口，可将 SplitContainer 控件添加到一个 Form 中，并将其 Dock 属性设置为 DockStyle.Fill。同时，将 TreeView 控件添加到 Form 并将其 Dock 属性设置为 DockStyle.Fill。若要完成布局，则添加一个 ListView 控件并将其 Dock 属性设置为 DockStyle.Fill，从而使 ListView 占据 Form 上的剩余空间。在运行时，用户可以使用拆分器调整两个控件的宽度。使用 FixedPanel 属性可指定某个控件不应随 Form 或其他容器一起调整大小。

使用 SplitterDistance 可指定拆分器开始的时候位于窗体上的位置。使用 SplitterIncrement 可指定拆分器一次移动多少像素。SplitterIncrement 的默认值是一个像素。

使用 Panel1MinSize 和 Panel2MinSize 可指定拆分器条能够移到靠近 SplitContainer 面板外边缘的程度。面板默认的最小大小值为 25 像素。

使用 Orientation 属性可指定水平方向。SplitContainer 的默认方向为垂直方向。

使用 BorderStyle 属性可指定 SplitContainer 的边框样式，并使其边框样式与添加到 SplitContainer 的控件的边框样式协调。

TableLayoutPanel 控件

表示一个面板，它可以在一个由行和列组成的网格中对其内容进行动态布局。

表示一个面板，它可以在一个由行和列组成的网格中对其内容进行动态布局。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class TableLayoutPanel : Panel, IExtenderProvider
```

备注

TableLayoutPanel 控件在网格中排列其内容。因为布局既可以在设计时执行，也可以在运行时执行，所以它会随应用程序环境的更改而动态更改。这样，面板中的控件可以适当地调整大小，从而对各种更改做出响应，如父控件的大小调整或者由于本地化带来的文本长度的更改。

任何 Windows 窗体控件都可以是 TableLayoutPanel 控件的子级，包括 TableLayoutPanel 的其他实例。这样，您就可以构造能够适应运行时的各种更改的复杂布局。

TableLayoutPanel 控件可以根据 RowCount、ColumnCount 和 GrowStyle 属性的值进行扩展，以容纳新添加的控件。将 RowCount 或 ColumnCount 属性的值设置为 0，将指定 TableLayoutPanel 在相应方向上取消绑定。

当 TableLayoutPanel 控件充满子控件以后，您也可以控制扩展的方向（水平或垂直）。默认情况下，TableLayoutPanel 控件通过添加行向下扩展。

如果希望行和列采取与默认行为不同的行为方式，可以通过使用 RowStyles 和 ColumnStyles 属性来控制行和列的属性。可以分别设置行或列的属性。

TableLayoutPanel 控件向其子控件添加以下属性：Cell、Column、Row、ColumnSpan 和 RowSpan。

可以通过设置子控件上的 ColumnSpan 或 RowSpan 属性，来合并 TableLayoutPanel 控件中的单元格。

子控件的停靠行为与其他容器控件相同。

TableLayoutPanel 中子控件的锚定行为与其他容器控件的行为不同。如果将子控件的 Anchor 属性值设置为 Left 或 Right，则将控件靠单元格左边框或由边框放置，距离边框的距离是控件的 Margin 属性和面板的 Padding 属性之和。如果同时设置了 Left 和 Right

值，控件将根据单元格的宽度调整大小，同时考虑了 Margin 和 Padding 值。Top 和 Bottom 定锚的行为与此类似。有关更多信息，请参见如何：在 TableLayoutPanel 控件中锚定和停靠子控件。

如果需要子控件模仿其他容器控件的默认定锚行为，可以调整 Margin 和 Padding 属性，以使控件的边框与单元格的边框保持固定的距离。

运行时，将一个子控件的 Column 和 Row 属性值设置为 -1，将使该控件移动到 TableLayoutPanel 控件的第一个空单元格中。空单元格将按从左到右、从上到下的搜索顺序选中。此顺序依赖于区域性，所以在从右向左 (RTL) 的布局中它将正确运行。设计时，将这些属性设置为 -1 时，子控件不会移动。

FlowLayoutPanel 控件

表示一个沿水平或垂直方向动态排放其内容的面板。

表示一个沿着水平或垂直方向动态排放其内容的面板。

命名空间：System.Windows.Forms

程序集：System.Windows.Forms（在 system.windows.forms.dll 中）

语法

```
[ComVisibleAttribute(true)]
[ClassInterfaceAttribute(ClassInterfaceType.AutoDispatch)]
public class FlowLayoutPanel : Panel, IExtenderProvider
```

备注

FlowLayoutPanel 控件沿着水平或垂直流向排列其内容。它的内容可以从一行换到下一行或从一列换到下一列。或者，还可以对它的内容进行剪裁，而不是进行换行。

可以通过设置 FlowDirection 属性的值来指定流向。在从右向左 (RTL) 的布局中，FlowLayoutPanel 控件会相应地反转其流向。还可以通过设置 WrapContents 属性的值来指定是对 FlowLayoutPanel 控件的内容进行换行还是剪裁。

任何 Windows 窗体控件（包括 FlowLayoutPanel 的其他实例）都可以是 FlowLayoutPanel 控件的子级。使用此功能，可以构造在运行时能够根据窗体的尺寸进行相应调整的复杂布局。*

子控件的停靠和锚定行为与其他容器控件的行为不同。停靠和锚定行为均相对于流向中的最大控件。有关更多信息，请参见如何：在 FlowLayoutPanel 控件中锚定和停靠子控件。

音频控件

SoundPlayer 控件

播放 .wav 格式的声音文件。加载声音和播放声音可以异步进行。
控制 .wav 文件中的声音播放。
命名空间：System.Media
程序集：System（在 system.dll 中）

语法

```
[SerializableAttribute]
public class SoundPlayer : Component, ISerializable
```

备注

SoundPlayer 类提供了加载和播放 .wav 文件的简单界面。SoundPlayer 类支持从文件路径、URL、包含 .wav 文件的 Stream 或包含 .wav 文件的嵌入资源中加载 .wav 文件。

要使用 SoundPlayer 类播放声音，请用 .wav 文件的路径配置 SoundPlayer 并调用某个播放方法。可以使用某个构造函数或通过设置 SoundLocation 或 Stream 属性来标识要播放的 .wav 文件。可以在播放前使用某个加载方法加载文件，或者将加载推迟到调用某个播放方法时。被配置为从 Stream 或 URL 中加载 .wav 文件的 SoundPlayer 必须在播放开始前将 .wav 文件加载到内存中。

可以同步或异步地加载或播放 .wav 文件。如果调用同步加载或播放方法，调用线程将一直等到方法返回，这可能会导致绘制和其他事件中断。调用异步加载或播放方法则允许调用线程继续执行，而不会中断。有关异步方法调用的更多信息，请参见如何：在后台运行操作。

当 SoundPlayer 加载完 .wav 文件后，它会引发 LoadCompleted 事件。可以检查事件处理程序中的 AsyncCompletedEventArgs，确定加载是成功还是失败。当音频源设置为新文件路径或 URL 时，引发 SoundLocationChanged 事件。当音频源设置为新 Stream 时，引发 StreamChanged 事件。