第3章 Windows 窗体与控件

Windows 窗体是以.NET Framework 为基础的一个新平台，主要用来开发 Windows 窗体应用程序（简称 Windows 应用程序）。一个 Windows 应用程序通常由窗体对象和控件对象构成，即使开发一个最简单的 Windows 应用程序，也必须了解窗体对象和控件对象的使用。

本章主要介绍 Windows 窗体的结构和常用属性、方法与事件，以及 Label、LinkLabel、TextBox、Button 几种常用控件的使用。

3.1 窗体

窗体（Form）就是平常所说的窗口，各种控件对象必须建立在窗体上。窗体对象是 Visual C#应用程序的基本构造模块，是运行 Windows 应用程序时与用户交互操作的实际窗口。窗体有自己的属性、方法和事件，用于控制其外观和行为。

3.1.1 窗体的结构

窗体是包含所有组成程序用户界面的其他控件的对象。在创建 Windows 应用程序项目时，Visual Studio 2012 会自动提供一个窗体，其组成结构如图 3.1 所示。

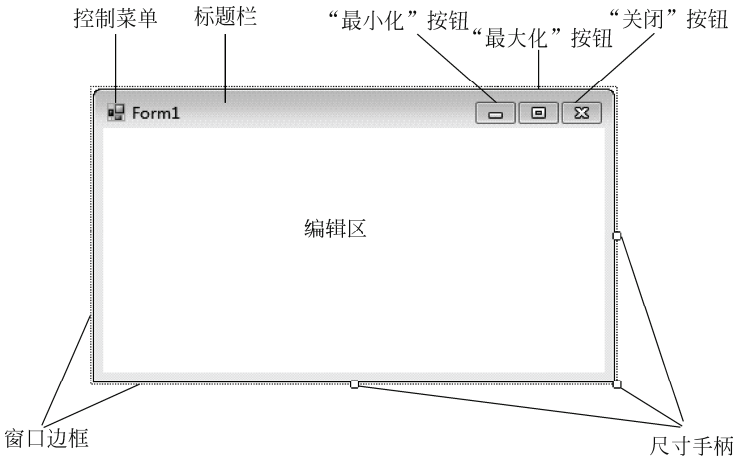


图 3.1 窗体的结构

窗体的结构与 Windows 的标准窗口一样，包含有控制菜单、标题栏、控制按钮、编辑区和窗口边框。

1．控制菜单

控制菜单是 Visualc#.NET 固有的一个菜单，在程序运行时，单击窗体左上角的图标将会显示该菜单。一般包含还原、移动、大小、最小化、最大化、关闭等菜单项。

2．标题栏

标题栏显示窗体的标题，标题一般为应用程序的名称。在创建 Windows 应用程序时，Visual Studio 2012 会将窗体的标题栏设置为 Form1。

3．控制按钮

控制按钮一般包括“最小化”按钮、“最大化”/“还原”按钮、“关闭”按钮。在程序运行时，单击“最小化”按钮可以把窗体最小化到任务栏成为一个按钮，单击“关闭”按钮则关闭窗体。单击“最大化”按钮可以使窗体扩大至整个屏幕，此时该按钮变为“还原”按钮，再次单击该按钮，可以使窗体恢复至初始状态。

4．编辑区

窗体的编辑区占据了窗口的大部分，是容纳控件对象的区域。在程序的设计模式下，可以编辑控件对象；在程序运行时，可以操作控件对象与程序进行交互。

5．窗口边框

在程序运行时，当鼠标指针指向窗口边框时，鼠标指针会变为双向箭头，拖动鼠标指针可以改变窗体大小。在程序的设计模式下，当鼠标指针指向尺寸手柄时，鼠标指针也会变为双向箭头，拖动鼠标指针可以改变窗体大小。

在创建 Windows 应用程序时，Visual Studio 2012 会将窗体文件命名为 Form1.cs（图 3.2），建议编程人员将其改为能够描述程序用途的名称。

在“解决方案资源管理器”中选择 Form1.cs，在“属性”窗口中显示出相应文件属性，在“文件名”属性框的右侧区域输入新的文件名即可。也可以直接在“解决方案资源管理器”中右击 Form1.cs，在弹出的快捷菜单中选择“重命名”选项，输入新的文件名即可。

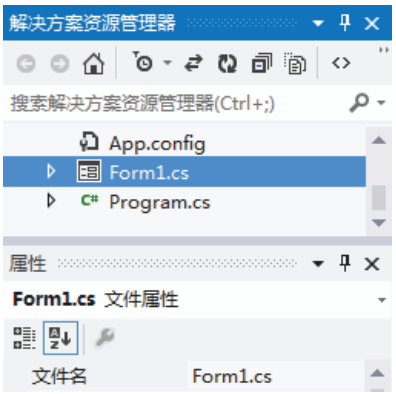


图 3.2　改变窗体的文件名

提示：

将应用程序的窗体文件名改为能够描述程序用途的名称，是一个良好的编程习惯。

3.1.2 窗体的属性

窗体有一些表现其特征的属性，可以通过设置这些属性控制窗体的外观。窗体的主要属性如表 3.1 所示。

表 3.1 窗体的主要属性

属　性	说　明
AcceptButton	窗体的“确定”按钮，当用户按 Enter 键时相当于单击了该按钮
Backcolor	窗体的背景颜色
BackgroundImage	窗体的背景图像
BackgroundImageLayout	窗体的背景图像的布局方式
CancelButton	窗体的“取消”按钮，当用户按 Esc 键时相当于单击了该按钮
ControlBox	指示是否显示窗体的控制菜单图标与控制按钮
Enabled	指示是否启用窗体
Font	窗体中控件的文本的默认字体
ForeColor	窗体中控件的文本的默认颜色
FormBorderStyle	窗体的边框和标题栏的外观与行为
Icon	窗体的图标
Location	窗体相对于屏幕左上角的位置
MaximizeBox	指示窗体右上角的标题栏是否具有“最大化”/“还原”按钮
MinimizeBox	指示窗体右上角的标题栏是否具有“最小化”按钮
Opacity	窗体的不透明度，默认值为 100%，表明完全不透明
ShowIcon	指示是否在窗体的标题栏中显示图标
ShowInTaskbar	指示窗体是否在任务栏中显示
Size	窗体的大小（宽度和高度）
StartPosition	窗体第一次出现时的位置
Text	窗体标题栏上显示的内容
TopMost	指示该窗体是否处于其他窗体之上
WindowState	窗体的初始可视状态（正常、最大化、最小化）

属性值的设置有两种方式：一种是在设计程序时，通过“属性”窗口实现；另一种是在运行程序时，通过代码实现。

通过代码设置对象属性的一般格式是：

```
对象名.属性名 = 属性值;
```

对于代码所在的窗体设置属性的格式是：

```
this.属性名 = 属性值;
```

3.1.3 窗体的方法

窗体具有一些方法，调用这些方法可以实现特定的操作。窗体常用的方法如表 3.2 所示。

表 3.2　窗体常用的方法

方　　法	说　　明
Close( )	关闭窗体
Hide( )	隐藏窗体
Show( )	以非模式化的方式显示窗体
ShowDialog( )	以模式化的方式显示窗体

关闭窗体与隐藏窗体的区别在于：关闭窗体是将窗体彻底销毁，之后无法对窗体进行任何操作；隐藏窗体只是使窗体不显示，可以使用 Show 或 ShowDialog 方法使窗体重新显示。

模式窗体与非模式窗体的区别在于：模式窗体在其关闭或隐藏前无法切换到该应用程序的其他窗体；非模式窗体则可以在窗体之间随意切换。

调用方法的一般格式为：

```
对象名.方法名（[参数列表]）
```

如果要对调用语句所在的窗体调用方法，则用 this 关键字（表示当前类的对象）代替对象名，即：

```
this.方法名([参数列表]);
```

在面向对象的程序设计中，还有一种特殊的方法称为静态方法，这种类型的方法通过类名调用。调用的一般格式为：

```
类名.静态方法名([参数列表]);
```

3.1.4　窗体的事件

窗体作为对象，能够执行方法并对事件做出响应。窗体的常用事件如表 3.3 所示。

表 3.3　窗体的常用事件

事　　件	说　　明
Load	当用户加载窗体时发生
Click	在窗体的空白位置，单击鼠标时发生
Activated	当窗体被激活，变为活动窗体时发生
Deactivate	当窗体失去焦点，变为不活动窗体时发生
FormClosing	当用户关闭窗体时，在关闭前发生
FormClosed	当用户关闭窗体时，在关闭后发生

如果要为窗体对象添加事件处理程序，首先在设计器窗口选中窗体对象，然后在“属性”窗口的事件列表中找到相应的事件并双击它，即可在代码窗口看到该窗体的事件处理程序。以 Form1 的 Load 事件为例，其事件处理程序的格式为：

```
private void Form1_Load(object sender, EventArgs e)
{
```

```
        //程序代码
    }
```

其中，Form1_Load 是事件处理程序的名称，所有对象的事件处理程序默认名称都是“对象名_事件名”；所有对象的事件处理程序都具有 sender 和 e 两个参数，参数 sender 代表事件的源，参数 e 代表与事件相关的数据。

3.1.5 创建应用程序的操作界面

应用程序的操作界面由各个对象组成，创建操作界面就是在窗体上绘制代表各个对象的控件。

1．添加控件

向窗体中添加一个控件的步骤如下（以按钮为例）。

（1）单击“工具箱”中的“公共控件”选项卡，出现各种控件。

（2）将鼠标移到 Button 控件上单击，然后移到中间的窗体，这时会看到鼠标指针变成十字线的形状。

（3）将十字线放在窗体的适当位置，单击窗体并按住鼠标左键不放，拖动鼠标画出一个矩形。

（4）松开鼠标左键，会看到一个 Button 控件被创建在窗体上，如图 3.3 所示。

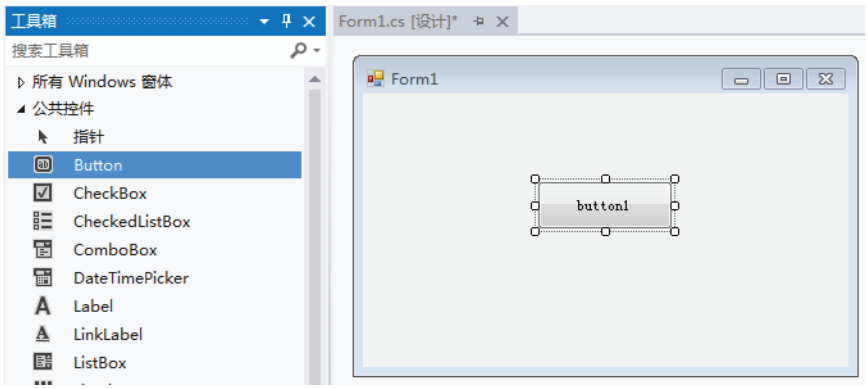


图 3.3　向窗体中添加 Button 控件

提示：

向窗体中添加控件的另一个简单方法，是双击工具箱中的控件。这样会在窗体的默认位置（如果先选定了某个控件对象，应是在该对象右下方位置，否则在窗体左上角）创建一个具有默认尺寸的控件，然后可以将该控件移到窗体中的其他位置。

2．选择控件

一个窗体上通常有多个控件，可以一次选择一个或多个控件。

如果要选择一个控件，单击该控件，即可选中该控件。

如果要选择多个控件，常用的方法有两种。一种方法是先选择第一个控件，然后按下 Shift 键（或 Ctrl 键）不放，依次单击要选择的其他控件，选择完毕后松开鼠标即可；另一种方法是在窗体的空白位置，单击窗体并按住左键不放，拖动鼠标画出一个矩形，然后松开鼠标，则该矩形区域内的控件都会被选中。

如果要撤销被选择的多个控件中的某个控件，只需按住 Shift 键（或 Ctrl 键）不放，单击要撤销的被选择控件。

3．调整控件的尺寸和位置

调整控件的尺寸和位置，可以通过设置控件的相应属性来实现。但在对控件尺寸和位置要求的精确度不高的情况下，最快捷的方法是在窗体设计器中直接用鼠标调整控件的尺寸和位置。

用鼠标调整控件尺寸的步骤如下：

（1）单击需要调整尺寸的控件，控件上出现 8 个尺寸手柄。

（2）将鼠标指针定位到尺寸手柄上，当指针变为双向箭头时按下鼠标左键，拖动该尺寸手柄直到控件达到所希望的大小为止。控件角上的 4 个尺寸手柄可以同时调整控件水平和垂直方向的大小，而边上的 4 个尺寸手柄调整控件一个方向的大小。

（3）松开鼠标左键。

提示：

也可以按 Shift 键加上箭头键，来调整选定控件的尺寸。

用鼠标调整控件位置的步骤如下：

（1）将鼠标指针指向要移动的控件，当鼠标指针变为十字箭头时，按下鼠标左键不放。

（2）用鼠标把该控件拖动到新位置。

（3）松开鼠标左键。

提示：

也可以通过键盘来调整选定控件的位置。每按一次箭头键，控件移动一个像素；如果按 Ctrl 键加上箭头键，控件每次移动一定的距离（多个像素），来与其他控件对齐。

4．对控件进行布局

对控件进行布局，可以通过“格式”菜单或“布局”工具栏实现。“布局”工具栏如图 3.4 所示。如果“布局”工具栏没有显示，可以通过“视图”菜单下的“工具栏”→“布局”命令来显示“布局”工具栏。如果工具栏上布局按钮没有全部显示，可以通过最右侧的下拉按钮来勾选显示。

图 3.4 “布局”工具栏

布局的内容包括对齐、大小、间距、叠放次序等。当多个控件被同时选中时，控件的所有布局功能都可用；只有一个控件被选中时，只有少数布局功能可用。

5．设置所有控件的 Tab 键顺序索引

Tab 键顺序是指当用户按下 Tab 键时，焦点在控件间移动的顺序。每个窗体都有自己的 Tab 键顺序，每个控件在窗体上也都有唯一的 Tab 键顺序索引。默认状态下，控件在窗体上的 Tab 键顺序索引与建立控件的顺序一致。如果要设置窗体上控件的 Tab 键顺序索引，可以分别对每个控件设置其 TabIndex 属性，也可以集中设置所有控件的 Tab 键顺序索引。

要集中设置所有控件的 Tab 键顺序索引，可以从“视图”菜单中选择“Tab 键顺序”命令。此时，窗体上每个控件的左上角都有一个蓝底白字的小方框，方框中白色的数字（从 0 开始）就是控件的当前 Tab 键顺序索引。如果需要改变多个控件的 Tab 键顺序索引，按照想设置的顺序依次单击各个控件，被单击过的控件，其左上角小方框变为白底蓝字，所有控件都被单击过之后，左上角小方框又变回蓝底白字。“Tab 键顺序”命令是一个切换命令，因此设置好所有控件的 Tab 键顺序索引之后，再次选择“Tab 键顺序”命令即可结束 Tab 键顺序索引的设置。

6．锁定所有控件

可以把窗体及该窗体上的所有控件进行锁定，锁定之后，窗体的尺寸及控件的位置和尺寸就无法通过鼠标或键盘操作来改变。锁定控件可以防止已处于理想位置的控件因为不小心而被移动。

如果要进行锁定操作，在窗体编辑区的任意位置右击，从弹出的快捷菜单中选择“锁定控件”命令即可。本操作只锁定选定窗体上的全部控件，不影响其他窗体上的控件。如果要调整锁定控件的位置和尺寸，可以在“属性”窗口中改变控件的 Location 和 Size 属性。“锁定控件”命令是一个切换命令，因此再次选择“锁定控件”命令即可解除锁定。

3.2 几种常用控件

下面介绍几种最常用的基本控件：标签、链接标签、文本框和按钮。

3.2.1 标签

标签（Label）控件的功能是显示不能编辑的文本信息，一般用于在窗体上进行文字说明。标签有 Name（名称）、AutoSize（自动尺寸）、BackColor（背景色）、BorderStyle（边框）、Enabled（可用）、Font（字体）、ForeColor（前景色）、Image（图像）、ImageAlign（图

像对齐方式）、Location（位置）、Locked（锁定）、Size（尺寸）、Text（文本）、TextAlign（文本排列）、Visible（可见）等属性。

1．设置标签的名称

任何对象都有名称，Name 属性指示代码中用来表示对象的名称。要设置 Label 控件的名称，首先选择 Label 控件，然后在“属性”窗口中设置 Name 属性为某个标识符即可。例如，有一个要显示“学生姓名”文本的标签，可以设置其 Name 属性为 lblStuName。

2．设置标签的文本

在 Label 控件中显示文本，使用 Text 属性。首先要选择 Label 控件，然后在“属性”窗口中设置该属性为某个字符串即可。

Label 控件中的文本默认的排列方式为靠上左对齐，通过设置 TextAlign 属性可以改变排列方式。TextAlign 属性值是 ContentAlignment 枚举类型，共有 9 个枚举值，默认值是 TopLeft。如果设置 TextAlign 为 TopCenter，排列方式为靠上居中。

3．设置标签的图像

Image 属性用来设置在标签上显示的图像。当在“属性”窗口中设置该属性时，单击该属性条，右端出现“…”按钮后单击它，会打开“选择资源”对话框，如图 3.5 所示。

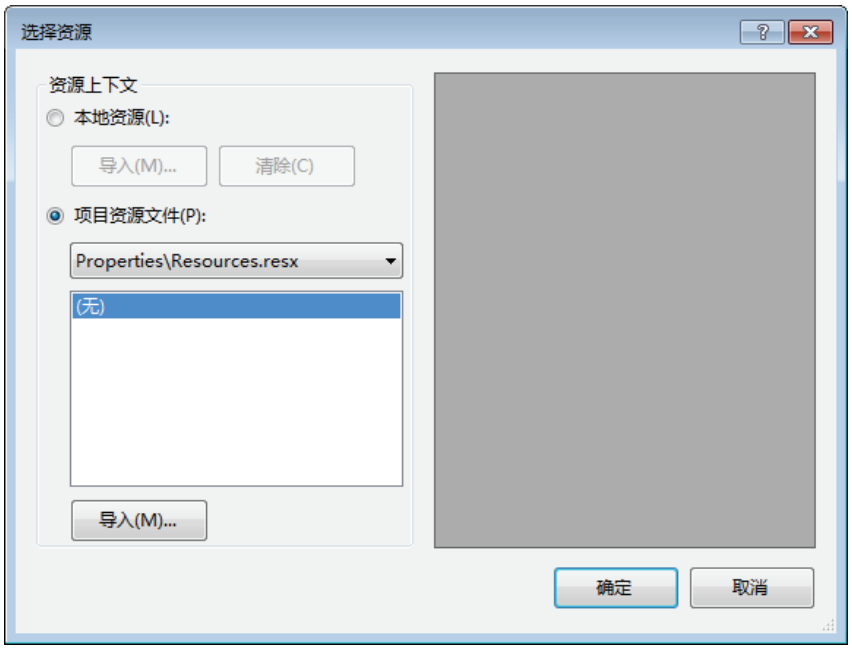


图 3.5 “选择资源”对话框

在“选择资源”对话框中，根据需要选择“本地资源”或“项目资源文件”，然后单击对应的“导入”按钮，在出现的“打开”对话框中选择所需的图像文件即可。

提示：

如果选择“本地资源”，程序运行时从指定位置的图像文件加载图像；如果选择“项目资源文件”，导入的图像文件会被复制到项目文件夹中的 Resources 文件夹下，程序运行时从 Resources 文件夹下的图像文件加载图像。

如果在运行时设置 Image 属性，可以使用 Image 类的静态方法 FromFile，格式如下：

```
对象名.Image = Image.FromFile("图像文件的路径及名称");
```

4．自动调整标签大小

AutoSize 属性决定标签文本能否根据文本大小自动调整标签大小。Label 控件的 AutoSize 属性默认值为 true，可以根据 Text 属性指定的文本自动调整标签的大小。如果 AutoSize 属性设置为 false，则标签将保持设计时定义的大小，在这种情况下，如果文本太长，则只能显示其中的一部分。当文本超过 Label 控件的宽度时，文本会自动换行，但在超过控件的高度时，超出的部分将无法显示出来。

5．标签的其他属性

描述 Label 控件的边框的属性是 BorderStyle，默认值为 none（无边框）。如果将该属性设成 FixedSingle，那么 Label 控件就有了一个黑色边框；如果将该属性设成 Fixed3D，那么 Label 控件就有了一个立体边框。

决定 Label 控件是否可见的属性是 Visible，默认值为 true（可见）。如果将该属性设成 false，那么 Label 控件将被隐藏。

还可以通过设置 Label 控件的 BackColor（取值 Transparent 无背景色）、ForeColor、Font 等属性来改变 Label 控件的其他外观；通过设置 Label 控件的 Location、Locked、Size 等属性来影响 Label 控件的位置和尺寸。

提示：

运行程序时，Label 控件不接受焦点，无法利用键盘或鼠标对其进行操作。Visual Studio 2012 中，如果未设置 Label 对象的 BackColor 属性，当设置了窗体的 BackColor 属性后 Label 对象的背景色也随之改变。

3.2.2 链接标签

链接标签（LinkLabel）控件的功能是显示带链接的文本信息，可以链接到对象（如其他窗体、本机文件）或网页。利用 LinkLabel 控件，可以向 Windows 窗体应用程序添加 Web 样式的链接。LinkLabel 不仅具有 Label 控件的所有属性，而且还有针对超链接和链接颜色的独特属性。

1．设置链接文本

在 LinkLabel 控件中显示文本，使用 Text 属性。设置好 Text 属性之后，所有文本都属

于链接的范围。如果要将文本的一部分设置为指向某个对象或网页的链接，还需要设置 LinkArea 属性。

LinkArea 属性用于获取或设置激活链接的文本区域（即文本中视为链接的范围）。该属性值是用包含两个数字的 LinkArea 对象表示的，这两个数字分别表示起始字符位置和字符数目。在“属性”窗口中，该属性值可以从键盘输入，也可以单击属性值右侧的小按钮，在弹出的 LinkArea 编辑器中选择要进行链接的文本范围。

2．设置链接颜色

与 LinkLabel 的颜色相关的属性有 3 个，分别是 LinkColor、ActiveLinkColor 和 Visited LinkColor。

（1）LinkColor 属性：获取或设置显示普通链接使用的颜色。

（2）ActiveLinkColor 属性：获取或设置显示活动链接（如单击鼠标时）的颜色。

（3）VisitedLinkColor 属性：获取或设置显示被访问过的链接所使用的颜色。当 Link Visited（链接是否被访问过）属性为 true 时，才能显示该颜色。

3．设置链接行为

LinkBehavior 属性，获取或设置一个表示链接行为的值。利用该属性，可以指定链接在 LinkLabel 控件中显示时的行为。

LinkBehavior 属性值为 LinkBehavior 枚举类型，共有 4 个成员，如表 3.4 所示，默认值为 SystemDefault。

表 3.4　LinkBehavior 枚举成员

成 员 名 称	说　　明
SystemDefault	此设置的行为取决于使用“控制面板”或 Internet Explorer 中的“Internet 选项”对话框设置的选项
AlwaysUnderline	该链接始终显示为带下画线的文本
HoverUnderline	仅当鼠标悬停在链接文本上时，该链接才显示带下画线的文本
NeverUnderline	链接文本从不带下画线（仍可使用 LinkColor 属性将该链接与其他文本区分开）

4．LinkClicked 事件

LinkClicked 事件是 LinkLabel 控件的主要事件，当单击 LinkLabel 控件内的链接文本时触发。

在窗体上双击 LinkLabel 控件，将在代码中添加 LinkClicked 事件处理程序的框架，然后在框架内部添加相应代码即可。

提示：

窗体或控件的大多数事件处理程序，都可以通过“属性”窗口添加。在“属性”窗口中单击“事件”按钮来切换到事件列表，然后双击相应的事件名，即可在代码中添加事件处理程序的框架。

。

【例 3-1】 标签与链接标签的简单应用。

使用 Label 和 LinkLabel 控件，设计一个打开对象或网页的程序，程序设计界面如图 3.6 所示。

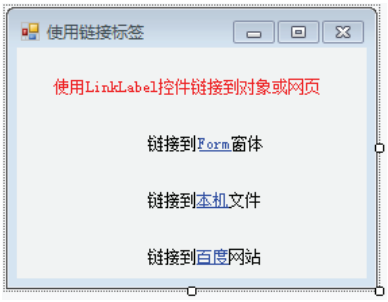


图 3.6 例 3-1 程序设计界面

具体步骤如下。

（1）设计界面。新建一个 C#的 Windows 应用程序，项目名称设置为 LinktoObjectAndWeb，向窗体中添加 1 个标签和 3 个链接标签，并按照图 3.6 所示调整控件位置和窗体尺寸。

（2）设置属性。窗体和各个控件的属性设置如表 3.5 所示。

表 3.5 例 3-1 对象的属性设置

对　象	属 性 名	属 性 值
Form1	Text	使用链接标签
label1	Text	使用 LinkLabel 控件链接到对象或网页
	ForeColor	Red
linkLabel1	Name	lnkForm
	Text	链接到 Form 窗体
	LinkArea	3，4
linkLabel2	Name	lnkFile
	Text	链接到本机文件
	LinkArea	3，2
linkLabel3	Name	lnkWeb
	Text	链接到百度网站
	LinkArea	3，2

（3）编写代码。依次双击 3 个链接标签，打开代码视图，分别在各个链接标签的 LinkClicked 事件处理程序中添加相应代码：

```
private void lnkForm_LinkClicked(object sender,LinkLabelLinkClicked EventArgs e)
{
    Form f2 = new Form();
    f2.Show();
    lnkForm.LinkVisited = true;
}
private void lnkFile_LinkClicked(object sender,LinkLabelLinkClicked EventArgs e)
{
    lnkFile.LinkVisited = true;
    //使用 Start 方法和一个本机文件路径，启动默认程序打开文件
    System.Diagnostics.Process.Start("Kiya.jpg");
    //此处使用相对路径，"Kiya.jpg"位于项目文件夹下的 bin\Debug 中
}
private void lnkWeb_LinkClicked(object sender,LinkLabelLinkClicked EventArgs e)
{
    lnkWeb.LinkVisited = true;
    //使用 Start 方法和一个 URL，启动默认浏览器打开网页
```

```
        System.Diagnostics.Process.Start("http://www.baidu.com");
    }
```

（4）运行程序。单击“启动调试”按钮或按 F5 键运行程序，在窗体中依次单击链接文本 Form、“本机”、“百度”查看结果。

3.2.3　文本框

文本框（TextBox）控件是程序界面上的主要输入对象，有时也用于输出。其主要功能是接收用户输入的信息，或显示系统提供的文本信息。在程序运行时，用户可以在文本框中编辑文本。

文本框具有标签的大多属性，如 Name、BackColor、BorderStyle、Enabled、Font、ForeColor、Location、Locked、Size、Text、TextAlign、Visible 等属性。

文本框还有一些自己特有的属性，如 MaxLength（最大长度）、Multiline（多行）、PasswordChar（密码字符）、ReadOnly（只读）、ScrollBars（滚动条）、SelectedText（选定的文本）、SelectionStart（选择起始点）、SelectionLength（选择长度）、TextLength（文本长度）、WordWrap（文本换行）等。

1．设计时设置文本框的文本

在 TextBox 控件中显示文本，使用 Text 属性。TextBox 在默认情况下只显示单行文本，且不显示滚动条。如果文本长度超过可用空间，则只能显示部分文本。

通过设置 Multiline、WordWrap 和 ScrollBars 3 个属性，可以改变 TextBox 的外观和行为。把 Multiline 属性设为 true，可以使 TextBox 在运行时接收或显示多行文本。WordWrap 属性的默认值为 true，即允许自动换行。只要没有水平方向滚动条，TextBox 中的多行文本会自动按字换行。ScrollBars 属性的默认值为 none（无滚动条），还有 Horizontal（水平）、Vertical（竖直）、Both（两者）3 个可取值。如果要显示水平滚动条，除了将 ScrollBars 属性值设置为 Horizontal，还需要将 WordWrap 属性值设置为 false。

自动换行省去了用户在行尾插入换行符的麻烦，当一行文本已超过所能显示的长度时，TextBox 自动将文本折回到下一行显示。如果用户因为特殊要求必须使用换行符，在设置 Text 属性时，在属性值处不能直接输入换行符，而需要在“属性”窗口中单击属性值右侧的下拉箭头，然后在下拉列表框中适当的位置输入换行符。

2．运行时设置文本框的文本

当一个 TextBox 首次得到焦点时，TextBox 的所有文本默认是选中的。用户可以用键盘和鼠标移动插入点，当 TextBox 失去焦点而后再得到时，插入点位置与用户最后设置的位置一样。在某些情况下，可能用户有特殊要求，例如，有时希望新字符出现在已有文本后面，有时希望新的输入替换原有文本。

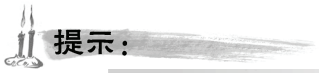
利用 TextBox 的 SelectionStart、SelectionLength 和 SelectedText 属性，可以控制 TextBox 的插入点和选择行为。这 3 个属性不能通过“属性”窗口设置，只能通过代码访问。

SelectionStart 属性是一个数字，代表选择文本的起始点，即 TextBox 文本内的插入点，

其中值 0 表示最左边的位置。如果其值大于或等于文本中的字符数，那么插入点将被放在最后一个字符之后。

SelectionLength 属性是一个设置插入点宽度的数值，用于指示选择文本的长度。把 SelectionLength 设为大于 0 的值，会选中并突出显示从当前插入点开始的 SelectionLength 个字符。如果有一段文本被选中，此时用户输入的文字将替换被选中的文本。

SelectedText 属性用于指示选定的文本。可以在运行时通过该属性来获取当前选定的文本，也可以给该属性赋值以替换当前选中的文本。如果没有选中的文本，给 SelectedText 属性赋值将在当前插入点插入文本。



提示：

如果在窗体加载时就让文本框中的文本选中，需要先设置文本框的 TabIndex 属性为 0，然后在窗体的 Load 事件方法中设置 SelectionStart 和 SelectionLength 属性。

3．密码文本框

密码文本框是文本框常用的一种特殊形式，它允许在用户输入密码的同时显示星号（*）之类的占位符。利用文本框的 PasswordChar 和 MaxLength 属性，可以实现密码框的功能。

PasswordChar 属性用于指定显示在文本框中的字符。例如，若希望在密码框中显示星号，则可在“属性”窗口中将 PasswordChar 属性指定为“*”，这样无论用户输入什么字符，文本框中都显示星号。

MaxLength 属性用于指定允许在文本框中输入的最大字符数。如果输入的字符数超过 MaxLength 指定的值，系统不接收多出的字符并发出嘟嘟声。

4．只读文本框

只读文本框不允许用户进行编辑操作，从而可以防止用户更改文本框内容。ReadOnly 属性可以实现只读文本框的功能，只需将该属性值设置为 true 即可。此时，用户可滚动文本框中的文本并将其突出显示，但不能做任何更改。ReadOnly 属性只影响程序运行时的用户交互，在运行时仍然可以通过代码更改文本框的内容。

5．文本框的常用方法

文本框的大多数方法都是用来进行文本操作，常用的方法有 AppendText（追加文本）、Clear（清除所有文本）、Copy（复制选定文本）、Cut（剪切选定文本）、Focus（获得焦点）、Paste（粘贴指定文本）、Select（选择指定范围的文本）、SelectAll（全选）等。

6．文本框的常用事件

文本框可以识别多个事件，常用的事件有 TextChanged（文本更改）、KeyDown（按下键）、KeyUp（释放键）、KeyPress（按下并释放键）、MouseDown（按下鼠标按钮）、MouseUp（释放鼠标按钮）、MouseMove（鼠标指针移过）等。

提示：

在窗体上双击 TextBox 控件，将在代码中添加 TextChanged 事件处理程序的框架，然后在框架内部添加相应代码即可。

3.2.4　按钮

按钮（Button）控件是应用程序中使用最多的控件对象之一，常用来接收用户的操作信息，激发相应的事件。

按钮具有标签的大多属性，如 Name、AutoSize、BackColor、Enabled、Font、ForeColor、Image、Location、Locked、Size、Text、TextAlign、Visible 等属性。按钮还有一些自己特有的属性，如 BackgroundImage（背景图像）、FlatStyle（样式）等。

1．创建键盘访问键快捷方式

Text 属性可以用来设置按钮上显示的文本，同时也可以用来创建按钮的访问键快捷方式。要为按钮创建访问键快捷方式，只需在作为访问键的字母前添加一个&符号。例如，要为按钮的文本 OK 创建访问键 O，应在字母 O 前添加连字符，即将按钮的 Text 属性设置为“&OK”。此时，字母 O 将带下画线，程序运行时按 Alt+O 组合键就相当于用鼠标单击按钮。

2．Click 事件

当用户用鼠标单击按钮时，将触发按钮的 Click 事件，这也是按钮最常响应的事件。

用鼠标单击按钮的过程中，还会触发一系列的事件，如果要在这些相关事件中附加事件处理程序，则应确保操作不发生冲突。单击按钮过程中，按钮相关事件发生的顺序为 MouseEnter、MouseMove、MouseDown、Click、MouseUp、MouseMove、MouseLeave。

提示：

在窗体上双击 Button 控件，将在代码中添加 Click 事件处理程序的框架，然后在框架内部添加相应代码即可。

3．增强按钮的视觉效果

可以通过设置 Image 属性给 Button 控件添加图标以增强视觉效果，然后设置 ImageAlign 属性来指定显示图标的位置。

也可以通过设置 BackgroundImage 属性给 Button 控件添加背景图像以增强视觉效果，然后设置 BackgroundImageLayout 属性来指定背景图像的布局。

【例 3-2】 文本框与按钮的简单应用。

使用 TextBox 和 Button 控件，设计一个显示密码原文的程序，程序设计界面如图 3.7 所示。

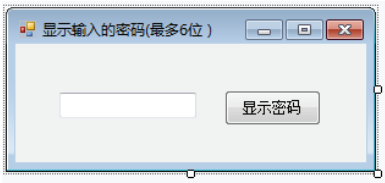


图 3.7　例 3-2 程序设计界面

具体步骤如下。

（1）设计界面。新建一个 C#的 Windows 应用程序，项目名称设置为 ShowPassword，向窗体中添加一个文本框和一个按钮，并按照图 3.7 所示调整控件位置和窗体尺寸。

（2）设置属性。窗体和各个控件的属性设置如表 3.6 所示。

表 3.6　例 3-2 对象的属性设置

对　　象	属　性　名	属　性　值
Form1	Text	显示输入的密码（最多 6 位）
textBox1	Name	txtPassword
	PasswordChar	*
	MaxLength	6
button1	Name	btnShow
	Text	显示密码

（3）编写代码。双击按钮，打开代码视图，在按钮的 Click 事件处理程序中，添加相应代码：

```
private void btnShow_Click(object sender, EventArgs e)
{
    //利用消息框 MessageBox 显示密码原文
    MessageBox.Show("输入的密码为："+txtPassword.Text, "密码原文");
    //Show 方法的第一个参数表示消息文本，第二个参数表示消息框标题
}
```

（4）运行程序。单击“启动调试”按钮或按 F5 键运行程序，在文本框中输入密码，单击“显示密码”按钮查看结果，如图 3.8 所示。

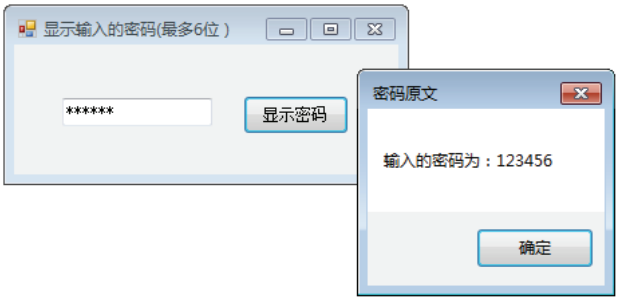


图 3.8　例 3-2 程序运行界面

3.2.5　控件的命名规则

窗体和控件都有自己的名称，可以通过 Name 属性进行命名。为了提高控件名称的可读性，建议在为控件命名时，在控件名称前面加上控件的类型名称缩写作为前缀，如窗体（frm）、标签（lbl）、按钮（btn）等。紧跟在 3 个（少数控件是 4 或 5 个）小写字母后面的则是该控件用途的简短描述，第一个字母建议大写，其他的使用小写；若有多个单词组成

对象名称，则建议每个单词的首字母都采用大写。例如，有一个要显示“学生姓名”文本的 Label 控件，其命名如图 3.9 所示。表 3.7 列出了窗体与常用控件名称的前缀约定。

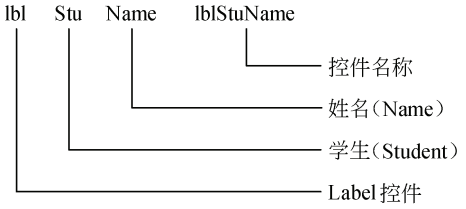


图 3.9　控件命名

表 3.7　窗体与常用控件的名称缩写

中　文　名	英　文　名	名称缩写	命名实例与含义
窗体	Form	frm	frmScore（成绩）
按钮	Button	btn	btnOk（确定）
复选框	CheckBox	chk	chkAgree（同意）
复选列表框	CheckedListBox	chl	chlCourse（课程）
颜色对话框	ColorDialog	cdlg	cdlgBC（背景色）
组合框	ComboBox	cbo 或 cmb	cboStuName（学生姓名）
日期时间选取器	DateTimePicker	dtp	dtpStart（开始）
浏览文件夹对话框	FolderBrowserDialog	fbdlg	fbdlgSave（保存）
字体对话框	FontDialog	fdlg	fdlgText（文本）
分组框	GroupBox	grp	grpColor（颜色）
水平滚动条	HScrollBar	hsb	hsbWidth（宽度）
图像列表	ImageList	img	imgTreeView（树状视图）
标签	Label	lbl	lblStuName（学生姓名）
链接标签	LinkLabel	lnk 或 llb	lnkFile（文件）
列表框	ListBox	lst	lstBook（书籍）
列表视图	ListView	lsv 或 lvw	lsvFiles（文件）
数字微调框	NumericUpDown	nup	nupVal（数值）
打开文件对话框	OpenFileDialog	odlg	odlgPic（图片）
面板	Panel	pnl	pnlInfo（信息）
图片框	PictureBox	pic	picPhoto（照片）
进度条	ProgressBar	prg	prgInstall（安装）
多格式文本框	RichTextBox	rtf 或 rtx	rtfEditor（编辑器）
单选按钮	RadioButton	rad 或 rdo	radSex（性别）
保存文件对话框	SaveFileDialog	sdlg	sdlgDoc （文档）
选项卡	TabControl	tab	tabStu（学生）
文本框	TextBox	txt	txtMoney（钱数）
计时器	Timer	tmr	tmrTraffic（交通）
树视图	TreeView	tvw	tvwBooks（书籍）
垂直滚动条	VScrollBar	vsb	vsbHeight（高度）

3.3 本章小结

本章主要介绍了 Windows 窗体和 Label、LinkLabel、TextBox、Button 几种常用控件，重点内容如下：

- Windows 窗体的结构。
- Windows 窗体的常用属性、方法与事件。
- 创建应用程序的操作界面。
- Label、LinkLabel、TextBox 和 Button 控件的使用。
- 控件的命名规则。

习　　题

1．选择题

（1）以模式化的方式显示窗体，需要使用（　　）方法。

A．Show　　B．ShowDialog　　C．ShowForm　　D．ShowFixed

（2）决定 Label 控件是否可见的属性是（　　）。

A．Hide　　B．Show　　C．Visible　　D．Enabled

（3）把 TextBox 控件的（　　）属性设为 true，可使其在运行时接收或显示多行文本。

A．WordWrap　　B．Multiline　　C．ScrollBars　　D．ShowMultiline

（4）利用文本框的（　　）属性，可以实现密码框的功能。

A．Password　　B．Passwords　　C．PasswordChar　　D．PasswordChars

（5）如果要为“取消”按钮的文本“Cancel”创建访问键 C，应将按钮的 Text 属性设置为（　　）。

A．“&Cancel”　　B．“%Cancel”　　C．“@Cancel”　　D．“^Cancel”

2．思考题

（1）关闭窗体与隐藏窗体有什么区别？

（2）模式窗体与非模式窗体有什么区别？

（3）简述 Label、Button 和 TextBox 控件的作用。

3．上机练习题

（1）编写一个简单的计算器，能够实现正整数的加、减、乘、除 4 种运算，设计界面如图 3.10 所示。

图 3.10　简单计算器

（2）编写一个提供常用网址的程序，可以快速访问“百度”、“新浪”、“腾讯”、“搜狐”、“网易”等网站。

（3）设计一个转换英文大小写的程序，输入字符时，自动将英文字母分别转换为大写和小写两种格式。

提示：

使用 Label 和 TextBox 控件设计，利用 TextBox 控件的 TextChanged 事件实现即时转换功能，转换后的两个字符串可以利用两个只读的文本框输出。