实 验 十六　用户自定义控件

16.1　实验目的

设计者面对 VS 工具箱提供的数量多、功能通用的控件总会有独特的构思，VS 支持用户自定义控件的外观与事件响应方法，本章实验介绍设计自定义控件的制作方法。

16.2　自定义控件简介

开发者使用 VS2012 提供的通用控件时会发现控件的表现不符合预想的效果，例如 ImageList 控件限制所有图片尺寸统一，使用 ImageList 控件的 ListView 控件显示多个图片时，图片被拉伸变形。如果希望图片保持原有比例，要采用自定义控件满足需求。通过窗体绘图原理结合事件响应机制，开发者可设计独特的控件。VS 工具提供三种创建自定义控件的方式：

1. 继承 UserControl 类重载 Paint 事件实现独特外观，较接近一般窗体设计过程。
2. 继承标准控件，修改已有控件的外观与方法。
3. 继承 Control 控件，编写全新的控件难度高。

16.3　自定义控件设计实现

通过继承 UserControl 类创建自定义控件的方法步骤如下：

1. 构想自定义控件的外观和事件处理方法。
2. 继承 UserControl 类生成自定义控件类。
3. 向自定义控件类添加成员属性。
4. 重载控件类的 OnPaint 函数。
5. 定义控件的事件代理变量。
6. 程序窗体对象中添加自定义控件。
7. 程序窗体代码注册自定义控件的事件处理函数。

图16-1演示一个自定义控件在鼠标点击时切换两种图片的效果。

新建一个普通的窗体应用程序，图16-2演示向项目添加用户控件的菜单，定义用户控件的类名为 RoundSwitch。

图16-3演示通过资源管理器添加两个图片 butn01.png 与 butn02.png 文件。

属性的元数据，主要包括 Category，Description，DefaultValue，Browserable 和 Editor，Category 指示属性在设计器中的分类，Description 指示设计器中的文本描述，DefaultValue 指示属性的默认值，Browserable 确定属性在设计器中是否可见。Editor 指示编辑属性时使用的编辑器，自定义控件往往其属性的编辑也是用户自定义的窗口。

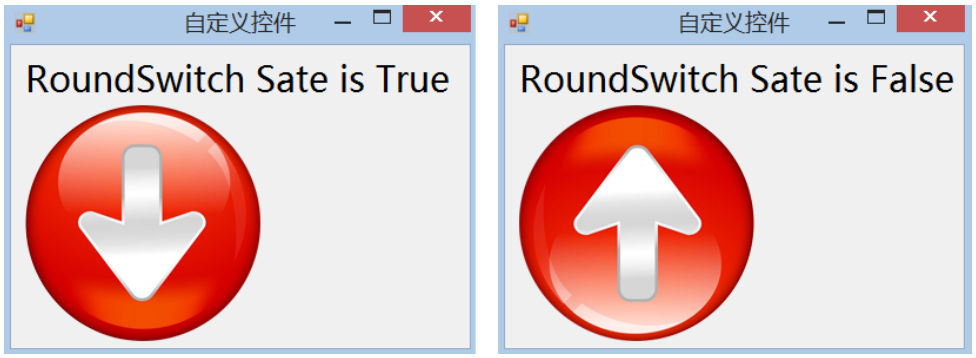

图 16-1 自定义控件运行效果

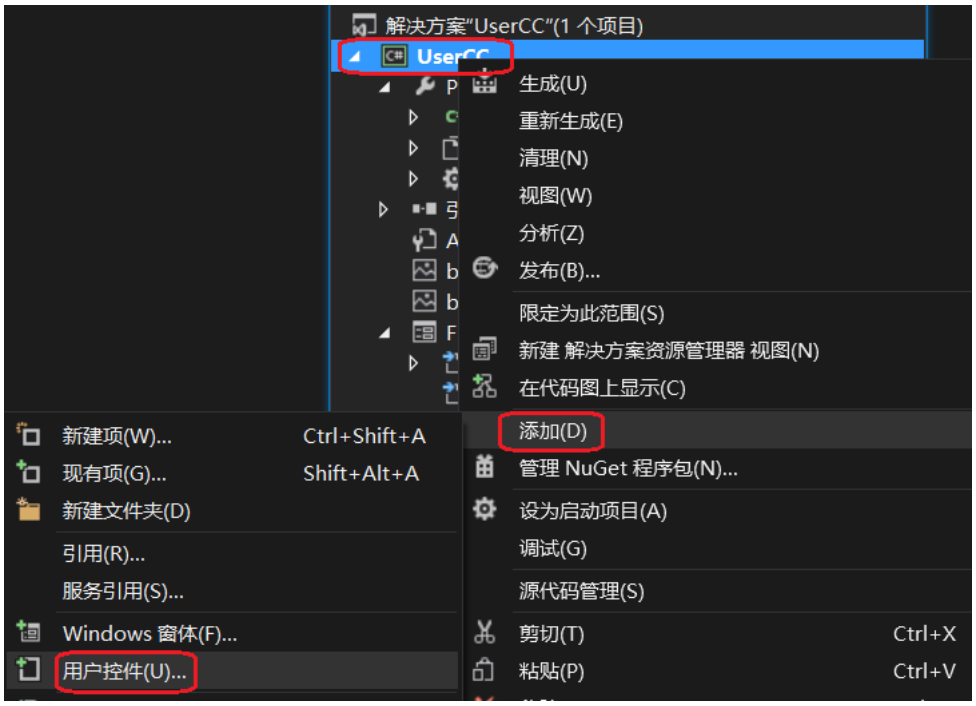

图 16-2 在项目中添加用户控件

下面的代码向类 RoundSwitch 添加一个 mSwitchOn 的布尔变量，在窗体设计状态的属性栏中会显示为 SwitchOn 属性。

```
//开关状态变量
public bool mSwitchOn;
//设置属性值在属性窗口的可见性
[Browsable(true)]
public bool SwitchOn
{
    get
    {
        return mSwitchOn;
    }
```

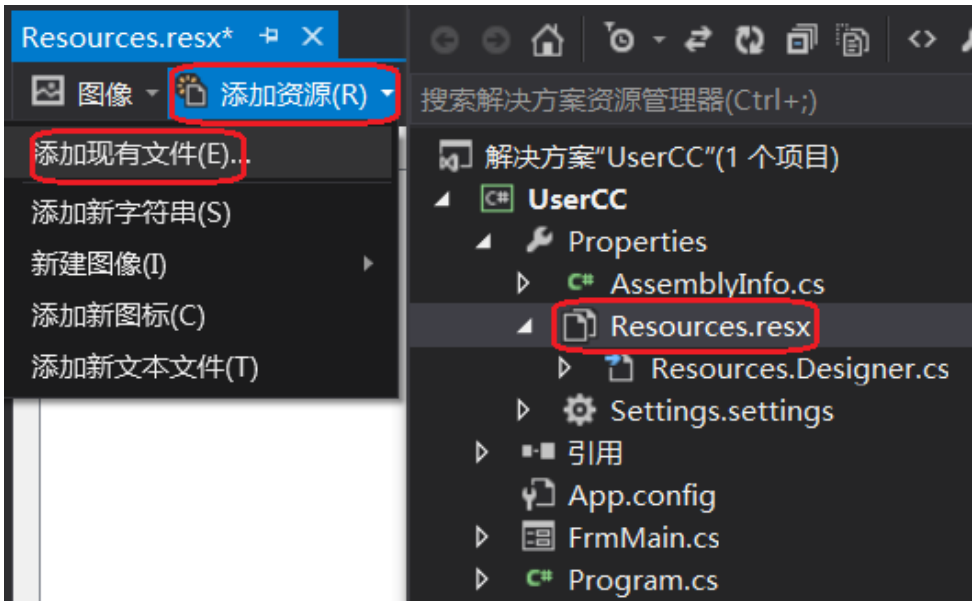

图 16-3 通过资源管理器添加图片资源

```
    set
    {
        mSwitchOn = value;
        Invalidate();
    }
}
```

继续在 RoundSwitch.cs 文中定义 DrawCtl 函数负责绘制控件的外观。

```
private void DrawCtl(PaintEventArgs e)
{
    //绘制图片内容
    Bitmap theBmp1;
    Bitmap theBmp2;
    Rectangle destRect = new Rectangle(0, 0, 200, 200);
    //绘制头图片
    theBmp1 = Resources.butn01;
    theBmp2 = Resources.butn02;
    if (mSwitchOn)
    {
        e.Graphics.DrawImage(theBmp1, destRect);
    }
    else
    {
        e.Graphics.DrawImage(theBmp2, destRect);
    }
```

```
}
```

重载控件的 OnPaint 事件，调用 DrawCtl 函数。

```
protected override void OnPaint(PaintEventArgs e)
{
  base.OnPaint(e);
  DrawCtl(e);
}
```

定义控件的外部事件代理变量，当控件相应事件发生时调用外部函数。

```
public delegate void ControlDelegate(bool t1);
public event ControlDelegate ProcessEvent;
public void AddMouseClickEvent(ControlDelegate cde)
{
  ProcessEvent += cde;
}
```

定义控件的鼠标点击响应事件，实现对控件属性的修改，并执行外部函数。

```
private void RoundSwitch_MouseClick(object sender, MouseEventArgs e)
{
  //控件响应用户鼠示点击，修改自身变量值
  mSwitchOn = !mSwitchOn;
  //调用控件的" 事件" 回调函数
  ProcessEvent(mSwitchOn);
  Invalidate();
}
```

在控件的构造函数中添加三行代码来提高实际显示效果，减少闪烁现象。

```
public RoundSwitch()
{
  InitializeComponent();
  //减少闪烁
  this.SetStyle(ControlStyles.UserPaint, true);
  this.SetStyle(ControlStyles.AllPaintingInWmPaint, true);
  this.SetStyle(ControlStyles.DoubleBuffer, true);
}
```

自定义控件的全部代码编写完成，编译项目后自定义控件会出现工具箱中，开发者像使用其它控件一样可将其拖放到窗体上。在窗体中定义控件的响应函数：

```
public void SwitchChange(bool t1)
{
  label1.Text=string.Format("RoundSwitch Sate is {0}",t1);
}
```

在窗体的 Load 事件把 SwitchChange 函数注册到控件的鼠标事件中。

```
private void FrmMain_Load(object sender, EventArgs e)
```

```
{
    roundSwitch1.SwitchOn = true;
    //向控件注册事件处理函数，用于切换要显示的窗体对象
    roundSwitch1.AddMouseClickEvent(SwitchChange);
}
```

16.4 在状态栏上显示自定义控件

.Net 自带的一个类：ToolStripControlHost。使用 ToolStripControlHost 类承载自定义控件或任何其他 Windows 窗体控件。若要自定义 ToolStripItem，请从 ToolStripControlHost 进行派生，并创建自定义实现。可以重写 OnSubscribeControlEvents 等方法，以处理由寄宿的控件引发的事件，也可以在属性中增加自定义功能，以增强寄宿的控件。

它继承自 ToolStripItem 类，承载自定义控件或 Windows 窗体控件，通过它可以让自定义控件和 Windows 窗体控件显示在菜单（MenuStrip）、工具栏（ToolStrip）、状态栏（StatusStrip）上了。

16.5 思考与练习

1. 完成本章的自定义控件设计。
2. VS 工具箱中的 ListView 的控件能多个图片，但是图片显示时会被拉伸，试设计能保持图片原有尺寸的 ListView 控件。