第 1 章

WinForm 窗体开发

带图像列表的系统登录程序

使用 Timer 组件实现冬奥会倒计时

自定义最大化、最小化和关闭按钮

图形化的导航界面

字母与 ASCII 码的转换

.....

实例 001　带图像列表的系统登录程序

源码位置：Code\01\001

实例说明

常用的管理软件一般都有系统登录验证模块，对进入系统的用户进行安全性检查，防止非法用户进入系统。本实例运行后，用户列表框中的每个用户都以图标的形式显示，增强了登录窗体的视觉效果。选择相应的图标，在“密码”文本框中输入正确的密码，将会在列表框中显示登录用户。实例运行效果如图 1.1 所示。

图 1.1　带图像列表的系统登录程序

关键技术

本实例使用了 ListView 控件。ListView 控件的 View 属性是一个 View 枚举值，用于获取或设置数据项在控件中的显示方式。

实现过程

（1）打开 Visual Studio 2022 开发环境，新建一个名为 UseImageList 的 Windows 窗体应用程序。

（2）更改默认窗体 Form1 的 Name 属性为 Frm_Main，向窗体中添加一个 ListView 控件，用于显示用户登录信息；添加两个 TextBox 控件，分别用于输入用户名和密码；添加两个 Button 按钮，分别用于登录系统和退出登录窗体。

（3）程序主要代码如下：

001-1

```
01  private void Method(DataTable dt)
02  {
03      lv_Person.Items.Clear();                                  //清空控件中所有数据项
04      for (int j = 0; j < dt.Rows.Count; j++)
05      {
06          if (j % 2 == 0)
07          {
08              lv_Person.Items.Add(                              //添加数据项和图像
09                  dt.Rows[j][0].ToString(), 0);
10          }
11          else
12          {
```

```
13              lv_Person.Items.Add(                                    //添加数据项和图像
14                  dt.Rows[j][0].ToString(), 1);
15          }
16      }
17  }
```

扩展学习

ListView 控件的 View 属性

ListView 控件的 View 属性用于设置数据项在控件中的显示方式，该属性是一个 View 枚举值，View 枚举值包括 LargeIcon、Details、SmallIcon、List 和 Tile。

实例 002　使用 Timer 组件实现冬奥会倒计时　源码位置：Code\01\002

实例说明

使用 Timer 组件， 可以按用户定义的时间间隔来引发事件。 引发的事件一般为周期性的， 每隔若干秒或若干毫秒执行一次。 本实例中使用 Timer 组件实现了 2022 年北京冬奥会倒计时功能， 实例运行效果如图 1.2 所示。

图 1.2　使用 Timer 组件实现冬奥会倒计时

关键技术

本实例主要用到了 Timer 组件的 Enabled 属性，用于获取或设置计时器的运行状态。

实现过程

（1）打开 Visual Studio 2022 开发环境，新建一个名为 TimeNow 的 Windows 窗体应用程序。

（2）更改默认窗体 Form1 的 Name 属性为 Frm_Main，向窗体中添加 8 个 TextBox 文本框控件，分别用于显示 2022 年北京冬奥会倒计时信息；添加一个 Timer 组件，用于间隔性地计算倒计时信息。

（3）程序主要代码如下：

002-1

```
01  private void timer1_Tick(object sender, EventArgs e)
02  {
03      DateTime get_time1 = DateTime.Now;                          //获取当前系统时间
04      DateTime sta_ontime1 = Convert.ToDateTime(                  //获取冬奥会开幕时间
05          Convert.ToDateTime(“2022-2-4 00:00:00”));
06      txtYear.Text = DateAndTime.DateDiff(                        //计算相隔年数
07          "yyyy", get_time1, sta_ontime1,
08          FirstDayOfWeek.Sunday,
09          FirstWeekOfYear.FirstFourDays).ToString();
10      txtMonth.Text = DateAndTime.DateDiff(                       //计算相隔月数
11          "m", get_time1, sta_ontime1,
12          FirstDayOfWeek.Sunday,
13          FirstWeekOfYear.FirstFourDays).ToString();
14      textday.Text = DateAndTime.DateDiff(                        //计算相隔天数
15          "d", get_time1, sta_ontime1,
16          FirstDayOfWeek.Sunday,
17          FirstWeekOfYear.FirstFourDays).ToString();
18      txtHour.Text = DateAndTime.DateDiff(                        //计算相隔小时数
19          "h", get_time1, sta_ontime1,
20          FirstDayOfWeek.Sunday,
21          FirstWeekOfYear.FirstFourDays).ToString();
22      txtmintue.Text = DateAndTime.DateDiff(                      //计算相隔分数
23          "n", get_time1, sta_ontime1,
24          FirstDayOfWeek.Sunday,
25          FirstWeekOfYear.FirstFourDays).ToString();
26      txtsecon.Text = DateAndTime.DateDiff(                       //计算相隔秒数
27          "s", get_time1, sta_ontime1,
28          FirstDayOfWeek.Sunday,
29          FirstWeekOfYear.FirstFourDays).ToString();
30      textBox1.Text = DateTime.Now.ToString();
31  }
```

扩展学习

使用 DateAndTime 类的 DateDiff 方法计算时间间隔

由于 DateAndTime 类定义在 Visual Basic 的程序集中，所以使用 DateAndTime 类时，首先要引用 Visual Basic 程序集并添加 Microsoft.Visual Basic 命名空间，然后即可方便地使用 DateAndTime 类的 DateDiff 方法。

实例 003　自定义最大化、最小化和关闭按钮

源码位置：Code\01\003

实例说明

用户在制作应用程序时，为了使用户界面更加美观，一般都自行设计窗体的外观，以及窗体的最大化、最小化和关闭按钮。本实例通过资源文件来存储窗体的外观，以及最大化、最小化和关闭按钮的图片，再通过鼠标移入、移出事件来实现按钮的动态效果。

实例运行效果如图 1.3 所示。

图 1.3　自定义最大化、最小化和关闭按钮

关键技术

本实例首先使用资源文件来存储窗体的外观、“最大化”“最小化”和“关闭”按钮的图片，然后使用窗体的 WindowState 属性实现窗体的最大化、最小化和还原操作。

实现过程

（1）打开 Visual Studio 2022 开发环境，新建一个名为 ControlFormStatus 的 Windows 窗体应用程序。

（2）更改默认窗体 Form1 的 Name 属性为 Frm_Main，向窗体中添加两个 Panel 控件，分别用来显示窗体标题栏和标题栏下面的窗体部分；添加 3 个 PictureBox 控件，分别用来表示“最大化”“最小化”和“关闭”按钮。

（3）程序主要代码如下：

003-1

```
///<summary>
///设置窗体最大化、最小化和关闭按钮的单击事件
///</summary>
///<param Frm_Tem="Form">窗体</param>
///<param n="int">标识</param>
public void FrmClickMeans(Form Frm_Tem, int n)
{
    switch (n)                                                          //窗体的操作样式
    {
        case 0:                                                         //窗体最小化
            Frm_Tem.WindowState = FormWindowState.Minimized;            //窗体最小化
            break;
        case 1:                                                         //实现窗体最大化和还原的切换
            {
                if (Frm_Tem.WindowState == FormWindowState.Maximized)   //如果窗体当前是最大化
                    Frm_Tem.WindowState = FormWindowState.Normal;       //还原窗体大小
                else
                    Frm_Tem.WindowState = FormWindowState.Maximized;    //窗体最大化
                break;
            }
        case 2:                                                         //关闭窗体
            Frm_Tem.Close();
            break;
    }
}
```

扩展学习

通过属性控制窗体的最大化和最小化

Windows 窗体提供了“最大化”和“最小化”按钮，开发人员可以根据需要设置这两个按钮可用或不可用，该功能主要通过设置 Windows 窗体的 MaximizeBox 属性和 MinimizeBox 属性来实现，其中 MaximizeBox 属性用来设置窗体的“最大化”按钮是否可用，MinimizeBox 属性用来设置窗体的“最小化”按钮是否可用。

实例 004　图形化的导航界面

源码位置：Code\01\004

实例说明

图形化的导航界面，顾名思义就是在窗体中使用图形导航代替传统的文字导航。与传统的文字导航界面相比，图形化导航界面的优势在于可以使得窗体界面更加美观、吸引人。本实例使用 C# 制作了一个图形化的导航界面，实例运行效果如图 1.4 所示。

图 1.4　图形化的导航界面

关键技术

本实例主要用到了 Button 控件的 BackColor 属性、FlatStyle 属性和 TextImageRelation 属性。BackColor 属性主要用来获取或设置控件的背景色，FlatStyle 属性主要用来获取或设置按钮控件的平面样式外观，TextImageRelation 属性主要用来获取或设置文本和图像之间的相对位置。

实现过程

（1）打开 Visual Studio 2022 开发环境，新建一个名为 ImageNavigationForm 的 Windows 窗体应用程序。

（2）更改默认窗体 Form1 的 Name 属性为 Frm_Main，在该窗体中添加一个 MenuStrip 控件，用来设计菜单栏；添加一个 ToolStrip 控件，用来设计工具栏；添加两个 Panel 控件，用来将窗体分割成两部分；添加 13 个 Button 控件，将它们的 BackColor 属性设置为 Transparent、FlatStyle 属性设置为 Flat、TextImageRelation 属性设置为 ImageBeforeText，这些 Button 控件用来设计图形化的

导航按钮。

（3）程序主要代码如下：

004-1

```
01  private void button1_Click(object sender, EventArgs e)
02  {
03      button5.Visible = true;                                    //设置button5控件可见
04      button6.Visible = true;                                    //设置button6控件可见
05      button7.Visible = true;                                    //设置button7控件可见
06  }
07  private void button2_Click(object sender, EventArgs e)
08  {
09      button8.Visible = true;                                    //设置button8控件可见
10      button9.Visible = true;                                    //设置button9控件可见
11      button10.Visible = true;                                   //设置button10控件可见
12  }
13  private void button3_Click(object sender, EventArgs e)
14  {
15      button11.Visible = true;                                   //设置button11控件可见
16      button12.Visible = true;                                   //设置button12控件可见
17      button13.Visible = true;                                   //设置button13控件可见
18  }
```

扩展学习

ToolStrip 控件的使用

ToolStrip 控件主要用来设计窗体的工具栏，使用该控件可以创建具有 Windows XP、Office、Internet Explorer 或自定义的外观和行为的工具栏及其他用户界面元素，这些元素支持溢出及运行时项重新排序。

实例 005　字母与 ASCII 码的转换

源码位置：Code\01\005

实例说明

ASCII（American Standard Code for Information Interchange，美国信息互换标准代码）是基于拉丁字母的编码系统，也是现今最通用的单字节编码系统。在程序设计中，可以方便地将字母转换为 ASCII 码，或将 ASCII 码转换为字母。实例运行效果如图 1.5 所示。

图 1.5　字母与 ASCII 码的转换

关键技术

本实例实现时主要用到了 Encoding 对象的 GetBytes 方法，GetBytes 方法接收一个字符串或字符数组作为参数，最后返回字节数组，可以根据字节数组得到字母的 ASCII 码。

实现过程

（1）打开 Visual Studio 2022 开发环境，新建一个名为 ASCII 的 Windows 窗体应用程序。

（2）更改默认窗体 Form1 的 Name 属性为 Frm_Main，更改 Text 属性为“字母与 ASCII 码的转换”，向窗体中添加一个 GroupBox 控件，向 GroupBox 控件中添加 4 个 TextBox 控件，分别用于输入和输出字符及 ASCII 码信息；向 GroupBox 控件中添加两个 Button 控件，分别用于将字母转换为 ASCII 码，或者将 ASCII 码转换为字母。

（3）程序主要代码如下：

005-1

```
01  private void btn_ToASCII_Click(object sender, EventArgs e)
02  {
03      if (txt_char.Text != string.Empty)                          //判断输入是否为空
04      {
05          if (Encoding.GetEncoding("unicode").                    //判断输入是否为字母
06              GetBytes(new char[] { txt_char.Text[0] })[1] == 0)
07          {
08              txt_ASCII.Text = Encoding.GetEncoding(              //获取字符的ASCII码值
09                  "unicode").GetBytes(txt_char.Text).ToString();
10          }
11          else
12          {
13              txt_ASCII.Text = string.Empty;                      //输出空字符串
14              MessageBox.Show("请输入字母！", "提示！");          //提示用户信息
15          }
16      }
17  }
18  private void btn_ToChar_Click(object sender, EventArgs e)
19  {
20      if (txt_ASCII2.Text != string.Empty)                        //判断输入是否为空
21      {
22          int P_int_Num;                                          //定义整型局部变量
23          if (int.TryParse(                                       //将输入的字符转换为数字
24              txt_ASCII2.Text, out P_int_Num))
25          {
26              txt_Char2.Text =
27                  ((char)P_int_Num).ToString();                   //将ASCII码转换为字符
28          }
29          else
30          {
31              MessageBox.Show(                                    //如果输入不符合要求，则弹出提示框
32                  "请输入正确的ASCII码值。", "错误！");
33          }
```

```
34        }
35    }
```

扩展学习

将字母显式转换为数值会得到字符的 ASCII 码值

Char 是值类型，可以将字母显式转换为整数数值，从而方便地得到字母的 ASCII 码。同样，可以将整数数值显式转换为 Char，从而得到字母。

实例 006　汉字与区位码的转换

源码位置：Code\01\006

实例说明

区位码是一个 4 位的十进制数，每个区位码都对应着一个唯一的汉字，区位码的前两位叫作区码，后两位叫作位码。考生在填写高考信息表时会用到汉字区位码，如在报考志愿表中也需要填写汉字区位码。在程序中可以方便地将汉字转换为区位码。实例运行效果如图 1.6 所示。

图 1.6　汉字与区位码的转换

关键技术

本实例重点在于向读者介绍将汉字字符转换成区位码的方法。转换汉字区位码的过程十分简单，首先通过 Encoding 对象的 GetBytes 方法获取汉字的字节数组，将字节数组的第一位和第二位分别转换为整型数值，然后将得到的两个整型数值分别减 160 后转换为字符串，连接两个字符串就组成了汉字区位码。

注意

Encoding对象的GetBytes方法提供了多个重载，可以接收字符串、字符数组等对象。

实现过程

（1）打开 Visual Studio 2022 开发环境，新建一个名为 ChineseCode 的 Windows 窗体应用程序。

（2）更改默认窗体 Form1 的 Name 属性为 Frm_Main，更改 Text 属性为“汉字与区位码的转换”，向窗体中添加一个 GroupBox 控件，向 GroupBox 控件中添加两个 TextBox 控件，分别用于输入汉字信息和输出区位码信息；向 GroupBox 控件中添加一个 Button 控件，用于将汉字转换为区位码。

（3）程序主要代码如下：

006-1

```
01  private void btn_Get_Click(object sender, EventArgs e)
02  {
03      if (txt_Chinese.Text != string.Empty)                          //判断输入是否为空
04      {
05          try
06          {
07              txt_Num.Text =                                         //获取汉字区位码信息
08                  getCode(txt_Chinese.Text);
09          }
10          catch (IndexOutOfRangeException ex)
11          {
12              MessageBox.Show(                                       //使用消息对话框提示异常信息
13                  ex.Message + "请输入正确的汉字", "出错！");
14          }
15      }
16  }
17  ///<summary>
18  ///获取汉字区位码方法
19  ///</summary>
20  ///<param name="strChinese">汉字字符</param>
21  ///<returns>返回汉字区位码</returns>
22  public string getCode(string Chinese)
23  {
24      string P_str_Code = "";
25      byte[] P_bt_array = new byte[2];                               //定义一个字节数组，用于存储汉字
26      P_bt_array = Encoding.Default.GetBytes(Chinese);               //为字节数组赋值
27      int front = (short)(P_bt_array[0] - '\0');                     //将字节数组的第一位转换成int类型
28      int back = (short)(P_bt_array[1] - '\0');                      //将字节数组的第二位转换成int类型
29      P_str_Code = (front - 160).ToString() + (back - 160).ToString();      //计算区位码
30      return P_str_Code;                                                    //返回区位码
31  }
```

扩展学习

使用 FileStream 对象将字节数组写入文件

使用 Encoding 对象的 GetBytes 方法可以获取字符串对象的字节数组，现在可以创建一个 FileStream 对象，方便地将字节数组写入文件中。同样，也可以从文件中读取字节数组，然后调用 Encoding 对象的 GetString 方法将字符数组转换为字符串。

实例 007　将汉字转换为拼音

源码位置：Code\01\007

实例说明

我们经常使用拼音或五笔等输入法向文档中输入汉字。使用拼音输入法获取汉字的过程是通过用户输入的拼音，输入法会智能地匹配到相应的汉字或汉字中的词并输出。本实例将介绍一个很有趣的功能，即将汉字转换为拼音。实例运行效果如图 1.7 所示。

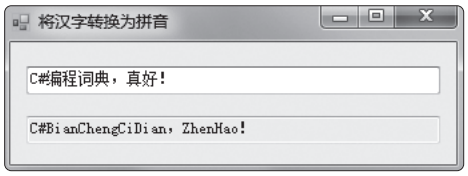

图 1.7　将汉字转换为拼音

关键技术

本实例使用了 PinYin 类，PinYin 类使用了正则表达式和 ToCharArray 方法。

> **注意**　正则表达式经常用于数字或字符串等信息的验证及提取。

实现过程

（1）打开 Visual Studio 2022 开发环境，新建一个名为 ChineseToABC 的 Windows 窗体应用程序。

（2）更改默认窗体 Form1 的 Name 属性为 Frm_Main，向窗体中添加两个 TextBox 控件，分别用于输入汉字信息和输出拼音信息。

（3）程序主要代码如下：

007-1

```
///<summary>
///将汉字转换为拼音的方法
///</summary>
///<param name="str">汉字字符串</param>
///<returns>拼音字符串</returns>
public string GetABC(string str)
{
    Regex reg = new Regex("^[\u4e00-\u9fa5]$");          //验证输入是否为汉字
    byte[] arr = new byte[2];                            //定义字节数组
    string pystr = "";                                   //定义字符串变量并添加引用
    char[] mChar = str.ToCharArray();                    //获取汉字对应的字符数组
    return GetStr(mChar, pystr, reg, arr);               //返回获取到的汉字拼音
}
```

扩展学习

使用正则表达式，可以方便地操作字符串，对字符串进行验证或提取。在本实例中，使用正则表达式验证字符串中的每一个字符是否为汉字，如果字符为汉字，则查找汉字的拼音。

实例 008　从字符串中分离文件路径、文件名及扩展名

源码位置：Code\01\008

实例说明

对文件进行操作时，首先要获取文件路径信息，然后创建文件对象，通过 I/O 流将

数据读取到内存中并进行处理。在操作文件过程中可能还需要提取文件的一些信息，如文件的路径、文件名及文件扩展名。实例运行效果如图 1.8 所示。

图 1.8　从字符串中分离文件路径、文件名及扩展名

关键技术

本实例使用字符串对象的 Substring 方法截取字符串，使用 LastIndexOf 方法查找字符或字符串在指定字符串中的索引。

实现过程

（1）打开 Visual Studio 2022 开发环境，新建一个名为 FilePathString 的 Windows 窗体应用程序。

（2）更改默认窗体 Form1 的 Name 属性为 Frm_Main，向窗体中添加 3 个 Label 控件，分别用于输出文件的路径、文件名及扩展名；添加一个 Button 控件，用于处理字符串中的路径信息。

（3）程序主要代码如下：

008-1

```
01  private void btn_Openfile_Click(object sender, EventArgs e)
02  {
03      if (openFileDialog1.ShowDialog() == DialogResult.OK)                //判断是否选择了文件
04      {
05          string P_str_all = openFileDialog1.FileName;                    //记录选择的文件全路径
06          //获取文件路径
07          string P_str_path = P_str_all.Substring(0, P_str_all.LastIndexOf("\\") + 1);
08          string P_str_filename =                                         //获取文件名
09              P_str_all.Substring(P_str_all.LastIndexOf("\\") + 1,
10              P_str_all.LastIndexOf(".") -
11              (P_str_all.LastIndexOf("\\") + 1));
12          string P_str_fileexc =                                          //获取文件扩展名
13              P_str_all.Substring(P_str_all.LastIndexOf(".") + 1,
14              P_str_all.Length - P_str_all.LastIndexOf(".") - 1);
15          lb_filepath.Text = "文件路径：  " + P_str_path;                  //显示文件路径
16          lb_filename.Text = "文件名称：  " + P_str_filename;              //显示文件名称
17          lb_fileexc.Text = "文件扩展名：  " + P_str_fileexc;              //显示文件扩展名
18      }
19  }
```

扩展学习

IndexOf 方法与 LastIndexOf 方法的异同

使用 IndexOf 方法与 LastIndexOf 方法都可以用来查找字符或字符串在指定字符串对象中的索