一、 基础开发实验资源

A0101 指导文档 Button 控件学习

1. 实验目的

该实验主要是让学生熟悉使用 Button 控件。

2. 实验设备

软件：visualstudio2010 及以上版本

3. 实验原理

3.1 Button 类

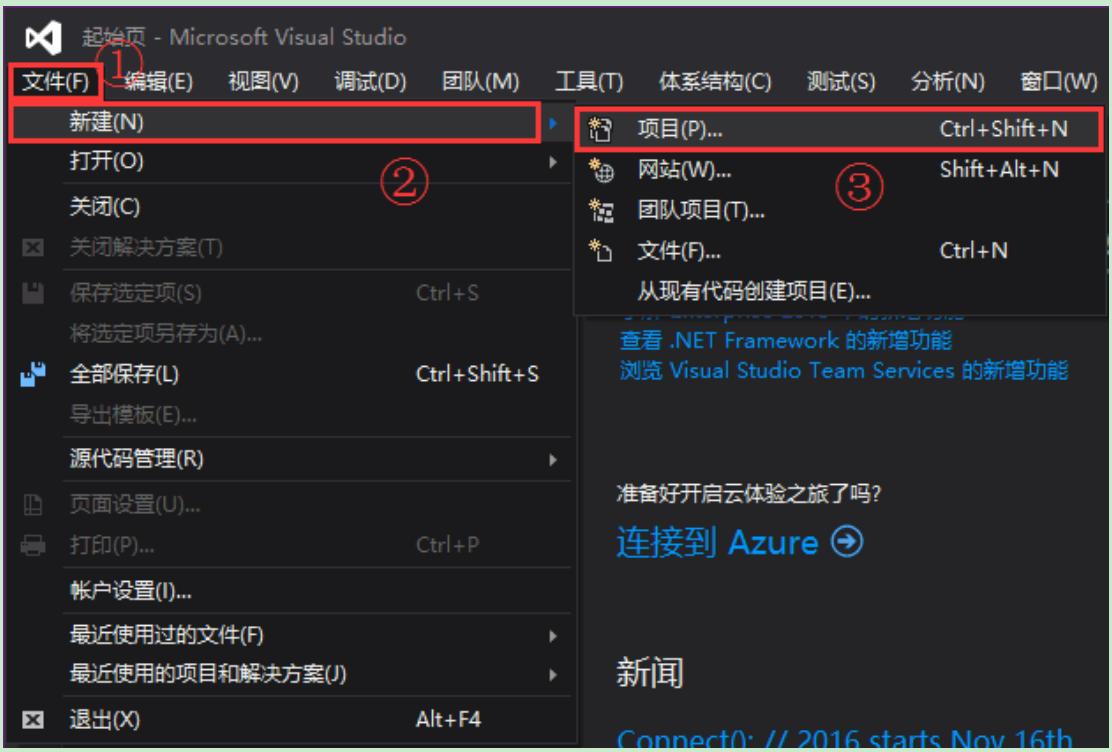
表示 Windows 按钮控件。
所属命名空间:System.Windows.Forms

3.2 关闭串口 Button 控件

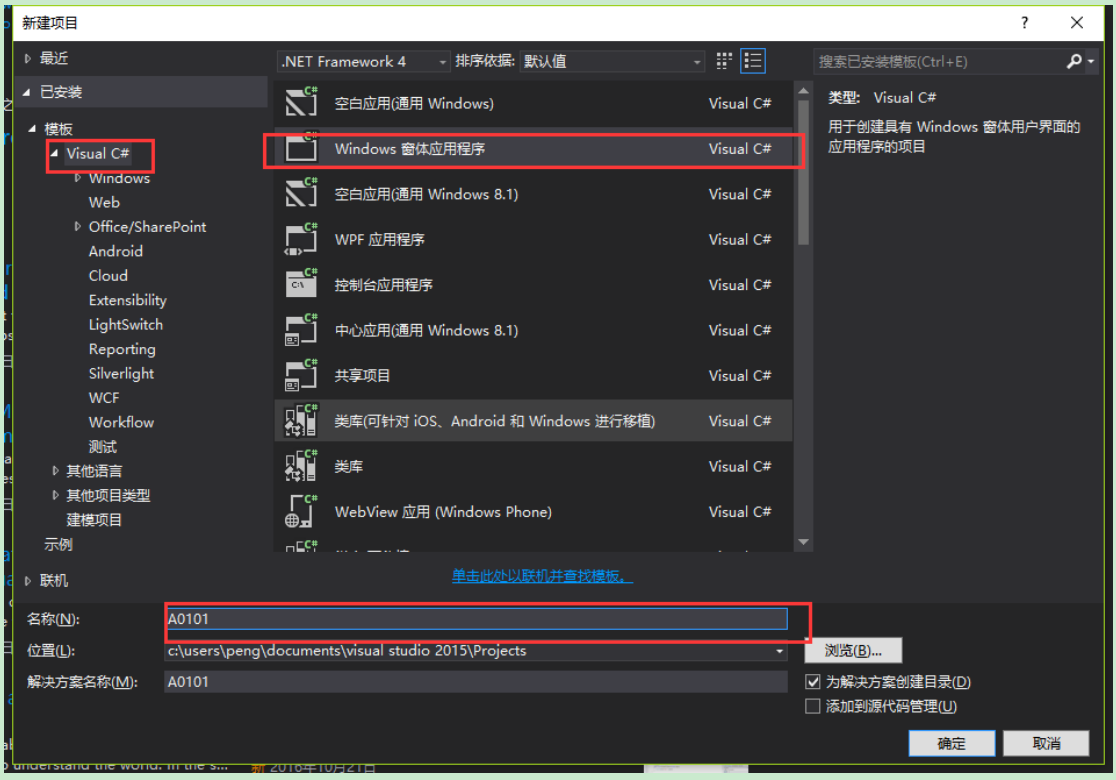
Windows 窗体 Button 控件允许用户通过单击来执行操作。Button 控件既可以显示文本，又可以显示图像。当该按钮被单击时，它看起来像是被按下，然后被释放。

4. 实验设计

（1）启动 visualstudio，文件→新建→项目。

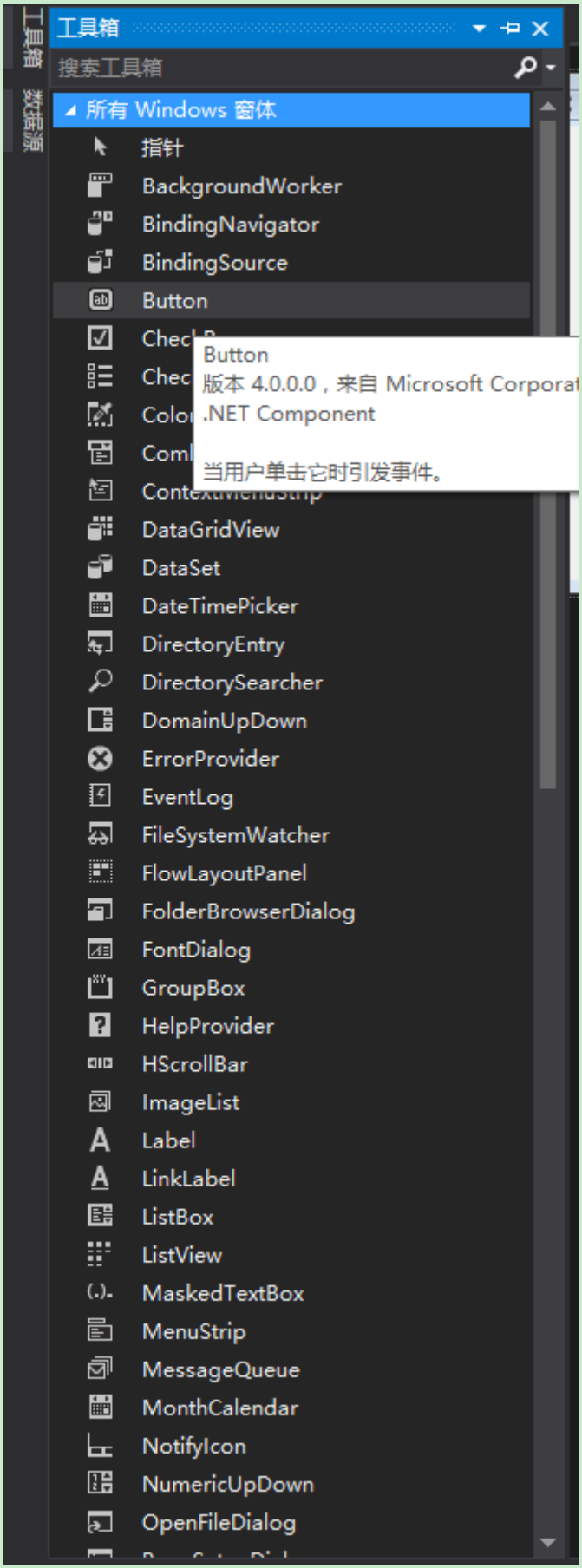


（2）选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。

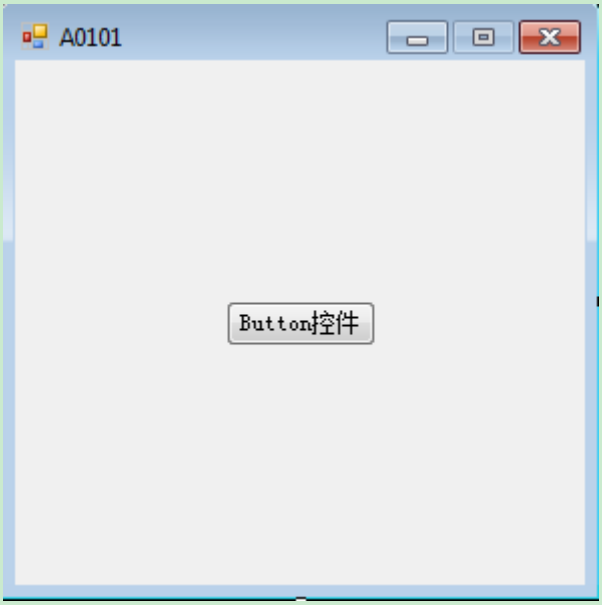


（3）界面设计及控件属性

在工具箱中找到 Button 按钮，双击或者拖拽都可以添加控件到窗体中。


工具箱
数据源
工具箱
搜索工具箱
所有 Windows 窗体
指针
BackgroundWorker
BindingNavigator
BindingSource
Button
Button
版本 4.0.0.0，来自 Microsoft Corporat
.NET Component
当用户单击它时引发事件。
DataGridView
DataSet
DateTimePicker
DirectoryEntry
DirectorySearcher
DomainUpDown
ErrorProvider
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowserDialog
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog

控件名称	Text 属性	Name 属性	功能
Form 窗体	A0101	FrmMain	
Button 控件	Button 控件	button1	鼠标单击后弹出控件名称



（4）控件分析：

常用属性及介绍：

Name：设置该控件的名称。

BackColor ：设置该控件的背景颜色，使用 Web 栏下的 Transparent（透明）可将控件设置为透明状。

BackgroundImage:设置该控件的背景图片。

BackgroundImageLayout：设置该控件背景图片的布局，一般选择 Stretch。

FlatStyle：设置控件外观。

FlatAppearance：此属性设置只对 FlatStyle 为 Flat 时有效；

其中：

BorderColor：设置按钮周围的边框颜色，BorderSize：设置按钮周围的边　框大小；

MouseDownBackColor：设置按钮被按下时工作区的颜色；

MouseOverBackColor：设置当鼠标指针位于控件边界内时按钮工作区的颜色。

Font：设置字体样式和大小。

ForeColor：设置字体颜色。

Image：在控件上显示图像。

Text：设置控件中显示的文本。

常用事件及介绍：

Click 事件：当用户用鼠标左键单击按钮控件时，将发生该事件。

MouseDown 事件：当用户在按钮控件上按下鼠标按钮时，将发生该事件。

MouseUp 事件：当用户在按钮控件上释放鼠标按钮时，将发生该事件。

添加按钮单击事件 button1_Click：

5. 实验代码解析

按钮单击事件代码如下所示：

```
  privatevoid button1_Click(object sender, EventArgs e)
    {
MessageBox.Show("控件的名称是：" + button1.Name);
    }
```

A0102 指导文档　CheckBox 控件学习

1. 实验目的

该实验主要是为了让学生熟悉使用 CheckBox 控件。

2. 实验设备

软件：visualstudio2010 及以上版本，

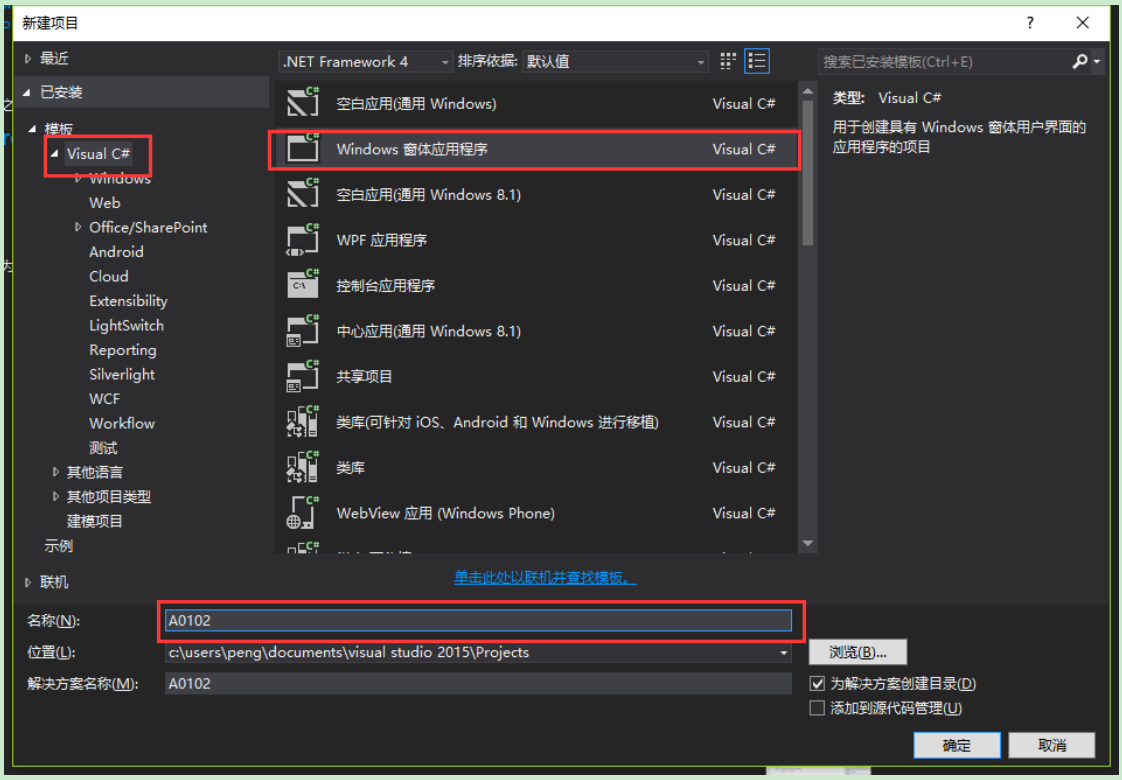
3. 实验原理

CheckBox 类
表示 Windows CheckBox 控件。
所属命名空间:System.Windows.Forms
CheckBox 控件
复选框，也叫做 CheckBox，是一种基础控件。.NET 的工具箱里包含这个控件，它可以通过其属性和方法完成复选的操作。为了完成更多复杂的需求，也出了第三方控件的复选框。只需要将其 dll 添加到工具箱里，就可以使用更多功能的复选框控件。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

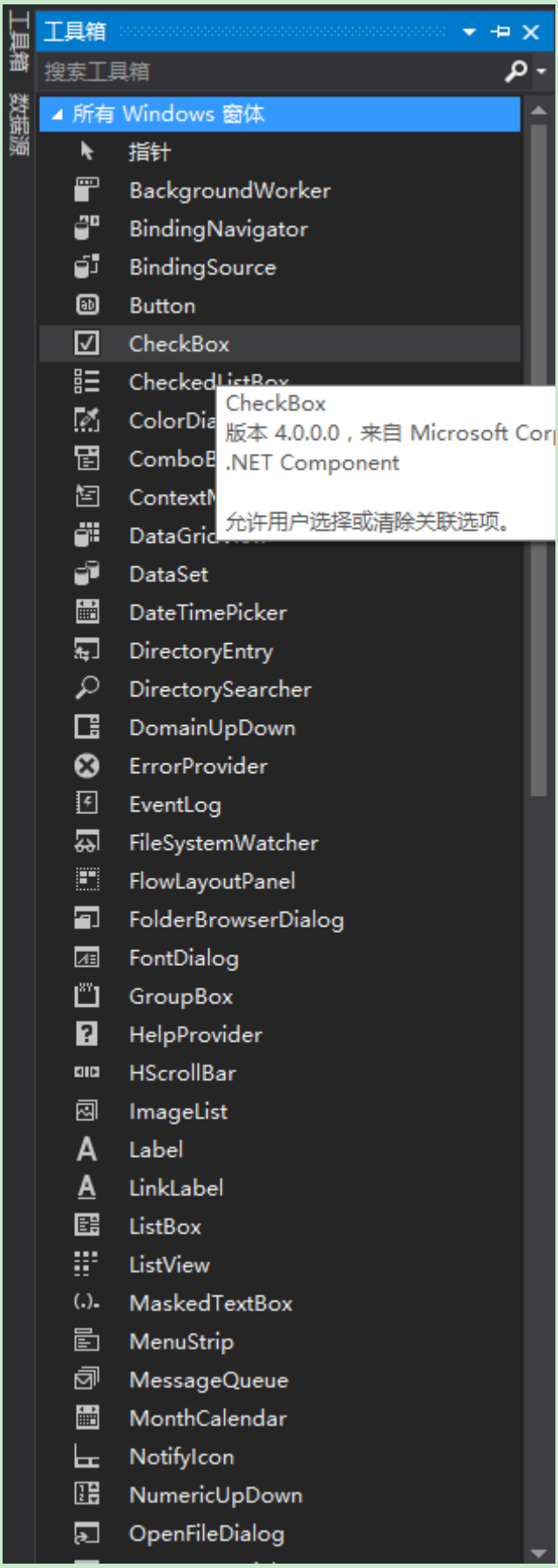
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 CheckBox 控件，双击或者拖拽都可以添加控件到窗体中。


工具箱
工具箱
搜索工具箱
所有 Windows 窗体
指针
BackgroundWorker
BindingNavigator
BindingSource
Button
CheckBox
CheckedListBox
ColorDia
ComboB
ContextM
DataGrid
DataSet
DateTimePicker
DirectoryEntry
DirectorySearcher
DomainUpDown
ErrorProvider
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowserDialog
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
CheckBox
版本 4.0.0.0，来自 Microsoft Cor
.NET Component
允许用户选择或清除关联选项。

控件名称	Text 属性	Name 属性	功能
Form 窗体	A0102	FrmMain	
CheckBox 控件	CheckBox 控件	checkBox1	获取勾选的 CheckBox 控件 Text 值， CheckBox 勾选与否事件弹出提示

4、控件分析：

常用属性及介绍

TextAlign 属性：用来设置控件中文字的对齐方式。

ThreeState 属性:用来返回或设置复选框是否能表示三种状态,如果属性值为 true 时，表示可以表示三种状态—选中、没选中和中间态（CheckState.Checked、CheckState.Unchecked 和 CheckState.Indeterminate），属性值为 false 时，只能表示两种状 态——选中和没选中。

Checked 属性：用来设置或返回复选框是否被选中，值为 true 时，表示复选框被选中，值为 false 时，表示复选框没被选中。当 ThreeState 属性值为 true 时，中间态也表示选中。

CheckState 属性:用来设置或返回复选框的状态。在 ThreeState 属性值为 false 时，取值有 CheckState.Checked 或 CheckState.Unchecked。在 ThreeState 属性值被设置为 True 时，CheckState 还可以取值 CheckState.Indeterminate，在此时，复选框显示为浅灰色选中状态，该状态通常表示该选项下的多个子选项未完全选中。

常用事件及介绍：

CheckStateChanged 事件：每当更改 CheckState 属性时发生。

Click 事件：单击组件时发生。

添加状态更改事件 checkBox1_CheckStateChanged

5. 实验代码解析

控件状态更改事件代码如下所示：

```
privatevoid checkBox1_CheckStateChanged(object sender, EventArgs e)
    {
if (checkBox1.Checked)//判读是否选中该控件
        {
MessageBox.Show("你已勾选，其Text值为：" + checkBox1.Text);
        }
else
        {
MessageBox.Show("你已取消勾选！");
        }
    }
```

A0103 指导文档　ContextMenuStrip 控件学习

1. 实验目的

该实验主要是为了让学生熟悉使用 ContextMenuStrip 控件。

2. 实验设备

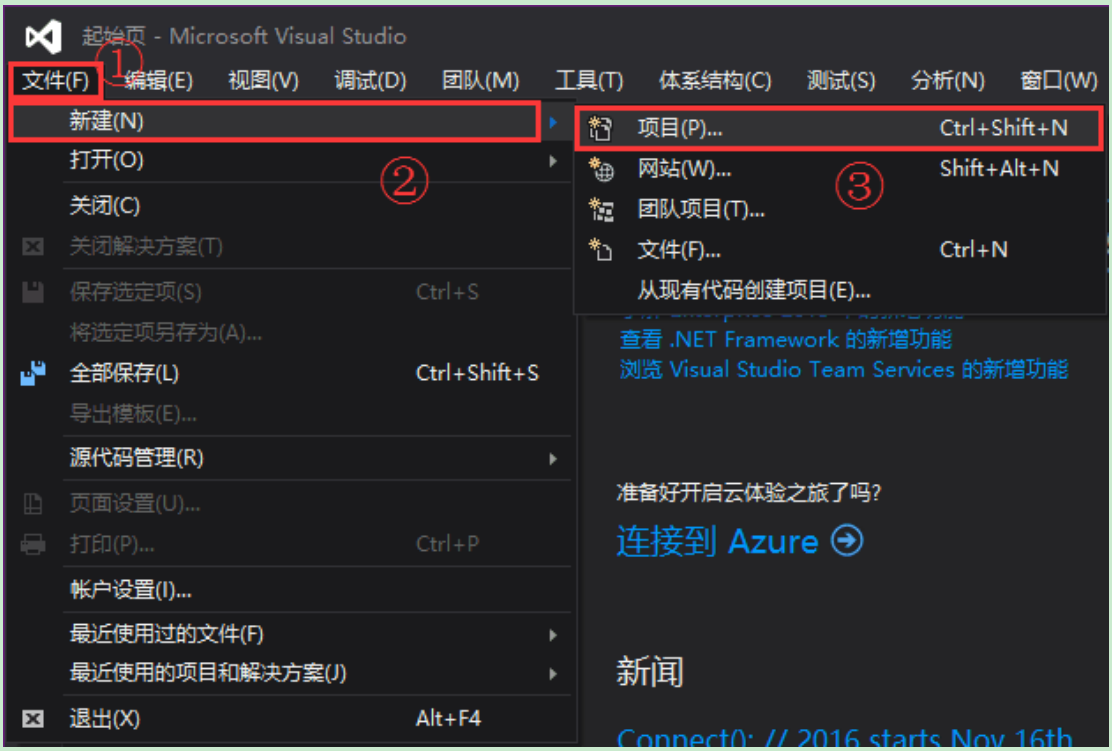
软件：visualstudio2010 及以上版本，

3. 实验原理

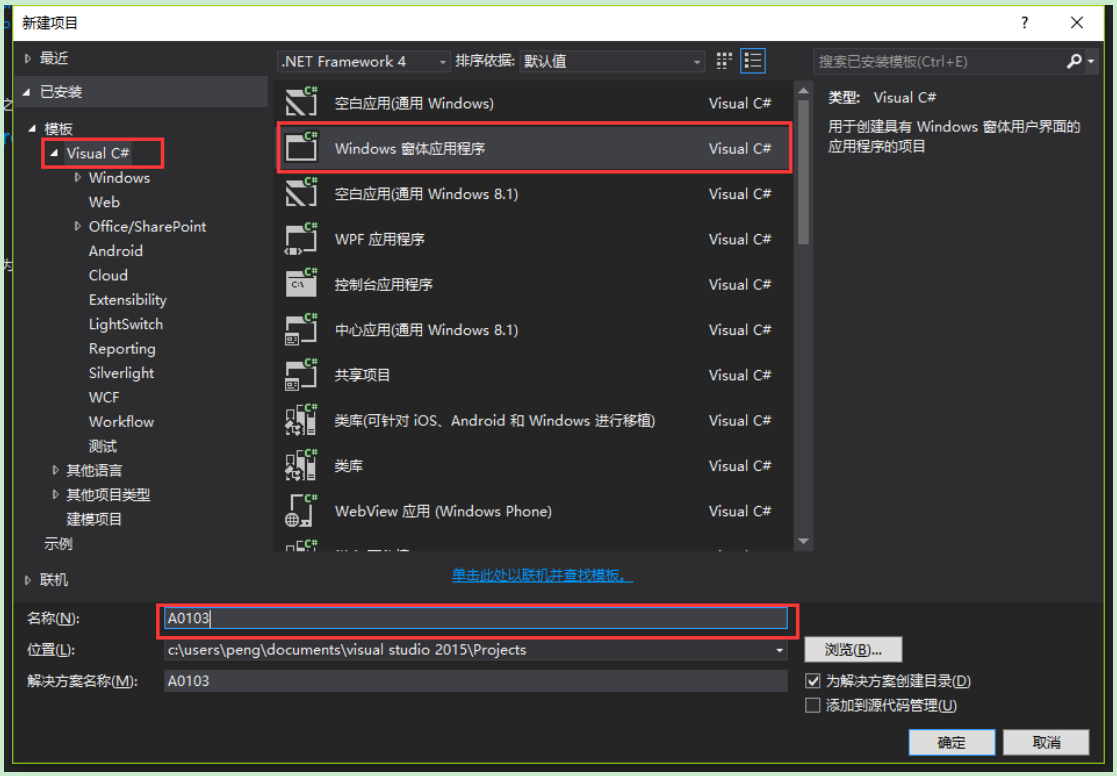
ContextMenuStrip 类
表示快捷菜单。
所属命名空间:System.Windows.Forms
ContextMenuStrip 控件
要显示弹出菜单,或在用户右击鼠标时显示一个菜单,就应使用 ContextMenuStrip 类。与 MenuStrip 一样，ContextMenuStrip 也是 ToolStripMenuItems 对象的容器，但它派生于 ToolStripDropDownMenu 。 ContextMenu 的创建与 MenuStrip 相同，也是添加 ToolStripMenuItems，定义每一项的 Click 事件，执行某个任务。弹出菜单应赋予特定的控件，为此，要设置控件的 ContextMenuStrip 属性。在用户右击该控件时，就显示该菜单。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。



2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。

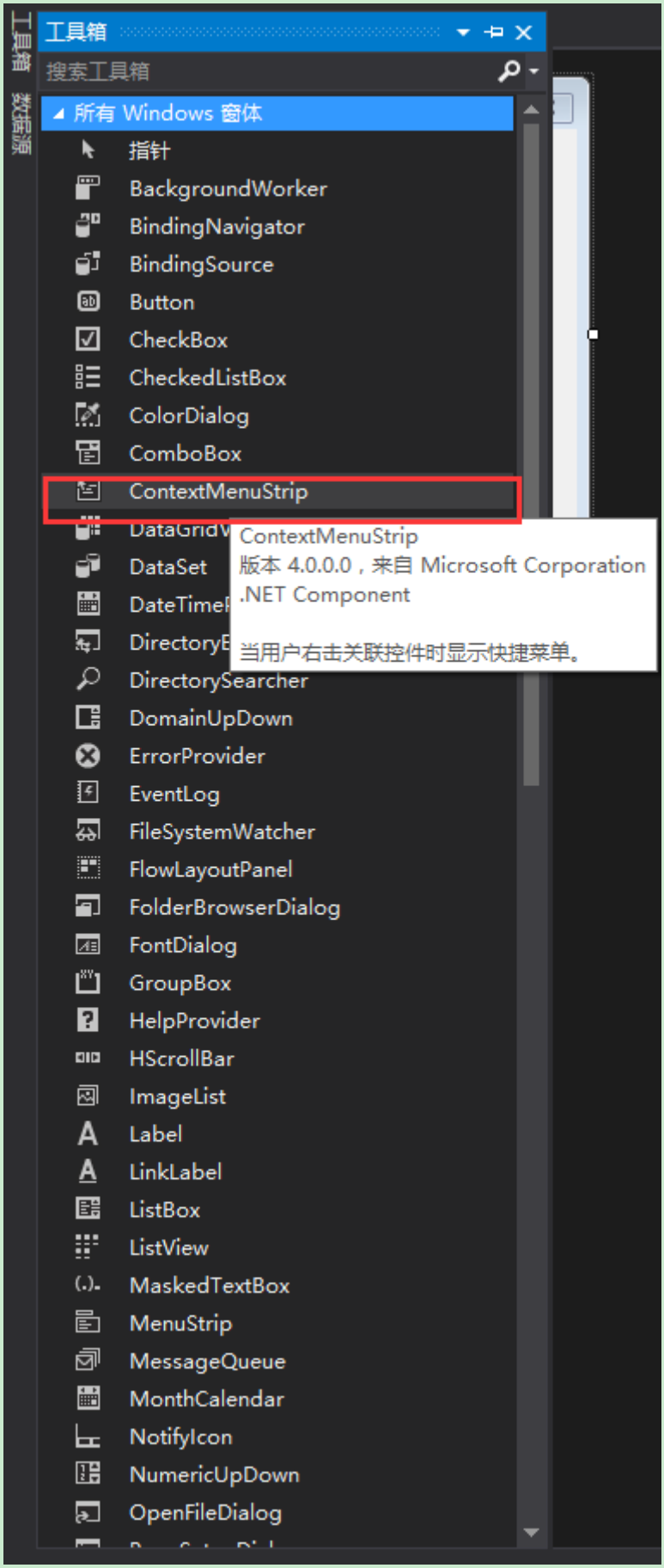


3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜

单，找到工具箱，如下图所示

在工具箱中找到 ContextMenuStrip 控件，双击或者拖拽都可以添加控件到窗体中


工具箱
工具箱
数据源
搜索工具箱
所有 Windows 窗体
指针
BackgroundWorker
BindingNavigator
BindingSource
Button
CheckBox
CheckedListBox
ColorDialog
ComboBox
ContextMenuStrip
DataSet
DirectorySearcher
DomainUpDown
ErrorProvider
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowserDialog
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
ContextMenuStrip
版本 4.0.0.0，来自 Microsoft Corporation
.NET Component
当用户右击关联控件时显示快捷菜单。

控件名称	Text 属性	Name 属性	功能
Form 窗体	A0103	FrmMain	
ContextMenuStrip 控件		contextMenuStrip1	展示 ContextMenuStrip 控件的 Items 集合，在控件上右键能弹出 ContextMenuStrip

4、控件解析

常用属性及介绍

Name 属性：设置该控件的名称。

BackClore 属性：设置该控件的背景颜色。

BackgroundImage 属性：设置该控件的背景图片。

BackgroundImageLayout 属性：设置该控件的背景图片布局。

Enabled 属性：指示是否启用该控件，True 为启用，False 为不启用。

Items 属性：设置在 ToolStrip 上显示的项的集合，即右击关联的控件时出现的快捷菜单。

ShowImageMargin 属性：指定是否显示图像边距。

ShowItemToolTips 属性：指定是否显示项的 ToolTip。

ShowCheckMargin 属性：指定是否显示选中边距。

常用事件及介绍：

Opened 事件：当 DropDown 已打开时发生。即右击弹出 Items 集合后发生。

Opening 事件：当 DropDown 正在打开时发生。即正在右击弹出 Items 集合时发生。

ItmeClicked 事件：当单击项时发生。即单击 Itmens 集合中的项时发生。

添加窗体启动事件 FrmMain_Load

添加选项单击事件 toolStripMenuItem1_Click

5. 实验代码解析

请大家了解 ContextMenuStrip 控件的原理，然后根据示例代码编写程序

窗体启动事件代码如下所示：

```
privatevoid FrmMain_Load(object sender, EventArgs e)
{
        contextMenuStrip1.Items.Add("复制");
        contextMenuStrip1.Items.Add("粘贴");
        contextMenuStrip1.Items.Add("撤销");
        contextMenuStrip1.Items.Add("添加");
}
```

添加选项单击事件代码如下所示：

```
privatevoid toolStripMenuItem1_Click(object sender, EventArgs e)
{
MessageBox.Show("大家好");
}
```

A0104 指导文档　datetimepicker 控件学习

1. 实验目的

主要目的是知道如何应用 datetimepicker 控件。

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

datetimepicker 类

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。

所属命名空间:System.Windows.Forms

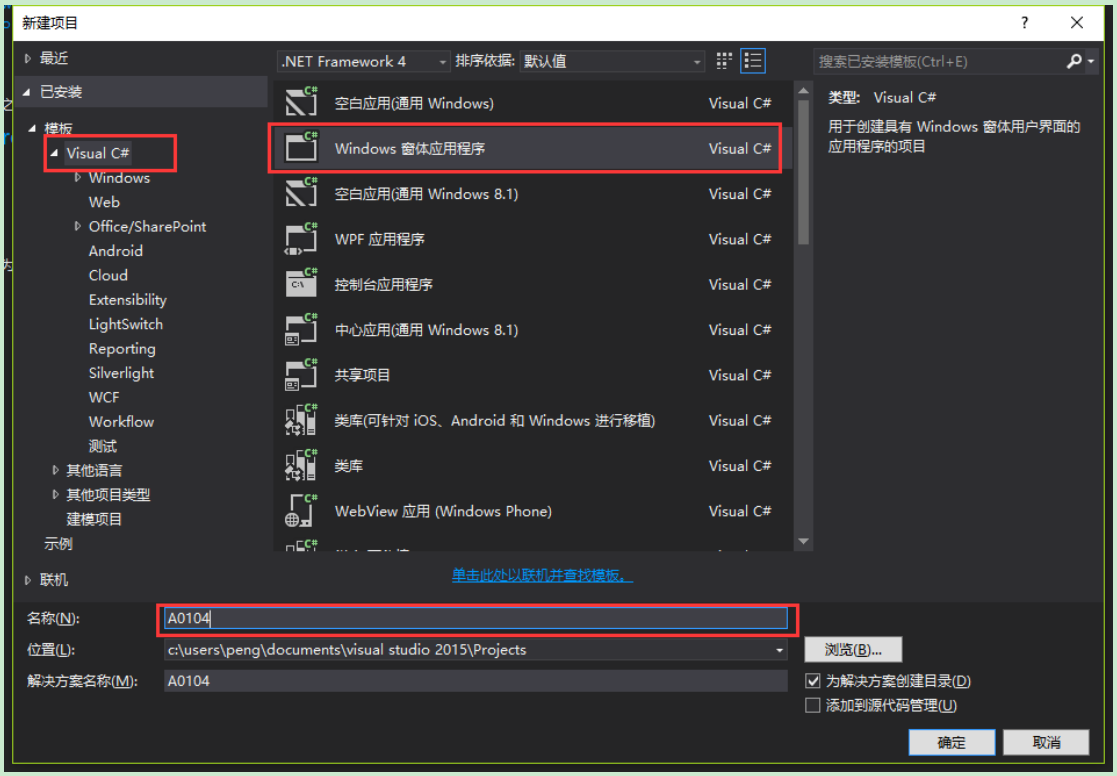
datetimepicker 控件

DateTimePicker 控件，控件一般用于让用户可以从日期列表中选择单个值。运行时，单击控件边上的下拉箭头，会显示为两个部分：一个下拉列表，一个用于选择日期的。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。

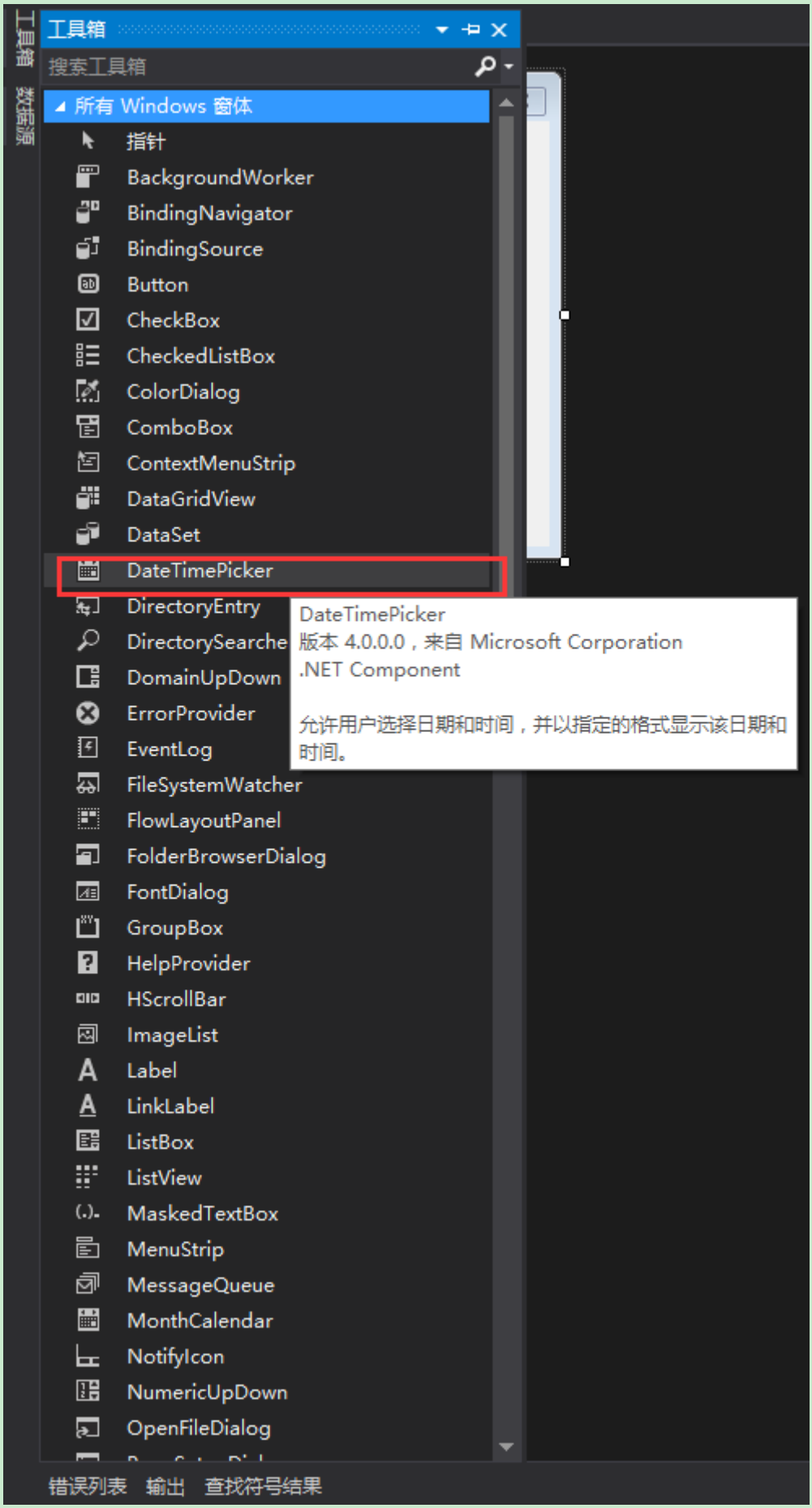


3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜

单，找到工具箱，如下图所示

在工具箱中找到 datetimepicker 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
所有 Windows 窗体
指针
BackgroundWorker
BindingNavigator
BindingSource
Button
CheckBox
CheckedListBox
ColorDialog
ComboBox
ContextMenuStrip
DataGridView
DataSet
DateTimePicker
DirectoryEntry
DirectorySearche
DomainUpDown
ErrorProvider
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowserDialog
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
DateTimePicker
版本 4.0.0.0，来自 Microsoft Corporation
.NET Component
允许用户选择日期和时间，并以指定的格式显示该日期和时间。
错误列表
输出
查找符号结果

控件名称	控件 Text 属性	控件 Name 属性	功能
Form1 窗体	A0104	frmMain	
datetimepicker 控件			获取选择时间
Button 控件	格式一:	button1	单击触发事件
Button 控件	格式二:	button2	单击触发事件

格式一如下图所示：

格式二如下图所示：

4、控件解析：

常用属性及介绍：

Name：设置该控件的名称。

BackColor ：设置该控件的背景颜色，使用系统栏下的 Highlight。

BackgroundImage:设置该控件的背景图片。

BackgroundImageLayout：设置该控件背景图片的布局，一般选择 Tile

FlatStyle：设置控件外观。

Font：设置字体样式和大小。

ForeColor：设置字体颜色。

Image：在控件上显示图像。

Text：设置控件中显示的文本。

常用事件及介绍
Click 事件：当用户用鼠标左键单击按钮控件时，将发生该事件。
添加格式一按钮单击事件 button1_Click
添加格式二按钮单击事件 button2_Click

5. 实验代码解析

请大家了解 datetimepicker 控件的原理，然后根据示例代码编写程序
格式一按钮单击事件代码如下所示：

```
private void button1_Click(object sender, EventArgs e)
{
  // 将 datetimepicker1 的格式属性设为自定义:
        this.dateTimePicker1.Format = DateTimePickerFormat.Custom;
 //自定义日期时间的显示格式一:
        this.dateTimePicker1.CustomFormat = "yyyy-MM-dd HH:mm:ss";
//获取系统的本地时间，通过自定义的格式一显示:
        this.dateTimePicker1.Value = DateTime.Now;
 }
```

格式二按钮单击事件代码如下所示：

```
private void button2_Click(object sender, EventArgs e)
 {
     // 将 datetimepicke1r 的格式属性设为自定义:
        this.dateTimePicker1.Format = DateTimePickerFormat.Custom;
     //自定义日期时间的显示格式二:
        this.dateTimePicker1.CustomFormat = "yyyy/MM/dd hh:mm:ss";
     //获取系统的本地时间，通过自定义的格式二显示:
        this.dateTimePicker1.Value = DateTime.Now;
 }
```

A0105 指导文档　FlowLayoutPanel 控件学习

1. 实验目的

主要目的是掌握如何向 FlowLayoutPanel 的添加子控件并显示。

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

FlowLayoutPanel 类

表示一个面板，它可以进行动态水平或垂直布局其内容。

所属命名空间:System.Windows.Forms

FlowLayoutPanel 控件

FlowLayoutPanel 控件在水平或垂直流方向排列其内容。可以将该控件的内容从一行换行至下一行，或者从一列换至下一列。还可以选择剪裁内容而不是换行。

可以通过设置 FlowDirection 属性的值来指定流方向。FlowLayoutPanel 控件在从右向左（RTL）的布局中正确地反转它的流方向。还可以通过设置 WrapContents 属性的值来指定是换行还是剪裁 FlowLayoutPanel 控件的内容。

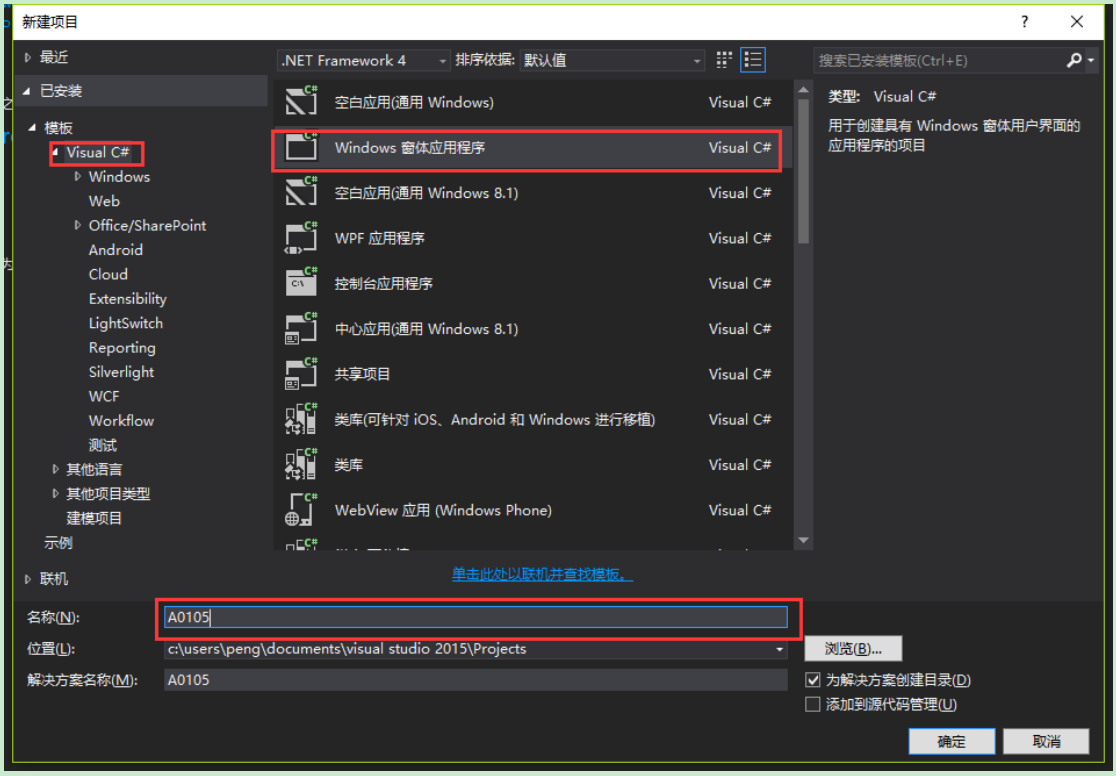
将 AutoSize 属性设置为 true 时，FlowLayoutPanel 控件自动调整大小以容纳其内容。它还向其子控件提供了 FlowBreak 属性。将 FlowBreak 属性的值设置为 true 会使 FlowLayoutPanel 控件停止在当前流方向布局控件并换到下一行或下一列。

任何 Windows 窗体控件都可以是 FlowLayoutPanel 控件的子控件，包括 FlowLayoutPanel 的其他实例。利用此功能，可以在运行时构造适应窗体尺寸的复杂布局。

4. 实验设计

1、 启动 visualstudio，文件→新建→项目。

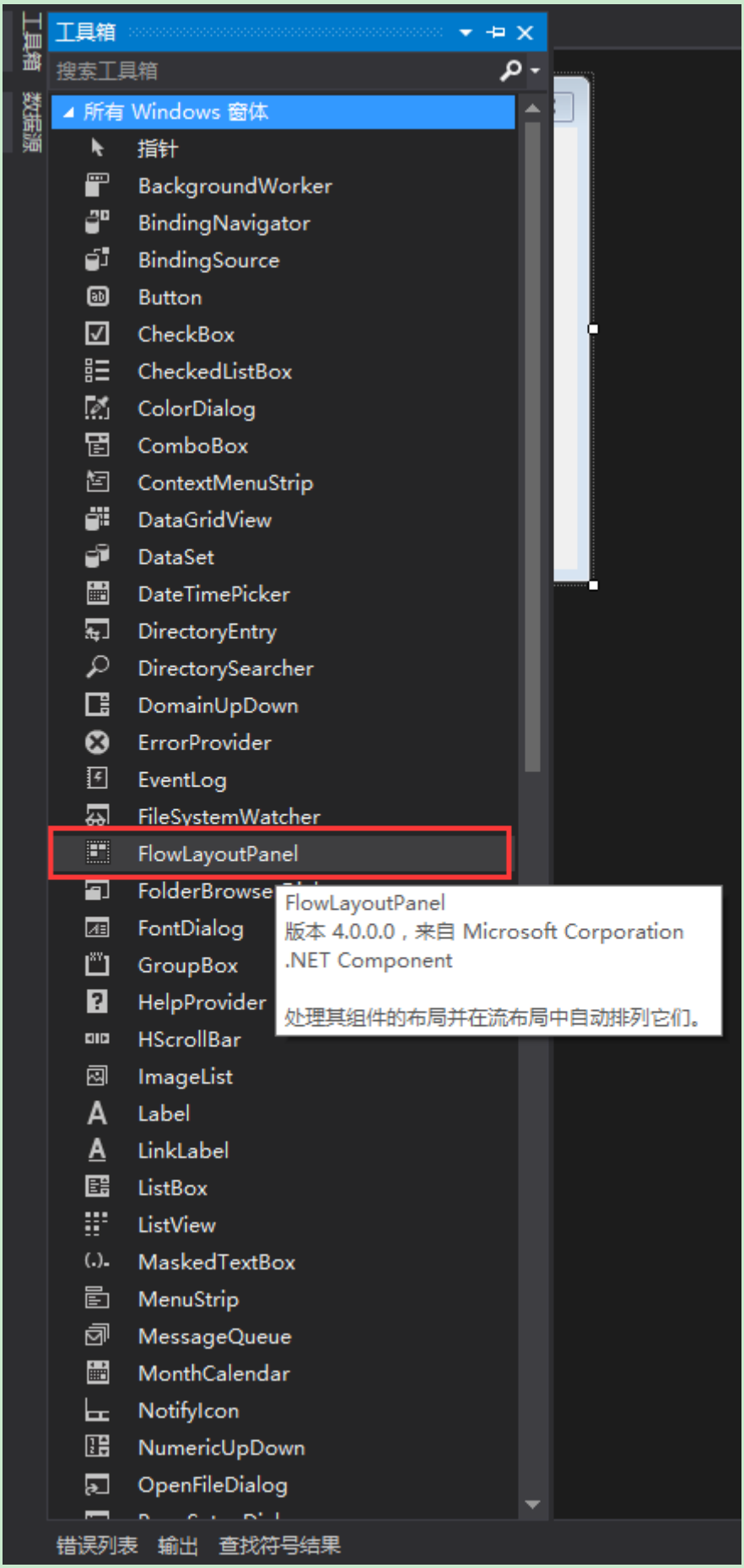
2、选择 Visual C#→Windows 窗体应用程序，输入名称→选择存储路径。



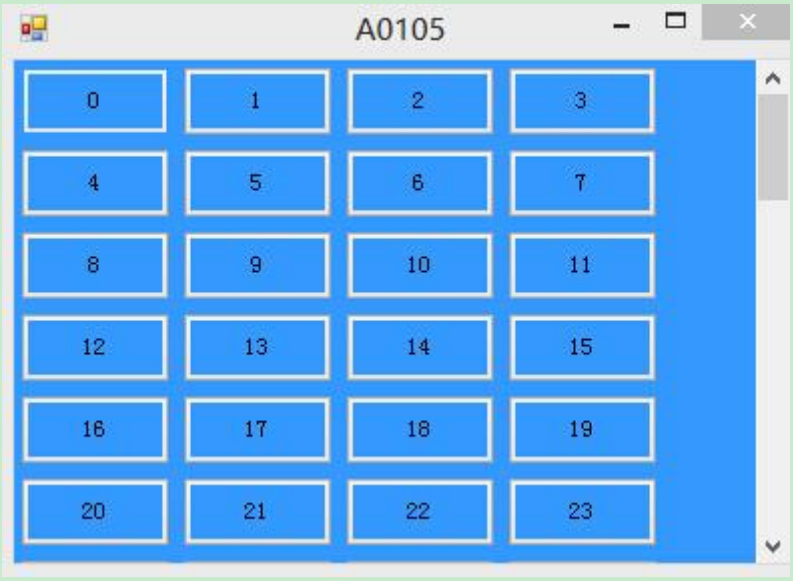
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 FlowLayoutPanel 控件，双击或者拖拽都可以添加控件到窗体中


工具箱
数据源
工具箱
搜索工具箱
所有 Windows 窗体
指针
BackgroundWorker
BindingNavigator
BindingSource
Button
CheckBox
CheckedListBox
ColorDialog
ComboBox
ContextMenuStrip
DataGridView
DataSet
DateTimePicker
DirectoryEntry
DirectorySearcher
DomainUpDown
ErrorProvider
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowse
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
FlowLayoutPanel
版本 4.0.0.0，来自 Microsoft Corporation
.NET Component
处理其组件的布局并在流布局中自动排列它们。
错误列表
输出
查找符号结果

控件名称	控件 Text 属性	控　件 Name 属性	功能
Form1 窗体	A0105	frmMain	
FlowLayoutPanel 控件			自动水平或者垂直排放子控件



4、控件解析

常用属性及介绍：

Name：设置该控件的名称。

BackColor ：设置该控件的背景颜色，使用系统栏下的 Highlight。

BackgroundImage:设置该控件的背景图片。

BackgroundImageLayout：设置该控件背景图片的布局，一般选择 Tile。

FlatStyle：设置控件外观。

Font：设置字体样式和大小。

ForeColor：设置字体颜色。

Image：在控件上显示图像。

Text：设置控件中显示的文本。

常用事件及介绍

Load 事件：每当用户加载窗体时发生。

添加窗体加载事件 Form1_Load

5. 实验代码解析

窗体加载事件具体代码如下：

```
    private void Form1_Load(object sender, EventArgs e)
    {
        for( int i=0; i<100;i++)     //for 循环 100 次, flowLayoutPanel1 将显示出 100 个子控件
```

```
        {
            this.button1 = new System.Windows.Forms.Button();//实例化对象
            this.button1.Size = new System.Drawing.Size(75, 35);//获取或设置子控件的尺寸大小
             this.button1.Text = i.ToString();//获取子控件的文本
             this.flowLayoutPanel1.Controls.Add(this.button1);//将子控件添加到flowLayoutPanel1中，并且显示
             }
        }
```

A0106 指导文档　ListView 控件学习

1. 实验目的

该实验的主要目的是为了让学生熟悉使用 ListView 控件。

2. 实验设备

软件：visualstudio2010 及以上版本，

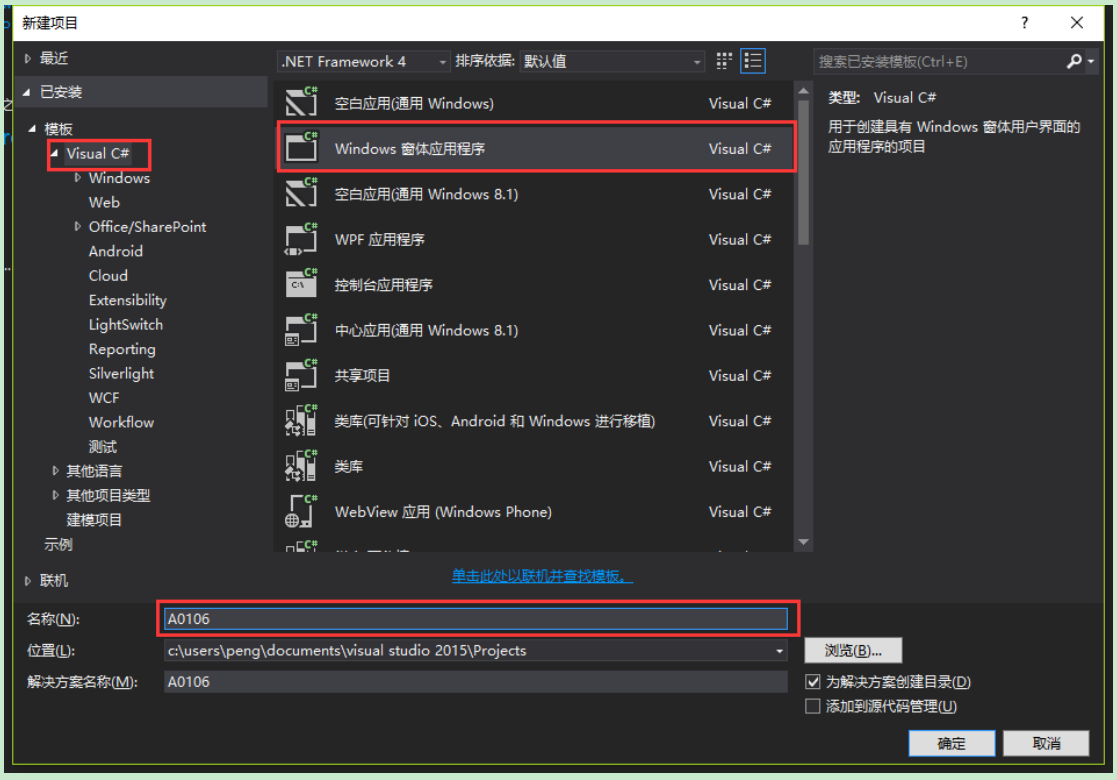
3. 实验原理

ListView 类
表示一个面板，它可以进行动态水平或垂直布局其内容。
所属命名空间:System.Web.UI.WebControls
ListView 控件
ListView 控件可使用四种不同视图显示项目。通过此控件，可将项目组成带有或不带有列标头的列，并显示伴随的图标和文本。 可使用 ListView 控件将称作 ListItem 对象的列表条目组织成下列四种不同的视图之一：1.大（标准）图标 2.小图标 3.列表 4.报表 View 属性决定在列表中控件使用何种视图显示项目。还可用 LabelWrap 属性控制列表中与项目关联的标签是否可换行显示。另外，还可管理列表中项目的排序方法和选定项目的外观。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

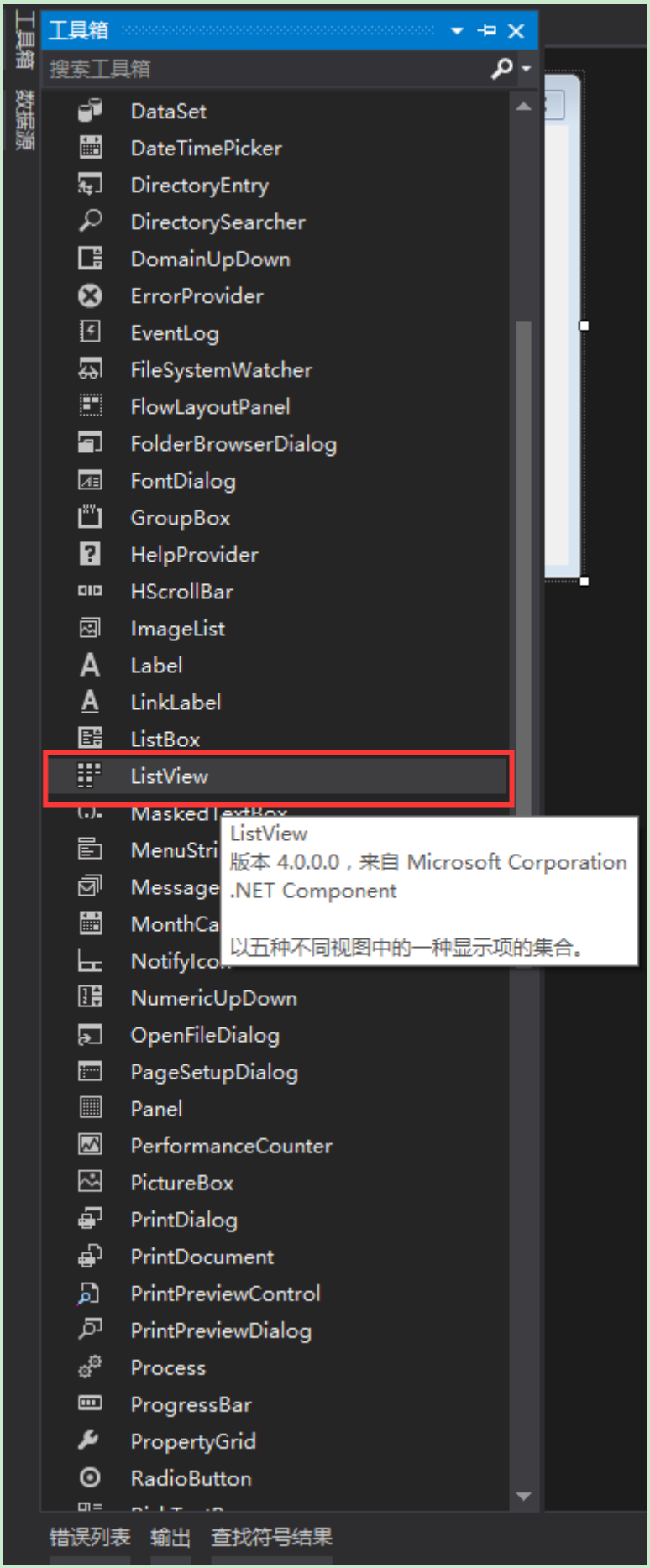
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



3、界面设计及控件属性

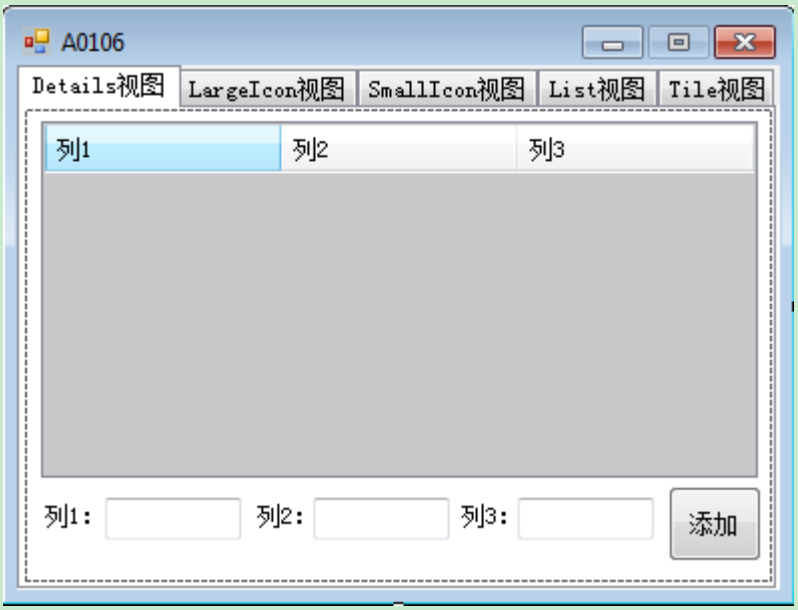
添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 ListView 控件。双击或者拖拽都可以添加控件到窗体中


工具箱
数据源
工具箱
搜索工具箱
DataSet
DateTimePicker
DirectoryEntry
DirectorySearcher
DomainUpDown
ErrorProvider
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowserDialog
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
ListView
版本 4.0.0.0，来自 Microsoft Corporation
.NET Component
以五种不同视图中的一种显示项的集合。
NumericUpDown
OpenFileDialog
PageSetupDialog
Panel
PerformanceCounter
PictureBox
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
PropertyGrid
RadioButton
错误列表
输出
查找符号结果

控件名称	Text 属性	Name 属性	功能
Form 窗体	A0106	FrmMain	
TabControl 控件		tabControl1	提供 ListView 各个不同视图的展示空间
ImageList 控件		iamgeList1	为 ListView 提供图像集合
ListView 控件		listView1	展示 ListView 的 Details 视图
		listView2	展示 ListView 的 LargeIcon 视图
		listView3	展示 ListView 的 SmallIcon 视图
		listView4	展示 ListView 的 List 视图
		listView5	展示 ListView 的 Title 视图
Label 控件	列 1：	label1	
	列 2：	label2	
	列 3：	label3	
TextBox 控件		txtbox_DColumn1	
		txtbox_DColumn2	
		txtbox_DColumn3	
Button 控件	添加	btn_DAdd	在 listView1 的 Detail 视图下添加 Items 集合中的一个项
	添加	btn_LAdd	在 listView2 的 LargeIcon 视图下添加 Items 集合中的一个项
	添加	btn_SAdd	在 listView3 的 SmallIcon 视图下添加 Items 集合中的一个项
	添加	btn_LiAdd	在 listView4 的 List 视图下添加 Items 集合中的一个项

	添加	btn_TAdd	在 listView5 的 Title 视图下添加 Items 集合中的一个项



4、控件解析

常用属性及介绍：

FullRowSelect 属性：设置是否行选择模式。(默认为 false) 提示：只有在 Details 视图该属性才有意义。

GridLines 属性:设置行和列之间是否显示网格线。(默认为 false)提示:只有在 Details 视图该属性才有意义。

AllowColumnReorder 属性:设置是否可拖动列标头来对改变列的顺序。(默认为 false) 提示：只有在 Details 视图该属性才有意义。

View 属性：获取或设置项在控件中的显示方式，包括 Details、LargeIcon、List、SmallIcon、Tile（默认为 LargeIcon）

MultiSelect 属性：设置是否可以选择多个项。(默认为 false)

HeaderStyle 属性：获取或设置列标头样式。

Clickable：列标头的作用类似于按钮，单击时可以执行操作　　（例如排序）。

NonClickable：列标头不响应鼠标单击。

None：不显示列标头。

LabelEdit 属性：设置用户是否可以编辑控件中项的标签，对于 Detail 视图，只能编辑行第一列的内容。(默认为 false)

CheckBoxes 属性：设置控件中各项的旁边是否显示复选框。(默认为 false)

LargeImageList 属性：大图标集。提示：只在 LargeIcon 视图使用。

SmallImageList 属性：小图标集。提示：只有在 SmallIcon 视图使用。

StateImageList 属性：图像蒙板。这些图像蒙板可用作 LargeImageList 和 SmallImageList 图像的覆盖图，这些图像可用于指示项的应用程序定义的状态。

SelectedItems 属性：获取在控件中选定的项。

CheckedItems 属性：获取控件中当前复选框选中的项。

Soritng 属性：对列表视图的项进行排序。(默认为 None)

Ascending：项按递增顺序排序。

Descending：项按递减顺序排序。

None：项未排序。

Scrollable 属性:设置当没有足够空间来显示所有项时是否显示滚动条。(默认为 true)

HoverSelection 属性:设置当鼠标指针悬停于项上时是否自动选择项。(默认为 false)

HotTracking 属性：设置当鼠标指针经过项文本时，其外观是否变为超链接的形式。(默认为 false)

HideSelection 属性：设置选定项在控件没焦点时是否仍突出显示。(默认为 false)

ShowGroups 属性：设置是否以分组方式显示项。(默认为 false)；

Groups 属性：设置分组的对象集合。

TopItem 属性：获取或设置控件中的第一个可见项，可用于定位。(效果类似于 EnsureVisible 方法)

常用事件及介绍：

AfterLabelEdit 事件：当用户编辑完项的标签时发生，需要 LabelEdit 属性为 true。

BeforeLabelEdit 事件：当用户开始编辑项的标签时发生。

ColumnClick 事件：当用户在列表视图控件中单击列标头时发生。

常用方法及介绍

BeginUpdate：避免在调用 EndUpdate 方法之前描述控件。当插入大量数据时，可以有效地避免控件闪烁，并能大大提高速度。

EndUpdate：在 BeginUpdate 方法挂起描述后，继续描述列表视图控件。(结束更新)

EnsureVisible：列表视图滚动定位到指定索引项的选项行。(效果类似于 TopItem 属性)

FindItemWithText：查找以给定文本值开头的第一个 ListViewItem。

FindNearestItem：按照指定的搜索方向，从给定点开始查找下一个项。提示：只有在 LargeIcon 或 SmallIcon 视图才能使用该方法。

添加窗体启动事件 FrmMain_Load

添加 Details 视图中的添加按钮事件 btn_DAdd_Click

添加 LargeIcon 视图中的添加按钮事件 btn_LAdd_Click

添加 SmallIcon 视图中的添加按钮事件 btn_SAdd_Click

添加 List 视图中的添加按钮事件 btn_LiAdd_Click

添加 Title 视图中的添加按钮事件 btn_TAdd_Click

添加列表激活事件 listView1_ItemActivate

5. 实验代码解析

（1）在 FrmMain 窗体中添加一个 TabControl 控件并将其 Dock 属性值设置为 Fill；然后在 TabPages 属性中添加五个 tabPage，其 Text 属性分别为：Detail 视图、LargeIcon 视图、SmallIcon 视图、List 视图、Title 视图。

（2）在 FrmMain 窗体中添加一个 ImageList 控件，其 Images 属性中添加了五张图片。

（3）在 TabControl 的 Details 视图下添加一个 ListView 控件并在其 Columns 属性中添加 3 个列其 Text 属性分别为列 1、列 2、列 3；SmallImageList 属性中选择 ImageList1，

View 属性选择 Details。

（4）在 TabControl 的 LargeIcon 视图下添加一个 ListView 控件，其 LargeImageList 属性设置为 imageList1，View 属性设置为 LargeIcon。

（5)在 TabControl 的 SmallIcon 视图下添加一个 ListView 控件，在其 SmallImagelist 属性中选择 imageList1 属性，View 属性设置为 SmallIcon。

（6）在 TabControl 的 List 视图下添加一个 ListView 控件，在其 SmallImagelist 属性中选择 imageList1 属性，View 属性设置为 List。

（7）在 TablControl 的 Title 视图下添加一个 ListView 控件，在其 LargeImagelist 属性中选择 imageList1 属性，View 属性设置为 Title。

（8）在 TabControl 的各个视图下添加一个 Button 控件。

（9）使用 ListView 的 BeginUpdate 方法可使 UI 暂时挂起，进行数据更新；使用 EndUpdate 方法则结束数据处理。

（10）实例化一个 ListViewItem 对象，设置 ImageList 属性来设置图像的索引，在 Detail 视图下其 Text 属性的值即为列 1 的值，使用 SubItems 的 Add 方法来为其它列添加值，其它视图下没有列直接使用 Text 属性的值添加；最后使用 ListView 的 Items 的 Add 方法将实例化的 ListItem 对象添加进去。

（11）在 FrmMian 的窗体 Load 事件中将第九和第十的功能项添加进其代码块即可，在各个视图下添加项是和 Load 事件中的代码一样的。

定义变量:

```
Random rd = newRandom();//用于生成随机数
int i = 0;
```

窗体启动事件 FrmMain_Load，具体代码如下：

```
privatevoid FrmMain_Load(object sender, EventArgs e)
        {
#region Details视图中ListView的设置

this.listView1.BeginUpdate();//数据更新，UI暂时挂起，直到EndUpdate绘制控件，可以有效避免闪烁并大大提高加载速度
for (i = 0; i < 5; i++)//添加5行数据
            {
ListViewItem lvi = newListViewItem();
                lvi.ImageIndex = i;//通过与ImageList绑定，显示ImageList中第i项图标
                lvi.Text = "subitem" + (i+1);
                lvi.SubItems.Add("第2列，第"+(i+1)+"行");
                lvi.SubItems.Add("第3列，第"+(i+1)+"行");
this.listView1.Items.Add(lvi);//添加到listView1的Items集合中去
            }
this.listView1.EndUpdate();//结束数据处理，UI界面一次性绘制。
#endregion

#region LargeIcon视图中ListView的设置
this.listView2.BeginUpdate();
for ( i = 0; i < 5; i++)
```

```
                {
ListViewItem lvi = newListViewItem();
                    lvi.ImageIndex = i;
                    lvi.Text = "itme"+(i+1);
this.listView2.Items.Add(lvi);
                }
this.listView2.EndUpdate();
#endregion

#region SmallIcon视图中ListView的设置
this.listView3.BeginUpdate();
for (i = 0; i < 5; i++)
                {
ListViewItem lvi = newListViewItem();
                    lvi.ImageIndex = i;
                    lvi.Text = "itme" + (i+1);
this.listView3.Items.Add(lvi);
                }
this.listView3.EndUpdate();
#endregion

#region List视图中的ListView的设置
this.listView4.BeginUpdate();
for (i = 0; i < 5; i++)
                {
ListViewItem lvi = newListViewItem();
                    lvi.ImageIndex = i;
                    lvi.Text = "itme" + (i + 1);
this.listView4.Items.Add(lvi);
                }
this.listView4.EndUpdate();
#endregion

#region Title视图中的Title的设置
this.listView5.BeginUpdate();
for (i = 0; i < 5; i++)
                {
ListViewItem lvi = newListViewItem();
                    lvi.ImageIndex = i;
                    lvi.Text = "itme" + (i + 1);
this.listView5.Items.Add(lvi);
                }
this.listView5.EndUpdate();
#endregion
```

```
        }
```

Details 视图中的添加按钮事件 btn_DAdd_Click，具体代码如下：

```
        privatevoid btn_DAdd_Click(object sender, EventArgs e)
          {
if (txtbox_DColumn1.Text.Trim() != "")
               {
if (txtbox_DColumn2.Text.Trim() != "")
                    {
if (txtbox_DColumn3.Text.Trim() != "")
                         {
ListViewItem lvi = newListViewItem();
                              lvi.ImageIndex = rd.Next(0,5);//随机选择imagelist5张图片
中的一种
                              lvi.Text = txtbox_DColumn1.Text.Trim();//第一列
                              lvi.SubItems.Add(txtbox_DColumn2.Text.Trim());//第二列
                              lvi.SubItems.Add(txtbox_DColumn3.Text.Trim());//第三列
this.listView1.Items.Add(lvi);
                         }
else
MessageBox.Show("列3不能为空！");
                    }
else
MessageBox.Show("列2不能为空！");
               }
else
MessageBox.Show("列1不能为空！");
        }
```

LargeIcon 视图中的添加按钮事件 btn_LAdd_Click，具体代码如下：

```
        privatevoid btn_LAdd_Click(object sender, EventArgs e)
        {
ListViewItem lvi = newListViewItem();
               lvi.ImageIndex = rd.Next(0,5);
               lvi.Text = "item" ;
this.listView2.Items.Add(lvi);
        }
```

SmallIcon 视图中的添加按钮事件 btn_SAdd_Click，具体代码如下：

```
        privatevoid btn_SAdd_Click(object sender, EventArgs e)
        {
ListViewItem lvi = newListViewItem();
               lvi.ImageIndex = rd.Next(0, 5);
               lvi.Text = "item" ;
this.listView3.Items.Add(lvi);
        }
```

List 视图中的添加按钮事件 btn_LiAdd_Click，具体代码如下：

```
        privatevoid btn_LiAdd_Click(object sender, EventArgs e)
        {
ListViewItem lvi = newListViewItem();
                lvi.ImageIndex = rd.Next(0, 5);
                lvi.Text = "item";
this.listView4.Items.Add(lvi);
        }
```

Title 视图中的添加按钮事件 btn_TAdd_Click，具体代码如下：

```
        privatevoid btn_TAdd_Click(object sender, EventArgs e)
        {
ListViewItem lvi = newListViewItem();
                lvi.ImageIndex = rd.Next(0, 5);
                lvi.Text = "item";
this.listView5.Items.Add(lvi);
        }
```

列表激活事件 listView1_ItemActivate，具体代码如下：

```
        privatevoid listView1_ItemActivate(object sender, EventArgs e)
{
MessageBox.Show("第一列：" + this.listView1.SelectedItems[0].Text + "\r\n第二列：" +
this.listView1.SelectedItems[0].SubItems[1].Text + "\r\n第三列：" +
this.listView1.SelectedItems[0].SubItems[2].Text);
        }
```

A0107 指导文档　MenuStrip 控件学习

1. 实验目的

该实验的目的是让学生知道如何为 MenuStrip 控件添加 Items 集合并展示

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

MenuStrip 类
为窗体提供菜单系统。
所属命名空间:System.Windows.Forms

MenuStrip 控件

在建立菜单系统时，要给 MenuStrip 添加 ToolStripMenu 对象。这可以在代码中完成，也可以在 Visual Studio 的设计器中进行。把一个 MenuStrip 控件拖放到设计器的一个窗体中，MenuStrip 就允许直接在菜单项上输入菜单文本[1] 。

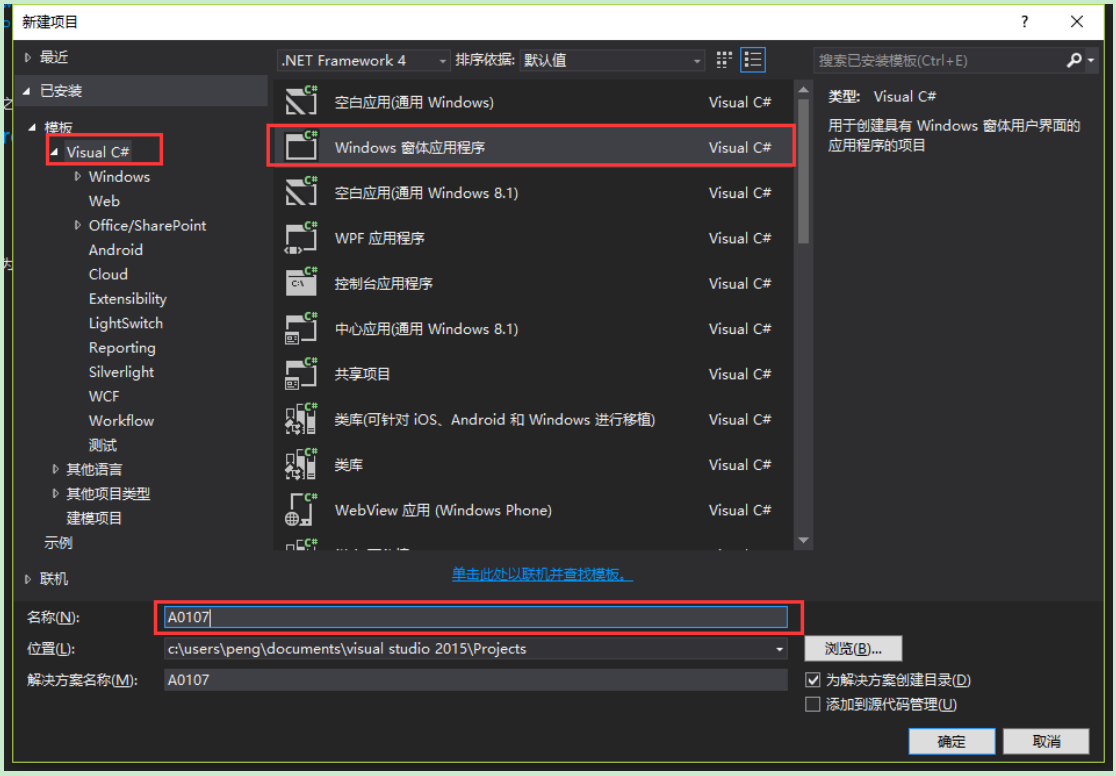
MenuStrip 控件只有两个额外的属性。GripStyle 使用 ToolStripGripStyle 枚举把栅格设置为可见或隐藏。

MdiWindowListItem 属性提取或返回 ToolStripMenuItem。这个 ToolStripMenuItem 是在 MDI 应用程序中显示所有已打开窗口的菜单。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

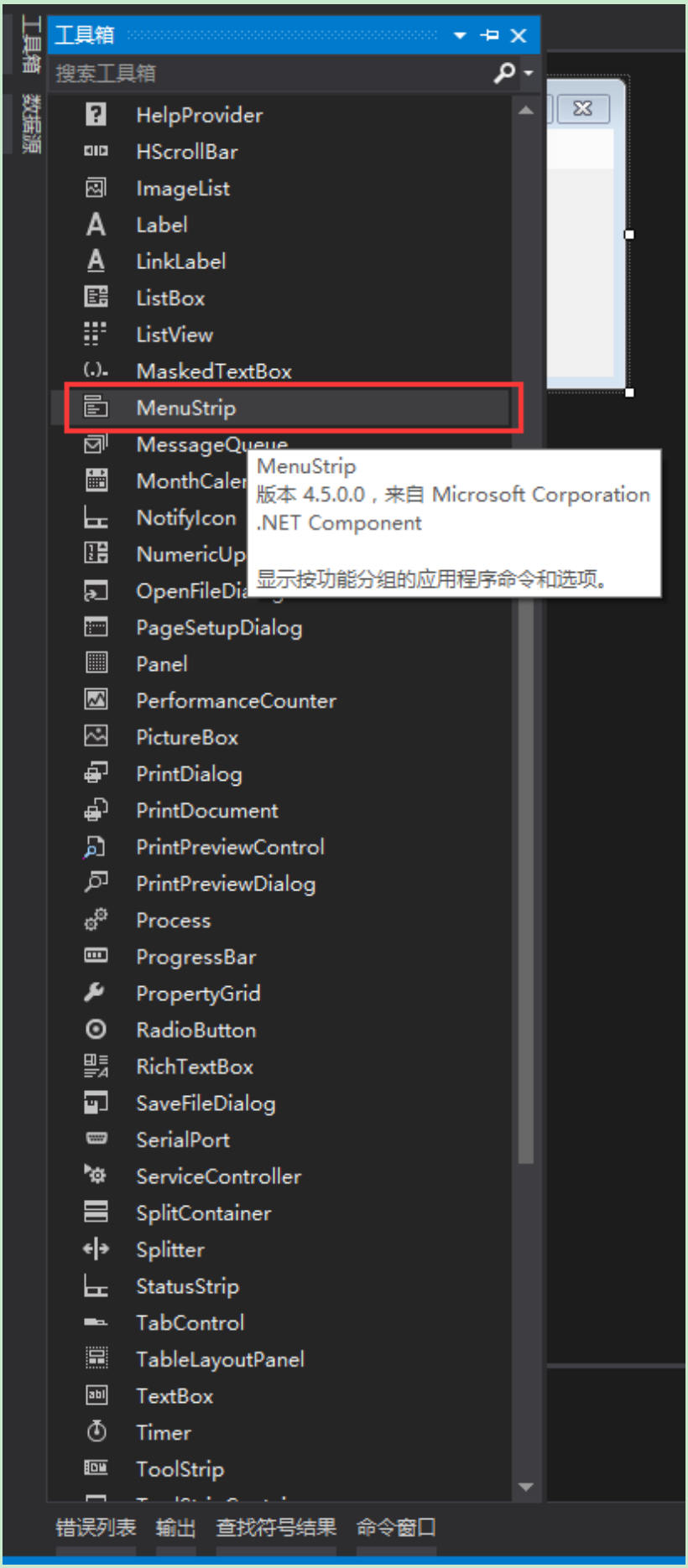
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



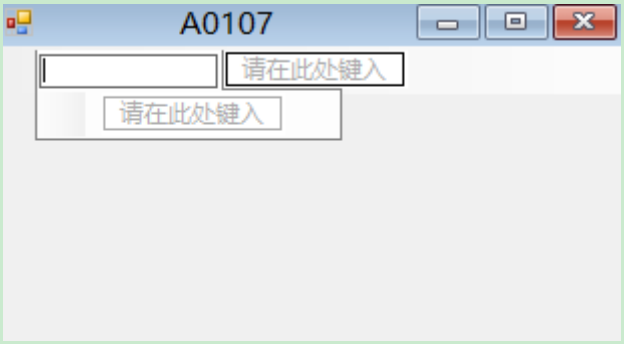
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 MenuStrip 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalen
NotifyIcon
NumericUp
OpenFileDia
PageSetupDialog
Panel
PerformanceCounter
PictureBox
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
PropertyGrid
RadioButton
RichTextBox
SaveFileDialog
SerialPort
ServiceController
SplitContainer
Splitter
StatusStrip
TabControl
TableLayoutPanel
TextBox
Timer
ToolStrip
MenuStrip
版本 4.5.0.0，来自 Microsoft Corporation
.NET Component
显示按功能分组的应用程序命令和选项。
错误列表
输出
查找符号结果
命令窗口

控件名称	控件 Text 属性	控件 Name 属性	功能
Form1 窗体	B00601	frmMain	
MenuStrip 控件	MenuStrip1	MenuStrip	应用程序菜单结构的容器



4、控件解析：

常用属性及介绍：

Name ： 表示一个控件或者窗体的名称。

BackColor： 控件和窗体的背景颜色。

BackgroundImage： 窗体的背景图片。

ContextMenuStrip： 鼠标单击右键，所出先的菜单。

Dock： 表示该控件在窗体的那个位置的停靠。

Enabled： 表示该控件是否有用 默认值 Ture 。 选 False 该控件则没有任何作用。

Font： 设置控件里字体的大小、字号、字体、和下划线。 ForeColor： 在控件里输入字体时， 字体的颜色。(默认值为黑色)。

Location： 表示该控件在窗体中的位置。X 坐标数值越大则控件会越往右，Y 坐标数值越大则控件越往下。

Size：该控件的大小 width 表示控件的宽度 heigth 表示控件的高度。Visible：是否隐藏该控件。 选 False 控件是看不见的。

Items： 获取属于 ToolStrip 控件的所有项

GripStyle:获取或设置用于重新定位控件的手柄的可见性

MdiWindowListItem:获取或设置用于重新定位控件的手柄的可见性

Stretch:获取或设置一个值,指示 MenuStrip 控件是否在其容器中从一段拉伸到另一端

ShowItemToolTips:获取或设置一个值,指示是否显示 MenuStrip 控件的工具提示

常用事件及介绍：

Click 事件：当用户用鼠标左键单击按钮控件时，将发生该事件。

Scroll 事件:当用户移动滚动框时发生

添加窗体加载事件 A0107_Load

5. 实验代码解析

窗体加载事件具体代码如下所示：

```
privatevoid A0107_Load(object sender, EventArgs e)
```

```
        {
//其中menuStrip1为MenuStrip控件
this.menuStrip1.Items.Add("视图");//为menuStrip1添加二级菜单项"视图"
this.menuStrip1.Items.Add("重构");//为menuStrip1添加二级菜单项"重构"
this.menuStrip1.Items.Add("项目");//为menuStrip1添加二级菜单项"项目"
this.menuStrip1.Items.Add("生成");//为menuStrip1添加二级菜单项"生成"
        }
```

A0108 指导文档　NumericUpDowm 控件学习

1. 实验目的

该实验的主要目的是让学生知道如何设置 NumericUpDowm 控件的显示格式、最大范围、最小范围以及如何获取 NumericUpDowm 控件的值。

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

NumericUpDowm 类

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

所属命名空间:System.Windows.Forms

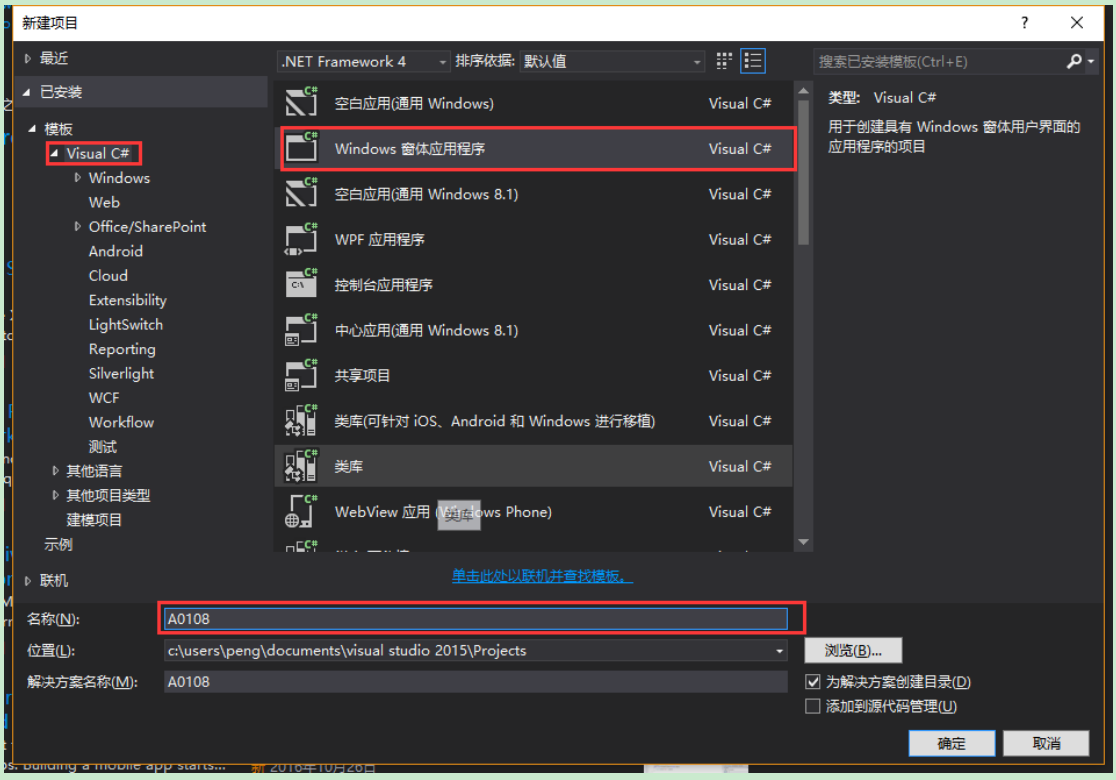
NumericUpDowm 控件

Windows 窗体 NumericUpDown 控件看起来像是一个文本框与一对箭头的组合，用户可以单击箭头来调整值。 该控件显示并设置选择列表中的单个数值。 用户可以通过单击向上和向下按钮、按向上键和向下键或键入一个数字来增大和减小数字。 单击向上键时，值沿最大值方向增加；单击向下键时，位置沿最小值方向移动。 说明此类控件很有用的一个示例是音乐播放器上的音量控件。 某些 Windows 控制面板应用程序中使用了数值 up-down 控件。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

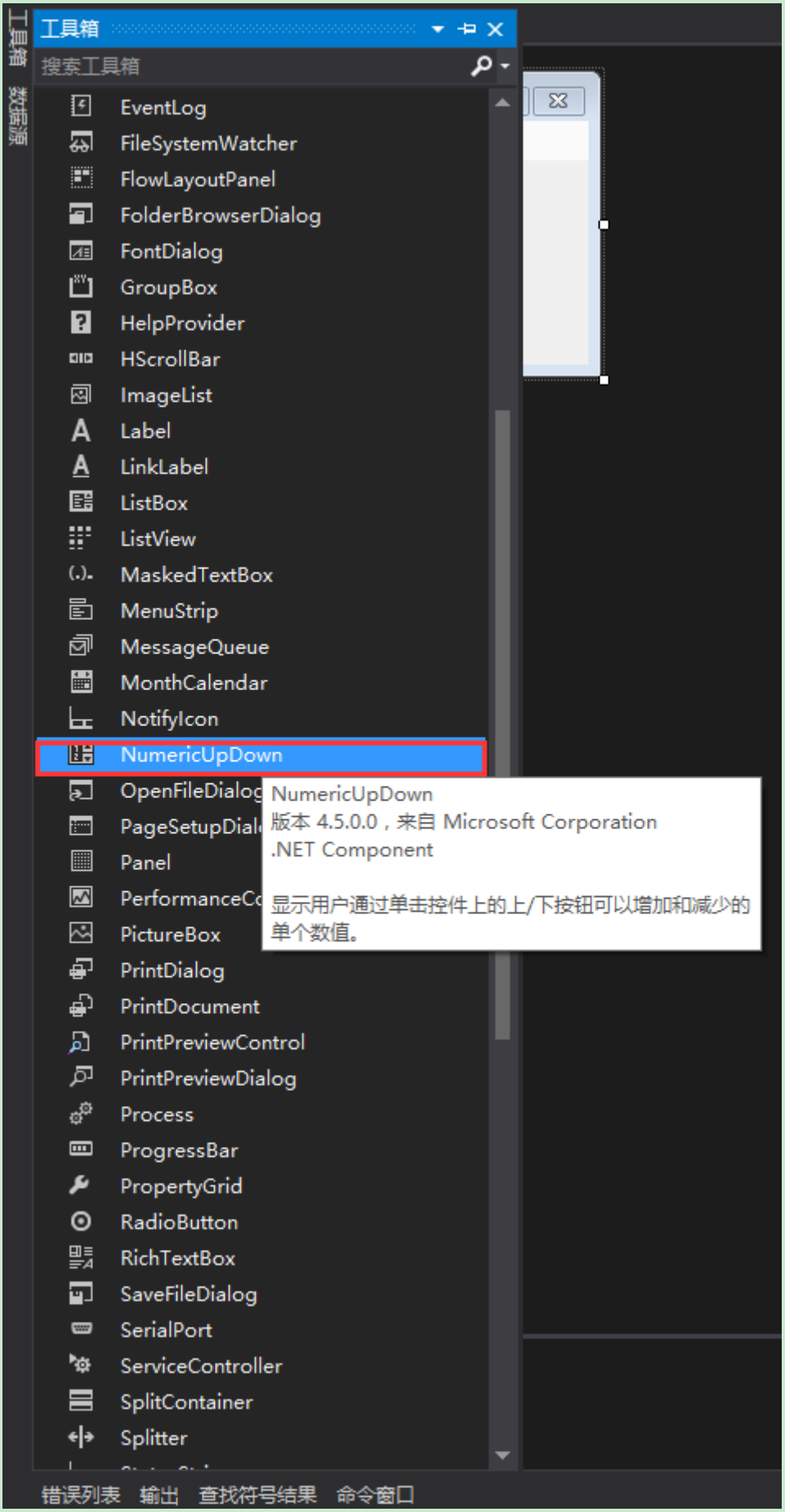
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



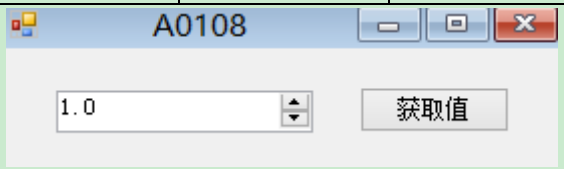
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 NumericUpDowm 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowserDialog
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
PageSetupDial
Panel
PerformanceC
PictureBox
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
PropertyGrid
RadioButton
RichTextBox
SaveFileDialog
SerialPort
ServiceController
SplitContainer
Splitter
NumericUpDown
版本 4.5.0.0，来自 Microsoft Corporation
.NET Component
显示用户通过单击控件上的上/下按钮可以增加和减少的单个数值。
错误列表
输出
查找符号结果
命令窗口

控件名称	控件 Text 属性	控件 Name 属性	功能
Form 窗体	A0108	frmForm	
NumerricUpDowm 控件	numericUpDowm1	NumerricUpDowm1	
Button 按钮	获取值	btnGetValue	获取 numerricUpDowm1 中的值



4、控件分析

常用属性及介绍：

Name ： 表示一个控件或者窗体的名称。

BackColor：控件和窗体的背景颜色。

BackgroundImage：窗体的背景图片。

Font：设置控件里字体的大小、字号、字体、和下划线。 ForeColor：在控件里输入字体时， 字体的颜色。(默认值为黑色)。

DecimalPlaces:设置 NumericUpDowm 控件要显示的小数点数。

Increment:设置 NumericUpDowm 的值每单击一下按钮时增加或减少的数量。

Maxinum：设置 NumericUpDowm 控件的最大值。

Minimum：设置 NumericUpDowm 控件的最小值。

ThousandsSeparator:设置 NumericUpDowm 控件是否在每三位十进制数之间插入千分位分隔符。

常用属性及介绍：

Click 事件：当用户用鼠标左键单击 NumericUpDowm 控件时，将发生该事件。

ValueChange 事件:当控件中的值更改时发生。

添加获取值按钮 btnGetValue_Click

5. 实验代码分析

（1）设置 NumericUpDowm 控件的显示格式、最大范围和最小范围

A)NumericUpDowm 的 DecimalPlaces 属性用来设置 NumericUpDowm 控件要显示的小数点数。

B)NumericUpDowm 的 Increment 属性用来设置 NumericUpDowm 的值每单击一下按钮时增加或减少的数量

C)NumericUpDowm 的 Maxinum 属性用来设置 NumericUpDowm 控件的最大值

D)NumericUpDowm 的 Minimum 属性用来设置 NumericUpDowm 控件的最小值

E)NumericUpDowm 的 ThousandsSeparator 属性用来设置 NumericUpDowm 控件是否在每三位十进制数之间插入千分位分隔符

（2）获取 NumericUpDowm 控件的值

利用按钮“获取值”的 Click 事件获取 NumericUpDowm 的值并通过 MessgBox 提示信息显示该值

获取值按钮单击事件代码如下所示

```
        privatevoid btnGetValue_Click(object sender, EventArgs e)
        {
MessageBox.Show(numericUpDown1.Value.ToString());//通过消息框提示的信息显示
numericUpDown1的当前值
        }
```

A0109 指导文档　PictureBox 控件学习

1. 实验目的

该实验的主要目的是让学生知道如何给 PictureBox 控件添加本地图片

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

PictureBox 类

表示用于显示图像的 Windows 图片框控件。

所属命名空间:System.Windows.Forms

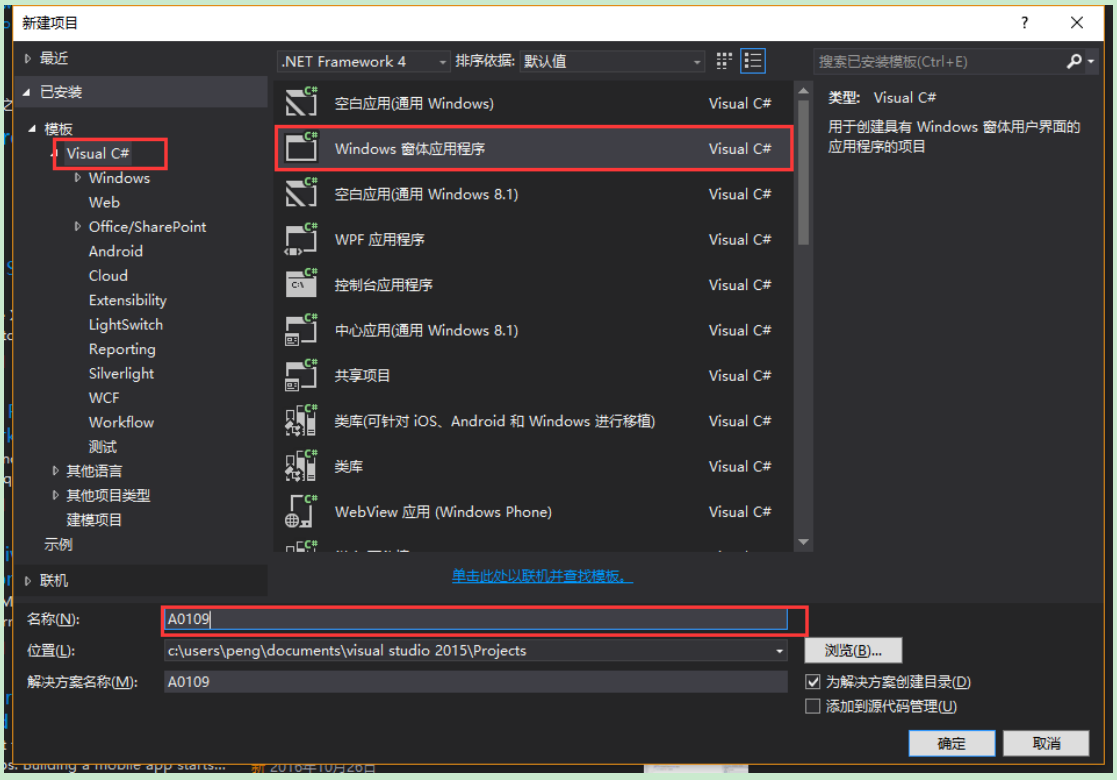
PictureBox 控件

PictureBox 控件可以显示来自位图、图标或者元文件，以及来自增强的元文件、JPEG 或 GIF 文件的图形。如果控件不足以显示整幅图象，则裁剪图象以适应控件的大小。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

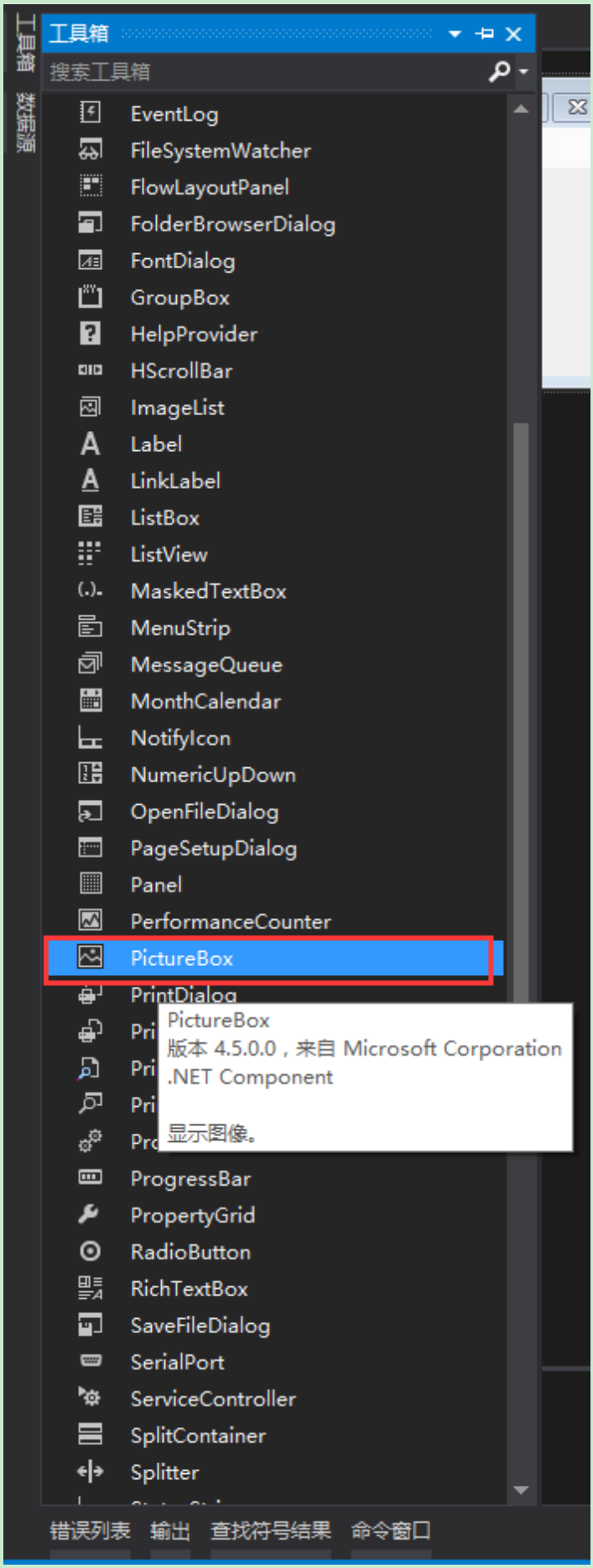
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



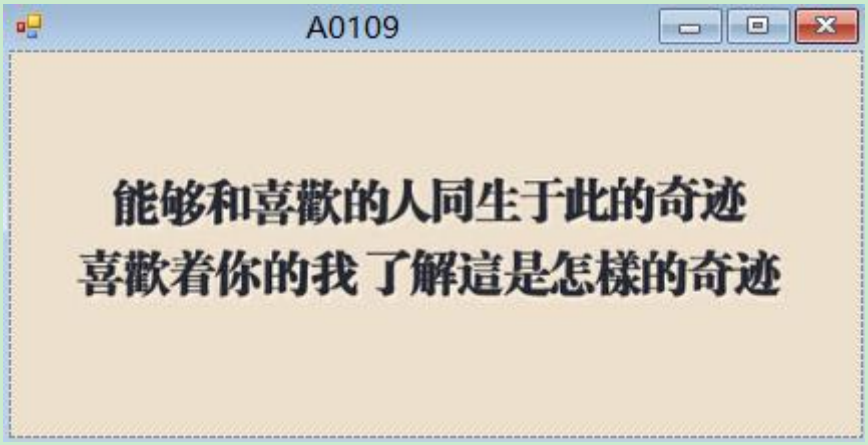
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 PictureBox 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowserDialog
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
PageSetupDialog
Panel
PerformanceCounter
PictureBox
PrintDialog
PictureBox
版本 4.5.0.0，来自 Microsoft Corporation
.NET Component
显示图像。
ProgressBar
PropertyGrid
RadioButton
RichTextBox
SaveFileDialog
SerialPort
ServiceController
SplitContainer
Splitter
错误列表
输出
查找符号结果
命令窗口

控件名称	控件 Text 属性	控件 Name 属性	功能
Form 窗体	A0109	frmMain	
PictureBox 控件		picLocalPicture	显示图片



4、控件分析：

常用属性及介绍：

Appearance 属性:返回或设置 MDIForm 或 Form 对象上的控件在设计时的绘图风格,在运行时是只读的。

BackColor:返回或设置对象的背景颜色。

ForeColor：返回或设置在对象里显示图片和文本的前景颜色。

DataChanged：返回或设置一个值，它指出被绑定的控件中的数据已被某进程改变，这个进程不是从当前记录中检索数据的进程。该在设计时不可用。

DataField：返回或设置数据使用者将被绑定到的字段名。

DataFormat：设置或返回 StdDataFormat 对象，一个绑定对象将附加到它。在设计时或运行时都可读写

DataMember：从数据供应程序提供的几个数据成员中返回或设置一个特定的数据成员。

DragIcon：返回或设置图标，它将在拖放操作中作为指针显示

DragMode：返回或设置一个值，确定在拖放操作中所使用的是手动还是自动拖放方式。

Enabled：返回或设置一个值，该值用来确定一个窗体或控件是否能够对用户产生的事件做出反应。

常用事件及介绍：

Click 事件：当用户用鼠标左键单击 PictureBox 控件时，将发生该事件。

DoubleClick 事件：当用户用鼠标左键双击 PictureBox 控件时，将发生该事件。

5. 实验代码解析

使 PictureBox 上显示指定的本地图片

往窗体中拖入一个 PictureBox 控件,通过鼠标拖动调整至适当大小或设置其 Dock 属性设置为 Fill(既填充整个窗体),设置其 Name 属性为 picLocalPicture,设置其 Image 属性:

单击 Image 属性右边的“...”，选择本地资源→导入→按路径选择需要指定的图片并选中(在此之前需要把该图片移动到本项目的工程文件的“图片”目录下,这样项目在其他任何电脑上都能使用)→选中图片单击确定后就能在 PictureBox 控件上显示本地图片了。

本实验无代码。

A0110 指导文档　RadioButton 控件学习

1. 实验目的

该实验主要是为了让学生熟悉使用 RadioButton 控件。

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

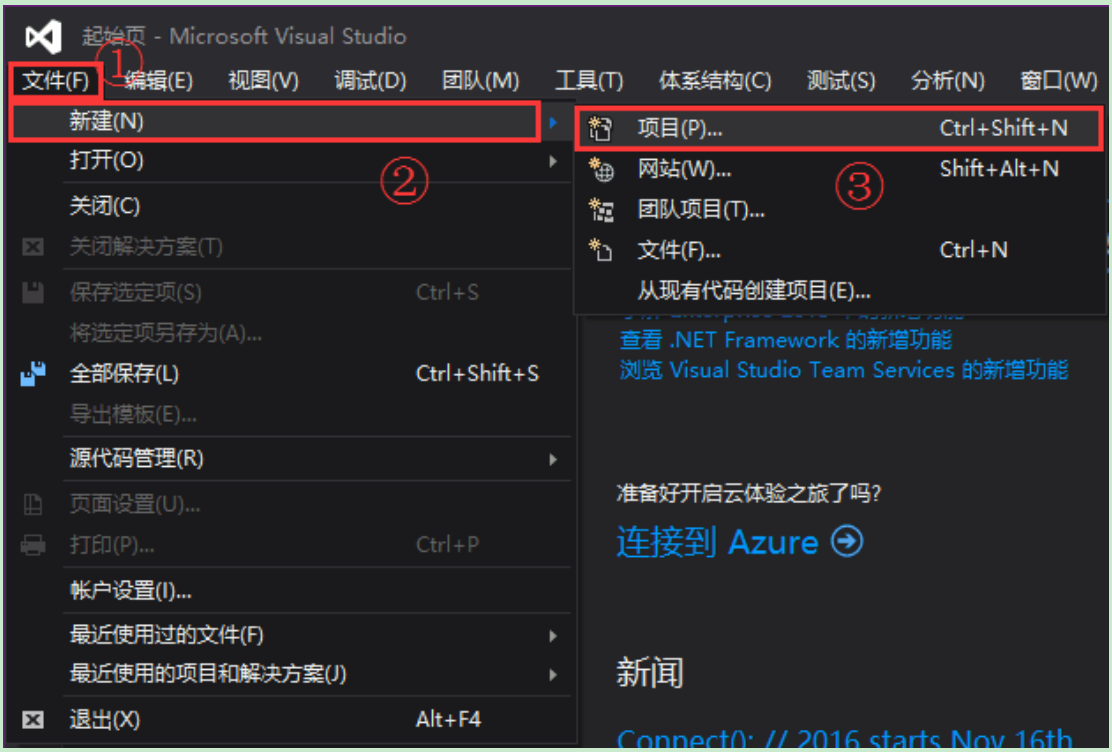
RadioButton 类

表示一个按钮，可以选择，但不是会清除，由用户。 IsChecked 属性 RadioButton 可以通过单击它，设置，但它可以只以编程方式清除。

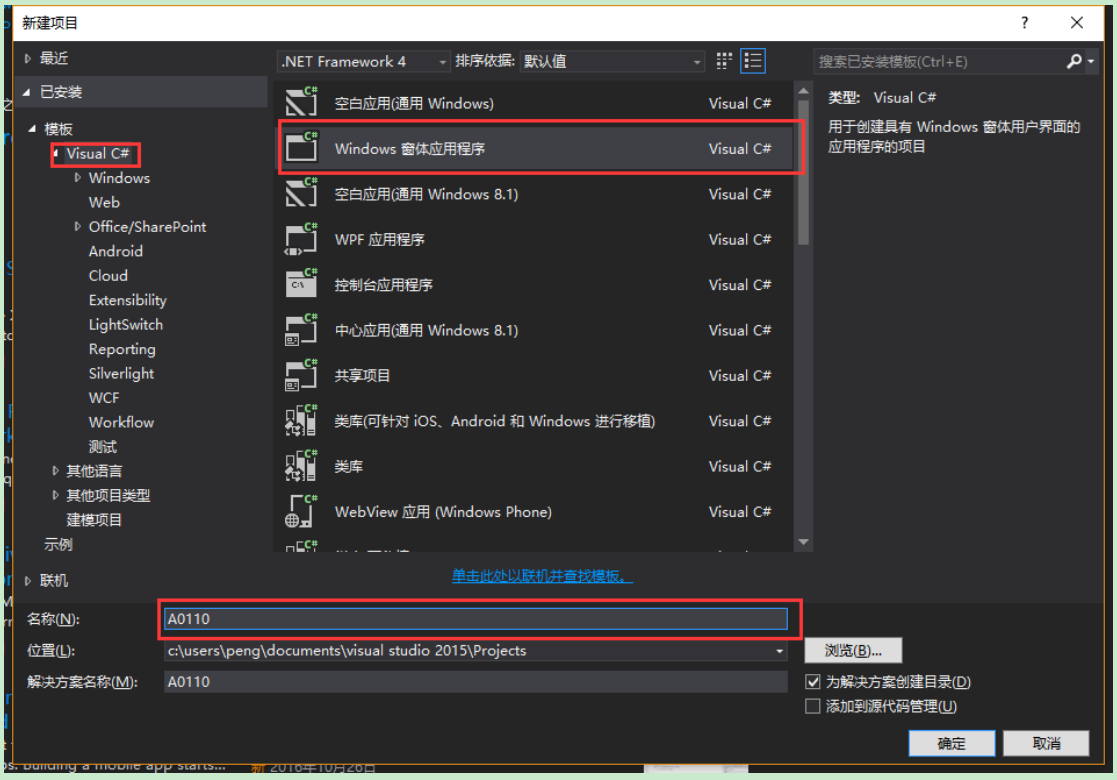
所属命名空间:System.Windows.Forms

4. 实验设计

1、启动 visualstudio，文件→新建→项目。



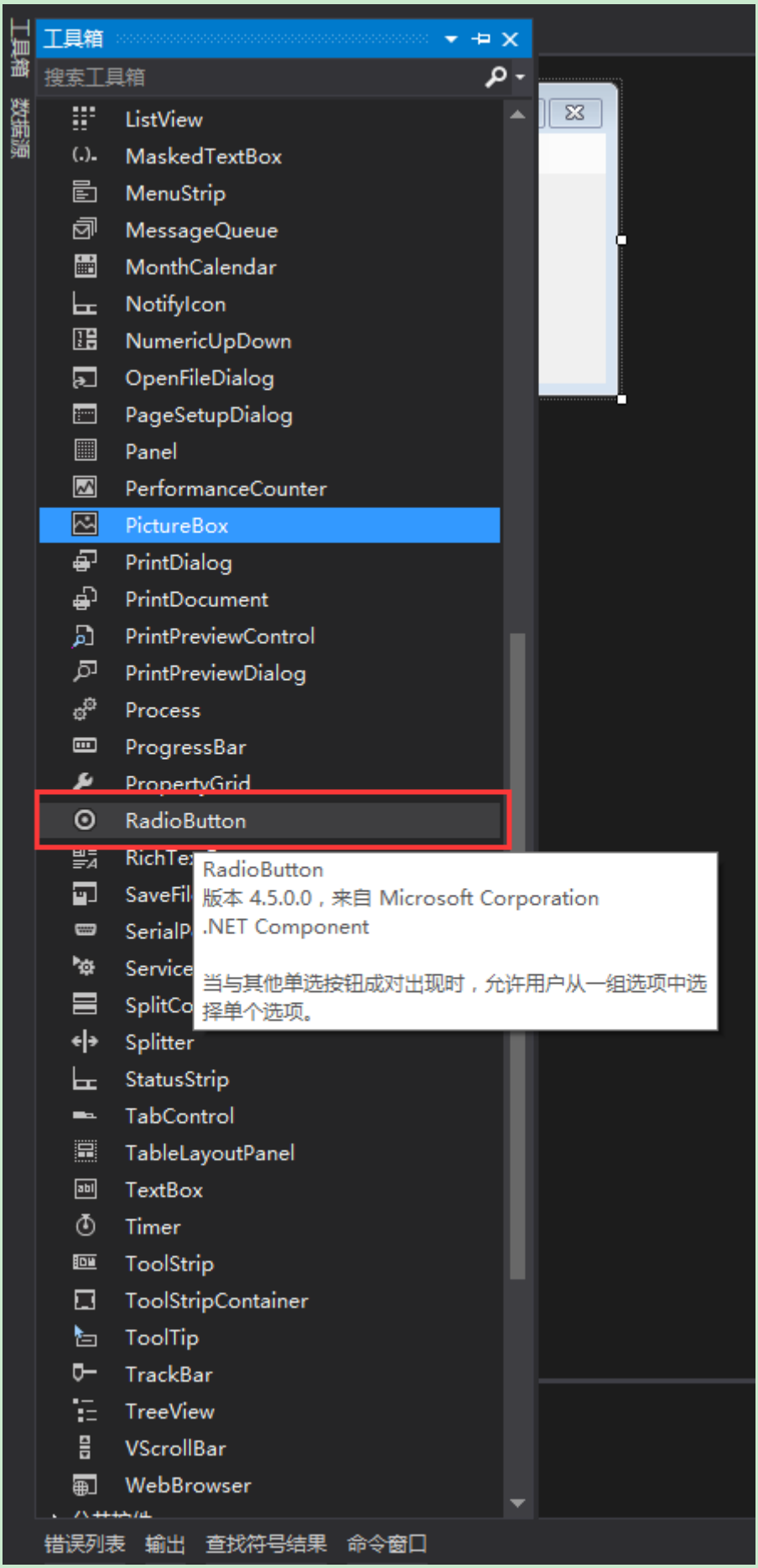
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



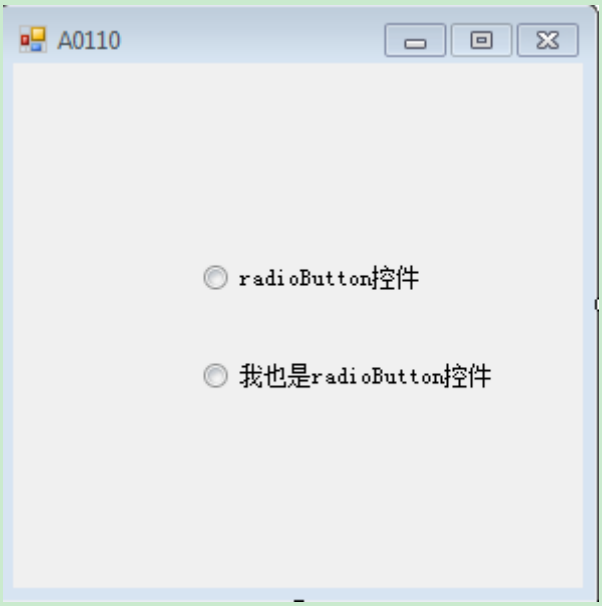
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 RadioButton 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
PageSetupDialog
Panel
PerformanceCounter
PictureBox
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
RadioButton
RadioButton
版本 4.5.0.0，来自 Microsoft Corporation
.NET Component
当与其他单选按钮成对出现时，允许用户从一组选项中选择单个选项。
Splitter
StatusStrip
TabControl
TableLayoutPanel
TextBox
Timer
ToolStrip
ToolStripContainer
ToolTip
TrackBar
TreeView
VScrollBar
WebBrowser
错误列表
输出
查找符号结果
命令窗口

控件名称	Text 属性	Name 属性	功能
Form 窗体	A0110	FrmMain	
RadioButton 控	radioButton1	radioButton 控件	
	radioButton2	我也是 radioButton 控件	



4、控件分析：

常用属性及介绍：

Checked 属性：用来设置或返回单选按钮是否被选中，选中时值为 true，没有选中时值为 false。

AutoCheck 属性：如果 AutoCheck 属性被设置为 true（默认），那么当选择该单选按钮时，将自动清除该组中所有其他单选按钮。对一般用户来说，不需改变该属性，采用默认值（true）即可。

Appearance 属性：用来获取或设置单选按钮控件的外观。当其取值为 Appearance.Button 时，将使单选按钮的外观像命令按钮一样：当选定它时，它看似已被按下。当取值为 Appearance.Normal 时，就是默认的单选按钮的外观。

Text 属性：用来设置或返回单选按钮控件内显示的文本，该属性也可以包含访问键，即前面带有“&”符号的字母，这样用户就可以通过同时按 Alt 键和访问键来选中控件。

常用事件及介绍：

Click 事件：当单击单选按钮时，将把单选按钮的 Checked 属性值设置为 true，同时发生 Click 事件。

CheckedChanged 事件：当 Checked 属性值更改时，将触发 CheckedChanged 事件。

添加单选按钮单击事件 radioButton1_Click

添加单选按钮单击事件 radioButton2_Click，

5. 实验代码解析

在 RadioButton 控件的 Click 事件代码块中返回其 Text 属性的值。

单选按钮单击事件代码如下所示：

```
        privatevoid radioButton1_Click(object sender, EventArgs e)
         {
MessageBox.Show("该控件的Text值为：" + radioButton1.Text);
         }
```

单选按钮单击事件代码如下所示：

```
        privatevoid radioButton2_Click(object sender, EventArgs e)
         {
MessageBox.Show("该控件的Text值为：" + radioButton2.Text);
         }
```

A0111 指导文档　SplitContainer 控件学习

1. 实验目的

该试验主要是为了让学上熟悉使用 SplitContainer 控件。

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

SplitContainer 类

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。

所属命名空间:System.Windows.Forms

SplitContainer 控件

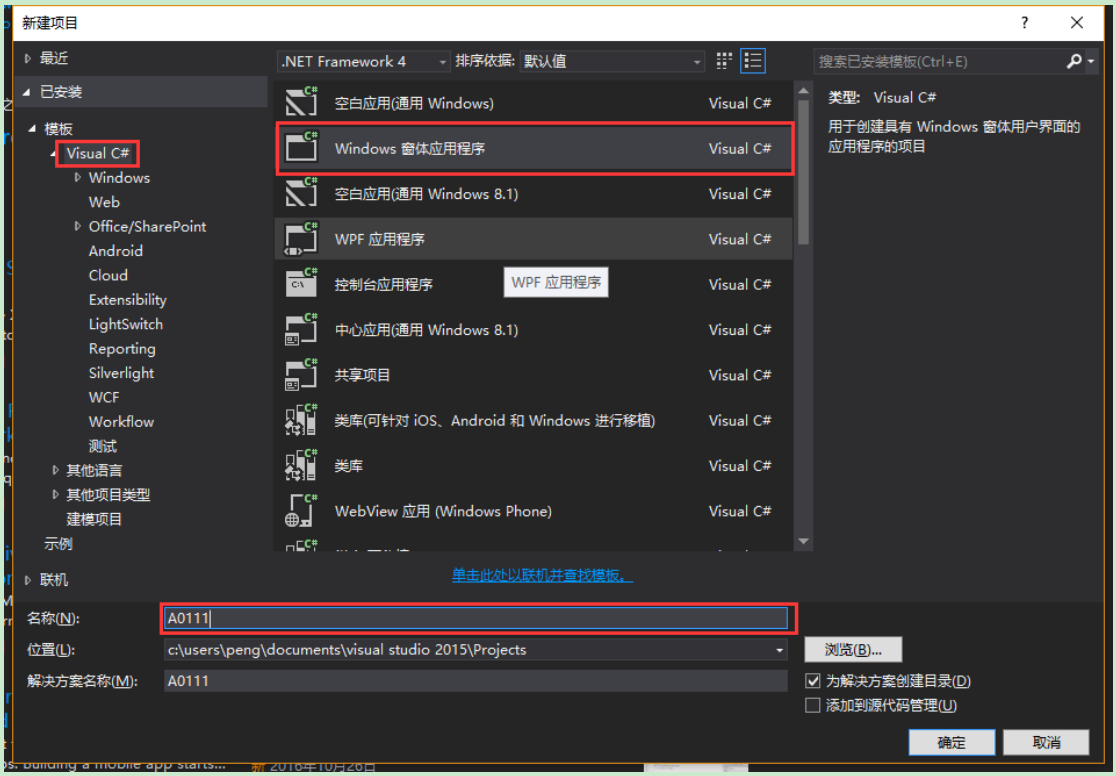
可以将 Windows 窗体 SplitContainer 控件看作是一个复合体，它是由一个可移动的拆分条分隔的两个面板。当鼠标指针悬停在该拆分条上时，指针将相应地改灰度校正状以显示该拆分条是可移动的。

使用 SplitContainer 控件可以创建复合的用户界面（通常，在一个面板中的选择决定了在另一个面板中显示哪些对象）。这种排列对于显示和浏览信息非常有用。拥有两个面板使您可以聚合不同区域中的信息，并且用户可以轻松地使用拆分条（也称为“拆分器”）调整面板的大小。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。

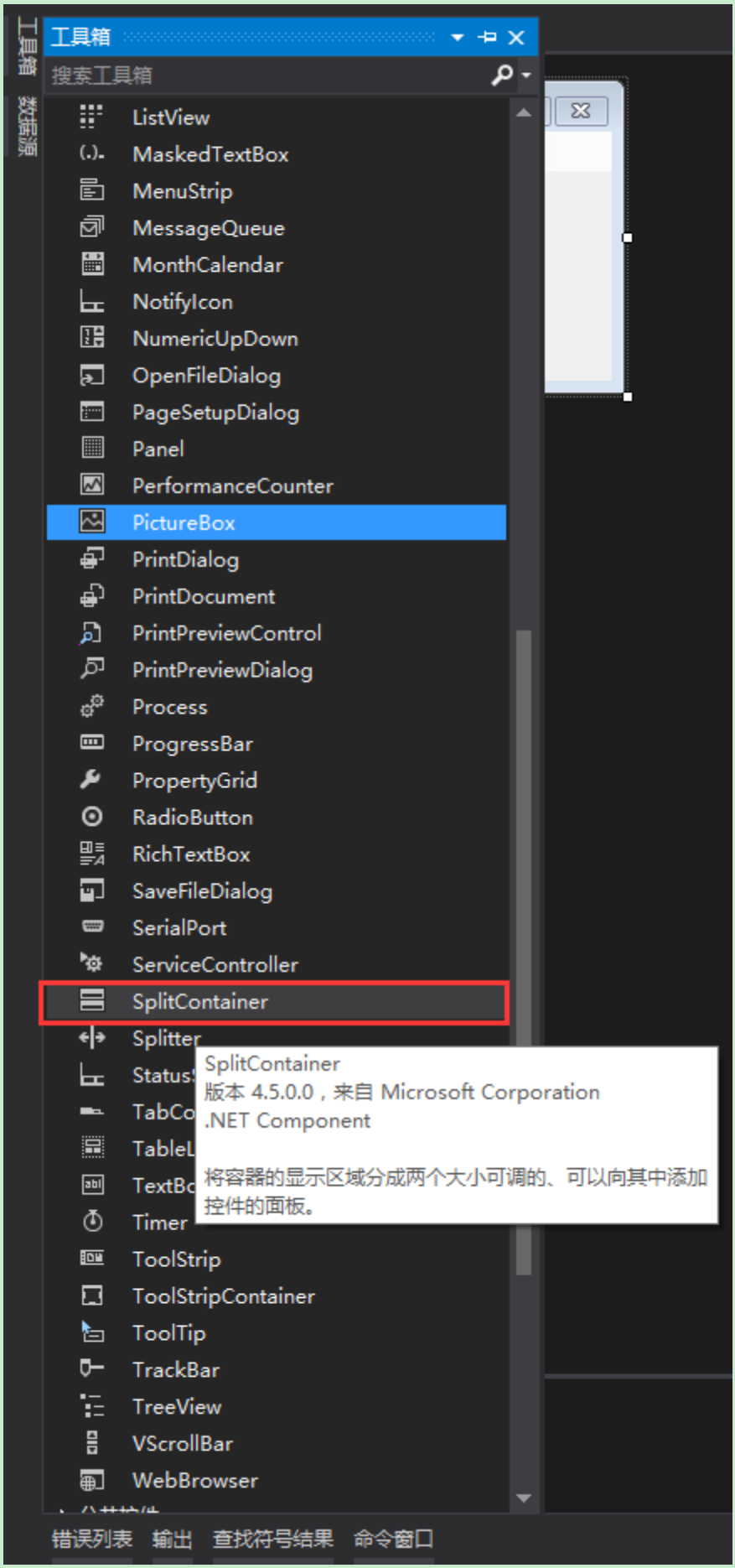


3、界面设计及控件属性

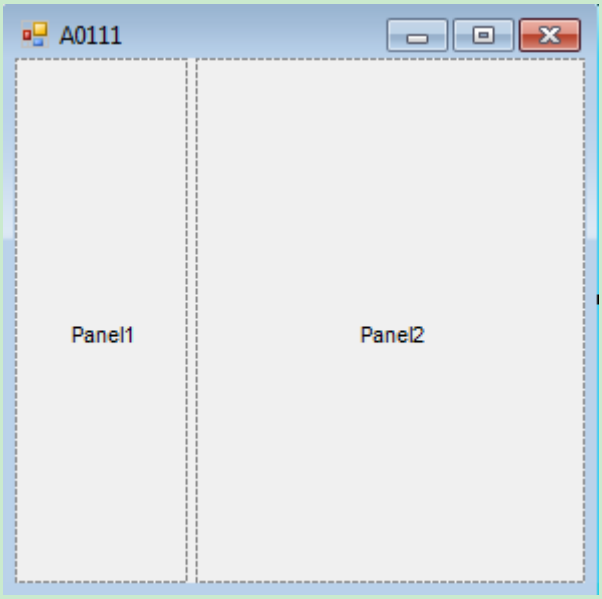
添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜

单，找到工具箱，如下图所示

在工具箱中找到 SplitContainer 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
PageSetupDialog
Panel
PerformanceCounter
PictureBox
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
PropertyGrid
RadioButton
RichTextBox
SaveFileDialog
SerialPort
ServiceController
SplitContainer
Splitter
SplitContainer
版本 4.5.0.0，来自 Microsoft Corporation
.NET Component
将容器的显示区域分成两个大小可调的、可以向其中添加控件的面板。
Timer
ToolStrip
ToolStripContainer
ToolTip
TrackBar
TreeView
VScrollBar
WebBrowser
错误列表
输出
查找符号结果
命令窗口

控件名称	Text 属性	Name 属性	功能
Form 窗体	A0111	FrmMain	
SplitContainer 控件		splitContainer1	将容器的显示区域分成两个大小可调的面板



4、控件解析：

常用属性及介绍：

Dock 属性：定义要绑定到容器的控件边框。

FixedPanel 属性：指示在事件调整大小期间，某个 SplitContainer 面板的大小应保持不变。

IsSplitterFixed 属性：确定拆分器能否固定，False 为不固定，True 为固定（即是否可以移动移动条）。

Orientation 属性：确定拆分器是水平的还是垂直的，Vertical 为垂直的，Horizontal 为水平的。

Panel1 属性：SplitContainer 中的左面板或上面板。

Panel1MinSize 属性：确定拆分器与 Panel1 的左边缘或上边缘的最小距离（以像素为单位）；

Panel1Collapsed 属性：确定 Panel1 是否折叠。

Panel2 属性：SplitContainer 中的右面板或下面板。

SplitterDistance 属性：确定拆分器与左边缘或上边缘的像素距离，即：可移动条与左边缘的初始距离。

SplitterIncrement 属性：调节以像素为单位的可移动条移动的距离

SplitterWidth 属性：拆分器的粗细，即：可移动条的粗细。

常用事件及介绍：

SplitterMoved 事件：在拆分器完成移动时发生。

SplitterMoving 事件：在拆分器正在移动时发生。

添加窗体加载事件 FrmMain_Load

5. 实验代码解析

在 FrmMain 窗体的 Load 事件代码块中添加如下代码块：

splitContainer1.FixedPanel = FixedPanel.Panel1; 这里你也可以直接在 FixedPanel 属性中选择。

窗体加载时间具体代码如下所示：

```
privatevoid FrmMain_Load(object sender, EventArgs e)
{
            splitContainer1.FixedPanel = FixedPanel.Panel1;//固定左边的Panel
}
```

A0112 指导文档　TabControl 控件学习

1. 实验目的

该实验主要是为了让学生熟悉使用 TabControl 控件。

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

TabControl 类

表示包含多个共享相同的空间在屏幕上的项的控件。

所属命名空间:System.Windows.Forms

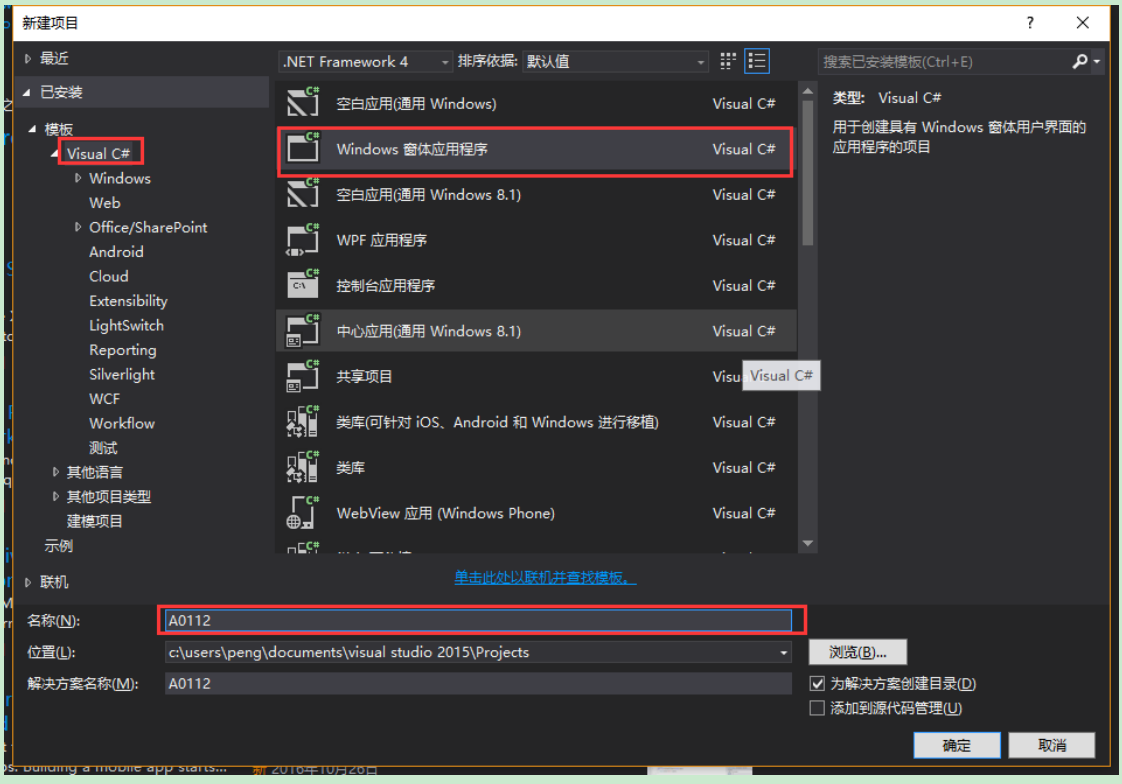
TabControl 控件

在 Windows 应用程序中，选项卡用于将相关的控件集中在一起，放在一个页面中用以显示多种综合信息。选项卡控件通常用于显示多个选项卡，其中每个选项卡均可包含图片和其他控件。选项卡相当于多窗体控件，可以通过设置多页面方式容纳其他控件。由于该控件的集约性，使得在相同操作面积可以执行多页面的信息操作，因此被广泛应用于 Windows 设计开发之中，被很多程序员所喜爱

4. 实验设计

1、 启动 visualstudio，文件→新建→项目。

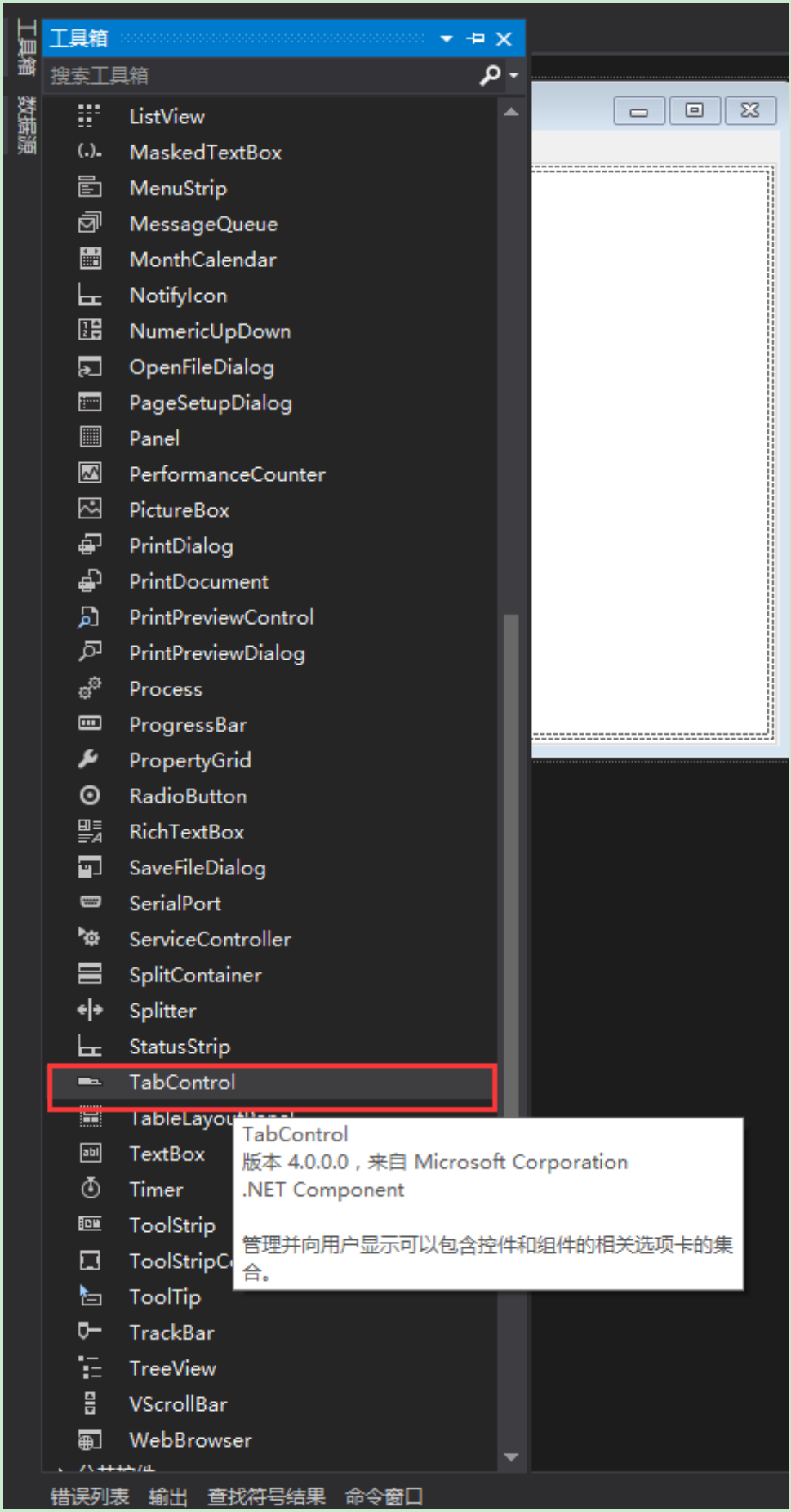
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



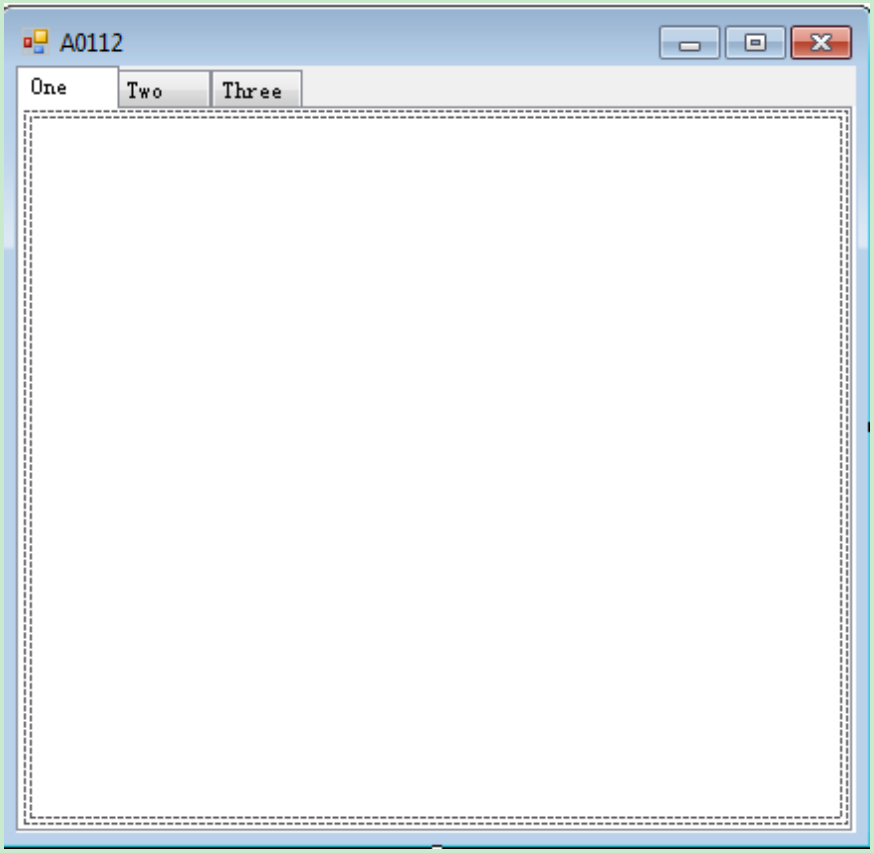
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 TabControl 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
PageSetupDialog
Panel
PerformanceCounter
PictureBox
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
PropertyGrid
RadioButton
RichTextBox
SaveFileDialog
SerialPort
ServiceController
SplitContainer
Splitter
StatusStrip
TabControl
TextBox
Timer
ToolStrip
ToolTip
TrackBar
TreeView
VScrollBar
WebBrowser
TabControl
版本 4.0.0.0，来自 Microsoft Corporation
.NET Component
管理并向用户显示可以包含控件和组件的相关选项卡的集合。
错误列表
输出
查找符号结果
命令窗口

控件名称	Text 属性	Name 属性	功能
Form 窗体	A0112	FrmMain	
TabControl 控件		tabControl1	



4、控件解析

常用属性及介绍：

Dock 属性：定义要绑定到容器的控件边框，为 Fill 时填充整个窗体。

TabPages 属性：TabControl 中的页面集合与设置。

常用事件及介绍：

SelectedIndexChanged 事件：SelectIndex 属性值更改时发生。

添加窗体加载事件 FrmMain_Load

5. 实验代码解析

（1）、在 FrmMain 窗体中添加 TabControl 控件后，将其 Dock 属性设置为 Fill，然后在其 TabPages 属性中添加三个 tabPage 页相应的 Text 属性值设置为：One、Two、Three。

（2）、在 TabControl 三个 tabPage 页中分别添加三个 PictureBox 控件，其 Dock 属性分别设置为 Fill。

（3）、在FrmMain窗体的Load事件代码块中分别向向PictureBox控件添加背景图片；实例化

一个Image类的对象，使用Image.FromFile()方法得到图像并赋值该对象，然后将该对象赋值给BackgroundImage；使用BackgroundImageLayout修改BackgroundImage的布局，这里选择Stretch；此处你也可以直接在PictureBox的属性窗口中修改。

窗体加载事件具体代码如下所示：

```
    privatevoid FrmMain_Load(object sender, EventArgs e)
    {
Image img = Image.FromFile("1.jpg");
                pictureBox1.BackgroundImage = img;//设置PictureBox1的背景图片
                pictureBox1.BackgroundImageLayout = ImageLayout.Stretch;//设置背景图片的
布局

                pictureBox2.BackgroundImage = Image.FromFile("2.jpg");
                pictureBox2.BackgroundImageLayout = ImageLayout.Stretch;

                pictureBox3.BackgroundImage = Image.FromFile("3.jpg");
                pictureBox3.BackgroundImageLayout = ImageLayout.Stretch;

TabPage tp = newTabPage("Four");

PictureBox pictureBox4 = newPictureBox();
                pictureBox4.Dock = DockStyle.Fill;
                pictureBox4.BackgroundImage = Image.FromFile("4.jpg");
                pictureBox4.BackgroundImageLayout = ImageLayout.Stretch;

                tp.Controls.Add(pictureBox4);//将pictureBox4添加到tabPage为Four的页面中
                tabControl1.TabPages.Add(tp);//将tabPage添加到TabControl中

    }
```

A0113 指导文档　Timer 控件学习

1. 实验目的

该实验的主要目的是让学生知道如何使用 Timer 控件

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

Timer 类

实现按用户定义的时间间隔引发事件的计时器。 此计时器最宜用于 Windows 窗体应用程序中，并且必须在窗口中使用。

所属命名空间:System.Windows.Forms

Timer 控件

Timer 控件用于背景进程中，它是不可见的。

对于 Timer 控件以外的其它控件的多重选择，不能设置 Timer 的 Enabled 属性。

在运行于 Windows 95 或 Windows NT 下的 Visual Basic 5.0 中可以有多个活动的定时器控件，对此，实际上并没有什么限制。

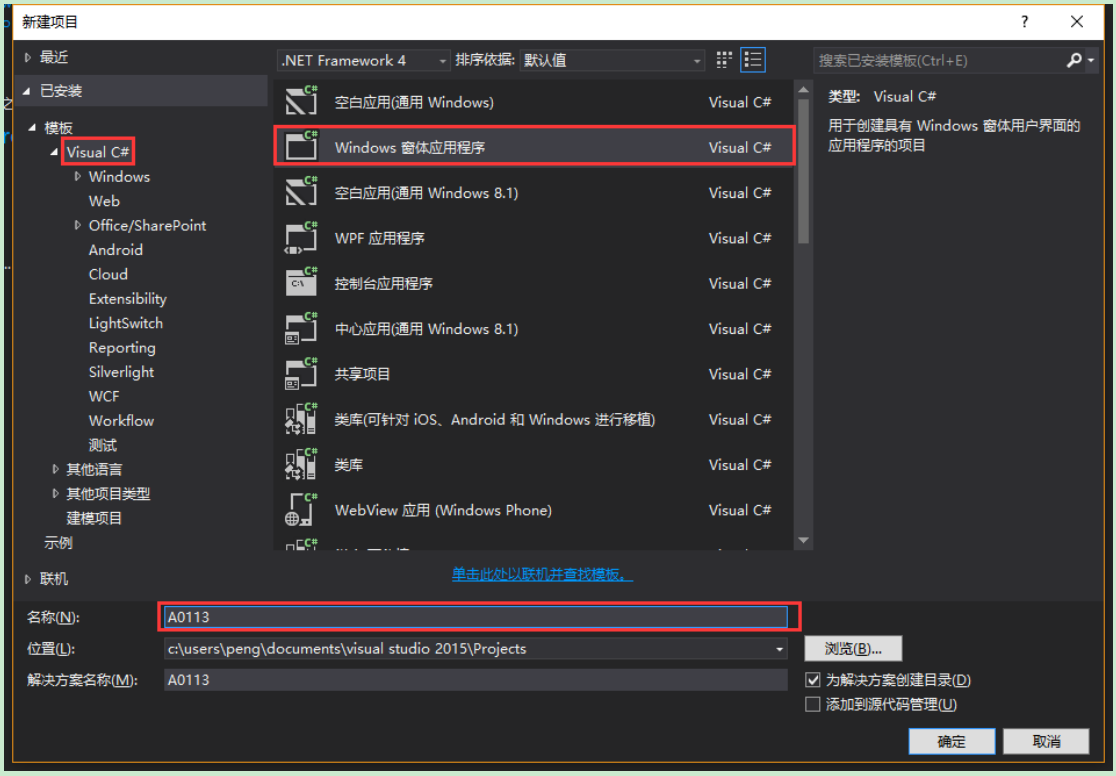
补充:Timer 控件通俗来说就是计时器,这是一个不可视控件.它的重要属性有Interval,Enabled.

它的 Tick 事件指的是每经过 Interval 属性指定的时间间隔时发生一次.

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

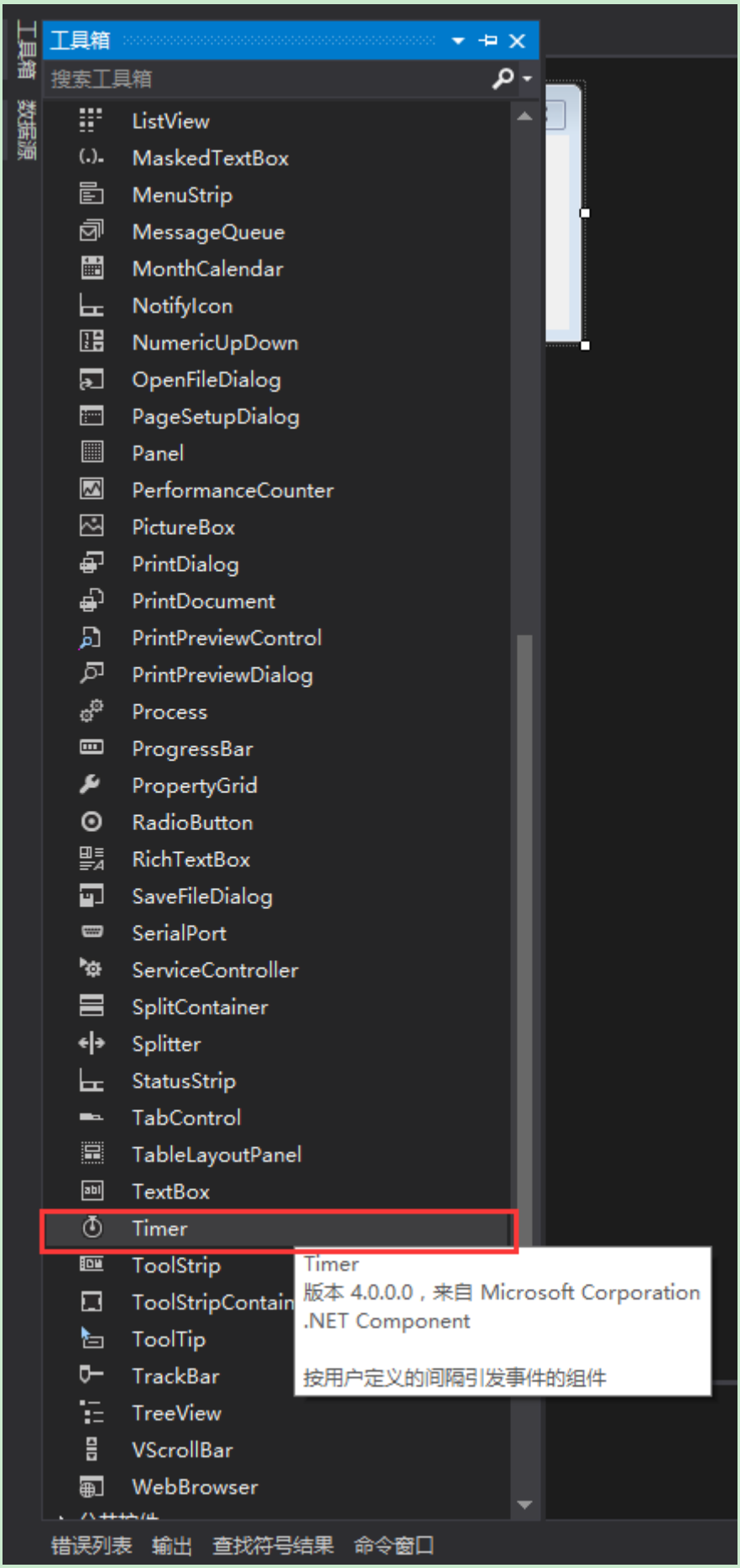
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



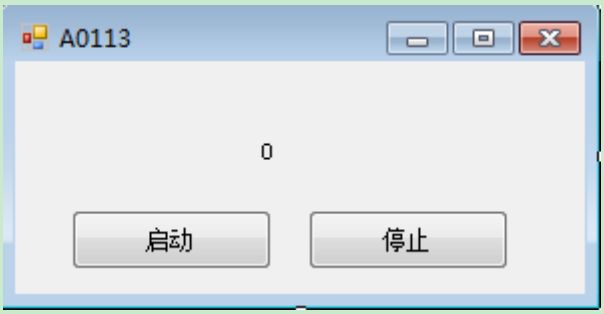
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 Timer 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
PageSetupDialog
Panel
PerformanceCounter
PictureBox
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
PropertyGrid
RadioButton
RichTextBox
SaveFileDialog
SerialPort
ServiceController
SplitContainer
Splitter
StatusStrip
TabControl
TableLayoutPanel
TextBox
Timer
ToolStrip
ToolStripContain
ToolTip
TrackBar
TreeView
VScrollBar
WebBrowser
Timer
版本 4.0.0.0，来自 Microsoft Corporation
.NET Component
按用户定义的间隔引发事件的组件
错误列表
输出
查找符号结果
命令窗口

控件名称	控件 Text 属性	控件 Name 属性	功能
Form 窗体	A0113	frmMain	
Timer 控件	timer1	timer1	按用户定义的时间间隔引发事件
Button 按钮	启动	btnStart	启动 Timer 控件
Button 按钮	停止	btnStop	停止 Timer 控件
Lable 控件	lblNumber	0	



4、控件解析

常用属性及介绍：

Name ： 表示一个控件或者窗体的名称。

Interval：设置 timer 的 Tick 事件触发频率,以毫秒为单位(每 xx 毫秒触发)

常用事件及介绍：

Tick 事件:在指定的时间间隔里调用此事件

添加 timer 控件时钟事件

添加启动按钮单击事件

添加停止按钮单击事件

5. 实验代码解析

A)设置 Timer 控件的时间间隔及事件

设置 Timer 控件的 Interval 属性(时间间隔)为 1000(既 1 秒),设置 Timer 控件的 Tick 事件,在该事件中编写如下代码:

```
i=i+1;
lblNumber.Text=i.ToString();
```

注：变量 i 需要在前面定义

B)启动或停止 Timer 控件

单击启动按钮将会启动 Timer 控件,程序将循环引发 Timer 控件的 Tick 事件

单击停止按钮程序将停止引发 Timer 控件的 Tick 事件

timer 控件时钟事件

```
privatevoid timer1_Tick(object sender, EventArgs e)
    {
            lblNumber.Text = i.ToString();
            i = i + 1;
```

```
//lblNumber.Text = i.ToString();//在lable标签"lblNumber"上显示i的值,当按下"启动按钮后,
标签lblNumber显示的值会每1s加1
//启动Timer后程序将会按照timer的事件间隔的时间循环引发tick事件
if (i > 10)
            {
                timer1.Stop();
                lblNumber.Text = i.ToString();
Thread.Sleep(1000);
                i = 0;
                lblNumber.Text = i.ToString();

            }
    }
```

启动按钮单击事件

```
    privatevoid btnStart_Click(object sender, EventArgs e)
    {
            timer1.Start();//启动Timer控件
    }
```

停止按钮单击事件

```
    privatevoid btnStop_Click(object sender, EventArgs e)
    {
            timer1.Stop();//停止Timer控件
    }
```

A0114 指导文档　ToolStrip 控件学习

1. 实验目的

主要目的是学会展示 ToolStrip 的 Items 集合。

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

ToolStrip 类
为 Windows 工具栏对象提供容器。
所属命名空间:System.Windows.Forms
ToolStrip 控件
使用 ToolStrip 及其关联的类，可以创建具有 Microsoft® Windows® XP、

Microsoft Office、Microsoft 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。ToolStrip 控件提供丰富的设计时体验，包括就地激活和编辑、自定义布局、漂浮（即工具栏共享水平或垂直空间的能力）。

尽管 ToolStrip 替换了早期版本的控件并添加了功能，但是仍可以在需要时选择保留 ToolBar 以备向后兼容和将来使用。

使用 ToolStrip 控件可以：

创建易于自定义的常用工具栏，让这些工具栏支持高级用户界面和布局功能，如停靠、漂浮、带文本和图像的按钮、下拉按钮和控件、“溢出”按钮和 ToolStrip 项的运行时重新排序。

支持操作系统的典型外观和行为。

对所有容器和包含的项进行事件的一致性处理，处理方式与其他控件的事件相同。

将项从一个 ToolStrip 拖到另一个 ToolStrip 内。

使用 ToolStripDropDown 中的高级布局创建下拉控件及用户界面类型编辑器。

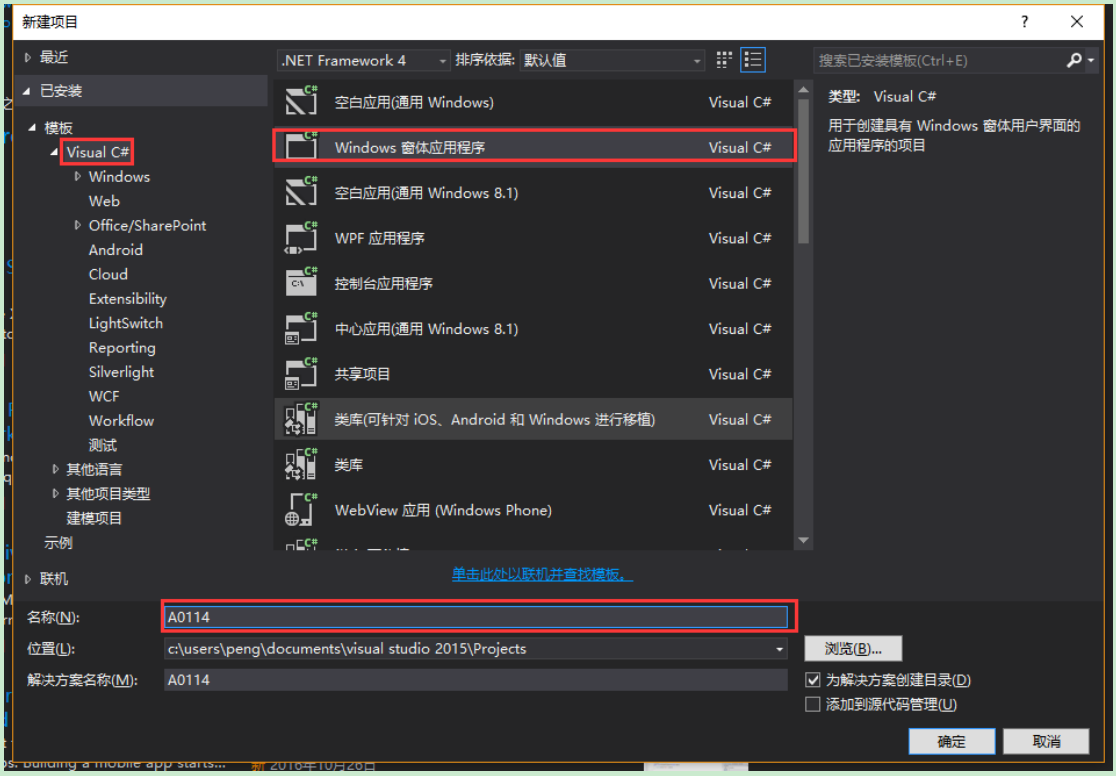
通过使用 ToolStripControlHost 类来使用 ToolStrip 中的其他控件，并为它们获取 ToolStrip 功能。

通过使用 ToolStripRenderer、ToolStripProfessionalRenderer 和 ToolStripManager 以及 ToolStripRenderMode 枚举和 ToolStripManagerRenderMode 枚举，可以扩展此功能并修改外观和行为。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

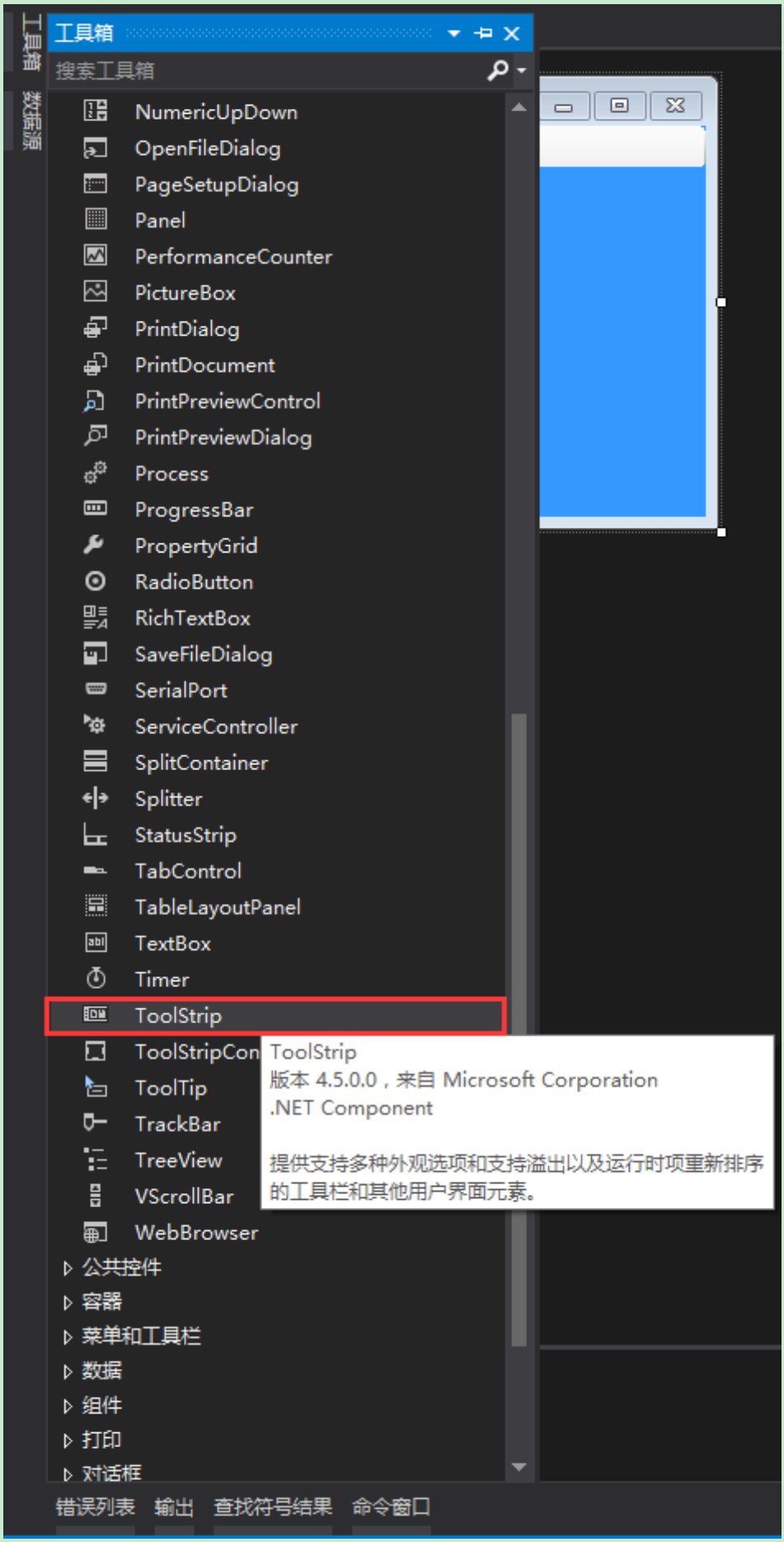
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



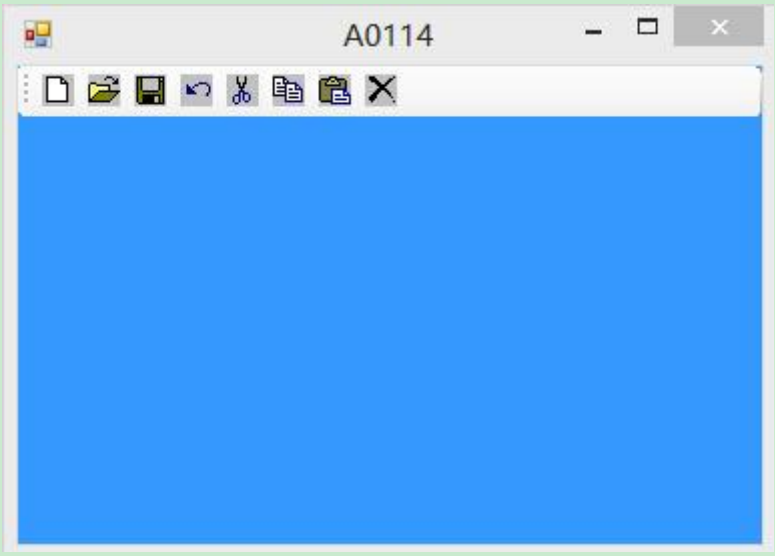
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 ToolStrip 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
NumericUpDown
OpenFileDialog
PageSetupDialog
Panel
PerformanceCounter
PictureBox
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
PropertyGrid
RadioButton
RichTextBox
SaveFileDialog
SerialPort
ServiceController
SplitContainer
Splitter
StatusStrip
TabControl
TableLayoutPanel
TextBox
Timer
ToolStrip
ToolStripCon
ToolTip
TrackBar
TreeView
VScrollBar
WebBrowser
ToolStrip
版本 4.5.0.0，来自 Microsoft Corporation
.NET Component
提供支持多种外观选项和支持溢出以及运行时项重新排序的工具栏和其他用户界面元素。
公共控件
容器
菜单和工具栏
数据
组件
打印
对话框
错误列表
输出
查找符号结果
命令窗口

控件名称	控件 Text 属性	控件 Name 属性	功能
Form1 窗体	A0114	frmMain	
ToolStrip 控件			多种外观选项的工具栏



4、控件解析：

常用属性及介绍：

Name：设置该控件的名称。

BackColor ：设置该控件的背景颜色，使用系统栏下的 Highlight。

BackgroundImage:设置该控件的背景图片。

BackgroundImageLayout：设置该控件背景图片的布局，一般选择 Tile。

FlatStyle：设置控件外观。

Font：设置字体样式和大小。

ForeColor：设置字体颜色。

Image：在控件上显示图像。

Items：在 ToolStrip 上显示的项的集合

Text：设置控件中显示的文本。

添加新建按钮单击事件 toolStripButtonNew_Click

添加打开按钮单击事件 toolStripButtonOpen_Click

添加保存按钮单击事件 toolStripButtonSave_Click

添加撤销按钮单击事件 toolStripButtonUndo_Click

添加剪切按钮单击事件 toolStripButtonCut_Click

添加复制按钮单击事件 toolStripButtonCopy_Click

添加粘贴按钮单击事件 toolStripButtonPaste_Click

添加删除按钮单击事件 toolStripButtonDelete_Click

5. 实验代码解析

向 Form1 窗体添加一个 ToolStrip 控件，并且向 Items 中添加集合，更改对应的属性，

Image 属性的图片来源于 bmp 文件夹。

Name 属性	ToolTipText 属性	Image 属性
toolStripButtonNew	新建	NEW.BMP
toolStripButtonOpen	打开	OPEN.BMP
toolStripButtonSave	保存	SAVE.BMP
toolStripButtonUndo	撤销	UNDO.BMP
toolStripButtonCut	剪切	CUT.BMP
toolStripButtonCopy	复制	COPY.BMP
toolStripButtonPaste	粘贴	PASTE.BMP
toolStripButtonDelete	删除	DELETE.BMP

新建按钮单击事件代码如下所示

```
privatevoid toolStripButtonNew_Click(object sender, EventArgs e)
    {
MessageBox.Show("我是新建按钮！");
    }
```

打开按钮单击事件代码如下所示

```
privatevoid toolStripButtonOpen_Click(object sender, EventArgs e)
    {
MessageBox.Show("我是打开按钮！");
    }
```

保存按钮单击事件代码如下所示

```
privatevoid toolStripButtonSave_Click(object sender, EventArgs e)
    {
MessageBox.Show("我是保存按钮");
     }
```

撤销按钮单击事件代码如下所示

```
privatevoid toolStripButtonUndo_Click(object sender, EventArgs e)
    {
MessageBox.Show("我是撤销按钮");
    }
```

剪切按钮单击事件代码如下所示

```
privatevoid toolStripButtonCut_Click(object sender, EventArgs e)
    {
MessageBox.Show("我是剪切按钮");
    }
```

复制按钮单击事件代码如下所示

```
privatevoid toolStripButtonCopy_Click(object sender, EventArgs e)
    {
MessageBox.Show("我是复制按钮");
    }
```

粘贴按钮单击事件代码如下所示

```
privatevoid toolStripButtonPaste_Click(object sender, EventArgs e)
    {
MessageBox.Show("我是粘贴按钮");
```

```
    }
    删除按钮单击事件代码如下所示
    privatevoid toolStripButtonDelete_Click(object sender, EventArgs e)
    {
MessageBox.Show("我是删除按钮");
    }
```

A0115 指导文档　TreeView 控件学习

1. 实验目的

主要目的是学会用 TreeView 动态添加 TreeNode。

2. 实验设备

软件：visualstudio2010 及以上版本，

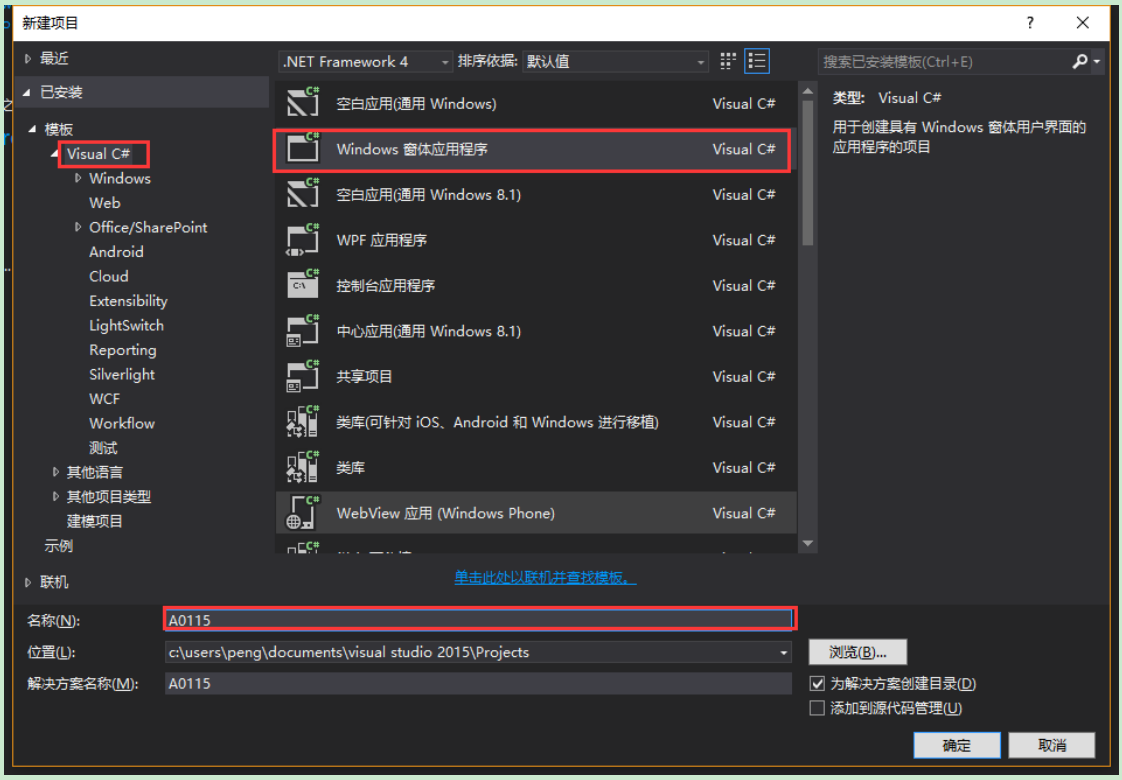
3. 实验原理

TreeView 类
显示标记的每个表示项的分层集合 TreeNode。
所属命名空间:System.Windows.Forms
TreeView 控件
TreeView 控件用来显示信息的分级视图，如同 Windows 里的资源管理器的目录。TreeView 控件中的各项信息都有一个与之相关的 Node 对象。TreeView 显示 Node 对象的分层目录结构，每个 Node 对象均由一个 Label 对象和其相关的位图组成。在建立 TreeView 控件后，我们可以展开和折叠、显示或隐藏其中的节点。TreeView 控件一般用来显示文件和目录结构、文档中的类层次、索引中的层次和其他具有分层目录结构的信息。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

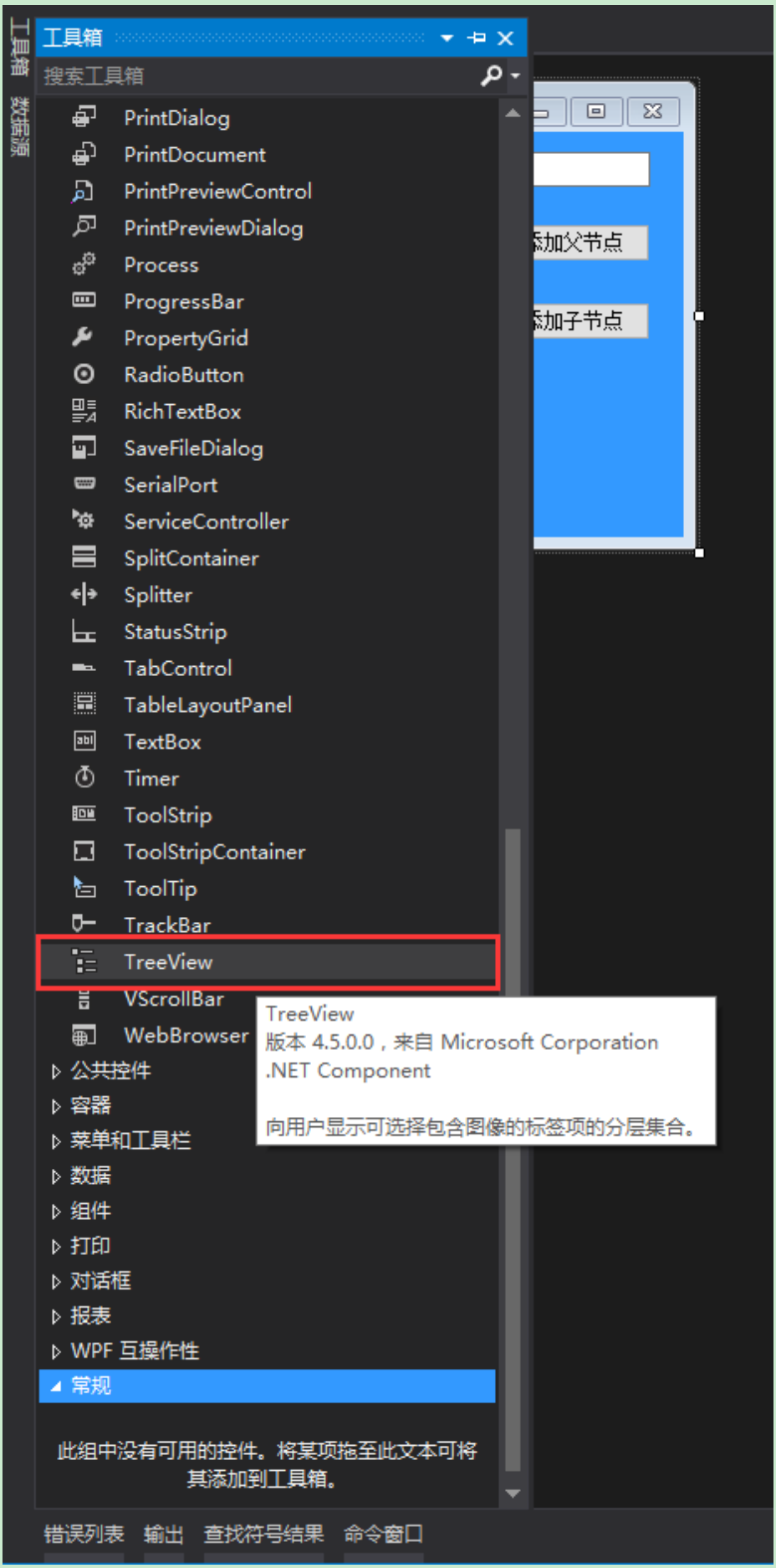
2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。



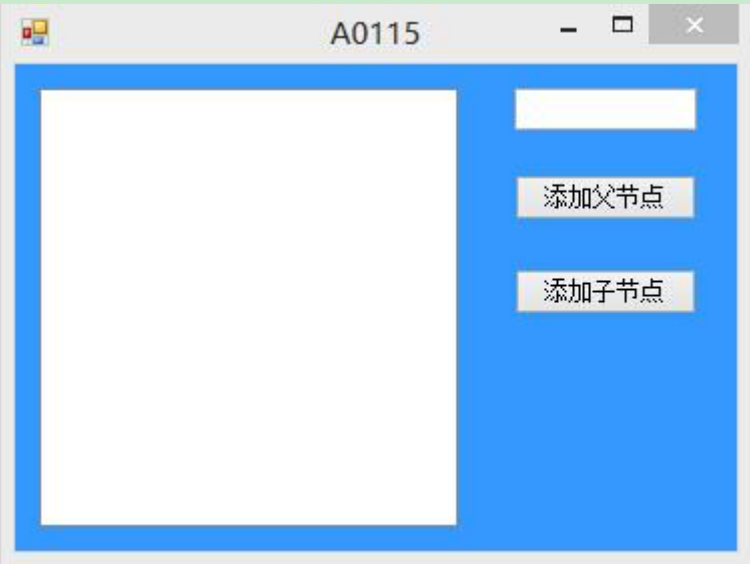
3、界面设计及控件属性

添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜单，找到工具箱，如下图所示

在工具箱中找到 datetimepicker 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
数据源
工具箱
搜索工具箱
PrintDialog
PrintDocument
PrintPreviewControl
PrintPreviewDialog
Process
ProgressBar
PropertyGrid
RadioButton
RichTextBox
SaveFileDialog
SerialPort
ServiceController
SplitContainer
Splitter
StatusStrip
TabControl
TableLayoutPanel
TextBox
Timer
ToolStrip
ToolStripContainer
ToolTip
TrackBar
TreeView
VScrollBar
WebBrowser
公共控件
容器
菜单和工具栏
数据
组件
打印
对话框
报表
WPF 互操作性
常规
此组中没有可用的控件。将某项拖至此文本可将其添加到工具箱。
TreeView
版本 4.5.0.0，来自 Microsoft Corporation
.NET Component
向用户显示可选择包含图像的标签项的分层集合。
添加父节点
添加子节点
错误列表
输出
查找符号结果
命令窗口

控件名称	控件 Text 属性	控件 Name 属性	功能
Form1 窗体	A0115	frmMain	
Button 按钮	添加父节点	button1	单击触发事件
Button 按钮	添加子节点	button2	单击触发事件
TextBox		textBox1	输入文本
TreeView 控件			显示包含图像的分层集合



4、控件解析：

常用属性及介绍：

Name：设置该控件的名称。

BackColor ：设置该控件的背景颜色，使用系统栏下的 Highlight。

BackgroundImage:设置该控件的背景图片。

BackgroundImageLayout：设置该控件背景图片的布局，一般选择 Tile。

FlatStyle：设置控件外观。

Font：设置字体样式和大小。

ForeColor：设置字体颜色。

Image：在控件上显示图像。

Text：设置控件中显示的文本。

常用事件及介绍：

Click 事件：当用户用鼠标左键单击按钮控件时，将发生该事件。

添加添加父节点按钮单击事件

添加添加子节点按钮单击事件

5. 实验代码解析

请大家了解 TreeView 控件的原理，然后根据示例代码编写程序

添加父节点方法：

```
    privatevoid AddParent()
    {
if (textBox1.Text != "")
                {    //创建一个节点对象，并初始化
TreeNode RootNode = newTreeNode(textBox1.Text);
//在TreeView组件中加入父节点
                    treeView1.Nodes.Add(RootNode);
                    textBox1.Text = null;//清空textBox1的文本
                    treeView1.ExpandAll();//展开所以树节点
                }
else
                {
MessageBox.Show("TextBox组件必须填入节点名称！", "提示信息", MessageBoxButtons.OK,
MessageBoxIcon.Information);
return;
                }
    }
```

添加父节点按钮单击事件代码如下所示：

```
    privatevoid button1_Click(object sender, EventArgs e)
    {
            AddParent();    //添加父节点方法
    }
```

添加子节点方法

```
    privatevoid AddChildNode()
    {    //首先判断是否选定组件中的位置
if (treeView1.SelectedNode == null)
            {
MessageBox.Show("请选择一个节点", "提示信息", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            }
else
            {
if (textBox1.Text != "")
                {    //创建一个节点对象，并初始化
TreeNode ChildNode = newTreeNode(textBox1.Text);
                    treeView1.SelectedNode.Nodes.Add(ChildNode);        //在TreeView组件
中加入子节点
                    treeView1.SelectedNode = ChildNode;
                    textBox1.Text = null;//清空textBox1的文本
                    treeView1.ExpandAll();//展开所以树节点
                }
else
                {
MessageBox.Show("TextBox组件必须填入节点名称！", "提示信息", MessageBoxButtons.OK,
```

```
MessageBoxIcon.Information);
return;
                }
            }
    }
```

添加子节点按钮单击事件代码如下所示：

```
privatevoid button2_Click(object sender, EventArgs e)
    {
            AddChildNode();//添加子节点方法
    }
```

A0116 指导文档　ComboBox 控件学习

1. 实验目的

主要目的是掌握向 ComboBox 控件中添加 Item 以及设置 DropDownClosed、SelectedIndexChanged 事件。

2. 实验设备

软件：visualstudio2010 及以上版本，

3. 实验原理

ComboBox 类

表示 Windows 组合框控件。

所属命名空间:System.Windows.Forms

ComboBox 控件

Windows 窗体 ComboBox 控件用于在下拉组合框中显示数据。默认情况下，ComboBox 控件分两个部分显示：顶部是一个允许用户键入列表项的文本框。第二部分是一个列表框，它显示一个项列表，用户可从中选择一项。有关组合框的其他样式的更多信息，请参见何时使用 Windows 窗体 ComboBox 而非 ListBox。

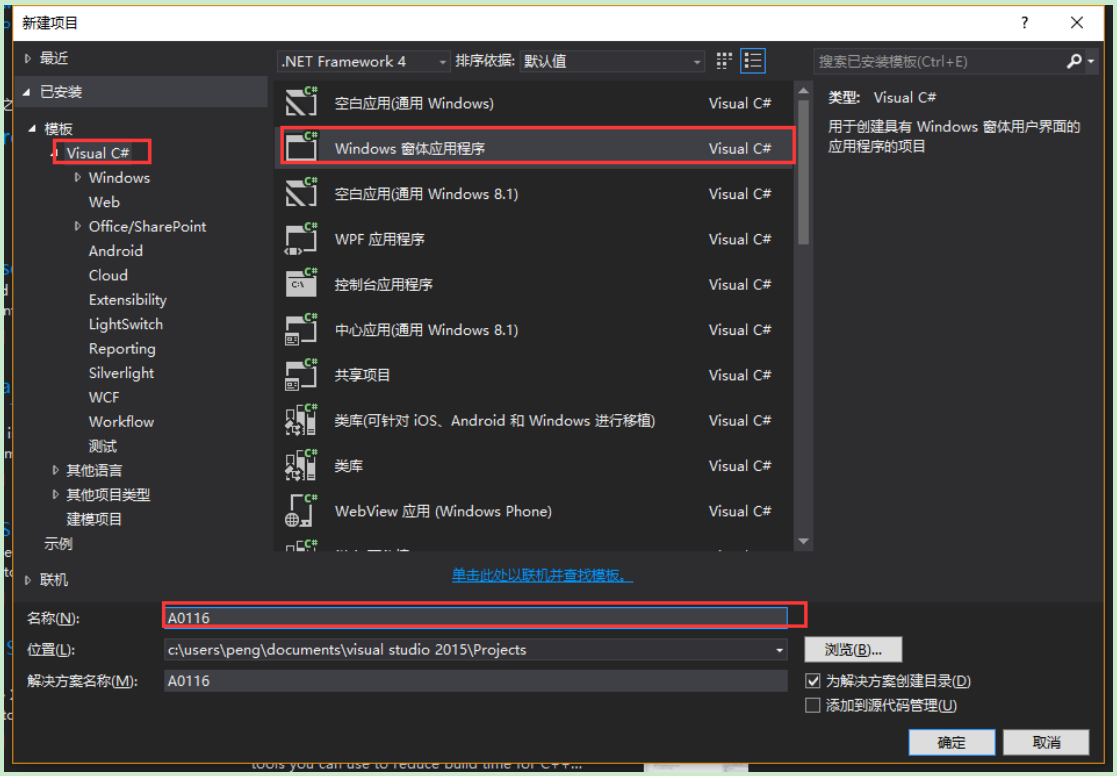
SelectedIndex 属性返回一个整数值，该值与选择的列表项相对应。通过在代码中更改 SelectedIndex 值，可以编程方式更改选择项；列表中的相应项将出现在组合框的文本框部分。如果未选择任何项，则 SelectedIndex 值为 -1。如果选择列表中的第一项，则 SelectedIndex 值为 0。SelectedItem 属性与 SelectedIndex 类似，但它返回项本身，通常是一个字符串值。Count 属性反映列表的项数，由于 SelectedIndex 是从零开始的，所以 Count 属性的值通常比 SelectedIndex 的最大可能值大一。

若要在 ComboBox 控件中添加或删除项，请使用 Add、Insert、Clear 或 Remove 方法。或者，可以在设计器中使用 Items 属性向列表添加项。

4. 实验设计

1、启动 visualstudio，文件→新建→项目。

2、选择 VisualC#→Windows 窗体应用程序，输入名称→选择存储路径。

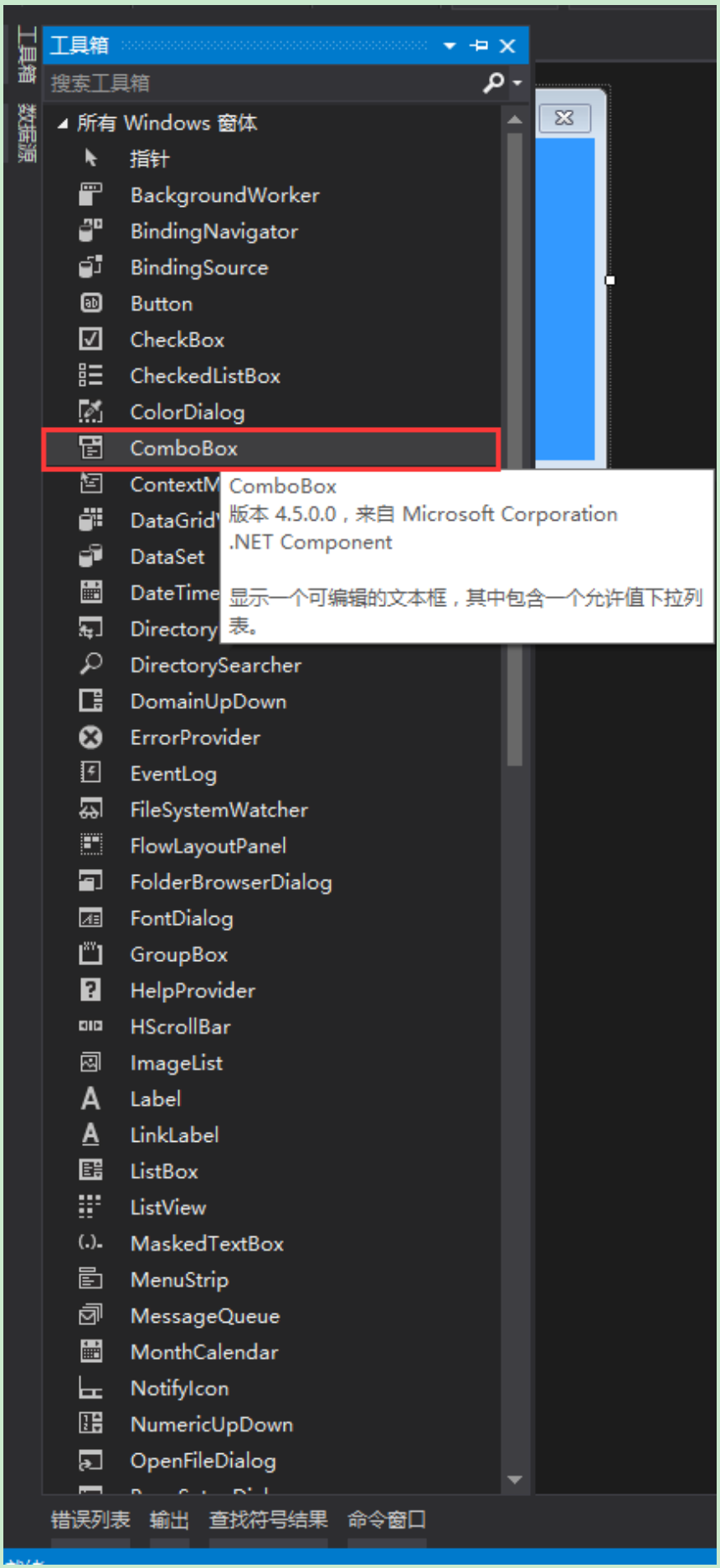


3、界面设计及控件属性

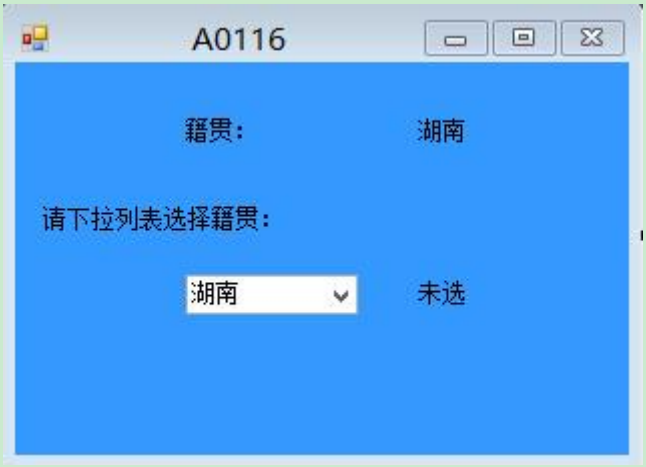
添加控件方法：打开工具箱，在窗体左上侧有工具栏选项，如果未找到，请打开视图菜

单，找到工具箱，如下图所示

在工具箱中找到 ComboBox 控件，双击或者拖拽都可以添加控件到窗体中，


工具箱
搜索工具箱
所有 Windows 窗体
指针
BackgroundWorker
BindingNavigator
BindingSource
Button
CheckBox
CheckedListBox
ColorDialog
ComboBox
ContextM
DataGrid
DataSet
DateTime
Directory
DirectorySearcher
DomainUpDown
ErrorProvider
EventLog
FileSystemWatcher
FlowLayoutPanel
FolderBrowserDialog
FontDialog
GroupBox
HelpProvider
HScrollBar
ImageList
Label
LinkLabel
ListBox
ListView
MaskedTextBox
MenuStrip
MessageQueue
MonthCalendar
NotifyIcon
NumericUpDown
OpenFileDialog
ComboBox
版本 4.5.0.0，来自 Microsoft Corporation
.NET Component
显示一个可编辑的文本框，其中包含一个允许值下拉列表。
错误列表
输出
查找符号结果
命令窗口

控件名称	控件 Text 属性	控件 Name 属性	功能
Form1 窗体	A0116	frmMain	
Label	未选	Label1	
Label	籍贯	Label2	
Label	湖南	Label3	
Label	请下拉列表选择籍贯：	Label4	
ComboBox 控件中	湖南		下拉组合框中显示数据



4、控件分析：

常用属性及介绍：

Name：设置该控件的名称。

BackColor ：设置该控件的背景颜色，使用系统栏下的 Highlight。

BackgroundImage:设置该控件的背景图片。

BackgroundImageLayout：设置该控件背景图片的布局，一般选择 Tile。

FlatStyle：设置控件外观。

Font：设置字体样式和大小。

ForeColor：设置字体颜色。

Image：在控件上显示图像。

Items：在组合框的项

Text：设置控件中显示的文本。

常用事件及介绍：

SelectedIndexChanged 事件：SelectedIndex 属性值更改时发生。

DropDownClosed 事件：指示组合框下拉部分关闭时可触发的事件。

Load 事件：每当用户加载窗体时发生。

添加窗体加载事件 Form1_Load
添加 ComboBox 选项更改事件 comboBox1_SelectedIndexChanged
添加 ComboBox 下拉列表框关闭事件 comboBox1_DropDownClosed

5. 实验代码解析

窗体加载时向 comboBox1 控件添加 Item，点击选择一个 Item，SelectedIndex 属性值发生改变，并且触发事件，组合框下拉部分关闭时也触发事件。

窗体加载事件代码如下所示：

```
privatevoid Form1_Load(object sender, EventArgs e)    //窗体加载
{
            comboBox1.Items.Add("湖北");    //向comboBox1控件添加Item
            comboBox1.Items.Add("山东");    //向comboBox1控件添加Item
            comboBox1.Items.Add("北京");    //向comboBox1控件添加Item
            comboBox1.Items.Add("重庆");    //向comboBox1控件添加Item
            comboBox1.Items.Add("天津");    //向comboBox1控件添加Item
            comboBox1.Items.Add("河北");    //向comboBox1控件添加Item
}
```

ComboBox 选项更改事件代码如下所示：

```
privatevoid comboBox1_SelectedIndexChanged(object sender, EventArgs e)
//SelectedIndex属性值改变时发生的事件
{
                        label3.Text = comboBox1.Text;           //将选中的数据通过label3
显示出来
}
```

ComboBox 下拉列表框关闭事件代码如下所示：

```
privatevoid comboBox1_DropDownClosed(object sender, EventArgs e)    //组合框下拉部
分已关闭触发的事件
{
            label1.Text = "已选";    //提示用户已经选择
            label4.Text = "";         //将未选之前的提示信息隐藏
}
```