

《C#程序设计》

第七章 Windows窗体及控件

武汉大学计算机学院 贾向阳



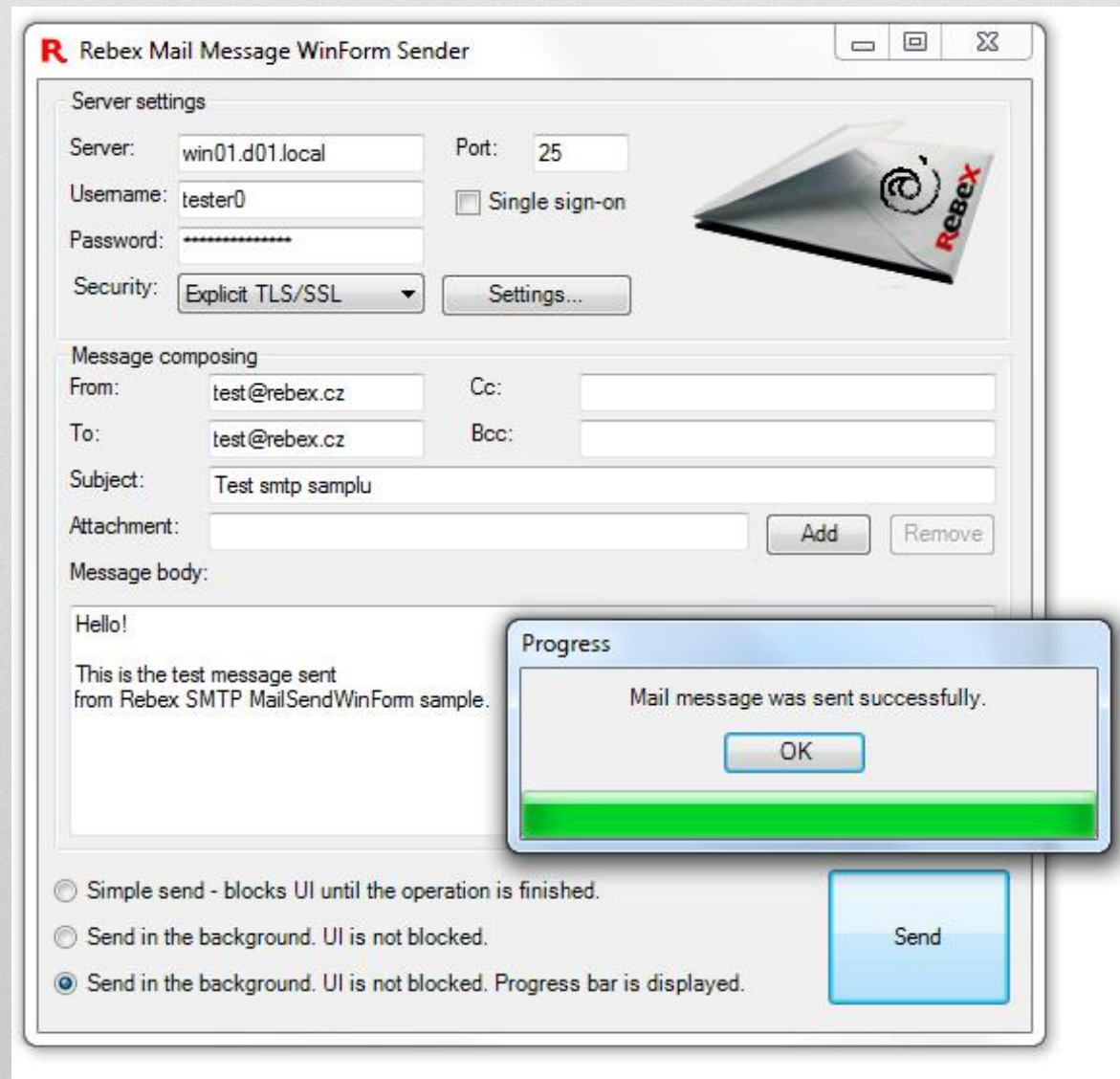
7.1 窗体应用程序概述



1、Windows 桌面应用开发技术

(1) Windows Form

- 提供了一套标准化控件和界面风格
- 事件驱动的可视化开发

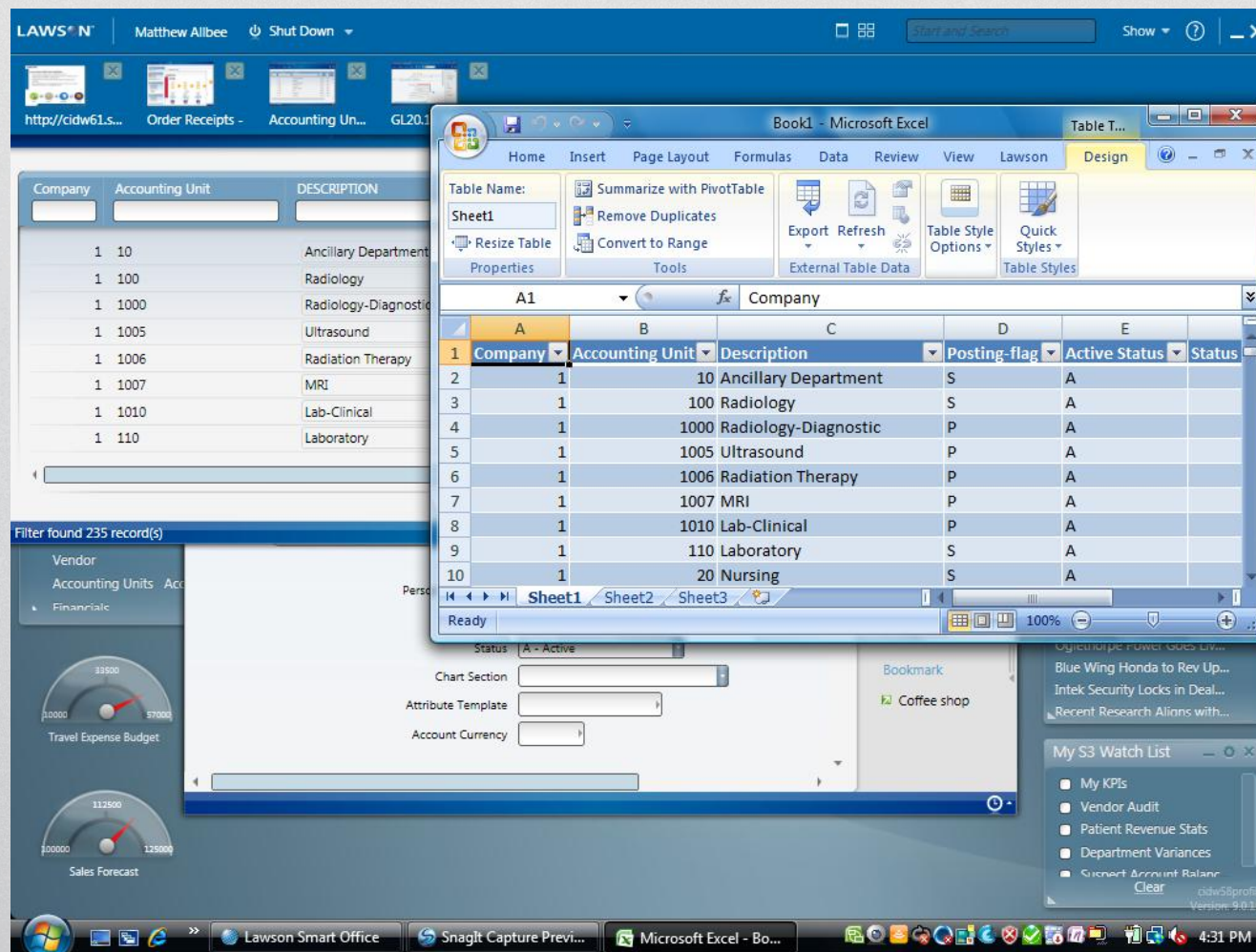




1、Windows 桌面应用开发技术

(2) WPF

- 微软继Winform之后的新一代桌面应用开发技术
- 更漂亮的界面、自适应分辨率
- 双向数据绑定





Windows 桌面应用开发技术

(3) DevExpress类库

- 提供了风格更现代的WinForms控件
- 商业软件

DevExpress - Employees

HOME VIEW

New Employee New Group New Items Delete Edit Print Meeting Task Mail Merge

Thank You Note Service Excellence Welcome To DevAV Employee Award Probation Notice

List Card Map It Custom Filter Find Getting Started Get Free Support Buy Now About

Search Employees (Ctrl + F)

DEPARTMENT

Engineering (Count=9)

FULL NAME	ADDRESS	CITY	STATE	ZIP CODE	EMAIL
Bart Arnaz	13109 Old Myford Rd	Irvine	CA	92602	barta@dx-email.com
Leah Simpson	6755 Newlin Ave				
Arnie Schwartz	125 W Elm St				
Billy Zimmer	1325 Prospect Dr				
Samantha Piper	200 E Ave 43				
Maggie Boxter	3101 W Harvard St				
Terry Bradley	4595 Cochran St				
Brad Farkus	1635 Woods Drive				
Stu Pizaro	200 N. Spring St				

Human Resources (Count=6)

Greta Sims	1700 S Grandview Dr.				
Sandra Johnson	4600 N Virginia Rd.				
Cindy Stanwick	2211 Bonita Dr.				
Marcus Orbison	501 N Main St				
Sandy Bright	7570 McGroarty Ter				
Ken Samuelson	12100 Mora Dr				

IT (Count=8)

Brett Wade	1120 Old Mill Rd.				
Taylor Riley	7776 Torreyson Dr				
Amelia Harper	527 W 7th St				
Wally Hobbs	10385 Shadow Oak Dr				
Brad Jameson	1100 Pico St				
Karen Goodson	309 Monterey Rd				
Morgan Kennedy	11222 Dilling St				
Violet Bailey	1418 Descanso Dr				

Management (Count=4)

John Heart	351 S Hill St.				
Samantha Bright	5801 Wilshire Blvd.				
Arthur Miller	3800 Homer St.				

CONTACT

First Name * Arthur

Last Name * Miller

Full Name Arthur Miller

Birth Date 7/11/1972

Title * CTO Prefix Mr

Address 3800 Homer St.

City Los Angeles

State CA ZIP code 90031

Home Phone (310) 555-6542

Mobile Phone * (310) 555-8583

Email * arthurm@dx-email.com

Skype arthurm_DX_skype

Department Management

Status Salaried

Hire Date 12/18/2007

Evaluations

CREATED ON	SUBJECT	MANAGER
11/15/2008	2008 Employee Review	John Heart
11/18/2009	2009 Employee Review	John Heart
11/30/2010	2010 Employee Review	John Heart
12/5/2011	2011 Employee Review	John Heart
11/19/2012	2012 Employee Review	John Heart
12/15/2013	2013 Employee Review	John Heart

Tasks

PRIORITY	DUE DATE	SUBJECT	DESCRIPTION	COMPLETION
High	3/10/2013	Deliver R&D Plans for 2013	Marketing strategy is set. R&D needs to deliver a detailed report on product develop...	100%
High	8/2/2013	Decide on Mobile Devices to Use in the Field	We need to decide whether we are going to use Surface tablets in the field or go wi...	100%
High	8/15/2013	Try New Touch-Enabled WinForms Apps	The field will get this rolled out next week. You should try the apps on the new Surfa...	100%
High	1/31/2014	Approval on Converting to New HDMI Specifi...	My report is not complete and in order to complete it, I need approval to invest \$250...	75%
High	3/18/2014	Return Merchandise Report	Need to see the number of returns this month as I'm told by accounting that our ref...	25%
High	4/24/2014	Approve Salary Increase Request	I am told by Bart that we have a salary freeze and that my request for an increase in s...	0%

Employees Customers Products

RECORDS: 51



WinForm 的窗体和控件类

命名System.Windows.Forms命名空间

- 提供了Windows GUI相关的类
- Form类：窗口和对话框
- Control类：各种控件



VS中开发WinForm应用

The image shows the Visual Studio IDE interface for a WinForm application. The following components are labeled with Chinese text and lines pointing to them:

- 标题栏** (Title Bar): Points to the top bar of the main window.
- 菜单栏** (Menu Bar): Points to the menu bar containing File, Edit, View, etc.
- 工具栏** (Toolbar): Points to the toolbar with icons for Save, Undo, etc.
- 工具箱** (Toolbox): Points to the left sidebar containing a list of Windows Forms controls like FlowLayoutPanel, GroupBox, etc.
- 窗体** (Form): Points to the central design area where the Form1 window is being designed.
- 窗体设计器窗口** (Form Designer Window): Points to the central design area.
- 解决方案资源管理器** (Solution Explorer): Points to the right sidebar showing the project structure (proj9-1).
- 属性窗口** (Properties Window): Points to the bottom right area showing the properties of the selected control (Form1).
- 属性含义说明** (Property Description): Points to the text description of the selected property (Text) in the Properties window.



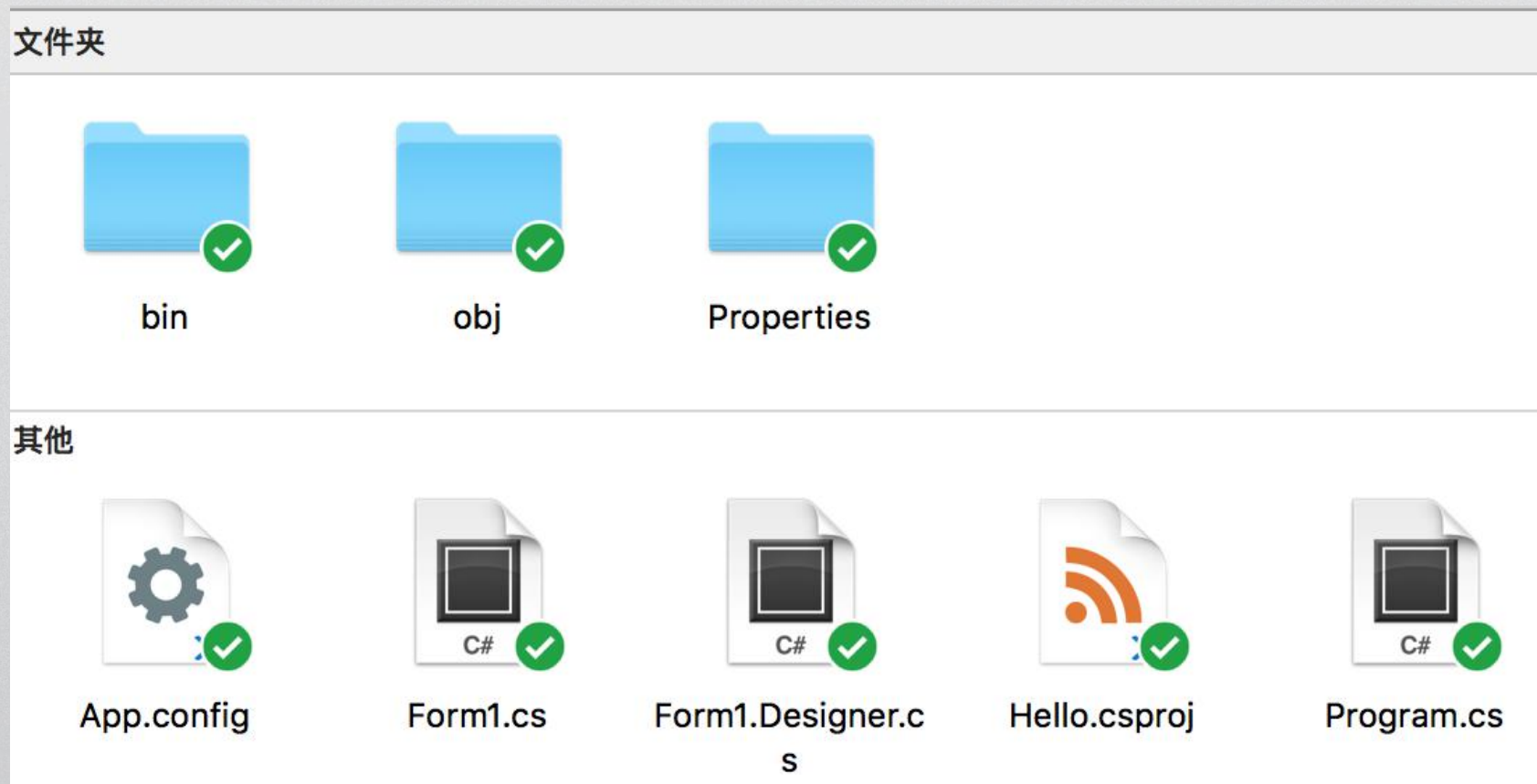
WinForm应用开发的主要步骤

- 创建窗体 (Form)
- 创建控件 (Control)
 - 创建组成界面的各种元素，如按钮，文本框等
- 指定布局 (Layout)
 - 排列控件之间的位置关系
- 响应事件 (Event)
 - 定义图形用户界面的事件和各界面元素对不同事件的响应，实现用户交互



WinForm程序的结构

使用可视化方式创建一个简单的窗体，代码结构如下：





WinForm程序的结构

```
static class Program
{
    /// <summary>
    /// 应用程序的主入口点。
    /// </summary>
    [STAThread]
    static void Main()
    {
        Application.EnableVisualStyles();
        Application.SetCompatibleTextRenderingDefault(false);
        Application.Run(new Form1());
    }
}
```

Program.cs类

- [STAThread]:单线程单元的COM线程模型
- Application类：实现应用程序相关的设置、启动、停止。
- Application.Run (new Form1())
 - 启动应用程序，创建Form1类的一个实例，作为初始界面



WinForm程序的结构

```
public partial class Form1 : Form{  
    public Form1() {  
        InitializeComponent();  
    }  
}
```

Form1.cs类

```
partial class Form1{  
    //必需的设计器变量。  
    private System.ComponentModel.IContainer components = null;  
  
    //清理所有正在使用的资源。  
    protected override void Dispose(bool disposing) {  
        if (disposing && (components != null)){  
            components.Dispose();  
        }  
        base.Dispose(disposing);  
    }  
}
```

Form1.Designer.cs类

#region Windows 窗体设计器生成的代码

```
private void InitializeComponent(){  
    this.components = new System.ComponentModel.Container();  
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;  
    this.ClientSize = new System.Drawing.Size(800, 450);  
    this.Text = "Form1";  
}  
#endregion
```



窗体上添加一个按钮后....

partial class Form1{

Form1.Designer.cs类

.....

private System.Windows.Forms.Button button1;

#region Windows 窗体设计器生成的代码

private void InitializeComponent(){

this.button1 = new System.Windows.Forms.Button();

this.SuspendLayout();

// button1

this.button1.Location = new System.Drawing.Point(56, 26);

this.button1.Name = "button1";

this.button1.Size = new System.Drawing.Size(75, 23);

this.button1.TabIndex = 0;

this.button1.Text = "button1";

this.button1.UseVisualStyleBackColor = true;

this.Controls.Add(this.button1);

//form1

this.components = new System.ComponentModel.Container();

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 450);

this.Text = "Form1";

this.ResumeLayout(false);

}

#endregion



WinForm程序的结构

为按钮添加一个Click事件处理函数....

```
partial class Form1{
```

```
    ...
```

```
    this.button1.TabIndex = 0;
```

```
    this.button1.Text = "button1";
```

```
    this.button1.UseVisualStyleBackColor = true;
```

```
    this.button1.Click += new System.EventHandler(this.button1_Click);
```

```
    ...
```

```
}
```

Form1.Designer.cs类

```
[STAThread]
```

```
static void Main() {
```

```
    Application.Run(new Form1 ());
```

```
}
```

```
private void button1_Click(object sender, System.EventArgs e){
```

```
    this.label1.Text = this.textBox1.Text.ToUpper();
```

```
}
```

Form1.cs类





窗体和控制件的布局

- 有4种属性可以用来在窗体中定位控件和调整控件的大小：
- **Location**--以像素为单位，设置控件的X坐标和Y坐标
- **Size**--以像素为单位，设置控件的宽度和高度
- **Anchor (锚点)**--把控件的某个边固定在容器上
- **Dock (停靠)**--把控件停靠在容器的某个位置

- 在一定意义上，前两者是**绝对布局**，后两者是**相对布局**

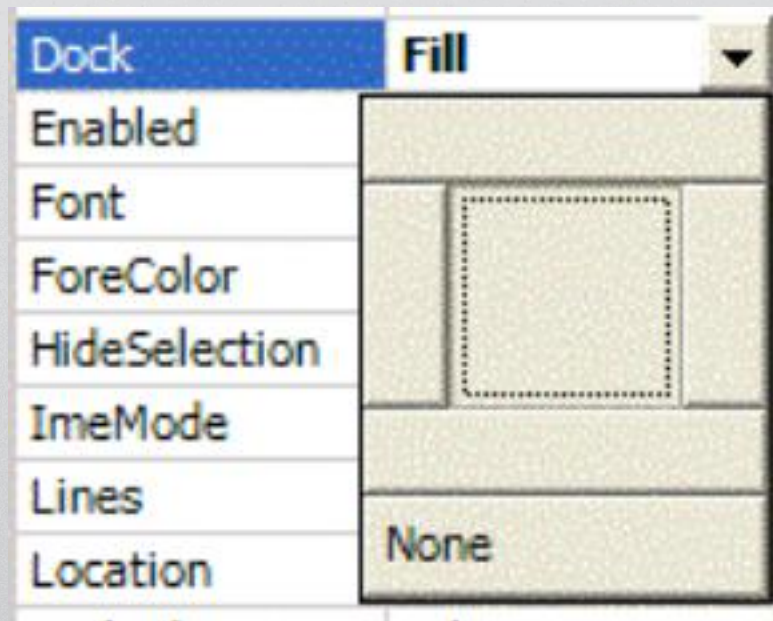


窗体和控制件的布局

■ Dock 属性

□ 设置控件边框停靠到容器某个边上，使控件随容器一起调整大小。

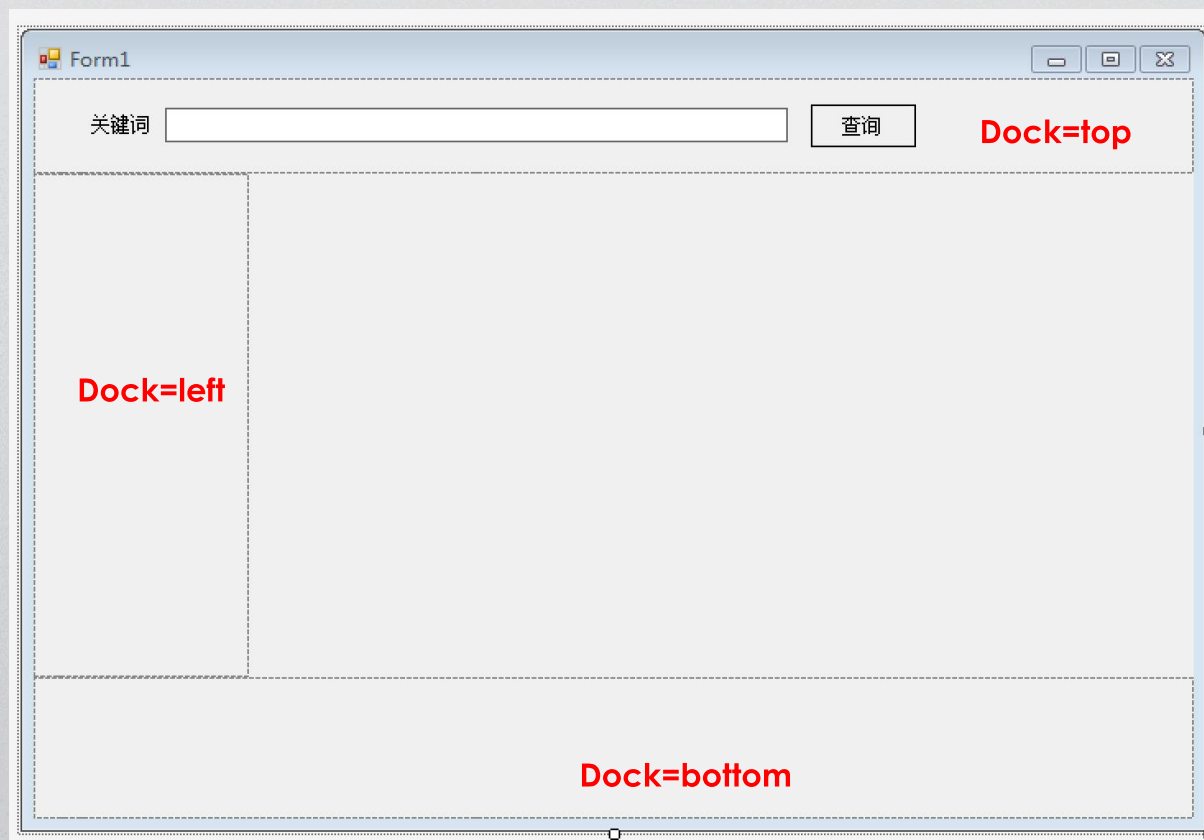
- Fill：四边拉伸
- Left：左停靠，上下拉伸
- Right：右停靠，上下拉伸
- Top：上停靠，左右拉伸
- Bottom：下停靠，左右拉伸





布局技巧

- 技巧1：使用容器类控件进行整体布局
 - 使用Dock属性让容器停靠在窗口的四个边上。

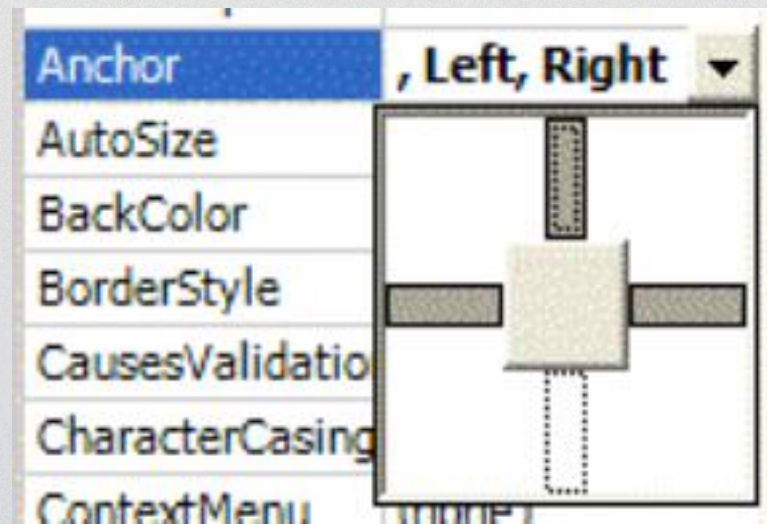




窗体和控件的布局

■ Anchor属性

- 将控件的边固定到容器的某个点上
 - 与左边保持固定距离：设置left
 - 与右边保持固定距离：设置right



■ 布局技巧2:利用锚点实现自适应大小

- 左右拉伸：设置left和right
- 上下拉伸：设置Top和Bottom



事件

- 低级事件与高级事件
 - 如KeyDown/KeyUp KeyPressed TextChange
- 事件及其注册
 - 事件/委托/事件参数
 - `btn.Click += new EventHandler(btn_Clicked)`
 - `void btn_Clicked(object sender, EventArgs e){.....}`
 - 可以使用匿名函数及Lambda表达式
- 示例：MouseEvent.cs

7.2 常用控件



控件



控件Control类

- Windows中控件都是

- System.Windows.Forms.Control的子类

- 实现了IDisposable等接口

- 详见

- [http://msdn.microsoft.com/zh-](http://msdn.microsoft.com/zh-cn/library/system.windows.forms.control(v=vs.110).aspx)

[cn/library/system.windows.forms.control\(v=vs.110\).aspx](http://msdn.microsoft.com/zh-cn/library/system.windows.forms.control(v=vs.110).aspx)





Control 的常用属性

➤ 外观

- Size(大小) , Width (宽度) , Height (高度)
- Location(位置) , Left, Right, Top, Bottom
- Font (字体) , ForeColor (前景色) , BackColor (背景色)

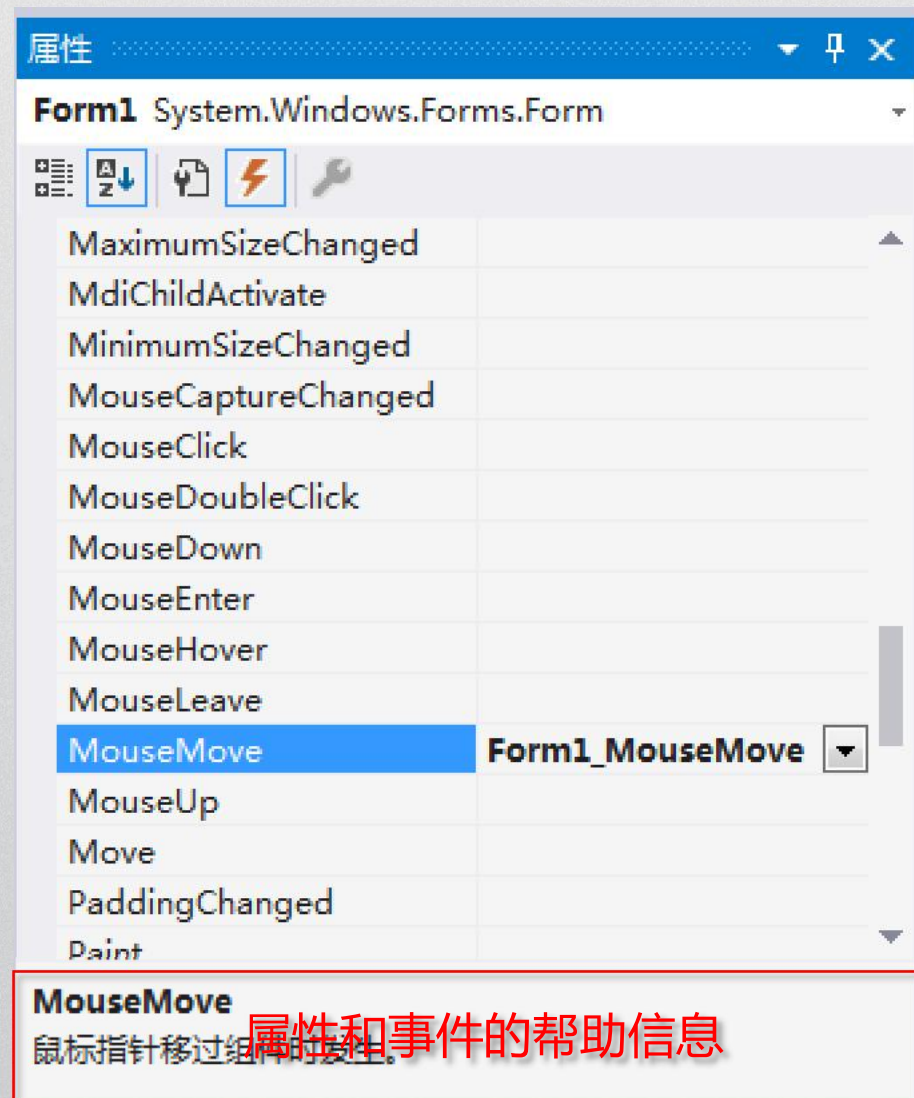
➤ 状态

- Visible (可见) , Enabled (使能)
- Text (文本) , BackgroundImage (背景图片)
- Tag (标记数据 , 类型为object任意类型)



Control 的常用事件

- KeyXXX 键盘事件
 - KeyDown KeyUp KeyPress
- MouseXXX 鼠标事件
 - MouseDown, MouseUp, MouseMove
 - MouseEnter, MouseHover, MouseLeave
- Click/DoubleClick事件
- GotFocus 事件
- TextChange 事件d





常用控件1：Label和LinkLabel

➤ 标签 Label

A Label

- 用来显示文本或图像

➤ 链接标签 LinkLabel

[A LinkLabel](#)

- 可以导航到一个web网站或者应用中的其他窗体
- 链接标签本身不具有导航功能，仅是显示手指形状的鼠标指针
- 利用LinkClicked的事件进行导航处理

```
this.linkLabel1.LinkClicked += this.linkLabel1_LinkClicked;
```

```
private void linkLabel1_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e){  
    this.label1.Text = "Trans to here.";  
}
```




常用控件2：文本框

➤ 文本框基类 **TextBoxBase**，含两类文本控件 **TextBox** 和 **RichTextBox**

- 常用属性

- **BorderStyle**：控件的边框形式
 - None, FixedSingle, Fixed3D
- **Text**：控件中的文本
- **MaxLength**: 控件允许输入的字数
-

- 常用事件

- **Clear**：清楚文本
- **Copy**：复制文本
- **Select**：选择文本
-

abl TextBox

ab Button



常用控件2：文本框

➤ TextBox类

- 添加的属性或事件
 - TextAlign：控件中文本的对齐方式
 - PasswordChar：输入密码时使用隐藏字符
 - AcceptsReturn：若为真，按enter换行，否则，激活窗体中的默认按钮
 - TextAlignChanged:当文本的对齐方式发生改变时激活的事件
 -

➤ RichTextBox类(支持文本的格式处理)

- 增加了40多种属性和许多方法
 - 如：AutoWordSelection，SelectionColor,CanPaste,Find，Redo等





常用控件2：文本框

➤ 文本框的常见操作

```
this.textBox0.Text = "textBox1";
```

```
this.textBox1.lines= new string []{ "first line","second line","third line") "};
```

```
this.textBox2.Multiline = true;
```

```
this.textBox2.Text = "Line One\r\nLine Two\r\nLine Three";
```

```
this.textBox2.ScrollBars = System.Windows.Forms.ScrollBars.Both;
```

```
this.textBox3.PasswordChar = '*';
```

```
this.textBox3.Text = "textBox3";
```




常用控件3：列表框



列表框 ListBox

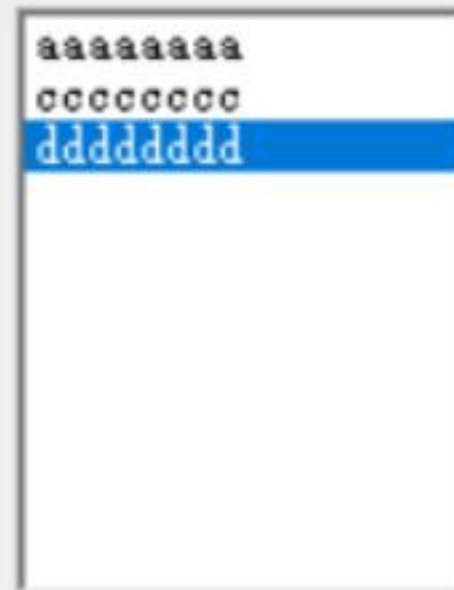


ListBox

- 可以自动添加滚动条，也可以在列表框中选择一项或多项
- 常用属性
 - **Sorted**：条目的有序或无序
 - **IntegralHeight**：若为真，列表框自动调整高度显示所有条目
 - **HorizontalScrollbar**：决定是否为过长的条目显示滚动条
- 添加条目

```
ListBox1.BeginUpdate();  
ListBox1.Item.Add("aaaaaaaaa");  
ListBox1.Item.Add("ccccccccc");  
ListBox1.Item.Add("ddddddddd");  
ListBox1.EndUpdate();
```

```
this.listBox1.Items.AddRange( new string []  
                                new string( 'a', 8 ),  
                                new string( 'c', 8 ),  
                                new string( 'd', 8 ),});
```

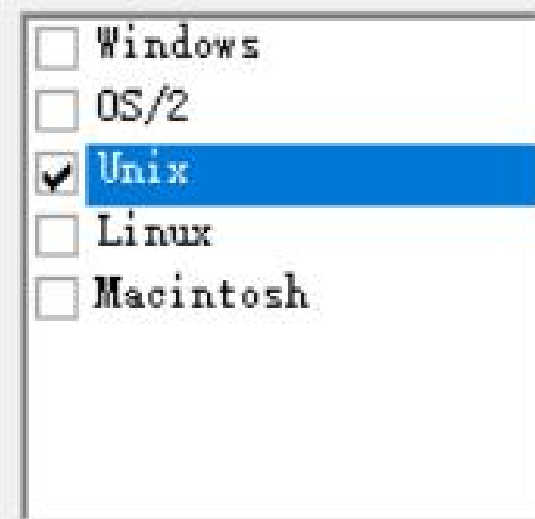




常用控件3：列表框

➤ CheckListBox 控件 CheckedListBox

- 增加了 CheckOnClick 属性
- GetItemChecked () , 选取指定条目时返回值为真。
- GetItemCheckedState () 指出一个条目的选取状态
- SetItemChecked () 设定条目的状态
- SetItemCheckedState () 设定条目的选取状态
- SetItemChecked () 返回值为真





常用控件3：列表框

➤ ComboBox 控件 ComboBox

- 列表框控件和编辑框控件的组合
- DropDownStyle属性决定其样式
 - ComboBoxStyle.DropDown:下拉式
 - ComboBoxStyle.Simple:简单样式
 - ComboBoxStyle.DropDownList:下拉列表式





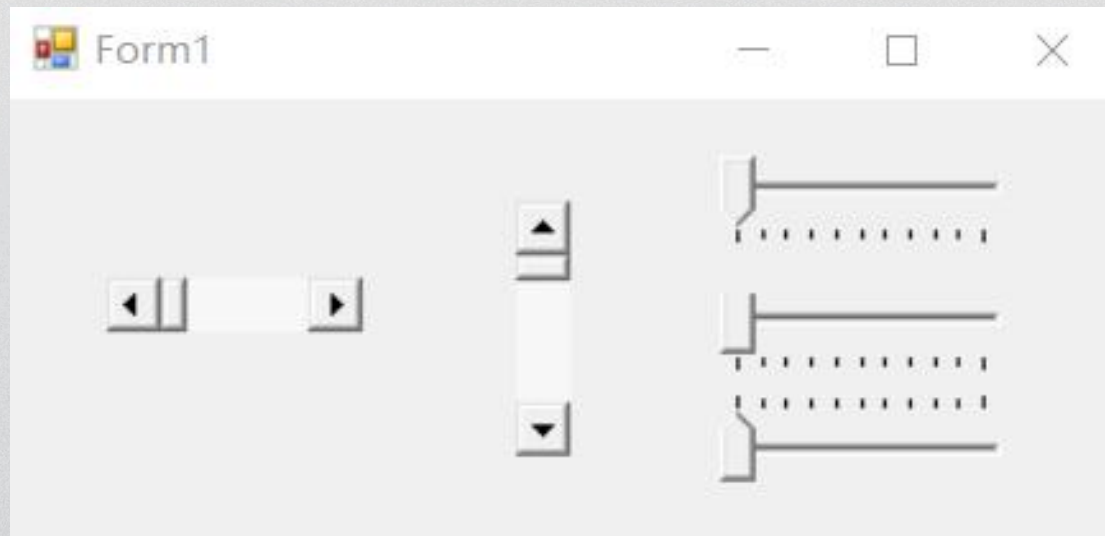
常用控件4：滚动条和进度条

➤ 滚动条

- 四个具有滑动能力的类
 - ScrollBar 滚动条基础类
 - HScrollBar 水平滚动条类
 - VScrollBar 垂直滚动条类
 - Trackbar 滑动条类

- 常用属性：

- Maximum(最大值) Minimum(最小值) Value(当前值)

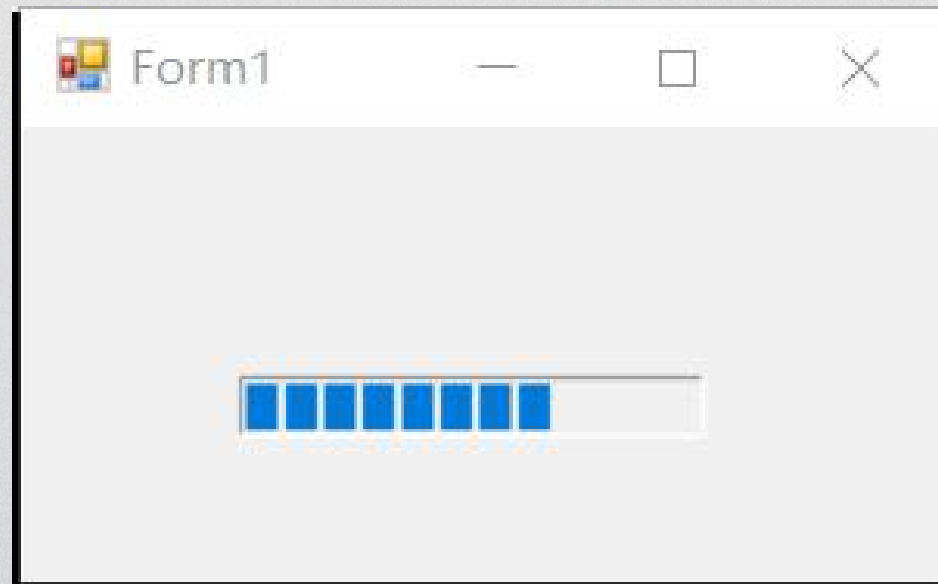




常用控件4：滚动条和进度条

➤ 进度条

- 由程序设定
- 常用属性
 - **Maximum(最大值)** : 默认值为100
 - **Minimum(最小值)** : 默认值为1
 - **Value (当前值)** : 可以用Increment () 和PerformStep () 改变value值





常用控件5：定时器，时间，日历

➤ 定时器（Timer）

- Interval属性以毫秒为单位设置时间间隔
- Start（）开始
- Stop（）停止
- Dispose（）资源释放四个具有滑动能力的类

```
this.timer1.Interval = 1000;
```

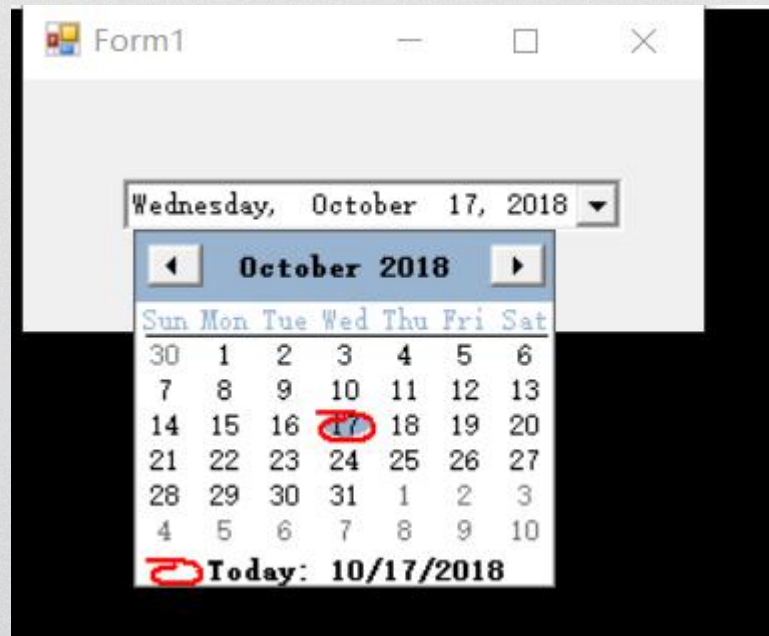
```
this.timer1.Start();
```




常用控件5：定时器，时间，日历

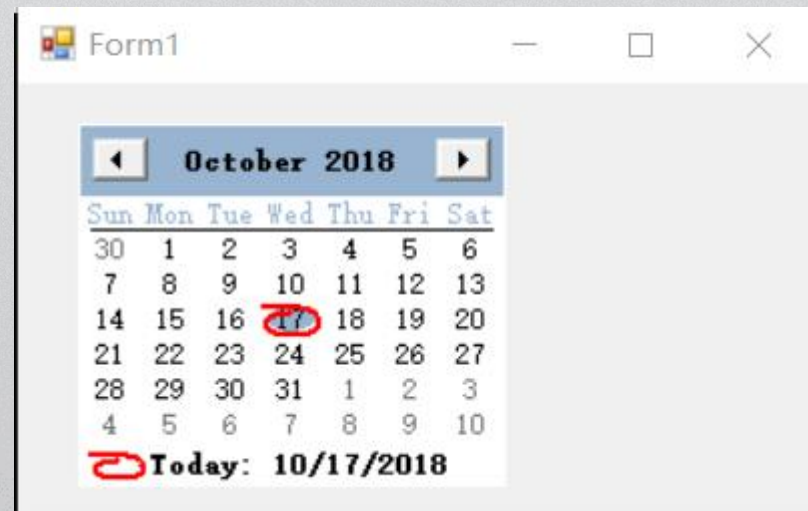
➤ DateTimePicker

- 选中日期
- 具有多种属性和格式



➤ MonthCalendar

- 对windows Calendar 控件的封装





常用控件6：图片框



PictureBox 图片框



PictureBox

- **SizeMode**属性：Normal,StretchImage, AutoSize,CenterImage,
- **Image**属性

```
this.pictureBox1 = new System.Windows.Forms.PictureBox();
```

```
this.pictureBox1.Location = new System.Drawing.Point(48, 16);
```

```
this.pictureBox1.Name = "pictureBox1";
```

```
this.pictureBox1.Size = new System.Drawing.Size(168, 96);
```

```
this.pictureBox1.TabIndex = 0;
```

```
this.pictureBox1.TabStop = false;
```

