事件

在控件的事件面板相应的事件后面双击创建事件处理函数 > 常用事件:click,leaded

事件处理函数

它是一个回调函数，默认参数sender和事件对象.sender指的是触发事件的控件对象 >在同一个解决方案里新建项目后需要右键新建的额项目选择设为启动项

控件

- 获取控件　直接使用控件的名称获取控件，`TB1.Text = "Hello"`(而不是非要用`FindName`)
- 控件的常用方法:Focus()
- 控件的通用常见属性:
    - Visibility
        - Visible　可见
        - Collapsed　不可见
    - IsEnabled
    - Background
    - FontSize
    - Content
    - HorizontalAlignment
    - VerticalAlignment
    - Margin
    - Width
    - Height
- TextBox
    - Text
    - Foreground　前景色
    - TextWrapping　自动换行
    - MaxLength

    > PasswordBox密码文本框的文本内容不是Text而是Password

- CheckBox
    - IsChecked　是否被选中　注意返回的是一个`bool?`类型(可空数据类型)，需要强转才能直接进行if判断(或者`==true`)
- RadioButton
    - GroupName
- DatePicker
    - SelectedDate　注意返回的是一个可空数据类型`DateTime?`

        > `DataTime.Today`今天(不含时刻)　`DataTime.Now`现在(日期+时刻)

- Image
    - Source　图片相对路径
- ProgressBar
    - Minimum

    - Maximum
    - Value
    - IsIndeterminate
- Menu
    - MenuItem
        - Header

布局

- StackPanel（堆叠面板）
    - Orientation 堆叠方向
        - Vertical
        - Horizontal
    - 代码示例

```
<Button>
    <StackPanel>
        <Image Source="1.jpg"></Image>
        <TextBlock Text="Hello!"></TextBlock>
    </StackPanel>
</Button>
```

- Grid（网格布局）
    - 代码示例

```
<Grid>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition></RowDefinition>
            <RowDefinition></RowDefinition>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Button Grid.Row="1" Grid.Column="1" Background="Blue"
Grid.ColumnSpan="2"></Button>
</Grid>
```

> Grid.Row为附加属性

- DockPanel（停靠面板）
    - DockPanel.Dock="Top"

> DockPanel.Dock为附加属性

- ToolBar

脚本

- 动态添加控件

```
Button b1 = new Button();
b1.Content = "Hello!";
sp1.RegisterName("b1",b1);
sp1.Children.Add(b1);
```

> 使用`b1.Name`设置的名字是不能使用`sp1.FindName`找到的

- 动态行定义，列定义

```
for(int i = 0; i < 10; i++)
        {
            RowDefinition rd = new RowDefinition();
            g1.RowDefinitions.Add(rd);
            ColumnDefinition cd = new ColumnDefinition();
            g1.ColumnDefinitions.Add(cd);
        }
```

- 多窗口
  - 新建一个窗口xaml，使用代码实例化显示该窗口

```
HelpWindow hw = new HelpWindow();
hw.ShowDialog();
```

> 主窗口向子窗口传值，都是通过属性，本质上就是类的互相调用
> 如果窗口使用`ShowDialog()`打开的，在打开的窗口中给DialogResult赋值会关闭窗口并将赋的值通过`ShowDialog()`的返回值返回

  - 在app.xmal中指定启动窗口
  - Window常用属性
    - Title
    - ResizeMode　设为`ResizeMode="NoResize"`后不能修改窗口大小
    - WindowStartupLocation　设为`WindowStartupLocation="CenterScreen"`窗口打开后显示在屏幕中央
    - WindowState　设为`WindowState="Maximized"`窗口打开时最大化
  - 模版窗口
    - OpenFileDialog（需要引入命名空间`using Microsoft.Win32;`）

```
OpenFileDialog ofd = new OpenFileDialog();
ofd.Filter = "文本文件|*.txt|图片|*.png|所有文件|*";
if (ofd.ShowDialog() == true)
```

```
{
    MessageBox.Show(ofd.FileName);
}
```

- SaveFileDialog

- 案例：选择图片显示

```
OpenFileDialog ofd = new OpenFileDialog();
if (ofd.ShowDialog() == true){
    bg1.Source = new BitmapImage(new Uri(ofd.FileName,
UriKind.Absolute));
}
```