C#系列 WinForm入门篇 /邵发

1.1 关于本篇

1

关于C#

应用场景:

- 桌面应用程序开发
  - 基于WinForms API
  - 基于WPF API
- Unity3D 游戏开发
- Asp.net 网站开发 (优先使用Java技术)

2

关于WinForms

WinForms : 基于Win32 API的C#封装

上手简单，控件丰富，可视化设计

在很短时间内掌握Windows桌面应用程序的开发

3

前置教程

前置教程: 《C#基础篇》

需要先对C#的基本语法和API有所了解

演示平台 Visual Studio 2019 ， 与基础篇里的配置相同

4

相关资源

PPT课件、QQ群、项目源码都可以官网下载

官网：http://afanihao.cn

源码的下载演示，见下一节课程。。

5

C#系列 WinForm入门篇 ../部发

2.1 第一个窗口

6

第一个窗口

打开 VS2019, 创建第一个窗口应用。。

项目类型:

C# | Windows | 桌面

Windows 窗体应用

参考《图文教程》文档

7

第一个窗口

运行程序，观察效果。。

8

可视化设计

可视化界面设计:

1 双击 Form1.cs，打开界面设计器

2 打开显示工具箱

3 拖一个按钮 Button 到界面上

再次运行程序，观察结果。。

9

C#系列 WinForm入门篇 /邵发

2.2 项目结构

10

项目结构

认识一下项目的结构:

App.config 应用配置

Form1.cs 源码文件 (窗口)

Form1.Designer.cs 源码文件 (界面设计)

Form1.resx 资源文件

Program.cs 源码文件 (Main方法)

11

项目结构

Form1 的两种打开方式

1 双击 Form1.cs ， 打开的是 界面设计器

2 右键，查看代码，打开的是 源码编辑器

结合使用：可视化界面设计 + 手工编写业务代码

12

项目结构

类的拆分：

Form1： Form1.cs + Form1.Designer.cs

关于 partial class 语法，参考《基础篇》 的4.4讲

13

C#系列 WinForm入门篇 /邵发

2.3 手工创建窗口

14

C#系列 WinForm入门篇 /邵发

3.1 添加控件

15

添加控件

演示：向窗口中添加一个按钮控件。。

1 打开显示工具箱

2 从工具箱中拖一个Button到窗口中

3 运行程序

观察：在 Form1.Designer.cs 中多了什么？

16

添加控件

Form1.cs : 业务代码

Form1.Designer.cs： 界面代码，自动生成

提示：Form1.Designer.cs 是设计器自动生成的，一般不要手工修改。

17

要点与细节

1 重点是它代码的调用关系

Form1()→ InitializeComponent()

18

C#系列 WinForm入门篇 /邵发

3.2 手动添加控件

19

手动添加控件

演示：手动创建一个控件，并添加到Form中。。

```
Button testButton = new Button();
this.Controls.Add(testButton);
```

20

手动添加控件

控件的位置和大小：

```
testButton.Location = new Point(40, 40);
testButton.Size = new Size(100, 40);
```

Location：相对于窗口左上角的 (x,y) 坐标

Size: 控件的宽度和高度，单位是像素

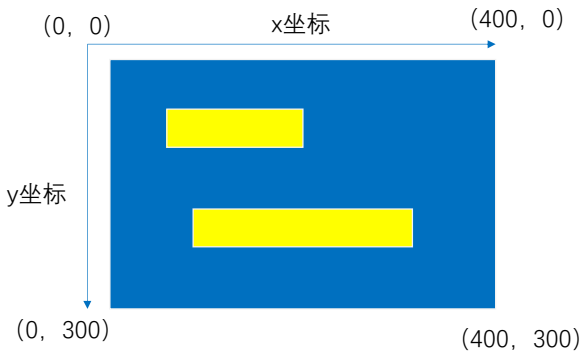
21

窗口坐标

横方向：x　竖直方向: y

宽度：width

高度：height



22

要点与细节

1 InitializeComponent() 界面初始化

在它之后添加自己的初始化代码

2 一般都是在界面设计器中添加，有时候需要手动添加控件

23

C#系列 WinForm入门篇 /邵发

4.1 事件处理

24

事件处理

演示：给按钮控件添加事件处理。。

1 右键点按钮，属性

2 切换到事件显示，Click事件

3 输入回调方法的名字，回车。。

则会自动生成一个用于事件处理的回调方法。。

25

事件处理

另外，在界面设计器上双击按钮时，会添加默认的事件处理。。

不过，默认的事件处理方法的名字不太好，不推荐。

26

要点与细节

1 事件处理回调是定义在 Form1.cs 中的

2 MessageBox.Show() 弹出一个消息框

3 在GUI程序中，控制台输出不起作用

不过，在调试状态下下Console.WriteLine() 还是可以看到打印输出

27

C#系列 WinForm入门篇 @邵发

4.2 手工事件处理

28

手工事件处理

演示：手工给按钮添加一个回调处理。。

1 在设计器里，给按钮起一个名字 (即字段名)

2 在Form1.cs里，添加一个回调方法

```
void OnTest(object sender, EventArgs e){}
```

3 添加事件处理

```
testButton.Click += new EventHandler(this.OnTest);
```

其中，委托/事件 的语法，参考 基础篇 的10、11章

29

手工事件处理

事件处理： System.EventHandler

delegate void EventHandler(object sender, EventArgs e)

其中，

sender : 事件发送者，即点中的控件

e : 事件的额外参数，比如鼠标点击的位置

30

自动 VS 手动

与上节课的源码对比：

自动方式：在 Form1.Designer.cs中自动生成代码

手动方式：在 Form1.cs 中手动添加事件处理

注：有的时候，需要手工方式添加事件处理。

31

要点与细节

1 Form1.Designer.cs 中的代码可以看，但不要修改

它是由设计器自动生成的。

2 如果不太理解 event 语法，也没关系，会写就行。

32

C#系列 WinForm入门篇 —邵发

4.3 (练习)显示时间

33

(练习)显示时间

练习：点击按钮，显示当前时间。。

1 添加一个Button，修改显示文本

2 添加一个TextBox ， 修改字段名 timeField

3 给按钮添加 Click事件处理。。。

34

要点与细节

1 Name 指的是字段名(变量名)

Text 指的是显示的文本

35

C#系列 WinForm入门篇 ../邵发

5.1 控件的布局

36

控件的布局

控件的布局：当窗口中有多个控件时，如何决定每个控件的位置和大小。

布局的方式：

1 可视化布局： 在设计器里拖放操作

2 手工布局： 用代码计算每个控件的位置

3 使用布局器： 用布局器自动布局

37

控件的布局

演示：添加几个控件，进行可视化布局 。。

按钮, Button

文本框, TextBox

图片框, PictureBox

38

控件的布局

本质上，是在窗口初始化的时候，使用代码设置了每个控件的位置和大小

例如，在 Form1.Designer.cs 中，

```
button1.Location = new Point(375, 12);
button1.Size = new Size(75, 23);
```

39

要点与细节

1 当窗口改变大小时，布局并不能够自动适应

所以，此种布局只适用于窗口大小固定不变的情况

40

C#系列 WinForm入门篇 /邵发

5.2 手工布局

41

手工布局

手工布局：用代码计算每个控件的位置

重写 OnLayout方法

```
override void OnLayout (LayoutEventArgs levent)
{
}
```

当窗口大小改变时，会自动调用这个方法重新布局

42

手工布局

演示：在窗口中添加几个控件，并实现手工布局。

其中，窗口的大小：

Size：窗口大小（含标题栏和边框）

ClientSize：仅窗口客户区的大小

43

要点与细节

1 TextBox： AutoSize = false

否则它会自己计算所需的大小，参考图文教程

2 区分窗口的 Size 和 ClientSize

3 窗口的自适应：当窗口大小变化时，会自动调用 OnLayout() 方法

思考：谁调用了这个方法？是系统框架自动调用的

44

C#系列 WinForm入门篇 /邵发

5.3 Anchor

45

Anchor

控件的两个通用的布局属性：

Anchor: 锚定，将控件固定于某个位置

Dock: 停靠，将控件停靠在一侧或中央

本节课介绍 Anchor 用法

46

Anchor

演示：设置一个控件的 Anchor 为 Top | Right

当窗口大小改变时，该控件锚定于窗口的右上角

即，与父窗口的上边距Top 和右边距 Right 保持不变

47

更多练习

1 锚定于左下角

- Anchor = Left | Bottom

2 锚定于右下角

- Anchor = Right | Bottom

48

更多练习

3 锚定于上边缘、水平拉伸

- Anchor = Top | Left | Right

4 锚定于上边缘、水平居中

- 水平居中
- Anchor = Top

49

更多练习

5 拉伸

- Anchor = Top | Right | Bottom | Left

6 居中

- 水平居中，垂直居中
- Anchor: None

50

C#系列 WinForm入门篇 /邵发

5.4 Dock

51

Dock

Dock：停靠，将控件停靠在一侧或中央

上 Top

下 Bottom

左 Left

右 Right

中 Fill

无 None

52

Dock

演示：Panel, 面板

1 添加一个Panel，停靠在上侧

2 添加一个Panel，依靠在左侧

3 添加一个PictureBox，依靠在中央

依靠于左右两侧时，可以调整宽度；上下两侧时，可以调整高度。

53

布局的嵌套

实际的项目中，界面布局可能有多个层次

比如，

一个Panel内部可能还会添加多个控件

54

要点与细节

1 当设置 Dock 属性时，Anchor属性无效

55

C#系列 WinForm入门篇 ——邵发

6.1 布局器

56

布局器

布局器 LayoutEngine：负责子控件的布局

默认地，一个Form 或 Panel 都自带了一个布局器

在窗口改变大小时，由窗口的布局器来负责调整布局

57

自定义的布局器

SimpleLayoutPanel： 自定义一个Panel，并自己实现一个LayoutEngine 。

观察代码。。

58

自定义的布局器

自定义布局器的 使用步骤:

1 工具|选项，Windows窗体设计器 | 常规

自动填充工具箱：设为True

2 添加自定义Panel或Control的类

3 生成解决方案 F7

4 重新打开Form1.cs，在工具箱界面可以看到自己的控件

59

小结

了解布局器 LayoutEngine 的作用

会使用自定义的布局器，知道怎么显示到工具箱中

注：不要求自己会写一个布局器

60

C#系列 WinForm入门篇 /邵发

6.2 FlowLayoutPanel

61

FlowLayoutPanel

FlowLayoutPanel， 流式布局

子控件依次排列，一行排满之后换行继续排。。

演示：。。

62

更多练习

练习:

1 布局的嵌套

Panel本身也是控件，也有Anchor/Dock属性

2 属性的设置

试着设一下 Padding 等属性

3 控件的选择

右键，选择控件或该控件所在的面板

63

要点与细节

1 注意 FlowLayoutPanel 本身不是布局器，它只是一个面板。是它内部实现了一个布局器。

64

C#系列 WinForm入门篇 /邵发

6.3 TableLayoutPanel

65

TableLayoutPanel

TableLayoutPanel， 表格布局

以表格的形式进行布局。。

演示：。。

66

TableLayoutPanel

练习：（以单行表格为例）

1 增加列、删除列

2 设置列的属性

列的宽度：

- 绝对值（固定大小）
- 百分比 （占据剩余空间的百分比)
- 自动大小（根据所需的空间自动分配)

67

TableLayoutPanel

演示：使用TableLayoutPanel 来布局界面

1 添加 TableLayoutPanel，停靠在上方

2 添加 Button，TextBox 到表格，设置列的宽度

3 设置 TextBox的Dock，填满单元格

4 添加 PictureBox，停靠在中央

68

要点与细节

1 表格中的控件也可以设置 Dock 属性

如何利用 Dock 属性，其规则是由布局器决定的

69

C#系列 WinForm入门篇 /邵发

6.4 AfDockLayout

70

AfDockLayout

至此，已经学了三种自带的布局器：

1 默认布局 (Anchor, Dock)

2 FlowLayoutPanel

3 TableLayoutPanel

对于初学练习来说，已经够用。但在项目中，还不够。

71

AfDockLayout

比如，默认的Dock停靠布局并不好用

演示对比：

第一种情况：先Left 后 Fill

第二种情况：先Fill 后 Left

最终的布局效果依赖于控件添加的顺序，非常不方便

72

AfDockLayout

AfDockLayout : 是对默认Dock布局的优化版

演示:

1 添加 Af.Winform.DockLayout.cs 到项目

2 重新生成项目

3 在工具箱里找到 AfDockLayout，添加到布局

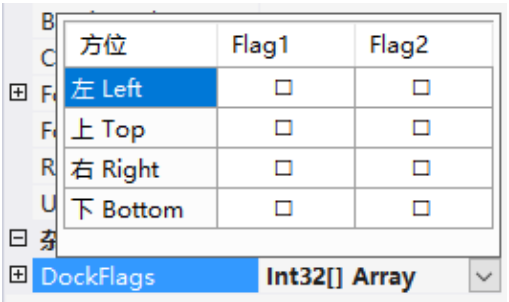
4 在面板中添加子控件，设置Dock

5 设置AfDockLayout.DockFlags属性

73

AfDockLayout

DockFlags : 决定四4个角由谁占据



74

AfDockLayout

AfDockLayout 的特点：

1 支持上、下、左、右、中位置，顺序无关

2 支持各个控件的 Margin

3 支持设置容器本身的Padding

75

C#系列 WinForm入门篇、邵发

7.1 常用控件

76

常用控件

本章介绍几个常用的控件

如 TextBox, CheckBox, ComoboBox 。。

列表、表格、树控件、图片框控件后面有介绍

77

常用控件

学习控件的使用：百度 + 官方文档

1 属性

- 行为：功能相关的属性
- 外观：显示相关的属性
- 杂项：该控件特有的属性

2 事件

78

C#系列 WinForm入门篇 /邵发

7.2 文本框

79

文本框

TextBox 文本框

用于输入单行或多行文本，常用单行输入模式

80

文本框

相关属性：

- 设计

  Name： 变量名

- 外观

  Text: 文本

  Font: 字体

81

文本框

相关属性：

- 行为

  Multiline： 单行模式 / 多行模式

  PasswordChar: 如果设置，则变成密码输入框

  ReadOnly： 只读模式

提示：不要全部试一遍，而是等需要的时候再来找

82

文本框

相关事件:

KeyPress : 按键事件， 常用于回车处理

83

C#系列 WinForm入门篇 /邵发

7.3 复选框

84

复选框

CheckBox 复选框

相关属性:

(外观) Text : 文本显示

(外观) Checked : 是/否

相关事件:

(操作) Click : 点击动作

(杂项) CheckedChanged : 选中状态发生变化

85

练习

练习：显示一个TextBox和一个CheckBox。

当选中时，以明文显示；取消选中时，以密码显示。

86

要点与细节

1 区分两种事件

Click : 用户手动点击

CheckedChanged ： 状态值发生变化，可能是用户点击，也可能是程序代码改变了这个值

例如，

```
checkBox1.Checked = true;
```

87

C#系列 WinForm入门篇 /邵发

7.4 下拉列表

88

下拉列表

ComboBox 下拉列表

1 添加数据项

- 在设计器里直接编辑

  属性 | 数据 | Items

- 也可以在构造方法里手工添加

  comboBox1.Items.Add("red");

89

下拉列表

2 获取选中的项

SelectedItem : 选中项的值, null表示未选中

SelectedIndex : 选中项的索引, -1表示未选中

3 事件

事件 | 行为 | SelectedIndexChanged

90

要点与细节

1 ComboBox 里的数据项可以是任意类型

可以是string，也可以是自定义的类型

参考《图文教程》文档

comboBox1.Items.Add(new Student(20200214,"邵发"));

91

C#系列 WinForm入门篇 /邵发

7.5 列表框

92

列表框

ListBox 列表框

两方面的功能:

- 展示： 展示一些项给用户看
- 选择： 让用户单选/或多选

93

列表框

属性:

(行为) SelectionMode : 单选 / 多选

事件:

(行为) SelectedIndexChanged

94

要点与细节

1 列表项数据可以是任意类型 object

本例中使用的是 Student 类型

实际在列表时，会调用 Student.ToString()来显示

2 界面显示优化，如行高、自定义显示

在下一篇：WinForm高级篇

95

C#系列 WinForm入门篇 /邵发

7.6 (练习)学生信息编辑

96

(练习)学生信息编辑

练习：实现一个学生信息的编辑器

- 学号, 姓名，性别，手机号
- 将数据保存到文件
- 启动时从文件读取

97

(练习)学生信息编辑

1 界面布局

- 添加需要的控件
- 修改显示文本 Text
- 手工对齐
- 修改控件名 Name

本例重点是业务逻辑，不使用布局器

98

(练习)学生信息编辑

2 保存功能

点保存时，将界面的数据保存到文件中

- 添加 NewtonSoft.json支持
- 添加按钮事件处理
- 将数据保存为JSON, 存到文件中

关于JSON的使用，参考 C#基础篇 的第18章

99

(练习)学生信息编辑

3 加载功能

当程序启动时，自动读取 student.txt中的数据

- 在构造方法中加载
- 读取文件，转成JSON
- 将数据显示到界面

作为练习，请课下自己完成。

100

C#系列 WinForm入门篇 /邵发

8.1 图片框

101

图片

Image : 抽象类，图像的统称

Bitmap: 具体类，位图，像素图

示例:

```
Bitmap img = new Bitmap("c:\\example\\123.jpg");
int w = img.Width;
int h = img.Height;
```

102

图片框

PictureBox 图片框控件：用于显示一个图片

设置缩放模式

```
picbox.SizeMode = PictureBoxSizeMode.Zoom;
```

显示图片

```
picbox.Image = img;
```

或者加载图片

```
picbox.Load("c:\\example\\123.jpg");
```

103

要点与细节

1 图片文件的路径，用windows路径（反斜杠）

"c:\\example\\123.jpg"

"c:/example/123.jpg" 不支持

104

C#系列 WinForm入门篇 《邵发

8.2 图片资源

105

图片的来源

Bitmap: 具体类，位图，像素图

图片的来源:

- 本地文件，如 "C:\\example\\123.jpg"
- 资源文件，本节课介绍
- 网络文件，如 http://afanihao.cn/img/java1.jpg

106

资源文件

资源文件:

Properties \ Resources.resx

可以添加字符串、位图、图标、音频等类型的资源

107

资源文件

添加资源:

1 双击 Resources.resx，打开资源编辑器

2 添加资源 | 添加现有文件

选择一张图片文件

3 修改资源名称

4 使用资源

```
Bitmap photo = Properties.Resources.Img_GeNie;
```

108

资源文件

在程序打包时，资源数据被打包到EXE程序中

对比：

1 添加一个图片(*.jpg)，生成项目，查看EXE大小

2 添加一个音频(*.wav)，生成项目，查看EXE大小

可以发现，EXE的体积显著增加。

思考：图片源文件删除了，会影响EXE的运行吗？

109

小结

了解WinForm里资源的概念

学会添加资源

学会在程序里使用资源对象

110

C#系列 WinForm入门篇、/邵发

9.1 复合控件

111

自定义控件

除了框架自带的标准控件外，还可以自定义控件

有三种方式：

1 复合控件：将标准控件组合起来

```
class YourControl : UserControl{}
```

2 扩展控件：继承于标准控件

```
class YourControl : Button {}
```

3 自定义控件：完全地自定义一个控件

```
class YourControl : Control {}
```

112

复合控件

复合控件：将标准控件组合起来，作为新的控件

展示：定义了一个搜索框控件

1 在工具箱里，显示 SearchBox 控件

2 将控件添加到窗口

3 设置控件的 Text 属性

4 添加事件处理：SearchEvent

113

复合控件

分析：

SearchBox 实际是由一个 TextBox 和一个 PictureBox 组合而成。它是一个复合控件。

- 自定义的属性
- 自定义的事件

114

要点与细节

1 工具|选项，Windows窗体设计器 | 常规

自动填充工具箱：设为True

115

C#系列 WinForm入门篇 /邵发

9.2 添加复合控件

116

添加复合控件

复合控件，一般也称为用户控件 UserControl

演示：添加一个复合控件。。

添加类 | 用户控件 (Windows 窗体)

117

在工具箱中显示

严格按以下步骤操作，才能显示：

1 工具|选项，Windows窗体设计器 | 常规

自动填充工具箱：设为True

2 添加自定义控件

3 生成解决方案 F7

4 重新打开Form1.cs，在工具箱界面可以看到自己的控件

118

修改复合控件

在窗口设计器中，更改复合控件的布局

参考《图文教程》文档

重新生成项目F7 | 重新打开Form1.cs 。。才能够在Form1中刷新显示

119

要点与细节

1 工具箱里显示的是当前项目中的自定义控件

项目中有的，它才会显示

2 自定义控件的刷新

修改自定义控件后，不会立即体现在Form1窗口

120

C#系列 WinForm入门篇 —邵发

9.3 使用复合控件

C#系列 WinForm入门篇 —邵发

9.4 自定义属性

自定义属性

在自定义控件时，可以添加一些属性，在设计器的属性面板里可以直接编辑。

演示：。。

123

自定义属性

属性可以添加一些Attribute限定

（相当于 Java里的注解语法）

例如：

```
[Browsable(true)]
[Category("Appearance")]
[DesignerSerializationVisibility(DesignerSerializationVisibility.Visible)]
```

124

自定义属性

属性也可以重写 。例如，可以重写 UserControl 的 Text 属性。。

```
public override string Text
{
}
```

125

C#系列 WinForm入门篇 —邵发

9.5 自定义事件

126

自定义事件

在自定义控件时，还可以添加自定义的事件..

自定义的事件会出现在事件面板里

演示：。。

参考《图文教程》文档

127

C#系列 WinForm入门篇 /邵发

10.1 控件的包装

128

控件的包装

控件的包装，就是把标准控件包装一层，是复合控件的一种特殊形式

比如，比较常见的是对 TextBox 的包装

```
public AfTextBox : UserControl
{
    public TextBox  edit;
}
```

129

控件的包装

TextBox不可调整高度、不可设置Padding，所以在布局时有点困扰。

演示：定义AfTextBox，实现一个可以调整高度的单行文本框。。

130

控件的包装

设计思路:

创建一个UserControl, 将TextBox包在里面， 显示的时候把TextBox定位在中央即可。

131

C#系列 WinForm入门篇 /邵发

10.2 AfTextBox

132

AfTextBox

演示：定义一个UserControl，实现对TextBox的包装

形如：

```
public AfTextBox : UserControl
{
}
```

133

AfTextBox

1 在设计器，拖一个 TextBox 进来

2 在代码里，手工对其布局（5.2讲）

```
override void OnLayout(LayoutEventArgs e)
{
}
```

134

AfTextBox

AfTextBox 的使用:

刷新项目，打开Form1.cs，将AfTestBox拖放到Form1中。。

可以看到，包装后的文本框高度是可调节的。

135

C#系列 WinForm入门篇 /邵发

10.3 单文件定义

136

单文件定义

默认的，一个UserControl 类分拆为两个CS文件

例如，

AfTextBox.cs

AfTextBox.Designer.cs

137

单文件定义

可以在一个单独的文件中定义，以方便重复使用

演示：

1 手工定义一个类 AfTextBox : UserControl

2 双击打开设计器，设计器会把界面代码写在同一个文件里

3 添加构造方法，调用 InitializeComponent

138

C#系列 WinForm入门篇——/部发

10.4 添加属性

139

添加属性

演示：给 AfTextBox 添加一些属性，使其和 TextBox 用法类似 。。

- Text 文本
- Font 字体
- BackColor 背景色
- ForeColor 前景色

140

C#系列 WinForm入门篇 /邵发

10.5 添加事件

141

添加事件

给 AfTextBox 添加一个回车事件 ReturnPressed

参考对照 9.5讲的过程

142

C#系列 WinForm入门篇 /邵发

11.1 对话框

143

对话框

对话框 Dialog，用于获取用户的输入

本章介绍对话框窗口的创建和使用

144

定义对话框

Form，代表一个窗口，普通窗口或对话框窗口

演示：

- 添加一个窗体
- 界面设计

和普通窗口没有什么不同。。

145

使用对话框

弹出对话框窗口：

```
MyDialog dlg = new MyDialog();
dlg.ShowDialog(this);
dlg.Dispose();
```

其中，

- dlg.Show() 作为普通窗口显示
- dlg.ShowDialog() 作为对话框窗口显示

146

思考

Form窗口对象含有非托管资源，需要手工销毁

但是：

为什么刚刚 ShowDialog()，就用 Dispose()把它销毁掉？

注：Dispose非托管资源，参考 基础篇的16.4讲

147

C#系列 WinForm入门篇，/邵发

11.2 对话框的阻塞

148

对话框的阻塞

阻塞 Blocked， 是对话框最重要的一个特性

示例：

```
MyDialog dlg = new MyDialog();
dlg.ShowDialog(this);
dlg.Dispose();
```

149

对话框的阻塞

演示：在 ShowDialog() 的前后添加一些打印

- 观察打印输出
- 单步执行、观察程序的阻塞

结论：只有当对话框被用户关闭以后，ShowDialog()才会返回，程序得以继续执行。。

150

使用对话框

阻塞的效果:

1 方法卡在ShowDialog 这一行，不往下执行

2 对话框窗口可以活动，后面的父窗口不可以活动

可以理解为，ShowDialog()内部有一个while循环，在一直等待窗口关闭事件。

151

C#系列 WinForm入门篇 //邵发

11.3 对话框的属性

152

对话框的属性

(外观) Text 窗口标题

(窗口样式) MaximizeBox 最大化按钮

(窗口样式) MinimizeBox 最小化按钮

(窗口样式) ShowInTaskbar 是否在任务栏显示

(布局) StartPosition 窗口显示位置

(外观) FormBorderStyle 边框设定/是否可以调整大小

153

要点与细节

1 普通窗口也具有这些属性，一样的设定

154

C#系列 WinForm入门篇，/邵发

11.4 对话框的返回

155

对话框的返回

本节课内容:

1 如何关闭对话框

2 如何取得对话框的输入

156

对话框的返回

当对话框关闭时，ShowDialog() 返回一个值

示例：

```
MyDialog dlg = new MyDialog();
DialogResult rc = dlg.ShowDialog();
dlg.Dispose();
```

怎么让对话框关闭呢？

157

对话框的返回

设置 Form.DialogResult 属性时，可以关闭对话框

示例：

```
void okButton_Click(object sender, EventArgs e)
{
    this.DialogResult = DialogResult.OK;
}
```

并且，这个 DialogResult 就是 ShowDialog() 的返回值

158

取得用户输入

先判断用户是点了‘确定’还是‘取消’

再从对话框中取得用户的输入值

```
DialogResult rc = dlg.ShowDialog();
if(rc == DialogResult.OK)
{
    string str = dlg.edit.Text;
    ……
}
```

159

小结

1 设置 DialogResult，有两层作用：

- 关闭对话框
- 设定返回值, 即 ShowDialog() 返回的值

160

C#系列 WinForm入门篇 /邵发

11.5 (练习)样式设定

161

(练习)样式设定

练习：创建一个样式设定对话框，用于设置字体和大小

演示：。。

162

(练习)样式设定

实现:

1 添加对话框 StyleDialog

- 添加控件，设置初始值
- 添加确定、取消按钮，设置 DialogResult
- 添加几个属性 FontFamily , FontSize

2 调用对话框

当用户点确定时，取得输入的值，后续处理。

163

C#系列 WinForm入门篇 /邵发

11.6 更多细节

164

更多细节

一、按钮的便捷设定

Button类有一个 DialogResult属性，可用于直接关闭当前对话框。

等效于手工添加回调、并设置 Form.DialogResult

165

更多细节

二、对话框的默认响应

AcceptButton: 当按回车键时触发

CancelButton: 当按ESC键时触发

另外，当点叉号关闭窗口时，相当于Cancel

166

C#系列 WinForm入门篇 -邵发

12.1 系统对话框

167

系统对话框

系统自带的一些对话框类：

- OpenFileDialog 打开文件对话框
- SaveFileDialog 保存文件对话框
- FolderBrowserDialog 目录选择对话框
- ColorDialog 颜色选择对话框
- FontDialog 字体选择对括框

在官方文档里，直接搜索查看示例即可

168

C#系列 WinForm入门篇 /邵发

12.2 (练习)图片查看器

169

(练习)图片查看器

练习：实现一个图片查看器

1 选择一个目录

2 列出该目录下所有的图片

*.jpg *.jpeg *.png

3 单击选择一项时，打开图片显示

170

C#系列 WinForm入门篇 /邵发

13.1 菜单栏

171

本章内容

本章内容:

- 菜单栏
- 工具栏
- 右键菜单

重点是右键菜单的实现。

172

菜单栏

菜单栏 MenuStrip， 支持可视化编辑

- 添加 MenuStrip
- 添加菜单、菜单项、分隔线
- 给菜单项设置属性

  Name 字段名，Text 文本显示，Image：图标
- 给菜单项添加事件处理（双击即可）

173

要点与细节

1 菜单栏不是本课程的重点，了解即可

https://www.iconfont.cn/

174

C#系列 WinForm入门篇 /邵发

13.2 工具栏

175

工具栏

工具栏 ToolStrip，主要用于显示工具按钮

演示：。。

176

要点与细节

1 工具栏的按钮代表一些常见的功能

工具按钮和菜单项是对应的，应选择同一个回调

177

C#系列 WinForm入门篇 ../邵发

13.3 右键菜单

178

右键菜单

右键菜单，ContextMenuStrip，即上下文菜单

例如，一个ListBox上右键点击时，显示上下文菜单

179

右键菜单

1 添加 ContextMenuStrip

在设计器里直接可视化编辑即可

2 给ListBox 添加鼠标事件 MouseUp

```
void listBox1_Mouse(sender, e)
```

3 弹出上下文菜单

```
contextMenuStrip1.Show(listBox1, e.Location);
```

180

右键菜单

4 区分上下文，作不同的处理

- 若点中了一项，则允许某些菜单 (修改/删除)
- 若点中任何项，则禁用某些菜单 (修改/删除)

181

要点与细节

1 根据鼠标点击的位置，判断点中了哪一项

```
int index = listBox1.IndexFromPoint(e.Location);
```

182

C#系列 WinForm入门篇 -邵发

14.1 列表控件

183

列表控件

列表控件 ListView

相当于 ListBox 的增强版，支持多列显示

最典型的例子：Windows的文件管理器的列表显示

184

列表控件

列表控件的几种视图:

Detail : 详情模式

List: 列表模式

LargeIcon : 大图标模式

SmallIcon : 小图标模式

185

列表控件

列表控件的几个特点:

- 显示模式可以切换
- 可以多字段显示
- 可以设置图标
- 标签可以编辑
- 每列可以单独排序

自定义绘制的技术，在WinForm高级篇讲解

186

列表控件

示例：添加一个ListView

设置显示模式

设置列标题

添加数据项

187

C#系列 WinForm入门篇 ../邵发

14.2 添加数据

188

添加数据

演示：用ListView将 C:\ 盘下的文件夹、文件列出..

对照代码讲解。。

189

添加数据

图标设置：

listView1.LargeImageList 用于大图标模式

listView1.SmallImageList 用于其他显示模式

在添加一项时，指定所使用的图片的索引

item = new ListViewItem(label, imageIndex)

190

添加数据

显示性能优化:

在批量添加数据时，为了避免界面频繁更新

```
listView1.BeginUpdate();
… 修改显示数据 …
listView1.EndUpdate();
```

191

C#系列 WinForm入门篇——/邵发

14.3 切换显示模式

192

切换显示模式

演示：添加右键菜单，实现4种显示模式的切换。。

- 设置大图标列表
- 添加右键菜单
  - 添加4个菜单项
  - 添加菜单项的点击处理
- 添加右键处理
  - 点击右键时，弹出菜单

193

要点与细节

1 显示菜单时，要根据当前的显示模式，设置各菜单项的选中状态

194

C#系列 WinForm入门篇 /邵发

14.4 列的排序

195

列的排序

演示：实现列表控件的排序功能。。

对照《图文教程 》进行

196

要点与细节

1 每个ListViewItem可以关联一个Tag对象

在WinForm里，Tag一般用于关联一个数据对象

197

C#系列 WinForm入门篇 —邵发

14.5 编辑标签

198

编辑标签

标签 Label：指每一个列表项的显示文本

演示：列表项的标签可以被直接编辑。。

199

编辑标签

步骤：

1 设置 LabelEdit 为 true，允许编辑

2 启动编辑

- 鼠标先选中、再点击即可启动编辑
- 也可以调用 ListViewItem.BeginEdit()启动编辑

200

编辑标签

3 编辑验证

有两个事件:

BeforeLabelEdit： 编辑前验证

AfterLabelEdit : 编辑之后验证

e.Item 表示编辑项的索引

e.Label 表示文本框中的值

e.CancelEdit 表示是否取消此次输入

201

要点与细节

1 当实现右键菜单方式时，需要记录当前点中的项

private ListViewItem mouseClickItem;

添加一个成员字段，在后续的菜单点击处理中使用

2 ListView控件只适合 Label 的编辑

如果想编辑所有字段，应该考虑使用表格控件

202

C#系列 WinForm入门篇 /邵发

15.1 表格视图

203

表格控件

表格视图 DataGridView ， 即表格控件

提行多行多列的表格状的数据展示

演示：以表格控件来展示学生数据。。

204

表格控件

基本操作:

1 添加一个表格控件 DataGridView

2 设置列数、列名

3 添加一行数据

显然，表格的每一个单元格都是可以编辑的。

205

要点与细节

1 指定一行数据 object[] row = ..

实际显示时，取对象的 ToString()进行显示

206

C#系列 WinForm入门篇 /邵发

15.2 表格的使用

207

表格的属性设置

几个基本的属性:

1 列设定 [杂项] Columns

2 列标题是否可见 [外观] ColumnHeadersVisible

3 行标题是否可见 [外观] RowHeadersVisible

4 允许用户添加 [行为] AllowUserToAddRows

208

表格的基础操作

1 增加一行数据

grid.Rows.Add()

2 获取所有行的数据

grid[col, row] ← 注意顺序

grid.Rows[i].Cells[j]

3 删除一行 grid.Rows.RemoveAt (i)

删除选中的行 grid.Rows.Remove( )

209

C#系列 WinForm入门篇 //邵发

15.3 单元格的编辑

210

单元格的编辑

单元格的编辑:

两种办法:

1 原位编辑，直接在表格里编辑

2 响应单元格的双击或右击，打开一个对话框

需自己定义一个对话框

211

单元格的编辑

直接编辑:

1 grid.Columns[0].ReadOnly = false;

ReadOnly为true时，此列不可直接编辑

2 启动编辑

选中该单元格，再单击之，则启动编辑

3 编辑后输入验证

当编辑后按回车，触发 [焦点]CellValidating 事件

212

更多练习

请自行完成，以对话框方式完成数据的修改

213

C#系列 WinForm入门篇 /邵发

15.4 单元格的自定义

214

单元格的自定义

DataGridView 的单元格是可以自定义的

演示：。。

215

单元格的自定义

单元格的自定义包含2个方面：

1 单元格的显示可以自定义

实现一个 DataGridViewCell

2 单元格的编辑器可以自定义

实现一个 IDataGridViewEditingControl

仅作了解，各种自定义技术都在 WinForm高级篇

216

C#系列 WinForm入门篇 /邵发

16.1 (实例)学生管理

217

(实例)学生管理

实例：实现一个学生信息管理的小程序

- 用表格显示学生记录
- 可以添加记录
- 可以编辑记录
- 可以删除记录
- 数据保存到JSON文件中

218

C#系列 WinForm入门篇 —/邵发

16.2 创建项目

219

创建项目

创建 WinForms 项目：

1 添加 DataGridView

2 编辑列 Columns

将最后一列的宽度设为 Fill

3 添加 Student.cs 类

添加几行数据，测试一下。。

220

创建项目

表格的属性设置:

1 ReadOnly : True不可编辑

2 AllowUserToAddRow : False 不允许用户添加行

3 SelectionMode : FullRowSelect 整行选择

4 表格的边框与背景色调整。。

221

C#系列 WinForm入门篇 —邵发

16.3 添加记录

222

添加记录

演示：右键菜单，点‘添加’，弹出对话框，添加一条记录。。

对照《图文教程》文档

223

要点与细节

1 GetCellAt（grid，location)

这是自己写的方法，根据鼠标的点击位置，判断点中了哪个的单元格。

224

C#系列 WinForm入门篇 —邵发

16.4 编辑记录

225

编辑记录

演示：右键菜单，点‘编辑’，弹出对话框，编辑一条现有的记录。

对照《图文教程》文档

226

要点与细节

1 StuEditDialog 既可用于新增，也可用于编辑

区别在于：编辑时需调用 InitValue()初始化

2 Tag的用途：给界面控件添加关联数据对象

227

C#系列 WinForm入门篇 —邵发

16.5 删除记录

228

删除记录

演示：选中一条记录，右键菜单，‘删除’。。

229

C#系列 WinForm入门篇 /邵发

16.6 数据的保存

230

删除记录

演示：用一个JSON文件来保存数据

student.dat

- 当新加记录时，保存到文件
- 当删除或更新时，同步到文件

231

删除记录

实现：

1 添加 NewtonSoft.Json 的支持

2 SaveData()，将学生数据保存到student.dat

3 LoadData()，从文件加载数据并显示到表格

关于JSON的使用，参考 C#基础篇 的18章

232

要点与细节

1 文件名的后缀是任意的

可以是 *.dat，也可以是 *.txt ， *.123 都行

2 真正的项目里，可能会用数据库来保存数据

233