**Winform入门基础**

类似记事本，电脑管理。这些都是界面应用程序

|  |
| --- |
| 注意： Winform开发的的窗体系统只能用于windows系统 |

Winform 上一个世纪框架

优点：

* 入门简单

缺点：

* 默认界面丑
* DPI(分辨率) 不同显示器可能会出现分辨率问题，但可以解决，不过比较极端

计算机只能进行二进制运算，显示器提供显示

假设显示器是1920 \* 1080 那么计算机存储的像素单元大小为：

1920\*1080\*3（3分别指R，G， B）

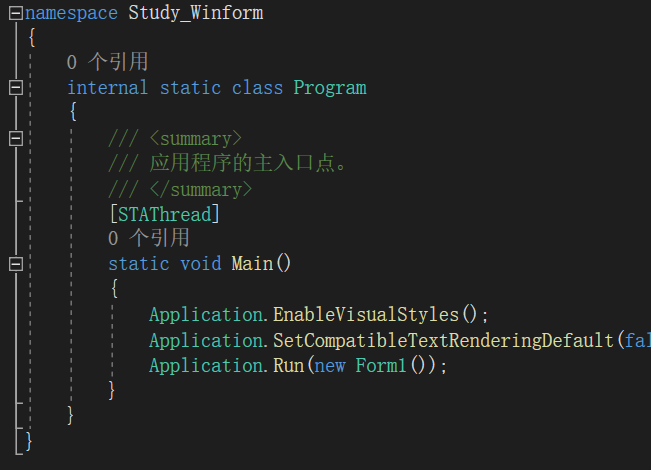
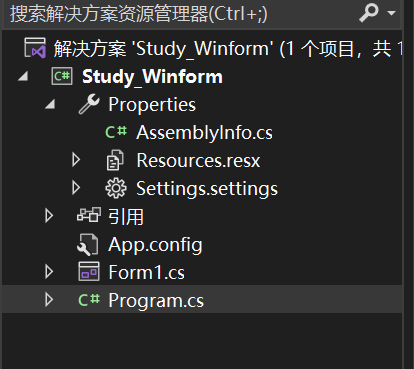
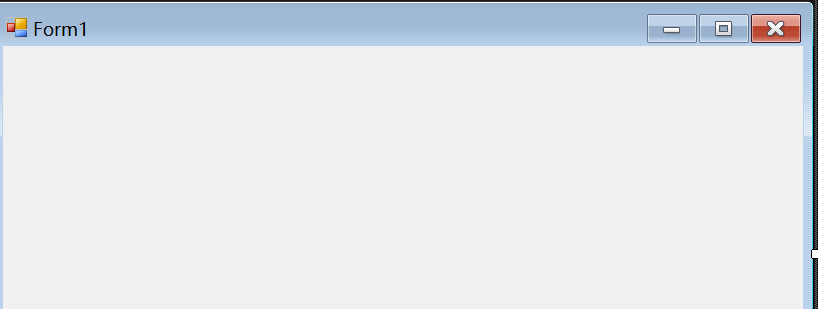
|  |
| --- |
|  |

与EazyX很像，整个窗口是基于屏幕进行计算，拖动就是不断擦除上一个位置的过程。

C#将Form(窗口)设计为一个类(Form)，实例化对象就是在创建这个窗口。

|  |
| --- |
| 准确的来说是类被映射到窗口上 |

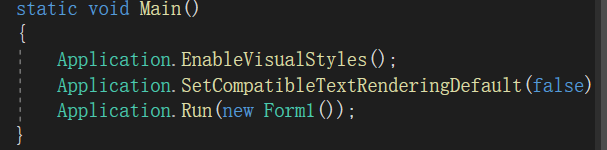
现在开始解析每一部分文件

* .csproj 项目管理文件，双击会打开当前项目
* Program.cs(**main方法在这里**) 项目运行的主要程序，程序会在这里运行
* Form1.cs **映射**到可视化界面上
* Form1(**partial 分部类,允许把类进行拆分**)又被分为两部分

1. Form1.cs（后台代码文件，右键窗口查看代码即可查阅到）

2. Form.Designer.cs（界面描述文件,作为Form1的子文件）

如果直接通过拖拽修改Form可视化窗口(改变**属性**也可)，Form.Designer会同步修改代码

* Form1继承Form,代表类会是一个窗口。
* 在Main方法中，

Application.Run(new Form1())等价于

|  |
| --- |
| Form1 form1 = new Form1(); //实例化一个窗口对象  Application.Run(form1); |

* 如何在后台创建一个新的组件(举例)

|  |
| --- |
| Button btn = new Button();  btn.Text = “你好”;  btn.left = 10; //位置距离左边10  btn.Top = 20; //位置距离顶部20  this.Controls.Add(btn); **//Controls是一个集合，用于存储所有的控件**  **//如果想知道控件有哪些属性，可以直接去工具箱中拖拽出来的控件的属性窗口查看。** |

* 如何前台添加控件(举例)

“视图”->“工具箱”，在“工具箱”中具有所有的windows窗体，可以直接拖拽到Form窗口上。

* 选中控件，直接双击，会在Form.cs中生成方法(Click事件方法)。

|  |
| --- |
| 注意：不建议修改Form.Designer.cs脚本，一般修改都是在Form.cs中修改。 |

例如:添加一个Button，双击生成事件方法

|  |
| --- |
| private void button1\_Click(**object sender**, **EventArgs e**)  {  //用户编辑  button1.Text = “123”;  } |

运行程序，原来的Button在生成出来的时候还是叫button1，当点击时候，就会变成123。

button1\_Click方法被谁调用了，当然是Form.Designer.cs，作为事件被存储。

* **object sender:** 表示触发事件的对象，例如点击按钮，触发事件对象就是button1。
* **EventArgs e :** 包含了与事件相关的信息。EventArgs是所有事件参数类的基类，它本身可能不包含太多信息，但不同的事件可能会使用它的派生类来传递特定的信息。

|  |
| --- |
| **Name的意义**：假如Button控件的属性Name不是button1，而是btnButton1，那么设计器中对象button1会变成对象btnButton1。  **Text**:该属性代表的是控件的称号，可以理解为人们常说的外号，他会在可视化界面显示出来。 |

那么同理，在Form1的窗口界面，点击控件，找到属性，能够查阅到有关控件的所有**属性**与**委托**，双击方法(雷电标识),编辑器会在Form1中添加对应委托的方法，这就是为什么双击控件会添加“点击事件”的方法。

至于各种委托的作用是什么，可以访问微软的官方文档进行查阅(也可以询问AI)

<https://learn.microsoft.com/zh-cn/dotnet/desktop/winforms/overview/?view=netframeworkdesktop-4.8&redirectedfrom=MSDN>

事件与委托(Event)

这里是C#的语法概念，欢迎访问微软的学习官网进行学习，这里不作讲解

设计后台生成控件（在Form1.cs的构造器中编写）

|  |
| --- |
| **Button button1 = new Button();**  public Form1()  {  InitializeComponent();    **button1.Location = new Point(10,10);** //控件坐标  **button1.Size = new Size(100,100);** //控件尺寸  **button1.Text = “我是代码写的”；**  //不添加到Controls集合中，无法在窗口中生成  **This.Controls.Add(button1);**  //添加点击事件方法，这里Button1\_Click方法是我按照之前给出的格式编写的  **Button1.Click += new System.EventHandler(Button1\_Click);**  } |

假设Button button1写在构造器内部，那么如何使事件添加成功

|  |
| --- |
| Private void Button1\_Click(object sender, EventArgs e)  {  //这里sender的作用就体现出来了，只要有点击按钮的操作，就会执行该命令  Button btn = (Button)sender;  btn.Text = “123”;  } |

登录界面设计

* MessageBox.Show(“我被点击了”); //显示一个弹窗



打开第二个界面

创建并打开第二个窗口的方法

|  |
| --- |
| Form2 form2 = new Form2(); //第一步： 创建实例  // 当打开第二个窗口，我们不使用Application.Run  // 而是通过创建实例，然后使用Show方法开启显示  Form2.Show(); //窗口显示  this.Hide(); //使用Hide隐藏显示  //这里不能用Close()方法，Close方法会关闭所有程序，当前程序被结束 |

|  |
| --- |
| 注意：当你关掉界面，程序还在运行。因为Application.Run(form1)中程序运行的是form1窗口，该界面没有关闭掉，那么程序将不会结束 |

这时候涉及到两个委托

* FormClosed 已经关闭执行的事件
* FormClosing 正在关闭执行的事件

如果想通过代码关闭进程

可以在Closing提供的方法中编写

|  |
| --- |
| Application.Exit(); //退出进程 |