Chapter 1 Introduction to Web Application Development Using ASP.NET

李莉

2017-9

实验 1：创建一个 ASP.NET Web 应用
实验 2：一个简单的货币换算器

实验 1：创建一个 ASP.NET Web 应用

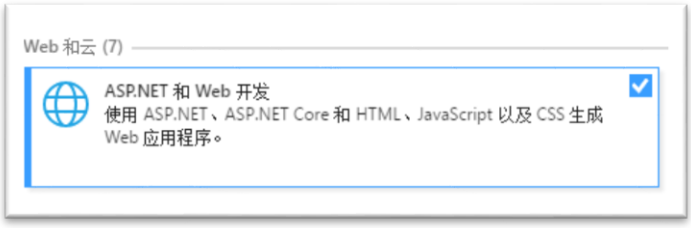
实验目的

演示 Web Form 和 MVC 应用的创建流程。
注：本实验指南使用 Visual Studio Community 2017，使用其他版本 Visual Studio，可能在选择项目模板上有些微区别，影响不大。

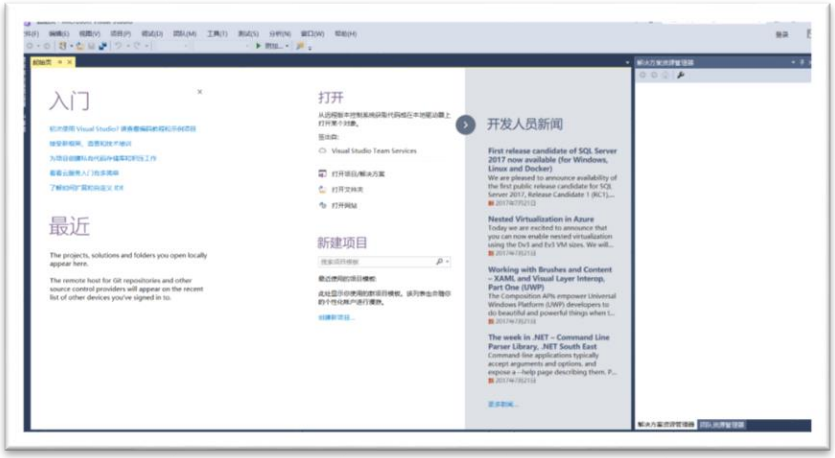
任务 1：安装 Visual Studio IDE

步骤一：访问 https://www.visualstudio.com，下载 Visual Studio Community Installer。

步骤二：运行 Visual Studio Community Installer，选择 ASP.NET 和 Web 开发。在右侧的边栏确保 ASP.NET MVC 勾选。确认后等待安装完成。



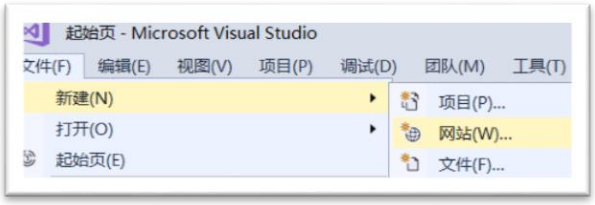
步骤三：启动 Visual Studio Community。



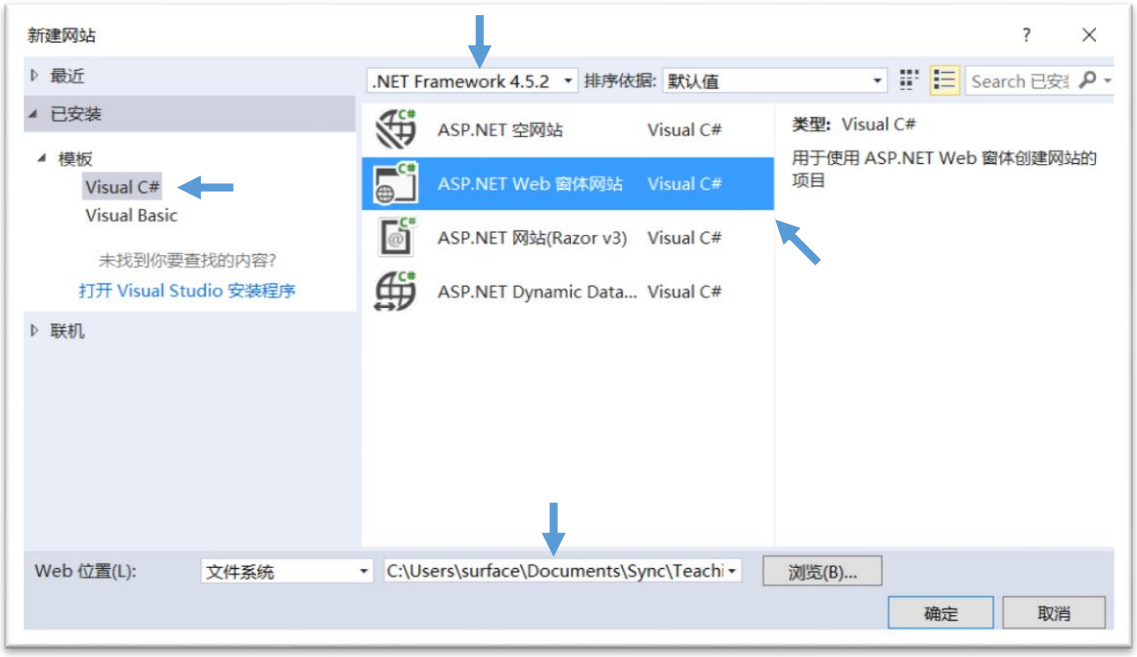
注 :如果是 Linux、MacOS 平台，可在步骤二选择安装跨平台的.NET Core 和 ASP.NET Core。

任务 2：创建 ASP.NET Web Form 应用

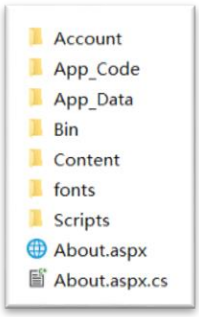
下面的实验步骤参考《Lab Guide》Task1.1~1.4。
步骤一：打开 Visual Studio，新建网站。



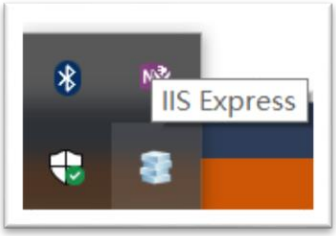
注 :也可以通过新建项目的方式创建应用，区别在于网站不会生成项目文件。详细区别可查看 MSDN 文档《Web Application Projects versus Web Site Projects in Visual Studio》
步骤二：选择语言、.NET 框架版本、项目模板、项目保存位置。

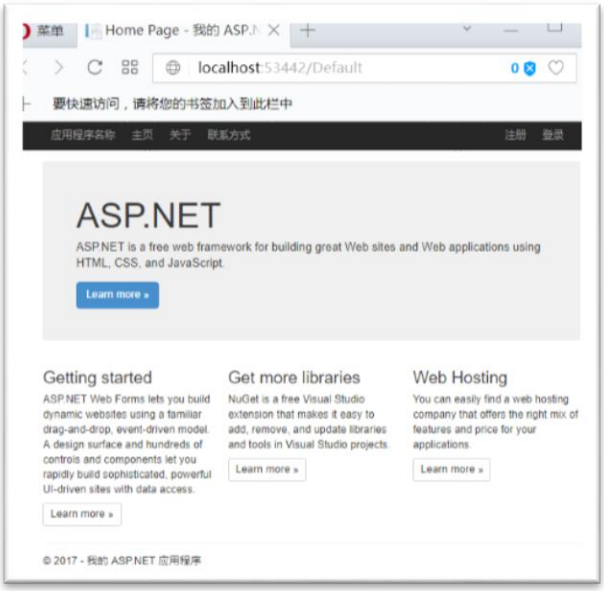


步骤三：打开网站文件夹，查看 Visual studio 自动生成的各种文件。



步骤四：运行网站。在 Visual studio 的 debug 按钮处选择浏览器类型。点击运行后，浏览器显示网页，任务栏可看到 IIS Express 运行中。





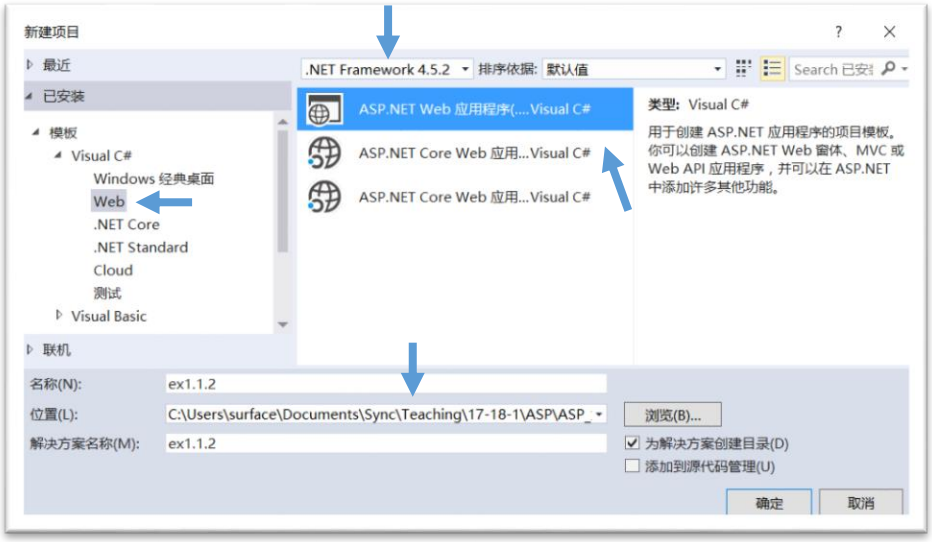
注 1：点击 Debug 按钮后，Visual Studio 启动 IIS Express Web 服务器，并将网站部署到服务器，最后启动浏览器向 Web 服务器发送请求。

注 2：开发 Web Form 应用，一般选择空模板。此处仅仅为了演示创建流程。

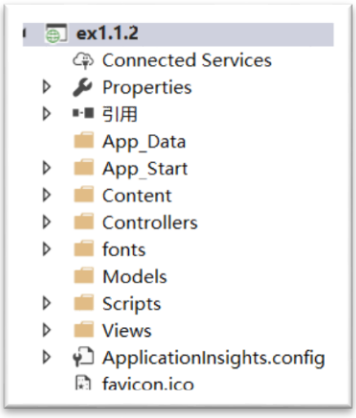
任务 3：创建 ASP.NET MVC 应用

下面的实验步骤参考《MVC Lab Guide》Task1.1~1.4。
步骤一：打开 Visual Studio，新建项目。

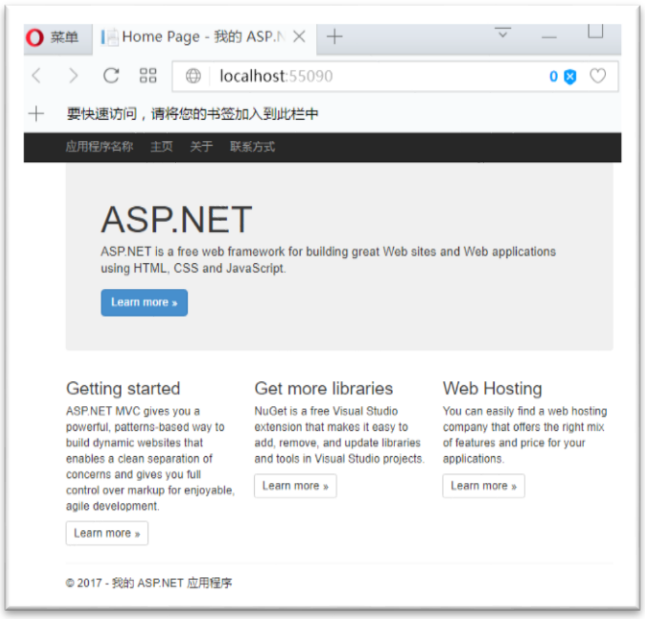
注：不要选择新建网站，没有 MVC 项目模板。
步骤二：选择语言、.NET 框架版本、项目模板、项目保存位置。在下一步对话框中选择 MVC 模板。



步骤三：在资源管理器，查看 Visual studio 自动生成的各种文件。



步骤四：运行网站。



实验总结

通过该实验，了解应用创建与调试的基本流程。

实验 2：一个简单的货币换算器

实验目的

理解 Postback 和页面生命周期。

问题描述

实现一个货币换算器，能够实现美元到多种货币的换算。用户输入美元数目，选择目标币种并确认后，网页显示换算结果。

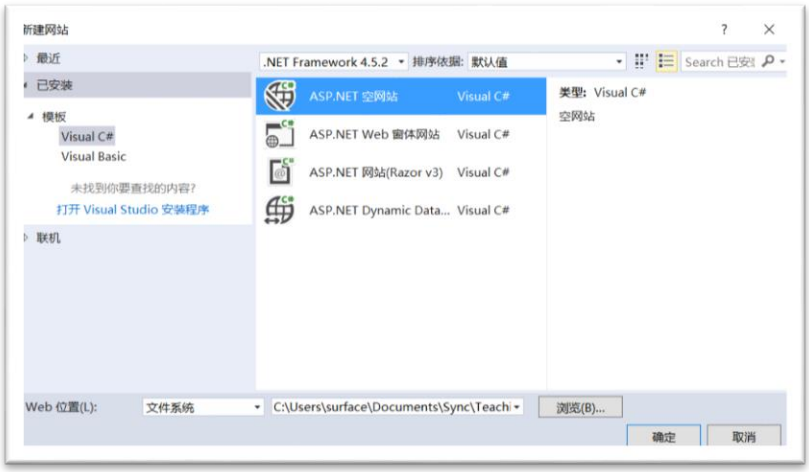
需在页面显示该处理过程触发的各种页面事件。

问题分析

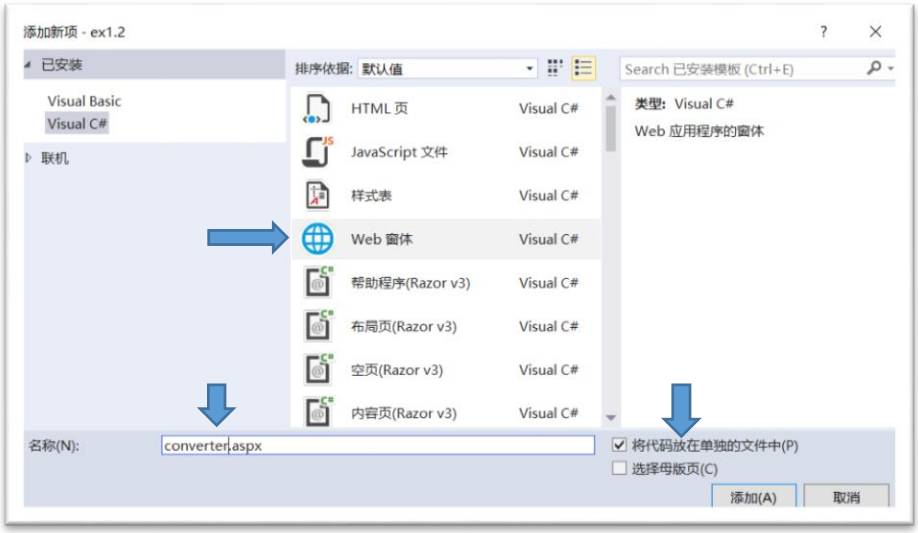
1. 通过一个 Web form 实现该应用。用户输入可以使用各种输入控件实现。
2. 为捕获页面事件，可以编写对应的事件处理函数。

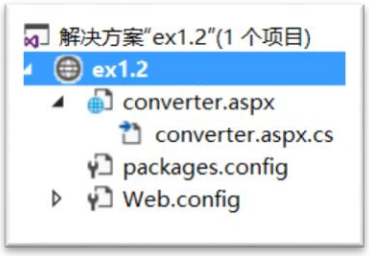
解决方案

步骤一：打开 Visual Studio，新建一个 Web Form 网站。模板选择空网站。



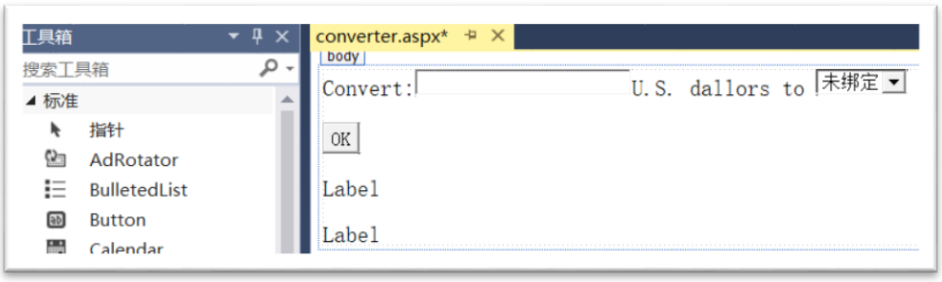
步骤二：通过资源管理器添加新项。模板选择 Web 窗体，命名为 converter.aspx。注意右下方勾选 code-behind 代码模型。添加成功后，在资源管理器可看到新建的 Web 窗体。





注：由于选择 code-behind 代码模型，该页面由两个文件组成，一个是保存标记代码的 converter.aspx，另一个是保存 C#代码的 converter.aspx.cs。

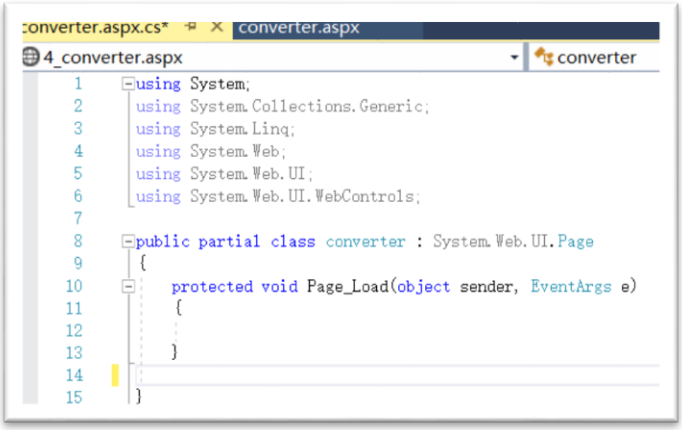
步骤三：在 Visual Studio 的设计窗口打开 converter 页面（新建后会自动打开）。设计器提供三种视图：源视图、设计视图、拆分视图。切换到设计视图，打开左侧工具箱，从标准控件栏拖放一些控件到页面。Textbox 用于输入美元数目，下拉列表用于选择目标币种。按钮用于确认。两个标签，一个用于显示换算结果，一个用于显示触发的页面事件。控件之外的文本，直接在设计视图输入。



步骤四 :切换到源视图,查看刚才拖放操作生成的页面源码。通过属性窗口可修改控件属性,例如文本等。

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="converter.aspx.cs"
Inherits="converter" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        Convert:<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        U.S. dallors to
        <asp:DropDownList ID="DropDownList1" runat="server">
        </asp:DropDownList>
        <br />
        <br />
        <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Text="OK" />
        <br />
        <br />
        <asp:Label ID="Label1" runat="server"></asp:Label>
        <br />
        <br />
        <asp:Label ID="Label2" runat="server"></asp:Label>

    </div>
    </form>
</body>
</html>
```

步骤五：在设计器的任何位置点击右键，选择查看代码。C#代码文件编辑窗口打开。



步骤六：在 Page_Load 事件处理中添加以下代码。表示在页面加载时动态绑定下拉列表框的数据。注意判断条件，这意味着下拉列表框只有在第一次请求时才需要绑定数据。

```
protected void Page_Load(object sender, EventArgs e)
{
    if (this.IsPostBack == false)
    {
        DropDownList1.Items.Add(new ListItem("Euros", "0.85"));
        DropDownList1.Items.Add(new ListItem("Japanese Yen", "110.33"));
        DropDownList1.Items.Add(new ListItem("Canadian Dollars", "1.2"));
    }
}
```

步骤七：切换到窗体文件，在设计视图双击按钮，生成点击事件处理。在事件处理函数中实现换算。

```
protected void Button1_Click(object sender, EventArgs e)
{
    decimal oldAmount;
    bool success = Decimal.TryParse(TextBox1.Text, out oldAmount);
    if (success)
    {
        ListItem item = DropDownList1.Items[DropDownList1.SelectedIndex];
        decimal newAmount = oldAmount * Decimal.Parse(item.Value);
        Label1.Text = oldAmount.ToString() + " U.S. dollars = ";
        Label1.Text += newAmount.ToString() + " " + item.Text;
    }
}
```

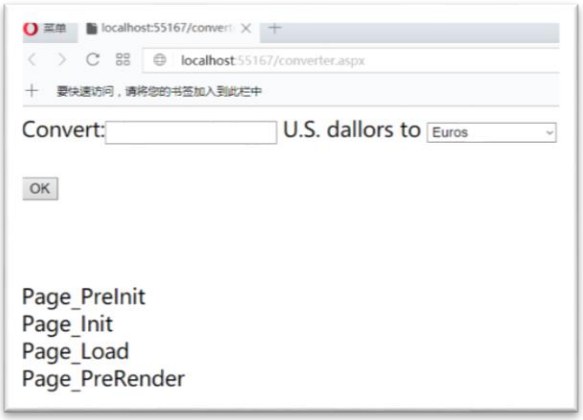
步骤八：切换到代码文件，添加感兴趣的页面和控件事件的处理函数。当该事件触发时，在页面的第二个标签上显示一行记录。

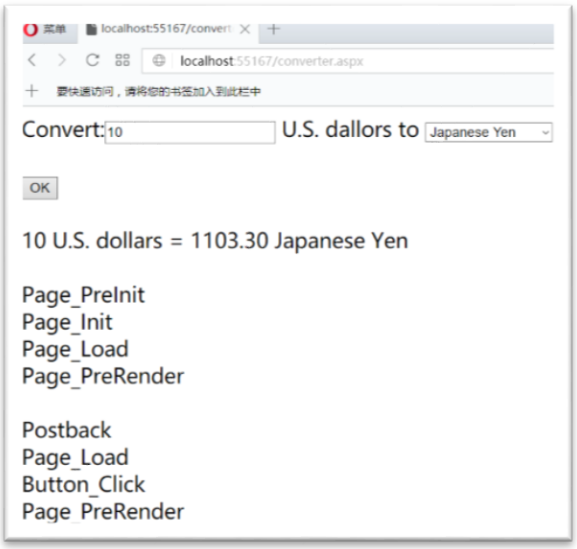
```
public partial class converter : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
      if (this.IsPostBack == false)
```

```
        {
            DropDownList1.Items.Add(new ListItem("Euros", "0.85"));
            DropDownList1.Items.Add(new ListItem("Japanese Yen", "110.33"));
            DropDownList1.Items.Add(new ListItem("Canadian Dollars",
"1.2"));
        }
        else
            Log("<br>Postback");
        Log("Page_Load");
    }


    protected void Button1_Click(object sender, EventArgs e)
    {
        decimal oldAmount;
        bool success = Decimal.TryParse(TextBox1.Text, out oldAmount);
        if (success)
        {
            // Retrieve the selected ListItem object by its index number.
            ListItem item =
DropDownList1.Items[DropDownList1.SelectedIndex];
            decimal newAmount = oldAmount * Decimal.Parse(item.Value);
            Label1.Text = oldAmount.ToString() + " U.S. dollars = ";
            Label1.Text += newAmount.ToString() + " " + item.Text;
        }
        Log("Button_Click");
    }
    protected void Page_PreInit(object sender, EventArgs e)
    {
        Log("Page_PreInit");
    }
    protected void Page_Init(object sender, EventArgs e)
    {
        Log("Page_Init");
    }
    protected void Page_PreRender(object sender, EventArgs e)
    {
        Log("Page_PreRender");
    }
    protected void Page_UnLoad(object sender, EventArgs e)
    {
        Log("Page_UnLoad");
    }
    private void Log(string entry)
```

```
    {
        Label2.Text += entry + "<br>";
    }
}
```

步骤九：运行页面，第一张图首次访问的页面。第二张图是点击 OK 按钮后的页面





实验总结

通过该实验理解用户的一些操作，例如点击按钮会触发回传。无论是首次请求还是回传，服务器端都会重复页面的生命周期。

思考题

1. 为什么在首次请求和回传中，Page_Unload 没有出现在打印信息中？难道 Page_Unload 事件未触发？
2. 为什么在回传中没有打印 Page_PreInt 和 Page_Int，难道这两个事件未触发？

提示：建议仔细阅读 MSDN 文档《ASP.NET Page Life Cycle Overview》https://msdn.microsoft.com/en-us/library/ms178472.aspx，理解页面生命周期各阶段的含义。