第5章 ASP.NET内置对象

本章学习目标

- 了解内置对象的基本原理
- 熟练掌握 Page 对象的使用方法
- 熟练掌握 Response 对象的使用方法
- 熟练掌握 Request 对象的使用方法
- 熟练掌握 Server 对象的使用方法
- 熟练掌握 Application 对象的使用方法
- 熟练掌握 Session 对象的使用方法
- 了解 Cookie 对象的使用方法
- 熟练掌握全局应用程序类中的事件

本章首先介绍内置对象的作用，然后对 ASP.NET 提供的 Page、Request、Response、Application、Session、Server、Cookie 等内置对象进行详细讲解，最后添加全局应用程序类，并对其中的事件进行应用测试。

5.1 Page 对象

5.1.1 Page 对象的属性和方法

扫一扫

Page 对象是 System.Web.UI 命名空间 Page 类的一个实例，是网页中所有服务器控件的容器，在 ASP.NET 中每个页面都派生自 Page 类，并继承该类所有公开的方法和属性。Page 对象的主要属性如表 5.1 所示，主要事件如表 5.2 所示。

表 5.1 Page 对象的主要属性

属 性 名	说 明
IsPostBack	获取一个值，指示页面是首次加载还是回发加载
IsValid	获取一个值，指示页面验证是否成功

说明：

1. IsPostBack 属性可以检查.aspx 页是否为“回发”，常用于判断页面是否为首次加载，若为首次加载，则该值为 False。

2. IsValid 属性可以判断页面中输入的所有内容是否通过验证，使用服务器端验证时常使用该属性。

表 5.2 Page 对象的主要事件

事 件 名	说 明
PreInit	当页面初始化时触发
PreLoad	在页面 Load 事件之前触发
Load	当服务器加载到 Page 对象时触发
Init	当服务器控件初始化时触发
PreRender	在加载控件之后、呈现控件之前触发
Unload	当服务器控件从内存中卸载时触发
InitComplete	当页初始化完成时触发
LoadComplete	当页加载结束时触发

5.1.2 Page 对象的应用

网页 C#代码中默认提供了 Page.Load 事件所对应的 Page_Load()方法，而其他的事件则需要编写其对应的方法，不需要如控件一样在源中声明。各事件触发执行的先后顺序为：Page. PreInit、Page. Init、Page. InitComplete、Page. PreLoad、Page. Load、Page. LoadComplete、Page.PreRender 和 Page.Unload。

【示例 5-1】 在 E 盘 ASP.NET 项目代码目录中创建 chapter5 子目录，将其作为网站根目录，创建名为 example5-1 的网页，演示加载页面时 Page 对象的各种事件的触发执行顺序。

(1) 在页面中添加相应控件，具体如图 5.1 所示。

图 5.1 Page 对象应用页面设计

(2) 设置相关控件的属性，如下列源代码所示。

```
<form id="form1" runat="server">
    <span>Page 对象事件的触发顺序</span><hr>
    <asp:Label ID="lblInfo" runat="server"></asp:Label>
    <br/>
    <asp:Button ID="btnSubmit" runat="server" OnClick="btnSubmit_Click" Text="提
交"/>
</form>
```

（3）为事件添加 C＃代码如下。

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        lblInfo.Text+="页面第一次加载";
    }
    else
    {
        lblInfo.Text+="页面第二次或第二次以上加载";
    }
    lblInfo.Text+="触发了 Page 对象的 Load 事件<br>";
}
protected void Page_Init(object sender, EventArgs e)
{
    lblInfo.Text+="触发了 Page 对象的 Init 事件<br>";
}
protected void Page_PreInit(object sender, EventArgs e)
{
    lblInfo.Text+="触发了 Page 对象的 PreInit 事件<br>";
}
protected void Page_InitComplete(object sender, EventArgs e)
{
    lblInfo.Text+="触发了 Page 对象的 InitComplete 事件<br>";
}
protected void Page_PreLoad(object sender, EventArgs e)
{
    lblInfo.Text+="触发了 Page 对象的 PreLoad 事件<br>";
}
protected void Page_LoadComplete(object sender, EventArgs e)
{
    lblInfo.Text+="触发了 Page 对象的 LoadComplete 事件<br>";
}
protected void Page_PreRender(object sender, EventArgs e)
{
    lblInfo.Text+="触发了 Page 对象的 PreRender 事件<br>";
}
protected void Page_Unload(object sender, EventArgs e)
{
    lblInfo.Text+="触发了 Page 对象的 Unload 事件<br>";
}
protected void btnSubmit_Click(object sender, EventArgs e)
{
```

```
        lblInfo.Text+="触发了按钮的 Click 事件<br>";
    }
```

（4）运行页面，页面显示相关事件执行顺序，如图 5.2 所示。单击“提交”按钮，页面显示“回发”时相关事件执行的顺序，如图 5.3 所示。

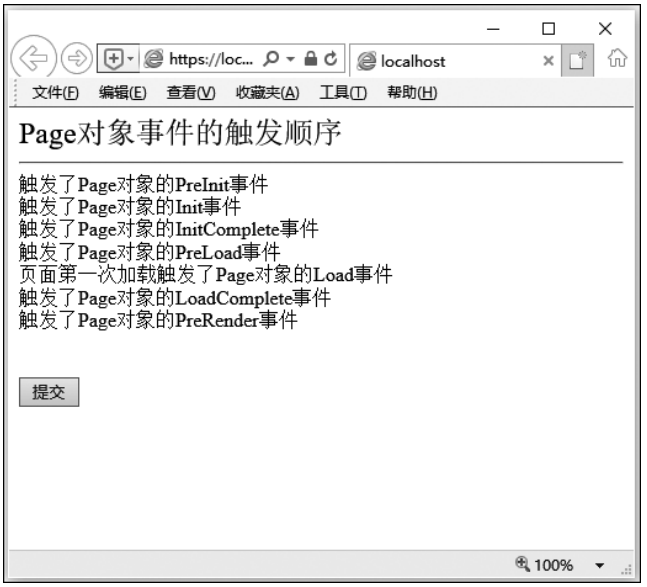

图 5.2　页面初次加载 Page 对象事件演示

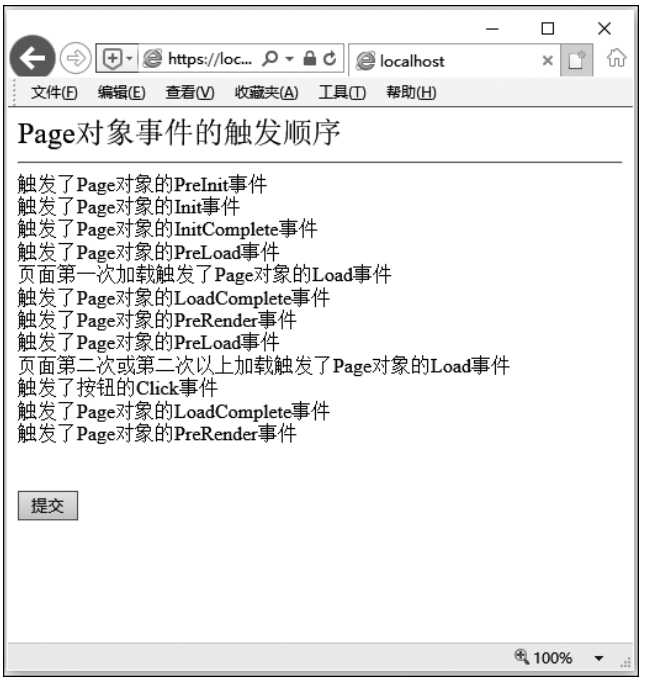

图 5.3　页面“回发”时 Page 对象事件演示

5.2 Response 对象


扫一扫

5.2.1 Response 对象的属性和方法

Response 对象是 HttpResponse 类的一个实例，封装来自 ASP.NET 操作的 HTTP 响应信息，提供对当前页的输出流访问，控制服务器发送给浏览器的信息，包括直接发送信息给浏览器、重新定向浏览器到另一个 URL，以及设置 Cookie 值等。Response 对象的主要属性如表 5.3 所示，主要方法如表 5.4 所示。

表 5.3 Response 对象的主要属性

属 性 名	说 明
Buffer	获取或设置一个值，该值指示是否缓冲输出，并在完成处理整个响应之后将其发送
BufferOutput	获取或设置一个值，该值指示是否缓冲输出，并在完成处理整个页之后将其发送
Cache	获取 Web 页的缓存策略，如过期时间、保密性、变化子句等
Charset	获取或设置输出流的 HTTP 字符集
Cookie	获取响应的 Cookie 集合
Expires	获取或设置在浏览器上缓存的页过期前的分钟数

表 5.4 Response 对象的主要方法

方 法 名	说 明
AppendCookie()	向响应对象的 Cookie 集合中增加一个 Cookie 对象
Clear()	清空缓冲区中的所有内容输出
Close()	关闭当前服务器到客户端的连接
End()	终止响应，并且将缓冲区中的输出发送到客户端
Redirect()	重定向当前请求
Write()	将信息写入 HTTP 的响应输出流
WriteFile()	将指定的文件直接写入 HTTP 的响应输出流

5.2.2 Response 对象的应用

【示例 5-2】 在 E 盘 ASP.NET 项目代码的 chapter5 目录下，添加一个文本文件，命名为“1.txt”，编写文本内容为“This is a text”，创建名为 example5-2 的网页，练习使用 Response 对象的各种属性和方法。

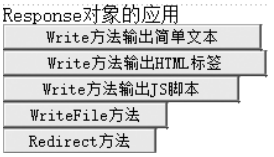

图 5.4 Response 对象应用页面设计

(1) 在页面中添加相应控件，具体如图 5.4 所示。

(2) 设置相关控件的属性，如下列源代码所示。

```
<form id="form1" runat="server">
    <div>
        Response 对象的应用<br/>
        <asp:Button ID="btnWriteTxt" runat="server"  Text="Write 方法输出简单文
本" OnClick="btnWriteTxt_Click"/>
        <br/>
        <asp:Button ID="btnWriteHTML" runat="server"  Text="Write 方法输出 HTML
标签" OnClick="btnWriteHTML_Click"/>
        <br/>
        <asp:Button ID="btnWriteJS" runat="server"  Text="Write 方法输出 JS 脚
本" OnClick="btnWriteJS_Click"/>
        <br/>
        <asp:Button ID="btnWriteFile" runat="server"  Text="WriteFile 方法"
OnClick="btnWriteFile_Click"/>
        <br/>
        <asp:Button ID="btnRedirect" runat="server"  Text="Redirect 方法"
OnClick="btnRedirect_Click"/>
      </div>
    </form>
```

(3) 为事件添加 C# 代码如下。

```
protected void btnWriteTxt_Click(object sender, EventArgs e)
{
    Response.Write("普通文字输出");
}
protected void btnWriteHTML_Click(object sender, EventArgs e)
{
    Response.Write("<a href=123>包含 html 标签的文本</a>");
}
protected void btnWriteJS_Click(object sender, EventArgs e)
{
    Response.Write("<script>alert('包含脚本文本');</script>");
}
protected void btnWriteFile_Click(object sender, EventArgs e)
{
    Response.WriteFile(@"E:\ASP.NET 项目代码\chapter51.txt");
}
protected void btnRedirect_Click(object sender, EventArgs e)
{
    Response.Redirect("example5-1.aspx");
}
```

(4) 运行页面，单击按钮，可获取 Response 对象各方法和属性的基本功能，单击“WriteFile 方法”按钮，运行时效果如图 5.5 所示。

图 5.5 Response 对象应用页面演示

5.3 Request 对象


扫一扫

5.3.1 Request 对象的属性和方法

Request 对象是 System.Web.HttpRequest 类的实例，当客户请求 ASP.NET 页面时，所有的请求信息，包括请求报头、请求方法、客户端基本信息等被封装在 Request 对象中，利用 Request 对象可以读取这些请求信息。Request 对象的主要属性如表 5.5 所示，主要方法如表 5.6 所示。

表 5.5 Request 对象的主要属性

属 性 名	说 明
Browser	获取正在请求的客户端浏览器功能的信息
Cookies	获取客户端发送的 Cookie 的集合
Form	获取表单变量的集合
FilePath	获取当前请求的虚拟路径
Param	获取地址栏中的参数集合
QueryString	获取 HTTP 查询字符串变量集合
UserHostAddress	获取远程客户端的 IP 主机地址
Url	获取有关当前请求的 URL 信息
UserHostName	获取远程客户端的 DNS 名称

表 5.6 Request 对象的主要方法

方 法 名	说 明
BinaryRead()	执行对当前输入流进行指定字节数的二进制读取
MapPath()	将请求的 URL 中的虚拟路径映射到服务器上的物理路径
SaveAs()	将 HTTP 请求保存到文件中

5.3.2 Request 对象的应用

【示例 5-3】 在 E 盘 ASP.NET 项目代码的 chapter5 目录下，创建名为 example5-3 的网页，练习使用 Request 对象的各种属性。

(1) 在页面中添加相应控件，具体如图 5.6 所示。

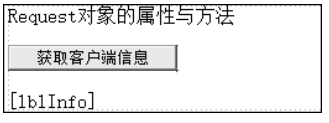

图 5.6 Request 对象应用页面设计

（2）设置相关控件的属性，如下列源代码所示。

```
<form id="form1" runat="server">
    Request 对象的属性与方法<br/>
    <br/>
     <asp:Button ID="btnGetInfo" runat="server" OnClick="btnGetInfo_Click"
Text="获取客户端信息"/>
    <br/>
    <asp:Label ID="lblInfo" runat="server"></asp:Label>
</form>
```

（3）为事件添加 C# 代码如下。

```
protected void btnGetInfo_Click(object sender, EventArgs e)
{
    lblInfo.Text="<br>客户端浏览器名称:"+Request.Browser.Type
          +"<br>版本号:"+Request.Browser.Version
          +"<br>客户端使用的操作系统:"+Request.Browser.Platform
          +"<br>客户端 IP 地址:"+Request.UserHostAddress
          +"<br>当前请求的 URL:"+Request.Url
          +"<br>当前请求的虚拟路径:"+Request.Path
          +"<br>当前请求的物理路径:"+Request.PhysicalPath;
}
```

（4）运行页面，单击相关按钮，运行结果如图 5.7 所示。

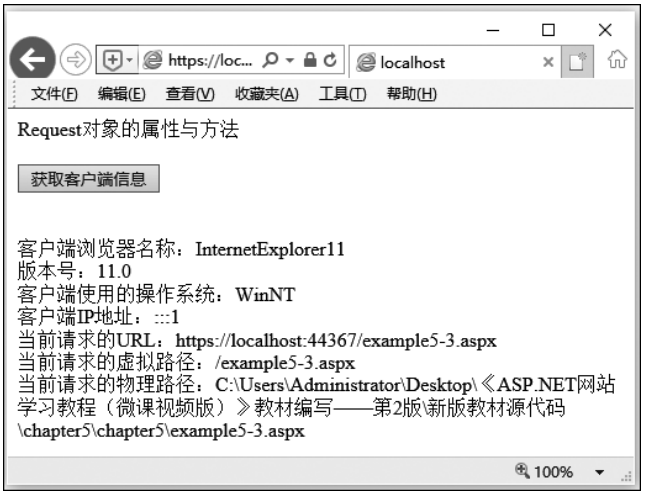

图 5.7 Request 对象应用页面演示

【示例 5-4】 在 E 盘 ASP.NET 项目代码的 chapter5 目录下，创建 example5-4 页面和 example5-4-2 页面，练习使用 Request 对象和 Response 对象进行地址栏传值。

（1）在 example5-4 网页和 example5-4-2 页面中添加相应控件，具体如图 5.8 和图 5.9 所示。

example5-4 页面：

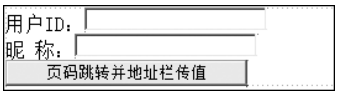

图 5.8 Request 对象应用的网页设计 1

example5-4-2 页面：

图 5.9 Request 对象应用的网页设计 2

（2）设置相关控件的属性，如下列源代码所示。

```
example5-4.aspx 页:
<form id="form1" runat="server">
        用户 ID:<asp:TextBox ID="txtID" runat="server"></asp:TextBox>
        <br/>
        昵 称:<asp:TextBox ID="txtName" runat="server"></asp:TextBox>
        <br/>
         <asp:Button ID="btnRedirect" runat="server" onclick=" btnRedirect_
Click " Text="页码跳转并地址栏传值"/>
</form>
example5-4-2.aspx 页:
<form id="form1" runat="server">
        欢迎登录<br/>
        ID:<asp:Label ID="lblID" runat="server"></asp:Label>
        <br/>
        Name:<asp:Label ID="lblName" runat="server"></asp:Label>
</form>
```

（3）为事件添加 C＃代码如下。

```
example5-4.cs:
protected void btnRedirect_Click(object sender, EventArgs e)
{
    Response.Redirect("example5-4-2.aspx? id="+txtID.Text+"&name="+txtName.
Text);
}
example5-4-2.cs:
protected void Page_Load(object sender, EventArgs e)
{
    if (Request.Params["id"] !=null && Request["name"] !=null)
    {
        lblID.Text=Request.Params["id"].ToString();
        lblName.Text=Request.Params["name"].ToString();
     }
     else
     {
        Response.Redirect("example5-4.aspx");
     }
}
```

（4）运行页面，在 example5-4 页面输入用户 ID、昵称，单击“页面跳转并地址栏传值”按钮提交服务器，如图 5.10 所示。在 example5-4-2 页面获取地址栏中的参数值，显示在页面中，如图 5.11 所示。

图 5.10 Request 对象应用页面演示 2

图 5.11 Request 对象应用页面演示 3

5.4 Server 对象

5.4.1 Server 对象的属性和方法

扫一扫

Server 对象是 HttpServerUtility 类的一个实例，提供对服务器属性和方法的访问功能，可以处理页面请求时所需的功能，如建立 COM 对象、字符串的编译码等工作。Server 对象的主要属性如表 5.7 所示，主要方法如表 5.8 所示。

表 5.7 Server 对象的主要属性

属 性 名	说 明
MachineName	获取服务器的名称
ScriptTimeOut	获取或设置请求的超时值(单位为秒)

表 5.8 Server 对象的主要方法

方 法 名	说 明
Execute()	执行指定的资源，并且在执行完之后再执行本页的代码
HtmlDecode()	对 HTML 编码的字符串进行解码
HtmlEncode()	对要在浏览器中显示的字符串进行 HTML 编码
MapPath()	获取指定相对路径在服务器上的物理路径
Transfer()	停止执行当前程序，执行指定的资源
UrlDecode()	对已被编码的 URL 字符串进行解码
UrlEncode()	对 URL 的字符串进行编码，通过 URL 从 Web 服务器到客户端进行 HTTP 传输