相关主题 RECOMMEND ARTICLE
- 你必须知道的.NET之对接口和抽象类
- ASP.Net中利用CSS实现多界面两法
- 将ASP页面转换成HTM静态页面的方法
- ASP.NET技术获取IP与MAC地址的方法
- ASP.NET移动开发之SelectionList控件
- ASP.NET中为GridView添加删除提示框
- 理解ASP.NET与客户端缓存之HTTP协议
- ASP.NET中Session的状态保持方式浅议

MORE

推荐文章 RECOMMEND ARTICLE
- 数据广播方案的优化
- 网络游戏的位置同步
- 游戏音乐制作案例之《战火 红色警戒》音效制作揭秘
- 英雄连Online 原画
- 游戏音乐制作案例之《乱武天下》
- 游戏音乐制作案例之《诛仙》
- 《鹿鼎记》最新原画
- MIDP2.1规范的新特性

MORE

热门文章 HOT ARTICLE
- [电子书下载]游戏设计 — 原理与实践
- [电子书下载]网络游戏开发
- 游戏设计全过程
- [电子书下载]游戏设计技术
- [电子书下载]游戏设计理论
- CS游戏人物模型制作教程
- CG人物插画基本流程
- [转贴]MAX高级人头教程

MORE

您的位置： .NET

文章标题	ASP.NET页面事件：顺序与回传详解		
来源：	[OGDEV]	浏览：	[373]

一、初始化

· 当页面被提交请求第一个方法永远是构造函数。您可以在构造函数里面初始一些自定义属性或对象，不过这时候因为页面还没有被完全初始化所以多少会有些限制。特别地，您需要使用HttpContext对象。当前可以使用的对象包括QueryString，Form以及Cookies集合，还有Cache对象。注意：在构造函数里是不允许使用Session的。

· 下一个将执行的方法是AddParsedSubObject方法，这个方法将添加所有独立的控件并把页面组成一个控件集合树，这个方法经常被一些高级的页面模板解决方案(Page Template Solutions)重写以便添加页面内容到页面模板(Page Template)中一些特殊的控件中。这个方法递归应用到所有的页面控件及相应的的每个子控件，所有的控件都是在这个方法中开始最早的初始化。

· 页面类中下一个将执行的方法是DeterminePostBackMode。这个方法允许您修改IsPostBack的值及相关的事件。如果您需要从数据库中加载ViewState这个方法将特别有用，因为ViewState只有在IsPostBack为真的情况下才会进行恢复。返回空将会导致强制执行非回传，返回Request.Form则强制执行一个回传。除非在特殊情况下，否则并不建议去操作这个，因为这个还会影响其他的事件。

· 下一个将要执行的方法是OnInit方法，一般这是第一个真正被使用的方法。这个方法触发时，所有页面定义中的控件执行初始化，这意味着所有在页面中定义的值应用到相应的控件上。不过，ViewState和传回的值还不会应用到控件上，因此，任何被代码或用户改变的值还没有被恢复到控件上。这个方法通常是最好的创建、重创建动态控件的好地方。

二、恢复及加载

· 下一个方法，LoadPageStateFromPersistenceMedium只会在页面被回传的时候才会被执行。如果因为使用Session或自定义存储方式，您修改了后面将要提到的影响ViewState保存方式的方法SavePageStateToPersistenceMedium，则这个方法需要被重写。默认的实现中ViewState是一种Base64格式编码，并且被保存在页面的隐藏域中，您可以使用这篇文章中提及的方法修改ViewState按以上两种方式保存。注意：这个方法并没有真正加载ViewState到页面或页面控件中。

· 当得到ViewState后，下一个方法LoadViewSate，将以递归的方式恢复ViewState到页面及各个页面控件或子控件中。这个方法执行后，每个控件都将恢复到上一次的状态，但是用户提交的数据还没有应用到控件上，因为他们不是ViewState的一部分。这个方法主要用于恢复您在其他事件中动态生成的控件的值，他们的值是您手动保存在ViewSate中，并且现在已经失效。

· 下一个方法是ProcessPostData，这个方法也同样是回传的时候才会被执行，并且不允许被重写，这个是页面基类的私有方法。这个方法通过匹配控件的名称恢复相应的用户提交的控件的值，到这一步意味着整个页面都已经被完全恢复了。唯一要记住的是所有动态控件的创建必须在这个方法之前。这个方法也是记录后面的改变事件的方法。

· 下一个方法是OnLoad方法，通常这是用得最多的方法，因为这个方法是页面生存期第一个恢复了所有值的地方。大多数代码根据判断IsPostBack来决定是否重新设置控件状态。您也可以在这个方法中调用Validate并且检查IsValid的值。也可以在这个方法中创建动态控件，并且该控件的所有的方法都会被执行以追上当前页面的状态包括ViewSate，不过不包括回传的值。

三、事件处理

· 下一个方法还是ProcessPostData，实际上就是前一个方法的另一次调用，它仍然是只在回传的时候执行并且由于是私有方法不可以被重写。如果您是第一次看页面的运行轨迹也许会觉得这个方法有些多余。但实际上这个方法是必要的因为在OnLoad中创建的动态控件也需要他们回传的值。任何在这以后创建的控件将可以得到他们的ViewState，但是不能再得到他们的回传的值，并且不会触发任何值改变事件(Change Event)。

· 下一个方法，RaiseChangedEvents，也是只在回传页面中执行，并且也因为是基类的私有方法所有不能被继承。在整个页面生存期中，是在这儿根据之前的ProcessPostData记录的控件的值和提交的值是否不同来触发值改变事件。您也许需要调用Validate或者检查IsValid的值。这里并没有特别的说明多个值改变事件的执行先后顺序。

· 下一个方法，RaisePostBackEvent，同样是因为是基类的私有方法不能被继承，同样也是只在回传页面中执行。除非使用了AutoPostBack，不然这是实际提交表单事件执行的地方，特别是按钮或者其实使用javascript提交表单等。如果还没有被手动调用过并且使用了验证控件，那么Validate会被调用。注意IE中有个BUG有时会允许提交但却不触发任何事件。

· 下一个方法是OnPreRender，一般这是在客户端展现页面之前改变页面及其控件的最后一次机会。您也可以在这个方法里面创建动态控件，并且所有的方法都会被执行以追上当前页面的状态包括ViewSate，但是私有方法将不会被执行，这意味着不会有回传的值并且不会有事件触发。由于IE中的BUG，这是一个没有事件赶上PostBack的好地方。

四、保存及显示

· 下一个方法是SaveViewState，不论是否是回传页面，均会递归的执行以保存页面及其所有控件的ViewState。ViewState基本上保存所有与定义在aspx中的原始值不同的值，不管是被代码还是用户所改变。注意控件值是根据他们在页面的控件树中的位置来保存的，所以如果动态控件后来加到了错误的位置将会导致混乱。

· 下一个方法是SavePageStateToPersistenceMedium真正的保存页面的ViewSate。这个方法随同LoadPageStateFromPersistenceMediumg 一起被重写以便保存ViewState到Session或其它自定义数据，而不是用隐藏域。这对于低带宽的用户来说是很有帮助的。并且对于移动设备来说，Session是默认设置。下面这篇文章描述了使用以上两种方

式保存ViewState的具体细节。注意在Asp.net中有个Bug：Asp.net要求必须提交__viewstate字段，即使是空的。

· 下一个方法是Render方法，该方法递归的创建并发送相应控件的html给浏览器。这个方法被一些页面模板方案重写以添加一些通用的页面头与脚而不使用服务器控件，他们总是有一些额外的东西。注意这儿的修改只能使用纯HTML，因为控件在这儿已经被生成了。您可以用StringBuilder，StringWriter，HtmlTextWriter捕获相应的HTML输出。

· 最后的方法是OnUnload，这个方法会调用相应的Dispose方法。这个方法提供机会以便清空该页面中使用的非托管资源，如关闭打开的文件句柄，以前打开的数据库连接等。注意这个方法是在页面已经发送到客户端以后执行的，所以它只有影响服务器对象，并且它不会显示在页面的显示轨迹中。这就是页面的生存期，对于每一次请求都是这么运行的。

表1：页面事件总结

方法回传控件

ConstructorAlwaysAll

AddParsedSubObjectAlwaysAll

DeterminePostBackModeAlwaysPage

OnInitAlwaysAll

LoadPageStateFromPersistenceMediumPostBackPage

LoadViewStatePostBackAll

ProcessPostData1PostBackPage

OnLoadAlwaysAll

ProcessPostData2PostBackPage

RaiseChangedEventsPostBackPage

RaisePostBackEventPostBackPage

OnPreRenderAlwaysAll

SaveViewStateAlwaysAll

SavePageStateToPersistenceMediumAlwaysPage

RenderAlwaysAll

OnUnloadAlwaysAll

无任何评论！

请您注意：

- 尊重网上道德，遵守《全国人大常委会关于维护互联网安全的决定》及中华人民共和国其他各项有关法律法规
- 尊重网上道德，遵守中华人民共和国的各项有关法律法规
- 承担一切因您的行为而直接或间接导致的民事或刑事法律责任
- 中国网游研发中心新闻留言板管理人员有权保留或删除其管辖留言中的任意内容
- 您在中国网游研发中心留言板发表的作品，中国网游研发中心有权在网站内转载或引用
- 参与本留言即表明您已经阅读并接受上述条款

发表评论：

昵　　称：

GO

联系EMAIL：