**WebLearn**

# aspx语法

## 1.前端语法

### 1）表格

<table>表<tr>行<td>列</td></tr></table>

标题或表头中，td换成th

属性colspan表示合并单元格

### 2）asp控件

微软定义了很多asp控件，只要拖动到aspx文件即可自动生成对应的标签，每一个控件（每一个标签）都有自己的功能。另外，目前使用到的是可以在aspx文件对应的cs类中，直接this点出控件名，并且绑定数据源DataBase后DataBind显示到aspx文件浏览的网页中。如Repeater。

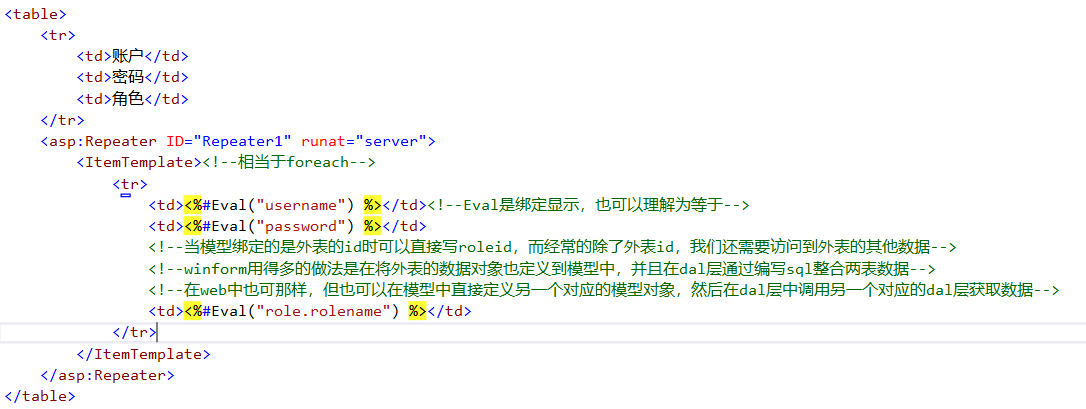
### 3）遍历数据源标签<ItemTemplate></ ItemTemplate>

在aspx文件对应的类中绑定了数据源后，在对应控件标签下使用此标签即相当于foreach遍历数据源。

### 4）绑定显示方法Eval（aspx语法）

Eval是JS的一个绑定显示的方法，在aspx文件中的语法是<%#Eval(“”)%>，参数是数据源中的表字段。另外，此方法可以理解为是=，因为数据源中存在某个类的实例，而需要显示的是这个类实例的一个属性，则可以填入此类实例点出属性。如<%#Eval(“RoleInfo.id”)%>即<%#Eval(“roleid”)%>。

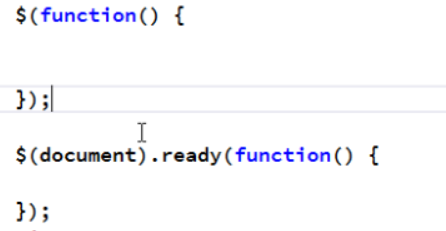
注意：此方法是单向绑定，即只能读数据，而不能更新数据。对应的Bind方法也是绑定显示，但它是双向绑定，可更新数据。



### 5）JQuery语法

$表示选取。

下图中两个写法其实是一样的，都是在dom加载完之后执行的。



#### $(“#元素id”)

即相当于DOM的getdocumentbyid，从而获取指定id的元素。此外，使用函数或者this关键字都需要使用$，如$(this)才表示获取了this关键字。标签元素的方法blur是失去焦点时触发的。另外，$(“:标签类型”)即可获取所有此类标签。方法html，可以替换标签中所有的html内容。Empty方法可清空数据。

#### 选择器：

还可以通过$(“#元素id 子元素类型”)获取元素下所有符合类型的子元素。子元素类型还可以传入标签名:contains(‘内容’)，表示获得具有该内容的对应标签的元素。

#### 用于判断输入

一般都是使用正则表达式，而且JS的正则表达式返回的对象有test方法可以校验字符串。而校验不通过则可以输出通知用户，$(“元素id”).test用来显示错误信息（此元素一般是span标签）

#### $.ajax

而用于校验输入与数据库之间的数据冲突时，即发送异步请求进行校验数据。可以使用$.ajax（阿贾克斯），也可以使用$.post，在post方法中传入一般处理程序、一个给一般处理程序调用的字段（格式一般是”字段名”:字段值）以及一个回调函数。其中，一般处理程序即为ashx文件。（ajax实际上是异步，异步见后面的笔记）

#### Css

css方法传入字符串display和字符串none即可隐藏标签。把后面的none该为block即可显示。

#### EasyUI的引用

1. 引用对应的jQuery的js库
2. 引用对应的需要使用的easyUI控件的js库
3. 引用easyUI的语言文件js库
4. （在jQuery引用的前面）引用css主题样式：首先是easyUI的样式，然后是图标的样式

显示EasyUI见14点模态。

### 6）HTML占位符

在html文件中的某个区域中，可以使用@参数名来生成占位符，此时在外部获取到此html模板之后，即可通过占位符把需要的其他前端代码放到HTML文件中去。如ashx

### 7）数据提交到后台

一般使用的是表单标签<form></form>。前台获取时，通过context的Request的Form集合就可以获得。

实际上，不管是Form还是QueryString，只有Request能点出来的集合均可以通过Request直接以集合的方式取得，在其索引器（this[]）中，Request先查询QueryString，再查询Form，最后是Cookie，按照这个顺序取索引对应的参数值。

### 8）空格

如果在前端代码中输入空格，那么最终连在一块的只会被识别成一个。因此可以使用&nbsp代替。

### 9）确认弹窗

如删除时询问是否确定删除。confirm(显示语句)：提示用户，包括是否按键。如放在a标签的onclick方法中时，会自动判断是否执行此超链接。

### 10）input标签

input标签可用作输入框，即type属性设为text。另外，此标签还可以用作按钮，type设为submit即可。

#### 不明白：

为何设置了submit的input按钮，点击之后会跳转到表单属性action对应的ashx文件呢？难道前端设定就是这样？

Type设为file之后，界面会显示一个按钮（选择文件）即一个显示文件名的标注。

### 11）<a>标签

target属性，设置为\_blank时表示另起一个窗口打开网页。

当点击a标签时，网页会发生跳转。若要使网页不跳转，有两种方法：

1. 在href中不填入#或者其他的地址，而是填入javascript:void(0)
2. 在a标签的click方法将结束时，return false。

### 12）iframe标签

可用来显示网页

### 13）select标签

下拉框标签，其中option标签可为其添加元素。

### 14）模态

即弹出对话框，使用户无法对主页面进行操作。调用模态窗体使用dialog（不传递参数），传入参数-字符串close即可以调用其关闭方法。

实际上，给一个模态对话框，其实是在主界面中搞一个div覆盖整个页面，然后调用JQuery的easyui。

## 2.一般处理程序ashx

### 1）ashx与其对应的cs类

在ashx的cs文件的ProcessRequest方法中调用参数context的request集合属性（记住：这是一个集合，需要用[]来得到对应字符串）。如前端$.post方法的第二个参数，即在[]中传入字段名即可。之后即可以用C#的语法判断这个参数值是否符合规则。之后再通过context的response属性的write方法向前端代码返回一个对象，由回调函数的参数接收。

### 2）ashx向前端的输出与request和response

另外，context上下文还可设置响应类型，即context的response属性的ContentType属性。（默认text/plain，常用text/html、image/jpg）

一般处理程序可以用来直接显示前端，通过text/html传入标签组成的字符串。连接网站时，如果要传入参数，需要在网站后面加上?参数1=参数1值&参数2=参数2值，之后通过ashx中context的Request属性的QueryString属性集合取得参数值。

Response属性buffer，默认true：使用缓冲区，即把所有显示都先写入缓冲区，然后统一输出。方法flush，立刻向前台输出缓冲区中的数据。Clear方法，清空缓冲区。ContentEncoding输出流（响应报文）编码。OutputStream拿到当前输出流。End终止响应，相当于调了flush方法，但是后面的代码（可能）不会继续执行。Response.Write是将前端语句写到前端代码的最开头。

### 3）ashx操作前端页面的回跳

当从某个ashx跳转到另一个ashx进行了一系列操作后希望回到原来的ashx页面，需要调用Response的方法Redirect方法。

### 4）使用html模板

使用File方法的ReadAllText即可获取文件模板。但是IO流只能使用绝对路径。故需要将相对路径修改为绝对路径，调用context的Request的MapPath方法即可。之后通过ReadAllText取到了模板的字符串类型，接着通过字符串的replace方法替换掉占位符即可。最后把替换后的字符串write输出到前台即可。

### 5）结束Http访问

使用response的End方法可以时一般处理程序直接结束，不向后执行。

### 6）判断网站是否本地网站

Request.UrlReferrer属性可获取访问的网站。Request.Url可获取本地网站。

## 3.aspx文件

aspx文件可以相当于是winform的窗体的设计界面，用于界面元素的生成。它是可以认为是由ashx文件与html模板组成的一个设计文件。即aspx文件实际上包含前台页面文件与后置类文件，前台页面文件继承与后置类文件。

一般不涉及到前端页面的操作，超级链接均使用ashx，有涉及的就使用aspx。例如删除，点一下就执行操作的，直接使用ashx。而修改，点一下还需要显示详细消息的，则使用aspx。

### Server对象：

MapPath通过相对路径获得绝对路径，UrlEncode编码、UrlDecode解码，HtmlEncode编码、HtmlDecode解码（html页面中把标签的<>分别换成&lt;和&gt;可以使此标签不被识别为html语句，而是普通字符串。这个编码即是把html语句转化为代表普通字符串）,Execute（在前端的<%%>标签中使用。类似iframe标签，但是查看源码时，使用frame标签的那块源码需要查看框架源码才可看到，而使用execute的相当于把源码从子网页嵌入到主网页中。Execute更有利于搜索引擎优化。）和Transfer（在后置类中使用，后台接管。即不显示当前页，而显示目标页。与Response.Redirect的区别：transfer是直接跳转到目标页、且只能在本网站间跳转，属于服务器内部接管，浏览器无法识别，而redirect是先访问当前页、再跳转到目标页、可以跳转任意网页）执行另一个页面的代码。

## 4.aspx语法与html与JavaScript等语法

实际上，aspx前台页面的文件中就可以使用html与JavaScript语法，微软还另外加入了一些与C#类互用的语法。

### 1）<% %>标签

<% %>标签就是表示调用C#代码，而且其中只能存放C#代码。若希望在其中添加html语句，则可以把C#代码拆分。如图



### 2）前台页面文件头

<% %>标签中@ Page表示这是一个页面，后面分别表示编程语言、是否会触发事件、后置类、父类。

在文件头中，添加Trace=”true”可以跟着整个asp.net工作流程的信息。

### 3）前台页面文件的=

<% %>标签中= 值（此值可以是后置类的一个值）表示要在此位置输出这个值。

### 4）<body>下第一个<form>标签

在aspx文件中，这个form是必须存在的，而且其属性runat必须设置为server。

### 5）网页图标文件

必须放在网站根目录下。

### 6）前台与后置类的交互

可以在后置类添加一个公共属性，赋值后在前台的HTML标签中通过<%%>标签调用即可。

也可以在后置类写好一个HTML字符串，然后直接在前台通过<%%>标签调用。

此外。与ashx不太一样，没有context。而是直接使用父类的成员Request（取参数）和Response（跳转页面）。

### 7）页面的一些操作交互

例如添加内容：当用户点击添加时，需要向对应的aspx请求数据，显示添加页面，而填充完页面后点击提交时，需要再向同个aspx请求数据，如何判断是请求查看还是请求提交数据？使用IsPostBack属性。

### 8）Page

Page.RegisterStartupScript方法，与Response.Write类似，也可以向前端写入前端语句，但是Page的这个方法是将语句写在代码末尾。

Page\_Init方法是C#中能被自动识别的方法（即创建Page类的派生类时，此方法不会自动生成，需要自己手动写出来，但是即使没有引用也可以根据代码执行），它是所有页面初始化时先执行，在Page\_Load方法执行之前就执行。

因为Page类是动态页面，有些方法按照规定的名称写入即可自动注册。

### 9）引入命名空间

使用%标签，Import关键字，后面接Namespace=””即可。

# 二、学习思路

## 1.模型设计、关系处理

当模型绑定的是外表的id时可以直接写外表id，而经常的除了外表id，我们还需要访问到外表的其他数据

winform用得多的做法是在将外表的数据对象也定义到模型中，并且在dal层通过编写sql整合两表数据

在web中也可那样，但也可以在模型中直接定义另一个对应的模型对象，然后在dal层中调用另一个对应的dal层获取数据

## 2.浏览器访问机制

当浏览器访问网站时，它会调用写好的前端代码先渲染显示，在这个过程中，若第一次访问则会调用所有文件然后进行渲染，之后浏览器会将样式文件CSS以及JS文件等缓存起来，这样就保证了下一次打开网站的时候效率的提高。

然而，当我们修改这些缓存的源文件后，浏览器却依然会调用缓存中的，而不会重新渲染。针对这个问题，我们一般会在前端代码调用这类文件时，在最后加上?v=，表示这个文件的版本号，所以在修改了对应的文件后，顺便在调用中把版本号做修改，这样浏览器访问网站的时候才会发现文件变了才会重新渲染。

## 3.Post请求与Get请求

从目前的理解来看，Post是通过正文的表单来传输数据的，而Get是通过网址的参数来传递数据的。默认Get。也就是说即使在前端使用表单，如果使用的是Get请求，它传递的数据还是通过参数的形式，即QueryString。

另外，浏览器输入地址的控件一般都有限制长度（如ie限制2kb）。所以传输文件等较大数据一般使用Post请求。

## 4.web的工作原理

浏览器向IIS服务器发出请求，IIS向服务端发出全新的请求，再由IIS从服务端那边把数据传到服务器。

### 注意：

IIS每次向服务端发出的都是全新的请求，即服务端上的每一个文件都是被重新渲染一边的。因此要使浏览器界面数值自增，无法直接在ashx文件中对某个变量++，而需要获取页面提交的表单数据来实现++。

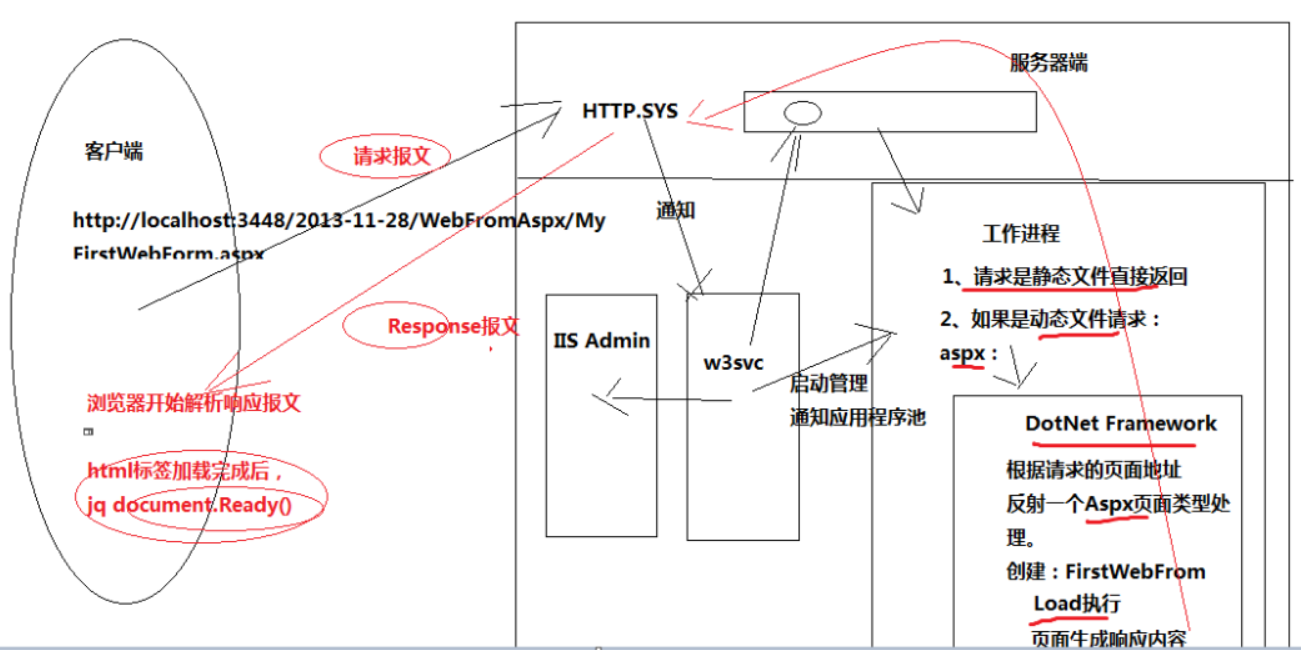
只有表单标签才能随着请求把数据提交到后台。三类：input、select、textarea，以name为key，以value为值提交到后台。

### Asp.net总结：

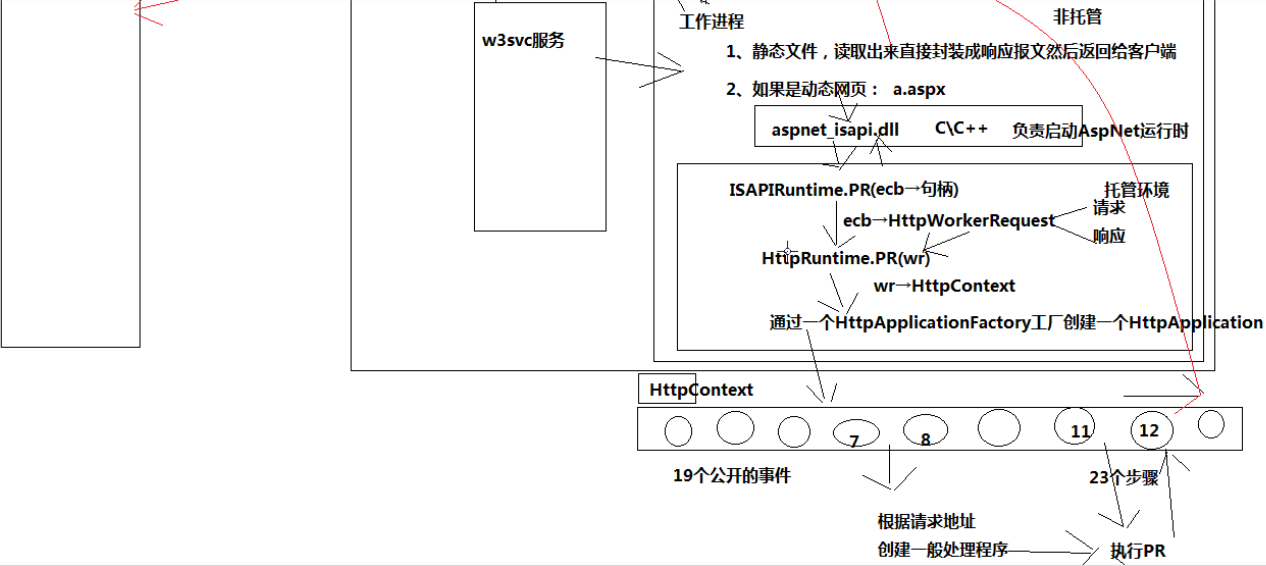
首先是浏览器端发送封装的http请求到服务器端到内核模块http.sys（负责应用程序池管道的维护），模块把请求放到管道中。另外有两个模块（iis和w3svc——控制程序池的启动，初始化，如果已经初始化好则通知取请求）以及应用程序池（存放工作进程w3wp.exe，管理w3svc服务）。

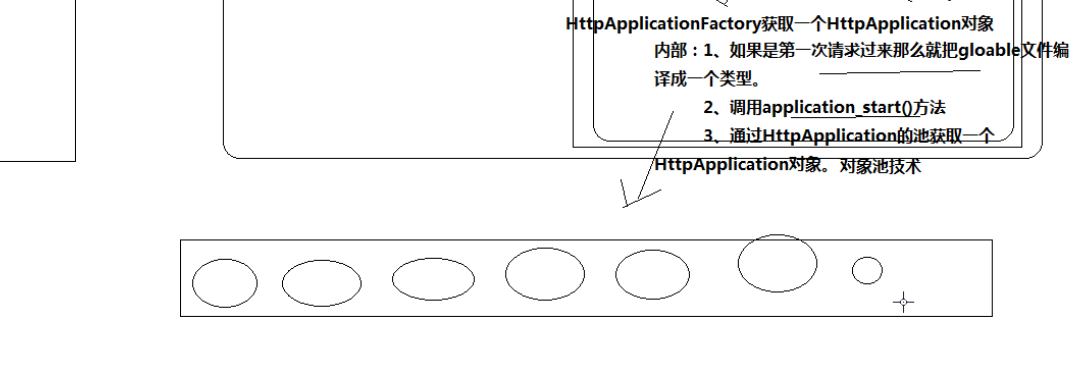
在应用程序池中，拿到请求后如果是静态的，则直接处理返回。如果是动态的，那么交给apsnet\_isapi扩展模块来处理，模块启动asp.net运行时，并把请求发送到托管环境。请求分发给托管环境后，到了托管类型ISAPIRuntime.PR(ecb句柄)，在类型内部把ecb句柄封装成HttpWorkerRequest对象（具有请求报文和响应报文（暂时是空））。接着把请求发送给HTTPRuntime.PR(wr)方法处理，在方法内部把wr封装成HTTPContext对象（即http请求上下文对象）。

封装好请求后，通过httpapplicationfactory工厂获取httpapplication对象（内部工作：1.如果是第一次请求过来，那么把global文件（在网站根目录的一个asax文件，继承自httpapplication）变成一个类型，2.调用Application\_Start方法（整个网站生命周期内只被调用一次，应用程序使用的时调用），3.通过httpapplication的池获取一个httpapplication对象（对象池技术）），从而控制管道的流动。

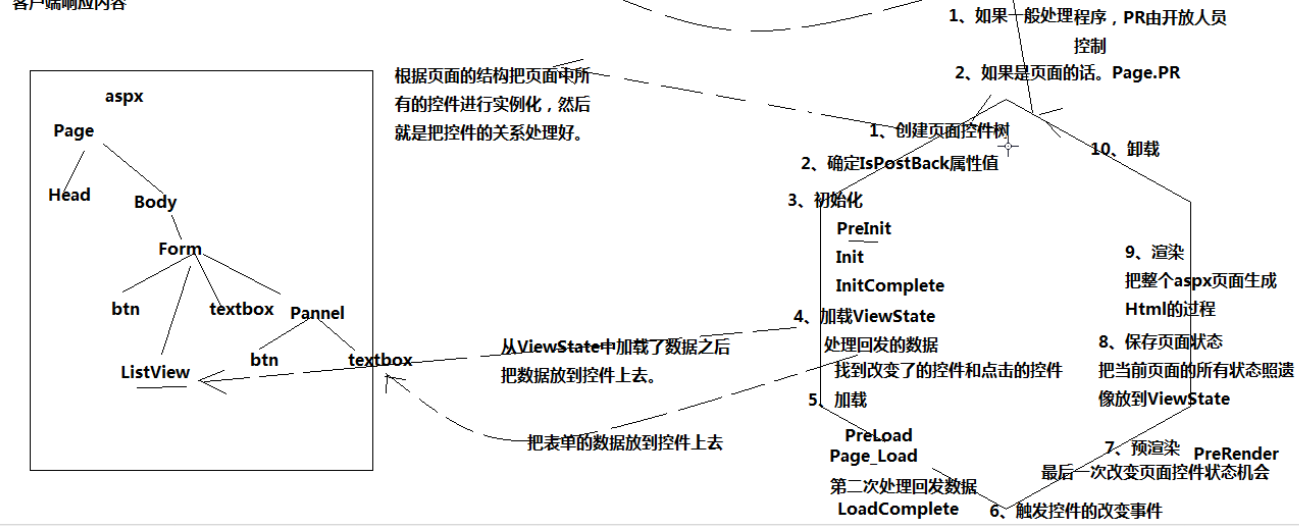


管道中有20多个步骤，其中有19个公开事件，在第7和8个事件之间根据请求地址创建页面对象或一般处理程序，第9个事件开始加载session（根据请求报文中的cookie的id找到对应的session对象相关联。当然，若一般处理程序没有实现接口IRequiresSessionState则直接越过这个一般处理程序对象，不加载会话），在11和12之间页面对象或一般处理程序调用PR方法（页面生命周期）。





在PR方法（一般处理程序的PR由开发人员控制，页面的PR靠Page.PR）中。PR方法的过程：1.创建页面控件树（根据页面的结构，把页面中的所有控件进行实例化，并把关系处理好），2.确定IsPostBack值，3.初始化（三个阶段：预初始化PreInit、初始化中Init、初始化完成InitComplete），4.加载ViewState、处理回发数据（找到改变了的控件和点击了的控件），5.加载Page\_Load（也是前中后三个阶段，第三个阶段不是在此处执行。其中将第二次处理回发数据），6.触发控件的改变事件、点击事件等（后触发Page\_Load的完成事件），7.预渲染（最后一次改变页面控件状态的机会。PreReader方法），8.保存页面状态（把当前页面所有状态照遗像放到ViewState），9.渲染（把整个aspx页面生成HTML的过程）



管道执行结束后，再次通过扩展模块调用内核模块，由内核模块向浏览器端发送响应。

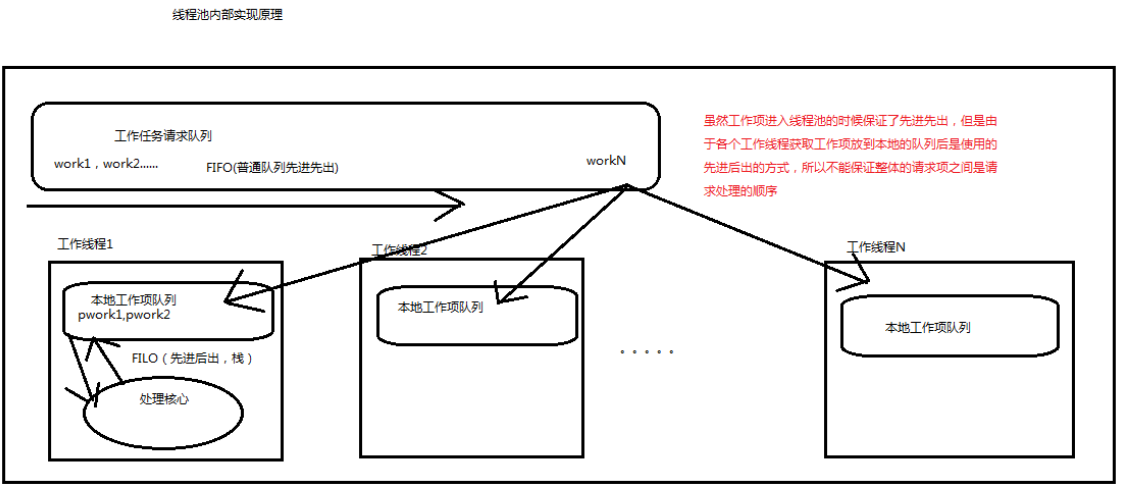
## 5.web应用程序工作流程

### 1）流程问题解答

浏览器加载页面和服务端加载aspx文件没有半毛钱关系？指的是：浏览器发送请求后，服务端处理后会把响应报文发送给浏览器，同时处理过程中涉及的所有资源都被释放，这样看起来就相当于没有关系。

ecb句柄是连接非托管环境和托管环境的纽带。ecb拿到请求流，然后将它封装成HTTPWorkerRequest，然后在进一步封装成HTTPContext。接着由工厂创建HttpApplication对象，由此对象负责运行管道。其中，管道流动时，上下文对象就跟着流动。

### 2）线程池内部实现原理



Runtime请求队列，排到的就拿给application处理

## 6.网站与web应用程序（website和aspnet）

VS中，网站改完之后，可以对某个页面直接通过浏览器浏览，而web应用程序不行。因为网站浏览时会自动生成，而web应用程序需要手动生成。

在网站项目中，其实仅仅是相当于创建一个文件夹，每一个网页都相当于一个独立的程序集，某一个报错不会影响另一个。而web应用程序则是仅一个程序集，每个页面都互相关联，当其中一个报错，另一个将无法运行。

网站没有命名空间，因为每个网页都是独立的，但需要把所有类放在App\_Code文件夹下面。命名空间仅为了防止页面混乱。Web应用程序不需要App\_Code文件夹，会报错。

暂时理解为：web应用程序比较安全，所以一般都是使用web应用程序。

## 7.数据传输方式

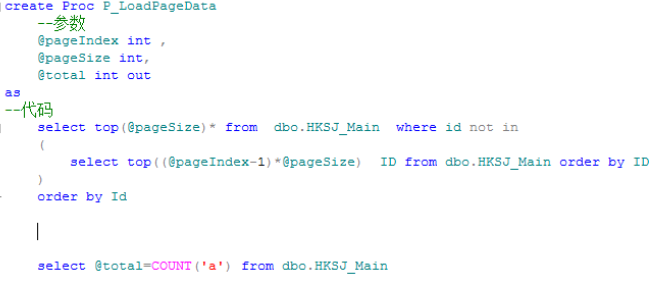
使用<form>标签传输数据时，默认enctype方式是urlencoded，它传输的数据格式是key1=value1&key2=value2，即是QueryString。只能用来传递简单的字符串，不可用来传输文件。

上传二进制的文件使用form-data方式。此时一般程序接收用request.files数组获取内容（不可直接使用request数组），得到HttpPostedFile对象，可以使用其属性与方法：Filename、SaveAs(绝对路径)。

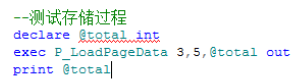
## 8.数据库操作

### 1）存储过程

使用Proc和as关键字创建存储过程。后面可加begin和end

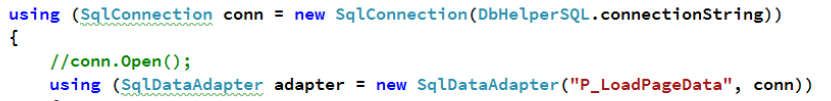


测试存储过程使用exec关键字，存储过程输出的内容使用declare关键字先声明一个变量。



要使存储过程返回多个表数据，只需要在as关键字后面的代码加多一条select的sql语句即可。

C#调用存储过程：如果有输出参数，就必须使用SqlDataAdapter。使用SqlDataReader取不到输出参数。



#### 优势

1. 比dotnet直接写sql脚本执行少了一块解析编译的过程。效率更快一点点。
2. 使用存储过程的时候，业务改变只需要修改存储过程即可，然后业务逻辑就发生变化了，不需要修改C#代码。

貌似方便

1. 传递sql脚本数据相对小

#### 缺点

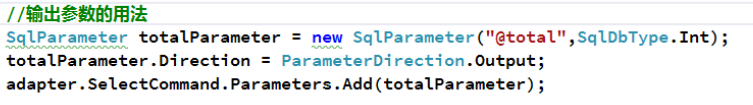
1. 使用存储过程，数据库可移植性差。如不同数据库移植后，存储过程要重写。
2. 把业务放到了存储过程里面去，相当于业务处理的压力放在数据库里去。（数据库介质存储在磁盘上，而磁盘速度比起内存慢，因此会导致速度变慢）。因此，业务在业务层处理，而数据库只是用来存储数据。

为啥这么做：1）解放数据库的压力2）数据库本身做集群很困难，业务去做分流、做集群容易。

### 2）count关键字

count(\*)表示找最短的列统计总行数。count中的参数表示某一列，若不存在则默认临时创建一列。

### 3）C#获取数据库输出参数



多了设置参数方向Direction，默认是输入参数。

## 9.页面状态保持

### 1）无状态http

不会记住上一次发生过什么。

个人理解：通过http访问网站时，网站的后置类即使有对数据进行自增等其他操作，每次访问网络时也将自动重置（重新创建一个后置类对象）。只有通过前端代码得到的变量才可以保持。

### 2）ViewState

直接在后置类写上ViewState[“key名”]=值，即会在运行时的前台插入一个隐藏标签。查看源码就可以看到值是一个由base64加密的字符串。

其实说白了就是一个隐藏域，就是被微软封装起来罢了。完全可以被隐藏域替代，定义公开的变量，然后在第一次请求页面时把变量值赋给一个隐藏域，后面请求时通过隐藏域的name去取即可。

#### 适合场景：

页面给自己的后置类传值。在asp中，可以通过拉拽TextBox控件来实现（其内部也是使用ViewState实现的）

VS的web控件很多都是通过ViewState实现的，因此有些会导致数据过长。如列表控件，展示很多数据时，ViewState会保存很长的数据。

#### 禁用ViewState

在前台代码中，开头页说明中添加EnableViewState属性为false。这样禁用的是整个页面。不过，不刷新页面，控件的值还是会保持，但是每次都是变化的（暂不是很明白）

也可以仅禁用控件的ViewState，设置控件的EnableViewState属性即可。

另外，WebForm的IsPostBack属性也依赖于ViewState。

### 3）Cookie

一小段文本信息（最大4096），与网站关联（一个网站只能访问对应的cookie），存储在浏览器端（数量有限，一般是20个，超出则新替旧。所有站点的cookie加起来不超过300个）

在浏览器调试中Resource可以查看cookie。

给浏览器端设置cookie，Request.Cookies[“key”].value=…即可。服务器获取cookie也是直接value。

给cookie设置生命时长。使用Expires属性。如果设置了过期时间，那么cookie就会放在浏览器所在的磁盘上，否则就放在内存中。

domain：给cookie设置域。

也可以new一个httpcookie对象，然后add到Cookies中。

#### 1）主域与子域

主域与子域：如[www.baidu.com](http://www.baidu.com)和zhidao.baiodu.com

主域的cookie：子域能访问浏览器在请求子域的页面时，会把主域的cookie也一起发送到后台服务器。

子域的cookie：只能子域访问。子域添加cookie的时候，可以设置域为主域，这样这个cookie就被主域访问。

#### 2）限制路径

cookie的存在并不只是通过名称来识别的，当path不同时，同名的cookie若已存在，但是修改了path重新生成新的Cookie后也会存在。即此时存在两个cookie，一个有path一个无。

当设置了path后，只有在该网站的指定的目录下可以使用此cookie

#### 3）多值cookie

Cookie可以是一个二维数组，即Cookies[“”][“”]

#### 4）如何删除Cookie？

只需要把Expires设置为过期即可。

## 10.Session

当第一次请求开始时，就会有一段具有有效期的会话session分配过来。如果超过了这个有效期时间没有再次请求，则下一次请求时，得到的会话session又是一个新的session。

会话其实就是持续的一段时间。为用户保存会话期间相关数据的存储空间。

Session对象本身就是字典数组

当第一次请求获得会话时，服务器会把sessionid放到cookie中，每次访问的时候都会发送这个id。管道中的第9个事件开始加载session，根据sessionid加载session对象，赋值给上下文的session属性，第10个事件获取到session。

在其他页面给cookie赋值的时候，如果存在乱码（如中文，在传输时可能乱码），那么可能导致cookie中的sessionid无法被识别，此时则将出现找不到sessionid的情况，最终就变成了每次都是新的请求，session保存不了数据。所以，在可能出现中文的地方，赋值时使用server.urlencode编码，取的时候使用server.urldecode解码。

在aspx前台代码中，标签sessionState可以设置session超时时间timeout（分钟）

asp.net会话实现机制（猜想，不太理解）：1.轮询，把所有的session对象进行遍历，过期的强制放到session池（存储着，等待需要时再调出来使用，而不用重新生成）；2.懒处理：如memcached（缓存），当用session对象的时候，在池子中取，若池子没有，则遍历所有找到过期的，没有找到才创建新的。

登录验证码是最典型的session使用案例。

### 注意：

在一般处理程序中使用session时，需要让此一般处理程序继承一个接口：IRequiresSessionState。这是一个标记接口，没有任何方法：指定目标HTTP处理程序需要对会话状态值具有读写访问权。

## 11.校验是否已登录

通过session来完成，增加一个session标记来判断。一般把这个判断放在自己定义的基类中，此基类继承自Page类，然后我们写的所有aspx的后置类均继承自自定义的基类。另外，此判断需要写到aspx后置类加载数据之前，即Page\_Load方法之前，故推荐写入Page\_Init方法中。

## 12.Application

还有个Application字典对象，与session类似，只是它是全局的，很少用。仅相当于全局变量。

在global.asax文件的后置类中，有很多application对应的方法，这些方法都是用来注册管道中执行的事件的。

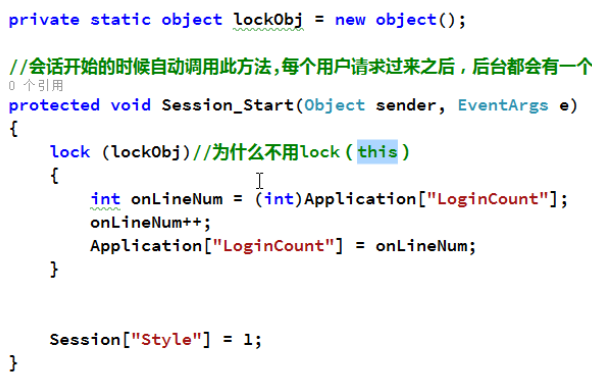
Application\_BeginRequest是第一个执行的事件，Application\_EndRequest是最后一个执行的事件，Application\_Error是整个网站出现异常的时候都会执行的方法。Application\_Start整个网站生命周期内只被调用一次。

Session\_Start和Session\_End表示会话开始与结束时执行的事件，可用来计算网站用户访问量。每个请求都有一个专门的线程在处理

### lock

为何锁住同一个全局变量，就可以使多人访问同一个其他变量时，就产生互斥呢？

暂时理解：锁同一个地址就是互斥。当不同用户都访问下图中的方法时，排前面的一个访问时就锁住了，紧紧挨着的第二个人即使是同时到的，也还是需要等待锁中的代码执行完成。



#### 字符串

相同的常量字符串，地址相同，因为字符串的回收机制不是普通的引用回收机制。new的才可能不同。



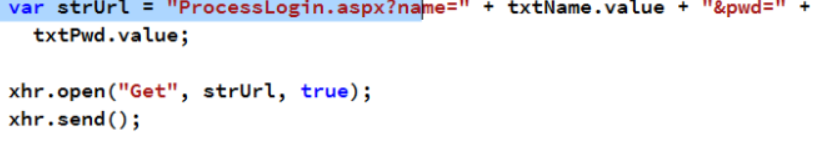
## 13.异步

### 用法：

首先在前台页面写一个JavaScript脚本中创建一个XMLHTTPRequest对象。



接着调用XMLHTTPRequest对象的open方法，第三个参数表示是否使用异步，设为true即可。接着使用send发送即可。另外，在open方法中，第一个参数是设置数据传输方式，若是Get则在第二个参数传地址时后面拼接参数，若是Post时，在send方法中传入参数。

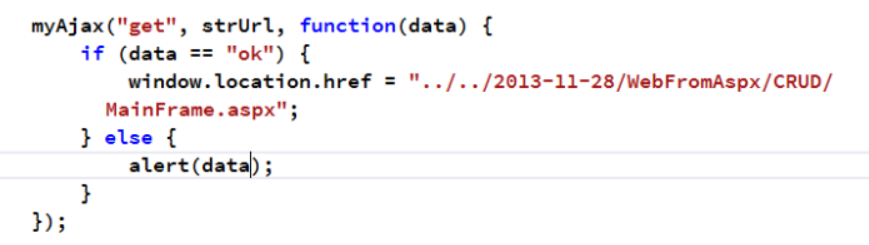


最后使用对象的方法onreadystatechange属性指向一个函数处理后置类的响应。在响应函数中，首先判断异步调用是否成功，判断对象的readystate属性是否等于4而且其status属性（状态码，200表示成功）是否等于200，若是说明异步响应成功，此时才开始判断后置类给的返回值是否正确。

若网站中有多处使用到异步，建议将其封装成一个函数，如下图：



调用时，传入对应的参数即可：



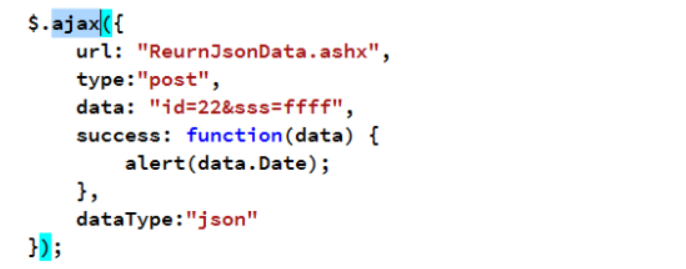
### JavaScript的参数：可传函数！！！

### JQuery

直接$.get就表示Get请求方式，此方法第一个参数是地址，第二个参数是请求参数（格式可以是json，也可以是key1=value1&key2=value2），第三个参数是回调函数。实际上，jQuery就是把上面所写的js的那个异步方法封装起来而已。

还有另外两种方法：post和getJSON。其中getJSON回调函数返回的对象是JSON格式的数据，若后置类不是返回JSON，那么前台获取到的数据将是空。

除了上面的方法外，还有一种最全面的方法$.ajax();，其参数为json格式。其中data键的值可以是如下格式，也可以是json格式；回调函数的data参数也是如此，而dataType设置的是回调函数参数的格式。ajax还有其他的参数：ASync可设置是否异步（默认异步），cache可设置是否启用缓存（false强迫当前请求必须去后台拿数据，不能用客户端缓存。默认true）



## 14.Json

最常见的格式：{key1:value1,key2:value2}。实际上json就是JavaScript的对象的格式。

也可以给json数据传入一个数组，此时相当于{0:数组第一个元素,1: 数组第二个元素}。

当遍历整个json数据时，实际上是遍历其key值。

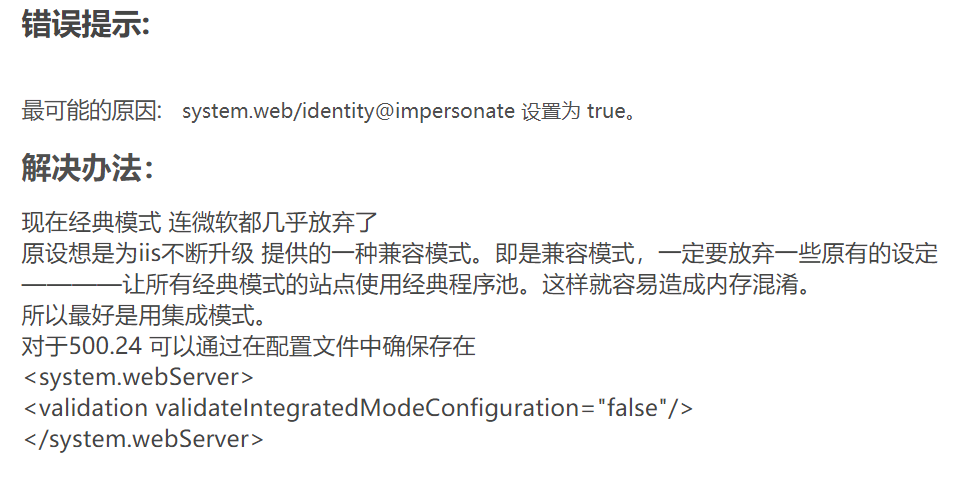
JavaScript的方法eval可以将字符串当做js脚本执行，当需要把一个具有json格式的字符串转化为json，也可以使用eval方法。var jsonData=eval(“(“+data+”)”)，其中字符串括号表示强制执行。当然这样的兼容性比较差。

要把一个具有json格式的字符串转化为json，使用jQuery是最好的，它封装了正则表达式。$.parseJSON方法

后台json转化：把一个集合转化为json字符串。使用System.Web.Script.Serialization.JavaScriptSerializer对象的方法serialize即可。

# 三、学习中遇到的问题

## 1.浏览网页出错



## 2.JS Not Found

明明有js引用路径，但是就是一直报错，暂时不知道为什么。

## 3.lock锁什么时候加

对象池中的某个对象被多个线程获取时，需要加锁。

## 4.反射的理解

暂时理解为：

通过某个类型或者动态字符串获取对应的Type类型，再通过Type类型去获取对象。

## 5.如何查看微软源码

使用软件Reflection即可查看微软源码

## 6.UTC时间

当前减去1970年1月1日后计算出的秒数。1970年之前的时间就是负数。