



吾生也有涯，而知也无涯。以有涯随无涯，殆已。

公告

昵称：Fish Li  
园龄：4年3个月  
荣誉：推荐博客  
粉丝：8009  
关注：5  
+加关注

最新随笔

- 1. 明源软件2014校园招聘
- 2. 明源软件2013社会招聘
- 3. IIS日志-网站运维的好帮手
- 4. ASP.NET常被忽视的一些细节
- 5. 在.net中序列化读写xml方法的总结

随笔分类(88)

- Ado.net(8)
- Ajax(13)
- Asp.net(35)
- C#基础(7)
- ClownFish(4)
- DataBase(6)
- IIS(3)
- 其它杂类(8)
- 性能优化(4)

推荐排行榜

- 1. 写自己的ASP.NET MVC框架 (上) (756)
- 2. 用Asp.net写自己的服务框架(751)
- 3. 细说Cookie(674)
- 4. 我心目中的Asp.net核心对象(634)
- 5. 细说 ASP.NET Cache 及其高级用法(574)
- 6. 细说 Form (表单)(565)
- 7. ClownFish：比手写代码还快的通用数据访问层(561)
- 8. 细说ASP.NET Forms身份认证 (493)
- 9. 在.net中读写config文件的各种方法(422)
- 10. 在.net中序列化读写xml方法的总结(412)
- 11. Fish Li 的一年博客总结(386)
- 12. Fish Li 该如何帮助您呢？(377)
- 13. 选择HttpHandler还是 HttpModule？(371)

博客园 首页 联系 管理 订阅 XML

随笔 - 60 文章 - 0 评论 - 5770

细说 Form (表单)

阅读目录

- 开始
- 简单的表单，简单的处理方式
- 表单提交，成功控件
- 多提交按钮的表单
- 上传文件的表单
- MVC Controller中多个自定义类型的传入参数
- F5刷新问题并不是WebForms的错
- 以Ajax方式提交整个表单
- 以Ajax方式提交部分表单
- 使用jQuery，就不要再拼URL了！
- id, name 有什么关系
- 使用C#模拟浏览器提交表单
- 资源链接

Form (表单)对于每个WEB开发人员来说，应该是再熟悉不过的东西了，可它却是页面与WEB服务器交互过程中最重要的信息来源。虽然Asp.net WebForms框架为了帮助我们简化开发工作，做了很完美的封装，让我们只需要简单地使用服务端控件就可以直接操作那些 HTML表单元素了。但我认为了解一些基础的东西，可以使我们不必束缚在WebForms框架上，以及遇到一些奇怪问题时，可以更从容地解决它们。

今天，我将和大家来聊聊表单，这个简单又基础的东西。我将站在HTML和单纯的Asp.net框架的角度来解释它们的工作方式，因此，本文不演示WebForms服务器控件的相关内容。

[回到顶部](#)

简单的表单，简单的处理方式

好了，让我们进入今天的主题，看看下面这个简单的HTML表单。

```
<form action="Handler1.ashx" method="post" >
<p>客户名称: <input type="text" name="CustomerName" style="width: 300px"
/></p>
<p>客户电话: <input type="text" name="CustomerTel" style="width: 300px"
/></p>
<p><input type="submit" value="提交" /></p>
</form>
```

关注 Fish Li

565 0

(请您对文章做出评价)

- 14. HttpContext.Current并非无处不在(349)
- 15. Session，有没有必要使用它？(332)
- 16. 细说 HttpHandler 的映射过程(299)
- 17. 细说 Request[]与 Request.Params[](294)
- 18. 各种AJAX方法的使用比较(293)
- 19. 看懂SqlServer查询计划(285)
- 20. ASP.NET Page 那点事(278)
- 21. C#客户端的异步操作(276)
- 22. MongoDB实战开发【零基础学习，附完整Asp.net示例】(273)
- 23. 写自己的ASP.NET MVC框架（下）(267)
- 24. 细说ASP.NET的各种异步操作(256)
- 25. 解决ASP.NET中的各种乱码问题(252)

阅读排行榜

- 1. 写自己的ASP.NET MVC框架（上）(83216)
- 2. 细说 Form (表单)(78668)
- 3. 细说Cookie(67149)
- 4. 如何在IIS6,7中部署ASP.NET网站(63487)
- 5. 细说ASP.NET Forms身份认证(53439)
- 6. 用Asp.net写自己的服务框架(49947)
- 7. 细说 ASP.NET Cache 及其高级用法(49622)
- 8. MongoDB实战开发【零基础学习，附完整Asp.net示例】(48386)
- 9. ClownFish：比手写代码还快的通用数据访问层(47395)
- 10. Session，有没有必要使用它？(44628)
- 11. 写自己的ASP.NET MVC框架（下）(42876)
- 12. 我心目中的Asp.net核心对象(40388)
- 13. 在.net中读写config文件的各种方法(37834)
- 14. 细说ASP.NET Windows身份认证(35244)
- 15. 通用数据访问层及Ajax服务端框架的综合示例，展示与下载(35128)
- 16. C#客户端的异步操作(34149)
- 17. 看懂SqlServer查询计划(33943)
- 18. 细说 HttpHandler 的映射过程(31195)
- 19. 在.net中序列化读写xml方法的总结(31152)
- 20. HttpContext.Current并非无处不在

在这个HTML表单中，我定义了二个文本输入框，一个提交按钮，表单将提交到Handler1.ashx中处理，且以POST的方式。

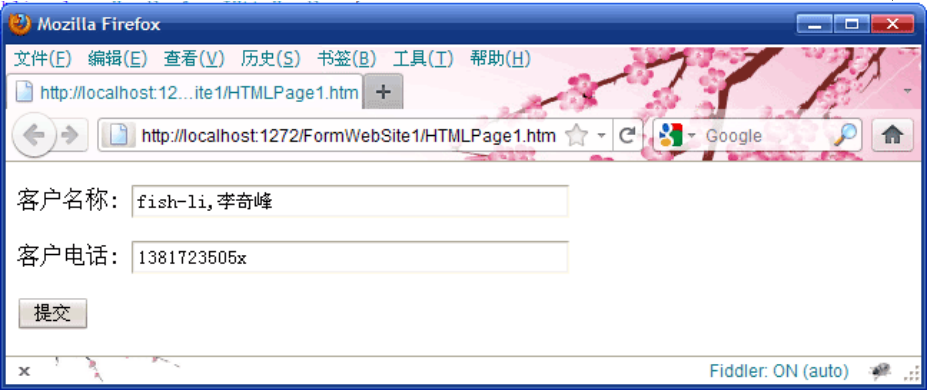
注意哦，如果我们想让纯静态页面也能向服务器提交数据，就可以采用这样方式来处理：将action属性指向一个服务器能处理的地址。

说明：当我们使用WebForms的服务器表单控件时，一般都会提交到页面自身来处理(action属性指向当前页面)，这样可以方便地使用按钮事件以及从服务器控件访问从浏览器提交的控件输入结果。

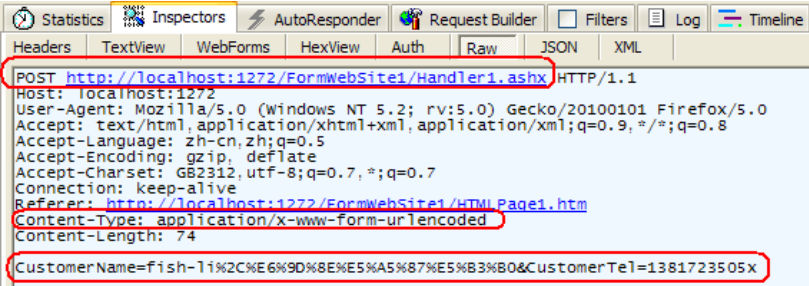
如果在URL重写时，希望能在页面回传时保持URL不变，即：action为重写后的URL，那么可以Page类中执行以下调用：

```
Form.Action = Request.RawUrl; // 受以下版本支持：3.5 SP1、3.0 SP1、2.0 SP1
```

好了，我们再回到前面那个HTML表单，看一下如果用户点击了“提交”按钮，浏览器是如何把表单的内容发出去的。在此，我们需要Fiddler工具的协助，请在提交表单前启动好Fiddler。我将这个表单的提交请求过程做了如下截图。

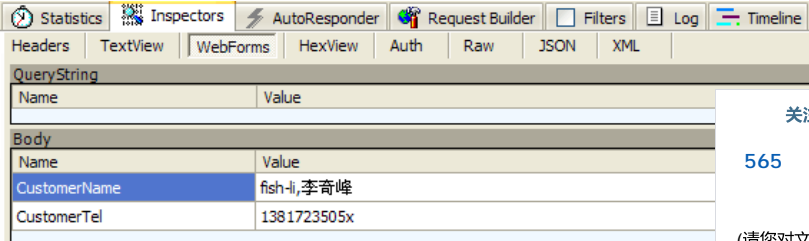


上图是将要提交的表单的输入情况，下图是用Fiddler看到的浏览器发出的请求内容。



在这张图片中，我们可以看到浏览器确实将请求发给了我前面在action中指定的地址，且以POST形式发出去的。表单的二个控件的输入值放在请求体中，且做了【编码】处理，编码的方式用请求头【Content-Type】说明，这样，当服务端收到请求后，就知道该如何读取请求的内容了。注意：表单的数据是以name1=value1&name2=value2 的形式提交的，其中name,value分别对应了表单控件的相应属性。

我们还可以在Fiddler中，将视图切换到WebForms选项卡，这样能更清楚地只查看浏览器提交的数据，如下图。



关注 Fish Li

5650

(请您对文章做出评价)

- 在(28564)
- 21. 细说ASP.NET的各种异步操作 (28504)
- 22. 解决ASP.NET中的各种乱码问题 (26300)
- 23. 细说 Request[]与 Request.Params[] (25309)
- 24. 选择HttpHandler还是 HttpModule ? (25075)
- 25. 各种AJAX方法的使用比较 (24290)
- 26. 晒晒我的通用数据访问层(23953)
- 27. ASP.NET Page 那点事(23566)
- 28. 晒晒我的Ajax服务端框架(23172)
- 29. ASP.NET页面优化，性能提升 8 倍的方法(22597)
- 30. MySql与SqlServer的一些常用用法的差别(21471)
- 31. 优化反射性能的总结（上） (21301)
- 32. ASP.NET程序也能像WinForm程序一样运行(19930)
- 33. IIS日志-网站运维的好帮手 (18795)
- 34. ASP.NET常被忽视的一些细节 (18386)
- 35. Fish Li 该如何帮助您呢？(18085)
- 36. 细说 ASP.NET控制HTTP缓存 (17110)
- 37. Fish Li 的一年博客总结(16256)
- 38. 【分享最爱的工具】专业源代码打包工具(15399)
- 39. 不修改代码就能优化ASP.NET网站性能的一些方法(15022)
- 40. 用ASP.NET写个SQLSERVER的小工具(14779)

最新评论

- 1. Re:优化反射性能的总结（上）  
楼主我使用Emit方式对属性取值，报错错误信息：Common Language Runtime detected an invalid program.委托定义:public delegate obj.....  
--扎伊尔天雕
- 2. Re:各种AJAX方法的使用比较  
\$.toJSON 是Jquery 自带的方法 还是 自己写的？  
--.Net菜鸟123
- 3. Re:各种AJAX方法的使用比较  
\$.toJSON  
--.Net菜鸟123
- 4. Re:我心目中的Asp.net核心对象  
轻松的语言，幽默的风格，传奇的故事，从一个个熟悉、不熟悉、似曾相识的术语中漫漫看到了ASP.NET的森林，让人豁然开朗，完全不同阅读其

看了客户端的页面和请求的内容，我们再来看看在服务端如何获取浏览器提交的表单的输入吧，代码如下：

```
string name = context.Request.Form["CustomerName"];
string tel = context.Request.Form["CustomerTel"];
```

代码很简单，直接根据表单控件的name属性访问Request.Form就可以了。

回到顶部

表单提交，成功控件

我们再来看一下浏览器是如何提交表单的，或者说，浏览器在提交表单时，要做哪些事情。

浏览器并不是将所有的表单控件全部发送到服务器的，而是会查找所有的【成功控件】，只将这些成功控件的数据发送到服务端，什么是成功控件呢？

简单来说，成功控件就是：每个表单中的控件都应该有一个name属性和“当前值”，在提交时，它们将以 name=value 的形式做为提交数据的一部分。

对于一些特殊情况，成功控件还有以下规定：

- 1. 控件不能是【禁用】状态，即指定【disabled="disabled"】。即：禁用的控件将不是成功控件。
- 2. 如果一个表单包含了多个提交按键，那么仅当用户点击的那个提交按钮才算是成功控件。
- 3. 对于checkbox控件来说，只有被用户勾选的才算是成功控件。
- 4. 对于radio button来说，只有被用户勾选的才算是成功控件。
- 5. 对于select控件来说，所有被选择的选项都做成功控件，name由select控件提供。
- 6. 对于file上传文件控件来说，如果它包含了选择的文件，那么它将是一个成功控件。

此外，浏览器不会考虑Reset按钮以及OBJECT元素。

注意：

- 1. 对于checkbox, radio button来说，如果它们被确认为成功控件，但没有为控件指定value属性，那么在表单提交时，将会以"on"做为它们的value
- 2. 如果在服务端读不到某个表单控件的值，请检查它是否满足以上规则。

提交方式：在前面的示例代码中，我为form指定了method="post"，这个提交方法就决定了浏览器在提交数据时，通过什么方式来传递它们。

如果是【post】，那么表单数据将放在请求体中被发送出去。

如果是【get】，那么表单数据将会追加到查询字符串中，以查询字符串的形式提交到服务端。

建议：表单通常还是以post方式提交比较好，这样可以不破坏URL，况且URL还有长度限制。

数据的编码：前面我将浏览器的请求细节用Fiddler做了个截图，从这个图中我们可以看到：控件输入的内容并不是直接发送的，而是经过一种编码规则来处理的。目前基本上只会只使用二种编码规则：

application/x-www-form-urlencoded 和 multipart/form-data，这两个规则的使用场景简单地讲就是：后者在上传文件时使用，其它情形则使用前者（默认）。

这个规则是在哪里指定的呢？其实form还有个enctype属性，用它就可以指定编码规则，当我在vs2008写代码时，会有以下提示：

```
<form action="Handler1.ashx" method="post" enctype=>
<p>客户名称: <input type="text" name="CustomerName" value="" />
<p>客户电话: <input type="text" name="CustomerTel" value="" />
<p><input type="submit" value="提交" /></p>
</form>
```

按照我前面说过的编码规则选择逻辑，application/x-www-form-urlencoded做为默认值，所以，一般情况下我们并不用显式指定。除非我们要上传文件了，那么此时必须设置 enctype="multipart/form-data"

好了，说了这么一大堆理论，我们再来看一下浏览是如何处理表单数据的。这个过程大致如下：

- 1. 识别所有的成功控件。
- 2. 为所有的成功控件创建一个数据集合，它们包含 control-name/current-value
- 3. 按照form.enctype指定的编码规则对前面准备好的数据进行编码。编码规则将放在

关注 Fish Li

565

0

(请您对文章做出评价)

他文档时只见一个个高深莫测的术语、只见树木不见森林的感觉。太强大了.....

--zhongzhy

5. Re:写自己的ASP.NET MVC框架 (上)

例如：现在有一个项目，二个团队A,B现在是不能让A团队人员使用B团队的代码（可以引用库文件），但是涉及到 controler 和view 这个如何划分与管理，有没有方案介绍？...

--阿超 -

6. Re:不修改代码就能优化ASP.NET 网站性能的一些方法  
学习了。大牛好久没更新文件了，！！

--冰点的爱

7. Re:细说 Form (表单)  
好文顶一下，收益很多

--耿会凤

8. Re:ASP.NET程序也能像WinForm 程序一样运行  
启用4.0后，不能将另存的配置文件名称改为Default.siteconfig,否则在启动的时候就会出现找不到FishAspnetLoader.resources文件的错误，导致无法启动。这样的话，就.....

--Doublejun

9. Re:看懂SqlServer查询计划  
好文章，转载了下，可以不？

--MR.lili

10. Re:细说ASP.NET Forms身份认证  
可以说说单点登录的实现机制吗

--快乐的灰太狼

11. Re:细说 ASP.NET Cache 及其高级用法  
不错的。

--robotman

12. Re:Fish Li 该如何帮助您呢？  
@自护意识哪有你这样面试的？...  
--梦想不在家！

13. Re:Session，有没有必要使用它？  
收了。必须使用。

--阿超 -

14. Re:Fish Li 该如何帮助您呢？  
楼主有没有兴趣点评一下《构建之法》我们想邀请一些有经验的工程师给软件工程的学生指导一下，像这个博客那样：...

--SoftwareTeacher

15. Re:解决ASP.NET中的各种乱码问题  
太感谢了 看了你的文章 解决了我的问题

--shilvyan

16. Re:ASP.NET程序也能像

【Content-Type】指出。

4. 提交编码后的数据。此时会区分post,get二种情况，提交的地址由form.action属性指定的。

[回到顶部](#)

## 多提交按钮的表单

用过Asp.net WebForms框架的人可能都写过这样的页面：一个页面中包含多个服务端按钮。处理方式嘛，也很简单：在每个按钮的事件处理器写上相应的代码就完事了，根本不用我们想太多。不过，对于不理解这背后处理过程的开发人员来说，当他们转到MVC框架下，可能会被卡住：MVC框架中可没有按钮事件！即使使用不用MVC框架，用ashx通用处理器的方式，也会遇到这种问题，怎么办？对于这个问题，本文将站在HTML角度给出二个最根本的解决办法。

**方法1：**根据【成功控件】定义，我们设置按钮的name，在服务端用name来区分哪个按钮的提交：

HTML代码

```
<form action="Handler1.ashx" method="post">
<p>客户名称: <input type="text" name="CustomerName" style="width: 300px" /></p>
<p>客户电话: <input type="text" name="CustomerTel" style="width: 300px" /></p>
<p><input type="submit" name="btnSave" value="保存" />
      <input type="submit" name="btnQuery" value="查询" />
</p>
</form>
```

服务端处理代码

```
// 注意：我们只要判断指定的name是否存在就可以了。
if( string.IsNullOrEmpty(context.Request.Form["btnSave"]) == false ) {
    // 保存的处理逻辑
}
if( string.IsNullOrEmpty(context.Request.Form["btnQuery"]) == false ) {
    // 查询的处理逻辑
}
```

**方法2：**我将二个按钮的name设置为相同的值（根据前面的成功控件规则，只有被点击的按钮才会提交），在服务端判断value，示例代码如下：

```
<form action="Handler1.ashx" method="post">
<p>客户名称: <input type="text" name="CustomerName" style="width: 300px" /></p>
<p>客户电话: <input type="text" name="CustomerTel" style="width: 300px" /></p>
<p><input type="submit" name="submit" value="保存" />
      <input type="submit" name="submit" value="查询" />
</p>
</form>
```

```
string action = context.Request.Form["submit"];
if( action == "保存" ) {
    // 保存的处理逻辑
}
if( action == "查询" ) {
    // 查询的处理逻辑
}
```

当然了，解决问题的方法很多，我们还可以在提交前修改form.action属性。对有些人会选择使用Filter的方式来处理。最终选择哪种方法，可根据各自喜好来选择。

关注 Fish Li

565

0

(请您对文章做出评价)

- WinForm程序一样运行  
可以提供下思路吗
- 涛褪荒芜
17. Re:看懂SqlServer查询计划  
请问下事例数据库在哪下载呢，链接不是啊
- 兰祯
18. Re:看懂SqlServer查询计划  
鲤鱼你好
- ICupid
19. Re:ClownFish：比手写代码还快的通用数据访问层  
能不能把源码放出来 让我们也学习学习
- yukling
20. Re:在.net中序列化读写xml方法的总结  
public class DynamicHelp{  
[XmlElement] public List Groups {  
get; set; } public Context Context {  
.....
- 萧亚生
21. Re:细说ASP.NET的各种异步操作  
学习了。
- 阿超 -
22. Re:MySql与SqlServer的一些常用用法的差别  
写的很好，有用到。
- 聆听雨点、 Kb
23. Re:写自己的ASP.NET MVC框架（上）  
说的很详细，学习了
- 大梦初醒
24. Re:细说Cookie  
受益匪浅！
- jia58960
25. Re:细说Cookie  
初学，有点看不懂，发现要学的东西太多了
- 孙苗青
26. Re:我心目中的Asp.net核心对象Mark
- Amarantin
27. Re:选择HttpHandler还是HttpModule？  
mark
- Amarantin
28. Re:C#客户端的异步操作  
很棒哦
- 时光未央岁月静好
29. Re:细说 ASP.NET Cache 及其高级用法  
学习了
- 轻狂 书生
30. Re:写自己的ASP.NET MVC框架（上）  
很好，很详细，学习了
- 拜月殿下

我可能更喜欢直接使用Ajax提交到一个具体的URL，这样也很直观，在服务端也就不用这些判断了。接着往下看吧。

[回到顶部](#)

## 上传文件的表单

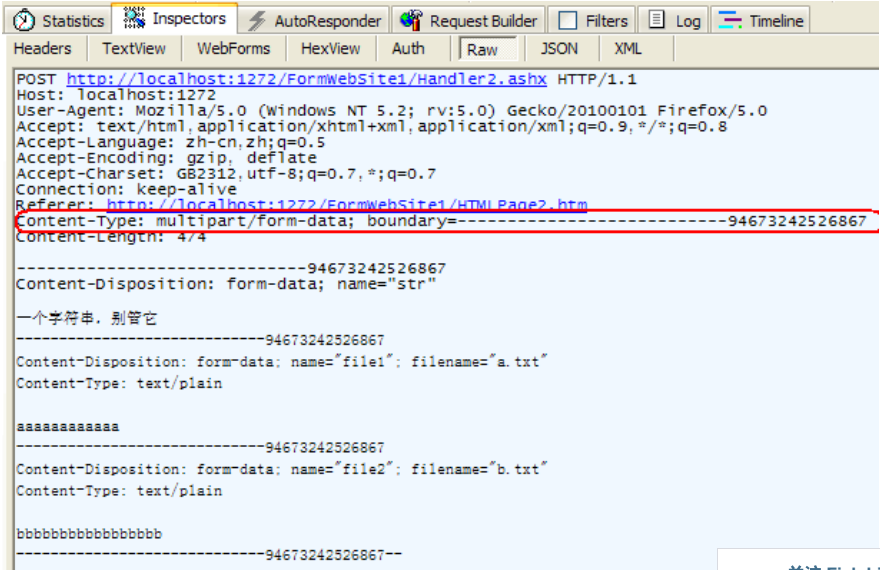
前面我说到“数据的编码”提到了form enctype，这个属性正是上传表单与普通表单的区别，请看以下示例代码：

```
<form action="Handler2.ashx" method="post" enctype="multipart/form-data">  
<p><input type="text" name="str" value="一个字符串，别管它" /></p>  
<p>要上传的文件 1<input type="file" name="file1" /></p>  
<p>要上传的文件 2<input type="file" name="file2" /></p>  
<p><input type="submit" value="提交" /></p>  
</form>
```

我将上传2个小文件



我们再来看看当我点击提交按钮时，浏览器发送的请求是个什么样子的：



注意我用红色边框框出来的部分，以及请求体中的内容。此时请求头Content-Type的值且还多了一个叫boundary的参数，它将告诉服务端：请求体的内容以这个标记来分开。每个分隔标记会单独占一行，且具体内容为：“--” + boundary，最后结束的分隔符 + boundary + “--” 也是独占一行。从图片中我们还可以发现，在请求体的每段数据

关注 Fish Li

5650

(请您对文章做出评价)



31. Re:看懂SqlServer查询计划  
那【Clustered Index Seek】、  
【Clustered Index Scan】、【Index  
Seek】、【Index Scan】、【Table  
Scan】这些查找方式中,如果要排序  
的话.....

--红色小内内

32. Re:写自己的ASP.NET MVC框架  
(下)  
楼主,能不能提供之前写的框架代  
码,让新手好入手看懂呀

--龙江&#183;谢

33. Re:MySql与SqlServer的一些常  
用用法的差别  
挺好,mark

--大漠孤烟~

34. Re:MongoDB实战开发【零基础  
学习,附完整Asp.net示例】  
写不很不错.很详细.

--葡萄糖

35. Re:Fish Li 的一年博客总结  
写的确实很好,学习了~~向你学习

--maxlin

36. Re:ClownFish:比手写代码还快  
的通用数据访问层  
很喜欢博主的文章,刚刚用豆约翰博  
客备份专家备份了您的全部博文.

--静夜\_思

37. Re:ClownFish:比手写代码还快  
的通用数据访问层  
很喜欢博主的文章,刚刚用豆约翰博  
客备份专家备份了您的全部博文

--静夜\_思

38. Re:写自己的ASP.NET MVC框架  
(上)  
我觉得.net4.0中的dynamic非常有  
意思.利用dynamic可以创造新的语  
法模式.可以看看,我觉得ORM框架  
也可以采用这种模式.在csharp中  
写的语句并不执行.而是做为模版  
经过二次编译再执行.....

--小鹿123

39. Re:写自己的ASP.NET MVC框架  
(下)  
在lz的文章里转来转去就是找不到  
ajax框架的代码下载的地方

--清风送明月

40. Re:解决ASP.NET中的各种乱  
码问题  
你好,大神,最近碰到了  
encodeURIComponent()与GB2312  
乱码问题这个问题,我是用了JQuery  
的serialize()序列化数据,看了你  
的文章学习很多,收获满满.但我  
是个菜鸟,你.....

--仔仔\_1992

Copyright ©2015 Fish Li

述信息。  
具体这些内容是如何生成的,可以参考本文后面的实现代码。

再来看看在服务端如何读取上传的文件。

```
HttpPostedFile file1 = context.Request.Files["file1"];
if( file1 != null && string.IsNullOrEmpty(file1.FileName) == false )
    file1.SaveAs(context.Server.MapPath("~/App_Data/") + file1.FileName);

HttpPostedFile file2 = context.Request.Files["file2"];
if( file2 != null && string.IsNullOrEmpty(file2.FileName) == false )
    file2.SaveAs(context.Server.MapPath("~/App_Data/") + file2.FileName);
```

或者

```
HttpFileCollection files = context.Request.Files;
foreach( string key in files.AllKeys ) {
    HttpPostedFile file = files[key];
    if( string.IsNullOrEmpty(file.FileName) == false )
        file.SaveAs(context.Server.MapPath("~/App_Data/") +
            file.FileName);
}
```

二种方法都行,前者更能体现控件的name与服务端读取的关系,后者在多文件上传时有更好的扩展性。

**安全问题:** 注意,上面示例代码中,这样的写法是极不安全的。正确的做法应该是:重新生成一个随机的文件名,而且最好能对文件内容检查,例如,如果是图片,可以调用.net的一些图形类打开文件,然后"另存"文件。总之,在安全问题面前只有一个原则:不要相信用户的输入,一定要检查或者转换。

[回到顶部](#)

## MVC Controller中多个自定义类型的传入参数

前面的所有示例代码中都有一个规律:在服务端读取浏览器提交的数据时,都会使用控件的name属性,基本上在Asp.net中就是这样处理。但是在MVC中,MS为了简化读取表单数据的代码,可以让我们直接在Controller的方法中直接以传入参数的形式指定,此时框架会自动根据方法的参数名查找对应的输入数据(当然也不止表单数据了)。下面举个简单的例子:

```
<form action="/Home/Submit" method="post">
<p>客户名称: <input type="text" name="Name" style="width: 300px" /></p>
<p>客户电话: <input type="text" name="Tel" style="width: 300px" /></p>
<p><input type="submit" value="提交" /></p>
</form>
```

Conntroller中的方法的签名:

```
public ActionResult Submit(Customer customer)
{
}

public ActionResult Submit(string name, string tel)
{
}
```

以上二种方法都是可以的,当然了,前者会比较好,但需要事先定义一个Customer类,

```
public class Customer
{
}
```

关注 Fish Li

565

0

(请您对文章做出评价)

```
public string Name { get; set; }

public string Tel { get; set; }
}
```

如果表单简单或者业务逻辑简单，我们或许一直也不会遇到什么麻烦，以上代码能很好的工作。但是，如果哪天我们有了新的业务要求，需要在这个表单中同时加上一些其它的内容，例如，要把业务员资料也一起录入进去。其中业务员的实体类定义如下：

```
public class Salesman
{
    public string Name { get; set; }

    public string Tel { get; set; }
}
```

Controller的接口需要修改成：

```
public ActionResult Submit(Customer customer, Salesman salesman)
{
}
```

这时，HTML表单又该怎么写呢？刚好，这二个类的（部分）属性名称一样，显然，前面表单中的Name,Tel就无法对应了。此时我们可以将表单写成如下形式：

```
<form action="/Home/Submit" method="post">
<p>客户名称: <input type="text" name="customer.Name" style="width: 300px" /></p>
<p>客户电话: <input type="text" name="customer.Tel" style="width: 300px" /></p>
<p>销售员名称: <input type="text" name="salesman.Name" style="width: 300px" /></p>
<p>销售员电话: <input type="text" name="salesman.Tel" style="width: 300px" /></p>
<p><input type="submit" value="提交" /></p>
</form>
```

注意Controller方法中的参数名与HTML表单中的name是有关系的。

[回到顶部](#)

## F5刷新问题并不是WebForms的错

刚才说到了MVC框架，再来说说WebForms框架。以前时常听到有些人在抱怨用WebForms的表单有F5的刷新重复提交问题。在此我想为WebForms说句公道话：这个问题并不是WebForms本身的问题，是浏览器的问题，只是如果您一直使用WebForms的较传统用法，是容易产生这个现象的。那么什么叫做【传统用法】呢？这里我就给我自己的定义吧：所谓的WebForms的传统用法是说：您的页面一直使用服务器控件的提交方式(postback)，在事件处理后，页面又进入再一次的重现过程，或者说：当前页面一直在使用POST方式向当前页面提交。

那么如何避开这个问题呢？办法大致有2种：

- 1. PRG模式(Post-Redirect-Get)，在事件处理后，调用重定向的操作Response.Redirect()，而不要在事件处理的后期再去给一些服务器控件绑定数据项了！  
建议：按钮事件只做一些提交数据的处理，将数据绑定的操作放在OnPreRender方法中处理，而不是它在每个事件中（遍地开花）。不过，这种方式下，可能伟大的ViewState就发挥不了大才了！您发现ViewState没用了，在Web.config中全局关掉后，又发现很多服务器控件的麻烦了！嗯，杯具有啊。  
这个话题说下去又没完没了，到此为止吧，不过，千万不要以为这种方法是在倒退哦。

关注 Fish Li

5650

(请您对文章做出评价)

2. 以Ajax方式提交表单，请继续阅读本文。

[回到顶部](#)

## 以Ajax方式提交整个表单

前面一直在说“浏览器提交表单”，事实上我们也可以用JavaScript提交表单，好处也有很多，比如前面所说的F5刷新问题。 以Ajax方式提交表单的更大好处它是异步的，还可以实现局部刷新，这些特性都是浏览器提交方式没有的。 前面我提到表单在提交时，浏览器要实现的4个步骤，基本上用JS来完成这个操作也是一样的。 但是，前面说的步骤好像很麻烦呢，有没有简单的方法来实现这个过程呢？ 嗯，有的，这里我将使用jQuery以及jquery.form.js这个插件来演示这个复杂过程的简单处理方案。

示例用的HTML表单还是我前面用的代码，完全不需要修改：

```
<form action="Handler1.ashx" method="post" >
<p>客户名称: <input type="text" name="CustomerName" style="width: 300px"
/></p>
<p>客户电话: <input type="text" name="CustomerTel" style="width: 300px"
/></p>
<p><input type="submit" value="提交" /></p>
</form>
```

JS代码如下：

```
$(function(){
    $('form').ajaxForm({
        success: function(responseText){
            alert(responseText);
        }
    });
});
```

是的，就是这么简单，只要调用ajaxForm()就行了。你也可以传入任何\$.ajax()能接受的参数。它的作用是：修改表单的提交方式，改成Ajax方式提交。**最终当用户点击“提交”按钮时**，此时不再是浏览器的提交行为了，而是使用Ajax的方式提交，提交的URL以及提交方法就是在FORM中指定的参数。

如果您希望要用用户点击某个按钮或者链接时，也能提交表单（不经过提交按钮），那么可以使用如下方法：

```
$(function(){
    $("#btnId").click(function(){
        $('form').ajaxSubmit({
            success: function(responseText){
                alert(responseText);
            }
        });
    });
});
```

变化很小，只需要将ajaxForm修改成ajaxSubmit就OK了。 与ajaxForm()不同，调用ajaxSubmit()方法将会立即提交表单。

[回到顶部](#)

## 以Ajax方式提交部分表单

在前面的示例中，我们看到以Ajax方式提交一个表单是非常容易的，它完全模拟了浏览过，有时我们可能需要只提交表单的一部分，为的是更好的局部更新，那么又该如何做呢？假如我有以下表单的一部分，我只希望在用户某个按钮时将它提交到服务端：

关注 Fish Li

565

0

(请您对文章做出评价)



```
<div id="divCustomerInfo">
<p>客户名称: <input type="text" name="CustomerName" style="width: 300px"
/></p>
<p>客户电话: <input type="text" name="CustomerTel" style="width: 300px"
/></p>
</div>
```

我们可以这样来提交这部分表单的数据：

```
$("#btnId").click(function(){
$.ajax({
url: "Handler1.ashx", type: "POST",
data: $('#divCustomerInfo :text').fieldSerialize(),
success: function(responseText){
alert(responseText);
}
});
return false;
});
```

注意关键的代码行：data: \$('#divCustomerInfo :text').fieldSerialize()

注意：此时将由您指定一个【jQuery选择器】来过滤要提交的控件，而不是使用成功控件的筛选逻辑。

或者，您也可以使用下面将要介绍的方法，仍然是使用 data: {} 的方式，但需要手工指定数据成员。

[回到顶部](#)

## 使用jQuery，就不要再拼URL了！

jQuery越来越流行，以至于在创建MVC项目时，VS IDE会把jQuery也准备好了，可能MS认为开发WEB项目离不开jQuery了。

的确，jQuery非常方便，尤其是在处理DOM时，不仅如此，在处理AJAX请求时，也非常方便。

不过，有件事却让我很纳闷：经常看到有人在使用jQuery实现Ajax时，把一堆参数放在URL中传递，当然了，发送GET请求嘛，这样做不错，但是，让我不解的是：URL是拼接起来的，而且代码又臭又长！

如果是一个简单的参数："aaa.aspx?id=" + xxId，这样也就罢了。但是当一堆参数拼接在一起时，可能一下子还看不清楚到底有几个什么样的参数。而且经验丰富一些的开发人员会发现这样做有时会有乱码问题，可能网上搜过后，知道还有编码的工作要处理，于是又加了一堆编码方法。到此为止，这段代码会让人看起来很累！

如果您平时也是这样做的，那么我今天就告诉您：不要再拼接URL了！\$.ajax()的参数不是有个data成员嘛，用它吧。看代码：

```
$.ajax({
url: "Handler1.ashx", type: "POST",
data: { id: 2, name: "aaa", tel: "~!@#$$%^&*()_+==<>|", xxxxx: "要多少
还可以写多少", encoding: "见鬼去吧。?& :)" },
success: function(responseText) {
$("#divResult").html(responseText);
}
});
```

你说什么，只能使用GET？哦，那就改一下 type 参数吧。

```
$.ajax({
url: "Handler1.ashx", type: "GET",
data: { id: 2, name: "aaa", tel: "~!@#$$%^&*()_+==<>|", xxxxx: "要多少
还可以写多少", encoding: "见鬼去吧。?& :)" },
success: function(responseText) {
$("#divResult").html(responseText);
}
```

关注 Fish Li

565 0

(请您对文章做出评价)

```
}  
});
```

看了这个示例，您还会继续拼URL吗？

说明：为了排版简单，我将参数放在一行了，建议实际使用时，不要挤在一行。

[回到顶部](#)

## id, name 有什么关系

通常我们在写HTML代码时，会给控件指定一个id属性，这个属性只供JS和CSS使用，在表单提交时，它不起任何作用。

在上面的示例代码中，可能data {}中的各个value就来源于各个不同的控件，那么为那些控件指定相应的id属性将会方便地找到它们。

但是如果不需要用JS和CSS控制的控件，或许它们只是用来显示一些数据（只读），那么就没有必要指定id属性，当然了，name属性也可以不用给出（避免提交无意义的数）。

[回到顶部](#)

## 使用C#模拟浏览器提交表单

浏览器也是一个普通的应用程序，.net framework也提供一些类也能让我们直接发起HTTP请求。今天我将再次用C#来模拟浏览器的提交请求，同时也可以加深对HTTP请求的理解。

示例代码分为二段，一段示范了使用application/x-www-form-urlencoded编码方式提交，另一段则示范了使用multipart/form-data的编码方式。

为了让大家能再次利用这些代码，我已将关键部分写成独立方法，希望当您有这方面的需求时能马上可以用上。代码如下：

### 1. application/x-www-form-urlencoded

```
/// <summary>  
/// 向指定的URL地址发起一个POST请求，同时可以上传一些数据项。  
/// </summary>  
/// <param name="url">要请求的URL地址</param>  
/// <param name="keyvalues">要上传的数据项</param>  
/// <param name="encoding">发送，接收的字符编码方式</param>  
/// <returns>服务器的返回结果</returns>  
static string SendHttpRequestPost(string url, Dictionary<string, string>  
keyvalues, Encoding encoding)  
{  
    if( string.IsNullOrEmpty(url) )  
        throw new ArgumentNullException("url");  
  
    string postData = null;  
    // 将数据项转变成 name1=value1&name2=value2 的形式  
    if( keyvalues != null && keyvalues.Count > 0 ) {  
        postData = string.Join("&",  
            (from kvp in keyvalues  
             let item = kvp.Key + "=" +  
HttpUtility.UrlEncode(kvp.Value)  
             select item  
            ).ToArray()  
        );  
    }  
  
    if( encoding == null )  
        encoding = Encoding.UTF8;
```

关注 Fish Li

565 0

(请您对文章做出评价)

```
HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);
request.Method = "POST";
request.ContentType = "application/x-www-form-urlencoded; charset="
+ encoding.WebName;

if( postData != null ) {
    byte[] buffer = encoding.GetBytes(postData);

    Stream stream = request.GetRequestStream();
    stream.Write(buffer, 0, buffer.Length);
    stream.Close();
}

using( WebResponse response = request.GetResponse() ) {
    using( StreamReader reader = new
StreamReader(response.GetResponseStream(), encoding) ) {
        return reader.ReadToEnd();
    }
}

// 调用上面方法的示例代码
string Test_SendHttpRequestPost()
{
    string url = "http://localhost:1272/FormWebSite1/Handler1.ashx";

    Dictionary<string, string> keyvalues = new Dictionary<string,
string>();
    keyvalues.Add("CustomerName", "我是李奇峰, $%+& ?#^/");
    keyvalues.Add("CustomerTel", "1381723505x");

    return SendHttpRequestPost(url, keyvalues, null);
}
```

2. **multipart/form-data** 。注意这部分代码有点复杂，因此我加了很多注释。

```
/// <summary>
/// 向指定的URL地址发起一个POST请求，同时可以上传一些数据项以及上传文件。
/// </summary>
/// <param name="url">要请求的URL地址</param>
/// <param name="keyvalues">要上传的数据项</param>
/// <param name="fileList">要上传的文件列表</param>
/// <param name="encoding">发送数据项，接收的字符编码方式</param>
/// <returns>服务器的返回结果</returns>
static string SendHttpRequestPost(string url, Dictionary<string, string>
keyvalues,
    Dictionary<string, string> fileList, Encoding encoding)
{
    if( fileList == null )
        return SendHttpRequestPost(url, keyvalues, encoding);

    if( string.IsNullOrEmpty(url) )
        throw new ArgumentNullException("url");

    if( encoding == null )
        encoding = Encoding.UTF8;

    HttpWebRequest request = (HttpWebRequest)WebRequest.Creat
request.Method = "POST"; // 要上传文件，一定要是POST方
```

关注 Fish Li

565 0

(请您对文章做出评价)

```
// 数据块的分隔标记, 用于设置请求头, 注意: 这个地方最好不要使用汉字。
string boundary = "-----" +
Guid.NewGuid().ToString("N");
// 数据块的分隔标记, 用于写入请求体。
// 注意: 前面多了一段: "--", 而且它们将独占一行。
byte[] boundaryBytes = Encoding.ASCII.GetBytes("\r\n--" + boundary +
"\r\n");

// 设置请求头。指示是一个上传表单, 以及各数据块的分隔标记。
request.ContentType = "multipart/form-data; boundary=" + boundary;

// 先得到请求流, 准备写入数据。
Stream stream = request.GetRequestStream();

if( keyvalues != null && keyvalues.Count > 0 ) {
    // 写入非文件的keyvalues部分
    foreach( KeyValuePair<string, string> kvp in keyvalues ) {
        // 写入数据块的分隔标记
        stream.Write(boundaryBytes, 0, boundaryBytes.Length);

        // 写入数据项描述, 这里的Value部分可以不用URL编码
        string str = string.Format(
            "Content-Disposition: form-data; name=\"{0}\"{1}\r\n\r\n{1}",
            kvp.Key, kvp.Value);

        byte[] data = encoding.GetBytes(str);
        stream.Write(data, 0, data.Length);
    }
}

// 写入要上传的文件
foreach( KeyValuePair<string, string> kvp in fileList ) {
    // 写入数据块的分隔标记
    stream.Write(boundaryBytes, 0, boundaryBytes.Length);

    // 写入文件描述, 这里设置一个通用的类型描述: application/octet-stream,
    具体的描述在注册表里有。
    string description = string.Format(
        "Content-Disposition: form-data; name=\"{0}\";
filename=\"{1}\"{1}\r\n" +
        "Content-Type: application/octet-stream\r\n\r\n",
        kvp.Key, Path.GetFileName(kvp.Value));

    // 注意: 这里如果不使用UTF-8, 对于汉字会有乱码。
    byte[] header = Encoding.UTF8.GetBytes(description);
    stream.Write(header, 0, header.Length);

    // 写入文件内容
    byte[] body = File.ReadAllBytes(kvp.Value);
    stream.Write(body, 0, body.Length);
}

// 写入结束标记
boundaryBytes = Encoding.ASCII.GetBytes("\r\n--" + bounda
"--\r\n");
stream.Write(boundaryBytes, 0, boundaryBytes.Length);

stream.Close();
```

关注 Fish Li

565

0

(请您对文章做出评价)

```
// 开始发起请求, 并获取服务器返回的结果。
using( WebResponse response = request.GetResponse() ) {
    using( StreamReader reader = new
StreamReader(response.GetResponseStream(), encoding) ) {
        return reader.ReadToEnd();
    }
}

// 调用上面方法的示例代码
string Test_SendHttpRequestPost2()
{
    string url = "http://localhost:1272/FormWebSite1/Handler2.ashx";

    Dictionary<string, string> keyvalues = new Dictionary<string,
string>();
    keyvalues.Add("Key1", "本示例代码由 Fish Li 提供");
    keyvalues.Add("Key2", "http://www.cnblogs.com/fish-li");
    keyvalues.Add("Key3", "来几个特殊字符：~!@#$$%^&*()-=_+{ }[]:;'\"<>?/., |
\\");

    Dictionary<string, string> fileList = new Dictionary<string,
string>();
    fileList.Add("file1", @"H:\AllTempFiles\ascx中文.gif");
    fileList.Add("file2", @"H:\AllTempFiles\asax中文.gif");

    return SendHttpRequestPost(url, keyvalues, fileList, Encoding.UTF8);
}
```

说明：上面的示例方法中，我并没有对KEY编码，是因为：我想大家选用的KEY应该是不需要编码的（英文字母与数字的组合）。  
而且，我也没加入对Cookie处理的那部分代码，如果您需要在发送请求时，保留Cookie，那么请参考我上一篇博客 [【细说Cookie】](#) 中的示例代码。

[回到顶部](#)

## 资源链接

- W3C Forms  
<http://www.w3.org/TR/html4/interact/forms.html>
- jquery.form.js  
<https://github.com/malsup/form>
- 更多的jquery.form演示  
<http://www.cnblogs.com/fish-li/archive/2011/05/02/2034010.html>

如果，您认为阅读这篇博客让您有些收获，不妨点击一下右下角的 [【推荐】](#) 按钮。  
如果，您希望更容易地发现我的新博客，不妨点击一下右下角的 [【关注 Fish Li】](#)。  
因为，我的写作热情也离不开您的肯定支持。

感谢您的阅读，如果您对我的博客所讲述的内容有兴趣，请继续关注我的后续博客，  
。


关注 Fish Li

5650

(请您对文章做出评价)



分类: [Asp.net](#)

绿色通道：[好文要顶](#) [关注我](#) [收藏该文](#) [与我联系](#) 



Fish Li

关注 - 5

粉丝 - 8009

荣誉：[推荐博客](#)  
[+加关注](#)

**推荐阅读：**★ [写自己的ASP.NET MVC框架（上）](#)     ★ [写自己的ASP.NET MVC框架（下）](#)  
**推荐阅读：**★ [ClownFish：比手写代码还快的通用数据访问层](#)     ★ [各种AJAX方法的使用比较](#)  
**我的资源：**★ [我的一些可供下载的资源列表](#)

« [上一篇：细说Cookie](#)  
» [下一篇：Session，有没有必要使用它？](#)

posted on 2011-07-17 21:15 [Fish Li](#) 阅读(78668) 评论(198) [编辑](#) [收藏](#)

[< Prev](#) [1](#) [2](#) [3](#) [4](#)

发表评论

#151楼[楼主] 2012-04-07 18:32 | Fish Li

@dreamhappy  
以下是我的回复，仅供参考：  
1. UTF-8  
2. 不科学。我认为网站只应该使用一种编码方式，所以你这样把问题搞复杂了。另外，如果是编码不匹配导致的乱码靠IsNullOrEmpty来判断也不靠谱。  

支持(0) 反对(0)

#152楼 2012-06-11 16:50 | 青石向晚

LZ看你写了这么多的东西真心佩服。  
我毕业前后是直接从.NET接触网络开发的，一开始就只会拖控件。后来逐渐学习了一些WEB开发本身的知识，但是还是感觉有很多东西会用，但是了解的不深。所以在这里，想请你帮我推荐一本书。怎样的书呢，是介绍原生态的WEB开发的书，不要ASP.NET的书。但不是那种介绍CSS的，就像这篇文章一样，对FORM啊INPUT之类HTML原生态的控件详细介绍的，对HTML本身的原理和知识做介绍的，我希望以后即使我不做.NET开发了，专做别的语言，也能对WEB本身的东西很了解。谢谢了。  

支持(1) 反对(0)

#153楼[楼主] 2012-06-11 18:14 | Fish Li

@青石向晚  
[引用](#)  
LZ看你写了这么多的东西真心佩服。  
想请你帮我推荐一本书。.....，就像这篇文章一样，对FORM啊INPUT之类HTML原生态的控件详细介绍的，.....  
  
我写这些东西没参考什么书，都是我自己的理解与总结啊。  
就实话，现在很怕见到这种要求，中午还有人发邮件给我，也是一样的要求。  
哎，头疼啊。  
  
我喜欢的书，你们未必会喜欢，你们需要的书，我未必认为有用。

#154楼 2012-06-11 20:41 | 青石向晚

@Fish Li  
好吧~不过你以后可以写篇文章介绍你喜欢的书哈哈~到时候我一定拜读~

支持(0) 反对(0)

[关注 Fish Li](#)

565 0

(请您对文章做出评价)

	支持(0) 反对(0)
<div>#155楼 2012-06-20 21:45   拉拉叟</div> <div>要是楼主能写篇介绍Fiddler技巧方面的文章那该多好啊</div>	支持(0) 反对(0)
<div>#156楼[楼主] 2012-06-21 12:31   Fish Li</div> <div>@拉拉叟 <a href="http://www.cnblogs.com/TankXiao/archive/2012/02/06/2337728.html">http://www.cnblogs.com/TankXiao/archive/2012/02/06/2337728.html</a></div>	支持(0) 反对(0)
<div>#157楼 2012-06-21 14:20   拉拉叟</div> <div>@Fish Li 呵呵 谢谢你的回复 笑脸</div>	支持(0) 反对(0)
<div>#158楼 2012-07-16 16:48   fycn01</div> <div>谢谢楼主，受益匪浅</div>	支持(0) 反对(0)
<div>#159楼 2012-08-23 16:08   咧嘴笑</div> <div>IT大哥，咋办，我感觉我做技术好吃力啊，本来技术就差，不知道从哪下手搞，我越来越懒了，不想创造，我是不是要完蛋了</div>	支持(0) 反对(0)
<div>#160楼 2012-09-23 20:25   纯粹的郭子</div> <div>看此文有点像玩Dota一样过瘾，看来我得玩两局！</div>	支持(0) 反对(0)
<div>#161楼 2012-10-16 14:57   息壤</div> <div>好文章。。。。。</div>	支持(0) 反对(0)
<div>#162楼 2012-11-23 11:37   依雪墨兰</div> <div>@Fish Li 很好的文章，特地登陆只为顶楼主</div>	支持(0) 反对(0)
<div>#163楼 2012-11-23 16:32   新手王</div> <div>学习中。。。</div>	支持(0) 反对(0)
<div>#164楼 2012-11-26 11:08   Gamain</div> <div>每次读楼主的文章都感觉原来自己以前什么都不知道！！！！</div>	

关注 Fish Li

5650

(请您对文章做出评价)

<div>#165楼 2012-11-30 19:35   目八双戈</div> <div>醍醐灌顶 谢谢哈</div> <div>支持(0) 反对(0)</div>
<div>#166楼 2012-12-03 14:46   Never too late</div> <div>引用dudu的话，过瘾啊，期待下篇~~哈哈</div> <div>支持(0) 反对(0)</div>
<div>#167楼[楼主] 2012-12-09 18:31   Fish Li</div> <div>@Never too late 兄弟啊，这是去年写的博客，它的下一篇早就写完了哦。</div> <div>支持(0) 反对(0)</div>
<div>#168楼 2012-12-10 10:43   Never too late</div> <div>@Fish Li 这个啊。没太注意。哈哈。</div> <div>支持(0) 反对(0)</div>
<div>#169楼 2012-12-24 15:22   hwtersha</div> <div>特地进来膜拜下，非常感谢这篇文章，让我学到了太多太多的东西。好多东西都是知道该这么用，看了这篇文章才找算是豁然开朗。如果刚学.net的时候看到这篇文章，能让我少走好多弯路，不过现在也不迟。 非常感谢这么细心耐心的博主</div> <div>支持(0) 反对(0)</div>
<div>#170楼[楼主] 2012-12-24 18:42   Fish Li</div> <div>@hwtersha 我也感谢你的评论。:)</div> <div>支持(0) 反对(0)</div>
<div>#171楼 2013-01-31 14:05   jinnsay</div> <div>受教了，写了很好，学到了很多。</div> <div>支持(0) 反对(0)</div>
<div>#172楼 2013-02-20 10:22   虫幻想</div> <div>受益匪浅 收藏了 谢谢 @Fish Li 浏览器提交表单的过程（包括关于成功控件的定义和提交的步骤），对不同的语言（比如Java）都通用吗</div> <div>支持(0) 反对(0)</div>
<div>#173楼 2013-04-06 19:17   cow-man</div> <div>这篇文章关注点在表单提交上面，将表单中提交相关的几个知识点都涵盖了，这篇文章学到两点：（1）表单提交相关知识点，包括成功控件，Content-Type，Jquery Ajax，HttpRequest。（2）由浅入深将这些知识点一个个说透。</div> <div></div>
<div>#174楼[楼主] 2013-04-06 21:01   Fish Li</div> <div>@cow-man</div> <div></div>

关注 Fish Li

5650

(请您对文章做出评价)

看来你也有收获啊。 :)	支持(0) 反对(0)
<div>#175楼 2013-04-07 00:02   Moriarty</div> <div>问一下哈，在使用 data: {} 的方式传入参数时，如何获取提交控件的中的值呢</div> <div>支持(0) 反对(0)</div>	
<div>#176楼 2013-04-30 22:32   Sky_lin</div> <div>fish兄，五一看了好几篇，最后都有代码下载的，为什么这篇没有呢？</div> <div>支持(0) 反对(0)</div>	
<div>#177楼[楼主] 2013-05-01 18:44   Fish Li</div> <div>@Moriarty 抱歉，一直忘记回复了。 你是说的jQuery的用法吧？如果是，那么这种方式就是key/value格式，与浏览器的提交格式一样，在服务端直接访问Request[] 就可以获取了。</div> <div>支持(0) 反对(0)</div>	
<div>#178楼[楼主] 2013-05-01 18:45   Fish Li</div> <div>@大侠酷裤马路 感谢评价。 :)</div> <div>支持(0) 反对(0)</div>	
<div>#179楼 2013-05-23 13:45   大侠酷裤马路</div> <div>真心感动，写的很好很详细</div> <div>支持(0) 反对(0)</div>	
<div>#180楼[楼主] 2013-05-24 20:26   Fish Li</div> <div>@大侠酷裤马路 多谢给予好评。 :)</div> <div>支持(0) 反对(0)</div>	
<div>#181楼 2013-08-02 17:12   淡淡然_然淡淡</div> <div>太他妈精辟了！跪拜！</div> <div>支持(0) 反对(0)</div>	
<div>#182楼[楼主] 2013-08-03 11:36   Fish Li</div> <div>@淡淡然_然淡淡 说话能不能文明一点！？</div> <div>支持(0) 反对(0)</div>	
<div>#183楼 2013-08-23 09:37   DoNetCoder</div> <div>大神，表单验证也讲点嘛 o(∩_∩)o</div> <div></div>	
<div>#184楼 2013-10-11 22:34   fulai_xy</div> <div></div> <div></div>	

关注 Fish Li

5650

(请您对文章做出评价)

- 好啊 厉害

支持(0) 反对(0)
- #185楼 2013-11-24 12:15 | fhlijys

非常感谢!!!写的太好了!!

支持(0) 反对(0)
- #186楼 2013-11-24 13:23 | Tony-Tse

mark，写得很全，受用了~

支持(0) 反对(0)
- #187楼 2013-11-24 16:31 | fhlijys

您好，看了您的文章，我知道怎么实现模拟提交了。如果我想提交后，直接在浏览器上显示收到的结果页面应该怎么做？

支持(0) 反对(0)
- #188楼 2013-11-28 11:27 | sdzhaodan

看了楼主的文章，让我重新燃起了学习的信心！

支持(0) 反对(0)
- #189楼 2014-01-03 11:51 | 康熙来乐

确实好文章，非常感谢楼主的知识分享！

支持(0) 反对(0)
- #190楼 2014-01-12 00:34 | pingzd

楼主一看就是很用心的人

支持(0) 反对(0)
- #191楼 2014-03-04 11:54 | uuexxe

一直在困惑，ajax既然可以提交数据，也可分post和get方法，为什么还要用form呢？虽然找到一些琐碎的理由，但总觉得不够充分

支持(0) 反对(0)
- #192楼 2014-04-12 16:28 | 月夜老K

@uuexxe  
这两者好像没有什么矛盾吧，在没有AJAX的时候都是用FORM，可能是习惯了吧，FORM的概念深入人心，你不用也行

支持(0) 反对(0)
- #193楼 2014-05-20 21:25 | willkins

虽说是我做Java的，但是看到这个还是挺有用的。

支持(0) 反对(0)

关注 Fish Li

5650

(请您对文章做出评价)



#194楼 2014-08-10 15:45 | 巴蒂尔

迫不及待想看了

支持(0) 反对(0)

#195楼 2014-09-18 16:31 | itjeff

受益匪浅！

支持(0) 反对(0)

#196楼 2014-10-16 22:23 | ㄟ(ˊoˋ)ㄎ

这么好的文章不顶一下过意不去啊！

支持(0) 反对(0)

#197楼 2015-01-27 14:11 | 面俱超人

以Ajax方式提交整个表单  
前面一直在说“浏览器提交表单”，事实上我们也可以用JavaScript提交表单，好处也有很多，比如前面所说的F5刷新问题。以Ajax方式提交表单的更大好处它是异步的，还可以实现局部刷新，这些特性都是浏览器提交方式没有的。前面我提到表单在提交时，浏览器要实现的4个步骤，基本上用JS来完成这个操作也是一样的。但是，前面说的步骤好像很麻烦呢，有没有简单的方法来实现这个过程呢？嗯，有的，这里我将使用jQuery以及jquery.form.js这个插件来演示这个复杂过程的简单处理方案。

示例用的HTML表单还是我前面用的代码，完全不需要修改：

<form action="Handler1.ashx" method="post" >  
<p>客户名称: <input type="text" name="CustomerName" style="width: 300px" /></p>  
<p>客户电话: <input type="text" name="CustomerTel" style="width: 300px" /></p>  
<p><input type="submit" value="提交" /></p>  
</form>

JS代码如下：

\$(function){  
\$('form').ajaxForm({  
success: function(responseText){  
alert(responseText);  
}  
});  
});  
  
还有回复么 这块没太看懂  
success: function(responseText){  
alert(responseText);  
} 这里还有什么作用呀action到Handler1.ashx就回不去原来的画面了，感觉就是POST过去的表单跟ajax没啥关系呀。

支持(0) 反对(0)

#198楼 2015-06-02 15:49 | 耿会凤

好文顶一下，收益很多

支持(0) 反对(0)

- 【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库
- 【推荐】融云即时通讯云 - 专注为 App 开发者提供IM云服务
- 【推荐】如何让你的程序拥有象Excel一样强大的数据编辑功能
- 【活动】RDS邀您6.5折体验PostgreSQL

关注 Fish Li

5650

(请您对文章做出评价)