[**使用jQuery.form插件，实现完美的表单异步提交**](http://www.cnblogs.com/heyuquan/p/form-plug-async-submit.html)

[传送门：异步编程系列目录……](http://www.cnblogs.com/heyuquan/archive/2013/04/26/3045827.html)

时间真快，转眼一个月快结束了，一个月没写博客了！手开始生了，怎么开始呢……

示例下载：[使用jQuery.form插件，实现完美的表单异步提交.rar](http://files.cnblogs.com/heyuquan/%E4%BD%BF%E7%94%A8jQuery.form%E6%8F%92%E4%BB%B6%EF%BC%8C%E5%AE%9E%E7%8E%B0%E5%AE%8C%E7%BE%8E%E7%9A%84%E8%A1%A8%E5%8D%95%E5%BC%82%E6%AD%A5%E6%8F%90%E4%BA%A4.rar)

抓住6月份的尾巴，今天的主题是

今天我想介绍的是一款jQuery的插件：[Jquery.form.js 官网](http://plugins.jquery.com/form/)。

通过该插件，我们可以非常简单的实现表单的异步提交，并实现文件上传、进度条显示等等。

现在我们从一个ASP.NET同步表单提交开始，然后再将其转化为异步的表单提交。我写了3种表单提交示例，并简单分析了各种方式的利弊。

当然主题还是使用jQuery表单插件轻松实现表单异步提交以及分析下该插件的源码。

ASP.NET服务器端控件实现同步表单提交

ASP.NET服务器控件最大特征就是标签包含ID和runat=”server”属性，在客服端页面内容中会输出类似<input type="hidden" name="\_\_VIEWSTATE" id="\_\_VIEWSTATE" value="/wEPDwUJNzgzNDMwNTMzZGR/8ZxOm5Tn0sfHNJaqE12hKqqJTQ==">标签，用于存储控件值数据。如下：

|  |
| --- |
| <form runat="server" id="server\_form" method=”post”>  <table border="1">  <tr>  <td>用户名：</td>  <td>  <asp:TextBox ID="txtLoginName" runat="server" AutoPostBack=”true” ></asp:TextBox>  </td>  </tr>  <tr>  <td colspan="2" style="text-align: center">  <asp:Button ID="btnSubmit" runat="server" Text="服务器控件同步提交模式" />  <asp:Button ID="btnUnSubmit" runat="server" OnClientClick="return false;" Text="不提交表单" />  </td>  </tr>  </table>  </form>  <asp:Label ID="labResponse" runat="server"></asp:Label> |

我们用ASP.NET服务器控件构建了一个表单，在ASP.NET页面中有这样限定：

1) 一页只能有一个**服务器端 Form** 标记，其他服务器端控件都在该表单中。

2) 页面中**服务器端Form**中任何导致页面回发的服务器端控件事件都会触发表单提交事件submit。比如：

a) 单击没有在OnClientClick事件中return false的服务器端按钮控件

b) 将AutoPostBack属性设置为true的TextBox、RadioButton、CheckBox、DropDownList等服务器端控件的值改动时都会触发页面回发。

c) 另外：type=”submit”的客服端标签<input type=”submit” />导致表单提交

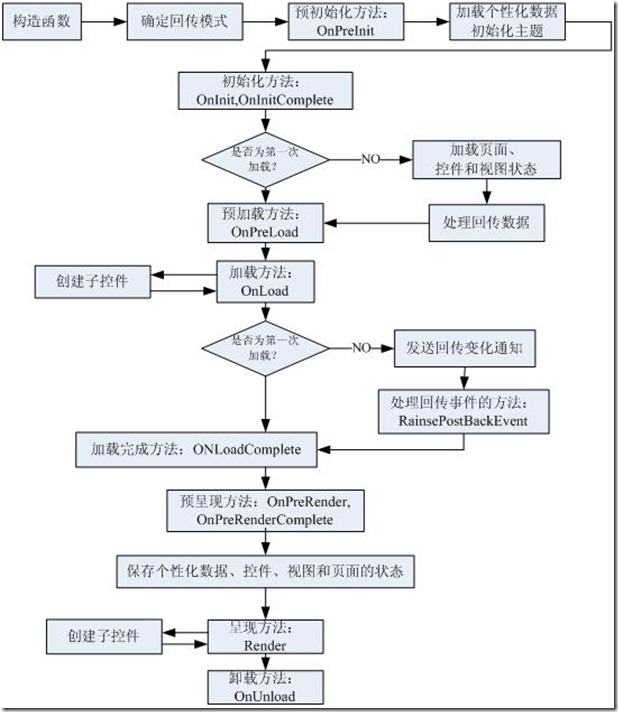
此方案优势：

1) 我们在后台可以非常轻易的获取服务器端控件的值，比如使用this. txtLoginName.Text访问控件的值或根据表单提交方式在this.Context.Request中获取表单元素值。

2) 我们在后台可以轻松设置页面服务器端控件的值，比如使用this. labResponse.Text = “表单提交成功”。

此方案劣势：

劣势很明显，效率低下，每一次导致的页面回发都会触发完整的ASP.NET页面生命周期，造成出现“白页”的情况。（更多描述请看：[ASP.NET编程模型之ASP.NET页面生命周期图解](http://developer.51cto.com/art/200908/141235.htm)）

[](http://images.cnitblog.com/blog/106337/201306/30213449-8de961087fd8487b97bf58576c6b17f0.jpg)

jQuery异步提交表单

现在我们已经意识到使用同步方式提交表单会造成出现“白页”的糟糕用户体验，那好，现在我使用上一篇分享的技术[《触碰jQuery：AJAX异步详解》](http://www.cnblogs.com/heyuquan/archive/2013/05/13/js-jquery-ajax.html)来将上面同步提交表单调整为异步提交表单的方式。

|  |
| --- |
| <form id="form1" method="post">  <table border="1">  <tr>  <td>用户名：</td>  <td>  <input type="text" name="loginName" /></td>  </tr>  <tr>  <td>爱 好：</td>  <td>  <input type="checkbox" name="cbLoveYy" value="1" />游泳  <input type="checkbox" name="cbLoveYx" value="1" />游戏  <input type="checkbox" name="cbLovePs" value="1" />爬山  </td>  </tr>  <tr>  <td colspan="2" style="text-align: center">  <input id="btnAjaxSubmit" type="submit" value="jQuery.ajax提交" />  </td>  </tr>  </table>  </form> |

jQuery提交代码如下：

|  |
| --- |
| <script type="text/javascript">  $(document).ready(function () {  $("#btnAjaxSubmit").click(function () {  var options = {  url: 'async\_submit\_test1.aspx?action=SaveUserInfo',  type: 'post',  dataType: 'text',  data: $("#form1").serialize(),  success: function (data) {  if (data.length > 0)  $("#responseText").text(data);  }  };  $.ajax(options);  return false;  });  });  </script> |

我们通过$("#form1").serialize()将表单元素的数据转化为字符串，然后通过$.ajax()执行异步请求资源。

方案：jQuery.ajax() + .aspx请求

此方案优势：

1) 我们不会感觉页面的“闪一闪”效果

2) 我们不会因为服务器耗时响应而导致出现“百页”的糟糕用户体验。

此方案劣势：

1) 此方案中我还是使用了aspx页面去响应请求，只是在后台通过action参数去做相应处理，尽管是异步操作但却完完整整的跑了一遍ASP.NET页面生命周期（这也是在Response.Write()输出完自己的东西后必须调用Response.End();来提前终止生命周期，否则页面信息也会一起返回）

2) jQuery库提供的序列化表单字符串方法不能收集文件上传的表单元素，如，$("#form1").serialize()。所以对于包含文件上传的表单我们还需通过<iframe>模拟异步表单提交。（<iframe>模拟异步表单提交的过程我将在分析jQuery.form插件的源码小节进行说明）

（jQuery库提供的序列化字符串的数据来源时表单的elements属性，而<input type=”file” />的表单元素不包含在elements中）

当然jQuery.ajax()也可以结合.ashx文件(一般处理文件)或其他方式实现高效异步请求，这边只是为了说明：异步请求aspx页面也会跑一边aspx页面生命周期的事实。

jQuery.form插件轻松实现表单提交

现在我们使用jQuery的表单插件[Jquery.form.js（官网）](http://plugins.jquery.com/form/)来实现异步表单提交。

1) 该插件需要Jquery最低版本为v1.5

2) 该插件提供了ajaxSubmit和ajaxForm两种表单提交方式，但不要对同一个表单同时使用两种方式。

现在我将通过“jQuery+jQuery.form插件+ashx(一般处理文件)”来实现一个高效的异步表单提交。

|  |
| --- |
| <form id="form1" action="ajaxOperation.ashx" method="post">  <table border="1">  <caption>jQuery.form.js异步提交方式</caption>  <tr>  <td>用户名：</td>  <td>  <input type="text" name="loginName" /></td>  </tr>  <tr>  <td colspan="2" style="text-align: center">  <button id="btnAjaxSubmit">ajaxSubmit提交</button>  &nbsp;  <input id="btnAjaxForm" type="submit" value="ajaxForm提交" />  </td>  </tr>  </table>  </form> |

1) 为<form>标签指定action值，指定使用ajaxOperation.ashx处理该表单请求。

2) 使用两个提交按钮btnAjaxSubmit和btnAjaxForm分别对应jQuery.form插件提供的两种表单提交API。

jQuery表单插件提交代码如下：

|  |
| --- |
| <script type="text/javascript">  $(document).ready(function () {  var options = {  success: function (data) {  $("#responseText").text(data);  }  };    // ajaxForm  $("#form1").ajaxForm(options);    // ajaxSubmit  $("#btnAjaxSubmit").click(function () {  $("#form1").ajaxSubmit(options);  });  });  </script> |

方案：jQuery.form.js插件 + .ashx请求

此方案优势：

1) 简简单单几句代码，我们就可以实现表单的提交，并且可灵活通过ajaxSubmit()函数基于任何事件的触发实现表单异步提交。

2) 支持文件上传功能，并在新浏览器中支持进度条更新。（在jQuery.form插件源码分析中会进行说明）

3) 与jQuery库完美结合，支持jQuery.ajax()函数触发的各种事件，支持jQuery.ajax()中所传递的参数。（在jQuery.form插件源码分析中会进行说明）

好了，这样短而易读的代码，这样的偷懒方式不正是我们追求的吗？那jQuery.form插件提供的表单提交API是否高效呢？内部又做了些什么？接下来跟着我看看jQuery.form插件内部实现吧。。。

jQuery.form插件源码分析

jQuery.form插件（[Jquery.form.js 官网](http://plugins.jquery.com/form/)），可以让我们非常简单的实现表单的异步提交、实现文件上传、进度条显示等等功能。

1. $(“form1”).ajaxSubmit(options)

1) ajaxSubmit是jQuery表单插件核心函数。非常灵活，因为它依赖于事件机制，只要有事件触发就能使用ajaxSubmit()提交表单，eg：超链接、图片、按钮的click事件。

2) options参数是

a) 一个函数，则为表单提交成功后调用的回调函数，即，options={success:function}。

b) options参数是一个集合，一个参数键值对

|  |  |
| --- | --- |
| 键名 | 描述 |
| type | （默认为表单的method属性值，若未设置取GET）  请求的类型，例如：POST、GET、PUT及PROPFIND。大小写不敏感。 |
| url | （默认取表单的action属性值，若未设置默认取window.location.href）  请求的URL地址，可以为绝对地址也可以为相对地址。 |
| data | （对象成员必须包含name和value属性）提供额外数据对象，通过$.param()函数返回序列化后的字符串，稍后会拼接到表单元素序列化的字符串之后。 |
| extraData | （此参数无需外部提供，由内部处理）  此参数是data在进行序列化成字符串之前的一个拷贝，只用于在表单包含<input type=”file” />并且是老浏览器。  因为在老浏览器中文件上传文件我们需要通过<iframe>来模拟异步提交，此时extraData会转变为<input type=”hidden” />元素包含在表单中，被一起提交到服务器。 |
| dataType | 一般不需自己设置。参数作用请看：[《jQuery.ajax()-dataType》](http://www.cnblogs.com/heyuquan/archive/2013/05/13/js-jquery-ajax.html#dataType) |
| traditional | 如果你想要用传统的方式来序列化数据，那么就设置为true。请参考[$.param()深度递归详解](http://www.update8.com/Manual/jquery/docs/jQuery.param/index.html)。 |
| delegation | （适用于ajaxForm）ajaxForm支持Jquery插件的委托方式（需要Jquery v1.7+），所以当你调用ajaxForm的时候其表单form不一定存在，但动态构建的form会在适当的时候调用ajaxSubmit。Eg：   |  | | --- | | $('#myForm').ajaxForm({  delegation: true,  target: '#output'  }); | |
| replaceTarget | （默认：false）与target参数共同起作用，True则执行replaceWirh()函数，false则执行html()函数 |
| target | 提供一个Html元素，在请求“成功”并且未设置dataType参数，则将返回的数据replaceWith()或html()掉对象原来的内容，再遍历对象调用success回调函数。   |  | | --- | | if (!options.dataType && options.target) {  var oldSuccess = options.success || function(){};  callbacks.push(function(data) {  var fn = options.replaceTarget ? 'replaceWith' : 'html';  $(options.target)[fn](data).each(oldSuccess, arguments);  });  } | |
| includeHidden | 在请求成功后，若设置执行clearForm()函数清空表单元素则会根据includeHidden设置决定如何清空隐藏域元素。  传递true，表示清空表单的所有隐藏域元素。  传递字符串，表示清空特殊匹配的隐藏域表单元素，eg：$('#myForm').clearForm('.special:hidden')，清空class属性包含special值的隐藏域 |
| clearForm | 请求成功时触发（同success），并用options. includeHidden做为回调函数参数。  回调函数：$form.clearForm(options.includeHidden); |
| resetForm | 请求成功时触发（同success）。  回调函数：$form.resetForm() |
| semantic | 布尔值，指示表单元素序列化时是否严格按照表单元素定义顺序。  在序列化只有<input type=”image” />元素会放在序列化字符串的最后，若semantic=true，则会按照它的定义顺序进行序列化。  若你服务器严格要求表单序列化字符串的顺序，则使用此参数进行控制。 |
| iframe | （默认：false）若有文件上传'input[type=file]:enabled[value!=""]'，指示是否应该使用<iframe>标签（在支持html5文件上传新特性的浏览器中不会使用iframe模式） |
| iframeTarget | 指定一个现有的<iframe>元素，否则将自动生成一个<iframe>元素以及name属性值。若现有的<iframe>元素没有设置name属性，则会自动生成一个name值 |
| iframeSrc | 为<iframe>元素设定src属性值 |
| 回调函数 | |
| beforeSerialize | 提供在将表单元素序列化为字符串之前，处理表单元素的回调函数。  签名：function(form,options)  函数说明：当前表单对象、options参数集合  返回值：返回false，表示终止表单提交操作。 |
| beforeSubmit | 提供在执行表单提交之前，处理数据的回调函数。  签名：function(a,form,options)  函数说明：通过formToArray(options.semantic, elements)返回的表单元素数组、当前表单对象、options参数集合  返回值：返回false，表示终止表单提交操作。 |

3) $(“form1”).ajaxSubmit(options) 内部直接或模拟jQuery.ajax(options)异步提交，所以也直接支持jQuery.ajax(options)所能处理的参数，并且支持jQuery.ajax(options)过程中所触发的[5个局部事件及6个全局事件](http://www.cnblogs.com/heyuquan/archive/2013/05/13/js-jquery-ajax.html#event)。

4) $(“form1”).ajaxSubmit(options) 内部将内部直接调用jQuery.ajax(options)返回的jqxhr对象或模拟的jqxhr对象进行了缓存，所以我们可以通过$(“#form1”).data(‘jqxhr’)获取到本次提交生成的jqxhr对象。

5) 更多jQuery.ajax()函数介绍请看：[《触碰jQuery：AJAX异步详解》](http://www.cnblogs.com/heyuquan/archive/2013/05/13/js-jquery-ajax.html)

ajaxSubmit函数处理流程：

1) 根据<form action=”” method=””>处理url、type参数以及success、iframeSrc等参数。

2) 触发beforeSerialize()回调函数

3) 序列化data参数和表单元素

4) 触发beforeSubmit()回调函数

5) 根据type参数处理options.data和options.url参数

6) 注册resetForm()和clearForm()回调函数

7) 注册将返回数据加载到options.target指定的元素上的回调函数

8) 注册success回调函数，若有options.target则循环该元素，并为每个子元素注册success回调函数

9) 处理<input type=”file” />文件上传元素

a) 不包含文件元素，直接调用jQuery.ajax()函数。

b) 包含文件元素，并且不支持XMLHttpRequest Level 2提供的文件上传新特性window.FormData。则通过**IFrame模拟表单异步提交**。

i. 调用fileUploadIframe()函数。

ii. 根据options. iframeTarget设置决定是创建一个<iframe>元素还是使用现有的<iframe>元素

iii. 模拟xhr对象以及jQuery.ajax()过程，以支持xhr对象返回和ajax事件触发

iv. 设置<form>的target指向<iframe>元素、encoding和enctype为“multipart/form-data”、method为”post”值等等

v. 处理options.extraData为<input type=”hidden” />元素并添加到<form>元素中。

vi. 调用<form>的submit()事件。（同步提交，但因为<form>的target指向<iframe>标签，所以刷新的是<iframe>中的内容，以此模拟异步提交）

c) 包含文件元素，并且支持XMLHttpRequest Level 2提供的新特性，则调用fileUploadXhr()函数，通过FormData()对象将数据传递给options.data参数，再调用jQuery.ajax(options)函数异步提交表单。并且XMLHttpRequest Level 2的新特性还支持进度条提示功能。（更多新特性请看：[《XMLHttpRequest Level 2 使用指南》](http://kb.cnblogs.com/page/157047/)）

10) 将内部jqxhr缓存起来，以供访问。$form.removeData('jqxhr').data('jqxhr', jqxhr);

11) 返回表单元素本身，以便符合jQuery的链式操作模式。

2. $(“form1”).ajaxForm(options)

是对$(“any”).ajaxSubmit(options)函数的一个封装，适用于表单提交的方式（注意，主体对象是<form>），会帮你管理表单的submit和提交元素（[type=submit],[type=image]）的click事件。在出发表单的submit事件时：阻止submit()事件的默认行为（同步提交的行为）并且调用$(this).ajaxSubmit(options)函数。

ajaxForm支持Jquery插件的委托方式（需要Jquery v1.7+），所以当你调用ajaxForm的时候其表单form不一定存在，ajaxSubmit将在适当的时候调用。Eg：

|  |  |
| --- | --- |
| 1  2  3  4 | $('#myForm').ajaxForm({  delegation: true,  target: '#output'  }); |

另外：如果你翻看原来码你可能会发现这样的绑定代码：.bind('submit.form-plugin', options, doAjaxSubmit)，即submit事件名后面有个”. form-plugin”。这是[jQuery事件命名空间](http://hi.baidu.com/jxbypp/item/87d26cd3043d871fd68ed033)语法，作用是方便事件的管理。

文件上传示例（被jQuery.form插件封装的相当简单，既然写了就也贴出来吧）

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | <form id="form1" action="ajaxOperation.ashx?Action=formUpload" method="post" enctype="multipart/form-data">  <table>  <tr>  <td>附件名字:</td>  <td>  <input type="text" name="fileName" /></td>  </tr>  <tr>  <td>附件:</td>  <td>  <input type="file" name="document" /></td>  </tr>  <tr>  <td colspan="2" style="align-content: center">  <input type="submit" value="模拟iframe提交表单" />  </td>  </tr>  </table>  </form>  <label id="responseText"></label> |

提交代码：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <script type="text/javascript">  $(function () {  var options = {  success: function (data) {  $("#responseText").text(data);  }  };    $("#form1").ajaxForm(options);  });  </script> |

3. $(“form1”).ajaxFormUnbind()

取消$(“”).ajaxForm(options)函数对指定表单绑定的submit和click事件。

4. $(“form1”).formToArray(semantic,elements)

序列化当前表单元素到一个数组中，每个数组元素都是包含name和value属性的对象。返回值是内部构件的一个数组元素，而elements参数将包含除<input type=”image”>以外的所有表单元素。

5. $(“form1”).formSerialize(semantic)

将表当前单元素序列化为字符串形式。

实现如下：

|  |  |
| --- | --- |
| 1  2  3 | $.fn.formSerialize = function(semantic) {  return $.param(this.formToArray(semantic));  }; |

6. $(“form1”).fieldSerialize(successful)

序列化包含name属性的表单元素为一个字符串。Successful参数标识是否获取type为reset、button、checkbox、radio、submit、image值得元素以及<select>的值。返回$(el).val()。

7. $(“form1”).fieldValue(successful) 或 $.fieldValue(element, successful)

获取指定表单中的表单元素或指定表单元素的值。Successful参数标识是否获取type为reset、button、checkbox、radio、submit、image值得元素以及<select>的值。返回$(el).val()。

8. $(“form1”).clearForm(includeHidden)

清空当前表单中input、select、textarea元素的值。includeHidden设置决定如何清空隐藏域元素。

a) 传递true，表示清空表单的所有隐藏域元素。

b) 传递字符串，表示清空特殊匹配的隐藏域表单元素，eg： $('#myForm').clearForm('.special:hidden')，清空class属性包含special值的隐藏域

9. $.(“form1”).clearFields(includeHidden) 和 $.(“form1”).clearInputs(includeHidden)

作用相同，清空当前表单中所有表单元素的指。includeHidden设置决定如何清空隐藏域元素。

a) 传递true，表示清空表单的所有隐藏域元素。

b) 传递字符串，表示清空特殊匹配的隐藏域表单元素，eg： $('#myForm').clearForm('.special:hidden')，清空class属性包含special值的隐藏域

10. $(“form1”).resetForm()

重置当前表单元素，导致所有表单元素重置到它的初始值。

11. $(“form1”).selected(select)

将当前表单元素中所有checkbox、radio设置为select。select参数为布尔值。

本文到此结束，通过此博文相信各位，

1. 再也不会去写龌龊的同步提交和使用aspx进行异步相应了。

2. 再也不用烦恼表单提交过程中各个控件的值如何获取的问题，并且通过jQuery.form表单插件轻松实现表单异步提交、文件上传及进度条显示。

3. 清楚的认识了jQuery.form表单在给我们提供便利的背后到底做了什么手脚(源码分析)。