[**详细解读Jquery函数：$.get(),$.post(),$.ajax(),$.getJSON()**](http://www.cnblogs.com/liuling/archive/2013/02/07/sdafsd.html)

一，$.get(url,[data],[callback])

说明：url为请求地址，data为请求数据的列表(是可选的，也可以将要传的参数写在url里面)，callback为请求成功后的回调函数，该函数接受两个参数，第一个为服务器返回的数据，第二个参数为服务器的状态，是可选参数。而其中，服务器返回数据的格式其实是字符串形势，并不是我们想要的json数据格式，在此引用只是为了对比说明。

1 $.get("data.php",$("#firstName.val()"),function(data){

2

3 　　$("#getResponse").html(data); }//返回的data是字符串类型

4

5 );

二，$.post(url,[data],[callback],[type])

说明：这个函数跟$.get()参数差不多，多了一个type参数，type为返回的数据类型，可以是html,xml,json等类型，如果我们设置这个参数为：json，那么返回的格式则是json格式的，如果没有设置，就 和$.get()返回的格式一样，都是字符串的。

[复制代码](javascript:void(0);)

1 $.post("emp.do?p=getAllEmp",{id:deptId,x:Math.random()},function(data){

2 var arry = eval("("+data+")");//去引号，将json字符串去引号编程json类型数组，也可以在$.post函数后面加一个参数"json"，指定接收的数据为json类型的

3 for(var i=0;i<arry.length;i++){

4 var op = new Option(arry[i].empName,arry[i].empId);

5 document.getElementById("emp").options.add(op);

6 }

7 });

[复制代码](javascript:void(0);)

也可以写成下面这样，返回的就是json类型数组了，就不要难过去引号了，可以直接遍历。

1 $.post("emp.do?p=getAllEmp",{id:deptId,x:Math.random()},function(arry){

2 for(var i=0;i<arry.length;i++){

3 var op = new Option(arry[i].empName,arry[i].empId);

4 document.getElementById("emp").options.add(op);

5 }

6 },"json");

三，$.ajax(opiton)

说明：$.ajax()这个函数功能强大，可以对ajax进行许多精确的控制，需要详细说明的请参照相关资料

[复制代码](javascript:void(0);)

1 $.ajax({

2 　　url: "ajax/ajax\_selectPicType.jsp",

3 　　data:{Full:"fu"},

4 　　type: "POST",

5 　　dataType:'json',

6 　　success:CallBack,

7 　　error:function(er){

8 　　BackErr(er);}

9 });

[复制代码](javascript:void(0);)

四，$.getJSON(url,[data],[callback])

说明:$.getJSON(url,[data],[callback])函数没有type参数，返回的是json类型的，不需要转换。

1 $.getJSON("dep.do?p=getAllDep",{x:Math.random()},function(arry){

2 for(var i=0;i<arry.length;i++){

3 var op = new Option(arry[i].deptName,arry[i].deptId);

4 document.getElementById("dep").options.add(op);

5 }

6 });

jquery的其他一些函数：

load( url, [data], [callback] ); 载入静态页面;  
url (String) 请求的HTML页的URL地址  
data (Map)(可选参数) 发送至服务器的 key/value 数据  
callback (Callback) (可选参数) 请求完成时(不需要是success的)的回调函数

load()方法可以轻松载入静态页面内容到指定jQuery对象。  
$('#ajax-div').load('data.html');

这样，data.html的内容将被载入到ID为ajax-div的DOM对象之内。你甚至可以通过制定ID来实现载入部分内容的Ajax操作，如：

$('#ajax-div').load('data.html#my-section');

实现GET和POST方法

get( url, [data], [callback] )  
url (String) 发送请求的URL地址.  
data (Map)(可选参数) 要发送给服务器的数据，以 Key/value 的键值对形式表示，会做为QueryString附加到请求URL中  
callback (Callback) (可选参数) 载入成功时回调函数(只有当Response的返回状态是success才是调用该方法)

很显然这是一个专门实现GET方式的函数，使用起来也相当的简单  
$.get('login.jsp', {  
   id      : 'Robin',  
   password: '123456',  
   gate    : 'index'  
  }, function(data, status) {  
   //data为返回对象，status为请求的状态  
   alert(data);  
   //此时假设服务器脚本会返回一段文字"你好，Robin！"，  
那么浏览器就会弹出对话框显示该段文字  
   alert(status);  
   //结果为success, error等等，但这里是成功时才能运行的函数  
  });

post( url, [data], [callback], [type] )

url (String) 发送请求的URL地址.  
data (Map)(可选参数) 要发送给服务器的数据，以 Key/value 的键值对形式表示  
callback (Callback) (可选参数) 载入成功时回调函数(只有当Response的返回状态是success才是调用该方法)  
type (String) (可选参数) 请求数据的类型，xml,text,json等

同样是jQuery提供的一个简便函数，其实用法  
   
$.post('regsiter.jsp', {  
   id:'Robin',  
   password: '123456',  
   type:'user'  
  },function(data, status) {  
   alert(data);  
  }, "json");

事件驱动的脚本载入函数：getScript()

getScript( url, [callback] )  
url (String) 待载入 JS 文件地址  
callback (Function) (可选) 成功载入后回调函数

getScript()函数可以远程载入JavaScript脚本并且执行。这个函数可以跨 域载入JS文件（神奇……？！）。这个函数的意义是巨大 的，它可以很大程度的缩减页面初次载入的代码量，因为你可以根据用户的交互来载入相应的JS文件，而不必在页面初始化的时候全部载入。  
   
$.getScript('ajaxEvent.js', function() {  
   alert("Scripts Loaded!");  
   //载入ajaxEvent.js，并且在成功载入后显示对话框提示。  
  });

构建数据通讯的桥梁：getJSON()

getJSON(url,[data],[callback])  
url (String) 发送请求地址  
data (Map) (可选) 待发送 Key/value 参数  
callback (Function) (可选) 载入成功时回调函数。

JSON是一种理想的数据传输格式，它能够很好的融合与JavaScript或其他宿主语 言，并且可以被JS直接使用。使用JSON相比传统的通过 GET、POST直接发送”裸体”数据，在结构上更为合理，也更为安全。至于jQuery的getJSON()函数，只是设置了JSON参数的 ajax()函数的一个简化版本。这个函数也是可以跨域使用的，相比get()、post()有一定优势。另外这个函数可以通过把请求url写 成”myurl?callback=X”这种格式，让程序执行回调函数X。

$.getJSON('feed.jsp',{  
   request: images,  
   id:      001,  
   size:    large  
   }, function(json) {  
    alert(json.images[0].link);  
    //此处json就是远程传回的json对象，假设其格式如下：  
    //{'images' : [  
    // {link: images/001.jpg, x :100, y : 100},  
    // {link: images/002.jpg, x : 200, y 200:}  
    //]};  
   }  
 );

更底层的ajax()函数  
虽然get()和post()函数非常简洁易用，但是对于更复杂的一些设计需求还是无法实现，比如在ajax发送的不同时段做出不同的动作等。jQuery提供一个更为具体的函数：ajax()。

ajax( options )  
ajax()提供了一大票参数，所以可以实现相当复杂的功能。

参数名 类型 描述   
url String (默认: 当前页地址) 发送请求的地址。   
type String (默认: “GET”) 请求方式 (“POST” 或 “GET”)， 默认为 “GET”。  
注意：其它 HTTP 请求方法，如 PUT 和 DELETE 也可以使用，但仅部分浏览器支持。   
timeout Number 设置请求超时时间（毫秒）。此设置将覆盖全局设置。   
async Boolean (默认: true) 默认设置下，所有请求均为异步请求。  
如果需要发送同步请求，请将此选项设置为 false。  
注意，同步请求将锁住浏览器，用户其它操作必须等待请求完成才可以执行。   
beforeSend Function 发送请求前可修改 XMLHttpRequest 对象的函数，如添加自定义 HTTP 头。

XMLHttpRequest 对象是唯一的参数。

function (XMLHttpRequest) {  
this; // the options for this ajax request  
}

   
cache Boolean (默认: true) jQuery 1.2 新功能，设置为 false 将不会从浏览器缓存中加载请求信息。   
complete Function 请求完成后回调函数 (请求成功或失败时均调用)。

参数： XMLHttpRequest 对象，成功信息字符串。

function (XMLHttpRequest, textStatus) {  
this; // the options for this ajax request  
}

   
contentType String (默认: “application/x-www-form-urlencoded”) 发送信息至服务器时内容编码类型。默认值适合大多数应用场合。   
data Object,  
String 发送到服务器的数据。将自动转换为请求字符串格式。GET 请求中将附加在 URL 后。  
查看 processData 选项说明以禁止此自动转换。必须为 Key/Value 格式。  
如果为数组，jQuery 将自动为不同值对应同一个名称。  
如 {foo:["bar1", "bar2"]} 转换为 ‘&foo=bar1&foo=bar2′。   
dataType String 预期服务器返回的数据类型。如果不指定，jQuery 将自动根据 HTTP 包 MIME 信息  
返回 responseXML 或 responseText，并作为回调函数参数传递，可用值:

“xml”: 返回 XML 文档，可用 jQuery 处理。

“html”: 返回纯文本 HTML 信息；包含 script 元素。

“script”: 返回纯文本 JavaScript 代码。不会自动缓存结果。

“json”: 返回 JSON 数据 。

“jsonp”: JSONP 格式。使用 JSONP 形式调用函数时，

如 “myurl?callback=?” jQuery 将自动替换 ? 为正确的函数名，以执行回调函数。  
   
error Function (默认: 自动判断 (xml 或 html)) 请求失败时将调用此方法。

这个方法有三个参数：XMLHttpRequest 对象，错误信息，（可能）捕获的错误对象。

function (XMLHttpRequest, textStatus, errorThrown) {  
// 通常情况下textStatus和errorThown只有其中一个有值  
this; // the options for this ajax request  
}

   
global Boolean (默认: true) 是否触发全局 AJAX 事件。设置为 false 将不会触发全局 AJAX 事件，

如 ajaxStart 或 ajaxStop 。可用于控制不同的Ajax事件  
   
ifModified Boolean (默认: false) 仅在服务器数据改变时获取新数据。

使用 HTTP 包 Last-Modified 头信息判断。  
   
processData Boolean (默认: true) 默认情况下，发送的数据将被转换为对象(技术上讲并非字符串)

以配合默认内容类型 “application/x-www-form-urlencoded”。

如果要发送 DOM 树信息或其它不希望转换的信息，请设置为 false。  
   
success Function 请求成功后回调函数。这个方法有两个参数：服务器返回数据，返回状态

function (data, textStatus) {  
// data could be xmlDoc, jsonObj, html, text, etc...  
this;  
 // the options for this ajax request  
}

你 可以指定xml、script、html、json作为其数据类型，可以为beforeSend、error、sucess、complete等状态设置 处理函数，众多其它参数也可以订完完全全定义用户的Ajax体验。下面的例子中，我们用ajax()来调用一个XML文档：

$.ajax({  
    url: 'doc.xml',  
    type: 'GET',  
    dataType: 'xml',  
    timeout: 1000,  
    error: function(){  
        alert('Error loading XML document');  
    },  
    success: function(xml){  
        alert(xml);  
  //此处xml就是XML的jQuery对象了，你可以用find()、next()或XPath等方法在里面寻找节点，  
和用jQuery操作HTML对象没有区别  
    }  
});

进一步了解AJAX事件  
前面讨论的一些方法都有自己的事件处理机制，从页面整体来说，都只能说是局部函数。jQuery提供了AJAX全局函数的定义，以满足特殊的需求。下面是jQuery提供的所有函数（按照触发顺序排列如下）：

ajaxStart  
(全局事件) 开始新的Ajax请求，并且此时没有其他ajax请求正在进行  
beforeSend  
(局部事件) 当一个Ajax请求开始时触发。如果需要，你可以在这里设置XMLHttpRequest对象  
ajaxSend  
(全局事件) 请求开始前触发的全局事件  
success  
(局部事件) 请求成功时触发。即服务器没有返回错误，返回的数据也没有错误  
ajaxSuccess  
全局事件全局的请求成功  
error  
(局部事件) 仅当发生错误时触发。你无法同时执行success和error两个回调函数  
ajaxError  
全局事件全局的发生错误时触发  
complete  
(局部事件) 不管你请求成功还是失败，即便是同步请求，你都能在请求完成时触发这个事件  
ajaxComplete  
全局事件全局的请求完成时触发  
ajaxStop  
(全局事件) 当没有Ajax正在进行中的时候，触发  
局部事件在之前的函数中都有介绍，我们主要来看看全局事件。对某个对象进行全局事件监听，那么全局中的AJAX动作，都会对其产生影响。比如，当页面在进行AJAX操作时，ID为”loading”的DIV就显示出来：

$("#loading").ajaxStart(function(){  
   $(this).show();  
 });

全局事件也可以帮助你编写全局的错误相应和成功相应，而不需要为每个AJAX请求独立设置。有必要指出，全局事件的参数是很有用的。除了 ajaxStart、ajaxOptions，其他事件均有event, XMLHttpRequest, ajaxOptions三个参数。第一个参数即事件本身；第二个是XHR对象；第三个是你传递的ajax参数对象。在一个对象里显示全局的AJAX情况：

$("#msg").beforeSend(function(e,xhr,o) {  
 $(this).html("正在请求"+o.url);  
 }).ajaxSuccess(function(e,xhr,o) {  
 $(this).html(o.url+"请求成功");  
 }).ajaxError(function(e,xhr,o) {  
 $(this).html(o.url+"请求失败");  
});

很显然，第三个参数也可以帮助你传递你在AJAX事件里加入的自定义参数。 在单个AJAX请求时，你可以将global的值设为false，以将此请求独立于AJAX的全局事件。

$.ajax({  
   url: "request.jsp",  
   global: false,  
   // 禁用全局Ajax事件.  
 });

如果你想为全局AJAX设置参数，你会用上ajaxSetup()函数。例如，将所有AJAX请求都传递到request.php，；禁用全局方法；强制用POST方法传递：

$.ajaxSetup({  
  url: "request.jsp",  
  global: false,  
  type: "POST"  
});

一些你不得不知道的方法  
写AJAX肯定离不开从页面获取相应的值。在这里简单列举一些方法：

val()   
val()函数可以返回表单组建的值，例如任何种类input的值。配合选择符操作，你可以轻易获取选项组、输入框、按钮等元素的值。

$("input[name='info']:text").val();  
//返回名字为info的文本框的值  
$("input[name='pass']:password").val();  
//返回名字为pass的密码框的值  
$("input[name='save']:radio").val();  
//返回名字为save的单选项的值  
//以此类推

serialize()

serialize函数可以帮你把表单对象的所有值都转换为字符串序列。如果你要写GET格式的请求，这个就非常方便了。  
serializeArray()  
和serialize()类似，只不过它返回的是JSON对象。