1. Web2.0的趋势

内容的聚合，mashup

1. 浏览器的限制

由于受到浏览器的限制，Ajax不允许跨域通信。如果尝试从不同的域请求数据，会出现安全错误。如果能控制数据驻留的远程服务器并且每个请求都前往同一域，就可以避免这些安全错误。但是，如果仅停留在自己的服务器上，Web 应用程序还有什么用处呢？如果需要从多个第三方服务器收集数据时，又该怎么办？

说了那么多，根本意思就是通过ajax的形式去调用其他网站的接口是不行的。

例如：

$.ajax({

url: "http://www.google.com/search?q=jquery",

success: function(json){

alert(json.count)

}

});

1. 解决方法

为了在页面当中整合第三方服务器的数据，有那么几种方法：

1. 克服该限制的一个相对简单的方法是让 Web 页面向它源自的 Web 服务器请求数据，并且让 Web 服务器像代理一样将请求转发给真正的第三方服务器。尽管该技术获得了普遍使用，但它是不可伸缩的。

我的理解：在后台代码中实现对第三方服务器的请求(API调用之类的)，因为后台代码不受跨域限制。对其中所说的“不可伸缩”不太理解

1. 另一种方式是使用框架要素在当前 Web 页面中创建新区域，并且使用 GET 请求获取任何第三方资源。不过，获取资源后，框架中的内容会受到同源策略的限制。

我的理解：就是使用iframe嵌套页面，但嵌套的页面同样会受到跨域的限制

1. 第三种方法就是“JSONP”，全称是“JSON with Padding”，padding的中文意思是“填充”，之后我们会明白“填充”的意义。
2. JSONP原理

JSONP需要完成的任务包括两个方面：

* 获取第三方服务器上的数据；
* 通过本地的JS代码对第三方的数据进行处理和渲染

浏览器虽然限制了Ajax的跨域通信，但允许在页面中插入动态的脚本元素。简单的讲就是从第三方服务器加载js代码是可行的，例如：

<script type="text/javascript" src="http://www.google.com/test.js"></script>

注释：但加载的js代码同样被视作是从当前域加载的，所以想在第三方的js代码中进行对第三方服务器的ajax调用同样是不行的。

所以可以通过第三方服务器生成动态的js代码来回调本地的js方法，而方法中的参数则由第三方服务器在后台获取，并以JSON的形式填充到JS方法当中，这也就是“JSON with Padding”中“padding”的真正意义。

应用过程当中，请求方（本地）向第三方服务器请求动态JS脚本，并将获取数据后需要回调的函数名以约定好的参数名（如callback等）发送给第三方服务器。

第三方服务器需要为JSONP请求开发相应的API，API中先获取JSONP请求需要的数据，然后以JSON的形式封装，再与请求方的回调函数名拼接在一起，动态生成请求方需要调用的JS代码。

用 Java servlet 实现的 JSONP 服务示例代码如下：

protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

//获取JSON数据

String jsonData = getDataAsJson(req.getParameter("symbol"));

//获取回调函数名

String callback = req.getParameter("callback");

//拼接动态JS代码

String output = callback + "(" + jsonData + ");

resp.setContentType("text/javascript");

PrintWriter out = resp.getWriter();

out.println(output);

// 输出为 jsonpFunc({"symbol" : "IBM", "price" : "91.42"});

}

假设这个JSONP服务的URL为http://www.google.com/jsonp，回调的函数名为jsonpFunc，那么可以这样发送JSONP请求：

<script type="text/javascript"

src="http://www.google.com/jsonp&company=IBM&callback=callback"></script>

其中的company=IBM为查询数据所需要传递的参数

附上一个**可用的实例**：[](jsonp.html)

但实际的开发中，JS库中一般都包含了更便于使用的JSONP方法，例如jquery和kissy。

以jquery为例，jsonp的调用形式如下

jQuery.getJSON(

"http://www.yourdomain.com/jsonp/ticker?symbol=IBM&callback=?",

function(data) {

alert("Symbol: " + data.symbol + ", Price: " + data.price);

}

);

其中回调函数名”callback”为”?”，即不需要用户指定，而是由jquery生成，互调函数也不需要单独定义，而是以参数的形式紧接在URL之后，URL中还可以附带供数据查询用的其他参数，如上例中的”symbol=IBM”

淘宝的前端JS库貌似是KISSY，附上KISSY中JSONP方法的文档地址：<http://docs.kissyui.com/kissy/docs/ajax/index.html#method_jsonp>

附上一个**可用的实例**：[](jquery_jsonp.html)

两个实例中请求的JSONP服务的URL均为：

<http://www.geonames.org/postalCodeLookupJSON?postalcode=10504&country=US&callback>=?

可以把callback参数先去掉，在浏览器中看一下返回的json数据。

1. 总结
2. 跨域的ajax请求无法实现
3. 浏览器允许动态js脚本注入
4. JSONP服务需要第三方提供相应的JSONP服务
5. Jquery和kissy已经为我们做了很多
6. 参考文档

<http://www.ibm.com/developerworks/cn/web/wa-aj-jsonp1/index.html>