目录

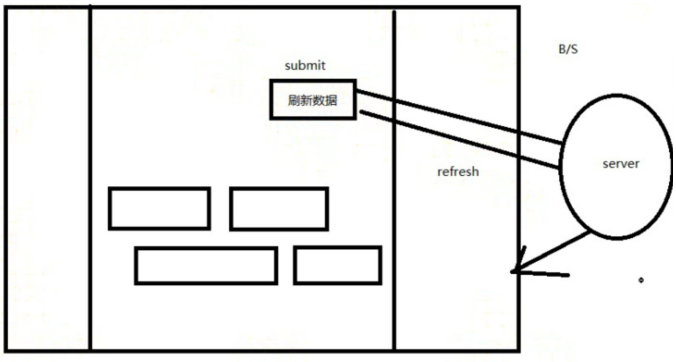
Ajax 跨域难题 - 原生 JS 和 jQuery 的实现对比

jQuery JavaScript ajax

- 讲解顺序：
  - AJAX 的概念及由来
  - JS 和 jQuery 中的 ajax
  - 浏览器机制
  - AJAX 跨域

AJAX 的概念

- 在讲解 AJAX 的概念之前，我先提一个问题。



这是一个典型的 B/S 模式。

- PS. B/S结构（Browser/Server，浏览器/服务器模式），是WEB兴起后的一种网络结构模式，WEB浏览器是客户端最主要的应用软件。这种模式统一了客户端，将系统功能实现的核心部分集中到服务器上，简化了系统的开发、维护和使用。客户机上只要安装一个浏览器

- B/S 模式

  - 优点：一个服务器可以响应多个客户端
  - 缺点：短链接（与 C/S 相反）
  - 特点：客户端主动去请求（request），服务器端被迫去响应（response）`这一点很重要`
  - 好处：大大减少了服务器的资源
  - 结论：服务器永远没办法主动告诉客户端一些信息

在这张图中，左侧是菜单，很早之前的做法是点击刷新按钮会对整个页面进行刷新，这样做的弊端是什么？如何不使用 AJAX 进行局部刷新?

- 弊端
  - I/O 网络吞吐量过大
  - 网页加载慢
  - 耗内存及 CPU
- 局部刷新
  - 很 low，但是只有唯一的方法：iframe
  - 虽然是局部刷新，但是是局部整体刷新

那么有什么办法能进行数据实时推送（例如股票价格）？

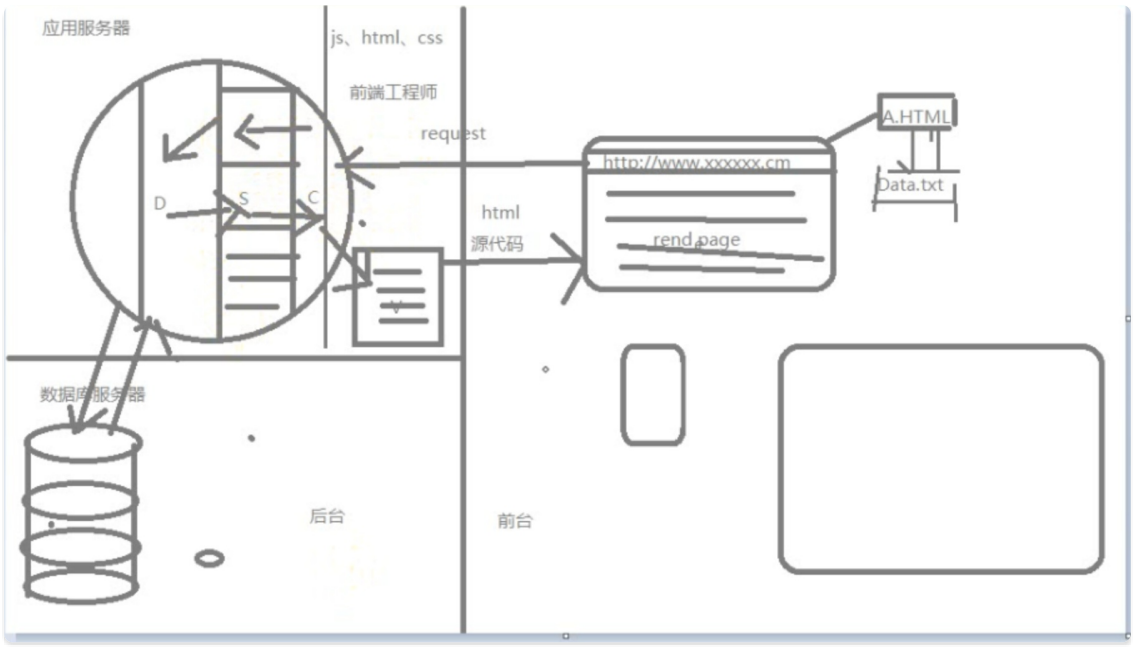
于是诞生了这几个方法。

- 轮询机制（依旧很占服务器资源，low）
- 服务器推送（提一下，想了解的下次，不是这次要讲的东西）

那么，iframe 最大的缺点是什么？

- 外面的 window 和 iframe 的 window 不是一个对象，也就是说，不在一个 DOM 树中！这样的话，外面内容和里面内容交互起来特别繁琐。
- AJAX 解决了这个问题。

了解 AJAX 之前先要了解一下前后端的交互过程。



A1-2

太多口述。。。

JS 和 jQuery 中的 ajax

- AJAX 全称：即“Asynchronous JavaScript And XML”（代表一种格式，比如：json、HTML、XML、text、jsonp等等）
- AJAX 的核心对象是 XMLHttpRequest 对象
- 查看方法（就是 network 里的 XHR）
- IE7及其他浏览器：XMLHttpRequest
- IE6及以下：ActiveXObject

```
// 简单的ajax请求
var xhr = new XMLHttpRequest();// 创建
xhr.onreadystatechange = function() {// onreadystatechange 不是检测方法，而是状态改变后更新的状态
    if (xhr.status == 200) {
        /*
        普及一下常用的状态码
        200：ok，服务器成功返回数据
        400：Bad Request，语法错误
        401：请求需要认证
        404：not found，找不到页面
        500：服务器遇到意外错误
        503：服务器正在维护或者过载无法完成请求
        其他百度去
        */
        if (xhr.readyState == 4) {
```

```
        /*
        xhr.readyState 是 ajax 状态
        普及一下这个4是什么意思:
        0: 创建服务
        1: 打开服务
        2: 发送服务
        3: 服务器响应
        4: 加载成功
        */
            var data = JSON.parse(xhr.responseText);// responseText 是返回的文本或对象
        }
    } else {
        // 如果不是正常返回
        console.log("数据返回失败！状态码" + xhr.status + "状态信息：" + xhr.statusText)
        // xhr.statusText是浏览器的错误信息，因为跨浏览器的时候，可能不太一致，不建议直接使用它
    }
}
xhr.open("post", "list.json?rand="+new Date(), true);// 打开，最后一个 bool 代表是否异步，这一步仅仅只配置了 ajax 的基本信息，而并没有对服务器请求
// rand=new Date()是为了让每一次请求 url 都不同，用来区分缓存
xhr.setRequestHeader("Content-Type","application/www-x-form-urlencoded");
xhr.send(null);// 向服务器发送请求
```

jQuery 的 ajax 实际上就是封装了上面的代码

```
$.ajax({
    type:"get",
    dataType: "json",
    url: url,
    async: true,
    success: function(){},
    error: function(){}
})


下面用 chrome 和 Firefox 来演示跨域问题。

```

浏览器机制

- 用 原生 AJAX 进行跨域请求，会发现报错了，仔细分析错误
- 得到结论：AJAX 本身是可以跨域的，但是浏览器的同源策略机制，限制了 XMLHttpRequest 请求，导致无法跨域，所以 AJAX 在浏览器中跨域是一个伪命题，而 AJAX 不是无法跨域的主谋。
- 以下几种情况会被浏览器视为不安全，而直接进行拦截
- 同源规则：同域名、端口、协议（只针对 XMLHttpRequest）
- www.a.com 和 a.com实际是配置的两个域名

编号	url	说明	是否允许通信
1	`http://www.a.com/a.js` `http://www.a.com/b.js`	同一域名下	允许

编号	url	说明	是否允许通信
2	`http://www.a.com/a/a.js` `http://www.a.com/b/b.js`	同一域名不同文件夹	允许
3	`http://www.a.com:8080/a.js` `http://www.a.com:9090/a.js`	同一域名不同端口号	不允许
4	`http://www.a.com/a.js` `https://www.a.com/b.js`	同一域名不同协议	不允许
5	`http://www.a.com/a.js` `http://192.168.4.158/b.js`	域名与域名对应的ip地址	不允许
6	`http://www.a.com/a.js` `http://github.a.com/b.js`	主域名相同，子域名不同	不允许
7	`http://www.a.com/a.js` `http://a.com/b.js`	同一域名，不同二级域名（同上）	不允许（cookie这种情况下也不允许访问）
8	`http://www.a.com/a.js` `http://www.b.com/b.js`	不同域名	不允许

AJAX 跨域

1. 代理
   - 向服务器发送对应的 url，服务器在后台做了一个代理，前端只需要访问A的服务器也就相当与访问了B的服务器，比如在A（`www.a.com/sever.php`）和B（`www.b.com/sever.php`）各有一个服务器，A的后端（`www.a.com/sever.php`）直接访问B的服务，然后把获取的响应值返回给前端。这种代理属于后台的技术，所以不展开叙述。
2. JSONP
   - 说白了就是利用同源策略的漏洞，利用创建标签的形式进行加载。

- JS 中的实现

```
1
2  //创建一个script元素
3
4  var  Scr = document.reateElement('script');
5
6  //声明类型
7
8  Scr.type='text/javascript';
9
10 //添加src属性，引入跨域访问的url
11
12 Scr.src=url;
13
14 //在页面中添加新创建的script元素
15
16 document.getElementsByTagName('body')[0].appendChild(Scr)
```

- jQuery 中的实现：

```
1 $.ajax({
2     url: 'http://192.168.1.114/yii/demos/test.js', //不同的域
3     type: 'GET', // jsonp模式只有GET是合法的
```

```
    data: {
        'action': 'aaron'
    }, // 预传参的数组
    dataType: 'jsonp', // 数据类型
    jsonp: 'backfunc', // 指定回调函数名，与服务器端接收的一致，并回传回来
})
```

整个流程就是：

客户端发送一个请求，规定一个可执行的函数名（这里就是jQuery做了封装的处理，自动帮你生成回调函数并把数据取出来供success属性方法来调用,不是传递的一个回调句柄），服务端接受了这个backfunc函数名，然后把数据通过实参的形式发送出去

其实就是jquery内部会转化成`http://192.168.1.114/yii/demos/test.js?backfunc=jQuery203003857393935322761 5_1402643146875&action=aaron`
然后动态加载
`<script type="text/javascript" src="http://192.168.1.114/yii/demos/test.js?backfunc=jQuery203003857393935322761 5_1402643146875&action=aaron"></script>`

3. XHR2
    - “XHR2” 全称 “XMLHttpRequest Level2” 是HTML5提供的方法，对跨域访问提供了很好的支持，并且还有一些新的功能。
    - IE10以下的版本都不支持
    - 只需要在服务器端头部加上下面两句代码：
    - header("Access-Control-Allow-Origin:*");
    - header("Access-Control-Allow-Methods:POST,GET");