跨域请求和跨域问题解决

跨域请求

跨域解决方案一——Access-Control-Allow-Origin

跨域解决方案二——JSONP - JSON with Padding

跨域解决方案三——后端代理

域：也称为源，协议+域名（ip、主机）+端口

协议：http,https

域名：ip、主机、域名，标识一台计算机在网络中的名称（位置）

端口：网络端口

比如： http://miaov.com:80

跨域：

发送请求所在的域与接收请求的域不一致，就会产生跨域请求

http://miaov.com/index.html -> http://baidu.com/index.html

同源策略 Same origin policy

安全方面的策略，该策略是浏览器的。

当浏览器的某个页面加载了某个资源的时候，浏览器会根据同源策略方案去检测给资源的来源，如果他的来源与当前页面的源不一致，则浏览器会拒绝接收。

浏览器中不是所有的资源请求都会使用该策略，比如通过**link**标签，**script**标签，**img**标签加载的资源不会使用同源策略，有的是会受到同源策略影响的，比如ajax对象

**跨域解决方案一——Access-Control-Allow-Origin**

**标准**浏览器会对同源策略进行了一个修正，利用标准浏览器下定义特殊头信息：Access-Control-Allow-Origin。

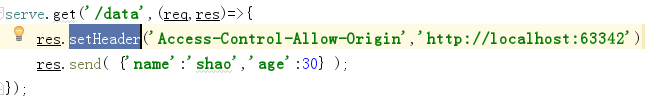
在服务器里面，传输的头信息设置Access-Control-Allow-Origin的值

如果接收到响应的头信息中包含'Access-Control-Allow-Origin'，该字段的值中如果有当前请求域的信息，则会通过验证。

Access-Control-Allow-Origin是用来设置允许通过的源，如果请求域在当前这个字段中设置的域中，那么该请求将会被允许

缺点：需要修改服务端。要在标准浏览器下面

服务端设置：



客户端设置：

clipboard.png



**跨域解决方案二——JSONP - JSON with Padding**

JSONP不是什么新技术，利用现在的一些特性来完成跨域数据的访问。

考虑：用非ajax的方式去实现无刷新的数据请求，需要有以下的几个问题要解决：

需要通过js来发送请求，这样才能够无刷新、js可操控数据

不受同源策略的影响

结论：script元素对象：这就是我们的选择

缺点：不能发送POST请求，不能把数据通过body进行传输

具体的流程：

新建script标签

改变script标签的src，将要跨域的地址赋值给src

往页面中添加script。只要一添加到页面，就会自动发送http的请求

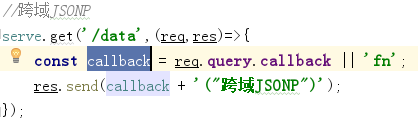
从服务端请求回来的内容会当做js解析执行

所以要和后端协商函数接口：

后端：后端输出一个函数调用，在函数调用中传入后端返给前端的数据

前端：前端定义函数的过程，当前端请求后端拿到数据的时候，会执行该数据（执行了那个函数调用）

服务端设置：



客户端设置：



应用：淘宝搜索框



跨域解决方案三——后端代理

前端请求同域下的后端，同域下后端再去请求跨域的后端，我们同域下的后端就是代理