cors(cross origin resource share)常见实现方案：

1. XMLHttpRequest:

预备知识：XMLHttpRequest跨域请求分为：

a).简单跨域请求：当同时满足下列两个条件时视为简单跨域请求

(1).请求方法是GET、HEAD或者POST，并且当请求方法是POST时，Content-Type必须是application/x-www-form-urlencoded, multipart/form-data或着text/plain中的一个值

(2). 请求中没有自定义HTTP头部。

对于简单跨域请求，浏览器要做的就是在HTTP请求中添加Origin Header，将JavaScript脚本所在域填充进去，向其他域的服务器请求资源。服务器端收到一个简单跨域请求后，根据资源权限配置，在响应头中添加Access-Control-Allow-Origin Header。浏览器收到响应后，查看Access-Control-Allow-Origin Header，如果当前域已经得到授权，则将结果返回给JavaScript。否则浏览器忽略此次响应

b).带预检的跨域请求（preflighted request）:不满足简单跨域请求条件的即为preflighted request.

带预检(Preflighted)的跨域请求需要浏览器在发送真实HTTP请求之前先发送一个OPTIONS的预检请求，检测服务器端是否支持真实请求进行跨域资源访问，真实请求的信息在OPTIONS请求中通过Access-Control-Request-Method Header和Access-Control-Request-Headers Header描述，此外与简单跨域请求一样，浏览器也会添加Origin Header。服务器端接到预检请求后，根据资源权限配置，在响应头中放入Access-Control-Allow-Origin Header,Access-Control-Allow-Methods Header和Access-Control-Allow-Headers Header，分别表示允许跨域资源请求的域、请求方法和请求头。此外，服务器端还可以加入Access-Control-Max-Age Header，允许浏览器在指定时间内，无需再发送预检请求进行协商，直接用本次协商结果即可。浏览器根据OPTIONS请求返回的结果来决定是否继续发送真实的请求进行跨域资源访问。这个过程对真实请求的调用者来说是透明的。

XMLHttpRequest支持通过withCredentials属性实现在跨域请求携带身份信息(Credential，例如Cookie或者HTTP认证信息)。浏览器将携带Cookie Header的请求发送到服务器端后，如果服务器没有响应Access-Control-Allow-Credentials Header，那么浏览器会忽略掉这次响应。

参考：

<http://fengchangjian.com/?tag=preflighted>

<http://www.freebuf.com/articles/web/18493.html>

测试环境：

域1：本地搭建wamp(window+apache+mysql+php),域名：127.0.0.1,文件名：cors.html

域2：百度云开发机,服务器是lighttpd,域名不方便透露,以[www.a.com](http://www.a.com)代替,文件名：test.php

测试详情：

你可能写的初始代码如下：

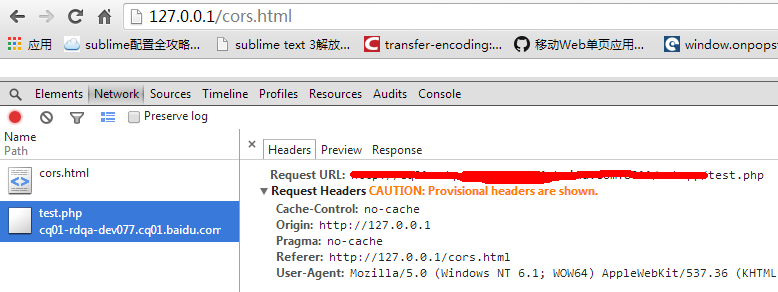
cors.html：



test.php:



结果：



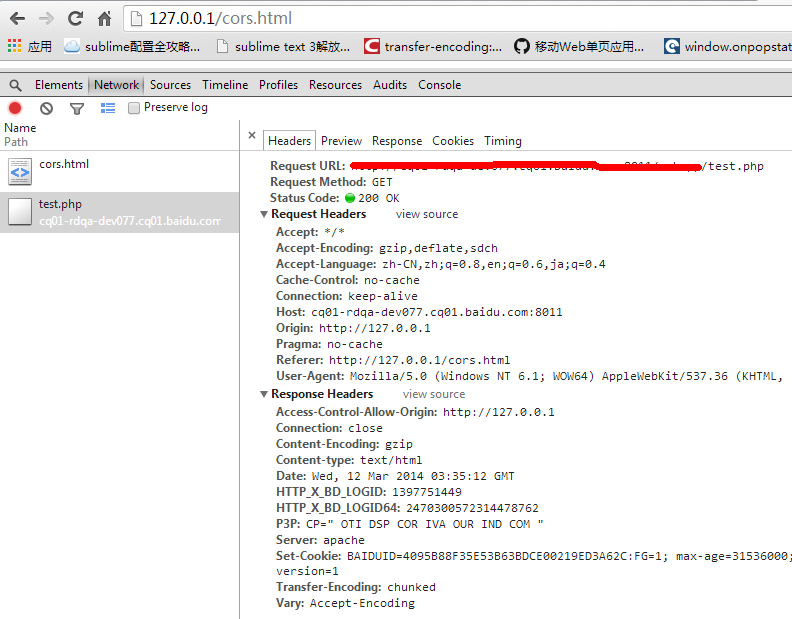
说明：

1. 上述请求为简单跨域请求，服务器端没有配置Access-Control-Allow-Origin导致出错

更改test.php文件为：

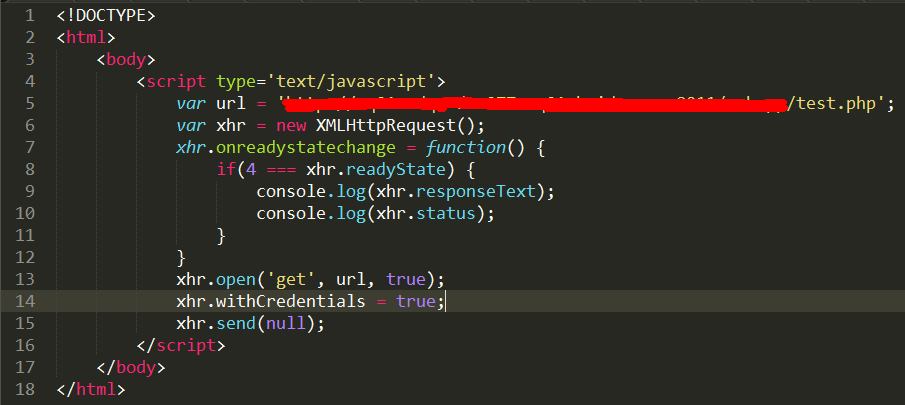


请求变为：

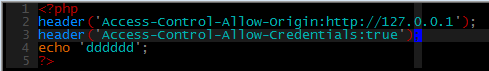


结果：成功，但是没有带cookie，若要带上cookie，需要修改文件如下：

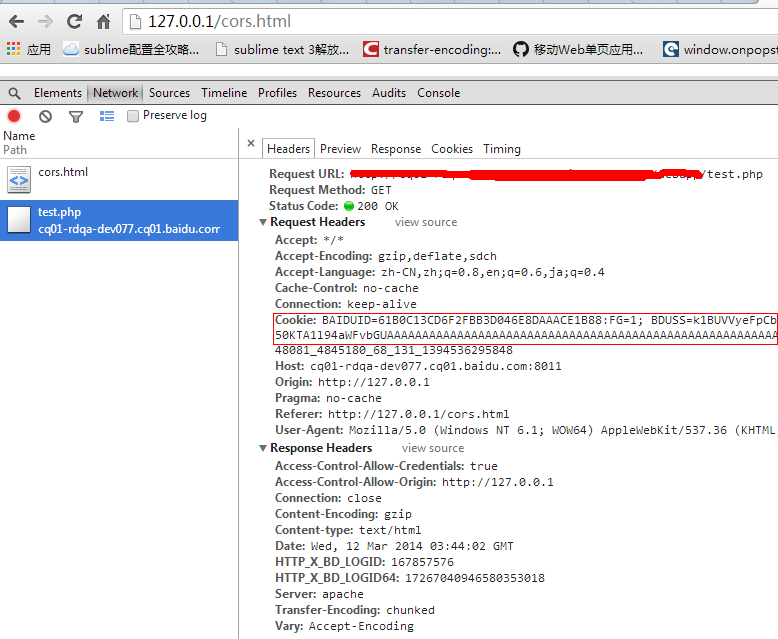
cors.html改为：



Test.php必须相应修改：



结果：



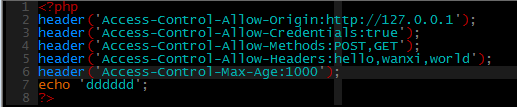
特别说明：如果配置了xhr.credentials为true，此时基于安全考虑,test.php中的header Access-Control-Allow-Origin不能配置为\*

非简单跨域请求(Preflighted Request):

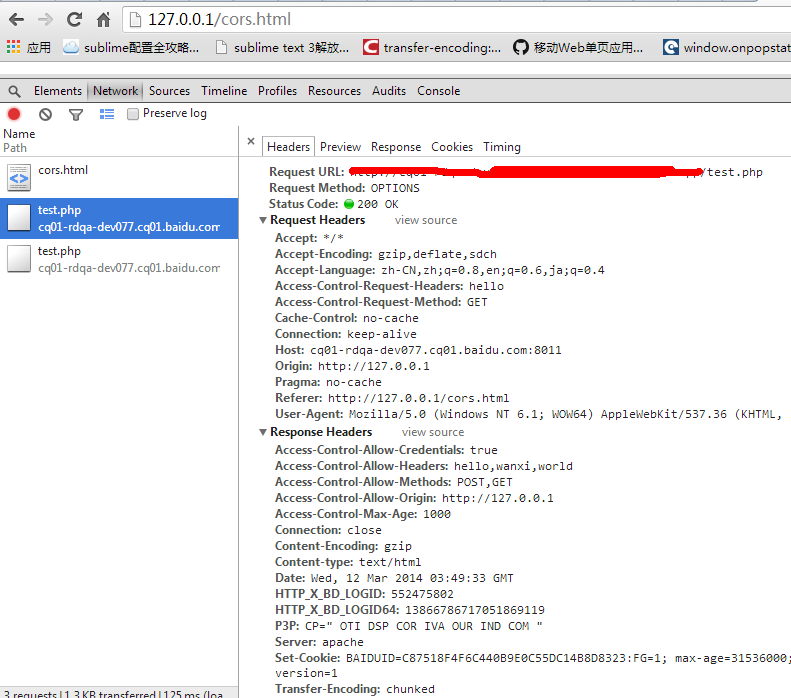
cors.html改为（添加了自定义头部，不满足简单跨域请求的第二个条件了）：



test.php改为：



结果：



结论：

1.test.php发了两次请求，第一次是preflighted request,request method为OPTIONS,浏览器发现自定义了头部,所以需要先发个请求去服务器端验证下这个自定义头部服务器是否允许，对比返回的结果中Access-Control-Allow-Headers发现可以，再发真正test.php请求。

2.jsonp,图像ping

原理：浏览器请求<img />和<script>标签中外链内容时不做扩展名判断，即<script>中src是.php文件也照样请求（亲自测试过），<img>也一样，对返回的数据直接当js或图片解析。

3.iframe

限制：主域必须相同，即必须形如map.baidu.com和baike.baidu.com,主域都是baidu.com