

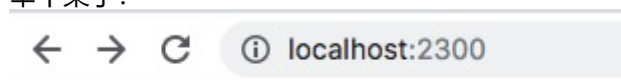
第一部分：什么是跨域？

lin:

首先，说一下同源，如果两个 url 的协议 域名 端口三者完全相同，就称之为同源

同源之间获取资源是不受限制的，如果不满足同源（也就是上面三个至少有一个不同），那么获取资源就会受限，一般我们称之为 跨域

举个栗子：



hello

比如现在有一个 url 是 <http://localhost:2300>

另一个地址是 <file:///Users/lin/Documents/workspace/ssd/fe17/kwyu/index.html>

注意这两个 url 的协议不一样，所以如果 index.html 向 localhost:2300 发送请求的话，就会被拦截下来

当然不是所有请求都会被拦截，但是大部分都会被拦截

【Q&A】

Q：拦截是显示undefined还是直接报错？

A：会报一个类似这样的错误，如下图



第二部分：如何解决跨域？

lin:

那要如何解决这个问题（跨域）呢？有很多个方案

我说一个现在常用的和一个古代常用的

现在常用的方案是 CORS

cors 说的是一种机制，其实相当于一个约定，就是用 http 头部字段来开一扇后门

服务器本来收不到浏览器发的请求（因为被拦截了），现在服务器说我可以给你开一扇门，但是你要符合一定的条件

请求如下图。浏览器会在原来的请求头部上新增一个字段 Origin，用来表示是从哪里发的，这个值一般是一个网址，我们这里是本地文件，所以值是 null

▼ Request Headers

⚠ Provisional headers are shown

Origin: null

Sec-Fetch-Mode: cors

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS
i/537.36

响应如下图。然后服务器响应会返回一个字段 Access-Control-Allow-Origin，表示能放行什么网址的请求，* 表示放行所有请求

▼ Response Headers

Access-Control-Allow-Headers: Content-Type

Access-Control-Allow-Origin: *

Connection: keep-alive

Content-Length: 0

Date: Thu, 05 Sep 2019 09:56:07 GMT

X-Powered-By: Express

浏览器本来会拦截跨域请求，但是加了 cors 这样的相关字段之后，浏览器就不会拦截

【Q&A】

Q: cors是直接往网址上加吗？

Q: 如何设置 cors？

Q: 引入即使用吗？我看老师给的文件内是引入了，还需要一些其他的设置吗

A:

浏览器会在原来的请求头部上新增一个字段 Origin，用来表示是从哪里发的，
一个网址，我们这里是本地文件，所以值是 null

PM5:56

然后服务器响应会返回一个字段 Access-Control-Allow-Origin，表示能放行什
求，* 表示放行所有请求

Q: 网址发送请求后如果被屏蔽才会响应cors机制吗？

A: 是浏览器发送的时候就会带上，如果服务器没有告诉浏览器可以放行，那么浏览器就会报错
目前不需要掌握如何实现的

先掌握 cors 是用来解决什么问题的，怎么解决等下会说

lin:

浏览器发请求的时候在头部字段加一个 Origin 用来表示自己从哪里来

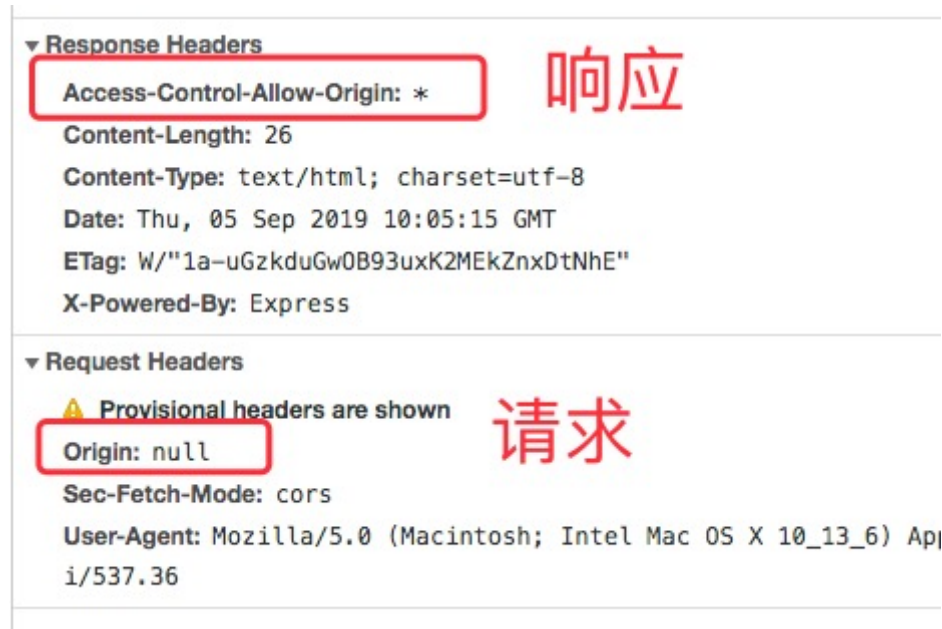
服务器收到请求返回响应的时候会在头部字段加一个 Access-Control-Allow-Origin 用来表示能接收从哪里来的请求

如果两个匹配上了，就继续。如果没有匹配上，浏览器就会报错

https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Access_control_CORS#%E7%AE%80%E5%8D%95%E8%A

浏览器会根据情况发送简单请求或者非简单请求

我发的链接就是说的简单请求，这种情况比较简单，就是浏览器发一个请求，然后服务器返回一个响应



那么非简单请求呢，比如前端与后端一般用 json 格式的数据进行通信，前端发送的请求头部字段会带有

Content-Type，并且值为 application/json，这种就是非简单请求

https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Access_control_CORS#%E9%A2%84%E6%A3%80%E8%AF%B7%E6%B1%82

当然 mdn 称之为 预检请求

【Q&A】

Q：这两个匹配是一一对应的关系吗？就是说，是不是具有唯一性？

A：包含关系

浏览器发送的请求 Origin 一定有一个确定的值，这个值如果被 Access-Control-Allow-Origin 包含，
那么就可以放行，不然不放行

lin:

预检请求说的是浏览器先发一个请求试探一下，如果服务器说放行，然后浏览器根据情况继续

▼ General

Request URL: http://localhost:2300/api/todo/all/cors

Request Method: OPTIONS

Status Code: 200 OK

Remote Address: [::1]:2300

Referrer Policy: no-referrer-when-downgrade

▼ Response Headers

Access-Control-Allow-Headers: Content-Type

Access-Control-Allow-Origin: *

Connection: keep-alive

Content-Length: 0

Date: Thu, 05 Sep 2019 11:37:03 GMT

X-Powered-By: Express

▼ Request Headers

⚠ Provisional headers are shown

Access-Control-Request-Headers: content-type

Access-Control-Request-Method: GET

Origin: null

Sec-Fetch-Mode: no-cors

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit:
i/537.36

预检请求会先发送一个 OPTIONS 方法（注意，平时一般用 GET 方法或者 post 方法），并且会在请求头部加上 Origin 和 Access-Control-Request-Headers 还有 Access-Control-Request-Method

服务器如果能够正确处理这个OPTIONS请求，浏览器才会把后面的请求发出来，如果没有正确处理，浏览器依然报错

▼ General

Request URL: http://localhost:2300/api/todo/all/cors

Request Method: OPTIONS

Status Code: 🟢 200 OK

Remote Address: [::1]:2300

Referrer Policy: no-referrer-when-downgrade

▼ Response Headers

Access-Control-Allow-Headers: Content-Type

Access-Control-Allow-Origin: *

Connection: keep-alive

Content-Length: 0

Date: Thu, 05 Sep 2019 11:41:06 GMT

X-Powered-By: Express

▼ Request Headers

⚠️ Provisional headers are shown

Access-Control-Request-Headers: content-type

Access-Control-Request-Method: GET

Origin: null

Sec-Fetch-Mode: no-cors

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36

那么如何正确处理呢


请求头部用了 content-type, 那么服务器就需要允许这个头部, 也就是 Access-Control-Allow-Headers: Content-Type

method 也是一样的道理, 不过因为例子中的 method 是 GET, 所以可以省略, 其他要写上

▼ General

Request URL: http://localhost:2300/api/todo/all/cors

Request Method: GET

Status Code:  200 OK

Remote Address: [::1]:2300

Referrer Policy: no-referrer-when-downgrade

▼ Response Headers

Access-Control-Allow-Origin: *

Content-Length: 26

Content-Type: text/html; charset=utf-8

Date: Thu, 05 Sep 2019 11:41:06 GMT

ETag: W/"1a-uGzkduGw0B93uxK2MEkZnxDtNhE"

X-Powered-By: Express

▼ Request Headers

 **Provisional headers are shown**

Content-Type: application/json

Origin: null

Sec-Fetch-Mode: cors

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36

正确处理之后，浏览器就会发出真正的请求

否则就是会当场报错处理流程就是这样的

【总结一下】

cors 说的是浏览器在发送请求的时候会带上一些特殊头部字段，然后服务器检查一下，如果能放行就会放行，如果不能放行就会报错

放行的话又会分为两种情况，简单请求很简单，不用说

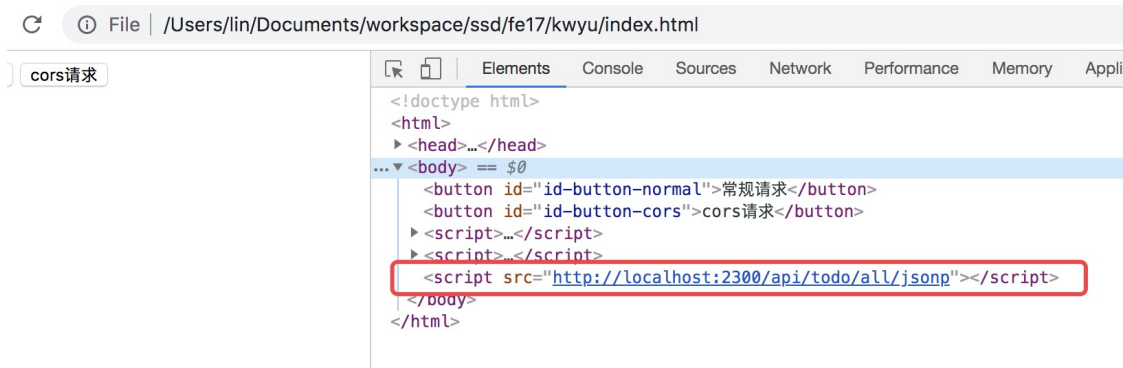
预检请求会先发送 OPTIONS 方法问服务器，服务器除了检查 Origin 还会检查 headers 和 method，都符合要求浏览器才会发出真正的请求，否则会报错

第三部分：JSONP方案

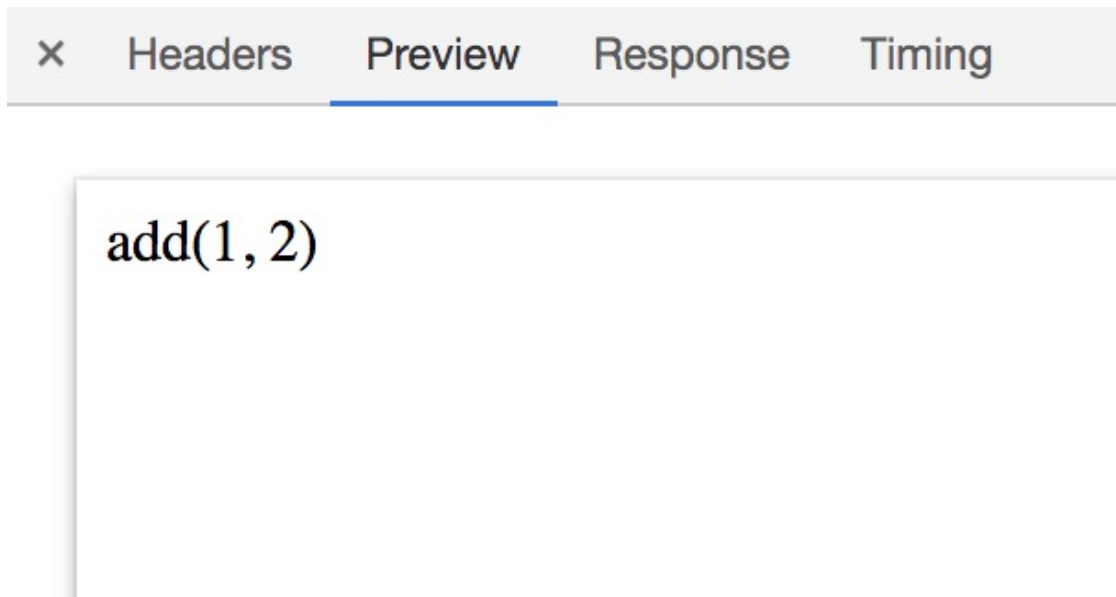
lin:

之前说跨域问题的时候，如果服务器不能放行的话，浏览器会报错

但是对于 script 标签来说，有一个折中方案，也就是 src 的值可以不受服务器限制



比如 index.html 里面的 script 标签，然后可以访问 /api/todo/all/jsonp 这种 url，浏览器会自动以 GET 方法发送这样的请求



这个请求的响应是一个字符串，而且这个字符串很奇怪，是 add(1, 2)

浏览器收到响应之后会把这个字符串当成代码，也就是会执行这个代码

换句话说，也就是会调用 add 函数，并且传入参数 1 和 参数 2

那么，如果我们提前定义了 add 这个函数，并且接收两个参数，实现逻辑，add(1, 2) 就能能够调用成功的

```
</script>
<script>
  const add = (a, b) => {
    log('a and b', a, b, a + b)
  }
</script>
<script src="http://localhost:2300/api/todo/all/jsonp"></script>
```

比如我提前定义好 add 函数，那么调用 add(1, 2) 就会成功

```
20 app.get('/api/todo/all/jsonp', (request, response) => {
21   ...let s = 'add(1, 2)'
22   ...response.send(s)
23 })
```

服务器实现是这样的，很简单
jsonp 就是这样的，我讲完了，大家有问题直接提

【Q&A】

Q: jsonp的作用类似于“把函数提前定义好了，后面直接调用”，这样理解对吗？

A: jsonp 的作用是返回和前端约定好的内容，这样前端就可以直接用

Q: 那我后端返回 function(arg1, arg2)这种字符串时，怎么知道前端有没有对应的方法？

A: 比如目前后端返回的是 add(1, 2)，那如果前端定义的函数不是这个名字，不就用不了了吗，扩展性太差了，所以要改一下

怎么改呢

```
▼ <script>
    const add = (a, b) => {
      log('a and b', a, b, a + b)
    }

    const add1 = (a, b) => {
      log('a + b', a + b)
    }

  </script>
  <script src="http://localhost:2300/api/todo/all/jsonp?
  callback=add1&param1=10&param2=20"></script>
```

发送 get 请求的时候把参数放在 url 的 query string 里面，比如 callback 就是函数名称，param1 和 param2 就分别是两个参数

```
20 app.get('/api/todo/all/jsonp', (request, response) => {
21   ...let query = request.query
22   ...let callback = query.callback
23   ...let a = query.param1
24   ...let b = query.param2
25   ...let s = `${callback}(${a}, ${b})`
26   ...log('s is', s)
27   ...response.send(s)
28 })
```

后端解析出 query string 里面的这些内容，然后把这些内容拼接成函数调用的形式

应用实例，访问地址为 `http://:::2300`
`s is add1(10, 20)`

现在返回的是 add1 这个函数，并且这个函数是前端决定的

request.query 获取的就是 query string 对应的字典


```
应用实例，访问地址为 http://:::2300
query { callback: 'add1', param1: '10', param2: '20' }
s is add1(10, 20)
```

这样的话，前端调用什么函数，传入什么参数，都是由前端自己实现的，后端只是一个中转站，什么都不用做，只是拼接需要格式的数据

实际上就有点类似 callback，后端只负责调用 callback，但是 callback 具体是什么函数，由前端决定

【Q&A】

Q：后端发送给前端数据后，前端也会按照同步和异步的顺序来执行是吗？

比如 add1是异步 add2是同步，后端按照add1， add2这个顺序接收和返回

前端也还是按照同步先异步后执行的顺序来执行函数

A：和同步还是异步没有关系

Q：如果有两个script src里分别存着两个函数 分别是异步和同步函数 传回前端的时候是不分同步和异步是吗

A：前端发请求，后端发响应

至于你说的同步还是异步和这个一点关系都没有

这是两个完全不同的内容

同步异步说的是调用函数的时候获取数据的方式

发请求收响应说的是 http 协议

完全不是一个层面的内容

Q：我的意思是 前端收到响应 调用函数还是会考虑同步异步问题吗 没有表述清楚

A：可以考虑，也可以不考虑，完全看前端自己的情况

和 jsonp 没有任何关系

【注】

我最后说一句，跨域问题一定是要后端解决的，要么用后端的能力（比如 node nginx 等）

如果一个人说前端能解决跨域，那么这个人

1，不懂跨域

2，不懂前端

【Q&A】

Q：因为是在script标签中发生的事情，所以在页面加载的时候等于就调用请求，然后后端返回对应的方法，然后前端再执行。等于整个动作是在 页面加载的时候完成，这么理解可以吗？

A：差不多吧，但实际上可以用 js 控制 script 标签的生成的

所以也可以不在页面加载的时候执行

Q：script标签插入页面时就会被执行吗？还是得刷新才执行。

A：插入就会执行的

script 标签插入到 dom 树的时候，就会自动用 get 方法往 src 对应的 url 发送请求

Q：嗯，那插入script标签时，就是等于发送了http请求

A：差不多

Q：假如说，现在我点击了某个按钮，触发了函数f1，然后f1向页面中插入了script标签，标签中其实是个跨域的请求，返回的数据是jsonp格式的，比如f2，然后前端执行f2。那么我现在有个疑问，为啥不在点击按钮时，直接前端执行f2呢，为什么要通过后端进行拼接，难道后端对于 f2的参数做了相应的处理吗，不只是简单的拼接。

A：对啊.....

要调用 API 才能拿到参数
比如 add1 的第三个参数就是数据库里面拿到的

【完】