正课:

1. ajax

2. 跨域:\*\*\*\*\*

1. ajax

$.ajax({

url:" ... ",

type: "get"|"post",

data:"变量=值&..."|{变量:值,...},

dataType:"json",

success:function(res){ //在响应成功结束后自动

//res就是响应的结果

},

error(){ 当请求出错时，自动触发 },

complete(){ 无论成功还是出错，只要请求结束就执行 }

})

3.x

.then(function(res){

...

})

2. 跨域:

什么是: 一个域名下的网站，请求另一个域名下的网站的资源

问题: ajax禁止发送跨域请求！

以下情况不允许发送ajax请求:

1. http://www.tedu.com ajax-> http://www.tmooc.cn

2. http://oa.tedu.com ajax-> http://hr.tedu.com

3. http://localhost:3000 ajax-> http://localhost:8000

4. https://baidu.com ajax-> http://baidu.com

443 80

5. http://localhost ajax-> http://127.0.0.1

如何跨域访问: 2种:

1. JSONP，需要客户端与服务端配合

什么是：填充式JSON——JSON with Padding

何时: 今后跨域，首选jsonp

如何:

基本思想：用其它可以跨域请求的元素，代替ajax

程序中都是用<script>元素代替ajax发送请求

方案一:

1. 服务端: 将要返回的数据，填充进一条字符串格式的js语句中，组成一条正确的可执行的js语句，再返回

2. 客户端: 添加<script src="服务器端地址">

结果: script可跨域请求到服务器返回的js语句，并在客户端立刻执行。

问题: 服务端返回的js语句是写死的，众口难调。

方案二:

1. 服务端: 返回一条自定义函数的调用语句

要求客户端必须执行指定名称的函数

2. 客户端: 提前定义一个与服务端同名的函数

函数有一个参数可接受服务器端的数据

函数内可执行任意操作

问题: 函数名是写死的，极容易发生冲突！

方案三:

1. 服务端：接受一个函数名参数

将客户端发来的函数名参数拼接到js语句的开头！

2. 客户端：<script src="地址?参数名=本地函数名"

问题: script是写死的！只能在页面加载时请求一次

方案四:

1. 服务端不变

2. 客户端: 动态创建script元素

script只要被加到页面上，就立刻自动请求

问题: script不断累积

解决: 在回调函数中自动删除script

其实: $.ajax也支持jsonp

$.ajax({

... ,

dataType:"jsonp",

...

})

强调: dataType:"jsonp"需要服务器端配合才能实现。

其实，dataType:"jsonp"的原理和方案四一致:

1. 也是通过在head中动态添加<script>发送请求

2. 也是通过?callback=函数名，发送函数名到服务端

3. 函数名其实是$.ajax自动为success函数生成的随机函数名。

2. CORS，只要服务器端允许即可

在服务器端添加响应头: node中:

res.writeHead(200,{

"Content-Type":"application/json;charset=utf-8",

"Access-Control-Allow-Origin":"指定来源域名"|"\*"

})