前后端交互02

课堂主题

1）CORS解决跨域

2）预检请求

3）利用koa-server-http-proxy实现服务端代理

4）前后分离开发

5）基于jwt鉴权

课堂目标

- 会使用CORS解决跨域问题
- 了解后端代理
- 会使用koa-server-http-proxy中间件实现代理
- 了解前后端分离工作场景

CORS跨域设置

CORS(Cross-origin resource sharing)，跨域资源共享，是一份浏览器技术的规范，用来避开浏览器的同源策略

简单来说就是解决跨域问题的除了jsonp外的另一种方法；比jsonp更加优雅。

1.('Access-Control-Allow-Origin', '*') //这个表示任意域名都可以访问，默认不能携带cookie了。(必须字段)

```
res.header('Access-Control-Allow-Origin', 'http://www.baidu.com'); //这样写，只有 www.baidu.com 可以访问。
res.header('Access-Control-Allow-Origin', '*'); //这个表示任意域名都可以访问。
```

2.Access-Control-Allow-Headers ： 设置允许requset设置的头部；

```
   res.header('Access-Control-Allow-Headers', 'Content-Type, Content-Length,
Authorization, Accept, X-Requested-With , yourHeaderFeild');
```

3.Access-Control-Expose-Headers 允许客户端获取的头部key；

('Access-Control-Expose-Headers'，'Content-Type, Content-Length, Authorization, Accept, X-Requested-With , yourHeaderFeild')

CORS请求时，`XMLHttpRequest`对象的`getResponseHeader()`方法只能拿到6个基本字段：`Cache-Control`、`Content-Language`、`Content-Type`、`Expires`、`Last-Modified`、`Pragma`。如果想拿到其他字段，就必须在`Access-Control-Expose-Headers`里面指定。

5、预检请求

- 简单的请求直接发送

  ```
  GET
  HEAD
  POST
  或者
  content-type
  text/plain
  multipart/form-data
  application/x-www-form-urlencoded
  ```

- 预检请求

  ```
  PUT
  DELETE
  CONNECT
  OPTIONS
  TRACE
  PATCH
  ```

- Access-Control-Max-Age用来指定本次预检请求的有效期，单位为秒，在此期间不用发出另一条预检请求。(预检请求)
  - 发送预检请求

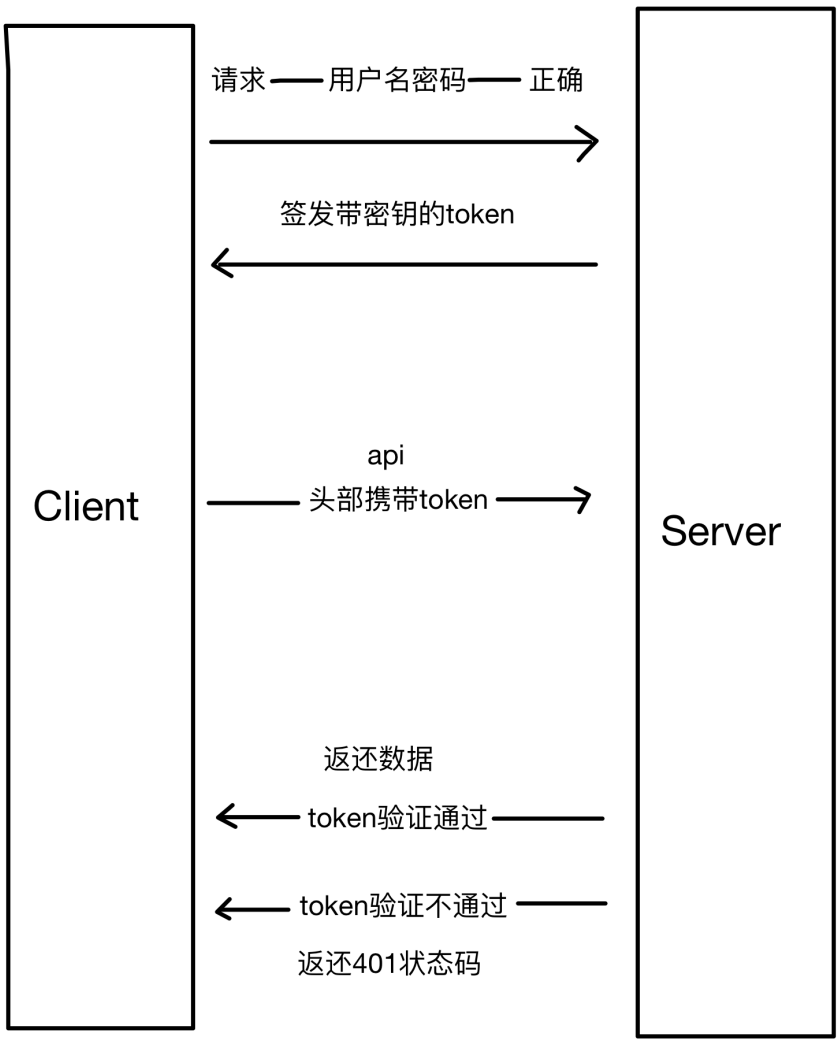
后端代理

- 跨域是浏览器规范，通过同服务器请求数据，不通过浏览器请求，也能解决浏览器限制；
- 转发请求
- 利用http模块实现简单的服务器转发
- 利用 koa-server-http-proxy中间件实现代理

```
app.use(koaServerHttpProxy('/api', {
    target: 'http://localhost:4000',
    pathRewrite: { '^/api': '' }
}))
```

jwt 鉴权

一、jwt：json web token是为了在网络应用环境间传递声明而执行的一种基于JSON的开放标准



二、生成token

- jsonwebtoken 模块

```
    const token = jwt.sign({
          name: reslut[0].username,
          _id: reslut[0].id
      }, 'my_token', { expiresIn: '2h' });
```

三、缓存token：cookie 或者是 locaStroage

四、token的组成

- 头部的基本信息；

```
{
  "typ": "JWT",
  "alg": "HS256"
}
```

- paload :存放自定义信息； 预定义信息有如下几个：

```
iss: 该JWT的签发者
sub: 该JWT所面向的用户
aud: 接收该JWT的一方
exp(expires): 什么时候过期，这里是一个Unix时间戳
iat(issued at): 在什么时候签发的
```

- signature 签名 哈希需要有secret；

Access-Control-Allow-Credentials:布尔值 true允许携带凭证；(可选字段)

```
//客户端设置允许携带用户凭证
xhr.withCredentials = true;

//服务端设置允许携带凭证
ctx.set("Access-Control-Allow-Credentials",true);
```

五、前端的认证

```
"Authorization","Bearer " + token
```

总结

1）CORS解决跨域

2）预检请求

3）后端代理原理

4）利用koa-server-http-proxy中间件实现代理

5）前后分离开发

下期预告

- 前后端交互03