axios知识点总结

axios是什么

1. axios是一个基于promise，可以用在浏览器和node中，常用来做前后端交互。
2. axios的特点
    - 从浏览器中创建XMLHttpRequests
    - 从node.js创建http请求
    - 支持Promise API
    - 拦截请求和响应（下面具体介绍）
    - 转换请求数据和响应数据（response.data 直接能拿响应的json数据，不用再进行转换）
    - 取消请求
    - 自动转换JSON数据
    - 客户端支持防御XSRF（XSRF全称是 cross-site request forgery（跨站点请求伪造），也称为CSRF）

axios使用

1. 安装

    > npm install axios -S 或者 yarn add axios
    >
    > 使用cdn：

2. git请求的使用

```
axios.get('/user')
.then(function (response) {
 //请求成功 获取服务端返回的数据：response.data
 console.log(response);
})
 .catch(function (error) {
 console.log(error);
});
```

3. post 请求的使用

```
axios.post('/user', {
 firstName: 'Fred',
 lastName: 'Flintstone'
})
.then(function (response) {
 //请求成功 获取服务端返回的数据：response.data
 console.log(response);
})
.catch(function (error) {
 console.log(error);
});
```

4. axios的常用配置项

```
axios({
// url:是用于请求的服务器 URL,还可以'/user?id=1'或者'user/id'
```

```
url: '/user',

// method:是创建请求时使用的方法
method: 'get', // default

// baseURL:将自动加在 url 前面，除非 url 是一个绝对 URL。
baseURL: 'https://some-domain.com/api/',

// headers: 是即将被发送的自定义请求头
headers: {'X-Requested-With': 'XMLHttpRequest'},

// params: 是即将与请求一起发送的 URL 参数
// 必须是一个无格式对象(plain object)或 URLSearchParams 对象
params: {
 ID: 12345
},

// adapter: 允许自定义处理请求，以使测试更轻松
// 返回一个 promise 并应用一个有效的响应 (查阅 [response docs](#response-
api)).
adapter: function (config) {
 /* ... */
},

// auth: 表示应该使用 HTTP 基础验证，并提供凭据
// 这将设置一个 `Authorization` 头，覆写掉现有的任意使用 `headers` 设置的自定
义 `Authorization`头
auth: {
 username: 'janedoe',
 password: 's00pers3cret'
},

// responseType: 表示服务器响应的数据类型，可以是 'arraybuffer', 'blob',
'document', 'json', 'text', 'stream'
// 设置响应的数据类型
responseType: 'json', // defaul

// 'proxy' 定义代理服务器的主机名称和端口
// `auth` 表示 HTTP 基础验证应当用于连接代理，并提供凭据
// 这将会设置一个 `Proxy-Authorization` 头，覆写掉已有的通过使用 `header` 设
置       的自定义 `Proxy-Authorization` 头。
proxy: {
 host: '127.0.0.1',
 port: 9000,
 auth: {
   username: 'mikeymike',
   password: 'rapunz3l'
 }
},

// `cancelToken` 指定用于取消请求的 cancel token
// （查看后面的 Cancellation 这节了解更多）
cancelToken: new CancelToken(function (cancel) {
})
})
```

axios实际应用

1. 前后端交互，向后台请求数据
2. 设置代理，进行跨域请求
3. 设置拦截器，在请求前后做操作，例如：设置请求验证
4. 适配器Adapters中做对应不同的处理

axios的原理实现

1. 定义一个Axios类，实现request方法，设置一个从外面传进来的config
2. axios.post,get等都是调用request方法进行封装调用
3. 添加适配器adapter，对所处的node环境和浏览器环境分别做适配，在axios的request方法中调用
4. 添加拦截器InterceptorManager，对所有请求逐个放到一个数组中，再进行逐个执行，达到拦截的目的，通过不断的返回promise对象，达到链式操作的目的
5. 最后将Axios.prototype.request方法赋给axios导出

暗号：axios