# Axios

这是一个基于Promise的Http客户端，并且可以用于浏览器和node.js。

<script src="https://unpkg.com/axios/dist/axios.min.js"></script>

<div id="app">

{{data}}

</div>

<script>

var vm = new Vue({

el: '#app',

data: {

data: ''

},

mounted() {

var p = axios.get('./data.json');

console.log(p); // Promise

p.then((res) => {

console.log(this);

this.data = res;

}).catch(function (err) {

console.log(err);

});

},

});

</script>

**需求：我们需要根据用户获取他的朋友们。并且通过列表把他们的朋友展示出来，现在服务器提供了两个接口：1.根据ID获取用户，2.根据用户获取用户的朋友。**

<div id="app">

<ul>

<li v-for="(friend, index) in friends" :key="index">{{friend.name}}</li>

</ul>

</div>

<script>

var vm = new Vue({

el: '#app',

data: {

user: {},

friends: []

},

mounted() {

axios.post('/users', {

id: '123'

}).then((res) => {

// 获取用户信息,执行我们的 赋值逻辑

this.user = res.data;

}).then((res) => {

// 通过用户信息,调用下一个接口，并返回它的 Promise ，我们这里调用 /users/friends

return axios.post('/users/friends', this.user);

}).then ((res) => {

// 获取 friends 信息，执行我们的 赋值逻辑....

this.friends = res.data

}).catch(function (err) {

console.log(err);

});

},

});

</script>

3个then方法把执行逻辑进行了分割，使逻辑变得更加清晰和可维护，其关键点在于axios会返回一个Promise对象。这是一个基于Promise的Http客户端，并且乐意用于浏览器和node.js。

**同时执行多个ajax请求**

axios.all([

axios.post('/users', {

id: '123'

}),

axios.post('/users/friends'),

]).then(axios.spread((usersData, friendsData) => {

this.user = usersData.data;

this.friends = friendsData.data

})).catch((err) => {

});