主要区别是 axios、fetch请求后都支持Promise对象API，ajax只能用回调函数。

ajax：

Ajax被认为是（Asynchronous JavaScript and XML）的缩写。现在，允许浏览器与服务器通信而无须刷新当前页面的技术都被叫做Ajax。

依赖的传输对象：XMLHttpRequest

ajax无需多言，如果想要更多了解，参考以下链接：ajax详解

axios：

示例：

```
axios({
    method: 'post',
    url: '/user/12345',
    data: {
        firstName: 'Fred',
        lastName: 'Flintstone'
    }
})
.then(function (response) {
    console.log(response);
})
.catch(function (error) {
    console.log(error);
});
```

Vue2.0之后推荐大家用axios替换JQuery ajax，想必让Axios进入了很多人的目光中。Axios本质上也是对原生XHR的封装，只不过它是Promise的实现版本，符合最新的ES规范，它有以下几条特性：

- 从 node.js 创建 http 请求
- 支持 Promise API
- 客户端支持防止CSRF
- 提供了一些并发请求的接口（重要，方便了很多的操作）

Axios既提供了并发的封装，也没有fetch的各种问题，而且体积也较小，推荐大家使用。

更多参考：

[axios-详解]

[axios-npm官网]

fetch

示例：

```
fetch(url).then(res => {
        console.log(res)
}).catch(err => {
        console.log(err)
})
```

Fetch API 提供了一个 JavaScript接口，用于访问和操纵HTTP管道的部分，例如请求和响应。它还提供了一个全局 fetch()方法，该方法提供了一种简单，合理的方式来跨网络异步获取资源。

Fetch 优点主要有：

- 语法简洁，更加语义化
- 基于标准 Promise 实现，支持 async/await
- 同构方便，使用 isomorphic-fetch

不过原生支持率并不高，引入下面这些 polyfill 后可以完美支持 IE8+：

- 由于 IE8 是 ES3，需要引入 ES5 的 polyfill: [es5-shim, es5-sham]
- 引入 Promise 的 polyfill: [es6-promise]
- 引入 fetch 探测库：[fetch-detector]
- 引入 fetch 的 polyfill: [fetch-ie8]
- 可选：如果你还使用了 jsonp，引入 [fetch-jsonp]
- 可选：开启 Babel 的 runtime 模式，现在就使用 async/await