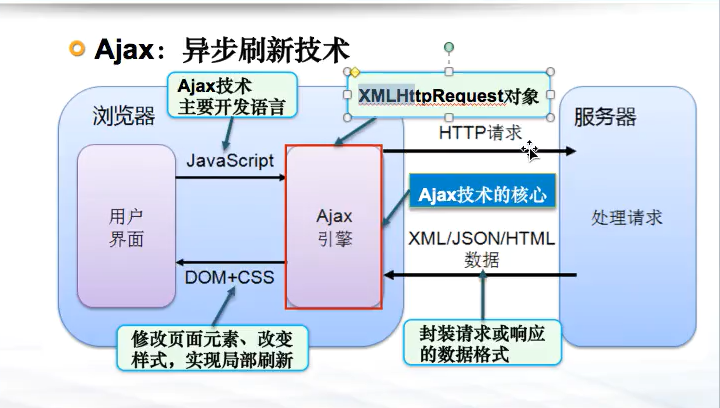
# AJAX

Ajax是JavaScript的一个延伸，他主要用于异步刷新。通常用于局部刷新，我们整个页面刷新的速度慢，用户体验会非常非常差。就以我们的bilibili大学为例吧：



这里在进行视频的播放，播放时右侧边栏是可以进行刷新的，如果我们用传统的JavaScript方法就需要整个页面进行刷新，我们的视频播放就会中断，影响用户体验，而用Ajax之后可以进行局部的刷新，就是右边推荐刷新了视频没有停止播放。

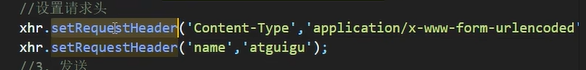
他的对象为XMLHttpRequest。其实是一个JavaScript对象。



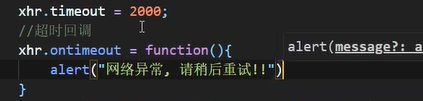
Ajax的代码示例:



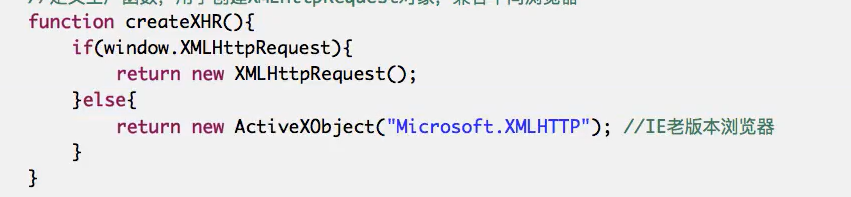
这段代码注释我写的很清楚了，但是对于新手来说有两点可能比较难理解，就是readystate和status这两方面。下面是他们俩的状态码参数。另外注意send是设置请求体的，只有post请求会用到send这玩意。



设置请求头是用上面代码实现的由于我懒得写去截取了一块代码其中xhr就是request只是我的命名不一样就是了。

请求超时问题。

这样就很清楚了。比较重要的一点是status和readystate两个的顺序不能相反，他们两个有先后的，不然&&这块会出现逻辑错误。同时监听事件onreadystatechange要放在send之前，请求发了再绑定没用了。这是两个小细节。



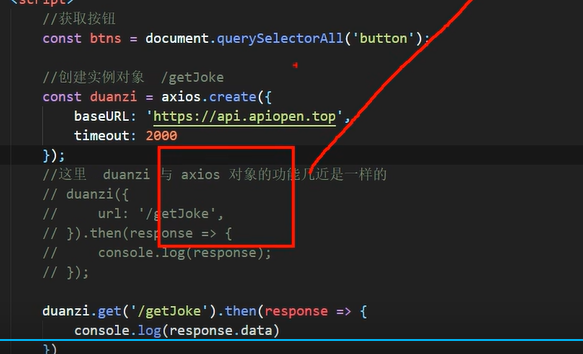
这是兼容性问题，主要就是为了IE这个老逼登写的，老版本IE跟似了一样，主流浏览器没啥问题。

Axios是基于XML和promise封装的方法。所以他返回的是promise对象可以.then.catch同时是异步的所以可以用async和await。

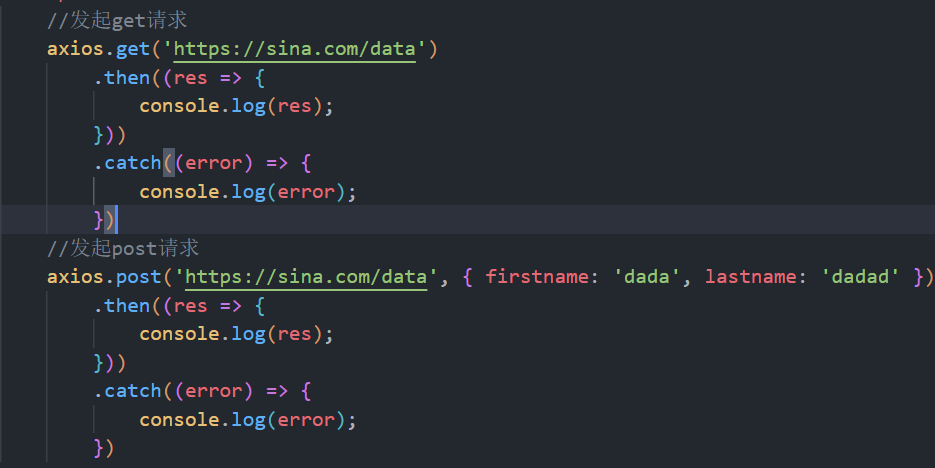
原生XHR语法请求可能有点麻烦，后来经过封装就有了axios这个方法，他是基于Ajax的封装。他的用法也很简单，用CDN，npm，yarn都能引入，引入后就可以直接写了。实际开发中的用法主要有以下几种。



Axios的默认配置。



这个也是一种默认配置的方法，创建一个Ajax实例，通过他来进行一些请求。请求的url是创建实例的Ip和下面请求的路径拼接成的。其中还包含了一个请求超时的问题，也就是那个timeout，请求为2s如果超过2s就算是超时了。

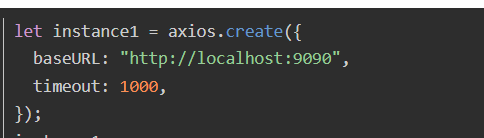


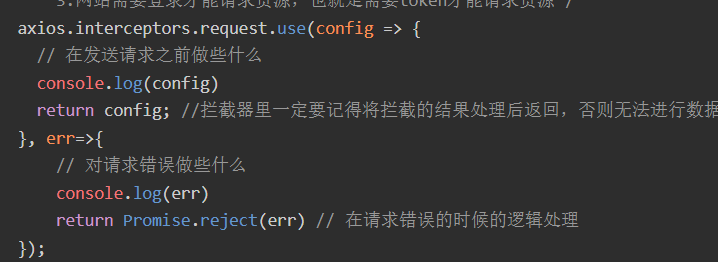
也可以携带参数发送请求，向后端发送数据。带参数发请求get是发数据给后端操作的,向后端发请求是请求数据的。



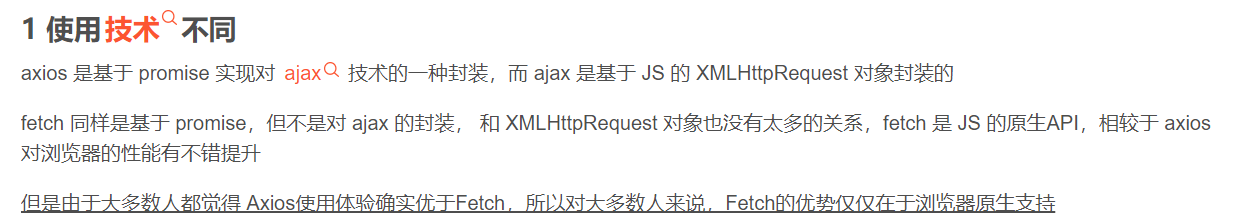
关于拦截器，axios有响应拦截器和请求拦截器，代码示例如下：主要功能是对响应和请求的数据进行一些操作，类似于node的中间件。

请求超时语法很简单，参数里加一个timeout即可





这是请求拦截器，响应拦截器把request换成response即可。

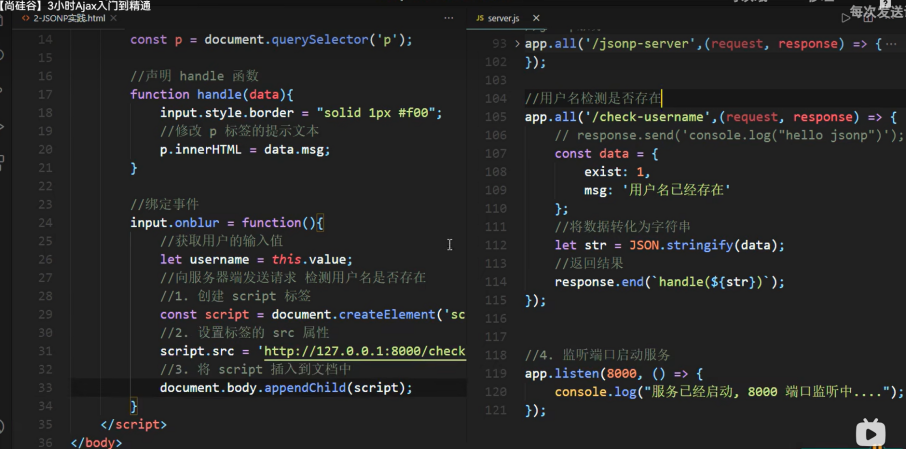
与axios相似的还有一个语法叫fetch，fetch与axios语法相似但没有请求拦截，封装默认值的方法，且其请求超时的语法非常麻烦，所以很少用。只会是面试时问fetch与axios的区别。

下面列举了一个fetch代码。



最后是Ajax的跨域问题。

说道这个问题就要从浏览器的同源策略说起，浏览器为了安全所以请求的数据只能是在同一协议，同一域名，同一端口下才能实现请求。其中任何一个出现不一样的都会导致跨域问题从而导致请求失败。官方提供的解决方法是CORS方法即在后端设置响应头，通过响应头的设置解决跨域问题。但是很多时候要前端解决跨域问题，这个时候用到的就是我们的jsonp这个东西。这其实就是一个JavaScript的漏洞，script标签可以无视同源策略，所以我们把后端响应的数据封装成script代码然后在前端用script引用。比如说我们在前端封装了一个方法，后端返回的是方法的调用，并且在调用里家了参数，这样script引入就可以拿到后端数据了。就可以实现一个请求，且这个请求是无视同源策略的，但是有一点是需要注意的，那就是这个请求他服务器端响应回来的一定是JS代码，否则不生效。无法使用，其原理就是script这个东西他是引入JS的，不是JS代码他不认识。这个方法只能get请求，面试爱问，但是局限性大。

大概就是这样的。