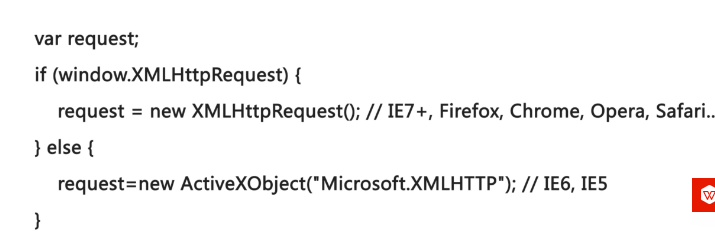
 AJAX 不是新的编程语言，而是一种使用现有标准的新方法。AJAX 即 Asynchronous JavaScript and XML（异步的 JavaScript 和 XML）。它可以在不重新加载整个页面的情况下完成与服务器交换数据并更新部分网页。

**判断服务器是否可以有xhr对象**



**注 :**设置请求头必须在open和send之间，而回调函数通常会在send之前。

1.创建XMLHTTPRequest对象

2.使用open方法设置和服务器的交互信息

3.设置发送的数据，开始和服务器端交互

4.注册事件

5.更新界面

//步骤一:创建异步对象

var ajax = new XMLHttpRequest();

//步骤二:设置请求的url参数,参数一是请求的类型,参数二是请求的url,可以带参数,动态的传递参数starName到服务端

ajax.open('get','getStar.php?starName='+name);

clipboard.png

//步骤三:发送请求

ajax.send();/**/.发送请求(get--null post--数据)**

//步骤四:注册事件 onreadystatechange 状态改变就会调用

ajax.onreadystatechange = function () {

if (ajax.readyState==4 &&ajax.status==200) {

//步骤五 如果能够进到这个判断 说明 数据 完美的回来了,并且请求的页面是存在的

　　　　console.log(ajax.responseText);//输入相应的内容

　　}

}

post请求实例

//创建异步对象

var xhr = new XMLHttpRequest();

//设置请求的类型及url

**注：po**st请求一定要添加请求头才行不然会报错

**注：GET请求的差数直接拼接在url上面POST请求的参数就不是放在url了，而是放在send里面，即请求体**

xhr.setRequestHeader("Content-type","application/x-www-form-urlencoded");

xhr.open('post', '02.post.php' );

//发送请求

xhr.send('name=fox&age=18');

xhr.onreadystatechange = function () {

// 这步为判断服务器是否正确响应

if (xhr.readyState == 4 && xhr.status == 200) {

console.log(xhr.responseText);

}

};

function ajax\_method(url,data,method,success) {

// 异步对象

var ajax = new XMLHttpRequest();

**get一般用于查询 获取操作 发送的信息对所有人可见 很多东西显示在url中 get请求是幂等请求一次和请求很多次结果一样所以也说它是安全的对所发信息的数量有限制**

**post请求一般用于发送表单数据 新建 修改 删除等操作 所有内容和值都在http请求体中对所发信息没有限制**

get 跟post 需要分别写不同的代码

if (method=='get') {

// get请求

if (data) {

// 如果有值

url+='?';

url+=data;

}else{

}

// 设置 方法 以及 url

ajax.open(method,url);

// send即可

ajax.send();

}else{

// post请求

// post请求 url 是不需要改变

ajax.open(method,url);

// 需要设置请求报文

ajax.setRequestHeader("Content-type","application/x-www-form-urlencoded")

// 判断data send发送数据

if (data) {

// 如果有值 从send发

ajax.send(data);

}else{

// 木有值 直接发送即可

ajax.send();

}

}

为了方便使用,我们可以把他封装进方法里面,要用的时候,直接调用就好了

// 注册事件

ajax.onreadystatechange = function () {

// 在事件中 获取数据 并修改界面显示

if (ajax.readyState==4&&ajax.status==200) {

// console.log(ajax.responseText);

// 将 数据 让 外面可以使用

// return ajax.responseText;

// 当 onreadystatechange 调用时 说明 数据回来了

// ajax.responseText;

// 如果说 外面可以传入一个 function 作为参数 success

success(ajax.responseText);

}

}

}

3.服务器响应处理（区分同步跟异步两种情况）

responseText 获得字符串形式的响应数据。

responseXML 获得XML 形式的响应数据。

**XMLHttpRequest 可以同步或异步返回 Web 服务器的响应。**

**①同步处理**

1. xhr.open("GET","info.txt",false); 2. xhr.send(); 3. document.getElementById("myDiv").innerHTML=xhr.responseText; //获取数据直接显示在页面上

②异步处理

相对来说比较复杂，要在请求状态改变事件中处理。

1. xhr.onreadystatechange=function() {

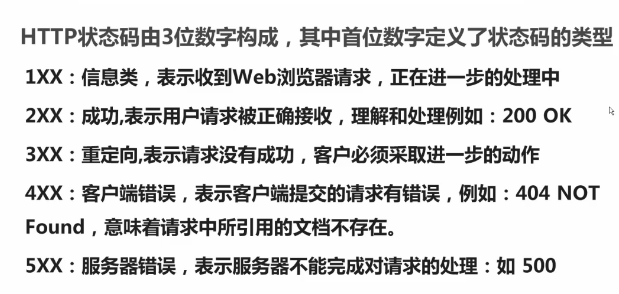
2. if (xhr.readyState==4 &&xhr.status==200) {

3. document.getElementById("myDiv").innerHTML=xhr.responseText;

4. } 5. }

XMLHttpRequest Level 1主要存在以下缺点：受同源策略的限制，不能发送跨域请求；不能发送二进制文件（如图片、视频、音频等），只能发送纯文本数据；发送和获取数据的过程中，无法实时获取进度信息，只能判断是否完成；

if (xhr.readystate == 4) { //状态码在200 到 300表示请求成功，状态码304表示资源没有被修改，可以直接使用缓存中的版本



readyState

0-（未初始化）还没有调用send()方法

1-（载入）已调用send()方法，正在发送请求

2-（载入完成）send()方法执行完成，已经接收到全部响应内容

3-（交互）正在解析响应内容

4-（完成）响应内容解析完成，可以在客户端调用了

status