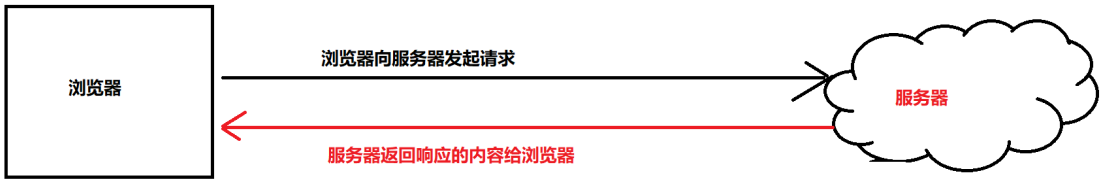
Ajax网络通讯

> 重点，难点，必用点

在Web端网络通讯方法有很多种



1. `ajax` 异步请求
2. `websocket` 双工套接字请求
3. `fetch` （后面ES6里面的）

现在我们主要的重点就放在Aajx上面

什么是Ajax

Ajax即**A**synchronous **J**avascript **A**nd **X**ML（异步JavaScript和XML）在 2005年被Jesse James Garrett提出的新术语，用来描述一种使用现有技术集合的'新'方法，包括: HTML 或 XHTML, CSS, JavaScript, DOM, XML, XSLT, 以及最重要的XMLHttpRequest。 [3] 使用Ajax技术网页应用能够快速地将增量更新呈现在用户界面上，而不需要重载（刷新）整个页面，这使得程序能够更快地回应用户的操作

上面的一段话就是百度百科对于Ajax解释，其实用标哥的话说就是一种浏览器向http/https服务器发起请求的一种方式，只是这种方式是一种异步的方式

假设我们现在一个服务器地址，如下

```
1  http://www.softeem.xin:8888/public/musicData/musicData.json
```

请问如何向上面这个地址发起请求?

Ajax的创建过程

Ajax的核心是`XMLHttpRequest`，它是浏览器的内置对象，用于发起网络请求，向`http/https`服务器发请求

场景：标哥（浏览器）想喝一杯奶茶，B27的楼底下就有一个蜜雪冰城的奶茶店（服务器），然后标哥不想自己去，决定让小岳岳跑一趟。这个时候我们就可以把小岳岳当成是`xhr`

小岳岳就要去B27（地址）的奶茶店去买奶茶（发起求）

第一步：标哥把小岳岳叫过来

```
var xhr = new XMLHttpRequest();
```

第二步：标哥告诉小岳岳地址在哪里，怎么去

```
xhr.open("GET",url,true);
```

第三步：立即让小岳岳去买奶茶

```
xhr.send();
```

第四步：监控小岳岳的变化

```
xhr.addEventListener("readystatechange", function () {
    if (xhr.readyState == 4 && xhr.status == 200) {
        // 第五步:拿到服务器响应的结果
        // 它是一个JSON字符串
        var result = xhr.responseText;
        var obj = JSON.parse(result);
        console.log(obj);
    }
})
```

现在我们把上面的代码整理合并

```
var url = "http://www.softeem.xin:8888/public/musicData/musicData.json";
function getData() {
    //第一步：创建ajax的对象
    var xhr = new XMLHttpRequest();
    //第二步：建立连接
    // GET代表请求方式，后面我们还会讲到POST请求
    // url代表请求的服务器地址
    // true代表异步请求
    xhr.open("GET", url, true);
    //第三步：发送请求
    xhr.send();
    // 第四步：监听请求状态的变化
    xhr.addEventListener("readystatechange", function () {
        if (xhr.readyState == 4 && xhr.status == 200) {
            // 第五步:拿到服务器响应的结果
            // 它是一个JSON字符串
            var result = xhr.responseText;
            var obj = JSON.parse(result);
```

```
            console.log(obj);
        }
    })

}
```

Ajax对象的分析

Ajax请求的核心对象是XMLHttpRequest，它是用来帮我们发请求的，我们要充分手了解这个东西

属性

1. `readyState` 代表ajax请求状态的变化，它一共有0~5个值
    0. 初始化阶段，准备阶段
    1. 载入阶段，开始发起请求
    2. 载入完成，请求发送完成
    3. 解析阶段，服务器处理请求的阶段
    4. 完成，浏览器接收服务器响应的数据
2. `status` 代表ajax请求结果的http的状态码，200代表成功
    - 200代表成功
    - 304代表缓存
    - 403无权限
    - 404代表路径错误
    - 5xx代表服务器错误，服务不能执行一个正确的请求
3. `timeout` 设置请求超时的时间，如果请求超过多长时间得到不到响应，那么我们就判断它失败
4. `responseText` 请求成功以后服务器返回的文本信息
5. `responseType` 设置默认的响应数据类型，这个值默认是 `text`
    - `text` 设置响应的类型为文本格式
    - `json` 设置响应的类型为 `json` 格式，如果设置这个格式，后面在 `response` 的结果就会自动的调用 `JSON.parse()` 进行转换
    - `document` 返回一个网页格式的文件
    - `blob` 代表返回的是一个二进制的数据
    - `arraybuffer` 代表返回的格式是一个缓冲区数组
6. `response` 服务器响应的结果，它不一定是文本信息，当我们的 `responseType` 设置为 `text` 的时候，就可以直接使用 `responesText` 来得到结果，但是如果不是 `text` 的时候就需要通过 `response` 来得到结果

方法

1. `open(method,url,async)` 打开某一个链接

    第一个参数 `method` 代表请求试，目前我们只学习了 `GET`，它常用的有 `GET/POST/PUT/DELETE/OPTIONS/PATCH` 等

    第二个参数代表请求服务器的地址

    第三个参数代表是否异步请求，默认是 `true` 【新版的浏览器只能设置为 `true`】
2. `sennd(params?)` 开始发送请求,send方法里面是有参数的，只是在 `get` 里面没有参数
3. `setRequestHeader` 设置请求头（后面的POST请求会用到）

事件

1. `onreadystatechange` 请求状态的变化
2. `ontimeout` 请求超时以后触发
3. `onerror` 请求失败以后会触发
4. `onabort` 请求取消的时候
5. `onprogress` 请求的进度条发生变化的时候

Ajax的封装

```
/**
 * Ajax请求的工具类
 * @author 杨标
 */

var AjaxHelper = {
    /**
     * ajax发起get请求
     * @param {string} url 请求的地址
     * @param {Function} callBack 请求成功以后的回调函数
     */
    get: function (url, callBack) {
        var xhr = this.init(callBack);
        xhr.open("GET", url, true);
        xhr.send();
    },
    post: function (url,callBack) {
        var xhr = this.init(callBack);
        xhr.open("POST", url, true);
        xhr.send();
    },
    /**
     * 初始化
     * @param {Function} callBack
     * @returns {XMLHttpRequest} 创建好的xhr对象
     */
    init: function (callBack) {
        var xhr = new XMLHttpRequest();
        xhr.addEventListener("readystatechange", function () {
            if (xhr.readyState == 4 && xhr.status == 200) {
                var result = xhr.response;
                if (typeof callBack === "function") {
                    callBack(result);
                }
            }
        });
        return xhr;
    }
```

```
39  }
```

同步与异步的概念

- 同步等待
- 异步执行

场景:

现在是下午，标哥讲了一天的课，要让小岳岳帮我去蜜雪冰城买杯奶茶，喝奶茶润一下嗓子再继续讲课

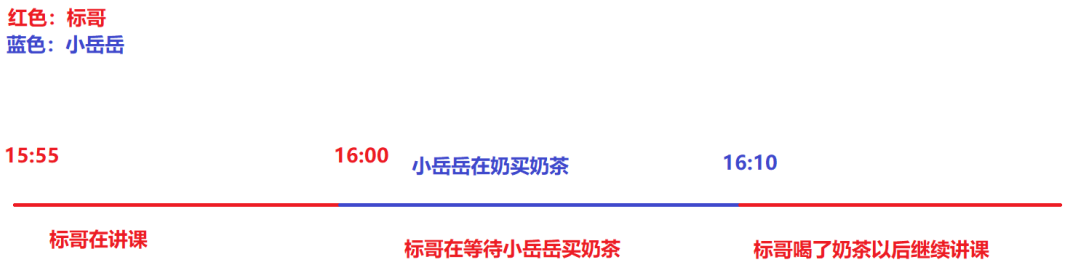
情况一：同步的场景

小岳岳拿着标哥给的5块钱屁颠屁颠的就跑了，标哥看着小岳岳离开的背影，心里久久不平静，它一定要等小岳岳把奶茶买回来，喝了以后再继续讲课

现在我们来分析一下

人物：标哥，小岳岳

事情：标哥要讲课，小岳岳要买奶茶



这种情况就是2个人在同一个时间线上面，它会造成时间阻塞

情况二：异步的情况

小岳岳拿着标哥的钱飞快的跑了出去，帮标哥买奶茶，标哥看着小岳岳这健步如飞的步伐，心中甚是感动，同时也看到了教室里面的学生看着标哥殷切的求知若渴的眼神，标哥不好意思停下来讲解，而是强忍着心中的疲惫继续讲课。不知过了多久，小岳岳回来了，标哥的课也讲得差不多了

这个时候2个时间线是不再不同一个线上面的，它们就是异步，不会造成时间阻塞