Ajax（自我总结）（里面有设置锚点的内容）

Ajax的基本概念 和优缺点及难点：

ajax 我认为是各种请求方式的总称。它的作用主要是，异步处理请求，以及局部刷新。

ajax: Asynchronous Javascript And XML ==> 异步的js和XML
以前更多的是使用XML的数据格式，但是前端技术发展到今天数据格式更多的是json

是一种前后端数据通信（交互）的一种技术，（是一种前端向后端请求数据的一种手段）前端向后端拿动态数据。

优点：

- **1、无刷新更新数据（局部刷新）** Ajax最大的优点就是能在不刷新整个页面的情况下维持与服务器通信。
- **2、异步与服务器通信**：使用异步的方式与服务器通信，不打断用户的操作。
- **3、前端与后端负载均衡**：将一些后端的作用移到前端，减少服务器与带宽的负担。
- **4、界面与应用分离**：Ajax使得界面与应用分离，也就是数据与呈现分离。

缺点：

- **1、Ajax干掉了Back与History功能，即对浏览器机制的破坏**：在动态更新页面的情况下，用户无法回到前一页的页面状态，因为浏览器仅能记忆历史记录中的静态页面。
- **2、安全问题**：Ajax技术给用户带来很好的用户体验的同时，也对IT行业带来了新的安全威胁；Ajax技术就如同对企业数据建立了一个直接通道，使得开发者在不经意间会暴露比以前更多的数据和服务器逻辑。
- **3、对搜索引擎支持较弱**

难点：

- **1、如何拿到数据**（掌握字段含义、服务器要什么客户端就给什么）
- **2、获取到数据之后，如何操作数据。**（主要难点）

注意：只要在服务器上，就尽量不要取中文名字，尽量为英文和数字结合的文件夹。

Ajax的交互模型(两种)

*AJax*有两种交互模型：一种为 *new XML* ；一种为*fetch .then*

第一种交互模型：new XML**

Ajax的经典模型可以叫为电话模式

- 1、先有一个电话 **创建Ajax对象** new XMLHttpRequest
- 2、输入号码（拨号） **填写地址** xhr.open('请求的方式'，'url地址+具体的参数'，是否异步（默认为true））
- 3、发送 **send( )**
- 4、等待 **xhr.onload**
- 5、 成功接收 **在xhr.onload中接受到数据 xhr.responseText**

```
btn.onclick = function(){
        let xhr = new XMLHttpRequest;     //1.创建ajax对象
        xhr.open('get','/get?user='+txt.value,true); //是否异步   2.填
写请求地址
        xhr.send();//3发送
        xhr.onload = function(){ //4.等待服务器响应
            // console.log(xhr.responseText); //5.接收的信息
            let data = JSON.parse(xhr.responseText);
            if(data.code == 0){
                txt.className = 'error';
            }else if(data.code == 1){
                txt.className = 'ok';
            }
        }
   };
```

第二种交互模型：fetch .then

```
fetch('/get?ren=' + txt.val)
.then((e)=>e.json())
.then(data=>{
    // 进行操作数据
})
```

Ajax的请求方式：

请求方式一共有四种：get、post、delete、put
这四种请求方式中，经常用的只有get 和post，这两种请求方式。

get请求方式：**

get请求过程：

- 1、浏览器请求tcp连接（第一次握手）
- 2、服务器答应进行tcp连接（第二次握手）
- 3、浏览器确定，并发送get请求头和数据（第三次握手）
- 4、服务器返回200 OK响应

get是通过url的方式去请求的服务器。

> **补充课堂：url**
> url：是同一资源定位符
> 常见的网址是这样的：https://www.baidu.com:80/item/url/110640?fr=aladdin
> **url的结构：**
>
> - 第一部分：
>   - 模式 / 协议（scheme）
>   - 常用超文本传输协议（Hypertext Transfer Protocol,缩写为HTTP）
>   - https –用安全套接字层传送的超文本传输协议
>   - ftp —文件传输协议
>   - file —当地电脑或网上分享的文件。
> - 第二部分：
>   - 域名 ip地址的一个别名 119.75.217.109 是百度的IP地址。
>   - 顶级域名：www.baidu .com
>   - 二级域名：baike.baidu.com
>   - 三级域名：baike.tieba.baidu.com
>
> .com 商用的国际域名
> .cn 供商用的中文域名
> .net 用于网络供应服务商（系统类的经常使用NET域名）
> .org 用于官方组织
> .edu 用于教育
> .gov 用于政府
>
> - 数字端口号：范围就是0到65535，刘丽娜网页服务的80端口， mongo：27017
> - 带 / 有可能是目录，也有可能是接口，也有可能是二次封装之前的目录呀，接口呀
> - ? 查询信息： ?到#之间的位置，window.location.search 包含问号，不包含#号，即可以读也可以写，写了查询信息是会刷新页面
> - `#` 之后的锚信息：window.location.hash 包含#号，即可以读也可以写，写了锚信息是不会刷新页面， window.onhashchange 当hash值发生变化的时候才会触发 。

get 请求的特性：

- 比post快

- get会将数据缓存起来。而post不会
- url字节长度是每个浏览器都是不一样的（体积是有限的）
- 相对来说是不安全的。
- 中文的转码问题（在Chrome中中文会自动转成URI编码，而IE10以下浏览器都不会自动转）
- 所以要手动编码
- 更擅长于，查找、展示数据、轻量的数据

URI：Uniform Resource Identifier，统一资源标识符。
URL：Unifiom Resource Locator ，统一资源定位符。
URN：Uniform Resource Name，统一资源名称。

> encodeURI('中文') -> 编码
> decodeURI（'%E7%8E%8B'） -> 转成中文。

```
// 第一种模式
btn.onclick = function(){
    let xhr = new XMLHttpRequest;
    xhr.open('get','/get?ren='+encodeURI(txt.value),true);
    xhr.send();
    xhr.onload = function(){
        console.log(xhr.responseText);
    }
}

// 第二个模式
fetch('/get?ren='+txt.value)
.then((e)=>e.json())
.then(data=>{
   console.log(data);
});
```

post请求方式：

请求过程

- 1、浏览器请求tcp连接（第一次握手）
- 2、服务器答应进行tcp连接（第二次握手）
- 3、浏览器确认，并发送post请求头（第三次握手）
- 4、服务器返回100，continue响应。
- 5、浏览器发送数据。
- 6、服务器返回200 OK响应。

与get的请求过程相比，get的总耗是post的2/3左右，口说无凭，网上有过测试。

优点、特征、应用场景：

- 服务器发送
- 安全一点点
- 体积，理论上可以无限大（但是一般后台人员都会限制）
- 在请求发送之前，需要设置请求头
- 一般用在用户信息上、比较大的数据传输。

```
// 第一种请求模式
btn.onclick = function(){
    let xhr = new XMLHttpRequest;
    xhr.open('post','/post',true);
    //设置头信息
    xhr.setRequestHeader('Content-Type','application/x-www-form-url
encoded');
    xhr.send('user='+txt.value);
    xhr.onload = function(){
        console.log(JSON.parse(xhr.responseText));
    }
}

// 第二种请求模式
fetch('/post',{
     method: 'post',
     headers: {
         'Content-Type': 'application/json'
     },
     body:JSON.stringify({'user':txt.value})
 }).then(e=>e.json())
 .then(e=>{
     console.log(e);
 });
```

设置锚点：(里面还包含了选项卡)

```
<style>
.active{
    background: yellow;
}
</style>
</head>
```

```
<body>
    <div id="box"></div>
    <p id="p"></p>
<script>
    let arr = ['新闻','体育','军事','头条'];
    let btns = document.getElementsByTagName('button');
    let hashNum;//为了对应btn的颜色
    let hash = window.location.hash; //一上来先获取hash
    if(!hash){//如果没有
        window.location.hash = 'p=0'; //强行设置一波0，这个时候直接就跑到了hashchange中
    }else{
        //如果有，就获取hash值
        hashNum = window.location.hash.split('=')[1]*1;
        p.innerHTML = arr[hashNum];
    }

    arr.forEach((e,i)=>{
        let btn = document.createElement('button');
        btn.innerHTML = e;
        btn.className = hashNum!=undefined && (e==arr[hashNum]?"active":'');
        btn.onclick = function(){
            window.location.hash = 'p='+i;
        }
        box.appendChild(btn)
    });

    window.onhashchange = function(){
        for(let i=0;i<btns.length;i++){
            btns[i].className = '';
        }
        let i = window.location.hash.split('=')[1]*1;
        btns[i].className = 'active';
        p.innerHTML = arr[i];
    }

</script>
</body>
```