Ajax

从入门到工作2020版

版权声明

- 本内容版权归属杭州饥人谷教育科技有限公司所有。
- 任何媒体、网站、个人未经本公司授权不得转载、链接、转帖，或以其他方式复制、发布、发表。
- 已获得饥人谷授权的媒体、网站、个人在使用时必须注明“资料来源：饥人谷”
- 如咨询课程，微信联系xiedaimala02 或xiedaimala03 ，或者点击链接直接咨询https://dwz.cn/pPsjntxY

如何与后端交互

1. Form表单提交

2. AJAX

3. WebSocket

Form表单提交

- 只支持GET和POST类型

```
<form action="/form.html" method="post">
  <input type="text" name="username" placeholder="username">
  <input type="password" name="password" placeholder="password" >
  <input type="submit">
</form>
```

缺点：提交后页面会刷新，用户体验不佳

AJAX

- 概念

✓ AJAX = Asynchronous JavaScript and XML（异步的 JavaScript 和 XML）

✓ 使用内置的XMLHttpRequest 和 fetch 对象，实现和服务端进行数据交互

✓ 优点：交互数据时页面不需要刷新，体验较好

如何与后端交互

1. Form表单提交
    1. 只支持GET和POST
    2. 有问无答，体验不佳

2. AJAX
    1. 支持
    2. XMLHttpRequest
    3. Fetch

3. WebSocket
    1. 可以由服务端主动发起

XMLHttpRequest

- 发送GET请求

- 创建对象
- 配置参数
- 绑定事件
- 发送请求

```
let xhr = new XMLHttpRequest()
xhr.open('GET', url, true)
xhr.onload = function(){}
xhr.send()
```

XMLHttpRequest

- 举例

```
let url = 'http://rap2api.taobao.org/app/mock/244238/weather?city=北京'
let xhr = new XMLHttpRequest()
xhr.open('GET', url, true)
xhr.onreadystatechange = function(){
  if(xhr.readyState === 4) {  // xhr.DONE === 4 也可
    if((xhr.status >= 200 && xhr.status < 300) || xhr.status === 304){
      console.log(JSON.parse(xhr.responseText))
    } else {
      console.log('服务器异常')
    }
  }
}
xhr.onerror = function(){   // 可选
  console.log('服务器异常')
}
xhr.send()
```

XMLHttpRequest

- 换种写法

```
let url = 'http://rap2api.taobao.org/app/mock/244238/weather?city=北京'
let xhr = new XMLHttpRequest()
xhr.open('GET', url, true)
xhr.onload = function(){
  if((xhr.status >= 200 && xhr.status < 300) || xhr.status === 304){
    console.log(xhr.response)
  } else {
    console.log('数据异常')
  }
}
xhr.responseType = 'json'  //兼容性稍微差一些
xhr.onerror = function(){
  console.log('服务器异常')
}
xhr.send()
```

XMLHttpRequest

- 发送POST请求

```
let url = 'http://rap2api.taobao.org/app/mock/244238/login'
let xhr = new XMLHttpRequest()

xhr.timeout = 3000  //可选，设置请求超时时间
xhr.open('POST', url, true)
xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded')
xhr.onload = function(){
  if((xhr.status >= 200 && xhr.status < 300) || xhr.status === 304){
    console.log(JSON.parse(xhr.responseText))
  } else {
    console.log('响应异常')
  }
}

xhr.ontimeout = function() { console.log('请求超时') }
xhr.onerror = function(){ console.log('服务器异常') }
xhr.send('username=hunger&password=12345678')
```

需要注意的细节

同步 vs 异步

```
let url = 'http://rap2api.taobao.org/app/mock/244238/weather?city=北京'
let xhr = new XMLHttpRequest()
xhr.open('GET', url, true)  // 异步写法，最后一个参数是true
xhr.onload = function(){
    let result = JSON.parse(xhr.responseText)
    console.log(result)
}
xhr.send()
```

```
let url = 'http://rap2api.taobao.org/app/mock/244238/weather?city=北京'
let xhr = new XMLHttpRequest()
xhr.open('GET', url, false)  // 同步写法，最后一个参数是false
xhr.send()
//可能会卡在这里等待很久
let result = JSON.parse(xhr.responseText))
console.log(result)
```

注意：不要使用同步写法

封装

```
const request = (url, params, onsuccess, onerror) => {
    url = url + '?' + Object.entries(params).map(arr => arr[0] + '=' + arr[1]).join('&')
    let xhr = new XMLHttpRequest()
    xhr.open('GET', url, true)   // 异步写法，最后一个参数是true
    xhr.onload = function(){
        if(xhr.status === 200  || xhr.status === 304) {
          onsuccess(JSON.parse(xhr.responseText))
        } else {
            onerror()
        }
    }
    xhr.onerror = onerror
    xhr.send()
}

request('http://rap2api.taobao.org/app/mock/244238/weathe', {city: '杭州'},
    data => {
        console.log('请求成功')
       console.log(data)
    },
    () => {
       console.log('接口异常')
    }
)
```

Post请求编码方式

- 告诉服务器请求参数的编码方式

- application/x-www-form-urlencoded
  - 参数变成 key1=value1&key2=value2 的形式
- multipart/form-data
  - 一般上传文件时使用

Post请求编码方式

- multipart/form-data

```
let formData = new FormData()
formData.append('username',  'abcdefg')
formData.append('password',  '123456')
let url = 'http://rap2api.taobao.org/app/mock/244238/register'
let xhr = new XMLHttpRequest()
xhr.open('POST', url, true)
xhr.onload = function(){
    if(xhr.status === 200  || xhr.status === 304) {
        console.log(JSON.parse(xhr.responseText))
    } else {
        console.log('接口异常')
    }
}
xhr.send(formData )
```

Post登录验证

```
let formData = new FormData()
formData.append('username', 'abcdefg')
formData.append('password', '123456')
let url = 'http://rap2api.taobao.org/app/mock/244238/register'
let xhr = new XMLHttpRequest()
xhr.open('POST', url, true)
xhr.onload = function(){
    if(xhr.status === 200  || xhr.status === 304) {
        console.log(JSON.parse(xhr.responseText))
    } else {
        console.log('接口异常')
    }
}
xhr.send(formData )
```

Post请求编码方式

- http://js.jirengu.com/suhuha/edit?html,js,output

```
const $ = s => document.querySelector(s)
const $submit = $('[type=submit]')
const $form = $('form')
const $msg = $('#msg')
let url = 'http://rap2api.taobao.org/app/mock/244238/register'
$form.onsubmit = function(e) {
  e.preventDefault()
  //这里校验
  let formData = new FormData($form)
  let xhr = new XMLHttpRequest()
  xhr.open('POST', url, true)
  xhr.onload = function(){
    if(xhr.status === 200  || xhr.status === 304) {
      let data = JSON.parse(xhr.responseText)
      $msg.innerText = data.msg
    } else {
      $msg.innerText = '接口异常'
    }
  }
  xhr.send(formData )
}
```

fetch

简单使用

```
let url = 'http://api.jirengu.com/getWeather.php?city=北京'

fetch(url).then(response => response.json() )
    .then(data => {
      document.body.innerText = data.results[0].weather_data[0].weather
   })
```

复杂使用

```
let url = 'http://rap2api.taobao.org/app/mock/244238/login'
let data = {username: 'jirengu', password: '123456'}

fetch(url, {
  method: 'POST',
  body: Object.entries(data).map(arr => arr[0] + '=' + arr[1]).join('&'),
  headers: {
    'Content-Type': 'application/x-www-form-urlencoded'
  }
}).then(res => res.json())
.catch(error => console.error('Error:', error))
.then(response => console.log('Success:', response))
```

更多使用

- 参考 https://developer.mozilla.org/zh-CN/docs/Web/API/Fetch_API/Using_Fetch

双工通信

Ajax轮询

- 每隔固定时间发一次请求

- ✓ 发请求，立即响应，返回空
- ✓ 发请求，立即响应，返回新数据
- ✓ 发请求，立即响应，返回数据为空
- ✓ 发请求，立即响应，返回数据为空
- ✓ ……

长轮询(Comet)

- 客户端
  - ✓ 发请求，等待响应
  - ✓ 当响应时，再次发请求

- 服务器端
  - ✓ 请求到来，如果没新数据，则不发
  - ✓ 当有新数据，需要通知客户端，再响应

试一试

- ajax轮询实现的简易聊天室
  - ✓ https://github.com/jirengu/chat-xhr-poll
- ajax长轮询(comet)实现的简易聊天室
  - ✓ https://github.com/jirengu/chat-comet
- ✓ websocket 简易聊天室
- ✓ https://github.com/jirengu/chat-websocket