AJAX

- AJAX简介
- XMLHttpRequest对象
- XMLHttpRequest的实例方法
- XMLHttpRequest实例的事件
- AJAX代码示例
- 学习网址

AJAX简介

浏览器与服务器之间，采用 HTTP 协议通信。用户在浏览器地址栏键入一个网址，或者通过网页表单向服务器提交内容，这时浏览器就会向服务器发出 HTTP 请求。

AJAX 是 Asynchronous JavaScript and XML 的缩写，指的是通过 JavaScript 的异步通信，从服务器获取 XML 文档，从中提取数据，再更新当前网页的对应部分，而不用刷新整个网页。后来，AJAX 这个词就成为 JavaScript 脚本发起 HTTP 通信的代名词，也就是说，只要用脚本发起通信，就可以叫做 AJAX 通信。

AJAX简介

具体来说，AJAX 包括以下几个步骤。

1. 创建 XMLHttpRequest 实例。
2. 发出 HTTP 请求。
3. 接收服务器传回的数据。
4. 更新网页数据

概括起来，就是一句话，AJAX 通过原生的XMLHttpRequest对象发出 HTTP 请求，得到服务器返回的数据后，再进行处理。现在，服务器返回的都是 JSON 格式的数据，XML 格式已经过时了，但是 AJAX 这个名字已经成了一个通用名词，字面含义已经消失了。

XMLHttpRequest对象

XMLHttpRequest对象是 AJAX 的主要接口，用于浏览器与服务器之间的通信。尽管名字里面有XML和Http，它实际上可以使用多种协议（比如file或ftp），发送任何格式的数据（包括字符串和二进制）。

XMLHttpRequest本身是一个构造函数，可以使用new命令生成实例。它没有任何参数。

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

一旦新建实例，就可以使用open()方法发出 HTTP 请求。

```
xhr.open('GET', 'http://www.example.com/page.php', true);
```

上面代码向指定的服务器网址，发出 GET 请求。

XMLHttpRequest对象

然后，指定回调函数，监听通信状态（readyState属性）的变化。

```
ajax.onreadystatechange = handleStateChange;

function handleStateChange() {
  // ...
}
```

上面代码中，一旦XMLHttpRequest实例的状态发生变化，就会调用监听函数handleStateChange。一旦拿到服务器返回的数据，AJAX 不会刷新整个网页，而是只更新网页里面的相关部分，从而不打断用户正在做的事情。注意，AJAX 只能向同源网址（协议、域名、端口都相同）发出 HTTP 请求，如果发出跨域请求，就会报错。

XMLHttpRequest对象

下面是XMLHttpRequest对象简单用法的完整例子。

```
var xhr = new XMLHttpRequest();

xhr.onreadystatechange = function(){
  // 通信成功时，状态值为4
  if (xhr.readyState === 4){
    if (xhr.status === 200){
      console.log(xhr.responseText);
    } else {
      console.error(xhr.statusText);
    }
  }
};

xhr.onerror = function (e) {
  console.error(xhr.statusText);
};

xhr.open('GET', '/endpoint', true);
xhr.send(null);
```

XMLHttpRequest对象

XMLHttpRequest.readyState返回一个整数，表示实例对象的当前状态。该属性只读。它可能返回以下值。

- 0，表示 XMLHttpRequest 实例已经生成，但是实例的open()方法还没有被调用。
- 1，表示open()方法已经调用，但是实例的send()方法还没有调用，仍然可以使用实例的setRequestHeader()方法，设定 HTTP 请求的头信息。
- 2，表示实例的send()方法已经调用，并且服务器返回的头信息和状态码已经收到。
- 3，表示正在接收服务器传来的数据体（body 部分）。这时，如果实例的responseType属性等于text或者空字符串，responseText属性就会包含已经收到的部分信息。
- 4，表示服务器返回的数据已经完全接收，或者本次接收已经失败。

XMLHttpRequest对象

XMLHttpRequest.responseType属性是一个字符串，表示服务器返回数据的类型。这个属性是可写的，可以在调用open()方法之后、调用send()方法之前，设置这个属性的值，告诉服务器返回指定类型的数据。

XMLHttpRequest.responseType属性可以等于以下值。

- “”（空字符串）：等同于text，表示服务器返回文本数据。
- “arraybuffer”：ArrayBuffer 对象，表示服务器返回二进制数组。
- “blob”：Blob 对象，表示服务器返回二进制对象。
- “document”：Document 对象，表示服务器返回一个文档对象。
- “json”：JSON 对象。
- “text”：字符串。

XMLHttpRequest对象

XMLHttpRequest.status属性返回一个整数，表示服务器回应的 HTTP 状态码。一般来说，如果通信成功的话，这个状态码是200；如果服务器没有返回状态码，那么这个属性默认是200。请求发出之前，该属性为0。

- 200, OK，访问正常
- 301, Moved Permanently，永久移动
- 302, Move temporarily，暂时移动
- 304, Not Modified，未修改
- 307, Temporary Redirect，暂时重定向
- 401, Unauthorized，未授权
- 403, Forbidden，禁止访问
- 404, Not Found，未发现指定网址
- 500, Internal Server Error，服务器发生错误

基本上，只有2xx和304的状态码，表示服务器返回是正常状态。

XMLHttpRequest的实例方法

XMLHttpRequest.open()方法用于指定 HTTP 请求的参数，或者说初始化 XMLHttpRequest 实例对象。它一共可以接受五个参数。

- method：表示 HTTP 动词方法，比如GET、POST、PUT、DELETE、HEAD等
- url：表示请求发送目标 URL。
- async：布尔值，表示请求是否为异步，默认为true。如果设为false，则send()方法只有等到收到服务器返回了结果，才会进行下一步操作。该参数可选。由于同步 AJAX 请求会造成浏览器失去响应，许多浏览器已经禁止在主线程使用，只允许 Worker 里面使用。所以，这个参数轻易不应该设为false。
- user：表示用于认证的用户名，默认为空字符串。该参数可选。
- password：表示用于认证的密码，默认为空字符串。该参数可选。

下面是发送 POST 请求的例子。

```
var xhr = new XMLHttpRequest();
xhr.open('POST', encodeURI('someURL'));
```

XMLHttpRequest的实例方法

XMLHttpRequest.send()方法用于实际发出 HTTP 请求。它的参数是可选的，如果不带参数，就表示 HTTP 请求只包含头信息，也就是只有一个 URL，典型例子就是 GET 请求；如果带有参数，就表示除了头信息，还带有包含具体数据的信息体，典型例子就是 POST 请求。

下面是发送 GET 请求的例子。

```
var xhr = new XMLHttpRequest();
xhr.open('GET',
  'http://www.example.com/?id=' + encodeURIComponent(id),
  true
);
xhr.send(null);

// 等同于
var data = 'id=' + encodeURIComponent(id);
xhr.open('GET', 'http://www.example.com', true);
xhr.send(data);
```

XMLHttpRequest实例的事件

readyStateChange 事件

readyState属性的值发生改变，就会触发 readyStateChange 事件。

我们可以通过onReadyStateChange属性，指定这个事件的监听函数，对不同状态进行不同处理。尤其是当状态变为 4 的时候，表示通信成功，这时回调函数就可以处理服务器传送回来的数据。

```
xmlhttp.onreadystatechange = state_Change;

function state_Change() {
    if (xmlhttp.readyState==4){
    // 4 = "loaded"
        if (xmlhttp.status==200){
        // 200 = "OK"
        }
    }
}
```

XMLHttpRequest实例的事件

progress 事件

上传文件时，XMLHTTPRequest 实例对象本身和实例的upload属性，都有一个progress事件，会不断返回上传的进度。

```
var xhr = new XMLHttpRequest();

function updateProgress (oEvent) {
  if (oEvent.lengthComputable) {
    var percentComplete = oEvent.loaded / oEvent.total;
  } else {
    console.log('无法计算进展');
  }
}

xhr.addEventListener('progress', updateProgress);

xhr.open();
```

XMLHttpRequest实例的事件

load 事件、error 事件、abort 事件

load 事件表示服务器传来的数据接收完毕，error 事件表示请求出错，

abort 事件表示请求被中断（比如用户取消请求）。

```
var xhr = new XMLHttpRequest();

xhr.addEventListener('load', transferComplete);
xhr.addEventListener('error', transferFailed);
xhr.addEventListener('abort', transferCanceled);

xhr.open();

function transferComplete() {
  console.log('数据接收完毕');
}

function transferFailed() {
  console.log('数据接收出错');
}

function transferCanceled() {
  console.log('用户取消接收');
}
```

AJAX代码示例：JavaScript

```javascript
var xmlhttp;
function loadURLDoc(url) {
  xmlhttp=null;
  if (window.XMLHttpRequest) {// code for Firefox, Mozilla, IE7, etc.
    xmlhttp=new XMLHttpRequest();
  }
  else if (window.ActiveXObject) {// code for IE6, IE5
    xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
  if (xmlhttp!=null) {
    xmlhttp.onreadystatechange=state_Change;
    xmlhttp.open("GET",url,true);
    xmlhttp.send(null);
  }
  else {
    alert("Your browser does not support XMLHTTP.");
  }
}

function state_Change() {
  if (xmlhttp.readyState==4 && xmlhttp.status==200) {// 4 = "loaded" and 200 = "OK"
    // doSomething
  } else {
    alert("Problem retrieving data:" + xmlhttp.statusText);
  }
}
```

AJAX代码示例：JQuery

GET请求

```
$.get("demo_test.asp",function(data,status){
  alert("Data: " + data + "\n Status: " + status);
});
```

POST请求

```
$.post("demo_test_post.asp",
{
  name:"Donald Duck",
  city:"Duckburg"
},
function(data,status){
  alert("Data: " + data + "\nStatus: " + status);
});
```

学习网址

- http://www.w3school.com.cn/jquery/jquery_ajax_intro.asp
- https://developer.mozilla.org/zh-CN/docs/Web/Guide/AJAX
- http://javascript.ruanyifeng.com/bom/ajax.html
- http://www.w3school.com.cn/ajax/index.asp