目录

[JavaScript回顾 1](#_Toc480722403)

[1.1 JavaScript 1](#_Toc480722404)

[1.2 Javascript组成 1](#_Toc480722405)

[1.2.1 JavaScript对象 2](#_Toc480722406)

[1.2.2 Browser对象 2](#_Toc480722407)

[1.2.3 HTML DOM对象 2](#_Toc480722408)

[1.3 应用 6](#_Toc480722409)

[1.3.1 showModalDialog运用 6](#_Toc480722410)

[1.3.2 显示服务端时间 6](#_Toc480722411)

[2 Ajax 6](#_Toc480722412)

[2.1 由来与描述 6](#_Toc480722413)

[2.2 原理与技术 7](#_Toc480722414)

[2.2.1 原理 7](#_Toc480722415)

[2.2.2 技术 7](#_Toc480722416)

[2.3 Ajax实现 7](#_Toc480722417)

[2.3.1 XMLHttpRequest 7](#_Toc480722418)

[2.3.2 实现步骤 10](#_Toc480722419)

[2.4 应用 12](#_Toc480722420)

[2.4.1 异步获取服务器时间 12](#_Toc480722421)

[2.4.2 二级联动 12](#_Toc480722422)

[2.5 优缺点 12](#_Toc480722423)

[3 Json 13](#_Toc480722424)

[3.1 描述 13](#_Toc480722425)

[3.2 规则 13](#_Toc480722426)

[3.3 运用 14](#_Toc480722427)

[3.3.1 直接定义Json 14](#_Toc480722428)

[3.3.2 解析转换Json 14](#_Toc480722429)

[3.4 优缺点 14](#_Toc480722430)

[3.5 载体小结 15](#_Toc480722431)

# JavaScript回顾

## JavaScript

JavaScript 由 Brendan Eich 发明。它于 1995 年出现在 Netscape 中（该浏览器已停止更新），并于 1997 年被 ECMA（欧洲计算机制造商协会）采纳并制定标准定义了名为 ECMAScript 的全新脚本语言。国际标准化组织及国际电工委员会（ISO/IEC）也采纳 ECMAScript 作为标准（ISO/IEC-16262）。从此，Web 浏览器就开始努力将 ECMAScript 作为 JavaScript 实现的基础。

JavaScript是一种轻量级的解释执行而非编译的脚本语言，它有如下几个特点：

1. 脚本语言：解释型语言，不需要先预编译而是在程序的运行过程中逐行解释并执行；
2. 简单：采用弱类型的变量类型,对使用的数据类型未做出严格的要求,是类似Java基本语句和控制的脚本语言；
3. 基于对象：可创建对象也可使用已有现成的对象；
4. 动态性：采用事件驱动的脚本语言,它不需要经过Web服务器就可以对用户的输入做出响应；
5. 跨平台性：不依赖于操作系统,仅需要浏览器的支持。

## Javascript组成

JavaScript 由 ECMAScript、BOM和DOM组成。

### JavaScript对象

|  |  |
| --- | --- |
| **对象名称** | **对象说明** |
| Array | 数组 |
| Boolean | 布尔对象 |
| Date | 日期时间 |
| Math | 数学对象 |
| Number | 数值对象 |
| String | 字符串对象 |
| RegExp | 正则表达式对象 |
| Functions | 函数构造器 |
| Events | 事件 |
| 其它 | 如ActiveX、XMLHttpRequest等 |

### Browser对象

Browser Object Model，浏览器对象模型；它包括的对象：

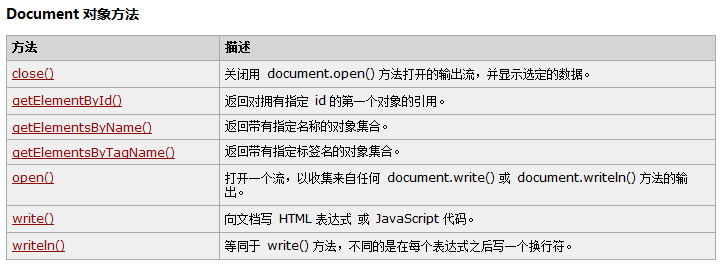
|  |  |
| --- | --- |
| **对象名称** | **对象说明** |
| Window | Window 对象表示浏览器中打开的窗口 |
| Navigator | Navigator 对象包含有关浏览器的信息；如：navigator.appVersion 属性可返回浏览器的平台和版本信息 |
| Screen | Screen 对象包含有关客户端显示屏幕的信息 |
| History | History 对象包含用户（在浏览器窗口中）访问过的 URL；如：history.back()返回上一个页面 |
| Location | Location 对象包含有关当前 URL 的信息；如：location.href返回当前窗口的URL |

### HTML DOM对象

Document Object Model，文档对象模型；DOM 把一个文档以树的形式表示；在 DOM中，每个部分都是节点：文档本身是文档节点 所有 HTML 元素是元素节点 所有 HTML 属性是属性节点 HTML 元素内的文本是文本节点 注释是注释节点。

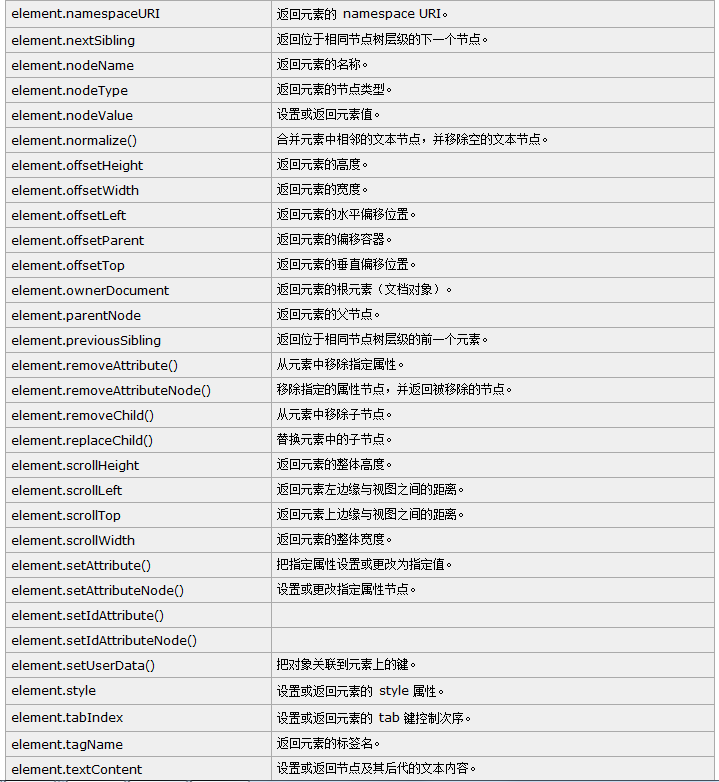
|  |  |
| --- | --- |
| **对象名称** | **对象说明** |
| Document | Document 对象；每个载入浏览器的 HTML 文档都会成为 Document 对象 |
| Element | Element 对象表示 HTML 元素。Element 对象可以拥有类型为元素节点、文本节点、注释节点的子节点 |
| Attribute | Attribute 对象表示 HTML 属性。HTML 属性始终属于 HTML 元素 |

Document 中常用方法：



Element 中常用属性/方法：





Attribute 中常用属性/方法：



## 应用

### showModalDialog运用

### 显示服务端时间

# Ajax

## 2.0 总结

没有获取某个请求头的方法。

## 由来与描述

Ajax (Asynchronous JavaScript And XML）的缩写。最初由Jesse James Garrett提出；而现在Ajax的覆盖面已经有了进一步的扩展，把允许浏览器与服务器通信而无需刷新当前页面的技术都称为Ajax。

Ajax并不是一种全新的技术，而更像一种技巧，是把过去的几种技术巧妙结合的技巧。真正与Ajax相关的新名词应该是XMLHttpRequest――一个最早在IE 5（ActiveX）中出现，后续在多数浏览器得到支持的用来实现异步通信的对象。Ajax的著名应用，如Google Map，Google Suggest等。

所以Ajax可以表述为：不用刷新整个页面便可与服务器通讯的办法；而可实现这个效果的方式有：

1. Flash
2. 页面框架FrameSet/iFrame
3. XMLHttpRequest：异步通讯对象，是js的一个拓展对象；无需引入任何外部脚本和包可以直接使用。XMLHttpRequest最早出现在ie5浏览器中叫ActiveXObject；后来其它近代浏览器都慢慢引入异步通讯对象都叫XMLHttpRequest；再后来ie在ie7+的版本中也将该对象取名为XMLHttpRequest。而现在；常常所说的ajax指的就是这个XMLHttpRequest对象的使用。

## 原理与技术

### 原理

在页面中触发一个事件，创建一个XMLHttpRequest异步通讯对象，把Http方法（get/post）、目标URL地址与服务器建立连接、发送数据以及监听XMLHttpRequest对象的状态变化以便请求发送情况。此时在页面中用户可以继续进行界面的交互，当服务器发送回响应内容并在请求完成时调用callback()函数，在函数中解析响应内容并通过dom规则动态地更新页面部分内容。

### 技术

在实现上述的交互过程中；涉及到的技术包括：

1. HTML、CSS；标准化地呈现内容
2. DOM；动态的操作文档内容和结构
3. XML；可拓展标记语言，用于客户端与服务器端之间传递信息；(ajax的响应内容载体可为html内容文本、xml、json)
4. JavaScript；综合处理各个技术协同工作

## Ajax实现

### XMLHttpRequest

#### 描述

XMLHttpRequest 异步通讯对象；所有现代浏览器（IE7+、Firefox、Chrome、Safari 以及 Opera）均内建 XMLHttpRequest 对象（IE5 和 IE6 使用 ActiveXObject）。XMLHttpRequest在不同浏览器上的实现是兼容的，所以可以用同样的方式访问XMLHttpRequest实例的属性和方法，与这个实例创建的方法无关。

XMLHttpRequest 用于在后台与服务器交换数据。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

#### 方法 3 + 3

|  |  |
| --- | --- |
| **方法** | **描述** |
| open(method,url,async) | 规定请求的类型、URL 以及是否异步处理请求。  method：请求的类型；GET 或 POST  url：文件在服务器上的位置  async：true（异步）或 false（同步） |
| send(string) | 将请求发送到服务器。  string：仅用于 POST 请求；当为GET时可不填string对应值或者null |
| setRequestHeader(header,value) | 向请求添加 HTTP 头。  header: 规定头的名称  value: 规定头的值  **注意：**POST请求时如需提交在send方法中提交表单值；一般设置：xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded"); |
| abort() | 停止当前请求 |
| getAllResponseHeaders() | 把Http请求的所有响应首部作为键/值对返回 |
| getResponseHeader(header) | 返回指定首部属性的值 |

#### 属性 2 + 2 + 2

|  |  |
| --- | --- |
| **属性** | **描述** |
| onreadystatechange | 存储函数（或函数名），每当 readyState 属性改变时，就会调用该函数。 |
| readyState | 存有 XMLHttpRequest 的状态。从 0 到 4 发生变化。  0: 请求未初始化。 还没有调用 open 方法  1: 服务器连接已建立。 open 方法已被调用，但 send 方法还没有被调用  2: 请求已接收。send 已被调用。请求已经开始  3: 请求处理中。服务器正在发送响应  4: 请求已完成，且响应已就绪 |
| responseText | 服务器的响应文本内容；如纯文本、html等 |
| responseXML | 服务器的响应内容，xml格式，xml dom对象；如可使用xmlDocument.getElementsByTagName("root") |
| status | 服务器发送的每一个响应的首部信息中的一个属性  常用状态码及其含义：  404 没找到页面(not found)  403 禁止访问(forbidden)  500 内部服务器出错(internal service error)  200 一切正常(ok)  304 没有被修改(not modified) |
| statusText | Http的状态响应文本（OK或Not Found等） |

### 实现步骤

实现ajax异步通讯只需要4步：

① 创建XMLHttpRequest对象；

② 设定onreadystatechange属性的回调方法；

③ 调用open()方法；

④ 调用send()方法。

1. **创建XMLHttpRequest对象**

为了应对所有的现代浏览器，包括 IE5 和 IE6，请检查浏览器是否支持 XMLHttpRequest 对象。如果支持，则创建 XMLHttpRequest 对象。如果不支持，则创建 ActiveXObject。

|  |
| --- |
| var xmlhttp;  if (window.XMLHttpRequest){// code for IE7+, Firefox, Chrome, Opera, Safari  xmlhttp=new XMLHttpRequest();  } else {// code for IE6, IE5  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");  } |

1. **设定onreadystatechange属性的回调方法**

在 Ajax 执行过程中，服务器会通知客户端当前的通信状态。这依靠更新 XMLHttpRequest 对象的 readyState 来实现。改变 readyState 属性是服务器对客户端连接操作的一种方式。每次 readyState 属性的改变都会触发 readystatechange 事件。

当 readyState 等于 4 且状态为 200 时，表示响应已就绪：

|  |
| --- |
| xmlhttp.onreadystatechange=function(){  if (xmlhttp.readyState==4 && xmlhttp.status==200){  //做一些事情  //这里可以接收的响应有：readyState、status、responseText、responseXML；  //如：alert(xmlhttp.responseText);  }  } |

**注意**：当要设置返回内容格式为xml时；需要在后台在设置返回内容时，要严格设置XML内容，不然用responseXML获取返回值为null。如：

① 需要加xml文件的头部：

<?xml version='1.0' encoding='UTF-8'?>

② 设置返回值内容类型为：

response.setContentType("text/xml;charset=UTF-8");

③ 注意xml中的各个标签需要开始和结束标签对应

1. **调用open(method,url,async)方法**

XMLHttpRequest与服务器建立连接。

Method：值可为GET或POST（大小写不敏感）；GET(可以在GET请求里通过附加在URL上的查询字符串来发送数据，不过数据大小限制为2000个字符)。若需要向服务器发送较多数据，可用POST。另；为了避免同一个URL在短时间内的缓存问题。可将时间戳或随机数追加到URL的最后，就能确保URL的唯一性，从而避免浏览器缓存结果。

url：路径字符串，指向你所请求的服务器上的那个文件。可以是绝对路径或相对路径。

async：表示请求是否要异步传输，默认值为true。若为true，则不需要等待服务器的响应可读取后面的脚本。若为false，那么当脚本处理过程经过这点时，会停下来，一直等到Ajax请求执行完毕再继续执行。

|  |
| --- |
| **get请求** |
| xmlhttp.open("GET","timeServlet?t=" + Math.random(),true);  xmlhttp.send(null); |

|  |
| --- |
| **post请求** |
| xmlhttp.open("POST","timeServlet",true);  xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");//此句代码应该在open方法之后，send方法之前  xmlhttp.send("fname=Bill&lname=Gates"); |

1. **调用send(data)方法**

将请求发送到服务器。

data：将要传递给服务器的字符串(格式为：param1=xxx&param2=yyy)。若是 GET 请求，则不会发送任何数据，给 send 方法传递 null 即：request.send(null);当向send()方法提供参数时，要确保open()中指定的方法是POST，如果没有数据作为请求体的一部分发送，则使用null。

## 应用

### 异步获取服务器时间

### 二级联动

## 优缺点

优点：按需取数据；最大程度的减少冗余请求；局部刷新页面；减少用户心理和实际的等待时间,带来更好的用户体验；基于xml标准化,并被广泛支持,不需安装插件；进一步促进页面和数据的分离等。

缺点：AJAX大量的使用了javascript和ajax引擎，这些取决于浏览器的支持.在编写的时候考虑对浏览器的兼容性；AJAX只是局部刷新,所以页面的后退按钮会失效；对流媒体还有移动设备的支持不是太好等。

# Json

## 描述

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。它基于JavaScript（Standard ECMA-262 3rd Edition - December 1999）的一个子集。 JSON采用完全独立于语言的文本格式。简单说就是javascript中的对象和数组，所以这两种结构就是对象和数组两种结构，通过这两种结构可以表示各种复杂的结构。JSON 是 JavaScript 原生格式，这意味着在 JavaScript 中处理 JSON 数据不需要任何特殊的 API 或工具包。

## 规则

JSON 语法是 JavaScript 对象表示语法的子集。

* 数据在名称/值对中
* 数据之间由逗号分隔
* 花括号保存对象
* 方括号保存数组

JSON 由对象和数组两种结构组成：

对象：对象在js中表示为“{}”括起来的内容，数据结构为 {key：value,key：value,...}的键值对的结构，在面向对象的语言中，key为对象的属性，value为对应的属性值，所以很容易理解，取值方法为 对象.key 获取属性值，这个属性值的类型可以是 数字、字符串、数组、对象、布尔值或null几种。

数组：数组在js中是中括号“[]”括起来的内容，数据结构为 ["java","javascript","vb",...]，取值方式和所有语言中一样，使用索引获取，字段值的类型可以是 数字、字符串、数组、对象、布尔值或null几种。

## 运用

### 直接定义Json

在javascript代码块中，自定义创建json对象：

|  |
| --- |
| var itcast = {"name":"itcast",  "age":"10",  "info":{"belong":"IT","postCode":"510631"},  "school":[{"location":"beijing"},{"location":"guangzhou"}]  };  alert(itcast.school[0].location1); |

### 解析转换Json

在js中如果是一个字符串的json文本内容；需要使用js的eval函数将文本对象转换为json对象。函数 eval 会把一个字符串当作它的参数；然后这个字符串会被当作 JavaScript 代码来执行。

|  |
| --- |
| var str = '{"name":"itcast","age":"10","info":{"belong":"IT","postCode":"510631"},"school":[{"location1":"beijing"},{"location2":"guangzhou"}]}';  var jso = eval("(" + str + ")");  alert(jso.school[1].location2); |

将原有的ajax二级联动的例子；把返回内容转为json对象并解析。

## 优缺点

优点：作为一种数据传输格式，JSON 与 XML 很相似，但是它更加灵巧。JSON 不需要从服务器端发送含有特定内容类型的首部信息和闭合标签。

缺点：当json文本中包含有函数等不太标准的语法时，无法使用工具查看json结构时去解析会比较困难、不容易阅读。

## 载体小结

若应用程序不需要与其他应用程序共享数据的时候, 使用 HTML 片段来返回数据是最简单的，配合innerHTML很高效。如果数据需要重用, JSON 文件是个不错的选择, 其在性能和文件大小方面有优势。当远程应用程序未知时, XML 文档是首选, 因为 XML 是 web 服务领域的通用语言。