目录

[HTML, XML和XHTML 1](#_Toc4238)

[CSS三大特性 2](#_Toc28745)

[JavaScript的组成 2](#_Toc17005)

[DOM的组成 2](#_Toc31157)

[DOM节点树 2](#_Toc1464)

[网页响应和DOM文档加载的步骤 2](#_Toc23377)

[BOM模型 3](#_Toc16)

[DOM, SAX和DOM4J 3](#_Toc10723)

[dom4j读写XML文档 3](#_Toc2598)

[JavaScript和jQuery 4](#_Toc7766)

[DOM对象和jQuery对象的转换 4](#_Toc6226)

[AJAX原理 4](#_Toc18025)

[AJAX对象常用属性 4](#_Toc26866)

[AJAX常用方法 4](#_Toc14366)

[发送异步请求步骤 4](#_Toc11955)

[AJAX乱码问题 5](#_Toc17752)

[AJAX缓存问题 5](#_Toc24114)

[jQuery对AJAX的支持 5](#_Toc22904)

[JSON 5](#_Toc16792)

[JSON的两种结构 5](#_Toc14606)

# HTML, XML和XHTML

|  |  |  |
| --- | --- | --- |
| HTML：超文本标记语言 | XML：可扩展标记语言 | XHTML：可扩展超文本标记语言 |
| 是[标准通用标记语言](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E9%80%9A%E7%94%A8%E6%A0%87%E8%AE%B0%E8%AF%AD%E8%A8%80)下的一个应用，使用标记标签来描述网页。"超文本"就是指页面内可以包含图片、[链接](https://baike.baidu.com/item/%E9%93%BE%E6%8E%A5" \t "_blank)、音乐、[程序](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F/71525)等非文字元素 | 是[标准通用标记语言](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E9%80%9A%E7%94%A8%E6%A0%87%E8%AE%B0%E8%AF%AD%E8%A8%80" \t "_blank)的子集，是一种用于标记电子文件使其具有结构性的[标记语言](https://baike.baidu.com/item/%E6%A0%87%E8%AE%B0%E8%AF%AD%E8%A8%80) | 是作为一种 XML应用被重新定义的HTML，与HTML4.01几乎是相同的，一个扮演着类似HTML的角色的XML |
| 用来显示数据 | 用来传输和存储数据 | 用来显示数据 |
| 标记是预定义的 | 可以自定义标记 ??? |  |
|  | 元素都须有关闭标签 | |
|  | 元素必须正确地嵌套 | |
|  | 大小写敏感大小写敏感 | |
| 属性值可不加引号 | 属性值须加引号 | |
| 多个连续的空格字符合并为一个 | 多个空格不会合并 | ? |
|  |  | 不支持属性最小化  <… checked="checked"> |
|  |  | 用 id 属性代替 name 属性 |
|  |  | 属性名称必须小写 |

# CSS三大特性

|  |  |
| --- | --- |
| 定义 | 一种用来表现[HTML](https://baike.baidu.com/item/HTML" \t "_blank)或[XML](https://baike.baidu.com/item/XML" \t "_blank)等文件样式的计算机语言。CSS不仅可以静态地修饰网页，还可以配合各种脚本语言动态地对网页各元素进行格式化。 |
| 继承性 | 子元素可以继承父元素的样式，注意点:   1. 不是所有的属性都可以继承，只有以color/font-/text-/line-开头的属性才可以继承 2. 在css中的继承不仅是儿子才可以继承,只要是后代就可以继承 3. css继承性中的特殊性 4. <a>的文字颜色和下划线是不能被继承的 5. <h>的文字大小是不能被继承的 |
| 层叠性 | css处理冲突的能力，在多个选择器选中"同一个标签"，然后又设置了"相同的属性"，后者会将前者覆盖 |
| 优先级 | 默认0 <继承 <通配符 <标签选择器1 <类选择器10 <id选择器100 <行内样式1000 <！important |

# JavaScript的组成

|  |  |
| --- | --- |
| ECMAScript | 一种由[Ecma国际](https://baike.baidu.com/item/Ecma%E5%9B%BD%E9%99%85" \t "_blank)通过ECMA-262标准化的脚本[程序设计语言](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E8%AE%BE%E8%AE%A1%E8%AF%AD%E8%A8%80" \t "_blank)，规定了JavaScript脚本的语法和基本对象（语法、类型、语句、关键字、保留字、运算符和对象）。每个宿主环境（如浏览器）都有它自己的ECMAScript接口的实现，然后这个实现又被扩展，包含了DOM和BOM。 |
| DOM | 定义了JavaScript操作HTML和XML文档的接口，提供了访问文档的途径以及操作方法。浏览器载入该文档后，根据DOM规范，会将文档以节点树形式表示出来。 |
| BOM | 定义了JavaScript操作浏览器的接口，提供了访问某些功能（如浏览器窗口大小、版本信息、浏览历史记录等）的途径以及操作方法。BOM没有相关标准，每个浏览器都有其自己对BOM的实现方式。 |

# DOM的组成

|  |  |
| --- | --- |
| core | 核心部分是结构化文档比较底层对象的集合，这一部分所定义的对象已经完全可以表达出任何HTML和XML文档中的数据了。 |
| HTML接口 | 专为操作具体HTML文档所提供的高级接口。 |
| XML接口 | 专为操作具体XML文档所提供的高级接口。 |

# DOM节点树

|  |  |
| --- | --- |
| 整个文档是一个文档节点 |  |
| 每个 HTML 标签是一个元素节点 |
| 包含在 HTML 元素中的文本是文本节点 |
| 每一个 HTML 属性是一个属性节点 |
| 每一个注释是一个注释节点 |

# 网页响应和DOM文档加载的步骤

|  |
| --- |
| 1. 域名解析 2. 解析html结构 3. 加载外部脚本和样式文件 4. 解析并执行脚本代码 5. 构造HTML DOM模型 // $(document).ready(function) 6. 加载图片等外文件 7. 页面加载完毕 // window.onload |

# BOM模型

|  |  |
| --- | --- |
|  | window对象是ECMAScript规范中的Global对象，  也是BOM中的顶级对象 |
| document对象是BOM顶级对象的一个属性，  也是DOM模型中的顶级对象 |
| location对象是window对象的属性，  也是document对象的属性 |

# JavaScript和jQuery

|  |  |
| --- | --- |
| JavaScript | 一种直译式脚本语言，是一种动态类型、弱类型、基于原型的语言，内置支持类型。它的解释器被称为JavaScript引擎，为浏览器的一部分，广泛用于客户端的脚本语言，最早是在[HTML](https://baike.baidu.com/item/HTML" \t "_blank)网页上使用，用来给HTML网页增加动态功能。 |
| jQuery | 一个快速、简洁的JavaScript框架。它封装JavaScript常用的功能代码，提供一种简便的JavaScript设计模式，优化HTML文档操作、事件处理、动画设计和Ajax交互。 |

原型，原型链，[作用域，作用域链](http://www.baidu.com/link?url=k_MlqjBEHHoWwDvjrHkM2ocQZ-OsmMnbd0sQcwtLLd4d-27rZ-9lb1yW14JZm2WhJN7vHSVTMn9fEvytXJQFxa)

# DOM对象和jQuery对象的转换

|  |  |
| --- | --- |
| 使用JavaScript的方法获取页面中的元素返回的对象就是dom对象。  使用jquery的方法获取页面中的元素返回的对象就是jQuery对象。jQuery对象其实就是DOM对象的包装集。 | |
| DOM转jQuery | var jq = $(dom) |
| jQuery转DOM | 利用数组下标： var dom = jq[0] |
| jQuery内置方法： vra dom = jq.get(0) |

# AJAX原理

|  |
| --- |
| 一种用来改善用户体验的技术，其本质是利用浏览器提供的一个特殊对象（XMLHttpRequest对象，或者简称为ajax对象）向服务器发送异步请求，服务器返回部分数据，浏览器利用这些数据对当前页面做局部更新。整个过程，页面无刷新，不打断用户的操作。 |
| 1. 编写JavaScript程序，发送Ajax请求 2. 编写服务器处理，返回处理结果（常用json格式） 3. 编写JavaScript程序，获取服务器结果，更新部分页面 |

# AJAX对象常用属性

|  |  |  |  |
| --- | --- | --- | --- |
| readyState | ajax对象与服务器通信的进展状态 | 0 | 尚未初始化 |
| 1 | 正在发送请求 |
| 2 | 请求完成 |
| 3 | 请求成功，正在接受数据 |
| 4 | 数据接收成功 |
| onreadystatechange | 绑定一个事件处理函数发生任何状态变化时的事件控制对象 | | |
| responseText | 获得服务器返回的文本数据 | | |
| responseXML | 获得服务器返回的xml数据 | | |
| status | 获得服务器返回的状态码 | | |

# AJAX常用方法

|  |  |
| --- | --- |
| getAllResponseHeaders() | 获取相应的所有Http头 |
| getResponseHeader() | 获取指定的的Http头 |
| setRequestHeader() | 添加指定的Http头 |
| open(get/psost, url) | 创建请求 |
| send() | 发送请求 |
| abort() | 取消请求 |

# 发送异步请求步骤

|  |  |  |
| --- | --- | --- |
| 获取ajax对象 | ie | xhr = new ActiveXObject('MicroSoft.XMLHttp'); |
| 其它 | xhr = new XMLHttpRequest(); |
| 创建请求 | get | xhr.open(请求类型，请求地址，异步或同步); |
| post | xhr.open('post','xxx.do',ture); true表示发送异步请求  xhr.setRequestHeader('content-type','application/x-www-form-urlencoded'); |
| 设置回调函数 | if(xhr.readyState == 4 && xhr.status == 200) {  var txt = xhr.responseText;  …DOM操作… } | |
| 发送请求 | get | xhr.send(null); |
| post | xhr.send(name=Value&name=Value…); |

# AJAX乱码问题

|  |  |  |  |
| --- | --- | --- | --- |
| post | 乱码原因 | 浏览器的Ajax对象对请求参数使用UTF-8编码，服务器默认使用iso-8859-1去解码 | |
| 解决方式 | request.setCharacterEncoding("utf-8"); | |
| get | 乱码原因 | IE的Ajax对象使用GBK对请求参数进行编码，其它浏览器使用UTF-8编码，服务器默认使用iso-8859-1解码 | |
| 解决方式 | 指定解码 | tomcat修改config/server.xml文件中<Connector URIEncoding="utf-8"> |
| 统一编码 | JavaScript内置函数encodeURI()，会使用utf-8对中文进行编码 |

# AJAX缓存问题

|  |  |
| --- | --- |
| IE提供的ajax对象在发送get请求时，会先查看是否访问过地址，如果该地址已经访问过，那么浏览器不再发送请求 | |
| 解决 | 在请求地址后面添加一个随机数 |

# jQuery对AJAX的支持

|  |  |  |
| --- | --- | --- |
| $.ajax({}) | 执行 AJAX请求，该方法通常用于其他方法不能完成的请求 | |
| $.ajax({name:value, name:value, ... }) | |
| url:"random.do" | 请求地址 |
| type:"get" | 请求类型 |
| data:{"name":"Sally","age":22} | 请求参数，字符串或对象 |
| dataType:"text" | 服务器返回的数据类型：json/text/html/xml/script |
| success:fn | 请求成功时执行的函数 |
| error:fn | 请求失败时执行的函数 |
| async:"true" | true是同步，false异步 |
| load() | 从服务器加载数据，并把返回的数据放置到指定的元素中  有请求参数时，load方法发送post请求，否则发送get请求 | |
| $(selector).load(url,data,function(response,status,xhr)) | |
| get() | 使用HTTP GET请求从服务器加载数据 | |
| $.get(URL,data,function(data,status,xhr),dataType) | |
| serialize() | 通过序列化表单值，创建URL编码文本字符串 | |
| $(selector).serialize() // name=ljh&phone=123&address=zs | |

# JSON

|  |
| --- |
| 是一种轻量级的数据交换格式。将要交换的数据先转换成一种与平台无关的数据格式（比如xml???），然后发送给接收方来处理。JSON相对于xml，文档更小，解析速度更快。 |

# JSON的两种结构

|  |  |  |  |
| --- | --- | --- | --- |
| "名称/值"对的集合 | 对象 | 语法 | {name:value, name:value, name:value} |
| 注意 | value可以string, number, true/false, null, object |
| value如果是string，必须使用""括起来 |
| name必须使用""括起来 |
| 值的有序列表 | 数组 | 语法 | [value, value, value] |
| 注意 | value可以是简单数据类型，也可以是object，数组类型 |