# Dom操作

## 章节目标

掌握获取元素的方法；

掌握获取和设置属性的方法；

掌握DOM的常用属性；

了解动态的创建标记；

## 什么是DOM

DOM是文档对象模型(Document Object Model)，是基于浏览器编程的一套API接口，一套对文档的内容进行抽象和概念化的方法。

W3C对DOM的定义是：“一个与系统平台和编程语言无关的接口，程序和脚本可以通过这个接口动态的访问和修改文档的内容、结构和样式”。 今天，几乎所有的浏览器都内置了对DOM的支持。

DOM中的D:

文档。当创建了一个网页并把它加载到web浏览器中时，DOM就在幕后悄然而生，它把你编写的网页文档转化为一个文档对象。

DOM中的O:

对象。对象是一种自足的数据集合。与某个对象相关联的变量称为这个对象的属性；这能通过某个特定的对象去调用的函数称为这个对象的方法。本章我们重点来看一下document对象的属性和方法。

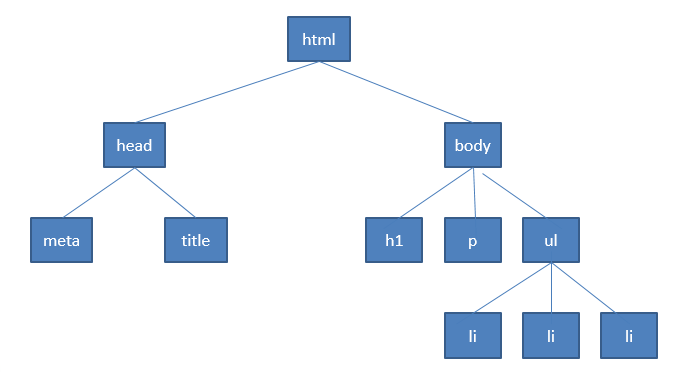
DOM中的M:

模型。含义是某种事物的表现形式。DOM代表着加载到浏览器窗口的当前网页，浏览器则提供了网页的模型，我们可以通过JavaScript去读取这个模型。

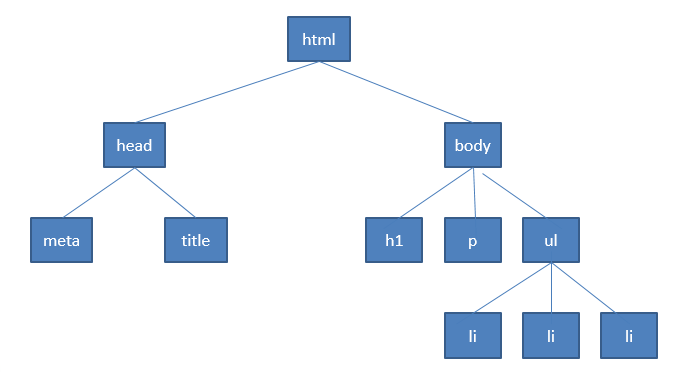
DOM把一篇文档表示成一棵树，如下：

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>Insert title here</title>  </head>  <body>  <h1>你最喜欢哪些明星</h1>  <p title=*"my answer" id="answer"*>我最喜欢的明星有：</p>  <ul id=*"stars"*>  <li>范冰冰</li>  <li>刘德华</li>  <li>王宝强</li>  </ul>  </body>  </html> |

在DOCTYPE之后，一个打开了的<html>标签标识整个文档的开始，网页中的所有其他元素均包含在这个元素中，这个<html>标签就是树根。深入一层，有<head>和<body>两个分支，它们位于同一层次且互不包含，它们是兄弟关系，有着共同的父元素<html>，又有各自的子元素。<head>元素有两个子元素：<meta>和<title>。<body>元素有三个子元素：<h1>、<p>和<ul>。同样，<ul>也是一个父元素，它有三个子元素，它们都是<li>元素。



我们将这个图倒过来不就是一棵树吗：



## 获取元素的方法

有3种方法可获取元素节点，分别是通过元素ID、通过标签名字和通过类名字来获取。

### getElementByID

它是document对象特有的函数。该方法返回一个与那个有着给定id属性值的**元素节点对应的对象**（注意：JavaScript语言区分大小写）。既然是方法，方法名后面必须跟一对圆括号，这里圆括号包含着方法的参数。getElementByID方法只有一个参数：想要获得元素id属性的值，这个id值必须放在单引号或双引号中。



该方法调用将**返回一个对象**（可以用typeof操作符验证这一点）。这个对象对应着document对象里的一个独一无二的元素，那个元素的HTML id属性值是stars。

例：test.js文件中



**注意：该方法只能用于 document 对象**

文档中的每一个元素都是一个对象。利用DOM提供的方法能得到任何一个对象。

我们经常需要与表单打交道，在提交表单时，我们经常需要做很多验证，比如非空验证、长度验证、内容验证等等。进行验证的前提就是我们能够获取表单元素的值。那么怎么样获得呢？首先是要获取到这个表单元素的对象吧，这时候我们的就可以使用这个方法来完成：

|  |
| --- |
| Html文件如下：  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>Insert title here</title>  <script type=*"text/javascript"* src=*"Form.js"*></script>  </head>  <body>  <form action=*""*>  <input type=*"text"* id=*"name"* />  <input type=*"submit"* value=*"提交"* onclick="showValue()">  </form>  </body>  </html>  js文件如下：  **function** showValue()  {  **var** name = document.getElementById("name");  alert(name.value);  } |

### getElementsByTagName

我们知道，并不是每个元素都定义一个独一无二的id值。getElementsByTagName方法可**返回一个对象数组**，每个对象分别对应着文档里有着给定标签的一个元素。该方法也只有一个参数，该参数是标签的名字。（注：不要与上面的方法混淆）

例：



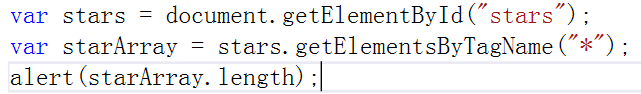
该调用返回一个对象数组，每个对象分别对应着document对象中的一个列表项元素。与其它数组一样，我们也可以利用length属性获得这个数组中元素的个数。



若想知道某份文档中总共有多少个元素节点，可以使用通配符“\*”作为参数：



还可以把上面两种方法结合起来使用，例如：我们想要获得id属性值是stars的元素包含多少个列表项，那么必须通过一个更具体的对象去调用这个方法：



### getElementsByClassName

HTML5 DOM中新增的一个方法。这个方法让我们能通过class属性中的类名来访问元素。



该方法也可以结合getElementById来使用。

### getElementsByName

getElementsByName() 方法可返回带有指定名称的对象的集合

格式：document.getElementsByName(name)

该方法与 getElementById() 方法相似，但是它查询元素的 name 属性，而不是 id 属性。

另外，因为一个文档中的 name 属性可能不唯一（如 HTML 表单中的单选按钮通常具有相同的 name 属性），所有 getElementsByName() 方法返回的是元素的数组，而不是一个元素。

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>getElementsByName</title>  <script type=*"text/javascript"*>  **function** getElements()  {  **var** x=document.getElementsByName("myInput");  alert(x.length);  }  </script>  </head>  <body>  <input name=*"myInput"* type=*"text"* /><br />  <input name=*"myInput"* type=*"text"* /><br />  <input name=*"myInput"* type=*"text"* /><br />  <input type=*"button"* onclick="getElements()"  value=*"一共多少name为'myInput'的元素?"* />  </body>  </html> |

## 获取和设置属性的方法

上一小节，介绍了三种获取特定元素的方法：getElementById、getElementsByTagName、getElementsByClassName。得到需要的元素后我们就可以设法获取它的各个属性。

### getAttribute方法

getAttribute方法只有一个参数，我们要查询的属性的名字。该方法不属于document对象，只能通过元素节点对象调用。比如与getElementById方法合用：

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>Insert title here</title>  <script type=*"text/javascript"*>  **function** getAttr() {  **var** answer = document.getElementById("answer");  alert(answer.getAttribute("title"));  }  </script>  </head>  <body>  <h1>你最喜欢哪些明星</h1>  <p title=*"my answer"* id=*"answer"*>我最喜欢的明星有：</p>  <ul id=*"stars"*>  <li>范冰冰</li>  <li>刘德华</li>  <li>王宝强</li>  </ul>  <input type=*"button"* value=*"获取属性"* onclick="getAttr()">  </body>  </html> |

首先通过getElementById方法获得id是“answer”的元素，然后通过getAttribute方法获得该元素的属性为“title”的值。

### setAttribute方法

上面介绍的方法都是用来获取信息，setAttribute方法允许我们用属性节点的值做修改。与getAttribute一样，setAttribute也只能用于元素节点。

格式：**element.setAttribute(attributeName,attributeValue);**

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>Insert title here</title>  <script type=*"text/javascript"*>  **function** setAttr() {  **var** answer = document.getElementById("answer");  alert(answer.getAttribute("title"));  answer.setAttribute("title", "your answer");  alert(answer.getAttribute("title"));  }  </script>  </head>  <body>  <h1>你最喜欢哪些明星</h1>  <p title=*"my answer"* id=*"answer"*>我最喜欢的明星有：</p>  <ul id=*"stars"*>  <li>范冰冰</li>  <li>刘德华</li>  <li>王宝强</li>  </ul>  <input type=*"button"* value=*"设置属性"* onclick="setAttr()">  </body>  </html> |

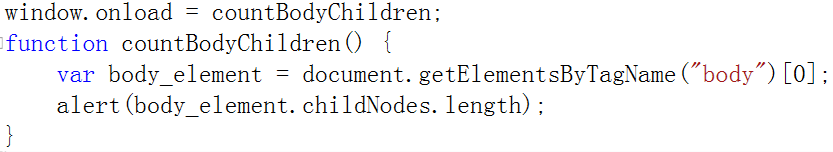
对title属性节点重新赋值为“you answer”。

注意：若要赋值的元素属性不存在，则会先创建这个属性，然后再设置它的值。SetAttribute做出的修改不会反映在文档本身的源代码里。这种“表里不一”的现象源自DOM的工作模式：先加载文档静态内容、再动态刷新，动态刷新不影响文档的静态内容（这正是DOM的真正威力：对页面内容进行刷新却不需要在浏览器里刷新页面）。

## 几个常用DOM属性

### childNodes属性

childNodes属性可以用来获取任何一个元素的所有子元素，是一个包含这个元素全部子元素的数组。例如，我们要获取某个文档的body元素的全体子元素，首先我们可以用getElementsByTagName得到body元素，每份文档只有一个body元素，所以它将是getElementsByTagName(“body”)方法中所返回数组中的第一个元素，这里，我们把它写在了一个方法中：



现在，我们想让这个函数在页面加载时执行，需要使用onload事件处理函数，在代码的末尾添加：



刷新页面，我们可以看到显示的数字和我们预期的好像并不一样。这是因为文档树的节点类型并非只有元素节点这一种。childNodes属性返回的数组包含所有类型的节点，而不仅仅是元素节点。事实上，文档中几乎每一样东西都是一个节点，甚至连空格和换行符都会被解释为节点，它们也全都包含在childNodes属性返回的数组中。所以上例中返回的结果才会比预期大。

### nodeType属性

每一个节点都有一个nodeType属性，可以使用nodeType属性来获取节点类型。nodeType属性共有12种取值，其中仅有三种较常用：

元素节点的nodeType属性值是1；

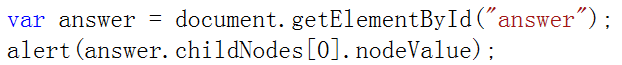
属性节点的nodeType属性值是2；

文本节点的nodeType属性值是3。



### nodeValue属性

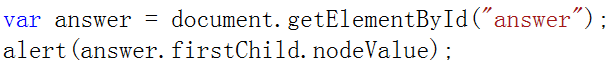
如果想改变一个文本节点的值，那就使用DOM提供的nodeValue属性，它可以用来得到（设置）一个节点的值。



首先，通过id获得p元素，p元素本身的nodeValue属性是一个空值，包含在p元素中的文本是另一种节点，它是p元素的第一个节点。因此，我们要用childNodes[0]获得第一个元素。

### firstChild、lastChild属性

数组元素childNodes[0]有个更直观易读的同义词：firstChild

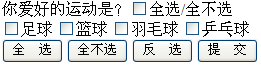


更加简短，且更具有可读性。

DOM还提供了一个与之对应的lastChild属性。代表childNodes数组的最后一个元素。

## 课后作业

### 全选/全不选



当 checkbox 节点有 checked 属性时, 在 HTML 页面上即表现为 "被选中"

1、利用 setAttribute("checked", "checked"); 实现选中, removeAttribute("checked"); 实现不被选中

2、利用 document.getElementById("checkedAll\_2").checked = "checked"; 实现选中, 利用 document.getElementById("checkedAll\_2").checked = null; 实现不被选中

3、利用 document.getElementById("checkedAll\_2").checked = true; 实现选中, 利用 document.getElementById("checkedAll\_2").checked =false; 实现不被选中