带\*符号表示不能兼容到ie8

带#符号表示通常用于document对象

目录

[操作节点本身 4](#_Toc506584081)

[事件相关 4](#_Toc506584082)

[.addEventListener(*event*, *function*, *useCapture*)\* 4](#_Toc506584083)

[.removeEventListener\* 5](#_Toc506584084)

[.focus() 5](#_Toc506584085)

[hasFocus() 5](#_Toc506584086)

[节点相关 5](#_Toc506584087)

[.cloneNode() 5](#_Toc506584088)

[.compareDocumentPosition()\* 5](#_Toc506584089)

[isEqualNode()\* 6](#_Toc506584090)

[isSameNode()\* 已废弃 6](#_Toc506584091)

[.nodeName 6](#_Toc506584092)

[.tagName 6](#_Toc506584093)

[.nodeType 6](#_Toc506584094)

[.nodeValue 7](#_Toc506584095)

[.textContent 7](#_Toc506584096)

[.tabIndex 8](#_Toc506584097)

[其他 8](#_Toc506584098)

[.clientHeight .clientWidth 8](#_Toc506584099)

[.accessKey 8](#_Toc506584100)

[.dir 8](#_Toc506584101)

[.contentEditable 8](#_Toc506584102)

[.lang 8](#_Toc506584103)

[.isContentEditable 9](#_Toc506584104)

[.isDefaultNamespace(url)# 9](#_Toc506584105)

[.namespaceURI\* 9](#_Toc506584106)

[isSupported(feature,version) 9](#_Toc506584107)

[.normalize() 9](#_Toc506584108)

[.offsetHeight .offsetWidth 9](#_Toc506584109)

[.offsetLeft .offsetTop 9](#_Toc506584110)

[.offsetParent 9](#_Toc506584111)

[.scrollHeight .scrollWidth 9](#_Toc506584112)

[.scrollLeft .scrollTop 10](#_Toc506584113)

[.ownerDocument 10](#_Toc506584114)

[.toString() 10](#_Toc506584115)

[操作元素节点 11](#_Toc506584116)

[添加 11](#_Toc506584117)

[.appendChild() 11](#_Toc506584118)

[.insertBefore(*newItem,existingItem*) 11](#_Toc506584119)

[获取 11](#_Toc506584120)

[子节点相关 11](#_Toc506584121)

[父节点相关 12](#_Toc506584122)

[.previousSibling 12](#_Toc506584123)

[.getElementsByTagName() 12](#_Toc506584124)

[. getElementsByClassName()\* 13](#_Toc506584125)

[返回文档中所有指定类名的元素集合，作为 NodeList 对象。 13](#_Toc506584126)

[.nextSibling 13](#_Toc506584127)

[.querySelector() 13](#_Toc506584128)

[.querySelectorAll() 13](#_Toc506584129)

[节点数组 13](#_Toc506584130)

[.item(index) 13](#_Toc506584131)

[.length 13](#_Toc506584132)

[设置（通常也可以获取） 13](#_Toc506584133)

[.innerHTML 13](#_Toc506584134)

[.innerText 14](#_Toc506584135)

[.replaceChild() 14](#_Toc506584136)

[删除 14](#_Toc506584137)

[.removeChild(node) 14](#_Toc506584138)

[获取节点文本值 14](#_Toc506584139)

[1. innerText（兼容性不好） 14](#_Toc506584140)

[2 .firstChild.nodeValue; 14](#_Toc506584141)

[操作属性节点 14](#_Toc506584142)

[普通操作attributes 14](#_Toc506584143)

[.attributes\*？ 14](#_Toc506584144)

[.getAttribute() 15](#_Toc506584145)

[.getAttributeNode() 15](#_Toc506584146)

[setAttribute() 15](#_Toc506584147)

[.setIdAttribute() 15](#_Toc506584148)

[.setIdAttributeNode() 15](#_Toc506584149)

[hasAttribute()\* 15](#_Toc506584150)

[hasAttributes()\* 15](#_Toc506584151)

[.removeAttribute() 15](#_Toc506584152)

[.removeAttributeNode() 15](#_Toc506584153)

[class 15](#_Toc506584154)

[.classlist\* 15](#_Toc506584155)

[方法 16](#_Toc506584156)

[.className 17](#_Toc506584157)

[.id 17](#_Toc506584158)

[.style 17](#_Toc506584159)

[.title 17](#_Toc506584160)

[其他 17](#_Toc506584161)

[.value 17](#_Toc506584162)

[.length 17](#_Toc506584163)

[属性 18](#_Toc506584164)

[.attribute[]（4个） 18](#_Toc506584165)

[.isId 18](#_Toc506584166)

[.name 18](#_Toc506584167)

[.value 18](#_Toc506584168)

[.specified 18](#_Toc506584169)

[.attributes 18](#_Toc506584170)

[.length 18](#_Toc506584171)

[.setNamedItem() 18](#_Toc506584172)

[.item() 18](#_Toc506584173)

[removeNamedItem() 19](#_Toc506584174)

[getNamedItem() 19](#_Toc506584175)

[创建属性 19](#_Toc506584176)

[createAttribute 19](#_Toc506584177)

[document 19](#_Toc506584178)

[宽高 19](#_Toc506584179)

[document.body 19](#_Toc506584180)

[新建节点 19](#_Toc506584181)

[createElement() 19](#_Toc506584182)

[.createTextNode() 20](#_Toc506584183)

[获取节点 20](#_Toc506584184)

[.forms 20](#_Toc506584185)

[.images 20](#_Toc506584186)

[四个get 20](#_Toc506584187)

[.links 20](#_Toc506584188)

[.scripts 20](#_Toc506584189)

[操作文档本身 20](#_Toc506584190)

[.open 20](#_Toc506584191)

[.close 21](#_Toc506584192)

[.write()和.writeln() 21](#_Toc506584193)

[.cookie 21](#_Toc506584194)

[.createComment() 21](#_Toc506584195)

[.doctype 21](#_Toc506584196)

[.title .URL 21](#_Toc506584197)

[节点其他操作 22](#_Toc506584198)

[.activeElement 22](#_Toc506584199)

[.documentElement 23](#_Toc506584200)

[.normalize() 23](#_Toc506584201)

[操作css 23](#_Toc506584202)

[querySelector(All) 23](#_Toc506584203)

[.className 23](#_Toc506584204)

[使用 23](#_Toc506584205)

[封装 23](#_Toc506584206)

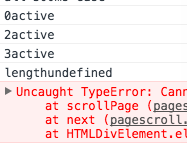
[obj.style. 24](#_Toc506584207)

[nodeList 25](#_Toc506584208)

[基本 25](#_Toc506584209)

[属性 25](#_Toc506584210)

[注意 25](#_Toc506584211)

[ 26](#_Toc506584212)

documtent和element没有直接关联，它们都继承了node,EventTarget

# 操作节点本身

## 事件相关

### .addEventListener(event, function, useCapture)\*

绑定事件

Internet Explorer 8 及更早IE版本不支持 addEventListener() 方法，，Opera 7.0 及 Opera 更早版本也不支持。 但是，对于这些不支持该函数的浏览器，你可以使用 **attachEvent()** 方法来添加事件句柄 (查看 "更多实例" 了解跨浏览器的解决方案)。

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *event* | 必须。要移除的事件名称。.  **注意:**不要使用 "on" 前缀。 例如，使用 "click" ,而不是使用 "onclick"。   **提示:** 所有 HTML DOM 事件，可以查看我们完整的 [HTML DOM Event 对象参考手册](http://www.runoob.com/jsref/dom_obj_event.asp)。 |
| *function* | 必须。指定要移除的函数。 |
| *useCapture* | 可选。布尔值，指定移除事件句柄的阶段。  可能值：   * true - 在捕获阶段移除事件句柄 * false- 默认。在冒泡阶段移除事件句柄   **注意:** 如果添加两次事件句柄，一次在捕获阶段，一次在冒泡阶段，你必须单独移除该事件。 |

### .removeEventListener\*

兼容性和add一样，对应可以使用 **detachEvent()** 方法来移除由 attachEvent() 方法添加的事件句柄

### .focus()

设置文档或元素获取焦点

**提示：**使用 [blur()](http://www.runoob.com/jsref/met-html-blur.asp) 方法来移除元素焦点

### hasFocus()

## 节点相关

### .cloneNode()

该方法将复制并返回调用它的节点的副本。如果传递给它的参数是 true，它还将递归复制当前节点的所有子孙节点。否则，它只复制当前节点。

### .compareDocumentPosition()\*

按照文档顺序，比较当前节点与指定节点的文档位置。

查看以上实例，返回值可能是：

1：没有关系，这两个节点不属于同一个文档。

2： 第一节点（P1）位于第二个节点后（P2）。

4：第一节点（P1）定位在第二节点（P2）前。

8： 第一节点（P1）位于第二节点内（P2）。

16： 第二节点（P2）位于第一节点内（P1）。

32:没有关系的，或是两个节点在同一元素的两个属性。

### isEqualNode()\*

**提示:**使用 [isSameNode() 方法](http://www.runoob.com/jsref/met-node-issamenode.html)来确定两个节点是否相同。

### isSameNode()\* 已废弃

火狐第十版以后不支持，因为该方法在 DOM 4后已经废弃该方法，你可以使用 === 来比较两个元素。

### .nodeName

返回元素的标记名（大写）

### .tagName

作为一个字符串返回某个元素的标记名（大写）

### .nodeType

返回元素的节点类型

如果节点是一个元素节点 , nodeName 属性将返回1。

如果节点是一个属性节点， nodeName 属性将返回2。

|  |  |  |  |
| --- | --- | --- | --- |
| **节点类型** | | **描述** | **子节点** |
| 1 | Element | 一个元素 | Element, Text, Comment, ProcessingInstruction, CDATASection, Entity参考手册 |
| 2 | Attr | 一个属性 | Text, Entity参考手册 |
| 3 | Text | 一个元素的文本内容 或属性 | None |
| 4 | CDATASection | 一个文档的CDATA部分（文本将 不会被解析器解析） | None |
| 5 | Entity参考手册 | 实体引用 | Element, ProcessingInstruction, Comment, Text, CDATASection, Entity参考手册 |
| 6 | Entity | 一个实体 | Element, ProcessingInstruction, Comment, Text, CDATASection, Entity参考手册 |
| 7 | ProcessingInstruction | 一个处理指令 | None |
| 8 | Comment | 一个注释 | None |
| 9 | Document | 整个文档（DOM树的根节点） | Element, ProcessingInstruction, Comment, DocumentType |
| 10 | DocumentType | 为文档实体提供接口 | None |
| 11 | DocumentFragment | 表示邻接节点和它们的子树。 | Element, ProcessingInstruction, Comment, Text, CDATASection, Entity参考手册 |
| 12 | Notation | 代表一个符号在DTD中的声明 | None |

### .nodeValue

根据节点的类型设置或返回节点的值。

**注意：** 如果你想返回元素的文本，记住文本通常是插入到文本节点中，所以**返回的是文本节点的节点值**(element.childNodes[0].nodeValue)。

**提示：** 另一种获取节点值的属性 [textContent 属性](http://www.runoob.com/jsref/prop-node-textcontent.html).

### .textContent

|  |  |
| --- | --- |
|  | 设置或返回一个节点和它的文本内容 |

### .tabIndex

可以改变tab键选取的索引顺序

<p><a id="1" href="http://www.w3cschool.cc">1</a></p>

document.getElementById('3').tabIndex="1";

## 其他

### .clientHeight .clientWidth

在页面上返回内容的可视高度（不包括边框，边距或滚动条）

### .accessKey

设置或返回accesskey一个元素

指的是快捷键

### .dir

设置或返回元素的文字方向。

### .contentEditable

contentEditable 属性用于设置或返回元素的内容是否可编辑。

**提示：** 你也可以使用 [isContentEditable](http://www.runoob.com/jsref/prop-html-iscontenteditable.html) 属性来查看元素的内容是否可编辑。

### .lang

lang 属性设置或返回元素的语言。

<body id="myid" lang="en-us">  
  
<script>  
var x=document.getElementsByTagName('body')[0];  
document.write("Body language: " + x.lang);  
document.write("<br>");  
document.write("An alternate way: ");  
document.write(document.getElementById('myid').lang);

### .isContentEditable

### .isDefaultNamespace(url)#

### .namespaceURI\*

返回命名空间的 URI。

### isSupported(feature,version)

### .normalize()

合并相邻的文本节点并删除空的文本节点。

### .offsetHeight .offsetWidth

返回元素的宽度，包括边框和填充，但不是边距

### .offsetLeft .offsetTop

返回当前元素的相对水平偏移位置的偏移容器，就是元素左边界到父级（有定位）元素左边界的长度

### .offsetParent

返回元素的偏移容器,也就是元素做相对偏移（对应offsetTop）的元素。

### .scrollHeight .scrollWidth

返回整个元素的高度（包括带滚动条的隐蔽的地方）

### .scrollLeft .scrollTop

返回当前视图中的实际元素的顶部边缘和顶部边缘之间的距离,和父级元素无关。

### .ownerDocument

ownerDocument 可返回某元素的根元素。

HTML中, HTML 文档本身是元素的根元素

<p id="demo">单击按钮获取li元素的根文字的节点类型</p>

<button onclick="myFunction()">点我</button>

<script>

function myFunction(){

var x=document.getElementById("demo");

x.innerHTML=x.ownerDocument.nodeType;

}



9表示document

### .toString()

一个元素转换成字符串

# 操作元素节点

## 添加

### .appendChild()

### .insertBefore(newItem,existingItem)

document.getElementById("myList").insertBefore(newItem,existingItem);

## 获取

### 子节点相关

#### .childNodes

返回元素的一个子节点的数组,如果有文本元素会将文本元素一起返回，哪怕里面是一个换行符也算。

#### children

只返回dom元素

#### .firstChild

返回元素的第一个子节点

#### .lastChild

返回最后一个子元素

#### hasChildNodes()

#### contains()

document.body.contains(node)

返回boolean。

### 父节点相关

#### offsetParent

指与位置有关的上级元素

#### .parentNode

返回元素的父节点

#### parentElement(ie)

parentElement 和 parentNode一样,但parentNode是W3C标准的parentElement 只在IE中可用.

### .previousSibling

### .getElementsByTagName()

返回指定标签名的所有子元素集合。

### [. getElementsByClassName()](http://www.runoob.com/jsref/met-element-getelementsbyclassname.html)\*

### 返回文档中所有指定类名的元素集合，作为 NodeList 对象。

### .nextSibling

返回某个元素之后紧跟的节点（处于同一树层级中）。

返回节点以节点对象返回。

**注意：** 如果元素紧跟后面没有节点则返回 null.

### .querySelector()

返回匹配指定 CSS 选择器元素的第一个子元素

### .querySelectorAll()

返回匹配指定 CSS 选择器元素的所有子元素节点列表

## 节点数组

### .item(index)

返回一个节点列表中指定索引的节点。

document.body.childNodes.item(0);

## .length

返回节点列表的节点数目。

## 设置（通常也可以获取）

### .innerHTML

设置或者返回元素的内容。

会替换掉原有的内容，可以先新建一个空节点，将内容放入空节点中。

innerHTML不能执行js!!!，插入的内容里有script元素是不能执行的

### .innerText

### .replaceChild()

替换一个子元素

### 删除

### .removeChild(node)

|  |  |
| --- | --- |
|  | 删除一个子元素 |

## 获取节点文本值

1. innerText（兼容性不好）  
   var obj=document.getElementById("test");  
   //非Mozilla浏览器：  
   obj.innerText;
2. firstChild.nodeValue;

其他的浏览器  
obj.firstChild.nodeValue;

# 操作属性节点

## 普通操作attributes

### .attributes\*？

在 Internet Explorer 8 以及更早的版本中，attributes 属性会返回元素所有可能属性的集合。

### .getAttribute()

### .getAttributeNode()

getAttributeNode() 方法从当前元素中通过名称获取属性节点

### setAttribute()

参数：属性节点

方法用于添加新的属性节点。

如果元素中已经存在指定名称的属性，那么该属性将被新属性替代。如果新属性替代了已有的属性，则返回被替代的属性，否则返回 NULL。

### .setIdAttribute()

### .setIdAttributeNode()

### [hasAttribute()](http://www.runoob.com/jsref/met-element-hasattribute.html)\*

### [hasAttributes()](http://www.runoob.com/jsref/met-element-hasattribute.html)\*

### .removeAttribute()

### .removeAttributeNode()

## class

### [.classlist](http://www.runoob.com/jsref/prop-element-classlist.html)\*



classList 属性返回元素的类名，作为 DOMTokenList 对象。

该属性用于在元素中添加，移除及切换 CSS 类。

classList 属性是只读的，但你可以使用 add() 和 remove() 方法修改它。

document.getElementById("myDIV").classList.add("mystyle");

#### add remove contains item toggle

|  |  |
| --- | --- |
| **属性** | **Description** |
| length | 返回类列表中类的数量  该属性是只读的 |

## 方法

|  |  |
| --- | --- |
| **方法** | **描述** |
| add(class1, class2, ...) | 在元素中添加一个或多个类名。  如果指定的类名已存在，则不会添加 |
| contains(class) | 返回布尔值，判断指定的类名是否存在。  可能值：   * true - 元素包已经包含了该类名 * false - 元素中不存在该类名 |
| item(index) | 返回类名在元素中的索引值。索引值从 0 开始。  如果索引值在区间范围外则返回 null |
| remove(class1, class2, ...) | 移除元素中一个或多个类名。  **注意：** 移除不存在的类名，不会报错。 |
| toggle(class, true|false) | 在元素中切换类名。  第一个参数为要在元素中移除的类名，并返回 false。  如果该类名不存在则会在元素中添加类名，并返回 true。   第二个是可选参数，是个布尔值用于设置元素是否强制添加或移除类，不管该类名是否存在。例如：  移除一个 class: element.classList.toggle("classToRemove", false);  添加一个 class: element.classList.toggle("classToAdd", true);  **注意：** Internet Explorer 或 Opera 12 及其更早版本不支持第二个参数。 |

### .className

设置或返回元素的class属性

### .id

### .style

### .title

# 其他





## .value

获取input的value值。

## .length

不是所有element都有length属性，很多都是undefined(div,option),select的length表示内部option的数量

# 属性

## .attribute[]（4个）

### .isId

是否为id属性

var x=document.getElementById("demo");

x.innerHTML=**x.attributes[0].isId**;

注意：火狐浏览器返回undefined

注意:Internet Explorer 或者 Opera不支持isId属性。

### .name

返回该属性节点的名字（attribute[]只有序号）

### .value

返回该属性节点的值

### .specified

判断该序号属性节点是否被指定值，true，false

## .attributes

### .length

### .setNamedItem()

设置指定属性节点(通过节点)

### .item()

返回节点属性中处于索引号的节点

### removeNamedItem()

删除指定属性节点

### getNamedItem()

从节点列表中返回的指定属性节点。

x.innerHTML=a.attributes.getNamedItem("onclick").textContent;

## 创建属性

### createAttribute

var h=document.getElementsByTagName("H1")[0];

var typ=document.createAttribute("class");

typ.nodeValue="democlass";

h.attributes.setNamedItem(typ);

# document

## 宽高

直接document.clientHeight之类是undefined。

### document.body

## 新建节点

### createElement()

var btn=document.createElement("BUTTON");

大小写都可以

只有document可以。

### .createTextNode()

文本节点

可以通过这个方法给style节点添加内容。（那些样式就是style内的文本）

## 获取节点

### .forms

返回当前页面所有表单的数组集合。

### .images

### 四个get



### .links

links 集合返回当期文档所有链接的数组。

**提示：** links 集合计算 <a href=""> 标签和 <area> 标签。

### .scripts

返回页面中所有脚本的集合。

## 操作文档本身

### .open

var w=window.open();

w.document.open();

w.document.write("<h1>Hello World!</h1>");

w.document.close();

结合起来可以打开一个新窗口，并写入内容。

### .close

var doc=document.open("text/html","replace");

var txt="<!DOCTYPE html><html><body>学习 HTML DOM 很有趣!</body></html>";

doc.write(txt);

doc.close();

### .write()和.writeln()

向文档写 HTML 表达式 或 JavaScript 代码。

### .cookie

返回当前文档所有 **键/值** 对的所有 cookies。

### .createComment()

|  |  |
| --- | --- |
|  | createComment() 方法可创建注释节点。 |

### .doctype

返回与文档相关的文档类型声明 (DTD)。

### .title .URL

.renameNode() 所有浏览器都不支持









## 节点其他操作

### .activeElement

返回当前获取焦点元素

### .documentElement

返回文档的根节点

### .normalize()

删除空文本节点，并连接相邻节点



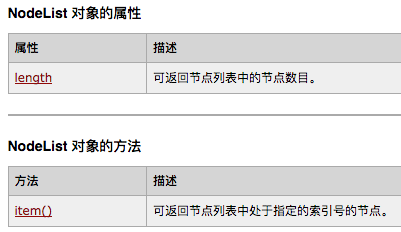
### styleSheets

获取所有样式表数组(不是style节点，而是style节点的sheet)。(对应的CSSStyleSheet对象)

不清楚兼容性

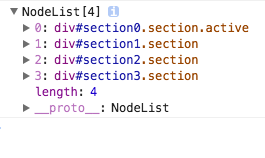
# nodeList

## 基本



也可以直接nodeList[i]

## 属性



## 注意

不要对nodelist用for in

length也包含在其中

# 