window对象下的DOM文档对象的理解

**目录：**

1. 什么是DOM

2. 数据结构

3. DOM树

4. 节点关系

5. 节点类型（总共有12种）

6. 节点的操作

7. 节点的属性，属性有继承性

8. 节点的方法，操作，增删改查，curd

9. node.appendChild() 和 innerHTML、nodeValue、node.cloneNode()的区别，即添加元素的方法的区别

10. 其他node的方法

11. offsetLeft、offsetTop、offsetParent

12. table的DOM操作

14. 节点操作分类

1. 什么是DOM

Document Object Model，文档对象模型

通过操作对象来操作文档；

该模型下提供了很多关于文档操作的方法。

2. 数据结构

**线性结构**。数据串联，一对一，平级，每个数据除了最前面的数据都有前置元素，除了最后面数据后置元素，与各数据相连，数据是平级的。比如买票

应用：队列(先进先出)，栈(后进先出)，链表。

比如：数组

特性：一对一，数据是平级

**树型结构**。一对多，到长的树，树形结构

比如：对象

特性：一个父级，多个元素

**图结构**。多对多

3. DOM树

把文档这种的结构转成对象这种结构，文档中每个标签就类似对象中的每个属性。并且模型给这种对象结构富有相应的特性和方法。

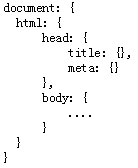
这种结构有一个名词：树 tree

我们把html生成的DOM称为：DOM树 DOM Tree

树结构中的每个数据称为：节点-元素

注意：原生DOM树是浏览器提供的，当DOM修改之后，浏览器马上渲染；

vue框架里面提供了虚拟DOM数，改变DOM之后，与原生的DOM对比，将改变之后内容进行渲染，没有改变的内容不会被覆盖。



4. 节点关系 （节点、子节点、后代节点、根节点）

节点是DOM树中的每一个元素（数据）的总称。（文本，文字注释）

父子关系

兄弟关系

子父关系

祖孙 关系

注意：每个节点的根节点是document

5. 节点类型（总共有12种）不同作用，不同特征

元素类型：1 div,span的标签

属性类型：2 id,class,type

文本类型：3 属性值，‘UTF-8’

注释类型：8 <!---->

文档类型：9 document，根节点，最前面的节点

文档声明类型：10 doctype

参考文档：https://developer.mozilla.org/zh-CN/docs/Web/API/Node/nodeType

6. 节点的操作

属性

方法

7. 节点的属性，属性有继承性

只要是节点类型的，那么都会有一个属性 nodeType

自身属性：

node.nodeType 节点类型，num

node.nodeName 节点名称,''字符串

node.nodeValue 节点值，''字符串

子级属性

node.childNodes 子节点集合， nodeList

node.firstChild 第一个子节点

node.lastChild 最后一个子节点

node.textContent 节点内容

node.children 子元素集合

node.firstElementChild 第一个子元素

node.lastElementChild 最后一个子元素

兄弟属性

node.nextSibling 下一个兄弟节点

node.previousSibling 上一个兄弟节点

node.nextElementSibling 下一个兄弟元素

node.previousElementSibling 上一个兄弟元素

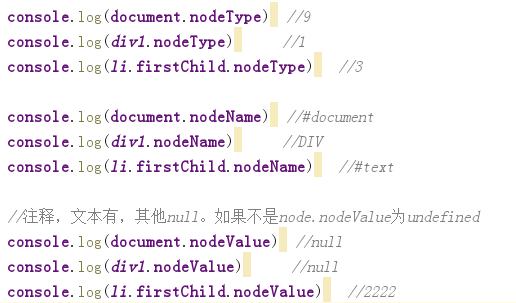
父级属性

node.ownerDocument 所属的文档对象，document

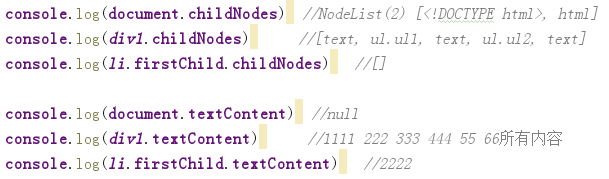
node.parentNode 父级节点node，HTML的父级节点是文本类型document

node.parentElement 父级元素Element

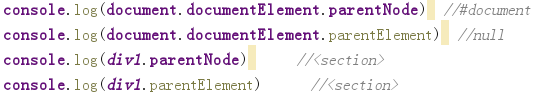
**自身的：**

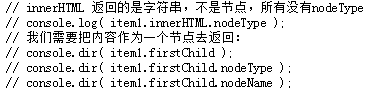


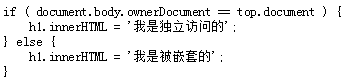
**子级的：**



**父级的：**







8. 节点的方法，操作，增删改查，curd

要搞清父级，子级的关系

document.createElement(tag[,propety]) 新增元素

返回值是新建的**element**。

只有顶处对象**document**有create方法。

因为node有12种，所以每种node都有不同的create方法

createTextNode 创建文本节点

createComment 创建注释节点

注意：创建出来的是js中一个元素对象，这个对象默认和页面中元素没有关系的，我们需要通过后续的一些方法把这个对象渲染到页面中指定的位置

node.appendChild(otherNode) 在父级后面添加元素、或互换子元素

**父.appendChild(原子或新)**

往node节点内添加other；

other可以是document中存在的，也可以是新增的；

**注意：**如果是添加文档中存在的元素，元素原来的位置就会移出，移入新的父级中。

node.insertBefore(newNode,oldNode) 最前边添加元素

**父.insertBefore(新或原子，原子)**

在父级中，往旧元素之前添加新元素

node.replaceChild(newNode,oldNode) 替换元素

**父.replaceChild(新或原子，原子)**

在父级中，newNode替换oldNode

node.removeChild(node) 删除元素

父级中，删除元素

node.cloneNode([deep(true)]) 克隆元素

返回值是新克隆的元素节点。

没有参数，只克隆标签，不克隆内容和方法；

有参数，克隆标签和方法，不克隆方法。

9. node.appendChild() 和 innerHTML、nodeValue、node.cloneNode()的区别，即添加元素的方法的区别

innerHTML 将文本当做HTML解析

属性

看F12中，只刷新父级下面的内容。

存在问题：只是字符串拼接，只复制了结构，没有拷贝js对象的方法

理解：它只是获取了html内容，我们通过innerHTML的方式来更改结构的时候，页面中的页面看起来是没有变化，但是其实是用新的html覆盖了原来的html，只是他们结构样式是一样，不过原来的html对应的对象上的数据全部不存在了。

node.nodeValue

属性，只有文本和注释有，是字符串，其他是null;

node.appendChild() 往父级元素中添加元素。

方法，

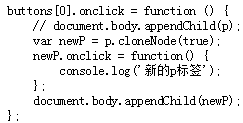
node.cloneNode() 返回克隆后的**节点**，是**对象**

方法。

没有参数，只克隆标签，不克隆内容和方法；

有参数，**深度克隆**，则该节点的所有后代节点也都会被克隆，不克隆方法。

注意，无论有没有参数，都不会克隆方法。



10. 其他node的方法

**node.contents(otherNode)**

返回的是布尔值，判断other是不是node的子元素。

如果 otherNode 是 node 的后代节点或是 node 节点本身.则返回true , 否则返回 false.

https://developer.mozilla.org/zh-CN/docs/Web/API/Node/contains

**node.compareDocumentPosition(otherNode)**

返回的是数字。

可以通过数字判断otherNode和node的关系。

https://developer.mozilla.org/zh-CN/docs/Web/API/Node/contains

**element.hasChildNodes()**

判断是否有子节点

11. offsetLeft、offsetTop、offsetParent

offsetParent:定位父级，自身没有定位也没有关系，没有定位父级就一级级往上找。

自身不用定位，到定位父级的left，top，没有定位父级，就到document的位置。

注意：ie7以下，父级没有定位，有zoom：1；就到该父级。

因为BFC的原因。ie没有BFC，当是有类似BFC的haslayout。

12. table的DOM操作

thead

tfooter

tbodies：一个body有很多tbody，如果没有tbody，js会自动生成一个tbody，用childNodes取时就比较麻烦

rows

cols

clipboard.png

13. prototype 和attribute 属性和属性的区别

prototype 是js对象中的属性

attribute 是HTML中的属性

比如：CheckBox中

在HTML中有属性checked

在js中有checked=true/false；

HTML的属性的设置和获取

设置属性：node.setAttibute('属性名','值')

获取属性：node.getAttribute('属性名')

删除属性：node.removeAttribute ('属性名')

js的属性的设置和获取

设置属性：obj.属性名=属性值

获取属性：obj.属性名 或 obj['属性名'] 或 obj[变量解析成属性名]

14. 节点操作分类

**得到一个DOM对象**——获取的是对象

创建

createElement()

createTextNode()

获取

getElementById()

getElementsByTagName()

getElementsByClassName()

querySelector()

querySelectorAll()

属性获取：

firstChild

lastChild

childNodes

previousSibling

nextSibling

parentNode

克隆

cloneNode()

**操作DOM对象**

获取对象中信息

属性

nodeType

nodeValue

nodeName

tagName

样式：

行间样式：element.style ：width/height/left/top/background

element.currentStyle

element.currentStyle

getComputedStyle(element)

内容的宽高：

clientWidth/clientHeight 宽高+padding

offsetWidth/offsetHeight 宽高+padding+border

clientLeft/clientTop 边框border

定位：

offsetLeft/offsetTop 偏移值

offsetParent 定位父级

getBoundingClientRect().left 到body的left/top

内容：

innerHTML

innerText

outerHTML

textContent

新建：document.createElement('tagName')

增加：node.appendChild(node)，node.insertBefore(newnode,othernode)

修改/替换 : node.replaceChild(newNode,oldNode)

删除：node.removeChild(node)