DOM:串门操作网页内容API标准

DOM Tree:网页中所有内容在内存中都是保存在一棵树型结构中，网页中每项内容都是树上的一个节点对象，唯一树根节点为document

节点对象：网页中每个内容都是DOM树上的一个节点对象

都有的三个属性：nodeType 节点类型 document 9 element 1 attribute 2 text 3

nodeName 节点名称 document #document element 全大写的标签名 attribute 属性名（不常用） text #text

nodeValue 节点值 document null element null attribute 属性值 text 文本内容

查找 4种

不需要查找可直接获得节点

document.documentElement html

document.head head

document.body body

document.forms[id/i] form

按节点间关系查找 已知一个节点找周围节点

节点数 所有都为节点

父子关系

elem.parentNode elem父节点

elem.childNodes elem子节点

elem.firstChild

elem.lastChild

兄弟关系

elem.previousSibling elem的前一个兄弟节点

elem.nextSibling elem的后一个兄弟节点

问题：受到看不见的空字符干扰

元素树，仅包含元素节点的树结构

父子关系

elem.parentElement elem父元素

elem.children elem子元素

elem.firstElementChild

elem.lastElementChild

兄弟关系

elem.previousElementSibling elem的前一个兄弟元素

elem.nextElementSibling elem的后一个兄弟元素

childNodes children 每次都是动态获取 都是动态集合

遍历指定父元素下的所有后代元素

定义函数仅遍历指定父元素下的所有直接子元素

对每个直接子节点调用和父节点完全相同的操作

深度优先

循环：

定义迭代器：可以依次获得每个后代元素节点的专门对象

var iterator = document.createNodelterator(

parent,NodeFilter.SHOW\_ELEMENT, null, false

)

循环调用迭代器，获得下一个节点对象：

var curr = iterator.nextNode(); 深度优先 如果返回null就结束

按HTML查找

按id查找 var elem = document.getElementById()

按标签名查找 var elems = parent.getElementsByTagName();

按name查找 var elems = document.getElementsByName();

按class查找 var elems = parent.getElementByClassName(); 一个class就可以调用

parent：可在任意父元素上吊用

选择器查找：

仅查找一个符合条件的 var elem = parent.querySelector(“选择器”); 返回一个元素 找不到为null 可以相对父元素查找

查找对个符合条件的元素 var elems = parent.querySelectorAll(“选择器”); 集合 找不到为空集合 返回非动态集合

有浏览器兼容问题

DOM操作基本步骤

构建DOM树

查找触发事件的元素

绑定事件

在事件处理函数中，查找要修改的元素

修改元素

按HTML查找和按选择器查找的差别

返回值：HTML返回动态集合 选择器按非动态集合

效率： 首次查找 HTML 效率高 选择器查找效率低

难易：HTML快 选择器慢

简单的用HTML 难的查找用选择器

修改

内容

HTML代码片段 elem.innerHTML 没兼容性问题

纯文本内容 elem.textContent 有兼容性问题

表单元素的值 elem.value

属性

HTML标准属性

核心DOM：最初DOM API 要求支持所有结构化文档

获取属性节点：var attNode = elem.attributes[i]

获取属性值： attrNode.value

var value = getAttrtube();

修改属性值：elem.setAttribute(“属性名”,”值”);

移除属性：elem.removeAttritube(“属性名”);

判断是否包含指定属性：elem.hasAttritube(“属性名”)

HTML DOM 对核心DOM的简化

将属性封装在元素对象上可以用.访问

获取属性值：elem.属性名

修改属性值：elem.属性名 = xxx

移除属性:elem.属性名 =“”

特例：class属性，ES对象中已经有class，所以HTML的class为className

状态属性:disabled selected checked

不能用核心DOM修改；因为是bool类型

只能用HTML DOM . 修改

自定义扩展属性

在元素上保存自定义数据

代替其他选择器，用来查找元素绑定事件

普通属性名 + 核心DOM

HTML5：定义属性时：data-属性名 = 值

获取修改： elem.dataset.属性名

固定用法：用自定义扩展属性代替其他选择器，为元素绑定事件

今后元素事件都要为元素添加自定义扩展属性

data-属性名=‘值’

只能用属性选择器查找：[data-属性名=值]

样式

内联样式elem.style.css 属性名：

去横线变驼峰

不能用elem.style.css获取属性名

用var a = gerComputedStyle();为只读样式

1. 样式

内部、外部

获得样式表对象 var sheet = document.styleSheets[i]

获得样式表中的cssRule:每个{}都是一个cssRule var rule = sheet.cssRules[i]

获得cssRule中的style对象的css属性 var value = rule.style.css 属性

解决：在css中准备好各种class

用程序，在对应情况下，选择对应的class

添加、删除

创建空元素

var a = document.creatElement(‘a’) <a></a>

设置必要属性

a.href a.innerHTML

将新元素添加到DOM树

parent.appendChild(a);将a追加到指定父元素下

parent.insertBefore(a,child)将a插入到指定父元素下的子元素之前

parent.replaceChild(a, child)用a替换指定父元素下的现有子元素

优化：尽量减少操作DOM数的次数，从而减少重排重绘

HTML加载：html DOM tree

↓

render tree layout布局定位阶段 paint

↑

css cssRules

现在内存中加好所有的子父元素，然后再添加到DOM数上

如果父元素已经在页面上了，应该使用文档片段：内存中临时存贮多个子元素的虚拟父元素

创建文旦片段 var frag = document.reateDocumentFragment();

将子元素加入文档片段

将温昂片段追加到父元素中

删除：parent.removeChild(a);

child.parentNode.removeChild(child);

HTML DOM 常用对象

Select

属性

selectedIndex 获得select中的option的下表位置

value option的值没有就innerHTML代替

options option所有的集合

options.length option的个数

.add(option)追加一个option 不支持优化

.remove(i)溢出i位置的option

Option

创建：var opt = new Option(text, value)

属性：.text .value .index

table

创建行分组

var thead = table.reateTHead();

var tbody = table.reateTBody();

var tfoot = table.reateTFoot();

删除行分组

table.deleteTHead()

table.deteteTFoot()

获取行分组

table.tHead table.tFoot table.tBodies[i]

行分组：

添加行 var tr=行分组.insertRow(i); i没有末尾追加 i=0 开头追加

删除行 行分组.deleteRow(i)

table.deleteRow(tr.rowIndex) rowIndex是相对于整个table的tr的属性

获取行 行分组.rows[i]

行

添加格var td = tr,insertCell(i)

删除格tr.deleteCell(i)

获取格tr.cells[i]

From 表单元素

获取 var from = document.forms[i/id]

属性：.elements 获取表单元素中的所有表单元素的集合

.elements.length == .length

方法：.submit() 代替submit，在程序中实现手动提交

获取：form.elements[i/id];或者 form.name

方法：.focus 让表单元素获得焦点

.blur 失去焦点