正课

1. 修改

2. 添加/删除

3. HTML DOM常用对象

1. 修改:

1. 内容:

2. 样式:

1. 获取或设置一个css属性:

只要修改一个css属性: 元素.style.css属性

只要获取一个css属性:

var style=getComputedStyle(元素)

style.css属性

问题: 一次只能修改一个css属性。如果一次要修改多个css属性，就需要写多句话才能完成。

2. 其实，只要批量修改样式，都用class。

3. 属性: 3种属性:

1. HTML标准属性: 2种方式:

1. 核心DOM:

元素.getAttribute("属性名")

元素.setAttribute("属性名","属性值")

元素.hasAttribute("属性名")

元素.removeAttribute("属性名")

2. HTML DOM:

元素.属性

特例: html: <ANY class="">

js: ANY.className

2. 状态属性: disabled checked selected

HTML: <ANY disabled>  
 <ANY checked>

特殊: 加上，就有用，不加，就无效，不需要给属性值。

js中: disabled checked selected 其实都是bool类型。

问题: 核心DOM的四个函数，只支持字符串类型的属性。不支持三大状态属性。

解决: 今后，只要使用三大状态属性，只能用.来访问。且值必须是bool类型

补: css选择器: 状态伪类:

:disabled :checked :selected

分别匹配禁用的元素，选中的元素

3. 自定义扩展属性:

什么是: HTML标准中没有规定的，程序员自发添加的属性。

何时: 2大类

1. 在客户端临时缓存业务数据——减少ajax请求的次数。

2. 代替其他选择器，作为选择触发事件的元素添加行为的标志。——可让行为和样式彻底分离——bootstrap

如何:

1. 添加自定义扩展属性:

普通自定义扩展属性:

<ANY 任意属性名="值"

HTML5中新规定，有福利:

所有自定义扩展属性名前必须加data-前缀

<ANY data-任意属性名="值"

2. 读取或修改自定义扩展属性:

只能用核心DOM:

ANY.getAttribute("属性名")

ANY.setAttribute("属性名","值")

不能用.访问。因为自定义扩展属性，是程序员自发添加到元素上的，DOM标准中没有定义，所以内存中的元素对象上是不包含自定义扩展属性的。而getAttribute()和setAttribute()，之所以可以访问自定义扩展属性，是因为每次都临时去页面上HTML元素代码中查找，而不是在对象中直接读取现成的属性。

3. HTML5福利:

如果添加自定义属性时，添加了data-前缀，则可用dataset自动收集所有data-开头的属性，然后用.访问。比如: <img src="1.png" data-md="1\_mg.png" data-lg="1\_lg.png">，如果想获得或修改md和lg属性的值:

可以: img.dataset.md

img.dataset.lg

强调:

1. dataset会自动收集所有data-开头的属性。不带data-前缀的属性，dataset是收集不到的。

2. dataset.后只需要写data-后的属性名即可。不用再写data-前缀。

4. 用自定义扩展属性作为条件，查找触发事件的元素,可用自定义扩展属性。比如：

<ul>

<li><i data-toggle="tab">click me!</i></li>

<li><i data-toggle="tab">click me!</i></li>

...

要找到所有i，可以:

var is=document.querySelectorAll(

"ul>li>[data-toggle=tab]"

)

2. 添加/删除元素:   
 1. 添加: 3步:

1. 先创建一个空元素

var 新元素对象=document.createElement("标签名")

比如: var a=document.createElement("a")

<a></a>

2. 设置元素的关键属性

a.href="url";

a.innerHTML="文本"

<a href="url">文本</a>

问题: 新创建的元素都没有在DOM树上。自然也就无法显示在网页中。

3. 将新元素添加到DOM树上指定位置下

父元素.appendChild(新元素)

将新元素放在父元素下最后一个直接子元素后——末尾追加

父元素.insertBefore(新元素, 旧元素)

将新元素放在父元素下一个现有旧元素之前

父元素.replaceChild(新元素, 旧元素)

将新元素替换父元素下一个现有旧元素

优化: 尽量减少操作DOM树的次数

为什么: 每操作一次DOM树，都会导致重排重绘。如果数据量大，又频繁操作DOM树，会导致重排重绘不过来。

如何:

1. 如果同时添加父元素和子元素时，应该先再内存中，将所有的子元素都添加到父元素上。最后再一次性将父元素整体添加到DOM树——只需要重排重绘一次。

2. 如果父元素已经在DOM树上了，而要添加多个平级子元素。要借助文档片段对象:

什么是文档片段对象: 内存中临时保存多个平级子元素的虚拟父元素

何时: 父元素已经在DOM树上了，而要添加多个平级子元素。

如何: 3步:

1. 创建文档片段对象:

var frag=document.createDocumentFragment();

2. 将多个平级子元素临时添加到文档片段对象中

frag.appendChild(子元素)

3. 将文档片段整体一次性添加到DOM树上指定父元素下。

父元素.appendChild(frag)

删除元素: 父元素.removeChild(子元素)

英文单词:

1. append 追加

2. Fragment 片段

3. iterable 可遍历的

enumerable 可遍历的