正课:

1. 修改

1. 修改:

3样:

1. 内容: 3个属性:

1. 获取或修改原始HTML片段: 元素.innerHTML

2. 获取或修改纯文本内容: 元素.textContent

vs innerHTML

1. textContent去掉了内嵌的标签

2. textContent将特殊符号翻译为正文

以上两个获得的都是双标记中，开始标记和结束标记之间的内容

问题: 表单元素是单标记，没有结束标记，所以也就没有innerHTML和textContent

3. 获取或设置表单元素的内容: 表单元素.value

2. 属性: 3种:

1. HTML标准属性:

什么是: HTML标准中规定的那些字符串类型的属性。比如: id, title, class, name, value, src, href...

获取或修改HTML标准属性: 2种方式：

1. 早期的核心DOM函数: 4个:

var 属性值=元素.getAttribute("属性名") 获取属性值

元素.setAttribute("属性名","新值") 修改属性值

var bool=元素.hasAttribute("属性名") 判断元素是否包含该属性

元素.removeAttribute("属性名") 移除元素上的指定属性

2. 后来的HTML DOM函数:

什么是HTML DOM: 是在旧版基础上，对部分常用函数和常用对象提供的简化版函数。

做的第一个简化: 每个元素对象中都提前预置好了所有HTML标准属性，只不过，值暂时为""。

今后在操作HTML标准属性时，只要用: 元素.属性名

特例:

html中: <元素 class="类名"

js中不能用.class，因为class是js的关键词。所以，凡是操作html中的class属性，一律更名为className。js中操作className等效于操作HTML中的class。

获取属性值: 元素.属性名

修改属性值: 元素.属性名=值

强调: 只要修改标准属性，值必须是字符串。如果给的不是字符串，DOM会自动转为字符串。

判断是否包含指定属性: 元素.属性名!==""，说明包含。因为所有标准属性的默认值是""。

移除属性: 元素.属性名="";

只要把元素的属性值改为空字符串，则当前属性失效。

问题: 以上函数，只能操作字符串类型的HTML标准属性。无法操作bool类型的HTML属性:

2. 状态属性:

什么是: HTML标准中规定的，值为bool类型的属性: disabled selected checked multiple ...

特征: 所有状态属性在HTML中使用时，都不带=和属性值。加上，就起作用。去掉就失去作用。

如何操作:

不能用核心DOM的4个函数（getAttribute(), setAttribute(), hasAttribute(), removeAttribute()），因为核心DOM的四个函数只能操作字符串类型的属性

只能用.来访问，值都是bool类型:

比如:

禁用一个按钮: btn.disabled=true

相当于:

<button disabled>...

启用一个按钮:btn.disabled=false

选中一个checkbox:

chb.checked=true;

相当于:

<input type="checkbox" checked>

取消选中一个checkbox:

chb.checked=false;

补: CSS中有一组状态伪类:

:disabled :checked :selected

专门用于选择处于某种状态的元素

比如: input:checked 选择的是被选中的input元素

button:disabled 选择的是被禁用的按钮

3. 自定义扩展属性:

什么是自定义扩展属性: HTML标准中没有规定的，程序员自发添加的属性。

何时: 2个典型用途:

1. 在客户端html元素上临时缓存数据

比如: data-target="id值";

2. 代替其他选择器，用来选择触发事件的元素，为元素绑定事件。

比如: data-toggle="dropdown"

如何:

1. 为元素添加自定义扩展属性:

行业规范:

<元素 data-自定义属性名="值"

2. 用自定义扩展属性作为查询条件，只能用属性选择器:

[data-属性名=值]

3. 获取或修改自定义属性的值:

不能用.访问自定义扩展属性:

因为自定义扩展属性是程序员在html标准之外，后天添加的自定义属性。则HTML DOM的元素对象中就不包含这些自定义扩展属性。

1. 可以用核心DOM: ——没有兼容性问题

元素.getAttribute("data-属性名")

元素.setAttribute("data-属性名",值)

为什么: getAttribute()每次都是去HTML代码中查找属性。

2. HTML5标准中规定: 有兼容性问题

可以 元素.dataset.属性名

其中dataset可以自动收集页面上所有data-开头的自定义扩展属性。这就是为什么自定义扩展属性名习惯上都要带data-前缀的原因！

3. 样式:

1. 修改内联样式:

每个元素都有一个style属性，代表html中开始标签中的style属性。

元素的style属性中又包含了所有css样式属性

修改元素对象的style属性中的css样式属性，等效于直接修改html中开始标签中的style=""

比如: 元素.style.backgroundColor="red"

等效于: <元素 style="background-color:red">

再比如: 显示隐藏:

元素.style.display="block"

等效于: <元素 style="display:block"

元素.style.display="none"

等效于: <元素 style="display:none"

强调: 所有带-的css属性，一律去横线，变驼峰命名。

比如: list-style-type -> listStyleType

background-color -> backgroundColor

2. 获取样式：

问题: 元素.style仅包含内联样式，不包含样式表中的样式。所以，如果用元素.style.css属性，也只能获得内联样式，无法获得样式表中定义的样式。

解决: 获得计算后的样式:

什么是计算后的样式: 最终应用到这个元素上的所有样式的集合。

何时: 今后，只要获取样式，都要获取计算后的样式。

如何: 2步:

1. 先获得计算后的样式对象

var style=getComputedStyle(元素)

获得指定元素计算后的完整样式对象

强调: getComputedStyle是内置的全局函数，可直接使用！

返回值: 是一个包含所有css属性的巨大的对象

2. 才是从样式对象中提取想要的css属性:

var 属性值=style.css属性

强调: 计算后的样式，都是只读的。

因为样式的来源不确定，不确定有多少元素正在共享该属性。所以不允许擅自修改。以免牵一发而动全身。

总结: 想修改一个元素的css属性:

元素.style.css属性=值

想获取一个元素的css属性:

getComputedStyle(元素).css属性

问题: 修改样式时，style.css属性一句话只能改一个css属性。而网页中一个效果的变化，可能同时涉及多个css说行。如果用style.css属性=值 修改，代码会很繁琐。而且，加样式时繁琐一次，去掉样式时同样繁琐！

解决: 当批量修改样式时，用className修改是最划算的！

总结:

修改:

1. 内容:

1. 原始html 片段: .innerHTML

2. 纯文本: .textContent

3. 表单元素的内容: .value

2. 属性:

1. HTML标标准属性:

1. 核心DOM: .getAttribute()

.setAttribute()

.hasAttribute()

.removeAttribute()

2. HTML DOM: 也可用.改

2. 状态属性: 只能用.改

3. 自定义扩展属性:

定义: <元素 data-自定义属性名="值"  
 获取或修改:

元素.getAttribute("data-属性名")

元素.setAttribute("data-属性名","值")

HTML5: 元素.dataset.属性名

3. 样式:

1. 修改内联样式: 元素.style.css属性="值"

2. 获取: getComputedStyle(元素).css属性

英文单词:

1. attribute 属性: html中，开始标签中写的HTML属性

property 属性: 内存中，对象上，可用.访问的属性

2. compute 计算

computed 计算后的

3. therefore 因此