正课:

1. 查找:

2. 修改:

3. 添加/删除:

1. 查找:

4. 按选择器查找: 2种情况:

1. 只查找一个符合条件的元素:

var elem=任意父元素.querySelector("选择器")

2. 查找多个符合条件的元素:

var elems=任意父元素.querySelectorAll("选择器")

\*\*\*鄙视: 优化: 按HTML查找 vs 按选择器查找

1. 返回值:

按HTML查找返回动态集合: 不实际存储属性值，每次访问集合内容，都重新查找DOM树

按选择器查找返回非动态集合: 实际存储所有属性值，每次访问集合内容，不需要重新查找DOM树

2. 首次查找效率:

按HTML查找快

按选择器查找慢

3. 易用性:

按选择器查找，简单又强大

按HTML查找，相对繁琐

总结: 1. 当仅靠一个条件就可找到元素时，首选按HTML查找

2. 当查找条件复杂时，首选按选择器查找，简单易用

2. 修改:

内容: 获取或修改元素的内容: 3种:

1. 获取或修改元素的原始HTML片段内容: elem.innerHTML

2. 获取或修改元素的纯文本内容: elem.textContent

vs innerHTML: 1. 将特殊符号翻译为正文

2. 去掉了内嵌标签

3. 获取或修改表单元素的内容: elem.value

属性: 3种:

1. HTML标准属性: HTML标准中规定的元素具有的属性

2种:

1. 核心DOM API: 最初DOM标准中指定的，可以访问所有结构化文档的API

4个:

1. 获取属性值: elem.getAttribute("属性名")

2. 修改属性值: elem.setAttribute("属性名","新属性值")

3. 判断是否包含指定属性: elem.hasAttribute("属性名")

4. 移除属性: elem.removeAttribute("属性名")

2. HTML DOM API: 对常用DOM API提供了简化版本

对于标准属性: HTML DOM已经将所有标准属性都提前内置在了元素对象中，可用.直接访问。如果没有出现在页面上的标准属性值，默认为""

elem.标准属性名

特例: html元素的class属性:

应该换成: elem.className => <elem class="">

因为class是ES中的保留字

2. 四大状态属性:

enabled disabled checked selected

值都不是字符串，而是bool类型

不能用核心DOM 4个API修改

只能用.直接访问。

补: CSS3: 状态伪类选择器:

:selected :checked :disabled :enabled

选中处于某种状态的元素

3. 自定义扩展属性:

什么是: HTML标准中没有规定的，程序员自行添加的属性

何时: 2种:

1. 在客户端临时缓存业务数据，减少向服务器端发送请求的次数

2. 代替其他选择器(id,class,元素)，为元素绑定行为

id 只能选一个元素

元素 限制太死板

class 是样式专用，不稳定

解决: 今后只要为元素添加行为，都应该用自定义扩展属性

bootstrap:

1. 为想要添加行为的元素，指定自定义扩展属性

<a data-toggle="dropdown"

2. 用属性选择器找到有自定义扩展属性的元素绑定行为:

var elems=

document.querySelectorAll("[data-toggle=dropdown]")

for(var elem of elems)

elem.on事件名=function(){ ... }

如何:

1. 定义时: <ANY 自定义属性名="值"

2. 获取或设置自定义属性的值:

不能用.修改，因为内存中的元素对象中不包含自定义扩展属性

用核心DOM API可以访问自定义扩展属性:

elem.getAttribute("属性名")

elem.setAttribute("属性名","新值")

elem.hasAttribute("属性名")

elem.removeAttribute("属性名")

HTML5:

1. 如果在定义自定义扩展属性时加了data-前缀

2. 在程序中: elem.dataset.属性名

其中: dataset可自动收集data-\*开头的属性

样式:

1. 修改一个css样式:

elem.style.css属性名="新值"

其中: style代表元素开始标签中的内联样式style属性

强调: 所有css属性名都要去-，变驼峰命名

2. 获取一个css样式:

错误: elem.style 只能获得内联样式

解决: 今后只要获取css属性值，都用计算后的样式

什么是计算后的样式: 最终应用到元素上的所有css属性的集合。

还将相对单位的属性值计算为绝对单位(px)

如何: 2步:

1. 先获得计算后的完整样式对象

var style=getComputedStyle(elem)

2. 从style中获得想要的css属性值

style.css属性名

强调: 计算后的所有样式，只读。因为计算后的样式来源不确定，不知道多少地方正在同时使用。

总结: 今后只要修改css属性，都用elem.style.css属性=值

今后只要获取css属性，都用getComputedStyle

问题: elem.style，一次只能修改一个css属性

如果批量修改多个css属性时，代码繁琐

解决: 今后只要批量应用样式，都用class