正课:

1. 查找

2. 修改

1. 查找:

按HTML查找:

问题: 每次只能按一个条件查找，如果查找条件复杂，则步骤很繁琐

解决: 选择器:

按选择器查找: 2个API

1. 只查找一个元素:

**var elem=parent.querySelector("选择器");**

2. 查找多个元素;

**var elems=parent.querySelectorAll("选择器");**

强调: 可在任意父元素上调用

返回非动态集合: 实际存储属性值，即使反复访问集合，也不会反复查找DOM树

优: 不会反复查找DOM树

缺: 首次查找速度慢

兼容性受制于浏览器的要求

（非动态集合 改className）

鄙视: 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

按选择器查找，返回非动态集合

2. 首次查找效率: 按HTML查找，快

按选择器查找，慢

3. 易用性: 按HTML查找， 繁琐  
 按选择器查找， 简洁

总结: 如果只靠一个条件，就可找到想要的元素时，首选按HTML查找。

如果查找条件复杂时，首选按选择器查找

2. 修改: 3种:

内容: 3种:

1. 开始标签到结束标签之间的原始HTML代码片段

.innerHTML

2. 开始标签到结束标签之间的纯文本正文

.textContent IE8: .innerText

多做2件事: 1. 去掉所有内嵌标签;

2. 转义字符翻译为正文;

3. 表单元素的内容/值: .value

属性: 3种:

1. HTML标准属性: HTML标准中规定的字符串类型的属性

2种:

1. 核心DOM API:

获取一个属性节点: 所有属性节点都保存在元素的attribute集合中

var attr\_node=elem.attributes[i/属性名]

var value=attr\_node.nodeValue;

获取属性值: elem.getAttribute("属性名")

修改属性值: elem.setAttribute("属性名","属性值")

判断是否包含属性: elem.hasAttribute("属性名")

移除属性: elem.removeAttribute("属性名")

特点: 优: 万能

缺: 繁琐

2. HTML DOM API: 专门操作HTML文档的简化版API

这是对部分API进行了简化

特点: 优: 简单

缺: 不是万能

HTML DOM将所有标准属性，都提前预定义在了元素对象上。可用.直接访问。默认值都是""。

获取属性值: elem.属性名

修改属性值: elem.属性名=属性值

判断是否包含属性: elem.属性名!==""

移除属性: elem.属性名=""

特例: .class属性 要更名为 .className

2. 四大状态属性: enabled disabled checked selected

因为不是字符串类型，所以不能用核心DOM API访问

只能用HTML DOM打.访问

3. 自定义扩展属性:

什么是: 开发人员在开始标签中添加的自定义的属性

何时: 2种:

1. 临时就近缓存业务数据，避免反复请求服务端

2. 代替class，批量选择多个元素

如何: 2种:

1. <ANY 自定义属性名="值"

获取自定义属性的值: 核心DOM的万能API

ANY.getAttribute("自定义属性名");

ANY.setAttribute("自定义属性名","值")

强调: 不能用HTML DOM的.访问

查找: 属性选择器[自定义属性名=值]

2. HTML5: <ANY data-自定义属性名="值"

获取或修改时: ANY.dataset.自定义属性名

查找: 属性选择器[data-自定义属性名=值]

样式: 2种:

1. 内联样式: elem.style.css属性="值"

强调: css属性必须去横线变驼峰

list-style-type -> listStyleType

background-color -> backgroundColor

获取完整样式:

错误: elem.style 只能获得内联样式

正确: 获得计算后的完整样式

var style=getComputedStyle(elem)

总结: 只要修改元素的样式,都用elem.style

只要获得样式，都用getComputedStyle

强调: getComputedStyle获得的计算后的样式是只读

因为属性的来源和共享范围不确定

2. 内部样式表/外部样式表:

3步:

1. 获得样式表对象:

var sheet=document.styleSheets[i]

2. 获得CSSRule对象: 样式表中每个{}

var rule=sheet.cssRules[i]

说明: 如果rule是keyframe，则还需要继续向下找一级

3. 获得CSSRule对象中的css属性值

var value=rule.style.css属性

问题: 每个elem.style.css属性只能修改一个css属性

如果批量应用/撤销多个样式时，代码会很繁琐

解决: 今后，只要批量修改样式，都用class属性