操作节点： appendChild()
insertBefore()
replaceChild()
removeChild()
cloneChild()
类数组对象：arguments，NodeList(childNodes)，HTMLCollection(getElenmentsBy...)

Element类型

可以通过nodeName或tagName获取标签名。有三个方法可以操作其特性：setAttribute(),getAttribute()和removeAttribute()。
Element类型是唯一一个使用attributes属性的节点类型。attributes属性包含一个NamedNodeMap，与NodeList相同，也是一个"动态"集合。元素的每一个特性都由一个Attr节点表示，每个节点保存在NamedNodeMap中。NamedNodeMap对象拥有如下方法：
getNamedItem(name),removeNamedItem(name),setNamedItem(name),item(pos)（获取位于数值pos位置处的节点）

添加动态脚本

```
function loadScript(url){
    var script = document.createElement('script');
    script.type = 'text/javascript';
    script.src = url;
    document.body.appendChild(script);
}
```

添加动态样式

```
function loadStyle(url){
    var link = document.createElement('link');
    link.rel = 'stylesheet';
    link.type = 'text/css';
    link.href = url;
    document.getElementByTagName('head')[0].appendChild(link);
}
```

querySelector()和querySelectorAll()
matchSelector()返回一个布尔类型。

DOM元素有以下五个属性：

- childElementCount
- firstElementChild
- lastElementChild
- previousElementSibling
- nextElementSibling

HTML5

1. 与类相关的扩充：getElementsByClassName
2. classList：是新集合类型DOMTokenList的事例，有length属性及以下几个方法：add(value),contains(value),remove(value),toggle(value)。
3. 焦点管理：document.activeElement，这个属性值会始终引用DOM中当前获得了焦点的元素。元素获得焦点的方法有页面加载、用户输入和在代码中调用focus()方法。文档加载期间，document.activeElement为null；文档加载完成后，document.activeElement中保存的是document.body元素。使用document.hasFocus()方法可以确定文档是否获得了焦点。
4. HTMLDocument的变化
    - readyState，可能的取值有两个，loading和complete。
    - 检测兼容模式还是标准模式：

    ```
    if(document.compatMode == 'CSS1Compat'){
        console.log('Standards mode);
    } else{
        console.log('Quirks mode);
    }
    ```

    - 获取head元素：document.head || document.getElementsByTagName('head')[0]
5. 字符集属性：charset或characterset（火狐）。
6. 自定义数据属性添加data-前缀。通过形如div.**dataset**.appId的形式获取或设置它们的值，中划线改为小驼峰。
7. innerHTML和outerHTML。读模式下，outerHTML会获取元素及其所有子节点的HTML标签；写模式下，outerHTML会根据指定的HTML创建DOM子树，然后用这个DOM子树完全替换调用元素。

内存和性能问题

需要插入大量新HTML标记时，使用innerHTML效率会很高。而当删除标签时，若元素绑定了事件或引用了某个js对象作为属性，上述的方法并不能解除绑定关系，会增加内存占用，因此需要手动删除被替换元素的事件处理程序和js对象属性。

scrollIntoView()

可以在任意元素上调用，通过滚动浏览器窗口或者某个容器元素，调用元素就可以出现在视口中。

事件

事件：文档或浏览器窗口发生的一些特定的交互瞬间。
DOM2级事件规定的事件流包括三个阶段：**事件捕获阶段**、**处于目标阶段**和**事件冒泡阶段**。
事件处理函数中的两个对象：event，可以直接访问事件变量；this，指向事件的目标元素。

```
var EventUtil = {
    addHandler: function(element, type, handler){
        if(element.addEventListener){//DOM2
            element.addEventListener(type, handler, false);
        } else if(element.attachEvent){//IE
            element.attachEvent('on' + type, handler);
        } else{//DOM0
            element['on' + type] = handler;
        }
    },
    getEvent: function(event){
        return event ? event : window.event;
    },
    getTarget: function(event){
        return event.target || event.srcElement;
    },
    preventDefault: function(event){
        if(event.preventDefault){
            event.preventDefault();
        }else{
            event.returnValue = false;
        }
    },
    removeHandler: function(element, type, handler){
        if(element.removeEventListener){//DOM2
            element.removeEventListener(type, handler, false);
        } else if(element.detachEvent){//IE
            element.detachEvent('on' + type, handler);
        } else{//DOM0
            element['on' + type] = null;
        }
    },
    stopPropagation: function(event){
        if(event.stopPropagation){
            event.stopPropagation();
        }else{
            event.cancelBubble = true;
        }
    }
}
```

事件类型

UI事件

- load
- unload
- abort
- error
- select
- resize
- scroll

焦点事件

- blur（不冒泡）
- focus（不冒泡）

鼠标与滚轮事件

- click：单击鼠标主键或者是回车键
- dbclick：双击鼠标主键
- mousedown：用户按下任意鼠标按钮是触发。
- mouseenter：鼠标光标从元素外部首次移动到元素范围之内时触发。不冒泡。鼠标移动到子元素时不触发。
- mouseleave：位于元素上方的鼠标光标移动到元素范围之外时触发。不冒泡。鼠标移动到子元素时不触发。
- mousemove：当鼠标指针在元素内部移动时反复触发。
- mouseout：在鼠标指针位于一个元素上方，然后用户将其移入另一个元素时触发。又移入的另一个元素可能位于前一个元素的外部，也可能是这个元素的子元素。
- mouseover：在指针位于一个元素外部，然后用户将其首次移入另一个元素边界之内时触发。
- mouseup：用户释放鼠标按钮时触发。
- mouseWheel：鼠标滚轮事件。

除mouseenter和mouseleave外，所有鼠标事件都冒泡。

组合键：shiftKey,altKey,ctrlKey,metaKey

相关元素：relatedTarget，在mouseover事件和mouseout事件中有。

鼠标按钮：button属性，有3个值：0表示鼠标主键，1表示中键，2表示次鼠标按钮。IE9有效。IE8及以下版本值表示的含义不同。

三个键盘事件

- keydown
- keypress
- keyup

文本事件

textInput:在文本插入文本框之前会出发textInput事件。

事件委托

```
<div id="container">
    <button id="blue">blue</button>
    <button id="green">green</button>
    <button id="red">red</button>
</div>
<script>
    document.getElementById('container').addEventListener('click', function (event) {
        var target = event.target;
        switch (target.id) {
            case 'blue':
                console.log('blue');
                break;
            case 'green':
                console.log('green');
                break;
            case 'red':
                console.log('red');
                break;
        }
    })
</script>
```

“空事件处理程序”会占用内存影响性能。可能产生“空事件处理程序”的情况有：

- 从文档中移除带有事件处理程序的元素；
- 用innerHTML替换页面中的某一部分；
- 卸载页面时。