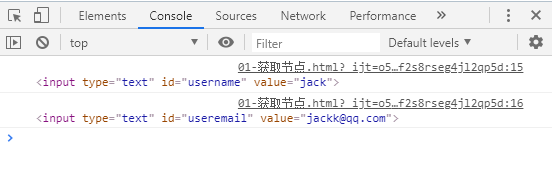
# 获取节点

通过id获取节点getElementsByTagName

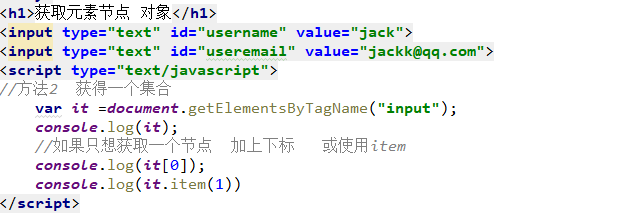
# 

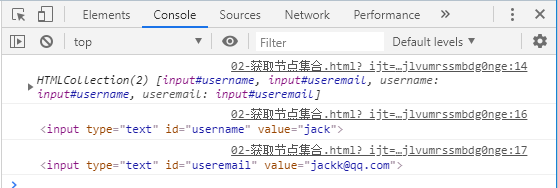


# 通过标签获取节点

# getElementsByTagName

该方法获取的是一个集合列表 （数组）





# 通过name属性值来获取节点





# 文本节点获取

Firstchild或者调用lastchild





# 子节点和兄弟节点

Firstchild、lastchild 父节点获得第一个/最后一个子节点

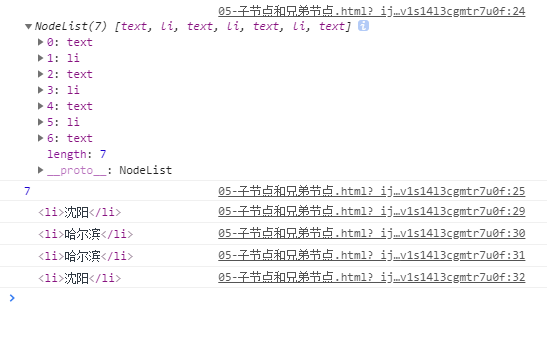
Nextsibling 获得下一个兄弟节点

Pervioussibling获得上一个兄弟节点

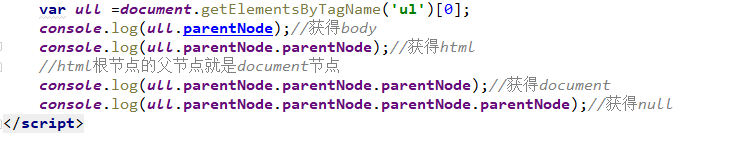
Childnodes父节点获得内部全部的子节点信息

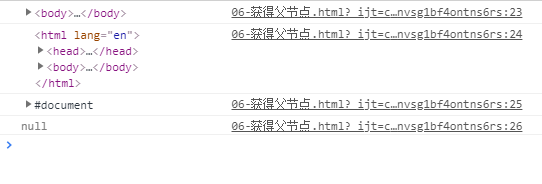
Length 获取“集合列表”的长度





# 父节点





# 属性操作

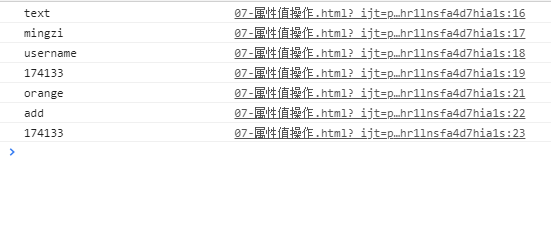
# 属性值操作

## 获取属性值

元素节点node.属性名字 //只能w3c规定的属性

元素节点node.getAttribute（属性名称）//规定的和自定义的都可以获取





## 修改属性值

元素节点node.属性名字=值 //只能w3c规定的属性

元素节点node.setAttribute（属性名称，值）//规定的和自定义的都可以设置





# 获取属性节点

获取节点类型nodetype；

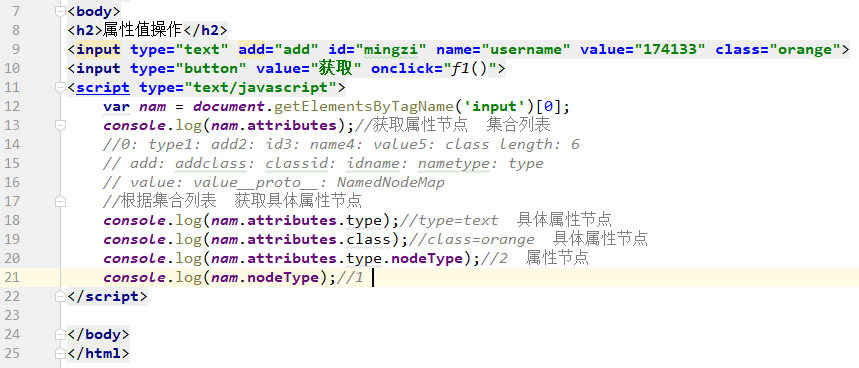
节点.nodetype

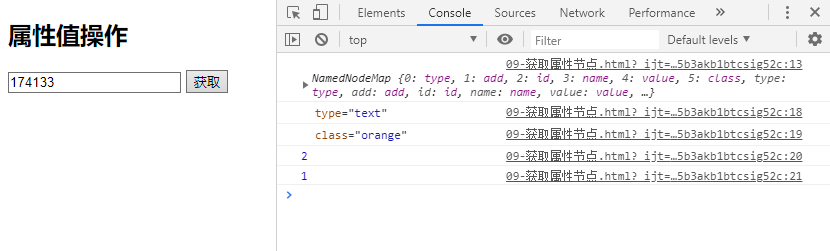
1---》元素节点

2---》属性节点

3---》文本节点

4---》文档节点





# 节点创建和追加

节点创建

元素节点：document.createElement(tag标签名称)

文本节点：document.createTextnode(文本内容)

属性设置node.setAttribute（名称，值）

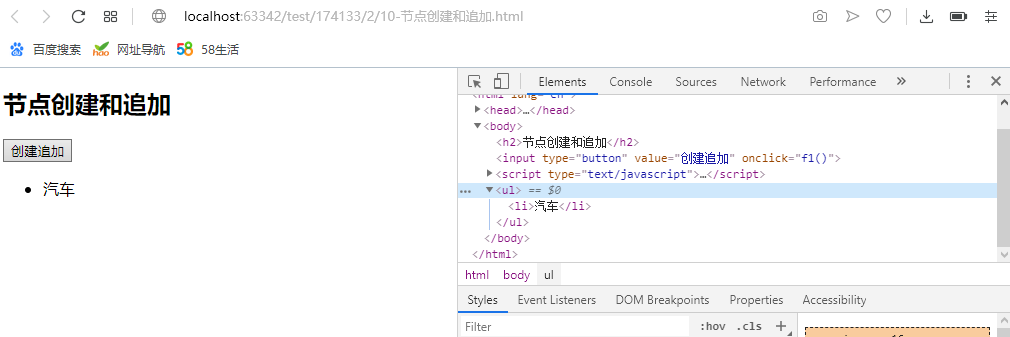
节点追加

父节点.appendchild(子节点)

父节点.insertBefore(newnode，oldnode) newnod放到oldnode的前边

父节点.replacechild（newnode,oldnode）newnod替换到oldnode的节点





# 添加多个节点



# 节点追加到指定位置





# 已有节点追加



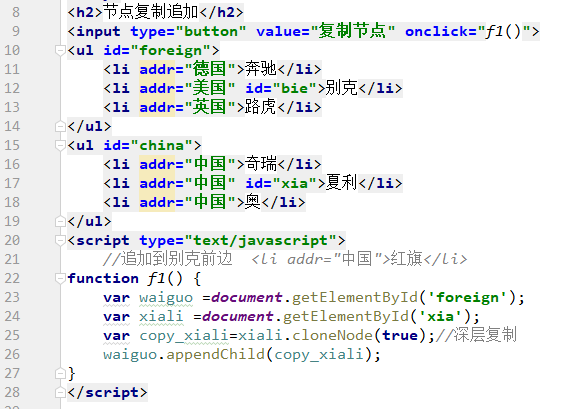


# 节点复制操作

被复制节点.cloneNode（true/false）

True 深层复制（本身节点和内部节点）

False浅层复制（本身节点）



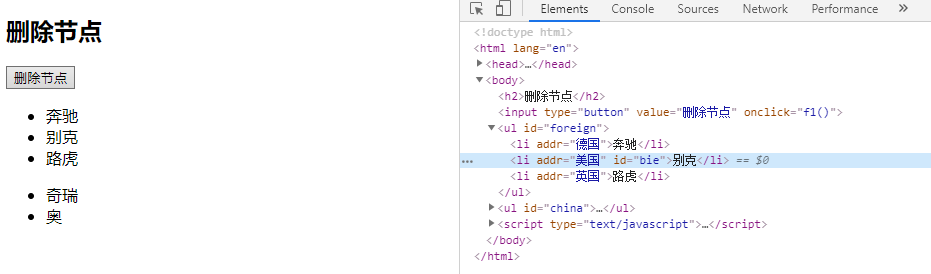


# 节点删除

父节点.removeChild.（子节点）

子节点.parentNode.removeChild（子节点）



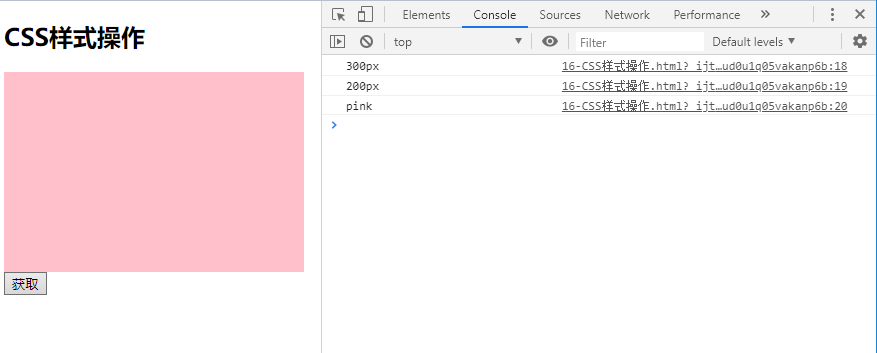


# Css样式的获取

注意：  
 dom操作CSS样式只能操作“行内样式”；

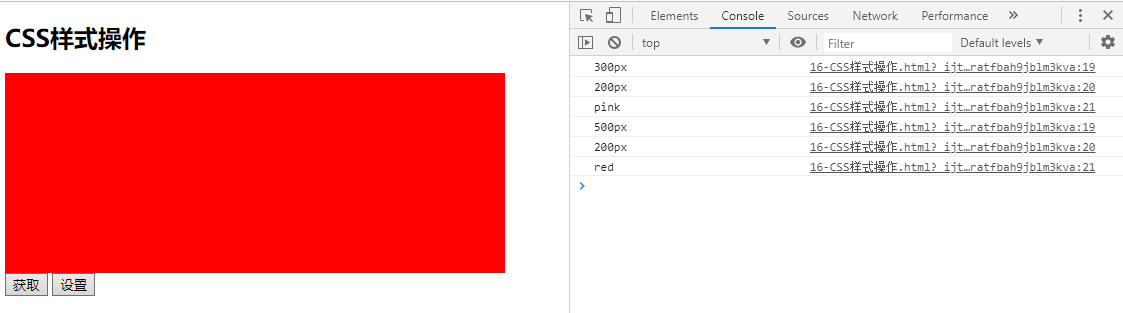
修改样式，有则修改，无则添加，修改后的样式变为行内样式





# 设置样式





# 事件操作

## 什么是事件

通过鼠标，键盘对浏览器页面所做的动作就是事件

事件一旦发生需要有事件处理，该处理称为“事件驱动”，事件驱动通常由函数承担

Onclick鼠标单击

Onmouseover鼠标移入

onmouseout鼠标移出

Onkeyup按键按下并抬起

Onkeydown键盘按下

Onchange内容改变

Onblur失去焦点

Onfocus 获取焦点

Onsubmit表单提交

## 设置事件

①<input type=”text” onclick=”函数名称()” />

function 函数名称(){this[window]}

②<input type=”text” onclick=”过程代码this[itnode]” />

③itnode.onclick = function(){this[itnode]} 匿名函数

④itnode.onclick = 函数名称; 有名函数

function 函数名称(){this[itnode]}

取消dom1级事件：

itnode.onclick = null;

以上是dom1级事件设置的4种具体表现形式，this关键字除了第①种其代表window对象，其他三种都代表事件节点对象本身。





### 2.2 dom2级方式事件设置

1) 主流浏览器方式(包括IE9以上 版本浏览器)：

itnode.addEventListener(事件类型，事件处理[，事件流]); //设置

itnode.removeEventListener(事件类型，事件处理[，事件流]); //取消

2) IE浏览器方式(IE6/7/8)：

itnode.attachEvent(事件类型，事件处理); //设置

itnode.detachEvent(事件类型，事件处理); //取消

事件类型：就是我们可以设置的具体事件，例如onclick/onmouseover等

主流浏览器方式没有”on标志”，例如addEventListener(‘click’，...);

IE浏览器方式有”on”标志，例如attachEvent(‘onclick’)

事件处理：事件驱动，可以是一个有名/匿名 函数

例如addEventListener(‘click’,function(){}/有名函数);

事件流：true捕捉型、[false冒泡型]

事件取消(removeEventListener/detachEvent)操作具体要求：

1. 事件处理 必须是有名函数，不可以是匿名函数。
2. 事件取消的参数与绑定的参数完全一致(数量/内容)

**dom2级事件设置的特点：**

1. 可以为同一个对象设置多个同类型事件。
2. 事件取消也非常灵活。
3. 对事件流也有很好的处理控制。

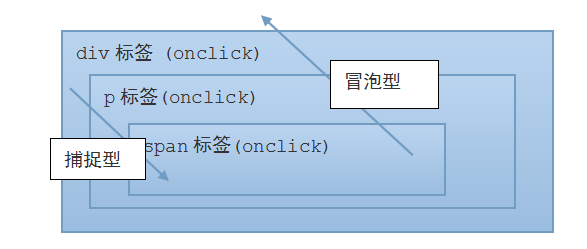
dom2级事件简单设置：

事件取消



## 3. 事件流

多个彼此嵌套元素，他们拥有相同的事件，最内部元素事件被触发后，外边多个元素的同类型事件也会被触发，多个元素他们同类型事件同时执行的效果称为“事件流”



事件流分为两种类型：

冒泡型：事件从内部往外部依次执行。

捕捉型：事件从外部往内部依次执行。

//addEventListener(类型，处理，事件流true捕捉/[false冒泡]);

**IE浏览器**从开始到后期事件流始终是“冒泡型”的，直到IE9以后版本两种都开始支持。

**网景的Navigator浏览器**(现在火狐浏览器的许多血统来源于navigator浏览器)一开始的事件流是”捕捉型”。后期事件流有改进，针对捕捉型、冒泡型都可以支持。