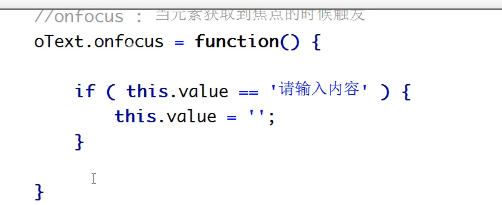
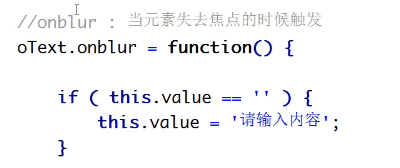
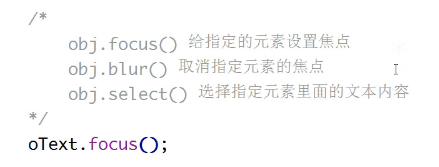
## 事件

### 5.1 焦点事件

焦点概念的提出，可以使浏览器区分用户输入的对象。但是并不是所有的元素（div）都能接收焦点，能够响应用户操作的元素才有焦点(button)。







### 5.2 事件对象

当一个事件发生时，事件的详细信息都会保存到一个对象中，这个对象即event对象。

ie：内置了event对象，默认值是null

chrome：内置了全局event对象，默认值是undefined

火狐：没有内置，只能通过事件函数第一个参数定义

标准浏览器：通过事件函数第一个参数获取

非标准浏览器：event

注意：只有事件方法触发才能捕获到事件对象。

兼容写法：

|  |
| --- |
| **function** *fn*(ev) {  **var** ev = ev || **event**; } |

clientX：当一个事件发生时，鼠标到页面可视区的距离（一直是0，一直到出现垂直滚动条，鼠标到滚动条的距离）

clientY：同上。

|  |
| --- |
| *//案例 div随着鼠标移动而移动 这里需要注意 当窗口出现滚动条后，div移动的距离需要计算滚动条距离* **var *box*** = **document**.getElementById(**'box'**); **document**.onmousemove = **function** (e) {  **var** ev = e || **event**;  **var** scrollTop = **document**.**documentElement**.**scrollTop** || **document**.**body**.**scrollTop**;  **var** scrollLeft = **document**.**documentElement**.**scrollLeft** || **document**.**body**.**scrollLeft**;  ***box***.**style**.**left** = ev.**clientX** + scrollLeft + **'px'**;  ***box***.**style**.**top** = ev.**clientY** + scrollTop + **'px'**; } |

### 5.3 事件冒泡

事件冒泡：当一个元素接收到事件的时候，会把接收到的所有事件传播给父级，一直到顶级window。事件冒泡与元素的显示界面无关，只与元素之间的结构关系有关。

冒泡机制：通常我们给一个元素添加了事件，只是添加了事件处理函数。并不是没有添加事件，就不能接收该事件。同理，冒泡事件中，子级触发了一个点击事件，而父级没有写点击事件的处理函数，那么父级仍然是接收了该事件，只是不知道要如何处理而已。

阻止冒泡：在当前要阻止冒泡的事件函数中调用event.cancelBubble = true;

### 5.4 事件绑定

以前的绑定形式：obj.onclick = fn; obj.onclick = fn2; fn2会覆盖fn1.

新的事件绑定

IE：obj.attachEvent(‘onclick’,fn1);

没有捕获，事件名有on，绑定多事件执行顺序为正序，this为window。

标准：obj.addEventListener(‘onclick’,fn1,false);

有捕获，事件名没有on，执行顺序倒序，this为触发该事件的对象。

false：冒泡方式；true：捕获方式，这是一个可选项，默认冒泡。

由于IE下，事件函数内部的this指向的是window，这是不能容忍的，我们需要修改这个指向，可以使用call()。

fn() 函数直接执行

fn.call() 函数使用call执行

直接执行，call执行在一定意义上是没有区别的，都是执行了fn函数。但是call方法可以传参，且第一个参数可以改变函数执行过程中this的指向，第二个参数就是原函数的参数列表。

fn(1,2)

fn.call(document.1,2)

如果第一个参数是null，代表不修改指向。

兼容写法：

|  |
| --- |
| **function** *bind*(obj,eName,fn) {  **if**(obj.addEventListener){  obj.addEventListener(eName,fn,**false**);  }**else** {  obj.attachEvent(**'on'** + eName,**function** () {  fn.call(obj);  });  } } |

由于函数的限制，IE下是没有捕获的，在标准浏览器下，默认不捕获，使用false，意思就是以冒泡形式触发函数。

<div class =’father>

<div class=’son’>

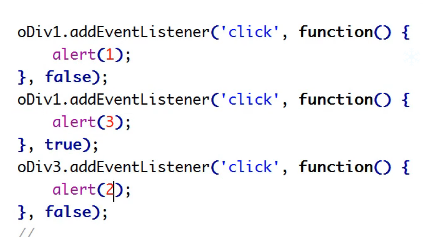
</div>

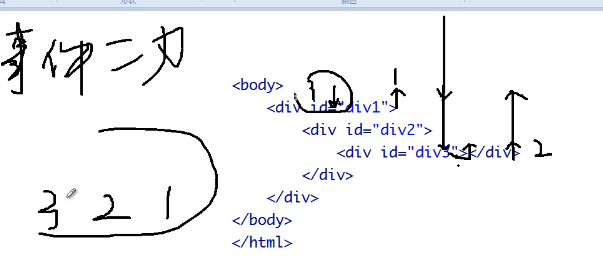
</div>

如果是false，冒泡形式：执行从son开始往外的事件函数

如果是true，捕获形式：执行从father开始往内的事件函数

案例：三个div，div1包裹着div2，div2包裹着div3





解析：

当我们点击div3的时候，事件是从最外层依次往里进入的，在进入时会按照层级依次循环div1，div2，div3是否捕获：

捕获（参数为true）则执行该事件，执行完传递给下一层，

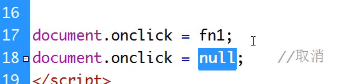
不捕获（参数为false时），直接传递给下一层，

到达真正的点击对象div3时，开始冒泡：

依次查看div3，div2，div1是否执行冒泡。

### 5.4 事件取消

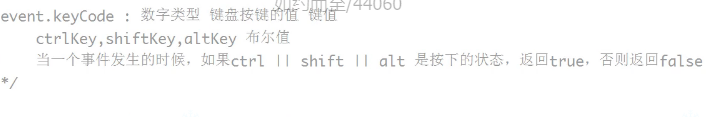
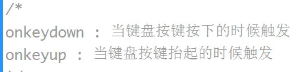
第一种绑定形式取消：



第二种绑定形式取消：

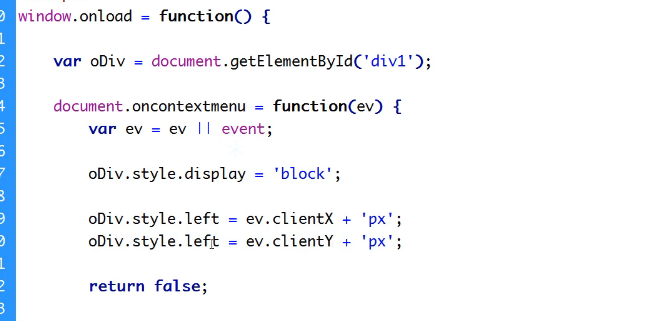


### 5.5 键盘事件



### 5.6 事件默认行为







    addEventListener绑定的事件需要通过event下面的event.preventDefault();来取消默认行为。

### 5.8 鼠标滚轮事件



滚轮属性：wheelDelta \ detail ---鼠标滚轮上下滚动的值

阻止默认事件：绑定写法：preventDefault

### 5.9 mouseover与mouseenter

mouseover/mouseout事件，鼠标经过的时候会触发多次，每遇到一个子元素就会触发一次。

mouseenter/mouseleave事件，鼠标经过的时候只会触发一次。

# 6 拖拽

## 1 拽的三个相关事件

onmousedown：选择元素

onmousemove：移动元素

onmouseup：释放元素

## 2 拖拽的初步实现

<style>  
 #div {  
 width: 100px;  
 height: 100px;  
 background-color: #5cb85c;  
 position: absolute;  
 }  
 </style>  
  
<div id="div"></div>

<script>  
 var div = document.getElementById('div');  
  
 div.onmousedown = function (ev) {  
  
 var ev = ev || event;  
 var disX = ev.clientX - this.offsetLeft;  
 var disY = ev.clientY - this.offsetTop;  
  
 *//在文档中一定，只要鼠标不离开文档，都会移动* document.onmousemove = function (ev) {*//计算移动在中的距离* var ev = ev || event;  
 div.style.left = ev.clientX - disX + 'px';  
 div.style.top = ev.clientY - disY +'px';  
 };  
  
 document.onmouseup = function () {  
 document.onmousemove = document.onmouseup = null;  
 };  
 }  
</script>

这里需要注意的地方：

1. 可拖拽的元素必须要有 定位 absolute
2. 上述代码中，拖拽速度过快会引起鼠标离开元素的bug

解决这个办法的方案是：将div内部的拖拽相关事件对象改为document

## 3 拖拽BUG

有时候，用户在拖拽时会额外将拖拽控件附近的文字等一起进行了选中，会引起文字、控件移动BUG。

解决方案----阻止默认行为：

文字被拖拽了是鼠标的down事件触发的，那么我们可以在down事件内阻止这个默认行为即可，即添加 return false。

但是这个解决方案只能在标准浏览器下实现，IE不行。在IE下有这样一个方法：setCapture() 可以设置全局捕获。即：当我们给一个元素设置全局捕获后，这个元素会捕获到所有其他元素后续发生的所有事件，事件发生时，都由设置了该方法的元素触发！

ie：有setCapture()，且有效果

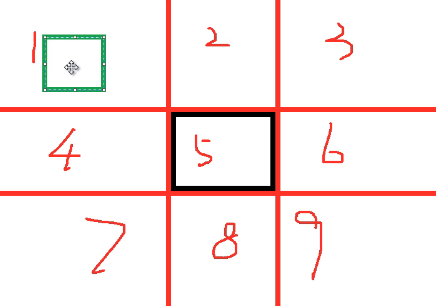
ff：有setCapture()，但没有效果

chrome：没有

|  |
| --- |
| **var *box*** = **document**.getElementById(**'box'**); ***box***.onmousedown = **function** (e) {  **var** e = e || **event**;  **var** disX = e.**clientX** - **this**.**offsetLeft**;  **var** disY = e.**clientY** - **this**.**offsetTop**;  **if**(***box***.setCapture){  *//解决IE下选中文字等其他元素后拖拽BUG*  ***box***.setCapture();  }  *//使用document是以为使用box会出现拖拽过快，动画跟不上*  **document**.onmousemove = **function** (e) {  **var** e = e || **event**;  ***box***.**style**.**left** = e.**clientX** - disX + **'px'**;  ***box***.**style**.**top** = e.**clientY** - disY +**'px'**;  };  **document**.onmouseup = **function** () {  **document**.**onmousemove** = **null**;  **if**(***box***.setCapture){  ***box***.releaseCapture(); *//抬起后释放所有 事件指向box* }  };  **return false**; *//解决 标准浏览器下 按住div前选中了文字 进行拖拽引起的bug* } |

## 4 碰撞检测

九宫格碰撞检测原理图：



判断： 当div的右侧位置小于九宫格左侧竖线位置---不碰撞

当div的左侧位置大于九宫格右侧竖线位置---不碰撞

当div的下方位置小于九宫格上方横线位置---不碰撞

当div的上方位置大于九宫格下方更显位置---不碰撞

具体代码见Code文件。