事件1

事件

鼠标事件

获取事件对象

我们可以通过事件对象来拿到鼠标事件、键盘事件所携带的信息。获取事件对象可以直接通过`window.event`拿到这个对象，但是由于火狐浏览器并不支持这个方法，所以我们一般需要在触发事件的函数中传入`event`参数，以适配火狐。

示例：获取事件对象

```
        document.onmouseover = function  (ev) {
            var evObj = ev || window.event;

            console.log(evObj);
        }
```

获取鼠标当前的坐标：clickX\clickY

`event.clientX`在可视区中，鼠标点击的x坐标
`event.clientY`在可视区中，鼠标点击的y坐标

示例：获取用户点击的坐标点

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <script type="text/javascript">
        document.onclick = function  (ev) {
            var evObj = ev || window.event;
            alert(evObj.clientX + ',' + evObj.clientY);
        }
    </script>
</head>
```

```
<body>

</body>
</html>
```

练习：一个跟随鼠标指针移动的红色块

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title></title>
    <style type="text/css">
        #div {
            width: 100px;
            height: 100px;
            background-color :red;
            position: absolute;
        }

    </style>
    <script type="text/javascript">
        // 鼠标移动时触发改事件
        document.onmousemove = function  (ev) {

            // 获取距离文档顶部的高度
            var oScrollTop = document.documentElement.scrollTop || docu
ment.body.scrollTop;
            // 获取距离文档左边的的宽度
            var oScrollLeft = document.documentElement.scrollLeft || do
cument.body.scrollLeft;

            // 获取鼠标事件
            var oEvent = event || ev;
            // 获取到div
            var oDiv = document.getElementById('div');

            // 设置div的位置
            oDiv.style.left = oEvent.clientX + oScrollLeft + 'px';
            oDiv.style.top = oEvent.clientY + oScrollTop + 'px';

        }
    </script>
</head>
<body>

    <div id = "div"></div>

</body>
</html>
```

练习：一串跟着鼠标移动的div块（同时回顾鼠标移动事件）

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title></title>
    <style type="text/css">
        #div {
            width: 10px;
            height: 10px;
            background-color :red;
            position: absolute;
        }

    </style>
    <script type="text/javascript">
        // 鼠标移动时触发改事件
        window.onload = function  () {

            var oDivs = document.getElementsByTagName('div');

            document.onmousemove = function  (ev) {
                var oEvent = ev || event;

                oDivs[0].style.left = oEvent.clientX+'px';
                oDivs[0].style.top = oEvent.clientY+'px';

                for (var i = oDivs.length-1; i > 0; i--) {
                    oDivs[i].style.left = oDivs[i-1].style.left;
                    oDivs[i].style.top = oDivs[i-1].style.top;

                };


            }


        }
    </script>
</head>
<body>

    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
```

```
        <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
        <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
        <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
        <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>
    <div id = "div"></div>


</body>
</html>
```

回顾：鼠标单击事件：onclick

点击鼠标左键，会触发该事件，例如：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <input type="button" value="按钮" id="btn1" />
    </body>
    <script type="text/javascript">
        var btn1 = document.querySelector("#btn1");
        btn1.onclick = function (){
            alert('a');
        }
    </script>
</html>
```

鼠标按下（onmousedown）和抬起（onmouseup）事件

鼠标的点击可以分为两个更具体的事件：按下，抬起。

鼠标按下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <input type="button" value="按钮" id="btn1" />
    </body>
    <script type="text/javascript">
        var btn1 = document.querySelector("#btn1");
        btn1.onmousedown = function (){
            alert('a');
        }
    </script>
</html>
```

按下后抬起：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <input type="button" value="按钮" id="btn1" />
    </body>
    <script type="text/javascript">
        var btn1 = document.querySelector("#btn1");
        btn1.onmouseup = function (){
            alert('a');
        }
    </script>
</html>
```

练习：使用鼠标拖拽一个div

(入门版:隐藏一个bug:当鼠标移动速度过快，离开div，则div不再跟随)

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <style type="text/css">
            #div1{
                width: 100px;
                height: 100px;
                background-color: #ccc;
                position: absolute;
            }
        </style>
    </head>
    <body>
        <div id="div1"></div>
    </body>
    <script type="text/javascript">
        var div1 = document.querySelector("#div1");

        div1.onmousedown = function (ev){

            var ev = ev||window.event;
            var disX = ev.clientX-div1.offsetLeft;
            var disY = ev.clientY-div1.offsetTop;

            div1.onmousemove=function (ev){

                var ev = ev||window.event;
                var x = ev.clientX-disX;
                var y = ev.clientY-disY;

                div1.style.left = x+"px";
                div1.style.top = y+"px";
            }
            div1.onmouseup=function (){
                div1.onmousemove = null;
            }
        }
    </script>
</html>
```

(优化:把鼠标事件添加在document上即可解决)

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
```

```
        <style type="text/css">
            #div1{
                width: 100px;
                height: 100px;
                background-color: #ccc;
                position: absolute;
            }
        </style>
    </head>
    <body>
        <div id="div1"></div>
    </body>
    <script type="text/javascript">
        var div1 = document.querySelector("#div1");

        div1.onmousedown = function (ev){

            var ev = ev||window.event;
            var disX = ev.clientX-div1.offsetLeft;
            var disY = ev.clientY-div1.offsetTop;

            document.onmousemove=function (ev){

                var ev = ev||window.event;
                var x = ev.clientX-disX;
                var y = ev.clientY-disY;

                div1.style.left = x+"px";
                div1.style.top = y+"px";
            }
            document.onmouseup=function (){
                document.onmousemove = null;
            }
        }
    </script>
</html>
```

鼠标双击事件：ondblclick

连续点击两次鼠标左键，则会触发鼠标的双击事件，如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
    </body>
    <script type="text/javascript">
```

```
    // 双击
        document.ondblclick=function (){
            alert('a');
        }
    </script>
</html>
```

右击事件:oncontextmenu

点击鼠标右键，会弹出一个菜单，这就是出发了系统默认的右击事件

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
    </body>
    <script type="text/javascript">
        document.oncontextmenu = function (){

            alert('a');
            // 阻止默认事件（后边会讲）
            return false;
        }
    </script>
</html>
```

键盘事件

按下按键：keydown

按下键盘上的按键，不抬起按键，会触发该事件。

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
    </body>
    <script type="text/javascript">
        document.onkeydown = function (ev){
```

```
//          alert('a');
            var ev = ev||window.event;

            console.log(ev);
        }
    </script>
</html>
```

keydown携带的一个参数是：keyCode，这个参数里的是每个按键的编码，我们可以通过编码来判断用户按的是哪个按键

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
    </body>
    <script type="text/javascript">
        document.onkeydown = function (ev){
//          alert('a');
            var ev = ev||window.event;

//          console.log(ev);
//          // 获取都用户按下按键的编码
            alert(ev.keyCode);
        }



    </script>
</html>
```

按下后抬起按键:keyup

按下按键，抬起，触发`keyup`事件：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
    </body>
```

```
    <script type="text/javascript">
        document.onkeyup = function (ev){
//          alert('a');
            var ev = ev||window.event;

//          console.log(ev);
            alert(ev.keyCode);
        }
    </script>
</html>
```

练习：使用键盘的上下左右键控制div移动

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style type="text/css">
        #div1 {
            width: 100px;
            height: 100px;
            left:10px;
            top:10px;
            background-color: gray;
            position: absolute;
        }
    </style>

    <script type="text/javascript">
        document.onkeydown = function  (ev) {
            var oEvent = ev || event;
            var oDiv = document.getElementById('div1');

            if (oEvent.keyCode == 37) {
                oDiv.style.left =  oDiv.offsetLeft - 10 +'px';
            } else if (oEvent.keyCode==38) {
                oDiv.style.top = oDiv.offsetTop - 10 +'px';
            } else if (oEvent.keyCode == 39) {
                oDiv.style.left = oDiv.offsetLeft + 10 +'px';
            } else if (oEvent.keyCode == 40) {
                oDiv.style.top = oDiv.offsetTop + 10 +'px';
            };
        }

    </script>
</head>
<body>
    <div id = "div1"> </div>
</body>
```

```
</html>
```

一些特殊的按键

- ctrlKey 返回boolean值，按下时为true
- shiftKey 返回boolean值，按下时为true
- altKey 返回boolean值，按下时为true

按下以上的按键，并不会返回键盘编码，而是布尔值。我们经常在论坛中发帖，可以按着control+回车键，就能直接发送蚊子了，我们就可以捕捉这两个按键返回的值来做对应的处理。

示例：按住control+enter键，提交留言框中的文字到留言板中

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>

    <script type="text/javascript">
        window.onload = function  () {
            var oText1 = document.getElementById('text1');
            var oText2 = document.getElementById('text2');
            var oBtn = document.getElementById('btn');

            // 鼠标点击提交按钮。进行留言
            oBtn.onclick = function  () {
                //点击提交按钮后，把留言框中的文字提交在留言区
                oText1.value += oText2.value + '\n';
                // 清空留言框
                oText2.value = '';
            };

            // 按下control+enter按钮，进行留言,因为当前焦点在留言框中，所以事件要加载留言框中
            oText2.onkeydown = function  (ev) {
                var oEvent = ev || event;

                // 按下回车键和control键
                if (oEvent.keyCode == 13 && oEvent.ctrlKey) {
                    //点击提交按钮后，把留言框中的文字提交在留言区
                oText1.value += oText2.value + '\n';
                // 清空留言框
                oText2.value = '';
                };
            }
        }
```

```
    </script>
</head>
<body>
    <textarea id = 'text1' roes = '10' cols = '40'></textarea>
    <br />

    <input type = "text" id = 'text2'/>
    <input type="button" value = '提交' id = 'btn'/>


</body>
</html>
```

练习：类似于qq的聊天

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <input type="text" id="txt1" />
        <ol id="ol1"></ol>
    </body>
    <script type="text/javascript">
        var txt1 = document.querySelector("#txt1");
        var ol1 = document.querySelector("#ol1");

        txt1.onkeydown = function (ev){
            var ev = ev||window.event;
//          console.log(ev);
            if (ev.keyCode == 13&&ev.ctrlKey == true){
//              alert('a');
                var li = document.createElement("li");
                li.innerHTML = this.value;

                ol1.insertBefore(li,ol1.firstElementChild);

            }
        }
    </script>
</html>
```

表单事件

点击submit按钮触发的事件

在表单中，通过点击submit按钮可以将表单中的额内容提交到指定的URL中，我们也可以指定submit触发的事件，进行自定义操作。

示例：点击submit按钮，弹出提示框

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <form id="form1" action="http://www.baidu.com" method="post">

            <input type="submit" value="提交"/>
        </form>
    </body>
    <script type="text/javascript">
        var form1 = document.querySelector("#form1");

        form1.onsubmit = function (){

            alert('a');
        }
    </script>
</html>
```

阻止默认事件

上边的示例虽然弹出了提示款，但是页面还是跳转到了百度，因为submit按钮默认就是执行表单提交的。如何做才能不让他跳转呢？这就需要我们拦截系统默认事件。我们可以通过`return false`来拦截系统默认的事件。

示例：拦截submit默认事件:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <form id="form1" action="http://www.baidu.com" method="post">
            <input type="text" name="user" id="user" />
            <input type="submit" value="提交"/>
        </form>
    </body>
```

```
    <script type="text/javascript">
        var form1 = document.querySelector("#form1");
        var user = document.querySelector("#user");
        form1.onsubmit = function (){

            if (user.value==""){
                alert("用户名不能为空");
                //阻止默认事件
                return false;
            }
        }
    </script>
</html>
```

除了返回一个false值，我们还可以使用`preventDefault`来拦截，只是`preventDefault`对ie6--ie8不兼容

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <form id="form1" action="http://www.baidu.com" method="post">
            <input type="text" name="user" id="user" />
            <input type="submit" value="提交"/>
        </form>
    </body>
    <script type="text/javascript">
        var form1 = document.querySelector("#form1");
        var user = document.querySelector("#user");
        form1.onsubmit = function (ev){

            var ev = ev||window.event;
            if (user.value==""){
                alert("用户名不能为空");
                //阻止默认事件
                //return false;->
                ev.preventDefault();//不兼容ie6-8
            }
        }
    </script>
</html>
```

输入框焦点事件:onfocus

输入框在被激活时，会触发`onfocus`事件，我们可以利用这个方法做一些简单的业务处理：如使用正则来验证用户输入的文本是否正确等。

示例：使用onfocuse实现输入框的placeholder效果

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <input type="text" id="txt1" value="用户名" />
    </body>
    <script type="text/javascript">
        var txt1 = document.querySelector("#txt1");
        txt1.onfocus = function (){
            this.value = "";
        }
        txt1.onblur = function (){

//          alert('a');
            this.value="用户名";
        }
    </script>
</html>
```

事件冒泡

子标签发生事件后，向父级发送该事件，一直追溯到document。如：点击一个嵌套在body中的button，则该button的onclick事件也会传递给body、document中，触发他们的onclick里触发的函数。

示例：

```
    <!DOCTYPE html>
    <html lang="en">
    <head>
    <meta charset="UTF-8">
    <title></title>
    <style type="text/css">
        div {
            width: 100px;
            height: 100px;
            background-color: red;
            display: none;
```

```
        }
    </style>


    <script type="text/javascript">
        window.onload = function () {

            var oBtn = document.getElementById('button');
            var oDiv = document.getElementById('div');

            // 点击buton后，button的事件会被触发
            oBtn.onclick = function() {
                oDiv.style.display = 'block';
                alert("button被点击了");
            }

            // 由于事件冒泡，作为父级的document的onclick事件也会被触发
            document.onclick = function() {

                oDiv.style.display = 'none';
                alert("document被点击了")
            }
        }
    </script>
    </head>
    <body>
    <input type = "button" value = "显示" id = "button">
    <div id = "div"></div>

    </body>
    </html>
```

由于事件冒泡会触发绑定在父标签上的同类型事件，会给我们的开发带来很多麻烦，所以我们需要取消事件冒泡。取消事件冒泡的方法：把`cancelBubble`的值设置为`true`

示例：

```
    <!DOCTYPE html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
    <title></title>
    <style type="text/css">
        div {
            width: 100px;
            height: 100px;
            background-color: red;
```

```
            display: none;
        }
    </style>


    <script type="text/javascript">
        window.onload = function () {

            var oBtn = document.getElementById('button');
            var oDiv = document.getElementById('div');

            // 点击buton后，button的事件会被触发
            oBtn.onclick = function(ev) {
                oDiv.style.display = 'block';
                alert("button被点击了");
                // 在这里取消事件冒泡，防止事件向父级传递
                // 兼容性考虑，有的浏览器的事件并不是'event'，所以把点击事件作为
参数传递过来
                var oEvent = ev || event;
                oEvent.cancelBubble=true;

            }

            // 由于事件冒泡，作为父级的document的onclick事件也会被触发
            document.onclick = function() {

                oDiv.style.display = 'none';
                alert("document被点击了")
            }
        }
    </script>
    </head>
    <body>
    <input type = "button" value = "显示" id = "button">
    <div id = "div"></div>

    </body>
    </html>
```

事件绑定

给事件绑定一个方法，出了上边讲过的意外，还可以使用`addEventListener`来给标签对应的事件添加方法，使用之前的那种方式，只能给事件绑定一个方法，而使用`addEventListener`则可以给一个事件添加多个方法。语法格式如下：

```
// 非IE浏览器
```

```
target.addEventListener(type,listener,useCapture);
```

- target： 文档节点、document、window 或 XMLHttpRequest。
- type： 字符串，事件名称，不含“on”，比如“click”、“mouseover”、“keydown”等。
- listener： 实现了 EventListener 接口或者是 JavaScript 中的函数。
- useCapture： 是否使用捕捉，一般用 false 。例如：
  ```
  document.getElementById("testText").addEventListener("keydown", function (event) { alert(event.keyCode); }, false);
  ```

IE浏览器下的语法：

```
// IE浏览器
target.attachEvent(type, listener);
```

- target： 文档节点、document、window 或 XMLHttpRequest。
- type： 字符串，事件名称，含“on”，比如“onclick”、“onmouseover”、“onkeydown”等。
- listener： 实现了 EventListener 接口或者是 JavaScript 中的函数。

有了绑定事件，自然有移除事件，根据是否是IE浏览器有两种写法：

非IE浏览器：

```
target.removeEventListener(type,listener,useCapture);
```

- target： 文档节点、document、window 或 XMLHttpRequest。
- type： 字符串，事件名称，不含“on”，比如“click”、“mouseover”、“keydown”等。
- listener： 实现了 EventListener 接口或者是 JavaScript 中的函数。
- useCapture： 是否使用捕捉，一般用 false。

IE浏览器：

```
target.detachEvent(type, listener);
```

- target： 文档节点、document、window 或 XMLHttpRequest。
- type： 字符串，事件名称，含“on”，比如“onclick”、“onmouseover”、“onkeydown”等。
- listener： 实现了 EventListener 接口或者是 JavaScript 中的函数。

两者使用的原理：可对执行的优先级不一样，实例讲解如下：

```
ele.attachEvent("onclick",method1);
ele.attachEvent("onclick",method2);
ele.attachEvent("onclick",method3);
```

执行顺序为method3->method2->method1

```
ele.addEventListener("click",method1,false);
ele.addEventListener("click",method2,false);
ele.addEventListener("click",method3,false);
```

执行顺序为method1->method2->method3

兼容后的方法

```
var func = function(){};
//例：addEvent(window,"load",func)
function addEvent(elem, type, fn) {
if (elem.attachEvent) {
elem.attachEvent('on' + type, fn);
return;
}
if (elem.addEventListener) {
elem.addEventListener(type, fn, false);
}
}
//例：removeEvent(window,"load",func)
function removeEvent(elem, type, fn) {
if (elem.detachEvent) {
elem.detachEvent('on' + type, fn);
return;
}
if (elem.removeEventListener) {
elem.removeEventListener(type, fn, false);
}
}
```

DOM事件共有两个版本，其实IE和非IE浏览器中，种绑定事件方法中的“事件”就是这两个不同的的标准，下表列出了DOM1和DOM2这两种事件标准的关系

DOM 0 Event	DOM 2 Event
onblur()	blur
onfocus()	focus
onchange()	change
onmouseover()	mouseover
onmouseout()	mouseout
onmousemove()	mousemove
onmousedown()	mousedown
onmouseup()	mouseup
onclick()	click
ondblclick()	dblclick
onkeydown()	keydown
onkeyup()	keyup
onkeypress()	keypress
onsubmit()	submit
onload()	load
onunload()	unload

示例:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
    </body>
    <script type="text/javascript">
//      window.onload=function (){
```

```
//          alert('a');
//      }
//      window.onload=function (){
//          alert('b');
//      }
    //有兼容问题-》ie6-8不兼容
    window.addEventListener("load",function (){
        alert('a');
    },false);//最标准的绑定事件的方法
//  document.oncontextmenu
    window.addEventListener("load",function (){
        alert('b');
    },false)//第三个参数的含义就是冒泡和下沉
//  widnow.attachEvent()

    //attachEvent("onload",function (){})->
    </script>
</html>
```