## **一, input放在a标签里面单击不能获取input的光标（IE环境下）**

双击才可以获得焦点，目前有的解决方案：

* 不要给a标签添加href属性；
* 不要在外面套上a标签。

## **二 隐藏input标签的光标**

项目需求：input值json加载，只读+光标隐藏，通用的解决方案有其他标签模拟，但是不能改input 所以解决方案为给input加下面这两个属性：

//只读

readonly="readonly" //隐藏光标

unselectable="on"

## **三 返回私有数组**

返回数组的一个副本，这样改动就不会影响原数组，只是副本而已

var array = (function () {

var days = ['Mo', 'Tu', 'We', 'Th', 'Fr', 'Sa', 'Su'];

return {

getDays: function () {

return days.slice();

}

}

})()

## **四 jquery选择器的扩展**

//jQuery contains 选择器，对Contains查找的内容不区分大小写

jQuery.expr[':'].Contains = function (a, i, m) {

return jQuery(a).text().toUpperCase()

.indexOf(m[3].toUpperCase()) >= 0;};

例子

<div>john</div><div>John</div><div>hey hey JOHN hey hey</div>

$('div:Contains("john")') //会选择到两个div

## **五 当一个变量被声明后，扩充其属性并不会改变原数据类型**

var a = 'foo';

a[1] = 'O';

console.log(0.1+0.2==0.3||a); //'foo'

## **六 闭包是函数的嵌套定义，而不是函数的嵌套调用**

function foo(){

console.log(a);}function bar () {

var a = 3;

foo(); }var a = 2;bar(); //2

## **七 DOMContentLoaded兼容IE9以下版本**

//jQuery的实现

// Mozilla, Opera and webkit nightlies currently support this event

if ( document.addEventListener ) {

// Use the handy event callback

document.addEventListener( "DOMContentLoaded", DOMContentLoaded, false );

// A fallback to window.onload, that will always work

window.addEventListener( "load", jQuery.ready, false );

// If IE event model is used

} else if ( document.attachEvent ) {

// ensure firing before onload,

// maybe late but safe also for iframes

document.attachEvent( "onreadystatechange", DOMContentLoaded );

// A fallback to window.onload, that will always work

window.attachEvent( "onload", jQuery.ready );

// If IE and not a frame

// continually check to see if the document is ready

var toplevel = false;

try {

toplevel = window.frameElement == null;

} catch(e) {}

if ( document.documentElement.doScroll && toplevel ) {

doScrollCheck();

}

}

//函数DOMContentLoaded的赋值

if ( document.addEventListener ) {

DOMContentLoaded = function() {

document.removeEventListener( "DOMContentLoaded", DOMContentLoaded, false );

jQuery.ready();

};

} else if ( document.attachEvent ) {

DOMContentLoaded = function() {

// Make sure body exists, at least, in case IE gets a little overzealous (ticket #5443).

if ( document.readyState === "complete" ) {

document.detachEvent( "onreadystatechange", DOMContentLoaded );

jQuery.ready();

}

};

}

// The DOM ready check for Internet Explorer

function doScrollCheck() {

if ( jQuery.isReady ) {

return;

}

try {

// If IE is used, use the trick by Diego Perini

// http://javascript.nwbox.com/IEContentLoaded/

document.documentElement.doScroll("left");

} catch(e) {

setTimeout( doScrollCheck, 1 );

return;

}

// and execute any waiting functions

jQuery.ready();

}

//网友的实现

// @win window reference

// @fn function reference

function contentLoaded(win, fn) {

var done = false, top = true,

doc = win.document,

root = doc.documentElement,

modern = doc.addEventListener,

add = modern ? 'addEventListener' : 'attachEvent',

rem = modern ? 'removeEventListener' : 'detachEvent',

pre = modern ? '' : 'on',

init = function(e) {

if (e.type == 'readystatechange' && doc.readyState != 'complete') {

return;

}

//load事件在win上，移除事件监听(readystatechange, DOMContentLoaded, load)

(e.type == 'load' ? win : doc)[rem](pre + e.type, init, false);

//保证fn只执行一次

if (!done && (done = true)) fn.call(win, e || e.type);

},

poll = function() {

try { root.doScroll('left'); } catch(e) { setTimeout(poll, 50); return; }

init('poll');

};

if (doc.readyState == 'complete') fn.call(win, 'lazy');

else {

if (!modern && root.doScroll) {

try { top = !win.frameElement; } catch(e) { }

if (top) poll();

}

doc[add](pre + 'DOMContentLoaded', init, false); //触发时，doc.readyState为'interactive'

doc[add](pre + 'readystatechange', init, false); //会触发两次，readystatechange先触发，再触发DOMContentLoaded, 最后才是load

win[add](pre + 'load', init, false);

}

}

## **八 document.querySelectorAll和getElementsByTagName的区别**

//html代码<ul>

<li>a</li>

<li>b</li>

<li>c</li></ul>

//script代码

var ul = document.querySelector('ul');

var li = ul.querySelectorAll('li'),

tagLi = ul.getElementsByTagName('li'); //动态取值

ul.appendChild(document.createElement('li'));

console.log(li.length); //3 li.toString()为[object NodeList]

console.log(tagLi.length); //4 tagLi.toString()为[object HTMLCollection]

### **九 a,input,textarea,img标签点击后出现虚框，黄框等；**

a{outline:none;}

### 十 **去苹果浏览器默认样式**

input,textarea,button { -webkit-appearance: none; border-radius:0px; border:none;}

a有click等事件，href写法

href="javascript:void(0)"

//Bootstrap模态框，不然jQuery报错

href="#"

**十一 及时清除浮动**

//调Bootstrap直接加入样式：class="clearfix",没调请加入下面样式

.clearfix:before,.clearfix:after{content: " ";display: table;}

.clearfix:after { clear: both;}

**十二 css禁止选中**

-moz-user-select: none;

-webkit-user-select: none;

-ms-user-select: none;

-khtml-user-select: none;

user-select: none;

**十三 .盒子边框溢出**

在移动端的时候,当我们指定了一个块级元素时，并且为其定义了边框，设置了其宽度为100％。按照盒子模型，就会发现该元素的左右边框各1个像素会溢了，导致出现横向滚动条，这时候我们可以为其添加-webkit-box-sizing:border-box

用来指定该盒子的大小包括边框的宽度。

**十四、div插入图片有间隙**

   解决：（1）div{font-size:0}

    （2）img{display：block}

**十五、双倍浮向，在IE6及以下版本，浮动的元素，有设置margin，出现**

**双倍外边距。**

   解决：给浮动的元素设置display：inline

**十六、在IE6及以下版本，部分块级元素拥有默认高度**

   解决：设置font-size:0

**十七、表单元素行高不一样**

   解决：input{float：left}

**十八、按钮的大小不一样**

   解决：在input外边套一层标签，添加按钮的样式，把按钮的默认

样式去掉

**十九、百分比的bug，在IE6及以下版本中，解析的时候四舍五入计算，**

50%+50% > 100%,

   解决：把右边的浮动的元素，clear：right

**二十、透明属性**

兼容其他浏览器写法：opacity:value;(value的取值范围0-1;

    例：opacity:0.5;

)

IE浏览器写法：filter:alpha(opacity=value);取值范围 0-100(整数

)

**二十一、margin塌陷**

当前元素(父元素里面第一个子元素)与父元素没有设置任何浮动的情

况下，设置margin-top后，会错误的把margin-top加在父级元素上，

解决：

    （1）给父级元素添加overflow:hidden;(推荐使用)

    （2）给父元素设置border

    （3）给父元素或者子元素设置float

**二十二、margin合并BUG**

当两个上下排列的元素，上元素有margin-bottom：30px；下面元素有

margin-top:20px；他们中间的距离不会叠加，而是会设置为较大的值

解决：给两个元素或者其中一个加上display：inline-block