

本文存下来作为备忘。

1,更改css样式四种情况：

A, `$("#p").("background-color","red");`

B, `var nome = {  
background:'red',  
margin:'10px 0 0'  
};`

`$("#p").css(nome);`

C, `$("#p").css(red);`(此情况仅为具体的颜色)

D, `$("#p").css({"background":"red","color":"white"})`

2, css若写在head部分：

`<script type="text/javascript">`

`$(function{`

`...`

`})`

`</script>`

3,jquery替代\$符号

`Var jq = jQuery.noConflict();`

`jq(focument).ready(function(){`

`...`

`})`

4,fadeTo淡出：

`$("#p").fadeTo("slow",0.6);`

5,animate:

`$("#p").animate({height:"70%",weight:100,marginLeft:"0.6in"},"slow")`

6,输出属性：

`$("#button").click(function(){`

`$("#p").text($("#this").css("background-color"));`

`})`

7,页面加载完毕即运行：

`$(function(){`

`Function div(){`

`$("#p").animate({width:100},"slow");`

`$("#p").animate({width:300},"slow");`

`};`

`Div();`

`})`

8所有带\*属性的选择

`$("#[id]").css("red");`选区所有id属性的

`$("#[id=main]").css("red");`选取id为main的

9,选取文件夹为file属性

10 注意空格

`$("#div#demo")`选取id为demo的div

`$("#div#demo")`选区div下id为demo的

11 bind 绑定格式

`$("#p").bind("click",function(){`

`});`

`$("#p").bind(`

`{clcik:function(){...}},`

`{clcik:function(){...}},`

`)`

12,设置并触发

`$("#input").focus(function(){`

`$("#input").css("background-color"," #fff");`

`$("#btn").click(function(){`

`$("#input").focus();`

`})`

`})`

13,change事件仅适用于text textarea select中

```
14, $("button").click(function(){
```

```
    $("p").slideToggle();
```

```
});
```

```
    $("p").dblclick(function(){
```

```
        $("button").click();
```

```
    })
```

15,delegate

```
    $("div").delegate("button","click",function(){
```

```
        $("p").slideToggle();
```

```
    });
```

16,图片未加载-error

```
    $("img").error(function(){
```

```
        $("img").replaceWith("<p>图片未加载</p>")
```

```
    })
```

17, 阻止链接打开——preventDefault

```
    $("a").click(function(event){
```

```
        Event.preventDefault();
```

```
        Alert("Defaultprevent:" + event.isDefaultPrevented ());
```

```
    })
```

效果是 a无法打开, 并且弹出消息框 Default prevent : true;

18,输出鼠标指针位置

```
    $(document).mouseover(function(e){
```

```
        $("span").text("X:"+e.pageX+",Y:"+e.pageY);
```

```
    })
```

19,event.target.nodeName为获得当前触发元素

20, 输出为自1970年1月1日至今的毫秒数

```
    $("btn").click(function(e){
```

```
        $("p").text(e.timeStamp);
```

```
    })
```

21,event.type表示事件类型

```
    $("p").bind('clickdblclick mouseover',function(event){
```

```
        $("div").text("事件:"+event.type);
```

```
    }) 当鼠标经过时 会出现结果事件 : mouseover
```

22, 键盘值

```
    $("input").keydown(function(event){
```

```
        $("div").text("Key:"+event.which)
```

```
    })
```

23,区分mouseover与mouseenter事件对应于

无论鼠标指针是经过被选元素还是其子元素都会触发mouseover事件 mouseout同

只有鼠标指针经过被选元素时才会触发mouseenter mouseleave同 ;

```
Varx = 0;
```

```
    $("p").mouseover(function(){
```

```
        $("div").text(x+=1);
```

```
    });结果为不断累加
```

```
    $("p").click(function(){
```

```
        $(this).fontSize+="6px");字体不断增大6px ;
```

```
    })
```

24, ready()函数不与body onload=""一起使用 ;

25, resize 调整窗口大触发 ;

26, 阻止表单提交

```
    $("button").submit(function(e){
```

```
        e.preventDefault();
```

```
    })
```

```
27,$(button).toggle(
```

```
    Function(){$("p").css("red")},
```

```
    Function(){$("p").css("red")},
```

```
    Function(){$("p").css("red")}
```

```
);
```

27,\$("p").toggle(true),元素一直显示 ; 若为false则为隐藏

28, trigger("select")触发select事件

它能匹配所有元素 而triggerHandler只会影响第一个匹配元素 ;

29,循环两次结束：

```
$(function(){
  Varx = 0;
  $("p").click(function(e){
    $("p").animate({fontSize:"+=6px"});
    x++;
    If(x>2){
      $(this).unbind(e);
    }
  });
})
```

30,取消绑定特定函数

```
Function alertMe(){
  Alert("123")
};
$(function(){
  $("p").click(alertMe);
  $("button").click(function(){
    $("p").unbind("click",alertMe);
  });
});
```

31, addClass 返回函数

```
$("p").addClass(function(n){
  Return'par' + n;
});
```

添加两个类

```
$("p").addClass("class1class2")
```

32,宽度减去50

```
$("#img").attr("width",function(n,v){
  Returnv - 50;
})
```

v为属性值()width的, n为index；

33, clone

```
$("p").append($("p").clone());
$("p").append($("p").clone(true));表示事件处理器也被同样复制；
```

34, \$("p").deatch();删除p元素；(并不是彻底删除)

```
Varx;
$("#btn1").click(function(){
  X=$("#p").deatch();
});删除p
$("#btn2").click(function(){
  $(body).append(x);
});添加p；
```

Empty()是指移除内容；

35, hasClass()指被选元素是否包括\*元素

```
$("#button").click(function(){
  Alert($("#p").hasClass("into"));
});输出结果为true或者false;
```

36,html()返回结果为()内容；

37, document.createElement("div")表示新创建div

38, toggleClass()对类 进行切换

39,css增加div宽度

```
$("#div").css("width",function(index,value){returnparseFloat(value)*1.2;})
```

40,stop()只适用于动画, 而clearQueue适用于任何排队函数。

41, \$("#stop").click(function(){

```
$("#box").stop(true);
```

});动画依然能完成；

```
$("#stop").click(function(){
```

```
$("#box").stop(true, true);
```

});动画不能完成；

42, toggle(),进行显示和消失的切换, toggle(true), 只能显示, 不能消失；

```
43,$("p").height(50,或者height("20%");
44,$("p").size();元素个数=".length";
45,每次增加10px宽度;
$("p").height(function(n,c){
Returnc+10;
});
46, offset表示文档相对位置;
X=$("p").offset();position也可以这样用;
x.left表示距离左边的距离;
$("p").offset({top:100,left:0});重新设置距离
```

```
$("p").offset(function(n,c){
newPos=newObject();
newPos.left= c.left+100;
returnnewPos;
});
```

```
NewPos= new Object();
newPos.left=100;
$(function(){
$("button").click(function(){
$("p").offset(newPos);
});
});
```

```
$("p").offset($("img").offset())
47, $("P").scrollLeft(100),把滚动条的位置设置为100px;
48, closest() 方法获得匹配选择器的第一个祖先元素,从当前元素开始沿 DOM 树向上。
Addself包括当前元素;
end() 方法结束当前链条中的最近的筛选操作,并将匹配元素集还原为之前的状态。
filter() 方法将匹配元素集合缩减为匹配指定选择器的元素。
49var item1 =$('li.item-1')[0];{_?
50,$("p").has("li").length? "Yes":"No" 判断P元素是否含有li元素,有则返回Yes;
51,.is("p")判断是否含有P;
52, $("ul").click(function(event){
```

```
Var$target=$(event.target);
If($target.is("li")){
$("target").css("background","red");
}
});点击一个列表,若含有li则当前点击列表变为红色;
```

```
53, $("li").click(function(){
Var$li=$(this),
isWithTwo=$li.is(function(){
return$("strong",this).length===2;
});
If(isWithTwo){
$li.css("background","green")
}else{
$li.css("background","red")
};
});若当前li元素内有两个"strong"元素时,点击则为绿色;
```

```
54, map() 把每个元素通过函数传递到当前匹配集合中,生成包含返回值的新的 jQuery 对象
$("p").append(
$(":checkbox").map(function(){
Returnthis.id;
});
);输出选框的id;
```

```
55, $("ul").nextUntil("li") ul下的所有元素知道li;
56, offsetParent()获得被定位的最近祖先元素。
```

57, \$("p").parentsUntil("span",".yes")选择p的类名为".yes"父级元素知道"span"

58.\$("p").slice(0,2)选取的为第2个；

59, \$(selector).data(name,value)向name添加数据；

```
60;function runIt(){
  $("div").show(1000);
  $("div").hide(1000);
};
```

runIt());循环播放；

61, dequeue()方法为匹配元素执行序列中的下一个函数。

ShowIt()表示一直显示， setTimeout(showIt,100)表示没100毫秒更新次；

62, get()方法获得由选择器指定的 DOM 元素。

```
$("#button").click(function(){
```

```
X = $("p").get(0);
```

```
$("#div").text(x.nodeName+": "+x.innerHTML)
```

```
});    表示P      p的内容；
```

63, \$("span",this)等于\$(this).find("span")

文章标签：[jquery](#) [javascript](#) [css](#) [html](#) [动画](#)

个人分类：[HTML和JavaScript](#)

此PDF由spygg生成,请尊重原作者版权!!!

我的邮箱:liushidc@163.com