2020.12~2020.12

**前端 - 7**

**JavaScript的jQuery库**

**jQuery对象 和 DOM对象**

**jQuery选择器**

**jQuery的常用API（ 样式、属性、效果和元素操作）**

**jQuery事件**

**jQuery插件**

**jQuery尺寸位置操作**

**jQuery基础**

**jQuery入门**

**目录**

[一．jQuery概述 10](#_Toc14879)

[1.jQuery下载与引入 10](#_Toc19687)

[2.jQuery的使用 11](#_Toc17353)

[顶级对象 ＄ 11](#_Toc18081)

[多库共存 11](#_Toc9356)

[入口函数 12](#_Toc28847)

[3.jQuery对象 13](#_Toc7088)

[jQuery对象转换为DOM对象 13](#_Toc28640)

[3.DOM对象 14](#_Toc10602)

[DOM对象转换为jQuery对象 14](#_Toc31084)

[二．jQuery获取元素 15](#_Toc30392)

[2. jQuery筛选方法 15](#_Toc19786)

[ parents（“选择器”） 父级筛选器 16](#_Toc23262)

[ find（“选择器”） 子级筛选器 16](#_Toc18485)

[案例 – 用jQuery做下拉菜单 17](#_Toc8232)

[案例 – 用jQuery做简易tab栏切换 18](#_Toc20953)

[1. jQuery选择器 19](#_Toc986)

[3. jQuery筛选选择器 19](#_Toc32139)

[ jQuery隐式迭代 20](#_Toc29327)

[ jQuery排他思想 21](#_Toc31147)

[ jQuery链式编程 22](#_Toc19602)

[三．jQuery元素样式 23](#_Toc2843)

[1.单一样式 23](#_Toc6646)

[1.1只写属性名，不写值 23](#_Toc32116)

[1.2写属性，也写值 23](#_Toc5240)

[2.多样式 24](#_Toc30178)

[3.类名 常用 24](#_Toc23923)

[ addClass( ) 添加类 25](#_Toc17861)

[ removeClass( ) 删除类 25](#_Toc10347)

[ toggleClass( ) 切换类 26](#_Toc916)

[案例 – 用增删类名改善简易tab栏切换 27](#_Toc9979)

[四．jQuery元素效果 28](#_Toc9103)

[1.隐藏删除和切换 28](#_Toc10539)

[ hide( ) 隐藏元素 28](#_Toc22335)

[ show( ) 显示元素 28](#_Toc15736)

[ toggle( ) 切换元素的显示隐藏 29](#_Toc29155)

[案例 - 显示隐藏下拉菜单 29](#_Toc13841)

[2.下拉上拉和切换 30](#_Toc16683)

[ slideDown( ) 下拉显示元素 30](#_Toc25883)

[ slideUp( ) 上拉隐藏元素 30](#_Toc1275)

[ slideToggle( ) 切换元素的下拉和上拉 31](#_Toc12285)

[案例 - 下拉上拉菜单 31](#_Toc17926)

[3.淡入淡出和切换 32](#_Toc23028)

[ fadeIn( ) 淡入显示元素 32](#_Toc2686)

[ fadeOut( ) 淡出隐藏元素 32](#_Toc10383)

[ fadeToggle( ) 切换元素的淡入和淡出 33](#_Toc8509)

[案例 - 盒子缓慢消失和显示 33](#_Toc22720)

[ fadeTo( ) 渐进调整元素透明度 34](#_Toc16128)

[案例 - 切换高亮显示 34](#_Toc20890)

[ stop( ) 停止动画队列 35](#_Toc18093)

[案例 - 用stop停止下拉菜单的动画排队 35](#_Toc10754)

[4.自定义动画 animate 36](#_Toc7368)

[案例 - 折叠卡片/手风琴效果 37](#_Toc213)

[五．jQuery元素属性 38](#_Toc10107)

[1. 获得/修改元素 固有属性 38](#_Toc25429)

[ prop( ) 获得固有属性的值 38](#_Toc26583)

[ prop( ) 修改固有属性的值 38](#_Toc7124)

[案例 - 复选框和单选框的全选/全不选 39](#_Toc15936)

[ :checked 筛选选择器 39](#_Toc21248)

[2. 获得/修改元素 自定义属性 40](#_Toc15142)

[ attr( ) 获得自定义属性 40](#_Toc55)

[ attr( ) 修改自定义属性 40](#_Toc9834)

[3. 获得/添加元素 数据缓存属性 41](#_Toc18891)

[ data( ) 获得H5新增data-开头属性 41](#_Toc31534)

[ data( ) 数据缓存 添加属性 41](#_Toc15394)

[六．jQuery元素内容 42](#_Toc17119)

[1.获得/修改元素 HTML结构+内容 42](#_Toc14097)

[ html( ) 获得结构+文本内容 42](#_Toc20909)

[ html( ) 添加/修改结构+文本内容 42](#_Toc10989)

[2.获得/修改元素 文本内容 43](#_Toc14307)

[ text( ) 仅获得文本内容 43](#_Toc19223)

[ text( ) 添加/修改文本内容 43](#_Toc5800)

[3.获得/修改input表单 内容 44](#_Toc18972)

[ val( ) 获得表单内容 44](#_Toc15419)

[ val( ) 添加/修改表单内容 44](#_Toc5020)

[案例 - 增添商品的数量 45](#_Toc19116)

[案例 - 根据修改的商品数量修改总价 46](#_Toc23185)

[ toFixed( number ) 保留小数点后位数 46](#_Toc1426)

[七．jQuery操作元素 47](#_Toc18097)

[1.遍历元素 47](#_Toc1626)

[ each( ) 遍历DOM元素 47](#_Toc8704)

[案例 - 商品总数量和总价 48](#_Toc21533)

[ $.each( ) 遍历数据 50](#_Toc12235)

[2.创建元素 51](#_Toc803)

[ $( ”标签+文本内容“) 51](#_Toc24740)

[3.内部添加元素 做子元素 52](#_Toc24955)

[ append( ) 添加到元素内部最后 52](#_Toc29463)

[ prepend( ) 添加到元素内部最前 52](#_Toc8527)

[4.外部添加元素 做兄弟元素 53](#_Toc20272)

[ after( ) 添加到元素外部最后 53](#_Toc14047)

[ before( ) 添加到元素外部最前 53](#_Toc6925)

[5.删除元素 54](#_Toc3908)

[ remove( ) 删除元素自身 54](#_Toc14601)

[ empty( ) 删除元素的内容 54](#_Toc11888)

[ html(“”) 删除元素的内容 54](#_Toc9288)

[案例 - 删除商品 55](#_Toc30213)

[6.拷贝/合并 对象 56](#_Toc1714)

[ $.extend( ) 拷贝对象 56](#_Toc9200)

[1.拷贝给的对象为空 56](#_Toc31637)

[2.拷贝给的对象原有内容 56](#_Toc20217)

[3.浅拷贝 57](#_Toc19591)

[4.深拷贝 57](#_Toc6300)

[八．jQuery事件 58](#_Toc8778)

[1.事件注册/解绑 58](#_Toc18158)

[元素.事件(执行程序) 单个事件注册 58](#_Toc12719)

[ on( ) 注册事件 推荐 59](#_Toc20052)

[1.on( ) 注册单个事件 59](#_Toc27865)

[2.on( ) 注册多个事件 59](#_Toc21085)

[3.on( ) 实现事件委派(事件委托) 60](#_Toc17228)

[3.on( ) 给动态生成的元素绑定事件 60](#_Toc7676)

[案例 - 简易留言板 61](#_Toc4004)

[ off( ) 解绑事件 62](#_Toc22255)

[ one( ) 仅执行一次事件 63](#_Toc2036)

[2.自动触发事件 63](#_Toc3362)

[元素.事件名( ) 63](#_Toc28179)

[ trigger( ) 63](#_Toc3649)

[ triggerHandler( ) 63](#_Toc26469)

[3.事件对象 64](#_Toc26337)

[阻止默认行为 64](#_Toc24548)

[阻止事件冒泡 64](#_Toc1508)

[案例 - 判断键盘按下 64](#_Toc14964)

[3.常用事件 65](#_Toc23557)

[ hover( ) 鼠标经过和离开 65](#_Toc24771)

[ change( ) input表单变化事件 65](#_Toc24745)

[九．jQuery尺寸、位置操作 66](#_Toc5959)

[1.元素尺寸 高度宽度 66](#_Toc28271)

[2.元素位置 67](#_Toc24091)

[ offset( ) 元素距离文档的位偏移 可读写 67](#_Toc15643)

[ position( ) 元素距离定位父元素的位偏移 只读 68](#_Toc11716)

[ scrollTop( )/scrollLeft 页面滚动后元素卷去的部分 69](#_Toc11196)

[案例 - 滚动页面显示导航条 70](#_Toc5931)

[案例 - 页面直接跳转到指定位置 71](#_Toc25932)

[案例 - 页面动画滚动到指定位置 71](#_Toc18505)

[用$(“body，html”) 代替 document 71](#_Toc31521)

**一．jQuery概述**

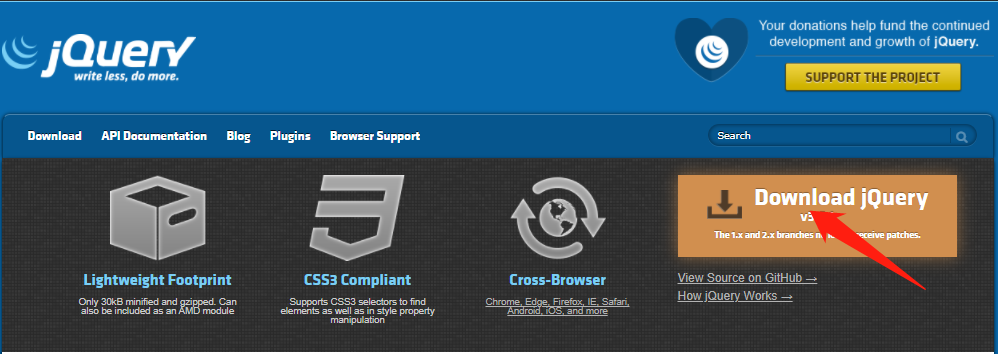
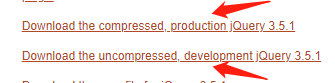
jQuery就是一个JS库( library)**，里面封装的是JS里的常用函数（方法）**。

jQuery的使用就是**调用函数**。通过调用这些封装好的函数实现了快速方便开发。

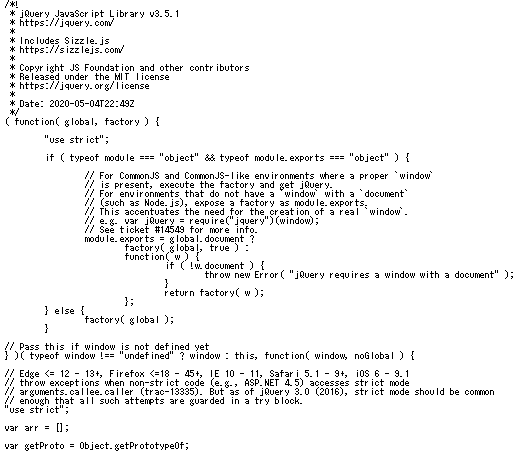
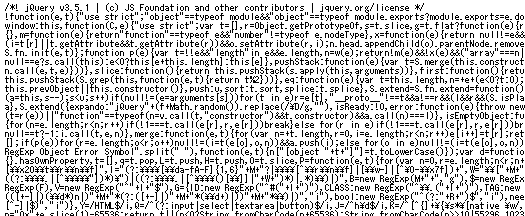
**1.jQuery下载与引入**

**下载**

jQuery官网下载 <https://jquery.com/>



分为development版本 **production版本（压缩版） 常用**

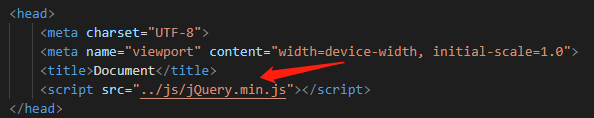


复制全选production版本的所有代码，存入一个JS 文件。命名**jQuery.min.js**

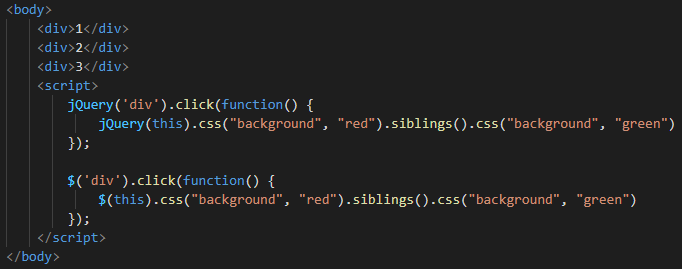
1

**引入**

jQuery被放入了一个JS文件，需要引入HTML页面文档。

在<head>内用**<script>引入jQuery.min.js文件**。

**2.jQuery的使用**

****

原生JS需要function 函数名（）{ } 声明函数，函数调用时： 函数名（）。

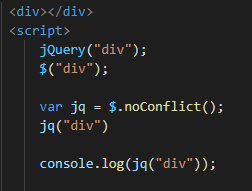
使用jQuery时候，要**写**jQuery( ) 或 $（）调用jQuery的函数。

**顶级对象 ＄**

**＄**是**jQuery**的别称，一般为了方便，直接使用**＄。**

**＄**是jQuery的顶级对象，相当于原生JS中的顶级对象window。

**多库共存**

解决jQuery的 $ 和其他JS库的 $ 冲突，

**方法一：**在jQuery中不使用 $,而是用jQuery

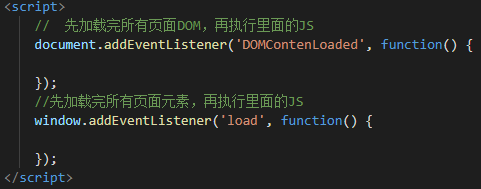
**方法二：**释放权限，用noConflict( )方法 让用户自定义

**var 自定义名 = $.noConflict( )**

**入口函数**

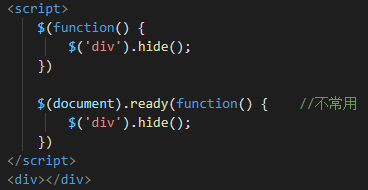
和原生JS一样，若代码不在目标元素的后面，JS从上到下执行会找不到该元素，无法执行。

原生JS用到了**load / DOMContentLoaded**窗口加载事件：



jQuery需要一个入口函数，类似原生的DOMContentLoaded窗口加载事件。

jQuery的入口函数是等仅所有DOM结果渲染完毕即可执内部代码。



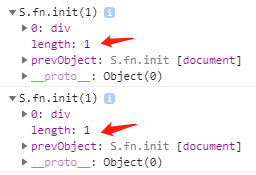
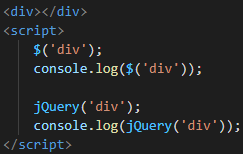
**3.jQuery对象**

用jQuery获取的元素对象，就是jQuery对象。

jQuery对象只能调用jQuery的方法函数，不能用DOM的属性和方法。

jQuery( ) 或 $( )

**用jQuery获得的元素对象是伪数组形式**。类似querySelectorAll



**jQuery对象转换为DOM对象**

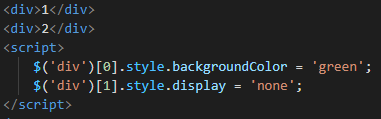
jQuery里封装的是常用JS函数但不是全部，

所以要使用没封装的方法和属性时，需要转换为DOM对象。

因为jQuery获取的元素是伪数组形式，用序号指定元素。

**$( 元素 )[ 序号 ]**

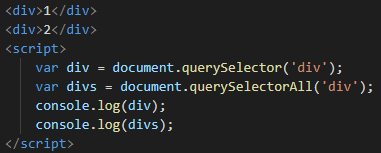
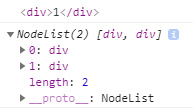




**3.DOM对象**

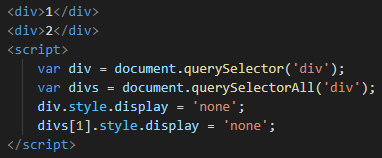
用原生JS获取的元素对象，就是DOM对象。

DOM对象只能用原生JS自带的DOM属性和方法，不能用jQuery的方法。



querySelector获得的是第一个该元素，

querySelectorAll是以**伪数组形式**获取所有元素

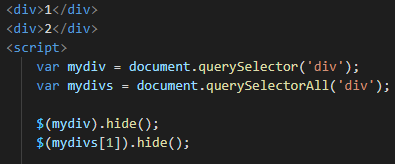


**DOM对象转换为jQuery对象**

jQuery的调用的方法写起来简单

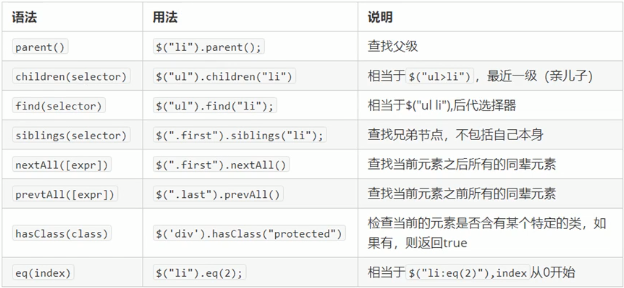
若DOM对象想使用jQuery的方法，需要转换位jQuery对象。

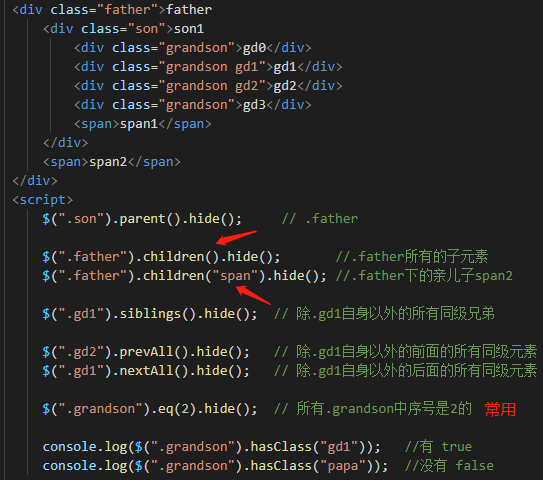
**$( 通过DOM获得的DOM对象 )**



**二．jQuery获取元素**

1. **jQuery筛选方法**

类似DOM的节点操作。可以获得父子兄元素



* **parents（“选择器”） 父级筛选器**

**元素 . parent（ ）只能获得最近一层的父元素（亲爸爸）**

parent( ) 要一层一层向上找父元素。会很麻烦。

**元素 . parents（ ）**返回当前元素的**全部祖先元素**。

**元素 . parents（“指定选择器” ）**返回当前元素的**全部祖先元素中的指定祖先元素**。

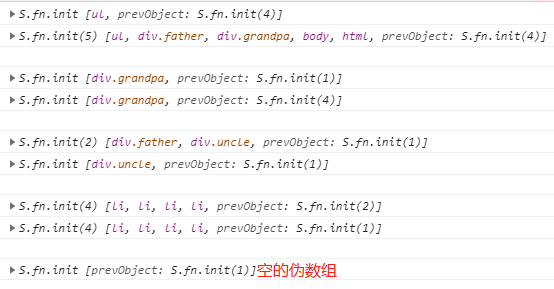
* **find（“选择器”） 子级筛选器**

**元素 .children( ) 只能获得最近一层的所有子元素（所有亲儿子）**

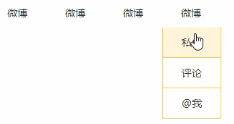
**元素 .children( “指定选择器”) 只能获得最近一层子元素中的指定元素。**

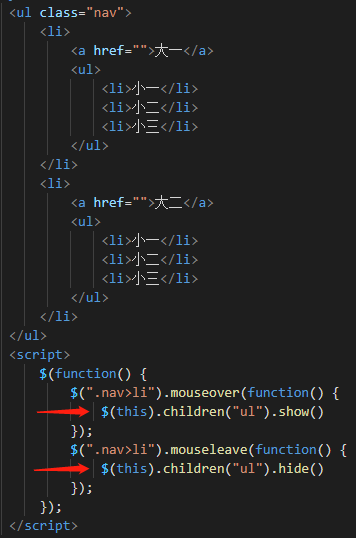
children( ) 要一层一层向上找父元素。会很麻烦。

**元素 .find（“指定选择器” ）**返回当前元素的**全部子元素中的指定子元素**。



**案例 –** 用jQuery做下拉菜单

****



利用jQuery的隐式迭代，给每个<li>绑定鼠标经过和鼠标离开事件。

利用 **$(this)** 指向当前触发事件的<li>的子元素里的<ul>

**案例 –** 用jQuery做简易tab栏切换





1.利用隐式迭代，给每个<li>绑定点击事件。

2.获得当前被点击的元素的序号：

**$( this ).index( ); 获得当前元素的序号**

3.在点击事件内，获得下面内容盒子的匹配序号的元素显示。

4.利用**链式编程**，把当前元素的其余兄弟都隐藏。

**1. jQuery选择器**



类似CSS与原生JS。可以用 **$( ) 里直接写选择器** 来获得元素。





1. **jQuery筛选选择器**

类似H5的结构伪类选择器。可以获得指定元素。



* **jQuery隐式迭代**

jQuery自动给**同一元素做相同操作**。

自动遍历了所有匹配的元素

自动给所有匹配的元素调用了方法

自动给所有匹配的元素添加了事件

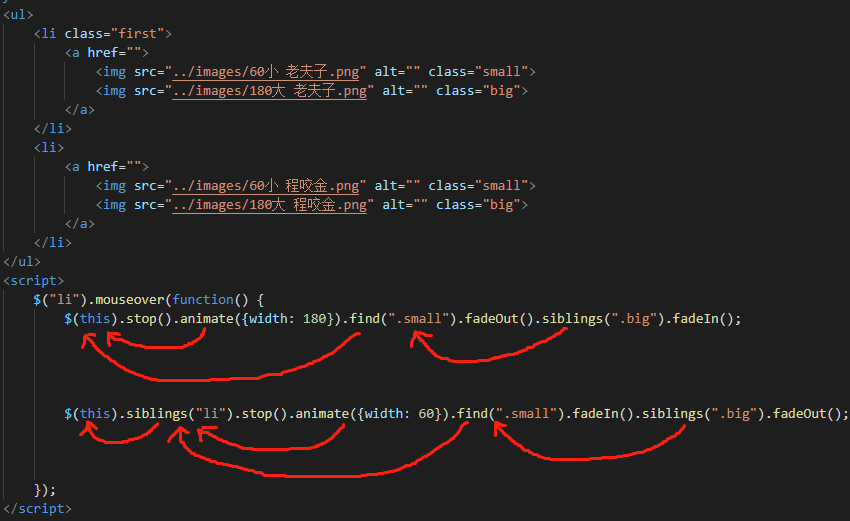
相当于省略了写for循环的一步

* **jQuery排他思想**



* **jQuery链式编程**

用一条链子的形式，以当前元素位目标向后写。



如上：

this 自身做动画、this的子元素 .samll 淡出、 **.samll 的兄弟 .big淡入**

this的兄弟li做动画、li的子元素 .samll淡入、 **.samll 的兄弟 .big淡出**

**三．jQuery元素样式**

**1.单一样式**

**1.1只写属性名，不写值**

1

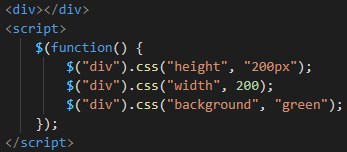
只获得属性的值，不会修改元素的样式

1

**1.2写属性，也写值**

1

修改元素该属性的值。

属性要加 ”“，**值也要加 ”“**。**值是数字时省略 ”“ 和单位px**。

**2.多样式**

**对象形式**



要修改多个属性时，可以采用对象的形式。

对象的属性是css样式属性，不加引号。和原生JS一样，符合类名采用**驼峰命名法**

对象的属性值是css样式属性值，要加引号，数字可以省略引号和px。



**3.类名 常用**

要修改多个样式时，用对象形式写起来还是麻烦。

可以在CSS里预先写一个类，里面写好样式，然后jQuery**添加**/调用该样式。

* **addClass( ) 添加类**

1



与原生JS的className不同，**className**是以行内样式**覆盖元素原来的类**。

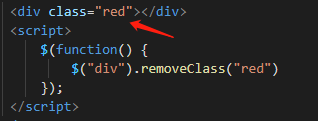
jQuery的**addClass不会覆盖**元素原来的类，只**是追加新的类（多类名）**

**1**

* **removeClass( ) 删除类**

1

删除元素原来自带的类名。



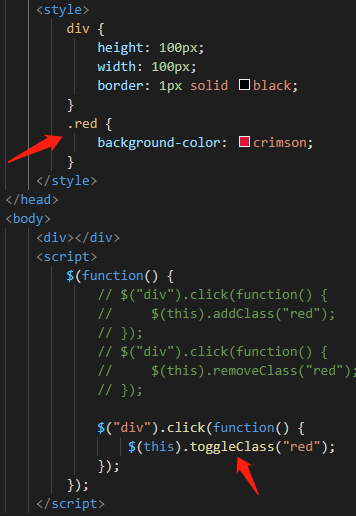
* **toggleClass( ) 切换类**

1

**若现在元素没有该属性，就添加该属性。**

**若现在元素有该属性的话，就删除该属性。**

比如，可以实现一个元素多次点击时，属性来回切换。



**案例 –** 用增删类名改善简易tab栏切换

给$(this)添加类，给其所有兄弟元素移除类。

**链式编程**写在一起。



**四．jQuery元素效果**

**1.隐藏删除和切换**

* **hide( ) 隐藏元素**

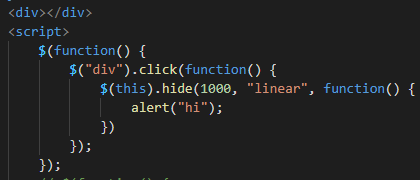
1类似原生JS里的style.display=‘none’

参数可以都省略不写。**一般不写参数使用无动画的hide( )**。

Speed：隐藏元素花费的时间。一般不写。

Easing：隐藏元素的运动曲线。一般不写。

Fn：回调函数，元素隐藏后执行。一般不写。

实际上不用参数

* **show( ) 显示元素**

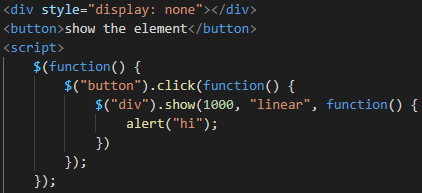
1类似原生JS里的style.display=‘block’

参数可以都省略不写。**一般不写参数使用无动画的show( )**。

Speed：显示元素花费的时间。一般不写。

Easing：显示元素的运动曲线。一般不写。

Fn：回调函数，元素显示后执行。一般不写。



实际上不用参数

* **toggle( ) 切换元素的显示隐藏**

**若元素隐藏，则让其显示**show( )

1**若元素显示，则让其隐藏**hide( )

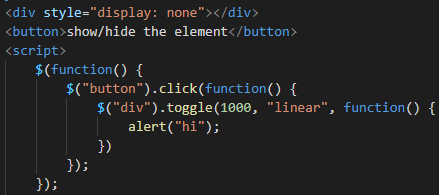
参数可以都省略不写。**一般不写参数使用无动画的toggle( )**。

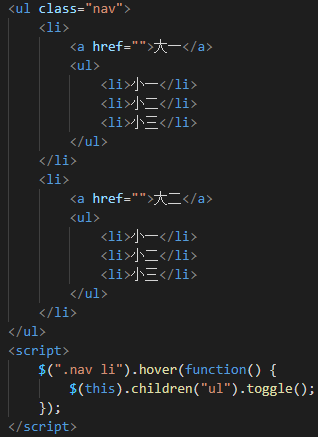
Speed：显示/隐藏元素花费的时间。一般不写。

Easing：显示/隐藏元素的运动曲线。一般不写。

Fn：回调函数，元素显示/隐藏后执行。一般不写。

如下：每次点击按钮，元素切换的显示和隐藏

实际不用参数

**案例** - 显示隐藏下拉菜单

**2.下拉上拉和切换**

* **slideDown( ) 下拉显示元素**

1

参数可以不写，也可以写参数实现动画效果。

Speed：下拉显示元素花费的时间。可以不写。

Easing：下拉显示元素的运动曲线。可以不写。

Fn：回调函数，元素下拉显示后执行。可以不写。

* **slideUp( ) 上拉隐藏元素**

1

参数可以不写，也可以写参数实现动画效果。

Speed：上拉隐藏元素花费的时间。可以不写。

Easing：上拉隐藏元素的运动曲线。可以不写。

Fn：回调函数，元素上拉隐藏后执行。可以不写。



* **slideToggle( ) 切换元素的下拉和上拉**

**若元素隐藏，则让其下拉显示**slideDown( )

**若元素显示，则让其上拉隐藏**slideUp( )

1

参数可以不写，也可以写参数实现动画效果。

Speed：下拉显示/上拉隐藏元素花费的时间。可以不写。

Easing：下拉显示/上拉隐藏元素的运动曲线。可以不写。

Fn：回调函数，元素下拉显示/上拉隐藏后执行。可以不写。

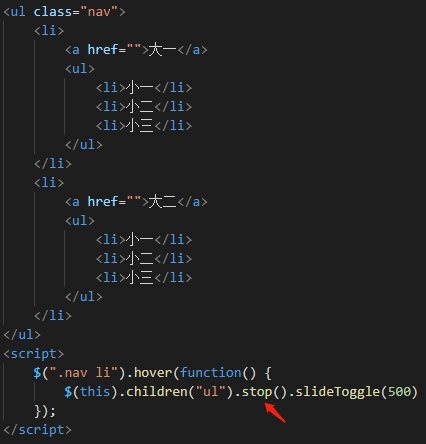
**案例** - 下拉上拉菜单

使用鼠标经过+离开的hover( )事件

在使用切换下拉上拉效果的**slideToggle( )**

因为是切换的动画效果，会产生**动画队列**。

需要用到stop( )消除当前动画效果的前一个动画效果。



**3.淡入淡出和切换**

* **fadeIn( ) 淡入显示元素**

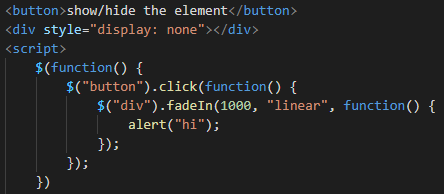
1比show( )的显示效果多了慢慢淡入的动画。

参数可以不写，也可以写参数实现动画效果。

Speed：淡入显示元素花费的时间。可以不写。

Easing：淡入显示元素的运动曲线。可以不写。

Fn：回调函数，元素下淡入显示后执行。可以不写。



* **fadeOut( ) 淡出隐藏元素**

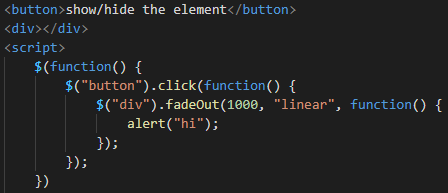
1比hide( )的隐藏效果多了慢慢淡出的动画。

参数可以不写，也可以写参数实现动画效果。

Speed：淡出隐藏元素花费的时间。可以不写。

Easing：淡出隐藏元素的运动曲线。可以不写。

Fn：回调函数，元素淡出隐藏后执行。可以不写。



* **fadeToggle( ) 切换元素的淡入和淡出**

**若元素隐藏，则让其淡入显示**fadeIn( )

**若元素显示，则让其淡出隐藏**fadeOut( )

1

参数可以不写，也可以写参数实现动画效果。

Speed：淡入显示/淡出隐藏元素花费的时间。可以不写。

Easing：淡入显示/淡出隐藏元素的运动曲线。可以不写。

Fn：回调函数，元素淡入显示/淡出隐藏后执行。可以不写。

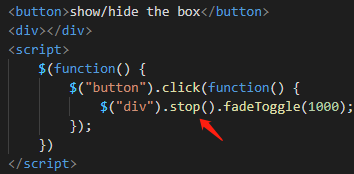
**案例** - 盒子缓慢消失和显示

使用click点击事件。每次点击<button>，盒子box就切换的显示和隐藏。

利用**fadeToggle( )**切换淡入淡出效果。

因为是切换的动画效果，会产生**动画队列。**

需要用到stop( )消除当前动画效果的前一个动画效果。



* **fadeTo( ) 渐进调整元素透明度**

可理解为CSS属性opacity的动画版。

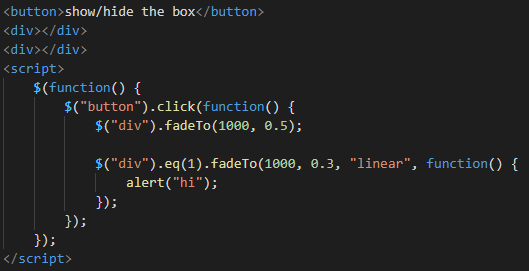
1

**Speed**：调整元素透明度花费的时间。**必须写**。

**Opacity：**和CSS一样。调整元素透明度，值为1~0。**必须写**

Easing：调整元素透明度的运动曲线。可以不写。

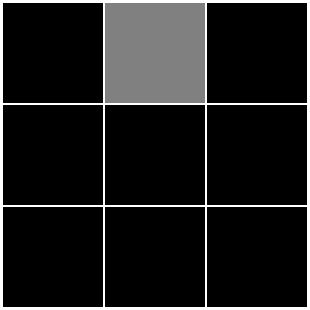
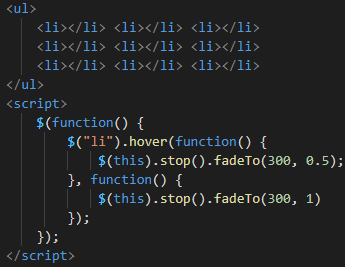
Fn：回调函数，调整元素透明度后执行。可以不写。



**案例** - 切换高亮显示

用hover(函数，函数) 实现鼠标经过的效果切换。

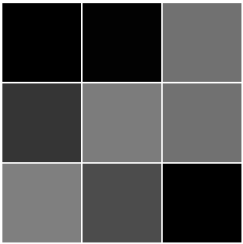
要用stop( ) 消除动画队列



* **stop( ) 停止动画队列**

动画一旦执行就一定要执行完。

若是短时间内多次触发，就会造成多个**动画**/**效果**的**排队执行**。影响美观。

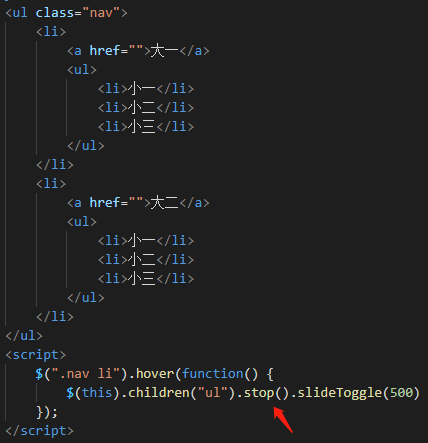
尤其是鼠标经过事件hover最明显。如下



****可以使用stop来停止一个动画的上一个动画的执行。停止动画的排队执行。

是停止该动画的上一个动画，要**写在该动画/效果的链式编程的前面**。

**案例** - 用stop停止下拉菜单的动画排队



**4.自定义动画 animate**

1

**Params**：要修改的属性样式，以**对象方式**传递，**必须写**。

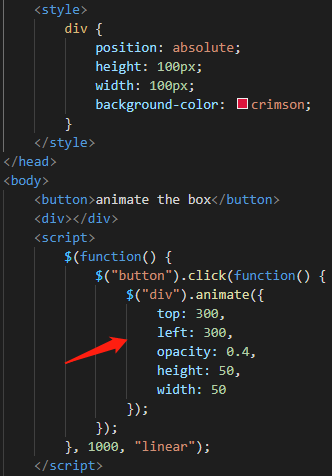
Speed：动画花费的时间。可以不写。

Easing：动画的运动曲线。可以不写。

Fn：回调函数，动画结束后执行。可以不写。

css( ) 和addClass( ) 是一下子瞬间修改样式

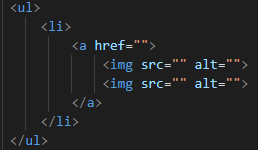
动画animate( ) 在修改的过程显示时间和运动曲线。并且动画的属性是同时改变。如下：

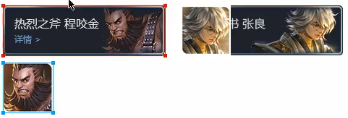


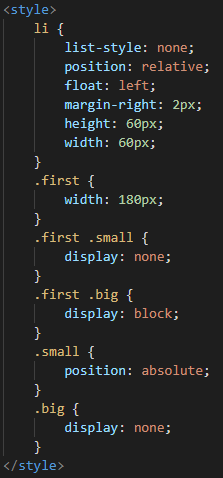
**案例** - 折叠卡片/手风琴效果



布局是一个盒子里用定位叠放两个图。

然后鼠标经过，盒子和两个图片分别做动画和显示隐藏





**五．jQuery元素属性**

**1. 获得/修改元素 固有属性**

* **prop( ) 获得固有属性的值**

1类似原生JS的 **element. 属性**

只写属性不写值，是获得该属性的值。属性名加引号。



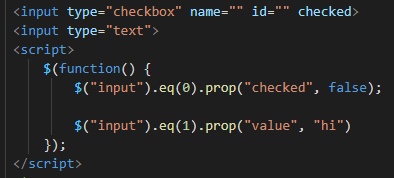
1



* **prop( ) 修改固有属性的值**

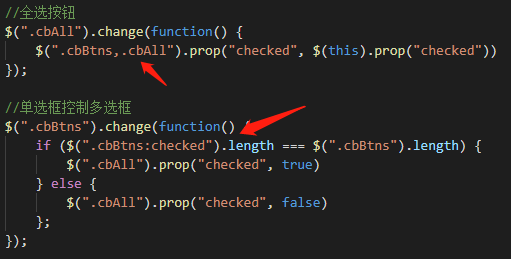
1类似原生JS的 **element. 属性=‘值’**

写了属性又写了属性值，是把该属性修改位该值。

属性名加引号，属性名和属性值用逗号隔开。

1

**案例** - 复选框和单选框的全选/全不选



1. **复选框控制单选框**

利用单击复选框后，所有单选框的checked属性跟随复选框的checked属性。

上下两个全选框，类名相同。

点击任何一个都会修改所有单选框的checked属性并一同修改另一个复选框。

用并集选择器。

1. **单选框控制复选框**

判断获得所有**被选中的单选框的个数** 是否等于 **所有单选框的个数**。

$( ”元素“ )是以**数组形式**获得所有该元素，该**元素的个数**就是**数组的长度length**

* **:checked 筛选选择器**

**$( ”input元素：checked“ )** 可以筛选出并获得**被选中的表单元素。**

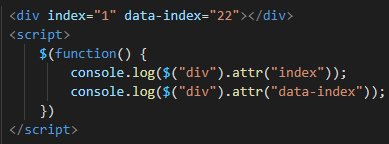
1. **获得/修改元素 自定义属性**

* **attr( ) 获得自定义属性**

1类似原生JS的**getAttribute( ‘属性’)**

只写属性不写值，是获得该自定义属性。自定义属性名加引号。

也可获得**H5新增的 data- 开头**的自定义属性

 1

* **attr( ) 修改自定义属性**

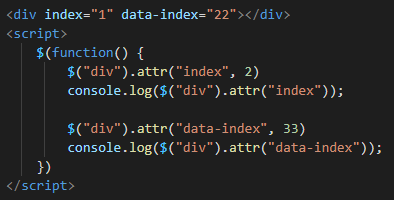
类似原生JS的**setAttribute( ‘属性’，‘值’)**

1

写了属性名又写值，是修改该自定义属性的值。

属性名加引号，属性名和属性值用逗号隔开。

也可修改**H5新增的 data- 开头**的自定义属性。



1

1. **获得/添加元素 数据缓存属性**

* **data( ) 获得H5新增data-开头属性**

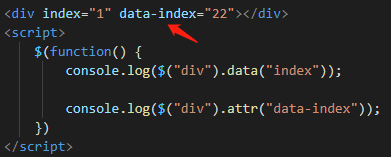
1

用attr( )可以获取普通自定义属性，也可获取data-开头的自定义属性。

获得的都是字符串，

用data( )只能获取data-开头的自定义属性。

获得的会看数据本身返回字符串 或数字型。

1

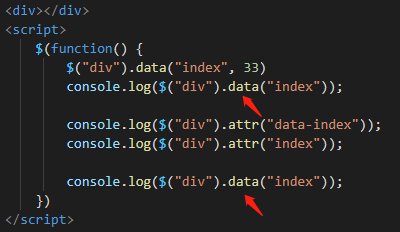
* **data( ) 数据缓存 添加属性**

1

给元素的缓存中添加属性，**添加的该属性不会显示在HTML结构中**。

1在缓存中添加的该元素**只能再被data( )获取**。

属性名省略data-开头。





**六．jQuery元素内容**

**1.获得/修改元素 HTML结构+内容**

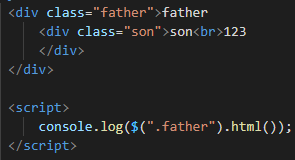
* **html( ) 获得结构+文本内容**

类似原生JS的**innerHTML**

1

括号里不写值，是获取该元素的**保留HTML结构**的内容。

以字符串形式获取HTML标签+文本内容。

1

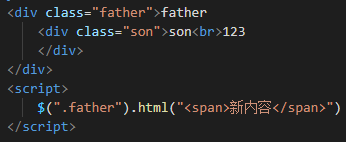
* **html( ) 添加/修改结构+文本内容**

1类似原生JS的**innerHTML=‘标签+文本内容’**

括号里要写HTML标签和文本内容，加引号。

括号里写内容是给该元素添加 / 修改（覆盖）内容。

1新内容会覆盖原先的结构内容。



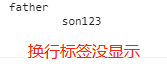
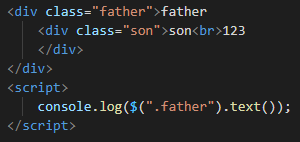
**2.获得/修改元素 文本内容**

* **text( ) 仅获得文本内容**

1类似原生JS的**innerText**

括号里不写值，是以字符串形式获取文本内容。

会分开显示父级和子级 的内容，但是不会返回HTML标签，只返回文本。

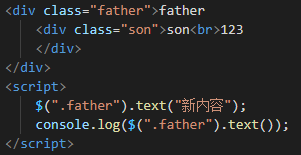


* **text( ) 添加/修改文本内容**

1类似原生JS的**innerText=”内容“**

括号里写内容是给该元素添加 / 修改（覆盖）内容。

新内容会覆盖原先的结构内容。

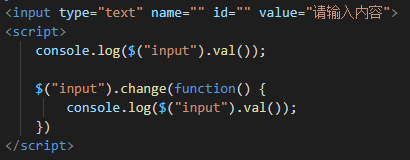
1

**3.获得/修改input表单 内容**

* **val( ) 获得表单内容**

类似原生JS的 **表单. value**

括号里不写值，是获取该表单的内容。

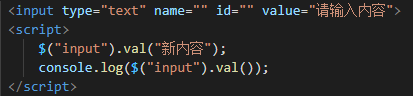
1

* **val( ) 添加/修改表单内容**

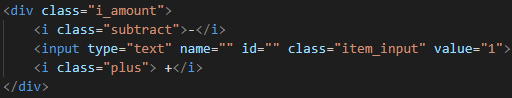
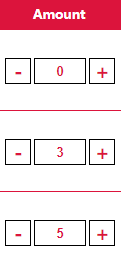
类似原生JS的 **表单. value=‘值’**

括号里写值，是修改/添加该表单的内容。

新内容会覆盖原先的结构内容。

1

**案例** - 增添商品的数量

 ****

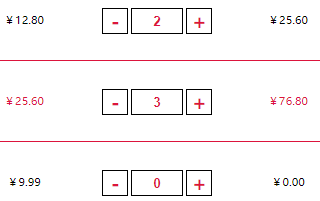
点击表单两侧的 + - 会增减表单的内容。

注意商品的个数不能减到0以下。

注意 + - 增减的是**他们的兄弟元素**input表单的内容。

**return false是直接结束判断。**

**案例** - 根据修改的商品数量修改总价



* **toFixed( number ) 保留小数点后位数**

**浮点的数字型数据 . toFixed( 2) 保留小数点后第2位数字。**

利用change表单变化事件，在表单内数字（商品个数）变化时，

获得表单的内容（个数）和单价（字符串变为数字型并去掉￥字符），相乘

再拼接加上￥字符后，赋值给标签内容。

注意分清 this指向、父级元素、父级的兄弟元素。

****

**七．jQuery操作元素**

**1.遍历元素**

jQuery虽然还有隐式迭代，但是**隐式迭代**是自动遍历**同类元素执行相同操作**。

jQuery的遍历元素，可以对同类元素进行不同操作，隐式迭代无法实现。

* **each( ) 遍历DOM元素**

1主要用作DOM处理，比如修改元素的样式、获取元素的内容等。

**index**：是每个元素的序号，可以自定义名为index或 i

**domEle**：是遍历的每个DOM元素。可以自定义名

**是DOM对象，不能用jQuery的方法函数**。

若要对每个对象使用jQuery方法，要转换为jQuery对象 **$(domEle)**

**function：是回调函数。**每遍历一个元素就执行一次回调函数。。



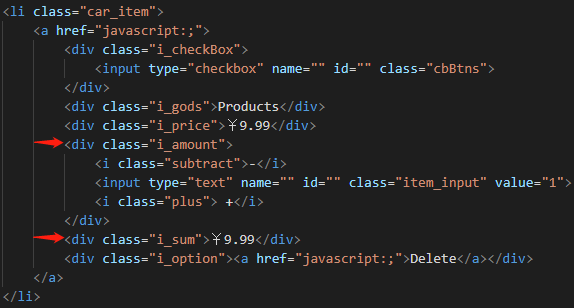
**案例** - 商品总数量和总价

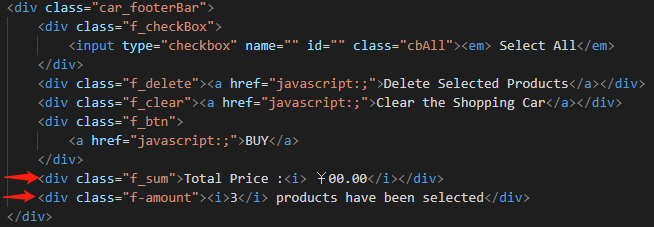


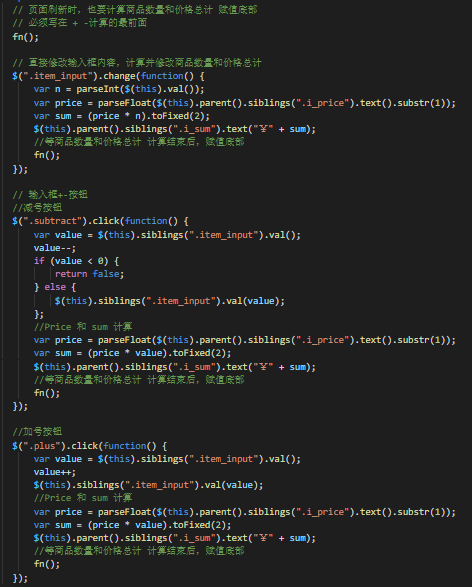
1

在+ - 计算完商品数量和价格总计后，调用该封装的函数。

**分别遍历相加商品的 个数和单价** ，然后分别相加，最后把结果分别赋值底部的标签内。







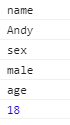
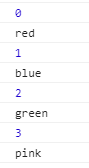
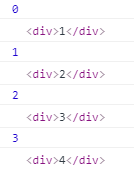
* **$.each( ) 遍历数据**

主要用于处理数据。**可以遍历DOM元素、对象、数组。**

**object** ：是遍历对象，要遍历谁就写谁。

**index：**是要遍历元素的索引号。索引号名可以自定义。

**element** ：是遍历的内容。



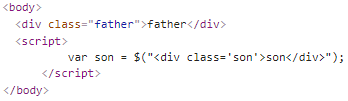
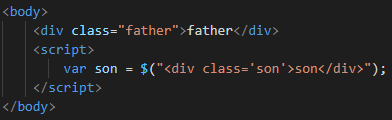
**2.创建元素**

* **$( ”标签+文本内容“)**

相当于原生JS的节点操作 **document.createElement（‘标签名’）**和 **innerHTML**



和原生JS一样，只创建元素不添加进某个父元素的话，新建元素不会再页面中显示。



**3.内部添加元素 做子元素**

* **append( ) 添加到元素内部最后**

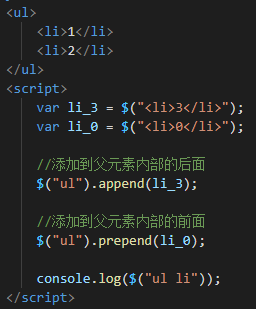
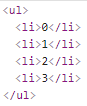
相当于原生JS的节点操作 **父元素.appendChild（ 子节点 ）**

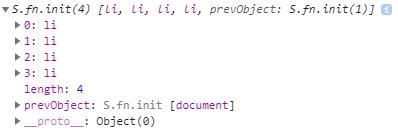
放到目标元素的**内部的最后面**

* **prepend( ) 添加到元素内部最前**



放到目标元素的**内部的最前面**





**4.外部添加元素 做兄弟元素**

* **after( ) 添加到元素外部最后**

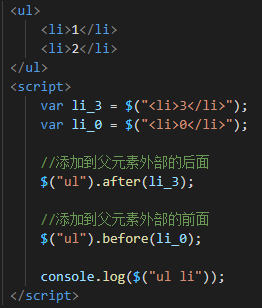
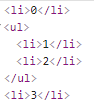


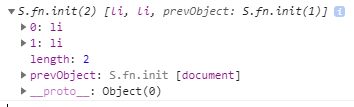
放到父元素的前面，作为父元素的兄弟元素

* **before( ) 添加到元素外部最前**



放到父元素的后面，作为父元素的兄弟元素





**5.删除元素**

* **remove( ) 删除元素自身**



删除元素自身+该元素的内容（子元素）。

* **empty( ) 删除元素的内容**



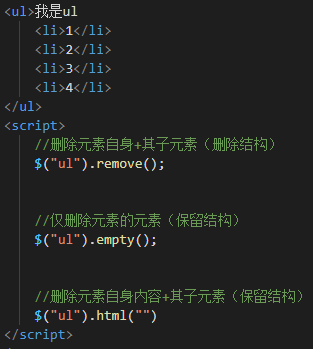
**仅删除元素的内容**（子元素），**保留该元素HTML结构**，页面中仍可见。

* **html(“”) 删除元素的内容**

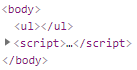


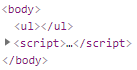
就是用html( )给元素内容赋值为空。

**仅删除元素的内容**（子元素），**保留该元素HTML结构**，页面中仍可见。

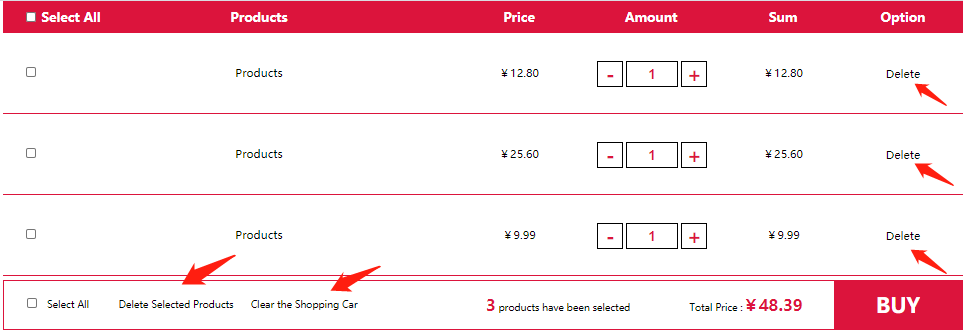


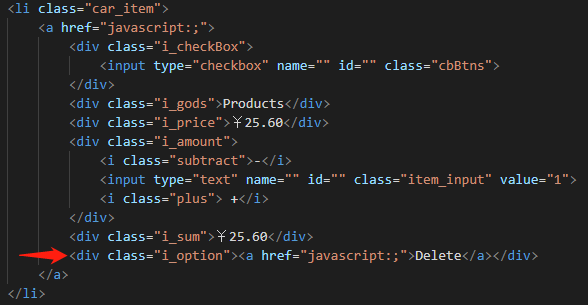
1





**案例** - 删除商品





1



**6.拷贝/合并 对象**

* **$.extend( ) 拷贝对象**

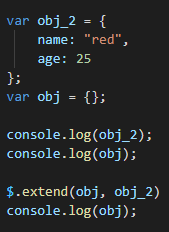
1

**$.extend( 深拷贝/浅拷贝 拷贝给谁， 拷贝谁)**

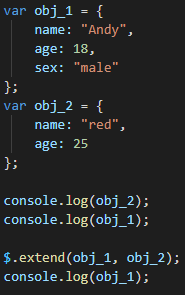
**true深拷贝，**不写的话**默认浅拷贝**。

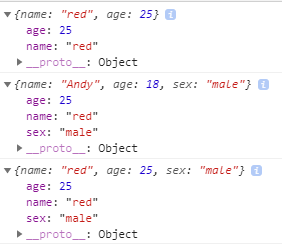
**1.拷贝给的对象为空**

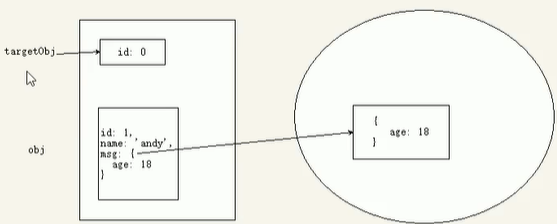
被拷贝的对象会完全复制给该空对象



**2.拷贝给的对象原有内容**

被拷贝的对象会**复制**给该拷贝给的对象里**没有的内容**，并**覆盖** 拷贝给的对象里**相同的内容**。



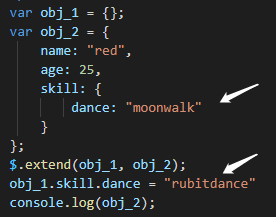
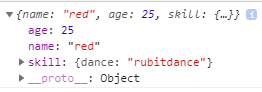


**3.浅拷贝**

默认浅拷贝。

会把被拷贝对象的**复杂数据（函数）的地址** 复制过去，

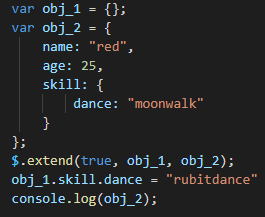
若在拷贝给的对象中修改该复杂数据，会沿着地址**直接修改被拷贝对象的复杂数据**

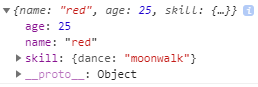


**4.深拷贝**

**true深拷贝，**

不拷贝地址，是直接把被拷贝对象的复杂数据类型复制一份，然后给拷贝给的对象。

在拷贝给的对象中修改该复杂数据，只是修改自己，并**不会修改被拷贝对象的复杂数据**

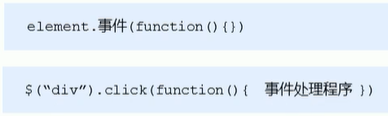


**八．jQuery事件**

**1.事件注册/解绑**

**元素.事件(执行程序) 单个事件注册**

事件名的写法和原生JS的类似，

比如mouseover、mouseout、click、keydown、keyup、blur、scroll等等

缺点是一次只能注册一个事件。若同一元素注册多个事件会麻烦，

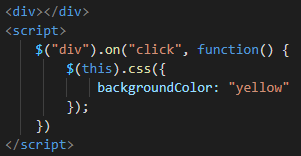


* 1**on( ) 注册事件 推荐**

可以注册单一事件、同一元素注册多个事件、操作匹配元素的子元素

**1.on( ) 注册单个事件**

**元素 . on( function( ){ } )**



**2.on( ) 注册多个事件**

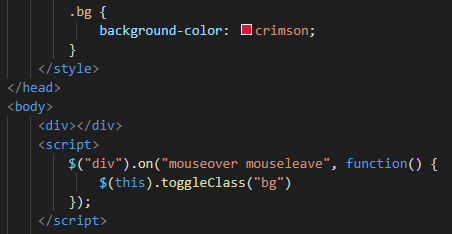
**元素 . on( { function( ){ } , function( ){ } } )**

以对象格式写多个事件，看作对象的方法，用逗号隔开。

事件名不加引号，键值对的形式写事件名和处理函数

若不同事件，只是切换同一样式的话，

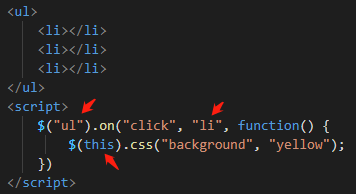
**元素 . on( ”事件1 事件2“ ，function( ){ })**



**3.on( ) 实现事件委派(事件委托)**

利用事件冒泡，把事件绑定到会触发事件的子元素的父元素上。

**触发事件元素的父元素. on(“事件”，”触发事件的元素“ )**

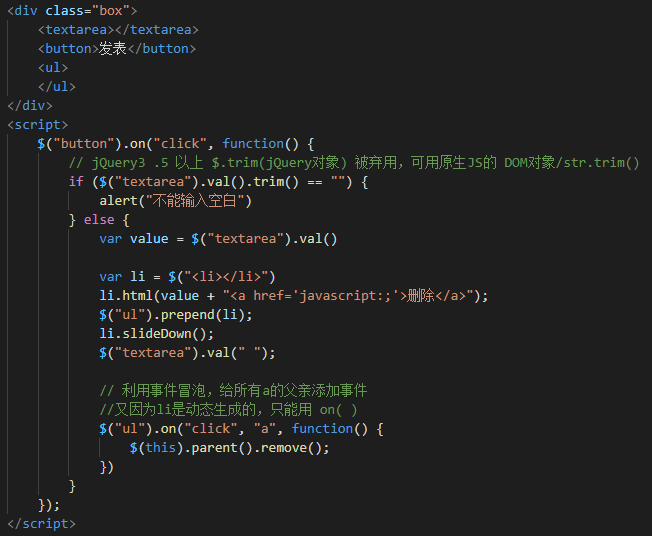
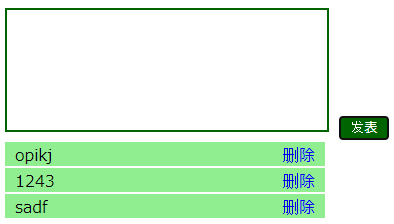


**3.on( ) 给动态生成的元素绑定事件**

当前不存在但将来后面会被创建添加的元素，要绑定事件只能用on( ).



**案例** - 简易留言板



* **off( ) 解绑事件**

可以移除on( ) 绑定的事件。

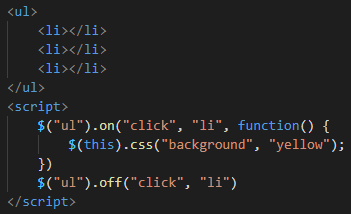
**元素. off( ) 解绑该元素的所有事件**

**元素. off( ”事件“ ) 解绑该元素的指定事件**



**off( ) 解绑 on( )的事件委托**

**绑定事件的元素. off(”事件“,”触发事件的对象”)**



* **one( ) 仅执行一次事件**

one( ) 绑定的事件只能触发一次，不需要解绑。

使用方法和on( )类似。

**2.自动触发事件**

比如 轮播图再点击左右箭头时会切换图片，自动播放时就是自动触发了点击箭头的事件

**元素.事件名( )**

会自动触发这个事件

1

* **trigger( )**

**元素 . trigger(“事件名” )**

1

* **triggerHandler( )**

**元素 . triggerHandler(“事件名” )**

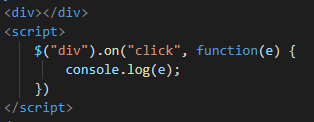
1

**不触发元素的默认行为**。

比如用 triggerHandler(“focus”)不会让input表单的光标闪烁

**3.事件对象**

1和原生JS的事件对象一样。



**阻止默认行为**

和原生JS的事件对象一样。

比如阻止<a>的跳转，或者阻止if{ }的继续判断、for的继续循环.....

**e.preventDefault( )**

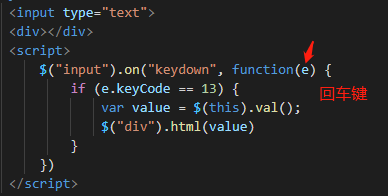
**return fasle**

**阻止事件冒泡**

和原生JS的事件对象一样。

**e.stopPropagation( )**

**案例** - 判断键盘按下

利用事件对象的event.keyCode

**3.常用事件**

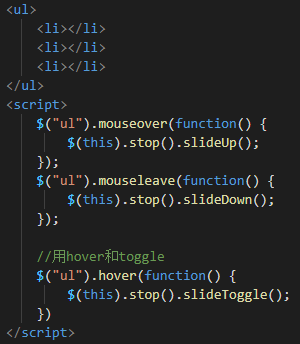
* **hover( ) 鼠标经过和离开**

可看作**mouseover 和mouseleave的简写**。

该事件包含了 **经过 和 离开**，所以事件函数要写两个执行函数，分别是经过和离开的效果

直接用可以切换的toggle系列

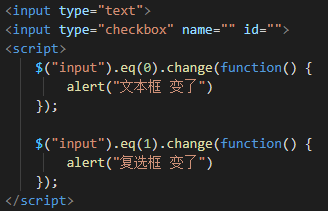
**hover( function( ){ toggle系列的动画效果 } )**



* **change( ) input表单变化事件**

**表单内容**改变了，就执行处理程序

**表单的选定状态**（**checked=true/false**）改变了，就就执行处理程序



**九．jQuery尺寸、位置操作**

**1.元素尺寸 高度宽度**

不写参数，是获得高度宽度，返回的是数字型。

写参数，是修改元素的高宽。参数可以不写单位。

1

**2.元素位置**

* **offset( ) 元素距离文档的位偏移 可读写**

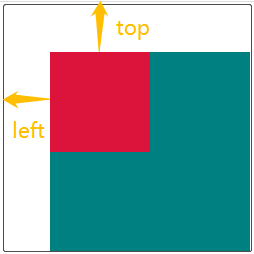
**元素 . offset( )**

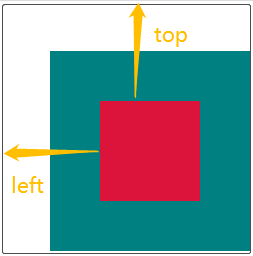
返回该元素**相对于文档document**的偏移坐标。

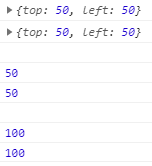
不写参数，是获得该元素**相对于文档**的**top和left**坐标。

**offset( )的返回值是对象形式，top和left**分别是对象的属性**。**

** 写参数，是修改元素的top和left。**







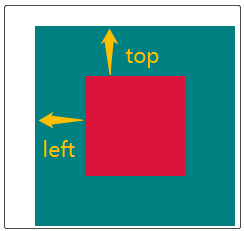
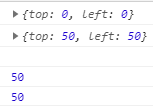
* **position( ) 元素距离定位父元素的位偏移 只读**

返回该元素**相对于带有定位的父元素**的偏移坐标。

**position( )的返回值是对象形式，top和left**分别是对象的属性**。**

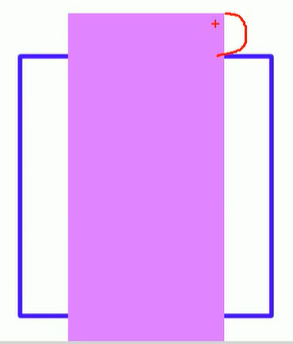
**只能获得位置，不能修改位置**。

和CSS的定位**position**一样，**父元素有定位则根据父元素，若父元素都没定位则根据文档。**



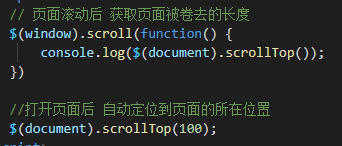
* **scrollTop( )/scrollLeft 页面滚动后元素卷去的部分**

和原生JS的BOOM的scroll类似。



可以获取**页面windo滚动后，文档document 的被卷去的部分。**

也可以设置页面打开时，页面所在的位置。



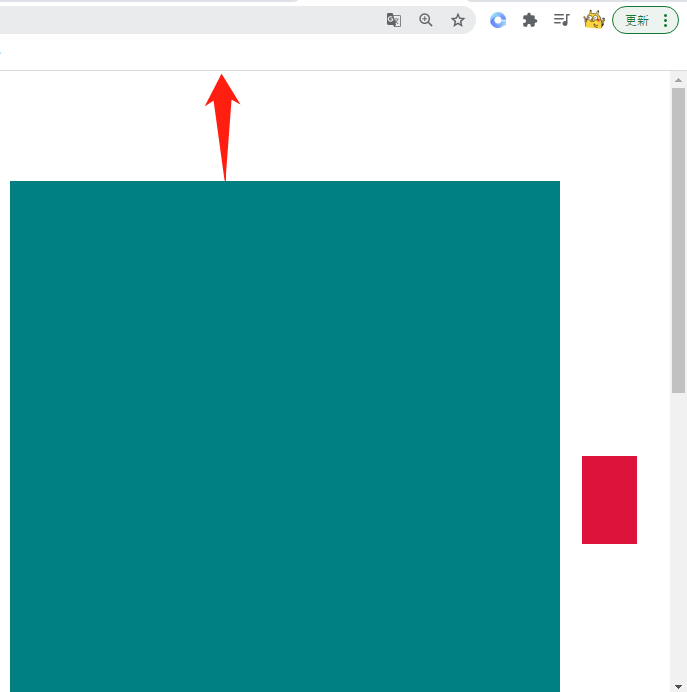
**案例** - 滚动页面显示导航条

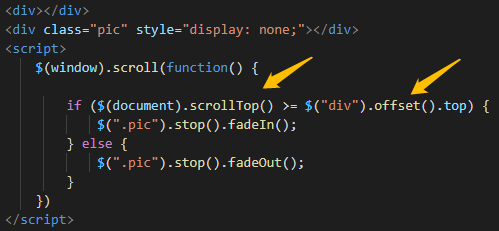
等页面滚动到指定位置后，才让红色盒子显示。

比如，指定页面滚动到 蓝色盒子的顶部位置时，红色盒子显示。

即，若页面卷去的部分 大于等于 蓝色盒子的顶部到页面距离，则显示红色盒子

若页面卷去的部分 小于 蓝色盒子的顶部到页面距离，则隐藏红色盒子





**案例** - 页面**直接跳转**到指定位置

点击了指定元素后，让页面**scrollTop( 指定位置 )**



**案例** - 页面**动画滚动**到指定位置

利用jQuery的**动画函数animate**( )的**scrollTop属性**

**元素 . animate( { scrollTop : 指定top位置 } )**

但是animate( )的让元素做动画，但是要滚动的是页面document。

**用$(“body，html”) 代替 document**

**案例** - 电梯导航