# jQuery基础语法

1. JavaScript概述

仓库：可以把很多东西放在这个仓库里面，找东西只需要到仓库里面查找到就可以了。

JavaSript库：即library,是一个封装好的特定的集合（方法和函数），从封装一大堆函数的角度理解库，就是在这个库中，封装了很多预先定义好的函数，比如动画animate、hide、show比如获取元素等。

简单理解：就是一个JS文件，里面对我们原生js代码进行了封装，存放到里面，这样我们可以快速高效的使用这些封装好的功能了。

比如jQuery就是为了快速方便的操作DOM，里面基本都是函数（方法）。

常见的JavaScript库：jQuery、Prototype、YUI、Dojo、Ext JS、移动端的zepto。

这些库都是对原生JavaScript的封装，内部都是用JavaScript实现的，我们主要学习的是jQuery。

1. jQuery的概述

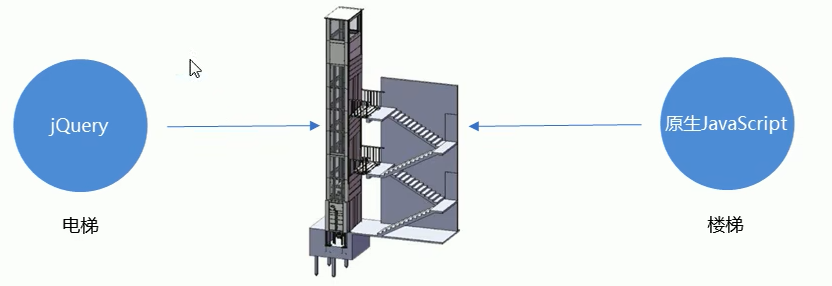
1、jQuery是一个快速、简洁的JavaScript库，其设计的宗旨是’write Less,Do More’，即倡导写更少的代码，做更多的事情。

2、j就是JavaScript; Query查询; 意思是查询js，把js中的DOM操作做了封装，我们可以快速的查询使用里面的功能。

3、jQuery封装了JavaScript常用的功能代码，优化了DOM操作，事件处理，动画设计和Ajax交互。学习jQuery本质：就是学习调用这些函数（方法）。

4、jQuery的优点：

* 轻量级。核心文件才几十kb，不会影响页面的加载速度。
* 跨浏览器兼容。基本兼容了现在主流的浏览器。
* 链式编程、隐式迭代。
* 对事件、样式、动画支持，大大简化了DOM操作
* 支持插件扩展开发，有丰富的第三方插件，如树形菜单、日期控件、轮播图等
* 免费、开源



1. jQuery的下载

官方网站：<https://jquery.com/>

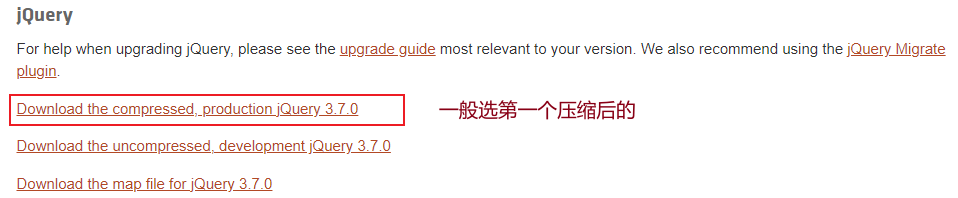
版本:

1x：兼容IE 678等低版本浏览器，官网不再更新

2x：不兼容IE 678等低版本浏览器，官网不再更新

3x：不兼容IE 678等低版本浏览器，是官网主要更新维护的版本

各个版本的下载：https://code.jquery.com/



1. jQuery的入口函数

函数一: $(document).ready(function(){

...//此处是页面DOM加载完成的入口

});

函数二：$(function(){

...//此处是页面DOM加载完成的入口

});

1. 等着DOM结构渲染完毕即可执行内部代码，不必等到所有外部资源加载完毕，jQuery帮我们完成了封装。
2. 相当于原生js的DOMContentLoaded。
3. 不同于原生js中的load事件是等页面文档、外部的js文件、css文件、图片加载完毕才执行的内部代码。
4. jQuery的顶级对象$
5. $是jQuery的别称，在代码中可以使用jQuery代替$,但一般为了方便，通常直接使用$.
6. $是jQuery的顶级对象，相当于原生javaScript中的window,把元素利用$包装成jQuery对象，就可以调用jQuery的方法。
7. jQuery对象和DOM对象
8. 用原生JS获取来的对象就是DOM对象
9. jQuery方法获取的元素就是jQuery对象
10. jQuery对象本质是：利用$对DOM对象包装后产生的对象（伪数组形式存储）
11. DOM对象和jQuery对象之间是可以相互转换的。
12. 因为原生js比jQuery更大，原生的一些属性和方法jQuery没有给我们封装，要想使用这些属性和方法需要把jQuery对象转换为DOM对象才能使用。
13. DOM对象转换为jQuery对象: $(DOM对象) $(‘div’)
14. jQuery对象转换为DOM对象(两种方式)

$(‘div’)[index] $(‘div’).get(index) index是索引号

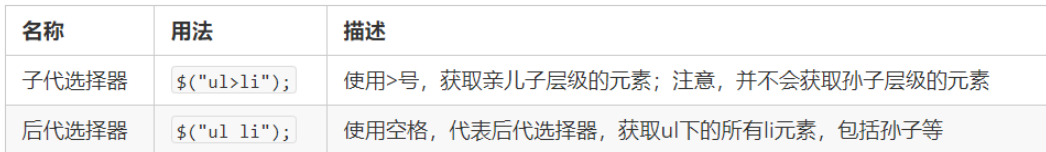
1. jQuery基础选择器
2. jQuery选择器

原生 JS 获取元素方式很多，很杂，而且兼容性情况不一致，因此 jQuery 给我们做了封装，使获取元素统一标准。

$(“选择器”) // 里面选择器直接写 CSS 选择器即可，但是要加引号



1. jQuery层级选择器



1. jQuery 设置样式

$('div').css('属性', '值')

案例：

$("div").css("background", "pink");

1. 隐式迭代

遍历内部 DOM 元素（伪数组形式存储）的过程就叫做隐式迭代。

简单理解：给匹配到的所有元素进行循环遍历，执行相应的方法，而不用我们再进行循环，简化我们的操作，方便我们调用。

1. jQuery的筛选选择器





重点记住： parent() children() find() siblings() eq()

**案例：微博下拉菜单案例**

1. jQuery的排他思想

想要多选一的效果，排他思想：当前元素设置样式，其余的兄弟元素清除样式。

$(this).css(“color”,”red”);

$(this).siblings(). css(“color”,””);

**案例：淘宝服饰精品案例**

① 核心原理：鼠标经过左侧盒子某个小li，就让内容区盒子相对应图片显示，其余的图片隐藏。

② 需要得到当前小li 的索引号，就可以显示对应索引号的图片

③ jQuery 得到当前元素索引号 $(this).index()

④ 中间对应的图片，可以通过 eq(index) 方法去选择

⑤ 显示元素 show() 隐藏元素 hide()

1. 链式编程

链式编程是为了节省代码量，看起来更优雅。

$(this).css('color', 'red').sibling().css('color', '');

1. jQuery样式操作
2. 操作CSS方法

jQuery 可以使用 css 方法来修改简单元素样式；也可以操作类，修改多个样式。

(1) 参数只写属性名，则是返回属性值

$(this).css(''color'');

(2) 参数是属性名，属性值，逗号分隔，是设置一组样式，属性必须加引号，值如果是数字可以不用跟单位和引号

$(this).css(''color'', ''red'');

(3) 参数可以是对象形式，方便设置多组样式。属性名和属性值用冒号隔开， 属性可以不用加引号，

$(this).css({ "color":"white","font-size":"20px"});

1. 设置类样式方法

作用等同于以前的 classList，可以操作类样式， 注意操作类里面的参数不要加点。

(1) 添加类

$(“div”).addClass(''current'');

(2) 移除类

$(“div”).removeClass(''current'');

(3) 切换类

$(“div”).toggleClass(''current'');

**案例：tab 栏切换**

① 点击上部的li，当前li 添加current类，其余兄弟移除类。

② 点击的同时，得到当前li 的索引号

③ 让下部里面相应索引号的item显示，其余的item隐藏

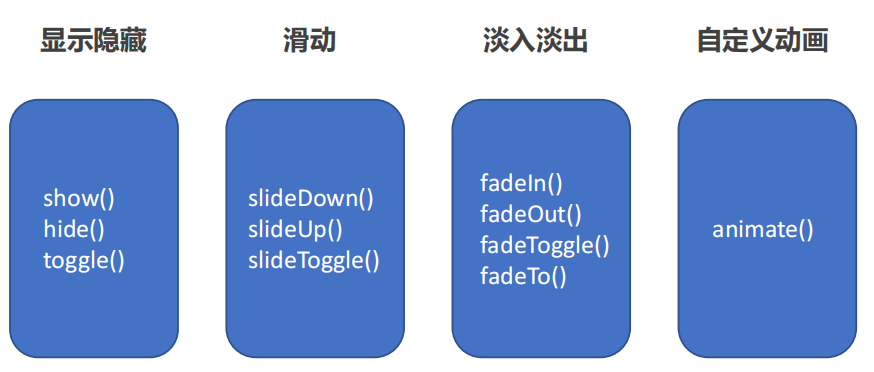
1. 类操作与className区别

原生 JS 中 className 会覆盖元素原先里面的类名。

jQuery 里面类操作只是对指定类进行操作，不影响原先的类名。相当于追加类名，不影响之前定义的类名。

1. jQuery效果

jQuery 给我们封装了很多动画效果，最为常见的如下：



1. 显示隐藏效果

1-1. 显示语法规范

show([speed,[easing],[fn]])

1-2. 显示参数

（1）参数都可以省略， 无动画直接显示。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

1-3. 隐藏语法规范

hide([speed,[easing],[fn]])

1-4. 隐藏参数

（1）参数都可以省略， 无动画直接显示。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

1-5. 切换语法规范

toggle([speed,[easing],[fn]])

1-6. 切换参数

（1）参数都可以省略， 无动画直接显示。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

建议：平时一般不带参数，直接显示隐藏即可。

1. 滑动效果

2-1. 下滑效果语法规范

slideDown([speed,[easing],[fn]])

2-2. 下滑效果参数

（1）参数都可以省略。

（2）speed:三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing:(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

2-3. 上滑效果语法规范

slideUp([speed,[easing],[fn]])

2-4. 上滑效果参数

（1）参数都可以省略。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

2-5. 滑动切换效果语法规范

slideToggle([speed,[easing],[fn]])

2-6. 滑动切换效果参数

（1）参数都可以省略。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

1. 事件切换

hover([over,]out)

（1）over:鼠标移到元素上要触发的函数（相当于mouseenter）

（2）out:鼠标移出元素要触发的函数（相当于mouseleave）

（3）如果只写一个函数，则鼠标经过和离开都会触发它

$(".nav>li").hover(function() {

                 $(this).children("ul").slideDown(200);

             }, function() {

                 $(this).children("ul").slideUp(200);

             });

1. 动画队列及其停止排队方法

4-1. 动画或效果队列

动画或者效果一旦触发就会执行，如果多次触发，就造成多个动画或效果排队执行。

4-2. 停止排队

stop()

1. stop() 方法用于停止动画或效果。

(2) 注意： stop() 写到动画或者效果的前面， 相当于停止结束上一次的动画。

1. 淡入淡出效果

5-1. 淡入效果语法规范

fadeIn([speed,[easing],[fn]])

5-2. 淡入效果参数

（1）参数都可以省略。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次

5-3. 淡出效果语法规范

fadeOut([speed,[easing],[fn]])

5-4. 淡出效果参数

（1）参数都可以省略。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次

5-5. 淡入淡出切换效果语法规范

fadeToggle([speed,[easing],[fn]])

5-6. 淡入淡出切换效果参数

（1）参数都可以省略。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

5-7. 渐进方式调整到指定的不透明度

fadeTo([[speed],opacity,[easing],[fn]])

5-8. 效果参数

（1）opacity 透明度必须写，取值 0~1 之间。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。必须写

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

**案例：页面突出显示效果**

利用fadeTo()

1. 自定义动画 animate

6-1. 语法

animate(params,[speed],[easing],[fn])

6-2. 参数

（1）params: 想要更改的样式属性，以对象形式传递，必须写。 属性名可以不用带引号,如果是复合属性则需要采取驼峰命名法 borderLeft。其余参数都可以省略。

（2）speed：三种预定速度之一的字符串(“slow”,“normal”, or “fast”)或表示动画时长的毫秒数值(如：1000)。

（3）easing：(Optional) 用来指定切换效果，默认是“swing”，可用参数“linear”。

（4）fn: 回调函数，在动画完成时执行的函数，每个元素执行一次。

**案例：王者荣耀手风琴效果**

① 鼠标经过某个小li 有两步操作：

② 当前小li 宽度变为 224px， 同时里面的小图片淡出，大图片淡入

③ 其余兄弟小li宽度变为69px， 小图片淡入， 大图片淡出

1. jQuery属性操作
2. 设置或获取元素固有属性值 prop()

所谓元素固有属性就是元素本身自带的属性，比如 <a> 元素里面的 href ，比如 <input> 元素里面的 type。

1-1. 获取属性语法

prop(''属性'')

1-2. 设置属性语法

prop(''属性'', ''属性值'')

Input元素的单选框常用的事件是change

1. 设置或获取元素自定义属性值 attr()

用户自己给元素添加的属性，我们称为自定义属性。 比如给 div 添加 index =“1”。

2-1. 获取属性语法

attr(''属性'') // 类似原生 getAttribute()

2-2. 设置属性语法

attr(''属性'', ''属性值'') // 类似原生 setAttribute()

改方法也可以获取 H5 自定义属性

1. 数据缓存 data()

data() 方法可以在指定的元素上存取数据，并不会修改 DOM 元素结构。一旦页面刷新，之前存放的数据都将被移除。

3-1. 附加数据语法

data(''name'',''value'') // 向被选元素附加数据

3-2. 获取数据语法

date(''name'') // 向被选元素获取数据

data() 方法可以在指定的元素上存取数据，并不会修改 DOM 元素结构。一旦页面刷新，之前存放的数据都将被移除。

同时，还可以读取 HTML5 自定义属性 data-index ，得到的是数字型

**案例：购物车案例模块-全选**

① 全选思路：里面3个小的复选框按钮（

j-checkbox）选中状态（checked）跟着全选按钮（checkall）走。

② 因为checked 是复选框的固有属性，此时我们需要利用prop()方法获取和设置该属性。

③ 把全选按钮状态赋值给3小复选框就可以了。

④ 当我们每次点击小的复选框按钮，就来判断：

⑤ 如果小复选框被选中的个数等于3 就应该把全选按钮选上，否则全选按钮不选。

⑥ :checked 选择器

:checked 查找被选中的表单元素。

1. jQuery文本属性值

主要针对元素的内容还有表单的值操作。

1. 普通元素内容 html( ) (相当于原生inner HTML)

html() // 获取元素的内容

1. 普通元素文本内容 text() (相当与原生 innerText)

html(''内容'') // 设置元素的内容

text() // 获取元素的文本内容

text(''文本内容'') // 设置元素的文本内容

1. 表单的值 val( ) (相当于原生value)

val() // 获取表单的值

val(''内容'') // 设置表单的值

**案例：购物车案例模块-增减商品数量**

① 核心思路：首先声明一个变量，当我们点击+号（increment），就让这个值++，然后赋值给文本框。

② 注意1： 只能增加本商品的数量， 就是当前+号的兄弟文本框（itxt）的值。

③ 修改表单的值是val() 方法

④ 注意2： 这个变量初始值应该是这个文本框的值，在这个值的基础上++。要获取表单的值

⑤ 减号（decrement）思路同理，但是如果文本框的值是1，就不能再减了。

**案例：购物车案例模块-修改商品小计**

① 核心思路：每次点击+号或者-号，根据文本框的值 乘以 当前商品的价格 就是 商品的小计

② 注意1： 只能增加本商品的小计， 就是当前商品的小计模块（

p-sum）

③ 修改普通元素的内容是text() 方法

④ 注意2： 当前商品的价格，要把￥符号去掉再相乘 截取字符串 substr(1)

⑤ parents(‘选择器’) 可以返回指定祖先元素

⑥ 最后计算的结果如果想要保留2位小数 通过 toFixed(2) 方法

⑦ 用户也可以直接修改表单里面的值，同样要计算小计。 用表单change事件

⑧ 用最新的表单内的值 乘以 单价即可 但是还是当前商品小计

1. jQuery元素操作

主要是遍历、创建、添加、删除元素操作。

1. 遍历元素

jQuery 隐式迭代是对同一类元素做了同样的操作。 如果想要给**同一类元素做不同操作，就需要用到遍历**。

语法1：

$("div").each(function (index, domEle) { xxx; }）

（1）each() 方法遍历匹配的每一个元素。主要用DOM处理。 each 每一个

（2）里面的回调函数有2个参数： index 是每个元素的索引号; demEle 是每个DOM元素对象，不是jquery对象

（3） 所以要想使用jquery方法，需要给这个dom元素转换为jquery对象 $(domEle)

语法2：

$.each(object，function (index, element) { xxx; }）

（1） $.each()方法可用于遍历任何对象。主要用于数据处理，比如数组，对象

（2）里面的函数有2个参数： index 是每个元素的索引号; element 遍历内容

**案例：购物车案例模块-计算总计和总额**

① 核心思路：把所有文本框里面的值相加就是总计数量。总额同理

② 文本框里面的值不相同，如果想要相加需要用到each遍历。声明一个变量，相加即可

③ 点击+号-号，会改变总计和总额，如果用户修改了文本框里面的值同样会改变总计和总额

④ 因此可以封装一个函数求总计和总额的， 以上2个操作调用这个函数即可。

⑤ 注意1： 总计是文本框里面的值相加用 val() 总额是普通元素的内容用text()

⑥ 要注意普通元素里面的内容要去掉￥并且转换为数字型才能相加

1. 创建元素

语法：

$(''<li></li>'');

动态的创建了一个 <li>

1. 添加元素

3-1. 内部添加

element.append(''内容'')

把内容放入匹配元素内部最后面，类似原生 appendChild。

element.prepend(''内容'')

把内容放入匹配元素内部最前面。

3-2. 外部添加

element.after(''内容'') // 把内容放入目标元素后面

element.before(''内容'') // 把内容放入目标元素前面

① 内部添加元素，生成之后，它们是父子关系。

② 外部添加元素，生成之后，他们是兄弟关系。

1. 删除元素

element.remove( ) // 删除匹配的元素（本身）

element.empty( ) // 删除匹配的元素集合中所有的子节点

element.html('' '') // 清空匹配的元素内容

① remove 删除元素本身。

② empt() 和 html('''') 作用等价，都可以删除元素里面的内容，只不过 html 还可以设置内容。

**案例：购物车案例模块-删除商品模块**

① 核心思路：把商品remove() 删除元素即可

② 有三个地方需要删除：

1. 商品后面的删除按钮 2. 删除选中的商品 3. 清理购物车

③ 商品后面的删除按钮： 一定是删除当前的商品，所以从 $(this) 出发

④ 删除选中的商品： 先判断小的复选框按钮是否选中状态，如果是选中，则删除对应的商品

⑤ 清理购物车： 则是把所有的商品全部删掉

**案例：购物车案例模块-选中商品添加背景**

① 核心思路：选中的商品添加背景，不选中移除背景即可

② 全选按钮点击：如果全选是选中的，则所有的商品添加背景，否则移除背景

③ 小的复选框点击： 如果是选中状态，则当前商品添加背景，否则移除背景

④ 这个背景，可以通过类名修改，添加类和删除类

1. jQuery尺寸、位置操作
2. jQuery 尺寸



*  以上参数为空，则是获取相应值，返回的是数字型。
*  如果参数为数字，则是修改相应值。
*  参数可以不必写单位。

1. jQuery 位置

位置主要有三个： offset()、position()、scrollTop()/scrollLeft()

（1）. offset() 设置或获取元素偏移

① offset() 方法设置或返回被选元素相对于文档的偏移坐标，跟父级没有关系。

② 该方法有2个属性 left、top 。

offset().top 用于获取距离文档顶部的距离，

offset().left 用于获取距离文档左侧的距离。

③ 可以设置元素的偏移：offset({ top: 10, left: 30 });

（2） position() 获取元素偏移

① position() 方法用于返回被选元素相对于带有定位的父级偏移坐标，如果父级都没有定位，则以文档为准。

② 该方法有2个属性 left、top。

position().top 用于获取距离定位父级顶部的距离，

position().left 用于获取距离定位父级左侧的距离。

③ 该方法只能获取。

（3）. scrollTop() / scrollLeft() 设置或获取元素被卷去的头部和左侧

① scrollTop() 方法设置或返回被选元素被卷去的头部。

② 不跟参数是获取，参数为不带单位的数字则是设置被卷去的头部。

**案例：带有动画的返回顶部**

① 核心原理： 使用animate动画返回顶部。

② animate动画函数里面有个scrollTop 属性，可以设置位置

③ 但是是**元素做动画**，因此 $(“body,html”).animate({scrollTop: 0})

**案例： 品优购电梯导航**

① 当我们滚动到 今日推荐 模块，就让电梯导航显示出来

② 点击电梯导航页面可以滚动到相应内容区域

③ 核心算法：因为电梯导航模块和内容区模块一一对应的

④ 当我们点击电梯导航某个小模块，就可以拿到当前小模块的索引号

⑤ 就可以把animate要移动的距离求出来：当前索引号内容区模块它的offset().top

⑥ 然后执行动画即可

**案例： 品优购电梯导航**

① 当我们点击电梯导航某个小li， 当前小li 添加current类，兄弟移除类名

② 当我们页面滚动到内容区域某个模块， 左侧电梯导航，相对应的小li模块，也会添加current类， 兄弟移除current类。

③ 触发的事件是页面滚动，因此这个功能要写到页面滚动事件里面。

④ 需要用到each，遍历内容区域大模块。 each里面能拿到内容区域每一个模块元素和索引号

⑤ 判断的条件： 被卷去的头部 大于等于 内容区域里面每个模块的offset().top

⑥ 就利用这个索引号找到相应的电梯导航小li添加类

1. jQuery事件
2. 事件注册

单个事件注册

语法：

element.事件(function(){})

$(“div”).click(function(){ 事件处理程序 })

其他事件和原生基本一致。

比如mouseover、mouseout、blur、focus、change、keydown、keyup、resize、scroll

1. 事件处理

2-1 . 事件处理 on() 绑定事件

on() 方法在匹配元素上绑定一个或多个事件的事件处理函数

语法：

element.on(events,[selector],fn)

(1). events:一个或多个用空格分隔的事件类型，如"click"或"keydown" 。

(2). selector: 元素的子元素选择器 。

(3). fn:回调函数 即绑定在元素身上的侦听函数。

①on( )方法优势1： **可以绑定多个事件，多个处理事件处理程序**。

$(“div”).on({

mouseover: function(){},

mouseout: function(){},

click: function(){}

});

如果事件处理程序相同

$(“div”).on(“**mouseover mouseout**”, function() {

$(this).toggleClass(“current”);

});

②on( ) 方法优势2： **可以事件委派操作** 。

事件委派的定义:把原来加给子元素身上的事件绑定在父元素身上，就是把事件委派给父元素。

$('ul').on('click', 'li', function() {

alert('hello world!');

});

在此之前有bind(), live() delegate()等方法来处理事件绑定或者事件委派，最新版本的请用on替代他们。

③on() 方法优势3： **动态创建的元素，click() 没有办法绑定事件， on() 可以给动态生成的元素绑定事件 。**

$(“div").on("click",”p”, function(){

alert("俺可以给动态生成的元素绑定事件")

});

$("div").append($("<p>我是动态创建的p</p>"));

**案例：发布微博案例**

① 点击发布按钮， 动态创建一个小li，放入文本框的内容和删除按钮， 并且添加到ul 中。

② 点击的删除按钮，可以删除当前的微博留言。

2-2 . 事件处理 off() 解绑事件

off( )方法 **可以移除通过 on() 方法添加的事件处理程序**。

$("p").off() // 解绑p元素所有事件处理程序

$("p").off( "click") // 解绑p元素上面的点击事件 后面的 foo 是侦听函数名

$("ul").off("click", "li"); // 解绑事件委托

如果有的事件只想触发一次， 可以使用 one() 来绑定事件。

2-3自动触发事件 trigger()

(1)有些事件希望自动触发, 比如轮播图自动播放功能跟点击右侧按钮一致。

可以利用定时器自动触发右侧按钮点击事件，不必鼠标点击触发。

element.click() // 第一种简写形式

element.trigger("type") // 第二种自动触发模式

$("p").on("click", function () {

alert("hi~");

});

$("p").trigger("click"); // 此时自动触发点击事件，不需要鼠标点击

(2)有些事件希望自动触发, 比如轮播图自动播放功能跟点击右侧按钮一致。可以利用定时器自动触发右侧按钮点击事件，不必鼠标点击触发。

element.triggerHandler(type) // 第三种自动触发模式

triggerHandler模式不会触发元素的默认行为，这是和前面两种的区别

1. 事件对象

**事件被触发，就会有事件对象的产生**。

element.on(events,[selector],function(event) {})

阻止默认行为： event.preventDefault() 或者 return false

阻止冒泡： event.stopPropagation()

1. jQuery其他方法

1. jQuery 对象拷贝

**如果想要把某个对象拷贝（合并）给另外一个对象使用，此时可以使用 $.extend()**

语法：

$.extend([deep], target, object1, [objectN])

(1). deep: 如果设为true 为深拷贝， 默认为false 浅拷贝

(2). target: 要拷贝的目标对象

(3). object1:待拷贝到第一个对象的对象。

(4). objectN:待拷贝到第N个对象的对象。

(5). 浅拷贝是把被拷贝的对象复杂数据类型中的地址拷贝给目标对象，修改目标对象会影响被拷贝对象。

(6). 深拷贝，前面加true， 完全克隆(拷贝的对象,而不是地址)，修改目标对象不会影响被拷贝对象。

$(function () {

            var targetObj = {};

            var obj = {

                id: 1,

                name: "andy"

            }

            $.extend(targetObj, obj);

 var targetObj = {

                id: 0

            };

            var obj = {

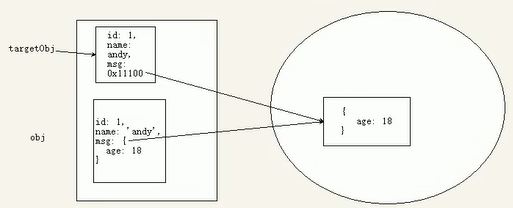
                id: 1,

                name: "andy"

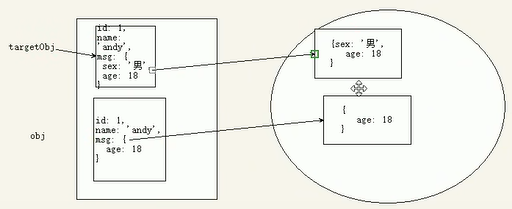
            }

            $.extend(targetObj, obj);

            console.log(targetObj); //会覆盖targetObj里面的数据，targetObj没有的数据会往里面添加



**targetObj与obj是指向同一个对象，当targetObj的age修改了表示右边对象的值修改了，实验obj的值也变了。**



//完全克隆(拷贝的对象,而不是地址)，修改目标对象不会影响被拷贝对象

            $.extend(true, targetObj1, obj1);

            //console.log(targetObj1);

            targetObj1.msg.age = 20; //修改age参数不会影响obj的age值，只会影响targetObj1

            console.log(targetObj1);//此时sex参数也在里面

            console.log(obj1); //此时sex参数不在里面

2. jQuery 多库共存

问题概述：

jQuery使用$作为标示符，随着jQuery的流行,其他 js 库也会用这$作为标识符， 这样一起使用会引起冲突。

客观需求：

需要一个解决方案，让jQuery 和其他的js库不存在冲突，可以同时存在，这就叫做多库共存。

jQuery 解决方案：

1. 把里面的 $ 符号 统一改为 jQuery。 比如 jQuery(''div'')

2. jQuery 变量规定新的名称：$.noConflict() var xx = $.noConflict();

3. jQuery 插件

jQuery 功能比较有限，想要更复杂的特效效果，可以借助于 jQuery 插件完成。

注意: 这些插件也是依赖于jQuery来完成的，所以必须要先引入jQuery文件，因此也称为 jQuery 插件。

3-1 jQuery 插件常用的网站：

(1). jQuery 插件库 http://www.jq22.com/

(2). jQuery 之家 http://www.htmleaf.com/

3-2 jQuery 插件使用步骤：

(1). 引入相关文件。（jQuery 文件 和 插件文件）

(2). 复制相关html、css、js (调用插件)。

3-3 jQuery 插件演示：

(1). 瀑布流

(2). 图片懒加载（图片使用延迟加载在可提高网页下载速度,它能帮助减轻服务器负载）

当我们页面滑动到可视区域，再显示图片。

我们使用jquery 插件库 EasyLazyload。

注意，此时的js引入文件和js调用必须写到 DOM元素（图片）最后面

(3). 全屏滚动（fullpage.js）

gitHub： https://github.com/alvarotrigo/fullPage.js

中文翻译网站： <http://www.dowebok.com/demo/2014/77/>

3-4 bootstrap JS 插件：

bootstrap 框架也是依赖于 jQuery 开发的，因此里面的 js插件使用 ，也必须引入jQuery 文件。

使用bootstrap 时先定义盒子div（class=container）再把对应html代码放入盒子

**案例：toDoList**

① 文本框里面输入内容，按下回车，就可以生成待办事项。

② 点击待办事项复选框，就可以把当前数据添加到已完成事项里面。

③ 点击已完成事项复选框，就可以把当前数据添加到待办事项里面。

**④ 但是本页面内容刷新页面不会丢失。**

**案例分析**

① 刷新页面不会丢失数据，因此需要用到本地存储 localStorage

**② 核心思路： 不管按下回车，还是点击复选框，都是把本地存储的数据加载到页面中，这样保证刷新关闭页面不会丢失数据**

③ 存储的数据格式：var todolist = [{ title : ‘xxx’, done: false}]

④ 注意点1： 本地存储 localStorage 里面只能存储字符串格式 ，因此需要把对象转换为字符串 JSON.stringify(data)。

⑤ 注意点2： 获取本地存储数据，需要把里面的字符串转换为对象格式JSON.parse() 我们才能使用里面的数据。

**案例：toDoList 按下回车把新数据添加到本地存储里面**

① 切记： 页面中的数据，都要从本地存储里面获取，这样刷新页面不会丢失数据，所以先要把数据保存到本地存储里面。

② 利用事件对象.keyCode判断用户按下回车键（13）。

③ 声明一个数组，保存数据。

④ 先要读取本地存储原来的数据（声明函数 getData()），放到这个数组里面。

⑤ 之后把最新从表单获取过来的数据，追加到数组里面。

⑥ 最后把数组存储给本地存储 (声明函数 savaDate())

**案例：toDoList 本地存储数据渲染加载到页面**

① 因为后面也会经常渲染加载操作，所以声明一个函数 load，方便后面调用

② 先要读取本地存储数据。（数据不要忘记转换为对象格式）

③ 之后遍历这个数据（$.each()），有几条数据，就生成几个小li 添加到 ol 里面。

④ 每次渲染之前，先把原先里面 ol 的内容清空，然后渲染加载最新的数据。

**案例：toDoList 删除操作**

① 点击里面的a链接，不是删除的li，而是删除本地存储对应的数据。

**② 核心原理：先获取本地存储数据，删除对应的数据，保存给本地存储，重新渲染列表li**

③ 我们可以给链接自定义属性记录当前的索引号

④ 根据这个索引号删除相关的数据----数组的splice(i, 1)方法

⑤ 存储修改后的数据，然后存储给本地存储

⑥ 重新渲染加载数据列表

⑦ 因为a是动态创建的，我们使用on方法绑定事件

**案例：toDoList 正在进行和已完成选项操作**

**① 当我们点击了小的复选框，修改本地存储数据，再重新渲染数据列表。**

② 点击之后，获取本地存储数据。

③ 修改对应数据属性 done 为当前复选框的checked状态。

④ 之后保存数据到本地存储

⑤ 重新渲染加载数据列表

⑥ load 加载函数里面，新增一个条件,如果当前数据的done为true 就是已经完成的，就把列表渲染加载到 ul 里面

⑦ 如果当前数据的done 为false， 则是待办事项，就把列表渲染加载到 ol 里面

**案例：toDoList 统计正在进行个数和已经完成个数**

① 在我们load 函数里面操作

② 声明2个变量 ：todoCount 待办个数 doneCount 已完成个数

③ 当进行遍历本地存储数据的时候， 如果 数据done为 false， 则 todoCount++, 否则 doneCount++

④ 最后修改相应的元素 text()