1. 使用jQuery
2. 是什么

jQuery是一个JS函数库：write less ，do more，它封装简化DOM操作（CRUD/Ajax）

1. 为什么用

强大选择器、隐式遍历（迭代）、读写合一、事件处理、链式调用、DOM操、样式操作…

1. 怎么用

首先需要引入jQuery库（本地引用/CDN远程引用），通过script标签引入，

然后使用jQuery核心函数（$/jQuery）及jQuery核心对象（执行$()返回的对象）

$(function(){})绑定文档加载完成的监听，

再在这个function里面添加事件监听

ps：添加监听就是添加回调函数

1. jQuery的两把利器
2. jQuery核心函数（直接可用）
3. jQuery库向外直接暴露的就是$/jQuery
4. 作为一般函数调用: $(param)，根据参数的不同有四种不同的类型

1).参数为函数 : 当DOM加载完成后，执行此回调函数。相当于window.onload=function()

2).参数为选择器字符串: 查找所有匹配的标签, 并将它们封装成jQuery对象

3).参数为DOM对象: 将dom对象封装成jQuery对象返回 $(this)

ps：this是什么，是发生事件的dom元素

4).参数为html标签字符串 (用得少): 创建标签对象并封装成jQuery对象

1. 作为对象使用: $.xxx()

1). $.each(obj/arr,function(key,value){}) : 隐式遍历数组

2). $.trim(str) : 去除两端的空格

1. jQuery核心对象
2. 执行jQuery核心函数得到

jQuery对象内部包含的是dom元素对象的伪数组对象（可能只有一个元素）

jQuery对象拥有很多有用的属性和方法, 让程序员能方便的操作dom

1. 属性/方法（伪数组）

length/size()，获取dom元素的个数

[index]/get(index)，获取下标为index的dom元素

each(function(index,domEle){ })，遍历所有dom元素

index()，获取当前dom元素在所有兄弟中的下标

1. jQuery核心函数
2. 选择器
3. 类型：

* 基本选择器

#id、tagName、.class、

selector1，selector2，selector3：并集

selector1selector2selector3：交集

* 层次选择次——查找子孙后代或兄弟

selector1>selector2：子元素

selector1 selector2：后代元素

* 过滤选择器（基本、内容、可见性、属性）——在原有选择器匹配的元素中进一步进行过滤的选择器；多个过滤选择器不是同时执行的，而是依次执行

:first、:last、:eq(index)、:not(selector)、:gt、:lt、:odd、:even、:hidden、:visible、

:[attrName]、:[attrName=value]

* 表单选择器

:input、:text、:checkbox:、tadio、:checked（选中的）

1. 工具
2. ajax
3. jQuery对象
4. 属性：操作标签的属性、标签体文本
5. 通用属性操作

attr(name) / attr(name,value)：读写(设置)非布尔值的标签属性（是覆盖的效果）

prop(name) / prop(name,value)：读写布尔值的标签属性

removeAttr(name) / removeProp(name)：删除属性

1. 操作class属性

addClass(classValue)：添加class（是累加的效果）

removeClass(classValue)：移除指定class

1. 操作value属性

val() / val(value)：读写标签的value

1. 操作标签体文本

html() / html(htmlString)：读写标签体文本

1. CSS
2. 操作CSS：$().css(name,value)
3. 操作位置

offset()：相对于页面左上角的位置。返回的对象包含top和left

position()：相对于父元素左上角的位置。返回的对象包含top和left

scrollTop()：得到/设置div或页面滚动条的坐标

读取页面时，使用$(document.documentElement/document.body)兼容IE/Chrome

设置页面时，使用$(‘html,body’)兼容IE/Chrome

1. 尺寸

height()、width()：内容尺寸，即height&width

innerHeight()、innerWidth()：内部尺寸，即height+padding&width+padding

outerHeight(false/true)、outerWidth(false/true)外部尺寸，即height+padding+border&

&width+padding+border，如果是true，则要加上margin

1. 筛选
2. 过滤

first()、last()、eq(index)

filter(selector)、not(selector)：对当前元素提要求

has(selector)：对子孙元素提要求

1. 查找（孩子-父母-兄弟）

children(selector)：子标签

find(selector)：后代标签

parent()：父标签

prevAll(selector)：前面的所有兄弟

siblings(selector)：所有兄弟标签

1. 文档处理（CUD）
2. 插入

append(‘’)：向选中元素的内部最后追加内容

prepend(‘’)：向选中元素的内部最前面添加内容

before(‘’)：向选中元素的前面添加内容

after(‘’)：向选中元素的后面添加内容

1. 替换

replaceWith(‘标签’)：替换选中的元素为replaceAll中的标签

1. 移除

empty()：删除匹配的元素集合中所有的子节点

remove()：从DOM中删除所有匹配的元素

1. 事件
2. 事件绑定与解绑

绑定：on(eve,fun)/eventName(fun)

解除：off()

停止事件冒泡：event.stopPropagation()

阻止事件默认行为：event.preventDefault()

1. mouseover与mouseenter的区别

mouseover在移入子元素时也会触发（认为离开了当前元素，对应mouseout

mouseenter只在移入当前元素时才触发，对应mouseleave

hover()使用的就是mouseenter()和mouseleave()

1. 事件委托

* 概念：

将子元素的事件监听委托给父元素处理

监听回调是加在了父辈元素上

当操作任何一个子元素时，时间会冒泡到父元素

父元素不会直接处理事件，而是根据event.target得到发生事件的子元素，通过这个子元素调用事件回调函数。回调函数的this是子元素，而不是父元素

* 实现：

设置事件委拖：父元素.delegate(‘子元素’，‘事件名’，function(){})

移除事件委托：父元素.undelegate(‘事件名‘)

* 好处

添加新的子元素，会自动有事件响应处理；减少事件监听的数量

1. 事件坐标

event.offsetX：原点是当前元素左上角

event.clientX：原点是窗口左上角

event.pageX：原点是页面左上角

1. 效果（内置动画）
2. 淡入淡出：通过不断改变元素的透明度（opacity）来实现的

fadeIn()淡入 fadeout()淡出 fadeToggle()切换出/入

1. 滑动：通过不断改变元素的高度实现

slideUp()带动画的收缩 slideDown()带动画的展开slideToggle()切换

1. 显示/隐藏：，默认是没有动画的，动画是通过不断改变宽高和透明度来实现

show()显示 hide()隐藏 toggle()切换

1. 自定义动画

animate({结束时的样式}，时间，函数)定义动画 stop()停止动画

1. 多库共存

如果有2个库都有$，就存在冲突，jQuery库可以释放$的使用权（jQuery.noConflict()），让另一个库可以正常使用，此时jQuery库只能使用jQuery了

1. 区别window.onload与$(document).ready()（相当于$(function(){}))

window.onload包括页面的图片加载完后才会回调（晚）、只能有一个监听回调

$(document).ready()页面加载完就回调（早）、可以有多个监听回调

1. jQuery插件
2. 扩展jQuery函数对象的方法

$.extend{

xxx:function(){ } //this是$

}

调用：$.xxx()

1. 扩展jQuery对象的方法

$.fn.extend{

xxx:function(){ } //this是jQuery对象

}

调用：$obj.xxx()