正课:

1. 什么是JQUERY

2. 为什么用JQUERY

3. 如何使用:

4. 查询:

1. 什么是JQUERY: 第三方开发的，极简化的DOM操作的函数库

2. 为什么:

1. 是DOM操作的终极简化: write less do more

1.增删改查

2.事件

3.动画

4.AJAX

学习jQuery其实还是在学DOM操作，只不过API简化

2. 解决了兼容性问题:

凡是jQuery允许使用的，都解决了兼容性问题

3. 如何:

下载: 官网

版本: 1.x 兼容旧浏览器

jquery-1.11.3.js 未压缩 体积大 可读性好

jquery-1.11.3.min.js 压缩 体积小 可读性差

差别: 去掉了注释和空字符(空格，缩进，换行)

极简化了变量和参数名

使用专门的压缩软件生成的

2.x 不再兼容旧浏览器

引入: <script src="jquery-1.11.3.js"></script>

1. 向window中加入了一个jQuery函数(查找，创建)

构造/工厂函数

2. 为今后生成的jQuery对象添加了公共的原型对象:

jQuery.fn: 其中封装了所有jQuery的简化版API

CDN: 内容分发网络: 根据客户端距离某台服务器的网络状况，智能选择最优的服务器下载资源

使用API:

1. 都要使用工厂函数创建一个jQuery对象

2种:

1. 直接用jQuery("selector")

查找符合selector要求的DOM元素，封装进jQuery对象中。

jQuery对象其实是一个封装DOM元素对象的类数组对象

为什么: 原生的DOM元素不具有jQuery的简化版API。必须封装进jQuery对象，才能调用简化版API

2. 如果已经获得一个DOM对象:

可用jQuery函数封装DOM对象为jQuery对象:

jQuery(DOM对象)

终极简化: 所有jQuery(...), 都可简化为$(...)

window.jQuery=window.$=jQuery;

2. 调用jQuery对象的简化版API，操作DOM

4. \*\*\*查找:

1. 按选择器查找:

基本选择器: 5个: 同CSS

#id element .class \* selector1,selector2,...

练习: 补充:

修改css样式: DOM: elem.style.css属性=值

JQ: $(...).css("属性名",值)

绑定事件: DOM: elem.onclick=function(){...}

JQ: $(...).click(function(){...})

层级选择器: 2组 4个: 同CSS

父子: parent child 所有后代

parent>child 直接子代

兄弟: child1+next 紧邻child1的下一个，且满足next的要求

child1~nextsiblings 不必紧邻，只要满足nextsiblings的要求

过滤选择器:

子元素过滤: 同CSS

选择在父元素下指定位置的子元素

:first-child 第一个子元素

:last-child 最后一个子元素

:nth-child(n/even/odd/2n/2n+1)

:only-child 仅有的一个子元素

问题: 旧浏览器不支持:last-child,:nth-child

解决: 凡是jQuery让使用的，都已经解决了兼容性问题

基本过滤: CSS中没有！

选择查找结果中指定位置的元素

:first/last

:even/odd

:eq/gt/lt(i)

:not(selector) 反选一切选择器

基本过滤 vs 子元素过滤:

子元素过滤 相对于自己的父元素内定位，下标从1开始

基本过滤 先将所有元素放入结果集合，再在结果集合中定义，下标从0开始

练习: 同时修改多个css属性;

$(...).css({

属性名:值,

属性名:值,

...

})

强调: jQuery中的css属性名都要去横线变驼峰

数字属性值，不用加单位

内容过滤: 根据元素的内容，选择满足条件的父元素

:parent 选择只要包含内容（文字,子元素）的任意父元素

:empty 选择不包含任何内容的空的父元素

:contains(text) 选择内容中包含指定文本的父元素

比如: span:contains("购物车")

:has(selector) 选择内容中包含符合selector要求的子元素的父元素

<li><a

<li><a class=active; li:has(a.active)

<li><a

可见性过滤: :visible

:hidden 只能选择dispaly为none的

和input type=hidden的

属性选择器: 同CSS

[属性名] 选择包含指定属性的元素

[属性名=值] 选择包含指定属性且属性值为指定值的元素

[属性名^=值] 以值开头

[属性名$=值] 以值结尾

[属性名\*=值] 包含值

[属性名!=值] :not([属性名=值])

selector1selector2... 而且

练习: jQuery事件处理函数的this可获得当前DOM对象

但无法使用jQuery简化版API

需要$(this)

练习: 添加/移除/判断class

$(...).addClass("class")

$(...).removeClass("class")

$(...).hasClass("class")

$(...).toggleClass("class"):

if($(...).hasClass("class")) $(...).removeClass("class")

else $(...).addClass("class")

jQuery API通用规律:

1. 所有API自带遍历:

对jQuery对象调用的修改，会自动应用到集合中每个DOM元素上

2. 所有API都返回正在查询的结果集合，可使用链式操作对同一个结果集合执行多个操作。

3. 一个API两用: 如果提供了新值，就修改。

每提供新值就获取

修改: $(...).css(属性名,值)

获取: var value=$(...).css(属性名)