准备: 微信小程序: WEB问题速查 小恐龙图标

1. 历届师兄师姐在今天课程中曾经问过的高频的经典的问题清单和解答: 小程序->首页->JQUERY-> day01

自己可能没有发现问题，但是想知道别的同学都问过哪些问题

2. 今日课程对应的重点案例的视频以及鄙视面试所需的扩展知识视频：小程序->在线->JQUERY-> day01

(学有余力!!!的同学或自己认为自己基础好的同学必须看!!!!!!)：

今日扩展视频：

小程序->在线->DOM->day04

5. 鼠标坐标x y 原生DOM拖拽效果

作业: 原生DOM页面滚动动画效果

作业: 学子商城DOM版 中 day02最后——放大镜

赠送：原生DOM轮播图

作业: 高频笔试题nodejs浏览器不同事件循环

3. 今日课程中包含的英语单词的翻译：小程序->单词->JQUERY->day01

英文不太好的同学，可以提前看一下第二天需要用到哪些英文单词。

约定: 2件事:

1. 第三阶段禁止上课跟着敲笔记和敲代码！谁抄笔记抄的勤，谁抄代码抄的勤，谁学的越烂！

2. 每个完整功能的案例/视频，至少写三遍: (如果需要画图，必须亲手画图)

(1). 连代码+注释抄一遍 —— 明白这个例子是要做什么事儿

(2). 保留注释，自己试着看着注释，将代码填回来 —— 复习函数和对象

(3). 删除所有代码和注释，自己试着用自己的话，把注释写回来，再翻译为代码！—— 只有自己可以写出人话的注释，这个例子才算自己会！

如果没有问题的同学: （享受vip一对一答疑的时间还剩9天）

1. 上课时那句话没听懂，笔记中那句话看不懂，案例中那句话看不懂，立刻问！

2. 小程序问题清单中哪个问题或解答看不懂，立刻问！

3. 小程序视频中哪个视频看不懂，立刻问

4. 个人项目中哪个功能不会做，立刻问

5. 平时刷笔试题时，遇到不会的笔试题，立刻问

谁问的问题越多！谁的培训费交的越值！

正课:

1. 什么是jQuery

2. 如何使用jQuery  
3. jQuery的原理

4. 查找元素

一. 什么是jQuery:

对应小程序视频: 小程序->在线->JQUERY->day01 1. 什么是jQuery，如何使用jQuery，CDN

1. 什么是: 第三方开发的执行DOM操作的极简化的函数库

(1). 第三方: 需要先下载才能用！——不是原生，不能直接用！

(2). 执行DOM操作: jQuery还是执行DOM增删改查+事件绑定操作，学习jQuery还是在学习DOM。并不是一项全新的知识。

(3). 极简化: jQuery对DOM操作的每一步代码都进行了简化！

(4). 函数库: jQuery中用函数来解决一切问题！jQuery中没有属性的！所有jQuery操作都要加上()，因为都是函数调用！

2. 为什么: 2个原因:

(1). 简单

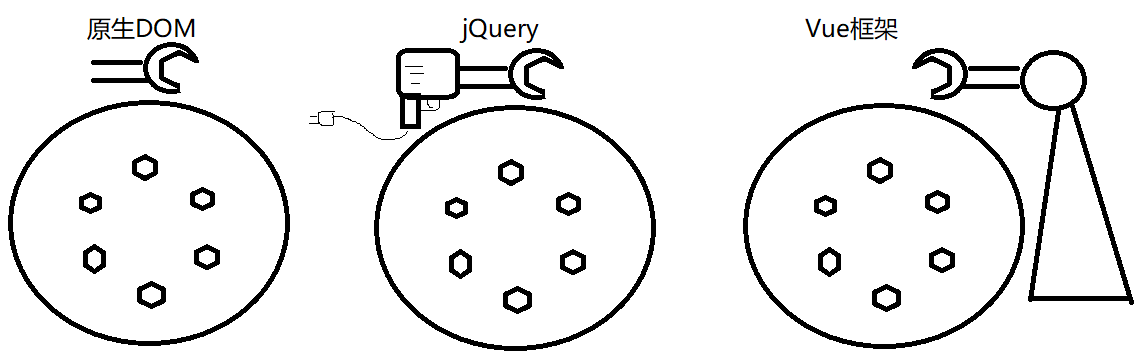
(2). 解决了大部分浏览器兼容性问题: 凡是jQuery让用的，都没有兼容性问题！

3. 何时: jQuery —— 用的越来越少了。除了旧项目维护还在用，新项目用的很少了！

4. 问题: 2个

(1). 只有PC端，没有移动端！

(2). 仅仅简化了DOM的每一步代码，但是没有从根本上简化开发流程，依然包含大量重复的低效率的代码！



二. 如何使用:

对应小程序视频: 小程序->在线->JQUERY->day01 2. 第一个jQuery程序 ...

1. 下载并引入页面: 2种:

(1). 去官网将jquery.js文件，下载到项目本地文件夹和项目源代码放在一起

a. 官网: jquery.com

b. 版本:

1.x 唯一兼容旧浏览器(IE8)

jquery-1.11.3.js

未压缩:

1). 拥有完备的注释，代码格式和见名知意的变量名——可读性好，但是体积大！

2). 适用于学习和开发之用。不适合于生产环境上线之用。因为体积大，客户端下载慢

压缩:

1). 删除了所有注释和代码格式，极简化了变量名——可读性差，但是体积小！

3). 不适用与学习和开发，更适用于生产环境上线之用。因为体积小，客户端下载快！

2.x 不再兼容旧浏览器，但是不支持ES6

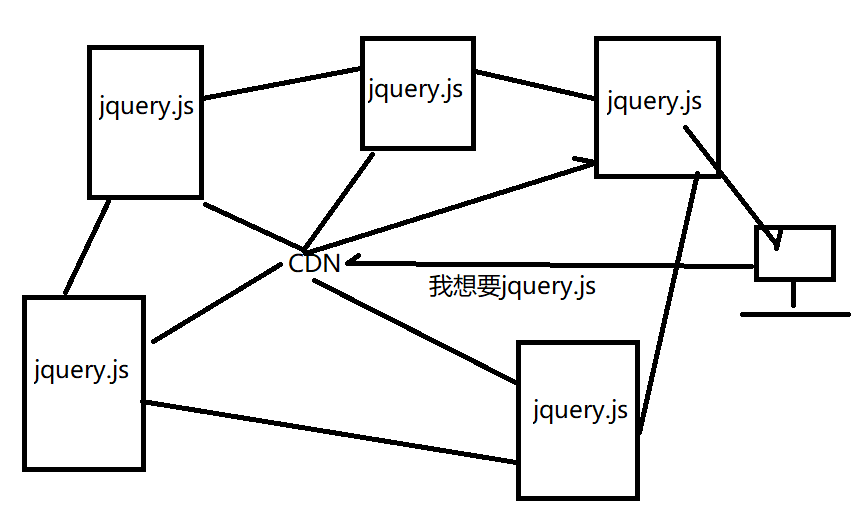
3.x不再兼容旧浏览器，开始支持ES6，比如for of ...

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

d. 缺点: js文件放在我们自己的服务器上，因为我们自己的服务器始终只在一个地点，距离近的用户访问快！如果用户距离我们服务器很远，则访问就会慢。

(2). 用CDN网络中共享的js文件:

a. CDN网络: 全球共享的可根据客户端距离某一台服务器的远近自动为客户端分配最优的服务器下载资源的智能的内容分发网络。



b. 何时: 如果我们使用全球公认的广泛使用的第三方js时，都可用CDN网络中的js文件，而不用下载到本地！

c. 优点: 无论用户身处何地，都可以从离自己最近的服务器下载资源，保证了最快的下载速度

d. 如何:

1). 不用下载

2). 用官网提供的CDN网址，直接引入网页

<script

src="https://code.jquery.com/jquery-1.12.4.js"

integrity="sha256-Qw82+bXyGq6MydymqBxNPYTaUXXq7c8v3CwiYwLLNXU="

crossorigin="anonymous"></script>

2. 示例: 我的第一个jQuery程序: 记录按钮点击次数

0\_DOM.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">  <meta charset="UTF-8">  <title></title>  </head>  <body>  <h1>jQuery/$工厂函数</h1>  <button id="btn1">click me(0)</button>  <script>  //DOM 4步  //1. 查找触发事件的元素  //本例中: 点击id为btn1的按钮触发变化，所以查找id为btn1的元素  var btn = document.getElementById("btn1");  //2. 绑定事件处理函数  //本例中: 单击按钮触发事件，所以为按钮绑定单击事件  btn.onclick = function () {  //3. 查找要修改的元素  //本例中: 单击按钮时，就是要修改自己: this  //4. 修改元素  //4.1 先取出当前按钮的内容中()中的数量，转为整数  var n = parseInt(  this.innerHTML  // "click me(10000)"  // 012345678901234  // | |  .slice(9, -1) //复习第一阶段slice函数用法  // "10000"  );  //4.2 将数量+1  n++;  //4.3 将新数量拼到()中，放回当前按钮的内容中  this.innerHTML = `click me(${n})`;  }  </script>  </body>  </html>  运行结果: |

1\_jQuery.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">  <meta charset="UTF-8">  <title></title>  </head>  <body>  <h1>jQuery/$工厂函数</h1>  <button id="btn1">click me(0)</button>  <script src="js/jquery-1.11.3.js"></script>  <script>  //jQuery还是按照  //DOM 4步  //1. 查找触发事件的元素  //本例中: 点击id为btn1的按钮触发变化，所以查找id为btn1的元素  //var btn=document.getElementById("btn1");  // new jQuery("#btn1"); //2件事儿:创建新子对象和查找DOM元素  // 等效于 ↑  var $btn = $("#btn1");  //2. 绑定事件处理函数  //本例中: 单击按钮触发事件，所以为按钮绑定单击事件  //btn.onclick=function(){  //自动 ↑ 翻译为  $btn.click(function () {  //3. 查找要修改的元素  //本例中: 单击按钮时，就是要修改自己: this  //$btn.click()自动翻译为button.onclick=function(){}  //所以function中的this->当前点击的DOM按钮元素  //this不能直接调用jquery家函数  //只能$(this)，才能创建一个包含this的jquery家子对象，才能调用简化版函数  var $this = $(this);  //4. 修改元素  //4.1 先取出当前按钮的内容中()中的数量，转为整数  var n = parseInt(  //this.innerHTML  //自动 ↑ 翻译为  $this.html()  // "click me(10000)"  // 012345678901234  // | |  .slice(9, -1) //复习第一阶段slice函数用法  // "10000"  );  //4.2 将数量+1  n++;  //4.3 将新数量拼到()中，放回当前按钮的内容中  //this.innerHTML=`click me(${n})`;  //自动 ↑ 翻译为  $this.html(`click me(${n})`);  })  </script>  </body>  </html>  运行结果: |

三. jQuery原理:

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

其实引入了一种新的类型:

(1). 构造函数: function jQuery("选择器"){

...

}

(2). 原型对象: jQuery.prototype={ 保存着所有jQuery简化版函数 }

2. 问题1: 如果想调用jQuery原型对象中保存的简化版函数，必须先创建jQuery类型的子对象

3. 解决: var btn=new jQuery("... ...")

结果: btn.click() btn.html() ... ...

强调: 只有jQuery家子对象，才能调用jQuery家简化版函数

DOM元素无法调用jQuery家简化版函数

jQuery家孩子也无法直接调用DOM家原生函数

——不是一家人不进一家门

4. 问题2: 因为jQuery原型对象中所有简化版函数几乎都是要操作DOM元素的！

比如: click() 代替了 onclick= 专门为元素绑定单击事件

html() 代替了 innerHTML 专门获取或修改元素的内容

但是，jQuery子对象怎么知道自己要操作哪个DOM元素呢?

5. 解决: jQuery()构造函数有一个选择器参数

(1). 当new jQuery("选择器")时，可传入任意一个css选择器

(2). new jQuery(): 2件大事儿

a. 和普通new一样，创建子对象，继承父对象，添加自有属性...

b. 还会自动用传入的选择器去DOM树上查找符合条件的DOM元素对象，保存进jQuery子对象中

6. 结果: 对jquery子对象调用简化版函数，简化版函数都会自动翻译为原生DOM对应的操作，并作用于当前jquery子对象中保存的DOM元素对象上

7. 问题: 每次想调用jQuery家简化版函数时，都要写new jQuery("...")，太麻烦

8. 解决: 2件事:

(1). 将new jQuery()这句话，封装在了jQuery构造函数内部

结果: 创建jQuery子对象时，不写new，也等效于new jQuery()了！

(2). 祖师爷觉得每次都写jQuery()，也太麻烦了！

$=jQuery //给jQuery起一个外号/别名, $

(3). 结果: 从此$()=jQuery()=new jQuery()

9. 总结: 今后只要想使用jQuery家的简化版函数，都要先创建jQuery家子对象，并用选择器查找要操作的DOM元素。

var $jquery对象=$("选择器") —— 所有jQuery操作的第一步

10. 问题: 如果jQuery子对象和DOM元素对象采用相同的命名规则，则很难区分到底是谁家孩子！

11. 解决: 行规: 只要是jQuery家子的对象名，必须以$开头

12. 问题: 用jQuery绑定的事件处理函数中this->?

13. 原理: 因为$btn.click(function(){ ...this... }) ——表面

会被自动 ↓ 翻译为

button.onclick=function(){ —— 本质

...this->当前正在单击的DOM按钮元素对象

}

14. 问题: this->DOM按钮，不能继续使用jQuery家简化版函数

15. 解决: 其实jQuery构造函数除了支持选择器查找之外，还支持直接将一个DOM元素放入jQuery子对象中，无需查找。所以，将来如果一个DOM元素，也想用jQuery函数，可以用$(DOM元素)包裹以下就变成了jQuery家对象了！

function jQuery("选择器" 或 一个DOM元素){

... ...

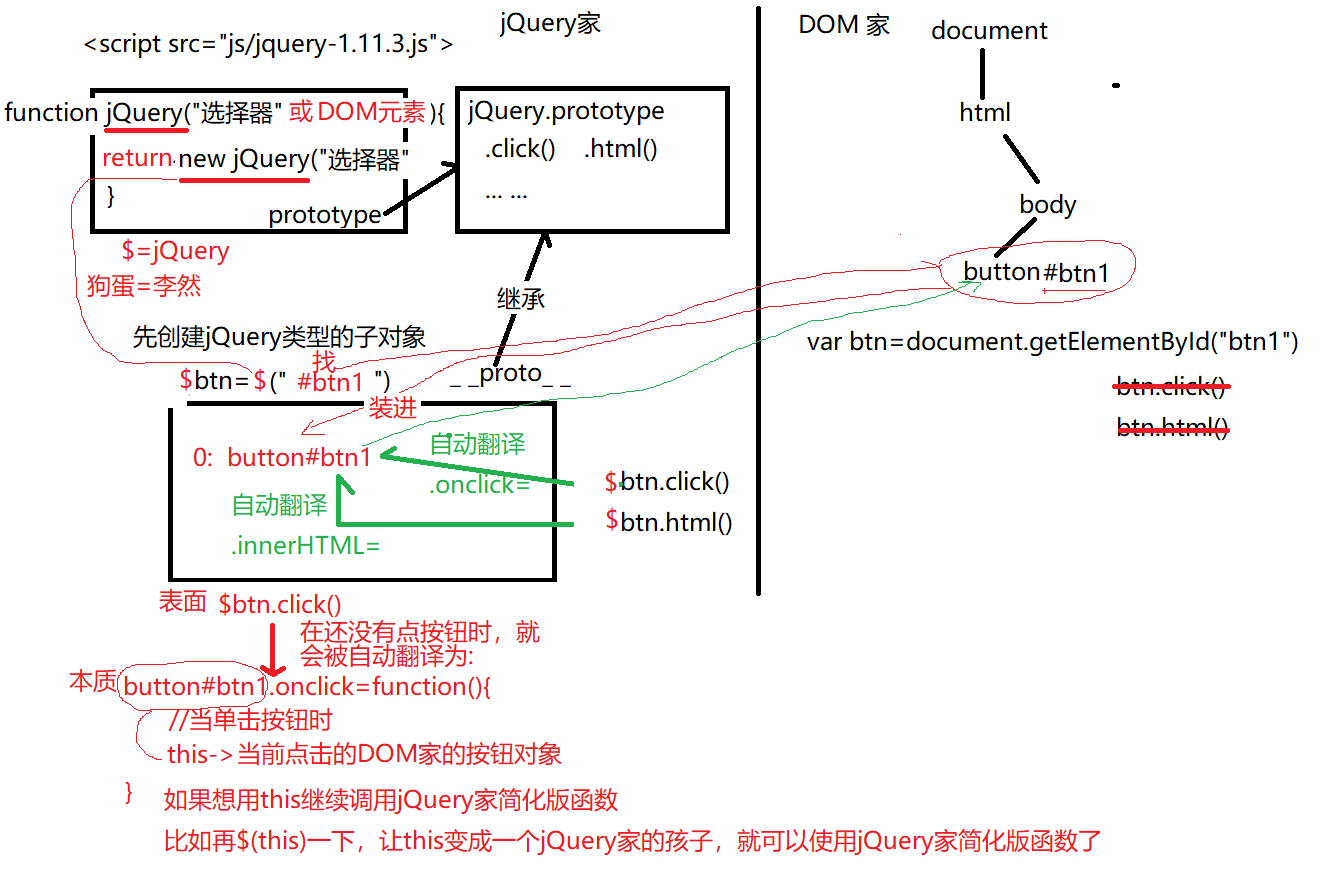
}

所以: var $btn=$("#btn1")

var $btn=$(this) this->点击的DOM按钮对象

16. 问题: 为什么必须把$(this)包裹以下用简化版函数？难道就不能this直接调用原生DOM函数吗？

17. 结论: 今后使用任何框架，最忌讳混搭！一定要专一！不同框架原理各不相同，返回值，参数，用法各不相同！如果混用！很容易懵逼！

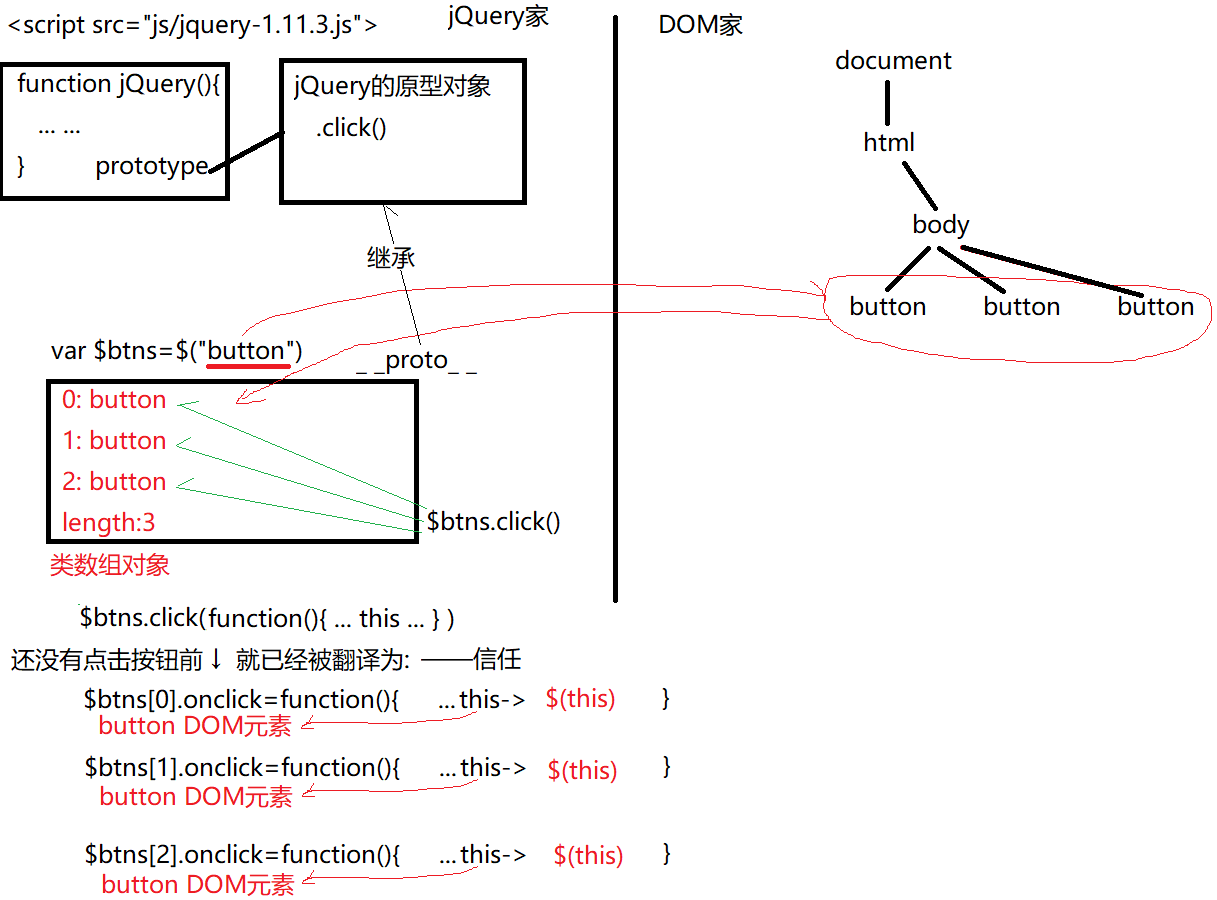


18. jQuery对象本质: 类数组对象，其中保存着所有找到的DOM元素对象

19. jQuery简化版函数三大特点:

(1). 自带for循环遍历:

a. 对整个jQuery对象调用一次简化版函数，等效于对其中每一个DOM元素对象分别调用一次简化版函数！



(2). 一个函数两用:

a. 凡是和修改(内容、属性、样式)有关的函数，都是一个函数两用

b. 一个函数，既能获取值，又能修改值

c. 通过重载方式判断本次该获取，还是该修改:

1). 如果调用函数时，没有传入新值，则函数默认执行获取旧值的操作

2). 如果调用函数时，传入了新值，则函数改为执行修改值的操作！

20. 示例: 多个按钮，点哪个按钮，就记录自己的点击次数

2\_jQueryAPI.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">  <meta charset="UTF-8">  <title></title>  </head>  <body>  <h1>jQueryAPI特点</h1>  <button id="btn1">click me(0)</button>  <button id="btn2">click me(0)</button>  <button id="btn3">click me(0)</button>  <script src="js/jquery-1.11.3.js"></script>  <script>  //为三个按钮绑定单击事件，记录单击次数  //DOM 4步  //1. 查找触发事件的元素  //本例中: 三个按钮都可以单击  var $btns = $("button");  console.log($btns);  //2. 绑定事件处理函数  //本例中: 每个按钮都能点，但是只需要对整个jQuery对象调用一次简化版函数即可  $btns.click(function () {  //alert("疼！");  //希望点那个按钮，让哪个按钮变成画  //this.innerHTML="❀";  //$(this).html("❀")  //3. 查找要修改的元素  //本例中: 单击按钮时，就是要修改自己: this  var $this = $(this);  //4. 修改元素  //4.1 先\*取出\*当前按钮的内容中()中的数量，转为整数  var n = parseInt(  $this.html() //()中没有传入新内容,是空的，所以html()本次执行获取  // "click me(10000)"  // 012345678901234  // | |  .slice( 9, -1) //复习第一阶段slice函数用法  // "10000"  );  //4.2 将数量+1  n++;  //4.3 将新数量拼到()中，\*修改\*回当前按钮的内容中  $this.html(`click me(${n})`);  //()中传入了新内容,不是空的，所以html(xxx)本次执行修改内容的操作  })  //button#btn1.onclick=function(){ this->button#btn1 }  //button#btn2.onclick=function(){ this->button#btn2 }  //button#btn3.onclick=function(){ this->button#btn3 }  </script>  </body>  </html>  运行结果: |

四. 查找元素:

1. jQuery支持所有CSS选择器: $("任意css选择器")

2. jQuery还新增了jQuery专属的个别选择器——仅jQuery可用！css不可用！

3. 基本过滤选择器:

(1). 回顾: CSS中的子元素过滤选择器:

a. 什么是: 按元素在其所在的父元素下的相对位置选择某个子元素

b. 包括: :first-child :last-child :nth-child(i) :only-child

c. 示例: 使用子元素过滤，选择指定元素

5\_child filter.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8" />  <title></title>  </head>  <body>  <h3>子元素过滤选择器.</h3>  <ul>  <li>child1-basic0</li>  <li>child2-basic1</li>  <li>child3-basic2</li>  </ul>  <ul>  <li>child1-basic3</li>  <li>child2-basic4</li>  <li>child3-basic5</li>  </ul>  <ul>  <li>child1-basic6</li>  </ul>  <script src="js/jquery-1.11.3.js"></script>  <script>  //查找每个ul中第一个li，让其边框变为红色  $("ul>li:first-child").css("border","1px solid red");  //查找每个ul中最后一个li，给其加绿色阴影  $("ul>li:last-child").css("box-shadow","0 0 5px green");  //查找每个ul中处于偶数位置的li,让其字体变为红色  $("ul>li:nth-child(2n)").css("color","red");  //查找每个ul中第2个li,修改其背景为黄色  $("ul>li:nth-child(2)").css("background-color","yellow")  //查找作为ul下唯一子元素的li  $("ul>li:only-child").css("padding","15px")  </script>  </body>  </html>  运行结果: |

(2). 基本过滤选择器:

对应小程序视频: 小程序->在线->JQUERY->day01 3. 子元素过滤child filter 。。。

a. 什么是: 先将所有符合条件的元素提取到一个大的集合中统一编号，然后按大排名的顺序，选择元素！——下标从0开始！

b. 包括:

1). :first 整个查询结果中第一个元素

2). :last 整个查询结果中最后一个元素

3). :even 整个查询结果中偶数下标位置的元素

偶数 4个字母

4). :odd 整个查询结果中奇数位置的元素

奇数 3个字母

5). :eq(i) 整个查询结果中下标为i位置的元素

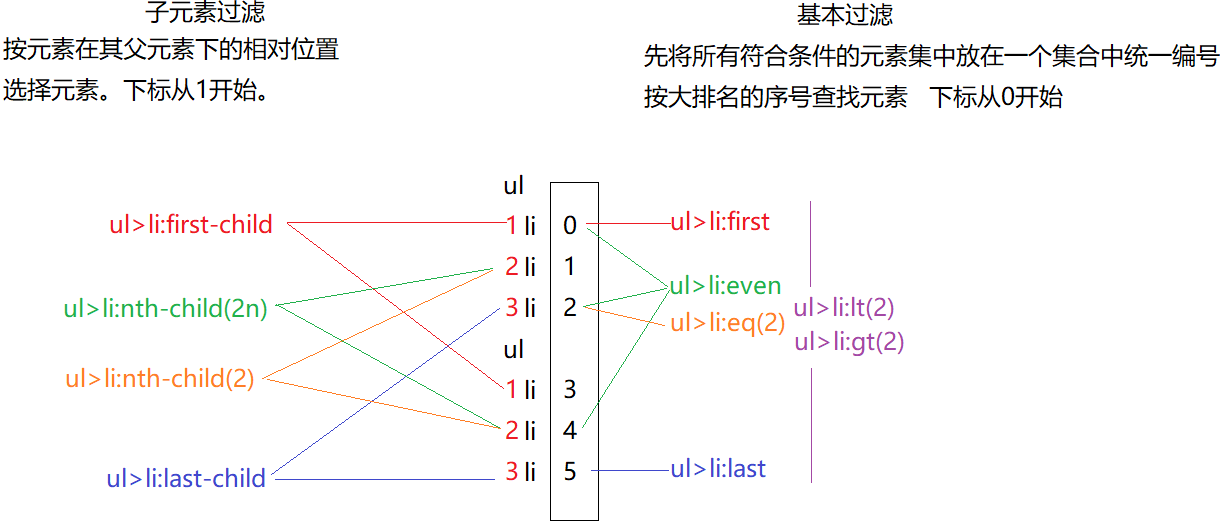
equal 等于

6). :gt(i) 整个查询结果中下标>i位置的元素

greater than > 大于

7). :lt(i) 整个查询结果中下标<i位置的元素

less than < 小于



c. 示例: 使用基本过滤，选择指定元素

6\_basic filer.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8" />  <title></title>  </head>  <body>  <h3>基本过滤选择器.</h3>  <ul>  <li>child1-basic0</li>  <li>child2-basic1</li>  <li>child3-basic2</li>  </ul>  <ul>  <li>child1-basic3</li>  <li>child2-basic4</li>  <li>child3-basic5</li>  </ul>  <ul>  <li>child1-basic6</li>  </ul>  <script src="js/jquery-1.11.3.js"></script>  <script>  //查找所有li中第一个li  $("ul>li:first").css("border","1px solid red")  //查找所有li中最后一个li  $("ul>li:last").css("box-shadow","0 0 5px green")  //查找处于正常人眼里的偶数位置的li  $("ul>li:odd").css("background-color","yellow");  //查找处于下标为偶数位置的li  $("ul>li:even").css("color","red")  //查找正常人眼里第2个li  $("ul>li:eq(1)").css("font-weight","bold")  //查找下标为2位置的li  $("ul>li:eq(2)").css("text-decoration","underline")  </script>  </body>  </html>  运行效果: |

(3). 示例: 按钮组

对应小程序视频: 小程序->在线->JQUERY->day01 作业:分别使用css和jquery实现按钮组效果

7\_basic filter2.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8" />  <title>...</title>  <style>  /\* 清除ul圆点，并所有li浮动 \*/  ul{  list-style:none  }  ul>li{  float:left;  padding: 5px 10px;  border:1px solid #555  }  /\* 纯css实现 \*/  /\* 设置第一个li的左上和左下圆角 \*/  /\* ul>li:first-child{  border-radius:5px 0 0 5px  } \*/  /\* 设置最后一个li的右上和右下圆角 \*/  /\* ul>li:last-child{  border-radius:0 5px 5px 0  } \*/  /\* 清除第一个li之后所有li兄弟的左边框 \*/  /\* ul>li~li{  border-left:0;  } \*/  </style>  </head>  <body>  <h1>实现按钮组效果</h1>  <ul>  <li>1</li>  <li>2</li>  <li>3</li>  <li>4</li>  </ul>  <script src="js/jquery-1.11.3.js"></script>  <script>  /\* 设置第一个li的左上和左下圆角 \*/  $("ul>li:first").css("border-radius","5px 0 0 5px")  /\* 设置最后一个li的右上和右下圆角 \*/  $("ul>li:last").css("border-radius","0 5px 5px 0")  /\* 清除第一个li之后所有li兄弟的左边框 \*/  $("ul>li:gt(0)").css("border-left",0)  </script>  <div style="clear:both"></div>  <h1>实现表格隔行变色效果</h1>  </body>  </html>  运行效果： |

|  |
| --- |
| 强调: 如果将来一个效果即可用css做，又可以用js做！  1. 一定首选CSS:  (1). 效率高！  css是归计算机绘图引擎解析，因为绘图引擎只负责绘图，所以效率高！  js是归计算机程序引擎解析，因为计算机不止运行js程序！还同时运行着其他很多程序。js不得不需要跟其他程序共用计算机的资源——效率低  (2). 便于迁移和维护：  今后框架中，对js的修改极大！现在的js，DOM，jQuery，到了框架中几乎全都不能用！都需要按框架要求重新写！但是只有css，框架不管！移植时，只要将css拷贝到框架中，无需修改照样正常运行！  2. 当需要人为操作交互效果时，就只能用js做。 |

尝鲜: 修改css属性:   
1. DOM中: 元素.style.css属性="属性值"

2. jq中: $(...).css("css属性","属性值")

总结: DOM操作万能4步:

1. 查找触发事件的元素

2. 为找到的元素绑定事件处理函数

3. 查找要修改的元素

4. 修改元素

总结: jQuery 不用背！！！用的时候来找到即可！

1. 查找元素: 4种:

(1). 按选择器查找:

var $jquery对象=$("选择器")

(2). 不需要查找就可直接获得的元素对象: 1个

a. $(document) 根节点对象

(3). 按节点间关系查找: 如果已经获得一个元素，找周围附近的元素时，2大类关系，6个属性

a. 父子关系: 4个属性:

1). 元素.parentElement  
 2). 元素.children

3). 元素.firstElementChild

4). 元素.lastElementChild

b. 兄弟关系: 2个属性

1). 元素.previousElementSibling

2). 元素.nextElementSibling

|  |
| --- |
| 总结: 查找方法的返回值:  1. 如果原函数返回下标位置i，则如果找不到，返回-1  2. 如果原函数返回一个对象或数组，则如果找不到，返回null  3. 如果原函数返回一个类数组对象，则如果找不到，返回空类数组对象{ length:0 } |

2. 修改: 3种

(1). 内容: 3种

a. 绝大多数情况下，用.innerHTML就够了！

b. 除非在获取内容时，确实希望去掉内嵌标签，只要文本时，才用textContent

c. 获取或修改表单元素的值: 表单元素.value

(2). 属性: 3种:

a. 字符串类型的HTML标准属性: 2种:

1). 旧核心DOM 4个函数

i. 元素.getAttribute("属性名")

ii. 元素.setAttribute("属性名","属性值")

iii. 元素.hasAttrbiute("属性名")

iv. 元素.removeAttribute("属性名")

2). 新HTML DOM 简写:

i. 元素.属性名

ii. 元素.属性名="属性值"

iii. 元素.属性名!==""

iv. 元素.属性名=""

b. bool类型的HTML标准属性: 只能用 "元素.属性名" 操作，~~不能用核心DOM4个函数~~

c. 自定义扩展属性: HTML5标准

1). 添加/修改自定义扩展属性:

i. HTML中: <元素 data-自定义属性名="属性值"

ii. 旧js中: ~~不能用.,~~ 只能用 元素.setAttribute("data-自定义属性名","属性值")

iii. HTML5标准: 元素.dataset.自定义属性名=属性值

2). 获取自定义扩展属性值:

i. 旧js中: ~~不能用.,~~ 只能用 元素.getAttribute("data-自定义属性名")

ii. HTML5标准: 元素.dataset.自定义属性名

3). 查找带有自定义扩展属性的元素: 属性选择器 [data-自定义属性=属性值]

(3). 样式:

a. 只修改一个css属性时: 元素.style.css属性="属性值"

b. 批量修改多个css属性时: 元素.className="class名"

c. 获取css属性值时: var style=getComputedStyle(元素对象)

style.css属性名

3. 添加元素:

(1). 只添加一个元素: 3步

a. 创建一个空元素

var 一个元素=document.createElement("元素名")

b. 为元素设置关键属性

新元素.属性名=属性值

c. 将新元素添加到指定的父元素下: 3种:

1). 末尾追加: 父元素.appendChild(新元素)

2). 插入到一个元素前: 父元素.insertBefore(新元素, 现有元素)

3). 替换现有元素: 父元素.replaceChild(新元素, 现有元素)

(2). 优化: 如果添加多个元素，尽量减少操作DOM树的次数

a. 如果同时添加父元素和子元素，应该先将子元素添加到父元素，最后再一次性将父元素添加到DOM树

b. 如果父元素已经在页面上，需要同时添加多个平级子元素，应该用文档片段: 3步

1). 先创建文档片段对象

var frag=document.createDocumentFragment();

2). 先将多个平级子元素加入到文档片段中

frag.appendChild(子元素)

3). 最后将文档片段添加到DOM树

父元素.appendChild(frag)

4. 删除元素: 父元素.removeChild(要删除的元素)

5. 事件:

(1). 事件绑定: 3种:

a. <元素 on事件名="事件处理函数()">

function 事件处理函数(){ ... }

b. 一个元素的一个事件属性上，只绑定一个事件处理函数，不会动态增加或减少:

元素.on事件名=function(){ ... }

c. 一个元素的一个事件上，需要同时绑定多个事件处理函数，还可能动态添加或移除

元素.addEventListener("事件名", 事件处理函数)

元素.removeEventListener("事件名", 原事件处理函数)

(2). 事件模型: 3个阶段: 捕获、目标触发、冒泡

(3). 获得事件对象: 定义事件绑定时: 元素.on事件名=function(e){ //触发事件时e=event }

(4). 取消冒泡/停止蔓延: e.stopPropagation()

(5). 只要多个平级子元素都要绑定相同的事件时: 利用冒泡/事件委托: 3步

a. 事件绑定在父元素上一份

b. 处理函数中，用e.target代替this获得实际点击的那个子元素

c. 处理函数中，先验证e.target的特征是想要的元素，才继续执行后续操作。

(6). 阻止元素自带的默认行为: e.preventDefault();

(7). 获得鼠标位置：

a. 鼠标位置相对屏幕左上角的距离: e.screenX e.screenY

b. 鼠标位置相对浏览器文档显示区左上角的距离 e.clientX e.clientY

c. 鼠标位置相对事件所在元素左上角的距离e.offsetX e.offsetY

(8). 窗口滚动事件:

a. 当滚动条滚动时自动触发: window.onscroll=function(){ ... }

b. 获取滚动条滚动过的距离:

var scrollTop=document.body.scrollTop||document.documentElement.scrollTop;

c. 控制滚动条滚动到指定位置: window.scrollTo(0, 要滚动到的位置)

总结: this 6种:

1. obj.fun() this->obj

2. new Fun() this->new正在创建的子对象

3. 类型名.prototype.共有方法=function(){ ... } this->将来调用这个共有方法的.前的子对象

4. fun() 和 (function(){ })() 和回调函数中的this->window

5. 事件处理函数中的this->当前正在触发事件的这个元素对象

6. $jquery对象.click(function(){ this->将来点击的那个DOM元素对象})

今日对应小程序视频:

小程序->在线->JQUERY->day01 1. 什么是jQuery，如何使用jQuery，CDN

2. 第一个jQuery程序 jQuery原理...

3. 子元素过滤child filter 基本过滤

作业: 分别使用css和jquery实现按钮组效果。。。

作业:

1. 复习今日小程序问题清单(小程序->在线->JQUERY->day01)，预习明日小程序问题清单(小程序->在线->jQuery->day02)

2. 看小程序视频分别使用css和jquery实现按钮组效果和表格隔行变色效果

小程序->在线->JQUERY->day01 作业: 分别使用css和jquery实现按钮组效果和表格隔行变色效果

3.（凡是觉得知识点串不起来）看小程序视频学习DOM版学子商城项目(持续到下周一之前)

小程序->在线->DOM->day04 扩展: DOM版学子商城...

家里环境xampp和nodejs能用，且第一阶段学的好的同学:

从小程序中网盘地址中下载xzserver\_start.zip，看视频中day01和day02，跟着做

家里环境xampp和nodejs不能用，或者第一阶段学的不好的同学：

从小程序中网盘地址中只下载public.zip，看视频中day01和day02，跟着做

然后，所有网页用live server运行

视频中所有ajax请求的服务器端接口地址，都改为我的新浪云服务器地址，就不需要你再运行服务器端了。也不需要你了解任何服务器端知识了:

/index 改为 http://xzserver.applinzi.com/index

/details 改为 http://xzserver.applinzi.com/details?lid=xxx