小程序: WEB问题速查

1. 问题清单: 小程序->首页->JQUERY->day02

2. 高清无码小视频: 小程序->在线->JQUERY->day02

今日扩展视频:

6.1 jQuery tabs和滑动过渡的tabs

9. jquery 事件委托 优化标签页、删除方块...

作业: jquery单击下拉菜单dropdown

作业: jQuery带多种动画效果的对话框...

在day02/5\_class2.html中直接做

不用今天都看完！！！！！！\*\*\*两个月后\*\*\*鄙视面试前\*\*\*看完就行\*\*\*！

3. 单词列表: 小程序->单词->JQUERY->day02

第三阶段上课约定:

1. 上课千万不要跟着记笔记和写代码！

2. 拼写错误！自己找！让我帮你找，你需要手抄20遍，给我拍照！

3. 现在正处在照猫画虎的阶段！每个DOM效果至少写3遍:

3.1 注释+代码抄一遍！

3.2 保留注释，删除代码，把代码再填回来！

3.3 删除注释和代码，试着用自己的话把注释写回来，根据自己的注释，把代码填回来

一个DOM效果，只有自己能写出人话注释，才算自己会！——标准！

如何提问:

1. 上课时，案例中，笔记中那句话哪行代码看不懂，立刻微信私信问我！

2. 小程序中问题清单里哪个问题和回答看不懂！立刻微信私信问我！

3. 个人项目中哪个功能不会做！立刻微信私信问我！

4. 刷笔试题时，遇到不会的笔试题！立刻微信私信问我！

下载源代码: 右键->链接另存为！

正课:

1. 修改

2. 按节点间关系查找

3. 添加/删除/替换/克隆

4. 事件绑定:

一. 修改: 3种东西，如果忘了，可回看DOM第二天的视频

凡是和修改相关的函数，都一个函数两用！

1. 内容: 3种: .html() .text() .val()

2. 属性: 3类

对应小程序视频: 小程序->在线->JQUERY->day02 2. jQuery修改...

(1). 字符串类型的HTML标准属性:

a. DOM中: 2种:

1). 旧核心DOM: 4个函数

i. 获取属性值: 元素.getAttribute("属性名")

ii. 修改属性值: 元素.setAttribute("属性名","属性值")

iii. 判断是否包含: 元素.hasAttribute("属性名")

iv. 移除属性: 元素.removeAttribute("属性名")

2). 新HTML DOM: 元素.属性名

property

b. jq中: 2种:

1). 代替旧核心DOM: $元素.attr("属性名", "属性值")

2). 代替新HTML DOM: $元素.prop("属性名","属性值")

(2). bool类型的HTML标准属性:

a. DOM中，只能用: 元素.属性名

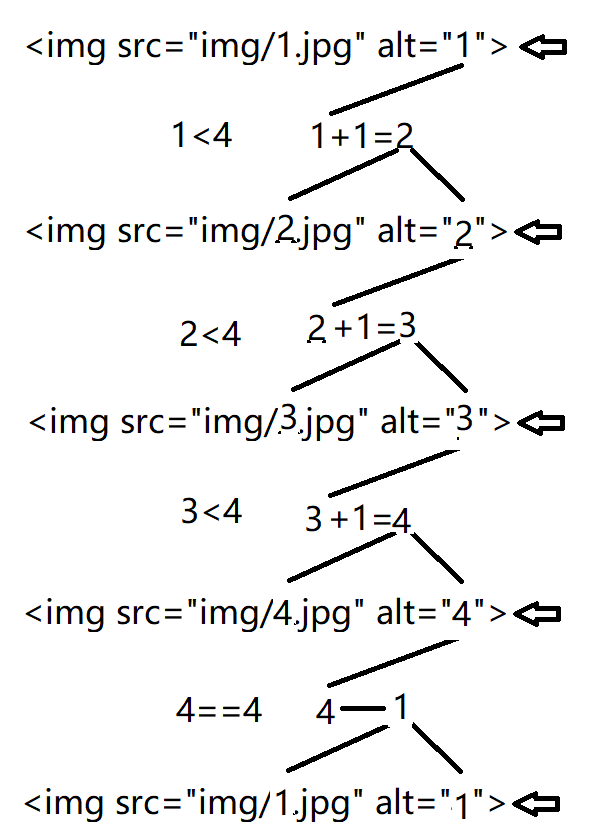
b. jq中，也只能用: $元素.prop("属性名", bool值)

(3). 自定义扩展属性:

a. DOM中，只能用旧核心DOM4个函数访问

b.jq中，也只能用$元素.attr("属性名","属性值")

(4). 示例: 点击图片切换下一张:



2\_attr.html

|  |
| --- |
| <!DOCTYPE html>  <html>   <head>    <title> new document </title>    <meta charset="utf-8">   </head>   <body>    <h1>操作元素的属性</h1>    <img src="img/1.jpg" alt="1">    <script src="js/jquery-1.11.3.js"></script>    <script>      //单击图片，切换下一张      //DOM 4步      //1. 查找触发事件的元素      //本例中: 单击img触发变化      var $img=$("img")      //2. 绑定事件处理函数      $img.click(function(){        //3. 查找要修改的元素        //本例中:要改的就是自己        var $this=$(this);        //4. 修改元素        //4.1 先获得现在正在看第几张图片        //本例中: 利用img元素的alt属性临时存储第几张图片的序号，所以获得alt属性值        //因为alt属性是字符串类型的HTML标准属性，所以即可用.attr()又可用.prop()        var i=parseInt($this.attr("alt"))        //4.2 如果i<4，就i++，否则i=1        if(i<4){          i++;        }else{          i=1;        }        //4.3 即要将i拼接成新的图片路径保存到当前图片的src属性中，又要修改当前图片的alt属性值为新的i。        //因为src属性也是字符串类型的HTML标准属性，所以即可用.attr()又可用.prop()        $this.attr({          src:`img/${i}.jpg`,          alt:i        })      })    </script>   </body>  </html>  运行结果: |

(5). 示例:点小图片，切换大图片

3\_attr2.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">    <meta charset="UTF-8">    <title></title>    <style>      body {        text-align: center;      }    </style>  </head>  <body>  <img src="img/1.jpg" data-target="img/1-l.jpg" class="my-small">  <img src="img/2.jpg" data-target="img/2-l.jpg" class="my-small">  <img src="img/3.jpg" data-target="img/3-l.jpg" class="my-small">  <img src="img/4.jpg" data-target="img/4-l.jpg" class="my-small">  <hr/>  <img src="img/1-l.jpg" class="my-big">  <script src="js/jquery-1.11.3.js"></script>  <script>    //点击小图片，下方my-big中显示大图片    //DOM 4步    //1. 查找触发事件的元素    //本例中:四个class为my-small的img元素都可点击    var $imgs=$(".my-small");    //2. 绑定事件处理函数函数    $imgs.click(function(){      //3. 查找要修改的元素      //本例中: 无论点哪个小图片，都要修改下方的大图片      var $big=$(".my-big");      //4. 修改元素      //4.1 获得当前点击的小图片上的data-target属性值中保存的对应的大图片路径      //因为data-target是自定义扩展属性，所以只能用.attr()，不能用.prop()      var target=$(this).attr("data-target");      //4.2 将获得的大图片路径设置为大图片的src属性值      //因为src属性值字符串类型的HTML标准属性，所以即能用.attr()，又能用.prop()      $big.attr("src",target);    })  </script>  </body>  </html>  运行结果: |

3. 样式:

对应小程序视频: 小程序->在线->JQUERY->day02 3. jquery修改...

(1). DOM中:

a. 修改内联样式: 元素.style.css属性名="属性值"

b. 获取样式: getComputedStyle(元素对象)

(2). jq中: 统一了！无论获取还是修改样式，之用.css()就够了:

a. $元素.css("css属性名","属性值")

b. 原理:

1). 如果调用.css()函数时，没给新属性值，则.css()默认执行获取属性值的操作。此时，.css()函数内自动执行的就是DOM中的getComputedStyle()，获得的就是计算后的样式！

2). 如果调用.css()函数时，给了新属性值，则.css()自动切换为修改属性值的操作。此时，.css()函数内自动执行的就是DOM中.style.css属性=新值，修改的就是内联样式。

(3). 批量修改一个元素的多个css属性:

a. DOM中: 元素.className="class名"

b. 问题: className="class名"是整体替换元素的class属性值！如果原来的元素被多个class名同时修饰，则很难只修改其中某一个class。

c. jq中: 提供了一套专门单独操作某一个class名的函数

1). 为元素添加一个新class:

$元素.addClass("class名")

2). 从元素上移除一个class:

$元素.removeClass("class名")

3). 判断一个元素是否包含某个class

$元素.hasClass("class名")

4). 在有和没有一个class之间来回切换

$元素.toggleClass("class名")

切换

内部封装的其实就是一个判断:

if($元素.hasClass("class名")){

$元素.removeClass("class名")

}else{

$元素.addClass("class名")

}

d. 示例: 使用class批量修改一个元素的多个css属性

4\_class.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">    <meta charset="UTF-8">    <title></title>    <style>      .btn {        padding: 5px 10px;        border-radius: 3px;        border: 1px solid #aaa;        outline: none;      }      /\*准备了两个class\*/      /\*按钮抬起时的样式\*/      .up {        background: #fff;        color: #333;      }      /\*按钮按下时的样式\*/      .down {        background: #ddd;        color: #fff;      }    </style>  </head>  <body>    <!--开局, 按钮默认是抬起的，所以已经提前放好了up样式类-->    <!--当单击按钮时，只要添加down样式类，就可覆盖up样式类。再次单击按钮，只要移除down样式类，就可恢复成up样式类-->    <button class="btn up">双态按钮</button>  <script src="js/jquery-1.11.3.js"></script>  <script>    //双态按钮: 让按钮的class在up和down之间切换    //DOM 4步    //1. 查找触发事件的元素    //本例中: 点按钮触发变化    var $btn=$("button");    //2. 绑定事件处理函数    $btn.click(function(){      //3. 查找要修改的元素      //本例中: 就要修改自己      // var $this=$(this)      //4. 修改元素      //如果当前按钮有down样式类      // if($this.hasClass("down")){      //   //就移除down      //   $this.removeClass("down");      // }else{//否则如果没有down样式类      //   //就添加down样式类      //   $this.addClass("down");      // }      //最简化写法:      $(this).toggleClass("down");            //自动在有down和没有down之间切换    })  </script>  </body>  </html>  运行结果: |

|  |
| --- |
| 简写: 如果同时修改一个元素的多个属性或css属性时都可简写为对象语法:  $元素.attr或prop或css({  属性名 : 属性值,  ... : ... })  如果css属性名中有-，两种选择: 要么加""，要么去横线变驼峰 |

二. 按节点间关系查找:

对应小程序视频: 小程序->在线->JQUERY->day02 6. jquery 按节点间关系查找

1. DOM中: 2大类关系，6个属性:

(1). 父子关系

a. 元素.parentElement

b. 元素.children

c. 元素.firstElementChild

d. 元素.lastElementChild

(2). 兄弟关系

a. 元素.previousElementSibling

b. 元素.nextElementSibling

2. jq中: 2大类关系，8个函数

(1). 父子关系: 3个函数

a. 获得一个元素的父元素: $元素.parent()

b. 获得一个元素的所有直接子元素:

1). $元素.children("选择器")

2). 增强版:

i. 加选择器，就只获得符合条件的子元素

ii. 不加选择器，默认获得所有直接子元素

3). 因为children可以加筛选条件了，所以就不再有专门的firstElementChild和lastElementChild

i. 获得元素下第一个直接子元素

$元素.children(":first-child")

ii. 获得元素下最后一个直接子元素

$元素.children(":last-child")

4). 问题: children只能在直接子元素中查找符合条件的元素。无法在所有后代中查找！

c. 新增了: 在当前元素所有后代中查找符合条件的元素。

$元素.find("选择器")

(2). 兄弟关系: 5个函数

a. 前一个兄弟: $元素.prev()

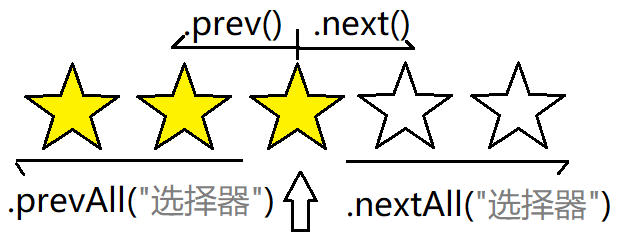
只能找前一个元素

b. 后一个兄弟: $元素.next()

只能找后一个元素

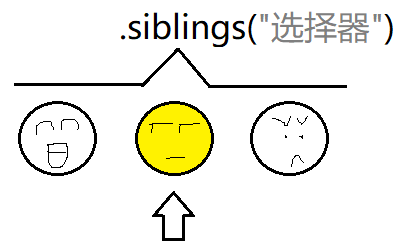
c. 之前所有兄弟: $元素.prevAll("选择器")

d. 之后所有兄弟: $元素.nextAll("选择器")



e. 选择除当前元素之外，其余所有兄弟:

$元素.siblings("选择器")



3. 示例: 使用节点间关系查找，选择指定的元素：

6\_traverse.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>    <meta charset="utf-8" />    <title>...</title>    <script>    </script>  </head>  <body>    <!--    ul.top>(li.parent>ul>li.child\*3)\*2 -->    <ul class="top">      <li class="parent1">parent1        <ul>          <li class="child">child1</li>          <li class="child">child2</li>          <li class="child">child3</li>        </ul>      </li>      <li class="parent2">parent2        <ul>          <li class="child">child1</li>          <li class="child">child2</li>          <li class="child">child3</li>        </ul>      </li>    </ul>    <script src="js/jquery-1.11.3.js"></script>    <script>      //修改class为top的ul的所有直接子元素      $("ul.top").children()      .css("border","1px solid red");      //修改class为top的ul的所有后代li      $("ul.top").find("li")      .css("box-shadow","0 0 5px green")      //为class为child的li绑定单击事件      $("ul.top li.child").click(function(){        //选择当前元素的下一个元素/前一个元素/之前所有/之后所有/除自己之外所有        $(this)        .siblings()//.nextAll()//.prevAll()//.prev()//.next()        .css("background-color","yellow")      })    </script>  </body>  </html>  运行结果: |

4. 示例: 标签页

对应小程序视频: 小程序->在线->JQUERY->day02 6.1 jquery tabs ...

7\_tabs.html

|  |
| --- |
| <!doctype html>  <html lang="en">   <head>    <meta charset="UTF-8">    <style>      .tabs{ list-style:none; padding:0 }      .tabs a{        text-decoration:none;        color:#000;        padding:6px 12px;        display:inline-block;      }      .tabs>li{        float:left;        border-bottom:1px solid #000;      }      .tabs>.active{        border:1px solid #000;        border-bottom:0;      }    </style>   </head>   <body>      <h1>使用属性选择器实现标签页头的切换</h1>      <ul class="tabs">        <li class="active">          <a data-toggle="tab" href="#">十元套餐</a>        </li>        <li>          <a data-toggle="tab" href="#">二十元套餐</a>        </li>        <li>          <a data-toggle="tab" href="#">三十元套餐</a>        </li>      </ul>      <script src="js/jquery-1.11.3.js"></script>      <script>        //DOM 4步        //1. 查找触发事件的元素        //本例中: 查找data-toggle属性为tab的元素        //      复习第二阶段属性选择器        var $as=$("[data-toggle=tab]")        //2. 绑定事件处理函数        $as.click(function(){          //3. 查找要修改的元素          //4. 修改元素          //给当前a元素的爹li加class active          $(this).parent() //当前a的爹li          .addClass("active")          //return $(this).parent()          //再给当前元素的爹li的所有兄弟标签去掉class active          .siblings().removeClass("active");          //链式操作！        })      </script>   </body>  </html>  运行结果: |

|  |
| --- |
| 总结: jQuery简化版函数3大特点:  (1). 自带for循环:  (2). 一个函数两用:  (3). 绝大多数函数都会自动返回正在操作的.前的主语jq对象。如果后一句话所需的.前的主语，刚好和前一句话返回的主语相同，则可以直接用.继续操作，不用把.前主语反复写2遍！——链式操作 |

三. 添加/删除/替换/克隆

对应小程序视频:小程序->在线->JQUERY->day02 7. jquery添加删除

1. 添加:

(1). DOM中:

a. 添加一个元素: 3步

1). 创建一个新的空元素:

var 新元素=document.createElement("标签名")

2). 设置关键属性:

3). 将新元素添加到DOM树: 3种:

i. 末尾追加:

父元素.appendChild(新元素)

ii. 插入到现有元素之前:

父元素.insertBefore(新元素, 现有元素)

iii. 替换现有元素:

父元素.replaceChild(新元素, 现有元素)

b. 批量添加多个平级子元素: 文档片段 3步：

1). 创建文档片段:

var frag=document.createDocumentFragment();

2). 将子元素临时添加到文档片段中

frag.appendChild(子元素)

3). 将文档片段对象添加到DOM树

父元素.appendChild(frag)

(2). jq中: 2步

a. 无论创建一个元素还是创建多个元素都只用$()就够了！

var $新元素=$(`HTML片段`)

b. 只剩一件事: 将新元素添加到DOM树，5种方式，10个函数

1). 末尾追加: $父元素.append($新元素)

return $父元素

可继续对父元素执行操作

$新元素.appendTo($父元素)

return $新元素

可继续对新元素执行操作

2). 开头插入: $父元素.prepend($新元素)

return $父元素

可继续对父元素执行操作

$新元素.prependTo($父元素)

return $新元素

可继续对新元素执行操作

3). 插入到现有元素之前: $现有元素.before($新元素)

return $现有元素

可继续对现有元素执行操作

$新元素.insertBefore($现有元素)

return $新元素

可继续对新元素执行操作

4). 插入到现有元素之后: $现有元素.after($新元素)

return $现有元素

可继续对现有元素执行操作

$新元素.insertAfter($现有元素)

return $新元素

可继续对新元素执行操作

5). 替换现有元素: ~~$现有元素~~.replaceWith($新元素)

return $现有元素

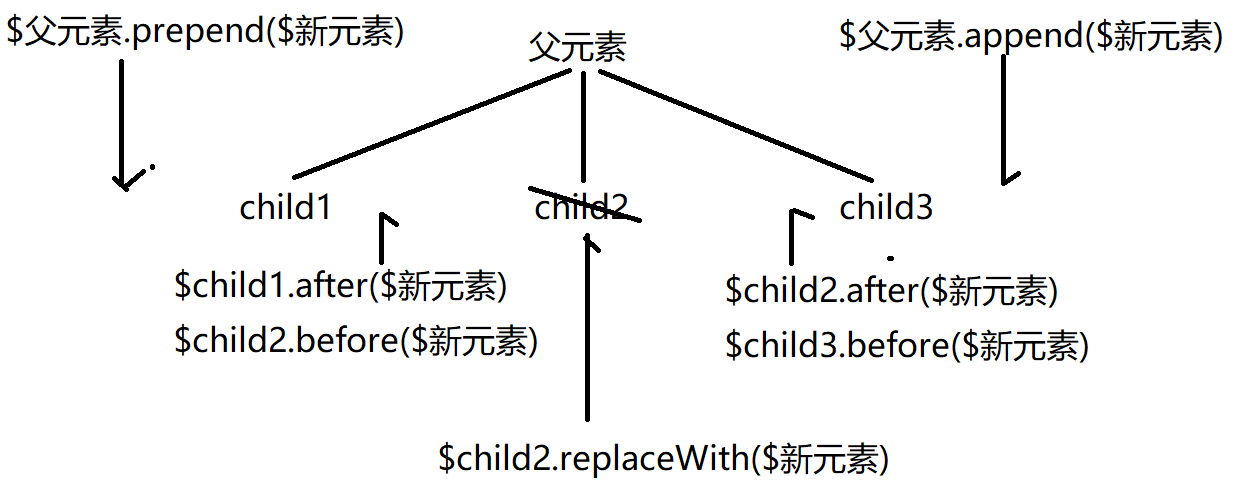
可继续对现有元素执行操作

$新元素.replaceAll($现有元素)

return $新元素

可继续对新元素执行操作

c. jquery这十个函数在将新元素添加到DOM树上时，只要遇到同时添加多个新元素的情况，在其内部，会自动使用文旦片段整体添加！——从此不用我们自己再写文档片段！



2. 删除: $元素.remove()

3. 示例: 添加删除方块  
9\_append.html —— 未使用链式操作优化

|  |
| --- |
| <!DOCTYPE html>  <html>   <head>    <title> new document </title>    <meta charset="utf-8">    <style>      .container {        border: 1px solid #aaa;  overflow:hidden;      }      .block {        float: left; margin: 10px;        border: 1px solid #aaa;        background: #faa;        width: 150px; height: 150px;      }      .block:hover{ box-shadow:0 5px 6px #000; }      .close {        float:right; padding:5px;        font-weight:bold; opacity:.2; cursor: pointer;      }      .close:hover { opacity: .5; }    </style>   </head>   <body>    <h1>添加/删除节点</h1>    <button id="add-block">添加区块</button>      <div class="container">      <!-- <div class="block">        <span class="close">×</span>      </div> -->    </div>      <script src="js/jquery-1.11.3.js"></script>    <script>      //DOM 4步      //1. 查找触发事件的元素      //本例中: 点按钮添加新方块      $("#add-block")      //2. 绑定事件处理函数      .click(function(){        //3. 查找要修改的元素        //本例中: 点击按钮向下方class为container的div中添加新方块        // var $div=$(".container");        //4. 修改元素        //4.1 先创建新方块元素:        $(`<div class="block">          <span class="close">×</span>        </div>`)        //4.2 为新方块设置随机背景色        .css("background-color",`rgb(${          parseInt(Math.random()\*255)        },${          parseInt(Math.random()\*255)        },${          parseInt(Math.random()\*255)        })`)        //4.3 将新方块元素添加到div中        .prependTo(".container");        //自动查找      });      //DOM 4步      //1. 查找触发事件的元素      //本例中: 因为多个×都能点击，所以事件应该绑在公共的父元素div.container上一份——事件委托      $(".container")      //2. 绑定事件处理函数      .click(function(e){        //事件委托第二步: e.target代替this        var $tar=$(e.target);        //事件委托第三步: 判断        //只有点在class为close的span上才执行删除操作        if($tar.is(".close")){          //3. 查找要修改的元素          //本例中: 点×删除的是×的父元素          $tar.parent()          //4. 修改元素          //删除元素          .remove();        }      })    </script>     </body>  </html> |

9\_append2.html —— 使用链式操作优化后:

|  |
| --- |
| <!DOCTYPE html>  <html>   <head>    <title> new document </title>    <meta charset="utf-8">    <style>      .container {        border: 1px solid #aaa;  overflow:hidden;      }      .block {        float: left; margin: 10px;        border: 1px solid #aaa;        background: #faa;        width: 150px; height: 150px;      }      .block:hover{ box-shadow:0 5px 6px #000; }      .close {        float:right; padding:5px;        font-weight:bold; opacity:.2; cursor: pointer;      }      .close:hover { opacity: .5; }    </style>   </head>   <body>    <h1>添加/删除节点</h1>    <button id="add-block">添加区块</button>      <div class="container">      <!-- <div class="block">        <span class="close">×</span>      </div> -->    </div>      <script src="js/jquery-1.11.3.js"></script>    <script>      //DOM 4步      //1. 查找触发事件的元素      //本例中: 点按钮添加新方块      $("#add-block")      //2. 绑定事件处理函数      .click(function(){        //3. 查找要修改的元素        //本例中: 点击按钮向下方class为container的div中添加新方块        // var $div=$(".container");        //4. 修改元素        //4.1 先创建新方块元素:        $(`<div class="block">          <span class="close">×</span>        </div>`)        //4.2 为新方块设置随机背景色        .css("background-color",`rgb(${          parseInt(Math.random()\*255)        },${          parseInt(Math.random()\*255)        },${          parseInt(Math.random()\*255)        })`)        //4.3 将新方块元素添加到div中        .prependTo(".container");        //自动查找      });      //DOM 4步      //1. 查找触发事件的元素      //本例中: 因为多个×都能点击，所以事件应该绑在公共的父元素div.container上一份——事件委托      $(".container")      //2. 绑定事件处理函数      .click(function(e){        //事件委托第二步: e.target代替this        var $tar=$(e.target);        //事件委托第三步: 判断        //只有点在class为close的span上才执行删除操作        if($tar.is(".close")){          //3. 查找要修改的元素          //本例中: 点×删除的是×的父元素          $tar.parent()          //4. 修改元素          //删除元素          .remove();        }      })    </script>     </body>  </html>  运行结果: |

补:$元素.is("选择器") 判断当前元素是否符合选择器的要求

4. 克隆元素：var $新元素=$旧元素.clone();

5. 示例: 选飞机

10\_replace\_clone.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">    <meta charset="UTF-8">    <title></title>    <style>      body {        text-align: center;      }    </style>  </head>  <body>    <h1>替换节点</h1>    <div id="chosen">      <img src="img/p0.png">    </div>    <hr />    <div id="list">      <img src="img/p3.png">      <img src="img/p4.png">      <img src="img/p5.png">    </div>    <script src="js/jquery-1.11.3.js"></script>    <script>      //DOM 4步      //1. 查找触发事件的元素      //本例中: 因为三个img都可点击，所以事件只绑定在父元素上一份即可——事件委托      $("#list")      //2. 绑定事件处理函数      .click(function(e){        //事件委托第二步: e.target代替this        var $tar=$(e.target);        //事件委托第三步: 判断当前元素是不是img        if($tar.is("img")){          //3. 查找要修改的元素          //本例中: 点下方的img只要修改上方的img即可          //4. 修改元素          //先将当前选中的img克隆一个元素副本          $tar.clone()          //用副本元素代替上方的img          .replaceAll("#chosen>img");          //自动查找        }      })    </script>  </body>  </html>  运行结果: |

四. 事件绑定:

对应小程序视频: 小程序->在线->JQUERY->day02 8. jquery 事件 on off

1. DOM中: 3种:

(1). 在html中绑定:

a. js中: function 事件处理函数(){ ... }

b. html 中: <元素 on事件名="事件处理函数()">

(2). 在js中用赋值方式绑定:

a. 元素.on事件名=function(){ ... }

(3). 在js中用添加事件监听对象的方式: ——最灵活的

a. 添加事件监听: 元素.addEventListener("事件名",事件处理函数)

b. 移除事件监听: 元素.removeEventListener("事件名", 原事件处理函数)

2. jq中: 统一为:

(1). 标准:

$元素.on("事件名", 事件处理函数)

$元素.off("事件名", 原事件处理函数)

(2). 简写: (只有最常用的几种事件才能用.简写)

$元素.事件名(事件处理函数)

比如: $元素.click(function(){ ... })

常用事件列表: 只要属于这个列表的事件，都可以用简写。反之，万一遇到这个列表之外的稀有事件，就必须用.on()

|  |
| --- |
| blur 失去焦点  change 下拉列表选中项改变  click 单击  dblclick 双击  focus 获得焦点  keydown 键盘按键按下  keyup 键盘按键抬起  mousedown 鼠标按键按下  mouseenter 鼠标进入(jq)  mouseleave 鼠标移出(jq)  mousemove 鼠标移动  mouseout 鼠标移出(dom)  mouseover 鼠标进入(dom)  mouseup 鼠标按键抬起  resize 窗口大小改变  scroll 网页滚动 |

3. 改进:

(1). DOM中，如果使用有名称的函数添加事件监听，则只能添加1个。

(2). jq中，即使使用有名称的函数添加事件监听，照样可以添加多个！移除时，只需要一次off，就可把所有同名的事件处理函数移除掉！

4. 示例: 点击按钮获得跟踪导弹，点击按钮失去跟踪导弹

12\_bind\_unbind.html

|  |
| --- |
| <!DOCTYPE html>  <html>   <head>    <title> new document </title>    <meta charset="utf-8">   </head>   <body>    <h1>事件绑定</h1>    <button id="btn1">发射子弹</button>    <button id="btn2">获得奖励</button>    <button id="btn3">失去奖励</button>    <script src="js/jquery-1.11.3.js"></script>    <script>      $("#btn1").click(function(){        console.log(`发射普通子弹......`)      })      function shoot2(){        alert(`发射跟踪导弹=>=>=>`)      }      $("#btn2").click(function(){        //        .addEventListener();        $("#btn1").click(shoot2)      })      $("#btn3").click(function(){        $("#btn1").off("click",shoot2)      })    </script>   </body>  </html>  运行结果: |

5. 事件委托:

对应小程序视频: 小程序->在线->JQUERY->day02 9. jquery 事件委托

(1). DOM中: 3步 ——标准的，且将来仍会使用的！

a. 事件只绑定在父元素上一份即可

b. 用e.target代替this

c. 判断e.target是否是想要的

(2). jq中: （仅jq中可用）

a. 事件也只绑定在父元素上一份即可

b. this又重新指向了目标元素，不用e.target

c. 不用自己写if判断，而是给.on()添加第二个选择器参数，作为判断条件，请.on()函数自动帮我们绑定当前元素是不是想要的

$父元素.on("事件名","选择器条件",function(){

this->e.target

})

d. 示例: 使用事件委托简写选飞机案例:

12\_replace\_clone2.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">    <meta charset="UTF-8">    <title></title>    <style>      body {        text-align: center;      }    </style>  </head>  <body>    <h1>替换节点</h1>    <div id="chosen">      <img src="img/p0.png">    </div>    <hr />    <div id="list">      <img src="img/p3.png">      <img src="img/p4.png">      <img src="img/p5.png">    </div>    <script src="js/jquery-1.11.3.js"></script>    <script>      //DOM 4步      //1. 查找触发事件的元素      //本例中: 因为三个img都可点击，所以事件只绑定在父元素上一份即可——事件委托      $("#list")      //2. 绑定事件处理函数      //     只允许img元素触发事件      //            ↓      .on("click","img",function(){        //this->e.target        //获得当前点击的飞机img        $(this)        //将当前飞机img克隆一个副本        .clone()        //用副本元素代替上方的img        .replaceAll("#chosen>img");          //自动查找      })    </script>  </body>  </html>  运行结果: |

总结: $共有4种用法:

1. $("选择器") 查找DOM元素，并封装进jquery对象中

2. $(DOM元素对象) 无需查找，直接将一个DOM元素包装为jquery对象

3. $(`HTML片段`) 创建新元素！

4. (待续...)

总结: jQuery

1. 引入jQuery.js: 2种

(1). 引入项目本地jquery.js文件:

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

(2). 引入CDN网络中共享的jquery.js文件:

<script src="官方提供的CDN上jquery.js文件地址">  
2. 创建jQuery类型子对象: 2种  
(1). 只有jQuery类型子对象才能使用jQuery家简化版函数。

DOM家元素对象无权直接使用jQuery家简化版函数。

所以只要使用jQuery家简化版函数，都要先创建jQuery家子对象，其中保存一个要操作的DOM元素对象。  
(2). 如何:2种:

a. 查找DOM元素对象，并保存进新创建的jQuery对象中:

var $jq子对象=$("选择器")

b. 不查找，直接将DOM元素对象保存进新创建的jQuery对象中:

var $jq子对象=$(DOM元素对象)

3. 原理:

(1). $=jQuery=new jQuery  
(2). jq子对象其实是一个类数组对象，可保存找到的多个DOM元素对象

(3). 对jq子对象调用简化版函数，会被自动翻译为对应的原生DOM的方法和属性。

所以jQuery中的所有简化版函数效果和DOM中原生方法和属性效果一样。

jQuery中的this、e、e.target等，和DOM中的完全一样！

4. jQuery简化版函数3大特点:

(1). 自带for循环: 对整个jquery子对象调用一次简化版函数，等效于对jQuery子对象中保存的每个DOM元素对象分别调用一次对等的DOM原生方法或属性——不用自己写for循环

(2). 一个函数两用: 调用函数时:

a. 没给新值作为参数，默认执行获取旧值的操作

b. 给了新值作为参数，自动切换为执行修改操作

(3). （待续...）

5. 查找元素:

(1). jQuery支持用所以CSS3选择器查找

(2). jQuery新增选择器: (css中不能用)

a. 基本过滤: (下标从0开始)

:first :last :eq(i) :lt(i) :gt(i) :even :odd

b. 内容过滤:

:contains(文本) :has(选择器) :parent :empty

c. 可见性过滤:

:visible :hidden(只能选择display:none和input type="hidden")

d. 表单元素过滤:   
:input :text :password :radio :checkbox ... ...  
(3).用节点间关系查找: 2大类关系, 8个函数

a. 父子关系：3个函数:

1). $元素.parent()

2). $元素.children("选择器")

3). $元素.find("选择器")

b. 兄弟关系：5个

$元素.prev() $元素.prevAll("选择器")

$元素.next() $元素.nextAll("选择器")

$元素.siblings("选择器")

6. 修改元素: 3种:

修改中的所有函数，都是一个函数两用！

(1). 内容: 3种:

a. 原始HTML内容:

$元素.html("新HTML内容") 代替.innerHTML

b. 纯文本内容:

$元素.text("纯文本内容") 代替.textContent

c. 表单元素的值:

$元素.val("新值") 代替.value

(2). 属性: 3种:

a. 字符串类型的HTML标准属性:2种:

1). $元素.attr("属性名","新属性值")

代替 元素.getAttribute()和setAttribute()

2). $元素.prop("属性名", bool值)

代替 元素.属性名=bool值  
 b. bool类型的HTML标准属性: 只有1种

$元素.prop("属性名", bool值)

代替 元素.属性名=bool值

c. 自定义扩展属性:只有一种:

$元素.attr("属性名","新属性值")

代替 元素.getAttribute()和setAttribute()

反过来总结:

$元素.attr()可修改一切字符串类型的属性(字符串类型HTML标准属性+自定义扩展属性)

$元素.prop()可修改一切可用.访问的属性(所有HTML标准属性)

(3). 样式:

a. 获取或修改单个css属性: 只有1种

$元素.css("css属性名", "属性值")

获取属性值时被翻译为getComputedStyle(元素)

修改属性值时被翻译为.style.css属性=属性值

b. 使用class批量修改样式:

1). $元素.addClass("class名")

2). $元素.removeClass("class名")

3). $元素.hasClass("class名")

4). $元素.toggleClass("class名")

|  |
| --- |
| 修改相关的函数都可同时修改多个属性值:  $元素.attr或prop或css({  属性名:"属性值",  ... : ...  }) |

8. 添加删除替换克隆元素:

(1). 添加新元素: 2步

a. 使用HTML片段批量创建新元素:

$(`HTML片段`)

b. 将新元素添加到DOM树: 5种方式，10个函数

1). 末尾追加:

$父元素.append($新元素)

$新元素.appendTo($父元素)  
 2). 开头插入: 新增:   
 $父元素.prepend($新元素)

$新元素.prependTo($父元素)

3). 插入到一个现有元素之前:

$现有元素.before($新元素)

$新元素.insertBefore($现有元素)

4). 插入到一个现有元素之后:

$现有元素.after($新元素)

$新元素.insertAfter($现有元素)

5). 替换现有元素:

$现有元素.replaceWith($新元素)

$新元素.replaceAll($现有元素)

(2). 删除元素: $元素.remove()

(3). 克隆元素: $元素.clone()

(4). 判断一个元素是否具有什么特征: $元素.is("选择器")

9. 事件绑定:

(1). 标准写法: $元素.on("事件名", 事件处理函数)

(2). 简写: $元素.事件名(事件处理函数)

今日对应小程序视频列表:

小程序->在线->JQUERY->day01

5. jquery修改元素的内容 html() ...

小程序->在线->JQUERY->day02

2. jQuery修改元素属性attr() prop()

3. jquery修改样式css()

6. jquery 按节点间关系查找

6.1 jquery tabs ...

7. jquery添加删除替换克隆

8. jquery 事件 on off

9. jquery 事件委托

作业:

1. 复习今日小程序问题清单: 小程序->首页->JQUERY->day02

2. 预习明日小程序视频:

小程序->在线->JQUERY->day02 9. jquery事件委托on off

小程序->在线->JQUERY->day03 1. 页面加载完成后自动执行

2. 鼠标事件 模拟触发

3. jquery动画

4. 类数组对象操作

3.（凡是觉得DOM知识点串不起来）继续看小程序视频学习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