小程序: WEB问题速查

1. 问题清单: 历届师兄师姐在今天课程中集中问过的高频的重点的问题！

那些自己发现不了问题，又迫切想知道其它同学都问了什么问题，的同学

小程序->首页->JQUERY->day01

当老师在回答其它同学的问题时，你不要闲着！你可以打开问题清单浏览别人曾经问过的经典问题！

2. 高清无码小视频: 小程序->在线->JQUERY->day01

2部分: 1. 课上重点知识和案例的回放！2. 重点高频笔试面试题讲解扩展视频

万一家里或中心网络断了！暂时看不了直播了！

因为某些事儿请假了，错过了当天的直播！

课下复习时，觉得TMOOC视频太大，浏览不方便或TMOOC不能看了！

——打开小程序中"在线"，就可下载观看当前所讲内容和当天所讲全部内容的回放。

基础比较好的，确实学有余力的同学，想拿高薪的同学，在我答疑时，不要闲着:

要看今天扩展视频: 无扩展视频

3. 单词列表: 小程序->单词->JQUERY->day01

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. 凡是拼写错误，自己懒得找！我可以帮你找，但是找到后，错误的单词你要手抄20遍，并拍照！

出错: 先开f12，console，看错误提示！

3. 每个带注释的案例至少写三遍:

(1). 注释+代码抄一遍

(2). 保留注释，删除代码，自己试着把代码翻译回来！

(3). 删除注释和代码，自己试着用自己的话把注释写回来！

一个例子只有自己可以把人话的注释写回来！才算自己会！

问题: 一听就会一做就废！原因: 你自己无法用人话精确描述自己想干什么！

如何提问: （vip待遇的剩余时间8天）

1. 上课时，案例中，笔记中那句话哪行代码看不懂，立刻问！

2. 小程序中问题清单里哪个问题和回答看不懂！立刻问！

3. 个人项目中哪个功能不会做！立刻问！

4. 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原理

4. 查找元素

5. 修改元素

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

一. 什么是jQuery

1. 什么是jQuery:

(1). 第三方开发的: 不是原生js，需要先下载才能使用，比如bootstrap

(2).执行DOM操作的: jQuery还是在做DOM增删改查+事件绑定

学习jQuery其实还是在学习DOM5件事

(3).极简化的: 因为原生DOM繁琐，所以jQuery对DOM五件事进行了简化

jQuery是简化版的DOM

(4).函数库: jQuery中都是用函数来解决一切问题！jQuery是没有属性的！

jQuery中一切操作都要加()

2. 为什么: 2个原因:

(1). 简单: jQuery将DOM操作的每一步几乎都进行了终极的简化

(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

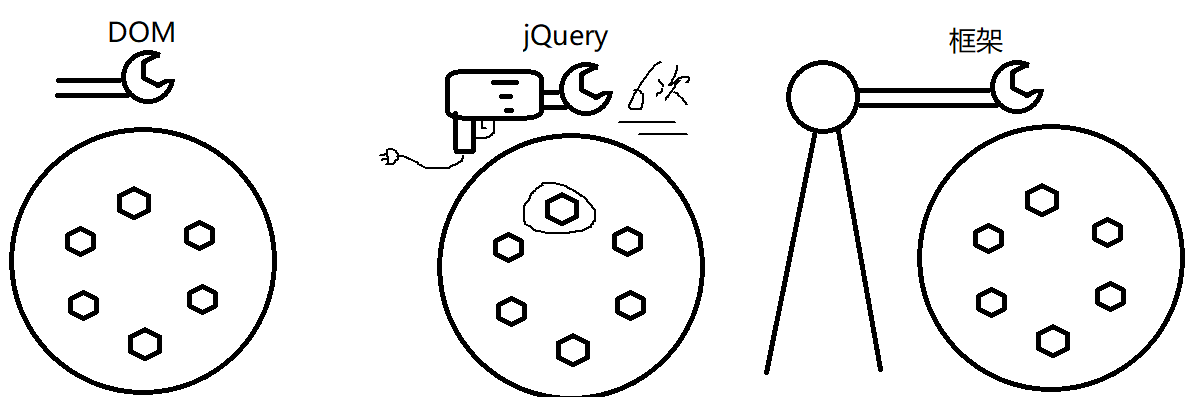
3. 何时: 在前端js框架出来之前，旧的项目和框架几乎都是用jquery做的！

但是，前端js框架出来之后，jquery使用越来越少！

4. 问题:

(1). 只有PC端，没有移动端！

(2). 仅仅对DOM的每一步进行了简化，但是没有彻底简化DOM开发的步骤！



二. 如何使用jQuery：

1. 下载: 官网: www.jquery.com

你们不用下载，在每天压缩包的js文件夹下已经下载好了

2. 版本:

(1). **1.x 唯一支持旧浏览器(IE8)**

a. 未压缩版: 包含完备的注释，代码结构和见名知意的变量名，所以适合学习和开发之用。但是体积大，不便于快速下载运行，所以不适合生产环境使用。

我们上课用jquery-1.11.3.js

b. 压缩版: 去掉了所有注释和代码结构，极简化了变量名，所以体积小，便于快速下载和运行，适合生产环境使用。但是可读性差，不适合学习和开发之用。

(2). 2.x 不再支持旧浏览器，也不支持ES6，比如不能用for of遍历jquery对象

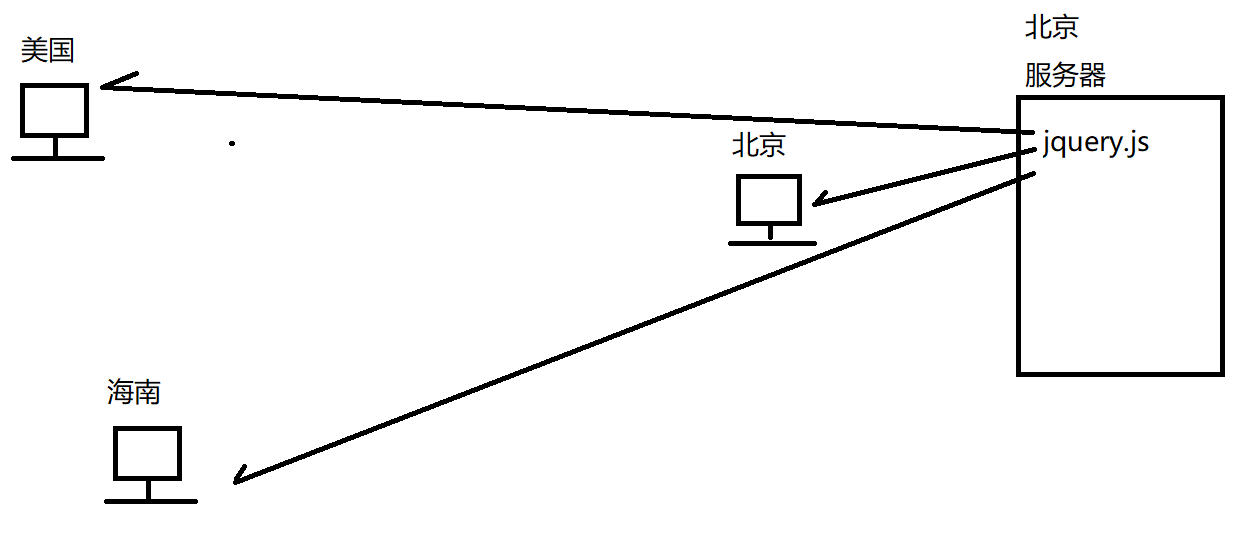
(3). 3.x 也不再支持旧浏览器，但是，开始支持ES6，比如可以用for of遍历jquery对象

3. 必须引入网页中才能使用: 2种:

(1). 将js文件下载到项目本地和项目文件放在一起使用相对路径加载到页面中

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

b. 问题: jquery.js文件只能部署在一个地点的服务器上。距离服务器位置不同的用户，访问时，下载时间会受距离的影响！

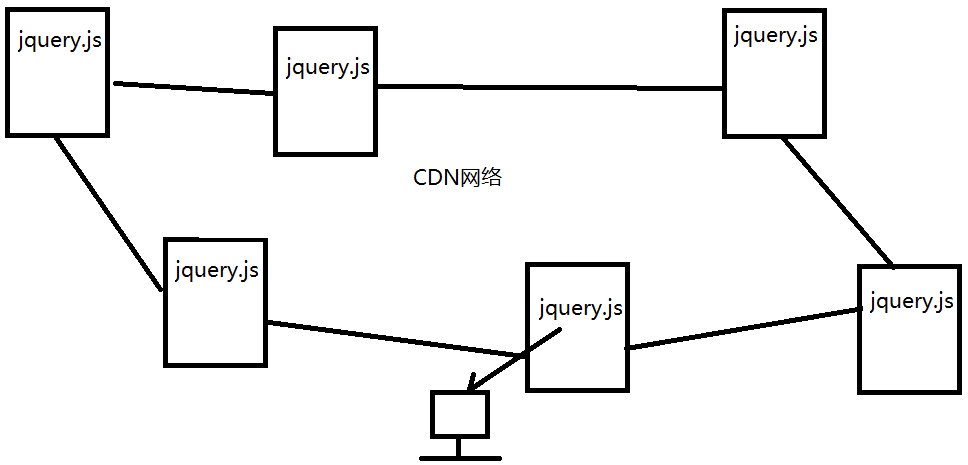


(2). 使用CDN网络中共享的jquery.js文件

a. CDN: (智能)内容分发网络

1). 可在全球多个地点的服务器上共享通用的第三方文件

2). 客户端访问CDN网络中的文件时，CDN网络会自动根据用户所在位置距离全球哪台服务器网络状况最优，来动态为客户端选择下载文件的服务器。



b. 优点: 无论身处何地的客户端下载共享的文件时，都能获得差不多一致的下载速度。

c. 如何: 其实所有第三方的框架或库，都提供了CDN网址

1). 不用下载js文件到本地

2).  **<script src="官方提供的CDN网址">**

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

4. 示例: 第一个jQuery程序: 点击按钮显示点击次数

1\_DOM.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">    <meta charset="UTF-8">    <title></title>  </head>  <body>    <button id="btn1">click me(0)</button>    <script>      //DOM 4步:      //1. 查找触发事件的元素      //本例中: 用户点id为btn1的按钮触发变化      var btn1=document.getElementById("btn1");      //2. 绑定事件处理函数      //本例中: 用户单击按钮触发变化      btn1.onclick=function(){        //3. 查找要修改的元素        //本例中: 就是要修改当前按钮自己，所以不用找        //就是this->btn1        //4. 修改元素        //本例中: 修改元素的内容,3步        //4.1 取出旧内容        //本例中: 取出按钮的内容，并截取()内的数字，转为整数        var n=parseInt(this.innerHTML.slice(9,-1));          //4.2 计算新值: 本例中: n++        n++;        //4.3 将新值放元素内容中        //本例中: 将新值n，拼接上原来完整的字符串，再放回按钮内容中        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>    <button id="btn1">click me(0)</button>    <!--0. 必须先引入jquery.js-->    <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: 用户点id为btn1的按钮触发变化      //var btn1=document.getElementById("btn1");      //           选择器      var $btn1=$("#btn1");      //2. 绑定事件处理函数      //本例中: 用户单击按钮触发变化      //btn1.onclick=function(){      $btn1.click(function(){        //3. 查找要修改的元素        //本例中: 就是要修改当前按钮自己，所以不用找        //就是this->btn1        var $this=$(this);//先不要问！        //4. 修改元素        //本例中: 修改元素的内容,3步        //4.1 取出旧内容        //本例中: 取出按钮的内容，并截取()内的数字，转为整数        //var n=parseInt(this.innerHTML.slice(9,-1));        var n=parseInt($this.html().slice(9,-1))        //4.2 计算新值: 本例中: n++        n++;        //4.3 将新值放元素内容中        //本例中: 将新值n，拼接上原来完整的字符串，再放回按钮内容中        //this.innerHTML=`click me(${n})`;        $this.html(`click me(${n})`)      //}      })    </script>  </body>  </html>  运行结果: |

三. jQuery原理:

1. 当我们引入<script src="js/jquery-1.11.3.js">时，其实是给内存中添加了一种新的类型，包含两部分:

(1). 构造函数: 专门负责创建该类型的子对象

function jQuery(){ ... }

(2). 原型对象: 替该类型所有子对象集中保存共有的方法

在jQuery.prototype中，集中保存的就是所有简化版的DOM操作的函数

(3). 只有jQuery构造函数创建的子对象才能使用jQuery原型对象中保存的简化版函数

别人家的子对象，无权使用！

2. 所以，要想使用jQuery家简化版函数，必须先创建jQuery家的子对象。如何创建:

(1). var jq子对象=new jQuery(...)

(2). 问题: jq子对象调用jQuery原型对象中的方法是为了对DOM元素执行操作！

所以，jq对象应该提前找到并保存住要操作的DOM元素才行

(3). 所以，创建jQuery对象时，必须提前放入一个要操作的DOM元素对象，2种:

a. 先用选择器查找DOM元素，再装入jq子对象中备用:

var jq子对象=new jQuery("选择器")

b. 如果已经获得一个DOM元素(this或e.target)，则无需再查找，就可直接放入jq子对象中:

var jq子对象=new jQuery(DOM元素对象)

(4). 问题: 每次操作一个DOM元素时，都需要重复写new jQuery()，太长了！

(5). 解决: 将new jQuery()封装进了jQuery()构造函数内部！

function jQuery( 选择器或DOM元素对象 ){

return new jQuery的真正构造函数( ... )

}

结果: 将来调用jQuery(xxx)等效于调用new jQuery(xxx)，省略了new！

(6). 问题: 作者连"jQuery"都懒得写

(7). 解决: 给jQuery起了个别名/外号: $=jQuery

(8). 总结: 今后创建jQuery子对象，只要写:

**var jq子对象=$("选择器"或DOM元素对象)**

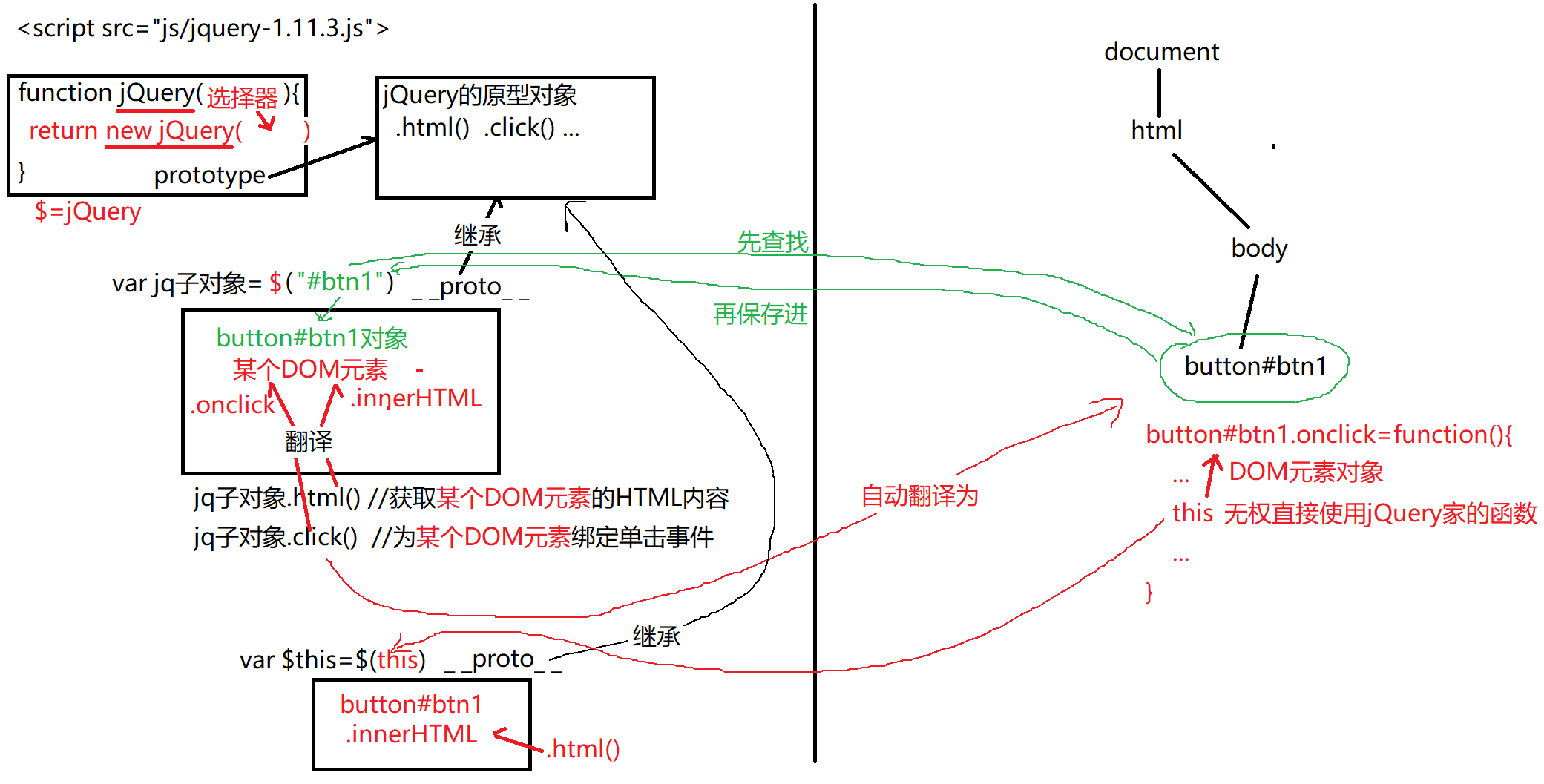
$()=>jQuery()=>new jQuery()

3. 结果: 获得了一个装有DOM元素对象的jQuery家子对象，简称jq对象

4. 对整个jq子对象调用简化版函数时，jq子对象会自动将简化版函数，翻译为原生的DOM对应的函数，作用到jq子对象中保存的DOM元素对象上

5. 变量命名习惯: 今后为了和DOM家的元素对象区分，所有**保存jQuery子对象的变量名，**习惯上都要**以$开头**！但是，不是必须！

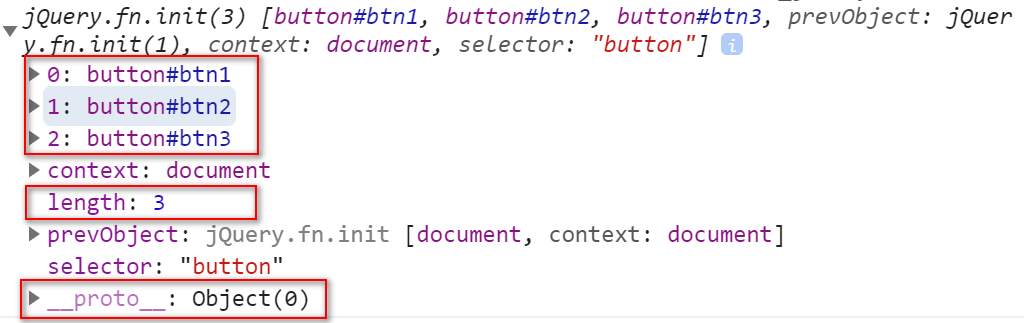
今后在jquery程序中，只要看到$开头的变量，几乎都是jQuery家的子对象，其中都保存着DOM元素对象。



6. jQuery子对象/查找结果对象的本质: 类数组对象

(1). 选择器找到几个DOM元素，jQuery子对象中就包含几个DOM元素对象。

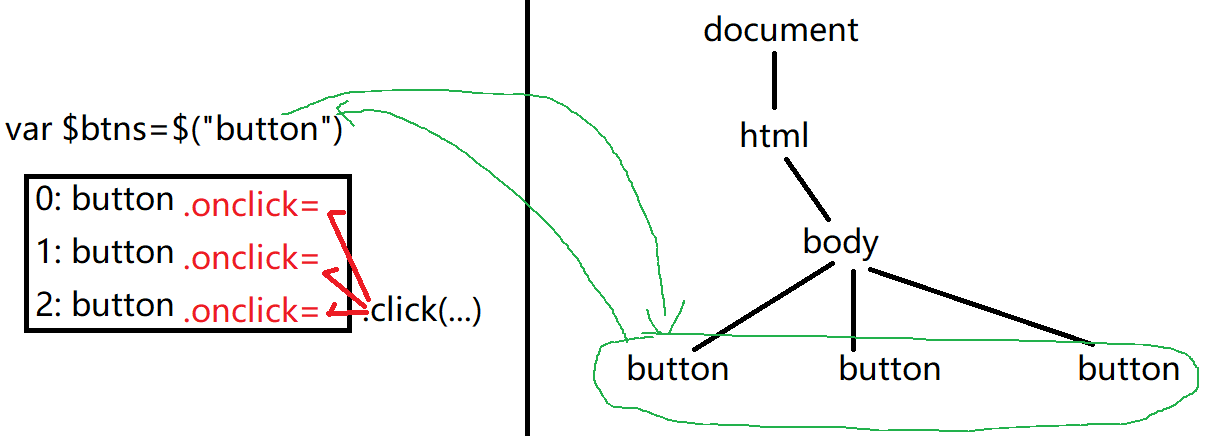
(2). 每个DOM元素对象都有下标。



7. jQuery简化版函数的三大通用特点:

(1). **每个jQuery简化版函数都自带for循环：**

只要对整个jQuery子对象调用一次简化版函数，就相当于对jQuery子对象内部保存的每个DOM元素都分别调用了一次对应的原生方法或属性！



(2). **凡是和修改相关的函数，都一个函数两用:** —— 重载！

a. 如果**没有给新值**，则该函数默认执行**读取**旧值的操作

比如: $btn1.html() 获取按钮的innerHTML内容

b. 如果**给了新值**，则该函数改为执行**修改**值的操作

比如: $btn1.html("hello") 修改按钮的内容为"hello"

(3). （碰到在讲...）

(4). 示例: 三个按钮，点击每个按钮，分别记录每个按钮的点击次数

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>      //DOM 4步      //1. 查找触发事件的元素      //本例中: 三个按钮都能点      //       css元素选择器      var $btns=$("button");      console.log($btns);//包含三个button元素对象的类数组对象      //2. 绑定事件处理函数      //本例中: 虽然三个按钮都能点击，但是jQuery简化版函数自带for循环，只要调用一次click()就可给三个按钮都绑定上单击事件处理函数      $btns.click(function(){        //3. 查找要修改的元素        //本例中: 就是要修改当前按钮自己，所以不用找        var $this=$(this);//this->DOM按钮对象，不能直接使用jQuery家函数.        //必须new jQuery(this)，转为jq子对象才有资格使用jQuery家的简化版函数。        //4. 修改元素        //本例中: 修改元素的内容,3步        //4.1 取出旧内容        //本例中: 取出按钮的内容，并截取()内的数字，转为整数        var n=parseInt($this.html().slice(9,-1))        //4.2 计算新值: 本例中: n++        n++;        //4.3 将新值放元素内容中        //本例中: 将新值n，拼接上原来完整的字符串，再修改回按钮内容中        $this.html(`click me(${n})`)      })      //自动翻译为三个onclick:      //$btns[0].onclick=function(){ ... }      //$btns[1].onclick=function(){ ... }      //$btns[2].onclick=function(){ ... }    </script>  </body>  </html>  运行结果: |

四. 查找元素: jQuery支持用所有CSS3选择器查找，并添加了少量jQuery独有选择器

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

1. 回顾: CSS中的子元素过滤选择器:

(1). 什么是: 根据元素在其父元素下的相对位置选择元素

(2). 包括:

a. :first-child 选择所有作为其所在父元素下的第一个孩子的元素

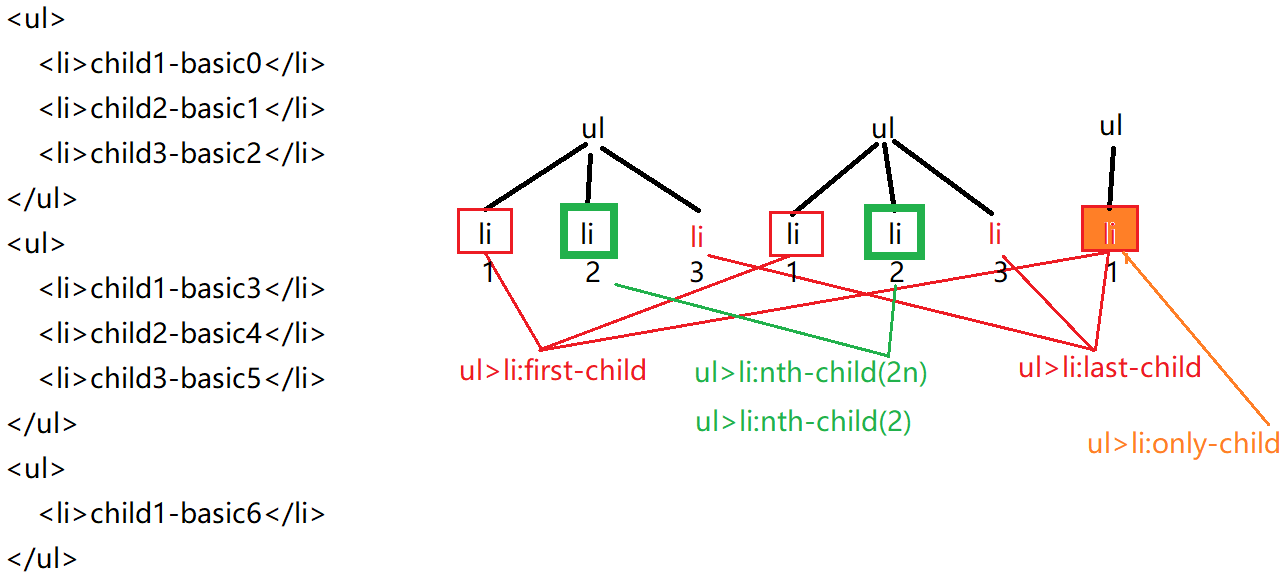
b. :last-child 选择所有作为其所在父元素下的最后一个孩子的元素

c. :nth-child(i) 选择所有作为其所在父元素下第i个子孩子的元素

css中的i都是从1开始！

d. :only-child 选择所有作为其所在父元素下的唯一一个孩子的元素

(3). 示例: 使用子元素过滤，选择指定的元素



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,设置他们的边框为红色      $("ul>li:first-child")      .css("border","1px solid red");      //查找每个ul中最后一个li,设置他们的字体为红色      $("ul>li:last-child")      .css("color","red");      //查找每个ul中处于偶数位置的，设置他们的阴影为绿色      $("ul>li:nth-child(2n)")      .css("box-shadow","0 0 5px green")      //查找每个ul中第2个li，设置其padding      $("ul>li:nth-child(2)")      .css("padding","10px 0")      //查找作为ul下唯一子元素的li，设置其背景色为橙色      $("ul>li:only-child")      .css("background-color","orange")    </script>  </body>  </html>  运行结果: |

2. 基本过滤选择器: （jQuery独有！）

补: 这些jQuery新增的独有选择器其实底层使用js程序模拟的！所以**下标从0开始**！且效率不如原生的css选择器高！

(1). 什么是: 先将所有符合条件的元素放在一个大的集合中，统一编号。按元素在大的结果集合中的下标位置选择对应的元素。

(2). 强调: 和元素在其父元素下的相对位置毫无关系！

(3). 何时: 今后希望按照元素的大排名位置选择元素，而不关心元素在其父元素下的相对位置时，都选择基本过滤选择器。

(4). 包括:

a. **:first** 所有符合条件的元素中，**第一个**元素

b. **:last** 所有符合条件的元素中，**最后一个**元素

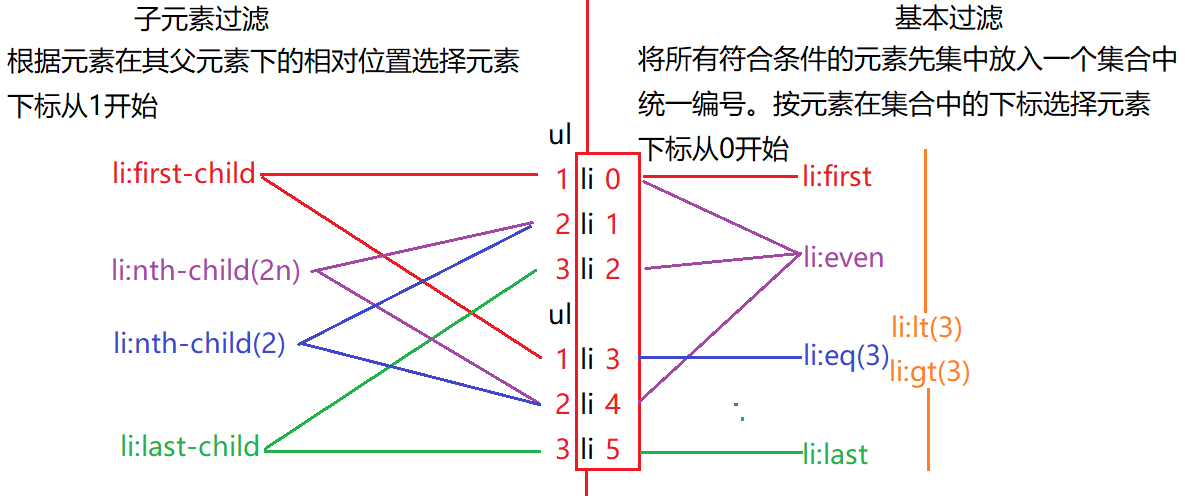
c. **:eq(i)**  equal 所有符合条件的元素中，下标**等于**i位置的一个元素(i从0开始)

d. **:lt(i)** less than 所有符合条件的元素中，**下标<i**位置的元素(i从0开始)

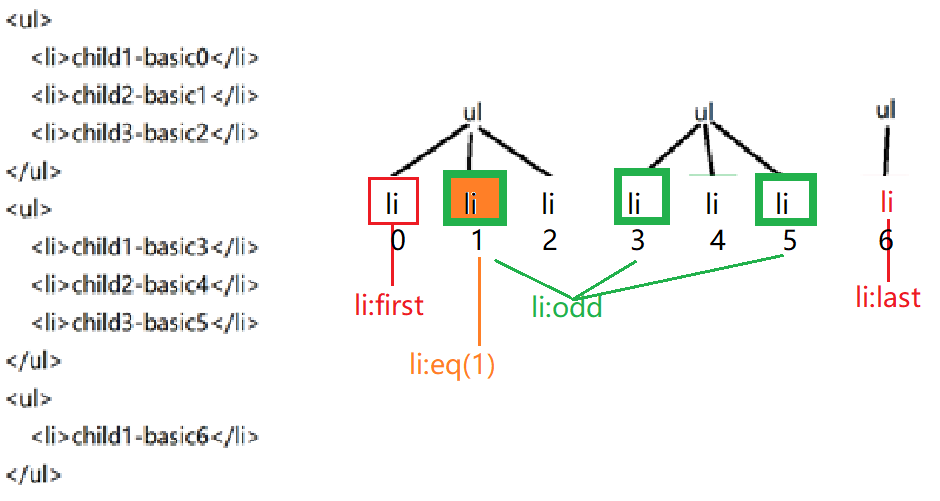
e. **:gt(i)** greater than 所有符合条件的元素中，**下标>i**位置的元素(i从0开始)

f. **:even**  所有符合条件的元素中，下标为**偶数**位置的元素**（非程序员视角是奇数）**

g.  **:odd**  所有符合条件的元素中，下标为**奇数**位置的元素**（非程序员视角是偶数）**



(5). 示例: 使用基本过滤选择器，选择指定的元素



6\_basic filter.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，加红边框      $("li:first").css("border","1px solid red");      //查找所有li中最后一个li，字体变为红色      $("li:last").css("color","red");      //查找处于偶数位置的li，加绿色阴影        $("li:odd").css("box-shadow","0 0 5px green")      //查找第2个li,背景变为橙色      $("li:eq(1)").css("background-color","orange")    </script>  </body>  </html>  运行结果: |

(6). 示例: 分别使用css和jquery实现按钮组效果

7\_basic filter.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>    <meta charset="utf-8" />    <title>...</title>    <style>      ul{list-style:none}      ul>li{        padding:5px 10px;        border:1px solid #555;        float:left;      }      /\*第一个li，左上，左下圆角\*/      ul>li:first-child{        border-radius:5px 0 0 5px;      }      /\*最后一个li，右上，右下圆角\*/      ul>li:last-child{        border-radius:0 5px 5px 0;      }      /\*去掉每个li右边相邻的li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>      // //第一个li左上左下圆角      // $("ul>li:first")      // .css("border-radius","5px 0 0 5px");      // //最后一个li右上右下圆角      // $("ul>li:last")      // .css("border-radius","0 5px 5px 0");      // //第一个li之后所有li去掉左边框      // $("ul>li:gt(0)").css("border-left",0)    </script>  </body>  </html>  运行结果： |

对应小程序视频: 小程序->在线->JQUERY->day01 4. jQuery其它选择器

3. 内容过滤: 根据元素内容的不同查找不同的元素  
 (1). **:contains(关键词)** 根据元素**开始标签到结束标签之间的内容中的关键词**查找元素

(2). **:has(选择器)**  根据元素内的子元素具有的特征**查找父元素**

(3). **:parent**  选择所有内容**不为空的元素**

(4)**. :empty** 选择所有内容**为空的元素**

说明:empty只能匹配开始标签和结束标签之间没有任何字符的元素

(5). 示例: 使用内容过滤选择器选择符合条件的元素

10\_content filter.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">    <meta charset="UTF-8">    <title></title>    <link rel="stylesheet" href="css/bootstrap.css"/>    <style>    </style>  </head>  <body>  <div class="container">  <h1>jQuery中的选择器——内容过滤选择器</h1>    <button>提交订单</button>    <button>Submit注册信息</button>    <button>马上提交</button>    <button>清空重填</button>    <hr/>    <div class="alert" id="alert1"></div>    <div class="alert" id="alert2">      <span class="close">×</span>    </div>    </div>  <script src="js/jquery-1.11.3.js"></script>  <script>    //选择包含"提交"二字的按钮，变为绿色按钮    $("button:contains(提交)")    .css("background-color","lightGreen")    //选中包含.close按钮的.alert元素，让它们变为红色的警告框    $(".alert:has(.close)")    .css("background-color","pink");    //选中不包含.close按钮的.alert元素，让它们变为绿色的警告框    $(".alert:not(:has(.close))")    .css("background-color","lightGreen");    //空的.alert，加红阴影    $(".alert:empty")    .css("box-shadow","0 0 5px red")    //非空的.alert，加绿阴影    $(".alert:parent")    .css("box-shadow","0 0 5px green")  </script>  </body>  </html>  运行结果: |

4. 可见性过滤: 根据元素是否可见来选择元素

(1). **:visible**  选择所有可见的元素

(2). **:hidden**  选择所有不可见的元素

强调: :hidden**只能选择**两种隐藏的元素: **display:none 和 input type="hidden"**

~~不能选择visibility:hidden和opacity:0~~

5. 表单元素过滤选择器: 专门选择表单元素的选择器

(1). **:input** 选择**所有**表单元素(input textarea select button )

vs input CSS元素选择器，仅选择input元素，不选择其他元素

(2). 其实每种input type都对应着一个专门的选择器:

a. **:text** 专门选择input type="text"的文本框

b. **:password**  专门选择input type="password"的密码框

c. **:radio**  专门选择input type="radio"的单选按钮

d. **:checkbox**  专门选择input type="checkbox"的复选框

...

(3). 示例: 点同意，控制其他表单元素的启用和禁用状态

13\_form state selector.html

|  |
| --- |
| <!DOCTYPE html>  <html>   <head>    <title>.....</title>    <meta charset="utf-8"/>    <style>    </style>   </head>   <body>    <form>      用户名:<input disabled></br>      密码:<input type="password" disabled></br>      <input type="checkbox">我同意本站的使用条款<br>      <button type="submit" disabled>提交注册信息</button>    </form>    <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: 用户点checkbox元素触发变化      var $chb=$(":checkbox");      //console.log($chb);      //2. 绑定事件处理函数      //本例中: 单击checkbox触发变化      $chb.click(function(){        //3. 查找要修改的元素        //本例中: 点checkbox是为了修改除checkbox以外其他表单元素的状态        var $others=$(":input:not(:checkbox)");        //console.log($others);        //4. 修改元素        //如果当前checkbox是选中的（checked==true），说明同意，其他表单元素启用(disabled=false)        //否则如果当前checkbox是未选中（checked==false）的，说明不同意，其他表单元素就禁用(disabled=true)        //总结规律:        //如果checked==true,则disabled=false        //如果checked==false,则disabled=true        //总结: **disabled=!checked 唱反调**        //只要让其他元素的disabled属性=!当前checkbox的checked属性        //语法错误:**jquery中全是函数，不能用属性！**        //$others.disabled=!this.checked;        //jq中:        var $this=$(this);        $others.prop("disabled",!$this.prop("checked"));        //                        ←-------------------        //                     取出当前checkbox的checked属性值        //                      ←        //                     取反        //←---------------------        //赋值给其他表单元素的disabled属性      })    </script>   </body>  </html>  运行结果: |

五. 修改: 3种:

**修改中的所有函数，都是一个函数两用！**

1. 内容: 3种:

对应小程序视频: 小程序->在线->JQUERY->day02 1. jquery修改元素的内容 html() ...

(1). 原始HTML内容: **$元素.html("新HTML内容")**  被翻译为-> .innerHTML

(2). 纯文本内容: **$元素.text("纯文本内容")** 被翻译为-> .textContent

(3). 表单元素的值: **$元素.val("新值")**  被翻译为-> .value

(4). 示例: 使用修改内容方式，实现表单验证

14\_html\_val.html

|  |
| --- |
| <!DOCTYPE html>  <html>   <head>    <title> new document </title>    <meta charset="utf-8">   </head>   <body>    <h1>操作元素的内容和值</h1>    <form action="">      用户名:<input name="uname">            <span></span><br>      密码:<input type="password" name="upwd">            <span></span><br>      <input type="submit" value="提交注册信息">    </form>    <script src="js/jquery-1.11.3.js"></script>    <script>      //正确时，使用图片:"<img src='img/ok.png'>"      //姓名错误时: "<img src='img/err.png'>用户名必须介于3~9位之间!"      //密码错误时: "<img src='img/err.png'>密码必须介于6~8位之间!"      //DOM 4步      //1. 查找触发事件的元素      //本例中: 文本框失去焦点时，触发验证和变化      $(":text")      //2. 绑定事件处理函数      //本例中: 失去焦点触发变化      .blur(function(){        //为了防止再使用原生DOM属性和方法，一开始就把this包装一下        var $this=$(this);        //3. 查找要修改的元素        //本例中: 修改当前文本框旁边的span元素        var $span=$this.next();//.nextElementSibling找的是下一个元素对象！所以返回结果也是一个包含找到的元素对象的jquery对象        //4. 修改元素        //获得当前文本框的内容        var value=$this.val();//.value -> 字符串        //如果验证当前文本框的内容通过        if(value.length>=3&&value.length<=9){          //则修改span的内容为<img src='img/ok.png'>          $span.html(`<img src='img/ok.png'>`);        }else{//否则如果验证文本框的内容不通过          //则修改span的内容为<img src='img/err.png'>用户名必须介于3~9位之间!          $span.html(`<img src='img/err.png'>用户名必须介于3~9位之间!`)        }      })    </script>   </body>  </html>  运行结果: |

2. 属性: 3种:

(1). 字符串类型的HTML标准属性:（待续...）  
 (2). bool类型的HTML标准属性:

a. DOM中只有一种操作方式: 元素.属性名

b. 问题: jquery是函数库，不能用属性

c. jq中也只有一种操作方式: 提供了一个专门的函数来代替.操作:

**$元素.prop("属性名", bool值)**

property的缩写，属性，意为获取或修改一个属性的值。

(3). 自定义扩展属性:

3. 样式:

(1). 获取或修改内联样式:

a. DOM中: 元素.style.css属性="属性值"

自动翻译为↑

b. jq中: **$元素.css("css属性名", "属性值")**

总结:

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:

a. 基本过滤:

: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

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, 6个属性

a. 父子关系：（待续...）

b. 兄弟关系：2个

1). 前一个兄弟: **$元素.prev()**

被自动翻译->元素.previousElementSibling

2). 下一个兄弟: **$元素.next()**

被自动翻译->元素.nextElementSibling

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")

被翻译为-> .innerHTML

b. 纯文本内容:

$元素.text("纯文本内容")

被翻译为-> .textContent

c. 表单元素的值:

$元素.val("新值")

被翻译为-> .value

(2). 属性: 3种:

a. 字符串类型的HTML标准属性:（待续...）  
b. bool类型的HTML标准属性:

$元素.prop("属性名", bool值)

被翻译为->元素.属性名=bool值

c. 自定义扩展属性:（待续...）

(3). 样式:

a. 获取或修改内联样式:

$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: （待续...）

8. 添加删除替换克隆元素: （待续...）

9. 事件绑定: **$jq对象.事件名(function(){ ... })**

今日对应小程序视频列表:

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

2. 第一个jQuery程序 jQuery原理

3. 子元素过滤 child filter 基本过滤

4. jQuery其它选择器

小程序->在线->JQUERY->day02 1. jquery修改元素的内容 html() ...

作业:

1. 复习今日小程序中问题清单: 小程序->首页->JQUERY->day01

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的效果

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. 看小程序视频完成表单验证效果：

小程序->在线->JQUERY->day02 1. jquery修改元素的内容 html() ...

4.（凡是觉得DOM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