# 5.jQuery

## 5.1.jQuery概述

**什么是jQuery**

jQuery 是一个“写的更少，但做的更多”的轻量级 JavaScript 函数库。

**jQuery的优势**

* 可以简化JavaScript代码
* 可以像css那样获取元素
* 可以修改css来控制页面效果
* 可以兼容常用的浏览器

**jQuery版本支持**

jQuery分为很多版本, 还分为未压缩版和压缩版, 根据需要选择对应的版本进行下载!

1.x 支持常用的浏览器和IE6+

2.x 支持常用的浏览器和IE9+

3.x 支持常用的浏览器和IE9+

注意: jQuery不兼容老版本. 因为jQuery升级除了会做一些内部优化之外, 还会删除以前的一些代码, 比如删除一些方法. 或者是添加一些新的方法。所以在升级之后， 以前的代码可能会无法执行。

**jQuery引入**

****

jQueyr类库其实就是一个普通的js文件, 和之前在html中引入js文件方式是一样的！！！

## 5.2.jQuery语法

**$介绍**

$符号等价于jQuery, 是jQuery单词的简写

例如：$()相当于调用jQuery()，该函数会返回一个jQuery对象, 这个jQuery对象中包含零个或多个html元素, 比如: $("div")，可以通过jQuery中提供的方法来操作这些匹配的元素，比如$("div").remove()；

**文档就绪事件**

所谓的文档就绪事件, 就是在整个html文档加载完之后立即触发, 执行一些操作，格式如下：

$(document).ready(function(){

//xxxx

});

该函数会在整个html文档加载完之后立即执行! 其作用相当于:

window.onload = function(){ //xxx }

**其简写形式为:**

$(function(){

//xxxx

});

**DOM对象和jQuery对象互相转化（案例1）**

JS对象只能调用JS上提供的属性和方法, 不能调用jQuery提供的属性和方法, 如果非要使用, 必须将JS对象转化成jQuery对象！反之亦然。

**dom对象转jQuery对象:**

var dom = document.getElementById("username");

var $jQuery= $(dom ); // js对象转成jQuery对象

**jQuery对象转dom对象:**

var $jQuery = $("#username");

//方式一:

var dom1 = $jQuery[0]; // jQuery对象转成js对象

//方式二:

var dom2 = $jQuery.get(0); // jQuery对象转成js对象

**案例一：jQuery对象和js对象的相互转换**

|  |
| --- |
| /\*通过点击按钮获取用户名的值，分别将jQuery对象转换成js对象和js对象转换为jQuery对象\*/  /\* 文档就绪事件 \*/  $(function(){  //为所有匹配的元素绑定点击事件  $("#btn").click(function(){  //jQuery对象  var $jQuery = $("#username");  //alert($jQuery.val());    //jQuery对象转成js对象:  //方式一:  var js1 = $jQuery[0];  //alert(js1.value);  //方式二:  var js2 = $jQuery.get(0);  //alert(js2.value);  //js对象转成jQuery对象:  var oInp = document.getElementById("username");  var $Inp = $(oInp);  alert($Inp.val());  });  }) |

## 5.3.jQuery选择器

### 5.3.1.基本选择器

（selector案例一）

**元素名选择器**

$("div") – 匹配所有的div元素

**class选择器**

$(".c1") – 匹配所有class值为c1的元素

$("div.c1") – 匹配所有class值为c1的div元素

**id选择器**

$("#d1") – 匹配所有id值为d1的元素

$("div#d1") – 匹配所有id值为d1的div元素

**\*号匹配符**

$("\*") – 匹配所有的元素

**多元素选择器**

$("div,span,#d1,.c1") – 匹配所有的div/span元素以及id值为d1的元素和class值为c1的元素。。。

**selector案例一：基本选择器**

|  |
| --- |
| //让整个页面加载完成  $(document).ready(function() {  /\* ---------基本选择器练习--------- \*/  //改变元素名为 <div> 的所有元素的背景色为 #FF69B4" id="b1"  $("#b1").click(function(){  $("div").css("background-color", "#FF69B4");  });  //改变 id 为 one 的元素的背景色为 #9ACD32" id="b2"  $("#b2").click(function(){  $("#one").css("background-color", "#9ACD32");  });  //改变 class 为 mini 的所有元素的背景色为 #FF0033" id="b3"  $("#b3").click(function(){  $(".mini").css("background-color", "#FF0033");  });  //改变所有元素的背景色为 #FDF5E6" id="b4"  $("#b4").click(function(){  $("\*").css("background-color", "#FDF5E6");  });  //改变所有的<span>元素和 id 为 two 的元素的背景色为 #FF1493" id="b5"  $("#b5").click(function(){  $("span,#two").css("background-color", "#FF1493");  });    //改变所有的<span>元素和 id 为 two 的,id为one的，class为 mini的所有的元素的背景色为 #006400" id="b6"  $("#b6").click(function(){  $("span,#two,#one,.mini").css("background-color", "#006400");  });  }); |

### 5.3.2.层级选择器

（selector案例二）

如果想通过DOM元素之间的层次关系来获取特定元素。例如子元素、兄弟元素等。则需要通过层次选择器。

$("div span") – 匹配div下所有的span元素

$("div>span") – 匹配div下所有的span子元素

$("div+span") – 匹配div后面紧邻的span兄弟元素

$("div~span") – 匹配div后面所有的span兄弟元素

**selector案例二：层级选择器**

|  |
| --- |
| $().ready(function() {  /\* ---------层级选择器练习--------- \*/  //改变 <body> 内所有 <div> 的背景色为 #F08080" id="b1"  $("#b1").click(function(){  $("body div").css("background-color", "#F08080");  });  /\*  $("#b1").click(function(){})  等价于:  document.getElementById("b1").onclick = function(){};  \*/  //改变 <body> 内子 <div> 的背景色为 #FF0033" id="b2"  $("#b2").click(function(){  $("body>div").css("background-color", "#FF0033");  });  //改变 id 为 one 的下一个 <div> 的背景色为 #0000FF" id="b3"  $("#b3").click(function(){  $("#one+div").css("background-color", "#0000FF");  });  //改变 id 为 two 的元素后面的所有兄弟<div>的元素的背景色为 #9ACD32" id="b4"  $("#b4").click(function(){  $("#two~div").css("background-color", "#0000FF");  });  //改变 id 为 two 的元素所有 <div> 兄弟元素的背景色为 #FF6347" id="b5"  $("#b5").click(function(){  $("#two").siblings("div").css("background-color", "#0000FF");    //$("#two").next("div").css("background-color", "#0000FF");//选择id为two元素后面紧邻的div兄弟元素  //$("#two").nextAll("div").css("background-color", "#0000FF");//选择id为two元素后面所有的div兄弟元素  //$("#two").prev("div").css("background-color", "#0000FF");//选择id为two元素前面紧邻的div兄弟元素  //$("#two").prevAll().css("background-color", "#0000FF");//选择id为two元素前面所有的兄弟元素  });  }) |

### 5.3.3.基本过滤选择器

（selector案例三）

过滤选择器主要是通过特定的过滤规则来筛选出所需的DOM元素, 该选择器都以 ":" 开头

$("div:first") – 匹配所有div中的第一个div元素

$("div:last") – 匹配所有div中的最后一个div元素

$("div:even") – 匹配所有div中索引值为偶数的div元素，0开始

$("div:odd") – 匹配所有div中索引值为奇数的div元素，0开始

$("div:eq(n)") – 匹配所有div中索引值为n的div元素，0开始

$("div:lt(n)") – 匹配所有div中索引值小于n的div元素，0开始

$("div:gt(n)") – 匹配所有div中索引值大于n的div元素，0开始

$("div:not(.one)") – 匹配所有class值不为one的div元素

**selector案例三：基本过滤选择器**

|  |
| --- |
| $(function() {  /\* ---------基本过滤选择器练习--------- \*/  //改变第一个 div 元素的背景色为 #FF6347" id="b1"  $("#b1").click(function(){  $("div:first").css("background-color", "#FF6347");  });  //改变最后一个 div 元素的背景色为 #9ACD32" id="b2"  $("#b2").click(function(){  $("div:last").css("background-color", "#FF6347");  });  //改变class不为 one 的所有 div 元素的背景色为 #FF0033" id="b3"  $("#b3").click(function(){  $("div:not(.one)").css("background-color", "#FF0033");  });  //改变索引值为等于 3 的 div 元素的背景色为 #006400" id="b4"  $("#b4").click(function(){  $("div:eq(3)").css("background-color", "#006400");  });  //改变索引值为小于 3 的 div 元素的背景色为 #FF69B4" id="b5"/>  $("#b5").click(function(){  $("div:lt(3)").css("background-color", "#FF69B4");  });    //改变索引值为大于 3 的 div 元素的背景色为 #F08080" id="b6"  $("#b6").click(function(){  $("div:gt(3)").css("background-color", "#F08080");  });    //改变索引值为偶数的 div 元素的背景色为 #FF6347" id="b7"  $("#b7").click(function(){  $("div:even").css("background-color", "#F08080");  });    //改变索引值为奇数的 div 元素的背景色为 #FF1493" id="b8"  $("#b8").click(function(){  $("div:odd").css("background-color", "#FF1493");  });  }) |

### 5.3.4.内容选择器

（selector案例四）

$("div:contians('abc')") – 匹配所有div中包含abc内容的div元素

如: <div>xxxabcxx</div>

$("div:has(p)") – 匹配所有包含p元素的div元素

如: <div><p></p></div>

$("div:empty") – 匹配所有内容为空的div元素

如: <div></div>

$("div:parent") – 匹配所有内容不为空的div元素

如: <div>xxxxx</div>

### 5.3.5.可见选择器

$("div:hidden") – 匹配所有隐藏的div元素

$("div:visible") – 匹配所有可见的div元素

**selector案例四：内容+可见选择器**

|  |
| --- |
| $(function() {  /\* ---------内容选择器--------- \*/  //改变含有文本 ‘id’ 的 div 元素的背景色为 #FF6347" id="b1" <div>xxidxx</div>    $("#b1").click(function(){  $("div:contains('id')").css("background-color", "#FF6347");  });  //改变空元素div(既不包含文本也不包含子元素)的背景色为 #9ACD32" id="b2"  $("#b2").click(function(){  $("div:empty").css("background-color", "#9ACD32");  });  //改变包含div子元素的div元素的背景色为 #FF0033" id="b3"  $("#b3").click(function(){  $("div:has(div)").css("background-color", "#FF0033");  });  //改变非空div元素的背景色为 #006400" id="b4"  $("#b4").click(function(){  $("div:parent").css("background-color", "#006400");  });  //====================================================  /\* ---------可见选择器练习--------- \*/  //改变所有可见 div 元素背景色为 #F08080" id="b5"  $("#b5").click(function() {  $("div:visible").css("background-color", "#F08080");  })    //让所有隐藏的div元素显示, 并改变背景色为 #FF69B4" id="b6"  $("#b6").click(function() {  $("div:hidden").css("background-color", "red").show();    })  }) |

### 5.3.6.属性选择器

（selector案例五）

$("div[id]") – 匹配所有具有id属性的div元素

$("div[id='d1']") – 匹配所有具有id属性并且值为d1的div元素

$("div[id!='d1']") – 匹配所有id属性值不为d1的div元素

$("div[id^='d1']") – 匹配所有id属性值以d1开头的div元素

$("div[id$='d1']") – 匹配所有id属性值以d1结尾的div元素

### 5.3.7.子元素选择器

$("div:nth-child(n)") – n从1开始, 匹配div中第n个子元素。

$("div:first-child") –匹配div中第1个子元素。

$("div:last-child") –匹配div中最后一个子元素。。。

**selector案例五：属性+子元素选择器**

|  |
| --- |
| $(function() {  /\* ---------属性选择器练习--------- \*/  //改变包含id属性的 div 元素的背景色为 #FF6347" id="b1"  $("#b1").click(function(){  $("div[id]").css("background-color", "#FF6347");  });  //改变包含属性title='aa' 的div 元素的背景色为 #9ACD32" id="b2"  $("#b2").click(function(){  $("div[title='aa']").css("background-color", "#9ACD32");  });  //改变type属性不等于button的input 元素的背景色为 #FF0033" id="b3"  $("#b3").click(function(){  $("input[type!='button']").css("background-color", "#9ACD32");  });    //=================================================  /\* ---------子元素选择器练习--------- \*/  //改变div 第二个子元素的背景色为 #006400" id="b4"  $("#b4").click(function() {  $("div:nth-child(2)").css("background-color", "#006400");  })  //改变div 第一个子元素的背景色为 #FF69B4" id="b5"  $("#b5").click(function() {  //$("div:nth-child(1)").css("background-color", "#FF69B4");  $("div:first-child").css("background-color", "#FF69B4");  //$("div:last-child").css("background-color", "#FF69B4");  })  }) |

### 5.3.8.表单选择器

（selector案例六）

$(":input") – 匹配所有的input文本框、密码框、单选框、复选框、select框、textarea、button。

$(":password") – 匹配所有的密码框

$(":radio") – 匹配所有的单选框

$(":checkbox") – 匹配所有的复选框

$(":checked") – 匹配所有的被选中的单选框/复选框/option

$("input:checked") – 匹配所有的被选中的单选框/复选框

$(":selected") – 匹配所有被选中的option选项

**selector案例六：表单选择器**

|  |
| --- |
| $(function() {  /\* ---------表单选择器练习--------- \*/  //改变所有:input 元素的背景色为 #F08080" id="b1"  $("#b1").click(function() {  $(":input").css("background-color", "#F08080");  alert($(":input").length);  });  //改变:password 元素的背景色为 #9ACD32" id="b2"  $("#b2").click(function() {  $(":password").css("background-color", "#9ACD32");  });  //弹出 :radio 元素的个数" id="b3"  $("#b3").click(function() {  alert($(":radio").length);  });  //弹出所有 :checked 元素的value值" id="b4"  $("#b4").click(function() {  $(":checked").each(function(){  alert($(this).val());  });  });    //弹出所有 :selected 元素的value值" id="b5"  $("#b5").click(function() {  $(":selected").each(function(){  alert($(this).val());  });  });  }) |

## 5.4.文档操作

parent()

$("#d1").parent() – 获取id为d1元素的父元素

parents()

$("#d1").parents() – 获取id为d1元素的祖先元素

$("#d1").parents("tr") – 获取id为d1元素的tr祖先元素

next()

$("div").next() – 获取所匹配元素后面紧邻的兄弟元素

$("div").next("span") – 获取所匹配元素后面紧邻的span兄弟元素

nextAll()

$("div").nextAll() – 获取所匹配元素后面所有的兄弟元素

$("div").nextAll("span") – 获取所匹配元素后面所有的span兄弟元素

prev()

$("div").prev() – 获取所匹配元素前面紧邻的兄弟元素

$("div").prev("span") – 获取所匹配元素前面紧邻的span兄弟元素

prevAll()

$("div").prevAll() – 获取所匹配元素前面所有的兄弟元素

$("div").prevAll("span") – 获取所匹配元素前面所有的span兄弟元素

siblings()

$("div").siblings() – 获取所匹配元素前后所有的兄弟元素

$("div").siblings("span") – 获取所匹配元素前后所有的span兄弟元素

find("span") – 获取所匹配元素后代中的所有span元素

each(function(){ }) – 遍历选中元素

$("<p>abc</p>") – 创建一个游离的p元素。

append()

$("div").append("<span></span>") –为所匹配元素追加一个span子元素

remove()

$("div").remove() – 删除所匹配元素

html()

$("div").html() – 获取所匹配元素的html内容

$("div").html("xxx") – 为所匹配元素设置html内容

text()

$("div").text() – 获取所匹配元素的文本内容

$("div").text("xxx") – 为所匹配元素设置文本内容

attr()

$("div").attr("id") – 获取所匹配元素的id属性值

$("div").attr("id", "xx") – 为所匹配元素设置id属性

css

$("div").css("width") – 获取所匹配元素的width样式属性值

$("div").css("width", "200px") – 为所匹配元素设置width样式属性

$("div").css({"width":"200px", "color":"red","font-size":"24px" }) ; – 为所匹配元素设置width样式属性

## 5.5.事件

click()

$("div").click(function(){}) – 为所匹配元素绑定点击事件

blur()

$("input").blur(function(){}) – 为所匹配元素绑定失去输入焦点事件

focus()

$("input").focus(function(){}) – 为所匹配元素绑定获得输入焦点事件

change()

$("select").change(function(){}) – 为所匹配元素绑定选项切换事件

ready()

$(document).ready(function(){}) – 文档就绪事件

其作用相当于: window.onload = function(){}

简写形式为:

$(function(){}) – 在整个文档加载完成后立即执行

## 5.6.效果

show()

$("div").show() – 将隐藏元素设置为显示(底层操作的是display);

hide()

$("div").show() – 将显示元素设置为隐藏(底层操作的是display);

toggle()

$("div").toggle() – 切换元素的可见状态, 如果元素显示则设置为隐藏, 如果元素隐藏则设置为可见.

**（Exercise案例）**

**Exercise案例一：仿QQ列表分组**

|  |
| --- |
| /\* --通过jQuery实现访QQ列表好友分组-- \*/  function openDiv(thisobj){  //thisobj --> 被点击的A标签  //在显示当前div之前, 先隐藏其他div  $(thisobj).parents("tr").siblings().find("div").hide();  $(thisobj).next().toggle();  } |

**Exercise案例二：二级联动下拉框**

|  |
| --- |
| var data = {  "北京市": ["海淀区","朝阳区","丰台区"],  "河北省": ["石家庄","唐山","秦皇岛"],  "辽宁省": ["沈阳","大连","鞍山"],  "山东省": ["青岛","济南","烟台"]  }    /\* --通过jQuery实现二级联动下拉框 -- \*/  function selectCity(thisobj){  //获取用户选中的省份  var prov = $(thisobj).val();  //获取省份对应的市区数据  var citys = data[prov];  //先清空之前的数据  $("#city").html("<option>--选择城市--</option>");    //循环遍历取出每一个城市作为option选项填充到第二个下拉框中  for(var i=0; i<citys.length; i++){  $("#city").append("<option>"+citys[i]+"</option>");  }  } |

**Exercise案例三：模拟员工信息管理系统**

|  |
| --- |
| //html文档加载完执行  $(function(){  //为"添加"按钮绑定点击事件(点击按钮添加员工信息到table中)  $("#add").click(function(){  //获取输入员工信息  var id = $("#box1 input[name='id']").val().trim();  var name = $("#box1 input[name='name']").val().trim();  var email = $("#box1 input[name='email']").val().trim();  var salary = $("#box1 input[name='salary']").val().trim();  // 校验员工信息  //员工信息不能为空  if(id == "" || name == "" || email == "" || salary == ""){  alert("员工信息不能为空");  return;  }  //员工id不能重复  var flag = false;  $("table tr").each(function(){  //判断id是否等于每行第二个td内的文本, 即id  if($(this).find("td:eq(1)").text() == id){  alert("ID已存在！");  flag = true;  return false;  }  });  if(flag){  return;  }  //将新员工的信息插入到员工列表中  var $tab = $("table");  var $tr = $("<tr></tr>");    var $td0 = $("<td><input type='checkbox'></td>");  var $td1 = $("<td>"+id+"</td>");  var $td2 = $("<td>"+name+"</td>");  var $td3 = $("<td>"+email+"</td>");  var $td4 = $("<td>"+salary+"</td>");  $tr.append($td0).append($td1).append($td2).append($td3).append($td4);  $tab.append($tr);  });  //为全选按钮绑定点击事件  $("#all").click(function(){  //获取全选状态  var check = $(this).attr("checked");  //将所有复选框状态与全选保持一致  $("table input[type='checkbox']").attr("checked",check);  });  //为"删除"绑定点击事件(点击"删除",删除所选员工信息)  $("#del").click(function(){  $("input:checked:not(input[id='all'])").parents("tr").remove();  });  //为"修改"绑定点击事件(点击"修改",修改指定ID的员工信息)  $("#upd").click(function(){  //获取修改的员工信息    var id = $("#box2 input[name='id']").val().trim();  var name = $("#box2 input[name='name']").val().trim();  var email = $("#box2 input[name='email']").val().trim();  var salary = $("#box2 input[name='salary']").val().trim();  //修改员工信息不能为空  if(id == "" || name == "" || email == "" || salary == ""){  alert("修改的信息不能为空!");  return;  }    //根据ID修改对应的信息  var flag = true;  $("table tr").each(function(){  //判断id是否等于每行第二个td内的文本, 即id  if($(this).find("td:eq(1)").text() == id){  //进行修改  $(this).find("td:eq(2)").text(name);  $(this).find("td:eq(3)").text(email);  $(this).find("td:eq(4)").text(salary);  flag = false;  return false;  }  })  if(flag){  alert("修改的员工ID不存在！");  }  });    }) |

## 5.7.JSON

**json是什么?**

json是JavaScript提供的一种轻量级数据交换(存储)格式

创建JS对象:

var person = {

"name":"张飞",

"age": 18,

"gender": "男"

}

**json语法**

{}括起来的就是一个对象

\* json的属性名只能用双引号引起来, 不能用单引号!

\* json的属性值:

数字（整数或浮点数）

字符串（在双引号中）

逻辑值（true 或 false）

数组（在方括号中）

对象（在花括号中）

null

**json和xml比较**

\* 可读性: xml比json具有更好的可读性

\* 解析度: xml解析麻烦, 而json是js提供的对象, 可以非常方便的进行解析

\* 流行度: xml虽然发展了很多年, 但是由于json易于操作, 所以在某些场景中json比xml更常用.

**json格式:**

**格式一:**

var p1 = {

"name" : "张三",

"age" : 18,

"gender" : "男"

}

**格式二**

var p2 = {

"name" : "张三",

"age" : 18,

"gender" : "男",

"friends" : ["李四","王五","赵六"],

"gf" : {

"name" : "小华",

"age" : "17",

"gender" : "女"

}

}

**格式三**

var p3 = {

"name" : "张三",

"age" : 18,

"gender" : "男",

"friends" : [

{

"name" : "李四",

"age" : 20,

"gender" : "男"

},{

"name" : "王五",

"age" : 19,

"gender" : "男"

},{

"name" : "赵六",

"age" : 21,

"gender" : "男"

}

]

}

## 5.8.作业练习

1. jQuery是什么？

答：jQuery就是javascript的一个库，把我们常用的一些功能进行了封装，方便我们来调用，提高我们的开发效率，极大地简化了 JavaScript 编程

1. 为什么要使用jQuery（或者jQuery的好处）?

答：因为jQuery是轻量级的框架，拥有强大的选择器，出色的DOM操作的封装，出色的浏览器兼容性，完善的Ajax，完善的文档，开源等。

1. jQuery选择器有哪些？

答：选择器大致分为:基本选择器，层次选择器，过滤选择器，表单选择器，可以根据情况适当说明一些选择器。

1. 说说你对json的理解

答：JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。它基于标准JavaScript 的一个子集,是一个 Js 对象或数组结构的字符串

JSON 有三类数据

1. 单个数据

有 number, string, boolean 和 null 四种类型数据

2. 多个有序的数据: 数组

用[]包含起来, 其元素可以是三类数据中的任意一种, 元素之间用,号隔开

3. 多个无序的数据: 对象

用{}包含起来, 其元素必须由 key-value 组成, key 是一个字符串, value 可以是

任意类型数据, key 与 value 之间用:号隔开, 两个 key-value 之间用,号隔开.