

jQuery简介

1. 介绍
2. 使用
 - 1) 引入
 - 2) 工厂函数 - `$()`
 - 3) 原生JavaScript对象与jQuery对象
 - 4) jQuery获取元素
 - 5) 操作元素内容
 - 6) 操作标签属性
 - 7) 操作标签样式
 - 8) 元素的创建,添加,删除
 - 9) 动画效果
 - 10) 数据与对象遍历
 - 11) jQuery事件处理
- 3.实战
 1. 页面效果
 2. 代码分析
 1. 页面元素
 2. 初始代码
 3. 绑定省份
 4. 绑定城市

jQuery简介

1. 介绍

jQuery是JavaScript的工具库，对原生JavaScript中的DOM操作、事件处理、包括数据处理和Ajax技术等
进行封装,提供更完善，更便捷的方法。

2. 使用

1) 引入

先引入jquery文件，才能使用jquery语法

1. CDN 有网（备用）
2. 本地文件（常用）

2) 工厂函数 - `$()`

"`$()`"函数用于获取元素节点，创建元素节点或将原生JavaScript对象转换为jquery对象,返回 jQuery 对象。
jQuery 对象实际是一个类数组对象，包含了一系列 jQuery 操作的方法。
例如：

```
//$( )获取元素节点,需传入字符串的选择器
$("h1")
$("#d1")
$(".c1")
$("body,h1,p")
//选择器的特点, 与样式选择器一致
```

3) 原生JavaScript对象与jQuery对象

原生JavaScript对象与jQuery对象的属性和方法不能混用。可以根据需要, 互相转换:

1. 原生JavaScript转换jQuery对象

\$(原生对象), 返回 jQuery 对象

2. jQuery对象转换原生JavaScript对象

- 方法一: 根据下标取元素,取出即为原生对象

var div = \$("div")[0];

- 方法二: 使用jQuery的get(index)取原生对象

var div2 = \$("div").get(0);

```
<div id="d1"></div>
<!-- 引入外部的jQuery文件 -->
<script src="js/jquery.min.js"></script>
<script>
    var d1 = document.getElementById('d1');
    console.log(d1); //DOM 对象
    //$( ) 工厂函数
    var $d1 = $('#d1');
    console.log($d1); //jQuery对象
    var $div = $('div'); //通过元素选择器查找元素
    console.log($div);

    //把DOM对象变成jQuery对象
    console.log($(d1)); //jQuery对象
    //将jQuery对象变成DOM对象
    console.log($d1[0]); //DOM对象
    console.log($d1.get(0));
</script>
```

4) jQuery获取元素

jQuery通过选择器获取元素, \$("选择器")

选择器分类:

1. 基础选择器

```
标签选择器: $("div")
ID 选择器: $("#d1")
类选择器: $(".c1")
群组选择器: $("body,p,h1")
```

2. 层级选择器

后代选择器: `$("div .c1")`
子代选择器: `$("div>span")`
相邻兄弟选择器: `$("h1+p")` 匹配选择器1后的第一个兄弟元素,同时要求兄弟元素满足选择器2
通用兄弟选择器: `$("h1~h2")` 匹配选择器1后所有满足选择器2的兄弟元素

```
<body>
  <ul id="parent">
    <li class="active"></li>
    <li></li>
    <li></li>
    <li></li>
    <li></li>
  </ul>
  <script src="/home/tarena/桌面/month03_web/JS/day05/jquery.min.js">
</script>
  <script>
    $(".active+li").css("background-color","red");
    $(".active~li").css("background-color","red");
  </script>
</body>
```

3. 过滤选择器, 需要结合其他选择器使用。

`:first`
匹配第一个元素 例:`$("p:first")`
`:last`
匹配最后一个元素 例:`$("p:last")`
`:odd`
匹配奇数下标对应的元素
`:even`
匹配偶数下标对应的元素
`:eq(index)`
匹配指定下标的元素
`:lt(index)`
匹配下标小于index的元素
`:gt(index)`
匹配下标大于index的元素
`:not(选择器)`
否定筛选,除()中选择器外,其他元素

```
<body>
  <ul id="parent">
    <li class="active"></li>
    <li></li>
    <li></li>
    <li></li>
    <li></li>
  </ul>
  <script src="/home/tarena/桌面/month03_web/JS/day05/jquery.min.js">
</script>
  <script>
    // JSON对象
    $("#parent>li").css({
      "width": "100%",
      "height": "50px",
      "background-color": "lightblue",
      "border": "1px solid red"
    })
```

```

$("li:first").css("background-color","red");
$("li:not(.active)").css('background-color','red');
</script>
</body>

```

5) 操作元素内容

```

html() //设置或读取标签内容,等价于原生innerHTML,可识别标签语法
text() //设置或读取标签内容,等价于innerText,不能识别标签
val() //设置或读取表单元素的值,等价于原生value属性

```

```

<input type="text" id="uname">
<button id="btn">show</button>
<div id="show"></div>
<script src="js/jquery.min.js"></script>
<script>
    //获取文本框的值
    // var $val = $('#uname').val();
    //当按钮btn单击时
    //$('#btn')是jQuery对象 没有onclick
    //需要把$('#btn')转换成$('#btn')[0] 变成DOM对象
    $('#btn')[0].onclick = function(){
        //将文本框的值放在div中显示
        //div.innerHTML = xxx
        $('#show').html($('#uname').val());
    }
</script>

```

6) 操作标签属性

1. attr("attrName","value")

设置或读取标签属性

2. prop("attrName","value")

设置或读取标签属性

注意:在设置或读取元素属性时,attr()和prop()基本没有区别;但是在读取或设置表单元素(按钮)的选中状态时,必须用prop()方法,attr()不会监听按钮选中状态的改变,只看标签属性checked是否书写

3. removeAttr("attrName")

移除指定属性

```

<form action="#">
    <div>
        用户名 <input type="text" id="uname" name="uname">
    </div>
    <div>
        密码 <input type="password" id="upwd" name="upwd">
    </div>
    <div>
        <input type="checkbox" id="isSave" name="isSave">
        记住密码
    </div>
    <input type="submit" >

```

```

</form>
<script src="js/jquery.min.js"></script>
<script>
    //获取复选框的checked属性
    console.log($('#isSave').attr('checked')); //undefined
    console.log($('#isSave').prop('checked')); //false

    //查看页面最后一个input控件
    console.log($('#input:last').attr('value'));
    $('#input:last').removeAttr('value');
    console.log($('#input:last').prop('value'));

    //查看密码框的name属性
    console.log($('#upwd').attr('name'));
    //将密码框的name属性改成password
    $('#upwd').attr('name', 'password');
    console.log($('#upwd').prop('name'));

</script>

```

7) 操作标签样式

1. 为元素添加id/class属性,对应选择器样式
2. 针对类选择器,提供操作class属性值的方法

```

addClass("className")    //添加指定的类名
removeClass("className") //移除指定的类型,如果参数省略,表示清空class属性值
toggleClass("className") //结合用户行为,实现动态切换类名.如果当前元素存在指定类名,则移除;
                        不存在则添加

```

3. 操作行内样式

```

css("属性名","属性值") //设置行内样式
css(JavaScriptON对象) //设置一组CSS样式
/*
JavaScriptON对象:常用数据传输格式
语法 :
{
    "width":"200px",
    "height":"200px",
    "color":"red"
}
*/

//完整代码如下:
<body>
    <ul id="parent">
        <li class="active"></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
    </ul>
    <script src="/home/tarena/桌面/month03_web/JS/day05/jquery.min.js">
</script>
    <script>
        // JSON对象

```

```

$( "#parent>li" ).css({
    "width": "100%",
    "height": "50px",
    "background-color": "lightblue",
    "border": "1px solid red"
});
</script>
</body>

```

8) 元素的创建,添加,删除

1. 创建：使用\$("标签语法"), 返回创建好的元素

```

var div = $("<div></div>"); //创建元素
div.html("动态创建").attr("id", "d1").css("color", "red"); //链式调用, 设置内容和属性
var h1 = $("<h1 id='d1'>一级标题</h1>"); //创建的同时设置内容, 属性和样式

```

```

<body>
  <div id="content" style="border:1px solid blue">
    <h3>div中原本的内容</h3>
  </div>
  <script src="js/jquery.min.js"></script>
  <script>
    var d1 = $('<div></div>');
    d1.html('动态创建的div中的内容').css('color', 'red');
    console.log(d1[0]);
    //向content添加d1 作为子元素添加
    $('#content').append(d1); //在末尾添加
    // $('#content').prepend(d1); //在第一个子元素位置添加(开头)

    //作为content兄弟元素添加
    // $('#content').after(d1); //在后面

    //移除页面元素
    $('h3').remove();
  </script>
</body>

```

1. 作为子元素添加

```

$obj.append(newObj); //在$obj的末尾添加子元素newObj
$obj.prepend(newObj); //作为第一个子元素添加至$obj中

举个栗子:
$("#content").append(d1);
$("#content").prepend(d1);

```

3. 作为兄弟元素添加

```

$obj.after(newObj); //在$obj的后面添加兄弟元素
$obj.before(newObj); //在$obj的前面添加兄弟元素

```

4. 移除元素

```
$obj.remove(); //移除$obj
```

```
<script src="/home/tarena/桌面/month03_web/JS/day05/jquery.min.js"></script>
<script>
    window.onload=function(){
        var lis=document.getElementsByClassName("item");
        for(var i=0;i<lis.length;i++){
            lis[i].onclick=function(){

                $(this).addClass("active").siblings().removeClass("active");
            }
        }
    }
</script>
```

9) 动画效果

1. 显示和隐藏

```
show(speed,callback)/hide(speed,callback)
```

- speed 可选。规定元素从隐藏到完全可见的速度。默认为 "0"。
- callback 可选。show 函数执行完之后，要执行的函数

```
$("img").hide(1500,function(){$(".btn").html("显示")}) //ms为单位
```

2. 通过使用滑动下拉和上推效果，显示隐藏的被选元素（只针对块元素）

```
slideDown(speed,callback)/slideUp(speed,callback)
```

3. 通过使用淡隐淡现方式显示效果，显示隐藏的被选元素

```
fadeOut(speed,callback)/fadeIn(speed,callback)
```

4. 动画函数，可以实现自定义动画 animate 函数

```
animate(styles,speed,callback)
```

- styles 必需。规定产生动画效果的 CSS 样式和值
- speed 可选。规定动画的速度。默认是 "normal"
- callback 可选。show 函数执行完之后，要执行的函数

```
<style>
    #ball{
        width: 50px;
        height: 50px;
        border-radius: 50%;
        background-color: orange;
        position: absolute;
        top: 5px;
        left: 5px;
```

```

    }
</style>
</head>
<body>
    <div id="ball"></div>
    <script src="js/jquery.min.js"></script>
    <script>
        $('#ball').animate({'left':'300px'},1000,function(){
            $('#ball').animate({'left':'5px'},1000,function(){
                $('#ball').fadeOut(1000);
            })
        });
    </script>
</body>

```

```

<body>
    <div id="imgs">
        
    </div>
    <button class="btn">隐藏</button>
    <button class="btn2">上推</button>
    <button class="btn3">淡隐</button>

    <script src="js/jquery.min.js"></script>
    <script>
        $('.btn').on('click',function(){
            //判断按钮的内容 如果是隐藏 就调用hide()
            //否则调用show()
            if($('.btn').html() == '隐藏'){
                //点击按钮 1.5s内隐藏图片
                $('#img').hide(1500,function(){
                    //当图片完全隐藏后执行此函数
                    //将按钮内容改为显示
                    $('.btn').html('显示');
                });
            }else{
                $('#img').show(1500,function(){
                    //当图片完全显示后执行此函数
                    //将按钮内容改为隐藏
                    $('.btn').html('隐藏');
                });
            }
        })

        $('.btn2').bind('click',function(){
            if($('.btn2').html() == '上推'){
                $('#imgs').slideUp(1500,function(){
                    $('.btn2').html('下拉');
                })
            }else{
                $('#imgs').slideDown(1500,function(){
                    $('.btn2').html('上推');
                })
            }
        })

        $('.btn3').bind('click',function(){
            if($('.btn3').html() == '淡隐'){

```



```

        $('#imgs').fadeOut(1500,function(){
            $('.btn3').html('淡显');
        })
    }else{
        $('#imgs').fadeIn(1500,function(){
            $('.btn3').html('淡隐');
        })
    }
}
})
</script>
</body>

```

10) 数据与对象遍历

1. \$(selector).each() 方法规定为每个匹配元素规定运行的函数

```
$(selector).each(function(index,element){})
```

必需。为每个匹配元素规定运行的函数。

- index - 选择器的 index 位置
- element - 当前的元素

2. \$.each()函数是框架提供的一个工具类函数，通过它，你可以遍历对象、数组的属性值并进行处理

```
$.each(Object, function(index, data){});
```

必需。为每个匹配元素规定运行的函数。

- index - 选择器的 index 位置
- data- 当前的数据

```

<title>Document</title>
<script src="js/jquery.min.js"></script>
<script>
    $(function(){
        $('#nav>li').each(function(index,element){
            //index 表示索引值
            //element 表示当前的元素
            // console.log($('#nav>li')[index]);
            console.log(element);
        })
        var arr = [1,2,3,4,5];
        $.each(arr,function(i,e){
            // console.log(arr[i]);
            // console.log(e);
        })
    })
</script>
</head>
<body>
    <ul id="nav">
        <li></li>
        <li></li>
        <li></li>
        <li></li>
    </ul>
</body>

```

11) jQuery事件处理

1. 文档加载完毕

原生JavaScript 方法: window.onload

jQuery:

```
//语法一
$(document).ready(function (){
    //文档加载完毕后执行
})
//语法二
$.ready(function (){
    //文档加载完毕后执行
})
//语法三
$(function(){
    //文档加载完毕后执行
})
```

区别:

原生onload事件不能重复书写, 会产生覆盖问题; jquery中对事件做了优化,可以重复书写ready方法,依次执行

2. 事件绑定方式

事件名称省略 on 前缀

```
//on("事件名称", function)
$("#div").on("click",function(){}); //新版本使用的多些
//bind("事件名称",function)
$("#div").bind("click",function(){}); //1.6-1.8间的版本
//事件名作为方法名
$("#div").click(function(){});
```

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Document</title>
  <style>
    #nav{
      margin: 0;
      padding: 0;
      list-style: none;
      border: 1px solid #aaa;
      overflow: hidden;
      position: fixed;
      top: 0;
    }
    #nav li {
      float: left;
    }
    #nav>.first{
      color: #aaa;
      margin: 12px 10px;
    }
  </style>
```

```

#nav>.item{
    margin: 10px;
    padding: 2px 10px;
    transition: all 0.3s;
}
#nav>.item:hover{
    cursor: pointer;
}
.active{
    background-color: #f00;
    color: #fff;
}
</style>
<script src="js/jquery.min.js"></script>
<script>
    $(function(){
        $('.item').each(function(i,e){
            //e 表示当前循环的元素 是DOM对象
            //需要转为jQuery对象才能调用click()
            $(e).click(function(){

                $(this).addClass('active').siblings().removeClass('active');
            })
        })
    })
</script>
</head>
<body>
    <ul id="nav">
        <li class="first">难度:</li>
        <li class="item active">全部</li>
        <li class="item">初级</li>
        <li class="item">中级</li>
        <li class="item">高级</li>
    </ul>
</body>

```

1. this表示事件的触发对象，在jquery中可以使用，注意转换类型。this为原生对象只能使用原生的属性和方法，可以使用\$(this)转换为jquery对象，使用jquery方法。

3.实战

1. 页面效果

请选择 ▼	请选择 ▼	请选择 ▼
请选择		
浙江省		
山东省		
广东省		
甘肃省		

山东省 ▼	请选择 ▼	请选择 ▼
	请选择	
	济南市	
	青岛市	
	济宁市	
	潍坊市	

山东省 ▼	青岛市 ▼	青岛一区 ▼
		请选择
		青岛一区
		青岛二区

2. 代码分析

1. 页面元素

```
<form action="#">
  <select name="prov" id="prov">
  </select>
  <select name="city" id="city">
  </select>
  <select name="area" id="area">
  </select>
</form>
```

2. 初始代码

```
var $prov = $("#prov");
var $city = $("#city");
var $area = $("#area");
$(function () {
  $prov.html("<option value='0'>请选择</option>")
  $city.html("<option value='0'>请选择</option>")
  $area.html("<option value='0'>请选择</option>")
  $.each(data, function (i, e) {
    $prov.append("<option value='" + e.provId + "'" + e.provname + "</option>")
  })
})
```

3. 绑定省份

```

$prov.on("change", function () {
    $.each(data, function (i, e) {
        if ($prov.val() == e.provId) {
            $city.html("<option value='0'>请选择</option>");
            $.each(e.citys, function (i, e2) {
                $city.append("<option value='" + e2.cityId + "'>"
                    + e2.cityname +
                    "</option>")
            })
        }
    })
    if ($prov.val() == 0) {
        $city.html("<option value='0'>请选择</option>");
    }
    if ($city.val() == 0) {
        $area.html("<option value='0'>请选择</option>");
    }
})

```

4. 绑定城市

```

$city.on("change", function () {
    $.each(data, function (i, e) {
        if ($prov.val() == e.provId) {
            $.each(e.citys, function (i, e2) {
                if ($city.val() == e2.cityId) {
                    $area.html("");
                    $.each(e2.areas, function (i, e3) {
                        $area.append("<option value='"
                            + e3.areaId + "'>" + e3
                            .areaname + "</option>")
                    })
                }
            })
        }
    })
    if ($city.val() == 0) {
        $area.html("<option value='0'>请选择</option>");
    }
})

```