

# jQuery

jQuery下载

针对导入问题

jQuery基本语法

查找标签

基本选择器

组合选择器/分组与嵌套

基本筛选器

属性选择器

form表单筛选器

筛选器方法

操作标签

操作类

css操作

位置操作

尺寸

文本操作

获取值操作

属性操作

文档处理

jQuery事件

jQuery绑定事件的方式

克隆事件

自定义模态框

左侧菜单

返回顶部

自定义登录校验

input框实时监控

hover事件

键盘按键按下事件

阻止后续事件执行

阻止事件冒泡

事件委托

页面加载

动画效果

补充

```
1  ""
2  jQuery内部封装了原生的js代码，还添加了很多的功能
3  能够通过书写更少的代码，完成js操作
4  类似于Python里的模块，在前端模块不叫模块，叫“类库”
5
6  兼容多个浏览器，使用jQuery就不需要考虑兼容问题
7
8  jQuery的宗旨：
9      write less do more
10     让你用更少的代码完成更多的事情
11
```

```
12 | python中导入模块需要消耗资源，
13 | jquery在使用的时候也需要导入，但是它的文件特别的小，基本不影响速度
14 | """
```

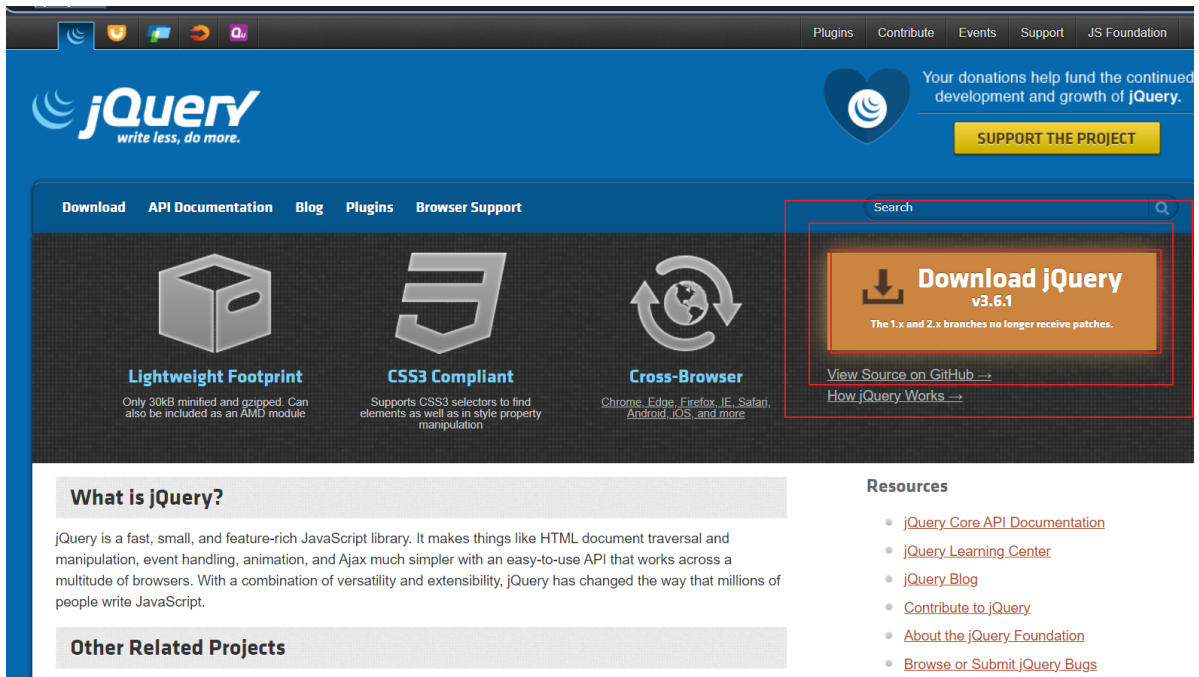
[菜鸟教程](#)

[jQuery API中文文档](#)

[BootCDN](#)

## jQuery下载

[官网](#)



To locally download these files, right-click the link and select "Save as..." from the m

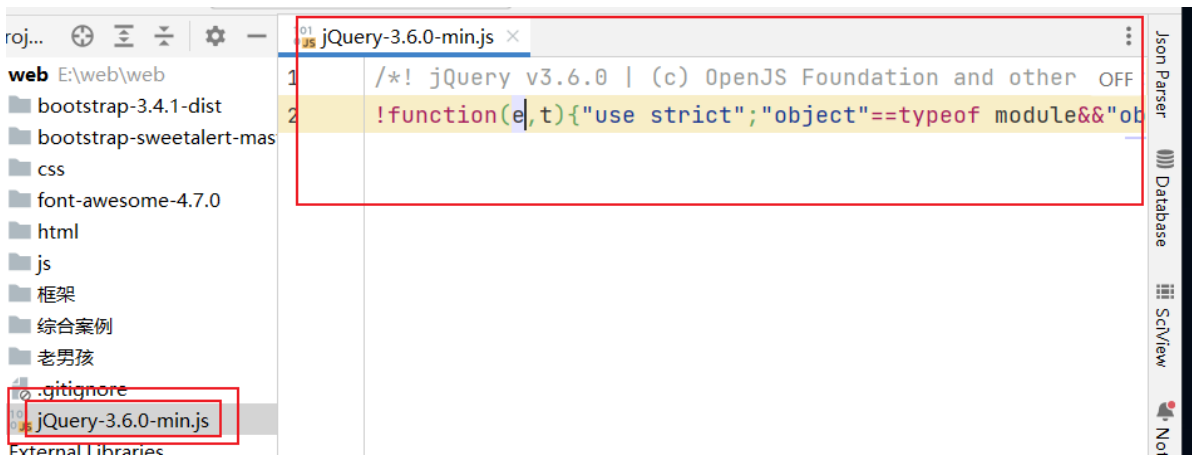
### jQuery

For help when upgrading jQuery, please see the [upgrade guide](#) most relevant to you [plugin](#).

Download the compressed, production jQuery 3.6.1 ← 压缩过的

[Download the uncompressed, development jQuery 3.6.1](#) ← 未压缩过的

- 打开压缩过的链接直接 **Ctrl+C** 复制，新建一个 **JS文件** 直接 **Ctrl+V** 进去就可以

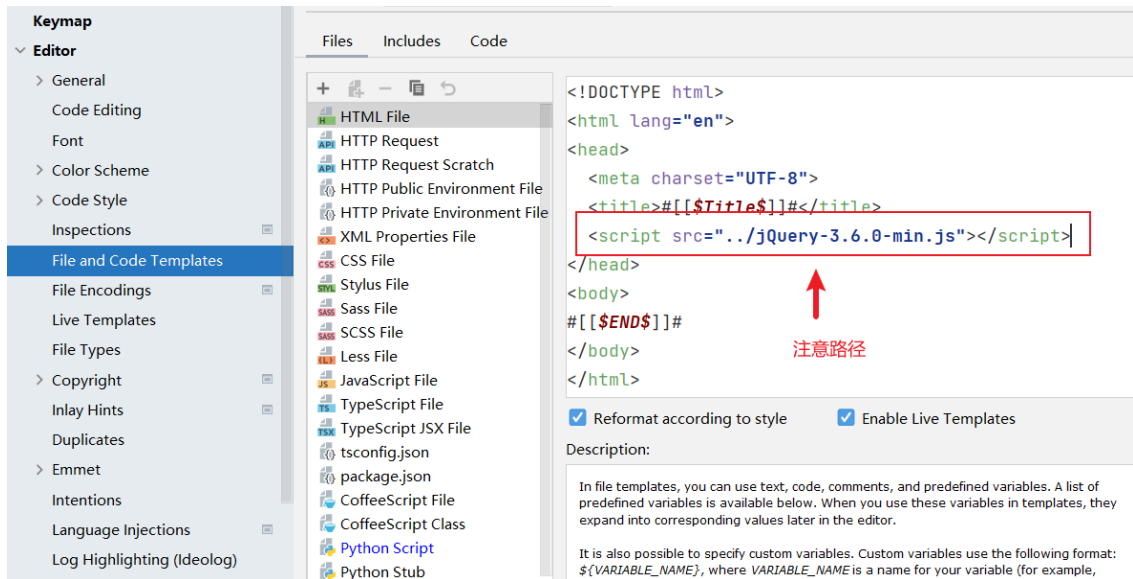


jQuery在使用前一定要确保已经导入了

## 针对导入问题

1. 文件下载到了本地，如何解决多个文件反复书写引入语句的代码

- 可以借助Pycharm自动初始化代码功能，完成自动添加



2. 如果不下载jQuery文件，可以直接引用 jQuery 提供的CDN服务（基于网络直接请求加载）

- CDN:内容分发网络
- CDN有免费的有收费的

- 前端免费的CDN网站: [BootCDN](#)



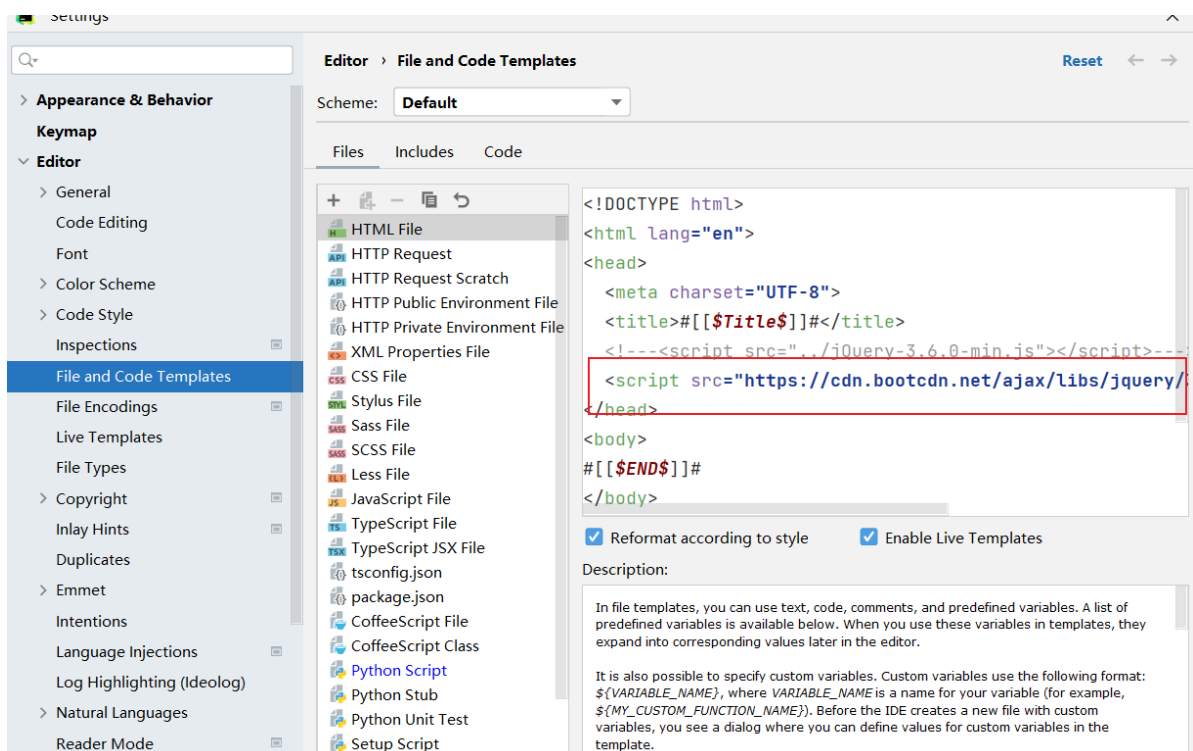
点击Query

直接复制即可，但是要保证计算机有网络，才可以使用。

版本: 3.6.1

<a href="https://cdn.bootcdn.net/ajax/libs/jquery/3.6.1/jquery.js">https://cdn.bootcdn.net/ajax/libs/jquery/3.6.1/jquery.js</a>	没有压缩的	复制 <script> 标签	复制链接
<a href="https://cdn.bootcdn.net/ajax/libs/jquery/3.6.1/jquery.min.js">https://cdn.bootcdn.net/ajax/libs/jquery/3.6.1/jquery.min.js</a>	压缩过后的，通过min区分是否压缩	复制到剪贴板	复制链接
<a href="https://cdn.bootcdn.net/ajax/libs/jquery/3.6.1/jquery.slim.js">https://cdn.bootcdn.net/ajax/libs/jquery/3.6.1/jquery.slim.js</a>		复制 <script> 标签	复制链接
<a href="https://cdn.bootcdn.net/ajax/libs/jquery/3.6.1/jquery.slim.min.js">https://cdn.bootcdn.net/ajax/libs/jquery/3.6.1/jquery.slim.min.js</a>		复制 <script> 标签	复制链接

```
1 <scriptsrc="https://cdn.bootcdn.net/ajax/libs/jquery/3.6.0/jquery.min.js">
  </script>
```



# jQuery基本语法

```
1  jQuery(选择器).action()
2  秉持着jQuery的宗旨，jQuery简写成$
3  jQuery()  等价于  $(())
4
5  #jQuery与js代码对比
6  eg:将p标签内部的文本颜色改为红色
7  let pEle=document.getElementById('p1')
8  pEle.style.color='red'
9  'red'
10 #jQuery操作标签
11 $('p').css('color','blue')
```

## 查找标签

### 基本选择器

```
1  //id选择器
2  $('#d1')
3  S.fn.init [div#d1]0: div#d1length: 1[[Prototype]]: Object(0)
4
5  //class选择器
6  $('.c1')
7  S.fn.init [p.c1, prevObject: S.fn.init(1)]0: p.c1length: 1prevObject:
  S.fn.init [document] [[Prototype]]: Object(0)
8
9  //标签选择器
10 $('span')
11 S.fn.init(3) [span, span, span, prevObject: S.fn.init(1)]0: span1: span2:
  spanlength: 3prevObject: S.fn.init [document] [[Prototype]]: Object(0)
12
13 //jQuery对象如何变成标签对象
14 $('#d1')[0]
15 <div id="d1">...</div>
16 document.getElementById('d1')
17 <div id="d1">...</div>
18
19 //标签对象变jQuery对象
20 $(document.getElementById('d1'))
21 S.fn.init [div#d1]
22 $(document.getElementById('d1'))[0] //又转成了标签对象
23 <div id="d1">...</div>
```

### 组合选择器/分组与嵌套

```
1  $('div')
2  S.fn.init(2) [div#d1, div.c1, prevObject: S.fn.init(1)]
3
4  $('div.c1')
5  S.fn.init [div.c1, prevObject: S.fn.init(1)]0: div.c1length: 1prevObject:
  S.fn.init [document] [[Prototype]]: Object(0)
6
7  $('div#d1')
```

```

8 S.fn.init [div#d1, prevObject: S.fn.init(1)]0: div#d1length: 1prevObject:
  S.fn.init [document][[Prototype]]: Object(0)
9
10 $('*')
11 S.fn.init(18) [html, head, meta, title, script, style, body, span, span,
  div#d1, span, p, span, span, div.c1, span, span, script, prevObject:
  S.fn.init(1)]
12
13 $('#d1,.c1,p') #并列+混用
14 S.fn.init(3) [div#d1, p, div.c1, prevObject: S.fn.init(1)]
15
16 ('div span') #后代
17 S.fn.init(3) [span, span, span, prevObject: S.fn.init(1)]
18
19 ('div>span')#儿子
20 S.fn.init(2) [span, span, prevObject: S.fn.init(1)]0: span1: spanlength:
  2prevObject: S.fn.init [document][[Prototype]]: Object(0)
21
22 ('div+span')#紧挨着的第一个（毗邻）
23 S.fn.init [span, prevObject: S.fn.init(1)]
24
25 ('div~span') #兄弟
26 S.fn.init(2) [span, span, prevObject: S.fn.init(1)]

```

## 基本筛选器

```

1 $('ul li') #ul下所有li
2 S.fn.init(10) [li, li, li, li, li, li, li.c1, li, li, li#d1, prevObject:
  S.fn.init(1)]
3
4 $('ul li:first') #大儿子
5 S.fn.init [li, prevObject: S.fn.init(1)]0: li1length: 1prevObject: S.fn.init
  [document][[Prototype]]: Object(0)
6
7 $('ul li:last')#小儿子
8 S.fn.init [li#d1, prevObject: S.fn.init(1)]0: li#d1length: 1prevObject:
  S.fn.init [document][[Prototype]]: Object(0)
9
10 $('ul li:eq(2)')#索引为2的
11 S.fn.init [document][[Prototype]]: Object(0)
12
13 $('ul li:even') #偶数
14 S.fn.init(5) [li, li, li, li.c1, li, prevObject: S.fn.init(1)]0: li1: li2:
  li3: li.c14: li1length: 5prevObject: S.fn.init [document][[Prototype]]:
  Object(0)
15
16 $('ul li:odd')#奇数
17 S.fn.init(5) [li, li, li, li, li#d1, prevObject: S.fn.init(1)]0: li1: li2:
  li3: li4: li#d1length: 5prevObject: S.fn.init [document][[Prototype]]:
  Object(0)
18
19 $('ul li:gt(2)')#索引大于2的
20 S.fn.init(7) [li, li, li, li.c1, li, li, li#d1, prevObject: S.fn.init(1)]
21
22 $('ul li:lt(2)')#索引小于2的

```

```

23 s.fn.init(2) [li, li, prevObject: s.fn.init(1)]
24
25 $('ul li:not("#d1")')#ul下所有li, 除了id为d1的
26 s.fn.init(9) [li, li, li, li, li, li, li.c1, li, li, prevObject:
s.fn.init(1)]
27
28 $('div:has("p")')#选取包含一个或多个标签在内的标签, 简单将就是div里面有p的div
29
29 s.fn.init [div, prevObject: s.fn.init(1)]

```

## 属性选择器

```

1  $('[username]')
2  <input type="text" id username="zhangsan">
3
4  $('[username="lisi"]')
5  <p username="lisi"></p>
6
7  $('p[username]')
8  <p username="lisi"></p>
9
10 $('[type]')
11
12 $('[type="text"]')

```

## form表单筛选器

```

1  $('input[type="text"]')
2  s.fn.init [input, prevObject: s.fn.init(1)]
3
4  $('input[type="password"]')
5  s.fn.init [input, prevObject: s.fn.init(1)]
6
7  $(':password') #等价于上面第二个
8  s.fn.init [input, prevObject: s.fn.init(1)]
9
10 $(':text') ##等价于上面第一个
11 s.fn.init [input, prevObject: s.fn.init(1)]
12
13 :text
14 :password
15 :file
16 :radio
17 :checkbox
18 :submit
19 :reset
20 :button
21 ...
22 表单的对象属性
23 :enabled
24 :disabled
25 :checked
26 :selected
27
28

```

```

29 $(':checked') #它会将checked 和selected都拿到
30 s.fn.init(2) [input, option, prevObject: s.fn.init(1)]0: input1:
optionlength: 2prevObject: s.fn.init [document][[Prototype]]: Object(0)
31
32 $(':checked')[0]
33 <input type="checkbox" value="222" checked>
34
35 $(':checked')[1]
36 <option value selected>333</option> slot
37
38 $(':selected') #它不会, 只拿selected
39 s.fn.init [option, prevObject: s.fn.init(1)]
40
41 $(':selected')[0]
42 <option value selected>333</option> slot
43
44 $('input:selected') #自己加一个限制条件
45 s.fn.init [prevObject: s.fn.init(1)]

```

## 筛选器方法

```

1 $('#d1').next() #同级别下一个
2
3 $('#d1').nextAll()
4
5
6 $('#d1').nextUntil('.c1') #下面所有, 直到.c1, 不包括.c1
7
8
9 $('.c1').prev() #上一个
10
11
12 $('.c1').prevAll()
13
14
15 $('.c1').prevUntil('#d2')
16
17
18 $('#d3').parent() #第一级父标签
19
20 $('#d3').parent().parent()
21
22
23 $('#d3').parent().parent()
24
25
26 $('#d3').parent().parent().parent()
27
28
29 $('#d3').parent().parent().parent().parent()
30
31
32 $('#d3').parent().parent().parent().parent().parent()
33
34

```



```

35 $('#d3').parents() #所有父标签
36 $('#d3').parentsUntil('body')
37
38 $('#d2').children()#儿子
39 s.fn.init(3) [span, p, span, prevObject: s.fn.init(1)]
40
41 $('#d2').siblings()#同级别上下所有
42
43 $('div p')等价于 $('div').find('p')
44 #find从某个区域内部筛选出想要的标签
45
46 #等价
47 $('div span:first')
48 $('div span').first()
49 $('div span:last')
50 $('div span').last()
51 $('div span:not("#d3")')
52 $('div span').not('#d3')

```

## 操作标签

### 操作类

```

1 #操作类
2 """
3 js版本                                jQuery版本
4 classList.add()                        addClass()
5 classList.remove()                    removeClass()
6 classList.contains()                  hasClass()
7 classList.toggle()                    toggleClass()
8 """
9

```

### css操作

```

1 #css操作
2 <p>111</p>
3 <p>222</p>
4 '''一行代码将第一个p标签变成红色，第二个p标签变成蓝色'''
5 $('p').first().css('color','red').next().css('color','blue')
6 #jQuery链式操作，使用jQuery可以做到一行代码操作很多标签
7 #jQuery对象调用jQuery方法之后返回的还是一个jQuery对象，也就可以继续调用
8
9 class MyClass(object):
10     def func1(self):
11         print('func1')
12         return self
13     def func2(self):
14         print('func2')
15         return self
16
17 obj=MyClass()
18 obj.func1().func2()

```

## 位置操作

```
1 #位置操作
2 offset()
3     $('p').offset()
4     {top: 100, left: 100}
5
6 position()
7     $('p').position()
8     {top: 100, left: 100}
9
10 scrollTop()
11     $(window).scrollTop() #括号内不加参数就是不获取
12     $(window).scrollTop(500)#加了参数就是设置
13
14 scrollLeft()
```

## 尺寸

```
1 #尺寸
2 $('p').height() #文本
3 20.8
4 $('p').width()
5 866
6 $('p').innerheight() #文本+padding
7 $('p').innerwidth()
8 $('p').outheight() #文本+padding+boder
9 $('p').outwidth()
```

## 文本操作

```
1 """
2 操作标签内部文本
3 js版本      jQuery版本
4 innerText    text()    括号内不加参数就是获取，加了参数就是设置
5 innerHTML    html()    括号内不加参数就是获取，加了参数就是设置
6 """
7 $('div').text()
8 '\n      独步江湖，东方不败\n    '
9 $('div').html()
10 '\n      <p> 独步江湖，东方不败</p>\n    '
11 $('div').text('人生苦短，我学Python')
12 s.fn.init [div, prevObject: s.fn.init(1)]
13 $('div').html('人生路，冷冷的夜光')
14 s.fn.init [div, prevObject: s.fn.init(1)]
15 $('div').text('<h1>人生苦短，我学Python</h1>')
16 s.fn.init [div, prevObject: s.fn.init(1)]
17 $('div').html('<h1>人生苦短，我学Python</h1>')
18 s.fn.init [div, prevObject: s.fn.init(1)]
```

## 获取值操作

```
1  ""
2  js      jQuery
3  .value  .val()
4  ""
5
6  $('#d1').val()
7  'vfdghjmhnbv'
8  $('#d1').val('520') #括号内不加参数是获取，加了参数是设置
9  s.fn.init [input#d1]
10 $('#d2')[0].files[0] #两个对象之间的转换
11
12 $(':checkbox').val(666)#所有的value值都改变成666
13
```

## 属性操作

```
1  ""
2  js      jQuery
3  setAttribute()  attr(name,value)
4  getAttribute()  attr(name)
5  removeAttribute()  removeAttr(name)
6
7
8  在用变量存储对象的时候，js推荐使用xxxEle    （标签对象）
9  在jQuery中推荐使用$xxxEle                    （jQuery对象）
10 ""
11 let $pEle = $('p')
12
13 $pEle.attr('id')
14
15 $pEle.attr('class')
16
17 $pEle.attr('class','c1')
18
19 $pEle.attr('id','666')
20
21 $pEle.attr('password','456666')
22
23 $pEle.removeAttr('password','456666')
24
25
26
27 ""对于标签上有的能够看到的属性和自定义属性用attr
28 对于返回布尔值比如checkbox,radio,option是否被选中用prop""
29 $('#i2').prop('checked')
30 true
31 $('#i3').prop('checked',true)
32 s.fn.init [input#i3]
33 $('#i3').prop('checked',false)
34 s.fn.init [input#i3]
```

## 文档处理

```
1  ""
2  js                jQuery
3  createElement('p')    $('<p>')
4
5  appendChild()        append()
6
7
8  ""
9  let $pEle = $('<p>')
10 $pEle.text('你好')
11 $pEle.attr('id', 'd6')
12
13 $('#d1').append($pEle) #和下面等价 , 内部尾部追加
14 $('#d1').append($pEle[0])
15 $pEle.appendTo($('#d1'))
16
17 $('#d1').prepend($pEle) #内部头部追加
18 $pEle.prependTo($('#d1'))
19
20 $('#d2').after($pEle) #放在某个标签后面
21 S.fn.init {}[[Prototype]]: Object(0)
22 $pEle.insertAfter($('#d2'))
23
24 $('#d1').before($pEle) #放在某个标签前面
25 $pEle.insertBefore($('#d1'))
26
27 $('#d1').remove() #将标签从DOM树中删除
28
29 $('#d1').empty() #清空标签内部所有的内容
30
31
```

## jQuery事件

### jQuery绑定事件的方式

```
1  //
2  <body>
3      <button id="d1">吻我</button>
4      <button id="d2">亲我</button>
5
6
7      <script>
8          //第一种
9          $('#d1').click(function () {
10              alert('别说话 吻我')
11          })
12
13          //第二种, 功能强大, 支持事件委托
14          $('#d2').on('click', function () {
15              alert('借钱给我, 后面还你')
16          })
17      </script>
```

## 克隆事件

```

1 <head>
2   <meta charset="UTF-8">
3   <meta http-equiv="X-UA-Compatible" content="IE=edge">
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <title>Document</title>
6   <script src="../../jQuery-3.6.0-min.js"></script>
7   <style>
8     #d1{
9       width: 100px;
10      height: 100px;
11      background-color: orange;
12      border: 1px solid blue;
13    }
14  </style>
15 </head>
16 <body>
17   <button id="d1">倚天剑，点击传送</button>
18
19
20
21
22   <script>
23     $('#d1').on('click',function(){
24       // console.log(this); //this指代的永远是当前被操作的标签对象
25       // $(this).clone().insertAfter($('body')) //clone默认只克隆html和
css, 不克隆事件
26       $(this).clone(true).insertAfter($('body')) //括号里面 加参数true即可克隆html,css, 事件
27     })
28   </script>
29 </body>

```

## 自定义模态框

```

1  /*模态框内部本质就是给标签移除或添加hide属性*/
2  <!DOCTYPE html>
3  <html lang="en">
4  <head>
5    <meta charset="UTF-8">
6    <meta http-equiv="X-UA-Compatible" content="IE=edge">
7    <meta name="viewport" content="width=device-width, initial-scale=1.0">
8    <script src="../../jQuery-3.6.0-min.js"></script>
9    <title>Document</title>
10   <style>
11     .cover{
12       position: fixed;
13       top: 0;
14       left: 0;
15       right: 0;
16       bottom: 0;
17       background-color: rgba(128,125,128,0.4);

```

```
18         z-index: 99;
19     }
20     .modal{
21         position: fixed;
22         height: 400px;
23         width: 400px;
24         background-color:white ;
25         top: 50%;
26         left:50%;
27         margin-left: -300PX;
28         margin-top: -200PX;
29         z-index: 100;
30     }
31     .hide{
32         display: none;
33     }
34 }
35 </style>
36 </head>
37 <body>
38     <div>我是最下面的</div>
39     <button id="d1">给我出来</button>
40     <div class="cover hide "></div>
41     <div class="modal hide">
42         <p>username: <input type="text" ></p>
43         <p>password:<input type="text" ></p>
44         <input type="button" value="确定" >
45         <input type="button" value="取消" id="d2">
46
47     </div>
48
49     <script>
50         let $coverEle=$('#.cover')
51         let $modalEle=$('#.modal')
52
53         $('#d1').click(function(){
54             //将两个div标签的hide属性移除
55             $coverEle.removeClass('hide')
56             $modalEle.removeClass('hide')
57
58         })
59         $('#d2').on('click',function(){
60             $coverEle.addClass('hide')
61             $modalEle.addClass('hide')
62         })
63     </script>
64
65 </body>
66 </html>
```

## 左侧菜单

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta http-equiv="X-UA-Compatible" content="IE=edge">
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8      <title>Document</title>
9      <script src="../../jQuery-3.6.0-min.js"></script>
10     <style>
11         * {
12             margin: 0;
13             padding: 0;
14
15         }
16
17         .left {
18             float: left;
19             background-color: darkgray;
20             width: 20%;
21             height: 100%;
22             position: fixed;
23         }
24
25         .title {
26             font-size: 20px;
27             color: white;
28             text-align: center;
29
30         }
31
32         .items {
33             border: 1px solid black;
34         }
35
36         .hide {
37             display: none;
38
39         }
40     </style>
41 </head>
42
43 <body>
44     <div class="left">
45         <div class="menu">
46             <div class="title">菜单一
47                 <div class="items">111</div>
48                 <div class="items">222</div>
49                 <div class="items">333</div>
50             </div>
51             <div class="title">菜单二
52                 <div class="items">111</div>
53                 <div class="items">222</div>
```

```

54         <div class="items">333</div>
55     </div>
56     <div class="title">菜单三
57         <div class="items">111</div>
58         <div class="items">222</div>
59         <div class="items">333</div>
60     </div>
61 </div>
62 </div>
63
64 <script>
65     $('<strong>.title</strong>').click(function () {
66         // 先给所有items加hide,
67         $('<strong>.items</strong>').addClass('hide')
68         //然后将被点击的标签内部的hide值移除
69         //this指代的是当前被操作的标签对象
70         $(this).children().removeClass('hide')
71     })
72 </script>
73 </body>
74
75 </html>

```

## 返回顶部

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Document</title>
8      <script src="../../jQuery-3.6.0-min.js"></script>
9      <style>
10         .hide{
11             display: none;
12         }
13         #d1{
14             position: fixed;
15             background-color: black;
16             right: 20px;
17             bottom: 20px;
18             height: 50px;
19             width: 50px;
20         }
21     </style>
22 </head>
23 <body>
24     <a href="" id="d1"></a>
25     <div style="height: 500px;background:red;"></div>
26     <div style="height: 500px;background:greenyellow;"></div>
27     <div style="height: 500px;background:blue;"></div>
28     <a href="#d1" class="hide">回到顶部</a>
29
30

```



```

31     <script>
32         $(window).scroll(function(){
33             if ($(window).scrollTop()>300) {
34                 $('#d1').removeClass('hide')
35             }else{
36                 $('#d1').addClass('hide')
37             }
38         })
39     </script>
40 </body>
41 </html>

```

## 自定义登录校验

```

1  /*获取用户的密码和用户名的时候，如果用户没有填写，应该给用户提示*/
2  <!DOCTYPE html>
3  <html lang="en">
4  <head>
5      <meta charset="UTF-8">
6      <meta http-equiv="X-UA-Compatible" content="IE=edge">
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8      <title>Document</title>
9      <script src="../../jQuery-3.6.0-min.js"></script>
10 </head>
11 <body>
12     <P>username:
13         <input type="text" id="username">
14         <span style="color: red;"></span>
15     </P>
16     <P>password:
17         <input type="text" id="password">
18         <span style="color: red;"></span>
19     </P>
20     <button id="d1">提交</button>
21
22
23     <script>
24         let $userName = $('#username')
25         let $password = $('#password')
26         $('#d1').click(function(){
27             //获取用户输入的用户名和密码，做校验.
28             let userName = $userName.val()
29             let password = $password.val()
30             if (!userName){
31                 $userName.next().text('用户名不能为空')
32             }
33             if (!password){
34                 $password.next().text('密码不能为空')
35             }
36         })
37         $('input').focus(function(){
38             $(this).next().text('')
39         })
40     </script>
41 </body>

```

```
42 | </html>
```

## input框实时监控

```
1  <body>
2      <input type="text " id="d1">
3
4      <script>
5          $('#d1').on('input',function(){
6              console.log(this.value);
7          })
8      </script>
9  </body>
```

## hover事件

```
1  <body>
2      <p id="d1">然后呢</p>
3
4      <script>
5          $('#d1').hover(
6              function () {
7                  alert('来宾几位? ') //悬浮
8              },
9              function(){
10                 alert('溜了溜了') //离开
11             }
12          )
13      </script>
14
15  </body>
```

## 键盘按键按下事件

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Document</title>
8      <script src="../../jQuery-3.6.0-min.js"></script>
9  </head>
10 <body>
11     <script>
12         $(window).keydown(function(event){
13             console.log(event.keyCode)
14             if (event.keyCode == 16){
15                 alert('您按下了shift键')
16             }
17         })
18     </script>
19 </body>
20 </html>
```

## 阻止后续事件执行

```
1 <form action="">
2     <span id="d1" style="color: red;"></span>
3     <input type="submit" id="d2">
4 </form>
5
6 <script>
7     $('#d2').click(function(e){
8         $('#d1').text('晚上好! ')
9         //阻止标签后续时间的执行 方式1
10        //return false
11        //阻止标签后续时间的执行 方式2
12        e.preventDefault()
13
14    })
15 </script>
```

## 阻止事件冒泡

```
1 <div id="d1">div
2     <p id="d2">p
3         <span id="d3">span </span>
4     </p>
5 </div>
6
7 <script>
8     $('#d1').click(function(){
9         alert('div')
10    })
11    $('#d2').click(function(){
12        alert('p')
13        return false
14    })
15    $('#d3').click(function(e){
16        alert('span')
17        //阻止事件冒泡的方式1
18        //return false
19        //阻止事件冒泡的方式2
20        e.stopPropagation()
21
22    })
23 </script>
```

## 事件委托

```

1  <button>兄弟</button>
2
3  <script>
4      //给页面上所有的button 标签绑定点击事件
5      // $('button').click(function(){ //无法影响到动态创建的标签
6      //     alert(12)
7      // })
8
9      //事件委托
10     $('body').on('click','button',function(){
11         alert(123) //将body里所有的点击事件交给的button按钮触发
12     }) //在指定的范围内，将事件委托给某个标签，无论该标签是事先写好的还是后面动态创建
    的
13 </script>

```

## 页面加载

```

1  /*等待页面加载完毕再执行代码*/
2  ""js中等待加载完毕执行代码""
3  window.onload=function(){
4      //js代码
5  }
6
7  ""
8  jquery中等待页面加载完毕""
9  #第一种
10 $(document).ready(function(){
11     //js代码
12 })
13 #第二种
14 $(function(){
15     //js代码
16 })
17 #第三种
18 直接写在body内部最下方
19

```

## 动画效果

```

1  
3
4  <script>
5      $('img').hide(5000)
6      $('img').show(5000)
7      $('img').slideUp(5000)
8      // $('img').slideDown(5000)
9      // $('img').fadeOut(5000)
10     // $('img').fadeIn(5000)
11     // $('img').fadeTo(5000,0.5)
12
13 </script>

```

## 补充

```
1 // each()
2
3 #第一种方式
4 $('div').each(function(index){console.log(index)})
5 VM196:1 0
6 VM196:1 1
7 VM196:1 2
8 VM196:1 3
9 VM196:1 4
10 VM196:1 5
11 VM196:1 6
12 VM196:1 7
13 VM196:1 8
14
15 $('div').each(function(index,obj){console.log(index,obj)})
16 VM262:1 0 <div>1</div>
17 VM262:1 1 <div>2</div>
18 VM262:1 2 <div>3</div>
19 VM262:1 3 <div>4</div>
20 VM262:1 4 <div>5</div>
21 VM262:1 5 <div>6</div>
22 VM262:1 6 <div>7</div>
23 VM262:1 7 <div>8</div>
24 VM262:1 8 <div>9</div>
25 S.fn.init(9) [div, div, div, div, div, div, div, div, div, prevObject:
  S.fn.init(1)]
26
27 #第二种方式
28 $.each([111,222,333,444],function(index,obj){console.log(index,obj)})
29 VM529:1 0 111
30 VM529:1 1 222
31 VM529:1 2 333
32 VM529:1 3 444
33 (4) [111, 222, 333, 444]
34
35 "" 有了each后就不用写for循环，更方便""
36 // data()
37
38 "" 能够让标签帮我们存储数据，并且用户肉眼看不见 ""
39 $('div').data('info','回来吧，我原谅你了!')
40 S.fn.init(10) [div#d1, div, div, div, div, div, div, div, div, div,
  prevObject: S.fn.init(1)]
41
42 $('div').first().data('info')
43 '回来吧，我原谅你了!'
44
45 $('div').last().data('info')
46 '回来吧，我原谅你了!'
47
48 $('div').first().removeData('info')
49 S.fn.init [div#d1, prevObject: S.fn.init(10)]
50
51 $('div').first().data('info')
```

