jQuery


```
"""
jQuery内部封装了原生的js代码，还添加了很多的功能
能够通过书写更少的代码，完成js操作
类似于Python里的模块，在前端模块不叫模块，叫”类库“

兼容多个浏览器，使用jQuery就不需要考虑兼容问题

jQuery的宗旨：
    write less do more
    让你用更少的代码完成更多的事情

```

```
python中导入模块需要消耗资源，
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
"""
```

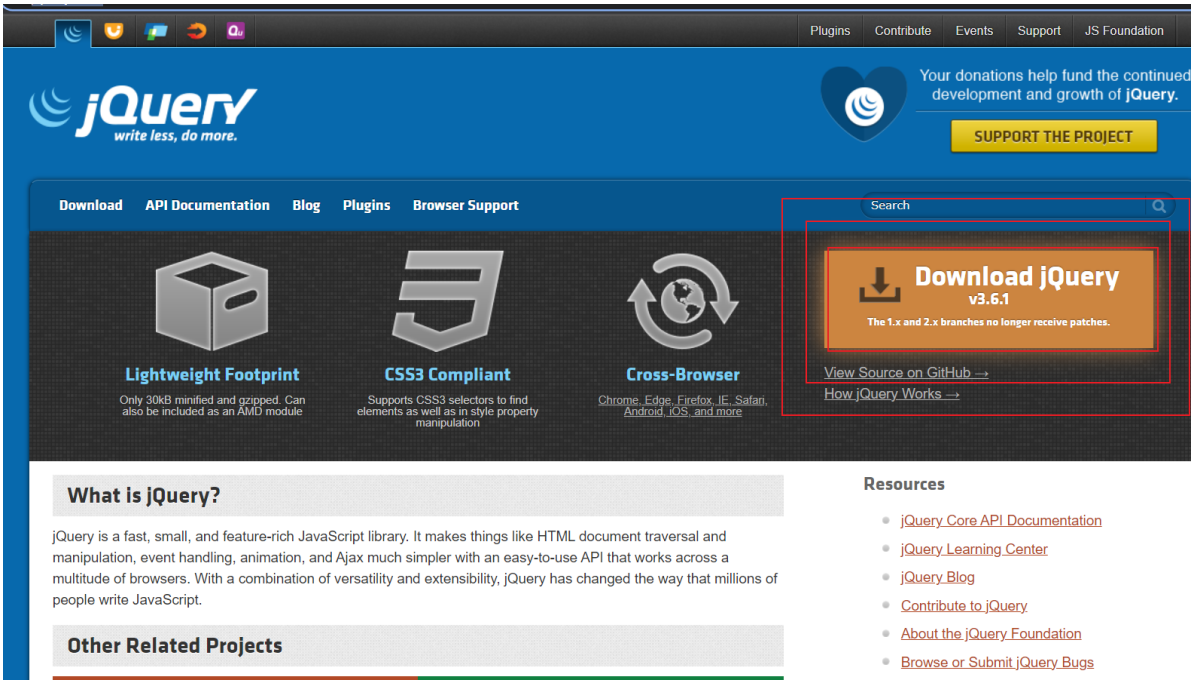
[菜鸟教程]

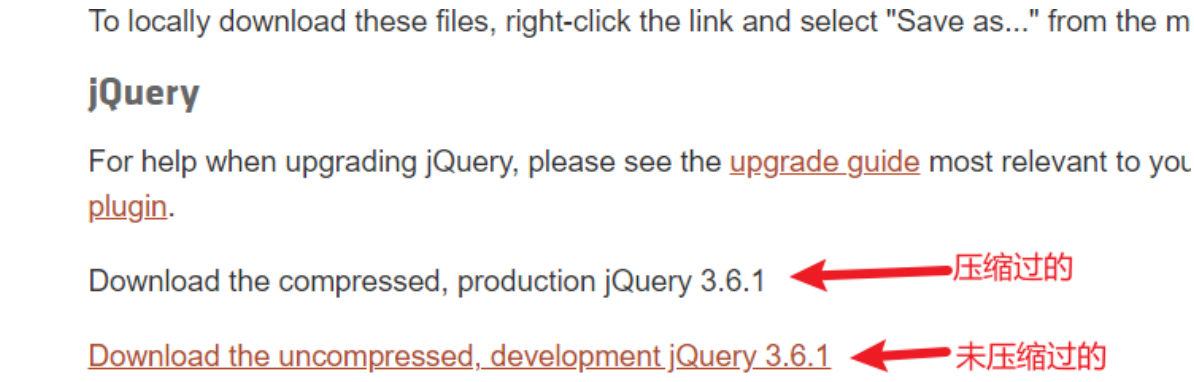
[jQuery API中文文档]

[BootCDN]

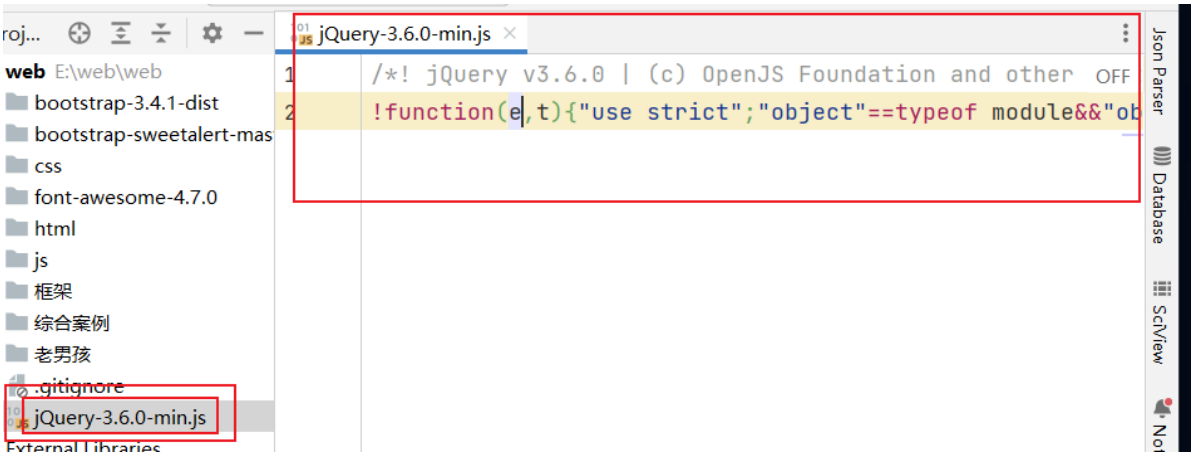
jQuery下载

[官网]





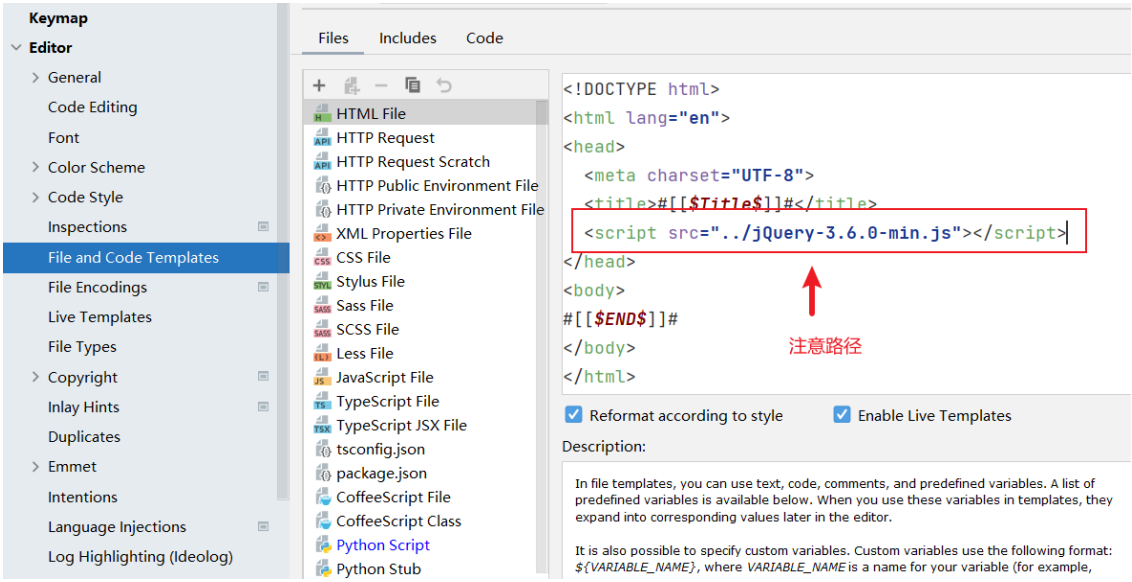
- **打开压缩过的链接直接 Ctrl+C复制，新建一个JS文件直接Ctrl+V进去就可以**



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



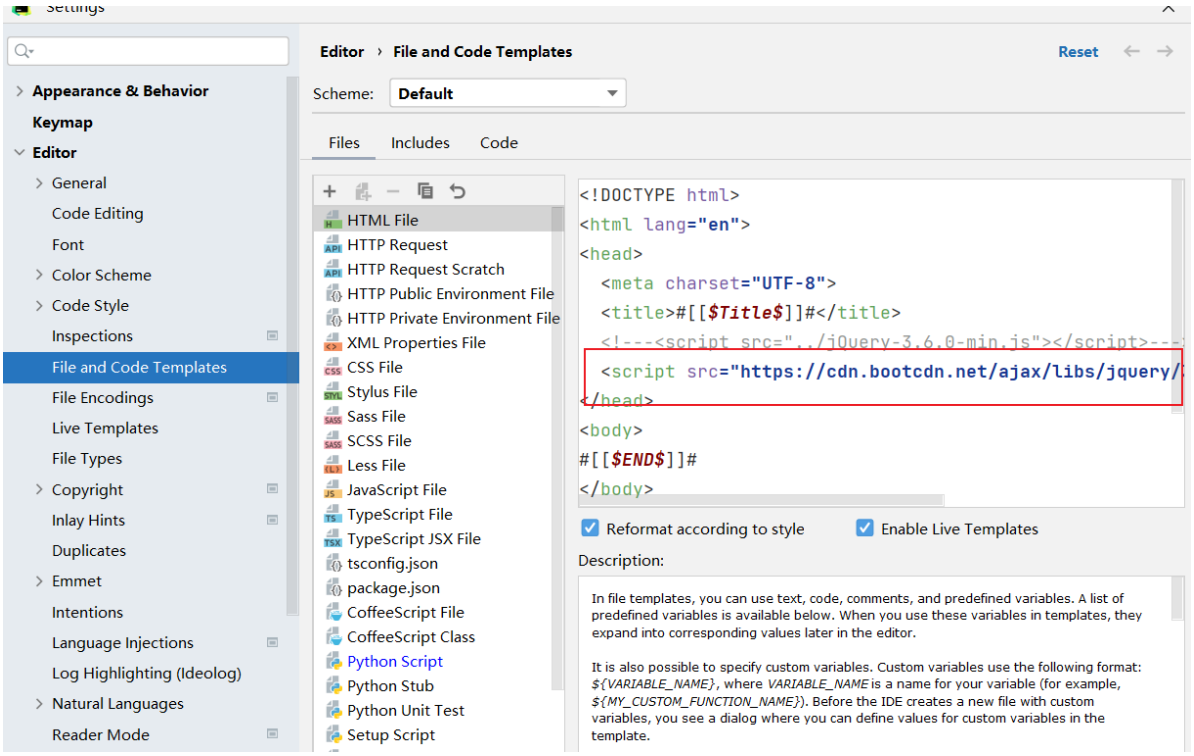
2. 如果不下载jQuery文件，可以直接引用`jQuery`提供的CDN服务（基于网络直接请求加载）
   - CDN:内容分发网络
   - CDN有免费的有收费的

- 前端免费的CDN网站: BootCDN

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


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



jQuery基本语法

```
jQuery(选择器).action()
秉持着jQuery的宗旨，jQuery简写成$
jQuery()  等价于 $()

#jQuery与js代码对比
eg:将p标签内部的文本颜色改为红色
let pELe=document.getElementById('p1')
pELe.style.color='red'
'red'
#jQuery操作标签
$('p').css('color','blue')
```

查找标签

基本选择器

```
//id选择器
$('#d1')
S.fn.init [div#d1]0: div#d1length: 1[[Prototype]]: Object(0)

//class选择器
$('.c1')
S.fn.init [p.c1, prevObject: S.fn.init(1)]0: p.c1length: 1prevObject:
S.fn.init [document][[Prototype]]: Object(0)

//标签选择器
$('span')
S.fn.init(3) [span, span, span, prevObject: S.fn.init(1)]0: span1: span2:
spanlength: 3prevObject: S.fn.init [document][[Prototype]]: Object(0)

//jQuery对象如何变成标签对象
$('#d1')[0]
<div id="d1">…</div>
document.getElementById('d1')
<div id="d1">…</div>

//标签对象变jQuery对象
$(document.getElementById('d1'))
S.fn.init [div#d1]
$(document.getElementById('d1'))[0] //又转成了标签对象
<div id="d1">…</div>
```

组合选择器/分组与嵌套

```
$('div')
S.fn.init(2) [div#d1, div.c1, prevObject: S.fn.init(1)]

$('div.c1')
S.fn.init [div.c1, prevObject: S.fn.init(1)]0: div.c1length: 1prevObject:
S.fn.init [document][[Prototype]]: Object(0)

$('div#d1')
```

```
S.fn.init [div#d1, prevObject: S.fn.init(1)]0: div#d1length: 1prevObject: S.fn.init [document][[Prototype]]: Object(0)

$('*')
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)]

$('#d1,.c1,p') #并列+混用
S.fn.init(3) [div#d1, p, div.c1, prevObject: S.fn.init(1)]

$('div span') #后代
S.fn.init(3) [span, span, span, prevObject: S.fn.init(1)]

$('div>span')#儿子
S.fn.init(2) [span, span, prevObject: S.fn.init(1)]0: span1: spanlength: 2prevObject: S.fn.init [document][[Prototype]]: Object(0)

$('div+span')#紧挨着的第一个（毗邻）
S.fn.init [span, prevObject: S.fn.init(1)]

$('div~span') #兄弟
S.fn.init(2) [span, span, prevObject: S.fn.init(1)]
```

基本筛选器

```
$('ul li') #ul下所有li
S.fn.init(10) [li, li, li, li, li, li, li.c1, li, li, li#d1, prevObject: S.fn.init(1)]

$('ul li:first') #大儿子
S.fn.init [li, prevObject: S.fn.init(1)]0: lilength: 1prevObject: S.fn.init [document][[Prototype]]: Object(0)

$('ul li:last')#小儿子
S.fn.init [li#d1, prevObject: S.fn.init(1)]0: li#d1length: 1prevObject: S.fn.init [document][[Prototype]]: Object(0)

$('ul li:eq(2)')#索引为2的
S.fn.init [document][[Prototype]]: Object(0)

$('ul li:even') #偶数
S.fn.init(5) [li, li, li, li.c1, li, prevObject: S.fn.init(1)]0: li1: li2: li3: li.c14: lilength: 5prevObject: S.fn.init [document][[Prototype]]: Object(0)

$('ul li:odd')#奇数
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)

$('ul li:gt(2)')#索引大于2的
S.fn.init(7) [li, li, li, li.c1, li, li, li#d1, prevObject: S.fn.init(1)]

$('ul li:lt(2)')#索引小于2的
```

```
S.fn.init(2) [li, li, prevObject: S.fn.init(1)]

$('ul li:not("#d1")')#ul下所有li，除了id为d1的
S.fn.init(9) [li, li, li, li, li, li, li.c1, li, li, prevObject:
S.fn.init(1)]

$('div:has("p")')#选取出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

S.fn.init [div, prevObject: S.fn.init(1)]
```

属性选择器

```
$('[username]')
<input type="text" id username="zhangsan">

$('[username="lisi"]')
<p username="lisi"></p>

$('p[username]')
<p username="lisi"></p>

$('[type]')

$('[type="text"]')
```

form表单筛选器

```
$('input[type="text"]')
S.fn.init [input, prevObject: S.fn.init(1)]

$('input[type="password"]')
S.fn.init [input, prevObject: S.fn.init(1)]

$(':password')  #等价于上面第二个
S.fn.init [input, prevObject: S.fn.init(1)]

$(':text')  ##等价于上面第一个
S.fn.init [input, prevObject: S.fn.init(1)]

:text
:password
:file
:radio
:checkbox
:submit
:reset
:button
...
表单的对象属性
:enabled
:disabled
:checked
:selected
```

```
$(':checked')  #它会将checked 和selected都拿到
S.fn.init(2) [input, option, prevObject: S.fn.init(1)]0: input1: optionlength: 2prevObject: S.fn.init [document][[Prototype]]: Object(0)

$(':checked')[0]
<input type="checkbox" value="222" checked>

$(':checked')[1]
<option value selected>333</option>   slot

$(':selected') #它不会，只拿selected
S.fn.init [option, prevObject: S.fn.init(1)]

$(':selected')[0]
<option value selected>333</option>   slot

$('input:selected') #自己加一个限制条件
S.fn.init [prevObject: S.fn.init(1)]
```

筛选器方法

```
$('#d1').next()   #同级别下一个

$('#d1').nextAll()


$('#d1').nextUntil('.c1') #下面所有，直到.c1,不包括.c1


$('.c1').prev()  #上一个


$('.c1').prevAll()


$('.c1').prevUntil('#d2')


$('#d3').parent()#第一级父标签

$('#d3').parent().parent()


$('#d3').parent().parent()


$('#d3').parent().parent().parent()


$('#d3').parent().parent().parent().parent()


$('#d3').parent().parent().parent().parent().parent()


```

```
$('#d3').parents()  #所有父标签
$('#d3').parentsUntil('body')

$('#d2').children()#儿子
S.fn.init(3) [span, p, span, prevObject: S.fn.init(1)]

$('#d2').siblings()#同级别上下所有

$('div p')等价于  $('div').find('p')
#find从某个区域内部筛选出想要的标签

#等价
$('div span:first')
$('div span').first()
$('div span:last')
$('div span').last()
$('div span:not(""#d3")')
$('div span').not('#d3')
```

操作标签

操作类

```
#操作类
"""
js版本                                  jQuery版本
classList.add()                          addClass()
classList.remove()                       removeClass()
classList.contains()                     hasClass()
classList.toggle()                       toggleClass()
"""

```

css操作

```
#css操作
    <p>111</p>
    <p>222</p>
'''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'''
$('p').first().css('color','red').next().css('color','blue')
#jQuery链式操作，使用jQuery可以做到一行代码操作很多标签
#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

class MyClass(object):
    def func1(self):
        print('func1')
        return self
    def func2(self):
        print('func2')
        return self

obj=MyClass()
obj.func1().func2()
```

位置操作

```
#位置操作
offset()
    $('p').offset()
    {top: 100, left: 100}

position()
    $('p').position()
    {top: 100, left: 100}

scrollTop()
    $(window).scrollTop()  #括号内不加参数就是不获取
    $(window).scrollTop(500)#加了参数就是设置

scrollLeft()
```

尺寸

```
#尺寸
$('p').height   #文本
20.8
$('p').width()
866
$('p').innerheight()  #文本+padding
$('p').innerwidth()
$('p').outheight()   #文本+padding+boder
$('p').outwidth()
```

文本操作

```
"""
操作标签内部文本
js版本          jQuery版本
innerText       text()    括号内不加参数就是获取，加了参数就是设置
innerHTML       html()   括号内不加参数就是获取，加了参数就是设置
"""
$('div').text()
'\n         独步江湖，东方不败\n     '
$('div').html()
'\n        <p> 独步江湖，东方不败</p>\n     '
$('div').text('人生苦短，我学Python')
S.fn.init [div, prevObject: S.fn.init(1)]
$('div').html('人生路，冷冷的夜光')
S.fn.init [div, prevObject: S.fn.init(1)]
$('div').text('<h1>人生苦短，我学Python</h1>')
S.fn.init [div, prevObject: S.fn.init(1)]
$('div').html('<h1>人生苦短，我学Python</h1>')
S.fn.init [div, prevObject: S.fn.init(1)]
```

获取值操作

```
"""
js          jQuery
.value      .val()
"""

$('#d1').val()
'vfdghjmhnbv'
$('#d1').val('520')  #括号内不加参数是获取，加了参数是设置
S.fn.init [input#d1]
$('#d2')[0].files[0]  #两个对象之间的转换

$(':checkbox').val(666)#所有的value值都改变成666

```

属性操作

```
"""
js                          jQuery
setAttribute()              attr(name,value)
getAttribute()              attr(name)
removeAttribute()           removeAttr(name)


在用变量存储对象的时候，js推荐使用xxxEle    （标签对象）
在jQuery中推荐使用$xxxELe              （jQuery对象）
"""
let $pEle = $('p')

$pEle.attr('id')

$pEle.attr('class')

$pEle.attr('class','c1')

$pEle.attr('id','666')

$pEle.attr('password','456666')

$pEle.removeAttr('password','456666')



 """对于标签上有的能够看到的属性和自定义属性用attr
 对于返回布尔值比如checkbox,radio,option是否被选中用prop"""
$('#i2').prop('checked')
true
$('#i3').prop('checked',true)
S.fn.init [input#i3]
$('#i3').prop('checked',false)
S.fn.init [input#i3]
```

文档处理

```
"""
js                    jQuery
createElement('p')      $('<p>')

appendChild()           append()


"""
let $pELe = $('<p>')
$pELe.text('你好')
$pELe.attr('id','d6')

$('#d1').append($pELe)#和下面等价 ,内部尾部追加
$('#d1').append($pELe[0])
$pELe.appendTo($('#d1'))

$('#d1').prepend($pELe)  #内部头部追加
$pELe.prependTo($('#d1'))

$('#d2').after($pELe)  #放在某个标签后面
S.fn.init {}[[Prototype]]: Object(0)
$pELe.insertAfter($('#d2'))

$('#d1').before($pELe)#放在某个标签前面
$pEle.insertBefore($('#d1'))

$('#d1').remove()#将标签从DOM树中删除

$('#d1').empty()#清空标签内部所有的内容


```

jQuery事件

jQUery绑定事件的方式

```
//
<body>
    <button id="d1">吻我</button>
    <button id="d2">亲我</button>


    <script>
        //第一种
        $('#d1').click(function () {
            alert('别说话 吻我')
        })

        //第二种，功能强大，支持事件委托
        $('#d2').on('click', function () {
            alert('借钱给我，后面还你')
        })
    </script>
```

```
</body>
```

克隆事件

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="../../jQuery-3.6.0-min.js"></script>
    <style>
        #d1{
            width: 100px;
            height: 100px;
            background-color: orange;
            border: 1px solid blue;

        }
    </style>
</head>
<body>
    <button id="d1">倚天剑，点击传送</button>



    <script>
        $('#d1').on('click',function(){
            // console.log(this);  //this指代的永远是当前被操作的标签对象
            // $(this).clone().insertAfter($('body')) //clone默认只克隆html和css，不克隆事件
            $(this).clone(true).insertAfter($('body')) //括号里面 加参数true即可克隆html,css，事件
        })
    </script>
</body>
```

自定义模态框

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

```
            z-index: 99;
        }
        .modal{
            position: fixed;
            height: 400px;
            width: 400px;
            background-color:white ;
            top: 50%;
            left:50%;
            margin-left: -300PX;
            margin-top: -200PX;
            z-index: 100;
        }
        .hide{
            display: none;

        }
    </style>
</head>
<body>
    <div>我是最下面的</div>
    <button id="d1">给i我出来</button>
    <div class="cover hide "></div>
    <div class="modal hide">
        <p>username：<input type="text" ></p>
        <p>password:<input type="text" ></p>
        <input type="button" value="确定" >
        <input type="button" value="取消" id="d2">

    </div>

    <script>
        let $coverEle=$('.cover')
        let $modalEle=$('.modal')

        $('#d1').click(function(){
            //将两个div标签的hide属性移除
            $coverEle.removeClass('hide')
            $modalEle.removeClass('hide')

        })
        $('#d2').on('click',function(){
            $coverEle.addClass('hide')
            $modalEle.addClass('hide')
        })
    </script>

</body>
</html>
```

左侧菜单

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="../../jQuery-3.6.0-min.js"></script>
    <style>
        * {
            margin: 0;
            padding: 0;

        }

        .left {
            float: left;
            background-color: darkgray;
            width: 20%;
            height: 100%;
            position: fixed;
        }

        .title {
            font-size: 20px;
            color: white;
            text-align: center;

        }

        .items {
            border: 1px solid black;
        }

        .hide {
            display: none;

        }
    </style>
</head>

<body>
    <div class="left">
        <div class="menu">
            <div class="title">菜单一
                <div class="items">111</div>
                <div class="items">222</div>
                <div class="items">333</div>
            </div>
            <div class="title">菜单二
                <div class="items">111</div>
                <div class="items">222</div>
```

```
                <div class="items">333</div>
            </div>
            <div class="title">菜单三
                <div class="items">111</div>
                <div class="items">222</div>
                <div class="items">333</div>
            </div>
        </div>
    </div>

    <script>
        $('.title').click(function () {
            // 先给所有items加hide,
            $('.items').addClass('hide')
            //然后将被点击的标签内部的hide值移除
            //this指代的是当前被操作的标签对象
            $(this).children().removeClass('hide')
        })
    </script>
</body>

</html>
```

返回顶部

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="../../jQuery-3.6.0-min.js"></script>
    <style>
        .hide{
            display: none;
        }
        #d1{
            position: fixed;
            background-color: black;
            right: 20PX;
            bottom: 20px;
            height: 50px;
            width: 50px;
        }
    </style>
</head>
<body>
    <a href="" id="d1"></a>
    <div style="height: 500px;background:red;"></div>
    <div style="height: 500px;background:greenyellow;"></div>
    <div style="height: 500px;background:blue;"></div>
    <a href="#d1" class="hide">回到顶部</a>


```

```
    <script>
        $(window).scroll(function(){
            if ($(window).scrollTop()>300) {
                $('#d1').removeClass('hide')
            }else{
                $('#d1').addClass('hide')
            }
        })
    </script>
</body>
</html>
```

自定义登录校验

```
/*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*/
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="../../jQuery-3.6.0-min.js"></script>
</head>
<body>
    <P>username:
        <input type="text" id="username">
        <span style="color: red;"></span>
    </P>
    <P>password:
        <input type="text" id="password">
        <span style="color: red;"></span>
    </P>
    <button id="d1">提交</button>


    <script>
        let $userName = $('#username')
        let $password =$('#password')
        $('#d1').click(function(){
            //获取用户输入的用户名和密码，做校验.
            let userName = $userName.val()
            let passWord =$passWord.val()
            if (!userName){
                $userName.next().text('用户名不能为空')
            }
            if (!passWord){
                $passWord.next().text('密码不能为空')
            }
        })
        $('input').focus(function(){
            $(this).next().text('')
        })
    </script>
</body>
```

```
</html>
```

input框实时监控

```
<body>
    <input type="text " id="d1">

    <script>
        $('#d1').on('input',function(){
            console.log(this.value);
        })
    </script>
</body>
```

hover事件

```
<body>
    <p id="d1">然后呢</p>

    <script>
        $('#d1').hover(
            function () {
            alert('来宾几位？')  //悬浮
        },
            function(){
                alert('溜了溜了')  //离开
            }

        )
    </script>
</body>
```

键盘按键按下事件

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="../../jQuery-3.6.0-min.js"></script>
</head>
<body>
    <script>
        $(window).keydown(function(event){
            console.log(event.keyCode)
            if (event.keyCode == 16){
                alert('您按下了shift键')
            }
        })
    </script>
</body>
</html>
```

阻止后续事件执行

```
<form action="">
        <span id="d1" style="color: red;"></span>
        <input type="submit" id="d2">
    </form>

    <script>
        $('#d2').click(function(e){
            $('#d1').text('晚上好! ')
            //阻止标签后续时间的执行  方式1
            //return false
            //阻止标签后续时间的执行  方式2
            e.preventDefault()

        })
    </script>
```

阻止事件冒泡

```
<div id="d1">div
        <p id="d2">p
            <span id="d3">span </span>
        </p>
    </div>

    <script>
        $('#d1').click(function(){
            alert('div')
        })
        $('#d2').click(function(){
            alert('p')
            return false
        })
        $('#d3').click(function(e){
            alert('span')
            //阻止事件冒泡的方式1
            //return false
            //阻止事件冒泡的方式2
            e.stopPropagation()

        })
    </script>
```

事件委托

```
 <button>兄弟</button>

    <script>
        //给页面上所有的button 标签绑定点击事件
        // $('button').click(function(){  //无法影响到动态创建的标签
        //     alert(12)
        // })

        //事件委托
        $('body').on('click','button',function(){
            alert(123)  //将body里所有的点击事件交给的button按钮触发
        }) //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
    </script>
```

页面加载

```
/*等待页面加载完毕再执行代码*/
"""js中等待加载完毕执行代码"""
window.onload=function(){
    //js代码
}

"""
jQuery中等待页面加载完毕"""
#第一种
$(document).ready(function(){
    //js代码
})
#第二种
$(function(){
    //js代码
})
#第三种
直接写在body内部最下方

```

动画效果

```
 <img
src="https://img1.baidu.com/it/u=3009731526,373851691&fm=253&app=138&size=w931&n=0&f=JPEG&fmt=auto?sec=1665939600&t=6bf33a6c1b93d6ba296da1f60c8d99be"
    alt="" width="500px" height="500px">

    <script>
        $('img').hide(5000)
        $('img').show(5000)
        $('img').slideUp(5000)
        // $('img').slideDown(5000)
        // $('img').fadeOut(5000)
        // $('img').fadeInt(5000)
        // $('img').fadeTo(5000,0.5)

    </script>
```

补充

```
// each()

#第一种方式
$('div').each(function(index){console.log(index)})
VM196:1 0
VM196:1 1
VM196:1 2
VM196:1 3
VM196:1 4
VM196:1 5
VM196:1 6
VM196:1 7
VM196:1 8

$('div').each(function(index,obj{console.log(index,obj)})
VM262:1 0 <div>1</div>
VM262:1 1 <div>2</div>
VM262:1 2 <div>3</div>
VM262:1 3 <div>4</div>
VM262:1 4 <div>5</div>
VM262:1 5 <div>6</div>
VM262:1 6 <div>7</div>
VM262:1 7 <div>8</div>
VM262:1 8 <div>9</div>
S.fn.init(9) [div, div, div, div, div, div, div, div, div, prevObject:
S.fn.init(1)]

#第二种方式
$.each([111,222,333,444],function(index,obj){console.log(index,obj)})
VM529:1 0 111
VM529:1 1 222
VM529:1 2 333
VM529:1 3 444
(4) [111, 222, 333, 444]

  """ 有了each后就不用写for循环，更方便"""
// data()

""" 能够让标签帮我们存储数据，并且用户肉眼看不见    """
$('div').data('info','回来吧，我原谅你了!')
S.fn.init(10) [div#d1, div, div, div, div, div, div, div, div, div,
prevObject: S.fn.init(1)]

$('div').first().data('info')
'回来吧，我原谅你了!'

$('div').last().data('info')
'回来吧，我原谅你了!'

$('div').first().removeData('info')
S.fn.init [div#d1, prevObject: S.fn.init(10)]

$('div').first().data('info')
```

```
undefined
```