JQ操作

2020年9月20日 14:28

- **操作标签**

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

```
# 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标签变成绿色"""
$('p').first().css('color','red').next().css('color','green')
# jQuery的链式操作 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当前jQuery对象 也就可以继续调用其他方法
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
offset()  # 相对于浏览器窗口
position()  # 相对于父标签

scrollTop()   # 需要了解
  $(window).scrollTop()
  0
  $(window).scrollTop()
  969
  $(window).scrollTop()  # 括号内不加参数就是获取
  1733
  $(window).scrollTop(0)  # 加了参数就是设置
  n.fn.init [Window]
  $(window).scrollTop(500)
  n.fn.init [Window]
scrollLeft()
```

```
# 尺寸
$('p').height()  # 文本
20
$('p').width()
1670
$('p').innerHeight()  # 文本+padding
26
$('p').innerWidth()
1674
$('p').outerHeight()  # 文本+padding+border
26
$('p').outerWidth()
1674
```

```
# 文本操作
"""
操作标签内部文本
js                      jQuery
innerText               text()  括号内不加参数就是获取加了就是设置
innerHTML               html()

$('div').text()
"

        有些话听听就过去了，不要在意，都是成年人！

"
$('div').html()
"
    <p>
        有些话听听就过去了，不要在意，都是成年人！
    </p>
"
$('div').text('你们都是我的大宝贝')
w.fn.init [div, prevObject: w.fn.init(1)]
$('div').html('你个臭妹妹')
w.fn.init [div, prevObject: w.fn.init(1)]
$('div').text('<h1>你们都是我的大宝贝</h1>')
w.fn.init [div, prevObject: w.fn.init(1)]
$('div').html('<h1>你个臭妹妹</h1>')
w.fn.init [div, prevObject: w.fn.init(1)]
"""
```

```
# 获取值操作
"""
js                          jQuery
.value                      .val()
"""
$('#d1').val()
"sasdasdsadsadad"
$('#d1').val('520快乐')  # 括号内不加参数就是获取加了就是设置

w.fn.init [input#d1]
$('#d2').val()
"C:\fakepath\01_测试路由.png"
$('#d2')[0].files[0]  # 牢记两个对象之间的转换
File {name: "01_测试路由.png", lastModified: 1557043083000,
lastModifiedDate: Sun May 05 2019 15:58:03 GMT+0800 (中国标准时间),
webkitRelativePath: "", size: 28733, …}
```

```
# 属性操作
"""
js中                                jQuery
setAttribute()                       attr(name,value)
getAttribute()                       attr(name)
removeAttribute()                    removeAttr(name)

在用变量存储对象的时候 js中推荐使用
  XXXEle       标签对象
jQuery中推荐使用
  $XXXEle      jQuery对象
"""
let $pEle = $('p')
undefined
$pEle.attr('id')
"d1"
$pEle.attr('class')
undefined
$pEle.attr('class','c1')
w.fn.init [p#d1.c1, prevObject: w.fn.init(1)]
$pEle.attr('id','id666')
w.fn.init [p#id666.c1, prevObject: w.fn.init(1)]
$pEle.attr('password','jason123')
w.fn.init [p#id666.c1, prevObject: w.fn.init(1)]
$pEle.removeAttr('password')
w.fn.init [p#id666.c1, prevObject: w.fn.init(1)]
```

```
"""
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
"""

$('#d3').attr('checked')
"checked"
$('#d2').attr('checked')
undefined
$('#d2').attr('checked')
undefined
$('#d4').attr('checked')
undefined
$('#d3').attr('checked')
"checked"
$('#d3').attr('checked','checked')  # 无效
w.fn.init [input#d3]


$('#d2').prop('checked')
false
$('#d2').prop('checked')
true
$('#d3').prop('checked',true)
w.fn.init [input#d3]
$('#d3').prop('checked',false)
w.fn.init [input#d3]
```

```
# 文档处理
"""
js                                  jQuery
createElement('p')                  $('<p>')
appendChild()                       append()

"""
let $pEle = $('<p>')
$pEle.text('你好啊 草莓要不要来几个?')
$pEle.attr('id','d1')
$('#d1').append($pEle)  # 内部尾部追加
$pEle.appendTo($('#d1'))

$('#d1').prepend($pEle)  # 内部头部追加
w.fn.init [div#d1]
$pEle.prependTo($('#d1'))
w.fn.init [p#d1, prevObject: w.fn.init(1)]
```

```
$('#d2').after($pEle)  # 放在某个标签后面
w.fn.init [p#d2]
$pEle.insertAfter($('#d1'))

$('#d1').before($pEle)
w.fn.init [div#d1]
$pEle.insertBefore($('#d2'))
```

```
$('#d1').remove()  # 将标签从DOM树中删除
w.fn.init [div#d1]

$('#d1').empty()  # 清空标签内部所有的内容
w.fn.init [div#d1]
```

事件

```
// 第一种
    $('#d1').click(function () {
            alert('别说话 吻我')
    });

// 第二种(功能更加强大 还支持事件委托)
    $('#d2').on('click',function () {
            alert('借我钱买草莓 后面还你')
    })
```

- 克隆事件

```
<button id="d1">屠龙宝刀，点击就送</button>

<script>
    $('#d1').on('click',function () {
        // console.log(this)  // this指代是当前被操作的标签对象
        // $(this).clone().insertAfter($('body'))  // clone默认情况下只克隆html和css 不克隆事件
        $(this).clone(true).insertAfter($('body'))  // 括号内加true即可克隆事件

    })
</script>
```

- 左侧菜单

```
<script>
    $('.title').click(function () {
        // 先给所有的items加hide
        $('.items').addClass('hide')
        // 然后将被点击标签内部的hide移除
        $(this).children().removeClass('hide')
    })
</script>
<!--尝试用一行代码搞定上面的操作-->
```

··模态框

```
let $coverEle = $('.cover')
let $modalEle = $('.modal')
$('#d1').on('click',function () {
    $coverEle.removeClass('hide')
    $modalEle.removeClass('hide')
})

$('#d2').click(function () {
    $coverEle.addClass('hide')
    $modalEle.addClass('hide')
})
```

- 返回顶部

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

- 自定义登陆校验

```
# 在获取用户的用户名和密码的时候 用户如果没有填写 应该给用户展示提示信息
<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>
```

- input实时监控

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

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

- hover事件

```
<script>
    // $("#d1").hover(function () {  // 鼠标悬浮 + 鼠标移开
    //     alert(123)
    // })

    $('#d1').hover(
        function () {
            alert('我来了')  // 悬浮
    },
        function () {
            alert('我溜了')  // 移开
        }
    )
</script>
```

- 键盘按键按下事件

```
<script>
    $(window).keydown(function (event) {
        console.log(event.keyCode)
        if (event.keyCode === 16){
            alert('你按了shift键')
        }
    })
</script>
```

阻止后续事件执行

```
<script>
    $('#d2').click(function (e) {
        $('#d1').text('宝贝 你能看到我吗?')
        // 阻止标签后续事件的执行 方式1
        // return false
        // 阻止标签后续事件的执行 方式2
        // e.preventDefault()
    })
</script>
```

阻止事件冒泡

```
<script>
        $('#d1').click(function () {
            alert('div')
        })
        $('#d2').click(function () {
            alert('p')
        })
        $('#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>是兄弟，就来砍我!!!</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(123)
    // })

    // 事件委托
    $('body').on('click','button',function () {
        alert(123)  // 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
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

```
$('#d1').hide(5000)
w.fn.init [div#d1]
$('#d1').show(5000)
w.fn.init [div#d1]
$('#d1').slideUp(5000)
w.fn.init [div#d1]
$('#d1').slideDown(5000)
w.fn.init [div#d1]
$('#d1').fadeOut(5000)
w.fn.init [div#d1]
$('#d1').fadeIn(5000)
w.fn.init [div#d1]
$('#d1').fadeTo(5000,0.4)
w.fn.init [div#d1]
```

补充

```
# each()
$('div')
w.fn.init(10) [div, div, div, div, div, div, div, div, div, div,
prevObject: w.fn.init(1)]
$('div').each(function(index){console.log(index)})
VM181:1 0
VM181:1 1
VM181:1 2
VM181:1 3
VM181:1 4
VM181:1 5
VM181:1 6
VM181:1 7
VM181:1 8
VM181:1 9
w.fn.init(10) [div, div, div, div, div, div, div, div, div, div,
prevObject: w.fn.init(1)]
$('div').each(function(index,obj){console.log(index,obj)})
VM243:1 0 <div>•1•</div>•
VM243:1 1 <div>•2•</div>•
VM243:1 2 <div>•3•</div>•
VM243:1 3 <div>•4•</div>•
VM243:1 4 <div>•5•</div>•
VM243:1 5 <div>•6•</div>•
VM243:1 6 <div>•7•</div>•
VM243:1 7 <div>•8•</div>•
VM243:1 8 <div>•9•</div>•
VM243:1 9 <div>•10•</div>•
# 第二种方式
$.each([111,222,333],function(index,obj){console.log(index,obj)})
VM484:1 0 111
VM484:1 1 222
VM484:1 2 333
(3) [111, 222, 333]
```

```
# 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！')
w.fn.init(10) [div#dl, div, div, div, div, div, div, div, div, div,
prevObject: w.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().data('xxx')
undefined
$('div').first().removeData('info')
w.fn.init [div#dl, prevObject: w.fn.init(10)]

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