事件处理绑定`on()`

1、on()方法优势1：

在匹配元素上绑定一个或多个事件的事件处理函数

语法：

```
element.on(events,[selsetor],fn)
```

1. `events`：一个或多个空格分割的事件类型，如“`click`”或“`keydown`”
2. `selector`：元素的子元素选择器
3. `fn`：回调函数，绑定在元素上的侦听函数

2、on()方法优势2：

可以使用事件委派操作。事件委派定义是，把原来加给子元素身上的事件绑定在父元素身上，把事件委派给父元素

```
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
        <li>4</li>
        <li>5</li>
    </ul>
    <script>
        $('ul').on('click','li',function(){
            console.log(123);
        })
    </script>
```

3、on()方法优势3：

动态创建的元素，`click()`没有办法绑定事件，`on()`可以给动态生成的元素绑定事件。

```
    <ol>
        <li>测试</li>
    </ol>

    <script>
        /* $('li').click(function(){
            console.log(123);
        }) */

        $('ol').on('click','li',function(){
            console.log(123);
```

```
        })

        var li1 = $('<li>测试1</li>');
        $('ol').append(li1);
    </script>
```

HW小案例：发布微博

链接如下：

```
```

事件处理解绑`off()`

`off()`方法可以移除通过`on()`方法添加的事件处理程序

```
// 解绑事件off()用法
$('div').off();          //解绑div身上所有的事件
$('div').off('click');  //只解绑div身上对应的点击事件,click 点击
$('ul').off('click','li');  //解绑通过事件委托的事件
```

代码片段：

```
    <div>123</div>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>

    <script>
        $('div').on({
            click : function(){
                console.log($(this));
            },
            mouseover : function(){
                $(this).css('color','red');
            }
        })

        $('ul').on('click','li',function(){
            console.log('this is li标签');
        })

        // 解绑事件off()用法
        $('div').off();          //解绑div身上所有的事件
        $('div').off('click');  //只解绑div身上对应的点击事件
        $('ul').off('click','li');  //解绑通过事件委托的事件
    </script>
```

`one()` 只触发一次事件

代码片段：

```
    <p>文字</p>
    <script>
        $('p').one('click',function(){
            console.log(123);
        })
    </script>
```

自动触发事件

1. 元素.事件()
2. 元素.trigger('事件')
3. 元素.triggerHandler('事件')
    - triggerHandler不会触发元素的默认行为。

```
    <div>123</div>
    <input type="text">

    <script>
        $('div').on('click',function(){
            console.log(123);
        })
        // 自动触发事件
        // 1.元素.事件()
        // $('div').click();
        // 2.元素.trigger('事件')
        // $('div').trigger('click');
        // 3.元素.triggerHandler('事件'),triggerHandler不会触发元素的默认行为。
        // $('div').triggerHandler('click');
        $('input').on('focus',function(){
            $(this).val('测试');
        })
        $('input').triggerHandler('focus');
    </script>
```

事件对象

事件被触发，就会有事件对象的产生

```
element.on(events,[selector],function(event){ })
```

- 阻止默认行为：`event.preventDefault()` 或者 `return false`

- 阻止冒泡：`event.stopPropagation()`

代码片段：

```
    <p>123</p>

    <script>
        $('p').on('click',function(event){
            // console.log(event);
            console.log('div点击了');
            event.stopPropagation();
        })
        $(document).on('click',function(event){
            console.log('文档页面也点击了');
        })
    </script>
```