



链滴

事件的三个阶段：捕获阶段 目标阶段 冒泡阶段及防止冒泡和捕获

作者：[BYwangbing](#)

原文链接：<https://ld246.com/article/1582987084602>

来源网站：[链滴](#)

许可协议：[署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)



最恰当的比喻：可以想象画在一张纸上的一组同心圆，如果你把手指放在圆心上，那么你的手指指向其实不是一个圆，而是纸上所有的圆。放到实际页面中就是，你点击一个按钮，事实上你还同时点击按钮所有的父元素。

事件流阶段是

- **捕获阶段** (从根节点开始顺着目标节点构建一条事件路径，即事件由页面元素接收，逐级向下，到体的元素)
- **目标阶段** (到达目标节点，即元素本身)
- **冒泡阶段** (从目标节点顺着捕获阶段构建的路径回去，即跟捕获相反具体元素本身，逐级向上，到面元素)

!!! 我觉得写一个 demo 更容易理解

思考：那是不是所有的事件都要经历这三个过程呢？

防止冒泡和捕获

- w3c 的方法是 `event.stopPropagation()`，IE 则是使用 `event.cancelBubble = true`

□ 事件处理过程中，阻止了事件冒泡，但不会阻止默认行为

- `return false`

□ 事件处理过程中，阻止了事件冒泡，也阻止了默认行为

```
window.event? window.event.cancelBubble = true : e.stopPropagation();
```

- `return false` 不仅阻止了事件往上冒泡，而且阻止了事件本身。

- `event.stopPropagation()` 则只阻止事件往上冒泡，不阻止事件本身。

取消默认事件

- w3c 的方法是 `e.preventDefault()`，IE 则是使用 `e.returnValue = false;`

▯▯`preventDefault` 它是事件对象(Event)的一个方法，作用是取消一个目标元素的默认行为

总结

- 当需要停止冒泡行为时，可以使用

```
function stopBubble(e) {
//如果提供了事件对象，则这是一个非IE浏览器
if ( e && e.stopPropagation )
    //因此它支持W3C的stopPropagation()方法
    e.stopPropagation();
else
    //否则，我们需要使用IE的方式来取消事件冒泡
    window.event.cancelBubble = true;
}
```

- 当需要阻止默认行为时，可以使用

```
//阻止浏览器的默认行为
function stopDefault( e ) {
    //阻止默认浏览器动作(W3C)
    if ( e && e.preventDefault )
        e.preventDefault();
    //IE中阻止函数器默认动作的方式
    else
        window.event.returnValue = false;
    return false;
}
```

事件注意点

1. event 代表事件的状态，例如触发 event 对象的元素、鼠标的位置及状态、按下的键等等；
2. event 对象只在事件发生的过程中才有效。

```
function a(e){
    var e = (e) ? e : ((window.event) ? window.event : null);
    var e = e || window.event; // firefox下window.event为null, IE下event为null
}
```

总结参考与: <http://caibaojian.com/javascript-stoppropagation-preventdefault.html>

补充 1: vue 防止冒泡和捕获

- `@click.stop` : 阻止事件冒泡
- `@click.prevent` : 阻止事件默认行为

- `@click.self`: 事件只作用在元素本身, 而不是其子元素

```
<!-- 阻止单击事件继续传播 -->
<a @click.stop="doThis"></a>
<!-- 提交事件不再重载页面 -->
<form @submit.prevent="onSubmit"></form>
<!-- 修饰符可以串联 -->
<a @click.stop.prevent="doThat"></a>
<!-- 只有修饰符 -->
<form @submit.prevent></form>
<!-- 只当在 event.target 是当前元素自身时触发处理函数 -->
<!-- 即事件不是从内部元素触发的 -->
<div @click.self="doThat">...</div>
```

补充 2: React 防止冒泡和捕获

由上述可知: 原生 JavaScript 有这两种 `event.cancelBubble = true` 或 `event.stopPropagation()`, 防止冒泡和捕获; 但是 React, 使用 `event.cancelBubble` 没有效果, `return false` 也没有效果

官方也说了, 在 React 中不能使用 `return false` 的方式阻止默认行为;

- 使用 `event.stopPropagation()` 成功干掉默认事件
- 还可以使用 `event.preventDefault()`;

阻止冒泡: `e.nativeEvent.stopImmediatePropagation()` 方法