Vue.js 为 v-on 提供了**事件修饰符**。之前提过，修饰符是由点开头的指令后缀来表示的。

* .stop
* .prevent
* .capture
* .self
* .once
* .passive

<!-- 阻止单击事件继续传播 -->

<a v-on:click.stop="doThis"></a>

<!-- 提交事件不再重载页面 -->

<form v-on:submit.prevent="onSubmit"></form>

<!-- 修饰符可以串联 -->

<a v-on:click.stop.prevent="doThat"></a>

<!-- 只有修饰符 -->

<form v-on:submit.prevent></form>

<!-- 添加事件监听器时使用事件捕获模式 -->

<!-- 即元素自身触发的事件先在此处理，然后才交由内部元素进行处理 -->

<div v-on:click.capture="doThis">...</div>

<!-- 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 v-on:click.self="doThat">...</div>

使用修饰符时，顺序很重要；相应的代码会以同样的顺序产生。因此，用 v-on:click.prevent.self 会阻止**所有的点击**，而 v-on:click.self.prevent 只会阻止对元素自身的点击

# URL中的Hash

**1.#的含义**

#代表网页中的一个位置，其右边的字符，就是该位置的标识符。比如

http://www.example.com/index.html#print

就是代表index.html中的print位置。浏览器会自动把print位置滚动到页面可视区域内。

设置方法：

step1：设置一个锚点<a href="#print">定位到print位置</a>

step2：在页面需要定位的内容加上id="print"。例如：<div id="print"></div>

测试：step1设置的锚点，step2中id为print的内容会滚动到页面顶端（可观察滚动条的距离）。同时，页面的url末端中会出现#print的哈希值。

**2.HTTP请求不包含#**

#号是用来指导浏览器动作的，对服务器端完全无用。所以，HTTP请求中不包含#。

比如，访问下面的网址：

http://jquery.com#hello

浏览器实际发出的请求时这样的：



可以看到，只是请求了http://jquery.com，没有请求"#hello"的部分。

**3.#后面的字符**

在第一个#后面出现的任何字符，都会被浏览器解读为位置标识符。这意味着，这些字符都不会被发送到服务器端。

比如，下面的URL的原意是指定一个颜色值：

http://jquery.com/?color=#fff

但是浏览器实际发出的请求是：



可以看到，"#fff"被省略了。只有将#转码为%23，浏览器才会将其作为实义字符处理。也就是说，上面的网址应该被写成：

http://jquery.com/?color=%23fff

**4.改变#不触发网页重载**

单单改变#后的内容，浏览器只会滚动到相应位置，不会重新加载网页。

浏览器不会重新向服务器请求页面。

**5.改变#会改变浏览器的访问历史**

每一次改变#后的部分，都会在浏览器的访问历史中增加一个记录，使用"后退"按钮，就可以回到上一个位置。

这对于ajax应用程序特别有用，可以用不同的#值，表示不同的访问状态，然后向用户给出可以访问某个状态的链接。

值得注意的是，上述规则对IE 6和IE 7不成立，它们不会因为#的改变而增加历史记录。

**6.window.location.hash读取#值**

window.location.hash这个属性可读可写。读取时，可以用来判断网页状态是否改变；写入时，则会在不重载网页的前提下，创造一条访问历史记录。

**7.onhashchange事件**

这是一个HTML 5新增的事件，当#值发生变化时，就会触发这个事件。IE8+、Firefox 3.6+、Chrome 5+、Safari 4.0+支持该事件。

它的使用方法有三种：

1.window.onhashchange = func;

2.<body onhashchange="func();">

3.window.addEventListener("hashchange", func, false);

**8.Google抓取#的机制**

默认情况下，Google的网络蜘蛛忽视URL的#部分。

但是，Google还[规定](http://www.google.com/support/webmasters/bin/answer.py?hl=en&answer=174992)，如果你希望Ajax生成的内容被浏览引擎读取，那么URL中可以使用"#!"，Google会自动将其后面的内容转成查询字符串\_escaped\_fragment\_的值。

比如，Google发现新版twitter的URL如下：

http://twitter.com/#!/username

就会自动抓取另一个URL：

http://twitter.com/?\_escaped\_fragment\_=/username

通过这种机制，Google就可以索引动态的Ajax内容。