**web前端：Unable to preventDefault inside passive event listener invocation的产生原因及解决**

f922f82b3e85b09d4b75bf0ffa3a935.png

**产生原因：**

条件1：使用监听事件时候，第三个参数配置了passive为true(说明指定了这个监听为一个被动监听)

window.addEventListener("touchmove", handler, { passive: true });

条件2：上条件中的被动监听的handler中调用了

function handler() {

//阻止默认事件，如点击<a>标签阻止其默认跳转事件

event.preventDefault();

//其他逻辑...

}

上述两个条件同时成立的时候，就会报标题中错误或警告。

**本质：**

谷歌浏览器对event.preventDefault()（默认事件阻止）的检测机制变化导致的（听不懂？直接看下面）。

老版本：当用户触发我们定义的监听事件时候（如条件1中的“touchmove”事件），浏览器会主动检测对应的handler代码中是否有event.preventDefault(),以便进行默认事件阻止。但是这样做会增加响应时间，用户体验受到影响；

新版本：默认不再检测event.preventDefault()，直接响应用户操作。这相当于把我们定义的事件监听都设置成了被动事件监听。具体代码就是：

window.addEventListener("touchmove", handler);

window.addEventListener("touchmove", handler, { passive: true });

上面两个代码效果相同，我们经常用第一条，不知不觉中定义了一个被动监听事件。

那么问题就来了，我们不知道我们的事件已经默认被定义为了被动事件监听。结果我们在这个事件监听中调用了event.preventDefault()，浏览器就不高兴了，报错，告诉你：

“你定义的事件不是一个被动事件监听吗？不就是告诉我为了提高响应速度不要处理event.preventDefault()吗？为啥你还要调用event.preventDefault()！”

看了我们已经找到问题所在了。

总结：被动事件监听不能调用event.preventDefault()。

**解决办法：**

1、声明事件监听的时候设置为主动事件监听：

window.addEventListener(‘touchmove’, handler, { passive: false})；

2、设置监听事件绑定的dom的CSS为：

touch-action:none

表示当触控事件发生在绑定的dom上时，不进行任何操作。