**海峡短款问题分析报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 项目名称 |  | 项目代号 |  |
| 文档名称 |  | 文档版本 | V1.0 |
| 起草 | 张婵娟 | 起草日期 | 2023-2-17 |
| 审批 |  | 审批日期 |  |



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 修改内容 | 页号 | 修改人 | 完成日期 |
| 1 | 初始创建 |  | 郭小军 | 2023-2-17 |
| 2 | 内容完善 |  | 张婵娟 | 2023-2-18 |
| 3 |  |  |  |  |
| 4 |  |  |  |  |

**变更记录**

注：对该文件内容增加、删除或修改均需填写此变更记录，详细记载变更信息，以保证其可追溯性。

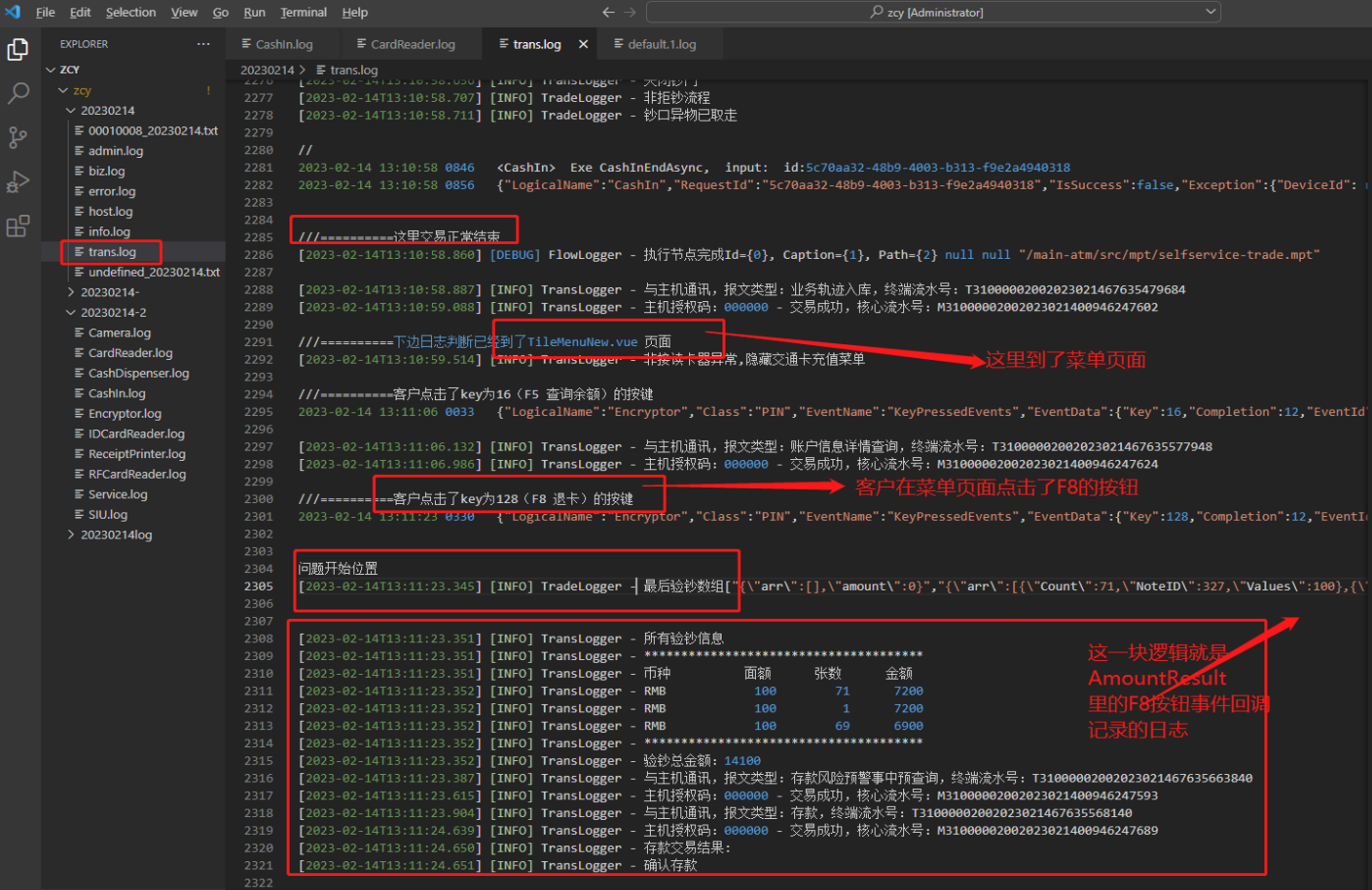
# 问题背景

## 生产问题描述

海峡银行在2023/2/14日，客户做存款交易期间，发生了一笔14100元的短款。

## 生产日志分析

生产日志分析如下：



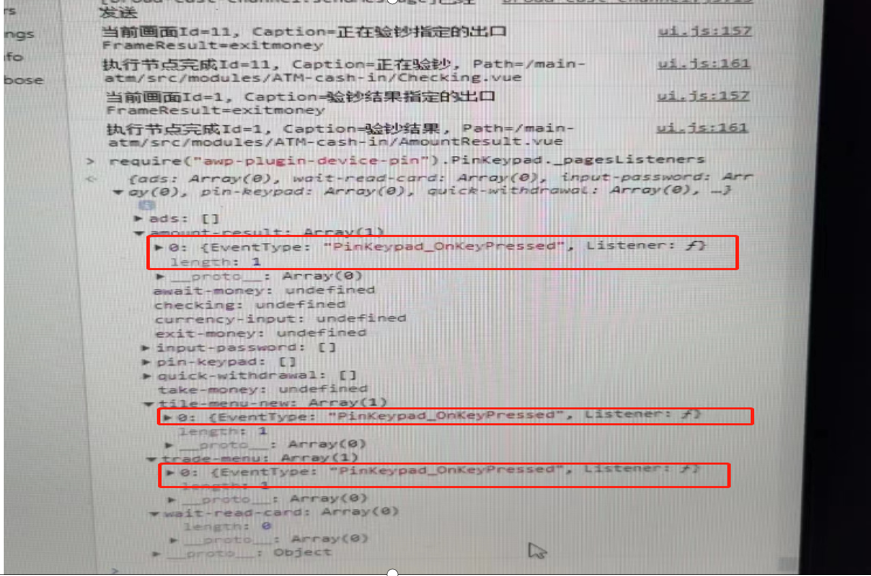
日志中可以看到，在存款交易已经结束的情况下（如上图的2286行日志），客户在菜单页面摁了F8的退卡按钮（如上图的2301行日志），又一次**异常的**触发了存款交易页面里的事件（如上图的2308行的日志），所以导致了发生了短款。

**总结异常：在菜单页面客户按F8（退卡）按键，执行了交易内的按键事件逻辑。**

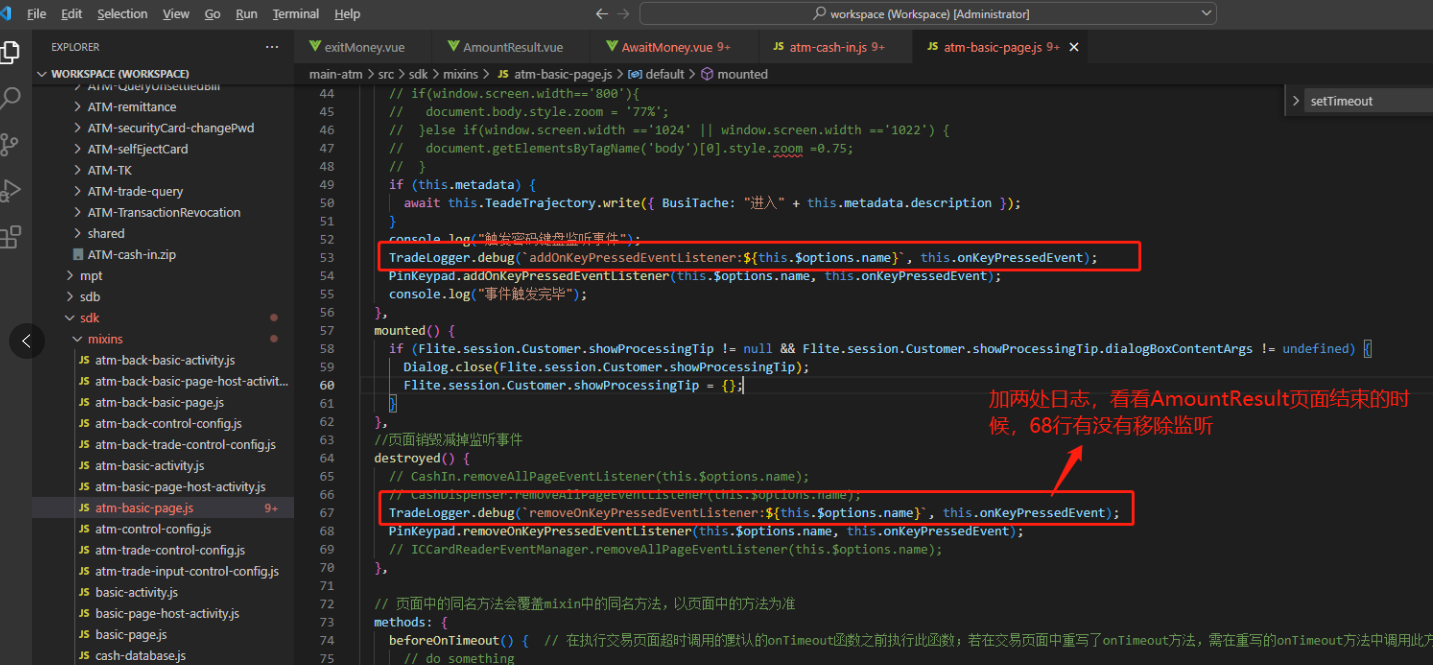
## **生产问题分析**

为什么会在菜单页面里还能触发交易页面中的事件逻辑呢？**说明在存款交易的页面中并没有成功移除事件，所以在菜单页面中此事件还存在着**。

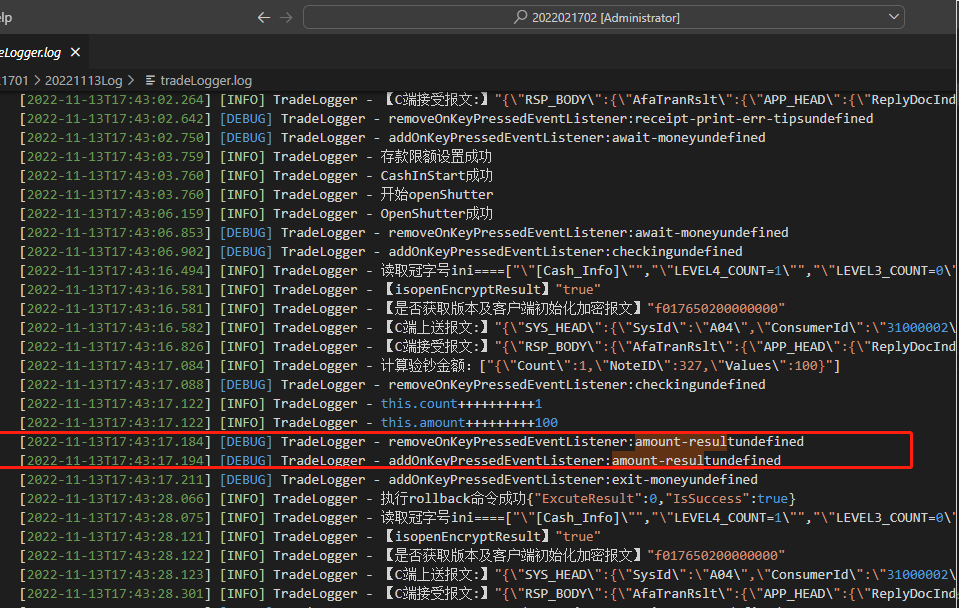
此时在菜单页面的控制台上也可以看到事件确实存在，如下图：



接下来，在事件添加移除的位置添加日志，查找事件为什么没有被成功移除掉，如下图：

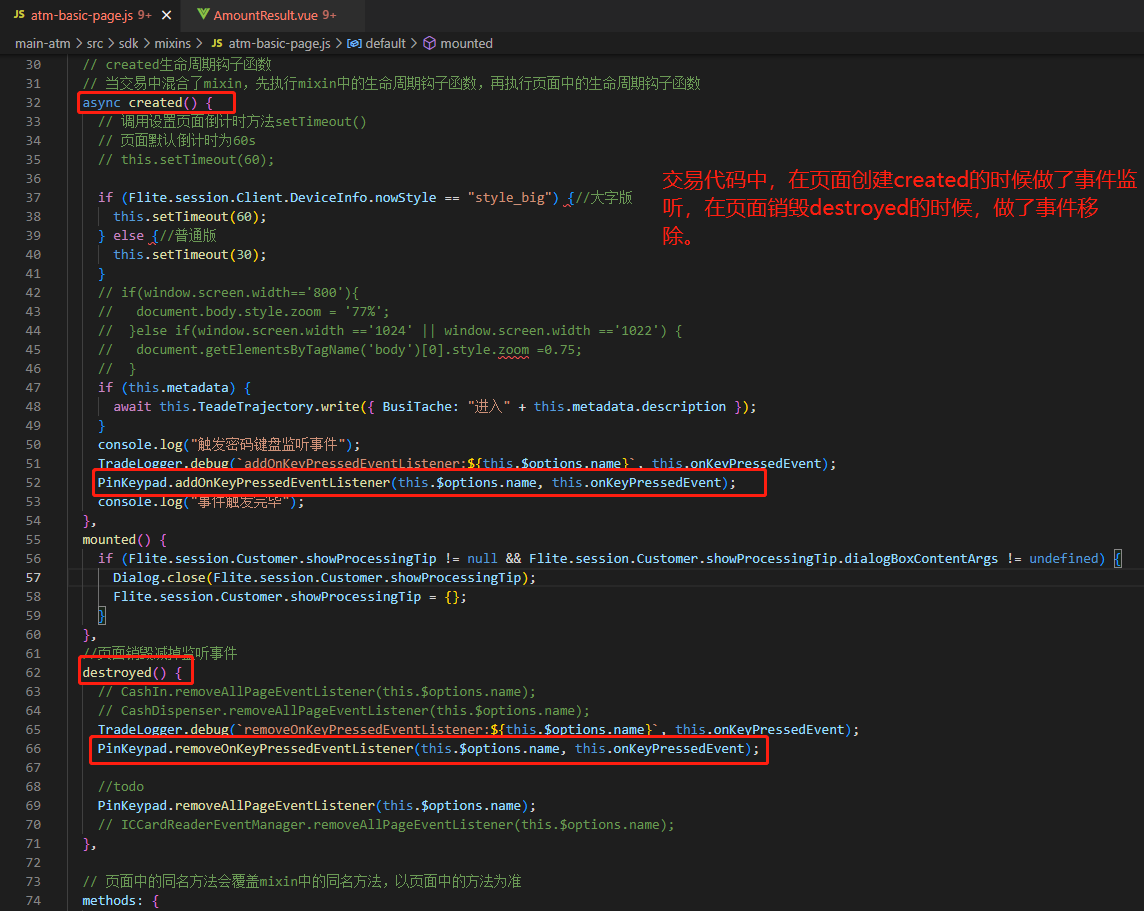


加了日志之后测试发现，**事件移除先于事件添加**，导致了事件没有被移除掉。如下图：

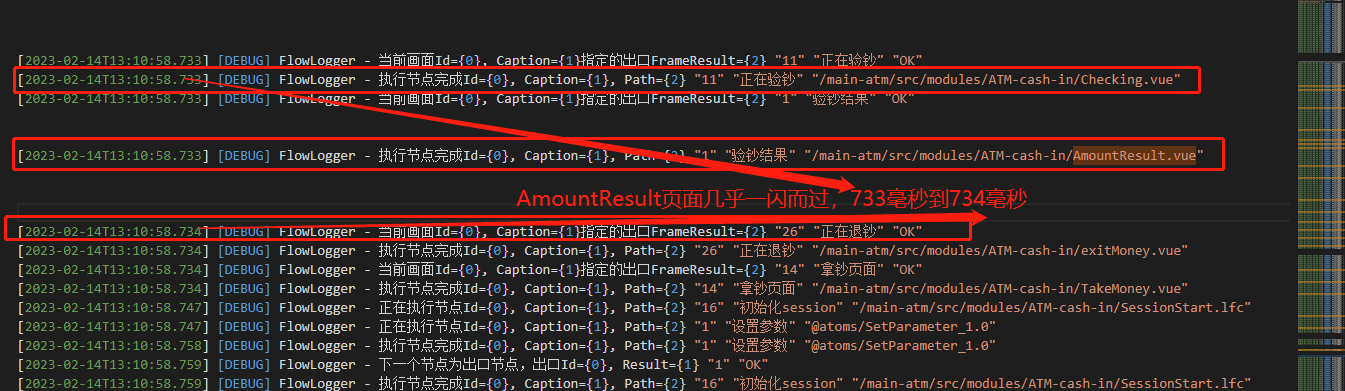


## **交易代码分析**

接下来，查看下代码位置，为什么**事件移除先于事件添加？**



66行代码比52行代码先执行，说明交易页面销毁的很快，日志也可以证明此页面的停留时间毫秒级别，如下图：



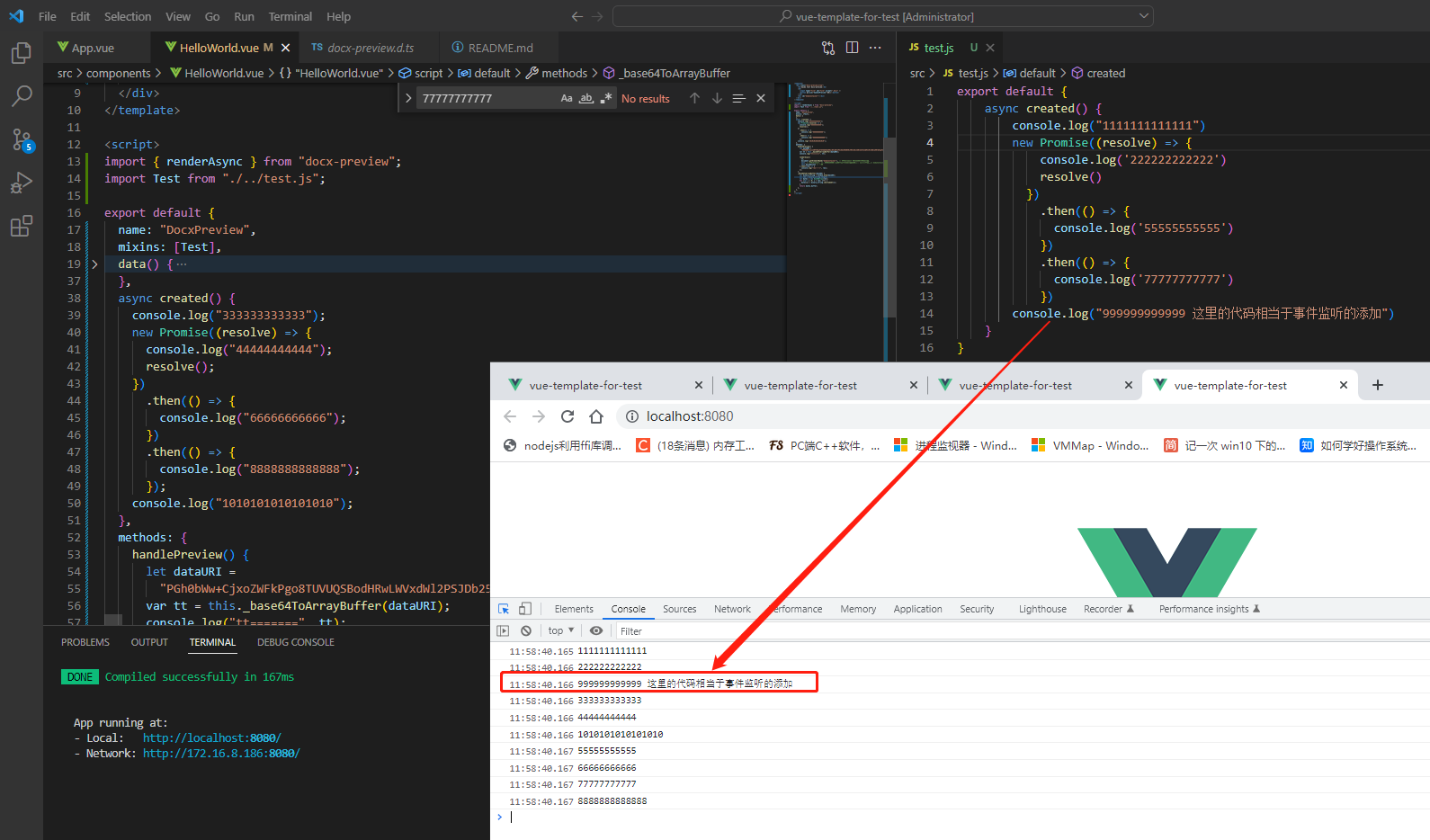
理论上，**交易页面创建先执行created里的逻辑，页面销毁后执行destroyed里的逻辑。**但是ResultAmount验钞结果页面一闪而过，所以此页面created，destroyed两个生命周期函数只是错开了毫秒级别的调用，**上图日志相当于前一毫秒调用了created，后一毫秒调用了destroyed。**

# 问题解决

## Vue框架中created调用

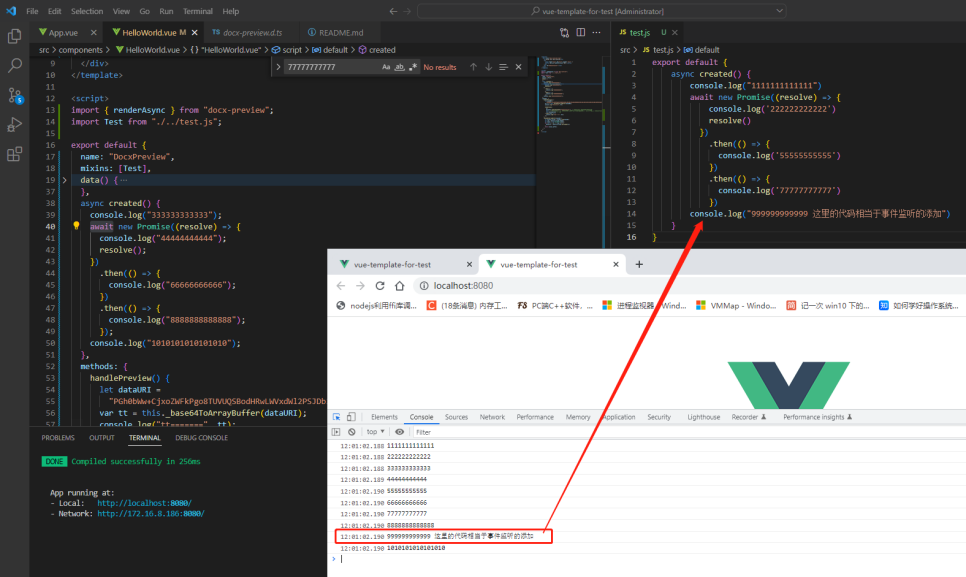
在解决问题之前，我们要先学习下Vue的created的调用。先通过一个简单的测试demo，模拟下验钞结果页面的执行，了解下vue中created里的逻辑调用顺序。

（1）created里的代码执行不加await，调用顺序如下：



如上图，不加await的时候，右侧14行执行结束之后才执行左侧的39行。

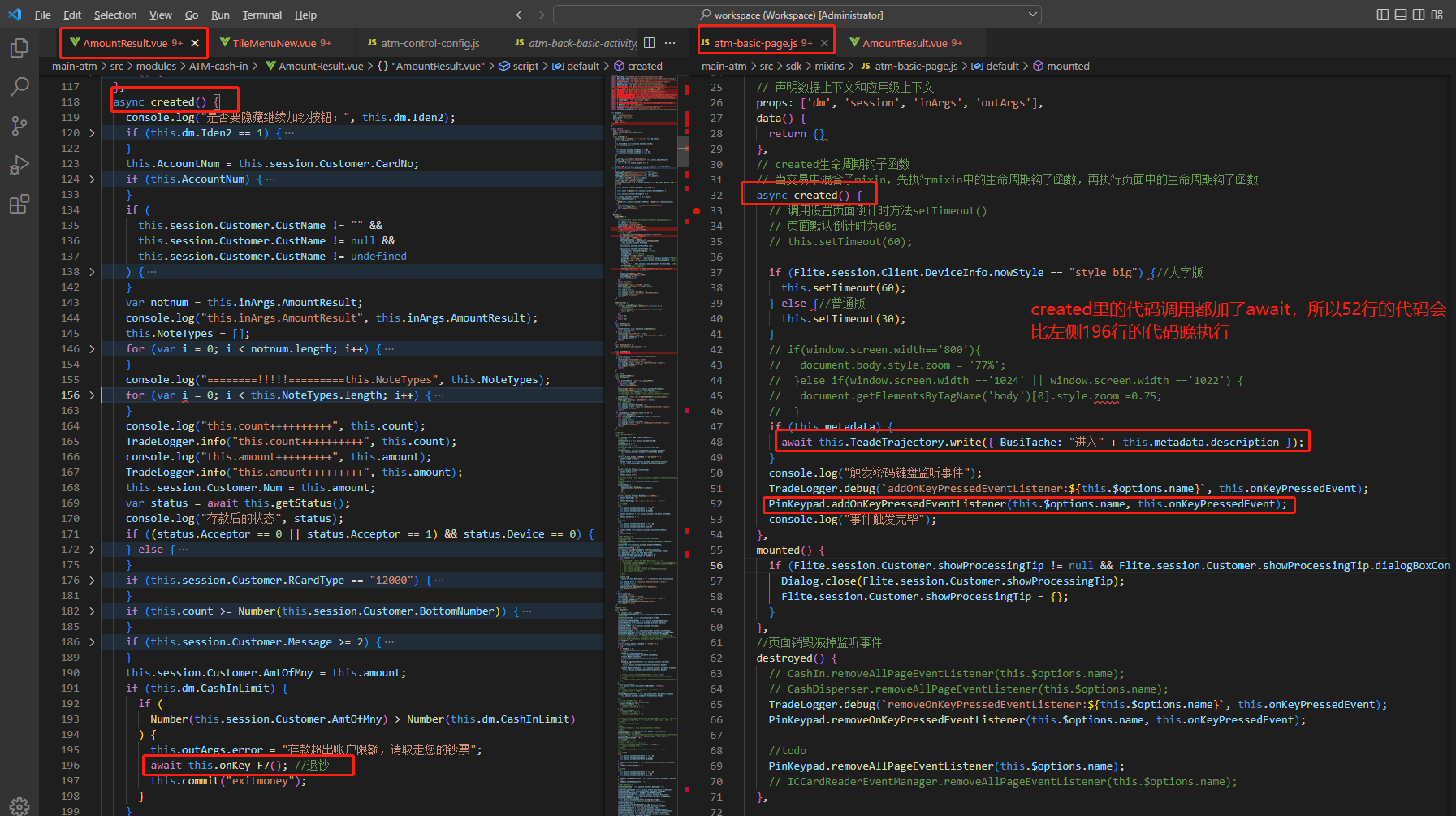
（2）Created里的代码执行加await，调用顺序如下：



如上图，加await的时候，右侧14行代码执行在左侧48行代码之后。

## 代码说明

所以我们回到生产问题的交易页面来查看代码，也是加了await执行的，如下图：



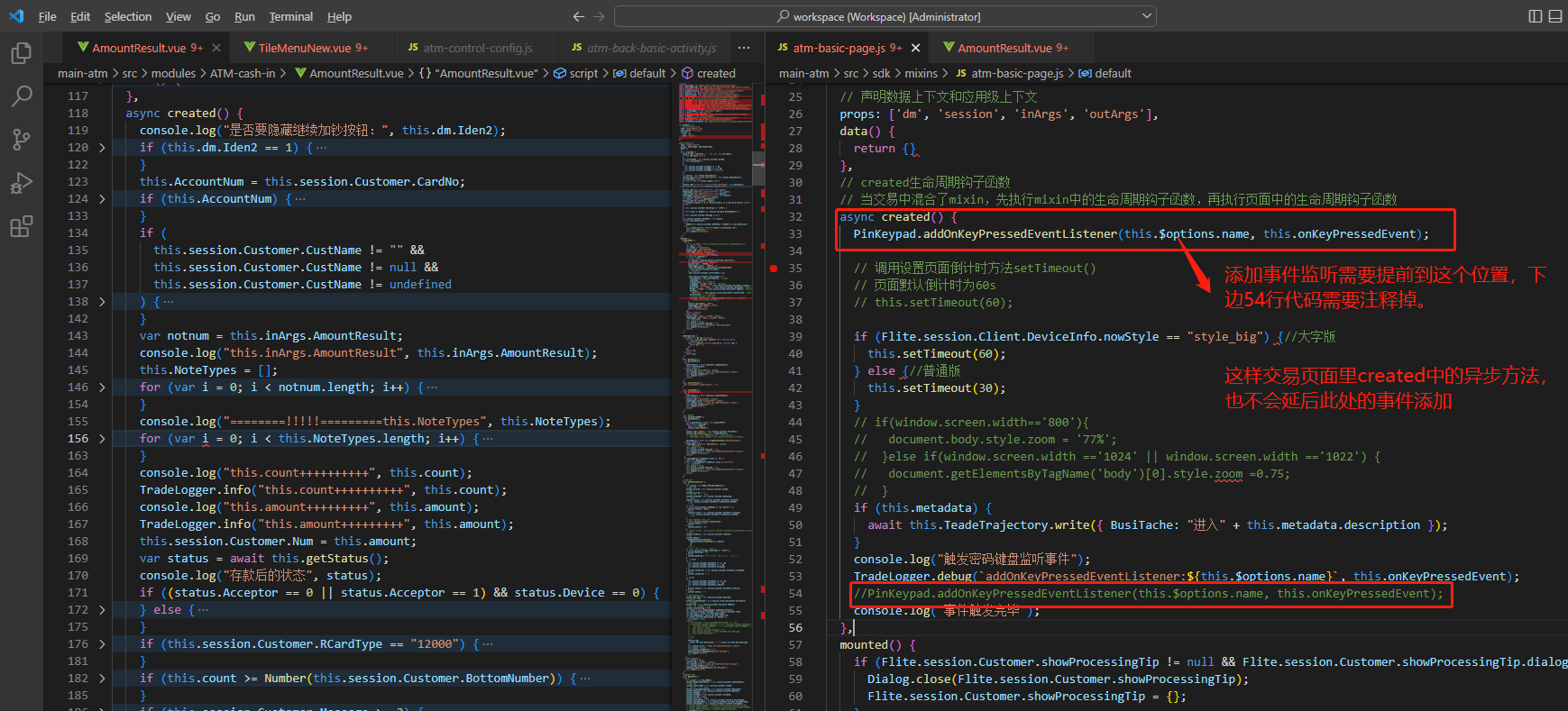
理论上，验钞结果页面（ResultAmount.vue）中mixins注入的created事件优先于页面的created执行。**但测试发现上图左侧（交易页面自己的created）created不会等到右侧（mixins注入进入的created）created结束之后再执行。**所以vue生命周期函数created方法上，其内部在调用mixin中的created方法和调用页面的created方法中，没有添加await，导致基类中created逻辑并没有完全执行完成就执行了页面中created。

总结：**created中代码调用加了await，会导致验钞页面（AmountResult.vue）中的异步方法执行结束之后，再执行事件监听。**

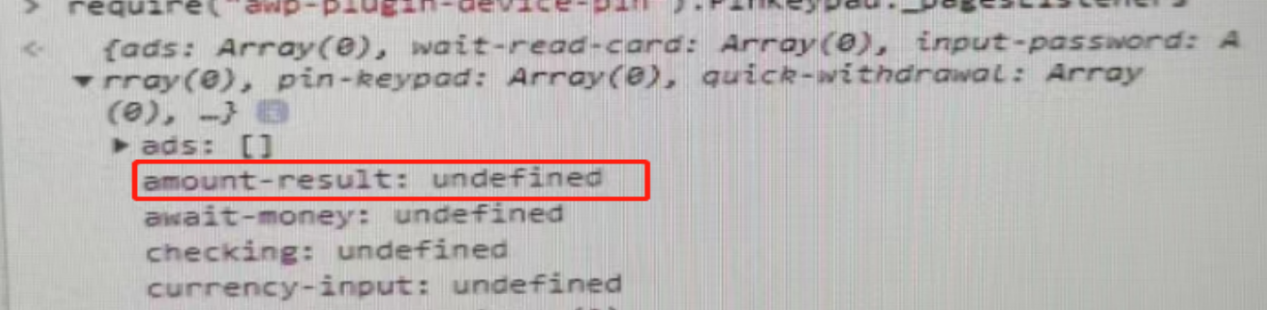
## 解决方案

开发代码需要注意，在页面创建时添加事件监听，页面销毁时移除事件监听。

海峡银行的代码里，销毁事件没有问题，添加事件执行的太晚，产生了运行时机偏差，所以出现了事件移除比事件添加先执行的情况。解决方案如下图：



将代码修改成上图之后，测试结果确认页面事件已经成功移除，如下图：



## 最终总结

问题发生时，执行的顺序：created调用->created中其他异步方法执行->AmountResult页面销毁->**执行destroyed->事件移除**->**created中调用添加事件**->其他逻辑。

由于created（里边添加事件），destroyed（里边移除事件）因为页面毫秒级别的停留（**created，destroyed两个方法调用的前后时间间隔很短**），**同时created里的事件添加并没有第一时间执行**，而是先走了其他异步逻辑之后才执行的事件添加，所以产生了时机问题，让事件添加晚于事件移除，导致了事件没有移除掉。