

RuoYi-Flowable-Plus

漏洞利用

分析

使用版本：v0.8.3

下载地址：<https://gitee.com/KonBAI-Q/ruoyi-flowable-plus>

安装：<http://konbai-q.gitee.io/ruoyi-flowable-plus-vuepress/document/%E5%BF%AB%E9%80%9F%E5%BC%80%E5%A7%8B/%E5%B7%A5%E7%A8%8B%E5%AF%BC%E5%85%A5.html%E5%AF%BC%E5%85%A5%E6%95%B0%E6%8D%AE%E5%BA%93>

漏洞利用

方法一（使用java类）：

条件：1.需要公共无参构造函数（因为使用了newInstance来实例化）

2.需要实现JavaDelegate接口，存在execute函数

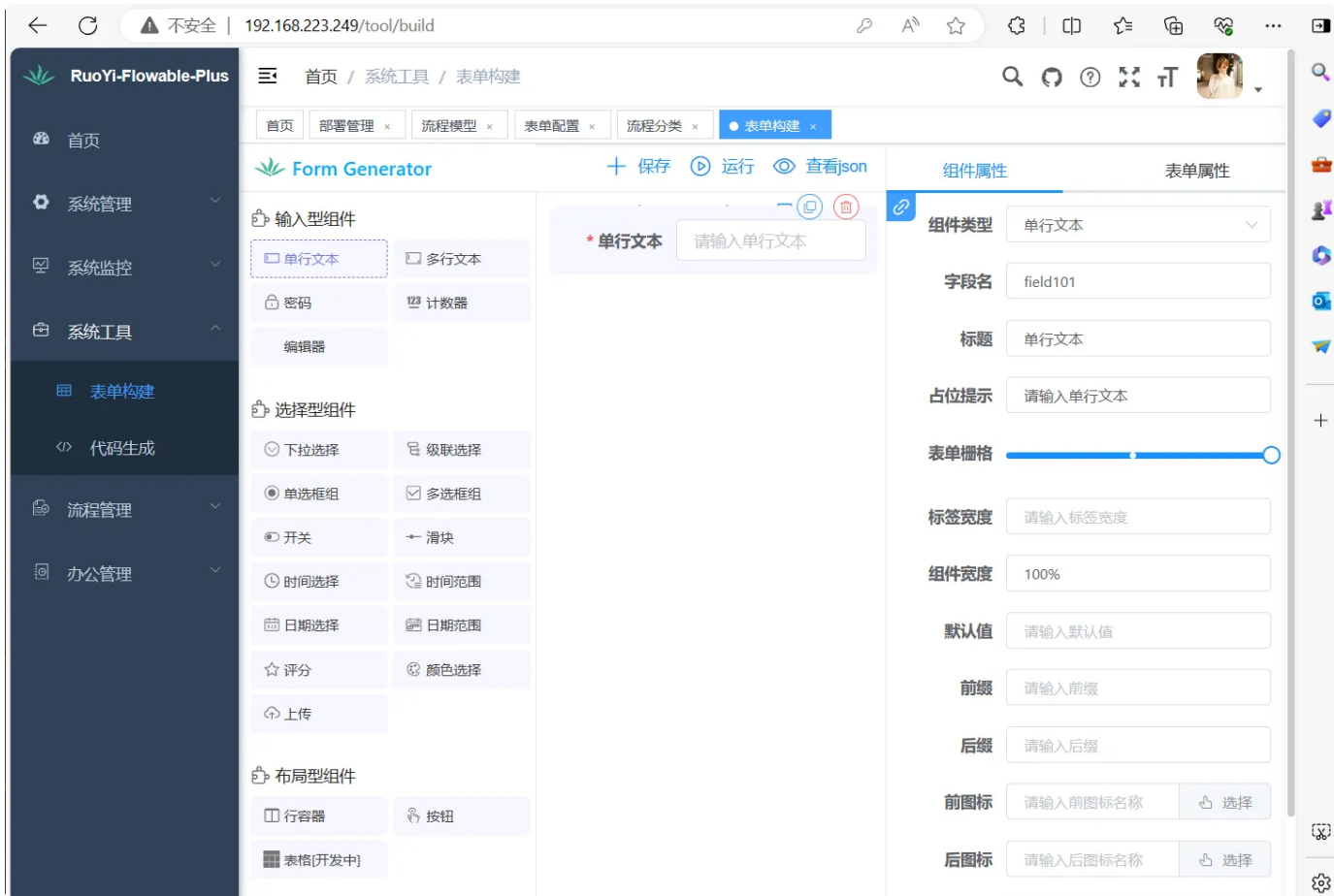
满足条件：

类一：org.flowable.engine.impl.test.NoOpServiceTask

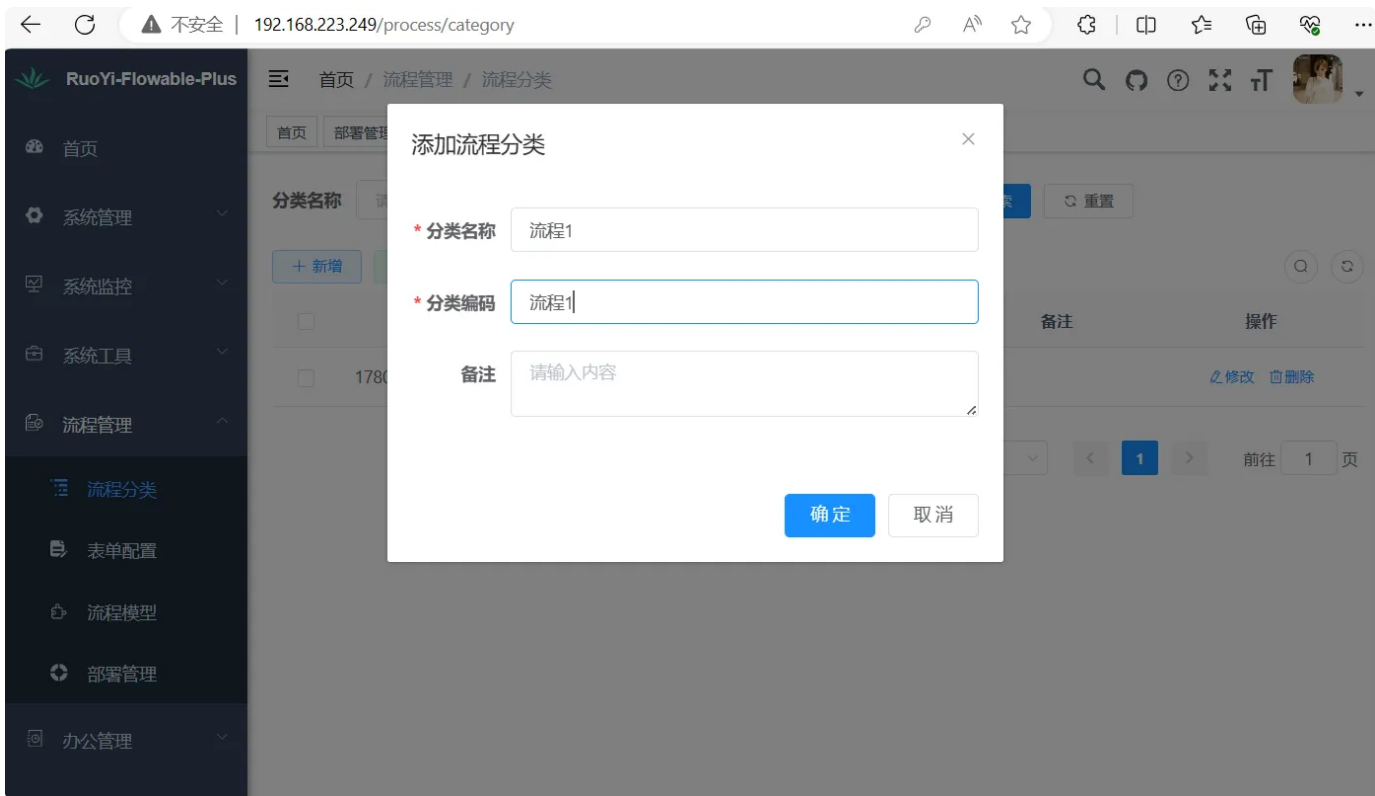
添加name表达式：

```
${''.getClass().forName('java.lang.Runtime').getMethod('getRuntime').invoke(null).exec('calc.exe')}
```

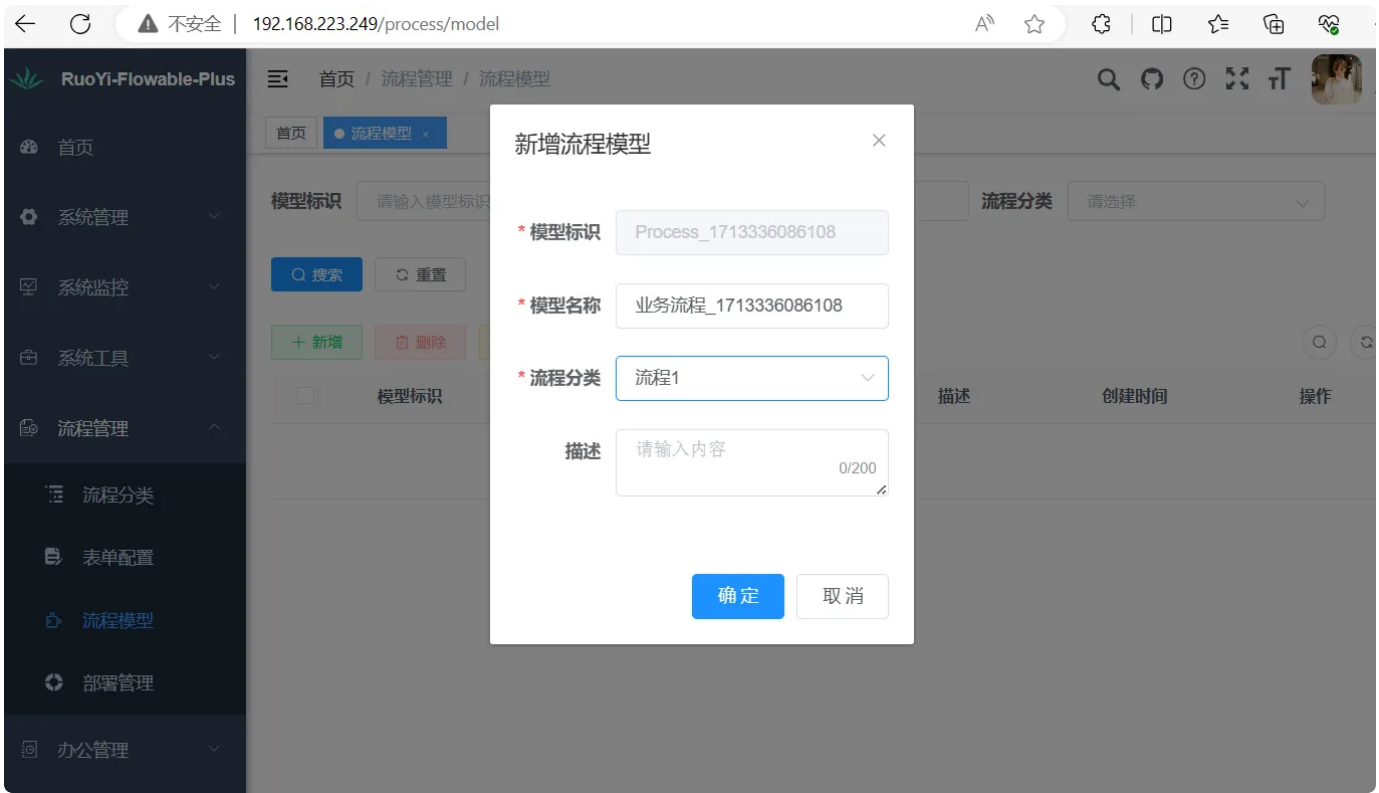
新建表单并保存，



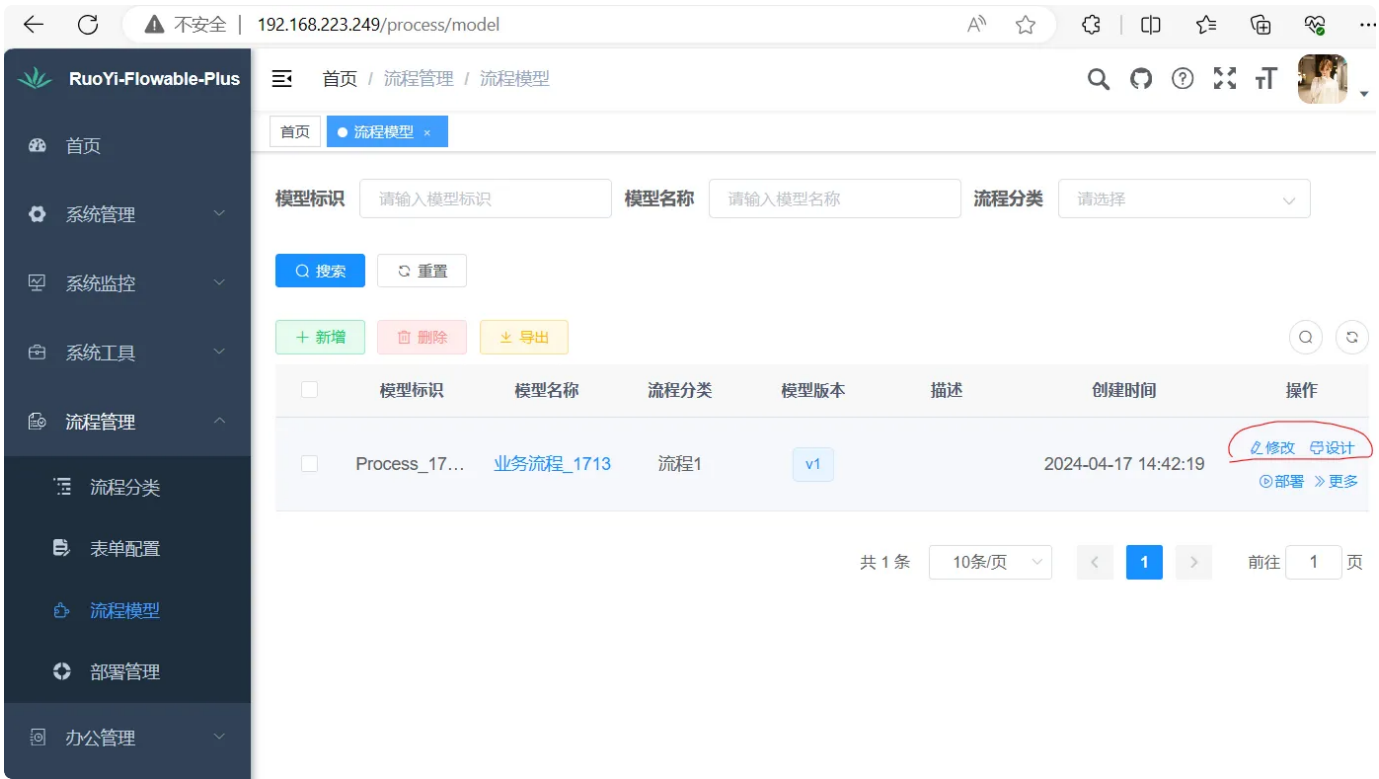
新增流程分类,



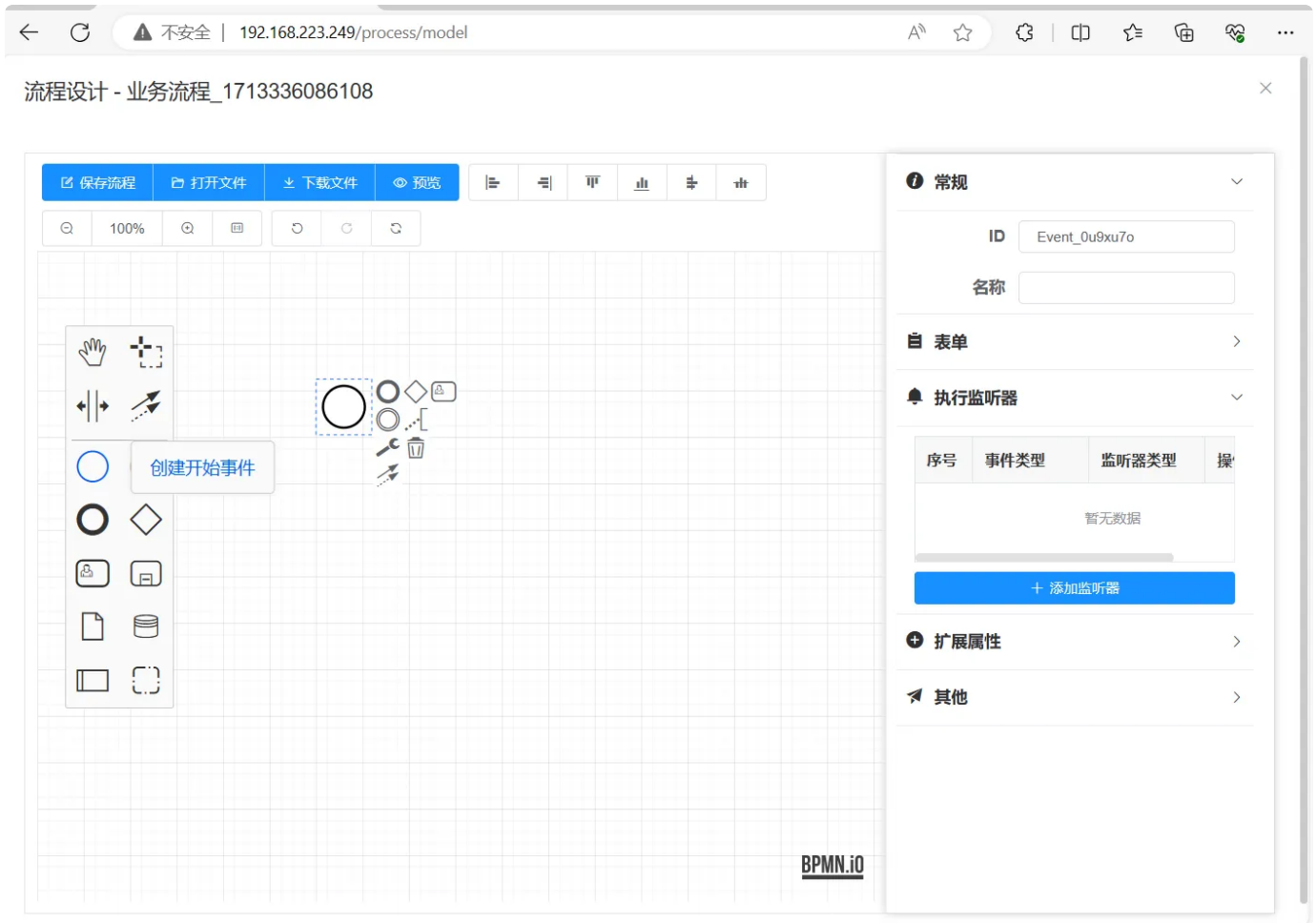
新增流程模型，



点击设计，



点击创建开始事件，并添加监听器，



设置监听器的内容,

浏览器地址栏显示：不安全 | 192.168.223.249/process/model

流程设计 - 业务流程_1713336086108

操作按钮：保存流程、打开文件、下载文件、预览

工具栏：100%、放大、缩小、重置

左侧工具栏包含：手形、移动、删除、复制、粘贴、撤销、重做、以及各种流程元素图标（圆、菱形、矩形、文档、数据库等）。

右侧配置面板：

执行监听器

- * 事件类型: start
- * 监听器类型: Java 类
- * Java类: org.flowable.engine.impl.test.NoOpServiceTask

注入字段: 添加字段

序号	字段名称	字段类型	字段值/表达式	操作
1	name	表达式	`\${"getClass().f...	编辑 移除

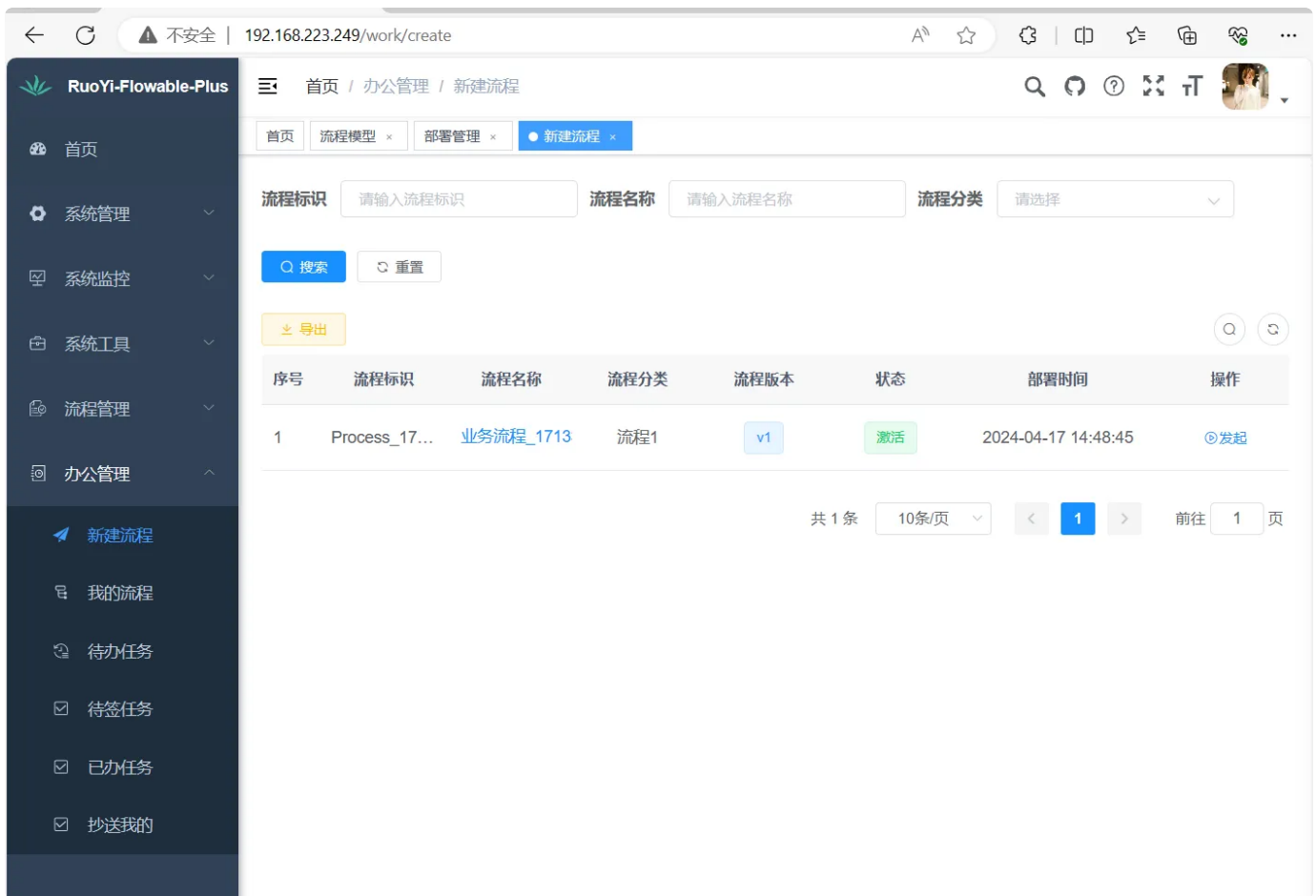
取消 保存

在选择我们之前新建的表单，

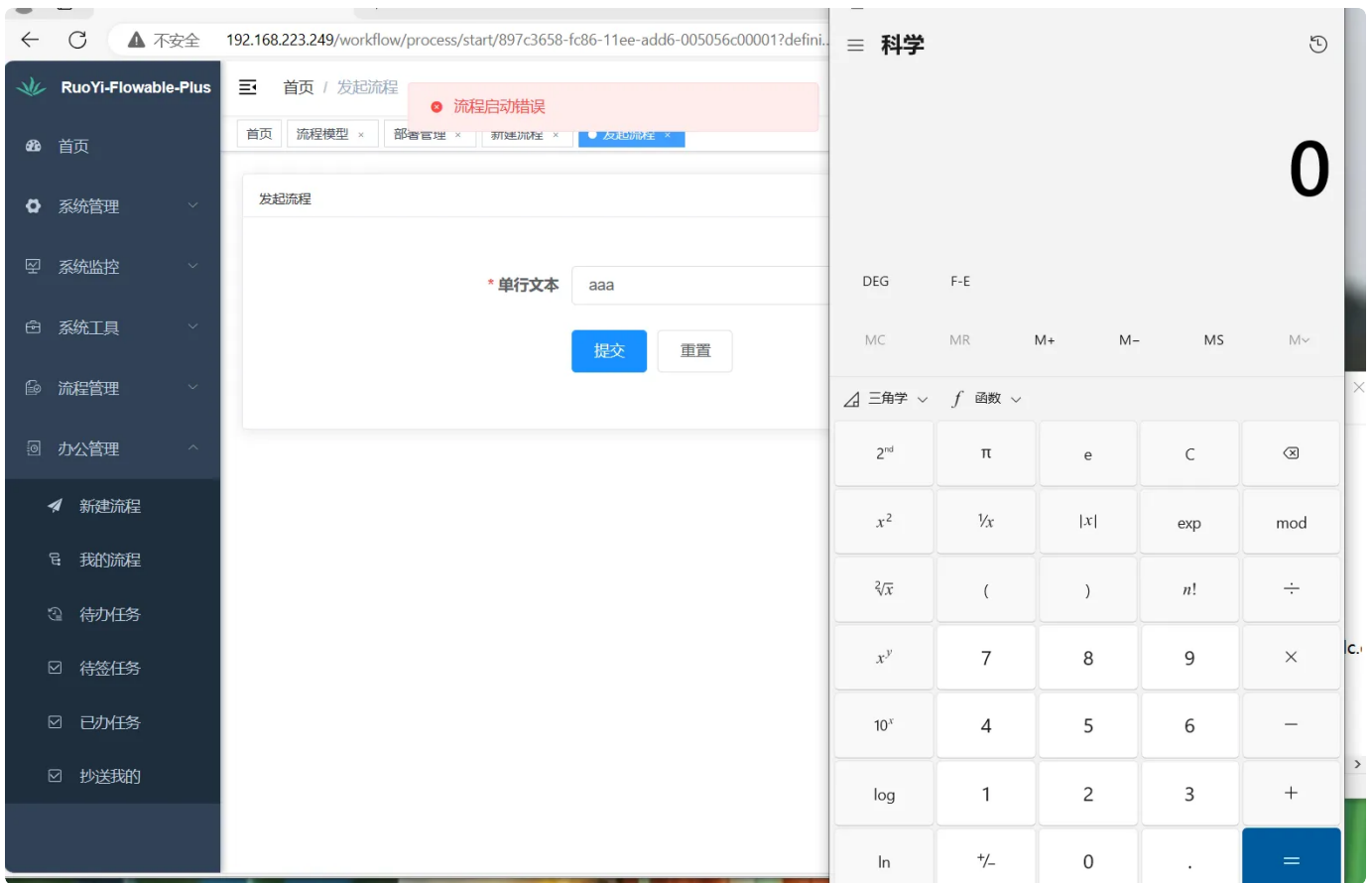
流程模型新增成功后，需要点击部署，



新建流程点击发起，



随意输入值点击提交，命令执行成功，

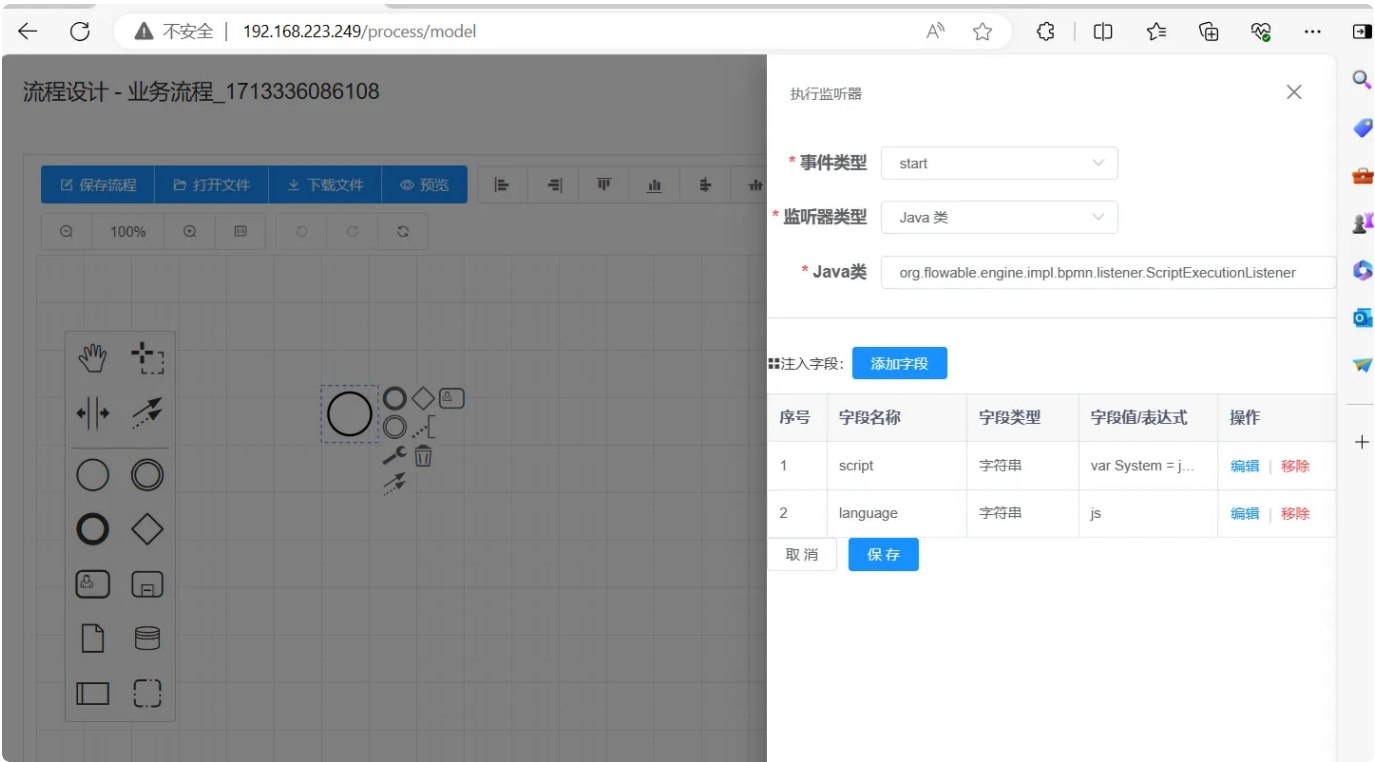


类二：org.flowable.engine.impl.bpmn.listener.ScriptExecutionListener

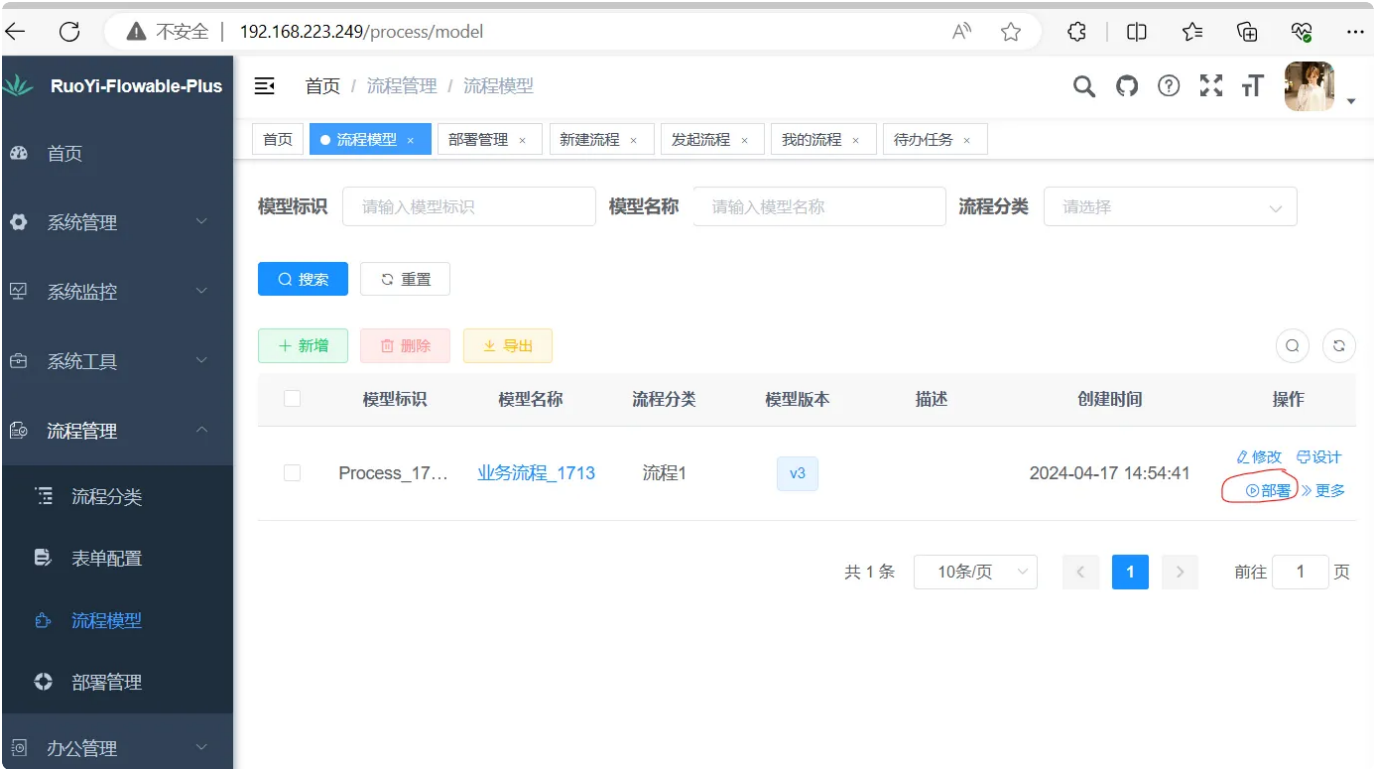
script: var System = java.lang.Runtime.getRuntime().exec("calc");

language: js

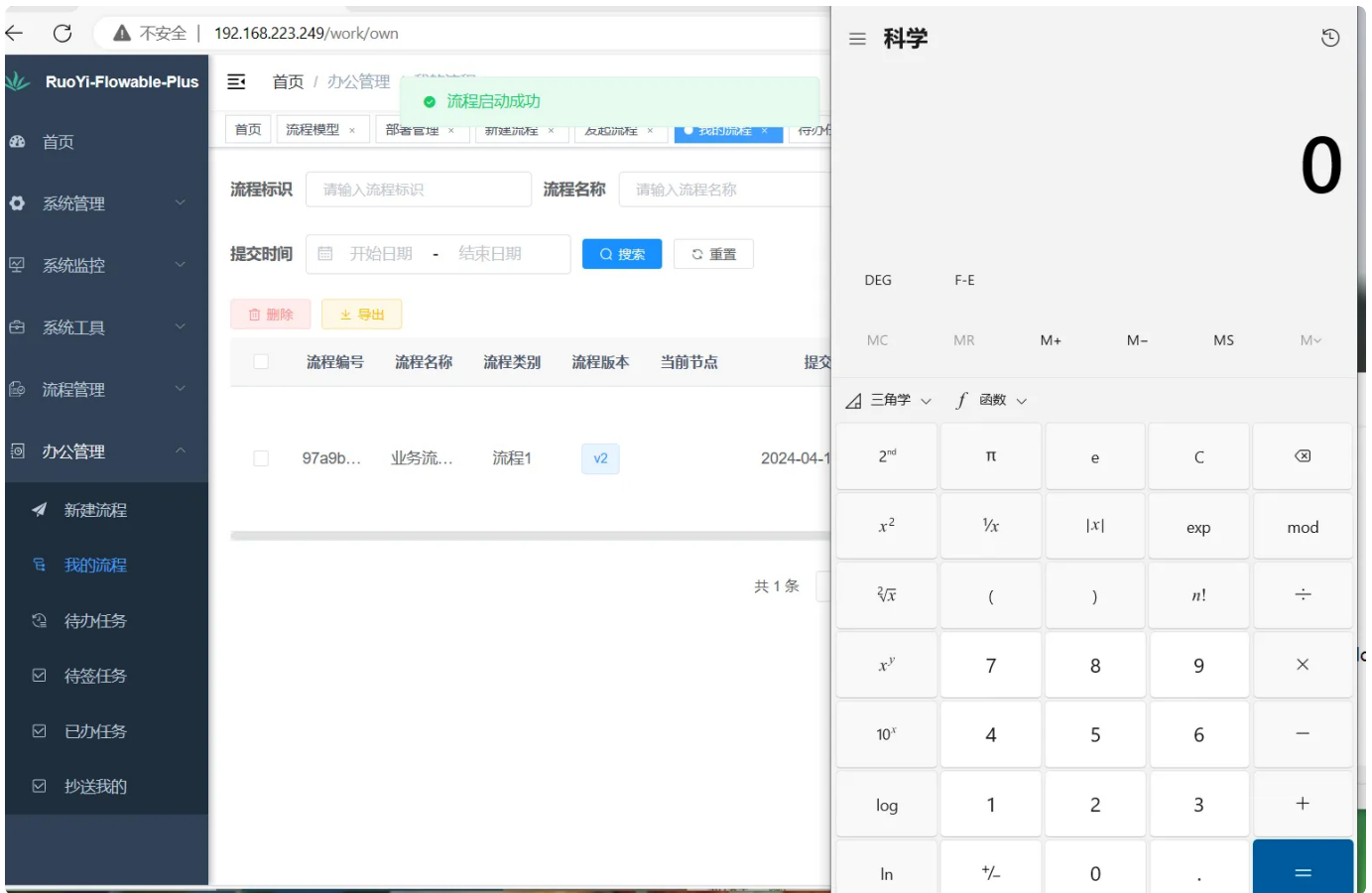
将流程模型里面的监听器改为以下内容，



需要点击部署（不然流程模型没有被更改），



新建流程中发起，点击提交成功触发漏洞，



方法二（使用表达式或者代理表达式）：

```
`${''.getClass().forName('java.lang.Runtime').getMethod('getRuntime').invoke(null).exec('calc.exe')}
```

流程设计 - 业务流程_1713336086108

保存流程

打开文件

下载文件

预览

100%

手

十字

左右

上下

圆

圆环

粗圆

菱形

人

库

文件

数据库

矩形

虚线

圆

圆环

粗圆

菱形

人

库

文件

数据库

矩形

虚线

执行监听器

* 事件类型

start

* 监听器类型

表达式

* 表达式

`$_getClass().forName('java.lang.Runtime').getMethod('getRuni`

注入字段:

添加字段

序号	字段名称	字段类型	字段值/表达式	操作
暂无数据				

取消

保存

← ↻ ⚠ 不安全 | 192.168.223.249/process/model

流程设计 - 业务流程_1713336086108

保存流程 打开文件 下载文件 预览

100%





执行监听器

* 事件类型

start

* 监听器类型

代理表达式

* 代理表达式

Runtime.getRuntime().getMethod("getRuntime").invoke(null).exec("calc.exe")

注入字段:

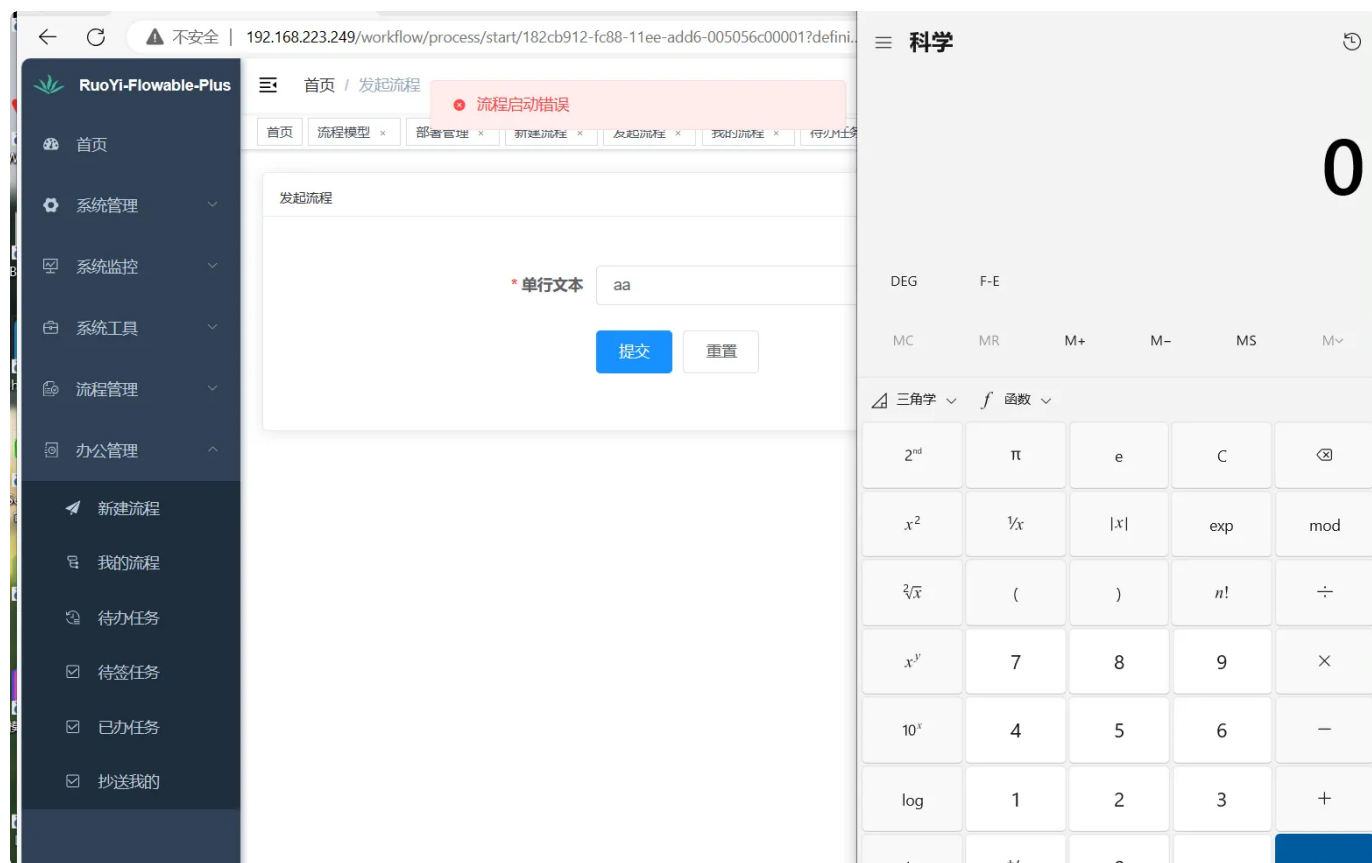
添加字段

序号	字段名称	字段类型	字段值/表达式	操作
暂无数据				

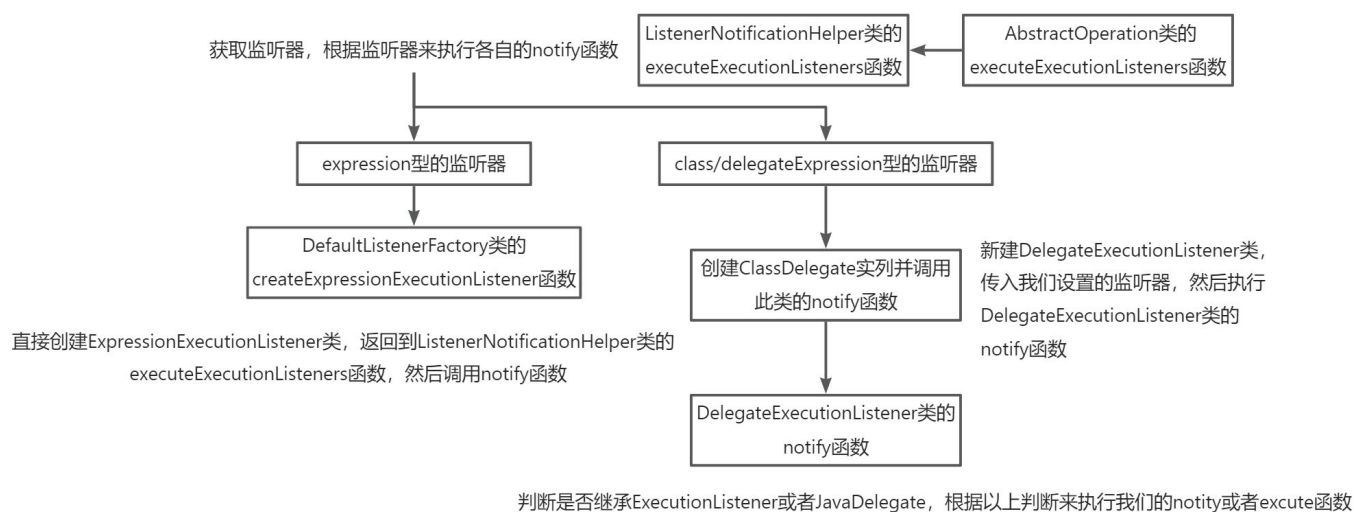
取消

保存

点击部署后，发起流程，命令执行成功，

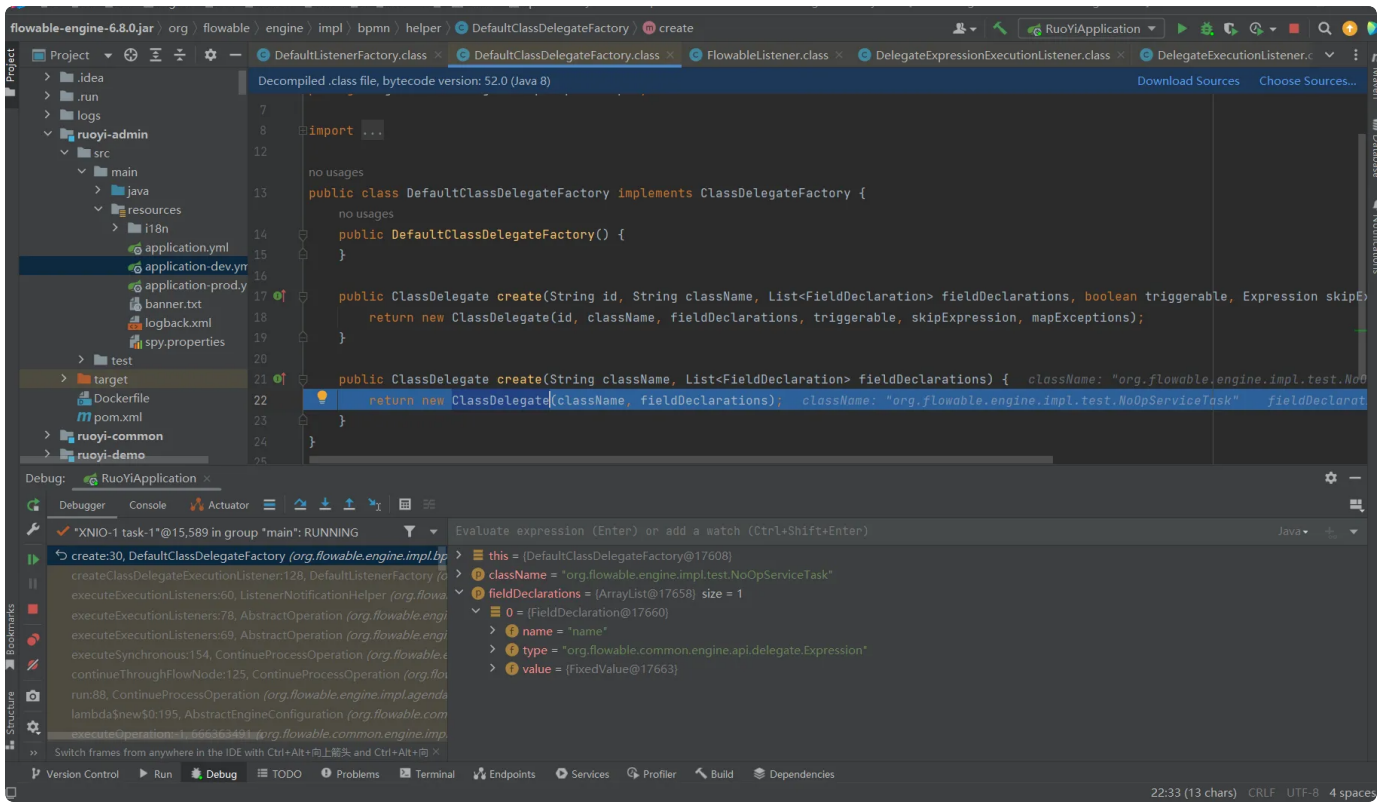


分析



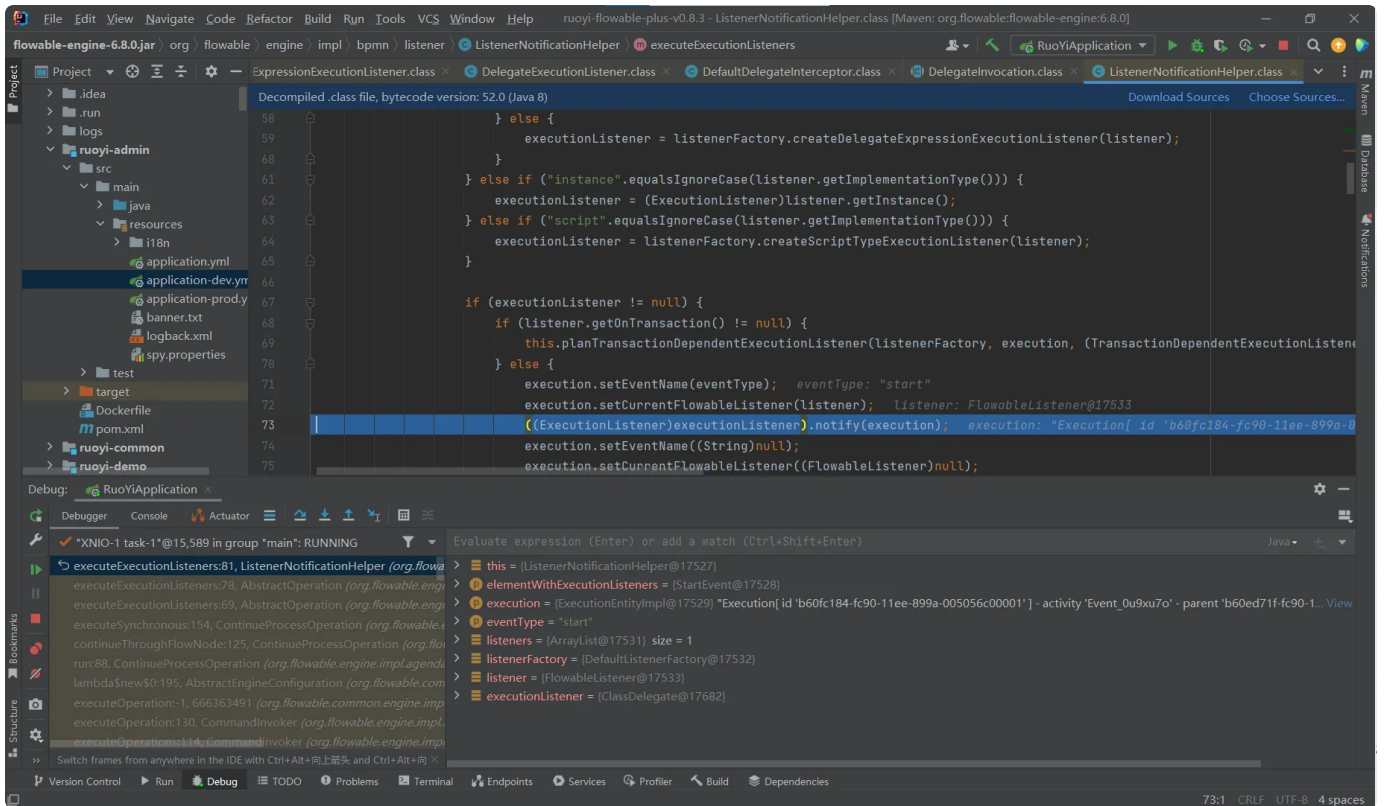
使用java类（代理表达式分析于java类利用方式相似）

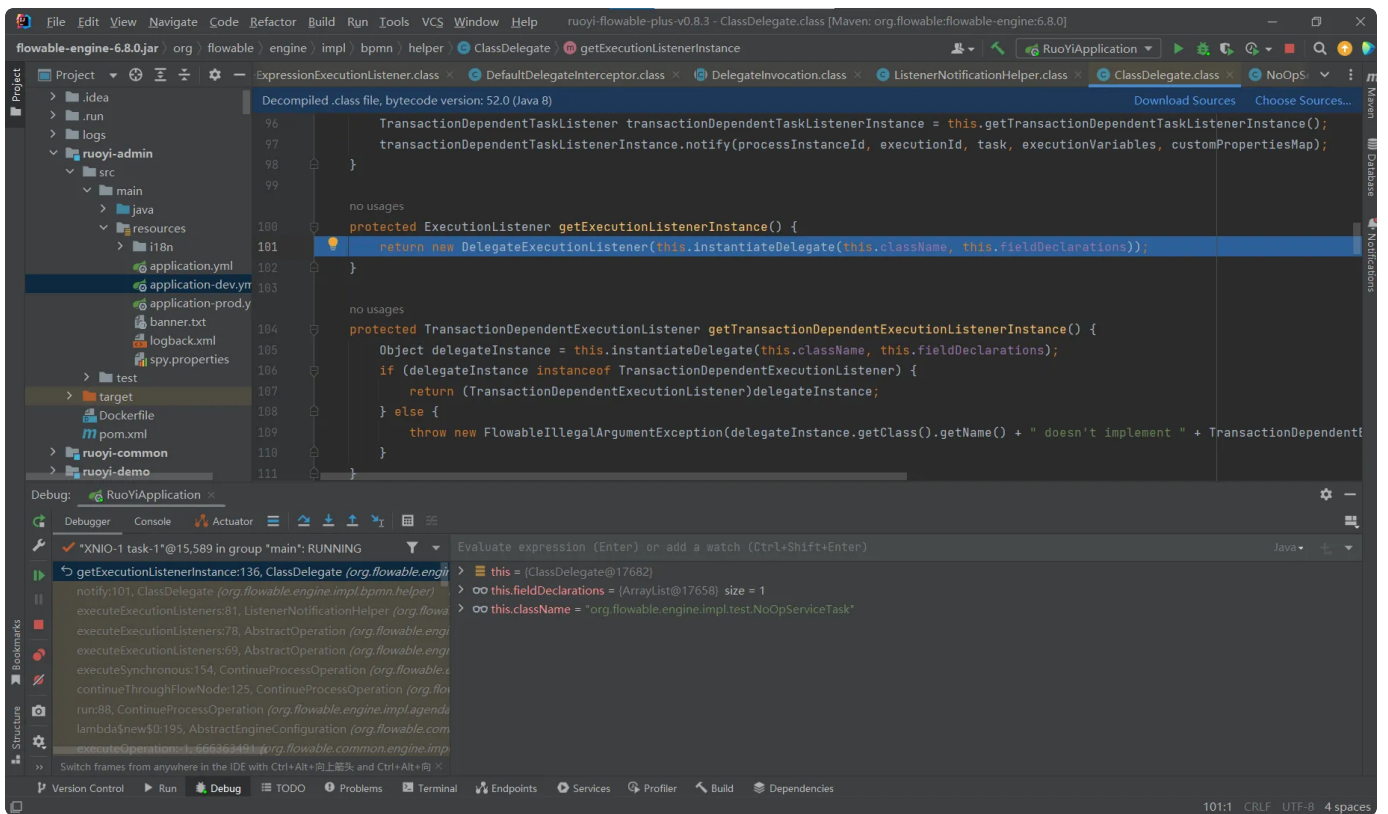
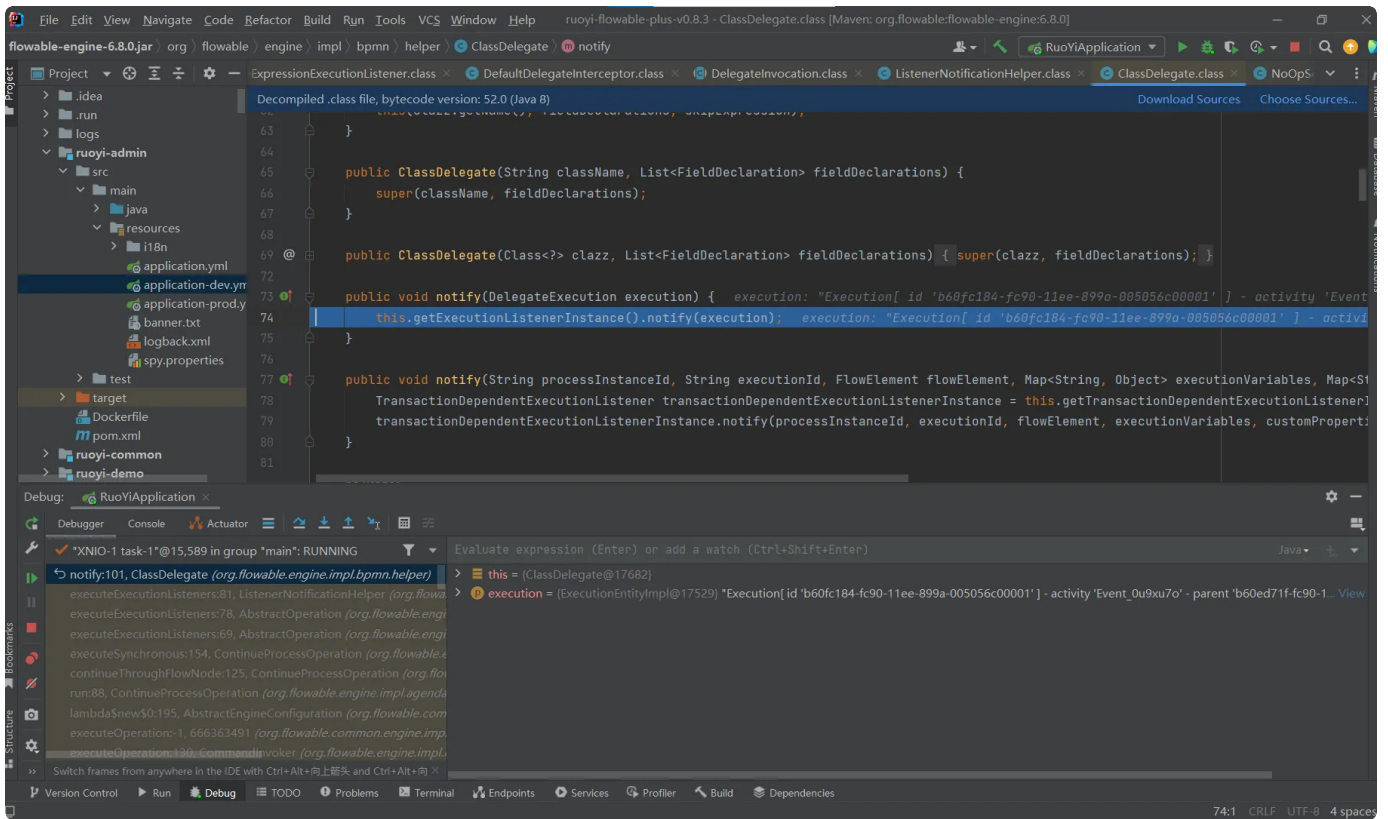
org.flowable.engine.impl.bpmn.listener.ListenerNotificationHelper#executeExecutionListeners

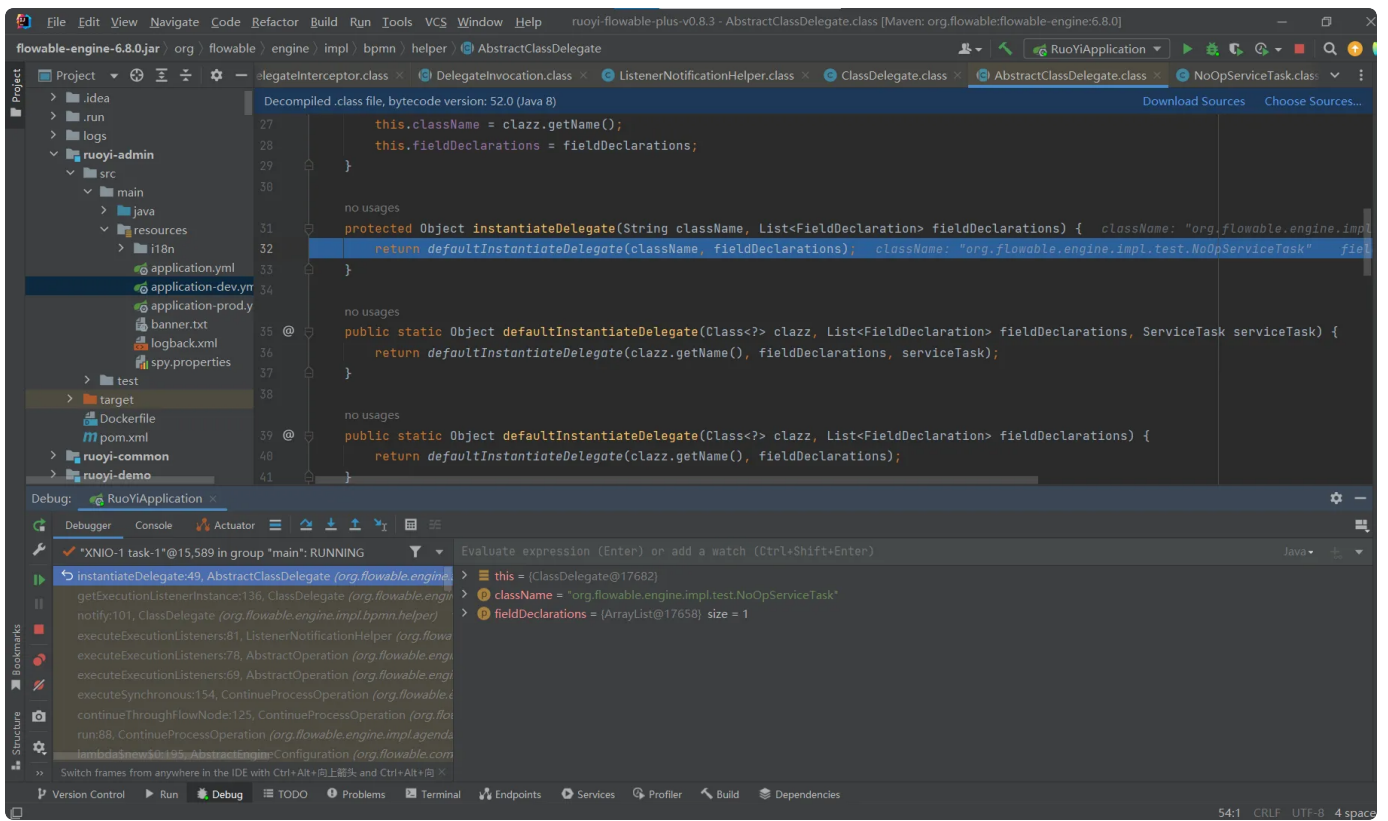


返回到

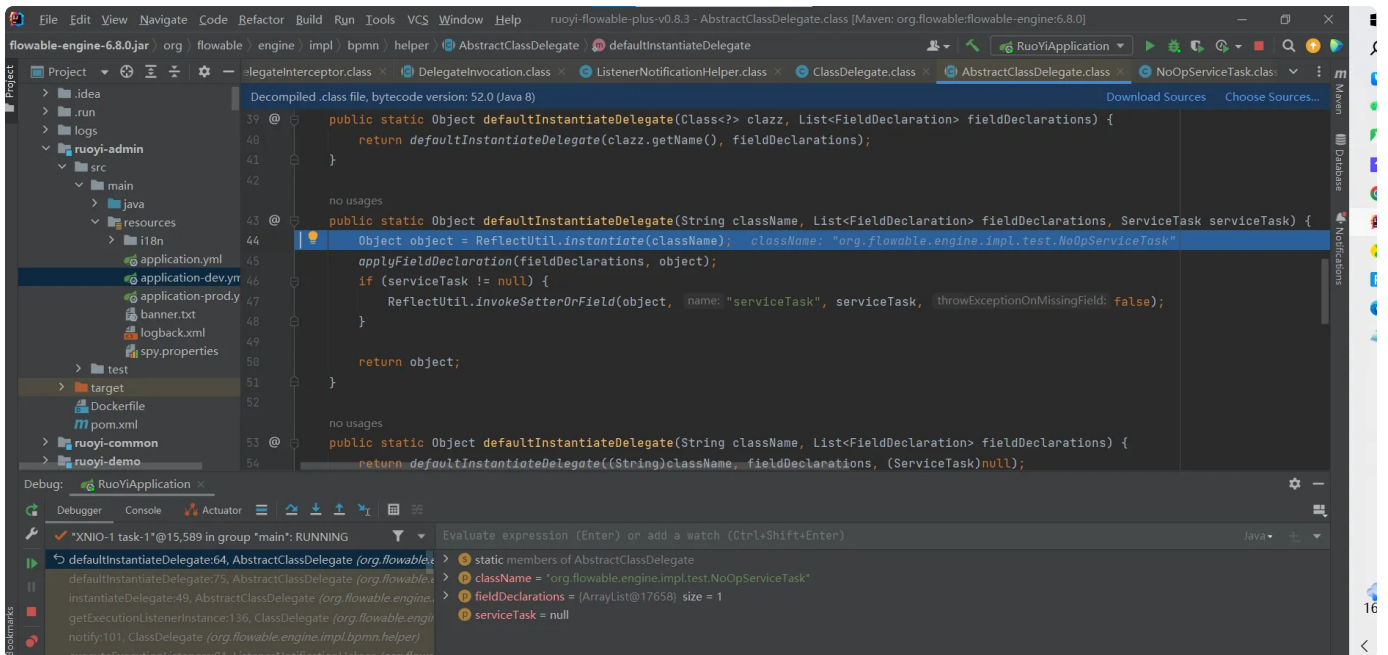
org.flowable.engine.impl.bpmn.listener.ListenerNotificationHelper#executeExecutionListeners,
执行ClassDelegate类notify函数,





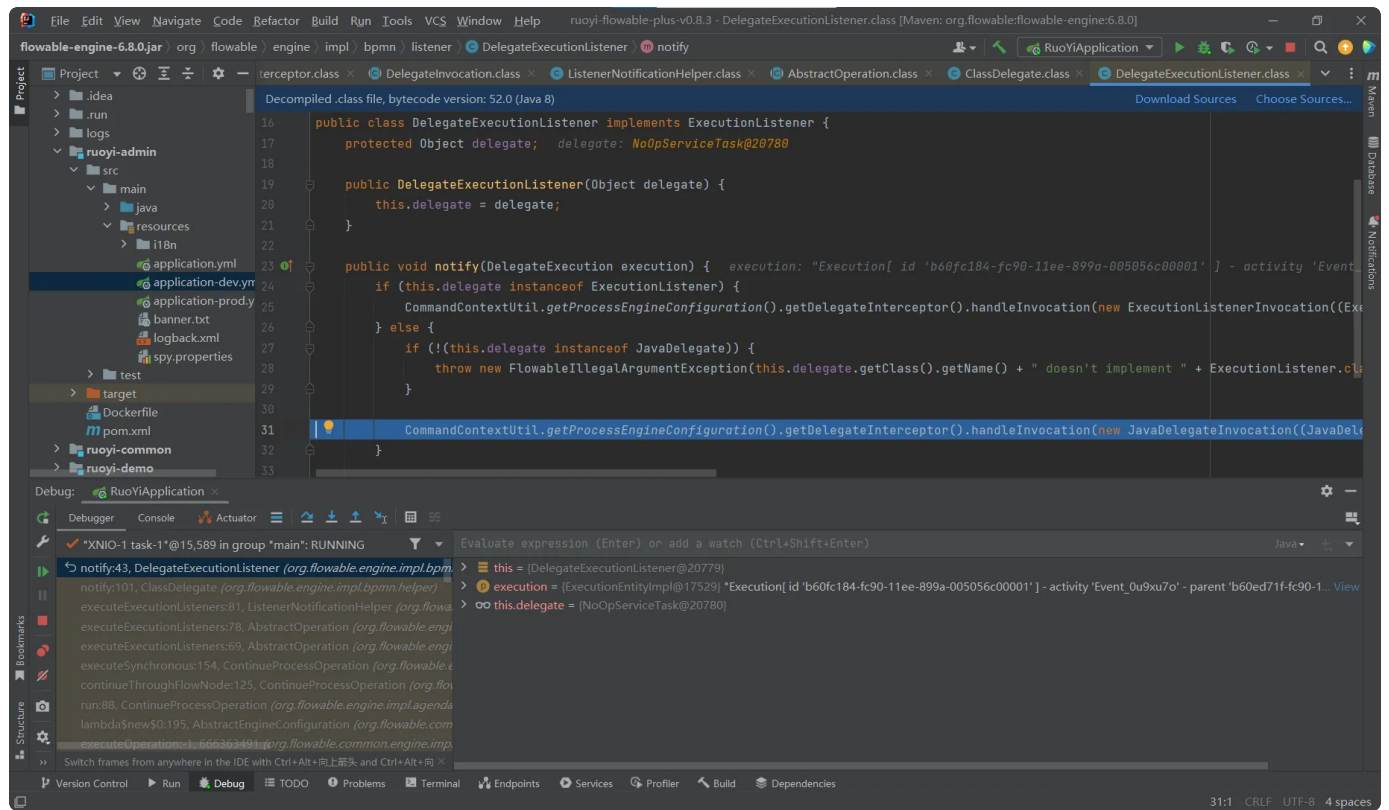


最终执行到defaultInstantiateDelegate函数，实例化我们的监听器，

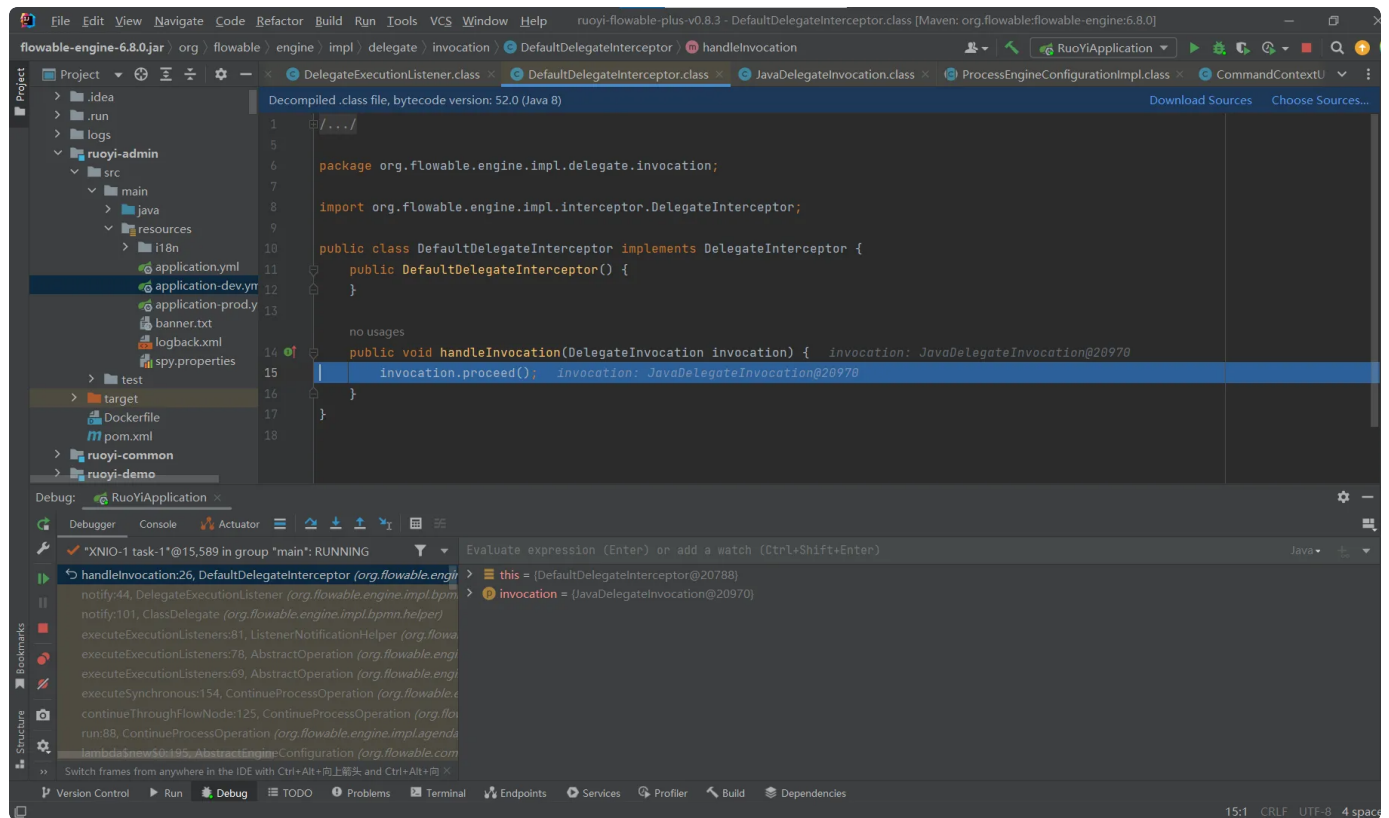


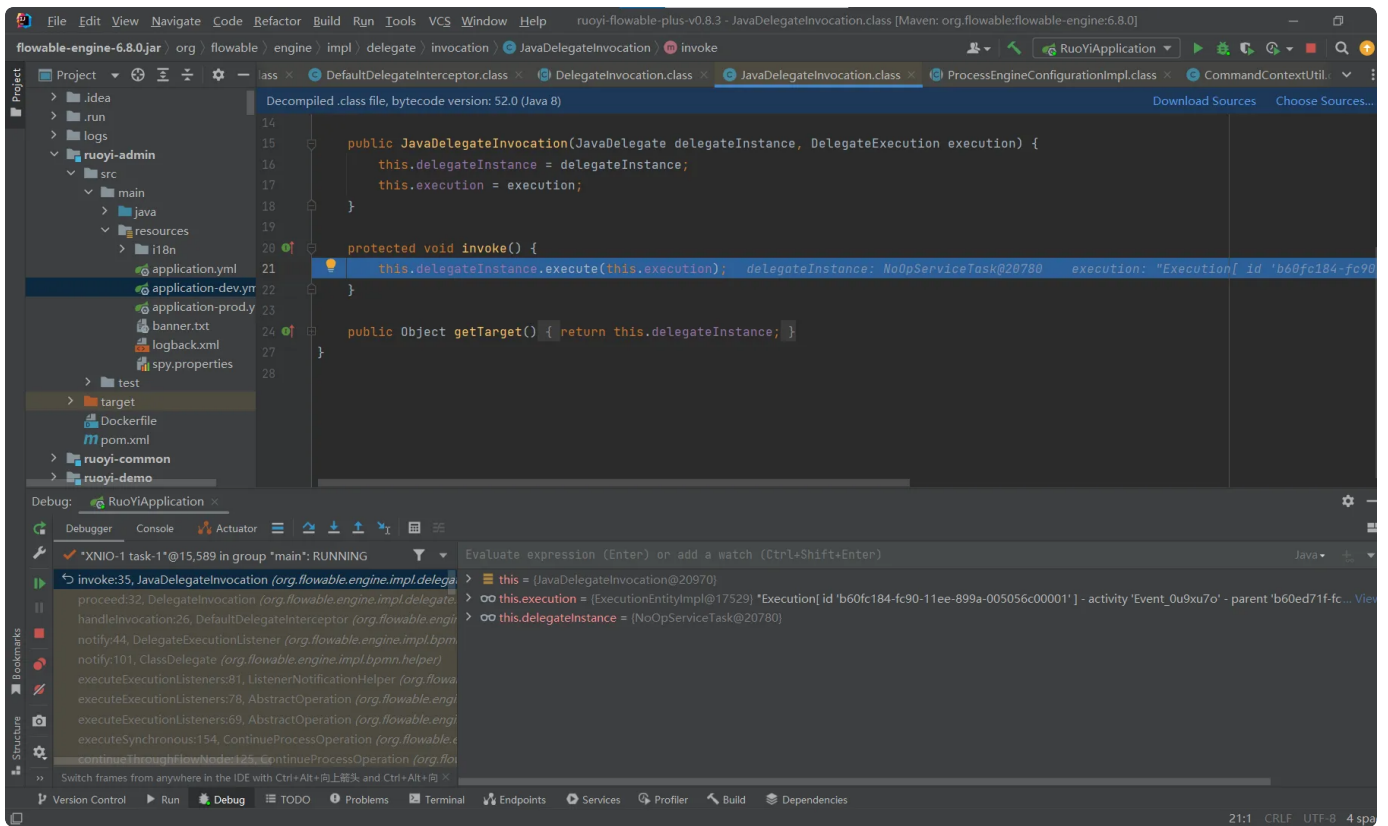
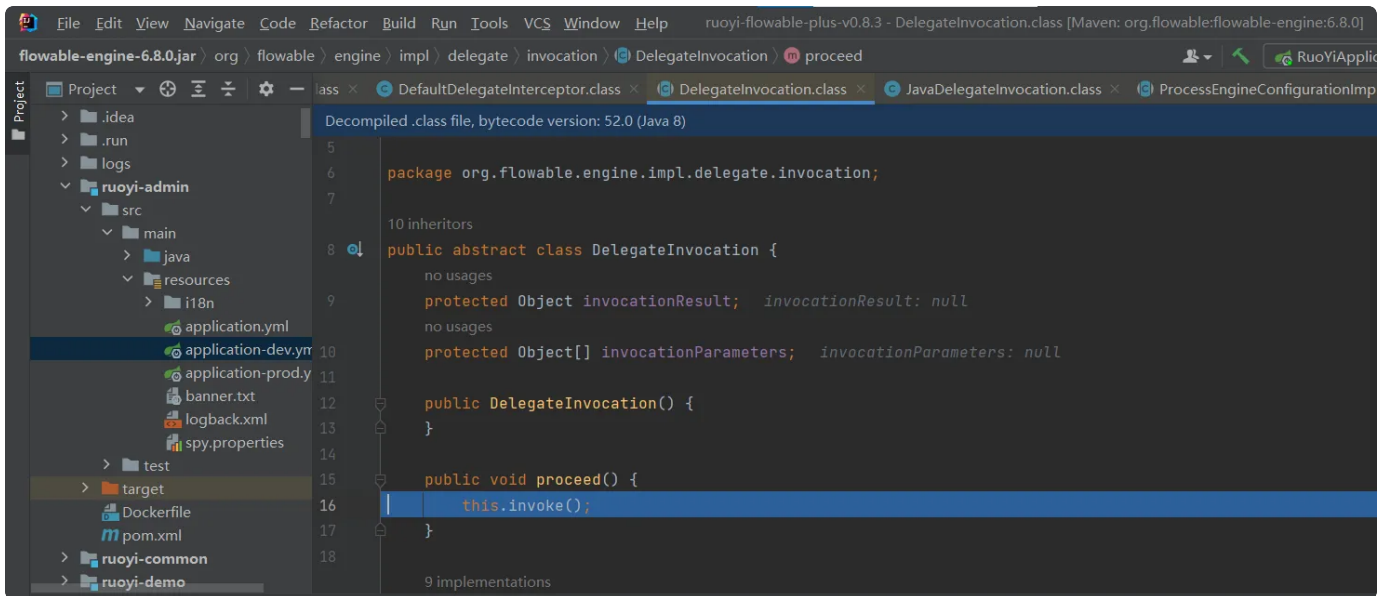
然后返回到DelegateExecutionListener的构造函数中，

注意，这里需要满足实现JavaDelegate接口或者ExcutionListener接口，

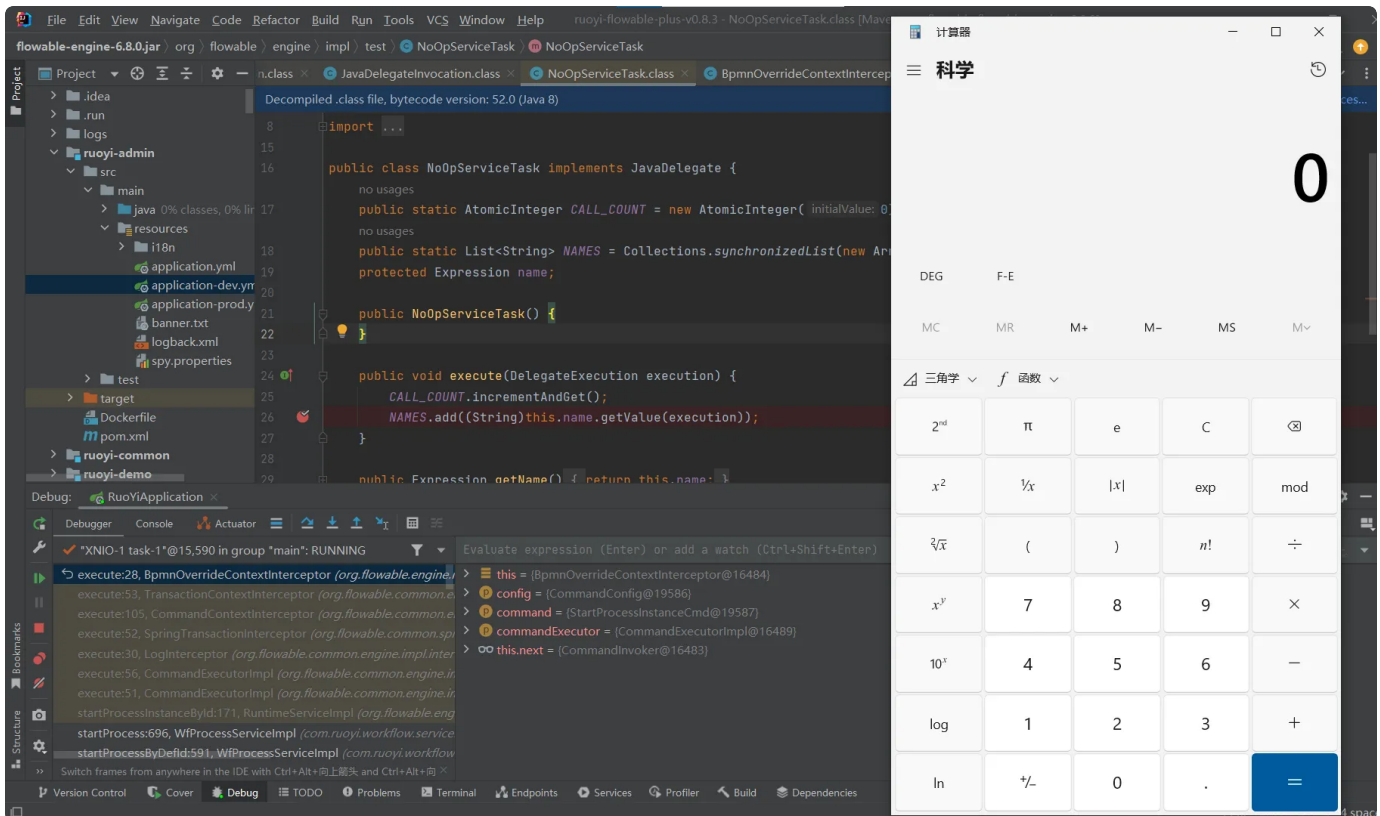
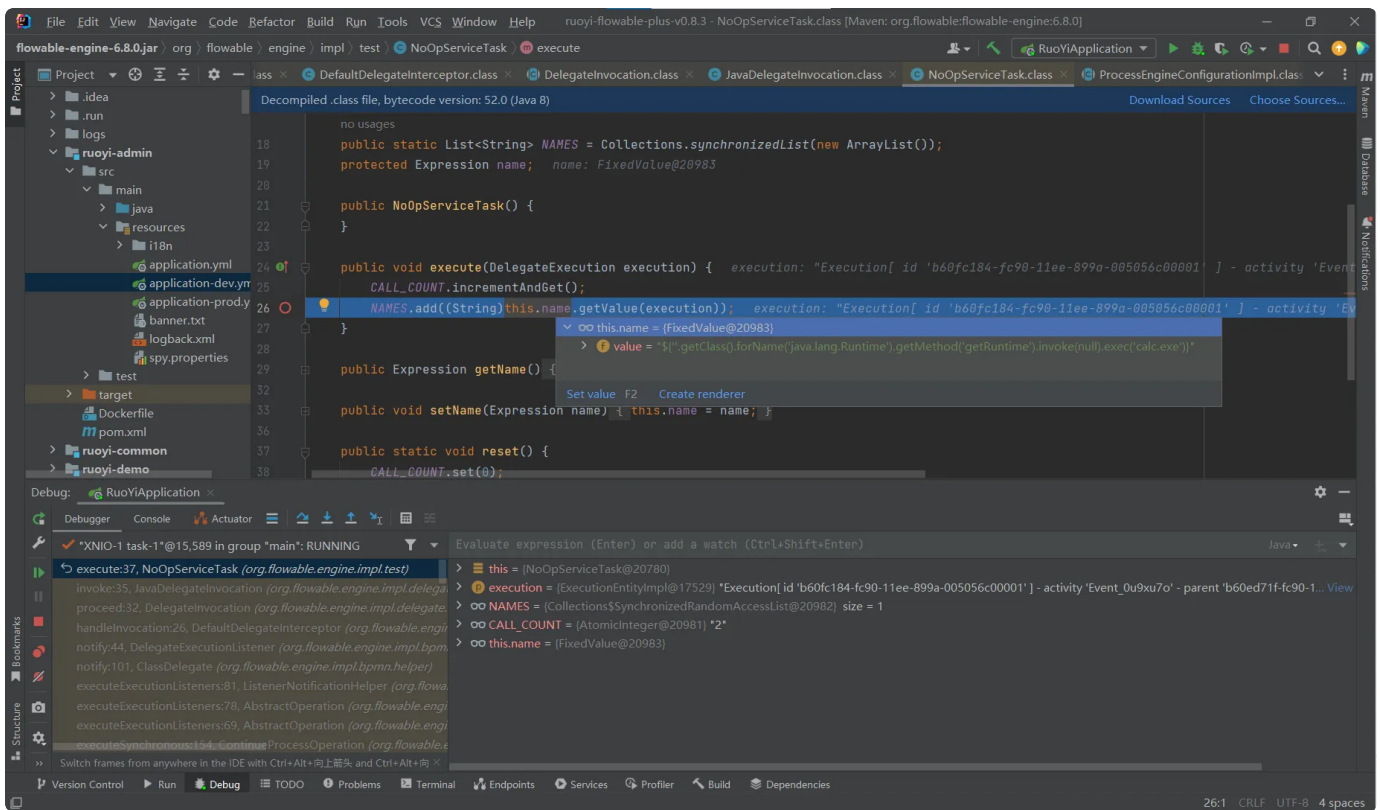


之后经过一系列函数调用，





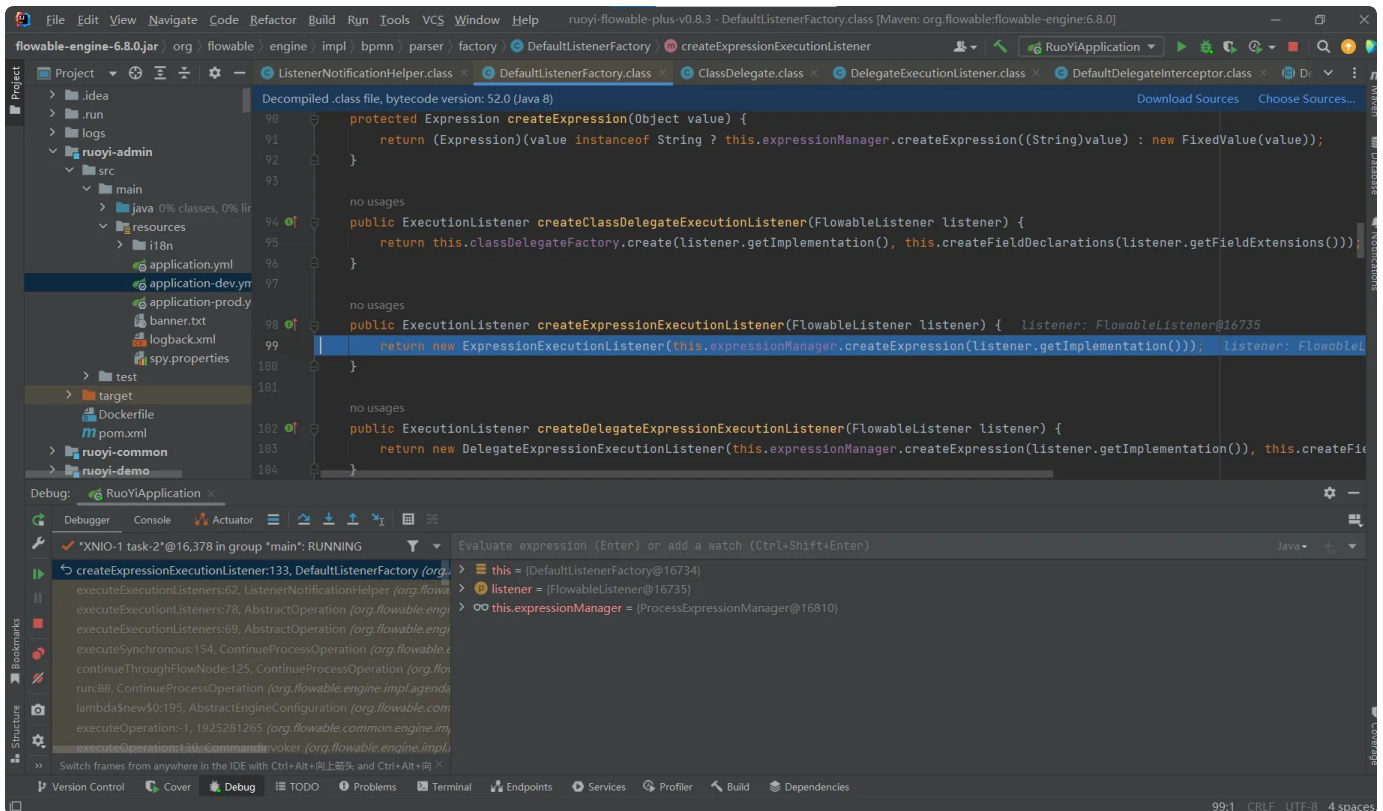
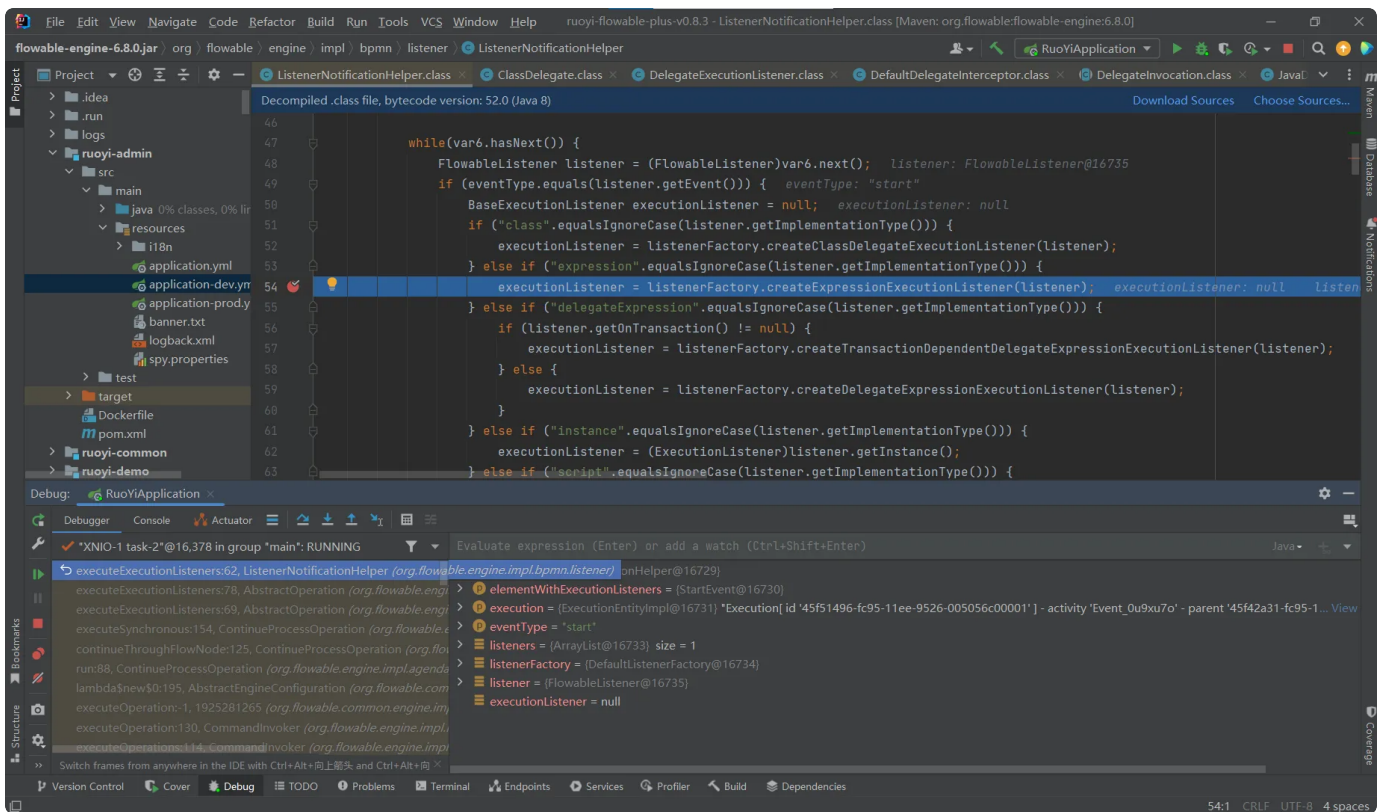
最终解析表达式，命令执行成功，



使用表达式

org.flowable.engine.impl.bpmn.listener.ListenerNotificationHelper#executeExecutionListeners

这里直接创建ExpressionExecutionListener类，



然后调用ExpressionExecutionListener类的notify函数，最终解析表达式，命令执行成功，

