驰骋BPM低代码二开

# 概述

## 基本说明

* 二开主要是流程、表单部分的二开，二开的接口是针对事件的开发，所谓的事件就是用户触发的动作。
* 流程事件比如：发送前、发送成功时、发送失败时、退回前、退回后、移交前、移交后、流程结束前、结束后。
* 单据实体事件：保存前、保存后、归当前、归档后、撤销归档前、撤销归档后。
* 表单事件比如：表单装载前、装载后、文本框数值变化、下拉框数值变化。
* 以上的事件激活的代码脚本触发业务逻辑。

## 方式选择

* 实现用户的业务需求我们提供了多种方式，技术人员可以根据自己的技术储备与偏好选择适合自己的方式。
* 比如：一个请假流程，发起流程做可休假天数判断，可以使用前端外挂、后端外挂、页面设置、流程全局事件类4中方式实现，这些方式后面章节都会讲解到。
* **实施人员：**如果您是实施人员会SQL存储过程可以使用事件配置方式实现，使用界面化的设置。
* **前端人员：**使用Typescript那就使用前端外挂实现，也可以使用全局TS流程事件类实现。
* **后端人员：**可以选择使用后端外挂实现，也可以使用重写java/C#全局事件类实现。

## 应用主题

* **流程：** 发送前、发送失败时、发送成功时、退回前、退回后、删除前、删除后、流程结束前、流程结束后。
* **单据：**提交审核前、归档前、归档后、撤销归档前、撤销归档后。
* **实体：**归档前、归档后、装载前、装载后。
* **高代码TS实体：**查询组件、单记录组件。

# 界面配置

* 顾名思义，就是使用页面配置的模式实现业务逻辑的交互。
* ccbpm把业务逻辑或者说规则存储到数据表里实现的一种方式。
* 执行方式：SQL、WebApi、业务单元等。
* 适合实施人员界面配置。

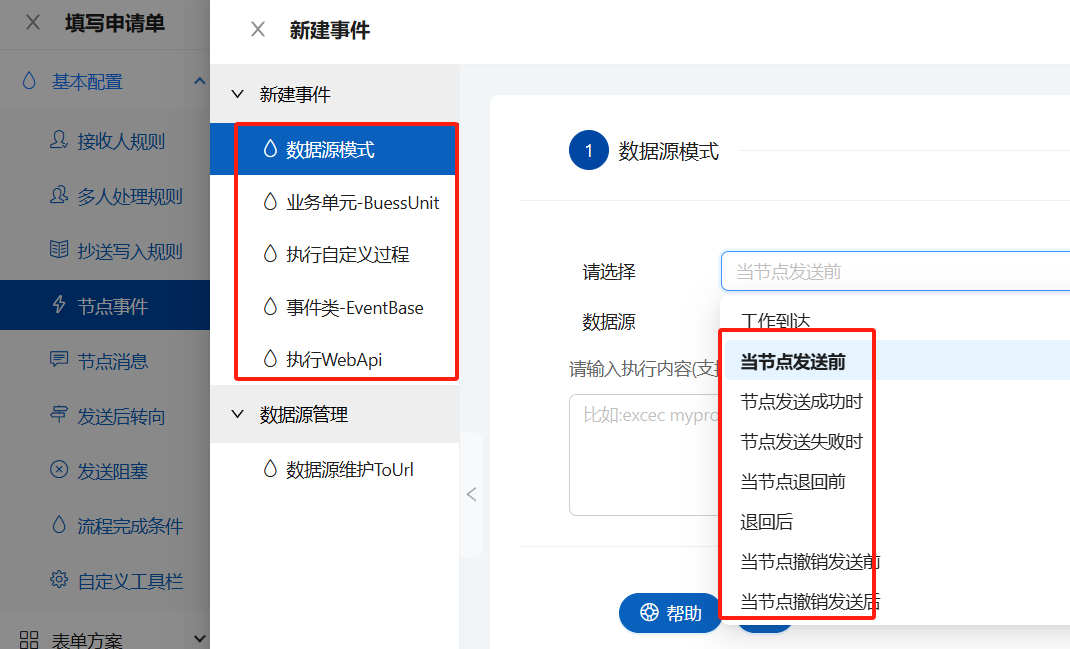
## 流程

* 流程图

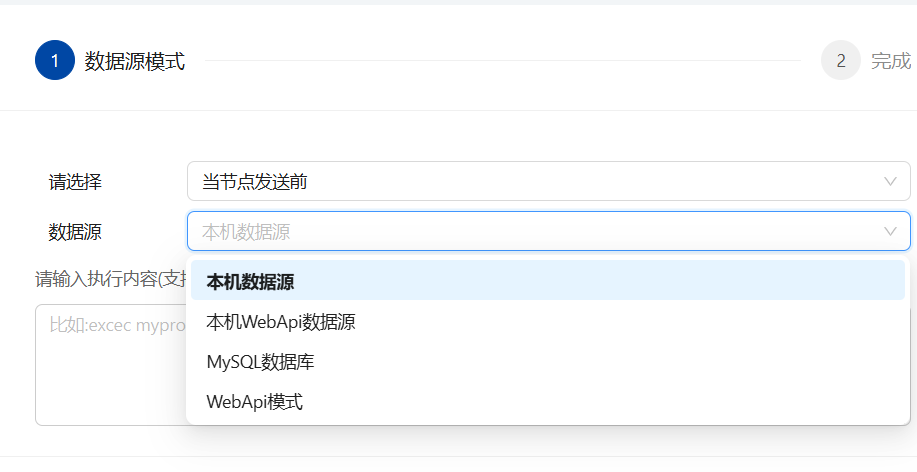


### 节点事件

* 在节点上点击右键，选择节点属性，选择节点事件进入节点事件的设置。

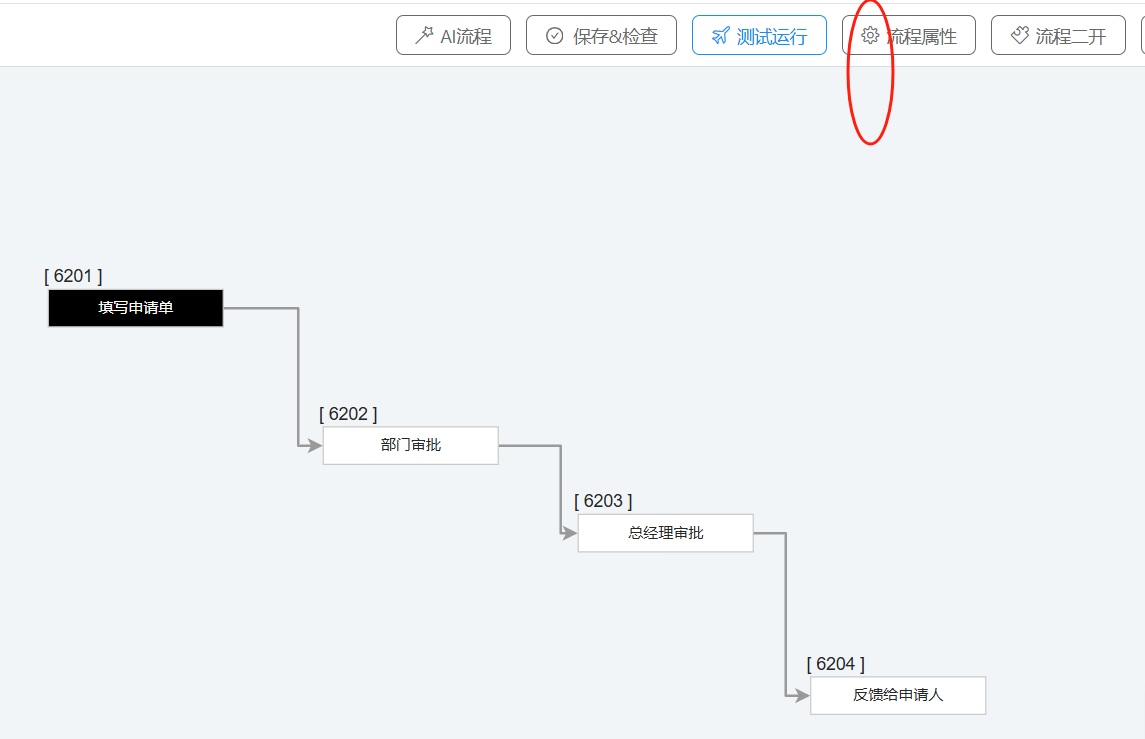


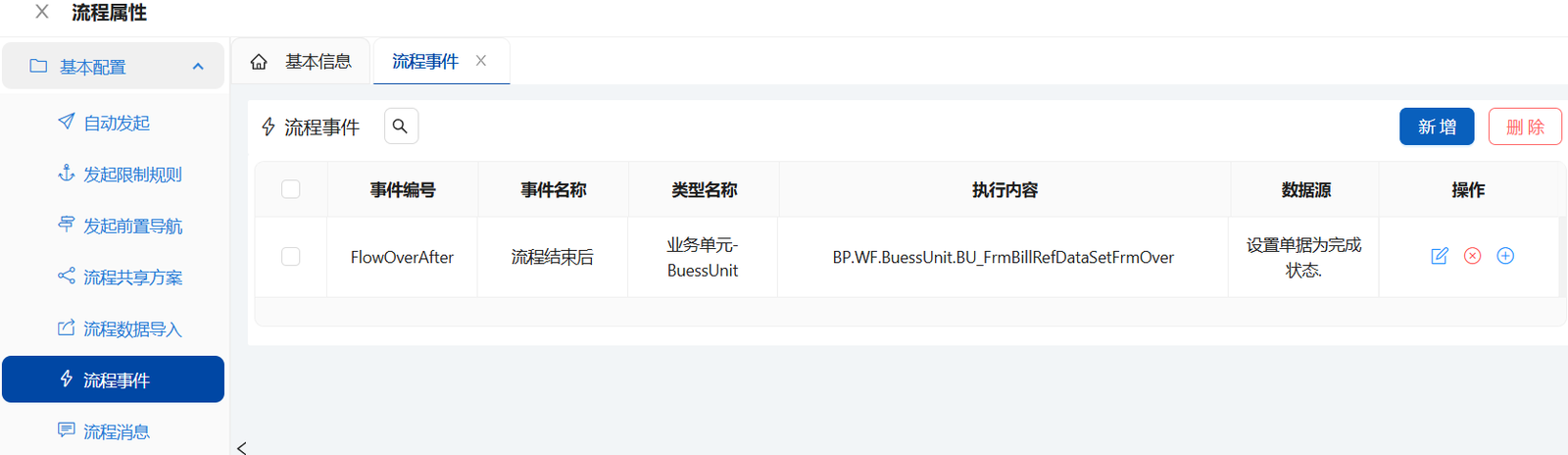
* 选择数据源



### 流程事件

* 进入流程设计器，点击工具栏上的流程属性按钮，进入流程事件设置。

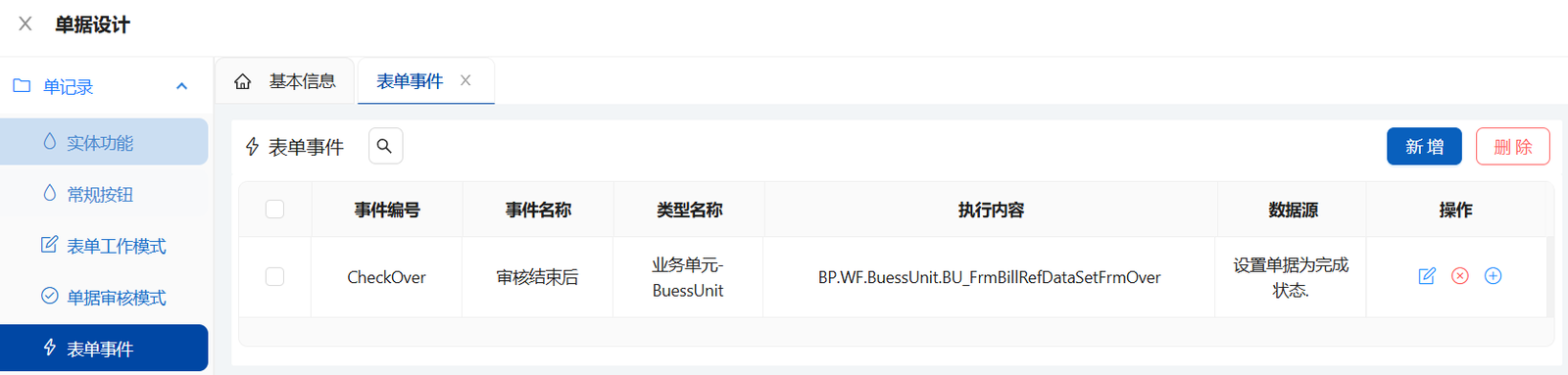


* 设置流程属性，新增事件。
* 
* 新建事件：
* 

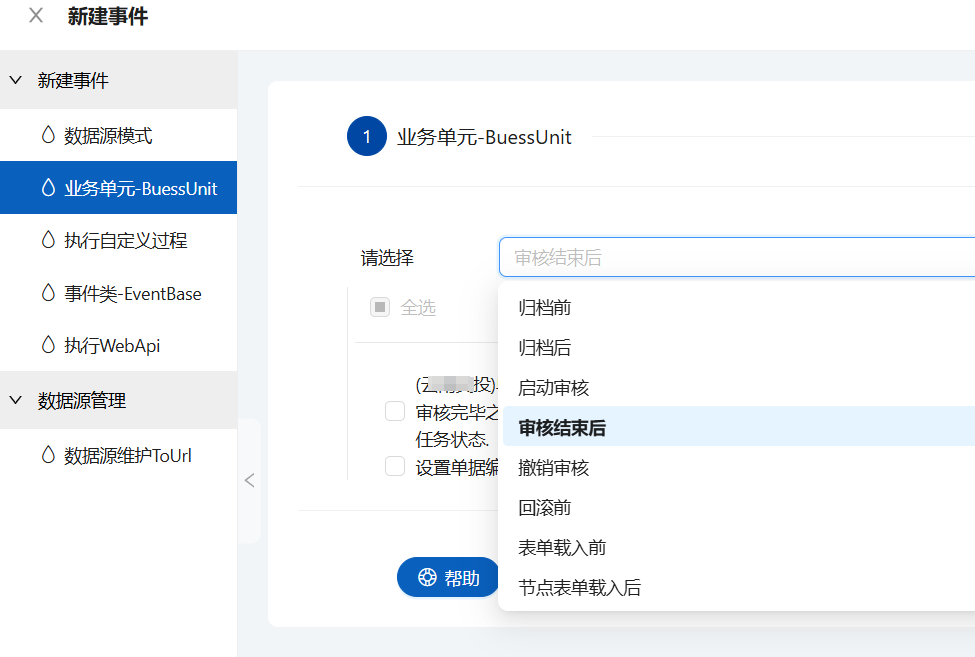
## 单据

操作路径：选择单据





**新建事件**



## 实体

* 同上。

# 前端外挂

## 概述

* 外挂是指在流程、表单运行的时候加载的一段代码，在不同的事件里触发它。
* 系统把外挂分为流程外挂、TS实体外挂2个类型。
* 外挂的业务内容载体是一个：Typescript的类，从基类上继承下来，重写相关的方法。
* 基类位置图
* 继承图
* 因为4大应用场景都与表单有关，所以都要从表单的基类继承下来。
* 每个应用的基类里都定义了自己的方法，以方便子类重写。

## 流程外挂WGFlow

* 解决流程应用的编写外挂的实体，作用在MyFlow.vue工作处理器。
* 优先级高于前端的全局外挂，意思是如果实现一个流程的外挂，全局的前端外挂就不执行了。

### 测试实例

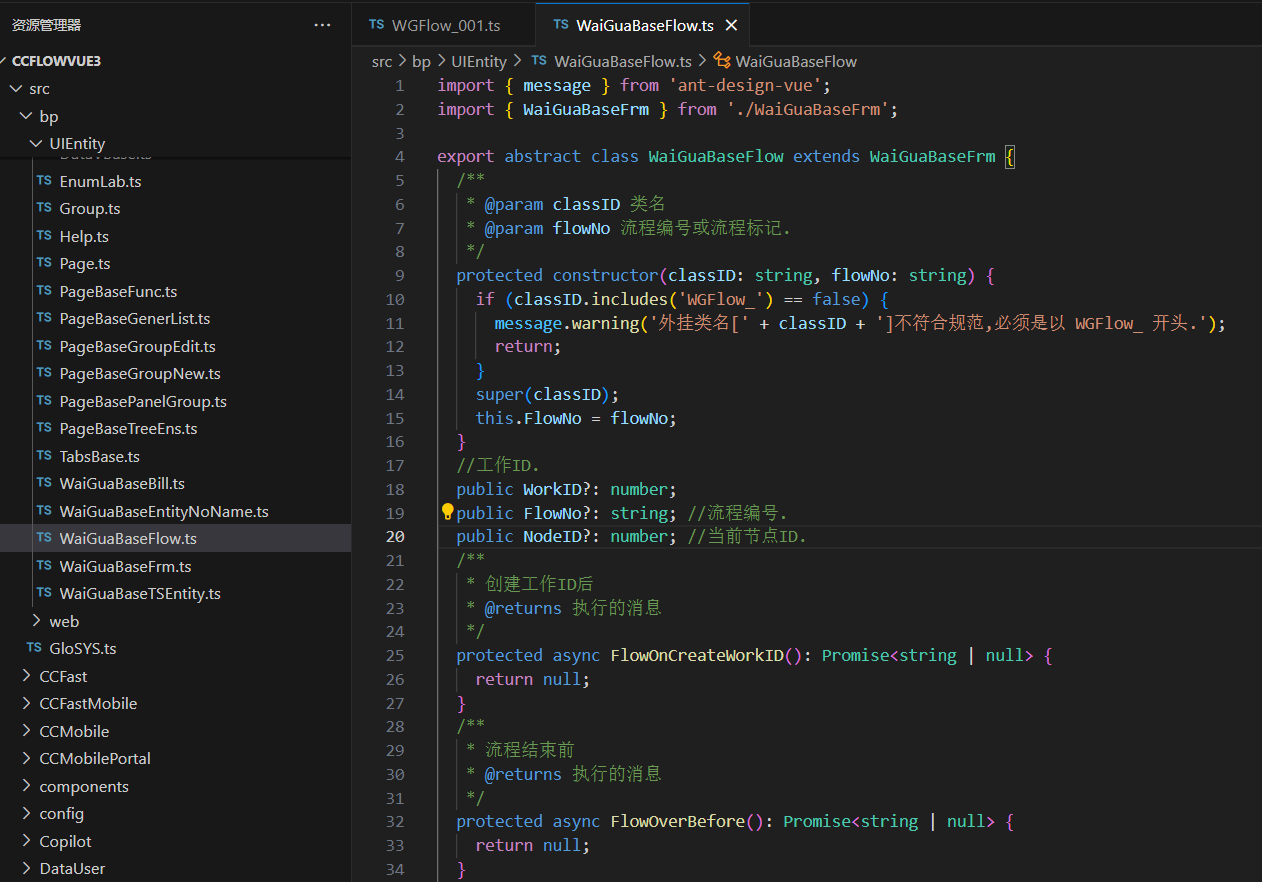
* 对应流程：



* 发送效果：
* 对应代码

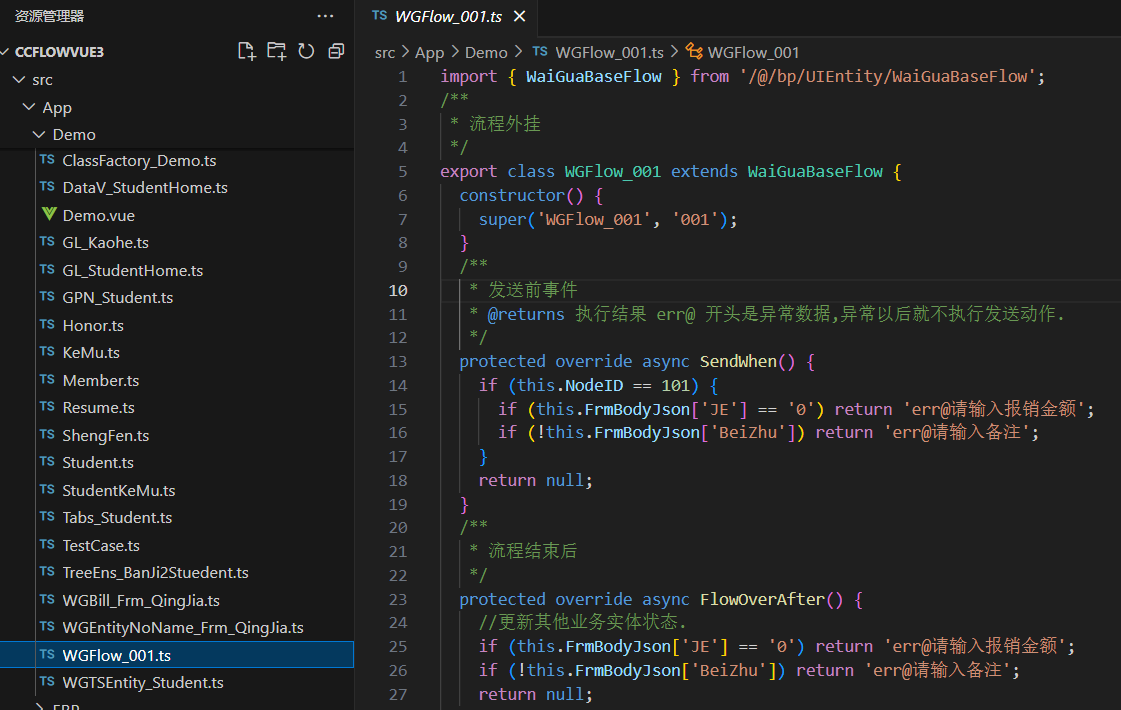
### 系统设计-基类的方法

* 位于\src\bp\UIEntity\WaiGuaBaseFlow.ts



### 重写事例

* 位于\src\App\Demo\WGFlow\_064.ts
* 提供的方法有：发送前、发送失败时、发送成功时、流程结束前、结束后、退回前、退回后、撤销前、撤消后等。

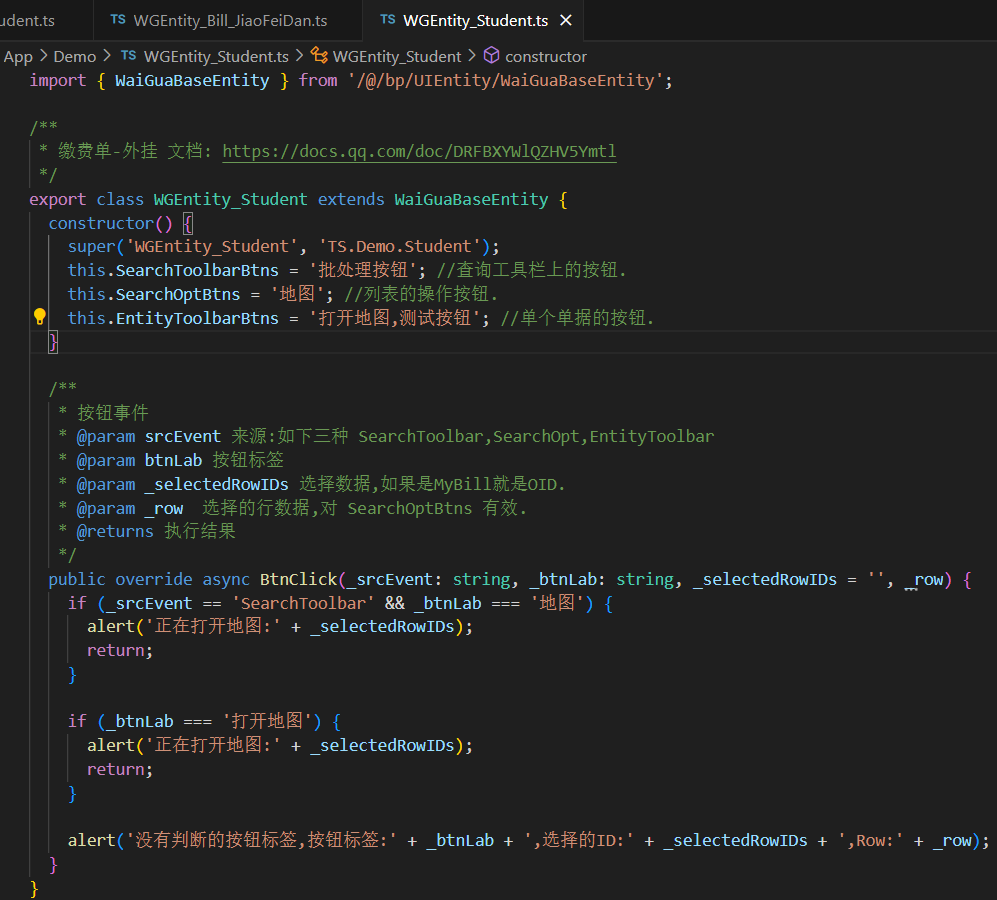


## 实体TSEntity外挂

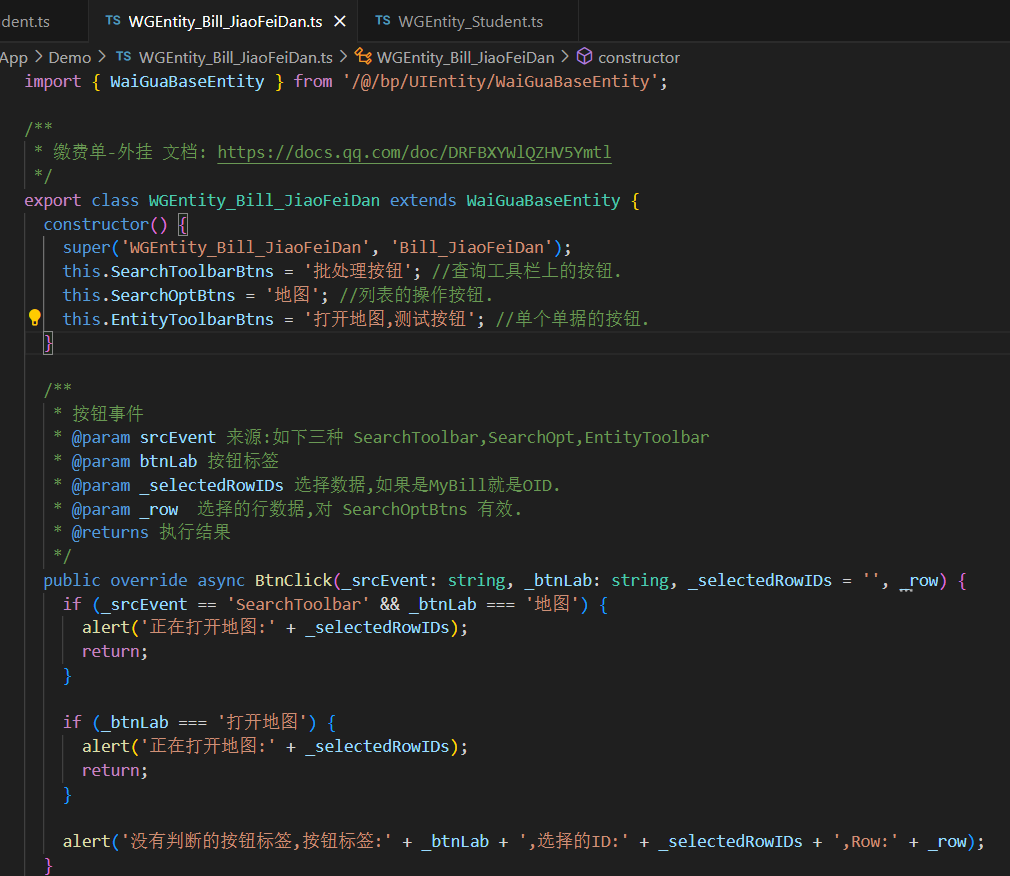
### 说明

* 有效范围：高代码实体，第代码单据，低代码实体

### 高代码Demo实例



### 低代码Demo实例



# 后端外挂

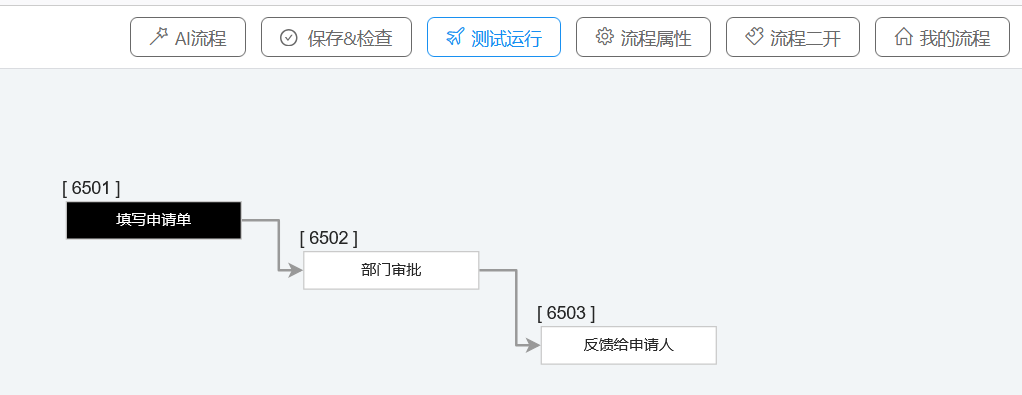
* 使用java或者c#代码的方式

## 流程外挂

* 详细说明：
* **设计实现：**同前端外挂一样，系统定义一个基类描述了方法，子类从基类上继承下来实现方法的重写。
* 对应流程：



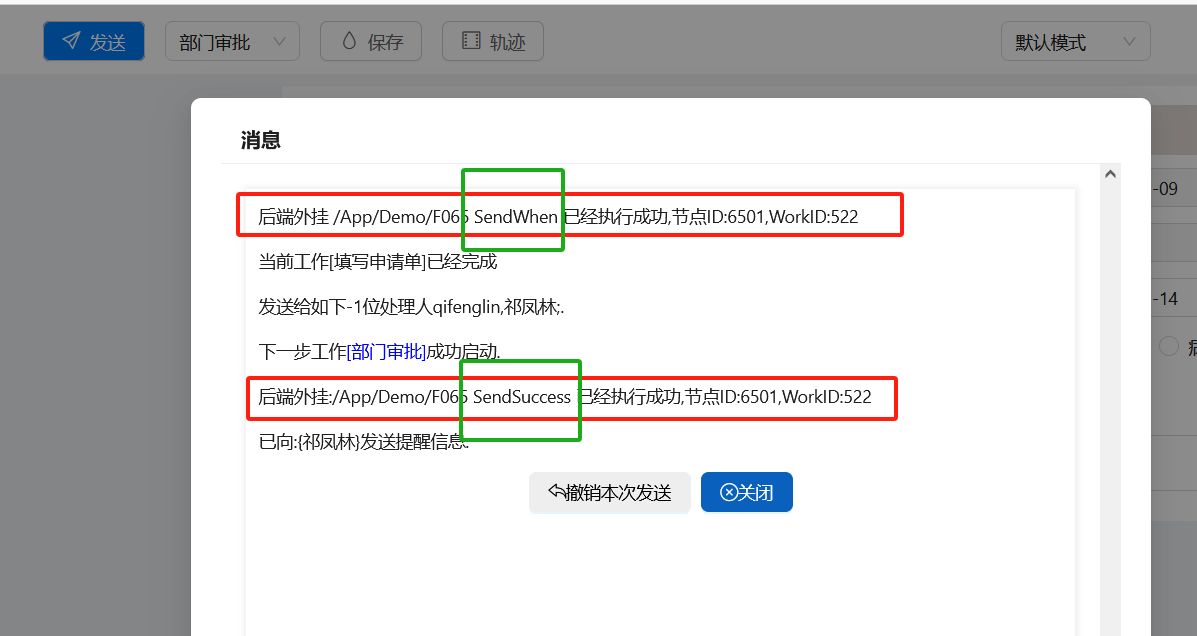
* 流程设计：



* 流程属性加载：



* 执行效果：-发送



* 执行效果：-退回



### 流程java外挂

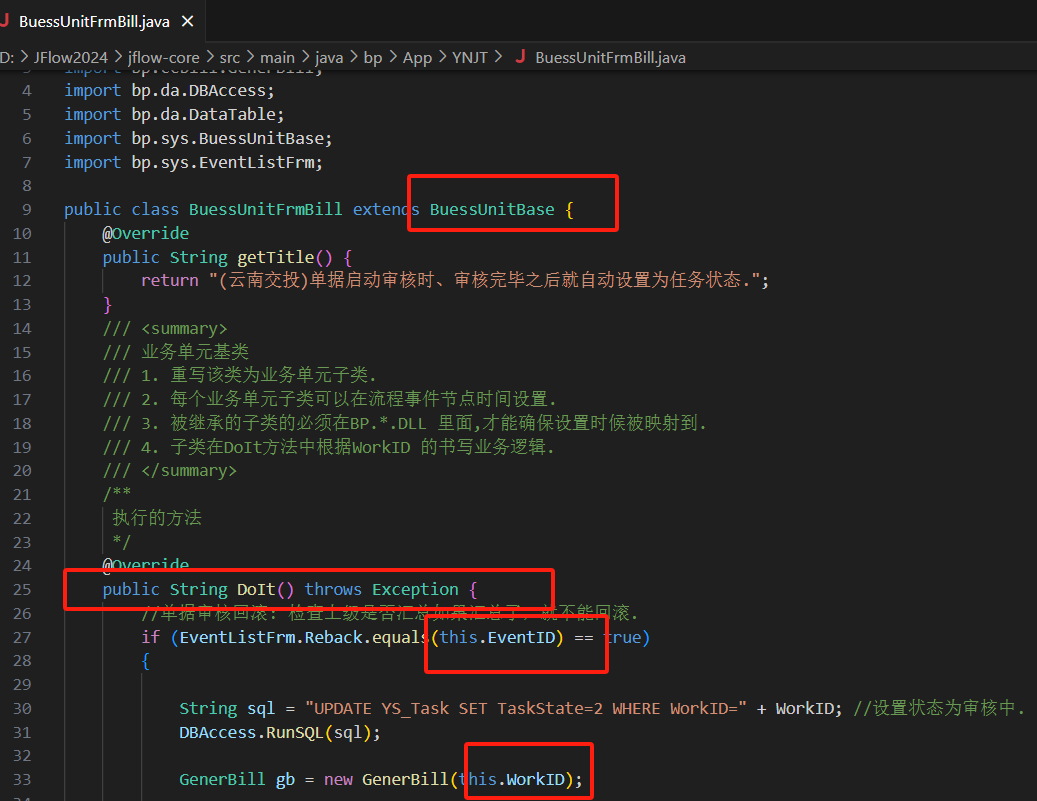
同下

### 流程c#外挂

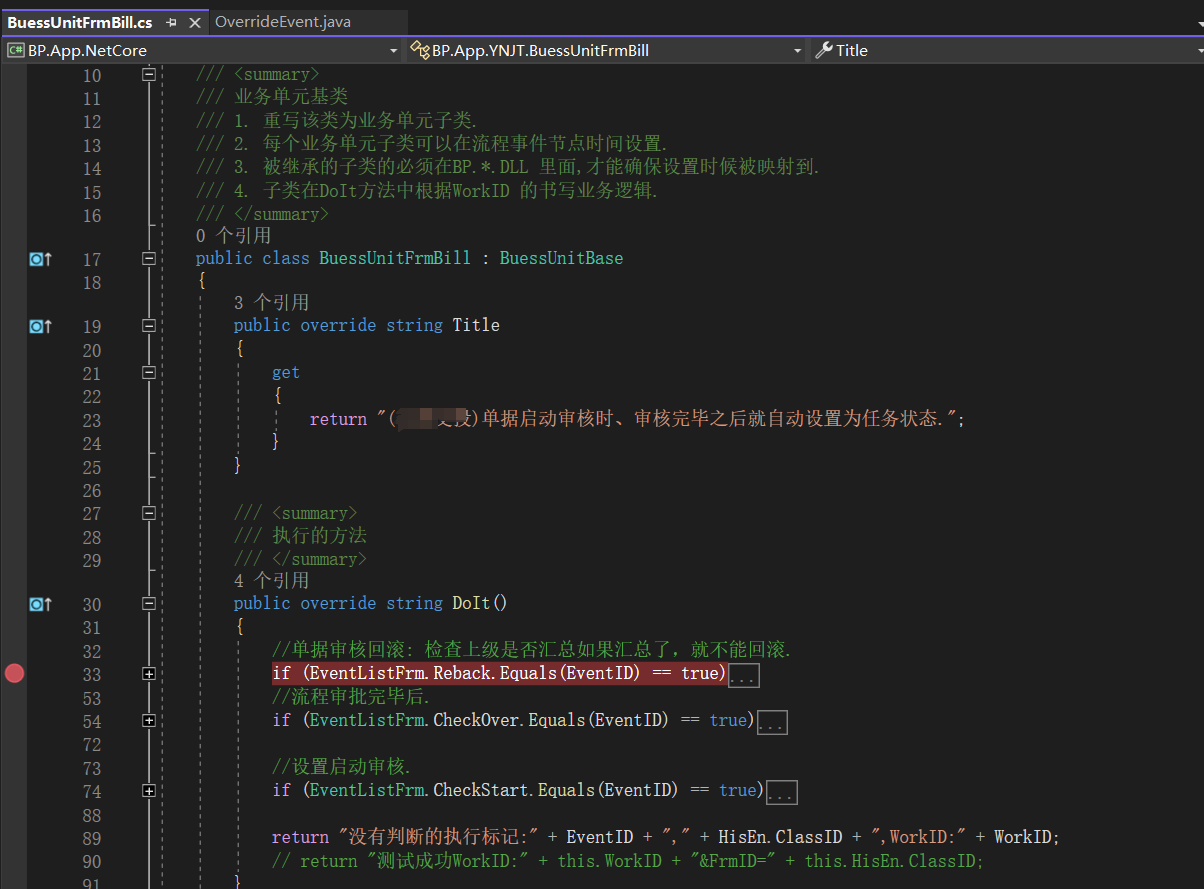
## 业务单元

* 业务单元是执行一段脚本、一个过程实现功能逻辑。
* ccbpm定义了一个基类，子类实现基类的Do方法，在该方法里实现业务。
* 业务单元可以配置到不同的应用，不同的事件里。
* 比如流程应用：可以做一个付款的业务单元，在不同的流程里节点发送成功的事件里调用付款业务逻辑。
* 比如单据应用：单据表单审核成功后，设置单据的状态，审核中、未审核、归档状态。

### 业务单元Java



### 业务单元C#



### 使用配置

* 在节点事件、流程事件、单据事件都有使用业务单元的选项，如下图。



### 系统提供的业务单元

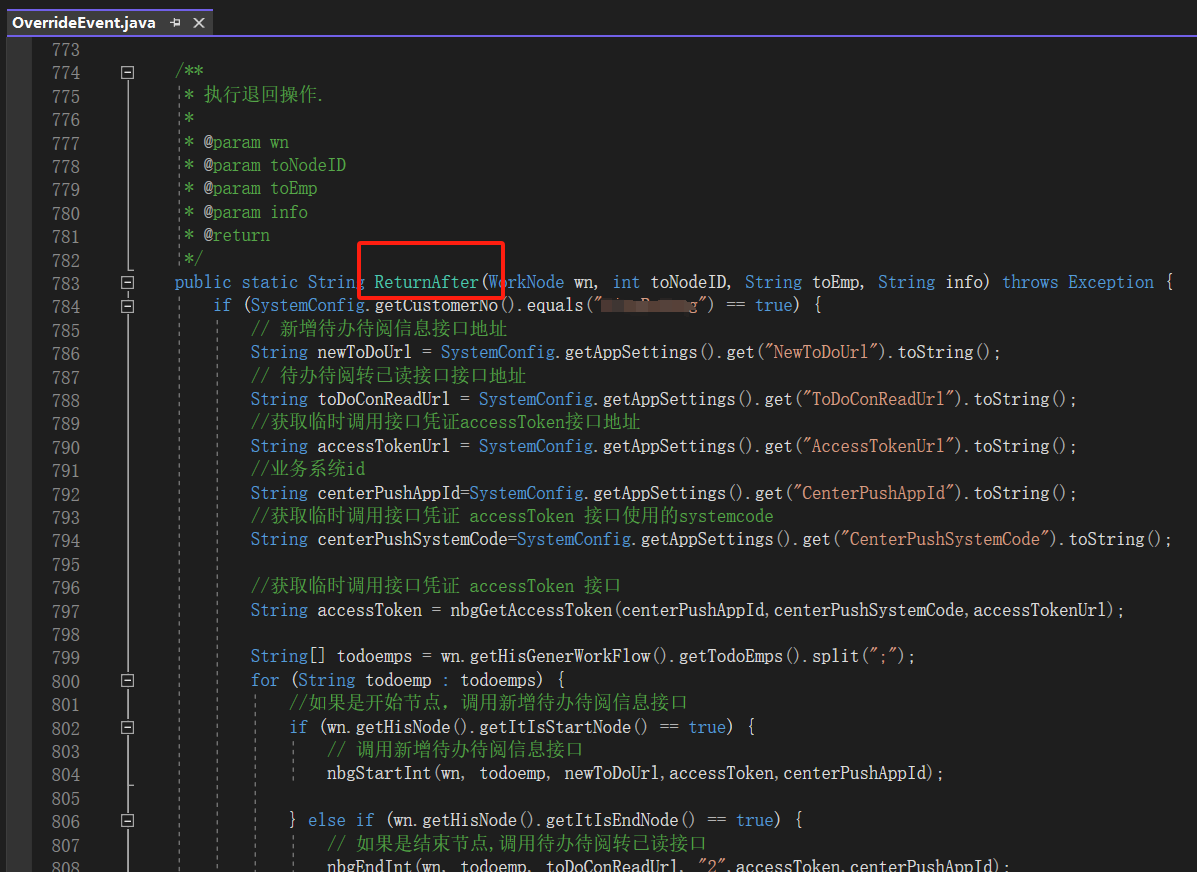
|  |  |  |
| --- | --- | --- |
| # | 单据 | 概述 |
| 1 |  | 设置单据审核状态 |
| 2 |  | 设置单据归档状态 |
| 3 |  | 设置单据编辑状态 |

## 流程全局事件

* 全局的流程都走的事件，比如：发送、退回、撤销、流程结束。
* 开发人员可以根据当前的流程编号、节点ID个性化的实现业务逻辑。

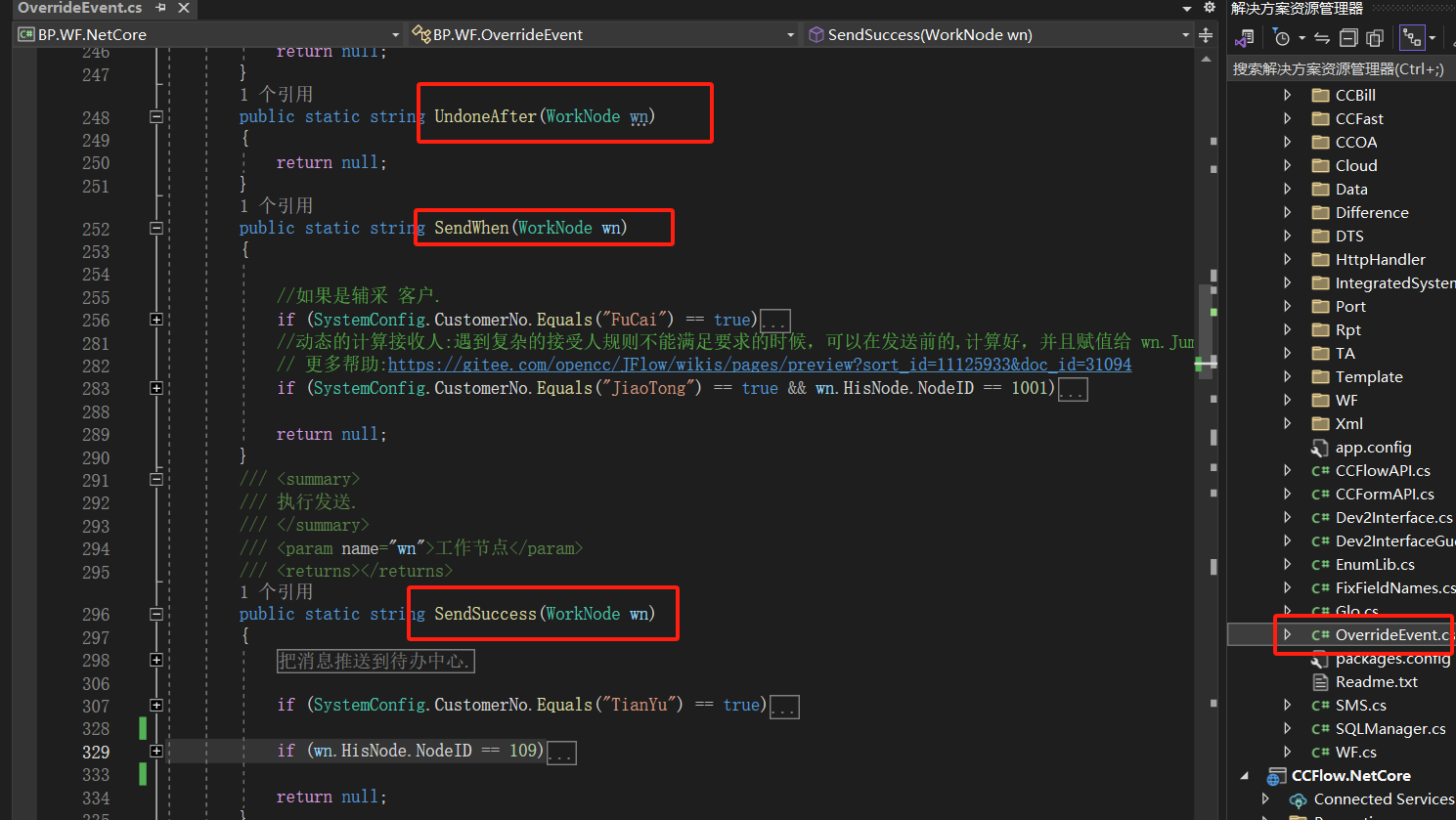
### 全局事件java

* 位置: src\main\java\bp\wf\OverrideEvent.java



### 全局事件C#

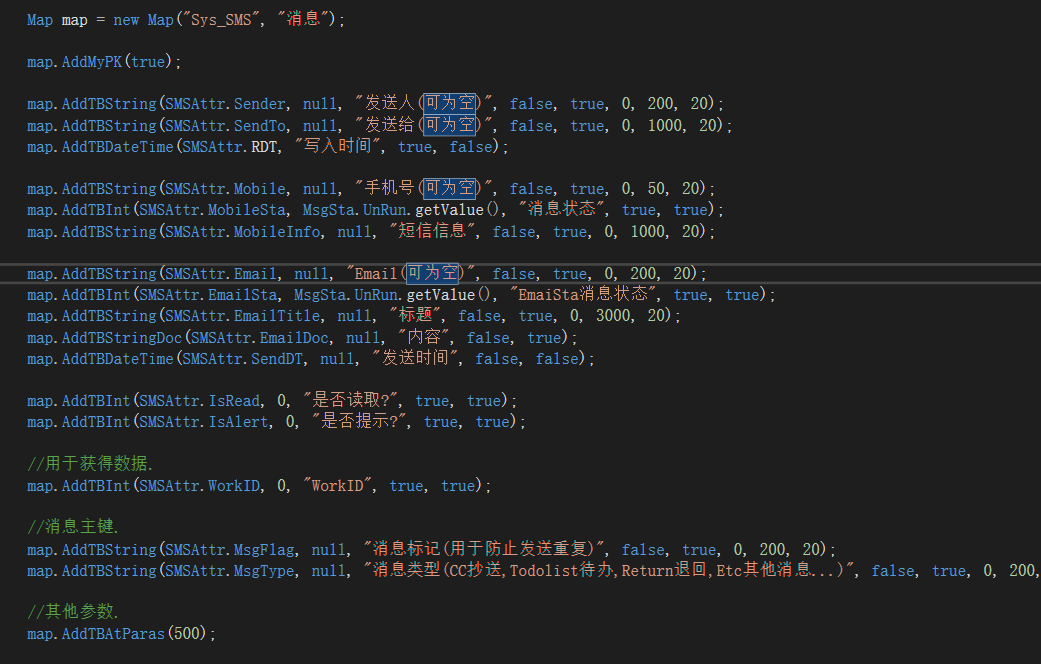
* 位置：\Components\BP.WF\OverrideEvent.cs
* 所有的发送、退回、转发、移交、撤销都走对应的重写方法。
* 开发人员可以根据当前的流程编号、节点ID个性化的实现业务逻辑。



# 消息接管开发

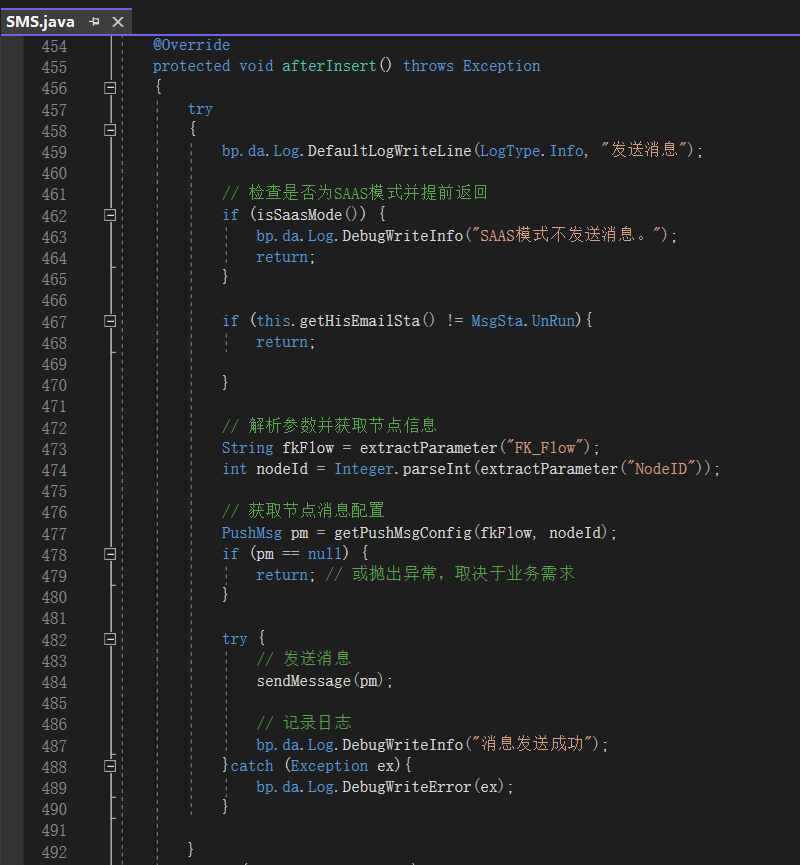
* **消息定义：**在与系统交互过程中有事件产生的信息，称为消息。
* 比如：在节点发送成功事件，需要把当事人（下一步的接收人）作为通知对象。退回后事件，把退回到的人作为消息通知的对象。
* **消息接管：**所有的消息都是通过一个方法插入到数据库里，只要在插入之前接管这些数据，把这些数据写入您所想要的系统就叫消息接管。
* 比如：把接单发送成功的消息写入手机短信、钉钉、企业微信、企业内部的即时通讯里。

## 消息内容



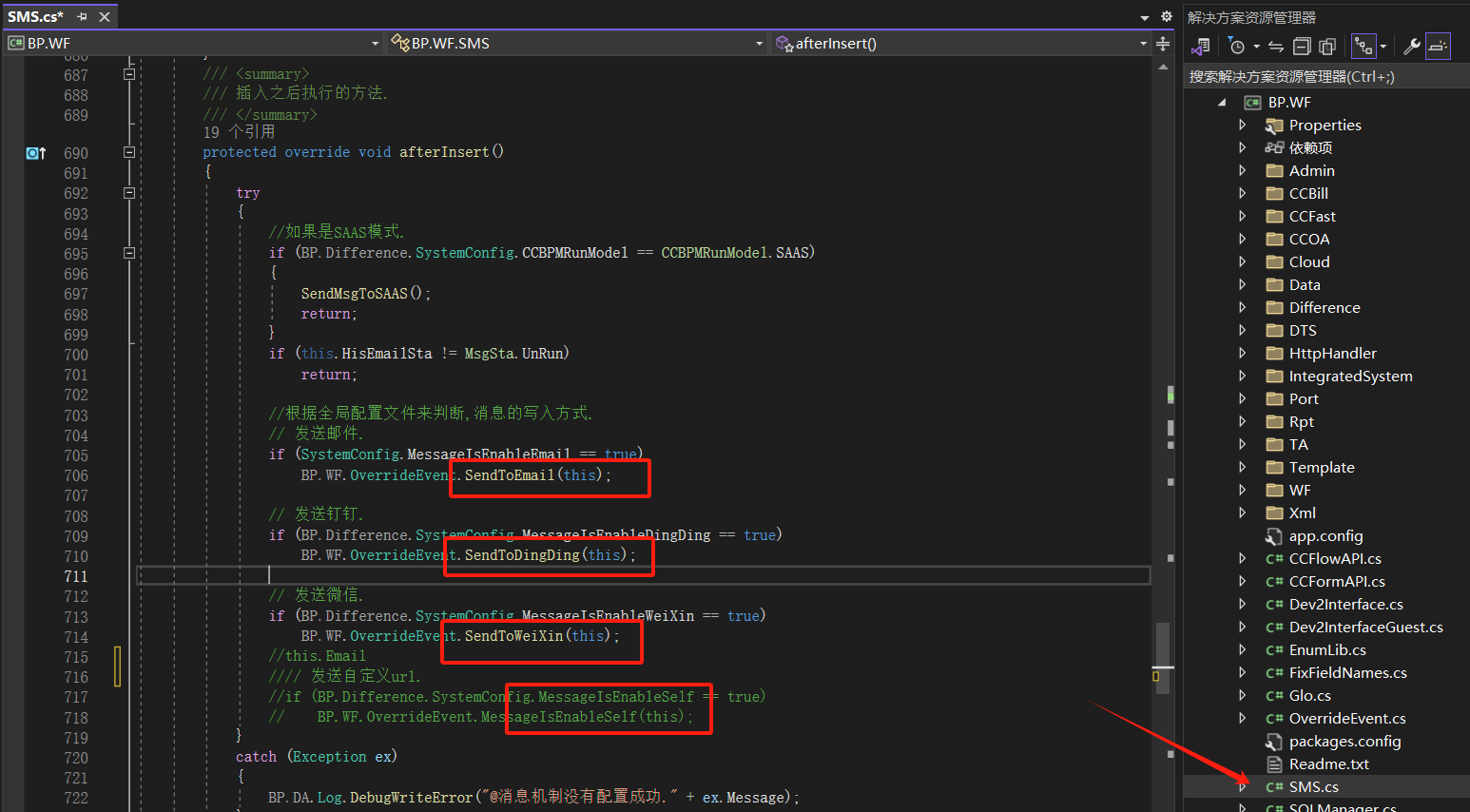
## For JFlow

位置：src\main\java\bp\wf\SMS.java



## For ccflow

* 位置：\Components\BP.WF\SMS.cs



# 应用API操作(前端)

* 应用API，包括流程API，与单据的API操作。
* 使用TypeScript的静态方法，封装了对流程、单据的操作。

## 流程API

* 使用一组静态的方法实现对流程的操作，更多参考：/App/Demo/TestCase.ts文件。

## 单据API

* 同上

# 应用API操作(后端)

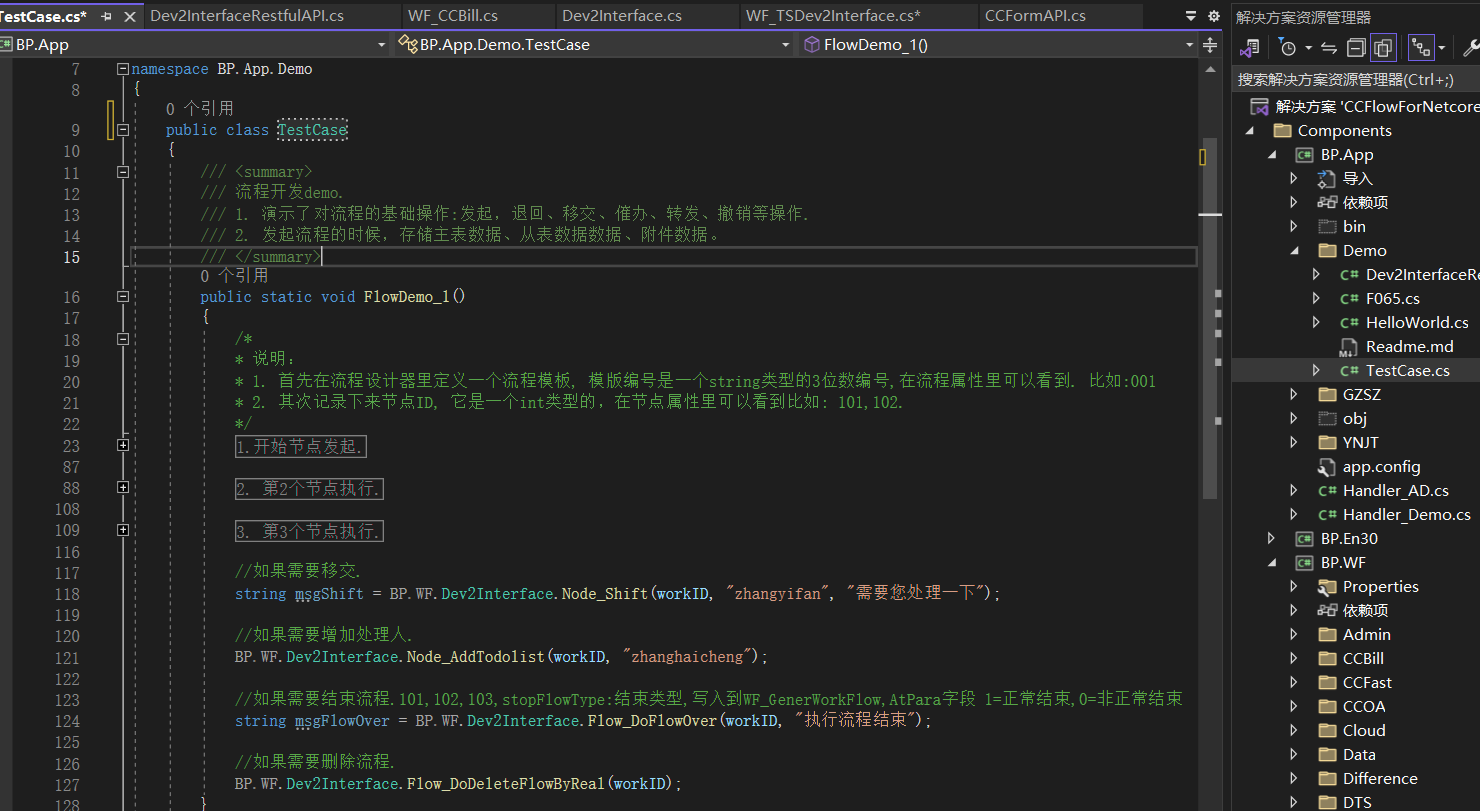
* API 是指ccbpm封装出来的访问应用的方法。
* 应用API分为流程与单据两种模式。

## 流程API.

* 流程API是对流程操作的方法接口，比如：流程实例的创建、发起、移交等。
* 更多详情参考：
* <https://gitee.com/opencc/JFlow/wikis/pages/preview?sort_id=8101800&doc_id=31094>
* **原装：**支持没有在java,.netcore内置的环境下可以执行的接口。
* **封装：**是指对接口进行resfull的封装的api。

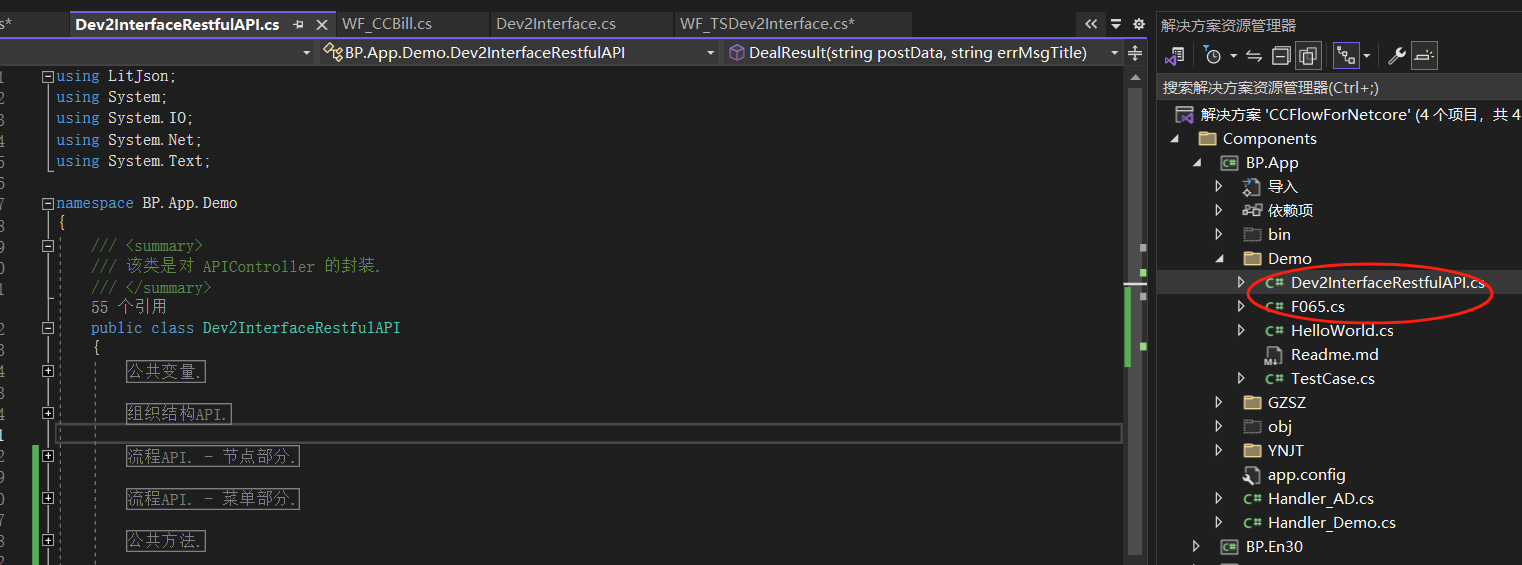
### 原装

* 代码实例：java位于./src/java/bp/app/Demo/TestCase.java
* .netcore位于:/Commpent/BP.App/Demo/TestCase.cs
* 如下图



### 封装

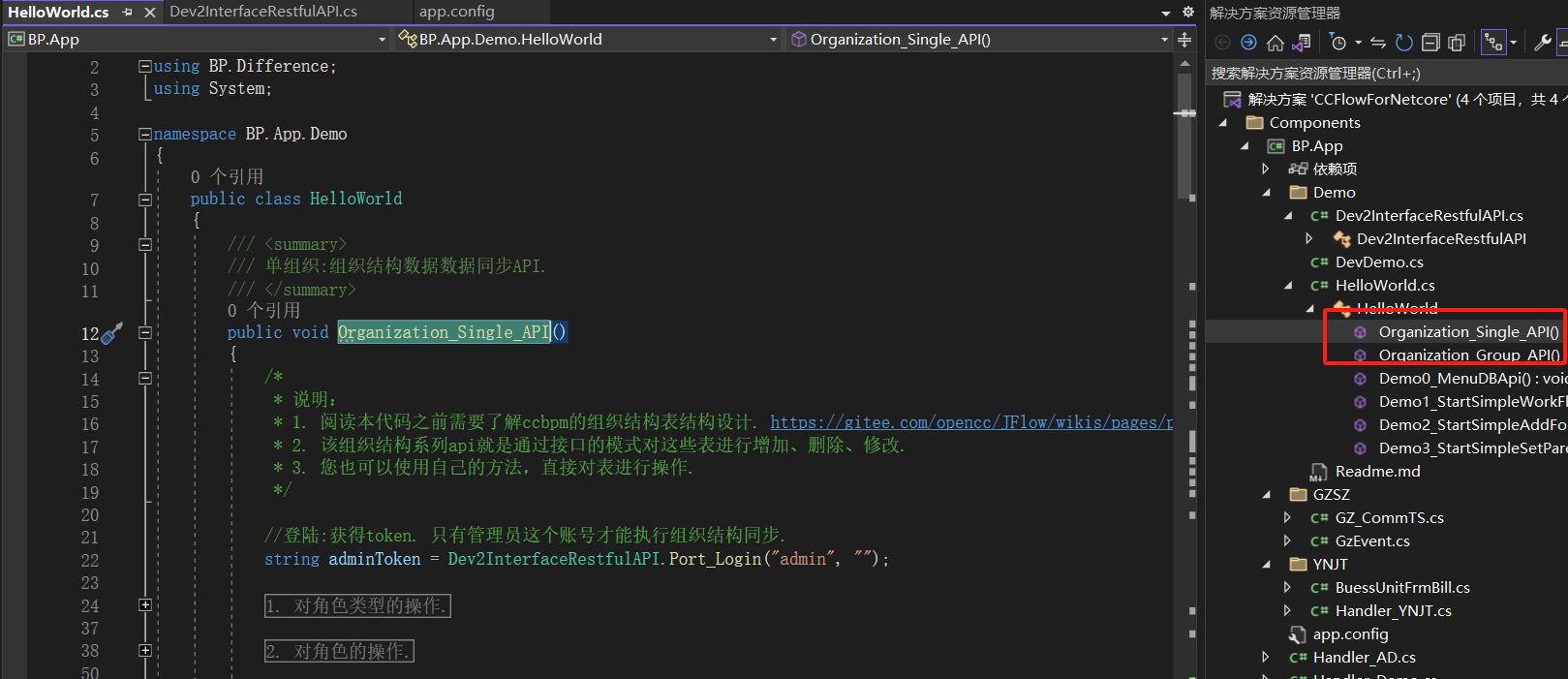
* 代码实例:
* java 位于./src/java/bp/app/Demo/TestCase.java
* .netcore位于:/CommpentBP.App/Demo/TestCase.cs



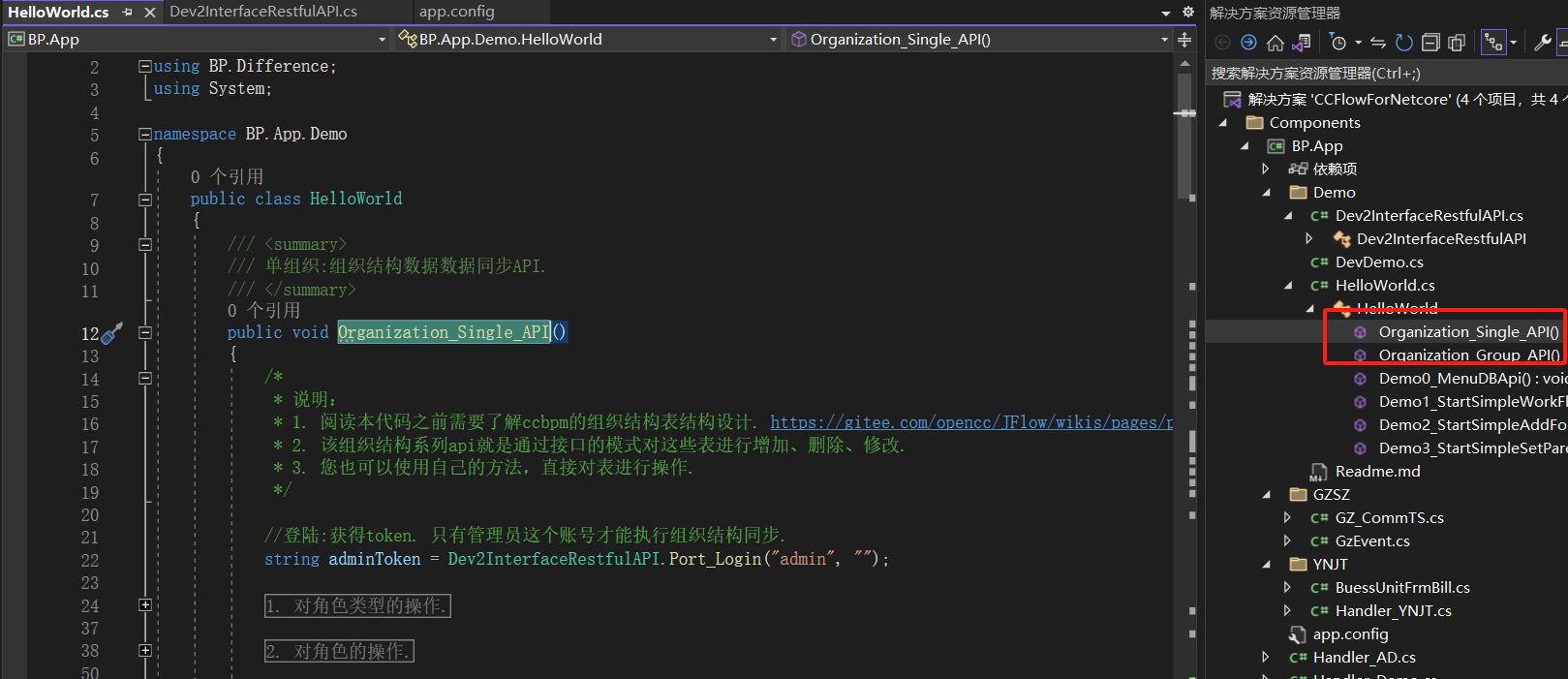
## 组织结构同步API

* 相关的组织结构集成，请参考：
* <https://gitee.com/opencc/JFlow/wikis/pages/preview?sort_id=8103125&doc_id=31094>

### For Java



### For C#



## 单据API.

### 原装

### 封装

# TypeScript在线编辑(开发中)

说明: