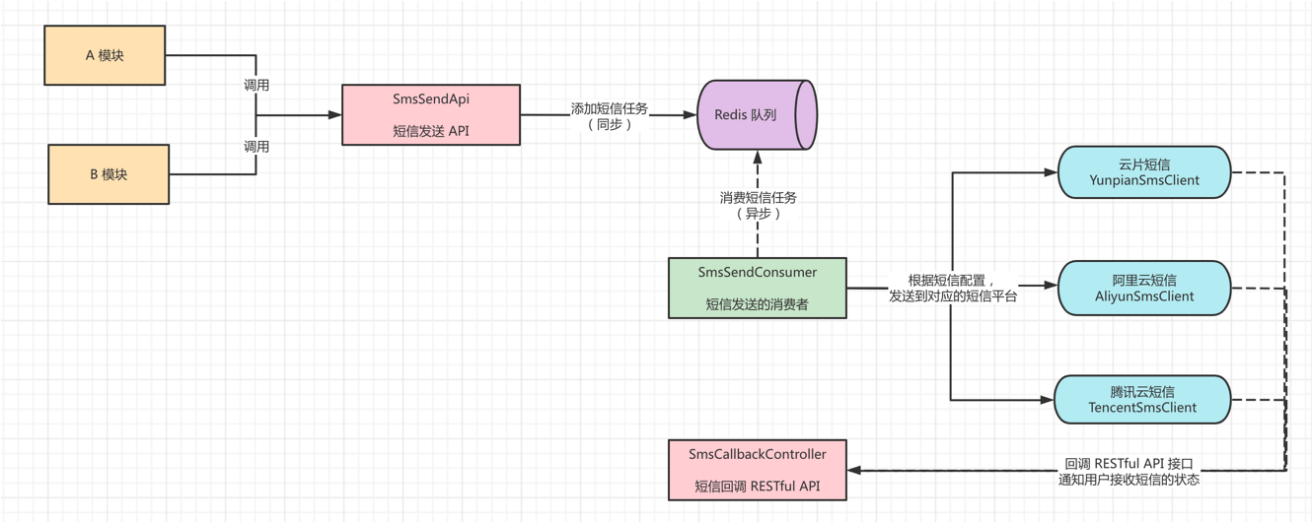


🎯 短信配置

本章节，介绍项目的短信功能。该功能提供统一的短信 API 给其它模块，使它们可以快速接入短信功能，无需关心不同短信平台的具体对接。

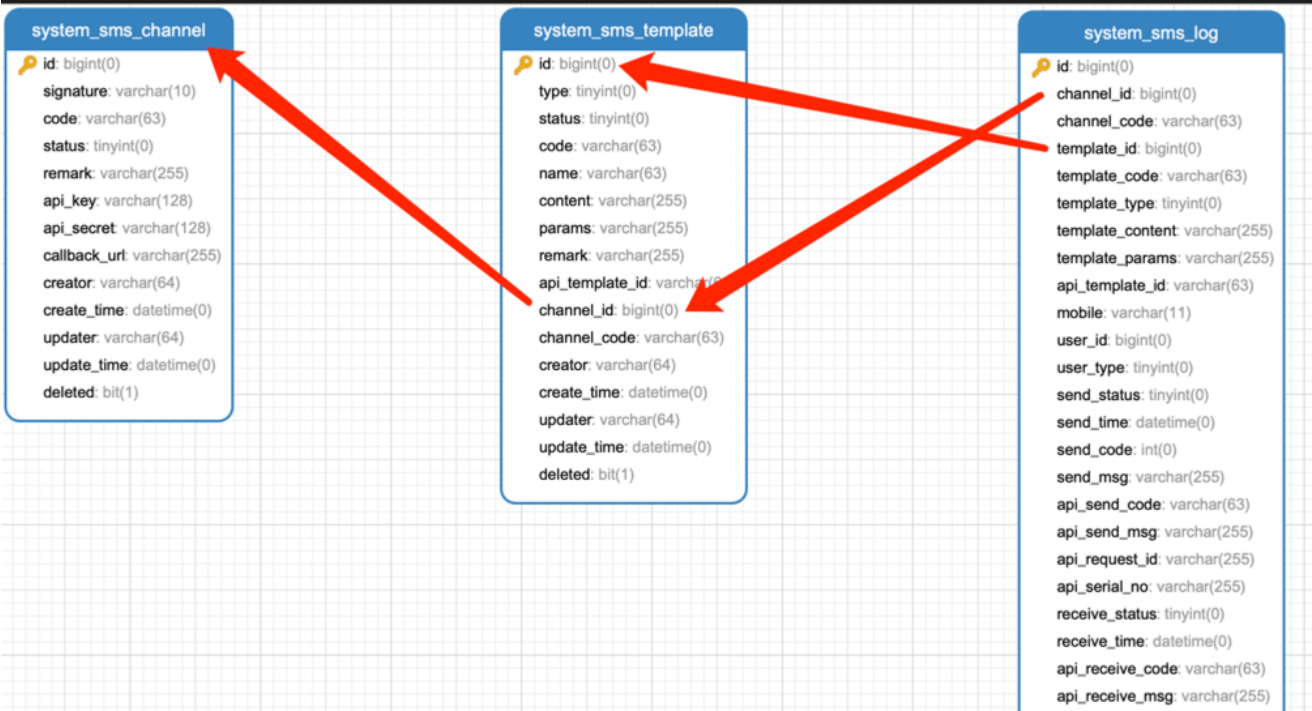
短信采用异步发送，基于 **RocketMQ 消息队列**，如下图所示：



友情提示：图中的【Redis 消息队列】，应该是【RocketMQ 消息队列】哈~

- [yudao-spring-boot-starter-biz-sms](#) 业务组件：封装不同短信平台的客户端。
- [yudao-module-system](#) 的 [sms](#) 业务模块，提供短信渠道、模板的配置，短信日志的查看，短信的发送等功能。

1. 表结构



2. 短信配置

本小节，讲解如何配置短信功能，整个过程如下：

- 新建一个短信【渠道】，配置对应短信平台的账号
- 新建一个短信【模版】，配置对应短信平台的模板
- 测试该短信模板，查看对应的短信【日志】，确认是否发送成功

2.1 新建短信渠道

① 点击 [系统管理 -> 短信管理 -> 短信渠道] 菜单，查看短信渠道的列表。如下图所示：

李道管理系统

首页 / 系统管理 / 短信管理 / 短信渠道

短信渠道

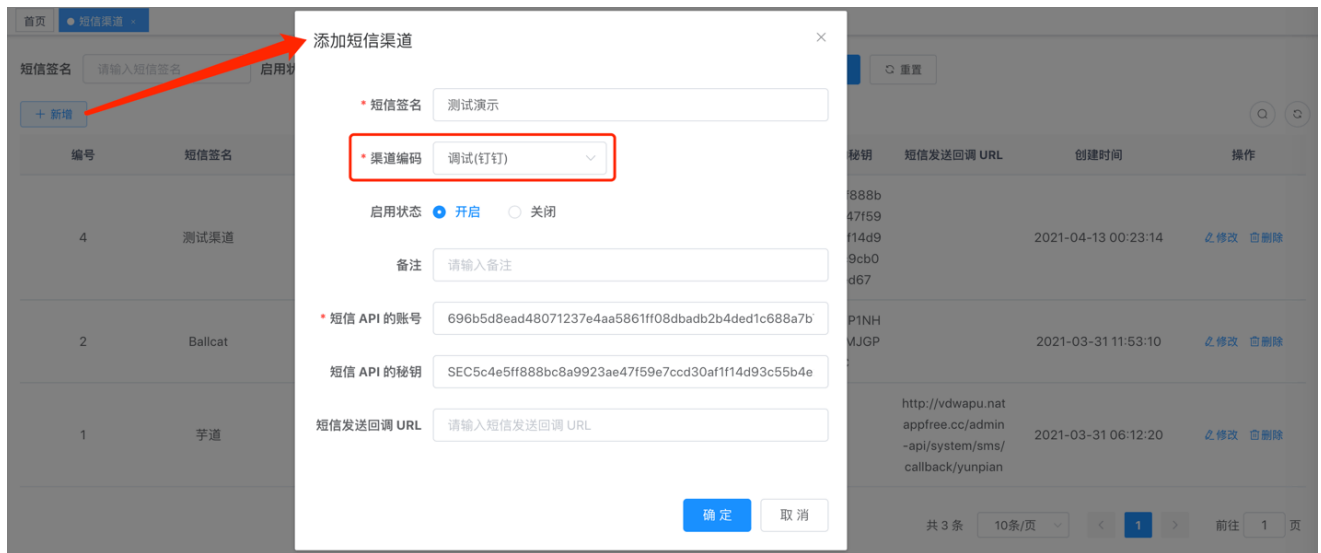
短信签名: 请输入短信签名 启用状态: 请选择启用状态 创建时间: 开始日期 - 结束日期 搜索 重置

+ 新增

编号	短信签名	渠道编码	启用状态	备注	短信 API 的账号	短信 API 的密钥	短信发送回调 URL	创建时间	操作
4	测试渠道	调试(钉钉)	开启	123	696b5d8ead48071237e4aa5861ff08dbadb2b4ded1c688a7b7c9afc615579859	SEC5c4e5ff888bc8a9923ae47f59e7ccd30af1f14d93c55b4e2c9cb094e35aeed67		2021-04-13 00:23:14	修改 删除
2	Ballcat	阿里云	开启	啦啦啦	LTAI5tCnKso2uG3kJ5gRav88	fGJ55NXL7P1NHNRmJ7DJaMJGP yE55C		2021-03-31 11:53:10	修改 删除
1	李道	云片	开启	呵呵呵	1555a14277cb8a608c45a9e6a80d510		http://vdiwapu.natappfree.cc/admin-api/system/sms/callback/yunpian	2021-03-31 06:12:20	修改 删除

共 3 条 10条/页 1 前往 1 页

② 点击 [新增] 按钮，选择渠道编码为【调试（钉钉）】，并填写信息如下图：



短信 API 的账号：696b5d8ead48071237e4aa5861ff08dbadb2b4ded1c688a7b7c9afc615579859

短信 API 的密钥：SEC5c4e5ff888bc8a9923ae47f59e7ccd30af1f14d93c55b4e2c9cb094e35aee

疑问 1：为什么选择渠道编码为【调试（钉钉）】？

该类型使用钉钉机器人来模拟短信发送，用于日常调试。

- 短信 API 的账号，对应机器人的 Webhook 的 `access_token` 参数
- 短信 API 的密钥，对应机器人的安全设置的加签

上图使用的配置，是芳芳自己的钉钉机器人。正式使用时，必须参考 [《钉钉开放平台——自定义机器人接入》](#) 文档，申请自己的专属机器人。

疑问 2：可以选择其它渠道编码吗？

当然可以，这里主要考虑部分同学暂时没有申请短信平台，所以使用【调试（钉钉）】渠道编码。

不同短信平台的配置，可见 [「6. 短信平台附录」](#) 小节。

2.2 新建短信模板

① 点击 [系统管理 -> 短信管理 -> 短信模板] 菜单，查看短信模板的列表。如下图所示：



② 点击 [新增] 按钮，选择刚创建的短信渠道，并填写信息如下图：



- 短信渠道编号：发送该短信模板时，使用的短信渠道，即使用哪个短信平台进行发送
- 模板编号：短信模板的唯一标识，使用短信 API 时，通过它标识使用的短信模板
- 模板内容：短信模板的内容，使用 `{var}` 作为占位符，例如说 `{name}`、`{code}` 等
- 短信 API 模板编号：短信平台的短信模板的编号，需要保证该模板在短信平台已经审核通过
- 开启状态：短信模板被禁用时，该短信模板将不发送短信，只记录短信日志

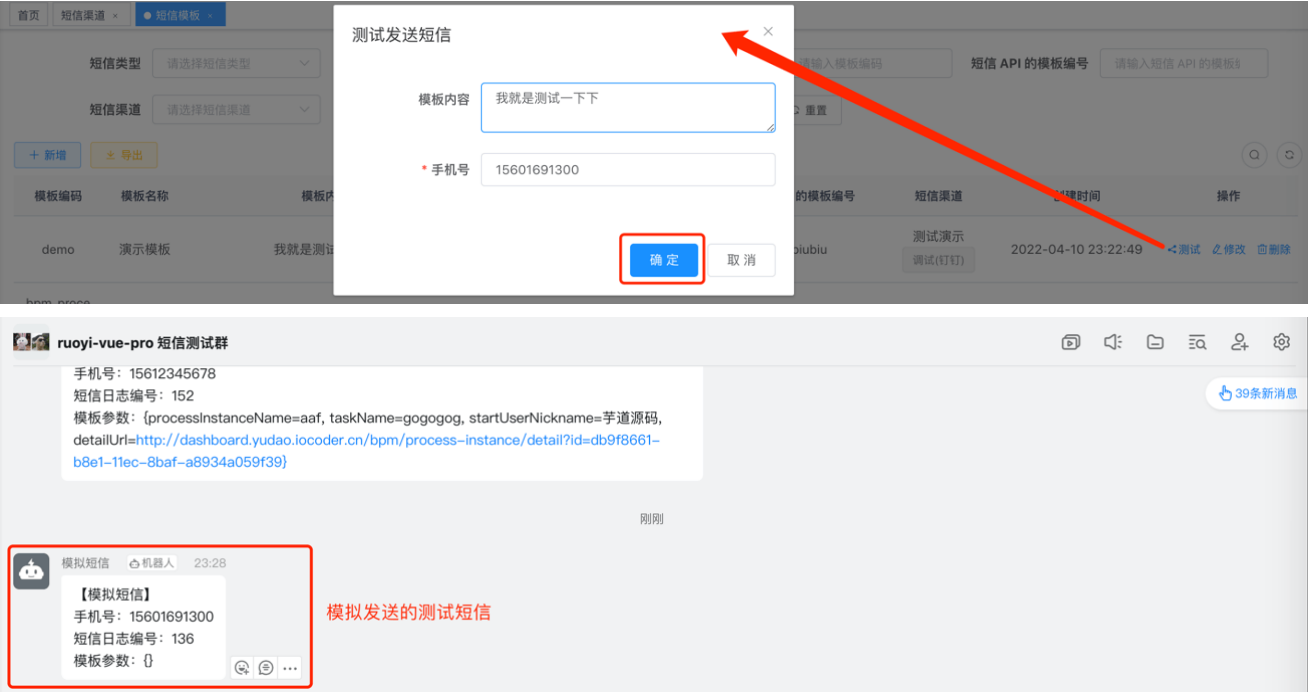
疑问：为什么设计短信模板的功能？

在一些场景下，需要修改短信模板所使用的短信平台。例如说：短信平台出现故障，或者切换短信平台等等。

此时，只需要修改短信模板的两个属性：短信渠道编号、短信 API 模板编号，无需重启应用。

2.3 查看短信日志

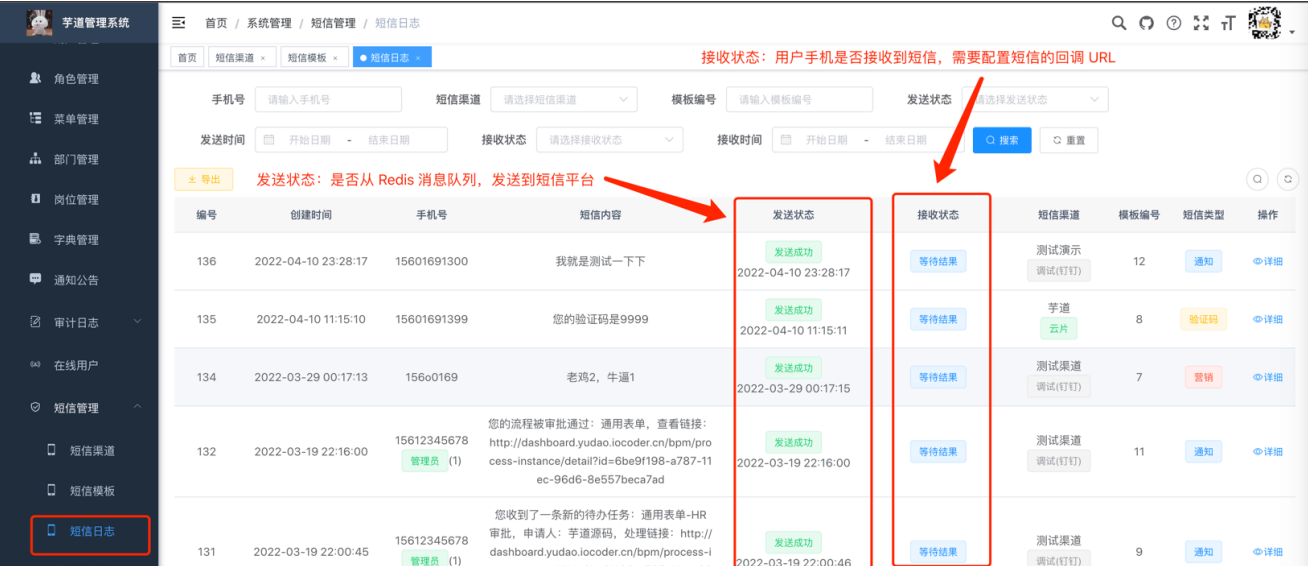
- ① 使用钉钉，扫码 图片 加入机器人所在的【ruoyi-vue-pro 短信测试群】，查看测试短信的模拟发送。
- ② 点击 [测试] 按钮，输入任一手机号，进行该短信模板的模拟发送。如下图所示：



友情提示：如果使用的短信渠道是阿里云、腾讯云等正式的短信平台，则会发送到填写的手机号中。例如说：



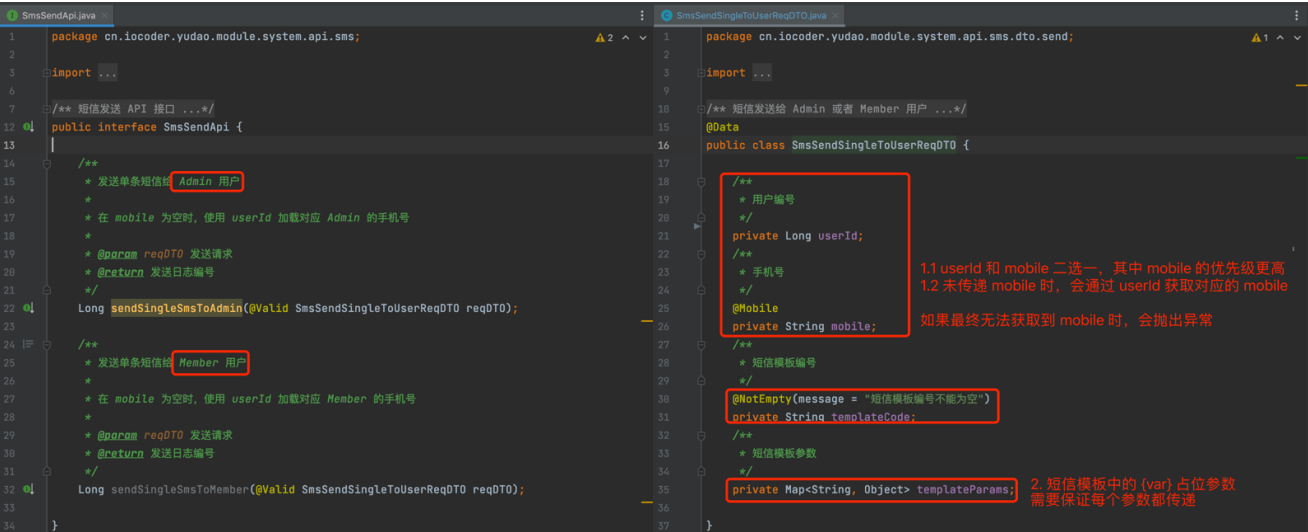
③ 点击 [系统管理 -> 短信管理 -> 短信日志] 采单，可以查看到每条短信的发送状态、接收状态。如下图所示：



3. 短信发送

3.1 SmsSendApi

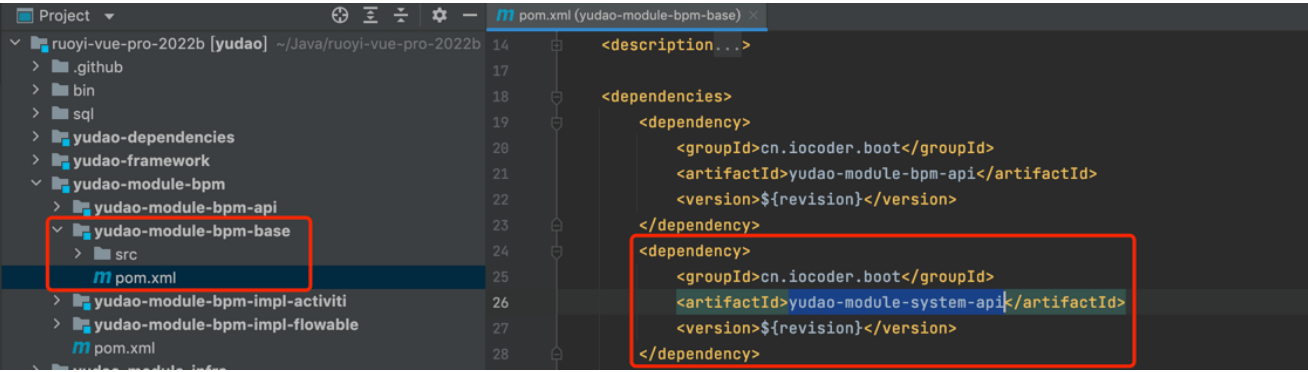
使用 `SmsSendApi` 进行短信的发送，支持多种用户类型。它的方法如下：



3.2 实战案例

以工作流申请通过时，发送短信为例子，讲解 `SmsSendApi` 的使用。

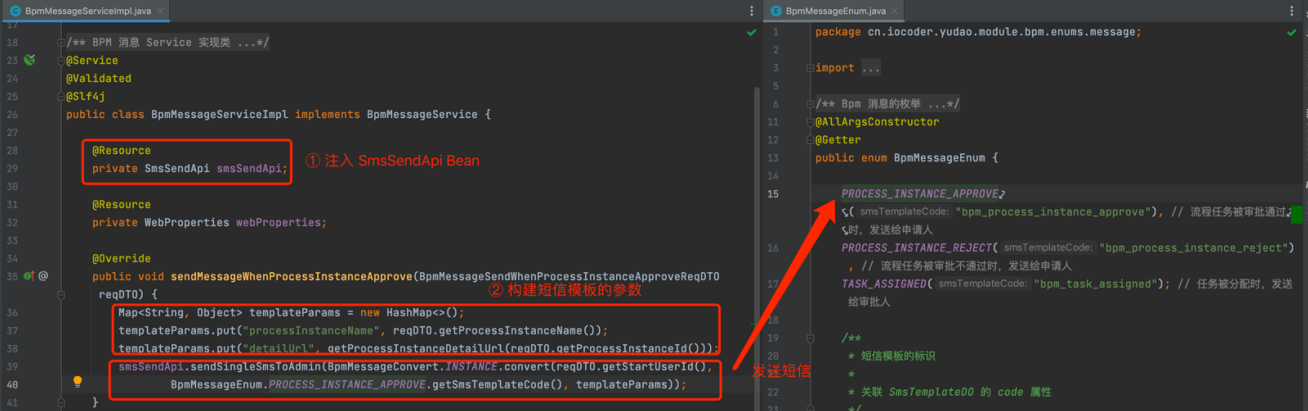
① 引入 `yudao-module-system-api` 依赖，如下图所示：



② 新建对应的短信模板，如下图所示：



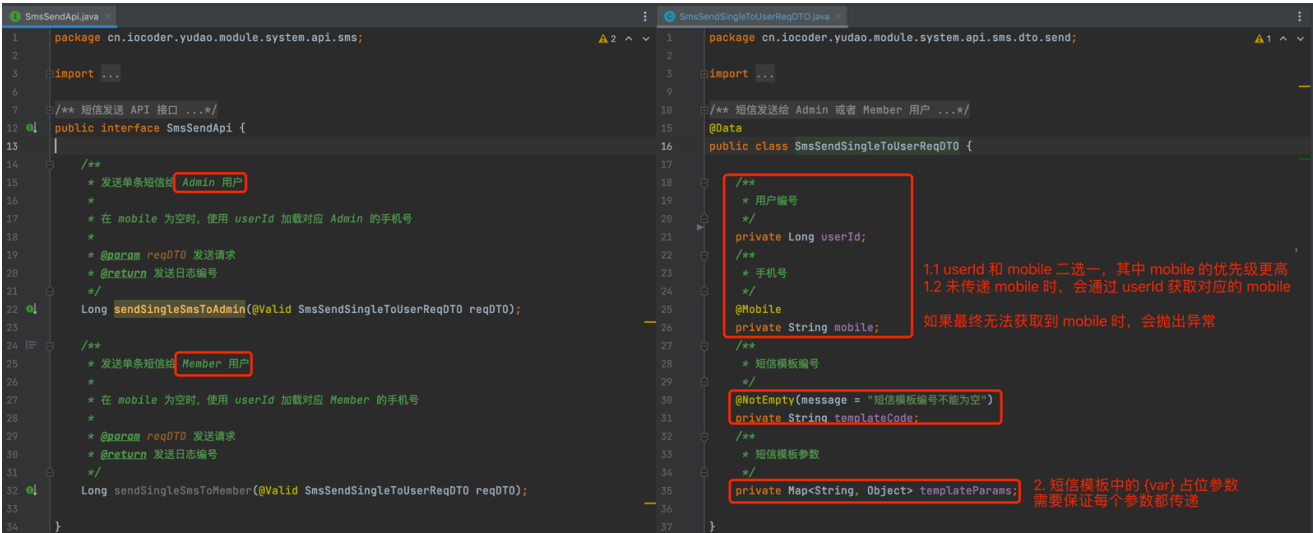
③ 使用 Spring 注入 SmsSendApi Bean，调用对应的短信发送方法。如下图所示：



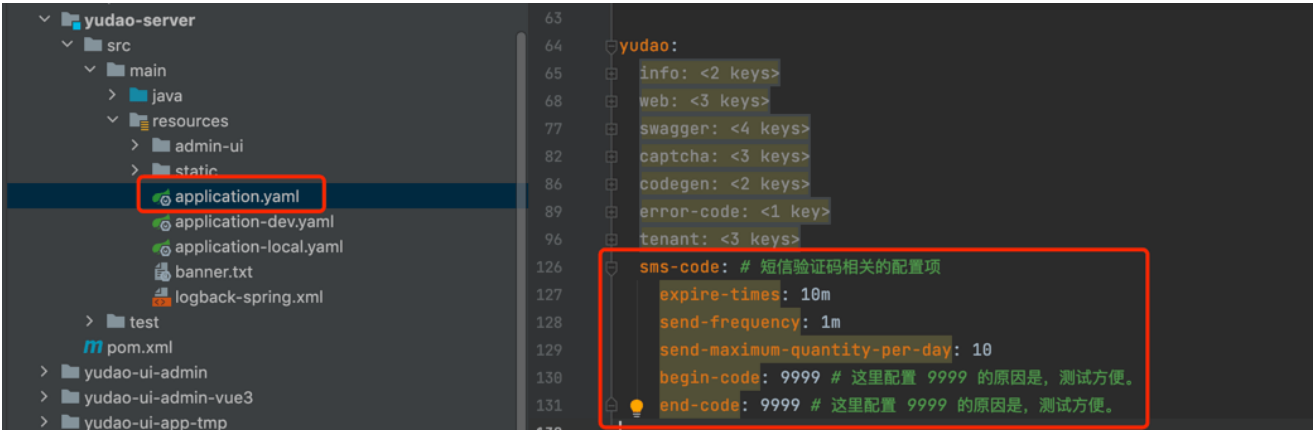
4. 验证码发送

4.1 SmsCodeApi

使用 [SmsCodeApi](#) 进行【验证码】短信的发送，例如说：用户手机验证码登录、用户忘记密码等等。它的方法如下：



验证码使用 `system_sms_code` 表进行存储，默认每天最多发送 10 条，每分钟发送 1 条，有效期为 10 分钟，可通过 `yudao.sms-code` 配置项进行自定义：



4.2 实战案例

以会员用户手机验证码登录为例子，讲解 `SmsCodeApi` 的使用。

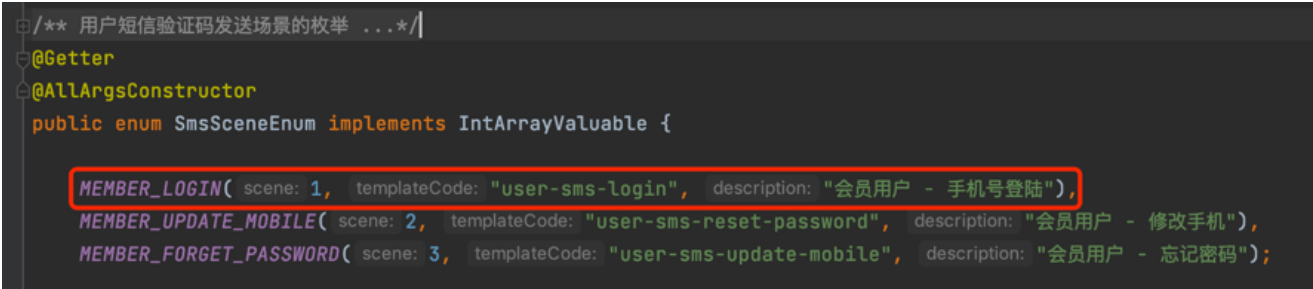
① 引入 `yudao-module-system-api` 依赖，如下图所示：



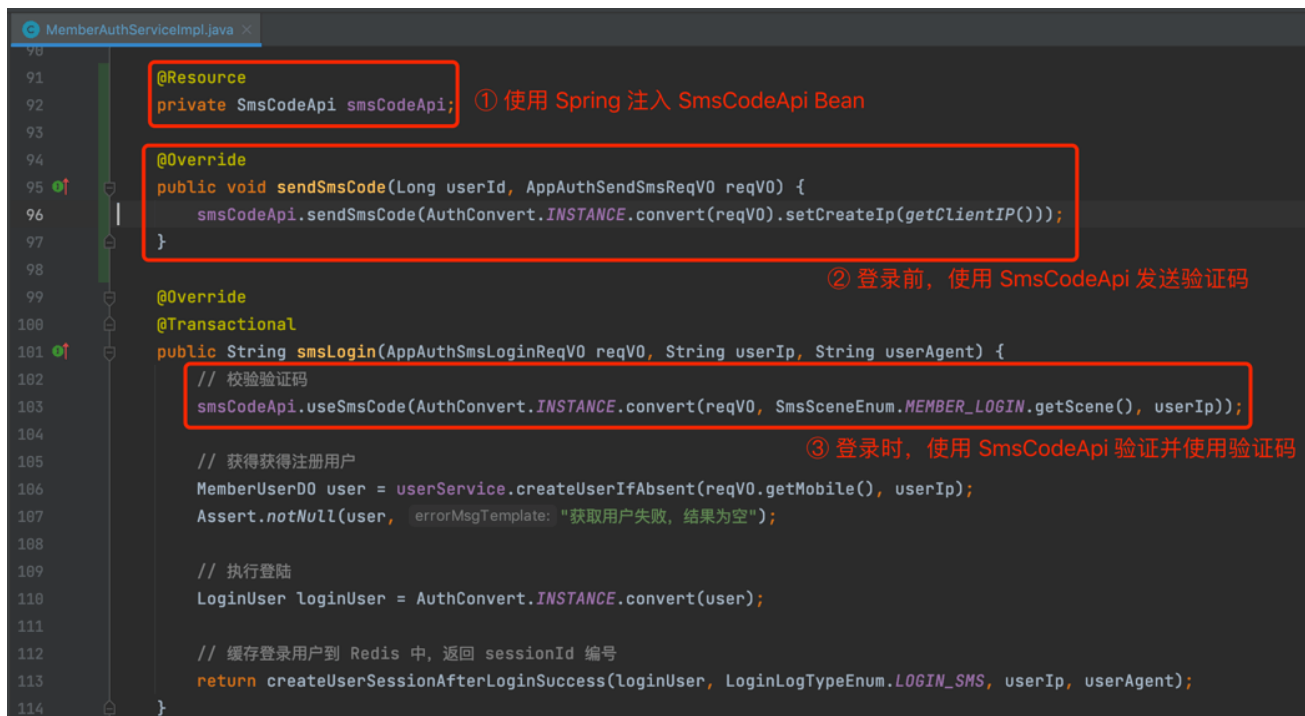
② 新建对应的短信模板，如下图所示：



③ 在 `SmsSceneEnum` 中，枚举会员用户的手机号登录的场景，如下图所示：



④ 使用 Spring 注入 SmsCodeApi Bean，调用对应的短信验证码的发送与使用方法。如下图 所示：

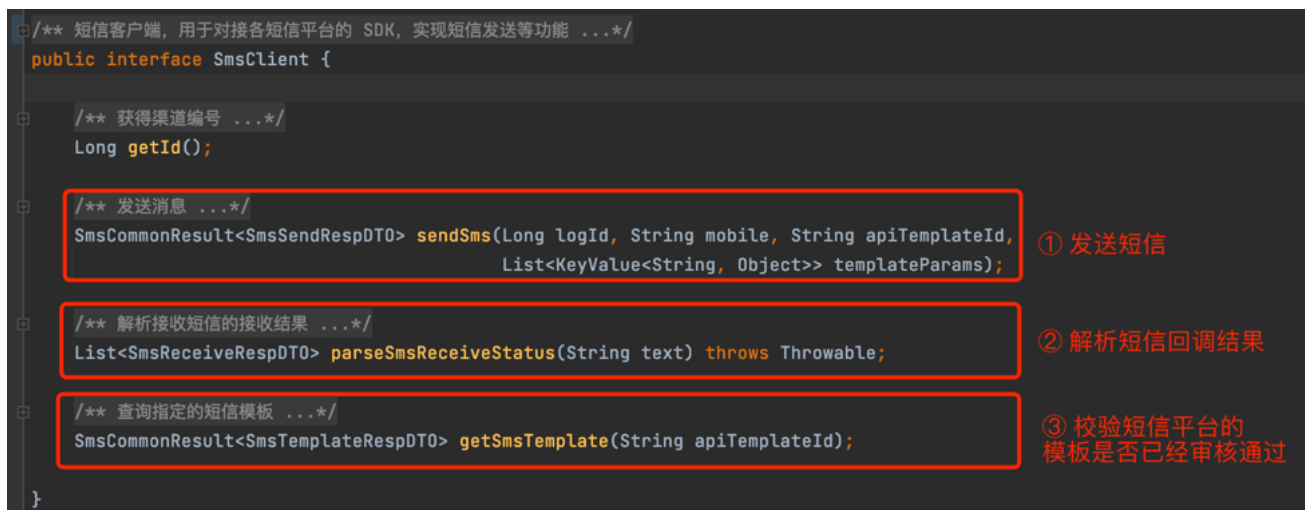


5. 短信客户端

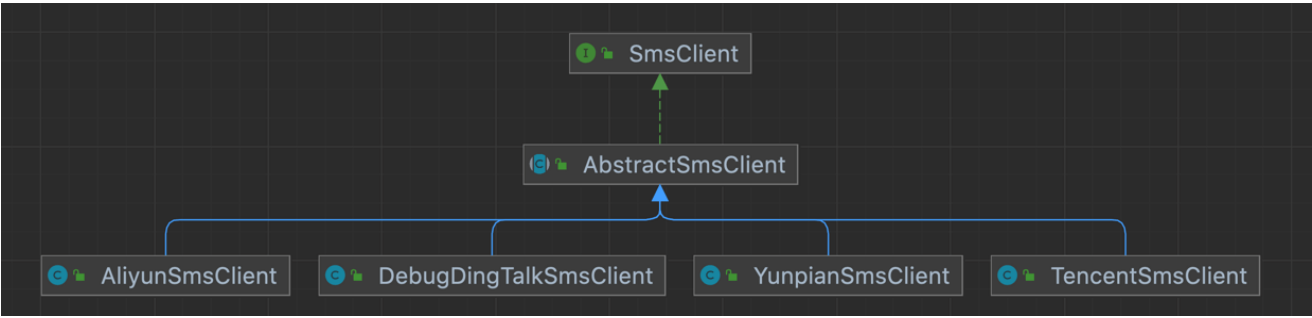
`yudao-spring-boot-starter-biz-sms` 业务组件，对接阿里云、腾讯云等短信平台，提供统一的短信客户端，提供给 `yudao-module-system` 的 `sms` 业务模块来调用。

5.1 SmsClient

`SmsClient` 接口，定义短信客户端的方法。代码如下：



每个短信平台，都对应一个 `SmsClient` 实现类。



5.2 SmsCodeMapping

SmsCodeMapping 接口，定义短信平台错误码转换成 标准错误码 的方法。代码如下：

SmsCodeMapping.java

```
1 package cn.iocoder.yudao.framework.sms.core.client;
2
3 import ...
4
5 /**
6  * 将 API 的错误码，转换为通用的错误码
7  * @see SmsCommonResult
8  * @see SmsFrameworkErrorCodeConstants
9  *
10  * @author 芋道源码
11  */
12 public interface SmsCodeMapping extends Function<String, ErrorCode> {
13 }
14
```

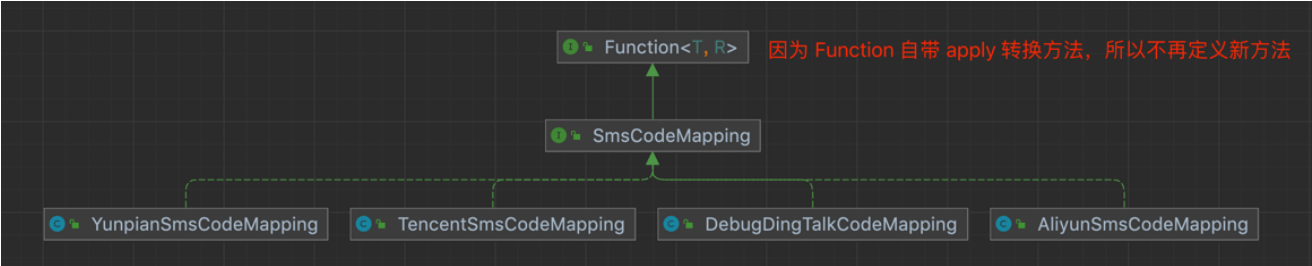
SmsFrameworkErrorCodeConstants.java

```
1 /**
2  * 短信框架的错误码枚举
3  * 短信框架，使用 2-001-000-000 段
4  *
5  * @author 芋道源码
6  */
7 public interface SmsFrameworkErrorCodeConstants {
8
9     ErrorCode SMS_UNKNOWN = new ErrorCode( code: 2001000000, message: "未知错误，需要解析");
10
11     // ===== 权限 / 限流等相关 2001000100 =====
12
13     ErrorCode SMS_PERMISSION_DENY = new ErrorCode( code: 2001000100, message: "没有发送短信的权限");
14
15     // 云片：可以配置 IP 白名单，只有在白名单中才可以发送短信
16     ErrorCode SMS_IP_DENY = new ErrorCode( code: 2001000100, message: "IP 不允许发送短信");
17
18     // 阿里云：将短信发送频率限制在正常的业务限流范围内。默认短信验证码：使用同一签名，对同一个手机号验证码，支持 1 条 / 分钟，5 条 / 小时，累计 10 条 / 天。
19     ErrorCode SMS_SEND_BUSINESS_LIMIT_CONTROL = new ErrorCode
20     ( code: 2001000102, message: "指定手机的发送限流");
21
22     // 阿里云：已经注册您在控制台设置的短信白名单号码，在短信发送时，安全设置 修改
23
24 }
25
```

AliyunSmsCodeMapping.java

```
1 package cn.iocoder.yudao.framework.sms.core.client.impl.aliyun;
2
3 import ...
4
5 /**
6  * 阿里云的 SmsCodeMapping 实现类
7  *
8  * 参见 https://help.aliyun.com/document_detail/101346.htm 文档
9  *
10  * @author 芋道源码
11  */
12 public class AliyunSmsCodeMapping implements SmsCodeMapping {
13
14     @Override
15     public ErrorCode apply(String apiCode) {
16         switch (apiCode) {
17             case "OK": return GlobalErrorCodeConstants.SUCCESS;
18             case "isv.ACCOUNT_NOT_EXISTS": return SmsFrameworkErrorCodeConstants.SMS_ACCOUNT_INVALID;
19             case "isv.ACCOUNT_ABNORMAL": return SmsFrameworkErrorCodeConstants.SMS_ACCOUNT_INVALID;
20             case "MissingAccessKeyId": return SmsFrameworkErrorCodeConstants.SMS_API_PARAM_ERROR;
21             case "isv.INVALID_PARAMETERS": return SmsFrameworkErrorCodeConstants.SMS_API_PARAM_ERROR;
22             case "isv.BUSINESS_LIMIT_CONTROL": return SmsFrameworkErrorCodeConstants.SMS_SEND_BUSINESS_LIMIT_CONTROL;
23             case "isv.DAY_LIMIT_CONTROL": return SmsFrameworkErrorCodeConstants.SMS_SEND_DAY_LIMIT_CONTROL;
24             case "isv.SMS_CONTENT_ILLEGAL": return SmsFrameworkErrorCodeConstants.SMS_TEMPLATE_INVALID;
25             case "isv.SMS_TEMPLATE_ILLEGAL": return SmsFrameworkErrorCodeConstants.SMS_TEMPLATE_INVALID;
26             case "isv.SIGNATURE_ILLEGAL": return SmsFrameworkErrorCodeConstants.SMS_SIGN_INVALID;
27             case "isv.SIGN_NAME_ILLEGAL": return SmsFrameworkErrorCodeConstants.SMS_SIGN_INVALID;
28             case "isv.AMOUNT_NOT_ENOUGH": return SmsFrameworkErrorCodeConstants.SMS_ACCOUNT_MONEY_NOT_ENOUGH;
29             case "isv.OUT_OF_SERVICE": return SmsFrameworkErrorCodeConstants.SMS_ACCOUNT_MONEY_NOT_ENOUGH;
30             case "isv.MOBILE_NUMBER_ILLEGAL": return SmsFrameworkErrorCodeConstants.SMS_MOBILE_INVALID;
31             case "isv.TEMPLATE_MISSING_PARAMETERS": return SmsFrameworkErrorCodeConstants.SMS_TEMPLATE_PARAM_ERROR;
32             case "isv.TEMPLATE_PARAM_ERROR": return SmsFrameworkErrorCodeConstants.SMS_TEMPLATE_PARAM_ERROR;
33         }
34     }
35 }
36
```

每个短信平台，都对应一个 SmsCodeMapping 实现类。



5.3 对接其它短信平台

如果你想要对接其它短信平台，自定义一个 SmsClient + SmsCodeMapping 实现类，并使用 SmsClientFactoryImpl 进行创建。代码如下：

```
private AbstractSmsClient createSmsClient(SmsChannelProperties properties) {
    SmsChannelEnum channelEnum = SmsChannelEnum.getByCode(properties.getCode());
    Assert.notNull(channelEnum, String.format("渠道类型(%s) 为空", channelEnum));
    // 创建客户端
    switch (channelEnum) {
        case ALIYUN: return new AliyunSmsClient(properties);
        case YUN_PIAN: return new YunpianSmsClient(properties);
        case DEBUG_DING_TALK: return new DebugDingTalkSmsClient(properties);
        case TENCENT: return new TencentSmsClient(properties);
    }
    // 创建失败, 错误日志 + 抛出异常
    log.error("[createSmsClient][配置({})] 找不到合适的客户端实现", properties);
    throw new IllegalArgumentException(String.format("配置(%s) 找不到合适的客户端实现", properties));
}
```

新增短信渠道的编码枚举,
并创建对应的短信客户端

6. 短信平台附录

一般情况下, 建议接入 2-3 个短信平台, 避免某个短信平台故障时, 影响业务的正常运行。例如说, 手机验证码的短信平台 A 故障时, 赶紧将短信验证码切换到短信平台 B 上, 否则用户将无法登录或是注册。

6.1 阿里云

- 短信 API 的账号、密钥, 可通过 [阿里云 —— AccessKey](#) 获取。
- 短信发送回调 URL, 可通过 [阿里云 —— 短信服务 —— 通用设置](#) 配置。

6.2 腾讯云

- 短信 API 的账号、密钥, 可通过 [腾讯云 —— API 密钥管理](#) 获取。

注意!!!

腾讯云需要额外使用 [SDKAppID](#) 参数, 它的账号需要采用 `SDKAppID secretId` 格式, 具体可见 [TencentSmsChannelProperties](#) 类。

- 短信发送回调 URL, 可通过 [腾讯云 —— 短信 —— 基础配置](#) 配置。

← [支付宝、微信退款接入](#)

[邮件配置](#) →



