# 短信服务器

作者：王长宏

时间：2016/08/03

## 功能概述：

为公司内部提供短信支持，达到不同渠道使用同一SDK实现，并且统一做容灾报警处理，减少开发同学在短信接入上的时间。

## 框架介绍

框架是由作者自己为短信服务器功能需求特殊定制的，架构在异步处理系统之上，由作者开发并提供维护，有3个请求入口：

1. Api 入口：入口文件 /api.php，用来提供api支持
2. 通用Web入口：入口文件/index.php，用来提供渠道方异步通知入口
3. Cli 入口：入口文件/cli.php，用来提供cli执行程序

## 系统介绍

### 目录：

* 1. base：存放核心基础文件目录（基类，mysql，router，autoload等）
  2. bin: shell 脚本文件目录（守护进程，用来守护cli进程）
  3. config：配置文件目录
  4. doc：文档目录
  5. library：公共方法目录（SmsSender插件，Notify系统等，公共方法等）
  6. model：数据库抽象文件目录
  7. templates：模板文件目录

### 主要系统功能

#### 异步处理系统

请参考《异步处理系统.docx》

#### SmsSender系统

SmsSender属于Notify系统中短信驱动器的实现，根据不同的短信channel，实现不同的SmsSender。要实现SmsSender必须继承\library\SmsSender\SmsSenderInterface，该接口规定了实现方法。对应文件：

1) . /library/SmsSender/\*

#### 发短信系统

为保证大数据并发的稳定性，发短信流程用消息队列实现，通过api入口请求会生成短信记录并放到队列里，由一个监控队列的程序（Notify->listen()）异步发送短信。详细流程请参考

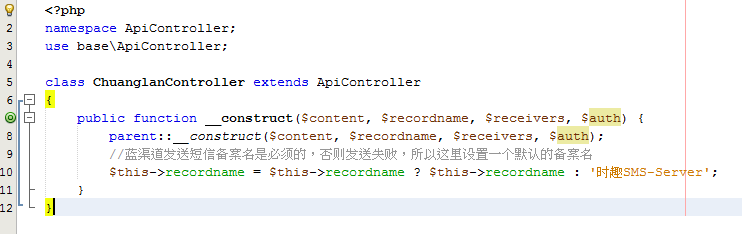
1. /doc/api接口.puml
2. /doc/base\_ApiController.puml
3. /doc/短信发送流程.puml

## 渠道接入

以接入创蓝短信渠道作为示例，按照发短信的流程，接入渠道共分为4个部分：通过api接受短信内容、通过SmsSender发送短信、监听渠道回调（发送状态、上行消息）、监听进程（或其他cli程序）部署，下面分别来介绍：

### 通过api接受短信内容

在ApiController文件夹中创建ChuanglanController.php 并实现 \ ApiController\ChuanglanController 类，并使他继承 \base\ApiController，如下：



在这个类里仅仅按这个渠道的特殊需求，重新定义了构造方法，可见通过api接收短信内容，并生成Notify数据都是由基类实现，那么基类里有哪些需要关注的方法呢？

1. ***构造函数***



（UML：/doc/uml/base\_ApiController.puml）

1. ***发送函数***

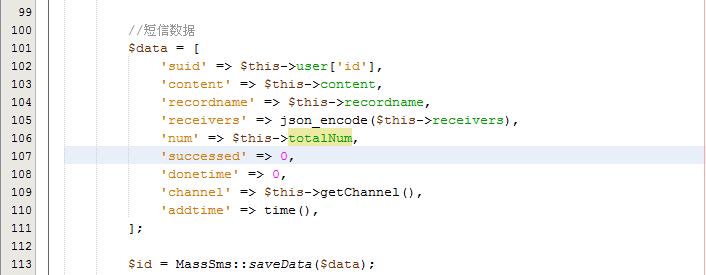


在接入渠道的时候，根据不同渠道的特点，选择覆盖基类的方法。最后只需调用：$this->sendToNotify($id, NotifyContent::TYPE\_SMS); 发送Notify数据($id：代表短信数据在数据库的主键，NotifyContent::TYPE\_SMS：表示Notify数据是sms类型的)，如果短信需要扣费，则调用：$this->auth->consume($this->totalNum)，最后返回200状态码。

### 实现SmsSender

在/library/SmsSender/下创建Chuanglan.php, 实现SmsSenderInterface 接口创建 \library\SmsSender\Chuanglan 类。文件名一定要个在第1步创建的类的前缀一样。

SmsSenderInterface 仅仅规定的一个方法：send($data), 这个$data就是在接受短信内容的时候保存的短信数据，如下：



具体的发送功能实现，是根据接入的短信渠道分别实现的，可以参考\library\SmsSender\Chuanglan 类。

### 监听渠道回调（发送状态、上行消息）

渠道方回调一般以http调用的方式触发，所以要实现Web调用接口。首先在WebController下面创建ChuanglanController.php，内容如下：



* + - * 1. 接受短信发送状态：

因为创蓝渠道返回短信发送状态是单条返回的，比如一起给1000个人发短信，那么就会产生1000条状态返回值，为了防止并发引起的服务器崩溃，所以接受短信的状态的请求直接压入Redis队列，用 异步的方式来处理发送状态。（这里的actionStatusp仅仅是PHP做测试用的，而运行环境的接受状态是用Ngnix+lua+redis实现的，访问地址http://hostname/chuanglan/status），访问actionStatusp方法的路径是http://hostname/chuanglan/statusp

* + - * 1. 接受上行短信消息

actionListen() 方法,仅仅把上行短信整理入库，调用方式：<http://hostname/chuanglan/listen>。

3)异步处理短信发送状态

在/CommandsController 文件夹下创建ChuanglanController.php实现 \CommandsController\ChuanglanController类，如下图：



调用actionListenSmsStatus的方式是:



### 监听进程（或其他cli程序）部署

因为监听程序是一直在后台运行的，所以需要把他部署到守护进程里，然后使用守护进程启动（参考三.2.d）