**品优购电商系统开发**

**第14章**

**HttpClient&阿里大于&用户中心**

传智播客.黑马程序员

# 课程目标

目标1：完成HTTPClient会GET和POST请求

目标2：在阿里云网站中注册短信功能

目标3：自己手动编写之前品优购写的接口

目标4：能在工程中使用HTTPClient调用短信接口

目标5：完成用户注册功能

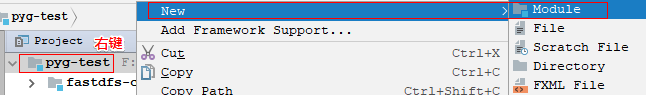
# 1.HttpClient介绍

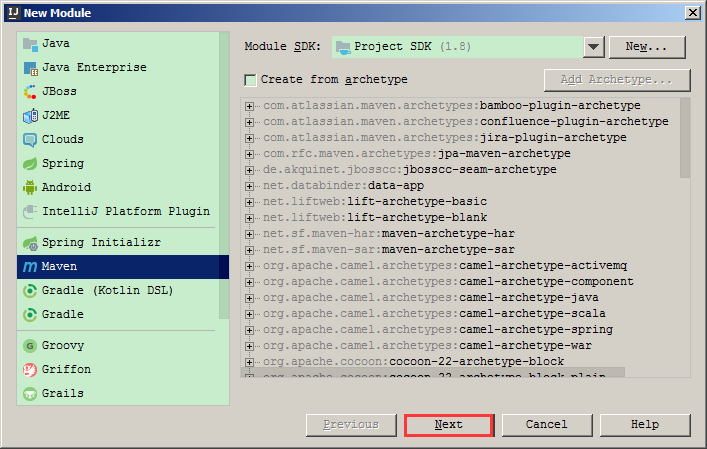
HttpClient 是 Apache Jakarta(雅加达，印度尼西亚首都)Common下的子项目，可以用来提供高效的、最新的、功能丰富的支持 HTTP 协议的客户端编程工具包，并且它支持HTTP 协议最新的版本和建议，HttpClient 提供的主要的功能。  
（1）实现了所有 HTTP 的方法（GET,POST,PUT,DELETE等）  
（2）支持自动转向  
（3）支持 HTTPS协议  
（4）支持代理服务器

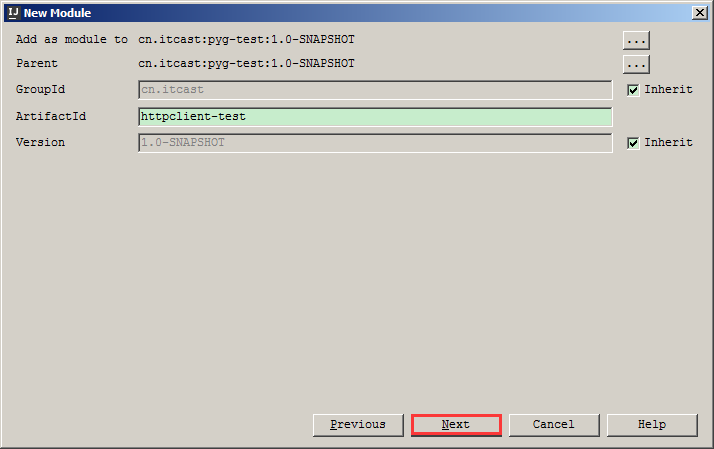
# 2.HttpClient入门

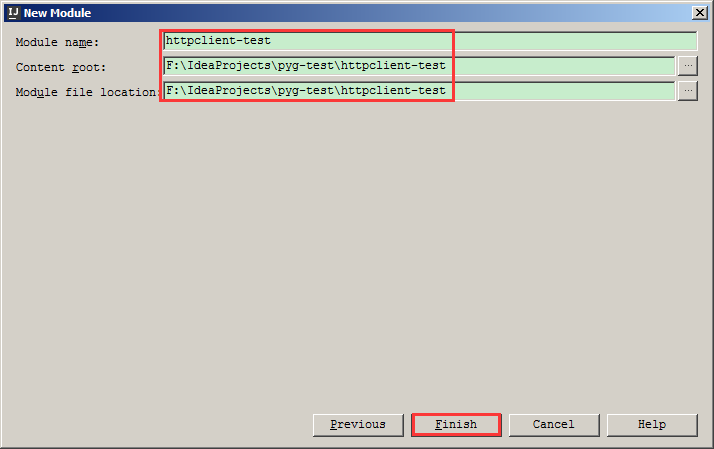
## 2.1搭建测试模块

切换到pyg-test项目，创建httpclient-test模块（打包方式jar）









httpclient-test/pom.xml

|  |
| --- |
| <**dependencies**>  <**dependency**>  <**groupId**>org.apache.httpcomponents</**groupId**>  <**artifactId**>httpclient</**artifactId**>  <**version**>4.5.3</**version**>  </**dependency**> </**dependencies**> |

## 2.2 GET请求

httpclient-test\src\test\java\cn\itcast\HttpClientTest.java

|  |
| --- |
| **public class** HttpClientTest {  */\*\* 定义HttpClient \*/* **private** CloseableHttpClient **httpClient** = HttpClients.*createDefault*();  @Test  **public void** get1() **throws** Exception{  *// 创建HttpGet对象* **HttpGet httpGet = new HttpGet("http://www.baidu.com");**  *// 执行请求，得到响应对象* CloseableHttpResponse response = **httpClient**.execute(httpGet);  *// 判断响应状态码 200* **if** (response.getStatusLine().getStatusCode() == 200){  *// 获取响应数据* String content = EntityUtils.*toString*(response.getEntity(),**"UTF-8"**);  System.***out***.println(content);  }  *// 关闭响应对象* response.close();  } } |

## 2.3 GET请求(带参数)

|  |
| --- |
| @Test **public void** get2() **throws** Exception{  *// 定义请求URI* URIBuilder uriBuilder = **new** URIBuilder(**"http://www.baidu.com/s"**);  *// 设置请求参数* uriBuilder.setParameter(**"wd"**,**"java"**);  *// 构建URI(http://www.baidu.com/s?wd=java)* URI uri = uriBuilder.build();  System.***out***.println(**"URI: "** + uri);  *// 创建HttpGet对象* HttpGet httpGet = **new** HttpGet(uri);  *// 执行请求，得到响应对象* CloseableHttpResponse response = **httpClient**.execute(httpGet);  *// 判断响应状态码 200* **if** (response.getStatusLine().getStatusCode() == 200){  *// 获取响应数据* String content = EntityUtils.*toString*(response.getEntity(), **"UTF-8"**);  System.***out***.println(content);  }  *// 关闭响应对象* response.close(); } |

## 2.4 POST请求

|  |
| --- |
| @Test **public void** post1() **throws** Exception{  *// 创建HttpPost对象* **HttpPost httpPost = new HttpPost("http://www.baidu.com");**  *// 执行请求，得到响应对象* CloseableHttpResponse response = **httpClient**.execute(httpPost);  *// 判断响应状态码 200* **if** (response.getStatusLine().getStatusCode() == 200){  *// 获取响应数据* String content = EntityUtils.*toString*(response.getEntity(), **"UTF-8"**);  System.***out***.println(content);  }  *// 关闭响应对象* response.close(); } |

## 2.5 POST请求(带参数)

|  |
| --- |
| @Test **public void** post2() **throws** Exception{  *// 创建HttpPost对象* HttpPost httpPost = **new** HttpPost(**"http://www.baidu.com"**);  ***// 定义List集合封装请求参数* List<NameValuePair> nvpList = new ArrayList<>();  *// 添加请求参数* nvpList.add(new BasicNameValuePair("username","admin"));  *// 添加请求参数* nvpList.add(new BasicNameValuePair("password","123456"));  *// 创建form表单实体* UrlEncodedFormEntity formEntity = new UrlEncodedFormEntity(nvpList,"UTF-8");  *// 设置请求参数实体* httpPost.setEntity(formEntity);**  *// 执行请求，得到响应对象* CloseableHttpResponse response = **httpClient**.execute(httpPost);  *// 判断响应状态码 200* **if** (response.getStatusLine().getStatusCode() == 200){  *// 获取响应数据* String content = EntityUtils.*toString*(response.getEntity(),**"UTF-8"**);  System.***out***.println(content);  }  *// 关闭响应对象* response.close(); } |

## 2.6 HttpClientUtils工具类

如果每次请求都需要自己写Get请求与Post请求代码特别麻烦，那么我们引入HttpClientUtils工具类简化开发(“资料\HttpClientUtils.java”)

|  |
| --- |
| @Test **public void** testGet(){  HttpClientUtils httpClientUtils = **new** HttpClientUtils(**false**);  String content = httpClientUtils.sendGet(**"http://www.baidu.com"**, **null**);  System.***out***.println(**"状态码："** + httpClientUtils.getStatusCode());  System.***out***.println(**"响应内容"** + content); } @Test **public void** testGetParam(){  HttpClientUtils httpClientUtils = **new** HttpClientUtils(**false**);  Map<String, String> param = **new** HashMap<>();  param.put(**"wd"**, **"java"**);  String content = httpClientUtils.sendGet(**"http://www.baidu.com/s"**,param);  System.***out***.println(**"状态码："** + httpClientUtils.getStatusCode());  System.***out***.println(**"响应内容"** + content); } |

# 3.阿里大于短信

## 3.1阿里大于简介

阿里大于是**阿里云**旗下产品，融合了三大运营商的通信能力，通过将传统通信业务和能力与互联网相结合，创新融合阿里巴巴生态内容，全力为中小企业和开发者提供优质服务阿里大于提供包括短信、语音、流量直充、私密专线、店铺手机号等个性化服务。通过阿里大于打通三大运营商通信能力，全面融合阿里巴巴生态，以开放API及SDK的方式向开发者提供通信和数据服务，更好地支撑企业业务发展和创新服务。

## 3.2准备工作

### 3.2.1注册账户

首先我们先进入“阿里大于” [www.alidayu.com](http://www.baidu.com/baidu.php?url=ZzmK000GQJC8hW1POcJEP4KEa9MqOc64Kv9VaSYkPBGRtKKgrM9zjpMjkg0QSyMnQxKRQWKjPMwVPRtuZHdquplwvyTojNksmJ4fXyH6PR2oD2XMXttLlESx3RspZ_3aMmqjEv4UOpBl4_suPNxrqoJgxNO3dD62O8C6tR-EGupmJLpXA6.7D_iuTZ-GYkYApacrSapBsAprFMw6wKL9JuBBHwmEukmr_X5is1gsdeEQx3txQ_erMktJOAEWkl_LT8FB8zzEWgEvNSQQPLuy--rexduYt8FWojG4THZgZG4THZy7MHWuxJB4THAe_tpqOudEmtxZ_L2N9h9mLU2Srqf.U1Yk0ZDq_2Q0eeH1YSn0IjvsC_jEV55H3sKGUHYznWR0u1dBuHms0ZNG5yF9pywd0ZKGujYk0APGujY1rjR0UgfqnH0YP7tknjDLg1DsnH-xn1msnfKopHYs0ZFY5Hm4P0K-pyfqnHfdndtznHDzPdtkP10snNtznW01PNtznWDLPdtznWDdPsKBpHYznjf0UynqnHR4PWRzPWbdP-t1PWn4rHmknj9xn1mYPjbsnHDYg100TgKGujYs0Z7Wpyfqn0KzuLw9u1Ys0AqvUjYzPWm4Qywlg1cvPW-mQywlg1msn1nVn7t3PHbVnsK9mWYsg100ugFM5H00TZ0qn0K8IM0qna3snj0snj0sn0KVIZ0qn0KbuAqs5H00ThCqn0KbugmqTAn0uMfqn0KspjYs0Aq15H00mMTqnH00UMfqn0K1XWY0IZN15HDLPjnzPjnzP1b4rjnzP1RvPWD10ZF-TgfqnHf3rjcYPjDLPjmvP6K1pyfqmyPbryNBmH6snjK9ujRdnsKWTvYqrjuAwjNarDRdnH03rH7aPsK9m1Yk0ZK85H00TydY5H00Tyd15H00XMfqn0KVmdqhThqV5HDsPjuxn7tsg100uA78IyF-gLK_my4GuZnqn7tsg1Kxn0KbIA-b5H00ugwGujYVnfK9TLKWm1Ys0ZNspy4Wm1Ys0Z7VuWYs0AuWIgfqn0KhXh6qn0KlTAkdT1Ys0A7buhk9u1Yk0APzm1YznWnz&us=0.0.0.0.0.0.10&us=0.0.0.0.0.0.30&ck=0.0.0.0.0.0.0.0&shh=www.baidu.com" \t "https://www.baidu.com/_blank) (https://dayu.aliyun.com/)





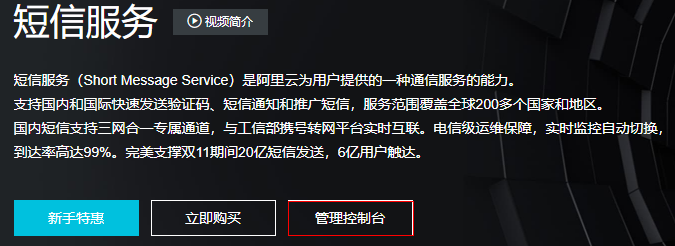
注册账号后，再在手机下载“阿里云”应用，登录，然后进行在线实名认证。

### 3.2.2登录系统

使用刚才注册的账号进行登录。







点击进入控制台：



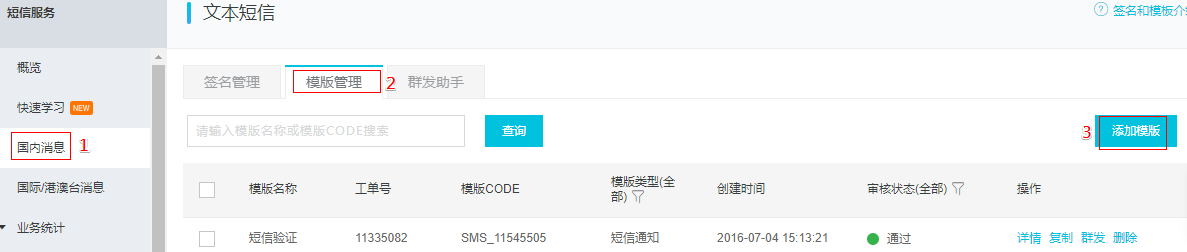


### 3.2.3申请签名





### 3.2.4申请模板





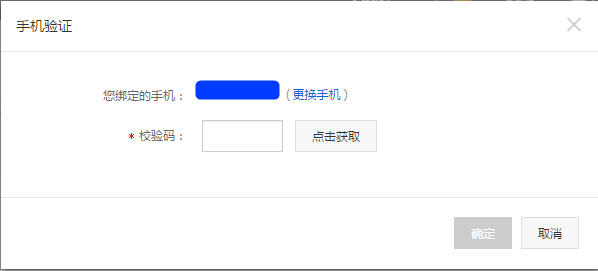
### 3.2.5创建accesskeys











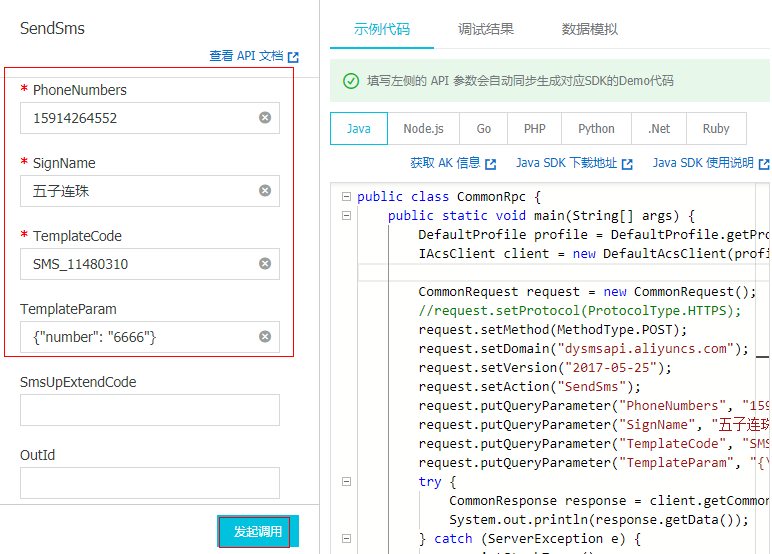


## 3.3下载SDK及DEMO

<https://help.aliyun.com/product/44282.html>



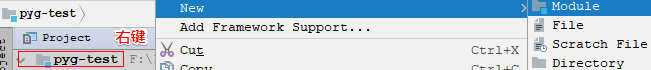


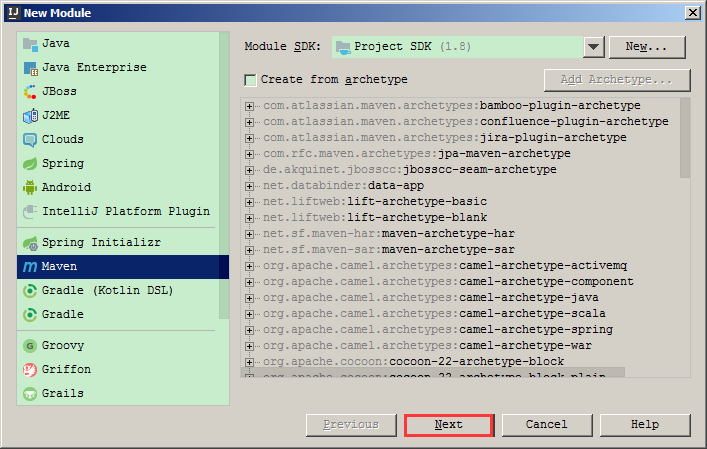


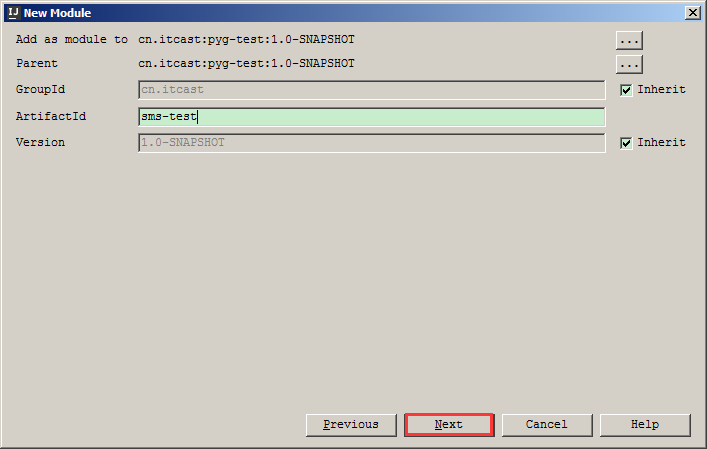
## 3.4发送短信测试

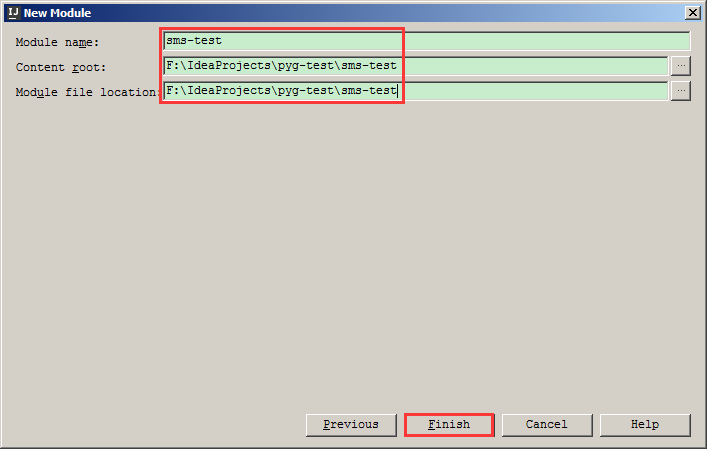
### 3.4.1搭建测试工程

在pyg-test项目中，创建sms-test模块(jar)。









sms-test/pom.xml加入依赖：

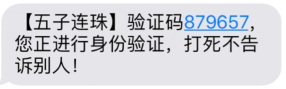
|  |
| --- |
| *<?***xml version="1.0" encoding="UTF-8"***?>* <**project xmlns="http://maven.apache.org/POM/4.0.0"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0  http://maven.apache.org/xsd/maven-4.0.0.xsd"**>  <**parent**>  <**artifactId**>pyg-test</**artifactId**>  <**groupId**>cn.itcast</**groupId**>  <**version**>1.0-SNAPSHOT</**version**>  </**parent**>  <**modelVersion**>4.0.0</**modelVersion**>  <**artifactId**>sms-test</**artifactId**>   <**dependencies**>  *<!-- 阿里大于 -->* <**dependency**>  <**groupId**>com.aliyun</**groupId**>  <**artifactId**>aliyun-java-sdk-core</**artifactId**>  <**version**>4.1.0</**version**>  </**dependency**>  *<!-- fastjson -->* <**dependency**>  <**groupId**>com.alibaba</**groupId**>  <**artifactId**>fastjson</**artifactId**>  <**version**>1.2.31</**version**>  </**dependency**>  </**dependencies**> </**project**> |

### 3.4.2短信发送代码

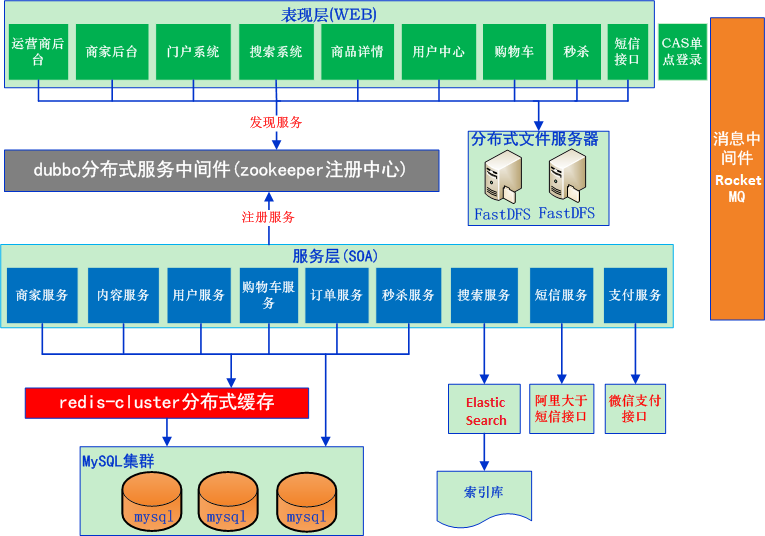
sms-test/src/main/java/cn/itcast/sms/util/包下，创建短信发送工具类SmsSendUtil.java：

|  |
| --- |
| */\*\*  \* 短信发送工具类  \*/* **public class** SmsSendUtil {   *// 产品域名* **private static final** String ***DOMAIN*** = **"dysmsapi.aliyuncs.com"**;  *// 签名KEY* **private static final** String ***ACCESS\_KEY\_ID*** = **"LTAIiKVKFzm3Vsri"**;  *// 签名密钥* **private static final** String ***ACCESS\_KEY\_SECRET*** = **"ww9nVlltvqfhjSWscfoVq04M7aItPY"**;  *// 短信模板ID: SMS\_11480310* **private static final** String ***TEMPLATE\_CODE*** = **"SMS\_11480310"**;  *// 短信签名* **private static final** String ***SIGN\_NAME*** = **"五子连珠"**;   **public static boolean** sendSms(String phone, String code){  **try**{  DefaultProfile profile = DefaultProfile  .*getProfile*(**"default"**, ***ACCESS\_KEY\_ID***, ***ACCESS\_KEY\_SECRET***);  IAcsClient client = **new** DefaultAcsClient(profile);  *// 封装请求参数* CommonRequest request = **new** CommonRequest();  *// 设置公用请求参数* request.setMethod(MethodType.***POST***);  request.setDomain(***DOMAIN***);  request.setVersion(**"2017-05-25"**);  request.setAction(**"SendSms"**);  *// 设置短信请求参数* request.putQueryParameter(**"PhoneNumbers"**, phone);  request.putQueryParameter(**"SignName"**, ***SIGN\_NAME***);  request.putQueryParameter(**"TemplateCode"**, ***TEMPLATE\_CODE***);  request.putQueryParameter(**"TemplateParam"**, **"{\"number\": \""**+code+**"\"}"**);   CommonResponse response = client.getCommonResponse(request);  System.***out***.println(**"响应数据："** + response.getData());  Map<String,String> resMap = JSON.*parseObject*(response.getData(),Map.**class**);   **return "OK"**.equals(resMap.get(**"Code"**));  } **catch** (Exception e) {  **throw new** RuntimeException(e);  }  }  **public static void** main(String[] args) {  **boolean** success = *sendSms*(**"15914264552"**, **"879657"**);  System.***out***.println(**"success = "** + success);  } } |

执行main方法我们就可以在手机上收到短信啦



# 搭建短信工程



## 4.1 pinyougou-sms-service(war)

与pinyougou-content-service服务层的构建方式一样，拷贝pinyougou-content-service模块中的web.xml、applicationContext-service.xml、log4j.properties。

pinyougou-sms-service/pom.xml：

|  |
| --- |
| <**properties**>  *<!-- 指定端口号 -->* <**port**>9004</**port**> </**properties**> <**dependencies**>  *<!-- 阿里大于 -->* <**dependency**>  <**groupId**>com.aliyun</**groupId**>  <**artifactId**>aliyun-java-sdk-core</**artifactId**>  <**version**>4.1.0</**version**>  </**dependency**> </**dependencies**> |

pinyougou-sms-service/src/main/resources/applicationContext-service.xml：

|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"***?>* <**beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"  xmlns:context="http://www.springframework.org/schema/context"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://code.alibabatech.com/schema/dubbo  http://code.alibabatech.com/schema/dubbo/dubbo.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd"**>   *<!-- 配置加载属性文件 -->* <**context:property-placeholder  location="classpath\*:props/\*.properties"**/>   *<!-- ############## 配置dubbo服务提供者 ############## -->  <!-- 配置当前应用的名称 -->* <**dubbo:application name="pinyougou-sms-service"**/>  *<!-- 配置zookeeper作为注册中心，注册服务地址 -->* <**dubbo:registry protocol="zookeeper" address="192.168.12.131:2181"**/>  *<!-- 用dubbo协议在20883端口暴露服务 -->* <**dubbo:protocol name="dubbo" port="20883"**/>  *<!-- 配置采用包扫描来暴露服务 -->* <**dubbo:annotation package="com.pinyougou.sms.service.impl"**/> </**beans**> |

## 4.2 pinyougou-sms-web(war)

与pinyougou-search-web构建方式一样，拷贝pinyougou-search-web模块中的web.xml、springmvc.xml、log4j.properties

pinyougou-sms-web/pom.xml：

|  |
| --- |
| <**properties**>  *<!-- 更改tomcat端口号 -->* <**port**>**9106**</**port**> </**properties**> |

pinyougou-sms-web/src/main/webapp/WEB-INF/web.xml：

|  |
| --- |
| *<?***xml version="1.0" encoding="UTF-8"***?>* <**web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns="http://java.sun.com/xml/ns/javaee"  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee  http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"  version="3.0"**>  *<!-- 配置Spring MVC前端控制器(核心控制器) -->* <**servlet**>  <**servlet-name**>pinyougou</**servlet-name**> <**servlet-class**>org.springframework.web.servlet.DispatcherServlet</**servlet-class**>  <**init-param**>  <**param-name**>contextConfigLocation</**param-name**>  <**param-value**>classpath:springmvc.xml</**param-value**>  </**init-param**>  <**load-on-startup**>1</**load-on-startup**>  </**servlet**>  <**servlet-mapping**>  <**servlet-name**>pinyougou</**servlet-name**>  <**url-pattern**>/</**url-pattern**>  </**servlet-mapping**>  *<!-- 配置全局编码过滤器 -->* <**filter**>  <**filter-name**>characterEncodingFilter</**filter-name**> <**filter-class**>org.springframework.web.filter.CharacterEncodingFilter</**filter-class**>  <**init-param**>  <**param-name**>encoding</**param-name**>  <**param-value**>UTF-8</**param-value**>  </**init-param**>  </**filter**>  <**filter-mapping**>  <**filter-name**>characterEncodingFilter</**filter-name**>  <**url-pattern**>/\*</**url-pattern**>  </**filter-mapping**> </**web-app**> |

pinyougou-sms-web/src/main/resources/springmvc.xml：

|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"***?>* <**beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:mvc="http://www.springframework.org/schema/mvc"  xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"  xmlns:context="http://www.springframework.org/schema/context"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/mvc  http://www.springframework.org/schema/mvc/spring-mvc.xsd  http://code.alibabatech.com/schema/dubbo  http://code.alibabatech.com/schema/dubbo/dubbo.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd"**>   *<!-- 开启组件扫描 -->* **<context:component-scan base-package="com.pinyougou.sms.controller"/>**  *<!-- 开启MVC注解驱动 -->* <**mvc:annotation-driven**>  *<!-- 配置消息转换 -->* <**mvc:message-converters**>  *<!-- 配置用fastjson作为json处理框架 -->* <**bean class="com.alibaba.fastjson.support.spring.FastJsonHttpMessageConverter"**>  *<!-- 配置支持的类型 -->* <**property name="supportedMediaTypes"  value="application/json;charset=UTF-8"**/>  </**bean**>  </**mvc:message-converters**>  </**mvc:annotation-driven**>  *<!-- 配置静态资源用WEB容器默认的servlet来处理 -->* <**mvc:default-servlet-handler**/>   *<!-- ############## 配置dubbo服务消费者 ############## -->  <!-- 配置当前应用的名称 -->* <**dubbo:application name="pinyougou-sms-web"**/>  *<!-- 配置zookeeper作为注册中心，注册服务地址 -->* <**dubbo:registry protocol="zookeeper" address="192.168.12.131:2181"**/>  *<!-- 配置采用包扫描来引用服务，产生服务接口的代理对象 -->* <**dubbo:annotation package="com.pinyougou.sms.controller"**/> </**beans**> |

## 4.3配置域名访问

修改hosts文件：127.0.0.1 sms.pinyougou.com

修改Nginx的nginx.conf配置反向代理：

|  |
| --- |
| **server {**  **listen 80;**  **server\_name sms.pinyougou.com;**  **proxy\_set\_header X-Forwarded-Host $host;**  **proxy\_set\_header X-Forwarded-Server $host;**  **location / {**  **proxy\_pass http://127.0.0.1:9106;**  **proxy\_connect\_timeout 600;**  **proxy\_read\_timeout 600;**  **}**  **}** |

## 4.4短信发送(服务层)

### 4.4.1服务接口

pinyougou-interface\src\main\java\com\pinyougou\service\SmsService.java

|  |
| --- |
| */\*\* 短信服务接口 \*/* **public interface** SmsService {  */\*\*  \* 发送短信方法  \** ***@param phone*** *手机号码  \** ***@param signName*** *签名  \** ***@param templateCode*** *短信模板  \** ***@param templateParam*** *模板参数(json格式)  \** ***@return*** *true 发送成功 false 发送失败  \*/* **public boolean** sendSms(String phone, String signName,  String templateCode, String templateParam); } |

### 4.4.2服务实现

pinyougou-sms-service\src\main\java\com\pinyougou\sms\service\impl\SmsServiceImpl.java

|  |
| --- |
| */\*\* 短信服务接口实现类 \*/* @Service **public class** SmsServiceImpl **implements** SmsService {  *// 产品域名* **private static final** String ***DOMAIN*** = **"dysmsapi.aliyuncs.com"**;  *// 签名KEY* @Value(**"${sms.accessKeyId}"**)  **private** String **accessKeyId**;  *// 签名密钥* @Value(**"${sms.accessKeySecret}"**)  **private** String **accessKeySecret**;   */\*\*  \* 发送短信方法  \** ***@param phone*** *手机号码  \** ***@param signName*** *签名  \** ***@param templateCode*** *短信模版  \** ***@param templateParam*** *模版参数(json格式)  \** ***@return*** *true 发送成功 false 发送失败  \*/* **public boolean** sendSms(String phone, String signName,  String templateCode, String templateParam){  **try** {   DefaultProfile profile = DefaultProfile  .*getProfile*(**"default"**, **accessKeyId**, **accessKeySecret**);  IAcsClient client = **new** DefaultAcsClient(profile);  *// 封装请求参数* CommonRequest request = **new** CommonRequest();  *// 设置公用请求参数* request.setMethod(MethodType.***POST***);  request.setDomain(***DOMAIN***);  request.setVersion(**"2017-05-25"**);  request.setAction(**"SendSms"**);  *// 设置短信请求参数* request.putQueryParameter(**"PhoneNumbers"**, phone);  request.putQueryParameter(**"SignName"**, signName);  request.putQueryParameter(**"TemplateCode"**, templateCode);  request.putQueryParameter(**"TemplateParam"**, templateParam);   CommonResponse response = client.getCommonResponse(request);  System.***out***.println(**"响应数据："** + response.getData());  Map<String,String> resMap = JSON.  *parseObject*(response.getData(),Map.**class**);   **return "OK"**.equals(resMap.get(**"Code"**));  }**catch** (Exception ex){  **throw new** RuntimeException(**"短信发送出现异常！"**, ex);  }  } } |

pinyougou-sms-service\src\main\resources\props\application.properties

|  |
| --- |
| **sms.accessKeyId**=**LTAIiKVKFzm3Vsri sms.accessKeySecret**=**ww9nVlltvqfhjSWscfoVq04M7aItPY** |

## 4.5短信发送接口(表现层)

pinyougou-sms-web\src\main\java\com\pinyougou\sms\controller\SmsController.java

|  |
| --- |
| @RestController @RequestMapping(**"/sms"**) **public class** SmsController {  @Reference(timeout=10000)  **private** SmsService **smsService**;  @PostMapping(**"/sendSms"**)  **public** Map<String, Object> sendSms(String phone, String signName,  String templateCode, String templateParam){  *// 发送短信* **boolean** success = **smsService**.sendSms(phone,  signName,templateCode, templateParam);  Map<String, Object> map = **new** HashMap<>();  map.put(**"success"**, success);  **return** map;  } } |

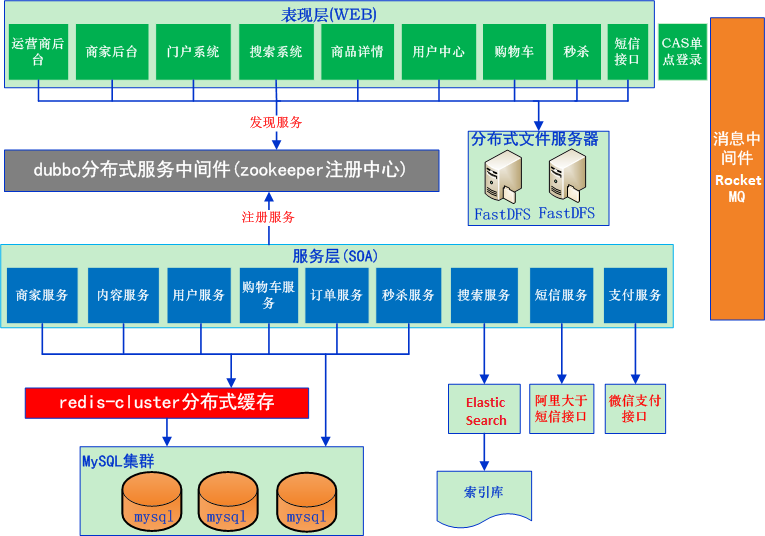
## 4.6短信发送接口文档

通过以上实现，在企业中都会统一向外部暴露接口让其他工程调用，那么负责写这个短信接口的人员要写一个开发文档（这个文档在企业中是必不可少的）；文档的格式：

**短信发送接口：**

|  |  |
| --- | --- |
| 请求方式 | POST |
| 请求URL | http://sms.pinyougou.com/sms/sendSms |
| 请求参数 | phone String 必须 待发送手机号 signName String 必须 短信签名-可在短信控制台中找到 templateCode String 必须 短信模板-可在短信控制台中找到  templateParam String 必须 模板中的变量替换 JSON 串,如模板内容为"亲爱的${name},您的验证码为${code}"时,此处的值为{**"name"** : **""**, **"code"** : **""**} |
| 示例 | 暂无 |
| 返回值 | JSON格式字符串：{**"success"** : **true|false**}  说明：true: 代表发送成功、false:代表发送失败 |

# 5.搭建用户中心工程



## 5.1 pinyougou-user-service(war)

与pinyougou-sms-service服务层的构建方式一样，拷贝pinyougou-sms-service模块中的web.xml、applicationContext-service.xml、log4j.properties。

pinyougou-user-service/pom.xml：

|  |
| --- |
| <**properties**>  *<!-- 指定端口号 -->* <**port**>9005</**port**> </**properties**> |

pinyougou-user-service/src/main/resources/applicationContext-service.xml：

|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"***?>* <**beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"  xmlns:context="http://www.springframework.org/schema/context"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://code.alibabatech.com/schema/dubbo  http://code.alibabatech.com/schema/dubbo/dubbo.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd"**>  *<!-- 配置加载属性文件 -->* <**context:property-placeholder  location="classpath\*:props/\*.properties"**/>  *<!-- 导入数据访问配置文件 -->* <**import resource="classpath:applicationContext-mapper.xml"**/>  *<!-- 导入redis配置文件 -->* <**import resource="classpath:applicationContext-redis.xml"**/>   *<!-- ############## 配置dubbo服务提供者 ############## -->  <!-- 配置当前应用的名称 -->* <**dubbo:application name="pinyougou-user-service"**/>  *<!-- 配置zookeeper作为注册中心，注册服务地址 -->* <**dubbo:registry protocol="zookeeper" address="192.168.12.131:2181"**/>  *<!-- 用dubbo协议在20884端口暴露服务 -->* <**dubbo:protocol name="dubbo" port="20884"**/>  *<!-- 配置采用包扫描来暴露服务 -->* <**dubbo:annotation package="com.pinyougou.user.service.impl"**/> </**beans**> |

## 5.2 pinyougou-user-web(war)

与pinyougou-sms-web构建方式一样，拷贝pinyougou-sms-web模块中的web.xml、springmvc.xml、log4j.properties

pinyougou-user-web/pom.xml：

|  |
| --- |
| <**properties**>  *<!-- 更改tomcat端口号 -->* <**port**>**9107**</**port**> </**properties**> |

pinyougou-user-web/src/main/webapp/WEB-INF/web.xml：

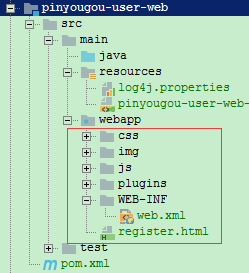
|  |
| --- |
| <**welcome-file-list**>  <**welcome-file**>register.html</**welcome-file**> </**welcome-file-list**> |

pinyougou-user-web/src/main/resources/springmvc.xml：

|  |
| --- |
| *<!-- 配置开启组件扫描 -->* <**context:component-scan base-package="com.pinyougou.user.controller"**/>  *<!-- ############## 配置dubbo服务消费者 ############## --> <!-- 配置当前应用的名称 -->* <**dubbo:application name="pinyougou-user-web"**/> *<!-- 配置zookeeper作为注册中心，注册服务地址 -->* <**dubbo:registry protocol="zookeeper" address="192.168.12.131:2181"**/> *<!-- 配置采用包扫描来引用服务，产生服务接口的代理对象 -->* <**dubbo:annotation package="com.pinyougou.user.controller"**/> |

## 5.3静态资源

拷贝【资料\用户中心静态资源】到pinyougou-user-web/src/main/webapp/目录：



## 5.4配置域名访问

修改hosts文件：127.0.0.1 user.pinyougou.com

修改Nginx的nginx.conf配置反向代理：

|  |
| --- |
| **server {**  **listen 80;**  **server\_name user.pinyougou.com;**  **proxy\_set\_header X-Forwarded-Host $host;**  **proxy\_set\_header X-Forwarded-Server $host;**  **location / {**  **proxy\_pass http://127.0.0.1:9107;**  **proxy\_connect\_timeout 600;**  **proxy\_read\_timeout 600;**  **}**  **}** |

# 6.用户中心【用户注册】

## 6.1需求分析

完成用户注册功能：



## 6.2基本注册功能实现

### 6.2.1前端代码

#### **6.2.1.1页面**

修改pinyougou-user-web/src/main/webapp/register.html，引入js：

|  |
| --- |
| <**script src="/js/vue.min.js"**></**script**> <**script src="/js/axios.min.js"**></**script**> <**script src="/js/controller/userController.js"**></**script**> |

register.html绑定表单：

|  |
| --- |
| <**form class="sui-form form-horizontal"**>  <**div class="control-group"**>  <**label class="control-label"**>用户名：</**label**>  <**div class="controls"**>  <**input type="text" placeholder="请输入你的用户名"  v-model="user.username"  class="input-xfat input-xlarge"**/>  </**div**>  </**div**>  <**div class="control-group"**>  <**label class="control-label"**>登录密码：</**label**>  <**div class="controls"**>  <**input type="password" placeholder="设置登录密码"  v-model="user.password"  class="input-xfat input-xlarge"**/>  </**div**>  </**div**>  <**div class="control-group"**>  <**label class="control-label"**>确认密码：</**label**>  <**div class="controls"**>  <**input type="password" placeholder="再次确认密码"  v-model="password"  class="input-xfat input-xlarge"**/>  </**div**>  </**div**>    <**div class="control-group"**>  <**label class="control-label"**>手机号：</**label**>  <**div class="controls"**>  <**input type="text" placeholder="请输入你的手机号"  v-model="user.phone"  class="input-xfat input-xlarge"**/>  </**div**>  </**div**>  <**div class="control-group"**>  <**label class="control-label"**>短信验证码：</**label**>  <**div class="controls"**>  <**input type="text" placeholder="短信验证码"  class="input-xfat input-xlarge"**/>  <**a href="#"**>获取短信验证码</**a**>  </**div**>  </**div**>    <**div class="control-group"**>  <**label class="control-label"**>**&nbsp;&nbsp;&nbsp;&nbsp;**</**label**>  <**div class="controls"**>  <**input name="m1" type="checkbox" value="2"**/>  <**span**>同意协议并注册《品优购用户协议》</**span**>  </**div**>  </**div**>  <**div class="control-group"**>  <**label class="control-label"**></**label**>  <**div class="controls btn-reg"**>  <**a class="sui-btn btn-block btn-xlarge btn-danger"  @click="save()"**>完成注册</**a**>  </**div**>  </**div**> </**form**> |

#### **6.2.1.2控制器**

pinyougou-user-web/src/main/webapp/js/controller/userController.js，在data与methods中定义：

|  |
| --- |
| *// 窗口加载完* ***window***.onload = **function** () {  **var** vue = **new Vue**({  **el** : **'#app'**, *// 元素绑定* **data** : { *// 数据模型* **user** : {}, *// 用户* **password** : **""** *// 确认密码* },  **methods** : { *// 操作方法* save : **function** () {  **if** (**this**.**user**.**password** != **this**.**password**){  *alert*(**"密码不一致，请重新输入！"**);  **return**;  }  **axios**.post(**"/user/save"**, **this**.**user**)  .then(**function**(response){  **if** (response.**data**){  *alert*(**"注册成功！"**);  vue.**user** = {};  vue.**password** = **""**;  }**else**{  *alert*(**"注册失败！"**);  }  });  }  },  created : **function** () { *// 创建生命周期* }  }); }; |

### 6.2.2后端代码

pinyougou-user-web\src\main\java\com\pinyougou\user\controller包下创建UserController.java

|  |
| --- |
| @RestController @RequestMapping(**"/user"**) **public class** UserController {  @Reference(timeout = 10000)  **private** UserService **userService**;  */\*\* 注册用户 \*/* @PostMapping(**"/save"**)  **public boolean** save(@RequestBody User user){  **try**{  **userService**.save(user);  **return true**;  }**catch** (Exception ex){  ex.printStackTrace();  }  **return false**;  } } |

pinyougou-user-service\src\main\java\com\pinyougou\user\service\impl包下,创建UserServiceImpl.java：

|  |
| --- |
| @Service(interfaceName = **"com.pinyougou.service.UserService"**) @Transactional **public class** UserServiceImpl **implements** UserService {  @Autowired  **private** UserMapper **userMapper**;  @Override  **public void** save(User user) {  **try** {  *// 创建日期* user.setCreated(**new** Date());  *// 修改日期* user.setUpdated(user.getCreated());  *// 密码加密* user.setPassword(DigestUtils.*md5Hex*(user.getPassword()));  **userMapper**.insertSelective(user);  }**catch** (Exception ex){  **throw new** RuntimeException(ex);  }  }  @Override  **public void** update(User user) {  }  @Override  **public void** delete(Serializable id) {  }  @Override  **public void** deleteAll(Serializable[] ids) {  }  @Override  **public** User findOne(Serializable id) {  **return null**;  }  @Override  **public** List<User> findAll() {  **return null**;  }  @Override  **public** List<User> findByPage(User user, **int** page, **int** rows) {  **return null**;  } } |

## 6.3注册判断短信验证码

### 6.3.1实现思路

点击页面上的”获取短信验证码”连接，向后端传递手机号。后端随机生成6位数字作为短信验证码，将其保存在redis中（手机号作为KEY），并发送到短信网关。

用户注册时，后端根据手机号查询redis中的验证码与用户填写的验证码是否相同，如果不同则提示用户不能注册。

### 6.3.2发送短信验证码

#### **6.3.2.1前端代码**

修改pinyougou-user-web/src/main/webapp/register.html(70行)：

|  |
| --- |
| <**a href="javascript:;" @click="sendCode();"**>获取短信验证码</**a**> |

pinyougou-user-web/src/main/webapp/js/controller/userController.js，在methods中，增加方法：

|  |
| --- |
| *// 发送短信验证码* sendCode : **function** () {  **if** (**this**.**user**.phone){  **axios**.get(**"/user/sendCode?phone="** + **this**.**user**.phone)  .then(**function**(response){  *alert*(response.**data** ? **"发送成功！"** : **"发送失败！"**);  });  }**else** {  *alert*(**"请输入手机号码！"**);  } } |

#### **6.3.2.2后端代码**

pinyougou-user-web/src/main/java/com/pinyougou/user/controller/UserController.java：

|  |
| --- |
| */\*\* 发送短信验证码 \*/* @GetMapping(**"/sendCode"**) **public boolean** sendCode(String phone){  **try** {  **if** (StringUtils.*isNoneBlank*(phone)) {  */\*\* 发送验证码 \*/* **userService**.sendCode(phone);  **return true**;  }  } **catch** (Exception e) {  e.printStackTrace();  }  **return false**; } |

pinyougou-interface\src\main\java\com\pinyougou\service\UserService.java：

|  |
| --- |
| */\*\* 发送短信验证码 \*/* **boolean** sendCode(String phone); |

pinyougou-user-service/src/main/java/com.pinyougou.user.service.impl/UserServiceImpl.java：

|  |
| --- |
| @Autowired **private** RedisTemplate<String,String> **redisTemplate**; @Value(**"${sms.url}"**) **private** String **smsUrl**; @Value(**"${sms.signName}"**) **private** String **signName**; @Value(**"${sms.templateCode}"**) **private** String **templateCode**; */\*\* 发送短信验证码 \*/* **public boolean** sendCode(String phone){  **try**{  */\*\* 生成6位随机数 \*/* String code = UUID.*randomUUID*().toString()  .replaceAll(**"-"**, **""**)  .replaceAll(**"[a-z|A-Z]"**,**""**)  .substring(0, 6);  System.***out***.println(**"验证码："** + code);  */\*\* 调用短信发送接口 \*/* HttpClientUtils httpClientUtils = **new** HttpClientUtils(**false**);  *// 创建Map集合封装请求参数* Map<String, String> param = **new** HashMap<>();  param.put(**"phone"**, phone);  param.put(**"signName"**, **signName**);  param.put(**"templateCode"**, **templateCode**);  param.put(**"templateParam"**, **"{\"number\":\""** + code + **"\"}"**);  *// 发送Post请求* String content = httpClientUtils.sendPost(**smsUrl**, param);  *// 把json字符串转化成Map* Map<String, Object> resMap = JSON.*parseObject*(content, Map.**class**);  */\*\* 存入Redis中(90秒) \*/* **redisTemplate**.boundValueOps(phone).set(code, 90, TimeUnit.***SECONDS***);  **return** (**boolean**)resMap.get(**"success"**);  }**catch** (Exception ex){  **throw new** RuntimeException(ex);  } } |

拷贝“资料\HttpClientUtils.java”到pinyougou-common\src\main\java\

com\pinyougou\common\util\包下面。

pinyougou-common/pom.xml加入依赖：

|  |
| --- |
| <**dependency**>  <**groupId**>org.apache.httpcomponents</**groupId**>  <**artifactId**>httpclient</**artifactId**> </**dependency**> |

pinyougou-user-service/src/main/resources/props/application.properties：

|  |
| --- |
| *# 接口请求地址* **sms.url**=**http://sms.pinyougou.com/sms/sendSms** *# 配置短信模版代号* **sms.templateCode**=**SMS\_11480310** *# 配置短信签名* **sms.signName**=**\u4e94\u5b50\u8fde\u73e0** |

### 6.3.3检验短信验证码

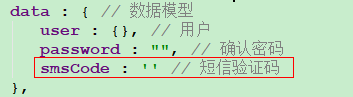
#### **6.3.3.1前端代码**

修改pinyougou-user-web/src/main/webapp/register.html页面，绑定变量：

|  |
| --- |
| <**input type="text" placeholder="短信验证码"  v-model="smsCode" class="input-xfat input-xlarge"/**> |

pinyougou-user-web/src/main/webapp/js/controller/userController.js：

* 在data中定义变量



* 修改methods中save方法

|  |
| --- |
| *// 用户注册* save : **function** () {  **if** (**this**.**user**.**password** != **this**.**password**){  *alert*(**"密码不一致，请重新输入！"**);  **return**;  }  **axios**.post(**"/user/save?smsCode="** + **this**.**smsCode**, **this**.**user**)  .then(**function**(response){  **if** (response.**data**){  *alert*(**"注册成功！"**);  vue.**user** = {};  vue.**password** = **""**;  vue.**smsCode** = **""**;  }**else**{  *alert*(**"注册失败！"**);  }  }); }, |

#### **6.3.3.2后端代码**

pinyougou-user-web/src/main/java/com/pinyougou/user/controller/UserController.java：

|  |
| --- |
| */\*\* 注册用户 \*/* @PostMapping(**"/save"**) **public boolean** save(@RequestBody User user, String smsCode){  **try** {  **boolean** ok = **userService**.checkSmsCode(user.getPhone(), smsCode);  **if**(ok){  **userService**.save(user);  **return true**;  }  } **catch** (Exception e) {  e.printStackTrace();  }  **return false**; } |

pinyougou-interface\src\main\java\com\pinyougou\service\UserService.java：

|  |
| --- |
| */\*\* 检查短信验证码是否正确 \*/* **boolean** checkSmsCode(String phone, String code); |

pinyougou-user-service\src\main\java\com\pinyougou\user\service\impl\UserServiceImpl.java：

|  |
| --- |
| */\*\* 检查短信验证码是否正确 \*/* **public boolean** checkSmsCode(String phone, String code){  */\*\* 获取Redis中存储的验证码 \*/* String sysCode = **redisTemplate**.boundValueOps(phone).get();  **return** StringUtils.*isNoneBlank*(sysCode) && sysCode.equals(code); } |