

第6章 移动端开发-体检预约

1. 移动端开发

1.1 移动端开发方式

随着移动互联网的兴起和手机的普及，目前移动端应用变得愈发重要，成为了各个商家的必争之地。例如，我们可以使用手机购物、支付、打车、玩游戏、订酒店、购票等，以前只能通过PC端完成的事情，现在通过手机都能够实现，而且更加方便，而这些都需要移动端开发进行支持，那如何进行移动端开发呢？

移动端开发主要有三种方式：

- 1、基于手机API开发（原生APP）
- 2、基于手机浏览器开发（移动web）
- 3、混合开发（混合APP）

1.1.1 基于手机API开发

手机端使用手机API，例如使用Android、ios 等进行开发，服务端只是一个数据提供者。手机端请求服务端获取数据（json、xml格式）并在界面进行展示。这种方式相当于传统开发中的C/S模式，即需要在手机上安装一个客户端软件。

这种方式需要针对不同的手机系统分别进行开发，目前主要有以下几个平台：

- 1、苹果ios系统版本，开发语言是Objective-C
- 2、安卓Android系统版本，开发语言是Java
- 3、微软Windows phone系统版本，开发语言是C#
- 4、塞班symbian系统版本，开发语言是C++

此种开发方式举例：手机淘宝、抖音、今日头条、大众点评

1.1.2 基于手机浏览器开发

生存在浏览器中的应用，基本上可以说是触屏版的网页应用。这种开发方式相当于传统开发中的B/S模式，也就是手机上不需要额外安装软件，直接基于手机上的浏览器进行访问。这就需要我们编写的html页面需要根据不同手机的尺寸进行自适应调节，目前比较流行的是html5开发。除了直接通过手机浏览器访问，还可以将页面内嵌到一些应用程序中，例如通过微信公众号访问html5页面。

这种开发方式不需要针对不同的手机系统分别进行开发，只需要开发一个版本，就可以在不同的手机上正常访问。

本项目会通过将我们开发的html5页面内嵌到微信公众号这种方式进行开发。

1.1.3 混合开发

是半原生半Web的混合类App。需要下载安装，看上去类似原生App，访问的内容是Web网页。其实就是把HTML5页面嵌入到一个原生容器里面。

1.2 微信公众号开发

要进行微信公众号开发，首先需要访问微信公众平台，官网：<https://mp.weixin.qq.com/>。

1.2.1 帐号分类

在微信公众平台可以看到，有四种帐号类型：服务号、订阅号、小程序、企业微信（原企业号）。

 <p>服务号</p> <p>给企业和组织提供更强大的业务服务与用户管理能力，帮助企业快速实现全新的公众号服务平台。</p>	 <p>订阅号</p> <p>为媒体和个人提供一种新的信息传播方式，构建与读者之间更好的沟通与管理模式。</p>	 <p>小程序</p> <p>一种新的开放能力，可以在微信内被便捷地获取和传播，同时具有出色的使用体验。</p>
 <p>企业微信 原企业号</p> <p>企业的专业办公管理工具。与微信一致的沟通体验，提供丰富免费的办公应用，并与微信消息、小程序、微信支付等互通，助力企业高效办公和管理。</p>		

帐号类型	功能介绍
订阅号	主要偏于为用户传达资讯（类似报纸杂志），认证前后都是每天只可以群发一条消息。（适用于个人和组织）
服务号	主要偏于服务交互（类似银行，114，提供服务查询），认证前后都是每个月可群发4条消息。（不适用于个人）
企业微信	企业微信是一个面向企业级市场的产品，是一个独立APP好用的基础办公沟通工具，拥有最基础和最实用的功能服务，专门提供给企业使用的IM产品。（适用于企业、政府、事业单位或其他组织）
小程序	是一种新的开放能力，开发者可以快速地开发一个小程序。小程序可以在微信内被便捷地获取和传播，同时具有出色的使用体验。
<p>温馨提示：</p> <p>1、如果想简单的发送消息，达到宣传效果，建议可选择订阅号；</p> <p>2、如果想用公众号获得更多的功能，例如开通微信支付，建议可以选择服务号；</p> <p>3、如果想用来管理内部企业员工、团队，对内使用，可申请企业微信；</p> <p>4、原企业号已升级为企业微信。</p>	

本项目会选择订阅号这种方式进行公众号开发。

1.2.2 注册帐号

要开发微信公众号，首先需要注册成为会员，然后就可以登录微信公众平台进行自定义菜单的设置。

注册页面：https://mp.weixin.qq.com/cgi-bin/registermidpage?action=index&lang=zh_CN&token=

请选择注册的帐号类型



订阅号

具有信息发布与传播的能力
适合个人及媒体注册



服务号

具有用户管理与提供业务服务的能力
适合企业及组织注册



小程序


具有出色的体验，可以被便捷地获取与传播
适合有服务内容企业和组织注册



企业微信
原企业号

具有实现企业内部沟通与协同管理的能力
适合企业客户注册

选择订阅号进行注册：

每个邮箱仅能申请一种帐号 

邮箱

激活邮箱

作为登录帐号，请填写未被微信公众平台注册，未被微信开放平台注册，未被个人微信号绑定的邮箱

邮箱验证码

激活邮箱后将收到验证邮件，请回填邮件中的6位验证码

密码

字母、数字或者英文符号，最短8位，区分大小写

确认密码

请再次输入密码

☐ 我同意并遵守《微信公众平台服务协议》

注册

输入邮箱、邮箱验证码、密码、确认密码等按照页面流程进行注册

1.2.3 自定义菜单

注册成功后就可以使用注册的邮箱和设置的密码进行登录，登录成功后点击左侧“自定义菜单”进入自定义菜单页面

在自定义菜单页面可以根据需求创建一级菜单和二级菜单，其中一级菜单最多可以创建3个，每个一级菜单下面最多可以创建5个二级菜单。每个菜单由菜单名称和菜单内容组成，其中菜单内容有3中形式：发送消息、跳转网页、跳转小程序。

1.2.4 上线要求

如果是个人用户身份注册的订阅号，则自定义菜单的菜单内容不能进行跳转网页，因为个人用户目前不支持微信认证，而跳转网页需要微信认证之后才有权限。

如果是企业用户，首先需要进行微信认证，通过后就可以进行跳转网页了，跳转网页的地址要求必须有域名并且域名需要备案通过。

2. 需求分析和环境搭建

2.1 需求分析

用户在体检之前需要进行预约，可以通过电话方式进行预约，此时会由体检中心客服人员通过后台系统录入预约信息。用户也可以通过手机端自助预约。本章节开发的功能为用户通过手机自助预约。

预约流程如下：

- 1、访问移动端首页
- 2、点击体检预约进入体检套餐列表页面
- 3、在体检套餐列表页面点击具体套餐进入套餐详情页面
- 4、在套餐详情页面点击立即预约进入预约页面
- 5、在预约页面录入体检人相关信息点击提交预约

效果如下图：



入职无忧体检套餐 (男女通用)

入职体检套餐

性别不限

18-60

粉红珍爱(女)升级TM12项筛查体检...

本套餐针对宫颈(TCT检查、HPV乳头瘤病毒筛查)、乳腺(彩超, 癌抗125), 甲状...

女

18-60

阳光爸妈升级肿瘤12项筛查 (男女通用)

本套餐主要针对常见肿瘤筛查, 肝肾、颈动脉、脑血栓、颅内血流筛查, 以及风湿、...

性别不限

55-100

孕前检查套餐 (女) -精英版

孕前检查套餐 (女) -精英版

女

18-50

珍爱高端升级肿瘤12项筛查 (男女通用)

本套餐是一款针对生化五项检查, 心, 肝, 胆, 胃, 甲状腺, 颈椎, 肺功能, 脑部检...

性别不限

14-20

套餐详情

项目名称	项目内容	项目解读
一般检查	身高 体重 体重指数 收缩压 舒张压	一般检查
视力色觉	裸视力 (右) 裸视力 (左) 矫正视力 (右) 矫正视力 (左) 色觉	视力色觉
血常规	白细胞计数 红细胞计数 血红蛋白 红细胞压积 平均红细胞体积 平均红细胞血红蛋白含量 平均红细胞血红蛋白浓度 红细胞分布宽度-变异系数 血小板计数 平均血小板体积 血小板分布宽度 淋巴细胞百分比 中间细胞百分比 中性粒细胞百分比 淋巴细胞绝对值 中间细胞绝对值 中性粒细胞绝对值 红细胞分布宽度-标准差 血小板压积	血常规
尿常规	尿比重 尿酸碱度 尿白蛋白 尿亚硝酸盐 尿蛋白质 尿糖 尿酮体 尿胆原 尿胆红素 尿隐血 尿镜检红细胞 尿镜检白细胞 上皮细胞 无机盐类 尿镜检蛋白定性	尿常规

立即预约



珍爱高端升级肿瘤12项筛查（男女单人）

本套餐是一款针对生化五项检查，心，肝，胆，胃，甲状腺，颈椎，肺功能，脑部检查（经颅多普勒）以及癌症筛查，适合大众人群体检的套餐。

女

20-70

?

预约须知

>

体检人信息

体检人

请输入姓名

性别

☒ 男 ☐ 女

手机号

请输入手机号

发送验证码

验证码

请输入验证码

身份证号

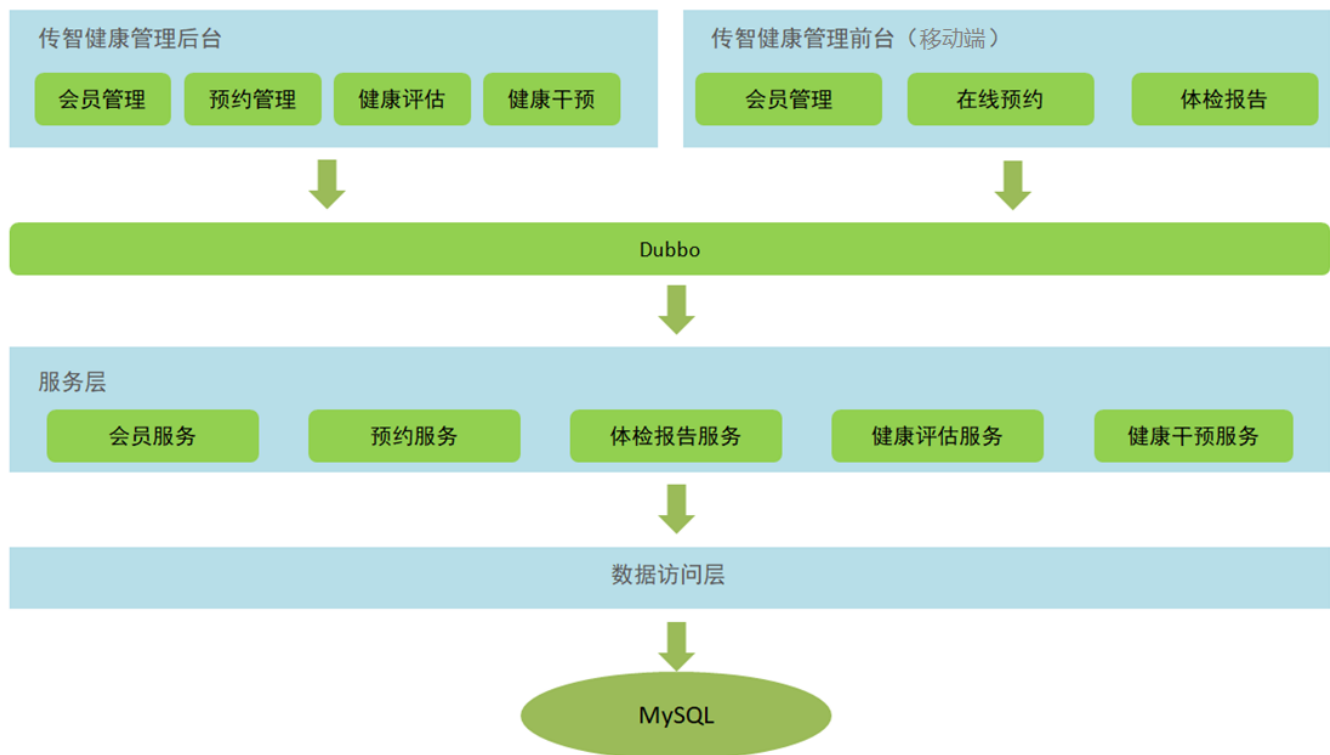
请输入身份证号

体检日期

提交预约

2.2 搭建移动端工程

本项目是基于SOA架构进行开发，前面我们已经完成了后台系统的部分功能开发，在后台系统中都是通过Dubbo调用服务层发布的服务进行相关的操作。本章节我们开发移动端工程也是同样的模式，所以我们也需要在移动端工程中通过Dubbo调用服务层发布的服务，如下图：



2.2.1 导入maven坐标

在health_common工程的pom.xml文件中导入阿里短信发送的maven坐标

```
<dependency>
  <groupId>com.aliyun</groupId>
  <artifactId>aliyun-java-sdk-core</artifactId>
  <version>3.3.1</version>
</dependency>
<dependency>
  <groupId>com.aliyun</groupId>
  <artifactId>aliyun-java-sdk-dysmsapi</artifactId>
  <version>1.0.0</version>
</dependency>
```

2.2.2 导入通用组件

在health_common工程中导入如下通用组件

ValidateCodeUtils工具类:

```
package com.itheima.utils;

import java.util.Random;

/**
 * 随机生成验证码工具类
 */
public class ValidateCodeUtils {
    /**
     * 随机生成验证码
     * @param length 长度为4位或者6位
     * @return
     */
    public static Integer generateValidateCode(int length){
        Integer code =null;
        if(length == 4){
            code = new Random().nextInt(9999);//生成随机数，最大为9999
            if(code < 1000){
                code = code + 1000;//保证随机数为4位数字
            }
        }else if(length == 6){
            code = new Random().nextInt(999999);//生成随机数，最大为999999
            if(code < 100000){
                code = code + 100000;//保证随机数为6位数字
            }
        }else{
            throw new RuntimeException("只能生成4位或6位数字验证码");
        }
        return code;
    }

    /**
     * 随机生成指定长度字符串验证码
     * @param length 长度
     * @return
     */
    public static String generateValidateCode4String(int length){
        Random rdm = new Random();
        String hash1 = Integer.toHexString(rdm.nextInt());

        String capstr = hash1.substring(0, length);
    }
}
```

```
        return capstr;  
    }  
}
```

SMSUtils工具类:

```

package com.itheima.utils;

import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.dysmsapi.model.v20170525.SendSmsRequest;
import com.aliyuncs.dysmsapi.model.v20170525.SendSmsResponse;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;

/**
 * 短信发送工具类
 */
public class SMSUtils {
    public static final String VALIDATE_CODE = "SMS_159620392";//发送短信
    验证码
    public static final String ORDER_NOTICE = "SMS_159771588";//体检预约成
    功通知

    /**
     * 发送短信
     * @param phoneNumbers
     * @param param
     * @throws ClientException
     */
    public static void sendShortMessage(String templateCode,String
    phoneNumbers,String param) throws ClientException{
        // 设置超时时间-可自行调整
        System.setProperty("sun.net.client.defaultConnectTimeout",
"10000");
        System.setProperty("sun.net.client.defaultReadTimeout", "10000");
        // 初始化ascClient需要的几个参数
        final String product = "Dysmsapi";// 短信API产品名称（短信产品名固
    定，无需修改）
        final String domain = "dysmsapi.aliyuncs.com";// 短信API产品域名
        （接口地址固定，无需修改）
        // 替换成你的AK

        final String accessKeyId = "accessKeyId";// 你的accessKeyId,参考本

```

文档步骤2

```
final String accessKeySecret = "accessKeySecret";// 你的
accessKeySecret, 参考本文档步骤2
// 初始化ascClient,暂时不支持多region (请勿修改)
IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou",
accessKeyId, accessKeySecret);
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", product,
domain);
IAcsClient acsClient = new DefaultAcsClient(profile);
// 组装请求对象
SendSmsRequest request = new SendSmsRequest();
// 使用post提交
request.setMethod(MethodType.POST);
// 必填:待发送手机号。支持以逗号分隔的形式进行批量调用,批量上限为1000
个手机号码,批量调用相对于单条调用及时性稍有延迟,验证码类型的短信推荐使用单条调用的
方式
request.setPhoneNumbers(phoneNumbers);
// 必填:短信签名-可在短信控制台中找到
request.setSignName("传智健康");
// 必填:短信模板-可在短信控制台中找到
request.setTemplateCode(templateCode);
// 可选: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
// 友情提示:如果JSON中需要带换行符,请参照标准的JSON协议对换行符的要求,
比如短信内容中包含\r\n的情况在JSON中需要表示成\\r\\n,否则会导致JSON在服务端解析
失败
request.setTemplateParam("{\"code\":\""+param+"\"}");
// 可选-上行短信扩展码(扩展码字段控制在7位或以下,无特殊需求用户请忽略
此字段)
// request.setSmsUpExtendCode("90997");
// 可选:outId为提供给业务方扩展字段,最终在短信回执消息中将此值带回给调
用者
// request.setOutId("yourOutId");
// 请求失败这里会抛ClientException异常
SendSmsResponse sendSmsResponse =
acsClient.getAcsResponse(request);
if (sendSmsResponse.getCode() != null &&
sendSmsResponse.getCode().equals("OK")) {
    // 请求成功
    System.out.println("请求成功");
}
```

```
}  
}
```

RedisMessageConstant常量类:

```
package com.itheima.constant;  
  
public class RedisMessageConstant {  
    public static final String SENDTYPE_ORDER = "001";//用于缓存体检预约时  
    发送的验证码  
    public static final String SENDTYPE_LOGIN = "002";//用于缓存手机号快速  
    登录时发送的验证码  
    public static final String SENDTYPE_GETPWD = "003";//用于缓存找回密码时  
    发送的验证码  
}
```

2.2.3 health_mobile

创建移动端工程health_mobile，打包方式为war，用于存放Controller，在Controller中通过Dubbo可以远程访问服务层相关服务，所以需要依赖health_interface接口工程。

pom.xml:

```

<?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>health_parent</artifactId>
    <groupId>com.itheima</groupId>
    <version>1.0-SNAPSHOT</version>
  </parent>
  <modelVersion>4.0.0</modelVersion>
  <artifactId>health_mobile</artifactId>
  <packaging>war</packaging>
  <name>healthmobile_web Maven Webapp</name>
  <url>http://www.example.com</url>
  <properties>
    <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
  <dependencies>
    <dependency>
      <groupId>com.itheima</groupId>
      <artifactId>health_interface</artifactId>
      <version>1.0-SNAPSHOT</version>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.tomcat.maven</groupId>
        <artifactId>tomcat7-maven-plugin</artifactId>
        <configuration>
          <!-- 指定端口 -->
          <port>80</port>
          <!-- 请求路径 -->

          <path></path>

```



```
        </configuration>
    </plugin>
</plugins>
</build>
</project>
```

静态资源（CSS、html、img等，详见资料）：



web.xml:

```

<!DOCTYPE web-app PUBLIC
"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/web-app_2_3.dtd" >
<web-app>
  <display-name>Archetype Created Web Application</display-name>
  <!-- 解决post乱码 -->
  <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>
    <init-param>
      <param-name>forceEncoding</param-name>
      <param-value>true</param-value>
    </init-param>
  </filter>
  <filter-mapping>
    <filter-name>CharacterEncodingFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <servlet>
    <servlet-name>springmvc</servlet-name>
    <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <!-- 指定加载的配置文件 ， 通过参数contextConfigLocation加载 -->
    <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>springmvc</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>

  <welcome-file-list>

```

```
    <welcome-file>/pages/index.html</welcome-file>  
  </welcome-file-list>  
</web-app>
```

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:p="http://www.springframework.org/schema/p"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    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">

    <mvc:annotation-driven>
        <mvc:message-converters register-defaults="true">
            <bean
class="com.alibaba.fastjson.support.spring.FastJsonHttpMessageConverter">
                <property name="supportedMediaTypes" value="application/json"/>
                <property name="features">
                    <list>
                        <value>WriteMapNullValue</value>
                        <value>WriteDateUseDateFormat</value>
                    </list>
                </property>
            </bean>
        </mvc:message-converters>
    </mvc:annotation-driven>
    <!-- 指定应用名称 -->
    <dubbo:application name="health_mobile" />
    <!--指定服务注册中心地址-->
    <dubbo:registry address="zookeeper://127.0.0.1:2181"/>
    <!--批量扫描-->
    <dubbo:annotation package="com.itheima.controller" />
    <!--
        超时全局设置 10分钟

        check=false 不检查服务提供方，开发阶段建议设置为false
    -->

```

`check=true` 启动时检查服务提供方，如果服务提供方没有启动则报错

```
-->  
<dubbo:consumer timeout="600000" check="false"/>  
<import resource="spring-redis.xml"></import>  
</beans>
```

spring-redis.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:p="http://www.springframework.org/schema/p"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    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">

    <context:property-placeholder location="classpath:redis.properties"
/>

    <!--Jedis连接池的相关配置-->
    <bean id="jedisPoolConfig"
class="redis.clients.jedis.JedisPoolConfig">
        <property name="maxTotal">
            <value>${redis.pool.maxActive}</value>
        </property>
        <property name="maxIdle">
            <value>${redis.pool.maxIdle}</value>
        </property>
        <property name="testOnBorrow" value="true"/>
        <property name="testOnReturn" value="true"/>
    </bean>

    <bean id="jedisPool" class="redis.clients.jedis.JedisPool">
        <constructor-arg name="poolConfig" ref="jedisPoolConfig" />
        <constructor-arg name="host" value="${redis.host}" />
        <constructor-arg name="port" value="${redis.port}" type="int" />
        <constructor-arg name="timeout" value="${redis.timeout}"
type="int" />

    </bean>
```

</beans>

redis.properties:

```
#最大分配的对象数
redis.pool.maxActive=200
#最大能够保持idle状态的对象数
redis.pool.maxIdle=50
redis.pool.minIdle=10
redis.pool.maxWaitMillis=20000
#当池内没有返回对象时，最大等待时间
redis.pool.maxWait=300

#格式: redis://:[密码]@[服务器地址]:[端口]/[db index]
#redis.uri = redis://:12345@127.0.0.1:6379/0

redis.host = 127.0.0.1
redis.port = 6379
redis.timeout = 30000
```

log4j.properties:

```
### direct log messages to stdout ###
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target=System.err
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d{ABSOLUTE} %5p %c{1}:%L
- %m%n

### direct messages to file mylog.log ###
log4j.appender.file=org.apache.log4j.FileAppender
log4j.appender.file.File=c:\\mylog.log
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern=%d{ABSOLUTE} %5p %c{1}:%L -
%m%n

### set log levels - for more verbose logging change 'info' to 'debug'
###

log4j.rootLogger=info, stdout
```

3. 套餐列表页面动态展示

移动端首页为/pages/index.html，效果如下：



点击体检预约直接跳转到体检套餐列表页面（/pages/setmeal.html）

3.1 完善页面

3.1.1 展示套餐信息


```

<ul class="list">
  <li class="list-item" v-for="setmeal in setmealList">
    <a class="link-page" :href="'setmeal_detail.html?id='+setmeal.id">
      
      <div class="item-body">
        <h4 class="ellipsis item-title">{{setmeal.name}}</h4>
        <p class="ellipsis-more item-desc">{{setmeal.remark}}</p>
        <p class="item-keywords">
          <span>{{setmeal.sex == '0' ? '性别不限' : setmeal.sex == '1' ?
'男':'女'}}</span>
          <span>{{setmeal.age}}</span>
        </p>
      </div>
    </a>
  </li>
</ul>

```

3.1.2 获取套餐列表数据

```

var vue = new Vue({
  el: '#app',
  data: {
    setmealList: []
  },
  mounted () {
    //发送ajax请求，获取所有的套餐数据，赋值给setmealList模型数据，用于页面
    展示
    axios.get("/setmeal/getAllSetmeal.do").then((res) => {
      if(res.data.flag){
        //查询成功，给模型数据赋值
        this.setmealList = res.data.data;
      }else{
        //查询失败，弹出提示信息
        this.$message.error(res.data.message);
      }
    });
  }
});

```

3.2 后台代码

3.2.1 Controller

在health_mobile工程中创建SetmealController并提供getSetmeal方法，在此方法中通过Dubbo远程调用套餐服务获取套餐列表数据

```
package com.itheima.controller;

import com.alibaba.dubbo.config.annotation.Reference;
import com.itheima.constant.MessageConstant;
import com.itheima.entity.Result;
import com.itheima.pojo.Setmeal;
import com.itheima.service.SetmealService;
import org.springframework.web.bind.annotation.*;
import java.util.List;

@RestController
@RequestMapping("/setmeal")
public class SetmealController {
    @Reference//(check = false)
    private SetmealService setmealService;

    //获取所有套餐信息
    @RequestMapping("/getSetmeal")
    public Result getSetmeal(){
        try{
            List<Setmeal> list = setmealService.findAll();
            return new Result(true,
MessageConstant.GET_SETMEAL_LIST_SUCCESS,list);
        }catch (Exception e){
            e.printStackTrace();
            return new
Result(true,MessageConstant.GET_SETMEAL_LIST_FAIL);
        }
    }
}
```

3.2.2 服务接口

在SetmealService服务接口中扩展findAll方法

```
public List<Setmeal> findAll();
```

3.2.3 服务实现类

在SetmealServiceImpl服务实现类中实现findAll方法

```
public List<Setmeal> findAll() {  
    return setmealDao.findAll();  
}
```

3.2.4 Dao接口

在SetmealDao接口中扩展findAll方法

```
public List<Setmeal> findAll();
```

3.2.5 Mapper映射文件

在SetmealDao.xml映射文件中扩展SQL语句

```
<select id="findAll" resultType="com.itheima.pojo.Setmeal">  
    select * from t_setmeal  
</select>
```

4. 套餐详情页面动态展示

前面我们已经完成了体检套餐列表页面动态展示，点击其中任意一个套餐则跳转到对应的套餐详情页面（/pages/setmeal_detail.html），并且会携带此套餐的id作为参数提交。

请求路径格式：http://localhost/pages/setmeal_detail.html?id=10

在套餐详情页面需要展示当前套餐的信息（包括图片、套餐名称、套餐介绍、适用性别、适用年龄）、此套餐包含的检查组信息、检查组包含的检查项信息等。

4.1 完善页面

4.1.1 获取请求参数中套餐id

在页面中已经引入了healthmobile.js文件，此文件中已经封装了getUrlParam方法可以根据URL请求路径中的参数名获取对应的值

```
function getUrlParam(paraName) {  
    var url = document.location.toString();  
    //alert(url);  
    var arrObj = url.split("?");  
    if (arrObj.length > 1) {  
        var arrPara = arrObj[1].split("&");  
        var arr;  
        for (var i = 0; i < arrPara.length; i++) {  
            arr = arrPara[i].split("=");  
            if (arr != null && arr[0] == paraName) {  
                return arr[1];  
            }  
        }  
        return "";  
    }  
    else {  
        return "";  
    }  
}
```

在setmeal_detail.html中调用上面定义的方法获取套餐id的值

```
<script>  
    var id = getUrlParam("id");  
</script>
```

4.1.2 获取套餐详细信息

```
<script>
    var vue = new Vue({
        el: '#app',
        data: {
            imgUrl: null, //套餐对应的图片链接
            setmeal: {}
        },
        mounted() {
            axios.post("/setmeal/findById.do?id=" + id).then((response)
=> {
                if(response.data.flag){
                    this.setmeal = response.data.data;
                    this.imgUrl = 'http://pqjroc654.bkt.clouddn.com/' +
this.setmeal.img;
                }
            });
        }
    });
</script>
```

4.1.3 展示套餐信息

```

<div class="contentBox">
  <div class="card">
    <div class="project-img">
      
    </div>
    <div class="project-text">
      <h4 class="tit">{{setmeal.name}}</h4>
      <p class="subtit">{{setmeal.remark}}</p>
      <p class="keywords">
        <span>{{setmeal.sex == '0' ? '性别不限' : setmeal.sex == '1' ?
'男':'女'}}</span>
        <span>{{setmeal.age}}</span>
      </p>
    </div>
  </div>
  <div class="table-listbox">
    <div class="box-title">
      <i class="icon-zhen"><span class="path1"></span><span
class="path2"></span></i>
      <span>套餐详情</span>
    </div>
    <div class="box-table">
      <div class="table-title">
        <div class="tit-item flex2">项目名称</div>
        <div class="tit-item flex3">项目内容</div>
        <div class="tit-item flex3">项目解读</div>
      </div>
      <div class="table-content">
        <ul class="table-list">
          <li class="table-item" v-for="checkgroup in
setmeal.checkGroups">
            <div class="item flex2">{{checkgroup.name}}</div>
            <div class="item flex3">
              <label v-for="checkitem in checkgroup.checkItems">
                {{checkitem.name}}
              </label>
            </div>
            <div class="item flex3">{{checkgroup.remark}}</div>
          </li>
        </ul>
      </div>
    </div>
  </div>

```

```
        </div>
        <div class="box-button">
            <a @click="toOrderInfo()" class="order-btn">立即预约</a>
        </div>
    </div>
</div>
</div>
```

4.2 后台代码

4.2.1 Controller

在SetmealController中提供findById方法

```
//根据id查询套餐信息
@RequestMapping("/findById")
public Result findById(int id){
    try{
        Setmeal setmeal = setmealService.findById(id);
        return new
Result(true,MessageConstant.QUERY_SETMEAL_SUCCESS,setmeal);
    }catch (Exception e){
        e.printStackTrace();
        return new Result(false,MessageConstant.QUERY_SETMEAL_FAIL);
    }
}
```

4.2.2 服务接口

在SetmealService服务接口中提供findById方法

```
public Setmeal findById(int id);
```

4.2.3 服务实现类

在SetmealServiceImpl服务实现类中实现findById方法

```
public Setmeal findById(int id) {  
    return setmealDao.findById(id);  
}
```

4.2.4 Dao接口

在SetmealDao接口中提供findById方法

```
public Setmeal findById(int id);
```

4.2.5 Mapper映射文件

此处会使用mybatis提供的关联查询，在根据id查询套餐时，同时将此套餐包含的检查组都查询出来，并且将检查组包含的检查项都查询出来。

SetmealDao.xml文件：


```

<resultMap type="com.itheima.pojo.Setmeal" id="baseResultMap">
    <id column="id" property="id"/>
    <result column="name" property="name"/>
    <result column="code" property="code"/>
    <result column="helpCode" property="helpCode"/>
    <result column="sex" property="sex"/>
    <result column="age" property="age"/>
    <result column="price" property="price"/>
    <result column="remark" property="remark"/>
    <result column="attention" property="attention"/>
    <result column="img" property="img"/>
</resultMap>
<resultMap type="com.itheima.pojo.Setmeal"
    id="findByIdResultMap"
    extends="baseResultMap">
    <collection property="checkGroups"
        javaType="ArrayList"
        ofType="com.itheima.pojo.CheckGroup"
        column="id"
        select="com.itheima.dao.CheckGroupDao.findCheckGroupById">
    </collection>
</resultMap>
<select id="findById" resultMap="findByIdResultMap">
    select * from t_setmeal where id=#{id}
</select>

```

CheckGroupDao.xml文件：

```

<resultMap type="com.itheima.pojo.CheckGroup" id="baseResultMap">
  <id column="id" property="id"/>
  <result column="name" property="name"/>
  <result column="code" property="code"/>
  <result column="helpCode" property="helpCode"/>
  <result column="sex" property="sex"/>
  <result column="remark" property="remark"/>
  <result column="attention" property="attention"/>
</resultMap>
<resultMap type="com.itheima.pojo.CheckGroup"
  id="findByIdResultMap"
  extends="baseResultMap">
  <collection property="checkItems"
    javaType="ArrayList"
    ofType="com.itheima.pojo.CheckItem"
    column="id"
    select="com.itheima.dao.CheckItemDao.findCheckItemById">
  </collection>
</resultMap>
<!--根据套餐id查询检查项信息-->
<select id="findCheckGroupById" resultMap="findByIdResultMap">
  select * from t_checkgroup
  where id
  in (select checkgroup_id from t_setmeal_checkgroup where setmeal_id=#
{id})
</select>

```

CheckItemDao.xml文件:

```

<!--根据检查组id查询检查项信息-->
<select id="findCheckItemById" resultType="com.itheima.pojo.CheckItem">
  select * from t_checkitem
  where id
  in (select checkitem_id from t_checkgroup_checkitem where
checkgroup_id=#{id})
</select>

```

5. 短信发送

5.1 短信服务介绍

目前市面上有很多第三方提供的短信服务，这些第三方短信服务会和各个运营商（移动、联通、电信）对接，我们只需要注册成为会员并且按照提供的开发文档进行调用就可以发送短信。需要说明的是这些短信服务都是收费的服务。

本项目短信发送我们选择的是阿里云提供的短信服务。

短信服务（Short Message Service）是阿里云为用户提供的一种通信服务的能力，支持快速发送短信验证码、短信通知等。三网合一专属通道，与工信部携号转网平台实时互联。电信级运维保障，实时监控自动切换，到达率高达99%。短信服务API提供短信发送、发送状态查询、短信批量发送等能力，在短信服务控制台上添加签名、模板并通过审核之后，可以调用短信服务API完成短信发送等操作。

5.2 注册阿里云账号

阿里云官网：<https://www.aliyun.com/>

点击官网首页免费注册跳转到如下注册页面：

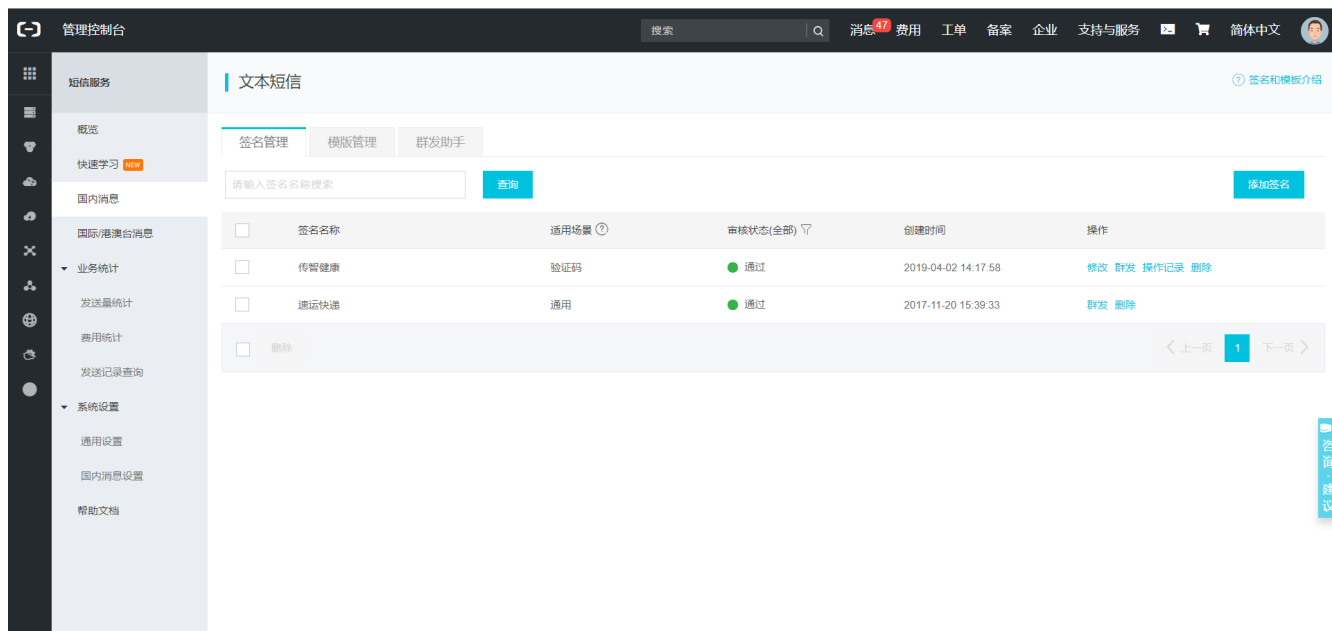
欢迎注册阿里云

设置会员名	
设置你的登录密码	
请再次输入你的密码	
+86	请输入手机号码
>>	请按住滑块，拖动到最右边
同意条款并注册	

☐ 《阿里云网站服务条款》 | 《法律声明和隐私权政策》

5.3 设置短信签名

注册成功后，点击登录按钮进行登录。登录后进入短信服务管理页面，选择国内消息菜单：



点击添加签名按钮:

添加签名

返回上层

针对网站、APP、小程序或公众号未上线的情况，平台只支持发送验证码；如已上线，可申请“通用”的适用场景，可发送验证码、短信通知。

* 签名:

长度限2 - 12个字符，建议为用户真实应用名/网站名/公司名

0/12

• 若签名 / 模版内容侵犯到第三方权益必须获得第三方真实授权

• 无须添加【】、()、[]符号，签名发送会自带【】符号，避免重复

• 了解更多[签名/模板申请规范](#)

* 适用场景:

验证码

通用 ?

• 个人用户可申请1个验证码签名，通用场景签名数量不限

申请说明:

请描述您的业务使用场景

0/200

确定

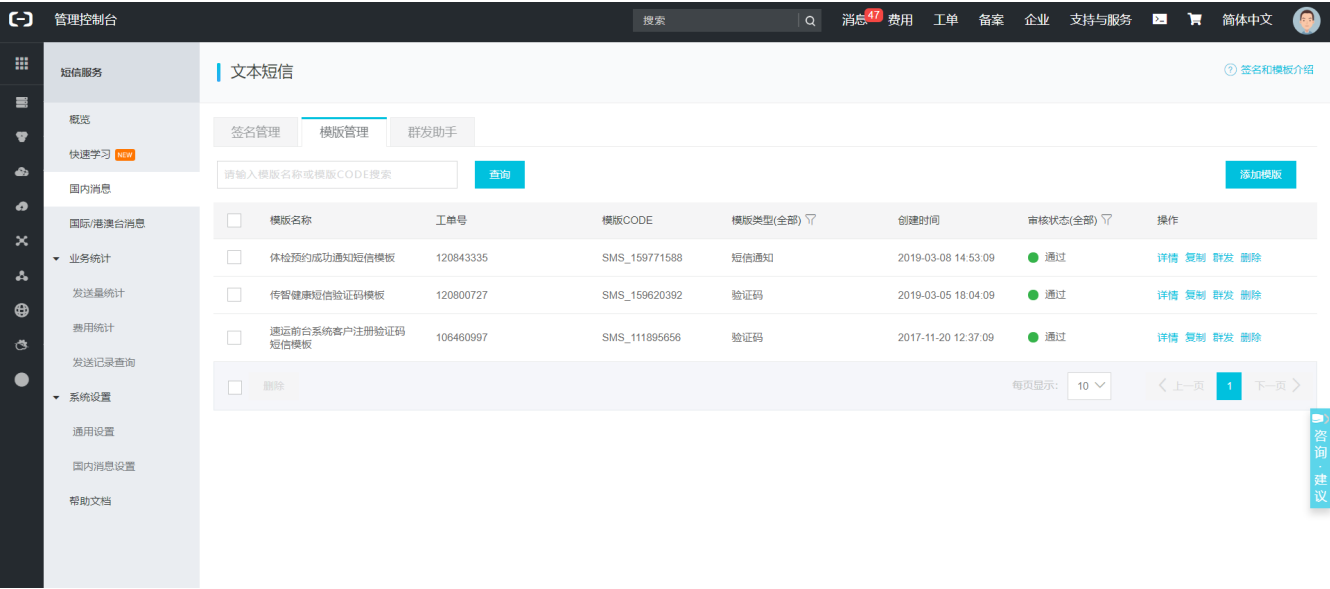
• 预计两小时完成审核

• 审核工作时间: 周一至周五9:00-23:00（法定节日顺延）

目前个人用户只能申请适用场景为验证码的签名

5.4 设置短信模板

在国内消息菜单页面中，点击模板管理标签页：



点击添加模板按钮：

- * 模版类型: ☒ 验证码 (0.045元/条)
- ☐ 短信通知 (0.045元/条)
- ☐ 推广短信 (0.055元/条) [升级为企业后启用](#)

* 模版名称: 0/30

* 模版内容:

0/500

想快速获得可用模版, 可使用[常用模版库](#)

- 验证码模板只支持验证码作为变量; 变量替换值<=6位数字或字母
- 不能发送营销/贷款/借款/中奖/抽奖类短信, 不支持金融理财&房产通知类短信(验证码除外)
- 签名/模版申请规范 https://help.aliyun.com/document_detail/55324.html

* 申请说明:

0/100

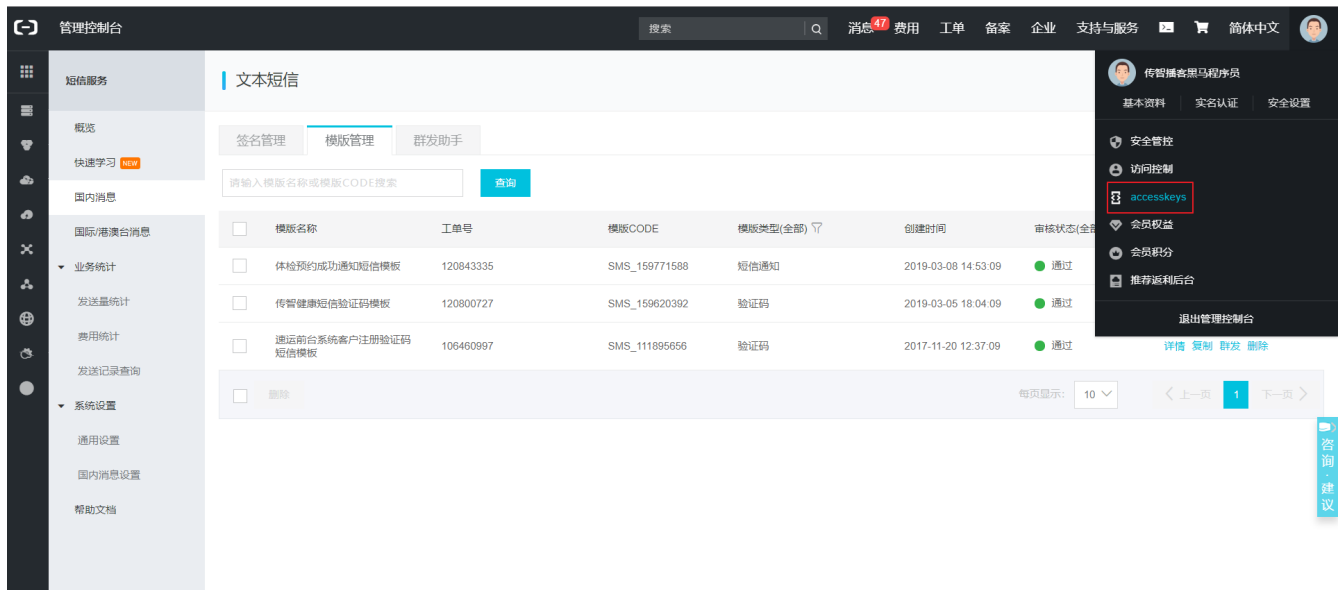
提交

模版预览

- 预计两小时完成审核
- 审核工作时间: 周一至周日9:00-23:00 (法定节日顺延)

5.5 设置access keys

在发送短信时需要进行身份认证, 只有认证通过才能发送短信。本小节就是要设置用于发送短信时进行身份认证的key和密钥。鼠标放在页面右上角当前用户头像上, 会出现下拉菜单:



点击accesskeys:

安全提示



提示信息云账号AccessKey是您访问阿里云API的密钥，具有该账户完全的权限，请您务必妥善保管！不要通过任何方式(eg, Github)将AccessKey公开到外部渠道，以避免被他人利用而造成 **安全威胁**。强烈建议您遵循 [阿里云安全最佳实践](#)，使用RAM子用户AccessKey来进行API调用。

继续使用AccessKey

开始使用子用户AccessKey

点击开始使用子用户AccessKey按钮:



STEP 1: 填写用户名

STEP 2: 选择权限

STEP 3: 创建用户及AK

SMS

系统 AliyunDysmsFullAccess

管理短信服务(SMS)的权限

系统 AliyunDysmsReadOnlyAccess

只读访问短信服务(SMS)的权限

系统 AliyunDysmsFullAccess

管理短信服务(SMS)的权限

系统 AliyunDysmsReadOnlyAccess

只读访问短信服务(SMS)的权限

上一步

开始创建

快速创建

手机验证

✕

✕

您绑定的手机: 185****9943 ([更换手机](#))

* 校验码:

57秒后重发

确定

取消

上一步

开始创建

快速创建子用户AccessKey

STEP 1: 填写用户名

STEP 2: 选择权限

STEP 3: 创建用户及AK

这是AccessKey可供下载的唯一机会，请及时保存！

用户 itcast 创建成功

您已成功新建用户和AccessKey，并且添加完成授权。

AccessKey详情

AccessKeyID:
LTAI5khuks10U9GE

AccessKeySecret:
8SIwTHwEvpvpFQKEDtBT8dp7XaFM8

查看用户详情

下载AccessKey

创建成功，其中AccessKeyID为访问短信服务时使用的ID，AccessKeySecret为密钥。

可以在用户详情页面下禁用刚刚创建的AccessKey：

管理控制台

搜索

消息 47 费用 工单 备案 企业 支持与服务

简体中文

用户详情

用户授权策略

用户加入的组

基本信息

编辑基本信息

用户名 itcast

UID 295373255482832878

创建时间 2019-04-17 14:33:52

显示名

手机

邮箱

备注 -

Web控制台登录管理

启用控制台登录

必须开启多因素认证 关闭

上次登录时间:

下次登录必须重置密码: 关闭

多因素认证设备

操作

类型

简介

启用状态

虚拟MFA设备

遵循TOTP标准算法来产生6位数字验证码的应用程序

未启用

启用虚拟MFA设备

用户AccessKey

创建AccessKey

AccessKey ID

状态

创建时间

操作

LTAI5khuks10U9GE

启用

2019-04-17 14:33:53

禁用

可以设置每日和每月短信发送上限：



由于短信服务是收费服务，所以还需要进行充值才能发送短信：



5.6 发送短信

5.6.1 导入maven坐标

```
<dependency>
  <groupId>com.aliyun</groupId>
  <artifactId>aliyun-java-sdk-core</artifactId>
  <version>3.3.1</version>
</dependency>
<dependency>
  <groupId>com.aliyun</groupId>
  <artifactId>aliyun-java-sdk-dysmsapi</artifactId>
  <version>1.0.0</version>
</dependency>
```

5.6.2 封装工具类

```

package com.itheima.utils;

import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.dysmsapi.model.v20170525.SendSmsRequest;
import com.aliyuncs.dysmsapi.model.v20170525.SendSmsResponse;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;

/**
 * 短信发送工具类
 */
public class SMSUtils {
    public static final String VALIDATE_CODE = "SMS_159620392";//发送短信
    验证码
    public static final String ORDER_NOTICE = "SMS_159771588";//体检预约成
    功通知

    /**
     * 发送短信
     * @param phoneNumbers
     * @param param
     * @throws ClientException
     */
    public static void sendShortMessage(String templateCode,String
    phoneNumbers,String param) throws ClientException{
        // 设置超时时间-可自行调整
        System.setProperty("sun.net.client.defaultConnectTimeout",
"10000");
        System.setProperty("sun.net.client.defaultReadTimeout", "10000");
        // 初始化ascClient需要的几个参数
        final String product = "Dysmsapi";// 短信API产品名称（短信产品名固
    定，无需修改）
        final String domain = "dysmsapi.aliyuncs.com";// 短信API产品域名
        （接口地址固定，无需修改）
        // 替换成你的AK

        final String accessKeyId = "accessKeyId";// 你的accessKeyId,参考本

```

文档步骤2

```
final String accessKeySecret = "accessKeySecret";// 你的
accessKeySecret, 参考本文档步骤2
// 初始化ascClient,暂时不支持多region (请勿修改)
IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou",
accessKeyId, accessKeySecret);
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", product,
domain);
IAcsClient acsClient = new DefaultAcsClient(profile);
// 组装请求对象
SendSmsRequest request = new SendSmsRequest();
// 使用post提交
request.setMethod(MethodType.POST);
// 必填:待发送手机号。支持以逗号分隔的形式进行批量调用,批量上限为1000
个手机号码,批量调用相对于单条调用及时性稍有延迟,验证码类型的短信推荐使用单条调用的
方式
request.setPhoneNumbers(phoneNumbers);
// 必填:短信签名-可在短信控制台中找到
request.setSignName("传智健康");
// 必填:短信模板-可在短信控制台中找到
request.setTemplateCode(templateCode);
// 可选: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
// 友情提示:如果JSON中需要带换行符,请参照标准的JSON协议对换行符的要求,
比如短信内容中包含\r\n的情况在JSON中需要表示成\\r\\n,否则会导致JSON在服务端解析
失败
request.setTemplateParam("{\"code\":\""+param+"\"}");
// 可选-上行短信扩展码(扩展码字段控制在7位或以下,无特殊需求用户请忽略
此字段)
// request.setSmsUpExtendCode("90997");
// 可选:outId为提供给业务方扩展字段,最终在短信回执消息中将此值带回给调
用者
// request.setOutId("yourOutId");
// 请求失败这里会抛ClientException异常
SendSmsResponse sendSmsResponse =
acsClient.getAcsResponse(request);
if (sendSmsResponse.getStatusCode() != null &&
sendSmsResponse.getStatusCode().equals("OK")) {
    // 请求成功
    System.out.println("请求成功");
}
```

```
}  
}
```

5.6.3 测试短信发送

```
public static void main(String[] args)throws Exception {  
    SMSUtils.sendShortMessage("SMS_159620392","13812345678","1234");  
}
```