# 尚马教育 JAVA 高级课程

# 短信与邮件

文档编号：C12

创建日期： 2017-07-07

最后修改日期：2021-02-26

版 本 号：V3.5

电子版文件名：尚马教育-第三阶段-12.短信与邮件.docx

**文档修改记录：**

| 更新日期 | 更新作者 | 更新说明 | 版本号 |
|---|---|---|---|
| 2017-07-30 | 张元林 | 初始版本 | V1.0 |
| 2018-08-01 | 王绍成 | Mybatis 版本更新 | V2.0 |
| 2019-08-09 | 冯勇涛 | 课件格式以及课程深度加深 | V3.0 |
| 2021-01-26 | 冯勇涛 | 修改阿里云短信 | V3.5 |

目录

# 1. Sms

## 1.1. 使用场景

Java 发送短信的功能是通过其他设备来实现的，同样是需要付费。

在早期，比较流行使用**短信猫**来发送短信，短信猫相当于一个简单的手机，可以插入 SIM 卡，通过 Java 串口编程来调用短信猫的 AT 指令，再由短信猫来发送短信。但是受限于设备的性能，需要大量发送短信的时候，要购买更好的设备来支持，成本较高。

近年来互联网上出现了许多短信发送平台，速度更快，收费更低，成本上远远优于自己搭建的短信平台，而且更加稳定。

市场上短信发送的平台有很多，每家公司会根据自身的资源与不同的短信平台合作，但是从程序角度出发，所有短信平台的使用流程大体相同。

当前文档以阿里云的短信平台为例，介绍一下如何使用 Java 发送短信。

## 1.2. 短信猫

短信猫，其实是一种用来收发短信的设备，他和我们用的手机一样，需要手机 SIM 卡的支持，在需要收发短信的时候，在短信猫里面插入一张我们平时用的手机卡，插上电源，通过（USB 或者串口、网口）数据线和电脑相连，在电脑的应用管理软件中就可以实现短信收发的功能。

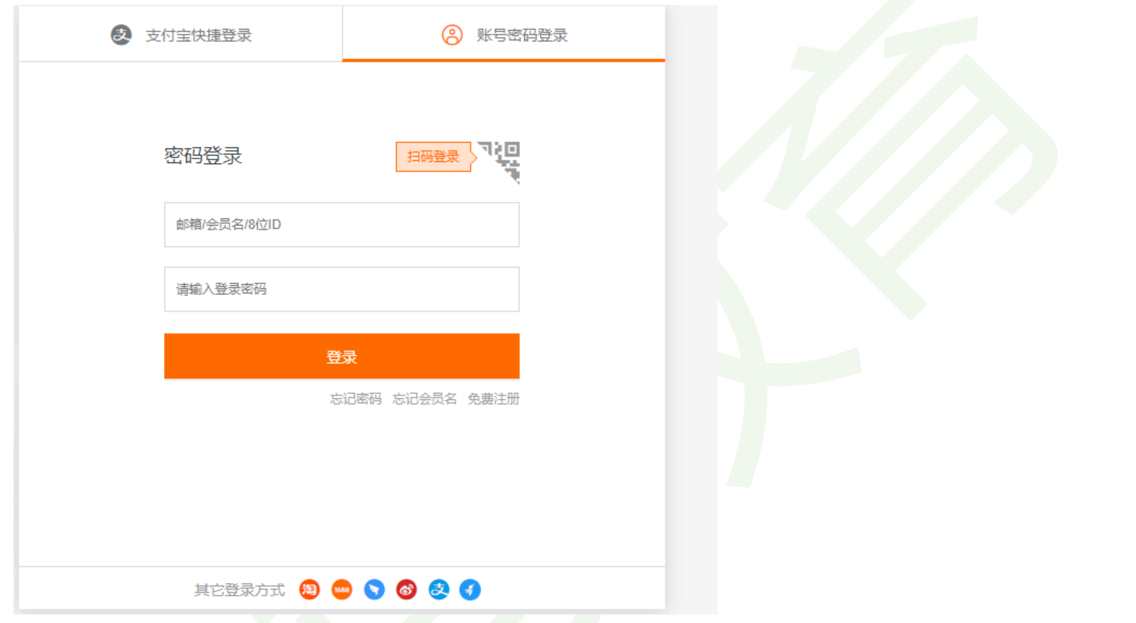
## 1.3. 云短信服务

这一步一般应该由运维人员搞定，但是多知道一些是没有坏处的。

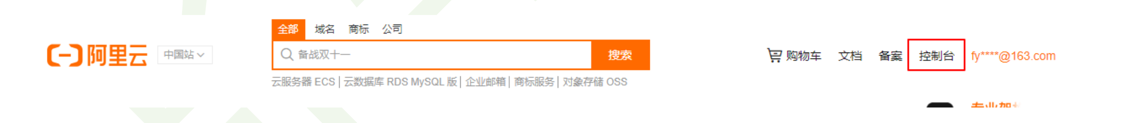
阿里云的官网地址:https://www.aliyun.com

如果是为公司申请，最好让其他人员申请一个公司账户，或者老板的账户，建议公司账户升级为企业用户，而不是使用个人账户，不然离职之后，很麻烦。

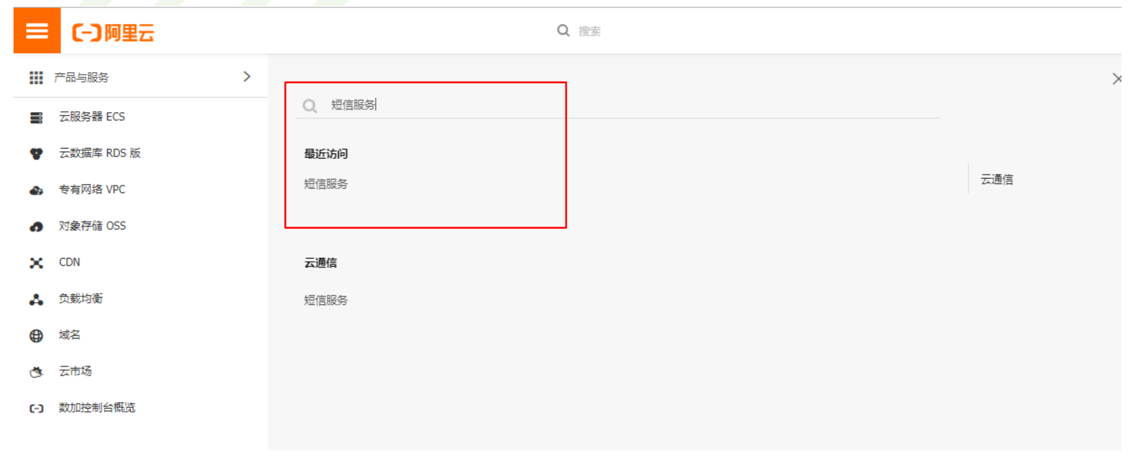
个人学习，可以直接使用淘宝登录。

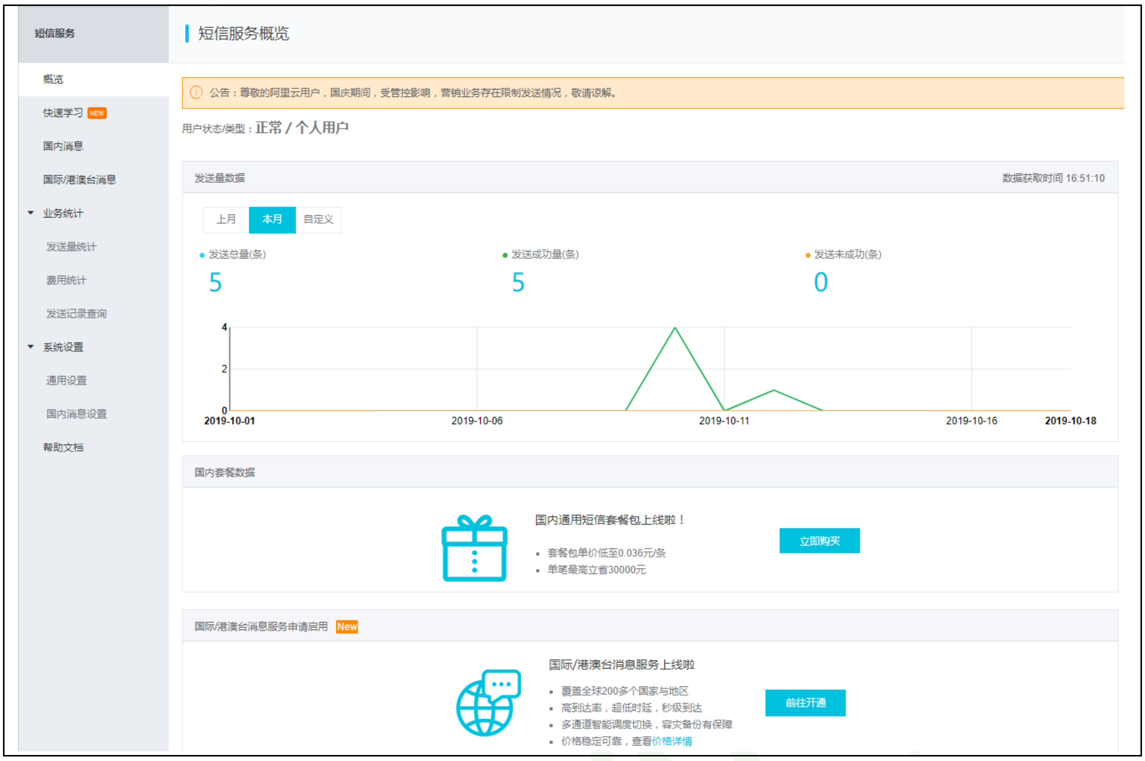


登录成功后，点击首页右上角的控制台



搜索短信服务





短信服务概览

公告：尊敬的阿里云用户，国庆期间，受管控影响，营销业务存在限制发送情况，敬请谅解。

用户状态/类型：正常 / 个人用户

发送量数据 数据获取时间 16:51:10

| 发送总量(条) | 发送成功量(条) | 发送未成功(条) |
|---|---|---|
| 5 | 5 | 0 |

国内套餐数据

国内通用短信套餐包上线啦！

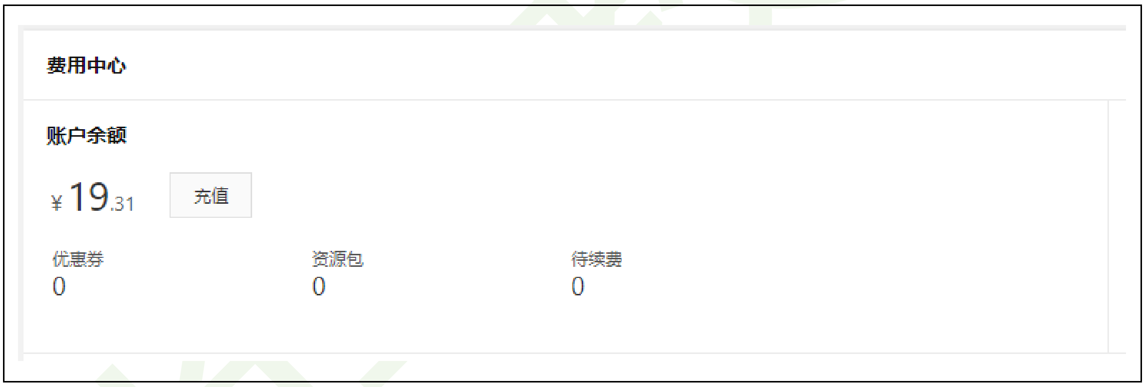
- 套餐包单价低至0.036元/条
- 单笔最高立省30000元

国际/港澳台消息服务申请启用

国际/港澳台消息服务上线啦

- 覆盖全球200多个国家与地区
- 高到达率，超低时延，秒级到达
- 多通道智能调度切换，容灾备份有保障
- 价格稳定可靠，查看价格详情

使用短信平台之前应该先充值，保证账户里的余额可以发送短信。



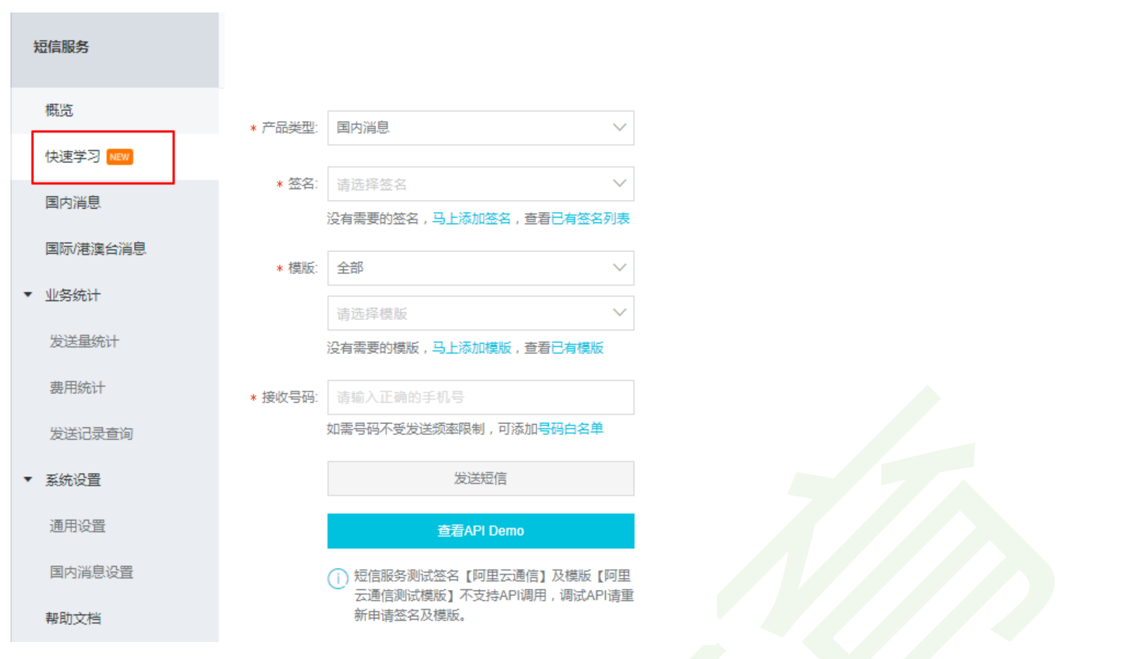
费用中心

账户余额

¥ 19.31

| 优惠券 | 资源包 | 待续费 |
|---|---|---|
| 0 | 0 | 0 |

快速学习



* 产品类型: 国内消息

* 签名: 请选择签名

没有需要的签名，马上添加签名，查看已有签名列表

* 模版: 全部

请选择模版

没有需要的模版，马上添加模版，查看已有模版

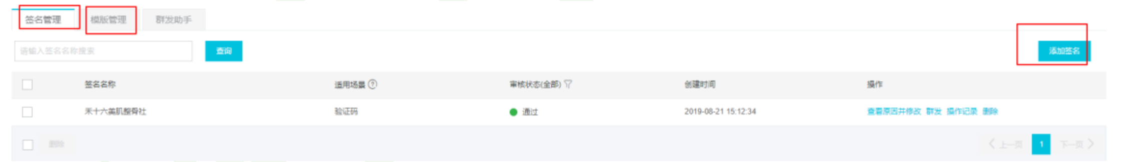
* 接收号码: 请输入正确的手机号

如需号码不受发送频率限制，可添加号码白名单

短信服务测试签名【阿里云通信】及模版【阿里云通信测试模版】不支持API调用，调试API请重新申请签名及模版。

## 1.3.1. 短信 API

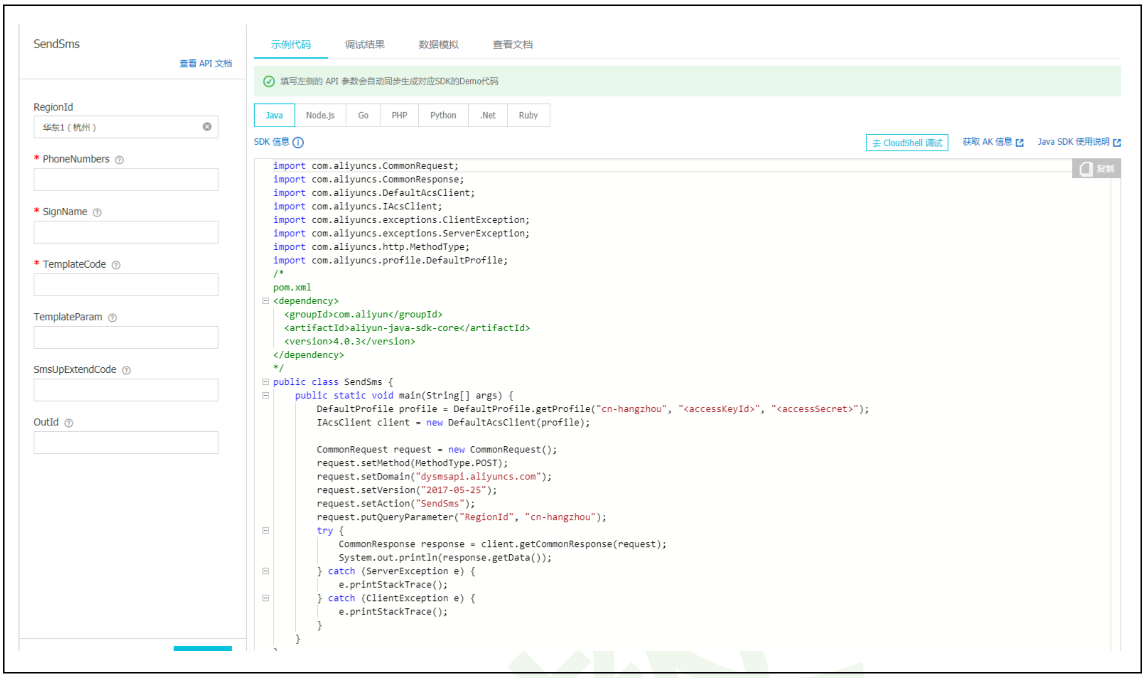
### 1.3.1.1. 添加签名与模板



| □ | 签名名称 | 适用场景 | 审核状态(全部) | 创建时间 | 操作 |
|---|---|---|---|---|---|
| □ | 禾十六美肌服务社 | 验证码 | ● 通过 | 2019-08-21 15:12:34 | 查看原因并修改 群发 操作记录 删除 |

签名和模板都搞定之后，就可以进一步研究如何使用 Java 程序调用

### 1.3.1.2. 查看 api



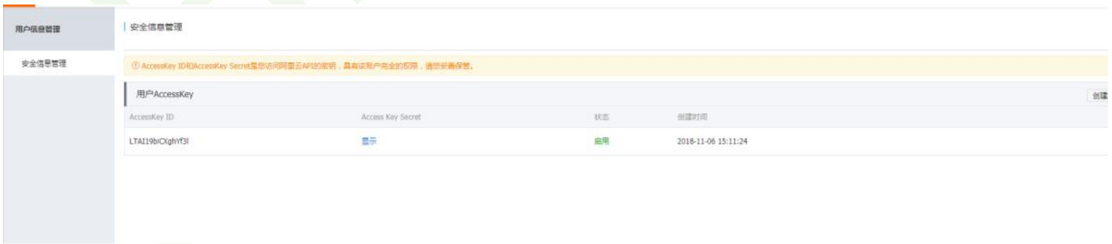
填写左侧的 API 参数会自动同步生成对应SDK的Demo代码

```java
import com.aliyuncs.CommonRequest;
import com.aliyuncs.CommonResponse;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.profile.DefaultProfile;
/*
pom.xml
<dependency>
  <groupId>com.aliyun</groupId>
  <artifactId>aliyun-java-sdk-core</artifactId>
  <version>4.0.3</version>
</dependency>
*/
public class SendSms {
    public static void main(String[] args) {
        DefaultProfile profile = DefaultProfile.getProfile("cn-hangzhou", "<accessKeyId>", "<accessSecret>");
        IAcsClient client = new DefaultAcsClient(profile);

        CommonRequest request = new CommonRequest();
        request.setMethod(MethodType.POST);
        request.setDomain("dysmsapi.aliyuncs.com");
        request.setVersion("2017-05-25");
        request.setAction("SendSms");
        request.putQueryParameter("RegionId", "cn-hangzhou");
        try {
            CommonResponse response = client.getCommonResponse(request);
            System.out.println(response.getData());
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
```

### 1.3.1.3. 获取授权码

什么是 AccessKey？网站中有详细的介绍，简单点说，就是程序调用的一个用户名密码，所以必须保密，其他人知道这个 AccessKey 之后，就可以调用你的个人用户了，使用你的模板发送短信。



安全信息管理

AccessKey ID和AccessKey Secret是您访问阿里云API的密钥，具有该账户完全的权限，请您妥善保管。

用户AccessKey

| AccessKey ID | Access Key Secret | 状态 | 创建时间 |
|---|---|---|---|
| LTAI19biCXghYf3l | 显示 | 启用 | 2018-11-06 15:11:24 |

记住这个 AccessKeyID 和 AccessKeySecret 一会写代码的时候会用到

### 1.3.1.4. 添加 maven 依赖

```xml
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-core</artifactId>
<version>4.5.3</version>
```

```
</dependency>
```

### 1.3.1.5. 代码示例

```
public static void send(String phone,String code) {
    DefaultProfile profile = DefaultProfile.getProfile("cn-hangzhou", "待填写", "待填写");
        IAcsClient client = new DefaultAcsClient(profile);

        CommonRequest request = new CommonRequest();
        request.setMethod(MethodType.POST);
        request.setDomain("dysmsapi.aliyuncs.com");
        request.setVersion("2017-05-25");
        request.setAction("SendSms");
        request.putQueryParameter("RegionId", "cn-hangzhou");
        request.putQueryParameter("PhoneNumbers",phone);
        request.putQueryParameter("SignName", "禾十六美肌整骨社");
        request.putQueryParameter("TemplateCode", "SMS_172738690");
        request.putQueryParameter("TemplateParam", "{\"code\":"+code+"}");
        try {
            CommonResponse response = client.getCommonResponse(request);
            System.out.println(response.getData());
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
  }
```

## 1.3.2. 查询发送信息

```
public static void main(String[] args) {
    DefaultProfile profile = DefaultProfile.getProfile("cn-hangzhou", "待填写", "待填写");
       IAcsClient client = new DefaultAcsClient(profile);

       CommonRequest request = new CommonRequest();
       request.setMethod(MethodType.POST);
       request.setDomain("dysmsapi.aliyuncs.com");
       request.setVersion("2017-05-25");
       request.setAction("QuerySendDetails");
       request.putQueryParameter("RegionId", "cn-hangzhou");
       request.putQueryParameter("PhoneNumber", "13767599447");
       request.putQueryParameter("SendDate", "20191010");
       request.putQueryParameter("PageSize", "10");
       request.putQueryParameter("CurrentPage", "1");
       try {
           CommonResponse response = client.getCommonResponse(request);
           System.out.println(response.getData());
       } catch (ServerException e) {
           e.printStackTrace();
       } catch (ClientException e) {
           e.printStackTrace();
       }
  }
```

# 2. 邮件发送

发送协议：SMTP

接收协议：POP3

## 2.1. 代码示例

```
String account="XXXXX@163.com";

    String auth="abc123456789";


    //SMTP 协议内容。

    Properties properties = new Properties();

    properties.put("mail.transport.protocol", "smtp");// 连接协议

     properties.put("mail.smtp.host", "smtp.163.com");// 主机名 smtp.163.com

     properties.put("mail.smtp.port", 465);// 默认端口号 25

     properties.put("mail.smtp.auth", "true");//服务端认证。

     properties.put("mail.smtp.ssl.enable", "true");// 设置是否使用 ssl 安全连接 ---一般都使用

     properties.put("mail.debug", "true");// 设置是否显示 debug 信息 true 会在控制台显示相关信息

     Session session = Session.getDefaultInstance(properties);//与邮件服务器的会话对象

    try {

        Address[] to = new Address[]{new InternetAddress("XXXX@qq.com")};//接收人

        Message msg = new MimeMessage(session);//主体，接收人，内容，附件。。。

        msg.setFrom(new InternetAddress(account));//发送人

        msg.setRecipients(RecipientType.TO, to);//接收人()

        msg.setSubject("测试邮件");//标题
        msg.setText("邮件内容");//内容


        Transport transport = session.getTransport();//邮差对象

        transport.connect(account, auth);


        transport.sendMessage(msg, msg.getAllRecipients());

    } catch (Exception e) {

        e.printStackTrace();

    }
```

## 2.2. 封装工具类

详见:邮件工具类 文件夹

MailSender：

```
package com.javasm.util.email;


import java.io.File;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import java.util.Properties;


import javax.activation.DataHandler;

import javax.activation.DataSource;

import javax.activation.FileDataSource;

import javax.mail.Message;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeBodyPart;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;
import javax.mail.internet.MimeUtility;



public class MailSender
{
    /** 默认 Content Type -> text/plain */
    public static final String DEFAULT_CONTENT_TYPE   = "text/plain";
    /** 默认字符编码 -> UTF-8 */
    public static final String DEFAULT_ENCODING        = GeneralHelper.DEFAULT_ENCODING;
    /** 默认 SMTP 端口 -> 25 */
    public static final int DEFAULT_PORT          = 25;

    private String host              = "";
    private String from              = "";
    private String user              = "";
    private String password          = "";
    private String subject       = "";
    private String text              = "";
    private String contentType = DEFAULT_CONTENT_TYPE;
    private String charset       = DEFAULT_ENCODING;
    private int port             = DEFAULT_PORT;
    private boolean auth         = true;
    private boolean needReceipt      = false;
    private Date sentDate        = null;
private List<String>         to                 = new ArrayList<String>();

private List<String>         cc                 = new ArrayList<String>();

private List<String>         bcc                = new ArrayList<String>();

private List<String>         replyTo            = new ArrayList<String>();

private List<String>         fileAcc       = new ArrayList<String>();

private List<MimeBodyPart>       byteAcc        = new ArrayList<MimeBodyPart>();


public int getPort()
{
    return port;
}


public void setPort(int port)
{
    this.port = port;
}


public boolean isAuth()
{
    return auth;
}


public void setAuth(boolean auth)
{
    this.auth = auth;
}


public String getCharset()
```

```
{
    return charset;
}

public void setCharset(String charset)
{
    this.charset = charset;
}

public String getContentType()
{
    return contentType;
}

public void setContentType(String contentType)
{
    this.contentType = contentType;
}

public boolean isNeedReceipt()
{
    return needReceipt;
}

public void setNeedReceipt(boolean needReceipt)
{
    this.needReceipt = needReceipt;
}
```

```
public String getFrom()
{
    return from;
}

public void setFrom(String from)
{
    this.from = from;
}

public String getHost()
{
    return host;
}

public void setHost(String host)
{
    this.host = host;
}

public String getPassword()
{
    return password;
}

public void setPassword(String password)
{
    this.password = password;
}
```

```
public String getSubject()
{
    return subject;
}

public void setSubject(String subject)
{
    this.subject = subject;
}

public String getText()
{
    return text;
}

public void setText(String text)
{
    this.text = text;
}

public Date getSentDate()
{
    return sentDate;
}

public void setSentDate(Date sentDate)
{
    this.sentDate = sentDate;
```

```
}

public String getUser()
{
    return user;
}

public void setUser(String user)
{
    this.user = user;
}

public List<String> getFileAcc()
{
    return fileAcc;
}

public List<MimeBodyPart> getByteAcc()
{
    return byteAcc;
}

public void setFileAcc(List<String> accessory)
{
    this.fileAcc = accessory;
}

public void setByteAcc(List<MimeBodyPart> accessory)
{
    this.byteAcc = accessory;
}

public List<String> getReplyTo()
{
    return replyTo;
}

public List<String> getTo()
{
    return to;
}

public void setTo(List<String> to)
{
    this.to = to;
}

public List<String> getCc()
{
    return cc;
}

public void setCc(List<String> cc)
{
    this.cc = cc;
}

public List<String> getBcc()
```

```
{
    return bcc;
}


public void setBcc(List<String> bcc)
{
    this.bcc = bcc;
}


public void addFileAcc(String accessory)
{
    fileAcc.add(accessory);
}


/**  添加  byte array  形式的附件  */
public void addByteAcc(byte[] accessory, String type, String fileName) throws Exception
{
    ByteArrayDataSource ds  = new ByteArrayDataSource(accessory, type, fileName);

    //BASE64Encoder enc = new BASE64Encoder();
    //fileName = "=?GBK?B?" + enc.encode(fileName.getBytes()) + "?=";
    fileName = MimeUtility.encodeText(fileName, charset, "B");

    MimeBodyPart mimeFile = new MimeBodyPart();
    mimeFile.setDataHandler(new DataHandler(ds));
    //mimeFile.setFileName(ds.getName());
    mimeFile.setFileName(fileName);

    byteAcc.add(mimeFile);
```

```
}

public void addReplyTo(String address)
{
    replyTo.add(address);
}

public void addTo(String address)
{
    to.add(address);
}

public void addCc(String address)
{
    cc.add(address);
}

public void addBcc(String address)
{
    bcc.add(address);
}

/** 发送邮件 */
public void send() throws Exception
{
    Transport transport = null;

    try
    {
```

```
Properties props = new Properties();

props.put("mail.transport.protocol", "smtp");

props.put("mail.smtp.auth", Boolean.toString(auth));

Session session = Session.getDefaultInstance(props, null);

MimeMessage msg = new MimeMessage(session);

msg.setFrom(new InternetAddress(from));


for(String i : to)

    msg.addRecipient(Message.RecipientType.TO, new InternetAddress(i));

for(String i : cc)

    msg.addRecipient(Message.RecipientType.CC, new InternetAddress(i));

for(String i : bcc)

    msg.addRecipient(Message.RecipientType.BCC, new InternetAddress(i));


if(replyTo.size() > 0)

{

    InternetAddress[] replyAddress = new InternetAddress[replyTo.size()];


    for(int i = 0; i < replyAddress.length; i++)

        replyAddress[i] = new InternetAddress((String)replyTo.get(i));


    msg.setReplyTo(replyAddress);

}


if(needReceipt)

    msg.addHeader("Disposition-Notification-To", from);


if(sentDate != null)

    msg.setSentDate(sentDate);
```

```
else
        msg.setSentDate(new Date());

msg.setSubject(subject, charset);

MimeMultipart mm        = new MimeMultipart();
MimeBodyPart mbText   = new MimeBodyPart();
mbText.setContent(text, contentType + ";charset=" + charset);
mm.addBodyPart(mbText);

for(String filePath : fileAcc)
{
        String fileName = (new File(filePath)).getName();
        fileName = MimeUtility.encodeText(fileName, charset, "B");

        MimeBodyPart mbFile           = new MimeBodyPart();
        DataSource datasource   = new FileDataSource(filePath);

        mbFile.setDataHandler(new DataHandler(datasource));
        mbFile.setFileName(fileName);
        mm.addBodyPart(mbFile);
}

for(MimeBodyPart part : byteAcc)
        mm.addBodyPart(part);

msg.setContent(mm);
msg.saveChanges();
```

```
                transport = session.getTransport();

                transport.connect(host, port, user, password);

                transport.sendMessage(msg, msg.getAllRecipients());

        }

        finally

        {

                if(transport != null) try{ transport.close(); } catch (Exception e) { }

        }

    }

}
```