# 开发

1、理解邮件开发涉及的基本概念

2、了解SMTP和POP3协议

3、掌握配置Foxmail

4、掌握JavaMail发送简单邮件

# 基本概念

1、JavaMail提供给开发者处理电子邮件相关的编程接口，它是Sun发布的用来处理email的API，它可以方便的执行一些常用的邮件传输，我们可以基于JavaMail开发出类似于Microsoft Outlook的应用程序。

2、电子邮箱：需要在邮件服务器上进行申请，确切的说，电子邮箱其实就是用户在邮件服务器上申请的一个账户，用户在邮件服务器上申请了一个账号后，邮件服务器就会为这个账号分配一定的空间，用户从而可以使用这个账号以及空间，发送电子邮件和保存别人发送过来的电子邮件。

3、邮件服务器：指的是一台电脑安装了邮件服务器软件，那么这台电脑称为是邮件服务器。要在Internet上提供电子邮件的功能，必须有专门的电子邮件服务器。这些服务器类似于现实生活中的邮局，它主要负责接收用户投递过来的邮件，并把邮件投递到邮件接收者的电子邮箱。

# 邮件收发协议

## SMTP协议（发送邮件）

SMTP：简单邮件传输协议，它是一组用于由源地址到目的地址传送邮件的规则，由它来控制信件的中转方式。SMTP协议属于TCP/IP协议族，它帮助每台计算机在发送或中转信件时找到下一个目的地。

SMTP服务器则是遵循SMTP协议的发送邮件服务器，用来发送或中转发出的电子邮件。通常把处理用户的SMTP请求（邮件发送请求）的邮件服务器称之为SMTP服务器，默认端口号是25。

## POP3协议（接收邮件）

POP3全称为“Post Office Protocol version 3”即邮局协议版本3，本协议主要用于支持使用客户端远程管理在服务器上的电子邮件。同样用户想从邮件服务器管理的电子邮件中接收一封电子邮件的话，它连上邮件服务器后，也需要遵循一定的通讯格式，POP3协议用于定义这种通讯格式。因而我们也把处理用户POP3请求（邮件接收请求）的邮件服务器称为POP3服务器，默认端口号是110。

## IMAP协议（获取邮件协议）

IMAP全称是Internet Mail Access Protocol，即Internet邮件访问协议，它是跟POP3类似邮件访问标准协议之一。不同的是，开启了IMAP后，您在电子邮件客户端收取的邮件仍然保留在服务器上，同时在客户端上的操作都会反馈到服务器上。如：删除邮件，标记已读等，服务器上的邮件也会做相应的动作。所以无论从浏览器登录邮箱或者客户端软件登录邮箱，看到的邮件以及状态都是一致的。默认的端口号是143。

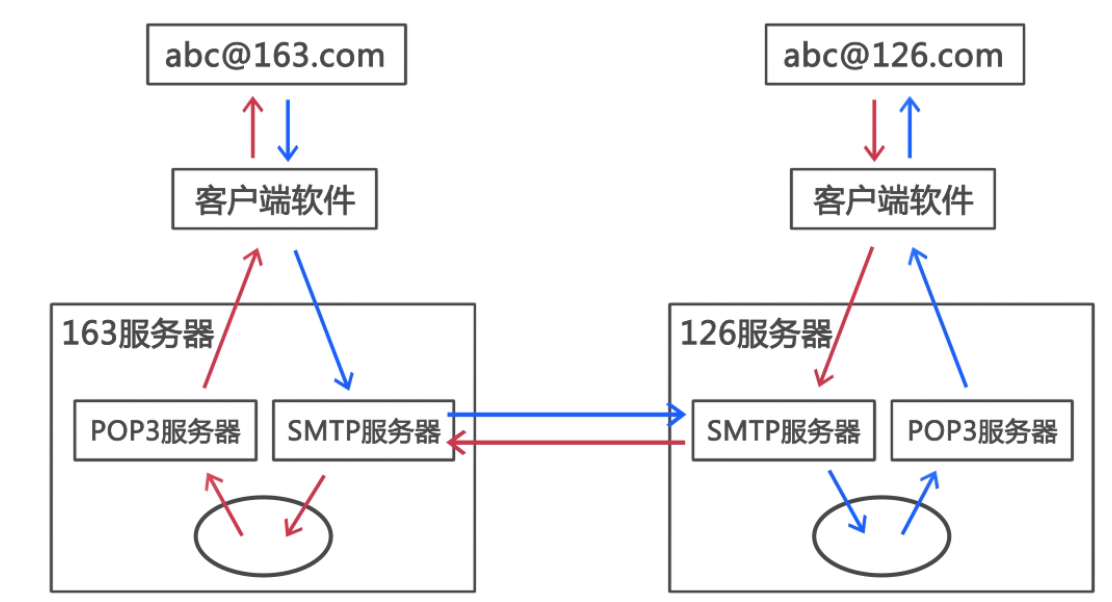
### IMAP与POP3的区别

[POP3](http://help.163.com/09/1223/14/5R7P6CJ600753VB8.html?servCode=6010376)协议允许电子邮件客户端下载服务器上的邮件，但是在客户端的操作（如移动邮件、标记已读等），不会反馈到服务器上，比如通过客户端收取了邮箱中的3封邮件并移动到其他文件夹，邮箱服务器上的这些邮件是没有同时被移动的 。而[IMAP](http://help.163.com/09/1223/14/5R7P6CJ600753VB8.html?servCode=6010376)提供webmail 与电子邮件客户端之间的双向通信，客户端的操作都会反馈到服务器上，对邮件进行的操作，服务器上的邮件也会做相应的动作。同时，**IMAP**像**POP3**那样提供了方便的邮件下载服务，让用户能进行离线阅读。**IMAP**提供的摘要浏览功能可以让你在阅读完所有的邮件到达时间、主题、发件人、大小等信息后才作出是否下载的决定。此外，**IMAP** 更好地支持了从多个不同设备中随时访问新邮件。



总之，IMAP 整体上为用户带来更为便捷和可靠的体验。POP3 更易丢失邮件或多次下载相同的邮件，但 IMAP 通过邮件客户端与webmail 之间的双向同步功能很好地避免了这些问题。注：若在web邮箱中设置了“保存到已发送”，使用客户端POP服务发信时，已发邮件也会自动同步到网页端“已发送”文件夹内。

## 邮件收发过程



# 邮箱服务器的安装和配置

下载易邮服务器

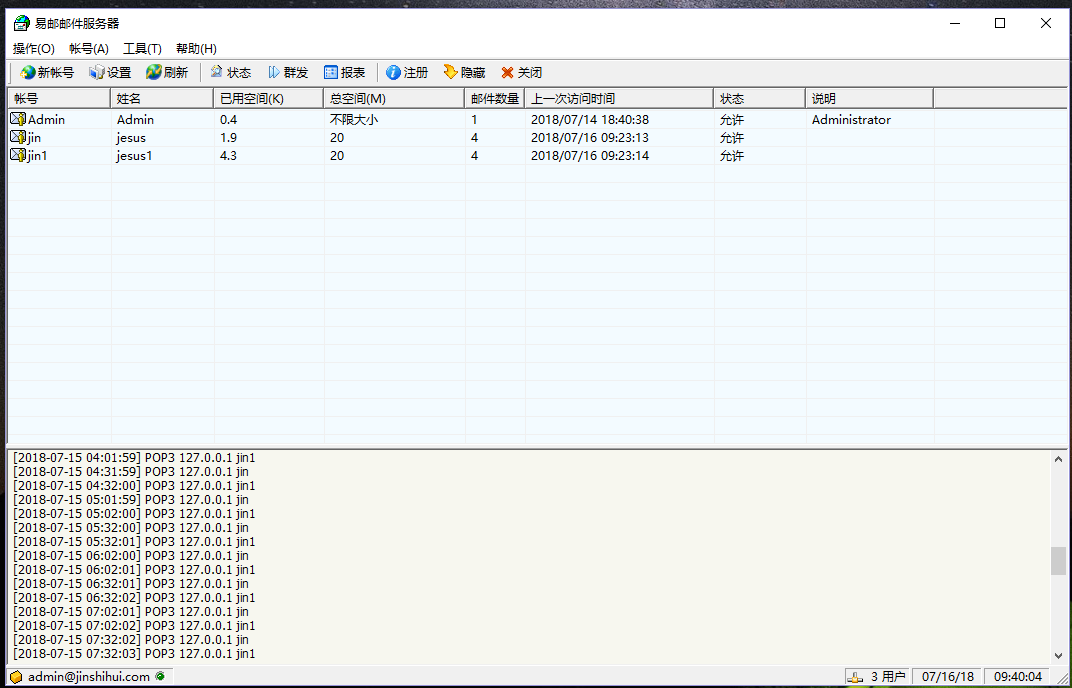
eyoumailserversetup.exe

操作如下：

首先需要添加账户，点击账号—新建账号



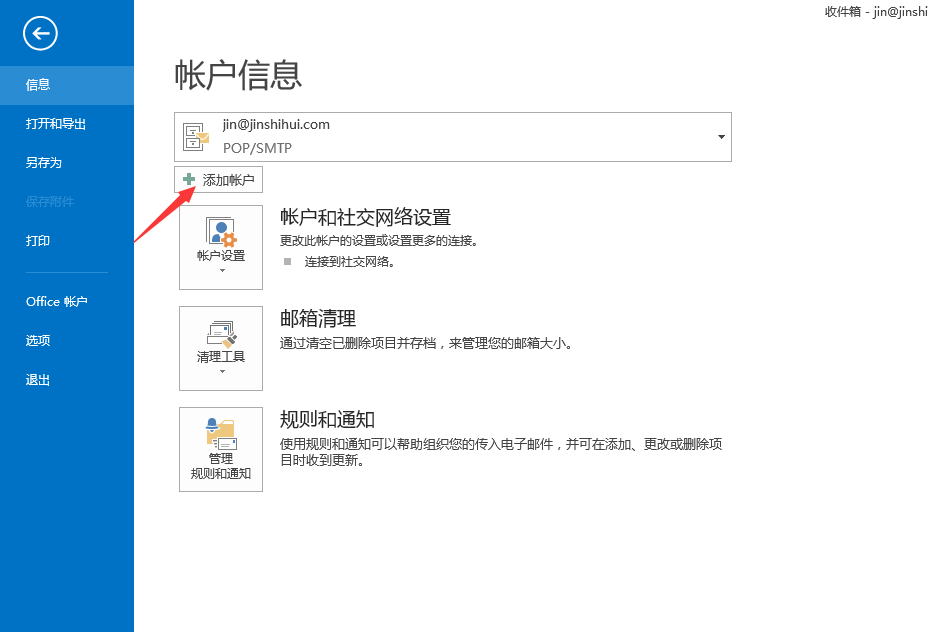
添加完账户，如下图所示：

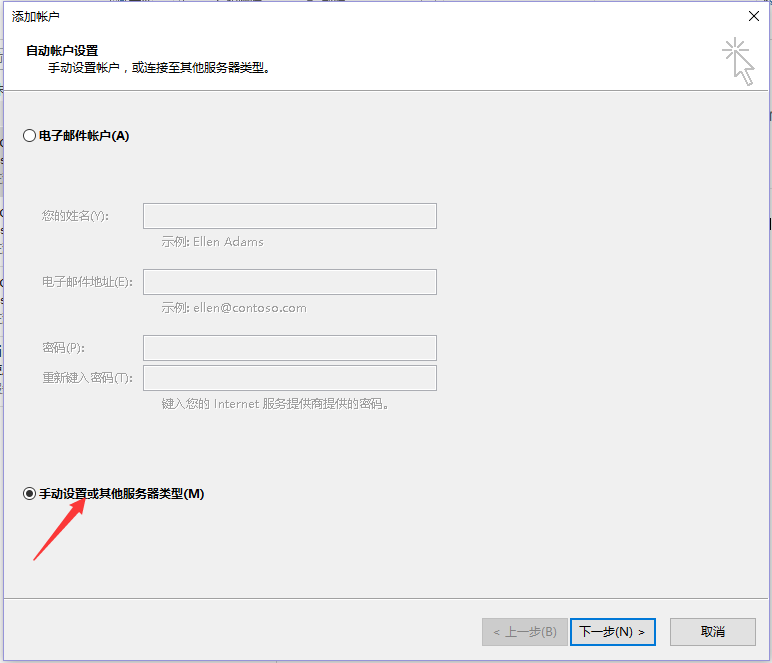


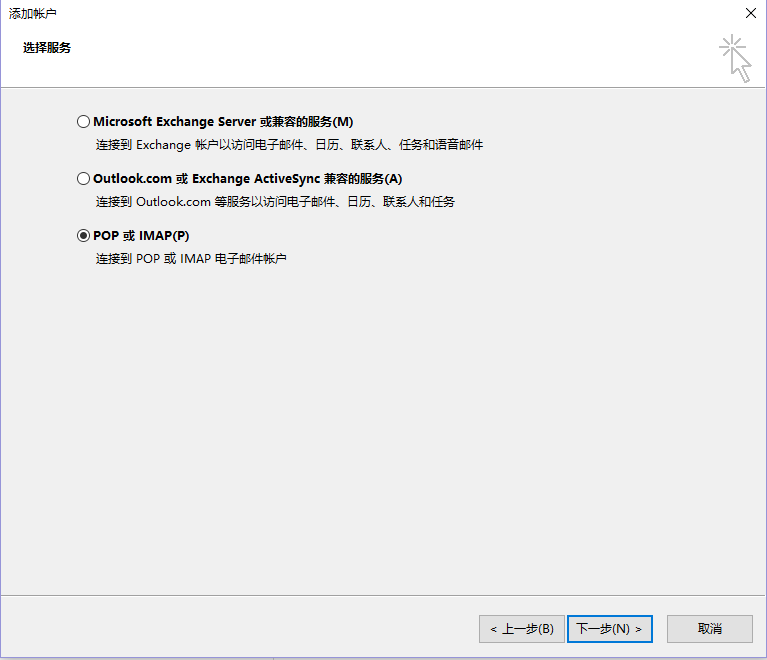
# 邮件客户端的安装和配置

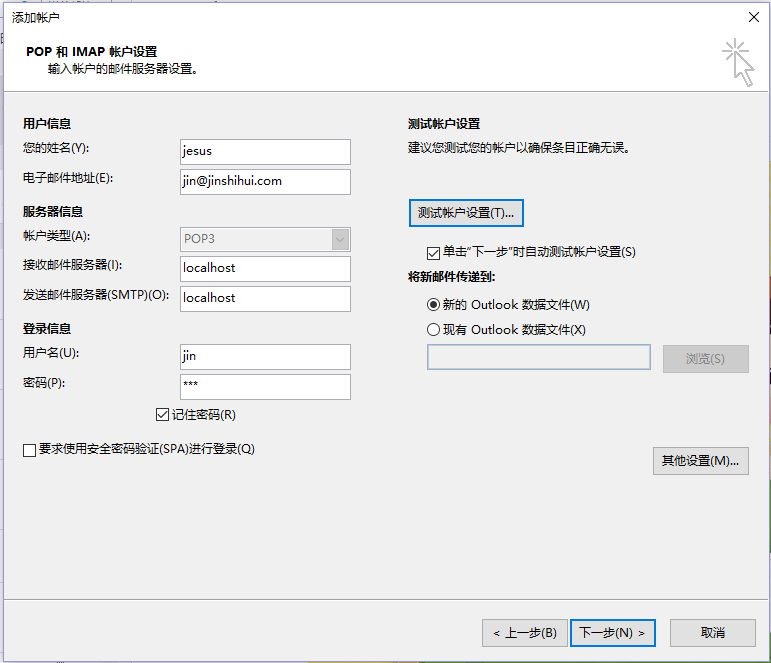
常用的邮件客户端有outlook和foxmail，在这里我们使用outlook客户端。

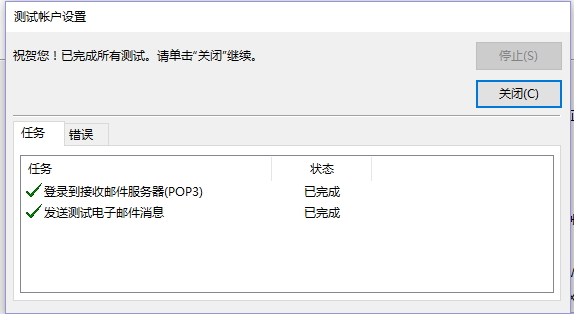
我们开始添加新的账户，选择文件-添加账户

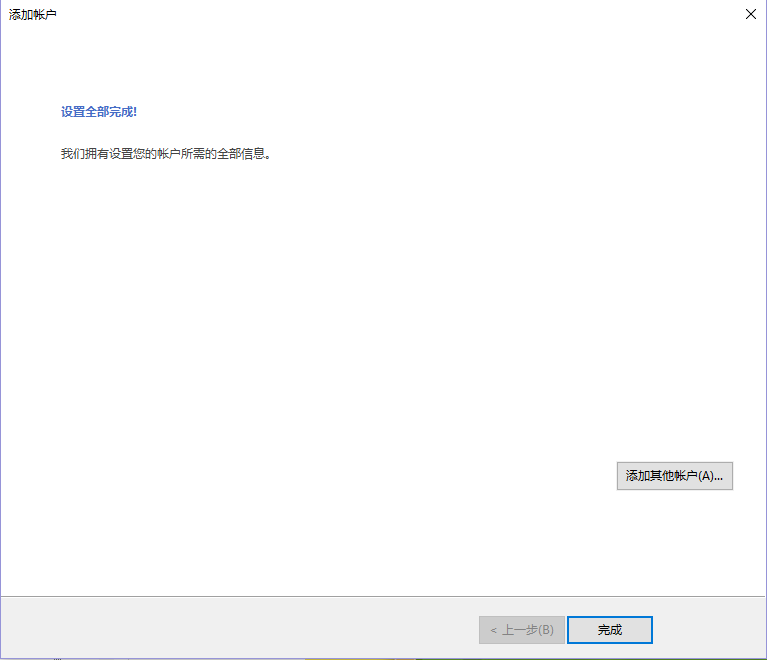












这样我们就添加账户成功了。

# 邮件发送案例

## 搭建开发环境

### 创建数据库和表

说明：使用的是oracle数据库

CREATE TABLE T\_USER(

userid INT PRIMARY KEY,

username VARCHAR(20) DEFAULT NULL,

PASSWORD VARCHAR(20) DEFAULT NULL,

nickname VARCHAR(20) DEFAULT NULL,

email VARCHAR(30) DEFAULT NULL,

state INT DEFAULT NULL,

CODE VARCHAR(64) DEFAULT NULL

)

这步就不写了，自己去创建

### 创建maven项目

### 引入相关的jar包（重点）

<dependency>

<groupId>javax.mail</groupId>

<artifactId>mail</artifactId>

<version>1.4.7</version>

</dependency>

### 设计注册页面

## 用户注册功能

### 编写用户注册的servlet层

### 编写用户注册的service层

### 编写用户注册的dao层

## 发送激活邮件

### 编写邮件发送工具类（重点）

主要代码如下：

/\*\*

\* 邮件发送的工具类

\* @author jinshihui

\*/

public class MailUtils {

/\*\*

\* 发送邮件的方法

\* @param to 接收邮件的人

\* @param code 激活码

\* @throws MessagingException

\* @throws AddressException

\*/

public static void sendMail(String to,String code) throws Exception{

//1、创建连接对象，连接到邮箱服务器

Properties pro = new Properties();

pro.setProperty("host", "localhost");

Session session = Session.getInstance(pro, new Authenticator() {

@Override

protected PasswordAuthentication getPasswordAuthentication() {

return new PasswordAuthentication("service", "123");

}

});

//2、创建邮件对象

Message message = new MimeMessage(session);

//设置发件人

message.setFrom(new InternetAddress("service@jinshihui.com"));

//设置收件人

message.setRecipient(RecipientType.TO, new InternetAddress(to));

//设置邮件的主题

message.setSubject("激活邮件");

//设置邮件的正文

message.setContent("<h1>请点击如下链接</h1><h3><a href='http://localhost:8080/demo/dada?code="+code+"'>http://localhost:8080/demo/dada?code="+code+"</a></h3>", "text/html;charset=UTF-8");

//3、发送一封邮件

Transport.send(message);

}

### 在Service层中调用邮件发送的方法

# 调用html模板发送html内容的邮件

## 开发思路

1、首先要有一个模板文件，我们要知道文件的路径

2、读取html模板文件的内容。

3、对模板中的参数进行替换

4、编写发送邮件的工具类

示例代码

参考https://blog.csdn.net/qq\_24448899/article/details/76945237这个url

# BodyPart类

这是一个抽象类，这个类对包含在多部分中的部分进行建模。子类提供实际的实现，BodyPart实现了Part接口。因此，它包含一组属性和一个“内容”。

# MimeBodyPart类

这个类表示MIME主体部分，它实现了BodyPart抽象类和MimePart接口。MimeBodyParts包含在MimeMultipart对象中。

## 方法

### public void setContent(Object obj, String type)

设置这部分内容的一种方便的方法。该部分在内部将内容包装在DataHandler中。

### public void setDataHandler(DataHandler dh)

该方法提供了设置该部分内容的机制。DataHandler包装实际内容。

### public void setFileName(String filename)

如果可能的话，设置与此部分相关的文件名。如果这部分表示从文件中加载的“附件”，那么很有用。文件名通常是一个简单的名字，不包括目录组件。

# Multipart类

这是一个抽象类，Multipart是一个容纳多个body部件的容器。Multipart提供检索和设置其子部分的方法。

## 方法

### public synchronized void addBodyPart(BodyPart part)

//向multipart添加一个部分，BodyPart被添加到现有部件列表中。

# MimeMultipart类

MimeMultipart类是一个Multipart抽象类的实现，它使用MIME约定来处理多部分数据。

# DataHandler类

DataHandler类为许多不同来源和格式的数据提供一致的接口。它使用datacontenthandler管理简单的流来字符串转换和相关操作。该对象对应于DataHandler特定实例中表示的数据的MIME类型。

## 方法

### public DataHandler(DataSource ds)

创建一个引用指定数据源的DataHandler实例。数据以字节流的形式存在。数据源将提供一个InputStream来访问数据

# DataSource接口

DataSource接口为JavaBeans激活框架提供了对任意数据集合的抽象。它以InputStreams和OutputStreams(在适当的情况下)的形式提供了该数据的类型以及对该数据的访问。

# FileDataSource类

FileDataSource类实现了一个简单的数据源对象，该对象封装了一个文件。它通过FileTypeMap对象提供数据类型服务。FileDataSource类将文件的数据类型委托给FileTypeMap类的子类。setFileTypeMap方法可用于为FileDataSource实例显式设置FileTypeMap。如果没有设置FileTypeMap, FileDataSource将调用FileTypeMap的getDefaultFileTypeMap方法来获取系统的默认FileTypeMap。

# Session类

会话类表示邮件会话，不进行子类化。它将邮件API使用的属性和默认值收集在一起。一个默认会话可以由桌面上的多个应用程序共享。也可以创建非共享会话. 会话类提供对实现存储、传输和相关类的协议提供者的访问。协议提供程序使用以下文件配置: javamail.providers、javamail.default.providers javamail.address.map javamail.default.address.map。

## 方法

### public static Session getInstance(Properties props)

//获取一个新的session对象。

参数props：保存相关属性的对象，客户端应该为JavaMail规范的附录A中列出的属性提供值，特别是：

mail.store.protocol

mail.transport.protocol

mail.host

mail.user

mail.from

在所有情况下作为默认的值是不可能工作的。

### public synchronized void setDebug(boolean debug)

设置此会话的调试设置，由于调试设置只能在创建会话之后才打开，要在会话构造函数中打开调试，请将传递给构造函数的Properties对象中的property mail.debug设置为true。mail.debug属性的值用于初始化每会话调试标志。对setDebug方法的后续调用操作每会话调试标志，对mail.debug属性没有影响。

# Message类

这个类模拟电子邮件消息。这是一个抽象类。子类提供实际的实现。消息包含一组属性和一个“内容”。文件夹中的消息还有一组标记，用于描述文件夹中的状态。

## 方法

### public abstract void setFrom(Address address)

//在信息中设置from属性

参数address表示发件人的地址

### public abstract void setRecipients(RecipientType type, Address[] addresses)

//设置收件人地址。指定类型的所有地址都被address参数替换。

### public abstract void setSubject(String subject)

//设置消息的主题

### public abstract void setSentDate(Date date)

//设置消息的发送时间

### public void setContent(Multipart mp)

该方法将给定的多部分对象设置为消息的内容

# Transport类

对消息传输进行建模的抽象类，子类提供实际的实现。注意，Transport继承了Service类，它提供了许多常用方法来命名传输、连接到传输以及监听连接事件。

## 方法

### public static void send(Message msg)

//发送一个信息。消息将被发送到消息中指定的所有收件人地址(从Message方法getallreceiver返回)，使用适合于每个地址的消息传输。send方法在发送消息之前调用saveChanges方法。