Санкт-Петербургский Политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Отчёт по лабораторной работе

Дисциплина: Сети и телекоммуникации

Тема: Клиент протокола SMTP

Выполнил студент гр. 43501/3 _____ В.Е. Бушин
(подпись)

Руководитель _____ К.Д. Вылегжанина
(подпись)

“_” _____2016 г.

Санкт-Петербург
2016

Задание

Разработать приложение для операционных систем семейства Windows или Linux, обеспечивающее функции клиента протокола SMTP. Основные возможности.
Приложение должно реализовывать следующие функции:
1) Создание нового письма, включающего такие поля, как From (отправитель), To (получатель), Subject (тема), Carbon copy (дополнительные адресаты), Blind copy (дополнительные скрытые адресаты), Body (текст)
2) Формирование всех необходимых заголовков письма, с тем, чтобы приёмная сторона не рассматривала данное письмо как спам.
3) Подключение к указанному SMTP-серверу и отсылка созданного письма
4) Подробное протоколирование соединения клиента с сервером

Поддерживаемые команды. Разработанное приложение должно реализовывать следующие команды протокола SMTP:

- HELO – передача серверу информации о домене пользователя
- MAIL FROM – передача серверу адреса отправителя письма
- RCPT TO – передача серверу адреса получателя письма
- DATA – передача серверу тела письма
- QUIT – завершение сеанса связи

Настройки приложения. Разработанное приложение должно обеспечивать настройку следующих параметров:
1) Собственное доменное имя для передачи в команде HELO
2) Адрес отправителя
3) IP-адрес или доменное имя сервера SMTP

Выполнение задания:

1. Реализация приложения

Протокол SMTP описан в RFC 5321. Клиент был реализован на языке Java, использовалась среда разработки IntelliJ IDEA Community Edition 2016.1.1.
Для работы с SMTP-сервером использовались методы классов из библиотеки JavaMail. Это Java API предназначенное для получения и отправки электронной почты с использованием протоколов SMTP, POP3 и IMAP.
Для создания нового письма и заполнения его полей использовались следующие методы класса MimeMessage:

- setFrom() – для заполнения поля отправителя письма;
- addRecipient() – для добавления получателя письма, а также дополнительных адресатов и скрытых дополнительных адресатов;
- setSubject() – для указания темы письма;
- setContent() – собственно содержимое самого письма.

Соединение с сервером происходит с помощью вызова метода connect() класса Transport следующим образом:

- connect(host, "login", "password") – где переменная host это доменное имя SMTP-сервера, а вместо "login" и "password" нужно указать логин и пароль пользователя.

Также в реализованном приложении сначала происходят настройки нового SMTP-соединения с помощью методов класса Properties.

2. Тестирование приложения

Для тестирования приложения использовался бесплатный почтовый SMTP-сервер компании Yandex – smtp.yandex.ru.
В процессе тестирования были проверены все основные возможности реализованного приложения, а именно: отправка письма, отправка копии письма и отправка скрытой копии письма. Отправленное письмо было получено всеми адресатами. При этом открытые получатели были указаны в списке рассылки (рис. 1,2), а скрытые получатели – не были указаны (рис. 3).

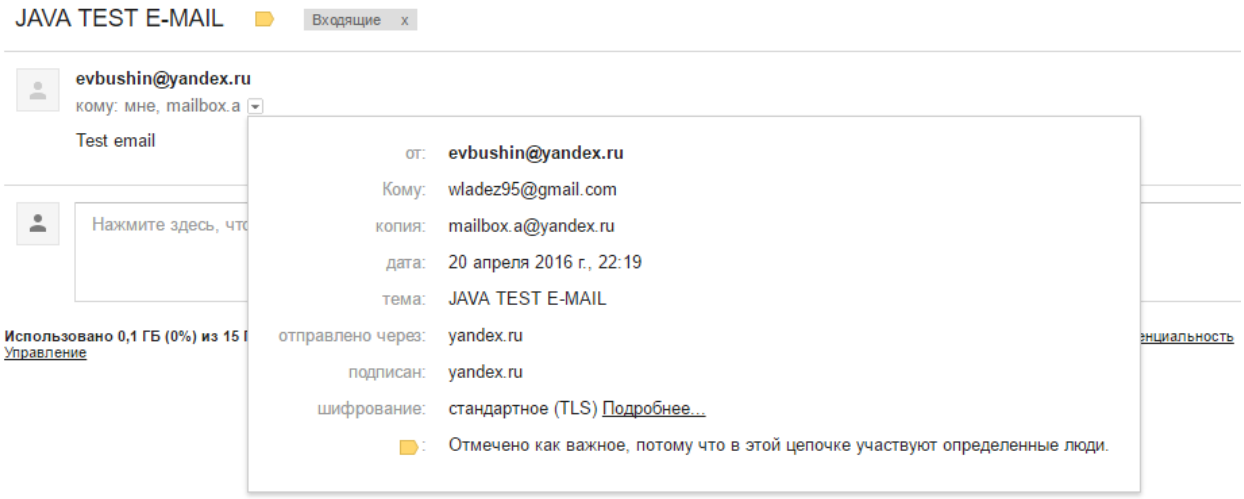

Рис.1. Письмо, полученное основным получателем.

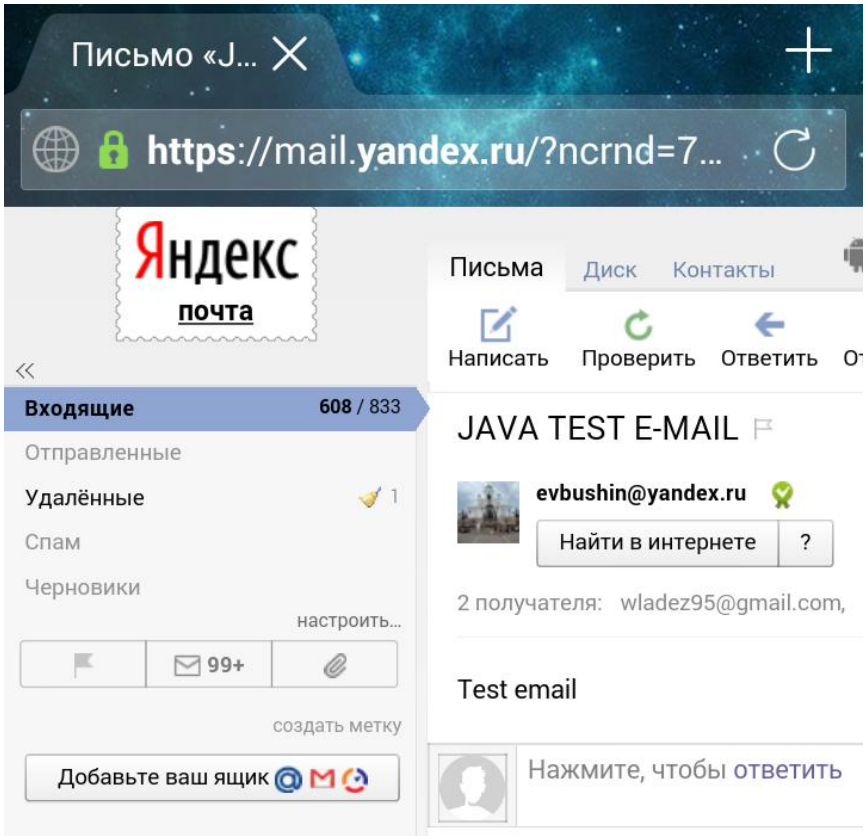

Рис. 2. Копия письма, полученная вторым получателем

Рис. 3. Скрытая копия письма, полученная третьим получателем.

Следующую информацию приложение выводит на консоль:

```
D:\visual studio\SMTPc\out\artifacts\SMTPc_jar>java -jar smtpc.jar
Setting up Mail Server Properties
Mail Server Properties have been setup successfully.
Getting Mail Session
DEBUG: JavaMail version 1.5.2
DEBUG: successfully loaded resource: /META-INF/javamail.default.providers
DEBUG: Tables of loaded providers
DEBUG: Providers Listed By Class Name: {com.sun.mail.smtp.SMTPSSLTransport=javax.mail.Provider[TRANSPORT,s
ORE,imaps,com.sun.mail.imap.IMAPSSLStore,Oracle], com.sun.mail.pop3.POP3SSLStore=javax.mail.Provider[STORE
mail.pop3.POP3Store,Oracle]}
DEBUG: Providers Listed By Protocol: {imaps=javax.mail.Provider[STORE,imaps,com.sun.mail.imap.IMAPSSLStore
un.mail.pop3.POP3Store,Oracle], pop3s=javax.mail.Provider[STORE,pop3s,com.sun.mail.pop3.POP3SSLStore,Oracl
DEBUG: successfully loaded resource: /META-INF/javamail.default.address.map
Mail Session has been created successfully.
Getting Session and sending mail

DEBUG: getProvider() returning javax.mail.Provider[TRANSPORT,smtp,com.sun.mail.smtp.SMTPTransport,Oracle]
DEBUG SMTP: useEhlo true, useAuth true
DEBUG SMTP: trying to connect to host "smtp.yandex.ru", port 25, isSSL false
220 smtp4h.mail.yandex.net ESMTP (Want to use Yandex.Mail for your domain? Visit http://pdd.yandex.ru)
DEBUG SMTP: connected to host "smtp.yandex.ru", port: 25

EHLO Wlad
250-smtp4h.mail.yandex.net
250-8BITMIME
250-PIPELINING
250-SIZE 42991616
250-STARTTLS
250-AUTH LOGIN PLAIN XOAUTH2
250-DSN
250 ENHANCEDSTATUSCODES
DEBUG SMTP: Found extension "8BITMIME", arg ""
DEBUG SMTP: Found extension "PIPELINING", arg ""
DEBUG SMTP: Found extension "SIZE", arg "42991616"
DEBUG SMTP: Found extension "STARTTLS", arg ""
DEBUG SMTP: Found extension "AUTH", arg "LOGIN PLAIN XOAUTH2"
DEBUG SMTP: Found extension "DSN", arg ""
DEBUG SMTP: Found extension "ENHANCEDSTATUSCODES", arg ""
STARTTLS
220 Go ahead
EHLO Wlad
250-smtp4h.mail.yandex.net
250-8BITMIME
250-PIPELINING
250-SIZE 42991616
250-AUTH LOGIN PLAIN XOAUTH2
250-DSN
250 ENHANCEDSTATUSCODES
```

Рис. 4. Вывод на консоль часть 1.

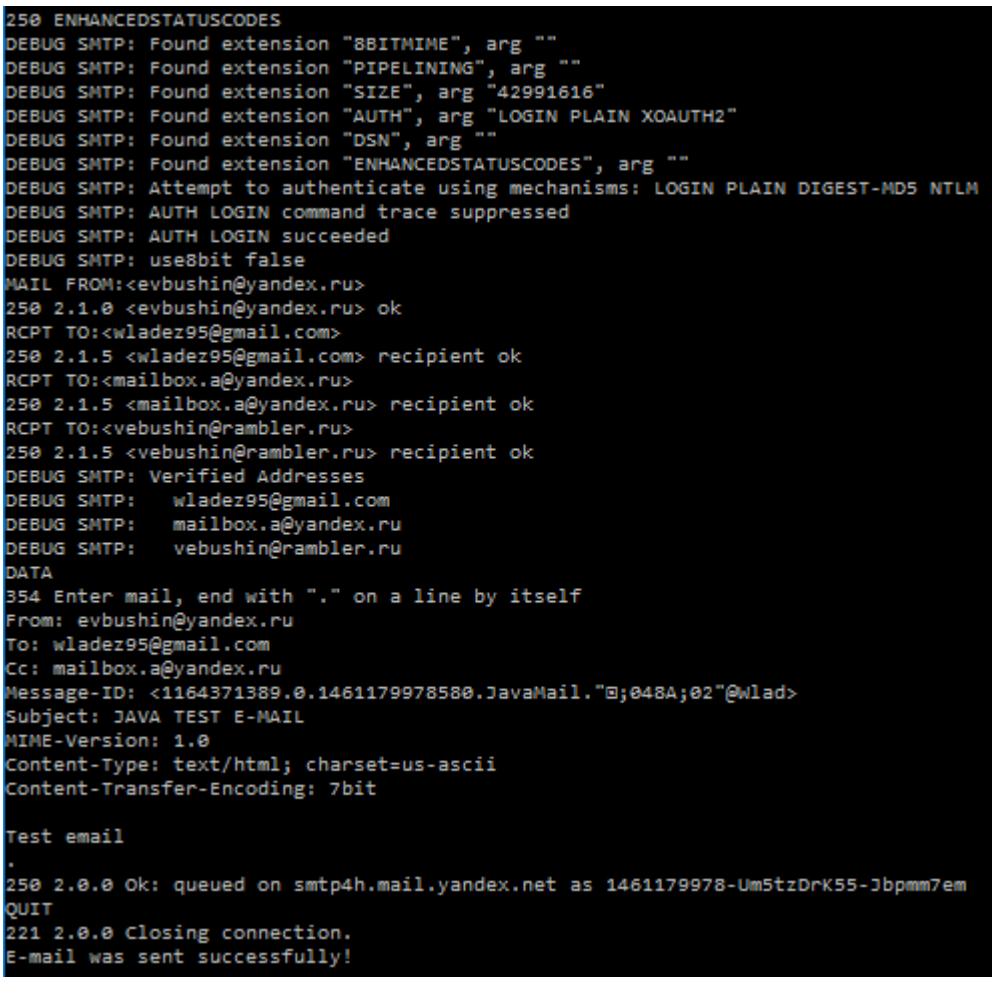

Рис. 5. Вывод на консоль часть 2.

Вывод

Был изучен протокол SMTP и разработан простейший SMTP-клиент, обеспечивающий функции протокола SMTP. Реализация этих функций не вызвала сложности, благодаря наличию открытой Java API JavaMail.

Реализованный клиент был протестирован соответственно исходному заданию. Тесты показали, что клиент работает корректно и все его требуемые функции реализованы.

Приложения

Листинги

Листинг основного модуля Main.java:

```
package ru.spbstu.icc.csse.smtpc;


import java.util.Properties;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;


public class Main {
    static Properties mailServerProperties;
    static Session mailSession;
    static MimeMessage generatedMailMessage;

    static String addressTo = "wladez95@gmail.com";
    static String addressFrom = "evbushin@yandex.ru";
    static String addressCC = "mailbox.a@yandex.ru";
    static String addressBCC = "vebushin@rambler.ru";
    static String host = "smtp.yandex.ru"; //через Яндекс

    public static void main(String args[]) throws MessagingException {
        generateAndSendEmail();
        System.out.println("E-mail was sent successfully!\n");
    }

    public static void generateAndSendEmail() throws MessagingException {
        System.out.println("Setting up Mail Server Properties");
        mailServerProperties = System.getProperties();
        mailServerProperties.put("mail.smtp.host", host);
        mailServerProperties.put("mail.smtp.port", "25");
        mailServerProperties.put("mail.smtp.auth", "true");
        mailServerProperties.put("mail.smtp.starttls.enable", "true");
        mailServerProperties.put("mail.debug", "true");
        System.out.println("Mail Server Properties have been setup
successfully.");

        System.out.println("Getting Mail Session");
        mailSession = Session.getDefaultInstance(mailServerProperties, null);
        generatedMailMessage = new MimeMessage(mailSession);
        generatedMailMessage.setFrom(new InternetAddress(addressFrom));
        generatedMailMessage.addRecipient(Message.RecipientType.TO, new
InternetAddress(addressTo));
        generatedMailMessage.addRecipient(Message.RecipientType.CC, new
InternetAddress(addressCC));
        generatedMailMessage.addRecipient(Message.RecipientType.BCC, new
InternetAddress(addressBCC));
        generatedMailMessage.setSubject("JAVA TEST E-MAIL");
        String emailBody = "Test email";
        generatedMailMessage.setContent(emailBody, "text/html");
        System.out.println("Mail Session has been created successfully.");

        System.out.println("Getting Session and sending mail\n");
        Transport transport = mailSession.getTransport("smtp");
```

```
        transport.connect(host, "login", "password");//здесь должны быть
указаны логин и пароль
        transport.sendMessage(generatedMailMessage,
generatedMailMessage.getAllRecipients());
        transport.close();
    }
}
```