

## Class Email\_Client

```
package EmailClient.com;
// your index number 200636H
// I am Thanushanth.
// My index no is 200636H.

import java.io.File;
import java.io.IOException;
import java.time.format.DateTimeFormatter;
import java.util.Scanner;

public class Email_Client {
    private DateTimeFormatter dateFormat =
DateTimeFormatter.ofPattern("yyyy/MM/dd"); // we get date given format
yyy/mm/dd

    public static void main(String[] args) throws IOException,
ClassNotFoundException {
        EmailSystem emailSystem = new EmailSystem();

        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter option type: \n"
            + "1 - Adding a new recipient\n"
            + "2 - Sending an email\n"
            + "3 - Printing out all the recipients who have birthdays\n"
            + "4 - Printing out details of all the emails sent\n"
            + "5 - Printing out the number of recipient objects in the
application");

        //When email client is start,We read the file
        emailSystem.readFile();
        //When email client is start ,we send the mails to people who have
birthday on today
        emailSystem.sendTodayBirthDayWish();
        System.out.println("Enter your option");
        while (scanner.hasNextInt()) {
            int option = scanner.nextInt();

            switch (option) {
                case 1:
                    // input format - Official: nimal,nimal@gmail.com,ceo
                    // Use a single input to get all the details of a
recipient
                    // code to add a new recipient
                    // store details in clientList.txt file
```

```
// Hint: use methods for reading and writing files

Scanner detail = new Scanner(System.in);
System.out.println("You are select adding a new recipient
\n Input format - Official: nimal,nimal@gmail.com,ceo \nEnter recipient
detail.....");

String recipientDetail = detail.nextLine();
emailSystem.addRecipientDetail(recipientDetail);
break;

case 2:
// input format - email, subject, content
// code to send an email
System.out.println("You are select Sending an email");
System.out.println("Enter the to MailID...");
Scanner data = new Scanner(System.in);
String mailID = data.nextLine(); // get the mailID from
user
System.out.println("Enter the sending mail subject...");
String subject = data.nextLine(); // get the sending
mail subject from user
System.out.println("Enter the sending mail body");
String body = data.nextLine(); // get the sending mail
body from user

emailSystem.sendTheMail(mailID.trim(), subject, body);
break;

case 3:
// input format - yyyy/MM/dd (ex: 2018/09/17)
// code to print recipients who have birthdays on the
given date
System.out.println("You are select printing out all the
recipients who have birthdays\n Input format - yyyy/MM/dd (ex: 2018/09/17)
\nEnter the searching birthday date.....");
Scanner date = new Scanner(System.in);
String inputDate = date.nextLine(); // get the date from
user
emailSystem.checkBirthDay(inputDate); // print
recipient name who have birthdays on the given date
break;

case 4:
// input format - yyyy/MM/dd (ex: 2018/09/17)
// code to print the details of all the emails sent on
the input date
System.out.println("You are select Printing out details
of all the emails sent\n Input format - yyyy/MM/dd (ex: 2018/09/17) \nEnter
the searching mail date.....");
Scanner searchMailDate = new Scanner(System.in);
String inputMailDate = searchMailDate.nextLine(); // get
the date from user
emailSystem.checkSendMailGivenDate(inputMailDate); //
print recipient name who have birthdays on the given date
break;

case 5:
```

```
        // code to print the number of recipient objects in the
application
        System.out.println("You are select printing out the
number of recipient objects in the application");
        System.out.println("the number of recipient objects in
the application => "+emailSystem.getRecipientObjectsInApplication());
        break;
    }

    // start email client
    // code to create objects for each recipient in clientList.txt
    // use necessary variables, methods and classes
    System.out.println("Enter your option");
}
}

// create more classes needed for the implementation (remove the public
access modifier from classes when you submit your code)
```

---

## Class EmailSystem

```
package EmailClient.com;

import java.io.*;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Scanner;

class EmailSystem {
    private final DateTimeFormatter dateFormat =
DateTimeFormatter.ofPattern("yyyy/MM/dd"); // we get date given format
yyy/mm/dd

    // we maintain sending mail history and serialize object
    //this will use for serialization and deserialization
    private SendMailHistory SerializeOrDeserializeObject = new
SendMailHistory();

    //we stop for repeated send birthday wish
    //this object use for check send birthday wish
    private CheckBirthDayWish checkSendBirthDayWish = new
CheckBirthDayWish();

    //We store recipient objects
    private ArrayList<Recipient> recipients = new ArrayList<Recipient>();
```

```
// We store BirthdayWishSendable objects
private ArrayList<BirthDayWishSendable> birthDayWishSender = new
ArrayList<BirthDayWishSendable>();

//create a new file
private File myFile = new File("src\\clientList.txt");

//read the clientList file
// create object => using call the method createRecipientObject
// store in array list
public void readFile(){
    System.out.println("read the file");
    //read file
    try {
        Scanner myReader = new Scanner(myFile);
        while (myReader.hasNextLine()) { // we read multiple detail
            String data = myReader.nextLine();
            Recipient recipient = createRecipientObject(data);
            recipients.add(recipient); // Add recipient in our array
list
        }
        myReader.close();
    } catch (FileNotFoundException e) {
        System.out.println("An error occurred.");
        e.printStackTrace();
    }
}

//add the recipient detail in my file
public void addRecipientDetail(String inputDetail) {

    // write the recipient detail in the file
    try (FileWriter fileWriter = new FileWriter(myFile, true);
        BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
        PrintWriter printWriter = new PrintWriter(bufferedWriter);) {
        printWriter.println(inputDetail);
    }
    catch (IOException i){
        i.printStackTrace();
    }

    // create a recipient object
    Recipient recipient = createRecipientObject(inputDetail);
    recipients.add(recipient); // Add recipient in our array list
}

//this method use for we create a recipient object
//return recipient object
public Recipient createRecipientObject (String inputDetail) {
    Recipient recipient=null;
    //find the recipient class
```

```
String recipientType = (inputDetail.split(":"))[0].trim();
String name = (inputDetail.split(":"))[1].split(",")[0].trim();
//Get the input from user and set name

// Check the recipient type
String emailAddress;
String designation;
LocalDate birthDate;
switch (recipientType) {
    case "Official":
        emailAddress =
(inputDetail.split(":"))[1].split(",")[1].trim(); // Get the input from user
and set email address

        designation =
(inputDetail.split(":"))[1].split(",")[2].trim(); // Get the input from user
and set designation

        //create an object using officialRecipient constructor
recipient = new OfficialRecipient(name, emailAddress,
designation);
        break;
    case "Office_friend":
        emailAddress =
(inputDetail.split(":"))[1].split(",")[1].trim(); //Get the input from user
and set email address

        designation =
(inputDetail.split(":"))[1].split(",")[2].trim(); //Get the input from user
and set designation

        birthDate =
LocalDate.parse((inputDetail.split(":"))[1].split(",")[3]), dateFormat);
//Get the input from user and set birth of date in yyyy/mm/dd format

        //create an object using officialFriend constructor
recipient = new OfficialFriend(name, emailAddress,
designation, birthDate);
        break;
    case "Personal":
        // check the input has a nickname
        if ((inputDetail.split(":"))[1].split(",").length == 4) {
            //input has a nike name
            String nickName =
(inputDetail.split(":"))[1].split(",")[1].trim(); //Get the input from user
and set nickname.

            emailAddress =
(inputDetail.split(":"))[1].split(",")[2].trim(); //Get the input from user
and set email address

            birthDate =
LocalDate.parse((inputDetail.split(":"))[1].split(",")[3]), dateFormat);
//Get the input from user and set birth of date in yyyy/mm/dd format

            //create an object using PersonalRecipient constructor
```

```

        recipient = new PersonalRecipient(name, nickName,
emailAddress, birthDate);
    } else {
        emailAddress =
(inputDetail.split(":"))[1].split(",")[1].trim(); //Get the input from user
and set email address
        birthDate =
LocalDate.parse((inputDetail.split(":"))[1].split(",")[2], dateFormat); //Get
the input from user and set birth of date in LocalDate type

        //create an object using PersonalRecipient constructor
        recipient = new PersonalRecipient(name, emailAddress,
birthDate);
    }
    break;
}
return recipient;
}

//this is check the birthday on today
//call sendTheMail method in email system class
//we stop repeated wish
public void sendTodayBirthDayWish() throws IOException,
ClassNotFoundException {
    LocalDate currentDate =LocalDate.now(); //get the today date

    // traversal our recipients array list
    for (int i=0; i<recipients.size(); i++){
        Recipient currentRecipient = recipients.get(i);

        if (currentRecipient instanceof OfficialFriend){
            if (currentDate.getDayOfMonth() == ((OfficialFriend)
currentRecipient).getDate().getDayOfMonth()
                && currentDate.getMonth() == ((OfficialFriend)
currentRecipient).getDate().getMonth()){
                // System.out.println((OfficialFriend)
currentRecipient).getBDayWish());
                //we stop for repeated send birthday wish
                if
(!checkSendBirthDayWish.checkForSendBirthdayWish(currentRecipient.getEmailAdd
ress())) {

                    //Send the birthday wish
                    sendTheMail(currentRecipient.getEmailAddress(),
"Happy BirthDay", ((OfficialFriend) currentRecipient).getBDayWish());
                    //write the emailID, whose birthday wish send

                    checkSendBirthDayWish.writeSendBirthdayWishEmailID(currentRecipient.getEmailA
ddress());
                }
            }
        } else if (currentRecipient instanceof PersonalRecipient) {
            if (currentDate.getDayOfMonth() == ((PersonalRecipient)
currentRecipient).getDate().getDayOfMonth()
                && currentDate.getMonth() == ((PersonalRecipient)

```

```
currentRecipient().getDate().getMonth()) {
    //System.out.println(((PersonalRecipient)
currentRecipient().getBDayWish());
    //we stop for repeated send birthday wish
    if
(!checkSendBirthDayWish.checkForSendBirthdayWish(currentRecipient().getEmailAdd
ress())) {
        //send the birthday wish
        sendTheMail(currentRecipient().getEmailAddress(),
"Happy Birthday", ((PersonalRecipient) currentRecipient().getBDayWish());
        //write the emailID, whose birthday wish send

checkSendBirthDayWish.writeSendBirthdayWishEmailID(currentRecipient().getEmailA
ddress());
    }
}

//we maintain the arraylist send birthday wish details
if (currentRecipient instanceof BirthDayWishSendable){

birthDayWishSender.add((BirthDayWishSendable)currentRecipient); // using down
casting
}

}

//check recipients who have birthdays on the given date
//print the name who have birthdays on the given date
public void checkBirthDay(String inputDate) {

    //change the data type String to LocalDate
    LocalDate givenDate = LocalDate.parse(inputDate,dateFormat);

    // traversal our recipients array list
    for (int i=0; i<recipients.size(); i++) {
        Recipient currentRecipient = recipients.get(i);
        //only official friend and personal recipient objects have
birthday date
        //So we will check OfficialFriend or PersonalRecipient
        if (currentRecipient instanceof OfficialFriend) {
            //we will check date and month
            if (givenDate.getDayOfMonth() == ((OfficialFriend)
currentRecipient().getDate().getDayOfMonth()
            && givenDate.getMonth() == ((OfficialFriend)
currentRecipient().getDate().getMonth()){
                // && givenDate.getYear() == ((OfficialFriend)
currentRecipient().getDate().getYear() ) {
                    //get the recipient name
                    //print the name
                    System.out.println(((OfficialFriend)
currentRecipient().getName()); // down cast
                }
            }
        }
    }
}
```

```

        else if (currentRecipient instanceof PersonalRecipient) {
            if (givenDate.getDayOfMonth() == ((PersonalRecipient)
currentRecipient).getDate().getDayOfMonth()
                && givenDate.getMonth() == ((PersonalRecipient)
currentRecipient).getDate().getMonth()) {
                //get the recipient name
                //print the name
                System.out.println(((PersonalRecipient)
currentRecipient).getName()); // down cast
            }
        }
    }
}

//create object in mail class
//then we will call sendMail method in mail class
//create SerilizableFile in src folder
//mail object is Serialized in our SerilizableFile
public void sendTheMail(String emailAddress, String subject, String body)
throws IOException, ClassNotFoundException {
    Mail mail = new Mail(emailAddress,subject, body );
    mail.sendMail();

    ArrayList<Mail> tempMailObjects = new ArrayList<Mail>();

    // Deserialization
    tempMailObjects = SerializeOrDeserializeObject.deserialization(
tempMailObjects);

    // Serialization
    SerializeOrDeserializeObject.serialization(tempMailObjects,mail);
}

//deserialized our serializableFile
public void checkSendMailGivenDate(String date) throws IOException {
    LocalDate givenDate = LocalDate.parse(date, dateFormat);
    ArrayList<Mail> tempMailObjects = new ArrayList<Mail>();
    Boolean isSendMail = false;

    ArrayList<Mail> deserializeMailObject;
    // Deserialization
    deserializeMailObject =
SerializeOrDeserializeObject.deserialization(tempMailObjects);
    if (deserializeMailObject.size() != 0) {
        for (int i = 0; i < deserializeMailObject.size(); i++) {
            if (givenDate.getYear() ==
deserializeMailObject.get(i).getSendMailDate().getYear()
                && givenDate.getMonth() ==
deserializeMailObject.get(i).getSendMailDate().getMonth()
                && givenDate.getDayOfMonth() ==
deserializeMailObject.get(i).getSendMailDate().getDayOfMonth()) {
                isSendMail = true;
                System.out.println("send to email address is : " +
deserializeMailObject.get(i).getToMailID());
                System.out.println("send mail subject is : " +
deserializeMailObject.get(i).getSubject());
            }
        }
    }
}

```



```
        System.out.println("send mail body is : " +
deserializeMailObject.get(i).getBody());
    }

    }
    if (!isSendMail) {
        System.out.println("No emails sent on the input date");
    }
} else {
    System.out.println("No emails sent on the input date");
}
}

//return the number of recipient objects in the application
public int getRecipientObjectsInApplication(){
    return Recipient.getNoOfRecipient();
}
}
```

---

## Class Mail

```
package EmailClient.com;

import javax.mail.*;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import java.io.Serializable;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.Properties;

class Mail implements Serializable {

    private LocalDateTime todayDate = LocalDateTime.now();
    private String toMailID;
    private String subject;
    private String body;
    private LocalDate sendMailDate;
    public Mail( String toMailID, String subject, String body) {
        this.toMailID = toMailID; // set Recipient's email ID
        this.subject = subject; // set email subject
        this.body = body; // set the email body
        sendMailDate = LocalDate.from(todayDate); //set the today date
    }
}
```

```
}

//mail is send when the method is call
public void sendMail(){
    final String username = "thanushanthk16@gmail.com";
    final String password = "hnyunhabgxmzpi";

    Properties prop = new Properties();
    prop.put("mail.smtp.host", "smtp.gmail.com");
    prop.put("mail.smtp.port", "587");
    prop.put("mail.smtp.auth", "true");
    prop.put("mail.smtp.starttls.enable", "true"); //TLS

    Session session = Session.getInstance(prop,
        new javax.mail.Authenticator() {
            protected PasswordAuthentication
getPasswordAuthentication() {
                return new PasswordAuthentication(username,
password);
            }
        });

    try {

        Message message = new MimeMessage(session);
        message.setFrom(new
InternetAddress("kanagarajahthanushanth@gmail.com"));
        message.setRecipients(
            Message.RecipientType.TO,
            InternetAddress.parse(toMailID)
        );
        message.setSubject(subject); // set the subject of sending mail
        message.setText(body); // set the sbody of sending mail

        Transport.send(message);

        System.out.println("mail sending is finished");

    } catch (MessagingException e) {
        e.printStackTrace();
    }

}

//get the date of sending mail
public LocalDate getSendMailDate () {
    return sendMailDate;
}

//get the subject of mail
public String getSubject(){
    return subject;
}

//get the body of mail
public String getBody(){
    return body;
}
```

```
}

//get the to mail address
public String getToMailID(){
    return toMailID;
}

}
```

---

## Class Recipient

```
package EmailClient.com;

// user can't create the object in Recipients so, we make abstract class
abstract class Recipient {
    private String name;
    private String emailAddress;
    private static int noOfRecipient; // We have to count number of
    recipients

    // create constructor
    public Recipient(String name, String emailAddress){
        this.name = name;
        this.emailAddress = emailAddress;
        noOfRecipient = getNoOfRecipient() + 1;
    }

    // get the recipient name
    public String getName (){
        return name;
    }

    //get the recipient email address
    public String getEmailAddress (){
        return emailAddress;
    }

    //get the number of recipient
    public static int getNoOfRecipient(){
        return noOfRecipient;
    }
}
```

---

### class OfficialRecipient

```
package EmailClient.com;

//user can create official recipients
class OfficialRecipient extends Recipient {
    private String designation;

    public OfficialRecipient(String name, String emailAddress, String
designation) {
        super(name, emailAddress);
        this.designation = designation;
    }

    // get the official recipient designation
    public String getDesignation () {
        return designation;
    }
}
```

---

### class OfficialFriend

```
package EmailClient.com;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

//user can create official friend recipients
class OfficialFriend extends OfficialRecipient implements
BirthDayWishSendable{
    private LocalDate birthDate;

    public OfficialFriend(String name, String emailAddress, String
designation, LocalDate birthDate) {
        super(name, emailAddress, designation);
        this.birthDate = birthDate;
    }

    // get the official friend recipient Birth of date
    public LocalDate getDate () {
        return birthDate;
    }

    @Override
    public String getBDayWish() {
        return "Wish you a Happy Birthday by Thanu ";
    }
}
```

```
}  
}
```

---

---

### class PersonalRecipient

```
package EmailClient.com;  
import java.time.LocalDate;  
import java.time.format.DateTimeFormatter;  
  
//user can create personal recipients  
class PersonalRecipient extends Recipient implements BirthdayWishSendable{  
    private String nickName;  
    private LocalDate birthDate;  
  
    public PersonalRecipient(String name, String nickName, String  
emailAddress, LocalDate birthDate) {  
        super(name, emailAddress);  
        this.nickName = nickName;  
        this.birthDate = (birthDate);  
    }  
  
    public PersonalRecipient(String name, String emailAddress, LocalDate  
birthDate) {  
        super(name, emailAddress);  
        this.nickName = null;  
        this.birthDate = birthDate;  
    }  
  
    // get the personal recipient nickName  
    public String getNickName (){  
        return nickName;  
    }  
  
    // get the personal recipients birth of date  
    public LocalDate getDate (){  
        return birthDate;  
    }  
  
    @Override  
    public String getBDayWish() {  
        return "Hugs and love on your birthday by Thanu";  
    }  
}
```

## interface BirthDayWishSendable

```
package EmailClient.com;

interface BirthDayWishSendable {
    public String getBDayWish();
}
```

---

## class SendMailHistory

```
package EmailClient.com;

import java.io.*;
import java.util.ArrayList;

class SendMailHistory {

    //create Serialize file or open
    File serializeFile = new File("SerializableFile.ser");

    //Serialization
    public void serialization( ArrayList<Mail> tempMailObjects, Mail mail) {
        // Serialization
        try {
            //Saving of object in a SerializableFile
            FileOutputStream fileOut = new FileOutputStream(serializeFile);
            ObjectOutputStream out = new ObjectOutputStream(fileOut);
            // Method for serialization of object
            for (int i = 0; i < tempMailObjects.size(); i++) {
                out.writeObject(tempMailObjects.get(i));
                out.flush();
            }
            out.writeObject(mail);
            out.flush();

            out.close();
            fileOut.close();

            System.out.println("Object has been serialized");
        } catch (IOException ex) {
            System.out.println("IOException is caught");
        }
    }

    // Deserialization
```

```
public ArrayList<Mail> deserialization(ArrayList<Mail> tempMailObjects)
throws IOException {
    // Deserialization
    if(serializeFile.length() != 0) {

        // Deserialize of multiple object
        // Reading the Multiple object from a file
        // store the mail object in ArrayList
        FileInputStream file = new FileInputStream(serializeFile);
        ObjectInputStream in = new ObjectInputStream(file);
        while (true) {
            // Method for deserialization of object
            try {
                Mail mailDeserialization = (Mail) in.readObject();
                tempMailObjects.add((Mail) mailDeserialization); // add
the mail object in ArrayList
            } catch (EOFException | ClassNotFoundException e) {
                break;
            }
        }
        in.close();
        file.close();
    }
    return tempMailObjects;
}
}
```

---

---

### class CheckBirthDayWish

```
package EmailClient.com;

import java.io.*;
import java.util.Scanner;

class CheckBirthDayWish {
    private File myFile = new File("src\\storeSendBDayWish.txt");

    //check for is the birthday wish send
    public boolean checkForSendBirthdayWish (String emailAddress){
        boolean isBirthdaySend = false;
        //read file
        try {
            Scanner myReader = new Scanner(myFile);
            while (myReader.hasNextLine()) { // we read multiple detail
                String data = myReader.nextLine();
                if(data.equals(emailAddress)){
                    isBirthdaySend = true;
                }
            }
        }
    }
}
```

```
        }  
        myReader.close();  
    } catch (FileNotFoundException e) {  
        System.out.println("An error occurred.");  
        e.printStackTrace();  
    }  
    return isBirthdaySend;  
}  
  
//Write emailID in our file, who's birthday wish send the email  
public void writeSendBirthdayWishEmailID(String emailAddress){  
    // send birthday wish emailID wrote in the file  
    try (FileWriter myFileWriter = new FileWriter(myFile, true);  
        BufferedWriter b = new BufferedWriter(myFileWriter);  
        PrintWriter p = new PrintWriter(b);) {  
        p.println(emailAddress);  
    }  
    catch (IOException i) {  
        i.printStackTrace();  
    }  
}  
}
```