

ÍNDICE

| | |
|---|----|
| Introducción | 3 |
| Objetivo | 3 |
| Requerimientos del sistema | 3 |
| Requerimientos de hardware | 3 |
| Requerimientos de software | 3 |
| Procesos | 4 |
| Ingreso al programa | 4 |
| Ingreso al programa | 5 |
| Elaboración del programa | 5 |
| Sugerencias | 23 |
| Referencias | 24 |

Introducción

El siguiente manual surge de la necesidad de explicar la estructura y el funcionamiento de un programa de correo electrónico como lo es it's not a mail.

Deseando facilitar la conexión de varios servicios y sea más cómodo para el usuario

Objetivo

Informar y especificar al usuario la estructura y conformación del programa a fin de poder hacer soporte, modificaciones y actualizaciones al programa en general.

Requerimientos del sistema

Requerimientos de hardware

- ☐ Equipo, teclado, mouse, monitor, dispositivo móvil.
- ☐ Memoria RAM 2 GB (equipo y dispositivo móvil)
- ☐ Tarjeta de red LAN y/o Wireless
- ☐ Procesador 1.6 GHz.

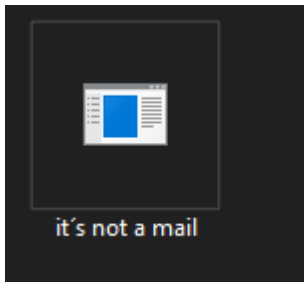
Requerimientos de software

- ☐ Sistema operativo (Windows 7 en adelante).
- ☐ Java 9.0.
- ☐ Conexión internet local.

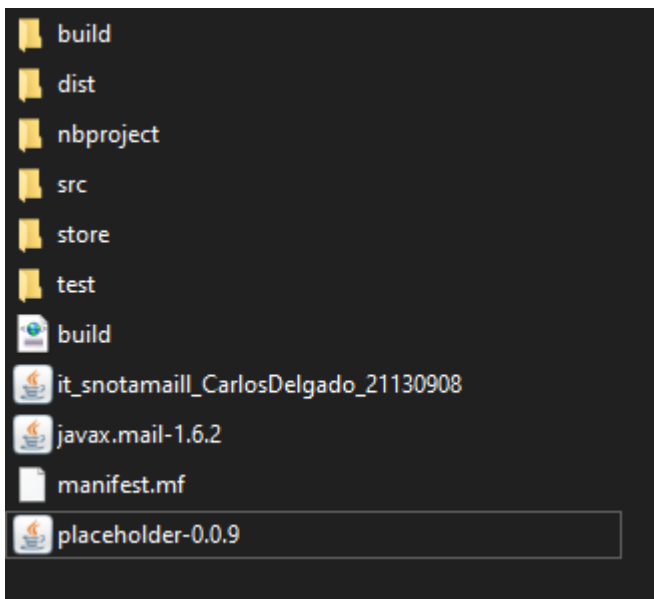
Procesos

Ingreso al programa



para ingresar al código del programa it's not a mail, lo primero que debes de hacer es descargar el archivo correspondiente. Usualmente se encuentra en un archivo zip.






Una vez ahí, descomprimir el archivo y su carpeta correspondiente, con el mismo nombre, en el desglose de las carpetas, encontraremos algunas como build, dist, nbproject, src, store, y los manuales técnicos y de usuario.










De estas carpetas, encontraremos que lo que más nos interesa sería la carpeta src, donde encontraremos los paquetes con las clases, y la carpeta store, donde encontraremos el archivo jar (ejecutable).

| | | |
|--|------------------------|---------------------|
|  Clases | 08/02/2022 11:13 p. m. | Carpeta de archivos |
|  Interfaces | 06/02/2023 10:28 a. m. | Carpeta de archivos |

clases

| | | |
|---|------------------------|-----------------------|
|  ComprobarConexionInternet | 08/02/2022 11:13 p. m. | Archivo de origen ... |
|  EnviarMail | 09/02/2022 11:20 a. m. | Archivo de origen ... |
|  Validacion | 08/02/2022 11:13 p. m. | Archivo de origen ... |

Interfaces

| | | |
|---|------------------------|-----------------------|
|  CrearMensaje.form | 06/02/2023 05:50 p. m. | Archivo FORM |
|  CrearMensaje | 06/02/2023 05:50 p. m. | Archivo de origen ... |
|  Login.form | 06/02/2023 05:57 p. m. | Archivo FORM |
|  Login | 06/02/2023 05:57 p. m. | Archivo de origen ... |
|  mailpng | 06/02/2023 01:45 a. m. | Archivo PNG |
|  Placeholder | 01/02/2023 02:49 p. m. | Archivo de origen ... |
|  placeholder-0.0.9 | 27/10/2013 05:11 p. m. | Executable Jar File |

Ingreso al programa

Elaboración del programa

Para la elaboración de este programa se realizan los siguientes pasos:

1. Abrir Netbeans

2. Crear un nuevo proyecto de tipo Java Application

3. Para una mejor organización, distribuimos en dos paquetes, un paquete de clases que denominaremos como tal contendrá las clases ComprobarConexionInternet, EnviarMail y Validación. Al otro paquete, lo denominamos como Interfaces, que contendrá el frame correspondiente al Login. Y el frame correspondiente a la creación del mensaje.

4. Para este proyecto, es necesario descargar las librerías correspondientes, denominadas

Javamail, activation y Placeholder, todas en su forma de jar desde la opción de agregar librerías.

5. Para empezar, trabajaremos sobre la clase EnviarMail, aquí declararemos todas las variables correspondientes a los correos, tal como correo, contraseña, destinatario, asunto, archivoAdjunto, mensaje, nombre, estado, las rutas, algunos componentes como JTextField y JTextArea que nos servirán más adelante para mostrar nuestro programa en el frame. Creando los constructores, creamos un método que nos servirá para correr nuestro programa, aquí mismo se declaran los servidores de cada correo, tal como Gmail, Hotmail, Yahoo, Live y Outlook. Estos servidores poseen una dirección las cuáles

pueden cambiar con el tiempo, aunque actualmente contamos con que Gmail usa la dirección "smtp.gmail.com", Live utiliza "smtp-live.com", Yahoo utiliza "smtp.mail.yahoo.com" y Hotmail y Outlook comparten la dirección "smtp.office365.com". Todos ellos, utilizando el puerto 587 como recomendación.

6. La librería Mail de java, nos proporcionara distintas clases y métodos que son necesarios para el uso de este programa, entre ellas BodyPart, Message, MessagingException, PasswordException, Session, Transport, Internet.

7. La librería de activation, nos proporciona los métodos DataHandler y FileDataSource.

8. Cada librería con sus métodos llamados realizan una determinada acción, para ello podemos ingresar a la página oficial de java que describe cada acción de estos métodos.

5. <http://dalila.sip.ucm.es/~manuel/JSW1/Slides/Librerias.pdf> Este es un link utilizado para investigar algunas de ellas.

9. El código de la clase EnviarMail deberá ser así:

```
package Clases;
```

```
import Interfaces.Login;
import java.util.Properties;
import javax.activation.DataHandler;
import javax.activation.FileDataSource;
import javax.mail.BodyPart;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.PasswordAuthentication;
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.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
```

```
public class EnviarMail implements Runnable {
    String correo, contraseña, nombre, dest, asu, msg, rutaArchivo;
    JTextField destinatario, asunto, archivoAdjunto;
    JTextArea mensaje;
    String[] vect;
    JLabel estado;

    public EnviarMail(String correo, String contraseña, JTextField destinatario,
        JTextField asunto, JTextField archivoAdjunto, JTextArea mensaje,
        String nombre, JLabel estado) {
        this.correo = correo;
        this.contraseña = contraseña;
        this.destinatario = destinatario;
        this.asunto = asunto;
        this.archivoAdjunto = archivoAdjunto;
        this.mensaje = mensaje;
        this.nombre = nombre;
        this.estado = estado;
        dest = destinatario.getText();
        asu = asunto.getText();
        msg = mensaje.getText();
        rutaArchivo = archivoAdjunto.getText();
        vect = dest.split(";");
    }
}
```



```

@Override
public void run() {
    if (rutaArchivo.equals("")) {

        final String usuario = correo;
        final String pass = contraseña;
        Properties props = new Properties();

        if(Login.servidor == 0){
            props.put("mail.transport.protocol", "smtp");
            props.put("mail.smtp.host", "smtp.gmail.com");
            props.put("mail.smtp.socketFactory.port", "587");
            props.put("mail.smtp.socketFactory.fallback", "false");
            props.put("mail.smtp.starttls.enable", "true");
            props.put("mail.smtp.auth", "true");
            props.put("mail.smtp.port", "587");
        }
        if(Login.servidor == 1){
            props.put("mail.transport.protocol", "smtp");
            props.put("mail.smtp.host", "smtp.live.com");
            props.put("mail.smtp.socketFactory.port", "587");
            props.put("mail.smtp.socketFactory.fallback", "false");
            props.put("mail.smtp.starttls.enable", "true");
            props.put("mail.smtp.auth", "true");
            props.put("mail.smtp.port", "587");
        }
        if(Login.servidor == 2){
            props.put("mail.transport.protocol", "smtp");
            props.put("mail.smtp.host", "smtp.mail.yahoo.com");
            props.put("mail.smtp.socketFactory.port", "587");
            props.put("mail.smtp.socketFactory.fallback", "false");
            props.put("mail.smtp.starttls.enable", "true");
            props.put("mail.smtp.auth", "true");
            props.put("mail.smtp.port", "587");
        }
        if(Login.servidor == 3){
            props.put("mail.transport.protocol", "smtp");
            props.put("mail.smtp.host", "smtp.office365.com");
            props.put("mail.smtp.socketFactory.port", "587");
            props.put("mail.smtp.socketFactory.fallback", "false");
            props.put("mail.smtp.starttls.enable", "true");
            props.put("mail.smtp.auth", "true");
            props.put("mail.smtp.port", "587");
        }

        Session session = Session.getInstance(props,
            new javax.mail.Authenticator() {
                protected PasswordAuthentication getPasswordAuthentication() {
                    return new PasswordAuthentication(usuario, pass);
                }
            });

        session.setDebug(false);
    }
}

```

```

try {
    estado.setText("Enviando Mensaje");
    for (int i = 0; i < vect.length; i++) {
        Message message = new MimeMessage(sesion);
        message.setFrom(new InternetAddress(usuario));
        message.setRecipients(Message.RecipientType.TO,
            InternetAddress.parse(vect[i]));
        message.setSubject(asu);
        message.setText(msg);
        Transport.send(message);
        JOptionPane.showMessageDialog(null, "Su mensaje fué enviado "
            + "exitosamente\na los siguientes contactos: \n" + vect[i]);
    }
    estado.setText("Mensaje enviado");

    mensaje.setText("");
    asunto.setText("");
    destinatario.setText("");
} catch (MessagingException e) {
    estado.setText("Error al enviar mensaje");
    JOptionPane.showMessageDialog(null, "Algo salio mal compruebe la conexcion a internet");
}
}

else {

    Properties props = new Properties();
    if(Login.servidor == 0){
        props.setProperty("mail.smtp.host", "smtp.gmail.com");
        props.setProperty("mail.smtp.starttls.enable", "true");
        props.setProperty("mail.smtp.port", "587");
        props.setProperty("mail.smtp.auth", "true");
    }
    if(Login.servidor == 1){
        props.setProperty("mail.smtp.host", "smtp.live.com");
        props.setProperty("mail.smtp.starttls.enable", "true");
        props.setProperty("mail.smtp.port", "587");
        props.setProperty("mail.smtp.auth", "true");
    }
    if(Login.servidor == 2){
        props.setProperty("mail.smtp.host", "smtp.mail.yahoo.com");
        props.setProperty("mail.smtp.starttls.enable", "true");
        props.setProperty("mail.smtp.port", "587");
        props.setProperty("mail.smtp.auth", "true");
    }
    if(Login.servidor == 3){
        props.put("mail.transport.protocol", "smtp");
        props.put("mail.smtp.host", "smtp.office365.com");
        props.put("mail.smtp.socketFactory.port", "587");
        props.put("mail.smtp.socketFactory.fallback", "false");
        props.put("mail.smtp.starttls.enable", "true");
    }
}

```

```

        props.put("mail.smtp.port", "587");
    }
    Session session = Session.getDefaultInstance(props);
    session.setDebug(false);
    BodyPart texto = new MimeBodyPart();

    try {
        estado.setText("Enviando Mensaje");
        for (int i = 0; i < vect.length-15; i++) {
            texto.setText(msg);
            BodyPart adjunto = new MimeBodyPart();
            adjunto.setDataHandler(new DataHandler(new FileDataSource(rutaArchivo)));
            adjunto.setFileName(nombre);
            MimeMultipart multiParte = new MimeMultipart();
            multiParte.addBodyPart(texto);
            multiParte.addBodyPart(adjunto);
            MimeMessage message = new MimeMessage(session);
            message.setFrom(new InternetAddress(correo));
            message.addRecipient(Message.RecipientType.TO, new InternetAddress(vect[i]));
            message.setSubject(asu);
            message.setContent(multiParte);
            Transport t = null;
            t = session.getTransport("smtp");
            t.connect(correo, contraseña);
            t.sendMessage(message, message.getAllRecipients());
            t.close();
            JOptionPane.showMessageDialog(null, "Su mensaje fué enviado "
                + "éxitosamente \na los siguientes contactos: \n" + vect[i]);
        }

        estado.setText("Mensaje enviado");
        mensaje.setText("");
        asunto.setText("");
        destinatario.setText("");
        archivoAdjunto.setText("");
    } catch (MessagingException ex) {
        estado.setText("Error al enviar mensaje");
        JOptionPane.showMessageDialog(null, "Algo salio mal compruebe "
            + "la conexion a internet");
    }
}

public void start(){
    new Thread(this).start();
}
}

```

Para la clase validación, haremos la declaración y uso de los servidores.
El código de la clase validación sería así:

```
package Clases;

import Interfaces.CrearMensaje;
import Interfaces.Login;
import java.util.Properties;
import java.util.logging.Level;
import javax.mail.MessagingException;
import javax.mail.Session;
import javax.mail.Transport;
import javax.swing.JLabel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

public class Validacion implements Runnable{

    JTextField correo;
    JPasswordField contraseña;
    JLabel error;
    Login jLogin;
    String mail, pass;

    public Validacion(JTextField correo, JPasswordField contraseña, JLabel error, Login jLogin) {
        this.correo = correo;
        this.contraseña = contraseña;
        this.error = error;
        this.jLogin = jLogin;
        mail = this.correo.getText();
        pass = this.contraseña.getText();
    }
}
```

```
@Override
public void run() {
    Properties props = new Properties();

    final String usuario = mail;
    final String contra = pass;

    if(Login.servidor==0)
    {
        props.put("mail.transport.protocol", "smtp");
        props.put("mail.smtp.host", "smtp.gmail.com");
        props.put("mail.smtp.socketFactory.port", "587");
        props.put("mail.smtp.socketFactory.fallback", "false");
        props.put("mail.smtp.starttls.enable", "true");
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.port", "587");
    }
    if(Login.servidor==1)
    {
        props.put("mail.transport.protocol", "smtp");
        props.put("mail.smtp.host", "smtp.live.com");
        props.put("mail.smtp.socketFactory.port", "587");
        props.put("mail.smtp.socketFactory.fallback", "false");
        props.put("mail.smtp.starttls.enable", "true");
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.port", "587");
    }
    if(Login.servidor==2)
    {
        props.put("mail.transport.protocol", "smtp");
```



```

if(Login.servidor==2)
{
    props.put("mail.transport.protocol", "smtp");
    props.put("mail.smtp.host", "smtp.mail.yahoo.com");
    props.put("mail.smtp.socketFactory.port", "587");
    props.put("mail.smtp.socketFactory.fallback", "false");
    props.put("mail.smtp.starttls.enable", "true");
    props.put("mail.smtp.auth", "true");
    props.put("mail.smtp.port", "587");
}
if(Login.servidor == 3){
    props.put("mail.transport.protocol", "smtp");
    props.put("mail.smtp.host", "smtp.office365.com");
    props.put("mail.smtp.socketFactory.port", "587");
    props.put("mail.smtp.socketFactory.fallback", "false");
    props.put("mail.smtp.starttls.enable", "true");
    props.put("mail.smtp.auth", "true");
    props.put("mail.smtp.port", "587");
}
Session sesion = Session.getInstance(props, new javax.mail.Authenticator()
{
    protected javax.mail.PasswordAuthentication getPasswordAuthentication()
    {
        return new javax.mail.PasswordAuthentication(usuario, contra);
    }
});

sesion.setDebug(false);

Transport transporte = null;

try{
    transporte = sesion.getTransport("smtp");
} catch (javax.mail.NoSuchProviderException ex){
    java.util.logging.Logger.getLogger(Validacion.class.getName()).log(Level.SEVERE, null, ex);
}

```

Por último, tenemos la clase para la conexión a internet, con la cual establecemos conexión con google.com para comprobar que el puerto y la red se encuentren en funcionamiento.

```
package Clases;

import java.net.Socket;

public class ComprobarConexionInternet {
    public boolean test()
    {
        String dir = "www.google.com";
        int puerto= 80;
        try
        {
            Socket s = new Socket(dir, puerto);
            if(s.isConnected()){
                System.out.println("Conexión establecida con la dirección: " + dir + " a través del puerto: " + puerto);
            }
        }
        catch (Exception e)
        {
            System.err.println("No se pudo establecer conexión con: " + dir + " a través del puerto: " + puerto);
            return false;
        }
        return true;
    }
}
```

Para el paquete Interfaces, con los frame del login y de crear mensaje, tenemos un diseño simple, entendible y fácil de manipular, su creación se basa en cuadros de texto, etiquetas y botones.

It's not a mail



Login

select an email service

Servidor: ☒ Gmail ☐ Hotmail ☐ Outlook ☐ Yahoo
☐ Live


email address:

Pasword:

Login


Exit

Actualizar Conexión a Internet

 — □ ×

It's not a mail

it's so much better



Update internet connection

to: caarlosd1g@gmail.com, alu.21130908@correo.itlalaguna.edu.mx,javamailxd@gmail.com

subject: mensaje de prueba para correo

mensaje de prueba para aplicacion it's not a mail

Send

clean

Upload file

Logout

Los códigos correspondientes al login, son:

```
package Interfaces;

import Clases.ComprobarConexionInternet;
import Clases.Validacion;
import com.placeholder.PlaceHolder;
import javax.swing.JOptionPane;

public class Login extends javax.swing.JFrame {

    public static String correoEnvia;
    public static String contraseña;
    private boolean conexion;
    public static int servidor;

    public Login() {
        initComponents();

        PlaceHolder correo = new PlaceHolder(jTF_CorreoElectronico, "Agregue su correo");
        PlaceHolder contraseña = new PlaceHolder(jPF_Contraseña, "agregue su contraseña");

        jLabError.setVisible(false);
    }

    public void verificarConexion()
    {
        ComprobarConexionInternet google = new ComprobarConexionInternet();
        conexion = google.test();
        System.out.println(conexion);
        jRB_Gmail.setEnabled(conexion);
    }
}
```

```
public void verificarConexion()
{
    ComprobarConexionInternet google = new ComprobarConexionInternet();
    conexion = google.test();
    System.out.println(conexion);
    jRB_Gmail.setEnabled(conexion);
    jRB_Hotmail.setEnabled(conexion);
    jRB_Outlook.setEnabled(conexion);
    jRB_Live.setEnabled(conexion);
    jRB_Yahoo.setEnabled(conexion);

    if(conexion)
    {
        System.out.println("Esta conectado a internet");
        this.jLab_EstadoInternet.setText("Esta conectado a internet");
        jTF_CorreoElectronico.setEnabled(true);
        jPF_Contraseña.setEnabled(true);
        jBut_Ingresar.setEnabled(true);
    }
    else
    {
        System.out.println("No hay conexion");
        jLab_EstadoInternet.setText("Sin conexión a internet");
        jTF_CorreoElectronico.setEnabled(false);
        jPF_Contraseña.setEnabled(false);
        jBut_Ingresar.setEnabled(false);
    }
}
```

```

private void jBut_ActualizarConexionActionPerformed(java.awt.event.ActionEvent evt) {
    verificarConexion();
}

private void jPF_ContraseñaActionPerformed(java.awt.event.ActionEvent evt) {

}

private void JBut_SalirActionPerformed(java.awt.event.ActionEvent evt) {

    this.dispose();
}

private void jBut_IngresarActionPerformed(java.awt.event.ActionEvent evt) {
    jBut_ActualizarConexion.doClick();
    if(conexion == true){

        if(jRB_Gmail.isSelected()){
            servidor = 0;
            jTF_CorreoElectronico.setEnabled(true);
            jPF_Contraseña.setEnabled(true);
        }
        if(jRB_Hotmail.isSelected())
            servidor = 3;

        if(jRB_Outlook.isSelected())
            servidor = 3;
    }
}

```

```

private void jBut_IngresarActionPerformed(java.awt.event.ActionEvent evt) {
    jBut_ActualizarConexion.doClick();
    if(conexion == true){

        if(jRB_Gmail.isSelected()){
            servidor = 0;
            jTF_CorreoElectronico.setEnabled(true);
            jPF_Contraseña.setEnabled(true);
        }
        if(jRB_Hotmail.isSelected())
            servidor = 3;

        if(jRB_Outlook.isSelected())
            servidor = 3;

        if(jRB_Live.isSelected())
            servidor = 1;

        if(jRB_Yahoo.isSelected())
            servidor = 2;

        Validation obj = new Validation(this.jTF_CorreoElectronico, this.jPF_Contraseña, jLabError, this);
        obj.start();
        jLabError.setVisible(false);
    }
    else
    {
        JOptionPane.showMessageDialog(null, "Intenta: \n      -Comprobar los cables de red, el módem y el router \n
    }
}

```

El código correspondiente al frame de creación de mensaje es:

El código correspondiente al frame de creación de mensaje es:

```
package Interfaces;

import Clases.ComprobarConexionInternet;
import Clases.EnviarMail;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;

public class CrearMensaje extends javax.swing.JFrame {

    public String adjunto;
    private boolean conexion;
    String correo;
    String contraseña;
    String nombre;
    String archivo;

    public CrearMensaje(String correo, String contra) {
        this.correo = correo;
        this.contraseña = contra;
        initComponents();
    }

    private CrearMensaje() {

    }

    public void verificarConexion()
    {
        ComprobarConexionInternet google = new ComprobarConexionInternet();
        conexion = google.test();
        System.out.println(conexion);

        if(conexion)
        {
            System.out.println("Esta conectado a internet");
        }
        else
        {
            System.out.println("No hay conexion");
            jLab_EstadoInternet.setText("Sin conexión a internet");
            jTF_Destinatario.setEnabled(conexion);
            jTF_Asunto.setEnabled(false);
            jTA_Mensaje.setEnabled(false);
            jBut_Enviar.setEnabled(false);
            jBut_Descartar.setEnabled(false);
            jBut_Adjuntar.setEnabled(false);
        }
    }
}
```

```

private void jButton_ActualizarConexionActionPerformed(java.awt.event.ActionEvent evt) {
    verificarConexion();
}

private void jButton_CerrarSesionActionPerformed(java.awt.event.ActionEvent evt) {
    this.dispose();
    Login ventana = new Login();
    ventana.setVisible(true);
}

private void jButton_AdjuntarActionPerformed(java.awt.event.ActionEvent evt) {
    JFileChooser fc = new JFileChooser();
    int option = fc.showOpenDialog(this);
    if(option == JFileChooser.APPROVE_OPTION) {
        archivo = fc.getSelectedFile().getPath();
        nombre = fc.getSelectedFile().getName();
        jTextField_Adjunto.setText(archivo);
    }
}

private void jButton_DescartarActionPerformed(java.awt.event.ActionEvent evt) {
    jTextField_Destinatario.setText("");
    jTextField_Asunto.setText("");
    jTextField_Mensaje.setText("");
    jTextField_Adjunto.setText("");
}

private void jButton_EnviarActionPerformed(java.awt.event.ActionEvent evt) {
    jButton_ActualizarConexion.doClick();

    if(conexion == true){
        if(jTextField_Destinatario.getText().equals("")){
            JOptionPane.showMessageDialog(null, "No ha agregado el destinatario");
        }
        else{
            int valor = 5;
            if(jTextField_Asunto.getText().equals("")){
                valor = JOptionPane.showConfirmDialog(rootPane, "¿Esta seguro que desea enviar el correo sin asunto?");
            }
            if(valor == 5 || valor == 0){
                EnviarMail obj = new EnviarMail(correo, contraseña, jTextField_Destinatario, jTextField_Asunto, jTextField_Adjunto, jTextField_Mensaje, nombre,
                obj.start();
                obj = null;
            }
        }
    }
}

```

Todos los códigos mostrados anteriormente, se tienen acceso desde el archivo rar. Este mismo proyecto, también se puede importar desde java para su modificación, y uso, puede ocurrir el error de que las librerías no se importen, por lo tanto, hay que importarlás directamente, dando click derecho en librerías y agregar.

Sugerencias

Algunas sugerencias que podemos dar para el correcto funcionamiento del programa son:

- Configurar los perfiles en los servidores como Google Gmail y Yahoo para que el perfil permita el envío de aplicaciones menos seguras, pues de no hacerse al correr el programa indicara datos erróneos, ya que no se tendrá acceso a las mismas.
- Las clases y los frames, están relacionadas unas con otras, por lo tanto, es necesario modificar el código correspondiente de cada llamado en las clases para tener un buen funcionamiento, ya sea al cambiar los servidores, o incluso en los componentes visuales, ya que estos se llaman en algunas clases como EnviarMail y también se hace uso de estos llamados en los mismos frames.
- Se pueden presentar fallos con los puertos, por lo tanto, hay que verificar la conexión de estos.

Referencias

A continuación, se presentan algunas páginas de internet, que se utilizaron como recursos para la

elaboración de este programa, dichos ejemplos fueron modificados para poder realizar un programa más completo.

□ <https://www.youtube.com/watch?v=s5RYPoQTXBg>

□ <https://www.youtube.com/watch?v=9OmqqMVWIFM>

□ <https://www.youtube.com/watch?v=Dj1t53SH7nk>

□ López, R. G. (2021, 15 marzo). JavaMail: Envía e-mails desde tu proyecto Java. Adictos al trabajo. Recuperado 9 de febrero de 2022, de

<https://www.adictosaltrabajo.com/2008/12/01/javamail/#:%7E:text=JavaMail%20se%20trata%20de%20una,instalaci%C3%B3n%20y%20uso%20de%20ella.>

□ Download activation.jar : activation « a « Jar File Download. (s. f.). 2015 Demo Source and Support Ltd. Recuperado 9 de febrero de 2022, de

<http://www.java2s.com/Code/Jar/a/Downloadactivationjar.htm>

□ Cómo agregar un 'placeholder' en un JTextField. (s. f.). Stack Overflow en español.

Recuperado 9 de febrero de 2022, de

<https://es.stackoverflow.com/questions/42187/c%C3%B3mo-agregar-unplaceholder-en-un-jtextfield>