



TECNOLÓGICO
NACIONAL DE MÉXICO

Tecnológico Nacional de México
Instituto Tecnológico de la laguna



Especialidad:

Ing. Sistemas Computacionales.

Docente:

M.A.E. Pablo Saucedo Martínez

Materia:

Tópicos Avanzados de Programación

Nombre:

Roberto Esquivel Troncoso

Núm. Control:

19130519

Unidad 1: Aplicación

Manual de técnico

ÍNDICE

1. Introducción.....	2
2. Login.....	2
2.1 Clase usada.....	2
2.2 Código del Frame	3
3. Mensaje nuevo	5
3.1 Clase usada.....	5
3.2 Código del Frame	9
4. Librería usada	16
4.1 Información	16
4.2 Enlace de descarga.....	16
5. Capturas de pantalla del funcionamiento	17

1. Introducción

Herramientas utilizadas para el desarrollo de la aplicación

- IDE NETBEANS 8.2 RC

Lenguaje de programación utilizado para este proyecto

- Java

Librerías usadas

- javamail 1.4.5

Aplicación

- Ejecuta el archivo appMailler.jar para iniciar la aplicación.

2. Login

2.1. Clases usadas

- Clase Validar
- Librerías usadas para verificar el correo electrónico.

```
import java.util.regex.Matcher;  
import java.util.regex.Pattern;
```

- Método correo valido, verifica que el correo introducido cumpla con las características definidas.

```

public class Validar{ // clase para la validacion del correo y contraseña

    public static boolean correoValido(String email){
        String emailPattern = "^[_a-z0-9-]+(\\.[_a-z0-9-]+)*@" +
            "[a-z0-9-]+(\\.[a-z0-9-]+)*(\\.[a-z]{2,4})$";

        Pattern pattern = Pattern.compile(emailPattern);

        if(email != null){ // verifica que la cadena enviada no sea nula
            Matcher matcher = pattern.matcher(email); // verifica que el correo sea valido
            if(matcher.matches()){
                return true; // si es valido retorna un verdadero
            }
            else
                return false; // si no es valido retorna un falso
        }
        return false; // si la cadena es nula se envia un falso
    }
}

```

2.2 Código del Frame

- Librerías empleadas
 - La primera librería ayuda a mostrar un icono personalizado en la aplicación.
 - La segunda librería muestra notificaciones personalizadas en la aplicación

```

import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

```

- Al iniciar la aplicación, se muestra el icono seleccionado
- Y pone el cursor sobre el nombre de usuario

```

public Login() {
    initComponents();
    // icono del frame
    setIconImage(new ImageIcon(getClass().getResource("/resources/icon.png")).getImage());

    //Se pone el foco en el nombre al iniciar
    jTextNombre.requestFocus();
}

```

- Código de botón entrar, este código verifica que el correo sea válido.

```
private void jButLoginActionPerformed(java.awt.event.ActionEvent evt) {  
    if(Validar.correoValido(jTextCorreo.getText())){  
        this.setVisible(false); // se quita el Frame del login  
        AppMailerGUI a = new AppMailerGUI(jTextNombre.getText(),jTextCorreo.getText(), jPassword.getText());  
        // se manda el correo y la contraseña al nuevo frame  
        a.setVisible(true); // se activa el Frame AppMailerGUI  
    }else  
        JOptionPane.showMessageDialog(null, "Correo invalido");  
}
```

- Al dar clic el foco se pasa de nombre a correo electrónico, después a contraseña y en contraseña se da clic en entrar.

```
private void jTextCorreoActionPerformed(java.awt.event.ActionEvent evt) {  
    // al dar enter se pone el foco en password  
    jPassword.requestFocus();  
}  
  
private void jPasswordActionPerformed(java.awt.event.ActionEvent evt) {  
    // al dar enter se da clic en automatico el boton entrar  
    jButLogin.doClick();  
}  
  
private void jTextNombreActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    jTextCorreo.requestFocus();  
}
```

3. Mensaje nuevo

3.1. Clases usadas

Clase Envio

- Librerías usadas

```
import java.util.Properties;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.activation.DataHandler;
import javax.activation.FileDataSource;
import javax.mail.Authenticator;
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.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeBodyPart;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;
import javax.swing.JOptionPane;
```

- Atributos empleados

```
public class Envio {
    // atributos de la clase
    private String destinatario; // guardar la cadena de los destinatarios
    // para despues separarla
    private String []vect; // en caso de ser mas de un correo a enviar
    // se crea una matriz para almacenar todos los correos
    private String asunto, mensaje; // asunto y mensaje a enviar
    private String correo, contrasena; // correo y contraseña

    // separador de correos electronicos, la funcion de este separador es dividir
    // los correos electronicos para almacenarlos en una matriz, esto ayuda a enviar
    // de uno en uno los mensajes a los diferentes correos electronicos
    private String separador = ";";
```

- Constructores de la clase

```
// constructores de la clase
public Envio(){ // contrsuctror vacio
    destinatario = null;
    asunto = null;
    mensaje = null;
    correo = "No-ID";
    contrasena = "No-ID";
}

// contrsuctor para el llenado de la informacion
public Envio(String destinatario,String asunto,String mensaje, String mail,String passs){
    this.destinatario = destinatario;
    this.asunto = asunto;
    this.mensaje = mensaje;
    this.correo = mail;
    this.contrasena = passs;
}
```

- Método para el envío del mensaje

```
//metodo para envio de correo sin adjunto
public void mensaje(boolean oculto){ // llega un booleano para verificar si es un mensaje oculto o no
// separa los correos electronicos mediante el split (";")
String[] vect = destinatario.split(separador);

// se prepara la pagina de gmail
Properties propiedad = new Properties();
    propiedad.setProperty("mail.smtp.auth","true");
    propiedad.setProperty("mail.smtp.starttls.enable","true");
    propiedad.setProperty("mail.smtp.host","smtp.gmail.com");
    propiedad.setProperty("mail.smtp.port", "587");
// se prepara la sesion para el gmail
    Session sesion = Session.getDefaultInstance(propiedad, new Authenticator() {
        protected PasswordAuthentication getPasswordAuthentication(){
            return new PasswordAuthentication(correo,contrasena);
        }
    });
    Message mail = new MimeMessage(sesion);

    try {
        for(int i=0;i<vect.length;i++){ // enviamos los correos electronicos ya sea 1 o n correos
            // correo electronico del usuario
            mail.setFrom(new InternetAddress(correo));
            // correo electronico a enviar el mensaje, si oculto es verdadero el mensaje se envia
            // en modo oculto
            mail.setSubject("Mensaje oculto");
            mail.setText(mensaje);
            mail.setRecipients(Message.RecipientType.TO, InternetAddress.parse(vect[i]));
            Transport.send(mail);
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

```

// en caso oculto
if(oculto)
    mail.addRecipient(Message.RecipientType.BCC,new InternetAddress(vect[i])); // oculta el correo
else
    mail.addRecipient(Message.RecipientType.TO,new InternetAddress(vect[i])); // sin ocultar el correo

// asunto del correo
mail.setSubject(asunto);
// mensaje deo correo
mail.setText(mensaje);

// se prepara el envio del mensaje
Transport transporte = sesion.getTransport("smtp");
// se conecta con la cuenta de gmail
transporte.connect(correo,contrasena);
// se envia el correo
if(oculto)
    transporte.sendMessage(mail, mail.getRecipients(Message.RecipientType.BCC)); // oculta el correo
else
    transporte.sendMessage(mail, mail.getRecipients(Message.RecipientType.TO)); // sin ocultar el correo

// se cierra el envio del mensaje
transporte.close();
// notifica que el correo se envio exitosamente
JOptionPane.showMessageDialog(null, "Correo enviado a "+vect[i]);
}

} catch (AddressException ex) {
    // se captura el error en caso de tener un error de acceso
    Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);
    // se notifica el error
    JOptionPane.showMessageDialog(null, "Error, verifique el correo y contraseña.");
} catch (MessagingException ex) {
    // se captura el error en caso de tener un mensaje con errores
    Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);
    //se notifica el error
    JOptionPane.showMessageDialog(null, "Error, ocurrio un problema al enviar el correo.");
}
}

```

- Método para el envío del mensaje con archivo adjunto

```

//envio con archivo adjunto
public void mensajeAdjunto(String ruta, boolean oculto){ // llega la ruta y un booleano que verifica
// si se enviara un correo oculto o no

// separa los correos electronicos mediante el split (",")
String[] vect = destinatario.split(separador);

// se prepara la pagina del gmail
Properties propiedad = new Properties();
propiedad.setProperty("mail.smtp.auth","true");
propiedad.setProperty("mail.smtp.starttls.enable","true");
propiedad.setProperty("mail.smtp.host","smtp.gmail.com");
propiedad.setProperty("mail.smtp.port", "587");
// se prepara el inicio de sesion
Session sesion = Session.getDefaultInstance(propiedad, new Authenticator() {
    protected PasswordAuthentication getPasswordAuthentication() {
        return new PasswordAuthentication(correo,contrasena);
    }
});
}

```



```

// Se compone la parte del texto
BodyPart texto = new MimeBodyPart();
try {
    for(int i=0;i<vect.length;i++){ // enviamos los correos electronicos ya sea 1 o n correos
        texto.setText(mensaje);
        // Se compone el adjunto con la imagen
        BodyPart adjunto = new MimeBodyPart();
        adjunto.setDataHandler(
            new DataHandler(new FileDataSource(ruta)));
        String nom = JOptionPane.showInputDialog("Introduce el nombre del documento: ");
        adjunto.setFileName(nom);
        JOptionPane.showMessageDialog(null, "Espere un momento, subiendo el archivo. Clic en aceptar para continuar.");
        // Clase MultiParte para agrupar texto e imagen.
        MimeMultipart multiParte = new MimeMultipart();
        multiParte.addBodyPart(texto);
        multiParte.addBodyPart(adjunto);

        // Utilizando la clase MimeMessage para el mensaje a enviar
        MimeMessage message = new MimeMessage(sesion);
        message.setFrom(new InternetAddress(vect[i])); // correos electornicos

        if(oculto)
            message.addRecipient(Message.RecipientType.BCC, new InternetAddress(vect[i]));
        else
            message.addRecipient(Message.RecipientType.TO, new InternetAddress(vect[i]));

        message.setSubject(asunto); // se agrega el asunto
        message.setContent(multiParte); // y el archivo adjunto
    }
}

```

```

// Se envia el correo.
Transport t = sesion.getTransport("smtp");
// se verifica el correo y contraseña
t.connect(correo,contrasena);
// se envia el mensaje y el archivo adjunto
t.sendMessage(message, message.getAllRecipients());
// se cierra el envio
t.close();
// se notifica si el correo se envio correctamente
JOptionPane.showMessageDialog(null, "Correo enviado a "+vect[i]);
}
} catch (MessagingException ex) {
    // se captura el error en caso de tener un error en el mensaje
    Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);
    //se notifica el error
    JOptionPane.showMessageDialog(null, "Error, ocurrio un problema al enviar el correo.");
}
}
}

```

3.2. Código del Frame

- Librerías usadas

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
```

- Atributos usados para almacenar información

```
public class AppMailerGUI extends javax.swing.JFrame {
    // almacenar el nombre, correo y contraseña
    private final String correo;
    private final String contrasena;

    // en caso de mandar un archivo se almacena la direccion y nombre
    private String arch;
    private String nom;

    // clase para guardar informacion del correo y el mensaje enviado
    JFileChooser jfch = new JFileChooser();
```

- Al iniciar el frame

```

public AppMailerGUI(String nombre, String corr, String contra) { // llega el correo y la contraseña del login
    initComponents();
    // icono para aplicacion
    setIconImage(new ImageIcon(getClass().getResource("/resources/icon.png")).getImage());

    // se almacena el correo y contraseña para el correo a enviar
    correo = corr;
    contrasena = contra;

    // se muestra en el frame el nombre y el correo electronico usados
    jLabNombre.setText("Nombre: "+nombre);
    jLabCorreo.setText("Correo electronico: "+corr);
}

```

- Botón enviar

```

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

    boolean oculto = jCheckOculto.isSelected();

    // usando la clase Envio, se asignan los datos de el destino, el asunto,
    // el texto de el mensaje, el correo y contraseña
    Envio mensaje = new Envio(jTextPara.getText(),jTextAsunto.getText(),
        jTextTexto.getText(),correo,contrasena);
    // se usa el if para verificar si el correo va a tener un archivo adjunto
    // o no
    if(jTextRuta.getText().length()>0){
        mensaje.mensajeAdjunto(jTextRuta.getText(),oculto); // en caso de tener un archivo adjunto
        // se usa el metodo de mensaje adjunto
    }else
        mensaje.mensaje(oculto); // si es un mensaje normal, se usa el metodo de mensaje
}

```

- Botón agregar archivo

```
private void jButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // se utiliza la clase JFileChooser para buscar en la computadora el archivo a enviar y no tener que escribirlo
    JFileChooser dlg = new JFileChooser();
    int option = dlg.showOpenDialog(this); // se abre la ventana
    if (option == JFileChooser.APPROVE_OPTION) { // se selecciona la opcion
        arch = dlg.getSelectedFile().getPath(); // se almacena el archivo
        nom = dlg.getSelectedFile().getName(); // y el nombre
        jTextRuta.setVisible(true); // se pone visible a la jText de la ruta
        jTextRuta.setText(arch); // se pone el archivo
        jTextRuta.setEditable(false); // se quita lo editable al jText de la ruta
    }
}
```

- Botón cancelar archivo

```
private void jButtonCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    // cancelar el documento a enviar
    jTextRuta.setText(""); // se quita el texto de la ruta
    jTextRuta.setVisible(false); // se quita la visibilidad de la ruta
}
```

- Opción regresar a login

```
private void jMenuItemLoginActionPerformed(java.awt.event.ActionEvent evt) {
    // Para cambiar de contraseña
    this.setVisible(false); // se cierra la ventana del Frame actual
    Login l = new Login(); // se crea un nuevo Frame de Login
    l.setVisible(true); // se pone visible
}
```

- Opción guardar destinatarios

```
private void jMenuGuardarDestinatariosActionPerformed(java.awt.event.ActionEvent evt) {  
    // guardar los correos electronicos  
    if(!jfch.showSaveDialog(null)!=0) // se abre la ventana  
        return;  
  
    FileWriter fw=null;  
    PrintWriter pw=null;  
  
    try {  
        //clase Escritura para guardar la cadena  
        fw = new FileWriter(jfch.getSelectedFile());  
        pw = new PrintWriter(fw);  
        //grabar los correos en un txt  
        pw.println(jTextPara.getText());  
    } catch (IOException ex) {  
        Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);  
    }  
    finally{  
        //cerrar los archivos, se pueden dañar y puede que no se guarden  
        if(pw!=null)  
            pw.close();  
  
        try {  
            if(fw!=null)  
                fw.close();  
        } catch (IOException ex) {  
            Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);  
        }  
    }  
}
```

- Opción guardar asunto, mensaje y adjunto

```

private void jMenuGuardarAMAActionPerformed(java.awt.event.ActionEvent evt) {
    // guardar el asunto, el mensaje y la ruta
    if(jfch.showSaveDialog(null)!=0)
        return;

    FileWriter fw=null;
    PrintWriter pw=null;

    try {
        //clase Escritura para guardar la cadena
        fw = new FileWriter(jfch.getSelectedFile());
        pw = new PrintWriter(fw);

//        grabar el asunto
        pw.println(jTextAsunto.getText());
        // se guarda el tamaño del mensaje
        pw.println(jTextTexto.getLineCount());
        // se guarda el mensaje
        pw.println(jTextTexto.getText());
        // se guarda la ruta
        pw.println(jTextRuta.getText());

```

```

    } catch (IOException ex) {
        Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);
    }
    finally{
        //cerrar los archivos, se pueden dañar y no se graben
        if(pw!=null)
            pw.close();

        try {
            if(fw!=null)
                fw.close();
        } catch (IOException ex) {
            Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}

```

- Opción abrir destinatario

```
private void jMenuAbrirDestinatariosActionPerformed(java.awt.event.ActionEvent evt) {  
    // abrir correos electronicos  
    if(jfch.showOpenDialog(null)!=0)  
        return;  
  
    FileReader fr = null;  
    BufferedReader br = null;  
    try {  
        // clase para lectura de archivos de texto  
        fr = new FileReader(jfch.getSelectedFile());  
        br = new BufferedReader(fr);  
  
        // se leen los correos y se insertan en el jText  
        jTextPara.setText(br.readLine());  
  
    } catch (FileNotFoundException ex) {  
        Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);  
    } catch (IOException ex) {  
        Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);  
    }  
    finally {  
        try {  
            if(br!=null)  
                br.close();  
            if(fr!=null)  
                fr.close();  
        } catch (IOException ex) {  
            Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);  
        }  
    }  
}
```

- Opción abrir asunto, mensaje y adjunto

```
private void jMenuItemAbrirAMAAActionPerformed(java.awt.event.ActionEvent evt) {
    // abrir mensajes guardados
    if(jfch.showOpenDialog(null)!=0)
        return;

    FileReader fr = null;
    BufferedReader br = null;
    try {
        // clase de lectura para los archivos de texto
        fr = new FileReader(jfch.getSelectedFile());
        br = new BufferedReader(fr);

//        se leen los datos y se insertan
// se inserta el asunto
jTextAreaAsunto.setText(br.readLine());

// es importante tener el tamaño del mensaje guardado
// ya que de lo contrario solo insertamos una sola linea de todo el
// mensaje a enviar
int tamaño = Integer.parseInt(br.readLine());
System.out.println("Tamaño "+tamaño);

String mensaje = "";
for(int i=0;i<tamaño;i++){
    mensaje += br.readLine()+"\n";
}
// se inserta el mensaje
jTextAreaTexto.setText(mensaje);
// se inserta la ruta del archivo adjunto
jTextAreaRuta.setText(br.readLine());

    } catch (FileNotFoundException ex) {
        Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);
    } catch (IOException ex) {
        Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);
    }
    finally {
        try {
            if(br!=null)
                br.close();
            if(fr!=null)
                fr.close();
        } catch (IOException ex) {
            Logger.getLogger(AppMailerGUI.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```


- Opción borrar y salir de la aplicación

```
private void jMenuBorrarActionPerformed(java.awt.event.ActionEvent evt) {  
    // borrar el correo electronico  
    jTextPara.setText("");  
    // borrar el asunto  
    jTextAsunto.setText("");  
    // borrar el mensaje  
    jTextTexto.setText("");  
    //borrar el archivo adjunto  
    jTextRuta.setText("");  
}  
  
private void jMenuSalirActionPerformed(java.awt.event.ActionEvent evt) {  
    // salir de la app  
    System.exit(0);  
}
```

4. Librería usada


1.1 Información

JavaMail-1.4.5 se trata de una librería desarrollada por SUN encaminada al envío de correos electrónicos directamente desde tu aplicación Java. El uso de ésta librería es muy sencillo, pero detallaremos paso a paso como realizar la instalación y uso de ella.

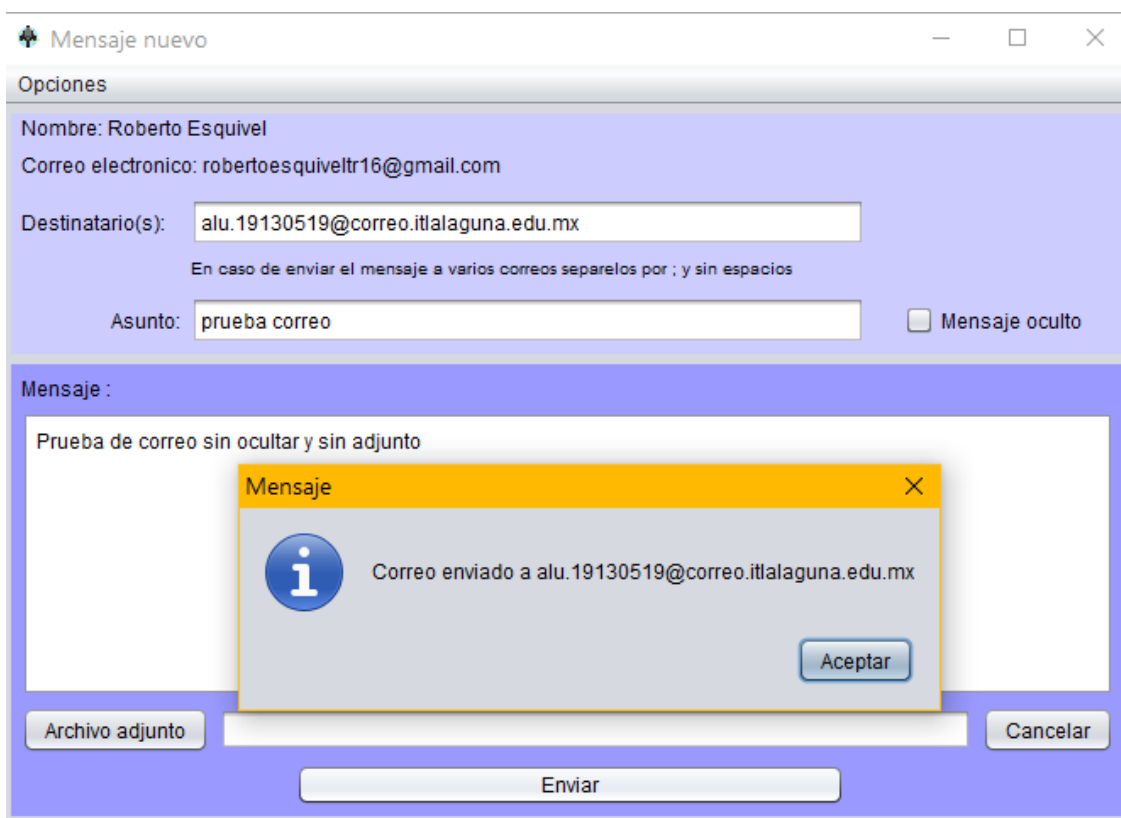
2.1 Enlace de descarga

<https://www.oracle.com/java/technologies/javamail145.html>

5. Capturas de pantalla del funcionamiento



A screenshot of a login window titled "Login". The window has a yellow title bar with standard Windows window controls. The main area has a blue background with the word "Bienvenido" at the top. Below it, there are three input fields: "Nombre:" with the text "Roberto Esquivel", "Correo:" with the text "robertoesquiveltr16@gmail.com", and "Contraseña:" with masked characters "*****". A grey button labeled "Entrar" is positioned below the password field.



A screenshot of a "Mensaje nuevo" (New Message) window. The window has a light blue title bar. Below the title bar is a section labeled "Opciones" (Options) with a light blue background. It contains the following fields: "Nombre: Roberto Esquivel", "Correo electronico: robertoesquiveltr16@gmail.com", "Destinatario(s):" with the text "alu.19130519@correo.itlalaguna.edu.mx", and "Asunto: prueba correo". There is a checkbox labeled "Mensaje oculto" which is currently unchecked. Below the "Opciones" section is a text area for the message body, containing the text "Prueba de correo sin ocultar y sin adjunto". At the bottom of the window are three buttons: "Archivo adjunto", "Enviar", and "Cancelar". A small dialog box titled "Mensaje" is overlaid on the message body, displaying an information icon and the text "Correo enviado a alu.19130519@correo.itlalaguna.edu.mx", with an "Aceptar" button.

prueba correo Recibidos x



robertoesquiveltr16@gmail.com

para mí ▾

Prueba de correo sin ocultar y sin adjunto

Prueba.

RECIBIDO!

OK.

← Responder

→ Reenviar

Mensaje nuevo

Opciones

Nombre: Roberto Esquivel

Correo electronico: robertoesquiveltr16@gmail.com

Destinatario(s):

En caso de enviar el mensaje a varios correos separelos por ; y sin espacios


Asunto:

☒ Mensaje oculto

Mensaje :

Prueba de correo oculto sin adjunto

Mensaje

 Correo enviado a alu.19130519@correo.itlalaguna.edu.mx

Aceptar

Archivo adjunto

Cancelar

Enviar

prueba correo 2 Recibidos x



robertoesquiveltr16@gmail.com

para ▾

Prue

de: **robertoesquiveltr16@gmail.com**
para:
fecha: 5 mar 2021 23:12
asunto: prueba correo 2
enviado por: gmail.com
firmado por: gmail.com
seguridad: Cifrado estándar (TLS) [Más información](#)
: Importante según el criterio de Google.

prueba correo 2 Recibidos x



robertoesquiveltr16@gmail.com

para ▾

23:12 (hace 0 minutos)



Prueba de correo oculto sin adjunto

Responder

Reenviar

Mensaje nuevo

Opciones

Nombre: Roberto Esquivel

Correo electronico: robertoesquiveltr16@gmail.com

Destinatario(s): alu.19130519@correo.itlalaguna.edu.mx

En caso de enviar el mensaje a varios correos separelos por ; y sin espacios

Asunto: prueba correo 2 ☐ Mensaje oculto

Mensaje :

Prueba correo adjunto

Archivo adjunto C:\Users\admin\Desktop\Manual de usuario.docx Cancelar

Enviar

Mensaje

Correo enviado a alu.19130519@correo.itlalaguna.edu.mx

Aceptar



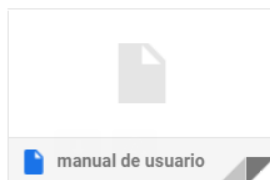
robertoesquiveltr16@gmail.com

para mí ▼

23:14 (hace 1 minuto)



Prueba correo adjunto



Prueba.

OK.

RECIBIDO.