MANUAL TÉCNICO DE APLICACIÓN

Desarrollador: Agustín Alexis Reyes Castillo

23130568

**INTRODUCCIÓN**

**OBJETIVO DEL MANUAL**

Este manual técnico busca explicar el funcionamiento del sistema de envío de correos del aplicativo “RapidMail”, con el fin de facilitar tanto el uso como el mantenimiento de este.

PÚBLICO OBJETIVO: Desarrolladores, testers, administradores del sistema y usuarios finales.

**RESUMEN DE LA APLICACIÓN**

El aplicativo “RapidMail” es un sistema rápido de envío de correos electrónicos por medio del puerto Gmail. Busca agilizar el proceso de envío, así como mejorar el manejo de los archivos y correos destinatarios, con el fin de ser más cómoda, eficiente y ágil para el usuario común que el resto de las aplicaciones con el mismo funcionamiento.

Se compone de una ventana de Login o inicio de sesión, donde el usuario debe acceder con una contraseña de aplicación, generada por medio de la autenticación en dos pasos de su correo electrónico.

También cuenta con la ventana inicial, donde se redacta el correo y se adjuntan y visualizan los archivos, y una ventana específica para selección de correos electrónicos destinatarios.

**ARQUITECTURA DEL SISTEMA**

Para la aplicación se utilizaron una serie de tecnologías aplicables al envío de correos y conexión a puertos SMTP. El lenguaje de programación utilizado en todo momento fue Java.

Versión de Java: Se hizo uso del JDK 17.

Versión de Windows: 11.

Versión de NetBeans: Esta aplicación se creó con Apache NetBeans 20

Librerías y Frameworks:

Versión de Java Mail: Se utilizó la versión de java mail 1.4

Versión de Java Activation: Se usó el Java Activation Framework (JAF) versión 1.1.1

Versión de fontbox: Se utilizó la versión 2.0.30.

Versión de PDFbox: Para la visualización de archivos pdf, se utilizó la versión 2.0.33 de esta librería.

Versión de commons-login: Se utilizó la versión de commons-login 1.3.5.

**DICCIONARIO DE APLICACIÓN**

|  |  |  |
| --- | --- | --- |
| INFORMACIÓN | NOMBRE | FUNCIÓN |
| CLASE | RapidMail | Es la clase inicial del proyecto. No se encuentra en uso |
| * MÉTODO | public static void main(String[] args) | Es el método principal del proyecto. No se encuentra en uso. |
| CLASE | FuncionesLogin | Contiene los métodos necesarios para validar tanto que el correo sea válido como real. |
| * método | public static boolean validarUsuario(String remitente, String clave) | Hace una conexión rápida al puerto para verificar que el correo y la contraseña son correctas. |
| * método | public static boolean validarCorreo(String correo) | Valida el formato del correo (si contiene @ y .) |
| clase | ConexionYValidacion | Es la clase que maneja el envío de los correos electrónicos, así como el archivamiento de los adjuntos en el mensaje. |
| * MÉTODO | public static boolean conectar(String remitente, String clave, String destinatario, String subject, String content) | Toma los elementos del correo electrónico y los junta para proceder a mandarlos. |
| * MÉTODO | public static Multipart tieneArchivo(String content) throws MessagingException, IOException | Se encarga de adjuntar los archivos en caso de que haya. Si el correo no cuenta con archivos, simplemente retorna el mensaje. |
| CLASE | LOGIN | Es el frame de inicio de sesión, en donde introduces tu código y contraseña para sus validaciones correspondientes. |
| * VARIABLE | textRemitente | Es el jText para introducir el correo electrónico |
| * VARIABLE | textClave | Es el jText para introducir la clave del correo |
| * VARIABLE | botonIngresar | Es el botón para ingresar con correo y contraseña, que lleva al usuario a la ventana principal de redacción del mensaje. |
| * Variable | remitenteGlobal | Es la cadena de texto que almacena el remitente para posteriores usos en parámetros de métodos como el de validación. |
| * VARIABLE | claveGlobal | Es la cadena de texto que almacena la clave del correo para posteriores usos en parámetros de métodos. |
| * MÉTODO | public static String getRemitente() | Método para llamar al remitenteGlobal |
| * MÉTODO | public static String getClave() | Método para llamar a la claveGlobal del usuario |
| * MÉTODO | public void botonIngresarActionPerformed (java.awt.event.ActionEvent evt) | Método para manejar el inicio de sesión, que cierra la ventana actual y abre la ventana de creación del mensaje. |
| CLASE | appCorreo | Es la clase con el jFrame principal para el envío de los correos. |
| * variable | botonMasDestinatarios | Label clickeable que abre la ventana para seleccionar múltiples destinatarios. |
| * variable | actualizarCorreos | Botón que debe presionarse después de seleccionar múltiples correos para actualizar |
| * variable | textDestinatarios | Caja de texto para introducir los destinatarios, separados por una coma entre ellos. |
| * variable | textAsunto | Caja de texto donde se escribe el asunto del correo. |
| * variable | textMensaje | Caja de texto donde se escribe el mensaje del texto. |
| * variable | basDestinatario | Label clickeable para borrar textDestinatarios |
| * variable | basAsunto | Label clickeable para borrar textAsunto |
| * variable | basMensaje | Label clickeable para borrar textMensaje |
| * variable | botonEnviarArchivos | Botón que abre un explorador de archivos para ingresar los archivos adjuntos. |
| * variable | listaArchivos | Lista donde se administran los archivos adjuntos |
| * variable | bas1 | Label clickeable para eliminar el primer archivo adjunto. |
| * variable | bas2 | Label clickeable para eliminar el segundo archivo adjunto. |
| * variable | bas3 | Label clickeable para eliminar el tercer archivo adjunto. |
| * variable | panelVisualizacion | Panel donde se visualizan los archivos adjuntos. |
| * variable | tamañoDelArchivo | Label donde se visualizan los tamaños de los archivos. |
| * variable | botonEnviar | Botón para enviar el correo a todos los destinatarios. |
| * variable | botonBorrarTodo | Botón que borra todos los campos de texto a excepción de los destinatarios. |
| * VARIABLE | imageLabel | Variable utilizada para el renderizado de las imágenes de previsualización de los archivos PDF. |
| * VARIABLE | archivosPermitidos | Es el número de archivos permitidos por correo electrónico (3 como máximo). |
| * variable | megasPermitidos | Es el peso máximo de cada archivo (Establecido en 20 megabytes). |
| * VARIABLE | basArray | Es el array de labels de botes de basura para los archivos, para gestionar la visibilidad según el número de archivos del correo. |
| * VARIABLE | Archivos | Es un arrayList de tipo File que almacena los archivos que el usuario desea enviar |
| * VARIABLE | destinatarios | Es el arrayList de correos electrónicos a los cuales se enviará el mensaje. |
| * método | private void botonEnviarArchivosActionPerformed (java.awt.event.ActionEvent evt) | Abre un explorador de archivos para seleccionar aquellos a enviar, además de hacer las validaciones de peso, extensión y número de archivos. |
| * método | private boolean validarExtensiones (File[] files) | Método usado para validar las extensiones de los archivos |
| * método | private boolean validarPeso (File[] files) | Se encarga de validar el peso de cada archivo. |
| * método | private boolean validarNumeroDeArchivos (File[] files) | Se encarga de validar que el número total de archivos no exceda el permitido. |
| * método | private void actualizarListView() | Se encarga de actualizar la lista donde se visualizan los archivos, en caso de ser modificada. También muestra u oculta los botes de basura correspondientes. |
| * método | private void botonEnviarActionPerformed (java.awt.event.ActionEvent evt) | Valida todos los correos de los destinatarios, luego envía a cada uno el correo electrónico deseado. |
| * método | private void basDestinatarioMouseClicked (java.awt.event.MouseEvent evt) | Limpia el campo de texto de destinatarios. |
| * método | private void basAsuntoMouseClicked (java.awt.event.MouseEvent evt) | Limpia el campo de texto de asunto. |
| * método | private void basMensajeMouseClicked (java.awt.event.MouseEvent evt) | Limpia el campo de texto de mensaje. |
| * método | private void bas1MouseClicked (java.awt.event.MouseEvent evt) | Remueve el elemento con índex 0 de la lista de archivos, actualiza la lista y limpia el panel de visualización y la label del tamaño del archivo. |
| * método | private void bas2MouseClicked (java.awt.event.MouseEvent evt) | Remueve el elemento con índex 1 de la lista de archivos, actualiza la lista y limpia el panel de visualización y la label del tamaño del archivo. |
| * método | private void bas3MouseClicked (java.awt.event.MouseEvent evt) | Remueve el elemento con índex 2 de la lista de archivos, actualiza la lista y limpia el panel de visualización y la label del tamaño del archivo. |
| * método | private void botonBorrarTodoActionPerformed (java.awt.event.ActionEvent evt) | Llama al método general de borrarTodo(). |
| * método | private void botonMasDestinatariosMouseClicked (java.awt.event.MouseEvent evt) | Abre el frame para seleccionar los diferentes destinatarios. |
| * método | private void actualizarCorreosActionPerformed (java.awt.event.ActionEvent evt) | Se encarga de añadir uno por uno los correos al campo de texto de destinatarios, donde son leídos para el envío del mensaje. |
| * método | private void listaArchivosValueChanged (javax.swing.event.ListSelectionEvent evt) | Se encarga de colocar la previsualización del archivo en el panel de visualización cuando clickeas en él, además de validar la extensión para renderizar. |
| * método | public void limpiarPanelVisualizacion () | Limpia el panel de visualización de archivos. |
| * método | public void renderPDF (String filePath) | Se encarga de transformar la primera página de un .pdf en una imagen para colocarla en el panel de visualización, además de escalarla al tamaño del panel. |
| * método | private BufferedImage smoothScale (BufferedImage img, int targetWidth, int targetHeight) | Escala la imagen poco a poco para perder menos pixeles al momento de achicarla. |
| * método | private BufferedImage resizeImage (BufferedImage original, int width, int height) | Es un método de escalado de imagen con interpolación bicúbica, que busca mantener un poco la calidad de la imagen original. |
| * método | public static ArrayList<File> getArchivos() | Método para obtener el arrayList de archivos. |
| * método | public void borrarTodo() | Método para borrar todos los campos de texto menos el de destinatarios. |
| * método | public static ArrayList<String> getDestinatarios() | Método para obtener el arrayList de destinatarios. |
| * método | public static void setDestinatarios(ArrayList<String> des) | Método para modificar el arreglo de destinatarios. |
| * método | public void setlabdes(String des) | Método para modificar el campo de texto de los destinatarios. |
| clase | CorreosAEnviar | Es la clase para seleccionar la cantidad de correos que recibirán el mensaje. |
| * VARIABLE | spinerCorreos | Es el jSpinner que determina la cantidad de correos a los que quieres enviar el mensaje. |
| * VARIABLE | tablaCorreos | Es donde se muestran y modifican los correos a los que quieres enviar el mensaje. |
| * VARIABLE | CorreoAEliminar | Es el campo de texto donde introduces el número del correo que deseas eliminar de la lista de destinatarios. |
| * VARIABLE | botonEliminarCorreo | Es el botón para eliminar el correo no deseado de la lista de destinatarios. |
| * VARIABLE | botonListoCorreos | Botón para cerrar la ventana, y pasar la nueva lista de correos a la ventana principal. |
| * VARIABLE | destinatarios | ArrayList para guardar cada uno de los destinatarios escritos. |
| * método | private void botonListoCorreosActionPerformed (java.awt.event.ActionEvent evt) | Actualiza los destinatarios y cierra la ventana. |
| * método | private void botonEliminarCorreoActionPerformed (java.awt.event.ActionEvent evt) | Se encarga de eliminar el correo no deseado de la lista de destinatarios. |
| * método | private void spinerCorreosStateChanged (javax.swing.event.ChangeEvent evt) | Actualiza el tamaño de la tabla para que contenga la cantidad de correos deseada. |
| * método | private void tablaCorreosKeyPressed (java.awt.event.KeyEvent evt) | Actualiza la lista de correos al presionar enter en la tabla de visualización. |
| * método | public void actualizarTabla() | Se encarga de actualizar la tabla de visualización de correos en caso de algún cambio. |

**INSTALACIÓN**

El archivo .zip debe contener la carpeta del proyecto. El ejecutable se encuentra en la carpeta “RapidMail/dist”, y su nombre es “RapidMail.jar”; sólo es necesario como requisito tener instalado el Java Development Kit (JDK) para poder ejecutarlo, y hacer doble clic sobre el mismo.

También se necesita contar con un correo electrónico con verificación en dos pasos activada, y una clave o contraseña de aplicación distinta a la clave utilizada en el correo electrónico.