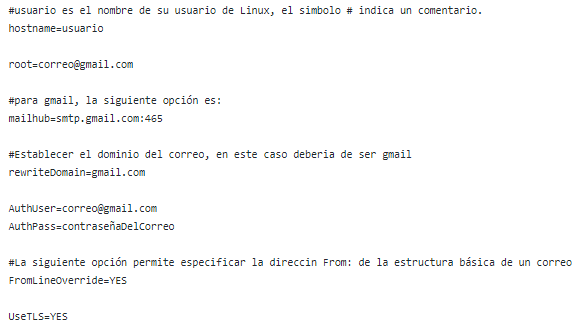
|  |
| --- |
| Diseño de Interfaces |
| Tarea 02 |
| Servidor de Correo electrónico |

|  |
| --- |
| Aldo Vargas  213495653 |

El objetivo de la siguiente actividad es el de crear una interfaz para un servidor de envío de correo electrónico, en donde un usuario pueda escribir correos con asunto y destinatario, para después enviarlos sin tener que abrir un explorador u otro programa gestor de correos.

Configuración

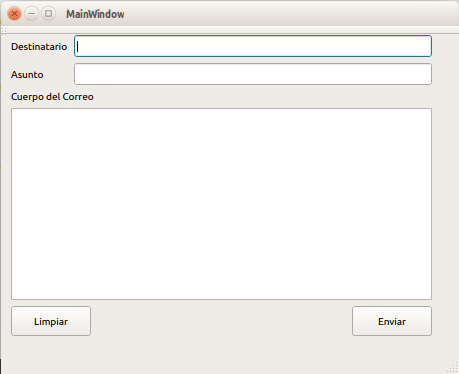
Antes de poder usar el programa es necesario configurar la cuenta de correo que fungirá como servidor, mediante SSMTP que será configurado con los datos de login del usuario y del host del correo.



Una vez configurado el archivo, ya es posible enviar correos.

Diseño de Interfaz

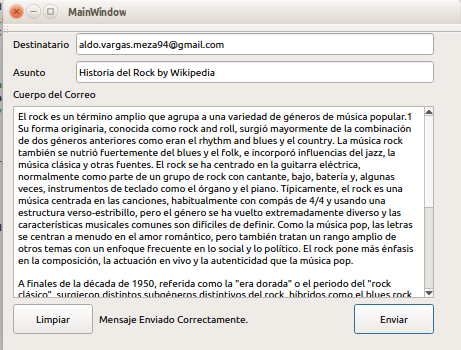
El diseño consta de 3 entradas de texto, las cuales compondrán dentro del código, un comando de envío junto con un archivo contenedor de un correo. Este correo será redactado y dirigido mediante el programa.

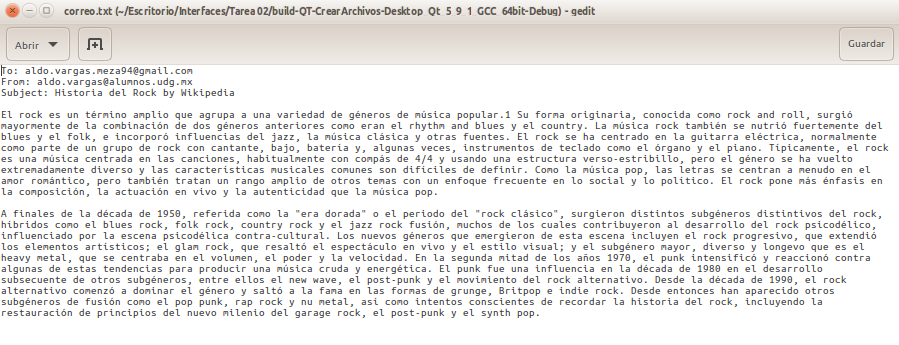
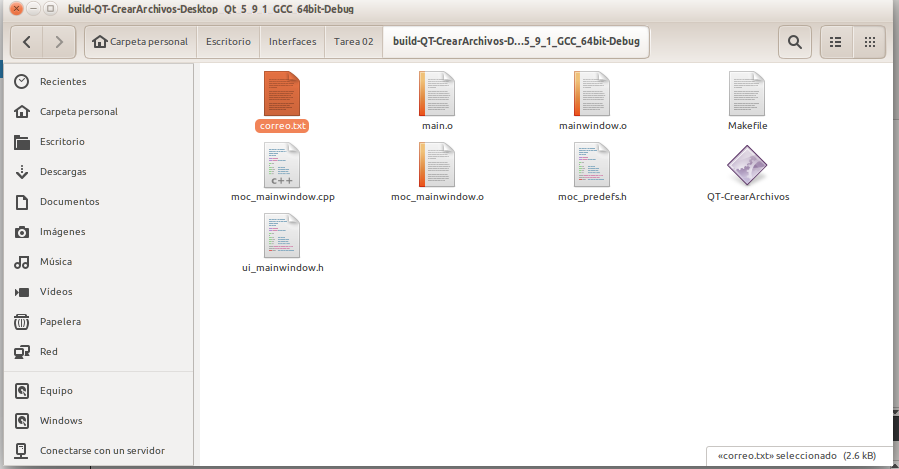


También hay dos botones, los cuales limpian los campos de entrada y envían el comando para crear el archivo de correo y enviarlo.

Envío de Correo

Al ingresar los campos y presionar enviar, el programa genera un archivo de texto en la carpeta de ejecutables. Dicho archivo tiene como plantilla el texto necesario para enviar el correo, complementándose con los textos ingresado en los campos de la interfaz.

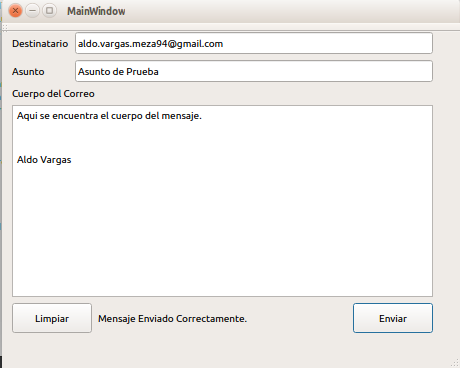




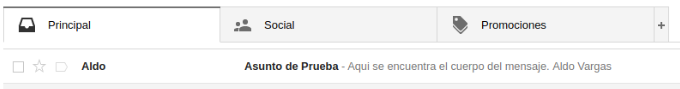
De esta forma el archivo ya ubicado en una ruta específica, puede ser cargado con el comando de ssmtp y ser enviado a un correo electrónico mediante un comando de consola.

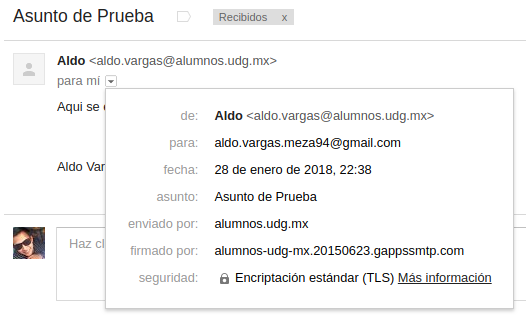
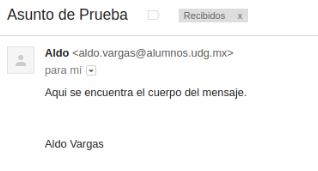
Funcionamiento

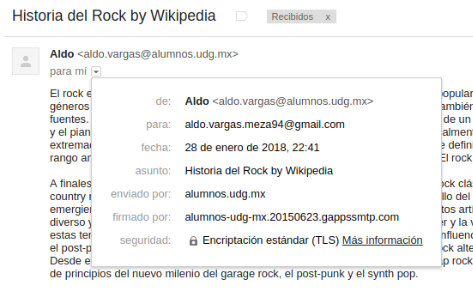
Una vez ingresados todos los campos, se presiona el botón enviar y esperamos a ver el mensaje de confirmación de envío correcto.



El mensaje es enviado correctamente y recibido con los datos especificados en cada campo.

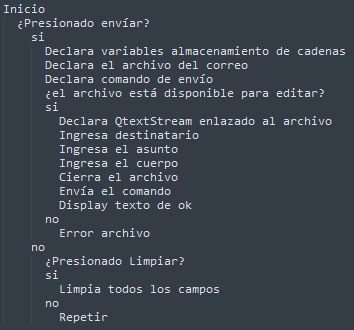


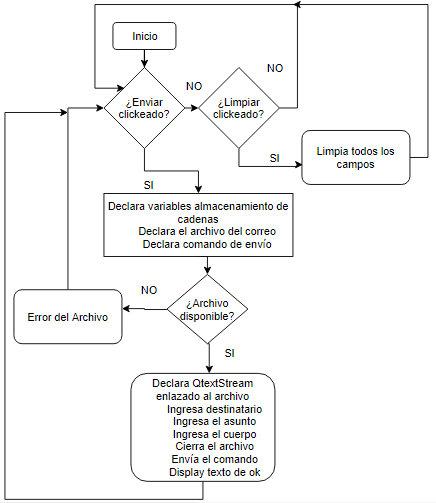




Descripción de funcionamiento

El funcionamiento del programa puede describirse en los siguientes pseudocódigo y diagrama de flujo





Manual para el Usuario

El programa a continuación mostrado es un cliente de correos, listo para redactar un mensaje y enviarlo.

El primer paso es **ingresar una dirección de correo valida**, la cual recibirá el mensaje en formato de texto. Después es necesario especificar el **asunto** en el siguiente campo, y por último **escribir el mensaje a enviar**. Una vez ingresados los datos al dar click al botón enviar y esperar unos segundos podremos ver un mensaje de envío correcto.

Si deseamos iniciar de nuevo, o mandar un nuevo correo podemos dar click al botón limpiar, el cual dejará la interfaz como nueva.

Bibliografía

<https://hetpro-store.com/TUTORIALES/ssmtp-enviar-correo-electronico-ubuntu-16-04/>

Códigos

Archivo .cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QFile>

#include <QTextStream>

#include <stdlib.h>

MainWindow::MainWindow(QWidget\* parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

/\* Configura la interfáz de usuario \*/

ui->setupUi(this);

}

/\* Destructor de objeto \*/

MainWindow::~*MainWindow*() { delete ui; }

void MainWindow::on\_buttonEnviar\_clicked()

{

/\* Creación de cadenas para almacenar datos de entrada \*/

QString dest = ui->textDestinatario->text();

QString subj = ui->textAsunto->text();

QString body = ui->textEmail->toPlainText();

/\* Creación del archivo mediante clase QFile \*/

QFile file("correo.txt");

/\* Comando de envío de archivo mediante SSMTP \*/

QString command = "ssmtp " + dest + "< /home/aldo/Escritorio/Interfaces/'Tarea "

"02'/build-QT-CrearArchivos-Desktop\_Qt\_5\_9\_1\_GCC\_64bit-Debug/"

"correo.txt";

if (file.*open*(QIODevice::WriteOnly | QIODevice::Text)) {

/\* Llenado de datos al archivo creado \*/

QTextStream fileData(&file);

fileData << "To: " << dest << endl;

fileData << "From: aldo.vargas@alumnos.udg.mx" << endl;

fileData << "Subject: " << subj << endl

<< endl;

fileData << body << endl;

file.*close*();

/\* Envío de archivo mediante comando \*/

system(command.toUtf8().constData());

ui->label\_4->setText("Mensaje Enviado Correctamente.");

}

else{

ui->label\_4->setText("Error en archivo de correo.");

}

}

void MainWindow::on\_buttonLimpiar\_clicked()

{

/\* Limpia todos los campos \*/

ui->textAsunto->clear();

ui->textDestinatario->clear();

ui->textEmail->clear();

ui->label\_4->clear();

}

Archivo .h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow {

Q\_OBJECT

public:

explicit MainWindow(QWidget\* parent = 0);

~*MainWindow*();

private slots:

/\* Funciones de Interfáz \*/

void on\_buttonEnviar\_clicked();

void on\_buttonLimpiar\_clicked();

private:

Ui::MainWindow\* ui;

};

#endif // MAINWINDOW\_H

Archivo .pro

#-------------------------------------------------

#

# Project created by QtCreator 2018-01-26T15:24:28

#

#-------------------------------------------------

QT += core gui

greaterThan(QT\_MAJOR\_VERSION, 4): QT += widgets

# Archivo ejecutable

TARGET = QT-CrearArchivos

TEMPLATE = app

# The following define makes your compiler emit warnings if you use

# any feature of Qt which as been marked as deprecated (the exact warnings

# depend on your compiler). Please consult the documentation of the

# deprecated API in order to know how to port your code away from it.

DEFINES += QT\_DEPRECATED\_WARNINGS

# You can also make your code fail to compile if you use deprecated APIs.

# In order to do so, uncomment the following line.

# You can also select to disable deprecated APIs only up to a certain version of Qt.

#DEFINES += QT\_DISABLE\_DEPRECATED\_BEFORE=0x060000 # disables all the APIs deprecated before Qt 6.0.0

# Enlaces de archivos en el programa

SOURCES += \

main.cpp \

mainwindow.cpp

HEADERS += \

mainwindow.h

FORMS += \

mainwindow.ui