Comprensión y Seguridad

Primera Fase

Integrantes del grupo:

Nieves Almodovar Alegria

Elizabeth Itati Balbín Medina

Andrés Fernández Espliguero

Alejandro Gómez López

Jaime Cremades Gomariz

Grupo de prácticas: Martes 11.00 - 13.00

ÍNDICE

1. Software y librerías utilizadas

En nuestro código hemos usado una amplia cantidad de librerías, entre las más destacadas estarían algunas como **java.security.Key y java.io.File** que probablemente sean las más importantes para llevar a cabo la práctica, usamos también librerías derivadas de estas (gestión de ficheros y seguridad), **java.io. FileInputStream, java.io.FileNotFoundException, java.io.FileOutputStream** y **javax.crypto.KeyGenerator, javax.crypto.spec.SecretKeySpec, javax.crypto.Cipher** y **java.security.NoSuchAlgorithmException**. A la hora de desarrollar la interfaz, hemos utilizado **javafx**, y por ende las siguientes librerías :

javafx.application.Application;

javafx.fxml.FXMLLoader;

javafx.scene.Scene;

javafx.scene.layout.BorderPane;

javafx.event.ActionEvent;

import javafx.fxml.FXML;

javafx.scene.text.Text;

javafx.stage.FileChooser;

javafx.stage.FileChooser.ExtensionFilter;

javafx.stage.Stage;

Que nos han sido de gran utilidad a la hora de desarrollar funcionalidades como el escoger un archivo (imagen, música, archivos de texto, etc..) para luego encriptar y desencriptar ese archivo.

Por último cabe destacar la librería **java.util.Base64** necesaria para pasar la clave a base64.

2. Manual de usuario

Primero que todo, tenemos el ejecutable, "CS.jar".

Imagen 1

Al darle doble click, se nos abrirá la siguiente ventana de la interfaz.

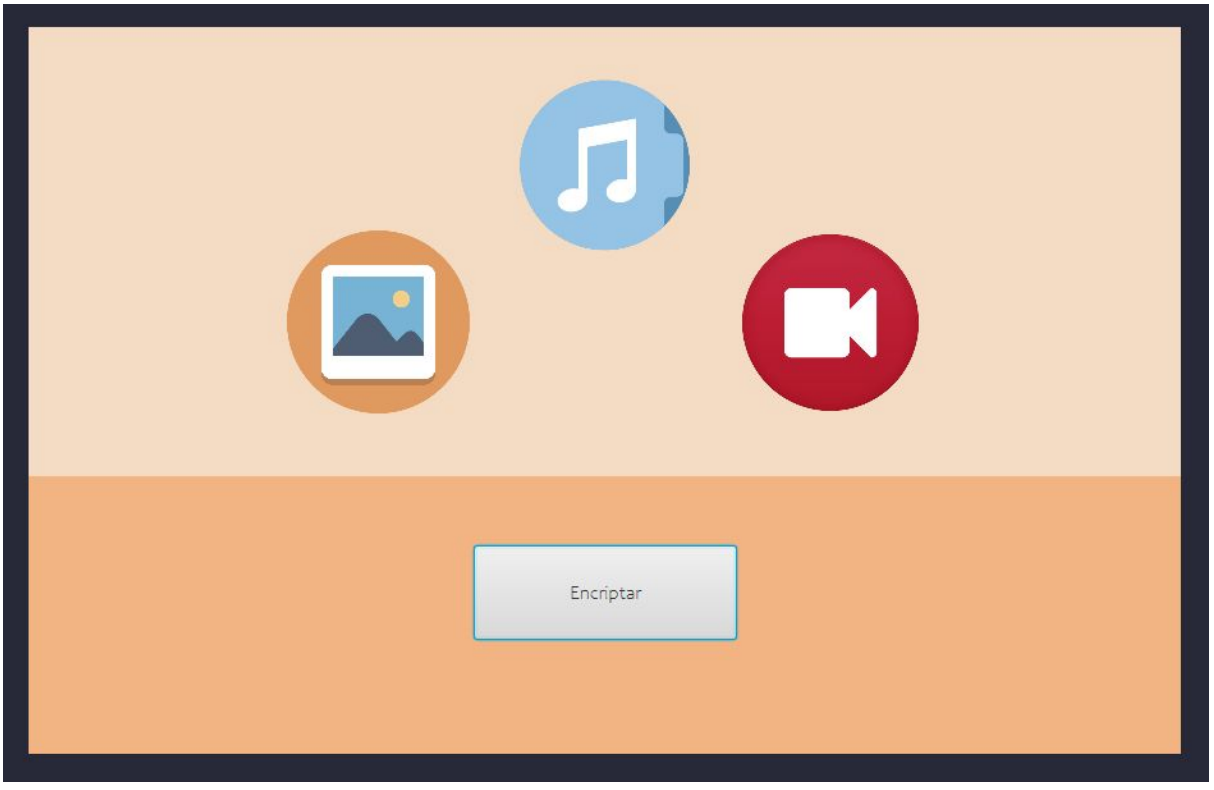

Imagen 2

Si pulsamos el botón "Encriptar", se nos abrirá una ventana y en él podremos seleccionar cualquier archivo multimedia (imagen, sonido o vídeo).

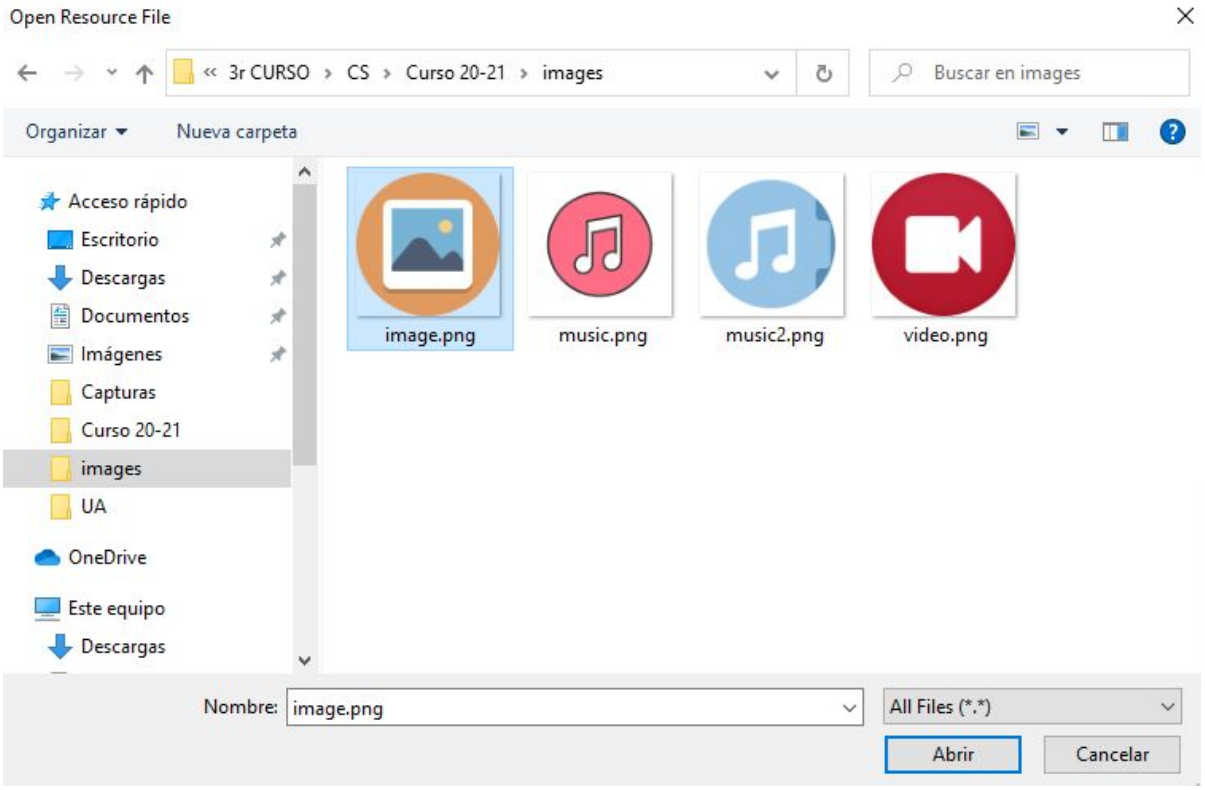

Imagen 3

En este ejemplo hemos pulsado imagen. Como se puede ver a continuación, se ha generado un fichero txt de Claves donde se guarda la clave generada junto a la imagen encriptada.

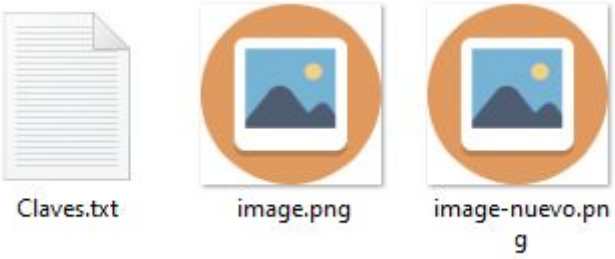

Imagen 4

El fichero "Claves.txt" contiene la clave:

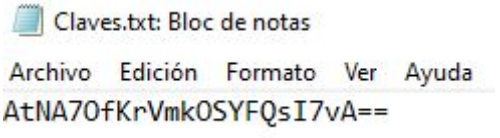

Imagen 5

3. Detalles sobre la implementación

El código está dividido en 4 clases: **Main.java**, **SamplerController.java**, **Sample.fmxl** y **ImageEncDev.java.**

Primero nos encontramos con **Sample.fxml**. En este archivo está definida toda la estructura, en lenguaje fxml, de nuestra interfaz gráfica. El lenguaje fxml está basado en xml, pero desarrollado especialmente por Oracle para trabajar con JavaFX.

En **Main.java** nos aseguramos de que nuestra interfaz de usuario, definida en Sample.fxml, se cargue correctamente y se muestre en la pantalla.

En el archivo **SamplerController.java**, controlamos las interacciones del usuario con la interfaz. De momento en esta primera versión de nuestra aplicación, sólo tenemos un botón de "Encriptar". Cuando pulsemos este botón, se "disparará" un evento que nos permitirá seleccionar el archivo multimedia que queremos encriptar y desencriptar, gracias al elemento **FileChooser**. (ver imágen 3).

Pasamos a generar la clave con la que encriptamos el archivo. Usaremos el tipo de encriptación AES en 128 bits. Creamos una clave aleatoria, que nos generará una clave única para cada archivo que queramos encriptar. El hecho de que esta clave sea aleatoria y no fija, hace que nuestra aplicación sea más segura. Ahora que tenemos el archivo seleccionado, su ruta y la clave generada, iremos haciendo llamadas a las funciones de ImageEncDec.java (explicado más adelante). Primero encriptamos el archivo, y guardamos la clave que nos permitirá desencriptar el archivo en un txt.

Un paso importante que realizamos es pasar la clave a BASE64 y guardarla en este formato en el fichero. Por último leeremos el fichero, decodificamos la clave de BASE64 para poder generar un objeto tipo Key idéntico al inicial (con el que podremos desencriptar).

Como paso final, desencriptamos el archivo multimedia y lo guardamos en la misma carpeta que se encontraba el original.

En concreto en el archivo **ImageEncDec.java** encontramos cuatro funciones claves:

getFile, que recibe el fichero que el usuario ha seleccionado y lee su contenido en bytes independientemente del tipo que sea (imagen, audio, video, etc.) y devuelve esos bytes o contenido para su uso posterior.

encryptPdfFile, que recibe una clave generada anteriormente y los bytes de contenido del fichero del usuario. En esencia, esta función crea un cipher para encriptación AES y lo inicializa con la clave pasada para después encriptar el contenido del fichero pasado y devolver ese contenido encriptado.

decryptPdfFile. de manera similar a la anterior función, esta recibe la clave y el contenido encriptado para crear un cipher que desencripte dicho contenido y devolverlo.

saveFile, en esta última función únicamente se coge el contenido (supuestamente sin encriptar) y se crea un archivo copia del original dado por el usuario en el mismo directorio donde se encuentre este. La copia que se cree siempre se llamará igual que el original pero añadiendo "-nuevo" al final.