Sistemas multimedia

Memoria proyecto final:

Conversor de medios multimedia

Universidad de Jaén

Autor:

Joaquín Rodríguez Ferrer - jrf00020

Índice:8

Recopilación de requisitos:

Requisitos mínimos que me planteo para el desarrollo:

- Las conversiones se podrán realizar con archivos formato audio, vídeo e imagen
- Las conversiones deben realizarse en segundo plano sin bloquear la interfaz
- Realizar el desarrollo con visión de ser utilizado a nivel escritorio
- La aplicación debe incorporar una interfaz de usuario

Requisitos opcionales:

- Extracción de audio procedente de un vídeo
- Especificar una carpeta y procesar todos los archivos que tengan
- las extensiones seleccionadas
- Aplicación de los principios básicos del diseño gráfico a la
- creación de la interfaz de usuario
- Inclusión de storyboards en la documentación

Inicialmente el desarrollo se enfocó de una forma diferente, pero debido a ciertos problemas durante el mismo, he tenido que reducir al mínimo el número de requisitos que implementar. Esto es con el objetivo de desarrollar una aplicación lo más funcional posible y cuidando cada uno de los apartados enumerados, para así evitar al mínimo la aparición de errores que puedan perjudicar la experiencia de usuario. Me habría gustado añadirle compresión de archivos, búsqueda recursiva, creación de perfiles, etc. Pero debido a las circunstancias me fue imposible.

Planificación:

He decidido que el lenguaje de programación que utilizaré será Python ya que tiene librerías actualizadas para la manipulación de archivos y son realmente sencillas de utilizar.

Estimación e itinerario:

ID	Descripción	Precedida por
A	Conversor de archivos de audio	-
B	Conversor de archivos de video	-
C	Conversor de archivos de imagen	-
D	Conversor de archivos de vídeo a audio	-
E	Conversión por carpetas	A, B, C, D
F	Creación de carpetas con el nombre deseado para almacenar archivos convertidos	A, B, C, D
G	Diseñar una interfaz gráfica para cada conversor teniendo en cuenta el UX	E, F
H	Diseñar un menú principal para conectar todos los conversores dentro de la aplicación	E, F
I	Implementar el diseño a cada conversor	G
J	Implementar el diseño del menú principal y conectar los conversores	H

Desarrollo

Para el desarrollo de la aplicación, decidí dividir la totalidad del proyecto en programas más sencillos que posteriormente uniré para realizar la aplicación final. Estos programas (o pasos) más simples se pueden ver reflejados en el apartado Estimación e itinerario.

A. Conversor de archivos de audio

El objetivo que quiero alcanzar en este apartado es que insertando la ruta de un archivo, el programa sea capaz de comprobar si, dentro de una lista predeterminada de tipos de archivo, este archivo es realmente un archivo de audio o no y en función de esto, proceder con un mensaje de error o realizar la conversión.
Utilicé diversas librerías de gestión de archivos y directorios como *os* o *glob*. Además, para la manipulación de archivos de audio importé *AudioSegment* de la librería de *pydub*.
Con todos estos componentes, fue posible realizar las conversiones de manera correcta.

B. Conversor de archivos de vídeo

El objetivo a alcanzar en esta etapa es que insertando la ruta de un archivo, el programa sea capaz de comprobar si este archivo es verdaderamente un archivo de vídeo o no y si realmente lo es, convierta el archivo a otro formato de vídeo.
Utilicé diversas librerías de gestión de archivos y directorios como *os* o *glob*. Aquí se utiliza por primera vez el framework *FFMPEG*, que nos permite realizar diversas tareas relacionadas con la multimedia de manera extremadamente sencilla.
Con todos estos componentes, fue posible realizar las conversiones de manera correcta.

C. Conversor de archivos de imagen

En este apartado, busco realizar la conversión de archivos de imagen a otros formatos de imagen.
Para esto vuelvo a utilizar librerías como *os* o *glob*, además de una librería propia para tratar archivos de imagen como es *PIL* / *Pillow.*
Este fue el conversor más complicado de desarrollar. Esto se debe a los problemas que se plantean al tratar de convertir archivos *.png* a otros tipos que no cuentan con canal *alpha* o de transparencia. Además descubrí que es imposible, para librerías de este tipo, realizar la conversión a *.jpg*, sino que se debe realizar a *.jpeg*. Todos estos aspectos han sido cubiertos correctamente tras hallar los diversos problemas en la comprobación de errores que se comenta en el apartado E.

D. Conversor de archivos de video a audio

El objetivo en este apartado es poder insertar la ruta de un archivo, que el programa sea capaz de comprobar si el archivo es de vídeo, separar la información correspondiente al audio del vídeo y que esta sea volcada dentro de un nuevo archivo.
Para esto, nuevamente, utilicé *os* y *glob*, además de *FFMPEG*.
Con todos estos componentes, fue posible realizar las conversiones de manera correcta.

E. Conversión por carpetas

Para este apartado, primero necesitaba tener todos los anteriores funcionando correctamente, así que también fue un paréntesis en el desarrollo para realizar una breve comprobación de posibles errores y fallos, de la cual pude corregir varios *bugs* que se producían sobre todo en la conversión de archivos de imagen. Tras esto, comencé con la puesta en marcha de la conversión por carpetas.
Mi idea era simple, implementar a los conversores ya creados una utilidad para poder comprobar si la dirección local que se le pasaba daba a un directorio o a un archivo y en función de esto realizar un método de conversión u otro.
Para la conversión de un directorio procedería de la siguiente manera:

1. Comprobar si la dirección introducida da a un directorio o a un archivo
2. Si es un directorio, buscar todos los archivos que coincidan con los tipos posibles del conversor en el que se esté trabajando
3. Proceder con la conversión de los archivos de manera individual

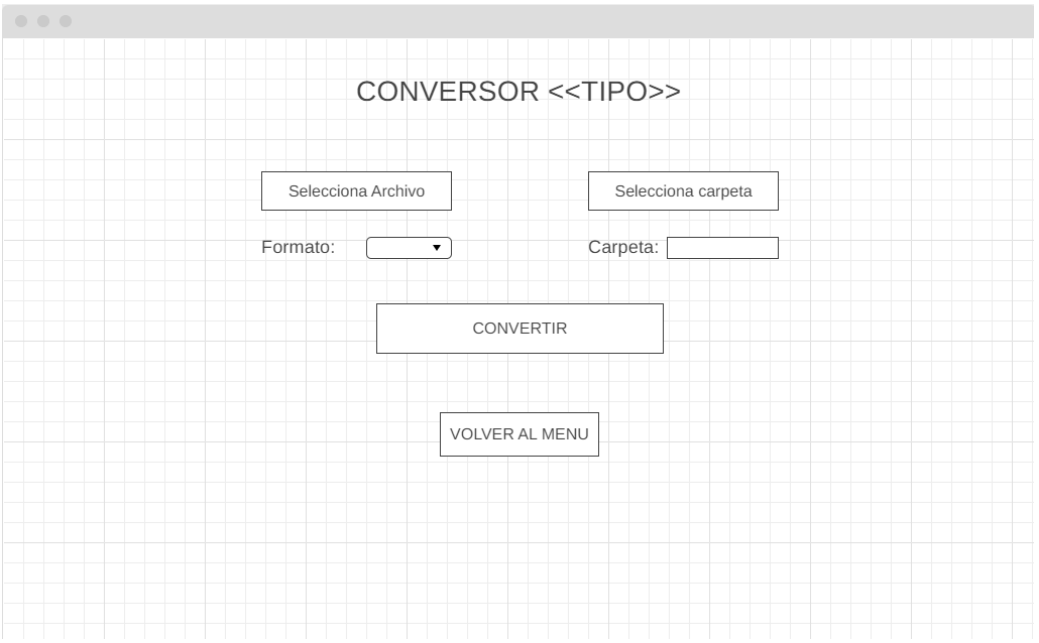
Para realizar todo esto, utilicé únicamente *os* y *glob*. En el conversor de imagen, además, tuve que utilizar *fnmatch*, ya que me estaba poniendo grandes problemas y debido a la falta de tiempo con la que contaba, me fue imposible resolverlo de otra manera más “limpia”.
En este punto, me percaté de que no había diseñado la aplicación para no bloquearse en un futuro, al estar utilizando la interfaz gráfica. Debido a esto, realicé un breve cambio en el código de cada conversor para que al encontrarse convirtiendo archivos utilizasen distintos hilos. Para esto utilicé la librería *threading*, que permite la manipulación de hilos en Python.

F. Creación de carpetas con el nombre deseado para almacenar archivos convertidos

Tras realizar diversas pruebas, me percaté de que los directorios en los que me encontraba se quedaban sumamente desordenados y con un montón de archivos los cuales me complicaban la vida para poder ver qué funcionaba y qué no, así que decidí implementar la creación de carpetas para poder almacenar los archivos que los usuarios creen. De esta manera, los archivos convertidos quedarán todos almacenados en un mismo sitio, mientras que los archivos originales no se moverán de su sitio en absoluto.
Para esto, una vez más, utilicé *os*.

G. Diseñar una interfaz gráfica para cada conversor teniendo en cuenta el UX

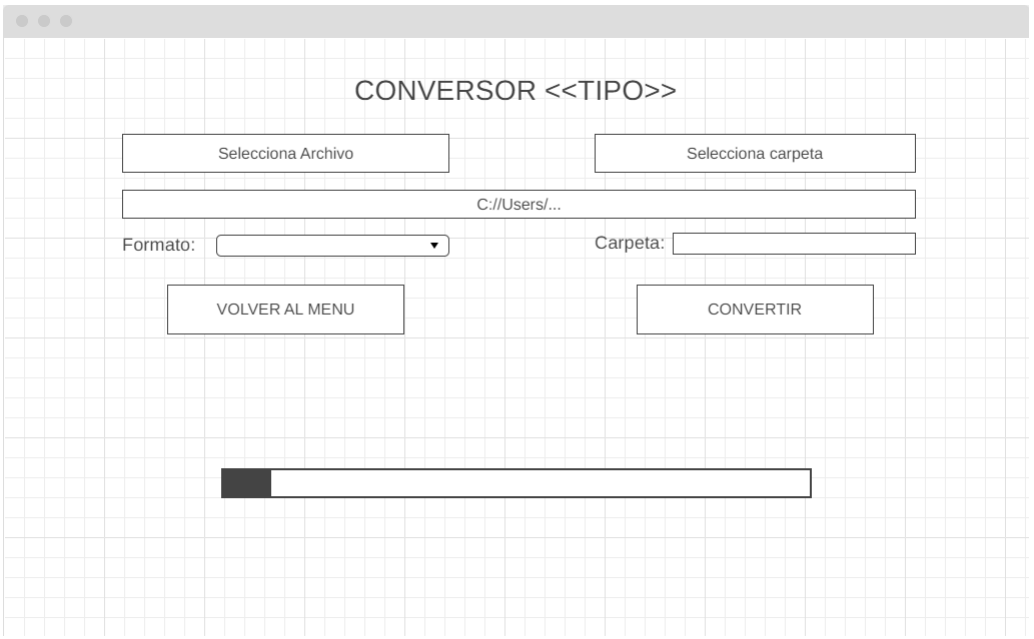
Una vez tenía todos las partes funcionales de mi aplicación en marcha, llegaba el momento de comenzar a pensar en la interfaz gráfica de la misma. Inicialmente, pensé en realizar un único diseño para todos los conversores de mi aplicación. En cada uno cambiarían ciertos apartados, obviamente, pero serían estilos similares. El diseño original es el siguiente:



Inicialmente, me parecía un buen diseño, pero me di cuenta de 2 fallos principales:

1. El usuario no tendría manera de ver que fichero o carpeta habría seleccionado
2. El usuario no tendría manera de ver el avance de la conversión

Por esto mismo, diseñé una nueva variante que, esta vez sí, sería la definitiva:

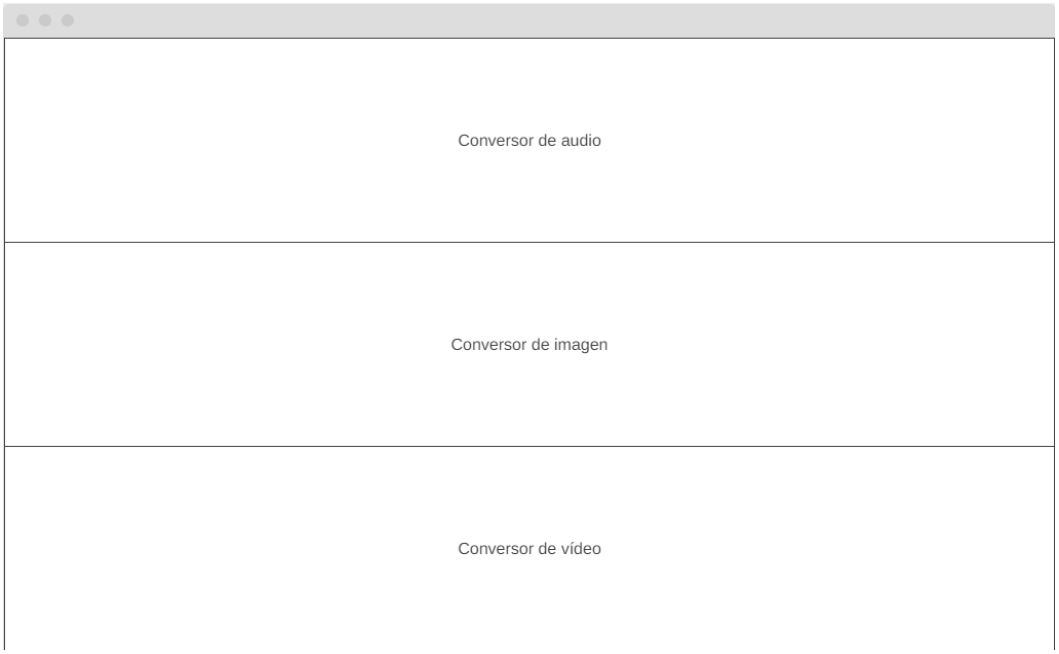


En este nuevo diseño sí que contaríamos con una forma de ver qué va a convertir el usuario, además de una barra de progreso que calculará el total de archivos que restan por

convertirse para irse rellenando. De esta forma el usuario sabrá mucho mejor qué está haciendo y el ritmo que lleva.

H. Diseñar un menú principal para conectar todos los conversores dentro de la aplicación

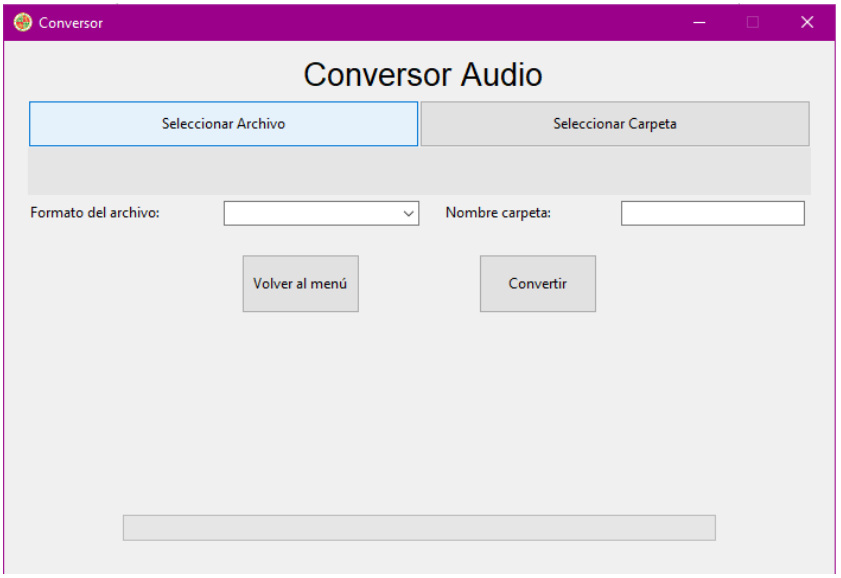
Ahora tocaba diseñar un menú principal para tener acceso a cada conversor de manera sencilla y recogida. Sería un diseño muy sencillo, para que en el caso de querer mejorar la aplicación en un futuro, me resultase sencillo:

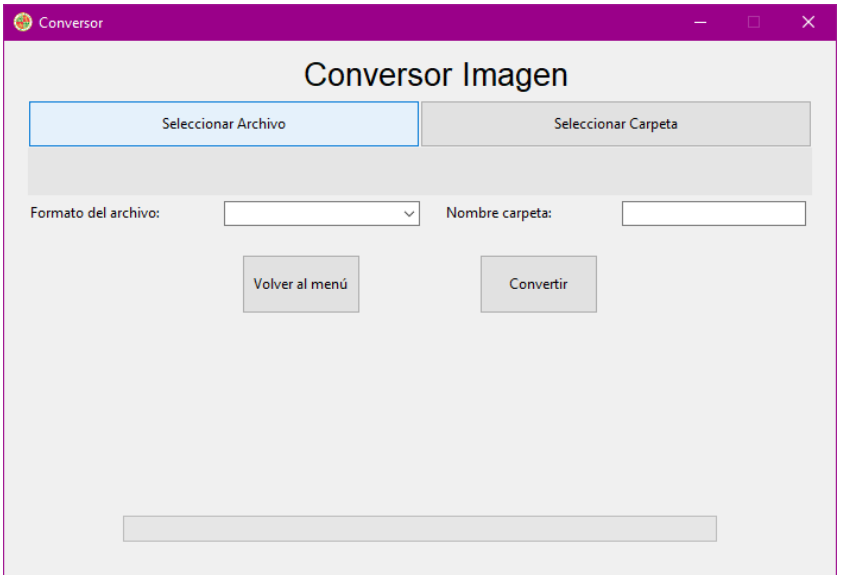


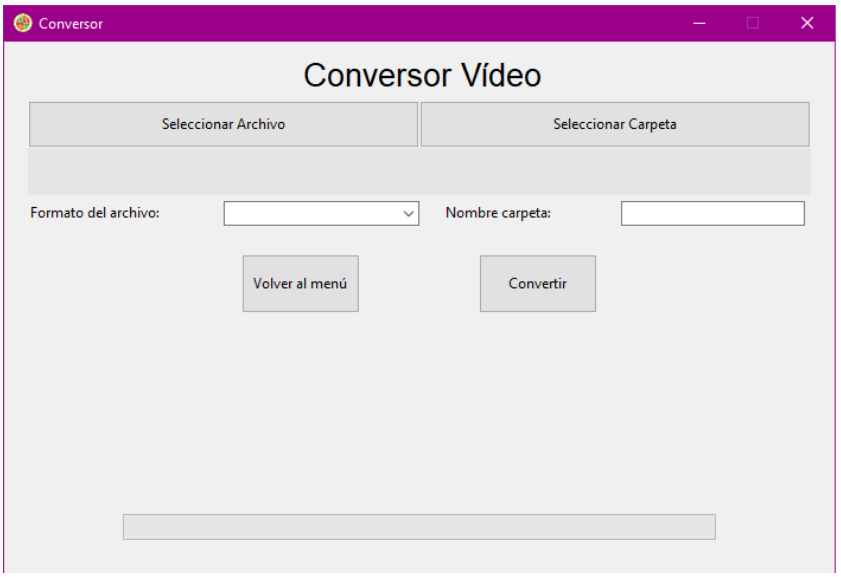
Literalmente, tres botones gigantes que permitan el acceso a cada conversor. No tiene más complicación.

I. Implementar el diseño a cada conversor

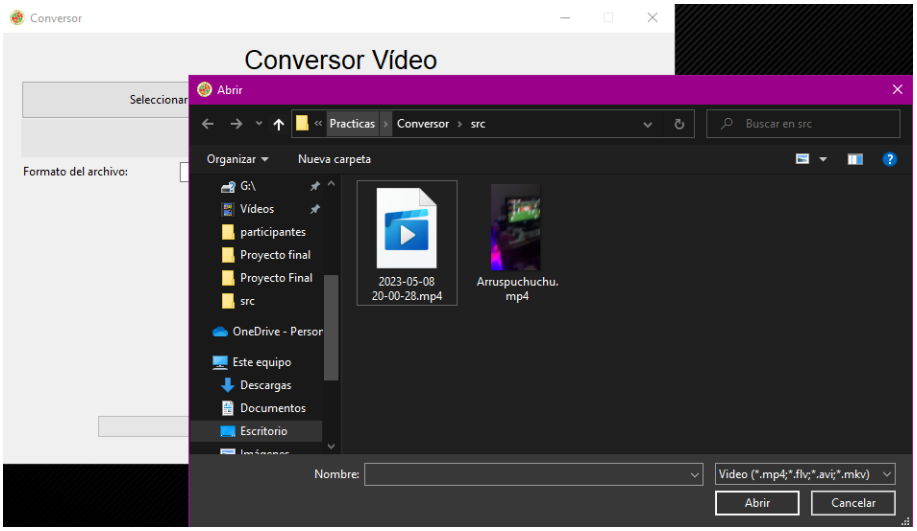
Primera fase de implementación gráfica. Tuve que investigar acerca de qué librería utilizaría para el desarrollo de la interfaz gráfica. Finalmente, me decanté por *Tkinter.* Esta le daría a mi aplicación un toque más “clásico”, que es lo que me viene a la mente al pensar en una aplicación para un conversor de archivos multimedia.
A continuación, muestro las capturas de los diferentes conversores ya creados:


Conversor
Conversor Audio
Seleccionar Archivo
Seleccionar Carpeta
Formato del archivo:
Nombre carpeta:
Volver al menú
Convertir


Conversor
Conversor Imagen
Seleccionar Archivo
Seleccionar Carpeta
Formato del archivo:
Nombre carpeta:
Volver al menú
Convertir

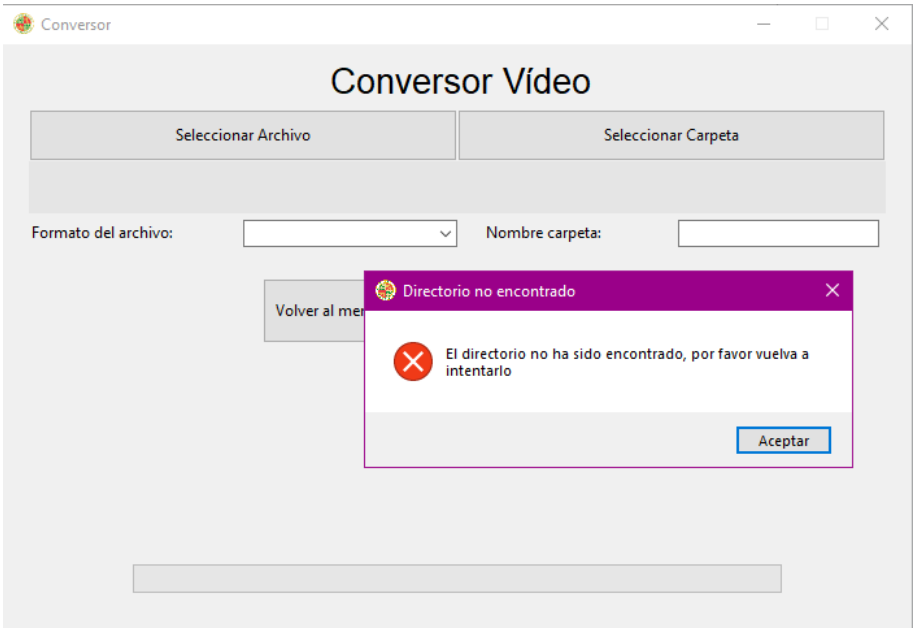

Conversor
Conversor Vídeo
Seleccionar Archivo
Seleccionar Carpeta
Formato del archivo:
Nombre carpeta:
Volver al menú
Convertir

Todos son prácticamente iguales a excepción del título, el desplegable y las opciones que se muestran al realizar la selección de un único archivo:

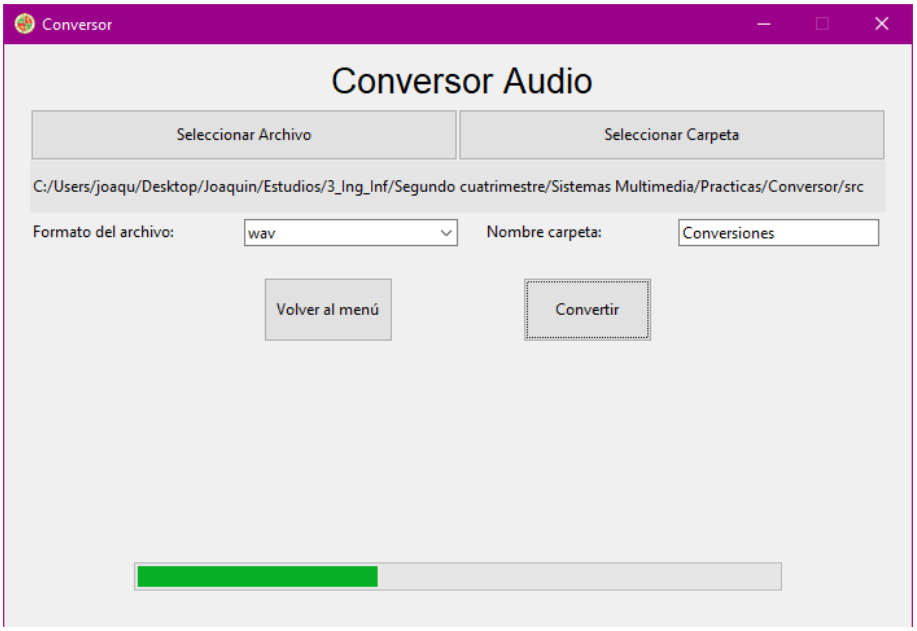


En la esquina inferior derecha, se puede apreciar el detalle de que nos está filtrando los elementos por los archivos de vídeo que la aplicación es capaz de procesar. Esto pasa exactamente igual con los otros tipos de conversores.

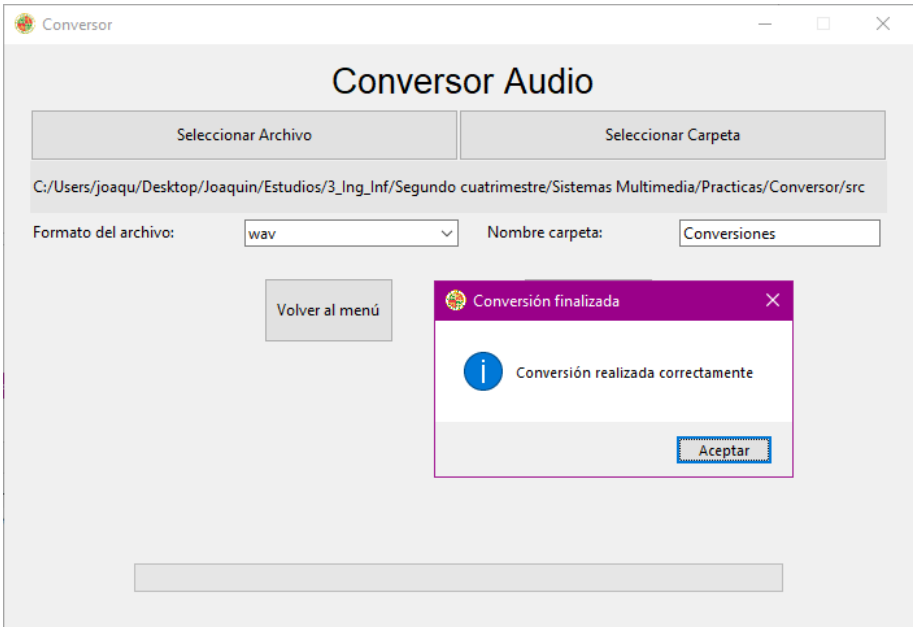
En las siguientes imágenes, se puede apreciar el funcionamiento de la barra de progreso, además de las diversas alertas que pueden aparecer durante el uso de la aplicación para avisar al usuario de los posibles errores que cometa durante su uso:



No se ha seleccionado ningún directorio o archivo



Conversión en progreso



Conversión finalizada

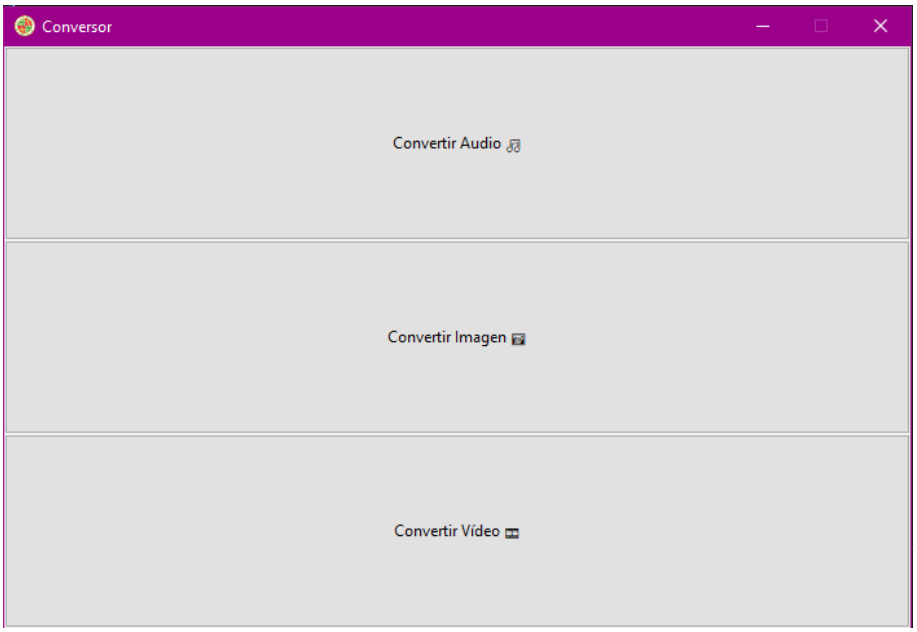
J. Implementar el diseño del menú principal y conectar los conversores

El último paso antes de tener mi aplicación terminada. Sin embargo, fue el más simple. Todo se resume a unas 30 líneas de código que me permiten clickar un botón que me enviará a cada una de las páginas de mi aplicación:

```python
class MenuFrame(ttk.Frame):
    def __init__(self, container, show_page):
        super().__init__(container)

        self.show_page = show_page

        myFont = font.Font(family='Helvetica')

        self.option1_button = ttk.Button(self, text="Convertir Audio 🎵", command=self.abrir_conversor_audio)
        self.option1_button.pack(fill=tk.BOTH, expand=True)

        self.option2_button = ttk.Button(self, text="Convertir Imagen 🖼", command=self.abrir_conversor_imagen)
        self.option2_button.pack(fill=tk.BOTH, expand=True)

        self.option3_button = ttk.Button(self, text="Convertir Vídeo 🎞", command=self.abrir_conversor_video)
        self.option3_button.pack(fill=tk.BOTH, expand=True)

    def abrir_conversor_audio(self):
        self.show_page(ConversorAudio)

    def abrir_conversor_imagen(self):
        self.show_page(ConversorImagen)

    def abrir_conversor_video(self):
        self.show_page(ConversorVideo)
```

Código del menú principal

Tras esto, mi aplicación estaba finalizada:



Cada uno de los botones llevaba correctamente a las páginas y desde cada página se podía volver correctamente al menú. La aplicación era funcional al 100%.

Librerías

- glob
- os
- PIL / pillow
- pydub
- shutil
- tkinter
- fnmatch
- threading

Requisitos

- Tener instalado FFMPEG