|  |
| --- |
| from tkinter import \* |
|  | from tkinter import filedialog |
|  | import pygame |
|  | import time |
|  | from mutagen.mp3 import MP3 |
|  | import tkinter.ttk as ttk |
|  |  |
|  | root = Tk() |
|  |  |
|  | root.title("MP3 Player") |
|  | root.geometry("500x400") |
|  |  |
|  | # Initialize Pygame |
|  | pygame.mixer.init() |
|  |  |
|  | # Create Function To Deal With Time |
|  | def play\_time(): |
|  | # Check to see if song is stopped |
|  | if stopped: |
|  | return |
|  |  |
|  | # Grab Current Song Time |
|  | current\_time = pygame.mixer.music.get\_pos() / 1000 |
|  | # Convert Song Time To Time Format |
|  | converted\_current\_time = time.strftime('%M:%S', time.gmtime(current\_time)) |
|  |  |
|  |  |
|  | # Reconstruct song with directory structure stuff |
|  | song = playlist\_box.get(ACTIVE) |
|  | song = f'C:/mp3/audio/{song}.mp3' |
|  |  |
|  | # Find Current Song Length |
|  | song\_mut = MP3(song) |
|  | global song\_length |
|  | song\_length = song\_mut.info.length |
|  | # Convert to time format |
|  | converted\_song\_length = time.strftime('%M:%S', time.gmtime(song\_length)) |
|  |  |
|  | # Check to see if song is over |
|  | if int(song\_slider.get()) == int(song\_length): |
|  | stop() |
|  |  |
|  | elif paused: |
|  | # Check to see if paused, if so - pass |
|  | pass |
|  |  |
|  | else: |
|  | # Move slider along 1 second at a time |
|  | next\_time = int(song\_slider.get()) + 1 |
|  | # Output new time value to slider, and to length of song |
|  | song\_slider.config(to=song\_length, value=next\_time) |
|  |  |
|  | # Convert Slider poition to time format |
|  | converted\_current\_time = time.strftime('%M:%S', time.gmtime(int(song\_slider.get()))) |
|  |  |
|  | # Output slider |
|  | status\_bar.config(text=f'Time Elapsed: {converted\_current\_time} of {converted\_song\_length} ') |
|  |  |
|  |  |
|  | # Add Current Time To Status Bar |
|  | if current\_time > 0: |
|  | status\_bar.config(text=f'Time Elapsed: {converted\_current\_time} of {converted\_song\_length} ') |
|  |  |
|  | # Create Loop To Check the time every second |
|  | status\_bar.after(1000, play\_time) |
|  |  |
|  | # Create Function To Add One Song To Playlist |
|  | def add\_song(): |
|  | song = filedialog.askopenfilename(initialdir='audio/', title="Choose A Song", filetypes=(("mp3 Files", "\*.mp3" ), )) |
|  | # Strip out directory structure and .mp3 from song title |
|  | song = song.replace("C:/mp3/audio/", "") |
|  | song = song.replace(".mp3", "") |
|  | # Add To End of Playlist |
|  | playlist\_box.insert(END, song) |
|  |  |
|  | # Create Function To Add Many Songs to Playlist |
|  | def add\_many\_songs(): |
|  | songs = filedialog.askopenfilenames(initialdir='audio/', title="Choose A Song", filetypes=(("mp3 Files", "\*.mp3" ), )) |
|  |  |
|  | # Loop thru song list and replace directory structure and mp3 from song name |
|  | for song in songs: |
|  | # Strip out directory structure and .mp3 from song title |
|  | song = song.replace("C:/mp3/audio/", "") |
|  | song = song.replace(".mp3", "") |
|  | # Add To End of Playlist |
|  | playlist\_box.insert(END, song) |
|  |  |
|  | # Create Function To Delete One Song From Playlist |
|  | def delete\_song(): |
|  | # Delete Highlighted Song From Playlist |
|  | playlist\_box.delete(ANCHOR) |
|  |  |
|  | # Create Function To Delete All Songs From Playlist |
|  | def delete\_all\_songs(): |
|  | # Delete ALL songs |
|  | playlist\_box.delete(0, END) |
|  |  |
|  | # Create Play Function |
|  | def play(): |
|  | # Set Stopped to False since a song is now playing |
|  | global stopped |
|  | stopped = False |
|  |  |
|  | # Reconstruct song with directory structure stuff |
|  | song = playlist\_box.get(ACTIVE) |
|  | song = f'C:/mp3/audio/{song}.mp3' |
|  |  |
|  | #Load song with pygame mixer |
|  | pygame.mixer.music.load(song) |
|  | #Play song with pygame mixer |
|  | pygame.mixer.music.play(loops=0) |
|  |  |
|  | # Get Song Time |
|  | play\_time() |
|  |  |
|  | # Create Stopped Variable |
|  | global stopped |
|  | stopped = False |
|  | def stop(): |
|  | # Stop the song |
|  | pygame.mixer.music.stop() |
|  | # Clear Playlist Bar |
|  | playlist\_box.selection\_clear(ACTIVE) |
|  |  |
|  | status\_bar.config(text='') |
|  |  |
|  | # Set our slider to zero |
|  | song\_slider.config(value=0) |
|  |  |
|  | # Set Stop Variable To True |
|  | global stopped |
|  | stopped = True |
|  |  |
|  |  |
|  |  |
|  | # Create Function To Play The Next Song |
|  | def next\_song(): |
|  | # Reset Slider position and status bar |
|  | status\_bar.config(text='') |
|  | song\_slider.config(value=0) |
|  |  |
|  | #Get current song number |
|  | next\_one = playlist\_box.curselection() |
|  | # Add One To The Current Song Number Tuple/list |
|  | next\_one = next\_one[0] + 1 |
|  |  |
|  | # Grab the song title from the playlist |
|  | song = playlist\_box.get(next\_one) |
|  | # Add directory structure stuff to the song title |
|  | song = f'C:/mp3/audio/{song}.mp3' |
|  | #Load song with pygame mixer |
|  | pygame.mixer.music.load(song) |
|  | #Play song with pygame mixer |
|  | pygame.mixer.music.play(loops=0) |
|  |  |
|  | # Clear Active Bar in Playlist |
|  | playlist\_box.selection\_clear(0, END) |
|  |  |
|  | # Move active bar to next song |
|  | playlist\_box.activate(next\_one) |
|  |  |
|  | # Set Active Bar To next song |
|  | playlist\_box.selection\_set(next\_one, last=None) |
|  |  |
|  | # Create function to play previous song |
|  | def previous\_song(): |
|  | # Reset Slider position and status bar |
|  | status\_bar.config(text='') |
|  | song\_slider.config(value=0) |
|  |  |
|  | #Get current song number |
|  | next\_one = playlist\_box.curselection() |
|  | # Add One To The Current Song Number Tuple/list |
|  | next\_one = next\_one[0] - 1 |
|  |  |
|  | # Grab the song title from the playlist |
|  | song = playlist\_box.get(next\_one) |
|  | # Add directory structure stuff to the song title |
|  | song = f'C:/mp3/audio/{song}.mp3' |
|  | #Load song with pygame mixer |
|  | pygame.mixer.music.load(song) |
|  | #Play song with pygame mixer |
|  | pygame.mixer.music.play(loops=0) |
|  |  |
|  | # Clear Active Bar in Playlist |
|  | playlist\_box.selection\_clear(0, END) |
|  |  |
|  | # Move active bar to next song |
|  | playlist\_box.activate(next\_one) |
|  |  |
|  | # Set Active Bar To next song |
|  | playlist\_box.selection\_set(next\_one, last=None) |
|  |  |
|  |  |
|  | # Create Paused Variable |
|  | global paused |
|  | paused = False |
|  |  |
|  | # Create Pause Function |
|  | def pause(is\_paused): |
|  | global paused |
|  | paused = is\_paused |
|  |  |
|  | if paused: |
|  | #Unpause |
|  | pygame.mixer.music.unpause() |
|  | paused = False |
|  | else: |
|  | #Pause |
|  | pygame.mixer.music.pause() |
|  | paused = True |
|  |  |
|  | #Create Volume Function |
|  | def volume(x): |
|  | pygame.mixer.music.set\_volume(volume\_slider.get()) |
|  |  |
|  | # Create a Slide Function For Song Positioning |
|  | def slide(x): |
|  | # Reconstruct song with directory structure stuff |
|  | song = playlist\_box.get(ACTIVE) |
|  | song = f'C:/mp3/audio/{song}.mp3' |
|  |  |
|  | #Load song with pygame mixer |
|  | pygame.mixer.music.load(song) |
|  | #Play song with pygame mixer |
|  | pygame.mixer.music.play(loops=0, start=song\_slider.get()) |
|  |  |
|  |  |
|  | # Create main Frame |
|  | main\_frame = Frame(root) |
|  | main\_frame.pack(pady=20) |
|  |  |
|  | # Create Playlist Box |
|  | playlist\_box = Listbox(main\_frame, bg="black", fg="green", width=60, selectbackground="green", selectforeground='black') |
|  | playlist\_box.grid(row=0, column=0) |
|  |  |
|  | # Create volume slider frame |
|  | volume\_frame = LabelFrame(main\_frame, text="Volume") |
|  | volume\_frame.grid(row=0, column=1, padx=20) |
|  |  |
|  | # Create Volume Slider |
|  | volume\_slider = ttk.Scale(volume\_frame, from\_=0, to=1, orient=VERTICAL, length=125, value=1, command=volume) |
|  | volume\_slider.pack(pady=10) |
|  |  |
|  | # Create Song Slider |
|  | song\_slider = ttk.Scale(main\_frame, from\_=0, to=100, orient=HORIZONTAL, length=360, value=0, command=slide) |
|  | song\_slider.grid(row=2, column=0, pady=20) |
|  |  |
|  | # Define Button Images For Controls |
|  | back\_btn\_img = PhotoImage(file='images/back50.png') |
|  | forward\_btn\_img = PhotoImage(file='images/forward50.png') |
|  | play\_btn\_img = PhotoImage(file='images/play50.png') |
|  | pause\_btn\_img = PhotoImage(file='images/pause50.png') |
|  | stop\_btn\_img = PhotoImage(file='images/stop50.png') |
|  |  |
|  |  |
|  | # Create Button Frame |
|  | control\_frame = Frame(main\_frame) |
|  | control\_frame.grid(row=1, column=0, pady=20) |
|  |  |
|  | # Create Play/Stop etc Buttons |
|  | back\_button = Button(control\_frame, image=back\_btn\_img, borderwidth=0, command=previous\_song) |
|  | forward\_button = Button(control\_frame, image=forward\_btn\_img, borderwidth=0, command=next\_song) |
|  | play\_button = Button(control\_frame, image=play\_btn\_img, borderwidth=0, command=play) |
|  | pause\_button = Button(control\_frame, image=pause\_btn\_img, borderwidth=0, command=lambda: pause(paused)) |
|  | stop\_button = Button(control\_frame, image=stop\_btn\_img, borderwidth=0, command=stop) |
|  |  |
|  | back\_button.grid(row=0, column=0, padx=10) |
|  | forward\_button.grid(row=0, column=1, padx=10) |
|  | play\_button.grid(row=0, column=2, padx=10) |
|  | pause\_button.grid(row=0, column=3, padx=10) |
|  | stop\_button.grid(row=0, column=4, padx=10) |
|  |  |
|  | # Create Main Menu |
|  | my\_menu = Menu(root) |
|  | root.config(menu=my\_menu) |
|  |  |
|  | # Create Add Song Menu Dropdows |
|  | add\_song\_menu = Menu(my\_menu, tearoff=0) |
|  | my\_menu.add\_cascade(label="Add Songs", menu=add\_song\_menu) |
|  | # Add One Song To Playlist |
|  | add\_song\_menu.add\_command(label="Add One Song To Playlist", command=add\_song) |
|  | # Add Many Songs to Playlist |
|  | add\_song\_menu.add\_command(label="Add Many Songs To Playlist", command=add\_many\_songs) |
|  |  |
|  | # Create Delete Song Menu Dropdowns |
|  | remove\_song\_menu = Menu(my\_menu, tearoff=0) |
|  | my\_menu.add\_cascade(label="Remove Songs", menu=remove\_song\_menu) |
|  | remove\_song\_menu.add\_command(label="Delete A Song From Playlist", command=delete\_song) |
|  | remove\_song\_menu.add\_command(label="Delete All Songs From Playlist", command=delete\_all\_songs) |
|  |  |
|  | # Create Status Bar |
|  | status\_bar = Label(root, text='', bd=1, relief=GROOVE, anchor=E) |
|  | status\_bar.pack(fill=X, side=BOTTOM, ipady=2) |
|  |  |
|  |  |
|  | # Temporary Label |
|  | my\_label = Label(root, text='') |
|  | my\_label.pack(pady=20) |
|  |  |
|  |  |
|  |  |
|  |  |
|  | root.mainloop() |