Report On

Music Player

Submitted in partial fulfillment of the requirements of the Mini project in

Semester IV of Second Year Computer Engineering

by

Neeraj Pal (Roll No. 12)

Amit Rane (Roll No. 16)

Anmol Shah (Roll No. 18)

Supervisor

Prof. Sweety Rupani



**University of Mumbai**

**Vidyavardhini's College of Engineering & Technology**

**Department of Computer Engineering**

****

**(2018-19)**

**Vidyavardhini's College of Engineering & Technology**

**Department of Computer Engineering**

**CERTIFICATE**

This is to certify that the project entitled "**Music Player**" is a bonafide work of " Neeraj Pal (Roll No. 12) , Amit Rane (Roll No. 16) , Anmol Shah (Roll No. 18)" submitted to the University of Mumbai in partial fulfillment of the requirement for the Mini project in semester IV of Second Year Computer Engineering.

**Supervisor**

|  |  |  |
| --- | --- | --- |
| Prof. Sweety Rupani |  |  |
| Internal Examiner |  | External Examiner |
| Dr Megha Trivedi  Head of Department |  | Dr. H.V. Vankudre  Principal |

**Abstract**

Music players are very import part of our daily life, and how user friendly or easy to use a person finds it decides whether a player is good or not .So a Music player which only plays music will not be an ideal choice. Hence we created a Music player that can perform basic functions like Play, Pause, Unpause the song, play the next and previous songs in the list.

The Music player is also able to control the volume using a volume toggle.

One of the challenges that people commonly face is to stay updated about current music trends, hence there should be a way to allow user to listen to only their favourite songs from the entire collection. To solve this problem we decided to make a playlist in which user can add or remove his/her favourite songs and listen to these favourite songs from playlist.

**Acknowledgement**

In performing our project, we had to take the help and guideline of some respected persons, who deserve our greatest gratitude. The completion of this project gives us much Pleasure. We would like to show our gratitude to **Prof. Sweety Rupani and Dr. Megha Trivedi** (Head of Department)for giving us a good guideline for project through numerous suggestions and ideas. We would also like to expand our deepest gratitude to all those who have directly and indirectly guided us in making this project a success.

Many people, especially our classmates and team members itself, have made valuable comment suggestions on this proposal which gave us an inspiration to improve our project. We thank all the people for their help directly and indirectly to complete our project and making it a great success.

**Contents**

**Pg. No**

|  |  |
| --- | --- |
| 1 Introduction | 6 |
| 2 Module Description | 7 |
| 2.1 tkinter | 7 |
| 2.2 pygame | 8 |
| 2.3 os module  2.4 mutagen | 8  9 |
| 3 Code | 10 |
| 4 Output | 19 |

5 Conclusion & References 20

**INTRODUCTION:**

To create our project “Music Player” we made use of python GUI. tkinter helped us to create python gui window on which we added variety of widgets that perform their respective task. These widgets when come together we get our final UI. Widgets like buttons, list box, scale, label etc are added.

Further os module was chosen so that it can browse our memory for a dfirectory containing all the songs, pygame module was used to play pause and resume the music and mutagen helped to extract the metadata from the song and add it to the list box.

Basic python is used to change icon of window, add a menu bar , show message box at relevant places and finally themes in tkinter was used to give an overall attractive look to entire Music player.

**MODULE DESCRIPTION:**

**TKINTER**:

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

**To create a tkinter:**

1. Importing the module – tkinter
2. Create the main window (container)
3. Add any number of widgets to the main window
4. Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the python code.

Note that the name of the module in Python 2.x is ‘Tkinter’ and in Python 3.x is ‘tkinter’.

Widgets used:

* Buttons
* Check Box
* Scale
* List Box
* Label
* Scroll Bar

**PYGAME:**

Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language.

pygame (**the library**) is a Free and Open Source python programming language library for making multimedia applications like games built on top of the excellent SDL library. Like SDL, pygame is highly portable and runs on nearly every platform and operating system. Millions of people have downloaded pygame itself, which is a whole lot of bits flying across the interwebs.

**OS MODULE:**

The OS module in python provides functions for interacting with the operating system. OS, comes under Python’s standard utility modules. This module provides a portable way of using operating system dependent functionality. The “os” and “os.path” modules include many functions to interact with the file system.

**MUTAGEN:**

Mutagen is a Python module to handle audio metadata. It supports ASF, FLAC, MP4, Monkey’s Audio, MP3, Musepack, Ogg Opus, Ogg FLAC, Ogg Speex, Ogg Theora, Ogg Vorbis, True Audio, WavPack, OptimFROG, and AIFF audio files. All versions of ID3v2 are supported, and all standard ID3v2.4 frames are parsed. It can read Xing headers to accurately calculate the bitrate and length of MP3s. ID3 and APEv2 tags can be edited regardless of audio format. It can also manipulate Ogg streams on an individual packet/page level.

Mutagen works with Python 2.7, 3.5+ (CPython and PyPy) on Linux, Windows and macOS, and has no dependencies outside the Python standard library. Mutagen is licensed under the GPL version 2 or later

**CODE:**

import os

from tkinter.filedialog import askdirectory

import pygame

from mutagen.mp3 import MP3

from mutagen.id3 import ID3

from tkinter import \*

import tkinter.messagebox

from tkinter import ttk

from ttkthemes import themed\_tk as tk

root = tk.ThemedTk()

root.get\_themes()

root.set\_theme("scidgrey")

#root.set\_theme("clam")

#root.set\_theme("scidpurple")

root.configure(bg="#cecac8")

root.geometry("650x600+300+50")

root.title('music player')

root.iconbitmap(r'D:\ostl\headphone.ico')

listofsongs = []

listofsongs1 = []

realnames = []

realnames1 = []

v = StringVar()

songlabel = ttk.Label(root,textvariable=v,width=42,relief=GROOVE)

index = 0

index1= -1

playindex=0

noofsongs=-1

def indexresult():

global playindex

cr=cb.get()

if cr==1:

playindex=0

cb=IntVar()

checkbutton=Checkbutton(root,text="play from playlist",bg="#cecac8",variable=cb,command=indexresult)

cr=cb.get()

checkbutton.place(x=450,y=310)

listbox = Listbox(root,width=42,height=17,relief=SOLID)

listbox.place(x=70,y=32)

label = Label(root,text='Music Player',relief= GROOVE,font = "Verdana 12 bold",bg="#cecac8",fg="#2c0e3a")

label.pack(fill=X)

scroll=Scrollbar(command=listbox.yview)

listbox.configure(yscrollcommand=scroll.set)

scroll.place(x=326,y=32,height=275)

listbox1 = Listbox(root,width=32,height=17,relief=SOLID)

listbox1.place(x=418,y=32)

scroll1=Scrollbar(command=listbox1.yview)

listbox1.configure(yscrollcommand=scroll1.set)

scroll1.place(x=614,y=32,height=275)

lengthlabel=Label(root,text="Total Length:--:--",font= "Airel 12 bold",bg="#cecac8",fg="#2c0e3a")

lengthlabel.place(x=455,y=500)

voldownphoto=PhotoImage(file ='D:\ostl\pvoldown.png')

voldownlabel=Label(root,image=voldownphoto,fg="#2c0e3a",bg="#cecac8")

voldownlabel.place(x=415,y=430)

volupphoto=PhotoImage(file ='D:\ostl\pvolup.png')

voluplabel=Label(root,image=volupphoto,fg="#2c0e3a",bg="#cecac8")

voluplabel.place(x=605,y=430)

def about():

tkinter.messagebox.showinfo('About Project','\t\t Title = Music player\n\nMade by Amit Rane Anmol Shah and Neeraj Pal.\n\nOur OSTL project uses tkinter, pygame, mutagen and OS module for creating a GUI based audio player.\n\nThis audio player can play, pause and unpause songs .\n\nIt also has options for playing next and previous songs. \n\nApart from these basic features the player can also create a playlist. ')

def directorychooser():

directory = askdirectory()

os.chdir(directory)

global noofsongs

for files in os.listdir(directory):

if files.endswith(".mp3"):

realdir = os.path.realpath(files)

audio = ID3(realdir)

realnames.append(audio['TIT2'].text[0])

listofsongs.append(files)

pygame.mixer.init()

pygame.mixer.music.load(listofsongs[0])

realnames.reverse()

for items in realnames:

listbox.insert(0,items)

noofsongs += 1

realnames.reverse()

menubar=Menu(root)

root.config(menu=menubar)

subMenu=Menu(menubar, tearoff=0)

subMenu1=Menu(menubar, tearoff=0)

menubar.add\_cascade(label="File", menu=subMenu)

subMenu.add\_command(label="Browse",command=directorychooser)

subMenu.add\_command(label="Exit",command=root.destroy)

menubar.add\_cascade(label="About", menu=subMenu1)

subMenu1.add\_command(label="About project",command=about)

pygame.mixer.init()

def updatelabel():

cr=cb.get()

if(cr==1):

global playindex

global songname

v.set(realnames1[playindex])

songlabel.place(x=70,y=315)

else:

global index

global songname

v.set(realnames[index])

songlabel.place(x=70,y=315)

def playsong():

cr=cb.get()

if (cr==1):

try:

global playindex

pygame.mixer.music.load(listofsongs1[playindex])

pygame.mixer.music.play()

lengthdetail()

updatelabel()

except:

tkinter.messagebox.showerror('File not selected','Open file and browse for song.')

else:

try:

global index

pygame.mixer.music.load(listofsongs[index])

pygame.mixer.music.play()

lengthdetail()

updatelabel()

except:

tkinter.messagebox.showerror('File not selected','Open file and browse for song.')

def nextsong():

cr=cb.get()

if (cr==1):

try:

global playindex

playindex += 1

if playindex>index1:

playindex=0

pygame.mixer.music.load(listofsongs1[playindex])

pygame.mixer.music.play()

lengthdetail()

updatelabel()

except:

tkinter.messagebox.showerror('File not selected','Open file and browse for song.')

else:

try:

global index

global noofsongs

index += 1

if index > noofsongs:

index=0

pygame.mixer.music.load(listofsongs[index])

pygame.mixer.music.play()

lengthdetail()

updatelabel()

except:

tkinter.messagebox.showerror('File not selected','Open file and browse for song.')

def prevsong():

cr=cb.get()

if(cr==1):

try:

global playindex

playindex -= 1

if playindex < 0:

playindex=index1

pygame.mixer.music.load(listofsongs1[playindex])

pygame.mixer.music.play()

lengthdetail()

updatelabel()

except:

tkinter.messagebox.showerror('File not selected','Open file and browse for song.')

else:

try:

global index

global noofsongs

index -= 1

if index < 0:

index=noofsongs

pygame.mixer.music.load(listofsongs[index])

pygame.mixer.music.play()

lengthdetail()

updatelabel()

except:

tkinter.messagebox.showerror('File not selected','Open file and browse for song.')

def pausesong():

try:

pygame.mixer.music.pause()

global paused

paused=1

v.set("")

except:

tkinter.messagebox.showerror('File not selected','Open file and browse for song.')

def unpausesong():

try:

global index

pygame.mixer.music.unpause()

global paused

paused=0

updatelabel()

except:

tkinter.messagebox.showerror('File not selected','Open file and browse for song.')

def setvol(val):

volume=float(val)/100

pygame.mixer.music.set\_volume(volume)

def lengthdetail():

cr=cb.get()

if(cr==1):

audio=MP3(listofsongs1[playindex])

total\_length=audio.info.length

min,sec=divmod(total\_length,60)

min=round(min)

sec=round(sec)

timeformat='{:02d}:{:02d}'.format(min,sec)

lengthlabel['text'] = "Total Length:"+timeformat

else:

audio=MP3(listofsongs[index])

total\_length=audio.info.length

min,sec=divmod(total\_length,60)

min=round(min)

sec=round(sec)

timeformat='{:02d}:{:02d}'.format(min,sec)

lengthlabel['text'] = "Total Length:"+timeformat

def addsong():

global index1

index1+=1

listofsongs1.append(listofsongs[index])

realnames1.append(realnames[index])

listbox1.insert(index1,realnames1[index1])

def removesong():

global index1

listbox1.delete(index1)

del listofsongs1[index1]

del realnames1[index1]

index1-=1

playphoto=PhotoImage(file = 'D:\ostl\play.png')

playbutton = ttk.Button(root,image=playphoto,command=playsong)

playbutton.place(x=45,y=345)

nextphoto=PhotoImage(file = 'D:\ostl\pnext1.png')

nextbutton = ttk.Button(root, image = nextphoto,command=nextsong)

nextbutton.place(x=225,y=450)

previousphoto=PhotoImage(file = 'D:\ostl\previous1.png')

previousbutton = ttk.Button(root,image=previousphoto,command=prevsong)

previousbutton.place(x=105,y=450)

pausephoto=PhotoImage(file = 'D:\ostl\pause.png')

pausebutton = ttk.Button(root,image=pausephoto,command=pausesong)

pausebutton.place(x=165,y=345)

unpausephoto=PhotoImage(file ='D:\ostl\punpause.png')

unpausebutton=ttk.Button(root,image=unpausephoto,command=unpausesong)

unpausebutton.place(x=285,y=345)

scale=ttk.Scale(root,from\_=0,to=100,orient=HORIZONTAL,length=158,command=setvol)

scale.set(70)

pygame.mixer.music.set\_volume(0.7)

scale.place(x=440,y=430)

addbutton=Button(root,text="ADD",width=8,font="Airel 14 bold",relief=SOLID,fg="#2c0e3a",command=addsong)

addbutton.place(x=415,y=360)

removebutton=Button(root,text="REMOVE",width=8,font="Airel 14 bold",relief=SOLID,fg="#2c0e3a",command=removesong)

removebutton.place(x=530,y=360)

def playonenter(event):

global play\_label

play\_label=Label(text="Play Button",font="Verdana 8")

play\_label.place(x=48,y=414)

def playonleave(event):

play\_label.destroy()

def nextonenter(event):

global next\_label

next\_label=Label(text="Next Button",font="Verdana 8")

next\_label.place(x=226,y=516)

def nextonleave(event):

next\_label.destroy()

def prevsonenter(event):

global prev\_label

prev\_label=Label(text="Previous Button",font="Verdana 8")

prev\_label.place(x=94,y=516)

def prevsonleave(event):

prev\_label.destroy()

def pauseonenter(event):

global pause\_label

pause\_label=Label(text="Pause Button",font="Verdana 8")

pause\_label.place(x=164,y=414)

def pauseonleave(event):

pause\_label.destroy()

def unpauseonenter(event):

global unpause\_label

unpause\_label=Label(text="Unpause Button",font="Verdana 8")

unpause\_label.place(x=277,y=412)

def unpauseonleave(event):

unpause\_label.destroy()

playbutton.bind("<Enter>",playonenter)

playbutton.bind("<Leave>",playonleave)

nextbutton.bind("<Enter>",nextonenter)

nextbutton.bind("<Leave>",nextonleave)

previousbutton.bind("<Enter>",prevsonenter)

previousbutton.bind("<Leave>",prevsonleave)

pausebutton.bind("<Enter>",pauseonenter)

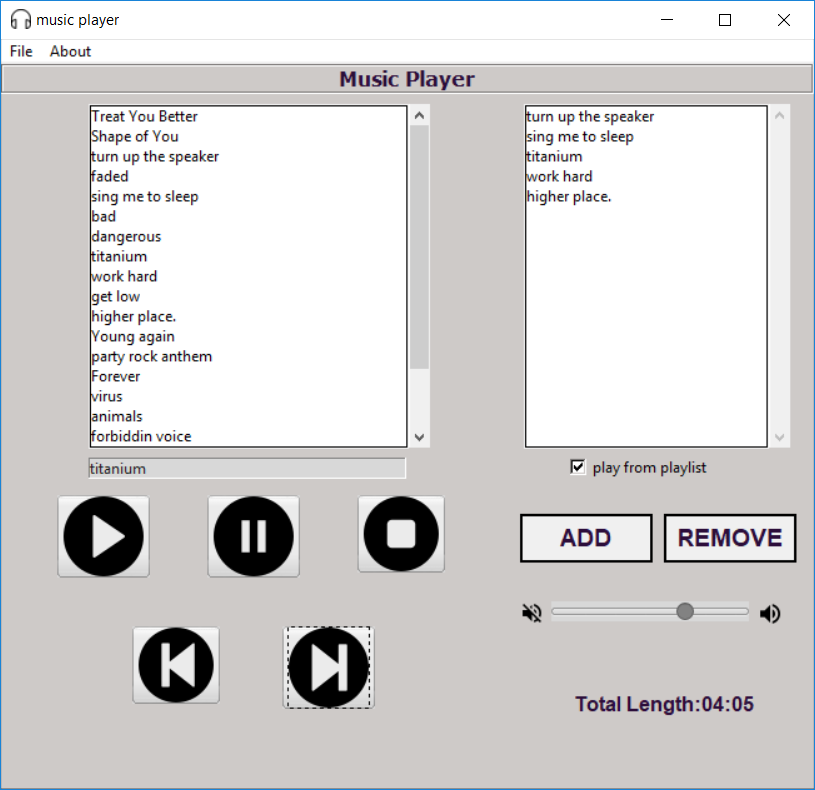
pausebutton.bind("<Leave>",pauseonleave)

unpausebutton.bind("<Enter>",unpauseonenter)

unpausebutton.bind("<Leave>",unpauseonleave)

root.mainloop()

**OUTPUT:**



**CONCLUSION:**

The Music Player can browse any directory and load the songs in the list.

We are able to perform operations like Play, Pause and Unpause the song successfully along with this we can play the Next and the Previous song by using the next and previous buttons. The volume of the song can also be controlled easily using the volume toggle and also the length of the current song is displayed for users help.

The Music Player also gives user ability to create their own playlist by adding or removing the songs using their respective buttons. The user can tick the play from playlist button to listen the songs from the playlist.

Hence we are able to implement the Music player successfully.

**REFERENCE:**

https://geeksforgeeks.org/

<https://www.tutorialspoint.com/>

<https://www.pygame.org/docs/ref/mixer.html>