MINI PROJECT

(2020-21)

Music Player Using Python GUI

Project Report

Department of Computer Engineering & Applications

Institute of Engineering & Technology

Supervised by:
Mr. Piyush Vashisth
(Assistant Professor)

Submitted by:
Parth Maheshwari(181500456)

Acknowledgment

It gives us a great sense of pleasure to present the synopsis of the B.Tech Mini Project (Music Player) undertaken during B.Tech IIIrd Year. This project in itself is going to be an acknowledgement to the inspiration, drive and technical assistance will be contributed to it by many individuals.

We owe special debt of gratitude to **Mr. Piyush Vashisth**, Assistant Professor Department of CEA, for providing us with an encouraging platform to develop this project, which thus helped us in shaping our abilities towards a constructive goal and for his constant support and guidance to our work. His sincerity, thoroughness and perseverance is been a constant source of inspiration for us. We believe that he will shower us with all his extensively experienced ideas and insightful comments at different stages of the project & also taught us about the latest industry-oriented technologies.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and co-operation.

TABLE OF CONTENTS

INTRODUCTION

Listening to music is a hobby of almost every person you meet around daily, for playing this music we need to have installed a music player in our device, each and every operating system whether it is Windows, Linux, Mac or even Android, Apple IOS also consist of a music player for playing your favorite songs.

So In this Project, we will be learning how to create a music player from scratch using the Python Programming Language.

PYTHON

Python is a high-level, general-purpose and a very popular programming language. Python programming language (latest Python 3) is being used in web development, Machine Learning applications, along with all cutting edge technology in Software Industry. Python Programming Language is very well suited for Beginners, also for experienced programmers with other programming languages like C++ and Java.

Python GUI

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

Some Best Python GUI

1. Kivy

Kivy is an OpenGL ES 2 accelerated framework for the creation of new user interfaces. It supports multiple platforms namely Windows, MacOSX, Linux, Android iOS and Raspberry Pi. It is open source and comes with over 20 widgets in its toolkit.

2.PyQT

PyQT is one of the favoured cross-platform Python bindings implementing the Qt library for the Qt (owned by Nokia) application development framework. Currently, PyQT is available for Unix/Linux, Windows, Mac OS X and Sharp Zaurus. It combines the best of Python and Qt and it up to the programmer to decide whether to create a program by coding or using Qt Designer to create visual dialogs.

3. Tkinter

Tkinter is commonly bundled with Python, using Tk and is Python's standard GUI framework. It is popular for its simplicity and graphical user interface. It is open source and available under the Python License.

4. WxPython

WxPython is an open source wrapper for cross-platform GUI library WxWidgets (earlier known as WxWindows) and implemented as a Python extension module. With WxPython you as a developer can create native applications for Windows, Mac OS and Unix.

If you're just beginning to develop applications in WxPython, here is a good simple tutorial you can go through.

5. PyGUI

PyGUI is a graphical application cross-platform framework for Unix, Macintosh and Windows. Compared to some other GUI frameworks, PyGUI is by far the simplest and lightweight of them all, as the API is purely in sync with Python.PyGUI inserts very less code between the GUI platform and Python application, hence the display of the application usually displays the natural GUI of the platform.

6 . PySide

PySide is a free and cross-platform GUI toolkit Qt initiated and sponsored by Nokia, Qt is a UI framework and a cross-platform application. PySide currently supports Linux/X11, Mac OS X, Maemo and Windows and, support for Android is in the plans for the near future.PySide provides tools to works with multimedia, XML documents, network, databases and GUI. A key feature of PySide is its API compatibility with PyQt4, so if you wish to migrate to PySide then the process will be hassle-free.

TKINTER GUI

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

Creating a GUI using tkinter is an easy task.

All you need to do is perform the following steps –

- Import the *Tkinter* module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

Pygame

Pygame is a Python module that works with computer graphics and sound libraries and designed with the power of playing with different multimedia formats like audio, video, etc. While creating our Music Player application, we will be using Pygame's mixer.music module for providing different functionality to our music player application that is usually related to the manipulation of the song tracks.

Command used to install pygame is:

```
pip install pygame
```

This module contains classes for loading Sound objects and controlling playback. There are basically four steps in order to do so:

1. Starting the mixer

```
mixer.init()
```

2. Loading the song.

```
mixer.music.load("song.mp3")
```

3. Setting the volume.

```
mixer.music.set_volume(0.7)
```

4.Start playing the song.

```
mixer.music.play()
```

The tkinter.ttk module provides access to the Tk themed widget set, introduced in Tk 8.5. If Python has not
The tkinter.ttk module provides access to the Tk themed widget set, introduced in Tk 8.5. If Python has n
The basic idea for tkinter.ttk is to separate, to the extent possible, the code implementing a widget's beha

TTK Module

The tkinter.ttk module provides access to the Tk themed widget set, introduced in Tk 8.5. If Python has not been compiled against Tk 8.5, this module can still be accessed if Tile has been installed. The former method using Tk 8.5 provides additional benefits including anti-aliased font rendering under X11 and window transparency (requiring a composition window manager on X11).

The basic idea for tkinter.ttk is to separate, to the extent possible, the code implementing a widget's behavior from the code implementing its appearance.

In our project we use TTK module for implementing slider for our Music Player which is use to navigate to which ever position we want our Music to be.

Mutagen Module

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.

Pygame itself doesn't allow to displays duration/length of particular music file.
So to calculate the length of a song we use Mutagen module.

Hardware and Software Requirements

a) Hardware (minimum):

- External Hard Drives for Backup
- Internet connection
- 4GB RAM
- Hard disk
- i3 Processor (6^{th} gen)
- Minimum resolution :1024x765 Display

b) Software:

- Pycharm
- Install Tkinter in Pycharm
- Install Pygame in Pycharm

Screenshots

```
from tkinter import *
import pygame
from tkinter import filedialog
import time
from mutagen.mp3 import MP3
import tkinter.ttk as ttk
```

Fig 1 -Importing of modules and packages

```
root = Tk()
root.title("Music")
root.iconbitmap("music-2-64.ico")
root.geometry("500x500")

#adding background image
bg = PhotoImage(file="bg3.png")
label1= Label(root,image = bg)
label1.place(x=0, y= 0)
```

Fig 2- Initialising Tkinter Window

```python
#add song fucntion
def add_song():
    song = filedialog.askopenfilename(initialdir='Music/', title="Choose a song ", filetypes=(("mp3 files", "*.mp3"),))

    #replacing full path of song to just name
    song = song.replace("C:/Users/parth/PycharmProjects/Tkinter/Music/","")
    song = song.replace(".mp3","")

    song_box.insert(END, song)
```

Fig 3-Adding songs to Tkinter Window

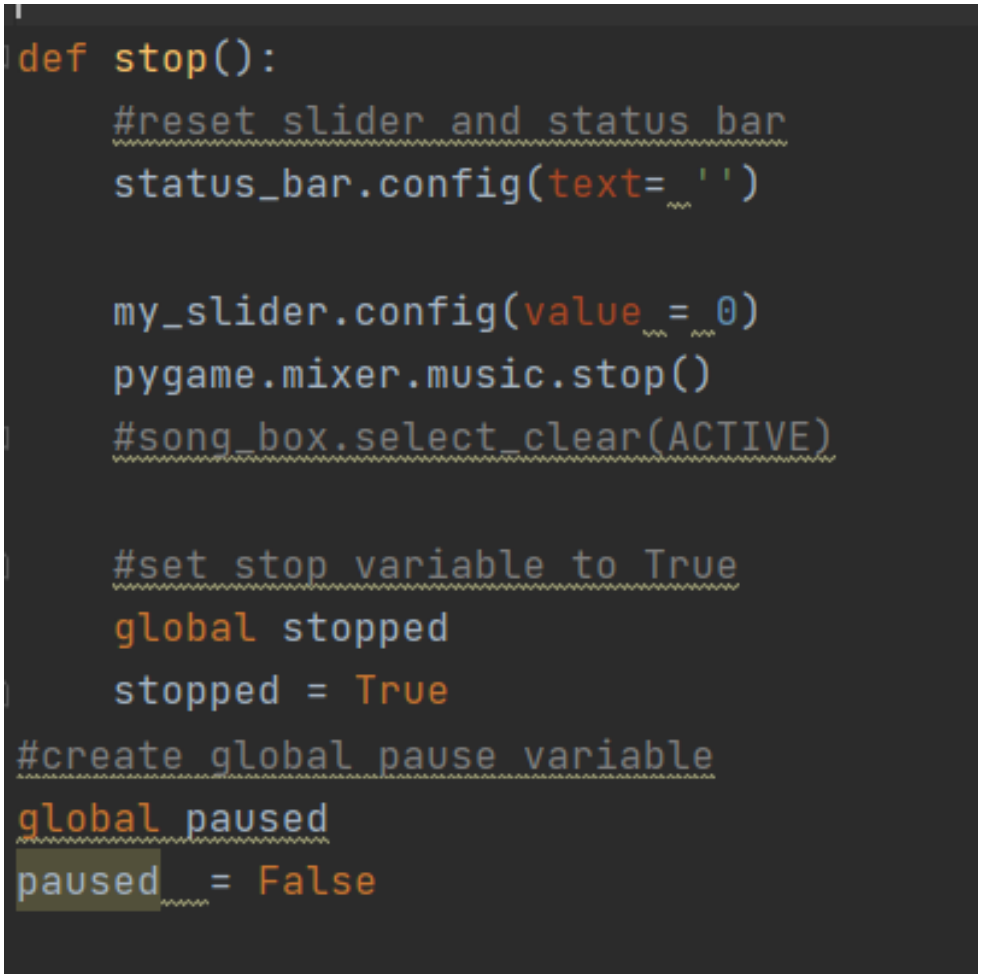
```
def play():
    #set stopped var to false so song can play
    global stopped
    stopped = False
    # gets whatever is highlighted in the list box
    song = song_box.get(ACTIVE)
    song = f'C:/Users/parth/PycharmProjects/Tkinter/Music/{song}.mp3'

    pygame.mixer.music.load(song)
    pygame.mixer.music.play(loops= 0)

    #call playtime fucntion to get time length of songs
    play_time()
    current_volume = pygame.mixer.music.get_volume()
    #slider_label.config(text=current_volume * 100)

    """#update slider to position
    slider_posistion = int(song_length)
    my_slider.config(to=slider_posistion , value = )
"""
#song stop function
global stopped
stopped = False
```

Fig 4- Play function



```
def stop():
    #reset slider and status bar
    status_bar.config(text='')

    my_slider.config(value = 0)
    pygame.mixer.music.stop()
    #song_box.select_clear(ACTIVE)

    #set stop variable to True
    global stopped
    stopped = True
#create global pause variable
global paused
paused  = False
```

Fig 5 - Stop function

```
#PLayer Control Buttons
back_btn = Button(controls_frame, image=back_btn_img, borderwidth=0,command=previous_song)
forward_btn =Button(controls_frame, image=forward_btn_img, borderwidth=0,command=next_song)
play_btn =Button(controls_frame, image=play_btn_img, borderwidth=0,command=play)
pause_btn =Button(controls_frame, image=pause_btn_img, borderwidth=0,command =lambda : pause(paused))
stop_btn =Button(controls_frame, image=stop_btn_img,borderwidth=0,command=stop)

back_btn.grid(row=0,column=0,padx=10,pady=10)
forward_btn.grid(row=0,column=1,padx=10,pady=10)
play_btn.grid(row=0,column=2,padx=10,pady=10)
pause_btn.grid(row=0,column=3,padx=10,pady=10)
stop_btn.grid(row=0,column=4,padx=10,pady=10)
```

Fig 6 - Button Creation

- **REFERENCES**
  - **Website References**
    - **GeekForGeeks**
    - **StudyTonight.com**
    - **Youtube**
  - **Faculty Guidelines**
    - Mr. Piyush Vashisth