Complete the code for the Tkinter application in the resulting project folder so that the application plays sound files in the **./music** folder and can load and save to the **playlist.m3u** playlist file (also in the **./music** folder). Such simple playlist files contain text where each line is the filename of a song, except for the first line which must only be
`#EXTM3U`

Let's say my **./music** folder file contains the files:

```
.config
.DS_Store
hello.ogg
Mixing.wav
Heya.mp3
```

The playlist.m3u file is a text file (we write strings into it and read strings from it) but its file name ends with **.m3u** so media players like VLC, iTunes, and Windows Media Player, know it is a playlist they can understand.
Since you will be using text data do not use **pickling.**
Ignore files whose names start with "." and those that are not .mp3, .ogg, or .wav files.

Follow the directions in the **TODO** comments in the code underneath.
The final product must look and work just like the following demonstration video:
(https://youtu.be/FtGOolVxrGE)

musicplayer.py code to be completed:

```
# Importing Required Modules & libraries
from tkinter import *
from tkinter.ttk import *
import pygame
import os

DEBUG = True

class MusicPlayer:
    """One object of this class represents a tkinter GUI
application that plays
    audio files and can write and read a .m3u playlist."""

    def __init__(self, root):
        """TODO: Creates a tkinter GUI application that plays
audio files and
        can write and read a .m3u playlist."""
        self.playlistfilename = 'playlist.m3u'
        self.root = root
        self.root.title("Music Player")
        self.root.geometry("1000x200+200+200")
```

```
        pygame.init()
        pygame.mixer.init()
        self.track = StringVar()
        self.status = StringVar()

        # Creating trackframe for songtrack label & trackstatus
label
        trackframe = LabelFrame(self.root, text="Song Track",
relief=GROOVE)
        trackframe.place(x=0, y=0, width=600, height=100)
        songtrack = Label(trackframe,
textvariable=self.track).grid(
            row=0, column=0, padx=10, pady=5)
        trackstatus = Label(trackframe,
textvariable=self.status).grid(
            row=0, column=1, padx=10, pady=5)

        # Creating buttonframe
        buttonframe = LabelFrame(self.root, text="Control
Panel", relief=GROOVE)
        # Inserting song control Buttons
        buttonframe.place(x=0, y=100, width=600, height=100)
        Button(buttonframe, text="Play",
command=self.playsong).grid(
            row=0, column=0, padx=10, pady=5)
        Button(buttonframe, text="Pause", command=self.pausesong
                        ).grid(row=0, column=1, padx=10,
pady=5)
        Button(buttonframe, text="Unpause",
command=self.unpausesong
                        ).grid(row=0, column=2, padx=10,
pady=5)
        Button(buttonframe, text="Stop",
command=self.stopsong).grid(
            row=0, column=3, padx=10, pady=5)
        # TODO: Insert playlist control Buttons

        # Creating songsframe
        songsframe = LabelFrame(self.root, text="Song Playlist",
relief=GROOVE)
        songsframe.place(x=600, y=0, width=400, height=200)
        scrol_y = Scrollbar(songsframe, orient=VERTICAL)
        self.playlist = Listbox(songsframe,
yscrollcommand=scrol_y.set,
                                selectbackground="gold",
                                selectmode=SINGLE,
relief=GROOVE)
```

```
        # Applying Scrollbar to playlist Listbox
        scrol_y.pack(side=RIGHT, fill=Y)
        scrol_y.config(command=self.playlist.yview)
        self.playlist.pack(fill=BOTH)

        # Changing directory for fetching songs
        os.chdir("./music")
        # Inserting songs into playlist
        self.refresh()

    def playsong(self):
        """Displays selected song and its playing status and
plays the song."""
        self.track.set(self.playlist.get(ACTIVE))
        self.status.set("-Playing")
        pygame.mixer.music.load(self.playlist.get(ACTIVE))
        pygame.mixer.music.play()

    def stopsong(self):
        """Displays stopped status and stops the song."""
        self.status.set("-Stopped")
        pygame.mixer.music.stop()

    def pausesong(self):
        """Displays the paused status and pauses the song."""
        self.status.set("-Paused")
        pygame.mixer.music.pause()

    def unpausesong(self):
        """Displays the playing status and unpauses the song."""
        self.status.set("-Playing")
        pygame.mixer.music.unpause()

    def removesong(self):
        """Deletes the active song from the playlist."""
        self.playlist.delete(ACTIVE)

    def loadplaylist(self):
        """
        TODO: Clears the current playlist and loads a previously
saved playlist
        from the music folder. A user firendly message is
appended to the status
        if a FileNotFoundError is caught(see the demo video).
        All other exception messages are
        appened to the status in their default string form.
        Ignore the lines that start with #.
```

```
        """

    def saveplaylist(self):
        """TODO: Save the current playlist to the playlist file
in the music
        folder. All exception messages are appened to the status
in their
        default string form.
        Make sure the first line of the file is only:
        #EXTM3U
        """

    def refresh(self):
        """
        TODO:
        Clears the current playlist and fills it with all valid
sound files
        from the music folder. All exception messages are
appened to the status
        in their default string form.
        Insert items into a tkinter
        Listbox."""


def main():
    """Create main window and start a MusicPlayer application on
it."""
    # Creating TK root window
    root = Tk()
    # Passing root to the MusicPlayer constructor
    app = MusicPlayer(root)
    # Start the main GUI loop
    root.mainloop()


if __name__ == "__main__":
    main()
```