Python Object-Oriented Program with Libraries

Unit 4: PyGame Tutorial

CHAPTER 5: SOUND AND MUSIC

DR. ERIC CHOU

IEEE SENIOR MEMBER

Objectives

- Play Sound and Music
- Mediainfo: Extracting the meta data from a media file.
- A simple but complete music player and programming interface.

Play Sounds

LECTURE 1

Play Sounds

- Playing sounds that are stored in sound files is even simpler than displaying images from image files. First, you must create a **`pygame.mixer.Sound`** object (which we will call Sound objects for short) by calling the **`pygame.mixer.Sound()`** constructor function. It takes one string parameter, which is the filename of the sound file. Pygame can load WAV, MP3, or OGG files.

Play Sounds

- To play this sound, call the Sound object's play() method. If you want to immediately stop the Sound object from playing call the stop() method. The stop() method has no arguments. Here is some sample code:

```
import pygame
import time
#pygame.mixer.pre_init(44100, 16, 2, 4096)
#frequency, size, channels, buffersize
pygame.init() #turn all of pygame on.
pygame.mixer.init()
bp = pygame.mixer.Sound('beeps.wav')
bp.play()
time.sleep(2) # wait and let the sound play for 1 second
bp.stop()
pygame.quit()
```

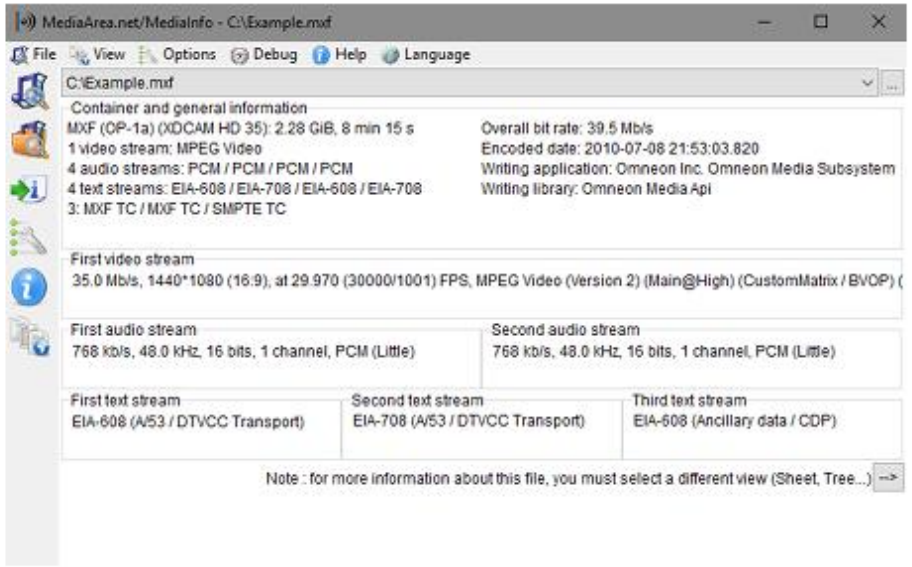
Play Sounds

- The program execution continues immediately after play() is called; it does not wait for the sound to finish playing before moving on to the next line of code.
- The Sound objects are good for sound effects to play when the player takes damage, slashes a sword, or collects a coin. But your games might also be better if they had background music playing regardless of what was going on in the game. Pygame can only load one music file to play in the background at a time. To load a background music file, call the pygame.mixer.music.load() function and pass it a string argument of the sound file to load. This file can be WAV, MP3, or MIDI format.

Play Sounds

- To begin playing the loaded sound file as the background music, call the pygame.mixer.music.play(-1, 0.0) function. The -1 argument makes the background music forever loop when it reaches the end of the sound file. If you set it to an integer 0 or larger, then the music will only loop that number of times instead of looping forever. The 0.0 means to start playing the sound file from the beginning. If you pass a larger integer or float, the music will begin playing that many seconds into the sound file. For example, if you pass 13.5 for the second parameter, the sound file with begin playing at the point 13.5 seconds in from the beginning.
- To stop playing the background music immediately, call the pygame.mixer.music.stop() function. This function has no arguments.

MediaArea MediaInfo Projects Support Us! Log in



MediaInfo

MediaInfo is a convenient unified display of the most relevant technical and tag data fo video and audio files.

Download MediaInfo

Version 21.03, Graphical User Interface with installer, for Windows

Other versions (packaging, OS, interface...) are also available ( )

See change log

You can also use MediaInfoOnline to test MediaInfo without installing any software on your computer.

Please **donate** to support further development

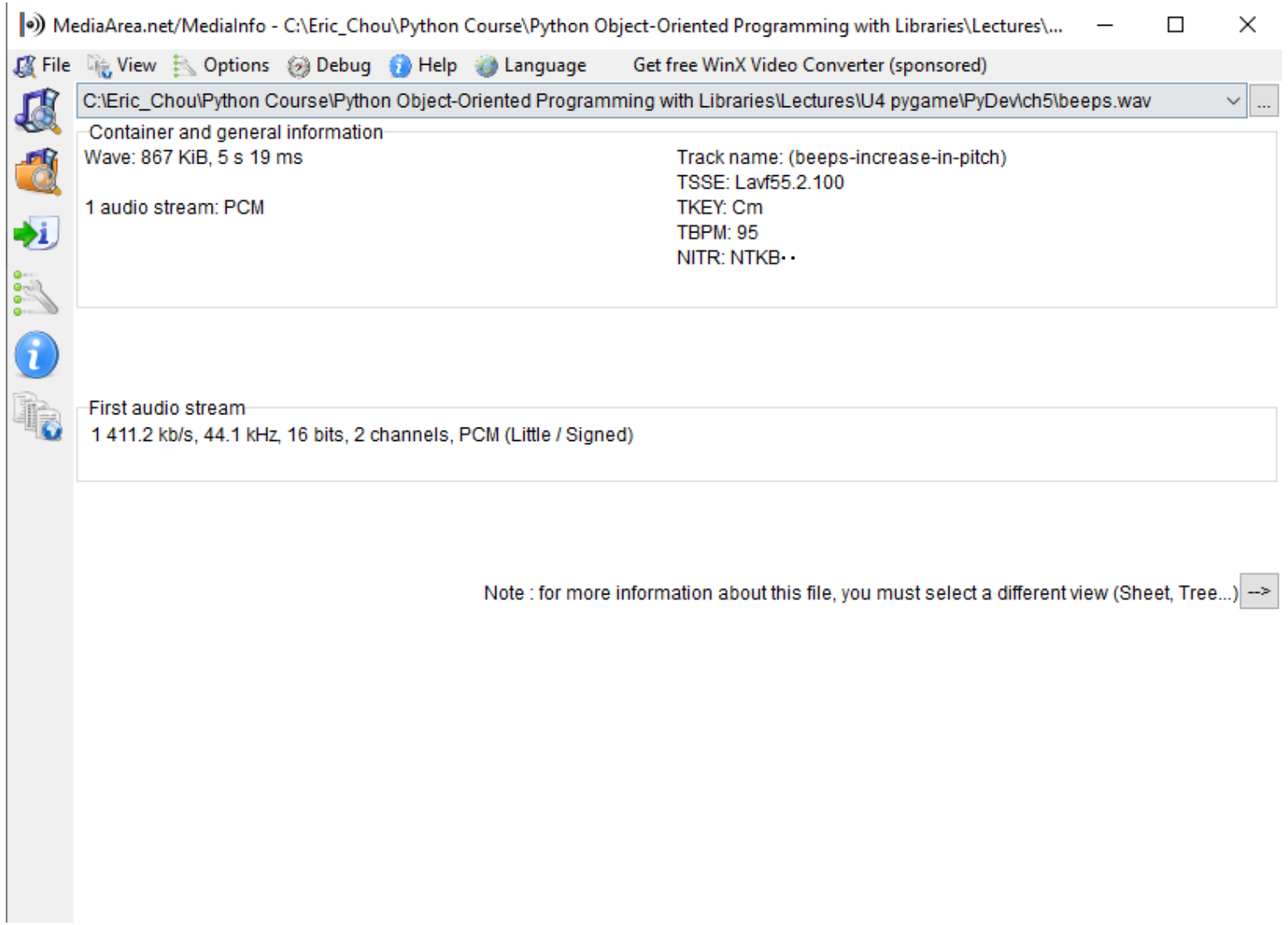
Mediainfo

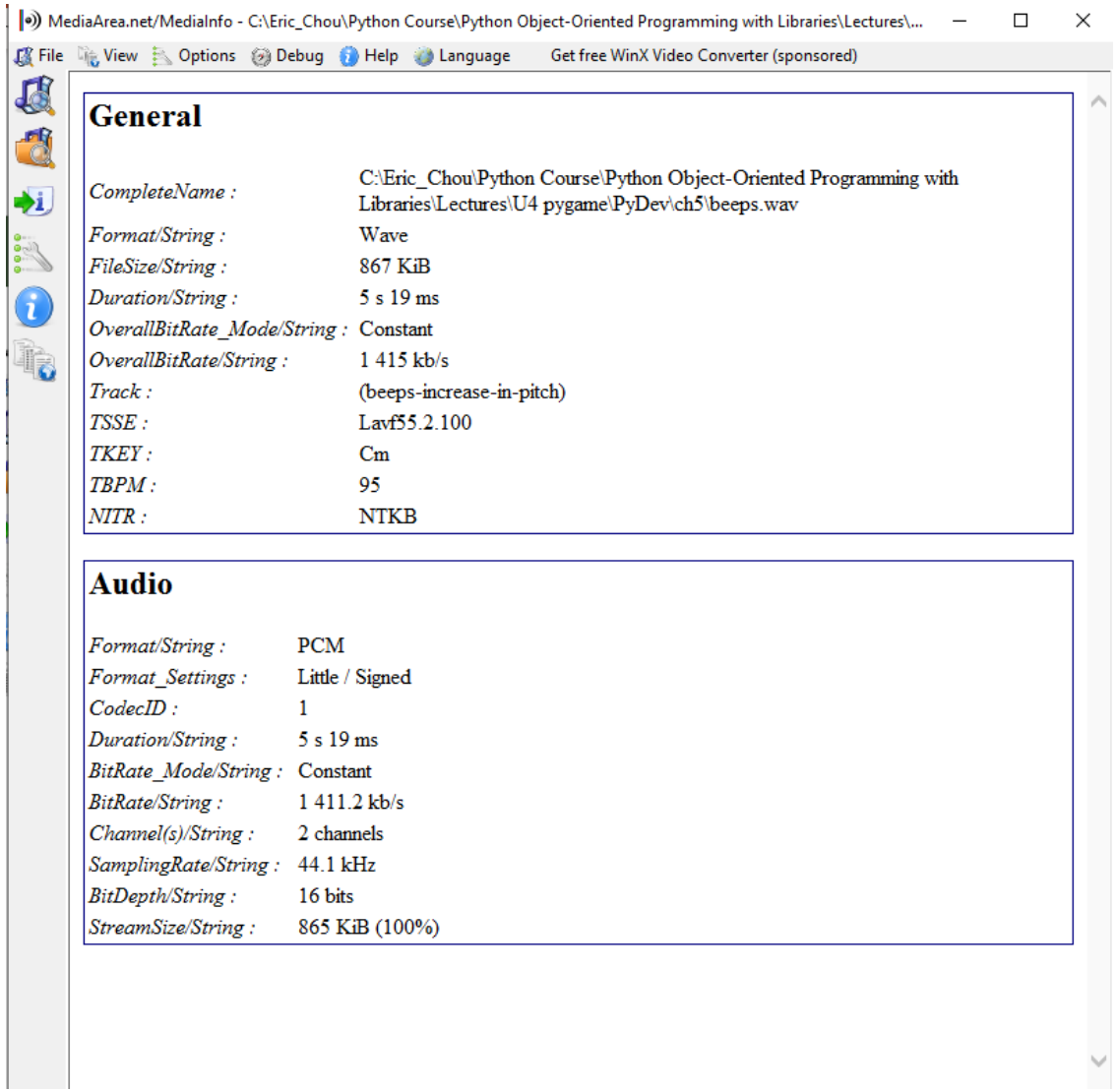
Website:

https://mediaarea.net/en/MediaInfo

The MediaInfo data display includes:

- Container: format, profile, commercial name of the format, duration, overall bit rate, writing application and library, title, author, director, album, track number, date, duration...
- Video: format, codec id, aspect, frame rate, bit rate, color space, chroma subsampling, bit depth, scan type, scan order...
- Audio: format, codec id, sample rate, channels, bit depth, language, bit rate...
- Text: format, codec id, language of subtitle...
- Chapters: count of chapters, list of chapters...


MediaArea.net/MediaInfo - C:\Eric_Chou\Python Course\Python Object-Oriented Programming with Libraries\Lectures\...
File
View
Options
Debug
Help
Language
Get free WinX Video Converter (sponsored)
C:\Eric_Chou\Python Course\Python Object-Oriented Programming with Libraries\Lectures\U4 pygame\PyDev\ch5\beeps.wav
...
Container and general information
Wave: 867 KiB, 5 s 19 ms
1 audio stream: PCM
Track name: (beeps-increase-in-pitch)
TSSE: Lavf55.2.100
TKEY: Cm
TBPM: 95
NITR: NTKB
First audio stream
1 411.2 kb/s, 44.1 kHz, 16 bits, 2 channels, PCM (Little / Signed)
Note : for more information about this file, you must select a different view (Sheet, Tree...)
-->


MediaArea.net/MediaInfo - C:\Eric_Chou\Python Course\Python Object-Oriented Programming with Libraries\Lectures\...
File
View
Options
Debug
Help
Language
Get free WinX Video Converter (sponsored)
General
CompleteName : C:\Eric_Chou\Python Course\Python Object-Oriented Programming with Libraries\Lectures\U4 pygame\PyDev\ch5\beeps.wav
Format/String : Wave
FileSize/String : 867 KiB
Duration/String : 5 s 19 ms
OverallBitRate_Mode/String : Constant
OverallBitRate/String : 1 415 kb/s
Track : (beeps-increase-in-pitch)
TSSE : Lavf55.2.100
TKEY : Cm
TBPM : 95
NITR : NTKB
Audio
Format/String : PCM
Format_Settings : Little / Signed
CodecID : 1
Duration/String : 5 s 19 ms
BitRate_Mode/String : Constant
BitRate/String : 1 411.2 kb/s
Channel(s)/String : 2 channels
SamplingRate/String : 44.1 kHz
BitDepth/String : 16 bits
StreamSize/String : 865 KiB (100%)

Play Music

LECTURE 2

Play Sound and Music

```
# Loading and playing a sound effect:
soundObj = pygame.mixer.Sound('beepingsound.wav')
soundObj.play()

# Loading and playing background music:
pygame.mixer.music.load('backgroundmusic.mp3')
pygame.mixer.music.play(-1, 0.0)
# ...some more of your code goes here...
pygame.mixer.music.stop()
```

Dub Spirit

Demo Program: music1.py

- A simple but complete example of sound effects and music player
- Basic operators for a music player

music1.py

```
import pygame
def playsound(soundfile):
    """Play sound through default mixer channel in blocking manner.
       This will load the whole sound into memory before playback """
    pygame.init()
    pygame.mixer.init()
    sound = pygame.mixer.Sound(soundfile)
    clock = pygame.time.Clock()
    sound.play()
    while pygame.mixer.get_busy():
        print("Playing...")
        clock.tick(1000)

def playmusic(soundfile):
    """Stream music with mixer.music module in blocking manner.
       This will stream the sound from disk while playing."""
    pygame.init()
    pygame.mixer.init()
    clock = pygame.time.Clock()
    pygame.mixer.music.load(soundfile)
    pygame.mixer.music.play()
    while pygame.mixer.music.get_busy():
        print("Playing...")
        clock.tick(1000)
```

```
def stopmusic():
    """stop currently playing music"""
    pygame.mixer.music.stop()

def getmixerargs():
    pygame.mixer.init()
    freq, size, chan = pygame.mixer.get_init()
    return freq, size, chan


def initMixer():
    BUFFER = 3072  # audio buffer size, number of samples since pygame 1.8.
    FREQ, SIZE, CHAN = getmixerargs()
    pygame.mixer.init(FREQ, SIZE, CHAN, BUFFER)


'''You definitely need test mp3 file (a.mp3 in example) in a directory
      1) replace a.mp3 file with it, say 'a.wav'
      2) In try except clause below replace "playmusic()" with "playsound()" ``'
```

```
def main():
    try:
        initMixer()
        filename = 'Dub_Spirit.mp3'
        playmusic(filename)
    except KeyboardInterrupt:   # to stop playing, press "ctrl-c"
        stopmusic()
        print("\nPlay Stopped by user")
    except Exception:
        print("unknown error")

    print("Done")

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