"""

File: midi.py

Author: Addy771

Description:

A script which converts MIDI files to WAV and optionally to MP3 using ffmpeg.

Works by playing each file and using the stereo mix device to record at the same time

"""

import pyaudio # audio recording

import wave # file saving

import pygame # midi playback

import fnmatch # name matching

import os # file listing

#### CONFIGURATION ####

do\_ffmpeg\_convert = True # Uses FFmpeg to convert WAV files to MP3. Requires ffmpeg.exe in the script folder or PATH

do\_wav\_cleanup = True # Deletes WAV files after conversion to MP3

sample\_rate = 44100 # Sample rate used for WAV/MP3

channels = 2 # Audio channels (1 = mono, 2 = stereo)

buffer = 1024 # Audio buffer size

mp3\_bitrate = 128 # Bitrate to save MP3 with in kbps (CBR)

input\_device = 1 # Which recording device to use. On my system Stereo Mix = 1

# Begins playback of a MIDI file

def play\_music(music\_file):

try:

pygame.mixer.music.load(music\_file)

except pygame.error:

print ("Couldn't play %s! (%s)" % (music\_file, pygame.get\_error()))

return

pygame.mixer.music.play()

# Init pygame playback

bitsize = -16 # unsigned 16 bit

pygame.mixer.init(sample\_rate, bitsize, channels, buffer)

# optional volume 0 to 1.0

pygame.mixer.music.set\_volume(1.0)

# Init pyAudio

format = pyaudio.paInt16

audio = pyaudio.PyAudio()

try:

# Make a list of .mid files in the current directory and all subdirectories

matches = []

for root, dirnames, filenames in os.walk("./"):

for filename in fnmatch.filter(filenames, '\*.mid'):

matches.append(os.path.join(root, filename))

# Play each song in the list

for song in matches:

# Create a filename with a .wav extension

file\_name = os.path.splitext(os.path.basename(song))[0]

new\_file = file\_name + '.wav'

# Open the stream and start recording

stream = audio.open(format=format, channels=channels, rate=sample\_rate, input=True, input\_device\_index=input\_device, frames\_per\_buffer=buffer)

# Playback the song

print("Playing " + file\_name + ".mid\n")

play\_music(song)

frames = []

# Record frames while the song is playing

while pygame.mixer.music.get\_busy():

frames.append(stream.read(buffer))

# Stop recording

stream.stop\_stream()

stream.close()

# Configure wave file settings

wave\_file = wave.open(new\_file, 'wb')

wave\_file.setnchannels(channels)

wave\_file.setsampwidth(audio.get\_sample\_size(format))

wave\_file.setframerate(sample\_rate)

print("Saving " + new\_file)

# Write the frames to the wave file

wave\_file.writeframes(b''.join(frames))

wave\_file.close()

# Call FFmpeg to handle the MP3 conversion if desired

if do\_ffmpeg\_convert:

os.system('ffmpeg -i ' + new\_file + ' -y -f mp3 -ab ' + str(mp3\_bitrate) + 'k -ac ' + str(channels) + ' -ar ' + str(sample\_rate) + ' -vn ' + file\_name + '.mp3')

# Delete the WAV file if desired

if do\_wav\_cleanup:

os.remove(new\_file)

# End PyAudio

audio.terminate()

except KeyboardInterrupt:

# if user hits Ctrl/C then exit

# (works only in console mode)

pygame.mixer.music.fadeout(1000)

pygame.mixer.music.stop()

raise SystemExit