pyrecaudiofile.py

March 6, 2017

1 Program - Pyrecaudiofile

Using Pyaudio, record sound from the audio device and store it into a file with the name in the argument, for a few seconds. Usage example: python pyrecaudiofile.py test.wav see also: https://people.csail.mit.edu/hubert/pyaudio/docs/

- Gerald Schuller, Dec. 2015

- **Import relevant modules and define the variables**

```
In [1]: import pyaudio
        import struct
        import math
        #import array
        import numpy
        import scipy
        import sys
        import wave

        CHUNK = 1024 #Blocksize
        WIDTH = 2 #2 bytes per sample
        CHANNELS = 1 #2
        RATE = 16000  #Sampling Rate in Hz
        RECORD_SECONDS = 12
```

- **Initialise audio port with its parameters and print out their channel and sampling rate information:**

```
In [ ]: for i in range(0, a):
            print("i = ",i)
            b = p.get_device_info_by_index(i)['maxInputChannels']
            print(b)
            b = p.get_device_info_by_index(i)['defaultSampleRate']
            print(b)

        stream = p.open(format=p.get_format_from_width(WIDTH),
                        channels=CHANNELS,
```

```
                    rate=RATE,
                    input=True,
                    output=True,
                    #input_device_index=3,
                    frames_per_buffer=CHUNK)
```

- **Provide filename for file to be recorded, through command line argument:**

```
In [ ]: print("filename=", sys.argv[1])
        sndfile=sys.argv[1]
        wf = wave.open(sndfile, 'wb')
        wf.setnchannels(CHANNELS)
        wf.setsampwidth(WIDTH)
        wf.setframerate(RATE)
```

- **Start recording by looping through blocks(CHUNKS) and write the data to the given file:**

```
In [ ]: for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
            #Reading from audio input stream into data with block length "CHUNK":
            data = stream.read(CHUNK)

            #Writing data to audio output file:
            wf.writeframes(data)

        print("* done")

        wf.close()
        stream.stop_stream()
        stream.close()

        p.terminate()
```