Coverage for **audio_calculations.py**: 100%

32 statements | 32 run | 0 missing | 0 excluded

```python
import soundfile as sf
import subprocess
import moviepy.editor as mp

class audio_calculations():
    """Opening file and calculations for LUF and True Peak.

    Will open a wav or mp4 file, test its peak value against
    a standard's peak value, and test its LUF value against a standard's LUF value.

    Attributes:
        file_path: The current audio file path to be tested. Can change frequently. String object.
    """

    def __init__(self, file_path):
        """Initializes the file to be tested.

        Will store the file to be tested.

        Args:
          self: The main object
          file_path: String containing the name of the file

        Raises:
          Any errors raised should be put here

        """
        # initialize path of the file being passed in
        self.file_path = file_path

    def get_file_path(self):
        """ Gives the current file name being stored by the object

        Returns the filename attribute being stored.

        Args:
            self: Instance of main object

        Returns:
            file_path: the file path of the audio file selected by the user

        Raises:
            Any errors raised should be put here

        """
        return self.file_path

    def select_file(self):
        """Opens the file, fetches its needed information, and calculates its LUFS and True peak values.

        Opens the selcted file, fetches its sample rate, data itself, and number of channels, and calculates its
        LUFS and True peak values.

        Args:
            self: A main Object.

        Returns:
            A tuple containing the selected file's data, sample rate, number of channels, LUFS value, and True peak value

        Raises:
            Add possible errors here.

```

```
        """
        fileType = self.get_file_path().split('.') #split file path on '.'
        fileType = fileType[-1] #take the last entry in the list from split as the file extension

        #if the file is an MP4 file then open using moviepy and extract the audio
        if fileType.upper() == 'MP4':
            clip = mp.VideoFileClip(self.get_file_path())
            audioFile = clip.audio
            data = audioFile.to_soundarray(None,44100)
            rate = 44100
        #else open as an audio (wav/flac) file
        else:
            data, rate = sf.read(self.get_file_path())

        if len(data.shape) > 1:
            n_channels = data.shape[1]
        else:
            n_channels = 1

        output_query = f"ffmpeg -i {self.get_file_path()} -af loudnorm=I=-16:print_format=summary -f null -"
        output = subprocess.getoutput(output_query)

        list_split = output.split('\n')

        for i in range(len(list_split) - 1, 0, -1):
            if list_split[i][0:16] == 'Input True Peak:':
                lufs_string = list_split[i - 1]
                peak_string = list_split[i]
                break

        lufs = float(lufs_string.split()[2])
        peak = float(peak_string.split()[3])
        wav_info = (data, rate, n_channels, lufs, peak)

        return wav_info
```

« index coverage.py v6.3.2, created at 2022-04-19 20:36 -0400