Bonus: Teach Kyle to dance

I have developed a script that tells Kyle to play a song, detect its beat, and do random "dance" moves at each beat. You are going to improve the code by adding more steps and variations.

The file dancetothebeat.py is located in the ~/Sandbox folder, so you should copy it to your personal directory in Students. For example, if the target directory's name is marko:

$ cp ~/Sandbox/dancetothebeat.py ~/Students/marko

The script dancetothebeat.py accepts command line arguments, so look inside of the Music directory to find a music file that you would like the robot to dance to. Copy that filename. For example, you could use rickrolled.wav or any other file that you may have already uploaded to the robot.

Put Kyle on the ground with a partner to keep him out of trouble.

From inside your student directory, run dancetothebeat.py and tell it to play Rick Astley with:

$ python dancetothebeat.py ~/Music/rickrolled.wav

Unfortunately this script will only work with wav files, so if you already have an mp3 on the machine that you would like the robot to dance to, you will need to convert it to a wav.

Thankfully this robot has a command line tool called ffmpeg that can be used to do just that. For example, if I would like to convert a file called ironman.mp3 to a wav format, you can use the command:

$ ffmpeg -i ~/Music/ironman.mp3 ~/Music/ironman.wav

Now you can get Kyle to dance to ironman.wav (he does a sort of ok job):

$ python dancetothebeat.py ~/Music/ironman.wav

Give Kyle more dance moves

The script that makes Kyle dance (dancetothebeat.py) is kind of complicated. We don't need to understand it all. Open up dancetothebeat.py with the nano text editor to take a peek under the hood:

$ nano dancetothebeat.py

Look for the following code:

```
def Step1():
    flash_eyes()
    flash_blinkers()
    random_wheels()

steps = [Step1]
```

Then look at the function definition for pyaudio_callback():

```
def pyaudio_callback(_in_data, _frame_count, _time_info, _status):
    samples, read = a_source()
    is_beat = a_tempo(samples)
    if is_beat:
        #samples += click
        steps[random.randint(0,len(steps)-1)]()
        print ('tick', time.time()) # avoid print in audio callback
    audiobuf = samples.tobytes()
    if read < hop_s:
        return (audiobuf, pyaudio.paComplete)
    return (audiobuf, pyaudio.paContinue)
```

The line **`steps[random.randint(0,len(steps)-1)]()`** will choose a random step in the array (technically called a "list" in Python) called **`steps[]`** and execute that step on a given beat in the music.

If you would like to add a new dance step, let's call it the function `Step2()`, you can define that function and then add it to the `steps[]` list. For example, to get the robot to just flash its "eyes", we can define `Step2()` as:

```
def Step2():
     flash_eyes()
```

and then add `Step2()` to the `steps[]` list. Make sure to leave off the parentheses when putting the function name into the list.

```
steps = [Step1, Step2]
```

Run the program and see if it works!

Be creative

Do you think you can add other functions to the dance moves? Like turning the robot's servo motor (its head)? Maybe you can figure out how to change the colors on its eyes, too?