Solenzio Konlé

Recueil d'idées pour l'implantation d'effets sonores dans le projet Endless Landscape

Ce document n'a pour but que de donner des conseils sur la façon de manipuler l'audio, il ne s'agit pas d'une méthode à suivre forcément à la lettre

Module: la bibliothèque pygame permet déjà de manipuler des extraits audio à l'aide notamment de l'extension pygame.mixer.
documentation disponible ici: *https://www.pygame.org/docs/ref/mixer.html*
La bibliothèque permet la plupart des transformations utiles comme jouer sur le volume, mettre des sons en boucle, créations de plusieurs pistes indépendantes etc...
En revanche, pour ce qui est de l'accélération (et donc du changement de fréquences) des sons, il n'y a pas de solutions directes ***(débuts d'idées de solutions*** *https://www.reddit.com/r/pygame/comments/2fc6uv/is_it_possible_change_the_pitch_of_an_audio_file/)*

Structure du code: On avait convenu que l'ambiance sonore devait être séparée en deux parties: une dépendante des événements du code (des changements de directions de la tête de lecture par exemple) et une autre indépendante, en passant des sons d'ambiances.
Côté utilisateur il faudrait sans doute deux listes de listes (indé et pas indé) se décomposant comme suit:
[[***n° du son,*** [***tuple de vitesse***], [***tuple de volume***], ***proba d'être tiré si c'est la liste pas indé***], ...] "n° du son" correspond au numéro du fichier donné dans le fichier son (par exemple 5 pour le son d_5 ***cf structure des fichiers***).
La taille de cette liste indiquera au programme combien de sons sont à prendre en compte.
Lorsque l'on déclenchera un son dans la partie dépendante (en cas d'événement déclencheur comme un changement de direction de lecture par exemple), il faudra donc tirer en paramètre le numéro du son (où on pondèrera le tirage par la probabilité de tirer tel ou tel numéro), le tuple de vitesse correspondant (qui est la borne min-max de l'intervalle dans lequel on piochera la vitesse à laquelle le son sera joué), tuple de volume (même idée que la vitesse).
Pareil pour la partie indépendante sauf qu'il n'y aura pas de question de proba il faudra juste toujours garder le channel occupé pour éviter les blancs.
Peut être que créer une classe de son pourrait être une idée avec comme attribut une piste, le son joué, le volume, la probabilité d'être joué, la vitesse... et en méthode surtout les getters et les setters (voire des fonctions qui tirent aléatoirement un nouveau volume etc...)

Structure des fichiers: On avait pensé à créer un dossier son à côté du script EP.py dans lequel on mettrait nos fichiers indépendants de la boucle en les nommant: i_n.wav (n allant de 0 à “nombre total de sons indépendants”-1) et dépendants de la boucle: d_n.wav. Il sera donc assez simple de les appeler dans le script.

Enfin l'extension .mp3 ne semble pas être très bien suportée par pygame, je n’ai pour l’instant qu’utiliser l’extension .wav .