Création d'un son

Pas d'importation nécessaire

playTone(freq)	Joue un son de fréquence freq [Hz] et d'une durée de 1000 ms (fonction bloquante)
playTone(freq, block=False)	Idem, mais en version non bloquante. Utile pour jouer plusieurs sons simultanément
playTone(freq, duration)	Joue un son de fréquence freq sur une durée duration [ms]
playTone([f1, f2, ...])	Joue plusieurs sons successifs d'une durée de 1000 ms, selon les fréquences présentes dans la liste passée en paramètre
playTone([(f1, d1), (f2, d2), ...])	Idem, en spécifiant au sein d'un tuple, pour chaque fréquence de la liste, la durée pendant laquelle il faut tenir le son en [ms]
playTone([("c", 700), ("e", 1500), ...])	Idem, mais en spécifiant les fréquences à l'aide du nom des notes (notation anglo-saxonne de Helmholz) et les durées en [ms]. La tessiture supportée va de « C » qui correspond au Do1 jusqu'au « h''' » qui correspond au Si5 (voir l'article « Fréquence des touches du piano sur Wikipedia »)
playTone([("c", 700), ("e", 1500), ...], instrument = "piano")	Idem, en sélectionnant le type d'instrument. Les instruments supportés sont les suivants : piano, guitar, harp, trumpet, xylophone, organ, violin, panflute, bird, seashore, ... (voir la spécification MIDI)
playTone([("c", 700), ("e", 1500), ...], instrument = "piano", volume=10)	Idem, en spécifiant le volume compris entre 0 et 100

Lecture d'un fichier son

from soundsystem import *

getWavMono(filename)	Retourne une liste d'échantillons correspondant au fichier mono filename. Avec "wav/xxx.wav", il est également possible de charger les fichiers depuis le sous-dossier _wav du dossier contenant l'archive tigerjython2.jar
getWavStereo(filename)	Idem, pour un fichier stéréo
getWavInfo(file)	Retourne une chaine de caractères contenant les informations à propos du fichier file telles que le taux d'échantillonnage, etc.
openSoundPlayer(filename)	Ouvre un lecteur sonore avec le fichier filename pour en permettre la lecture avec les fonctions de lecture présentées par la suite
openMonoPlayer(filename)	Idem, en ouvrant un lecteur mono. Il est possible d'ouvrir des fichiers stéréo en lecture mono en faisant la moyenne entre les deux canaux
openStereoPlayer(filename)	Idem, en ouvrant un lecteur stéréo. Il est possible d'ouvrir un fichier mono (les deux canaux seront alors identiques)
openSoundPlayerMP3(filename)	Comme openSoundPlayer(), mais pour des fichiers MP3
openMonoPlayerMP3(filename)	Comme openMonoPlayer(), mais pour des fichiers MP3
openStereoPlayerMP3(filename)	Comme openStereoPlayer(), mais pour des fichiers MP3
play()	Joue le son depuis la position actuelle et retourne immédiatement (fonction non bloquante)
blockingPlay()	Idem, mais en ne retournant que lorsque la lecture est terminée (fonction bloquante)
advanceFrames(n)	Avance la lecture de n échantillons depuis la position courante
advanceTime(t)	Idem, en avançant de t millisecondes depuis la position courante

getCurrentPos()	Retourne la position courante du curseur de lecture
getCurrentTime()	Retourne le temps de lecture courant
rewindFrames(n)	Recule le curseur de lecture de n échantillons à partir de la position de lecture courante
rewindTime(t)	Idem, en reculant de t millisecondes à partir de la position de lecture courante
stop()	Arrête la lecture et réinitialise le curseur de lecture à la position initiale
setVolume(v)	Ajuste le volume selon la valeur v comprise entre 0 et 100
isPlaying()	Retourne True si la lecture du clip n'est pas encore terminée
mute(bool)	Passe le système en mode muet lorsque le paramètre bool est True et audible lorsque bool est False.
playLoop()	Lecture en boucle : la lecture du clip est relancée indéfiniment depuis le début
replay()	Rejoue le clip une fois supplémentaire
delay(time)	Ajoute au programme un temps d'attente de time millisecondes, utile pour jouer des silences

Capture sonore et Enregistrement

from soundsystem import *

openMonoRecorder()	Ouvre un enregistreur de sons mono
openStereoRecorder()	Ouvre un enregistreur de sons stéréo
capture()	Débute la capture sonore
stopCapture()	Suspend la capture sonore
getCapturedBytes()	Retourne les enregistrements capturés octet à octet sous forme de liste
getCapturedSound()	Retourne les échantillons enregistrés comme une liste de nombres entiers. En mode stéréo, la suite d'échantillons alterne entre chacun des deux canaux (les positions paires correspondent au premier canal et les positions impaires au deuxième canal)
writeWavFile(samples, filename)	Sauve les échantillons présents dans la liste d'échantillons samples dans le fichier WAV filename

Transformation de Fourier rapide (FFT = Fast Fourier Transform)

fft(samples, n)	Transforme les n premières valeurs de la liste samples. Retourne une liste de n // 2 valeurs spectrales équidistantes (floats) comprises entre 0 et fs/2 Hz avec un écart (résolution) de fs/n
sine(A, f, t)	Génère une onde sinusoïdale d'amplitude A et de fréquence f (phase 0) pour chaque valeur t
square(A, f, t)	Idem, mais en générant une onde carrée
sawtooth(A, f, t)	Idem, mais en générant une onde en dents de scie
triangle(A, f, t)	Idem, mais en générant une onde triangulaire
chirp(A, f, t)	Génère, pour chaque valeur de t, une onde sinusoïdale d'amplitude A et dont la fréquence augmente avec le temps (fréquence initiale f)