SON SFML

(Le document est écrit en Comic Sans MS, juste pour le plaisir de Nodius)

SFML fournit deux classes pour jouer de l’audio : sf::Sound et sf::Music

sf::Sound est une classe légère qui joue des données audio chargées avec sf::SoundBuffer. Il faut l’utiliser pour des sons relativement courts qui peuvent tenir sans problème en mémoire, et qui ne doit souffrir d’aucun lag. (Ex : des Bruits de pas, saut, coups de feu, etc..)

sf::Music lit en flux (stream) et ne charge pas toutes les données en mémoire. Cette classe est utilisée pour jouer des musiques compressées (.ogg) qui durent plusieurs minutes. Cela empêche la surconsommation de mémoire inutilement.

Les fonctions de base pour la lecture de la musique(music.[\*]) ou des sons(sound.[\*])

Il faut inclure le SFML AUDIO! #include <SFML/Audio.hpp>

1. // on démarre la lecture
2. sound.play();
4. // on avance de deux secondes
5. sound.setPlayingOffset(sf::seconds(2));
7. // on met en pause
8. sound.pause();
10. // on reprend la lecture
11. sound.play();
13. // on arrête la lecture et on se replace au début
14. sound.stop();
16. // on change le pitch du son
17. sound.setPitch(1.2);
19. // le pitch est est un facteur qui modifie la fréquence perçue du son : plus que 1 rend le son plus aigü, moins que 1 rend le son plus grave, et 1 restitue le son d'origine. Attention, la modification du pitch a un effet de bord : cela change aussi la vitesse de lecture.
21. //on change le volume
22. sound.volume(50);
24. //Sa valeur est comprise entre 0 (muet) et 100 (volume maximum). La valeur par défaut est 100.
26. //on change le loop
27. sound.setLoop(**true**);

*Charger et jouer un son*

Avant toute chose, il faut comprendre que le son n’est pas stocké dans sf::Sound mais dans une classe à part qui est sf::SoundBuffer. Cette classe encapsule les données audios, qui ne sont ni plus ni moins qu’un tableau d’entiers 16 bits. (Échantillons audio)

Alors en pratique cela ressemble à cela

1. sf::SoundBuffer buffer;
2. // on charge quelque chose dans le buffer...
4. sf::Sound sound;
5. sound.setBuffer(buffer);
6. sound.play();

Voici un exemple de ce que peut contenir sf::SoundBuffer

1. #include <SFML/Audio.hpp>
3. **int** main()
4. {
5. sf::SoundBuffer buffer;
6. **if** (!buffer.loadFromFile("sound.wav"))
7. **return** -1;
9. ...
11. **return** 0;
12. }

\*Vous pouvez affecter le même buffer à plusieurs sons. Vous pouvez même les jouer tous en même temps.

Jouer une musique

Contrairement à sf::Sound, sf::Music ne précharge pas les données audios, elle les lit à la demande directement depuis la source. Il est important de noter que contrairement à sf::SoundBuffer qui utilise loadFromFile, sf::Music utilise openFromFile car il OUVRE le son ce qui permet de pouvoir le faire jouer ou on le souhaite par la suite avec music.play();

Voici un exemple

1. sf::Music music;
2. **if** (!music.openFromFile("music.ogg"))
3. **return** -1; // erreur
4. music.play();

Erreurs courantes

Buffer audio détruit

L’erreur la plus courante est de déclarer un buffer dans une portée réduite (telle qu’une fonction) et le laisser mourir à la fin alors qu’un son l’utilise toujours.

Exemple

1. sf::Sound loadSound(std::string filename)
2. {
3. sf::SoundBuffer buffer; // ce buffer est local à la fonction, il sera détruit...
4. buffer.loadFromFile(filename);
5. **return** sf::Sound(buffer);
6. } // ... ici
8. sf::Sound sound = loadSound("s.wav");
9. sound.play(); // ERREUR : le buffer du son n'existe plus, le comportement est indéterminé

Un son ne garde qu’un pointeur vers le buffer que vous lui affectez, il n’en fait pas de copie. Il faut gérer correctement la durée de vie des buffers.

Même chose pour la musique

La musique a besoin de sa source aussi longtemps qu’elle est jouée, puisqu’elle charge les données au fur et à mesure de la lecture.

Exemple

1. // on démarre avec un fichier audio en mémoire (imaginez que l'on vient de l'extraire d'une archive zip)
2. std::vector<**char**> fileData = ...;
4. // on le joue
5. sf::Music music;
6. music.openFromMemory(&fileData[0], fileData.size());
7. music.play();
9. // "ok, il semblerait que l'on puisse se passer du fichier maintenant"
10. fileData.clear();
12. // ERREUR: la musique était toujours en train de lire le contenu de fileData! le comportement est maintenant indéterminé // ERREUR : le buffer du son n'existe plus, le comportement est indéterminé

Pour terminer, sf::Music n’est pas copiable, donc le compilateur interdit cette manipulation

Exemple

1. sf::Music music;
2. sf::Music anotherMusic = music; // ERREUR
4. **void** doSomething(sf::Music music)
5. {
6. ...
7. }
8. sf::Music music;
9. doSomething(music); // ERREUR (la fonction devrait prendre son paramètre par référence, pas par copie)