FluidSynth

Architecture interne minimum de base

Ceresa jean-jacques: first writing 29/04/2010

1. Architecture interne de FluidSynth

Ce chapitre décrit l'architecture interne de la librairie FluidSynth dans le but de permettre au développeur d'une application de comprendre les divers cas d'applications ainsi que l'organisation multi-tâches.
La librairie contient plusieurs composants dont un minimum est nécessaire. Les autres composant ne sont nécessaires que si l'application en à besoin En fonction des besoins on peut construire une l'application simple ou complexe si les besoins sont complexes. De même l'architecture multi-tâches de l'application peut être mono-tâches ou multi-tâches en fonction des besoins.

1.1. Architecture minimum de base

Les composants de base minimum sont ceux avec lesquels l'application peut faire quelque chose.
Ces composants exposent des API que l'application utilise. Ces composants sont des objets:

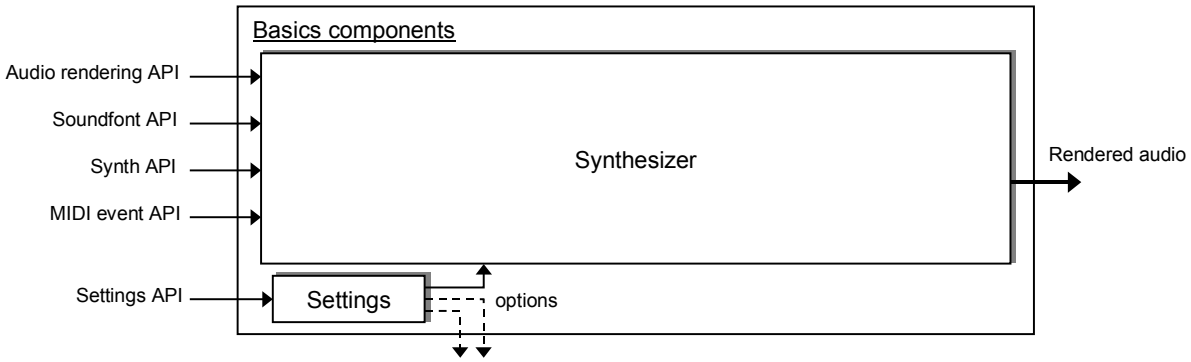


Fig. 1
Les objets de bases nécessaire sont:
L'objet Settings (voir 1.1.1) et l'objet Synthétiseur (voir 1.1.2).

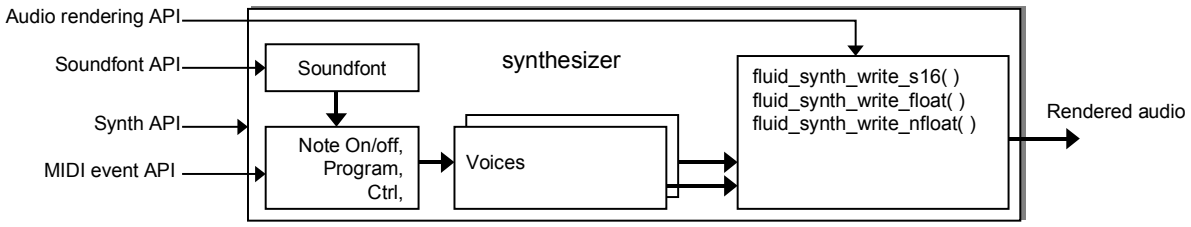
1.1.1. Objet de base Settings: **fluid_settings**

L'objet ***fluid_settings*** détient les options de configuration de tous les autres composants.
Ces options sont utilisées par les autres objets (paramètres de configuration), il doit être le premier objet créé. Il expose un de fonctionsAPI *Settings* qui permettent à l'application:

- La création d'un objet fluid_settings.
- Le positionnement/lecture des options dans l'objet fluid_settings.

1.1.2. Objet de base Synthétiseur: **fluid_synth**

Fig. 2



L'objet ***fluid_synth*** est créé après l'objet *settings* et permet à l'application de produire de l'audio.
Effectivement , l'objet fluid_synth expose un jeu de fonctions API qui permet à l'application:

- La création de l'objet.
- Le chargement d'un fichier SoundFont à l'aide des fonctions *API Soundfont*.
- Le jeu d'événements Midi (Program change,Note On/off, controleurs ..),à l'aide des fonctions *API Midi event*.

- La production de l'audio correspondant aux événements Midi en cours, à l'aide des fonctions *API production audio*.
  L'audio est produite dans des buffers fournis par l'application.
- L'application peut positionner/lire certains paramètre globaux à l'aide des fonctions *API Synth*

1.1.3. Exemple d'application utilisant l'architecture de base

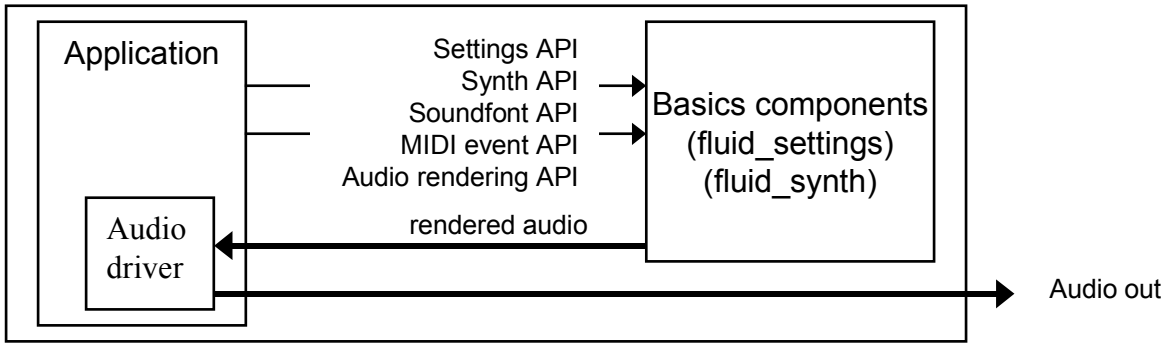


Fig. 3

Les exemples d'applications utilisant seulement l'architecture minimum sont nombreux:

- Production audio intégrée dans une application nécessitant un accompagnement de sons (jeux vidéo,système d'informations..).
- Instruments d'analyse acoustique, générateurs de signaux.

1.1.4. Comportement Multi-tâches

Les objets de l'architecture de base (fluid_settings, fluid_synth) ne contiennent que des fonctions. Il n'y a pas de tâches. L'application peut être composée elle-même d'une seule tâche qui appelle les fonctions API et produit l'audio à l'aide de pilote audio personnalisé.
Dans ce cas les fonctions API *Midi event* et API *audio rendering* étant appelées par la même tâche de façon synchrone (non simultané), il n'y a pas d'accès critique à l'intérieur des composants de base. Ainsi , il ne sera pas nécessaire de positionner les options de protection d'accès (*synth.threadsafe-api (voir* ***Erreur! Source du renvoi introuvable.****)*, *synth.parallel-render (voir* ***Erreur! Source du renvoi introuvable.*** *)*