Flex Music Maker

Multimedia Authoring 2009

Group 16, Richard van Willegen, 1473484

Concept

The Flex Music Maker is an application created in Flex, that allows the user to be creative with music. It can be seen as a game, because the main goal of the application is to have fun. However, games usually provide a challenge and the user will receive some kind of feedback on his performance to this challenge. The challenge of this application is to create music that sounds good. The only way to know how well you are doing is by using your ears.

The application contains audio clips that can be loaded and played. The clips are mostly single instrument clips. It's up to the user to select clips that sound good together.

Most audio software contains a lot of buttons and sliders. One of the goals of this project was to minimize the number of buttons and sliders. It uses a two dimensional field in which the active clips are displayed as icons. The position of the icons in the field determine the values. Clips that are placed higher in the field will have higher volumes. By dragging the clips around, the values will change.

Functions

- Loading of an external sound bank, containing the sound clips
- Download the clips the user selects to use
- Drag and drop interface to launch clips
- Sequencer triggers the new clips at the next bar (4 beats), to keep them in sync
- Change the volume of clips by dragging clip icons along the y axis
- A spectrum at the background provides visual feedback to the mixed sound
- Progress bars provide visual feedback to the sequencer bar progression and clip playback

Implementation

The program was built using Flex Builder 3 and Actionscript 3. Actionscript provides enough functionality to do any sound processing, but the flash player has unreliable performance, which makes the looping and syncing of sound clips difficult.

The first problem was to loop a sound smoothly. Actionscript provides an easy to use interface for playing mp3s, but mp3s are not clean and could not be looped without hearing a short glitch in the sound.

Actionscript 3 allows the use of audio sample data as a byte array. This allows the use of wav files, which can be loaded as binary data and converted to the format of actionscript sound data. The Popforge library for actionscript (*code.google.com/p/popforge/*) contained a class which converts wav data to sound data. I used this library to do this.

Even when using wav data there was still a very small glitch when looping the sound, caused by a delay in re-triggering the sound. The popforge library also contains an audio buffer class, which allows continuous playback of audio. It uses audio buffers of a multiple of 2048 samples, which do not seem to have a delay in re-triggering. Still unaware of why this is, I implemented buffers of 4 x 2048 samples, that will continuously be filled with audio sample data from the clips. This has resulted in smooth looping of the clips.

The second problem was the syncing of the clips. When one clip plays the next should be launched in sync with the first one or else the sounds would become a mess.

To keep the sounds in sync, the soundbank has to contain clips that have the same bpm (beats per minute). I chose 125, because it has a round number of milliseconds per beat. My first idea was to use the Timer class in actionscript to trigger the new clips at the next bar. This would be done by calling a function after the number of milliseconds of one bar have passed, using a Timer interval. It appeared the Timer does not trigger the function exactly after the specified milliseconds of time, but merely at some point around that time, maybe up to 100 milliseconds late.

Using a timer is therefor not possible. I solved the problem by counting audio samples. A bar has a fixed amount of audio samples, which makes it possible to count them and know which sample will be the first sample of a bar. To do this I created a sequencer class, which creates an empty sound for each available channel in the soundmixer. The empty sounds are all launched simultaneously at the start of the program. When a new clip is added, it requests a slot in the sequencer. The sequencer will then replace the empty sound samples of the sound buffer matching the selected slot with the samples from the clip, starting at the correct sample in the correct buffer.

One channel of the soundmixer is used for a sync buffer, which is an empty buffer, that is looped and re-triggers all the buffers each time it is done playing, instead of each buffer re-triggering itself. When there is a glitch in the playback, caused by cpu usage or whatever reason the flash player may have to glitch, the glitch will often not be exactly the same for each channel, causing them to run out of sync. The sync channel will then cause all the channels to be re-synced at the next trigger, which would not happen when all the channels re-trigger themselves.

Improvements

The flash player has some performance issues, which still causes the sound to glitch now and then. This increases with the amount of clips playing simultaneously.

The program only supports volume changes when dragging clips around. This could be extended with audio effects, that can affect the clips when the clip icons get closer to the effect icons.

The volume change is applied to the Y position of the clips in the field. Another effect could also be applied to the X position.

The program now only supports 125 bpm tempo. It could be extended to support any bpm.