Musificator

Technische Dokumentation

Einleitung - Prämisse

Der Musificator ist ein Programm, das in HTML5 und JavaScript realisiert wurde. Es verwendet die WebAudioAPI Java Script um Sound Dateien zu verändern.
Ziel ist es, den User eine vorhandene Melodie mit Hilfe von vorgefertigten Effekten modifizieren zu lassen. Gearbeitet wurde mit Visual Studio Code, GitKraken und Fruity Loops.

Die Sound Schleife

Die Sound Samples sind .wav Audio Dateien, die in einer Schleife wiederholt abgespielt werden. Die Ausgabe jedes einzelnen Samples ist grundsätzlich aktiv, doch wird eine zwischengeschaltete gain Node auf 0 gesetzt um keinen Sound entstehen zu lassen. Dadurch lösen wir Probleme mit der zeitlichen Gleichschaltung der Samples und ermöglichen so, die einzelnen Samples in der Mitte einer Schleife erklingen zu lassen, ohne auf den nächsten Durchlauf warten zu müssen.

Die Schleife ist keine zeitlich getaktete Schleife, sondern die Taktung hängt davon ab, ob die Audio Dateien noch abgespielt werden. Wenn alle drei auf “paused” gestellt sind, werden sie alle gleichzeitig wieder abgespielt. Die Samples sind von der Länge gleich lang, um keine Unterbrechungen in der Ausgabe zu haben.

Die Modifikatoren

Die Audio Nodes mit den Sound Dateien werden nicht direkt durch den Gain zur Destination geführt, sondern laufen noch durch eine Reihe von anderen Nodes, um verschiedene Effekte zu schaffen. Diese können sich von Sample zu Sample leicht unterscheiden, da nicht zu jeder Art von Sample der gleiche Effekt passt, aber zum größten Teil sind diese überall gleich. Den gesamten Weg eines Sound Samples bis zur Destination Node nennen wir anschließend “Stream”.

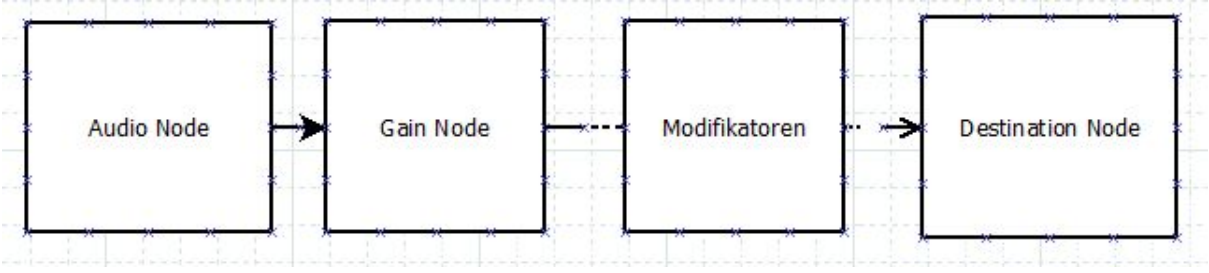

Abb. 1: Aufbau eines Audio Streams

Die erste Effect Node, durch die unser Audiosignal fließt, ist eine Waveshaper Node. Diese nutzt eine Sinus Kurve um den Klang zu verzerren. Wir haben die Verzerrung eher langsam und grob berechnet, damit die Verzerrung nicht die späteren Modifikatoren zu stark überlagert.

Die zweite Effect Node ist ein “lowpass” oder “highpass” filter. Je nachdem um welchen Stream es sich handelt. Dadurch sollen explizit die sehr tiefen oder hohen Töne aus dem Klang herausgefiltert werden. Bei der Deaktivierung dieser Filter schaltet das Programm auf einen “allpass” Filter, da es nicht möglich war, diesen Filter einfach nur zu deaktivieren.

Die dritte Effect Node rechnet die Qualität der Audio-Ausgabe herrunter. Dadurch “rauscht” der Klang stärker.

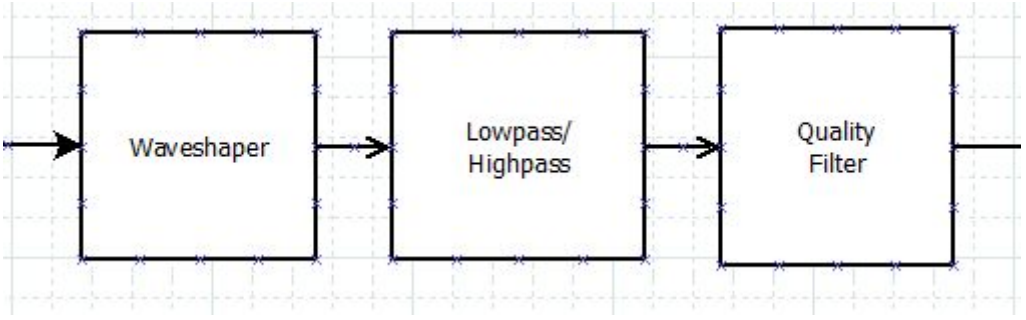

Abb. 2: Verbindung der Modifikatoren

Music Samples

Die einzelnen Musiksamples sind aus einem eigens erstellten Musikstück (mit Fruity Loops) extrahiert und in drei einzelne Samples aufgeteilt, die alle gleich lang sind.

Bedienungsoberfläche

Die komplette visuelle Ausgabe ist mit Javascript, HTML und CSS erstellt. Dabei stellt HTML die inhaltliche Struktur da, das heißt alle Elemente wie Buttons, Texte und der Canvas Bildschirm sind in HTML angelegt worden.

Um das Ganze optisch richtig anzupassen und die Struktur anzulegen, wurde CSS verwendet, es soll den Inhalt und das Layout voneinander trennen. Die anfänglichen Hoverelemente der Buttons sind mit CSS programmiert und die Aufteilung der Bedienelemente ebenfalls. Farblich steht die ButtonBox-Farbe für eine grafische Animation und dient der Übersicht. Nachdem die Buttons einmal aktiviert wurden, erfolgt die weitere Änderung der Farbe über Javascript.

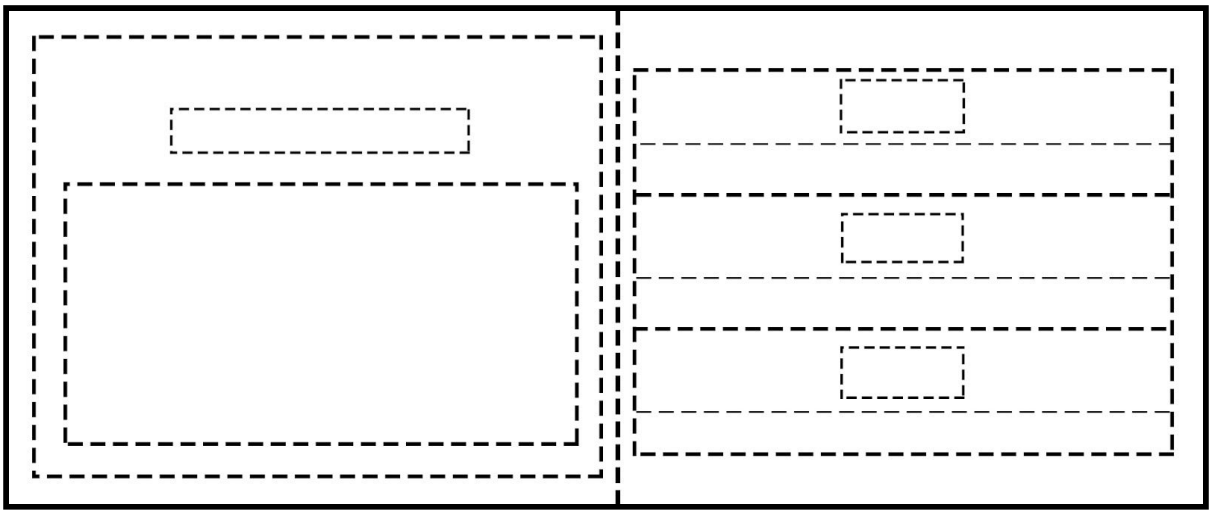

Abb. 3: HTML/ CSS Aufbau

Visuelle Ausgabe

Die animierte Ausgabe wurde mit dem JavaScript Element “Canvas” erstellt und ist mit der JS Funktion der Filter verbunden. Die Wellenbewegung setzt ein, sobald ein Musikstück anfängt zu hören ist. Die Wellenbewegung wird dabei durch eine Sinusformel berechnet und visualisiert. Je nach Filter und Funktion wird dieses Wellenbewegung verstärkt oder abgeschwächt. Sämtliche Effekte können aufeinander aufbauen und sich verstärken.

Effect 1 erhöht die Wellenbewegung und ist direkt an die Filterfunktion geknüpft.
Effect 2 vergrößert entweder den Wert des Radius vom Kreis oder erstellt um das zweite blaue Quadrat einen Rahmen.
Effect 3 lässt die Werte für den Radius vergrößern und verkleinern, beim blauen Quadrat verändert es die Start- und Endwerte des Rechtecks.

Der sogenannte “Schweif” entsteht durch das langsame Füllen des Canvas Bereichs mit einer verringerten Sichtbarkeit.


musificator
On
Effect 1 off
Effect 2 off
Effect 3 off
Off
Effect 1 off
Effect 2 off
Effect 3 off
Off
Effect 1 off
Effect 2 off
Effect 3 off