# Web Audio API

Web Audio API позволяет нам создавать звуки прямо в браузере. Это делает ваши сайты, приложения и игры более увлекательными и интересными. Вы даже можете разрабатывать специфичные для музыки приложения, такие как драм-машины или синтезаторы. В этой статье, мы узнаем о том как работать с Web Audio API, разрабатывая некоторые увлекательные и простые проекты.

Все аудио операции в Web Audio API обрабатывается внутри *аудио контекста (audio context)*. Интерфейс AudioContext — это основа Web Audio, он предоставляет необходимые для создания различных элементов Web Audio функции и способ передачи звука «железу» и устройствам вывода звука.

Каждая базовая операция выполняется с аудио *узлами (nodes)*, которые связаны между собой и образовывают *граф маршрутизации аудио (audio routing graph)*. Перед воспроизведением любого звука, вам необходимо создать этот аудио контекст. Это очень похоже на то как вы создаёте контекст для рисования внутри элемента <canvas>.

Помимо предоставления необходимых функций, этот интерфейс обладает двумя важными свойствами: destination и listener, оба read-only. Свойство destination можно представить как средство связи с аудио-аппаратурой. Свойство listener представляет собой тот объект, который слушает весь звук в игре, например, персонаж или камера.

Существует три типа источников:

1. Осциллятор - математически вычисляемые звуки;
2. Аудио сэмплы - из аудио/видео файлов;
3. Аудио поток - аудио из веб-камер или микрофонов;

## Давайте начнём с осциллятора

Осциллятор - это повторяющийся сигнал. У него есть частота и пиковая амплитуда. Одна из наиболее важных характеристик осциллятора, помимо частоты и амплитуды это форма его сигнала. Четыре наиболее часто используемых форм осциллятора это синусоидальная, треугольная, квадратная и зубчатая.

### **Перед тем как начать**

Как уже упоминалось, поддержка Web Audio API не является универсальной, поэтому лучше проверить, что API доступен в браузере пользователя.

После этой простой проверки мы можем использовать функциональность Web Audio API.

### **Простой генератор**

Чтобы узнать, какие звуки он может генерировать самостоятельно, давайте использовать audioContext для создания OscillatorNode:

const oscillator = audioContext.createOscillator();

Это все, что нам нужно для создания звука в браузере - AudioContext и OscillatorNode. Но сначала нам нужно «подключить» генератор к нашему аудиоконтексту:

oscillator.connect(audioContext.destination);

API веб-аудио пытается имитировать аналоговую цепочку сигналов. Мы подключаем наш входной сигнал (генератор) к цифровому усилителю мощности (audioContext), который затем передает сигнал на динамики.

Давайте запустим наш генератор:

oscillator.start();

Вы должны услышать звук, сопоставимый с гудком. Поздравляем, вы создаете музыку с помощью Web Audio API! Конечно, никто не хочет слышать один и тот же звук. Вы можете остановить наш генератор таким образом:

oscillator.stop();

Как только AudioNode остановлен, его нельзя запустить снова! Для возобновления воспроизведения необходимо создать новый AudioNode.

Методы start и stop принимают один параметр типа number. Значение параметра используется для планирования событий запуска / остановки:

/\* Воспроизвести звук через 10 секунд \*/

oscillator.start(audioContext.currentTime + 10);

/\* Выключить звук через 10 секунд \*/

oscillator.stop(audioContext.currentTime + 20);

### **Управление звуком с помощью AudioParams**

Регистрируя объект oscillator, мы получаем что-то вроде этого (конкретные значения свойств опущены, поскольку они могут быть разными в зависимости от устройства / браузера):

console.log(oscillator);

/\*

{

channelCount: number,

context: AudioContext,

detune: AudioParam,

type: 'sine' | 'sawtooth' | 'triangle' | 'square'

frequency: AudioParam,

numberOfInputs: number,

numberOfOutputs: number,

onended: function

...

}

\*/

Свойство, которое имеет наибольшее значение для наших целей, - это oscillator.frequency

console.log(oscillator.frequency);

/\*

{

defaultValue: number,

maxValue: number,

minValue: number,

value: number // Probably 440 (A4)

}

\*/

Значение frequency нашего генератора реализует интерфейс AudioParam. Звуком AudioNode, можно манипулировать с помощью его свойств AudioParam. Однако прямое переназначение свойства значения AudioParam устарело в пользу вспомогательных методов.

/\* Не делайте этого \*/

oscillator.frequency.value = 500;

Если мы хотим, чтобы наш oscillator излучал «Bb» вместо «A», мы должны сделать что-то вроде этого:

/\* Частота (в Hz) Bb4 равна 466,16 \*/

oscillator

.frequency

.setValueAtTime(466.16, audioContext.currentTime);

или

/\* Медленный переход на Bb4 в течение 10 секунд \*/

oscillator

.frequency

.exponentialRampToValueAtTime(

466.16,

audioContext.currentTime + 10

);

### **Вывод**

Создание и управление аудио в браузере проще, чем когда-либо, благодаря Web Audio API. С его помощью веб-разработчики могут воссоздавать ретро-сигналы в 3-5 строчек кода.