**Урок №13. Подключение аудио**

**Вводный блок (5 минут)**

**Цели и задачи урока:**

1. Научимся подключать аудио в проект.
2. Создадим звуковые эффекты для действий персонажа.

**Познание нового (10 минут)**

В играх, когда происходят какие-либо действия, обычно это сопровождается звуковыми эффектами. Сдача квеста, поднятие предмета, прыжок - все может сопровождаться какими-либо звуками.

На этом занятии мы загрузим звуковые эффекты и применим их для действий персонажа.

Начнем с создания audio:

const canvas = document.querySelector('canvas');

const context = canvas.getContext('2d');

hitsound = new Audio("hit.wav");

hitsound.addEventListener('loadeddata', () => {

hitsound.play();

})

Помимо этого, в любой момент аудио можно поставить на паузу. Для этого необходимо воспользоваться командой:

hitsound.pause();

При этом, если будет остановка, то если снова воспроизвести трек, то он начнется с той же секунды, где он и был остановлен.

Формат аудио может быть mp3, ogg и wav.

В играх чаще всего используют wav формат, поскольку он обладает самым высоким качеством.

**Работа над проектом (25 минут)**

Тема: Создание нового проекта и написание программы, в которой добавляются звуковые эффекты и создается аудиоплеер.

**Задание**

1. Создайте монетку в виде объекта, который должен содержать следующие свойства: изображение, звук сбора монетки, координаты и размеры (13.1)
2. Создайте персонажа в виде объекта, который должен содержать следующие свойства: изображение, координаты и размеры. Добавьте управление для персонажа (13.1)

Создаем коллизию и звуковое сопровождение взятия монетки (13.2)

function move() {

player.x += dx;

player.y += dy;

**if (collide(player, coin)){**

**coin.sound.play();**

**}**

context.clearRect(0, 0, canvas.width, canvas.height);

context.drawImage(coin.image, coin.x, coin.y, coin.w, coin.h);

context.drawImage(player.image, player.x, player.y, player.w, player.h);

requestAnimationFrame(move)

}

Функция collide (функцию можно дать готовую ученикам) проверяет все крайние точки прямоугольника, в котором располагаются объекты и срабатывает тогда, когда идет пересечение.

Функция является универсальной и в дальнейших занятиях также будет использована:

function collide(obj1, obj2) {

if (obj1.x > obj2.x

&& obj1.x < obj2.x + obj2.w

&& obj1.y > obj2.y

&& obj1.y < obj2.y + obj2.h

) {

return true;

}

if (obj1.x + obj1.w > obj2.x

&& obj1.x + obj1.w < obj2.x + obj2.w

&& obj1.y > obj2.y

&& obj1.y < obj2.y + obj2.h

) {

return true;

}

if (obj1.x + obj1.w > obj2.x

&& obj1.x + obj1.w < obj2.x + obj2.w

&& obj1.y + obj1.h > obj2.y

&& obj1.y + obj1.h < obj2.y + obj2.h

) {

return true;

}

if (obj1.x > obj2.x

&& obj1.x < obj2.x + obj2.w

&& obj1.y + obj1.h > obj2.y

&& obj1.y + obj1.h < obj2.y + obj2.h

) {

return true;

}

return false;

}

**Некомпьютерная активность (10 минут)**

Музыкальные волны:

1. Возьмите листы бумаги или картона и маркеры разных цветов.
2. Попросите участников нарисовать простые волнообразные линии, которые будут представлять звуковые волны.
3. Включите музыку и попросите участников двигать и изменять форму своих "звуковых волн" в соответствии с музыкой.
4. Обсудите, как в JavaScript Canvas можно визуализировать аудио, анализируя и отображая его характеристики, такие как амплитуда и частота.

**Работа над проектом (35 минут)**

**Задание:** Используя полученные навыки о подключении аудио, создайте свой аудиоплеер, который будет работать по нажатию на кнопки по следующим правилам:

1. Кнопка пробел отвечает за старт и остановку аудиофайла
2. Кнопка N отвечает за переход на следующий трек
3. Кнопка P отвечает за переход на предыдущий трек

Помимо этого, на экран выводить наименование текущей песни (13.3)

****

**Примерный интерфейс**

**Рефлексия (5 минут)**

"Что нового вы узнали сегодня?"

"Как подключать Audio к проекту? Какие есть команды для управления Audio?"