Analisador de frequências e notas musicais

Até o momento deste tutorial, e pelas pesquisas que realizei, não há ferramentas dentro do próprio Construct 3 que auxiliem na captação de frequência de uma nota musical. O que torna difícil a produção de jogos musicais que verifiquem as notas tocadas. Então para isso, usaremos a combinação Construct 3 + Java Script.

Mesmo sabendo que eu mostrarei aqui apenas como aplicar e modificar o código JS, e não cria-lo do zero, recomendo os próximos passos para quem já tenha um domínio razoável da ferramenta Construct.

Neste tutorial, ensinarei como verificar qual nota esta sendo tocada. Porém, utilizei como base um único instrumento, a flauta doce. Então será utilizada a seguinte tabela de frequência com suas notas correspondentes:

N°	Nota	Frequência (Hz)	Período (s)	Comprimento de Onda (m)
60	C 4	523.251099	0.001911	0.657428
61	C# 4	554.365234	0.001804	0.620529
62	D 4	587.329529	0.001703	0.585702
63	D# 4	622.253906	0.001607	0.552829
64	E 4	659.255127	0.001517	0.521801
65	F 4	698.456482	0.001432	0.492515
66	F# 4	739.988831	0.001351	0.464872
67	G 4	783.990845	0.001276	0.438781
68	G# 4	830.609375	0.001204	0.414154
69	A 4	880.	0.001136	0.390909
70	A# 4	932.327576	0.001073	0.368969
71	B 4	987.766602	0.001012	0.34826

Para quem é leigo no assunto, a tradução das notas seria algo como:

C4 = DÓ

D4 = RÉ

E4 = MI

F4 = FÁ

G4 = SOL

A4 = LÁ

B4 = SI

Se você deseja usar outro instrumento, basta trocar os valores de frequência para que eles estejam de acordo com o instrumento desejado.

Com a tabela em mãos, vamos começar a desenvolver o nosso verificador de notas.

Com base no seguinte código

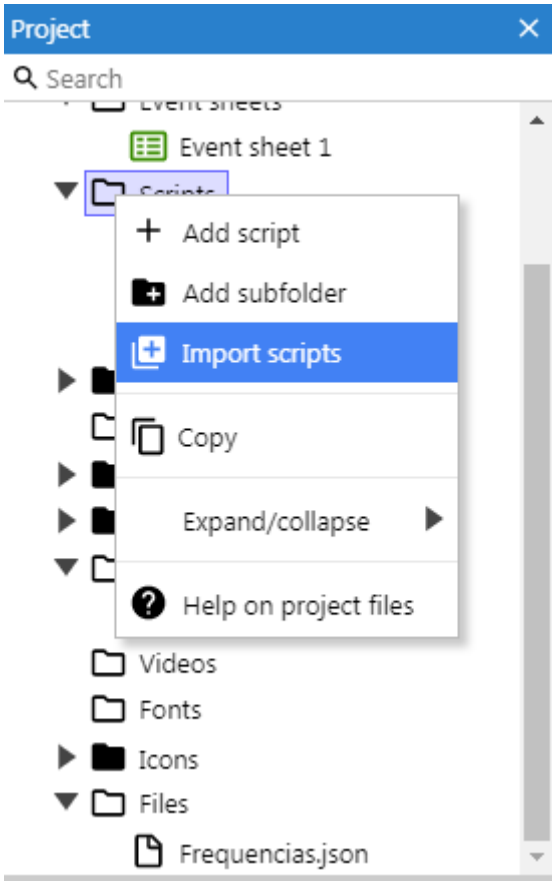
https://codepen.io/michaeljancsy/pen/OPrBGa

Criei adaptações para que ele possa ser utilizado no Construct 3 sem muitas complicações, então para esse tutorial, baixe a minha versão e os outros dois códigos de complemento (adaptados do modelo presente no Construct 3)

https://github.com/BCBr/ConstructPitchAnaliser

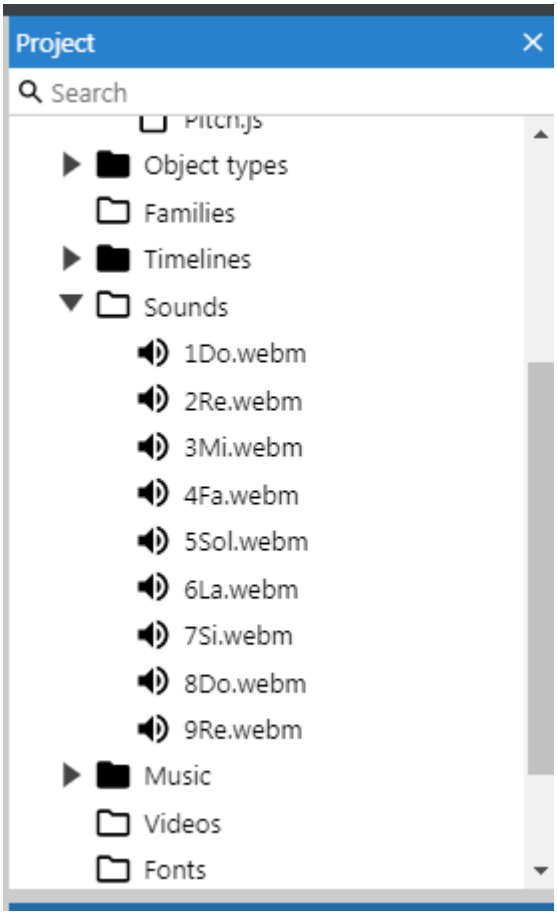
Se quiser pode baixar o projeto completo caso tenha duvidas.

Crie um novo projeto no Construct, e insira os códigos adaptados na pasta "Scripts":



Na mesma pasta, crie um script com o nome "Main". Nele iremos adicionar as funções que poderão ser chamadas nos "Event Sheets".

Importe os áudios do seu projeto para a pasta "Sounds". Nesse tutorial usaremos os próprios sons das notas na flauta.



Clique no script “Main” que você criou. Primeiro precisamos iniciar o script "AudioManager" e carregar todos os arquivos de áudio que serão analisados. Para isso siga os seguintes passos:

-1º Defina uma variável do tipo "let" com o nome de "audioManager" e valor "null":

```
1
2  // Audio manager class for handling audio playback.
3  let audioManager = null;
4
```

-2º Para cada arquivo de som que você deseja verificar a frequência, crie uma variável do tipo "let", no caso para exemplificar, irei fazer só o "DÓ" e o "RÉ" também com valor "null":

```
4
5  // References to loaded audio files as global variables
6  let DO = null;
7  let RE = null;
8
```

3º Chame a função "runOnStartup" do Construct para iniciar o carregamento dos sons assim que o jogo for iniciado:

```
runOnStartup(async runtime =>
{

});
```

4º Dentro dela, instancie o "AudioManager" junto com o "runtime"

```
runOnStartup(async runtime =>
{
    // Initialise the audio manager. See AudioManager.js for details.
    audioManager = new AudioManager(runtime);

});
```

5º Escreva o seguinte código para carregar os arquivos de áudio desejados e coloca-los em suas respectivas variáveis (não se esqueça de colocar o nome do arquivo de áudio e sua extensão dentro da função "loadSound()")

```
    // During the loading screen, load both sound files as
    // AudioBuffers and the music track all in parallel, so
    // they are ready for immediate playback on startup.
    [DO, RE] = await Promise.all([
        audioManager.loadSound("1Do.webm"),
        audioManager.loadSound("2Re.webm"),
    ]);
```

Pronto, no final você vai ter um script parecido com isso:

```
runOnStartup(async runtime =>
{
    // Initialise the audio manager. See AudioManager.js for details.
    audioManager = new AudioManager(runtime);
    //criaAudioContext();

    // During the loading screen, load both sound files as
    // AudioBuffers and the music track all in parallel, so
    // they are ready for immediate playback on startup.
    [DO, RE] = await Promise.all([
        audioManager.loadSound("1Do.webm"),
        audioManager.loadSound("2Re.webm"),
    ]);
});
```

Isso garante o pré carregamento dos seus áudios.

A segunda fase é definir uma função pra cada arquivo de áudio carregado na fase anterior. Assim, poderemos chama-la pelo "Event Sheets".

A função deve seguir a seguinte estrutura

```
function NomeDaFuncao()
{
        setMainBuffer("variavel onde o arquivo de som foi carregado");
        setAnaliser();
        audioManager.playSound("variavel onde o arquivo de som foi carregado");
}
```

Seguindo as variáveis feitas nesse tutorial, faremos duas funções assim:

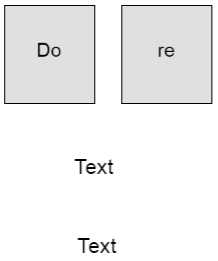
```
26  function p1()
27  {
28      setMainBuffer(DO);
29      setAnaliser();
30      audioManager.playSound(DO);
31  }
32
33  function p2()
34  {
35      setMainBuffer(RE);
36      setAnaliser();
37      audioManager.playSound(RE);
38  }
```

Terminamos a parte de script JS, se você esta seguindo os passos corretamente, seu script deve estar parecido com isso:

```
2   // Audio manager class for handling audio playback.
3   let audioManager = null;
4
5   // References to loaded audio files as global variables
6   let DO = null;
7   let RE = null;
8
9
10  runOnStartup(async runtime =>
11  {
12      // Initialise the audio manager. See AudioManager.js for details.
13      audioManager = new AudioManager(runtime);
14      //criaAudioContext();
15
16      // During the loading screen, load both sound files as
17      // AudioBuffers and the music track all in parallel, so
18      // they are ready for immediate playback on startup.
19      [DO, RE] = await Promise.all([
20          audioManager.loadSound("1Do.webm"),
21          audioManager.loadSound("2Re.webm"),
22      ]);
23  });
24
25  // These functions are called by the button click events.
26  function p1()
27  {
28      setMainBuffer(DO);
29      setAnaliser();
30      audioManager.playSound(DO);
31  }
32
33  function p2()
34  {
35      setMainBuffer(RE);
36      setAnaliser();
37      audioManager.playSound(RE);
38  }
```

Agora ensinarei como chamar as funções que criamos e analisar a frequência dos sons.

Crie no "layout" os objetos que serão utilizados em cena. Aqui eu criei um botão para cada. Também crie duas caixas de texto que serão utilizadas para mostrar a nota e a frequência do som que esta sendo tocado.



Além disso, crie um objeto AJAX para ficar mais fácil montar a sua "array". Depois crie uma "array" Json. Lembra da tabela no inicio do tutorial? É agora que ela entra. Como é MUITO difícil um som chegar numa frequência exata, normalmente o som de uma nota varia num intervalo de frequência, por isso, utilizaremos a tabela anterior como base para criarmos outra com um intervalo aproximado pra mais e pra menos que corresponderam a nota desejada.

Cada linha (y) pertence a uma nota musical.

Na 1ª coluna (x = 0) é onde esta o nome da nota.

Na 2ª coluna (x = 1) esta o valor mínimo que a frequência pode chegar para ser considerada dentro da nota musical.

Na 3ª coluna (x = 2) esta o valor máximo que a frequência pode chegar para ser considerada dentro da nota musical. Ou seja, o valor da frequência ficar entre o mínimo e o máximo da nota, quer dizer que o som está na frequência da nota.

E na 4ª coluna (x = 3) esta o valor de o qual tiramos como base os outros valores, ele não é utilizado no código.

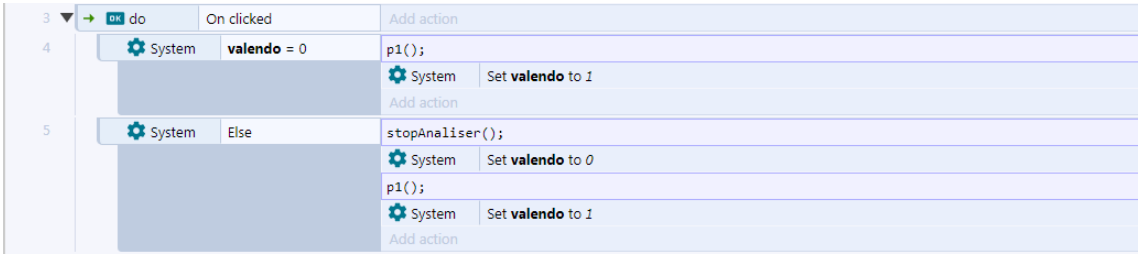
Width 4 Height 9 Depth 1 Sheet 0

	0	1	2	3
0	DÓ	513	535	523
1	RÉ	570	605	587
2	MI	645	671	659
3	FA	687	712	698
4	SOL	766	800	783
5	LA	860	900	880
6	SI	970	1002	987
7	DÓ+	1030	1060	1046
8	RÉ+	1155	1210	1174

Carregue o Json numa “array” com o AJAX no "On Start".

Crie uma variável global do tipo numérica, com o nome de "tocando". Vamos definir que se ela tiver valor 0, o som não esta tocando, e se ela tiver o valor 1, quer dizer que é para o som estar tocando.

Em seguida, crie as seguintes ações e eventos. (para chamar uma função JS como na imagem abaixo, clique com o botão direito em "Add action" e em "Add script", então escreva a chamada de função)

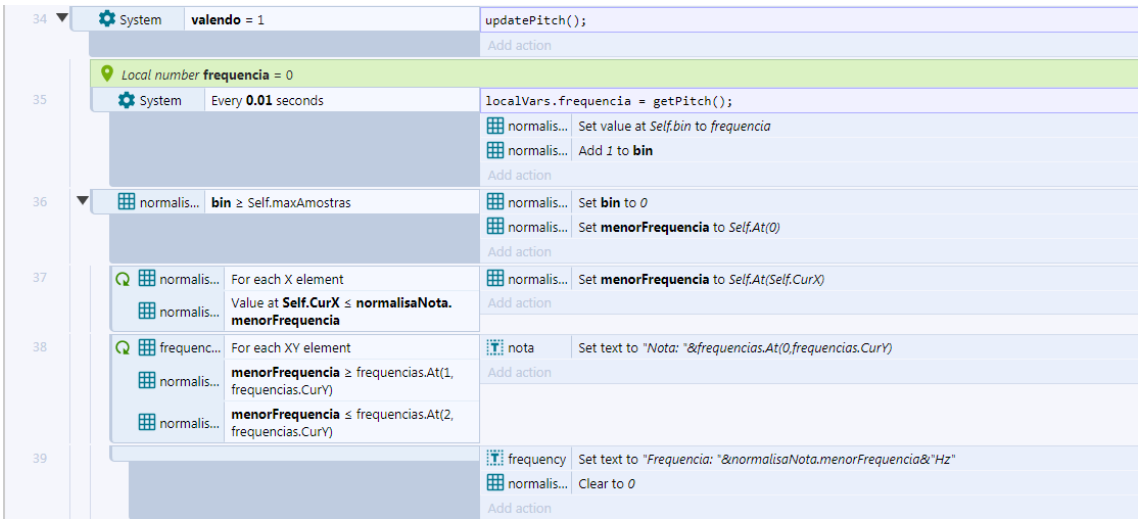


Em suma, o que acabamos de fazer foi dizer que quando o botão "do" for clicado, ele vai verificar se tem algum som tocando, se não estiver tocando, ele vai chamar a função js "p1()" e vai definir a variável tocando como 1. Caso algum som já esteja sendo tocado, ele vai chamar a função "stopAnaliser()" que ira limpar o cache do "Analiser" e interrompe-lo, em seguida vai setar "tocando = 0" para definir que ele parou de tocar, chamar "p1()" e definir

"tocando = 1". Essas são as funções que fizemos no JS, portanto, elas vão ativar o som que definimos para elas.

Até aqui já conseguimos fazer o som da nota tocar e pode ser analisado, porem, a analise de frequência da nota ainda não será precisa e nem será mostrada. Antes de arrumarmos isso, leve em consideração que a frequência aparecera com ruídos, por exemplo se estiver captando um valor por volta de 523 que é o valor de "Dó", as vezes ela poderá captar um valores de 1000 pra cima, o q não tem nada haver com a nota desejada. Para solucionarmos isso, vamos analisar um conjunto de 10 valores no intervalo de 0,1 segundos.

Para que isso funcione, crie um objeto "array" [10,0,0], crie uma variável com o nome de "maxAmostras" com valor "10", crie também duas variáveis "bin" e "menorFrequencia" com valor 0. Por fim, repita os eventos abaixo.



Esses eventos farão o seguinte:

-1º Se o som estiver tocando, será chamada a função "updatePitch();" JS que analisa a frequência em tempo real.

-2º A cada 0,01 segundo é setada a variável local "frequencia" com o valor da frequência atual provinda da função "getPitch()" (localVars.frequencia = getPitch();).

-3º Ainda a cada 0,01 segundo é colocado o valor da "frequencia" em cada lugar da array anteriormente criada.

-4º Quando conseguir o numero de amostras o suficiente para definir a nota, será verificada qual é a menor frequência dentre das 10 amostras.

-5º Com a menor frequência, verificamos na nossa tabela de notas em qual intervalo a frequência esta, e com base nesse intervalo é possível saber qual a nota que está tocando.

-6º Mostra no objeto de texto a nota e a frequência do som que esta sendo tocado.

E pronto, com essa base, é possível verificar a frequência e suas notas correspondentes. Você também pode adaptar esse código para pegar a frequência de um microfone e verificar se esta acertando as notas de sua flauta. Espero que tenham gostado.