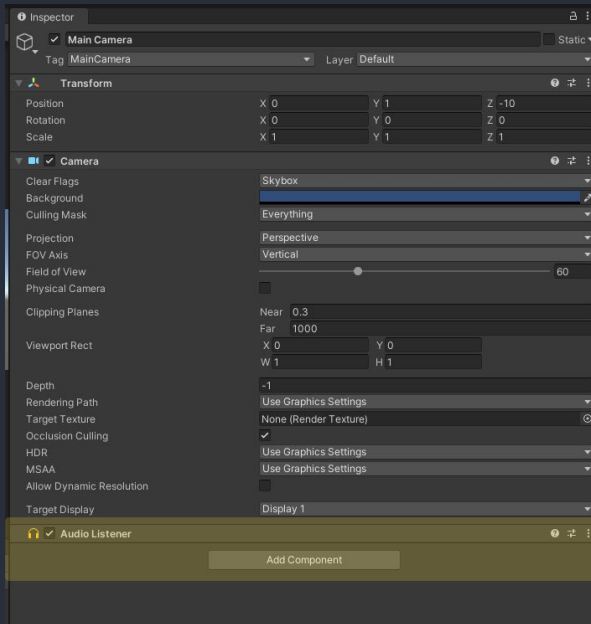


Unity

Platformer 2D: Adicionando Som



Audio Listener

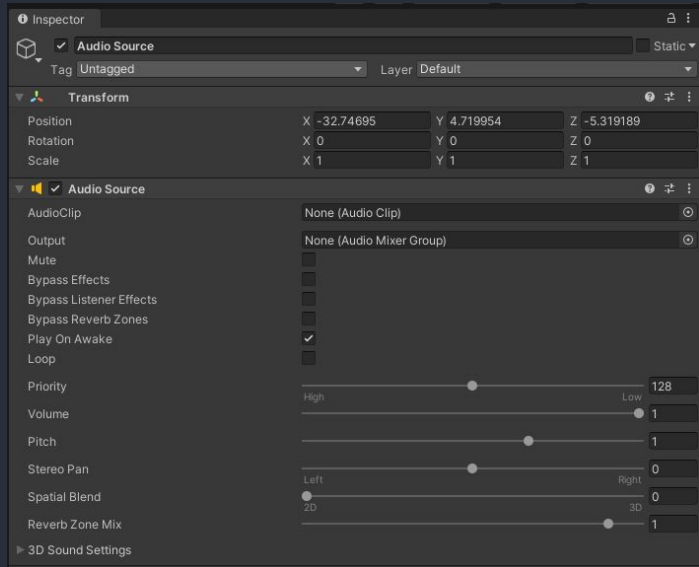


A primeira coisa que precisamos conhecer, para entender como o sistema de áudio funciona, é:

*É necessário ter um **ouvinte** e um **emissor** para que algum som seja tocado.*

*Na Unity, o ouvinte é o componente chamado **Audio Listener**. Geralmente ele está presente na câmera da cena. Afinal, queremos que o ouvinte esteja posicionado no mesmo lugar que a câmera. Dessa forma, faz sentido ele estar colocado no mesmo gameobject.*

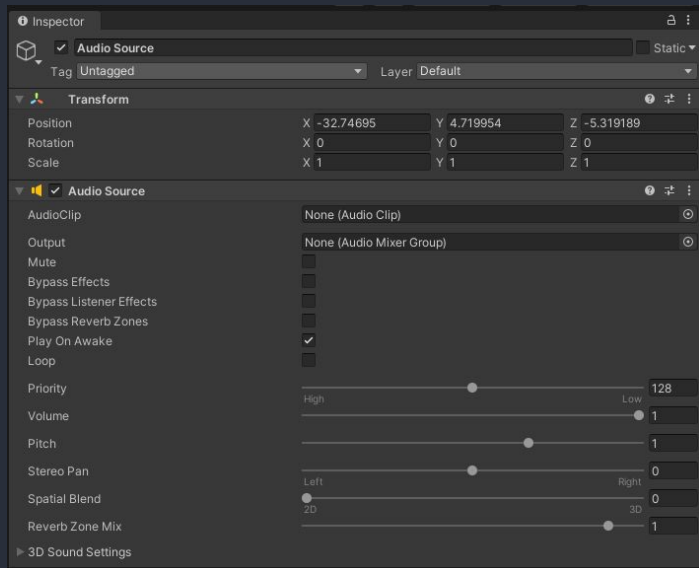
Audio Source



Podemos ter diferentes tipos de emissores, porém o mais comum é o **Audio Source**.

Ele é o componente que vai “tocar” algum som.

Audio Source



Principais funcionalidades

AudioClip: Aqui colocamos o nosso arquivo de música que será tocado.

Output: podemos definir qual Audio Mixer Group vamos usar (vamos ver mais a frente sobre)

Play on Awake: o arquivo tocará automaticamente quando a cena começar rodando, no Awake.

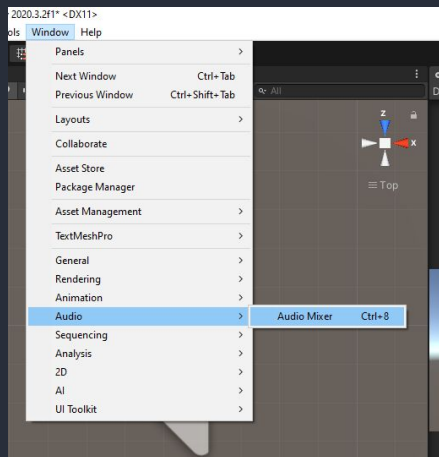
Loop: tocaremos esse som em repetição?

Audio Source

Para tocarmos um som do AudioSource por código, podemos utilizar a seguinte maneira:

```
public AudioSource audioSource;  
  
1 reference  
public void Play()  
{  
    audioSource.Play();  
}
```

Audio Mixer



Podemos ir mais a fundo no setup e configuração do áudio na Unity. Idealmente, precisamos sempre pensar na manutenção futura do projeto. Tanto em código, quanto em arte, e som.

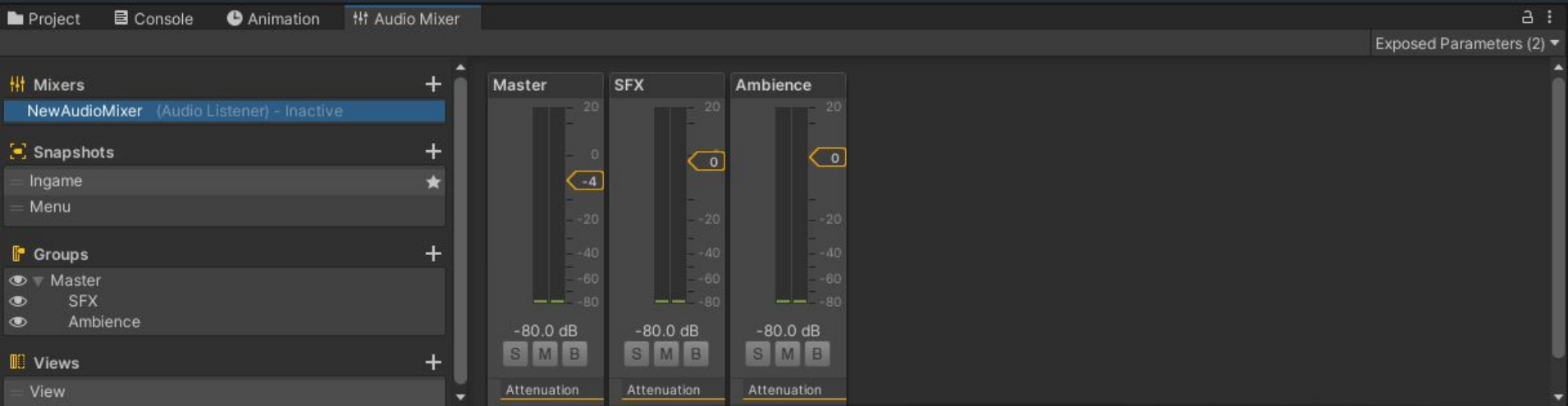
*Pensando nisso, temos uma ferramenta bastante útil e importante: o **Audio Mixer**.*

Com ele, podemos manter a configuração do som isolada da implementação, facilitando para o compositor do projeto realizar os ajustes necessários, e testá-los no projeto.

Para abrir o Audio Mixer, podemos utilizar as teclas CTRL + 8, ou abrir pelo menu ao lado.

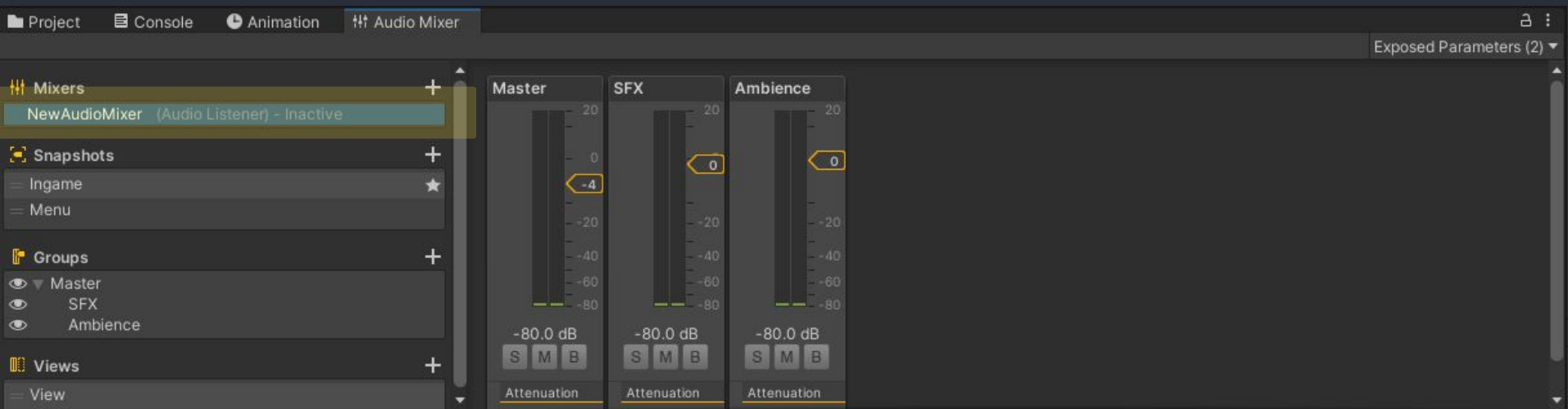
Audio Mixer

Abaixo podemos ver a janela do Audio Mixer.



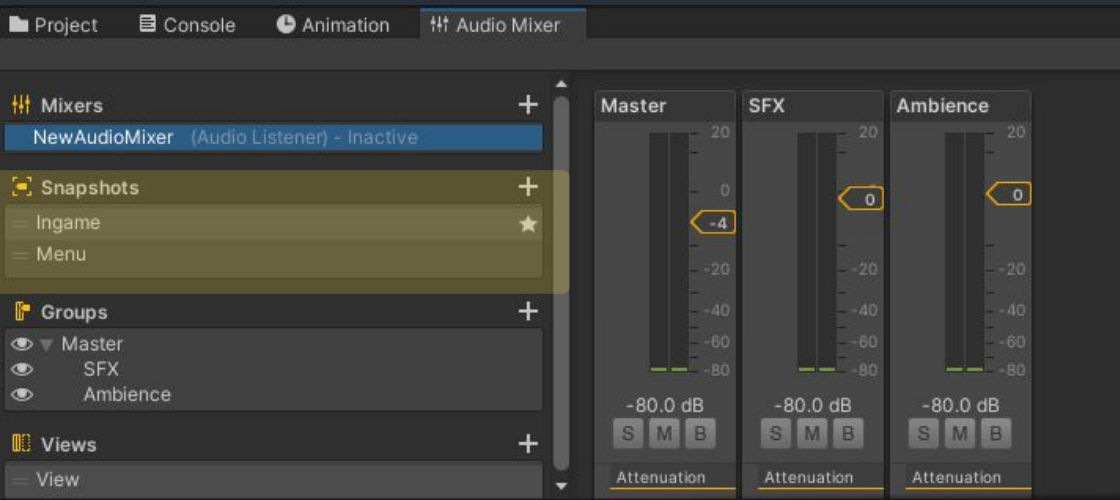
Audio Mixer

Aqui é onde adicionamos os **mixers**. Podemos ter mais de um mixer no jogo. Cada mixer funciona como um “pacote” de Snapshots, Groups e Views.



Audio Mixer

Snapshots: funcionam como uma “memória” das configurações. Vamos supor que queremos 2 comportamentos diferentes do nosso áudio no jogo: em um o group SFX fica mais baixa, e em outro, o Ambience fica mais baixo. Podemos salvar como 2 diferentes Snapshots, e ativarmos eles no código, fazendo a transição, como mostra a função abaixo:



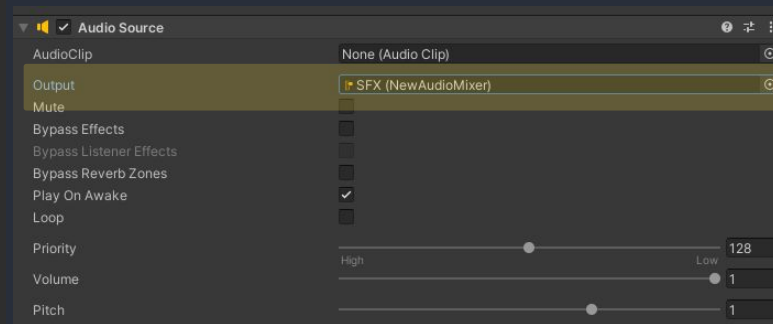
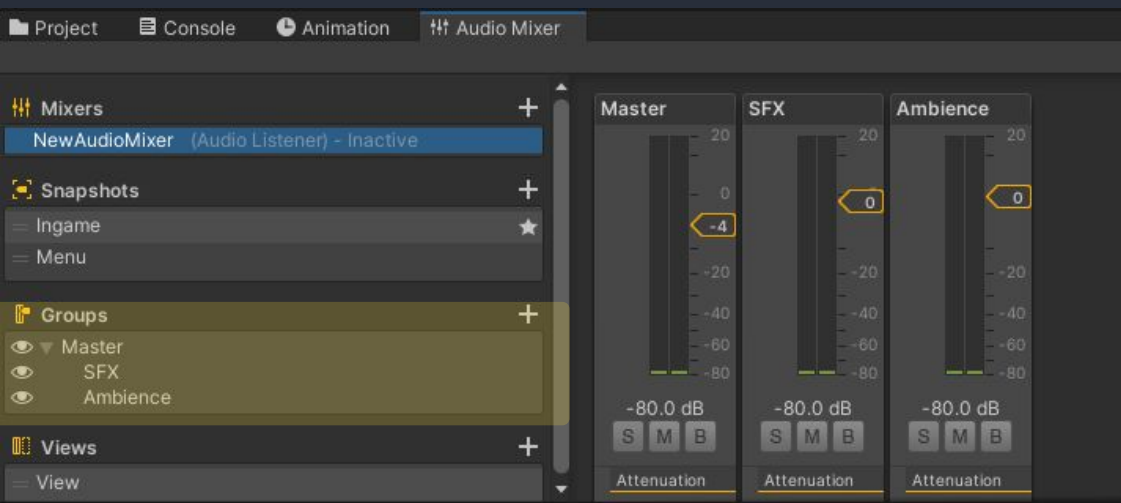
```
Unity Script | 0 references
public class AudioTransition : MonoBehaviour
{
    public AudioManagerSnapshot snapshot;
    public float transitionTime = .1f;

    0 references
    public void MakeTransition()
    {
        snapshot.TransitionTo(transitionTime);
    }
}
```

Audio Mixer

Groups: são as “tracks” do nosso jogo. Geralmente, é utilizado groups como Music, SFX e Ambience.

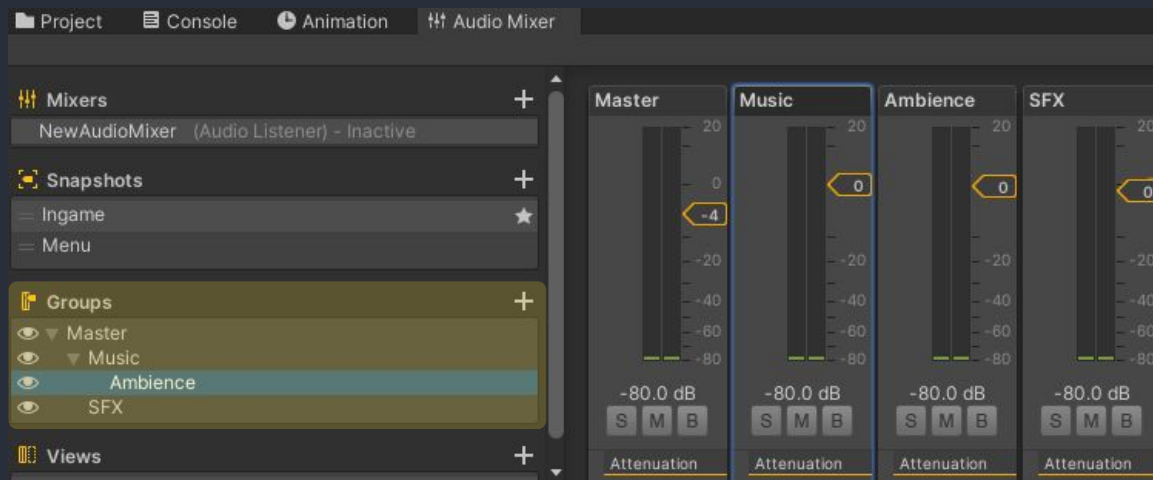
Quando utilizamos um Audio Source para tocar o nosso som, podemos seleccionar em qual Group esse som será tocado. Ou seja, qual é a saída de áudio, ou o Output.



Audio Mixer

Groups: Podemos também alterar a ordem de influência entre os grupos, arrastando pra cima/baixo na lista. Isso faz com que possamos controlar vários grupos de uma vez, se for necessário.

No exemplo abaixo, criamos um grupo novo, chamado Music. Ele é agora pai do Ambience. Ou seja, sempre que mudarmos o volume do grupo Music, o Ambience terá o seu volume afetado também.

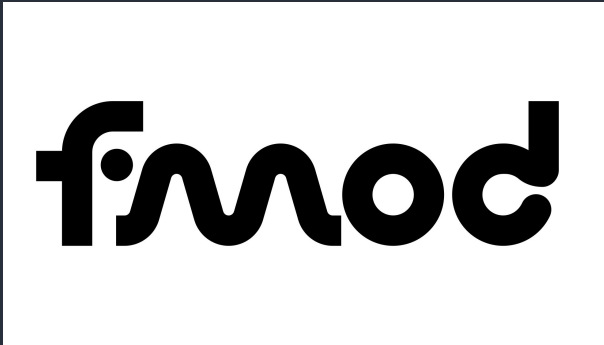


Middlewares

Além do sistema padrão da unity para som, no mercado profissional existem soluções bastante robustas que costumam ser utilizadas.

*Essas soluções são chamadas de **Middlewares**. Ou seja, no caso do audio, uma solução intermediária, conectando a Unity com outro software robusto específico para audio.*

Dentre os mais usados, com certeza podemos destacar os 2 abaixo.



Para mais informações, clique no logo de cada software.

Links úteis

<https://docs.unity3d.com/Manual/AudioOverview.html>

<https://docs.unity3d.com/Manual/class-AudioSource.html>

<https://learn.unity.com/tutorial/working-with-audio-components-2019-3>