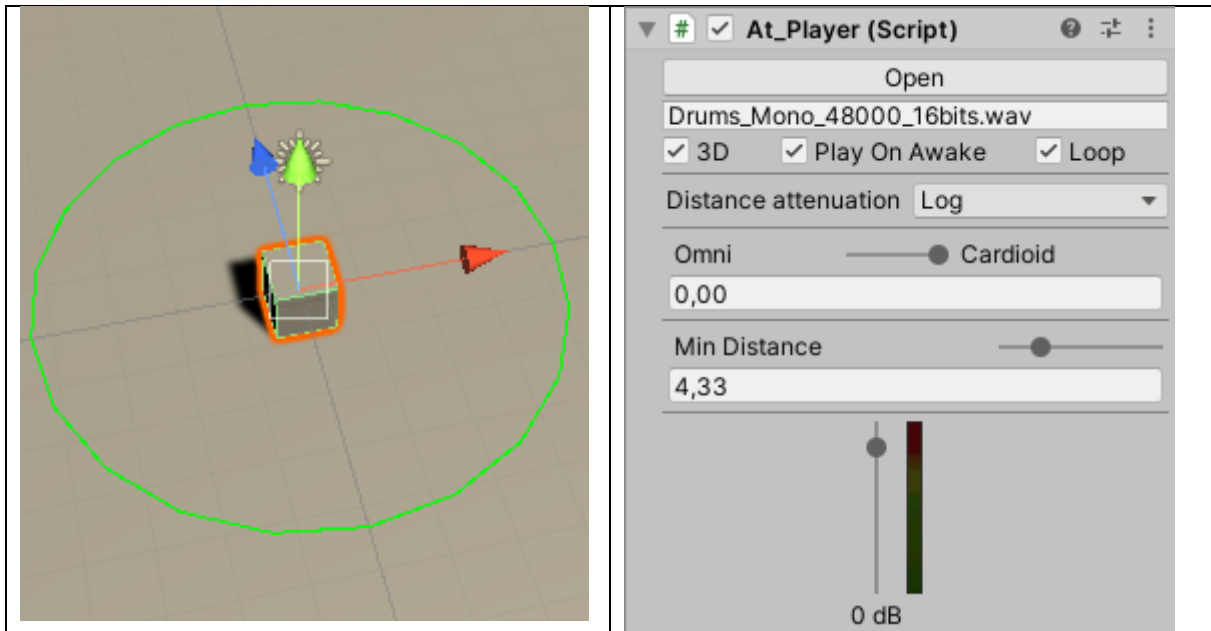


At_3DAudioEngine - Unity

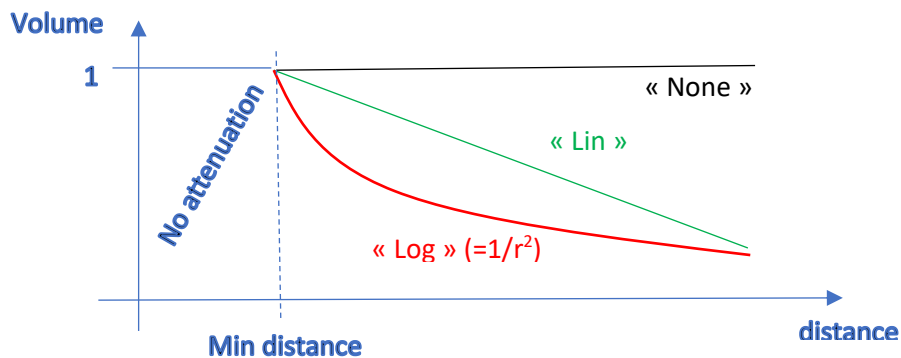
What's new ?– version.2.1 Alpha – 20/07/2021

1 - At_Player

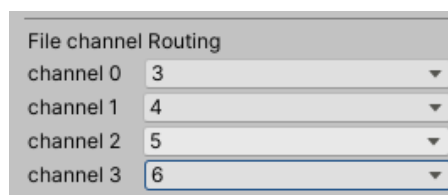
The At_Player has now a green circle) representing the minimum distance (only displayed in the scene view) when the player is “3D”.



This parameter influences the decay of the source volume with distance according to the roll-off curve, which is defined by the “Distance attenuation” parameter (“None”, “Lin” or “Log”) :



For “2D” player, you can now decide to which output channel you want to route and audio file channel.

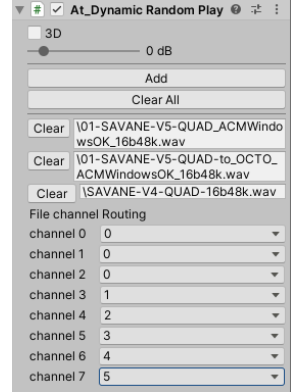


Warning : you must set up the channel routing each time you change the speaker configuration

2- At_DynamicPlayer

We now use two libraries “Ookii.Dialogs” and “System.Windows.Forms” which implement the native file dialog system of Windows (<https://github.com/gkngkc/UnityStandaloneFileBrowser>)

You can then select multiple files at the same time.



The screenshot shows the 'At_Dynamic Random Player' console. It has a '3D' checkbox, a volume slider at 0 dB, and buttons for 'Add' and 'Clear All'. Below these are three file selection buttons, each with a 'Clear' button. The 'File channel Routing' section shows a list of channels (0-7) with dropdown menus for routing. Channel 0 is set to 0, channel 1 to 0, channel 2 to 0, channel 3 to 1, channel 4 to 2, channel 5 to 3, channel 6 to 4, and channel 7 to 5.

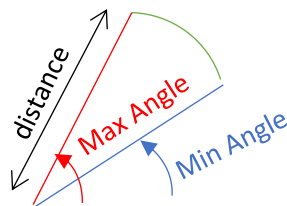
We also add the channel routing if the At_DynamicRandomPlayer is 2D (3D uncheck)

/!\ Because not all audio files have the same amount of channels, the number of channels for routing is the greater number of channel. In this example, there is one file with 8 channels and two files with 4 channels with an 6 channels output. Then 8 channels are displayed for routing.

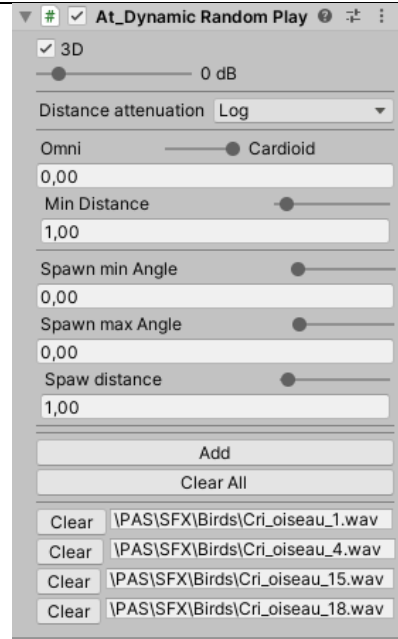
Please avoid this. We will not provide a fine way to independently route the channels of each audio file.

If “3D” is checked we also add new parameters and a debug visualization of the At_DynamicRandomPlayer.

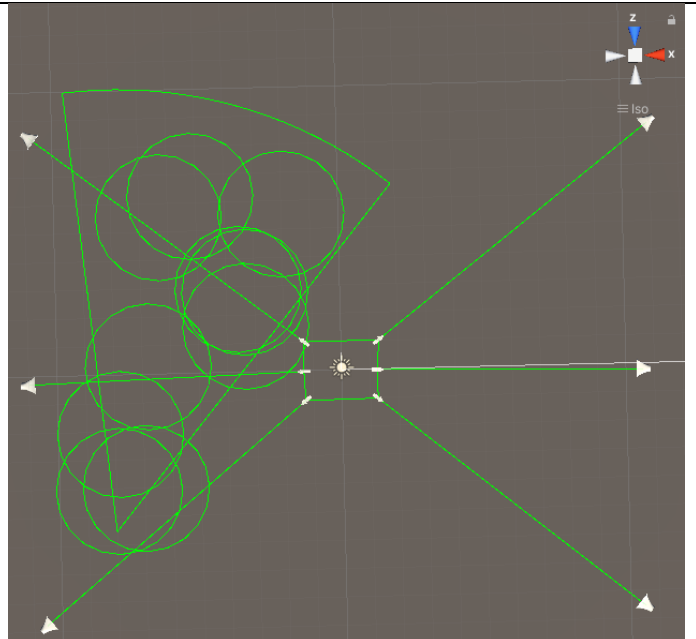
- First of all we now define a zone for limiting a spawning random position of the 3D At_Player generated. This zone is defined by 3 parameters : 2 angles and a distance



- We also add a minimum distance for the generated 3D At_Player
- And the generated 3D At_Player are represented with their green circle of minimum distance
- At last, the GameObject are destroyed when the audio file reach the end.



The screenshot shows the 'At_Dynamic Random Player' console with the '3D' checkbox checked. It includes a 'Distance attenuation' dropdown set to 'Log', a 'Cardioid' directional pattern, and sliders for 'Min Distance' (1.00), 'Spawn min Angle' (0.00), 'Spawn max Angle' (0.00), and 'Spaw distance' (1.00). There are 'Add' and 'Clear All' buttons, and a list of four audio files with 'Clear' buttons next to each.



The screenshot shows the 3D debug visualization in the Unity Hierarchy. It displays a central point with several green circles representing the spawning zone. The circles are connected by lines, forming a complex shape. The 'Iso' button is visible in the top right corner.

3 - At_MasterOutput



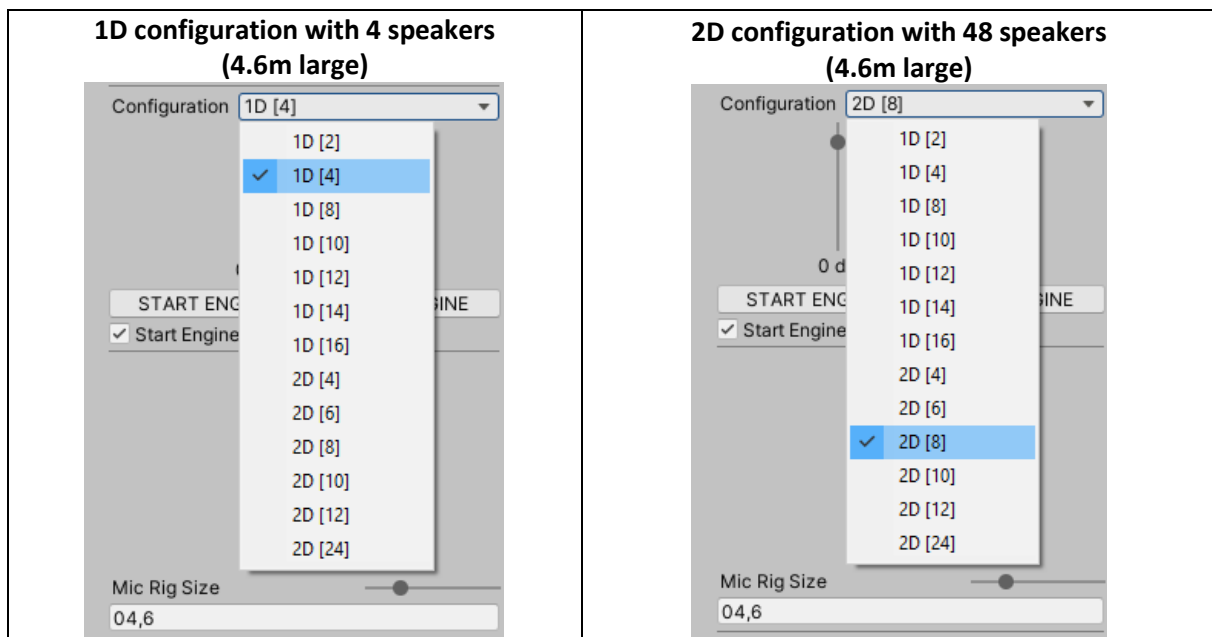
Now, when you add the At_MasterOutput component to an object, it automatically add :

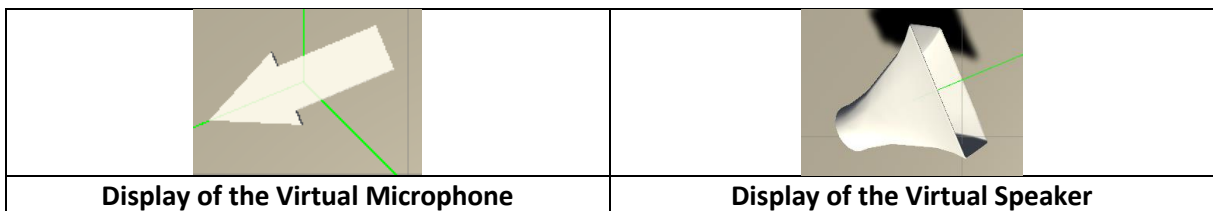
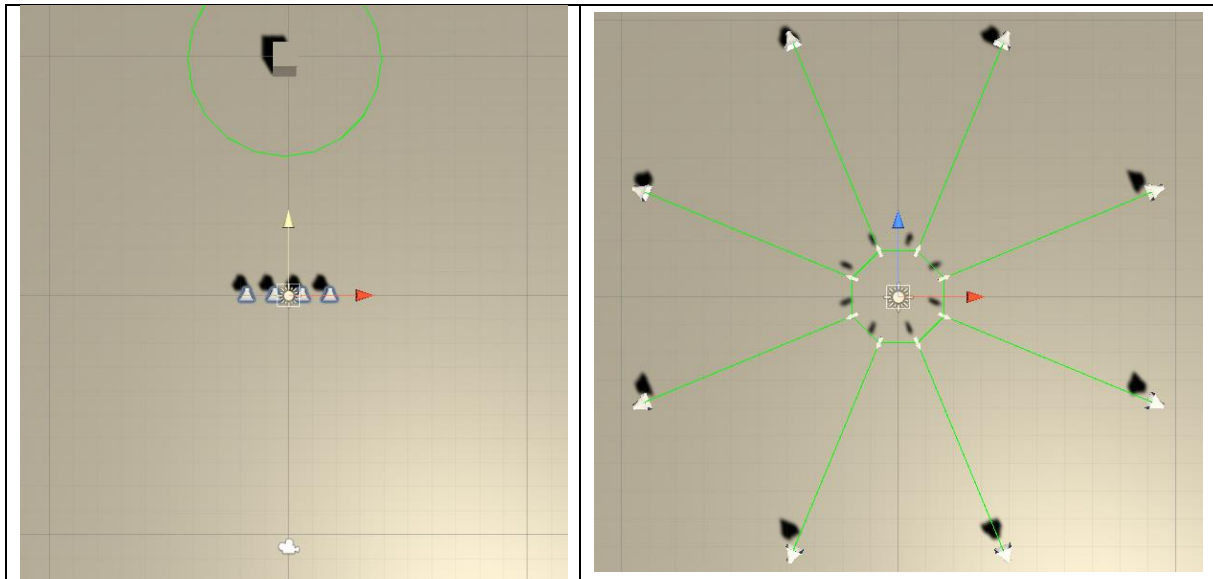
- The At_Mixer component (no change)
- The new **At_ExternAssets** component which is used to define two important external folders :
 - (a) The “Audio Extern Asset Folder” where your audio file are
 - (b) The “States Extern Asset Folder” where all the json file maintaining the state of the audio engine are written and read.

/!\ you absolutely need to set this path before doing anything else. Just clic the buttons and select the corresponding folders in your hard drive.

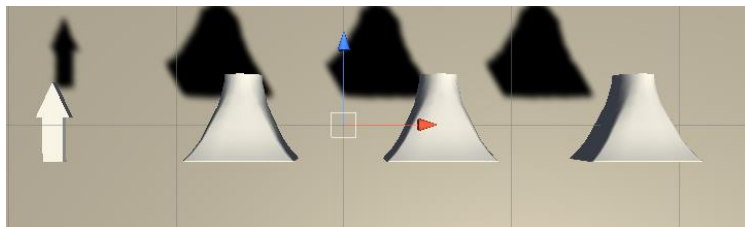
/!\ The paths are saved using the “PlayerPrefs” class of Unity. This mean you must re-build the app if the paths need to be changed.

First, the At_MasterOutput GUI present only speakers configuration suitable for the exhibition (1D and 2D). When a new speaker configuration is chosen, virtual speakers and virtual microphones are automatically added to the scene (see exemple below), according the “Mic Rig Size” parameter (1 Unit = 1 meter).





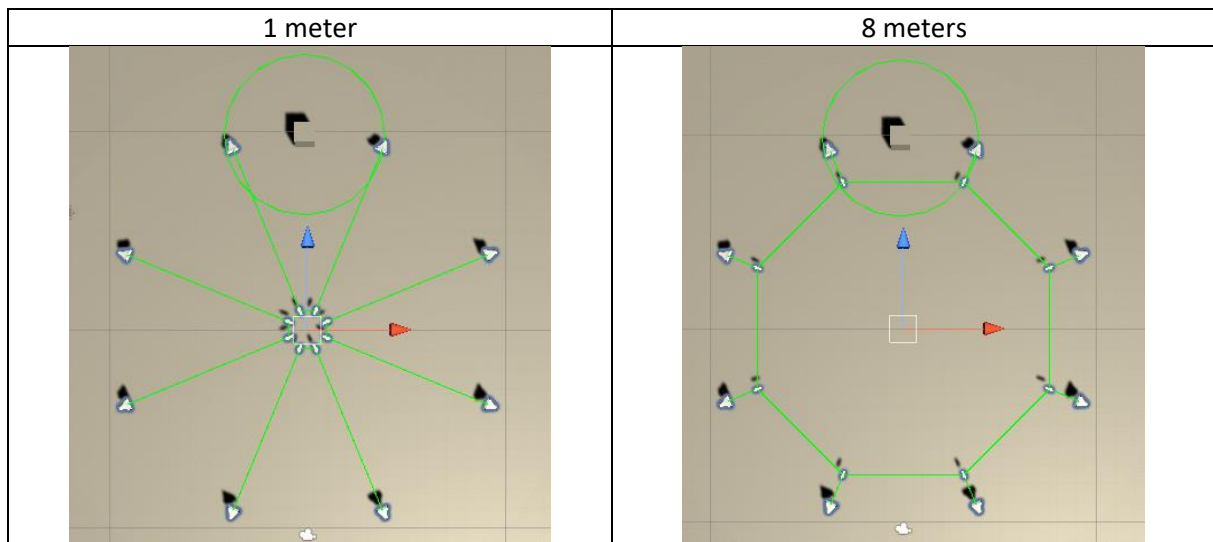
- In 1D configuration, virtual microphone and speakers are coincident (you can't see the Virtual Microphone) :



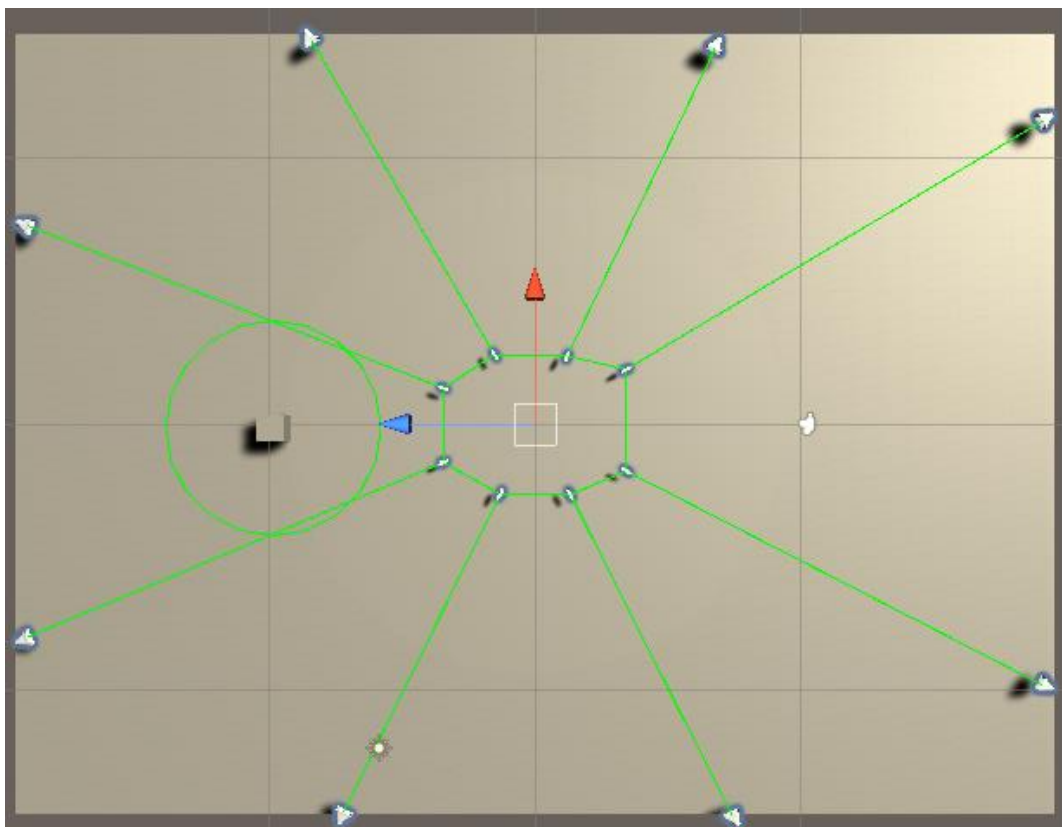
A virtual source should be in front of the virtual microphone (resp. behind the virtual speaker array).

- In 2D configuration, we draw a green polygon passing through all virtual microphone which gives a visual representation of the size of the "Virtual Mic Rig" to the user. Moreover, we draw lines from a virtual microphone to its corresponding virtual speakers. This will help to understand the modification of the shape of the "Virtual Mic Rig" when moving a virtual speaker in space.

First you can adjust the "Mic Rig Size" parameter to fulfill your needs. For example :



When a Virtual Speaker is moved, the shape of the Virtual Microphone Rig will change accordingly to optimize the coherence of the spatialization effects in the room. Feel free to also modify the “Mic Rig Size” parameter to adjust the size of the “Virtual Mic Rig”.



WARNING

When a Speaker config is added for the first time in the scene when selecting a speaker configuration, the GameObject added for Virtual speakers and Microphones will not be saved in the Scene if you close the Unity Editor. However, as long as you move a “Virtual Speaker”, all the GameObjects will be persistent.

