How to create a voice pack using the Custom NPC Voices Framework (U9)

This guide will go through the different steps required for setting up a custom voice pack for the Custom NPC Voices Framework (https://www.nexusmods.com/bladeandsorcery/mods/1298). The guide is split into two parts. Part one covers the basics of setting up Unity and the B&S modding SDK. If you already are up and running with Unity and the SDK, you can jump to part two which covers the creation of your voice pack.

If you should have any questions or improvement suggestion about this guide, feel free to DM or @mpe3us on the B&S Discord.

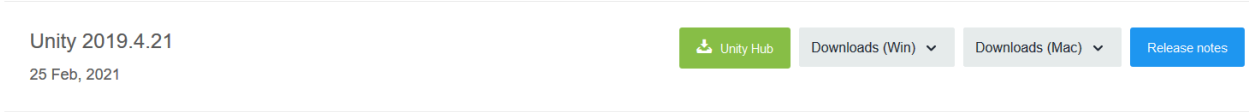
Part 1: Installing Unity and the B&S modding SDK.

1.1 Download & install Unity 2019.3.14f1

B&S is developed using Unity and as such, the modding SDK is based on Unity as well.
For U9 of B&S, the modding SDK uses Unity 2019.4.21f1.

Download link: https://unity3d.com/get-unity/download/archive

Scroll down until you find the correct version:



I recommend installing Unity using Unity Hub as it makes it easier to manage your projects and future Unity versions.

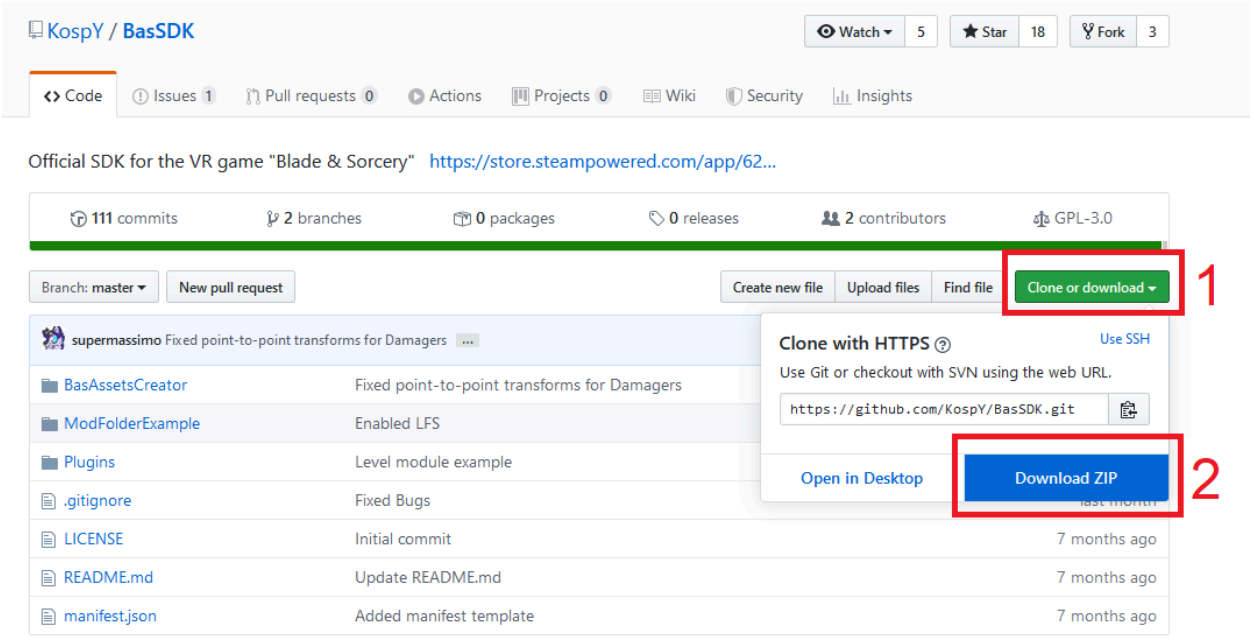
After downloading the .exe file, start the install process and follow the instructions.

While it is installing you can move on to the next step.

1.2 Download & extract the modding SDK

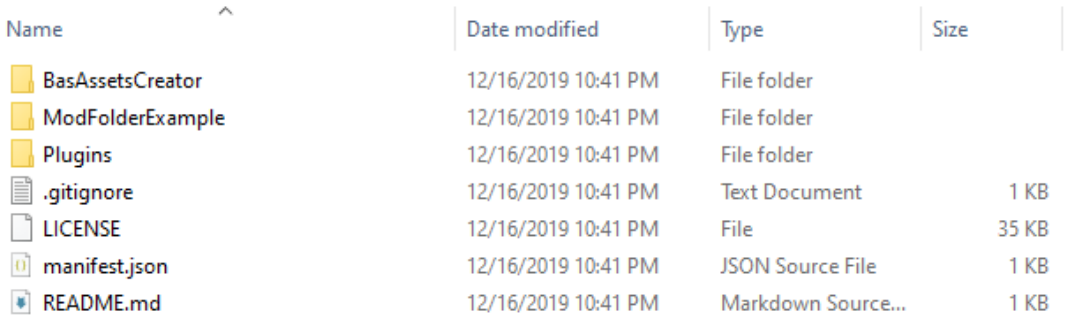
The official modding SDK for B&S can be downloaded here:
https://github.com/KospY/BasSDK/tree/master

On the page, first click “Clone or download” and then “Download ZIP” as seen below:



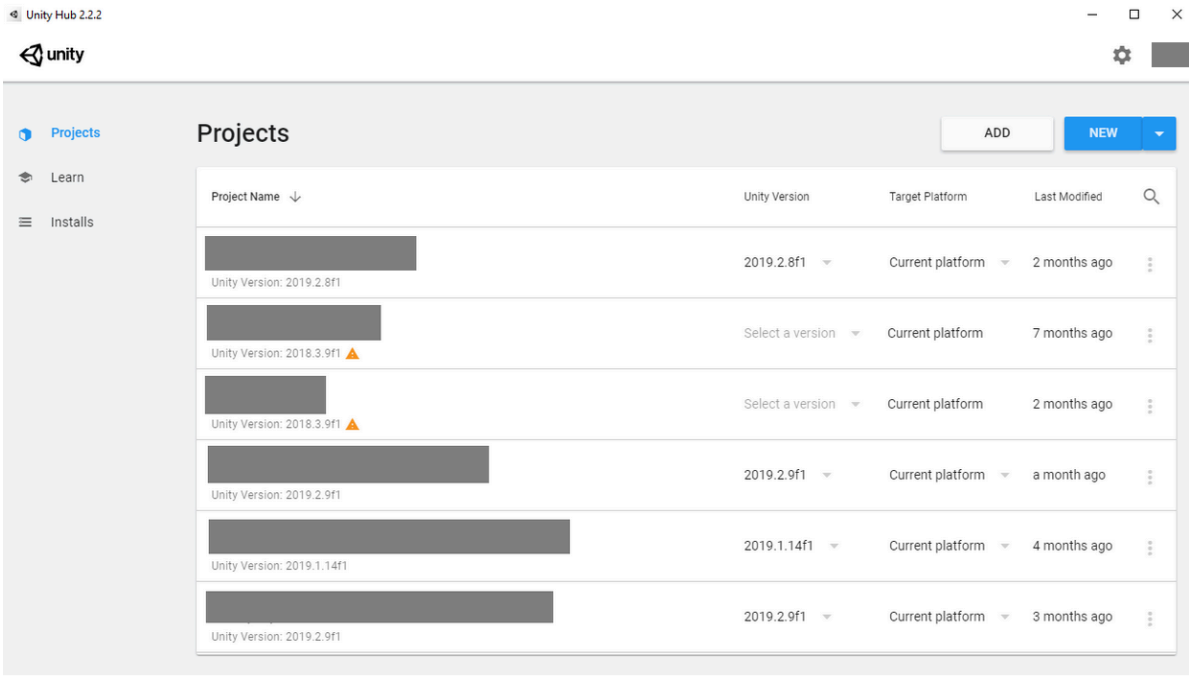
Put the .zip anywhere you like, e.g. on the Desktop and extract it using e.g. WinRar or 7zip.

After extracting the .zip, the contents should look this:



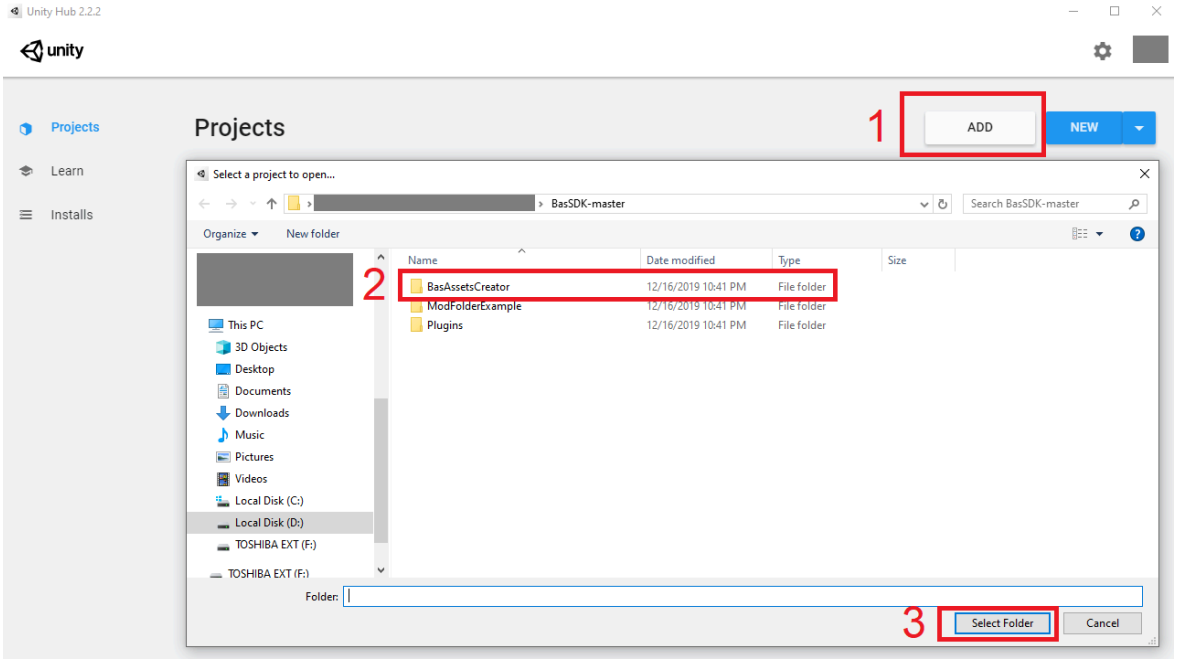
1.3 Open Unity and add the SDK as a project

After Unity has been installed, open up Unity Hub. It should look something like this:

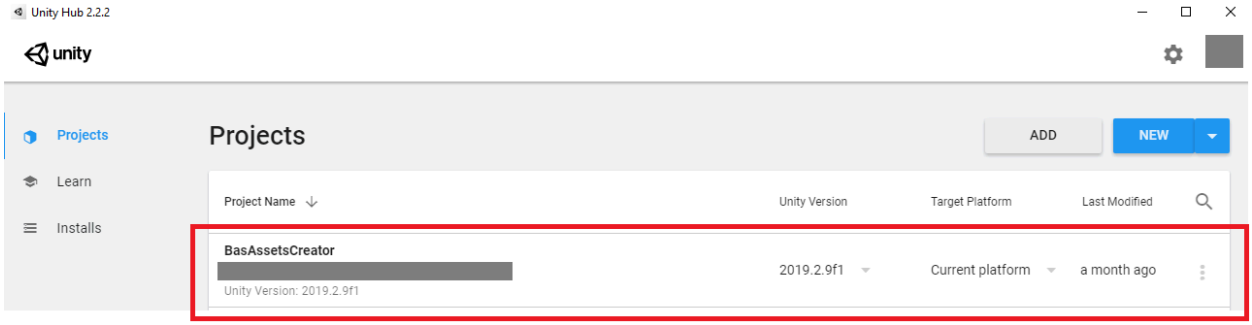


However, the first time that you open Unity Hub the “Projects” section will be empty, but don’t mind that.

Now, press the ADD button. Locate the folder “BasAssetCreator” (**“Bas-SDK” for U9+**) in the extracted folder and click it *once* such that it gets highlighted. Finally, press “Select Folder” as illustrated below:

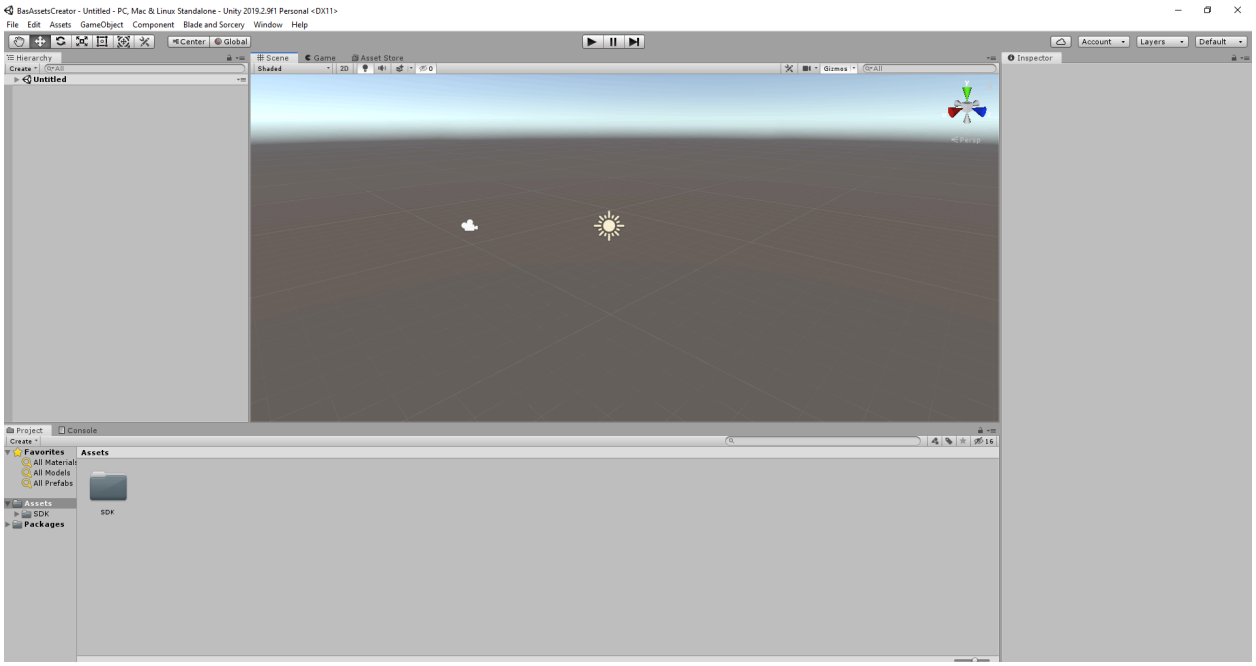


After doing so, you should see a project has gotten added to your “Projects” settings:



Now click on the project to open it (can take a while to load it the first time you open it, but just give it time).

After the project has been fully opened, it should look something like this:



Now you are all set, and we can move on to creating voice pack.

Part 2: Creating the voice pack

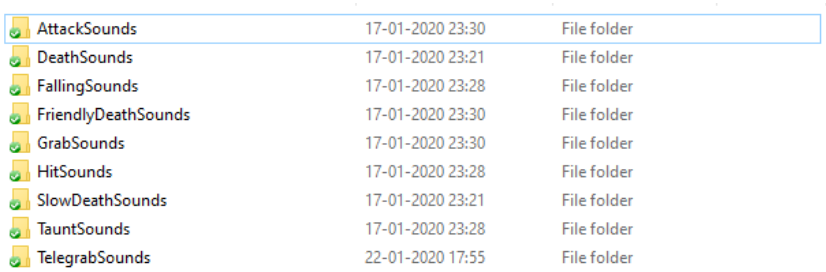
2.1 Audio assets & voice events

First things first, you need some audio clips for your voice pack. The audio clips can be *anything* as long as you have them in the format of either .wav, .mp3, .ogg or .aif.
I do recommend that all of your audio clips are lower than 5 seconds in length and definitely not longer than 10 seconds to avoid potential performance related issues.
Make sure that the voice clips have a decent volume level, as there is a limit for high you can set it in-game. You can use Audacity for boosting the volume of your clips.

To give you an idea of which type of voice lines would be fitting for a voice pack, here are the different “voice events” which is currently available:

- "**attackSounds**": Triggers when an NPC performs a basic attack (the short attack animations).
- "**attackBigSounds**": Triggers when an NPC performs an advanced attack (the longer attack animations)
- "**hitSounds**": Triggers when the NPC takes damage
- "**deathSounds**": Triggers when the NPC dies “normally”
- "**slowDeathSounds**": Triggers when the NPC dies “slowly” (e.g. a stab kill)
- "**fallingSounds**": Triggers when the NPC is falling (e.g. from a ledge)
- "**tauntSounds**": Triggers randomly within a predetermined time interval
- "**friendlyDeathSounds**": Triggers when a friendly NPC dies
- "**grabSounds**": Triggers when you grab an NPC
- "**telegrabSounds**": Triggers when you use telekinesis on an NPC with the gravity spell

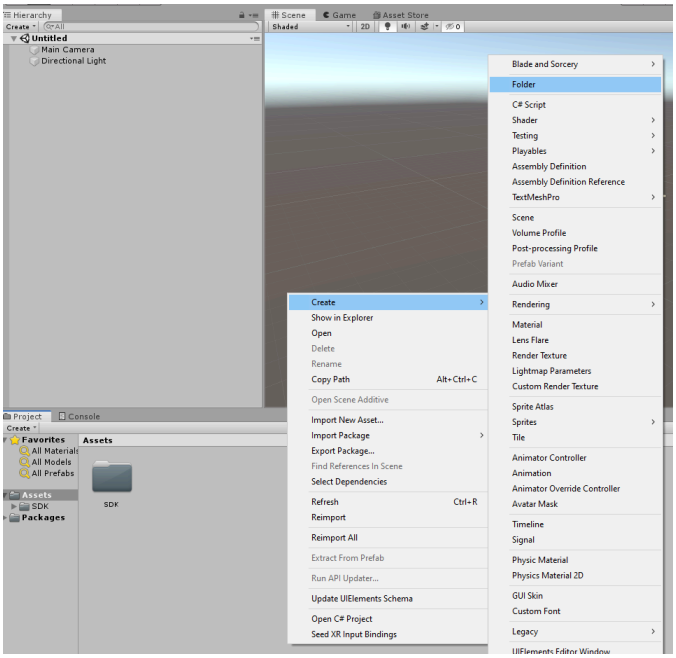
At this point I recommend that you categorize your audio clips into different folders depending on which voice event you plan on having each of your voice lines associated with (see image below).
However, you are not required to have audio clips for each voice event for your voice pack to work. You can also use the same audio clips for different voice events e.g. using “deathSounds” as “slowDeathSounds”.



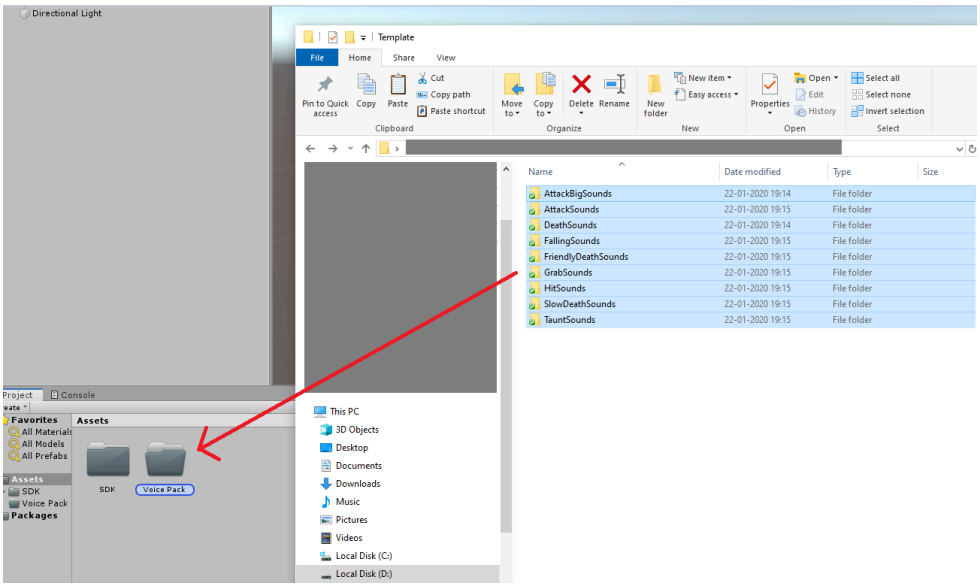
After you have decided which audio clips you want to use and categorized, it’s time to import them in Unity.

2.2 Importing the audio clips in Unity

Before importing audio clips into Unity, start by creating a folder inside the Unity project which will contain your categorized audio clip folders. You can create a folder in Unity by right-clicking in the “Projects” window, followed by clicking “Create” and finally “Folder”:



Select all of your categorized folders containing the audio clips and drag them inside your newly created folder in Unity:

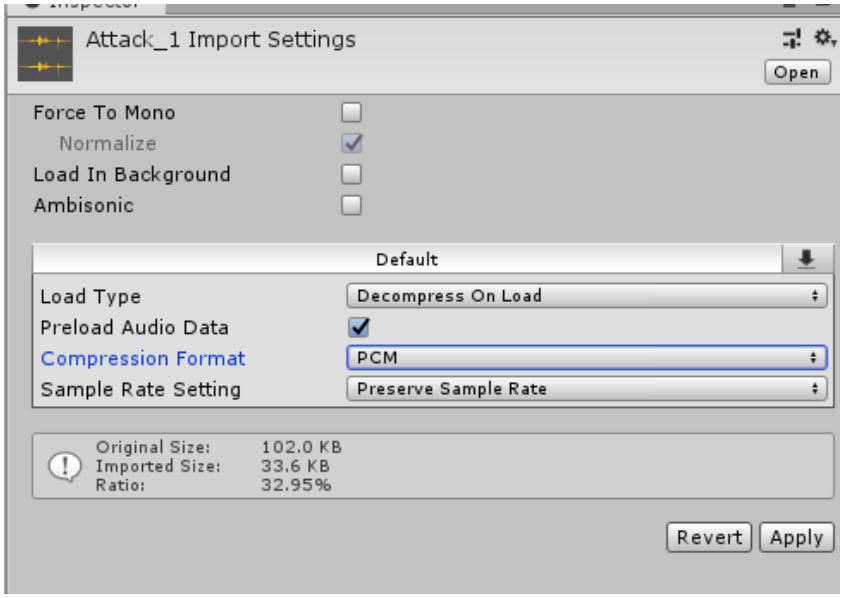


Depending on the size and number of audio clips, it can take a bit for Unity to load them.

2.3 Audio clip import settings

After the audio clips have been imported and loaded into Unity, it is important that you configure the audio clips to use the correct import settings. **Otherwise the audio won't work with the game's NPC speech system**.

For each audio clip, you want to set it up with the following settings:

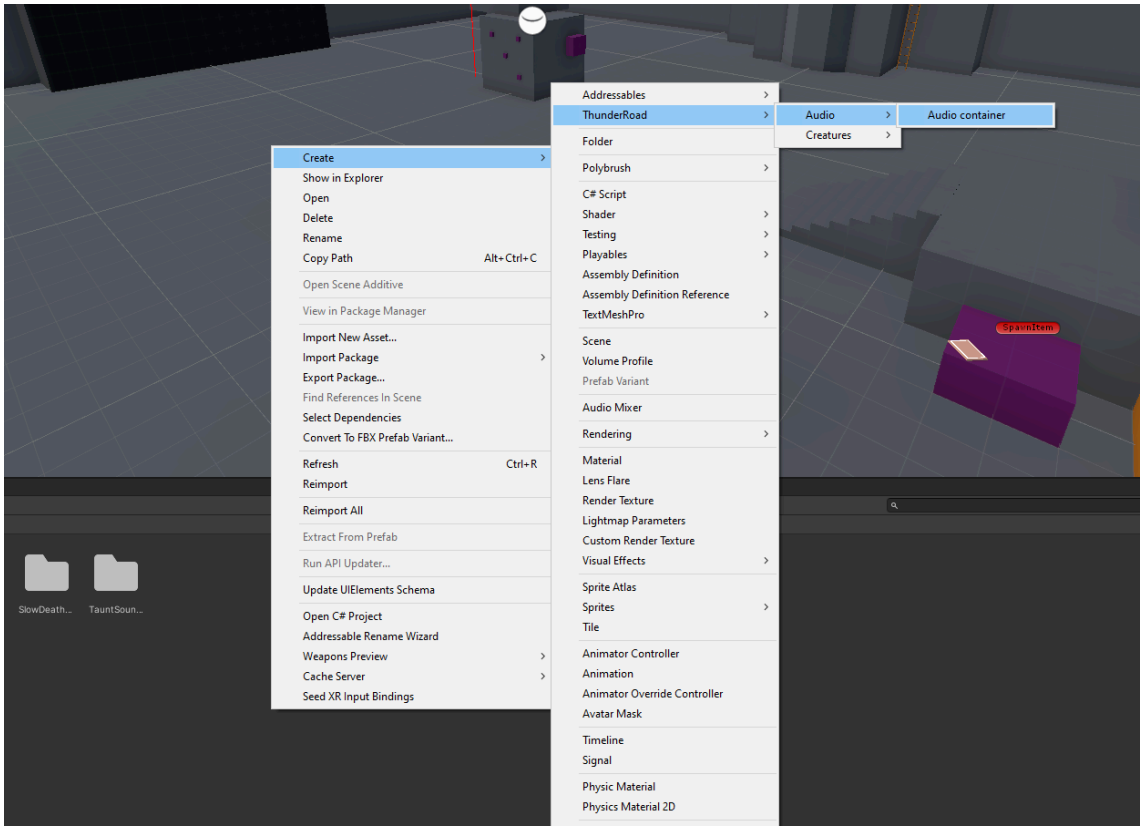


You can select multiple audio clips at the same time, so you don't have to go through each individual clip. **Make sure that you hit Apply after changing the settings**.

2.4 Audio Containers

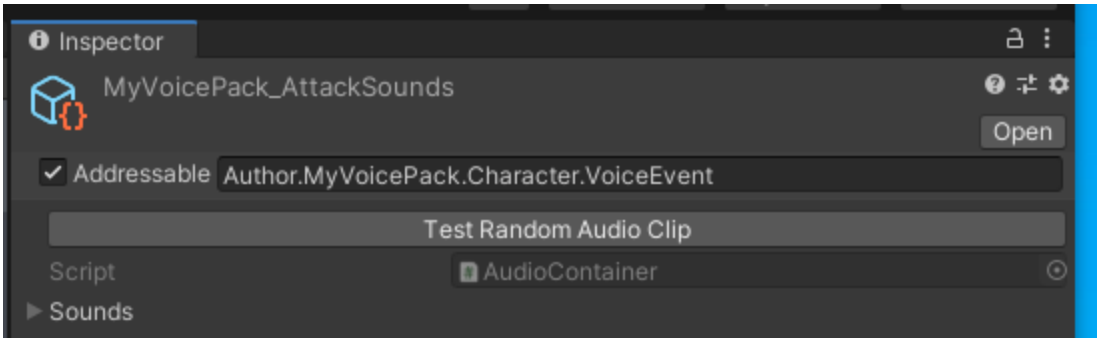
Now we need to set up an AudioContainer for each voice category, which is what we will end up exporting to a new mod folder in the StreamingAssets folder.

To create a new AudioContainer, right-click in the “Projects” window, and then click “Create” -> “ThunderRoad” -> “Audio” -> “AudioContainer”:



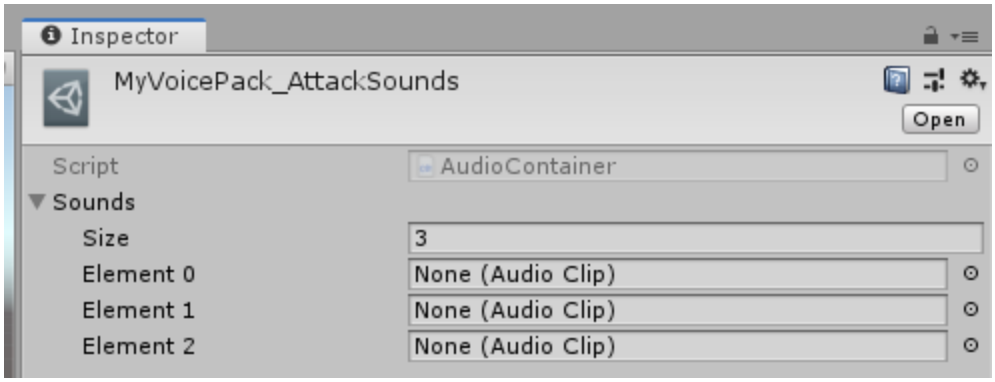
Using the AttackSounds first as an example, name the AudioContainer to something that makes sense, e.g. MyVoicePack_AttackSounds, as we are going to need the name of the AudioContainer later. (MyVoicePack should be the name of your voice pack).

Now select the newly created AudioContainer. First we want to mark the AudioContainer as an Addressable in the Inspector by ticking the following box, followed by typing a unique name:

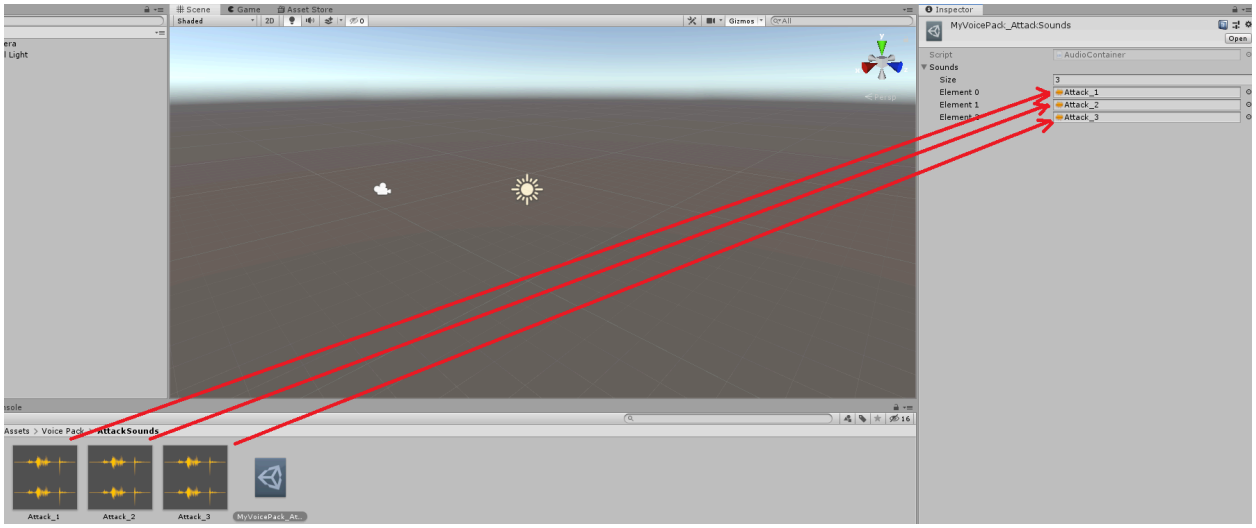


It's important that you name the Addressable something unique. In the above example, the Addressable has been named 'Author.MyVoicePack.Character.VoiceEvent'. You should replace Author with your username, MyVoicePack with the name of your voice pack, Character with the name the character's voice (e.g. Male01 or Darth Vader) and VoiceEvent with the voice event type which represents clips in the AudioContainer (e.g. AttackSounds or DeathSounds).
We are also going to use the name of Addressable again later on.

Now it's time to add your audio clips to the AudioContainer. Using the AttackSounds as an example, I have 3 audio clips that I want to use.
For this example, it means that we want to set the 'Size' of 'Sounds' to 3:



By doing so, we get 3 slots for audio clips which want to fill out with our 3 attack sounds.
We do this by dragging each individual clip into an open slot:



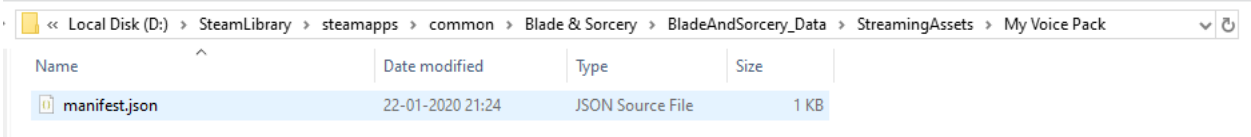
And now we are done with AudioContainer for the AttackSounds!

Repeat the process of creating an AudioContainer for each voice category:
(1) Give it a unique name, (2) Mark as Addressable and assign it a unique name, (3) Assign the appropriate audio clips to the ‘Sounds’ list.

In the end should have up to 10 different AudioContainers each with their own unique Addressable name.

2.5 Build and export the Addressables

Before building and exporting the Addressables, we first need to create a mod folder for the voice pack.
This is done by going to the StreamingAssets folder of the game and creating a new folder with the name of your voicepack. In here, you first want to create an empty text file called “manifest.json” file like this:



In the manifest.json file, you need to type the following (remember to include the curly brackets):

```
{
        "Name": "My voicepack",
        "Description": "Description about my voice pack",
        "Author": "Your Name",
        "ModVersion": "1.0",
        "GameVersion": "U9.1"
}
```

Where the info in **bold** is replaced by your custom info.

For building and exporting the Addressables, refer to this section here of the official SDK wiki: https://github.com/KospY/BasSDK/wiki/The-SDK:-Addressables

After Unity is done building and exporting the Addressables you should see a ".hash", ".json", and ".bundle" file in your mod folder.

The Addressables are now built, exported and complete and now it's time to bind them to a VoicedCreature.

2.6 Setting up a VoicedCreature json

To avoid setting up a VoicedCreature json from scratch, I recommend that you download the NPC Voicepack Overhaul Sample here:
https://www.nexusmods.com/bladeandsorcery/mods/1298

Inside of the mod folder "VoiceFramework_Sample", copy the "VoicedCreature_Test.json" file to your own mod folder and rename "Test" to something unique, e.g. the name of your voice pack or assetbundle.
Make sure that the file still starts with "VoicedCreature_" otherwise the framework will not be able to recognize the file.

Now open the VoicedCreature file using your favorite text or code editor, I personally recommend Notepad++, Sublime Text or Visual Studio Code. You can also just use Notepad if you prefer, but the former will highlight any potential syntax errors you might have in your json file.

If you copied the VoicedCreature_Test file, you should see this inside the file:


```
{
  "id": "VoicedCreature_Test",
  "creatureIDs": [ ],
  "brainIDs": [ ],
  "containerIDs": [ ],
  "maleVoiceSets": [
          {
          "id": "Test",
          "attackSounds": "mpe3us.AudioGroup.Voice.Test.Attack",
          "attackBigSounds": "mpe3us.AudioGroup.Voice.Test.AttackBig",
          "hitSounds": "mpe3us.AudioGroup.Voice.Test.Hit",
          "deathSounds": "mpe3us.AudioGroup.Voice.Test.Death",
          "slowDeathSounds": "mpe3us.AudioGroup.Voice.Test.SlowDeath",
          "fallingSounds": "mpe3us.AudioGroup.Voice.Test.Falling",
          "grabSounds": "mpe3us.AudioGroup.Voice.Test.Grab",
          "telegrabSounds": "mpe3us.AudioGroup.Voice.Test.Grab",
          "friendlyDeathSounds": "mpe3us.AudioGroup.Voice.Test.FriendlyDeath",
          "tauntSounds": "mpe3us.AudioGroup.Voice.Test.Taunt",
          "probabilityWeight": 1.0,
          "volume": 1.0
          }
  ],
  "femaleVoiceSets": [
          {
          "id": "Test",
          "attackSounds": "mpe3us.AudioGroup.Voice.Test.Attack",
          "attackBigSounds": "mpe3us.AudioGroup.Voice.Test.AttackBig",
          "hitSounds": "mpe3us.AudioGroup.Voice.Test.Hit",
          "deathSounds": "mpe3us.AudioGroup.Voice.Test.Death",
          "slowDeathSounds": "mpe3us.AudioGroup.Voice.Test.SlowDeath",
          "fallingSounds": "mpe3us.AudioGroup.Voice.Test.Falling",
          "grabSounds": "mpe3us.AudioGroup.Voice.Test.Grab",
          "telegrabSounds": "mpe3us.AudioGroup.Voice.Test.Grab",
          "friendlyDeathSounds": "mpe3us.AudioGroup.Voice.Test.FriendlyDeath",
          "tauntSounds": "mpe3us.AudioGroup.Voice.Test.Taunt",
          "probabilityWeight": 1.0,
          "volume": 1.0
          }
  ]
}
```


Let’s go through each setting and configure them.

“id”: The unique identifier for your VoicedCreature json file. Rename this to e.g. the name of your json file. **Make sure that it is unique**.

"creatureIDs": A list containing the IDs of the creatures (NPCs) that should have their default voices replaced. For example, if you want to have your voice pack affect all default creatures your creatureIDs should look like this:
"creatureIDs": ["HumanMale", "HumanFemale"]
Leave the list blank if you don’t want to use this feature.

"brainIDs": A list containing the IDs of brains that should have their default voices replaced. This is useful for having different voices depending on wave difficulty. Take a look at the “Brains” folder inside of the “Default” folder for the different IDs. Leave the list blank if you don’t want to use this feature.

"containerIDs": A list containing the IDs of CreatureTables that should have their default voices replaced. This is useful for having different voices depending on the NPC “faction”, e.g. the barbarians in the Arena or the cultists in the Ruins. Take a look at the “CreatureTables” folder inside of the “Default” folder for the different IDs.Leave the list blank if you don’t want to use this feature.

"**maleVoiceSets**" and “**femaleVoiceSets**”: This is where we are going to reference the individual AudioContainers from your AssetBundle. If you voice set is for a male, when you need to set up “maleVoiceSets” otherwise “femaleVoiceSets”.

Using the “attackSounds” as an example, it currently looks like this:
"attackSounds": "mpe3us.AudioGroup.Voice.Test.Attack"

Replace the entirety of ‘mpe3us.AudioGroup.Voice.Test.Attack’ with the name of your Addressable of your AudioContainer containing the attack sounds (*see section 2.4*).
Repeat this process for each voice category.

If you do not plan on adding more voice sets to your voice pack, then make sure to delete the gender counterpart of what you just edited. E.g. if only have populated “maleVoiceSets” then set “femaleVoiceSets” to:
“femaleVoiceSets”: [ ]
I recommend taking a look at the “VoicedCreature_Sample.json” file from the sample voice pack for an advanced setup of multiple voice sets.

"**probabilityWeight**": Multiplier for the chance of this particular voiceset to be picked for an NPC if multiple are available. Usually you just want to leave this at 1.0.

“**Volume**”: Volume for the voice set. 0.7 is the default value for the default in-game voices. Setting the value higher than 1.0 won’t have an effect.

For an alternative approach for setting up the VoicedCreature json, take a look at the JSON Generator For Voice Framework by Jome.

2.7 Testing the voice pack

When you are done setting up the VoicedCreature and ensured that it is formatted correctly, it is finally time to test the voice pack in-game.

Make sure that you have the Custom NPC Voices Framework installed, otherwise your voice pack will not work.
So when you get around to uploading your mod on Nexus, make sure to set the Custom NPC Voices Framework (https://www.nexusmods.com/bladeandsorcery/mods/1296) as a required mod, which will notify users with an automatic pop-up when downloading your voice pack.
If you absolutely insist on making your mod work standalone, you can copy the “VoiceFramework.dll” and “Level_Settings.json” inside of your mod folder. *But be aware* that by doing so, it can break other installed voice packs as they might depend on a newer version of the framework as I put new updates out.

If your voice pack does not seem to be working make sure to look out for any potential errors in the in-game console (press F8 on the keyboard to see it). The console should give you indications of what the problem could be.
Otherwise make sure to have all other voice packs uninstalled when testing yours such that you are guaranteed that only your voice pack have the chance of being activated.

And that’s it! See the next section for tips and tricks.

Tips and Tricks

(1) If multiple voice packs are installed and referencing the same IDs, they simply get added to the pool of possible voices that the given NPC can get. So don't worry about your voice pack conflicting with another.
(2) Remember that you don't need a unique AudioContainer for each voice event/category. E.g. you can use one set of death sounds for both "deathSounds" and "slowDeathSounds" when setting up the VoicedCreature json.
(3) You can control the trigger chance for individual voice clips by adding the same audio clip multiple times to the "Sounds" list in the given AudioContainer.
(4) If you feel like the audio level of your voice clips is too low, even though you already have configured at the volume to the max of 1.0, you need to boost the volume of the audio clip of the audio file itself, e.g. using Audacity (https://www.audacityteam.org/). Make sure you re-import the audio clips in Unity and re-build and re-export the AssetBundle if you make changes to audio clips.

Troubleshooting & common problems

(1) Make sure that you don't have any typos in the AudioContainer references, e.g. "attackSounds": "mpe3us.AudioGroup.Voice.Test.Attack"
(2) Make sure that you are using the correct Unity version. Addressables built in a wrong Unity version can result in the AudioContainers not getting loaded.
(3) Make sure that you specified at least one ID in either creatureIDs, brainIDs or containerIDs. Otherwise, no voices will get replaced.