Supersonic Audio Player

Introduction

I made this audio player because I wanted to have more control when handling audio for my games. The package includes a *prefab* named *AudioPlayer* that acts as a singleton, it has several methods for playing tracks and sound effects. Simply drag and drop this from the folder named *Prefabs* into your scene.

The package also contains a demo scene, and a script for demonstration, look in the folder named *Demo*.

Audio Mixers

Three audio mixers are included in the package: the *Master Mixer*, the *Tracks Mixer*, and the *Sound Effects Mixer*. They can all be located in the folder named *Mixers*.

Master Mixer

This is the master mixer. It has two groups. One is linked to the Tracks Mixer, this group is intended to serve as a master channel for all the tracks. The other is linked to the Sound Effects Mixer which is intended to serve as a master channel for all sound effects.

Tracks Mixer

Create a new group in this mixer for each of your tracks. By doing so, the overall volume for all tracks, and adding of channel effects to be used for all tracks can easily be done on the Tracks group in the Master Mixer.

Sound Effects Mixer

Add a new group in this mixer for every sound effect. By doing so, the overall volume for all sound effects, and adding of channel effects to be used for all sound effects can easily be done on the Sound Effects group in the Master Mixer.

The Audio Player

To use the *Audio Player* simply drag and drop the Audio Player prefab into your scene.

The Audio Player has properties for *Audio Mixer Groups*, which points to the Master group, Tracks group, and Sound Effects group. These are mainly used as fallbacks if no specific Audio Mixer Group is set for a track or sound effect.

There is also a property for which prefab to use when instantiating a sound effect. The default prefab has an *Audio Source*, and a script called *Audio Source Extended* which extends the functionality of the regular Audio Source.

If property *Allow Duplicates Per Frame* is checked the same *Audio Clip* can be instantiated multiple times per frame.

Working with Tracks

Create a Track

Create a variable of the type **`Track`** in a script that inherits from **`MonoBehaviour`**, make sure to include the namespace **`Supersonic`**.

```
using Supersonic;

public class MyClass : MonoBehaviour
{
	public Track MyTrack;
```

Save and have a look at the variable in the *Inspector* in Unity. The following properties are editable in the Inspector for a **`Track`**:

Clip

The Audio Clip to use. Select one from your assets.

Group

The Audio Mixer Group to use, if not set it will use the *Tracks Master Group*. Create a new Audio Mixer Group in the Tracks Mixer and select it from here.

Reverse

Plays the Audio Clip reversed. If this is checked the *Intro* property gets disabled.

Mute

Mutes the Audio Clip.

Volume

Volume for the Audio Clip. Note that this is the volume for the Audio Source when the clip is playing. The volume can also be adjusted in a later stage through the Audio Mixer Group.

Pitch

Pitch for the Audio Clip, a value of 1 equals no pitch.

Intro

If this is checked another property gets visible *Intro End Time* which tells at what point in time, in seconds, the intro ends. When the track ends it will restart from this point. Intro gets disabled if *Reverse* is checked.

Intro End Time

At what point in time, in seconds, the intro ends. When the track ends it will restart from this point. Only visible if *Intro* is checked.

Loop

Enables looping. If *Intro* is checked looping will be enabled by default when the game is running.

Play a Track

Play as Track

For playing a track use the following method and pass the **`Track`** to play as an argument:

```
public void PlayTrack(
      Track track,
      bool destroySoundEffects = false,
      bool muteSoundEffects = false
)
```

By default the parameter **`destroySoundEffects`** is set to false. If it's set to true all sound effects that are currently playing will be destroyed when the track starts.

The other parameter **`muteSoundEffects`** is set to false by default, which means that new sound effects can be played when this track starts playing. If it's set to true new sound effects can't be played until a new track is started.

```
// Call with default values for optional parameters
AudioPlayer.Instance.PlayTrack(MyTrack);
// destroySoundEffects has been set to true
AudioPlayer.Instance.PlayTrack(MyTrack, true);
// muteSoundEffects has been set to true
AudioPlayer.Instance.PlayTrack(MyTrack, true, true);
// Without using directive
Supersonic.AudioPlayer.Instance.PlayTrack(MyTrack);
```

Play as Temporary Track

When playing a track as temporary, the track that is playing right now is stored, and will start to play automatically again when the temporary track ends. This is suitable to use when for example the character recieves a new item.

```
public void PlayTemporaryTrack(
      Track track,
      bool restartCurrentTrackWhenDone = false,
      bool destroySoundEffects = false,
      bool muteSoundEffects = false,
      bool destroySoundEffectsWhenDone = false
)
```

If the optional parameter **`restartCurrentTrackWhenDone`** is set to false, the current track will be paused, and when the temporary track ends the current track will start to play from the same time. If it's set to true, the current track will restart from the beginning when the temporary track ends.

By default the parameter **`destroySoundEffects`** is set to false. If it's set to true all sound effects that are currently playing will be destroyed when the track starts.

The other parameter **`muteSoundEffects`** is set to false by default, which means that new sound effects can be played when this track starts playing. If it's set to true new sound effects can't be played until a new track is started.

The last optional parameter **`destroySoundEffectsWhenDone`**, if set to true it will destroy all the sound effects that are playing when the temporary track ends.

```
// Call with default values for optional parameters
AudioPlayer.Instance.PlayTemporaryTrack(MyTrack);
// restartCurrentTrackWhenDone has been set to true
AudioPlayer.Instance.PlayTemporaryTrack(MyTrack, true);
// Destroy sound effects both when the track starts and ends
AudioPlayer.Instance.PlayTemporaryTrack(CollectJiggyBgm, false, true,
false, true);
```

Play and Stop as Pause Track

If the game has music when it's paused the following methods can be used for handling that. By calling **`PlayPauseTrack`** the current track is stored and will not start to play again until a call to **`StopPauseTrack`** has been made.

```
public void PlayPauseTrack(
      Track track,
      bool restartCurrentTrackWhenDone = false,
```

```
      bool pauseSoundEffects = true
)
```

If the optional parameter **restartCurrentTrackWhenDone** is set to false, the current track will be paused, and when the game is resumed by calling **StopPauseTrack** the current track will start to play from the same time. If it's set to true, the current track will restart from the beginning when the game is resumed.

If **pauseSoundEffects** is set to true all playing sound effects will be paused, and when calling **StopPauseTrack** they will be resumed. If it's set to false, all playing sound effects will be destroyed.

To resume the game a call to **StopPauseTrack** must be made. Note that this method is only relevant if a call to **PlayPauseTrack** has been made earlier.

```
public void StopPauseTrack(
      bool destroyUiSoundEffects = true
)
```

When **destroyUiSoundEffects** is set to true, all sound effects that has started to play when the game was paused will be destroyed. If it's set to false, they will play until they are finished, or destroyed if a call to another method with an instruction to destroy sound effects is made.

```
// Example
public void PauseGame()
{
      // Toggles timeScale to 0 or 1
      Time.timeScale = Convert.ToInt16(!Convert.ToBoolean(Time.timeScale));

      if (Time.timeScale == 0)
      {
            AudioPlayer.Instance.PlayPauseTrack(MyTrack);
      }
      else
      {
            AudioPlayer.Instance.StopPauseTrack();
      }
}
```

Play and Stop as Pause Silently

These methods should be used if the game's pause menu doesn't contain any music. By calling **PlayPauseSilently** the current track is stored and will not start to play again until a call to **StopPauseSilently** has been made.

```
public void PlayPauseSilently(
      bool restartCurrentTrackWhenDone = false,
      bool pauseSoundEffects = true
)
```

If the optional parameter **`restartCurrentTrackWhenDone`** is set to false, the current track will be paused, and when the game is resumed by calling **`StopPauseSilently`** the current track will start to play from the same time. If it's set to true, the current track will restart from the beginning when the game is resumed.

If **`pauseSoundEffects`** is set to true all playing sound effects will be paused, and when calling **`StopPauseSilently`** they will be resumed. If it's set to false, all playing sound effects will be destroyed.

To resume the game a call to **`StopPauseSilently`** must be made. Note that this method is only relevant if a call to **`PlayPauseSilently`** has been made earlier.

```
public void StopPauseSilently(
      bool destroyUiSoundEffects = true
)
```

When **`destroyUiSoundEffects`** is set to true, all sound effects that has started to play when the game was paused will be destroyed. If it's set to false, they will play until they are finished, or destroyed if a call to another method with an instruction to destroy sound effects is made.

```
// Example
public void PauseGameSilently()
{
      // Toggles timeScale to 0 or 1
      Time.timeScale = Convert.ToInt16(!Convert.ToBoolean(Time.timeScale));

      if (Time.timeScale == 0)
      {
            AudioPlayer.Instance.PlayPauseSilently();
      }
      else
      {
            AudioPlayer.Instance.StopPauseSilently();
      }
}
```

Working with Sound Effects

Create a Sound Effect

Create a variable of the type **`SoundEffect`** in a script that inherits from **`MonoBehaviour`**, make sure to include the namespace **`Supersonic`**.

```
using Supersonic;

public class MyClass : MonoBehaviour
{
	public SoundEffect MySoundEffect;
```

Save and have a look at the variable in the *Inspector* in Unity. The following properties are editable in the Inspector for a **`SoundEffect`**:

Clip

The Audio Clip to use. Select one from your assets.

Group

The Audio Mixer Group to use, if not set it will use the *Sound Effects Master Group*. Create a new Audio Mixer Group in the Sound Effects Mixer and select it from here.

Reverse

Plays the Audio Clip reversed.

Mute

Mutes the Audio Clip.

Volume

Volume for the Audio Clip. Note that this is the volume for the Audio Source when the clip is playing. The volume can also be adjusted in a later stage through the Audio Mixer Group.

Pitch

Pitch for the Audio Clip, a value of 1 equals no pitch.

Random Volume

Option for randomizing the volume each time the sound effect is played. When this is checked two new properties gets visible in the Inspector: *Min Volume* and *Max Volume*. When the sound effect is playing, the volume is randomized between these two constants.

Min Volume

The minimum value for randomized volume. Only visible if *Random Volume* is checked.

Max Volume

The maximum value for randomized volume. Only visible if *Random Volume* is checked.

Random Pitch

Option for randomizing the pitch each time the sound effect is played. When this is checked two new properties gets visible in the Inspector: *Min Pitch* and *Max Pitch*. When the sound effect is playing, the pitch is randomized between these two constants.

Min Pitch

The minimum value for randomized pitch. Only visible if *Random Pitch* is checked.

Max Pitch

The maximum value for randomized pitch. Only visible if *Random Pitch* is checked.

Loops

Determines how many times the Audio Clip should be played by default when invoked. Can be overridden when calling **`PlaySoundEffect2D`** or **`PlaySoundEffect3D`**.

Same Volume For Each Loop

If checked the volume will only be randomized once, thus the same volume will be used for each loop. If it's not checked the volume will be randomized for each loop. Only visible if *Random Volume* is checked.

Same Pitch For Each Loop

If checked the pitch will only be randomized once, thus the same pitch will be used for each loop. If it's not checked the pitch will be randomized for each loop. Only visible if *Random Pitch* is checked.

Play a Sound Effect

Play Sound Effect as 2D

For playing a sound effect as a 2D sound, i.e. it will have the same volume wherever it's played in the world, use the following method and pass the **`SoundEffect`** to play as an argument:

```
public void PlaySoundEffect2D(
      SoundEffect soundEffect,
      int loops = 0
)
```

The optional parameter **`loops`** is a way to override the default number of loops, which is set on the property *Loops* in the Inspector. Passing a value of zero to the method indicates that the default number of loops should be used, **not** that it should loop zero times.

```
// Plays a sound effect as 2D
AudioPlayer.Instance.PlaySoundEffect2D(MySoundEffect);
// Ignores the value of Loops from the Inspector and loops 10 times
AudioPlayer.Instance.PlaySoundEffect2D(MySoundEffect, 10);
```

Play Sound Effect as 3D

Playing a sound effect as a 3D sound means that its volume will attenuate over distance, if the *Audio Listener* is too far from the Audio Source the sound won't be audible. This distance can be changed on the Audio Source used by changing the property *Max Distance*. The default Audio Source used is located in the folder named Prefabs.

Use the following method and pass the **`SoundEffect`** to play as an argument:

```
public void PlaySoundEffect3D(
      SoundEffect soundEffect,
      Vector3 position,
      int loops = 0
)
```

The **`Position`** parameter indicates where in the world the sound effect should be played at.

The optional parameter **`loops`** is a way to override the default number of loops, which is set on the property *Loops* in the Inspector. Passing a value of zero to the method indicates that the default number of loops should be used, **not** that it should loop zero times.

```
// Plays a sound effect as 3D at the position of the current object
AudioPlayer.Instance.PlaySoundEffect3D(MySoundEffect, transform.position);
// Ignores the value of Loops from the Inspector and loops 10 times
AudioPlayer.Instance.PlaySoundEffect3D(MySoundEffect, transform.position,
10);
```

Destroy Sound Effects

If a call to this method is made all sound effects currently playing will be destroyed.

```
public void DestroySoundEffects()
```

Track Methods Reference

AudioPlayer.PlayTrack Method

Plays the specified track.

```
public void PlayTrack(
      Track track,
      bool destroySoundEffects = false,
      bool muteSoundEffects = false
)
```

Parameters

Track
      Type: Supersonic.Track
      The track to be played.
destroySoundEffects (Optional)
      Type: System.Boolean
      Destroy all sound effects currently playing.
muteSoundEffects (Optional)
      Type: System.Boolean
      Mute all upcoming sound effects for the duration of this track.

AudioPlayer.PlayTemporaryTrack Method

Plays the specified track and pauses the current track (which is resumed when the specified track has ended.

```
public void PlayTemporaryTrack(
      Track track,
      bool restartCurrentTrackWhenDone = false,
      bool destroySoundEffects = false,
      bool muteSoundEffects = false,
      bool destroySoundEffectsWhenDone = false
)
```

Parameters

Track

Type: Supersonic.Track
The track to be played

restartCurrentTrackWhenDone (Optional)
Type: System.Boolean
If true it restarts the current track when the specified track has ended. If false the current track is paused.

destroySoundEffects (Optional)
Type: System.Boolean
Destroy all sound effects currently playing.

muteSoundEffects (Optional)
Type: System.Boolean
Mute all upcoming sound effects for the duration of this track.

destroySoundEffectsWhenDone (Optional)
Type: System.Boolean
Destroy all sound effects when the temporary track has finished playing.

AudioPlayer.PlayPauseTrack Method

Pauses the current track, use this for a pause menu with music.

```
public void PlayPauseTrack(
      Track track,
      bool restartCurrentTrackWhenDone = false,
      bool pauseSoundEffects = true
)
```

Parameters

Track
Type: Supersonic.Track
The track to be played

restartCurrentTrackWhenDone (Optional)
Type: System.Boolean
If true it restarts the current track when the specified track has ended. If false the current track is paused.

pauseSoundEffects (Optional)
Type: System.Boolean
If true it pauses all sound effects currently playing. If false it destroys all sound effects currently playing.

AudioPlayer.StopPauseTrack Method

Resumes the current track, can only be called if "PlayPauseTrack" has been called earlier.

```
public void StopPauseTrack(
      bool destroyUiSoundEffects = true
)
```

Parameters

destroyUiSoundEffects (Optional)
Type: System.Boolean
Destroys all sound effects that have been initiated while the game was paused.

AudioPlayer.PlayPauseSilently Method

Pauses the current track, use this for a pause menu with NO music.

```
public void PlayPauseSilently(
      bool restartCurrentTrackWhenDone = false,
      bool pauseSoundEffects = true
)
```

Parameters

restartCurrentTrackWhenDone (Optional)
Type: System.Boolean
If true it restarts the current track when the specified track has ended. If false the current track is paused.
pauseSoundEffects (Optional)
Type: System.Boolean
If true it pauses all sound effects currently playing. If false it destroys all sound effects currently playing.

AudioPlayer.StopPauseSilently Method

Resumes the current track, can only be called if "PlayPauseSilently" has been called earlier.

```
public void StopPauseSilently(
      bool destroyUiSoundEffects = true
)
```

Parameters

destroyUiSoundEffects (Optional)
Type: System.Boolean
Destroys all sound effects that have been initiated while the game was paused.

Sound Effect Methods Reference

AudioPlayer.PlaySoundEffect2D Method

Plays the specified sound effect as a 2D sound.

```
public void PlaySoundEffect2D(
      SoundEffect soundEffect,
      int loops = 0
)
```

Parameters

soundEffect
Type: Supersonic.SoundEffect
The sound effect to be played.
loops (Optional)
Type: System.Int32
Overrides the default number of loops if set to greater than 0.

AudioPlayer.PlaySoundEffect3D Method

Plays the specified sound effect as a 3D sound.

```
public void PlaySoundEffect3D(
      SoundEffect soundEffect,
      Vector3 position,
      int loops = 0
)
```

Parameters

soundEffect
Type: Supersonic.SoundEffect
The sound effect to be played.

Position
: Type: **Vector3**
: The position where the sound effect should be played.

loops (Optional)
: Type: System.Int32
: Overrides the default number of loops if set to greater than 0.

AudioPlayer.DestroySoundEffects Method

Destroy all sound effects currently playing.

```
public void DestroySoundEffects()
```