Android Native Audio

Android Native Audio (ANA) is an asset for Unity 4.3.4-5.x on Android. It provides easy access to the native Android audio system for low-latency audio playback.

ANA supports split application binaries (APK + OBB files), and will automatically find and load your sound files in either location.

There is an included demo scene to let you quickly test the difference in latency between Unity and ANA. It is located at `Assets\Android Native Audio\ANA Demo.unity`.

ANA is written in 100% Unity C#. All asset code is located in `Assets\Android Native Audio\Android Native Audio.cs`.
(The `com_android_vending_expansion_zipfile.jar` plugin located in `Assets\Plugins\Android` is the standard Android library for reading OBB files.)

ANA also includes a full set of PlayMaker actions and a demo scene. These are located in `Assets\Android Native Audio\PlayMaker.zip`. Just unzip that file anywhere in your `Assets` folder to use them. For more information on PlayMaker and using actions, see the official website.
http://hutonggames.com/

Note that audio latency can have many contributing factors, including hardware and Android itself. ANA only removes (most of) the latency generated by Unity. This will be different on every device. On some devices the reduction in latency will be large, on others it will be small.

If you have any questions, problems, or feedback, please don't hesitate to drop me a line.
You can contact me at: support@ChristopherCreates.com

Understanding Android Audio

ANA is simple to use, but it works in ways that are different from Unity's audio system. There are three core concepts to know:

Pool: The pool is the central Android resource that manages everything else. (It is a direct implementation of `android.media.SoundPool` in Java.)
Sound: A sound is an individual audio file loaded into memory. A sound is only loaded, played, and unloaded.
Stream: A stream is an audio channel currently playing a sound. All other operations such as pause and resume are performed on streams.

The life cycle of ANA works like this:

1. Make the pool
2. Load a sound
3. Play the sound
4. Optionally modify the stream (pause, volume, stop, etc)
5. Optionally unload the sound
6. Optionally release the pool

Android Audio Files

All audio files for ANA must be located in `Assets/StreamingAssets`. This is a special Unity folder that makes raw files available directly in the APK (or OBB). You can create sub-folders under StreamingAssets to organize your files.

All audio files for ANA must be in an Android-compatible format. See the official documentation for details:
http://developer.android.com/guide/appendix/media-formats.html#core

For best performance, I suggest mono WAV (PCM) files at 16 bits and 44,100 Hz.

Examples

When the scene loads:

```
AndroidNativeAudio.makePool(1);
soundID = AndroidNativeAudio.load("Effect.wav");
```

Later, play the sound:

```
streamID = AndroidNativeAudio.play(soundID);
```

Other operations are very intuitive:

```
AndroidNativeAudio.pause(streamID);
AndroidNativeAudio.resume(streamID);
AndroidNativeAudio.setVolume(streamID, volume);
```

When you're done, you might want to unload a sound or release the whole pool:

```
AndroidNativeAudio.unload(soundID);
AndroidNativeAudio.releasePool();
```

It's that easy!

Tips

- ANA is intended for short, small sound effects. Use Unity's normal audio for longer clips such as background music.
- Make sure your audio files are in the right place and the right format (see above).
- Remember that most Android devices don't have much memory. Be mindful of how many sounds and streams you load at once, and be sure to release the pool when you're done with it.
- Loading a sound file takes some amount of time. Be sure to load them ahead of when you need to play them (ideally when the scene loads).

Scripting Reference

The AndroidNativeAudio class is designed to closely follow the native `android.media.SoundPool`. If you'd like to know more about what's going on behind the scenes, the official documentation is very useful: http://developer.android.com/reference/android/media/SoundPool.html

Debug Logging

At the top of `AndroidNativeAudio.cs` you will find a `DEBUG` variable. Set it to `true` to enable logging of ANA activity. You can monitor Unity logging on Android by running "`<Android SDK Path>\platform-tools\adb logcat -s Unity`".

`public static int` **`load`**`(string` **`audioFile`**`, bool` *`usePersistentDataPath`* `= false)`

> Loads a sound file. Returns the sound ID if successful, -1 if the load fails. Use `makePool` before loading.
>
> `audioFile` - The path to the sound file, relative to Assets\StreamingAssets. (Unless using `usePersistentDataPath`, see below.)
>
> `usePersistentDataPath` - Makes `audioFile` relative to `Application.persistentDataPath`.

`public static void` **`makePool`**`(int` **`maxStreams`**`)`

> Makes an Android native audio pool.
>
> `maxStreams` - The maximum number of streams. (The maximum number of simultaneously playing sounds.)

`public static void` **`pause`**`(int` **`streamID`**`)`

> Pauses a stream.
>
> `streamID` - The ID of the stream to pause.

`public static void` **`pauseAll`**`()`

> Pauses all streams.

`public static int` **`play`**`(int` **`soundID`**`, float` *`leftVolume`* `= 1, float` *`rightVolume`* `= -1, int` *`priority`* `= 1, int` *`loop`* `= 0, float` *`rate`* `= 1)`

Plays a sound. Returns the stream ID if successful, -1 if the play fails. Use `load` before playing.

`soundID` - The ID of the sound to play.

`leftVolume` - The left volume to play at (0.0 - 1.0). If `rightVolume` is omitted, this value will be used for both.

`rightVolume` - The right volume to play at (0.0 - 1.0). Defaults to `leftVolume`.

`priority` - The priority of this stream. If the number of simultaneously playing streams exceeds `maxStreams` in `makePool`, higher priority streams will play and lower priority streams will not.

`loop` - How many times to loop the sound. A value of 0 will play once, -1 will loop until stopped.

`rate` - The rate to play at. A value of 0.5 will play at half speed, 2 will play at double speed.

`public static void` **`releasePool`**`()`

Releases the audio pool resources.

`public static void` **`resume`**`(int` **`streamID`**`)`

Resumes a paused stream.

`streamID` - The ID of the stream to resume.

`public static void` **`resumeAll`**`()`

Resumes all paused streams.

`public static void` **`setLoop`**`(int` **`streamID`**`, int` **`loop`**`)`

Sets the loop of a stream.

`streamID` - The ID of the stream to change.

`loop` - How many times to loop the sound. A value of 0 will play once, -1 will loop until stopped.

`public static void` **`setPriority`**`(int` **`streamID`**`, int` **`priority`**`)`

Sets the priority of a stream.

`streamID` - The ID of the stream to change.

`priority` - The priority of this stream. If the number of simultaneously playing streams exceeds `maxStreams` in `makePool`, higher priority streams will play and lower priority streams will not.

`public static void` **`setRate`**`(int` **`streamID`**`, float` **`rate`**`)`

Sets the rate of a stream.

`streamID` - The ID of the stream to change.

`rate` - The rate to play at. A value of 0.5 will play at half speed, 2 will play at double speed.

`public static void` **`setVolume`**`(int` **`streamID`**`, float` **`leftVolume`**`, float` *`rightVolume`* `= -1)`

Sets the volume of a stream.

`streamID` - The ID of the stream to change.

`leftVolume` - The left volume to play at (0.0 - 1.0). If `rightVolume` is omitted, this value will be used for both.

`rightVolume` - The right volume to play at (0.0 - 1.0). Defaults to `leftVolume`.

`public static void` **`stop`**`(int` **`streamID`**`)`

Stops a stream.

`streamID` - The ID of the stream to stop.

`public static bool` **`unload`**`(int` **`soundID`**`)`

Unloads a sound from the pool. Returns true if unloaded, false if previously unloaded.

`soundID` - The ID of the sound to unload.

If you like Android Native Audio, please tell everyone!

[Leave a review on the Asset Store.]

If you don't like Android Native Audio, please tell me!

Let me know how I can help: support@ChristopherCreates.com

Thank you!

These creations are a labor of love for me. It's very rewarding to know that what I make goes out into the world and helps people. Please feel free to contact me if you need any assistance or have any feedback.

Christopher