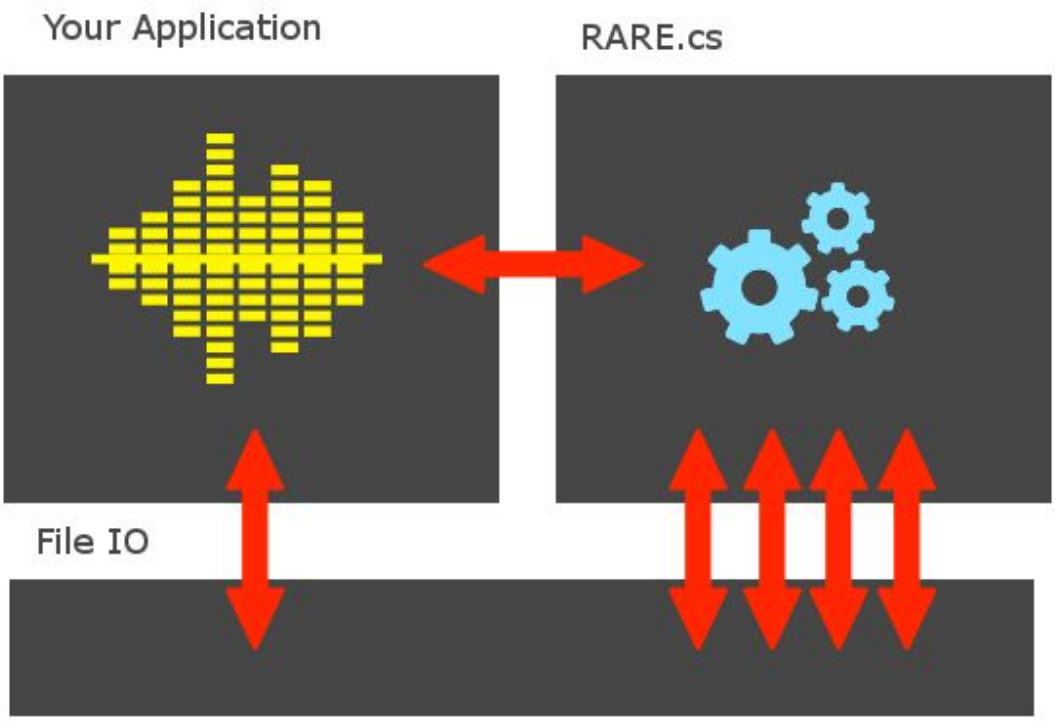
Runtime Audio Record & Export



Video: https://www.youtube.com/watch?v=aIEH0xqbLMI&feature=youtu.be
FAQ at the end of this document

PLEASE READ THIS CAREFULLY ;D

Runtime Audio Record & Export is an easy ready-to-go multiplatform audio recording solution for Unity 4.6 - 5x that can record and export .wav files in IOS, Android, Windows and Mac. It's a simple robust system that allows a user to effortlessly record from the device microphone or the in-game audio listener and export that to a file stored on the device which can be accessed by the user. This saves you the hassle of worrying about fileIO unless you really need to.

Features
- IOS/Android Support
- Music Visualizer
- Waveform Editor
- Reduced latency on low end systems
- Example scenes
- Runtime import of mp3/wav to Audioclip

- Audio Source Wav Export
- Audio Listener Wav Export
- Runtime Mic Recording/ Export
- Audio recording /microphone to audiosource
- Trim AudioClip
- Almost Real-Time Microphone Playback
- Android & iOS sharing

Originally this system was built as a solution for our music creation app. Hopefully this asset provides as much utility as it has provided us! Feel free to contact us if you have any questions or recommendations.

If for any reason this asset does not live up to your expectations please contact us here.
vaytricks@gmail.com

Documentation

Runtime Audio Record & Export was designed to work fluently with the native Unity Audio in order to work on the wide range of platforms unity supports. The default export location is Application.persistentFilePath.

Example Scene Overview

The examples are in the RARE_Example Scene, they work with Unity 5 gui to access the api. Note: the following examples are all in the same scene.

MicRecordExample Allows for easy mic recording. One click starts the mic, and the second click stops the microphone and exports a wav file. The default max record time is 10 minutes but it can be changed. Any extra space on the end of the recording will be trimmed off. So if you have a variety of recordings to make but they are all less than 6 minutes you would set the maximum length to 360 seconds.

AudioListenerRecordExample There is an example for recording what is heard in game through AudioListener.GetOutput data to a Wav file and audio source file. This could be used to play back something that was just heard in game. This is an example for recording and exporting an audio clip from an audiosource component to wav. Click the play button to play an audio clip and adjust its parameters, then use the recording button to capture the audio and export the file as a wav.

AudioClipExportExample Directly write an audio clip to file.

Visualizer All the in-game audio is displayed with 16 frequency bands from bass on the left to treble on the right. It covers the entire range of human hearing. You can use the bandBuffer[] array to get the audio data. Red small rectangles are used to show when a particular frequency band gets too loud.

Waveform Editor The waveform editor is visual way to trim and play audio. The drawing of the waveform is triggered in BasicAudio.cs in the DropdownChanged function. You can use the three sliders on the waveform editor to control left trim, right trim, and playback position. All of these are tied to functions in BasicAudio.cs and RARE.cs.

Note: for these examples to work in your own scene, the RARE.cs script needs to be on the gameObject with the audioListener component (normally the camera), and there cannot be any audiosources on this object.

API Examples

Microphone Recording:

Start Mic Recording
RARE.Instance.StartMicRecording(int maxLengthInSeconds = 300)

Arguments:

- The max predicted length of the recording in seconds. If your prediction is too long the excess silence will be trimmed off.

Stop Mic Recording
RARE.Instance.StopMicRecording(string fileName, Action<AudioClip, string> callBackFunction = null, GameObject popUp = null)

Arguments:

- The recording's name as a string
- An optional callBackFunction such as clipLoaded(AudioClip myClip) that will get the return audioclip when the clip is done being processed
- An optional popUp GameObject to notify the user that the audio is processing (best for long recordings).

Mic (almost) Real-Time Playback
During a mic recording if you want the audio to be routed through the devices speakers turn on the Mic Play Back using this function.

RARE.Instance.SetMicPlayBack(float trueOrFalse);

If you want to change it's volume use:

RARE.Instance.SetMicPlayBackVolume(float volume);

where "volume" is a float between 0.0f and 1.0f.

Audio Listener Recording:

Start AudioListener Recording
RARE.Instance.StartAudioListenerRecording()

Arguments: none

Stop AudioListener Recording
RARE.Instance.StopAudioListenerRecording(string fileName, Action<AudioClip, string> callBackFunction = null, GameObject popUp = null)

Arguments:

- The recording's name as a string
- An optional callBackFunction such as clipLoaded(AudioClip myClip) that will get the return audioclip when the clip is done being processed
- An optional popUp GameObject to notify the user that the audio is processing (best for long recordings).

Export AudioClip to Wav:

ExportClip
RARE.Instance.ExportClip(string fileName, AudioClip myClip, Action<AudioClip, string> callBackFunction = null, GameObject popUp = null)

Arguments:

- The recording's name as a string
- An optional callBackFunction such as clipLoaded(AudioClip myClip) that will get the return audioclip when the clip is done being processed
- The audioclip you want to export
- An optional popUp GameObject to notify the user that the audio is processing (best for long recordings).

Trim AudioClip:

Trim Front of AudioClip (returns AudioClip)
RARE.Instance.TrimFrontOfAudioClip(AudioClip myClip, int frontIndex)

Arguments:
- The AudioClip you want to trim
- The front index where the new clip will start from

Return:
- Returns the new trimmed audioclip

Trim End of AudioClip (returns AudioClip)
RARE.Instance.TrimFrontOfAudioClip(AudioClip myClip, int endIndex)

Arguments:
- The AudioClip you want to trim
- The end index where the new clip will end at

Return:
- Returns the new trimmed audioclip

Split AudioClip (returns AudioClip KeyValuePair)
RARE.Instance.SplitAudioClip(AudioClip myClip, int splitIndex)

Arguments:
- The AudioClip you want to split
- The end index where the clip will be split in half

Return:
- Returns KeyValuePair<AudioClip, AudioClip>

Get AudioClips:

Get AudioClip from File (passes AudioClip to callback function)
RARE.Instance.GetAudioClipFromFile(string fileName, Action<AudioClip, string> callBackFunction, string fileType = “wav”)

Arguments:
- Filename
- Callback function that the clip and filename will be passed to when it is done loading
- File type, accepted: wav, mp3, ogg

Return:
- void

Get AudioClip from URL (passes AudioClip to callback function)
RARE.Instance.GetAudioClipFromURL(string urlPath, Action<AudioClip, string> callBackFunction, string fileType = “wav”)

Arguments:
- urlPath where your audio is stored online
- Callback function that the clip and urlpath will be passed to when it is done loading
- File type, accepted: wav, mp3, ogg

Return:
- void

Managing Recordings:

Record Volume
Whether you are recording with the audio listener, or the microphone an audioclip you can adjust it’s output volume before the process. You can call this with a slider or just normally. It only accepts floats from 0.0f to 1.0f. 1.0f being max volume

RARE.Instance.OutputVolume(float input);

File Management
Each recording needs to have a unique filename or else it will be overwritten. It is recommended to use some sort of naming convention to keep track of your files like we

used in BasicAudio.cs. You can use playerprefs or file searching to keep track of older files. If you have any questions about this feel free to contact us.

Anroid & iOS Sharing:

Share on android and iOS by using this function:
NativeShare.Share(string body, string filePath = null, string mimeType = "audio/wav", string chooserText = "Select sharing app")

Example:
NativeShare.Share ("Body Message", Application.persistentDataPath +"/"+ currentAsrc.clip.name + ".wav");

MimeType and chooserText can be changed for Android if you want, but this is optional. To share other types of audio you can use “audio/*” or something like “audio/mp3” for mimetype. The chooserText variable is the text that shows when the native sharing dialogue opens.

Device Compilation:

We have tested RARE with PC, Mac, Android, and iOS. It should work out of the box with no extra setup in player settings / Xcode etc… If you have questions let us know.

Frequently Asked Questions:

Can I save recordings to external memory?

You can save to internal memory for all devices. We also have a native share feature for emailing or saving recordings into applications like google drive. In order to get external sharing to an SD card directly you will have to change the file path in RARE.cs to that device's specific SD Card path since each android device tends to have a different path for that.

How do I make sure i’m not overwriting my files if they have the same name?

You can use the CheckFileName function in BasicAudio.cs to make sure you are not overwriting a file. Here is an example of how you can use the API to do recording and exporting. CheckFileName will manage your file names by adding a number to the end of duplicate files.

```
public void ListenerStartStop(){
	if (isRecording) {
		//whether you use cliploaded or not is up to you
		RARE.Instance.StopAudioListenerRecording (CheckFileName("Name of Recording"), ClipLoaded);
		isRecording = false;
	} else {
		isRecording = true;
		RARE.Instance.StartAudioListenerRecording ();
	}
}
```

Where do the recordings get saved to on my device?

Here is where your recordings are being saved.
https://docs.unity3d.com/ScriptReference/Application-persistentDataPath.html

What I would recommend is putting this line in your start function and printing it to a text component on the screen. This will show you where it is being saved.

YourTextComponent.text = Application.persistentDataPath;

The file path is different for every device but on android it is usually something like:
Internal Storage > Android > data > com.YourCompany.YourAppName > files

RARE comes with an easy sharing function you might want to try out so you can easily share the recordings via email, google drive, etc...

Who is Vaytricks?

https://www.vaytricks.com/
http://andrewnakas.com/
https://nescroft.com/

If you have any problems you can contact us at vaytricks@gmail.com and we will get right back to you. If you find this asset useful please leave us a review on the unity asset store!