Contents

Ariel Plugin for Unitz - Documentation

Figure 1: ariel thumbnail

Table of contents

Setup

Prerequisites Installation

Quickstart

Editor pre-generation Runtime generation

Features

Audio effects Tags

API Reference

Ariel Text-To-Speech (Remote)   Ariel Text-To-Speech (Local)   Ariel Glossary   Ariel Save Wav   Ariel Common Types

Others

If you have any question, do not hesitate to contact us through our Discord server or by mail at contact@xandimmersion.com

Speakers

> The available speakers depend on the API Key provided to you. The following list is the default list of speakers available for the plugin. If you have a specific request, please contact us.
>
> We are providing customized speaker training for your project. If you are interested in a specific speaker or language, please get in touch with us.

Local speakers are available for offline use. Online speakers require an internet connection.

Name	Type	Gender	Language(s)
Bryan	*Online*	male	English, Spanish, French, German, Italian, Portuguese, Polish, Turkish, Russian, Dutch, Czech, classic Arabic, Chinese, Japanese, Korean
Bryan	*Local*	male	English
Abrogail	*Online*	female	English, Spanish, French, German, Italian, Portuguese, Polish, Turkish, Russian, Dutch, Czech, classic Arabic, Chinese, Japanese, Korean
Abrogail	*Local*	female	English

API Reference

← Table of contents

On this page

Introduction

The Ariel plugin provides a set of methods to generate audio speech from text sentences. The plugin can be used in the Unity Editor and in runtime projects. The plugin is divided into two parts: the **Remote** and the **Local** version. The **Remote** version uses the Ariel API to generate the audio, while the **Local** version uses a local server to generate the audio. The **Local** version is faster and can be used offline, but the audio quality is lower than the **Remote** version.

The plugin is divided into four main classes:

- **Ariel Voice Remote**: The remote version of the Ariel plugin. It uses the Ariel API to generate the audio.
- **Ariel Voice Local**: The local version of the Ariel plugin. It uses a local server to generate the audio.
- **Glossary**: A class to manage glossaries. A glossary is a list of sentences and their corresponding audio files.
- **Ariel Save Wav**: A class to save audio files to disk.

The plugin also provides a set of common types used by the other classes.

Back to Top

Ariel Voice Remote

Ariel Remote Class

The Ariel Remote class is used to generate audio speech from text sentences using the Ariel API. The class provides methods to get available speakers, generate audio from text, save audio to disk, and get the next available filename.

This is a static class, because we are not storing any data in the class itself. We are just using it to call the methods. You don't need to create an instance of this class to use it.

Get available Speakers (Remote)

C# Function (public | async): `ArielRemote.GetSpeakers` Namespace: `ArielVoiceRemote`

Description Gets the available speakers from the Ariel API. Using your personal Api Key, you can get the list of available speakers and languages. These may vary depending on your subscription plan.

Parameters

Name	Type	Description
apiKey	string	The API key used to authenticate the request.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
speakers	Task<SpeakerSettings>	An object containing all available speakers and languages. For more information, see Speaker Settings.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `apiKey` is null or empty.
`UnityWebRequestException`	Thrown when there is an error with the web request.
`JsonException`	Thrown when there is an error parsing the JSON response.

Back to Top

Text To Audio (Remote | Editor Version)

C# Function (public | async): `ArielRemote.TextToAudio` Namespace: `ArielVoiceRemote`

Description Generates audio from text using the Ariel API. The audio is generated using the specified speaker, text, and other settings. The audio is saved to the specified path.

Parameters

Name	Type	Description
option	string	The speaker name.
phrase	string	The text to convert to audio.
title	string	The name of the audio file.
octave	float	The pitch of the audio.
speed	float	The speed of the audio.
effect	string	The audio effect to apply.
monostereo	string	The mono/stereo setting.
volume	float	The volume of the audio.
highSampleRate	string	The high sample rate setting.
voiceImprovement	string	The voice improvement setting.
savePath	string	The path to save the audio file.
apiKey	string	The API key required to access the Ariel API.
position	int	The position of the text in the list of generation elements.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
result	Task<bool>	Indicates whether the operation was successful.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `apiKey`, `phrase`, or `option` is null or empty.
`UnityWebRequestException`	Thrown when there is an error with the web request.
`AudioClipException`	Thrown when there is an error creating the audio clip.

Back to Top

Text To Audio (Remote | Runtime Version)

C# Function (public | async): `ArielRemote.TextToAudioRuntime` Namespace: `ArielVoiceRemote`

Description Generates audio from text using the Ariel API. The audio is generated using the specified speaker, text, and other settings. It returns an Audio Clip that you can use in your Runtime project.

Parameters

Name	Type	Description
option	string	The speaker name.
phrase	string	The text to convert to audio.
octave	float	The pitch of the audio.
speed	float	The speed of the audio.
effect	string	The audio effect to apply.
monostereo	string	The mono/stereo setting.
volume	float	The volume of the audio.
highSampleRate	string	The high sample rate setting.
voiceImprovement	string	The voice improvement setting.
apiKey	string	The API key required to access the Ariel API.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
audioClip	Task<AudioClip>	The generated audio clip.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `apiKey`, `phrase`, or `option` is null or empty.
`UnityWebRequestException`	Thrown when there is an error with the web request.
`AudioClipException`	Thrown when there is an error creating the audio clip.

Back to Top

Save Audio Clip

C# Function (public): `ArielRemote.SaveAudioClip` Namespace: `ArielVoiceRemote`

Description Saves an audio clip to disk.

If being used in Editor mode, it is recommended to use the Ariel Text To Audio function, because it already saves the Audio Clip directly.

If being used in Runtime, you will need to save the asset to the Resources folder, but it is recommended to use the Audio Clip directly as a variable to bind it to an audio source.

Parameters

Name	Type	Description
audioClip	AudioClip	The audio clip to save.
title	string	The title of the audio file.
savePath	string	The path to save the audio file.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
result	bool	Indicates whether the audio clip was saved successfully.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `audioClip` or `savePath` is null or empty.
`IOException`	Thrown when there is an error saving the audio file.

Back to Top

Get Next Filename

C# Function (private): `ArielRemote.getNextFileName` Namespace: `ArielVoiceRemote`

Description Generates the next available file name. The file name is generated by appending a number to the initial file name, if it already exists.

Parameters

Name	Type	Description
fileName	string	The initial file name.

Return values

Name	Type	Description
nextFileName	string	The next available file name.

Exceptions

Name	Description
`IOException`	Thrown when there is an error generating the next file name.

Back to Top

Ariel Voice Local

Ariel Local Class

The Ariel Local class is used to generate audio speech from text sentences using a local server. The class provides methods to start and stop the local server, get available speakers, generate audio from text, save audio to disk, and get the next available filename.

This is not a static class, because we are storing data (such as the server process) in the class itself. You need to create an instance of this class to use it.

Class Attributes (Local)

Name	Type	Description
url	string	The base URL of the local server.
processName	string	The name of the local server process.
port	int	The port number used by the local server.
process	Process	The process object for the local server.

Back to Top

Start Local Server

C# Function (public): `ArielLocal.StartServer` Namespace: `ArielVoiceLocal`

Description Starts the local server. The server must be running to generate audio from text.

Parameters

Name	Type	Description
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
result	bool	Indicates whether the server started successfully.

Exceptions

Name	Description
`ProcessException`	Thrown when there is an error starting the server process.

Back to Top

Shutdown Local Server

C# Function (public): `ArielLocal.ShutdownServer` Namespace: `ArielVoiceLocal`

Description Stops the local server. The server should be stopped when it is no longer needed.

> Make sure to stop the server when exiting the game, otherwise it will keep running as a background task and needs to be closed manually in the task manager.

Parameters

Name	Type	Description
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
result	bool	Indicates whether the server was stopped successfully.

Exceptions

Name	Description
`ProcessException`	Thrown when there is an error stopping the server process.

Back to Top

Restart Server

C# Function (public): `ArielLocal.RestartServer` Namespace: `ArielVoiceLocal`

Description Restarts the local server. The server must be running to generate audio from text.

Return values

Name	Type	Description
result	bool	Indicates whether the server was restarted successfully.

Exceptions

Name	Description
`ProcessException`	Thrown when there is an error restarting the server process.

Back to Top

Force Restart Server

C# Function (public): `ArielLocal.ForceRestartServer` Namespace: `ArielVoiceLocal`

Description Restarts the local server.

Return values

Name	Type	Description
result	bool	Indicates whether the server was restarted successfully.

Exceptions

Name	Description
`ProcessException`	Thrown when there is an error restarting the server process.

Back to Top

Is Server Running

C# Function (public): `ArielLocal.IsServerRunning` Namespace: `ArielVoiceLocal`

Description Checks if the local server is running already or not.

Return values

Name	Type	Description
result	bool	Indicates whether the server is running.

Back to Top

Kill Process On Port

C# Function (private): `ArielLocal.KillProcessOnPort` Namespace: `ArielVoiceLocal`

Description Kills the process running on the specified port.

Parameters

Name	Type	Description
killPort	int	The port number to check for an active process.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
result	bool	Indicates whether the process was successfully killed.

Exceptions

Name	Description
`ProcessException`	Thrown when there is an error killing the process.

Back to Top

Check Service Running On Port

C# Function (private): `ArielLocal.CheckServiceRunningOnPort` Namespace: `ArielVoiceLocal`

Description Checks if a service is running on the specified port.

Parameters

Name	Type	Description
checkPort	int	The port number to check for a running service.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
state	ServerState	An object indicating the status of the port and server process.

Exceptions

Name	Description
`ProcessException`	Thrown when there is an error checking the server status.

Back to Top

Get available Speakers (Local)

C# Function (public): `ArielLocal.GetSpeakers` Namespace: `ArielVoiceLocal`

Description Gets the available speakers that are available for usage with the local server. It returns all local `.onnx` models that also have a corresponding `{speaker_name}.onnx.json` file stored in the `Assets\ArielVoiceGeneration\Local\models` folder. If you want to have access to more voices, get in touch with X&Immersion.

Parameters

Name	Type	Description
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
speakers	SpeakerSettings	An object containing all available speakers and languages. For more information, see Speaker Settings.

Back to Top

Text To Audio (Local | Editor Version)

C# Function (public | async): `ArielLocal.TextToAudio` Namespace: `ArielVoiceLocal`

Description Generates audio from text using the local server. The audio is generated using the specified speaker, text, and other settings. The audio is saved to the specified path.

Parameters

Name	Type	Description
option	string	The voice option to use.
phrase	string	The text to convert to audio.
title	string	The title of the audio file.
octave	float	The pitch of the audio.
speed	float	The speed of the audio.
effect	string	The audio effect to apply.
monostereo	string	The mono/stereo setting.
volume	float	The volume of the audio.
highSampleRate	string	The high sample rate setting.
voiceImprovement	string	The voice improvement setting.
savePath	string	The path to save the audio file.
apiKey	string	The API key required to access the Ariel API.
position	int	The position of the text in the list.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
result	Task<bool>	Indicates whether the operation was successful.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `apiKey`, `phrase`, or `option` is null or empty.
`UnityWebRequestException`	Thrown when there is an error with the web request.
`AudioClipException`	Thrown when there is an error creating the audio clip.

Back to Top

Text To Audio (Local | Runtime Version)

C# Function (public | async): `ArielLocal.TextToAudioRuntime` Namespace: `ArielVoiceLocal`

Description Generates audio from text using the local server. The audio is generated using the specified speaker, text, and other settings. It returns an Audio Clip that you can use in your Runtime project.

Parameters

Name	Type	Description
option	string	The voice option to use.
phrase	string	The text to convert to audio.
octave	float	The pitch of the audio.
speed	float	The speed of the audio.
effect	string	The audio effect to apply.
monostereo	string	The mono/stereo setting.
volume	float	The volume of the audio.
highSampleRate	string	The high sample rate setting.
voiceImprovement	string	The voice improvement setting.
apiKey	string	The API key required to access the Ariel API.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
audioClip	Task<AudioClip>	The generated audio clip.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `apiKey`, `phrase`, or `option` is null or empty.
`UnityWebRequestException`	Thrown when there is an error with the web request.
`AudioClipException`	Thrown when there is an error creating the audio clip.

Back to Top

Save Audio Clip (Local)

C# Function (public): `ArielLocal.SaveAudioClip` Namespace: `ArielVoiceLocal`

Description Saves an audio clip to disk.

If being used in Editor mode, it is recommended to use the Ariel Text To Audio function, because it already saves the Audio Clip directly.

If being used in Runtime, you will need to save the asset to the Resources folder, but it is recommended to use the Audio Clip directly as a variable to bind it to an audio source.

Parameters

Name	Type	Description
audioClip	AudioClip	The audio clip to save.
title	string	The title of the audio file.
savePath	string	The path to save the audio file.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
result	bool	Indicates whether the audio clip was saved successfully.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `audioClip` or `savePath` is null or empty.
`IOException`	Thrown when there is an error saving the audio file.

Back to Top

Get Next Filename (Local)

C# Function (private): `ArielLocal.getNextFileName` Namespace: `ArielVoiceLocal`

Description Generates the next available file name. The file name is generated by appending a number to the initial file name, if it already exists.

Parameters

Name	Type	Description
fileName	string	The initial file name.

Return values

Name	Type	Description
nextFileName	string	The next available file name.

Exceptions

Name	Description
`IOException`	Thrown when there is an error generating the next file name.

Back to Top

Glossary

Glossary Class

The Glossary class is used to manage glossaries. A glossary is used to change the way a speaker pronounces a specific word. The class provides methods to write, read, and edit glossary files, as well as to compare a sentence to a glossary.

Glossary Class Attributes

Name	Type	Description	Default Value
sentence	string	The sentence to generate.	””
newGlossaryName	string	The name of the new glossary.	””
glossary	TextAsset	The glossary to read or edit.	null
wordList	WordList	The list of words and their pronunciations.	new WordList()

Words Class

The Words class represents a word and its pronunciation in the glossary.

Class Attributes

Name	Type	Description	Default Value
word	string	The word in the glossary.	””
pronunciation	string	The pronunciation of the word.	””

WordList Class

The WordList class represents a list of words and their pronunciations in the glossary.

Class Attributes

Name	Type	Description	Default Value
words	List	The list of words and their pronunciations.	new List()

Write CSV

C# Function (public): `ArielGlossary.WriteCSV`

Description Generates a new glossary or overwrites an existing one. The glossary is saved as a CSV file.

Parameters None

Return values None

Exceptions None

Read CSV

C# Function (private): `ArielGlossary.ReadCSV`

Description Reads the current glossary and populates the word list.

Parameters None

Return values None

Exceptions None

Edit CSV

C# Function (public): `ArielGlossary.EditCSV`

Description Edits the current glossary and saves the changes to the CSV file.

Parameters None

Return values None

Exceptions None

Compare Sentence To Glossary

C# Function (private): `ArielGlossary.CompareSentenceToGlossary`

Description Compares a sentence to the glossary and replaces words in the sentence with their pronunciations from the glossary.

Parameters None

Return values None

Exceptions None

Back to Top

Ariel Save Wav

Ariel Save Wav Class

C# Class: `ArielSavWav` Namespace: `ArielVoiceGenSav`

Description

The Ariel Save Wav class is used to save audio bytes to a WAV file.

Save bytes to file

C# Function (public): `ArielSavWav.Save`

> If you are using this function in a Runtime project, you will need to save the asset to the Resources folder, but it is recommended to use the Audio Clip directly as a variable to bin it to an audio source.

Description Saves an AudioClip as a WAV file.

Parameters

Name	Type	Description
filename	string	The name of the file to save.
clip	AudioClip	The AudioClip to save.
logs	bool	Indicate if logs should be printed to the console.

Return values

Name	Type	Description
result	bool	Indicates whether the file was saved successfully.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `filename` or `clip` is null or empty.
`IOException`	Thrown when there is an error saving the file.

Trim Silence

C# Function (public): `ArielSaveWav.TrimSilence`

Description Trims silence from the beginning and end of an AudioClip.

Parameters

Name	Type	Description
clip	AudioClip	The AudioClip to trim.
min	float	The minimum amplitude to consider as non-silence.

Return values

Name	Type	Description
result	AudioClip	A new AudioClip with the silence trimmed.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `clip` is null.

Trim Silence (List of samples)

C# Function (public): `ArielSaveWav.TrimSilence`

Description Trims silence from the beginning and end of a list of samples.

Parameters

Name	Type	Description
samples	List	The list of samples to trim.
min	float	The minimum amplitude to consider as non-silence.

Name	Type	Description
channels	int	The number of channels in the audio.
hz	int	The sample rate of the audio.
_3D	bool	Indicates whether the audio should be 3D.
stream	bool	Indicates whether the audio should be streamed.

Return values

Name	Type	Description
result	AudioClip	A new AudioClip with the silence trimmed.

Exceptions

Name	Description
`ArgumentNullException`	Thrown when the `samples` list is null.

Back to Top

Ariel Common Types

Ariel TTS Class (Editor Version)

C# Class: `ArielTts` Namespace: `ArielCommonTypes`

Description

The Ariel TTS class is used for the Editor Version of the Ariel plugin. It contains all required information for each Audio line to generate. You can use it in Runtime projects, to keep track of your generation settings.

Class Attributes

Name	Type	Description	Default Value
Phrase	string	The text to convert to audio.	””
Octave	float	The pitch of the audio.	0.0f
Speed	float	The speed of the audio.	1.0f
Title	string	The title of the audio file.	””
Selected_O	int	The selected option.	0
Selected_L	int	The selected language.	9
glossaryToUsePath	string	The path to the glossary to use.	””

Name	Type	Description	Default Value
useGlossary	bool	Indicates whether to use the glossary.	false
Effect	int	The audio effect to apply.	0
MonoStereo	string	The mono/stereo setting.	””
isStereo	bool	Indicates whether the audio is stereo.	false
Volume	float	The volume of the audio.	1.0f
HighSampleRate	string	The high sample rate setting.	””
isHSRate	bool	Indicates whether to use high sample rate.	false
AdvancedOptions	bool	Indicates whether to use advanced options.	false
VoiceImprovement	string	The voice improvement setting.	””
useVoiceImprovement	bool	Indicates whether to use voice improvement.	false

Back to Top

Ariel TTS Class (local)

C# Class: `ArielLocalApiCall` Namespace: `ArielCommonTypes`

Description

The Ariel Local API Call class is used for the Local Version of the Ariel plugin. It is an internal class that is used within the ArielVoiceLocal class. It contains all information that is passed to the local server to generate the audio.

Class Attributes

Name	Type	Description	Default Value
modelPath	string	The path to the model.	””
sentence	string	The text to convert to audio.	””
outputType	string	The output type.	””
format	string	The format of the audio.	””
semitones	int	The number of semitones to shift.	0
speed	float	The speed of the audio.	1.0f
volume	float	The volume of the audio.	1.0f
voice_improvement	bool	Indicates whether to use voice improvement.	false
high_framerate	bool	Indicates whether to use high frame rate.	false
telephone	bool	Indicates whether to apply telephone effect.	false
cave	bool	Indicates whether to apply cave effect.	false
smallcave	bool	Indicates whether to apply small cave effect.	false
gasmask	bool	Indicates whether to apply gas mask effect.	false
badreception	bool	Indicates whether to apply bad reception effect.	false
nextroom	bool	Indicates whether to apply next room effect.	false
alien	bool	Indicates whether to apply alien effect.	false
alien2	bool	Indicates whether to apply alien2 effect.	false

Name	Type	Description	Default Value
stereo	bool	Indicates whether the audio is stereo.	false

Back to Top

Ariel Csv

C# Class: `ArielCsv` Namespace: `ArielCommonTypes`

Description

The Ariel Csv class is used to manage the glossary csv files. The class keeps track of the glossary file path and the list of glossary items.

Class Attributes

Name	Type	Description	Default Value
Word	string	The word in the CSV file.	””
Pronunciation	string	The pronunciation of the word.	””

Back to Top

Speaker

C# Class: `Speaker` Namespace: `ArielCommonTypes`

The Speaker class represents a speaker. The class provides properties for the speaker’s name, language, and other settings. This class is marked as `Serializable`.

Class Attributes

Name	Type	Description	Default Value
name	string	The name of the speaker.	””
id	int	The ID of the speaker.	0
emotion	List	The list of emotions the speaker can express.	new List()
language	List	The list of languages the speaker can speak.	new List()
gender	string	The gender of the speaker.	””

Back to Top

Speaker Object

The Speaker Object class represents the return type of the GetSpeakers method. The class provides an array of speakers.

Class Attributes

Name	Type	Description	Default Value
speakers	Speaker[]	The array of speakers.	new Speaker[0]

Back to Top

Language

C# Class: `Language` Namespace: `ArielCommonTypes`

Description

The Language class represents a language. The class provides properties for the language's name and unified code representation for each language. This class is marked as `Serializable`.

Class Attributes

Name	Type	Description	Default Value
name	string	The name of the language.	””
code	string	The code of the language.	””

Back to Top

Language Codes

C# Class: `LanguageCodes` Namespace: `ArielCommonTypes`

The Language Codes class provides a list of language codes.

Class Attributes

Name	Type	Description	Default Value
languages	List	The list of languages.	new List()

Language Codes

Language	Code
Chinese	zh-cn
Korean	ko
Dutch	nl
Turkish	tr
Swedish	sv
Indonesian	id
Filipino	fil
Japanese	ja
Ukrainian	uk
Greek	el
Czech	cs
Finnish	fi
Romanian	ro
Russian	ru
Danish	da
Bulgarian	bg
Malay	ms
Slovak	sk
Croatian	hr
Arabic	ar
Tamil	ta
English	en
Polish	pl
German	de
Spanish	es
French	fr
Italian	it
Hindi	hi
Portuguese	pt

Back to Top

Speaker Settings

C# Class: `SpeakerSettings` Namespace: `ArielCommonTypes`

Description

The Speaker Settings class represents the return value of the GetSpeakers (Remote) and GetSpeakers (Local) method. The class provides an array of speakers and languages.

Class Attributes

Name	Type	Description	Default Value
speakers	Speaker[]	The array of speakers.	new Speaker[0]
languages	Language[]	The array of languages.	new Language[32]

Back to Top

Server State

C# Class: `ServerState` Namespace: `ArielCommonTypes`

Description

The Server State class represents the state of the server. The class provides properties for the server's status and other information.

Class Attributes

Name	Type	Description	Default Value
portInUse	bool	Indicates whether the port is in use.	true
arielServerProcessRunning	bool	Indicates whether the Ariel server process is running.	false

Back to Top

Features

← Table of contents

On this page

Audio effects Tags

Audio effects

When generating audio speech with the Ariel plugin, you can apply audio effects to the generated audio. Those effects are optional and can be combined to create the desired sound atmosphere. Here is a list of all audio effects available:

Name	Description
Telephone	The voice sounds like it's coming from a phone
Cave	The voice sounds like the speaker is in a cave
Small cave	The voice sounds like the speaker is in a small cave
Gas mask	The voice sounds like the speaker has a gas mask
Bad reception	The voice sounds like it's coming from a phone with a bad reception
Next room	The voice sounds like the speaker is in the next room
Alien	An alien audio effect is added to the voice
Alien 2 (alt)	An other alien audio effect (like in the space) is added to the voice
Stereo	The audio file have two channels (the mono channel is duplicated)

> [!TIP] the **Alien** effect is automatically applied to the voices *Xalith*, as well as the **Alien2** effect is automatically applied to the voices *Zephyr* and *Yorgon*. They cannot be removed.

Tags

There are two types of tags that can be used:

- Pause Tags
- Emotion Tags

Tags availability: The tag system is currently only supported on the Remote version of Ariel.

Pause Tags

In your sentence, you can enter a silence tag for a custom pause. Write `<pause Xs>` or `<pause Xms>` where ***X*** is the duration (in seconds or milliseconds). For example:

```
Hi, how are you? <pause 3s> My name is Jane.
```

Emotion Tags

Emotion tags can be applied to modify the tone of the generated speech. You can use tags like `<emotion happy>`, `<emotion sad>`, etc. For example:

```
This is amazing! <emotion happy> I'm thrilled!
```

The emotion tags are active for the generated text until the next emotion tag in the current text. If no emotion tag is found, the default emotion is neutral. For example:

```
I'm feeling great today!<emotion happy> Let’s celebrate! <emotion
angry>I'm so mad! Go away.
```

In this example the first sentence will be generated with a neutral emotion, the second sentence with an happy emotion, and the last two sentences with an angry emotion.

Emotion Availability: Not every speaker has emotions available. If the speaker does not support emotions, the emotion tags will be ignored and the default emotion is used. To check which speakers support emotions, please refer to the X&Immersion Create APP.

Tag Examples

To help you understand how each tag works, here are a few examples: - For pauses: `This is a test.<pause 2s> Please wait.` - For emotions: `I'm feeling great today!<emotion happy> Let’s celebrate!`

# Others

← **Table of contents**

On this page

Plugin project settings

Some parameters, such as the API Endpoint URL and the API key used to authenticate the request, are required by the API. You can get your personal API Key by contacting us at.

Package a project

The Ariel plugin works on packaged projects for Windows. Other operating systems may work as well, but we do not officially support them.

No runtime generated sentences

If your project does **not** use runtime generated sentences, meaning all speech are pre-generated using *Ariel Text-To-Speech (Remote)* or *Ariel Text-To-Speech (Local)* are never used outside the editor, then you can simply remove the Ariel plugin before packaging the project.

> [!CAUTION] If you try to package a project that uses Ariel Editor elements at runtime, the build package will fail.

Quickstart

← Table of contents

> [!IMPORTANT] Please ensure that the Ariel plugin is installed and enabled within your project. Please read this section for more details on how to install the Ariel plugin.

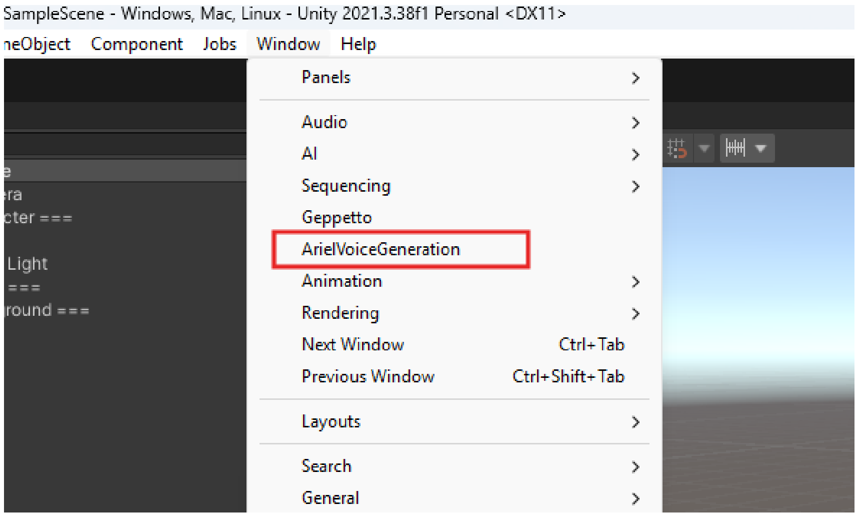
On this page

Editor Pre-generation   Runtime generation   Glossary

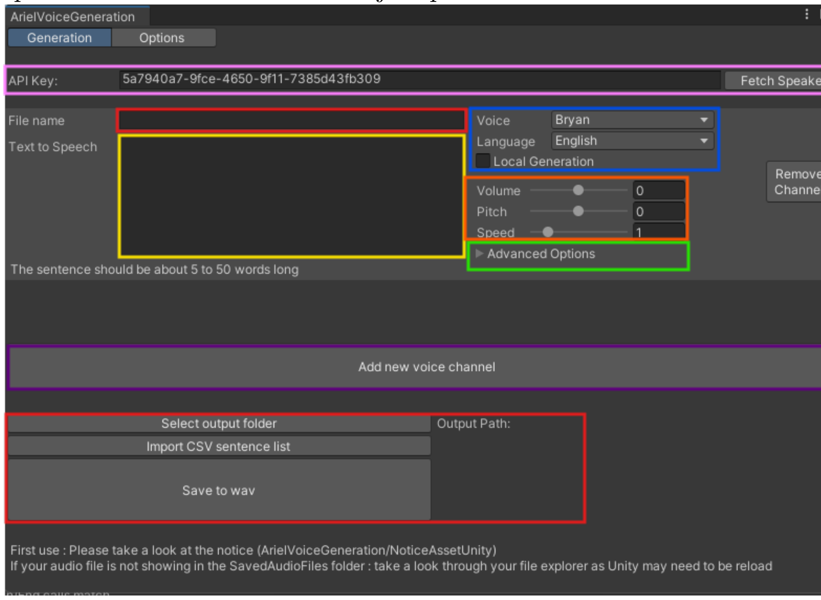
Editor Pre-generation

The editor interface allows you to generate audio sentences and download them as a file. The audio files can then be imported to your Project.

1. On the Editor main window, go to **Window** > **ArielVoiceGeneration**:



2. The Ariel Editor Window should open. Enter the sentence and adjust parameters according to your needs:



API Key: 5a7940a7-9fce-4650-9f11-7385d43fb309

Voice: Bryan
Language: English
Volume 0
Pitch 0
Speed 1

The sentence should be about 5 to 50 words long

First use : Please take a look at the notice (ArielVoiceGeneration/NoticeAssetUnity)
If your audio file is not showing in the SavedAudioFiles folder : take a look through your file explorer as Unity may need to be reload

- On the 🟪 Pink box, enter the API Key provided to you. The API Key is required to generate the audio file. If you do not have an API Key, please contact us at *contact@xandimmersion.com*. If no speakers are available, you need to press on *Fetch Speakers* to get all local and remote speakers.
- On the 🟨 Yellow box, enter the sentence that will be generated by the API. The sentence supports tags, like `<pause 1s>`.
- On the 🟦 Blue box, select the speaker and the language that will be used to generate the sentence. You can use the fields *Language* Dropdown to

see which languages exist for the currently selected speaker. By checking the *Local Generation* Checkbox you are switching between the local and the remote TTS generation. The speakers are displayed accordingly. See speakers list.

- On the 🟧 Orange box, adjust the pitch (in semitones), the audio speed and the volume (in dB) that will be applied to the audio file before saving it. You can also use your favorite game audio software tool to change those settings more dynamically **after** the file was saved.
- On the 🟩 Green box, you can add audio effects that will be applied to the audio file. Those audio effects are optional. You can also select Stereo Audio, High Sample Rate (44kHz) and a voice Improvement algorithm which will enhance the generated audio. You can also select if you want to use a previously generated Glossary.
- On the 🟪 Purple box, you can add new voice channels to add another voice generation setting. You can add and remove as many channels as you want. All the channels will be generated.
- On the 🟥 Red box, Select the audio file destination and filename. We recommend to always use the *WAV* format, as MP3 is not natively supported with Unreal Engine yet.

Runtime generation

Info

To create a game that supports Text-To-Speech inside a game, you will need to use the API provided by the plugin. The API is available to anyone that has a subscription with us. Read the API Reference thoroughly to understand how to use the API.

It is recommended to use the API in a C# script to generate the audio file.

Our Editor can help you to generate the audio file and test the different settings before using the API in your game. It used the same API as the plugin.

> Be sure to not use the Editor Functions in a packaged game. The Editor functions are only available in the Editor.

Runtime Demo

To learn how to use the basic functionality of our tool, we created an easy to use Demo Script that you can use to generate audio files. It is located in the `ArielVoiceGeneration/Demo` folder. The script is named `ArielRuntimeDemo.cs`. Please ensure to review the script for detailed instructions on how to implement the audio generation process.

The following will describe a step by step guide on how to use the Demo Script:

1. Create a new Unity project

   We recommend using Unity Version 2021.3.38f1 for compatibility with the Ariel plugin. Other versions may work, but we have not tested all of them.

2. Add the `ArielVoiceGeneration` Plugin to your project following the installation instructions in Setup. Make sure the Network Compatibility is set as described.

3. Create a new empty scene (or use an existing one) in your project using `File` > `New Scene`.

4. Add an Event System to your Scene by right-clicking in the Hierarchy and selecting `UI` > `Event System`.

5. Add an Audio Component to your Event System by selecting the Event System in the Hierarchy and clicking `Add Component` in the Inspector. Search for `Audio Source` and add it.

6. Add the `ArielRuntimeDemo.cs` script to your project. You can find it in the `ArielVoiceGeneration/Demo` folder. Drag it onto your Event System in the Inspector to add it.

7. Add the speaker name and your Api Key into the public boxes (as shown in the screenshot below).

   For checking the available speakers, you can use the `Fetch Speakers` button in the Ariel Editor or on our Create APP.

8. Now run your game. A grey text box will show up. Enter the text you want to generate and press the `Enter` key. The audio file will be generated and played back.

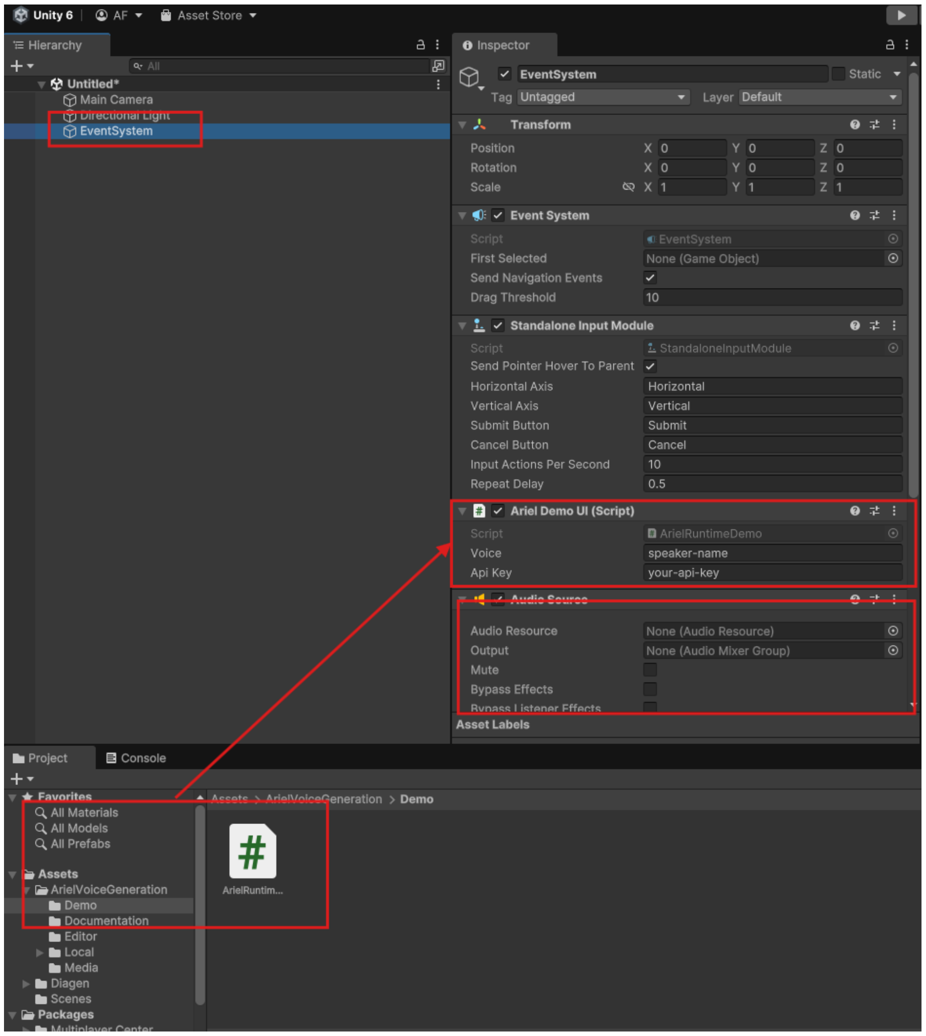
   Check out the Script to see how you can use the API in your game. Further functionality can be added to the script to enhance the user experience. This information can be found in the API Reference.

Glossary

The glossary is a list of words that are pronounced differently than they are written. The glossary is used to improve the pronunciation of the generated audio. The glossary is a JSON file that can be generated by the plugin. The glossary is optional and can be used in the editor interface.

Using the Glossary you can Add or Remove words that are not pronounced correctly by the TTS engine. The Glossary is a JSON file that can be generated by the plugin.

You can save your glossary to a local file and load it in the editor interface.



EventSystem Static
Tag Untagged Layer Default

Transform

Position	X	0	Y	0	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

Event System

Script	EventSystem
First Selected	None (Game Object)
Send Navigation Events	✓
Drag Threshold	10

Standalone Input Module

Script	StandaloneInputModule
Send Pointer Hover To Parent	✓
Horizontal Axis	Horizontal
Vertical Axis	Vertical
Submit Button	Submit
Cancel Button	Cancel
Input Actions Per Second	10
Repeat Delay	0.5

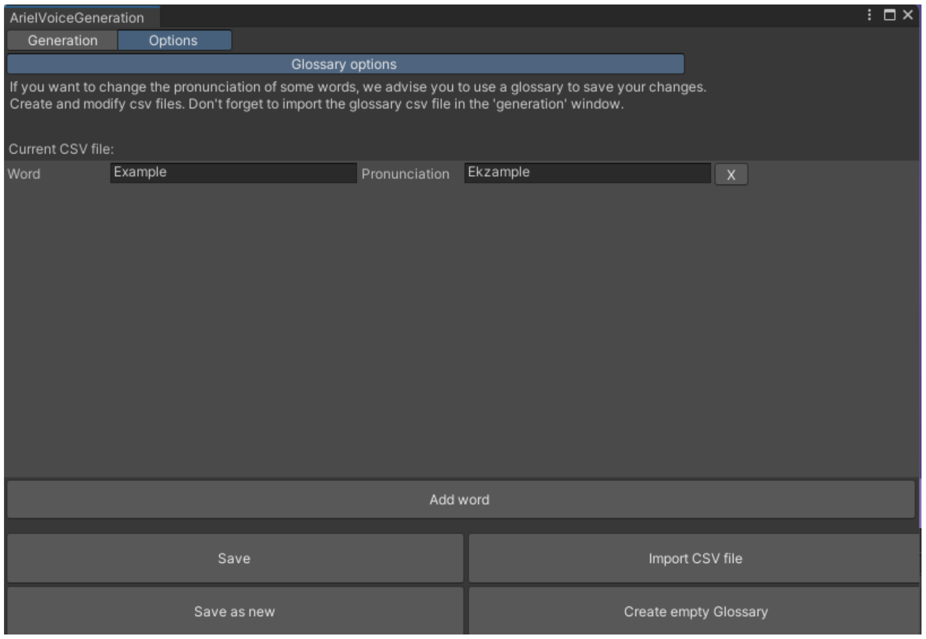
Ariel Demo UI (Script)

Script	ArielRuntimeDemo
Voice	speaker-name
Api Key	your-api-key

Audio Source

Audio Resource	None (Audio Resource)
Output	None (Audio Mixer Group)
Mute	
Bypass Effects	
Bypass Listener Effects	

Figure 2: Demo Setup


ArielVoiceGeneration

Generation | Options

Glossary options

If you want to change the pronunciation of some words, we advise you to use a glossary to save your changes.
Create and modify csv files. Don't forget to import the glossary csv file in the 'generation' window.

Current CSV file:

Word: Example — Pronunciation: Ekzample — X

Add word

Save | Import CSV file

Save as new | Create empty Glossary

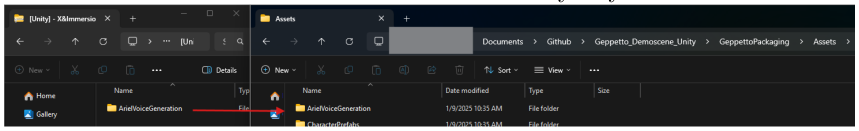
# Setup

← Table of contents

> The Ariel plugin is compatible was developed in **Unity Version 2021.3.38f1**. It is recommended to use this version to avoid compatibility issues.

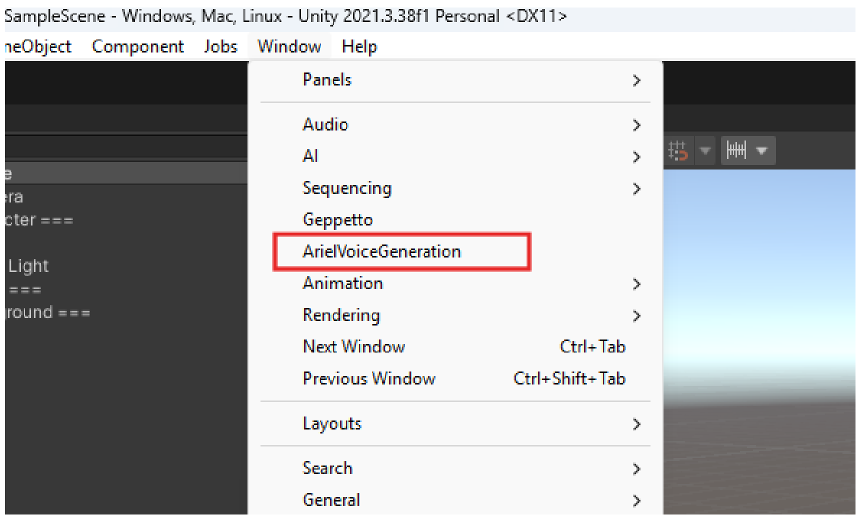
Installation

From a ZIP archive (source code)

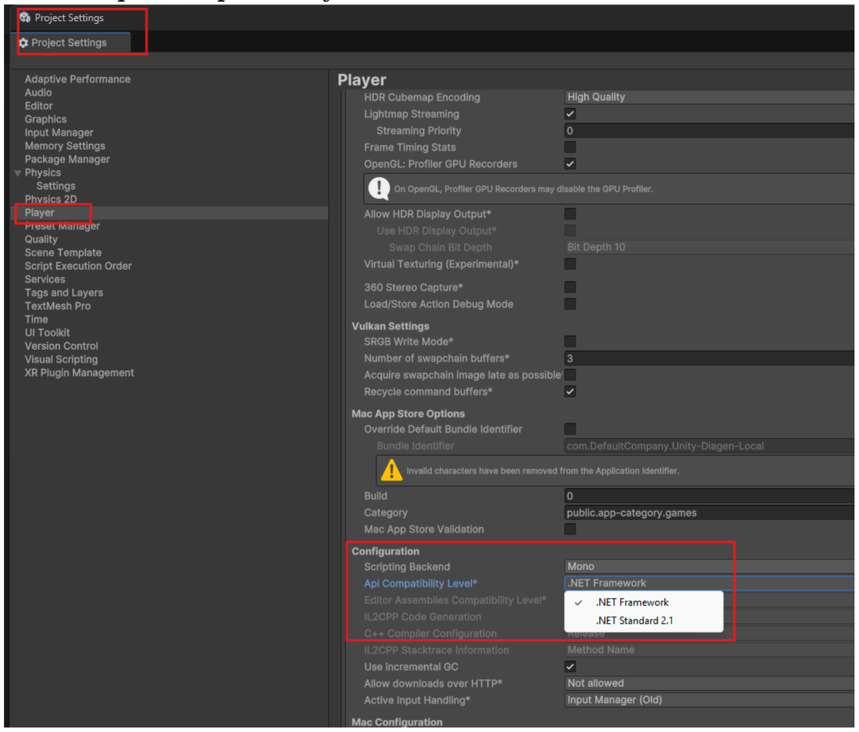
0. If not already done, create a new project with the Engine version of your choice and navigate to the project's root directory.
1. Extract the content from the ZIP archive to the root directory of your Unreal project: 

2. Reopen your project.
3. Navigate to **Window** and make sure ArielVoiceGeneration is listed in the

dropdown menu.



4. The API used requires the .Net 4.0 API compatibility level. To set this up, navigate to **Edit** > **Project Settings** > **Player** > **Other Settings** and set the **Api Compatibility Level** to **.Net Framework** or **.NET 4.x**.


Project Settings

Project Settings

Adaptive Performance
Audio
Editor
Graphics
Input Manager
Memory Settings
Package Manager
Physics
Settings
Physics 2D
Player
Preset Manager
Quality
Scene Template
Script Execution Order
Services
Tags and Layers
TextMesh Pro
Time
UI Toolkit
Version Control
Visual Scripting
XR Plugin Management

Player

HDR Cubemap Encoding	High Quality
Lightmap Streaming	✓
Streaming Priority	0
Frame Timing Stats	
OpenGL: Profiler GPU Recorders	✓
On OpenGL, Profiler GPU Recorders may disable the GPU Profiler.	
Allow HDR Display Output*	
Use HDR Display Output*	
Swap Chain Bit Depth	Bit Depth 10
Virtual Texturing (Experimental)*	
360 Stereo Capture*	
Load/Store Action Debug Mode	
Vulkan Settings	
SRGB Write Mode*	
Number of swapchain buffers*	3
Acquire swapchain image late as possible*	
Recycle command buffers*	✓
Mac App Store Options	
Override Default Bundle Identifier	
Bundle Identifier	com.DefaultCompany.Unity-Diagen-Local
Invalid characters have been removed from the Application Identifier.	
Build	0
Category	public.app-category.games
Mac App Store Validation	
Configuration	
Scripting Backend	Mono
Api Compatibility Level*	.NET Framework
Editor Assemblies Compatibility Level*	✓ .NET Framework / .NET Standard 2.1
IL2CPP Code Generation	
C++ Compiler Configuration	Release
IL2CPP Stacktrace Information	Method Name
Use incremental GC	✓
Allow downloads over HTTP*	Not allowed
Active Input Handling*	Input Manager (Old)
Mac Configuration	