RT-Voice PRO

Hearing is understanding

Documentation

crosstales LLC

Date: 26. March 2020

Version: 2020.1.2

Table of Contents

Thank you for buying our asset "RT-Voice PRO"!

If you have any questions about this asset, send us an email at rtvoice@crosstales.com.
Please don't forget to rate it or write a little review – it would be very much appreciated.

1. Overview

Did you ever want to make a game for people with **visual impairments** or **reading difficulties**? Or want your players to **not have to read too much**? Or would you listen to just the dialogues in your game **without consulting a voice-actor** in early stages of development? Then RT-Voice is your **time-saving** solution to do so!
RT-Voice uses TTS-voices already integrated in your system to pronounce any written text at runtime.

All of this happens without intermediate steps: the transformation effects **instantaneously** - and, if needed, **simultaneously**!

2. Features

2.1. Convert text to voice

- **Instant conversion** from text to speech - generated **during runtime**!
- **Side effect:** the continuous audio generation **saves a lot of memory**!
- **No need for voice actors** during the testing phase of your game!
- Filter voices by **name**, **culture** and/or **gender**
- **Several voices at once** are possible (e.g. for scenes in a public place, where many people are talking at the same time)
- **Fine tuning** for your voices with **speed, pitch** and **volume**!
- Support for **SSML** and **EmotionML**!
- **Current word**, **visemes** and **phomenes** on Windows and iOS - including **marker functions**!
- Generated audio **can be stored** in files reusable within Unity!
- **1-infinite synchronized speakers** for a single AudioSource!
- Simple **sequence** and **dialogue system**
- **No** performance drops!

2.2. Documentation & control

- **Test** voices within the editor!
- Powerful **API** for **maximum control!**
- Detailed **demo scenes**!
- Comprehensive **documentation** and **support**!
- Full **C# source code**

2.3. Compatibility

- Supports **all build platforms!**
- **Native providers** for **Windows**, **macOS**, **Android** and **iOS**
- Compatible with:
    - **AWS Polly**
    - **Azure (Bing Speech)**
    - **MaryTTS**
    - **eSpeak** and **eSpeak-NG**
    - **Klattersynth**
    - **WebGL Speech Synthesis**
    - **Google Cloud Text-To-Speak**
- Works with **Windows**, **Mac** and **Linux editors**!
- Compatible with **Unity 2017.4 – 2020**
- Supports **AR** and **VR**!
- Works with **Online Check**
- **PlayMaker** **actions**!

2.4. Integrations

- SALSA
- Localized Dialogs & Cutscenes (LDC)
- Dialogue System for Unity
- PlayMaker
- Adventure Creator
- LipSync
- SLATE
- Cinema Director
- Quest System Pro
- NPC Chat
- Klattersynth
- WebGL Speech Synthesis
- Google Cloud Text To Speech
- Online Check

2.5. Platform-specific features and limitations

2.5.1. Overview

Provider	Offline	SSML	Audio files	Words
Windows	yes	yes	yes	yes
macOS	yes	no	yes	no
Android	yes	no	yes	no
iOS	yes	no	no	yes
WSA (UWP)	yes	yes	yes	no
MaryTTS	no[1]	yes	yes	no
eSpeak	yes	yes	yes	no
AWS Polly	no	yes	yes	no
Klattersynth	yes	no	no	no
SAPI Unity	yes	yes	no	no
WebGL Speech Synthesis	no	no	no	no
Azure (Bing Speech)	no	yes	yes	no
Google Cloud TTS	no	yes	yes	no

2.5.2. Windows

- Native rate is internally limited to 20 logarithmic distributed steps
- .NET 4.0 or higher must be installed
- Minimum Windows version: 7
- Maximal number of characters per speech: 32'000 (>35min)

Important note: not all SAPI-voices support SSML! If you experience a wrong voice speaking your text, the selected voice is most likely not SSML-compatible. In this case, remove all SSML-tags from your text and let RTV speak again or you could enable "Auto Clear Tags" on the Speaker-component.

2.5.3. macOS

- Native pitch has no effect
- Native volume has no effect
- Minimum macOS version: 10.6
- Maximal number of characters per speech: 256'000 (>4h 45min)

---

1   Standalone: MaryTTS could be bundled with the app installer and run locally

2.5.4. Android

- Only one native voice at the time (can be solved by generating audio)
- Volume has no effect for generated audio (can be adjusted via AudioSource)
- Minimum Android version: 4.0.3 (API 15)
- Maximal number of characters per speech: 3'999 (>5min)

2.5.5. iOS

- Only one active native voice at the time
- Current word but no phonemes and visemes
- Minimum iOS-version: 9.0
- Maximal number of characters per speech: n/a

2.5.6. WSA (UWP)

- Native audio in XAML-builds (.NET only)
- Minimum SDK-version: 10.0
- Maximal number of characters per speech: 64'000 (>1h 15min)

2.5.7. MaryTTS

- Platforms: all
- Support for RAWMARYXML, SSML and EmotionML
- No native audio (only generated audio files)
- Minimum MaryTTS-version: 5.0
- Maximal number of characters per speech: depends on the server request size, but 8'000 (>10min) is realistic. Higher numbers can lead to timeouts.

2.5.8. eSpeak

- Platforms: Windows, macOS, Linux and Android
- Minimum eSpeak version: 1.4.0
- Maximal number of characters per speech: 32'000 (>30min)

Important note: eSpeak must be installed on the target machine.

http://espeak.sourceforge.net/

eSpeak-NG[2] is also supported:

https://github.com/espeak-ng/espeak-ng

2   to use it, change **Config.TTS_LINUX** to "speak_ng".

2.5.9. AWS Polly

- Platforms: all except WebGL and WSA (UWP)
- No native audio (only generated audio files)
- Maximal number of characters per speech: depends on the maximal request size, but 16'000 (>20min) is realistic. Higher numbers can lead to timeouts.

2.5.10.   Klattersynth

- Platforms: all
- Unity AssetStore: https://assetstore.unity.com/packages/slug/95453
- Maximal number of characters per speech: n/a

2.5.11.   SAPI Unity

- Platforms: Windows standalone (IL2CPP)
- Only one active native voice at the time
- Native rate is internally limited to 20 logarithmic distributed steps
- Maximal number of characters per speech: 64'000 (>1h 15min)

2.5.12.   WebGL Speech Synthesis

- Platforms: WebGL and Standalone
- Unity AssetStore: https://assetstore.unity.com/packages/slug/81861
- Maximal number of characters per speech: 32'000 (>30min)

2.5.13.   Azure (Bing Speech)

- Platforms: all except WebGL
- No native audio (only generated audio)
- Maximal number of characters per speech: depends on the maximal request size, but 16'000 (>20min) is realistic. Higher numbers can lead to timeouts.

2.5.14.   Google Cloud Text-To-Speech

- Platforms: all
- Unity AssetStore: https://assetstore.unity.com/packages/slug/115170
- Maximal number of characters per speech: depends on the maximal request size, but 16'000 (>20min) is realistic. Higher numbers can lead to timeouts.

3. Demonstration

The asset comes with many demo scenes to show the main usage.

3.1. Speech

This demo scene shows how to transform written lines into speech. Choose your preferred voice.

3.2. Dialog

In this demo scene you can act out a dialogue between two "people". You can choose a different voice for both participants.
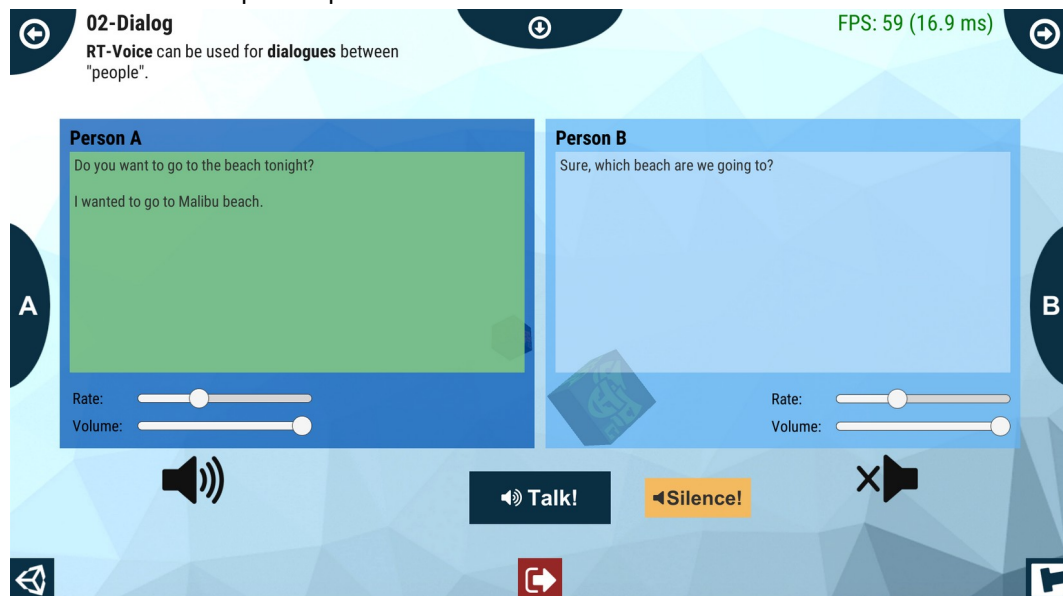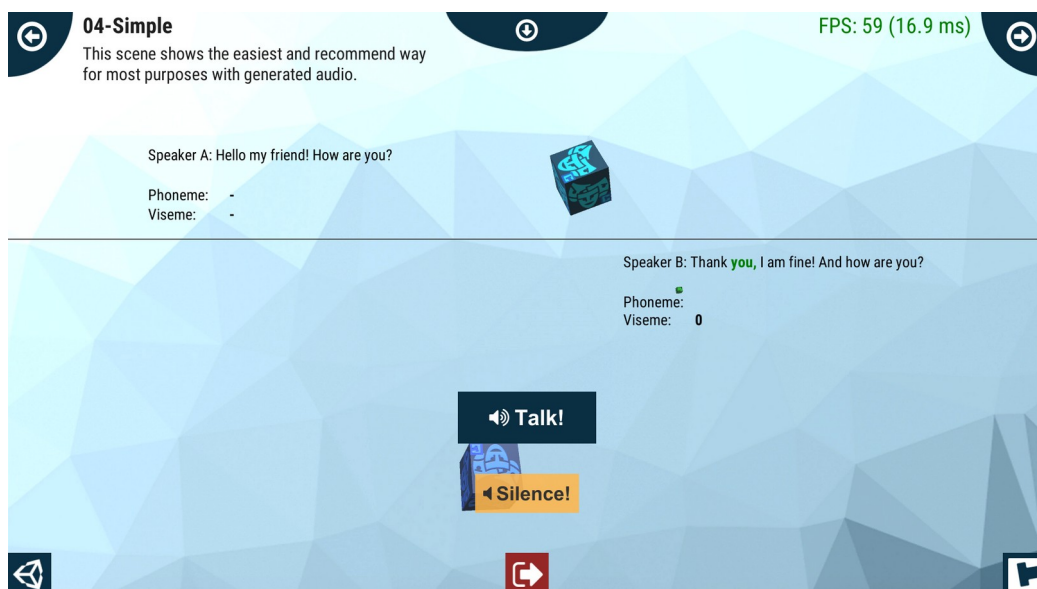
3.3. SimpleNative

The "SimpleNative" scene shows the easiest way for native audio.

3.4. Simple

The "Simple" scene shows the easiest and recommended way for most purposes with generated audio.

3.5. 3DAudio

This scene demonstrates 3D positioned and looped audio.

Needs the <u>Unity Standard Assets</u>-package.

3.6. Loudspeakers

This scene demonstrates 3D positioned loudspeakers with only one audio origin (looped). Needs the <u>Unity Standard Assets</u>-package.

3.7. SendMessage

This scene shows the usage of Unity's "SendMessage".

3.8. Sequencer

This scene shows the usage of our simple sequencer.

3.9. Exact and Exact_Native

These two scenes are showing how you can build applications with exact timing between audio and animations (e.g. lip sync).

3.10. SpeechText

This scene shows how to speak or store generated audio (see the result inside the folder "_generatedAudio").

3.11. SpeechText
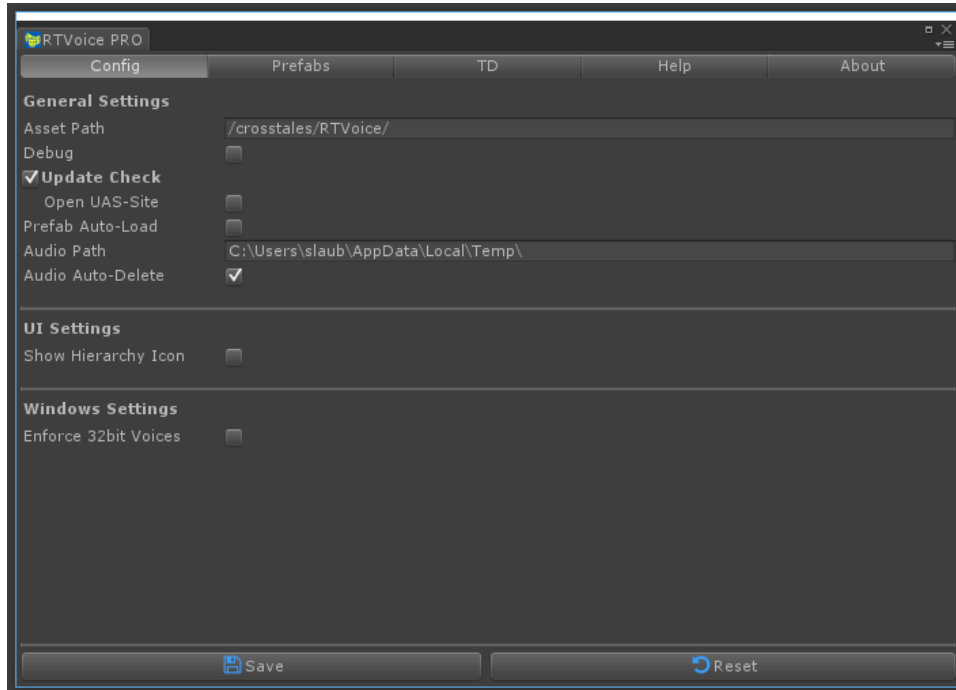
This scene shows how to speak text files with a voice (e.g. random dialogues of NPCs).

3.12. AudioFileGenerator

This scene shows how-to generate audio files from text files.

4. Setup

RT-Voice has global settings under "Edit\Preferences…" and under "Tools\RTVoice PRO\ Configuration…":

4.1. Schema

The following graphic explains the relationships between all relevant components:
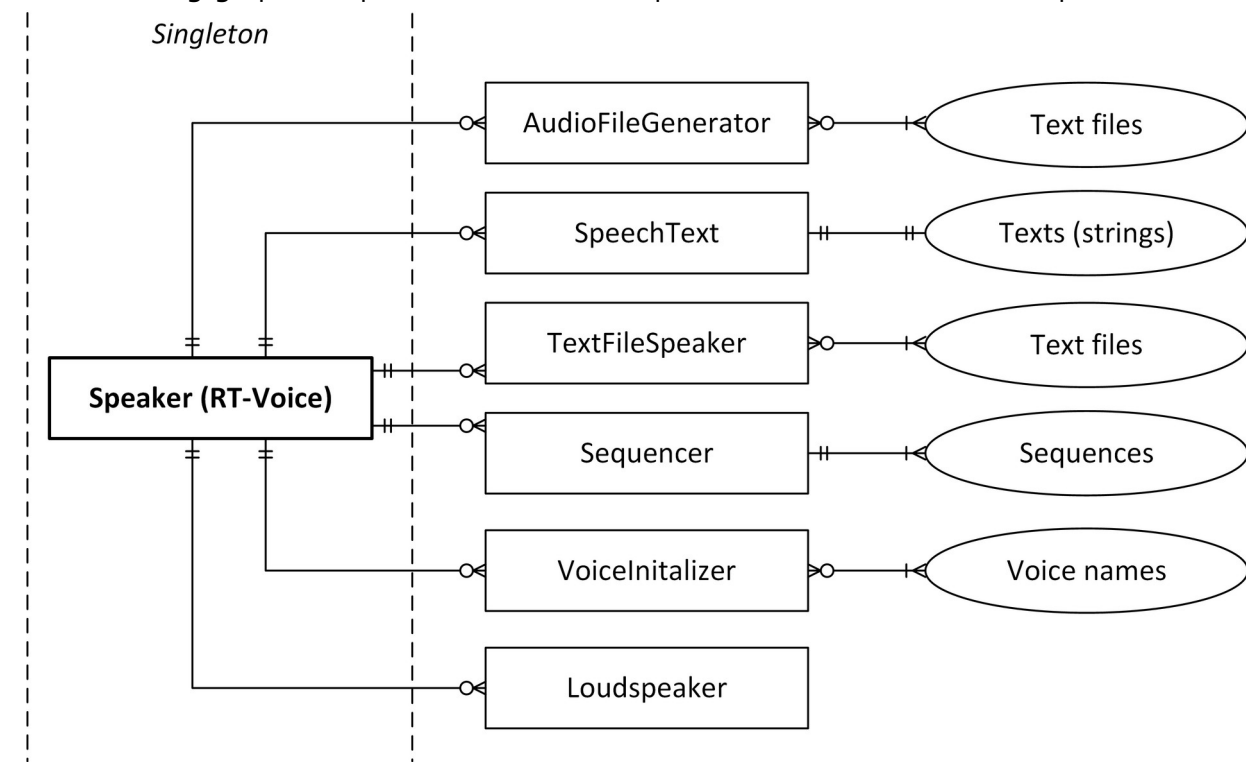
4.2. Add RT-Voice

There are four ways to add RT-Voice to your project:

1. Add the prefab **RTVoice** from Assets/Plugins/crosstales/RTVoice/Prefabs to the scene
2. Or go to *Tools => RTVoice PRO => Prefabs => **RTVoice***
3. Right-click in the *hierarchy-window => RTVoice PRO => **RTVoice***
4. Add it from the Prefabs-tab:
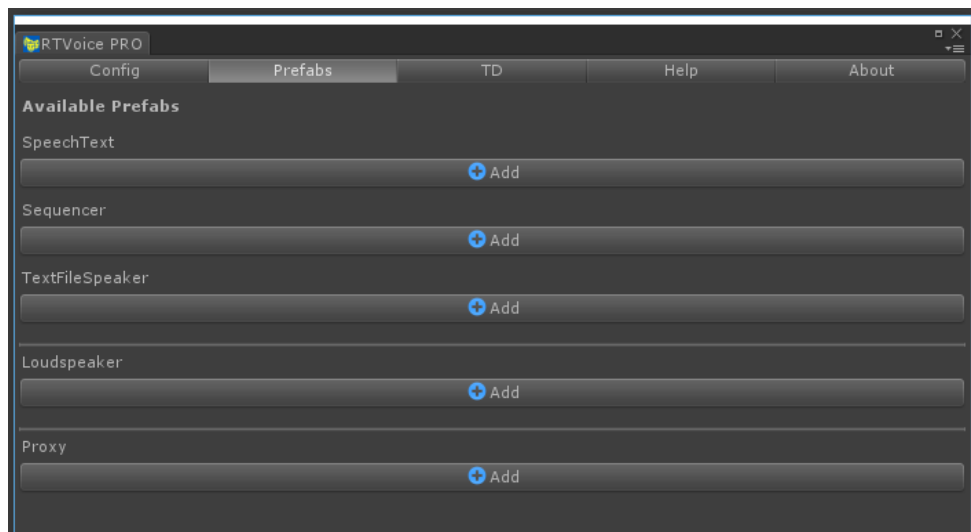
4.3. Other components

The other components can be added in the same way as "RTVoice".

4.3.1. AudioFileGenerator

This scene generates audio files from text files with lines like:

```
#Text;Output file (without extension);Voice name;Rate;Pitch;Volume
This is a test speech;Speeches\Mary01;cmu-slt-hsmm;1.2;0.85;0.95
```
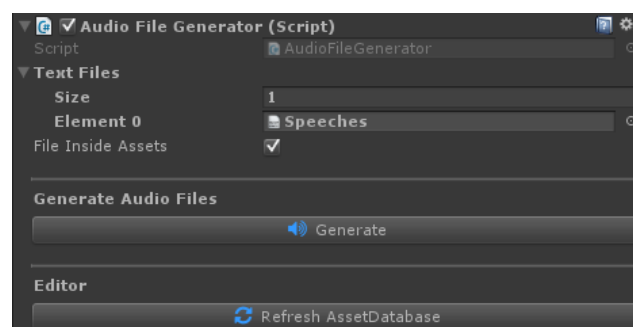
4.3.2. SpeechText

Allows to speak and store generated audio.
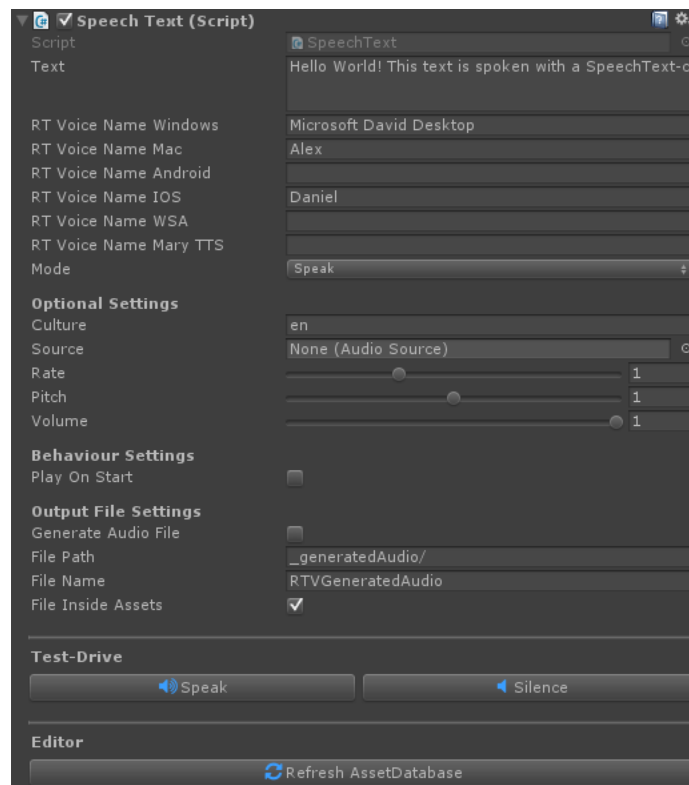
4.3.3. Sequencer

Simple sequencer for dialogues.

4.3.4. TextFileSpeaker

Allows to speak text files.
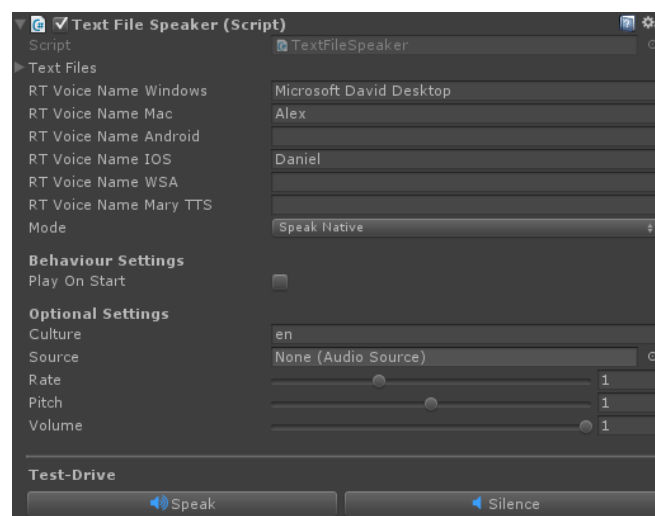
4.3.5. Loudspeaker

Loudspeaker for an AudioSource.

This is useful to use a speech on multiple locations in the game.

4.3.6. VoiceInitalizer

This component allows to initalize voices to provide lag-free speeches. It's especially useful for Android.

Add it together with RT-Voice to your first scene (e.g. splash screen).

4.3.7. Paralanguage

Paralanguage allows to use sounds inside a speech:
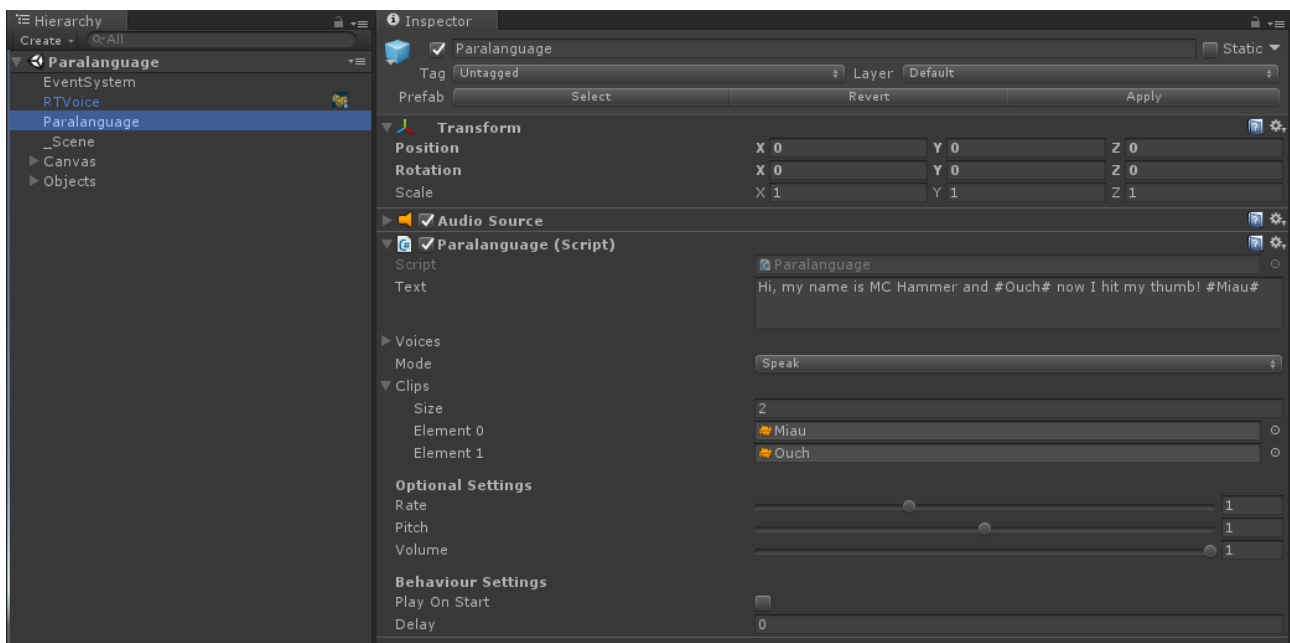
It works like this: add 1-n audio clips (names without spaces). Then add your text and fill in the desired clips with the **#** as pre- and postfix.

4.4. Differences between standard and native mode

In the **standard** mode the TTS-system of your OS will **convert** your text to an **audio** file and return it to **Unity** as an "**AudioSource**" for further use (like changing the volume, pitch etc.).

On the other hand, the **native** mode **delegates** the speech-task **entirely** to the underlying TTS-system (outside of Unity). You are **losing** some **control** but it uses slightly **less performance**.

We clearly **recommend** using the **standard** mode.

4.5. Speaker.cs vs. LiveSpeaker.cs

"Speaker.cs" is the main class of "RT-Voice" and presents the API via static methods. "LiveSpeaker.cs" on the other hand is a wrapper for "Speaker.cs" and presents the API as normal C#-instance via public methods. The main usage of "LiveSpeaker.cs" is as a receiver for "SendMessage"-calls.

4.6. MaryTTS

MaryTTS is an open-source TTS with a server, client and many voices.

It enables TTS under all Unity platforms.

You can customize everything by yourself, just follow their guides:

http://mary.dfki.de/

To enable MaryTTS, simply check "MaryTTS" in the RTVoice-component and configure the URL and port.

4.6.1. Important

The default server in RT-Voice is the test server from MaryTTS.

Never release a product with the **default configuration** and install your own server (local/remote)!

4.6.2. Account for our MaryTTS-service

We offer a service for MaryTTS. It's currently free and in early beta-stage, this means it could be sometimes slow or unavailable.

If you're interested in getting a test account, contact us.

4.6.3. Installation guide

We created a guide which should help you install a MaryTTS-server with HTTPS (needed for the WebGL-platform).

You can find it under "Assets/crosstales/RTVoice/Documentation/MaryTTS.pdf".

5. API

The asset contains various methods and the most important are explained here.

Make sure to **include** the **name space** in your relevant source files:

```
using Crosstales.RTVoice;
```

5.1. Speaker

The "Speaker.cs" is a singleton and contains the following static methods.

5.1.1. Speak

Speaks a text with a given voice and optional AudioSource.

For example:

```
//Immediately speak "hello world" with the first available voice
Speaker.Speak("hello world", audioSource);


//Immediately speak "hello world" with the first English voice (if available
else it uses the first voice on your OS)
Speaker.Speak("hello world", audioSource, Speaker.VoiceForCulture("en"));


// Prepare speak "hello world" with the first available voice (without
AudioSource.Play() - this is up to you). With this technique, you can prepare
all audio texts of a scene and you can modify the AudioSource as you like!
Speaker.Speak("hello world", audioSource, null, false);
```

5.1.2. SpeakNative

Speaks a text with a given voice.

For example:

```
//Speak "hello world" with the first available voice
Speaker.SpeakNative("hello world");


//Speak "hello world" with the first English voice (if available else it uses
the first voice on your OS)
Speaker.SpeakNative("hello world", Speaker.VoiceForCulture("en"));
```

5.1.3. **Silence**

Silence all active TTS-voices.

Example:

```
//Silence all voices
Speaker.Silence();
```

5.1.4. **Voices**

`List<Voice> Voices`

Returns all available voices (alphabetically ordered by 'Name').

5.1.5. **VoicesForGender**

`List<Voice> VoicesForGender(Model.Enum.Gender gender, string culture)`

Returns all available voices for a given gender and optional culture (alphabetically ordered by 'Name').

5.1.6. **VoiceForGender**

`Voice VoiceForGender(Model.Enum.Gender gender, string culture, int index, string fallbackCulture)`

Returns a voice for the given gender and optional culture/index/fallbackCulture.

5.1.7. **VoicesForCulture**

`List<Voice> VoicesForCulture(string culture)`

Returns all available voices for a given culture (alphabetically ordered by 'Name').

5.1.8. **VoiceForCulture**

`Voice VoiceForCulture(string culture, int index, string fallbackCulture)`

Returns a voice for the given culture and optional index/fallbackCulture.

5.1.9. **VoiceForName**

`Voice VoiceForName(string name)`

Returns the voice for the given name or null if not found.

5.1.10.   **Cultures**

`List<string> Cultures`

Returns all available cultures (alphabetically ordered by 'Culture').

5.2. Callbacks

There are various callbacks available. Subscribe them in the "OnEnable"-method and unsubscribe in "OnDisable".

5.2.1. Voices ready

`VoicesReady();`

`VoicesReady` **OnVoicesReady**;

Triggered as soon as the voices of a provider are ready to use.

5.2.2. Speak start and complete

`SpeakStart(Wrapper wrapper);`

`SpeakStart` **OnSpeakStart**;

Triggered whenever a speak is started.

`SpeakComplete(Wrapper wrapper);`

`SpeakComplete` **OnSpeakComplete**;

Triggered whenever a native speak is completed.

5.2.1. Current word (native, Windows and iOS only)

`SpeakCurrentWord(Wrapper wrapper, string[] speechTextArray, int wordIndex);`

`SpeakCurrentWord` **OnSpeakCurrentWord**;

Triggered whenever a new word is spoken (native, Windows and iOS only).

5.2.2. Current phoneme (native, Windows only)

`SpeakCurrentPhoneme(Wrapper wrapper, string phoneme);`

`SpeakCurrentPhoneme` **OnSpeakCurrentPhoneme**;

Triggered whenever a new phoneme is spoken (native mode, Windows only).

5.2.3. Current viseme (native, Windows only)

`SpeakCurrentViseme(Wrapper wrapper, string viseme);`

`SpeakCurrentViseme` **OnSpeakCurrentViseme**;

Triggered whenever a new viseme is spoken (native mode, Windows only).

5.2.4. Speak audio generation start and complete

`SpeakAudioGenerationStart(Wrapper wrapper);`

`SpeakAudioGenerationStart` **`OnSpeakAudioGenerationStart`**`;`

Triggered whenever a speak audio generation is started.

`SpeakAudioGenerationComplete(Wrapper wrapper);`

`SpeakAudioGenerationComplete` **`OnSpeakAudioGenerationComplete`**`;`

Triggered whenever a speak audio generation is completed.

5.2.5. Provider change

`ProviderChange(string provider);`

`ProviderChange` **`OnProviderChange`**`;`

Triggered whenever a provider changes (e.g. from Windows to MaryTTS).

5.2.6. Errors

`ErrorInfo(string info);`

`ErrorInfo` **`OnErorInfo`**`;`

Triggered whenever an error occurs.

5.2.7. Example

Wait until all voices are ready. Then speak the text and get informed when it starts and completes:

```
public void OnEnable() {
 // Subscribe event listeners
 Speaker.OnVoicesReady += voicesReady;
 Speaker.OnSpeakStart += speakStart;
 Speaker.OnSpeakComplete += speakComplete;
}

public void OnDisable() {
 // Unsubscribe event listeners
 Speaker.OnVoicesReady -= voicesReady;
 Speaker.OnSpeakStart -= speakStart;
 Speaker.OnSpeakComplete -= speakComplete;
}

private void voicesReady() {
 Debug.Log("voicesReady: " + wrapper);
 Speaker.SpeakNative("Hello world!");
}

private void speakStart(Model.Wrapper wrapper) {
 Debug.Log("speakStart: " + wrapper);
}

private void speakComplete(Model.Wrapper wrapper) {
 Debug.LogWarning("speakComplete: " + wrapper);
}
```

5.3. Complete API

For more details, please see the RTVoice-api.pdf

6. Additional voices

RT-Voice works great with third-party voices (e.g. [IVONA](#), [Cereproc](#) etc.).

6.1. Windows

All SAPI5-compatible voices are supported. Microsoft also provides a wide range of voices for different languages:

https://www.microsoft.com/en-us/download/details.aspx?id=27224

To install and use those voices follow this manual:

http://superuser.com/a/872573

To verify the voices, go to the path "C:\Windows\System32\Speech\SpeechUX" and double-click on "sapi.cpl". You should see the voices in the "Voice selection".

6.1.1. Important

Don't install those Microsoft voices or RTVoice won't work:

- hui hui
- hun yee
- han han

6.2. macOS

Apple delivers many voices for different languages. To add or customize them, follow the tutorial below:

http://osxdaily.com/2011/07/25/how-to-add-new-voices-to-mac-os-x/

6.3. Android

You can add various voices on your Android phone:

http://www.geoffsimons.com/2012/06/7-best-android-text-to-speech-engines.html

There is also a possibility to download high quality voices:

http://www.androidauthority.com/google-text-to-speech-engine-659528/

6.4. iOS

You can only change the quality of the installed voices:

https://support.apple.com/en-us/HT202362

6.5. WSA (UWP)

No information so far. If you know a working guide, please let us know.

6.6. MaryTTS

http://mary.dfki.de/

6.7. eSpeak

http://espeak.sourceforge.net/languages.html

7. Setup eSpeak

eSpeak and esSpeak-NG are open-source TTS solutions that work under Windows, macOS, Linux and Android.

To take advantage of it, please install it on your PC.

7.1. Windows

http://sourceforge.net/projects/espeak/files/espeak/espeak-1.48/setup_espeak-1.48.04.exe

7.1. macOS

http://sourceforge.net/projects/espeak/files/espeak/espeak-1.45/espeak-1.45.04-OSX.zip

7.2. Linux

```
sudo apt-get install espeak
```

7.3. eSpeak-NG

Please see the installation details:

https://github.com/espeak-ng/espeak-ng#binaries

8. Third-party support (PlayMaker etc.)

„RT-Voice" supports various products from other companies. Please import the desired packages from the „3rd party"-folder.

9. Verify installation

Check if RT-Voice is installed:

```
#if CT_RTV
     Debug.Log("RTV installed: " + Util.Constants.ASSET_VERSION);
#else
     Debug.LogWarning("RTV NOT installed!");
#endif
```

10. Upgrade to new version

Follow this steps to upgrade your version of "RT-Voice PRO":

1. Update "RT-Voice PRO" to the latest version from the "Unity AssetStore"
2. Inside your project in Unity, go to menu "File" => "New Scene"
3. Delete the "Assets/Plugins/crosstales/RTVoice" folder from the Project-view[3]
4. Import the latest version downloaded from the "Unity AssetStore"

11.Important notes

After this setup, the "RT-Voice" is ready to use. It is important to know that it uses the **singleton**-pattern, which means that **once instantiated**, the "RT-Voice" will **live until** the application is **terminated**.

Remember: it must be instantiated before you try to access it! Otherwise it's not possible to use it.

---

3   Before 2.9.3: Assets/crosstales/RTVoice

12. Problems, improvements etc.

If you encounter any problems with this asset, just [send us an email](#) with a problem description and the invoice number and we will try to solve it.

13. Release notes

See "VERSIONS.txt" under "Assets/crosstales/RTVoice/Documentation" or online:

https://crosstales.com/media/data/assets/rtvoice/VERSIONS.txt

14. Credits

The icons are based on [Font Awesome](#).

15. Contact and further information

crosstales LLC

Schanzeneggstrasse 1

CH-8002 Zürich

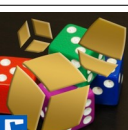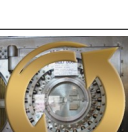|	
Homepage:	https://www.crosstales.com/en/portfolio/rtvoice/
Email:	rtvoice@crosstales.com
AssetStore:	https://assetstore.unity.com/lists/crosstales-42213
Forum:	http://forum.unity3d.com/threads/rt-voice-run-time-text-to-speech-solution.340046/
Documentation:	https://www.crosstales.com/media/data/assets/rtvoice/RTVoice-doc.pdf
API:	http://www.crosstales.com/en/assets/rtvoice/api/
WebGL-Demo:	https://www.crosstales.com/media/data/assets/rtvoice/webgl/
Windows-Demo:	https://www.crosstales.com/media/data/assets/rtvoice/downloads/RTVoice_demo_win.zip
Mac-Demo:	https://www.crosstales.com/media/data/assets/rtvoice/downloads/RTVoice_demo_mac.zip
Android-Demo:	https://www.crosstales.com/media/rtvoice/RTVoice.apk
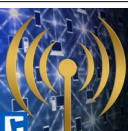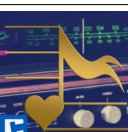
16. Our other assets

3D Skybox	Those beautiful packages contain professional 8k, HDR, stereoscopic 360° real-world skyboxes for your projects.
Bad Word Filter	The "Bad Word Filter" (aka profanity or obscenity filter) is exactly what the title suggests: a tool to filter swearwords and other "bad sentences".
DJ	DJ is a player for external music-files. It allows a user to play his own sound inside any Unity-app. It can also read ID3-tags.
File Browser	File Browser is a wrapper for native file dialogs on Windows, macOS, Linux and UWP (WSA).
Online Check	You need a reliable solution to check for **Internet availability**? Here it is!
Radio	Radio allows implementing free music from Internet radio stations into your project..
True Random	True Random can generate "true random" numbers for you and your application. The randomness comes from atmospheric noise, which for many purposes is better than the pseudo-random number algorithms typically used in computer programs.
Turbo Backup	Turbo Backup is the fastest and safest way to backup your Unity project. It only stores the difference between the last backup, this makes it incredible fast.
Turbo Builder	Turbo Builder creates builds for multiple platforms in one click. It works together with Turbo Switch to offer an incredible fast build pipeline.

	Turbo Switch is a Unity editor extension to reduce the time for assets to import during platform switches. We measured speed improvements up to 100x faster than the built-in switch in Unity.
Turbo Switch	