

Package: text2speech (via r-universe)

October 1, 2026

Type Package

Title Text to Speech Conversion

Description Converts text into speech using various text-to-speech (TTS) engines and provides an unified interface for accessing their functionality. With this package, users can easily generate audio files of spoken words, phrases, or sentences from plain text data. The package supports multiple TTS engines, including Google's 'Cloud Text-to-Speech API', 'Amazon Polly', Microsoft's 'Cognitive Services Text to Speech REST API', the 'Speechify Text-to-Speech API', and a free TTS engine called 'Coqui TTS'.

Version 1.2.0

License GPL-3

Suggests aws.polly, covr, patrick, rmarkdown, stringi, testthat (>= 3.0.0)

Encoding UTF-8

VignetteBuilder knitr

URL <https://github.com/jhudsl/text2speech>

BugReports <https://github.com/jhudsl/text2speech/issues>

Imports aws.signature, cli, dplyr, googleAuthR, googleLanguageR, htr2, jsonlite, knitr, magrittr, conrad, tidyr, tuneR, utils, withr

Roxygen list(markdown = TRUE)

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libssl-dev

Repository <https://jhudsl.r-universe.dev>

Date/Publication 2026-09-29 19:11:24 UTC

RemoteUrl <https://github.com/jhudsl/text2speech>

RemoteRef HEAD

RemoteSha ba024110bb0a0d63ba67c64d14843319aaa99d99

Contents

pcm_to_wav	2
play_audio	3
set_coqui_path	4
tts	4
tts_auth	7
tts_bind_wav	8
tts_default_voice	9
tts_speak_engine	9
tts_voices	10
Index	13

pcm_to_wav	<i>Convert PCM to WAV</i>
------------	---------------------------

Description

Accepts PCM audio data as input and generates a corresponding Wav file

Usage

```
pcm_to_wav(  
  input,  
  output = tempfile(fileext = ".wav"),  
  sample_rate = 16000,  
  extensible = FALSE  
)
```

Arguments

input	output from ‘get_synthesis‘ from aws.polly or PCM filename
output	output file for Wav file
sample_rate	Sampling rate for tuneR::Wave
extensible	passed to tuneR::writeWave

Value

A filename of the output

Examples

```
## Not run:
fname = system.file("extdata", "pcm_file.wav", package = "text2speech")
res = pcm_to_wav(fname)
testthat::expect_error(tuneR::readWave(fname))
testthat::expect_is(tuneR::readWave(res), "Wave")

## End(Not run)
## Not run:
if (requireNamespace("aws.polly", quietly = TRUE)) {
  text = "hey, ho, let's go!"
  if (tts_amazon_auth()) {
    res = tts_amazon(text, output_format = "wav")
  }
}

## End(Not run)
```

play_audio

Play audio in a browser

Description

This uses HTML5 audio tags to play audio in your browser.

Usage

```
play_audio(audio = "output.wav", html = "player.html")
```

Arguments

audio	The file location of the audio file. Must be supported by HTML5.
html	The html file location that will be created to host the audio file.

Details

Borrowed from googleLanguageR::gl_talk_player()

Examples

```
## Not run:
  play_audio(audio = "audio.wav", html = "player.html")

## End(Not run)
```

set_coqui_path	<i>Point to local coqui tts Executable File</i>
----------------	---

Description

Function to set an option that points to the local coqui tts Executable File `tts`.

Usage

```
set_coqui_path(path)
```

Arguments

path	path to the local coqui tts Executable File
------	---

Details

List of possible file path locations for the local coqui tts Executable File

Linux /usr/bin/tts, /usr/local/bin/tts

Mac /opt/homebrew/Caskroom/miniforge/base/bin/tts

Windows C:\\Program Files\\tts

Value

Returns nothing, function sets the option variable `path_to_coqui`.

Examples

```
set_coqui_path("~/path/to/tts")
```

tts	<i>Text-to-Speech (Speech Synthesis)</i>
-----	--

Description

Convert text-to-speech using various engines, including Amazon Polly, Coqui TTS, Google Cloud Text-to-Speech API, Microsoft Cognitive Services Text to Speech REST API, and Speechify Text-to-Speech API.

With the exception of Coqui TTS and Speechify, all these engines are accessible as R packages:

- `aws.polly` is a client for Amazon Polly.
- `googleLanguageR` is a client to the Google Cloud Text-to-Speech API.
- `conrad` is a client to the Microsoft Cognitive Services Text to Speech REST API

Speechify is called directly through its REST API (<https://docs.speechify.ai>) using the key in the `SPEECHIFY_API_KEY` environment variable.

Usage

```
tts(  
  text,  
  output_format = c("mp3", "wav"),  
  service = c("amazon", "google", "microsoft", "coqui", "speechify"),  
  bind_audio = TRUE,  
  ...  
)  
  
tts_amazon(  
  text,  
  output_format = c("mp3", "wav"),  
  voice = "Joanna",  
  bind_audio = TRUE,  
  save_local = FALSE,  
  save_local_dest = NULL,  
  ...  
)  
  
tts_google(  
  text,  
  output_format = c("mp3", "wav"),  
  voice = "en-US-Standard-C",  
  bind_audio = TRUE,  
  save_local = FALSE,  
  save_local_dest = NULL,  
  ...  
)  
  
tts_microsoft(  
  text,  
  output_format = c("mp3", "wav"),  
  voice = NULL,  
  bind_audio = TRUE,  
  save_local = FALSE,  
  save_local_dest = NULL,  
  ...  
)  
  
tts_coqui(  
  text,  
  exec_path,  
  output_format = c("wav", "mp3"),  
  model_name = "tacotron2-DDC_ph",  
  vocoder_name = "ljspeech/univnet",  
  bind_audio = TRUE,  
  save_local = FALSE,  
  save_local_dest = NULL,
```

```

    ...
  )

  tts_speechify(
    text,
    output_format = c("mp3", "wav"),
    voice = "geffen_32",
    model = "simba-3.2",
    bind_audio = TRUE,
    save_local = FALSE,
    save_local_dest = NULL,
    ...
  )

```

Arguments

<code>text</code>	A character vector of text to be spoken
<code>output_format</code>	Format of output files: "mp3" or "wav"
<code>service</code>	Service to use (Amazon, Google, Microsoft, Coqui, or Speechify)
<code>bind_audio</code>	Should the <code>tts_bind_wav()</code> be run on after the audio has been created, to ensure that the length of text and the number of rows is consistent?
<code>...</code>	Additional arguments
<code>voice</code>	Full voice name
<code>save_local</code>	Should the audio file be saved locally?
<code>save_local_dest</code>	If to be saved locally, destination where output file will be saved
<code>exec_path</code>	System path to Coqui TTS executable
<code>model_name</code>	(Coqui TTS only) Deep Learning model for Text-to-Speech Conversion
<code>vocoder_name</code>	(Coqui TTS only) Voice coder used for speech coding and transmission
<code>model</code>	(Speechify only) Model used for synthesis. The default, "simba-3.2", is English only; use "simba-3.0" for voices in other languages

Value

A standardized tibble featuring the following columns:

- `index` : Sequential identifier number
- `original_text` : The text input provided by the user
- `text` : In case `original_text` exceeds the character limit, `text` represents the outcome of splitting `original_text`. Otherwise, `text` remains the same as `original_text`.
- `wav` : Wave object (S4 class)
- `file` : File path to the audio file
- `audio_type` : The audio format, either mp3 or wav
- `duration` : The duration of the audio file
- `service` : The text-to-speech engine used

Examples

```
## Not run:
# Amazon Polly
tts("Hello world! This is Amazon Polly", service = "amazon")

tts("Hello world! This is Coqui TTS", service = "coqui")

tts("Hello world! This is Google Cloud", service = "google")

tts("Hello world! This is Microsoft", service = "microsoft")

tts("Hello world! This is Speechify", service = "speechify")

## End(Not run)
```

tts_auth

*Authentication for Text-to-Speech (Speech Synthesis) Engines***Description**

Verify the authentication status of different text-to-speech engines, including Amazon Polly, Coqui TTS, Google Cloud Text-to-Speech API, Microsoft Cognitive Services Text to Speech REST API, and Speechify Text-to-Speech API.

Usage

```
tts_auth(
  service = c("amazon", "google", "microsoft", "coqui", "speechify"),
  key_or_json_file = NULL,
  ...
)

tts_amazon_auth(key_or_json_file = NULL, ...)

tts_google_auth(key_or_json_file = NULL, ...)

tts_microsoft_auth(key_or_json_file = NULL, ...)

tts_coqui_auth()

tts_speechify_auth(key_or_json_file = NULL, ...)
```

Arguments

service	Service to use (Amazon, Google, Microsoft, Coqui, or Speechify)
key_or_json_file	Either an API key (for Microsoft or Speechify) or JSON file (for Google)
...	Additional arguments

Details

To determine the availability of Coqui TTS, `tts_auth()` examines whether the `tts` executable exists on local system.

For Speechify, the API key is read from the `SPEECHIFY_API_KEY` environment variable. Passing a key as `key_or_json_file` sets that variable for the session. Keys are created at <https://platform.speechify.ai/api-keys>.

Value

A logical indicator of authorization

Examples

```
# Amazon Polly
tts_auth("amazon")

# Google Cloud Text-to-Speech API
tts_auth("google")

# Microsoft Cognitive Services Text to Speech REST API
tts_auth("microsoft")

# Coqui TTS
tts_auth("coqui")

# Speechify Text-to-Speech API
tts_auth("speechify")
```

tts_bind_wav	<i>Bind WAVs together</i>
--------------	---------------------------

Description

As the data are split due to limits of the API, `tts_bind_wav()` allows the text and the results to be harmonized

Usage

```
tts_bind_wav(result, same_sample_rate = TRUE)
```

Arguments

- `result` A data.frame from `tts()`.
- `same_sample_rate` A logical value indicating whether to force the same sample rate.

Value

A data.frame with the same structure as that of tts

Examples

```
## Not run:
# Same sample rate
tts_bind_wav(res, same_sample_rate = TRUE)

# Different sample rate
tts_bind_wav(res, same_sample_rate = FALSE)

## End(Not run)
```

tts_default_voice	<i>Default voice for text-to-speech engine</i>
-------------------	--

Description

Default voice for text-to-speech engine

Usage

```
tts_default_voice(
  service = c("amazon", "google", "microsoft", "coqui", "speechify")
)
```

Arguments

service	Text-to-speech engine
---------	-----------------------

tts_speak_engine	<i>Speak Engine for knitr</i>
------------------	-------------------------------

Description

Speak Engine for knitr

Usage

```
tts_speak_engine(options)
```

Arguments

options	A list of chunk options. Usually this is just the object options passed to the engine function; see knit_engines
---------	--

Value

A character string generated from the source code and output using the appropriate output hooks.

Examples

```
## Not run:
knitr::knit_engines$set(speak = tts_speak_engine)
options = list(
  code = "hey let's go to the park",
  eval = FALSE,
  label = "random",
  fig.path = tempdir(),
  echo = TRUE, results = "asis",
  engine = "speak")
tts_speak_engine(options)
  if (tts_auth("google")) {
    options$eval = TRUE
    tts_speak_engine(options)
  }

## End(Not run)
```

tts_voices

Text-to-Speech (Speech Synthesis) Voices

Description

Various services offer a range of voice options:

- Amazon Polly : <https://docs.aws.amazon.com/polly/latest/dg/available-voices.html>
- Microsoft Cognitive Services Text to Speech REST API : <https://learn.microsoft.com/en-us/azure/cognitive-services/speech-service/language-support?tabs=tts#voice-styles-and-roles>
- Google Cloud Text-to-Speech API : <https://cloud.google.com/text-to-speech/docs/voices>
- Coqui TTS : <https://huggingface.co/spaces/coqui/CoquiTTS>
- Speechify Text-to-Speech API : <https://docs.speechify.ai/build/api-reference/v1/voices/get>

Usage

```
tts_voices(
  service = c("amazon", "google", "microsoft", "coqui", "speechify"),
  ...
)

tts_amazon_voices(...)
```

```

tts_microsoft_voices(region = "westus")

tts_google_voices(...)

tts_coqui_voices()

tts_speechify_voices(...)

```

Arguments

service	Service to use (Amazon, Google, Microsoft, Coqui, or Speechify)
...	Additional arguments to service voice listings.
region	(Microsoft only) Region of your Microsoft Speech Service API Key

Value

(Amazon, Microsoft, Google, and Speechify) A standardized `data.frame` featuring the following columns:

- `voice` : Name of the voice
- `language` : Spoken language
- `language_code` : Abbreviation for the language of the speaker
- `gender` : Male or female
- `service` : The text-to-speech engine used

(Coqui TTS) A `tibble` featuring the following columns:

- `language` : Spoken language
- `dataset` : Dataset the deep learning model was trained on
- `model_name` : Name of deep learning model
- `service` : The text-to-speech engine used

Examples

```

# Amazon Polly
if (tts_auth(service = "amazon")) {
  tts_voices(service = "amazon")
}

# Microsoft Cognitive Services Text to Speech REST API
if (tts_auth(service = "microsoft")) {
  tts_voices(service = "microsoft")
}

# Google Cloud Text-to-Speech API
if (tts_auth(service = "google")) {
  tts_voices(service = "google")
}

```

```
}

# Coqui TTS
if (tts_auth(service = "coqui")) {
  tts_voices(service = "coqui")
}

# Speechify Text-to-Speech API
if (tts_auth(service = "speechify")) {
  tts_voices(service = "speechify")
}
```

Index

knit_engines, 9

pcm_to_wav, 2

play_audio, 3

set_coqui_path, 4

tts, 4

tts(), 8

tts_amazon(tts), 4

tts_amazon_auth(tts_auth), 7

tts_amazon_voices(tts_voices), 10

tts_auth, 7

tts_bind_wav, 8

tts_bind_wav(), 6

tts_coqui(tts), 4

tts_coqui_auth(tts_auth), 7

tts_coqui_voices(tts_voices), 10

tts_default_voice, 9

tts_google(tts), 4

tts_google_auth(tts_auth), 7

tts_google_voices(tts_voices), 10

tts_microsoft(tts), 4

tts_microsoft_auth(tts_auth), 7

tts_microsoft_voices(tts_voices), 10

tts_speak_engine, 9

tts_speechify(tts), 4

tts_speechify_auth(tts_auth), 7

tts_speechify_voices(tts_voices), 10

tts_voices, 10

tuneR::Wave, 2

tuneR::writeWave, 2