

# Package ‘ari’

August 19, 2019

**Type** Package

**Title** Automated R Instructor

**Version** 0.3.2

**Description** Create videos from 'R Markdown' documents, or images and audio files. These images can come from image files or HTML slides, and the audio files can be provided by the user or computer voice narration can be created using 'Amazon Polly'. The purpose of this package is to allow users to create accessible, translatable, and reproducible lecture videos. See <https://aws.amazon.com/polly/> for more information.

**SystemRequirements** ffmpeg (>= 3.2.4)

**Depends** R (>= 3.1.0)

**Imports** text2speech (>= 0.2.4), tuneR, webshot, purrr, rmarkdown, xml2, rvest, tools, progress, hms

**Suggests** testthat, grDevices, xaringan, aws.signature, aws.polly

**License** MIT + file LICENSE

**URL** <http://github.com/seankross/ari>

**BugReports** <http://github.com/seankross/ari/issues>

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Author** Sean Kross [aut, cre],  
John Muschelli [ctb]

**Maintainer** Sean Kross <sean@seankross.com>

**Repository** CRAN

**Date/Publication** 2019-08-19 19:50:02 UTC

**R topics documented:**

ari_burn_subtitles . . . . .	2
ari_example . . . . .	3
ari_narrate . . . . .	3
ari_spin . . . . .	4
ari_stitch . . . . .	6
ari_talk . . . . .	7
ffmpeg_codecs . . . . .	8
ffmpeg_exec . . . . .	8
set_audio_codec . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

ari_burn_subtitles	<i>Burn Subtitles into a video</i>
--------------------	------------------------------------

---

**Description**

Burn Subtitles into a video

**Usage**

```
ari_burn_subtitles(video, srt, verbose = FALSE)
```

**Arguments**

video	Video in mp4 format
srt	Subtitle file in srt format
verbose	print diagnostic messages. If > 1, then more are printed

**Value**

Name of output video

**Note**

This needs ffmpeg that was compiled with `--enable-libass` as per <https://trac.ffmpeg.org/wiki/HowToBurnSubtitlesIntoVideo>

---

ari_example	<i>Get the path to an ari example file</i>
-------------	--

---

### Description

This function allows you to quickly access files that are used in the ari documentation.

### Usage

```
ari_example(path = NULL)
```

### Arguments

path	The name of the file. If no argument is provided then all of the example files will be listed.
------	--

### Value

A character string

### Examples

```
ari_example("ari_intro.Rmd")
```

---

ari_narrate	<i>Create a video from slides and a script</i>
-------------	--

---

### Description

`ari_narrate` creates a video from a script written in markdown and HTML slides created with [rmarkdown](#) or a similar package. This function uses [Amazon Polly](#) via [ari\\_spin](#).

### Usage

```
ari_narrate(script, slides, output = tempfile(fileext = ".mp4"),
  voice = "Joanna", service = "amazon",
  capture_method = c("vectorized", "iterative"), subtitles = FALSE,
  ..., verbose = FALSE, audio_codec = get_audio_codec(),
  video_codec = get_video_codec(), cleanup = TRUE)
```

**Arguments**

script	Either a markdown file where every paragraph will be read over a corresponding slide, or an .Rmd file where each HTML comment will be used for narration.
slides	A path or URL for an HTML slideshow created with <a href="#">rmarkdown</a> , <a href="#">xaringan</a> , or a similar package.
output	The path to the video file which will be created.
voice	The voice you want to use. See <a href="#">tts_voices</a> for more information about what voices are available.
service	speech synthesis service to use, passed to <a href="#">tts</a> .
capture_method	Either "vectorized" or "iterative". The vectorized mode is faster though it can cause screens to repeat. If making a video from an <a href="#">ioslides_presentation</a> you should use "iterative".
subtitles	Should a .srt file be created with subtitles? The default value is FALSE. If TRUE then a file with the same name as the output argument will be created, but with the file extension .srt.
...	Arguments that will be passed to <a href="#">webshot</a> .
verbose	print diagnostic messages. If > 1, then more are printed
audio_codec	The audio encoder for the splicing. If this fails, try copy.
video_codec	The video encoder for the splicing. If this fails, see <code>ffmpeg -codecs</code>
cleanup	If TRUE, interim files are deleted

**Value**

The output from [ari\\_spin](#)

**Examples**

```
## Not run:

#
ari_narrate(system.file("test", "ari_intro_script.md", package = "ari"),
            system.file("test", "ari_intro.html", package = "ari"),
            voice = "Joey")

## End(Not run)
```

---

ari\_spin

*Create a video from images and text*


---

**Description**

Given equal length vectors of paths to images (preferably .jpgs or .pngs) and strings which will be [synthesized](#) by [Amazon Polly](#) or any other synthesizer available in [tts](#), this function creates an .mp4 video file where each image is shown with its corresponding narration. This function uses [ari\\_stitch](#) to create the video.

## Usage

```
ari_spin(images, paragraphs, output = tempfile(fileext = ".mp4"),
         voice = "Joanna", service = "amazon", subtitles = FALSE, ...)
```

## Arguments

images	A vector of paths to images.
paragraphs	A vector strings that will be spoken by Amazon Polly.
output	A path to the video file which will be created.
voice	The voice you want to use. See <a href="#">tts_voices</a> for more information about what voices are available.
service	speech synthesis service to use, passed to <a href="#">tts</a>
subtitles	Should a .srt file be created with subtitles? The default value is FALSE. If TRUE then a file with the same name as the output argument will be created, but with the file extension .srt.
...	additional arguments to <a href="#">ari_stitch</a>

## Details

This function needs to connect to [Amazon Web Services](#) in order to create the narration. You can find a guide for accessing AWS from R [here](#). For more information about how R connects to Amazon Polly see the [aws.polly](#) documentation [here](#).

## Value

The output from [ari\\_stitch](#)

## Examples

```
## Not run:

slides <- system.file("test", c("mab2.png", "mab1.png"),
  package = "ari")
sentences <- c("Welome to my very interesting lecture.",
  "Here are some fantastic equations I came up with.")
ari_spin(slides, sentences, voice = "Joey")

## End(Not run)
```

ari\_stitch

*Create a video from images and audio***Description**

Given a vector of paths to images (preferably .jpgs or .pngs) and a flat list of [Waves](#) of equal length this function will create an .mp4 video file where each image is shown with its corresponding audio. Take a look at the [readWave](#) function if you want to import your audio files into R. Please be sure that all images have the same dimensions.

**Usage**

```
ari_stitch(images, audio, output = tempfile(fileext = ".mp4"),
  verbose = FALSE, cleanup = TRUE, ffmpeg_opts = "",
  divisible_height = TRUE, audio_codec = get_audio_codec(),
  video_codec = get_video_codec(), video_sync_method = "-1",
  audio_bitrate = NULL, video_bitrate = NULL,
  pixel_format = "yuv420p", fast_start = TRUE, deinterlace = TRUE,
  stereo_audio = TRUE)
```

**Arguments**

images	A vector of paths to images.
audio	A list of Waves from tuneR.
output	A path to the video file which will be created.
verbose	print diagnostic messages. If > 1, then more are printed
cleanup	If TRUE, interim files are deleted
ffmpeg_opts	additional options to send to ffmpeg. This is an advanced option, use at your own risk
divisible_height	Make height divisible by 2, which may be required if getting "height not divisible by 2" error.
audio_codec	The audio encoder for the splicing. If this fails, try copy.
video_codec	The video encoder for the splicing. If this fails, see <code>ffmpeg -codecs</code>
video_sync_method	Video sync method. Should be "auto" or "vfr" or a numeric. See <a href="https://ffmpeg.org/ffmpeg.html">https://ffmpeg.org/ffmpeg.html</a> .
audio_bitrate	Bit rate for audio. Passed to <code>-b:a</code> .
video_bitrate	Bit rate for video. Passed to <code>-b:v</code> .
pixel_format	pixel format to encode for 'ffmpeg'.
fast_start	Adding 'faststart' flags for YouTube and other sites, see <a href="https://trac.ffmpeg.org/wiki/Encode/YouTube">https://trac.ffmpeg.org/wiki/Encode/YouTube</a>
deinterlace	should the video be de-interlaced, see <a href="https://ffmpeg.org/ffmpeg-filters.html">https://ffmpeg.org/ffmpeg-filters.html</a> , generally for YouTube
stereo_audio	should the audio be forced to stereo, corresponds to <code>'-ac 2'</code>

## Details

This function uses **FFmpeg** which you should be sure is installed before using this function. If running `Sys.which("ffmpeg")` in your R console returns an empty string after installing FFmpeg then you should set the path to FFmpeg on your computer to an environmental variable using `Sys.setenv(ffmpeg = "path/to/ffmpeg")`. The environmental variable will always override the result of `Sys.which("ffmpeg")`.

## Value

A logical value, with the attribute `outfile` for the output file.

## Examples

```
## Not run:
if (ffmpeg_version_sufficient()) {
  result = ari_stitch(
    ari_example(c("mab1.png", "mab2.png")),
    list(tuneR::noise(), tuneR::noise())
  )
}

## End(Not run)
```

---

ari\_talk

*Create spoken audio files*

---

## Description

A simple function for demoing how spoken text will sound.

## Usage

```
ari_talk(paragraphs, output = tempfile(fileext = ".wav"),
         voice = "Joanna", service = "amazon")
```

## Arguments

<code>paragraphs</code>	A vector strings that will be spoken by Amazon Polly.
<code>output</code>	A path to the audio file which will be created.
<code>voice</code>	The voice you want to use. See <a href="#">tts_voices</a> for more information about what voices are available.
<code>service</code>	speech synthesis service to use, passed to <a href="#">tts</a>

## Value

A Wave output object, with the attribute `outfile` of the output file name.

ffmpeg\_codecs

*Get Codecs for ffmpeg*

---

**Description**

Get Codecs for ffmpeg

**Usage**

```
ffmpeg_codecs()

ffmpeg_video_codecs()

ffmpeg_audio_codecs()

ffmpeg_muxers()

ffmpeg_version()

ffmpeg_version_sufficient()

check_ffmpeg_version()
```

**Value**

A 'data.frame' of codec names and capabilities

**Examples**

```
## Not run:
if (ffmpeg_version_sufficient()) {
  ffmpeg_codecs()
  ffmpeg_video_codecs()
  ffmpeg_audio_codecs()
}

## End(Not run)
```

---

ffmpeg\_exec*Get Path to ffmpeg Executable*

---

**Description**

Get Path to ffmpeg Executable



**Usage**

```
ffmpeg_exec(quote = FALSE)
```

```
have_ffmpeg_exec()
```

**Arguments**

quote                    should [shQuote](#) be run before returning?

**Value**

The path to the ffmpeg executable, or an error.

**Note**

This looks using `'Sys.getenv("ffmpeg")'` and `'Sys.which("ffmpeg")'` to find `'ffmpeg'`. If `'ffmpeg'` is not in your PATH, then please set the path to `'ffmpeg'` using `'Sys.setenv(ffmpeg = "/path/to/ffmpeg")'`

**Examples**

```
## Not run:
if (have_ffmpeg_exec()) {
  ffmpeg_exec()
}

## End(Not run)
```

---

set\_audio\_codec

*Set Default Audio and Video Codecs*

---

**Description**

Set Default Audio and Video Codecs

**Usage**

```
set_audio_codec(codec)
```

```
set_video_codec(codec = "libx264")
```

```
get_audio_codec()
```

```
get_video_codec()
```

```
audio_codec_encode(codec)
```

```
video_codec_encode(codec)
```

**Arguments**

codec                   The codec to use or get for audio/video. Uses the 'ffmpeg\_audio\_codec' and 'ffmpeg\_video\_codec' options to store this information.

**Value**

A 'NULL' output

**See Also**

[ffmpeg\_codecs()] for options

**Examples**

```
## Not run:
if (have_ffmpeg_exec()) {
  print(ffmpeg_version())
  get_audio_codec()
  set_audio_codec(codec = "libfdk_aac")
  get_audio_codec()
  set_audio_codec(codec = "aac")
  get_audio_codec()
}
if (have_ffmpeg_exec()) {
  get_video_codec()
  set_video_codec(codec = "libx265")
  get_video_codec()
  set_video_codec(codec = "libx264")
  get_video_codec()
}
## empty thing
if (have_ffmpeg_exec()) {
  video_codec_encode("libx264")

  audio_codec_encode("aac")
}

## End(Not run)
```

# Index

ari\_burn\_subtitles, 2  
ari\_example, 3  
ari\_narrate, 3  
ari\_spin, 3, 4, 4  
ari\_stitch, 4, 5, 6  
ari\_talk, 7  
audio\_codec\_encode (set\_audio\_codec), 9  
aws.polly, 5  
  
check\_ffmpeg\_version (ffmpeg\_codecs), 8  
  
ffmpeg\_audio\_codecs (ffmpeg\_codecs), 8  
ffmpeg\_codecs, 8  
ffmpeg\_exec, 8  
ffmpeg\_muxers (ffmpeg\_codecs), 8  
ffmpeg\_version (ffmpeg\_codecs), 8  
ffmpeg\_version\_sufficient  
    (ffmpeg\_codecs), 8  
ffmpeg\_video\_codecs (ffmpeg\_codecs), 8  
  
get\_audio\_codec (set\_audio\_codec), 9  
get\_video\_codec (set\_audio\_codec), 9  
  
have\_ffmpeg\_exec (ffmpeg\_exec), 8  
  
ioslides\_presentation, 4  
  
readWave, 6  
rmarkdown, 3, 4  
  
set\_audio\_codec, 9  
set\_video\_codec (set\_audio\_codec), 9  
shQuote, 9  
synthesize, 4  
  
tts, 4, 5, 7  
tts\_voices, 4, 5, 7  
  
video\_codec\_encode (set\_audio\_codec), 9  
  
Wave, 6  
webshot, 4