

Package ‘gifski’

May 2, 2021

Type Package

Title Highest Quality GIF Encoder

Version 1.4.3-1

Description Multi-threaded GIF encoder written in Rust: <<https://gif.ski/>>. Converts images to GIF animations using pngquant's efficient cross-frame palettes and temporal dithering with thousands of colors per frame.

License MIT + file LICENSE

URL <https://gif.ski/> (upstream), <https://github.com/r-rust/gifski> (devel)

BugReports <https://github.com/r-rust/gifski/issues>

SystemRequirements Cargo (rustc package manager)

Encoding UTF-8

RoxygenNote 7.1.1

Suggests ggplot2, gapminder

Language en-US

NeedsCompilation yes

Author Jeroen Ooms [aut, cre] (<<https://orcid.org/0000-0002-4035-0289>>), Kornel Lesiński [cph] (Gifski Rust library)

Maintainer Jeroen Ooms <jeroen@berkeley.edu>

Repository CRAN

Date/Publication 2021-05-02 04:30:02 UTC

R topics documented:

gifski 2

Index 4

`gifski`*Gifski*

Description

Gifski converts image frames to high quality GIF animations. Either provide input png files, or automatically render animated graphics from the R graphics device.

Usage

```
gifski(  
  png_files,  
  gif_file = "animation.gif",  
  width = 800,  
  height = 600,  
  delay = 1,  
  loop = TRUE,  
  progress = TRUE  
)  
  
save_gif(  
  expr,  
  gif_file = "animation.gif",  
  width = 800,  
  height = 600,  
  delay = 1,  
  loop = TRUE,  
  progress = TRUE,  
  ...  
)
```

Arguments

<code>png_files</code>	vector of png files
<code>gif_file</code>	output gif file
<code>width</code>	gif width in pixels
<code>height</code>	gif height in pixel
<code>delay</code>	time to show each image in seconds
<code>loop</code>	if the gif should be repeated. Set to FALSE to only play once, or a number to indicate how many times to repeat after the first.
<code>progress</code>	print some verbose status output
<code>expr</code>	an R expression that creates graphics
<code>...</code>	other graphical parameters passed to png

Examples

```
# Manually convert png files to gif
png_path <- file.path(tempdir(), "frame%03d.png")
png(png_path)
par(ask = FALSE)
for(i in 1:10)
  plot(rnorm(i * 10), main = i)
dev.off()
png_files <- sprintf(png_path, 1:10)
gif_file <- tempfile(fileext = ".gif")
gifski(png_files, gif_file)
unlink(png_files)
utils::browseURL(gif_file)

# Example borrowed from gganimate
library(gapminder)
library(ggplot2)
makeplot <- function(){
  datalist <- split(gapminder, gapminder$year)
  lapply(datalist, function(data){
    p <- ggplot(data, aes(gdpPercap, lifeExp, size = pop, color = continent)) +
      scale_size("population", limits = range(gapminder$pop)) + geom_point() + ylim(20, 90) +
      scale_x_log10(limits = range(gapminder$gdpPercap)) + ggtitle(data$year) + theme_classic()
    print(p)
  })
}

# High Definition images:
gif_file <- file.path(tempdir(), 'gapminder.gif')
save_gif(makeplot(), gif_file, 1280, 720, res = 144)
utils::browseURL(gif_file)
```

Index

`gifski`, [2](#)

`png`, [2](#)

`save_gif(gifski)`, [2](#)