Package 'ragg'

August 28, 2019

Type Package

Title Graphic Devices Based on AGG

Version 0.1.3

Maintainer Thomas Lin Pedersen <thomas.pedersen@rstudio.com>

Description Anti-Grain Geometry (AGG) is a high-quality and high-performance 2D drawing library. The 'ragg' package provides a set of graphic devices based on AGG to use as alternative to the raster devices provided through the 'grDevices' package.

License MIT + file LICENSE

URL https://ragg.r-lib.org

BugReports https://github.com/r-lib/ragg/issues

Encoding UTF-8

RoxygenNote 6.1.1

SystemRequirements C++11, freetype2 libpng

Suggests

Imports systemfonts

NeedsCompilation yes

Author Thomas Lin Pedersen [cre, aut]
(<https://orcid.org/0000-0002-5147-4711>),
Maxim Shemanarev [aut, cph] (Author of AGG),
Tony Juricic [ctb, cph] (Contributor to AGG),
Milan Marusinec [ctb, cph] (Contributor to AGG),
Spencer Garrett [ctb] (Contributor to AGG),
RStudio [cph]

Repository CRAN

Date/Publication 2019-08-28 10:40:05 UTC

R topics documented:

agg_capture 2
agg_png 3
agg_ppm 4
agg_tiff 5

Index **7**

`agg_capture` *Draw to a buffer that can be accessed directly*

Description

Usually the point of using a graphic device is to create a file or show the graphic on the screen. A few times we need the image data for further processing in R, and instead of writing it to a file and then reading it back into R the 'agg_capture()' device lets you get the image data directly from the buffer. In contrast to the other devices this device returns a function, that when called will return the current state of the buffer.

Usage

```
agg_capture(width = 480, height = 480, units = "px",
  pointsize = 12, background = "white", res = 72)
```

Arguments

`width`	The dimensions of the device
`height`	The dimensions of the device
`units`	The unit 'width' and 'height' is measured in, in either pixels (''px''), inches (''in''), millimeters (''mm''), or centimeter (''cm'').
`pointsize`	The default pointsize of the device in pt
`background`	The background colour of the device
`res`	The resolution of the device. This setting will govern how device dimensions given in inches, centimeters, or millimeters will be converted to pixels. Further, it will be used to scale text sizes and linewidths

Value

A function that when called returns the current state of the buffer. The return value of the function depends on the 'native' argument. If 'FALSE' (default) the return value is a 'matrix' of colour values and if 'TRUE' the return value is a 'nativeRaster' object.

Examples

```
cap <- agg_capture()
plot(1:10, 1:10)

# Get the plot as a matrix
raster <- cap()

# Get the plot as a nativeRaster
raster_n <- cap(native = TRUE)

dev.off()

# Look at the output
plot(as.raster(raster))
```

---

agg_png *Draw to a PNG file*

Description

The PNG (Portable Network Graphic) format is one of the most ubiquitous today, due to its versatiliity and widespread support. It supports transparancy as well as both 8 and 16 bit colour. The device uses default compression and filtering and will not use a colour palette as this is less useful for antialiased data. This means that it might be possible to compress the resulting image even more if size is of concern (though the defaults are often very good). In contrast to [grDevices::png()] the date and time will not be written to the file, meaning that similar plot code will produce identical files (a good feature if used with version control). It will, however, write in the dimensions of the image based on the ‘res‘ argument.

Usage

```
agg_png(filename = "Rplot%03d.png", width = 480, height = 480,
  units = "px", pointsize = 12, background = "white", res = 72,
  bitsize = 8)
```

Arguments

filename	The name of the file. Follows the same semantics as the file naming in [grDevices::png()], meaning that you can provide a [sprintf()] compliant string format to name multiple plots (such as the default value)
width	The dimensions of the device
height	The dimensions of the device
units	The unit ‘width‘ and ‘height‘ is measured in, in either pixels (‘’px’‘), inches (‘’in’‘), millimeters (‘’mm’‘), or centimeter (‘’cm’‘).
pointsize	The default pointsize of the device in pt

background	The background colour of the device
res	The resolution of the device. This setting will govern how device dimensions given in inches, centimeters, or millimeters will be converted to pixels. Further, it will be used to scale text sizes and linewidths
bitsize	Should the device record colour as 8 or 16bit

Examples

```
file <- tempfile(fileext = '.png')
agg_png(file)
plot(sin, -pi, 2*pi)
dev.off()
```

---

agg_ppm *Draw to a PPM file*

Description

The PPM (Portable Pixel Map) format defines one of the simplest storage formats available for image data. It is basically a raw 8bit RGB stream with a few bytes of information in the start. It goes without saying, that this file format is horribly inefficient and should only be used if you want to play around with a simple file format, or need a file-based image stream.

Usage

```
agg_ppm(filename = "Rplot%03d.ppm", width = 480, height = 480,
  units = "px", pointsize = 12, background = "white", res = 72)
```

Arguments

filename	The name of the file. Follows the same semantics as the file naming in [grDevices::png()], meaning that you can provide a [sprintf()] compliant string format to name multiple plots (such as the default value)
width, height	The dimensions of the device
units	The unit ‘width‘ and ‘height‘ is measured in, in either pixels (‘’px’‘), inches (‘’in’‘), millimeters (‘’mm’‘), or centimeter (‘’cm’‘).
pointsize	The default pointsize of the device in pt
background	The background colour of the device
res	The resolution of the device. This setting will govern how device dimensions given in inches, centimeters, or millimeters will be converted to pixels. Further, it will be used to scale text sizes and linewidths

Examples

```
file <- tempfile(fileext = '.ppm')
agg_ppm(file)
plot(sin, -pi, 2*pi)
dev.off()
```

`agg_tiff` *Draw to a TIFF file*

Description

The TIFF (Tagged Image File Format) format is a very versatile raster image storage format that supports 8 and 16bit colour mode, true transparency, as well as a range of other features not relevant to drawing from R (e.g. support for different colour spaces). The storage mode of the image data is not fixed and different compression modes are possible, in contrast to PNGs one-approach-fits-all. The default (uncompressed) will result in much larger files than PNG, and in general PNG is a better format for many of the graphic types produced in R. Still, TIFF has its purposes and sometimes this file format is explicetly requested.

Usage

```
agg_tiff(filename = "Rplot%03d.tiff", width = 480, height = 480,
  units = "px", pointsize = 12, background = "white", res = 72,
  compression = "none", bitsize = 8)
```

Arguments

`filename`	The name of the file. Follows the same semantics as the file naming in [grDevices::png()], meaning that you can provide a [sprintf()] compliant string format to name multiple plots (such as the default value)
`width`	The dimensions of the device
`height`	The dimensions of the device
`units`	The unit ‘width‘ and ‘height‘ is measured in, in either pixels (‘’px’‘), inches (‘’in’‘), millimeters (‘’mm’‘), or centimeter (‘’cm’‘).
`pointsize`	The default pointsize of the device in pt
`background`	The background colour of the device
`res`	The resolution of the device. This setting will govern how device dimensions given in inches, centimeters, or millimeters will be converted to pixels. Further, it will be used to scale text sizes and linewidths
`compression`	The compression type to use for the image data. The standard options from the [grDevices::tiff()] function are available under the same name.
`bitsize`	Should the device record colour as 8 or 16bit

Transparency

TIFF have support for true transparency, meaning that the pixel colour is stored in pre-multiplied form. This is in contrast to pixels being stored in plain format, where the alpha values more function as a mask. The utility of this is not always that important, but it is one of the benefits of TIFF over PNG so it should be noted.

Note

‘’jpeg’‘ compression is only available if ragg is compiled with a version of ‘libtiff‘ where jpeg support has been turned on.

Examples

```
file <- tempfile(fileext = '.tiff')
# Use jpeg compression
agg_tiff(file, compression = 'lzw+p')
plot(sin, -pi, 2*pi)
dev.off()
```

Index

agg_capture, 2
agg_png, 3
agg_ppm, 4
agg_tiff, 5