Package 'terminalgraphics'

August 28, 2025

Type Package

Title Graphical Output in Terminals

Version 0.1.1

Description Defines a graphics device and functions for graphical output in terminal emulators that support graphical output. Currently terminals that support the Terminal Graphics Protocol (<https://sw.kovidgoyal.net/kitty/graphics-protocol/>) and terminal supporting Sixel (<https://en.wikipedia.org/wiki/Sixel>) are supported.

BugReports https://codeberg.org/djvanderlaan/terminalgraphics/issues

URL https://codeberg.org/djvanderlaan/terminalgraphics

Depends R (>= 4.1.0)

Imports Rcpp, utils, graphics, grDevices, ragg, base64enc

Suggests magick

LinkingTo Rcpp

License GPL-3

SystemRequirements A terminal emulator supporting the Terminal Graphics Protocol (<https://sw.kovidgoyal.net/kitty/graphics-protocol/>) or Sixel.

LazyLoad yes

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation yes

Author Jan van der Laan [aut, cre] (ORCID: <https://orcid.org/0000-0002-0693-1514>)

Maintainer Jan van der Laan <r@eoos.dds.nl>

Repository CRAN

Date/Publication 2025-08-28 07:40:02 UTC

Contents

`has_sixel_support` *Determine if terminal supports Sixel*

Description

Determine if terminal supports Sixel

Usage

```
has_sixel_support(warn = FALSE, throw = FALSE)
```

Arguments

`warn`	show warnings when the protocol is not supported.
`throw`	throw an error when we are running in a kitty terminal.

Value

Returns `TRUE` if the current terminal supports the Sixel and `FALSE` otherwise. Will also return `FALSE` when not running in a terminal.

Examples

```
if (has_sixel_support()) {
  cat("Yeeh, your terminal supports the sixel!")
}
```

has_tgp_support *Determine if terminal supports Terminal Graphics Protocol*

Description

Determine if terminal supports Terminal Graphics Protocol

Usage

```
has_tgp_support(warn = FALSE, throw = FALSE)
```

Arguments

warn	show warnings when the protocol is not supported.
throw	throw an error when we are running in a kitty terminal.

Value

Returns TRUE if the current terminal supports the Terminal Graphics Protocol and FALSE otherwise. Will also return FALSE when not running in a terminal.

Examples

```
if (has_tgp_support()) {
  cat("Yeeh, your terminal supports the terminal graphics protocol!")
}
```

is_kitty *Determine if we are running in a kitty terminal*

Description

Determine if we are running in a kitty terminal

Usage

```
is_kitty()
```

Value

Returns TRUE if R is running in a kitty terminal and FALSE otherwise.

Examples

```
if (is_kitty()) {
  cat("Yeeh, you are running kitty!")
}
```

kitty_colors *Get the colors used in the kitty terminal*

Description

Get the colors used in the kitty terminal

Usage

```
kitty_colors()

kitty_background()

kitty_foreground()

kitty_palette()
```

Details

To get the background and foreground colors, `kitten query-terminal` is called. To get all colors and the palette `kitty @get-colors` is called using `system`. However, for the last to work `allow_remote_control` needs to be set to `true` in the config file for kitty.

Value

`kitty_colors` returns a data.frame with the colors from the theme used by kitty. `kitty_background` returns the background color (character vector with the hex-code). `kitty_foreground` returns the foreground color. `kitty_palette` returns a vector with the 9 main accent colors from the theme.

See Also

`term_background`, `term_foreground`, `term_palette` for functions that try to return the colors used in any terminal. When running in kitty, these will call `kitty_background` etc.

Examples

```
if (is_kitty()) {
  cat("The background color is '", kitty_background(), "'.")
}
```

png2sixel — *Dump a PNG image to the terminal in Sixel format*

Description

Dump a PNG image to the terminal in Sixel format

Usage

```
png2sixel(filename)
```

Arguments

filename	filename of the PNG image

Value

Called for it's side effect of writing the image to the terminal (standard out). Returns NULL invisibly.

png2terminal — *Dump a PNG image to the terminal*

Description

Dump a PNG image to the terminal

Usage

```
png2terminal(filename, method = c("auto", "tgp", "sixel"))
```

Arguments

filename	filename of the PNG image
method	method with which the graphical output is written to the terminal. In case of 'auto', it is first checked if the Terminal Graphics Protocol is supported and if so, this is used. Otherwise, Sixel is used.

Value

Called for it's side effect of writing the image to the terminal (standard out). Returns NULL invisibly.

See Also

See png2tgp for output in Terminal Graphics Protocol. See png2sixel for output in Sixel format.

png2tgp — *Dump a PNG image to the terminal using the Terminal Graphics Protocol*

Description

Dump a PNG image to the terminal using the Terminal Graphics Protocol

Usage

```
png2tgp(filename)
```

Arguments

`filename`	filename of the PNG image

Details

The Terminal Graphics Protocol is not supported by many Terminal Emulators. The most notable terminal emulator supporting the protocol is Kitty.

Value

Called for it's side effect of writing the image to the terminal (standard out). Returns `NULL` invisibly.

raster2sixel — *Dump an image raster to the terminal in Sixel format*

Description

Dump an image raster to the terminal in Sixel format

Usage

```
raster2sixel(raster)
```

Arguments

`raster`	the image 'raster' e.g. the output of `as.raster`.

Details

Sixel is a bitmap format supported by some terminals. See, for example, https://en.wikipedia.org/wiki/Sixel.

Value

Called for it's side effect of writing the image to the terminal (standard out). Returns `NULL` invisibly.

See Also

See `raster2tgp` for output using Terminal Graphics Protocol.

`raster2terminal` *Dump an image raster to the terminal*

Description

Dump an image raster to the terminal

Usage

```
raster2terminal(raster, method = c("auto", "tgp", "sixel"), ...)
```

Arguments

`raster`	the image 'raster' e.g. the output of `as.raster`.
`method`	method with which the graphical output is written to the terminal. In case of 'auto', it is first checked if the Terminal Graphics Protocol is supported and if so, this is used. Otherwise, Sixel is used.
`...`	passed on to the underlying method such as `raster2tgp` and `raster2sixel`.

Value

Called for it's side effect of writing the image to the terminal (standard out). Returns `NULL` invisibly.

See Also

See `raster2tgp` for output in Terminal Graphics Protocol. See `raster2sixel` for output in Sixel format.

raster2tgp *Dump an image raster to the terminal using the Terminal Graphics Protocol*

Description

Dump an image raster to the terminal using the Terminal Graphics Protocol

Usage

```
raster2tgp(raster, compress = TRUE)
```

Arguments

raster	the image 'raster' e.g. the output of as.raster.
compress	compress the data before sending to the terminal.

Details

The Terminal Graphics Protocol is not supported by many Terminal Emulators. The most notable terminal emulator supporting the protocol is Kitty.

Value

Called for it's side effect of writing the image to the terminal (standard out). Returns NULL invisibly.

See Also

See raster2sixel for output in Sixel format. See png2terminal for writing a PNG image to the terminal.

sixel *Terminal Graphics Protocol Device*

Description

Terminal Graphics Protocol Device

Usage

```
sixel(
 width = getOption("term_width", max(480, min(1200, term_width(), term_height()/0.8))),
  height = getOption("term_height", 0.8 * width),
  units = "px",
  res = getOption("term_res", NA),
  ...,
  term_col = getOption("term_col", FALSE),
  term_bg = term_col,
  term_fg = term_col
)
```

Arguments

width	The width of the image. Passed on to agg_capture.
height	The height of the image. Passed on to agg_capture.
units	The units in which 'height' and 'width' are given. Passed on to agg_capture.
res	The resulution of the image. Passed on to agg_capture.
...	passed on to the underlying agg_capture device.
term_col	Logical value indicating that the foreground and background colors used in the plot should be set to that of the terminal.
term_bg	Logical value indicating that the background color used in the plot should be set to that of the terminal.
term_fg	Logical value indicating that the foreground color used in the plot should be set to that of the terminal.

Details

The function tries to detect whether the terminal supports Sixel. If not, it will issue a warning but still output the image to the terminal. Terminals that to not support Sixel will ignore the output. The warning is shown only once. Please file an issue when the terminal does support Sixel while has_sixel_support returns FALSE. The warning can be disabled using options(warned_sixel_support = TRUE).

Value

sixel is called for its side effect of opening a graphics device. Invisibly returns an list with two functions: plot will plot the current contents of the device in the terminal and update will plot the current contents of the device in the terminal if the contents have changed since the last plot.

term_replot will redraw the content of the device in the terminal. In principle term_replot is called automatically when the contents of the device changed. This function can be used to force plotting.

When term_bg = TRUE the background color of the graphics device ('bg') will be set using par. When term_fg = TRUE the foreground color ('fg', 'col', 'col.axis', 'col.lab', 'col.main', and 'sub') will be set using par.

Examples

```
if (has_sixel_support()) {
  sixel()
  plot(rnorm(100))
  abline(h = 0, lwd = 2, col = term_palette()[1])
}
```

`term_background` *Get the colors used in the terminal*

Description

Get the colors used in the terminal

Usage

```
term_background()

term_foreground()

term_palette()
```

Details

Getting the color palette used in the terminal is terminal specific. Currently only the kitty terminal is supported. For other terminals default colors are returned.

Value

`term_background` and `term_foreground` will return a length 1 vector with a color. `term_palette` will return a vector of colors.

See Also

`kitty_colors` for the functions returning the specific colors used in the kitty terminal.

Examples

```
term_background()
```

term_dim *Get the dimensions of the terminal window in pixels*

Description

Get the dimensions of the terminal window in pixels

Usage

```
term_dim()
```

Value

An integer vector with the width and height of the terminal in pixels (`x_pixels` and `y_pixels`) and the number of text columns and rows in the terminal window (`columns` and `rows`).

These values can be zero when the terminal does not support querying the size. Some terminals only support querying the number of columns and rows. Under windows the return value will always be zero as querying the size depends on POSIX support.

See Also

`term_width` and `term_height` for only obtaining the width and height in pixels respectively.

term_height *Get the height of the terminal window in pixels*

Description

Get the height of the terminal window in pixels

Usage

```
term_height()
```

Value

An integer with the number of pixels the terminal is high. This value can be zero when the terminal does not support querying the size.

See Also

`term_width` for the terminal width and `term_dim` for all dimensions including the dimensions in rows and columns.

term_replot *Replot the current device in the terminal.*

Description

Replot the current device in the terminal.

Usage

```
term_replot()
```

Value

Called for outputting the current contents of a `sixel` or `tgp` device into the terminal.

term_width *Get the with of the terminal window in pixels*

Description

Get the with of the terminal window in pixels

Usage

```
term_width()
```

Value

An integer with the number of pixels the terminal is wide This value can be zero when the terminal does not support querying the size. Under windows the return value will always be zero as querying the size depends on POSIX support.

See Also

`term_height` for the terminal height and `term_dim` for all dimensions including the dimensions in rows and columns.

tgp *Terminal Graphics Protocol Device*

Description

Terminal Graphics Protocol Device

Usage

```
tgp(
 width = getOption("term_width", max(480, min(1200, term_width(), term_height()/0.8))),
  height = getOption("term_height", 0.8 * width),
  units = "px",
  res = getOption("term_res", NA),
  ...,
  term_col = getOption("term_col", FALSE),
  term_bg = term_col,
  term_fg = term_col
)
```

Arguments

width	The width of the image. Passed on to agg_capture.
height	The height of the image. Passed on to agg_capture.
units	The units in which 'height' and 'width' are given. Passed on to agg_capture.
res	The resolution of the image. Passed on to agg_capture.
...	passed on to the underlying agg_capture device.
term_col	Logical value indicating that the foreground and background colors used in the plot should be set to that of the terminal.
term_bg	Logical value indicating that the background color used in the plot should be set to that of the terminal.
term_fg	Logical value indicating that the foreground color used in the plot should be set to that of the terminal.

Details

The function tries to detect whether the terminal supports Terminal Graphics Protocol. If not, it will issue a warning but still output the image to the terminal. Terminals that to not support Terminal Graphics Protocol will ignore the output. The warning is shown only once. Please file an issue when the terminal does support Terminal Graphics Protocol while has_tgp_support returns FALSE. The warning can be disabled using options(warned_tgp_support = TRUE).

Value

`tgp` is called for its side effect of opening a graphics device. Invisibly returns an list with two functions: `plot` will plot the current contents of the device in the terminal and `update` will plot the current contents of the device in the terminal if the contents have changed since the last plot.

`term_replot` will redraw the content of the device in the terminal. In principle `term_replot` is called automatically when the contents of the device changed. This function can be used to force plotting.

When `term_bg = TRUE` the background color of the graphics device ('bg') will be set using `par`. When `term_fg = TRUE` the foreground color ('fg', 'col', 'col.axis', 'col.lab', 'col.main', and 'sub') will be set using `par`.

Examples

```
if (has_tgp_support()) {
  # Open device
  tgp()
  plot(rnorm(100))
  abline(h = 0, lwd = 2, col = term_palette()[1])
}
```

Index