Package ‘rphylopic’

November 18, 2025

Title Get Silhouettes of Organisms from PhyloPic

Description Work with the PhyloPic Web Service (<http://api-docs.phylopic.org/v2/>)
to fetch silhouette images of organisms. Includes functions for adding
silhouettes to both base R plots and ggplot2 plots.

Version 1.6.0

License GPL (>= 3)

Language en-US

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 4.0)

Imports ape, ggplot2, jsonlite, grid, graphics, png, grImport2 (>=
0.3.3), rsvg (>= 2.6.0), httr, curl, methods, lifecycle,
pbapply, knitr, scales

Suggests testthat (>= 3.0.0), vdiffr (>= 1.0.0), rmarkdown, covr,
phytools, palaeoverse, ggtree, deeptime, palmerpenguins, maps,
sf

Config/testthat/edition 3

URL https://rphylopic.palaeoverse.org,
https://github.com/palaeoverse/rphylopic,
https://palaeoverse.org

BugReports https://github.com/palaeoverse/rphylopic/issues

VignetteBuilder knitr

NeedsCompilation no

Author William Gearty [aut, cre] (ORCID:
<https://orcid.org/0000-0003-0076-3262>),
Lewis A. Jones [aut] (ORCID: <https://orcid.org/0000-0003-3902-8986>),
Scott Chamberlain [ctb] (ORCID:
<https://orcid.org/0000-0003-1444-9135>),
Martin R. Smith [ctb] (ORCID: <https://orcid.org/0000-0001-5660-1727>),
David L. Miller [ctb] (ORCID: <https://orcid.org/0000-0002-9640-6755>)

Maintainer William Gearty <`willgearty@gmail.com`>

Repository CRAN

Date/Publication 2025-11-18 08:30:03 UTC

Contents

`add_phylopic` *Annotate a ggplot plot with PhyloPics*

Description

Specify existing images, taxonomic names, or PhyloPic uuids to add PhyloPic silhouettes as a separate layer to an existing ggplot plot.

Usage

```
add_phylopic(
  img = NULL,
  name = NULL,
  uuid = NULL,
  filter = NULL,
  x,
  y,
  ysize = deprecated(),
  height = NA,
```

```
    width = NA,
    alpha = 1,
    color = NA,
    fill = "black",
    horizontal = FALSE,
    vertical = FALSE,
    angle = 0,
    hjust = 0.5,
    vjust = 0.5,
    remove_background = TRUE,
    verbose = FALSE
  )
```

Arguments

`img`	A Picture or png array object, e.g., from using `get_phylopic()`.
`name`	`character`. A taxonomic name to be passed to `get_uuid()`.
`uuid`	`character`. A valid uuid for a PhyloPic silhouette (such as that returned by `get_uuid()` or `pick_phylopic()`).
`filter`	`character`. Filter by usage license if `name` is defined. Use "by" to limit results to images which do not require attribution, "nc" for images which allows commercial usage, and "sa" for images without a ShareAlike clause. The user can also combine these filters as a vector.
`x`	`numeric`. x value of the silhouette center.
`y`	`numeric`. y value of the silhouette center.
`ysize`	**[Deprecated]** use the `height` or `width` argument instead.
`height`	`numeric`. Height of the silhouette in coordinate space. If "NA", the default, and `width` is "NA", the silhouette will be as large as fits in the plot area. If "NA" and `width` is specified, the height is determined by the aspect ratio of the original image. One or both of `height` and `width` must be "NA".
`width`	`numeric`. Width of the silhouette in coordinate space. If "NA", the default, and `height` is "NA", the silhouette will be as large as fits in the plot area. If "NA", the default, and `height` is specified, the width is determined by the aspect ratio of the original image. One or both of `height` and `width` must be "NA".
`alpha`	`numeric`. A value between 0 and 1, specifying the opacity of the silhouette (0 is fully transparent, 1 is fully opaque).
`color`	`character`. Color of silhouette outline. If "original" or NA is specified, the original color of the silhouette outline will be used (usually the same as "transparent"). To remove the outline, you can set this to "transparent".
`fill`	`character`. Color of silhouette. If "original" is specified, the original color of the silhouette will be used (usually the same as "black"). If `color` is specified and `fill` is NA, `color` will be used as the fill color (for backwards compatibility). To remove the fill, you can set this to "transparent".
`horizontal`	`logical`. Should the silhouette be flipped horizontally?
`vertical`	`logical`. Should the silhouette be flipped vertically?

`angle`	`numeric`. The number of degrees to rotate the silhouette clockwise. The default is no rotation.
`hjust`	`numeric`. A numeric vector between 0 and 1 specifying horizontal justification (left = 0, center = 0.5, right = 1). Note that, due to the enforcement of the silhouette's aspect ratio, there may be unexpected behavior due to interactions between the aspect ratio of the plot and the aspect ratio of the silhouette.
`vjust`	`numeric`. A numeric vector between 0 and 1 specifying vertical justification (top = 1, middle = 0.5, bottom = 0). Note that, due to the enforcement of the silhouette's aspect ratio, there may be unexpected behavior due to interactions between the aspect ratio of the plot and the aspect ratio of the silhouette.
`remove_background`	`logical`. Should any white background be removed from the silhouette(s)? See `recolor_phylopic()` for details.
`verbose`	`logical`. Should the attribution information for the used silhouette(s) be printed to the console (see `get_attribution()`)?

Details

One (and only one) of `img`, `name`, or `uuid` must be specified. Use parameters `x`, `y`, and `ysize` to place the silhouette at a specified position on the plot. The aspect ratio of the silhouette will always be maintained.

`x` and/or `y` may be vectors of numeric values if multiple silhouettes should be plotted at once. In this case, any other arguments (except for `remove_background`) may also be vectors of values, which will be recycled as necessary.

When specifying a horizontal and/or vertical flip **and** a rotation, the flip(s) will always occur first. If you would like to customize this behavior, you can flip and/or rotate the image within your own workflow using `flip_phylopic()` and `rotate_phylopic()`.

Note that png array objects can only be rotated by multiples of 90 degrees. Also, outline colors do not currently work for png array objects.

Examples

```
## Not run:
# Put a silhouette behind a plot based on a taxonomic name
library(ggplot2)
ggplot(iris) +
  add_phylopic(x = 6.1, y = 3.2, name = "Iris", alpha = 0.2) +
  geom_point(aes(x = Sepal.Length, y = Sepal.Width))

# Put a silhouette in several places based on UUID
posx <- runif(10, 0, 10)
posy <- runif(10, 0, 10)
heights <- runif(10, 0.4, 2)
angle <- runif(10, 0, 360)
hor <- sample(c(TRUE, FALSE), 10, TRUE)
ver <- sample(c(TRUE, FALSE), 10, TRUE)
fills <- sample(c("black", "darkorange", "grey42", "white"), 10,
  replace = TRUE)
```

```
alpha <- runif(10, 0.3, 1)

p <- ggplot(data.frame(cat.x = posx, cat.y = posy), aes(cat.x, cat.y)) +
  geom_blank() +
  add_phylopic(uuid = "23cd6aa4-9587-4a2e-8e26-de42885004c9",
               x = posx, y = posy, height = heights,
               fill = fills, alpha = alpha, angle = angle,
               horizontal = hor, vertical = ver)
p + ggtitle("R Cat Herd!!")

## End(Not run)
```

`add_phylopic_base` *Add PhyloPics to a base R plot*

Description

Specify existing images, taxonomic names, or PhyloPic uuids to add PhyloPic silhouettes on top of an existing base R plot (like `points()`).

Usage

```
add_phylopic_base(
  img = NULL,
  name = NULL,
  uuid = NULL,
  filter = NULL,
  x = NULL,
  y = NULL,
  ysize = deprecated(),
  height = NULL,
  width = NULL,
  alpha = 1,
  color = NA,
  fill = "black",
  horizontal = FALSE,
  vertical = FALSE,
  angle = 0,
  hjust = 0.5,
  vjust = 0.5,
  remove_background = TRUE,
  verbose = FALSE
)
```

Arguments

`img`	A Picture or png array object, e.g., from using `get_phylopic()`.
`name`	`character`. A taxonomic name to be passed to `get_uuid()`.

`uuid`	`character`. A valid uuid for a PhyloPic silhouette (such as that returned by `get_uuid()` or `pick_phylopic()`).
`filter`	`character`. Filter by usage license if `name` is defined. Use "by" to limit results to images which do not require attribution, "nc" for images which allows commercial usage, and "sa" for images without a ShareAlike clause. The user can also combine these filters as a vector.
`x`	`numeric`. x value of the silhouette center. If "NULL", the default, the mean value of the x-axis is used.
`y`	`numeric`. y value of the silhouette center. If "NULL", the default, the mean value of the y-axis is used.
`ysize`	**[Deprecated]** use the `height` or `width` argument instead.
`height`	`numeric`. Height of the silhouette in coordinate space. If "NULL", the default, and `width` is also "NULL", the silhouette will be as large as fits in the plot area. If "NULL" and `width` is specified, the height is determined by the aspect ratio of the original image. One or both of `height` and `width` must be "NULL".
`width`	`numeric`. Width of the silhouette in coordinate space. If "NULL", the default, and `height` is also "NULL", the silhouette will be as large as fits in the plot area. If "NULL" and `height` is specified, the width is determined by the aspect ratio of the original image. One or both of `height` and `width` must be "NULL".
`alpha`	`numeric`. A value between 0 and 1, specifying the opacity of the silhouette (0 is fully transparent, 1 is fully opaque).
`color`	`character`. Color of silhouette outline. If "original" or NA is specified, the original color of the silhouette outline will be used (usually the same as "transparent"). To remove the outline, you can set this to "transparent".
`fill`	`character`. Color of silhouette. If "original" is specified, the original color of the silhouette will be used (usually the same as "black"). If `color` is specified and `fill` is NA, `color` will be used as the fill color (for backwards compatibility). To remove the fill, you can set this to "transparent".
`horizontal`	`logical`. Should the silhouette be flipped horizontally?
`vertical`	`logical`. Should the silhouette be flipped vertically?
`angle`	`numeric`. The number of degrees to rotate the silhouette clockwise. The default is no rotation.
`hjust`	`numeric`. A numeric vector between 0 and 1 specifying horizontal justification (left = 0, center = 0.5, right = 1).
`vjust`	`numeric`. A numeric vector between 0 and 1 specifying vertical justification (top = 1, middle = 0.5, bottom = 0).
`remove_background`	`logical`. Should any white background be removed from the silhouette(s)? See `recolor_phylopic()` for details.
`verbose`	`logical`. Should the attribution information for the used silhouette(s) be printed to the console (see `get_attribution()`)?

Details

One (and only one) of `img`, `name`, or `uuid` must be specified. Use parameters `x`, `y`, `hjust`, and `vjust` to place the silhouette at a specified position on the plot. If `height` and `width` are both unspecified, then the silhouette will be plotted to the full height and/or width of the plot. The aspect ratio of `Picture` objects will always be maintained (even when a plot is resized). However, if the plot is resized after plotting a silhouette, the absolute size and/or position of the silhouette may change.

Any argument (except for `remove_background`) may be a vector of values if multiple silhouettes should be plotted. In this case, all other arguments may also be vectors of values, which will be recycled as necessary to the length of the longest vector argument.

When specifying a horizontal and/or vertical flip **and** a rotation, the flip(s) will always occur first. If you would like to customize this behavior, you can flip and/or rotate the image within your own workflow using `flip_phylopic()` and `rotate_phylopic()`.

Note that png array objects can only be rotated by multiples of 90 degrees. Also, outline colors do not currently work for png array objects.

Examples

```
## Not run:
# single image
plot(1, 1, type = "n", main = "A cat")
add_phylopic_base(uuid = "23cd6aa4-9587-4a2e-8e26-de42885004c9",
                  x = 1, y = 1, height = 0.4)

# lots of images using a uuid
posx <- runif(10, 0, 1)
posy <- runif(10, 0, 1)
size <- runif(10, 0.1, 0.3)
angle <- runif(10, 0, 360)
hor <- sample(c(TRUE, FALSE), 10, TRUE)
ver <- sample(c(TRUE, FALSE), 10, TRUE)
fills <- sample(c("black", "darkorange", "grey42", "white"), 10,
               replace = TRUE)

plot(posx, posy, type = "n", main = "A cat herd")
add_phylopic_base(uuid = "23cd6aa4-9587-4a2e-8e26-de42885004c9",
                  x = posx, y = posy, height = size,
                  fill = fills, angle = angle,
                  horizontal = hor, vertical = ver)

# Example using a cat background
cat <- get_phylopic("23cd6aa4-9587-4a2e-8e26-de42885004c9")
# setup plot area
plot(posx, posy, type = "n", main = "A cat herd, on top of a cat",
     xlim = c(0, 1), ylim = c(0, 1))
# plot background cat
add_phylopic_base(img = cat, alpha = 0.2)
# overlay smaller cats
add_phylopic_base(img = cat, x = posx, y = posy, height = size, alpha = 0.8)

## End(Not run)
```

`add_phylopic_legend` *Add a PhyloPic legend for a base R plot*

Description

Specify existing images, taxonomic names, or PhyloPic uuids to add PhyloPic silhouettes as a legend to an existing base R plot (like `legend()`).

Usage

```
add_phylopic_legend(
  x,
  y = NULL,
  legend,
  img = NULL,
  name = NULL,
  uuid = NULL,
  ysize = deprecated(),
  height = NULL,
  color = NA,
  fill = "black",
  ...
)
```

Arguments

`x`	`numeric` or `character`. Either the x coordinate to be used to position the legend or a keyword accepted by `legend()` such as "topleft", "topright", "bottomleft", and "bottomright".
`y`	`numeric`. The y coordinate to be used to position the legend. Can be `NULL` (default) if using keywords in `x`.
`legend`	`character`. A character vector of the labels to appear in the legend.
`img`	A Picture or png array object, e.g., from using `get_phylopic()`.
`name`	`character`. A taxonomic name to be passed to `get_uuid()`.
`uuid`	`character`. A valid uuid for a PhyloPic silhouette (such as that returned by `get_uuid()` or `pick_phylopic()`).
`ysize`	**[Deprecated]** use the `height` argument instead.
`height`	`numeric`. Height of the legend silhouette(s). The width is determined by the aspect ratio of the original image.
`color`	`character`. Color of silhouette outline. If "original" or NA is specified, the original color of the silhouette outline will be used (usually the same as "transparent"). To remove the outline, you can set this to "transparent".

`fill`	`character`. Color of silhouette. If "original" is specified, the original color of the silhouette will be used (usually the same as "black"). If `color` is specified and `fill` is NA, `color` will be used as the fill color (for backwards compatibility). To remove the fill, you can set this to "transparent".
`...`	Additional arguments passed to `legend()`.

Details

This function can be used to add PhyloPic silhouettes as a legend to a base R plot. Arguments available in `legend()` can be used and passed via `...`. Note that not all arguments in `legend()` are compatible with `add_phylopic_legend()`. These include arguments for modifying lines (e.g. `lty`, `lwd`, `seg.len`), points (e.g. `pch`, `pt.lwd`), and shading (e.g. `angle` and `density`). This is due to `add_phylopic_legend()` using `add_phylopic_base()` to generate the legend symbols. However, arguments for adjusting the legend appearance such as text (e.g. `cex`), legend box (e.g. `bg`), and color (e.g. `border`) are compatible.

Examples

```
# Get UUIDs
uuids <- get_uuid(name = "Canis lupus", n = 2)
# Generate empty plot
plot(0:10, 0:10, type = "n", main = "Wolves")
# Add data points
add_phylopic_base(uuid = uuids,
  color = "black", fill = c("blue", "green"),
  x = c(2.5, 7.5), y = c(2.5, 7.5), height = 2)
# Add legend
add_phylopic_legend(uuid = uuids,
  height = 0.5, color = "black", fill = c("blue", "green"),
  x = "bottomright", legend = c("Wolf 1", "Wolf 2"),
  bg = "lightgrey")
```

`add_phylopic_tree` *Add PhyloPics to a phylogenetic tree plotted with base R*

Description

Specify existing images, taxonomic names, or PhyloPic uuids to add PhyloPic silhouettes alongside the associated leaves of a phylogenetic tree that has been plotted in the active graphics device using the base R graphics functions. The current functionality assumes that the tree is not in a circular configuration and has a "rightwards" direction.

Usage

```
add_phylopic_tree(
  tree,
  tip = names(img) %||% names(uuid) %||% names(name) %||% name,
  img = NULL,
```

```
    name = if (is.null(img) && is.null(uuid)) tip else NULL,
    uuid = NULL,
    align = "tip",
    width,
    padding = NULL,
    relWidth = 0.06,
    relPadding = if (align == "tip") 1/200 else -1/200,
    hjust = if (align == "tip") 0 else 1,
    ...
)
```

Arguments

`tree`	`phylo`. The phylogenetic tree object on which to add the silhouette.
`tip`	`character`. The tip labels against which to add the silhouettes. If not specified, the names of the `img`, `uuid` or `name` vector are used.
`img`	A Picture or png array object, e.g., from using `get_phylopic()`.
`name`	`character`. A taxonomic name to be passed to `get_uuid()`.
`uuid`	`character`. A valid uuid for a PhyloPic silhouette (such as that returned by `get_uuid()` or `pick_phylopic()`).
`align`	`character`. Should each silhouette be aligned to its respective tip (`"tip"`, the default) or to the right-hand side of the plotting area (`"plot"`)? If `"tip"` is specified, the silhouette is placed at the x coordinate of the respective tip, plus any horizontal padding specified by `padding` or `relPadding`. If `"plot"` is specified, the silhouette is placed at the right-hand side of the plotting area, determined by `par("usr")`, plus any horizontal padding specified by `padding` or `relPadding`.
`width`, `relWidth`	`numeric`. The width of each silhouette, in the plot coordinate system (`width`) or relative to the size of the plotting area (`relWidth`). If "NULL" and `height` is specified, the width is determined by the aspect ratio of the original image. One of height and width must be "NULL".
`padding`, `relPadding`	`numeric`. Horizontal padding for each silhouette from its respective x value, in the plot coordinate system (`padding`) or relative to the size of the plotting area (`relPadding`). Negative values offset to the left.
`hjust`	`numeric`. A numeric vector between 0 and 1 specifying horizontal justification (left = 0, center = 0.5, right = 1).
`...`	Further arguments to pass to `add_phylopic_base()`.

Author(s)

Martin R. Smith (martin.smith@durham.ac.uk)

See Also

For trees plotted using **ggtree**, see `geom_phylopic()`.

Examples

```
## Not run:
# Load the ape library to work with phylogenetic trees
library("ape")

# Read a phylogenetic tree
tree <- ape::read.tree(text = "(cat, (dog, mouse));")

# Set a large right margin to accommodate the silhouettes
par(mar = c(1, 1, 1, 10))

# Plot the tree
plot(tree)

# Add a PhyloPic silhouette of a cat to the tree
add_phylopic_tree(
  tree, # Must be the tree that was plotted
  "cat", # Which leaf should the silhouette be plotted against?
  uuid = "23cd6aa4-9587-4a2e-8e26-de42885004c9", # Silhouette to plot
  relWidth = 0.2,
  fill = "brown"
)

## End(Not run)
```

`browse_phylopic` *Browse PhyloPic for a given taxonomic name or uuid*

Description

This function provides a convenient way to browse PhyloPic for a given taxonomic name of uuid.

Usage

```
browse_phylopic(name = NULL, uuid = NULL)
```

Arguments

`name`	`character`. A taxonomic name. Various taxonomic levels are supported (e.g. species, genus, family).
`uuid`	`character`. A PhyloPic image uuid, as acquired by `get_uuid()`.

Details

This function returns a PhyloPic url for an input `name` or `uuid` and opens the user's default web browser at this url. If no `name` or `uuid` is supplied, the base url of PhyloPic images is returned and opened instead.

Value

A `character` vector of a valid PhyloPic url for the specified `name` or `uuid`. If no `name` or `uuid` is supplied, the base url of PhyloPic images is returned.

Examples

```
## Not run:
url <- browse_phylopic(name = "Acropora cervicornis")

## End(Not run)
```

`flip_phylopic` *Flip a PhyloPic silhouette along its horizontal and/or vertical axis*

Description

The picture can be a Picture or png array object, e.g., from using `get_phylopic()`.

Usage

```
flip_phylopic(img, horizontal = TRUE, vertical = FALSE)
```

Arguments

`img`	A Picture or png array object, e.g., from using `get_phylopic()`.
`horizontal`	`logical`. Should the silhouette be flipped horizontally?
`vertical`	`logical`. Should the silhouette be flipped vertically?

Value

A Picture or png array object (matching the type of `img`)

See Also

Other transformations: `recolor_phylopic()`, `rotate_phylopic()`

`geom_phylopic` *Geom for adding PhyloPic silhouettes to a plot*

Description

This geom acts like `ggplot2::geom_point()`, except that the specified silhouettes are used as points. Silhouettes can be specified by their `name`, `uuid`, or image objects (`img`).

Usage

```
geom_phylopic(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  na.rm = FALSE,
  show.legend = FALSE,
  inherit.aes = TRUE,
  remove_background = TRUE,
  verbose = FALSE,
  filter = NULL
)
```

Arguments

`mapping` Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

`data` The data to be displayed in this layer. There are three options:

If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.

A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.

A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).

`stat` The statistical transformation to use on the data for this layer. When using a `geom_*()` function to construct a layer, the `stat` argument can be used the override the default coupling between geoms and stats. The `stat` argument accepts the following:

- A `Stat` ggproto subclass, for example `StatCount`.
- A string naming the stat. To give the stat as a string, strip the function name of the `stat_` prefix. For example, to use `stat_count()`, give the stat as `"count"`.

	• For more information and other ways to specify the stat, see the layer stat documentation.
`position`	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following: • The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`. • For more information and other ways to specify the position, see the layer position documentation.
`...`	Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `...`. Unknown arguments that are not part of the 4 categories below are ignored. • Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data. • When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept. • Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept. • The `key_glyph` argument of `layer()` may also be passed on through `...`. This can be one of the functions described as key glyphs, to change the display of the layer in the legend.
`na.rm`	If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.
`show.legend`	logical. Should this layer be included in the legends? `FALSE`, the default, never includes, `NA` includes if any aesthetics are mapped, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.
`inherit.aes`	If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.
`remove_background`	`logical`. Should any white background be removed from the silhouette(s)? See `recolor_phylopic()` for details.

`verbose`	`logical`. Should the attribution information for the used silhouette(s) be printed to the console (see `get_attribution()`)?
`filter`	`character`. Filter by usage license if using the `name` aesthetic. Use "by" to limit results to images which do not require attribution, "nc" for images which allows commercial usage, and "sa" for images without a ShareAlike clause. The user can also combine these filters as a vector.

Details

One (and only one) of the `img`, `name`, or `uuid` aesthetics must be specified. The `img` aesthetic can be Picture objects or png array objects, e.g., from using `get_phylopic()`. Use the `x` and `y` aesthetics to place the silhouettes at specified positions on the plot. The `height` or `width` aesthetic specifies the height or width, respectively, of the silhouettes in the units of the `y` axis (only one is allowed). The aspect ratio of the silhouettes will always be maintained. The `hjust` and `vjust` aesthetics can be used to manage the justification of the silhouettes with respect to the `x` and `y` coordinates.

The `color` (default: NA), `fill` (default: "black"), and `alpha` (default: 1) aesthetics can be used to change the outline color, fill color, and transparency (outline and fill) of the silhouettes, respectively. If `color` is specified and `fill` is NA `color` will be used as the fill color (for backwards compatibility). If "original" is specified for the `color` aesthetic, the original color of the silhouette outline will be used (usually the same as "transparent"). If "original" is specified for the `fill` aesthetic, the original color of the silhouette body will be used (usually the same as "black"). To remove the fill or outline, you can set `fill` or `color` to "transparent", respectively.

The `horizontal` and `vertical` aesthetics can be used to flip the silhouettes. The `angle` aesthetic can be used to rotate the silhouettes. When specifying a horizontal and/or vertical flip **and** a rotation, the flip(s) will always occur first. If you would like to customize this behavior, you can flip and/or rotate the image within your own workflow using `flip_phylopic()` and `rotate_phylopic()`.

Note that png array objects can only be rotated by multiples of 90 degrees. Also, outline colors do not currently work for png array objects.

Aesthetics

geom_phylopic understands the following aesthetics:

- **x** (required)
- **y** (required)
- **img** *or* **uuid** *or* **name** (one, and only one, required)
- height *or* width (optional, maximum of only one allowed)
- ysize **[Deprecated]** Deprecated in favor of height or width
- color *or* colour
- fill
- alpha
- horizontal
- vertical
- angle
- hjust

- vjust

  Learn more about setting these aesthetics in `add_phylopic()`.

Examples

```
## Not run:
library(ggplot2)
df <- data.frame(x = c(2, 4), y = c(10, 20),
                 name = c("Felis silvestris catus", "Odobenus rosmarus"))
ggplot(df) +
  geom_phylopic(aes(x = x, y = y, name = name),
                fill = "purple", height = 10) +
  facet_wrap(~name) +
  coord_cartesian(xlim = c(1,6), ylim = c(5, 30))

## End(Not run)
```

`get_attribution` *Get PhyloPic attribution data*

Description

This function provides a convenient way to obtain attribution data for PhyloPic images via an image uuid returned by `get_uuid()`.

Usage

```
get_attribution(uuid = NULL, img = NULL, text = FALSE, permalink = FALSE)
```

Arguments

`uuid`	`character`. A vector of valid uuid(s) for PhyloPic silhouette(s) such as that returned by `get_uuid()` or `pick_phylopic()`.
`img`	A Picture or png array object from `get_phylopic()`. A list of these objects can also be supplied. If `img` is supplied, `uuid` is ignored. Defaults to NULL.
`text`	`logical`. Should attribution information be returned as a text paragraph? Defaults to `FALSE`.
`permalink`	`logical`. Should a permalink be created for this collection of `uuid`(s)? Defaults to `FALSE`.

Details

This function returns image `uuid` specific attribution data, including: contributor name, contributor uuid, contributor contact, image uuid, license, and license abbreviation. If `text` is set to `TRUE`, a text paragraph with the contributor name, year of contribution, and license type is printed and image attribution data is returned invisibly (i.e. using `invisible()`. If `permalink` is set to `TRUE`, a permanent link (hosted by PhyloPic) will be generated. This link can be used to view and share details about the image silhouettes, including contributors and licenses.

Value

A `list` of PhyloPic attribution data for an image `uuid` or a text output of relevant attribution information.

Examples

```
## Not run:
# Get valid uuid
uuid <- get_uuid(name = "Acropora cervicornis")
# Get attribution data for uuid
attri <- get_attribution(uuid = uuid)

# Get list of valid uuids
uuids <- get_uuid(name = "Scleractinia", n = 5)
# Get attribution data for uuids
get_attribution(uuid = uuids, text = TRUE)
# Get attribution data for uuids and create permalink
get_attribution(uuid = uuids, text = TRUE, permalink = TRUE)

## End(Not run)
```

`get_phylopic` *Retrieve an image for a given PhyloPic uuid*

Description

This retrieves a PhyloPic silhouette as a vectorized or rasterized object.

Usage

```
get_phylopic(
  uuid = NULL,
  format = "vector",
  source = FALSE,
  height = 512,
  preview = FALSE
)

## S3 method for class 'Picture'
print(x, ...)

## S3 method for class 'phylopic'
print(x, ...)
```

Arguments

`uuid`	`character`. A PhyloPic image uuid.
`format`	`character`. Format of the image. To return a vectorized image, use "vector". To return a rasterized image, use "raster" and specify a desired `height`.
`source`	`logical`. If `FALSE` (the default), the vector file generated by PhyloPic is used to generate the image in the desired format. If `TRUE`, the original source file is used. Note that only PNG and SVG source files are supported for the "raster" `format` and only SVG source files are supported for the "vector" `format`.
`height`	`numeric`. If `format` is "raster", this is the desired height of the raster image in pixels. This is ignored if `format` is "vector".
`preview`	`logical`. If `preview` is `TRUE`, the returned image is plotted. Defaults to `FALSE`.
`x`	A Picture or png array object, e.g., from using `get_phylopic()`.
`...`	Ignored

Details

The `height` argument is ignored if the `format` argument is not set to "raster". If `format` is "raster", the `height` argument specifies the height of the desired raster object. The width of this raster object will be determined by the original aspect ratio of the silhouette. If a pre-rendered raster exists with the desired dimensions, it will be downloaded from PhyloPic. If not, the vector image from PhyloPic will be used to render a raster object of the desired size.

Value

If `format` is "vector", a Picture object is returned. If `format` is "raster", a png array representing the rasterized image is returned. Either way, the uuid and download url are included as the "uuid" and "url" attributes, respectively.

Examples

```
## Not run:
# uuid
uuid <- "9fae30cd-fb59-4a81-a39c-e1826a35f612"

# Get data for an image
img_svg <- get_phylopic(uuid, format = "vector") # vector format
img_png <- get_phylopic(uuid, format = "raster") # raster format

## End(Not run)
```

`get_uuid` *Get a PhyloPic uuid*

Description

This function provides a convenient way to obtain a valid uuid or image url for an input taxonomic name. As multiple silhouettes can exist for each species in PhyloPic, this function extracts the primary image.

Usage

```
get_uuid(name = NULL, img = NULL, n = 1, filter = NULL, url = FALSE)
```

Arguments

`name`	`character`. A taxonomic name. Various taxonomic levels are supported (e.g. species, genus, family). NULL can also be supplied which will skip the taxonomic filtering of the PhyloPic database.
`img`	A Picture or png array object from `get_phylopic()`. A list of these objects can also be supplied. If `img` is supplied, `name` and `n` are ignored. Defaults to NULL.
`n`	`numeric`. How many uuids should be returned? Depending on the requested `name`, multiple silhouettes might exist. If `n` exceeds the number of available images, all available uuids will be returned. This argument defaults to 1.
`filter`	`character`. Filter uuid(s) by usage license. Use "by" to limit results to image uuids which do not require attribution, "nc" for image uuids which allow commercial usage, and "sa" for image uuids without a ShareAlike clause. The user can also combine these filters as a vector.
`url`	`logical`. If `FALSE` (default), only the uuid is returned. If `TRUE`, a valid PhyloPic image url of the uuid is returned.

Details

This function returns uuid(s) or image url (svg) for an input `name`. If a specific image is desired, the user can make use of pick_phylopic to visually select the desired uuid/url.

Value

A `character` vector of a valid PhyloPic uuid or svg image url.

Examples

```
## Not run:
uuid <- get_uuid(name = "Acropora cervicornis")
uuid <- get_uuid(name = "Dinosauria", n = 5, url = TRUE)

## End(Not run)
```

`phylopic_key_glyph` *Use PhyloPics as legend key glyphs*

Description

Specify existing images, taxonomic names, or PhyloPic uuids to use PhyloPic silhouettes as legend key glyphs in a ggplot plot.

Usage

```
phylopic_key_glyph(img = NULL, name = NULL, uuid = NULL)
```

Arguments

`img`	A Picture or png array object, e.g., from using `get_phylopic()`.
`name`	`character`. A taxonomic name to be passed to `get_uuid()`.
`uuid`	`character`. A valid uuid for a PhyloPic silhouette (such as that returned by `get_uuid()` or `pick_phylopic()`).

Details

One (and only one) of `img`, `name`, or `uuid` must be specified.

This argument may also be a vector/list of valid values if multiple silhouettes should be used as key glyphs. In this case, the specified silhouettes will be used as ordered as key glyphs one by one, with recycling as necessary.

Note that the sizes of the silhouettes in the legend are currently maximized based on the size of the key. This size can be modified using the `legend.key.size` argument in `ggplot2::theme()`. Therefore, the silhouettes will not reflect the size aesthetic, and this function should not be used for size legends.

Examples

```
## Not run:
library(ggplot2)
df <- data.frame(x = c(2, 4), y = c(10, 20),
                 name = c("Felis silvestris catus", "Odobenus rosmarus"))
ggplot(df) +
  geom_phylopic(aes(x = x, y = y, name = name, color = name), size = 10,
                show.legend = TRUE,
                key_glyph = phylopic_key_glyph(name =
                                               c("Felis silvestris catus",
                                                 "Odobenus rosmarus"))) +
  coord_cartesian(xlim = c(1,6), ylim = c(5, 30))

## End(Not run)
```

`pick_phylopic` *Pick a PhyloPic image from available options*

Description

This function provides a visually interactive way to pick an image and valid uuid for an input taxonomic name. As multiple silhouettes can exist for each organism in PhyloPic, this function is useful for choosing the right image/uuid for the user.

Usage

```
pick_phylopic(
  name = NULL,
  n = 5,
  uuid = NULL,
  view = 1,
  filter = NULL,
  auto = NULL
)
```

Arguments

`name`	`character`. A taxonomic name. Different taxonomic levels are supported (e.g. species, genus, family).
`n`	`numeric`. How many uuids should be viewed? Depending on the requested `name`, multiple silhouettes may exist. If `n` exceeds the number of available images, all available uuids will be returned. Defaults to 5. Only relevant if `name` supplied.
`uuid`	`character`. A vector (or list) of valid PhyloPic silhouette uuids, such as that returned by `get_uuid()` or `resolve_phylopic()`.
`view`	`numeric`. Number of silhouettes that should be plotted at the same time. Defaults to 1.
`filter`	`character`. Filter uuid(s) by usage license. Use "by" to limit results to image uuids which do not require attribution, "nc" for image uuids which allow commercial usage, and "sa" for image uuids without a ShareAlike clause. The user can also combine these filters. Only relevant if `name` supplied.
`auto`	`numeric`. This argument allows the user to automate input into the menu choice. If the input value is `1`, the first returned image will be selected. If the input value is `2`, requested images will be automatically cycled through with the final image returned. If the input value is `3`, a list of attribution information for each image is returned (this functionality is principally intended for testing). If `NULL` (default), the user must interactively respond to the called menu.

Details

This function allows the user to visually select the desired image from a pool of silhouettes available for the input `name`.

Note that while the `view` argument can be any positive integer, weaker/older computers may have issues displaying very large numbers of images at the same time (i.e. `view` > 9). If no images are displayed in your plotting environment, try decreasing the value of `view`.

Value

A Picture object is returned. The uuid of the selected image is saved as the "uuid" attribute of the returned object and is also printed to console.

Examples

```
## Not run:
# Defaults pane layout
img <- pick_phylopic(name = "Canis lupus", n = 5)
# 3 x 3 pane layout
img <- pick_phylopic(name = "Scleractinia", n = 9, view = 9)

## End(Not run)
```

`plot.Picture` *Preview a PhyloPic silhouette*

Description

Preview a raster or vector representation of a PhyloPic silhouette. This will plot the silhouette on a new page in your default plotting device.

Usage

```
## S3 method for class 'Picture'
plot(x, ...)

## S3 method for class 'phylopic'
plot(x, ...)
```

Arguments

`x`	A Picture or png array object, e.g., from using `get_phylopic()`.
`...`	Other arguments passed on to `grImport2::grid.picture()` or `grid::grid.raster()`.

`recolor_phylopic` *Recolor a PhyloPic image*

Description

Function to recolor and change alpha levels of a PhyloPic image.

Usage

```
recolor_phylopic(
  img,
  alpha = 1,
  color = NULL,
  fill = NULL,
  remove_background = TRUE
)
```

Arguments

`img`	A Picture or png array object, e.g., from using `get_phylopic()`.
`alpha`	`numeric`. A value between 0 and 1, specifying the opacity of the silhouette.
`color`	`character`. Color to make the outline of the silhouette. If NULL, the outline color is not changed.
`fill`	`character`. Color to make the body of the silhouette. If NULL, the body color is not changed.
`remove_background`	`logical`. Should any white background be removed? Only useful if `img` is a Picture object. See details.

Details

Some PhyloPic silhouettes do not have a transparent background. Consequently, when color is used with vectorized versions of these images, the entire image–including the background–is recolored. Setting `remove_background` to `TRUE` (the default) will remove any white parts of the image (which should only be the background).

Value

A Picture or png array object (matching the type of `img`)

See Also

Other transformations: `flip_phylopic()`, `rotate_phylopic()`

`resolve_phylopic` *Use a taxonomic name from another database to get a PhyloPic image UUID*

Description

This function takes a supplied taxonomic name, queries it via the specified external API, resolves the API's returned taxonomic ID to the PhyloPic taxonomic node database, then retrieves PhyloPic image uuids (or urls) for that node.

Usage

```
resolve_phylopic(
  name,
  api = "gbif.org",
  hierarchy = FALSE,
  max_ranks = 5,
  n = 1,
  filter = NULL,
  url = FALSE
)
```

Arguments

`name`	`character`. A full or partial taxonomic name to be queried via the specified `api`.
`api`	`character`. The API in which to query `name`. See Details for the available options.
`hierarchy`	`logical`. Whether the taxonomic hierarchy of `name` should be retrieved from the API and used to get several sets of PhyloPic image uuids (or urls).
`max_ranks`	`numeric`. The maximum number of taxonomic ranks that should be included if `hierarchy` is `TRUE`.
`n`	`numeric`. How many uuids should be returned? Depending on the requested `name`, multiple silhouettes might exist. If `n` exceeds the number of available images, all available uuids will be returned. This argument defaults to 1.
`filter`	`character`. Filter uuid(s) by usage license. Use "by" to limit results to image uuids which do not require attribution, "nc" for image uuids which allow commercial usage, and "sa" for image uuids without a ShareAlike clause. The user can also combine these filters as a vector.
`url`	`logical`. If `FALSE` (default), only the uuid is returned. If `TRUE`, a valid PhyloPic image url of the uuid is returned.

Details

If `hierarchy` is `FALSE`, the specified `name` is queried via the specified `api`. The matched id is then resolved to the matching taxonomic node in the PhyloPic database. If `hierarchy` is `TRUE`, the full taxonomic hierarchy for `name` is retrieved from the specified `api`, those taxonomic names are subset to `max_ranks` ranks (starting from the specified `name` and ascending the hierarchy). Then each of those names is resolved to the matching taxonomic node in the PhyloPic database (where possible). In either case, `get_uuid()` is then used to retrieve `n` image UUID(s) for each taxonomic name.

Note that while the names of the returned list are the taxonomic names as reported by the specified `api`, the PhyloPic images that are returned are associated with whatever taxonomic node that taxonomic name resolves to in the PhyloPic database. This means that, if `hierarchy` is `TRUE`, the same images may be returned for multiple taxonomic names. Also, if a particular taxonomic name does not resolve to any node in the PhyloPic database, no images will be returned for that name.

The following APIs are available for querying (`api`):

- "eol.org": the Encyclopedia of Life (note: `hierarchy = TRUE` is not currently available for this API) ("eol" is also allowed)
- "gbif.org": the Global Biodiversity Information Facility ("gbif" is also allowed)
- "marinespecies.org": the World Registor of Marine Species ("worms" is also allowed)
- "opentreeoflife.org": the Open Tree of Life ("otol" is also allowed)
- "paleobiodb.org": the Paleobiology Database ("pbdb" is also allowed)

Value

A `list` where each value is `n` (or fewer) PhyloPic image uuids (or urls if `url = TRUE`) and each name is the taxonomic name as matched and reported by the specified `api`. If `hierarchy` is `FALSE`, the list has length 1. If `hierarchy` is `TRUE`, the list has maximum length `max_ranks`.

Examples

```
## Not run:
# get a uuid for a single name
resolve_phylopic(name = "Canis lupus")
# get uuids for the taxonomic hierarchy
resolve_phylopic(name = "Velociraptor mongoliensis", api = "paleobiodb.org",
                 hierarchy = TRUE, max_ranks = 3)

## End(Not run)
```

`rotate_phylopic` *Rotate a PhyloPic silhouette*

Description

The picture can be a Picture or png array object, e.g., from using `get_phylopic()`. Note that png array objects can only be rotated by multiples of 90 degrees.

Usage

```
rotate_phylopic(img, angle = 90)
```

Arguments

`img`	A Picture or png array object, e.g., from using `get_phylopic()`.
`angle`	`numeric`. The number of degrees to rotate the silhouette clockwise.

Value

A Picture or png array object (matching the type of `img`)

See Also

Other transformations: `flip_phylopic()`, `recolor_phylopic()`

`save_phylopic` *Save a PhyloPic image*

Description

`save_phylopic` is a convenient function for saving a PhyloPic silhouette fetched via `get_phylopic`. Available formats for saving are those available in the grDevices package (i.e. pdf, png, svg, tiff, jpeg, and bmp).

Usage

```
save_phylopic(img = NULL, path = NULL, bg = "transparent", ...)
```

Arguments

`img`	`picture`. A Picture object. For PhyloPic silhouettes, this is generated by `get_phylopic` and `pick_phylopic`.
`path`	`character`. The output file path. The name of the file and file extension should be included (e.g. "./phylopic.png"). If this argument is `NULL` (default), the file is automatically saved in the current working directory as a png file.
`bg`	`character`. The desired background color. By default this argument is set to "transparent". Note: .jpeg files are always exported with a background colour.
`...`	Additional arguments available for the called grDevices function. The called function is determined by the file extension used in `path` (e.g. png).

Details

This function can be used to save a PhyloPic image obtained via the `get_phylopic` or any object of class `Picture` from the grImport2 package. Any file format available in the grDevices package can be used (i.e. pdf, png, svg, tiff, jpeg, and bmp).

Value

An image is saved. Nothing is returned to the R interface.

Examples

```
## Not run:
# Get image data
img <- get_phylopic("27356f15-3cf8-47e8-ab41-71c6260b2724")
# Save file
save_phylopic(img = img, path = "./phylopic.png",
              width = 500, height = 500)

## End(Not run)
```

---

scale_height_continuous

Scales for phylopic height or width

Description

`scale_height_continuous()` scales the height of silhouettes whereas `scale_width_continuous()` scales the height of silhouettes.

Usage

```
scale_height_continuous(
  name = waiver(),
  breaks = waiver(),
  labels = waiver(),
  limits = NULL,
  range = c(1, 6),
  transform = "identity",
  trans = lifecycle::deprecated(),
  guide = "none"
)

scale_width_continuous(
  name = waiver(),
  breaks = waiver(),
  labels = waiver(),
  limits = NULL,
  range = c(1, 6),
  transform = "identity",
  trans = lifecycle::deprecated(),
  guide = "none"
)
```

Arguments

`name`	The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If `NULL`, the legend title will be omitted.
`breaks`	One of: - `NULL` for no breaks - `waiver()` for the default breaks computed by the transformation object - A numeric vector of positions - A function that takes the limits as input and returns breaks as output (e.g., a function returned by `scales::extended_breaks()`). Note that for position scales, limits are provided after scale expansion. Also accepts rlang lambda function notation.
`labels`	One of: - `NULL` for no labels - `waiver()` for the default labels computed by the transformation object - A character vector giving labels (must be same length as `breaks`) - An expression vector (must be the same length as breaks). See ?plotmath for details. - A function that takes the breaks as input and returns labels as output. Also accepts rlang lambda function notation.
`limits`	One of: - `NULL` to use the default scale range - A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum - A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang lambda function notation. Note that setting limits on positional scales will **remove** data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see `coord_cartesian()`).
`range`	a numeric vector of length 2 that specifies the minimum and maximum size of the plotting symbol after transformation.
`transform`	For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time". A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called `transform_<name>`. If transformations require arguments, you can call them from the scales package, e.g. `scales::transform_boxcox(p = 2)`. You can create your own transformation with `scales::new_transform()`.
`trans`	**[Deprecated]** Deprecated in favour of `transform`.
`guide`	A function used to create a guide or its name. See `guides()` for more information.

Index