Package ‘ggpolypath’
September 25, 2023

Type Package
Title Polygons with Holes for the Grammar of Graphics
Version 0.3.0
Description Tools for working with polygons with holes in 'ggplot2', with a new 'geom' for drawing a 'polypath' applying the 'evenodd' or 'winding' rules.
BugReports https://github.com/mdsumner/ggpolypath/issues/
Depends R (>= 3.1), ggplot2 (>= 2.1.0)
Suggests rmarkdown, knitr
LazyData yes
License GPL-3
VignetteBuilder knitr
RoxygenNote 7.2.3
Encoding UTF-8
NeedsCompilation no
Author Michael D. Sumner [aut, cre], Kohske Takahashi [ctb] (original author of 'geom_holygon')
Maintainer Michael D. Sumner <mdsumner@gmail.com>
Repository CRAN
Date/Publication 2023-09-25 05:20:02 UTC

R topics documented:

dathome .......................................................... 2
gardenstate ....................................................... 3
geom_polypath ................................................... 3

Index 7
**dathome**

Simple polygon data

---

**Description**

A "home" profile of three objects with multiple parts as two related data frames.

**Format**

`dathome` is the metadata, a data frame with columns:

- **name** A descriptive name
- **colour** A colour to distinguish each object
- **FAD** An arbitrary numeric value
- **object_*** Key attribute, linking this object to its geometry in `maphome`

`maphome` is the geometry, a data frame with columns:

- **object_*** Key attribute, linking this row to its metadata in `dathome`
- **branch_*** Group attribute, unique values identify a closed ring
- **island_*** Logical, TRUE for "island" vs. "hole"
- **order_*** Numeric value to identify sort order within branch
- **x_**, **y_** x and y coordinate

**Details**

`maphome` is the geometry

**Examples**

```r
ggplot(maphome) + aes(x = x_, y = y_, group = branch_, fill = object_) + geom_polypath() + geom_path() + facet_wrap(~object_, nrow = nrow(dathome))```
gardenstate
Province polygons with inland waters as holes.

Description
A data frame of coordinates and geometry classifiers of the garden state, South Australia.

Format
gardenstate is the geometry, a data frame with columns:

x,y  x and y coordinate
id   Key attribute for the objects
piece,part Group attribute, unique values identify a closed ring, part is the part 'id' within an object
hole Logical, FALSE for "island" vs. "hole"
order Numeric value to identify sort order within branch

Details
The PROJ.4 string for this map is:
+proj=lcc +lat_1=-47 +lat_2=-17 +lat_0=-32 +lon_0=136 +x_0=0 +y_0=0 +ellps=WGS84
+towgs84=0,0,0,0,0,0,0 +units=m +no_defs

Examples
gs <- ggplot(gardenstate)
gs <- gs + aes(x = x, y = y, group = group, fill = id)
gs + geom_polypath() + geom_path()

geom_polypath
Geom polypath, a polygon filled path that can include holes.

Description
Polygons are drawn by tracing a 'path' of linked vertices and applying rule to differentiate the inside and the outside of the area traversed. The 'evenodd' rule provides the normal expected behaviour seen in simple GIS geometry and is immune to self-intersections and the orientation of the path (clockwise or anti-clockwise). The 'winding' rule behaves differently for self-intersections depending on relative orientation of the interacting paths.
Usage

```r
gem_polypath(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  rule = "winding",
  ...
)
```

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, either as a `ggproto` Geom subclass or as a string naming the stat stripped of the `stat_` prefix (e.g. "count" rather than "stat_count").

- **position**: Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use `position_jitter`), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.

- **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? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, 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()`.

- **rule**: Character value specifying the path fill mode: either "winding" or "evenodd", see `polypath`

- **...**: Other arguments passed on to `layer()`. These are often aesthetics, used to set an aesthetic to a fixed value, like `colour = "red"` or `size = 3`. They may also be parameters to the paired geom/stat.
Details


Value

a ggplot2 layer

See Also

`polypath` and `pathGrob geom_polygon` for the implementation on `polygonGrob`, `geom_map` for a convenient way to tie the values and coordinates together, `geom_path` for an unfilled polygon, `geom_ribbon` for a polygon anchored on the x-axis

Examples

# When using `geom_polypath`, you will typically need two data frames:
# one contains the coordinates of each polygon (positions), and the
# other the values associated with each polygon (values). An id
# variable links the two together.
# Normally this would not be created manually, but by using `\code{\link{fortify}}`
# to generate it from the Spatial classes in the 'sp' package.

## the built-in data `\code{\link{home}}` uses nested data frames
library(ggplot2)
ggplot(maphome) + aes(x = x_, y = y_, group = branch_, fill = factor(object_)) +
geom_polypath()

## this is the same example built from scratch
positions = data.frame(x = c(0, 0, 46, 46, 0, 7, 13, 13, 7, 7, 18, 24, 24, 18, 18, 31, 37, 37, 31, 31, 18.4, 18.4, 18.6, 18.8, 18.8, 18.6, 18.4, 31, 31, 37, 37, 31, 0, 21, 31, 37, 46, 0, 18, 18, 24, 24, 18, 18.4, 18.6, 18.8, 18.8, 18.6, 18.4, 18.4),
y = c(0, 19, 19, 0, 0, 6, 6, 13, 13, 6, 1, 1, 12, 1, 4, 4, 11, 11, 4, 6.899999999999999, 7.499999999999999, 7.499999999999999, 7.699999999999999, 7.499999999999999, 6.899999999999999, 6.699999999999999, 6.899999999999999, 27, 34, 34, 24, 27, 32, 27, 24, 19, 19, 1, 12, 12, 1, 1, 6.899999999999999, 6.699999999999999, 6.699999999999999, 7.499999999999999, 7.499999999999999, 6.899999999999999, 6.899999999999999),
values <- data.frame(  
id = unique(positions$id),  
value = c(2, 5.4, 3)  
)  
# manually merge the two together
datapoly <- merge(values, positions, by = c("id"))

# the entire house
(house <- ggplot(datapoly, aes(x = x, y = y)) + geom_polypath(aes(fill = value, group = group)))

# just the front wall (and chimney), with its three parts, the first of which has three holes
wall <- ggplot(datapoly[datapoly$id == 1, ], aes(x = x, y = y))
wall + geom_polypath(aes(fill = id, group = group))
Index

* datasets
  geom_poly PATH, 3

aes(), 4
borders(), 4
dathome, 2, 2
fortify(), 4
gardenstate, 3
geom_map, 5
geom_path, 5
geom_polygon, 5
geom_poly PATH, 3
geom_ribbon, 5
GeomPoly Path (geom_poly PATH), 3
ggplot(), 4
layer(), 4
maphome, 2
maphome (dathome), 2
pathGrob, 5
polygonGrob, 5
polypath, 4, 5