Package 'ggnext'

August 26, 2026

Title A Next-Generation Grammar of Graphics

Version 0.1.0

Description An implementation of the Grammar of Graphics described by Wilkinson (2005, ISBN:978-0-387-24544-7), built on 'S7' classes. The familiar grammar vocabulary of aesthetics, geometries, statistics, scales, coordinates, facets and themes composed with '+' is preserved; every constructor that takes an argument also accepts a plot as the first argument of a native pipe ('|>') stage, so the two styles are interchangeable. The package is extended with built-in interactivity, animation, an exact-data export, a plot linter that flags common statistical-graphics mistakes before a figure ships, and a catalogue covering layout diagrams (Sankey, treemap, network, radar), machine-learning diagnostics (SHAP, receiver operating characteristic, calibration, partial dependence) and clinical reporting (Kaplan-Meier, forest, swimmer, CONSORT).
Static output is written to Scalable Vector Graphics; interactive output is a self-contained 'HTML' document using a canvas element and vanilla 'JavaScript'. Both render targets consume the same computed-geometry buffer, giving a single source of truth for layer geometry. Layout and estimation algorithms follow their published descriptions, including Bruls, Huizing and van Wijk (2000) <doi:10.1007/978-3-7091-6783-0_4> for squarified treemaps, Fruchterman and Reingold (1991) <doi:10.1002/spe.4380211102> for force-directed graphs, and Kaplan and Meier (1958) <doi:10.1080/01621459.1958.10501452> for survival curves.

URL https://itsmdivakaran.github.io/ggnext/, https://github.com/itsmdivakaran/ggnext

BugReports https://github.com/itsmdivakaran/ggnext/issues

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1.0)

Imports S7, grDevices, stats, utils

Suggests knitr, markdown, rmarkdown, testthat (>= 3.0.0), xml2

Config/testthat/edition 3

Collate 'aes.R' 'build.R' 'animation.R' 'classes.R' 'api.R' 'colors.R' 'facet.R' 'marks.R' 'stats.R' 'geoms.R' 'geoms-base.R' 'geoms-clinical.R' 'geoms-layout.R' 'geoms-ml.R' 'stats-special.R' 'geoms-special.R' 'ggnext-package.R' 'json.R' 'labels.R' 'logo.R' 'scale-discrete.R' 'theme.R' 'pipe.R' 'plot-data.R' 'render.R' 'scale-math.R' 'site.R' 'stats-base.R' 'stats-clinical.R' 'stats-layout.R' 'stats-layout2.R' 'stats-ml.R' 'svg.R' 'validate.R'

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Mahesh Divakaran [aut, cre, cph]

Maintainer Mahesh Divakaran <itsmdivakaran@gmail.com>

Repository CRAN

Date/Publication 2026-08-26 18:40:02 UTC

Contents

`aes` *Construct aesthetic mappings*

Description

`aes()` captures unevaluated expressions that describe how columns of the data map onto visual properties. The first two unnamed arguments are taken as `x` and `y`. Expressions are evaluated later, against the layer data, in the environment where `aes()` was called.

Usage

```
aes(...)
```

Arguments

`...`	Name-value pairs of aesthetics and expressions, e.g. `aes(displ, hwy, color = class)`. Supported aesthetics: `x`, `y`, `color` (alias `colour`), `size`.

Value

An object of class `ggnext_aes`.

Examples

```
aes(speed, dist)
aes(x = speed, y = dist, color = gear)
```

`animate` *Animate a plot over a transition variable*

Description

Animation is a property of the plot, not a separate verb chain: the same geometry pipeline runs once per level of `by`, and the interactive target plays the resulting frames with a scrubber and play/pause control. Because frames are ordinary geometry buffers, positions are interpolated by the player rather than recomputed.

Usage

```
animate(by, duration = 800, easing = "cubic-in-out", loop = TRUE)
```

Arguments

`by`	Transition variable: an unquoted column name or a string.
`duration`	Milliseconds per frame.
`easing`	`"linear"`, `"cubic"`, or `"cubic-in-out"`.
`loop`	Restart automatically after the last frame.

Details

Animated plots render to the interactive target (like `interact()`). `render(p, target = "static")` still produces a static SVG of the whole data, so animation never blocks a static export.

Value

An Animation spec to add to a plot with `+`.

Examples

```
d <- data.frame(
  x = rep(1:5, 3), y = c(1:5, (1:5)^1.5, (1:5)^2),
  step = rep(c(1, 2, 3), each = 5)
)
p <- ggnext(d, aes(x, y)) + geom_point(size = 5) + animate(step)
html <- render(p)
```

`Animation` *Animation: a transition spec bound to a data variable*

Description

Created by `animate()`. Frames are computed from the same geometry pipeline as static output — one buffer per frame value — and the interactive target plays them with a scrubber.

Usage

```
Animation(
  var = character(0),
  duration = integer(0),
  easing = character(0),
  loop = logical(0)
)
```

Arguments

`var`	Name of the transition variable.
`duration`	Milliseconds per frame.
`easing`	Easing curve: `"linear"`, `"cubic"`, `"cubic-in-out"`.
`loop`	Restart automatically at the end.

Value

An S7 object of class `Animation`; usually built by `animate()`.

build_marks *Build drawing primitives ("marks") from scaled aesthetic values*

Description

The single seam between the grammar and the renderers. Each geom method returns a list of marks; a mark is a plain named list with a `type` field (currently `"circle"`; the full catalog will add `"rect"`, `"line"`, `"path"`, `"polygon"`, `"text"`) plus type-specific fields in *normalized panel coordinates* (x/y in [0, 1], y pointing up). Because both render targets consume marks, adding a geom never touches renderer code, and adding a renderer never touches geom code.

Usage

```
build_marks(geom, ...)
```

Arguments

`geom`	A Geom subclass instance.
`...`	Method arguments; all methods take `scaled`, a named list of scaled aesthetic vectors (`x`, `y` in [0, 1]; `color` as hex; `size` as radius in px; `alpha` in [0, 1]).

Value

List of marks.

build_site *Build the ggnext documentation site*

Description

Renders every gallery example with the package itself and writes a self-contained static site. No external site generator, no CDN assets.

Usage

```
build_site(dir, quiet = FALSE, gallery = TRUE, cookbook = TRUE)
```

Arguments

`dir`	Output directory (created if needed). There is no default — CRAN policy prohibits writing to the user's home filespace by default, so a path must always be supplied explicitly.
`quiet`	Suppress progress messages.

gallery	Render the gallery figures. Every example is drawn with ggnext itself at build time, which is the bulk of the build cost. Pass FALSE to rewrite the pages without redrawing the figures — useful when iterating on prose, and what makes the example below quick. The pages still build; they simply reference whatever figures are already in dir.
cookbook	Also build the Cookbook page by knitting the worked reference shipped in inst/examples/. Needs the knitr and markdown packages; the page is skipped with a message if either is missing. It renders ~100 plots, so it is the other slow part.

Value

The output directory, invisibly.

Examples

```
# The figures and the cookbook are the slow parts of a real build; this
# writes the pages only. For the full site, pass a directory of your
# choosing, e.g. build_site("docs").
out <- file.path(tempdir(), "ggnext-site")
build_site(out, quiet = TRUE, gallery = FALSE, cookbook = FALSE)
list.files(out)
```

ColorScale *ColorScale: palette override for the color aesthetic*

Description

Created by scale_color_manual() / scale_color_gradient().

Usage

```
ColorScale(palette = character(0), name = NULL, type = character(0))
```

Arguments

palette	Character vector of hex colors (two, for a gradient).
name	Optional legend title.
type	"discrete" or "continuous".

Value

An S7 object of class ColorScale; usually built by scale_color_manual() or scale_color_gradient().

compute_stat *Apply a stat's transformation to evaluated aesthetic values*

Description

Methods receive `values`, a named list of evaluated aesthetic vectors, and return a named list of (possibly transformed) aesthetic vectors.

Usage

```
compute_stat(stat, ...)
```

Arguments

`stat`	A Stat subclass instance.
`...`	Method arguments; all methods take `values` (named list of evaluated aesthetic vectors).

Value

Named list of (possibly transformed) aesthetic vectors.

Coord *Coord: position transformation for normalized coordinates*

Description

Base class for coordinate systems. A coord receives positions already normalized to [0, 1] by the scales and returns (possibly transformed) normalized positions. Cartesian is the identity; polar / flipped / 3D coords will override this seam without touching scales or geoms.

Usage

```
Coord(name = character(0))
```

Arguments

`name`	Human-readable coord name.

Value

An S7 object of class `Coord`, the base class for coordinate systems.

CoordCartesian *CoordCartesian: the identity coordinate system*

Description

CoordCartesian: the identity coordinate system

Usage

```
CoordCartesian(flip = FALSE)
```

Arguments

flip	Swap x and y (see coord_flip()).

Value

An S7 object of class CoordCartesian; usually built by coord_cartesian() or coord_flip().

CoordPolar *CoordPolar: polar coordinate system*

Description

Maps the normalized x axis onto angle and the normalized y axis onto radius, so lines, areas, and points become circular. This is the engine behind radar/spider charts, pie-like wedges, and circular bar charts.

Usage

```
CoordPolar(theta = "x", start = 0, direction = 1, inner = 0)
```

Arguments

theta	Which axis becomes the angle: "x" (default) or "y".
start	Angle in radians for normalized position 0 (default: 12 o'clock).
direction	1 for clockwise, -1 for counter-clockwise.
inner	Inner radius as a fraction of the outer radius (a donut hole); 0 fills to the center.

Value

An S7 object of class CoordPolar; usually built by coord_polar().

coord_cartesian *Cartesian coordinate system*

Description

The identity coordinate system.

Usage

```
coord_cartesian(flip = FALSE)
```

Arguments

flip	Swap the x and y axes (horizontal bars, forest plots).

Value

A CoordCartesian object to add with +.

coord_flip *Flipped Cartesian coordinates*

Description

Draws the plot with x and y swapped — the usual way to get horizontal bars or a readable categorical axis with long labels.

Usage

```
coord_flip()
```

Value

A CoordCartesian object to add with +.

Examples

```
d <- data.frame(g = c("alpha", "beta", "gamma"), v = c(3, 7, 5))
ggnext(d, aes(g, v)) + geom_col() + coord_flip()
```

coord_polar *Polar coordinates*

Description

Bends the panel into a circle: the `theta` axis becomes the angle and the other axis becomes the radius. This is the engine behind radar charts (`geom_radar()`), pie/donut wedges, and circular bar charts.

Usage

```
coord_polar(theta = "x", start = 0, direction = 1, inner = 0)
```

Arguments

`theta`	Which axis maps to angle: `"x"` (default) or `"y"`.
`start`	Angle in radians for the first position; `0` is 12 o'clock.
`direction`	`1` clockwise (default), `-1` counter-clockwise.
`inner`	Inner radius as a fraction of the outer radius — use e.g. `0.3` for a donut hole.

Value

A CoordPolar object to add to a plot with `+`.

Examples

```
d <- data.frame(g = c("A", "B", "C", "D"), v = c(4, 7, 3, 6))
ggnext(d, aes(g, v)) + geom_col() + coord_polar()
```

coord_transform *Transform normalized positions through a coordinate system*

Description

Transform normalized positions through a coordinate system

Usage

```
coord_transform(coord, ...)
```

Arguments

`coord`	A Coord subclass instance.
`...`	Method arguments; all methods take `x` and `y`, numeric vectors of normalized positions.

Value

List with transformed `x` and `y`.

`Facet` *Facet: split a plot into panels by data values*

Description

Base class for faceting. Created by `facet_wrap()` / `facet_grid()`.

Usage

```
Facet(
  vars = character(0),
  ncol = NULL,
  nrow = NULL,
  scales = character(0),
  type = character(0)
)
```

Arguments

`vars`	Character vector of faceting variable names.
`ncol`, `nrow`	Panel grid shape (`facet_wrap` only; `NULL` auto-sizes).
`scales`	`"fixed"` (shared axes, the default) or `"free"`, `"free_x"`, `"free_y"` (per-panel axes).
`type`	`"wrap"` or `"grid"`.

Value

An S7 object of class `Facet`; usually built by `facet_wrap()` or `facet_grid()`.

`facet_grid` *Lay panels out in a rows-by-columns grid*

Description

`facet_grid(rows, cols)` builds a two-way panel matrix: one row per level of `rows`, one column per level of `cols`.

Usage

```
facet_grid(rows, cols = NULL, scales = "fixed")
```

Arguments

rows	Variable defining the panel rows (unquoted name or string).
cols	Variable defining the panel columns; NULL for a single column.
scales	As in facet_wrap().

Value

A Facet object to add to a plot with +.

Examples

```
d <- transform(mtcars, cyl = factor(cyl), am = factor(am))
ggnext(d, aes(disp, mpg)) + geom_point() + facet_grid(am, cyl)
```

---

facet_wrap *Wrap panels into a grid*

Description

Splits the data by one or more variables and lays the resulting panels out in a rectangular grid, wrapping to a new row as needed.

Usage

```
facet_wrap(vars, ncol = NULL, nrow = NULL, scales = "fixed")
```

Arguments

vars	Faceting variables: unquoted names (facet_wrap(cyl)), a character vector, or several names.
ncol, nrow	Force a panel-grid shape; NULL picks a near-square layout.
scales	"fixed" (all panels share axes — best for comparison), "free_x", "free_y", or "free" (each panel scales to its own data).

Value

A Facet object to add to a plot with +.

Examples

```
ggnext(iris, aes(Sepal.Length, Sepal.Width)) +
  geom_point() +
  facet_wrap(Species)
```

Geom *Geom: geometric representation of computed values*

Description

Base class for geoms. A geom turns scaled (normalized-to-[0, 1]) aesthetic values into a list of render-target-agnostic drawing primitives ("marks"); see `build_marks()`.

Usage

```
Geom(name = character(0), default_params = list(), required_aes = c("x", "y"))
```

Arguments

`name`	Human-readable geom name.
`default_params`	Named list of default visual parameters, used when neither an aesthetic mapping nor a literal layer parameter supplies a value.
`required_aes`	Aesthetics that must be present (post-stat) for this geom to draw; defaults to `x` and `y`.

Value

An S7 object of class `Geom`, the base class for geometric layers.

geom_abline *Sloped reference line*

Description

Draws `y = intercept + slope * x` across the panel. Like `geom_hline()` and `geom_vline()` it ignores the plot's `aes()`, so it never disturbs the data mapping.

Usage

```
geom_abline(
  intercept = 0,
  slope = 1,
  color = NULL,
  linewidth = NULL,
  alpha = NULL,
  dash = NULL
)
```

Arguments

`intercept`, `slope`
: Line parameters. `slope = 1, intercept = 0` gives the identity line used in agreement and calibration plots.

`color`, `linewidth`, `alpha`, `dash`
: Appearance.

Value

A Layer to add with +.

Examples

```
ggnext(cars, aes(speed, dist)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 3, dash = "5,4")
```

---

geom_ae_heatmap *Adverse-event incidence heatmap*

Description

Incidence by preferred term and treatment arm (or severity grade), shaded by rate and annotated with counts.

Usage

```
geom_ae_heatmap(mapping = NULL, data = NULL, label = TRUE, label_size = NULL)
```

Arguments

`mapping`, `data`	Standard layer overrides. Map the arm/grade to `x`, the AE term to `y`, and the incidence to `size`.
`label`	Annotate each cell with its value.
`label_size`	Annotation size in px.

Value

A Layer to add with +.

Examples

```
d <- expand.grid(arm = c("Placebo", "Low", "High"),
                 ae = c("Nausea", "Fatigue", "Headache"))
d$pct <- c(5, 12, 22, 8, 15, 26, 3, 6, 11)
ggnext(d, aes(arm, ae, size = pct)) + geom_ae_heatmap()
```

geom_alluvial *Alluvial diagram (ordered, repeated-measures categorical flow)*

Description

Tracks the same subjects through an ordered sequence of stages (e.g. visit 1 category -> visit 2 category -> visit 3 category), unlike `geom_sankey()`, which draws a general directed flow graph from an edge list. Internally it tallies every consecutive-stage transition and reuses `geom_sankey()`'s node-stacking and ribbon-path layout.

Usage

```
geom_alluvial(
  mapping = NULL,
  data = NULL,
  alpha = NULL,
  node_width = NULL,
  label = TRUE
)
```

Arguments

`mapping, data`	Standard layer overrides. Map the stage to `x` (an ordered factor keeps its declared stage order), the category at that stage to `y`, and the subject id to `group`.
`alpha`	Ribbon opacity.
`node_width`	Node bar width as a fraction of the panel.
`label`	Draw node labels.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  subject = rep(1:8, each = 3),
  visit = rep(c("Baseline", "Week 4", "Week 8"), 8),
  response = c(
    "SD", "SD", "PR", "SD", "PR", "PR", "SD", "SD", "SD",
    "PR", "PR", "CR", "SD", "PR", "PR", "SD", "SD", "PD",
    "PR", "CR", "CR", "SD", "PD", "PD"
  )
)
d$visit <- factor(d$visit, levels = c("Baseline", "Week 4", "Week 8"))
ggnext(d, aes(x = visit, y = response, group = subject)) +
  geom_alluvial() +
  theme_void()
```

geom_area — *Area layer (filled to the zero baseline)*

Description

Area layer (filled to the zero baseline)

Usage

```
geom_area(mapping = NULL, data = NULL, color = NULL, alpha = NULL)
```

Arguments

mapping, data, color, alpha	As in geom_point().

Value

A Layer.

geom_bar — *Bar chart layer (counts per category)*

Description

Counts rows at each x. For pre-computed heights use geom_col().

Usage

```
geom_bar(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  width = 0.8,
  position = "stack"
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
width	Bar width as a fraction of the slot.
position	"stack" (default), "dodge", or "identity".

Value

A Layer.

`geom_bland_altman` *Bland-Altman agreement plot*

Description

Difference against mean for two measurement methods, with the mean bias and 95% limits of agreement drawn as reference lines — the standard method-comparison plot.

Usage

```
geom_bland_altman(
  mapping = NULL,
  data = NULL,
  color = NULL,
  size = NULL,
  alpha = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Map the two methods' measurements to `x` and `y`; the stat computes mean and difference.
`color`	Point color.
`size`	Point radius.
`alpha`	Point opacity.

Value

A Layer to add with `+`.

Examples

```
set.seed(1)
a <- rnorm(60, 100, 12)
d <- data.frame(method_a = a, method_b = a + rnorm(60, 2, 5))
ggnext(d, aes(method_a, method_b)) + geom_bland_altman()
```

geom_blank *Blank layer*

Description

Draws nothing. Useful to establish scales without showing data, or as a placeholder in code that conditionally adds a layer.

Usage

```
geom_blank(mapping = NULL, data = NULL)
```

Arguments

mapping, data	Standard layer overrides.

Value

A Layer to add with +.

Examples

```
ggnext(cars, aes(speed, dist)) + geom_blank()
```

geom_boxplot *Boxplot layer*

Description

Boxplot layer

Usage

```
geom_boxplot(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  width = 0.6,
  coef = 1.5
)
```

Arguments

`mapping`, `data`, `color`, `alpha`	As in `geom_point()`.
`width`	Box width in x slot units.
`coef`	Whisker length multiplier (Tukey's 1.5 by default).

Value

A Layer.

---

geom_bump *Bump chart (rank over time)*

Description

Rank trajectories drawn with sigmoid interpolation between periods, so crossings read cleanly instead of as sharp zigzags.

Usage

```
geom_bump(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL,
  points = TRUE
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Map time to `x`, rank to `y`, and the series to `color`.
`color`, `alpha`, `linewidth`	Appearance.
`points`	Draw a marker at each period.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  year = rep(2021:2023, 3),
  rank = c(1, 2, 3, 2, 1, 1, 3, 3, 2),
  team = rep(c("A", "B", "C"), each = 3)
)
ggnext(d, aes(year, rank, color = team)) +
  geom_bump() + scale_y_reverse()
```

---

geom_calibration *Calibration curve*

Description

Bins predicted probabilities and plots the observed event rate in each bin against the mean prediction, with the diagonal marking perfect calibration. Points off the diagonal show over- or under-confidence.

Usage

```
geom_calibration(
  mapping = NULL,
  data = NULL,
  bins = 10,
  color = NULL,
  diagonal = TRUE
)
```

Arguments

mapping, data	Standard layer overrides. Map the predicted probability to x and the binary outcome (0/1) to y.
bins	Number of probability bins.
color	Curve color.
diagonal	Draw the perfect-calibration reference line.

Value

A Layer to add with +.

Examples

```
set.seed(1)
p <- runif(300)
d <- data.frame(pred = p, obs = rbinom(300, 1, p^1.3))
ggnext(d, aes(pred, obs)) + geom_calibration()
```

geom_chord *Chord diagram*

Description

Entities sit on a circle; each relationship is a ribbon whose ends are arcs proportional to the flow. Ribbon interiors are quadratic curves pulled toward the circle center.

Usage

```
geom_chord(
  mapping = NULL,
  data = NULL,
  alpha = NULL,
  label = TRUE,
  label_size = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Requires `x` (source), `xend` (target), and `y` (flow value).
`alpha`	Ribbon opacity.
`label`, `label_size`	Entity labels around the rim.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  from = c("A", "A", "B", "C"), to = c("B", "C", "C", "A"),
  n = c(5, 3, 7, 2)
)
ggnext(d, aes(x = from, xend = to, y = n)) + geom_chord() + theme_void()
```

geom_col *Column chart layer (bar heights from the data)*

Description

Column chart layer (bar heights from the data)

Usage

```
geom_col(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  width = 0.8,
  position = "stack"
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
width	Bar width as a fraction of the slot.
position	"stack" (default), "dodge", or "identity".

Value

A Layer.

---

geom_concordance *Concordance / agreement plot*

Description

Scatter of two methods' measurements against the identity line, with Lin's concordance correlation coefficient (CCC) annotated in the corner. Unlike geom_bland_altman() (difference vs. mean, for spotting systematic bias), this reads agreement directly off how close the cloud sits to y = x.

Usage

```
geom_concordance(
  mapping = NULL,
  data = NULL,
  color = NULL,
  size = NULL,
  alpha = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map the two methods' measurements to x and y.
color	Point color.
size	Point radius.
alpha	Point opacity.

Value

A Layer to add with +.

Examples

```
set.seed(1)
a <- rnorm(50, 10, 2)
d <- data.frame(method1 = a, method2 = a * 0.95 + rnorm(50, 0, 0.5))
ggnext(d, aes(method1, method2)) + geom_concordance()
```

---

geom_confusion_matrix *Confusion matrix*

Description

A tile heatmap of predicted vs actual classes, shaded by row-normalized rate and annotated with counts, so class imbalance does not hide errors.

Usage

```
geom_confusion_matrix(
  mapping = NULL,
  data = NULL,
  normalize = "row",
  label_size = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map the predicted class to x, the actual class to y. Provide counts via size, or pass raw per-observation rows and let the stat count them.
normalize	"row" (default), "col", "all", or "none" — which total the shading is relative to.
label_size	Annotation size in px.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  predicted = c("cat", "cat", "dog", "dog", "dog", "cat"),
  actual    = c("cat", "dog", "dog", "dog", "cat", "cat")
)
ggnext(d, aes(predicted, actual)) + geom_confusion_matrix()
```

geom_consort *CONSORT participant flow diagram*

Description

Boxes and arrows tracing participants from screening through analysis, laid out automatically from a stage/count table.

Usage

```
geom_consort(mapping = NULL, data = NULL, box_fill = NULL, label_size = NULL)
```

Arguments

mapping, data	Standard layer overrides. Map the stage label to label and the participant count to size; map x to a stage index to control ordering.
box_fill	Box fill color.
label_size	Text size in px.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  stage = c("Assessed for eligibility", "Randomised",
            "Received allocation", "Analysed"),
  n = c(420, 300, 291, 285)
)
ggnext(d, aes(label = stage, size = n)) + geom_consort()
```

geom_cor *Correlation matrix heatmap*

Description

All pairwise Pearson correlations among the numeric columns of data, shaded on a diverging scale (negative to positive) and annotated with the coefficient. Takes the raw wide data frame directly rather than through aes(), since the stat needs the whole numeric block at once to compute pairwise correlations, not row-wise aesthetic vectors — the same pattern geom_dendrogram() uses.

Usage

```
geom_cor(
  data,
  vars = NULL,
  label = TRUE,
  label_size = NULL,
  low = "#C1462F",
  mid = "#F2F0EA",
  high = "#12A594"
)
```

Arguments

`data`	A data frame; correlations are computed over its numeric columns.
`vars`	Character vector of numeric columns to include; default all numeric columns in `data`.
`label`	Annotate each cell with its correlation coefficient.
`label_size`	Annotation text size in px.
`low`, `mid`, `high`	Diverging fill colors for correlation -1, 0, +1.

Value

A Layer to add with +.

Examples

```
ggnext() +
  geom_cor(mtcars, vars = c("mpg", "cyl", "disp", "hp", "wt")) +
  labs(title = "Correlation matrix")
```

---

geom_count *Counted scatter*

Description

A scatter where the point area encodes how many observations share that position — the fix for a scatter of rounded or discrete values where overplotting hides the mass.

Usage

```
geom_count(mapping = NULL, data = NULL, color = NULL, alpha = NULL)
```

Arguments

`mapping`, `data`	Standard layer overrides.
`color`, `alpha`	Appearance.

Value

A Layer to add with +.

Examples

```
d <- data.frame(x = c(1, 1, 1, 2, 2, 3), y = c(1, 1, 2, 2, 2, 3))
ggnext(d, aes(x, y)) + geom_count()
```

`geom_crossbar` *Crossbar: an interval box with the estimate marked*

Description

A hollow box spanning `ymin` to `ymax` with a heavier line at `y` — the compact summary used in dose-response and subgroup tables.

Usage

```
geom_crossbar(
  mapping = NULL,
  data = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL,
  width = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Requires `x`, `y`, `ymin`, `ymax`.
`color`, `linewidth`, `alpha`	Appearance.
`width`	Box width in x-axis units.

Value

A Layer to add with +.

Examples

```
d <- data.frame(g = c("a", "b"), m = c(2, 3), lo = c(1, 2), hi = c(4, 5))
ggnext(d, aes(g, m, ymin = lo, ymax = hi)) + geom_crossbar()
```

geom_cuminc *Cumulative incidence (competing risks)*

Description

Aalen-Johansen cumulative incidence curves by event type, the correct estimator when competing events make 1 - Kaplan-Meier biased upward.

Usage

```
geom_cuminc(mapping = NULL, data = NULL, linewidth = NULL)
```

Arguments

mapping, data	Standard layer overrides. Map follow-up time to `time`, the event code to `status` (0 = censored, 1, 2, ... = event types), and optionally a treatment arm to `color`.
linewidth	Line width.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  t = rexp(120, 0.1),
  ev = sample(0:2, 120, replace = TRUE, prob = c(.4, .35, .25))
)
ggnext(d, aes(time = t, status = ev)) + geom_cuminc()
```

geom_curve *Curved connector*

Description

A quadratic curve between two points, for annotation leaders and relationship diagrams where a straight segment would overlap the data.

Usage

```
geom_curve(
  mapping = NULL,
  data = NULL,
  curvature = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Requires x, y, xend, yend.
curvature	How far the curve bows; negative bows the other way.
color, linewidth, alpha	Appearance.

Value

A Layer to add with +.

Examples

```
d <- data.frame(x = 1, y = 1, xe = 3, ye = 3)
ggnext(d, aes(x, y, xend = xe, yend = ye)) + geom_curve()
```

---

geom_decision_boundary

Decision-boundary region plot

Description

Shades a grid of predictions to show a two-dimensional classifier's decision regions; overlay geom_point() for the training data.

Usage

```
geom_decision_boundary(mapping = NULL, data = NULL, alpha = NULL)
```

Arguments

mapping, data	Standard layer overrides. Map the grid coordinates to x/y and the predicted class to color.
alpha	Region opacity.

Value

A Layer to add with +.

Examples

```
g <- expand.grid(x = seq(0, 1, 0.05), y = seq(0, 1, 0.05))
g$cls <- ifelse(g$x + g$y > 1, "a", "b")
ggnext(g, aes(x, y, color = cls)) + geom_decision_boundary()
```

---

geom_dendrogram *Hierarchical clustering dendrogram*

Description

Runs stats::hclust() (base R, not an added dependency) on the Euclidean distances between rows of data and draws the merge tree as the classic bracket shape: two vertical drops joined by one horizontal bar at the merge height. Takes the raw wide data frame directly (like geom_cor()), since the layout needs the whole numeric block at once, not row-wise aesthetics.

Usage

```
geom_dendrogram(
  data,
  vars = NULL,
  method = "complete",
  color = NULL,
  linewidth = NULL
)
```

Arguments

data	A data frame with one row per observation to cluster.
vars	Character vector of numeric columns to cluster on; default all numeric columns in data.
method	Linkage method, passed to stats::hclust() (default "complete").
color, linewidth	Line appearance.

Value

A Layer to add with +.

Examples

```
ggnext() +
  geom_dendrogram(mtcars[1:10, c("mpg", "hp", "wt", "qsec")]) +
  theme_void()
```

geom_density *Density curve layer*

Description

Density curve layer

Usage

```
geom_density(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL,
  adjust = 1
)
```

Arguments

mapping, data, color, alpha	As in geom_point(); map only x.
linewidth	Curve stroke width.
adjust	Bandwidth multiplier.

Value

A Layer.

geom_dose_response *Dose-response curve*

Description

A four-parameter log-logistic curve fitted to dose-response data, with a confidence band and the fitted EC50 marked.

Usage

```
geom_dose_response(
  mapping = NULL,
  data = NULL,
  color = NULL,
  points = TRUE,
  ec50 = TRUE
)
```

Arguments

mapping, data	Standard layer overrides. Map dose to x and response to y.
color	Curve color.
points	Draw the observed points.
ec50	Mark the fitted EC50 with a vertical line.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  dose = rep(c(0.1, 1, 10, 100, 1000), each = 3),
  resp = c(5, 7, 6, 18, 22, 20, 52, 48, 55, 82, 79, 85, 95, 97, 93)
)
ggnext(d, aes(dose, resp)) + geom_dose_response() + scale_x_log10()
```

---

geom_dotplot *Dot plot*

Description

A histogram built from one dot per observation, stacked within its bin. Better than bars for small samples, where it shows every data point.

Usage

```
geom_dotplot(
  mapping = NULL,
  data = NULL,
  binwidth = NULL,
  color = NULL,
  size = NULL,
  alpha = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map the values to x.
binwidth	Bin width in data units.
color, size, alpha	Appearance.

Value

A Layer to add with +.

Examples

```
ggnext(cars, aes(speed)) + geom_dotplot(binwidth = 2)
```

`geom_dumbbell` *Dumbbell layer (before/after per category)*

Description

Map `x` (start value), `xend` (end value), and `y` (category).

Usage

```
geom_dumbbell(
  mapping = NULL,
  data = NULL,
  color = NULL,
  size = NULL,
  alpha = NULL,
  linewidth = NULL,
  color_start = NULL,
  color_end = NULL
)
```

Arguments

`mapping`, `data`, `color`, `size`, `alpha`	As in `geom_point()`; `color` styles the connector.
`linewidth`	Connector width.
`color_start`, `color_end`	Endpoint dot colors.

Value

A Layer.

`geom_embedding` *Embedding scatter (t-SNE / UMAP / PCA)*

Description

A two-dimensional embedding scatter with optional cluster hulls, so cluster shape is visible rather than inferred from point color alone.

Usage

```
geom_embedding(
  mapping = NULL,
  data = NULL,
  hull = TRUE,
  size = NULL,
  alpha = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map the two embedding dimensions to x and y, and the cluster to color.
hull	Draw a convex hull around each cluster.
size	Point radius.
alpha	Point opacity.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  d1 = c(rnorm(30), rnorm(30, 4)), d2 = c(rnorm(30), rnorm(30, 3)),
  cl = rep(c("a", "b"), each = 30)
)
ggnext(d, aes(d1, d2, color = cl)) + geom_embedding()
```

---

geom_errorbar *Error bar layer (*ymin *to* ymax *with caps)*

Description

Error bar layer (ymin to ymax with caps)

Usage

```
geom_errorbar(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL,
  width = NULL
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
linewidth	Stroke width.
width	Cap width in x data/slot units.

Value

A Layer.

---

geom_errorbarh *Horizontal error bars*

Description

The horizontal counterpart of geom_errorbar(), for intervals that run along x — forest plots and tornado charts.

Usage

```
geom_errorbarh(
  mapping = NULL,
  data = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL,
  height = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Requires y, xmin, xmax.
color, linewidth, alpha	Appearance.
height	Cap height in y-axis units.

Value

A Layer to add with +.

Examples

```
d <- data.frame(g = c("a", "b"), lo = c(1, 2), hi = c(4, 5))
ggnext(d, aes(y = g, xmin = lo, xmax = hi)) + geom_errorbarh()
```

geom_forecast_band *Time-series forecast with confidence bands*

Description

Historical actuals as a solid line, the forecast dashed, and one or two nested confidence ribbons.

Usage

```
geom_forecast_band(mapping = NULL, data = NULL, color = NULL, linewidth = NULL)
```

Arguments

mapping, data	Standard layer overrides. Map time to x, the value to y, the interval to ymin/ymax, and the actual-vs-forecast split to group.
color	Line color.
linewidth	Line width.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  t = 1:10, v = c(1:6, 7, 8, 9, 10),
  lo = c(rep(NA, 6), 6, 6.5, 7, 7.5),
  hi = c(rep(NA, 6), 8, 9.5, 11, 12.5),
  part = rep(c("actual", "forecast"), c(6, 4))
)
ggnext(d, aes(t, v, ymin = lo, ymax = hi, group = part)) +
  geom_forecast_band()
```

geom_forest *Forest plot*

Description

Point estimates with confidence intervals down a category axis, with a dashed no-effect reference line — the standard display for hazard/odds ratios, subgroup analyses, and meta-analyses. Marker area encodes study weight when size is mapped.

Usage

```
geom_forest(
  mapping = NULL,
  data = NULL,
  ref = 1,
  color = NULL,
  linewidth = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map the estimate to x, the study/subgroup to y, and the interval to xmin/xmax (via ymin/ymax, which are swapped for you). Map size to weight.
ref	Reference line position: 1 for ratios (default), 0 for mean differences, NA to omit.
color	Marker and whisker color.
linewidth	Whisker width.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  study = c("Trial A", "Trial B", "Trial C", "Pooled"),
  hr = c(0.82, 0.71, 0.95, 0.83),
  lo = c(0.65, 0.52, 0.78, 0.74),
  hi = c(1.03, 0.97, 1.16, 0.93),
  weight = c(30, 22, 28, 100)
)
ggnext(d, aes(hr, study, ymin = lo, ymax = hi, size = weight)) +
  geom_forest() +
  labs(title = "Hazard ratio by trial", x = "Hazard ratio (95% CI)", y = NULL)
```

---

geom_freqpoly *Frequency polygon*

Description

The same binning as geom_histogram(), drawn as a line through the bin centres. Easier to overlay across groups than filled bars.

Usage

```
geom_freqpoly(
  mapping = NULL,
  data = NULL,
  bins = 30,
  binwidth = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides.
`bins`, `binwidth`	Binning, as in `geom_histogram()`.
`color`, `linewidth`, `alpha`	Appearance.

Value

A Layer to add with +.

Examples

```
ggnext(cars, aes(speed)) + geom_freqpoly(bins = 8)
```

---

geom_function *Curve of a function*

Description

Evaluates `fn` over the panel's x range and draws the result. Handy for overlaying a theoretical density or a reference curve on data.

Usage

```
geom_function(
  fn,
  xlim = c(0, 1),
  n = 101,
  color = NULL,
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

`fn`	A function of one numeric argument.
`xlim`	Length-2 range to evaluate over.
`n`	Number of evaluation points.
`color`, `linewidth`, `alpha`	Appearance.

Value

A Layer to add with +.

Examples

```
ggnext(data.frame(x = c(-3, 3)), aes(x)) +
  geom_function(dnorm, xlim = c(-3, 3))
```

---

geom_funnel *Conversion funnel*

Description

Horizontally centered bars, one per stage, whose widths are proportional to the value — the standard conversion/drop-off view.

Usage

```
geom_funnel(
  mapping = NULL,
  data = NULL,
  alpha = NULL,
  label = TRUE,
  label_size = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Map the stage to `x` (ordered top to bottom) and the count to `y`.
`alpha`	Bar opacity.
`label`	Draw the stage label and value inside each bar.
`label_size`	Label size in px.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  stage = factor(c("Visits", "Signups", "Trials", "Paid"),
                 levels = c("Visits", "Signups", "Trials", "Paid")),
  n = c(10000, 3200, 1100, 420)
)
ggnext(d, aes(stage, n, color = stage)) + geom_funnel() + theme_void()
```

`geom_histogram` *Histogram layer*

Description

Histogram layer

Usage

```
geom_histogram(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  bins = 30,
  binwidth = NULL,
  position = "stack"
)
```

Arguments

`mapping`, `data`, `color`, `alpha`	As in `geom_point()`.
`bins`	Number of bins (ignored when `binwidth` is given).
`binwidth`	Bin width in data units.
`position`	`"stack"` (default), `"dodge"`, or `"identity"`.

Value

A Layer.

geom_hline *Horizontal reference line*

Description

Horizontal reference line

Usage

```
geom_hline(
  yintercept,
  color = NULL,
  linewidth = NULL,
  alpha = NULL,
  dash = NULL
)
```

Arguments

`yintercept`	Numeric vector of y positions.
`color`, `linewidth`, `alpha`	Styling.
`dash`	Dash pattern; defaults to `"4,3"` (dashed).

Value

A Layer.

geom_hr *Hazard ratio forest plot (Cox proportional hazards)*

Description

Fits a Cox proportional-hazards model from scratch (Newton-Raphson on the Breslow partial likelihood; see `stat_coxph()`) and plots the resulting hazard ratio and Wald 95% CI per group, down a category axis with a no-effect reference line at HR = 1 — `geom_forest()` paired with the Cox engine so the whole model-to-plot pipeline is one call.

Usage

```
geom_hr(
  mapping = NULL,
  data = NULL,
  ref_level = NULL,
  color = NULL,
  linewidth = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map time, status (1/TRUE = event, 0/FALSE = censored), and the treatment arm to group (or color, which doubles as group).
ref_level	Reference level for the hazard ratio; NULL (default) uses the first observed group.
color	Marker and whisker color.
linewidth	Whisker width.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  time = c(rexp(60, 0.08), rexp(60, 0.05)),
  status = rbinom(120, 1, 0.8),
  arm = rep(c("placebo", "treatment"), each = 60)
)
ggnext(d, aes(time = time, status = status, group = arm)) +
  geom_hr(ref_level = "placebo") +
  labs(title = "Hazard ratio (Cox model)", x = "Hazard ratio (95% CI)", y = NULL)
```

---

geom_importance *Feature importance plot*

Description

A ranked point-and-whisker plot of feature importance, with optional whiskers from ymin/ymax (e.g. permutation-importance standard errors) and a reference line at zero importance. Built on the same point+whisker rendering as geom_forest().

Usage

```
geom_importance(mapping = NULL, data = NULL, color = NULL, linewidth = NULL)
```

Arguments

mapping, data	Standard layer overrides. Map the importance value to x and the feature name to y; map ymin/ymax for whiskers.
color	Marker and whisker color.
linewidth	Whisker width.

Details

A discrete axis otherwise sorts alphabetically; pass y as a factor already ordered by importance (ascending, since the first level sits at the panel's bottom) to get the classic "most important at the top" layout — the same convention geom_waterfall() uses for category order.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  feature = c("age", "income", "tenure", "region"),
  imp = c(0.42, 0.31, 0.18, 0.05),
  se = c(0.05, 0.04, 0.03, 0.02)
)
d$feature <- factor(d$feature, levels = d$feature[order(d$imp)])
ggnext(d, aes(imp, feature, ymin = imp - se, ymax = imp + se)) +
  geom_importance() +
  labs(title = "Permutation importance", x = "Importance", y = NULL)
```

---

geom_jitter *Jittered point layer (strip charts)*

Description

geom_point() with uniform positional noise; deterministic per seed.

Usage

```
geom_jitter(
  mapping = NULL,
  data = NULL,
  color = NULL,
  size = NULL,
  alpha = NULL,
  width = NULL,
  height = NULL,
  seed = 42
)
```

Arguments

mapping, data, color, size, alpha	As in geom_point().
width, height	Jitter half-ranges in data units.
seed	RNG seed.

Value

A Layer.

geom_km *Kaplan-Meier survival curve layer*

Description

Map `time` and `status` (1/TRUE = event, 0/FALSE = censored); map `color` to compare groups. Pair with `geom_km_risktable()` for the classic number-at-risk strip.

Usage

```
geom_km(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL,
  conf_int = FALSE
)
```

Arguments

`mapping`, `data`, `color`, `alpha`	As in `geom_point()`.
`linewidth`	Curve width.
`conf_int`	Draw a 95% confidence band (Greenwood's formula, with a log-log transform that keeps the bounds inside [0, 1]).

Value

A Layer.

Examples

```
set.seed(1)
d <- data.frame(
  t = c(rexp(40, 0.1), rexp(40, 0.07)),
  ev = rbinom(80, 1, 0.7),
  arm = rep(c("placebo", "drug"), each = 40)
)
ggnext(d, aes(time = t, status = ev, color = arm)) +
  geom_km(conf_int = TRUE)
```

`geom_km_risktable` *Number-at-risk table for a Kaplan-Meier curve*

Description

Draws the classic "number at risk" strip beneath a survival curve: for each group, the count of subjects still at risk (`time >= tick`) at a handful of tick times, laid out as an overlay in the bottom ~18% of the panel. It is a single layer, not a separate sub-panel — add it alongside `geom_km()` on the same plot.

Usage

```
geom_km_risktable(
  mapping = NULL,
  data = NULL,
  breaks = NULL,
  label_size = NULL
)
```

Arguments

`mapping, data`	Standard layer overrides. Map `time`, `status`, and (optionally) `color`/`group` exactly as for `geom_km()`.
`breaks`	Tick times to report counts at; `NULL` (default) picks about six round numbers spanning the data with `pretty()`.
`label_size`	Text size in px.

Details

This layer must share the same `time`/`status`/grouping mapping and data as the `geom_km()` layer it accompanies, so the two are trained on the same time domain and its tick columns line up with the curve above them.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  t = c(rexp(40, 0.1), rexp(40, 0.07)),
  ev = rbinom(80, 1, 0.7),
  arm = rep(c("placebo", "drug"), each = 40)
)
ggnext(d, aes(time = t, status = ev, color = arm)) +
  geom_km() +
  geom_km_risktable()
```

geom_label *Text on a background plate*

Description

Like geom_text(), but each label sits on an opaque rounded box, so it stays readable over dense data.

Usage

```
geom_label(
  mapping = NULL,
  data = NULL,
  color = NULL,
  fill = NULL,
  size = NULL,
  alpha = NULL,
  padding = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Requires label.
color	Text and border colour.
fill	Plate fill colour.
size	Text size in px.
alpha	Plate opacity.
padding	Plate padding as a fraction of the text size.

Value

A Layer to add with +.

Examples

```
d <- data.frame(x = c(1, 2), y = c(2, 1), l = c("alpha", "beta"))
ggnext(d, aes(x, y, label = l)) + geom_label()
```

`geom_learning_curve` *Learning curve*

Description

Training and validation score against training-set size or epoch, the standard read on whether a model is data-limited or over-fitting.

Usage

```
geom_learning_curve(
  mapping = NULL,
  data = NULL,
  band = TRUE,
  linewidth = NULL,
  points = TRUE
)
```

Arguments

`mapping, data`	Standard layer overrides. Map size/epoch to `x`, the score to `y`, and the split ("train"/"validation") to `color`.
`band`	Draw a ribbon between `ymin` and `ymax` when supplied.
`linewidth`	Line width.
`points`	Draw markers at each measured point.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  n = rep(c(50, 100, 200, 400), 2),
  score = c(.75, .82, .86, .88, .70, .78, .83, .86),
  split = rep(c("train", "validation"), each = 4)
)
ggnext(d, aes(n, score, color = split)) + geom_learning_curve()
```

`geom_leverage` *Leverage / Cook's distance plot*

Description

Standardized residual against leverage (hat value), the companion to `geom_residual()` for spotting influential observations. Map `size` to Cook's distance for the classic three-way read (leverage, residual size, influence) in one scatter.

Usage

```
geom_leverage(
  mapping = NULL,
  data = NULL,
  color = NULL,
  size = NULL,
  alpha = NULL
)
```

Arguments

`mapping, data`	Standard layer overrides. Map leverage to `x` and the standardized residual to `y`; map `size` to Cook's distance.
`color`	Point color.
`size`	Point radius (used when `size` is not mapped).
`alpha`	Point opacity.

Value

A Layer to add with +.

Examples

```
m <- lm(dist ~ speed, cars)
d <- data.frame(
  hat = hatvalues(m), rstd = rstandard(m), cooksd = cooks.distance(m)
)
ggnext(d, aes(hat, rstd, size = cooksd)) +
  geom_leverage() +
  labs(title = "Leverage vs standardized residual",
       x = "Leverage (hat value)", y = "Standardized residual")
```

`geom_lift_gain` *Lift and cumulative gain curves*

Description

Sorts observations by predicted score and plots the cumulative share of positives captured against the share of the population targeted — the standard way to size a marketing or triage cutoff.

Usage

```
geom_lift_gain(
  mapping = NULL,
  data = NULL,
  type = "gain",
  color = NULL,
  baseline = TRUE
)
```

Arguments

`mapping, data`	Standard layer overrides. Map the score to `score` and the binary outcome to `truth`.
`type`	`"gain"` (cumulative captured) or `"lift"` (ratio to random).
`color`	Curve color.
`baseline`	Draw the random-targeting reference.

Value

A Layer to add with +.

Examples

```
set.seed(1)
s <- runif(200)
d <- data.frame(score = s, y = rbinom(200, 1, s))
ggnext(d, aes(score = score, truth = y)) + geom_lift_gain()
```

geom_line *Line layer (points connected in x order)*

Description

Line layer (points connected in x order)

Usage

```
geom_line(
  mapping = NULL,
  data = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL,
  dash = NULL
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
linewidth	Stroke width in px.
dash	SVG dash pattern (e.g. "4,3"); "" for solid.

Value

A Layer.

geom_linerange *Vertical interval without a marker*

Description

The interval alone; add geom_point() for an estimate marker, or use geom_pointrange() which draws both.

Usage

```
geom_linerange(
  mapping = NULL,
  data = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Requires `x`, `ymin`, `ymax`.
`color`, `linewidth`, `alpha`	Appearance.

Value

A Layer to add with +.

Examples

```
d <- data.frame(g = c("a", "b"), lo = c(1, 2), hi = c(4, 5))
ggnext(d, aes(g, ymin = lo, ymax = hi)) + geom_linerange()
```

`geom_missing_pattern` *Missing-data pattern plot*

Description

A tile grid of which variables are observed vs. missing, one row per *distinct missingness pattern* (most frequent at the top, annotated with how many observations share it) rather than one row per observation — the standard compact view for spotting structured missingness (e.g. a block of variables always missing together).

Usage

```
geom_missing_pattern(
  data,
  vars = NULL,
  label = TRUE,
  label_size = NULL,
  observed = "#D8DCE6",
  missing = "#C1462F"
)
```

Arguments

`data`	A data frame to check for missing values.
`vars`	Character vector of columns to include; default all columns in `data`.
`label`	Annotate missing cells.
`label_size`	Annotation text size in px.
`observed`, `missing`	Tile fill colors for observed vs. missing cells.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  age = c(25, NA, 30, 40, NA, 33),
  bmi = c(22, 24, NA, NA, 27, 23),
  sbp = c(120, 118, 130, NA, 125, 121)
)
ggnext() + geom_missing_pattern(d) + labs(title = "Missing-data patterns")
```

---

geom_nelson_aalen *Nelson-Aalen cumulative hazard curve layer*

Description

Map `time` and `status` (1/TRUE = event, 0/FALSE = censored) exactly as for `geom_km()`; map `color` to compare groups. The cumulative hazard H(t) is the additive counterpart of the Kaplan-Meier survival curve — useful when the *rate* of events, not the surviving fraction, is what matters.

Usage

```
geom_nelson_aalen(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL
)
```

Arguments

`mapping`, `data`, `color`, `alpha`	As in `geom_point()`.
`linewidth`	Curve width.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(t = rexp(80, 0.08), ev = rbinom(80, 1, 0.75))
ggnext(d, aes(time = t, status = ev)) +
  geom_nelson_aalen() +
  labs(title = "Cumulative hazard", x = "Time", y = "H(t)")
```

`geom_network` *Network (node-link) diagram*

Description

Lays out a graph with a from-scratch Fruchterman-Reingold force simulation (repulsion between all nodes, attraction along edges, with a cooling schedule) and draws nodes and edges in one unified grammar — node and edge aesthetics come from the same `aes()`.

Usage

```
geom_network(
  mapping = NULL,
  data = NULL,
  node_size = NULL,
  edge_color = NULL,
  edge_width = NULL,
  label = TRUE,
  label_size = NULL,
  alpha = NULL,
  iterations = 200,
  seed = 1
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Requires `x` (source node) and `xend` (target node); optional `size` weights the edges.
`node_size`	Node radius in px.
`edge_color`, `edge_width`	Edge appearance.
`label`, `label_size`	Node labels.
`alpha`	Node opacity.
`iterations`	Force-simulation steps; more is slower but tidier.
`seed`	Random seed for the initial layout, so plots reproduce.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  from = c("A", "A", "B", "C", "D", "E"),
  to = c("B", "C", "C", "D", "E", "A")
)
ggnext(d, aes(x = from, xend = to)) + geom_network() + theme_void()
```

`geom_parallel` *Parallel coordinates plot*

Description

Each observation is a line crossing one vertical axis per variable — the standard way to eyeball high-dimensional structure and clusters. Each axis is independently rescaled to [0, 1] by the stat, so variables in different units are comparable.

Usage

```
geom_parallel(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Map the variable to `x`, the value to `y`, and the observation id to `group`.
`color`, `alpha`, `linewidth`	Appearance.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  id = rep(1:3, each = 3),
  var = rep(c("a", "b", "c"), 3),
  val = c(1, 9, 4, 3, 5, 8, 7, 2, 6)
)
ggnext(d, aes(var, val, group = id, color = factor(id))) + geom_parallel()
```

`geom_partial_dependence` *Partial dependence and ICE curves*

Description

A thick average partial-dependence line over thin per-observation ICE curves. Map `group` to the observation id to draw the ICE spaghetti.

Usage

```
geom_partial_dependence(
  mapping = NULL,
  data = NULL,
  color = NULL,
  ice = TRUE,
  ice_alpha = 0.25,
  linewidth = NULL
)
```

Arguments

`mapping, data`	Standard layer overrides. Map the feature to `x` and the prediction to `y`; map `group` for ICE curves.
`color`	Curve color.
`ice`	Draw the individual ICE curves under the average.
`ice_alpha`	Opacity of the ICE curves.
`linewidth`	Width of the average line.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  x = rep(1:10, 5), id = rep(1:5, each = 10),
  pred = as.vector(sapply(1:5, function(i) (1:10) * 0.1 * i + rnorm(10, 0, .1)))
)
ggnext(d, aes(x, pred, group = id)) + geom_partial_dependence()
```

geom_path *Path layer (points connected in data order)*

Description

Path layer (points connected in data order)

Usage

```
geom_path(
  mapping = NULL,
  data = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL,
  dash = NULL
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
linewidth	Stroke width in px.
dash	SVG dash pattern (e.g. "4,3"); "" for solid.

Value

A Layer.

geom_point *Scatter-plot layer: one point per observation*

Description

Draws a circle mark for each row of the layer data - the default view of a relationship between two continuous variables.

Usage

```
geom_point(
  mapping = NULL,
  data = NULL,
  stat = stat_identity(),
  color = NULL,
  size = NULL,
  alpha = NULL
)
```

Arguments

mapping	Layer-specific aesthetic mapping (optional; merged over the plot-level mapping, layer winning on conflicts).
data	Layer-specific data frame (optional).
stat	A Stat instance; defaults to stat_identity().
color	Literal point color (any R color spec) when color is not a mapped aesthetic.
size	Literal point radius in pixels when size is not mapped.
alpha	Point opacity in [0, 1].

Value

A Layer object to add to a plot with +.

Examples

```
ggnext(cars, aes(speed, dist)) + geom_point(color = "steelblue", size = 4)
```

---

geom_pointrange *Point-range layer (interval plus midpoint; forest-plot building block)*

Description

A horizontal forest plot is this geom with the categorical variable on x — a dedicated geom_forest() with flipped coordinates is a milestone.

Usage

```
geom_pointrange(
  mapping = NULL,
  data = NULL,
  color = NULL,
  size = NULL,
  alpha = NULL,
  linewidth = NULL
)
```

Arguments

mapping, data, color, size, alpha	As in geom_point().
linewidth	Stroke width.

Value

A Layer.

`geom_polygon` *Polygons*

Description

Joins the points of each group into a closed shape, in data order. Map `group` when the data holds several polygons.

Usage

```
geom_polygon(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL,
  border = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides.
`color`	Fill colour when `color` is not mapped.
`alpha`	Fill opacity.
`linewidth`	Outline width in px; `0` (default) draws no outline.
`border`	Outline colour when `linewidth` > `0`.

Value

A Layer to add with +.

Examples

```
d <- data.frame(x = c(1, 3, 2), y = c(1, 1, 3))
ggnext(d, aes(x, y)) + geom_polygon()
```

geom_pr *Precision-recall curve*

Description

Sweeps the classification threshold and plots recall (x) against precision (y), with a horizontal baseline at the class prevalence (instead of the diagonal `geom_roc()` uses) — the more informative curve when positives are rare.

Usage

```
geom_pr(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL
)
```

Arguments

`mapping, data`	Standard layer overrides. Map `truth` (actual class; for factors the second level is the positive class) and `score` (predicted score); map `color` to compare models.
`color, alpha`	As in `geom_point()`.
`linewidth`	Curve width.

Value

A Layer to add with +.

Examples

```
set.seed(1)
s <- runif(200)
d <- data.frame(truth = rbinom(200, 1, s * 0.3), score = s)
ggnext(d, aes(truth = truth, score = score)) + geom_pr()
```

geom_qq *Quantile-quantile plot*

Description

Sample quantiles against the quantiles of a reference distribution; points on a straight line mean the sample follows it. Add `geom_qq_line()` for the reference.

Usage

```
geom_qq(
  mapping = NULL,
  data = NULL,
  distribution = stats::qnorm,
  color = NULL,
  size = NULL,
  alpha = NULL
)

geom_qq_line(
  mapping = NULL,
  data = NULL,
  distribution = stats::qnorm,
  color = NULL,
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map the values to x.
distribution	Quantile function of the reference distribution (default stats::qnorm).
color, size, alpha	Appearance.
linewidth	Line width for the reference line.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(v = rnorm(100))
ggnext(d, aes(v)) + geom_qq() + geom_qq_line()
```

---

geom_quantile *Quantile regression lines*

Description

Fits a linear model to chosen conditional quantiles, showing how the spread of y changes with x — not just its mean, as geom_smooth() does.

Usage

```
geom_quantile(
  mapping = NULL,
  data = NULL,
  quantiles = c(0.25, 0.5, 0.75),
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

mapping, data	Standard layer overrides.
quantiles	Quantiles to fit (default the quartiles).
linewidth, alpha	Appearance.

Value

A Layer to add with +.

Examples

```
ggnext(cars, aes(speed, dist)) +
  geom_point(alpha = 0.5) +
  geom_quantile()
```

---

geom_radar *Radar (spider / star) chart*

Description

Draws one closed polygon per group across a set of categorical axes. Pair with coord_polar() for the familiar circular form; without it the same layer reads as a parallel-coordinates profile.

Usage

```
geom_radar(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL,
  points = TRUE
)
```

Arguments

mapping, data	Standard layer overrides. Map the axes to x, the values to y, and the series to color (or group).
color	Line/fill color when color is not mapped.
alpha	Fill opacity of the polygon interior.
linewidth	Outline width in px.
points	Draw a marker at each vertex.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  axis = rep(c("Speed", "Power", "Range", "Cost", "Safety"), 2),
  value = c(8, 6, 7, 4, 9, 5, 9, 4, 8, 6),
  model = rep(c("A", "B"), each = 5)
)
ggnext(d, aes(axis, value, color = model)) +
  geom_radar() +
  coord_polar() +
  theme_minimal()
```

---

geom_raster *Raster*

Description

A tile grid drawn without cell borders — the right choice for a dense heatmap or an image, where borders would swamp the data.

Usage

```
geom_raster(mapping = NULL, data = NULL, color = NULL, alpha = NULL)
```

Arguments

mapping, data, color, alpha	As in geom_point().

Value

A Layer to add with +.

Examples

```
g <- expand.grid(x = 1:20, y = 1:20)
g$z <- as.vector(outer(1:20, 1:20, function(a, b) sin(a / 3) * cos(b / 3)))
ggnext(g, aes(x, y, color = z)) + geom_raster()
```

---

geom_rect *Rectangles*

Description

Draws an axis-aligned rectangle per row from explicit corners. Use it for highlight bands, annotation boxes, and any hand-placed block; for a regular grid shaded by value use `geom_tile()`.

Usage

```
geom_rect(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL,
  border = NULL
)
```

Arguments

`mapping, data`	Standard layer overrides. Requires `xmin`, `xmax`, `ymin` and `ymax`.
`color`	Fill colour when `color` is not mapped.
`alpha`	Fill opacity.
`linewidth`	Border width in px; `0` (default) draws no border.
`border`	Border colour when `linewidth > 0`.

Value

A Layer to add with +.

Examples

```
d <- data.frame(x1 = c(1, 3), x2 = c(2, 5), y1 = c(1, 2), y2 = c(4, 3))
ggnext(d, aes(xmin = x1, xmax = x2, ymin = y1, ymax = y2)) + geom_rect()
```

`geom_residual` *Residual diagnostic plot*

Description

Residuals against fitted values with a zero reference line and a loess trend, the first plot to look at when checking a linear model.

Usage

```
geom_residual(
  mapping = NULL,
  data = NULL,
  color = NULL,
  smooth = TRUE,
  size = NULL,
  alpha = NULL
)
```

Arguments

`mapping, data`	Standard layer overrides. Map fitted values to `x` and residuals to `y`.
`color`	Point color.
`smooth`	Overlay a loess trend through the residuals.
`size`	Point radius.
`alpha`	Point opacity.

Value

A Layer to add with `+`.

Examples

```
m <- lm(dist ~ speed, cars)
d <- data.frame(fitted = fitted(m), resid = resid(m))
ggnext(d, aes(fitted, resid)) + geom_residual()
```

geom_ribbon *Ribbon layer (band between* ymin *and* ymax*)*

Description

Ribbon layer (band between ymin and ymax)

Usage

```
geom_ribbon(mapping = NULL, data = NULL, color = NULL, alpha = NULL)
```

Arguments

mapping, data, color, alpha	As in geom_point().

Value

A Layer.

geom_ridgeline *Ridgeline plot (joyplot)*

Description

One density curve per group, offset vertically so distributions can be compared at a glance. Groups are ordered by their factor levels, first level at the bottom.

Usage

```
geom_ridgeline(
  mapping = NULL,
  data = NULL,
  scale = 1.6,
  color = NULL,
  alpha = NULL,
  linewidth = NULL,
  bw = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map the value to x and the group to y (or color).
scale	Height of each ridge as a multiple of the row spacing; values above 1 make ridges overlap.
color, alpha, linewidth	Appearance.
bw	Density bandwidth; NULL uses a Silverman rule of thumb.

Value

A Layer to add with +.

Examples

```
ggnext(iris, aes(Sepal.Length, Species)) + geom_ridgeline()
```

`geom_roc` *ROC curve layer*

Description

Map `truth` (actual class; for factors the second level is the positive class) and `score` (predicted score); map `color` to compare models.

Usage

```
geom_roc(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  linewidth = NULL
)
```

Arguments

`mapping`, `data`, `color`, `alpha`	As in `geom_point()`.
`linewidth`	Curve width.

Value

A Layer.

geom_rug *Marginal rug*

Description

A short tick per observation along the panel edges, showing the raw values behind a density, smooth, or scatter.

Usage

```
geom_rug(
  mapping = NULL,
  data = NULL,
  sides = NULL,
  length = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides.
`sides`	Which edges to draw on, as a string of `"t"`, `"r"`, `"b"`, `"l"` (default `"bl"`).
`length`	Tick length as a fraction of the panel.
`color`, `linewidth`, `alpha`	Appearance.

Value

A Layer to add with +.

Examples

```
ggnext(cars, aes(speed, dist)) + geom_point() + geom_rug()
```

geom_sankey *Sankey / alluvial flow diagram*

Description

Shows how quantities move between stages. Nodes are drawn as bars at each stage and flows as curved ribbons whose thickness is the value.

Usage

```
geom_sankey(
  mapping = NULL,
  data = NULL,
  alpha = NULL,
  node_width = NULL,
  label = TRUE
)
```

Arguments

mapping, data	Standard layer overrides. Requires x (source node), xend (target node), and y (flow value).
alpha	Ribbon opacity.
node_width	Node bar width as a fraction of the panel.
label	Draw node labels.

Details

The layout (node positions, ribbon paths) is computed from scratch: nodes are stacked within their stage in decreasing size, and ribbons leave and enter nodes in the same order, which is what keeps the crossings readable.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  from = c("Visited", "Visited", "Signed up", "Signed up"),
  to = c("Signed up", "Left", "Purchased", "Churned"),
  n = c(400, 600, 150, 250)
)
ggnext(d, aes(x = from, xend = to, y = n)) +
  geom_sankey() +
  theme_void()
```

---

geom_segment *Segment layer (x,y) to (xend,yend)*

Description

Segment layer (x,y) to (xend,yend)

Usage

```
geom_segment(
  mapping = NULL,
  data = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
linewidth	Stroke width in px.

Value

A Layer.

---

geom_shap *SHAP value beeswarm*

Description

One horizontal swarm per feature showing every observation's SHAP value, with points nudged vertically so overlapping values stay countable. Color the points by the feature value to read the direction of effect.

Usage

```
geom_shap(mapping = NULL, data = NULL, size = NULL, alpha = NULL)
```

Arguments

mapping, data	Standard layer overrides. Map the SHAP value to x and the feature name to y; map color to the feature value.
size	Point radius in px.
alpha	Point opacity.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  feature = rep(c("age", "income", "tenure"), each = 40),
  shap = c(rnorm(40, 0.3, 0.2), rnorm(40, -0.1, 0.3), rnorm(40, 0, 0.15)),
  value = runif(120)
)
ggnext(d, aes(shap, feature, color = value)) +
  geom_shap() +
  geom_vline(0, dash = "3,3") +
  labs(title = "SHAP value by feature", x = "SHAP value", y = NULL)
```

---

geom_shift *Categorical shift plot*

Description

How subjects move between categories from baseline to follow-up (lab grade shifts, response categories), drawn as a counted tile grid.

Usage

```
geom_shift(mapping = NULL, data = NULL, label = TRUE)
```

Arguments

mapping, data	Standard layer overrides. Map the baseline category to x and the follow-up category to y.
label	Annotate cells with counts.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  baseline = c("G0", "G0", "G1", "G1", "G2", "G0"),
  followup = c("G0", "G1", "G1", "G2", "G2", "G0")
)
ggnext(d, aes(baseline, followup)) + geom_shift()
```

geom_silhouette *Clustering silhouette plot*

Description

Sorted silhouette widths grouped by cluster, with the mean marked — the standard visual check on how well-separated a clustering is.

Usage

```
geom_silhouette(mapping = NULL, data = NULL, alpha = NULL)
```

Arguments

mapping, data	Standard layer overrides. Map the silhouette width to x and the cluster to y (or color).
alpha	Bar opacity.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  cluster = rep(c("1", "2", "3"), each = 20),
  width = c(runif(20, .3, .9), runif(20, .1, .7), runif(20, -.1, .6))
)
ggnext(d, aes(width, cluster, color = cluster)) + geom_silhouette()
```

geom_smd *Baseline covariate balance (standardized mean difference) plot*

Description

The standardized mean difference (Cohen's d) between two groups for each covariate, down a category axis with a reference line at SMD = 0 and a dashed line at the conventional 0.1 "acceptable imbalance" threshold — the standard baseline balance check for a randomized or propensity-matched comparison.

Usage

```
geom_smd(data, group, vars = NULL, threshold = 0.1, color = NULL)
```

Arguments

data	A data frame with one row per subject: a two-level group column and one or more numeric covariate columns.
group	Name of the two-level grouping column (treatment arm).
vars	Character vector of covariate columns; default all numeric columns other than group.
threshold	Dashed-line threshold in SMD units; NULL or 0 omits it.
color	Marker and whisker color.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  arm = rep(c("treatment", "control"), each = 50),
  age = c(rnorm(50, 55, 8), rnorm(50, 58, 9)),
  bmi = c(rnorm(50, 27, 4), rnorm(50, 27.5, 4)),
  sbp = c(rnorm(50, 130, 12), rnorm(50, 129, 11))
)
ggnext() +
  geom_smd(d, group = "arm") +
  labs(title = "Baseline balance", x = "Standardized mean difference", y = NULL)
```

---

geom_smooth *Smoothed trend layer*

Description

Smoothed trend layer

Usage

```
geom_smooth(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  method = "loess",
  se = TRUE,
  linewidth = NULL,
  level = 0.95
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
method	"loess" (default) or "lm".
se	Draw the confidence band.
linewidth	Trend line width.
level	Confidence level.

Value

A Layer.

---

geom_spaghetti *Individual longitudinal trajectories (spaghetti plot)*

Description

One thin line per subject with an optional bold group mean overlaid — the standard way to show within-subject change without hiding spread.

Usage

```
geom_spaghetti(
  mapping = NULL,
  data = NULL,
  mean_line = TRUE,
  alpha = NULL,
  linewidth = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map time to x, the measurement to y, and the subject to group.
mean_line	Overlay the group mean trajectory.
alpha	Opacity of the individual lines.
linewidth	Width of the individual lines.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  week = rep(0:4, 8), id = rep(1:8, each = 5),
  score = as.vector(sapply(1:8, function(i) 50 + i + (0:4) * 2 + rnorm(5, 0, 3)))
)
ggnext(d, aes(week, score, group = id)) + geom_spaghetti()
```

---

geom_spider_response *Oncology spider plot*

Description

Percent change from baseline over time, one trajectory per subject, with the +20% (progression) and -30% (response) RECIST thresholds marked. Distinct from `geom_radar()`, which is a multivariate star chart.

Usage

```
geom_spider_response(
  mapping = NULL,
  data = NULL,
  thresholds = TRUE,
  linewidth = NULL,
  points = TRUE
)
```

Arguments

`mapping, data`	Standard layer overrides. Map time to `x`, percent change to `y`, and the subject to `group` (or `color`).
`thresholds`	Draw the RECIST +20% / -30% reference lines.
`linewidth`	Line width.
`points`	Draw a marker at each visit.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  month = rep(c(0, 2, 4, 6), 3),
  pct = c(0, -20, -35, -40, 0, 10, 25, 40, 0, -5, -10, -8),
  subject = rep(c("S1", "S2", "S3"), each = 4)
)
ggnext(d, aes(month, pct, color = subject)) + geom_spider_response()
```

geom_spoke *Spokes (vector field)*

Description

A segment from each point at a given angle and radius — the usual way to draw a vector or wind field.

Usage

```
geom_spoke(
  mapping = NULL,
  data = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL
)
```

Arguments

`mapping`, `data`	Standard layer overrides. Requires `x`, `y`, and the `angle` (radians) and `radius` columns supplied via `xend`/`yend`.
`color`, `linewidth`, `alpha`	Appearance.

Value

A Layer to add with +.

Examples

```
g <- expand.grid(x = 1:5, y = 1:5)
g$angle <- atan2(g$y - 3, g$x - 3)
g$radius <- 0.4
ggnext(g, aes(x, y, xend = angle, yend = radius)) + geom_spoke()
```

geom_step *Step layer (staircase line)*

Description

Step layer (staircase line)

Usage

```
geom_step(
  mapping = NULL,
  data = NULL,
  color = NULL,
  linewidth = NULL,
  alpha = NULL,
  dash = NULL
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
linewidth	Stroke width in px.
dash	SVG dash pattern (e.g. "4,3"); "" for solid.

Value

A Layer.

---

geom_stream *Streamgraph*

Description

Stacked areas centered on a wiggle baseline rather than zero, which keeps every band's thickness readable as it changes over time.

Usage

```
geom_stream(mapping = NULL, data = NULL, alpha = NULL)
```

Arguments

mapping, data	Standard layer overrides. Map time to x, value to y, and the series to color.
alpha	Band opacity.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  t = rep(1:6, 3),
  v = c(2, 4, 6, 5, 3, 2, 1, 3, 5, 8, 6, 4, 5, 4, 3, 4, 6, 7),
  grp = rep(c("a", "b", "c"), each = 6)
)
ggnext(d, aes(t, v, color = grp)) + geom_stream() + theme_minimal()
```

---

geom_swimmer *Swimmer plot*

Description

One horizontal bar per subject showing time on treatment, with optional arrowheads for subjects still ongoing at the data cutoff — the standard patient-level timeline in oncology reporting.

Usage

```
geom_swimmer(
  mapping = NULL,
  data = NULL,
  bar_height = 0.6,
  arrow = TRUE,
  alpha = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Map duration to x, the subject to y, and (optionally) response category to color. Map label to a logical "still ongoing" flag to draw arrowheads.
bar_height	Bar thickness in category-slot units.
arrow	Draw arrowheads for ongoing subjects.
alpha	Bar opacity.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  subject = paste0("S", 1:6),
  months = c(4, 9, 14, 6, 20, 11),
  response = c("PR", "CR", "CR", "SD", "PR", "SD"),
  ongoing = c(FALSE, FALSE, TRUE, FALSE, TRUE, FALSE)
)
```

```
ggnext(d, aes(months, subject, color = response, label = ongoing)) +
  geom_swimmer() +
  labs(title = "Time on treatment", x = "Months", y = NULL)
```

`geom_text` *Text label layer*

Description

Text label layer

Usage

```
geom_text(
  mapping = NULL,
  data = NULL,
  color = NULL,
  fontsize = NULL,
  alpha = NULL,
  anchor = NULL
)
```

Arguments

`mapping`, `data`, `color`, `alpha`	As in `geom_point()`; map `label`.
`fontsize`	Font size in px.
`anchor`	`"middle"`, `"start"`, or `"end"`.

Value

A Layer.

`geom_tile` *Tile layer (heatmap cells)*

Description

Map a continuous `color` for the classic heatmap look.

Usage

```
geom_tile(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  width = NULL,
  height = NULL
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
width, height	Tile size in data units (default: data resolution).

Value

A Layer.

---

geom_treemap *Treemap*

Description

Area-proportional nested rectangles, laid out with the squarified algorithm (Bruls, Huizing & van Wijk 2000) so tiles stay close to square and stay comparable by area.

Usage

```
geom_treemap(
  mapping = NULL,
  data = NULL,
  alpha = NULL,
  label = TRUE,
  label_size = NULL
)
```

Arguments

mapping, data	Standard layer overrides. Requires size (the area value) and label (the tile name); map color to shade by category.
alpha	Tile opacity.
label	Draw tile labels where they fit.
label_size	Label size in px.

Value

A Layer to add with +.

Examples

```
d <- data.frame(
  region = c("North", "South", "East", "West", "Central"),
  revenue = c(52, 38, 27, 19, 11)
)
ggnext(d, aes(size = revenue, label = region, color = region)) +
  geom_treemap() +
  theme_void()
```

---

geom_upset *UpSet plot (set intersections)*

Description

Replaces unreadable 4+ way Venn diagrams: a bar chart of intersection sizes above a dot matrix showing which sets each bar belongs to.

Usage

```
geom_upset(mapping = NULL, data = NULL, alpha = NULL, dot_size = NULL)
```

Arguments

mapping, data	Standard layer overrides. Requires label (the set membership of each observation, e.g. "A&B").
alpha	Bar opacity.
dot_size	Membership-dot radius in px.

Value

A Layer to add with +.

Examples

```
d <- data.frame(sets = c("A", "A&B", "B", "A&B&C", "C", "A&B", "A"))
ggnext(d, aes(label = sets)) + geom_upset() + theme_void()
```

geom_violin *Violin layer*

Description

Violin layer

Usage

```
geom_violin(
  mapping = NULL,
  data = NULL,
  color = NULL,
  alpha = NULL,
  width = 0.9
)
```

Arguments

mapping, data, color, alpha	As in geom_point().
width	Maximum violin width in x slot units.

Value

A Layer.

geom_vline *Vertical reference line*

Description

Vertical reference line

Usage

```
geom_vline(
  xintercept,
  color = NULL,
  linewidth = NULL,
  alpha = NULL,
  dash = NULL
)
```

Arguments

`xintercept`	Numeric vector of x positions.
`color`, `linewidth`, `alpha`	Styling.
`dash`	Dash pattern; defaults to `"4,3"` (dashed).

Value

A Layer.

`geom_waterfall` *Waterfall chart layer (running total of signed changes)*

Description

Categories plot in level order; pass `x = factor(x, levels = unique(x))` to keep them in data order.

Usage

```
geom_waterfall(
  mapping = NULL,
  data = NULL,
  alpha = NULL,
  width = 0.8,
  color_pos = NULL,
  color_neg = NULL
)
```

Arguments

`mapping`, `data`, `alpha`	As in `geom_point()`.
`width`	Bar width in x slot units.
`color_pos`, `color_neg`	Bar colors for increases/decreases.

Value

A Layer.

`geom_waterfall_response` *RECIST best-response waterfall*

Description

Subjects ordered by best percent change from baseline, one bar each, with the +20% / -30% RECIST thresholds marked. Distinct from `geom_waterfall()`, which shows a running total.

Usage

```
geom_waterfall_response(
  mapping = NULL,
  data = NULL,
  thresholds = TRUE,
  alpha = NULL
)
```

Arguments

`mapping, data`	Standard layer overrides. Map the subject to `x` (or omit and let ordering supply it) and percent change to `y`; map `color` to response category.
`thresholds`	Draw the RECIST reference lines.
`alpha`	Bar opacity.

Value

A Layer to add with +.

Examples

```
set.seed(1)
d <- data.frame(
  subject = paste0("S", 1:20),
  pct = sort(runif(20, -75, 45), decreasing = TRUE)
)
ggnext(d, aes(subject, pct)) +
  geom_waterfall_response() +
  labs(title = "Best response", y = "% change from baseline", x = NULL)
```

ggnext *Create a new ggnext plot*

Description

The entry point of the grammar: bind a default data frame and a default aesthetic mapping, then add layers, scales, and coordinate systems with +.

Usage

```
ggnext(data = NULL, mapping = NULL, width = 640, height = 480)
```

Arguments

data	A data frame used by all layers unless a layer overrides it.
mapping	Default aesthetic mapping created with aes().
width, height	Device size in pixels (default 640 x 480).

Value

A GgnextPlot object.

Examples

```
p <- ggnext(cars, aes(speed, dist)) + geom_point()
svg <- render(p)
```

GgnextPlot *GgnextPlot: the complete plot specification*

Description

Built by ggnext() and grown with +. Holds everything needed to compute geometry; rendering never reaches back past the computed buffer.

Usage

```
GgnextPlot(
  data = NULL,
  mapping = NULL,
  layers = list(),
  scales = list(),
  coord = Coord(),
  interaction = NULL,
  theme = NULL,
```

```
  labels = NULL,
  color_scale = NULL,
  facet = NULL,
  animation = NULL,
  size = NULL
)
```

Arguments

data	Default data frame for all layers.
mapping	Default aesthetic mapping from aes().
layers	List of Layer objects.
scales	Named list of Scale objects, keyed by aesthetic.
coord	A Coord subclass instance.
interaction	An Interact spec, or NULL for a static plot.
theme	A Theme object controlling the plot chrome.
labels	A Labels object with the title block and axis titles.
color_scale	A ColorScale palette override, or NULL.
facet	A Facet spec, or NULL for a single panel.
animation	An Animation spec, or NULL.
size	Device size in pixels, c(width, height).

Value

An S7 object of class GgnextPlot, as returned by ggnext().

---

ggnext_logo *Draw the ggnext hex sticker*

Description

Generates the package logo as an SVG: a hexagon containing a small scatter with a fitted trend, drawn with the same coordinate and color machinery the package uses for real plots.

Usage

```
ggnext_logo(
  file = NULL,
  width = 520,
  dark = FALSE,
  style = c("wordmark", "monogram"),
  tagline = c("NEXT-GENERATION", "GRAMMAR OF GRAPHICS")
)
```

Arguments

file	Output path; NULL returns the SVG as a string.
width	Sticker width in pixels (height follows the hex ratio).
dark	Use the dark palette variant.
style	"wordmark" or "monogram".
tagline	Text under the wordmark, as one string or a character vector of lines (two read better than one long line). "" omits it. Ignored by the monogram style, which shows ggnext there instead.

Details

Two styles are available. "wordmark" (the default) spells out ggnext under the artwork with a tagline; "monogram" sets a large GG in the middle, which stays legible at favicon and app-icon sizes where a seven-character wordmark turns to mush.

Value

The SVG document as a length-1 character vector, invisibly when file is supplied.

Examples

```
svg <- ggnext_logo()
substr(svg, 1, 30)

# App-icon variant.
icon <- ggnext_logo(style = "monogram", width = 256)
```

---

ggtitle *Set the plot title (and optionally the subtitle)*

Description

Set the plot title (and optionally the subtitle)

Usage

```
ggtitle(label, subtitle = NULL)
```

Arguments

label	Title text.
subtitle	Optional subtitle text.

Value

A Labels object to add to a plot with +.

Interact *Add interactivity to a plot*

Description

Plots are static-first: `render(p)` and printing produce a static SVG image. Adding `interact()` with `+` opts the plot into the interactive HTML/canvas target — `render(p)` then produces the interactive page instead (an explicit `render(p, target = )` always wins).

Usage

```
Interact(tooltip = NULL, zoom = logical(0), brush = logical(0))

interact(tooltip = TRUE, zoom = TRUE, brush = TRUE)
```

Arguments

`tooltip`	`TRUE` (default) shows the mapped x/y values on hover; `FALSE` disables the tooltip; a character vector of column names (e.g. `tooltip = c("model", "hwy")`) shows those columns from the layer data instead.
`zoom`	Enable scroll-to-zoom and double-click-to-reset.
`brush`	Enable brush-to-zoom: drag a rectangle on the plot to zoom to it (double-click still resets).

Value

An Interact spec to add to a plot with `+`.

Examples

```
p <- ggnext(cars, aes(speed, dist)) + geom_point()
render(p)                 # static SVG
pi <- p + interact()      # same plot, interactive by default
html <- render(pi)        # HTML/canvas with tooltip + zoom + brush
```

Labels *Labels: plot title block and axis/legend titles*

Description

Created by `labs()`, `ggtitle()`, `xlab()`, `ylab()`; added with `+`. Any field left `NULL` falls back to the default (for axes, the deparsed aesthetic mapping).

Usage

```
Labels(
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  tag = NULL,
  x = NULL,
  y = NULL,
  color = NULL,
  size = NULL,
  fill = NULL
)
```

Arguments

`title`, `subtitle`, `caption`, `tag`	Plot title block text.
`x`, `y`	Axis titles.
`color`, `size`, `fill`	Legend titles.

Value

An S7 object of class `Labels`; usually built by `labs()`.

`labs` *Set plot and axis labels*

Description

Every argument is optional; only the ones you supply are changed, so `labs()` can be added repeatedly to build up a title block.

Usage

```
labs(
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  tag = NULL,
  x = NULL,
  y = NULL,
  color = NULL,
  size = NULL,
  fill = NULL
)
```

Arguments

`title`	Plot title, drawn above the panel.
`subtitle`	Subtitle, drawn under the title in a lighter style.
`caption`	Caption, drawn bottom-right (source notes, methods).
`tag`	Panel tag (e.g. `"A"`), drawn top-left — for multi-panel figures in publications.
`x`, `y`	Axis titles; override the deparsed `aes()` expression.
`color`, `size`, `fill`	Legend titles for the matching aesthetic.

Value

A Labels object to add to a plot with +.

Examples

```
ggnext(cars, aes(speed, dist)) +
  geom_point() +
  labs(
    title = "Stopping distance rises with speed",
    subtitle = "1920s road tests, 50 observations",
    x = "Speed (mph)",
    y = "Stopping distance (ft)",
    caption = "Source: datasets::cars"
  )
```

`Layer` *Layer: one geom + stat + data/mapping overrides*

Description

Layer: one geom + stat + data/mapping overrides

Usage

```
Layer(
  geom = Geom(),
  stat = Stat(),
  mapping = NULL,
  data = NULL,
  params = list(),
  inherit = TRUE
)
```

Arguments

geom	A Geom subclass instance.
stat	A Stat subclass instance.
mapping	Layer-level aesthetic mapping (or NULL to inherit).
data	Layer-level data (or NULL to inherit the plot data).
params	Named list of literal visual parameters (e.g. size = 4).
inherit	Whether the layer merges the plot-level mapping into its own (TRUE for data layers; reference-line layers that synthesize their own data set FALSE).

Value

An S7 object of class Layer, as returned by the geom_*() constructors.

---

lims *Set both axis limits at once*

Description

Set both axis limits at once

Usage

```
lims(x = NULL, y = NULL)
```

Arguments

x	Length-2 numeric vector for the x axis, or NULL.
y	Length-2 numeric vector for the y axis, or NULL.

Value

A list of scales, which + adds one at a time.

plot_check — *Print a plot's validation findings and return the plot, unchanged*

Description

A pipe-friendly wrapper around `validate_plot()`: prints the report as a side effect and returns `plot` invisibly, so it can sit inside a pipeline without interrupting it.

Usage

```
plot_check(plot)
```

Arguments

`plot`	A GgnextPlot object.

Value

`plot`, invisibly.

Examples

```
ggnext(cars, aes(speed, dist)) |>
  geom_point() |>
  plot_check() |>
  invisible()
```

plot_data — *Extract the exact data a plot draws*

Description

Returns the computed values behind each layer — post-stat, post-position, post-facet — in data units (never normalized coordinates). For a `geom_point()` layer that is the x/y you supplied; for `geom_histogram()` it is the bin centers, counts, and bin edges actually drawn; for `geom_boxplot()` the quartiles and whisker ends.

Usage

```
plot_data(plot, layer = NULL, panel = NULL)
```

Arguments

plot	A GgnextPlot.
layer	Which layer to return: an integer index, or NULL (the default) for a list of all layers. With one layer, the data frame itself is returned rather than a one-element list.
panel	Which facet panel to return: an integer index, NULL (the default) for all panels combined with a panel column, or "list" to get a list per panel.

Details

Only columns the layer genuinely uses are returned. Constant grouping columns are dropped, so a plot with no group aesthetic has no group column.

Value

A data frame, or a list of data frames.

Examples

```
p <- ggnext(cars, aes(speed, dist)) + geom_point()
head(plot_data(p))

# Stats: what geom_histogram() actually drew.
h <- ggnext(cars, aes(speed)) + geom_histogram(bins = 5)
plot_data(h)

# Facets keep a panel column.
f <- ggnext(iris, aes(Sepal.Length, Sepal.Width)) +
  geom_point() + facet_wrap(Species)
head(plot_data(f))
```

---

plot_size *Set the output size*

Description

Set the output size

Usage

```
plot_size(width, height)
```

Arguments

width, height	Device size in pixels.

Value

A ggnext_size object to add to a plot with +.

Examples

```
ggnext(cars, aes(speed, dist)) + geom_point() + plot_size(900, 600)
```

---

render *Render a ggnext plot*

Description

Computes the plot geometry once and serializes it to the chosen render target: a standalone SVG document ("static") or a self-contained interactive HTML page with a <canvas> element, hover tooltips, and scroll-to-zoom ("interactive").

Usage

```
render(plot, ...)
```

Arguments

plot	A GgnextPlot object.
...	Method arguments. The GgnextPlot method takes target ("static" for SVG, "interactive" for HTML/canvas; defaults to "static" unless the plot carries an interact() spec) and file (optional path to write the output to — use .svg for the static target and .html for the interactive one).

Value

The rendered document as a length-1 character vector of class ggnext_render (printed as a short summary, not the full source), invisibly when file is given.

Examples

```
p <- ggnext(cars, aes(speed, dist)) + geom_point()
svg <- render(p)                          # static SVG (the default)
html <- render(p + interact())            # interactive HTML/canvas
```

Scale *Scale: map data values onto a normalized visual range*

Description

Base class for scales. A scale owns one positional aesthetic, learns its domain from data ("training"), and maps data values onto [0, 1].

Usage

```
Scale(aesthetic = character(0), name = NULL, breaks = NULL, labels = NULL)
```

Arguments

aesthetic	Which aesthetic this scale governs ("x" or "y").
name	Axis title; NULL means "use the mapped expression's label".
breaks	Explicit tick positions, or NULL for automatic breaks.
labels	Explicit tick labels (same length as breaks), a function applied to the break values, or NULL for automatic formatting.

Value

An S7 object of class Scale, the base class for scales.

ScaleContinuous *ScaleContinuous: linear continuous positional scale*

Description

ScaleContinuous: linear continuous positional scale

Usage

```
ScaleContinuous(
  aesthetic = "x",
  limits = NULL,
  name = NULL,
  breaks = NULL,
  labels = NULL,
  trans = "identity",
  expand = 0.05
)
```

Arguments

aesthetic	"x" or "y".
limits	Optional numeric length-2 vector fixing the domain; NULL (default) trains the domain from the data.
name	Optional axis title.
breaks	Explicit tick positions in data units, or NULL for automatic breaks.
labels	Tick labels: a character vector, a function applied to the break values, or NULL for automatic formatting.
trans	Axis transform: "identity", "log10", "sqrt", or "reverse".
expand	Fraction of the data span padded onto each end of the axis.
	The domain property (the trained data range) is filled in during the plot build, not at construction.

Value

An S7 object of class ScaleContinuous; usually built by scale_x_continuous() rather than called directly.

ScaleDiscrete *ScaleDiscrete: positional scale for categorical data*

Description

Automatically selected when a positional aesthetic maps to character, factor, or logical data; construct explicitly via scale_x_discrete() / scale_y_discrete() to fix the level order.

Usage

```
ScaleDiscrete(aesthetic = "x", limits = NULL, name = NULL)
```

Arguments

aesthetic	"x" or "y".
limits	Optional character vector fixing the levels (and their order); NULL trains levels from the data.
name	Optional axis title.
	The levels property (the trained categories) is filled in during the plot build, not at construction.

Value

An S7 object of class ScaleDiscrete; usually built by scale_x_discrete() rather than called directly.

scale_breaks *Compute tick breaks for a trained scale*

Description

Compute tick breaks for a trained scale

Usage

```
scale_breaks(scale, ...)
```

Arguments

scale	A trained Scale subclass instance.
...	Method arguments; all methods take n, the target tick count.

Value

Numeric vector of tick values in data units.

scale_color_gradient *Set the continuous color gradient*

Description

Set the continuous color gradient

Usage

```
scale_color_gradient(low = "#1A2E59", high = "#5FD0A5", name = NULL)
```

Arguments

low, high	Gradient endpoint colors.
name	Optional legend title.

Value

A ColorScale object to add with +.

Examples

```
ggnext(iris, aes(Sepal.Length, Sepal.Width, color = Petal.Length)) +
  geom_point() +
  scale_color_gradient(low = "#FFF3B0", high = "#9E2A2B")
```

scale_color_manual *Set the discrete color palette*

Description

Overrides the default colorblind-safe palette for discrete color mappings. Levels take colors in order.

Usage

```
scale_color_manual(values, name = NULL)
```

Arguments

values	Character vector of colors (names or hex).
name	Optional legend title.

Value

A ColorScale object to add with +.

Examples

```
ggnext(iris, aes(Sepal.Length, Sepal.Width, color = Species)) +
  geom_point() +
  scale_color_manual(c("#D55E00", "#0072B2", "#009E73"))
```

scale_map *Map data values through a trained scale onto [0, 1]*

Description

Map data values through a trained scale onto [0, 1]

Usage

```
scale_map(scale, ...)
```

Arguments

scale	A trained Scale subclass instance.
...	Method arguments; all methods take values (numeric data vector) and optionally expand, a padded (expanded) domain to map against instead of the trained domain, as produced during the plot build.

Value

Numeric vector of normalized positions.

scale_train *Train a scale on data values*

Description

Learns the scale's domain (its trained data range). User-supplied `limits` win over the data range. Returns an updated scale — S7 objects are values, so training is a functional update, not a mutation.

Usage

```
scale_train(scale, ...)
```

Arguments

`scale`	A Scale subclass instance.
`...`	Method arguments; all methods take `values`, the raw data vector mapped to this scale's aesthetic.

Value

The scale, with `domain` filled in.

scale_x_continuous *Continuous scale for the x axis*

Description

Controls the axis title, limits, tick positions, tick labels, the padding around the data, and the axis transform.

Usage

```
scale_x_continuous(
  name = NULL,
  limits = NULL,
  breaks = NULL,
  labels = NULL,
  expand = 0.05,
  trans = "identity"
)

scale_y_continuous(
  name = NULL,
  limits = NULL,
  breaks = NULL,
```

```
  labels = NULL,
  expand = 0.05,
  trans = "identity"
)
```

Arguments

name	Axis title (defaults to the mapped expression).
limits	Optional numeric length-2 domain; NULL trains from data.
breaks	Explicit tick positions in data units, or NULL for automatic "nice number" breaks.
labels	Tick labels: a character vector the same length as breaks, a function applied to the break values (e.g. function(x) paste0("$", x)), or NULL for automatic formatting.
expand	Fraction of the data span to pad onto each end of the axis (default 5%). Use 0 for a tight axis.
trans	Axis transform: "identity", "log10", "sqrt", or "reverse".

Value

A ScaleContinuous object to add with +.

Examples

```
ggnext(cars, aes(speed, dist)) +
  geom_point() +
  scale_x_continuous(
    name = "Speed (mph)",
    breaks = c(5, 10, 15, 20, 25),
    expand = 0
  ) +
  scale_y_continuous(labels = function(v) paste0(v, " ft"))
```

---

scale_x_discrete *Discrete positional scale for the x axis*

Description

Discrete positional scale for the x axis

Usage

```
scale_x_discrete(limits = NULL, name = NULL)
```

Arguments

limits	Optional character vector fixing the levels and their order.
name	Optional axis title.

Value

A ScaleDiscrete object to add with +.

`scale_x_log10` *Log10 and square-root scales*

Description

Shorthand for `scale_*_continuous(trans = "log10")` / `"sqrt"`.

Usage

```
scale_x_log10(
  name = NULL,
  limits = NULL,
  breaks = NULL,
  labels = NULL,
  expand = 0.05
)

scale_y_log10(
  name = NULL,
  limits = NULL,
  breaks = NULL,
  labels = NULL,
  expand = 0.05
)

scale_x_sqrt(
  name = NULL,
  limits = NULL,
  breaks = NULL,
  labels = NULL,
  expand = 0.05
)

scale_y_sqrt(
  name = NULL,
  limits = NULL,
  breaks = NULL,
  labels = NULL,
  expand = 0.05
)

scale_x_reverse(
  name = NULL,
```

```
  limits = NULL,
  breaks = NULL,
  labels = NULL,
  expand = 0.05
)

scale_y_reverse(
  name = NULL,
  limits = NULL,
  breaks = NULL,
  labels = NULL,
  expand = 0.05
)
```

Arguments

name	Axis title (defaults to the mapped expression).
limits	Optional numeric length-2 domain; NULL trains from data.
breaks	Explicit tick positions in data units, or NULL for automatic "nice number" breaks.
labels	Tick labels: a character vector the same length as breaks, a function applied to the break values (e.g. function(x) paste0("$", x)), or NULL for automatic formatting.
expand	Fraction of the data span to pad onto each end of the axis (default 5%). Use 0 for a tight axis.

Value

A ScaleContinuous object to add with +.

Examples

```
d <- data.frame(x = 10^(1:5), y = 1:5)
ggnext(d, aes(x, y)) + geom_point() + scale_x_log10()
```

---

scale_y_discrete *Discrete positional scale for the y axis*

Description

Discrete positional scale for the y axis

Usage

```
scale_y_discrete(limits = NULL, name = NULL)
```

Arguments

limits	Optional character vector fixing the levels and their order.
name	Optional axis title.

Value

A ScaleDiscrete object to add with +.

Stat *Stat: statistical transformation applied to layer data*

Description

Base class for statistical transformations. A stat receives the evaluated aesthetic values and returns (possibly new) values for the geom to draw.

Usage

```
Stat(
  name = character(0),
  provides = character(0),
  discrete_provides = character(0)
)
```

Arguments

name	Human-readable stat name.
provides	Aesthetics this stat computes itself. A geom's required aesthetics are validated against the user's mapping *plus* these, so e.g. `geom_point(stat = stat_bin())` is accepted even though the user maps only `x` - the stat supplies `y`.
discrete_provides	Positional aesthetics (`"x"` and/or `"y"`) that this stat always computes as *categorical* values, even when nothing maps that aesthetic before the stat runs (e.g. a Cox-model geom whose `y` is a set of treatment-arm labels the stat itself derives). The build's scale-type detection normally infers continuous vs. discrete from the raw, pre-stat mapping - the one signal it cannot see for an aesthetic that is *entirely* stat-computed; listing it here is that missing signal. When set, the discrete scale's levels are trained from the layer's `group` aesthetic (present pre-stat for every layer) rather than from the (absent) raw `x`/`y`.

Value

An S7 object of class `Stat`, the base class for statistical transformations.

stat_bin *Bin observations (histogram counts)*

Description

Bin observations (histogram counts)

Usage

```
stat_bin(bins = 30, binwidth = NULL)
```

Arguments

bins	Number of bins (ignored when binwidth is given).
binwidth	Bin width in data units.

Value

A StatBin object, to pass as a layer's stat.

stat_boxplot *Five-number summary for boxplots*

Description

Five-number summary for boxplots

Usage

```
stat_boxplot(width = 0.6, coef = 1.5)
```

Arguments

width	Box width in x slot units.
coef	Whisker length multiplier.

Value

A StatBoxplot object, to pass as a layer's stat.

stat_count *Count observations at each x*

Description

Count observations at each x

Usage

```
stat_count(width = 0.8)
```

Arguments

width	Bar width as a fraction of the x resolution.

Value

A StatCount object, to pass as a layer's stat.

stat_coxph *Hazard ratios from a Cox proportional-hazards model*

Description

Hazard ratios from a Cox proportional-hazards model

Usage

```
stat_coxph(ref_level = NULL)
```

Arguments

ref_level	Reference level; NULL uses the first observed value.

Value

A StatCoxph object, to pass as a layer's stat.

stat_density *Kernel density estimate*

Description

Kernel density estimate

Usage

```
stat_density(n = 256, adjust = 1)
```

Arguments

n	Number of evaluation points.
adjust	Bandwidth multiplier.

Value

A StatDensity object, to pass as a layer's stat.

stat_identity *Identity statistical transformation*

Description

Passes layer data through untransformed. This is the default stat for geoms that draw the data as given, such as geom_point() and geom_line().

Usage

```
stat_identity()
```

Value

A StatIdentity object, to pass as a layer's stat.

stat_jitter *Jitter positions for strip charts*

Description

Jitter positions for strip charts

Usage

```
stat_jitter(width = NULL, height = NULL, seed = 42)
```

Arguments

width	Horizontal jitter half-range in data units.
height	Vertical jitter half-range.
seed	RNG seed (fixed for reproducible renders).

Value

A StatJitter object, to pass as a layer's stat.

stat_km *Kaplan-Meier survival estimate*

Description

Kaplan-Meier survival estimate

Usage

```
stat_km(conf_int = FALSE)
```

Arguments

conf_int	Compute a 95% confidence band (Greenwood's formula).

Value

A StatKM object, to pass as a layer's stat.

stat_km_risktable *Number-at-risk table for a Kaplan-Meier curve*

Description

Number-at-risk table for a Kaplan-Meier curve

Usage

```
stat_km_risktable(breaks = NULL)
```

Arguments

breaks	Explicit tick times; NULL (default) uses pretty().

Value

A StatKMRiskTable object, to pass as a layer's stat.

stat_nelson_aalen *Nelson-Aalen cumulative hazard estimate*

Description

Nelson-Aalen cumulative hazard estimate

Usage

```
stat_nelson_aalen()
```

Value

A StatNelsonAalen object, to pass as a layer's stat.

stat_pr *Empirical precision-recall curve*

Description

Empirical precision-recall curve

Usage

```
stat_pr()
```

Value

A StatPR object, to pass as a layer's stat.

stat_roc *Empirical ROC curve*

Description

Empirical ROC curve

Usage

```
stat_roc()
```

Value

A `StatROC` object, to pass as a layer's `stat`.

stat_smooth *Fitted trend line with confidence band*

Description

Fitted trend line with confidence band

Usage

```
stat_smooth(method = "loess", se = TRUE, n = 80, level = 0.95)
```

Arguments

`method`	`"loess"` (default) or `"lm"`.
`se`	Include the confidence band.
`n`	Number of grid points.
`level`	Confidence level.

Value

A `StatSmooth` object, to pass as a layer's `stat`.

stat_waterfall *Running totals for waterfall charts*

Description

Running totals for waterfall charts

Usage

```
stat_waterfall(width = 0.8)
```

Arguments

width	Bar width in x slot units.

Value

A StatWaterfall object, to pass as a layer's stat.

stat_ydensity *Mirrored y-density for violins*

Description

Mirrored y-density for violins

Usage

```
stat_ydensity(width = 0.9, n = 101)
```

Arguments

width	Maximum violin width in x slot units.
n	Number of density evaluation points.

Value

A StatYdensity object, to pass as a layer's stat.

Theme *Customize theme settings*

Description

Overrides individual theme settings. By default the overrides apply to `theme_ggnext()`; pass `base` to restyle a different preset.

Usage

```
Theme(
  name = character(0),
  background = character(0),
  panel_fill = character(0),
  grid_color = character(0),
  grid_color_minor = character(0),
  axis_color = character(0),
  label_color = character(0),
  title_color = character(0),
  subtitle_color = character(0),
  strip_fill = character(0),
  strip_color = character(0),
  legend_text_color = character(0),
  panel_border = character(0),
  font = character(0),
  title_font = character(0),
  tick_font_size = integer(0),
  title_font_size = integer(0),
  plot_title_size = integer(0),
  plot_subtitle_size = integer(0),
  caption_size = integer(0),
  strip_font_size = integer(0),
  legend_font_size = integer(0),
  title_face = character(0),
  tick_len = integer(0),
  grid_major_x = logical(0),
  grid_major_y = logical(0),
  axis_line_x = logical(0),
  axis_line_y = logical(0),
  ticks_x = logical(0),
  ticks_y = logical(0),
  axis_text_x = logical(0),
  axis_text_y = logical(0),
  axis_title_x = logical(0),
  axis_title_y = logical(0),
  legend_position = character(0),
  point_palette = character(0),
```

```
  gradient_low = character(0),
  gradient_high = character(0)
)

theme(..., base = NULL)
```

Arguments

`name`	Preset name (informational).
`background`	Plot background color.
`panel_fill`	Panel (data area) background color.
`grid_color`	Major gridline color; `""` hides major gridlines.
`grid_color_minor`	Minor gridline color; `""` (default) draws none.
`axis_color`	Axis line and tick color.
`label_color`	Tick label and axis title color.
`title_color`	Plot title color.
`subtitle_color`	Subtitle and caption color.
`strip_fill`, `strip_color`	Facet strip background and text colors.
`legend_text_color`	Legend label color.
`panel_border`	Panel border color; `""` (default) draws none.
`font`	CSS font-family stack used for all text.
`title_font`	Font stack for the plot title; `""` inherits `font`.
`tick_font_size`	Tick label size (px).
`title_font_size`	Axis title size (px).
`plot_title_size`, `plot_subtitle_size`, `caption_size`	Title block sizes (px).
`strip_font_size`	Facet strip label size (px).
`legend_font_size`	Legend label size (px).
`title_face`	Plot title weight: `"bold"` or `"normal"`.
`tick_len`	Tick mark length (px).
`grid_major_x`, `grid_major_y`	Draw major gridlines per axis.
`axis_line_x`, `axis_line_y`	Draw the axis lines.
`ticks_x`, `ticks_y`	Draw tick marks.

`axis_text_x`, `axis_text_y`
: Draw tick labels.

`axis_title_x`, `axis_title_y`
: Draw axis titles.

`legend_position`
: `"right"`, `"bottom"`, or `"none"`.

`point_palette`
: Optional character vector overriding the discrete color palette for this plot.

`gradient_low`, `gradient_high`
: Optional continuous-gradient endpoints.

`...`
: Named Theme properties to override, e.g. `theme(panel_fill = "white", grid_color = "grey85")`.

`base`
: A Theme to start from; defaults to `theme_ggnext()`.

Value

A Theme to add to a plot with +.

Examples

```
ggnext(cars, aes(speed, dist)) + geom_point() +
  theme(panel_fill = "#FFF8F0", grid_major_x = FALSE)

# Restyle a preset rather than the default.
ggnext(cars, aes(speed, dist)) + geom_point() +
  theme(base = theme_dark(), plot_title_size = 22)
```

---

theme_classic *Classic theme: white panel, black axes, no gridlines*

Description

The traditional statistical-journal look — axis lines and ticks only.

Usage

```
theme_classic()
```

Value

A Theme to add to a plot with +.

theme_dark *Dark theme*

Description

Dark theme

Usage

```
theme_dark()
```

Value

A Theme to add to a plot with +.

theme_ggnext *The default ggnext theme*

Description

Light background with a softly tinted panel and white gridlines.

Usage

```
theme_ggnext()
```

Value

A Theme to add to a plot with +.

Examples

```
p <- ggnext(cars, aes(speed, dist)) + geom_point() + theme_dark()
```

theme_minimal *Minimal theme: no panel fill, light grey gridlines*

Description

Minimal theme: no panel fill, light grey gridlines

Usage

```
theme_minimal()
```

Value

A Theme to add to a plot with +.

theme_modern *Modern theme: airy, high-contrast, horizontal rules only*

Description

Editorial styling — a large title, hairline horizontal gridlines, no vertical grid or axis lines.

Usage

```
theme_modern()
```

Value

A Theme to add to a plot with +.

theme_void *Void theme: data only, no chrome at all*

Description

Useful for treemaps, network graphs, and sparkline-style output.

Usage

```
theme_void()
```

Value

A Theme to add to a plot with +.

validate_plot *Check a plot for common statistical-graphics mistakes*

Description

Runs a small set of rules over a plot's layers and scales - an empty plot, a continuous geom (`geom_point()`, `geom_line()`, ...) mapped to a categorical `y`, a mapped aesthetic that is mostly missing, a discrete color/fill scale with more levels than a legend can usefully show, and a `sqrt`-transformed scale fed negative values. Each is a heuristic, not a guarantee - `validate_plot()` reports what commonly goes wrong, not what is definitely wrong with this particular plot.

Usage

```
validate_plot(plot)
```

Arguments

plot	A GgnextPlot object.

Value

An object of class ggnext_check: a list of findings, each with rule, status (currently always "warn"), and message. Prints as a short report; empty when nothing was found.

Examples

```
p <- ggnext(mtcars, aes(mpg, as.character(cyl))) + geom_point()
validate_plot(p)
```

---

write_plot_data *Write the exact plot data to CSV*

Description

Convenience wrapper around plot_data() and utils::write.csv() for publishing the numbers behind a figure next to the figure itself.

Usage

```
write_plot_data(plot, file, layer = NULL)
```

Arguments

plot	A GgnextPlot.
file	Output path. With multiple layers and no layer given, the layer index is inserted before the extension (fig.csv -> fig-1.csv, fig-2.csv).
layer	Optional layer index (see plot_data()).

Value

The file path(s) written, invisibly.

Examples

```
p <- ggnext(cars, aes(speed, dist)) + geom_point()
out <- file.path(tempdir(), "cars.csv")
write_plot_data(p, out)
```

xlab *Set the x axis title*

Description

Set the x axis title

Usage

```
xlab(label)
```

Arguments

label	Axis title text.

Value

A Labels object to add to a plot with +.

xlim *Set continuous axis limits*

Description

A shorthand for `scale_x_continuous(limits = )`. Data outside the limits is still computed by stats but clipped to the panel when drawn.

Usage

```
xlim(...)

ylim(...)
```

Arguments

...	Two numbers giving the lower and upper limit, or one length-2 numeric vector.

Value

A scale object to add to a plot with +.

Examples

```
ggnext(cars, aes(speed, dist)) + geom_point() + xlim(0, 30) + ylim(0, 150)
```

`ylab` *Set the y axis title*

Description

Set the y axis title

Usage

```
ylab(label)
```

Arguments

`label`	Axis title text.

Value

A Labels object to add to a plot with +.

Index