Geoms

Geometric objects, or **geoms** for short, perform the actual rendering of the layer, controlling the type of plot that you create. For example, using a point geom will create a scatterplot, while using a line geom will create a line plot.

* Graphical primitives:
  + geom\_blank(): display nothing. Most useful for adjusting axes limits using data.
  + geom\_point(): points.
  + geom\_path(): paths.
  + geom\_ribbon(): ribbons, a path with vertical thickness.
  + geom\_segment(): a line segment, specified by start and end position.
  + geom\_rect(): rectangles.
  + geom\_polyon(): filled polygons.
  + geom\_text(): text.
* One variable:
  + Discrete:
    - geom\_bar(): display distribution of discrete variable.
  + Continuous
    - geom\_histogram(): bin and count continuous variable, display with bars.
    - geom\_density(): smoothed density estimate
    - geom\_dotplot(): stack individual points into a dot plot.
    - geom\_freqpoly(): bin and count continuous variable, display with lines.
* Two variables:
  + Both continuous:
    - geom\_point(): scatterplot.
    - geom\_quantile(): smoothed quantile regression.
    - geom\_rug(): marginal rug plots.
    - geom\_smooth(): smoothed line of best fit.
    - geom\_text(): text labels.
  + Show distribution:
    - geom\_bin2d(): bin into rectangles and count.
    - geom\_density2d(): smoothed 2d density estimate.
    - geom\_hex(): bin into hexagons and count.
  + At least one discrete:
    - geom\_count(): count number of point at distinct locations
    - geom\_jitter(): randomly jitter overlapping points.
  + One continuous, one discrete:
    - geom\_bar(stat = "identity"): a bar chart of precomputed summaries
    - geom\_boxplot(): boxplots.
    - geom\_dotplot(): carefully adjust location of overlapping points.
    - geom\_violin(): show density of values in each group.
  + One time, one continuous
    - geom\_area(): area plot.
    - geom\_line(): line plot.
    - geom\_step(): step plot.
  + Display error:
    - geom\_crossbar(): vertical bar with center.
    - geom\_errorbar(): error bars.
    - geom\_linerange(): vertical line.
    - geom\_pointrange(): vertical line with center.
  + Spatial
    - geom\_map(): fast version of geom\_polygon() for map data.
* Three variables:
  + geom\_contour(): contours.
  + geom\_tile(): tile the plane with rectangles.
  + geom\_raster(): fast version of geom\_tile() for equal sized tiles.

Each geom has a set of aesthetics that it understands, some of which *must* be provided. For example, the point geoms requires x and y position, and understands colour, size and shape aesthetics. A bar requires height (ymax), and understands width, border colour and fill colour. Each geom lists its aesthetics in the docuementation.

Some geoms differ primarily in the way that they are parameterised. For example, you can draw a square in three ways:

* By giving geom\_tile() the location (x and y) and dimensions (width and height).
* By giving geom\_rect() top (ymax), bottom (ymin), left (xmin) and right (xmax) positions.
* By giving geom\_polygon() a four row data frame with the x and y positions of each corner.

Other related geoms are:

* geom\_segment(), and geom\_line()
* geom\_area() and geom\_ribbon().

If alternative parameterisations are available, picking the right one for your data will usually make it much easier to draw a plot you want.

Stats

A statistical transformation, or **stat**, transforms the data, typically by summarising it in some manner. For example, a useful stat is the smoother, which calculates the smoothed mean of y, conditional on x. You’ve already used many of ggplot2’s stats because they’re used behind the scenes for to generate many important geoms:

* stat\_bin(): geom\_bar(), geom\_freqpoly(), geom\_histogram()
* stat\_bin2d(): geom\_bin2d()
* stat\_bindot(): geom\_dotplot()
* stat\_binhex(): geom\_hex()
* stat\_boxplot(): geom\_boxplot()
* stat\_contour(): geom\_contour()
* stat\_quantile(): geom\_quantile()
* stat\_smooth(): geom\_smooth()
* stat\_sum(): geom\_count()

You’ll rarely call these functions directly, but they are useful to know about because their documentation often provides more detail about the corresponding statistical transformation.

Other stats can’t be created with a geom\_ function:

* stat\_ecdf(): compute a empirical cumulative distribution plot.
* stat\_function(): compute y values from a function of x values.
* stat\_summary(): summarise y values at distinct x values.
* stat\_summary2d(), stat\_summary\_hex(): summarised binned values.
* stat\_qq(): perform calculations for a quantile-qunatile plot.
* stat\_spoke(): convert angle and radius to position.
* stat\_unique(): remove duplicated rows.

There are two ways to use these functions. You can either add a stat\_() function and override the default geom, or add a geom\_() function and override the default stat:

### **Generated variables**

Internally, a stat takes a data frame as input and returns a data frame as output, and so a stat can add new variables to the original dataset. It is possible to map aesthetics to these new variables. For example, stat\_bin, the statistic used to make histograms, produces the following variables:

* count, the number of observations in each bin
* density, the density of observations in each bin (percentage of total / bar width)
* x, the centre of the bin

These generated variables can be used instead of the variables present in the original dataset. For example, the default histogram geom assigns the height of the bars to the number of observations (count), but if you’d prefer a more traditional histogram, you can use the density (density). The following example shows a density histogram of carat from the diamonds dataset.

Position adjustments

Position adjustments apply minor tweaks to the position of elements within a layer. Three adjustments apply primarily to bars:

* position\_dodge(): place overlapping bars (or boxplots) side-by-side.
* position\_stack(): stack overlapping bars (or areas) on top of each other.
* position\_fill(): stack overlapping bars, scaling so the top is always at 1.

dplot <- ggplot(diamonds, aes(clarity, fill = cut)) +

theme(legend.position = "none")

dplot + geom\_bar()

dplot + geom\_bar(position = "fill")

dplot + geom\_bar(position = "dodge")

There are three position adjustments that are primarily useful for points:

* position\_nudge(): move points by a fixed offset.
* position\_jitter(): add a little random noise to every position.
* position\_jitterdodge(): dodge points within groups, then add a little random noise.

Theme Layer

Use theme() to modify individual components of a theme, allowing you to control the appearance of all non-data components of the plot. theme() only affects a single plot: see [**theme\_update**](http://ggplot2.tidyverse.org/reference/theme_get.html) if you want modify the active theme, to affect all subsequent plots.

theme(line, rect, text, title, aspect.ratio, axis.title, axis.title.x,

axis.title.x.top, axis.title.y, axis.title.y.right, axis.text, axis.text.x,

axis.text.x.top, axis.text.y, axis.text.y.right, axis.ticks, axis.ticks.x,

axis.ticks.y, axis.ticks.length, axis.line, axis.line.x, axis.line.y,

legend.background, legend.margin, legend.spacing, legend.spacing.x,

legend.spacing.y, legend.key, legend.key.size, legend.key.height,

legend.key.width, legend.text, legend.text.align, legend.title,

legend.title.align, legend.position, legend.direction, legend.justification,

legend.box, legend.box.just, legend.box.margin, legend.box.background,

legend.box.spacing, panel.background, panel.border, panel.spacing,

panel.spacing.x, panel.spacing.y, panel.grid, panel.grid.major,

panel.grid.minor, panel.grid.major.x, panel.grid.major.y, panel.grid.minor.x,

panel.grid.minor.y, panel.ontop, plot.background, plot.title, plot.subtitle,

plot.caption, plot.margin, strip.background, strip.placement, strip.text,

strip.text.x, strip.text.y, strip.switch.pad.grid, strip.switch.pad.wrap, ...,

**complete** **=** FALSE, **validate** **=** TRUE)

 Arguments

|  |  |
| --- | --- |
| **line** | all line elements (element\_line) |
| **rect** | all rectangular elements (element\_rect) |
| **text** | all text elements (element\_text) |
| **title** | all title elements: plot, axes, legends (element\_text; inherits from text) |
| **aspect.ratio** | aspect ratio of the panel |
| **axis.title** | label of axes (element\_text; inherits from text) |
| **axis.title.x** | x axis label (element\_text; inherits from axis.title) |
| **axis.title.x.top** | x axis label on top axis (element\_text; inherits from axis.title.x) |
| **axis.title.y** | y axis label (element\_text; inherits from axis.title) |
| **axis.title.y.right** | y axis label on right axis (element\_text; inherits from axis.title.y) |
| **axis.text** | tick labels along axes (element\_text; inherits from text) |
| **axis.text.x** | x axis tick labels (element\_text; inherits from axis.text) |
| **axis.text.x.top** | x axis tick labels on top axis (element\_text; inherits from axis.text.x) |
| **axis.text.y** | y axis tick labels (element\_text; inherits from axis.text) |
| **axis.text.y.right** | y axis tick labels on right axis (element\_text; inherits from axis.text.y) |
| **axis.ticks** | tick marks along axes (element\_line; inherits from line) |
| **axis.ticks.x** | x axis tick marks (element\_line; inherits from axis.ticks) |
| **axis.ticks.y** | y axis tick marks (element\_line; inherits from axis.ticks) |
| **axis.ticks.length** | length of tick marks (unit) |
| **axis.line** | lines along axes (element\_line; inherits from line) |
| **axis.line.x** | line along x axis (element\_line; inherits from axis.line) |
| **axis.line.y** | line along y axis (element\_line; inherits from axis.line) |
| **legend.background** | background of legend (element\_rect; inherits from rect) |
| **legend.margin** | the margin around each legend (margin) |
| **legend.spacing** | the spacing between legends (unit) |
| **legend.spacing.x** | the horizontal spacing between legends (unit); inherits from legend.spacing |
| **legend.spacing.y** | the horizontal spacing between legends (unit); inherits from legend.spacing |
| **legend.key** | background underneath legend keys (element\_rect; inherits from rect) |
| **legend.key.size** | size of legend keys (unit) |
| **legend.key.height** | key background height (unit; inherits from legend.key.size) |
| **legend.key.width** | key background width (unit; inherits from legend.key.size) |
| **legend.text** | legend item labels (element\_text; inherits from text) |
| **legend.text.align** | alignment of legend labels (number from 0 (left) to 1 (right)) |
| **legend.title** | title of legend (element\_text; inherits from title) |
| **legend.title.align** | alignment of legend title (number from 0 (left) to 1 (right)) |
| **legend.position** | the position of legends ("none", "left", "right", "bottom", "top", or two-element numeric vector) |
| **legend.direction** | layout of items in legends ("horizontal" or "vertical") |
| **legend.justification** | anchor point for positioning legend inside plot ("center" or two-element numeric vector) or the justification according to the plot area when positioned outside the plot |
| **legend.box** | arrangement of multiple legends ("horizontal" or "vertical") |
| **legend.box.just** | justification of each legend within the overall bounding box, when there are multiple legends ("top", "bottom", "left", or "right") |
| **legend.box.margin** | margins around the full legend area, as specified using [**margin**](http://ggplot2.tidyverse.org/reference/element.html) |
| **legend.box.background** | background of legend area (element\_rect; inherits from rect) |
| **legend.box.spacing** | The spacing between the plotting area and the legend box (unit) |
| **panel.background** | background of plotting area, drawn underneath plot (element\_rect; inherits from rect) |
| **panel.border** | border around plotting area, drawn on top of plot so that it covers tick marks and grid lines. This should be used with fill=NA (element\_rect; inherits from rect) |
| **panel.spacing** | spacing between facet panels (unit) |
| **panel.spacing.x** | horizontal spacing between facet panels (unit; inherits from panel.spacing) |
| **panel.spacing.y** | vertical spacing between facet panels (unit; inherits from panel.spacing) |
| **panel.grid** | grid lines (element\_line; inherits from line) |
| **panel.grid.major** | major grid lines (element\_line; inherits from panel.grid) |
| **panel.grid.minor** | minor grid lines (element\_line; inherits from panel.grid) |
| **panel.grid.major.x** | vertical major grid lines (element\_line; inherits from panel.grid.major) |
| **panel.grid.major.y** | horizontal major grid lines (element\_line; inherits from panel.grid.major) |
| **panel.grid.minor.x** | vertical minor grid lines (element\_line; inherits from panel.grid.minor) |
| **panel.grid.minor.y** | horizontal minor grid lines (element\_line; inherits from panel.grid.minor) |
| **panel.ontop** | option to place the panel (background, gridlines) over the data layers. Usually used with a transparent or blank panel.background. (logical) |
| **plot.background** | background of the entire plot (element\_rect; inherits from rect) |
| **plot.title** | plot title (text appearance) (element\_text; inherits from title) left-aligned by default |
| **plot.subtitle** | plot subtitle (text appearance) (element\_text; inherits from title) left-aligned by default |
| **plot.caption** | caption below the plot (text appearance) (element\_text; inherits from title) right-aligned by default |
| **plot.margin** | margin around entire plot (unit with the sizes of the top, right, bottom, and left margins) |
| **strip.background** | background of facet labels (element\_rect; inherits from rect) |
| **strip.placement** | placement of strip with respect to axes, either "inside" or "outside". Only important when axes and strips are on the same side of the plot. |
| **strip.text** | facet labels (element\_text; inherits from text) |
| **strip.text.x** | facet labels along horizontal direction (element\_text; inherits from strip.text) |
| **strip.text.y** | facet labels along vertical direction (element\_text; inherits from strip.text) |
| **strip.switch.pad.grid** | space between strips and axes when strips are switched (unit) |
| **strip.switch.pad.wrap** | space between strips and axes when strips are switched (unit) |
| **...** | additional element specifications not part of base ggplot2. If supplied validate needs to be set to FALSE. |
| **complete** | set this to TRUE if this is a complete theme, such as the one returned by theme\_grey(). Complete themes behave differently when added to a ggplot object. Also, when setting complete = TRUE all elements will be set to inherit from blank elements. |
| **validate** | TRUE to run validate\_element, FALSE to bypass checks. |