Package: ggtypst (via r-universe)

May 22, 2026

Title Render High-Quality Text and Formulas in 'ggplot2' by Typst

Version 0.1.0

Author Hao Cheng [aut, cre, cph]

Maintainer Hao Cheng <Yousa-Mirage@foxmail.com>

Description Provides a seamless integration of the 'Typst' typesetting engine into 'ggplot2'. It allows users to render complex mathematical formulas, equations, and rich text directly in plot annotations, axis labels, and titles without requiring a local Typst or LaTeX installation. Under the hood, it leverages Rust FFI to compile Typst code into SVG.

URL https://github.com/Yousa-Mirage/ggtypst, https://yousa-mirage.github.io/ggtypst/

BugReports https://github.com/Yousa-Mirage/ggtypst/issues

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/rextendr/version 0.4.2

Config/Needs/website pkgdown, pkgload

SystemRequirements Cargo (Rust's package manager), rustc >= 1.89.0

Depends R (>= 4.2), ggplot2

Imports S7, cli, grImport2, grDevices, grid, rlang, rsvg

Suggests knitr, pkgdown, rmarkdown, spelling, svglite, testthat (>= 3.0.0), vdiffr

VignetteBuilder knitr

Config/testthat/edition 3

Config/testthat/parallel true

Collate 'annotate-typst.R' 'annotate-math.R' 'build-source.R' 'diagnostic.R' 'typst-svg.R' 'grob.R' 'helper.R' 'element-typst.R' 'element-math.R' 'extendr-wrappers.R' 'geom-typst.R' 'geom-math.R' 'mitex.R' 'zzz.R'

Language en-US

Config/pak/sysreqs libjpeg-dev libpng-dev librsvg2-dev libxml2-dev libclang-dev

Repository https://r-multiverse-staging.r-universe.dev

Date/Publication 2026-03-13 05:56:54 UTC

RemoteUrl https://github.com/Yousa-Mirage/ggtypst

RemoteRef a075de59066ac2024c07b9c235310fedb4f9a23e

RemoteSha a075de59066ac2024c07b9c235310fedb4f9a23e

Contents

`annotate_math_mitex` *Annotate a Plot with MiTeX-Converted LaTeX Math*

Description

`annotate_math_mitex()` is the LaTeX-input companion to `annotate_math_typst()`. It accepts LaTeX-style math, converts it to Typst math through MiTeX, and renders the converted result as a single annotation. It will normalize the outer `$...$` or `$$...$$` delimiters automatically.

This allows you to use LaTeX math syntax in your plot, which you may be more familiar with :).

Usage

```
annotate_math_mitex(
  latex_math_code,
  x,
  y,
  hjust = 0.5,
```

```
  vjust = 0.5,
  scale = 1,
  size = NULL,
  size.unit = "pt",
  color = NULL,
  colour = NULL,
  alpha = NULL,
  face = NULL,
  fontface = NULL,
  angle = NULL,
  lineheight = NULL,
  math_family = NULL,
  inline = FALSE
)
```

Arguments

`latex_math_code`	A single LaTeX math string. Outer `$...$` or `$$...$$` delimiters are optional.
`x`, `y`	The annotation position in data coordinates.
`hjust`, `vjust`	Horizontal and vertical justification for the rendered grob (`0` = bottom, `0.5` = center, `1` = top).
`scale`	A positive scaling factor applied to the rendered Typst size.
`size`	Optional text size.
`size.unit`	The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.
`color`, `colour`	Optional text color. RGB or color name are supported.
`alpha`	Optional color alpha multiplier in `[0, 1]`.
`face`, `fontface`	Optional math face. Only `"plain"` and `"bold"` are supported, because math text in Typst is italic by default.
`angle`	Optional text rotation angle in degrees.
`lineheight`	Optional line height value. May be negative.
`math_family`	Optional font family for math content. The default math font is `New Computer Modern Math`. To render a math expression, you don't need to set this and even don't need to have `New Computer Modern Math` installed on your system. Typst has embedded this font by default.
`inline`	Whether to render inline math. Default `FALSE` renders display-style math.

Details

MiTeX is a LaTeX-to-Typst converter. It should be stable and reliable for typical LaTeX math expressions. If you find any LaTeX math that MiTeX fails to convert properly, you can report an issue to `ggtypst` first.

Value

A ggplot2 layer.

See Also

annotate_typst(), annotate_math_typst()

Examples

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  annotate_math_mitex(
    latex_math_code = r"(\eta = \frac{mpg}{wt}, \text{written in LaTeX math})",
    x = 3.5,
    y = 30,
    size = 18,
    face = "bold"
  ) +
  theme_minimal()
```

---

annotate_math_typst *Annotate a Plot with Typst Math*

Description

annotate_math_typst() is the math-specialized variant of annotate_typst(). It treats the annotation text as Typst math expression and then renders it to a math visual representation. It will normalize the outer $...$ or $ ... $ delimiters automatically.

Usage

```
annotate_math_typst(
  typst_math_code,
  x,
  y,
  hjust = 0.5,
  vjust = 0.5,
  scale = 1,
  size = NULL,
  size.unit = "pt",
  color = NULL,
  colour = NULL,
  alpha = NULL,
  face = NULL,
  fontface = NULL,
  angle = NULL,
  lineheight = NULL,
  math_family = NULL,
```

```
  inline = FALSE
)
```

Arguments

typst_math_code	A single Typst math string. Outer `$...$` or `$ ... $` delimiters are optional.
x, y	The annotation position in data coordinates.
hjust, vjust	Horizontal and vertical justification for the rendered grob (`0` = bottom, `0.5` = center, `1` = top).
scale	A positive scaling factor applied to the rendered Typst size.
size	Optional text size.
size.unit	The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.
color, colour	Optional text color. RGB or color name are supported.
alpha	Optional color alpha multiplier in `[0, 1]`.
face, fontface	Optional math face. Only `"plain"` and `"bold"` are supported, because math text in Typst is italic by default.
angle	Optional text rotation angle in degrees.
lineheight	Optional line height value. May be negative.
math_family	Optional font family for math content. The default math font is `New Computer Modern Math`. To render a math expression, you don't need to set this and even don't need to have `New Computer Modern Math` installed on your system. Typst has embedded this font by default.
inline	Whether to render inline math. Default `FALSE` renders display-style math.

Value

A ggplot2 layer.

See Also

annotate_typst(), annotate_math_mitex()

Examples

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  annotate_math_typst(
    typst_math_code = r"(sum_(i=1)^n x_i)",
    x = 3.5,
    y = 30,
    size = 18,
    face = "bold"
  ) +
  theme_minimal()
```

`annotate_typst` *Annotate a Plot with Typst Text*

Description

`annotate_typst()` is a more powerful replacement for `ggplot2::annotate()` that renders a single Typst string and inserts that grob into a `ggplot2` plot. Use it when you want one manually positioned note, callout, or mixed text-and-math label.

Usage

```
annotate_typst(
  typst_code,
  x,
  y,
  hjust = 0.5,
  vjust = 0.5,
  scale = 1,
  size = NULL,
  size.unit = "pt",
  color = NULL,
  colour = NULL,
  alpha = NULL,
  face = NULL,
  fontface = NULL,
  angle = NULL,
  lineheight = NULL,
  family = NULL,
  math_family = NULL
)
```

Arguments

`typst_code`	A single Typst source string to render.
`x`, `y`	The annotation position in data coordinates.
`hjust`, `vjust`	Horizontal and vertical justification for the rendered grob (`0` = bottom, `0.5` = center, `1` = top).
`scale`	A positive scaling factor applied to the rendered Typst size.
`size`	Optional text size.
`size.unit`	The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.
`color`, `colour`	Optional text color. RGB or color name are supported.
`alpha`	Optional color alpha multiplier in `[0, 1]`.
`face`, `fontface`	Optional text face: `"plain"`, `"bold"`, `"italic"`, or `"bold.italic"`.

`angle`	Optional text rotation angle in degrees.
`lineheight`	Optional line height value. May be negative.
`family`	Optional text font family. The family must be available to Typst. If `NULL` or not found, the default family will be used. If you want to show specific languages or characters (e.g., Chinese, Japanese, emoji), you may need to set this.
`math_family`	Optional font family for math content. The default math font is `New Computer Modern Math`. To render a math expression, you don't need to set this and even don't need to have `New Computer Modern Math` installed on your system. Typst has embedded this font by default.

Value

A `ggplot2` layer.

See Also

`annotate_math_typst()`, `annotate_math_mitex()`

Examples

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  annotate_typst(
    typst_code = r"(#rect(radius: 5pt)[*Fuel economy* #linebreak() $sum_(i=1)^n x_i$])",
    x = 3.5,
    y = 30,
    size = 18,
    color = "red"
  ) +
  theme_minimal()
```

`element_math_mitex` *Render Theme Elements with MiTeX-Converted LaTeX Math*

Description

`element_math_mitex()` is the LaTeX-input companion to `element_math_typst()`. It accepts LaTeX-style math, converts it to Typst math through MiTeX, and renders the converted result as the element's visual representation. It will normalize the outer `$...$` or `$$...$$` delimiters automatically.

This allows you to use LaTeX math syntax in your plot, which you may be more familiar with :).

Usage

```
element_math_mitex(
  hjust = NULL,
  vjust = NULL,
  scale = NULL,
  size = NULL,
  size.unit = "pt",
  color = NULL,
  colour = NULL,
  alpha = NULL,
  face = NULL,
  fontface = NULL,
  angle = NULL,
  lineheight = NULL,
  math_family = NULL,
  inline = FALSE,
  margin = NULL,
  debug = NULL,
  inherit.blank = FALSE
)
```

Arguments

`hjust`, `vjust`	Horizontal and vertical justification for the rendered grob (`0` = bottom, `0.5` = center, `1` = top).
`scale`	A positive scaling factor applied to the rendered Typst size.
`size`	Optional font size.
`size.unit`	The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.
`color`, `colour`	Optional text color. RGB or color name are supported.
`alpha`	Optional color alpha multiplier in `[0, 1]`.
`face`, `fontface`	Optional math face. Only `"plain"` and `"bold"` are supported, because math text in Typst is italic by default.
`angle`	Optional text rotation angle in degrees.
`lineheight`	Optional line height value. May be negative.
`math_family`	Optional font family for math content. The default math font is `New Computer Modern Math`. To render a math expression, you don't need to set this and even don't need to have `New Computer Modern Math` installed on your system. Typst has embedded this font by default.
`inline`	Whether to render inline math. Default `FALSE` renders display-style math.
`margin`	Optional text margins created with `ggplot2::margin()`.
`debug`	If `TRUE`, draw a coloured rectangle behind each label for debugging layout.
`inherit.blank`	If `TRUE`, `element_blank()` parents suppress this element.

Details

MiTeX is a LaTeX-to-Typst converter. It should be stable and reliable for typical LaTeX math expressions. If you find any LaTeX math that MiTeX fails to convert properly, you can report an issue to `ggtypst` first.

Value

A ggplot2 theme element.

See Also

`element_typst()`, `element_math_typst()`

Examples

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  labs(x = r"(\eta = \frac{mpg}{wt}, \text{written in LaTeX math})") +
  theme_minimal() +
  theme(axis.title.x = element_math_mitex(size = 14))
```

---

element_math_typst *Render Theme Elements with Typst Math*

Description

`element_math_typst()` is the math-specialized variant of `element_typst()`. It treats the theme element's text as Typst math expression and then renders it as the element's visual representation. It will normalize the outer `$...$` or `$ ... $` delimiters automatically.

Usage

```
element_math_typst(
  hjust = NULL,
  vjust = NULL,
  scale = NULL,
  size = NULL,
  size.unit = "pt",
  color = NULL,
  colour = NULL,
  alpha = NULL,
  face = NULL,
  fontface = NULL,
  angle = NULL,
  lineheight = NULL,
  math_family = NULL,
  inline = FALSE,
  margin = NULL,
```

```
  debug = NULL,
  inherit.blank = FALSE
)
```

Arguments

`hjust`, `vjust`	Horizontal and vertical justification for the rendered grob (`0` = bottom, `0.5` = center, `1` = top).
`scale`	A positive scaling factor applied to the rendered Typst size.
`size`	Optional font size.
`size.unit`	The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.
`color`, `colour`	Optional text color. RGB or color name are supported.
`alpha`	Optional color alpha multiplier in `[0, 1]`.
`face`, `fontface`	Optional math face. Only `"plain"` and `"bold"` are supported, because math text in Typst is italic by default.
`angle`	Optional text rotation angle in degrees.
`lineheight`	Optional line height value. May be negative.
`math_family`	Optional font family for math content. The default math font is `New Computer Modern Math`. To render a math expression, you don't need to set this and even don't need to have `New Computer Modern Math` installed on your system. Typst has embedded this font by default.
`inline`	Whether to render inline math. Default `FALSE` renders display-style math.
`margin`	Optional text margins created with `ggplot2::margin()`.
`debug`	If `TRUE`, draw a coloured rectangle behind each label for debugging layout.
`inherit.blank`	If `TRUE`, `element_blank()` parents suppress this element.

Value

A ggplot2 theme element.

See Also

`element_typst()`, `element_math_mitex()`

Examples

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  labs(
    title = r"("Matrix Title:" mat(0, 1; 1, 0))",
    y = r"($sum_(i=1)^n c_i$)"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_math_typst(size = 18, face = "bold"),
    axis.title.y = element_math_typst(size = 14)
  )
```

`element_typst` *Render Theme Elements with Typst Text*

Description

`element_typst()` is a more powerful replacement for `ggplot2::element_text()` and `ggtext::element_markdown` that renders each theme element through Typst. Use it for plot titles, subtitles, axis titles, axis text, facet strips, legend text, and other theme slots that accept text elements.

Usage

```
element_typst(
  hjust = NULL,
  vjust = NULL,
  scale = NULL,
  size = NULL,
  size.unit = "pt",
  color = NULL,
  colour = NULL,
  alpha = NULL,
  face = NULL,
  fontface = NULL,
  angle = NULL,
  lineheight = NULL,
  family = NULL,
  math_family = NULL,
  margin = NULL,
  debug = NULL,
  inherit.blank = FALSE
)
```

Arguments

`hjust`, `vjust`	Horizontal and vertical justification for the rendered grob (`0` = bottom, `0.5` = center, `1` = top).
`scale`	A positive scaling factor applied to the rendered Typst size.
`size`	Optional font size.
`size.unit`	The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.
`color`, `colour`	Optional text color. RGB or color name are supported.
`alpha`	Optional color alpha multiplier in `[0, 1]`.
`face`, `fontface`	Optional text face: `"plain"`, `"bold"`, `"italic"`, or `"bold.italic"`.
`angle`	Optional text rotation angle in degrees.
`lineheight`	Optional line height value. May be negative.

`family`	Optional text font family. The family must be available to Typst. If `NULL` or not found, the default family will be used. If you want to show specific languages or characters (e.g., Chinese, Japanese, emoji), you may need to set this.
`math_family`	Optional font family for math content. The default math font is `New Computer Modern Math`. To render a math expression, you don't need to set this and even don't need to have `New Computer Modern Math` installed on your system. Typst has embedded this font by default.
`margin`	Optional text margins created with `ggplot2::margin()`.
`debug`	If `TRUE`, draw a coloured rectangle behind each label for debugging layout.
`inherit.blank`	If `TRUE`, `element_blank()` parents suppress this element.

Value

A ggplot2 theme element.

See Also

`element_math_typst()`, `element_math_mitex()`

Examples

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  labs(
    title = r"(*Typst Title* #linebreak() with inline math $E = m c^2$)",
    x = r"(X-axis powered by _Typst_: $hat(y) = beta_0 + beta_1 x$)"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_typst(size = 18, color = "red"),
    axis.title.x = element_typst(size = 14, color = "#40a2b4")
  )
```

---

geom_math_mitex *Plot Labels in Data with MiTeX-Converted LaTeX Math*

Description

`geom_math_mitex()` is the LaTeX-input companion to `geom_math_typst()` that renders data-driven LaTeX-style math labels by converting each `label` to Typst through MiTeX before rendering. It will normalize the outer `$...$` or `$$...$$` delimiters automatically.

This allows you to use LaTeX math syntax in your plot, which you may be more familiar with :).

Usage

```
geom_math_mitex(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  nudge_x = 0,
  nudge_y = 0,
  inline = FALSE,
  size.unit = "pt",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).
`stat`	The statistical transformation to use on the data for this layer. When using a `geom_*()` function to construct a layer, the `stat` argument can be used to override the default coupling between geoms and stats. The `stat` argument accepts the following: • A `Stat` ggproto subclass, for example `StatCount`. • A string naming the stat. To give the stat as a string, strip the function name of the `stat_` prefix. For example, to use `stat_count()`, give the stat as `"count"`. • For more information and other ways to specify the stat, see the layer stat documentation.
`position`	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following: • The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position.

- A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`.
- For more information and other ways to specify the position, see the layer position documentation.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

`nudge_x`, `nudge_y` Horizontal and vertical nudge offsets. Supply these instead of `position` to shift labels after the default identity position.

`inline` Whether to render inline math. Default `FALSE` renders display-style math.

`size.unit` The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use `TRUE`. If `NA`, all levels are shown in legend, but unobserved levels are omitted.

`inherit.aes` If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and

aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `annotation_borders()`.

Details

MiTeX is a LaTeX-to-Typst converter. It should be stable and reliable for typical LaTeX math expressions. If you find any LaTeX math that MiTeX fails to convert properly, you can report an issue to `ggtypst` first.

Value

A ggplot2 layer that can be added to a plot.

See Also

`geom_typst()`, `geom_math_typst()`, `ggplot2::geom_label()`

Examples

```
math_labels <- data.frame(
  wt = c(2.1, 3.2, 4.8),
  mpg = c(32, 24, 16),
  label = c(
    r"(\sum_{i=1}^n x_i)",
    r"(x^2 + y^2)",
    r"(\int_0^1 x \, dx)"
  )
)

ggplot(math_labels, aes(wt, mpg, label = label)) +
  geom_point() +
  geom_math_mitex(nudge_y = 1, size = 14, show.legend = FALSE) +
  theme_minimal()
```

`geom_math_typst` *Plot Labels in Data with Typst Math*

Description

`geom_math_typst()` is the math-specialized variant of `geom_typst()` that renders data-driven Typst math labels. Each `label` is treated as Typst math expression. It will normalize the outer `$...$` or `$ ... $` delimiters automatically.

Usage

```
geom_math_typst(
  mapping = NULL,
  data = NULL,
  stat = "identity",
```

```
  position = "identity",
  ...,
  nudge_x = 0,
  nudge_y = 0,
  inline = FALSE,
  size.unit = "pt",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).
`stat`	The statistical transformation to use on the data for this layer. When using a `geom_*()` function to construct a layer, the `stat` argument can be used to override the default coupling between geoms and stats. The `stat` argument accepts the following: • A `Stat` ggproto subclass, for example `StatCount`. • A string naming the stat. To give the stat as a string, strip the function name of the `stat_` prefix. For example, to use `stat_count()`, give the stat as `"count"`. • For more information and other ways to specify the stat, see the layer stat documentation.
`position`	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following: • The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`. • For more information and other ways to specify the position, see the layer position documentation.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

`nudge_x`, `nudge_y` Horizontal and vertical nudge offsets. Supply these instead of `position` to shift labels after the default identity position.

`inline` Whether to render inline math. Default `FALSE` renders display-style math.

`size.unit` The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use `TRUE`. If `NA`, all levels are shown in legend, but unobserved levels are omitted.

`inherit.aes` If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `annotation_borders()`.

Value

A ggplot2 layer that can be added to a plot.

See Also

geom_typst(), geom_math_mitex(), ggplot2::geom_label()

Examples

```
math_labels <- data.frame(
  wt = c(2.1, 3.2, 4.8),
  mpg = c(32, 24, 16),
  label = c(r"(sum_(i=1)^n x_i)", r"(x^2 + y^2)", r"(integral_0^1 x dif x)")
)

ggplot(math_labels, aes(wt, mpg, label = label)) +
  geom_point() +
  geom_math_typst(nudge_y = 1, size = 14, show.legend = FALSE) +
  theme_minimal()
```

---

geom_typst *Plot Labels in Data with Typst Text*

Description

geom_typst() is a more powerful replacement for ggplot2::annotate() and ggtext::geom_richtext() that renders a vector of raw Typst labels at data positions. Each label is compiled independently with Typst. Parameter and aesthetic names follow the conventions of ggplot2::geom_label().

Usage

```
geom_typst(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  nudge_x = 0,
  nudge_y = 0,
  size.unit = "pt",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

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

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

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

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

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

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

- A `Stat` ggproto subclass, for example `StatCount`.
- A string naming the stat. To give the stat as a string, strip the function name of the `stat_` prefix. For example, to use `stat_count()`, give the stat as `"count"`.
- For more information and other ways to specify the stat, see the layer stat documentation.

`position` A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following:

- The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position.
- A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`.
- For more information and other ways to specify the position, see the layer position documentation.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.

- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

`nudge_x`, `nudge_y`
: Horizontal and vertical nudge offsets. Supply these instead of `position` to shift labels after the default identity position.

`size.unit`
: The unit of `size`. Defaults to points (`"pt"`). Use `"mm"` for ggplot2-style text sizes.

`na.rm`
: If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`show.legend`
: logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use `TRUE`. If `NA`, all levels are shown in legend, but unobserved levels are omitted.

`inherit.aes`
: If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `annotation_borders()`.

Value

A ggplot2 layer that can be added to a plot.

Aesthetics

`geom_typst()` understands the following aesthetics (required aesthetics are in bold):

- `x`
- `y`
- `label` Raw Typst source code. `label` can be mapped in `ggplot2::aes()` or supplied as a constant value, for example `geom_typst(label = "Hello")`, to render the same Typst label for every row in the layer.
- `hjust`
- `vjust`
- `scale`
- `size`
- `colour` (alias `color`)
- `alpha`

- face (alias fontface)
- angle
- lineheight
- family
- math_family

To learn more about these theme arguments, see annotate_typst(). If mapped colour values are already literal colours such as "#1E66F5", add ggplot2::scale_colour_identity() so ggplot2 treats them as colours instead of discrete categories.

See Also

geom_math_typst(), geom_math_mitex(), ggplot2::geom_label()

Examples

```
labels <- data.frame(
  wt = c(2, 3.5, 4.5),
  mpg = c(15, 30, 10),
  label = c(
    r"(*Toyota Corolla* )",
    r"(*Fiat 128*, $y = x$)",
    r"(*_Maserati Bora_*)"
  ),
  colour = c("blue", NA, "red")
)

ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  geom_typst(
    data = labels,
    aes(wt, mpg, label = label, colour = colour),
    scale = 1.1,
    size = 14,
    show.legend = FALSE
  ) +
  scale_colour_identity() +
  theme_minimal()
```

Index