Package: highdir (via r-universe)

May 29, 2026

Title Backend-Agnostic Figure Builder for 'highcharter' and 'ggplot2'

Version 0.5.0

Description Provides a backend-agnostic 'API' for creating data visualizations using 'highcharter' (interactive) or 'ggplot2' (static). Figures are defined once via a specification object and can be rendered to either backend without modifying the calling code. Supports both declarative and layered workflows, flexible theming and colour palettes, optional 'JavaScript' enhancements, and tools for exporting figures and interactive exploration via a 'shiny' app.

URL https://folkehelsestats.github.io/highdir/, https://github.com/folkehelsestats/highdir

BugReports https://github.com/folkehelsestats/highdir/issues

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports highcharter (>= 0.9.4), ggplot2 (>= 3.4.0), htmlwidgets (>= 1.6.0), jsonlite (>= 1.8.0), rlang (>= 1.1.0), viridis (>= 0.6.0), shiny (>= 1.7.0), tools, utils, stats

Suggests sf (>= 1.0.0), maps (>= 3.4.0), webshot2, bslib (>= 0.7.0), testthat (>= 3.0.0), withr (>= 2.5.0), data.table, knitr, rmarkdown, tibble

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake libglpk-dev make libicu-dev libuv1-dev libxml2-dev libssl-dev zlib1g-dev

Repository https://folkehelsestats.r-universe.dev

Date/Publication 2026-05-29 22:51:53 UTC

RemoteUrl https://github.com/folkehelsestats/highdir

RemoteRef HEAD

RemoteSha f4884ac6eda33c9ca666c8b8c9347307219fefb7

Contents

`+.hd` *Add Layers to an hd Object*

Description

Implements the `+` operator for `hd()` objects, mirroring how `ggplot2` builds plots incrementally. Recognised right-hand sides:

Usage

```
## S3 method for class 'hd'
e1 + e2
```

Arguments

e1	An hd object (left-hand side).
e2	An hd_geom or hd_opts object (right-hand side).

Details

hd_geom **object** Sets the geometry (from any hd_geom_*() call). Adding a second geom replaces the first where highdir renders one geometry per figure.

hd_opts **object** Sets presentation options. Adding a second hd_opts replaces the first.

Unknown right-hand sides produce an informative error rather than silently being ignored.

Value

The updated hd object (invisibly, so the chain prints once).

Examples

```
df <- data.frame(age = c("18-24", "25-34"), pct = c(42, 55))

# Chain layers
p <- hd(df, x = "age", y = "pct") +
  hd_geom_column() +
  hd_opts(title = "Demo")

# Reuse a partial object with different opts
base <- hd(df, x = "age", y = "pct") + hd_geom_column()
base + hd_opts(title = "English title")
base + hd_opts(title = "Norsk tittel")
```

---

alco1 *Alcohol Consumption Data*

Description

Annual estimates of alcohol consumption, including adjusted mean values, standard errors, and 95% confidence intervals. Includes an alternative unit (adj_enhet) with corresponding uncertainty estimates.

Format

A data frame with 14 rows and 9 variables:

Source

Helsedirektoratet. *Tall om alkohol*. https://www.helsedirektoratet.no/rapporter/tall-om-alkohol

---

alco2 *Alcohol Consumption by Gender*

Description

Annual estimates of alcohol consumption stratified by kjønn (gender), including adjusted means, standard errors, and 95\ intervals. Also includes an alternative unit (`adj_enhet`) with uncertainty.

Format

A data frame with 28 rows and 10 variables:

Source

Helsedirektoratet. *Tall om alkohol*. https://www.helsedirektoratet.no/rapporter/tall-om-alkohol

---

geom_args *Show Arguments for a Geometry*

Description

Prints the required and optional `...` arguments accepted by a geometry when used with `hd_make()`. This is the primary discoverability tool for geometry-specific arguments that do not appear in `hd_make()`'s signature.

Usage

```
geom_args(type = NULL)
```

Arguments

`type`	Character. Geometry name, e.g. `"line"`, `"ranked_bar"`. If `NULL` (default), prints a summary for every registered geometry.

Value

A data frame of argument metadata, invisibly. The primary purpose is the side-effect of printing.

Why this exists

`hd_make()` uses `...` for all geometry-specific arguments so its own signature stays clean regardless of how many geometries are registered. The trade-off is that users cannot see available args from `hd_make()` alone. `geom_args()` solves that: it reads the `required_args` and `optional_args` fields registered for each geometry and presents them in a readable table.

Examples

```
geom_args("line")
geom_args("ranked_bar")
geom_args("arearange")
geom_args()           # all registered geometries
```

---

get_palette *Retrieve a Named Palette*

Description

Retrieve a Named Palette

Usage

```
get_palette(name)
```

Arguments

name	Character. Palette name (see list_palettes()).

Value

Character vector of colours, or NULL if not found.

Examples

```
get_palette("hdir")
```

---

gg_theme *Build a ggplot2 Theme Object*

Description

Constructs a ggplot2 theme by resolving a base theme, then merging colour and font overrides – exactly mirroring what hd_theme() does for the highcharter backend.

Usage

```
gg_theme(theme = NULL, colors = NULL, font = NULL)
```

Arguments

theme	Character name string, ggplot2 theme object, or NULL. NULL reads from getOption("highdir.gg_theme'
colors	Character vector, palette name string, or NULL. Resolved colours are stored on the returned object and applied by ggplot_engine() via scale_color_manual / scale_fill_manual. NULL reads from getOption("highdir.colors").
font	Character or NULL. Font family applied to all text elements via theme(text = element_text(family = font)). NULL reads from getOption("highdir.font").

Details

Priority for each argument:

- theme: explicit argument > getOption("highdir.gg_theme") > "classic"
- colors: explicit argument > getOption("highdir.colors") > NULL
- font: explicit argument > getOption("highdir.font") > NULL

Called automatically inside ggplot_engine(); also useful for applying the package theme to a ggplot built outside highdir.

Built-in name strings and their ggplot2 equivalents:

Name	ggplot2 function
"classic" (default)	theme_classic()
"minimal"	theme_minimal()
"bw"	theme_bw()
"light"	theme_light()
"dark"	theme_dark()
"void"	theme_void()
"grey" / "gray"	theme_grey()

Value

An object of class "hd_gg_theme" - a list with two fields: $theme (a ggplot2 theme object with font baked in) and $colors (a resolved character vector or NULL). ggplot_engine() unpacks both. The object can also be added directly to a ggplot with + via the +.gg method (only the theme is applied; colors are handled separately by apply_gg_colors()).

Examples

```
gg_theme()                                    # session defaults
gg_theme("bw")                                # theme_bw(), session colors/font
gg_theme("classic", colors = c("#025169", "#7C145C"))
gg_theme("minimal", font = "Source Sans Pro")
gg_theme(ggplot2::theme_bw(base_size = 14), font = "mono")
```

hd *Initialise a Composable highdir Figure*

Description

Creates an hd object that accumulates geometry and presentation layers via +, then renders when printed. This mirrors the way ggplot2 builds plots: data mapping is declared first; visual decisions are added incrementally; nothing is rendered until the object hits the console (or an explicit print() / knit call).

Usage

```
hd(
  data,
  x = NULL,
  y = NULL,
  group = NULL,
  n = NULL,
  colour = NULL,
  backend = getOption("highdir.backend", "highcharter")
)
```

Arguments

data	A data.frame **or** an hd_spec() object. When an hd_spec is supplied every other mapping argument (x, y, ...) is ignored - the spec carries them already.
x	Character. Column name for the x-axis variable. Ignored when data is an hd_spec.
y	Character. Column name for the y-axis variable. Ignored when data is an hd_spec.
group	Character or NULL. Column used to split data into multiple series. Ignored when data is an hd_spec.
n	Character or NULL. Column of raw counts shown in highcharter tooltips alongside the y value. Ignored when data is an hd_spec.
colour	Character or NULL. ggplot2 colour aesthetic column. Defaults to group when NULL and group is set. Ignored when data is an hd_spec.
backend	Character. Rendering engine - "highcharter" (default, interactive) or "ggplot2" (static), or any engine added with register_backend(). Falls back to getOption("highdir.backend", "highcharter").

Details

hd() accepts either raw data columns **or** an existing hd_spec() object. Passing an hd_spec is the recommended bridge for code that already constructs specs separately (e.g. in Shiny or a reporting pipeline).

Value

An S3 object of class "hd" with slots:

$spec An hd_spec() object.

$geom NULL until a + hd_geom_*() layer is added.

$opts An hd_opts() object (defaults until overridden).

$backend Character. The resolved engine name.

See Also

hd_geom_column(), hd_geom_line(), hd_geom_arearange(), hd_opts(), hd_make()

Examples

```
df <- data.frame(
  age = rep(c("18-24", "25-34", "35-44", "45-54"), each = 2),
  sex = rep(c("Male", "Female"), 4),
  pct = c(42, 38, 55, 61, 48, 52, 60, 57),
  n   = c(120, 115, 200, 210, 180, 175, 160, 155)
)

# Composable style
hd(df, x = "age", y = "pct", group = "sex") +
  hd_geom_column() +
  hd_opts(title = "Health survey", ylim = c(0, 80))

# Pass an existing hd_spec
spec <- hd_spec(df, x = "age", y = "pct", group = "sex", n = "n")
hd(spec) +
  hd_geom_line(smooth = TRUE) +
  hd_opts(title = "Trend")

# Switch backend per figure
hd(df, x = "age", y = "pct", backend = "ggplot2") +
  hd_geom_column() +
  hd_opts(title = "Static version")
```

---

hd_add_js *Inject JavaScript into a Highcharts Widget*

Description

Appends custom JavaScript to a highchart object via chart.events.<where>. Use this for hand-written callbacks and plugins. For Highcharts built-in modules (accessibility, exporting, etc.) use highcharter::hc_add_dependency() instead.

Usage

```
hd_add_js(
  hc,
  code = NULL,
  file = NULL,
  plugin = NULL,
  where = c("load", "render")
)
```

Arguments

hc	A highchart object.
code	Character or NULL. Raw JavaScript string.
file	Character or NULL. Path to a .js file to read and inject.
plugin	Character or NULL. Name of a plugin bundled in inst/js/.
where	"load" (default) or "render".

Details

Exactly one of code, file, or plugin must be supplied.

Value

The highchart object with JS injected.

Examples

```
if (interactive()) {
spec <- hd_spec(mtcars, "wt", "mpg")
fig  <- hd_make(spec, "scatter")
fig  <- hd_add_js(fig, code = "console.log('chart loaded');")
}
```

---

hd_app *Launch the highdir Shiny GUI*

Description

Opens an interactive browser-based application for building figures with highcharter or ggplot2 without writing R code.

Usage

```
hd_app(return.app = FALSE)
```

Arguments

`return.app`	Logical. When deploying to server like Shiny.io to return the app object instead of launching it.

Details

The UI (`inst/app/ui.R`), server (`inst/app/server.R`) and shared setup (`inst/app/global.R`) live in `inst/app/` so the folder can be deployed independently to Shiny Server or shinyapps.io.

Value

Launches a Shiny app; does not return a value.

Features

- Upload datasets in any format supported by **rio** (CSV, XLSX, SPSS, Stata, RDS, ...).
- Choose geometry (`column`, `line`, `scatter`, `arearange`, `pie`), backend, axis variables, and group column.
- Set title, subtitle, caption, colour palette, and HC theme.
- Toggle JS hover band per figure.
- Render on demand with the **Draw** button.
- Download as HTML / JSON / PNG (highcharter) or PNG / SVG (ggplot2).
- Copy the equivalent `hd_make()` call from the **R code** tab.

See Also

`hd_make()`, `hd_spec()`, `hd_opts()`, `hd_save()`

`hd_geom_arearange` *Add an Arearange (Confidence Band) Layer*

Description

Geometry layer for ribbon / confidence-interval charts. Unlike the other `hd_geom_*()` functions, `ymin` and `ymax` are **required** named arguments (they map to column names in `spec$data`) rather than optional `...` extras. This makes the contract explicit at the call site instead of burying required information inside `...`.

Usage

```
hd_geom_arearange(ymin = NULL, ymax = NULL, ...)
```

Arguments

ymin	Character. Column name for the lower bound of the range.
ymax	Character. Column name for the upper bound of the range.
...	Additional optional arguments forwarded to the geom function (e.g. show_line = FALSE, single_colour = "#025169"). Run geom_args("arearange") for the full list.

Value

An S3 object of class "hd_geom" for use with +.hd.

Examples

```
# Single series
spec_ar1 <- hd_spec(alco1,
                    x    = "year",
                    y    = "adj_enhet")

opts_ar <- hd_opts(
  title    = "Alcohol consumption with 95% CI",
  subtitle = "Source: Norwegian Directorate of Health",
  ylim     = c(0, 30),
  ylab = "Number of units alcohol"
)

hd_make(spec_ar1, "arearange", opts_ar,
       ymin = "lower_enhet", ymax = "upper_enhet")

# Static ggplot2 version
hd_make(spec_ar1, "arearange", opts_ar,
       ymin = "lower_enhet", ymax = "upper_enhet", backend = "ggplot2")

#' # Multi-series with group column
# Grouped by kjonn
spec_ar2 <- hd_spec(alco2,
                    x     = "year",
                    y     = "adj_enhet",
                    group = "kjonn")

hd_make(spec_ar2, "arearange", opts_ar,
        ymin = "lower_enhet", ymax = "upper_enhet")
```

---

hd_geom_column *Column Geometry Layer for hd Objects*

Description

`hd_geom_column()` creates a column geometry layer that is added to an `hd()` object via `+`. The layer records the geometry type and any geometry-specific arguments; rendering only happens when the `hd` object is printed.

Usage

```
hd_geom_column(...)
```

Arguments

`...`	Geometry-specific arguments forwarded to `hd_make()`.

Value

An S3 object of class `"hd_geom"` for use with `+.hd`.

Examples

```
survey <- data.frame(
  age_group = rep(c("18-24", "25-34", "35-44", "45-54", "55-64"), each = 2),
  kjonn     = rep(c("Male", "Female"), times = 5),
  pct       = c(42, 38, 55, 61, 48, 52, 60, 57, 65, 70),
  n         = c(120, 115, 200, 210, 180, 175, 160, 155, 140, 145)
)

spec_col <- hd_spec(survey,
                    x     = "age_group",
                    y     = "pct",
                    group = "kjonn",
                    n     = "n")

opts_col <- hd_opts(
  title    = "Alcohol use by age group and kjonn",
  subtitle = "Source: Norwegian Directorate of Health",
  ylim     = c(0, 100),
  yint     = 20,
  ylab     = "Percentage (%)"
)

# Interactive (default)
hd_make(spec_col, "column", opts_col)

# Static ggplot2
hd_make(spec_col, "column", opts_col, backend = "ggplot2")

# Composable style
p <- hd(survey, x = "age_group", y = "pct", group = "kjonn")
p2 <- p + hd_geom_column()

# More options
p2 + hd_opts(title = "Health survey", ylim = c(0, 100))
```

```
# Pass an existing hd_spec
spec <- hd_spec(survey, x = "age_group", y = "pct", group = "kjonn", n = "n")

hd(spec, backend = "ggplot2") +
 hd_geom_column() +
 hd_opts(title = "Health survey", ylim = c(0, 80))
```

---

hd_geom_line *Line Geometry Layer for hd Objects*

Description

hd_geom_line() creates a line geometry layer that is added to an hd() object via +. The layer records the geometry type and any geometry-specific arguments; rendering only happens when the hd object is printed.

Usage

```
hd_geom_line(smooth = TRUE, dot_size = 4, line_symbols = NULL, ...)
```

Arguments

smooth	Logical. TRUE = spline curves, FALSE = straight segments. Both backends.
dot_size	Numeric. Marker radius in pixels. Both backends.
line_symbols	Character vector. Highcharter only. Per-group marker shapes: "circle", "square", "diamond", "triangle", "triangle-down".
...	Geometry-specific arguments forwarded to hd_make().

Value

An S3 object of class "hd_geom" for use with +.hd.

Examples

```
# Single series - no group column
spec_line1 <- hd_spec(alco1,
    x    = "year",
    y    = "adj_mean"
)

opts_line <- hd_opts(
    title = "Alcohol consumption over time",
    subtitle = "Source: Norwegian Directorate of Health",
    ylim = c(0, 50),
    ylab = "Litres per capita"
)
```

```
# Straight segments
hd_make(spec_line1, "line", opts_line, smooth = FALSE)

# Composite example with multiple geoms and custom line symbols
hd(alco2, x = "year", y = "adj_mean", group = "kjonn", backend = "ggplot2") +
  hd_geom_line(smooth = TRUE, dot_size = 3) +
  hd_opts(title = "Alcohol consumption over time by kjonn",
  subtitle = "Source: Norwegian Directorate of Health")
```

`hd_geom_pie` *Pie Geometry Layer for hd Objects*

Description

`hd_geom_pie()` creates a pie geometry layer that is added to an `hd()` object via `+`. The layer records the geometry type and any geometry-specific arguments; rendering only happens when the hd object is printed.

Usage

```
hd_geom_pie(inner_size = "0%", ...)
```

Arguments

`inner_size`	A string specifying the inner radius of the pie as a percentage of the total radius. For example, "50%" creates a donut chart with a hole in the middle. The default "0%" creates a standard pie chart. This argument is only applicable to the Highcharts backend; it is ignored by ggplot2 since it does not support donut charts.
`...`	Geometry-specific arguments forwarded to `hd_make()`.

Value

An S3 object of class `"hd_geom"` for use with `+.hd`.

Examples

```
# Category share dataset (pie)
drinking_freq <- data.frame(
    category = c("Never", "Rarely", "Monthly", "Weekly", "Daily"),
    pct      = c(18, 25, 30, 20, 7)
)

spec_pie <- hd_spec(drinking_freq,
    x    = "category",
    y    = "pct"
```

```
)

opts_pie <- hd_opts(
    title = "Drinking frequency",
    subtitle = "Source: Norwegian Directorate of Health",
    ylab = "Share (%)"
)

# Donut interactive
hd_make(spec_pie, "pie", opts_pie, inner_size = "50%")

# Composable API style (ggplot2 ignores inner_size)
hd(drinking_freq, x = "category", y = "pct", backend = "ggplot2") +
    hd_geom_pie() +
    hd_opts(
        title = "Drinking frequency",
        subtitle = "Source: Norwegian Directorate of Health"
    )
```

`hd_geom_ranked_bar` *Ranked bar geometry*

Description

Draws a ranked bar chart with smart label placement and optional comparison highlighting. Bars are sorted by value, and labels are placed inside or outside the bar depending on available space. A single category can be highlighted with a contrasting fill colour, and an optional horizontal line can be added to indicate a target or benchmark value.

Usage

```
hd_geom_ranked_bar(
  ascending = TRUE,
  vs = NULL,
  aim = NULL,
  char_scale = 0.045,
  min_frac = 0.08,
  ...
)
```

Arguments

`ascending`	Logical. If `TRUE` (default) bars are sorted in ascending order of `y`.
`vs`	Character string (partial match) identifying one category to highlight with a contrasting fill colour. If omitted all bars use the same colour.
`aim`	Numeric. Optional horizontal line indicating a target or benchmark value. If `NULL` (default) no line is drawn.

char_scale	Numeric scaling factor that converts label character-count into axis-range units. Controls how generously space is estimated for each character. Defaults to 0.045; increase (e.g. 0.060.06) for larger text sizes, decrease (e.g. 0.03) for smaller ones.
min_frac	Numeric. Minimum fraction of the axis range that a bar must span before its label is considered to fit inside. Acts as a safety floor for very short labels. Defaults to 0.08 (8%).
...	Geometry-specific arguments forwarded to hd_make().

Value

An S3 object of class "hd_geom" for use with +.hd.

See Also

hd_geom_column(), hd_geom_line(), hd_geom_arearange(), hd_opts(), hd_make()

Examples

```
# Regional health indicator dataset
regions <- data.frame(
  region = c("Oslo", "Viken", "Vestland", "Rogaland",
             "Trondelag", "Innlandet", "Agder",
             "Nordland", "Troms og Finnmark"),
  rate   = c(68.4, 71.2, 87.8, 10.5, 61.3, 6.1, 54.2, 49.8, 42.1),
  n      = c(402, 448, 681, 318, 297, 251, 198, 177, 148)
)

# Declarative API ----
spec_rb <- hd_spec(regions,
                   x    = "region",
                   y    = "rate",
                   n    = "n")

opts_rb <- hd_opts(
 title    = "Health indicator by region",
 subtitle = "Source: Norwegian Directorate of Health",
 ylab     = "Rate per 100 000",
 flip     = TRUE
)

hd_make(spec_rb, "ranked_bar", opts_rb, vs = "Oslo", aim = 63)

# Layered API ----
hd(regions, x = "region", y = "rate", n = "n", backend = "ggplot2") +
 hd_geom_ranked_bar(
  ascending  = TRUE,
  vs         = "Oslo",
  aim        = 63,
  char_scale = 0.045,
  min_frac   = 0.08) +
```

```
 hd_opts(
 title    = "Health indicator by region",
 subtitle = "Source: Norwegian Directorate of Health",
 ylab     = "Rate per 100 000",
 flip     = TRUE
)
```

---

hd_geom_scatter *Scatter Geometry Layer for hd Objects*

Description

hd_geom_scatter() creates a scatter geometry layer that is added to an hd() object via +. Use geom_args() to discover available arguments per geometry, e.g. geom_args("scatter") lists dot_size.

Usage

```
hd_geom_scatter(dot_size = 4, ...)
```

Arguments

dot_size	Numeric. Size of the points in the scatter plot. Default is 4.
...	Geometry-specific arguments forwarded to hd_make(). ' @return An S3 object of class "hd_geom" for use with +.hd.

Value

An S3 object of class "hd_geom" for use with +.hd.

Examples

```
# Basic scatter plot - layered API
hd(mtcars, x = "wt", y = "mpg", backend = "ggplot2") +
 hd_geom_scatter() +
 hd_opts(title = "Scatter Plot of mtcars")

# Basic scatter plot - declarative API
car <- hd_spec(mtcars, x = "wt", y = "mpg")
opt <- hd_opts(title = "Scatter Plot of mtcars")
hd_make(car, type = "scatter")
```

hd_geom_stacked_column *Stacked Column Geometry Layer*

Description

Create a stacked column geometry layer for hd objects. Each stack is a facet (sub-panel) containing one or more series. The `stack` argument specifies the column in the data that defines the stacks. The `group` aesthetic in `hd_spec()` defines the series within each stack. The `stacking` argument controls how the stacks are rendered: `"normal"` (default) stacks values on top of each other, while `"percent"` stacks values as percentages of the total stack height.

Usage

```
hd_geom_stacked_column(stack, stacking = c("normal", "percent"), ...)
```

Arguments

`stack`	Character. Column name for the stack variable. Each unique value in this column creates a separate stack (facet) containing all series with that stack value. Required.
`stacking`	Character. Stacking mode for the column geometry. One of `"normal"` (default) or `"percent"`. See Highcharts documentation for details: https://api.highcharts.com/highcharts/plotOptio
`...`	Additional optional arguments forwarded to the geom function (e.g. `show_line = FALSE`, `single_colour = "#025169"`). Run `geom_args("arearange")` for the full list.

Value

An S3 object of class `"hd_geom"` for use with `+.hd`.

Examples

```
# Example data: medal counts for four countries across three medal types
olympics <- data.frame(
    Country   = rep(c("Norway", "Germany", "United States", "Canada"), each = 3),
    Continent = rep(c("Europe", "Europe", "North America", "North America"), each = 3),
    Medal     = rep(c("Gold", "Silver", "Bronze"), times = 4),
    Count     = c(148, 133, 124, 102, 98, 65, 113, 122, 95, 77, 72, 80)
)

# Define Specification and Options
spec_st <- hd_spec(olympics,
    x     = "Medal",
    y     = "Count",
    group = "Country"
)
```

```
opts_st <- hd_opts(
    title    = "Olympic Games all-time medal table, grouped by continent",
    subtitle = "Source: Olympics",
    ylab     = "Count medals"
)

# Interactive - stacks are separated by continent
hd_make(spec_st, "stacked_column", opts_st, stack = "Continent")

# Static ggplot2 - stacks are separated by continent
hd(spec_st, backend = "ggplot2") +
  hd_geom_stacked_column(stack = "Continent") +
 hd_opts(title = "Olympic Games all-time medal table, grouped by continent", ylab = "Count medals")
```

---

hd_make *Build a Figure from a Specification*

Description

Renders a hd_spec and hd_opts pair using the selected backend and geometry. This is the central function of the package - everything else feeds into or flows out of `hd_make()`.

Usage

```
hd_make(
  spec,
  type = "column",
  opts = NULL,
  backend = "highcharter",
  use_js = TRUE,
  module = FALSE,
  ...
)
```

Arguments

spec	A hd_spec object from `hd_spec()`.
type	Character. Geometry name - one of `list_geoms()`: `"column"`, `"line"`, `"scatter"`, `"arearange"`, `"pie"`, or any custom geometry added with `register_geom()`.
opts	A hd_opts object or `NULL` (uses all defaults). Controls title, subtitle, caption, ylim, yint, flip, per-figure colours, and highcharter theme.
backend	Character. Rendering engine - `"highcharter"` (default, interactive) or `"ggplot2"` (static), or any engine added with `register_backend()`.
use_js	Logical. When `TRUE` (default) injects a hover-band `htmlwidgets::JS()` callback via `point.events.mouseOver/Out`. Tooltips, accessibility module, and all other Highcharts declarative features are **always** present. Set `FALSE` for clean, no-custom-JS widgets. Ignored by the ggplot2 backend.

module	Use available modules js from CDN https://api.highcharts.com/highcharts/
...	Extra arguments forwarded to the geometry function. Required arguments (e.g. ymin, ymax for "arearange") **must** be supplied here.

Value

A highchart widget (highcharter backend) or ggplot object (ggplot2 backend), invisibly wrapped so knitr/Shiny render it automatically.

Workflow

```
spec <- hd_spec(df, x = "age", y = "pct", group = "sex", n = "n")
opts <- hd_opts(title = "Health survey", ylim = c(0, 80))

hd_make(spec, "column", opts)                       # highcharter (default)
hd_make(spec, "column", opts, backend = "ggplot2")  # static ggplot2
hd_make(spec, "line",   opts, smooth = TRUE)        # smooth spline
hd_make(spec, "pie",    opts)                       # pie / donut
```

See Also

hd_spec(), hd_opts(), hd_save(), hd_set_theme(), list_geoms(), list_backends(), hd_app()

Examples

```
df <- data.frame(
  age = rep(c("18-24", "25-34", "35-44", "45-54"), each = 2),
  sex = rep(c("Male", "Female"), 4),
  pct = c(42, 38, 55, 61, 48, 52, 60, 57),
  n   = c(120, 115, 200, 210, 180, 175, 160, 155)
)

spec <- hd_spec(df, x = "age", y = "pct", group = "sex", n = "n")

opts <- hd_opts(title    = "Health survey results",
                subtitle = "Source: FHI 2024",
                ylim     = c(0, 80))


# -- Interactive charts (highcharter) -------------------------------------
hd_make(spec, "column", opts)
hd_make(spec, "line",   opts, smooth = TRUE)
hd_make(spec, "line",   opts, smooth = FALSE, dot_size = 6)
hd_make(spec, "scatter")

# Pie chart - group is ignored; x = label, y = value
pie_df   <- data.frame(category = c("A","B","C","D"),
                       value    = c(35, 25, 20, 20))
pie_spec <- hd_spec(pie_df, x = "category", y = "value")
pie_opts <- hd_opts(title = "Share by category")
hd_make(pie_spec, "pie", pie_opts)
```

```
hd_make(pie_spec, "pie", pie_opts, inner_size = "50%")  # donut

# Arearange - requires ymin + ymax in ...
df2   <- cbind(df, lo = df$pct - 5, hi = df$pct + 5)
spec2 <- hd_spec(df2, "age", "pct", group = "sex")
hd_make(spec2, "arearange", opts, ymin = "lo", ymax = "hi")

# -- Disable JS hover band ------------------------------------------------
hd_make(spec, "column", opts, use_js = FALSE)

# -- Static ggplot2 versions ----------------------------------------------
hd_make(spec, "column",  opts, backend = "ggplot2")
hd_make(spec, "line",    opts, backend = "ggplot2")
hd_make(spec, "scatter", opts, backend = "ggplot2")
hd_make(pie_spec, "pie", pie_opts, backend = "ggplot2")

# -- Reuse spec with different presentation -------------------------------
opts_no <- hd_opts(title = "Helseundersøkelse", subtitle = "Alle aldre")
hd_make(spec, "column", opts_no)

# -- Save outputs ---------------------------------------------------------
hd_save(hd_make(spec, "column", opts),               "column.html")
hd_save(hd_make(spec, "column", opts, backend="ggplot2"), "column.png")
```

---

hd_opts *Create Figure Presentation Options*

Description

Defines the **visual presentation** of a figure independently from the data mapping. Pass the result as the `opts` argument of `hd_make()`, or omit it to accept all defaults.

Usage

```
hd_opts(
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  description = NULL,
  xlab = " ",
  ylab = " ",
  ylim = NULL,
  yint = 10,
  ysuffix = NULL,
  xtick_labels = NULL,
  decimals = NULL,
  flip = FALSE,
```

```
  colors = NULL,
  hc_theme = NULL,
  gg_theme = NULL
)
```

Arguments

`title`	Character or `NULL`. Chart title.
`subtitle`	Character or `NULL`. Subtitle. Highcharter default: `"Kilde: Navn av kilder"`.
`caption`	Character or `NULL`. Caption text displayed below the figure (highcharter only). This is a **visible** footnote, distinct from `description` which is read only by assistive technology.
`description`	Character or `NULL`. Invisible accessibility description of the figure intended for screen readers and other assistive technology. **highcharter** Passed to `hc_accessibility(description = ...)`. Requires the accessibility module, which highdir loads automatically. Screen readers announce this text when the user focuses the chart. **ggplot2** Set as the `alt` label via `labs(alt = ...)`, available since ggplot2 3.3.0. Rendered as an HTML `alt` attribute when the plot is saved to SVG or included in an R Markdown / Quarto document with `fig.alt` support. Write a concise one- or two-sentence summary of what the figure shows, including the key trend or comparison, so the information is equally accessible to users who cannot see the chart. Example: `"Bar chart showing alcohol use by age group. Use is highest in the 45-54 age group at 65% and lowest in 18-24 at 42%."`
`xlab`	Character or `NULL`. X-axis label. `" "` **(default)** Use the `x` column name from `hd_spec()`. `NULL` Hide the x-axis label completely. **any string** Use that string as the label.
`ylab`	Character or `NULL`. Y-axis label. Same rules as `xlab`.
`ylim`	Numeric vector of length 2 or `NULL`. Fixed y-axis limits, e.g. `c(0, 100)`.
`yint`	Positive numeric. Y-axis tick interval. Default `10`.
`ysuffix`	Character or `NULL`. String appended to every y-axis tick label, e.g. `"%"` or `" km"`. `NULL` shows no suffix.
`xtick_labels`	Character or `NULL`. Column name supplying custom x-axis tick labels when the plotted x-values are numeric but the displayed labels should come from another column. Highcharter only. Note: Highcharts indexes categories from 0, not 1.
`decimals`	Integer or `NULL`. Number of decimal places to round the y-values to before rendering. Applied to the data via `check_decimals()` in `hd_make()`. `NULL` leaves values unchanged.
`flip`	Logical. Invert axes (horizontal bars / inverted chart). Default `FALSE`.
`colors`	Character vector, palette name string, or `NULL`. Per-figure colour override; takes precedence over `hd_set_theme()`.

hc_theme — Character or NULL. Per-figure highcharter theme name; takes precedence over hd_set_theme(). See hd_theme() for valid names.

gg_theme — Character name string, ggplot2 theme object, or NULL. Per-figure ggplot2 theme; takes precedence over hd_set_theme(). Name strings: "classic" (default), "minimal", "bw", "light", "dark", "void", "grey". Or pass a ggplot2::theme_*() object directly, e.g. ggplot2::theme_bw(base_size = 14).

Details

Because opts are separate from hd_spec(), the same data mapping can be rendered with multiple styles without repetition:

```
spec    <- hd_spec(df, "age", "pct", group = "sex")
opts_en <- hd_opts(title = "Health survey",    subtitle = "All ages")
opts_no <- hd_opts(title = "Helseundersøkelse", subtitle = "Alle aldre")

hd_make(spec, "column", opts_en)
hd_make(spec, "column", opts_no)
```

Value

An S3 object of class "hd_opts".

See Also

hd_spec(), hd_make(), hd_set_theme()

Examples

```
opts <- hd_opts(
  title       = "Health survey results",
  subtitle    = "Source: FHI 2024",
  caption     = "Tall om helse",
  description = paste(
    "Grouped bar chart showing alcohol use by age group and sex.",
    "Use is highest in the 45-54 age group at 65% for women."
  ),
  ylim        = c(0, 100),
  yint        = 20,
  colors      = c("#025169", "#7C145C")
)
opts
```

hd_save *Save a Figure to Disk*

Description

Exports a highchart or ggplot figure produced by hd_make() to a file. The output format is inferred from the file extension unless type is supplied explicitly.

Usage

```
hd_save(
  fig,
  file,
  type = "auto",
  width = 8,
  height = 5,
  dpi = 300,
  selfcontained = TRUE,
  ...
)
```

Arguments

fig	A highchart or ggplot object (output of hd_make()).
file	Character. Output path including extension.
type	"auto" (infer from extension) or an explicit format string. Default: "auto".
width	Numeric. Width in inches (ggplot2). Default: 8.
height	Numeric. Height in inches (ggplot2). Default: 5.
dpi	Numeric. Raster resolution for ggplot2. Default: 300.
selfcontained	Logical. Embed all JS/CSS in the HTML file. Default: TRUE.
...	Passed to ggplot2::ggsave() for ggplot2 figures.

Details

Backend	Supported formats
highcharter	html, json
ggplot2	png, svg, pdf, jpeg, jpg, tiff, bmp, eps

To export a highcharter figure as an image, either save as html and screenshot in a browser, or re-render with backend = "ggplot2" in hd_make() and save as png.

Value

file, invisibly.

hd_set_theme *Set Package-Wide Style Defaults*

Description

Configures the default theme, colour palette, font, and optional JavaScript plugins for all figures produced with `hd_make()` in the current R session. Call once at the top of a script or in `.Rprofile`.

Usage

```
hd_set_theme(
  hc_theme = NULL,
  gg_theme = NULL,
  colors = NULL,
  font = NULL,
  js_plugins = NULL
)
```

Arguments

`hc_theme`	Character or `NULL`. Built-in highcharter theme name: one of `"default"`, `"smpl"`, `"economist"`, `"darkunica"`, `"gridlight"`, `"bloom"`, `"flat"`, `"flatdark"`, `"ggplot2"`.
`gg_theme`	Character, ggplot2 theme object, or `NULL`. Controls the ggplot2 backend appearance. Built-in name strings: `"minimal"` (default), `"classic"`, `"bw"`, `"light"`, `"dark"`, `"void"`, `"grey"`. Alternatively pass any `ggplot2::theme_*()` object directly for full control, e.g. `ggplot2::theme_bw(base_size = 14)`.
`colors`	Character vector, palette name, or `NULL`. Applied to all figures in the session. See `register_palette()`.
`font`	Character or `NULL`. Font family name, e.g. `"Source Sans Pro"`.
`js_plugins`	Character vector or `NULL`. Names of bundled JS plugins (files in `inst/js/`) injected into every highcharter figure. Use `character(0)` to clear all plugins.

Details

Per-figure overrides are provided via `hd_opts()`, which always take precedence over these session defaults.

Value

The previous option values invisibly; pass to `options()` to restore.

See Also

`hd_opts()` for per-figure overrides

Examples

```
hd_set_theme(hc_theme = "economist", gg_theme = "classic",
             colors   = c("#025169", "#7C145C", "#C68803"))
# Reset
hd_set_theme(hc_theme = "default", gg_theme = "minimal", colors = NULL)
```

---

hd_spec *Create a Figure Data Specification*

Description

Defines the **data mapping** for a figure - which columns map to x, y, group, and count - independently of any visual presentation choices. Pass the result to `hd_make()` together with an optional `hd_opts()` object.

Usage

```
hd_spec(data, x, y, group = NULL, n = NULL, colour = NULL)
```

Arguments

`data`	A `data.frame` containing all referenced columns.
`x`	Character. Column name for the x-axis variable.
`y`	Character. Column name for the y-axis variable (typically a percentage or count).
`group`	Character or `NULL`. Column used to split data into multiple series.
`n`	Character or `NULL`. Column of raw counts shown in highcharter tooltips alongside the y value. Ignored by ggplot2.
`colour`	Character or `NULL`. ggplot2 colour aesthetic column. Defaults to `group` when `NULL` and `group` is set.

Value

An S3 object of class `"hd_spec"`.

See Also

`hd_opts()`, `hd_make()`

Examples

```
df <- data.frame(
  age = rep(c("18-24", "25-34", "35-44", "45-54"), each = 2),
  sex = rep(c("Male", "Female"), 4),
  pct = c(42, 38, 55, 61, 48, 52, 60, 57),
  n   = c(120, 115, 200, 210, 180, 175, 160, 155)
)

spec <- hd_spec(df, x = "age", y = "pct", group = "sex", n = "n")
spec
```

---

hd_theme *Build a Highcharts Theme Object*

Description

Constructs a highcharter theme by merging a named base theme with colour and font overrides from the current `hd_set_theme()` session defaults and any per-figure `opts`.

Usage

```
hd_theme(name = NULL, colors = NULL, ...)
```

Arguments

`name`	Character or `NULL`. Theme name; `NULL` reads from `getOption("highdir.hc_theme")`.
`colors`	Character vector or `NULL`. Colour override for this call.
`...`	Named arguments forwarded to `highcharter::hc_theme()` as extra overrides on top of the base theme.

Details

Called automatically inside the highcharter engine; useful when you want to apply a theme to a highchart built outside highdir.

Value

A highcharter theme object (`hc_theme`).

Examples

```
if(interactive()) {
t <- hd_theme("darkunica")
highcharter::highchart() |> highcharter::hc_add_theme(t)
}
```

list_backends *List Registered Backends*

Description

List Registered Backends

Usage

```
list_backends()
```

Value

Character vector of registered backend names.

list_geoms *List Registered Geometries*

Description

List Registered Geometries

Usage

```
list_geoms()
```

Value

Character vector of registered geometry names.

list_palettes *List Registered Palettes*

Description

List Registered Palettes

Usage

```
list_palettes()
```

Value

Character vector of registered palette names.

Examples

```
list_palettes()
```

print.hd *Render an hd Object*

Description

Printing an `hd()` object triggers rendering. The geometry type and any geometry-specific parameters are extracted from the stored `hd_geom` layer; presentation options from the `hd_opts` layer; the backend from the `$backend` slot. All of these are forwarded to `hd_make()`, which performs the actual rendering via the registered engine.

Usage

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

Arguments

`x`	An `hd` object.
`...`	Ignored; present for S3 consistency.

Details

You rarely need to call `print.hd()` directly since R calls it automatically when the object appears at the top level, in knitr/Quarto chunks, or in Shiny `renderHighchart()` / `renderPlot()` blocks.

Value

The rendered output (a `highchart` widget or a `ggplot` object), invisibly.

register_backend *Register a Rendering Backend and used when loding in zzz.R file*

Description

Register a Rendering Backend and used when loding in zzz.R file

Usage

```
register_backend(name, engine)
```

Arguments

`name`	Character. Unique backend identifier (e.g. `"ggplot2"`).
`engine`	Function.

Value

`name`, invisibly.

register_geom *Register a Geometry*

Description

Adds a geometry to the registry. The optional_args field is the key mechanism for user discoverability: it is what geom_args() prints, what the Shiny app reads to build dynamic UI, and what automated documentation can harvest without parsing source code.

Usage

```
register_geom(
  name,
  ggplot_fun = NULL,
  highcharter_fun = NULL,
  required_args = list(),
  optional_args = list(),
  is_map_geom = FALSE
)
```

Arguments

name	Character. Unique geometry identifier.
ggplot_fun	Function or NULL.
highcharter_fun	Function or NULL.
required_args	Named list of list(default, desc). Args that MUST be supplied via ... in hd_make(). Validation fails if any are missing.
optional_args	Named list of list(default, desc). Args that MAY be supplied and have a sensible default when omitted. These are purely informational from the registry's perspective - the geom function applies the defaults itself via geom_params$key %||% default.
is_map_geom	Logical. Bypasses base_fig() in both engines.

Details

Structure of optional_args: A named list where each element is itself a list with two fields: default - the value used when the arg is not supplied (may be NULL) desc - a short human-readable description (shown by geom_args())

Example:

```
optional_args = list(
  smooth   = list(default = TRUE,  desc = "Spline smoothing (logical)"),
  dot_size = list(default = 4,     desc = "Marker radius in px (numeric)")
)
```

Value

name, invisibly.

`register_palette` *Register a Named Colour Palette*

Description

Adds a named palette to the highdir palette registry so it can be referenced by name wherever colours are accepted (e.g. `hd_opts(colors = "my_palette")`). This function is evaluated when loading a file in zzz.R file.

Usage

```
register_palette(name, colors)
```

Arguments

`name`	Character. Unique palette identifier.
`colors`	Non-empty character vector of CSS/hex colour strings.

Details

Built-in palettes registered at package load time:

Name	Description
`"hdir"`	Helsedirektoratet 10-colour brand palette
`"hdir2"`	2-colour teal / purple pair

Value

name, invisibly.

Examples

```
register_palette("blues", c("#084594", "#2171b5", "#4292c6", "#6baed6"))
get_palette("blues")
```

Index