



sdvplotR :: CHEAT SHEET

Team logos, wordmarks, player headshots and team colours for **28 leagues** (pro, college, minor and junior hockey, spring football, soccer, cricket) in **ggplot2** plots, **gt** and **reactable** tables, and **plotly**, **ggraph** and **leaflet**, behind one `league` argument. Built on **ggpath**, after the approach of **nflplotR**, **cfbplotR**, **nbaplotR** and **mlbplotR**. Page 1: ggplot2 marks, scales and axes. Page 2: teams, interactive, surfaces, coordinates. Page 3: gt images, themes, colour. Page 4: gt layout, saving, reactable.

sdvplotR v0.1.0

R ≥ 4.1 · ggplot2 ≥ 4.0 · ggpath ≥ 1.1

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GGPLOT2 · MARKS & SCALES

INSTALL & LOAD

```
install.packages("sdvplotR")
# dev: sportsdataverse.r-universe.dev
pak::pak("sportsdataverse/sdvplotR")
library(sdvplotR); library(ggplot2)
```

Every team helper takes `league`, a key from `sdv_leagues()` ("nfl" "nba" "cfb" "nhl" "ahl" "soccer" ..., page 2). Teams: any abbreviation, name or id `sdv_resolve()` reads (ESPN, nflverse, nba_api, MLB Stats, CFBD, ...), or a conference ("SEC", "B1G", "AFC").

Cookbook: [Getting started ↗](#)

LOGOS, WORDMARKS & HEADSHOTS

Geom	Required aes
<code>geom_sdv_logos()</code>	<code>x y team</code>
<code>geom_sdv_wordmarks()</code>	<code>x y team (NFL, MLB, MiLB)</code>
<code>geom_sdv_headshots()</code>	<code>x y player_id</code>
<code>geom_from_path()</code>	<code>x y path (any image)</code>

Optional aes and defaults: `alpha`, `colour` ("b/w" greyscales), `angle` = 0, `hjust` = 0.5, `vjust` = 0.5, `width` = 1, `height` = 1 (npc; a typical logo is `width` = 0.075). Logos and wordmarks also take `season` (an aes or a constant).

```
df <- data.frame(
  team = sdv_teams("nba")$abbr,
  a = rnorm(30), b = rnorm(30))
ggplot(df, aes(a, b)) +
  geom_mean_lines(aes(x0 = a, y0 = b)) +
  geom_sdv_logos(aes(team = team),
    league = "nba", width = 0.07)
```

HIGHLIGHT A GROUP

```
nfl <- subset(sdv_example_standings,
  league == "nfl")
west <- c("DEN", "KC", "LAC", "LV")
ggplot(nfl, aes(points_for,
  points_against)) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, !team %in% west),
    league = "nfl", width = 0.05,
    alpha = 0.25) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, team %in% west),
    league = "nfl", width = 0.1)
```

Rows draw in order, so the last layer and row end on top. `colour` = "b/w" greys logos instead of fading them, which reads better on a dark background.

Cookbook: [Logos, wordmarks, headshots ↗](#)

GEOM ARGUMENTS

```
geom_sdv_logos(mapping = NULL,
  data = NULL, stat = "identity",
  position = "identity", ..., league,
  id_system = "auto", strict = FALSE,
  na.rm = FALSE, show.legend = FALSE,
  inherit.aes = TRUE)
geom_sdv_headshots(..., league,
  id_system = "espn")
```

`geom_sdv_wordmarks()` takes the logo arguments; headshots take no `season` or `strict`. Headshot `id_system`: "espn" = ESPN athlete ids in every league; "gsis" = nflverse NFL ids; "league" = NBA / WNBA Stats PERSON_ID, MLBAM, NHL API.

The ggproto objects `GeomSDVLogo`, `GeomSDVwordmark`, `GeomSDVheadshot` and `ggpath`'s `GeomFromPath`, `GeomRefLines` are exported for extensions.

ERA LOGOS

```
old <- data.frame(x = 1:4, y = 1,
  team = c("QUE", "HFD", "ATL", "WPG"),
  season = c(1990, 1995, 2005, 2015))
ggplot(old, aes(x, y)) +
  geom_sdv_logos(aes(team = team,
    season = season), league = "nhl",
    width = 0.2) + theme_void()
```

`season`: the mark worn that year (ending year for split seasons); it also picks who held a reused code. NHL: every identity since 1917-18 (QUE HFD); AHL, ECHL, OHL, QMJHL, WHL, USHL, PWHL, PHF, UFL, XFL, AAF: dated marks; NFL / WNBA: relocated identities (STL SD OAK; SA TUL). Elsewhere today's mark. NHL marks are SVG: needs `rsvg`.

REFERENCE LINES

```
p + geom_mean_lines(aes(x0 = a, y0 = b))
p + geom_median_lines(aes(x0 = a),
  colour = "grey40", linetype = 1)
```

From `ggpath`. Optional aes `x0`, `y0`; defaults `colour` = "red", `linewidth` = 0.5, `linetype` = 2 (dashed), `alpha` = NA.

ANY IMAGE: GEOM_FROM_PATH

```
df$url <- sdv_logo_url(df$team, "nba",
  variant = "dark")
ggplot(df, aes(a, b)) +
  geom_from_path(aes(path = url),
    width = 0.07)
```

`path` is a URL or a local file; same optional aes as the team geoms. Conferences resolve by key or other name: `sdv_logo_url(c("SEC", "B1G"), "cfb")`; a team's code wins (cfb "MAC" is Macalester).

COLOUR & FILL SCALES

```
scale_color_sdv(league, which = "primary",
  values = NULL, ..., aesthetics = "colour",
  breaks = waiver(), na.value = "grey50",
  guide = NULL, alpha = NA)
scale_fill_sdv(...) # aesthetics = "fill"
```

Maps team keys to official colours; `which` = "secondary" for the second colour, `alpha` fades them (`scales::alpha()`), `values` replaces them, ... goes to `scale_colour_manual()`. `scale_colour_sdv()` is an alias.

```
ggplot(df, aes(team, b)) +
  geom_col(aes(colour = team, fill = team)) +
  scale_color_sdv("nba", "secondary") +
  scale_fill_sdv("nba", alpha = 0.4)
```



NFL primary / secondary, as `sdv_team_colors("nfl")`, `which` = "all" returns them.

Cookbook: [Team colours and themes ↗](#)

TITLES

```
p + ggtitle_image(title_image = "SEA",
  title = "Seattle Seahawks Analysis",
  image_height = 20, image_side = "left",
  league = "nfl") +
  theme_title_image(size = 20, hjust = 0.5)
```

`title_image`: a team key (logo resolved) or any image URL. `image_height` = 15 px by default; `image_side` = "left" or "right"; subtitle. `theme_title_image(...)` is required and passes ... to `ggtext::element_markdown()`.

LOGO AT EACH BAR'S END

```
nba <- subset(sdv_example_standings,
  league == "nba")
ggplot(nba, aes(wins,
  reorder(team, wins))) +
  geom_col(aes(fill = team)) +
  geom_sdv_logos(aes(x = wins + 3,
    team = team), league = "nba",
    height = 0.028) +
  scale_fill_sdv("nba")
```

`height` sizes the logo here: each row is about a thirtieth of the panel.

LOGO AXES

Route	Calls
theme	<code>element_sdv_logo()</code> · <code>element_sdv</code>
element	<code>_wordmark()</code> · <code>element_sdv_headshot()</code>
scale+ theme	<code>scale_x_sdv()</code> / <code>scale_y_sdv()</code> · <code>scale_x_sdv_headshots()</code> + <code>theme_x_sdv()</code> / <code>theme_y_sdv()</code>

```
# element route (size in cm)
ggplot(df, aes(team, b)) + geom_col() +
  theme(axis.text.x =
    element_sdv_logo("nba", size = 1))
```

`element_sdv_logo(league, alpha = NULL, colour = NA, hjust = NULL, vjust = NULL, size = 0.5, season, id_system)`; `_wordmark()` the same; `_headshot()` takes `id_system`, no `season`.

```
# scale route (size in px; needs ggtext)
ggplot(df, aes(b, team)) + geom_col() +
  scale_y_sdv(league = "nba", size = 12) +
  theme_y_sdv()
```

`scale_x_sdv(..., league, expand, guide, position = "bottom", size = 12, season, id_system)`; `scale_y_sdv()` has position = "left". Size is the height on x, the width on y. `scale_x_sdv_headshots()` (size = 20) and `scale_y_sdv_headshots()` (size = 30) take `id_system`. `theme_x_sdv()` / `theme_y_sdv()` render those labels as markdown.

Cookbook: [Logo axes and titles ↗](#)

HEADSHOTS ON A SCATTER

```
ggplot(scorers, aes(ppg, ts)) +
  geom_sdv_headshots(aes(player_id =
    athlete_id), league = "nba",
    height = 0.12)
```

ESPN ids: the `athlete_id` of `hoopR`, `wehoop`, `cfbfastR` rosters. NFL `gsis` ids (`id_system` = "gsis") go through a headshot map built from nflverse rosters (1999 on); `sdv_clear_cache()` re-reads it.

IMAGES IN THEME ELEMENTS

```
p + theme(plot.background =
  element_sdv_raster("court.png"))
```

`element_sdv_raster(image_path, x, y, width, height, just = "centre", hjust = 0.5, vjust = 0.5, interpolate = TRUE, ...)`: one image in a rectangle; wraps `ggpath`'s `element_raster()` (also exported).

`element_path(alpha = 1, colour = NA, hjust, vjust, angle = 0, size = unit(0.5, "cm"))`: axis text that already holds image paths or URLs.



sdvplotR :: TEAMS, INTERACTIVE & SURFACES

The shared SportsDataverse team index behind every helper: **28 leagues**, any team identifier resolved to one canonical `team_id`, colours, palettes and archived logo URLs. Plus logos on [plotly](#), [ggiraph](#) and [leaflet](#), premade tier plots, team-coloured playing surfaces drawn with [sportyR](#), and converters that put shot and event coordinates from stats.nba.com, Euroleague and ten soccer providers on one frame.

sdvplotR v0.1.0

sportyR for surfaces · rsvg for SVG logos

GALLERY ↗

COOKBOOKS ↗

TEAM DATA · LEAGUES, KEYS, COLOURS

LEAGUES & TEAMS

```
sdv_leagues() # the 28 league keys
sdv_teams("nfl")
sdv_teams("cfb", include_conferences = TRUE)
```

Group	sdv_teams()	columns
ids, names	league team_id abbr name short_name location	
grouping	program conference_id conference	
colours	color_primary color_secondary color_source	

`team_id` is the canonical key (the ESPN id where ESPN has one). Conference rows ("SEC", "AFC") draw logos only.

RESOLVE ANY IDENTIFIER

```
sdv_resolve(c("GSW", "Golden State Warriors",
"XYZ"), "nba")
#> "9" "9" NA + one warning
sdv_resolve(c("OAK", "SD", "STL"), "nfl")
#> "13" "24" "14" LV, LAC, LA
sdv_suggest("Kansas Cty Chiefs", "nfl")
```

`sdv_resolve(x, league, season = NULL, id_system = "auto", strict = FALSE)` reads abbreviations, names and ids from ESPN, nflverse, MLB Stats, nba_api, NHL, HockeyTech, NCAA, PFF, CFBD, Sports Reference, FanGraphs and more. `id_system` names one ("nhl_id" only explicitly). `season` picks who held a reused code (ending year for split seasons). Unresolved: NA and one warning (`strict = TRUE` errors). `sdv_suggest(x, league, n = 5)` lists candidates, never picks.

Every helper that takes a team resolves it this way, with `league` and `id_system`. `strict`: the logo-URL, colour, tier and mark-drawing helpers; not `sdv_marks()`, `element_sdv_*`(), `scale_x/y_sdv()`, `sdv_colorway()`, `sdv_team_factor()`.

TEAM FACTOR

```
sdv_team_factor(c("KC", "BUF", "invalid"),
league = "nfl")
#> KC BUF <NA> Levels: BUF KC
```

Levels ordered by abbreviation; unresolved values become NA , so scales drop them.

COLOURS & PALETTES

```
sdv_team_colors("nfl", "KC")
sdv_team_colors("nfl", c("KC", "BUF"),
which = "all")
sdv_team_colors("nba", which = "all")
sdv_color_palette("nba",
teams = c("BOS", "NY"))
```

```
sdv_team_colors(league, team = NULL, which =
"primary", season, id_system, strict,
include_conferences = FALSE) : a named hex vector ( NA
where a team does not resolve); which = "all" gives
"primary, secondary" strings, or with team = NULL a
data frame team_id abbr name primary secondary.
sdv_color_palette(league, teams = NULL, which =
"primary", ...): named colours for scale_*_manual() .
```

IMAGE URLS & MARKS

```
sdv_logo_url(c("KC", "Buffalo Bills",
"OAK"), "nfl")
sdv_logo_url("NYJ", "nfl",
variant = "scoreboard")
sdv_logo_url("QUE", "nhl", season = 1990)
sdv_logo_url("KC", "nfl",
mark_type = "wordmark")
sdv_marks("KC", "nfl") # every mark
sdv_headshot_url("2544", "nba",
id_system = "league")
```

`sdv_logo_url(team, league, season = NULL, variant = "default", mark_type = "logo", id_system, strict) : the SportsDataverse archive's immutable copy. variant: "default", "dark", or a name from sdv_marks() ("scoreboard", NFL "grayscale", MLB "cap_on_light", ...); a team without it gets its closest mark of the same polarity, an unknown name errors. mark_type = "wordmark": NFL, MLB (and MiLB).`

`sdv_headshot_url(player_id, league, id_system = "espn") : ESPN headshots for nfl nba wnba mlb nhl cfb mbb wbb; "gsis" and "league" as on page 1. NA for a missing or malformed id.`

CACHE & VERSIONS

Index and logo manifest come from the SportsDataverse CDN, cached for a day; a copy ships in the package (`options(sdvplotR.offline = TRUE)`). `sdv_versions()` : what is loaded, from where. `sdv_clear_cache()` : drop index, manifest, NFL headshot map and ggpath images.

COVERAGE

Leagues	Teams
nfl nba mlb nhl wnba	32 · 30 · 30 · 32 · 15
nbagl milb pwhl phf	32 · 286 · 12 · 8
ufl xfl usfl aaf	11 · 11 · 9 · 8
ahl echl ohl whl qmjhl ushl	61 · 59 · 27 · 23 · 30 · 17
cfb mbb wbb	687 · 366 · 364
ncaa_baseball softball mhokey hockey	414 · 415 · 110 · 48
soccer · cricket	2,631 · 111

Team rows in the bundled index 0aeef853ec2b ; the live index can differ.

PREMADE PLOT: TEAM TIERS

```
tiers <- data.frame(
tier_no = c(1, 1, 2, 2, 3),
team = c("KC", "DET", "BAL", "PHI", "BUF"))
sdv_team_tiers(tiers, league = "nfl",
theme = "light", no_line_below_tier = 2)
```

`sdv_team_tiers(data, league, title, subtitle, caption = NULL, tier_desc, presort = FALSE, alpha = 0.8, width = 0.075, no_line_below_tier = NULL, devel = FALSE, theme = "dark", variant = "auto", season, id_system, strict) . data needs tier_no and team (optional tier_rank). Title "NFL Team Tiers" from league; NULL drops one. devel = TRUE plots keys, not logos; "auto": dark marks on the dark theme.`

Cookbook: [College hoops tiers](#) ↗

INTERACTIVE: GGIRAPH, PLOTLY, LEAFLET

```
p <- ggplot(d, aes(x, y)) +
geom_sdv_logos_interactive(aes(team = team,
tooltip = team, data_id = team),
league = "nfl", width = 0.1)
ggiraph::girafe(ggobj = p)
plotly::plot_ly(d, x = ~x, y = ~y) |>
add_sdv_logos(d$x, d$y, d$team, "nfl") |>
sdv_colorway("nfl", d$team)
leaflet::leaflet(d) |> leaflet::addTiles() |>
addSdvLogoMarkers(~lon, ~lat, ~team,
league = "nfl")
```

`geom_sdv_{logos,wordmarks,headshots}_interactive()` : images plus an invisible hover point (`hover_size = 8`). `plotly`: `add_sdv_logos(p, x, y, team, league, size = 0.08, ...)`, `add_sdv_wordmarks()`, `add_sdv_headshots()`, `sdv_axis_logos(p, teams, league, axis)` tick logos, `sdv_colorway()`; `ggplotly()` drops them. `addSdvLogoMarkers(..., size = 32)`: no logo keeps the default marker.

Cookbook: [Interactive plots](#) ↗

PLAYING SURFACES

```
sdv_surface("nba", "BOS",
center_logo = TRUE) +
geom_point(aes(x, y), data = shots)
```

league	sportyR surface
nfl aaf ufl usfl xfl cfb	geom_football("nfl" / "ncaa")
nba nbagl wnba mbb wbb fiba	geom_basketball(); fiba 28 × 15 m
mlb milb ncaa_baseball	geom_baseball()
nhl ahl echl ohl qmjhl ushl pwhl phf ncaa_mhokey ncaa_whockey	geom_hockey(); whl on the NHL rink
soccer	geom_soccer("fifa"), 105 × 68 m

`sdv_surface(league, team = NULL, season = NULL, center_logo = FALSE, ...)` (not cricket, ncaa_softball). A `team` paints the paint and apron, end zones, or centre line, circle and boards; baseball stays regulation, soccer gets a centre logo only. Stylized, not real floors. Origin at centre, in feet (yards for football, metres for soccer and FIBA); ... goes to `sportyR` (`rotation`, `xlims`, `color_updates`).

Cookbook: [Surfaces and shot charts](#) ↗

SHOT & EVENT COORDINATES

```
# stats.nba.com legacy frame -> feet
shots |> sdv_court_coords()
# Euroleague (cm, hoop origin) -> FIBA m
pts |> sdv_court_coords(
x_column = "coord_x", y_column = "coord_y",
provider = "euroleague")
# soccer events -> 105 x 68 m pitch
ev |> sdv_pitch_coords("statsbomb")
```

`sdv_court_coords(data, x_column = "x_legacy", y_column = "y_legacy", provider = "nba")` adds `court_x` `court_y`: basket at -41.75 ft (-12.425 m for Euroleague), half court at 0; `court_y` -25 to 25 ft, negative = shooter's left. From hoopR's `nba_shotchartdetail()` : `sdv_court_coords(res$Shot_Chart_Detail, "LOC_X", "LOC_Y")` .

`sdv_pitch_coords(data, provider, x_column = NULL, y_column = NULL, flip = NULL, pitch_length = NULL, pitch_width = NULL)` adds `pitch_x` `pitch_y` in metres from the centre spot (`pitch_x` -52.5 = defended goal line). `flip`: TRUE / FALSE or a logical column, as `flip = "away"`, turns rows half a turn. `pitch_length / pitch_width`: tracking only. Providers: `opta` (statsperform) `wyscout` `statsbomb` `uefa` `impect` `espn`, tracking `tracab` `skillcorner` `secondspectrum` `metrca` .

Cookbook: [NHL shot map](#) ↗ · [Soccer](#) ↗



sdvplotR :: gt TABLES

Team images and colours for **gt**, plus the **gtUtils** toolkit (by Andrew Weatherman): 20 themes, legends, colouring, annotations, layouts and image export. Every helper takes a `gt_tbl` as its first argument (`gt_object`) and returns one, so they chain with `|>`. Examples run on `sdv_example_standings` (real NFL and NBA final standings). Page 3: images, themes, colour, legends, focus. Page 4: format, layout, saving, reactable.

sdvplotR v0.1.0

gt ≥ 0.10 · paletteer optional

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GT · IMAGES, THEMES, COLOUR

TEAM IMAGES IN CELLS

```
gt_sdv_logos(gt_object, columns, league,
  height = 30, locations = NULL,
  include_name = FALSE, season = NULL,
  id_system = "auto", strict = FALSE)
gt_sdv_wordmarks(gt_object, columns,
  league, height = 30, locations = NULL,
  season = NULL, id_system, strict)
gt_sdv_headshots(gt_object, columns,
  league, height = 30, locations = NULL,
  id_system = "espn")
gt_sdv_cols_label(gt_object,
  columns = everything(), league,
  height = 30, mark_type = "logo",
  id_system, season, strict)
```

`height` in px. `locations` (`cells_body()`, `cells_stub()`, `cells_column_labels()`, `cells_row_groups()`) replaces `columns`. `include_name` = `TRUE` keeps the text after the logo; `season` shows one season's marks in every cell. Wordmarks: NFL, MLB (and MiLB). `gt_sdv_cols_label()` turns column *labels* into images: `mark_type` = "logo" "wordmark" "headshot".

STACKED NAME IN TEAM COLOUR

```
gt_merge_stack_team_color(gt_object,
  col1, col2, team_color, league,
  font_size_top = 14,
  font_size_bottom = 12, color = "black",
  background = NULL, season, id_system)
```

`col1` on top in bold caps, `col2` below in the team's colour; a primary under 4.5:1 contrast gives way to the secondary or is darkened. Apply a theme first: `background` = `NULL` reads it.

```
x <- subset(sdv_example_standings,
  league == "nfl",
  c(team, team_name, division))
gt(x) |>
  gt_merge_stack_team_color(team_name,
    division, team, league = "nfl")
```

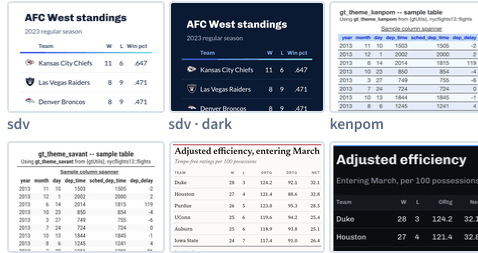
EXAMPLE

```
library(gt)
nfc <- subset(sdv_example_standings,
  division == "NFC Wins",
  c(team, team_name, wins, losses,
    last_season_wins))
gt(nfc) |>
  gt_sdv_logos(columns = team,
    league = "nfl", height = 28) |>
  gt_theme_sdv()
```

Cookbook: [Team tables with the gt toolkit ↗](#)

THEMES

```
gt_theme_sdv(gt, style = "light")
gt_theme_sdv_team(gt, team = "KC",
  league = "nfl")
```



`sdv` `sdv · dark` `kenpom`

`savant` `broadsheet` `midnight`

`gt_theme_sdv()`: Chivo titles and labels, a Lato body in tubular figures, and the SDV gradient (#3346F0 → #7FE6DC) as one horizon line under the labels; `style` = "dark" sets it on the SportsDataverse navy.

`gt_theme_sdv_team()`: the sdv table in one team's colours (title block primary, horizon line secondary); `team` = `NULL` keeps SportsDataverse colours; takes `season`, `id_system`.

Every `gt_theme_<name>(gt_object, ..., density)` takes `density` = "comfortable" "compact" "social" (default comfortable; compact for almanac, scoreboard, terminal) and passes ... to `gt::tab_options()`, applied last.

Option	Themes (default)
accent	almanac "#8C2F1E" · booktabs, swiss, tuftes "#111111" · broadsheet "#A6081A" · brutalist "#FF3B00" · midnight "#5B8DEF" · scoreboard "#0E1621" · terminal "#FFB86C"
style	sdv, sofa "light" / "dark" · tier "dark" / "light"
other	drench color = "#123F5E" · broadsheet paper = "white" or "salmon" · almanac stripe (NA = off)
none	athletic gtutils kenpom ncaa pl savant

All 20: almanac athletic booktabs broadsheet brutalist drench gtutils kenpom midnight ncaa pl savant scoreboard sdv sdv_team sofa swiss terminal tier tuftes (each `gt_theme_<name>()`).

```
gt(nfc) |>
  gt_theme_broadsheet(paper = "salmon",
    density = "compact")
gt(nfc) |>
  gt_theme_terminal(accent = "#7EE787")
```

Cookbook: [SportsDataverse table themes ↗](#)

PREVIEW & PALETTES

```
gt_theme_preview(nfc, themes = c(
  "gt_theme_sdv", "gt_theme_kenpom"))
gt_color_ranks(..., palette = pal_midnight)
```

`gt_theme_preview(data, themes = NULL, n = 5, ncol = 3, density = "compact", file = NULL, ...)`: `NULL` shows every theme; themes needs full function names. ... goes to `gt_grid()`.

`pal_midnight`: a rank palette that reads on midnight / terminal.



COLOUR BY VALUE

Function	Key arguments
gt_color_ranks()	fill cells: palette domain reverse, na_color = "white", autocolor_text = TRUE
gt_color_pills()	pill: fill_type = "continuous" / "rank", format_type, suffix, digits
gt_percentile_bar()	bar+marker: domain = c(0, 100), scale = "auto" (0-1 → 0-100), na_label = "-"
gt_color_results()	row colours: result_column = "result", result_type = "wl" / "binary"
gt_row_accent()	edge bar: column, palette, side = "left", hide = TRUE
gt_indicator_boxes()	boxes: indicator_vals = c(0, 1), color_yes, show_text
gt_tiers()	tier list: levels, colors, tier_column = "tier", style = "dark"

Most take columns and rows (an expression such as `wins > 10`, or indices). `palette` also takes a paletter name, e.g. "viridis:mako"; domain = `NULL` uses the data's range.

Cookbook: [Percentile bars and cut lines ↗](#)

TIER LIST

```
gt(seeds) |> # tier + logo URL columns
  gt_tiers(levels = c("Bye", "Wild card"),
    colors = c("#1B7837", "#B2182B")) |>
  gt_legend_discrete()
```

Columns other than `tier_column` render as images (`image_columns` narrows them; `img_height` = "55px"), e.g. a column of `sdv_logo_url()` URLs. Levels may be one named vector, `c(Bye = "#1B7837")`.

Cookbook: [Tier lists ↗](#)

LEGENDS

Function	Key arguments
gt_legend_continuous()	bar: type = "continuous" "steps" "blocks", n_bins = 5, labels ("edges"), title, location = "bottom", width = 200
gt_legend_discrete()	swatches: key_info (label = colour), heading, location = "top", shape, direction
gt(nfc) > gt_color_ranks(wins) > gt_legend_continuous(type = "steps", title = "Wins")	

The continuous legend reads the columns, palette and domain the colouring step recorded; the discrete one reads the key `gt_tiers()` recorded, so `key_info` can be left out.

```
gt(afc_west) |>
  gt_legend_discrete(c("Playoffs" =
    "#CCE7F5", "Missed" = "#F5CCCC"))
```

Cookbook: [Headers, legends, captions ↗](#)

HIGHLIGHT & FOCUS

Function	Key arguments
gt_highlight_cells()	columns, condition (~ .x > 0.7 or a logical mask), fill = "#FFF3B0"
gt_highlight_na()	fill = "#F0F0F0", missing_text, na_strings = "NA"
gt_outliers()	method = "iqr" "sd" "bounds", threshold (1.5/3), side, symbol, note
gt_spotlight()	rows; dims the rest (dim_color = "auto", WCAG AA); accent_color, if_none
gt_bold_rows()	rows, text_color = "black", highlight_color
gt_group_strips()	band row groups: color = "#F5F5F5", start = 2

```
gt(nfc) |>
  gt_spotlight(team == "DET",
    accent_color = "#0076B6") |>
  gt_outliers(wins, method = "sd",
    note = TRUE)
gt(nfc) |>
  gt_highlight_cells(c(wins, losses),
    ~ .x >= 10, bold = TRUE)
```



sdvplotR :: gt LAYOUT & reactable

The rest of the gtUtils toolkit for **gt**: number formats, rules and border bars, headers and captions, multi-table layouts and image export for posting. Then the **reactable** helpers: team logos, wordmarks and headshots in cells and headers, and team-coloured bars and fills. Examples continue with `nfc` (the NFC North rows of `sdv_example_standings`, page 3).

sdvplotR v0.1.0

webshot2 to save · reactable for tables

GALLERY ↗

COOKBOOKS ↗

GT · FORMAT, LAYOUT, SAVE

FORMAT

Function	Key arguments
<code>gt_delta()</code>	change column to - from: column_label = "Change", percent = FALSE, decimals = 1, arrows = FALSE, after
<code>gt_fmt_rank()</code>	1 → 1st: superscript = TRUE, suffix_size = "0.7em"
<code>gt_fmt_tally()</code>	W-L(-T) in one cell: separator = "-" , share = FALSE, share_of = 1, share_location = "inline" / "column"
<code>gt_significance()</code>	stars from p_columns: levels = c(0.01, 0.05, 0.1), symbols, hide_p = TRUE
<code>gt_scale_note()</code>	divide and say so: divisor = 1000 ("Figures in thousands."), where = "source_note" "label" "both"
<code>gt_wrap_labels()</code>	break long labels: width = 12, balance = TRUE
<code>gt_set_font()</code>	one family for the table: font_family, from_google_font = TRUE, weight, style
<code>gt(nfc) > gt_fmt_tally(c(wins, losses)) > gt_delta(from = last_season_wins, to = wins)</code>	

Cookbook: [Rolling-window wins](#) ↗

BORDERS & RULES

Function	Key arguments
<code>gt_border_grid()</code>	full grid: color = "black", weight = 1, include_labels = FALSE
<code>gt_border_bars_top() / _bottom()</code>	colour bars: colors, bar_height = 10, img, text, text_size = 18
<code>gt_cutline()</code>	line after rows: after, label, color = "#A6081A", style = "dashed", gap
<code>gt(nfc) > gt_border_bars_top(c("#0076B6", "#B0B7BC")) > gt_cutline(after = c(1, 3), label = c("Champion", NA))</code>	

Cookbook: [Border bars](#) ↗

HEADERS & ANNOTATIONS

Function	Key arguments
<code>gt_title_header()</code>	title, subtitle, kicker, date, one *_style list each
<code>gt_column_subheaders()</code>	per column list(heading, subtitle) in ...; subtitle_color = "#808080"
<code>gt_538_caption()</code>	top_caption over a rule, bottom_caption, align = "right"
<code>gt_marginalia()</code>	a column as notes: width = 220, italic = TRUE, rule = TRUE
<code>gt_social_tag()</code>	handles + icons: accounts, caption, stack = FALSE, separator = " "
<code>gt_watermark()</code>	text or image, opacity = 0.06, size = "60%", angle
<code>gt(nfc) > gt_title_header("NFC North", kicker = "2025 SEASON", subtitle = "Final standings") > gt_column_subheaders(wins = list(heading = "W", subtitle = "2025")) > gt_social_tag(c(x = "@SportsDataverse", gh = "sportsdataverse"))</code>	

`accounts` names map to Font Awesome brand icons, with aliases (x / twitter, ig, bsky, gh). A Date in date prints as "July 21, 2026".

Cookbook: [Headers, legends, captions](#) ↗

SNAKE A LONG TABLE

<code>nba <- subset(sdv_example_standings, league == "nba", c(team, wins, losses)) gt(nba) > gt_snake(n_cols = 3)</code>

Thirty rows become three blocks of ten, each with its own labels. Body-cell styling applied before `gt_snake()` carries through; formats and content transforms (such as `gt_sdv_logos()`) go after it. `gt_snake_align()` reshapes a precomputed mask or colour frame to match.

LEARN MORE

GALLERY ↗ every chart and table in one place

Start: [Getting started](#) · **All:** [articles](#) · [reference](#)

Per league: [NFL](#) · [CFB](#) · [NBA](#) · [WNBA](#) · [MLB](#) · [NHL](#) · [MBB](#) · [WBB](#) · [Soccer](#)

Recipes: [NFL EPA](#) · [CFB table](#) · [WBB standings](#) · [head-to-head card](#)

Leaderboards: [dashboards with Quarto](#) · **Automation:** [social graphics](#)

LAYOUT

Function	Key arguments
<code>gt_grid()</code>	tables in a grid: tables, ncol = 2, labels, title, caption, gap = 24, align = "top", file
<code>gt_stack_tables()</code>	one above another: gap = 16, align = "center", title, file
<code>gt_snake()</code>	long table in blocks: n_cols = 2 or rows_per_col, gap = 20, fill = ""
<code>gt_snake_align()</code>	reshape a parallel frame (a mask for gt_highlight_cells()) to match
<code>gt_grid(list(afc_tbl, nfc_tbl), ncol = 2, title = "2025 standings", file = "standings.png")</code>	

Grids and stacks share `subtitle`, `source_note`, `caption_rule`, the *_style lists, and the save arguments `bg` = "white", `whitespace` = 50, `zoom` = 2.

Cookbook: [Faceted tables](#) ↗ · [Schedule matrix](#) ↗

SAVE AS AN IMAGE

Function	Key arguments
<code>gt_save_crop()</code>	PNG/JPG, trimmed and padded: file = NULL, bg = "white", whitespace = 50, zoom = 2, expand = 5, width
<code>gt_social_crop()</code>	pad to aspect_ratio = "1:1", "16:9", "4:5" or a number; whitespace = 60, gravity = "center"
<code>gt_save_batch()</code>	one image per group: data, group, fn, file, dir, match_width = TRUE
<code>tbl > gt_save_crop("table.png", whitespace = 40, width = 1200)</code> <code># one image per conference nfl <- subset(sdv_example_standings, league == "nfl") build <- function(df, group) gt(df) > tab_header(title = group) gt_save_batch(nfl, conference, build, "nfl-{group}.png", dir = "out")</code>	

Saving runs headless Chrome through `webshot2` (Chrome gets 60 s to start). `file` = NULL returns the PNG bytes. `dir` has no default, so a batch never lands in the working directory unasked.

Cookbook: [Saving and posting](#) ↗ · [Social posting](#) ↗

REACTABLE

REACTABLE TABLES

Helper	Arguments (defaults)
<code>reactable_sdv_logos()</code>	league, variant = "default" ("dark", a named variant), height = 30, default_img = NULL
<code>reactable_sdv_wordmarks()</code>	the same
<code>reactable_sdv_headshots()</code>	league, height = 40, default_img, id_system = "espn"
<code>reactable_sdv_team_color_bar()</code>	data, team_col, league, which = "primary", max_value = NULL, na_color = "grey70"
<code>reactable_sdv_team_color_bg()</code>	as the bar, with alpha = 0.15 in place of max_value
<code>reactable_sdv_cols_label()</code>	.data, ... (to every colDef()), league, variant, height = 30

```
library(reactable)
reactable(df, columns = list(
  team = colDef(html = TRUE,
    cell = reactable_sdv_logos("nfl")),
  wins = colDef(style =
    reactable_sdv_team_color_bar(
      df, "team", league = "nfl"))))
# columns named by team key -> logo headers
reactable(h2h,
  columns = reactable_sdv_cols_label(
    h2h, league = "nba"))
```

All but headshots also take `season`, `id_system`, `strict`. The first three are `cell` renderers (`colDef(html = TRUE)`); unresolved values show as text, or `default_img`; a variant that fails to load swaps to the default image. The colour helpers are `style` functions: the bar's length is the value over `max_value` (the column maximum by default). `reactable_sdv_cols_label()` leaves out columns it cannot resolve, so they keep their plain names.

Cookbook: [reactable integration](#) ↗

SIBLINGS

Python **sdvplot** (PyPI) and JavaScript **@sportsdataverse/sdvplot** + **sdvtables** (npm) share `sdvplotR`'s team index, logo archive and argument names; images render through [ggpath](#).



sdvplotR :: CHEAT SHEET

Team logos, wordmarks, player headshots and team colours for **28 leagues** (pro, college, minor and junior hockey, spring football, soccer, cricket) in **ggplot2** plots, **gt** and **reactable** tables, and **plotly**, **ggraph** and **leaflet**, behind one `league` argument. Built on **ggpath**, after the approach of **nflplotR**, **cfbplotR**, **nbaplotR** and **mlbplotR**. Page 1: ggplot2 marks, scales and axes. Page 2: teams, interactive, surfaces, coordinates. Page 3: gt images, themes, colour. Page 4: gt layout, saving, reactable.

sdvplotR v0.1.0

R ≥ 4.1 · ggplot2 ≥ 4.0 · ggpath ≥ 1.1

[GALLERY ↗](#) [COOKBOOKS ↗](#)

GGPLOT2 · MARKS & SCALES

INSTALL & LOAD

```
install.packages("sdvplotR")
# dev: sportsdataverse.r-universe.dev
pak::pak("sportsdataverse/sdvplotR")
library(sdvplotR); library(ggplot2)
```

Every team helper takes `league`, a key from `sdv_leagues()` (`"nfl"` `"nba"` `"cfb"` `"nhl"` `"ahl"` `"soccer"` ..., page 2). Teams: any abbreviation, name or id `sdv_resolve()` reads (ESPN, nflverse, nba_api, MLB Stats, CFBD, ...), or a conference (`"SEC"` , `"BIG"` , `"AFC"`).

Cookbook: [Getting started ↗](#)

LOGOS, WORDMARKS & HEADSHOTS

Geom	Required aes
<code>geom_sdv_logos()</code>	<code>x</code> <code>y</code> team
<code>geom_sdv_wordmarks()</code>	<code>x</code> <code>y</code> team (NFL, MLB, MiLB)
<code>geom_sdv_headshots()</code>	<code>x</code> <code>y</code> player_id
<code>geom_from_path()</code>	<code>x</code> <code>y</code> path (any image)

Optional aes and defaults: `alpha` , `colour` (`"b/w"` greyscales), `angle` = 0, `hjust` = 0.5, `vjust` = 0.5, `width` = 1, `height` = 1 (npc; a typical logo is `width` = 0.075). Logos and wordmarks also take `season` (an aes or a constant).

```
df <- data.frame(
  team = sdv_teams("nba")$abbr,
  a = rnorm(30), b = rnorm(30))
ggplot(df, aes(a, b)) +
  geom_mean_lines(aes(x0 = a, y0 = b)) +
  geom_sdv_logos(aes(team = team),
    league = "nba", width = 0.07)
```

HIGHLIGHT A GROUP

```
nfl <- subset(sdv_example_standings,
  league == "nfl")
west <- c("DEN", "KC", "LAC", "LV")
ggplot(nfl, aes(points_for,
  points_against)) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, !team %in% west),
    league = "nfl", width = 0.05,
    alpha = 0.25) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, team %in% west),
    league = "nfl", width = 0.1)
```

Rows draw in order, so the last layer and row end on top. `colour` = `"b/w"` greys logos instead of fading them, which reads better on a dark background.

Cookbook: [Logos, wordmarks, headshots ↗](#)

GEOM ARGUMENTS

```
geom_sdv_logos(mapping = NULL,
  data = NULL, stat = "identity",
  position = "identity", ..., league,
  id_system = "auto", strict = FALSE,
  na.rm = FALSE, show.legend = FALSE,
  inherit.aes = TRUE)
geom_sdv_headshots(..., league,
  id_system = "espn")
```

`geom_sdv_wordmarks()` takes the logo arguments; headshots take no `season` or `strict`. Headshot `id_system`: `"espn"` = ESPN athlete ids in every league; `"gsis"` = nflverse NFL ids; `"league"` = NBA / WNBA Stats PERSON_ID, MLBAM, NHL API.

The ggproto objects `GeomSDVLogo`, `GeomSDVwordmark`, `GeomSDVheadshot` and `ggpath`'s `GeomFromPath`, `GeomReflines` are exported for extensions.

ERA LOGOS

```
old <- data.frame(x = 1:4, y = 1,
  team = c("QUE", "HFD", "ATL", "WPG"),
  season = c(1990, 1995, 2005, 2015))
ggplot(old, aes(x, y)) +
  geom_sdv_logos(aes(team = team,
    season = season), league = "nhl",
  width = 0.2) + theme_void()
```

`season`: the mark worn that year (ending year for split seasons); it also picks who held a reused code. NHL: every identity since 1917-18 (`QUE` `HFD`); AHL, ECHL, OHL, QMJHL, WHL, USHL, PWHL, PHF, UFL, XFL, AAF: dated marks; NFL / WNBA: relocated identities (`STL` `SD` `OAK`; `SA` `TUL`). Elsewhere today's mark. NHL marks are SVG: needs `rsvg`.

REFERENCE LINES

```
p + geom_mean_lines(aes(x0 = a, y0 = b))
p + geom_median_lines(aes(x0 = a),
  colour = "grey40", linetype = 1)
```

From `ggpath`. Optional aes `x0`, `y0`; defaults `colour` = `"red"`, `linewidth` = 0.5, `linetype` = 2 (dashed), `alpha` = NA.

ANY IMAGE: GEOM_FROM_PATH

```
df$url <- sdv_logo_url(df$team, "nba",
  variant = "dark")
ggplot(df, aes(a, b)) +
  geom_from_path(aes(path = url),
    width = 0.07)
```

`path` is a URL or a local file; same optional aes as the team geoms. Conferences resolve by key or other name: `sdv_logo_url(c("SEC", "BIG"), "cfb")`; a team's code `wins` (`cfb` "MAC" is Macalester).

COLOUR & FILL SCALES

```
scale_color_sdv(league, which = "primary",
  values = NULL, ..., aesthetics = "colour",
  breaks = waiver(), na.value = "grey50",
  guide = NULL, alpha = NA)
scale_fill_sdv(...) # aesthetics = "fill"
```

Maps team keys to official colours; which = `"secondary"` for the second colour, `alpha` fades them (`scales::alpha()`), `values` replaces them, ... goes to `scale_colour_manual()`. `scale_colour_sdv()` is an alias.

```
ggplot(df, aes(team, b)) +
  geom_col(aes(colour = team, fill = team)) +
  scale_color_sdv("nba", "secondary") +
  scale_fill_sdv("nba", alpha = 0.4)
```



NFL primary / secondary, as `sdv_team_colors("nfl")`, which = `"all"` returns them.

Cookbook: [Team colours and themes ↗](#)

TITLES

```
p + ggtitle_image(title_image = "SEA",
  title = "Seattle Seahawks Analysis",
  image_height = 20, image_side = "left",
  league = "nfl") +
  theme_title_image(size = 20, hjust = 0.5)
```

`title_image`: a team key (logo resolved) or any image URL. `image_height` = 15 px by default; `image_side` = `"left"` or `"right"`; subtitle. `theme_title_image(...)` is required and passes ... to `ggtext::element_markdown()`.

LOGO AT EACH BAR'S END

```
nba <- subset(sdv_example_standings,
  league == "nba")
ggplot(nba, aes(wins,
  reorder(team, wins))) +
  geom_col(aes(fill = team)) +
  geom_sdv_logos(aes(x = wins + 3,
    team = team), league = "nba",
  height = 0.028) +
  scale_fill_sdv("nba")
```

`height` sizes the logo here: each row is about a thirtieth of the panel.

LOGO AXES

Route	Calls
theme	<code>element_sdv_logo()</code> · <code>element_sdv</code>
element	<code>_wordmark()</code> · <code>element_sdv_headshot()</code>
scale+ theme	<code>scale_x_sdv()</code> / <code>scale_y_sdv()</code> · <code>scale_x_sdv_headshots()</code> + <code>theme_x_sdv()</code> / <code>theme_y_sdv()</code>

```
# element route (size in cm)
ggplot(df, aes(team, b)) + geom_col() +
  theme(axis.text.x =
    element_sdv_logo("nba", size = 1))
```

`element_sdv_logo(league, alpha = NULL, colour = NA, hjust = NULL, vjust = NULL, size = 0.5, season, id_system)`; `_wordmark()` the same; `_headshot()` takes `id_system`, no `season`.

```
# scale route (size in px; needs ggtext)
ggplot(df, aes(b, team)) + geom_col() +
  scale_y_sdv(league = "nba", size = 12) +
  theme_y_sdv()
```

`scale_x_sdv(..., league, expand, guide, position = "bottom", size = 12, season, id_system)`; `scale_y_sdv()` has position = `"left"`. Size is the height on x, the width on y. `scale_x_sdv_headshots()` (size = 20) and `scale_y_sdv_headshots()` (size = 30) take `id_system`, `theme_x_sdv()` / `theme_y_sdv()` render those labels as markdown.

Cookbook: [Logo axes and titles ↗](#)

HEADSHOTS ON A SCATTER

```
ggplot(scorers, aes(ppg, ts)) +
  geom_sdv_headshots(aes(player_id =
  athlete_id), league = "nba",
  height = 0.12)
```

ESPN ids: the `athlete_id` of `hoopR`, `whoop`, `cfbfastR` rosters. NFL `gsis` ids (`id_system` = `"gsis"`) go through a headshot map built from nflverse rosters (1999 on); `sdv_clear_cache()` re-reads it.

IMAGES IN THEME ELEMENTS

```
p + theme(plot.background =
  element_sdv_raster("court.png"))
```

`element_sdv_raster(image_path, x, y, width, height, just = "centre", hjust = 0.5, vjust = 0.5, interpolate = TRUE, ...)`: one image in a rectangle; wraps `ggpath`'s `element_raster()` (also exported).

`element_path(alpha = 1, colour = NA, hjust, vjust, angle = 0, size = unit(0.5, "cm"))`: axis text that already holds image paths or URLs.



sdvplotR :: TEAMS, INTERACTIVE & SURFACES

The shared SportsDataverse team index behind every helper: **28 leagues**, any team identifier resolved to one canonical `team_id`, colours, palettes and archived logo URLs. Plus logos on **plotly**, **ggiraph** and **leaflet**, premade tier plots, team-coloured playing surfaces drawn with **sportyR**, and converters that put shot and event coordinates from stats.nba.com, Euroleague and ten soccer providers on one frame.

sdvplotR v0.1.0

sportyR for surfaces · rsvg for SVG logos

[GALLERY ↗](#) [COOKBOOKS ↗](#)

TEAM DATA · LEAGUES, KEYS, COLOURS

LEAGUES & TEAMS

```
sdv_leagues() # the 28 league keys
sdv_teams("nfl")
sdv_teams("cfb", include_conferences = TRUE)

Group      sdv_teams() columns
ids,        league team_id abbr name short_name
names      location

grouping    program conference_id conference
colours     color_primary color_secondary
            color_source

team_id is the canonical key (the ESPN id where ESPN has one).
Conference rows ("SEC", "AFC") draw logos only.
```

RESOLVE ANY IDENTIFIER

```
sdv_resolve(c("GSW", "Golden State Warriors",
              "XYZ"), "nba")
#> "9" "9" NA + one warning
sdv_resolve(c("OAK", "SD", "STL"), "nfl")
#> "13" "24" "14" LV, LAC, LA
sdv_suggest("Kansas Cty Chiefs", "nfl")

sdv_resolve(x, league, season = NULL, id_system =
"auto", strict = FALSE) reads abbreviations, names
and ids from ESPN, nflverse, MLB Stats, nba_api, NHL,
HockeyTech, NCAA, PFF, CFBD, Sports Reference, FanGraphs
and more. id_system names one ("nhl_id" only
explicitly). season picks who held a reused code (ending
year for split seasons). Unresolved: NA and one warning
(strict = TRUE errors). sdv_suggest(x, league, n =
5) lists candidates, never picks.

Every helper that takes a team resolves it this way, with
league and id_system. strict: the logo-URL, colour,
tier and mark-drawing helpers; not sdv_marks(),
element_sdv_*(), scale_x/y_sdv(), sdv_colorway(),
sdv_team_factor().
```

TEAM FACTOR

```
sdv_team_factor(c("KC", "BUF", "invalid"),
               league = "nfl")
#> KC BUF <NA> Levels: BUF KC

Levels ordered by abbreviation; unresolved values become
NA, so scales drop them.
```

COLOURS & PALETTES

```
sdv_team_colors("nfl", "KC")
sdv_team_colors("nfl", c("KC", "BUF"),
               which = "all")
sdv_team_colors("nba", which = "all")
sdv_color_palette("nba",
                 teams = c("BOS", "NY"))

sdv_team_colors(league, team = NULL, which =
"primary", season, id_system, strict,
include_conferences = FALSE): a named hex vector (NA
where a team does not resolve); which = "all" gives
"primary, secondary" strings, or with team = NULL a
data frame team_id abbr name primary secondary.
sdv_color_palette(league, teams = NULL, which =
"primary", ...): named colours for scale_*_manual().
```

IMAGE URLS & MARKS

```
sdv_logo_url(c("KC", "Buffalo Bills",
               "OAK"), "nfl")
sdv_logo_url("NYJ", "nfl",
             variant = "scoreboard")
sdv_logo_url("QUE", "nhl", season = 1990)
sdv_logo_url("KC", "nfl",
             mark_type = "wordmark")
sdv_marks("KC", "nfl") # every mark
sdv_headshot_url("2544", "nba",
                 id_system = "league")

sdv_logo_url(team, league, season = NULL,
             variant = "default", mark_type = "logo",
             id_system, strict): the SportsDataverse archive's
immutable copy. variant: "default", "dark", or a
name from sdv_marks() ("scoreboard", NFL
"grayscale", MLB "cap_on_light", ...); a team without
it gets its closest mark of the same polarity, an unknown
name errors. mark_type = "wordmark": NFL, MLB (and
MiLB).

sdv_headshot_url(player_id, league, id_system =
"espn"): ESPN headshots for nfl nba wnba mlb nhl cfb mbb
wbb; "gsis" and "league" as on page 1. NA for a missing
or malformed id.
```

CACHE & VERSIONS

Index and logo manifest come from the SportsDataverse
CDN, cached for a day; a copy ships in the package
(options(sdvplotR.offline = TRUE)).
sdv_versions(): what is loaded, from where.
sdv_clear_cache(): drop index, manifest, NFL headshot
map and ggpath images.

COVERAGE

Leagues	Teams
nfl nba mlb nhl wnba	32 · 30 · 30 · 32 · 15
nbagl mlb pwhl phf	32 · 286 · 12 · 8
ufl xfl usfl aaf	11 · 11 · 9 · 8
ahl echl ohl whl qmjhl ushl	61 · 59 · 27 · 23 · 30 · 17
cfb mbb wbb	687 · 366 · 364
ncaa_baseball softball mhockey whockey	414 · 415 · 110 · 48
soccer · cricket	2,631 · 111

Team rows in the bundled index 0aeef853ec2b; the live
index can differ.

PREMADE PLOT: TEAM TIERS

```
tiers <- data.frame(
  tier_no = c(1, 1, 2, 2, 3),
  team = c("KC", "DET", "BAL", "PHI", "BUF"))
sdv_team_tiers(tiers, league = "nfl",
               theme = "light", no_line_below_tier = 2)

sdv_team_tiers(data, league, title, subtitle,
               caption = NULL, tier_desc, presort = FALSE,
               alpha = 0.8, width = 0.075, no_line_below_tier =
NULL, devel = FALSE, theme = "dark", variant =
"auto", season, id_system, strict). data needs
tier_no and team (optional tier_rank). Title "NFL
Team Tiers" from league; NULL drops one. devel =
TRUE plots keys, not logos; "auto": dark marks on the dark
theme.
```

Cookbook: [College hoops tiers ↗](#)

INTERACTIVE: GGIRAPH, PLOTLY, LEAFLET

```
p <- ggplot(d, aes(x, y)) +
  geom_sdv_logos_interactive(aes(team = team,
                                tooltip = team, data_id = team),
                           league = "nfl", width = 0.1)
ggiraph::girafe(ggobj = p)
plotly::plot_ly(d, x = ~x, y = ~y) |>
  add_sdv_logos(d$x, d$y, d$team, "nfl") |>
  sdv_colorway("nfl", d$team)
leaflet::leaflet(d) |> leaflet::addTiles() |>
  addSdvLogoMarkers(~lon, ~lat, ~team,
                   league = "nfl")

geom_sdv_{logos,wordmarks,headshots}
_interactive(): images plus an invisible hover point
(hover_size = 8). plotly: add_sdv_logos(p, x, y,
team, league, size = 0.08, ...),
add_sdv_wordmarks(), add_sdv_headshots(),
sdv_axis_logos(p, teams, league, axis) tick logos,
sdv_colorway(); ggplotly() drops them.
addSdvLogoMarkers(..., size = 32): no logo keeps the
default marker.
```

Cookbook: [Interactive plots ↗](#)

PLAYING SURFACES

```
sdv_surface("nba", "BOS",
            center_logo = TRUE) +
  geom_point(aes(x, y), data = shots)

league      sportyR surface
nfl aaf ufl usfl xfl cfb  geom_football("nfl" /
                                "ncaa")

nba nbagl wnba mbb wbb fiba  geom_basketball(); fiba 28
                                × 15 m

mlb mlb ncaa_baseball  geom_baseball()

nhl ahl echl ohl qmjhl ushl pwhl  geom_hockey(); whl on the
phf ncaa_mhockey        NHL rink
ncaa_whockey

soccer  geom_soccer("fifa"), 105
                                × 68 m

sdv_surface(league, team = NULL, season = NULL,
             center_logo = FALSE, ...) (not cricket, ncaa_softball).
A team paints the paint and apron, end zones, or centre line,
circle and boards; baseball stays regulation, soccer gets a
centre logo only. Stylized, not real floors. Origin at centre, in
feet (yards for football, metres for soccer and FIBA); ...
goes to sportyR (rotation, xlims, color_updates).
```

Cookbook: [Surfaces and shot charts ↗](#)

SHOT & EVENT COORDINATES

```
# stats.nba.com legacy frame -> feet
shots |> sdv_court_coords()
# Euroleague (cm, hoop origin) -> FIBA m
pts |> sdv_court_coords(
  x_column = "coord_x", y_column = "coord_y",
  provider = "euroleague")
# soccer events -> 105 x 68 m pitch
ev |> sdv_pitch_coords("statsbomb")

sdv_court_coords(data, x_column = "x_legacy",
                 y_column = "y_legacy", provider = "nba") adds
court_x court_y: basket at -41.75 ft (-12.425 m for
Euroleague), half court at 0; court_y -25 to 25 ft, negative =
shooter's left. From hoopR's nba_shotchartdetail():
sdv_court_coords(res$Shot_Chart_Detail, "LOC_X",
                 "LOC_Y").

sdv_pitch_coords(data, provider, x_column =
NULL, y_column = NULL, flip = NULL, pitch_length
= NULL, pitch_width = NULL) adds pitch_x pitch_y
in metres from the centre spot (pitch_x -52.5 = defended
goal line). flip: TRUE / FALSE or a logical column, as flip
= "away", turns rows half a turn. pitch_length /
pitch_width: tracking only. Providers: opta
(statsperform) wyscout statsbomb uefa impect
espn, tracking tracab skillcorner secondspectrum
metrica.
```

Cookbook: [NHL shot map ↗](#) · [Soccer ↗](#)



sdvplotR :: gt TABLES

Team images and colours for **gt**, plus the **gtUtils** toolkit (by Andrew Weatherman): 20 themes, legends, colouring, annotations, layouts and image export. Every helper takes a `gt_tbl` as its first argument (`gt_object`) and returns one, so they chain with `|>`. Examples run on `sdv_example_standings` (real NFL and NBA final standings). Page 3: images, themes, colour, legends, focus. Page 4: format, layout, saving, reactable.

sdvplotR v0.1.0

gt ≥ 0.10 · paletteer optional

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GT · IMAGES, THEMES, COLOUR

TEAM IMAGES IN CELLS

```
gt_sdv_logos(gt_object, columns, league,
  height = 30, locations = NULL,
  include_name = FALSE, season = NULL,
  id_system = "auto", strict = FALSE)
gt_sdv_wordmarks(gt_object, columns,
  league, height = 30, locations = NULL,
  season = NULL, id_system, strict)
gt_sdv_headshots(gt_object, columns,
  league, height = 30, locations = NULL,
  id_system = "espn")
gt_sdv_cols_label(gt_object,
  columns = everything(), league,
  height = 30, mark_type = "logo",
  id_system, season, strict)
```

`height` in px. `locations` (`cells_body()`, `cells_stub()`, `cells_column_labels()`, `cells_row_groups()`) replaces `columns.include_name = TRUE` keeps the text after the logo; `season` shows one season's marks in every cell. Wordmarks: NFL, MLB (and MiLB). `gt_sdv_cols_label()` turns column *labels* into images: `mark_type = "logo"` "wordmark" "headshot".

STACKED NAME IN TEAM COLOUR

```
gt_merge_stack_team_color(gt_object,
  col1, col2, team_color, league,
  font_size_top = 14,
  font_size_bottom = 12, color = "black",
  background = NULL, season, id_system)
```

`col1` on top in bold caps, `col2` below in the team's colour; a primary under 4.5:1 contrast gives way to the secondary or is darkened. Apply a theme first: `background = NULL` reads it.

```
x <- subset(sdv_example_standings,
  league == "nfl",
  c(team, team_name, division))
gt(x) |>
  gt_merge_stack_team_color(team_name,
    division, team, league = "nfl")
```

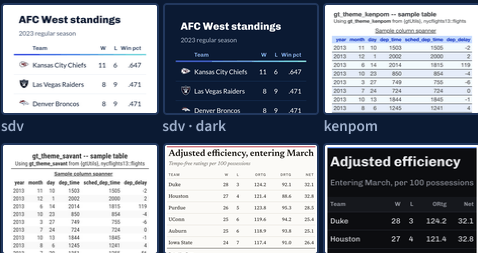
EXAMPLE

```
library(gt)
nfc <- subset(sdv_example_standings,
  division == "NFC North",
  c(team, team_name, wins, losses,
    last_season_wins))
gt(nfc) |>
  gt_sdv_logos(columns = team,
    league = "nfl", height = 28) |>
  gt_theme_sdv()
```

Cookbook: [Team tables with the gt toolkit ↗](#)

THEMES

```
gt_theme_sdv(gt, style = "Light")
gt_theme_sdv_team(gt, team = "KC",
  league = "nfl")
```



savant broadsheet midnight

`gt_theme_sdv()` : Chivo titles and labels, a Lato body in tabular figures, and the SDV gradient (#3346F0 → #7FE6DC) as one horizon line under the labels; `style = "dark"` sets it on the SportsDataverse navy.

`gt_theme_sdv_team()` : the sdv table in one team's colours (title block primary, horizon line secondary); `team = NULL` keeps SportsDataverse colours; takes `season`, `id_system`.

Every `gt_theme<name>(gt_object, ..., density)` takes `density = "comfortable"` "compact" "social" (default comfortable; compact for almanac, scoreboard, terminal) and passes ... to `gt::tab_options()`, applied last.

Option	Themes (default)
accent	almanac "#8C2F1E" · booktabs, swiss, tuftes "#111111" · broadsheet "#A6081A" · brutalist "#FF3B00" · midnight "#5B8DEF" · scoreboard "#0E1621" · terminal "#FFB86C"
style	sdv, sofa "light" / "dark" · tier "dark" / "light"
other	drench color = "#123F5E" · broadsheet paper = "white" or "salmon" · almanac stripe (NA = off)
none	athletic gtutils kenpom ncaa pl savant

All 20: almanac athletic booktabs broadsheet brutalist drench gtutils kenpom midnight ncaa pl savant scoreboard sdv sdv_team sofa swiss terminal tier tuftes (each `gt_theme<name>()`).

```
gt(nfc) |>
  gt_theme_broadsheet(paper = "salmon",
    density = "compact")
gt(nfc) |>
  gt_theme_terminal(accent = "#7EE787")
```

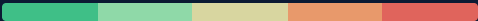
Cookbook: [SportsDataverse table themes ↗](#)

PREVIEW & PALETTES

```
gt_theme_preview(nfc, themes = c(
  "gt_theme_sdv", "gt_theme_kenpom"))
gt_color_ranks(..., palette = pal_midnight)
```

`gt_theme_preview(data, themes = NULL, n = 5, ncol = 3, density = "compact", file = NULL, ...)`: `NULL` shows every theme; themes needs full function names. ... goes to `gt_grid()`.

`pal_midnight`: a rank palette that reads on midnight / terminal.



COLOUR BY VALUE

Function	Key arguments
gt_color_ranks()	fill cells: palette domain reverse, na_color = "white", autocolor_text = TRUE
gt_color_pills()	pill: fill_type = "continuous" / "rank", format_type, suffix, digits
gt_percentile_bar()	bar+marker: domain = c(0, 100), scale = "auto" (0-1 → 0-100), na_label = "-"
gt_color_results()	row colours: result_column = "result", result_type = "wl" / "binary"
gt_row_accent()	edge bar: column, palette, side = "left", hide = TRUE
gt_indicator_boxes()	boxes: indicator_vals = c(0, 1), color_yes, show_text
gt_tiers()	tierlist: levels, colors, tier_column = "tier", style = "dark"

Most take `columns` and `rows` (an expression such as `wins > 10`, or indices). `palette` also takes a paletteer name, e.g. "viridis:mako"; `domain = NULL` uses the data's range.

Cookbook: [Percentile bars and cut lines ↗](#)

TIER LIST

```
gt(seeds) |> # tier + logo URL columns
  gt_tiers(levels = c("Bye", "Wild card"),
    colors = c("#1B7837", "#B2182B")) |>
  gt_legend_discrete()
```

Columns other than `tier_column` render as images (`image_columns` narrows them; `img_height = "55px"`), e.g. a column of `sdv_logo_url()` URLs. Levels may be one named vector, `c(Bye = "#1B7837")`.

Cookbook: [Tier lists ↗](#)

LEGENDS

Function	Key arguments
gt_legend_continuous()	bar: type = "continuous" "steps" "blocks", n_bins = 5, labels ("edges"), title, location = "bottom", width = 200
gt_legend_discrete()	swatches: key_info (label = colour), heading, location = "top", shape, direction
gt(nfc) > gt_color_ranks(wins) > gt_legend_continuous(type = "steps", title = "Wins")	

The continuous legend reads the columns, palette and domain the colouring step recorded; the discrete one reads the key `gt_tiers()` recorded, so `key_info` can be left out.

```
gt(afc_west) |>
  gt_legend_discrete(c("Playoffs" =
    "#CCE7F5", "Missed" = "#F5CCCC"))
```

Cookbook: [Headers, legends, captions ↗](#)

HIGHLIGHT & FOCUS

Function	Key arguments
gt_highlight_cells()	columns, condition (~ .x > 0.7 or a logical mask), fill = "#FFF3B0"
gt_highlight_na()	fill = "#F0F0F0", missing_text, na_strings = "NA"
gt_outliers()	method = "iqr" "sd" "bounds", threshold (1.5/3), side, symbol, note
gt_spotlight()	rows; dims the rest (dim_color = "auto", WCAG AA); accent_color, if_none
gt_bold_rows()	rows, text_color = "black", highlight_color
gt_group_strips()	band row groups: color = "#F5F5F5", start = 2

```
gt(nfc) |>
  gt_spotlight(team == "DET",
    accent_color = "#0076B6") |>
  gt_outliers(wins, method = "sd",
    note = TRUE)
gt(nfc) |>
  gt_highlight_cells(c(wins, losses),
    ~ .x >= 10, bold = TRUE)
```



sdvplotR :: gt LAYOUT & reactable

The rest of the gtUtils toolkit for **gt**: number formats, rules and border bars, headers and captions, multi-table layouts and image export for posting. Then the **reactable** helpers: team logos, wordmarks and headshots in cells and headers, and team-coloured bars and fills. Examples continue with `nfc` (the NFC North rows of `sdv_example_standings`, page 3).

sdvplotR v0.1.0

webshot2 to save · reactable for tables

GALLERY ↗

COOKBOOKS ↗

GT · FORMAT, LAYOUT, SAVE

FORMAT

Function	Key arguments
<code>gt_delta()</code>	change column to - from: <code>column_label = "Change"</code> , <code>percent = FALSE</code> , <code>decimals = 1</code> , <code>arrows = FALSE</code> , <code>after</code>
<code>gt_fmt_rank()</code>	1→1st: <code>superscript = TRUE</code> , <code>suffix_size = "0.7em"</code>
<code>gt_fmt_tally()</code>	W-L(-T) in one cell: <code>separator = "-"</code> , <code>share = FALSE</code> , <code>share_of = 1</code> , <code>share_location = "inline"</code> / <code>"column"</code>
<code>gt_significance()</code>	stars from <code>p_columns</code> : <code>levels = c(0.01, 0.05, 0.1)</code> , <code>symbols</code> , <code>hide_p = TRUE</code>
<code>gt_scale_note()</code>	divide and say so: <code>divisor = 1000</code> ("Figures in thousands."), <code>where = "source_note"</code> "label" "both"
<code>gt_wrap_labels()</code>	break long labels: <code>width = 12</code> , <code>balance = TRUE</code>
<code>gt_set_font()</code>	one family for the table: <code>font_family</code> , <code>from_google_font = TRUE</code> , <code>weight</code> , <code>style</code>
<code>gt(nfc) ></code> <code>gt_fmt_tally(c(wins, losses)) ></code> <code>gt_delta(from = last_season_wins,</code> <code>to = wins)</code>	

Cookbook: [Rolling-window wins ↗](#)

BORDERS & RULES

Function	Key arguments
<code>gt_border_grid()</code>	full grid: <code>color = "black"</code> , <code>weight = 1</code> , <code>include_labels = FALSE</code>
<code>gt_border_bars_top()</code> / <code>_bottom()</code>	colour bars: <code>colors</code> , <code>bar_height = 10</code> , <code>img</code> , <code>text</code> , <code>text_size = 18</code>
<code>gt_cutline()</code>	line after rows: <code>after</code> , <code>label</code> , <code>color = "#A6081A"</code> , <code>style = "dashed"</code> , <code>gap</code>
<code>gt(nfc) ></code> <code>gt_border_bars_top(c("#0076B6", "#B0B7BC")) ></code> <code>gt_cutline(after = c(1, 3),</code> <code>label = c("Champion", NA))</code>	

Cookbook: [Border bars ↗](#)

HEADERS & ANNOTATIONS

Function	Key arguments
<code>gt_title_header()</code>	<code>title</code> , <code>subtitle</code> , <code>kicker</code> , <code>date</code> , <code>one * _style</code> list each
<code>gt_column_subheaders()</code>	per column list(<code>heading</code> , <code>subtitle</code>) in ...; <code>subtitle_color = "#808080"</code>
<code>gt_538_caption()</code>	<code>top_caption</code> over a rule, <code>bottom_caption</code> , <code>align = "right"</code>
<code>gt_marginalia()</code>	a column as notes: <code>width = 220</code> , <code>italic = TRUE</code> , <code>rule = TRUE</code>
<code>gt_social_tag()</code>	handles + icons: <code>accounts</code> , <code>caption</code> , <code>stack = FALSE</code> , <code>separator = " "</code>
<code>gt_watermark()</code>	<code>text</code> or <code>image</code> , <code>opacity = 0.06</code> , <code>size = "60%"</code> , <code>angle</code>

```
gt(nfc) |>
  gt_title_header("NFC North",
    kicker = "2025 SEASON",
    subtitle = "Final standings") |>
  gt_column_subheaders(wins = list(
    heading = "W", subtitle = "2025")) |>
  gt_social_tag(c(x = "@SportsDataverse",
    gh = "sportsdataverse"))
```

`accounts` names map to Font Awesome brand icons, with `aliases` (x / twitter, ig, bsky, gh). A Date in date prints as "July 21, 2026".

Cookbook: [Headers, legends, captions ↗](#)

SNAKE A LONG TABLE

```
nba <- subset(sdv_example_standings,
  league == "nba", c(team, wins, losses))
gt(nba) |> gt_snake(n_cols = 3)
```

Thirty rows become three blocks of ten, each with its own labels. Body-cell styling applied before `gt_snake()` carries through; formats and content transforms (such as `gt_sdv_logos()`) go after it. `gt_snake_align()` reshapes a precomputed mask or colour frame to match.

LEARN MORE

GALLERY ↗ every chart and table in one place

Start: Getting started · **All:** articles · reference

Per league: NFL · CFB · NBA · WNBA · MLB · NHL · MBB · WBB · Soccer

Recipes: NFL EPA · CFB table · WBB standings · head-to-head card

Leaderboards: dashboards with Quarto · **Automation:** social graphics

LAYOUT

Function	Key arguments
<code>gt_grid()</code>	tables in a grid: <code>tables</code> , <code>ncol = 2</code> , <code>labels</code> , <code>title</code> , <code>caption</code> , <code>gap = 24</code> , <code>align = "top"</code> , <code>file</code>
<code>gt_stack_tables()</code>	one above another: <code>gap = 16</code> , <code>align = "center"</code> , <code>title</code> , <code>file</code>
<code>gt_snake()</code>	long table in blocks: <code>n_cols = 2</code> or <code>rows_per_col</code> , <code>gap = 20</code> , <code>fill = ""</code>
<code>gt_snake_align()</code>	reshape a parallel frame (a mask for <code>gt_highlight_cells()</code>) to match
<code>gt_grid(list(afc_tbl, nfc_tbl),</code> <code>ncol = 2, title = "2025 standings",</code> <code>file = "standings.png")</code>	

Grids and stacks share `subtitle`, `source_note`, `caption_rule`, the `*_style` lists, and the save arguments `bg = "white"`, `whitespace = 50`, `zoom = 2`.

Cookbook: [Faceted tables ↗](#) · [Schedule matrix ↗](#)

SAVE AS AN IMAGE

Function	Key arguments
<code>gt_save_crop()</code>	PNG/JPG, trimmed and padded: <code>file = NULL</code> , <code>bg = "white"</code> , <code>whitespace = 50</code> , <code>zoom = 2</code> , <code>expand = 5</code> , <code>width</code>
<code>gt_social_crop()</code>	pad to aspect_ratio = "1:1", "16:9", "4:5" or number; <code>whitespace = 60</code> , <code>gravity = "center"</code>
<code>gt_save_batch()</code>	one image per group: <code>data</code> , <code>group</code> , <code>fn</code> , <code>file</code> , <code>dir</code> , <code>match_width = TRUE</code>
<code>tbl > gt_save_crop("table.png",</code> <code>whitespace = 40,</code> <code>width = 1200)</code> <code># one image per conference</code> <code>nfl <- subset(sdv_example_standings,</code> <code>league == "nfl")</code> <code>build <- function(df, group)</code> <code> gt(df) > tab_header(title = group)</code> <code> gt_save_batch(nfl, conference, build,</code> <code> "nfl-{group}.png", dir = "out")</code>	

Saving runs headless Chrome through `webshot2` (Chrome gets 60 s to start). `file = NULL` returns the PNG bytes. `dir` has no default, so a batch never lands in the working directory unasked.

Cookbook: [Saving and posting ↗](#) · [Social posting ↗](#)

REACTABLE

REACTABLE TABLES

Helper	Arguments (defaults)
<code>reactable_sdv_logos()</code>	<code>league</code> , <code>variant = "default"</code> ("dark", a named variant), <code>height = 30</code> , <code>default_img = NULL</code>
<code>reactable_sdv_wordmarks()</code>	the same
<code>reactable_sdv_headshots()</code>	<code>league</code> , <code>height = 40</code> , <code>default_img</code> , <code>id_system = "espn"</code>
<code>reactable_sdv_team_color_bar()</code>	<code>data</code> , <code>team_col</code> , <code>league</code> , <code>which = "primary"</code> , <code>max_value = NULL</code> , <code>na_color = "grey70"</code>
<code>reactable_sdv_team_color_bg()</code>	as the bar, with <code>alpha = 0.15</code> in place of <code>max_value</code>
<code>reactable_sdv_cols_label()</code>	.data, ... (to every <code>colDef()</code>), <code>league</code> , <code>variant</code> , <code>height = 30</code>

```
library(reactable)
reactable(df, columns = list(
  team = colDef(html = TRUE,
    cell = reactable_sdv_logos("nfl")),
  wins = colDef(style =
    reactable_sdv_team_color_bar(
      df, "team", league = "nfl"))))
# columns named by team key -> logo headers
reactable(h2h,
  columns = reactable_sdv_cols_label(
    h2h, league = "nba"))
```

All but headshots also take `season`, `id_system`, `strict`. The first three are `cell` renderers (`colDef(html = TRUE)`); unresolved values show as text, or `default_img`; a variant that fails to load swaps to the default image. The colour helpers are `style` functions: the bar's length is the value over `max_value` (the column maximum by default). `reactable_sdv_cols_label()` leaves out columns it cannot resolve, so they keep their plain names.

Cookbook: [reactable integration ↗](#)

SIBLINGS

Python `sdvplot` (PyPI) and JavaScript `@sportsdataverse/sdvplot` + `sdvtables` (npm) share `sdvplotR`'s team index, logo archive and argument names; images render through `ggpauth`.



sdvplotR :: CHEAT SHEET

Team logos, wordmarks, player headshots and team colours for **28 leagues** (pro, college, minor and junior hockey, spring football, soccer, cricket) in **ggplot2** plots, **gt** and **reactable** tables, and **plotly**, **ggiraph** and **leaflet**, behind one `league` argument. Built on `ggpath`, after the approach of `nflplotR`, `cfbplotR`, `nbaplotR` and `mlbplotR`. Page 1: `ggplot2` marks, scales and axes. Page 2: teams, interactive, surfaces, coordinates. Page 3: `gt` images, themes, colour. Page 4: `gt` layout, saving, `reactable`.

sdvplotR v0.1.0

R ≥ 4.1 · ggplot2 ≥ 4.0 · ggpath ≥ 1.1

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GGPLOT2 · MARKS & SCALES

INSTALL & LOAD

```
install.packages("sdvplotR")
# dev: sportsdataverse.r-universe.dev
pak::pak("sportsdataverse/sdvplotR")
library(sdvplotR); library(ggplot2)
```

Every team helper takes `league`, a key from `sdv_leagues()` (`"nfl"` `"nba"` `"cfb"` `"nhl"` `"ahl"` `"soccer"` ...; page 2). Teams: any abbreviation, name or id `sdv_resolve()` reads (ESPN, nflverse, nba_api, MLB Stats, CFBD, ...), or a conference (`"SEC"`, `"B1G"`, `"AFC"`).

Cookbook: [Getting started ↗](#)

LOGOS, WORDMARKS & HEADSHOTS

Geom	Required aes
<code>geom_sdv_logos()</code>	<code>x</code> <code>y</code> <code>team</code>
<code>geom_sdv_wordmarks()</code>	<code>x</code> <code>y</code> <code>team</code> (NFL, MLB, MiLB)
<code>geom_sdv_headshots()</code>	<code>x</code> <code>y</code> <code>player_id</code>
<code>geom_from_path()</code>	<code>x</code> <code>y</code> <code>path</code> (any image)

Optional aes and defaults: `alpha`, `colour` (`"b/w"` greyscales), `angle` = 0, `hjust` = 0.5, `vjust` = 0.5, `width` = 1, `height` = 1 (npc; a typical logo is `width` = 0.075). Logos and wordmarks also take `season` (an aes or a constant).

```
df <- data.frame(
  team = sdv_teams("nba")$abbr,
  a = rnorm(30), b = rnorm(30))
ggplot(df, aes(a, b)) +
  geom_mean_lines(aes(x0 = a, y0 = b)) +
  geom_sdv_logos(aes(team = team),
    league = "nba", width = 0.07)
```

HIGHLIGHT A GROUP

```
nfl <- subset(sdv_example_standings,
  league == "nfl")
west <- c("DEN", "KC", "LAC", "LV")
ggplot(nfl, aes(points_for,
  points_against)) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, !team %in% west),
    league = "nfl", width = 0.05,
    alpha = 0.25) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, team %in% west),
    league = "nfl", width = 0.1)
```

Rows draw in order, so the last layer and row end on top. `colour` = `"b/w"` greys logos instead of fading them, which reads better on a dark background.

Cookbook: [Logos, wordmarks, headshots ↗](#)

GEOM ARGUMENTS

```
geom_sdv_logos(mapping = NULL,
  data = NULL, stat = "identity",
  position = "identity", ..., league,
  id_system = "auto", strict = FALSE,
  na.rm = FALSE, show.legend = FALSE,
  inherit.aes = TRUE)
geom_sdv_headshots(..., league,
  id_system = "espn")
```

`geom_sdv_wordmarks()` takes the logo arguments; headshots take no `season` or `strict`. Headshot `id_system`: `"espn"` = ESPN athlete ids in every league; `"gsis"` = nflverse NFL ids; `"league"` = NBA / WNBA Stats PERSON_ID, MLBAM, NHL API.

The `ggproto` objects `GeomSDVLogo`, `GeomSDVwordmark`, `GeomSDVheadshot` and `ggpath`'s `GeomFromPath`, `GeomRefLines` are exported for extensions.

ERA LOGOS

```
old <- data.frame(x = 1:4, y = 1,
  team = c("QUE", "HFD", "ATL", "WPG"),
  season = c(1990, 1995, 2005, 2015))
ggplot(old, aes(x, y)) +
  geom_sdv_logos(aes(team = team,
    season = season), league = "nhl",
  width = 0.2) + theme_void()
```

`season`: the mark worn that year (ending year for split seasons); it also picks who held a reused code. NHL: every identity since 1917-18 (`QUE` `HFD`); AHL, ECHL, OHL, QMJHL, WHL, USHL, PWHL, PHF, UFL, XFL, AAF: dated marks; NFL / WNBA: relocated identities (`STL` `SD` `OAK`; `SA` `TUL`). Elsewhere today's mark. NHL marks are SVG: needs `rsvg`.

REFERENCE LINES

```
p + geom_mean_lines(aes(x0 = a, y0 = b))
p + geom_median_lines(aes(x0 = a),
  colour = "grey40", linetype = 1)
```

From `ggpath`. Optional aes `x0`, `y0`; defaults `colour` = `"red"`, `linewidth` = 0.5, `linetype` = 2 (dashed), `alpha` = NA.

ANY IMAGE: GEOM_FROM_PATH

```
df$url <- sdv_logo_url(df$team, "nba",
  variant = "dark")
ggplot(df, aes(a, b)) +
  geom_from_path(aes(path = url),
    width = 0.07)
```

`path` is a URL or a local file; same optional aes as the team geoms. Conferences resolve by key or other name: `sdv_logo_url(c("SEC", "B1G", "cfb"))`; a team's code wins (`cfb` "MAC" is Macalester).

COLOUR & FILL SCALES

```
scale_color_sdv(league, which = "primary",
  values = NULL, ..., aesthetics = "colour",
  breaks = waiver(), na.value = "grey50",
  guide = NULL, alpha = NA)
scale_fill_sdv(...) # aesthetics = "fill"
```

Maps team keys to official colours; `which` = `"secondary"` for the second colour, `alpha` fades them (`scales::alpha()`), `values` replaces them, ... goes to `scale_colour_manual()`. `scale_colour_sdv()` is an alias.

```
ggplot(df, aes(team, b)) +
  geom_col(aes(colour = team, fill = team)) +
  scale_color_sdv("nba", "secondary") +
  scale_fill_sdv("nba", alpha = 0.4)
```



NFL primary / secondary, as `sdv_team_colors("nfl")`, `which` = `"all"` returns them.

Cookbook: [Team colours and themes ↗](#)

TITLES

```
p + ggtitle_image(title_image = "SEA",
  title = "Seattle Seahawks Analysis",
  image_height = 20, image_side = "left",
  league = "nfl") +
  theme_title_image(size = 20, hjust = 0.5)
```

`title_image`: a team key (logo resolved) or any image URL. `image_height` = 15 px by default; `image_side` = `"left"` or `"right"`; subtitle. `theme_title_image(...)` is required and passes ... to `ggtext::element_markdown()`.

LOGO AT EACH BAR'S END

```
nba <- subset(sdv_example_standings,
  league == "nba")
ggplot(nba, aes(wins,
  reorder(team, wins))) +
  geom_col(aes(fill = team)) +
  geom_sdv_logos(aes(x = wins + 3,
    team = team), league = "nba",
  height = 0.028) +
  scale_fill_sdv("nba")
```

`height` sizes the logo here: each row is about a thirtieth of the panel.

LOGO AXES

Route	Calls
<code>theme</code>	<code>element_sdv_logo()</code> · <code>element_sdv</code>
<code>element</code>	<code>_wordmark()</code> · <code>element_sdv_headshot()</code>
<code>scale+ theme</code>	<code>scale_x_sdv()</code> / <code>scale_y_sdv()</code> · <code>scale_x_sdv_headshots()</code> + <code>theme_x_sdv()</code> / <code>theme_y_sdv()</code>

```
# element route (size in cm)
ggplot(df, aes(team, b)) + geom_col() +
  theme(axis.text.x =
    element_sdv_logo("nba", size = 1))
```

`element_sdv_logo(league, alpha = NULL, colour = NA, hjust = NULL, vjust = NULL, size = 0.5, season, id_system)`; `_wordmark()` the same; `_headshot()` takes `id_system`, no `season`.

```
# scale route (size in px; needs ggtext)
ggplot(df, aes(b, team)) + geom_col() +
  scale_y_sdv(league = "nba", size = 12) +
  theme_y_sdv()
```

`scale_x_sdv(..., league, expand, guide, position = "bottom", size = 12, season, id_system)`; `scale_y_sdv()` has position = `"left"`. Size is the height on x, the width on y. `scale_x_sdv_headshots()` (`size` = 20) and `scale_y_sdv_headshots()` (`size` = 30) take `id_system`. `theme_x_sdv()` / `theme_y_sdv()` render those labels as markdown.

Cookbook: [Logo axes and titles ↗](#)

HEADSHOTS ON A SCATTER

```
ggplot(scorers, aes(ppg, ts)) +
  geom_sdv_headshots(aes(player_id =
    athlete_id), league = "nba",
  height = 0.12)
```

ESPN ids: the `athlete_id` of `hoopR`, `wehoop`, `cfbfastR` rosters. NFL `gsis` ids (`id_system` = `"gsis"`) go through a headshot map built from nflverse rosters (1999 on); `sdv_clear_cache()` re-reads it.

IMAGES IN THEME ELEMENTS

```
p + theme(plot.background =
  element_sdv_raster("court.png"))
```

`element_sdv_raster(image_path, x, y, width, height, just = "centre", hjust = 0.5, vjust = 0.5, interpolate = TRUE, ...)`: one image in a rectangle; wraps `ggpath`'s `element_raster()` (also exported).

`element_path(alpha = 1, colour = NA, hjust, vjust, angle = 0, size = unit(0.5, "cm"))`: axis text that already holds image paths or URLs.



sdvplotR :: TEAMS, INTERACTIVE & SURFACES

The shared SportsDataverse team index behind every helper: **28 leagues**, any team identifier resolved to one canonical `team_id`, colours, palettes and archived logo URLs. Plus logos on **plotly**, **ggiraph** and **leaflet**, premade tier plots, team-coloured playing surfaces drawn with **sportyR**, and converters that put shot and event coordinates from stats.nba.com, Euroleague and ten soccer providers on one frame.

sdvplotR v0.1.0

sportyR for surfaces · rsvg for SVG logos

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

TEAM DATA · LEAGUES, KEYS, COLOURS

LEAGUES & TEAMS

```
sdv_leagues() # the 28 league keys
sdv_teams("nfl")
sdv_teams("cfb", include_conferences = TRUE)
```

Group	sdv_teams()	columns
ids, names	league team_id abbr name short_name location	
grouping	program conference_id conference	
colours	color_primary color_secondary color_source	

`team_id` is the canonical key (the ESPN id where ESPN has one). Conference rows ("SEC", "AFC") draw logos only.

RESOLVE ANY IDENTIFIER

```
sdv_resolve(c("GSW", "Golden State Warriors",
"XYZ"), "nba")
#> "9" "9" NA + one warning
sdv_resolve(c("OAK", "SD", "STL"), "nfl")
#> "13" "24" "14" LV, LAC, LA
sdv_suggest("Kansas Cty Chiefs", "nfl")
```

`sdv_resolve(x, league, season = NULL, id_system = "auto", strict = FALSE)` reads abbreviations, names and ids from ESPN, nflverse, MLB Stats, nba_api, NHL, HockeyTech, NCAA, PFF, CFBD, Sports Reference, FanGraphs and more. `id_system` names one ("nhl_id" only explicitly). `season` picks who held a reused code (ending year for split seasons). Unresolved: NA and one warning (`strict = TRUE` errors). `sdv_suggest(x, league, n = 5)` lists candidates, never picks.

Every helper that takes a team resolves it this way, with `league` and `id_system`. `strict`: the logo-URL, colour, tier and mark-drawing helpers; not `sdv_marks()`, `element_sdv_*()`, `scale_x/y_sdv()`, `sdv_colorway()`, `sdv_team_factor()`.

TEAM FACTOR

```
sdv_team_factor(c("KC", "BUF", "invalid"),
  league = "nfl")
#> KC BUF <NA> Levels: BUF KC
```

Levels ordered by abbreviation; unresolved values become NA, so scales drop them.

COLOURS & PALETTES

```
sdv_team_colors("nfl", "KC")
sdv_team_colors("nfl", c("KC", "BUF"),
  which = "all")
sdv_team_colors("nba", which = "all")
sdv_color_palette("nba",
  teams = c("BOS", "NY"))
```

`sdv_team_colors(league, team = NULL, which = "primary", season, id_system, strict, include_conferences = FALSE)`: a named hex vector (NA where a team does not resolve); `which = "all"` gives "primary, secondary" strings, or with `team = NULL` a dataframe `team_id abbr name primary secondary`. `sdv_color_palette(league, teams = NULL, which = "primary", ...)`: named colours for `scale_*_manual()`.

IMAGE URLS & MARKS

```
sdv_logo_url(c("KC", "Buffalo Bills",
"OAK"), "nfl")
sdv_logo_url("NYJ", "nfl",
  variant = "scoreboard")
sdv_logo_url("QUE", "nhl", season = 1990)
sdv_logo_url("KC", "nfl",
  mark_type = "wordmark")
sdv_marks("KC", "nfl") # every mark
sdv_headshot_url("2544", "nba",
  id_system = "league")
```

`sdv_logo_url(team, league, season = NULL, variant = "default", mark_type = "logo", id_system, strict)`: the SportsDataverse archive's immutable copy. `variant`: "default", "dark", or a name from `sdv_marks()` ("scoreboard", NFL "grayscale", MLB "cap_on_light", ...); a team without it gets its closest mark of the same polarity, an unknown name errors. `mark_type = "wordmark"`: NFL, MLB (and MiLB).

`sdv_headshot_url(player_id, league, id_system = "espn")`: ESPN headshots for nfl nba wnba mlb nhl cfb mbb wbb; "gsis" and "league" as on page 1. NA for a missing or malformed id.

CACHE & VERSIONS

Index and logo manifest come from the SportsDataverse CDN, cached for a day; a copy ships in the package (`options(sdvplotR.offline = TRUE)`). `sdv_versions()`: what is loaded, from where. `sdv_clear_cache()`: drop index, manifest, NFL headshot map and ggpath images.

COVERAGE

Leagues	Teams
nfl nba mlb nhl wnba	32 · 30 · 30 · 32 · 15
nbagl milb pwhl phf	32 · 286 · 12 · 8
ufl xfl usfl aaf	11 · 11 · 9 · 8
ahl echl ohl whl qmjhl ushl	61 · 59 · 27 · 23 · 30 · 17
cfb mbb wbb	687 · 366 · 364
ncaa_baseball softball mhockey whockey	414 · 415 · 110 · 48
soccer · cricket	2,631 · 111

Team rows in the bundled index `0aeef853ec2b`; the live index can differ.

PREMADE PLOT: TEAM TIERS

```
tiers <- data.frame(
  tier_no = c(1, 1, 2, 2, 3),
  team = c("KC", "DET", "BAL", "PHI", "BUF"))
sdv_team_tiers(tiers, league = "nfl",
  theme = "light", no_line_below_tier = 2)
```

`sdv_team_tiers(data, league, title, subtitle, caption = NULL, tier_desc, presort = FALSE, alpha = 0.8, width = 0.075, no_line_below_tier = NULL, devel = FALSE, theme = "dark", variant = "auto", season, id_system, strict)`. data needs tier_no and team (optional tier_rank). Title "NFL Team Tiers" from league; NULL drops one. `devel = TRUE` plots keys, not logos; "auto": dark marks on the dark theme.

Cookbook: [College hoops tiers ↗](#)

INTERACTIVE: GGIRAPH, PLOTLY, LEAFLET

```
p <- ggplot(d, aes(x, y)) +
  geom_sdv_logos_interactive(aes(team = team,
  tooltip = team, data_id = team),
  league = "nfl", width = 0.1)
ggiraph::girafe(ggobj = p)
plotly::plot_ly(d, x = ~x, y = ~y) |>
  add_sdv_logos(d$x, d$y, d$team, "nfl") |>
  sdv_colorway("nfl", d$team)
leaflet::leaflet(d) |> leaflet::addTiles() |>
  addSdvLogoMarkers(~lon, ~lat, ~team,
  league = "nfl")
```

`geom_sdv_{logos,wordmarks,headshots}_interactive()`: images plus an invisible hover point (`hover_size = 8`). `plotly`: `add_sdv_logos(p, x, y, team, league, size = 0.08, ...)`, `add_sdv_wordmarks()`, `add_sdv_headshots()`, `sdv_axis_logos(p, teams, league, axis)` tick logos, `sdv_colorway()`; `ggplotly()` drops them. `addSdvLogoMarkers(..., size = 32)`: no logo keeps the default marker.

Cookbook: [Interactive plots ↗](#)

PLAYING SURFACES

```
sdv_surface("nba", "BOS",
  center_logo = TRUE) +
  geom_point(aes(x, y), data = shots)
```

league	sportyR surface
nfl aaf ufl usfl xfl cfb	<code>geom_football("nfl" / "ncaa")</code>
nba nbagl wnba mbb wbb fiba	<code>geom_basketball()</code> ; fiba 28 × 15 m
mlb milb ncaa_baseball	<code>geom_baseball()</code>
nhl ahl echl ohl qmjhl ushl pwhl phf ncaa_mhockey ncaa_whockey	<code>geom_hockey()</code> ; whl on the NHL rink
soccer	<code>geom_soccer("fifa")</code> , 105 × 68 m

`sdv_surface(league, team = NULL, season = NULL, center_logo = FALSE, ...)` (not cricket, ncaa_softball). A team paints the paint and apron, end zones, or centre line, circle and boards; baseball stays regulation, soccer gets a centre logo only. Stylized, not real floors. Origin at centre, in feet (yards for football, metres for soccer and FIBA); ... goes to `sportyR( rotation, xlims, color_updates )`.

Cookbook: [Surfaces and shot charts ↗](#)

SHOT & EVENT COORDINATES

```
# stats.nba.com legacy frame -> feet
shots |> sdv_court_coords()
# Euroleague (cm, hoop origin) -> FIBA m
pts |> sdv_court_coords(
  x_column = "coord_x", y_column = "coord_y",
  provider = "euroleague")
# soccer events -> 105 x 68 m pitch
ev |> sdv_pitch_coords("statsbomb")
```

`sdv_court_coords(data, x_column = "x_legacy", y_column = "y_legacy", provider = "nba")` adds `court_x court_y`: basket at -41.75 ft (-12.425 m for Euroleague), half court at 0; `court_y` -25 to 25 ft, negative = shooter's left. From hoopR's `nba_shotchartdetail()`: `sdv_court_coords(res$Shot_Chart_Detail, "LOC_X", "LOC_Y")`.

`sdv_pitch_coords(data, provider, x_column = NULL, y_column = NULL, flip = NULL, pitch_length = NULL, pitch_width = NULL)` adds `pitch_x pitch_y` in metres from the centre spot (`pitch_x` -52.5 = defended goal line). `flip`: TRUE / FALSE or a logical column, as `flip = "away"`, turns rows half a turn. `pitch_length / pitch_width`: tracking only. Providers: opta (statsperform) wyscout statsbomb uefa impect espn, tracking tracab skillcorner secondspectrum metrica.

Cookbook: [NHL shot map ↗](#) · [Soccer ↗](#)



sdvplotR :: gt TABLES

Team images and colours for **gt**, plus the **gtUtils** toolkit (by Andrew Weatherman): 20 themes, legends, colouring, annotations, layouts and image export. Every helper takes a `gt_tbl` as its first argument (`gt_object`) and returns one, so they chain with `|>`. Examples run on `sdv_example_standings` (real NFL and NBA final standings). Page 3: images, themes, colour, legends, focus. Page 4: format, layout, saving, reactable.

sdvplotR v0.1.0

gt ≥ 0.10 · paletteer optional

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GT · IMAGES, THEMES, COLOUR

TEAM IMAGES IN CELLS

```
gt_sdv_logos(gt_object, columns, league,
  height = 30, locations = NULL,
  include_name = FALSE, season = NULL,
  id_system = "auto", strict = FALSE)
gt_sdv_wordmarks(gt_object, columns,
  league, height = 30, locations = NULL,
  season = NULL, id_system, strict)
gt_sdv_headshots(gt_object, columns,
  league, height = 30, locations = NULL,
  id_system = "espn")
gt_sdv_cols_label(gt_object,
  columns = everything(), league,
  height = 30, mark_type = "logo",
  id_system, season, strict)
```

`height` in px. `locations` (`cells_body()`, `cells_stub()`, `cells_column_labels()`, `cells_row_groups()`) replaces `columns`. `include_name` = `TRUE` keeps the text after the logo; `season` shows one season's marks in every cell. Wordmarks: NFL, MLB (and MiLB). `gt_sdv_cols_label()` turns column *labels* into images: `mark_type` = "logo" "wordmark" "headshot".

STACKED NAME IN TEAM COLOUR

```
gt_merge_stack_team_color(gt_object,
  col1, col2, team_col, league,
  font_size_top = 14,
  font_size_bottom = 12, color = "black",
  background = NULL, season, id_system)
```

`col1` on top in bold caps, `col2` below in the team's colour; a primary under 4.5:1 contrast gives way to the secondary or is darkened. Apply a theme first: `background` = `NULL` reads it.

```
x <- subset(sdv_example_standings,
  league == "nfl",
  c(team, team_name, division))
gt(x) |>
  gt_merge_stack_team_color(team_name,
    division, team, league = "nfl")
```

EXAMPLE

```
library(gt)
nfc <- subset(sdv_example_standings,
  division == "NFC Wins",
  c(team, team_name, wins, losses,
    last_season_wins))
gt(nfc) |>
  gt_sdv_logos(columns = team,
    league = "nfl", height = 28) |>
  gt_theme_sdv()
```

Cookbook: [Team tables with the gt toolkit ↗](#)

THEMES

```
gt_theme_sdv(gt, style = "light")
gt_theme_sdv_team(gt, team = "KC",
  league = "nfl")
```



savant broadsheet midnight

`gt_theme_sdv()`: Chivo titles and labels, a Lato body in tubular figures, and the SDV gradient (#3346F0 → #7FE6DC) as one horizon line under the labels; `style` = "dark" sets it on the SportsDataverse navy.

`gt_theme_sdv_team()`: the sdv table in one team's colours (title block primary, horizon line secondary); `team` = `NULL` keeps SportsDataverse colours; takes `season`, `id_system`.

Every `gt_theme_<name>(gt_object, ..., density)` takes `density` = "comfortable" "compact" "social" (default comfortable; compact for almanac, scoreboard, terminal) and passes ... to `gt::tab_options()`, applied last.

Option	Themes (default)
accent	almanac "#8C2F1E" · booktabs, swiss, tuftes "#111111" · broadsheet "#A6081A" · brutalist "#FF3B00" · midnight "#5B8DEF" · scoreboard "#0E1621" · terminal "#FFB86C"
style	sdv, sofa "light" / "dark" · tier "dark" / "light"
other	drench color = "#123F5E" · broadsheet paper = "white" or "salmon" · almanac stripe (NA = off)
none	athletic gtutils kenpom ncaa pl savant

All 20: almanac athletic booktabs broadsheet brutalist drench gtutils kenpom midnight ncaa pl savant scoreboard sdv sdv_team sofa swiss terminal tier tuftes (each `gt_theme_<name>()`).

```
gt(nfc) |>
  gt_theme_broadsheet(paper = "salmon",
    density = "compact")
gt(nfc) |>
  gt_theme_terminal(accent = "#7EE787")
```

Cookbook: [SportsDataverse table themes ↗](#)

PREVIEW & PALETTES

```
gt_theme_preview(nfc, themes = c(
  "gt_theme_sdv", "gt_theme_kenpom"))
gt_color_ranks(..., palette = pal_midnight)
```

`gt_theme_preview(data, themes = NULL, n = 5, ncol = 3, density = "compact", file = NULL, ...)`: `NULL` shows every theme; themes needs full function names. ... goes to `gt_grid()`.

`pal_midnight`: a rank palette that reads on midnight / terminal.

COLOUR BY VALUE

Function	Key arguments
gt_color_ranks()	fill cells: palette domain reverse, na_color = "white", autocolor_text = TRUE
gt_color_pills()	pill: fill_type = "continuous" / "rank", format_type, suffix, digits
gt_percentile_bar()	bar+marker: domain = c(0, 100), scale = "auto" (0-1 → 0-100), na_label = "-"
gt_color_results()	row colours: result_column = "result", result_type = "wl" / "binary"
gt_row_accent()	edge bar: column, palette, side = "left", hide = TRUE
gt_indicator_boxes()	boxes: indicator_vals = c(0, 1), color_yes, show_text
gt_tiers()	tier list: levels, colors, tier_column = "tier", style = "dark"

Most take columns and rows (an expression such as `wins > 10`, or indices). `palette` also takes a paletter name, e.g. "viridis:mako"; domain = `NULL` uses the data's range.

Cookbook: [Percentile bars and cut lines ↗](#)

TIER LIST

```
gt(seeds) |> # tier + logo URL columns
  gt_tiers(levels = c("Bye", "Wild card"),
    colors = c("#1B7837", "#B2182B")) |>
  gt_legend_discrete()
```

Columns other than `tier_column` render as images (`image_columns` narrows them; `img_height` = "55px"), e.g. a column of `sdv_logo_url()` URLs. Levels may be one named vector, `c(Bye = "#1B7837")`.

Cookbook: [Tier lists ↗](#)

LEGENDS

Function	Key arguments
gt_legend_continuous()	bar: type = "continuous" "steps" "blocks", n_bins = 5, labels ("edges"), title, location = "bottom", width = 200
gt_legend_discrete()	swatches: key_info (label = colour), heading, location = "top", shape, direction
gt(nfc) > gt_color_ranks(wins) > gt_legend_continuous(type = "steps", title = "Wins")	

The continuous legend reads the columns, palette and domain the colouring step recorded; the discrete one reads the key `gt_tiers()` recorded, so `key_info` can be left out.

```
gt(afc_west) |>
  gt_legend_discrete(c("Playoffs" =
    "#CCE7F5", "Missed" = "#F5CCCC"))
```

Cookbook: [Headers, legends, captions ↗](#)

HIGHLIGHT & FOCUS

Function	Key arguments
gt_highlight_cells()	columns, condition (~ .x > 0.7 or a logical mask), fill = "#FFF3B0"
gt_highlight_na()	fill = "#F0F0F0", missing_text, na_strings = "NA"
gt_outliers()	method = "iqr" "sd" "bounds", threshold (1.5/3), side, symbol, note
gt_spotlight()	rows; dims the rest (dim_color = "auto", WCAG AA); accent_color, if_none
gt_bold_rows()	rows, text_color = "black", highlight_color
gt_group_strips()	band row groups: color = "#F5F5F5", start = 2

```
gt(nfc) |>
  gt_spotlight(team == "DET",
    accent_color = "#0076B6") |>
  gt_outliers(wins, method = "sd",
    note = TRUE)
gt(nfc) |>
  gt_highlight_cells(c(wins, losses),
    ~ .x >= 10, bold = TRUE)
```



sdvplotR :: gt LAYOUT & reactable

The rest of the gtUtils toolkit for **gt**: number formats, rules and border bars, headers and captions, multi-table layouts and image export for posting. Then the **reactable** helpers: team logos, wordmarks and headshots in cells and headers, and team-coloured bars and fills. Examples continue with `nfc` (the NFC North rows of `sdv_example_standings`, page 3).

sdvplotR v0.1.0

webshot2 to save · reactable for tables

GALLERY ↗

COOKBOOKS ↗

GT · FORMAT, LAYOUT, SAVE

FORMAT

Function	Key arguments
<code>gt_delta()</code>	change column to - from: column_label = "Change", percent = FALSE, decimals = 1, arrows = FALSE, after
<code>gt_fmt_rank()</code>	1 → 1st: superscript = TRUE, suffix_size = "0.7em"
<code>gt_fmt_tally()</code>	W-L(-T) in one cell: separator = "-" , share = FALSE, share_of = 1, share_location = "inline" / "column"
<code>gt_significance()</code>	stars from p_columns: levels = c(0.01, 0.05, 0.1), symbols, hide_p = TRUE
<code>gt_scale_note()</code>	divide and say so: divisor = 1000 ("Figures in thousands."), where = "source_note" "label" "both"
<code>gt_wrap_labels()</code>	break long labels: width = 12, balance = TRUE
<code>gt_set_font()</code>	one family for the table: font_family, from_google_font = TRUE, weight, style
<code>gt(nfc) > gt_fmt_tally(c(wins, losses)) > gt_delta(from = last_season_wins, to = wins)</code>	

Cookbook: [Rolling-window wins](#) ↗

BORDERS & RULES

Function	Key arguments
<code>gt_border_grid()</code>	full grid: color = "black", weight = 1, include_labels = FALSE
<code>gt_border_bars_top() / _bottom()</code>	colour bars: colors, bar_height = 10, img, text, text_size = 18
<code>gt_cutline()</code>	line after rows: after, label, color = "#A6081A", style = "dashed", gap
<code>gt(nfc) > gt_border_bars_top(c("#0076B6", "#B0B7BC")) > gt_cutline(after = c(1, 3), label = c("Champion", NA))</code>	

Cookbook: [Border bars](#) ↗

HEADERS & ANNOTATIONS

Function	Key arguments
<code>gt_title_header()</code>	title, subtitle, kicker, date, one *_style list each
<code>gt_column_subheaders()</code>	per column list(heading, subtitle) in ...; subtitle_color = "#808080"
<code>gt_538_caption()</code>	top_caption over a rule, bottom_caption, align = "right"
<code>gt_marginalia()</code>	a column as notes: width = 220, italic = TRUE, rule = TRUE
<code>gt_social_tag()</code>	handles + icons: accounts, caption, stack = FALSE, separator = " "
<code>gt_watermark()</code>	text or image, opacity = 0.06, size = "60%", angle
<code>gt(nfc) > gt_title_header("NFC North", kicker = "2025 SEASON", subtitle = "Final standings") > gt_column_subheaders(wins = list(heading = "W", subtitle = "2025")) > gt_social_tag(c(x = "@SportsDataverse", gh = "sportsdataverse"))</code>	

`accounts` names map to Font Awesome brand icons, with aliases (x / twitter, ig, bsky, gh). A Date in date prints as "July 21, 2026".

Cookbook: [Headers, legends, captions](#) ↗

SNAKE A LONG TABLE

<code>nba <- subset(sdv_example_standings, league == "nba", c(team, wins, losses)) gt(nba) > gt_snake(n_cols = 3)</code>

Thirty rows become three blocks of ten, each with its own labels. Body-cell styling applied before `gt_snake()` carries through; formats and content transforms (such as `gt_sdv_logos()`) go after it. `gt_snake_align()` reshapes a precomputed mask or colour frame to match.

LEARN MORE

GALLERY ↗ every chart and table in one place

Start: [Getting started](#) · **All:** [articles](#) · [reference](#)

Per league: [NFL](#) · [CFB](#) · [NBA](#) · [WNBA](#) · [MLB](#) · [NHL](#) · [MBB](#) · [WBB](#) · [Soccer](#)

Recipes: [NFL EPA](#) · [CFB table](#) · [WBB standings](#) · [head-to-head card](#)

Leaderboards: [dashboards with Quarto](#) · **Automation:** [social graphics](#)

LAYOUT

Function	Key arguments
<code>gt_grid()</code>	tables in a grid: tables, ncol = 2, labels, title, caption, gap = 24, align = "top", file
<code>gt_stack_tables()</code>	one above another: gap = 16, align = "center", title, file
<code>gt_snake()</code>	long table in blocks: n_cols = 2 or rows_per_col, gap = 20, fill = ""
<code>gt_snake_align()</code>	reshape a parallel frame (a mask for gt_highlight_cells()) to match
<code>gt_grid(list(afc_tbl, nfc_tbl), ncol = 2, title = "2025 standings", file = "standings.png")</code>	

Grids and stacks share `subtitle`, `source_note`, `caption_rule`, the `*_style` lists, and the `save` arguments `bg = "white"`, `whitespace = 50`, `zoom = 2`.

Cookbook: [Faceted tables](#) ↗ · [Schedule matrix](#) ↗

SAVE AS AN IMAGE

Function	Key arguments
<code>gt_save_crop()</code>	PNG/JPG, trimmed and padded: file = NULL, bg = "white", whitespace = 50, zoom = 2, expand = 5, width
<code>gt_social_crop()</code>	pad to aspect_ratio = "1:1", "16:9", "4:5" or a number; whitespace = 60, gravity = "center"
<code>gt_save_batch()</code>	one image per group: data, group, fn, file, dir, match_width = TRUE
<code>tbl > gt_save_crop("table.png", whitespace = 40, width = 1200) # one image per conference nfl <- subset(sdv_example_standings, league == "nfl") build <- function(df, group) gt(df) > tab_header(title = group) gt_save_batch(nfl, conference, build, "nfl-{group}.png", dir = "out")</code>	

Saving runs headless Chrome through `webshot2` (Chrome gets 60 s to start). `file = NULL` returns the PNG bytes. `dir` has no default, so a batch never lands in the working directory unasked.

Cookbook: [Saving and posting](#) ↗ · [Social posting](#) ↗

REACTABLE

REACTABLE TABLES

Helper	Arguments (defaults)
<code>reactable_sdv_logos()</code>	league, variant = "default" ("dark", a named variant), height = 30, default_img = NULL
<code>reactable_sdv_wordmarks()</code>	the same
<code>reactable_sdv_headshots()</code>	league, height = 40, default_img, id_system = "espn"
<code>reactable_sdv_team_color_bar()</code>	data, team_col, league, which = "primary", max_value = NULL, na_color = "grey70"
<code>reactable_sdv_team_color_bg()</code>	as the bar, with alpha = 0.15 in place of max_value
<code>reactable_sdv_cols_label()</code>	.data, ... (to every colDef()), league, variant, height = 30

```
library(reactable)
reactable(df, columns = list(
  team = colDef(html = TRUE,
    cell = reactable_sdv_logos("nfl")),
  wins = colDef(style =
    reactable_sdv_team_color_bar(
      df, "team", league = "nfl"))))
# columns named by team key -> logo headers
reactable(h2h,
  columns = reactable_sdv_cols_label(
    h2h, league = "nba"))
```

All but headshots also take `season`, `id_system`, `strict`. The first three are `cell` renderers (`colDef(html = TRUE)`); unresolved values show as text, or `default_img`; a variant that fails to load swaps to the default image. The colour helpers are `style` functions: the bar's length is the value over `max_value` (the column maximum by default). `reactable_sdv_cols_label()` leaves out columns it cannot resolve, so they keep their plain names.

Cookbook: [reactable integration](#) ↗

SIBLINGS

Python [sdvplot](#) (PyPI) and JavaScript [@sportsdataverse/sdvplot](#) + [sdvtables](#) (npm) share `sdvplotR`'s team index, logo archive and argument names; images render through [ggpath](#).



sdvplotR :: CHEAT SHEET

Team logos, wordmarks, player headshots and team colours for **28 leagues** (pro, college, minor and junior hockey, spring football, soccer, cricket) in **ggplot2** plots, **gt** and **reactable** tables, and **plotly**, **ggiraph** and **leaflet**, behind one `league` argument. Built on ggpath, after the approach of nflplotR, cfbplotR, nbaplotR and mlbplotR. Page 1: ggplot2 marks, scales and axes. Page 2: teams, interactive, surfaces, coordinates. Page 3: gt images, themes, colour. Page 4: gt layout, saving, reactable.

sdvplotR v0.1.0

R ≥ 4.1 · ggplot2 ≥ 4.0 · ggpath ≥ 1.1

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GGPLOT2 · MARKS & SCALES

INSTALL & LOAD

```
install.packages("sdvplotR")
# dev: sportsdataverse.r-universe.dev
pak::pak("sportsdataverse/sdvplotR")
library(sdvplotR); library(ggplot2)
```

Every team helper takes `league`, a key from `sdv_leagues()` ("nfl" "nba" "cfb" "nhl" "ahl" "soccer" ..., page 2). Teams: any abbreviation, name or id `sdv_resolve()` reads (ESPN, nflverse, nba_api, MLB Stats, CFBD, ...), or a conference ("SEC", "B1G", "AFC").

Cookbook: [Getting started ↗](#)

LOGOS, WORDMARKS & HEADSHOTS

Geom	Required aes
<code>geom_sdv_logos()</code>	<code>x y team</code>
<code>geom_sdv_wordmarks()</code>	<code>x y team (NFL, MLB, MiLB)</code>
<code>geom_sdv_headshots()</code>	<code>x y player_id</code>
<code>geom_from_path()</code>	<code>x y path (any image)</code>

Optional aes and defaults: `alpha`, `colour` ("b/w" greyscales), `angle` = 0, `hjust` = 0.5, `vjust` = 0.5, `width` = 1, `height` = 1 (npc; a typical logo is `width` = 0.075). Logos and wordmarks also take `season` (an aes or a constant).

```
df <- data.frame(
  team = sdv_teams("nba")$abbr,
  a = rnorm(30), b = rnorm(30))
ggplot(df, aes(a, b)) +
  geom_mean_lines(aes(x0 = a, y0 = b)) +
  geom_sdv_logos(aes(team = team),
    league = "nba", width = 0.07)
```

HIGHLIGHT A GROUP

```
nfl <- subset(sdv_example_standings,
  league == "nfl")
west <- c("DEN", "KC", "LAC", "LV")
ggplot(nfl, aes(points_for,
  points_against)) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, !team %in% west),
    league = "nfl", width = 0.05,
    alpha = 0.25) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, team %in% west),
    league = "nfl", width = 0.1)
```

Rows draw in order, so the last layer and row end on top. `colour` = "b/w" greys logos instead of fading them, which reads better on a dark background.

Cookbook: [Logos, wordmarks, headshots ↗](#)

GEOM ARGUMENTS

```
geom_sdv_logos(mapping = NULL,
  data = NULL, stat = "identity",
  position = "identity", ..., league,
  id_system = "auto", strict = FALSE,
  na.rm = FALSE, show.legend = FALSE,
  inherit.aes = TRUE)
geom_sdv_headshots(..., league,
  id_system = "espn")
```

`geom_sdv_wordmarks()` takes the logo arguments; headshots take no `season` or `strict`. Headshot `id_system`: "espn" = ESPN athlete ids in every league; "gsis" = nflverse NFL ids; "league" = NBA / WNBA Stats PERSON_ID, MLBAM, NHL API.

The ggproto objects `GeomSDVLogo`, `GeomSDVwordmark`, `GeomSDVheadshot` and ggpath's `GeomFromPath`, `GeomReflines` are exported for extensions.

ERA LOGOS

```
old <- data.frame(x = 1:4, y = 1,
  team = c("QUE", "HFD", "ATL", "WPG"),
  season = c(1990, 1995, 2005, 2015))
ggplot(old, aes(x, y)) +
  geom_sdv_logos(aes(team = team,
    season = season), league = "nhl",
  width = 0.2) + theme_void()
```

`season`: the mark worn that year (ending year for split seasons); it also picks who held a reused code. NHL: every identity since 1917-18 (QUE HFD); AHL, ECHL, OHL, QMJHL, WHL, USHL, PWHL, PHF, UFL, XFL, AAF: dated marks; NFL / WNBA: relocated identities (STL SD OAK; SA TUL). Elsewhere today's mark. NHL marks are SVG: needs `rsvg`.

REFERENCE LINES

```
p + geom_mean_lines(aes(x0 = a, y0 = b))
p + geom_median_lines(aes(x0 = a),
  colour = "grey40", linetype = 1)
```

From ggpath. Optional aes `x0`, `y0`; defaults `colour` = "red", `linewidth` = 0.5, `linetype` = 2 (dashed), `alpha` = NA.

ANY IMAGE: GEOM_FROM_PATH

```
df$url <- sdv_logo_url(df$team, "nba",
  variant = "dark")
ggplot(df, aes(a, b)) +
  geom_from_path(aes(path = url),
    width = 0.07)
```

`path` is a URL or a local file; same optional aes as the team geoms. Conferences resolve by key or other name: `sdv_logo_url(c("SEC", "B1G"), "cfb")`; a team's code wins (cfb "MAC" is Macalester).

COLOUR & FILL SCALES

```
scale_color_sdv(league, which = "primary",
  values = NULL, ..., aesthetics = "colour",
  breaks = waiver(), na.value = "grey50",
  guide = NULL, alpha = NA)
scale_fill_sdv(...) # aesthetics = "fill"
```

Maps team keys to official colours; which = "secondary" for the second colour, `alpha` fades them (`scales::alpha()`), `values` replaces them, ... goes to `scale_colour_manual()`. `scale_colour_sdv()` is an alias.

```
ggplot(df, aes(team, b)) +
  geom_col(aes(colour = team, fill = team)) +
  scale_color_sdv("nba", "secondary") +
  scale_fill_sdv("nba", alpha = 0.4)
```



NFL primary / secondary, as `sdv_team_colors("nfl", which = "all")` returns them.

Cookbook: [Team colours and themes ↗](#)

TITLES

```
p + ggtitle_image(title_image = "SEA",
  title = "Seattle Seahawks Analysis",
  image_height = 20, image_side = "left",
  league = "nfl") +
  theme_title_image(size = 20, hjust = 0.5)
```

`title_image`: a team key (logo resolved) or any image URL. `image_height` = 15 px by default; `image_side` = "left" or "right"; subtitle. `theme_title_image(...)` is required and passes ... to `ggtext::element_markdown()`.

LOGO AT EACH BAR'S END

```
nba <- subset(sdv_example_standings,
  league == "nba")
ggplot(nba, aes(wins,
  reorder(team, wins))) +
  geom_col(aes(fill = team)) +
  geom_sdv_logos(aes(x = wins + 3,
    team = team), league = "nba",
  height = 0.028) +
  scale_fill_sdv("nba")
```

`height` sizes the logo here: each row is about a thirtieth of the panel.

LOGO AXES

Route	Calls
theme	<code>element_sdv_logo()</code> · <code>element_sdv</code>
element	<code>_wordmark()</code> · <code>element_sdv_headshot()</code>
scale+ theme	<code>scale_x_sdv()</code> / <code>scale_y_sdv()</code> · <code>scale_x_sdv_headshots()</code> + <code>theme_x_sdv()</code> / <code>theme_y_sdv()</code>

```
# element route (size in cm)
ggplot(df, aes(team, b)) + geom_col() +
  theme(axis.text.x =
    element_sdv_logo("nba", size = 1))
```

`element_sdv_logo(league, alpha = NULL, colour = NA, hjust = NULL, vjust = NULL, size = 0.5, season, id_system)`; `_wordmark()` the same; `_headshot()` takes `id_system`, no `season`.

```
# scale route (size in px; needs ggtext)
ggplot(df, aes(b, team)) + geom_col() +
  scale_y_sdv(league = "nba", size = 12) +
  theme_y_sdv()
```

`scale_x_sdv(..., league, expand, guide, position = "bottom", size = 12, season, id_system)`; `scale_y_sdv()` has position = "left". Size is the height on x, the width on y. `scale_x_sdv_headshots()` (size = 20) and `scale_y_sdv_headshots()` (size = 30) take `id_system`, `theme_x_sdv()` / `theme_y_sdv()` render those labels as markdown.

Cookbook: [Logo axes and titles ↗](#)

HEADSHOTS ON A SCATTER

```
ggplot(scorers, aes(ppg, ts)) +
  geom_sdv_headshots(aes(player_id =
    athlete_id), league = "nba",
  height = 0.12)
```

ESPN ids: the `athlete_id` of hoopR, wehoop, cfbfastR rosters. NFL gsis ids (`id_system` = "gsis") go through a headshot map built from nflverse rosters (1999 on); `sdv_clear_cache()` re-reads it.

IMAGES IN THEME ELEMENTS

```
p + theme(plot.background =
  element_sdv_raster("court.png"))
```

`element_sdv_raster(image_path, x, y, width, height, just = "centre", hjust = 0.5, vjust = 0.5, interpolate = TRUE, ...)`: one image in a rectangle; wraps ggpath's `element_raster()` (also exported).

`element_path(alpha = 1, colour = NA, hjust, vjust, angle = 0, size = unit(0.5, "cm"))`: axis text that already holds image paths or URLs.



sdvplotR :: TEAMS, INTERACTIVE & SURFACES

The shared SportsDataverse team index behind every helper: **28 leagues**, any team identifier resolved to one canonical `team_id`, colours, palettes and archived logo URLs. Plus logos on **plotly**, **ggiraph** and **leaflet**, premade tier plots, team-coloured playing surfaces drawn with **sportyR**, and converters that put shot and event coordinates from stats.nba.com, Euroleague and ten soccer providers on one frame.

sdvplotR v0.1.0

sportyR for surfaces · rsvg for SVG logos

GALLERY ↗

COOKBOOKS ↗

TEAM DATA · LEAGUES, KEYS, COLOURS

LEAGUES & TEAMS

```
sdv_leagues() # the 28 league keys
sdv_teams("nfl")
sdv_teams("cfb", include_conferences = TRUE)

Group      sdv_teams() columns
ids,        league team_id abbr name short_name
names      location

grouping    program conference_id conference
colours     color_primary color_secondary
            color_source

team_id is the canonical key (the ESPN id where ESPN has one).
Conference rows ("SEC", "AFC") draw logos only.
```

RESOLVE ANY IDENTIFIER

```
sdv_resolve(c("GSW", "Golden State Warriors",
              "XYZ"), "nba")
#> "9" "9" NA + one warning
sdv_resolve(c("OAK", "SD", "STL"), "nfl")
#> "13" "24" "14" LV, LAC, LA
sdv_suggest("Kansas Cty Chiefs", "nfl")

sdv_resolve(x, league, season = NULL, id_system =
"auto", strict = FALSE) reads abbreviations, names
and ids from ESPN, nflverse, MLB Stats, nba_api, NHL,
HockeyTech, NCAA, PFF, CFBD, Sports Reference, FanGraphs
and more. id_system names one ("nhl_id" only
explicitly). season picks who held a reused code (ending
year for split seasons). Unresolved: NA and one warning
(strict = TRUE errors). sdv_suggest(x, league, n =
5) lists candidates, never picks.

Every helper that takes a team resolves it this way, with
league and id_system. strict: the logo-URL, colour,
tier and mark-drawing helpers; not sdv_marks(),
element_sdv_*(), scale_x/y_sdv(), sdv_colorway(),
sdv_team_factor().
```

TEAM FACTOR

```
sdv_team_factor(c("KC", "BUF", "invalid"),
               league = "nfl")
#> KC BUF <NA> Levels: BUF KC

Levels ordered by abbreviation; unresolved values become
NA, so scales drop them.
```

COLOURS & PALETTES

```
sdv_team_colors("nfl", "KC")
sdv_team_colors("nfl", c("KC", "BUF"),
               which = "all")
sdv_team_colors("nba", which = "all")
sdv_color_palette("nba",
                 teams = c("BOS", "NY"))

sdv_team_colors(league, team = NULL, which =
"primary", season, id_system, strict,
include_conferences = FALSE): a named hex vector (NA
where a team does not resolve); which = "all" gives
"primary, secondary" strings, or with team = NULL a
data frame team_id abbr name primary secondary.
sdv_color_palette(league, teams = NULL, which =
"primary", ...): named colours for scale_*_manual().
```

IMAGE URLS & MARKS

```
sdv_logo_url(c("KC", "Buffalo Bills",
               "OAK"), "nfl")
sdv_logo_url("NYJ", "nfl",
             variant = "scoreboard")
sdv_logo_url("QUE", "nhl", season = 1990)
sdv_logo_url("KC", "nfl",
             mark_type = "wordmark")
sdv_marks("KC", "nfl") # every mark
sdv_headshot_url("2544", "nba",
                 id_system = "league")

sdv_logo_url(team, league, season = NULL,
             variant = "default", mark_type = "logo",
             id_system, strict): the SportsDataverse archive's
immutable copy. variant: "default", "dark", or a
name from sdv_marks() ("scoreboard", NFL
"grayscale", MLB "cap_on_light", ...); a team without
it gets its closest mark of the same polarity, an unknown
name errors. mark_type = "wordmark": NFL, MLB (and
MiLB).

sdv_headshot_url(player_id, league, id_system =
"espn"): ESPN headshots for nfl nba wnba mlb nhl cfb mbb
wbb; "gsis" and "league" as on page 1. NA for a missing
or malformed id.
```

CACHE & VERSIONS

Index and logo manifest come from the SportsDataverse
CDN, cached for a day; a copy ships in the package
(options(sdvplotR.offline = TRUE)).
sdv_versions(): what is loaded, from where.
sdv_clear_cache(): drop index, manifest, NFL headshot
map and ggpath images.

COVERAGE

Leagues	Teams
nfl nba mlb nhl wnba	32 · 30 · 30 · 32 · 15
nbagl mlb pwhl phf	32 · 286 · 12 · 8
ufl xfl usfl aaf	11 · 11 · 9 · 8
ahl echl ohl whl qmjhl ushl	61 · 59 · 27 · 23 · 30 · 17
cfb mbb wbb	687 · 366 · 364
ncaa_baseball softball mhockey whockey	414 · 415 · 110 · 48
soccer · cricket	2,631 · 111

Team rows in the bundled index 0aeef853ec2b; the live
index can differ.

PREMADE PLOT: TEAM TIERS

```
tiers <- data.frame(
  tier_no = c(1, 1, 2, 2, 3),
  team = c("KC", "DET", "BAL", "PHI", "BUF"))
sdv_team_tiers(tiers, league = "nfl",
               theme = "light", no_line_below_tier = 2)

sdv_team_tiers(data, league, title, subtitle,
               caption = NULL, tier_desc, presort = FALSE,
               alpha = 0.8, width = 0.075, no_line_below_tier =
NULL, devel = FALSE, theme = "dark", variant =
"auto", season, id_system, strict). data needs
tier_no and team (optional tier_rank). Title "NFL
Team Tiers" from league; NULL drops one. devel =
TRUE plots keys, not logos; "auto": dark marks on the dark
theme.

Cookbook: College hoops tiers ↗
```

INTERACTIVE: GGIRAPH, PLOTLY, LEAFLET

```
p <- ggplot(d, aes(x, y)) +
  geom_sdv_logos_interactive(aes(team = team,
                                tooltip = team, data_id = team),
                           league = "nfl", width = 0.1)
ggiraph::girafe(ggobj = p)
plotly::plot_ly(d, x = ~x, y = ~y) |>
  add_sdv_logos(d$x, d$y, d$team, "nfl") |>
  sdv_colorway("nfl", d$team)
leaflet::leaflet(d) |> leaflet::addTiles() |>
  addSdvLogoMarkers(~lon, ~lat, ~team,
                  league = "nfl")

geom_sdv_{logos,wordmarks,headshots}
_interactive(): images plus an invisible hover point
(hover_size = 8). plotly: add_sdv_logos(p, x, y,
team, league, size = 0.08, ...),
add_sdv_wordmarks(), add_sdv_headshots(),
sdv_axis_logos(p, teams, league, axis) tick logos,
sdv_colorway(); ggplotly() drops them.
addSdvLogoMarkers(..., size = 32): no logo keeps the
default marker.

Cookbook: Interactive plots ↗
```

PLAYING SURFACES

```
sdv_surface("nba", "BOS",
           center_logo = TRUE) +
  geom_point(aes(x, y), data = shots)

league      sportyR surface
nfl aaf ufl usfl xfl cfb  geom_football("nfl" /
                                "ncaa")

nba nbagl wnba mbb wbb fiba  geom_basketball(); fiba 28
                                × 15 m

mlb mlb ncaa_baseball      geom_baseball()

nhl ahl echl ohl qmjhl ushl pwhl  geom_hockey(); whl on the
phf ncaa_mhockey           NHL rink
ncaa_whockey

soccer      geom_soccer("fifa"), 105
                                × 68 m

sdv_surface(league, team = NULL, season = NULL,
             center_logo = FALSE, ...) (not cricket, ncaa_softball).
A team paints the paint and apron, end zones, or centre line,
circle and boards; baseball stays regulation, soccer gets a
centre logo only. Stylized, not real floors. Origin at centre, in
feet (yards for football, metres for soccer and FIBA); ...
goes to sportyR (rotation, xlims, color_updates).

Cookbook: Surfaces and shot charts ↗
```

SHOT & EVENT COORDINATES

```
# stats.nba.com legacy frame -> feet
shots |> sdv_court_coords()
# Euroleague (cm, hoop origin) -> FIBA m
pts |> sdv_court_coords(
  x_column = "coord_x", y_column = "coord_y",
  provider = "euroleague")
# soccer events -> 105 x 68 m pitch
ev |> sdv_pitch_coords("statsbomb")

sdv_court_coords(data, x_column = "x_legacy",
                 y_column = "y_legacy", provider = "nba") adds
court_x court_y: basket at -41.75 ft (-12.425 m for
Euroleague), half court at 0; court_y -25 to 25 ft, negative =
shooter's left. From hoopR's nba_shotchartdetail():
sdv_court_coords(res$Shot_Chart_Detail, "LOC_X",
                 "LOC_Y").

sdv_pitch_coords(data, provider, x_column =
NULL, y_column = NULL, flip = NULL, pitch_length
= NULL, pitch_width = NULL) adds pitch_x pitch_y
in metres from the centre spot (pitch_x -52.5 = defended
goal line). flip: TRUE / FALSE or a logical column, as flip
= "away", turns rows half a turn. pitch_length /
pitch_width: tracking only. Providers: opta
(statsperform) wyscout statsbomb uefa impect
espn, tracking tracab skillcorner secondspectrum
metrica.

Cookbook: NHL shot map ↗ · Soccer ↗
```



sdvplotR :: gt TABLES

Team images and colours for **gt**, plus the **gtUtils** toolkit (by Andrew Weatherman): 20 themes, legends, colouring, annotations, layouts and image export. Every helper takes a `gt_tbl` as its first argument (`gt_object`) and returns one, so they chain with `|>`. Examples run on `sdv_example_standings` (real NFL and NBA final standings). Page 3: images, themes, colour, legends, focus. Page 4: format, layout, saving, reactable.

sdvplotR v0.1.0

gt ≥ 0.10 · paletteer optional

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GT · IMAGES, THEMES, COLOUR

TEAM IMAGES IN CELLS

```
gt_sdv_logos(gt_object, columns, league,
  height = 30, locations = NULL,
  include_name = FALSE, season = NULL,
  id_system = "auto", strict = FALSE)
gt_sdv_wordmarks(gt_object, columns,
  league, height = 30, locations = NULL,
  season = NULL, id_system, strict)
gt_sdv_headshots(gt_object, columns,
  league, height = 30, locations = NULL,
  id_system = "espn")
gt_sdv_cols_label(gt_object,
  columns = everything(), league,
  height = 30, mark_type = "logo",
  id_system, season, strict)
```

`height` in px. `locations` (`cells_body()`, `cells_stub()`, `cells_column_labels()`, `cells_row_groups()`) replaces `columns`. `include_name` = `TRUE` keeps the text after the logo; `season` shows one season's marks in every cell. Wordmarks: NFL, MLB (and MiLB). `gt_sdv_cols_label()` turns column *labels* into images: `mark_type` = "logo" "wordmark" "headshot".

STACKED NAME IN TEAM COLOUR

```
gt_merge_stack_team_color(gt_object,
  col1, col2, team_col, league,
  font_size_top = 14,
  font_size_bottom = 12, color = "black",
  background = NULL, season, id_system)
```

`col1` on top in bold caps, `col2` below in the team's colour; a primary under 4.5:1 contrast gives way to the secondary or is darkened. Apply a theme first: `background` = `NULL` reads it.

```
x <- subset(sdv_example_standings,
  league == "nfl",
  c(team, team_name, division))
gt(x) |>
  gt_merge_stack_team_color(team_name,
    division, team, league = "nfl")
```

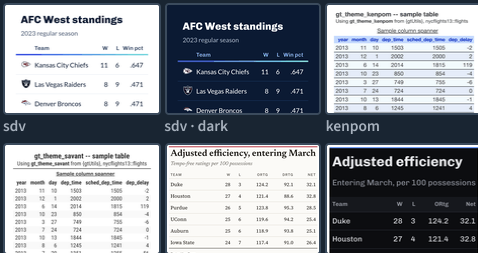
EXAMPLE

```
library(gt)
nfc <- subset(sdv_example_standings,
  division == "NFC North",
  c(team, team_name, wins, losses,
    last_season_wins))
gt(nfc) |>
  gt_sdv_logos(columns = team,
    league = "nfl", height = 28) |>
  gt_theme_sdv()
```

Cookbook: [Team tables with the gt toolkit ↗](#)

THEMES

```
gt_theme_sdv(gt, style = "Light")
gt_theme_sdv_team(gt, team = "KC",
  league = "nfl")
```



savant broadsheet midnight

`gt_theme_sdv()` : Chivo titles and labels, a Lato body in tabular figures, and the SDV gradient (#3346F0 → #7FE6DC) as one horizon line under the labels; `style` = "dark" sets it on the SportsDataverse navy.

`gt_theme_sdv_team()` : the sdv table in one team's colours (title block primary, horizon line secondary); `team` = `NULL` keeps SportsDataverse colours; takes `season`, `id_system`. Every `gt_theme_<name>(gt_object, ..., density)` takes `density` = "comfortable" "compact" "social" (default comfortable; compact for almanac, scoreboard, terminal) and passes ... to `gt::tab_options()`, applied last.

Option	Themes (default)
accent	almanac "#8C2F1E" · booktabs, swiss, tuftes "#111111" · broadsheet "#A6081A" · brutalist "#FF3B00" · midnight "#5B8DEF" · scoreboard "#0E1621" · terminal "#FFB86C"
style	sdv, sofa "light" / "dark" · tier "dark" / "light"
other	drench color = "#123F5E" · broadsheet paper = "white" or "salmon" · almanac stripe (NA = off)
none	athletic gtutils kenpom ncaa pl savant

All 20: almanac athletic booktabs broadsheet brutalist drench gtutils kenpom midnight ncaa pl savant scoreboard sdv sdv_team sofa swiss terminal tier tuftes (each `gt_theme_<name>()`).

```
gt(nfc) |>
  gt_theme_broadsheet(paper = "salmon",
    density = "compact")
gt(nfc) |>
  gt_theme_terminal(accent = "#7EE787")
```

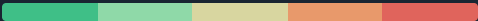
Cookbook: [SportsDataverse table themes ↗](#)

PREVIEW & PALETTES

```
gt_theme_preview(nfc, themes = c(
  "gt_theme_sdv", "gt_theme_kenpom"))
gt_color_ranks(..., palette = pal_midnight)
```

`gt_theme_preview(data, themes = NULL, n = 5, ncol = 3, density = "compact", file = NULL, ...)`: `NULL` shows every theme; themes needs full function names. ... goes to `gt_grid()`.

`pal_midnight`: a rank palette that reads on midnight / terminal.



COLOUR BY VALUE

Function	Key arguments
gt_color_ranks()	fill cells: palette domain reverse, na_color = "white", autocolor_text = TRUE
gt_color_pills()	pill: fill_type = "continuous" / "rank", format_type, suffix, digits
gt_percentile_bar()	bar+marker: domain = c(0, 100), scale = "auto" (0-1 → 0-100), na_label = "-"
gt_color_results()	row colours: result_column = "result", result_type = "wl" / "binary"
gt_row_accent()	edge bar: column, palette, side = "left", hide = TRUE
gt_indicator_boxes()	boxes: indicator_vals = c(0, 1), color_yes, show_text
gt_tiers()	tierlist: levels, colors, tier_column = "tier", style = "dark"

Most take `columns` and `rows` (an expression such as `wins > 10`, or indices). `palette` also takes a paletteer name, e.g. "viridis::mako"; `domain` = `NULL` uses the data's range.

Cookbook: [Percentile bars and cut lines ↗](#)

TIER LIST

```
gt(seeds) |> # tier + logo URL columns
  gt_tiers(levels = c("Bye", "Wild card"),
    colors = c("#1B7837", "#B2182B")) |>
  gt_legend_discrete()
```

Columns other than `tier_column` render as images (`image_columns` narrows them; `img_height` = "55px"), e.g. a column of `sdv_logo_url()` URLs. Levels may be one named vector, `c(Bye = "#1B7837")`.

Cookbook: [Tier lists ↗](#)

LEGENDS

Function	Key arguments
gt_legend_continuous()	bar: type = "continuous" "steps" "blocks", n_bins = 5, labels ("edges"), title, location = "bottom", width = 200
gt_legend_discrete()	swatches: key_info (label = colour), heading, location = "top", shape, direction
gt(nfc) > gt_color_ranks(wins) > gt_legend_continuous(type = "steps", title = "Wins")	

The continuous legend reads the columns, palette and domain the colouring step recorded; the discrete one reads the key `gt_tiers()` recorded, so `key_info` can be left out.

```
gt(afc_west) |>
  gt_legend_discrete(c("Playoffs" =
    "#CCE7F5", "Missed" = "#F5CCCC"))
```

Cookbook: [Headers, legends, captions ↗](#)

HIGHLIGHT & FOCUS

Function	Key arguments
gt_highlight_cells()	columns, condition (~ .x > 0.7 or a logical mask), fill = "#FFF3B0"
gt_highlight_na()	fill = "#F0F0F0", missing_text, na_strings = "NA"
gt_outliers()	method = "iqr" "sd" "bounds", threshold (1.5/3), side, symbol, note
gt_spotlight()	rows; dims the rest (dim_color = "auto", WCAG AA); accent_color, if_none
gt_bold_rows()	rows, text_color = "black", highlight_color
gt_group_strips()	band row groups: color = "#F5F5F5", start = 2

```
gt(nfc) |>
  gt_spotlight(team == "DET",
    accent_color = "#0076B6") |>
  gt_outliers(wins, method = "sd",
    note = TRUE)
gt(nfc) |>
  gt_highlight_cells(c(wins, losses),
    ~ .x >= 10, bold = TRUE)
```



sdvplotR :: gt LAYOUT & reactable

The rest of the gtUtils toolkit for **gt**: number formats, rules and border bars, headers and captions, multi-table layouts and image export for posting. Then the **reactable** helpers: team logos, wordmarks and headshots in cells and headers, and team-coloured bars and fills. Examples continue with `nfc` (the NFC North rows of `sdv_example_standings`, page 3).

sdvplotR v0.1.0

webshot2 to save · reactable for tables

GALLERY ↗

COOKBOOKS ↗

GT · FORMAT, LAYOUT, SAVE

FORMAT

Function	Key arguments
<code>gt_delta()</code>	change column to - from: column_label = "Change", percent = FALSE, decimals = 1, arrows = FALSE, after
<code>gt_fmt_rank()</code>	1→1st: superscript = TRUE, suffix_size = "0.7em"
<code>gt_fmt_tally()</code>	W-L(-T) in one cell: separator = "-" , share = FALSE, share_of = 1, share_location = "inline" / "column"
<code>gt_significance()</code>	stars from p_columns: levels = c(0.01, 0.05, 0.1), symbols, hide_p = TRUE
<code>gt_scale_note()</code>	divide and say so: divisor = 1000 ("Figures in thousands."), where = "source_note" "label" "both"
<code>gt_wrap_labels()</code>	break long labels: width = 12, balance = TRUE
<code>gt_set_font()</code>	one family for the table: font_family, from_google_font = TRUE, weight, style

```
gt(nfc) |>
  gt_fmt_tally(c(wins, losses)) |>
  gt_delta(from = last_season_wins,
    to = wins)
```

Cookbook: [Rolling-window wins](#) ↗

BORDERS & RULES

Function	Key arguments
<code>gt_border_grid()</code>	full grid: color = "black", weight = 1, include_labels = FALSE
<code>gt_borderBars_top()</code> / <code>_bottom()</code>	colour bars: colors, bar_height = 10, img, text, text_size = 18
<code>gt_cutline()</code>	line after rows: after, label, color = "#A6081A", style = "dashed", gap

```
gt(nfc) |>
  gt_borderBars_top(
    c("#0076B6", "#B0B7BC")) |>
  gt_cutline(after = c(1, 3),
    label = c("Champion", NA))
```

Cookbook: [Border bars](#) ↗

HEADERS & ANNOTATIONS

Function	Key arguments
<code>gt_title_header()</code>	title, subtitle, kicker, date, one *_style list each
<code>gt_column_subheaders()</code>	per column list(heading, subtitle) in ...; subtitle_color = "#808080"
<code>gt_538_caption()</code>	top_caption over a rule, bottom_caption, align = "right"
<code>gt_marginalia()</code>	a column as notes: width = 220, italic = TRUE, rule = TRUE
<code>gt_social_tag()</code>	handles + icons: accounts, caption, stack = FALSE, separator = " "
<code>gt_watermark()</code>	text or image, opacity = 0.06, size = "60%", angle

```
gt(nfc) |>
  gt_title_header("NFC North",
    kicker = "2025 SEASON",
    subtitle = "Final standings") |>
  gt_column_subheaders(wins = list(
    heading = "W", subtitle = "2025")) |>
  gt_social_tag(c(x = "@SportsDataverse",
    gh = "sportsdataverse"))
```

accounts names map to Font Awesome brand icons, with aliases (x / twitter, ig, bsky, gh). A Date in date prints as "July 21, 2026".

Cookbook: [Headers, legends, captions](#) ↗

SNAKE A LONG TABLE

```
nba <- subset(sdv_example_standings,
  league == "nba", c(team, wins, losses))
gt(nba) |> gt_snake(n_cols = 3)
```

Thirty rows become three blocks of ten, each with its own labels. Body-cell styling applied before `gt_snake()` carries through; formats and content transforms (such as `gt_sdv_logos()`) go after it. `gt_snake_align()` reshapes a precomputed mask or colour frame to match.

LEARN MORE

GALLERY ↗ every chart and table in one place

Start: Getting started · **All:** articles · reference

Per league: NFL · CFB · NBA · WNBA · MLB · NHL · MBB · WBB · Soccer

Recipes: NFL EPA · CFB table · WBB standings · head-to-head card

Leaderboards: dashboards with Quarto · **Automation:** social graphics

LAYOUT

Function	Key arguments
<code>gt_grid()</code>	tables in a grid: tables, ncol = 2, labels, title, caption, gap = 24, align = "top", file
<code>gt_stack_tables()</code>	one above another: gap = 16, align = "center", title, file
<code>gt_snake()</code>	long table in blocks: n_cols = 2 or rows_per_col, gap = 20, fill = ""
<code>gt_snake_align()</code>	reshape a parallel frame (a mask for gt_highlight_cells()) to match
<code>gt_grid(list(afc_tbl, nfc_tbl), ncol = 2, title = "2025 standings", file = "standings.png")</code>	

Grids and stacks share subtitle, source_note, caption_rule, the *_style lists, and the save arguments `bg = "white", whitespace = 50, zoom = 2`.

Cookbook: [Faceted tables](#) ↗ · [Schedule matrix](#) ↗

SAVE AS AN IMAGE

Function	Key arguments
<code>gt_save_crop()</code>	PNG/JPG, trimmed and padded: file = NULL, bg = "white", whitespace = 50, zoom = 2, expand = 5, width
<code>gt_social_crop()</code>	pad to aspect_ratio = "1:1", "16:9", "4:5" or number; whitespace = 60, gravity = "center"

```
gt_save_batch() one image per group: data, group,
fn, file, dir, match_width = TRUE
```

```
tbl |> gt_save_crop("table.png",
  whitespace = 40,
  width = 1200)

# one image per conference
nfl <- subset(sdv_example_standings,
  league == "nfl")
build <- function(df, group)
  gt(df) |> tab_header(title = group)
gt_save_batch(nfl, conference, build,
  "nfl-{group}.png", dir = "out")
```

Saving runs headless Chrome through webshot2 (Chrome gets 60 s to start). `file = NULL` returns the PNG bytes. `dir` has no default, so a batch never lands in the working directory unasked.

Cookbook: [Saving and posting](#) ↗ · [Social posting](#) ↗

REACTABLE

REACTABLE TABLES

Helper	Arguments (defaults)
<code>reactable_sdv_logos()</code>	league, variant = "default" ("dark", a named variant), height = 30, default_img = NULL
<code>reactable_sdv_wordmarks()</code>	the same
<code>reactable_sdv_headshots()</code>	league, height = 40, default_img, id_system = "espn"
<code>reactable_sdv_team_color_bar()</code>	data, team_col, league, which = "primary", max_value = NULL, na_color = "grey70"
<code>reactable_sdv_team_color_bg()</code>	as the bar, with alpha = 0.15 in place of max_value
<code>reactable_sdv_cols_label()</code>	.data, ... (to every colDef()), league, variant, height = 30

```
library(reactable)
reactable(df, columns = list(
  team = colDef(html = TRUE,
    cell = reactable_sdv_logos("nfl")),
  wins = colDef(style =
    reactable_sdv_team_color_bar(
      df, "team", league = "nfl"))))
# columns named by team key -> logo headers
reactable(h2h,
  columns = reactable_sdv_cols_label(
    h2h, league = "nba"))
```

All but headshots also take `season, id_system, strict`. The first three are `cell` renderers (`colDef(html = TRUE)`); unresolved values show as text, or `default_img`; a variant that fails to load swaps to the default image. The colour helpers are `style` functions: the bar's length is the value over `max_value` (the column maximum by default). `reactable_sdv_cols_label()` leaves out columns it cannot resolve, so they keep their plain names.

Cookbook: [reactable integration](#) ↗

SIBLINGS

Python **sdvplot** (PyPI) and JavaScript **@sportsdataverse/sdvplot** + **sdvtables** (npm) share **sdvplotR**'s team index, logo archive and argument names; images render through **ggpath**.



sdvplotR :: CHEAT SHEET

Team logos, wordmarks, player headshots and team colours for **28 leagues** (pro, college, minor and junior hockey, spring football, soccer, cricket) in **ggplot2** plots, **gt** and **reactable** tables, and **plotly**, **ggiraph** and **leaflet**, behind one `league` argument. Built on ggpath, after the approach of nflplotR, cfbplotR, nbaplotR and mlbplotR. Page 1: ggplot2 marks, scales and axes. Page 2: teams, interactive, surfaces, coordinates. Page 3: gt images, themes, colour. Page 4: gt layout, saving, reactable.

sdvplotR v0.1.0

R ≥ 4.1 · ggplot2 ≥ 4.0 · ggpath ≥ 1.1

GALLERY ↗

COOKBOOKS ↗

GGPLOT2 · MARKS & SCALES

INSTALL & LOAD

```
install.packages("sdvplotR")
# dev: sportsdataverse.r-universe.dev
pak::pak("sportsdataverse/sdvplotR")
library(sdvplotR); library(ggplot2)
```

Every team helper takes `league`, a key from `sdv_leagues()` ("nfl" "nba" "cfb" "nhl" "ahl" "soccer" ..., page 2). Teams: any abbreviation, name or id `sdv_resolve()` reads (ESPN, nflverse, nba_api, MLB Stats, CFBD, ...), or a conference ("SEC", "B1G", "AFC").

Cookbook: [Getting started](#) ↗

LOGOS, WORDMARKS & HEADSHOTS

Geom	Required aes
<code>geom_sdv_logos()</code>	<code>x y team</code>
<code>geom_sdv_wordmarks()</code>	<code>x y team (NFL, MLB, MiLB)</code>
<code>geom_sdv_headshots()</code>	<code>x y player_id</code>
<code>geom_from_path()</code>	<code>x y path (any image)</code>

Optional aes and defaults: `alpha`, `colour` ("b/w" greyscales), `angle` = 0, `hjust` = 0.5, `vjust` = 0.5, `width` = 1, `height` = 1 (npc; a typical logo is `width` = 0.075). Logos and wordmarks also take `season` (an aes or a constant).

```
df <- data.frame(
  team = sdv_teams("nba")$abbr,
  a = rnorm(30), b = rnorm(30))
ggplot(df, aes(a, b)) +
  geom_mean_lines(aes(x0 = a, y0 = b)) +
  geom_sdv_logos(aes(team = team),
    league = "nba", width = 0.07)
```

HIGHLIGHT A GROUP

```
nfl <- subset(sdv_example_standings,
  league == "nfl")
west <- c("DEN", "KC", "LAC", "LV")
ggplot(nfl, aes(points_for,
  points_against)) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, !team %in% west),
    league = "nfl", width = 0.05,
    alpha = 0.25) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, team %in% west),
    league = "nfl", width = 0.1)
```

Rows draw in order, so the last layer and row end on top. `colour` = "b/w" greys logos instead of fading them, which reads better on a dark background.

Cookbook: [Logos, wordmarks, headshots](#) ↗

GEOM ARGUMENTS

```
geom_sdv_logos(mapping = NULL,
  data = NULL, stat = "identity",
  position = "identity", ..., league,
  id_system = "auto", strict = FALSE,
  na.rm = FALSE, show.legend = FALSE,
  inherit.aes = TRUE)
geom_sdv_headshots(..., league,
  id_system = "espn")
```

`geom_sdv_wordmarks()` takes the logo arguments; headshots take no `season` or `strict`. Headshot `id_system`: "espn" = ESPN athlete ids in every league; "gsis" = nflverse NFL ids; "league" = NBA / WNBA Stats PERSON_ID, MLBAM, NHL API.

The ggproto objects `GeomSDVLogo`, `GeomSDVwordmark`, `GeomSDVheadshot` and ggpath's `GeomFromPath`, `GeomRefLines` are exported for extensions.

ERA LOGOS

```
old <- data.frame(x = 1:4, y = 1,
  team = c("QUE", "HFD", "ATL", "WPG"),
  season = c(1990, 1995, 2005, 2015))
ggplot(old, aes(x, y)) +
  geom_sdv_logos(aes(team = team,
    season = season), league = "nhl",
    width = 0.2) + theme_void()
```

`season`: the mark worn that year (ending year for split seasons); it also picks who held a reused code. NHL: every identity since 1917-18 (QUE HFD); AHL, ECHL, OHL, QMJHL, WHL, USHL, PWHL, PHF, UFL, XFL, AAF: dated marks; NFL / WNBA: relocated identities (STL SD OAK; SA TUL). Elsewhere today's mark. NHL marks are SVG: needs `rsvg`.

REFERENCE LINES

```
p + geom_mean_lines(aes(x0 = a, y0 = b))
p + geom_median_lines(aes(x0 = a),
  colour = "grey40", linetype = 1)
```

From ggpath. Optional aes `x0`, `y0`; defaults `colour` = "red", `linewidth` = 0.5, `linetype` = 2 (dashed), `alpha` = NA.

ANY IMAGE: GEOM_FROM_PATH

```
df$url <- sdv_logo_url(df$team, "nba",
  variant = "dark")
ggplot(df, aes(a, b)) +
  geom_from_path(aes(path = url),
    width = 0.07)
```

`path` is a URL or a local file; same optional aes as the team geoms. Conferences resolve by key or other name: `sdv_logo_url(c("SEC", "B1G", "cfb"))`; a team's code wins (cfb "MAC" is Macalester).

COLOUR & FILL SCALES

```
scale_color_sdv(league, which = "primary",
  values = NULL, ..., aesthetics = "colour",
  breaks = waiver(), na.value = "grey50",
  guide = NULL, alpha = NA)
scale_fill_sdv(...) # aesthetics = "fill"
```

Maps team keys to official colours; `which` = "secondary" for the second colour, `alpha` fades them (`scales::alpha()`), `values` replaces them, ... goes to `scale_colour_manual()`. `scale_colour_sdv()` is an alias.

```
ggplot(df, aes(team, b)) +
  geom_col(aes(colour = team, fill = team)) +
  scale_color_sdv("nba", "secondary") +
  scale_fill_sdv("nba", alpha = 0.4)
```



NFL primary / secondary, as `sdv_team_colors("nfl", which = "all")` returns them.

Cookbook: [Team colours and themes](#) ↗

TITLES

```
p + ggtitle_image(title_image = "SEA",
  title = "Seattle Seahawks Analysis",
  image_height = 20, image_side = "left",
  league = "nfl") +
  theme_title_image(size = 20, hjust = 0.5)
```

`title_image`: a team key (logo resolved) or any image URL. `image_height` = 15 px by default; `image_side` = "left" or "right"; subtitle. `theme_title_image(...)` is required and passes ... to `ggtext::element_markdown()`.

LOGO AT EACH BAR'S END

```
nba <- subset(sdv_example_standings,
  league == "nba")
ggplot(nba, aes(wins,
  reorder(team, wins))) +
  geom_col(aes(fill = team)) +
  geom_sdv_logos(aes(x = wins + 3,
    team = team), league = "nba",
    height = 0.028) +
  scale_fill_sdv("nba")
```

`height` sizes the logo here: each row is about a thirtieth of the panel.

LOGO AXES

Route	Calls
theme	<code>element_sdv_logo()</code> · <code>element_sdv</code>
element	<code>_wordmark()</code> · <code>element_sdv_headshot()</code>
scale+ theme	<code>scale_x_sdv()</code> / <code>scale_y_sdv()</code> · <code>scale_x_sdv_headshots()</code> + <code>theme_x_sdv()</code> / <code>theme_y_sdv()</code>

```
# element route (size in cm)
ggplot(df, aes(team, b)) + geom_col() +
  theme(axis.text.x =
    element_sdv_logo("nba", size = 1))
```

`element_sdv_logo(league, alpha = NULL, colour = NA, hjust = NULL, vjust = NULL, size = 0.5, season, id_system)`; `_wordmark()` the same; `_headshot()` takes `id_system`, no `season`.

```
# scale route (size in px; needs ggtext)
ggplot(df, aes(b, team)) + geom_col() +
  scale_y_sdv(league = "nba", size = 12) +
  theme_y_sdv()
```

`scale_x_sdv(..., league, expand, guide, position = "bottom", size = 12, season, id_system)`; `scale_y_sdv()` has position = "left". Size is the height on x, the width on y. `scale_x_sdv_headshots()` (size = 20) and `scale_y_sdv_headshots()` (size = 30) take `id_system`. `theme_x_sdv()` / `theme_y_sdv()` render those labels as markdown.

Cookbook: [Logo axes and titles](#) ↗

HEADSHOTS ON A SCATTER

```
ggplot(scorers, aes(ppg, ts)) +
  geom_sdv_headshots(aes(player_id =
    athlete_id), league = "nba",
    height = 0.12)
```

ESPN ids: the `athlete_id` of hoopR, wehoop, cfbfastR rosters. NFL gsis ids (`id_system` = "gsis") go through a headshot map built from nflverse rosters (1999 on); `sdv_clear_cache()` re-reads it.

IMAGES IN THEME ELEMENTS

```
p + theme(plot.background =
  element_sdv_raster("court.png"))
```

`element_sdv_raster(image_path, x, y, width, height, just = "centre", hjust = 0.5, vjust = 0.5, interpolate = TRUE, ...)`: one image in a rectangle; wraps ggpath's `element_raster()` (also exported).

`element_path(alpha = 1, colour = NA, hjust, vjust, angle = 0, size = unit(0.5, "cm"))`: axis text that already holds image paths or URLs.



sdvplotR :: TEAMS, INTERACTIVE & SURFACES

The shared SportsDataverse team index behind every helper: **28 leagues**, any team identifier resolved to one canonical `team_id`, colours, palettes and archived logo URLs. Plus logos on **plotly**, **ggiraph** and **leaflet**, premade tier plots, team-coloured playing surfaces drawn with **sportyR**, and converters that put shot and event coordinates from stats.nba.com, Euroleague and ten soccer providers on one frame.

sdvplotR v0.1.0

sportyR for surfaces · rsvg for SVG logos

GALLERY ↗

COOKBOOKS ↗

TEAM DATA · LEAGUES, KEYS, COLOURS

LEAGUES & TEAMS

```
sdv_leagues() # the 28 league keys
sdv_teams("nfl")
sdv_teams("cfb", include_conferences = TRUE)
```

Group	sdv_teams()	columns
ids, names	league team_id abbr name short_name location	
grouping	program conference_id conference	
colours	color_primary color_secondary color_source	

`team_id` is the canonical key (the ESPN id where ESPN has one). Conference rows (`"SEC"`, `"AFC"`) draw logos only.

RESOLVE ANY IDENTIFIER

```
sdv_resolve(c("GSW", "Golden State Warriors",
              "XYZ"), "nba")
#> "9" "9" NA + one warning
sdv_resolve(c("OAK", "SD", "STL"), "nfl")
#> "13" "24" "14" LV, LAC, LA
sdv_suggest("Kansas Cty Chiefs", "nfl")
```

`sdv_resolve(x, league, season = NULL, id_system = "auto", strict = FALSE)` reads abbreviations, names and ids from ESPN, nflverse, MLB Stats, nba_api, NHL, HockeyTech, NCAA, PFF, CFBD, Sports Reference, FanGraphs and more. `id_system` names one (`"nhl_id"` only explicitly). `season` picks who held a reused code (ending year for split seasons). Unresolved: `NA` and one warning (`strict = TRUE` errors). `sdv_suggest(x, league, n = 5)` lists candidates, never picks.

Every helper that takes a team resolves it this way, with `league` and `id_system`. `strict`: the logo-URL, colour, tier and mark-drawing helpers; not `sdv_marks()`, `element_sdv_*()`, `scale_x/y_sdv()`, `sdv_colorway()`, `sdv_team_factor()`.

TEAM FACTOR

```
sdv_team_factor(c("KC", "BUF", "invalid"),
               league = "nfl")
#> KC BUF <NA> Levels: BUF KC
```

Levels ordered by abbreviation; unresolved values become `NA`, so scales drop them.

COLOURS & PALETTES

```
sdv_team_colors("nfl", "KC")
sdv_team_colors("nfl", c("KC", "BUF"),
               which = "all")
sdv_team_colors("nba", which = "all")
sdv_color_palette("nba",
                 teams = c("BOS", "NY"))
```

`sdv_team_colors(league, team = NULL, which = "primary", season, id_system, strict, include_conferences = FALSE)`: a named hex vector (NA where a team does not resolve); `which = "all"` gives "primary, secondary" strings, or with `team = NULL` a dataframe `team_id abbr name primary secondary`. `sdv_color_palette(league, teams = NULL, which = "primary", ...)`: named colours for `scale_*_manual()`.

IMAGE URLS & MARKS

```
sdv_logo_url(c("KC", "Buffalo Bills",
               "OAK"), "nfl")
sdv_logo_url("NYJ", "nfl",
             variant = "scoreboard")
sdv_logo_url("QUE", "nhl", season = 1990)
sdv_logo_url("KC", "nfl",
             mark_type = "wordmark")
sdv_marks("KC", "nfl") # every mark
sdv_headshot_url("2544", "nba",
                 id_system = "league")
```

`sdv_logo_url(team, league, season = NULL, variant = "default", mark_type = "logo", id_system, strict)`: the SportsDataverse archive's immutable copy. `variant`: "default", "dark", or a name from `sdv_marks()` ("scoreboard", NFL "grayscale", MLB "cap_on_light", ...); a team without it gets its closest mark of the same polarity, an unknown name errors. `mark_type = "wordmark"`: NFL, MLB (and MiLB).

`sdv_headshot_url(player_id, league, id_system = "espn")`: ESPN headshots for nfl nba wnba mlb nhl cfb mbb wbb; "gsis" and "league" as on page 1. NA for a missing or malformed id.

CACHE & VERSIONS

Index and logo manifest come from the SportsDataverse CDN, cached for a day; a copy ships in the package (`options(sdvplotR.offline = TRUE)`). `sdv_versions()`: what is loaded, from where. `sdv_clear_cache()`: drop index, manifest, NFL headshot map and ggpath images.

COVERAGE

Leagues	Teams
nfl nba mlb nhl wnba	32 · 30 · 30 · 32 · 15
nbagl milb pwhl phf	32 · 286 · 12 · 8
ufl xfl usfl aaf	11 · 11 · 9 · 8
ahl echl ohl whl qmjhl ushl	61 · 59 · 27 · 23 · 30 · 17
cfb mbb wbb	687 · 366 · 364
ncaa_baseball softball mhokey whockey	414 · 415 · 110 · 48
soccer · cricket	2,631 · 111

Team rows in the bundled index `0aeef853ec2b`; the live index can differ.

PREMADE PLOT: TEAM TIERS

```
tiers <- data.frame(
  tier_no = c(1, 1, 2, 2, 3),
  team = c("KC", "DET", "BAL", "PHI", "BUF"))
sdv_team_tiers(tiers, league = "nfl",
               theme = "light", no_line_below_tier = 2)
```

`sdv_team_tiers(data, league, title, subtitle, caption = NULL, tier_desc, presort = FALSE, alpha = 0.8, width = 0.075, no_line_below_tier = NULL, devel = FALSE, theme = "dark", variant = "auto", season, id_system, strict)`. data needs `tier_no` and team (optional `tier_rank`). Title "NFL Team Tiers" from `league`; `NULL` drops one. `devel = TRUE` plots keys, not logos; "auto": dark marks on the dark theme.

Cookbook: [College hoops tiers](#) ↗

INTERACTIVE: GGIRAPH, PLOTLY, LEAFLET

```
p <- ggplot(d, aes(x, y)) +
  geom_sdv_logos_interactive(aes(team = team,
                                tooltip = team, data_id = team),
                           league = "nfl", width = 0.1)
ggiraph::girafe(ggobj = p)
plotly::plot_ly(d, x = ~x, y = ~y) |>
  add_sdv_logos(d$x, d$y, d$team, "nfl") |>
  sdv_colorway("nfl", d$team)
leaflet::leaflet(d) |> leaflet::addTiles() |>
  addSdvLogoMarkers(~lon, ~lat, ~team,
                   league = "nfl")
```

`geom_sdv_{logos,wordmarks,headshots}_interactive()`: images plus an invisible hover point (`hover_size = 8`). `plotly`: `add_sdv_logos(p, x, y, team, league, size = 0.08, ...)`, `add_sdv_wordmarks()`, `add_sdv_headshots()`, `sdv_axis_logos(p, teams, league, axis)` tick logos, `sdv_colorway()`; `ggplotly()` drops them. `addSdvLogoMarkers(..., size = 32)`: no logo keeps the default marker.

Cookbook: [Interactive plots](#) ↗

PLAYING SURFACES

```
sdv_surface("nba", "BOS",
            center_logo = TRUE) +
  geom_point(aes(x, y), data = shots)
```

league	sportyR surface
nfl aaf ufl usfl xfl cfb	geom_football("nfl" / "ncaa")
nba nbagl wnba mbb wbb fiba	geom_basketball(); fiba 28 × 15 m
mlb milb ncaa_baseball	geom_baseball()
nhl ahl echl ohl qmjhl ushl pwhl phf ncaa_mhokey ncaa_whockey	geom_hockey(); whl on the NHL rink
soccer	geom_soccer("fifa"), 105 × 68 m

`sdv_surface(league, team = NULL, season = NULL, center_logo = FALSE, ...)` (not cricket, ncaa_softball). A team paints the paint and apron, end zones, or centre line, circle and boards; baseball stays regulation, soccer gets a centre logo only. Stylized, not real floors. Origin at centre, in feet (yards for football, metres for soccer and FIBA); ... goes to `sportyR` (rotation, xlims, color_updates).

Cookbook: [Surfaces and shot charts](#) ↗

SHOT & EVENT COORDINATES

```
# stats.nba.com legacy frame -> feet
shots |> sdv_court_coords()
# Euroleague (cm, hoop origin) -> FIBA m
pts |> sdv_court_coords(
  x_column = "coord_x", y_column = "coord_y",
  provider = "euroleague")
# soccer events -> 105 x 68 m pitch
ev |> sdv_pitch_coords("statsbomb")
```

`sdv_court_coords(data, x_column = "x_legacy", y_column = "y_legacy", provider = "nba")` adds `court_x court_y`: basket at -41.75 ft (-12.425 m for Euroleague), half court at 0; `court_y` -25 to 25 ft, negative = shooter's left. From hoopR's `nba_shotchartdetail()`: `sdv_court_coords(res$Shot_Chart_Detail, "LOC_X", "LOC_Y")`.

`sdv_pitch_coords(data, provider, x_column = NULL, y_column = NULL, flip = NULL, pitch_length = NULL, pitch_width = NULL)` adds `pitch_x pitch_y` in metres from the centre spot (`pitch_x` -52.5 = defended goal line). `flip`: TRUE / FALSE or a logical column, as `flip = "away"`, turns rows half a turn. `pitch_length / pitch_width`: tracking only. Providers: `opta` (statsperform) `wyscout` `statsbomb` `uefa` `impect` `espn`, tracking `tracab` `skillcorner` `secondspectrum` `metrca`.

Cookbook: [NHL shot map](#) ↗ · [Soccer](#) ↗



sdvplotR :: gt TABLES

Team images and colours for **gt**, plus the **gtUtils** toolkit (by Andrew Weatherman): 20 themes, legends, colouring, annotations, layouts and image export. Every helper takes a `gt_tbl` as its first argument (`gt_object`) and returns one, so they chain with `|>`. Examples run on `sdv_example_standings` (real NFL and NBA final standings). Page 3: images, themes, colour, legends, focus. Page 4: format, layout, saving, readable.

sdvplotR v0.1.0

gt ≥ 0.10 · paletteer optional

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GT · IMAGES, THEMES, COLOUR

TEAM IMAGES IN CELLS

```
gt_sdv_logos(gt_object, columns, league,
  height = 30, locations = NULL,
  include_name = FALSE, season = NULL,
  id_system = "auto", strict = FALSE)
gt_sdv_wordmarks(gt_object, columns,
  league, height = 30, locations = NULL,
  season = NULL, id_system, strict)
gt_sdv_headshots(gt_object, columns,
  league, height = 30, locations = NULL,
  id_system = "espn")
gt_sdv_cols_label(gt_object, columns,
  height = 30, mark_type = "logo",
  id_system, season, strict)
```

`height` in px. `locations` (`cells_body()`, `cells_stub()`, `cells_column_labels()`, `cells_row_groups()`) replaces `columns`. `include_name` = `TRUE` keeps the text after the logo; `season` shows one season's marks in every cell. Wordmarks: NFL, MLB (and MiLB). `gt_sdv_cols_label()` turns column *labels* into images: `mark_type` = "logo" "wordmark" "headshot".

STACKED NAME IN TEAM COLOUR

```
gt_merge_stack_team_color(gt_object,
  col1, col2, team_color, league,
  font_size_top = 14,
  font_size_bottom = 12, color = "black",
  background = NULL, season, id_system)
```

`col1` on top in bold caps, `col2` below in the team's colour; a primary under 4.5:1 contrast gives way to the secondary or is darkened. Apply a theme first: `background` = `NULL` reads it.

```
x <- subset(sdv_example_standings,
  league == "nfl",
  c(team, team_name, division))
gt(x) |>
  gt_merge_stack_team_color(team_name,
    division, team, league = "nfl")
```

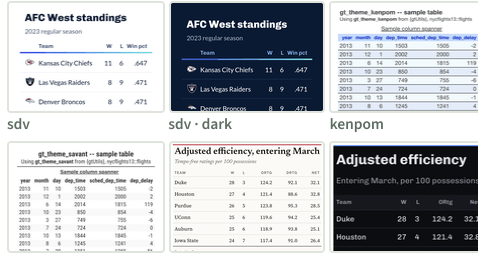
EXAMPLE

```
library(gt)
nfc <- subset(sdv_example_standings,
  division == "NFC Wins",
  c(team, team_name, wins, losses,
    last_season_wins))
gt(nfc) |>
  gt_sdv_logos(columns = team,
    league = "nfl", height = 28) |>
  gt_theme_sdv()
```

Cookbook: [Team tables with the gt toolkit ↗](#)

THEMES

```
gt_theme_sdv(gt, style = "light")
gt_theme_sdv_team(gt, team = "KC",
  league = "nfl")
```



`sdv` `sdv · dark` `kenpom` `savant` `broadsheet` `midnight`

`gt_theme_sdv()`: Chivo titles and labels, a Lato body in tubular figures, and the SDV gradient (#3346F0 → #7FE6DC) as one horizon line under the labels; `style` = "dark" sets it on the SportsDataverse navy.

`gt_theme_sdv_team()`: the `sdv` table in one team's colours (title block primary, horizon line secondary); `team` = `NULL` keeps SportsDataverse colours; takes `season`, `id_system`.

Every `gt_theme_<name>(gt_object, ..., density)` takes `density` = "comfortable" "compact" "social" (default comfortable; compact for almanac, scoreboard, terminal) and passes ... to `gt::tab_options()`, applied last.

Option	Themes (default)
accent	almanac "#8C2F1E" · booktabs, swiss, tuftes "#111111" · broadsheet "#A6081A" · brutalist "#FF3B00" · midnight "#5B8DEF" · scoreboard "#0E1621" · terminal "#FFB86C"
style	sdv, sofa "light" / "dark" · tier "dark" / "light"
other	drench color = "#123F5E" · broadsheet paper = "white" or "salmon" · almanac stripe (NA = off)
none	athletic gtutils kenpom ncaa pl savant

All 20: almanac athletic booktabs broadsheet brutalist drench gtutils kenpom midnight ncaa pl savant scoreboard sdv sdv_team sofa swiss terminal tier tuftes (each `gt_theme_<name>()`).

```
gt(nfc) |>
  gt_theme_broadsheet(paper = "salmon",
    density = "compact")
gt(nfc) |>
  gt_theme_terminal(accent = "#7EE787")
```

Cookbook: [SportsDataverse table themes ↗](#)

PREVIEW & PALETTES

```
gt_theme_preview(nfc, themes = c(
  "gt_theme_sdv", "gt_theme_kenpom"))
gt_color_ranks(..., palette = pal_midnight)
```

`gt_theme_preview(data, themes = NULL, n = 5, ncol = 3, density = "compact", file = NULL, ...)`: `NULL` shows every theme; themes needs full function names. ... goes to `gt_grid()`.

`pal_midnight`: a rank palette that reads on midnight / terminal.



COLOUR BY VALUE

Function	Key arguments
gt_color_ranks()	fill cells: palette domain reverse, na_color = "white", autocolor_text = TRUE
gt_color_pills()	pill: fill_type = "continuous" / "rank", format_type, suffix, digits
gt_percentile_bar()	bar+marker: domain = c(0, 100), scale = "auto" (0-1 → 0-100), na_label = "-"
gt_color_results()	row colours: result_column = "result", result_type = "wl" / "binary"
gt_row_accent()	edge bar: column, palette, side = "left", hide = TRUE
gt_indicator_boxes()	boxes: indicator_vals = c(0, 1), color_yes, show_text
gt_tiers()	tier list: levels, colors, tier_column = "tier", style = "dark"

Most take columns and rows (an expression such as `wins > 10`, or indices). `palette` also takes a paletter name, e.g. "viridis:mako"; domain = `NULL` uses the data's range.

Cookbook: [Percentile bars and cut lines ↗](#)

TIER LIST

```
gt(seeds) |> # tier + logo URL columns
  gt_tiers(levels = c("Bye", "Wild card"),
    colors = c("#1B7837", "#B2182B")) |>
  gt_legend_discrete()
```

Columns other than `tier_column` render as images (`image_columns` narrows them; `img_height` = "55px"), e.g. a column of `sdv_logo_url()` URLs. Levels may be one named vector, `c(Bye = "#1B7837")`.

Cookbook: [Tier lists ↗](#)

LEGENDS

Function	Key arguments
gt_legend_continuous()	bar: type = "continuous" "steps" "blocks", n_bins = 5, labels ("edges"), title, location = "bottom", width = 200
gt_legend_discrete()	swatches: key_info (label = colour), heading, location = "top", shape, direction
gt(nfc) > gt_color_ranks(wins) > gt_legend_continuous(type = "steps", title = "Wins")	

The continuous legend reads the columns, palette and domain the colouring step recorded; the discrete one reads the key `gt_tiers()` recorded, so `key_info` can be left out.

```
gt(afc_west) |>
  gt_legend_discrete(c("Playoffs" =
    "#CCE7F5", "Missed" = "#F5CCCC"))
```

Cookbook: [Headers, legends, captions ↗](#)

HIGHLIGHT & FOCUS

Function	Key arguments
gt_highlight_cells()	columns, condition (~ .x > 0.7 or a logical mask), fill = "#FFF3B0"
gt_highlight_na()	fill = "#F0F0F0", missing_text, na_strings = "NA"
gt_outliers()	method = "iqr" "sd" "bounds", threshold (1.5/3), side, symbol, note
gt_spotlight()	rows; dims the rest (dim_color = "auto", WCAG AA); accent_color, if_none
gt_bold_rows()	rows, text_color = "black", highlight_color
gt_group_strips()	band row groups: color = "#F5F5F5", start = 2

```
gt(nfc) |>
  gt_spotlight(team == "DET",
    accent_color = "#0076B6") |>
  gt_outliers(wins, method = "sd",
    note = TRUE)
gt(nfc) |>
  gt_highlight_cells(c(wins, losses),
    ~ .x >= 10, bold = TRUE)
```



sdvplotR :: gt LAYOUT & reactable

The rest of the gtUtils toolkit for **gt**: number formats, rules and border bars, headers and captions, multi-table layouts and image export for posting. Then the **reactable** helpers: team logos, wordmarks and headshots in cells and headers, and team-coloured bars and fills. Examples continue with **nfc** (the NFC North rows of `sdv_example_standings`, page 3).

sdvplotR v0.1.0

webshot2 to save · reactable for tables

GALLERY ↗

COOKBOOKS ↗

GT · FORMAT, LAYOUT, SAVE

FORMAT

Function	Key arguments
gt_delta()	change column to - from: <code>column_label = "Change"</code> , <code>percent = FALSE</code> , <code>decimals = 1</code> , <code>arrows = FALSE</code> , <code>after</code>
gt_fmt_rank()	1 → 1st: <code>superscript = TRUE</code> , <code>suffix_size = "0.7em"</code>
gt_fmt_tally()	W-L(-T) in one cell: <code>separator = "-"</code> , <code>share = FALSE</code> , <code>share_of = 1</code> , <code>share_location = "inline"</code> / <code>"column"</code>
gt_significance()	stars from <code>p_columns</code> : <code>levels = c(0.01, 0.05, 0.1)</code> , <code>symbols</code> , <code>hide_p = TRUE</code>
gt_scale_note()	divide and say so: <code>divisor = 1000</code> ("Figures in thousands."), <code>where = "source_note"</code> <code>"label"</code> <code>"both"</code>
gt_wrap_labels()	break long labels: <code>width = 12</code> , <code>balance = TRUE</code>
gt_set_font()	one family for the table: <code>font_family</code> , <code>from_google_font = TRUE</code> , <code>weight</code> , <code>style</code>
<pre>gt(nfc) > gt_fmt_tally(c(wins, losses)) > gt_delta(from = last_season_wins, to = wins)</pre>	

Cookbook: [Rolling-window wins](#) ↗

BORDERS & RULES

Function	Key arguments
gt_border_grid()	full grid: <code>color = "black"</code> , <code>weight = 1</code> , <code>include_labels = FALSE</code>
gt_border_bars_top() / _bottom()	colour bars: <code>colors</code> , <code>bar_height = 10</code> , <code>img</code> , <code>text</code> , <code>text_size = 18</code>
gt_cutline()	line after rows: <code>after</code> , <code>label</code> , <code>color = "#A6081A"</code> , <code>style = "dashed"</code> , <code>gap</code>
<pre>gt(nfc) > gt_border_bars_top(c("#0076B6", "#B0B7BC")) > gt_cutline(after = c(1, 3), label = c("Champion", NA))</pre>	

Cookbook: [Border bars](#) ↗

HEADERS & ANNOTATIONS

Function	Key arguments
gt_title_header()	<code>title</code> , <code>subtitle</code> , <code>kicker</code> , <code>date</code> , <code>one * _style</code> list each
gt_column_subheaders()	per column list (<code>heading</code> , <code>subtitle</code>) in ...; <code>subtitle_color = "#808080"</code>
gt_538_caption()	<code>top_caption</code> over a rule, <code>bottom_caption</code> , <code>align = "right"</code>
gt_marginalia()	a column as notes: <code>width = 220</code> , <code>italic = TRUE</code> , <code>rule = TRUE</code>
gt_social_tag()	handles + icons: <code>accounts</code> , <code>caption</code> , <code>stack = FALSE</code> , <code>separator = " "</code>
gt_watermark()	<code>text</code> or <code>image</code> , <code>opacity = 0.06</code> , <code>size = "60%"</code> , <code>angle</code>
<pre>gt(nfc) > gt_title_header("NFC North", kicker = "2025 SEASON", subtitle = "Final standings") > gt_column_subheaders(wins = list(heading = "W", subtitle = "2025")) > gt_social_tag(c(x = "@SportsDataverse", gh = "sportsdataverse"))</pre>	

`accounts` names map to Font Awesome brand icons, with aliases (`x` / `twitter`, `ig`, `bsky`, `gh`). A Date in `date` prints as "July 21, 2026".

Cookbook: [Headers, legends, captions](#) ↗

SNAKE A LONG TABLE

<pre>nba <- subset(sdv_example_standings, league == "nba", c(team, wins, losses)) gt(nba) > gt_snake(n_cols = 3)</pre>

Thirty rows become three blocks of ten, each with its own labels. Body-cell styling applied before `gt_snake()` carries through; formats and content transforms (such as `gt_sdv_logos()`) go after it. `gt_snake_align()` reshapes a precomputed mask or colour frame to match.

LEARN MORE

GALLERY ↗ every chart and table in one place

Start: [Getting started](#) · **All:** [articles](#) · [reference](#)

Per league: [NFL](#) · [CFB](#) · [NBA](#) · [WNBA](#) · [MLB](#) · [NHL](#) · [MBB](#) · [WBB](#) · [Soccer](#)

Recipes: [NFL EPA](#) · [CFB table](#) · [WBB standings](#) · [head-to-head card](#)

Leaderboards: [dashboards with Quarto](#) · **Automation:** [social graphics](#)

LAYOUT

Function	Key arguments
gt_grid()	tables in a grid: <code>tables</code> , <code>ncol = 2</code> , <code>labels</code> , <code>title</code> , <code>caption</code> , <code>gap = 24</code> , <code>align = "top"</code> , <code>file</code>
gt_stack_tables()	one above another: <code>gap = 16</code> , <code>align = "center"</code> , <code>title</code> , <code>file</code>
gt_snake()	long table in blocks: <code>n_cols = 2</code> or <code>rows_per_col</code> , <code>gap = 20</code> , <code>fill = ""</code>
gt_snake_align()	reshape a parallel frame (a mask for <code>gt_highlight_cells()</code>) to match
<pre>gt_grid(list(afc_tbl, nfc_tbl), ncol = 2, title = "2025 standings", file = "standings.png")</pre>	

Grids and stacks share `subtitle`, `source_note`, `caption_rule`, the `*_style` lists, and the `save` arguments `bg = "white"`, `whitespace = 50`, `zoom = 2`.

Cookbook: [Faceted tables](#) ↗ · [Schedule matrix](#) ↗

SAVE AS AN IMAGE

Function	Key arguments
gt_save_crop()	PNG / JPG, trimmed and padded: <code>file = NULL</code> , <code>bg = "white"</code> , <code>whitespace = 50</code> , <code>zoom = 2</code> , <code>expand = 5</code> , <code>width</code>
gt_social_crop()	pad to <code>aspect_ratio = "1:1"</code> , <code>"16:9"</code> , <code>"4:5"</code> or a number; <code>whitespace = 60</code> , <code>gravity = "center"</code>
gt_save_batch()	one image per group: <code>data</code> , <code>group</code> , <code>fn</code> , <code>file</code> , <code>dir</code> , <code>match_width = TRUE</code>
<pre>tbl > gt_save_crop("table.png", whitespace = 40, width = 1200) # one image per conference nfl <- subset(sdv_example_standings, league == "nfl") build <- function(df, group) gt(df) > tab_header(title = group) gt_save_batch(nfl, conference, build, "nfl-{group}.png", dir = "out")</pre>	

Saving runs headless Chrome through `webshot2` (Chrome gets 60 s to start). `file = NULL` returns the PNG bytes. `dir` has no default, so a batch never lands in the working directory unasked.

Cookbook: [Saving and posting](#) ↗ · [Social posting](#) ↗

REACTABLE

REACTABLE TABLES

Helper	Arguments (defaults)
<code>reactable_sdv_logos()</code>	<code>league</code> , <code>variant = "default"</code> ("dark", a named variant), <code>height = 30</code> , <code>default_img = NULL</code>
<code>reactable_sdv_wordmarks()</code>	the same
<code>reactable_sdv_headshots()</code>	<code>league</code> , <code>height = 40</code> , <code>default_img</code> , <code>id_system = "espn"</code>
<code>reactable_sdv_team_color_bar()</code>	<code>data</code> , <code>team_col</code> , <code>league</code> , <code>which = "primary"</code> , <code>max_value = NULL</code> , <code>na_color = "grey70"</code>
<code>reactable_sdv_team_color_bg()</code>	as the bar, with <code>alpha = 0.15</code> in place of <code>max_value</code>
<code>reactable_sdv_cols_label()</code>	<code>.data</code> , ... (to every <code>colDef()</code>), <code>league</code> , <code>variant</code> , <code>height = 30</code>

```
library(reactable)
reactable(df, columns = list(
  team = colDef(html = TRUE,
    cell = reactable_sdv_logos("nfl")),
  wins = colDef(style =
    reactable_sdv_team_color_bar(
      df, "team", league = "nfl"))))
# columns named by team key -> logo headers
reactable(h2h,
  columns = reactable_sdv_cols_label(
    h2h, league = "nba"))
```

All but headshots also take `season`, `id_system`, `strict`. The first three are `cell` renderers (`colDef(html = TRUE)`); unresolved values show as text, or `default_img`; a variant that fails to load swaps to the default image. The colour helpers are `style` functions: the bar's length is the value over `max_value` (the column maximum by default). `reactable_sdv_cols_label()` leaves out columns it cannot resolve, so they keep their plain names.

Cookbook: [reactable integration](#) ↗

SIBLINGS

Python **sdvplot** (PyPI) and JavaScript **@sportsdataverse/sdvplot** + **sdvtables** (npm) share `sdvplotR`'s team index, logo archive and argument names; images render through **ggpath**.



sdvplotR :: CHEAT SHEET

Team logos, wordmarks, player headshots and team colours for **28 leagues** (pro, college, minor and junior hockey, spring football, soccer, cricket) in **ggplot2** plots, **gt** and **reactable** tables, and **plotly**, **ggiraph** and **leaflet**, behind one `league` argument. Built on ggpath, after the approach of nflplotR, cfbplotR, nbaplotR and mlbplotR. Page 1: ggplot2 marks, scales and axes. Page 2: teams, interactive, surfaces, coordinates. Page 3: gt images, themes, colour. Page 4: gt layout, saving, reactable.

sdvplotR v0.1.0

R ≥ 4.1 · ggplot2 ≥ 4.0 · ggpath ≥ 1.1

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GGPLOT2 · MARKS & SCALES

INSTALL & LOAD

```
install.packages("sdvplotR")
# dev: sportsdataverse.r-universe.dev
pak::pak("sportsdataverse/sdvplotR")
library(sdvplotR); library(ggplot2)
```

Every team helper takes `league`, a key from `sdv_leagues()` (`"nfl"` `"nba"` `"cfb"` `"nhl"` `"ahl"` `"soccer"` ..., page 2). Teams: any abbreviation, name or id `sdv_resolve()` reads (ESPN, nflverse, nba_api, MLB Stats, CFBD, ...), or a conference (`"SEC"` , `"BIG"` , `"AFC"`).

Cookbook: [Getting started ↗](#)

LOGOS, WORDMARKS & HEADSHOTS

Geom	Required aes
<code>geom_sdv_logos()</code>	<code>x</code> <code>y</code> team
<code>geom_sdv_wordmarks()</code>	<code>x</code> <code>y</code> team (NFL, MLB, MiLB)
<code>geom_sdv_headshots()</code>	<code>x</code> <code>y</code> player_id
<code>geom_from_path()</code>	<code>x</code> <code>y</code> path (any image)

Optional aes and defaults: `alpha` , `colour` (`"b/w"` greyscales), `angle` = 0, `hjust` = 0.5, `vjust` = 0.5, `width` = 1, `height` = 1 (npc; a typical logo is `width` = 0.075). Logos and wordmarks also take `season` (an aes or a constant).

```
df <- data.frame(
  team = sdv_teams("nba")$abbr,
  a = rnorm(30), b = rnorm(30))
ggplot(df, aes(a, b)) +
  geom_mean_lines(aes(x0 = a, y0 = b)) +
  geom_sdv_logos(aes(team = team),
    league = "nba", width = 0.07)
```

HIGHLIGHT A GROUP

```
nfl <- subset(sdv_example_standings,
  league == "nfl")
west <- c("DEN", "KC", "LAC", "LV")
ggplot(nfl, aes(points_for,
  points_against)) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, !team %in% west),
    league = "nfl", width = 0.05,
    alpha = 0.25) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, team %in% west),
    league = "nfl", width = 0.1)
```

Rows draw in order, so the last layer and row end on top. `colour` = `"b/w"` greys logos instead of fading them, which reads better on a dark background.

Cookbook: [Logos, wordmarks, headshots ↗](#)

GEOM ARGUMENTS

```
geom_sdv_logos(mapping = NULL,
  data = NULL, stat = "identity",
  position = "identity", ..., league,
  id_system = "auto", strict = FALSE,
  na.rm = FALSE, show.legend = FALSE,
  inherit.aes = TRUE)
geom_sdv_headshots(..., league,
  id_system = "espn")
```

`geom_sdv_wordmarks()` takes the logo arguments; headshots take no `season` or `strict`. Headshot `id_system`: `"espn"` = ESPN athlete ids in every league; `"gsis"` = nflverse NFL ids; `"league"` = NBA / WNBA Stats PERSON_ID, MLBAM, NHL API.

The ggproto objects `GeomSDVLogo`, `GeomSDVwordmark`, `GeomSDVheadshot` and ggpath's `GeomFromPath`, `GeomReflines` are exported for extensions.

ERA LOGOS

```
old <- data.frame(x = 1:4, y = 1,
  team = c("QUE", "HFD", "ATL", "WPG"),
  season = c(1990, 1995, 2005, 2015))
ggplot(old, aes(x, y)) +
  geom_sdv_logos(aes(team = team,
    season = season), league = "nhl",
  width = 0.2) + theme_void()
```

`season`: the mark worn that year (ending year for split seasons); it also picks who held a reused code. NHL: every identity since 1917-18 (`QUE` `HFD`); AHL, ECHL, OHL, QMJHL, WHL, USHL, PWHL, PHF, UFL, XFL, AAF: dated marks; NFL / WNBA: relocated identities (`STL` `SD` `OAK`; `SA` `TUL`). Elsewhere today's mark. NHL marks are SVG: needs `rsvg`.

REFERENCE LINES

```
p + geom_mean_lines(aes(x0 = a, y0 = b))
p + geom_median_lines(aes(x0 = a),
  colour = "grey40", linetype = 1)
```

From ggpath. Optional aes `x0`, `y0`; defaults `colour` = `"red"`, `linewidth` = 0.5, `linetype` = 2 (dashed), `alpha` = NA.

ANY IMAGE: GEOM_FROM_PATH

```
df$url <- sdv_logo_url(df$team, "nba",
  variant = "dark")
ggplot(df, aes(a, b)) +
  geom_from_path(aes(path = url),
    width = 0.07)
```

`path` is a URL or a local file; same optional aes as the team geoms. Conferences resolve by key or other name: `sdv_logo_url(c("SEC", "BIG"), "cfb")`; a team's code wins (`cfb` "MAC" is Macalester).

COLOUR & FILL SCALES

```
scale_color_sdv(league, which = "primary",
  values = NULL, ..., aesthetics = "colour",
  breaks = waiver(), na.value = "grey50",
  guide = NULL, alpha = NA)
scale_fill_sdv(...) # aesthetics = "fill"
```

Maps team keys to official colours; which = `"secondary"` for the second colour, `alpha` fades them (`scales::alpha()`), `values` replaces them, ... goes to `scale_colour_manual()`. `scale_colour_sdv()` is an alias.

```
ggplot(df, aes(team, b)) +
  geom_col(aes(colour = team, fill = team)) +
  scale_color_sdv("nba", "secondary") +
  scale_fill_sdv("nba", alpha = 0.4)
```



NFL primary / secondary, as `sdv_team_colors("nfl", which = "all")` returns them.

Cookbook: [Team colours and themes ↗](#)

TITLES

```
p + ggtitle_image(title_image = "SEA",
  title = "Seattle Seahawks Analysis",
  image_height = 20, image_side = "left",
  league = "nfl") +
  theme_title_image(size = 20, hjust = 0.5)
```

`title_image`: a team key (logo resolved) or any image URL. `image_height` = 15 px by default; `image_side` = `"left"` or `"right"`; subtitle. `theme_title_image(...)` is required and passes ... to `ggtext::element_markdown()`.

LOGO AT EACH BAR'S END

```
nba <- subset(sdv_example_standings,
  league == "nba")
ggplot(nba, aes(wins,
  reorder(team, wins))) +
  geom_col(aes(fill = team)) +
  geom_sdv_logos(aes(x = wins + 3,
    team = team), league = "nba",
  height = 0.028) +
  scale_fill_sdv("nba")
```

`height` sizes the logo here: each row is about a thirtieth of the panel.

LOGO AXES

Route	Calls
theme	<code>element_sdv_logo()</code> · <code>element_sdv</code>
element	<code>_wordmark()</code> · <code>element_sdv_headshot()</code>
scale+ theme	<code>scale_x_sdv()</code> / <code>scale_y_sdv()</code> · <code>scale_x_sdv_headshots()</code> + <code>theme_x_sdv()</code> / <code>theme_y_sdv()</code>

```
# element route (size in cm)
ggplot(df, aes(team, b)) + geom_col() +
  theme(axis.text.x =
    element_sdv_logo("nba", size = 1))
```

`element_sdv_logo(league, alpha = NULL, colour = NA, hjust = NULL, vjust = NULL, size = 0.5, season, id_system)`; `_wordmark()` the same; `_headshot()` takes `id_system`, no `season`.

```
# scale route (size in px; needs ggtext)
ggplot(df, aes(b, team)) + geom_col() +
  scale_y_sdv(league = "nba", size = 12) +
  theme_y_sdv()
```

`scale_x_sdv(..., league, expand, guide, position = "bottom", size = 12, season, id_system)`; `scale_y_sdv()` has position = `"left"`. Size is the height on x, the width on y. `scale_x_sdv_headshots()` (`size` = 20) and `scale_y_sdv_headshots()` (`size` = 30) take `id_system`. `theme_x_sdv()` / `theme_y_sdv()` render those labels as markdown.

Cookbook: [Logo axes and titles ↗](#)

HEADSHOTS ON A SCATTER

```
ggplot(scorers, aes(ppg, ts)) +
  geom_sdv_headshots(aes(player_id =
    athlete_id), league = "nba",
  height = 0.12)
```

ESPN ids: the `athlete_id` of hoopR, wehoop, cfbfastR rosters. NFL `gsis` ids (`id_system` = `"gsis"`) go through a headshot map built from nflverse rosters (1999 on); `sdv_clear_cache()` re-reads it.

IMAGES IN THEME ELEMENTS

```
p + theme(plot.background =
  element_sdv_raster("court.png"))
```

`element_sdv_raster(image_path, x, y, width, height, just = "centre", hjust = 0.5, vjust = 0.5, interpolate = TRUE, ...)`: one image in a rectangle; wraps ggpath's `element_raster()` (also exported).

`element_path(alpha = 1, colour = NA, hjust, vjust, angle = 0, size = unit(0.5, "cm"))`: axis text that already holds image paths or URLs.



sdvplotR :: TEAMS, INTERACTIVE & SURFACES

The shared SportsDataverse team index behind every helper: **28 leagues**, any team identifier resolved to one canonical `team_id`, colours, palettes and archived logo URLs. Plus logos on **plotly**, **ggiraph** and **leaflet**, premade tier plots, team-coloured playing surfaces drawn with **sportyR**, and converters that put shot and event coordinates from stats.nba.com, Euroleague and ten soccer providers on one frame.

sdvplotR v0.1.0

sportyR for surfaces · rsvg for SVG logos

GALLERY ↗

COOKBOOKS ↗

TEAM DATA · LEAGUES, KEYS, COLOURS

LEAGUES & TEAMS

```
sdv_leagues() # the 28 league keys
sdv_teams("nfl")
sdv_teams("cfb", include_conferences = TRUE)
```

Group	sdv_teams()	columns
ids, names	league team_id abbr name short_name location	
grouping	program conference_id conference	
colours	color_primary color_secondary color_source	

`team_id` is the canonical key (the ESPN id where ESPN has one). Conference rows ("SEC", "AFC") draw logos only.

RESOLVE ANY IDENTIFIER

```
sdv_resolve(c("GSW", "Golden State Warriors",
"XYZ"), "nba")
#> "9" "9" NA + one warning
sdv_resolve(c("OAK", "SD", "STL"), "nfl")
#> "13" "24" "14" LV, LAC, LA
sdv_suggest("Kansas Cty Chiefs", "nfl")
```

`sdv_resolve(x, league, season = NULL, id_system = "auto", strict = FALSE)` reads abbreviations, names and ids from ESPN, nflverse, MLB Stats, nba_api, NHL, HockeyTech, NCAA, PFF, CFBD, Sports Reference, FanGraphs and more. `id_system` names one ("nhl_id" only explicitly). `season` picks who held a reused code (ending year for split seasons). Unresolved: NA and one warning (`strict = TRUE` errors). `sdv_suggest(x, league, n = 5)` lists candidates, never picks.

Every helper that takes a team resolves it this way, with `league` and `id_system`. `strict`: the logo-URL, colour, tier and mark-drawing helpers; not `sdv_marks()`, `element_sdv_*`(), `scale_x/y_sdv()`, `sdv_colorway()`, `sdv_team_factor()`.

TEAM FACTOR

```
sdv_team_factor(c("KC", "BUF", "invalid"),
league = "nfl")
#> KC BUF <NA> Levels: BUF KC
```

Levels ordered by abbreviation; unresolved values become NA, so scales drop them.

COLOURS & PALETTES

```
sdv_team_colors("nfl", "KC")
sdv_team_colors("nfl", c("KC", "BUF"),
which = "all")
sdv_team_colors("nba", which = "all")
sdv_color_palette("nba",
teams = c("BOS", "NY"))
```

`sdv_team_colors(league, team = NULL, which = "primary", season, id_system, strict, include_conferences = FALSE)`: a named hex vector (NA where a team does not resolve); `which = "all"` gives "primary, secondary" strings, or with `team = NULL` a dataframe `team_id abbr name primary secondary`. `sdv_color_palette(league, teams = NULL, which = "primary", ...)`: named colours for `scale_*_manual()`.

IMAGE URLS & MARKS

```
sdv_logo_url(c("KC", "Buffalo Bills",
"OAK"), "nfl")
sdv_logo_url("NYJ", "nfl",
variant = "scoreboard")
sdv_logo_url("QUE", "nhl", season = 1990)
sdv_logo_url("KC", "nfl",
mark_type = "wordmark")
sdv_marks("KC", "nfl") # every mark
sdv_headshot_url("2544", "nba",
id_system = "league")
```

`sdv_logo_url(team, league, season = NULL, variant = "default", mark_type = "logo", id_system, strict)`: the SportsDataverse archive's immutable copy. `variant`: "default", "dark", or a name from `sdv_marks()` ("scoreboard", NFL "grayscale", MLB "cap_on_light", ...); a team without it gets its closest mark of the same polarity, an unknown name errors. `mark_type = "wordmark"`: NFL, MLB (and MiLB).

`sdv_headshot_url(player_id, league, id_system = "espn")`: ESPN headshots for nfl nba wnba mlb nhl cfb mbb wbb; "gsis" and "league" as on page 1. NA for a missing or malformed id.

CACHE & VERSIONS

Index and logo manifest come from the SportsDataverse CDN, cached for a day; a copy ships in the package (`options(sdvplotR.offline = TRUE)`). `sdv_versions()`: what is loaded, from where. `sdv_clearn_cache()`: drop index, manifest, NFL headshot map and ggpath images.

COVERAGE

Leagues	Teams
nfl nba mlb nhl wnba	32 · 30 · 30 · 32 · 15
nbagl mlb pwhl phf	32 · 286 · 12 · 8
ufl xfl usfl aaf	11 · 11 · 9 · 8
ahl echl ohl whl qmjhl ushl	61 · 59 · 27 · 23 · 30 · 17
cfb mbb wbb	687 · 366 · 364
ncaa_baseball softball mhockey whockey	414 · 415 · 110 · 48
soccer · cricket	2,631 · 111

Team rows in the bundled index 0aeef853ec2b; the live index can differ.

PREMADE PLOT: TEAM TIERS

```
tiers <- data.frame(
tier_no = c(1, 1, 2, 2, 3),
team = c("KC", "DET", "BAL", "PHI", "BUF"))
sdv_team_tiers(tiers, league = "nfl",
theme = "light", no_line_below_tier = 2)
```

`sdv_team_tiers(data, league, title, subtitle, caption = NULL, tier_desc, presort = FALSE, alpha = 0.8, width = 0.075, no_line_below_tier = NULL, devel = FALSE, theme = "dark", variant = "auto", season, id_system, strict)`. data needs `tier_no` and team (optional `tier_rank`). Title "NFL Team Tiers" from `league`; NULL drops one. `devel = TRUE` plots keys, not logos; "auto": dark marks on the dark theme.

Cookbook: [College hoops tiers](#) ↗

INTERACTIVE: GGIRAPH, PLOTLY, LEAFLET

```
p <- ggplot(d, aes(x, y)) +
geom_sdv_logos_interactive(aes(team = team,
tooltip = team, data_id = team),
league = "nfl", width = 0.1)
ggiraph::girafe(ggobj = p)
plotly::plot_ly(d, x = ~x, y = ~y) |>
add_sdv_logos(d$x, d$y, d$team, "nfl") |>
sdv_colorway("nfl", d$team)
leaflet::leaflet(d) |> leaflet::addTiles() |>
addSdvLogoMarkers(~lon, ~lat, ~team,
league = "nfl")
```

`geom_sdv_{logos,wordmarks,headshots}_interactive()`: images plus an invisible hover point (`hover_size = 8`). `plotly`: `add_sdv_logos(p, x, y, team, league, size = 0.08, ...)`, `add_sdv_wordmarks()`, `add_sdv_headshots()`, `sdv_axis_logos(p, teams, league, axis)` tick logos, `sdv_colorway()`; `ggplotly()` drops them. `addSdvLogoMarkers(..., size = 32)`: no logo keeps the default marker.

Cookbook: [Interactive plots](#) ↗

PLAYING SURFACES

```
sdv_surface("nba", "BOS",
center_logo = TRUE) +
geom_point(aes(x, y), data = shots)
```

league	sportyR surface
nfl aaf ufl usfl xfl cfb	geom_football("nfl" / "ncaa")
nba nbagl wnba mbb wbb fiba	geom_basketball(); fiba 28 × 15 m
mlb mlb ncaa_baseball	geom_baseball()
nhl ahl echl ohl qmjhl ushl pwhl phf ncaa_mhockey ncaa_whockey	geom_hockey(); whl on the NHL rink
soccer	geom_soccer("fifa"), 105 × 68 m

`sdv_surface(league, team = NULL, season = NULL, center_logo = FALSE, ...)` (not cricket, ncaa_softball). A team paints the paint and apron, end zones, or centre line, circle and boards; baseball stays regulation, soccer gets a centre logo only. Stylized, not real floors. Origin at centre, in feet (yards for football, metres for soccer and FIBA); ... goes to `sportyR` (`rotation`, `xlims`, `color_updates`).

Cookbook: [Surfaces and shot charts](#) ↗

SHOT & EVENT COORDINATES

```
# stats.nba.com legacy frame -> feet
shots |> sdv_court_coords()
# Euroleague (cm, hoop origin) -> FIBA m
pts |> sdv_court_coords(
x_column = "coord_x", y_column = "coord_y",
provider = "euroleague")
# soccer events -> 105 x 68 m pitch
ev |> sdv_pitch_coords("statsbomb")
```

`sdv_court_coords(data, x_column = "x_legacy", y_column = "y_legacy", provider = "nba")` adds `court_x` `court_y`: basket at -41.75 ft (-12.425 m for Euroleague), half court at 0; `court_y` -25 to 25 ft, negative = shooter's left. From hoopR's `nba_shotchartdetail()`: `sdv_court_coords(res$Shot_Chart_Detail, "LOC_X", "LOC_Y")`.

`sdv_pitch_coords(data, provider, x_column = NULL, y_column = NULL, flip = NULL, pitch_length = NULL, pitch_width = NULL)` adds `pitch_x` `pitch_y` in metres from the centre spot (`pitch_x` -52.5 = defended goal line). `flip`: TRUE / FALSE or a logical column, as `flip = "away"`, turns rows half a turn. `pitch_length` / `pitch_width`: tracking only. Providers: opta (statsperform) wyscout statsbomb uefa impect espn, tracking tracab skillcorner secondspectrum metrica.

Cookbook: [NHL shot map](#) ↗ · [Soccer](#) ↗



sdvplotR :: gt TABLES

Team images and colours for **gt**, plus the **gtUtils** toolkit (by Andrew Weatherman): 20 themes, legends, colouring, annotations, layouts and image export. Every helper takes a `gt_tbl` as its first argument (`gt_object`) and returns one, so they chain with `|>`. Examples run on `sdv_example_standings` (real NFL and NBA final standings). Page 3: images, themes, colour, legends, focus. Page 4: format, layout, saving, reactable.

sdvplotR v0.1.0

gt ≥ 0.10 · paletteer optional

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GT · IMAGES, THEMES, COLOUR

TEAM IMAGES IN CELLS

```
gt_sdv_logos(gt_object, columns, league,
  height = 30, locations = NULL,
  include_name = FALSE, season = NULL,
  id_system = "auto", strict = FALSE)
gt_sdv_wordmarks(gt_object, columns,
  league, height = 30, locations = NULL,
  season = NULL, id_system, strict)
gt_sdv_headshots(gt_object, columns,
  league, height = 30, locations = NULL,
  id_system = "espn")
gt_sdv_cols_label(gt_object,
  columns = everything(), league,
  height = 30, mark_type = "logo",
  id_system, season, strict)
```

`height` in px. `locations` (`cells_body()`, `cells_stub()`, `cells_column_labels()`, `cells_row_groups()`) replaces `columns.include_name = TRUE` keeps the text after the logo; `season` shows one season's marks in every cell. Wordmarks: NFL, MLB (and MiLB). `gt_sdv_cols_label()` turns column *labels* into images: `mark_type = "logo"` "wordmark" "headshot".

STACKED NAME IN TEAM COLOUR

```
gt_merge_stack_team_color(gt_object,
  col1, col2, team_color, league,
  font_size_top = 14,
  font_size_bottom = 12, color = "black",
  background = NULL, season, id_system)
```

`col1` on top in bold caps, `col2` below in the team's colour; a primary under 4.5:1 contrast gives way to the secondary or is darkened. Apply a theme first: `background = NULL` reads it.

```
x <- subset(sdv_example_standings,
  league == "nfl",
  c(team, team_name, division))
gt(x) |>
  gt_merge_stack_team_color(team_name,
    division, team, league = "nfl")
```

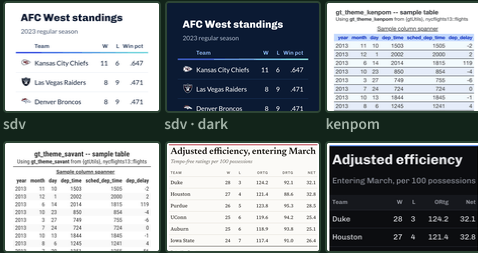
EXAMPLE

```
library(gt)
nfc <- subset(sdv_example_standings,
  division == "NFC North",
  c(team, team_name, wins, losses,
    last_season_wins))
gt(nfc) |>
  gt_sdv_logos(columns = team,
    league = "nfl", height = 28) |>
  gt_theme_sdv()
```

Cookbook: [Team tables with the gt toolkit ↗](#)

THEMES

```
gt_theme_sdv(gt, style = "Light")
gt_theme_sdv_team(gt, team = "KC",
  league = "nfl")
```



savant broadsheet midnight

`gt_theme_sdv()` : Chivo titles and labels, a Lato body in tabular figures, and the SDV gradient (`#3346F0 → #7FE6DC`) as one horizon line under the labels; `style = "dark"` sets it on the SportsDataverse navy.

`gt_theme_sdv_team()` : the sdv table in one team's colours (title block primary, horizon line secondary); `team = NULL` keeps SportsDataverse colours; takes `season`, `id_system`. Every `gt_theme<name>(gt_object, ..., density)` takes `density = "comfortable"` "compact" "social" (default comfortable; compact for almanac, scoreboard, terminal) and passes ... to `gt::tab_options()`, applied last.

Option	Themes (default)
accent	almanac <code>"#8C2F1E"</code> · booktabs, swiss, tuftes <code>"#111111"</code> · broadsheet <code>"#A6081A"</code> · brutalist <code>"#FF3B00"</code> · midnight <code>"#5B8DEF"</code> · scoreboard <code>"#0E1621"</code> · terminal <code>"#FFB86C"</code>
style	sdv, sofa <code>"light"</code> / <code>"dark"</code> · tier <code>"dark"</code> / <code>"light"</code>
other	drench color = <code>"#123F5E"</code> · broadsheet paper = "white" or "salmon" · almanac stripe (NA = off)
none	athletic gtutils kenpom ncaa pl savant

All 20: almanac athletic booktabs broadsheet brutalist drench gtutils kenpom midnight ncaa pl savant scoreboard sdv sdv_team sofa swiss terminal tier tuftes (each `gt_theme<name>()`).

```
gt(nfc) |>
  gt_theme_broadsheet(paper = "salmon",
    density = "compact")
gt(nfc) |>
  gt_theme_terminal(accent = "#7EE787")
```

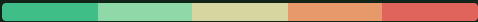
Cookbook: [SportsDataverse table themes ↗](#)

PREVIEW & PALETTES

```
gt_theme_preview(nfc, themes = c(
  "gt_theme_sdv", "gt_theme_kenpom"))
gt_color_ranks(..., palette = pal_midnight)
```

`gt_theme_preview(data, themes = NULL, n = 5, ncol = 3, density = "compact", file = NULL, ...)`: `NULL` shows every theme; themes needs full function names. ... goes to `gt_grid()`.

`pal_midnight`: a rank palette that reads on midnight / terminal.



COLOUR BY VALUE

Function	Key arguments
gt_color_ranks()	fill cells: palette domain reverse, na_color = "white", autocolor_text = TRUE
gt_color_pills()	pill: fill_type = "continuous" / "rank", format_type, suffix, digits
gt_percentile_bar()	bar+marker: domain = c(0, 100), scale = "auto" (0-1 → 0-100), na_label = "-"
gt_color_results()	row colours: result_column = "result", result_type = "wl" / "binary"
gt_row_accent()	edge bar: column, palette, side = "left", hide = TRUE
gt_indicator_boxes()	boxes: indicator_vals = c(0, 1), color_yes, show_text
gt_tiers()	tierlist: levels, colors, tier_column = "tier", style = "dark"

Most take `columns` and `rows` (an expression such as `wins > 10`, or indices). `palette` also takes a paletteer name, e.g. "viridis::mako"; `domain = NULL` uses the data's range.

Cookbook: [Percentile bars and cut lines ↗](#)

TIER LIST

```
gt(seeds) |> # tier + logo URL columns
  gt_tiers(levels = c("Bye", "Wild card"),
    colors = c("#1B7837", "#B2182B")) |>
  gt_legend_discrete()
```

Columns other than `tier_column` render as images (`image_columns` narrows them; `img_height = "55px"`), e.g. a column of `sdv_logo_url()` URLs. Levels may be one named vector, `c(Bye = "#1B7837")`.

Cookbook: [Tier lists ↗](#)

LEGENDS

Function	Key arguments
gt_legend_continuous()	bar: type = "continuous" "steps" "blocks", n_bins = 5, labels ("edges"), title, location = "bottom", width = 200
gt_legend_discrete()	swatches: key_info (label = colour), heading, location = "top", shape, direction
gt(nfc) > gt_color_ranks(wins) > gt_legend_continuous(type = "steps", title = "Wins")	

The continuous legend reads the columns, palette and domain the colouring step recorded; the discrete one reads the key `gt_tiers()` recorded, so `key_info` can be left out.

```
gt(afc_west) |>
  gt_legend_discrete(c("Playoffs" =
    "#CCE7F5", "Missed" = "#F5CCCC"))
```

Cookbook: [Headers, legends, captions ↗](#)

HIGHLIGHT & FOCUS

Function	Key arguments
gt_highlight_cells()	columns, condition (~ .x > 0.7 or a logical mask), fill = "#FFF3B0"
gt_highlight_na()	fill = "#F0F0F0", missing_text, na_strings = "NA"
gt_outliers()	method = "iqr" "sd" "bounds", threshold (1.5/3), side, symbol, note
gt_spotlight()	rows; dims the rest (dim_color = "auto", WCAG AA); accent_color, if_none
gt_bold_rows()	rows, text_color = "black", highlight_color
gt_group_strips()	band row groups: color = "#F5F5F5", start = 2

```
gt(nfc) |>
  gt_spotlight(team == "DET",
    accent_color = "#0076B6") |>
  gt_outliers(wins, method = "sd",
    note = TRUE)
gt(nfc) |>
  gt_highlight_cells(c(wins, losses),
    ~ .x >= 10, bold = TRUE)
```



sdvplotR :: gt LAYOUT & reactable

The rest of the gtUtils toolkit for **gt**: number formats, rules and border bars, headers and captions, multi-table layouts and image export for posting. Then the **reactable** helpers: team logos, wordmarks and headshots in cells and headers, and team-coloured bars and fills. Examples continue with `nfc` (the NFC North rows of `sdv_example_standings`, page 3).

sdvplotR v0.1.0

webshot2 to save · reactable for tables

GALLERY ↗

COOKBOOKS ↗

GT · FORMAT, LAYOUT, SAVE

FORMAT

Function	Key arguments
<code>gt_delta()</code>	change column to - from: <code>column_label = "Change"</code> , <code>percent = FALSE</code> , <code>decimals = 1</code> , <code>arrows = FALSE</code> , <code>after</code>
<code>gt_fmt_rank()</code>	1→1st: <code>superscript = TRUE</code> , <code>suffix_size = "0.7em"</code>
<code>gt_fmt_tally()</code>	W-L(-T) in one cell: <code>separator = "-"</code> , <code>share = FALSE</code> , <code>share_of = 1</code> , <code>share_location = "inline"</code> / <code>"column"</code>
<code>gt_significance()</code>	stars from p_columns: <code>levels = c(0.01, 0.05, 0.1)</code> , <code>symbols</code> , <code>hide_p = TRUE</code>
<code>gt_scale_note()</code>	divide and say so: <code>divisor = 1000</code> ("Figures in thousands."), <code>where = "source_note"</code> "label" "both"
<code>gt_wrap_labels()</code>	break long labels: <code>width = 12</code> , <code>balance = TRUE</code>
<code>gt_set_font()</code>	one family for the table: <code>font_family</code> , <code>from_google_font = TRUE</code> , <code>weight</code> , <code>style</code>
<pre>gt(nfc) > gt_fmt_tally(c(wins, losses)) > gt_delta(from = last_season_wins, to = wins)</pre>	

Cookbook: [Rolling-window wins](#) ↗

BORDERS & RULES

Function	Key arguments
<code>gt_border_grid()</code>	full grid: <code>color = "black"</code> , <code>weight = 1</code> , <code>include_labels = FALSE</code>
<code>gt_borderBars_top()</code> / <code>_bottom()</code>	colour bars: <code>colors</code> , <code>bar_height = 10</code> , <code>img</code> , <code>text</code> , <code>text_size = 18</code>
<code>gt_cutline()</code>	line after rows: <code>after</code> , <code>label</code> , <code>color = "#A6081A"</code> , <code>style = "dashed"</code> , <code>gap</code>
<pre>gt(nfc) > gt_borderBars_top(c("#0076B6", "#B0B7BC")) > gt_cutline(after = c(1, 3), label = c("Champion", NA))</pre>	

Cookbook: [Border bars](#) ↗

HEADERS & ANNOTATIONS

Function	Key arguments
<code>gt_title_header()</code>	<code>title</code> , <code>subtitle</code> , <code>kicker</code> , <code>date</code> , one <code>*_style</code> list each
<code>gt_column_subheaders()</code>	per column list(heading, subtitle) in ...; <code>subtitle_color = "#808080"</code>
<code>gt_538_caption()</code>	<code>top_caption</code> over a rule, <code>bottom_caption</code> , <code>align = "right"</code>
<code>gt_marginalia()</code>	a column as notes: <code>width = 220</code> , <code>italic = TRUE</code> , <code>rule = TRUE</code>
<code>gt_social_tag()</code>	handles + icons: <code>accounts</code> , <code>caption</code> , <code>stack = FALSE</code> , <code>separator = " "</code>
<code>gt_watermark()</code>	<code>text</code> or <code>image</code> , <code>opacity = 0.06</code> , <code>size = "60%"</code> , <code>angle</code>

```
gt(nfc) |>
  gt_title_header("NFC North",
    kicker = "2025 SEASON",
    subtitle = "Final standings") |>
  gt_column_subheaders(wins = list(
    heading = "W", subtitle = "2025")) |>
  gt_social_tag(c(x = "@SportsDataverse",
    gh = "sportsdataverse"))
```

`accounts` names map to Font Awesome brand icons, with aliases (x / twitter, ig, bsky, gh). A Date in date prints as "July 21, 2026".

Cookbook: [Headers, legends, captions](#) ↗

SNAKE A LONG TABLE

```
nba <- subset(sdv_example_standings,
  league == "nba", c(team, wins, losses))
gt(nba) |> gt_snake(n_cols = 3)
```

Thirty rows become three blocks of ten, each with its own labels. Body-cell styling applied before `gt_snake()` carries through; formats and content transforms (such as `gt_sdv_logos()`) go after it. `gt_snake_align()` reshapes a precomputed mask or colour frame to match.

LEARN MORE

GALLERY ↗ every chart and table in one place

Start: Getting started · **All:** articles · reference

Per league: NFL · CFB · NBA · WNBA · MLB · NHL · MBB · WBB · Soccer

Recipes: NFL EPA · CFB table · WBB standings · head-to-head card

Leaderboards: dashboards with Quarto · **Automation:** social graphics

LAYOUT

Function	Key arguments
<code>gt_grid()</code>	tables in a grid: <code>tables</code> , <code>ncol = 2</code> , <code>labels</code> , <code>title</code> , <code>caption</code> , <code>gap = 24</code> , <code>align = "top"</code> , <code>file</code>
<code>gt_stack_tables()</code>	one above another: <code>gap = 16</code> , <code>align = "center"</code> , <code>title</code> , <code>file</code>
<code>gt_snake()</code>	long table in blocks: <code>n_cols = 2</code> or <code>rows_per_col</code> , <code>gap = 20</code> , <code>fill = ""</code>
<code>gt_snake_align()</code>	reshape a parallel frame (a mask for <code>gt_highlight_cells()</code>) to match
<pre>gt_grid(list(afc_tbl, nfc_tbl), ncol = 2, title = "2025 standings", file = "standings.png")</pre>	

Grids and stacks share `subtitle`, `source_note`, `caption_rule`, the `*_style` lists, and the `save` arguments `bg = "white"`, `whitespace = 50`, `zoom = 2`.

Cookbook: [Faceted tables](#) ↗ · [Schedule matrix](#) ↗

SAVE AS AN IMAGE

Function	Key arguments
<code>gt_save_crop()</code>	PNG/JPG, trimmed and padded: <code>file = NULL</code> , <code>bg = "white"</code> , <code>whitespace = 50</code> , <code>zoom = 2</code> , <code>expand = 5</code> , <code>width</code>
<code>gt_social_crop()</code>	pad to aspect_ratio = "1:1", "16:9", "4:5" or number; <code>whitespace = 60</code> , <code>gravity = "center"</code>
<code>gt_save_batch()</code>	one image per group: <code>data</code> , <code>group</code> , <code>fn</code> , <code>file</code> , <code>dir</code> , <code>match_width = TRUE</code>

```
tbl |> gt_save_crop("table.png",
  whitespace = 40,
  width = 1200)

# one image per conference
nfl <- subset(sdv_example_standings,
  league == "nfl")
build <- function(df, group)
  gt(df) |> tab_header(title = group)
gt_save_batch(nfl, conference, build,
  "nfl-{group}.png", dir = "out")
```

Saving runs headless Chrome through `webshot2` (Chrome gets 60 s to start). `file = NULL` returns the PNG bytes. `dir` has no default, so a batch never lands in the working directory unasked.

Cookbook: [Saving and posting](#) ↗ · [Social posting](#) ↗

REACTABLE

REACTABLE TABLES

Helper	Arguments (defaults)
<code>reactable_sdv_logos()</code>	<code>league</code> , <code>variant = "default"</code> ("dark", a named variant), <code>height = 30</code> , <code>default_img = NULL</code>
<code>reactable_sdv_wordmarks()</code>	the same
<code>reactable_sdv_headshots()</code>	<code>league</code> , <code>height = 40</code> , <code>default_img</code> , <code>id_system = "espn"</code>
<code>reactable_sdv_team_color_bar()</code>	<code>data</code> , <code>team_col</code> , <code>league</code> , <code>which = "primary"</code> , <code>max_value = NULL</code> , <code>na_color = "grey70"</code>
<code>reactable_sdv_team_color_bg()</code>	as the bar, with <code>alpha = 0.15</code> in place of <code>max_value</code>
<code>reactable_sdv_cols_label()</code>	.data, ... (to every <code>colDef()</code>), <code>league</code> , <code>variant</code> , <code>height = 30</code>

```
library(reactable)
reactable(df, columns = list(
  team = colDef(html = TRUE,
    cell = reactable_sdv_logos("nfl")),
  wins = colDef(style =
    reactable_sdv_team_color_bar(
      df, "team", league = "nfl"))))
# columns named by team key -> logo headers
reactable(h2h,
  columns = reactable_sdv_cols_label(
    h2h, league = "nba"))
```

All but headshots also take `season`, `id_system`, `strict`. The first three are `cell` renderers (`colDef(html = TRUE)`); unresolved values show as text, or `default_img`; a variant that fails to load swaps to the default image. The colour helpers are `style` functions: the bar's length is the value over `max_value` (the column maximum by default). `reactable_sdv_cols_label()` leaves out columns it cannot resolve, so they keep their plain names.

Cookbook: [reactable integration](#) ↗

SIBLINGS

Python **sdvplot** (PyPI) and JavaScript **@sportsdataverse/sdvplot** + **sdvtables** (npm) share `sdvplotR`'s team index, logo archive and argument names; images render through **ggpath**.



sdvplotR :: CHEAT SHEET

Team logos, wordmarks, player headshots and team colours for **28 leagues** (pro, college, minor and junior hockey, spring football, soccer, cricket) in **ggplot2** plots, **gt** and **reactable** tables, and **plotly**, **ggiraph** and **leaflet**, behind one `league` argument. Built on **ggpath**, after the approach of **nflplotR**, **cfbplotR**, **nbaplotR** and **mlbplotR**. Page 1: ggplot2 marks, scales and axes. Page 2: teams, interactive, surfaces, coordinates. Page 3: gt images, themes, colour. Page 4: gt layout, saving, reactable.

sdvplotR v0.1.0

R ≥ 4.1 · ggplot2 ≥ 4.0 · ggpath ≥ 1.1

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GGPLOT2 · MARKS & SCALES

INSTALL & LOAD

```
install.packages("sdvplotR")
# dev: sportsdataverse.r-universe.dev
pak::pak("sportsdataverse/sdvplotR")
library(sdvplotR); library(ggplot2)
```

Every team helper takes `league`, a key from `sdv_leagues()` ("nfl" "nba" "cfb" "nhl" "ahl" "soccer" ..., page 2). Teams: any abbreviation, name or id `sdv_resolve()` reads (ESPN, nflverse, nba_api, MLB Stats, CFBD, ...), or a conference ("SEC", "B1G", "AFC").

Cookbook: [Getting started ↗](#)

LOGOS, WORDMARKS & HEADSHOTS

Geom	Required aes
<code>geom_sdv_logos()</code>	<code>x y team</code>
<code>geom_sdv_wordmarks()</code>	<code>x y team (NFL, MLB, MiLB)</code>
<code>geom_sdv_headshots()</code>	<code>x y player_id</code>
<code>geom_from_path()</code>	<code>x y path (any image)</code>

Optional aes and defaults: `alpha`, `colour` ("b/w" greyscales), `angle` = 0, `hjust` = 0.5, `vjust` = 0.5, `width` = 1, `height` = 1 (npc; a typical logo is `width` = 0.075). Logos and wordmarks also take `season` (an aes or a constant).

```
df <- data.frame(
  team = sdv_teams("nba")$abbr,
  a = rnorm(30), b = rnorm(30))
ggplot(df, aes(a, b)) +
  geom_mean_lines(aes(x0 = a, y0 = b)) +
  geom_sdv_logos(aes(team = team),
    league = "nba", width = 0.07)
```

HIGHLIGHT A GROUP

```
nfl <- subset(sdv_example_standings,
  league == "nfl")
west <- c("DEN", "KC", "LAC", "LV")
ggplot(nfl, aes(points_for,
  points_against)) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, !team %in% west),
    league = "nfl", width = 0.05,
    alpha = 0.25) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, team %in% west),
    league = "nfl", width = 0.1)
```

Rows draw in order, so the last layer and row end on top. `colour` = "b/w" greys logos instead of fading them, which reads better on a dark background.

Cookbook: [Logos, wordmarks, headshots ↗](#)

GEOM ARGUMENTS

```
geom_sdv_logos(mapping = NULL,
  data = NULL, stat = "identity",
  position = "identity", ..., league,
  id_system = "auto", strict = FALSE,
  na.rm = FALSE, show.legend = FALSE,
  inherit.aes = TRUE)
geom_sdv_headshots(..., league,
  id_system = "espn")
```

`geom_sdv_wordmarks()` takes the logo arguments; headshots take no `season` or `strict`. Headshot `id_system`: "espn" = ESPN athlete ids in every league; "gsis" = nflverse NFL ids; "league" = NBA / WNBA Stats PERSON_ID, MLBAM, NHL API.

The ggproto objects `GeomSDVLogo`, `GeomSDVwordmark`, `GeomSDVheadshot` and `ggpath`'s `GeomFromPath`, `GeomRefLines` are exported for extensions.

ERA LOGOS

```
old <- data.frame(x = 1:4, y = 1,
  team = c("QUE", "HFD", "ATL", "WPG"),
  season = c(1990, 1995, 2005, 2015))
ggplot(old, aes(x, y)) +
  geom_sdv_logos(aes(team = team,
    season = season), league = "nhl",
    width = 0.2) + theme_void()
```

`season`: the mark worn that year (ending year for split seasons); it also picks who held a reused code. NHL: every identity since 1917-18 (QUE HFD); AHL, ECHL, OHL, QMJHL, WHL, USHL, PWHL, PHF, UFL, AFL, AAF: dated marks; NFL / WNBA: relocated identities (STL SD OAK; SA TUL). Elsewhere today's mark. NHL marks are SVG: needs `rsvg`.

REFERENCE LINES

```
p + geom_mean_lines(aes(x0 = a, y0 = b))
p + geom_median_lines(aes(x0 = a),
  colour = "grey40", linetype = 1)
```

From `ggpath`. Optional aes `x0`, `y0`; defaults `colour` = "red", `linewidth` = 0.5, `linetype` = 2 (dashed), `alpha` = NA.

ANY IMAGE: GEOM_FROM_PATH

```
df$url <- sdv_logo_url(df$team, "nba",
  variant = "dark")
ggplot(df, aes(a, b)) +
  geom_from_path(aes(path = url),
    width = 0.07)
```

`path` is a URL or a local file; same optional aes as the team geoms. Conferences resolve by key or other name: `sdv_logo_url(c("SEC", "B1G", "cfb"))`; a team's code wins (cfb "MAC" is Macalester).

COLOUR & FILL SCALES

```
scale_color_sdv(league, which = "primary",
  values = NULL, ..., aesthetics = "colour",
  breaks = waiver(), na.value = "grey50",
  guide = NULL, alpha = NA)
scale_fill_sdv(...) # aesthetics = "fill"
```

Maps team keys to official colours; `which` = "secondary" for the second colour, `alpha` fades them (`scales::alpha()`), `values` replaces them, ... goes to `scale_colour_manual()`. `scale_colour_sdv()` is an alias.

```
ggplot(df, aes(team, b)) +
  geom_col(aes(colour = team, fill = team)) +
  scale_color_sdv("nba", "secondary") +
  scale_fill_sdv("nba", alpha = 0.4)
```



NFL primary / secondary, as `sdv_team_colors("nfl", which = "all")` returns them.

Cookbook: [Team colours and themes ↗](#)

TITLES

```
p + ggtitle_image(title_image = "SEA",
  title = "Seattle Seahawks Analysis",
  image_height = 20, image_side = "left",
  league = "nfl") +
  theme_title_image(size = 20, hjust = 0.5)
```

`title_image`: a team key (logo resolved) or any image URL. `image_height` = 15 px by default; `image_side` = "left" or "right"; subtitle. `theme_title_image(...)` is required and passes ... to `ggtext::element_markdown()`.

LOGO AT EACH BAR'S END

```
nba <- subset(sdv_example_standings,
  league == "nba")
ggplot(nba, aes(wins,
  reorder(team, wins))) +
  geom_col(aes(fill = team)) +
  geom_sdv_logos(aes(x = wins + 3,
    team = team), league = "nba",
    height = 0.028) +
  scale_fill_sdv("nba")
```

`height` sizes the logo here: each row is about a thirtieth of the panel.

LOGO AXES

Route	Calls
theme	<code>element_sdv_logo()</code> · <code>element_sdv</code>
element	<code>_wordmark()</code> · <code>element_sdv_headshot()</code>
scale+ theme	<code>scale_x_sdv()</code> / <code>scale_y_sdv()</code> · <code>scale_x_sdv_headshots()</code> + <code>theme_x_sdv()</code> / <code>theme_y_sdv()</code>

```
# element route (size in cm)
ggplot(df, aes(team, b)) + geom_col() +
  theme(axis.text.x =
    element_sdv_logo("nba", size = 1))
```

`element_sdv_logo(league, alpha = NULL, colour = NA, hjust = NULL, vjust = NULL, size = 0.5, season, id_system)`; `_wordmark()` the same; `_headshot()` takes `id_system`, no `season`.

```
# scale route (size in px; needs ggtext)
ggplot(df, aes(b, team)) + geom_col() +
  scale_y_sdv(league = "nba", size = 12) +
  theme_y_sdv()
```

`scale_x_sdv(..., league, expand, guide, position = "bottom", size = 12, season, id_system)`; `scale_y_sdv()` has position = "left". Size is the height on x, the width on y. `scale_x_sdv_headshots()` (size = 20) and `scale_y_sdv_headshots()` (size = 30) take `id_system`. `theme_x_sdv()` / `theme_y_sdv()` render those labels as markdown.

Cookbook: [Logo axes and titles ↗](#)

HEADSHOTS ON A SCATTER

```
ggplot(scorers, aes(ppg, ts)) +
  geom_sdv_headshots(aes(player_id =
    athlete_id), league = "nba",
    height = 0.12)
```

ESPN ids: the `athlete_id` of `hoopR`, `wehoop`, `cfbfastR` rosters. NFL gsis ids (`id_system` = "gsis") go through a headshot map built from nflverse rosters (1999 on); `sdv_clear_cache()` re-reads it.

IMAGES IN THEME ELEMENTS

```
p + theme(plot.background =
  element_sdv_raster("court.png"))
```

`element_sdv_raster(image_path, x, y, width, height, just = "centre", hjust = 0.5, vjust = 0.5, interpolate = TRUE, ...)`: one image in a rectangle; wraps `ggpath`'s `element_raster()` (also exported).

`element_path(alpha = 1, colour = NA, hjust, vjust, angle = 0, size = unit(0.5, "cm"))`: axis text that already holds image paths or URLs.



sdvplotR :: TEAMS, INTERACTIVE & SURFACES

The shared SportsDataverse team index behind every helper: **28 leagues**, any team identifier resolved to one canonical `team_id`, colours, palettes and archived logo URLs. Plus logos on **plotly**, **ggiraph** and **leaflet**, premade tier plots, team-coloured playing surfaces drawn with **sportyR**, and converters that put shot and event coordinates from stats.nba.com, Euroleague and ten soccer providers on one frame.

sdvplotR v0.1.0

sportyR for surfaces · rsvg for SVG logos

GALLERY ↗

COOKBOOKS ↗

TEAM DATA · LEAGUES, KEYS, COLOURS

LEAGUES & TEAMS

```
sdv_leagues() # the 28 league keys
sdv_teams("nfl")
sdv_teams("cfb", include_conferences = TRUE)
```

Group	sdv_teams()	columns
ids, names	league team_id abbr name short_name location	
grouping	program conference_id conference	
colours	color_primary color_secondary color_source	

`team_id` is the canonical key (the ESPN id where ESPN has one). Conference rows (`"SEC"`, `"AFC"`) draw logos only.

RESOLVE ANY IDENTIFIER

```
sdv_resolve(c("GSW", "Golden State Warriors",
              "XYZ"), "nba")
#> "9" "9" NA + one warning
sdv_resolve(c("OAK", "SD", "STL"), "nfl")
#> "13" "24" "14" LV, LAC, LA
sdv_suggest("Kansas Cty Chiefs", "nfl")
```

`sdv_resolve(x, league, season = NULL, id_system = "auto", strict = FALSE)` reads abbreviations, names and ids from ESPN, nflverse, MLB Stats, nba_api, NHL, HockeyTech, NCAA, PFF, CFBD, Sports Reference, FanGraphs and more. `id_system` names one (`"nhl_id"` only explicitly). `season` picks who held a reused code (ending year for split seasons). Unresolved: `NA` and one warning (`strict = TRUE` errors). `sdv_suggest(x, league, n = 5)` lists candidates, never picks.

Every helper that takes a team resolves it this way, with `league` and `id_system`. `strict`: the logo-URL, colour, tier and mark-drawing helpers; not `sdv_marks()`, `element_sdv_*()`, `scale_x/y_sdv()`, `sdv_colorway()`, `sdv_team_factor()`.

TEAM FACTOR

```
sdv_team_factor(c("KC", "BUF", "invalid"),
               league = "nfl")
#> KC BUF <NA> Levels: BUF KC
```

Levels ordered by abbreviation; unresolved values become `NA`, so scales drop them.

COLOURS & PALETTES

```
sdv_team_colors("nfl", "KC")
sdv_team_colors("nfl", c("KC", "BUF"),
               which = "all")
sdv_team_colors("nba", which = "all")
sdv_color_palette("nba",
                 teams = c("BOS", "NY"))
```

`sdv_team_colors(league, team = NULL, which = "primary", season, id_system, strict, include_conferences = FALSE)`: a named hex vector (NA where a team does not resolve); `which = "all"` gives "primary, secondary" strings, or with `team = NULL` a dataframe `team_id abbr name primary secondary`. `sdv_color_palette(league, teams = NULL, which = "primary", ...)`: named colours for `scale_*_manual()`.

IMAGE URLS & MARKS

```
sdv_logo_url(c("KC", "Buffalo Bills",
               "OAK"), "nfl")
sdv_logo_url("NYJ", "nfl",
             variant = "scoreboard")
sdv_logo_url("QUE", "nhl", season = 1990)
sdv_logo_url("KC", "nfl",
             mark_type = "wordmark")
sdv_marks("KC", "nfl") # every mark
sdv_headshot_url("2544", "nba",
                 id_system = "league")
```

`sdv_logo_url(team, league, season = NULL, variant = "default", mark_type = "logo", id_system, strict)`: the SportsDataverse archive's immutable copy. `variant`: "default", "dark", or a name from `sdv_marks()` ("scoreboard", NFL "grayscale", MLB "cap_on_light", ...); a team without it gets its closest mark of the same polarity, an unknown name errors. `mark_type = "wordmark"`: NFL, MLB (and MiLB).

`sdv_headshot_url(player_id, league, id_system = "espn")`: ESPN headshots for nfl nba wnba mlb nhl cfb mbb wbb; "gsis" and "league" as on page 1. NA for a missing or malformed id.

CACHE & VERSIONS

Index and logo manifest come from the SportsDataverse CDN, cached for a day; a copy ships in the package (`options(sdvplotR.offline = TRUE)`). `sdv_versions()`: what is loaded, from where. `sdv_clear_cache()`: drop index, manifest, NFL headshot map and ggpath images.

COVERAGE

Leagues	Teams
nfl nba mlb nhl wnba	32 · 30 · 30 · 32 · 15
nbagl milb pwhl phf	32 · 286 · 12 · 8
ufl xfl usfl aaf	11 · 11 · 9 · 8
ahl echl ohl whl qmjhl ushl	61 · 59 · 27 · 23 · 30 · 17
cfb mbb wbb	687 · 366 · 364
ncaa_baseball softball mhockey whockey	414 · 415 · 110 · 48
soccer · cricket	2,631 · 111

Team rows in the bundled index `0aeef853ec2b`; the live index can differ.

PREMADE PLOT: TEAM TIERS

```
tiers <- data.frame(
  tier_no = c(1, 1, 2, 2, 3),
  team = c("KC", "DET", "BAL", "PHI", "BUF"))
sdv_team_tiers(tiers, league = "nfl",
               theme = "light", no_line_below_tier = 2)
```

`sdv_team_tiers(data, league, title, subtitle, caption = NULL, tier_desc, presort = FALSE, alpha = 0.8, width = 0.075, no_line_below_tier = NULL, devel = FALSE, theme = "dark", variant = "auto", season, id_system, strict)`. data needs `tier_no` and `team` (optional `tier_rank`). Title "NFL Team Tiers" from `league`; `NULL` drops one. `devel = TRUE` plots keys, not logos; "auto": dark marks on the dark theme.

Cookbook: [College hoops tiers](#) ↗

INTERACTIVE: GGIRAPH, PLOTLY, LEAFLET

```
p <- ggplot(d, aes(x, y)) +
  geom_sdv_logos_interactive(aes(team = team,
                                tooltip = team, data_id = team),
                           league = "nfl", width = 0.1)
ggiraph::girafe(ggobj = p)
plotly::plot_ly(d, x = ~x, y = ~y) |>
  add_sdv_logos(d$x, d$y, d$team, "nfl") |>
  sdv_colorway("nfl", d$team)
leaflet::leaflet(d) |> leaflet::addTiles() |>
  addSdvLogoMarkers(~lon, ~lat, ~team,
                   league = "nfl")
```

`geom_sdv_{logos,wordmarks,headshots}_interactive()`: images plus an invisible hover point (`hover_size = 8`). `plotly`: `add_sdv_logos(p, x, y, team, league, size = 0.08, ...)`, `add_sdv_wordmarks()`, `add_sdv_headshots()`, `sdv_axis_logos(p, teams, league, axis)` tick logos, `sdv_colorway()`; `ggplotly()` drops them. `addSdvLogoMarkers(..., size = 32)`: no logo keeps the default marker.

Cookbook: [Interactive plots](#) ↗

PLAYING SURFACES

```
sdv_surface("nba", "BOS",
            center_logo = TRUE) +
  geom_point(aes(x, y), data = shots)
```

league	sportyR surface
nfl aaf ufl usfl xfl cfb	geom_football("nfl" / "ncaa")
nba nbagl wnba mbb wbb fiba	geom_basketball(); fiba 28 × 15 m
mlb milb ncaa_baseball	geom_baseball()
nhl ahl echl ohl qmjhl ushl pwhl phf ncaa_mhockey ncaa_whockey	geom_hockey(); whl on the NHL rink
soccer	geom_soccer("fifa"), 105 × 68 m

`sdv_surface(league, team = NULL, season = NULL, center_logo = FALSE, ...)` (not cricket, ncaa_softball). A team paints the paint and apron, end zones, or centre line, circle and boards; baseball stays regulation, soccer gets a centre logo only. Stylized, not real floors. Origin at centre, in feet (yards for football, metres for soccer and FIBA); ... goes to `sportyR` (rotation, xlims, color_updates).

Cookbook: [Surfaces and shot charts](#) ↗

SHOT & EVENT COORDINATES

```
# stats.nba.com legacy frame -> feet
shots |> sdv_court_coords()
# Euroleague (cm, hoop origin) -> FIBA m
pts |> sdv_court_coords(
  x_column = "coord_x", y_column = "coord_y",
  provider = "euroleague")
# soccer events -> 105 x 68 m pitch
ev |> sdv_pitch_coords("statsbomb")
```

`sdv_court_coords(data, x_column = "x_legacy", y_column = "y_legacy", provider = "nba")` adds `court_x` `court_y`: basket at -41.75 ft (-12.425 m for Euroleague), half court at 0; `court_y` -25 to 25 ft, negative = shooter's left. From hoopR's `nba_shotchartdetail()`: `sdv_court_coords(res$Shot_Chart_Detail, "LOC_X", "LOC_Y")`.

`sdv_pitch_coords(data, provider, x_column = NULL, y_column = NULL, flip = NULL, pitch_length = NULL, pitch_width = NULL)` adds `pitch_x` `pitch_y` in metres from the centre spot (`pitch_x` -52.5 = defended goal line). `flip`: TRUE / FALSE or a logical column, as `flip = "away"`, turns rows half a turn. `pitch_length / pitch_width`: tracking only. Providers: `opta` (statsperform) `wyscout` `statsbomb` `uefa` `impect` `espn`, tracking `tracab` `skillcorner` `secondspectrum` `metrca`.

Cookbook: [NHL shot map](#) ↗ · [Soccer](#) ↗



sdvplotR :: gt TABLES

Team images and colours for **gt**, plus the **gtUtils** toolkit (by Andrew Weatherman): 20 themes, legends, colouring, annotations, layouts and image export. Every helper takes a `gt_tbl` as its first argument (`gt_object`) and returns one, so they chain with `|>`. Examples run on `sdv_example_standings` (real NFL and NBA final standings). Page 3: images, themes, colour, legends, focus. Page 4: format, layout, saving, reactable.

sdvplotR v0.1.0

gt ≥ 0.10 · paletteer optional

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GT · IMAGES, THEMES, COLOUR

TEAM IMAGES IN CELLS

```
gt_sdv_logos(gt_object, columns, league,
  height = 30, locations = NULL,
  include_name = FALSE, season = NULL,
  id_system = "auto", strict = FALSE)
gt_sdv_wordmarks(gt_object, columns,
  league, height = 30, locations = NULL,
  season = NULL, id_system, strict)
gt_sdv_headshots(gt_object, columns,
  league, height = 30, locations = NULL,
  id_system = "espn")
gt_sdv_cols_label(gt_object,
  league, height = 30, mark_type = "logo",
  id_system, season, strict)
```

`height` in px. `locations` (`cells_body()`, `cells_stub()`, `cells_column_labels()`, `cells_row_groups()`) replaces `columns`. `include_name` = `TRUE` keeps the text after the logo; `season` shows one season's marks in every cell. Wordmarks: NFL, MLB (and MiLB). `gt_sdv_cols_label()` turns column *labels* into images: `mark_type` = "logo" "wordmark" "headshot".

STACKED NAME IN TEAM COLOUR

```
gt_merge_stack_team_color(gt_object,
  col1, col2, team_color, league,
  font_size_top = 14,
  font_size_bottom = 12, color = "black",
  background = NULL, season, id_system)
```

`col1` on top in bold caps, `col2` below in the team's colour; a primary under 4.5:1 contrast gives way to the secondary or is darkened. Apply a theme first: `background` = `NULL` reads it.

```
x <- subset(sdv_example_standings,
  league == "nfl",
  c(team, team_name, division))
gt(x) |>
  gt_merge_stack_team_color(team_name,
    division, team, league = "nfl")
```

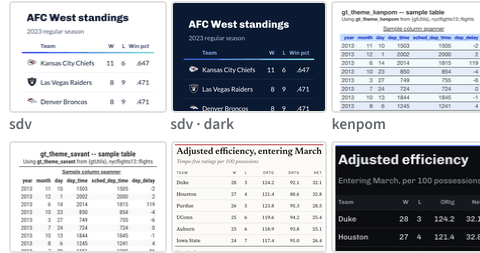
EXAMPLE

```
library(gt)
nfc <- subset(sdv_example_standings,
  division == "NFC Wins",
  c(team, team_name, wins, losses,
    last_season_wins))
gt(nfc) |>
  gt_sdv_logos(columns = team,
    league = "nfl", height = 28) |>
  gt_theme_sdv()
```

Cookbook: [Team tables with the gt toolkit ↗](#)

THEMES

```
gt_theme_sdv(gt, style = "light")
gt_theme_sdv_team(gt, team = "KC",
  league = "nfl")
```



savant broadsheet midnight

`gt_theme_sdv()`: Chivo titles and labels, a Lato body in tubular figures, and the SDV gradient (#3346F0 → #7FE6DC) as one horizon line under the labels; `style` = "dark" sets it on the SportsDataverse navy.

`gt_theme_sdv_team()`: the sdv table in one team's colours (title block primary, horizon line secondary); `team` = `NULL` keeps SportsDataverse colours; takes `season`, `id_system`.

Every `gt_theme_<name>(gt_object, ..., density)` takes `density` = "comfortable" "compact" "social" (default comfortable; compact for almanac, scoreboard, terminal) and passes ... to `gt::tab_options()`, applied last.

Option	Themes (default)
accent	almanac "#8C2F1E" · booktabs, swiss, tuftes "#111111" · broadsheet "#A6081A" · brutalist "#FF3B00" · midnight "#5B8DEF" · scoreboard "#0E1621" · terminal "#FFB86C"
style	sdv, sofa "light" / "dark" · tier "dark" / "light"
other	drench color = "#123F5E" · broadsheet paper = "white" or "salmon" · almanac stripe (NA = off)
none	athletic gtutils kenpom ncaa pl savant

All 20: almanac athletic booktabs broadsheet brutalist drench gtutils kenpom midnight ncaa pl savant scoreboard sdv sdv_team sofa swiss terminal tier tuftes (each `gt_theme_<name>()`).

```
gt(nfc) |>
  gt_theme_broadsheet(paper = "salmon",
    density = "compact")
gt(nfc) |>
  gt_theme_terminal(accent = "#7EE787")
```

Cookbook: [SportsDataverse table themes ↗](#)

PREVIEW & PALETTES

```
gt_theme_preview(nfc, themes = c(
  "gt_theme_sdv", "gt_theme_kenpom"))
gt_color_ranks(..., palette = pal_midnight)
```

`gt_theme_preview(data, themes = NULL, n = 5, ncol = 3, density = "compact", file = NULL, ...)`: `NULL` shows every theme; themes needs full function names. ... goes to `gt_grid()`.

`pal_midnight`: a rank palette that reads on midnight / terminal.



COLOUR BY VALUE

Function	Key arguments
gt_color_ranks()	fill cells: palette domain reverse, na_color = "white", autocolor_text = TRUE
gt_color_pills()	pill: fill_type = "continuous" / "rank", format_type, suffix, digits
gt_percentile_bar()	bar+marker: domain = c(0, 100), scale = "auto" (0-1 → 0-100), na_label = "-"
gt_color_results()	row colours: result_column = "result", result_type = "wl" / "binary"
gt_row_accent()	edge bar: column, palette, side = "left", hide = TRUE
gt_indicator_boxes()	boxes: indicator_vals = c(0, 1), color_yes, show_text
gt_tiers()	tier list: levels, colors, tier_column = "tier", style = "dark"

Most take columns and rows (an expression such as `wins > 10`, or indices). `palette` also takes a paletteer name, e.g. "viridis:mako"; domain = `NULL` uses the data's range.

Cookbook: [Percentile bars and cut lines ↗](#)

TIER LIST

```
gt(seeds) |> # tier + logo URL columns
  gt_tiers(levels = c("Bye", "Wild card"),
    colors = c("#1B7837", "#B2182B")) |>
  gt_legend_discrete()
```

Columns other than `tier_column` render as images (`image_columns` narrows them; `img_height` = "55px"), e.g. a column of `sdv_logo_url()` URLs. Levels may be one named vector, `c(Bye = "#1B7837")`.

Cookbook: [Tier lists ↗](#)

LEGENDS

Function	Key arguments
gt_legend_continuous()	bar: type = "continuous" "steps" "blocks", n_bins = 5, labels ("edges"), title, location = "bottom", width = 200
gt_legend_discrete()	swatches: key_info (label = colour), heading, location = "top", shape, direction
gt(nfc) > gt_color_ranks(wins) > gt_legend_continuous(type = "steps", title = "Wins")	

The continuous legend reads the columns, palette and domain the colouring step recorded; the discrete one reads the key `gt_tiers()` recorded, so `key_info` can be left out.

```
gt(afc_west) |>
  gt_legend_discrete(c("Playoffs" = "#CCE7F5", "Missed" = "#F5CCCC"))
```

Cookbook: [Headers, legends, captions ↗](#)

HIGHLIGHT & FOCUS

Function	Key arguments
gt_highlight_cells()	columns, condition (~ .x > 0.7 or a logical mask), fill = "#FFF3B0"
gt_highlight_na()	fill = "#F0F0F0", missing_text, na_strings = "NA"
gt_outliers()	method = "iqr" "sd" "bounds", threshold (1.5/3), side, symbol, note
gt_spotlight()	rows; dims the rest (dim_color = "auto", WCAG AA); accent_color, if_none
gt_bold_rows()	rows, text_color = "black", highlight_color
gt_group_strips()	band row groups: color = "#F5F5F5", start = 2

```
gt(nfc) |>
  gt_spotlight(team == "DET",
    accent_color = "#0076B6") |>
  gt_outliers(wins, method = "sd",
    note = TRUE)
gt(nfc) |>
  gt_highlight_cells(c(wins, losses),
    ~ .x >= 10, bold = TRUE)
```



sdvplotR :: gt LAYOUT & reactable

The rest of the gtUtils toolkit for **gt**: number formats, rules and border bars, headers and captions, multi-table layouts and image export for posting. Then the **reactable** helpers: team logos, wordmarks and headshots in cells and headers, and team-coloured bars and fills. Examples continue with `nfc` (the NFC North rows of `sdv_example_standings`, page 3).

sdvplotR v0.1.0

webshot2 to save · reactable for tables

GALLERY ↗

COOKBOOKS ↗

GT · FORMAT, LAYOUT, SAVE

FORMAT

Function	Key arguments
<code>gt_delta()</code>	change column to - from: column_label = "Change", percent = FALSE, decimals = 1, arrows = FALSE, after
<code>gt_fmt_rank()</code>	1 → 1st: superscript = TRUE, suffix_size = "0.7em"
<code>gt_fmt_tally()</code>	W-L(-T) in one cell: separator = "-" ", share = FALSE, share_of = 1, share_location = "inline" / "column"
<code>gt_significance()</code>	stars from p_columns: levels = c(0.01, 0.05, 0.1), symbols, hide_p = TRUE
<code>gt_scale_note()</code>	divide and say so: divisor = 1000 ("Figures in thousands."), where = "source_note" "label" "both"
<code>gt_wrap_labels()</code>	break long labels: width = 12, balance = TRUE
<code>gt_set_font()</code>	one family for the table: font_family, from_google_font = TRUE, weight, style
<code>gt(nfc) > gt_fmt_tally(c(wins, losses)) > gt_delta(from = last_season_wins, to = wins)</code>	

Cookbook: [Rolling-window wins](#) ↗

BORDERS & RULES

Function	Key arguments
<code>gt_border_grid()</code>	full grid: color = "black", weight = 1, include_labels = FALSE
<code>gt_border_bars_top() / _bottom()</code>	colour bars: colors, bar_height = 10, img, text, text_size = 18
<code>gt_cutline()</code>	line after rows: after, label, color = "#A6081A", style = "dashed", gap
<code>gt(nfc) > gt_border_bars_top(c("#0076B6", "#B0B7BC")) > gt_cutline(after = c(1, 3), label = c("Champion", NA))</code>	

Cookbook: [Border bars](#) ↗

HEADERS & ANNOTATIONS

Function	Key arguments
<code>gt_title_header()</code>	title, subtitle, kicker, date, one *_style list each
<code>gt_column_subheaders()</code>	per column list(heading, subtitle) in ...; subtitle_color = "#808080"
<code>gt_538_caption()</code>	top_caption over a rule, bottom_caption, align = "right"
<code>gt_marginalia()</code>	a column as notes: width = 220, italic = TRUE, rule = TRUE
<code>gt_social_tag()</code>	handles + icons: accounts, caption, stack = FALSE, separator = " "
<code>gt_watermark()</code>	text or image, opacity = 0.06, size = "60%", angle
<code>gt(nfc) > gt_title_header("NFC North", kicker = "2025 SEASON", subtitle = "Final standings") > gt_column_subheaders(wins = list(heading = "W", subtitle = "2025")) > gt_social_tag(c(x = "@SportsDataverse", gh = "sportsdataverse"))</code>	

`accounts` names map to Font Awesome brand icons, with aliases (x / twitter, ig, bsky, gh). A Date in date prints as "July 21, 2026".

Cookbook: [Headers, legends, captions](#) ↗

SNAKE A LONG TABLE

<code>nba <- subset(sdv_example_standings, league == "nba", c(team, wins, losses)) gt(nba) > gt_snake(n_cols = 3)</code>

Thirty rows become three blocks of ten, each with its own labels. Body-cell styling applied before `gt_snake()` carries through; formats and content transforms (such as `gt_sdv_logos()`) go after it. `gt_snake_align()` reshapes a precomputed mask or colour frame to match.

LEARN MORE

GALLERY ↗ every chart and table in one place

Start: [Getting started](#) · **All:** [articles](#) · [reference](#)

Per league: [NFL](#) · [CFB](#) · [NBA](#) · [WNBA](#) · [MLB](#) · [NHL](#) · [MBB](#) · [WBB](#) · [Soccer](#)

Recipes: [NFL EPA](#) · [CFB table](#) · [WBB standings](#) · [head-to-head card](#)

Leaderboards: [dashboards with Quarto](#) · **Automation:** [social graphics](#)

LAYOUT

Function	Key arguments
<code>gt_grid()</code>	tables in a grid: tables, ncol = 2, labels, title, caption, gap = 24, align = "top", file
<code>gt_stack_tables()</code>	one above another: gap = 16, align = "center", title, file
<code>gt_snake()</code>	long table in blocks: n_cols = 2 or rows_per_col, gap = 20, fill = ""
<code>gt_snake_align()</code>	reshape a parallel frame (a mask for gt_highlight_cells()) to match
<code>gt_grid(list(afc_tbl, nfc_tbl), ncol = 2, title = "2025 standings", file = "standings.png")</code>	

Grids and stacks share `subtitle`, `source_note`, `caption_rule`, the `*_style` lists, and the `save` arguments `bg = "white"`, `whitespace = 50`, `zoom = 2`.

Cookbook: [Faceted tables](#) ↗ · [Schedule matrix](#) ↗

SAVE AS AN IMAGE

Function	Key arguments
<code>gt_save_crop()</code>	PNG/JPG, trimmed and padded: file = NULL, bg = "white", whitespace = 50, zoom = 2, expand = 5, width
<code>gt_social_crop()</code>	pad to aspect_ratio = "1:1", "16:9", "4:5" or a number; whitespace = 60, gravity = "center"
<code>gt_save_batch()</code>	one image per group: data, group, fn, file, dir, match_width = TRUE
<code>tbl > gt_save_crop("table.png", whitespace = 40, width = 1200)</code> <code># one image per conference nfl <- subset(sdv_example_standings, league == "nfl") build <- function(df, group) gt(df) > tab_header(title = group) gt_save_batch(nfl, conference, build, "nfl-{group}.png", dir = "out")</code>	

Saving runs headless Chrome through `webshot2` (Chrome gets 60 s to start). `file = NULL` returns the PNG bytes. `dir` has no default, so a batch never lands in the working directory unasked.

Cookbook: [Saving and posting](#) ↗ · [Social posting](#) ↗

REACTABLE

REACTABLE TABLES

Helper	Arguments (defaults)
<code>reactable_sdv_logos()</code>	league, variant = "default" ("dark", a named variant), height = 30, default_img = NULL
<code>reactable_sdv_wordmarks()</code>	the same
<code>reactable_sdv_headshots()</code>	league, height = 40, default_img, id_system = "espn"
<code>reactable_sdv_team_color_bar()</code>	data, team_col, league, which = "primary", max_value = NULL, na_color = "grey70"
<code>reactable_sdv_team_color_bg()</code>	as the bar, with alpha = 0.15 in place of max_value
<code>reactable_sdv_cols_label()</code>	.data, ... (to every colDef()), league, variant, height = 30

```
library(reactable)
reactable(df, columns = list(
  team = colDef(html = TRUE,
    cell = reactable_sdv_logos("nfl")),
  wins = colDef(style =
    reactable_sdv_team_color_bar(
      df, "team", league = "nfl"))))
# columns named by team key -> logo headers
reactable(h2h,
  columns = reactable_sdv_cols_label(
    h2h, league = "nba"))
```

All but headshots also take `season`, `id_system`, `strict`. The first three are `cell` renderers (`colDef(html = TRUE)`); unresolved values show as text, or `default_img`; a variant that fails to load swaps to the default image. The colour helpers are `style` functions: the bar's length is the value over `max_value` (the column maximum by default). `reactable_sdv_cols_label()` leaves out columns it cannot resolve, so they keep their plain names.

Cookbook: [reactable integration](#) ↗

SIBLINGS

Python `sdvplot` (PyPI) and JavaScript `@sportsdataverse/sdvplot` + `sdvtables` (npm) share `sdvplotR`'s team index, logo archive and argument names; images render through [ggpath](#).



sdvplotR :: CHEAT SHEET

Team logos, wordmarks, player headshots and team colours for **28 leagues** (pro, college, minor and junior hockey, spring football, soccer, cricket) in **ggplot2** plots, **gt** and **reactable** tables, and **plotly**, **ggiraph** and **leaflet**, behind one `league` argument. Built on ggpath, after the approach of nflplotR, cfbplotR, nbaplotR and mlbplotR. Page 1: ggplot2 marks, scales and axes. Page 2: teams, interactive, surfaces, coordinates. Page 3: gt images, themes, colour. Page 4: gt layout, saving, reactable.

sdvplotR v0.1.0

R ≥ 4.1 · ggplot2 ≥ 4.0 · ggpath ≥ 1.1

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GGPLOT2 · MARKS & SCALES

INSTALL & LOAD

```
install.packages("sdvplotR")
# dev: sportsdataverse.r-universe.dev
pak::pak("sportsdataverse/sdvplotR")
library(sdvplotR); library(ggplot2)
```

Every team helper takes `league`, a key from `sdv_leagues()` (`"nfl"`, `"nba"`, `"cfb"`, `"nhl"`, `"ahl"`, `"soccer"`, ..., page 2). Teams: any abbreviation, name or id `sdv_resolve()` reads (ESPN, nflverse, nba_api, MLB Stats, CFBD, ...), or a conference (`"SEC"`, `"B1G"`, `"AFC"`).

Cookbook: [Getting started ↗](#)

LOGOS, WORDMARKS & HEADSHOTS

Geom	Required aes
<code>geom_sdv_logos()</code>	<code>x</code> <code>y</code> <code>team</code>
<code>geom_sdv_wordmarks()</code>	<code>x</code> <code>y</code> <code>team</code> (NFL, MLB, MiLB)
<code>geom_sdv_headshots()</code>	<code>x</code> <code>y</code> <code>player_id</code>
<code>geom_from_path()</code>	<code>x</code> <code>y</code> <code>path</code> (any image)

Optional aes and defaults: `alpha`, `colour` (`"b/w"` greyscales), `angle` = 0, `hjust` = 0.5, `vjust` = 0.5, `width` = 1, `height` = 1 (npc; a typical logo is `width` = 0.075). Logos and wordmarks also take `season` (an aes or a constant).

```
df <- data.frame(
  team = sdv_teams("nba")$abbr,
  a = rnorm(30), b = rnorm(30))
ggplot(df, aes(a, b)) +
  geom_mean_lines(aes(x0 = a, y0 = b)) +
  geom_sdv_logos(aes(team = team),
    league = "nba", width = 0.07)
```

HIGHLIGHT A GROUP

```
nfl <- subset(sdv_example_standings,
  league == "nfl")
west <- c("DEN", "KC", "LAC", "LV")
ggplot(nfl, aes(points_for,
  points_against)) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, !team %in% west),
    league = "nfl", width = 0.05,
    alpha = 0.25) +
  geom_sdv_logos(aes(team = team),
    data = subset(nfl, team %in% west),
    league = "nfl", width = 0.1)
```

Rows draw in order, so the last layer and row end on top. `colour` = `"b/w"` greys logos instead of fading them, which reads better on a dark background.

Cookbook: [Logos, wordmarks, headshots ↗](#)

GEOM ARGUMENTS

```
geom_sdv_logos(mapping = NULL,
  data = NULL, stat = "identity",
  position = "identity", ..., league,
  id_system = "auto", strict = FALSE,
  na.rm = FALSE, show.legend = FALSE,
  inherit.aes = TRUE)
geom_sdv_headshots(..., league,
  id_system = "espn")
```

`geom_sdv_wordmarks()` takes the logo arguments; headshots take no `season` or `strict`. Headshot `id_system`: `"espn"` = ESPN athlete ids in every league; `"gsis"` = nflverse NFL ids; `"league"` = NBA / WNBA Stats PERSON_ID, MLBAM, NHL API.

The ggproto objects `GeomSDVLogo`, `GeomSDVwordmark`, `GeomSDVheadshot` and ggpath's `GeomFromPath`, `GeomReflines` are exported for extensions.

ERA LOGOS

```
old <- data.frame(x = 1:4, y = 1,
  team = c("QUE", "HFD", "ATL", "WPG"),
  season = c(1990, 1995, 2005, 2015))
ggplot(old, aes(x, y)) +
  geom_sdv_logos(aes(team = team,
    season = season), league = "nhl",
  width = 0.2) + theme_void()
```

`season`: the mark worn that year (ending year for split seasons); it also picks who held a reused code. NHL: every identity since 1917-18 (`QUE` `HFD`); AHL, ECHL, OHL, QMJHL, WHL, USHL, PWHL, PHF, UFL, XFL, AAF: dated marks; NFL / WNBA: relocated identities (`STL` `SD` `OAK`; `SA` `TUL`). Elsewhere today's mark. NHL marks are SVG: needs `rsvg`.

REFERENCE LINES

```
p + geom_mean_lines(aes(x0 = a, y0 = b))
p + geom_median_lines(aes(x0 = a),
  colour = "grey40", linetype = 1)
```

From ggpath. Optional aes `x0`, `y0`; defaults `colour` = `"red"`, `linewidth` = 0.5, `linetype` = 2 (dashed), `alpha` = NA.

ANY IMAGE: GEOM_FROM_PATH

```
df$url <- sdv_logo_url(df$team, "nba",
  variant = "dark")
ggplot(df, aes(a, b)) +
  geom_from_path(aes(path = url),
    width = 0.07)
```

`path` is a URL or a local file; same optional aes as the team geoms. Conferences resolve by key or other name: `sdv_logo_url(c("SEC", "B1G"), "cfb")`; a team's code `wins` (`cfb` "MAC" is Macalester).

COLOUR & FILL SCALES

```
scale_color_sdv(league, which = "primary",
  values = NULL, ..., aesthetics = "colour",
  breaks = waiver(), na.value = "grey50",
  guide = NULL, alpha = NA)
scale_fill_sdv(...) # aesthetics = "fill"
```

Maps team keys to official colours; which = `"secondary"` for the second colour, `alpha` fades them (`scales::alpha()`), `values` replaces them, ... goes to `scale_colour_manual()`. `scale_colour_sdv()` is an alias.

```
ggplot(df, aes(team, b)) +
  geom_col(aes(colour = team, fill = team)) +
  scale_color_sdv("nba", "secondary") +
  scale_fill_sdv("nba", alpha = 0.4)
```



NFL primary / secondary, as `sdv_team_colors("nfl")`, which = `"all"` returns them.

Cookbook: [Team colours and themes ↗](#)

TITLES

```
p + ggtitle_image(title_image = "SEA",
  title = "Seattle Seahawks Analysis",
  image_height = 20, image_side = "left",
  league = "nfl") +
  theme_title_image(size = 20, hjust = 0.5)
```

`title_image`: a team key (logo resolved) or any image URL. `image_height` = 15 px by default; `image_side` = `"left"` or `"right"`; subtitle. `theme_title_image(...)` is required and passes ... to `ggtext::element_markdown()`.

LOGO AT EACH BAR'S END

```
nba <- subset(sdv_example_standings,
  league == "nba")
ggplot(nba, aes(wins,
  reorder(team, wins))) +
  geom_col(aes(fill = team)) +
  geom_sdv_logos(aes(x = wins + 3,
    team = team), league = "nba",
  height = 0.028) +
  scale_fill_sdv("nba")
```

`height` sizes the logo here: each row is about a thirtieth of the panel.

LOGO AXES

Route	Calls
<code>theme</code>	<code>element_sdv_logo()</code> · <code>element_sdv_wordmark()</code> · <code>element_sdv_headshot()</code>
<code>scale+theme</code>	<code>scale_x_sdv()</code> / <code>scale_y_sdv()</code> · <code>scale_x_sdv_headshots()</code> + <code>theme_x_sdv()</code> / <code>theme_y_sdv()</code>

```
# element route (size in cm)
ggplot(df, aes(team, b)) + geom_col() +
  theme(axis.text.x =
    element_sdv_logo("nba", size = 1))
```

`element_sdv_logo(league, alpha = NULL, colour = NA, hjust = NULL, vjust = NULL, size = 0.5, season, id_system)`; `_wordmark()` the same; `_headshot()` takes `id_system`, no `season`.

```
# scale route (size in px; needs ggtext)
ggplot(df, aes(b, team)) + geom_col() +
  scale_y_sdv(league = "nba", size = 12) +
  theme_y_sdv()
```

`scale_x_sdv(..., league, expand, guide, position = "bottom", size = 12, season, id_system)`; `scale_y_sdv()` has position = `"left"`. Size is the height on x, the width on y. `scale_x_sdv_headshots()` (`size` = 20) and `scale_y_sdv_headshots()` (`size` = 30) take `id_system`, `theme_x_sdv()` / `theme_y_sdv()` render those labels as markdown.

Cookbook: [Logo axes and titles ↗](#)

HEADSHOTS ON A SCATTER

```
ggplot(scorers, aes(ppg, ts)) +
  geom_sdv_headshots(aes(player_id =
    athlete_id), league = "nba",
  height = 0.12)
```

ESPN ids: the `athlete_id` of hoopR, wehoop, cfbfastR rosters. NFL `gsis` ids (`id_system` = `"gsis"`) go through a headshot map built from nflverse rosters (1999 on); `sdv_clear_cache()` re-reads it.

IMAGES IN THEME ELEMENTS

```
p + theme(plot.background =
  element_sdv_raster("court.png"))
```

`element_sdv_raster(image_path, x, y, width, height, just = "centre", hjust = 0.5, vjust = 0.5, interpolate = TRUE, ...)`: one image in a rectangle; wraps ggpath's `element_raster()` (also exported).

`element_path(alpha = 1, colour = NA, hjust, vjust, angle = 0, size = unit(0.5, "cm"))`: axis text that already holds image paths or URLs.



sdvplotR :: TEAMS, INTERACTIVE & SURFACES

The shared SportsDataverse team index behind every helper: **28 leagues**, any team identifier resolved to one canonical `team_id`, colours, palettes and archived logo URLs. Plus logos on **plotly**, **ggiraph** and **leaflet**, premade tier plots, team-coloured playing surfaces drawn with **sportyR**, and converters that put shot and event coordinates from stats.nba.com, Euroleague and ten soccer providers on one frame.

sdvplotR v0.1.0

sportyR for surfaces · rsvg for SVG logos

GALLERY ↗

COOKBOOKS ↗

TEAM DATA · LEAGUES, KEYS, COLOURS

LEAGUES & TEAMS

```
sdv_leagues() # the 28 league keys
sdv_teams("nfl")
sdv_teams("cfb", include_conferences = TRUE)
```

Group	sdv_teams() columns
ids, names	league team_id abbr name short_name location
grouping	program conference_id conference
colours	color_primary color_secondary color_source

`team_id` is the canonical key (the ESPN id where ESPN has one). Conference rows ("SEC", "AFC") draw logos only.

RESOLVE ANY IDENTIFIER

```
sdv_resolve(c("GSW", "Golden State Warriors",
"XYZ"), "nba")
#> "9" "9" NA + one warning
sdv_resolve(c("OAK", "SD", "STL"), "nfl")
#> "13" "24" "14" LV, LAC, LA
sdv_suggest("Kansas Cty Chiefs", "nfl")
```

`sdv_resolve(x, league, season = NULL, id_system = "auto", strict = FALSE)` reads abbreviations, names and ids from ESPN, nflverse, MLB Stats, nba_api, NHL, HockeyTech, NCAA, PFF, CFBD, Sports Reference, FanGraphs and more. `id_system` names one ("nhl_id" only explicitly). `season` picks who held a reused code (ending year for split seasons). Unresolved: NA and one warning (`strict = TRUE` errors). `sdv_suggest(x, league, n = 5)` lists candidates, never picks.

Every helper that takes a team resolves it this way, with `league` and `id_system`. `strict`: the logo-URL, colour, tier and mark-drawing helpers; not `sdv_marks()`, `element_sdv_*()`, `scale_x/y_sdv()`, `sdv_colorway()`, `sdv_team_factor()`.

TEAM FACTOR

```
sdv_team_factor(c("KC", "BUF", "invalid"),
  league = "nfl")
#> KC BUF <NA> Levels: BUF KC
```

Levels ordered by abbreviation; unresolved values become NA, so scales drop them.

COLOURS & PALETTES

```
sdv_team_colors("nfl", "KC")
sdv_team_colors("nfl", c("KC", "BUF"),
  which = "all")
sdv_team_colors("nba", which = "all")
sdv_color_palette("nba",
  teams = c("BOS", "NY"))
```

`sdv_team_colors(league, team = NULL, which = "primary", season, id_system, strict, include_conferences = FALSE)`: a named hex vector (NA where a team does not resolve); `which = "all"` gives "primary, secondary" strings, or with `team = NULL` a dataframe `team_id abbr name primary secondary`. `sdv_color_palette(league, teams = NULL, which = "primary", ...)`: named colours for `scale_*_manual()`.

IMAGE URLS & MARKS

```
sdv_logo_url(c("KC", "Buffalo Bills",
"OAK"), "nfl")
sdv_logo_url("NYJ", "nfl",
  variant = "scoreboard")
sdv_logo_url("QUE", "nhl", season = 1990)
sdv_logo_url("KC", "nfl",
  mark_type = "wordmark")
sdv_marks("KC", "nfl") # every mark
sdv_headshot_url("2544", "nba",
  id_system = "league")
```

`sdv_logo_url(team, league, season = NULL, variant = "default", mark_type = "logo", id_system, strict)`: the SportsDataverse archive's immutable copy. `variant`: "default", "dark", or a name from `sdv_marks()` ("scoreboard", NFL "grayscale", MLB "cap_on_light", ...); a team without it gets its closest mark of the same polarity, an unknown name errors. `mark_type = "wordmark"`: NFL, MLB (and MiLB).

`sdv_headshot_url(player_id, league, id_system = "espn")`: ESPN headshots for nfl nba wnba mlb nhl cfb mbb wbb; "gsis" and "league" as on page 1. NA for a missing or malformed id.

CACHE & VERSIONS

Index and logo manifest come from the SportsDataverse CDN, cached for a day; a copy ships in the package (`options(sdvplotR.offline = TRUE)`). `sdv_versions()`: what is loaded, from where. `sdv_clear_cache()`: drop index, manifest, NFL headshot map and ggpath images.

COVERAGE

Leagues	Teams
nfl nba mlb nhl wnba	32 · 30 · 30 · 32 · 15
nbagl mlb pwhl phf	32 · 286 · 12 · 8
ufl xfl usfl aaf	11 · 11 · 9 · 8
ahl echl ohl whl qmjhl ushl	61 · 59 · 27 · 23 · 30 · 17
cfb mbb wbb	687 · 366 · 364
ncaa_baseball softball mhockey whockey	414 · 415 · 110 · 48
soccer · cricket	2,631 · 111

Team rows in the bundled index 0aeef853ec2b; the live index can differ.

PREMADE PLOT: TEAM TIERS

```
tiers <- data.frame(
  tier_no = c(1, 1, 2, 2, 3),
  team = c("KC", "DET", "BAL", "PHI", "BUF"))
sdv_team_tiers(tiers, league = "nfl",
  theme = "light", no_line_below_tier = 2)
```

`sdv_team_tiers(data, league, title, subtitle, caption = NULL, tier_desc, presort = FALSE, alpha = 0.8, width = 0.075, no_line_below_tier = NULL, devel = FALSE, theme = "dark", variant = "auto", season, id_system, strict)`. data needs `tier_no` and `team` (optional `tier_rank`). Title "NFL Team Tiers" from `league`; NULL drops one. `devel = TRUE` plots keys, not logos; "auto": dark marks on the dark theme.

Cookbook: [College hoops tiers](#) ↗

INTERACTIVE: GGIRAPH, PLOTLY, LEAFLET

```
p <- ggplot(d, aes(x, y)) +
  geom_sdv_logos_interactive(aes(team = team,
  tooltip = team, data_id = team),
  league = "nfl", width = 0.1)
ggiraph::girafe(ggobj = p)
plotly::plot_ly(d, x = ~x, y = ~y) |>
  add_sdv_logos(d$x, d$y, d$team, "nfl") |>
  sdv_colorway("nfl", d$team)
leaflet::leaflet(d) |> leaflet::addTiles() |>
  addSdvLogoMarkers(~lon, ~lat, ~team,
  league = "nfl")
```

`geom_sdv_{logos,wordmarks,headshots}_interactive()`: images plus an invisible hover point (`hover_size = 8`). `plotly`: `add_sdv_logos(p, x, y, team, league, size = 0.08, ...)`, `add_sdv_wordmarks()`, `add_sdv_headshots()`, `sdv_axis_logos(p, teams, league, axis)` tick logos, `sdv_colorway()`; `ggplotly()` drops them. `addSdvLogoMarkers(..., size = 32)`: no logo keeps the default marker.

Cookbook: [Interactive plots](#) ↗

PLAYING SURFACES

```
sdv_surface("nba", "BOS",
  center_logo = TRUE) +
  geom_point(aes(x, y), data = shots)
```

league	sportyR surface
nfl aaf ufl usfl xfl cfb	geom_football("nfl" / "ncaa")
nba nbagl wnba mbb wbb fiba	geom_basketball(); fiba 28 × 15 m
mlb mlb ncaa_baseball	geom_baseball()
nhl ahl echl ohl qmjhl ushl pwhl phf ncaa_mhockey ncaa_whockey	geom_hockey(); whl on the NHL rink
soccer	geom_soccer("fifa"), 105 × 68 m

`sdv_surface(league, team = NULL, season = NULL, center_logo = FALSE, ...)` (not cricket, ncaa_softball). A team paints the paint and apron, end zones, or centre line, circle and boards; baseball stays regulation, soccer gets a centre logo only. Stylized, not real floors. Origin at centre, in feet (yards for football, metres for soccer and FIBA); ... goes to `sportyR` (`rotation`, `xlims`, `color_updates`).

Cookbook: [Surfaces and shot charts](#) ↗

SHOT & EVENT COORDINATES

```
# stats.nba.com legacy frame -> feet
shots |> sdv_court_coords()
# Euroleague (cm, hoop origin) -> FIBA m
pts |> sdv_court_coords(
  x_column = "coord_x", y_column = "coord_y",
  provider = "euroleague")
# soccer events -> 105 x 68 m pitch
ev |> sdv_pitch_coords("statsbomb")
```

`sdv_court_coords(data, x_column = "x_legacy", y_column = "y_legacy", provider = "nba")` adds `court_x` `court_y`: basket at -41.75 ft (-12.425 m for Euroleague), half court at 0; `court_y` -25 to 25 ft, negative = shooter's left. From hoopR's `nba_shotchartdetail()`: `sdv_court_coords(res$Shot_Chart_Detail, "LOC_X", "LOC_Y")`.

`sdv_pitch_coords(data, provider, x_column = NULL, y_column = NULL, flip = NULL, pitch_length = NULL, pitch_width = NULL)` adds `pitch_x` `pitch_y` in metres from the centre spot (`pitch_x` -52.5 = defended goal line). `flip`: TRUE / FALSE or a logical column, as `flip = "away"`, turns rows half a turn. `pitch_length` / `pitch_width`: tracking only. Providers: opta (statsperform) wyscout statsbomb uefa impect espn, tracking tracab skillcorner secondspectrum metrica.

Cookbook: [NHL shot map](#) ↗ · [Soccer](#) ↗



sdvplotR :: gt TABLES

Team images and colours for **gt**, plus the **gtUtils** toolkit (by Andrew Weatherman): 20 themes, legends, colouring, annotations, layouts and image export. Every helper takes a `gt_tbl` as its first argument (`gt_object`) and returns one, so they chain with `|>`. Examples run on `sdv_example_standings` (real NFL and NBA final standings). Page 3: images, themes, colour, legends, focus. Page 4: format, layout, saving, reactable.

sdvplotR v0.1.0

gt ≥ 0.10 · paletteer optional

[GALLERY ↗](#)

[COOKBOOKS ↗](#)

GT · IMAGES, THEMES, COLOUR

TEAM IMAGES IN CELLS

```
gt_sdv_logos(gt_object, columns, league,
  height = 30, locations = NULL,
  include_name = FALSE, season = NULL,
  id_system = "auto", strict = FALSE)
gt_sdv_wordmarks(gt_object, columns,
  league, height = 30, locations = NULL,
  season = NULL, id_system, strict)
gt_sdv_headshots(gt_object, columns,
  league, height = 30, locations = NULL,
  id_system = "espn")
gt_sdv_cols_label(gt_object,
  columns = everything(), league,
  height = 30, mark_type = "logo",
  id_system, season, strict)
```

`height` in px. `locations` (`cells_body()`, `cells_stub()`, `cells_column_labels()`, `cells_row_groups()`) replaces `columns.include_name = TRUE` keeps the text after the logo; `season` shows one season's marks in every cell. Wordmarks: NFL, MLB (and MiLB). `gt_sdv_cols_label()` turns column *labels* into images: `mark_type = "logo"` "wordmark" "headshot".

STACKED NAME IN TEAM COLOUR

```
gt_merge_stack_team_color(gt_object,
  col1, col2, team_col, league,
  font_size_top = 14,
  font_size_bottom = 12, color = "black",
  background = NULL, season, id_system)
```

`col1` on top in bold caps, `col2` below in the team's colour; a primary under 4.5:1 contrast gives way to the secondary or is darkened. Apply a theme first: `background = NULL` reads it.

```
x <- subset(sdv_example_standings,
  league == "nfl",
  c(team, team_name, division))
gt(x) |>
  gt_merge_stack_team_color(team_name,
    division, team, league = "nfl")
```

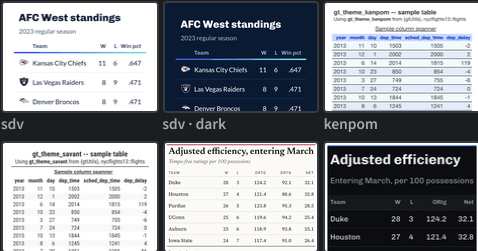
EXAMPLE

```
library(gt)
nfc <- subset(sdv_example_standings,
  division == "NFC North",
  c(team, team_name, wins, losses,
    last_season_wins))
gt(nfc) |>
  gt_sdv_logos(columns = team,
    league = "nfl", height = 28) |>
  gt_theme_sdv()
```

Cookbook: [Team tables with the gt toolkit ↗](#)

THEMES

```
gt_theme_sdv(gt, style = "Light")
gt_theme_sdv_team(gt, team = "KC",
  league = "nfl")
```



savant broadsheet midnight

`gt_theme_sdv()` : Chivo titles and labels, a Lato body in tabular figures, and the SDV gradient (#3346F0 → #7FE6DC) as one horizon line under the labels; `style = "dark"` sets it on the SportsDataverse navy.

`gt_theme_sdv_team()` : the sdv table in one team's colours (title block primary, horizon line secondary); `team = NULL` keeps SportsDataverse colours; takes `season`, `id_system`.

Every `gt_theme<name>(gt_object, ..., density)` takes `density = "comfortable"` "compact" "social" (default comfortable; compact for almanac, scoreboard, terminal) and passes ... to `gt::tab_options()`, applied last.

Option	Themes (default)
accent	almanac "#8C2F1E" · booktabs, swiss, tuftes "#111111" · broadsheet "#A6081A" · brutalist "#FF3B00" · midnight "#5B8DEF" · scoreboard "#0E1621" · terminal "#FFB86C"
style	sdv, sofa "light" / "dark" · tier "dark" / "light"
other	drench color = "#123F5E" · broadsheet paper = "white" or "salmon" · almanac stripe (NA = off)
none	athletic gtutils kenpom ncaa pl savant

All 20: almanac athletic booktabs broadsheet brutalist drench gtutils kenpom midnight ncaa pl savant scoreboard sdv sdv_team sofa swiss terminal tier tuftes (each `gt_theme<name>()`).

```
gt(nfc) |>
  gt_theme_broadsheet(paper = "salmon",
    density = "compact")
gt(nfc) |>
  gt_theme_terminal(accent = "#7EE787")
```

Cookbook: [SportsDataverse table themes ↗](#)

PREVIEW & PALETTES

```
gt_theme_preview(nfc, themes = c(
  "gt_theme_sdv", "gt_theme_kenpom"))
gt_color_ranks(..., palette = pal_midnight)
```

`gt_theme_preview(data, themes = NULL, n = 5, ncol = 3, density = "compact", file = NULL, ...)`: `NULL` shows every theme; themes needs full function names. ... goes to `gt_grid()`.

`pal_midnight`: a rank palette that reads on midnight / terminal.

COLOUR BY VALUE

Function	Key arguments
gt_color_ranks()	fill cells: palette domain reverse, na_color = "white", autocolor_text = TRUE
gt_color_pills()	pill: fill_type = "continuous" / "rank", format_type, suffix, digits
gt_percentile_bar()	bar+marker: domain = c(0, 100), scale = "auto" (0-1 → 0-100), na_label = "-"
gt_color_results()	row colours: result_column = "result", result_type = "wl" / "binary"
gt_row_accent()	edge bar: column, palette, side = "left", hide = TRUE
gt_indicator_boxes()	boxes: indicator_vals = c(0, 1), color_yes, show_text
gt_tiers()	tierlist: levels, colors, tier_column = "tier", style = "dark"

Most take `columns` and `rows` (an expression such as `wins > 10`, or indices). `palette` also takes a paletteer name, e.g. "viridis::mako"; `domain = NULL` uses the data's range.

Cookbook: [Percentile bars and cut lines ↗](#)

TIER LIST

```
gt(seeds) |> # tier + logo URL columns
  gt_tiers(levels = c("Bye", "Wild card"),
    colors = c("#1B7837", "#B2182B")) |>
  gt_legend_discrete()
```

Columns other than `tier_column` render as images (`image_columns` narrows them; `img_height = "55px"`), e.g. a column of `sdv_logo_url()` URLs. Levels may be one named vector, `c(Bye = "#1B7837")`.

Cookbook: [Tier lists ↗](#)

LEGENDS

Function	Key arguments
gt_legend_continuous()	bar: type = "continuous" "steps" "blocks", n_bins = 5, labels ("edges"), title, location = "bottom", width = 200
gt_legend_discrete()	swatches: key_info (label = colour), heading, location = "top", shape, direction
gt(nfc) > gt_color_ranks(wins) > gt_legend_continuous(type = "steps", title = "Wins")	

The continuous legend reads the columns, palette and domain the colouring step recorded; the discrete one reads the key `gt_tiers()` recorded, so `key_info` can be left out.

```
gt(afc_west) |>
  gt_legend_discrete(c("Playoffs" = "#CCE7F5", "Missed" = "#F5CCCC"))
```

Cookbook: [Headers, legends, captions ↗](#)

HIGHLIGHT & FOCUS

Function	Key arguments
gt_highlight_cells()	columns, condition (~ .x > 0.7 or a logical mask), fill = "#FFF3B0"
gt_highlight_na()	fill = "#F0F0F0", missing_text, na_strings = "NA"
gt_outliers()	method = "iqr" "sd" "bounds", threshold (1.5/3), side, symbol, note
gt_spotlight()	rows; dims the rest (dim_color = "auto", WCAG AA); accent_color, if_none
gt_bold_rows()	rows, text_color = "black", highlight_color
gt_group_strips()	band row groups: color = "#F5F5F5", start = 2

```
gt(nfc) |>
  gt_spotlight(team == "DET",
    accent_color = "#0076B6") |>
  gt_outliers(wins, method = "sd",
    note = TRUE)
gt(nfc) |>
  gt_highlight_cells(c(wins, losses),
    ~ .x >= 10, bold = TRUE)
```



sdvplotR :: gt LAYOUT & reactable

The rest of the gtUtils toolkit for **gt**: number formats, rules and border bars, headers and captions, multi-table layouts and image export for posting. Then the **reactable** helpers: team logos, wordmarks and headshots in cells and headers, and team-coloured bars and fills. Examples continue with `nfc` (the NFC North rows of `sdv_example_standings`, page 3).

sdvplotR v0.1.0

webshot2 to save · reactable for tables

GALLERY ↗

COOKBOOKS ↗

GT · FORMAT, LAYOUT, SAVE

FORMAT

Function	Key arguments
<code>gt_delta()</code>	change column to - from: <code>column_label = "Change"</code> , <code>percent = FALSE</code> , <code>decimals = 1</code> , <code>arrows = FALSE</code> , <code>after</code>
<code>gt_fmt_rank()</code>	1→1st: <code>superscript = TRUE</code> , <code>suffix_size = "0.7em"</code>
<code>gt_fmt_tally()</code>	W-L(-T) in one cell: <code>separator = "-"</code> , <code>share = FALSE</code> , <code>share_of = 1</code> , <code>share_location = "inline"</code> / <code>"column"</code>
<code>gt_significance()</code>	stars from p_columns: <code>levels = c(0.01, 0.05, 0.1)</code> , <code>symbols</code> , <code>hide_p = TRUE</code>
<code>gt_scale_note()</code>	divide and say so: <code>divisor = 1000</code> ("Figures in thousands."), <code>where = "source_note"</code> "label" "both"
<code>gt_wrap_labels()</code>	break long labels: <code>width = 12</code> , <code>balance = TRUE</code>
<code>gt_set_font()</code>	one family for the table: <code>font_family</code> , <code>from_google_font = TRUE</code> , <code>weight</code> , <code>style</code>
<pre>gt(nfc) > gt_fmt_tally(c(wins, losses)) > gt_delta(from = last_season_wins, to = wins)</pre>	

Cookbook: [Rolling-window wins](#) ↗

BORDERS & RULES

Function	Key arguments
<code>gt_border_grid()</code>	full grid: <code>color = "black"</code> , <code>weight = 1</code> , <code>include_labels = FALSE</code>
<code>gt_border_bars_top()</code> / <code>_bottom()</code>	colour bars: <code>colors</code> , <code>bar_height = 10</code> , <code>img</code> , <code>text</code> , <code>text_size = 18</code>
<code>gt_cutline()</code>	line after rows: <code>after</code> , <code>label</code> , <code>color = "#A6081A"</code> , <code>style = "dashed"</code> , <code>gap</code>
<pre>gt(nfc) > gt_border_bars_top(c("#0076B6", "#B0B7BC")) > gt_cutline(after = c(1, 3), label = c("Champion", NA))</pre>	

Cookbook: [Border bars](#) ↗

HEADERS & ANNOTATIONS

Function	Key arguments
<code>gt_title_header()</code>	<code>title</code> , <code>subtitle</code> , <code>kicker</code> , <code>date</code> , one <code>*_style</code> list each
<code>gt_column_subheaders()</code>	per column list(heading, subtitle) in ...; <code>subtitle_color = "#808080"</code>
<code>gt_538_caption()</code>	<code>top_caption</code> over a rule, <code>bottom_caption</code> , <code>align = "right"</code>
<code>gt_marginalia()</code>	a column as notes: <code>width = 220</code> , <code>italic = TRUE</code> , <code>rule = TRUE</code>
<code>gt_social_tag()</code>	handles + icons: <code>accounts</code> , <code>caption</code> , <code>stack = FALSE</code> , <code>separator = " "</code>
<code>gt_watermark()</code>	<code>text</code> or <code>image</code> , <code>opacity = 0.06</code> , <code>size = "60%"</code> , <code>angle</code>

```
gt(nfc) |>
  gt_title_header("NFC North",
                  kicker = "2025 SEASON",
                  subtitle = "Final standings") |>
  gt_column_subheaders(wins = list(
    heading = "W", subtitle = "2025")) |>
  gt_social_tag(c(x = "@SportsDataverse",
                  gh = "sportsdataverse"))
```

`accounts` names map to Font Awesome brand icons, with aliases (x / twitter, ig, bsky, gh). A Date in date prints as "July 21, 2026".

Cookbook: [Headers, legends, captions](#) ↗

SNAKE A LONG TABLE

```
nba <- subset(sdv_example_standings,
              league == "nba", c(team, wins, losses))
gt(nba) |> gt_snake(n_cols = 3)
```

Thirty rows become three blocks of ten, each with its own labels. Body-cell styling applied before `gt_snake()` carries through; formats and content transforms (such as `gt_sdv_logos()`) go after it. `gt_snake_align()` reshapes a precomputed mask or colour frame to match.

LEARN MORE

GALLERY ↗ every chart and table in one place

Start: [Getting started](#) · **All:** [articles](#) · [reference](#)

Per league: [NFL](#) · [CFB](#) · [NBA](#) · [WNBA](#) · [MLB](#) · [NHL](#) · [MBB](#) · [WBB](#) · [Soccer](#)

Recipes: [NFL EPA](#) · [CFB table](#) · [WBB standings](#) · [head-to-head card](#)

Leaderboards: [dashboards with Quarto](#) · **Automation:** [social graphics](#)

LAYOUT

Function	Key arguments
<code>gt_grid()</code>	tables in a grid: <code>tables</code> , <code>ncol = 2</code> , <code>labels</code> , <code>title</code> , <code>caption</code> , <code>gap = 24</code> , <code>align = "top"</code> , <code>file</code>
<code>gt_stack_tables()</code>	one above another: <code>gap = 16</code> , <code>align = "center"</code> , <code>title</code> , <code>file</code>
<code>gt_snake()</code>	long table in blocks: <code>n_cols = 2</code> or <code>rows_per_col</code> , <code>gap = 20</code> , <code>fill = ""</code>
<code>gt_snake_align()</code>	reshape a parallel frame (a mask for <code>gt_highlight_cells()</code>) to match
<pre>gt_grid(list(afc_tbl, nfc_tbl), ncol = 2, title = "2025 standings", file = "standings.png")</pre>	

Grids and stacks share `subtitle`, `source_note`, `caption_rule`, the `*_style` lists, and the `save` arguments `bg = "white"`, `whitespace = 50`, `zoom = 2`.

Cookbook: [Faceted tables](#) ↗ · [Schedule matrix](#) ↗

SAVE AS AN IMAGE

Function	Key arguments
<code>gt_save_crop()</code>	PNG/JPG, trimmed and padded: <code>file = NULL</code> , <code>bg = "white"</code> , <code>whitespace = 50</code> , <code>zoom = 2</code> , <code>expand = 5</code> , <code>width</code>
<code>gt_social_crop()</code>	pad to aspect_ratio = "1:1", "16:9", "4:5" or number; <code>whitespace = 60</code> , <code>gravity = "center"</code>
<code>gt_save_batch()</code>	one image per group: <code>data</code> , <code>group</code> , <code>fn</code> , <code>file</code> , <code>dir</code> , <code>match_width = TRUE</code>

```
tbl |> gt_save_crop("table.png",
                  whitespace = 40,
                  width = 1200)

# one image per conference
nfl <- subset(sdv_example_standings,
             league == "nfl")
build <- function(df, group)
  gt(df) |> tab_header(title = group)
gt_save_batch(nfl, conference, build,
             "nfl-{group}.png", dir = "out")
```

Saving runs headless Chrome through `webshot2` (Chrome gets 60 s to start). `file = NULL` returns the PNG bytes. `dir` has no default, so a batch never lands in the working directory unasked.

Cookbook: [Saving and posting](#) ↗ · [Social posting](#) ↗

REACTABLE

REACTABLE TABLES

Helper	Arguments (defaults)
<code>reactable_sdv_logos()</code>	<code>league</code> , <code>variant = "default"</code> ("dark", a named variant), <code>height = 30</code> , <code>default_img = NULL</code>
<code>reactable_sdv_wordmarks()</code>	the same
<code>reactable_sdv_headshots()</code>	<code>league</code> , <code>height = 40</code> , <code>default_img</code> , <code>id_system = "espn"</code>
<code>reactable_sdv_team_color_bar()</code>	<code>data</code> , <code>team_col</code> , <code>league</code> , <code>which = "primary"</code> , <code>max_value = NULL</code> , <code>na_color = "grey70"</code>
<code>reactable_sdv_team_color_bg()</code>	as the bar, with <code>alpha = 0.15</code> in place of <code>max_value</code>
<code>reactable_sdv_cols_label()</code>	.data, ... (to every <code>colDef()</code>), <code>league</code> , <code>variant</code> , <code>height = 30</code>

```
library(reactable)
reactable(df, columns = list(
  team = colDef(html = TRUE,
    cell = reactable_sdv_logos("nfl")),
  wins = colDef(style =
    reactable_sdv_team_color_bar(
      df, "team", league = "nfl"))))
# columns named by team key -> logo headers
reactable(h2h,
  columns = reactable_sdv_cols_label(
    h2h, league = "nba"))
```

All but headshots also take `season`, `id_system`, `strict`. The first three are `cell` renderers (`colDef(html = TRUE)`); unresolved values show as text, or `default_img`; a variant that fails to load swaps to the default image. The colour helpers are `style` functions: the bar's length is the value over `max_value` (the column maximum by default). `reactable_sdv_cols_label()` leaves out columns it cannot resolve, so they keep their plain names.

Cookbook: [reactable integration](#) ↗

SIBLINGS

Python [sdvplot](#) (PyPI) and JavaScript [@sportsdataverse/sdvplot](#) + [sdvtables](#) (npm) share `sdvplotR`'s team index, logo archive and argument names; images render through [ggpath](#).