

Package: cfbplotR (via r-universe)

October 12, 2026

Title CFB Logo Plots in 'ggplot2'

Version 0.1.0

Description A set of functions to visualize College Football analysis
in 'ggplot2'.

License MIT + file LICENSE

URL <https://cfbplotR.sportsdataverse.org>,
<https://github.com/sportsdataverse/cfbplotR>

BugReports <https://github.com/sportsdataverse/cfbplotR/issues>

Depends R (>= 4.1.0)

Imports cli (>= 3.0.0), dplyr, ggpath (>= 1.1.0), ggplot2 (>= 4.0.0),
glue, grid, gt, magrittr (>= 2.0.0), rlang (>= 0.4.11), scales
(>= 1.1.0)

Suggests cfbfastR, forcats, ggtext (>= 0.1.1), gridtext (>= 0.1.4),
knitr, remotes, rmarkdown, rstudioapi (>= 0.13), sjmisc (>= 2.8.7), testthat (>= 3.0.0), tidyr (>= 1.0.0), vdiffr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs
cmake make libmagick++-dev gsfonts libuv1-dev libxml2-dev libssl-dev libnode-dev

Repository <https://sportsdataverse.r-universe.dev>

Date/Publication 2026-10-12 10:10:19 UTC

RemoteUrl <https://github.com/sportsdataverse/cfbplotR>

RemoteRef HEAD

RemoteSha 53417b11658c6c3226e6a967ede7d6c722f70823

Contents

.cfbplotR_clear_cache	2
add_athlete_id_col	3
cfb_team_factor	4
cfb_team_tiers	4
clean_school_names	6
clean_team_abbrs	7
data	7
element	8
geom_cfb_headshots	12
geom_cfb_wordmarks	15
GeomCFBheads	19
ggpreview	22
ggtitle_image	24
gt_cfb	25
gt_cfb_cols_label	26
gt_stack_team	27
scale_axes_cfb	28
scale_color_cfb	31
theme_cfb	33
valid_team_names	34
Index	35

.cfbplotR_clear_cache	<i>Clear the cfbplotR (ggpath) image cache</i>
-----------------------	--

Description

Clears the image cache used by the ggpath rendering backend. ggpath (as of its current release) does not expose a public cache-clearing function, so this is a safe no-op that is forward-compatible: if a future ggpath version exports clear_cache() this will call it automatically.

Usage

```
.cfbplotR_clear_cache()
```

Value

Invisibly NULL.

add_athlete_id_col	<i>Add Athlete ID's to data frame</i>
--------------------	---------------------------------------

Description

This function attempts to add ESPN athlete ID's to a data frame using the roster data in the cfbfastR-data repo. The function is experimental and not guaranteed to be accurate.

Usage

```
add_athlete_id_col(df, name_col, team_col = NULL, headshot_urls = FALSE)
```

Arguments

df	a data frame.
name_col	the column in df with the player names to join with the roster data.
team_col	Optional column with team names to join with the roster data to reduce the chance of matching with two players from different teams with the same name. If NULL and df has a column named "team" or "school," the function will use those as team_col. The function also checks df for a column named "season" to match names to rosters going back to 2009.
headshot_urls	logical to return headshot URLs. If TRUE, the output has an additional column called "headshot_url" with links for player headshots.

Value

the original df with extra columns:

athlete_id athlete ESPN ID.

headshot_url url of athlete's headshot.

Examples

```
x <- data.frame(
  player_name = c("Britain Covey", "JT Daniels")
)

add_athlete_id_col(x, player_name)

x$season <- c(2021, 2021)
add_athlete_id_col(x, player_name)

x$team = c("Utah", "Georgia")
add_athlete_id_col(x, player_name, team, headshot_urls = TRUE)
```

cfb_team_factor	<i>Order CFB teams as a factor</i>
-----------------	------------------------------------

Description

Creates a factor of cleaned, valid CFB team names. Levels are restricted to teams present in `valid_team_names()` so that downstream ggplot2 scales drop invalid entries gracefully.

Usage

```
cfb_team_factor(teams)
```

Arguments

`teams` character vector of team names.

Value

An ordered factor of the cleaned, valid team names.

Examples

```
cfb_team_factor(c("Georgia", "Alabama", "ohio state"))
```

cfb_team_tiers	<i>Create CFB Team Tiers</i>
----------------	------------------------------

Description

This function sets up a ggplot to visualize CFB team tiers.

Usage

```
cfb_team_tiers(
  data,
  title = "CFB Team Tiers, 2021 as of Week 6",
  subtitle = "created with the #cfbplotR Tiermaker",
  caption = NULL,
  tier_desc = c(`1` = "Playoff", `2` = "Very Good", `3` = "Medium", `4` = "Bad", `5` =
    "What are they doing?", `6` = "", `7` = ""),
  presort = FALSE,
  alpha = 0.8,
  width = 0.075,
  no_line_below_tier = NULL,
  devel = FALSE
)
```

Arguments

<code>data</code>	A data frame that has to include the variables <code>tier_no</code> (the number of the tier starting from the top tier no. 1) and <code>team</code> (the team name). <code>team</code> should be one of <code>valid_team_names()</code> . If data includes the variable <code>tier_rank</code> , these ranks will be used within each tier. Otherwise, if <code>presort = FALSE</code> , the function will assume that data is already sorted and if <code>presort = TRUE</code> , teams will be sorted alphabetically within tiers.
<code>title</code>	The title of the plot. If <code>NULL</code> , it will be omitted.
<code>subtitle</code>	The subtitle of the plot. If <code>NULL</code> , it will be omitted.
<code>caption</code>	The caption of the plot. If <code>NULL</code> , it will be omitted.
<code>tier_desc</code>	A named vector consisting of the tier descriptions. The names must equal the tier numbers from <code>tier_no</code>
<code>presort</code>	If <code>FALSE</code> (the default) the function assumes that the teams are already sorted within the tiers. Will otherwise sort alphabetically.
<code>alpha</code>	The alpha channel of the logos, i.e. transparency level, as a numerical value between 0 and 1.
<code>width</code>	The desired width of the logo in npc (Normalised Parent Coordinates).
<code>no_line_below_tier</code>	Vector of tier numbers. The function won't draw tier separation lines below these tiers. This is intended to be used for tiers that shall be combined (see examples).
<code>devel</code>	Determines if logos shall be rendered. If <code>FALSE</code> (the default), logos will be rendered on each run. If <code>TRUE</code> the team names will be plotted instead of the logos. This is much faster and helps with the plot development.

Examples

```
if (requireNamespace("sjmisc", quietly = TRUE)) {
  library(ggplot2)
  library(dplyr, warn.conflicts = FALSE)
  team_names <- valid_team_names("FBS")
  # remove conference logos from this example
  team_names <- team_names[1:30]

  # Build the team tiers data frame
  # This is completely random!
  df <- data.frame(
    tier_no = sample(1:5, length(team_names), replace = TRUE),
    team = team_names
  ) %>%
  dplyr::group_by(tier_no) %>%
  dplyr::mutate(tier_rank = sample(1:n(), n()))

  # Plot team tiers
  cfb_team_tiers(df)

  # Create a combined tier which is useful for tiers with lots of teams that
```

```

# should be split up in two or more rows. This is done by setting an empty
# string for the tier 5 description and removing the tier separation line
# below tier number 4.
# This example also shows how to turn off the subtitle and add a caption
cfb_team_tiers(df,
  subtitle = NULL,
  caption = "This is the caption",
  tier_desc = c("1" = "Playoff",
               "2" = "Very Good",
               "3" = "Medium",
               "4" = "A Combined Tier",
               "5" = ""),
  no_line_below_tier = 4)

# For the development of the tiers, it can be useful to turn off logo image
# rendering as this can take quite a long time. By setting `devel = TRUE`, the
# logo images are replaced by team abbreviations which is much faster
cfb_team_tiers(df,
  tier_desc = c("1" = "Playoff",
               "2" = "Very Good",
               "3" = "",
               "4" = "A Combined Tier",
               "5" = ""),
  no_line_below_tier = c(2, 4),
  devel = TRUE)
}

```

clean_school_names	<i>Standardize NCAA School Names</i>
--------------------	--------------------------------------

Description

This function standardizes college names to cfbplotR defaults. This helps for joins and plotting.

Usage

```
clean_school_names(school, keep_non_matches = TRUE)
```

Arguments

school	a character vector of names
keep_non_matches	If TRUE (the default) an element of school that can't be matched to any of the internal mapping vectors will be kept as is. Otherwise it will be replaced with NA.

Value

A character vector with the length of school and cleaned team abbreviations if they are included in team_name_mapping. Non matches may be replaced with NA (depending on the value of keep_non_matches).

Examples

```
x <- c("utah", "San Jose State", "Hawaii", "UTSA", "SLC", "USC")
# use current location and keep non matches
clean_school_names(x)

# replace non matches
clean_school_names(x, keep_non_matches = FALSE)
```

clean_team_abbrs	<i>Clean CFB team abbreviations</i>
------------------	-------------------------------------

Description

Alias of `clean_school_names()` for naming parity with the `nflplotR`/`nbaplotR` family.

Usage

```
clean_team_abbrs(school, keep_non_matches = TRUE)
```

Arguments

- `school` a character vector of names.
- `keep_non_matches` If TRUE (the default) an element of `school` that can't be matched to any of the internal mapping vectors will be kept as is. Otherwise it will be replaced with NA.

Value

A character vector of cleaned team names.

data	Data in the package for reference
------	--

Description

- A dataset containing the school names, colors & logos for all college teams (NCAA DI/DII/DIII and FBS/FCS).
- A dataset mapping from abbreviation or alternate name forms to most commonly used.

Usage

```
logo_ref

team_name_mapping
```

Format

A data frame with 830+ rows and 6 variables:

school School name

type Conference-tier (i.e. P5, G5, etc.)

logo primary school logo from ESPN.com

color current primary school color

alt_color current secondary school color

wordmark Wordmark for school if available

A vector with 1100+ name variations

Short Name to Full Name

element

Theme Elements for Image Grobs

Description

In conjunction with the [ggplot2::theme](#) system, the following `element_` functions enable images in non-data components of the plot, e.g. axis text.

- `element_cfb_logo()`: draws college team logos instead of their names.
- `element_cfb_wordmark()`: draws college team wordmarks instead of their names.
- `element_cfb_headshot()`: draws player headshots instead of their ESPN player IDs.
- `element_path()`: draws images from valid image URLs instead of the URL.

Usage

```
element_cfb_logo(
  alpha = NULL,
  colour = NA,
  hjust = NULL,
  vjust = NULL,
  color = NULL,
  size = 0.5
)
```

```
element_cfb_wordmark(
  alpha = NULL,
  colour = NA,
  hjust = NULL,
  vjust = NULL,
  color = NULL,
  size = 0.5
)
```

```
)

element_cfb_headshot(
  alpha = NULL,
  colour = NA,
  hjust = NULL,
  vjust = NULL,
  color = NULL,
  size = 0.5
)

## S3 method for class 'element_cfb_logo'
element_grob(
  element,
  label = "",
  x = NULL,
  y = NULL,
  alpha = NULL,
  colour = NULL,
  hjust = NULL,
  vjust = NULL,
  size = NULL,
  ...
)

## S3 method for class 'element_cfb_wordmark'
element_grob(
  element,
  label = "",
  x = NULL,
  y = NULL,
  alpha = NULL,
  colour = NULL,
  hjust = NULL,
  vjust = NULL,
  size = NULL,
  ...
)

## S3 method for class 'element_cfb_headshot'
element_grob(
  element,
  label = "",
  x = NULL,
  y = NULL,
  alpha = NULL,
  colour = NULL,
  hjust = NULL,
```

```

    vjust = NULL,
    size = NULL,
    ...
  )

```

Arguments

alpha	The alpha channel, i.e. transparency level, as a numerical value between 0 and 1.
colour, color	The image will be colorized with this color. Use the special character "b/w" to set it to black and white. For more information on valid color names in ggplot2 see https://ggplot2.tidyverse.org/articles/ggplot2-specs.html?q=colour#colour-and-fill .
hjust, vjust	The horizontal and vertical adjustment respectively. Must be a numerical value between 0 and 1.
size	The output grob size in cm (!).
element	element type
label	Label
x	x grid units position
y	y grid units position
...	Other arguments to be passed to underlying ggplot function

Details

The elements translate CFB team names or players' ESPN IDs into logo images or player headshots, respectively.

Value

An S3 object of class `element`.

See Also

[geom_cfb_logos\(\)](#), [geom_cfb_wordmarks\(\)](#), [geom_cfb_headshots\(\)](#), and [geom_from_path\(\)](#) for more information on valid team names, player IDs, and other parameters.

Examples

```

library(cfbplotR)
library(ggplot2)

team_abbr <- valid_team_names()
# remove conference logos from this example
team_abbr <- team_abbr[1:8]

df <- data.frame(
  random_value = runif(length(team_abbr), 0, 1),
  teams = team_abbr

```

```

)

# use logos for x-axis
ggplot(df, aes(x = teams, y = random_value)) +
  geom_col(aes(color = teams, fill = teams), width = 0.5) +
  scale_color_cfb(alt_colors = team_abbr) +
  scale_fill_cfb(alpha = 0.4) +
  theme_minimal() +
  theme(axis.text.x = element_cfb_logo())

# use logos for y-axis
ggplot(df, aes(y = teams, x = random_value)) +
  geom_col(aes(color = teams, fill = teams), width = 0.5) +
  scale_color_cfb(alt_colors = team_abbr) +
  scale_fill_cfb(alpha = 0.4) +
  theme_minimal() +
  theme(axis.text.y = element_cfb_logo())

#####
# Headshot Examples
#####
library(cfbplotR)
library(ggplot2)

# Silence an nflreadr message that is irrelevant here
old <- options(nflreadr.cache_warning = FALSE)

dfh <- data.frame(
  random_value = runif(9, 0, 1),
  player_id = c("4361182",
                "4426385",
                "4567048",
                "4372519",
                "4429013",
                "4240069",
                "4360932",
                "4362874",
                "4429299")
)

# use headshots for x-axis
ggplot(dfh, aes(x = player_id, y = random_value)) +
  geom_col(width = 0.5) +
  theme_minimal() +
  theme(axis.text.x = element_cfb_headshot())

# use headshots for y-axis
ggplot(dfh, aes(y = player_id, x = random_value)) +
  geom_col(width = 0.5) +
  theme_minimal() +
  theme(axis.text.y = element_cfb_headshot())
#####
# Wordmarks and other Images

```

```
#####

library(ggplot2)

df <- dplyr::mutate(mtcars,
  team = sample(c("Utah", "Arizona State", "Oregon", "UCLA"), nrow(mtcars), TRUE),
  player = sample(
    c("4361182", "4426385", "4567048", "4429013"),
    nrow(mtcars),
    TRUE
  )
)

ggplot(df, aes(x = mpg, y = disp)) +
  geom_point() +
  facet_wrap(vars(team)) +
  labs(
    title = tools::toTitleCase("These are random teams and data"),
    subtitle = "I just want to show how the cfbplotR theme elements work",
    caption =
      paste0("https://raw.githubusercontent.com/sportsdataverse/cfbplotR/",
        "main/man/figures/logo.png")
  ) +
  theme_minimal() +
  theme(
    plot.title.position = "plot",
    plot.title = element_text(face = "bold"),
    axis.title = element_blank(),
    # make wordmarks of team abbreviations
    strip.text = element_cfb_wordmark(size = 1),
    # load image from url in caption
    plot.caption = element_path(hjust = 1, size = 0.4)
  )

```

geom_cfb_headshots

ggplot2 Layer for Visualizing CFB Player Headshots

Description

This geom is used to plot CFB player headshots instead of points in a ggplot. It requires x, y aesthetics as well as a valid ESPN player id from cfbfastR

Usage

```
geom_cfb_headshots(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",

```

```

    ...,
    na.rm = FALSE,
    show.legend = FALSE,
    inherit.aes = TRUE
  )

```

Arguments

mapping	Set of aesthetic mappings created by aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to ggplot(). A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
stat	The statistical transformation to use on the data for this layer. When using a <code>geom_*()</code> function to construct a layer, the <code>stat</code> argument can be used to override the default coupling between geoms and stats. The <code>stat</code> argument accepts the following: • A Stat ggproto subclass, for example <code>StatCount</code>. • A string naming the stat. To give the stat as a string, strip the function name of the <code>stat_</code> prefix. For example, to use <code>stat_count()</code>, give the stat as <code>"count"</code>. • For more information and other ways to specify the stat, see the layer stat documentation.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following: • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the <code>position_</code> prefix. For example, to use <code>position_jitter()</code>, give the position as <code>"jitter"</code>. • For more information and other ways to specify the position, see the layer position documentation.
...	Other arguments passed on to ggplot2::layer() . These are often aesthetics, used to set an aesthetic to a fixed value. See the below section "Aesthetics" for a full list of possible arguments.
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.

<code>show.legend</code>	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
<code>inherit.aes</code>	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. <code>annotation_borders()</code> .

Aesthetics

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

- **x** - The x-coordinate.
- **y** - The y-coordinate.
- **player_id** - The players' ESPN id.
- `alpha = NULL` - The alpha channel, i.e. transparency level, as a numerical value between 0 and 1.
- `colour = NULL` - The image will be colored with this colour. Use the special character "b/w" to set it to black and white. For more information on valid colour names in ggplot2 see <https://ggplot2.tidyverse.org/articles/ggplot2-specs.html?q=colour#colour-and-fill>
- `angle = 0` - The angle of the image as a numerical value between 0° and 360°.
- `hjust = 0.5` - The horizontal adjustment relative to the given x coordinate. Must be a numerical value between 0 and 1.
- `vjust = 0.5` - The vertical adjustment relative to the given y coordinate. Must be a numerical value between 0 and 1.
- `width = 1.0` - The desired width of the image in npc (Normalised Parent Coordinates). The default value is set to 1.0 which is *big* but it is necessary because all used values are computed relative to the default. A typical size is `width = 0.075` (see below examples).
- `height = 1.0` - The desired height of the image in npc (Normalised Parent Coordinates). The default value is set to 1.0 which is *big* but it is necessary because all used values are computed relative to the default. A typical size is `height = 0.1` (see below examples).

Examples

```
## Not run:
library(cfbplotR)
library(ggplot2)

df <- data.frame(
  a = rep(1:3, 3),
  b = sort(rep(1:3, 3), decreasing = TRUE),
  player_id = c("3926231",
                "4362874",
                "4239086",
                "4685720",
```

```

      "4374303",
      "4361741",
      "4362075",
      "1",
      "4426348"),
  player_name = c("Britain Covey",
                  "Matt Corral",
                  "Desmond Ridder",
                  "Bryce Young",
                  "JT Daniels",
                  "Brock Purdy",
                  "Cameron Rising",
                  "Non.Match",
                  "Jayden Daniels")
)

# set a custom fill colour for one player
df$colour <- ifelse(df$a == 2 & df$b == 2, NA, "b/w")

# scatterplot of the headshots
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_headshots(aes(player_id = player_id), height = 0.2) +
  geom_label(aes(label = player_name), nudge_y = -0.35, alpha = 0.5) +
  coord_cartesian(xlim = c(0.75, 3.25), ylim = c(0.7, 3.25)) +
  theme_void()

# apply alpha as constant
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_headshots(aes(player_id = player_id), height = 0.2, alpha = 0.5) +
  geom_label(aes(label = player_name), nudge_y = -0.35, alpha = 0.5) +
  coord_cartesian(xlim = c(0.75, 3.25), ylim = c(0.7, 3.25)) +
  theme_void()

# apply colour as an aesthetic
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_headshots(aes(player_id = player_id, colour = colour), height = 0.2) +
  geom_label(aes(label = player_name), nudge_y = -0.35, alpha = 0.5) +
  coord_cartesian(xlim = c(0.75, 3.25), ylim = c(0.7, 3.25)) +
  scale_colour_identity() +
  theme_void()

## End(Not run)

```

Description

This geom is used to plot college team wordmarks instead of points in a ggplot. It requires x, y aesthetics as well as a valid school name. The latter can be checked with [valid_team_names\(\)](#).

Usage

```
geom_cfb_wordmarks(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  na.rm = FALSE,
  show.legend = FALSE,
  inherit.aes = TRUE
)
```

Arguments

- | | |
|----------|---|
| mapping | Set of aesthetic mappings created by aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. |
| data | <p>The data to be displayed in this layer. There are three options:</p> <p>If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to ggplot().</p> <p>A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created.</p> <p>A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).</p> |
| stat | <p>The statistical transformation to use on the data for this layer. When using a <code>geom_*()</code> function to construct a layer, the <code>stat</code> argument can be used to override the default coupling between geoms and stats. The <code>stat</code> argument accepts the following:</p> <ul style="list-style-type: none"> • A Stat ggproto subclass, for example <code>StatCount</code>. • A string naming the stat. To give the stat as a string, strip the function name of the <code>stat_</code> prefix. For example, to use <code>stat_count()</code>, give the stat as <code>"count"</code>. • For more information and other ways to specify the stat, see the layer stat documentation. |
| position | <p>A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following:</p> <ul style="list-style-type: none"> • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the <code>position_</code> prefix. For example, to use <code>position_jitter()</code>, give the position as <code>"jitter"</code>. • For more information and other ways to specify the position, see the layer position documentation. |

...	Other arguments passed on to <code>ggplot2::layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value. See the below section "Aesthetics" for a full list of possible arguments.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. <code>annotation_borders()</code> .

Value

A ggplot2 layer (`ggplot2::layer()`) that can be added to a plot created with `ggplot2::ggplot()`.

Aesthetics

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

- **x** - The x-coordinate.
- **y** - The y-coordinate.
- **team** - The team name or abbreviation. Must be one of `valid_team_names()`.
- `alpha = NULL` - The alpha channel, i.e. transparency level, as a numerical value between 0 and 1.
- `colour = NULL` - The image will be colored with this colour. Use the special character "b/w" to set it to black and white. For more information on valid colour names in ggplot2 see <https://ggplot2.tidyverse.org/articles/ggplot2-specs.html?q=colour#colour-and-fill>
- `angle = 0` - The angle of the image as a numerical value between 0° and 360°.
- `hjust = 0.5` - The horizontal adjustment relative to the given x coordinate. Must be a numerical value between 0 and 1.
- `vjust = 0.5` - The vertical adjustment relative to the given y coordinate. Must be a numerical value between 0 and 1.
- `width = 1.0` - The desired width of the image in npc (Normalised Parent Coordinates). The default value is set to 1.0 which is *big* but it is necessary because all used values are computed relative to the default. A typical size is `width = 0.1` (see below examples).
- `height = 1.0` - The desired height of the image in npc (Normalised Parent Coordinates). The default value is set to 1.0 which is *big* but it is necessary because all used values are computed relative to the default. A typical size is `height = 0.1` (see below examples).

Examples

```
library(cfbplotR)
library(ggplot2)

teams <- valid_team_names("P5")[1:32]

df <- data.frame(
  a = rep(1:8, 4),
  b = sort(rep(1:4, 8), decreasing = TRUE),
  teams = teams
)

# keep alpha == 1 for all teams including an "A"
matches <- grepl("A", teams)
df$alpha <- ifelse(matches, 1, 0.2)
# also set a custom fill colour for the non "A" teams
df$colour <- ifelse(matches, NA, "gray")

# scatterplot of all wordmarks
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_wordmarks(aes(team = teams), width = 0.12) +
  geom_label(aes(label = teams), nudge_y = -0.20, alpha = 0.5) +
  scale_x_continuous(expand = expansion(add = 0.5)) +
  theme_void()

# apply alpha via an aesthetic from inside the dataset `df`
# please note that you have to add scale_alpha_identity() to use the alpha
# values in your dataset!
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_wordmarks(aes(team = teams, alpha = alpha), width = 0.12) +
  geom_label(aes(label = teams), nudge_y = -0.20, alpha = 0.5) +
  scale_x_continuous(expand = expansion(add = 0.5)) +
  scale_alpha_identity() +
  theme_void()

# apply alpha and colour via an aesthetic from inside the dataset `df`
# please note that you have to add scale_alpha_identity() as well as
# scale_color_identity() to use the alpha and colour values in your dataset!
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_wordmarks(aes(team = teams, alpha = alpha, colour = colour), width = 0.12) +
  geom_label(aes(label = teams), nudge_y = -0.20, alpha = 0.5) +
  scale_x_continuous(expand = expansion(add = 0.5)) +
  scale_alpha_identity() +
  scale_color_identity() +
  theme_void()

# apply alpha as constant for all logos
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_wordmarks(aes(team = teams), width = 0.12, alpha = 0.6) +
  geom_label(aes(label = teams), nudge_y = -0.20, alpha = 0.5) +
  scale_x_continuous(expand = expansion(add = 0.5)) +
  theme_void()
```

Description

This geom is used to plot college football team and conference logos instead of points in a ggplot. It requires x, y aesthetics as well as a valid CFB team name or abbreviation. The latter can be checked with [valid_team_names\(\)](#).

Usage

GeomCFBheads

```
geom_cfb_logos(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  na.rm = FALSE,
  show.legend = FALSE,
  inherit.aes = TRUE
)
```

GeomCFBlogo

Arguments

mapping	Set of aesthetic mappings created by aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to ggplot(). A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
stat	The statistical transformation to use on the data for this layer. When using a <code>geom_*()</code> function to construct a layer, the <code>stat</code> argument can be used to override the default coupling between geoms and stats. The <code>stat</code> argument accepts the following:

	• A Stat ggproto subclass, for example StatCount. • A string naming the stat. To give the stat as a string, strip the function name of the <code>stat_</code> prefix. For example, to use <code>stat_count()</code>, give the stat as "count". • For more information and other ways to specify the stat, see the layer stat documentation.
<code>position</code>	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following: • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the <code>position_</code> prefix. For example, to use <code>position_jitter()</code>, give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
<code>...</code>	Other arguments passed on to <code>ggplot2::layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value. See the below section "Aesthetics" for a full list of possible arguments.
<code>na.rm</code>	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
<code>show.legend</code>	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
<code>inherit.aes</code>	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. <code>annotation_borders()</code> .

Format

An object of class `GeomCFBheads` (inherits from `Geom`, `ggproto`, `gg`) of length 5.

An object of class `GeomCFBlogo` (inherits from `Geom`, `ggproto`, `gg`) of length 5.

Aesthetics

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

- **x** - The x-coordinate.
- **y** - The y-coordinate.
- **team** - The team name or abbreviation. Must be one of `valid_team_names()`.
- `alpha = NULL` - The alpha channel, i.e. transparency level, as a numerical value between 0 and 1.
- `colour = NULL` - The colour of the image. Managed with `scale_color_*`

- `angle = 0` - The angle of the image as a numerical value between 0° and 360° .
- `hjust = 0.5` - The horizontal adjustment relative to the given x coordinate. Must be a numerical value between 0 and 1.
- `vjust = 0.5` - The vertical adjustment relative to the given y coordinate. Must be a numerical value between 0 and 1.
- `width = 1.0` - The desired width of the image in npc (Normalised Parent Coordinates). The default value is set to 1.0 which is *big* but it is necessary because all used values are computed relative to the default. A typical size is `width = 0.075` (see below examples).
- `height = 1.0` - The desired height of the image in npc (Normalised Parent Coordinates). The default value is set to 1.0 which is *big* but it is necessary because all used values are computed relative to the default. A typical size is `height = 0.1` (see below examples).

Examples

```
library(cfbplotR)
library(ggplot2)

team <- valid_team_names()
team <- team[1:32]

df <- data.frame(
  a = rep(1:8, 4),
  b = sort(rep(1:4, 8), decreasing = TRUE),
  teams = team
)

# keep alpha == 1 for all teams including an "A"
matches <- grepl("A", team)
df$alpha <- ifelse(matches, 1, 0.2)

# scatterplot of all logos
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_logos(aes(team = teams), width = 0.075) +
  geom_label(aes(label = teams), nudge_y = -0.35, alpha = 0.5) +
  theme_void()

# apply alpha via an aesthetic from inside the dataset `df`
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_logos(aes(team = teams, alpha = alpha), width = 0.075) +
  geom_label(aes(label = teams), nudge_y = -0.35, alpha = 0.5) +
  scale_alpha_identity() +
  theme_void()

# apply alpha as constant for all logos
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_logos(aes(team = teams), width = 0.075, alpha = 0.6) +
  geom_label(aes(label = teams), nudge_y = -0.35, alpha = 0.5) +
  theme_void()

# it's also possible to plot conference logos
conf <- data.frame(a = 1:2, b = 0, teams = c("Pac-12", "ACC"))
```

```
ggplot(conf, aes(x = a, y = b)) +
  geom_cfb_logos(aes(team = teams), width = 0.3) +
  geom_label(aes(label = teams), nudge_y = -0.4, alpha = 0.5) +
  coord_cartesian(xlim = c(0.5,2.5), ylim = c(-0.75,.75)) +
  theme_void()

# it's also possible to color the logos
# make teams with an A red
df$color <- ifelse(matches, "red", NA)
ggplot(df, aes(x = a, y = b)) +
  geom_cfb_logos(aes(team = teams, color = color), width = 0.075) +
  geom_label(aes(label = teams), nudge_y = -0.35, alpha = 0.5) +
  scale_color_identity() +
  theme_void()
```

ggpreview

Preview ggplot in Specified Dimensions

Description

This function previews a ggplot in its actual dimensions in order to see how it will look when saved. It is also significantly faster than the default preview in RStudio for ggplots created using cfbplotR. This is copied directly from nflplotR

Usage

```
ggpreview(
  plot = ggplot2::last_plot(),
  width = NA,
  height = NA,
  asp = NULL,
  dpi = 300,
  device = "png",
  units = c("in", "cm", "mm", "px"),
  scale = 1,
  limitsize = TRUE,
  bg = NULL,
  ...
)
```

Arguments

plot	Plot to save, defaults to last plot displayed.
width, height	Plot size in units expressed by the units argument. If not supplied, uses the size of the current graphics device.

asp	The aspect ratio of the plot calculated as width / height. If this is a numeric value (and not NULL) the height of the plot will be recalculated to height = width / asp.
dpi	Plot resolution. Also accepts a string input: "retina" (320), "print" (300), or "screen" (72). Only applies when converting pixel units, as is typical for raster output types.
device	Device to use. Can either be a device function (e.g. png), or one of "eps", "ps", "tex" (pictex), "pdf", "jpeg", "tiff", "png", "bmp", "svg" or "wmf" (windows only). If NULL (default), the device is guessed based on the filename extension.
units	One of the following units in which the width and height arguments are expressed: "in", "cm", "mm" or "px".
scale	Multiplicative scaling factor.
limitsize	When TRUE (the default), ggsave() will not save images larger than 50x50 inches, to prevent the common error of specifying dimensions in pixels.
bg	Background colour. If NULL, uses the plot.background fill value from the plot theme.
...	Other arguments passed on to the graphics device function, as specified by device.

Examples

```
library(cfbplotR)
library(ggplot2)

team <- valid_team_names()
# remove conference logos from this example
team <- team[1:20]

df <- data.frame(
  random_value = runif(length(team), 0, 1),
  teams = team
)

# use logos for x-axis
# note that the plot is assigned to the object "p"
p <- ggplot(df, aes(x = teams, y = random_value)) +
  geom_col(aes(color = teams, fill = teams), width = 0.5) +
  scale_color_cfb(alt_colors = valid_team_names()) +
  scale_fill_cfb(alpha = 0.7) +
  scale_x_cfb() +
  theme_minimal() +
  theme_x_cfb()

# preview p with defined width and aspect ratio (only available in RStudio)
if (rstudioapi::isAvailable()){
  ggpreview(p, width = 5, asp = 16/9)
}
```

ggtitle_image

*Functions for adding an image to the title of a ggplot***Description**

These functions work together to place an image to the left or right of the title in a ggplot. [ggtitle_image](#) is the main function but must be used with either [theme_title_image\(\)](#) or setting the `plot.title` argument in [ggplot2::theme\(\)](#) to [ggtext::element_markdown\(\)](#)

Usage

```
ggtitle_image(
  title_image = ggplot2::waiver(),
  title = ggplot2::waiver(),
  image_height = 15,
  image_side = c("left", "right"),
  subtitle = ggplot2::waiver()
)

theme_title_image(...)
```

Arguments

<code>title_image</code>	The URL of the image to add to the title. If one of valid_team_names() , the school logo will be used automatically.
<code>title</code>	The text for the title. The title has to be defined here as the function overwrites any current title text.
<code>image_height</code>	The height of the image in pixels.
<code>image_side</code>	One of "left" or "right". Places the image on either side of the title text.
<code>subtitle</code>	Optional text for the subtitle. Does not overwrite current subtitle if left blank.
<code>...</code>	Other arguments passed on to ggtext::element_markdown() .

See Also

[ggtitle_image\(\)](#), [theme_title_image\(\)](#)

Examples

```
library(cfbplotR)
library(ggplot2)

logo_url <- "https://raw.githubusercontent.com/sportsdataverse/cfbplotR/main/man/figures/logo.png"

p <- ggplot(mtcars, aes(x = hp, y = mpg)) +
  geom_point() +
  labs(title = "This Title will be overwritten",
```

```

      subtitle = "This is the Subtitle")

if (utils::packageVersion("gridtext") > "0.1.4"){
  p +
    ggtitle_image(title_image = logo_url,
                  title = "This is the Title",
                  image_height = 40,
                  image_side = "right") +
    theme_title_image(size = 20, hjust = 0.5)
}
if (utils::packageVersion("gridtext") > "0.1.4"){
  p +
    ggtitle_image(title_image = "Utah",
                  title = "This it the Title",
                  image_height = 20,
                  image_side = "left",
                  subtitle = "This overwrites the old subtitle") +
    theme(plot.title = ggtext::element_markdown(size = 20, hjust = 0.5))
}

```

gt_cfb

Add logos into rows of a gt table

Description

The `gt_fmt_cfb_logo` and `gt_fmt_cfb_headshot` functions take an existing `gt_tbl` object and converts college football team names from `valid_team_names()` into inline team logos or ESPN player ID's (or `headshot_url` from `cfbfastR::cfbd_team_rosters()` function) into inline player headshots. This is a wrapper around `gtExtras::gt_image_rows()` written by Tom Mock, which is a wrapper around `gt::text_transform()` + `gt::web_image()/gt::local_image()` with the necessary boilerplate already applied.

Usage

```
gt_fmt_cfb_logo(gt_object, columns, height = 30)
```

```
gt_fmt_cfb_wordmark(gt_object, columns, height = 30)
```

```
gt_fmt_cfb_headshot(gt_object, columns, height = 30)
```

Arguments

<code>gt_object</code>	An existing gt table object of class <code>gt_tbl</code>
<code>columns</code>	The columns wherein changes to cell data colors should occur.
<code>height</code>	<i>Height of image</i> scalar<numeric integer> // default: 30 The absolute height of the image in the table cell (in "px" units). By default, this is set to "30px".

Value

An object of class gt_tbl.

Figures

Examples

```
library(gt)
library(cfbplotR)

df <- data.frame(team = valid_team_names()[1:8],
                 logo = valid_team_names()[1:8],
                 wordmark = valid_team_names()[1:8])

table <- df %>%
  gt() %>%
  gt_fmt_cfb_logo(columns = "logo") %>%
  gt_fmt_cfb_wordmark(columns = "wordmark")

df <- data.frame(
  player = c("Britain Covey", "Cameron Rising", "Non.Match"),
  team = c("Utah", "Utah", "BYU")
) %>%
  add_athlete_id_col(player)
table_2 <- df %>%
  gt() %>%
  gt_fmt_cfb_headshot(athlete_id)
```

gt_cfb_cols_label	<i>Render CFB logos in gt column labels</i>
-------------------	---

Description

Replace gt column label text with CFB team logos. Each bare column name passed via ... is treated as a team name; the label for that column is replaced with the corresponding logo image.

Usage

```
gt_cfb_cols_label(gt_object, ..., height = 30)
```

Arguments

gt_object	a gt_tbl.
...	<tidy-select>-style bare column names whose labels are team names to replace with logos.
height	image height in px.

Value

A `gt_tbl`.

Examples

```
library(gt)
library(cfbplotR)

df <- data.frame(Georgia = 1:3, Alabama = 4:6)
gt(df) |> gt_cfb_cols_label(Georgia, Alabama)
```

<code>gt_stack_team</code>	<i>Merge and stack text from two columns in <code>gt</code> and color one with school colors</i>
----------------------------	--

Description

The `gt_merge_stack_team_color()` function takes an existing `gt` table and merges column 1 and column 2, stacking column 1's text on top of column 2's. Top text is in all caps with black bold text, while the lower text is smaller and colored by the team name in another column. This is a slightly modified version of `gtExtras::gt_merge_stack()` written by Tom Mock.

Usage

```
gt_merge_stack_team_color(
  gt_object,
  col1,
  col2,
  team_col,
  font_size_top = 14,
  font_size_bottom = 12,
  color = "black"
)
```

Arguments

<code>gt_object</code>	An existing <code>gt</code> table object of class <code>gt_tbl</code>
<code>col1</code>	The column to stack on top. Will be converted to all caps, with black and bold text.
<code>col2</code>	The column to merge and place below. Will be smaller and the school color that corresponds to <code>team_col</code> .
<code>team_col</code>	The column of team names that match <code>valid_team_names()</code> for the color of the bottom.
<code>font_size_top</code>	the font size for the top text.
<code>font_size_bottom</code>	the font size for the bottom text.
<code>color</code>	The color for the top text.

Value

An object of class `gt_tbl`.

Figures**Examples**

```
library(gt)
teams <-
  "https://github.com/sportsdataverse/cfbfastR-data/raw/main/team_info/rds/cfb_team_info_2020.rds"
team_df <- readRDS(url(teams))

stacked_tab <- team_df %>%
  dplyr::transmute(logo = school,school,mascot,conference,city,state) %>%
  head(8) %>%
  gt::gt() %>%
  gt_merge_stack_team_color(school,mascot,school) %>%
  gt_fmt_cfb_logo(columns = c("logo","conference"))
```

scale_axes_cfb

Axis Scales for CFB Team Logos

Description

These functions map CFB team names to their team logos and make them available as axis labels

Usage

```
scale_x_cfb(
  ...,
  expand = ggplot2::waiver(),
  guide = ggplot2::waiver(),
  position = "bottom",
  size = 12
)

scale_y_cfb(
  ...,
  expand = ggplot2::waiver(),
  guide = ggplot2::waiver(),
  position = "left",
  size = 12
)

scale_x_cfb_headshots(
```

```

    ...,
    expand = ggplot2::waiver(),
    guide = ggplot2::waiver(),
    position = "bottom",
    size = 20
)

scale_y_cfb_headshots(
  ...,
  expand = ggplot2::waiver(),
  guide = ggplot2::waiver(),
  position = "left",
  size = 30
)

```

Arguments

... Arguments passed on to [discrete_scale](#)

breaks One of:

- NULL for no breaks
- `waiver()` for the default breaks (the scale limits)
- A character vector of breaks
- A function that takes the limits as input and returns breaks as output. Also accepts rlang [lambda](#) function notation.

limits One of:

- NULL to use the default scale values
- A character vector that defines possible values of the scale and their order
- A function that accepts the existing (automatic) values and returns new ones. Also accepts rlang [lambda](#) function notation.

drop Should unused factor levels be omitted from the scale? The default, TRUE, uses the levels that appear in the data; FALSE includes the levels in the factor. Please note that to display every level in a legend, the layer should use `show.legend = TRUE`.

na.translate Unlike continuous scales, discrete scales can easily show missing values, and do so by default. If you want to remove missing values from a discrete scale, specify `na.translate = FALSE`.

na.value If `na.translate = TRUE`, what aesthetic value should the missing values be displayed as? Does not apply to position scales where NA is always placed at the far right.

aesthetics The names of the aesthetics that this scale works with.

minor_breaks One of:

- NULL for no minor breaks
- `waiver()` for the default breaks (none for discrete, one minor break between each major break for continuous)
- A numeric vector of positions

	• A function that given the limits returns a vector of minor breaks. Also accepts rlang <code>lambda</code> function notation. When the function has two arguments, it will be given the limits and major break positions.
<code>labels</code>	One of the options below. Please note that when <code>labels</code> is a vector, it is highly recommended to also set the <code>breaks</code> argument as a vector to protect against unintended mismatches. • <code>NULL</code> for no labels • <code>waiver()</code> for the default labels computed by the transformation object • A character vector giving labels (must be same length as <code>breaks</code>) • An expression vector (must be the same length as <code>breaks</code>). See <code>?plot-math</code> for details. • A function that takes the <code>breaks</code> as input and returns labels as output. Also accepts rlang <code>lambda</code> function notation.
<code>call</code>	The call used to construct the scale for reporting messages.
<code>super</code>	The super class to use for the constructed scale
<code>expand</code>	For position scales, a vector of range expansion constants used to add some padding around the data to ensure that they are placed some distance away from the axes. Use the convenience function <code>expansion()</code> to generate the values for the <code>expand</code> argument. The defaults are to expand the scale by 5% on each side for continuous variables, and by 0.6 units on each side for discrete variables.
<code>guide</code>	A function used to create a guide or its name. See <code>guides()</code> for more information.
<code>position</code>	For position scales, The position of the axis. <code>left</code> or <code>right</code> for y axes, <code>top</code> or <code>bottom</code> for x axes.
<code>size</code>	The logo size in pixels. It is applied as height for an x-scale and as width for an y-scale.

Details

The scale translates the CFB team names into raw image html and places the html as axis labels. Because of the way ggplots are constructed, it is necessary to adjust the `ggplot2::theme()` after calling this scale. This can be done by calling `theme_x_cfb()` or `theme_y_cfb()` or alternatively by manually changing the relevant `axis.text` to `ggtext::element_markdown()`.

See Also

`theme_x_cfb()`, `theme_y_cfb()`

Examples

```
library(ggplot2)

#####
# Headshot Example
#####
```

```
dfh <- data.frame(
  random_value = runif(9, 0, 1),
  player_id = c("4361182",
                "4426385",
                "4567048",
                "4372519",
                "4429013",
                "4240069",
                "4360932",
                "4362874",
                "4429299")
)

# use headshots for y-axis
ggplot(dfh, aes(y = player_id, x = random_value)) +
  geom_col(width = 0.5) +
  scale_y_cfb_headshots() +
  theme_minimal() +
  theme_y_cfb()
```

scale_color_cfb

Scale for college football team colors

Description

These functions allows you to map college football team names as levels to the color and fill aesthetics

Usage

```
scale_color_cfb(
  alt_colors = NULL,
  values = NULL,
  ...,
  aesthetics = "colour",
  breaks = ggplot2::waiver(),
  na.value = "grey50",
  guide = NULL,
  alpha = NA
)
```

```
scale_colour_cfb(
  alt_colors = NULL,
  values = NULL,
  ...,
  aesthetics = "colour",
  breaks = ggplot2::waiver(),
```

```

    na.value = "grey50",
    guide = NULL,
    alpha = NA
  )

scale_fill_cfb(
  alt_colors = NULL,
  values = NULL,
  ...,
  aesthetics = "fill",
  breaks = ggplot2::waiver(),
  na.value = "grey50",
  guide = NULL,
  alpha = NA
)

```

Arguments

<code>alt_colors</code>	Vector of team names to use the alternate color of.
<code>values</code>	If <code>NULL</code> (the default) use the internal team color vectors. Otherwise a set of aesthetic values to map data values to. The values will be matched in order (usually alphabetical) with the limits of the scale, or with breaks if provided. If this is a named vector, then the values will be matched based on the names instead. Data values that don't match will be given <code>na.value</code> .
<code>...</code>	Arguments passed on to <code>scale_color_manual</code>
<code>aesthetics</code>	Character string or vector of character strings listing the name(s) of the aesthetic(s) that this scale works with. This can be useful, for example, to apply colour settings to the colour and fill aesthetics at the same time, via <code>aesthetics = c("colour", "fill")</code> .
<code>breaks</code>	One of: • <code>NULL</code> for no breaks • <code>waiver()</code> for the default breaks (the scale limits) • A character vector of breaks • A function that takes the limits as input and returns breaks as output
<code>na.value</code>	The aesthetic value to use for missing (NA) values
<code>guide</code>	A function used to create a guide or its name. If <code>NULL</code> (the default) no guide will be plotted for this scale. See ggplot2::guides() for more information.
<code>alpha</code>	Factor to modify color transparency via a call to scales::alpha() . If <code>NA</code> (the default) no transparency will be applied. Can also be a vector of alphas. All alpha levels must be in range $[0, 1]$.

Examples

```

library(cfbplotR)
library(ggplot2)

```

```
df <- data.frame(
  y = 6:9,
  teams = c("Alabama", "Florida State", "Oregon", "Utah")
)
ggplot(df, aes(x = teams, y = y)) +
  geom_col(aes(color = teams, fill = teams), size = 2) +
  scale_color_cfb(alt_colors = df$teams) +
  scale_fill_cfb() +
  theme_minimal()
```

theme_cfb

Theme for CFB Team Logos

Description

These functions are convenience wrappers around a theme call that activates markdown in x-axis and y-axis labels using `ggtext::element_markdown()`.

Usage

```
theme_x_cfb()
```

```
theme_y_cfb()
```

Details

These functions are a wrapper around the function calls `ggplot2::theme(axis.text.x = ggtext::element_markdown())` as well as `ggplot2::theme(axis.text.y = ggtext::element_markdown())`. They are made to be used in conjunction with `scale_x_cfb()` and `scale_y_cfb()` respectively.

See Also

`theme_x_cfb()`, `theme_y_cfb()`

Examples

```
library(cfbplotR)
library(ggplot2)

team_abbr <- valid_team_names()
# remove conference logos from this example
team_abbr <- team_abbr[1:16]

df <- data.frame(
  random_value = runif(length(team_abbr), 0, 1),
  teams = team_abbr
)

if (utils::packageVersion("gridtext") > "0.1.4"){
```

```

ggplot(df, aes(x = teams, y = random_value)) +
  geom_col(aes(color = teams, fill = teams), width = 0.5) +
  scale_color_cfb(alt_colors = team_abbr) +
  scale_fill_cfb(alpha = 0.4) +
  scale_x_cfb() +
  theme_minimal() +
  # theme_*_cfb requires gridtext version > 0.1.4
  theme_x_cfb()
}

```

valid_team_names

Output Valid CFB Team Names and Abbreviations

Description

Output Valid CFB Team Names and Abbreviations

Usage

```

valid_team_names(
  division = c("FBS", "P5", "G5", "FCS", "DII", "DIII", "Conference", "hoopR", "Other")
)

```

Arguments

division	Division to filter for. Options include:
	• FBS • P5 • G5 • FCS • DII • DIII • Conference • hoopR • Other

Examples

```

# List valid team abbreviations excluding duplicates
valid_team_names("FBS")
valid_team_names("FCS")
valid_team_names("DII")
valid_team_names("DIII")
valid_team_names("Conference")

```

Index

- * **datasets**
 - GeomCFBheads, [19](#)
- * **data**
 - data, [7](#)
- .cfbplotR_clear_cache, [2](#)
- add_athlete_id_col, [3](#)
- aes(), [13](#), [16](#), [19](#)
- annotation_borders(), [14](#), [17](#), [20](#)
- cfb_team_factor, [4](#)
- cfb_team_tiers, [4](#)
- clean_school_names, [6](#)
- clean_school_names(), [7](#)
- clean_team_abbrs, [7](#)
- data, [7](#)
- discrete_scale, [29](#)
- element, [8](#)
- element_cfb_headshot(element), [8](#)
- element_cfb_logo(element), [8](#)
- element_cfb_wordmark(element), [8](#)
- element_grob.element_cfb_headshot(element), [8](#)
- element_grob.element_cfb_logo(element), [8](#)
- element_grob.element_cfb_wordmark(element), [8](#)
- expansion(), [30](#)
- fortify(), [13](#), [16](#), [19](#)
- geom_cfb_headshots, [12](#)
- geom_cfb_headshots(), [10](#)
- geom_cfb_logos(GeomCFBheads), [19](#)
- geom_cfb_logos(), [10](#)
- geom_cfb_wordmarks, [15](#)
- geom_cfb_wordmarks(), [10](#)
- geom_from_path(), [10](#)
- GeomCFBheads, [19](#)
- GeomCFBlogo(GeomCFBheads), [19](#)
- ggplot(), [13](#), [16](#), [19](#)
- ggplot2::ggplot(), [17](#)
- ggplot2::guides(), [32](#)
- ggplot2::layer(), [13](#), [17](#), [20](#)
- ggplot2::theme, [8](#)
- ggplot2::theme(), [24](#), [30](#)
- ggpreview, [22](#)
- ggtext::element_markdown(), [24](#), [30](#), [33](#)
- ggtitle_image, [24](#), [24](#)
- ggtitle_image(), [24](#)
- gt_cfb, [25](#)
- gt_cfb_cols_label, [26](#)
- gt_fmt_cfb_headshot(gt_cfb), [25](#)
- gt_fmt_cfb_logo(gt_cfb), [25](#)
- gt_fmt_cfb_wordmark(gt_cfb), [25](#)
- gt_merge_stack_team_color(gt_stack_team), [27](#)
- gt_stack_team, [27](#)
- guides(), [30](#)
- lambda, [29](#), [30](#)
- layer position, [13](#), [16](#), [20](#)
- layer stat, [13](#), [16](#), [20](#)
- logo_ref(data), [7](#)
- png, [23](#)
- scale_axes_cfb, [28](#)
- scale_color_cfb, [31](#)
- scale_colour_cfb(scale_color_cfb), [31](#)
- scale_fill_cfb(scale_color_cfb), [31](#)
- scale_x_cfb(scale_axes_cfb), [28](#)
- scale_x_cfb(), [33](#)
- scale_x_cfb_headshots(scale_axes_cfb), [28](#)
- scale_y_cfb(scale_axes_cfb), [28](#)
- scale_y_cfb(), [33](#)
- scale_y_cfb_headshots(scale_axes_cfb), [28](#)

`scales::alpha()`, [32](#)

`team_name_mapping` (data), [7](#)

`theme_cfb`, [33](#)

`theme_title_image` (ggtitle_image), [24](#)

`theme_title_image()`, [24](#)

`theme_x_cfb` (theme_cfb), [33](#)

`theme_x_cfb()`, [30](#), [33](#)

`theme_y_cfb` (theme_cfb), [33](#)

`theme_y_cfb()`, [30](#), [33](#)

`valid_team_names`, [34](#)

`valid_team_names()`, [4](#), [5](#), [15](#), [17](#), [19](#), [20](#), [24](#)