Package ‘worldfootballR’

November 26, 2022

Type Package
Title Extract and Clean World Football (Soccer) Data
Version 0.6.2
Description Allow users to obtain clean and tidy football (soccer) game, team and player data. Data is collected from a number of popular sites, including 'FBref', transfer and valuations data from 'Transfermarkt' and shooting location and other match stats data from 'Understat' and 'fotmob'. It gives users the ability to access data more efficiently, rather than having to export data tables to files before being able to complete their analysis.

License GPL-3
BugReports https://github.com/JaseZiv/worldfootballR/issues
Depends R (>= 4.0.0)
Imports dplyr, glue, htr, janitor, jsonlite, lubridate, magrittr, progress, purrr, qdapRegex, readr, rlang, rvest (>= 1.0.0), stats, stringi, stringr, tidyselect, utils, withr, xml2, tibble, cli, curl
Suggests knitr, rmarkdown, testthat
Encoding UTF-8
RoxygenNote 7.2.1
NeedsCompilation no
Author Jason Zivkovic [aut, cre, cph], Tony ElHabr [ctb], Tan Ho [ctb], Samuel H [ctb]
Maintainer Jason Zivkovic <jaseziv83@gmail.com>
Repository CRAN
Date/Publication 2022-11-26 04:00:02 UTC
R topics documented:

- `fb_advanced_match_stats` .......................................................... 3
- `fb_big5_advanced_season_stats` .................................................. 4
- `fb_league_urls` ................................................................. 5
- `fb_match_lineups` ................................................................. 6
- `fb_match_report` ................................................................. 7
- `fb_match_results` ................................................................. 7
- `fb_match_shooting` ................................................................. 8
- `fb_match_summary` ................................................................. 9
- `fb_match_urls` ................................................................. 10
- `fb_player_goal_logs` .............................................................. 11
- `fb_player_match_logs` ............................................................ 11
- `fb_player_scouting_report` ..................................................... 12
- `fb_player_season_stats` .......................................................... 13
- `fb_player_urls` ................................................................. 14
- `fb_season_team_stats` ............................................................ 15
- `fb_squad_wages` ................................................................. 16
- `fb_teams_urls` ................................................................. 16
- `fb_team_goal_logs` ............................................................... 17
- `fb_team_match_log_stats` ....................................................... 18
- `fb_team_match_results` .......................................................... 18
- `fb_team_player_stats` ............................................................ 19
- `fotmob_get_league_ids` .......................................................... 20
- `fotmob_get_league_matches` .................................................... 20
- `fotmob_get_league_tables` ...................................................... 22
- `fotmob_get_matches_by_date` ................................................... 23
- `fotmob_get_match_details` ....................................................... 24
- `fotmob_get_match_info` .......................................................... 24
- `fotmob_get_match_players` ....................................................... 25
- `fotmob_get_match_team_stats` .................................................. 26
- `fotmob_get_season_stats` ........................................................ 27
- `get_advanced_match_stats` ...................................................... 30
- `get_match_lineups` ............................................................... 31
- `get_match_report` ............................................................... 31
- `get_match_results` ............................................................... 32
- `get_match_shooting` ............................................................. 33
- `get_match_summary` ............................................................. 33
- `get_match_urls` ................................................................. 34
- `get_player_market_values` ..................................................... 35
- `get_season_team_stats` .......................................................... 36
- `get_team_match_results` ........................................................ 37
- `load_fb_big5_advanced_season_stats` ....................................... 37
- `load_fotmob_matches_by_date` ................................................. 38
- `load_fotmob_match_details` .................................................... 39
- `load_match_comp_results` ....................................................... 40
- `load_match_results` ............................................................. 41
- `load_understat_league_shots` ................................................... 42
fb_advanced_match_stats

Get FBref advanced match stats

Description

Returns data frame of selected statistics for each match, for either whole team or individual players. Multiple URLs can be passed to the function, but only one 'stat_type' can be selected. Replaces the deprecated function get_advanced_match_stats()

Usage

fb_advanced_match_stats(match_url, stat_type, team_or_player, time_pause = 3)
fb_big5_advanced_season_stats

Arguments

match_url the three character country code for all countries
stat_type the type of team statistics the user requires
team_or_player result either summarised for each team, or individual players
time_pause the wait time (in seconds) between page loads

The statistic type options (stat_type) include:
"summary", "passing", "passing_types", "defense", "possession", "misc", "keeper"

Value

returns a dataframe of a selected team statistic type for a selected match(es)

Examples

## Not run:
try({
  urls <- fb_match_urls(country = "AUS", gender = "F", season_end_year = 2021, tier = "1st")
  df <- fb_advanced_match_stats(match_url=urls, stat_type="possession", team_or_player="player")
})

## End(Not run)

fb_big5_advanced_season_stats

Big 5 Euro League Season Stats

Description

Returns data frame of selected statistics for seasons of the big 5 Euro leagues, for either whole team or individual players. Multiple seasons can be passed to the function, but only one ‘stat_type’ can be selected

Usage

fb_big5_advanced_season_stats(
  season_end_year,
  stat_type,
  team_or_player,
  time_pause = 3
)
Arguments

- `season_end_year`: the year(s) the season concludes
- `stat_type`: the type of team statistics the user requires
- `team_or_player`: result either summarised for each team, or individual players
- `time_pause`: the wait time (in seconds) between page loads

The statistic type options (stat_type) include:
- "standard",
- "shooting",
- "passing",
- "passing_types",
- "gca",
- "defense",
- "possession",
- "playing_time",
- "misc",
- "keepers",
- "keepers_adv"

Value

returns a dataframe of a selected team or player statistic type for a selected season(s)

Examples

```r
## Not run:
try({
  fb_big5_advanced_season_stats(season_end_year=2021, stat_type="possession", team_or_player="player")
})
## End(Not run)
```

---

**fb_league_urls**

**Get fbref League URLs**

**Description**

Returns the URLs for season leagues of a selected country

**Usage**

`fb_league_urls(country, gender, season_end_year, tier = "1st")`

**Arguments**

- `country`: the three character country code
- `gender`: gender of competition, either "M" or "F"
- `season_end_year`: the year the season(s) concludes (defaults to all available seasons)
- `tier`: the tier of the league, ie '1st' for the EPL or '2nd' for the Championship and so on

**Value**

returns a character vector of all fbref league URLs for selected country, season, gender and tier
fb_match_lineups

Get FBref match lineups

Description

Returns lineups for home and away teams for a selected match. Replaces the deprecated function get_match_lineups.

Usage

fb_match_lineups(match_url, time_pause = 3)

Arguments

match_url the fbref.com URL for the required match
time_pause the wait time (in seconds) between page loads

Value

returns a dataframe with the team lineups for a selected match

Examples

## Not run:
try({
  match <- fb_match_urls(country = "AUS", gender = "F", season_end_year = 2021, tier = "1st")[1]
  df <- fb_match_lineups(match_url = match)
})

## End(Not run)
fb_match_report

Get FBref match report

Description

Returns match report details for selected matches. Replaces the deprecated function get_match_report.

Usage

fb_match_report(match_url, time_pause = 3)

Arguments

match_url the fbref.com URL for the required match
time_pause the wait time (in seconds) between page loads

Value

returns a dataframe with the match details for a selected match

Examples

## Not run:
try({
  match <- fb_match_urls(country = "AUS", gender = "F", season_end_year = 2021, tier = "1st")[[1]]
  df <- fb_match_report(match_url = match)
})

## End(Not run)

fb_match_results

Get FBref match results

Description

Returns the game results for a given league season(s). Replaces the deprecated function get_match_results.

Usage

fb_match_results(
  country,
  gender,
  season_end_year,
  tier = "1st",
  non_dom_league_url = NA
)

fb_match_shooting

Get FBref match shooting event data

Description

Returns detailed player shooting data for home and away teams for a selected match(es). Replaces the deprecated function get_match_shooting.

Usage

fb_match_shooting(match_url, time_pause = 3)

Arguments

match_url the fbref.com URL for the required match
time_pause the wait time (in seconds) between page loads

Value

returns a dataframe
Examples

```r
## Not run:
try({
  match <- "https://fbref.com/en/matches/bf52349b/Fulham-Arsenal-September-12-2020-Premier-League"
  df <- fb_match_shooting(match_url = match)
})

## End(Not run)
```

---

**fb_match_summary**  
*Get FBref match summary*

**Description**

Returns match summary data for selected match URLs, including goals, subs and cards. Replaces the deprecated function `get_match_summary`.

**Usage**

```r
fb_match_summary(match_url, time_pause = 3)
```

**Arguments**

- `match_url` the fbref.com URL for the required match
- `time_pause` the wait time (in seconds) between page loads

**Value**

returns a dataframe with the match events (goals, cards, subs) for selected matches.

**Examples**

```r
## Not run:
try({
  match <- fb_match_urls(country = "AUS", gender = "F", season_end_year = 2021, tier = "1st")[[1]]
  df <- fb_match_summary(match_url = match)
})

## End(Not run)
```
fb_match_urls

Description

Returns the URL for each match played for a given league season. Replaces the deprecated get_match_urls.

Usage

fb_match_urls(
  country,  
gender,  
season_end_year,  
tier = "1st",  
non_dom_league_url = NA,  
time_pause = 3
)

Arguments

country the three character country code  
gender the gender of competition, either "M" or "F", or both  
season_end_year the year the season(s) concludes  
tier the tier of the league, ie '1st' for the EPL or '2nd' for the Championship and so on  
non_dom_league_url the URL for Cups and Competitions found at https://fbref.com/en/comps/  
time_pause the wait time (in seconds) between page loads

Value

returns a character vector of all fbref match URLs for selected competition, season and gender

Examples

## Not run:
try{
  fb_match_urls(country = "ENG", gender = "M", season_end_year = c(2019:2021), tier = "1st")
  non_dom <- "https://fbref.com/en/comps/218/history/Friendlys-M-Seasons"
  fb_match_urls(country = "", gender = "M", season_end_year = 2021, non_dom_league_url = non_dom)
}

## End(Not run)
**fb_player_goal_logs**  
*Get player goal logs*

**Description**

Returns the player's career goal and assist logs

**Usage**

```r
fb_player_goal_logs(player_urls, time_pause = 3, goals_or_assists = "goals")
```

**Arguments**

- `player_urls`: the URL(s) of the player(s)
- `time_pause`: the wait time (in seconds) between page loads
- `goals_or_assists`: select whether to return data of "goals" (the default), "assists", or "both"

**Value**

returns a dataframe of the player's goals and assists

**Examples**

```r
## Not run:
try({
  # for single players:
  fb_player_goal_logs(player_urls = jwp_url, goals_or_assists = "goals")
})
## End(Not run)
```

---

**fb_player_match_logs**  
*Get fbref Player Match Logs*

**Description**

Returns all match logs for a selected player, season and stat type

**Usage**

```r
fb_player_match_logs(player_url, season_end_year, stat_type, time_pause = 3)
```
fb_player_scouting_report

Arguments

player_url  the URL of the player (can come from fb_player_urls())
season_end_year  the year the season(s) concludes
stat_type  the type of statistic required
time_pause  the wait time (in seconds) between page loads

The statistic type options (stat_type) include:
"summary", "keepers", "passing", "passing_types", "gca", "defense", "possession", "misc"

Value

returns a dataframe of a player’s match logs for a season

Examples

try(
    season_end_year = 2021, stat_type = 'summary')
)

fb_player_scouting_report

Get fbref Full Player Scouting Report

Description

Returns the scouting report for a selected player

Usage

fb_player_scouting_report(player_url, pos_versus, time_pause = 3)

Arguments

player_url  the URL of the player (can come from fb_player_urls())
pos_versus  either "primary" or "secondary" as fbref offer comparisons against multiple positions
time_pause  the wait time (in seconds) between page loads

Value

returns a dataframe of a player’s full scouting information for all seasons available on FBref
Examples

```r
## Not run:
try{
  fb_player_scouting_report(player_url = "https://fbref.com/en/players/d70ce98e/Lionel-Messi", 
pos_versus = "primary")
}

# to filter for the last 365 days:
fb_player_scouting_report(player_url = "https://fbref.com/en/players/d70ce98e/Lionel-Messi", 
pos_versus = "primary") %>% dplyr::filter(scouting_period == "Last 365 Days")

# to get secondary positions
fb_player_scouting_report(player_url = "https://fbref.com/en/players/d70ce98e/Lionel-Messi", 
pos_versus = "secondary")

# for the 2020-2021 La Liga season
fb_player_scouting_report(player_url = "https://fbref.com/en/players/d70ce98e/Lionel-Messi", 
pos_versus = "secondary") %>% dplyr::filter(scouting_period == "2020-2021 La Liga")
}
```

## End(Not run)

---

**fb_player_season_stats**

Get *fbref* Player Season Statistics

Description

Returns the historical season stats for a selected player(s) and stat type

Usage

```r
fb_player_season_stats(player_url, stat_type, time_pause = 3)
```

Arguments

- `player_url` : the URL(s) of the player(s) (can come from `fb_player_urls()`)
- `stat_type` : the type of statistic required
- `time_pause` : the wait time (in seconds) between page loads

The statistic type options (stat_type) include:

"standard", "shooting", "passing", "passing_types", "gca", "defense", "possession" "playing_time", "misc", "keeper", "keeper_adv"

Value

returns a dataframe of a player’s historical season stats
fb_player_urls

Get fbref Player URLs

Description
Returns the URLs for all players for a given team.

Usage
fb_player_urls(team_url, time_pause = 3)

Arguments
  team_url  the player’s team URL (can be from fb_team_urls())
  time_pause  the wait time (in seconds) between page loads

Value
returns a character vector of all fbref player URLs for a selected team

Examples
## Not run:
try({
})

## End(Not run)
fb_season_team_stats  

Get FBref season team stats

Description

Returns different team season statistics results for a given league season and stat type Replaces the deprecated function get_season_team_stats

Usage

```r
fb_season_team_stats(
  country,
  gender,
  season_end_year,
  tier,
  stat_type,
  time_pause = 3
)
```

Arguments

- `country` the three character country code for all countries
- `gender` gender of competition, either "M", "F" or both
- `season_end_year` the year the season(s) concludes
- `tier` the tier of the league, ie '1st' for the EPL or '2nd' for the Championship and so on
- `stat_type` the type of team statistics the user requires
- `time_pause` the wait time (in seconds) between page loads

The statistic type options (stat_type) include:

- "league_table"
- "league_table_home_away"
- "standard"
- "keeper"
- "keeper_adv"
- "shooting"
- "passing"
- "passing_types"
- "goal_shot_creation"
- "defense"
- "possession"
- "playing_time"
- "misc"

Value

returns a dataframe of a selected team statistic type for a selected league season

Examples

```r
## Not run:
try({
  fb_season_team_stats("ITA", "M", 2021, "1st", "defense")
})

## End(Not run)
```
fb_squad_wages  
Get team player wages

Description

Returns all player wages from FBref via Capology

Usage

fb_squad_wages(team_urls, time_pause = 3)

Arguments

  team_urls      the URL(s) of the teams(s) (can come from fb_teams_urls())
  time_pause     the wait time (in seconds) between page loads

Value

returns a dataframe with all available estimated player wages for the selected team(s)

Examples

## Not run:
try({
  # for single teams:
  man_city_url <- "https://fbref.com/en/squads/b8fd03ef/Manchester-City-Stats"
  fb_squad_wages(team_urls = man_city_url)
})

## End(Not run)

fb_teams_urls  
Get fbref Team URLs

Description

Returns the URLs for all teams for a given league

Usage

fb_teams_urls(league_url, time_pause = 3)

Arguments

  league_url      the league URL (can be from fb_league_urls())
  time_pause      the wait time (in seconds) between page loads
Value

returns a character vector of all fbref team URLs for a selected league

Examples

```r
## Not run:
try({
})
## End(Not run)
```

---

fb_team_goal_logs  

*Get team goal logs*

Description

Returns the team’s season goal logs

Usage

```r
fb_team_goal_logs(team_urls, time_pause = 3, for_or_against = "for")
```

Arguments

- `team_urls`: the URL(s) of the team(s) (can come from `fb_teams_urls()
- `time_pause`: the wait time (in seconds) between page loads
- `for_or_against`: select whether to return data of goals "for" (the default), goals "against", or "both"

Value

returns a dataframe of the team’s goals scored and conceded in the season

Examples

```r
## Not run:
try({
  # for single teams:
  man_city_url <- "https://fbref.com/en/squads/b8fd03ef/Manchester-City-Stats"
  fb_team_goal_logs(team_urls = man_city_url, for_or_against = "for")
})
## End(Not run)
```
**fb_team_match_log_stats**

*Get team match log stats*

**Description**

Returns all match statistics for a team(s) in a given season

**Usage**

```r
fb_team_match_log_stats(team_urls, stat_type, time_pause = 3)
```

**Arguments**

- `team_urls`: the URL(s) of the teams(s) (can come from `fb_teams_urls()`)
- `stat_type`: the type of statistic required
- `time_pause`: the wait time (in seconds) between page loads

The statistic type options (stat_type) include:
- "shooting"
- "keeper"
- "passing"
- "passing_types"
- "gca"
- "defense"
- "misc"

**Value**

returns a dataframe with the selected stat outputs of all games played by the selected team(s)

**Examples**

```r
## Not run:
try({
  # for single teams:
  man_city_url <- "https://fbref.com/en/squads/b8fd03ef/Manchester-City-Stats"
  fb_team_match_log_stats(team_urls = man_city_url, stat_type = "passing")
})
## End(Not run)
```

**fb_team_match_results**

*Get FBref team match results*

**Description**

Returns all game results for a team in a given season Replaces the deprecated function `get_team_match_results`

**Usage**

```r
fb_team_match_results(team_url, time_pause = 3)
```
fb_team_player_stats

Arguments

- team_url: the URL for the team season
- time_pause: the wait time (in seconds) between page loads

Value

returns a dataframe with the results of all games played by the selected team(s)

Examples

```r
## Not run:
try({
# for single teams:
man_city_url <- "https://fbref.com/en/squads/b8fd03ef/Manchester-City-Stats"
fteam.match_results(man_city_url)
})
## End(Not run)
```

fb_team_player_stats  Get fbref Team’s Player Season Statistics

Description

Returns the team’s players season stats for a selected team(s) and stat type

Usage

```r
fb_team_player_stats(team_urls, stat_type, time_pause = 3)
```

Arguments

- team_urls: the URL(s) of the teams(s) (can come from fb_teams_urls())
- stat_type: the type of statistic required
- time_pause: the wait time (in seconds) between page loads
  The statistic type options (stat_type) include:
  "standard", "shooting", "passing", "passing_types", "gca", "defense", "possession" "playing_time", "misc", "keeper", "keeper_adv"

Value

returns a dataframe of all players of a team’s season stats
### fotmob_get_league_matches

Returns match results for all matches played on the selected date from fotmob.com.

#### Usage

```r
fotmob_get_league_matches(country, league_name, league_id, cached = TRUE)
```

#### Arguments

- `country`: Country of the league.
- `league_name`: Name of the league.
- `league_id`: ID of the league.
- `cached`: Whether to load the data frame from the data CSV. This is faster and most likely what you want to do, unless you identify a league that’s being tracked by fotmob that’s not in this pre-saved CSV.

---

### fotmob_get_league_ids

Get fotmob league ids

#### Description

Returns a dataframe of the league ids available on fotmob.

#### Usage

```r
fotmob_get_league_ids(cached = TRUE)
```

#### Arguments

- `cached`: Whether to load the data frame from the data CSV. This is faster and most likely what you want to do, unless you identify a league that’s being tracked by fotmob that’s not in this pre-saved CSV.
Arguments

country  Three character country code. Can be one or multiple. If provided, `league_name` must also be provided (of the same length).

league_name  League names. If provided, `country` must also be provided (of the same length).

league_id  Fotmob ID for the league. Only used if `country` and `league_name` are not specified.

cached  Whether to load the dataframe from the data CSV. This is faster and most likely what you want to do, unless you identify a league that’s being tracked by fotmob that’s not in this pre-saved CSV.

Value

returns a dataframe of league matches

Examples

```r
try({
  library(dplyr)
  library(tidyr)

  # one league
  fotmob_get_league_matches(
    country = "ENG",
    league_name = "Premier League"
  )

  # one league, by id
  fotmob_get_league_matches(
    league_id = 47
  )

  # multiple leagues (could also use ids)
  league_matches <- fotmob_get_league_matches(
    country = c("ENG", "ESP"),
    league_name = c("Premier League", "LaLiga")
  )

  # probably the data that you care about
  league_matches %>%
    dplyr::select(match_id = id, home, away) %>%
    tidyr::unnest_wider(c(home, away), names_sep = ",")
})
```
fotmob_get_league_tables

*Get standings from fotmob*

**Description**

Returns league standings from fotmob.com. 3 types are returned: all, home, away

**Usage**

```r
definition
fotmob_get_league_tables(country, league_name, league_id, cached = TRUE)
definition
```

**Arguments**

- `country` Three character country code. Can be one or multiple. If provided, `league_name` must also be provided (of the same length)
- `league_name` League names. If provided, `country` must also be provided (of the same length).
- `league_id` Fotmob ID for the league. Only used if `country` and `league_name` are not specified.
- `cached` Whether to load the dataframe from the data CSV. This is faster and most likely what you want to do, unless you identify a league that’s being tracked by fotmob that’s not in this pre-saved CSV.

**Value**

returns a dataframe of league standings

**Examples**

```r
try{
library(dplyr)
library(tidyr)

# one league
fotmob_get_league_tables(
    country = "ENG",
    league_name = "Premier League"
)

# one league, by id
fotmob_get_league_tables(
    league_id = 47
)

# multiple leagues (could also use ids)
league_tables <- fotmob_get_league_tables(
    country = c("ENG", "ESP" ),
```

```r
```
fotmob_get_matches_by_date

Get fotmob match results by date

Description

Returns match results for all matches played on the selected date from fotmob.com

Usage

fotmob_get_matches_by_date(dates)

Arguments

dates a vector of string-formatted dates in "Ymd" format, e.g. "20210926". An attempt is made to coerce the input to the necessary format if a date is passed in.

Value

returns a dataframe of match results

Examples

try({
library(dplyr)
library(tidyr)

results <- fotmob_get_matches_by_date(date = c("20210925", "20210926"))
results %>%
  dplyr::select(primary_id, ccode, league_name = name, match_id)
})
fotmob_get_match_details

Get fotmob match details by match id

Description

Returns match details from fotmob.com

Usage

fotmob_get_match_details(match_ids)

Arguments

match_ids  a vector of strings or numbers representing matches

Value

returns a dataframe of match shots

Examples

try({
library(dplyr)
library(tidyr)
results <- fotmob_get_matches_by_date(date = "20210926")
match_ids <- results %>%
dplyr::select(primary_id, ccode, league_name = name, match_id) %>%
dplyr::filter(league_name == "Premier League", ccode == "ENG") %>%
dplyr::pull(match_id)
match_ids # 3609987 3609979
details <- fotmob_get_match_details(match_ids)
})

fotmob_get_match_info

Get fotmob match info by match id

Description

Returns match info from fotmob.com

Usage

fotmob_get_match_info(match_ids)
fotmob_get_match_players

Arguments

match_ids a vector of strings or numbers representing matches

Value

returns a dataframe of match info

Examples

try({
  library(dplyr)
  library(tidyr)
  results <- fotmob_get_matches_by_date(date = "20210926")
  match_ids <- results %>%
    dplyr::select(primary_id, ccode, league_name = name, match_id) %>%
    dplyr::filter(league_name == "Premier League", ccode == "ENG") %>%
    dplyr::pull(match_id)
  match_ids # 3609987 3609979
  details <- fotmob_get_match_info(match_ids)
})
Examples

```
try(
  library(dplyr)
  library(tidyr)
  ## single match
  players <- fotmob_get_match_players(3610132)
  salah_id <- "292462"
  players %>%
    dplyr::filter(id == salah_id) %>%
    dplyr::select(player_id = id, stats) %>%
    tidyr::unnest(stats)

  ## multiple matches
  fotmob_get_match_players(c(3609987, 3609979))
)
```

---

**fotmob_get_match_team_stats**

*Get fotmob match top team stats by match id*

**Description**

Returns match top team stats from fotmob.com

**Usage**

```
fotmob_get_match_team_stats(match_ids)
```

**Arguments**

- `match_ids`: a vector of strings or numbers representing matches

**Value**

returns a dataframe of match top team stats

**Examples**

```
try({
  library(dplyr)
  library(tidyr)
  results <- fotmob_get_matches_by_date(date = "20210926")
  match_ids <- results %>%
    dplyr::select(primary_id, ccode, league_name = name, match_id) %>%
    dplyr::filter(league_name == "Premier League", ccode == "ENG") %>%
    dplyr::pull(match_id)
```
fotmob_get_season_stats

Get season statistics from fotmob

Description

Returns team or player season-long statistics standings from fotmob.com.

Usage

fotmob_get_season_stats(
  country,
  league_name,
  league_id,
  season_name,
  team_or_player = c("team", "player"),
  stat_name,
  stat_league_name = league_name,
  cached = TRUE
)

Arguments

country Three character country code. Can be one or multiple. If provided, 'league_name' must also be provided (of the same length)
league_name League names. If provided, 'country' must also be provided (of the same length).
league_id Fotmob ID for the league. Only used if 'country' and 'league_name' are not specified.
season_name Season names in the format "2021/2022". Multiple allowed. If multiple leagues are specified, season stats are retrieved for each league.
team_or_player return statistics for either "team" or "player". Can only be one or the other.
stat_name the type of statistic. Can be more than one. 'stat_name' must be one of the following for "player":
  • Accurate long balls per 90
  • Accurate passes per 90
  • Assists
  • Big chances created
  • Big chances missed
  • Blocks per 90
• Chances created
• Clean sheets
• Clearances per 90
• Expected assist (xA)
• Expected assist (xA) per 90
• Expected goals (xG)
• Expected goals (xG) per 90
• Expected goals on target (xGOT)
• FotMob rating
• Fouls committed per 90
• Goals + Assists
• Goals conceded per 90
• Goals per 90
• Goals prevented
• Interceptions per 90
• Penalties conceded
• Penalties won
• Possession won final 3rd per 90
• Red cards
• Save percentage
• Saves per 90
• Shots on target per 90
• Shots per 90
• Successful dribbles per 90
• Successful tackles per 90
• Top scorer
• xG + xA per 90
• Yellow cards

For "team", 'stat_name' must be one of the following:
• Accurate crosses per match
• Accurate long balls per match
• Accurate passes per match
• Average possession
• Big chances created
• Big chances missed
• Clean sheets
• Clearances per match
• Expected goals
• FotMob rating
• Fouls per match
• Goals conceded per match
• Goals per match
fotmob_get_season_stats

- Interceptions per match
- Penalties awarded
- Penalties conceded
- Possession won final 3rd per match
- Red cards
- Saves per match
- Shots on target per match
- Successful tackles per match
- xG conceded
- Yellow cards

Fotmob has changed these stat names over time, so this list may be out-dated. If you try an invalid stat name, you should see an error message indicating which ones are available.

stat_league_name

Same format as ‘league_name’. If not provided explicitly, then it takes on the same value as ‘league_name’. If provided explicitly, should be of the same length as ‘league_name’ (or ‘league_id’ if ‘league_name’ is not provided).

Note that not Fotmob currently only goes back as far as ‘2016/2017’. Some leagues may not have data for that far back.

cached

Whether to load the dataframe from the data CSV. This is faster and most likely what you want to do, unless you identify a league that’s being tracked by fotmob that’s not in this pre-saved CSV.

Value

returns a dataframe of team or player stats

Examples

```r
try({
  epl_team_xg_2021 <- fotmob_get_season_stats(
    country = "ENG",
    league_name = "Premier League",
    season = "2020/2021",
    stat_name = "Expected goals",
    team_or_player = "team"
  )
})
```
get_advanced_match_stats

Get advanced match stats

Description

Returns data frame of selected statistics for each match, for either whole team or individual players. Multiple URLs can be passed to the function, but only one 'stat_type' can be selected.

Usage

get_advanced_match_stats(match_url, stat_type, team_or_player, time_pause = 3)

Arguments

match_url the three character country code for all countries
stat_type the type of team statistics the user requires
team_or_player result either summarised for each team, or individual players
time_pause the wait time (in seconds) between page loads

The statistic type options (stat_type) include:
"summary", "passing", "passing_types", "defense", "possession", "misc", "keeper"

Value

returns a dataframe of a selected team statistic type for a selected match(es)

Examples

## Not run:
try({
  urls <- get_match_urls(country = "AUS", gender = "F", season_end_year = 2021, tier = "1st")
  df <- get_advanced_match_stats(match_url=urls, stat_type="possession", team_or_player="player")
})

## End(Not run)
get_match_lineups

Get match lineups

Description
Returns lineups for home and away teams for a selected match

Usage
get_match_lineups(match_url, time_pause = 3)

Arguments
match_url: the fbref.com URL for the required match
time_pause: the wait time (in seconds) between page loads

Value
returns a dataframe with the team lineups for a selected match

Examples
## Not run:
try{
match <- get_match_urls(country = "AUS", gender = "F", season_end_year = 2021, tier = "1st")[1]
df <- get_match_lineups(match_url = match)
}
## End(Not run)

get_match_report

Get match report

Description
Returns match report details for selected matches

Usage
get_match_report(match_url, time_pause = 3)

Arguments
match_url: the fbref.com URL for the required match
time_pause: the wait time (in seconds) between page loads
get_match_results

**Value**
returns a dataframe with the match details for a selected match

**Examples**

```r
## Not run:
try({
  match <- get_match_urls(country = "AUS", gender = "F", season_end_year = 2021, tier = "1st")[1]
  df <- get_match_report(match_url = match)
})
## End(Not run)
```

---

**Description**
Returns the game results for a given league season(s)

**Usage**

```r
get_match_results(
  country, 
  gender, 
  season_end_year, 
  tier = "1st", 
  non_dom_league_url = NA
)
```

**Arguments**

- **country**: the three character country code
- **gender**: gender of competition, either "M" or "F"
- **season_end_year**: the year(s) the season concludes
- **tier**: the tier of the league, ie '1st' for the EPL or '2nd' for the Championship and so on
- **non_dom_league_url**: the URL for Cups and Competitions found at https://fbref.com/en/comps/

**Value**
returns a dataframe with the results of the competition, season and gender
get_match_shooting

Examples

## Not run:
try({
df <- get_match_results(country = c("ITA"), gender = "M", season_end_year = 2021)
# for results from English Championship:
df <- get_match_results(country = "ENG", gender = "M", season_end_year = 2021, tier = "2nd")
# for international friendlies:
})

## End(Not run)

get_match_shooting  Get match shooting event data

Description

Returns detailed player shooting data for home and away teams for a selected match(es)

Usage

get_match_shooting(match_url, time_pause = 3)

Arguments

match_url  the fbref.com URL for the required match
time_pause  the wait time (in seconds) between page loads

Value

returns a dataframe

Examples

## Not run:
try({
match <- "https://fbref.com/en/matches/bf52349b/Fulham-Arsenal-September-12-2020-Premier-League"
df <- get_match_shooting(match_url = match)
})

## End(Not run)
get_match_summary  Get match summary

Description
Returns match summary data for selected match URLs, including goals, subs and cards

Usage
get_match_summary(match_url, time_pause = 3)

Arguments
match_url  the fbref.com URL for the required match
time_pause  the wait time (in seconds) between page loads

Value
returns a dataframe with the match events (goals, cards, subs) for selected matches

Examples
## Not run:
try({
match <- get_match_urls(country = "AUS", gender = "F", season_end_year = 2021, tier = "1st")[,1]
df <- get_match_summary(match_url = match)
})
## End(Not run)

get_match_urls  Get match URLs

Description
Returns the URL for each match played for a given league season

Usage
get_match_urls(
  country,
gender,
season_end_year,
tier = "1st",
non_dom_league_url = NA,
time_pause = 3
)
get_player_market_values

Arguments

country     the three character country code
gender      gender of competition, either "M" or "F", or both
season_end_year   the year the season(s) concludes
tier         the tier of the league, ie ‘1st’ for the EPL or ’2nd’ for the Championship and so on
non_dom_league_url the URL for Cups and Competitions found at https://fbref.com/en/comps/
time_pause   the wait time (in seconds) between page loads

Value

returns a character vector of all fbref match URLs for selected competition, season and gender

Examples

## Not run:
try(
  get_match_urls(country = "ENG", gender = "M", season_end_year = c(2019:2021), tier = "1st")
  non_dom <- "https://fbref.com/en/comps/218/history/Friendlies-M-Seasons"
  get_match_urls(country = "", gender = "M", season_end_year = 2021, non_dom_league_url = non_dom)
)

## End(Not run)

get_player_market_values

Get player market values

Description

Returns data frame of player valuations (in Euros) from transfermarkt.com

Usage

get_player_market_values(country_name, start_year, league_url = NA)

Arguments

country_name  the country of the league’s players
start_year    the start year of the season (2020 for the 20/21 season)
league_url    league url from transfermarkt.com. To be used when country_name not available in main function

Value

returns a dataframe of player valuations for country/seasons
get_season_team_stats  

Get season team stats

Description

Returns different team season statistics results for a given league season and stat type

Usage

get_season_team_stats(
  country,
  gender,
  season_end_year,
  tier,
  stat_type,
  time_pause = 3
)

Arguments

country the three character country code for all countries
gender gender of competition, either "M", "F" or both
season_end_year the year the season(s) concludes
tier the tier of the league, ie '1st' for the EPL or '2nd' for the Championship and so on
stat_type the type of team statistics the user requires
time_pause the wait time (in seconds) between page loads

The statistic type options (stat_type) include:

"league_table", "league_table_home_away", "standard", "keeper", "keeper_adv", "shooting", "passing", "passing_types", "goal_shot_creation", "defense", "possession", "playing_time", "misc"

Value

returns a dataframe of a selected team statistic type for a selected league season

Examples

## Not run:
try({
  get_season_team_stats("ITA", "M", 2021, "1st", "defense")
})

## End(Not run)
get_team_match_results

Get team match results

Description

Returns all game results for a team in a given season

Usage

get_team_match_results(team_url, time_pause = 3)

Arguments

team_url: the URL for the team season

time_pause: the wait time (in seconds) between page loads

Value

returns a dataframe with the results of all games played by the selected team(s)

Examples

```r
## Not run:
try({
# for single teams:
man_city_url <- "https://fbref.com/en/squads/b8fd03ef/Manchester-City-Stats"
get_team_match_results(man_city_url)
})
## End(Not run)
```

load_fb_big5_advanced_season_stats

Load Big 5 Euro League Season Stats

Description

Loading version of fb_big5_advanced_season_stats. Returns data frame of selected statistics for seasons of the big 5 Euro leagues, for either whole team or individual players. Multiple seasons can be passed to the function, but only one ‘stat_type’ can be selected
load_fotmob_matches_by_date

Load pre saved fotmob match ids by date

Description

Loading version of fotmob_get_matches_by_date. Goes back to August 2017.

Usage

load_fotmob_matches_by_date(country, league_name, league_id, cached = TRUE)
Arguments

- **country**: Three character country code. Can be one or multiple. If provided, 'league_name' must also be provided (of the same length).
- **league_name**: League names. If provided, 'country' must also be provided (of the same length).
- **league_id**: Fotmob ID for the league. Only used if 'country' and 'league_name' are not specified.
- **cached**: Whether to load the dataframe from the data CSV. This is faster and most likely what you want to do, unless you identify a league that's being tracked by fotmob that's not in this pre-saved CSV.

Value

- returns a dataframe of league match ids

Examples

```r
## Not run:
try({
## just load match ids
load_fotmob_matches_by_date(
    country = "ENG",
    league_name = "Premier League"
)

## can also do it for multiple leagues
load_fotmob_matches_by_date(
    country = c("ENG", "ESP"),
    league_name = c("Premier League", "LaLiga")
)
})

## End(Not run)
```

load_fotmob_match_details

Load pre saved fotmob match details

Description

Loading version of `fotmob_get_match_details`, but for all seasons for which data is available, not just the current season. Note that fotmobonly has match details going back to the 2020-21 season for most leagues.

Usage

```r
load_fotmob_match_details(country, league_name, league_id, cached = TRUE)
```
Arguments

- **country**: Three character country code. Can be one or multiple. If provided, ‘league_name’ must also be provided (of the same length).
- **league_name**: League names. If provided, ‘country’ must also be provided (of the same length).
- **league_id**: Fotmob ID for the league. Only used if ‘country’ and ‘league_name’ are not specified.
- **cached**: Whether to load the dataframe from the data CSV. This is faster and most likely what you want to do, unless you identify a league that’s being tracked by fotmob that’s not in this pre-saved CSV.

Value

returns a dataframe of league matches

Examples

```r
## Not run:
try({
  # one league
  load_fotmob_match_details(
    country = "ENG",
    league_name = "Premier League"
  )

  ## this is the same output format as the following
  fotmob_get_match_details(match_id = 3411352)

  # one league, by id
  load_fotmob_match_details(league_id = 47)

  # multiple leagues (could also use ids)
  load_fotmob_match_details(
    country = c("ENG", "ESP"),
    league_name = c("Premier League", "LaLiga")
  )
})
## End(Not run)
```

Description

Returns the game results for a competition(s), ie League cups or international competitions from FBref. comp_name comes from https://github.com/JaseZiv/worldfootballR_data/tree/master/data/match_results_cups#readme.
Usage

load_match_comp_results(comp_name)

Arguments

comp_name the three character country code

Value

returns a dataframe with the results of the competition name

Examples

try({
  df <- load_match_comp_results(
    comp_name = "Coppa Italia"
  )
  # for multiple competitions:
  cups <- c("FIFA Women's World Cup",
            "FIFA World Cup")
  df <- load_match_comp_results(
    comp_name = cups
  )
})

load_match_results Load match results

Description

Loading version of get_match_results Returns the game results for a given league season(s) from FBref

Usage

load_match_results(country, gender, season_end_year, tier)

Arguments

country the three character country code
gender gender of competition, either "M" or "F"
season_end_year the year(s) the season concludes
tier the tier of the league, ie '1st' for the EPL or '2nd' for the Championship and so on
load_understat_league_shots

Value

returns a dataframe with the results of the competition, season and gender

Examples

```r
try({
df <- load_match_results(
country = c("ITA"), gender = "M", season_end_year = 2021, tier = "1st"
)
# for results from English 1st div for men and women:
df <- load_match_results(
country = "ENG", gender = c("M", "F"), season_end_year = 2021, tier = "1st"
)
})
```

load_understat_league_shots

*Load Understat league shot locations*

Description

Loading version of `understat_league_season_shots`, but for all seasons Returns shooting locations for all matches played in the selected league

Usage

`load_understat_league_shots(league)`

Arguments

- `league`:
  - the available leagues in Understat as outlined below
  - The leagues currently available for Understat are: "EPL", "La liga", "Bundesliga", "Serie A", "Ligue 1", "RFPL"

Value

returns a dataframe of shooting locations for a selected league

Examples

```r
## Not run:
try({
df <- load_understat_league_shots(league="Serie A")
})
## End(Not run)
```
player_dictionary_mapping

Player Mapping Dictionary

Description

Returns data frame of players from the top 5 Euro leagues, their player URL and their respective Transfermarkt URL. Currently only for the players who have been in the top 5 leagues since the 2017-2018 season.

Usage

player_dictionary_mapping()

Value

returns a dataframe of FBref players and respective Transfermarkt URL

Examples

try({
  mapped_players <- player_dictionary_mapping()
})

player_transfer_history

Get player transfer history

Description

Returns data frame of player(s) transfer history from transfermarkt.com

Usage

player_transfer_history(player_urls)

Arguments

player_urls the player url(s) from transfermarkt

Value

returns a dataframe of player transfers
tm_expiring_contracts  Get expiring contracts

Description
Returns a data frame of players with expiring contracts for a selected league and time period

Usage
```r
tm_expiring_contracts(country_name, contract_end_year, league_url = NA)
```

Arguments
- `country_name`: the country of the league’s players
- `contract_end_year`: the year the contract is due to expire
- `league_url`: league url from transfermarkt.com. To be used when `country_name` not available in main function

Value
returns a dataframe of expiring contracts in the selected league

tm_league_debutants  Get league debutants

Description
Returns a data frame of debutants for a selected league

Usage
```r
tm_league_debutants(
  country_name,
  league_url = NA,
  debut_type,
  debut_start_year,
  debut_end_year
)
```
Arguments

country_name  the country of the league’s players
league_url  league url from transfermarkt.com. To be used when country_name not available in main function
debut_type  whether you want ‘league’ debut or ‘pro’ debut
debut_start_year  the season start year of the beginning of the period you want results for
debut_end_year  the season start year of the end of the period you want results for

Value

returns a dataframe of players who debuted in the selected league

Description

Returns a data frame of all currently injured players players for a selected league

Usage

tm_league_injuries(country_name, league_url = NA)

Arguments

country_name  the country of the league’s players
league_url  league url from transfermarkt.com. To be used when country_name not available in main function

Value

returns a dataframe of injured players in the selected league
### tm_league_team_urls

**Get transfermarkt Team URLs**

**Description**

Returns the URLs for all teams for a given league season

**Usage**

```r
tm_league_team_urls(country_name, start_year, league_url = NA)
```

**Arguments**

- `country_name`  
  the country of the league’s players

- `start_year`  
  the start year of the season (2020 for the 20/21 season)

- `league_url`  
  league url from transfermarkt.com. To be used when `country_name` not available in main function

**Value**

returns a character vector of all transfermarkt team URLs for a selected league

### tm_matchday_table

**Get weekly league table**

**Description**

Returns the league table for each chosen matchday from transfermarkt

**Usage**

```r
tm_matchday_table(country_name, start_year, matchday, league_url = NA)
```

**Arguments**

- `country_name`  
  the country of the league’s players

- `start_year`  
  the start year of the season (2020 for the 20/21 season)

- `matchday`  
  the matchweek number. Can be a vector of matchdays

- `league_url`  
  league url from transfermarkt.com. To be used when `country_name` not available in main function

**Value**

returns a dataframe of the table for a selected league and matchday
Examples

```r
## Not run:
try({
  tm_matchday_table(country_name="England", start_year="2020", matchday=1)
  tm_matchday_table(country_name="England", start_year="2020", matchday=c(1:5))
})
## End(Not run)
```

### Description

tm_player_bio

Get transfermarkt player bios

### Usage

```r
tm_player_bio(player_urls)
```

### Arguments

- `player_urls`: player url(s) from transfermarkt

### Value

returns a dataframe of player bios

### Examples

```r
## Not run:
try({
  player_url <- "https://www.transfermarkt.com/eden-hazard/profil/spieler/50202"
  tm_player_bio(player_url)
})
## End(Not run)
```
tm_player_injury_history

*Get player injury history*

**Description**
Returns data frame of a player’s injury history transfermarkt.com

**Usage**

```
tm_player_injury_history(player_urls)
```

**Arguments**

- `player_urls` player url(s) from transfermarkt

**Value**
returns a dataframe of player injury history

---

tm_player_market_values

*Get Transfermarkt player market values*

**Description**
Returns data frame of player valuations (in Euros) from transfermarkt.com Replaces the deprecated function `get_player_market_values`

**Usage**

```
tm_player_market_values(country_name, start_year, league_url = NA)
```

**Arguments**

- `country_name` the country of the league’s players
- `start_year` the start year of the season (2020 for the 20/21 season)
- `league_url` league url from transfermarkt.com. To be used when `country_name` not available in main function

**Value**
returns a dataframe of player valuations for country/seasons
tm_player_transfer_history

Description
Returns data frame of player(s) transfer history from transfermarkt.com Replaces the deprecated function player_transfer_history

Usage
```
tm_player_transfer_history(player_urls)
```

Arguments
- `player_urls` the player url(s) from transfermarkt

Value
returns a dataframe of player transfers

---

tm_squad_stats

Get squad player stats

Description
Returns basic stats for players of a team season

Usage
```
tm_squad_stats(team_url)
```

Arguments
- `team_url` transfermarkt.com team url for a season

Value
returns a dataframe of all player stats for team(s)
tm_staff_job_history  Get Staff Member's job history

Description
Returns all roles a selected staff member(s) has held and performance data

Usage
tm_staff_job_history(staff_urls)

Arguments
staff_urls  transfermarkt.com staff(s) url (can use tm_league_staff_urls() to get)

Value
returns a data frame of all roles a selected staff member(s) has held and performance data

---

tm_team_player_urls  Get transfermarkt Player URLs

Description
Returns the transfermarkt URLs for all players for a given team

Usage
tm_team_player_urls(team_url)

Arguments
team_url  the player's team URL (can be from tm_league_team_urls())

Value
returns a character vector of all transfermarkt player URLs for a selected team
tm_team_staff_history  Get team staff history

Description

Returns all people who have held the selected role in a team’s history

Usage

tm_team_staff_history(team_urls, staff_role = "Manager")

Arguments

  team_urls  transfermarkt.com team(s) url for a season
  staff_role  the role description which can be found here: https://github.com/JaseZiv/worldfootballR_data/blob/master/raw-data/transfermarkt_staff/tm_staff_types.csv

Value

  returns a data frame of all selected staff roles for a team(s) history

---

tm_team_staff_urls  Get transfermarkt Club Staff URLs

Description

Returns the transfermarkt URLs for all staff of selected roles for a given team

Usage

tm_team_staff_urls(team_urls, staff_role)

Arguments

  team_urls  the staff member’s team URL (can be from tm_league_team_urls())
  staff_role  role of the staff member URLs required for with options including:
  "Manager", "Assistant Manager", "Goalkeeping Coach", "Fitness Coach", "Conditioning Coach"

Value

  returns a character vector of all transfermarkt staff URLs for a selected team(s)
tm_team_transfers  
*Get team transfers*

**Description**

Returns all transfer arrivals and departures for a given team season.

**Usage**

tm_team_transfers(team_url, transfer_window = "all")

**Arguments**

- team_url: transfermarkt.com team url for a season
- transfer_window: which window the transfer occurred - options include "all" for both, "summer" or "winter"

**Value**

returns a dataframe of all team transfers

---

tm_team_transfer_balances  
*Team transfer balances*

**Description**

Returns all team’s transfer aggregated performances for a chosen league season.

**Usage**

tm_team_transfer_balances(country_name, start_year, league_url = NA)

**Arguments**

- country_name: the country of the league’s players
- start_year: the start year of the season (2020 for the 20/21 season)
- league_url: league url from transfermarkt.com. To be used when country_name not available in main function

**Value**

returns a dataframe of the summarised financial transfer performance of all teams for a league season
understat_league_match_results

Get Understat season match results

Description

Returns match results for all matches played in the selected league season from Understat.com

Usage

understat_league_match_results(league, season_start_year)

Arguments

league              the available leagues in Understat as outlined below
season_start_year  the year the season started

The leagues currently available for Understat are: "EPL", "La liga", "Bundesliga", "Serie A", "Ligue 1", "RFPL"

Value

returns a dataframe of match results for a selected league season

understat_league_season_shots

Get Understat league season shot locations

Description

Returns shooting locations for all matches played in the selected league season from Understat.com

Usage

understat_league_season_shots(league, season_start_year)

Arguments

league              the available leagues in Understat as outlined below
season_start_year  the year the season started

The leagues currently available for Understat are: "EPL", "La liga", "Bundesliga", "Serie A", "Ligue 1", "RFPL"

Value

returns a dataframe of shooting locations for a selected league season
understat_match_shots  Get Understat match shot locations

Description
Returns shooting locations for a selected match from Understat.com

Usage
understat_match_shots(match_url)

Arguments
match_url  the URL of the match played

Value
returns a dataframe of shooting locations for a selected team season

understat_player_shots
Get all Understat shot locations for a player

Description
Returns shooting locations for a selected player for all matches played from Understat.com

Usage
understat_player_shots(player_url)

Arguments
player_url  the URL of a selected player

Value
returns a dataframe of shooting locations for a selected player
understat_team_meta  
*Get Understat team info*

**Description**
Retrieve Understat team metadata, including team URLs. Similar to `understatr::get_team_meta`.

**Usage**

```r
understat_team_meta(team_names)
```

**Arguments**

- `team_names`  
a vector of team names (can be just 1)

**Value**

a data.frame

**Examples**

```r
## Not run:
try({
  understat_team_meta(team_name = c("Liverpool", "Manchester City"))
})
## End(Not run)
```

understat_team_players_stats  
*Get Understat team player stats*

**Description**
Retrieve Understat team player stats.

**Usage**

```r
understat_team_players_stats(team_url)
```

**Arguments**

- `team_url`  
the URL of the team season

**Value**

a dataframe of player stats for a selected team season
understat_team_season_shots

Get Understat team season shot locations

Description
Returns shooting locations for all matches played by a selected team from Understat.com

Usage
understat_team_season_shots(team_url)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>team_url</td>
<td>the URL of the team season</td>
</tr>
</tbody>
</table>

Value
returns a dataframe of shooting locations for a selected team season

understat_team_stats_breakdown

Get Understat team statistics breakdowns

Description
Returns a data frame for the selected team(s) with stats broken down in different ways. Breakdown groups include:

Usage
understat_team_stats_breakdown(team_urls)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>team_urls</td>
<td>the url(s) of the teams in question</td>
</tr>
</tbody>
</table>

Details


Value
returns a dataframe of all stat groups and values
Index

fb_advanced_match_stats, 3
fb_big5_advanced_season_stats, 4
fb_league_urls, 5
fb_match_lineups, 6
fb_match_report, 7
fb_match_results, 7
fb_match_shooting, 8
fb_match_summary, 9
fb_match_urls, 10
fb_player_goal_logs, 11
fb_player_match_logs, 11
fb_player_scouting_report, 12
fb_player_season_stats, 13
fb_player_urls, 14
fb_season_team_stats, 15
fb_squad_wages, 16
fb_team_goal_logs, 17
fb_team_match_log_stats, 18
fb_team_match_results, 18
fb_team_player_stats, 19
fb_teams_urls, 16
fotmob.get League_ids, 20
fotmob.get League_matches, 20
fotmob.get League_tables, 22
fotmob.get Match_details, 24
fotmob.get Match_info, 24
fotmob.get Match_players, 25
fotmob.get Match_team_stats, 26
fotmob.get matches by date, 23
fotmob.get season_stats, 27
get advanced_match_stats, 30
get_match_lineups, 31
get_match_report, 31
get_match_results, 32
get_match_shooting, 33
get_match_summary, 34
get_match_urls, 34
get_player_market_values, 35
get_season_team_stats, 36
get_team_match_results, 37
load_fb_big5_advanced_season_stats, 37
load_fotmob_match_details, 39
load_fotmob_matches_by_date, 38
load_match_comp_results, 40
load_match_results, 41
load_understat_league_shots, 42
player_dictionary_mapping, 43
player_transfer_history, 43
tm_expiring_contracts, 44
tm_league_debutants, 44
tm_league_injuries, 45
tm_league_team_urls, 46
tm_matchday_table, 46
tm_player_bio, 47
tm_player_injury_history, 48
tm_player_market_values, 48
tm_player_transfer_history, 49
tm_squad_stats, 49
tm_staff_job_history, 50
tm_team_player_urls, 50
tm_team_staff_history, 51
tm_team_staff_urls, 51
tm_team_transfer_balances, 52
tm_team_transfers, 52
understat_league_match_results, 53
understat_league_season_shots, 53
understat_match_shots, 54
understat_player_shots, 54
understat_team_meta, 55
understat_team_players_stats, 55
understat_team_season_shots, 56
understat_team_stats_breakdown, 56

57