

# Package ‘fflr’

April 6, 2022

**Title** Retrieve ESPN Fantasy Football Data

**Version** 2.0.3

**Description** Format the raw data from the ESPN fantasy football API  
<<https://fantasy.espn.com/apis/v3/games/ffl/>> as data frames.  
Retrieve data on public leagues, rosters, athletes, and matches.

**License** MIT + file LICENSE

**Depends** R (>= 2.10)

**Imports** httr (>= 1.4.2), jsonlite (>= 1.7.2), tibble (>= 3.1.3)

**Suggests** knitr (>= 1.34), rmarkdown (>= 2.11), spelling (>= 2.2),  
testthat (>= 3.0.0), xml2 (>= 1.3.2)

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Encoding** UTF-8

**Language** en-US

**LazyData** true

**RoxygenNote** 7.1.2

**NeedsCompilation** no

**Author** Kiernan Nicholls [aut, cre, cph]

**Maintainer** Kiernan Nicholls <kiernann@protonmail.com>

**Repository** CRAN

**Date/Publication** 2022-04-06 19:52:29 UTC

## R topics documented:

acquisition_settings . . . . .	3
all_players . . . . .	3
best_roster . . . . .	4
combine_history . . . . .	5
draft_recap . . . . .	6
draft_settings . . . . .	6

espn_games . . . . .	7
fflr . . . . .	8
ffl_id . . . . .	8
ffl_info . . . . .	9
ffl_seasons . . . . .	10
finance_settings . . . . .	10
league_info . . . . .	11
league_members . . . . .	12
league_messages . . . . .	12
league_name . . . . .	13
league_simulation . . . . .	14
league_size . . . . .	15
league_standings . . . . .	15
league_status . . . . .	16
league_teams . . . . .	17
list_players . . . . .	18
live_scoring . . . . .	20
nfl_players . . . . .	20
nfl_schedule . . . . .	21
nfl_teams . . . . .	22
opponent_ranks . . . . .	22
player_acquire . . . . .	23
player_info . . . . .	23
player_news . . . . .	24
player_outlook . . . . .	25
pro_events . . . . .	25
pro_schedule . . . . .	26
pro_scores . . . . .	27
recent_activity . . . . .	27
roster_score . . . . .	28
roster_settings . . . . .	29
schedule_settings . . . . .	29
scoring_settings . . . . .	30
start_roster . . . . .	31
stat_corrections . . . . .	32
team_abbrev . . . . .	32
team_roster . . . . .	33
tidy_schedule . . . . .	34
tidy_scores . . . . .	35
trade_settings . . . . .	36
transaction_counter . . . . .	36

---

acquisition\_settings    *League waiver settings*

---

**Description**

The type, days, and details of a league waiver process.

**Usage**

```
acquisition_settings(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of waiver settings by season.

**See Also**

Other league settings functions: [draft\\_settings\(\)](#), [finance\\_settings\(\)](#), [league\\_info\(\)](#), [league\\_name\(\)](#), [league\\_size\(\)](#), [roster\\_settings\(\)](#), [schedule\\_settings\(\)](#), [scoring\\_settings\(\)](#), [trade\\_settings\(\)](#)

**Examples**

```
acquisition_settings(leagueId = "42654852")
```

---

all\_players    *All fantasy players (deprecated)*

---

**Description**

See [list\\_players\(\)](#).

**Usage**

```
all_players(...)
```

**Arguments**

... Arguments passed to the new `list_players()` function.

**See Also**

Other player functions: `list_players()`, `player_info()`, `player_news()`, `player_outlook()`, `recent_activity()`, `transaction_counter()`

**Examples**

```
## Not run:
all_players()

## End(Not run)
```

---

best_roster	<i>Sort the optimal fantasy roster</i>
-------------	--

---

**Description**

Uses the roster settings for each league to find the best possible combinations of players to score the most fantasy points.

**Usage**

```
best_roster(
  leagueId = ffl_id(),
  useScore = c("actualScore", "projectedScore"),
  scoringPeriodId = NULL,
  ...
)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
useScore	One of "projectedScore" or "actualScore" (default).
scoringPeriodId	Integer week of NFL season. By default, NULL will use the current week (see <code>ffl_week()</code> ). Scoring periods are always one week in length, whereas matchups might be longer.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Details**

If `scoringPeriodId` is the current week (the default), then actual scoring might be incomplete (see `projectedScore` argument).

**Value**

A dataframe (or list) with optimal rosters.

**See Also**

Other roster functions: [roster\\_score\(\)](#), [start\\_roster\(\)](#), [team\\_roster\(\)](#)

**Examples**

```
best_roster(leagueId = "42654852", scoringPeriodId = 1)
```

---

combine_history	<i>Combine league history with current season</i>
-----------------	---

---

**Description**

Runs a function `fun` twice, once with the `leagueHistory` set to `TRUE` and once set to `FALSE`. Combined the output of both runs into a single data frame.

**Usage**

```
combine_history(fun, ...)
```

**Arguments**

<code>fun</code>	A function with the <code>leagueHistory</code> argument.
<code>...</code>	Additional arguments passed to the function used in <code>fun</code> .

**Value**

A data frame of combined outputs.

**Examples**

```
combine_history(tidy_scores, leagueId = "252353")
```

---

draft_recap	<i>Fantasy draft history</i>
-------------	------------------------------

---

**Description**

Return the sequential result of a fantasy draft pick, whether snake or salary cap format.

**Usage**

```
draft_recap(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame(s) of draft picks.

**See Also**

Other league functions: [league\\_members\(\)](#), [league\\_messages\(\)](#), [league\\_standings\(\)](#), [league\\_status\(\)](#), [league\\_teams\(\)](#), [tidy\\_schedule\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
draft_recap(leagueId = "42654852")
```

---

draft_settings	<i>League draft settings</i>
----------------	------------------------------

---

**Description**

The type, date, and pick order of a league draft.

**Usage**

```
draft_settings(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of league draft settings by season.

**See Also**

Other league settings functions: [acquisition\\_settings\(\)](#), [finance\\_settings\(\)](#), [league\\_info\(\)](#), [league\\_name\(\)](#), [league\\_size\(\)](#), [roster\\_settings\(\)](#), [schedule\\_settings\(\)](#), [scoring\\_settings\(\)](#), [trade\\_settings\(\)](#)

**Examples**

```
draft_settings(leagueId = "42654852")
```

---

espn_games	<i>List all fantasy games</i>
------------	-------------------------------

---

**Description**

List all fantasy games

**Usage**

```
espn_games()
```

**Value**

A tibble of fantasy games.

**See Also**

Other Game information: [ffl\\_info\(\)](#), [ffl\\_seasons\(\)](#)

**Examples**

```
espn_games()
```

---

 fflr

*fflr: A package for collecting ESPN fantasy football data*


---

### Description

The fflr package parses the JSON data returned by the ESPN v3 API into tidy data frames for easy analysis. The package also includes data objects for the NFL with players, teams, and the 2021 schedule. The league must first be made viewable to the public by the league manager, which can be done on the basic settings page on the ESPN website. Functions can then take the numeric league ID found in the URL to return data from the API. This league ID can be set as a global option named "fflr.leagueId" with options().

---

 ffl\_id

*Get ESPN fantasy league ID*


---

### Description

Retrieve league ID from global options, as an input, or from a URL.

### Usage

```
ffl_id(leagueId = getOption("fflr.leagueId"), overwrite = FALSE)
```

### Arguments

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
overwrite	logical; If an <code>fflr.leagueId</code> option exists, should it be temporarily changed for your current session.

### Details

Since many users request data from the same ESPN league when using this package, you can use this function to set, call, or extract the unique ESPN league ID. By default, this function uses `getOption("fflr.leagueId")` to look for a default league ID defined in your `options()`. If no such option exists, and one is provided to the `leagueId` argument, the option will be temporarily defined for your current session. If a URL starting with `http` is provided, the numeric league ID will be extracted, defined as the temporary option, and returned as a character string.

### Value

A numeric `leagueId` as a character vector with length one.



## Examples

```
options(fflr.leagueId = "42654852")
ffl_id()
ffl_id(
  leagueId = "https://fantasy.espn.com/football/team?leagueId=42654852",
  overwrite = TRUE
)
```

---

ffl\_info

*Get fantasy football information*

---

## Description

Information on the current fantasy football season, with functions to quickly access and modify certain information (like the current seasonId or scoringPeriodId).

## Usage

```
ffl_info()

ffl_year(offset = 0)

ffl_week(offset = 0)
```

## Arguments

offset            Add negative or positive values.

## Value

A list of season information.

## See Also

Other Game information: [espn\\_games\(\)](#), [ffl\\_seasons\(\)](#)

## Examples

```
str(ffl_info())
Sys.time()
ffl_year()
ffl_week(-1)
```

---

ffl_seasons	<i>List past fantasy football seasons</i>
-------------	---

---

**Description**

List past fantasy football seasons

**Usage**

```
ffl_seasons()
```

**Value**

A tibble of fantasy football seasons.

**See Also**

Other Game information: [espn\\_games\(\)](#), [ffl\\_info\(\)](#)

**Examples**

```
ffl_seasons()
```

---

finance_settings	<i>League finance settings</i>
------------------	--------------------------------

---

**Description**

The off-site fees assigned to various roster moves and transactions.

**Usage**

```
finance_settings(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of finance settings by season.

**See Also**

Other league settings functions: [acquisition\\_settings\(\)](#), [draft\\_settings\(\)](#), [league\\_info\(\)](#), [league\\_name\(\)](#), [league\\_size\(\)](#), [roster\\_settings\(\)](#), [schedule\\_settings\(\)](#), [scoring\\_settings\(\)](#), [trade\\_settings\(\)](#)

**Examples**

```
finance_settings(leagueId = "42654852")
```

---

league_info	<i>League information</i>
-------------	---------------------------

---

**Description**

Basic information on a ESPN fantasy football league, like the name, size, and season length.

**Usage**

```
league_info(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of league information by season.

**See Also**

Other league settings functions: [acquisition\\_settings\(\)](#), [draft\\_settings\(\)](#), [finance\\_settings\(\)](#), [league\\_name\(\)](#), [league\\_size\(\)](#), [roster\\_settings\(\)](#), [schedule\\_settings\(\)](#), [scoring\\_settings\(\)](#), [trade\\_settings\(\)](#)

**Examples**

```
league_info(leagueId = "42654852")
```

---

league_members	<i>Fantasy league teams</i>
----------------	-----------------------------

---

**Description**

The teams in a league and their owners.

**Usage**

```
league_members(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A dataframe (or list) with league members.

**See Also**

Other league functions: [draft\\_recap\(\)](#), [league\\_messages\(\)](#), [league\\_standings\(\)](#), [league\\_status\(\)](#), [league\\_teams\(\)](#), [tidy\\_schedule\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
league_members(leagueId = "42654852")
```

---

league_messages	<i>Fantasy league teams</i>
-----------------	-----------------------------

---

**Description**

The emails, chats, notes, and messages sent by league members.

**Usage**

```
league_messages(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A tibble of messages.

**See Also**

Other league functions: [draft\\_recap\(\)](#), [league\\_members\(\)](#), [league\\_standings\(\)](#), [league\\_status\(\)](#), [league\\_teams\(\)](#), [tidy\\_schedule\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
league_messages(leagueId = "42654852")
```

---

league_name	<i>League name</i>
-------------	--------------------

---

**Description**

League name

**Usage**

```
league_name(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A character vector.

**See Also**

Other league settings functions: [acquisition\\_settings\(\)](#), [draft\\_settings\(\)](#), [finance\\_settings\(\)](#), [league\\_info\(\)](#), [league\\_size\(\)](#), [roster\\_settings\(\)](#), [schedule\\_settings\(\)](#), [scoring\\_settings\(\)](#), [trade\\_settings\(\)](#)

**Examples**

```
league_name(leagueId = "42654852")
```

---

league_simulation	<i>League standing simulation</i>
-------------------	-----------------------------------

---

**Description**

The ESPN algorithm simulates the entire season according to the projection and matchup schedule to calculate the probability of a team winning their division and making the playoffs.

**Usage**

```
league_simulation(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of simulated team standings.

**Examples**

```
league_simulation(leagueId = "42654852")
```

---

league_size	<i>League size</i>
-------------	--------------------

---

**Description**

League size

**Usage**

```
league_size(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of league size by season.

**See Also**

Other league settings functions: [acquisition\\_settings\(\)](#), [draft\\_settings\(\)](#), [finance\\_settings\(\)](#), [league\\_info\(\)](#), [league\\_name\(\)](#), [roster\\_settings\(\)](#), [schedule\\_settings\(\)](#), [scoring\\_settings\(\)](#), [trade\\_settings\(\)](#)

**Examples**

```
league_size(leagueId = "42654852")
```

---

league_standings	<i>League standings</i>
------------------	-------------------------

---

**Description**

Return the current and projected standings, win streak, total wins, losses, and points scored for and against each team.

**Usage**

```
league_standings(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of team standings.

**See Also**

Other league functions: [draft\\_recap\(\)](#), [league\\_members\(\)](#), [league\\_messages\(\)](#), [league\\_status\(\)](#), [league\\_teams\(\)](#), [tidy\\_schedule\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
league_standings(leagueId = "42654852")
```

---

league_status	<i>League status</i>
---------------	----------------------

---

**Description**

Current information about a league: the date activated, current week, starting week, final week, past seasons, teams joined, and waiver status.

**Usage**

```
league_status(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of league status by season.



**See Also**

Other league functions: [draft\\_recap\(\)](#), [league\\_members\(\)](#), [league\\_messages\(\)](#), [league\\_standings\(\)](#), [league\\_teams\(\)](#), [tidy\\_schedule\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
league_status(leagueId = "42654852")
```

---

league_teams	<i>Fantasy league teams</i>
--------------	-----------------------------

---

**Description**

The teams in a league and their owners.

**Usage**

```
league_teams(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Details**

If any team has multiple owners, the `memberId` column will be a list of unique owner member ID strings per team (see [league\\_members\(\)](#)).

**Value**

A dataframe (or list) with league teams.

**See Also**

Other league functions: [draft\\_recap\(\)](#), [league\\_members\(\)](#), [league\\_messages\(\)](#), [league\\_standings\(\)](#), [league\\_status\(\)](#), [tidy\\_schedule\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
league_teams(leagueId = "42654852")
```

---

list_players	<i>Find fantasy players</i>
--------------	-----------------------------

---

### Description

Filter fantasy players by their position, availability, professional team, and/or injury status. Sort and limit the responses in the same way as is done in the ESPN Fantasy Football website.

### Usage

```
list_players(
  leagueId = ffl_id(),
  sort = "ROST",
  position = NULL,
  status = "AVAILABLE",
  injured = NULL,
  proTeam = NULL,
  scoreType = c("STANDARD", "PPR"),
  limit = 50
)
```

### Arguments

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
sort	The column from which to sort the data. Options match those on the ESPN website: <ul style="list-style-type: none"> <li>• "PLAYER" = Alphabetical by player name</li> <li>• "PROJ" = Projection is ESPN's projected fantasy score for a player's upcoming game.</li> <li>• "SCORE" = Actual score for <code>scoringPeriodId</code></li> <li>• "OPRK" = Opponent Rank shows how a player's upcoming NFL opponent performs against that player's position. Low numbers mean it may be a tough opponent; high numbers an easier opponent.</li> <li>• "START" = Start Percentage shows the number of fantasy leagues a player is started in divided by the number of leagues he is eligible in. This helps indicate how the public views a player.</li> <li>• "ROST" = Rostered Percentage shows the number of fantasy leagues in which a player is on a roster divided by the total number of fantasy leagues. This helps indicate how the public views a player.</li> <li>• "CHANGE" = Plus/Minus shows the change in %ROST over the last week. This will help show which players are hot and cold at a given moment.</li> <li>• "PRK" = Position Rank shows how a player stacks up against other players at his position. No. 1 is best.</li> <li>• "FPTS" = Total fantasy points scored thus far in the season.</li> </ul>

	<ul style="list-style-type: none"> <li>• "AVG" = Average fantasy points scored in each game started.</li> <li>• "LAST" = Last shows the player's fantasy score in his team's last game.</li> </ul>
position	<p>Abbreviation of player positions to filter, NULL for all:</p> <ul style="list-style-type: none"> <li>• "QB" = Quarterback</li> <li>• "RB" = Running Back</li> <li>• "WR" = Wide Receiver</li> <li>• "TE" = Tight End</li> <li>• "FLEX" = Running Backs, Wide Receivers and Tight Ends can be used in this position</li> <li>• "D/ST" = Defense and Special Teams</li> <li>• "K" = Kicker</li> </ul>
status	<p>Availability status of player, one or more from:</p> <ul style="list-style-type: none"> <li>• "ALL"</li> <li>• "AVAILABLE" (default)</li> <li>• "FREEAGENT"</li> <li>• "WAIVERS"</li> <li>• "ONTEAM"</li> </ul>
injured	Whether to return only injured or healthy players. Use NULL (default) for all players, TRUE for injured players, and FALSE for healthy players.
proTeam	The abbreviation or ID of the professional team from which players should be returned. See <code>pro_teams()</code> for a list of all possible team abbreviations.
scoreType	The type of scoring used: "STANDARD" or "PPR."
limit	The limit of players to return. Use "" or NULL to return all. Defaults to 50, which is the default limit used by ESPN. Removing the limit can make the request take a long time.

**Value**

A data frame of players.

**See Also**

Other player functions: [all\\_players\(\)](#), [player\\_info\(\)](#), [player\\_news\(\)](#), [player\\_outlook\(\)](#), [recent\\_activity\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
list_players("42654852", proTeam = "Mia", sort = "START", limit = 3)
```

---

live_scoring	<i>Live matchup scoreboard</i>
--------------	--------------------------------

---

**Description**

The current and projected score for each ongoing match.

**Usage**

```
live_scoring(leagueId = nfl_id(), yetToPlay = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
yetToPlay	If TRUE, <a href="#">pro_schedule()</a> and the "mRoster" view are called to determine how many starting players have <i>yet</i> started playing.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of scores by period.

**See Also**

Other scoring functions: [tidy\\_scores\(\)](#)

**Examples**

```
live_scoring(leagueId = "42654852", yetToPlay = FALSE)
```

---

nfl_players	<i>2021 NFL Players</i>
-------------	-------------------------

---

**Description**

All available ESPN fantasy football players as of the 2021 season, week 1.

**Usage**

```
nfl_players
```

**Format**

A data frame with 1,063 rows and 11 variables:

**playerId** Unique ESPN player ID  
**firstName** First name  
**lastName** Last name  
**proTeam** Professional NFL team  
**defaultPositionId** Position: QB, RB, WR, TE, D/ST  
**jersey** Jersey number  
**weight** Weight in integer pounds  
**height** Height in integer inches  
**age** Current age in integer year  
**dateOfBirth** Date of birth  
**debutYear** Season debuted in league ...

**Source**

<http://sports.core.api.espn.com/v2/sports/football/leagues/nfl/seasons/2021/athletes/>

---

nfl\_schedule

*2021 NFL Schedule*

---

**Description**

The 2021 NFL season schedule by team, as of September 10th.

**Usage**

nfl\_schedule

**Format**

A data frame with 544 rows and 6 variables:

**seasonId** Season year  
**scoringPeriodId** Scoring period  
**matchupId** Unique ID for professional matchup  
**proTeam** Professional team abbreviation  
**opponent** Professional team opponent  
**isHome** Whether this is the home team  
**date** Matchup start date and time ...

**Source**

[https://fantasy.espn.com/apis/v3/games/ffl/seasons/2021?view=proTeamSchedules\\_wl](https://fantasy.espn.com/apis/v3/games/ffl/seasons/2021?view=proTeamSchedules_wl)

---

nfl\_teams

2021 NFL Teams

---

**Description**

The 32 professional NFL teams as of the 2021 season.

**Usage**

```
nfl_teams
```

**Format**

A data frame with 33 rows and 6 columns:

**proTeamId** Unique team ID

**abbrev** Professional team abbreviation

**location** Professional team geographic location

**name** Professional team full nickname

**byeWeek** Bye week, no game played

**conference** NFL conference ...

**Source**

[https://fantasy.espn.com/apis/v3/games/ffl/seasons/2021?view=proTeamSchedules\\_wl](https://fantasy.espn.com/apis/v3/games/ffl/seasons/2021?view=proTeamSchedules_wl)

---

opponent\_ranks

*NFL team performance against positions*


---

**Description**

The average opposition team point differential by position.

**Usage**

```
opponent_ranks(leagueId = ffl_id())
```

**Arguments**

**leagueId** Numeric league ID or ESPN fantasy page URL. Defaults to `getOption("fflr.leagueId")`. Function fails if no ID is found.

**Value**

A data frame of team performance against position.

**Examples**

```
opponent_ranks()
```

---

player_acquire	<i>Roster acquisition history</i>
----------------	-----------------------------------

---

**Description**

The date and method of each player's acquisition onto a fantasy roster.

**Usage**

```
player_acquire(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of roster players with acquisition method and date.

**Examples**

```
player_acquire(leagueId = "42654852")
```

---

player_info	<i>Individual player information</i>
-------------	--------------------------------------

---

**Description**

Individual player information

**Usage**

```
player_info(playerId)
```

**Arguments**

playerId	A single player ID number.
----------	----------------------------

**Value**

A list or row of a single player's information.

**See Also**

Other player functions: [all\\_players\(\)](#), [list\\_players\(\)](#), [player\\_news\(\)](#), [player\\_outlook\(\)](#), [recent\\_activity\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
player_info(playerId = 15847)
```

---

player\_news

*Player news*

---

**Description**

The free and premium ESPN stories on given players. A maximum of 50 stories can be returned at a time.

**Usage**

```
player_news(playerId, parseHTML = FALSE)
```

**Arguments**

playerId      A single player ID number.  
parseHTML     Should HTML stories be parsed with [xml2::read\\_html\(\)](#)?

**Value**

A data frame of news stories.

**See Also**

Other player functions: [all\\_players\(\)](#), [list\\_players\(\)](#), [player\\_info\(\)](#), [player\\_outlook\(\)](#), [recent\\_activity\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
player_news(playerId = "15847")
```



---

player_outlook	<i>Player outlooks</i>
----------------	------------------------

---

**Description**

All available weekly ESPN outlook writeups for NFL players.

**Usage**

```
player_outlook(leagueId = ffl_id(), limit = 50)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
limit	The limit of players to return. Use "" or NULL to return all. Defaults to 50, which is the default limit used by ESPN. Removing the limit can make the request take a long time.

**Value**

A data frame of player outlooks by scoring period.

**See Also**

Other player functions: [all\\_players\(\)](#), [list\\_players\(\)](#), [player\\_info\(\)](#), [player\\_news\(\)](#), [recent\\_activity\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
player_outlook()
```

---

pro_events	<i>Professional games</i>
------------	---------------------------

---

**Description**

Data on the status of NFL games, including scores and odds, kickoff time, and broadcast information.

**Usage**

```
pro_events()
```

**Value**

A data frame of NFL events.

**See Also**

Other professional football functions: [pro\\_schedule\(\)](#)

**Examples**

```
pro_events()
```

---

pro_schedule	<i>Professional schedule</i>
--------------	------------------------------

---

**Description**

The opponents each team faces every week in a regular season.

**Usage**

```
pro_schedule(seasonId = ffl_year())
```

**Arguments**

seasonId      Season schedule (2004-present), defaults to [ffl\\_year\(\)](#).

**Value**

Data frame of team opponents by week.

**See Also**

Other professional football functions: [pro\\_events\(\)](#)

**Examples**

```
pro_schedule(seasonId = ffl_year(-2))
```

---

pro_scores	<i>Professional scores</i>
------------	----------------------------

---

**Description**

The tidy data frame of scores by team.

**Usage**

```
pro_scores()
```

**Value**

A data frame of NFL scores.

**Examples**

```
pro_scores()
```

---

recent_activity	<i>Roster moves</i>
-----------------	---------------------

---

**Description**

The individual proposed and executed transactions, trades, and waiver moves.

**Usage**

```
recent_activity(
  leagueId = ffl_id(),
  leagueHistory = FALSE,
  scoringPeriodId = NULL,
  ...
)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
scoringPeriodId	Integer week of NFL season. By default, NULL will use the current week (see <a href="#">ffl_week()</a> ). Scoring periods are always one week in length, whereas matchups might be longer.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Details**

As of November 2021, activity data related to trades coming from the API is flawed. The `items` list column containing the players involved in a trade will only contain data for *rejected* trades (with an `executionType` of "CANCEL"). For accepted and upheld trades, that `items` element is NULL or an empty list. This flaw comes from the API itself, not processing done by this package.

**Value**

A data frame of transactions and roster moves.

**See Also**

Other player functions: [all\\_players\(\)](#), [list\\_players\(\)](#), [player\\_info\(\)](#), [player\\_news\(\)](#), [player\\_outlook\(\)](#), [transaction\\_counter\(\)](#)

**Examples**

```
recent_activity(leagueId = "42654852", scoringPeriodId = 2)
```

---

roster_score	<i>Sum of starting scores in a roster</i>
--------------	---

---

**Description**

For a given roster tibble, sum the starting scores.

**Usage**

```
roster_score(roster, useScore = c("actualScore", "projectedScore"))
```

**Arguments**

`roster` A roster data frame from [team\\_roster\(\)](#).  
`useScore` One of "projectedScore" or "actualScore" (default).

**Value**

A starting score as double.

**See Also**

Other roster functions: [best\\_roster\(\)](#), [start\\_roster\(\)](#), [team\\_roster\(\)](#)

**Examples**

```
roster_score(team_roster(leagueId = "42654852"))[[1]]
```

---

roster_settings	<i>League roster settings</i>
-----------------	-------------------------------

---

**Description**

The number of players and positions on a fantasy football roster.

**Usage**

```
roster_settings(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of league roster settings by season.

**See Also**

Other league settings functions: [acquisition\\_settings\(\)](#), [draft\\_settings\(\)](#), [finance\\_settings\(\)](#), [league\\_info\(\)](#), [league\\_name\(\)](#), [league\\_size\(\)](#), [schedule\\_settings\(\)](#), [scoring\\_settings\(\)](#), [trade\\_settings\(\)](#)

**Examples**

```
roster_settings(leagueId = "42654852")
```

---

schedule_settings	<i>League schedule settings</i>
-------------------	---------------------------------

---

**Description**

The length of a fantasy season and the match periods for each week.

**Usage**

```
schedule_settings(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of league schedule settings by season.

**See Also**

Other league settings functions: `acquisition_settings()`, `draft_settings()`, `finance_settings()`, `league_info()`, `league_name()`, `league_size()`, `roster_settings()`, `scoring_settings()`, `trade_settings()`

**Examples**

```
schedule_settings(leagueId = "42654852")
```

---

scoring_settings	<i>League scoring settings</i>
------------------	--------------------------------

---

**Description**

The scoring system used and points awarded for various actions.

**Usage**

```
scoring_settings(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of league scoring settings by season.

**See Also**

Other league settings functions: [acquisition\\_settings\(\)](#), [draft\\_settings\(\)](#), [finance\\_settings\(\)](#), [league\\_info\(\)](#), [league\\_name\(\)](#), [league\\_size\(\)](#), [roster\\_settings\(\)](#), [schedule\\_settings\(\)](#), [trade\\_settings\(\)](#)

**Examples**

```
scoring_settings(leagueId = "42654852")
```

---

start\_roster

*Starting roster*

---

**Description**

The starting 9 man roster using standard roster slots. In the future this function may be adapted to take roster slots from [roster\\_settings\(\)](#).

**Usage**

```
start_roster(roster)
```

**Arguments**

roster            A roster data frame from [team\\_roster\(\)](#).

**Value**

A data frame of starters on a roster.

**See Also**

Other roster functions: [best\\_roster\(\)](#), [roster\\_score\(\)](#), [team\\_roster\(\)](#)

**Examples**

```
start_roster(team_roster(leagueId = "42654852"))[[1]]
```

---

stat_corrections	<i>Stat corrections</i>
------------------	-------------------------

---

**Description**

Weekly retroactive stat corrections by player.

**Usage**

```
stat_corrections(date = Sys.Date(), limit = 100)
```

**Arguments**

date	A date in the scoring week to return. Defaults to system date.
limit	The limit of corrections to return. Use "" or NULL to return all. Defaults to 100, which is the default limit used by ESPN. Removing the limit can make the request take a long time.

**Value**

A data frame of stat corrections.

**Examples**

```
stat_corrections(date = "2021-09-13")
```

---

team_abbrev	<i>Convert team ID to abbreviation</i>
-------------	--

---

**Description**

Convert team ID to abbreviation

**Usage**

```
team_abbrev(teamId, teams = league_teams(leagueId = nfl_id()))
```

**Arguments**

teamId	A integer vector of team numbers to convert.
teams	A table of teams, like that from <a href="#">league_teams()</a> .

**Value**

A factor vector of team abbreviations.



**Examples**

```
team_abbrev(teamId = 2, teams = league_teams(leagueId = "42654852"))
```

---

team_roster	<i>Fantasy team rosters</i>
-------------	-----------------------------

---

**Description**

The roster of all teams in a league.

**Usage**

```
team_roster(
  leagueId = ffl_id(),
  leagueHistory = FALSE,
  scoringPeriodId = NULL,
  ...
)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
scoringPeriodId	Integer week of NFL season. By default, NULL will use the current week (see <a href="#">ffl_week()</a> ). Scoring periods are always one week in length, whereas matchups might be longer.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A dataframe (or list) with league teams.

**See Also**

Other roster functions: [best\\_roster\(\)](#), [roster\\_score\(\)](#), [start\\_roster\(\)](#)

**Examples**

```
team_roster(leagueId = "42654852", scoringPeriodId = 1)
```

---

tidy_schedule	<i>Fantasy match schedule</i>
---------------	-------------------------------

---

## Description

The opponents each team faces every week in a fantasy regular season. Returned in a tidy format where each row is a single team with an indication of home-away status. There are two rows per matchup, one for each team.

## Usage

```
tidy_schedule(leagueId = ffl_id(), leagueHistory = FALSE, ...)  
tidy_matchups(...)
```

## Arguments

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the <code>leagueHistory</code> version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

## Value

A data frame(s) of match opponents.

## See Also

Other league functions: [draft\\_recap\(\)](#), [league\\_members\(\)](#), [league\\_messages\(\)](#), [league\\_standings\(\)](#), [league\\_status\(\)](#), [league\\_teams\(\)](#), [transaction\\_counter\(\)](#)

## Examples

```
tidy_schedule(leagueId = "42654852")
```

---

tidy_scores	<i>Fantasy matchup scores</i>
-------------	-------------------------------

---

## Description

The score of each team in a matchup or scoring period and the match outcome.

## Usage

```
tidy_scores(leagueId = ffl_id(), leagueHistory = FALSE, useMatchup = TRUE, ...)
```

## Arguments

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the <code>leagueHistory</code> version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
useMatchup	logical; Whether scoring should be summarized by <code>matchupPeriodId</code> (default) or <code>scoringPeriodId</code> . The later always relates to a single week of the NFL season, while fantasy matchups might span several scoring periods, especially in the playoffs.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside <code>view</code> .

## Details

`expectedWins` are calculated by comparing a team score against all *other* scores for a given matchup period. This statistic expresses how a team would fair if the schedule was random. The highest scoring team is thus expected to earn 1 win and the lowest scoring team would expect to win 0 matchups.

## Value

A tidy data frame of scores by team and matchup/scoring period.

## See Also

Other scoring functions: [live\\_scoring\(\)](#)

## Examples

```
tidy_scores(leagueId = "42654852", useMatchup = FALSE)
```

---

trade_settings	<i>League trade settings</i>
----------------	------------------------------

---

**Description**

The time each trade can stand, votes needed to veto, and season deadline.

**Usage**

```
trade_settings(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of league trade settings by season.

**See Also**

Other league settings functions: [acquisition\\_settings\(\)](#), [draft\\_settings\(\)](#), [finance\\_settings\(\)](#), [league\\_info\(\)](#), [league\\_name\(\)](#), [league\\_size\(\)](#), [roster\\_settings\(\)](#), [schedule\\_settings\(\)](#), [scoring\\_settings\(\)](#)

**Examples**

```
trade_settings(leagueId = "42654852")
```

---

transaction_counter	<i>League transactions</i>
---------------------	----------------------------

---

**Description**

Summary of transactions and roster changes made during a season by team.

**Usage**

```
transaction_counter(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

```
budget_summary(leagueId = ffl_id(), leagueHistory = FALSE, ...)
```

**Arguments**

leagueId	Numeric league ID or ESPN fantasy page URL. Defaults to <code>getOption("fflr.leagueId")</code> . Function fails if no ID is found.
leagueHistory	logical; Should the leagueHistory version of the API be called? If TRUE, a list of results is returned, with one element for each historical year of the league.
...	Additional queries passed to <code>httr::GET()</code> . Arguments are converted to a named list and passed to query alongside view.

**Value**

A data frame of transaction counts by team.

**See Also**

Other league functions: [draft\\_recap\(\)](#), [league\\_members\(\)](#), [league\\_messages\(\)](#), [league\\_standings\(\)](#), [league\\_status\(\)](#), [league\\_teams\(\)](#), [tidy\\_schedule\(\)](#)

Other player functions: [all\\_players\(\)](#), [list\\_players\(\)](#), [player\\_info\(\)](#), [player\\_news\(\)](#), [player\\_outlook\(\)](#), [recent\\_activity\(\)](#)

**Examples**

```
transaction_counter(leagueId = "42654852")
```

# Index

- \* **Game information**
  - espn\_games, 7
  - ffl\_info, 9
  - ffl\_seasons, 10
- \* **datasets**
  - nfl\_players, 20
  - nfl\_schedule, 21
  - nfl\_teams, 22
- \* **league functions**
  - draft\_recap, 6
  - league\_members, 12
  - league\_messages, 12
  - league\_standings, 15
  - league\_status, 16
  - league\_teams, 17
  - tidy\_schedule, 34
  - transaction\_counter, 36
- \* **league settings functions**
  - acquisition\_settings, 3
  - draft\_settings, 6
  - finance\_settings, 10
  - league\_info, 11
  - league\_name, 13
  - league\_size, 15
  - roster\_settings, 29
  - schedule\_settings, 29
  - scoring\_settings, 30
  - trade\_settings, 36
- \* **player functions**
  - all\_players, 3
  - list\_players, 18
  - player\_info, 23
  - player\_news, 24
  - player\_outlook, 25
  - recent\_activity, 27
  - transaction\_counter, 36
- \* **professional football functions**
  - pro\_events, 25
  - pro\_schedule, 26
- \* **roster functions**
  - best\_roster, 4
  - roster\_score, 28
  - start\_roster, 31
  - team\_roster, 33
- \* **scoring functions**
  - live\_scoring, 20
  - tidy\_scores, 35
- acquisition\_settings, 3, 7, 11, 14, 15, 29–31, 36
- all\_players, 3, 19, 24, 25, 28, 37
- best\_roster, 4, 28, 31, 33
- budget\_summary(transaction\_counter), 36
- combine\_history, 5
- draft\_recap, 6, 12, 13, 16, 17, 34, 37
- draft\_settings, 3, 6, 11, 14, 15, 29–31, 36
- espn\_games, 7, 9, 10
- ffl\_id, 8
- ffl\_info, 7, 9, 10
- ffl\_seasons, 7, 9, 10
- ffl\_week(ffl\_info), 9
- ffl\_week(), 4, 27, 33
- ffl\_year(ffl\_info), 9
- ffl\_year(), 26
- fflr, 8
- finance\_settings, 3, 7, 10, 11, 14, 15, 29–31, 36
- httr::GET(), 3, 4, 6, 7, 10–17, 20, 23, 27, 29, 30, 33–37
- league\_info, 3, 7, 11, 11, 14, 15, 29–31, 36
- league\_members, 6, 12, 13, 16, 17, 34, 37
- league\_members(), 17
- league\_messages, 6, 12, 12, 16, 17, 34, 37

league\_name, [3](#), [7](#), [11](#), [13](#), [15](#), [29–31](#), [36](#)  
league\_simulation, [14](#)  
league\_size, [3](#), [7](#), [11](#), [14](#), [15](#), [29–31](#), [36](#)  
league\_standings, [6](#), [12](#), [13](#), [15](#), [17](#), [34](#), [37](#)  
league\_status, [6](#), [12](#), [13](#), [16](#), [16](#), [17](#), [34](#), [37](#)  
league\_teams, [6](#), [12](#), [13](#), [16](#), [17](#), [17](#), [34](#), [37](#)  
league\_teams(), [32](#)  
list\_players, [4](#), [18](#), [24](#), [25](#), [28](#), [37](#)  
list\_players(), [3](#), [4](#)  
live\_scoring, [20](#), [35](#)

nfl\_players, [20](#)  
nfl\_schedule, [21](#)  
nfl\_teams, [22](#)

opponent\_ranks, [22](#)

player\_acquire, [23](#)  
player\_info, [4](#), [19](#), [23](#), [24](#), [25](#), [28](#), [37](#)  
player\_news, [4](#), [19](#), [24](#), [24](#), [25](#), [28](#), [37](#)  
player\_outlook, [4](#), [19](#), [24](#), [25](#), [28](#), [37](#)  
pro\_events, [25](#), [26](#)  
pro\_schedule, [26](#), [26](#)  
pro\_schedule(), [20](#)  
pro\_scores, [27](#)

recent\_activity, [4](#), [19](#), [24](#), [25](#), [27](#), [37](#)  
roster\_score, [5](#), [28](#), [31](#), [33](#)  
roster\_settings, [3](#), [7](#), [11](#), [14](#), [15](#), [29](#), [30](#), [31](#),  
[36](#)  
roster\_settings(), [31](#)

schedule\_settings, [3](#), [7](#), [11](#), [14](#), [15](#), [29](#), [29](#),  
[31](#), [36](#)  
scoring\_settings, [3](#), [7](#), [11](#), [14](#), [15](#), [29](#), [30](#),  
[30](#), [36](#)  
start\_roster, [5](#), [28](#), [31](#), [33](#)  
stat\_corrections, [32](#)

team\_abbrev, [32](#)  
team\_roster, [5](#), [28](#), [31](#), [33](#)  
team\_roster(), [28](#), [31](#)  
tidy\_matchups(tidy\_schedule), [34](#)  
tidy\_schedule, [6](#), [12](#), [13](#), [16](#), [17](#), [34](#), [37](#)  
tidy\_scores, [20](#), [35](#)  
trade\_settings, [3](#), [7](#), [11](#), [14](#), [15](#), [29–31](#), [36](#)  
transaction\_counter, [4](#), [6](#), [12](#), [13](#), [16](#), [17](#),  
[19](#), [24](#), [25](#), [28](#), [34](#), [36](#)

xml2::read\_html(), [24](#)