

Package: nhlscraper (via r-universe)

May 31, 2026

Title Scraper for National Hockey League Data

Version 0.6.1.9000

Description Scrapes and cleans data from the 'NHL' and 'ESPN' APIs into data.frames and lists. Wraps 125+ endpoints documented in <https://github.com/RentoSaijo/nhlscraper/wiki> from high-level multi-season summaries and award winners to low-level decisecond replays and bookmakers' odds, making them more accessible. Features cleaning and visualization tools, primarily for play-by-plays.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports httr2 (>= 1.2.0), jsonlite (>= 2.0.0), xml2 (>= 1.5.0), arrow (>= 23.0.0)

Suggests testthat (>= 3.0.0), knitr (>= 1.50.0), rmarkdown (>= 2.29.0)

Config/testthat/edition 3

VignetteBuilder knitr

URL <https://rentosaijo.github.io/nhlscraper/>,
<https://github.com/RentoSaijo/nhlscraper>

BugReports <https://github.com/RentoSaijo/nhlscraper/issues>

Copyright Copyright: NHL and the NHL Shield are registered trademarks of the National Hockey League. NHL and NHL team marks are the property of the NHL and its teams.

Config/pak/sysreqs cmake libxml2-dev libssl-dev

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

Date/Publication 2026-05-24 15:51:55 UTC

RemoteUrl <https://github.com/rentosaijo/nhlscraper>

RemoteRef HEAD

RemoteSha d315527302e3afe2f547e3230d55a4aba8d5d0f8

Contents

add_deltas	6
add_goalie_biometrics	7
add_shift_times	8
add_shooter_biometrics	8
attendance	9
award_winners	9
awards	10
boxscore	10
bracket	11
calculate_expected_goals	11
coach_career_statistics	12
coach_franchise_statistics	13
coaches	13
combine_reports	14
contracts	14
countries	15
draft_picks	15
draft_prospects	16
draft_rankings	16
draft_tracker	17
drafts	17
draw_NHL_rink	18
espn_futures	18
espn_game_odds	19
espn_game_summary	19
espn_games	20
espn_injuries	20
espn_play_by_play	21
espn_player_summary	21
espn_players	22
espn_team_summary	22
espn_teams	23
espn_transactions	23
expansion_draft_picks	24
expansion_drafts	24
franchise_playoff_situational_results	25
franchise_season_statistics	25
franchise_statistics	26
franchise_team_statistics	26
franchise_versus_franchise	27
franchises	27
game_odds	28
game_rosters	28
game_type_now	29
games	29
gc_play_by_play	30

gc_play_by_play_raw	30
gc_play_by_plays	31
gc_play_by_plays_raw	32
gc_summary	32
general_managers	33
get_attendance	33
get_award_winners	34
get_awards	34
get_bracket	34
get_configuration	35
get_countries	35
get_draft_picks	35
get_draft_rankings	36
get_draft_tracker	36
get_drafts	37
get_espn_athlete	37
get_espn_athletes	37
get_espn_coach	38
get_espn_coach_career	38
get_espn_coaches	38
get_espn_event	39
get_espn_event_odds	39
get_espn_event_officials	40
get_espn_event_play_by_play	40
get_espn_event_stars	40
get_espn_events	41
get_espn_futures	41
get_espn_injuries	41
get_espn_team	42
get_espn_teams	42
get_espn_transactions	42
get_franchise_season_by_season	43
get_franchise_team_totals	43
get_franchise_vs_franchise	44
get_franchises	44
get_game_boxscore	45
get_game_landing	45
get_game_story	46
get_games	46
get_gc_play_by_play	47
get_glossary	47
get_goalie_leaders	48
get_goalie_milestones	48
get_goalie_statistics	49
get_goalies	49
get_officials	49
get_partner_odds	50
get_player_game_log	50

get_player_landing	51
get_players	51
get_schedule	52
get_scoreboards	52
get_scores	53
get_season_now	53
get_seasons	53
get_series	54
get_series_schedule	54
get_shift_charts	54
get_skater_leaders	55
get_skater_milestones	55
get_skater_statistics	56
get_skaters	56
get_spotlight_players	56
get_standings	57
get_standings_information	57
get_streams	58
get_team_prospects	58
get_team_roster	59
get_team_roster_statistics	59
get_team_schedule	60
get_team_scoreboard	60
get_team_seasons	61
get_team_statistics	61
get_teams	61
get_tv_schedule	62
get_venues	62
get_wsc_play_by_play	63
glossary	63
goalie_edge_five_versus_five	64
goalie_edge_leaders	65
goalie_edge_save_percentage	66
goalie_edge_seasons	67
goalie_edge_shot_location	67
goalie_edge_summary	68
goalie_game_report	69
goalie_game_scoring	70
goalie_game_statistics	70
goalie_leaders	71
goalie_milestones	72
goalie_regular_statistics	72
goalie_report_configurations	73
goalie_scoring	73
goalie_season_report	74
goalie_season_statistics	75
goalie_series_statistics	75
goalie_statistics	76

ig_game_cumulative_expected_goals	76
ig_game_shot_locations	77
location	78
lottery_odds	78
nhlscraper	79
officials	80
penalty_shots	81
ping	81
player_game_log	82
player_seasons	82
player_summary	83
players	84
playoff_season_statistics	84
ps	85
replay	85
replays	86
roster	86
roster_statistics	87
schedule	88
scores	88
season_now	89
seasons	89
series	90
series_schedule	90
shift_chart	91
shift_chart_summaries	92
shift_chart_summary	92
shift_charts	93
shifts	93
skater_edge_leaders	94
skater_edge_seasons	94
skater_edge_shot_location	95
skater_edge_shot_speed	96
skater_edge_skating_distance	97
skater_edge_skating_speed	98
skater_edge_summary	99
skater_edge_zone_time	100
skater_game_report	101
skater_leaders	102
skater_milestones	102
skater_playoff_statistics	103
skater_regular_statistics	104
skater_report_configurations	104
skater_season_report	105
skater_season_statistics	106
skater_series_statistics	106
skater_statistics	107
spotlight_players	107

standings	108
standings_rules	108
streams	109
team_edge_leaders	109
team_edge_seasons	110
team_edge_shot_location	110
team_edge_shot_speed	111
team_edge_skating_distance	112
team_edge_skating_speed	113
team_edge_summary	114
team_edge_zone_time	115
team_game_report	116
team_logos	117
team_month_schedule	118
team_prospects	118
team_report_configurations	119
team_season_report	120
team_season_schedule	121
team_season_statistics	121
team_seasons	122
team_week_schedule	123
teams	123
tv_schedule	124
venues	124
wsc_play_by_play	125
wsc_play_by_play_raw	126
wsc_play_by_plays	126
wsc_play_by_plays_raw	127
wsc_summary	127
x_game_cumulative_expected_goals	128
x_game_shot_locations	129

Index**130**

add_deltas

*Add event-to-event deltas to a play-by-play***Description**

add_deltas() adds event-to-event deltas in raw and normalized x/y, distance, and angle for a play-by-play. Sequences are bounded by faceoffs: each sequence begins at a faceoff, faceoff rows do not look backward across the boundary, and subsequent events are compared to the most recent prior valid-spatial event in the same faceoff-bounded sequence. Shootout and penalty-shot rows (0101/1010) are left as NA and do not serve as anchors for later rows. When multiple events in a sequence share the same recorded second, zero-time denominators used internally are replaced by $1 / n$, where n is the number of events in that same second within the sequence.

Usage

```
add_deltas(play_by_play)
```

Arguments

play_by_play data.frame of play-by-play(s) using the current public schema returned by [gc_play_by_play\(\)](#), [gc_play_by_plays\(\)](#), [wsc_play_by_play\(\)](#), or [wsc_play_by_plays\(\)](#)

Value

data.frame with one row per event (play) and added columns: eventIdPrev, secondsElapsedInSequence, dSecondsElapsedInSequence, dxCoord, dyCoord, dxCoordNorm, dyCoordNorm, dDistance, dAngle, dxCoordPerSecond, dyCoordPerSecond, dxCoordNormPerSecond, dyCoordNormPerSecond, dDistancePerSecond, dAnglePerSecond

add_goalie_biometrics *Add goalie biometrics to (a) play-by-play(s)*

Description

add_goalie_biometrics() adds goalie biometrics (height, weight, hand, and age at game date) to (a) play-by-play(s) for shot attempts. Goalie identity is resolved from goalieInNetId first and goaliePlayerIdAgainst second. If neither is available on a row, the added goalie biometrics are left as NA.

Usage

```
add_goalie_biometrics(play_by_play)
```

Arguments

play_by_play data.frame of play-by-play(s) using the current public schema returned by [gc_play_by_play\(\)](#), [gc_play_by_plays\(\)](#), [wsc_play_by_play\(\)](#), or [wsc_play_by_plays\(\)](#)

Value

data.frame with one row per event (play) and added columns: goalieHeight, goalieWeight, goalieHandCode, and goalieAge

add_shift_times *Add on-ice shift times to a play-by-play*

Description

add_shift_times() adds SecondsRemainingInShift, SecondsElapsedInShift, and SecondsElapsedInPeriodSinceLA columns for the on-ice goalies and skaters already present in a public play-by-play. It accepts either a single game play-by-play plus `shift_chart()` data or a season aggregate plus `shift_charts()` data.

Usage

```
add_shift_times(play_by_play, shift_chart)
```

Arguments

play_by_play data.frame of play-by-play(s) using the current public schema returned by `gc_play_by_play()`, `gc_play_by_plays()`, `wsc_play_by_play()`, or `wsc_play_by_plays()`

shift_chart data.frame returned by `shift_chart()` or `shift_charts()`

Value

data.frame with one row per event (play) and added or updated scalar on-ice shift timing columns

Examples

```
pbp <- gc_play_by_play(game = 2023030417)
sc <- shift_chart(game = 2023030417)
pbp <- add_shift_times(pbp, sc)
```

add_shooter_biometrics *Add shooter biometrics to (a) play-by-play(s)*

Description

add_shooter_biometrics() adds shooter biometrics (height, weight, hand, age at game date, and position) to (a) play-by-play(s) for shot attempts.

Usage

```
add_shooter_biometrics(play_by_play)
```

Arguments

`play_by_play` data.frame of play-by-play(s) using the current public schema returned by `gc_play_by_play()`, `gc_play_by_plays()`, `wsc_play_by_play()`, or `wsc_play_by_plays()`

Value

data.frame with one row per event (play) and added columns: `shooterHeight`, `shooterWeight`, `shooterHandCode`, `shooterAge`, and `shooterPositionCode`

attendance	<i>Access the attendance by season and game type</i>
------------	--

Description

`attendance()` retrieves the attendance by season and game type as a `data.frame` where each row represents season and includes detail on date/season filtering windows and chronological context.

Usage

```
attendance()
```

Value

data.frame with one row per season

Examples

```
all_attendance <- attendance()
```

award_winners	<i>Access all the award winners/finalists</i>
---------------	---

Description

`award_winners()` retrieves all the award winners/finalists as a `data.frame` where each row represents winner/finalist and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and player identity, role, handedness, and biographical profile.

Usage

```
award_winners()
```

Value

data.frame with one row per winner/finalist

Examples

```
all_award_winners <- award_winners()
```

awards	<i>Access all the awards</i>
--------	------------------------------

Description

`awards()` retrieves all the awards as a `data.frame` where each row represents award and includes detail on recognition, leaderboard, and milestone-watch context.

Usage

```
awards()
```

Value

`data.frame` with one row per award

Examples

```
all_awards <- awards()
```

boxscore	<i>Access the boxscore for a game, team, and position</i>
----------	---

Description

`boxscore()` retrieves the boxscore for a game, team, and position as a `data.frame` where each row represents player and includes detail on player identity, role, handedness, and biographical profile plus production, workload, efficiency, and result-level performance outcomes.

Usage

```
boxscore(game = 2023030417, team = "home", position = "forwards")
```

Arguments

<code>game</code>	integer ID (e.g., 2025020275); see <code>games()</code> for reference
<code>team</code>	character of 'h'/'home' or 'a'/'away'
<code>position</code>	character of 'f'/'forwards', 'd'/'defensemen', or 'g'/'goalies'

Value

`data.frame` with one row per player

Examples

```
boxscore_COL_forwards_Martin_Necas_legacy_game <- boxscore(  
  game = 2025020275,  
  team = 'H',  
  position = 'F'  
)
```

bracket	<i>Access the playoff bracket for a season</i>
---------	--

Description

`bracket()` retrieves the playoff bracket for a season as a `data.frame` where each row represents series and includes detail on team identity, affiliation, and matchup-side context.

Usage

```
bracket(season = season_now())
```

Arguments

`season` integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

`data.frame` with one row per series

Examples

```
bracket_20242025 <- bracket(season = 20242025)
```

calculate_expected_goals	<i>Calculate the expected goals for all the shots in (a) play-by-plays</i>
--------------------------	--

Description

`calculate_expected_goals()` scores shot events with `nhlscraper`'s built-in ridge expected-goals model. The runtime model is a fixed six-partition system: `sd` (5v5), `ev` (other even strength), `pp` (power play), `sh` (short-handed), `en` (empty net against), and `ps` (penalty shot; trained on penalty-shot and shootout-style rows). The `legacy_model` argument is accepted for backward compatibility but ignored.

Usage

```
calculate_expected_goals(play_by_play, model = NULL)
```

```
calculate_xG(play_by_play, model = NULL)
```

Arguments

`play_by_play` data.frame of play-by-play(s) using the current public schema returned by `gc_play_by_play()`, `gc_play_by_plays()`, `wsc_play_by_play()`, or `wsc_play_by_plays()`. Legacy alias-only columns such as `typeDescKey`, `period`, `SOGFor`, `SOGAgainst`, and `SOGDifferential` are no longer backfilled by the xG scorer.

`model` deprecated legacy model selector; ignored

Value

data.frame with one row per event (play) and added xG column

Examples

```
# May take >5s, so skip.

pbp <- gc_play_by_play()
pbp_with_xg <- calculate_expected_goals(play_by_play = pbp)
```

```
coach_career_statistics
```

Access the career statistics for all the coaches

Description

`coach_career_statistics()` retrieves the career statistics for all the coaches as a data.frame where each row represents coach and includes detail on ranking movement, points pace, and division/conference position signals.

Usage

```
coach_career_statistics()
```

```
coach_career_stats()
```

Value

data.frame with one row per coach

Examples

```
coach_career_stats <- coach_career_statistics()
```

`coach_franchise_statistics`*Access the statistics for all the coaches by franchise and game type*

Description

`coach_franchise_statistics()` retrieves the statistics for all the coaches by franchise and game type as a `data.frame` where each row represents franchise per coach per game type and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and player identity, role, handedness, and biographical profile.

Usage

```
coach_franchise_statistics()
```

```
coach_franchise_stats()
```

Value

`data.frame` with one row per franchise per coach per game type

Examples

```
coach_franchise_stats <- coach_franchise_statistics()
```

`coaches`*Access all the coaches*

Description

`coaches()` retrieves all the coaches as a `data.frame` where each row represents coach and includes detail on team identity, affiliation, and matchup-side context plus player identity, role, handedness, and biographical profile.

Usage

```
coaches()
```

Value

`data.frame` with one row per coach

Examples

```
all_coaches <- coaches()
```

combine_reports	<i>Access the draft combine reports</i>
-----------------	---

Description

combine_reports() retrieves the draft combine reports as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile.

Usage

```
combine_reports()
```

Value

data.frame with one row per player

Examples

```
combine_reports <- combine_reports()
```

contracts	<i>Access all contracts from packaged internal data</i>
-----------	---

Description

contracts() loads preprocessed contract records bundled with the package and returns a cleaned data.frame with package-consistent column names, season IDs, numeric money fields, and team/player identifiers.

Usage

```
contracts()
```

Value

data.frame with one row per contract

Examples

```
all_contracts <- contracts()
```

countries	<i>Access all the countries</i>
-----------	---------------------------------

Description

`countries()` retrieves all the countries as a `data.frame` where each row represents country and includes detail on reference metadata, regional context, and media availability detail.

Usage

```
countries()
```

Value

`data.frame` with one row per country

Examples

```
all_countries <- countries()
```

draft_picks	<i>Access all the draft picks</i>
-------------	-----------------------------------

Description

`draft_picks()` retrieves all the draft picks as a `data.frame` where each row represents pick and includes detail on team identity, affiliation, and matchup-side context plus player identity, role, handedness, and biographical profile.

Usage

```
draft_picks()
```

Value

`data.frame` with one row per pick

Examples

```
# May take >5s, so skip.  
all_draft_picks <- draft_picks()
```

draft_prospects	<i>Access all the draft prospects</i>
-----------------	---------------------------------------

Description

draft_prospects() retrieves all the draft prospects as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile plus broadcast carriage, media availability, and viewing-link metadata.

Usage

```
draft_prospects()
```

Value

data.frame with one row per player

Examples

```
# May take >5s, so skip.
all_prospects <- draft_prospects()
```

draft_rankings	<i>Access the draft rankings for a class and category</i>
----------------	---

Description

draft_rankings() retrieves the draft rankings for a class and category as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile plus draft-board context, scouting background, and pick/round progression.

Usage

```
draft_rankings(class = season_now()%%10000, category = 1)
```

Arguments

class	integer in YYYY (e.g., 2017); see drafts() for reference
category	integer in 1:4 (where 1 = North American Skaters, 2 = International Skaters, 3 = North American Goalies, and 4 = International Goalies) OR character of 'NAS'/'NA Skaters'/'North American Skaters', 'INTLS'/'INTL Skaters'/'International Skaters', 'NAG'/'NA Goalies'/'North American Goalies', 'INTLG'/'INTL Goalies'/'International Goalies'

Value

data.frame with one row per player

Examples

```
draft_rankings_INTL_Skaters_2017 <- draft_rankings(  
  class = 2017,  
  category = 2  
)
```

draft_tracker	<i>Access the real-time draft tracker</i>
---------------	---

Description

draft_tracker() retrieves the real-time draft tracker as a data.frame where each row represents player and includes detail on team identity, affiliation, and matchup-side context, player identity, role, handedness, and biographical profile, and venue/location geography and regional metadata.

Usage

```
draft_tracker()
```

Value

data.frame with one row per player

Examples

```
draft_tracker <- draft_tracker()
```

drafts	<i>Access all the drafts</i>
--------	------------------------------

Description

drafts() retrieves all the drafts as a data.frame where each row represents draft and includes detail on venue/location geography and regional metadata.

Usage

```
drafts()
```

Value

data.frame with one row per draft

Examples

```
all_drafts <- drafts()
```

draw_NHL_rink	<i>Draw a full NHL rink</i>
---------------	-----------------------------

Description

draw_NHL_rink() draws a full NHL rink such that the x and y coordinates span -100 to 100 and -43 to +43, respectively. Use `graphics::points()` to create custom graphs; check out an example on the online documentation!

Usage

```
draw_NHL_rink()
```

Value

NULL

Examples

```
draw_NHL_rink()
```

espn_futures	<i>Access the ESPN futures for a season</i>
--------------	---

Description

espn_futures() retrieves the ESPN futures for a season as a data.frame where each row represents type and includes detail on betting market snapshots with side/total prices and provider variation.

Usage

```
espn_futures(season = season_now())
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see `seasons()` for reference

Value

nested data.frame with one row per type (outer) and book (inner)

Examples

```
ESPN_futures_20252026 <- espn_futures(20252026)
```

espn_game_odds	<i>Access the ESPN odds for a game</i>
----------------	--

Description

`espn_game_odds()` retrieves the ESPN odds for a game as a `data.frame` where each row represents provider and includes detail on team identity, affiliation, and matchup-side context plus betting market snapshots with side/total prices and provider variation.

Usage

```
espn_game_odds(game = 401777460)
```

Arguments

`game` integer ID (e.g., 401777460); see `espn_games()` for reference

Value

`data.frame` with one row per provider

Examples

```
ESPN_odds_SCF_20242025 <- espn_game_odds(game = 401777460)
```

espn_game_summary	<i>Access the ESPN summary for a game</i>
-------------------	---

Description

`espn_game_summary()` retrieves the ESPN summary for a game as a nested list that separates summary and detail blocks for date/season filtering windows and chronological context, venue/location geography and regional metadata, and playoff-series progression, round status, and series leverage.

Usage

```
espn_game_summary(game = 401777460)
```

Arguments

`game` integer ID (e.g., 401777460); see `espn_games()` for reference

Value

list with various items

Examples

```
ESPN_summary_SCF_20242025 <- espn_game_summary(game = 401777460)
```

espn_games	<i>Access the ESPN games for a season</i>
------------	---

Description

espn_games() retrieves the ESPN games for a season as a data.frame where each row represents ESPN and includes detail on game timing, matchup state, scoring flow, and situational event detail.

Usage

```
espn_games(season = season_now())
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per ESPN game

Examples

```
ESPN_games_20242025 <- espn_games(season = 20242025)
```

espn_injuries	<i>Access the real-time ESPN injury reports</i>
---------------	---

Description

espn_injuries() retrieves the real-time ESPN injury reports as a data.frame where each row represents team and includes detail on availability status tracking for injuries or transactions.

Usage

```
espn_injuries()
```

Value

nested data.frame with one row per team (outer) and player (inner)

Examples

```
ESPN_injuries_now <- espn_injuries()
```

espn_play_by_play *Access the ESPN play-by-play for a game*

Description

espn_play_by_play() retrieves the ESPN play-by-play for a game as a `data.frame` where each row represents event and includes detail on game timeline state, period/clock progression, and matchup flow, team identity, affiliation, and matchup-side context, and situational splits across home/road, strength state, and overtime/shootout states.

Usage

```
espn_play_by_play(game = 401777460)
```

```
espn_pbp(game = 401777460)
```

Arguments

game integer ID (e.g., 401777460); see [espn_games\(\)](#) for reference

Value

`data.frame` with one row per event (play)

Examples

```
ESPN_pbp_SCF_20242025 <- espn_play_by_play(game = 401777460)
```

espn_player_summary *Access the ESPN summary for a player*

Description

espn_player_summary() retrieves the ESPN summary for a player as a `data.frame` where each row represents one result and includes detail on game timing, matchup state, scoring flow, and situational event detail.

Usage

```
espn_player_summary(player = 3988803)
```

Arguments

player integer ID (e.g., 3988803); see [espn_players\(\)](#) for reference

Value

data.frame with one row

Examples

```
ESPN_summary_Charlie_McAvoy <- espn_player_summary(player = 3988803)
```

espn_players	<i>Access all the ESPN players</i>
--------------	------------------------------------

Description

espn_players() retrieves all the ESPN players as a data.frame where each row represents ESPN player and includes detail on person-level profile context and performance history with situational splits.

Usage

```
espn_players()
```

Value

data.frame with one row per ESPN player

Examples

```
all_ESPN_players <- espn_players()
```

espn_team_summary	<i>Access the ESPN summary for a team</i>
-------------------	---

Description

espn_team_summary() retrieves the ESPN summary for a team as a data.frame where each row represents one result and includes detail on game timing, matchup state, scoring flow, and situational event detail.

Usage

```
espn_team_summary(team = 1)
```

Arguments

team integer ID (e.g., 1); see [espn_teams\(\)](#) for reference

Value

data.frame with one row

Examples

```
ESPN_summary_Boston_Bruins <- espn_team_summary(team = 1)
```

espn_teams	<i>Access all the ESPN teams</i>
------------	----------------------------------

Description

espn_teams() retrieves all the ESPN teams as a data.frame where each row represents ESPN team and includes detail on team composition, matchup context, and season progression detail.

Usage

```
espn_teams()
```

Value

data.frame with one row per ESPN team

Examples

```
all_ESPN_teams <- espn_teams()
```

espn_transactions	<i>Access the ESPN transactions for a season</i>
-------------------	--

Description

espn_transactions() retrieves the ESPN transactions for a season as a data.frame where each row represents transaction and includes detail on availability and transaction-status tracking detail.

Usage

```
espn_transactions(season = season_now())
```

Arguments

season	integer in YYYYYYYYY (e.g., 20242025); the summer of the latter year is included
--------	--

Value

data.frame with one row per transaction

Examples

```
ESPN_transactions_20242025 <- espn_transactions(season = 20242025)
```

```
expansion_draft_picks
```

Access all the expansion draft picks

Description

`expansion_draft_picks()` retrieves all the expansion draft picks as a `data.frame` where each row represents pick and includes detail on date/season filtering windows and chronological context plus team identity, affiliation, and matchup-side context.

Usage

```
expansion_draft_picks()
```

Value

`data.frame` with one row per pick

Examples

```
all_expansion_draft_picks <- expansion_draft_picks()
```

```
expansion_drafts
```

Access all the expansion drafts

Description

`expansion_drafts()` retrieves all the expansion drafts as a `data.frame` where each row represents expansion draft and includes detail on date/season filtering windows and chronological context plus reference definitions and rules-framework information.

Usage

```
expansion_drafts()
```

Value

`data.frame` with one row per expansion draft

Examples

```
all_expansion_drafts <- expansion_drafts()
```

```
franchise_playoff_situational_results
```

Access the playoff series results for all the franchises by situation

Description

`franchise_playoff_situational_results()` retrieves the playoff series results for all the franchises by situation as a `data.frame` where each row represents franchise per situation and includes detail on team identity, affiliation, and matchup-side context.

Usage

```
franchise_playoff_situational_results()
```

Value

`data.frame` with one row per franchise per situation

Examples

```
franchise_playoff_situational_results <-  
  franchise_playoff_situational_results()
```

```
franchise_season_statistics
```

Access the statistics for all the franchises by season and game type

Description

`franchise_season_statistics()` retrieves the statistics for all the franchises by season and game type as a `data.frame` where each row represents franchise per season per game type and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
franchise_season_statistics()
```

```
franchise_season_stats()
```

Value

`data.frame` with one row per franchise per season per game type

Examples

```
# May take >5s, so skip.  
franchise_season_stats <- franchise_season_statistics()
```

`franchise_statistics` *Access the all-time statistics for all the franchises by game type*

Description

`franchise_statistics()` retrieves the all-time statistics for all the franchises by game type as a `data.frame` where each row represents franchise per game type and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
franchise_statistics()
```

```
franchise_stats()
```

Value

`data.frame` with one row per franchise per game type

Examples

```
franchise_stats <- franchise_statistics()
```

`franchise_team_statistics`
Access the all-time statistics for all the franchises by team and game type

Description

`franchise_team_statistics()` retrieves the all-time statistics for all the franchises by team and game type as a `data.frame` where each row represents team per franchise per game type and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
franchise_team_statistics()
```

```
franchise_team_stats()
```

Value

`data.frame` with one row per team per franchise per game type

Examples

```
franchise_team_stats <- franchise_team_statistics()
```

```
franchise_versus_franchise
```

Access the all-time statistics versus other franchises for all the franchises by game type

Description

`franchise_versus_franchise()` retrieves the all-time statistics versus other franchises for all the franchises by game type as a `data.frame` where each row represents franchise per franchise per game type and includes detail on date/season filtering windows and chronological context plus team identity, affiliation, and matchup-side context.

Usage

```
franchise_versus_franchise()
```

```
franchise_vs_franchise()
```

Value

`data.frame` with one row per franchise per franchise per game type

Examples

```
# May take >5s, so skip.  
franchise_vs_franchise <- franchise_versus_franchise()
```

```
franchises
```

Access all the franchises

Description

`franchises()` retrieves all the franchises as a `data.frame` where each row represents franchise and includes detail on team identity, affiliation, and matchup-side context.

Usage

```
franchises()
```

Value

`data.frame` with one row per franchise

Examples

```
all_franchises <- franchises()
```

game_odds	<i>Access the real-time game odds for a country by partnered bookmaker</i>
-----------	--

Description

game_odds() retrieves the real-time game odds for a country by partnered bookmaker as a data.frame where each row represents game and includes detail on betting market lines, prices, and provider-level context.

Usage

```
game_odds(country = "US")
```

Arguments

country two-letter code (e.g., 'CA'); see [countries\(\)](#) for reference

Value

data.frame with one row per game

Examples

```
game_odds_CA <- game_odds(country = 'CA')
```

game_rovers	<i>Access the rosters for a game</i>
-------------	--------------------------------------

Description

game_rovers() retrieves the rosters for a game as a data.frame where each row represents player and includes detail on team identity, affiliation, and matchup-side context plus player identity, role, handedness, and biographical profile.

Usage

```
game_rovers(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per player

Examples

```
rosters_Martin_Necas_legacy_game <- game_rosters(game = 2025020275)
```

game_type_now	<i>Access the game type as of now</i>
---------------	---------------------------------------

Description

game_type_now() retrieves the game type as of now and returns a scalar integer used as the current-context default in season/game-type dependent wrappers.

Usage

```
game_type_now()
```

Value

integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season)

Examples

```
game_type_now <- game_type_now()
```

games	<i>Access all the games</i>
-------	-----------------------------

Description

games() retrieves all the games as a data.frame where each row represents game and includes detail on game timeline state, period/clock progression, and matchup flow plus date/season filtering windows and chronological context.

Usage

```
games()
```

Value

data.frame with one row per game

Examples

```
# May take >5s, so skip.
all_games <- games()
```

gc_play_by_play *Access the GameCenter (GC) play-by-play for a game*

Description

gc_play_by_play() retrieves the GameCenter (GC) play-by-play for a game as a data.frame where each row represents an event. The returned schema is the cleaned, public-facing play-by-play schema, including canonical names such as periodNumber, eventTypeCode, eventTypeDescKey, homeShots, shotsFor, penaltyTypeDescKey, penaltyDuration, servedByPlayerId, goalieInNetId, and HTML-report-derived on-ice player ID columns such as homeGoaliePlayerId, awayGoaliePlayerId, homeSkater1PlayerId, and any additional overflow skater slots required by the game. HTML report skater and goalie IDs are returned whenever they can be matched back to a supported row, even when the raw situationCode is stale. Use add_shift_times() with shift_chart() (or shift_charts()) to add on-ice shift timing columns.

Usage

```
gc_play_by_play(game = 2023030417)
```

```
gc_pbp(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see games() for reference

Value

data.frame with one row per event (play)

Examples

```
# May take >5s, so skip.
gc_pbp_Martin_Necas_legacy_game <- gc_play_by_play(game = 2025020275)
```

gc_play_by_play_raw *Access the raw GameCenter (GC) play-by-play for a game*

Description

gc_play_by_play_raw() returns the raw flattened GameCenter play-by-play as served by the NHL API for one game. Use gc_play_by_play() for the cleaned public schema that repairs common clock/order defects and appends the derived public columns.

Usage

```
gc_play_by_play_raw(game = 2023030417)
```

```
gc_pbp_raw(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per event (play)

Examples

```
gc_raw_Martin_Necas_legacy_game <- gc_play_by_play_raw(game = 2025020275)
```

gc_play_by_plays *Access the GameCenter (GC) play-by-plays for a season*

Description

`gc_play_by_plays()` loads the GC play-by-plays for a given season.

Usage

```
gc_play_by_plays(season = 20242025)
```

```
gc_pbps(season = 20242025)
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per event (play) per game

Examples

```
# May take >5s, so skip.  
gc_pbps_20212022 <- gc_play_by_plays(season = 20212022)
```

`gc_play_by_plays_raw` *Access the raw GameCenter (GC) play-by-plays for a season*

Description

`gc_play_by_plays_raw()` loads the raw GC play-by-plays for a given season.

Usage

```
gc_play_by_plays_raw(season = 20242025)
```

```
gc_pbps_raw(season = 20242025)
```

Arguments

`season` integer in YYYYYYYYY (e.g., 20242025); see `seasons()` for reference

Value

data.frame with one row per raw event (play) per game

Examples

```
# May take >5s, so skip.  
gc_pbps_raw_20212022 <- gc_play_by_plays_raw(season = 20212022)
```

`gc_summary` *Access the GameCenter (GC) summary for a game*

Description

`gc_summary()` retrieves the GameCenter (GC) summary for a game as a nested list that separates summary and detail blocks for game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and venue/location geography and regional metadata.

Usage

```
gc_summary(game = 2023030417)
```

Arguments

`game` integer ID (e.g., 2025020275); see `games()` for reference

Value

list of various items

Examples

```
gc_summary_Martin_Necas_legacy_game <- gc_summary(game = 2025020275)
```

general_managers	<i>Access all the general managers</i>
------------------	--

Description

general_managers() retrieves all the general managers as a data.frame where each row represents general manager and includes detail on player identity, role, handedness, and biographical profile.

Usage

```
general_managers()
```

```
gms()
```

Value

data.frame with one row per general manager

Examples

```
all_GMs <- general_managers()
```

get_attendance	<i>Access the attendance by season and game type</i>
----------------	--

Description

get_attendance() is deprecated. Use [attendance\(\)](#) instead.

Usage

```
get_attendance()
```

Value

data.frame with one row per season

get_award_winners	<i>Access all the award winners/finalists</i>
-------------------	---

Description

get_award_winners() is deprecated. Use [award_winners\(\)](#) instead.

Usage

```
get_award_winners()
```

Value

data.frame with one row per winner/finalist

get_awards	<i>Access all the awards</i>
------------	------------------------------

Description

get_awards() is deprecated. Use [awards\(\)](#) instead.

Usage

```
get_awards()
```

Value

data.frame with one row per award

get_bracket	<i>Access the playoff bracket for a season</i>
-------------	--

Description

get_bracket() is deprecated. Use [bracket\(\)](#) instead.

Usage

```
get_bracket(season = season_now())
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per series

get_configuration	<i>Access the configurations for team, skater, and goalie reports</i>
-------------------	---

Description

get_configuration() is defunct. Use [team_report_configurations\(\)](#), [skater_report_configurations\(\)](#), and/or [goalie_report_configurations\(\)](#) instead.

Usage

```
get_configuration()
```

get_countries	<i>Access all the countries</i>
---------------	---------------------------------

Description

get_countries() is deprecated. Use [countries\(\)](#) instead.

Usage

```
get_countries()
```

Value

data.frame with one row per country

get_draft_picks	<i>Access all the draft picks</i>
-----------------	-----------------------------------

Description

get_draft_picks() is deprecated. Use [draft_picks\(\)](#) instead.

Usage

```
get_draft_picks()
```

Value

data.frame with one row per pick

get_draft_rankings *Access the draft rankings for a year and player type*

Description

get_draft_rankings() is deprecated. Use [draft_rankings\(\)](#) instead.

Usage

```
get_draft_rankings(year = season_now()%%10000, player_type = 1)
```

Arguments

year integer in YYYY (e.g., 2017); see [drafts\(\)](#) for reference

player_type integer in 1:4 (where 1 = North American Skaters, 2 = International Skaters, 3 = North American Goalies, and 4 = International Goalies)

Value

data.frame with one row per player

get_draft_tracker *Access the real-time draft tracker*

Description

get_draft_tracker() is deprecated. Use [draft_tracker\(\)](#) instead.

Usage

```
get_draft_tracker()
```

Value

data.frame with one row per player

get_drafts	<i>Access all the drafts</i>
------------	------------------------------

Description

get_drafts() is deprecated. Use [drafts\(\)](#) instead.

Usage

```
get_drafts()
```

Value

data.frame with one row per draft

get_espn_athlete	<i>Access the ESPN summary for an athlete (player) and season</i>
------------------	---

Description

get_espn_athlete() is defunct. Use [espn_player_summary\(\)](#) instead.

Usage

```
get_espn_athlete()
```

get_espn_athletes	<i>Access all the ESPN athletes (players)</i>
-------------------	---

Description

get_espn_athletes() is deprecated. Use [espn_players\(\)](#) instead.

Usage

```
get_espn_athletes()
```

Value

data.frame with one row per ESPN athlete (player)

get_espn_coach	<i>Access the ESPN statistics for a coach and (multiple) season(s)</i>
----------------	--

Description

get_espn_coach() is defunct. Use [coach_career_statistics\(\)](#) instead.

Usage

```
get_espn_coach()
```

get_espn_coach_career	<i>Access the career ESPN statistics for a coach</i>
-----------------------	--

Description

get_espn_coach_career() is defunct. Use [coach_career_statistics\(\)](#) instead.

Usage

```
get_espn_coach_career()
```

get_espn_coaches	<i>Access the ESPN coaches for a season</i>
------------------	---

Description

get_espn_coaches() is defunct. Use [coaches\(\)](#) instead.

Usage

```
get_espn_coaches()
```

get_espn_event	<i>Access the ESPN summary for an event (game)</i>
----------------	--

Description

get_espn_event() is deprecated. Use [espn_game_summary\(\)](#) instead.

Usage

```
get_espn_event(event = 401777460)
```

Arguments

event integer ID (e.g., 401777460); see [espn_games\(\)](#) for reference

Value

data.frame with one row per event (game)

get_espn_event_odds	<i>Access the ESPN odds for an event (game)</i>
---------------------	---

Description

get_espn_event_odds() is deprecated. Use [espn_game_odds\(\)](#) instead.

Usage

```
get_espn_event_odds(event = 401777460)
```

Arguments

event integer ID (e.g., 401777460); see [espn_games\(\)](#) for reference

Value

data.frame with one row per provider

get_espn_event_officials

Access the officials for an ESPN event (game)

Description

get_espn_event_officials() is defunct. Use [gc_summary\(\)](#) and/or [wsc_summary\(\)](#) instead.

Usage

```
get_espn_event_officials()
```

get_espn_event_play_by_play

Access the ESPN play-by-play for an event (game)

Description

get_espn_event_play_by_play() is deprecated. Use [espn_play_by_play\(\)](#) instead.

Usage

```
get_espn_event_play_by_play(event = 401777460)
```

Arguments

event integer ID (e.g., 401777460); see [espn_games\(\)](#) for reference

Value

data.frame with one row per play

get_espn_event_stars *Access the three stars for an ESPN event (game)*

Description

get_espn_event_stars() is defunct. Use [gc_summary\(\)](#) and/or [wsc_summary\(\)](#) instead.

Usage

```
get_espn_event_stars()
```

get_espn_events	<i>Access the ESPN events (games) by start and end dates</i>
-----------------	--

Description

get_espn_events() is defunct. Use [espn_games\(\)](#) instead.

Usage

```
get_espn_events()
```

get_espn_futures	<i>Access the ESPN futures for a season</i>
------------------	---

Description

get_espn_futures() is defunct. Use [espn_futures\(\)](#) instead.

Usage

```
get_espn_futures()
```

get_espn_injuries	<i>Access the real-time ESPN injury reports</i>
-------------------	---

Description

get_espn_injuries() is deprecated. Use [espn_injuries\(\)](#) instead.

Usage

```
get_espn_injuries()
```

Value

nested data.frame with one row per team (outer) and player (inner)

get_espn_team *Access the ESPN summary for a team and season*

Description

get_espn_team() is defunct. Use [espn_team_summary\(\)](#) instead.

Usage

```
get_espn_team()
```

get_espn_teams *Access all the ESPN teams for a season*

Description

get_espn_teams() is defunct. Use [espn_teams\(\)](#) instead.

Usage

```
get_espn_teams()
```

get_espn_transactions *Access the ESPN transactions by start and end dates*

Description

get_espn_transactions() is defunct. Use [espn_transactions\(\)](#) instead.

Usage

```
get_espn_transactions()
```

get_franchise_season_by_season

Access the statistics for all the franchises by season and game type

Description

get_franchise_season_by_season() is deprecated. Use [franchise_season_statistics\(\)](#) instead.

Usage

```
get_franchise_season_by_season()
```

Value

data.frame with one row per franchise per season per game type

get_franchise_team_totals

Access the all-time statistics for all the franchises by team and game type

Description

get_franchise_team_totals() is deprecated. Use [franchise_team_statistics\(\)](#) instead.

Usage

```
get_franchise_team_totals()
```

Value

data.frame with one row per team per franchise per game type

get_franchise_vs_franchise

Access the all-time statistics versus other franchises for all the franchises by game type

Description

get_franchise_vs_franchise() is deprecated. Use [franchise_versus_franchise\(\)](#) instead.

Usage

```
get_franchise_vs_franchise()
```

Value

data.frame with one row per franchise per franchise per game type

get_franchises

Access all the franchises

Description

get_franchises() is deprecated. Use [franchises\(\)](#) instead.

Usage

```
get_franchises()
```

Value

data.frame with one row per franchise

get_game_boxscore *Access the boxscore for a game, team, and player type*

Description

get_game_boxscore() is deprecated. Use [boxscore\(\)](#) instead.

Usage

```
get_game_boxscore(game = 2023030417, team = "home", player_type = "forwards")
```

Arguments

game	integer ID (e.g., 2025020275); see games() for reference
team	character of 'home' or 'away'
player_type	character of 'forwards', 'defense', or 'goalies'

Value

data.frame with one row per player

get_game_landing *Access the GameCenter (GC) summary for a game*

Description

get_game_landing() is deprecated. Use [gc_summary\(\)](#) instead.

Usage

```
get_game_landing(game = 2023030417)
```

Arguments

game	integer ID (e.g., 2025020275); see games() for reference
------	--

Value

list of various items

get_game_story	<i>Access the World Showcase (WSC) summary for a game</i>
----------------	---

Description

get_game_story() is deprecated. Use [wsc_summary\(\)](#) instead.

Usage

```
get_game_story(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

list of various items

get_games	<i>Access all the games</i>
-----------	-----------------------------

Description

get_games() is deprecated. Use [games\(\)](#) instead.

Usage

```
get_games()
```

Value

data.frame with one row per game

get_gc_play_by_play *Access the GameCenter (GC) play-by-play for a game*

Description

get_gc_play_by_play() is deprecated. Use [gc_play_by_play\(\)](#) instead.

Usage

```
get_gc_play_by_play(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per event (play)

get_glossary *Access the glossary*

Description

get_glossary() is deprecated. Use [glossary\(\)](#) instead.

Usage

```
get_glossary()
```

Value

data.frame with one row per terminology

get_goalie_leaders	<i>Access the goalie statistics leaders for a season, game type, and category</i>
--------------------	---

Description

get_goalie_leaders() is deprecated. Use [goalie_leaders\(\)](#) instead.

Usage

```
get_goalie_leaders(season = "current", game_type = "", category = "wins")
```

Arguments

season	integer in YYYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
category	character of 'wins', 'shutouts', 'savePctg', or 'goalsAgainstAverage'

Value

data.frame with one row per player

get_goalie_milestones	<i>Access the goalies on milestone watch</i>
-----------------------	--

Description

get_goalie_milestones() is deprecated. Use [goalie_milestones\(\)](#) instead.

Usage

```
get_goalie_milestones()
```

Value

data.frame with one row per player

get_goalie_statistics *Access various reports for all the goalies by season or game*

Description

get_goalie_statistics() is defunct. Use [goalie_season_report\(\)](#) or [goalie_game_report\(\)](#) instead.

Usage

```
get_goalie_statistics()
```

get_goalies *Access all the goalies for a range of seasons*

Description

get_goalies() is defunct. Use [players\(\)](#) instead.

Usage

```
get_goalies()
```

get_officials *Access all the officials*

Description

get_officials() is deprecated. Use [officials\(\)](#) instead.

Usage

```
get_officials()
```

Value

data.frame with one row per official

get_partner_odds	<i>Access the real-time game odds for a country by partnered bookmaker</i>
------------------	--

Description

get_partner_odds() is deprecated. Use [game_odds\(\)](#) instead.

Usage

```
get_partner_odds(country = "US")
```

Arguments

country two-letter code (e.g., 'CA'); see [countries\(\)](#) for reference

Value

data.frame with one row per game

get_player_game_log	<i>Access the game log for a player, season, and game type</i>
---------------------	--

Description

get_player_game_log() is deprecated. Use [player_game_log\(\)](#) instead.

Usage

```
get_player_game_log(player = 8478402, season = "now", game_type = "")
```

Arguments

player integer ID (e.g., 8480039); see [players\(\)](#) for reference
season integer in YYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference
game_type integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season)
 OR character of 'pre', 'regular', or 'playoff'/'post'; see [seasons\(\)](#) for reference;
 most functions will NOT support pre-season

Value

data.frame with one row per game

get_player_landing *Access the summary for a player*

Description

`get_player_landing()` is deprecated. Use [player_summary\(\)](#) instead.

Usage

```
get_player_landing(player = 8478402)
```

Arguments

`player` integer ID (e.g., 8480039); see [players\(\)](#) for reference

Value

list with various items

get_players *Access all the players*

Description

`get_players()` is deprecated. Use [players\(\)](#) instead.

Usage

```
get_players()
```

Value

data.frame with one row per player

get_schedule *Access the schedule for a date*

Description

get_schedule() is deprecated. Use [schedule\(\)](#) instead.

Usage

```
get_schedule(date = "2025-01-01")
```

Arguments

date character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see [seasons\(\)](#) for reference

Value

data.frame with one row per game

get_scoreboards *Access the scoreboards for a date*

Description

get_scoreboards() is deprecated. Use [scores\(\)](#) instead.

Usage

```
get_scoreboards(date = "now")
```

Arguments

date character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see [seasons\(\)](#) for reference

Value

data.frame with one row per game

get_scores	<i>Access the scores for a date</i>
------------	-------------------------------------

Description

get_scores() is deprecated. Use [scores\(\)](#) instead.

Usage

```
get_scores(date = "now")
```

Arguments

date	character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see seasons() for reference
------	---

Value

data.frame with one row per game

get_season_now	<i>Access the season and game type as of now</i>
----------------	--

Description

get_season_now() is defunct. Use [season_now\(\)](#) and/or [game_type_now\(\)](#) instead.

Usage

```
get_season_now()
```

get_seasons	<i>Access all the seasons</i>
-------------	-------------------------------

Description

get_seasons() is deprecated. Use [seasons\(\)](#) instead.

Usage

```
get_seasons()
```

Value

data.frame with one row per season

get_series	<i>Access the playoff series for a season and round</i>
------------	---

Description

get_series() is defunct.

Usage

```
get_series()
```

get_series_schedule	<i>Access the playoff schedule for a season and series</i>
---------------------	--

Description

get_series_schedule() is deprecated. Use [series_schedule\(\)](#) instead.

Usage

```
get_series_schedule(season = season_now(), series = "a")
```

Arguments

season	integer in YYYYYYYYY (e.g., 20242025); see seasons() for reference
series	one-letter code (e.g., 'O'); see series() and/or bracket() for reference

Value

data.frame with one row per game

get_shift_charts	<i>Access the shift charts for a game</i>
------------------	---

Description

get_shift_charts() is deprecated. Use [shift_chart\(\)](#) instead.

Usage

```
get_shift_charts(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per shift

get_skater_leaders	<i>Access the skater statistics leaders for a season, game type, and category</i>
--------------------	---

Description

get_skater_leaders() is deprecated. Use [skater_leaders\(\)](#) instead.

Usage

```
get_skater_leaders(season = "current", game_type = "", category = "points")
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

game_type integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see [seasons\(\)](#) for reference; most functions will NOT support pre-season

category character of 'assists', 'goals', 'goalsSh', 'goalsPp', 'points', 'penaltyMins', 'toi', 'plusMinus', or 'faceoffLeaders'

Value

data.frame with one row per player

get_skater_milestones	<i>Access the skaters on milestone watch</i>
-----------------------	--

Description

get_skater_milestones() is deprecated. Use [skater_milestones\(\)](#) instead.

Usage

```
get_skater_milestones()
```

Value

data.frame with one row per player

get_skater_statistics *Access various reports for all the skaters by season or game*

Description

get_skater_statistics() is defunct. Use [skater_season_report\(\)](#) or [skater_game_report\(\)](#) instead.

Usage

```
get_skater_statistics()
```

get_skaters *Access all the skaters for a range of seasons*

Description

get_skaters() is defunct. Use [players\(\)](#) instead.

Usage

```
get_skaters()
```

get_spotlight_players *Access the spotlight players*

Description

get_spotlight_players() is deprecated. Use [spotlight_players\(\)](#) instead.

Usage

```
get_spotlight_players()
```

Value

data.frame with one row per player

get_standings	<i>Access the standings for a date</i>
---------------	--

Description

get_standings() is deprecated. Use [standings\(\)](#) instead.

Usage

```
get_standings(date = "2025-01-01")
```

Arguments

date	character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see seasons() for reference
------	---

Value

data.frame with one row per team

get_standings_information	<i>Access the standings rules by season</i>
---------------------------	---

Description

get_standings_information() is deprecated. Use [standings_rules\(\)](#) instead.

Usage

```
get_standings_information()
```

Value

data.frame with one row per season

get_streams	<i>Access all the streams</i>
-------------	-------------------------------

Description

get_streams() is deprecated. Use [streams\(\)](#) instead.

Usage

```
get_streams()
```

Value

data.frame with one row per stream

get_team_prospects	<i>Access the prospects for a team and position</i>
--------------------	---

Description

get_team_prospects() is deprecated. Use [team_prospects\(\)](#) instead.

Usage

```
get_team_prospects(team = "NJD", player_type = "forwards")
```

Arguments

team	three-letter code (e.g., 'COL'); see teams() for reference
player_type	character of 'forwards', 'defensemen', or 'goalies'

Value

data.frame with one row per player

get_team_roster	<i>Access the roster for a team, season, and player type</i>
-----------------	--

Description

get_team_roster() is deprecated. Use [roster\(\)](#) instead.

Usage

```
get_team_roster(team = "NJD", season = "current", player_type = "forwards")
```

Arguments

team	three-letter code (e.g., 'COL'); see teams() for reference
season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
player_type	character of 'forwards', 'defensemen', or 'goalies'

Value

data.frame with one row per player

get_team_roster_statistics	<i>Access the roster statistics for a team, season, game type, and player type</i>
----------------------------	--

Description

get_team_roster_statistics() is deprecated. Use [roster_statistics\(\)](#) instead.

Usage

```
get_team_roster_statistics(
  team = "NJD",
  season = "now",
  game_type = 2,
  player_type = "skaters"
)
```

Arguments

team	three-letter code (e.g., 'COL'); see teams() for reference
season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
player_type	character of 'skaters' or 'goalies'

Value

data.frame with one row per player

get_team_schedule *Access the schedule for a team and season*

Description

get_team_schedule() is deprecated. Use [team_season_schedule\(\)](#) instead.

Usage

```
get_team_schedule(team = "NJD", season = "now")
```

Arguments

team three-letter code (e.g., 'COL'); see [teams\(\)](#) for reference
season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per game

get_team_scoreboard *Access the team scoreboard as of now*

Description

get_team_scoreboard() is defunct.

Usage

```
get_team_scoreboard()
```

get_team_seasons *Access the season(s) and game type(s) in which a team played*

Description

get_team_seasons() is deprecated. Use [team_seasons\(\)](#) instead.

Usage

```
get_team_seasons(team = "NJD")
```

Arguments

team three-letter code (e.g., 'COL'); see [teams\(\)](#) for reference

Value

data.frame with one row per season

get_team_statistics *Access various reports for all the teams by season or game*

Description

get_team_statistics() is defunct. Use [team_season_report\(\)](#) and/or [team_game_report\(\)](#) instead.

Usage

```
get_team_statistics()
```

get_teams *Access all the teams*

Description

get_teams() is deprecated. Use [teams\(\)](#) instead.

Usage

```
get_teams()
```

Value

data.frame with one row per team

get_tv_schedule	<i>Access the NHL Network TV schedule for a date</i>
-----------------	--

Description

get_tv_schedule() is deprecated. Use [tv_schedule\(\)](#) instead.

Usage

```
get_tv_schedule(date = "now")
```

Arguments

date	character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see seasons() for reference
------	---

Value

data.frame with one row per program

get_venues	<i>Access all the venues</i>
------------	------------------------------

Description

get_venues() is deprecated. Use [venues\(\)](#) instead.

Usage

```
get_venues()
```

Value

data.frame with one row per venue

get_wsc_play_by_play *Access the World Showcase (WSC) play-by-play for a game*

Description

get_wsc_play_by_play() is deprecated. Use [wsc_play_by_play\(\)](#) instead.

Usage

```
get_wsc_play_by_play(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per event (play)

glossary *Access the glossary*

Description

glossary() retrieves the glossary as a data.frame where each row represents terminology and includes detail on reference definitions and rules-framework information.

Usage

```
glossary()
```

Value

data.frame with one row per terminology

Examples

```
glossary <- glossary()
```

```
goalie_edge_five_versus_five
```

Access the EDGE 5 vs. 5 statistics for a goalie, season, game type, and category

Description

`goalie_edge_five_versus_five()` retrieves the EDGE 5 vs. 5 statistics for a goalie, season, game type, and category as a nested list that separates summary and detail blocks for production, workload, efficiency, and result-level performance outcomes plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
goalie_edge_five_versus_five(
  player = 8476945,
  season = "now",
  game_type = "",
  category = "details"
)
```

```
goalie_edge_5_vs_5(
  player = 8476945,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

<code>player</code>	integer ID (e.g., 8478406)
<code>season</code>	integer in YYYYYYYYY (e.g., 20242025); see goalie_edge_seasons() for reference
<code>game_type</code>	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see goalie_edge_seasons() for reference; most functions will NOT support pre-season
<code>category</code>	character of 'd'/'details' or 'l'/'l10'/'last 10'

Value

list with four items (category = 'details') or `data.frame` with one row per game (category = 'last 10')

Examples

```
Mackenzie_Blackwood_L10_5_vs_5_regular_20242025 <- goalie_edge_five_versus_five(
  player = 8478406,
```

```
season    = 20242025,  
game_type = 2,  
category  = 'L'  
)
```

goalie_edge_leaders *Access the goalie EDGE statistics leaders for a season and game type*

Description

goalie_edge_leaders() retrieves the goalie EDGE statistics leaders for a season and game type as a nested list that separates summary and detail blocks for NHL EDGE style tracking outputs and relative-performance context.

Usage

```
goalie_edge_leaders(season = "now", game_type = "")
```

Arguments

season	integer in YYYYYYYYY (e.g., 20242025); see goalie_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/ 'post'; see goalie_edge_seasons() for reference; most functions will NOT support pre-season

Value

list of various items

Examples

```
goalie_EDGE_leaders_regular_20242025 <- goalie_edge_leaders(  
  season    = 20242025,  
  game_type = 2  
)
```

goalie_edge_save_percentage

Access the EDGE save percentage statistics for a goalie, season, game type, and category

Description

goalie_edge_save_percentage() retrieves the EDGE save percentage statistics for a goalie, season, game type, and category as a nested list that separates summary and detail blocks for NHL EDGE style tracking outputs and relative-performance context.

Usage

```
goalie_edge_save_percentage(
  player = 8476945,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

player	integer ID (e.g., 8478406)
season	integer in YYYYYYYY (e.g., 20242025); see goalie_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see goalie_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 'l'/'l10'/'last 10'

Value

list with two items (category = 'details') or data.frame with one row per game (category = 'last 10')

Examples

```
Mackenzie_Blackwood_L10_sP_regular_20242025 <-
  goalie_edge_save_percentage(
    player = 8478406,
    season = 20242025,
    game_type = 2,
    category = 'L'
  )
```

goalie_edge_seasons *Access the season(s) and game type(s) in which there exists goalie EDGE statistics*

Description

goalie_edge_seasons() retrieves the season(s) and game type(s) in which there exists goalie EDGE statistics as a data.frame where each row represents season and includes detail on date/season filtering windows and chronological context plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
goalie_edge_seasons()
```

Value

data.frame with one row per season

Examples

```
goalie_EDGE_seasons <- goalie_edge_seasons()
```

goalie_edge_shot_location *Access the EDGE shot location statistics for a goalie, season, game type, and category*

Description

goalie_edge_shot_location() retrieves the EDGE shot location statistics for a goalie, season, game type, and category as a data.frame where each row represents shot location and includes detail on production, workload, efficiency, and result-level performance outcomes plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
goalie_edge_shot_location(  
  player = 8476945,  
  season = "now",  
  game_type = "",  
  category = "details"  
)
```

Arguments

player	integer ID (e.g., 8478406)
season	integer in YYYYYYYYY (e.g., 20242025); see goalie_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see goalie_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/details' or 't'/totals'

Value

data.frame with one row per shot location

Examples

```
Mackenzie_Blackwood_shot_location_totals_regular_20242025 <-
  goalie_edge_shot_location(
    player = 8478406,
    season = 20242025,
    game_type = 2,
    category = 'T'
  )
```

`goalie_edge_summary` *Access the EDGE summary for a goalie, season, and game type*

Description

`goalie_edge_summary()` retrieves the EDGE summary for a goalie, season, and game type as a nested list that separates summary and detail blocks for player identity, role, handedness, and biographical profile plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
goalie_edge_summary(player = 8476945, season = "now", game_type = "")
```

Arguments

player	integer ID (e.g., 8478406)
season	integer in YYYYYYYYY (e.g., 20242025); see goalie_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see goalie_edge_seasons() for reference; most functions will NOT support pre-season

Value

list of various items

Examples

```
Mackenzie_Blackwood_EDGE_summary_regular_20242025 <- goalie_edge_summary(  
  player = 8478406,  
  season = 20242025,  
  game_type = 2  
)
```

goalie_game_report	<i>Access various reports for a season, game type, and category for all the goalies by game</i>
--------------------	---

Description

goalie_game_report() retrieves various reports for a season, game type, and category for all the goalies by game as a data.frame where each row represents game per goalie and includes detail on game timeline state, period/clock progression, and matchup flow, player identity, role, handedness, and biographical profile, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
goalie_game_report(  
  season = season_now(),  
  game_type = game_type_now(),  
  category = "summary"  
)
```

Arguments

season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
category	character (e.g., 'advanced'); see goalie_report_configurations() for reference

Value

data.frame with one row per game per goalie

Examples

```
# May take >5s, so skip.
advanced_goalie_game_report_playoffs_20212022 <-
  goalie_game_report(
    season    = 20212022,
    game_type = 3,
    category  = 'advanced'
  )
```

`goalie_game_scoring` *Access the scoring statistics for all the goalies by game*

Description

`goalie_game_scoring()` retrieves the scoring statistics for all the goalies by game as a `data.frame` with detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and team identity, affiliation, and matchup-side context.

Usage

```
goalie_game_scoring()
```

Value

`data.frame` with one row per player per game

Examples

```
goalie_game_scoring <- goalie_game_scoring()
```

`goalie_game_statistics`
Access the statistics for all the goalies by game

Description

`goalie_game_statistics()` retrieves the statistics for all the goalies by game as a `data.frame` with detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and team identity, affiliation, and matchup-side context.

Usage

```
goalie_game_statistics()

goalie_game_stats()
```

Value

data.frame with one row per goalie per game

Examples

```
goalie_game_stats <- goalie_game_statistics()
```

goalie_leaders	<i>Access the goalie statistics leaders for a season, game type, and category</i>
----------------	---

Description

goalie_leaders() retrieves the goalie statistics leaders for a season, game type, and category as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile.

Usage

```
goalie_leaders(season = "current", game_type = "", category = "wins")
```

Arguments

season	integer in YYYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff/post'; see seasons() for reference; most functions will NOT support pre-season
category	character of 'w/' wins', 's/' shutouts', 's%'/'sP'/'save %'/'save percentage', or 'gaa'/'goals against average'

Value

data.frame with one row per player

Examples

```
GAA_leaders_regular_20242025 <- goalie_leaders(
  season    = 20242025,
  game_type = 2,
  category  = 'GAA'
)
```

goalie_milestones *Access the goalies on milestone watch*

Description

goalie_milestones() retrieves the goalies on milestone watch as a data.frame where each row represents player and includes detail on date/season filtering windows and chronological context, player identity, role, handedness, and biographical profile, and ranking movement, points pace, and division/conference position signals.

Usage

```
goalie_milestones()
```

Value

data.frame with one row per player

Examples

```
goalie_milestones <- goalie_milestones()
```

goalie_regular_statistics *Access the career regular season statistics for all the goalies*

Description

goalie_regular_statistics() retrieves the career regular season statistics for all the goalies as a data.frame where each row represents goalie and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and player identity, role, handedness, and biographical profile.

Usage

```
goalie_regular_statistics()
```

```
goalie_regular_stats()
```

Value

data.frame with one row per goalie

Examples

```
goalie_career_regular_statistics <- goalie_regular_statistics()
```

`goalie_report_configurations`*Access the configurations for goalie reports*

Description

`goalie_report_configurations()` retrieves the configurations for goalie reports as a nested list that separates summary and detail blocks for situational splits across home/road, strength state, and overtime/shootout states plus configuration catalogs for valid report categories and filters.

Usage

```
goalie_report_configurations()
```

```
goalie_report_configs()
```

Value

list with various items

Examples

```
goalie_report_configs <- goalie_report_configurations()
```

`goalie_scoring`*Access the career scoring statistics for all the goalies*

Description

`goalie_scoring()` retrieves the career scoring statistics for all the goalies as a `data.frame` where each row represents player and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and player identity, role, handedness, and biographical profile.

Usage

```
goalie_scoring()
```

Value

`data.frame` with one row per player

Examples

```
goalie_scoring <- goalie_scoring()
```

goalie_season_report *Access various reports for a season, game type, and category for all the goalies by season*

Description

goalie_season_report() retrieves various reports for a season, game type, and category for all the goalies by season as a data.frame where each row represents player and includes detail on date/season filtering windows and chronological context, player identity, role, handedness, and biographical profile, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
goalie_season_report(  
  season = season_now(),  
  game_type = game_type_now(),  
  category = "summary"  
)
```

Arguments

season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
category	character (e.g., 'advanced'); see goalie_report_configurations() for reference

Value

data.frame with one row per player

Examples

```
# May take >5s, so skip.  
advanced_goalie_season_report_playoffs_20212022 <-  
  goalie_season_report(  
    season = 20212022,  
    game_type = 3,  
    category = 'advanced'  
  )
```

`goalie_season_statistics`*Access the statistics for all the goalies by season, game type, and team.*

Description

`goalie_season_statistics()` retrieves the statistics for all the goalies by season, game type, and team as a `data.frame` where each row represents player per season per game type, separated by team if applicable and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and player identity, role, handedness, and biographical profile.

Usage

```
goalie_season_statistics()
```

```
goalie_season_stats()
```

Value

`data.frame` with one row per player per season per game type, separated by team if applicable

Examples

```
goalie_season_stats <- goalie_season_statistics()
```

`goalie_series_statistics`*Access the playoff statistics for all the goalies by series*

Description

`goalie_series_statistics()` retrieves the playoff statistics for all the goalies by series as a `data.frame` where each row represents player per series and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and player identity, role, handedness, and biographical profile.

Usage

```
goalie_series_statistics()
```

```
goalie_series_stats()
```

Value

`data.frame` with one row per player per series

Examples

```
goalie_series_stats <- goalie_series_statistics()
```

```
goalie_statistics
```

Access the career statistics for all the goalies

Description

`goalie_statistics()` retrieves the career statistics for all the goalies as a `data.frame` where each row represents player and includes detail on team identity, affiliation, and matchup-side context, player identity, role, handedness, and biographical profile, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
goalie_statistics()
```

```
goalie_stats()
```

Value

`data.frame` with one row per player

Examples

```
goalie_stats <- goalie_statistics()
```

```
ig_game_cumulative_expected_goals
```

Save an Instagram (IG) share-able cumulative expected goals (xG) time-series plot for a game

Description

`ig_game_cumulative_expected_goals()` saves an IG share-able cumulative xG time-series plot for a given game as a PNG.

Usage

```
ig_game_cumulative_expected_goals(game = 2023030417, model = NULL, save = TRUE)
```

```
ig_game_cum_xG(game = 2023030417, model = NULL)
```

Arguments

game	integer ID (e.g., 2025020275); see <code>games()</code> for reference
model	deprecated legacy model selector; ignored
save	logical only FALSE for tests

Value

NULL

Examples

```
# May take >5s, so skip.
ig_game_cumulative_expected_goals(
  game = 2023030417,
  save = FALSE
)
```

```
ig_game_shot_locations
```

Save an Instagram (IG) share-able shot-location plot for a game

Description

`ig_game_shot_locations()` saves an IG share-able shot location plot for a given game.

Usage

```
ig_game_shot_locations(
  game = 2023030417,
  team = "home",
  model = NULL,
  save = TRUE
)
```

```
ig_game_shot_locs(game = 2023030417, team = "home", model = NULL)
```

Arguments

game	integer ID (e.g., 2025020275); see <code>games()</code> for reference
team	character of 'h'/'home' or 'a'/'away'
model	deprecated legacy model selector; ignored
save	logical only FALSE for tests

Value

NULL

Examples

```
# May take >5s, so skip.
ig_game_shot_locations(
  game = 2023030417,
  team = 'H',
  save = FALSE
)
```

location	<i>Access the location for a zip code</i>
----------	---

Description

location() retrieves the location for a zip code as a data.frame where each row represents team and includes detail on venue/location geography and regional metadata.

Usage

```
location(zip = 10001)
```

Arguments

zip integer (e.g., 48304)

Value

data.frame with one row per team

Examples

```
Cranbrook_Schools <- location(48304)
```

lottery_odds	<i>Access the draft lottery odds</i>
--------------	--------------------------------------

Description

lottery_odds() retrieves the draft lottery odds as a data.frame where each row represents draft lottery and includes detail on draft-cycle evaluation, ranking, and selection tracking detail.

Usage

```
lottery_odds()
```

Value

data.frame with one row per draft lottery

Examples

```
lottery_odds <- lottery_odds()
```

nhlscraper

nhlscraper: Scrape, clean, and visualize NHL data

Description

nhlscraper is a minimum-dependency R package to scrape, clean, and visualize NHL data via the NHL and ESPN APIs. It primarily wraps 125+ **endpoints** from high-level multi-season summaries and award winners to low-level decisecond replays and bookmakers' odds, making them significantly more accessible. It also features cleaning and visualization functions, primarily for play-by-plays, to help analyze the data.

Prerequisite

- R/RStudio; you can check out my **tutorial** if you are not familiar.

Disclosure

Detailed documentation for each scraping function will be released gradually over time. Because each function can return a large amount of information, users are encouraged to explore the provided examples to discover what is available. Most, if not all, of the endpoints accessed by this package are unofficially documented (i.e., hidden), so it is important to use them responsibly and with respect for the NHL's data servers. Endpoints serving historical or otherwise mostly static data should ideally be queried once and then stored locally (for example, in a MySQL database) for further analysis, rather than being called repeatedly; the **load functions** attempt to mitigate this risk by providing pre-scraped data, so please make use of them. The exact rate limits for these APIs are not publicly known, so users are asked to avoid excessive or abusive querying to help ensure continued access for everyone.

History

Prior to the NHL API rework in 2023, Drew Hynes documented a comprehensive list of known **endpoints**, and several R packages, such as **nhlapi** and **hockeyR**, were built to access them. However, after the NHL completely transformed its API structure, all of these packages became mostly defunct as their authors understandably chose not to continue maintaining them. The community gathered around work by Zachary Maludzinski to discover and share new **endpoints**, but progress naturally slowed once most of the "main" endpoints were identified. Over the summer of 2025, I began reverse-engineering many of the remaining, undocumented endpoints while searching for additional data for future research, with a particular focus on NHL EDGE and Records data. After I shared those **findings**, the effort to expand and refine this map of the APIs accelerated, and with the addition of David Fleischer's **effort**, we ultimately identified 400+ new endpoints. In parallel, I also discovered many new endpoints for the ESPN API, extending beyond what Joseph Wilson had already **compiled**.

Author(s)

Maintainer: Rento Saijo <rentosaijo0527@gmail.com> ([ORCID](#)) [copyright holder]

Other contributors:

- Lars Skytte <lars.sunesen.skytte@gmail.com> [contributor]
- Jack Pallotta <jackjpallotta@gmail.com> ([ORCID](#)) [contributor]

See Also

Useful links:

- <https://rentosaijo.github.io/nhlscraper/>
- <https://github.com/RentoSaijo/nhlscraper>
- Report bugs at <https://github.com/RentoSaijo/nhlscraper/issues>

officials

Access all the officials

Description

`officials()` retrieves all the officials as a data.frame where each row represents official and includes detail on player identity, role, handedness, and biographical profile plus coach/management/officiating identity and assignment history.

Usage

```
officials()
```

Value

data.frame with one row per official

Examples

```
all_officials <- officials()
```

penalty_shots	<i>Access all the penalty shots</i>
---------------	-------------------------------------

Description

penalty_shots() retrieves all the penalty shots as a data.frame where each row represents penalty shot and includes detail on game timeline state, period/clock progression, and matchup flow plus date/season filtering windows and chronological context.

Usage

```
penalty_shots()
```

```
pss()
```

Value

data.frame with one row per penalty shot

Examples

```
all_pss <- penalty_shots()
```

ping	<i>Ping</i>
------	-------------

Description

ping() is defunct.

Usage

```
ping()
```

player_game_log	<i>Access the game log for a player, season, and game type</i>
-----------------	--

Description

player_game_log() retrieves the game log for a player, season, and game type as a data.frame where each row represents game and includes detail on game timeline state, period/clock progression, and matchup flow plus production, workload, efficiency, and result-level performance outcomes.

Usage

```
player_game_log(player = 8478402, season = "now", game_type = "")
```

Arguments

player	integer ID (e.g., 8480039); see players() for reference
season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season

Value

data.frame with one row per game

Examples

```
Martin_Necas_game_log_regular_20242025 <- player_game_log(
  player = 8480039,
  season = 20242025,
  game_type = 2
)
```

player_seasons	<i>Access the season(s) and game type(s) in which a player played</i>
----------------	---

Description

player_seasons() retrieves the season(s) and game type(s) in which a player played as a data.frame where each row represents season and includes detail on date/season filtering windows and chronological context.

Usage

```
player_seasons(player = 8478402)
```

Arguments

player integer ID (e.g., 8480039); see [players\(\)](#) for reference

Value

data.frame with one row per season

Examples

```
Martin_Necas_seasons <- player_seasons(player = 8480039)
```

player_summary *Access the summary for a player*

Description

`player_summary()` retrieves the summary for a player as a nested list that separates summary and detail blocks for player identity, role, handedness, and biographical profile.

Usage

```
player_summary(player = 8478402)
```

Arguments

player integer ID (e.g., 8480039); see [players\(\)](#) for reference

Value

list with various items

Examples

```
Martin_Necas_summary <- player_summary(player = 8480039)
```

players	<i>Access all the players</i>
---------	-------------------------------

Description

players() retrieves all the players as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile.

Usage

```
players()
```

Value

data.frame with one row per player

Examples

```
# May take >5s, so skip.  
all_players <- players()
```

playoff_season_statistics	<i>Access the playoff statistics by season</i>
---------------------------	--

Description

playoff_season_statistics() retrieves the playoff statistics by season as a data.frame where each row represents season and includes detail on date/season filtering windows and chronological context.

Usage

```
playoff_season_statistics()
```

```
playoff_season_stats()
```

Value

data.frame with one row per season

Examples

```
playoff_season_stats <- playoff_season_statistics()
```

ps *Access all the penalty shots*

Description

ps() is deprecated. Use [penalty_shots\(\)](#) instead.

Usage

```
ps()
```

Value

data.frame with one row per penalty shot

replay *Access the replay for an event*

Description

replay() retrieves the replay for an event as a data.frame where each row represents decisecond and includes detail on team identity, affiliation, and matchup-side context plus player identity, role, handedness, and biographical profile.

Usage

```
replay(game = 2023030417, event = 866)
```

Arguments

game	integer ID (e.g., 2025020262); see games() for reference
event	integer ID (e.g., 751); see gc_play_by_play() and/or wsc_play_by_play() for reference; must be a 'goal' event

Value

data.frame with one row per decisecond

Examples

```
Gabriel_Landeskog_first_regular_goal_back_replay <- replay(
  game = 2025020262,
  event = 751
)
```

replays	<i>Access the replays for a season</i>
---------	--

Description

replays() loads the replays for a given season.

Usage

```
replays(season = 20242025)
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per decisecond

Examples

```
# May take >5s, so skip.
replays_20252026 <- replays(season = 20252026)
```

roster	<i>Access the roster for a team, season, and position</i>
--------	---

Description

roster() retrieves the roster for a team, season, and position as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile.

Usage

```
roster(team = 1, season = "current", position = "forwards")
```

Arguments

team integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see [teams\(\)](#) for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

position character of 'f'/'forwards', 'd'/'defensemen', or 'g'/'goalies'

Value

data.frame with one row per player

Examples

```
COL_defensemen_20242025 <- roster(
  team    = 21,
  season  = 20242025,
  position = 'D'
)
```

roster_statistics *Access the roster statistics for a team, season, game type, and position*

Description

roster_statistics() retrieves the roster statistics for a team, season, game type, and position as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile plus production, workload, efficiency, and result-level performance outcomes.

Usage

```
roster_statistics(
  team = 1,
  season = "now",
  game_type = "",
  position = "skaters"
)
```

```
roster_stats(team = 1, season = "now", game_type = "", position = "skaters")
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
position	character of 's'/'skaters' or 'g'/'goalies'

Value

data.frame with one row per player

Examples

```
COL_goalies_statistics_regular_20242025 <- roster_statistics(
  team      = 21,
  season    = 20242025,
  game_type = 2,
  position  = 'G'
)
```

schedule	<i>Access the schedule for a date</i>
----------	---------------------------------------

Description

`schedule()` retrieves the schedule for a date as a `data.frame` where each row represents game and includes detail on game timing, matchup state, scoring flow, and situational event detail.

Usage

```
schedule(date = Sys.Date())
```

Arguments

date	character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see seasons() for reference
------	---

Value

`data.frame` with one row per game

Examples

```
schedule_Halloween_2025 <- schedule(date = '2025-10-31')
```

scores	<i>Access the scores for a date</i>
--------	-------------------------------------

Description

`scores()` retrieves the scores for a date as a `data.frame` where each row represents game and includes detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and team identity, affiliation, and matchup-side context.

Usage

```
scores(date = "now")
```

Arguments

date character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see [seasons\(\)](#) for reference

Value

data.frame with one row per game

Examples

```
scores_Halloween_2025 <- scores(date = '2025-10-31')
```

season_now	<i>Access the season as of now</i>
------------	------------------------------------

Description

season_now() retrieves the season as of now and returns a scalar integer used as the current-context default in season/game-type dependent wrappers.

Usage

```
season_now()
```

Value

integer in YYYYYYYY (e.g., 20242025)

Examples

```
season_now <- season_now()
```

seasons	<i>Access all the seasons</i>
---------	-------------------------------

Description

seasons() retrieves all the seasons as a data.frame where each row represents season and includes detail on date/season filtering windows and chronological context.

Usage

```
seasons()
```

Value

data.frame with one row per season

Examples

```
all_seasons <- seasons()
```

series	<i>Access all the playoff series by game</i>
--------	--

Description

`series()` retrieves all the playoff series by game as a `data.frame` where each row represents game per series and includes detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and playoff-series progression, round status, and series leverage.

Usage

```
series()
```

Value

`data.frame` with one row per game per series

Examples

```
# May take >5s, so skip.
all_series <- series()
```

series_schedule	<i>Access the playoff schedule for a season and series</i>
-----------------	--

Description

`series_schedule()` retrieves the playoff schedule for a season and series as a `data.frame` where each row represents game and includes detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and team identity, affiliation, and matchup-side context.

Usage

```
series_schedule(season = season_now() - 10001, series = "a")
```

Arguments

season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
series	one-letter code (e.g., 'O'); see series() and/or bracket() for reference

Value

data.frame with one row per game

Examples

```
SCF_schedule_20212022 <- series_schedule(  
  season = 20212022,  
  series = '0'  
)
```

shift_chart	<i>Access the shift chart for a game</i>
-------------	--

Description

shift_chart() retrieves the shift chart for a game as a data.frame where each row represents shift and includes detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and team identity, affiliation, and matchup-side context.

Usage

```
shift_chart(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per shift

Examples

```
shifts_Martin_Necas_legacy_game <- shift_chart(game = 2025020275)
```

shift_chart_summaries *Access the shift chart summaries for a season*

Description

shift_chart_summaries() loads the shift chart summaries for a given season.

Usage

```
shift_chart_summaries(season = 20242025)
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per player per period

Examples

```
# May take >5s, so skip.  
shift_chart_summaries_20212022 <- shift_chart_summaries(season = 20212022)
```

shift_chart_summary *Access the shift chart time-on-ice summary for a game*

Description

shift_chart_summary() retrieves per-player, per-period time-on-ice splits from the HTML shift reports.

Usage

```
shift_chart_summary(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per player per period

Examples

```
shift_summary_2023030417 <- shift_chart_summary(game = 2023030417)
```

shift_charts	<i>Access the shift charts for a season</i>
--------------	---

Description

shift_charts() loads the shift charts for a given season.

Usage

```
shift_charts(season = 20242025)
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per event (play) per game

Examples

```
# May take >5s, so skip.  
shift_charts_20212022 <- shift_charts(season = 20212022)
```

shifts	<i>Access the shift charts for a game</i>
--------	---

Description

shifts() is deprecated. Use [shift_chart\(\)](#) instead.

Usage

```
shifts(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per shift

skater_edge_leaders *Access the skater EDGE statistics leaders for a season and game type*

Description

skater_edge_leaders() retrieves the skater EDGE statistics leaders for a season and game type as a nested list that separates summary and detail blocks for NHL EDGE style tracking outputs and relative-performance context.

Usage

```
skater_edge_leaders(season = "now", game_type = "")
```

Arguments

season	integer in YYYYYYYYY (e.g., 20242025); see skater_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see skater_edge_seasons() for reference; most functions will NOT support pre-season

Value

list of various items

Examples

```
skater_EDGE_leaders_regular_20242025 <- skater_edge_leaders(
  season    = 20242025,
  game_type = 2
)
```

skater_edge_seasons *Access the season(s) and game type(s) in which there exists skater EDGE statistics*

Description

skater_edge_seasons() retrieves the season(s) and game type(s) in which there exists skater EDGE statistics as a data.frame where each row represents season and includes detail on date/season filtering windows and chronological context plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
skater_edge_seasons()
```

Value

data.frame with one row per season

Examples

```
skater_EDGE_seasons <- skater_edge_seasons()
```

```
skater_edge_shot_location
```

Access the EDGE shot location statistics for a skater, season, game type, and category

Description

`skater_edge_shot_location()` retrieves the EDGE shot location statistics for a skater, season, game type, and category as a `data.frame` where each row represents shot location and includes detail on production, workload, efficiency, and result-level performance outcomes plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
skater_edge_shot_location(
  player = 8478402,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

<code>player</code>	integer ID (e.g., 8480039); see players() for reference
<code>season</code>	integer in YYYYYYYYY (e.g., 20242025); see skater_edge_seasons() for reference
<code>game_type</code>	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see skater_edge_seasons() for reference; most functions will NOT support pre-season
<code>category</code>	character of 'd'/details' or 't'/'totals'

Value

data.frame with one row per shot location

Examples

```
Martin_Necas_shot_location_totals_regular_20242025 <-
  skater_edge_shot_location(
    player    = 8480039,
    season    = 20242025,
    game_type = 2,
    category  = 'T'
  )
```

```
skater_edge_shot_speed
```

Access the EDGE shot speed statistics for a skater, season, game type, and category

Description

skater_edge_shot_speed() retrieves the EDGE shot speed statistics for a skater, season, game type, and category as a nested list that separates summary and detail blocks for NHL EDGE style tracking outputs and relative-performance context.

Usage

```
skater_edge_shot_speed(
  player = 8478402,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

player	integer ID (e.g., 8480039); see players() for reference
season	integer in YYYYYYYYY (e.g., 20242025); see skater_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see skater_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 'h'/'hardest'

Value

list with six items (category = 'details') or data.frame with one row per shot (category = 'hardest')

Examples

```
Martin_Necas_hardest_shots_regular_20242025 <- skater_edge_shot_speed(
  player = 8480039,
  season = 20242025,
  game_type = 2,
  category = 'H'
)
```

```
skater_edge_skating_distance
```

Access the EDGE skating distance statistics for a skater, season, game type, and category

Description

`skater_edge_skating_distance()` retrieves the EDGE skating distance statistics for a skater, season, game type, and category as a `data.frame` where each row represents strength state and includes detail on team identity, affiliation, and matchup-side context, player identity, role, handedness, and biographical profile, and tracking/spatial detail such as location, speed, distance, and zone distribution.

Usage

```
skater_edge_skating_distance(
  player = 8478402,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

<code>player</code>	integer ID (e.g., 8480039); see players() for reference
<code>season</code>	integer in YYYYYYYY (e.g., 20242025); see skater_edge_seasons() for reference
<code>game_type</code>	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see skater_edge_seasons() for reference; most functions will NOT support pre-season
<code>category</code>	character of 'd'/'details' or 'l'/'last 10'

Value

`data.frame` with one row per strength state (category = 'details') or game (category = 'last 10')

Examples

```
Martin_Necas_L10_skating_distance_regular_20242025 <-
  skater_edge_skating_distance(
    player    = 8480039,
    season    = 20242025,
    game_type = 2,
    category  = 'L'
  )
```

```
skater_edge_skating_speed
```

Access the EDGE skating speed statistics for a skater, season, game type, and category

Description

skater_edge_skating_speed() retrieves the EDGE skating speed statistics for a skater, season, game type, and category as a nested list that separates summary and detail blocks for NHL EDGE style tracking outputs and relative-performance context.

Usage

```
skater_edge_skating_speed(
  player = 8478402,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

player	integer ID (e.g., 8480039); see players() for reference
season	integer in YYYYYYYYY (e.g., 20242025); see skater_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see skater_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 't'/'top'/'top speeds'

Value

list with four items (category = 'details') or data.frame with one row per burst (category = 'top speeds')

Examples

```
Martin_Necas_top_speeds_regular_20242025 <- skater_edge_skating_speed(  
  player = 8480039,  
  season = 20242025,  
  game_type = 2,  
  category = 'T'  
)
```

skater_edge_summary *Access the EDGE summary for a skater, season, and game type*

Description

skater_edge_summary() retrieves the EDGE summary for a skater, season, and game type as a nested list that separates summary and detail blocks for player identity, role, handedness, and biographical profile plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
skater_edge_summary(player = 8478402, season = "now", game_type = "")
```

Arguments

player	integer ID (e.g., 8480039); see players() for reference
season	integer in YYYYYYYYY (e.g., 20242025); see skater_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see skater_edge_seasons() for reference; most functions will NOT support pre-season

Value

list of various items

Examples

```
Martin_Necas_EDGE_summary_regular_20242025 <- skater_edge_summary(  
  player = 8480039,  
  season = 20242025,  
  game_type = 2  
)
```

skater_edge_zone_time *Access the EDGE zone time statistics for a skater, season, game type, and category*

Description

skater_edge_zone_time() retrieves the EDGE zone time statistics for a skater, season, game type, and category as a data.frame where each row represents strength state and includes detail on NHL EDGE style tracking outputs and relative-performance context.

Usage

```
skater_edge_zone_time(  
  player = 8478402,  
  season = "now",  
  game_type = "",  
  category = "details"  
)
```

Arguments

player	integer ID (e.g., 8480039); see players() for reference
season	integer in YYYYYYYY (e.g., 20242025); see skater_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see skater_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 's'/'starts'

Value

data.frame with one row per strength state (category = 'details') or list with six items (category = 'starts')

Examples

```
Martin_Necas_starts_regular_20242025 <- skater_edge_zone_time(  
  player = 8480039,  
  season = 20242025,  
  game_type = 2,  
  category = 'S'  
)
```

skater_game_report	<i>Access various reports for a season, game type, and category for all the skaters by game</i>
--------------------	---

Description

skater_game_report() retrieves various reports for a season, game type, and category for all the skaters by game as a data.frame where each row represents game per player and includes detail on game timeline state, period/clock progression, and matchup flow, player identity, role, handedness, and biographical profile, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
skater_game_report(  
  season = season_now(),  
  game_type = game_type_now(),  
  category = "summary"  
)
```

Arguments

season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
category	character (e.g., 'puckPossessions'); see skater_report_configurations() for reference

Value

data.frame with one row per game per player

Examples

```
# May take >5s, so skip.  
possession_skater_game_report_playoff_20212022 <-  
  skater_game_report(  
    season = 20212022,  
    game_type = 3,  
    category = 'puckPossessions'  
  )
```

skater_leaders	<i>Access the skater statistics leaders for a season, game type, and category</i>
----------------	---

Description

skater_leaders() retrieves the skater statistics leaders for a season, game type, and category as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile.

Usage

```
skater_leaders(season = "current", game_type = "", category = "points")
```

Arguments

season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff/post'; see seasons() for reference; most functions will NOT support pre-season
category	string of 'a'/assists', 'g'/goals', 'shg'/shorthanded goals', 'ppg'/powerplay goals', 'p'/points', 'pim'/penalty minutes'/penalty infraction minutes', 'toi'/time on ice', 'pm'/plus minus', or 'f'/faceoffs'

Value

data.frame with one row per player

Examples

```
TOI_leaders_regular_20242025 <- skater_leaders(
  season    = 20242025,
  game_type = 2,
  category  = 'TOI'
)
```

skater_milestones	<i>Access the skaters on milestone watch</i>
-------------------	--

Description

skater_milestones() retrieves the skaters on milestone watch as a data.frame where each row represents player and includes detail on date/season filtering windows and chronological context, player identity, role, handedness, and biographical profile, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
skater_milestones()
```

Value

data.frame with one row per player

Examples

```
skater_milestones <- skater_milestones()
```

skater_playoff_statistics

Access the career playoff statistics for all the skaters

Description

skater_playoff_statistics() retrieves the career playoff statistics for all the skaters as a data.frame where each row represents player and includes detail on team identity, affiliation, and matchup-side context, player identity, role, handedness, and biographical profile, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
skater_playoff_statistics()
```

```
skater_playoff_stats()
```

Value

data.frame with one row per player

Examples

```
skater_playoff_stats <- skater_playoff_statistics()
```

`skater_regular_statistics`*Access the career regular season statistics for all the skaters*

Description

`skater_regular_statistics()` retrieves the career regular season statistics for all the skaters as a `data.frame` where each row represents player and includes detail on team identity, affiliation, and matchup-side context, player identity, role, handedness, and biographical profile, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
skater_regular_statistics()
```

```
skater_regular_stats()
```

Value

`data.frame` with one row per player

Examples

```
skater_regular_stats <- skater_regular_statistics()
```

`skater_report_configurations`*Access the configurations for skater reports*

Description

`skater_report_configurations()` retrieves the configurations for skater reports as a nested `list` that separates summary and detail blocks for production, workload, efficiency, and result-level performance outcomes, situational splits across home/road, strength state, and overtime/shootout states, and configuration catalogs for valid report categories and filters.

Usage

```
skater_report_configurations()
```

```
skater_report_configs()
```

Value

`list` with various items

Examples

```
skater_report_configs <- skater_report_configurations()
```

`skater_season_report` *Access various reports for a season, game type, and category for all the skaters by season*

Description

`skater_season_report()` retrieves various reports for a season, game type, and category for all the skaters by season as a `data.frame` where each row represents player and includes detail on date/season filtering windows and chronological context, player identity, role, handedness, and biographical profile, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
skater_season_report(  
  season = season_now(),  
  game_type = game_type_now(),  
  category = "summary"  
)
```

Arguments

<code>season</code>	integer in YYYYYYYYY (e.g., 20242025); see seasons() for reference
<code>game_type</code>	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
<code>category</code>	character (e.g., 'puckPossessions'); see skater_report_configurations() for reference

Value

`data.frame` with one row per player

Examples

```
# May take >5s, so skip.  
possession_skater_season_report_playoff_20212022 <-  
  skater_season_report(  
    season = 20212022,  
    game_type = 3,  
    category = 'puckPossessions'  
  )
```

`skater_season_statistics`*Access the statistics for all the skaters by season, game type, and team*

Description

`skater_season_statistics()` retrieves the statistics for all the skaters by season, game type, and team as a `data.frame` where each row represents player per season per game type, separated by team if applicable and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and player identity, role, handedness, and biographical profile.

Usage`skater_season_statistics()``skater_season_stats()`**Value**

`data.frame` with one row per player per season per game type, separated by team if applicable

Examples

```
# May take >5s, so skip.  
skater_season_stats <- skater_season_statistics()
```

`skater_series_statistics`*Access the playoff statistics for all the skaters by series*

Description

`skater_series_statistics()` retrieves the playoff statistics for all the skaters by series as a `data.frame` where each row represents player per series and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and player identity, role, handedness, and biographical profile.

Usage`skater_series_statistics()``skater_series_stats()`**Value**

`data.frame` with one row per player per series

Examples

```
# May take >5s, so skip.  
skater_series_stats <- skater_series_statistics()
```

skater_statistics *Access the career statistics for all the skaters*

Description

skater_statistics() retrieves the career statistics for all the skaters as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile plus production, workload, efficiency, and result-level performance outcomes.

Usage

```
skater_statistics()
```

```
skater_stats()
```

Value

data.frame with one row per player

Examples

```
skater_stats <- skater_statistics()
```

spotlight_players *Access the spotlight players*

Description

spotlight_players() retrieves the spotlight players as a data.frame where each row represents player and includes detail on team identity, affiliation, and matchup-side context plus player identity, role, handedness, and biographical profile.

Usage

```
spotlight_players()
```

Value

data.frame with one row per player

Examples

```
spotlight_players <- spotlight_players()
```

standings	<i>Access the standings for a date</i>
-----------	--

Description

`standings()` retrieves the standings for a date as a `data.frame` where each row represents team and includes detail on date/season filtering windows and chronological context, production, workload, efficiency, and result-level performance outcomes, and ranking movement, points pace, and division/conference position signals.

Usage

```
standings(date = "now")
```

Arguments

date	character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see seasons() for reference
------	---

Value

`data.frame` with one row per team

Examples

```
standings_Halloween_2025 <- standings(date = '2025-10-31')
```

standings_rules	<i>Access the standings rules by season</i>
-----------------	---

Description

`standings_rules()` retrieves the standings rules by season as a `data.frame` where each row represents season and includes detail on date/season filtering windows and chronological context.

Usage

```
standings_rules()
```

Value

`data.frame` with one row per season

Examples

```
standings_rules <- standings_rules()
```

streams	<i>Access all the streams</i>
---------	-------------------------------

Description

`streams()` retrieves all the streams as a data.frame where each row represents stream and includes detail on reference metadata, regional context, and media availability detail.

Usage

```
streams()
```

Value

data.frame with one row per stream

Examples

```
all_streams <- streams()
```

team_edge_leaders	<i>Access the team EDGE statistics leaders for a season and game type</i>
-------------------	---

Description

`team_edge_leaders()` retrieves the team EDGE statistics leaders for a season and game type as a nested list that separates summary and detail blocks for NHL EDGE style tracking outputs and relative-performance context.

Usage

```
team_edge_leaders(season = "now", game_type = "")
```

Arguments

season	integer in YYYYYYYYY (e.g., 20242025); see team_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see team_edge_seasons() for reference; most functions will NOT support pre-season

Value

list of various items

Examples

```
team_EDGE_leaders_regular_20242025 <- team_edge_leaders(
  season = 20242025,
  game_type = 2
)
```

team_edge_seasons	<i>Access the season(s) and game type(s) in which there exists team EDGE statistics</i>
-------------------	---

Description

team_edge_seasons() retrieves the season(s) and game type(s) in which there exists team EDGE statistics as a data.frame where each row represents season and includes detail on date/season filtering windows and chronological context plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
team_edge_seasons()
```

Value

data.frame with one row per season

Examples

```
team_EDGE_seasons <- team_edge_seasons()
```

team_edge_shot_location	<i>Access the EDGE shot location statistics for a team, season, game type, and category</i>
-------------------------	---

Description

team_edge_shot_location() retrieves the EDGE shot location statistics for a team, season, game type, and category as a data.frame where each row represents location and includes detail on production, workload, efficiency, and result-level performance outcomes plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
team_edge_shot_location(
  team = 1,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
season	integer in YYYYYYYYY (e.g., 20242025); see team_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see team_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 't'/'totals'

Value

data.frame with one row per location (category = 'details') or combination of strength state and position (category = 'totals')

Examples

```
COL_shot_location_totals_regular_20242025 <- team_edge_shot_location(
  team      = 21,
  season    = 20242025,
  game_type = 2,
  category  = 'T'
)
```

team_edge_shot_speed *Access the EDGE shot speed statistics for a team, season, game type, and category*

Description

team_edge_shot_speed() retrieves the EDGE shot speed statistics for a team, season, game type, and category as a data.frame where each row represents position and includes detail on team identity, affiliation, and matchup-side context, player identity, role, handedness, and biographical profile, and ranking movement, points pace, and division/conference position signals.

Usage

```
team_edge_shot_speed(
  team = 1,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
season	integer in YYYYYYYY (e.g., 20242025); see team_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see team_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 'h'/'hardest'

Value

data.frame with one row per position (category = 'details') or shot (category = 'hardest')

Examples

```
COL_hardest_shots_regular_20242025 <- team_edge_shot_speed(
  team      = 21,
  season    = 20242025,
  game_type = 2,
  category  = 'H'
)
```

team_edge_skating_distance

Access the EDGE skating distance statistics for a team, season, game type, and category

Description

team_edge_skating_distance() retrieves the EDGE skating distance statistics for a team, season, game type, and category as a data.frame where each row represents combination of strength state and position and includes detail on team identity, affiliation, and matchup-side context, ranking movement, points pace, and division/conference position signals, and NHL EDGE style tracking outputs and relative-performance context.

Usage

```
team_edge_skating_distance(
  team = 1,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
season	integer in YYYYYYYY (e.g., 20242025); see team_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see team_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 'l'/'l10'/'last 10'

Value

data.frame with one row per combination of strength state and position (category = 'details') or game (category = 'last 10') game

Examples

```
COL_L10_skating_distance_regular_20242025 <- team_edge_skating_distance(
  team      = 21,
  season    = 20242025,
  game_type = 2,
  category  = 'L'
)
```

team_edge_skating_speed

Access the EDGE skating speed statistics for a team, season, game type, and category

Description

team_edge_skating_speed() retrieves the EDGE skating speed statistics for a team, season, game type, and category as a data.frame where each row represents position and includes detail on team identity, affiliation, and matchup-side context, player identity, role, handedness, and biographical profile, and ranking movement, points pace, and division/conference position signals.

Usage

```
team_edge_skating_speed(
  team = 1,
  season = "now",
  game_type = "",
  category = "details"
)
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
season	integer in YYYYYYYY (e.g., 20242025); see team_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see team_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 't'/'top'/'top speeds'

Value

data.frame with one row per position (category = 'details') or burst (category = 'top speeds')

Examples

```
COL_top_speeds_regular_20242025 <- team_edge_skating_speed(
  team      = 21,
  season    = 20242025,
  game_type = 2,
  category  = 'T'
)
```

team_edge_summary	<i>Access the EDGE summary for a team, season, and game type</i>
-------------------	--

Description

`team_edge_summary()` retrieves the EDGE summary for a team, season, and game type as a nested list that separates summary and detail blocks for team identity, affiliation, and matchup-side context plus NHL EDGE style tracking outputs and relative-performance context.

Usage

```
team_edge_summary(team = 1, season = "now", game_type = "")
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
season	integer in YYYYYYYY (e.g., 20242025); see team_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see team_edge_seasons() for reference; most functions will NOT support pre-season

Value

list of various items

Examples

```
COL_EDGE_summary_regular_20242025 <- team_edge_summary(  
  team      = 21,  
  season    = 20242025,  
  game_type = 2  
)
```

team_edge_zone_time *Access the EDGE zone time statistics for a team, season, game type, and category*

Description

team_edge_zone_time() retrieves the EDGE zone time statistics for a team, season, game type, and category as a data.frame where each row represents strength state and includes detail on NHL EDGE style tracking outputs and relative-performance context.

Usage

```
team_edge_zone_time(  
  team = 1,  
  season = "now",  
  game_type = "",  
  category = "details"  
)
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
season	integer in YYYYYYYY (e.g., 20242025); see team_edge_seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see team_edge_seasons() for reference; most functions will NOT support pre-season
category	character of 'd'/'details' or 'dS'/'dSOG'/'dShot'/'shot differential'

Value

data.frame with one row per strength state (category = 'details') or list with four items (category = 'shot differential')

Examples

```
COL_dS_regular_20242025 <- team_edge_zone_time(
  team      = 21,
  season    = 20242025,
  game_type = 2,
  category  = 'dS'
)
```

team_game_report	<i>Access various reports for a season, game type, and category for all the teams by game</i>
------------------	---

Description

`team_game_report()` retrieves various reports for a season, game type, and category for all the teams by game as a data.frame where each row represents game per team and includes detail on game timeline state, period/clock progression, and matchup flow, team identity, affiliation, and matchup-side context, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
team_game_report(
  season = season_now(),
  game_type = game_type_now(),
  category = "summary"
)
```

Arguments

season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
category	character (e.g., 'leadingtrailing'); see team_report_configurations() for reference

Value

data.frame with one row per game per team

Examples

```
situational_team_game_report_playoffs_20212022 <- team_game_report(
  season = 20212022,
  game_type = 3,
  category = 'leadingtrailing'
)
```

team_logos	<i>Access all the team logos</i>
------------	----------------------------------

Description

team_logos() retrieves all the team logos as a data.frame where each row represents logo and includes detail on team identity, affiliation, and matchup-side context.

Usage

```
team_logos()
```

Value

data.frame with one row per logo

Examples

```
all_team_logos <- team_logos()
```

team_month_schedule *Access the schedule for a team and month*

Description

team_month_schedule() retrieves the schedule for a team and month as a data.frame where each row represents game and includes detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and team identity, affiliation, and matchup-side context.

Usage

```
team_month_schedule(team = 1, month = "now")
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
month	character in 'YYYY-MM' (e.g., '2025-01'); see seasons() for reference

Value

data.frame with one row per game

Examples

```
COL_schedule_December_2025 <- team_month_schedule(  
  team = 21,  
  month = '2025-12'  
)
```

team_prospects *Access the prospects for a team and position*

Description

team_prospects() retrieves the prospects for a team and position as a data.frame where each row represents player and includes detail on player identity, role, handedness, and biographical profile.

Usage

```
team_prospects(team = 1, position = "forwards")
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
position	character of 'f'/'forwards', 'd'/'defensemen', or 'g'/'goalies'

Value

data.frame with one row per player

Examples

```
COL_forward_prospects <- team_prospects(  
  team = 21,  
  position = 'F'  
)
```

team_report_configurations

Access the configurations for team reports

Description

team_report_configurations() retrieves the configurations for team reports as a nested list that separates summary and detail blocks for situational splits across home/road, strength state, and overtime/shootout states plus configuration catalogs for valid report categories and filters.

Usage

```
team_report_configurations()
```

```
team_report_configs()
```

Value

list with various items

Examples

```
team_report_configs <- team_report_configurations()
```

team_season_report	<i>Access various reports for a season, game type, and category for all the teams by season</i>
--------------------	---

Description

team_season_report() retrieves various reports for a season, game type, and category for all the teams by season as a data.frame where each row represents team and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
team_season_report(
  season = season_now(),
  game_type = game_type_now(),
  category = "summary"
)
```

Arguments

season	integer in YYYYYYYYY (e.g., 20242025); see seasons() for reference
game_type	integer in 1:3 (where 1 = pre-season, 2 = regular season, 3 = playoff/post-season) OR character of 'pre', 'regular', or 'playoff'/'post'; see seasons() for reference; most functions will NOT support pre-season
category	character (e.g., 'leadingtrailing'); see team_report_configurations() for reference

Value

data.frame with one row per team

Examples

```
situational_team_season_report_playoffs_20212022 <- team_season_report(
  season = 20212022,
  game_type = 3,
  category = 'leadingtrailing'
)
```

team_season_schedule *Access the schedule for a team and season*

Description

team_season_schedule() retrieves the schedule for a team and season as a data.frame where each row represents game and includes detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and team identity, affiliation, and matchup-side context.

Usage

```
team_season_schedule(team = 1, season = "now")
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
season	integer in YYYYYYYY (e.g., 20242025); see seasons() for reference

Value

data.frame with one row per game

Examples

```
COL_schedule_20252026 <- team_season_schedule(  
  team = 21,  
  season = 20252026  
)
```

team_season_statistics

Access the statistics for all the teams by season and game type

Description

team_season_statistics() retrieves the statistics for all the teams by season and game type as a data.frame where each row represents team per season per game type and includes detail on date/season filtering windows and chronological context, team identity, affiliation, and matchup-side context, and production, workload, efficiency, and result-level performance outcomes.

Usage

```
team_season_statistics()

team_season_stats()
```

Value

data.frame with one row per team per season per game type

Examples

```
# May take >5s, so skip.
team_season_statistics <- team_season_statistics()
```

team_seasons	<i>Access the season(s) and game type(s) in which a team played</i>
--------------	---

Description

team_seasons() retrieves the season(s) and game type(s) in which a team played as a data.frame where each row represents season and includes detail on date/season filtering windows and chronological context.

Usage

```
team_seasons(team = 1)
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see teams() for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
------	---

Value

data.frame with one row per season

Examples

```
COL_seasons <- team_seasons(team = 21)
```

team_week_schedule	<i>Access the schedule for a team and week since a date</i>
--------------------	---

Description

`team_week_schedule()` retrieves the schedule for a team and week since a date as a `data.frame` where each row represents game and includes detail on game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and team identity, affiliation, and matchup-side context.

Usage

```
team_week_schedule(team = 1, date = "now")
```

Arguments

team	integer ID (e.g., 21), character full name (e.g., 'Colorado Avalanche'), OR three-letter code (e.g., 'COL'); see <code>teams()</code> for reference; ID is preferable as there now exists duplicate three-letter codes (i.e., 'UTA' for 'Utah Hockey Club' and 'Utah Mammoth')
date	character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see <code>seasons()</code> for reference

Value

`data.frame` with one row per game

Examples

```
COL_schedule_Family_Week_2025 <- team_week_schedule(  
  team = 21,  
  date = '2025-10-06'  
)
```

teams	<i>Access all the teams</i>
-------	-----------------------------

Description

`teams()` retrieves all the teams as a `data.frame` where each row represents team and includes detail on team identity, affiliation, and matchup-side context.

Usage

```
teams()
```

Value

data.frame with one row per team

Examples

```
all_teams <- teams()
```

tv_schedule	<i>Access the NHL Network TV schedule for a date</i>
-------------	--

Description

tv_schedule() retrieves the NHL Network TV schedule for a date as a data.frame where each row represents program and includes detail on date/season filtering windows and chronological context.

Usage

```
tv_schedule(date = "now")
```

Arguments

date	character in 'YYYY-MM-DD' (e.g., '2025-01-01'); see seasons() for reference
------	---

Value

data.frame with one row per program

Examples

```
tv_schedule_Halloween_2025 <- tv_schedule(date = '2025-10-31')
```

venues	<i>Access all the venues</i>
--------	------------------------------

Description

venues() retrieves all the venues as a data.frame where each row represents venue and includes detail on venue/location geography and regional metadata.

Usage

```
venues()
```

Value

data.frame with one row per venue

Examples

```
all_venues <- venues()
```

wsc_play_by_play	<i>Access the World Showcase (WSC) play-by-play for a game</i>
------------------	--

Description

wsc_play_by_play() retrieves the World Showcase (WSC) play-by-play for a game as a data.frame where each row represents an event. The returned schema follows the same cleaned public-facing naming as gc_play_by_play(), including servedByPlayerId, goalieInNetId, and utc immediately after secondsElapsedInGame while omitting GC-only clip fields. It also includes the same HTML-report-derived on-ice player ID columns added to the GC output, including dynamically expanded overflow skater slots when needed. HTML report skater and goalie IDs are returned whenever they can be matched back to a supported row, even when the raw situationCode is stale. Use [add_shift_times\(\)](#) with [shift_chart\(\)](#) (or [shift_charts\(\)](#)) to add on-ice shift timing columns.

Usage

```
wsc_play_by_play(game = 2023030417)
```

```
wsc_pbp(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per event (play)

Examples

```
# May take >5s, so skip.  
wsc_pbp_Martin_Necas_legacy_game <- wsc_play_by_play(game = 2025020275)
```

wsc_play_by_play_raw *Access the raw World Showcase (WSC) play-by-play for a game*

Description

wsc_play_by_play_raw() returns the raw flattened World Showcase play-by-play as served by the NHL API for one game. Use [wsc_play_by_play\(\)](#) for the cleaned public schema that repairs common clock/order defects and appends the derived public columns.

Usage

```
wsc_play_by_play_raw(game = 2023030417)
```

```
wsc_pbp_raw(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

data.frame with one row per event (play)

Examples

```
wsc_raw_Martin_Necas_legacy_game <- wsc_play_by_play_raw(game = 2025020275)
```

wsc_play_by_plays *Access the World Showcase (WSC) play-by-plays for a season*

Description

wsc_play_by_plays() loads the WSC play-by-plays for a given season.

Usage

```
wsc_play_by_plays(season = 20242025)
```

```
wsc_pbps(season = 20242025)
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per event (play) per game

Examples

```
# May take >5s, so skip.
wsc_pbps_20212022 <- wsc_play_by_plays(season = 20212022)
```

wsc_play_by_plays_raw *Access the raw World Showcase (WSC) play-by-plays for a season*

Description

wsc_play_by_plays_raw() loads the raw WSC play-by-plays for a given season.

Usage

```
wsc_play_by_plays_raw(season = 20242025)

wsc_pbps_raw(season = 20242025)
```

Arguments

season integer in YYYYYYYYY (e.g., 20242025); see [seasons\(\)](#) for reference

Value

data.frame with one row per raw event (play) per game

Examples

```
# May take >5s, so skip.
wsc_pbps_raw_20212022 <- wsc_play_by_plays_raw(season = 20212022)
```

wsc_summary *Access the World Showcase (WSC) summary for a game*

Description

wsc_summary() retrieves the World Showcase (WSC) summary for a game as a nested list that separates summary and detail blocks for game timeline state, period/clock progression, and matchup flow, date/season filtering windows and chronological context, and venue/location geography and regional metadata.

Usage

```
wsc_summary(game = 2023030417)
```

Arguments

game integer ID (e.g., 2025020275); see [games\(\)](#) for reference

Value

list of various items

Examples

```
wsc_summary_Martin_Necas_legacy_game <- wsc_summary(game = 2025020275)
```

```
x_game_cumulative_expected_goals
```

Save an X (Twitter) share-able cumulative expected goals (xG) time-series plot for a game

Description

`x_game_cumulative_expected_goals()` saves an X share-able cumulative xG time-series plot for a given game as a PNG.

Usage

```
x_game_cumulative_expected_goals(game = 2023030417, model = NULL, save = TRUE)
```

```
x_game_cum_xG(game = 2023030417, model = NULL)
```

Arguments

<code>game</code>	integer ID (e.g., 2025020275); see games() for reference
<code>model</code>	deprecated legacy model selector; ignored
<code>save</code>	logical only FALSE for tests

Value

NULL

Examples

```
# May take >5s, so skip.
x_game_cumulative_expected_goals(
  game = 2023030417,
  save = FALSE
)
```

x_game_shot_locations *Save an X (Twitter) share-able shot-location plot for a game*

Description

x_game_shot_locations() saves an X share-able shot-location plot for a given game.

Usage

```
x_game_shot_locations(  
  game = 2023030417,  
  team = "home",  
  model = NULL,  
  save = TRUE  
)
```

```
x_game_shot_locs(game = 2023030417, team = "home", model = NULL)
```

Arguments

game	integer ID (e.g., 2025020275); see games() for reference
team	character of 'h'/'home' or 'a'/'away'
model	deprecated legacy model selector; ignored
save	logical only FALSE for tests

Value

NULL

Examples

```
# May take >5s, so skip.  
x_game_shot_locations(  
  game = 2023030417,  
  team = 'H',  
  save = FALSE  
)
```

Index

add_deltas, 6
add_goalie_biometrics, 7
add_shift_times, 8
add_shift_times(), 30, 125
add_shooter_biometrics, 8
attendance, 9
attendance(), 33
award_winners, 9
award_winners(), 34
awards, 10
awards(), 34

boxscore, 10
boxscore(), 45
bracket, 11
bracket(), 34, 54, 90

calculate_expected_goals, 11
calculate_xG
 (calculate_expected_goals), 11
coach_career_statistics, 12
coach_career_statistics(), 38
coach_career_stats
 (coach_career_statistics), 12
coach_franchise_statistics, 13
coach_franchise_stats
 (coach_franchise_statistics),
 13
coaches, 13
coaches(), 38
combine_reports, 14
contracts, 14
countries, 15
countries(), 28, 35, 50

draft_picks, 15
draft_picks(), 35
draft_prospects, 16
draft_rankings, 16
draft_rankings(), 36

draft_tracker, 17
draft_tracker(), 36
drafts, 17
drafts(), 16, 36, 37
draw_NHL_rink, 18

espn_futures, 18
espn_futures(), 41
espn_game_odds, 19
espn_game_odds(), 39
espn_game_summary, 19
espn_game_summary(), 39
espn_games, 20
espn_games(), 19, 21, 39–41
espn_injuries, 20
espn_injuries(), 41
espn_pbp (espn_play_by_play), 21
espn_play_by_play, 21
espn_play_by_play(), 40
espn_player_summary, 21
espn_player_summary(), 37
espn_players, 22
espn_players(), 21, 37
espn_team_summary, 22
espn_team_summary(), 42
espn_teams, 23
espn_teams(), 22, 42
espn_transactions, 23
espn_transactions(), 42
expansion_draft_picks, 24
expansion_drafts, 24

franchise_playoff_situational_results,
 25
franchise_season_statistics, 25
franchise_season_statistics(), 43
franchise_season_stats
 (franchise_season_statistics),
 25
franchise_statistics, 26

- franchise_stats (franchise_statistics), 26
- franchise_team_statistics, 26
- franchise_team_statistics(), 43
- franchise_team_stats
 - (franchise_team_statistics), 26
- franchise_versus_franchise, 27
- franchise_versus_franchise(), 44
- franchise_vs_franchise
 - (franchise_versus_franchise), 27
- franchises, 27
- franchises(), 44

- game_odds, 28
- game_odds(), 50
- game_rosters, 28
- game_type_now, 29
- game_type_now(), 53
- games, 29
- games(), 10, 28, 30–32, 45–47, 55, 63, 77, 85, 91–93, 125–129
- gc_pbp (gc_play_by_play), 30
- gc_pbp_raw (gc_play_by_play_raw), 30
- gc_pbps (gc_play_by_plays), 31
- gc_pbps_raw (gc_play_by_plays_raw), 32
- gc_play_by_play, 30
- gc_play_by_play(), 7–9, 12, 30, 47, 85
- gc_play_by_play_raw, 30
- gc_play_by_plays, 31
- gc_play_by_plays(), 7–9, 12
- gc_play_by_plays_raw, 32
- gc_summary, 32
- gc_summary(), 40, 45
- general_managers, 33
- get_attendance, 33
- get_award_winners, 34
- get_awards, 34
- get_bracket, 34
- get_configuration, 35
- get_countries, 35
- get_draft_picks, 35
- get_draft_rankings, 36
- get_draft_tracker, 36
- get_drafts, 37
- get_espn_athlete, 37
- get_espn_athletes, 37
- get_espn_coach, 38
- get_espn_coach_career, 38
- get_espn_coaches, 38
- get_espn_event, 39
- get_espn_event_odds, 39
- get_espn_event_officials, 40
- get_espn_event_play_by_play, 40
- get_espn_event_stars, 40
- get_espn_events, 41
- get_espn_futures, 41
- get_espn_injuries, 41
- get_espn_team, 42
- get_espn_teams, 42
- get_espn_transactions, 42
- get_franchise_season_by_season, 43
- get_franchise_team_totals, 43
- get_franchise_vs_franchise, 44
- get_franchises, 44
- get_game_boxscore, 45
- get_game_landing, 45
- get_game_story, 46
- get_games, 46
- get_gc_play_by_play, 47
- get_glossary, 47
- get_goalie_leaders, 48
- get_goalie_milestones, 48
- get_goalie_statistics, 49
- get_goalies, 49
- get_officials, 49
- get_partner_odds, 50
- get_player_game_log, 50
- get_player_landing, 51
- get_players, 51
- get_schedule, 52
- get_scoreboards, 52
- get_scores, 53
- get_season_now, 53
- get_seasons, 53
- get_series, 54
- get_series_schedule, 54
- get_shift_charts, 54
- get_skater_leaders, 55
- get_skater_milestones, 55
- get_skater_statistics, 56
- get_skaters, 56
- get_spotlight_players, 56
- get_standings, 57
- get_standings_information, 57
- get_streams, 58
- get_team_prospects, 58

- get_team_roster, 59
- get_team_roster_statistics, 59
- get_team_schedule, 60
- get_team_scoreboard, 60
- get_team_seasons, 61
- get_team_statistics, 61
- get_teams, 61
- get_tv_schedule, 62
- get_venues, 62
- get_wsc_play_by_play, 63
- glossary, 63
- glossary(), 47
- gms (general_managers), 33
- goalie_edge_5_vs_5
 - (goalie_edge_five_versus_five), 64
- goalie_edge_five_versus_five, 64
- goalie_edge_leaders, 65
- goalie_edge_save_percentage, 66
- goalie_edge_seasons, 67
- goalie_edge_seasons(), 64–66, 68
- goalie_edge_shot_location, 67
- goalie_edge_summary, 68
- goalie_game_report, 69
- goalie_game_report(), 49
- goalie_game_scoring, 70
- goalie_game_statistics, 70
- goalie_game_stats
 - (goalie_game_statistics), 70
- goalie_leaders, 71
- goalie_leaders(), 48
- goalie_milestones, 72
- goalie_milestones(), 48
- goalie_regular_statistics, 72
- goalie_regular_stats
 - (goalie_regular_statistics), 72
- goalie_report_configs
 - (goalie_report_configurations), 73
- goalie_report_configurations, 73
- goalie_report_configurations(), 35, 69, 74
- goalie_scoring, 73
- goalie_season_report, 74
- goalie_season_report(), 49
- goalie_season_statistics, 75
- goalie_season_stats
 - (goalie_season_statistics), 75
- goalie_series_statistics, 75
- goalie_series_stats
 - (goalie_series_statistics), 75
- goalie_statistics, 76
- goalie_stats (goalie_statistics), 76
- graphics::points(), 18
- ig_game_cum_xG
 - (ig_game_cumulative_expected_goals), 76
- ig_game_cumulative_expected_goals, 76
- ig_game_shot_locations, 77
- ig_game_shot_locs
 - (ig_game_shot_locations), 77
- location, 78
- lottery_odds, 78
- nhlscraper, 79
- nhlscraper-package (nhlscraper), 79
- officials, 80
- officials(), 49
- penalty_shots, 81
- penalty_shots(), 85
- ping, 81
- player_game_log, 82
- player_game_log(), 50
- player_seasons, 82
- player_summary, 83
- player_summary(), 51
- players, 84
- players(), 49–51, 56, 82, 83, 95–100
- playoff_season_statistics, 84
- playoff_season_stats
 - (playoff_season_statistics), 84
- ps, 85
- pss (penalty_shots), 81
- replay, 85
- replays, 86
- roster, 86
- roster(), 59
- roster_statistics, 87
- roster_statistics(), 59
- roster_stats (roster_statistics), 87
- schedule, 88
- schedule(), 52

- scores, 88
- scores(), 52, 53
- season_now, 89
- season_now(), 53
- seasons, 89
- seasons(), 11, 18, 20, 31, 32, 34, 48, 50, 52–55, 57, 59, 60, 62, 69, 71, 74, 82, 86–90, 92, 93, 101, 102, 105, 108, 117, 118, 120, 121, 123, 124, 126, 127
- series, 90
- series(), 54, 90
- series_schedule, 90
- series_schedule(), 54
- shift_chart, 91
- shift_chart(), 8, 30, 54, 93, 125
- shift_chart_summaries, 92
- shift_chart_summary, 92
- shift_charts, 93
- shift_charts(), 8, 30, 125
- shifts, 93
- skater_edge_leaders, 94
- skater_edge_seasons, 94
- skater_edge_seasons(), 94–100
- skater_edge_shot_location, 95
- skater_edge_shot_speed, 96
- skater_edge_skating_distance, 97
- skater_edge_skating_speed, 98
- skater_edge_summary, 99
- skater_edge_zone_time, 100
- skater_game_report, 101
- skater_game_report(), 56
- skater_leaders, 102
- skater_leaders(), 55
- skater_milestones, 102
- skater_milestones(), 55
- skater_playoff_statistics, 103
- skater_playoff_stats
 - (skater_playoff_statistics), 103
- skater_regular_statistics, 104
- skater_regular_stats
 - (skater_regular_statistics), 104
- skater_report_configs
 - (skater_report_configurations), 104
- skater_report_configurations, 104
- skater_report_configurations(), 35, 101, 105
- skater_season_report, 105
- skater_season_report(), 56
- skater_season_statistics, 106
- skater_season_stats
 - (skater_season_statistics), 106
- skater_series_statistics, 106
- skater_series_stats
 - (skater_series_statistics), 106
- skater_statistics, 107
- skater_stats(skater_statistics), 107
- spotlight_players, 107
- spotlight_players(), 56
- standings, 108
- standings(), 57
- standings_rules, 108
- standings_rules(), 57
- streams, 109
- streams(), 58
- team_edge_leaders, 109
- team_edge_seasons, 110
- team_edge_seasons(), 109, 111–116
- team_edge_shot_location, 110
- team_edge_shot_speed, 111
- team_edge_skating_distance, 112
- team_edge_skating_speed, 113
- team_edge_summary, 114
- team_edge_zone_time, 115
- team_game_report, 116
- team_game_report(), 61
- team_logos, 117
- team_month_schedule, 118
- team_prospects, 118
- team_prospects(), 58
- team_report_configs
 - (team_report_configurations), 119
- team_report_configurations, 119
- team_report_configurations(), 35, 117, 120
- team_season_report, 120
- team_season_report(), 61
- team_season_schedule, 121
- team_season_schedule(), 60
- team_season_statistics, 121
- team_season_stats
 - (team_season_statistics), 121

team_seasons, [122](#)
team_seasons(), [61](#)
team_week_schedule, [123](#)
teams, [123](#)
teams(), [58–61](#), [86](#), [87](#), [111–116](#), [118](#), [119](#),
[121–123](#)
tv_schedule, [124](#)
tv_schedule(), [62](#)

venues, [124](#)
venues(), [62](#)

wsc_pbp (wsc_play_by_play), [125](#)
wsc_pbp_raw (wsc_play_by_play_raw), [126](#)
wsc_pbps (wsc_play_by_plays), [126](#)
wsc_pbps_raw (wsc_play_by_plays_raw),
[127](#)
wsc_play_by_play, [125](#)
wsc_play_by_play(), [7–9](#), [12](#), [63](#), [85](#), [126](#)
wsc_play_by_play_raw, [126](#)
wsc_play_by_plays, [126](#)
wsc_play_by_plays(), [7–9](#), [12](#)
wsc_play_by_plays_raw, [127](#)
wsc_summary, [127](#)
wsc_summary(), [40](#), [46](#)

x_game_cum_xG
 (x_game_cumulative_expected_goals),
 [128](#)
x_game_cumulative_expected_goals, [128](#)
x_game_shot_locations, [129](#)
x_game_shot_locs
 (x_game_shot_locations), [129](#)