

# Package ‘oddsapiR’

May 9, 2026

**Title** Access Live Sports Odds from the Odds API

**Version** 0.0.3

**Description** A utility to quickly obtain clean and tidy sports odds from The Odds API <<https://the-odds-api.com>>.

**License** MIT + file LICENSE

**URL** <https://oddsapiR.sportsdataverse.org/> (docs),  
<https://github.com/sportsdataverse/oddsapiR> (repo)

**BugReports** <https://github.com/sportsdataverse/oddsapiR/issues>

**SystemRequirements** pandoc (>= 1.12.3), pandoc-citeproc

**Depends** R (>= 4.0.0)

**Imports** cli (>= 3.4.1), data.table (>= 1.14.0), dplyr (>= 1.0.10), glue, httr (>= 0.5), janitor, jsonlite, magrittr, rlang (>= 1.0.4), rvest (>= 1.0.0), tidyr (>= 1.0.0)

**Suggests** crayon (>= 1.3.4), curl, DBI, ggplot2, ggrepel, gt, knitr, progressr (>= 0.6.0), qs (>= 0.25.1), Rcpp (>= 1.0.7), RcppParallel (>= 5.1.4), rmarkdown, RSQLite, stats, stringi, stringr (>= 1.5.0), testthat, tibble (>= 3.0), tidyselect (>= 1.2.0), usethis (>= 1.6.0), xml2 (>= 1.3)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.2.3

**NeedsCompilation** no

**Author** Saiem Gilani [aut, cre]

**Maintainer** Saiem Gilani <[saiem.gilani@gmail.com](mailto:saiem.gilani@gmail.com)>

**Repository** CRAN

**Date/Publication** 2023-03-19 18:50:02 UTC

## Contents

csv_from_url	2
progressively	2
rds_from_url	3
register_toa	3
toa_event_odds	4
toa_requests	7
toa_sports	8
toa_sports_keys	9
toa_sports_odds	9
toa_sports_odds_history	11
toa_sports_scores	13

<b>Index</b>	<b>15</b>
--------------	-----------

---

csv_from_url	<b>Load .csv / .csv.gz file from a remote connection</b>
--------------	--

---

### Description

This is a thin wrapper on `data.table::fread`

### Usage

```
csv_from_url(...)
```

### Arguments

... passed to `data.table::fread`

### Value

a dataframe as created by `data.table::fread()`

---

progressively	<b>Progressively</b>
---------------	----------------------

---

### Description

This function helps add progress-reporting to any function - given function `f()` and progressor `p()`, it will return a new function that calls `f()` and then (on-exiting) will call `p()` after every iteration.

### Usage

```
progressively(f, p = NULL)
```

**Arguments**

- f a function to add progressr functionality to.
- p a progressor function as created by `progressr::progressor()`

**Details**

This is inspired by purrr's `safely`, `quietly`, and `possibly` function decorators.

**Value**

a function that does the same as `f` but it calls `p()` after iteration.

---

rds\_from\_url                      **Load .rds file from a remote connection**

---

**Description**

**Load .rds file from a remote connection**

**Usage**

```
rds_from_url(url)
```

**Arguments**

- url a character url

**Value**

a dataframe as created by [readRDS\(\)](#)

---

register\_toa                      **Odds API Key Registration**

---

**Description**

Save your API Key as a system environment variable `ODDS_API_KEY`

**Usage**

- `toa_key()`
- `has_toa_key()`
- `check_toa_key()`

## Details

To get access to an API key, follow the instructions at <https://the-odds-api.com>

### Using the key:

You can save the key for consistent usage by adding `ODDS_API_KEY=XXXX-YOUR-API-KEY-HERE-XXXXX` to your `.Renviron` file (easily accessed via `usethis::edit_r_environ()`).

Run `usethis::edit_r_environ()`, a new script will pop open named `.Renviron`, **THEN** paste the following in the new script that pops up (**without** quotations)

```
ODDS_API_KEY = XXXX-YOUR-API-KEY-HERE-XXXXX
```

Save the script and restart your RStudio session, by clicking Session (in between Plots and Build) and click Restart R

(there also exists the shortcut `Ctrl + Shift + F10` to restart your session).

If set correctly, from then on you should be able to use any of the `toa_` functions without any other changes.

### For less consistent usage:

At the beginning of every session or within an R environment, save your API key as the environment variable `ODDS_API_KEY` (**with** quotations) using a command like the following.

```
Sys.setenv(ODDS_API_KEY = "XXXX-YOUR-API-KEY-HERE-XXXXX")
```

## Value

Called as a side-effect to ensure that a user has an API key stored in their environment before making a call to the Odds API service.

---

toa_event_odds	<b>Find odds for the sports which are accessible through the Odds API</b>
----------------	---

---

## Description

**Get the odds for the sports which the Odds API provides coverage**

```
try(toa_sports_odds(sport_key = 'baseball_mlb',
                    regions = 'us',
                    markets = 'spreads',
                    odds_format = 'decimal',
                    date_format = 'iso'))
```

**Usage**

```

toa_event_odds(
  sport_key,
  event_id,
  regions = "us",
  markets = "spreads",
  odds_format = "decimal",
  date_format = "iso",
  bookmakers = NULL
)

```

**Arguments**

sport_key	The sport_key to look up odds for. See toa_sports() for a full lookup of sport_key values.
event_id	The event_id to look up odds for. See toa_sports_odds() for a full lookup of event_id values.
regions	The region to pull odds from. Options include: <ul style="list-style-type: none"> <li>• us</li> <li>• uk</li> <li>• us</li> <li>• eu</li> <li>• au Multiple can be specified if comma delimited.</li> </ul>
markets	The type of odds to return. Multiple can be specified if comma delimited. Options include: <ul style="list-style-type: none"> <li>• h2h</li> <li>• spreads</li> <li>• totals</li> <li>• outrights</li> <li>• h2h_lay</li> <li>• outrights_lay</li> <li>• alternate_spreads</li> <li>• alternate_totals</li> <li>• btts</li> <li>• draw_no_bet</li> <li>• h2h_3_way</li> </ul> NFL Player Props: <ul style="list-style-type: none"> <li>• player_pass_tds</li> <li>• player_pass_yds</li> <li>• player_pass_completions</li> <li>• player_pass_attempts</li> <li>• player_pass_interceptions</li> <li>• player_pass_longest_completion</li> </ul>

- player\_rush\_yds
- player\_rush\_attempts
- player\_rush\_longest
- player\_receptions
- player\_reception\_yds
- player\_reception\_longest

#### NBA + NCAAAB Player Props:

- player\_points
- player\_rebounds
- player\_assists
- player\_threes
- player\_double\_double
- player\_blocks
- player\_steals
- player\_turnovers
- player\_points\_rebounds\_assists
- player\_points\_rebounds
- player\_points\_assists
- player\_rebounds\_assists

#### NHL Player Props:

- player\_points
- player\_power\_play\_points
- player\_assists
- player\_blocked\_shots
- player\_shots\_on\_goal

#### Player Props Documentation

odds_format	The format in which to return odds. Options include: <ul style="list-style-type: none"> <li>• decimal</li> <li>• american</li> </ul>
date_format	Date format. Options include: <ul style="list-style-type: none"> <li>• iso</li> <li>• unix</li> </ul>
bookmakers	Comma-separated list of bookmakers to be returned. If both bookmakers and regions are specified, bookmakers takes precedence. Bookmakers can be from any region. Every group of 10 bookmakers counts as 1 request. For example for a single market, specifying up to 10 bookmakers counts as 1 request. Specifying between 11 and 20 bookmakers counts as 2 requests

**Details**

```
try(toa_event_odds(sport_key = 'basketball_ncaab',
  event_id = '48db9c3293a52baab881d95d38f37a98',
  regions = 'us',
  markets = 'player_points',
  odds_format = 'decimal',
  date_format = 'iso'))
```

**Value**

Sports for which The Odds API provides betting information for as a tibble:

col_name	types
id	character
sport_key	character
sport_title	character
commence_time	character
home_team	character
away_team	character
bookmaker_key	character
bookmaker	character
last_update	character
market_key	character
outcomes_name	character
outcomes_price	numeric
outcomes_point	numeric

---

 toa\_requests

**Find out your usage and remaining calls for your key from The Odds API**

---

**Description**

**Get your usage and remaining calls for your key from The Odds API**

```
toa_requests()
```

**Usage**

```
toa_requests()
```

**Value**

Returns a tibble of The Odds API key usage with the following columns:

col_name	types
requests_remaining	integer
requests_used	integer

---

 toa\_sports

**Find sports for which odds are accessible through the Odds API**


---

### Description

**Get the Sports for which the Odds API provides coverage**

```
toa_sports(all_sports=TRUE)
```

### Usage

```
toa_sports(all_sports = TRUE)
```

### Arguments

`all_sports` (*Logical* required): If true, returns all sports and if false, returns only active sports. Defaults to true.

### Value

Sports for which The Odds API provides betting information for as a tibble:

col_name	types
key	character
group	character
title	character
description	character
active	logical
has_outrights	logical

### Examples

```
try(toa_sports(all_sports = TRUE))
```

---

toa_sports_keys	<b>Sports for which odds are accessible through the Odds API</b>
-----------------	--

---

**Description**

A data set mapping Sports Events/League names to keys for end-user ease.

**Usage**

```
data(toa_sports_keys)
```

**Format**

A data frame with 5 variables:

key - Sport key group - Sport group (non-league description) title - Sport title description - Sport description has\_outrights - Whether the sport or event has outright victories.

---

toa_sports_odds	<b>Find odds for the sports which are accessible through the Odds API</b>
-----------------	---

---

**Description**

**Get the odds for the sports which the Odds API provides coverage**

```
try(toa_sports_odds(sport_key = 'baseball_mlb',  
                    regions = 'us',  
                    markets = 'spreads',  
                    odds_format = 'decimal',  
                    date_format = 'iso'))
```

**Usage**

```
toa_sports_odds(  
  sport_key,  
  regions = "us",  
  markets = "spreads",  
  odds_format = "decimal",  
  date_format = "iso"  
)
```

**Arguments**

sport_key	The sport_key to look up odds for. See toa_sports() for a full lookup of sport_key values.
regions	The region to pull odds from. Options include: <ul style="list-style-type: none"> <li>• us</li> <li>• uk</li> <li>• us</li> <li>• eu</li> <li>• au Multiple can be specified if comma delimited.</li> </ul>
markets	The type of odds to return. Multiple can be specified if comma delimited. Options include: <ul style="list-style-type: none"> <li>• h2h</li> <li>• spreads</li> <li>• totals</li> </ul>
odds_format	The format in which to return odds. Options include: <ul style="list-style-type: none"> <li>• decimal</li> <li>• american</li> </ul>
date_format	Date format. Options include: <ul style="list-style-type: none"> <li>• iso</li> <li>• unix</li> </ul>

**Value**

Sports for which The Odds API provides betting information for as a tibble:

col_name	types
id	character
sport_key	character
sport_title	character
commence_time	character
home_team	character
away_team	character
bookmaker_key	character
bookmaker	character
last_update	character
market_key	character
outcomes_name	character
outcomes_price	numeric
outcomes_point	numeric

**Examples**

```
try(toa_sports_odds(sport_key = 'basketball_ncaab',
  regions = 'us',
```

```
markets = 'spreads',
odds_format = 'decimal',
date_format = 'iso'))
```

---

toa\_sports\_odds\_history

**Find odds history for the sports which are accessible through the Odds API**

---

### Description

**Get the odds history for the sports which the Odds API provides coverage**

```
try(toa_sports_odds(sport_key = 'baseball_mlb',
                    regions = 'us',
                    markets = 'spreads',
                    odds_format = 'decimal',
                    date_format = 'iso'))
```

### Usage

```
toa_sports_odds_history(
  sport_key,
  event_ids,
  date,
  regions = "us",
  markets = "spreads",
  odds_format = "decimal",
  date_format = "iso",
  bookmakers = NULL
)
```

### Arguments

sport_key	The sport_key to look up odds for. See toa_sports() for a full lookup of sport_key values.
event_ids	The event_id's to look up odds for. See toa_sports_odds() for a full lookup of event_id values.
date	The timestamp of the data snapshot to be returned, specified in ISO8601 format. The historical odds API will return the closest snapshot equal to or earlier than the provided date parameter Example : 2022-10-10T12:15:00Z
regions	The region to pull odds from. Options include: <ul style="list-style-type: none"><li>• us</li><li>• uk</li></ul>

	<ul style="list-style-type: none"> <li>• us</li> <li>• eu</li> <li>• au Multiple can be specified if comma delimited.</li> </ul>
markets	The type of odds to return. Multiple can be specified if comma delimited. Options include: <ul style="list-style-type: none"> <li>• h2h</li> <li>• spreads</li> <li>• totals</li> </ul>
odds_format	The format in which to return odds. Options include: <ul style="list-style-type: none"> <li>• decimal</li> <li>• american</li> </ul>
date_format	Date format. Options include: <ul style="list-style-type: none"> <li>• iso</li> <li>• unix</li> </ul>
bookmakers	Comma-separated list of bookmakers to be returned. If both bookmakers and regions are specified, bookmakers takes precedence. Bookmakers can be from any region. Every group of 10 bookmakers counts as 1 request. For example for a single market, specifying up to 10 bookmakers counts as 1 request. Specifying between 11 and 20 bookmakers counts as 2 requests

### Details

```
try(toa_sports_odds_history(sport_key = 'basketball_ncaab',
  event_ids = '48db9c3293a52baab881d95d38f37a98',
  date = '2023-03-18T12:15:00Z',
  regions = 'us',
  markets = 'spreads',
  odds_format = 'decimal',
  date_format = 'iso',
  bookmakers = NULL))
```

### Value

Sports for which The Odds API provides betting information for as a tibble:

col_name	types
id	character
sport_key	character
sport_title	character
commence_time	character
home_team	character
away_team	character
bookmaker_key	character
bookmaker	character
last_update	character
market_key	character

outcomes_name	character
outcomes_price	numeric
outcomes_point	numeric

---

toa_sports_scores	<b>Find scores for the sports which are accessible through the Odds API</b>
-------------------	---

---

### Description

**Get the scores for the sports which the Odds API provides coverage**

```
try(toa_sports_scores(sport_key = 'baseball_mlb',
                     days_from = NULL,
                     date_format = 'iso'))
```

### Usage

```
toa_sports_scores(sport_key, days_from = 1, date_format = "iso")
```

### Arguments

sport_key	( <i>string</i> , required): The sport_key to look up odds for. See toa_sports() for a full lookup of sport_key values.
days_from	( <i>integer</i> , optional): Integer from 1 to 3. Defaults to 1.
date_format	( <i>string</i> , optional): Date format. Options include: <ul style="list-style-type: none"> <li>• iso</li> <li>• unix</li> </ul>

### Value

Sports scores which The Odds API provides scores information for as a tibble:

col_name	types
id	character
sport_key	character
sport_title	character
commence_time	character
completed	logical
home_team	character
away_team	character
scores	logical
last_update	logical

**Examples**

```
try(toa_sports_scores(sport_key = 'baseball_mlb',  
                     days_from = NULL,  
                     date_format = 'iso'))
```

# Index

## \* **Betting**

- toa\_event\_odds, 4
- toa\_requests, 7
- toa\_sports, 8
- toa\_sports\_odds, 9
- toa\_sports\_odds\_history, 11
- toa\_sports\_scores, 13

## \* **Internal**

- csv\_from\_url, 2
- progressively, 2
- rds\_from\_url, 3

## \* **Lines**

- toa\_event\_odds, 4
- toa\_requests, 7
- toa\_sports, 8
- toa\_sports\_odds, 9
- toa\_sports\_odds\_history, 11
- toa\_sports\_scores, 13

## \* **datasets**

- toa\_sports\_keys, 9

check\_toa\_key(register\_toa), 3

csv\_from\_url, 2

data.table::fread(), 2

has\_toa\_key(register\_toa), 3

progressively, 2

rds\_from\_url, 3

readRDS(), 3

register\_toa, 3

toa\_event\_odds, 4

toa\_key(register\_toa), 3

toa\_requests, 7

toa\_sports, 8

toa\_sports\_keys, 9

toa\_sports\_odds, 9

toa\_sports\_odds\_history, 11

toa\_sports\_scores, 13