Package ‘oddsapiio’

November 13, 2026

Title Client for the 'Odds-API.io' Sports Betting Odds API

Version 0.2.1

Description Query live and upcoming sports betting odds from the 'Odds-API.io' REST API <https://docs.odds-api.io>. Covers sports, markets, bookmakers, leagues, participants, events, per-event and multi-event odds, odds deltas, odds movements, odds summaries, value bets, arbitrage bets, dropping odds, historical events, odds and closing lines, bet settlement and grading, outright (futures) odds and results, and bookmaker id mappings across 450+ bookmakers, returned as tidy data frames. An API key from a paid plan is required.

License MIT + file LICENSE

URL https://odds-api.io, https://docs.odds-api.io, https://github.com/odds-api-io/odds-api-r

BugReports https://github.com/odds-api-io/odds-api-r/issues

Depends R (>= 4.0.0)

Encoding UTF-8

Imports httr2 (>= 1.1.0), jsonlite, stats, utils

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Markus Bjermeland [aut, cre]

Maintainer Markus Bjermeland <markus@odds-api.io>

Repository CRAN

Date/Publication 2026-11-13 09:00:02 UTC

Contents

oa_arbitrage_bets . . . 2
oa_bookmakers . . . 4
oa_closing_lines . . . 4
oa_dropping_odds . . . 6
oa_event . . . 7
oa_events . . . 8
oa_grade . . . 9
oa_historical_events . . . 10
oa_historical_odds . . . 11
oa_historical_search . . . 12
oa_leagues . . . 13
oa_live_events . . . 13
oa_mappings . . . 14
oa_markets . . . 15
oa_multi_odds . . . 16
oa_odds . . . 17
oa_odds_movements . . . 18
oa_odds_summary . . . 19
oa_odds_updated . . . 20
oa_outrights . . . 21
oa_outright_odds . . . 22
oa_outright_results . . . 22
oa_participants . . . 23
oa_search_events . . . 24
oa_selected_bookmakers . . . 25
oa_settlements . . . 25
oa_sports . . . 26
oa_value_bets . . . 27

Index **28**

`oa_arbitrage_bets` *Get arbitrage bets*

Description

Returns pre-match price gaps across the given bookmakers where the best prices for every outcome imply a total probability under 100%, with the legs and a stake split for each. Prices move quickly and bookmakers can void or limit bets, so check every leg before acting on one.

Usage

```
oa_arbitrage_bets(
  bookmakers,
  sport = NULL,
  league = NULL,
  markets = NULL,
  min_profit = NULL,
  all_combinations = FALSE,
  limit = NULL,
  include_event_details = FALSE,
  api_key = NULL
)
```

Arguments

bookmakers	Character vector of bookmaker names to compare. Required by the API.
sport	Optional sport slug, e.g. "football".
league	Optional league slug; requires sport.
markets	Optional market names to keep (max 40), e.g. c("ML", "Totals").
min_profit	Optional minimum profitMargin in percent (0 to 100), e.g. 1.5.
all_combinations	If TRUE, return every bookmaker combination per event, market and line instead of only the best one.
limit	Optional maximum number of results (default 50, max 500).
include_event_details	If TRUE, event information is included.
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per arbitrage opportunity. The legs and optimalStakes columns are list-columns of data frames.

Examples

```
## Not run:
arbs <- oa_arbitrage_bets(c("Bet365", "SingBet"), sport = "football",
                          min_profit = 1, include_event_details = TRUE)
arbs[order(-arbs$profitMargin), ]

## End(Not run)
```

oa_bookmakers *List available bookmakers*

Description

Returns the bookmakers covered by 'Odds-API.io'. This endpoint does not require an API key. Bookmaker coverage by sport and market is listed at https://odds-api.io/sportsbooks#coverage.

Usage

```
oa_bookmakers(api_key = NULL)
```

Arguments

api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per bookmaker, with columns name, active, createdAt, status ("ok", "degraded" or "down") and reason (NA when the status is ok, otherwise "maintenance" or "unavailable").

Examples

```
## Not run:
bookmakers <- oa_bookmakers()
subset(bookmakers, status != "ok")

## End(Not run)
```

oa_closing_lines *Get closing lines for settled events*

Description

Returns the closing odds of every settled event across up to 10 leagues in one request, for closing line value and model work. The window can be up to 366 days, so a full season fits in one sweep. Each response holds at most 100 events; use limit and skip to paginate. Requires a paid plan.

Usage

```
oa_closing_lines(
  sport,
  leagues,
  from,
  to,
  markets,
  bookmakers,
  limit = NULL,
  skip = NULL,
  api_key = NULL
)
```

Arguments

sport	Sport slug, e.g. "football".
leagues	Character vector of up to 10 league slugs.
from, to	Window bounds (RFC 3339 strings or POSIXct/Date), at most 366 days apart.
markets	Character vector of market names, e.g. c("ML", "Totals"). Required.
bookmakers	Character vector of bookmaker names (max 30). Required.
limit	Optional number of events per response (max 100).
skip	Optional pagination offset.
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A tidy data frame in the same format as oa_odds(), one row per event, bookmaker, market and line.

Examples

```
## Not run:
cl <- oa_closing_lines("football", c("england-premier-league", "spain-laliga"),
                       from = "2026-08-01T00:00:00Z",
                       to = "2026-08-31T23:59:59Z",
                       markets = "ML", bookmakers = c("Bet365", "SingBet"))

## End(Not run)
```

`oa_dropping_odds` *Get dropping odds*

Description

Returns prices at sharp bookmakers that have dropped, sorted by the size of the drop, the latest movement or kickoff. Requires a paid plan.

Usage

```
oa_dropping_odds(
  sport = NULL,
  league = NULL,
  leagues = NULL,
  markets = NULL,
  time_window = NULL,
  sort = NULL,
  min_drop = NULL,
  limit = NULL,
  page = NULL,
  include_event_details = FALSE,
  api_key = NULL
)
```

Arguments

`sport`	Optional sport slug, e.g. `"football"`.
`league`	Optional league slug; requires `sport`. Cannot be combined with `leagues`.
`leagues`	Optional character vector of league slugs.
`markets`	Optional market names, e.g. `c("Spread", "Totals")`.
`time_window`	Window for the drop: `"opening"` (the API default), `"12h"`, `"24h"` or `"48h"`.
`sort`	`"drop"` (the API default), `"recent"` or `"kickoff"`.
`min_drop`	Optional minimum drop in percent.
`limit`	Optional results per page (default 50, max 200).
`page`	Optional page number, starting at 1.
`include_event_details`	If `TRUE`, event information is included.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A data frame with one row per dropping price. Attributes `total_count`, `page` and `per_page` describe the pagination.

Examples

```
## Not run:
drops <- oa_dropping_odds("football", min_drop = 10,
                          include_event_details = TRUE)
attr(drops, "total_count")

## End(Not run)
```

`oa_event` *Get one event*

Description

Get one event

Usage

```
oa_event(event_id, api_key = NULL)
```

Arguments

`event_id`	Event id, from `oa_events()`.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A one-row data frame describing the event.

Examples

```
## Not run:
oa_event(123456)

## End(Not run)
```

`oa_events` *List events for a sport*

Description

Returns upcoming, live, settled or cancelled events. When `to` is omitted the API returns events within the next 14 days, hard-capped at 5000 rows per response; use `limit` and `skip` to paginate.

Usage

```
oa_events(
  sport,
  league = NULL,
  status = NULL,
  from = NULL,
  to = NULL,
  limit = NULL,
  skip = NULL,
  bookmaker = NULL,
  participant_id = NULL,
  api_key = NULL
)
```

Arguments

`sport`	Sport slug, e.g. `"football"`. See `oa_sports()`.
`league`	Optional league slug, e.g. `"england-premier-league"`.
`status`	Optional statuses to include: any of `"pending"`, `"live"`, `"settled"`, `"cancelled"`. A vector is collapsed to a comma-separated list.
`from, to`	Optional RFC 3339 date-times bounding the event window, e.g. `"2026-09-01T00:00:00Z"`, or `POSIXct`/`Date` values.
`limit`	Optional maximum number of events (capped at 5000).
`skip`	Optional pagination offset, used together with `limit`.
`bookmaker`	Optional bookmaker name; only events with odds from that bookmaker are returned.
`participant_id`	Optional team or participant id to filter by.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Details

Scores are flattened into columns such as `scores.home`, `scores.away`, `scores.periods.p1.home` and `scores.periods.ft.home`. Period keys are `p1` to `pN` (each period on its own), `ft` (end of regulation), `ot` (after overtime) and `ap` (after penalties).

Value

A data frame with one row per event.

Examples

```
## Not run:
events <- oa_events("football", league = "england-premier-league")
live <- oa_events("football", status = c("pending", "live"))

## End(Not run)
```

`oa_grade` *Grade bet selections*

Description

Grades up to 50 bet selections against final results. One bad selection never fails the batch: it comes back as `ungradeable` with a reason. Requires a paid plan.

Usage

```
oa_grade(selections, api_key = NULL)
```

Arguments

`selections`	A data frame (or a list of lists) with one row per selection. Columns: `eventId` and `market` (required), plus `side`, `hdp`, `label`, `tieRule` and `ref` where needed. Use market names, lines and labels exactly as served by `oa_odds()`; handicaps are stated for the home side. `NA` values are left out of the request.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A data frame with one row per selection, in the order sent, with columns `ref`, `eventId`, `result` (`"won"`, `"lost"`, `"void"`, `"half_won"`, `"half_lost"`, `"pending"` or `"ungradeable"`), `reason`, `final` and, for score markets, `basis.*` columns.

Examples

```
## Not run:
bets <- data.frame(
  ref = c("bet-1", "bet-2"),
  eventId = c(74919888, 74919888),
  market = c("ML", "Spread"),
  side = c("home", "away"),
  hdp = c(NA, -0.75)
)
```

```
oa_grade(bets)

## End(Not run)
```

`oa_historical_events` *List historical events*

Description

Returns finished events for one league in a date range of at most 31 days. Historical data is available from December 2025.

Usage

```
oa_historical_events(sport, league, from, to, api_key = NULL)
```

Arguments

`sport`	Sport slug, e.g. `"football"`.
`league`	League slug, e.g. `"england-premier-league"`. Required.
`from, to`	Start and end of the window, as RFC 3339 strings such as `"2026-09-01T00:00:00Z"` or as `POSIXct`/`Date` values. At most 31 days apart.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A data frame with one row per event, in the same format as `oa_events()`, including final scores (`scores.home`, `scores.away` and period columns such as `scores.periods.p1.home` and `scores.periods.ft.home`).

Examples

```
## Not run:
past <- oa_historical_events("football", "england-premier-league",
                             from = "2026-09-01T00:00:00Z",
                             to = "2026-09-30T23:59:59Z")

## End(Not run)
```

`oa_historical_odds` *Get historical odds for a finished event*

Description

Returns the closing odds of a finished event.

Usage

```
oa_historical_odds(event_id, bookmakers, markets = NULL, api_key = NULL)
```

Arguments

`event_id`	Event id, from `oa_historical_events()`.
`bookmakers`	Character vector of bookmaker names (max 30), e.g. `c("Bet365", "SingBet")`. Required by the API.
`markets`	Optional character vector of market names to keep (max 40), e.g. `c("ML", "Spread", "Totals")`. Matching is case-insensitive but must be the full name; see `oa_markets()` or the market reference at `https://odds-api.io/markets`. All markets are returned when omitted.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A tidy data frame in the same format as `oa_odds()`.

Examples

```
## Not run:
past <- oa_historical_events("football", "england-premier-league",
                             "2026-09-01T00:00:00Z",
                             "2026-09-30T23:59:59Z")
oa_historical_odds(past$id[1], c("Bet365", "SingBet"), markets = "ML")

## End(Not run)
```

`oa_historical_search` *Search historical events*

Description

Searches finished events by team name across all leagues. Returns up to 100 events, newest first, from a window of at most 31 days (the API defaults to the last 31 days).

Usage

```
oa_historical_search(
  query,
  sport = NULL,
  from = NULL,
  to = NULL,
  api_key = NULL
)
```

Arguments

`query`	Team name fragment, at least 3 characters.
`sport`	Optional sport slug.
`from`, `to`	Optional window bounds (RFC 3339 strings or `POSIXct`/`Date`), at most 31 days apart.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A data frame with one row per event.

Examples

```
## Not run:
oa_historical_search("Arsenal", sport = "football")

## End(Not run)
```

oa_leagues *List leagues for a sport*

Description

List leagues for a sport

Usage

```
oa_leagues(sport, all = FALSE, api_key = NULL)
```

Arguments

sport	Sport slug, e.g. "football". See oa_sports().
all	If TRUE, also include leagues without active events.
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per league (columns name, slug and eventsCount).

Examples

```
## Not run:
leagues <- oa_leagues("football")
head(leagues)

## End(Not run)
```

oa_live_events *List live events*

Description

Returns events that are in play now, with their current scores and, when available, the match clock (clock.minute, clock.period, clock.running, ...).

Usage

```
oa_live_events(sport = NULL, api_key = NULL)
```

Arguments

sport	Optional sport slug, e.g. "football".
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per live event.

Examples

```
## Not run:
live <- oa_live_events("football")
live[, c("id", "home", "away", "scores.home", "scores.away")]

## End(Not run)
```

`oa_mappings` *Get native id mappings*

Description

Links events and selections to a bookmaker's own event, market and outcome ids (for example Polymarket condition and token ids, or Kalshi tickers). Look up by `event_ids`, by `market_ids` (optionally with `outcome_id`), or leave all three out for a bulk sync of upcoming and live events, paged with `cursor`. Requires a paid plan, and the bookmaker must be in your selection.

Usage

```
oa_mappings(
  bookmaker,
  event_ids = NULL,
  market_ids = NULL,
  outcome_id = NULL,
  market = NULL,
  active = NULL,
  sport = NULL,
  league = NULL,
  limit = NULL,
  cursor = NULL,
  api_key = NULL
)
```

Arguments

`bookmaker`	Bookmaker name, e.g. `"Polymarket"` or `"Kalshi"`.
`event_ids`	Optional vector of up to 10 event ids.
`market_ids`	Optional vector of up to 10 bookmaker market ids.
`outcome_id`	Optional bookmaker outcome id.
`market`	Optional exact market name as served by `oa_odds()` (case-sensitive).
`active`	Optional `TRUE` for outcomes the bookmaker still lists, `FALSE` for withdrawn ones. Both when omitted.

sport, league	Optional bulk sync filters; league requires sport.
limit	Optional rows per page in bulk sync (1 to 5000).
cursor	Optional next_cursor attribute from the previous page.
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per mapping, including eventId, market, hdp, side, nativeEventId, marketId, outcomeId and active. Attribute next_cursor is NULL on the last page.

Examples

```
## Not run:
oa_mappings("Polymarket", event_ids = 71515842)

page <- oa_mappings("Kalshi", sport = "american-football", limit = 5000)
while (!is.null(attr(page, "next_cursor"))) {
  page <- oa_mappings("Kalshi", sport = "american-football", limit = 5000,
                      cursor = attr(page, "next_cursor"))
}

## End(Not run)
```

---

oa_markets *List markets*

Description

Returns the market catalogue: every market currently served, with how to parse it. Use it to look up exact market names for the markets argument of oa_odds() and related functions. Market names are, for example, "ML", "Spread", "Totals", "Both Teams To Score" and "Totals HT". This endpoint does not require an API key. The same catalogue is browsable per sport at https://odds-api.io/markets, for example https://odds-api.io/markets/football.

Usage

```
oa_markets(sport = NULL, api_key = NULL)
```

Arguments

sport	Optional sport slug, e.g. "football". When omitted, markets for all sports are returned.
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per market and columns `sport`, `name`, `shape` (`"moneyline"`, `"line"`, `"outcomes"`, `"yesno"` or `"labelled"`), `period`, `prematch` and `live`.

Examples

```
## Not run:
markets <- oa_markets("football")
subset(markets, live)

## End(Not run)
```

`oa_multi_odds` *Get odds for several events*

Description

Fetches odds for up to 10 events in one request (counted as one request against your rate limit).

Usage

```
oa_multi_odds(event_ids, bookmakers, markets = NULL, api_key = NULL)
```

Arguments

`event_ids`	Vector of up to 10 event ids.
`bookmakers`	Character vector of bookmaker names (max 30), e.g. `c("Bet365", "SingBet")`. Required by the API.
`markets`	Optional character vector of market names to keep (max 40), e.g. `c("ML", "Spread", "Totals")`. Matching is case-insensitive but must be the full name; see `oa_markets()` or the market reference at `https://odds-api.io/markets`. All markets are returned when omitted.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A tidy data frame in the same format as `oa_odds()`, with rows for every event.

Examples

```
## Not run:
events <- oa_events("football", league = "england-premier-league",
                    limit = 10)
odds <- oa_multi_odds(events$id, bookmakers = "Bet365", markets = "ML")

## End(Not run)
```

oa_odds *Get odds for an event*

Description

Fetches odds for one event from up to 30 bookmakers and returns them as a tidy data frame with one row per price line.

Usage

```
oa_odds(event_id, bookmakers, markets = NULL, api_key = NULL)
```

Arguments

event_id	Event id, from oa_events().
bookmakers	Character vector of bookmaker names (max 30), e.g. c("Bet365", "SingBet"). Required by the API.
markets	Optional character vector of market names to keep (max 40), e.g. c("ML", "Spread", "Totals"). Matching is case-insensitive but must be the full name; see oa_markets() or the market reference at https://odds-api.io/markets. All markets are returned when omitted.
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per odds line, with columns eventId, bookmaker, market, updatedAt and the price columns present in the response (such as home, draw, away, hdp, over, under). Prices are decimal odds as strings, as served by the API.

Examples

```
## Not run:
odds <- oa_odds(123456, bookmakers = c("Bet365", "SingBet"),
                markets = c("ML", "Totals"))
subset(odds, market == "ML")

## End(Not run)
```

`oa_odds_movements` *Get odds movements for a market*

Description

Returns the recorded pre-match price history of one market at one bookmaker. History is kept for a fixed set of bookmakers; the API returns a 404 that names the markets or lines with history when the request has none.

Usage

```
oa_odds_movements(
  event_id,
  bookmaker,
  market,
  market_line = NULL,
  api_key = NULL
)
```

Arguments

`event_id`	Event id.
`bookmaker`	Bookmaker name, case-insensitive, e.g. `"Bet365"`.
`market`	Market name, case-insensitive, e.g. `"ML"`, `"Spread"`, `"Totals"`.
`market_line`	Optional handicap line, e.g. `2.5`. Leave it out for the main line of `Spread` and `Totals`, or for markets without a line.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A data frame with one row per recorded price, oldest first, with the price columns (`home`, `draw`, `away`; on Totals the over price is in `home` and the under price in `away`), `hdp` where the market has a line, `timestamp` (Unix milliseconds) and `time` (`POSIXct`, UTC). Attributes `opening` and `latest` hold the opening and latest prices.

Examples

```
## Not run:
mv <- oa_odds_movements(123456, "Bet365", "ML")
plot(mv$time, mv$home, type = "s")

## End(Not run)
```

`oa_odds_summary` *Get an odds summary across bookmakers*

Description

Returns the best, average, median, worst and margin-free fair price per market line across bookmakers, for odds comparison. Available on Growth plans and above; other plans receive a fixed sample response, flagged by the `demo` attribute.

Usage

```
oa_odds_summary(
  event_id,
  markets = NULL,
  lines = NULL,
  bookmakers = NULL,
  include_exchanges = FALSE,
  api_key = NULL
)

oa_odds_summary_multi(
  event_ids,
  markets = NULL,
  lines = NULL,
  bookmakers = NULL,
  include_exchanges = FALSE,
  api_key = NULL
)
```

Arguments

`event_id`	Event id.
`markets`	Optional market names (max 20). The API default is `c("ML", "Spread", "Totals")`.
`lines`	`"main"` (the API default) for each market's main line, or `"all"` for every line priced by at least 3 bookmakers.
`bookmakers`	Optional bookmakers to restrict the summary to; each must be in your selected bookmakers.
`include_exchanges`	If `TRUE`, add the best exchange back price per outcome as `exchange*` columns.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.
`event_ids`	Vector of up to 10 event ids.

Value

A data frame with one row per event, market, line and outcome, with columns such as `eventId`, `market`, `hdp`, `main`, `books`, `outcome`, `best`, `bestBookmakers`, `average`, `median`, `worst` and `fair`. Attribute `demo` is `TRUE` when the response is the sample.

Examples

```
## Not run:
s <- oa_odds_summary(123456, markets = c("ML", "Totals"))
s[, c("market", "hdp", "outcome", "best", "fair")]

## End(Not run)
```

---

oa_odds_updated *Get odds updated since a timestamp*

Description

Returns a delta: only the markets of one bookmaker that changed since `since` (at most 90 seconds ago). Pass the `next_since` attribute of the result as `since` on your next call to continue with no gap. Updates can arrive twice, so merge by `eventId`, `bookmaker` and `market`. A market absent from a response has not changed.

Usage

```
oa_odds_updated(since, bookmaker, sport, markets, limit = NULL, api_key = NULL)
```

Arguments

`since`	Unix timestamp in seconds, or a `POSIXct`, at most 90 seconds old.
`bookmaker`	Bookmaker name, e.g. `"Bet365"`.
`sport`	Sport slug, e.g. `"football"`.
`markets`	Character vector of market names (required, max 40), e.g. `c("ML", "Spread", "Totals")`.
`limit`	Optional maximum number of events (1 to 10000).
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A tidy data frame in the same format as `oa_odds()`, with attributes `next_since` (pass it as `since` on the next call) and `server_time`, both Unix seconds.

Examples

```
## Not run:
upd <- oa_odds_updated(Sys.time() - 20, "Bet365", "football",
                       markets = c("ML", "Totals"))
upd2 <- oa_odds_updated(attr(upd, "next_since"), "Bet365", "football",
                        markets = c("ML", "Totals"))

## End(Not run)
```

`oa_outrights` *List outright tournaments*

Description

Returns outright (futures) tournaments for a sport, sorted by date. Golf is the first sport: PGA Tour, DP World Tour, LIV Golf and the majors. Outrights are a paid add-on; a key without it gets an error that carries the API's message and a link to add it.

Usage

```
oa_outrights(sport = "golf", league = NULL, status = NULL, api_key = NULL)
```

Arguments

`sport`	Sport slug, e.g. `"golf"`.
`league`	Optional league slug: `"pga-tour"`, `"dp-world-tour"`, `"liv-golf"` or `"majors"`.
`status`	Optional statuses: any of `"pending"`, `"live"`, `"settled"`, `"cancelled"`. The API default is pending and live.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A data frame with one row per tournament and columns `id`, `name`, `date`, `status`, `sport.*`, `league.*` and `bookmakerCount` (how many bookmakers you can request price it now).

Examples

```
## Not run:
golf <- oa_outrights("golf", league = "pga-tour")
subset(golf, bookmakerCount > 0)

## End(Not run)
```

oa_outright_odds *Get outright odds for a tournament*

Description

Golf markets are "Winner", "Top 5", "Top 5 (Incl. Ties)", "Top 10", "Top 10 (Incl. Ties)", "Top 20", "Top 20 (Incl. Ties)" and "Make Cut". Top N settles a boundary tie under the bookmaker's dead-heat rules; Top N (Incl. Ties) pays a boundary tie in full.

Usage

```
oa_outright_odds(outright_id, bookmakers, api_key = NULL)
```

Arguments

outright_id	Tournament id, from oa_outrights().
bookmakers	Character vector of bookmaker names (max 30).
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per bookmaker, market and runner, with columns outrightId, bookmaker, market, updatedAt, label (golfer), participantId and the prices: odds (and layOdds on exchanges) for Winner and Top N, yes/no for Make Cut. Prices are decimal odds as strings.

Examples

```
## Not run:
golf <- oa_outrights("golf")
odds <- oa_outright_odds(golf$id[1], c("DraftKings", "Betfair Exchange"))
subset(odds, market == "Winner")

## End(Not run)
```

oa_outright_results *Get outright results for a tournament*

Description

Get outright results for a tournament

Usage

```
oa_outright_results(outright_id, api_key = NULL)
```

Arguments

outright_id	Tournament id, from oa_outrights().
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per golfer and columns label, participantId, position (shared by tied golfers, NA for cut, withdrawn or disqualified), tied, madeCut and status. It is empty until the tournament is settled. Attribute winners is a named list of the participant ids that won each market, and attribute status holds the tournament status.

Examples

```
## Not run:
res <- oa_outright_results(1001)
attr(res, "winners")[["Top 10"]]

## End(Not run)
```

---

oa_participants *List participants (teams and players)*

Description

List participants (teams and players)

Usage

```
oa_participants(sport, search = NULL, api_key = NULL)

oa_participant(participant_id, api_key = NULL)
```

Arguments

sport	Sport slug, e.g. "football".
search	Optional name filter, e.g. "Chelsea".
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.
participant_id	Participant id, for example homeId or awayId from oa_events().

Value

A data frame with one row per participant (columns id, name, sport and, for esports, game and shortCode).

Examples

```
## Not run:
oa_participants("football", search = "Chelsea")

## End(Not run)
```

`oa_search_events` *Search events*

Description

Searches events by team or league name. Returns up to 10 events ranked by relevance. Matching is case- and accent-insensitive.

Usage

```
oa_search_events(query, status = NULL, api_key = NULL)
```

Arguments

`query`	Search term, at least 3 characters.
`status`	`"upcoming"` (the API default) searches events that have not started; `"live"` searches in-play events.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A data frame with one row per matching event.

Examples

```
## Not run:
oa_search_events("Liverpool")
oa_search_events("Liverpool", status = "live")

## End(Not run)
```

oa_selected_bookmakers

Manage the bookmakers selected on your API key

Description

Plans include a fixed number of bookmakers. `oa_selected_bookmakers()` lists the bookmakers selected on your key, `oa_select_bookmakers()` adds bookmakers to the selection and `oa_clear_selected_bookmakers()` clears it. Changing the selection is limited by your plan.

Usage

```
oa_selected_bookmakers(api_key = NULL)

oa_select_bookmakers(bookmakers, api_key = NULL)

oa_clear_selected_bookmakers(api_key = NULL)
```

Arguments

`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.
`bookmakers`	Character vector of bookmaker names, e.g. `c("Bet365", "SingBet")`.

Value

A character vector of the selected bookmaker names (invisibly for `oa_select_bookmakers()` and `oa_clear_selected_bookmakers()`).

Examples

```
## Not run:
oa_selected_bookmakers()
oa_select_bookmakers(c("Bet365", "SingBet"))

## End(Not run)
```

oa_settlements

Get settlement records for events

Description

Returns the result status of up to 50 events. `final` is `TRUE` once you can stop polling. Requires a paid plan.

Usage

```
oa_settlements(event_ids, api_key = NULL)
```

Arguments

event_ids	Vector of 1 to 50 event ids.
api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per known event and columns eventId, sport, status ("pending", "live", "awaiting_result", "settled" or "cancelled"), final, settledAt and, for settled events, score columns (scores.home, scores.away and period columns with keys p1 to pN, ft, ot and ap). Attribute not_found lists ids the API has never seen.

Examples

```
## Not run:
s <- oa_settlements(c(74919888, 73101436))
attr(s, "not_found")

## End(Not run)
```

---

oa_sports *List available sports*

Description

Returns the sports covered by 'Odds-API.io'. This endpoint does not require an API key.

Usage

```
oa_sports(api_key = NULL)
```

Arguments

api_key	API key. Defaults to the ODDS_API_KEY environment variable. Not required for this endpoint.

Value

A data frame with one row per sport (columns name and slug).

Examples

```
## Not run:
sports <- oa_sports()
head(sports)

## End(Not run)
```

`oa_value_bets` *Get value bets*

Description

Returns prices at one bookmaker that are above a fair-odds estimate calculated from the market consensus.

Usage

```
oa_value_bets(
  bookmaker,
  sport = NULL,
  league = NULL,
  include_event_details = FALSE,
  api_key = NULL
)
```

Arguments

`bookmaker`	Bookmaker name, e.g. `"Bet365"`. Required by the API.
`sport`	Optional sport slug filter, e.g. `"baseball"`.
`league`	Optional league slug filter; requires `sport` to be set.
`include_event_details`	If `TRUE`, event information (teams, date, sport, league) is included in the result.
`api_key`	API key. Defaults to the `ODDS_API_KEY` environment variable. Not required for this endpoint.

Value

A data frame with one row per value bet, including the bet side, expected value and bookmaker odds. `expectedValue` is bookmaker odds / fair odds x 100: 103.42 is a 3.42% edge, so the edge in percent is `expectedValue - 100`. Values below 100 are a negative edge.

Examples

```
## Not run:
vb <- oa_value_bets("Bet365", include_event_details = TRUE)
# Bets with at least a 3% edge
vb[vb$expectedValue - 100 >= 3, ]

## End(Not run)
```

Index

oa_arbitrage_bets, 2
oa_bookmakers, 4
oa_clear_selected_bookmakers (oa_selected_bookmakers), 25
oa_closing_lines, 4
oa_dropping_odds, 6
oa_event, 7
oa_events, 8
oa_events(), *7*, *10*, *17*, *23*
oa_grade, 9
oa_historical_events, 10
oa_historical_events(), *11*
oa_historical_odds, 11
oa_historical_search, 12
oa_leagues, 13
oa_live_events, 13
oa_mappings, 14
oa_markets, 15
oa_markets(), *11*, *16, 17*
oa_multi_odds, 16
oa_odds, 17
oa_odds(), *5*, *9*, *11*, *14–16*, *20*
oa_odds_movements, 18
oa_odds_summary, 19
oa_odds_summary_multi (oa_odds_summary), 19
oa_odds_updated, 20
oa_outright_odds, 22
oa_outright_results, 22
oa_outrights, 21
oa_outrights(), *22, 23*
oa_participant (oa_participants), 23
oa_participants, 23
oa_search_events, 24
oa_select_bookmakers (oa_selected_bookmakers), 25
oa_selected_bookmakers, 25
oa_settlements, 25
oa_sports, 26
oa_sports(), *8*, *13*
oa_value_bets, 27