Package 'pinnacle.API'

October 14, 2022

Type Package

Version 2.3.3

Title A Wrapper for the Pinnacle API

URL https://github.com/marcoblume/pinnacle.API#README

Author Nicholas Jhirad, Marco Blume, Aaron Jacobs, Amine Gassem

Maintainer Marco Blume <marco.blume@pinnaclesports.com>

Description An interface to the API by Pinnacle that allows Pinnacle customers to interact with the sports market data in R.See <https://www.pinnacle.com/en/api> for more information. The Pinnacle API can be used to place wagers, retrieve line information, retrieve account information.Please be aware that the TOC of Pinnacle apply <https://www.pinnacle.com/en/termsandconditions>. An account with Pinnacle is necessary to use the Pinnacle API.

LazyData true

License GPL-3

BugReports https://github.com/marcoblume/pinnacle.API/issues

Imports data.table (>= 1.10.0), openssl, magrittr, purrr, uuid, httr, jsonlite, rjson

RoxygenNote 6.0.1

Suggests testthat (>= 2.0.0), yaml (>= 2.1.15)

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-02 17:59:34 UTC

R topics documented:

`AcceptTermsAndConditions` *Accept terms and conditions, only run once per session, must agree to terms or functions will not work*

Description

Accept terms and conditions, only run once per session, must agree to terms or functions will not work

Usage

```
AcceptTermsAndConditions(accepted = FALSE)
```

Arguments

`accepted`	Default=FALSE , BOOLEAN

Examples

```
AcceptTermsAndConditions(accepted=TRUE)
```

`authorization` *Authorization for the Pinnacle API*

Description

Authorization for the Pinnacle API

Usage

```
authorization(user, pwd)
```

Arguments

`user`	Pinnacle Username
`pwd`	Pinnacle Password

`CheckTermsAndConditions` *Prompts User for Terms and Conditions, otherwise stops running function*

Description

Prompts User for Terms and Conditions, otherwise stops running function

Usage

```
CheckTermsAndConditions()
```

Examples

```
CheckTermsAndConditions()
```

FixNames *Fix up names to be shorter*

Description

Fix up names to be shorter

Usage

```
FixNames(dt, lastx = 2)
```

Arguments

dt	a data.table with pinnacle.API standard names
lastx	last x names to use as part of an identifier

Value

same data.frame with names fixed to length x indicators

GetAPIEndpoint *Gets the current API endpoint*

Description

Gets the current API endpoint

Usage

```
GetAPIEndpoint()
```

Value

the currently set API endpoint

Examples

```
SetAPIEndpoint("https://api.pinnaclesports.com/v2/")
GetAPIEndpoint()
SetAPIEndpoint("https://api.pinnaclesports.com")
```

GetBetsList *Get a list of running/settled bets*

Description

Get a list of running/settled bets

Usage

```
GetBetsList(betids = NULL, betlist = c("SETTLED", "RUNNING"),
  fromDate = as.POSIXlt(Sys.Date(), tz = "UTC") - 15 * 24 * 60 * 60,
  toDate = as.POSIXlt(Sys.Date(), tz = "UTC") + 24 * 60 * 60)
```

Arguments

betids	a vector of betids (overrides betlist) default = NULL
betlist	Either 'SETTLED' or 'RUNNING' Default Behavior shows both
fromDate	Iso8061 Date Default: 15 days prior in UTC, as.POSIXct(Sys.Date(), tz = 'UTC')-15*24*60*60
toDate	Iso8061 Date Default: 1 day ahead in UTC (to counter possible fencepost situations), as.POSIXct(Sys.Date(), tz = 'UTC') + 24*60*60

Value

A list of bets and associated details

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetBetsList()
```

GetBettingStatus *Check the Pinnacle API's Betting Status*

Description

Checks whether betting through the API is currently enabled. Betting, particularly betting on live events, may be closed during maintenance.

Usage

```
GetBettingStatus()
```

Details

This function will raise an error if the API does not return HTTP status `OK`. For information on the possible errors, see the API documentation for Get Betting Status.

Value

A string containing the betting status of the API, which should be one of

- `ALL_BETTING_ENABLED`
- `ALL_LIVE_BETTING_CLOSED`
- `ALL_BETTING_CLOSED`

Examples

```
SetCredentials("TESTAPI", "APITEST")
AcceptTermsAndConditions(accepted = TRUE)
GetBettingStatus()
```

`GetClientBalance` *Get Client Balance*

Description

Get Client Balance

Usage

```
GetClientBalance(force = TRUE)
```

Arguments

`force`	Default=TRUE , boolean if TRUE force a reload of the data if FALSE use cached data

Value

vector client balance parameter

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetClientBalance()
```

GetCurrencies *Get the list of supported Currencies*

Description

Get the list of supported Currencies

Usage

```
GetCurrencies(force = TRUE)
```

Arguments

force	Default=TRUE, boolean if TRUE force a reload of the data if FALSE use cached data

Value

a data frame with these columns:

- Currency Code
- Exchange Rate to USD
- Currency Name

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetCurrencies()
```

GetFixtures *Get Non-Settled Events for a Given Sport*

Description

Queries the event listing for a given sport, which can be filtered by league and/or event ID, and narrowed to include only live events.

Usage

```
GetFixtures(sportid, leagueids = NULL, eventids = NULL, since = NULL,
  islive = FALSE)
```

Arguments

sportid	An integer giving the sport. If this is missing in interactive mode, a menu of options is presented to the user.
leagueids	A vector of league IDs, or NULL.
eventids	A vector of event IDs, or NULL.
since	To receive only listings updated since the last query, set since to the value of last from the previous fixtures response. Otherwise it will query all listings.
islive	When TRUE, retrieve only live events.

Details

This function will raise an error if the API does not return HTTP status OK. For information on the possible errors, see the API documentation for Get Fixtures.

Value

A data frame with rows containing matching events and columns containing sport, league, and event information. Not all sports return the same listing format – in particular, only baseball listings will have pitcher information.

See Also

See GetSettledFixtures to retrieve settled events, or GetSpecialFixtures to retrieve special contestants for a sport.

Examples

```
SetCredentials("TESTAPI", "APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetFixtures(sportid = 41, leagueids = 191545)
```

GetInrunning *GetInrunning*

Description

GetInrunning

Usage

```
GetInrunning()
```

Value

A dataframe containing the current State of live events

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetInrunning()
```

GetLeagues *Get Leagues for Sport(s) by name*

Description

Returns all Leagues for the Sport(s)

Usage

```
GetLeagues(sports, force = TRUE, regex = FALSE)
```

Arguments

sports	character vector of sports names.
force	Default=FALSE, boolean if TRUE force a reload of the data if FALSE use cached data
regex	Default=FALSE, boolean if TRUE , retreives sports id using regular expression on names

Value

a data frame having columns:

- LeagueID
- LinesAvailable
- HomeTeam
- AllowRoundRobin
- LeagueName

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetLeagues("Badminton")
```

`GetLeaguesByID` *Get Leagues for Sport(s) by ID*

Description

Returns all Leagues for the Sport(s)

Usage

```
GetLeaguesByID(sportid, force = TRUE)
```

Arguments

`sportid`	integer vector of sports IDs
`force`	boolean whether to get new data (TRUE) or use cached data (FALSE)

Value

a data frame having columns:

- LeagueID
- LinesAvailable
- HomeTeam
- AllowRoundRobin
- LeagueName

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetLeaguesByID(1)
```

`GetLine` *Get Lines (Use to get more detail on a single line, but the GetOdds or showOddsDF versions are intended for large amounts of data)*

Description

Get Lines (Use to get more detail on a single line, but the GetOdds or showOddsDF versions are intended for large amounts of data)

Usage

```
GetLine(sportid, leagueids, eventid, periodnumber, betType, team = NULL,
  side = NULL, handicap = NULL, oddsFormat = "AMERICAN", force = TRUE)
```

Arguments

sportid	The sport ID
leagueids	integer vector of leagueids.
eventid	numeric xxxxx
periodnumber	xxxxx
betType	xxxx
team	xxxx
side	xxx
handicap	xxx
oddsFormat	xxx
force	passed along to GetSports

Value

returns a data frame with columns:

- SportID
- Last
- League
- LeagueID
- EventID
- StartTime
- HomeTeamName
- AwayTeamName
- Rotation Number
- Live Status
- Status
- Parlay Status

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
 GetLine(sportId=1,leagueids=191545,eventId=495418854,
 periodNumber=0,team="TEAM1",betType="Moneyline")
```

GetOdds *Get Odds*

Description

Get Odds

Usage

```
GetOdds(sportid, leagueids = NULL, since = NULL, islive = 0,
  oddsformat = "AMERICAN", tableformat = "mainlines", force = TRUE)
```

Arguments

sportid	(optional) The sport id for which to retrieve the fixutres
leagueids	(optional) integer vector of leagueids.
since	(optional) numeric This is used to receive incremental updates. Use the value of last from previous response.
islive	boolean if TRUE retrieves ONLY live events
oddsformat	default AMERICAN, see API manual for more options
tableformat	• 'mainlines' (default), only shows mainlines • 'long' for a single record for each spread/total on an event, • 'wide' for all lines as one record, • 'subtables' all lines for spreads/totals stored as nested tables
force	boolean if FALSE, functions using cached data will use the values since the last force

Value

data.frame of odds

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted = TRUE)
# We can run without parameters, and will be given a selection of sports
GetOdds()
```

GetPassword *Get your Password*

Description

Get your Password

Usage

```
GetPassword()
```

Value

Current Password in plaintext

GetSettledFixtures *Get Settled Fixtures*

Description

Get Settled Fixtures

Usage

```
GetSettledFixtures(sportid, leagueids = NULL, since = FALSE)
```

Arguments

sportid	(optional) an integer giving the sport, if missing, a menu of options is presented
leagueids	(optional) integer vector with league IDs.
since	(optional), numeric, This is used to receive incremental updates. Use the value of 'last' from previous fixtures response.

Value

a data.frame of settled fixtures

Examples

```
SetCredentials("TESTAPI", "APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetSettledFixtures()
```

GetSettledSpecialFixtures *Get Settled Special Fixtures*

Description

Get Settled Special Fixtures

Usage

```
GetSettledSpecialFixtures(sportid, leagueids = NULL, since = NULL)
```

Arguments

sportid	(optional) an integer giving the sport, if missing, a menu of options is presented
leagueids	(optional) integer vector with league IDs.
since	(optional) numeric This is used to receive incremental updates. Use the value of last from previous fixtures response.

Value

a data.frame of settled special fixtures

Examples

```
SetCredentials("TESTAPI", "APITEST")
AcceptTermsAndConditions(accepted=TRUE)
# Can be run without arguments
GetSettledSpecialFixtures()
```

GetSpecialFixtures *Get Special Fixtures*

Description

Get Special Fixtures

Usage

```
GetSpecialFixtures(sportid, leagueids = NULL, category = NULL,
  eventid = NULL, specialid = NULL, since = NULL)
```

Arguments

sportid	(optional) an integer giving the sport, if missing, a menu of options is presented
leagueids	(optional) integer vector with league IDs.
category	(optional) See API Manual
eventid	(optional) Associated event ID
specialid	(optional) Associated special ID
since	(optional) numeric This is used to receive incremental updates. Use the value of last from previous fixtures response.

Value

a data.frame of special fixtures

Examples

```
SetCredentials("TESTAPI", "APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetSpecialFixtures()
```

GetSpecialLine *Get the Line for a Special Contestant*

Description

Queries the current line and odds for a given contestant in a special.

Usage

```
GetSpecialLine(specialId, contestantId, oddsFormat = "AMERICAN")
```

Arguments

specialId	The ID of the special for the contestant.
contestantId	The ID of the contestant.
oddsFormat	Format for the returned odds. One of "AMERICAN", "DECIMAL", "HONGKONG", "INDONESIAN", or "MALAY".

Details

This function will raise an error if the API does not return HTTP status OK. For information on the possible errors, see the API documentation for Get Special Line.

Value

A data frame with the following columns:

`status` When a line ID is retrieved this will contain the code `"SUCCESS"`. Otherwise it may contain `"NOT_EXISTS"` or `"OFFLINE"`.

`specialId` The ID of the special.

`contestantId` The ID of the contestant.

`minRiskStake` Minimum bettable risk amount.

`maxRiskStake` Maximum bettable risk amount.

`minWinStake` Minimum bettable win amount.

`maxWinStake` Maximum bettable win amount.

`lineId` Line ID needed to place a bet.

`price` Latest price.

`handicap` Handicap value, if applicable.

See Also

See `GetLine` to retrieve non-special lines, `GetSpecialFixtures` to query available special contestants, and `PlaceSpecialBet` to actually wager on a contestant.

Examples

```
SetCredentials("TESTAPI", "APITEST")
AcceptTermsAndConditions(accepted = TRUE)

# This contest is unlikely to exist, but serves as an example
# of the format.
GetSpecialLine(specialId = 101, contestantId = 102,
               oddsFormat = "AMERICAN")
```

`GetSpecialOdds` *Get Special Odds*

Description

Get Special Odds

Usage

```
GetSpecialOdds(sportid, leagueids = NULL, since = NULL,
  oddsformat = "AMERICAN", tableformat = "clean", force = TRUE)
```

Arguments

sportid	(optional) The sport id for which to retrieve the fixutres
leagueids	(optional) integer vector of leagueids.
since	(optional) numeric This is used to receive incremental updates. Use the value of last from previous response.
oddsformat	default AMERICAN, see API manual for more options
tableformat	• 'clean' default should return each contestant records • 'long' for a single record for each spread/total on an event, • 'wide' for all lines as one record, • 'subtables' all lines for spreads/totals stored as nested tables
force	boolean if FALSE, functions using cached data will use the values since the last force

Value

data.frame of odds

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetSpecialOdds()
```

GetSports *Get Sports*

Description

Returns all sports with the status whether they currently have lines or not

Usage

```
GetSports(force = TRUE)
```

Arguments

force	Default=TRUE, boolean if TRUE force a reload of the data if FALSE use cached data

Value

a data frame with these columns:

- SportID
- LinesAvailable
- SportName

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
GetSports()
```

GetUsername *Get your Username*

Description

Get your Username

Usage

```
GetUsername()
```

Value

A String, your current username

Examples

```
SetCredentials("TESTAPI","APITEST")
GetUsername()
```

PlaceBet *Place Bet*

Description

Place bet in the system

Usage

```
PlaceBet(stake, sportId, eventId, periodNumber, lineId, betType,
  altLineId = NULL, team = NULL, side = NULL, acceptBetterLine = TRUE,
  winRiskStake = "RISK", oddsFormat = "AMERICAN")
```

Arguments

stake	numeric Wager amount in currency
sportId	numeric the sport id
eventId	numeric the vent id
periodNumber	numeric Number of the period , see Pinnacle API manual
lineId	numeric ID of the line
betType	BET_TYPE • SPREAD • MONEYLINE • TOTAL_POINTS • TEAM_TOTAL_POINTS
altLineId	numeric ID of the alternate line (lineID must also be included)
team	Default = NULL , , see Pinnacle API manual • TEAM1 • TEAM2 • DRAW
side	Defaulat = NULL , , see Pinnacle API manual • OVER • UNDER
acceptBetterLine	Default=TRUE ,boolean Whether or not to accept a bet when there is a line change in favor of the client
winRiskStake	Default="RISK", either place the stake to RISK/WIN • WIN • RISK
oddsFormat	Default="AMERICAN", Desired Odds Format • AMERICAN • DECIMAL • HONGKONG • INDONESIAN • MALAY

Value

list containing :

- status If Status is PROCESSED_WITH_ERROR errorCode will be in the response
- errorCode

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
PlaceBet (stake=10,
          sportId=1,
          eventId=495418854,
          periodNumber=0,
          lineId=222136736,
          betType="MONEYLINE",
          team="TEAM2",
          acceptBetterLine=TRUE,
          winRiskStake="WIN",
          oddsFormat="AMERICAN")
```

`PlaceParlayBet` *PlaceParlayBet*

Description

Place parlay or round robin parlay bet in the system

Usage

```
PlaceParlayBet(riskAmount, legslist, roundRobinOptions = c("Parlay",
  "TwoLegRoundRobin", "ThreeLegRoundRobin", "FourLegRoundRobin",
  "FiveLegRoundRobin", "SixLegRoundRobin", "SevenLegRoundRobin",
  "EightLegRoundRobin")[1], oddsFormat = "AMERICAN",
  acceptBetterLine = TRUE)
```

Arguments

`riskAmount`	numeric Wager amount in currency

`legslist`	A list of wagers, where each wager must be in named list format. Required named values are: legBetType, lineId, either team or side, and periodNumber. Optional named values are: altLineId, pitcher1MustStart, or pitcher2MustStart. See the API Manual for more info
	• lineId
	• altLineId OPTIONAL
	• sportId
	• eventId
	• periodNumber
	• legBetType
	  – MONEYLINE
	  – SPREAD
	  – TOTAL

- TEAMTOTAL
- team/side using one will invalidate the other
- pitcher1MustStart OPTIONAL
- pitcher2MustStart OPTIONAL

`roundRobinOptions`
: one of the round robin options, default is 'Parlay'

- Parlay
- TwoLegRoundRobin
- ThreeLegRoundRobin
- FourLegRoundRobin
- FiveLegRoundRobin
- SixLegRoundRobin
- SevenLegRoundRobin
- EightLegRoundRobin

`oddsFormat`
: default:'AMERICAN'

- AMERICAN
- DECIMAL
- HONGKONG
- INDONESIAN
- MALAY

`acceptBetterLine`
: Default TRUE ,boolean Whether or not to accept a bet when there is a line change in favor of the client

Value

list containing:

- status If Status is PROCESSED_WITH_ERROR errorCode will be in the response
- errorCode

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
parlay1 <-  list(lineId = 222136736,
                 sportId=1,
                 eventId = 495418854,
                 periodNumber=0,
                 legBetType = "MONEYLINE",
                 team = 'TEAM1')
parlay2 <- list(lineId = 223187865,
                 sportId=1,
                 eventId = 496997901,
                 periodNumber=0,
                 legBetType = "TOTAL_POINTS",
```

```
                  side = 'OVER')
legslist <- list(parlay1,parlay2)


PlaceParlayBet(riskAmount=10,
               legslist=legslist,
               roundRobinOptions="Parlay",
               oddsFormat="AMERICAN" ,
               acceptBetterLine=TRUE)
```

PlaceSpecialBet *Place a Special Bet on a Given Contestant*

Description

Place a wager on a contestant in a given special line.

Usage

```
PlaceSpecialBet(stake, lineId, specialId, contestantId,
  acceptBetterLine = TRUE, winRiskStake = "RISK", oddsFormat = "AMERICAN")
```

Arguments

stake	The amount to be wagered.
lineId	The line to wager on. See GetSpecialLine.
specialId	The ID of the special offer.
contestantId	The ID of the contestant wagered on.
acceptBetterLine	Whether or not to accept a bet when there is a line change in favour of this wager.
winRiskStake	Whether the stake is the risk or win amount. One of "RISK" or "WIN".
oddsFormat	Format for the returned odds. One of "AMERICAN", "DECIMAL", "HONGKONG", "INDONESIAN", or "MALAY".

Details

This function will raise an error if the API does not return HTTP status OK, which is not precisely the same as an assurance that the wager was placed successfully (see the Value section). For information on the possible errors, see the API documentation for Place Special Bet.

Value

A data frame with the following columns:

`status` When the wager is placed this will contain code `"ACCEPTED"`. Otherwise it will contain code `"PROCESSED_WITH_ERROR"`.

`errorCode` When the wager is not accepted, this column will contain a code for the particular error involved; otherwise it will be `NA`.

`uniqueRequestId` A unique ID associated with the wager.

When the wager is accepted, the data frame will also contain the following:

`betId` A unique ID for the newly created bet.

`betterLineWasAccepted` Whether or not the bet was accepted on a line that changed in favour of wager.

When the wager is not accepted, the data frame may also contain `lineId` and `specialBet` columns with NA values.

See Also

See `PlaceBet` to make non-special wagers, `GetSpecialFixtures` to query available special contestants, and `GetSpecialLine` to get their associated lines.

Examples

```
SetCredentials("TESTAPI", "APITEST")
AcceptTermsAndConditions(accepted = TRUE)

# This contest is unlikely to exist, but serves as an example
# of the format.
line <- GetSpecialLine(specialId = 101, contestantId = 102,
                       oddsFormat = "AMERICAN")

if (!is.na(line$lineId)) {
  PlaceSpecialBet(stake = 100, lineId = line$lineId,
                  specialId = 101, contestantId = 102,
                  acceptBetterLine = TRUE,
                  winRiskStake = "RISK",
                  oddsFormat = "AMERICAN")
}
```

SetAPIEndpoint *Sets the API endpoint to use*

Description

Sets the API endpoint to use

Usage

```
SetAPIEndpoint(url = "https://api.pinnaclesports.com")
```

Arguments

url	a url, default value is the usual API endpoint

Value

void

Examples

```
SetAPIEndpoint("https://api.pinnaclesports.com")
SetAPIEndpoint()
```

SetCredentials *Set your pinnaclesports.com user credentials*

Description

Set your pinnaclesports.com user credentials

Usage

```
SetCredentials(username, password)
```

Arguments

username	Your username
password	Your password

Examples

```
SetCredentials("TESTAPI","APITEST")
```

showOddsDF *showOddsDF - Takes a GetOdds JSON response and combines with Fixtures and Inrunning*

Description

showOddsDF - Takes a GetOdds JSON response and combines with Fixtures and Inrunning

Usage

```
showOddsDF(sportid, leagueids = NULL, since = NULL, islive = 0,
  force = TRUE, tableformat = "mainlines", namesLength = 3,
  attachLeagueInfo = TRUE, oddsformat = "AMERICAN", fixtures_since = NULL)
```

Arguments

sportid	(optional) The sportid to get odds from, if none is given, a list of options and a prompt are provided
leagueids	numeric vector of leagueids - can get as output from GetLeagues
since	numeric This is used to receive incremental updates. this will give all lines that have changed odds.
islive	boolean if TRUE retrieves ONLY live events
force	boolean default set to TRUE, forces a reload of the cache.
tableformat	• 'mainlines' (default), only shows mainlines • 'long' for a single record for each spread/total on an event, • 'wide' for all lines as one record, • 'subtables' all lines for spreads/totals stored as nested tables
namesLength	how many identifiers to use in the names, default is 3
attachLeagueInfo	whether or not to include league information in the data
oddsformat	default AMERICAN, see API manual for more options bettable leagues
fixtures_since	if set, get only fixtures that were posted since last.

Value

a dataframe combining GetOdds and GetFixtures data, containing NA's where levels of factors do not have a value. Naming convention is as follows, Example: spread.altLineId.N is the altLineId associated with spread.hdp.(N+1) whereas spread.hdp refers to the mainline. spread.altLineId is the first alternate, and equivalent to spread.altLineId.0

Examples

```
SetCredentials("TESTAPI","APITEST")
AcceptTermsAndConditions(accepted=TRUE)
# Run without arguments, it will prompt you for the sport
showOddsDF()
```

Index