Package ‘SC2API’

October 12, 2022

Title  Blizzard SC2 API Wrapper
Version  1.0.0
Description  A wrapper for Blizzard’s Starcraft II (a 2010 real-time strategy game) Application Programming Interface (API). All documented API calls are implemented in an easy-to-use and consistent manner.
License  MIT + file LICENSE
LazyData  true
Encoding  UTF-8
Imports  data.table, httr, jsonlite
Suggests  rmarkdown, knitr, httptest, ggplot2, testthat
VignetteBuilder  knitr
RoxygenNote  7.1.1
NeedsCompilation  no
Author  Samuel Morrissette [cre, aut]
Maintainer  Samuel Morrissette <samuel.morrissette01@gmail.com>
Repository  CRAN
Date/Publication  2020-09-25 13:20:06 UTC

R topics documented:

authorization ................................................................. 2
get_gm_leaderboard .......................................................... 3
get_ladder ................................................................. 4
get_ladder_data ............................................................ 6
get_ladder_ids ............................................................ 7
get_ladder_summary ........................................................ 8
get_last_played ............................................................ 10
get_league_counts .......................................................... 11
get_league_data ............................................................ 12
get_legacy_achievements .................................................. 14
get_legacy_ladder .......................................................... 15
The Blizzard API uses OAuth 2.0 for authorization. For more information on how Blizzard uses OAuth in their API, visit https://develop.battle.net/documentation/guides/using-oauth.

Before using the Blizzard API, one must first create a client in the Blizzard Developer Portal and obtain a valid client ID and client secret. For more information on getting started, see: Getting Started.

Once a client has been created, use set_token and supply the client id and client secret as arguments to set an environment variable for all future API calls.

Once set_token() has been used, an access token can be removed from the environment using remove_token()

Note that access tokens are set to expire in 24 hours and, subsequently, a new token must be used for any future API calls.

Usage

set_token(client_id, client_secret, access_token, verbose = FALSE)
validate_token(access_token)
unset_token()

Arguments

client_id, client_secret
A client ID and client secret can be obtained from the Blizzard Developer Portal. For more information on creating a client, visit Getting Started.

access_token
An OAuth 2.0 access token required to use the Blizzard API. Access tokens can be obtained by using set_token with a valid client ID and client secret.

verbose
If verbose is set to TRUE, your access token will be printed on screen.
Note
Access tokens expire after 24 hours.

References
- Blizzard Developer Portal
- Using OAuth
- Getting Started

Examples
```r
# Not run:
# Get and set a token as an environment variable
set_token(client_id = "YOUR CLIENT ID", client_secret = "YOUR CLIENT SECRET")

# Set an access token that you have already retrieved as an environment variable
set_token(access_token = "YOUR TOKEN")

# End(Not run)

# Ensure that a valid token is currently set as an environment variable
validate_token()
# Check if a token is valid
validate_token("TEST TOKEN")

# Remove token from environment variable
unset_token()
```

---

**get_gm_leaderboard**  
*Grandmaster Leaderboard*

**Description**
Provides a full listing of players currently in the grandmaster leaderboard. Also provides other information such as player profile information, records (match record, MMR, etc.), and clans.

**Usage**
```r
get_gm_leaderboard(region_id, host_region = "us")
```

**Arguments**
- `region_id`  
  A numeric argument indicating the region of the profile.
  - 1 = US Region
  - 2 = EU Region
  - 3 = KR/TW Region
  - 5 = CN Region
**get_ladder**

**host_region**
The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

**Note**
This API call is currently not supported for the China region (region_id = 5).

**References**
- Blizzard Community API Documentation
- Regionality and APIs
- Grandmaster League

**See Also**
Other ladder data API calls: get_season()

**Examples**
```python
# Obtain GM leaderboard for the Korea region
try(get_gm_leaderboard(region_id = 3))
```

---

### get_ladder

**Ladder Details and Profile Rank**

**Description**
Provides information about a particular ladder and the individual’s rank and status within that ladder (i.e. rank, MMR, etc.).

**Usage**
```python
get_ladder(region_id, realm_id, profile_id, ladder_id, host_region = "us")
```

**Arguments**
- **region_id**: A numeric argument indicating the region of the profile.
  - 1 = US Region
  - 2 = EU Region
  - 3 = KR/TW Region
  - 5 = CN Region
- **realm_id**: A numeric argument indicating the realm of the profile. A realm is a subset of the region.
get_ladder

- US Region
  - 1 = US
  - 2 = LatAm
- EU Region
  - 1 = Europe
  - 2 = Russia
- KR/TW Region
  - 1 = Korea
  - 2 = Taiwan

profile_id  A unique, numeric identifier for an individual’s profile.
ladder_id  A unique identifier for a particular ladder. With the exception of Grandmaster, leagues (bronze, silver, etc.), are separated into tiers (1,2,3) which are further separated into divisions. These divisions, or ladders, each have a unique identifier.

host_region  The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

References

- Blizzard Community API Documentation
- Regionality and APIs

See Also

Other profile API calls: get_ladder_summary(), get_metadata(), get_profile(), get_static()

Examples

```r
# Obtaining the overall ladder performance of a profile.
try(
  ladderData <- get_ladder_summary(region_id = 1, realm_id = 1, profile_id = 4716773)

  # Choose a single ladder ID
  ladderID <- ladderData$allLadderMemberships$ladderId[1]

  # Get full ladder information and the profile's performance in this ladder
  get_ladder(region_id = 1, realm_id = 1, profile_id = 4716773, ladder_id = ladderID)
)
```
Description

Provides data of players in a particular ladder. This includes MMR, points, win/loss record, time of joining, time of a player’s last game, and more.

Usage

```r
get_ladder_data(ladder_id, host_region = "us")
```

Arguments

- `ladder_id`: A unique identifier for a particular ladder. With the exception of grandmaster, leagues (bronze, silver, etc.), are separated into tiers (1,2,3) which are further separated into divisions. These divisions, or ladders, each have a unique identifier.
- `host_region`: The host region that the API call will be sent to. For `get_ladder_data`, the host region MUST be the region that the ladder is a part of. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

Note

For `get_ladder_data`, the host region MUST be the region that the ladder is a part of.

References

- Blizzard Community API Documentation
- Regionality and APIs

See Also

Other game data API calls: `get_league_data()`, `get_season_data()`

Examples

```r
### Obtain battle tags and MMR of players in a particular ladder.

# Get full ladder data
try({
  data <- get_ladder_data(ladder_id = 289444, host_region = "us")

  # Player ratings
  ratings <- data$team$rating
})
```
get_ladder_ids

# Get battle tags using list indexing with sapply
tags <- sapply(data$team$member, function(x) x$character_link$battle_tag)
}

---

get_ladder_ids League Ladder ID's

Description

Provides ladder ID's for all divisions in a league's tiers.

Usage

get_ladder_ids(season_id, queue_id, team_type, league_id, host_region)

Arguments

season_id    A numeric argument indicating a particular ladder season. Currently, league data is only available for season 28 and higher (i.e. data prior to this season is inaccessible).
queue_id     • 1 = WoL 1v1
             • 2 = WoL 2v2
             • 3 = WoL 3v3
             • 4 = WoL 4V4
             • 101 = HotS 1v1
             • 102 = HotS 2v2
             • 103 = HotS 3v3
             • 104 = HotS 4v4
             • 201 = LotV 1v1
             • 202 = LotV 2v2
             • 203 = LotV 3v3
             • 204 = LotV 4v4
             • 206 = LotV Archon

team_type    • 0 = Arranged
             • 1 = Random

league_id    • 0 = Bronze
             • 1 = Silver
             • 2 = Gold
             • 3 = Platinum
             • 4 = Diamond
             • 5 = Masters
             • 6 = Grandmaster
**get_ladder_summary**

**host_region**

The host region that the API call will be sent to. For `get_league_data`, the host region affects the data you will receive (i.e. different regions will result in different data). Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

**Value**

List of ladder ID’s separated by tier. For grandmaster league, an integer is returned.

**Note**

Data is only available for season 28 and higher.

**References**

- Blizzard Community API Documentation
- Regionality and APIs

**Examples**

```r
# Get all ladder ID’s for Season 35, LotV 2v2, randomly gathered teams, # Bronze league, Korean region.
try({
  data <- get_ladder_ids(season_id = 35,
    queue_id = 202,
    team_type = 1,
    league_id = 0,
    host_region = "kr"
})
```

---

**get_ladder_summary**  Profile Ladder Summary

**Description**

Provides a detailed list of ladder membership, profile showcases, and placement matches.

**Usage**

```r
get_ladder_summary(region_id, realm_id, profile_id, host_region = "us")
```
get_ladder_summary

Arguments

region_id A numeric argument indicating the region of the profile.
- 1 = US Region
- 2 = EU Region
- 3 = KR/TW Region
- 5 = CN Region

realm_id A numeric argument indicating the realm of the profile. A realm is a subset of the region.
- US Region
  - 1 = US
  - 2 = LatAm
- EU Region
  - 1 = Europe
  - 2 = Russia
- KR/TW Region
  - 1 = Korea
  - 2 = Taiwan

profile_id A unique, numeric identifier for an individual’s profile.

host_region The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

References

- Blizzard Community API Documentation
- Regionality and APIs

See Also

Other profile API calls: get_ladder(), get_metadata(), get_profile(), get_static()

Examples

# Request ladder summary of a particular profile in the U.S. region and U.S. realm.
try(get_ladder_summary(region_id = 1, realm_id = 1, profile_id = 4716773))
get_last_played

Last Played Match

Description
Get the time of the last played match in a player's match history.

Usage
get_last_played(region_id, realm_id, profile_id, host_region = "us")

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>region_id</td>
<td>A numeric argument indicating the region of the profile.</td>
</tr>
<tr>
<td></td>
<td>• 1 = US Region</td>
</tr>
<tr>
<td></td>
<td>• 2 = EU Region</td>
</tr>
<tr>
<td></td>
<td>• 3 = KR/TW Region</td>
</tr>
<tr>
<td></td>
<td>• 5 = CN Region</td>
</tr>
<tr>
<td>realm_id</td>
<td>A numeric argument indicating the realm of the profile.</td>
</tr>
<tr>
<td></td>
<td>• US Region</td>
</tr>
<tr>
<td></td>
<td>• 1 = US</td>
</tr>
<tr>
<td></td>
<td>• 2 = LatAm</td>
</tr>
<tr>
<td></td>
<td>• EU Region</td>
</tr>
<tr>
<td></td>
<td>• 1 = Europe</td>
</tr>
<tr>
<td></td>
<td>• 2 = Russia</td>
</tr>
<tr>
<td></td>
<td>• KR/TW Region</td>
</tr>
<tr>
<td></td>
<td>• 1 = Korea</td>
</tr>
<tr>
<td></td>
<td>• 2 = Taiwan</td>
</tr>
<tr>
<td>profile_id</td>
<td>A unique, numeric identifier for an individual's profile.</td>
</tr>
<tr>
<td>host_region</td>
<td>The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of &quot;us&quot;, &quot;eu&quot;, &quot;kr&quot;, &quot;tw&quot;, &quot;cn&quot;. For more information on regionality, refer to Regionality and APIs.</td>
</tr>
</tbody>
</table>

Note
Data is only available for season 28 and higher.

References
- Blizzard Community API Documentation
- Regionality and APIs
Examples

```r
# Get last played match for a particular profile
try(get_last_played(1, 4716773, host_region = "us"))
```

---

**get_league_counts**  
*League and Tier Counts*

**Description**

Provides both league (i.e. bronze, silver, etc.) and tier (1, 2, 3) player counts.

**Usage**

```r
get_league_counts(season_id, queue_id, team_type, league_id, host_region)
```

**Arguments**

- **season_id**  
  A numeric argument indicating a particular ladder season. Currently, league data is only available for season 28 and higher (i.e. data prior to this season is inaccessible).

- **queue_id**  
  - 1 = WoL 1v1
  - 2 = WoL 2v2
  - 3 = WoL 3v3
  - 4 = WoL 4v4
  - 101 = HotS 1v1
  - 102 = HotS 2v2
  - 103 = HotS 3v3
  - 104 = HotS 4v4
  - 201 = LotV 1v1
  - 202 = LotV 2v2
  - 203 = LotV 3v3
  - 204 = LotV 4v4
  - 206 = LotV Archon

- **team_type**  
  - 0 = Arranged
  - 1 = Random

- **league_id**  
  - 0 = Bronze
  - 1 = Silver
  - 2 = Gold
  - 3 = Platinum
  - 4 = Diamond
  - 5 = Masters
get_league_data

League Data

Description

League data is divided into 3 tiers for each league (with the exception of grandmaster, which only has 1 tier) and further divided into a number of divisions depending on how many players are in a given league. League data contains the number of divisions, the unique ladder ID of each division and the total player count contained within each division.

Usage

get_league_data(season_id, queue_id, team_type, league_id, host_region = "us")

Value

A list with tier counts and the overall league count.

Note

Data is only available for season 28 and higher.

References

- Blizzard Community API Documentation
- Regionality and APIs

Examples

# Get league counts for Season 35, LoTV 2v2, randomly gathered teams, # Bronze league, Korean region.
try({
  data <- get_league_counts(season_id = 35,
                            queue_id = 202,
                            team_type = 1,
                            league_id = 0,
                            host_region = "kr")
})
get_league_data

Arguments

- **season_id**: A numeric argument indicating a particular ladder season. Currently, league data is only available for season 28 and higher (i.e. data prior to this season is inaccessible).
- **queue_id**: 1 = WoL 1v1, 2 = WoL 2v2, 3 = WoL 3v3, 4 = WoL 4v4, 101 = HotS 1v1, 102 = HotS 2v2, 103 = HotS 3v3, 104 = HotS 4v4, 201 = LotV 1v1, 202 = LotV 2v2, 203 = LotV 3v3, 204 = LotV 4v4, 206 = LotV Archon
- **team_type**: 0 = Arranged, 1 = Random
- **league_id**: 0 = Bronze, 1 = Silver, 2 = Gold, 3 = Platinum, 4 = Diamond, 5 = Masters, 6 = Grandmaster
- **host_region**: The host region that the API call will be sent to. For get_league_data, the host region affects the data you will receive (i.e. different regions will result in different data). Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

Note

League data is only available for season 28 and higher.

References

- Blizzard Community API Documentation
- Regionality and APIs

See Also

Other game data API calls: get_ladder_data(), get_season_data()
get_legacy_achievements

Examples

# Get full league data for Season 30, LotV 1v1, arranged teams, # Masters league, U.S. region.
try({
  data <- get_league_data(season_id = 30,
                           queue_id = 201,
                           team_type = 0,
                           league_id = 5,
                           host_region = "us")
})

get_legacy_achievements

Legacy API - Available Achievements

Description

Provides a listing of available achievements in Starcraft II.

Usage

get_legacy_achievements(region_id, host_region = "us")

Arguments

region_id A numeric argument indicating the region of the profile.
  • 1 = US Region
  • 2 = EU Region
  • 3 = KR/TW Region
  • 5 = CN Region

host_region The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

Note

Legacy API call. It is recommended to use get_static instead.

References

• Blizzard Community API Documentation
• Regionality and APIs
get_legacy_ladder

See Also

- get_static

Other legacy API calls:
- get_legacy_ladders()
- get_legacy_ladder()
- get_legacy_match_history()
- get_legacy_profile()
- get_legacy_rewards()

---

get_legacy_ladder  Legacy API - Ladder Details

Description

Provides a listing of players in a given ladder. Also provides other information such as their ladder record, points, profile information, and clan.

Usage

get_legacy_ladder(region_id, ladder_id, host_region = "us")

Arguments

- **region_id**: A numeric argument indicating the region of the profile.
  - 1 = US Region
  - 2 = EU Region
  - 3 = KR/TW Region
  - 5 = CN Region
- **ladder_id**: A unique identifier for a particular ladder. With the exception of Grandmaster, leagues (bronze, silver, etc.), are separated into tiers (1,2,3) which are further separated into divisions. These divisions, or ladders, each have a unique identifier.
- **host_region**: The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

Note

Legacy API call. It is recommended to use get_ladder_data instead.

References

- Blizzard Community API Documentation
- Regionality and APIs

See Also

- get_ladder_data

Other legacy API calls:
- get_legacy_achievements()
- get_legacy_ladders()
- get_legacy_match_history()
- get_legacy_profile()
- get_legacy_rewards()
get_legacy_ladders Legacy API - Profile Ladder Summary

Description
Provides information about a profile’s performance in the current season, previous season, and showcase entries.

Usage
get_legacy_ladders(region_id, realm_id, profile_id, host_region = "us")

Arguments
region_id A numeric argument indicating the region of the profile.
- 1 = US Region
- 2 = EU Region
- 3 = KR/TW Region
- 5 = CN Region
realm_id A numeric argument indicating the realm of the profile. A realm is a subset of the region.
- US Region
  - 1 = US
  - 2 = LatAm
- EU Region
  - 1 = Europe
  - 2 = Russia
- KR/TW Region
  - 1 = Korea
  - 2 = Taiwan
profile_id A unique, numeric identifier for an individual’s profile.
host_region The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

Note
Legacy API call. For similar information, use get_ladder_summary.

References
- Blizzard Community API Documentation
- Regionality and APIs
get_legacy_match_history

See Also

get_ladder_summary

Other legacy API calls: get_legacy_achievements(), get_legacy_ladder(), get_legacy_match_history(), get_legacy_profile(), get_legacy_rewards()

get_legacy_match_history

Legacy API - Profile Match History

Description

Provides information about a profile’s recent match history (last 25 matches, win/loss, timestamp, etc.).

Usage

get_legacy_match_history(region_id, realm_id, profile_id, host_region = "us")

Arguments

region_id

A numeric argument indicating the region of the profile.

• 1 = US Region
• 2 = EU Region
• 3 = KR/TW Region
• 5 = CN Region

realm_id

A numeric argument indicating the realm of the profile. A realm is a subset of the region.

• US Region
  – 1 = US
  – 2 = LatAm
• EU Region
  – 1 = Europe
  – 2 = Russia
• KR/TW Region
  – 1 = Korea
  – 2 = Taiwan

profile_id

A unique, numeric identifier for an individual’s profile.

host_region

The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.
**get_legacy_profile**

**Note**

Although this is a legacy API call, there is no other call available to obtain a profile’s match history.

**References**

- Blizzard Community API Documentation
- Regionality and APIs

**See Also**

Other legacy API calls: `get_legacy_achievements()`, `get_legacy_ladders()`, `get_legacy_ladder()`, `get_legacy_profile()`, `get_legacy_rewards()`

**Examples**

```r
# Obtain recent 1v1 results for a profile in the U.S. region
try({
  matches <- get_legacy_match_history(region_id = 1, realm_id = 1, profile_id = 4716773)
  matches[matches$type=="1v1","decision"]
})
```

---

**get_legacy_profile**  
*Legacy API - Profile Data*

**Description**

Provides summary data for an individual’s profile such as campaign completion, career ladder finishes, earned achievements, and much more.

**Usage**

```r
get_legacy_profile(region_id, realm_id, profile_id, host_region = "us")
```

**Arguments**

- **region_id**: A numeric argument indicating the region of the profile.
  - 1 = US Region
  - 2 = EU Region
  - 3 = KR/TW Region
  - 5 = CN Region

- **realm_id**: A numeric argument indicating the realm of the profile. A realm is a subset of the region.
  - US Region
    - 1 = US
get_legacy_rewards

- 2 = LatAm
- EU Region
  - 1 = Europe
  - 2 = Russia
- KR/TW Region
  - 1 = Korea
  - 2 = Taiwan

profile_id A unique, numeric identifier for an individual’s profile.

host_region The host region that the API call will be sent to. For most API calls, the same
data will be returned regardless of which region the request is sent to. Must be
one of “us”, “eu”, “kr”, “tw”, “cn”. For more information on regionality, refer to
Regionality and APIs.

Note
Legacy API call. It is recommended to use get_profile instead.

References
- Blizzard Community API Documentation
- Regionality and APIs

See Also
get_profile

Other legacy API calls: get_legacy_achievements(), get_legacy_ladders(), get_legacy_ladder(),
get_legacy_match_history(), get_legacy_rewards()
get_metadata

Profile Metadata

Description
Provides metadata for an individual’s profile including their display name, profile URL, and avatar URL.

Usage
get_metadata(region_id, realm_id, profile_id, host_region = "us")

Arguments
region_id  A numeric argument indicating the region of the profile.
• 1 = US Region
• 2 = EU Region
• 3 = KR/TW Region
• 5 = CN Region

host_region  The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

Note
Legacy API call. It is recommended to use get_static instead.

References
• Blizzard Community API Documentation
• Regionality and APIs

See Also
get_static
Other legacy API calls: get_legacy_achievements(), get_legacy_ladders(), get_legacy_ladder(), get_legacy_match_history(), get_legacy_profile()
get_player

Description
Provides metadata for an individual’s account including a list of profiles associated with the account, as well as their respective display names, profile URLs and avatar URLs.

Usage
get_player(account_id, host_region = "us")
get_profile

**Arguments**

- **account_id**: A unique identifier for an individual’s account.
- **host_region**: The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

**References**

- Blizzard Community API Documentation
- Regionality and APIs

---

**get_profile**

**Profile Data**

---

**Description**

Provides summary data for an individual’s profile such as campaign completion, career ladder finishes, earned achievements, and more.

**Usage**

```python
get_profile(region_id, realm_id, profile_id, host_region = "us")
```

**Arguments**

- **region_id**: A numeric argument indicating the region of the profile.
  - 1 = US Region
  - 2 = EU Region
  - 3 = KR/TW Region
  - 5 = CN Region
- **realm_id**: A numeric argument indicating the realm of the profile. A realm is a subset of the region.
  - US Region
    - 1 = US
    - 2 = LatAm
  - EU Region
    - 1 = Europe
    - 2 = Russia
  - KR/TW Region
    - 1 = Korea
    - 2 = Taiwan
- **profile_id**: A unique, numeric identifier for an individual’s profile.
get_season

The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

References

- Blizzard Community API Documentation
- Regionality and APIs

See Also

Other profile API calls: get_ladder_summary(), get_ladder(), get_metadata(), get_static()

Examples

```r
# Request profile summary of a particular profile in the U.S. region and U.S. realm.
try(get_profile(region_id = 1, realm_id = 1, profile_id = 4716773))
```

---

get_season  

*Current Season Information*

Description

Provides the current season ID, starting date, and ending date.

Usage

```r
get_season(region_id, host_region = "us")
```

Arguments

- **region_id**: A numeric argument indicating the region of the profile.
  - 1 = US Region
  - 2 = EU Region
  - 3 = KR/TW Region
  - 5 = CN Region

- **host_region**: The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

References

- Blizzard Community API Documentation
- Regionality and APIs
get_season_data

See Also

Other ladder data API calls: `get_gm_leaderboard()`

Examples

```r
# Obtain current season information for the European region
try(get_season(region_id = 2))
```

---

**get_season_data**  
*Season Data*

**Description**

Provides start and ending times for a given season.

**Usage**

```r
get_season_data(season_id, host_region = "us")
```

**Arguments**

- `season_id`  
  A numeric argument indicating a particular ladder season. Currently, league data is only available for season 28 and higher (i.e. data prior to this season is inaccessible).

- `host_region`  
  The host region that the API call will be sent to. For `get_season_data`, the host region affects the data you will receive (i.e. different regions will result in different data). Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

**Note**

Season data is only available for season 28 and higher.

**References**

- [Blizzard Community API Documentation](#)
- [Regionality and APIs](#)

**See Also**

Other game data API calls: `get_ladder_data()`, `get_league_data()`
get_static

Examples

# Get season start and end times for season 35 in the European region.
try{
  data <- get_season_data(season_id = 35, host_region = "eu")
  as.POSIXct(data$start_timestamp, origin = "1970-01-01")
  as.POSIXct(data$end_timestamp, origin = "1970-01-01")
}

get_static Static Profile Data

Description
Provides static information (achievements, categories, criteria, and rewards) about SC2 profiles in a given region.

Usage
get_static(region_id, host_region = "us")

Arguments

region_id A numeric argument indicating the region of the profile.
  • 1 = US Region
  • 2 = EU Region
  • 3 = KR/TW Region
  • 5 = CN Region

host_region The host region that the API call will be sent to. For most API calls, the same data will be returned regardless of which region the request is sent to. Must be one of "us", "eu", "kr", "tw", "cn". For more information on regionality, refer to Regionality and APIs.

References

• Blizzard Community API Documentation

See Also
Other profile API calls: get_ladder_summary(), get_ladder(), get_metadata(), get_profile()
Examples

# Request static data of profiles in the EU region. Request is sent through
# the U.S. host region.
try(get_static(region_id = 2, host_region = "us"))

# Request static data of profiles in the China region. The request must be
# sent to the China gateway.
try(get_static(region_id = 5, host_region = "cn"))

---

SC2API SC2API package

---

Description

A simple wrapper written in R for Blizzard’s Starcraft II API. Enables users to retrieve various data pertaining to accounts, profiles, seasons and ladders.

Details

Before using, it is recommended to view the README on GitHub
Further documentation can be found in Blizzard’s Starcraft II API documentation.
Index

* **game data API calls**
  - get_ladder_data, 6
  - get_league_data, 12
  - get_season_data, 24

* **ladder data API calls**
  - get_gm_leaderboard, 3
  - get_season, 23

* **legacy API calls**
  - get_legacy_achievements, 14
  - get_legacy_ladder, 15
  - get_legacy_ladders, 16
  - get_legacy_match_history, 17
  - get_legacy_profile, 18
  - get_legacy_rewards, 19

* **profile API calls**
  - get_ladder, 4
  - get_ladder_summary, 8
  - get_metadata, 20
  - get_profile, 22
  - get_static, 25

authorization, 2

get_gm_leaderboard, 3, 24
get_ladder, 4, 9, 21, 23, 25
get_ladder_data, 6, 13, 15, 24
get_ladder_ids, 7
get_ladder_summary, 5, 8, 16, 17, 21, 23, 25
get_last_played, 10
get_league_counts, 11
get_league_data, 6, 12, 24
get_legacy_achievements, 14, 15, 17–20
get_legacy_ladder, 15, 17–20
get_legacy_ladders, 15, 16, 18–20
get_legacy_match_history, 15, 17, 17, 19, 20
get_legacy_profile, 15, 17, 18, 18, 20
get_legacy_rewards, 15, 17–19, 19
get_metadata, 5, 9, 20, 23, 25
get_player, 21

get_profile, 5, 9, 19, 21, 22, 25
get_season, 4, 23
get_season_data, 6, 13, 24
get_static, 5, 9, 14, 15, 20, 21, 23, 25

SC2API, 26
set_token, 2
set_token (authorization), 2
unset_token (authorization), 2
validate_token (authorization), 2