Package ‘spotifyr’

November 2, 2021

Title  R Wrapper for the 'Spotify' Web API
Version  2.2.3
Date  2021-11-02
Description  An R wrapper for pulling data from the 'Spotify' Web API
  <https://developer.spotify.com/documentation/web-api/> in bulk, or post items on a
  'Spotify' user's playlist.
Maintainer  Daniel Antal <daniel.antal@ceemid.eu>
Depends  R (>= 3.3.0)
Imports  dplyr, purrr, tidyr, htrr, lubridate, jsonlite, readr, rvest,
  stringr, tibble, janitor, rlang, ggridges, magrittr, genius,
  assertthat, xml2
License  MIT + file LICENSE
Encoding  UTF-8
RoxygenNote  7.1.2
Suggests  testthat (>= 3.0.0)
URL  https://github.com/charlie86/spotifyr
BugReports  https://github.com/charlie86/spotifyr/issues
Config/testthat/edition  3
NeedsCompilation  no
Author  Charlie Thompson [aut],
  Daniel Antal [aut, cre] (<https://orcid.org/0000-0001-7513-6760>),
  Josiah Parry [aut] (<https://orcid.org/0000-0001-9910-865X>),
  Donal Phipps [aut],
  Tom Wolff [aut]
Repository  CRAN
Date/Publication  2021-11-02 18:00:02 UTC
R topics documented:

add_tracks_to_playlist ............................................. 3
change_playlist_details ............................................ 4
check_me_following .................................................... 5
check_users_following ............................................... 6
create_playlist ....................................................... 7
dedupe_album_names ................................................ 8
follow_artists_or_users ............................................. 9
follow_playlist ....................................................... 9
get_album ............................................................. 10
get_albums ............................................................ 11
get_album_tracks .................................................... 12
get_artist ............................................................. 13
get_artists ........................................................... 14
get_artist_albums ..................................................... 14
get_artist_audio_features ......................................... 16
get_artist_top_tracks .............................................. 17
get_categories ...................................................... 18
get_category ......................................................... 18
get_category_playlists ............................................ 19
get_discography ..................................................... 20
get_featured_playlists ............................................. 21
get_genre_artists .................................................... 22
get_label_artists .................................................... 24
get_my_currently_playing .......................................... 25
get_my_current_playback .......................................... 26
get_my_devices ....................................................... 27
get_my_followed_artists ........................................... 27
get_my_playlists ..................................................... 28
get_my_profile ....................................................... 29
get_my_recently_played ............................................ 30
get_my_saved_albums ............................................... 31
get_my_saved_tracks ............................................... 32
get_my_top_artists_or_tracks .................................... 33
get_new_releases ................................................... 34
get_playlist .......................................................... 35
get_playlist_audio_features ...................................... 36
get_playlist_cover_image ......................................... 37
get_playlist_tracks ................................................ 37
get_recommendations ............................................... 39
get_recommendations_all .......................................... 44
get_related_artists ................................................. 45
get_show .............................................................. 46
get_shows ............................................................. 46
get_show_episodes .................................................. 47
get_spotify_access_token ......................................... 48
get_spotify_authorization_code ................................. 49
**add_tracks_to_playlist**

*Add Tracks to User's Playlist*

**Description**

Add one or more tracks to a user’s playlist.

**Usage**

```python
add_tracks_to_playlist(
    playlist_id,
    uris,
    position = NULL,
    authorization = get_spotify_authorization_code()
)
```
Arguments

- **playlist_id**: Required. The Spotify ID for the playlist.
- **uris**: Optional. A character vector of Spotify track URIs to add. For example: 
  
  ```r
  uris = c("61H97kuKpi6kJQRnUEIlh", "2q6vlgBJ432KeZZNt2ZZBV")
  ```
  A maximum of 100 tracks can be added in one request. The uris will be formed as uris = c("spotify:track:" uris_prefix in your vector it will not be duplicated, otherwise it will be added.
- **position**: Optional. Integer indicating the position to insert the tracks, a zero-based index. For example, to insert the tracks in the first position: `position = 0`; to insert the tracks in the third position: `position = 2`. If omitted, the tracks will be appended to the playlist. Tracks are added in the order they are listed in the query string or request body.
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user.

Adding tracks to the current user’s public playlists requires authorization of the playlist-modify-public scope; adding tracks to the current user’s private playlist (including collaborative playlists) requires the playlist-modify-private scope. See Using Scopes.

Value

The response from the Spotify Web API on the POST request, with a meaningful error message if the operation was not successful.

See Also

Other playlist functions: `change_playlist_details()`, `check_users_following()`, `create_playlist()`, `get_my_playlists()`, `get_playlist()`, `get_user_playlists()`, `remove_tracks_from_playlist()`, `tidy()`

---

**change_playlist_details**

*Change a playlist’s name and public/private state. (The user must, of course, own the playlist.)*

---

Description

Change a playlist’s name and public/private state. (The user must, of course, own the playlist.)

Usage

```r
change_playlist_details(
  playlist_id, name = NULL, public = NULL,
)```
check_me_following

    collaborative = NULL,
    description = NULL,
    authorization = get_spotify_authorization_code()
)

Arguments

playlist_id Required. The Spotify ID for the playlist.
name Optional String containing the name for the new playlist, for example “Your Coolest Playlist”. This name does not need to be unique; a user may have several playlists with the same name.
public Optional. Boolean. If TRUE the playlist will be public. If FALSE it will be private.
collaborative Optional. Boolean. If TRUE the playlist will become collaborative and other users will be able to modify the playlist in their Spotify client. Note: you can only set collaborative to TRUE on non-public playlists.
description Optional. String containing the playlist description as displayed in Spotify Clients and in the Web API.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to get_spotify_authorization_code(). The access token must have been issued on behalf of the current user.

Value

The response from the Spotify Web API on the PUT request, with a meaningful error message if the operation was not successful.

See Also

Other playlist functions: add_tracks_to_playlist(), check_users_following(), create_playlist(), get_my_playlists(), get_playlist(), get_user_playlists(), remove_tracks_from_playlist(), tidy()

check_me_following Check Me Following

Description

Check if the current user is following specific artists or users.

Usage

check_me_following(type, ids, authorization = get_spotify_authorization_code())
check_users_following

Arguments

- **type**: Required. String of the ID type: either "artist" or "user".
- **ids**: Required. A character vector of the artist or the user Spotify IDs to check. For example: `ids = c("74ASZWbe4lXaubB36ztrGX","08td7MxkoHQxWnAYD8dQ")`. A maximum of 50 IDs can be sent in one request.
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Getting details of the artists or users the current user follows requires authorization of the user-follow-read scope. See Using Scopes.

Value

A data frame containing three columns, "type", the "id" of the artist or users and "is_following" as a logical, boolean variable.

---

check_users_following  Check if Users Follow a Playlist

Description

Check if particular users are following a particular playlist.

Usage

```r
check_users_following(
  playlist_id,
  ids,
  authorization = get_spotify_authorization_code()
)
```

Arguments

- **playlist_id**: Required. The Spotify ID of the playlist.
- **ids**: Required. Spotify User IDs; the ids of the users that you want to check to see if they follow the playlist. Maximum: 5 ids.
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Following a playlist can be done publicly or privately. Checking if a user publicly follows a playlist doesn’t require any scopes; if the user is publicly following the playlist, this endpoint returns TRUE. Checking if the user is privately following a playlist is only possible for the current user when that user has granted access to the playlist-read-private scope. See Using Scopes.
create_playlist

Value

A data frame containing three columns, "user_id" for the particular users, "playlist_id" for the particular playlists, "is_following" as a logical, boolean variable.

See Also

Other playlist functions: add_tracks_to_playlist(), change_playlist_details(), create_playlist(), get_my_playlists(), get_playlist(), get_user_playlists(), remove_tracks_from_playlist(), tidy()

create_playlist Create Playlist for User

Description

Create a playlist for a Spotify user. The playlist will be empty until you add tracks.

Usage

create_playlist(
  user_id,
  name,
  public = TRUE,
  collaborative = FALSE,
  description = NULL,
  authorization = get_spotify_authorization_code()
)

Arguments

user_id Required. The user’s Spotify user ID.
name Required. String containing the name for the new playlist, for example "Your Coolest Playlist". This name does not need to be unique; a user may have several playlists with the same name.
public Optional. Boolean. Defaults to TRUE. If TRUE the playlist will be public. If FALSE it will be private. To be able to create private playlists, the user must have granted the playlist-modify-private scope.
collaborative Optional. Boolean. Defaults to FALSE. If TRUE the playlist will be collaborative. Note that to create a collaborative playlist you must also set public to FALSE. To create collaborative playlists you must have granted playlist-modify-private and playlist-modify-public scopes.
description Optional. String containing the playlist description as displayed in Spotify Clients and in the Web API.
authorization

Required. A valid access token from the Spotify Accounts service. See the Web
API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user.

Creating a public playlist for a user requires authorization of the playlist-modify-public scope; creating a private playlist requires the playlist-modify-private scope. See Using Scopes.

Value

The response from the Spotify Web API on the POST request, with a meaningful error message if the operation was not successful.

See Also

Other playlist functions: `add_tracks_to_playlist()`, `change_playlist_details()`, `check_users_following()`, `get_my_playlists()`, `get_playlist()`, `get_user_playlists()`, `remove_tracks_from_playlist()`, `tidy()`

---

dedupe_album_names  Remove duplicate album names

Description

Use fuzzy matching to remove duplicate album names (including reissues, remasters, etc).

Usage

```r
dedupe_album_names(
  df,
  album_name_col = "album_name",
  album_release_year_col = "album_release_year"
)
```

Arguments

- `df`  Data frame with album name
- `album_name_col`  String of field name containing album names
- `album_release_year_col`  String of field name containing album release year

Value

The original data frame with distinct `album_name` rows, keeping as much as possible the original album release (and not re-releases.)
follow_artists_or_users

Add Current User as Follower Artists or Other Users

Description

Add the current user as a follower of one or more artists or other Spotify users.

Usage

```
follow_artists_or_users(
  type,
  ids = NULL,
  authorization = get_spotify_authorization_code()
)
```

Arguments

- **type**: Required. The ID type: either "artist" or "user".
- **ids**: Optional. A character vector of the artist or the user Spotify IDs. For example: `ids = c("74ASZWbe41XaubB36ztrGX","08td7MxkoHQkXnWAYD6Q")`. A maximum of 50 IDs can be sent in one request.
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Modifying the list of artists or users the current user follows requires authorization of the user-follow-modify scope. See Using Scopes.

Value

The response from the Spotify Web API on the PUT request, with a meaningful error message if the operation was not successful.

follow_playlist

Add Current User to Followers of Playlist.

Description

Add Current User to Followers of Playlist.

Usage

```
follow_playlist(
  playlist_id,
  public = FALSE,
  authorization = get_spotify_authorization_code()
)
```
Arguments

playlist_id  Required. The Spotify ID of the playlist. Any playlist can be followed, regardless of its public/private status, as long as you know its playlist ID.

public  Optional. Defaults to TRUE. If TRUE the playlist will be included in the user’s public playlists, if FALSE it will remain private. To be able to follow playlists privately, the user must have granted the playlist-modify-private scope.

authorization  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to 

The access token must have been issued on behalf of the current user. Following a publicly followed playlist for a user requires authorization of the playlist-modify-public scope; following a privately followed playlist requires the playlist-modify-private scope. See Using Scopes.

Note that the scopes you provide relate only to whether the current user is following the playlist publicly or privately (i.e. showing others what they are following), not whether the playlist itself is public or private.

Value

The response from the Spotify Web API on the PUT request, with a meaningful error message if the operation was not successful.

get_album

Get Spotify catalog information for a single album.

Description

Get Spotify catalog information for a single album.

Usage

get_album(id, market = NULL, authorization = get_spotify_access_token())

Arguments

id  The Spotify ID for the album.

market  Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking

authorization  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to 

Value

Returns a data frame of results containing album data. See the official documentation for more information.
get_albums

See Also

Other album functions: get_albums(), get_discography()

get_albums

Get Spotify catalog information for multiple albums identified by their Spotify IDs.

Description

Get Spotify catalog information for multiple albums identified by their Spotify IDs.

Usage

get_albums(
  ids,
  market = NULL,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)

Arguments

ids Required. A character vector of the Spotify IDs for the albums. Maximum: 20 IDs.

market Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking

authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotifyr::get_spotify_access_token()

include_meta_info Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value


See Also

Other album functions: get_album(), get_discography()
get_album_tracks

Get Spotify catalog information about an album's tracks. Optional parameters can be used to limit the number of tracks returned.

Description

Get Spotify catalog information about an album's tracks. Optional parameters can be used to limit the number of tracks returned.

Usage

```r
get_album_tracks(
  id,
  limit = 20,
  offset = 0,
  market = NULL,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)
```

Arguments

- `id` Required. The Spotify ID for the album.
- `limit` Optional. Maximum number of results to return. Default: 20 Minimum: 1 Maximum: 50 Note: The limit is applied within each type, not on the total response. For example, if the limit value is 3 and the type is c("artist","album"), the response contains 3 artists and 3 albums.
- `offset` Optional. The index of the first album to return. Default: 0 (the first album). Maximum offset (including limit): 10,000. Use with limit to get the next page of albums.
- `market` Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking
- `authorization` Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`
- `include_meta_info` Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to `FALSE`. 
get_artist

Value

Returns a data frame of results containing album data. See the official API documentation for more information.

See Also

Other track functions: get_playlist_tracks(), get_tracks(), get_track()

get_artist(id, authorization = get_spotify_access_token())

Arguments

id The Spotify ID for the artist.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotifyr::get_spotify_access_token()

Value


See Also

Other artist functions: get_artist_albums(), get_artist_top_tracks(), get_artists(), get_my_followed_artists(), get_related_artists()
get_artists

Get Spotify catalog information for multiple artists identified by their Spotify IDs.

Description

Get Spotify catalog information for multiple artists identified by their Spotify IDs.

Usage

get_artists(
  ids,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)

Arguments

ids Required. A character vector of the Spotify IDs for the artists. Maximum: 50 IDs.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.
include_meta_info Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value


See Also

Other artist functions: `get_artist_albums()`, `get_artist_top_tracks()`, `get_artist()`, `get_my_followed_artists()`, `get_related_artists()`
get_artist_albums

Usage

get_artist_albums(
  id,
  include_groups = c("album", "single", "appears_on", "compilation"),
  market = NULL,
  limit = 20,
  offset = 0,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)

Arguments

id                               The Spotify ID for the artist.
include_groups                   Optional. A character vector of keywords that will be used to filter the response. If not supplied, all album types will be returned. Valid values are: "album" "single" "appears_on" "compilation"
                               For example: include_groups = c("album", "single")
market                           Optional. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()
                               An ISO 3166-1 alpha-2 country code or the string "from_token". Supply this parameter to limit the response to one particular geographical market. For example, for albums available in Sweden: market = "SE". If not given, results will be returned for all markets and you are likely to get duplicate results per album, one for each market in which the album is available!
limit                            Optional. Maximum number of results to return.
                               Default: 20 Minimum: 1 Maximum: 50
offset                           Optional. The index of the first album to return.
                               Default: 0 (i.e., the first album).
                               Use with limit to get the next set of albums.
authorization                    Required. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.
include_meta_info                Optional.

Value

get_artist_audio_features

Get Audio Features For Artists’ Discography

Description

Get audio feature information for all or part of an artists’ discography.

Usage

```r
get_artist_audio_features(
  artist = NULL,
  include_groups = "album",
  return_closest_artist = TRUE,
  dedupe_albums = TRUE,
  market = NULL,
  authorization = get_spotify_access_token()
)
```

Arguments

- **artist** Required. String of either an artist name or an artist Spotify ID. If an artist name is provided, `search_spotify()` will be used to find a Spotify ID matching the name provided.
- **include_groups** Optional. A character vector of keywords that will be used to filter the response. Defaults to "album". Valid values are: "album" "single" "appears_on" "compilation"
- **return_closest_artist** Optional. Boolean.
- **dedupe_albums** Optional. Logical, boolean parameter, defaults to TRUE.
- **market** Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". An ISO 3166-1 alpha-2 country code or the string "from_token". Supply this parameter to limit the response to one particular geographical market. For example, for albums available in Sweden: market = "SE". If not given, results will be returned for all markets and you are likely to get duplicate results per album, one for each market in which the album is available!
- **authorization** Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`
**get_artist_top_tracks**

*Get Spotify catalog information about an artist’s top tracks by country.*

**Value**

Returns a data frame of results containing track audio features data. See the [Spotify Web API documentation](https://developer.spotify.com/documentation/web-api/reference/artists/get-several-artists/) for more information.

**See Also**

Other musicology functions: `get_genre_artists()`, `get_playlist_audio_features()`, `get_track_audio_analysis()`, `get_track_audio_features()`, `get_user_audio_features()`

```r
get_artist_top_tracks
```

**Description**

Get Spotify catalog information about an artist’s top tracks by country.

**Usage**

```r
get_artist_top_tracks(
  id,
  market = "US",
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)
```

**Arguments**

- `id`: The Spotify ID for the artist.
- `market`: Required. An ISO 3166-1 alpha-2 country code or the string "from_token". Defaults to "US".
- `include_meta_info`: Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to `FALSE`.

**Value**


**See Also**

Other artist functions: `get_artist_albums()`, `get_artists()`, `get_artist()`, `get_my_followed_artists()`, `get_related_artists()`
#### get_categories

Get a list of Spotify categories

**Usage**

```r
get_categories(authorization = get_spotify_access_token(), df = TRUE)
```

**Arguments**

- `df`: Should the results be formatted as a data frame? If `FALSE`, the full response JSON will be returned as a list; defaults to `TRUE`.

**Value**

Returns a data frame of results containing album data. See https://developer.spotify.com/documentation/web-api/reference/browse/get-list-categories/ for more information.

#### get_category

Get a single category used to tag items

**Description**

Get a single category used to tag items in Spotify (on, for example, the Spotify player’s “Browse” tab).

**Usage**

```r
get_category(
  category_id,
  country = NULL,
  locale = NULL,
  authorization = get_spotify_access_token()
)
```
get_category_playlists

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>category_id</td>
<td>Required. The Spotify ID for the category.</td>
</tr>
<tr>
<td>country</td>
<td>Optional. A country: an ISO 3166-1 alpha-2 country code. Provide this parameter to ensure that the category exists for a particular country.</td>
</tr>
<tr>
<td>locale</td>
<td>Optional. The desired language, consisting of an ISO 639-1 language code and an ISO 3166-1 alpha-2 country code, joined by an underscore. For example: es_MX, meaning &quot;Spanish (Mexico)&quot;. Provide this parameter if you want the category strings returned in a particular language. Note that, if locale is not supplied, or if the specified language is not available, the category strings returned will be in the Spotify default language (American English).</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to <code>spotifyr::get_spotify_access_token()</code></td>
</tr>
</tbody>
</table>

Value


get_category_playlists

Get a list of Spotify playlists tagged with a particular category

Description

Get a list of Spotify playlists tagged with a particular category

Usage

```r
get_category_playlists(
  category_id = "party",
  country = NULL,
  limit = 20,
  offset = 0,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>category_id</td>
<td>Required. The Spotify ID for the category.</td>
</tr>
</tbody>
</table>
get_discography

Retrieve Artist Discography with Song Lyrics and Audio Info

Description

Retrieve the entire discography of an artist with the lyrics of each song and the associated audio information. Returns the song data as a nested tibble. This way we can easily see each album, artist, and song title before expanding our data.

Usage

get_discography(artist, authorization = get_spotify_access_token())

Arguments

- **artist**
  - The quoted name of the artist. Spelling matters, capitalization does not.
- **authorization**
  - Authorization token for Spotify web API. Defaults to get_spotify_access_token().

Value

A nested tibble. See tidyr::nest.

See Also

Other album functions: get_albums(), get_album()
get_featured_playlists

Examples

rex_orange <- get_discography("Rex Orange County")

get_featured_playlists

Get list of Spotify featured playlists

Description

Get a list of Spotify featured playlists (as shown, for example, on a Spotify player’s ‘Browse’ tab)

Usage

get_featured_playlists(
    locale = NULL,
    country = NULL,
    timestamp = NULL,
    limit = 20,
    offset = 0,
    authorization = get_spotify_access_token(),
    include_meta_info = FALSE
)

Arguments

locale Optional. The desired language, consisting of an ISO 639-1 language code and an ISO 3166-1 alpha-2 country code, joined by an underscore. For example: es_MX, meaning "Spanish (Mexico)". Provide this parameter if you want the category strings returned in a particular language. Note that, if locale is not supplied, or if the specified language is not available, the category strings returned will be in the Spotify default language (American English). The locale parameter, combined with the country parameter, may give odd results if not carefully matched. For example country=SE&locale=de_DE will return a list of categories relevant to Sweden but as German language strings.

country Optional. A country: an ISO 3166-1 alpha-2 country code. Provide this parameter if you want the list of returned items to be relevant to a particular country. If omitted, the returned items will be relevant to all countries.

timestamp Optional. A timestamp in ISO 8601 format: yyyy-MM-ddTHH:mm:ss. Use this parameter to specify the user’s local time to get results tailored for that specific date and time in the day. If not provided, the response defaults to the current UTC time. Example: “2014-10-23T09:00:00” for a user whose local time is 9AM. If there were no featured playlists (or there is no data) at the specified time, the response will revert to the current UTC time.
get_genre_artists

limit

offset
Optional. The index of the first item to return. Defaults to 0, i.e., the first object. Use with limit to get the next set of items.

authorization
Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()

include_meta_info
Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value


Examples

```r
## Get Flemish-Dutch playlists from Belgium:

get_featured_playlists(
  country = 'BE',
  locale = 'nl_BE'
)
```

get_genre_artists

Search for Artists by Genre

Description

Get Spotify Catalog information about artists belonging to a given genre.

Usage

```r
get_genre_artists(
  genre = character(),
  market = NULL,
  limit = 20,
  offset = 0,
  authorization = get_spotify_access_token()
)
```
get_genre_artists

Arguments

genre
Required.
String of genre name to search for
For example: genre = "wonky".

market
Optional.
An ISO 3166-1 alpha-2 country code or the string "from_token".
If a country code is specified, only artists with content that is playable in that
market is returned.
Note:
- If market is set to "from_token", and a valid access token is specified in the
  request header, only content playable in the country associated with the user
  account is returned.
- Users can view the country that is associated with their account in the account
  settings. A user must grant access to the user-read-private scope prior to when
  the access token is issued.

limit
Optional.
Maximum number of results to return.
Default: 20
Minimum: 1
Maximum: 50

offset
Optional.
The index of the first result to return.
Default: 0 (the first result).
Maximum offset (including limit): 10,000.
Use with limit to get the next page of search results.

authorization
Required. A valid access token from the Spotify Accounts service. See the Web
API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()

Value
A data frame of the artists belonging to the genre with data and metadata about the artists in a tibble.

See Also
Other musicology functions: get_artist_audio_features(), get_playlist_audio_features(),
get_track_audio_analysis(), get_track_audio_features(), get_user_audio_features()

Examples

get_genre_artists('wonky')
get_label_artists  

**Search for artists by label**

**Description**

Get Spotify Catalog information about artists belonging to a given label.

**Usage**

```r
get_label_artists(
  label = character(),
  market = NULL,
  limit = 20,
  offset = 0,
  authorization = get_spotify_access_token()
)
```

**Arguments**

- **label**: Required.
  String of label name to search for
  For example: `label = "brainfeeder"`.

- **market**: Optional.
  An ISO 3166-1 alpha-2 country code or the string "from_token".
  If a country code is specified, only artists with content that is playable in that market is returned.
  Note:
  - If market is set to "from_token", and a valid access token is specified in the request header, only content playable in the country associated with the user account is returned.
  - Users can view the country that is associated with their account in the account settings. A user must grant access to the user-read-private scope prior to when the access token is issued.

- **limit**: Optional.
  Maximum number of results to return.
  Default: 20
  Minimum: 1
  Maximum: 50

- **offset**: Optional.
  The index of the first result to return.
  Default: 0 (the first result).
  Maximum offset (including limit): 10,000.
  Use with limit to get the next page of search results.

- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`
get_my_currently_playing

Value

A data frame with the label information of the artists.

Examples

get_label_artists('brainfeeder')

get_my_currently_playing

Get the object currently being played on the user’s Spotify account.

Description

Get the object currently being played on the user’s Spotify account.

Usage

get_my_currently_playing(
  market = NULL,
  authorization = get_spotify_authorization_code()
)

Arguments

market An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking

authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to get_spotify_access_token(). The access token must have been issued on behalf of the current user.

The access token must have the user-read-currently-playing and/or user-read-playback-state scope authorized in order to read information.

Value

Returns a data frame of results containing user profile information. See for more information.

See Also

Other player functions: get_my_current_playback(), get_my_devices(), get_my_recently_played(), pause_my_playback(), seek_to_position(), set_my_repeat_mode(), skip_my_playback_previous(), skip_my_playback(), start_my_playback(), toggle_my_shuffle(), transfer_my_playback()
get_my_current_playback

Get information about the user’s current playback state, including track, track progress, and active device.

Description

Get information about the user’s current playback state, including track, track progress, and active device.

Usage

```r
get_my_current_playback(
    market = NULL,
    authorization = get_spotify_authorization_code()
)
```

Arguments

- `market` An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking
- `authorization` Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-read-playback-state` scope authorized in order to read information.

Value

Returns a list containing user playback information. See the official Spotify Web API documentation for more information.

See Also

Other player functions: `get_my_currently_playing()`, `get_my_devices()`, `get_my_recently_played()`, `pause_my_playback()`, `seek_to_position()`, `set_my_repeat_mode()`, `skip_my_playback_previous()`, `skip_my_playback()`, `start_my_playback()`, `toggle_my_shuffle()`, `transfer_my_playback()`
### get_my_devices

Get information about a user's available devices.

#### Description

Get information about a user's available devices.

#### Usage

```r
get_my_devices(authorization = get_spotify_authorization_code())
```

#### Arguments

- `authorization` Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-read-playback-state` scope authorized in order to read information.

#### Value

Returns a data frame of results containing user device information. See the official Spotify Web API documentation for more information.

#### See Also

Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_recently_played()`, `pause_my_playback()`, `seek_to_position()`, `set_my_repeat_mode()`, `skip_my_playback_previous()`, `skip_my_playback()`, `start_my_playback()`, `toggle_my_shuffle()`, `transfer_my_playback()`

---

### get_my_followed_artists

Get My Followed Artists

#### Description

Get the current user's followed artists.

#### Usage

```r
get_my_followed_artists(
  limit = 20,
  after = NULL,
  authorization = get_spotify_authorization_code(),
  include_meta_info = FALSE
)
```
Arguments

**limit**  

**after**  
Optional. The last artist ID retrieved from the previous request.

**authorization**  
Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Getting details of the artists or users the current user follows requires authorization of the `user-follow-read` scope. See Using Scopes.

**include_meta_info**  
Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value

A data frame containing my followed artists with some of full information about the followed artists.

See Also

Other artist functions: `get_artist_albums()`, `get_artist_top_tracks()`, `get_artists()`, `get_artist()`, `get_related_artists()`

---

**get_my_playlists**  
*Get List of My Playlists*

Description

Get a list of the playlists owned or followed by the current Spotify user.

Usage

```r
get_my_playlists(
  limit = 20,
  offset = 0,
  authorization = get_spotify_authorization_code(),
  include_meta_info = FALSE
)
```

Arguments

**limit**  
Optional.  
Maximum number of playlists to return.  
Default: 20  
Minimum: 1  
Maximum: 50
**get_my_profile**

Get User Profile Information

**Description**

Get detailed profile information about the current user (including the current user’s username).

**Usage**

```r
get_my_profile(authorization = get_spotify_authorization_code())
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
</table>
| `authorization`   | Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`.
|                   | The access token must have been issued on behalf of the current user. Reading the user’s email address requires the `user-read-email` scope; reading country and product subscription level requires the `user-read-private` scope. See Using Scopes. |
get_my_recently_played

Value


get_my_recently_played

Get Current User’s Recently Played Tracks

Description

Get Current User’s Recently Played Tracks

Usage

get_my_recently_played(
  limit = 20,
  after = NULL,
  before = NULL,
  authorization = get_spotify_authorization_code(),
  include_meta_info = FALSE
)

Arguments

limit


after

Optional. A Unix timestamp in milliseconds. Returns all items after (but not including) this cursor position. If after is specified, before must not be specified.

before

Optional. A Unix timestamp in milliseconds. Returns all items before (but not including) this cursor position. If before is specified, after must not be specified.

authorization

Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token(). The access token must have been issued on behalf of the current user.

include_meta_info

Optional. Boolean indicating whether to include full result, with meta information such as "before", "after", and "limit". Defaults to FALSE. The access token must have the user-read-recently-played scope authorized in order to read the user’s recently played tracks.

Value

Returns a list or data frame of results containing the most recently played tracks for the current user.
get_my_saved_albums

See Also

Other player functions: get_my_current_playback(), get_my_currently_playing(), get_my_devices(), pause_my_playback(), seek_to_position(), set_my_repeat_mode(), skip_my_playback_previous(), skip_my_playback(), start_my_playback(), toggle_my_shuffle(), transfer_my_playback()

get_my_saved_albums Get Current User’s Saved Albums

Description

Get a list of the albums saved in the current Spotify user’s ‘Your Music’ library.

Usage

```r
get_my_saved_albums(
  limit = 20,
  offset = 0,
  market = NULL,
  authorization = get_spotify_authorization_code(),
  include_meta_info = FALSE
)
```

Arguments

- **limit**  
  Optional.  
  Maximum number of albums to return.  
  Default: 20  
  Minimum: 1  
  Maximum: 50  

- **offset**  
  Optional.  
  The index of the first albums to return.  
  Default: 0 (the first object). Maximum offset: 100,000. Use with limit to get the next set of albums.

- **market**  
  Optional.  
  An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking.

- **authorization**  
  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotifyr::get_spotify_authorization_code(). The access token must have been issued on behalf of the current user.

- **include_meta_info**  
  Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.
get_my_saved_tracks

Value


Description

Get a list of the songs saved in the current Spotify user’s ‘Your Music’ library.

Usage

get_my_saved_tracks(
  limit = 20,
  offset = 0,
  market = NULL,
  authorization = get_spotify_authorization_code(),
  include_meta_info = FALSE
)

Arguments

limit Optional.
Maximum number of tracks to return.
Default: 20
Minimum: 1
Maximum: 50

offset Optional.
The index of the first track to return.
Default: 0 (the first object). Maximum offset: 100,000. Use with limit to get the next set of tracks.

market Optional.
An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking

authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotyr::get_spotify_authorization_code(). The access token must have been issued on behalf of the current user.

include_meta_info Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.
get_my_top_artists_or_tracks

Value


get_my_top_artists_or_tracks

*Get User’s Top Artists or Tracks*

Description

Get the current user’s top artists or tracks based on calculated affinity.

Usage

```r
get_my_top_artists_or_tracks(
  type = "artists",
  limit = 20,
  offset = 0,
  time_range = "medium_term",
  authorization = get_spotify_authorization_code(),
  include_meta_info = FALSE
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Required. The type of entity to return. Defaults to <em>artists</em>, the valid alternative is <em>tracks</em>.</td>
</tr>
<tr>
<td>limit</td>
<td>Optional. Maximum number of results to return. Default: 20 Minimum: 1 Maximum: 50</td>
</tr>
<tr>
<td>offset</td>
<td>Optional. The index of the first entity to return. Default: 0 (i.e., the first track). Use with limit to get the next set of entities.</td>
</tr>
<tr>
<td>time_range</td>
<td>Optional. Over what time frame the affinities are computed. Valid values: long_term (calculated from several years of data and including all new data as it becomes available), medium_term (approximately last 6 months), short_term (approximately last 4 weeks). Default: medium_term.</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to <code>spotifyr::get_spotify_access_token()</code></td>
</tr>
<tr>
<td>include_meta_info</td>
<td>Optional. Boolean indicating whether to include full result, with meta information such as &quot;total&quot;, and &quot;limit&quot;. Defaults to FALSE.</td>
</tr>
</tbody>
</table>
get_new_releases

Value

Returns a data frame of results containing track or album data. See the official API documentation for more information.

See Also

Other personalization functions: get_recommendations_all(), get_recommendations()

get_new_releases  Get new releases

Description

Get a list of new album releases featured in Spotify (shown, for example, on a Spotify player’s “Browse” tab).

Usage

get_new_releases(
  country = NULL,
  limit = 20,
  offset = 0,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)

Arguments

country  Optional. A country: an ISO 3166-1 alpha-2 country code. Provide this parameter if you want the list of returned items to be relevant to a particular country. If omitted, the returned items will be relevant to all countries.


offset  Optional. The index of the first item to return. Default: 0 (the first object). Use with limit to get the next set of items.

authorization  Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()

include_meta_info  Optional. Boolean indicating whether to include full result, with meta information such as "country", "offset", and "limit". Defaults to FALSE.

Value

Returns a data frame of results containing new releases.

Examples

```r
## Get new Swedish music
get_new_releases(country = 'SE', limit = 5)
```

get_playlist  Get User Playlist

Description

Get a playlist owned by a Spotify user.

Usage

```r
get_playlist(
  playlist_id,
  fields = NULL,
  market = NULL,
  authorization = get_spotify_access_token()
)
```

Arguments

- `playlist_id`: Required. The Spotify ID for the playlist.
- `fields`: Optional. Filters for the query: a comma-separated list of the fields to return. If omitted, all fields are returned. For example, to get just the playlist’s description and URI:
  ```r
  fields = c("description","uri")
  ```
  A dot separator can be used to specify non-reoccurring fields, while parentheses can be used to specify reoccurring fields within objects. For example, to get just the added date and user ID of the adder:
  ```r
  fields = "tracks.items(added_at,added_by.id)"
  ```
  Use multiple parentheses to drill down into nested objects, for example:
  ```r
  fields = "tracks.items(track(name,href,album(name,href)))"
  ```
  Fields can be excluded by prefixing them with an exclamation mark, for example:
  ```r
  fields = "tracks.items(track(name,href,album(!name,href)))"
  ```
- `market`: Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking.
- `authorization`: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Both Public and Private playlists belonging to any user are retrievable on provision of a valid access token. Defaults to `spotifyr::get_spotify_access_token()`.
get_playlist_audio_features

Get Features and Popularity of Playlists on Spotify

Description

This function returns the popularity and audio features for every song for a given set of playlists on Spotify.

Usage

get_playlist_audio_features(
  username,
  playlist_uris,
  authorization = get_spotify_access_token()
)

Arguments

username String of Spotify username. Can be found on the Spotify app.
playlist_uris Character vector of Spotify playlist uris. Can be found within the Spotify App.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.

Value

A data frame with the audio featurees and popularity variables of playlists.

See Also

Other musicology functions: `get_artist_audio_features()`, `get_genre_artists()`, `get_track_audio_analysis()`, `get_track_audio_features()`, `get_user_audio_features()`.
get_playlist_cover_image

Examples

playlist_username <- 'spotify'
playlist_uri <- c('37i9dQZF1E9T1oFsQFg98K', '37i9dQZF1CyQNOI21QVf3p')
playlist_audio_features <- get_playlist_audio_features(playlist_username, playlist_uri)

Description

Get the current image associated with a specific playlist.

Usage

get_playlist_cover_image(
  playlist_id,
  authorization = get_spotify_authorization_code()
)

Arguments

playlist_id Required. The Spotify ID for the playlist.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotifyr::get_spotify_authorization_code(). The access token must have been issued on behalf of the current user. Current playlist image for both Public and Private playlists of any user are retrievable on provision of a valid access token.

Value

Returns a data frame of results containing playlist cover image information. See the official Spotify Web API Documentation for more information.

get_playlist_tracks

Description

Get full details of the tracks of a playlist owned by a Spotify user.
Usage

```r
get_playlist_tracks(
  playlist_id,
  fields = NULL,
  limit = 100,
  offset = 0,
  market = NULL,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)
```

Arguments

- **playlist_id**: Required. The Spotify ID for the playlist.
- **fields**: Optional. Filters for the query: a comma-separated list of the fields to return. If omitted, all fields are returned. For example, to get just the playlist's creation date and album information: `fields = c("added_at","track.album")`. A dot separator can be used to specify non-reoccurring fields, while parentheses can be used to specify reoccurring fields within objects. For example, to get just the added date and user ID of the adder: `fields = "tracks.items(added_at,added_by.id)"`. Use multiple parentheses to drill down into nested objects, for example: `fields = "tracks.items(track(name,href,album(name,href)))"`. Fields can be excluded by prefixing them with an exclamation mark, for example: `fields = "tracks.items(track(name,href,album(!name,href)))"`.
- **limit**: Optional. Maximum number of tracks to return. Default: 100 Minimum: 1 Maximum: 100
- **offset**: Optional. The index of the first track to return. Default: 0 (the first object).
- **market**: Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Both Public and Private playlists belonging to any user are retrievable on provision of a valid access token. Defaults to `spotifyr::get_spotify_access_token()`.
- **include_meta_info**: Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.
**Value**


**See Also**

Other track functions: `get_album_tracks()`, `get_tracks()`, `get_track()`

---

**get_recommendations**  
Create a playlist-style listening experience based on seed artists, tracks and genres.

**Description**

All parameters are optional, but at least one of `seed_artists`, `seed_tracks` and `seed_genres` must be given.

**Usage**

```r
get_recommendations(  
  limit = 20,  
  market = NULL,  
  seed_artists = NULL,  
  seed_genres = NULL,  
  seed_tracks = NULL,  
  max_acousticness = NULL,  
  max_danceability = NULL,  
  max_duration_ms = NULL,  
  max_energy = NULL,  
  max_instrumentalness = NULL,  
  max_key = NULL,  
  max_liveness = NULL,  
  max_loudness = NULL,  
  max_mode = NULL,  
  max_popularity = NULL,  
  max_speechiness = NULL,  
  max_tempo = NULL,  
  max_time_signature = NULL,  
  max_valence = NULL,  
  min_acousticness = NULL,  
  min_danceability = NULL,  
  min_duration_ms = NULL,  
  min_energy = NULL,  
  min_instrumentalness = NULL,  
  min_key = NULL,  
  min_liveness = NULL,
)```
get_recommendations

min_loudness = NULL,
min_mode = NULL,
min_popularity = NULL,
min_speechiness = NULL,
min_tempo = NULL,
min_time_signature = NULL,
min_valence = NULL,
target_acousticness = NULL,
target_danceability = NULL,
target_duration_ms = NULL,
target_energy = NULL,
target_instrumentalness = NULL,
target_key = NULL,
target_liveness = NULL,
target_loudness = NULL,
target_mode = NULL,
target_popularity = NULL,
target_speechiness = NULL,
target_tempo = NULL,
target_time_signature = NULL,
target_valence = NULL,
authorization = get_spotify_access_token(),
include_seeds_in_response = FALSE
)

Arguments

limit

Optional. The target size of the list of recommended tracks. For seeds with unusually small pools or when highly restrictive filtering is applied, it may be impossible to generate the requested number of recommended tracks. Debugging information for such cases is available in the response. Default: 20. Minimum: 1. Maximum: 100.

market

Optional. An ISO 3166-1 alpha-2 country code or the string from_token. Provide this parameter if you want to apply Track Relinking. Because min_* and target_* are applied to pools before relinking, the generated results may not precisely match the filters applied. Original, non-relinked tracks are available via the linked_from attribute of the relinked track response.

seed_artists

A character vector of Spotify IDs for seed artists. Up to 5 seed values may be provided in any combination of seed_artists, seed_tracks and seed_genres.

seed_genres

A character vector of any genres in the set of available genre seeds. Up to 5 seed values may be provided in any combination of seed_artists, seed_tracks and seed_genres.

seed_tracks

A character vector of Spotify IDs for a seed track. Up to 5 seed values may be provided in any combination of seed_artists, seed_tracks and seed_genres.

max_acousticness

Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ acousticness.
get_recommendations

max_danceability
Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ danceability.

max_duration_ms
Optional. Integer indicating a hard ceiling on recommended tracks’ duration in milliseconds.

max_energy
Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ energy.

max_instrumentalness
Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ instrumentalness.

max_key
Optional. Integer ranging from 1 to 12 indicating a hard ceiling on recommended tracks’ key.

max_liveness
Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ liveness.

max_loudness
Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ loudness.

max_mode
Optional. Integer ranging from 0 to 1 indicating a hard ceiling on recommended tracks’ mode.

max_popularity
Optional. Integer ranging between 0 and 100 indicating a hard ceiling on recommended tracks’ popularity.

max_speechiness
Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ speechiness.

max_tempo
Optional. Numeric value indicating a hard ceiling on recommended tracks’ tempo.

max_time_signature
Optional. Integer indicating a hard ceiling on recommended tracks’ time signature.

max_valence
Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ valence.

min_acousticness
Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ acousticness.

min_danceability
Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ danceability.

min_duration_ms
Optional. Integer indicating a hard floor on recommended tracks’ duration in milliseconds.

min_energy
Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ energy.

min_instrumentalness
Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ instrumentalness.
get_recommendations

min_key
Optional. Integer ranging from 1 to 12 indicating a hard floor on recommended tracks’ key.

min_liveness
Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ liveness.

min_loudness
Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ loudness.

min_mode
Optional. Integer ranging from 0 to 1 indicating a hard floor on recommended tracks’ mode.

min_popularity
Optional. Integer ranging between 0 and 100 indicating a hard floor on recommended tracks’ popularity.

min_speechiness
Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ speechiness.

min_tempo
Optional. Numeric value indicating a hard floor on recommended tracks’ tempo.

min_time_signature
Optional. Integer indicating a hard floor on recommended tracks’ time signature.

min_valence
Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ valence.

target_acousticness
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ acousticness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_danceability
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ danceability. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_duration_ms
Optional. Integer indicating a target value for recommended tracks’ duration in milliseconds. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_energy
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ energy. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_instrumentalness
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ instrumentalness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.
get_recommendations

**target_key**
Optional. Integer ranging from 1 to 12 indicating a target value for recommended tracks’ key. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_liveness**
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ liveness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_loudness**
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ loudness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_mode**
Optional. Integer ranging from 0 to 1 indicating a target value for recommended tracks’ mode. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_popularity**
Optional. Integer ranging between 0 and 100 indicating a target value for recommended tracks’ popularity. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_speechiness**
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ speechiness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_tempo**
Optional. Numeric value indicating a target value for recommended tracks’ tempo. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_time_signature**
Optional. Integer indicating a target value for recommended tracks’ time signature. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_valence**
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ valence. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6`
and target_danceability = 0.8. All target values will be weighed equally in ranking results.

**authorization**  
Required. A valid access token from the Spotify Accounts service. See the [Web API authorization guide](https://developer.spotify.com/documentation/web-api/guides/authorization/) for more details. Defaults to `spotifyr::get_spotify_access_token()`.

**include_seeds_in_response**  
Optional. Boolean for whether to include seed object in response. Defaults to `FALSE`.

### Value

Returns a data frame of results recommendations. See the official [Spotify Web API documentation](https://developer.spotify.com/documentation/web-api/) for more information.

### See Also

Other personalization functions: `get_my_top_artists_or_tracks()`, `get_recommendations_all()`

### Examples

```r
get_recommendations(market = 'SE', seed_genres = 'rock')
```

---

### `get_recommendations_all`

*Get recommendations for unlimited vector of track IDs*

---

### Description

Get recommendations for a submitted vector of track IDs, with no limit on the number of seed tracks.

### Usage

```r
get_recommendations_all(track_ids, valence = NULL)
```

### Arguments

- **track_ids**: A vector containing the IDs of the tracks you’d like recommendations for.
- **valence**: The target valence for the recommendations.

### Details

This is a wrapper for the `get_recommendations` function, which provides a workaround for the limit of 5 seed tracks per recommendation call. The function splits a supplied vector of track IDs into subsets of length 5, then applies a `get_recommendations` call, 5 tracks at a time. This should generate a data frame of recommended tracks, with length equal to the supplied vector of track ids.
get_related_artists

Value
Returns a data frame containing binded recommendations from the Spotify API in a single data frame.

See Also
Other personalization functions: get_my_top_artists_or_tracks(), get_recommendations()

Examples

```r
get_recommendations_all(
  track_ids = c("5VIpLopHgoIkCSSj7JPCMA", "3QRYG0FFUYv4qJneE4DX1gR")
)
```

get_related_artists  Get Spotify catalog information about artists similar to a given artist. Similarity is based on analysis of the Spotify community's listening history.

Description
Get Spotify catalog information about artists similar to a given artist. Similarity is based on analysis of the Spotify community's listening history.

Usage

```r
get_related_artists(
  id,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The Spotify ID for the artist.</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to <code>spotifyr::get_spotify_access_token()</code></td>
</tr>
<tr>
<td>include_meta_info</td>
<td>Optional. Boolean indicating whether to include full result, with meta information such as &quot;total&quot;, and &quot;limit&quot;. Defaults to FALSE.</td>
</tr>
</tbody>
</table>

Value
get_shows

See Also

Other artist functions: `get_artist_albums()`, `get_artist_top_tracks()`, `get_artists()`, `get_artist()`, `get_my_followed_artists()`

---

get_show

*Get Spotify catalog information for a single show.*

**Description**

Get Spotify catalog information for a single show.

**Usage**

```r
get_show(id, market = NULL, authorization = get_spotify_authorization_code())
```

**Arguments**

- `id` The Spotify ID for the show.
- `market` Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". If a country code is specified, only shows and episodes that are available in that market will be returned. If a valid user access token is specified in the request header, the country associated with the user account will take priority over this parameter.
- `authorization` Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Reading the user's resume points on episode objects requires the user-read-playback-position scope. See Using Scopes.

**Value**

Returns a data frame of results containing show data. See the official documentation for more information.

---

get_shows

*Get Spotify catalog information for multiple shows identified by their Spotify IDs.*

**Description**

Get Spotify catalog information for multiple shows identified by their Spotify IDs.
get_show_episodes

Usage

get_shows(
  ids,
  market = NULL,
  authorization = get_spotify_authorization_code(),
  include_meta_info = FALSE
)

Arguments

ids Required. A character vector of the Spotify IDs for the shows. Maximum: 20 IDs.

market Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". If a country code is specified, only shows and episodes that are available in that market will be returned. If a valid user access token is specified in the request header, the country associated with the user account will take priority over this parameter.

authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to get_spotify_authorization_code(). The access token must have been issued on behalf of the current user.

include_meta_info Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value

Returns a data frame of results containing show data. See https://developer.spotify.com/documentation/web-api/reference/shows/get-several-shows/ for more information.

get_show_episodes Get Spotify catalog information about an show’s episodes. Optional parameters can be used to limit the number of episodes returned.

Description

Get Spotify catalog information about an show’s episodes. Optional parameters can be used to limit the number of episodes returned.

Usage

get_show_episodes(
  id,
  limit = 20,
  offset = 0,
)
Arguments

- **id**: Required. The Spotify ID for the show.

- **limit**: Optional.
  Maximum number of results to return.
  Default: 20
  Minimum: 1
  Maximum: 50

- **offset**: Optional.
  The index of the first album to return.
  Default: 0 (the first album).
  Maximum offset (including limit): 10,000.
  Use with limit to get the next set of episodes.

- **market**: Optional.
  An ISO 3166-1 alpha-2 country code or the string "from_token". If a country code is specified, only shows and episodes that are available in that market will be returned. If a valid user access token is specified in the request header, the country associated with the user account will take priority over this parameter.

- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`.
  The access token must have been issued on behalf of the current user.
  Reading the user's resume points on episode objects requires the user-read-playback-position scope. See Using Scopes.

- **include_meta_info**: Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value

Returns a data frame of results containing show data. See the official API documentation for more information.

---

**get_spotify_access_token**

*Get Spotify Access Token*

Description

This function creates a Spotify access token.
get_spotify_authorization_code

Usage

get_spotify_access_token(
  client_id = Sys.getenv("SPOTIFY_CLIENT_ID"),
  client_secret = Sys.getenv("SPOTIFY_CLIENT_SECRET")
)

Arguments

client_id Defaults to System Environment variable "SPOTIFY_CLIENT_ID"
client_secret Defaults to System Environment variable "SPOTIFY_CLIENT_SECRET"

Value

The Spotify Access Token of the current user or a meaningful error message.

See Also

Other authentication functions: get_spotify_authorization_code()

Examples

token <- get_spotify_access_token()

get_spotify_authorization_code

Get Spotify Authorization Code

Description

This function creates a Spotify authorization code. See httr::oauth2.0_token.

Usage

get_spotify_authorization_code(
  client_id = Sys.getenv("SPOTIFY_CLIENT_ID"),
  client_secret = Sys.getenv("SPOTIFY_CLIENT_SECRET"),
  scope = scopes()
)

Arguments

client_id Defaults to System Envioronment variable "SPOTIFY_CLIENT_ID"
client_secret Defaults to System Envioronment variable "SPOTIFY_CLIENT_SECRET"
scope Space delimited string of spotify scopes, found here: https://developer.spotify.com/documentation/general/guides/scopes/. All scopes are selected by default
Value

The Spotify Web API Token 2.0 reference class object (see `httr::oauth2.0_token`), or an error message.

See Also

Other authentication functions: `get_spotify_access_token()`

Examples

```r
authorization <- get_spotify_authorization_code()
```

---

### get_track

Get Spotify catalog information for a single track identified by its unique Spotify ID.

#### Usage

```r
get_track(id, market = NULL, authorization = get_spotify_access_token())
```

#### Arguments

- **id**
  - The Spotify ID for the track.
- **market**
  - Optional.
  - An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking.
- **authorization**
  - Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.

#### Value


#### See Also

Other track functions: `get_album_tracks()`, `get_playlist_tracks()`, `get_tracks()`
**get_tracks**

Get Spotify catalog information for a single track identified by its unique Spotify ID.

**Description**

Get Spotify catalog information for a single track identified by its unique Spotify ID.

**Usage**

```r
get_tracks(
  ids,
  market = NULL,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)
```

**Arguments**

- **ids**
  - The Spotify ID for the track.
- **market**
  - Optional.
  - An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking.
- **authorization**
  - Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.
- **include_meta_info**
  - Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

**Value**


**See Also**

Other track functions: `get_album_tracks()`, `get_playlist_tracks()`, `get_track()`
get_track_audio_analysis

*Get a detailed audio analysis for a single track identified by its unique Spotify ID.*

**Description**

Get a detailed audio analysis for a single track identified by its unique Spotify ID.

**Usage**

```
get_track_audio_analysis(id, authorization = get_spotify_access_token())
```

**Arguments**

- **id**
  - The Spotify ID for the track.
- **authorization**

**Value**


**See Also**

Other musicology functions: `get_artist_audio_features()`, `get_genre_artists()`, `get_playlist_audio_features()`, `get_track_audio_features()`, `get_user_audio_features()`

get_track_audio_features

*Get audio features of tracks*

**Description**

Get audio feature information for up to 100 tracks identified by their unique Spotify IDs.

**Usage**

```
get_track_audio_features(ids, authorization = get_spotify_access_token())
```
get_user_audio_features

get_user_audio_features

Description

Get audio feature information for a users’ playlists.

Usage

get_user_audio_features(
    username = NULL,
    authorization = get_spotify_access_token()
)

Arguments

username Required. String of Spotify username.
authorization Required. A valid access token from the Spotify Accounts service. See the Web
API authorization guide for more details. Defaults to package::get_spotify_access_token()

Value

Returns a data frame of results containing track audio features data. See https://developer.spotify.com/documentation/web-api/reference/tracks/get-several-audio-features/ for more information.

See Also

Other musicology functions: get_artist_audio_features(), get_genre_artists(), get_playlist_audio_features(), get_track_audio_analysis(), get_user_audio_features()
get_user_playlists  

Get List of User Playlists

Description
Get a list of the playlists owned or followed by a Spotify user.

Usage

get_user_playlists(
   user_id,
   limit = 20,
   offset = 0,
   authorization = get_spotify_authorization_code(),
   include_meta_info = FALSE
)

Arguments

user_id  Required. The user’s Spotify user ID.
limit    Optional.  Maximum number of playlists to return. Default: 20  
          Minimum: 1  
          Maximum: 50

offset   Optional.  The index of the first playlist to return. Default: 0 (the  
          first object). Maximum offset: 100,000. Use with limit to get the  
          next set of playlists.

authorization  Required. A valid access token from the Spotify Accounts service.  
               See the Web API authorization Guide for more details. Defaults to  
               spotifyr::get_spotify_authorization_code(). The access token must  
               have been issued on behalf of the current user. Private playlists  
               are only retrievable for the current user and requires the playlist-read-private  
               scope to have been authorized by the user. Note that this scope alone will not  
               return collaborative playlists, even though they are always private.  
               Collaborative playlists are only retrievable for the current user and requires the  
               playlist-read-collaborative scope to have been authorized by the user.

include_meta_info  Optional. Boolean indicating whether to include full result, with meta information  
                    such as "total", and "limit". Defaults to FALSE.

Value

Returns a data frame of results containing user playlist information. See the official Spotify Web  
API documentation for more information.
**get_user_profile**

Get User Public Profile Information

**Description**

Get public profile information about a Spotify user.

**Usage**

```r
get_user_profile(user_id, authorization = get_spotify_access_token())
```

**Arguments**

- **user_id** Required. The user’s Spotify user ID.
- **authorization** Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.

**Value**


---

**pause_my_playback**

Pause Playback

**Description**

Pause playback on the user’s device.

**Usage**

```r
pause_my_playback(
  device_id = NULL,
  authorization = get_spotify_authorization_code()
)
```
remove_tracks_from_playlist

Arguments

device_id  Optional. The id of the device this command is targeting. If not supplied, the
user's currently active device is the target.

authorization  Required. A valid access token from the Spotify Accounts service. See the Web
API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.  
The access token must have been issued on behalf of the current user.
The access token must have the user-modify-playback-state scope autho-

Value

The response from the Spotify Web API on the PUT request, with a meaningful error message if the
operation was not successful.

See Also

Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_devices()`,
`get_my_recently_played()`, `seek_to_position()`, `set_my_repeat_mode()`, `skip_my_playback_previous()`,
`skip_my_playback()`, `start_my_playback()`, `toggle_my_shuffle()`, `transfer_my_playback()`

remove_tracks_from_playlist

Remove Tracks from User's Playlist

Description

Remove one or more tracks from a user's playlist.

Usage

```r
remove_tracks_from_playlist(
  playlist_id,
  uris,
  authorization = get_spotify_authorization_code()
)
```

Arguments

playlist_id  Required. The Spotify ID for the playlist.

uris  Optional. A character vector of Spotify track URIs to add. For example
uris = "spotify:track:4iV5W9uYEdYUVa79Axb7Rh","spotify:track:1301WleyT98MSxVHPZCA6M"
A maximum of 100 tracks can be removed in one request.

authorization  Required. A valid access token from the Spotify Accounts service. See the Web
API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`.  
The access token must have been issued on behalf of the current user.
Removing tracks to the current user's public playlists requires authorization of
the playlist-modify-public scope; removing tracks from the current user's private playlist (including collaborative playlists) requires the playlist-modify-private scope. See Using Scopes.

Value

The response from the Spotify Web API on the DELETE request, with a meaningful error message if the operation was not successful.

See Also

Other playlist functions: add_tracks_to_playlist(), change_playlist_details(), check_users_following(), create_playlist(), get_my_playlists(), get_playlist(), get_user_playlists(), tidy()

---

### scopes

<table>
<thead>
<tr>
<th>Valid Authorization Scopes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

**Description**

A vector of valid scopes for get_spotify_authorization_code.

**Usage**

scopes()

**Value**

A character vector of valid authorization scopes for the Spotify Web API. See Spotify Web API Authorization Scopes

**Examples**

scopes()

---

**search_spotify**  **Search for an Item**

**Description**

Get Spotify Catalog information about artists, albums, tracks or playlists that match a keyword string. For more information see the official documentation.
Usage

```r
search_spotify(
  q,
  type = c("album", "artist", "playlist", "track"),
  market = NULL,
  limit = 20,
  offset = 0,
  include_external = NULL,
  authorization = get_spotify_access_token(),
  include_meta_info = FALSE
)
```

Arguments

- **q**: Required.
  Search query keywords and optional field filters and operators.

- **type**: A character vector of item types to search across.
  Valid types are `album`, `artist`, `playlist`, and `track`.
  Search results include hits from all the specified item types.
  For example: `q = "name:abacab"` and `type = c("album","track")` returns both albums and tracks with "abacab" included in their name.

- **market**: Optional.
  An ISO 3166-1 alpha-2 country code or the string "from_token".
  If a country code is specified, only artists, albums, and tracks with content that is playable in that market is returned.

  - Playlist results are not affected by the market parameter.
  - If market is set to "from_token", and a valid access token is specified in the request header, only content playable in the country associated with the user account, is returned.
  - Users can view the country that is associated with their account in the account settings. A user must grant access to the user-read-private scope prior to when the access token is issued.

- **limit**: Optional.
  Maximum number of results to return.
  Default: 20
  Minimum: 1
  Maximum: 50

  - Note: The limit is applied within each type, not on the total response.
  - For example, if the limit value is 3 and the type is `c("artist","album")`, the response contains 3 artists and 3 albums.

- **offset**: Optional.
  The index of the first result to return.
  Default: 0 (the first result).
  Maximum offset (including limit): 10,000.
  Use with limit to get the next page of search results.
seek_to_position

include_external
Optional. Possible values: audio
If include_external = "audio" is specified the response will include any relevant audio content that is hosted externally.
By default external content is filtered out from responses.

authorization
Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()

include_meta_info
Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value
A tibble with detailed information about the searched album, artist, playlist, track or their combination.

Examples

search_spotify('radiohead', 'artist')

seek_to_position

Seeks to the given position in the user’s currently playing track.

Description
Seeks to the given position in the user’s currently playing track.

Usage

seek_to_position(
  position_ms,
  device_id = NULL,
  authorization = get_spotify_authorization_code()
)

Arguments

position_ms Required. Integer indicating the position in milliseconds to seek to. Must be a positive number. Passing in a position that is greater than the length of the track will cause the player to start playing the next song.

device_id Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.


authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user.

The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

Value

The response from the Spotify Web API on the POST request, with a meaningful error message if the operation was not successful.

See Also

Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_devices()`, `get_my_recently_played()`, `pause_my_playback()`, `set_my_repeat_mode()`, `skip_my_playback_previous()`, `skip_my_playback()`, `start_my_playback()`, `toggle_my_shuffle()`, `transfer_my_playback()`

---

**set_my_repeat_mode**  
*Set User Playback Mode*

**Description**

Set the repeat mode for the user's playback. Options are repeat-track, repeat-context, and off.

**Usage**

```r
def set_my_repeat_mode(
    state, 
    device_id = NULL, 
    authorization = get_spotify_authorization_code() 
)
```

**Arguments**

- **state** Required.  
  "track", "context", or "off" "track" will repeat the current track.  
  "context" will repeat the current context.  
  "off" will turn repeat off  

- **device_id** Optional. The id of the device this command is targeting. If not supplied, the user's currently active device is the target.

- **authorization** Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user.

  The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

---
set_my_volume

Value

The response from the Spotify Web API on the PUT request, with a meaningful error message if the operation was not successful.

See Also

Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_devices()`, `get_my_recently_played()`, `pause_my_playback()`, `seek_to_position()`, `skip_my_playback_previous()`, `skip_my_playback()`, `start_my_playback()`, `toggle_my_shuffle()`, `transfer_my_playback()`

---

**set_my_volume**  
*Set User Device Volume*

Description

Set the volume for the user’s current playback device.

Usage

```r
set_my_volume(
  volume_percent = 50,
  device_id = NULL,
  authorization = get_spotify_authorization_code()
)
```

Arguments

- **volume_percent**  
  Required integer value. The volume to set. Must be a value from 0 to 100 inclusive. Defaults to 50.

- **device_id**  
  Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.

- **authorization**  
  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

Value

The response from the Spotify Web API on the PUT request, with a meaningful error message if the operation was not successful.
skip_my_playback  
**Skips to Next Track**

**Description**

Skips to next track in the user’s queue.

**Usage**

```r
skip_my_playback(
  device_id = NULL,
  authorization = get_spotify_authorization_code()
)
```

**Arguments**

- `device_id`: Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.
- `authorization`: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

**Value**

The response from the Spotify Web API on the POST request, with a meaningful error message if the operation was not successful.

**See Also**

Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_devices()`, `get_my_recently_played()`, `pause_my_playback()`, `seek_to_position()`, `set_my_repeat_mode()`, `skip_my_playback_previous()`, `start_my_playback()`, `toggle_my_shuffle()`, `transfer_my_playback()`

---

skip_my_playback_previous

**Skips to previous track in the user’s queue.**

**Description**

Skips to previous track in the user’s queue.
Usage

```r
skip_my_playback_previous(
    device_id = NULL,
    authorization = get_spotify_authorization_code()
)
```

Arguments

- `device_id` Optional. The id of the device this command is targeting. If not supplied, the user's currently active device is the target.
- `authorization` Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

Value

The response from the Spotify Web API on the POST request, with a meaningful error message if the operation was not successful.

See Also

Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_devices()`, `get_my_recently_played()`, `pause_my_playback()`, `seek_to_position()`, `set_my_repeat_mode()`, `skip_my_playback()`, `start_my_playback()`, `toggle_my_shuffle()`, `transfer_my_playback()`

---

**spotifyr**

**spotifyr package**

---

**Description**

A Quick and Easy Wrapper for Pulling Track Audio Features from Spotify’s Web API in Bulk

**Details**

See the README on GitHub

**artist functions**

Retrieve information related to artists. **get_artist**: Get the data of a single artist. **get_artists**: Get the data of multiple artist. **get_related_artists**: Get that of related artist to an original artist. **get_artist_albums**: Get artists who appear on an album.
track functions
Retrieve information related to individual song recordings (songs, concerts).
- get_track
- get.tracks
- get_playlist_tracks

album functions
Retrieve information about albums. See also get_album_tracks.

playlist functions
Work with playlists.
- add_tracks_to_playlist
- change_playlist_details
- get_user_playlists

player functions
Interact with the user’s devices and players.

personalization functions
- get_my_top_artists_or_tracks

musicology functions
Functions related to the contents of the music.
- get_track_audio_analysis
- get_artist_audio_features
- get_playlist_audio_features
- get_user_audio_features
- get_genre_artists

lyrics functions
Functions related to the lyrics of the music, such as
- get_discography

search functions
Search for an artist, song or other keyword.
The main function is search_spotify.
**start my playback**

**authentication functions**

Helper functions to provide authentication function for requests. Never reveal in public documents, markdown files the returned values of these functions:

get_spotify_authorization_code
get_spotify_access_token.

---

**start my playback**  
*Skips to previous track in the user’s queue.*

**Description**

Skips to previous track in the user’s queue.

**Usage**

```r
start_my_playback(
  device_id = NULL,
  context_uri = NULL,
  uris = NULL,
  offset = NULL,
  position_ms = NULL,
  authorization = get_spotify_authorization_code()
)
```

**Arguments**

- **device_id**  
  Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.

- **context_uri**  
  Optional. String of the Spotify URI of the context to play. Valid contexts are albums, artists, playlists. Example `context_uri = "spotify:album:1Je1IMj1Bcx1Fz0WE7oPT"`.

- **uris**  
  Optional. A character vector of the Spotify track URIs to play. For example:  
  ```r
  "uris": c("spotify:track:4iV5W9uYEdYVu79AXb7Rh","spotify:track:1301WleyT98MSxVHPZCA6M")
  ```

- **offset**  
  Optional. A named list indicating from where the context playback should start.  
  Only available when `context_uri` corresponds to an album or playlist object, or when the `uris` parameter is used.  
  ```r
  "position" is zero based and can't be negative. Example: "offset" = list("position" = 5).
  "uri" is a string representing the uri of the item to start at. Example: "offset" = list("uri" = "spotify:track:1301WleyT98MSxVHPZCA6M").
  ```

- **position_ms**  
  Optional. Integer indicating from what position to start playback. Must be a positive number. Passing in a position that is greater than the length of the track will cause the player to start playing the next song.

- **authorization**  
  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.  
  The access token must have been issued on behalf of the current user.  
  The access token must have the `user-modify-playback-state` scope authorized in order to control playback.
Value

The response from the Spotify Web API on the POST request, with a meaningful error message if the operation was not successful.

See Also

Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_devices()`, `get_my_recently_played()`, `pause_my_playback()`, `seek_to_position()`, `set_my_repeat_mode()`, `skip_my_playback_previous()`, `skip_my_playback()`, `toggle_my_shuffle()`, `transfer_my_playback()`

---

**tidy  Tidy a Playlist**

Description

tidy extracts and tidies the data frame containing track level information that is returned from `get_playlist` as a tibble.

Usage

tidy(x, ...)

### S3 method for class 'playlist'
tidy(x, ...)

### S3 method for class 'playlist'
print(x, ...)

Arguments

x  A playlist object generated from `get_playlist()`.

...  Unused.

Value

Return or print a tidy version a tibble containing the track level information that is returned from `get_playlist`.

See Also

Other playlist functions: `add_tracks_to_playlist()`, `change_playlist_details()`, `check_users_following()`, `create_playlist()`, `get_my_playlists()`, `get_playlist()`, `get_user_playlists()`, `remove_tracks_from_playlist()`
toggle_my_shuffle

Examples

```r
fall <- get_playlist("4GSV6uJzlbtTCPJhnVU1o8")
tidy(fall)
print(fall)
```

toggle_my_shuffle  Toggle shuffle on or off for user’s playback.

Description

Toggle shuffle on or off for user’s playback.

Usage

```r
toggle_my_shuffle(
  state,
  device_id = NULL,
  authorization = get_spotify_authorization_code()
)
```

Arguments

- **state** Required.
  - `TRUE`: Shuffle user’s playback
  - `FALSE`: Do not shuffle user’s playback
- **device_id** Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.
- **authorization** Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifrr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

Value

The response from the Spotify Web API on the PUT request, with a meaningful error message if the operation was not successful.

See Also

Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_devices()`, `get_my_recently_played()`, `pause_my_playback()`, `seek_to_position()`, `set_my_repeat_mode()`, `skip_my_playback_previous()`, `skip_my_playback()`, `start_my_playback()`, `transfer_my_playback()`
**transfer_my_playback**  
*Transfer playback to a new device and determine if it should start playing.*

**Description**
Transfer playback to a new device and determine if it should start playing.

**Usage**
```r
transfer_my_playback(
  device_ids,  
  play = FALSE,  
  authorization = get_spotify_authorization_code()
)
```

**Arguments**
- **device_ids**  
  Required. A character vector containing the ID of the device this on which playback should be started/transfered. Note: only a single device_id is currently supported.

- **play**  
  Optional.  
  TRUE: Ensure playback happens on new device  
  FALSE (default): keep the current playback state  
  Note that a value of FALSE for the play parameter when also transferring to another device_id will not pause playback. To ensure that playback is paused on the new device you should send a pause command to the currently active device before transferring to the new device_id.

- **authorization**  
  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the user-modify-playback-state scope authorized in order to control playback.

**Value**
The response from the Spotify Web API on the PUT request, with a meaningful error message if the operation was not successful.

**See Also**
Other player functions: `get_my_current_playback()`, `get_my_currently_playing()`, `get_my_devices()`, `get_my_recently_played()`, `pause_my_playback()`, `seek_to_position()`, `set_my_repeat_mode()`, `skip_my_playback_previous()`, `skip_my_playback()`, `start_my_playback()`, `toggle_my_shuffle()`
unfollow_playlist  
Remove Current User from Followers of Playlist.

**Description**

Remove Current User from Followers of Playlist.

**Usage**

```r
unfollow_playlist(
  playlist_id,
  authorization = get_spotify_authorization_code()
)
```

**Arguments**

- **playlist_id**  
  Required. The Spotify ID of the playlist that is to be no longer followed.

- **authorization**  
  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Unfollowing a publicly followed playlist for a user requires authorization of the `playlist-modify-public` scope; unfollowing a privately followed playlist requires the `playlist-modify-private` scope. See Using Scopes. Note that the scopes you provide relate only to whether the current user is following the playlist publicly or privately (i.e. showing others what they are following), not whether the playlist itself is public or private.

**Value**

The response from the Spotify Web API on the DELETE request, with a meaningful error message if the operation was not successful.

---

validate_parameters  
Assertion for Correct API Requests.

**Description**

Assertions are made to give early and precise error messages for wrong API call parameters.
validate_parameters

Usage

validate_parameters(
    artists_or_tracks = NULL,
    artist_or_user = NULL,
    market = NULL,
    country = NULL,
    limit = NULL,
    offset = NULL,
    locale = NULL,
    volume_percent = NULL,
    time_range = NULL,
    position_ms = NULL,
    state = NULL,
    include_meta_info = NULL
)

Arguments

artists_or_tracks
    The type parameter must be either 'artists' or 'tracks'.

artist_or_user
    "The type parameter must be either 'artist' or 'user'."

market
    Optional.
    An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking

country
    Optional.
    An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking

limit

offset
    Optional. The index of the first item to return. Defaults to 0, i.e., the first object. Use with limit to get the next set of items.

locale
    Optional. The desired language, consisting of an ISO 639-1 language code and an ISO 3166-1 alpha-2 country code, joined by an underscore. For example: es_MX, meaning "Spanish (Mexico)". Provide this parameter if you want the category strings returned in a particular language. Note that, if locale is not supplied, or if the specified language is not available, the category strings returned will be in the Spotify default language (American English). The locale parameter, combined with the country parameter, may give odd results if not carefully matched. For example country=SE&locale=de_DE will return a list of categories relevant to Sweden but as German language strings.

volume_percent
    Required integer value. The volume to set. Must be a value from 0 to 100 inclusive. Defaults to 50.

time_range
    Optional. Over what time frame the affinities are computed. Valid values: long_term (calculated from several years of data and including all new data as it becomes available), medium_term (approximately last 6 months), short_term (approximately last 4 weeks). Default: medium_term.
validate_parameters

- **position_ms**: Optional. Integer indicating from what position to start playback. Must be a positive number. Passing in a position that is greater than the length of the track will cause the player to start playing the next song.

- **state**: The state parameter must be exactly one of 'track', 'context' or 'off'.

- **include_meta_info**: Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

**Details**

These assertions are called from various wrapper functions. However, you can also call this function directly to make sure that you are adding (programatically) the correct parameters to a call.

All validate_parameters parameters default to NULL. Asserts the correct parameter values for any values that are not NULL.

**Value**

A boolean if the parameter matches the Spotify Web API parameter range.
Index

* album functions
  get_album, 10
  get_albums, 11
  get_discography, 20

* artist functions
  get_artist, 13
  get_artist_albums, 14
  get_artist_top_tracks, 17
  get_artists, 14
  get_followed_artists, 27
  get_related_artists, 45

* audio
  get_playlist_audio_features, 36

* authentication functions
  get_spotify_access_token, 48
  get_spotify_authorization_code, 49

* authorization functions
  scopes, 57

* features
  get_playlist_audio_features, 36

* label functions
  get_label_artists, 24

* musicology functions
  get_artist_audio_features, 16
  get_genre_artists, 22
  get_playlist_audio_features, 36
  get_track_audio_analysis, 52
  get_track_audio_features, 52
  get_user_audio_features, 53

* personalization functions
  get_my_top_artists_or_tracks, 33
  get_recommendations, 39
  get_recommendations_all, 44

* player functions
  get_my_current_playback, 26
  get_my_currently_playing, 25
  get_my_devices, 27
  get_my_recently_played, 30
  pause_my_playback, 55

  seek_to_position, 59
  set_my_repeat_mode, 60
  skip_my_playback, 62
  skip_my_playback_previous, 62
  start_my_playback, 65
  toggle_my_shuffle, 67
  transfer_my_playback, 68

* playlist functions
  add_tracks_to_playlist, 3
  change_playlist_details, 4
  check_users_following, 6
  create_playlist, 7
  get_my_playlists, 28
  get_playlist, 35
  get_user_playlists, 54
  remove_tracks_from_playlist, 56
  tidy, 66

* playlists
  get_playlist_audio_features, 36

* search functions
  search_spotify, 57

* track functions
  get_album_tracks, 12
  get_playlist_tracks, 37
  get_track, 50
  get_tracks, 51

* track
  get_playlist_audio_features, 36

  add_tracks_to_playlist, 3, 5, 7, 8, 29, 36, 55, 57, 64, 66

  change_playlist_details, 4, 4, 7, 8, 29, 36, 55, 57, 64, 66

  check_me_following, 5
  check_users_following, 4, 5, 6, 8, 29, 36, 55, 57, 66

  create_playlist, 4, 5, 7, 29, 36, 55, 57, 66

  dedupe_album_names, 8

72
follow_artist, 9
follow_artists_or_users, 9
follow_playlist, 9

get_artist, 10, 11, 20
get_album, 10, 11, 20
get_album_tracks, 12, 39, 50, 51, 64
get_albums, 11, 11, 20
get_artist, 13, 14, 16, 17, 28, 46, 63
get_artist_albums, 13, 14, 16, 17, 28, 46, 63
get_artist_audio_features, 16, 23, 36, 52, 53, 64
get_artist_top_tracks, 13, 14, 16, 17, 28, 46
get_artists, 13, 14, 16, 17, 28, 46, 63
get_categories, 18
get_category, 18
get_category_playlists, 19
get_discography, 11, 20, 64
get_featured_playlists, 21
get_genre_artists, 17, 22, 36, 52, 53, 64
get_label_artists, 24
get_my_current_playback, 25, 26, 27, 31, 56, 60–63, 66–68
get_my_currently_playing, 25, 26, 27, 31, 56, 60–63, 66–68
get_my_devices, 25, 26, 27, 31, 56, 60–63, 66–68
get_my_followed_artists, 13, 14, 16, 17, 27, 46
get_my_playlists, 4, 5, 7, 8, 28, 36, 55, 57, 66
get_my_profile, 29
get_my_recently_played, 25–27, 30, 56, 60–63, 66–68
get_my_saved_albums, 31
get_my_saved_tracks, 32
get_my_top_artists_or_tracks, 33, 44, 45, 64
get_new_releases, 34
get_playlist, 4, 5, 7, 8, 29, 35, 55, 57, 66
get_playlist_audio_features, 17, 23, 36, 52, 53, 64
get_playlist_cover_image, 37
get_playlist_tracks, 13, 37, 50, 51, 64
get_recommendations, 34, 39, 44, 45
get_recommendations_all, 34, 44, 44
get_related_artists, 13, 14, 16, 17, 28, 45, 63
get_show, 46
get_show_episodes, 47
get_shows, 46
get_spotify_access_token, 48, 50, 65
get_spotify_authorization_code, 49, 49, 57, 65
get_track, 13, 39, 50, 51, 64
get_track_audio_analysis, 17, 23, 36, 52, 53, 64
get_track_audio_features, 17, 23, 36, 52, 53
get_tracks, 13, 39, 50, 51, 64
get_user_audio_features, 17, 23, 36, 52, 53, 64
get_user_playlists, 4, 5, 7, 8, 29, 36, 54, 57, 64, 66
get_user_profile, 55
nest, 20
oauth2.0_token, 49, 50
pause_my_playback, 25–27, 31, 55, 60–63, 66–68
print_playlist(tidy), 66
remove_tracks_from_playlist, 4, 5, 7, 8, 29, 36, 55, 56, 66
scopes, 57
search_spotify, 57, 64
set_my_repeat_mode, 25–27, 31, 56, 60, 62, 63, 66–68
set_my_volume, 61
skip_my_playback, 25–27, 31, 56, 60, 61, 62, 63, 66–68
spotifyr, 63
start_my_playback, 25–27, 31, 56, 60–63, 65, 67, 68
tidy, 4, 5, 7, 8, 29, 36, 55, 57, 66, 66
toggle_my_shuffle, 25–27, 31, 56, 60–63, 66, 67, 68
transfer_my_playback, 25–27, 31, 56, 60–63, 66, 67, 68
unfollow_playlist, 69
validate_parameters, 69, 71