Pyfy Documentation

Release 2.2.0

Omar Ryhan

Oct 27, 2021

CONTENTS

1 Setup **3**

2 Quick Start **5**

3 Getting Started **7**

4 Authentication **9**

5 API endpoints **11**
- 5.1 Albums . . . 11
- 5.2 Artists . . . 11
- 5.3 Browse . . . 12
- 5.4 Episodes . . . 13
- 5.5 Follow . . . 13
- 5.6 User Library . . . 14
- 5.7 Personalization . . . 15
- 5.8 Player . . . 15
- 5.9 Playlists . . . 17
- 5.10 Search . . . 18
- 5.11 Shows . . . 18
- 5.12 Tracks . . . 19
- 5.13 Users Profile . . . 19

6 Pagination **21**

7 Sync Client API **23**

8 Async Client API **47**

9 Exceptions API **71**

10 Credentials API **73**

11 Testing **77**
- 11.1 Unit tests . . . 77
- 11.2 Integration tests . . . 77

12 Indices and tables **79**

Python Module Index **81**

Index **83**

Pyfy is a Sync + Async Pythonic Spotify Client that focuses on ease of use in personal projects and API stability and security for production grade codebases.

CHAPTER

ONE

SETUP

```
$ pip install pyfy
```

CHAPTER

TWO

QUICK START

Sync

```
from pyfy import Spotify

spt = Spotify('your_access_token')

spt.play()
spt.volume(85)
spt.next()
spt.pause()
```

Async

```
import asyncio
from pyfy import AsyncSpotify

spt = AsyncSpotify('your_access_token')

async def search():
    return await spt.search('A tout le monde')

search_result = asyncio.run(search())
```

CHAPTER

THREE

GETTING STARTED

You should start by creating client credentials from Spotify's Developer's console.

Next, edit your application's settings and set a Redirect URL. If it's for personal use then set it as:

> http://localhost:9000 *Port can be any port of choice, not necessarily 9000*

Next, copy your:

1. Client ID
2. Client Secret
3. Redirect URL (That you just set)

Next, figure out the scopes that you think you'll need from here: https://developer.spotify.com/documentation/general/guides/scopes/

e.g. `["user-library-modify", "app-remote-control"]`

Next, follow the first authentication scheme from below (it's the one you'll most likely need, unless you're sure otherwise)

CHAPTER

FOUR

AUTHENTICATION

1. Authorization Code Flow (OAuth2) (recommended)

Suitable if you want to access user-related resources. e.g. user-playlists, user-tracks etc.

Click here for full working examples with Sanic(async) and Flask(sync).

```
from pyfy import Spotify, ClientCreds, UserCreds, AuthError, ApiError

client = ClientCreds(
    client_id='clientid',
    client_secret='client_secret',
    redirect_uri='https://localhost:9000",
    scopes=["user-library-modify", "app-remote-control"]
)
spt = Spotify(client_creds=client)

def authorize():
    # Fist step of OAuth, Redirect user to spotify's authorization endpoint
    if spt.is_oauth_ready:
        return redirect(spt.auth_uri())

# Authorization callback
def callback(grant):
    try:
        user_creds = spt.build_credentials(grant=grant)
    except AuthError as e:
        abort(401)
        logging.info(e.msg)
        logging.info(e.http_response)
    else:
        db.insert(user_creds)
        return redirect(url_for_home)

def get_user_tracks():
    try:
        return json.dumps(spt.user_tracks())
    except ApiError:
        abort(500)
```

2. User's Access Token: get from here

Same as the Authorization Code Flow above but without a refresh token. Suitable for quick runs.

```
from pyfy import Spotify

spt = Spotify('your access token')
```

3. Client Credentials Flow (OAuth2): get from here

Suitable for when you want to access public information quickly. (Accessing user information is porhibited using this method)

```
from pyfy import ClientCreds, Spotify

client = ClientCreds(client_id=client_id, client_secret=client_secret)
spt = Spotify(client_creds=client)
spt.authorize_client_creds()
```

CHAPTER

FIVE

API ENDPOINTS

5.1 Albums

- **Get an album:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.albums
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/albums/get-album/
- **Get an album's tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.album_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/albums/get-albums-tracks/
- **Get several albums:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.albums
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/albums/get-several-albums/

5.2 Artists

- **Get an artist:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.artists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/artists/get-artist/
- **Artist albums:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.artist_albums
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/artists/get-artists-albums/
- **Artist top tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.artist_top_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/artists/get-artists-top-tracks/
- **Artist related artists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.artist_related_artists

  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/artists/get-related-artists/

- **Get several artists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.artists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/artists/get-several-artists/

5.3 Browse

- **Get a category:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.category
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/browse/get-category/
- **Get a category's playlists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.category_playlist
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/browse/get-categorys-playlists/
- **Get list of categories:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.categories
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/browse/get-list-categories/
- **Get a list of featured playlists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.featured_playlists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/browse/get-list-featured-playlists/
- **Get a list of new releases:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.new_releases
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/browse/get-list-new-releases/
- **Get recommendations based on seeds:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.recommendations
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/browse/get-recommendations/

5.4 Episodes

- **Get an episode:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/episodes/get-an-episode/
- **Get several episodes:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/episodes/get-several-episodes/

5.5 Follow

- **Check if Current User Follows Artists or Users:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.follows_users
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/follow/check-current-user-follows/
- **Check if Users Follow a Playlist:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.follows_playlist
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/follow/check-user-following-playlist/
- **Follow Artists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.follow_artists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/follow/follow-artists-users/
- **Follow Users:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.follow_artists
  - Web API reference: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.follow_users
- **Follow a playlist:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.follow_playlist
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/follow/follow-playlist/
- **Get User's Followed Artists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.followed_artists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/follow/get-followed/
- **Unfollow Artists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.unfollow_artists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/follow/unfollow-artists-users/

- **Unfollow Users:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.unfollow_users
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/follow/unfollow-artists-users/
- **Unfollow Playlist:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.unfollow_playlist
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/follow/unfollow-playlist/

5.6 User Library

- **Check User's Saved Albums:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.owns_albums
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/check-users-saved-albums/
- **Check User's Saved Shows:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/check-users-saved-shows/
- **Check User's Saved Tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.owns_tracks
  - Web API reference: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.owns_tracks
- **Get Current User's Saved Albums:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.user_albums
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/get-users-saved-albums/
- **Get User's Saved Shows:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/get-users-saved-shows/
- **Get a User's Saved Tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.user_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/get-users-saved-tracks/
- **Remove Albums for Current User:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.delete_albums
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/remove-albums-user/
- **Remove User's Saved Shows:**

  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/remove-shows-user/

- **Remove User's Saved Tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.delete_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/remove-tracks-user/

- **Save Albums for Current User:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.save_albums
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/save-albums-user/

- **Save Shows for Current User:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/save-shows-user/

- **Save Tracks for User:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.save_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/library/save-tracks-user/

5.7 Personalization

- **Get a User's Top Artists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.user_top_artists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/personalization/get-users-top-artists-and-tracks/

- **Get a User's Top Tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.artist_top_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/personalization/get-users-top-artists-and-tracks/

5.8 Player

- **Add an Item to the User's Playback Queue:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.queue
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/add-to-queue/

- **Get a User's Available Devices:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.devices

  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/get-a-users-available-devices/

- **Get Information About The User's Current Playback:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.currently_playing_info
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/get-information-about-the-users-current-playback/
- **Get Current User's Recently Played Tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.recently_played_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/get-recently-played/
- **Get the User's Currently Playing Track:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.currently_playing
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/get-the-users-currently-playing-track/
- **Pause a User's Playback:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.pause
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/pause-a-users-playback/
- **Seek To Position In Currently Playing Track:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.seek
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/seek-to-position-in-currently-playing-track/
- **Set Repeat Mode On User's Playback:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.repeat
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/set-repeat-mode-on-users-playback/
- **Set Volume For User's Playback:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.volume
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/set-volume-for-users-playback/
- **Skip User's Playback To Next Track:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.next
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/skip-users-playback-to-next-track/
- **Skip User's Playback To Previous Track:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.previous
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/skip-users-playback-to-previous-track/
- **Start/Resume a User's Playback:**

  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.play
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/start-a-users-playback/

- **Toggle Shuffle For User's Playback:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.shuffle
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/toggle-shuffle-for-users-playback/

- **Transfer a User's Playback:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.playback_transfer
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/player/transfer-a-users-playback/

5.9 Playlists

- **Add playlist items:**
  - Docs: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.add_playlist_tracks
  - Web API Reference: https://developer.spotify.com/documentation/web-api/reference/playlists/add-tracks-to-playlist/

- **Edit playlist:**
  - Pyfy: https://developer.spotify.com/documentation/web-api/reference/playlists/change-playlist-details/
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/change-playlist-details/

- **Create playlist:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.create_playlist
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/create-playlist/

- **List a user's playlists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.user_playlists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/get-a-list-of-current-users-playlists/

- **Playlist cover:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/get-playlist-cover/

- **List a playlist:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.playlist
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/get-playlist/

- **List a playlist items:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.playlist_tracks

  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/get-playlists-tracks/

- **Remove playlist items:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.delete_playlist_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/remove-tracks-playlist/
- **Reorder playlist items:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.reorder_playlist_track
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/reorder-playlists-tracks/
- **Replace playlist items:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.replace_playlist_tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/replace-playlists-tracks/
- **Upload custom playlist cover image:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/upload-custom-playlist-cover/
- **List current user playlists:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.user_playlists
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/playlists/get-a-list-of-current-users-playlists/

5.10 Search

- **Search for an item:**
  - Pyfy: https://developer.spotify.com/documentation/web-api/reference/search/search/
  - Web API reference: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.search

5.11 Shows

- **Get a Show:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/shows/get-a-show/
- **Get Several Shows:**
  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/shows/get-several-shows/
- **Get a Show's Episodes:**

  - Pyfy: **TODO**
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/shows/get-shows-episodes/

5.12 Tracks

- **Get Audio Analysis for a Track:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.track_audio_analysis
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-analysis/
- **Get Audio Features for a Track:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.tracks_audio_features
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-features/
- **Get Audio Features for Several Tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.tracks_audio_features
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/tracks/get-several-audio-features/
- **Get Several Tracks:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/tracks/get-several-tracks/
- **Get a Track:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.tracks
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/tracks/get-track/

5.13 Users Profile

- **Get Current User's Profile:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.user_profile
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/users-profile/get-current-users-profile/
- **Get a User's Profile:**
  - Pyfy: https://pyfy.readthedocs.io/en/latest/#pyfy.sync_client.Spotify.user_profile
  - Web API reference: https://developer.spotify.com/documentation/web-api/reference/users-profile/get-users-profile/

CHAPTER

SIX

PAGINATION

```
from pyfy import Spotify

user_creds = {'access_token': '...', 'refresh_token': '....'}

spt = Spotify(user_creds=user_creds)

user_top_tracks = spt.user_top_tracks(limit=5)

next_page_1 = spt.next_page(user_top_tracks)
next_page_2 = spt.next_page(next_page_1)

previous_page_1 = spt.previous_page(next_page_2)
previous_page_1 === next_page_1  # True
```

CHAPTER

SEVEN

SYNC CLIENT API

class pyfy.sync_client.**Spotify**(*access_token=None*, *client_creds=<pyfy.creds.ClientCreds object>*, *user_creds=None*, *ensure_user_auth=False*, *proxies={}*, *timeout=7*, *max_retries=10*, *backoff_factor=0.1*, *default_to_locale=True*, *cache=True*, *populate_user_creds=True*)

Bases: pyfy.base_client._BaseClient

Spotify's Synchronous Client

Parameters

- **client_creds** (pyfy.creds.ClientCreds) – A client credentials model
- **user_creds** (pyfy.creds.UserCreds) – A user credentials model
- **ensure_user_auth** (*bool*) –
  - Whether or not to fail upon instantiation if user_creds provided where invalid and not refresheable.
  - Default: False
- **proxies** –
  - socks or http proxies
  - http://docs.python-requests.org/en/master/user/advanced/#proxies & http://docs.python-requests.org/en/master/user/advanced/#socks
- **timeout** (*int*) –
  - Seconds before request raises a timeout error
  - Default: 7
- **max_retries** (*int*) –
  - Max retries before a request fails
  - Default: 10
- **backoff_factor** (*float*) –
  - Factor by which requests delays the next request when encountring a 429 too-many-requests error
  - Default: 0.1
- **default_to_locale** (*bool*) –
  - Will pass methods decorated with @_default_to_locale the user's locale if available.
  - Default: True

- **cache** –
  - Whether or not to cache HTTP requests for the user
  - Default: True
- **populate_user_creds** (*bool*) –
  - Sets user_creds info from Spotify to client's user_creds object. e.g. country.
  - Default: True

IS_ASYNC = False

add_playlist_tracks(**args*, ***kwargs*)
Add tracks to a playlist

Parameters

- **playlist_id** –
  - Required
- **track_ids** (*str, list*) –
  - Required
- **position** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

album_tracks(**args*, ***kwargs*)
List tracks of an album

Parameters

- **album_id** (*str*) –
  - Required
- **market** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

albums(**args*, ***kwargs*)
List Albums

Parameters

- **album_ids** (*str, list*) –

– Required

- **market** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

artist_albums(**args*, ***kwargs*)
List albums of an artist

Parameters

- **artist_id** (*str*) –
  - Required
- **include_groups** –
  - Optional
- **market** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

artist_related_artists(**args*, ***kwargs*)
List artists related to an artist

Parameters **artist_id** (*str*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

artist_top_tracks(**args*, ***kwargs*)
List top tracks of an artist

Parameters

- **artist_id** (*str*) –
  - Required
- **country** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

artists(**args*, ***kwargs*)
List artists

Parameters **artist_ids** (*str, list*) –

Returns

- Required

Return type dict

Raises *pyfy.excs.ApiError* –

auth_uri(*state=None*, *client_id=None*, *scopes=None*, *redirect_uri=None*, *show_dialog=None*, *response_type=None*)
Generates OAuth2 URI for authentication Arguments will default to the attributes of self.client_creds

Parameters

- **client_id** (*str*) – OAuth2 client_id (Defaults to self.client_creds.client_id)
- **scopes** (*list*) – OAuth2 scopes. (Defaults to self.client_creds.scopes)
- **redirect_uri** (*str*) – OAuth2 redirect uri. (Defaults to self.client_creds.redirect_uri)
- **show_dialog** (*bool*) – if set to false, Spotify will not show a new authentication request if user already authorized the client (Defaults to self.client_creds.show_dialog)
- **response_type** (*str*) – Defaults to "code" for OAuth2 Authorization Code Flow

Returns OAuth2 Auth URI

Return type str

authorize_client_creds(*client_creds=None*)
Authorize with client credentials oauth flow i.e. Only with client secret and client id.

Call this to send request using client credentials.

https://developer.spotify.com/documentation/general/guides/authorization-guide/

Note: This will give you limited access to most endpoints

Parameters **client_creds** (pyfy.creds.ClientCreds) – Client Credentials object. Defaults to `self.client_creds`.

Raises **pyfy.excs.AuthErrror** –

available_genre_seeds(**args*, ***kwargs*)
Available genre seeds

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

build_user_creds(*grant*, *set_user_creds=True*)
Second part of OAuth2 authorization code flow, Raises an AuthError if unauthorized

Parameters

- **grant** (*str*) – Code returned to user after authorizing your application
- **set_user_creds** (*bool*) – Whether or not to set the user created to the client as the current active user

Returns User Credentials Model

Return type *pyfy.creds.UserCreds*

categories(**args*, ***kwargs*)
List Categories

Parameters

- **country** –
  - Optional
- **locale** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

category(**args*, ***kwargs*)
List Category

Parameters

- **category_id** –
  - Required
- **country** –
  - Optional
- **locale** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

category_playlist(**args*, ***kwargs*)
List playlists from a category

Parameters

- **category_id** –
  - Required
- **country** –

- Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

create_playlist(**args*, ***kwargs*)
Creates a playlist

Parameters

- **name** –
  - Required
- **description** –
  - Optional
- **public** –
  - Optional
  - Default: False
- **collaborative** –
  - Optional
  - default: False

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

currently_playing(**args*, ***kwargs*)
Lists currenly playing

Parameters **market** (*str*) –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

currently_playing_info(**args*, ***kwargs*)
Lists currently playing info

Parameters **market** (*str*) –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

delete_albums(**args*, ***kwargs*)
Delete Albums

Parameters **album_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

delete_playlist(**args*, ***kwargs*)
An alias to unfollow_playlist

Parameters **playlist_id** –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

delete_playlist_tracks(**args*, ***kwargs*)
Delete tracks from a playlist

https://developer.spotify.com/console/delete-playlist-tracks/

Examples

`track_ids` types supported:

```
1) 'track_id'
2) ['track_id', 'track_id', 'track_id']
3) [
    {
        'id': track_id,
        'positions': [
            position1, position2
        ]
    },
    {
        'id': track_id,
        'positions': position1
    },
    track_id
]
```

Parameters

- **playlist_id** –
  - Required
- **track_ids** –
  - Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`delete_tracks`(**args*, ***kwargs*)
Delete user's tracks

Parameters **track_ids** (*str, list*) –

Returns

- Required

Return type dict

Raises *pyfy.excs.ApiError* –

`devices`(**args*, ***kwargs*)
Lists user's devices

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`featured_playlists`(**args*, ***kwargs*)
Featured Playlists

Parameters

- **country** –
  - Optional
- **locale** –
  - Optional
- **timestamp** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`follow_artists`(**args*, ***kwargs*)
Follow an artist(s)

Parameters **artist_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follow_playlist(**args*, ***kwargs*)
Follows a playlist

Parameters

- **playlist_id** –
  - Required
- **public** –
  - Optional
  - Default: False

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follow_users(**args*, ***kwargs*)
Follow a user

Parameters **user_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

followed_artists(**args*, ***kwargs*)
List artists followed by current user

Parameters

- **after** –
  - Optional
- **limit** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follows_artists(**args*, ***kwargs*)
Whether or not current user follows an artist(s)

Parameters **artist_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follows_playlist(**args*, ***kwargs*)
Lists whether or not user follows a playlist

Parameters

- **playlist_id** –
  - Required
- **user_ids** (*list, str*) –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follows_users(**args*, ***kwargs*)
Whether or not current user follows a user(s)

Parameters **user_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

property is_active
Checks if user_creds or client_creds are valid (depending on who was last set)

property is_oauth_ready
Whether Client Credentials have enough information to perform OAuth2 Authorization Code FLow

Returns

bool:

property is_premium
Checks whether user is premium or not

Returns

Return type bool

me(**args*, ***kwargs*)
List current user's profile

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

new_releases(**args*, ***kwargs*)
New Releases

Parameters

- **country** –
  - Optional
- **limit** –
  - Optional
- **offset** –

– Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

next(**args*, ***kwargs*)
Next playback

Parameters **device_id** –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

next_page(**args*, ***kwargs*)
Next Page

Note:

- You can either provide a response or a url
- Providing a URL will be slightly faster as Pyfy will not have to search for the key in the response dict

---

Parameters

- **response** (*dict*) –
  - Optional
- **url** (*str*) –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

owns_albums(**args*, ***kwargs*)
Whether or not current user owns an album(s)

Parameters **album_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

owns_tracks(**args*, ***kwargs*)
Lists whether or not current user owns tracks

Parameters **track_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

pause(**args*, ***kwargs*)
Pauses playback

Parameters **device_id** (*str*) –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

play(**args*, ***kwargs*)
Starts playback

Play a list of one or more tracks, or a specific artist, album or playlist. Only one of track_ids, album_id, artist_id, playlist_id should be specified. Start playback at offset_position OR offset_uri, only if artist_id is not being used.

Parameters

- **track_ids** (*list, tuple, str*) –
  - Optional
  - List, string or tuple containing track ID(s).
- **album_id** (*str*) –
  - Optional
- **artist_id** (*str*) –
  - Optional
- **playlist_id** (*str*) –
  - Optional
- **device_id** (*str*) –
  - Optional
- **offset_position** (*int*) –
  - Optional
- **offset_uri** (*str*) –
  - Optional
- **poition_ms** (*int*) –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

playback_transfer(**args*, ***kwargs*)
Transfer playback to another device

Parameters **device_ids** (*list, str*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

playlist(**args*, ***kwargs*)
Lists playlist

Parameters

- **playlist_id** –
  - Required
- **market** –
  - Optional
- **fields** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

playlist_cover(**args*, ***kwargs*)
Get a Playlist Cover Image

Parameters **playlist_id** –

- Required

Returns

Return type list

Raises *pyfy.excs.ApiError* –

playlist_tracks(**args*, ***kwargs*)
List tracks in a playlist

Parameters

- **playlist_id** –
  - Required
- **market** –
  - Optional
- **fields** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

populate_user_creds()
Populates self.user_creds with Spotify's info on user. Data is fetched from self.me() and set to user recursively

previous(**args*, ***kwargs*)
Previous Playback

Parameters **device_id** –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

previous_page(**args*, ***kwargs*)
Previous Page

Note:

- You can either provide a response or a url
- Providing a URL will be slightly faster as Pyfy will not have to search for the key in the response dict

---

Parameters

- **response** (*dict*) –
  - Optional
- **url** (*str*) –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

queue(**args*, ***kwargs*)
Add an item to the end of the user's current playback queue

Parameters

- **track_id** (*str*) –
  - Required
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

recently_played_tracks(**args*, ***kwargs*)
Lists recently played tracks

Parameters

- **limit** (*int*) –
  - Optional
- **after** –
  - Optional
- **before** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

recommendations(**args*, ***kwargs*)
List Recommendations

https://developer.spotify.com/documentation/web-api/reference/browse/get-recommendations/

Parameters

- **limit** –
  - Optional
- **market** –
  - Optional
- **seed_artists** –
  - Optional
- **seed_genres** –
  - Optional
- **seed_tracks** –
  - Optional
- **min_acousticness** –
  - Optional
- **max_acousticness** –
  - Optional
- **target_acousticness** –
  - Optional
- **min_danceability** –
  - Optional
- **max_danceability** –
  - Optional
- **target_danceability** –

  - Optional
- **min_duration_ms** –
  - Optional
- **max_duration_ms** –
  - Optional
- **target_duration_ms** –
  - Optional
- **min_energy** –
  - Optional
- **max_energy** –
  - Optional
- **target_energy** –
  - Optional
- **min_instrumentalness** –
  - Optional
- **max_instrumentalness** –
  - Optional
- **target_instrumentalness** –
  - Optional
- **min_key** –
  - Optional
- **max_key** –
  - Optional
- **target_key** –
  - Optional
- **min_liveness** –
  - Optional
- **max_liveness** –
  - Optional
- **target_liveness** –
  - Optional
- **min_loudness** –
  - Optional
- **max_loudness** –
  - Optional
- **target_loudness** –

  - Optional
- **min_mode** –
  - Optional
- **max_mode** –
  - Optional
- **target_mode** –
  - Optional
- **min_popularity** –
  - Optional
- **max_popularity** –
  - Optional
- **target_popularity** –
  - Optional
- **min_speechiness** –
  - Optional
- **max_speechiness** –
  - Optional
- **target_speechiness** –
  - Optional
- **min_tempo** –
  - Optional
- **max_tempo** –
  - Optional
- **target_tempo** –
  - Optional
- **min_time_signature** –
  - Optional
- **max_time_signature** –
  - Optional
- **target_time_signature** –
  - Optional
- **min_valence** –
  - Optional
- **max_valence** –
  - Optional
- **target_valence** –

– Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

reorder_playlist_track(**args*, ***kwargs*)
Reorder tracks in a playlist

Parameters

- **playlist_id** –
  - Required
- **range_start** –
  - Optional
- **range_length** –
  - Optional
- **insert_before** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

repeat(**args*, ***kwargs*)
Toggle repeat

Parameters

- **state** –
  - Optional
  - Default: 'context'
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

replace_playlist_tracks(**args*, ***kwargs*)
Replace all tracks of a playlist with tracks of your choice

Parameters

- **playlist_id** –
  - Required
- **track_ids** –
  - track_ids not full URIs
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

save_albums(**args*, ***kwargs*)
Save Albums

Parameters **album_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

save_tracks(**args*, ***kwargs*)
Save tracks

Parameters **track_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

search(**args*, ***kwargs*)
Search

Examples

tracks parameter example:

```
'track' or ['track'] or 'artist' or ['track','artist']
```

Parameters

- **q** –
  - Query
  - Required
- **types** –
  - Optional
  - Default: `'track'`
- **market** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

seek(**args*, ***kwargs*)
Seek Playback

Parameters

- **posiotion_ms** –
  - Required
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

shuffle(**args*, ***kwargs*)
Shuffle Playback

Parameters

- **state** –
  - Optional
  - Default: True
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

track_audio_analysis(**args*, ***kwargs*)
List audio analysis of a track

Parameters **track_id** –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

tracks(**args*, ***kwargs*)
List tracks

Parameters

- **track_ids** (*str, list*) –
  - Required
- **market** –

– Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

tracks_audio_features(**args*, ***kwargs*)
List audio features of tracks

Parameters **track_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

unfollow_artists(**args*, ***kwargs*)
Unfollow artist(s)

Parameters **artist_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

unfollow_playlist(**args*, ***kwargs*)
Unfollow a playlist

Parameters **playlist_id** –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

unfollow_users(**args*, ***kwargs*)
Unfollow user(s)

Parameters **user_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

update_playlist(**args*, ***kwargs*)
Updates a playlist

Parameters

- **playlist_id** –
  – Required
- **name** –

- Optional
- **description** –
  - Optional
- **public** –
  - Optional
- **collaborative** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

user_albums(**args*, ***kwargs*)
Albums owned by current user

Parameters

- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

property user_creds

user_playlists(**args*, ***kwargs*)
Lists playlists owned by a user

Parameters

- **user_id** –
  - Optional
  - Defaults to user's
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

user_profile(**args*, ***kwargs*)
List a user's profile

Parameters **user_id** (*str*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`user_top_artists`(**args*, ***kwargs*)
List top artists of a user

Parameters

- **time_range** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`user_top_tracks`(**args*, ***kwargs*)
List top tracks of a user

Parameters

- **time_range** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`user_tracks`(**args*, ***kwargs*)
List user's tracks

Parameters

- **market** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`volume`(**args*, ***kwargs*)
Change volume

Parameters

- **volume_percent** (*int*) –
  - Required
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

CHAPTER

EIGHT

ASYNC CLIENT API

class pyfy.sync_client.**Spotify**(*access_token=None*, *client_creds=<pyfy.creds.ClientCreds object>*, *user_creds=None*, *ensure_user_auth=False*, *proxies={}*, *timeout=7*, *max_retries=10*, *backoff_factor=0.1*, *default_to_locale=True*, *cache=True*, *populate_user_creds=True*)

Bases: pyfy.base_client._BaseClient

Spotify's Synchronous Client

Parameters

- **client_creds** (pyfy.creds.ClientCreds) – A client credentials model
- **user_creds** (pyfy.creds.UserCreds) – A user credentials model
- **ensure_user_auth** (*bool*) –
  - Whether or not to fail upon instantiation if user_creds provided where invalid and not refresheable.
  - Default: False
- **proxies** –
  - socks or http proxies
  - http://docs.python-requests.org/en/master/user/advanced/#proxies & http://docs.python-requests.org/en/master/user/advanced/#socks
- **timeout** (*int*) –
  - Seconds before request raises a timeout error
  - Default: 7
- **max_retries** (*int*) –
  - Max retries before a request fails
  - Default: 10
- **backoff_factor** (*float*) –
  - Factor by which requests delays the next request when encountring a 429 too-many-requests error
  - Default: 0.1
- **default_to_locale** (*bool*) –
  - Will pass methods decorated with @_default_to_locale the user's locale if available.
  - Default: True

- **cache** –
  - Whether or not to cache HTTP requests for the user
  - Default: True
- **populate_user_creds** (*bool*) –
  - Sets user_creds info from Spotify to client's user_creds object. e.g. country.
  - Default: True

IS_ASYNC = False

add_playlist_tracks(**args*, ***kwargs*)
Add tracks to a playlist

Parameters

- **playlist_id** –
  - Required
- **track_ids** (*str, list*) –
  - Required
- **position** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

album_tracks(**args*, ***kwargs*)
List tracks of an album

Parameters

- **album_id** (*str*) –
  - Required
- **market** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

albums(**args*, ***kwargs*)
List Albums

Parameters

- **album_ids** (*str, list*) –

– Required

- **market** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

artist_albums(**args*, ***kwargs*)
List albums of an artist

Parameters

- **artist_id** (*str*) –
  - Required
- **include_groups** –
  - Optional
- **market** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

artist_related_artists(**args*, ***kwargs*)
List artists related to an artist

Parameters **artist_id** (*str*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

artist_top_tracks(**args*, ***kwargs*)
List top tracks of an artist

Parameters

- **artist_id** (*str*) –
  - Required
- **country** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

artists(**args*, ***kwargs*)
List artists

Parameters **artist_ids** (*str, list*) –

Returns

- Required

Return type dict

Raises *pyfy.excs.ApiError* –

auth_uri(*state=None*, *client_id=None*, *scopes=None*, *redirect_uri=None*, *show_dialog=None*, *response_type=None*)
Generates OAuth2 URI for authentication Arguments will default to the attributes of self.client_creds

Parameters

- **client_id** (*str*) – OAuth2 client_id (Defaults to self.client_creds.client_id)
- **scopes** (*list*) – OAuth2 scopes. (Defaults to self.client_creds.scopes)
- **redirect_uri** (*str*) – OAuth2 redirect uri. (Defaults to self.client_creds.redirect_uri)
- **show_dialog** (*bool*) – if set to false, Spotify will not show a new authentication request if user already authorized the client (Defaults to self.client_creds.show_dialog)
- **response_type** (*str*) – Defaults to "code" for OAuth2 Authorization Code Flow

Returns OAuth2 Auth URI

Return type str

authorize_client_creds(*client_creds=None*)
Authorize with client credentials oauth flow i.e. Only with client secret and client id.

Call this to send request using client credentials.

https://developer.spotify.com/documentation/general/guides/authorization-guide/

Note: This will give you limited access to most endpoints

Parameters **client_creds** (pyfy.creds.ClientCreds) – Client Credentials object. Defaults to `self.client_creds`.

Raises **pyfy.excs.AuthErrror** –

available_genre_seeds(**args*, ***kwargs*)
Available genre seeds

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

build_user_creds(*grant*, *set_user_creds=True*)
Second part of OAuth2 authorization code flow, Raises an AuthError if unauthorized

Parameters

- **grant** (*str*) – Code returned to user after authorizing your application
- **set_user_creds** (*bool*) – Whether or not to set the user created to the client as the current active user

Returns User Credentials Model

Return type *pyfy.creds.UserCreds*

categories(**args*, ***kwargs*)
List Categories

Parameters

- **country** –
  - Optional
- **locale** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

category(**args*, ***kwargs*)
List Category

Parameters

- **category_id** –
  - Required
- **country** –
  - Optional
- **locale** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

category_playlist(**args*, ***kwargs*)
List playlists from a category

Parameters

- **category_id** –
  - Required
- **country** –

- Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

create_playlist(**args*, ***kwargs*)
Creates a playlist

Parameters

- **name** –
  - Required
- **description** –
  - Optional
- **public** –
  - Optional
  - Default: False
- **collaborative** –
  - Optional
  - default: False

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

currently_playing(**args*, ***kwargs*)
Lists currenly playing

Parameters **market** (*str*) –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

currently_playing_info(**args*, ***kwargs*)
Lists currently playing info

Parameters **market** (*str*) –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

delete_albums(**args*, ***kwargs*)
Delete Albums

Parameters **album_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

delete_playlist(**args*, ***kwargs*)
An alias to unfollow_playlist

Parameters **playlist_id** –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

delete_playlist_tracks(**args*, ***kwargs*)
Delete tracks from a playlist

https://developer.spotify.com/console/delete-playlist-tracks/

Examples

`track_ids` types supported:

```
1) 'track_id'
2) ['track_id', 'track_id', 'track_id']
3) [
    {
        'id': track_id,
        'positions': [
            position1, position2
        ]
    },
    {
        'id': track_id,
        'positions': position1
    },
    track_id
]
```

Parameters

- **playlist_id** –
  - Required
- **track_ids** –
  - Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`delete_tracks`(**args*, ***kwargs*)
Delete user's tracks

Parameters **track_ids** (*str, list*) –

Returns

- Required

Return type dict

Raises *pyfy.excs.ApiError* –

`devices`(**args*, ***kwargs*)
Lists user's devices

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`featured_playlists`(**args*, ***kwargs*)
Featured Playlists

Parameters

- **country** –
  - Optional
- **locale** –
  - Optional
- **timestamp** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`follow_artists`(**args*, ***kwargs*)
Follow an artist(s)

Parameters **artist_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follow_playlist(**args*, ***kwargs*)
Follows a playlist

Parameters

- **playlist_id** –
  - Required
- **public** –
  - Optional
  - Default: False

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follow_users(**args*, ***kwargs*)
Follow a user

Parameters **user_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

followed_artists(**args*, ***kwargs*)
List artists followed by current user

Parameters

- **after** –
  - Optional
- **limit** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follows_artists(**args*, ***kwargs*)
Whether or not current user follows an artist(s)

Parameters **artist_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follows_playlist(**args*, ***kwargs*)
Lists whether or not user follows a playlist

Parameters

- **playlist_id** –
  - Required
- **user_ids** (*list, str*) –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

follows_users(**args*, ***kwargs*)
Whether or not current user follows a user(s)

Parameters **user_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

property is_active
Checks if user_creds or client_creds are valid (depending on who was last set)

property is_oauth_ready
Whether Client Credentials have enough information to perform OAuth2 Authorization Code FLow

Returns

bool:

property is_premium
Checks whether user is premium or not

Returns

Return type bool

me(**args*, ***kwargs*)
List current user's profile

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

new_releases(**args*, ***kwargs*)
New Releases

Parameters

- **country** –
  - Optional
- **limit** –
  - Optional
- **offset** –

– Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

next(**args*, ***kwargs*)
Next playback

Parameters **device_id** –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

next_page(**args*, ***kwargs*)
Next Page

Note:

- You can either provide a response or a url
- Providing a URL will be slightly faster as Pyfy will not have to search for the key in the response dict

---

Parameters

- **response** (*dict*) –
  - Optional
- **url** (*str*) –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

owns_albums(**args*, ***kwargs*)
Whether or not current user owns an album(s)

Parameters **album_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

owns_tracks(**args*, ***kwargs*)
Lists whether or not current user owns tracks

Parameters **track_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

pause(**args*, ***kwargs*)
Pauses playback

Parameters **device_id** (*str*) –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

play(**args*, ***kwargs*)
Starts playback

Play a list of one or more tracks, or a specific artist, album or playlist. Only one of track_ids, album_id, artist_id, playlist_id should be specified. Start playback at offset_position OR offset_uri, only if artist_id is not being used.

Parameters

- **track_ids** (*list, tuple, str*) –
  - Optional
  - List, string or tuple containing track ID(s).
- **album_id** (*str*) –
  - Optional
- **artist_id** (*str*) –
  - Optional
- **playlist_id** (*str*) –
  - Optional
- **device_id** (*str*) –
  - Optional
- **offset_position** (*int*) –
  - Optional
- **offset_uri** (*str*) –
  - Optional
- **poition_ms** (*int*) –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

playback_transfer(**args*, ***kwargs*)
Transfer playback to another device

Parameters **device_ids** (*list, str*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

playlist(**args*, ***kwargs*)
Lists playlist

Parameters

- **playlist_id** –
  - Required
- **market** –
  - Optional
- **fields** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

playlist_cover(**args*, ***kwargs*)
Get a Playlist Cover Image

Parameters **playlist_id** –

- Required

Returns

Return type list

Raises *pyfy.excs.ApiError* –

playlist_tracks(**args*, ***kwargs*)
List tracks in a playlist

Parameters

- **playlist_id** –
  - Required
- **market** –
  - Optional
- **fields** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

populate_user_creds()
Populates self.user_creds with Spotify's info on user. Data is fetched from self.me() and set to user recursively

previous(**args*, ***kwargs*)
Previous Playback

Parameters **device_id** –

- Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

previous_page(**args*, ***kwargs*)
Previous Page

Note:

- You can either provide a response or a url
- Providing a URL will be slightly faster as Pyfy will not have to search for the key in the response dict

---

Parameters

- **response** (*dict*) –
  - Optional
- **url** (*str*) –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

queue(**args*, ***kwargs*)
Add an item to the end of the user's current playback queue

Parameters

- **track_id** (*str*) –
  - Required
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

recently_played_tracks(**args*, ***kwargs*)
Lists recently played tracks

Parameters

- **limit** (*int*) –
  - Optional
- **after** –
  - Optional
- **before** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

recommendations(**args*, ***kwargs*)
List Recommendations

https://developer.spotify.com/documentation/web-api/reference/browse/get-recommendations/

Parameters

- **limit** –
  - Optional
- **market** –
  - Optional
- **seed_artists** –
  - Optional
- **seed_genres** –
  - Optional
- **seed_tracks** –
  - Optional
- **min_acousticness** –
  - Optional
- **max_acousticness** –
  - Optional
- **target_acousticness** –
  - Optional
- **min_danceability** –
  - Optional
- **max_danceability** –
  - Optional
- **target_danceability** –

  - Optional
- **min_duration_ms** –
  - Optional
- **max_duration_ms** –
  - Optional
- **target_duration_ms** –
  - Optional
- **min_energy** –
  - Optional
- **max_energy** –
  - Optional
- **target_energy** –
  - Optional
- **min_instrumentalness** –
  - Optional
- **max_instrumentalness** –
  - Optional
- **target_instrumentalness** –
  - Optional
- **min_key** –
  - Optional
- **max_key** –
  - Optional
- **target_key** –
  - Optional
- **min_liveness** –
  - Optional
- **max_liveness** –
  - Optional
- **target_liveness** –
  - Optional
- **min_loudness** –
  - Optional
- **max_loudness** –
  - Optional
- **target_loudness** –

  - Optional
- **min_mode** –
  - Optional
- **max_mode** –
  - Optional
- **target_mode** –
  - Optional
- **min_popularity** –
  - Optional
- **max_popularity** –
  - Optional
- **target_popularity** –
  - Optional
- **min_speechiness** –
  - Optional
- **max_speechiness** –
  - Optional
- **target_speechiness** –
  - Optional
- **min_tempo** –
  - Optional
- **max_tempo** –
  - Optional
- **target_tempo** –
  - Optional
- **min_time_signature** –
  - Optional
- **max_time_signature** –
  - Optional
- **target_time_signature** –
  - Optional
- **min_valence** –
  - Optional
- **max_valence** –
  - Optional
- **target_valence** –

– Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

reorder_playlist_track(**args*, ***kwargs*)
Reorder tracks in a playlist

Parameters

- **playlist_id** –
  - Required
- **range_start** –
  - Optional
- **range_length** –
  - Optional
- **insert_before** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

repeat(**args*, ***kwargs*)
Toggle repeat

Parameters

- **state** –
  - Optional
  - Default: 'context'
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

replace_playlist_tracks(**args*, ***kwargs*)
Replace all tracks of a playlist with tracks of your choice

Parameters

- **playlist_id** –
  - Required
- **track_ids** –
  - track_ids not full URIs
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

save_albums(**args*, ***kwargs*)
Save Albums

Parameters **album_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

save_tracks(**args*, ***kwargs*)
Save tracks

Parameters **track_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

search(**args*, ***kwargs*)
Search

Examples

tracks parameter example:

```
'track' or ['track'] or 'artist' or ['track','artist']
```

Parameters

- **q** –
  - Query
  - Required
- **types** –
  - Optional
  - Default: `'track'`
- **market** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

seek(**args*, ***kwargs*)
Seek Playback

Parameters

- **posiotion_ms** –
  - Required
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

shuffle(**args*, ***kwargs*)
Shuffle Playback

Parameters

- **state** –
  - Optional
  - Default: True
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

track_audio_analysis(**args*, ***kwargs*)
List audio analysis of a track

Parameters **track_id** –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

tracks(**args*, ***kwargs*)
List tracks

Parameters

- **track_ids** (*str, list*) –
  - Required
- **market** –

– Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

tracks_audio_features(**args*, ***kwargs*)
List audio features of tracks

Parameters **track_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

unfollow_artists(**args*, ***kwargs*)
Unfollow artist(s)

Parameters **artist_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

unfollow_playlist(**args*, ***kwargs*)
Unfollow a playlist

Parameters **playlist_id** –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

unfollow_users(**args*, ***kwargs*)
Unfollow user(s)

Parameters **user_ids** (*str, list*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

update_playlist(**args*, ***kwargs*)
Updates a playlist

Parameters

- **playlist_id** –
  – Required
- **name** –

- Optional
- **description** –
  - Optional
- **public** –
  - Optional
- **collaborative** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

user_albums(**args*, ***kwargs*)
Albums owned by current user

Parameters

- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

property user_creds

user_playlists(**args*, ***kwargs*)
Lists playlists owned by a user

Parameters

- **user_id** –
  - Optional
  - Defaults to user's
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

user_profile(**args*, ***kwargs*)
List a user's profile

Parameters **user_id** (*str*) –

- Required

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`user_top_artists`(**args*, ***kwargs*)
List top artists of a user

Parameters

- **time_range** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`user_top_tracks`(**args*, ***kwargs*)
List top tracks of a user

Parameters

- **time_range** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`user_tracks`(**args*, ***kwargs*)
List user's tracks

Parameters

- **market** –
  - Optional
- **limit** –
  - Optional
- **offset** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

`volume`(**args*, ***kwargs*)
Change volume

Parameters

- **volume_percent** (*int*) –
  - Required
- **device_id** –
  - Optional

Returns

Return type dict

Raises *pyfy.excs.ApiError* –

CHAPTER

NINE

EXCEPTIONS API

exception pyfy.excs.**ApiError**(*msg*, *http_response=None*, *http_request=None*, *e=None*)

Bases: *pyfy.excs.SpotifyError*

Almost any HTTP error other than 401 raises this error https://developer.spotify.com/documentation/web-api/#response-schema // regular error object

msg

Error msg returned from Spotify

Type str

http_response

Full HTTP response

http_request

Full HTTP request that caused this error

code

HTTP status code

Type int

args

with_traceback()

Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception pyfy.excs.**AuthError**(*msg*, *http_response=None*, *http_request=None*, *e=None*)

Bases: *pyfy.excs.SpotifyError*

Raised when a 401 or any Authentication error is encountered https://developer.spotify.com/documentation/web-api/#response-schema // authentication error object

msg

Error msg returned from Spotify

Type str

http_response

Full HTTP response

http_request

Full HTTP request that caused this error

code

HTTP status code

Type int

args

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

exception pyfy.excs.SpotifyError
Bases: Exception

Base error class for ApiError and AuthError

args

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

CHAPTER

TEN

CREDENTIALS API

pyfy.creds.**ALL_SCOPES = ['streaming', 'app-remote-control', 'user-follow-modify', 'user-follow-read', 'playlist-read-private', 'playlist-modify-private', 'playlist-read-collaborative', 'playlist-modify-public', 'user-modify-playback-state', 'user-read-playback-state', 'user-read-currently-playing', 'user-read-private', 'user-read-email', 'user-library-read', 'user-library-modify', 'user-top-read', 'user-read-recently-played']**

List of all scopes provided by Spotify

class pyfy.creds.**ClientCreds**(*client_id=None*, *client_secret=None*, *scopes=None*, *redirect_uri=None*, *show_dialog=False*)

Bases: pyfy.creds._Creds

OAuth2 Client Credentials

Parameters

- **client_id** (*str*) – OAuth2 client_id
- **client_secret** (*str*) – OAuth2 client_secret
- **scopes** (*list*) – OAuth2 scopes. Defaults to all scopes
- **redirect_uri** (*str*) – OAuth2 redirect uri. Defaults to http://localhost
- **show_dialog** (*bool*) – if set to false, Spotify will not show a new authentication request if user already authorized the client

property access_is_expired

Returns:

bool: Whether access token expired or not

get(*key*)

property is_oauth_ready

load_from_env()

Load client creds from OS environment

SPOTIFY_CLIENT_ID, SPOTIFY_CLIENT_SECRET and SPOTiFY_REDIRECT_URI environment variables must be present

load_from_json(*path=None*, *name=None*)

Loads credentials from JSON file

Parameters

- **path** (*str*) – path of the directory the file is located in
- **name** (*str*) – name of the file.

pickle(*path=None*, *name=None*)
Pickles Credentials

Parameters

- **path** (*str*) – path of the directory to store pickle in
- **name** (*str*) – name of the file.

save_as_json(*path=None*, *name=None*)
Saves credentials as a json file

Parameters

- **path** (*str*) – path of the directory you want to save the file in
- **name** (*str*) – name of the file.

classmethod unpickle(*path=None*, *name=None*)
Loads a Credentials Pickle from file

Parameters

- **path** (*str*) – path of the directory you want to unpickle from
- **name** (*str*) – name of the file.

class pyfy.creds.**UserCreds**(*access_token=None*, *refresh_token=None*, *scopes=None*, *expiry=None*, *user_id=None*)
Bases: pyfy.creds._Creds

OAuth2 User Credentials + Spotify's User info

Note: For convenience, if you set the populate_user_creds flag to True in any of Pyfy's clients, this will set all of Spotify's basic information on user to this model

---

Parameters

- **access_token** (*str*) – OAuth2 access token
- **refresh_token** (*str*) – OAuth2 refresh token
- **scopes** (*list*) – OAuth2 scopes
- **expiry** (*datetime.datetime*) – Datetime access token expires
- **user_id** (*str*) – Not to be confused with OpenID, this is the user's Spotify ID

birthdate
From Spotify's /me endpoint

Type str

country
From Spotify's /me endpoint

Type str

display_name
From Spotify's /me endpoint

Type str

email
From Spotify's /me endpoint

Type str

external_urls
From Spotify's /me endpoint

Type dict

followers
From Spotify's /me endpoint

Type dict

href
From Spotify's /me endpoint

Type str

id
From Spotify's /me endpoint

Type str

images
From Spotify's /me endpoint

Type list

product
From Spotify's /me endpoint

Type str

type
From Spotify's /me endpoint

Type str

uri
From Spotify's /me endpoint

Type str

property access_is_expired
Returns:

bool: Whether access token expired or not

get(*key*)

load_from_env()
Load user creds from env

SPOTIFY_ACCESS_TOKEN and SPOTIFY_REFRESH_TOKEN environment variables must be present

This method will not fail if it didn't find a refresh token, but will fail if no access token was found

load_from_json(*path=None*, *name=None*)
Loads credentials from JSON file

Parameters

- **path** (*str*) – path of the directory the file is located in
- **name** (*str*) – name of the file.

pickle(*path=None*, *name=None*)
Pickles Credentials

Parameters

- **path** (*str*) – path of the directory to store pickle in
- **name** (*str*) – name of the file.

save_as_json(*path=None*, *name=None*)
Saves credentials as a json file

Parameters

- **path** (*str*) – path of the directory you want to save the file in
- **name** (*str*) – name of the file.

classmethod unpickle(*path=None*, *name=None*)
Loads a Credentials Pickle from file

Parameters

- **path** (*str*) – path of the directory you want to unpickle from
- **name** (*str*) – name of the file.

CHAPTER

ELEVEN

TESTING

11.1 Unit tests

```
$ tox
```

11.2 Integration tests

1. Open tox.ini and change thoee values to:
    1. `SPOTIFY_CLIENT_ID` Create an app
    2. `SPOTIFY_CLIENT_SECRET` Create an app
    3. `SPOTIFY_ACCESS_TOKEN` Get one or perform OAuth2 Auth Code Flow.

        Note: Check all scopes when getting an access token.

    4. `SPOTIFY_REFRESH_TOKEN`

        Note: To avoid manually refreshing your access token from the dev console, run the Oauth2 example in the examples dir. Then copy and paste the refresh token returned to your tox file.

    5. `SPOTIFY_REDIRECT_URI = 'http://localhost:5000/callback/spotify'`

        Note: You have to register this callback in your Application's dashboard https://developer.spotify.com/dashboard/applications

    6. `PYFY_TEST_INTEGRATION_SYNC` = `true`
    7. `PYFY_TEST_INTEGRATION_ASYNC = true`
2. Run:

    Note:

    - This will run some tests using your client ID, client secret and access token.
    - Unfortunately Spotify does not have a sandbox API, so we have to test it against the live API

- Tests will carefully teardown all resources created and/or modified
- Integration tests will not be abusive to the API and should only test for successful integration with minimum API calls
- OAuth2 flow isn't tested in the tests folder (yet). Instead you can manually test it in the examples folder by running: `pip install flask pyfy && python examples/oauth2.py`

```
$ tox
```

CHAPTER

TWELVE

INDICES AND TABLES

- genindex
- modindex
- *Search*

PYTHON MODULE INDEX

p

pyfy.creds, 73
pyfy.excs, 71

INDEX

A

access_is_expired (*pyfy.creds.ClientCreds property*), 73
access_is_expired (*pyfy.creds.UserCreds property*), 75
add_playlist_tracks() (*pyfy.sync_client.Spotify method*), 24, 48
album_tracks() (*pyfy.sync_client.Spotify method*), 24, 48
albums() (*pyfy.sync_client.Spotify method*), 24, 48
ALL_SCOPES (*in module pyfy.creds*), 73
ApiError, 71
args (*pyfy.excs.ApiError attribute*), 71
args (*pyfy.excs.AuthError attribute*), 71
args (*pyfy.excs.SpotifyError attribute*), 72
artist_albums() (*pyfy.sync_client.Spotify method*), 25, 49
artist_related_artists() (*pyfy.sync_client.Spotify method*), 25, 49
artist_top_tracks() (*pyfy.sync_client.Spotify method*), 25, 49
artists() (*pyfy.sync_client.Spotify method*), 26, 50
auth_uri() (*pyfy.sync_client.Spotify method*), 26, 50
AuthError, 71
authorize_client_creds() (*pyfy.sync_client.Spotify method*), 26, 50
available_genre_seeds() (*pyfy.sync_client.Spotify method*), 26, 50

B

birthdate (*pyfy.creds.UserCreds attribute*), 74
build_user_creds() (*pyfy.sync_client.Spotify method*), 26, 50

C

categories() (*pyfy.sync_client.Spotify method*), 27, 51
category() (*pyfy.sync_client.Spotify method*), 27, 51
category_playlist() (*pyfy.sync_client.Spotify method*), 27, 51
ClientCreds (*class in pyfy.creds*), 73
code (*pyfy.excs.ApiError attribute*), 71
code (*pyfy.excs.AuthError attribute*), 71
country (*pyfy.creds.UserCreds attribute*), 74
create_playlist() (*pyfy.sync_client.Spotify method*), 28, 52
currently_playing() (*pyfy.sync_client.Spotify method*), 28, 52
currently_playing_info() (*pyfy.sync_client.Spotify method*), 28, 52

D

delete_albums() (*pyfy.sync_client.Spotify method*), 29, 53
delete_playlist() (*pyfy.sync_client.Spotify method*), 29, 53
delete_playlist_tracks() (*pyfy.sync_client.Spotify method*), 29, 53
delete_tracks() (*pyfy.sync_client.Spotify method*), 30, 54
devices() (*pyfy.sync_client.Spotify method*), 30, 54
display_name (*pyfy.creds.UserCreds attribute*), 74

E

email (*pyfy.creds.UserCreds attribute*), 74
external_urls (*pyfy.creds.UserCreds attribute*), 75

F

featured_playlists() (*pyfy.sync_client.Spotify method*), 30, 54
follow_artists() (*pyfy.sync_client.Spotify method*), 30, 54
follow_playlist() (*pyfy.sync_client.Spotify method*), 31, 55
follow_users() (*pyfy.sync_client.Spotify method*), 31, 55
followed_artists() (*pyfy.sync_client.Spotify method*), 31, 55
followers (*pyfy.creds.UserCreds attribute*), 75
follows_artists() (*pyfy.sync_client.Spotify method*), 31, 55
follows_playlist() (*pyfy.sync_client.Spotify method*), 31, 55
follows_users() (*pyfy.sync_client.Spotify method*), 32, 56

G

get() (*pyfy.creds.ClientCreds method*), 73
get() (*pyfy.creds.UserCreds method*), 75

H

href (*pyfy.creds.UserCreds attribute*), 75
http_request (*pyfy.excs.ApiError attribute*), 71
http_request (*pyfy.excs.AuthError attribute*), 71
http_response (*pyfy.excs.ApiError attribute*), 71
http_response (*pyfy.excs.AuthError attribute*), 71

I

id (*pyfy.creds.UserCreds attribute*), 75
images (*pyfy.creds.UserCreds attribute*), 75
is_active (*pyfy.sync_client.Spotify property*), 32, 56
IS_ASYNC (*pyfy.sync_client.Spotify attribute*), 24, 48
is_oauth_ready (*pyfy.creds.ClientCreds property*), 73
is_oauth_ready (*pyfy.sync_client.Spotify property*), 32, 56
is_premium (*pyfy.sync_client.Spotify property*), 32, 56

L

load_from_env() (*pyfy.creds.ClientCreds method*), 73
load_from_env() (*pyfy.creds.UserCreds method*), 75
load_from_json() (*pyfy.creds.ClientCreds method*), 73
load_from_json() (*pyfy.creds.UserCreds method*), 75

M

me() (*pyfy.sync_client.Spotify method*), 32, 56
module
    pyfy.creds, 73
    pyfy.excs, 71
msg (*pyfy.excs.ApiError attribute*), 71
msg (*pyfy.excs.AuthError attribute*), 71

N

new_releases() (*pyfy.sync_client.Spotify method*), 32, 56
next() (*pyfy.sync_client.Spotify method*), 33, 57
next_page() (*pyfy.sync_client.Spotify method*), 33, 57

O

owns_albums() (*pyfy.sync_client.Spotify method*), 33, 57
owns_tracks() (*pyfy.sync_client.Spotify method*), 33, 57

P

pause() (*pyfy.sync_client.Spotify method*), 34, 58
pickle() (*pyfy.creds.ClientCreds method*), 73
pickle() (*pyfy.creds.UserCreds method*), 75
play() (*pyfy.sync_client.Spotify method*), 34, 58
playback_transfer() (*pyfy.sync_client.Spotify method*), 34, 58
playlist() (*pyfy.sync_client.Spotify method*), 35, 59
playlist_cover() (*pyfy.sync_client.Spotify method*), 35, 59
playlist_tracks() (*pyfy.sync_client.Spotify method*), 35, 59
populate_user_creds() (*pyfy.sync_client.Spotify method*), 36, 60
previous() (*pyfy.sync_client.Spotify method*), 36, 60
previous_page() (*pyfy.sync_client.Spotify method*), 36, 60
product (*pyfy.creds.UserCreds attribute*), 75
pyfy.creds
    module, 73
pyfy.excs
    module, 71

Q

queue() (*pyfy.sync_client.Spotify method*), 36, 60

R

recently_played_tracks() (*pyfy.sync_client.Spotify method*), 36, 60
recommendations() (*pyfy.sync_client.Spotify method*), 37, 61
reorder_playlist_track() (*pyfy.sync_client.Spotify method*), 40, 64
repeat() (*pyfy.sync_client.Spotify method*), 40, 64
replace_playlist_tracks() (*pyfy.sync_client.Spotify method*), 40, 64

S

save_albums() (*pyfy.sync_client.Spotify method*), 41, 65
save_as_json() (*pyfy.creds.ClientCreds method*), 74
save_as_json() (*pyfy.creds.UserCreds method*), 76
save_tracks() (*pyfy.sync_client.Spotify method*), 41, 65
search() (*pyfy.sync_client.Spotify method*), 41, 65
seek() (*pyfy.sync_client.Spotify method*), 42, 66
shuffle() (*pyfy.sync_client.Spotify method*), 42, 66
Spotify (*class in pyfy.sync_client*), 23, 47
SpotifyError, 72

T

track_audio_analysis() (*pyfy.sync_client.Spotify method*), 42, 66
tracks() (*pyfy.sync_client.Spotify method*), 42, 66
tracks_audio_features() (*pyfy.sync_client.Spotify method*), 43, 67
type (*pyfy.creds.UserCreds attribute*), 75

U

unfollow_artists() (*pyfy.sync_client.Spotify method*), 43, 67
unfollow_playlist() (*pyfy.sync_client.Spotify method*), 43, 67
unfollow_users() (*pyfy.sync_client.Spotify method*), 43, 67
unpickle() (*pyfy.creds.ClientCreds class method*), 74
unpickle() (*pyfy.creds.UserCreds class method*), 76
update_playlist() (*pyfy.sync_client.Spotify method*), 43, 67
uri (*pyfy.creds.UserCreds attribute*), 75
user_albums() (*pyfy.sync_client.Spotify method*), 44, 68
user_creds (*pyfy.sync_client.Spotify property*), 44, 68
user_playlists() (*pyfy.sync_client.Spotify method*), 44, 68
user_profile() (*pyfy.sync_client.Spotify method*), 44, 68
user_top_artists() (*pyfy.sync_client.Spotify method*), 45, 69
user_top_tracks() (*pyfy.sync_client.Spotify method*), 45, 69
user_tracks() (*pyfy.sync_client.Spotify method*), 45, 69
UserCreds (*class in pyfy.creds*), 74

V

volume() (*pyfy.sync_client.Spotify method*), 46, 70

W

with_traceback() (*pyfy.excs.ApiError method*), 71
with_traceback() (*pyfy.excs.AuthError method*), 71
with_traceback() (*pyfy.excs.SpotifyError method*), 72