
python-lichess Documentation

python-lichess

Oct 25, 2020

Contents

1	API Methods	1
2	Formats	5
3	Authentication	7
4	Custom PGNs	9
5	API Client Configuration	11
6	Introduction	13
7	Installing	15
	Python Module Index	17
	Index	19

The module `lichess.api` provides thin wrappers around the lichess API.

In addition to the API parameters, each function takes optional `format`, `auth`, and `client` arguments.

Endpoints that return collections (like `user_games`) stream the results by returning a generator.

`lichess.api.user` (`username`, `**kwargs`)
Wrapper for the `GET /api/user/<username>` endpoint.

```
>>> user = lichess.api.user('thibault')
>>> print(user.get('perfs', {}).get('blitz', {}).get('rating'))
1617
```

`lichess.api.users_by_team` (`team`, `**kwargs`)
Wrapper for the `GET /api/team/{name}/users` endpoint. Returns a generator that streams the user data.

```
>>> users = lichess.api.users_by_team('coders')
>>> ratings = [u.get('perfs', {}).get('blitz', {}).get('rating') for u in users]
>>> print(ratings)
[1349, 1609, ...]
```

`lichess.api.users_by_ids` (`ids`, `**kwargs`)
Wrapper for the `POST /api/users` endpoint. Returns a generator that splits the IDs into multiple requests as needed.

Note: Use `users_status` when possible, since it is cheaper and not rate-limited.

```
>>> users = lichess.api.users_by_ids(['thibault', 'cyanfish'])
>>> ratings = [u.get('perfs', {}).get('blitz', {}).get('rating') for u in users]
>>> print(ratings)
[1617, 1948]
```

`lichess.api.users_status` (`ids`, `**kwargs`)
Wrapper for the `GET /api/users/status` endpoint. Returns a generator that makes requests for additional pages as needed.

Note: This endpoint is cheap and not rate-limited. Use it instead of `users_by_ids` when possible.

```
>>> users = lichess.api.users_status(['thibault', 'cyanfish'])
>>> online_count = len([u for u in users if u.get('online')])
>>> print(online_count)
1
```

`lichess.api.user_activity` (*username*, ***kwargs*)
Wrapper for the `GET /api/user/<username>/activity` endpoint.

`lichess.api.game` (*game_id*, ***kwargs*)
Wrapper for the `GET /api/game/{id}` endpoint.

By default, returns a dict representing a JSON game object. Use `format=PGN` for a PGN string or `format=PYCHESS` for a `python-chess` game object.

```
>>> game = lichess.api.game('Qa7FJNk2')
>>> print(game['moves'])
e4 e5 Nf3 Nc6 Bc4 Qf6 d3 h6 ...
```

```
>>> from lichess.format import PGN, PYCHESS
>>> pgn = lichess.api.game('Qa7FJNk2', format=PGN)
>>> print(pgn)
[Event "Casual rapid game"]
...
```

```
>>> game_obj = lichess.api.game('Qa7FJNk2', format=PYCHESS)
>>> print(game_obj.end().board())
. . k . R b r .
. p p r . N p .
p . . . . . p
. . . . .
. . . p . . . .
P . . P . . . P
. P P . . P P .
. . K R . . . .
```

`lichess.api.games_by_ids` (*ids*, ***kwargs*)
Wrapper for the `POST /games/export/_ids` endpoint. Returns a generator that splits the IDs into multiple requests as needed.

`lichess.api.user_games` (*username*, ***kwargs*)
Wrapper for the `GET /api/user/<username>/games` endpoint.

By default, returns a generator that streams game objects. Use `format=PGN` for a generator of game PGNs, `format=SINGLE_PGN` for a single PGN string, or `format=PYCHESS` for a generator of `python-chess` game objects.

```
>>> games = lichess.api.user_games('cyanfish', max=50, perfType='blitz')
>>> print(next(games)['moves'])
e4 e5 Nf3 Nc6 Bc4 Qf6 d3 h6 ...
```

```
>>> from lichess.format import PGN, SINGLE_PGN, PYCHESS
>>> pgns = lichess.api.user_games('cyanfish', max=50, format=PGN)
>>> print(next(pgns))
[Event "Casual rapid game"]
...
```

```
>>> pgn = lichess.api.user_games('cyanfish', max=50, format=SINGLE_PGN)
>>> print(pgn)
[Event "Casual rapid game"]
...
```

```
>>> game_objs = lichess.api.user_games('cyanfish', max=50, format=PYCHESS)
>>> print(next(game_objs).end().board())
. . k . R b r .
. p p r . N p .
P . . . . . P
. . . . .
. . . p . . . .
P . . P . . . P
. P P . . P P .
. . K R . . . .
```

`lichess.api.tournaments` (***kwargs*)

Wrapper for the `GET /api/tournament` endpoint.

`lichess.api.tournament` (*tournament_id*, ***kwargs*)

Wrapper for the `GET /api/tournament/<tournamentId>` endpoint.

`lichess.api.tournament_standings` (*tournament_id*, ***kwargs*)

Wrapper for the `GET /api/tournament/<tournamentId>` endpoint. Returns a generator that makes requests for additional pages as needed.

`lichess.api.tv_channels` (***kwargs*)

Wrapper for the `GET /tv/channels` endpoint.

The module `lichess.format` lets you choose the format for games and other data (`JSON`, `PGN`, `SINGLE_PGN`, or `PYCHESS`).

`lichess.format.PGN = <lichess.format._Pgn object>`
Produces a PGN string, or a generator for PGN strings of each game.

```
>>> from lichess.format import PGN
>>>
>>> pgn = lichess.api.game('Qa7FJNk2', format=PGN)
>>> print(pgn)
[Event "Casual rapid game"]
...
>>> pgn = lichess.api.user_games('cyanfish', max=50, format=PGN)
>>> print(len(list(pgn)))
50
```

`lichess.format.SINGLE_PGN = <lichess.format._SinglePgn object>`
Produces a PGN string, possibly containing multiple games.

```
>>> from lichess.format import SINGLE_PGN
>>>
>>> pgn = lichess.api.user_games('cyanfish', max=50, format=SINGLE_PGN)
>>> print(pgn)
[Event "Casual rapid game"]
...
```

`lichess.format.PYCHESS = <lichess.format._PyChess object>`
Produces a `python-chess` game object, or a generator for multiple game objects.

```
>>> from lichess.format import PYCHESS
>>>
>>> game = lichess.api.game('Qa7FJNk2', format=PYCHESS)
>>> print(game.end().board())
. . k . R b r .
```

(continues on next page)

(continued from previous page)

```
. p p r . N p .  
p . . . . . P  
. . . . .  
. . . p . . . .  
P . . P . . . P  
. P P . . P P .  
. . K R . . . .
```

`lichess.format.JSON = <lichess.format._Json object>`

Produces a dict representing a JSON object, or a generator for multiple dicts. This is the default format.

```
>>> from lichess.format import JSON  
>>>  
>>> game = lichess.api.game('Qa7FJnk2', format=JSON) # or leave out  
>>> print(game['players']['white']['user']['id'])  
cyanfish
```

CHAPTER 3

Authentication

Authentication lets you download games at a faster rate and access private data.

The simplest way to authenticate is to create an OAuth access token on lichess.org, and use the `auth` parameter like so:

```
>>> import lichess.api
>>>
>>> games = lichess.api.user_games('cyanfish', max=100, auth='your-token-here')
```

You can also authenticate using your username and password (but please, try and avoid this):

```
>>> import lichess.api
>>>
>>> cookie = lichess.api.login('cyanfish', 'this-is-not-my-real-password')
>>> games = lichess.api.user_games('cyanfish', max=100, auth=cookie)
```


The module `lichess.pgn` provides functions to generate custom PGNs from games in JSON format.

If you only need a default PGN, see the `lichess.format` module for an easier way to get it.

`lichess.pgn.from_game(game, headers=None)`

Converts a JSON game to a PGN string.

Game The game object.

Headers An optional dictionary with custom PGN headers.

```
>>> game = lichess.api.game('Qa7FJNk2', with_moves=1)
>>> pgn = lichess.pgn.from_game(game)
>>> print(pgn)
[Event "Casual rapid game"]
...
```

`lichess.pgn.io_from_game(game, headers=None)`

Like `from_game`, except it wraps the result in `StringIO`.

This allows easy integration with the `python-chess` library. But if this is all you need, see the `lichess.format` module for an easier way.

Game The game object.

Headers An optional dictionary with custom PGN headers.

```
>>> import lichess.api
>>> import lichess.pgn
>>> import chess.pgn
>>>
>>> api_game = lichess.api.game('Qa7FJNk2', with_moves=1)
>>> game = chess.pgn.read_game(lichess.pgn.io_from_game(api_game))
>>> print(game.end().board())
. . k . R b r .
. p p r . N p .
```

(continues on next page)

(continued from previous page)

```
p . . . . . p
. . . . .
. . . p . . . .
P . . P . . . P
. P P . . P P .
. . K R . . . .
```

`lichess.pgn.from_games` (*games*, *headers=None*)

Converts an enumerable of JSON games to a PGN string.

Games The enumerable of game objects.

Headers An optional dictionary with (shared) custom PGN headers.

```
>>> import itertools
>>>
>>> games = lichess.api.user_games('cyanfish', with_moves=1)
>>> pgn = lichess.pgn.from_games(itertools.islice(games, 5))
>>> print(pgn.count('\n'))
66
```

`lichess.pgn.save_games` (*games*, *path*, *headers=None*)

Saves an enumerable of JSON games to a PGN file.

Games The enumerable of game objects.

Path The path of the .pgn file to save.

Headers An optional dictionary with (shared) custom PGN headers.

```
>>> import itertools
>>>
>>> games = lichess.api.user_games('cyanfish', with_moves=1)
>>> lichess.pgn.save_games(itertools.islice(games, 5), 'mylast5games.pgn')
```

API Client Configuration

The *DefaultApiClient* is used to perform the actual HTTP requests. It also manages rate-limiting and retries.

If you need more functionality, you can subclass it. To use a custom client, set *default_client* or use the *client* parameter in each API method wrapper.

exception `lichess.api.ApiError`

The base class for API exceptions.

exception `lichess.api.ApiHttpError` (*http_status*, *url*, *response_text*)

The class for API exceptions caused by an HTTP error code.

class `lichess.api.DefaultApiClient` (*base_url=None*, *max_retries=None*)

The default API client, with immediate HTTP calls and basic rate-limiting functionality.

base_url = `'https://lichess.org/'`

The base lichess API URL.

This does not include the `/api/` prefix, since some APIs don't use it.

max_retries = `-1`

The maximum number of retries after rate-limiting before an exception is raised. -1 for infinite retries.

call (*path*, *params=None*, *post_data=None*, *auth=None*, *format=<lichess.format._Json object>*, *object_type='public_api'*)

Makes an API call, prepending *base_url* to the provided path. HTTP GET is used unless *post_data* is provided.

Consecutive calls use a 1s delay. If HTTP 429 is received, retries after a 1min delay.

on_rate_limit (*url*, *retry_count*)

A handler called when HTTP 429 is received.

Raises an exception when *max_retries* is exceeded.

`lichess.api.default_client` = `<lichess.api.DefaultApiClient object>`

The client object used to communicate with the lichess API.

Initially set to an instance of *DefaultApiClient*.

This is a client library for the lichess.org API. It is designed to be:

- Easy to use
- Customizable when you need it
- Adaptable to API changes
- Easy to integrate with [python-chess](#)

Have a look at some short examples. For more, check out the full doc in the table of contents.

Getting a user's rating:

```
>>> import lichess.api
>>>
>>> user = lichess.api.user('thibault')
>>> print(user['perfs']['blitz']['rating'])
1617
```

Checking who's online and playing:

```
>>> import lichess.api
>>>
>>> users = list(lichess.api.users_status(['thibault', 'cyanfish']))
>>> online = [u['id'] for u in users if u.get('online')]
>>> playing = [u['id'] for u in users if u.get('playing')]
>>> print(online, playing)
['thibault', 'cyanfish'] ['cyanfish']
```

Saving a PGN of a user's last 200 games:

```
>>> import lichess.api
>>> from lichess.format import SINGLE_PGN
>>>
>>> pgn = lichess.api.user_games('thibault', max=200, format=SINGLE_PGN)
```

(continues on next page)

(continued from previous page)

```
>>> with open('last200.pgn', 'w') as f:  
>>>     f.write(pgn)
```

Integrating with python-chess:

```
>>> import lichess.api  
>>> from lichess.format import PYCHESS  
>>>  
>>> game = lichess.api.game('Qa7FJNk2', format=PYCHESS)  
>>> print(game.end().board())  
. . k . R b r .  
. p p r . N p .  
p . . . . . p  
. . . . .  
. . . p . . . .  
P . . P . . . P  
. P P . . P P .  
. . K R . . . .
```

CHAPTER 7

Installing

```
pip install python-lichess
```

Python Module Index

|

`lichess.api`, 11
`lichess.auth`, 7
`lichess.format`, 5
`lichess.pgn`, 9

A

ApiError, 11
ApiHttpError, 11

B

base_url (*lichess.api.DefaultApiClient* attribute), 11

C

call() (*lichess.api.DefaultApiClient* method), 11

D

default_client (*in module lichess.api*), 11
DefaultApiClient (*class in lichess.api*), 11

F

from_game() (*in module lichess.pgn*), 9
from_games() (*in module lichess.pgn*), 10

G

game() (*in module lichess.api*), 2
games_by_ids() (*in module lichess.api*), 2

I

io_from_game() (*in module lichess.pgn*), 9

J

JSON (*in module lichess.format*), 6

L

lichess.api (*module*), 1, 11
lichess.auth (*module*), 7
lichess.format (*module*), 5
lichess.pgn (*module*), 9

M

max_retries (*lichess.api.DefaultApiClient* attribute),
11

O

on_rate_limit() (*lichess.api.DefaultApiClient*
method), 11

P

PGN (*in module lichess.format*), 5
PYCHESS (*in module lichess.format*), 5

S

save_games() (*in module lichess.pgn*), 10
SINGLE_PGN (*in module lichess.format*), 5

T

tournament() (*in module lichess.api*), 3
tournament_standings() (*in module lichess.api*),
3
tournaments() (*in module lichess.api*), 3
tv_channels() (*in module lichess.api*), 3

U

user() (*in module lichess.api*), 1
user_activity() (*in module lichess.api*), 2
user_games() (*in module lichess.api*), 2
users_by_ids() (*in module lichess.api*), 1
users_by_team() (*in module lichess.api*), 1
users_status() (*in module lichess.api*), 1