
InstaLiveCLI

Release v1.0.0

Jun 17, 2021

Contents:

1	InstaLiveCLI	1
1.1	InstaLiveCLI package	1
2	Indices and tables	7
	Python Module Index	9
	Index	11

1.1 InstaLiveCLI package

1.1.1 Submodules

1.1.2 InstaLiveCLI.InstLiveCLI module

```
class InstaLiveCLI.InstLiveCLI.InstLiveCLI (username="", password="", settings="",  
                                             auth="", saved=False)  
    Bases: object  
  
    API_URL = 'https://i.instagram.com/api/v1/'  
    DEVICE_SETS = {'android_release': '9.0', 'android_version': '28', 'app_version': '1'  
    IG_SIG_KEY = '4f8732eb9ba7d1c8e8897a75d6474d4eb3f5279137431b2aafb71fafe2abe178'  
    LastJson = None  
    LastResponse = None  
    SIG_KEY_VERSION = '4'  
    USER_AGENT = 'Instagram 136.0.0.34.124 Android (28/9.0; 640dpi; 1440x2560; samsung; st'  
    add_post_live_to_igtv (description, title)  
    broadcastMessage = ''  
    broadcast_id = None  
    cookie_jar = None  
    create_broadcast ()  
        Creating broadcast live  
        broadcast_id, stream_server, stream_key will be saved on Class Variable.
```

Returns if broadcast created

Return type bool

delete_post_live ()

Deleting video after broadcast is stopped

Returns If video successfully deleted

Return type bool

end_broadcast ()

Stopping current broadcast from API

Returns if broadcast is stopped

Return type bool

export_settings (filename)

Exporting all the settings to json file

Parameters **filename** (str) – filename, for example: “settings.json”

generate_UUID (t: bool = True, seed=None)

generating UUID

Parameters

- **t** (bool, optional) – if you want return raw uuid. Defaults to True.
- **seed** (str, optional) – seed string. Defaults to None.

Returns UUID1 string

Return type str

generate_device_id (seed)

generate unique device id from seed

Parameters **seed** (str) – seed string

Returns unique formatted device id

Return type str

generate_signature (data, skip_quote=False)

get_broadcast_status ()

Returning current broadcast info status if exists

Returns broadcast status

Return type str

get_code_challenge_required (path, choice=0)

send code challenge choice choices: 0 - SMS, 1 - EMAIL :param path: api path :type path: str :param choice: choice challenge . Defaults to 0. :type choice: int, optional

get_comments ()

Getting all comments on live broadcast

Returns list of comments

Return type dict

get_post_live_thumbnails ()

```

get_viewer_list()
    Get all of viewers from broadcast

    Returns Returns two list user, and ids

    Return type list

import_settings(filename)
    Import and load settings from json file

    Parameters filename (str) – filename, for example: “settings.json”

    Returns settings configuration

    Return type dict

isLoggedIn = False

is_cli = False

is_running = False

last_comment_ts = 1

last_csrf_token = None

live_info()
    Printing broadcast information to the console

    Broadcast ID, Server URL, Stream Key, Viewer Count, Status Broadcast.

load_cookies(cookie_string)
    Loads cookie from Cookiestring to Cookie jar and then import it to session’s cookiejar

    Parameters cookie_string (str) – cookies string

load_settings(cached_auth)
    load all the settings from python dictionary

    Parameters cached_auth (dict) – settings configuration

login(force=False)
    Login to api

    becareful if you want to use force, it might rate limit, even worst banned you if you send too many times.

    Parameters force (bool, optional) – forcing to send login requests. Defaults to False.

    Returns if logged in

    Return type bool

mute_comments()
    Mute current broadcast comments

    Returns if current comment muted

    Return type bool

password = None

previewHeight = 1920

previewWidth = 1080

rank_token = None

```

```

s = <requests.sessions.Session object>

save_settings = False

sendNotification = True

send_comment (msg)
    Sending comment to broadcast

    Parameters msg (str) – message to send

    Returns if comment sent

    Return type bool

send_request (endpoint, post=None, login=False, headers: dict = {})
    Sending requests to instagram API

    Parameters

    • endpoint (str) – API endpoint instagram

    • post (dict, optional) – data for post to api. Defaults to None.

    • login (bool, optional) – [description]. Defaults to False.

    Raises Exception – Failing send requests

    Returns if requests is success

    Return type bool

set_code_challenge_required (path, code)
    setting code challenge

    Parameters

    • path (str) – path api

    • code (str) – code challenge

set_proxy (proxy=None)
    Set proxy to session

    Parameters proxy (string, optional) – user:password@ip:port. Defaults to None.

set_user (username, password)
    Set user to class variable instance

    Parameters

    • username (str) – Username of your Instagram

    • password (str) – Password of your Instagram

settings
    Helper property that extracts the settings that you should cache in addition to username and password.

    Returns all the settings and cookies

    Return type dict

share_to_story = False

start ()
    Starting CLI APP

start_broadcast ()
    Starting current broadcast
    
```


Returns If broadcast is started

Return type bool

stop()

stream_key = None

stream_server = None

token = None

two_factor(*code*=")

sending verification if there's two factor needed

Returns if two factor passed

Return type bool

two_factor_identifier = None

two_factor_last_number = None

two_factor_required = False

unmute_comment()

Unmute current broadcast comments

Returns if current comment unmuted

Return type bool

upload_live_thumbnails()

username = None

username_id = None

uuid = None

wave(*user_id*)

Waving to specific user_id

Parameters **user_id**(*str*) – id of user you want to wave

Returns if user waved

Return type bool

1.1.3 InstaLiveCLI.http module

class InstaLiveCLI.http.**ClientCookieJar**(*cookie_string=None, policy=None*)

Bases: http.cookiejar.CookieJar

Custom CookieJar that can be pickled to/from strings

auth_expires

dump(*force_cookies=None*)

pickle cookie_jar to strings

Parameters **force_cookies**(*cookie_jar, optional*) – cookie_jar session. Defaults to None.

Returns pickled cookie_jar strings

Return type str

expires_earliest

For backward compatibility

1.1.4 InstaLiveCLI.util module

`InstaLiveCLI.util.from_json(json_object)`

Converting and encode bytes value from json object

Parameters `json_object` (*dict*) – json dictionary

Returns `json_object`

Return type `dict`

`InstaLiveCLI.util.to_json(python_object)`

Converting bytes value on object to json

Parameters `python_object` (*dict*) – A dict that contain bytes types on value

Raises `TypeError` – If json is not serializable

Returns Returns decoded bytes value

Return type `dict`

1.1.5 Module contents

CHAPTER 2

Indices and tables

- `genindex`
- `modindex`
- `search`

i

- InstaLiveCLI, 6
- InstaLiveCLI.http, 5
- InstaLiveCLI.InstaliveCLI, 1
- InstaLiveCLI.util, 6

A

add_post_live_to_igtv() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
1
API_URL (InstaLiveCLI.InstaLiveCLI.InstaLiveCLI attribute), 1
auth_expires (InstaLiveCLI.http.ClientCookieJar attribute), 5

B

broadcast_id (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI attribute),
1
broadcastMessage (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI attribute),
1

C

ClientCookieJar (class in InstaLiveCLI.http), 5
cookie_jar (InstaLiveCLI.InstaLiveCLI.InstaLiveCLI attribute), 1
create_broadcast() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
1

D

delete_post_live() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2
DEVICE_SETS (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI attribute),
1
dump() (InstaLiveCLI.http.ClientCookieJar method), 5

E

end_broadcast() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2

expires_earliest (InstaLive-
CLI.http.ClientCookieJar attribute), 5
export_settings() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2

F

from_json() (in module InstaLiveCLI.util), 6

G

generate_device_id() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2
generate_signature() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2
generate_UUID() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2
get_broadcast_status() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2
get_code_challenge_required() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2
get_comments() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2
get_post_live_thumbnails() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2
get_viewer_list() (InstaLive-
CLI.InstaLiveCLI.InstaLiveCLI method),
2

I

IG_SIG_KEY (InstaLiveCLI.InstaLiveCLI.InstaLiveCLI attribute), 1

import_settings() (InstaLiveCLI.InstLiveCLI method), 3
 InstaLiveCLI (class in InstaLiveCLI.InstLiveCLI), 1
 InstaLiveCLI (module), 6
 InstaLiveCLI.http (module), 5
 InstaLiveCLI.InstLiveCLI (module), 1
 InstaLiveCLI.util (module), 6
 is_cli (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 3
 is_running (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 3
 isLoggedInIn (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 3

L

last_comment_ts (InstaLiveCLI.InstLiveCLI attribute), 3
 last_csrf_token (InstaLiveCLI.InstLiveCLI attribute), 3
 LastJson (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 1
 LastResponse (InstaLiveCLI.InstLiveCLI attribute), 1
 live_info() (InstaLiveCLI.InstLiveCLI method), 3
 load_cookies() (InstaLiveCLI.InstLiveCLI method), 3
 load_settings() (InstaLiveCLI.InstLiveCLI method), 3
 login() (InstaLiveCLI.InstLiveCLI.InstLiveCLI method), 3

M

mute_comments() (InstaLiveCLI.InstLiveCLI method), 3

P

password (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 3
 previewHeight (InstaLiveCLI.InstLiveCLI attribute), 3
 previewWidth (InstaLiveCLI.InstLiveCLI attribute), 3

R

rank_token (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 3

S

s (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 3
 save_settings (InstaLiveCLI.InstLiveCLI attribute), 4
 send_comment() (InstaLiveCLI.InstLiveCLI method), 4
 send_request() (InstaLiveCLI.InstLiveCLI method), 4
 sendNotification (InstaLiveCLI.InstLiveCLI attribute), 4
 set_code_challenge_required() (InstaLiveCLI.InstLiveCLI.InstLiveCLI method), 4
 set_proxy() (InstaLiveCLI.InstLiveCLI method), 4
 set_user() (InstaLiveCLI.InstLiveCLI.InstLiveCLI method), 4
 settings (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 4
 share_to_story (InstaLiveCLI.InstLiveCLI attribute), 4
 SIG_KEY_VERSION (InstaLiveCLI.InstLiveCLI attribute), 1
 start() (InstaLiveCLI.InstLiveCLI.InstLiveCLI method), 4
 start_broadcast() (InstaLiveCLI.InstLiveCLI method), 4
 stop() (InstaLiveCLI.InstLiveCLI.InstLiveCLI method), 5
 stream_key (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 5
 stream_server (InstaLiveCLI.InstLiveCLI attribute), 5
 T
 to_json() (in module InstaLiveCLI.util), 6
 token (InstaLiveCLI.InstLiveCLI.InstLiveCLI attribute), 5
 two_factor() (InstaLiveCLI.InstLiveCLI method), 5


```
two_factor_identifier      (InstaLive-
    CLI.InstaLiveCLI.InstaLiveCLI  attribute),
5
two_factor_last_number    (InstaLive-
    CLI.InstaLiveCLI.InstaLiveCLI  attribute),
5
two_factor_required       (InstaLive-
    CLI.InstaLiveCLI.InstaLiveCLI  attribute),
5
```

U

```
unmute_comment()          (InstaLive-
    CLI.InstaLiveCLI.InstaLiveCLI  method),
5
upload_live_thumbnails()  (InstaLive-
    CLI.InstaLiveCLI.InstaLiveCLI  method),
5
USER_AGENT (InstaLiveCLI.InstaLiveCLI.InstaLiveCLI
    attribute), 1
username (InstaLiveCLI.InstaLiveCLI.InstaLiveCLI at-
    tribute), 5
username_id               (InstaLive-
    CLI.InstaLiveCLI.InstaLiveCLI  attribute),
5
uuid (InstaLiveCLI.InstaLiveCLI.InstaLiveCLI at-
    tribute), 5
```

W

```
wave() (InstaLiveCLI.InstaLiveCLI.InstaLiveCLI
    method), 5
```