
instagram_private_api Documentation

Release 0.1.4

felixbreuer

Feb 04, 2020

1	Features	3
1.1	Installation	3
1.2	Usage	4
1.3	FAQ	4
1.4	Developer Interface	6
	Python Module Index	37
	Index	39

A Python wrapper for the Instagram private API with no 3rd party dependencies. Supports both the app and web APIs.

- Supports many functions that are only available through the official app
- Compatibility patch available to match the public API methods
- Beta Python 3 support.

1.1 Installation

1.1.1 Pip

Install via pip

```
$ pip install instapi
```

Update your install with the latest release

```
$ pip install instapi --upgrade
```

Force an update from source

```
$ pip install instapi --upgrade --force-reinstall
```

1.1.2 Source Code

The library is maintained on GitHub. Feel free to clone the repository.

```
git clone git://github.com/breuerfelix/instapi.git
```

1.2 Usage

The private app API client emulates the official app and has a larger number of functions.

1.2.1 App API

```
from instapi import Client, ClientCompatPatch

user_name = 'YOUR_LOGIN_USER_NAME'
password = 'YOUR_PASSWORD'

api = Client(user_name, password)
results = api.feed_timeline()
items = results.get('items', [])
for item in items:
    # Manually patch the entity to match the public api as closely as possible,
    ↪ optional
    # To automatically patch entities, initialise the Client with auto_patch=True
    ClientCompatPatch.media(item)
    print(media['code'])
```

1.2.2 Avoiding Re-login

You are advised to persist/cache the auth cookie details to avoid logging in every time you make an api call. Excessive logins is a surefire way to get your account flagged for removal. It's also advisable to cache the client details such as user agent, etc together with the auth details.

The saved auth cookie can be reused for up to 90 days.

An example of how to save and reuse the auth setting can be found in the [examples](#).

1.3 FAQ

- *Can I create accounts with this library?*
- *Can I _____ with this library?*
- *What does error code XXX mean?*
- *“Your version of Instagram is out of date. Please upgrade your app to log in to Instagram.”*
- *How to direct message/share?*
- *What does sentry_block error mean?*
- *Why are the captions not posted?*
- *What does checkpoint_challenge_required, challenge_required mean?*

1.3.1 Can I create accounts with this library?

No. This library will not support account creation because of abuse by spammers. If you need an account, use the official app or website.

1.3.2 Can I _____ with this library?

This library is limited to what the mobile app interface can do. If you can't do it on those platforms, you can't do it through the library.

1.3.3 What does error code XXX mean?

- **400:** Bad request. Please check the parameters specified.
- **403:** The method requires authentication (web client) or the request has been denied by IG.
- **404:** The entity requested is not found (web client) or the endpoint does not exist.
- **429:** Too many requests. You're making too many calls.

IG may also return other 4XX or 5XX codes.

1.3.4 “Your version of Instagram is out of date. Please upgrade your app to log in to Instagram.”

Instagram is rejecting the app version that the lib is using.

If discarding the cached auth and relogging in does not work, you may need to:

1. update the lib, or
2. extract the latest signature key and version from the latest [Instagram APK](#) or from <https://github.com/mgp25/Instagram-API/blob/master/src/Constants.php>.

With the new sig key and app version, you can modify the client like so

```
new_app_version = '10.3.2'
new_sig_key = '5ad7d6f013666cc93c88fc8af940348bd067b68f0dce3c85122a923f4f74b251'
new_key_ver = '4' # does not freq change
new_ig_capa = '3ToAAA==' # does not freq change

api = Client(
    user_name, password,
    app_version=new_app_version,
    signature_key=new_sig_key,
    key_version= new_key_ver,
    ig_capabilities=new_ig_capa)
```

1.3.5 How to direct message/share?

There are no plans to implement direct messaging/sharing functions.

1.3.6 What does `sentry_block` error mean?

This is the response for detected spam/bot behavior. Stop using the api in whatever way that triggered this response.

1.3.7 Why are the captions not posted?

This is due to your account / access location (IP) being soft-blocked.

1.3.8 What does `checkpoint_challenge_required`, `challenge_required` mean?

Your access attempt has been flagged. Login manually to pass the required challenge.

1.4 Developer Interface

This page of the documentation will cover all methods and classes available to the developer.

The api currently has two main interfaces:

- **App API**

- `instapi.Client`
- `instapi.ClientCompatPatch`
- `instapi.ClientError`
- `instapi.ClientLoginError`
- `instapi.ClientLoginRequiredError`
- `instapi.ClientCookieExpiredError`
- `instapi.ClientThrottledError`
- `instapi.ClientReqHeadersTooLargeError`
- `instapi.ClientConnectionError`
- `instapi.ClientCheckpointRequiredError`
- `instapi.ClientChallengeRequiredError`
- `instapi.ClientSentryBlockError`
- `instapi.MediaRatios`
- `instapi.MediaTypes`

1.4.1 App API

class `instapi.Client` (*username*, *password*, ****kwargs**)

Main API client class for the private app api.

__init__ (*username*, *password*, ****kwargs**)

Parameters

- **username** – Login username

- **password** – Login password
- **kwargs** – See below

Keyword Arguments

- **auto_patch**: Patch the api objects to match the public API. Default: False
- **drop_incompat_key**: Remove api object keys that is not in the public API. Default: False
- **timeout**: Timeout interval in seconds. Default: 15
- **api_url**: Override the default api url base
- **cookie**: Saved cookie string from a previous session
- **settings**: A dict of settings from a previous session
- **on_login**: Callback after successful login
- **proxy**: Specify a proxy ex: 'http://127.0.0.1:8888' (ALPHA)
- **proxy_handler**: Specify your own proxy handler

Returns

authenticated_user_id

The current authenticated user id

authenticated_user_name

The current authenticated user name

autocomplete_user_list()

User list for autocomplete

block_friend_reel (*user_id*)

Hide your stories from a specific user

Parameters **user_id** – User id

Returns

```
{
    "status": "ok",
    "incoming_request": false,
    "is_blocking_reel": true,
    "followed_by": false,
    "is_muting_reel": false,
    "outgoing_request": false,
    "following": false,
    "blocking": true,
    "is_private": false
}
```

blocked_reels()

Get list of users from whom you've hid your stories

blocked_user_list()

Get list of blocked users

broadcast_comment (*broadcast_id*, *comment_text*)

Post a comment to a live broadcast

Parameters

- **broadcast_id** – Broadcast id

- **comment_text** – Comment text

Returns

broadcast_comments (*broadcast_id*, *last_comment_ts=0*)

Get a live broadcast's latest comments

Parameters

- **broadcast_id** – Broadcast id
- **last_comment_ts** –

Returns

broadcast_heartbeat_and_viewercount (*broadcast_id*)

Get a live broadcast's heartbeat and viewer count

Parameters **broadcast_id** – Broadcast id

Returns

broadcast_info (*broadcast_id*)

Get broadcast information. Known broadcast_status values: 'active', 'interrupted', 'stopped', 'hard_stop'

Parameters **broadcast_id** – Broadcast Id

Returns

```
{
  "status": "ok",
  "broadcast_status": "active",
  "media_id": "12345678934374208_123456789",
  "cover_frame_url": "https://scontent-hkg3-1.cdninstagram.com/
↪something.jpg",
  "broadcast_owner": {
    "username": "abc",
    "friendship_status": {
      "incoming_request": false,
      "followed_by": false,
      "outgoing_request": false,
      "following": false,
      "blocking": false,
      "is_private": false
    },
    "profile_pic_url": "http://scontent-hkg3-1.cdninstagram.com/
↪somethingelse.jpg",
    "profile_pic_id": "1234567850644676241_123456789",
    "full_name": "ABC",
    "pk": 123456789,
    "is_verified": true,
    "is_private": false
  },
  "dash_abr_playback_url": null,
  "broadcast_message": "",
  "published_time": 1485312576,
  "dash_playback_url": "https://scontent-hkg3-1.cdninstagram.com/
↪hvideo-ash1/v/dash-hd/spmething.mpd",
  "rtmp_playback_url": "rtmp://svelivestream007.16.ash1.facebook.
↪com:16000/live-hd/something",
  "id": 178591123456789,
```

(continues on next page)

(continued from previous page)

```

    "viewer_count": 9000.0
  }

```

broadcast_like (*broadcast_id*, *like_count=1*)

Like a live broadcast

Parameters

- **broadcast_id** – Broadcast id
- **like_count** –

Returns

broadcast_like_count (*broadcast_id*, *like_ts=0*)

Get a live broadcast's like count

Parameters **broadcast_id** – Broadcast id

Returns

bulk_delete_comments (*media_id*, *comment_ids*)

Bulk delete comment

Parameters

- **media_id** – Media id
- **comment_ids** – List of comment ids

Returns

```

{"status": "ok"}

```

bulk_translate (*comment_ids*)

Get translations of comments

Parameters **comment_ids** – list of comment/caption IDs

Returns

change_profile_picture (*photo_data*)

Change profile picture

Parameters **photo_data** – byte string of image

Returns

check_username (*username*)

Check username

Parameters **username** –

Returns

```

{
  "status": "ok",
  "available": false,
  "username": "xxx",
  "error_type": "username_is_taken",
  "error": "The username xxx is not available."
}

```

collection_feed (*collection_id*, ***kwargs*)

Get the items in a collection.

Parameters *collection_id* – Collection ID

Returns

comment_inline_replies (*media_id*, *comment_id*, *max_id*, ***kwargs*)

Get inline comment replies. Check for ‘next_max_child_cursor’ from `media_comments()` to determine if there are inline comment replies to retrieve.

Parameters

- **media_id** – Media id
- **comment_id** – Comment id
- **max_id** – The comment’s ‘next_max_child_cursor’ value from ‘media_comments()’

Returns

comment_like (*comment_id*)

Like a comment

Parameters *comment_id* –

Returns

```
{"status": "ok"}
```

comment_likers (*comment_id*)

Get users who have liked a comment

Parameters *comment_id* –

Returns

comment_replies (*media_id*, *comment_id*, ***kwargs*)

Get comment replies. Fixed at 20 replies returned per page. Check for ‘has_more_tail_child_comments’, ‘next_max_child_cursor’ to determine if there are more replies to page through.

Parameters

- **media_id** – Media id
- **comment_id** – Comment id
- **kwargs** – **max_id**: For pagination

Returns

comment_unlike (*comment_id*)

Unlike a comment

Parameters *comment_id* –

Returns

```
{"status": "ok"}
```

classmethod compatible_aspect_ratio (*size*)

Helper method to check aspect ratio for standard uploads

Parameters *size* – tuple of (width, height)

Returns True/False

configure (*upload_id*, *size*, *caption*=", *location*=None, *disable_comments*=False, *is_sidecar*=False)
Finalises a photo upload. This should not be called directly. Use `post_photo()` instead.

Parameters

- **upload_id** –
- **size** – tuple of (width, height)
- **caption** –
- **location** – a dict of venue/location information, from `location_search()` or `location_fb_search()`
- **disable_comments** –
- **is_sidecar** – bool flag for album upload

Returns

configure_to_reel (*upload_id*, *size*)

Finalises a photo story upload. This should not be called directly. Use `post_photo_story()` instead.

Parameters

- **upload_id** –
- **size** – tuple of (width, height)

Returns

configure_video (*upload_id*, *size*, *duration*, *thumbnail_data*, *caption*=", *location*=None, *disable_comments*=False, *is_sidecar*=False)
Finalises a video upload. This should not be called directly. Use `post_video()` instead.

Parameters

- **upload_id** –
- **size** – tuple of (width, height)
- **duration** – in seconds
- **thumbnail_data** – byte string of thumbnail photo
- **caption** –
- **location** – a dict of venue/location information, from `location_search()` or `location_fb_search()`
- **disable_comments** –
- **is_sidecar** – bool flag for album upload

Returns

configure_video_to_reel (*upload_id*, *size*, *duration*, *thumbnail_data*)

Finalises a video story upload. This should not be called directly. Use `post_video_story()` instead.

Parameters

- **upload_id** –
- **size** – tuple of (width, height)
- **duration** – in seconds
- **thumbnail_data** – byte string of thumbnail photo

Returns

configure_video_to_tv (*upload_id, size, duration, thumbnail_data, title, caption*)

Finalises a video IGTV upload. This should not be called directly. Use `post_video_tv()` instead.

Parameters

- **upload_id** –
- **size** – tuple of (width, height)
- **duration** – in seconds
- **thumbnail_data** – byte string of thumbnail photo
- **title** –
- **caption** –

Returns

cookie_jar

The client's cookiejar instance.

create_collection (*name, added_media_ids=None*)

Create a new collection.

Parameters

- **name** – Name for the collection
- **added_media_ids** – list of media_ids

Returns

```
{
  "status": "ok",
  "collection_id": "1700000000123",
  "cover_media": {
    "media_type": 1,
    "original_width": 1080,
    "original_height": 1080,
    "id": 1492726080000000,
    "image_versions2": {
      "candidates": [
        {
          "url": "http://scontent-xx4-1.cdninstagram.com/...123.jpg",
          "width": 1080,
          "height": 1080
        },
        ...
      ]
    }
  },
  "collection_name": "A Collection"
}
```

csrftoken

The client's current csrf token

current_user ()

Get current user info

delete_collection (*collection_id*)

Delete a collection.

Parameters `collection_id` – Collection ID

Returns

```
{  
  "status": "ok"  
}
```

`delete_comment` (*media_id*, *comment_id*)

Delete a comment

Parameters

- **`media_id`** – Media id
- **`comment_id`** – Comment id

Returns

```
{"status": "ok"}
```

`delete_like` (*media_id*, *module_name*='feed_timeline')

Unlike a post

Parameters

- **`media_id`** –
- **`module_name`** – Example: 'feed_timeline', 'video_view', 'photo_view'

Returns

```
{"status": "ok"}
```

`delete_media` (*media_id*)

Delete a media

Parameters **`media_id`** – Media id

Returns

```
{"status": "ok", "did_delete": true}
```

`direct_v2_inbox` ()

Get v2 inbox

`direct_v2_thread` (*thread_id*, ***kwargs*)

Get v2 thread

Parameters

- **`thread_id`** –
- **`kwargs`** –
 - **`cursor`**: For pagination

Returns

`disable_comments` (*media_id*)

Disable comments for a media

Parameters **`media_id`** –

Returns

```
{ "status": "ok" }
```

disable_post_notifications (*user_id*)

Turn off post notifications for specified user.

Parameters *user_id* –

Returns

disable_presence_status ()

Disable presence status setting

discover_chaining (*user_id*)

Get suggested users

Parameters *user_id* –

Returns

discover_channels_home ()

Discover channels home

discover_top_live (***kwargs*)

Get top live broadcasts

Parameters *kwargs* –

- *max_id*: For pagination

Returns

edit_collection (*collection_id*, *added_media_ids*)

Add media IDs to an existing collection.

Parameters

- *collection_id* – Collection ID
- *added_media_ids* – list of media IDs

Returns Returns same object as [`create\_collection\(\)`](#)

edit_media (*media_id*, *caption*, *usertags=None*)

Edit a media's caption

Parameters

- *media_id* – Media id
- *caption* – Caption text
- *usertags* – array of *user_ids* and positions in the format below:

```
usertags = [  
    { "user_id": 4292127751, "position": [0.625347, 0.4384531] }  
]
```

Returns

edit_profile (*first_name*, *biography*, *external_url*, *email*, *phone_number*, *gender*)

Edit profile

Parameters

- *first_name* –

- **biography** –
- **external_url** –
- **email** – Required.
- **phone_number** –
- **gender** – male: 1, female: 2, unspecified: 3

Returns

enable_comments (*media_id*)

Enable comments for a media

Parameters *media_id* –

Returns

```
{"status": "ok"}
```

enable_post_notifications (*user_id*)

Turn on post notifications for specified user.

Parameters *user_id* –

Returns

enable_presence_status ()

Enable presence status setting

explore (***kwargs*)

Get explore items

Parameters *kwargs* –

- **max_id**: For pagination

Returns

feed_liked (***kwargs*)

Get liked feed

Parameters *kwargs* –

- **max_id**: For pagination. Taken from `next_max_id` in the previous page.

Returns

feed_location (*location_id*, *rank_token*, ***kwargs*)

This endpoint is believed to be obsolete. Do not use. Replaced by `location_section()`.

Parameters

- **location_id** –
- **rank_token** – Required for paging through a single feed and can be generated with `generate_uuid()`. You should use the same `rank_token` for paging through a single location.

Returns

feed_only_me (***kwargs*)

Get feed of archived media

:param *kwargs*

feed_popular (**kwargs)

Get popular feed. This endpoint is believed to be obsolete. Do not use.

feed_tag (tag, rank_token, **kwargs)

Get tag feed

Parameters

- **tag** –
- **rank_token** – Required for paging through a single feed and can be generated with `generate_uuid()`. You should use the same rank_token for paging through a single tag feed.
- **kwargs** –

Returns

feed_timeline (**kwargs)

Get timeline feed. To get a new timeline feed, you can mark a set of media as seen by setting `seen_posts = comma-separated list of media IDs`. Example: `api.feed_timeline(seen_posts='123456789_12345,987654321_54321')`

friendships_block (user_id)

Block a user

Parameters **user_id** – User id

Returns

```
{
  "status": "ok",
  "incoming_request": false,
  "is_blocking_reel": false,
  "followed_by": false,
  "is_muting_reel": false,
  "outgoing_request": false,
  "following": false,
  "blocking": true,
  "is_private": false
}
```

friendships_create (user_id)

Follow a user

Parameters **user_id** – User id

Returns

```
{
  "status": "ok",
  "friendship_status": {
    "incoming_request": false,
    "followed_by": false,
    "outgoing_request": false,
    "following": true,
    "blocking": false,
    "is_private": false
  }
}
```

friendships_destroy (*user_id*, ***kwargs*)

Unfollow a user

Parameters

- **user_id** – User id
- **kwargs** –

Returns

```
{
  "status": "ok",
  "incoming_request": false,
  "is_blocking_reel": false,
  "followed_by": false,
  "is_muting_reel": false,
  "outgoing_request": false,
  "following": false,
  "blocking": false,
  "is_private": false
}
```

friendships_pending ()

Get pending follow requests

friendships_show (*user_id*)

Get friendship status with user id

Parameters **user_id** –

Returns

```
{
  "status": "ok",
  "incoming_request": false,
  "is_blocking_reel": false,
  "followed_by": false,
  "is_muting_reel": false,
  "outgoing_request": false,
  "following": false,
  "blocking": false,
  "is_private": false
}
```

friendships_show_many (*user_ids*)

Get friendship status with multiple user ids

Parameters **user_ids** – list of user ids

Returns

```
{
  "status": "ok",
  "friendship_statuses": {
    "123456789": {
      "following": false,
      "incoming_request": true,
      "outgoing_request": false,
      "is_private": false
    }
  }
}
```

(continues on next page)

(continued from previous page)

```
}  
}
```

friendships_unblock (*user_id*)

Unblock a user

Parameters *user_id* – User id**Returns**

```
{  
    "status": "ok",  
    "incoming_request": false,  
    "is_blocking_reel": false,  
    "followed_by": false,  
    "is_muting_reel": false,  
    "outgoing_request": false,  
    "following": false,  
    "blocking": false,  
    "is_private": false  
}
```

generate_adid (*seed=None*)

Generate an Advertising ID based on the login username since the Google Ad ID is a personally identifying but resettable ID.

Returns**classmethod generate_deviceid** (*seed=None*)

Generate an android device ID

Parameters *seed* – Seed value to generate a consistent device ID**Returns****static generate_useragent** (***kwargs*)

Helper method to generate a useragent string based on device parameters

Parameters *kwargs* –

- *app_version*
- *android_version*
- *android_release*
- *brand*
- *device*
- *model*
- *dpi*
- *resolution*
- *chipset*

Returns A compatible user agent string**classmethod generate_uuid** (*return_hex=False, seed=None*)

Generate uuid

Parameters

- **return_hex** – Return in hex format
- **seed** – Seed value to generate a consistent uuid

Returns

highlight_create (*media_ids*, *cover_media_id=None*, *title='Highlights'*, *source='self_profile'*)

Create a new highlight

Parameters

- **media_ids** – A list of media_ids
- **cover_media_id** – The media_id for the highlight cover image
- **title** – Title of the highlight
- **module** – The UI module via which the highlight is created

highlight_delete (*highlight_id*)

Deletes specified highlight

Parameters **highlight_id** – highlight_id, example 'highlight:1770000'

highlight_edit (*highlight_id*, *cover_media_id=None*, *added_media_ids=[]*, *removed_media_ids=[]*, *title=None*, *source='story_viewer'*)

Edits a highlight

Parameters

- **highlight_id** – highlight_id, example 'highlight:1770000'
- **cover_media_id** – The media_id for the highlight cover image
- **added_media_ids** – List of media_id to be added
- **removed_media_ids** – List of media_id to be removed
- **title** – Title of the highlight
- **module** – The UI module via which the highlight is created

highlights_user_feed (*user_id*)

Returns a user's highlight tray

Parameters **user_id** –

ignore_user (*user_id*)

Ignore a user's follow request.

Parameters **user_id** –

Returns

insights ()

Get insights :param day: :return:

link (*contacts*, ***kwargs*)

Sync contacts with instagram app

Parameters **contacts** – list of contact entities. Examples: {

```
    'first_name':      'khatam_testing',    'phone_numbers':    ['+989395405909'],
    'email_addresses': ['khatam_testing@gmail.com']
},
```

Returns

- list of user accounts created based on contacts

location_fb_search (*query*, *rank_token*, *exclude_list=[]*, ***kwargs*)

Search for locations by query text

Parameters

- **query** – search terms
- **rank_token** – Required for paging through a single feed. See examples/pagination.py
- **exclude_list** – List of numerical location IDs to exclude
- **kwargs** –

Returns

location_info (*location_id*)

Get a location info

Parameters **location_id** –

Returns

```
{
  "status": "ok",
  "location": {
    "external_source": "facebook_places",
    "city": "",
    "name": "Berlin Brandenburger Tor",
    "facebook_places_id": 114849465334163,
    "address": "Pariser Platz",
    "lat": 52.51588,
    "pk": 229573811,
    "lng": 13.37892
  }
}
```

location_related (*location_id*, ***kwargs*)

Get related locations

Parameters **location_id** –

Returns

location_search (*latitude*, *longitude*, *query=None*, ***kwargs*)

Location search

Parameters

- **latitude** –
- **longitude** –
- **query** –

Returns

location_section (*location_id*, *rank_token*, *tab='ranked'*, ***kwargs*)

Get a location feed

Parameters

- **location_id** –

- **rank_token** – Required for paging through a single feed and can be generated with `generate_uuid()`. You should use the same rank_token for paging through a single location.
- **tab** – One of ‘ranked’, ‘recent’

Kwargs extract: return the array of media items only **page:** for pagination **next_media_ids:** array of media_id (int) for pagination **max_id:** for pagination

Returns

location_stories (*location_id*, ***kwargs*)

Get a location story feed

Parameters

- **location_id** –
- **rank_token** – Required for paging through a single feed and can be generated with `generate_uuid()`. You should use the same rank_token for paging through a single location.

Returns

login ()

Login.

logout ()

Logout user

media_comments (*media_id*, ***kwargs*)

Get media comments. Fixed at 20 comments returned per page.

Parameters

- **media_id** – Media id
- **kwargs** – **max_id:** For pagination

Returns

media_info (*media_id*)

Get media info

Parameters **media_id** –

Returns

media_likers (*media_id*, ***kwargs*)

Get users who have liked a post

Parameters **media_id** –

Returns

media_likers_chrono (*media_id*)

EXPERIMENTAL ENDPOINT, INADVISABLE TO USE. Get users who have liked a post in chronological order

Parameters **media_id** –

Returns

media_n_comments (*media_id*, *n=150*, *reverse=False*, ***kwargs*)

Helper method to retrieve n number of comments for a media id

Parameters

- **media_id** – Media id
- **n** – Minimum number of comments to fetch
- **reverse** – Reverse list of comments (ordered by created_time)
- **kwargs** –

Returns

media_only_me (*media_id*, *media_type*, *undo=False*)

Archive/unarchive a media so that it is only viewable by the owner.

Parameters

- **media_id** –
- **media_type** – One of `MediaTypes.PHOTO`, `MediaTypes.VIDEO`, or `MediaTypes.CAROUSEL`
- **undo** – bool

Returns

```
{"status": "ok"}
```

media_permalink (*media_id*)

Get media permalink

Parameters **media_id** –

Returns

media_seen (*reels*)

Mark multiple stories as seen

Parameters **reels** – A list of reel media objects, or a dict of `media_ids` and timings as defined below.

```
{
    "1309763051087626108_124317_124317": ["1470355944_1470372029"],
    "1309764045355643149_124317_124317": ["1470356063_1470372039"],
    "1309818450243415912_124317_124317": ["1470362548_1470372060"],
    "1309764653429046112_124317_124317": ["1470356135_1470372049"],
    "1309209597843679372_124317_124317": ["1470289967_1470372013"]
}

where
    1309763051087626108_124317 = <media_id>,
    124317 = <media.owner_id>
    1470355944_1470372029 is <media_created_time>_<view_time>
```

Returns

media_undo_only_me (*media_id*, *media_type*)

Undo making a media only me.

Parameters

- **media_id** –
- **media_type** – One of `MediaTypes.PHOTO`, `MediaTypes.VIDEO`, or `MediaTypes.CAROUSEL`

medias_info (*media_ids*)

Get multiple media infos

Parameters *media_ids* – list of media ids

Returns

megaphone_log (*log_type='feed_aysf', action='seen', reason='', **kwargs*)

A tracking endpoint of sorts

Parameters

- *log_type* –
- *action* –
- *reason* –
- *kwargs* –

Returns

news (***kwargs*)

Get news feed of accounts the logged in account is following. This returns the items in the ‘Following’ tab.

news_inbox ()

Get inbox feed of activity related to the logged in account. This returns the items in the ‘You’ tab.

oembed (*url, **kwargs*)

Get oembed info

Parameters

- *url* –
- *kwargs* –

Returns

phone_id

Current phone ID. For use in certain functions.

post_album (*medias, caption='', location=None, **kwargs*)

Post an album of up to 10 photos/videos.

Parameters

- **medias** – an iterable list/collection of media dict objects

```
medias = [
    {"type": "image", "size": (720, 720), "data": "..."},
    {
        "type": "image", "size": (720, 720),
        "usertags": [{"user_id": 4292127751, "position": [0.625347,
→0.4384531]}],
        "data": "...",
    },
    {"type": "video", "size": (720, 720), "duration": 12.4,
→"thumbnail": "...", "data": "..."}
]
```

- *caption* –
- *location* –

Returns

post_comment (*media_id*, *comment_text*)

Post a comment. Comment text validation according to https://www.instagram.com/developer/endpoints/comments/#post_media_comments

Parameters

- **media_id** – Media id
- **comment_text** – Comment text

Returns

```
{
  "comment": {
    "status": "Active",
    "media_id": 123456789,
    "text": ":)",
    "created_at": 1479453671.0,
    "user": {
      "username": "x",
      "has_anonymous_profile_picture": false,
      "profile_pic_url": "http://scontent-sit4-1.cdninstagram.com/
↪abc.jpg",
      "full_name": "x",
      "pk": 123456789,
      "is_verified": false,
      "is_private": false
    },
    "content_type": "comment",
    "created_at_utc": 1479482471,
    "pk": 17865505612040669,
    "type": 0
  },
  "status": "ok"
}
```

post_like (*media_id*, *module_name*='feed_timeline')

Like a post

Parameters

- **media_id** – Media id
- **module_name** – Example: 'feed_timeline', 'video_view', 'photo_view'

Returns

```
{"status": "ok"}
```

post_photo (*photo_data*, *size*, *caption*="", *upload_id*=None, *to_reel*=False, ***kwargs*)

Upload a photo.

[CAUTION] FLAKY, IG is very finicky about sizes, etc, needs testing.

Parameters

- **photo_data** – byte string of the image
- **size** – tuple of (width, height)
- **caption** –

- **upload_id** –
- **to_reel** – a Story photo
- **kwargs** –
 - **location**: a dict of venue/location information, from `location_search()` or `location_fb_search()`
 - **disable_comments**: bool to disable comments

Returns

post_photo_story (*photo_data*, *size*)

Upload a photo story

Parameters

- **photo_data** – byte string of the image
- **size** – tuple of (width, height)

Returns

post_video (*video_data*, *size*, *duration*, *thumbnail_data*, *caption*=", *title*=None, *to_reel*=False, *to_tv*=False, ***kwargs*)

Upload a video

[CAUTION] FLAKY, IG is very picky about sizes, etc, needs testing.

Parameters

- **video_data** – byte string or a file-like object of the video content
- **size** – tuple of (width, height)
- **duration** – in seconds
- **thumbnail_data** – byte string of the video thumbnail content
- **caption** –
- **title** – to IGTV upload
- **to_reel** – post to reel as Story
- **to_reel** – post to IGTV
- **kwargs** –
 - **location**: a dict of venue/location information, from `location_search()` or `location_fb_search()`
 - **disable_comments**: bool to disable comments
 - **max_retry_count**: maximum attempts to reupload. Default 10.

Returns

post_video_igtv (*video_data*, *size*, *duration*, *thumbnail_data*, *title*, *caption*)

Upload a video IGTV

Parameters

- **video_data** – byte string or a file-like object of the video content
- **size** – tuple of (width, height)
- **duration** – in seconds

- **thumbnail_data** – byte string of the video thumbnail content
- **title** –
- **caption** –

Returns

post_video_story (*video_data, size, duration, thumbnail_data*)

Upload a video story

Parameters

- **video_data** – byte string or a file-like object of the video content
- **size** – tuple of (width, height)
- **duration** – in seconds
- **thumbnail_data** – byte string of the video thumbnail content

Returns

presence_status ()

Get presence status setting

radio_type

For use in certain endpoints

ranked_recipients ()

Get ranked recipients

recent_recipients ()

Get recent recipients

classmethod reel_compatible_aspect_ratio (*size*)

Helper method to check aspect ratio for story uploads

Parameters **size** – tuple of (width, height)

Returns True/False

static reel_ratios ()

Deprecated. Use MediaRatios.reel instead. Acceptable min, max values of with/height ratios for a story upload

Returns tuple of (min. ratio, max. ratio)

reels_media (*user_ids, **kwargs*)

Get multiple users' reel/story media

Parameters

- **user_ids** – list of user IDs
- **kwargs** –

Returns

reels_tray (***kwargs*)

Get story reels tray

remove_follower (*user_id*)

Remove a follower.

Parameters **user_id** –

Returns

remove_profile_picture ()

Remove profile picture

replay_broadcast_comments (*broadcast_id*, *starting_offset=0*, *encoding_tag='instagram_dash_remuxed'*)

Get comments for a post live broadcast.

Parameters

- **broadcast_id** –
- **starting_offset** –
- **encoding_tag** –

Returns

replay_broadcast_likes (*broadcast_id*, *starting_offset=0*, *encoding_tag='instagram_dash_remuxed'*)

Get likes for a post live broadcast.

Parameters

- **broadcast_id** –
- **starting_offset** –
- **encoding_tag** –

Returns

save_photo (*media_id*, *added_collection_ids=None*)

Save a photo

Parameters

- **media_id** – Media id
- **added_collection_ids** – optional list of collection IDs to add the media to

Returns

```
{ "status": "ok" }
```

saved_feed (***kwargs*)

Get saved photo feed

Parameters **kwargs** –

- **count**: Limit the number of items returned

Returns

search_igtv (*text*)

Search igtv

Parameters **text** – Search term

search_users (*query*, ***kwargs*)

Search users

Parameters **query** – Search string

Returns

self_feed (***kwargs*)

Get authenticated user's own feed

set_account_private()

Make account private

set_account_public()

Make account public

set_presence_status() (*disabled*)

Set presence status setting

Parameters **disabled** – True if disabling, else False

set_reel_block_status() (*user_ids*, *block_status*='block')

Unhide your stories from a specific user

Parameters

- **user_ids** – list of user IDs
- **block_status** – One of 'block', 'unblock'

Returns

```
{
  "friendship_statuses": {
    "123456790": {
      "following": true,
      "is_private": false,
      "incoming_request": false,
      "outgoing_request": false,
      "is_blocking_reel": true,
      "is_muting_reel": false
    },
    "123456791": {
      "following": true,
      "is_private": false,
      "incoming_request": false,
      "outgoing_request": false,
      "is_blocking_reel": true,
      "is_muting_reel": false
    }
  },
  "status": "ok"
}
```

set_reel_settings() (*message_prefs*, *allow_story_reshare*=None, *reel_auto_archive*=None, *save_to_camera_roll*=None)

Set story message replies settings

Parameters

- **message_prefs** – One of 'anyone', 'following', 'off'
- **allow_story_reshare** – bool
- **auto_archive** – One of 'on', 'off'
- **save_to_camera_roll** – bool

Returns

```
{
  "message_prefs": "off",
```

(continues on next page)

(continued from previous page)

```

    "status": "ok"
}

```

settings

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

static standard_ratios()

Deprecated. Use `MediaRatios.standard` instead. Acceptable min, max values of with/height ratios for a standard media upload

Returns tuple of (min. ratio, max. ratio)

stickers (*sticker_type='static_stickers', location=None*)

Get sticker assets

Parameters

- **sticker_type** – One of ['static_stickers']
- **location** – dict containing 'lat', 'lng', 'horizontalAccuracy'. Example: {'lat': "", 'lng': "", 'horizontalAccuracy': ""} 'horizontalAccuracy' is a float in meters representing the estimated horizontal accuracy [https://developer.android.com/reference/android/location/Location.html#getAccuracy\(\)](https://developer.android.com/reference/android/location/Location.html#getAccuracy())

Returns**stories_archive** (***kwargs*)

Returns the authenticated user's story archive. The returned items's id value is passed to `reels_media()` to retrieve

Example:

```

archived_stories = api.stories_archive()
if archived_stories.get('items'):
    item_ids = [a['id'] for a in archived_stories['items']]
    archived_stories_media = api.reels_media(user_ids=item_ids)

```

Returns

```

{
  "items": [{
    "timestamp": 1510090000,
    "media_count": 3,
    "id": "archiveDay:1710000000",
    "reel_type": "archive_day_reel",
    "latest_reel_media": 1510090000
  }],
  "num_results": 1,
  "more_available": false,
  "max_id": null,
  "status": "ok"
}

```

story_viewers (*story_pk, **kwargs*)

Get list of story viewers

Parameters

- **story_pk** – Story media's PK identifier, e.g. "1700000123"

- **kwargs** – **max_id**: For pagination

Returns

suggested_broadcasts (***kwargs*)

Get suggested broadcasts

Parameters **kwargs** –

Returns

sync (*prelogin=False*)

Synchronise experiments.

tag_follow (*tag*)

Follow a tag

Parameters **tag** –

Returns

tag_follow_suggestions ()

Get suggestions for tags to follow

tag_info (*tag*)

Get tag info

Parameters **tag** –

Returns

tag_related (*tag, **kwargs*)

Get related tags

Parameters **tag** –

Returns

tag_search (*text, rank_token, exclude_list=[], **kwargs*)

Search tag

Parameters

- **text** – Search term
- **rank_token** – Required for paging through a single feed. See examples/pagination.py
- **exclude_list** – List of numerical tag IDs to exclude
- **kwargs** –

Returns

tag_section (*tag, tab='top', **kwargs*)

Get a tag feed section

Parameters

- **tag** – tag text (without '#')
- **tab** – One of 'top', 'recent', 'places'

Kwargs **extract**: return the array of media items only **page**: for pagination **next_media_ids**: array of media_id (int) for pagination **max_id**: for pagination

Returns

tag_unfollow (*tag*)

Unfollow a tag

Parameters **tag** –

Returns

tags_user_following (*user_id*)

Get tags a user is following

Parameters **user_id** –

Returns

timezone_offset

Timezone offset in seconds. For use in certain functions.

token

For compatibility. Equivalent to `csrftoken()`

top_live_status (*broadcast_ids*)

Get status for a list of broadcast_ids

Returns

top_search (*query*)

Search for top matching hashtags, users, locations

Parameters **query** – search terms

Returns

translate (*object_id, object_type*)

Parameters

- **object_id** – id value for the object
- **object_type** – One of [1, 2, 3] where 1 = CAPTION - unsupported 2 = COMMENT - unsupported 3 = BIOGRAPHY

Returns

tvchannel (*channel_id, **kwargs*)

Get channel

Parameters **channel_id** – One of ‘for_you’, ‘chrono_following’, ‘popular’, ‘continue_watching’ (as returned by `tvguide()`) or for a user ‘user_12345’ where user_id = ‘12345’

tvguide ()

TV guide to popular, following, suggested channels, etc

unblock_friend_reel (*user_id*)

Unhide your stories from a specific user

Parameters **user_id** – User id

Returns

```
{
    "status": "ok",
    "incoming_request": false,
    "is_blocking_reel": false,
    "followed_by": false,
```

(continues on next page)

(continued from previous page)

```
"is_muting_reel": false,  
"outgoing_request": false,  
"following": false,  
"blocking": true,  
"is_private": false  
}
```

unlink()

Unsync contacts with instagram

Returns

- list of users created based on contacts

unsave_photo (*media_id*, *removed_collection_ids=None*)

Unsave a photo

Parameters

- **media_id** –
- **removed_collection_ids** – optional list of collection IDs to remove the media from

Returns

```
{ "status": "ok" }
```

user_agent

Returns the useragent string that the client is currently using.

user_broadcast (*user_id*)

Helper method to get a user's broadcast if there is one currently live. Returns none otherwise.

Parameters **user_id** –**Returns****user_detail_info** (*user_id*, ***kwargs*)

EXPERIMENTAL ENDPOINT, INADVISABLE TO USE. Get user detailed info

Parameters

- **user_id** –
- **kwargs** –
 - **max_id**: For pagination
 - **min_timestamp**: For pagination

Returns**user_feed** (*user_id*, ***kwargs*)

Get the feed for the specified user id

Parameters

- **user_id** –
- **kwargs** –
 - **max_id**: For pagination
 - **min_timestamp**: For pagination

Returns**user_followers** (*user_id*, *rank_token*, ***kwargs*)

Get user followers

Parameters

- **user_id** –
- **rank_token** – Required for paging through a single feed and can be generated with `generate_uuid()`. You should use the same rank_token for paging through a single user followers.
- **kwargs** –
 - **query**: Search within the user followers
 - **max_id**: For pagination

Returns**user_following** (*user_id*, *rank_token*, ***kwargs*)

Get user followings

Parameters

- **user_id** –
- **rank_token** – Required for paging through a single feed and can be generated with `generate_uuid()`. You should use the same rank_token for paging through a single user following.
- **kwargs** –
 - **query**: Search within the user following
 - **max_id**: For pagination

Returns**user_info** (*user_id*)

Get user info for a specified user id

Parameters **user_id** –**Returns****user_map** (*user_id*)

Get a list of geo-tagged media from a user

Parameters **user_id** – User id**Returns****user_reel_media** (*user_id*, ***kwargs*)

Get user story/reel media

Parameters

- **user_id** –
- **kwargs** –

Returns**user_reel_settings** ()

Get user reel settings

user_story_feed (*user_id*)

Get a user's story feed and current/replay broadcasts (if available)

Parameters *user_id* –

Returns

username_feed (*user_name*, ***kwargs*)

Get the feed for the specified user name

Parameters

- **user_name** –
- **kwargs** –
 - **max_id**: For pagination
 - **min_timestamp**: For pagination

Returns

username_info (*user_name*)

Get user info for a specified user name :param user_name: :return:

usertag_feed (*user_id*, ***kwargs*)

Get a usertag feed

Parameters

- **user_id** –
- **kwargs** –

Returns

usertag_self_remove (*media_id*)

Remove your own user tag from a media post

Parameters *media_id* – Media id

Returns

static validate_useragent (*value*)

Helper method to validate a useragent string for format correctness

Parameters *value* –

Returns

class `instapi.ClientCompatPatch`

Utility to make entities from the private api similar to the ones from the public one by adding the necessary properties, and if required, remove any incompatible properties (to save storage space for example).

__init__

Initialize self. See help(type(self)) for accurate signature.

classmethod `comment` (*comment*, *drop_incompat_keys=False*)

Patch a comment object

classmethod `list_user` (*user*, *drop_incompat_keys=False*)

Patch a list user object, example in `Client.user_following()`, `Client.user_followers()`, `Client.search_users()`

classmethod `media` (*media*, *drop_incompat_keys=False*)

Patch a media object

```
classmethod user (user, drop_incompat_keys=False)  
    Patch a user object  
exception instapi.ClientError (msg, code=None, error_response="")  
    Generic error class, catch-all for most client issues.  
exception instapi.ClientLoginError (msg, code=None, error_response="")  
    Raised when login fails.  
exception instapi.ClientLoginRequiredError (msg, code=None, error_response="")  
    Raised when login is required.  
exception instapi.ClientCookieExpiredError (msg, code=None, error_response="")  
    Raised when cookies have expired.  
class instapi.MediaRatios  
    Class holding valid aspect ratios (width: height) for media uploads.  
  
    reel = (0.5625, 0.75)  
        Acceptable min, max values of with/height ratios for a story or a IGTV upload  
  
    standard = (0.8, 1.9148936170212767)  
        Acceptable min, max values of with/height ratios for a standard media upload  
  
class instapi.MediaTypes  
    Psuedo enum-ish/lookup class for media types.  
  
    CAROUSEL = 8  
        Carousel/Album type  
  
    PHOTO = 1  
        Photo type  
  
    VIDEO = 2  
        Video type  
  
    static id_to_name (media_type_id)  
        Convert a media type ID to its name  
  
    static name_to_id (media_type_name)  
        Convert a media type name to its ID
```


i

instapi, 6

Symbols

`__init__` (*instapi.ClientCompatPatch* attribute), 34
`__init__` () (*instapi.Client* method), 6

A

`authenticated_user_id` (*instapi.Client* attribute), 7
`authenticated_user_name` (*instapi.Client* attribute), 7
`autocomplete_user_list` () (*instapi.Client* method), 7

B

`block_friend_reel` () (*instapi.Client* method), 7
`blocked_reels` () (*instapi.Client* method), 7
`blocked_user_list` () (*instapi.Client* method), 7
`broadcast_comment` () (*instapi.Client* method), 7
`broadcast_comments` () (*instapi.Client* method), 8
`broadcast_heartbeat_and_viewercount` () (*instapi.Client* method), 8
`broadcast_info` () (*instapi.Client* method), 8
`broadcast_like` () (*instapi.Client* method), 9
`broadcast_like_count` () (*instapi.Client* method), 9
`bulk_delete_comments` () (*instapi.Client* method), 9
`bulk_translate` () (*instapi.Client* method), 9

C

`CAROUSEL` (*instapi.MediaTypes* attribute), 35
`change_profile_picture` () (*instapi.Client* method), 9
`check_username` () (*instapi.Client* method), 9
`Client` (class in *instapi*), 6
`ClientCompatPatch` (class in *instapi*), 34
`ClientCookieExpiredError`, 35
`ClientError`, 35
`ClientLoginError`, 35
`ClientLoginRequiredError`, 35

`collection_feed` () (*instapi.Client* method), 9
`comment` () (*instapi.ClientCompatPatch* class method), 34
`comment_inline_replies` () (*instapi.Client* method), 10
`comment_like` () (*instapi.Client* method), 10
`comment_likers` () (*instapi.Client* method), 10
`comment_replies` () (*instapi.Client* method), 10
`comment_unlike` () (*instapi.Client* method), 10
`compatible_aspect_ratio` () (*instapi.Client* class method), 10
`configure` () (*instapi.Client* method), 10
`configure_to_reel` () (*instapi.Client* method), 11
`configure_video` () (*instapi.Client* method), 11
`configure_video_to_reel` () (*instapi.Client* method), 11
`configure_video_to_tv` () (*instapi.Client* method), 11
`cookie_jar` (*instapi.Client* attribute), 12
`create_collection` () (*instapi.Client* method), 12
`csrftoken` (*instapi.Client* attribute), 12
`current_user` () (*instapi.Client* method), 12

D

`delete_collection` () (*instapi.Client* method), 12
`delete_comment` () (*instapi.Client* method), 13
`delete_like` () (*instapi.Client* method), 13
`delete_media` () (*instapi.Client* method), 13
`direct_v2_inbox` () (*instapi.Client* method), 13
`direct_v2_thread` () (*instapi.Client* method), 13
`disable_comments` () (*instapi.Client* method), 13
`disable_post_notifications` () (*instapi.Client* method), 14
`disable_presence_status` () (*instapi.Client* method), 14
`discover_chaining` () (*instapi.Client* method), 14
`discover_channels_home` () (*instapi.Client* method), 14
`discover_top_live` () (*instapi.Client* method), 14

E

`edit_collection()` (*instapi.Client method*), 14
`edit_media()` (*instapi.Client method*), 14
`edit_profile()` (*instapi.Client method*), 14
`enable_comments()` (*instapi.Client method*), 15
`enable_post_notifications()` (*instapi.Client method*), 15
`enable_presence_status()` (*instapi.Client method*), 15
`explore()` (*instapi.Client method*), 15

F

`feed_liked()` (*instapi.Client method*), 15
`feed_location()` (*instapi.Client method*), 15
`feed_only_me()` (*instapi.Client method*), 15
`feed_popular()` (*instapi.Client method*), 15
`feed_tag()` (*instapi.Client method*), 16
`feed_timeline()` (*instapi.Client method*), 16
`friendships_block()` (*instapi.Client method*), 16
`friendships_create()` (*instapi.Client method*), 16
`friendships_destroy()` (*instapi.Client method*), 16
`friendships_pending()` (*instapi.Client method*), 17
`friendships_show()` (*instapi.Client method*), 17
`friendships_show_many()` (*instapi.Client method*), 17
`friendships_unblock()` (*instapi.Client method*), 18

G

`generate_adid()` (*instapi.Client method*), 18
`generate_deviceid()` (*instapi.Client class method*), 18
`generate_useragent()` (*instapi.Client static method*), 18
`generate_uuid()` (*instapi.Client class method*), 18

H

`highlight_create()` (*instapi.Client method*), 19
`highlight_delete()` (*instapi.Client method*), 19
`highlight_edit()` (*instapi.Client method*), 19
`highlights_user_feed()` (*instapi.Client method*), 19

I

`id_to_name()` (*instapi.MediaTypes static method*), 35
`ignore_user()` (*instapi.Client method*), 19
`insights()` (*instapi.Client method*), 19
`instapi` (*module*), 6

L

`link()` (*instapi.Client method*), 19

`list_user()` (*instapi.ClientCompatPatch class method*), 34
`location_fb_search()` (*instapi.Client method*), 20
`location_info()` (*instapi.Client method*), 20
`location_related()` (*instapi.Client method*), 20
`location_search()` (*instapi.Client method*), 20
`location_section()` (*instapi.Client method*), 20
`location_stories()` (*instapi.Client method*), 21
`login()` (*instapi.Client method*), 21
`logout()` (*instapi.Client method*), 21

M

`media()` (*instapi.ClientCompatPatch class method*), 34
`media_comments()` (*instapi.Client method*), 21
`media_info()` (*instapi.Client method*), 21
`media_likers()` (*instapi.Client method*), 21
`media_likers_chrono()` (*instapi.Client method*), 21
`media_n_comments()` (*instapi.Client method*), 21
`media_only_me()` (*instapi.Client method*), 22
`media_permalink()` (*instapi.Client method*), 22
`media_seen()` (*instapi.Client method*), 22
`media_undo_only_me()` (*instapi.Client method*), 22
`MediaRatios` (*class in instapi*), 35
`medias_info()` (*instapi.Client method*), 22
`MediaTypes` (*class in instapi*), 35
`megaphone_log()` (*instapi.Client method*), 23

N

`name_to_id()` (*instapi.MediaTypes static method*), 35
`news()` (*instapi.Client method*), 23
`news_inbox()` (*instapi.Client method*), 23

O

`oembed()` (*instapi.Client method*), 23

P

`phone_id` (*instapi.Client attribute*), 23
`PHOTO` (*instapi.MediaTypes attribute*), 35
`post_album()` (*instapi.Client method*), 23
`post_comment()` (*instapi.Client method*), 24
`post_like()` (*instapi.Client method*), 24
`post_photo()` (*instapi.Client method*), 24
`post_photo_story()` (*instapi.Client method*), 25
`post_video()` (*instapi.Client method*), 25
`post_video_igtv()` (*instapi.Client method*), 25
`post_video_story()` (*instapi.Client method*), 26
`presence_status()` (*instapi.Client method*), 26

R

`radio_type` (*instapi.Client attribute*), 26
`ranked_recipients()` (*instapi.Client method*), 26
`recent_recipients()` (*instapi.Client method*), 26

reel (*instapi.MediaRatios attribute*), 35
 reel_compatible_aspect_ratio() (*instapi.Client class method*), 26
 reel_ratios() (*instapi.Client static method*), 26
 reels_media() (*instapi.Client method*), 26
 reels_tray() (*instapi.Client method*), 26
 remove_follower() (*instapi.Client method*), 26
 remove_profile_picture() (*instapi.Client method*), 26
 replay_broadcast_comments() (*instapi.Client method*), 27
 replay_broadcast_likes() (*instapi.Client method*), 27

S

save_photo() (*instapi.Client method*), 27
 saved_feed() (*instapi.Client method*), 27
 search_igtv() (*instapi.Client method*), 27
 search_users() (*instapi.Client method*), 27
 self_feed() (*instapi.Client method*), 27
 set_account_private() (*instapi.Client method*), 27
 set_account_public() (*instapi.Client method*), 28
 set_presence_status() (*instapi.Client method*), 28
 set_reel_block_status() (*instapi.Client method*), 28
 set_reel_settings() (*instapi.Client method*), 28
 settings (*instapi.Client attribute*), 29
 standard (*instapi.MediaRatios attribute*), 35
 standard_ratios() (*instapi.Client static method*), 29
 stickers() (*instapi.Client method*), 29
 stories_archive() (*instapi.Client method*), 29
 story_viewers() (*instapi.Client method*), 29
 suggested_broadcasts() (*instapi.Client method*), 30
 sync() (*instapi.Client method*), 30

T

tag_follow() (*instapi.Client method*), 30
 tag_follow_suggestions() (*instapi.Client method*), 30
 tag_info() (*instapi.Client method*), 30
 tag_related() (*instapi.Client method*), 30
 tag_search() (*instapi.Client method*), 30
 tag_section() (*instapi.Client method*), 30
 tag_unfollow() (*instapi.Client method*), 30
 tags_user_following() (*instapi.Client method*), 31
 timezone_offset (*instapi.Client attribute*), 31
 token (*instapi.Client attribute*), 31
 top_live_status() (*instapi.Client method*), 31
 top_search() (*instapi.Client method*), 31

translate() (*instapi.Client method*), 31
 tvchannel() (*instapi.Client method*), 31
 tvguide() (*instapi.Client method*), 31

U

unblock_friend_reel() (*instapi.Client method*), 31
 unlink() (*instapi.Client method*), 32
 unsave_photo() (*instapi.Client method*), 32
 user() (*instapi.ClientCompatPatch class method*), 34
 user_agent (*instapi.Client attribute*), 32
 user_broadcast() (*instapi.Client method*), 32
 user_detail_info() (*instapi.Client method*), 32
 user_feed() (*instapi.Client method*), 32
 user_followers() (*instapi.Client method*), 33
 user_following() (*instapi.Client method*), 33
 user_info() (*instapi.Client method*), 33
 user_map() (*instapi.Client method*), 33
 user_reel_media() (*instapi.Client method*), 33
 user_reel_settings() (*instapi.Client method*), 33
 user_story_feed() (*instapi.Client method*), 33
 username_feed() (*instapi.Client method*), 34
 username_info() (*instapi.Client method*), 34
 usertag_feed() (*instapi.Client method*), 34
 usertag_self_remove() (*instapi.Client method*), 34

V

validate_useragent() (*instapi.Client static method*), 34
 VIDEO (*instapi.MediaTypes attribute*), 35