
Python-Fitbit Documentation

Release 0.3.0

Issac Kelly, Percy Perez, Brad Pitcher

January 24, 2017

| | | |
|----------|---------------------------|-----------|
| 1 | Quickstart | 3 |
| 2 | Fitbit API | 5 |
| 3 | Indices and tables | 11 |

This is a complete python implementation of the Fitbit API.

It uses oAuth for authentication, it supports both us and si measurements

Quickstart

If you are only retrieving data that doesn't require authorization, then you can use the unauthorized interface:

```
import fitbit
unauth_client = fitbit.Fitbit('<consumer_key>', '<consumer_secret>')
# certain methods do not require user keys
unauth_client.food_units()
```

Here is an example of authorizing with OAuth 2.0:

```
# You'll have to gather the tokens on your own, or use
# ./gather_keys_oauth2.py
authd_client = fitbit.Fitbit('<consumer_key>', '<consumer_secret>',
                             access_token='<access_token>', refresh_token='<refresh_token>')
authd_client.sleep()
```

Fitbit API

Some assumptions you should note. Anywhere it says `user_id=None`, it assumes the current `user_id` from the credentials given, and passes a – through the API. Anywhere it says `date=None`, it should accept either `None` or a date or datetime object (anything with proper strftime will do), or a string formatted as `%Y-%m-%d`.

```
class fitbit.Fitbit (client_id, client_secret, system='en_US', **kwargs)
```

```
__COLLECTION_RESOURCE (resource, date=None, user_id=None, data=None)
```

Retrieving and logging of each type of collection data.

Arguments: resource, defined automatically via curry [date] defaults to today [user_id] defaults to current logged in user [data] optional, include for creating a record, exclude for access

This implements the following methods:

```
body(date=None, user_id=None, data=None)
activities(date=None, user_id=None, data=None)
foods_log(date=None, user_id=None, data=None)
foods_log_water(date=None, user_id=None, data=None)
sleep(date=None, user_id=None, data=None)
heart(date=None, user_id=None, data=None)
bp(date=None, user_id=None, data=None)
```

•<https://dev.fitbit.com/docs/>

```
__DELETE_COLLECTION_RESOURCE (resource, log_id)
```

deleting each type of collection data

Arguments: resource, defined automatically via curry log_id, required, log entry to delete

This builds the following methods:

```
delete_body(log_id)
delete_activities(log_id)
delete_foods_log(log_id)
delete_foods_log_water(log_id)
delete_sleep(log_id)
delete_heart(log_id)
delete_bp(log_id)
```

```
__food_stats (user_id=None, qualifier='')
```

This builds the convenience methods on initialization:

```
recent_foods(user_id=None, qualifier='')
favorite_foods(user_id=None, qualifier='')
frequent_foods(user_id=None, qualifier='')
```

- <https://dev.fitbit.com/docs/food-logging/#get-favorite-foods>
- <https://dev.fitbit.com/docs/food-logging/#get-frequent-foods>
- <https://dev.fitbit.com/docs/food-logging/#get-recent-foods>

_resource_goal (*resource*, *data*={}, *period*=None)

Handles GETting and POSTing resource goals of all types

accept_invite (*other_user_id*)

Convenience method for `respond_to_invite`

activities_daily_goal (*calories_out*=None, *active_minutes*=None, *floors*=None, *distance*=None, *steps*=None)

Implements the following APIs for period equal to daily

<https://dev.fitbit.com/docs/activity/#get-activity-goals> <https://dev.fitbit.com/docs/activity/#update-activity-goals>

Pass no arguments to get the daily activities goal. Pass any one of the optional arguments to set that component of the daily activities goal.

Arguments: * *calories_out* – New goal value; in an integer format * *active_minutes* – New goal value; in an integer format * *floors* – New goal value; in an integer format * *distance* – New goal value; in the format X.XX or integer * *steps* – New goal value; in an integer format

activities_list ()

<https://dev.fitbit.com/docs/activity/#browse-activity-types>

activities_weekly_goal (*distance*=None, *floors*=None, *steps*=None)

Implements the following APIs for period equal to weekly

<https://dev.fitbit.com/docs/activity/#get-activity-goals> <https://dev.fitbit.com/docs/activity/#update-activity-goals>

Pass no arguments to get the weekly activities goal. Pass any one of the optional arguments to set that component of the weekly activities goal.

Arguments: * *distance* – New goal value; in the format X.XX or integer * *floors* – New goal value; in an integer format * *steps* – New goal value; in an integer format

activity_detail (*activity_id*)

<https://dev.fitbit.com/docs/activity/#get-activity-type>

activity_stats (*user_id*=None, *qualifier*='')

- <https://dev.fitbit.com/docs/activity/#activity-types>
- <https://dev.fitbit.com/docs/activity/#get-favorite-activities>
- <https://dev.fitbit.com/docs/activity/#get-recent-activity-types>
- <https://dev.fitbit.com/docs/activity/#get-frequent-activities>

This implements the following methods:

```
recent_activities(user_id=None, qualifier='')
favorite_activities(user_id=None, qualifier='')
frequent_activities(user_id=None, qualifier='')
```

add_alarm(*device_id*, *alarm_time*, *week_days*, *recurring=False*, *enabled=True*, *label=None*, *snooze_length=None*, *snooze_count=None*, *vibe='DEFAULT'*)
<https://dev.fitbit.com/docs/devices/#add-alarm> *alarm_time* should be a timezone aware datetime object.

add_favorite_activity(*activity_id*)
<https://dev.fitbit.com/docs/activity/#add-favorite-activity>

add_favorite_food(*food_id*)
<https://dev.fitbit.com/docs/food-logging/#add-favorite-food>

body_fat_goal(*fat=None*)
Implements the following APIs

- <https://dev.fitbit.com/docs/body/#get-body-goals>
- <https://dev.fitbit.com/docs/body/#update-body-fat-goal>

Pass no arguments to get the body fat goal. Pass a *fat* argument to update the body fat goal.

Arguments: * *fat* – Target body fat in %; in the format X.XX

body_weight_goal(*start_date=None*, *start_weight=None*, *weight=None*)
Implements the following APIs

- <https://dev.fitbit.com/docs/body/#get-body-goals>
- <https://dev.fitbit.com/docs/body/#update-weight-goal>

Pass no arguments to get the body weight goal. Pass *start_date*, *start_weight* and optionally *weight* to set the weight goal. *weight* is required if it hasn't been set yet.

Arguments: * *start_date* – Weight goal start date; in the format yyyy-MM-dd * *start_weight* – Weight goal start weight; in the format X.XX * *weight* – Weight goal target weight; in the format X.XX

create_food(*data*)
<https://dev.fitbit.com/docs/food-logging/#create-food>

delete_alarm(*device_id*, *alarm_id*)
<https://dev.fitbit.com/docs/devices/#delete-alarm>

delete_favorite_activity(*activity_id*)
<https://dev.fitbit.com/docs/activity/#delete-favorite-activity>

delete_favorite_food(*food_id*)
<https://dev.fitbit.com/docs/food-logging/#delete-favorite-food>

food_detail(*food_id*)
<https://dev.fitbit.com/docs/food-logging/#get-food>

food_goal(*calories=None*, *intensity=None*, *personalized=None*)
Implements the following APIs

<https://dev.fitbit.com/docs/food-logging/#get-food-goals> <https://dev.fitbit.com/docs/food-logging/#update-food-goal>

Pass no arguments to get the food goal. Pass at least *calories* or *intensity* and optionally *personalized* to update the food goal.

Arguments: * *calories* – Manual Calorie Consumption Goal; *calories*, integer; * *intensity* – Food Plan intensity; (MAINTENANCE, EASIER, MEDIUM, KINDAHARD, HARDER); * *personalized* – Food Plan type; True or False

food_units()
<https://dev.fitbit.com/docs/food-logging/#get-food-units>

get_alarms (*device_id*)

<https://dev.fitbit.com/docs/devices/#get-alarms>

get_badges (*user_id=None*)

<https://dev.fitbit.com/docs/friends/#badges>

get_bodyfat (*base_date=None, user_id=None, period=None, end_date=None*)

<https://dev.fitbit.com/docs/body/#get-body-fat-logs> *base_date* should be a datetime.date object (defaults to today), *period* can be '1d', '7d', '30d', '1w', '1m', '3m', '6m', '1y', 'max' or None *end_date* should be a datetime.date object, or None.

You can specify *period* or *end_date*, or neither, but not both.

get_bodyweight (*base_date=None, user_id=None, period=None, end_date=None*)

<https://dev.fitbit.com/docs/body/#get-weight-logs> *base_date* should be a datetime.date object (defaults to today), *period* can be '1d', '7d', '30d', '1w', '1m', '3m', '6m', '1y', 'max' or None *end_date* should be a datetime.date object, or None.

You can specify *period* or *end_date*, or neither, but not both.

get_devices ()

<https://dev.fitbit.com/docs/devices/#get-devices>

get_friends (*user_id=None*)

<https://dev.fitbit.com/docs/friends/#get-friends>

get_friends_leaderboard (*period*)

<https://dev.fitbit.com/docs/friends/#get-friends-leaderboard>

get_meals ()

<https://dev.fitbit.com/docs/food-logging/#get-meals>

get_sleep (*date*)

<https://dev.fitbit.com/docs/sleep/#get-sleep-logs> *date* should be a datetime.date object.

intraday_time_series (*resource, base_date='today', detail_level='1min', start_time=None, end_time=None*)

The intraday time series extends the functionality of the regular time series, but returning data at a more granular level for a single day, defaulting to 1 minute intervals. To access this feature, one must fill out the Private Support form here (see <https://dev.fitbit.com/docs/help/>). For details on the resources available and more information on how to get access, see:

<https://dev.fitbit.com/docs/activity/#get-activity-intraday-time-series>

invite_friend (*data*)

<https://dev.fitbit.com/docs/friends/#invite-friend>

invite_friend_by_email (*email*)

Convenience Method for <https://dev.fitbit.com/docs/friends/#invite-friend>

invite_friend_by_userid (*user_id*)

Convenience Method for <https://dev.fitbit.com/docs/friends/#invite-friend>

list_subscriptions (*collection=''*)

<https://dev.fitbit.com/docs/subscriptions/#getting-a-list-of-subscriptions>

log_activity (*data*)

<https://dev.fitbit.com/docs/activity/#log-activity>

log_sleep (*start_time, duration*)

<https://dev.fitbit.com/docs/sleep/#log-sleep> *start_time* should be a datetime object. We will be using the year, month, day, hour, and minute.

reject_invite (*other_user_id*)

Convenience method for `respond_to_invite`

respond_to_invite (*other_user_id*, *accept=True*)

<https://dev.fitbit.com/docs/friends/#respond-to-friend-invitation>

search_foods (*query*)

<https://dev.fitbit.com/docs/food-logging/#search-foods>

subscription (*subscription_id*, *subscriber_id*, *collection=None*, *method='POST'*)

<https://dev.fitbit.com/docs/subscriptions/>

time_series (*resource*, *user_id=None*, *base_date='today'*, *period=None*, *end_date=None*)

The time series is a LOT of methods, (documented at urls below) so they don't get their own method. They all follow the same patterns, and return similar formats.

Taking liberty, this assumes a `base_date` of today, the current user, and a 1d period.

<https://dev.fitbit.com/docs/activity/#activity-time-series>
time-series

<https://dev.fitbit.com/docs/body/#body-time-series>
time-series

<https://dev.fitbit.com/docs/food-logging/#food-or-water-time-series>
time-series

<https://dev.fitbit.com/docs/heart-rate/#heart-rate-time-series>
time-series

<https://dev.fitbit.com/docs/sleep/#sleep-time-series>
time-series

update_alarm (*device_id*, *alarm_id*, *alarm_time*, *week_days*, *recurring=False*, *enabled=True*, *label=None*, *snooze_length=None*, *snooze_count=None*, *vibe='DEFAULT'*)

<https://dev.fitbit.com/docs/devices/#update-alarm> `alarm_time` should be a timezone aware datetime object.

user_profile_get (*user_id=None*)

Get a user profile. You can get other user's profile information by passing `user_id`, or you can get the current user's by not passing a `user_id`

<https://dev.fitbit.com/docs/user/>

user_profile_update (*data*)

Set a user profile. You can set your user profile information by passing a dictionary of attributes that will be updated.

<https://dev.fitbit.com/docs/user/#update-profile>

water_goal (*target=None*)

Implements the following APIs

<https://dev.fitbit.com/docs/food-logging/#get-water-goal> <https://dev.fitbit.com/docs/food-logging/#update-water-goal>

Pass no arguments to get the water goal. Pass `target` to update it.

Arguments: * `target` – Target water goal in the format X.X, will be set in unit based on locale

Indices and tables

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

Symbols

`_COLLECTION_RESOURCE()` (fitbit.Fitbit method), 5
`_DELETE_COLLECTION_RESOURCE()` (fitbit.Fitbit method), 5
`_food_stats()` (fitbit.Fitbit method), 5
`_resource_goal()` (fitbit.Fitbit method), 6

A

`accept_invite()` (fitbit.Fitbit method), 6
`activities_daily_goal()` (fitbit.Fitbit method), 6
`activities_list()` (fitbit.Fitbit method), 6
`activities_weekly_goal()` (fitbit.Fitbit method), 6
`activity_detail()` (fitbit.Fitbit method), 6
`activity_stats()` (fitbit.Fitbit method), 6
`add_alarm()` (fitbit.Fitbit method), 6
`add_favorite_activity()` (fitbit.Fitbit method), 7
`add_favorite_food()` (fitbit.Fitbit method), 7

B

`body_fat_goal()` (fitbit.Fitbit method), 7
`body_weight_goal()` (fitbit.Fitbit method), 7

C

`create_food()` (fitbit.Fitbit method), 7

D

`delete_alarm()` (fitbit.Fitbit method), 7
`delete_favorite_activity()` (fitbit.Fitbit method), 7
`delete_favorite_food()` (fitbit.Fitbit method), 7

F

`Fitbit` (class in fitbit), 5
`food_detail()` (fitbit.Fitbit method), 7
`food_goal()` (fitbit.Fitbit method), 7
`food_units()` (fitbit.Fitbit method), 7

G

`get_alarms()` (fitbit.Fitbit method), 7
`get_badges()` (fitbit.Fitbit method), 8
`get_bodyfat()` (fitbit.Fitbit method), 8

`get_bodyweight()` (fitbit.Fitbit method), 8
`get_devices()` (fitbit.Fitbit method), 8
`get_friends()` (fitbit.Fitbit method), 8
`get_friends_leaderboard()` (fitbit.Fitbit method), 8
`get_meals()` (fitbit.Fitbit method), 8
`get_sleep()` (fitbit.Fitbit method), 8

I

`intraday_time_series()` (fitbit.Fitbit method), 8
`invite_friend()` (fitbit.Fitbit method), 8
`invite_friend_by_email()` (fitbit.Fitbit method), 8
`invite_friend_by_userid()` (fitbit.Fitbit method), 8

L

`list_subscriptions()` (fitbit.Fitbit method), 8
`log_activity()` (fitbit.Fitbit method), 8
`log_sleep()` (fitbit.Fitbit method), 8

R

`reject_invite()` (fitbit.Fitbit method), 8
`respond_to_invite()` (fitbit.Fitbit method), 9

S

`search_foods()` (fitbit.Fitbit method), 9
`subscription()` (fitbit.Fitbit method), 9

T

`time_series()` (fitbit.Fitbit method), 9

U

`update_alarm()` (fitbit.Fitbit method), 9
`user_profile_get()` (fitbit.Fitbit method), 9
`user_profile_update()` (fitbit.Fitbit method), 9

W

`water_goal()` (fitbit.Fitbit method), 9