
OBPlatform

Release 1.1.0

Wei Mu @ BESTLab

Mar 30, 2022

CONTENTS

1	Features	3
2	Installation	5
2.1	poetry	5
2.2	pip	5
2.3	conda	5
2.4	mamba	6
3	Example	7
4	Usage	9
4.1	Available behavior types	9
5	Changelog	11
6	API Reference	13
6.1	API Reference	13
	Python Module Index	15
	Index	17

A package to interact and download behavior data from [ASHRAE Global Occupant Behavior Database](#). Currently available on PyPI and conda-forge. More features coming in the future.

FEATURES

- List all behavior types available in the database.
- Download data archive (ZIP file) based on behavior type and study id inputs (with progress bar).
- Query studies based on (behaviors, countries, cities, (building type + room type))
- Query available behavior types based on study ids

INSTALLATION

2.1 poetry

```
poetry install
```

2.2 pip

```
pip install --upgrade obplatform
```

2.3 conda

```
conda install -c conda-forge obplatform
```

Note: For Python 3.10: If you see an error like the following when resolving dependencies, it's caused by a [bug](#) in conda with Python 3.10.

```
Collecting package metadata (current_repodata.json): done
Solving environment: failed with initial frozen solve. Retrying with flexible solve.
Collecting package metadata (repodata.json): done
Solving environment: failed with initial frozen solve. Retrying with flexible solve.

PackagesNotFoundError: The following packages are not available from current channels:

- python=3.1
```

Three possible solutions:

1. Create a new conda environment with Python <3.10.
2. Upgrade conda to a new version. (conda released 4.11.0 on 11/22/2021 at GitHub, which fixed the bug for Python 3.10. However, it will still take some time before conda 4.11.0 is available on Anaconda Cloud).
3. Use [mamba](#), which is a reimplementation of the conda package manager in C++. It is much faster and contains less bugs.

2.4 mamba

Once you activate the environment through conda or micromamba:

```
mamba install -c conda-forge obplatform
```

EXAMPLE

```
import logging
import zipfile

import pandas as pd

from obplatform import Connector, logger

connector = Connector()

# List all behaviors available in the database
print(connector.list_behaviors())

# Print progress information
# Comment out the following line to hide progress information
logger.setLevel(logging.INFO)

# Download Plug Load + Occupant Presence behaviors from study 22, 11, and 2.
connector.download_export(
    "data.zip",
    ["Plug_Load", "Occupancy_Measurement"],
    ["22", "11", "2"],
    show_progress_bar=True, # False to disable progress bar
)

behavior_type = "Plug_Load"
study_id = "22"

zf = zipfile.ZipFile("data.zip")
df = pd.read_csv(zf.open(f"{behavior_type}_Study{study_id}.csv"))
print(df.head())

# List all behaviors available in study 1, 2, 3, and 4
json_study_behaviors = connector.list_behaviors_in_studies(studies=["1", "2", "3", "4"])
print(json_study_behaviors)

# List all studies available in the database, filtered by behavior types,
# countries, cities, {building type, room_type} combinations.
json_studies = connector.list_studies(
    behaviors=["Occupancy_Measurement", "Plug_Load"],
    countries=["USA", "UK"],
```

(continues on next page)

(continued from previous page)

```
cities=["Palo Alto", "Coventry", "San Antonio"],
buildings=[
    {
        "building_type": "Educational",
        "room_type": "Classroom",
    },
    {
        "building_type": "Educational",
        "room_type": "Office",
    },
    {
        "building_type": "Residential",
        "room_type": "Single-Family House",
    },
],
)
print(json_studies)
```

4.1 Available behavior types

Please only use the following names as input. e.g. Please use `Lighting_Status` (listed below) instead of `Lighting Adjustment`(displayed on the website).

```
'Plug_Load', 'Fan_Status', 'Door_Status', 'HVAC_Measurement', 'Lighting_Status',  
→ 'Occupant_Number', 'Occupancy_Measurement', 'Other_HeatWave', 'Other_Role of habits in_  
→ consumption', 'Other_IAQ in Affordable Housing', 'Shading_Status', 'Window_Status'
```

In the next version, the package will auto detect either type of input and convert to the correct query parameter.

Note: Study 2 is a special case. It has very large source files (> 2 GB) so we compressed all data in study 2 as a single `.tar.gz`file. In the example above, `data.zip` contains a `tar.gz`file along with several separate `csv` files from other studies. When writing libraries to read from `csv` file from the downloaded `zip`, Study 2 should be treated as a special case.

CHANGELOG

- 2021-11-18: Release 0.1.3
- 2021-11-19: Release 0.1.4, fixed a minor issue with Python 3.10.0
- 2021-11-23: Release 1.0.0
 - Breaking changes:
 - * Behavior type (query field) “Occupancy” has been renamed to “Occupancy_Measurement” to keep the name consistent. The example above has been changed accordingly. The server will reject query field “Occupancy”.
 - Added endpoint to check backend server health
 - Added endpoint to query available behavior types based on Study IDs
- 2021-12-01: Release 1.1.0
 - Added endpoint to query available studies based on (behaviors, countries, cities, (building type + room type))
- 2021-12-10:
 - Breaking change: renamed Appliance Usage to Plug Load on the server end. The example code has been changed accordingly.

API REFERENCE

6.1 API Reference

class obplatform.Connector(endpoint='https://api.ashraeobdatabase.com')

Bases: object

Connector to the remote database

Initialize the connector

Parameters **endpoint** (*str*) – The endpoint of the remote database, currently this should be “https://api.ashraeobdatabase.com”

__init__(endpoint='https://api.ashraeobdatabase.com')

Initialize the connector

Parameters **endpoint** (*str*) – The endpoint of the remote database, currently this should be “https://api.ashraeobdatabase.com”

list_behaviors()

Lists all behaviors available in the database

Return type List[Dict[str, Any]]

Returns

List of dicts showing all behaviors in the database. The “key” field is what users should use to query and download the data. The “label” field is what is displayed to users on the website

For example, {“label”: “Occupant Presence”, “key”: “Occupancy”, “disabled”: false} “Occupant Presence” is the behavior name shown on the website, “Occupancy” is what users should use in API and other functions in this module to query and download the data from the database.

download_export(filename, behavior_ids, studies, show_progress_bar=False, chunk_size=1024000)

Download the data archive (ZIP) for the given behaviors

Parameters

- **filename** (*str*) – The filename to save the archive
- **behavior_ids** (*List[int | str]*) – List of behavior ids to download
- **studies** (*List[int | str]*) – List of study ids to download
- **show_progress_bar** (*bool*) – Whether to show a progress bar
- **chunk_size** (*Optional[int]*) – The size of each chunk to download. If None, download the whole file. Default is 1000 * 1024 Bytes.

Return type None

check_health()

Check the health of the remote database

Return type bool

Returns True if the backend server is working, False otherwise

list_behaviors_in_studies(studies)

List available behaviors in each study

Parameters **studies** (*int* / *str*) – List of study ids to query

Return type List[Dict[str, Any]]

Returns JSON encoded result of study id and behaviors in the study

list_studies(behaviors, countries, cities, buildings)

Query available studies based on behaviors, countries, cities and buildings. This function works in the same way as clicking through the “Export” page on the website.

Parameters

- **behaviors** (*List[int / str]*) – List of behavior ids to query
- **countries** (*List[str]*) – List of country names to query
- **cities** (*List[str]*) – List of city names to query
- **buildings** (*List[Dict[str, Any]]*) – List of building types and room types to query. Must be in the following format:

```
[
  {
    "building_type": "Educational",
    "room_type": "Classroom",
  },
  {
    "building_type": "Educational",
    "room_type": "Office",
  },
  {
    "building_type": "Residential",
    "room_type": "Single-Family House",
  },
]
```

Return type List[Dict[str, Any]]

Returns JSON encoded result of study ids available in the database filtered by the criteria.

PYTHON MODULE INDEX

O

obplatform, [13](#)

Symbols

`__init__()` (*obplatform.Connector method*), 13

C

`check_health()` (*obplatform.Connector method*), 14

`Connector` (*class in obplatform*), 13

D

`download_export()` (*obplatform.Connector method*),
13

L

`list_behaviors()` (*obplatform.Connector method*), 13

`list_behaviors_in_studies()` (*obplatform.Connector method*), 14

`list_studies()` (*obplatform.Connector method*), 14

M

`module`
 obplatform, 13

O

`obplatform`
 module, 13