

---

## phue: A Python library for Philips Hue

Full featured Python library to control the Philips Hue lighting system.

### Features

- Compliant with the Philips Hue API 1.0
- Support for Lights
- Support for Groups
- Support for Schedules
- Support for Scenes
- Support for Sensors
- Compatible with Python 2.6.x and upwards
- Compatible with Python 3
- No dependencies
- Simple structure, single phue.py file
- Work in a procedural way or object oriented way

### Installation

#### Using distutils

```
1 sudo easy_install phue
```

or

```
1 pip install phue
```

#### Manually

phue consists of a single file (phue.py) that you can put in your python search path or in site-packages (or dist-packages depending on the platform) You can also simply run it by putting it in the same directory as you main script file or start a python interpreter in the same directory. phue works with Python 2.6.x, 2.7.x and 3.x

---

## Examples

### Basic usage

Using the `set_light` and `get_light` methods you can control pretty much all the parameters :

```
1  #!/usr/bin/python
2
3  from phue import Bridge
4
5  b = Bridge('ip_of_your_bridge')
6
7  # If the app is not registered and the button is not pressed, press the
   button and call connect() (this only needs to be run a single time)
8  b.connect()
9
10 # Get the bridge state (This returns the full dictionary that you can
   explore)
11 b.get_api()
12
13 # Prints if light 1 is on or not
14 b.get_light(1, 'on')
15
16 # Set brightness of lamp 1 to max
17 b.set_light(1, 'bri', 254)
18
19 # Set brightness of lamp 2 to 50%
20 b.set_light(2, 'bri', 127)
21
22 # Turn lamp 2 on
23 b.set_light(2, 'on', True)
24
25 # You can also control multiple lamps by sending a list as lamp_id
26 b.set_light([1,2], 'on', True)
27
28 # Get the name of a lamp
29 b.get_light(1, 'name')
30
31 # You can also use light names instead of the id
32 b.get_light('Kitchen')
33 b.set_light('Kitchen', 'bri', 254)
34
35 # Also works with lists
36 b.set_light(['Bathroom', 'Garage'], 'on', False)
37
38 # The set_light method can also take a dictionary as the second
   argument to do more fancy stuff
39 # This will turn light 1 on with a transition time of 30 seconds
40 command = {'transitiontime' : 300, 'on' : True, 'bri' : 254}
41 b.set_light(1, command)
```

---

## Light Objects

If you want to work in a more object-oriented way, there are several ways you can get Light objects.

### Get a flat list of light objects

```
1
2 lights = b.lights
3
4 # Print light names
5 for l in lights:
6     print(l.name)
7
8 # Set brightness of each light to 127
9 for l in lights:
10     l.brightness = 127
```

### Get Light objects as dictionaries

```
1 # Get a dictionary with the light id as the key
2 lights = b.get_light_objects('id')
3
4 # Get the name of light 1, set the brightness to 127
5 lights[1].name
6 lights[1].brightness = 127
7
8 # Get a dictionary with the light name as the key
9 light_names = b.get_light_objects('name')
10
11 # Set the brightness of the bulb named "Kitchen"
12 light_names["Kitchen"].brightness = 254
13
14 # Set lights using name as key
15 for light in ['Kitchen', 'Bedroom', 'Garage']:
16     light_names[light].on = True
17     light_names[light].hue = 15000
18     light_names[light].saturation = 120
19
20 # Get a flat list of the light objects (same as calling b.lights)
21 lights_list = b.get_light_objects('list')
22
23 for light in lights_list:
24     light.on = True
25     light.brightness = 127
```

## Setting Transition Times

In the Hue API, transition times are specified in deciseconds (tenths of a second). This is not tracked as a device setting, but rather needs to be applied on each individual transition command you want

---

to control the time of.

This can be done by specifying a `transitiontime` keyword when calling `set_light` on the bridge:

```
1 # Set brightness of lamp 1 to max, rapidly
2 b.set_light(1, 'bri', 254, transitiontime=1)
```

As a convenience, the `Light` class implements a wrapper that remembers a specified transition time for that light, and applies it automatically to every transition:

```
1 light = light_names['Kitchen']
2 light.transitiontime = 2
3 # this next transition will happen rapidly
4 light.brightness = 20
```

Note that there is a known bug where turning a light off with the `transitiontime` specified can cause the brightness level to behave erratically when the light is turned back on. See this discussion. This package attempts to work around this issue by automatically resetting the brightness when necessary, but this may not work in all cases.

Transition times from 0-300 deciseconds (i.e. 0 - 30 seconds) have been tested to work.

## Groups

You can also work with the groups functionality of the Bridge. If groups aren't working, try re-setting the bridge by unplugging it and plugging it back again.

```
1
2 # List groups
3 b.get_group()
4
5 # List group 1
6 b.get_group(1)
7
8 # Get name of group 1
9 b.get_group(1, 'name')
10
11 # Get lights in group 1
12 b.get_group(1, 'lights')
13
14 # Create a group with lights 1 and 3
15 b.create_group('Kitchen', [1,3])
16
17 # Rename group with id 1
18 b.set_group(1, 'name', 'New Group Name')
19
20 # Change lights within group 1
21 b.set_group(1, 'lights', [3,4])
```

---

```
22
23 # Turn group 1 off
24 b.set_group(1, 'on', False)
25
26 # Delete group 2
27 b.delete_group(1)
```

## Schedules

You can view, create and delete schedules using the following methods. Note that updates to the Hue API now use local time instead of UTC. If you have issues with schedules not triggering correctly, double check that the time zone is set correctly on your Hue Bridge and that your time in your code is not in UTC by default.

```
1
2 # Get the list of different schedules
3 b.get_schedule()
4
5 # Get the data of a particular schedules
6 b.get_schedule(1)
7
8 # Create a schedule for a light, arguments are name, time, light_id,
9   data (as a dictionary) and optional description
10 data = {'on': False, 'transitiontime': 600}
11 b.create_schedule('My schedule', '2012-11-12T22:34:00', 1, data, '
12   Bedtime' )
13
14 # Create a schedule for a group, same as above but with a group_id
15   instead of light_id
16 data = {'on': False, 'transitiontime': 600}
17 b.create_group_schedule('My schedule', '2012-11-12T22:34:00', 0, data,
18   'Bedtime' )
19
20 # Delete a schedule
21 b.delete_schedule(1)
```

## Using phue with Max/MSP via Jython

You can use the phue library within Max/MSP by using Nick Rothwell's Jython objects. He recently updated the version to support Jython 2.7 which is required for phue to work.

Download it here: <https://github.com/cassiel/net.loadbang.jython>

---

## **Using phue on iOS via Pythonista**

You can use phue on your iOS device via the Pythonista app. This is a great way to build quick prototypes on iOS as you don't need to compile anything, you can code directly from the device itself.

See this little example:

<http://www.youtube.com/embed/6K-fxWG6JSs>

## **Acknowledgments**

Huge thanks to <http://rsmck.co.uk/hue> for hacking the protocol !

## **License**

MIT - <http://opensource.org/licenses/MIT>

“Hue Personal Wireless Lighting” is a trademark owned by Koninklijke Philips Electronics N.V., see [www.meethue.com](http://www.meethue.com) for more information. I am in no way affiliated with the Philips organization.