

Phillips hue

Set it up

Step 1

Connect your bridge to the network

Step 2

Find the IP address of the bridge

<https://discovery.meethue.com/>

Step 3

Once you have the address load the test app by visiting the follow address in your web browser

**https://<bridge ip
address>/debug/clip.html**

The interface looks like this:

URL: is the local address of a specific resource (thing) inside the hue system. It could be light, a group of lights or many more things. This is the object you'll be interacting with in this command.

Body: is the part of the message which describes what you want to change and how. Here you enter, in JSON format, the resource name and value you'd like to change/add.

method: here you have a choice of the 4 HTTPS methods the hue call can use.

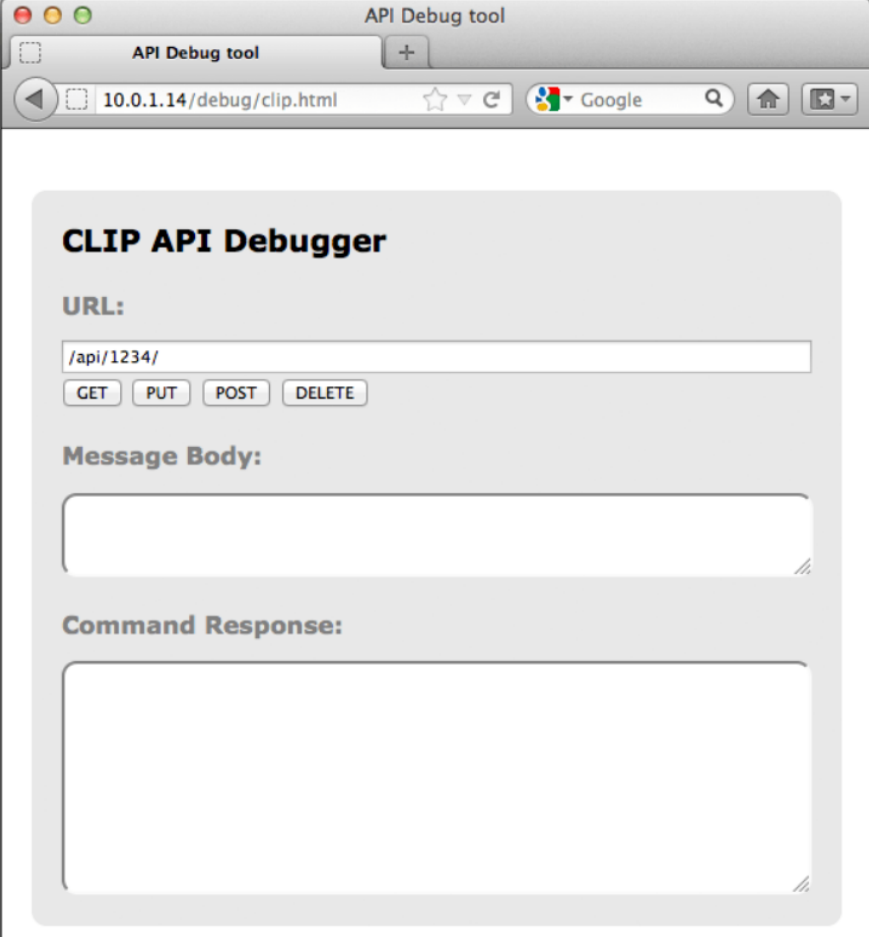
GET: this is the command to fetch all information about the addressed resource

PUT: this is the command to modify an addressed resource

POST: this is the command to create a new resource inside the addressed resource

DELETE: this is the command to delete the addressed resource

Response: In this area you'll see the response to your command. Also, in JSON format.



The screenshot shows a web browser window titled "API Debug tool" with a tab for "API Debug tool". The address bar shows "10.0.1.14/debug/clip.html". The main content area is titled "CLIP API Debugger" and contains the following fields and buttons:

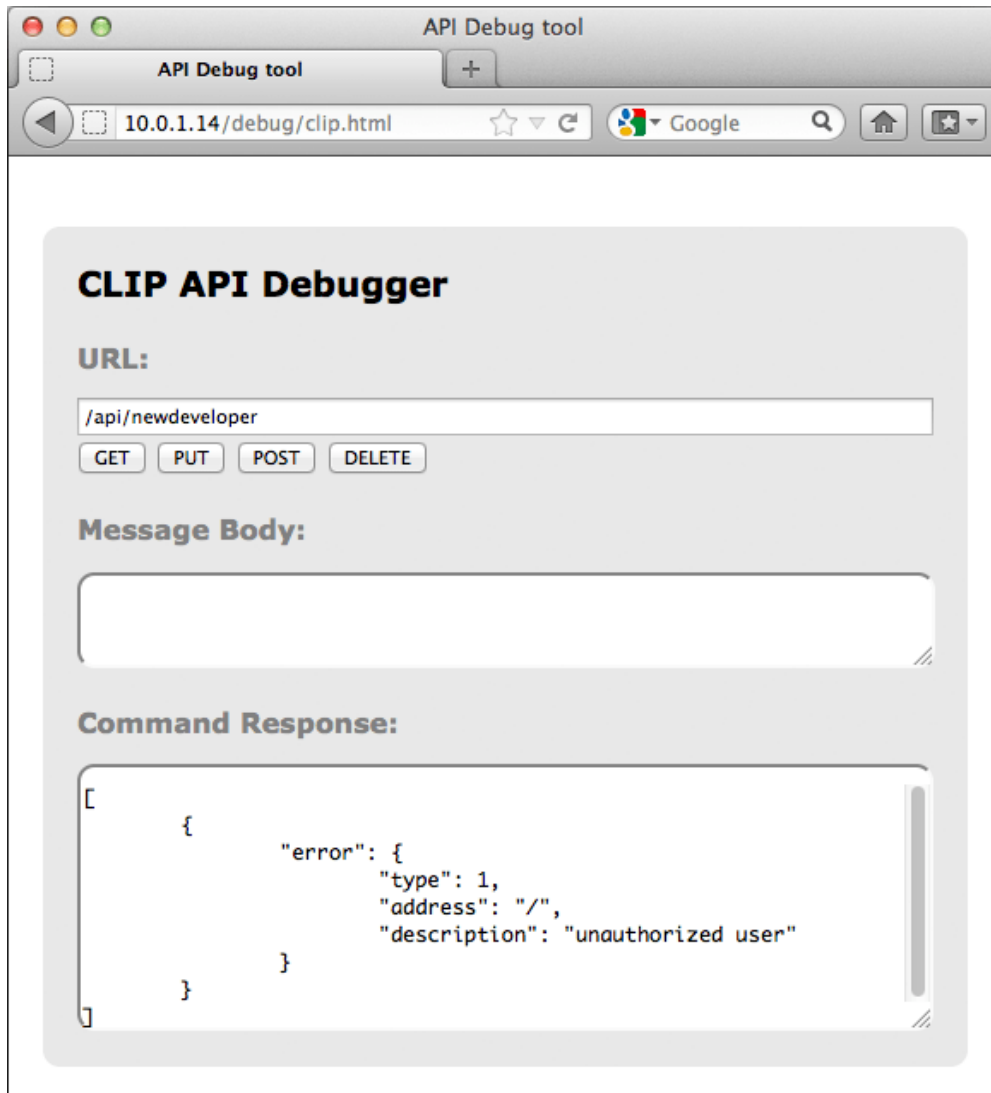
- URL:** A text input field containing "/api/1234/".
- Method:** Four buttons labeled "GET", "PUT", "POST", and "DELETE".
- Message Body:** A large text area for entering JSON data.
- Command Response:** A large text area for displaying the response.

Add a user

Change the information:

URL: **/api/newdeveloper**

Method: **GET**



This is the command to fetch all information in the bridge. You didn't get much back and that's because you're using an unauthorized username "newdeveloper".

We need to use the randomly generated username that the bridge creates for you.

Change the information:

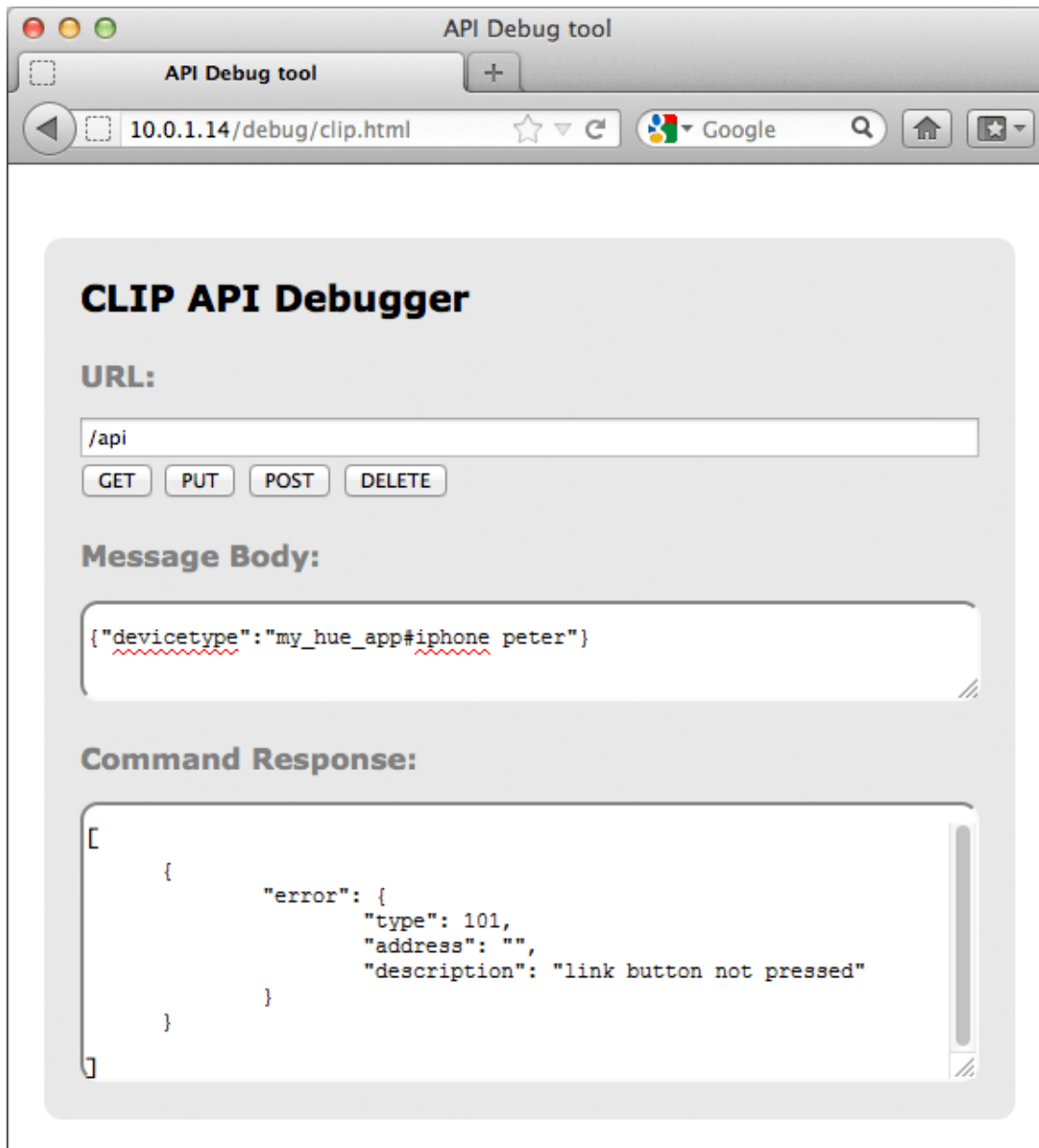
URL: **/api**

Body: **{ "devicetype": "my_hue_app#iphone peter" }**

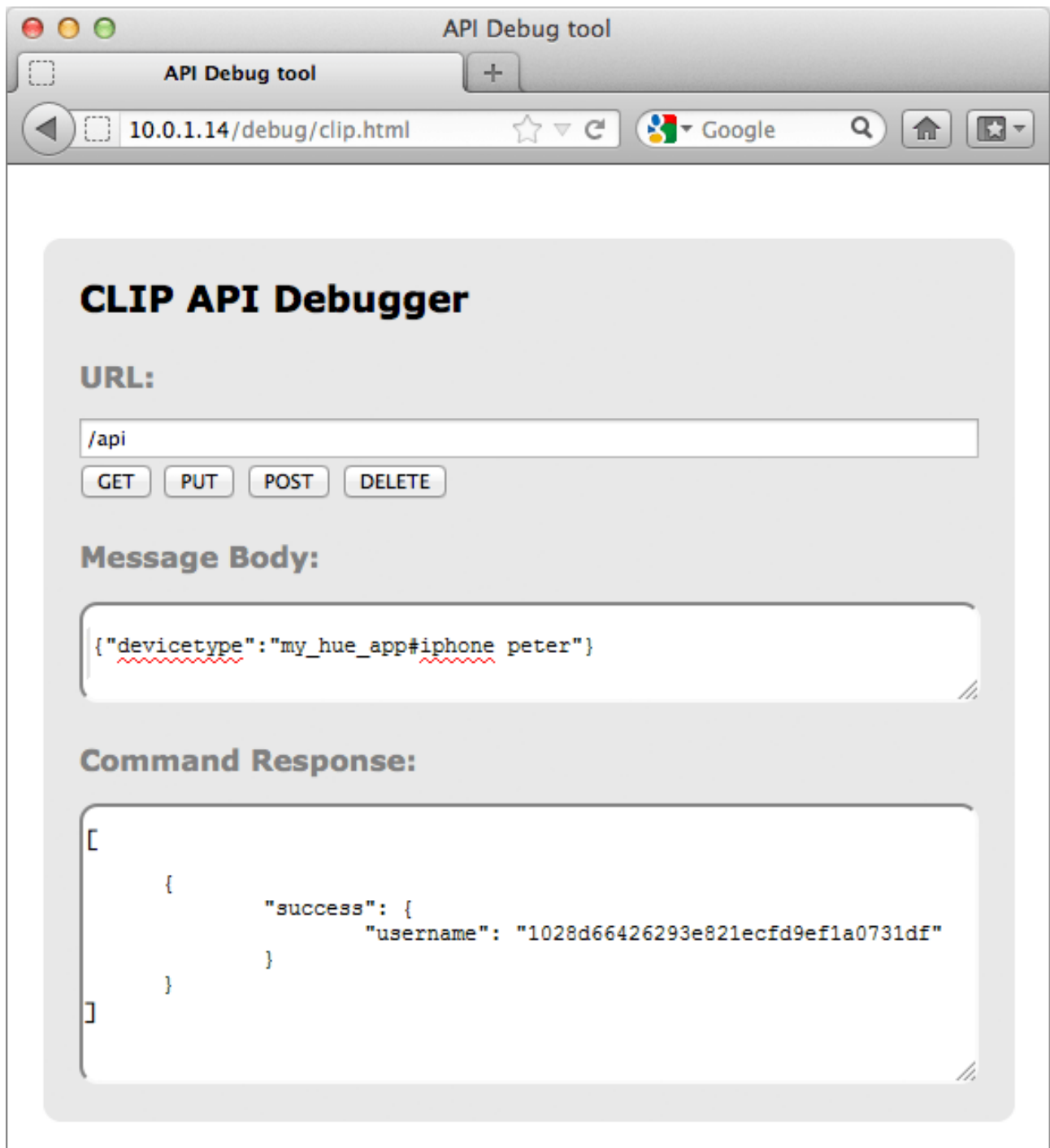
Method: **POST**

This command creates a new resource inside /api (where usernames sit) with the following properties.

When you press the POST button you should get back an error message letting you know that you have to press the link button. This is our security step so that only apps you want to control your lights can. By pressing the button, we prove that the user has physical access to the bridge.



Go and press the button on the bridge and then press the POST button again and you should get a success response like below.



Congratulations you've just created an authorized user (**1028d66426293e821ecfd9ef1a0731df**), which we'll use from now on!

Turning a light on and off

Change the information:

URL: **https://<bridge ip address>/api/1028d66426293e821ecfd9ef1a0731df/lights**

Method: **GET**

You should get a JSON response with all the lights in your system and their names.

URL: **https://<bridge ip address>/api/1028d66426293e821ecfd9ef1a0731df/lights/1**

Method: **GET**

In this response you can see all of the resources this light has. The most interesting ones are inside the state object as these are the ones, we'll have to interact with to control the light

URL: **https://<bridge ip address>/api/1028d66426293e821ecfd9ef1a0731df/lights/1/state**

Body: **{"on": "false"}**

Method: **PUT**

Looking at the command you are sending we're addressing the "state" object of light one and telling it to modify the "on" value inside it to false (or off). When you press the PUT button the light should turn off. Change the value in the body to true and the light will turn on again.

URL: **https://<bridge ip address>/api/1028d66426293e821ecfd9ef1a0731df/lights/1/state**

Body: **{"on": true, "sat": 254, "bri": 254, "hue": 10000}**

Method: **PUT**

We're interacting with the same "state" attributes here but now we're modifying a couple more attributes. We're making sure the light is on by setting the "on" resource to true. We're also making sure the saturation (intensity) of the colors and the brightness is at its maximum by setting the "sat" and "bri" resources to 254. Finally, we're telling the system to set the "hue" (a measure of color) to 10000 points (hue runs from 0 to 65535). Try changing the hue value and keep pressing the PUT button and see the color of your light changing running through different colors

Assignment

Step 1

Go in groups of 4
Choose a hue box
Find the IP address

Step 2

Check if there is a connect to hue
Make a connect to the API 'OpenWeather'

Step 3

Program the lights based on the weather from the API
Choose a city
Find the data weather from that city
Change the light based on the weather
Update the light every 3 min for the city, so the color will be updated every time the weather change
Choose a color, if you are meet by a weather, that the program doesn't know