NEW MEDIA &
COMMUNICATION
TECHNOLOGY

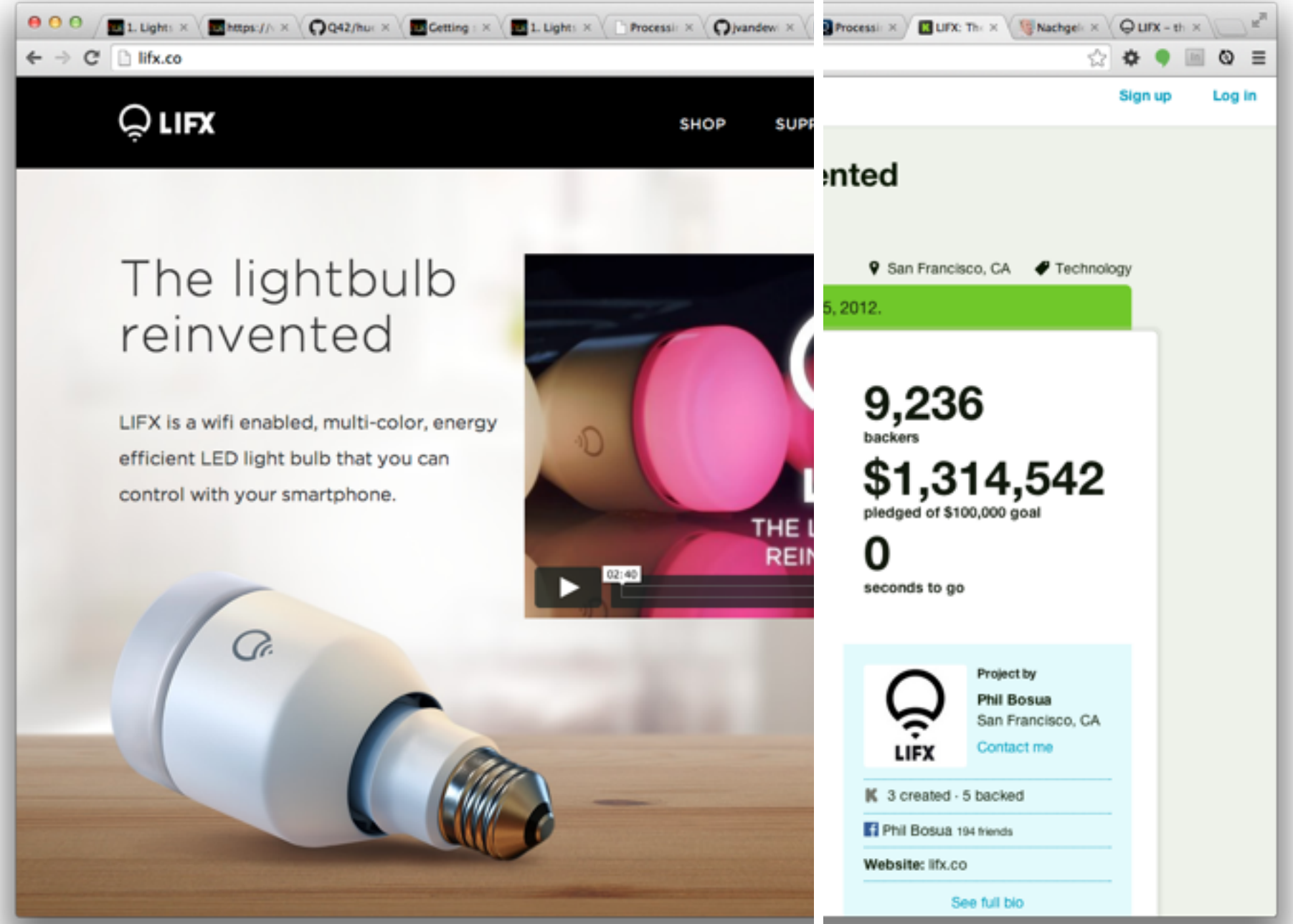
Yes, HUE can!

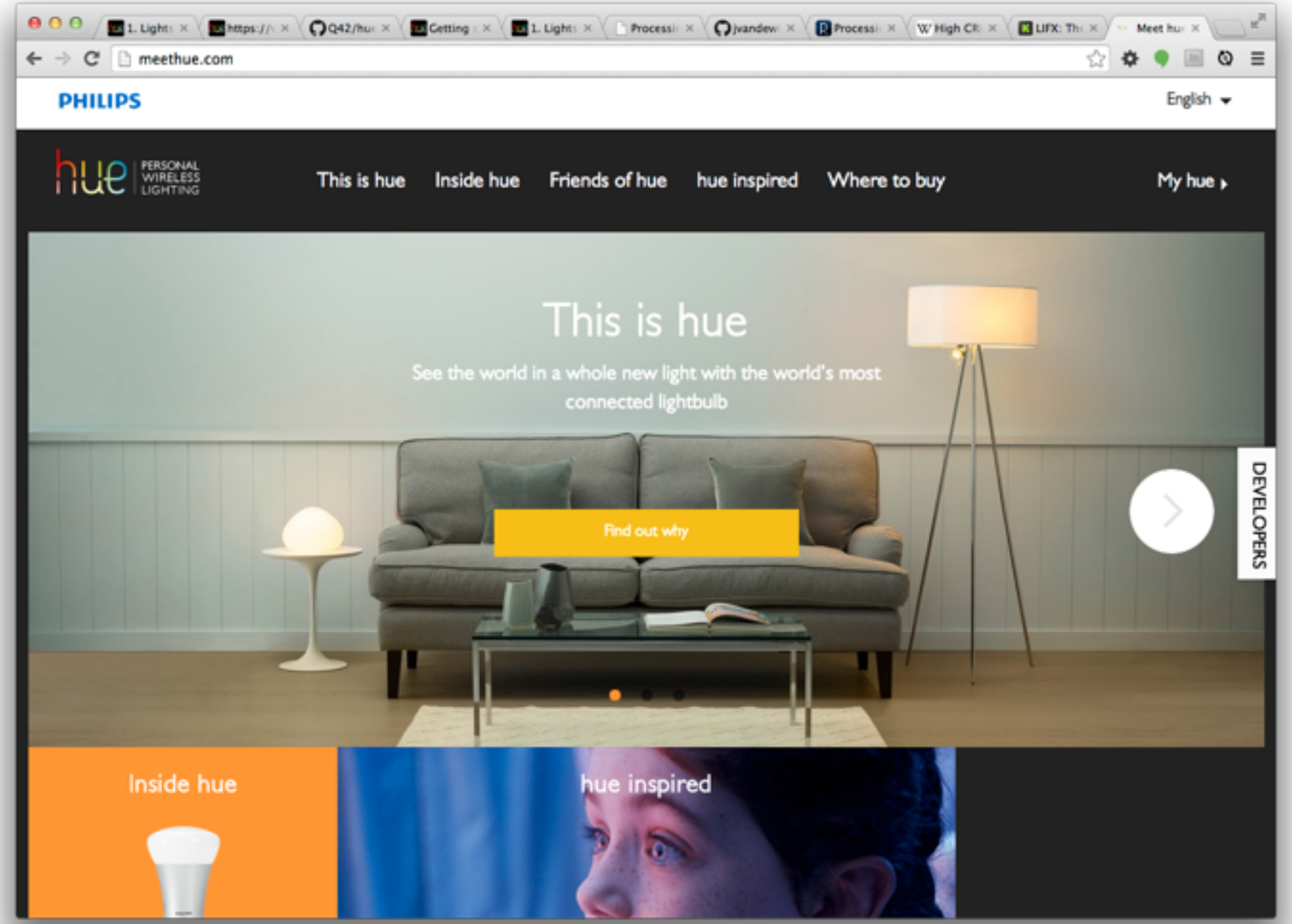
Philips Hue, LifeX, …

A little history

10/2012 Kickstarter project: LifeX
11/2012 Philips Hue
http://blog.saikoled.com/post/47025049702/lifx-versus-the-hue

Alternatieven duiken op: AppLight / IWY / Wemo Plugs (stopcontacten & varia)


lifx.co
LIFX
SHOP
The lightbulb reinvented
LIFX is a wifi enabled, multi-color, energy efficient LED light bulb that you can control with your smartphone.
02:40
Sign up
Log in
San Francisco, CA
Technology
9,236
backers
$1,314,542
pledged of $100,000 goal
0
seconds to go
Project by
Phil Bosua
San Francisco, CA
Contact me
3 created · 5 backed
Phil Bosua 194 friends
Website: lifx.co
See full bio

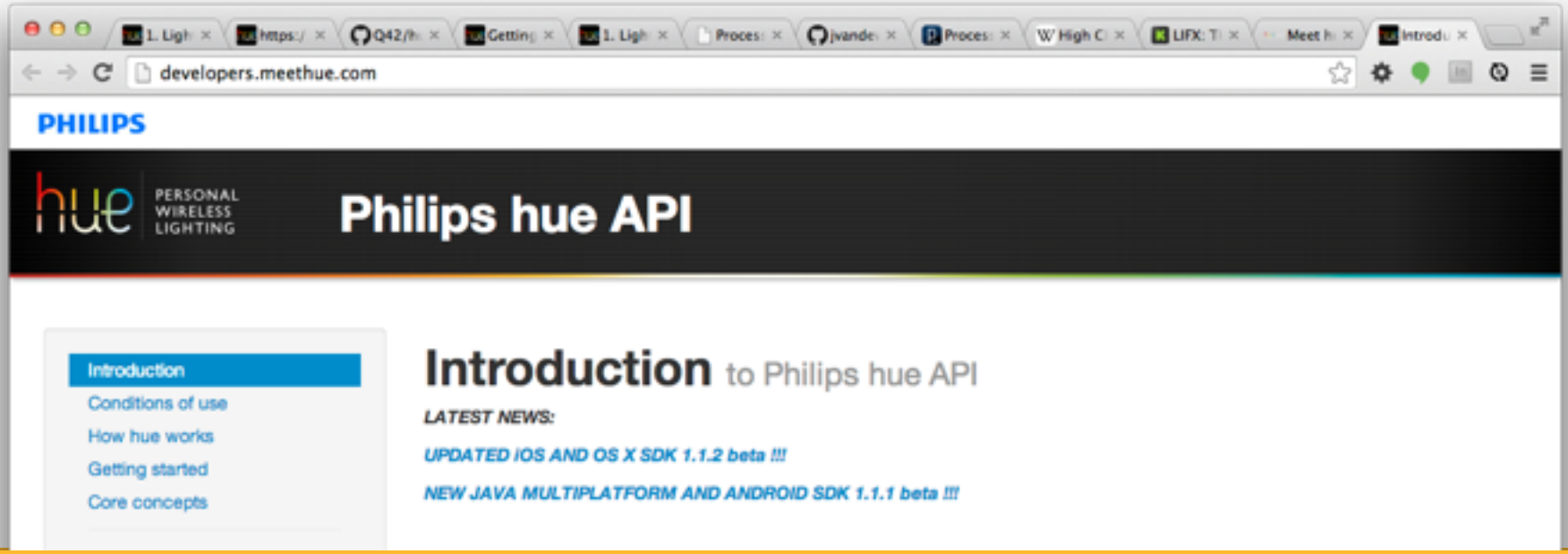

meethue.com
PHILIPS
English
hue
PERSONAL WIRELESS LIGHTING
This is hue
Inside hue
Friends of hue
hue inspired
Where to buy
My hue
This is hue
See the world in a whole new light with the world's most connected lightbulb
Find out why
DEVELOPERS
Inside hue
hue inspired

Philips Hue

Uitgebreide developer mogelijkheden:

Documentatie
http://developers.meethue.com/

http://www.developers.meethue.com/documentation/core-concepts



Stappen

1. Bridge connecteren met de router

2. IP-adress van de Philips Hue router zoeken via UPnP discovery:
   https://www.meethue.com/api/nupnp
   Werkt niet in schoolomgeving.

3. verbinding maken met de bridge: http //<bridge ip address>/debug/clip.html

New developer

1. Een nieuwe, unieke developer (token) aanmaken:

```
{"devicetype":"test-
user","username":"bart"}
```

! minimum 10 karakters in de username
! geen speciale tekens

POST

2. Verbinding maken met de Bridge door op de link button te duwen (op de router)

3. Opnieuw POST klikken

Lampen opvragen

1. Het aantal lichten opvragen die met de Bridge verbonden zijn:

URL: `http://172.23.251.3/api/eigengebruikersnaam/lights/`

GET

2. Een specifiek licht opvragen:

URL: `http://172.23.251.3/api/eigengebruikersnaam/lights/1`

GET

Hue: result

Alle gegevens die je kan uitlezen van een lamp:


```
{
    "state": {
            "on": true,
            "bri": 200,
            "hue": 26200,
            "sat": 200,
            "xy": [
                    0.2607,
                    0.6376
            ],
            "alert": "none",
            "effect": "none",
            "colormode": "hs",
            "reachable": false
    },
    "type": "Color light",
    "name": "LightStrips 1",
    "modelid": "LST001",
    "swversion": "66010400",
    "pointsymbol": {
            "1": "none",
            "2": "none",
            "3": "none",
            "4": "none",
            "5": "none",
            "6": "none",
            "7": "none",
            "8": "none"
    }
}
```

Hue: lampen aan/uit zetten

1. Het aantal lichten opvragen die met de Bridge verbonden zijn:

URL: `http://172.23.251.3/api/eigengebruikersnaam/lights/1/state/`

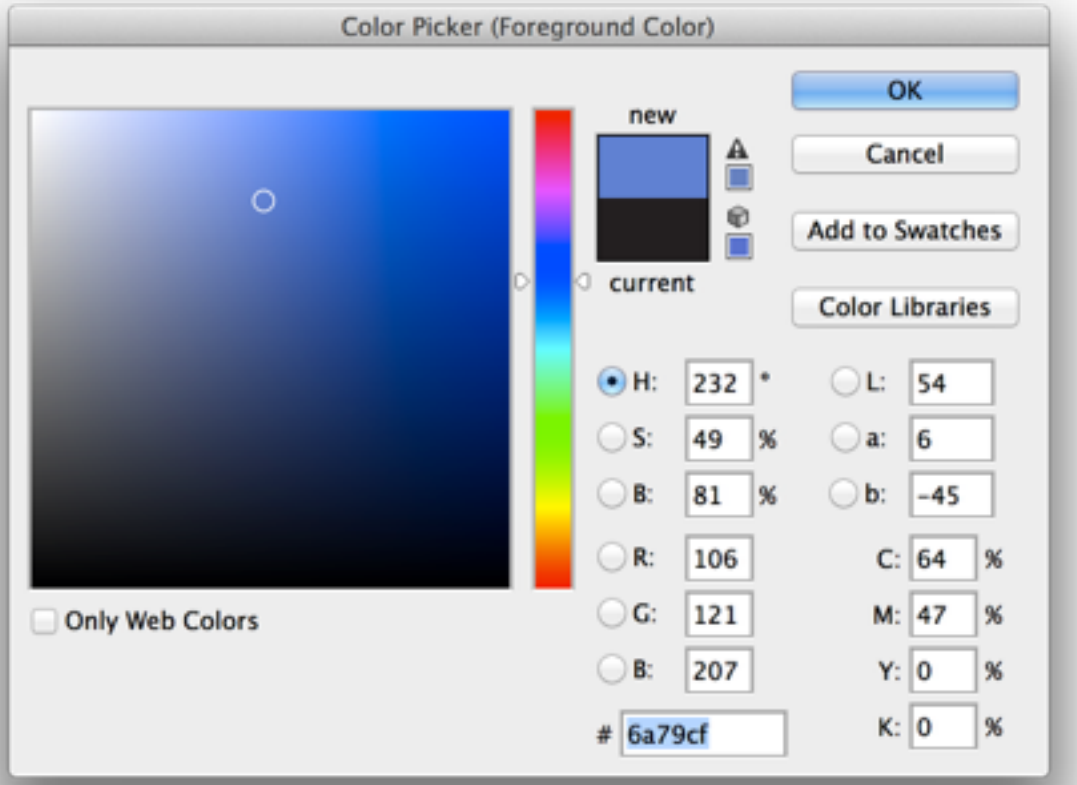
Body: `{"on":false}`

PUT

Hue, Saturation, Brightness

Philips Hue API
maximum waarden:

Hue: 65535
Saturation: 255
Brightness: 255



Kleuren

http://www.developers.meethue.com/documentation/core-concepts

Hue	Final-x	Final-y
0	0.3972	0.4564
12750	0.5425	0.4196
25500	0.41	0.51721
46920	0.1691	0.0441
56100	0.4149	0.1776
65280	0.6679	0.3181

Hue: HSB (kleur) instellen

1. Het aantal lichten opvragen die met de Bridge verbonden zijn:

URL: `http://172.23.251.3/api/eigengebruikersnaam/lights/1/state/`

Body: `{"on":true, "sat":255, "bri":255, "hue":65000}`

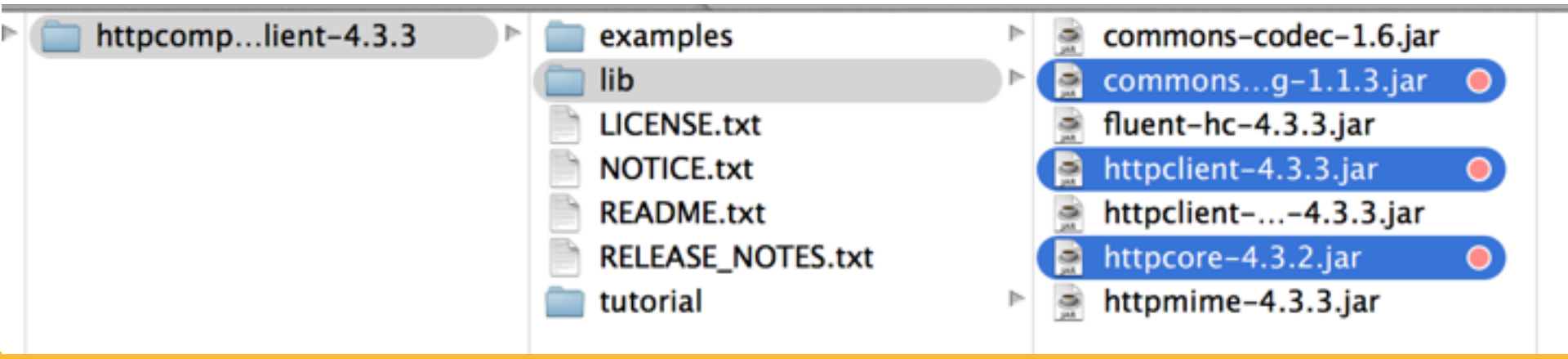
PUT

Processing

Om vanuit Processing HTTP, GET, POST, … requests te doen maken we gebruik van de Apache HttpComponents library

http://hc.apache.org/httpcomponents-client-ga/
http://hc.apache.org/
https://hc.apache.org/downloads.cgi

Importeren in je Processing sketch:
httpClient.jar, *httpcore.jar, commons-logging.jar* in je Processing applicatie slepen, of via Sketch > Add File



httpComponents

Een nieuwe instantie aanmaken
Een connectie interface opzetten

```
HttpClient httpClient;

String IP = "172.23.251.3";
String dev = "klausdelanghe";

void setup() {
  httpClient = HttpClientBuilder.create().build();
  setColor(1, true, 20000, 255, 255, 2);
}

void setColor(int lamp, boolean on, int hue, int sat, int bri, int trans) {
  try {
    StringEntity se = new StringEntity("{\"on\":"+on+", \"hue\":"+hue+", \"bri\"
    HttpPut httpPut = new HttpPut("http://"+IP+"/api/"+dev+"/lights/"+lamp+"/sta
    httpPut.setEntity(se);
```

httpComponents

se: De parameters - in dit geval via PUT - die moeten doorgestuurd worden
httpPut: Nieuwe connectie (url) aanmaken
response: Resultaat
entity: Als de transfer terug is vrijgegeven kan deze voor een volgende (her)gebruikt worden

```
try {
  StringEntity se = new StringEntity("{\"on\":"+on+", \"hue\":"+hue+", \"bri\
  HttpPut httpPut = new HttpPut("http://"+IP+"/api/"+dev+"/lights/"+lamp+"/st
  httpPut.setEntity(se);

  HttpResponse response = httpClient.execute(httpPut);

  HttpEntity entity = response.getEntity();

  if (entity != null) entity.consumeContent();
}
catch(Exception e) {
  e.printStackTrace();
}
}
```

httpComponents

stop: Bij het eindigen van de applicatie / sketch de connectie uit het (werk)geheugen verwijderen

```
void stop() {
  httpClient.getConnectionManager().shutdown();
  super.stop();
}
```