

# From Code to Community

Uli Hitzel  
PythonWebConf21  
March 2021



```
route('/keys/<id>', method='PUT')
    if not _check_key(id):
        return dict({"info": "Key does not exist already?"})
    redis_key = redis.get(redis_key)
    if redis_key is None:
        response.status = 400
        return dict({"info": "Key name is invalid."})

    # Get User ID and quota
    user_id = api_auth["id"]
    user_quota = _get_user_quota(user_id)
    current_number_of_resources = _user_resources_number(user_id)

    # Does the key have a valid format?
    if not _valid_identifier(str(id)) != True:
        response.status = 400
        return dict({"info": "Key identifier is not valid."})

    # Are we allowed to create more objects?
    if user_quota != 0 and current_number_of_resources > user_quota:
        response.status = 400
        return dict({"info": "Quota exceeded."})

    # Increase the counter of resources for this user
    rc.incr("NUMRES:"+str(user_id))
```

# How do you run an Open Source Project?



1. Take some Source Code
2. Dump in on GitHub
3. Done!

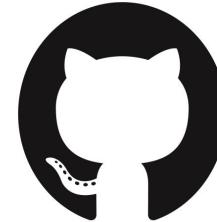




# Architect



/in/uhitzel



/u1i/slides

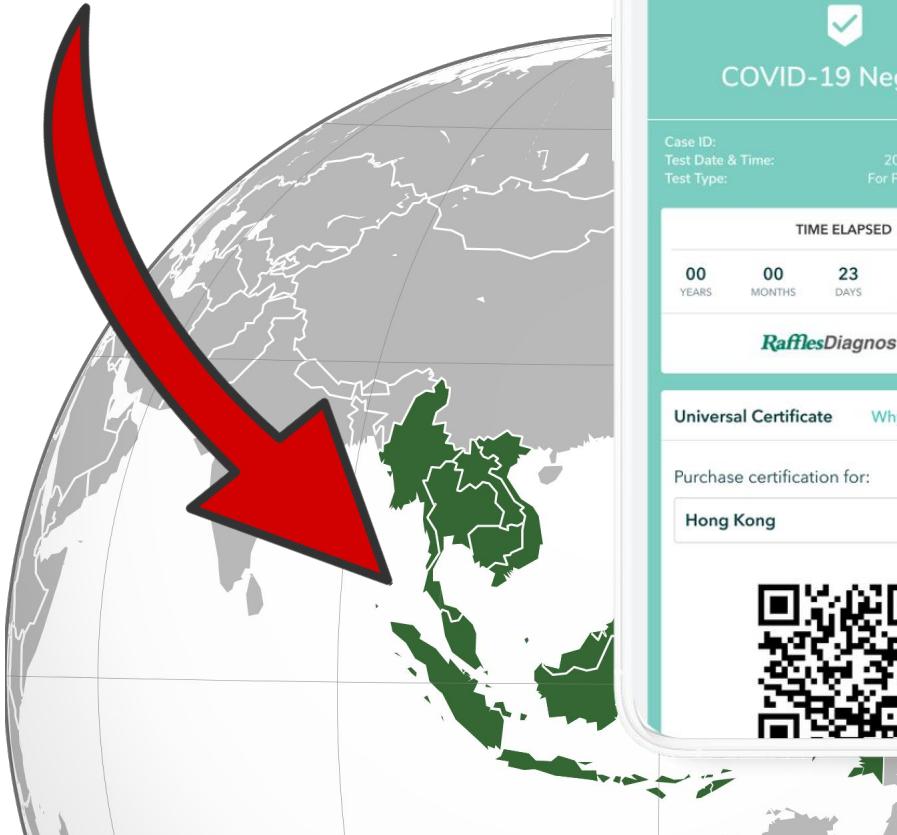


IBM



Y!





# Agenda

1 – Open Source &  
How To Run Projects

2 – Bambleweeny

3 – Get involved!



release v0.31 Docker Pulls 1451 license MIT issues 4 open

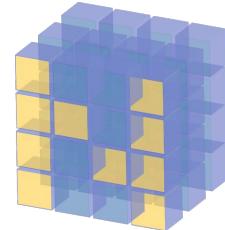
GET	/keys/{id}	Read Key	🔒
PUT	/keys/{id}	Write Key	🔒
DELETE	/keys/{id}	Delete Key	🔒
GET	/keys/	List Keys	🔒
GET	/incr/{id}	Increase Key	🔒



TensorFlow



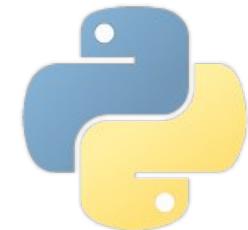
Scrapy



NumPy



Requests  
http for humans



# Open Source



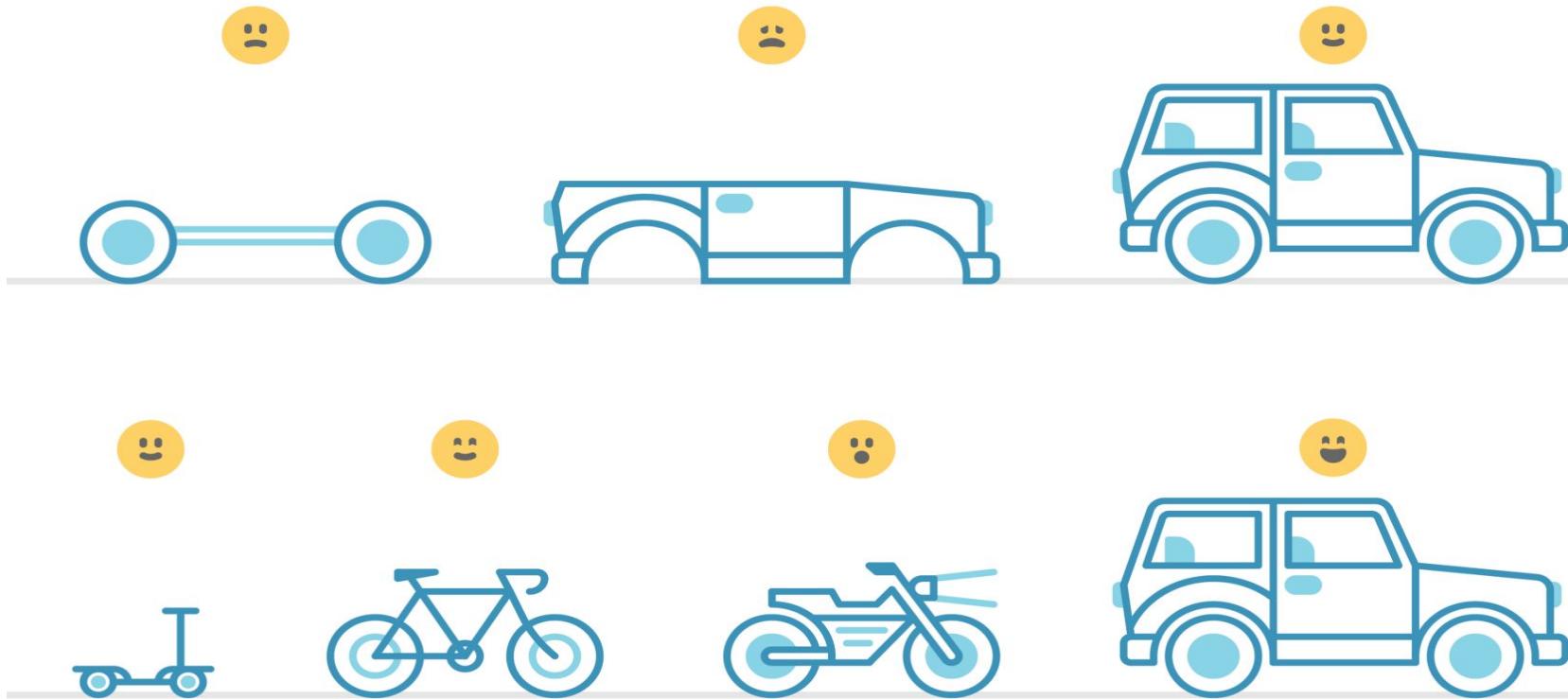
Community  
powered  
Innovation

# Open Source

People will **see** my  
Code

People will **use** my  
Stuff







# Contribute

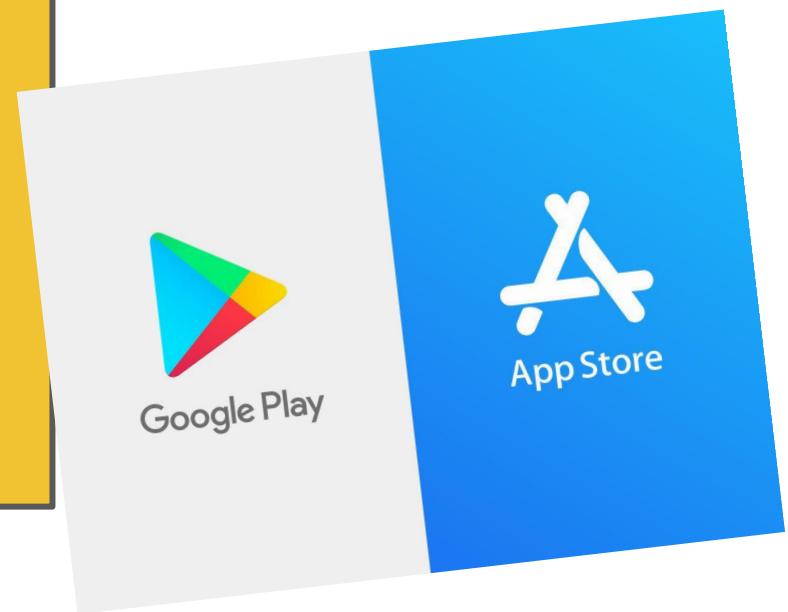
- Code
- Documentation & Translation
- Using it
  - Opening Tickets
  - Spreading the Word

# Why do it?

- You get inspired & grow
- Make new connections
- Give back to the  
Community
- It's rewarding & fun



# You're on the 'App Store'!



# Your RedisConf19 session "Bambleweeny: Redis with HTTP & OAuth" has been accepted!

ⓘ You replied on Mon 2/25/2019 18:06

DN

Dave Nielsen <dave@redislabs.com>

Sat 2/23/2019 01:55



Uli Hitzel ▾

Dear Uli,

Thank you for submitting your talk to RedisConf19.

On behalf of the Speaker Selection Committee, I'm delighted to let you know that your talk has been accepted and we'd be honored to have you speak at RedisConf.



# How do you run an Open Source Project?



# You'll have to be:

A Developer  
+ Marketing Person  
+ Project Manager



# I'm probably here



# Solve a real Problem

1



Build something  
you're using  
yourself

# Turn it into a 'Living Thing'

- GitHub
- Markdown
- Issues, Versions, Releases
- Contributions & Pull Requests

2





# Market it!

3

- Branding: Naming, Logo
- Superb Documentation
- Great 'First 20 Minutes' Experience
- Promote it

majime



/u1i



## bambleweeny

Bambleweeny is lightweight key-value store and message broker based on HTTP/REST.



Python



19



6

python

2.7

release

v0.36

Docker Pulls

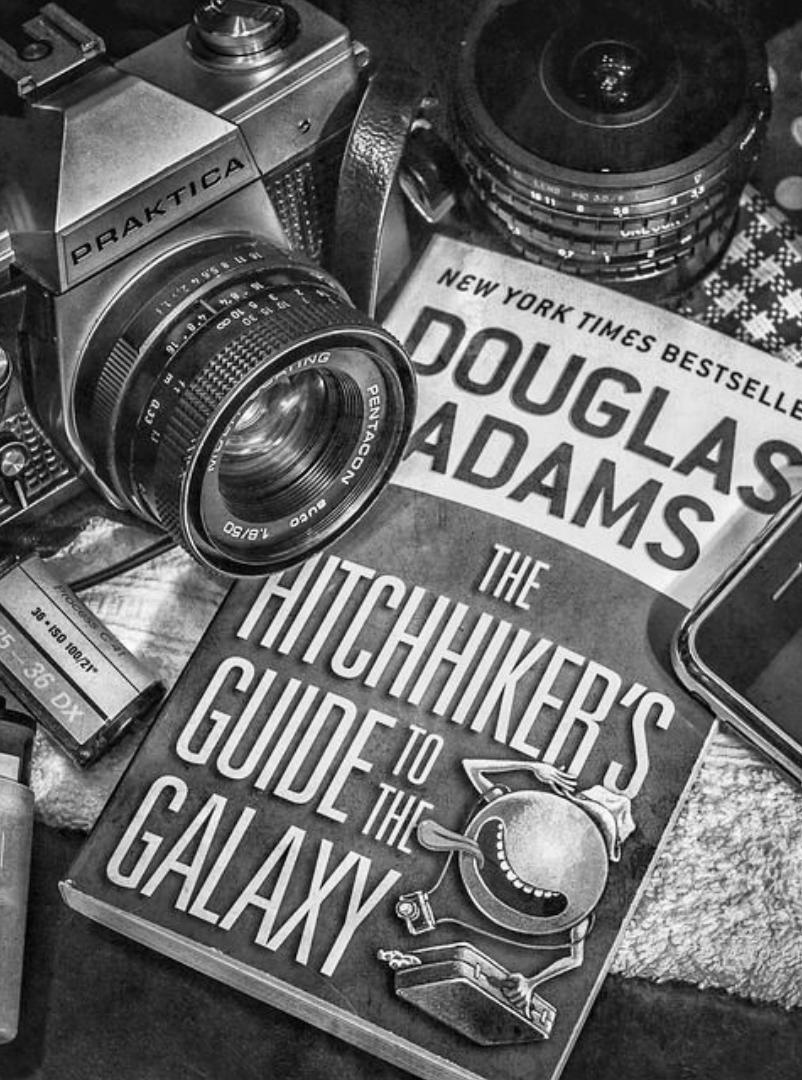
2497

issues

5 open



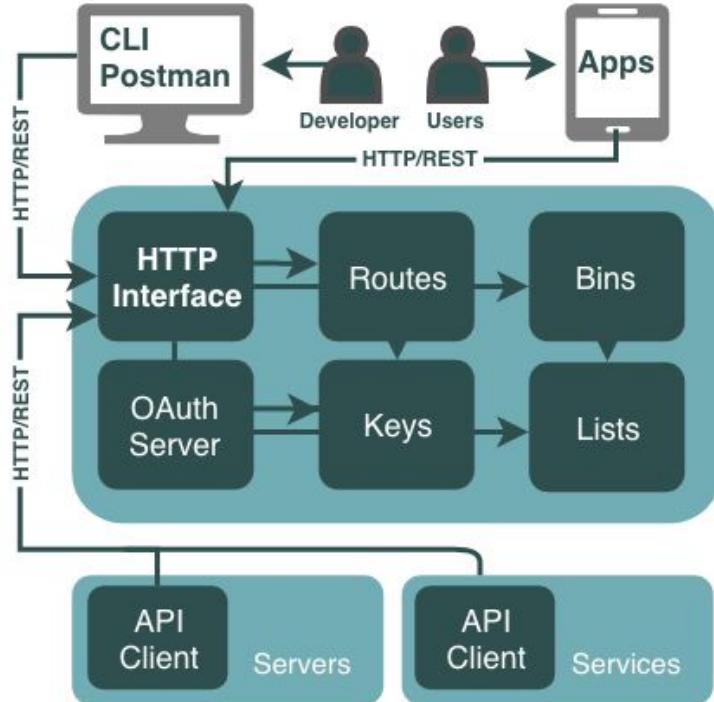
bamble  
weeny



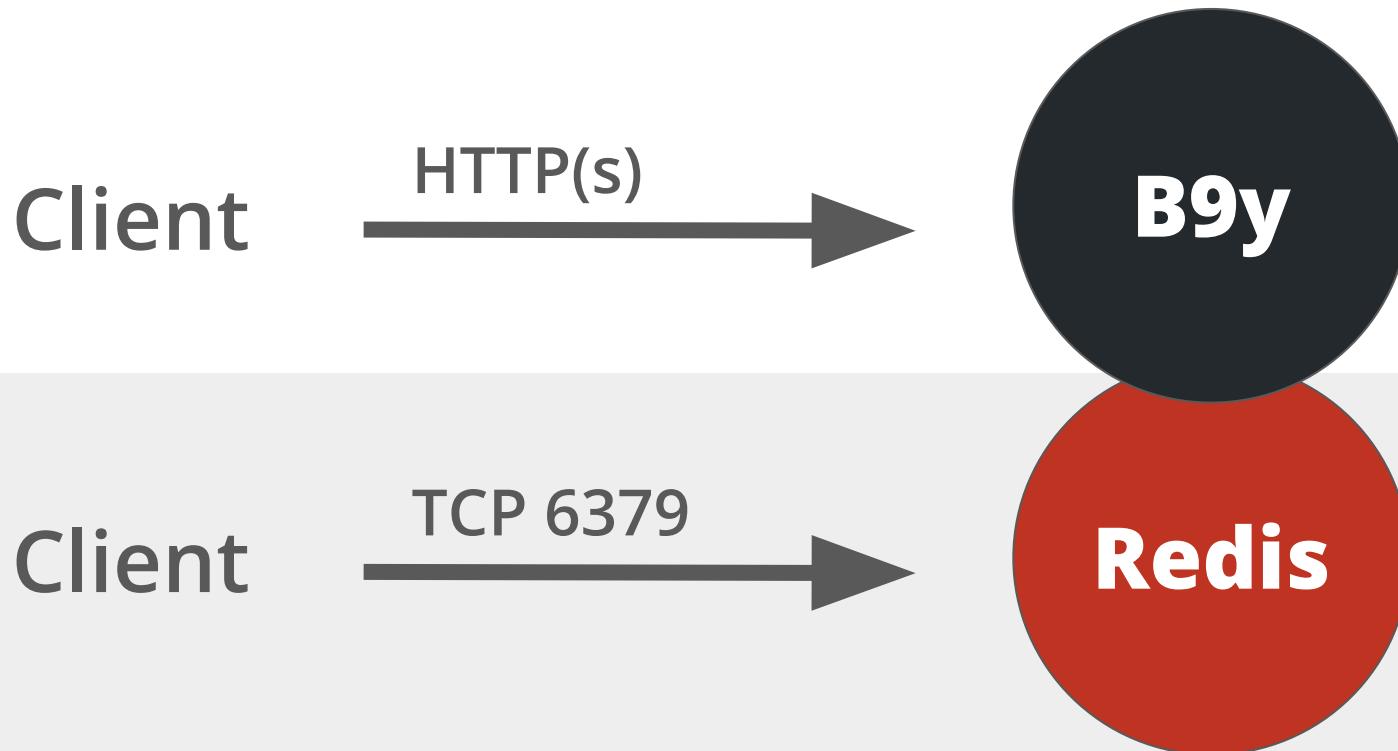
The Bambleweeny  
57 Submeson  
Brain is part of the  
Finite  
Improbability  
Generator.

# Bambleweeny

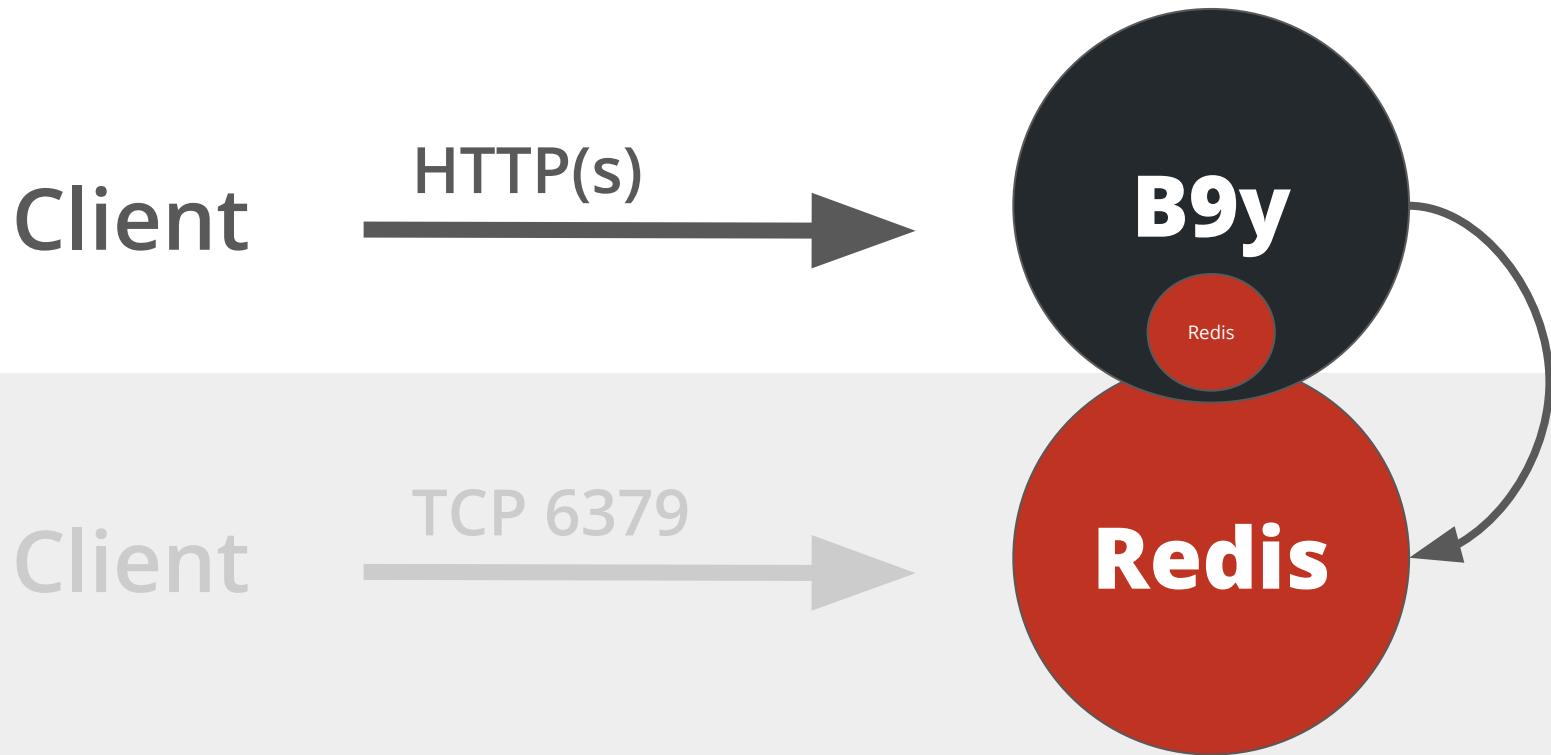
- Lightweight HTTP/REST based Key-Value Store & Message Broker
- Multi-Tenancy → Manage Identities, Access & Quotas
- Fast, Easy to Use & Well-Documented
- Written in Python, using Redis, deployable in a tiny Container

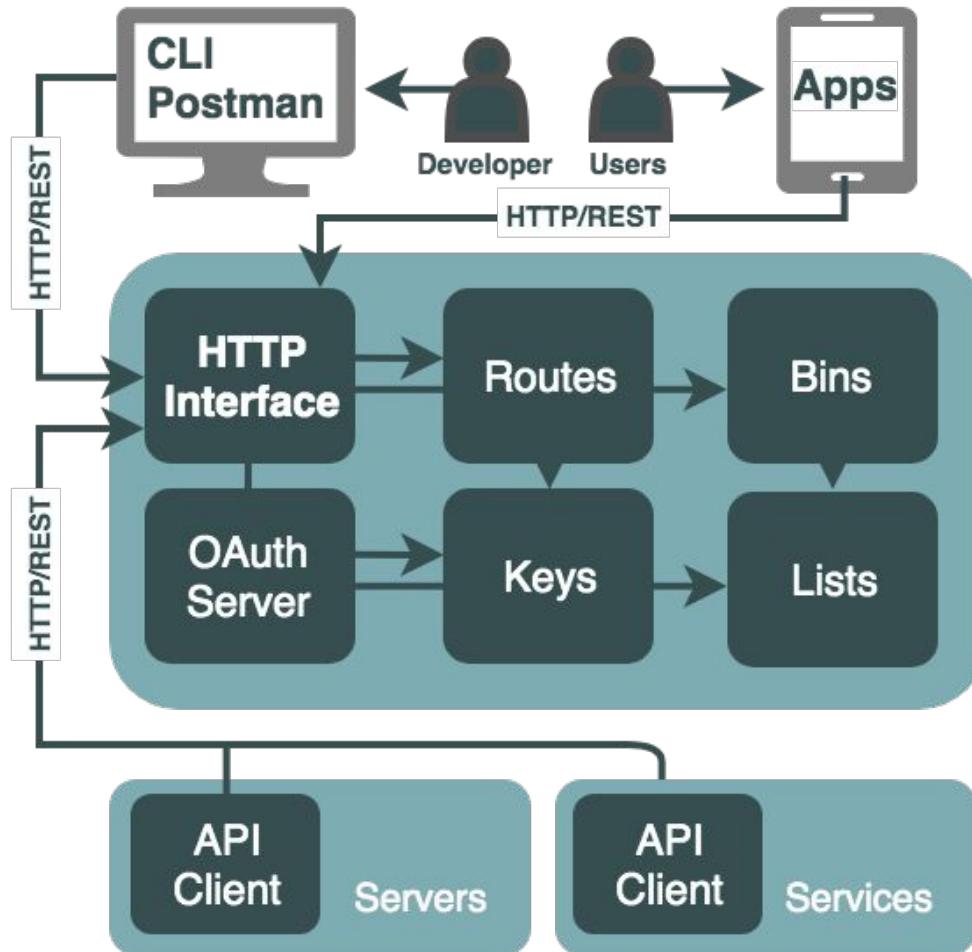


# In Simple Terms



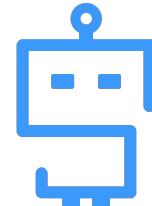
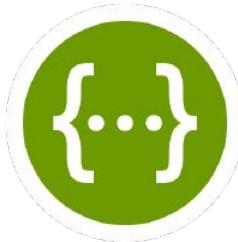
# Self Contained or External Redis





# REST API

- CRUD Interfaces for Keys & Lists
- Design First, Code Later
- API Specs at /swagger
- Use to render Documentation & SDKs



## Keys

**GET** /keys/{id} Read Key

**PUT** /keys/{id} Write Key

**DELETE** /keys/{id} Delete Key

**GET** /keys/ List Keys

**GET** /incr/{id} Increase Key

## Routes

**GET** /routes/{id} Read Key

**POST** /routes Create Route

## Lists

**POST** /lists/{id} Add Item to List

**GET** /lists/{id} Get Item from List

**DELETE** /lists/{id} Delete List

**GET** /lists Get All Lists



bambleweeny

22 requests

▶ config

▶ routes

▶ bins

▶ auth

▶ keys

...

**GET** Read Key

**PUT** Write Key

**DEL** Delete Key

**GET** List Keys

▶ save

▶ info

▶ incr

▶ lists

▶ users

**POST** Get Access Token

**PUT** Write Key

No Environment

Examples (0)

## ▶ Write Key

Send

Save

PUT

http://localhost:8080/keys/redisconf19

Params

Authorization

Headers (3)

Body

Pre-request Script

Tests

Cookies

Code

Comments (0)

none

form-data

x-www-form-urlencoded

raw

binary

JSON (application/json)

Beautify

~~1 hello world!~~

Body

Cookies (4)

Headers (5)

Test Results

Status: 200 OK

Time: 10 ms

Size: 170 B

Save

Download

Pretty

Raw

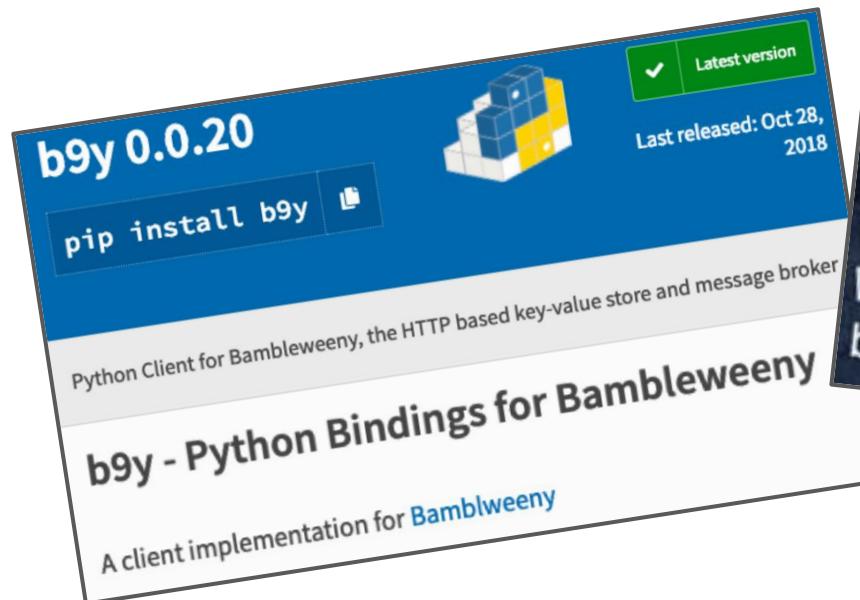
Preview

JSON

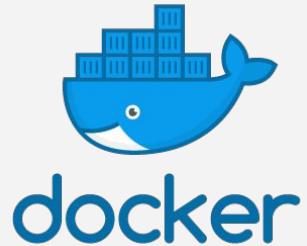


```
1 {  
2   "info": "ok"  
3 }
```

# Python Client Library & CLI



# Easy to Run



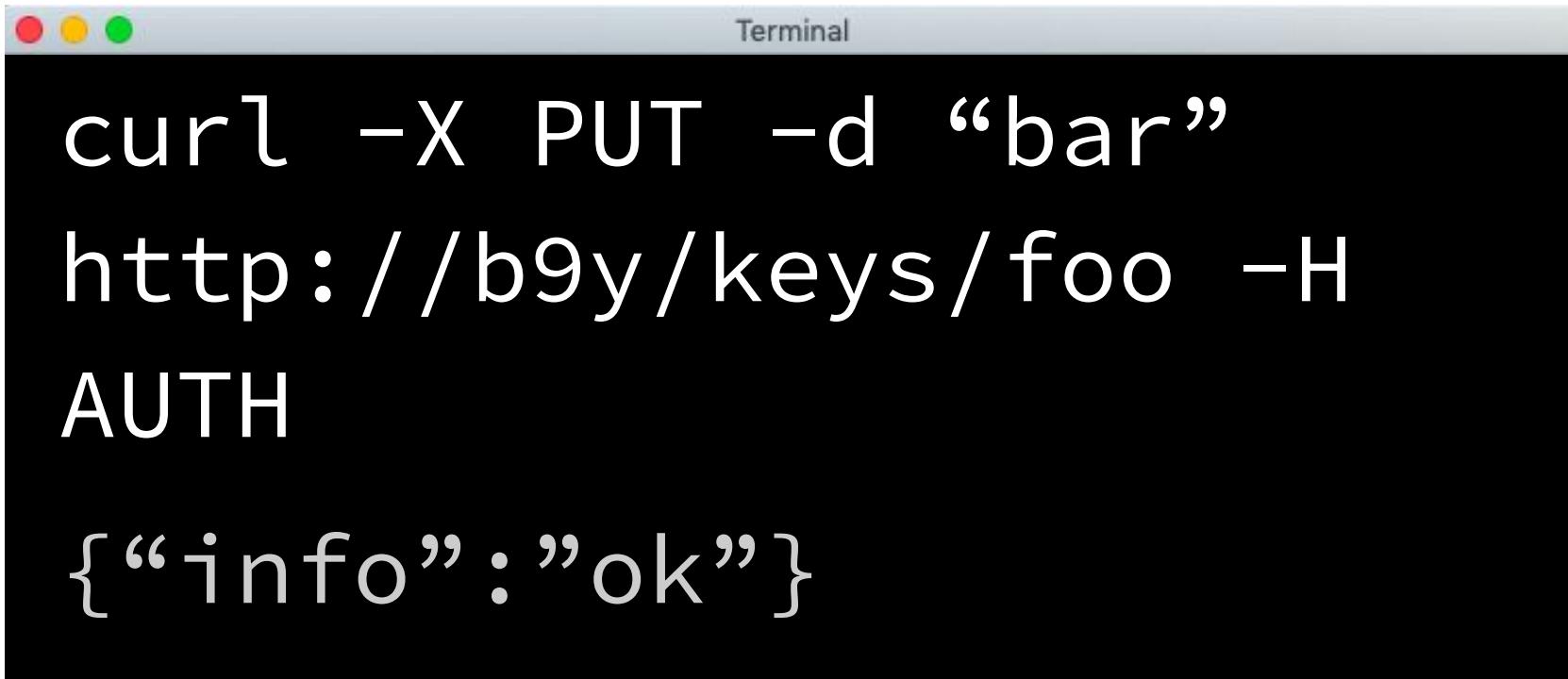
# cURL: Get 'foo'



Terminal

```
curl http://b9y/keys/foo  
-H AUTH  
bar
```

# cURL: Set ‘foo’ = ‘bar’

A screenshot of a Mac OS X Terminal window. The window has the standard red, yellow, and green close buttons at the top left. The title bar in the center says "Terminal". The main area of the window contains white text on a black background. It shows a cURL command being run to set the value of the 'foo' key to 'bar'. The command is: "curl -X PUT -d “bar” http://b9y/keys/foo -H AUTH". Below the command, the terminal displays the JSON response: {"info": "ok"}

```
curl -X PUT -d “bar”
http://b9y/keys/foo -H
AUTH
{"info": "ok"}
```

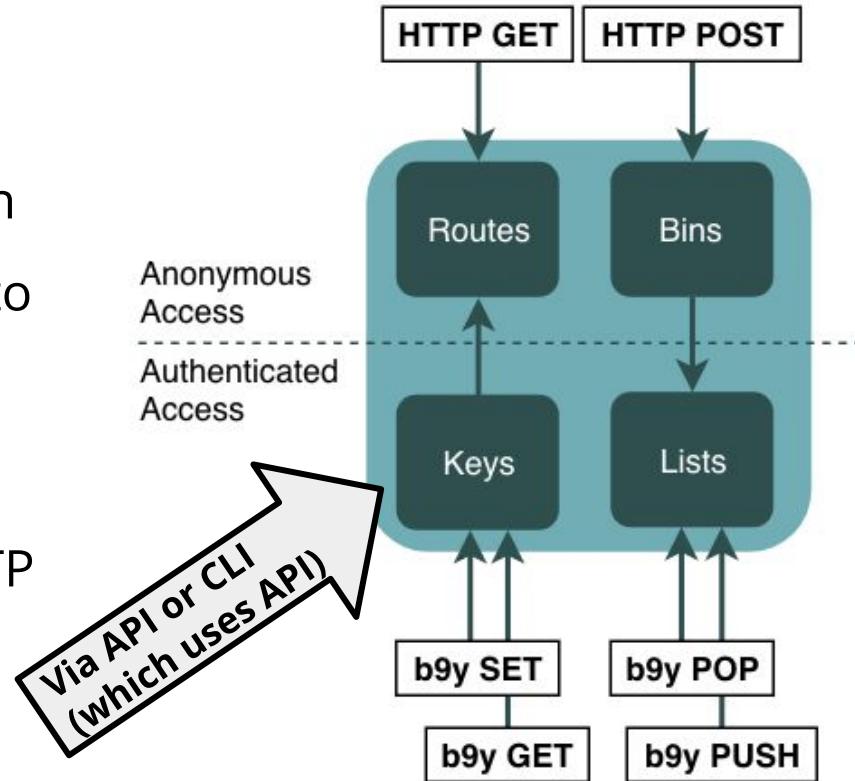
# cURL: Allow public access to 'foo'

```
Terminal  
echo '{"key":"foo",  
"content_type":"text/plain"}' | curl -X  
POST -d @- http://b9y/routes -H AUTH  
/routes/125e6a6f-c3f3-403b-b096-8997871  
curl http://b9y/routes/125e6a6f-c3f3-403  
bar
```



# OAuth & Access Concepts

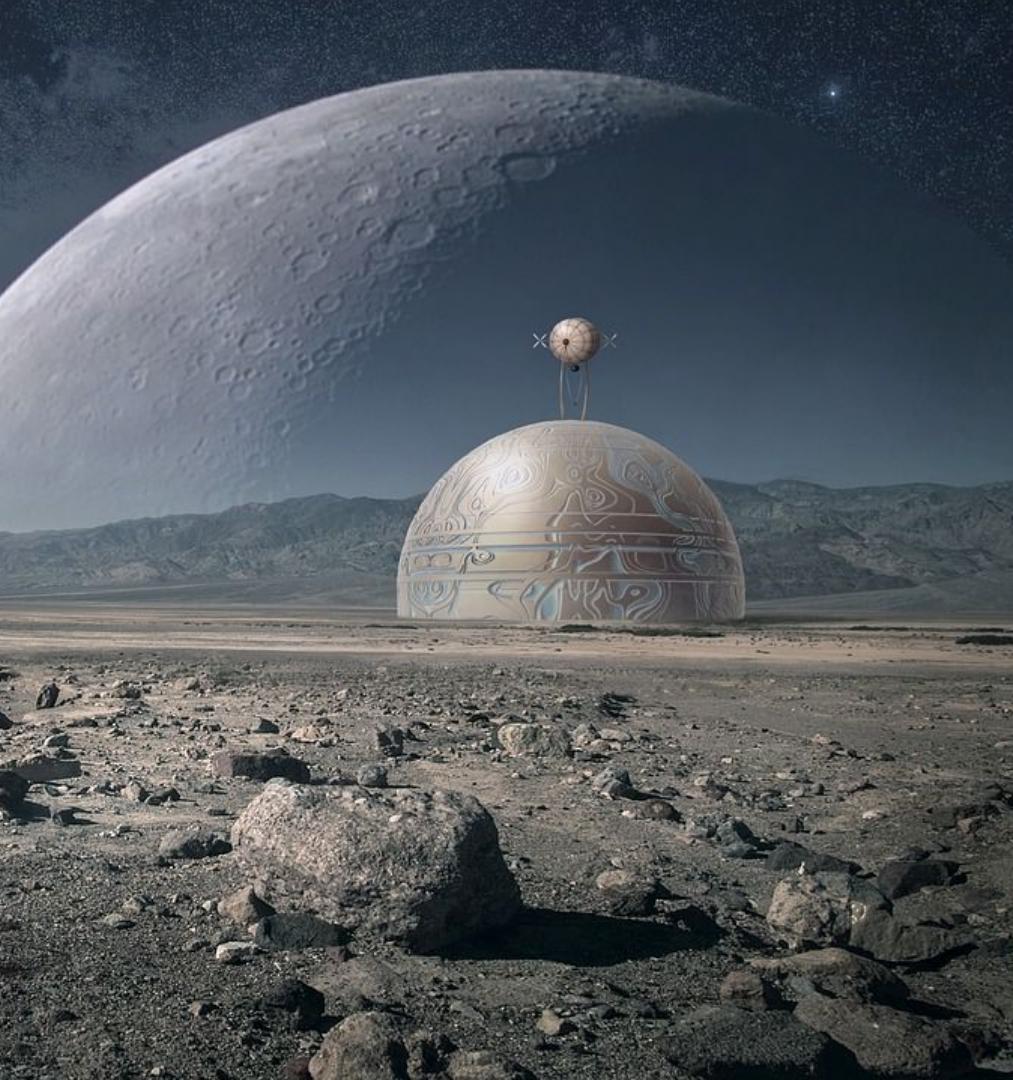
- Keys & Lists are Private
- Bearer Token via /auth/token
- Routes: Public READ Access to Keys via plain HTTP
- Bins: Public WRITE Access (“push”) to Lists via plain HTTP



# HTTP(s) is expensive

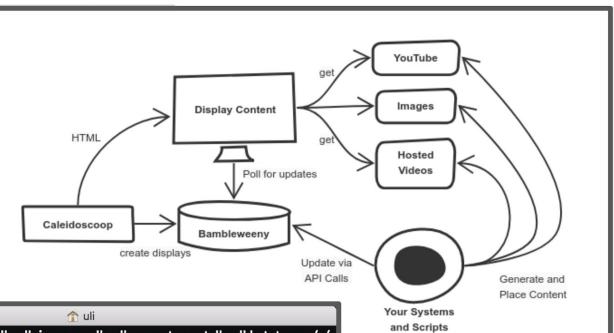
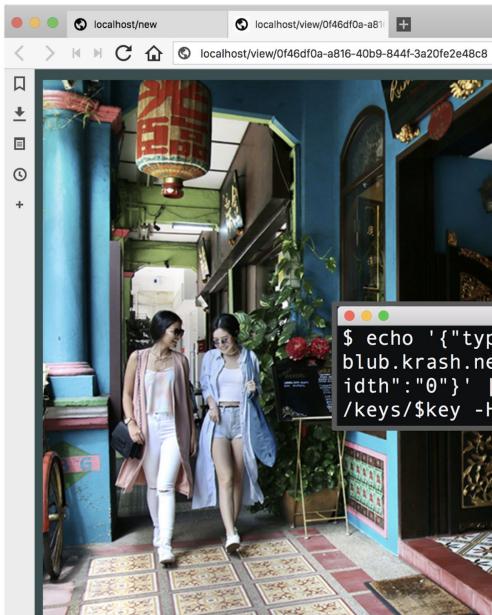
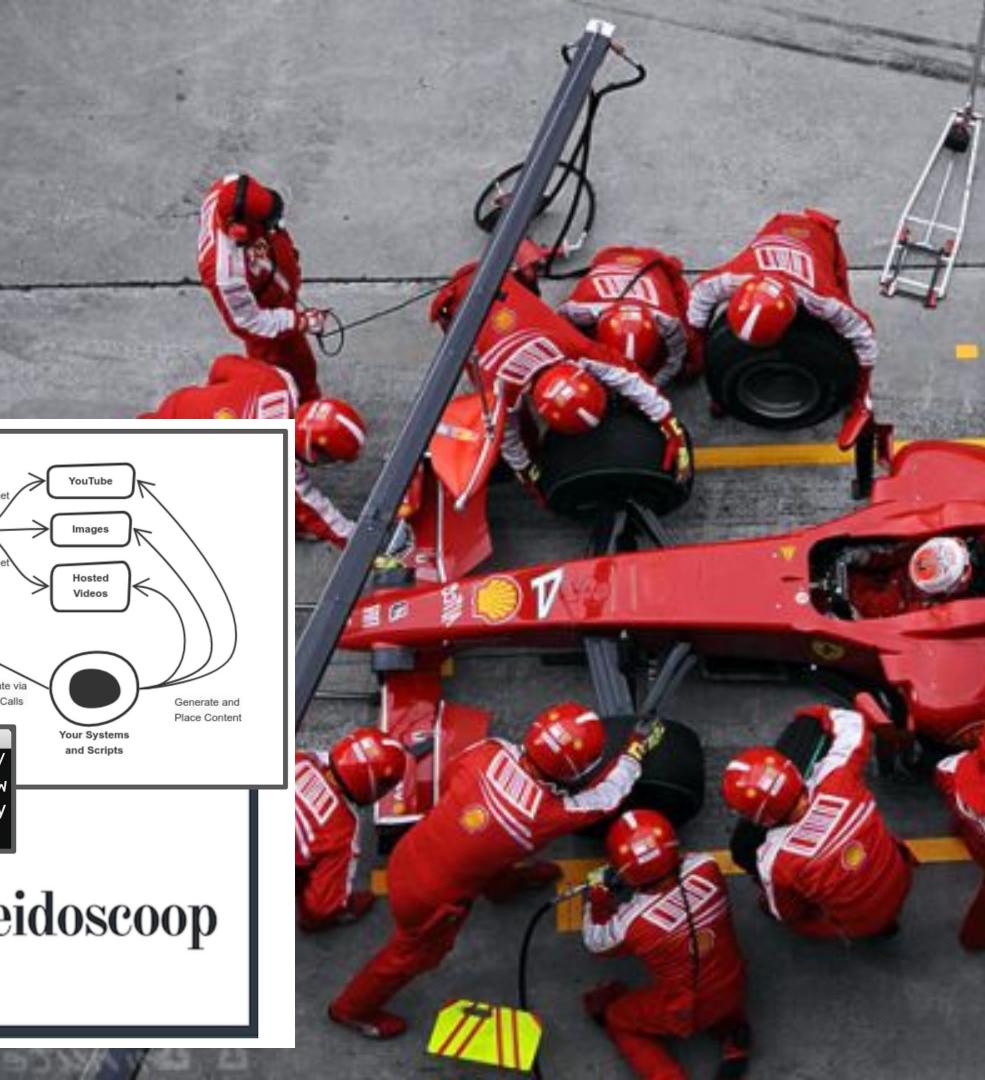
1x vCPU 1 GB RAM (AWS t2.micro)	Reads / second	Writes / second
<b>Redis 4.01</b>	<b>56k</b>	<b>55k</b>
<b>Bambleweeny</b>	<b>540</b>	<b>400</b>





Kubernetes  
Microservices  
Kafka  
Elastic  
Zookeeper  
MongoDB

# Use Case: Fast Prototyping

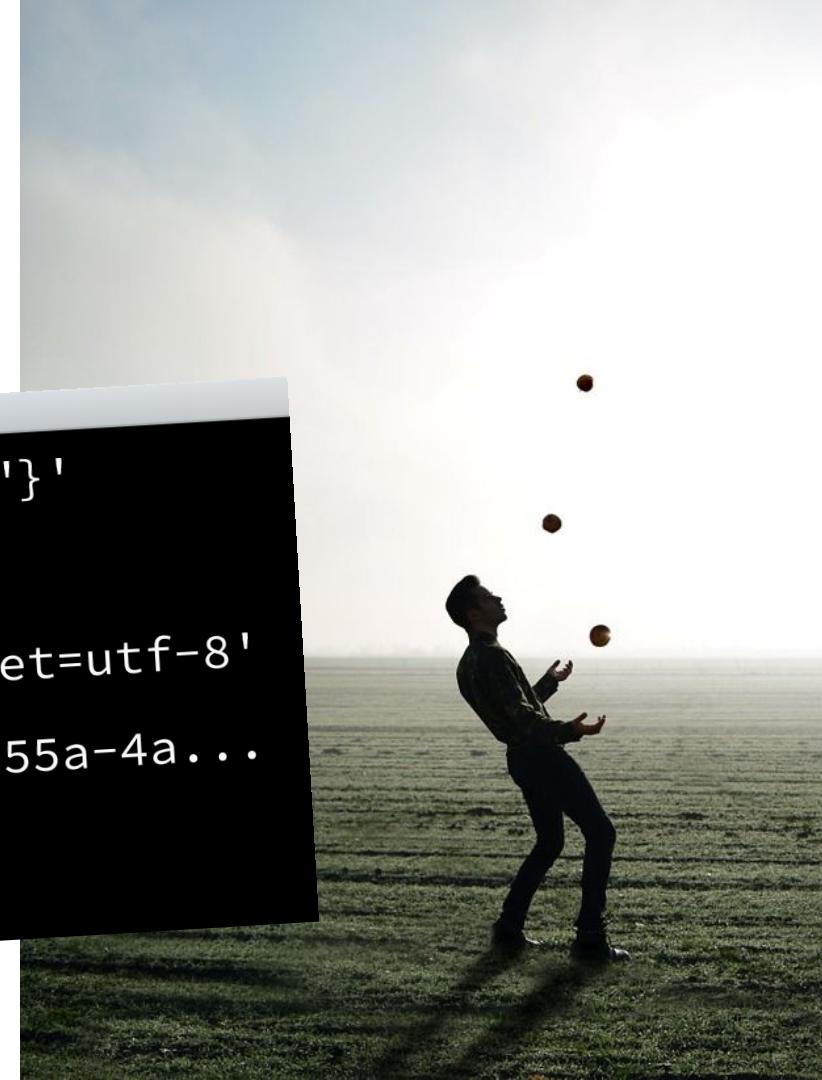


```
$ echo '{"type": "image", "content": "http://blub.krash.net/katong.jpg", "height": "0", "width": "0"}' | curl -X PUT -d @- http://b9y/keys/$key -H $AUTH
```

caleidoscoop

# Use Case: POCs & API Mocking

```
Terminal
set api '{"message": "!@[message]"}'
set message 'cool stuff!'
route api 'application/json;charset=utf-8'
curl http://b9y/routes/cf670f2b-755a-4a...
{"message": "cool stuff!"}
```



# Get involved

- Try it out!
- Help make it better → open GitHub issues for bugs & new features
- Spread the Word!



/u1i/bambleweeny

