

From Code to Community

Uli Hitzel
Python Singapore User Group
November 2019



```
put('/keys/<id>', method='PUT')
    if key:
        auth = _authenticate()
        if not auth['authenticated']:
            response.status = 401
            return dict({"info": "Unauthorized."})

    # 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 name is invalid."})

    # Construct Resource Location from user_id and id
    redis_key = "KEY:" + str(user_id) + ":" + str(id)

    # Does the key exist already?
    if not exists = rc.get(redis_key):
        keyexists == None:
            # 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!

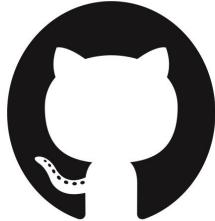




Developer Advocate



/in/uhitzel



/u1i/slides



IBM



Agenda

1 – Open Source &
How To Run Projects

2 – Bambleweeny

3 – Get involved!



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

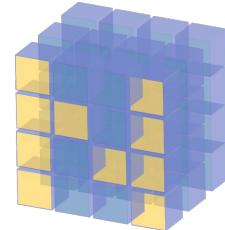
GET	/keys/{id}	Read Key	🔒
PUT	/keys/{id}	Write Key	🔒
DELETE	/keys/{id}	Delete Key	🔒
GET	/keys/	List Keys	🔒
GET	/incr/{id}	Increase Key	🔒



TensorFlow



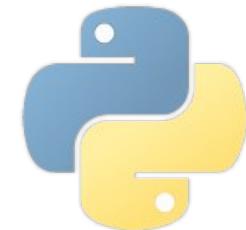
Scrapy



NumPy



Requests
http for humans



Open Source



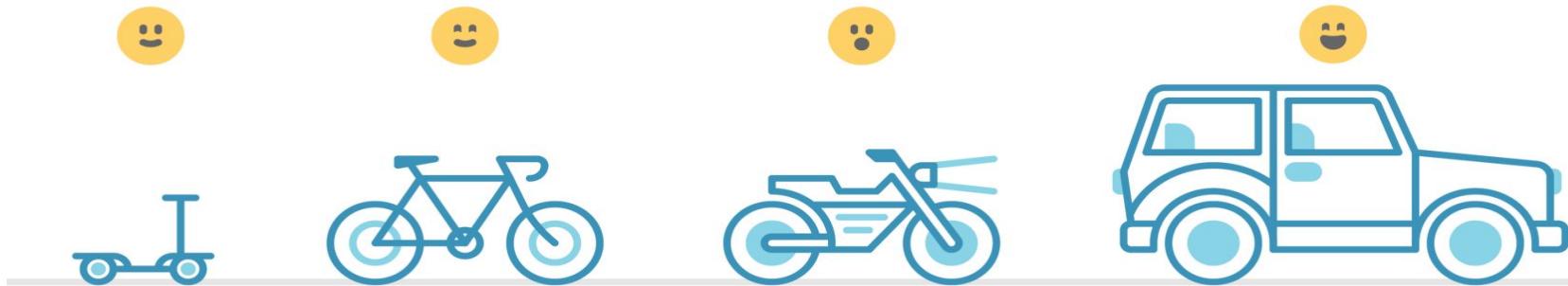
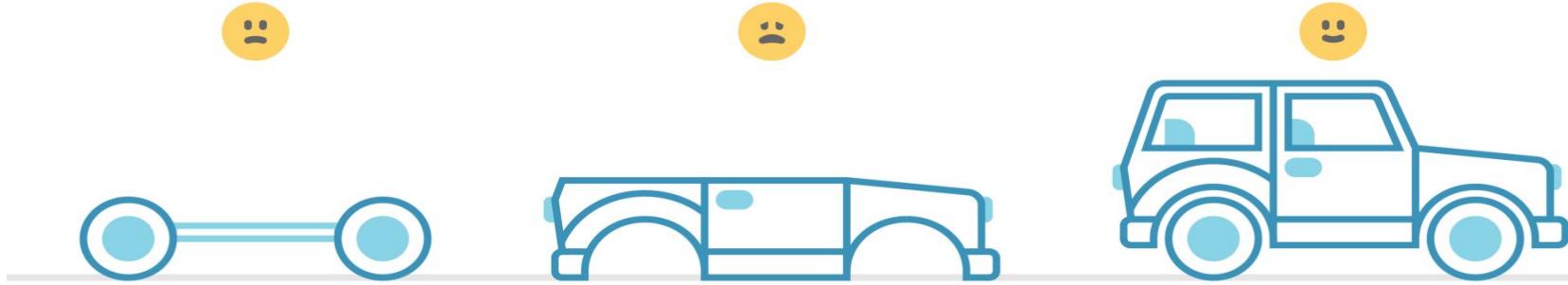
Community
powered
Innovation

Open Source

People will **see** my
Code

People will **use** my
Stuff



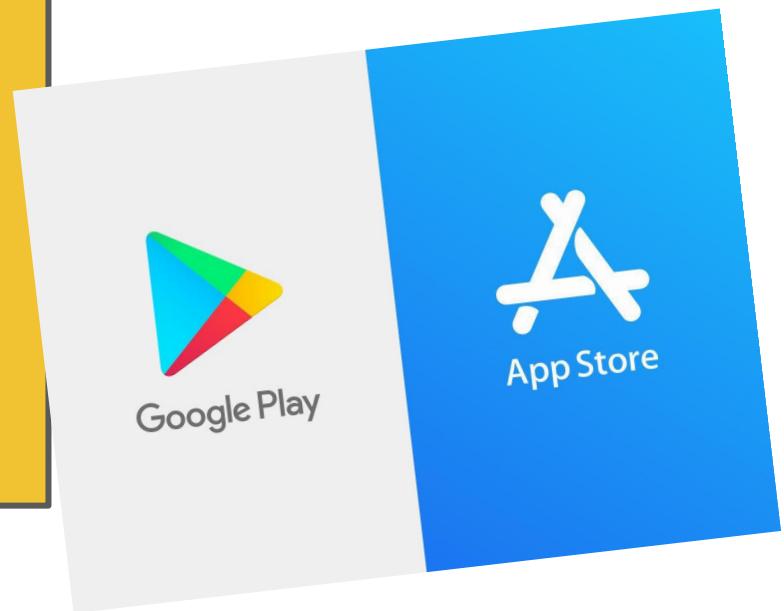


Why do it?

- 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



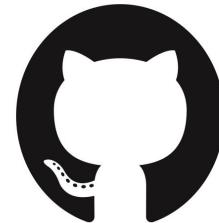


Market it!

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

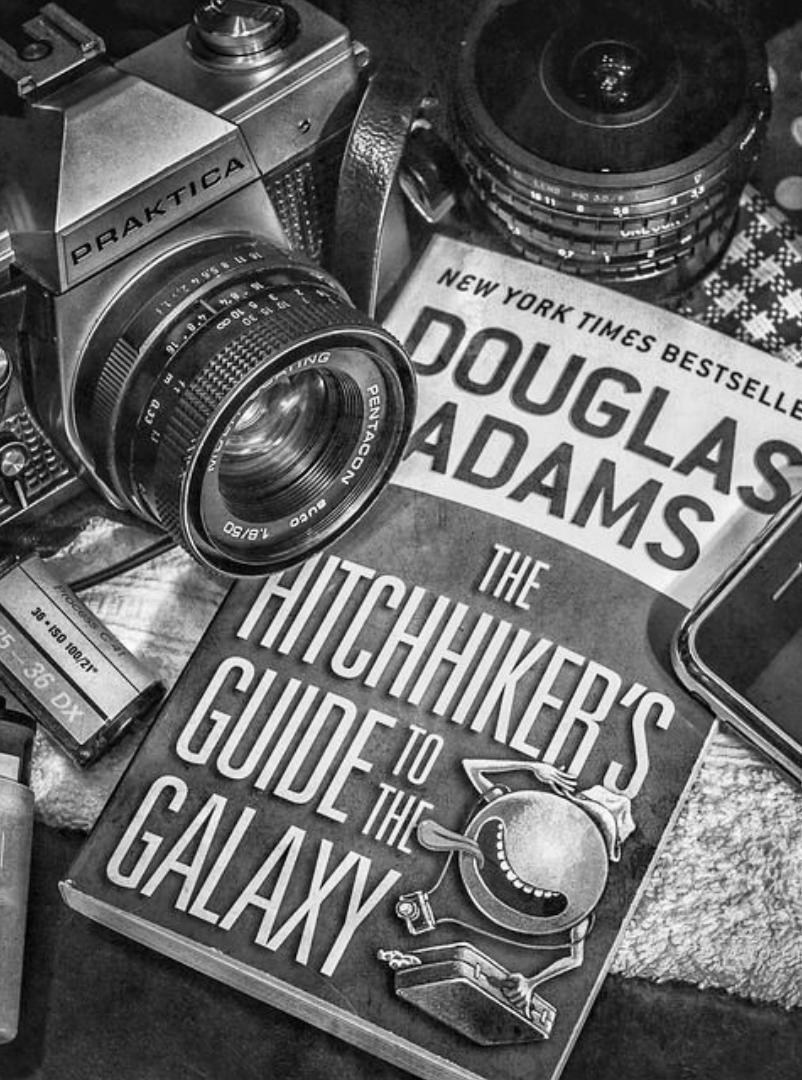
2497

issues

5 open



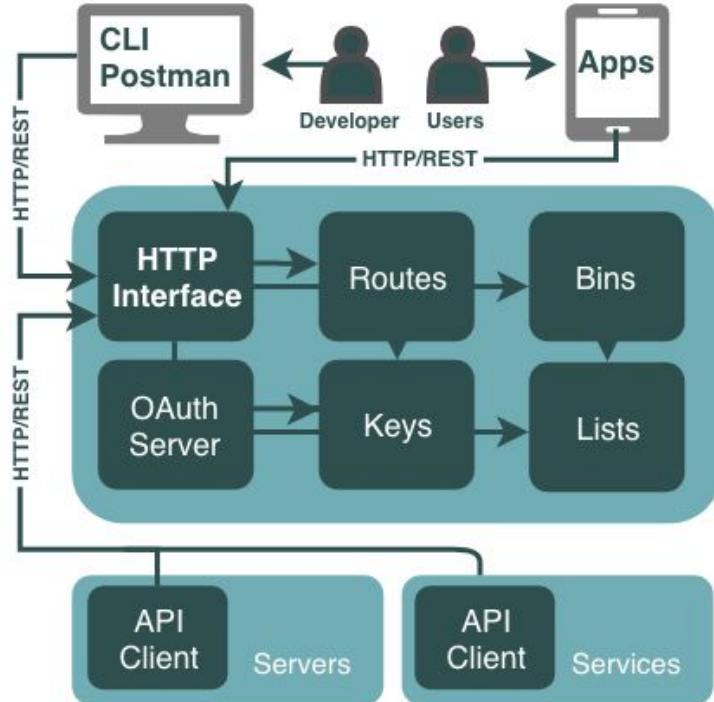
bamble
weeny



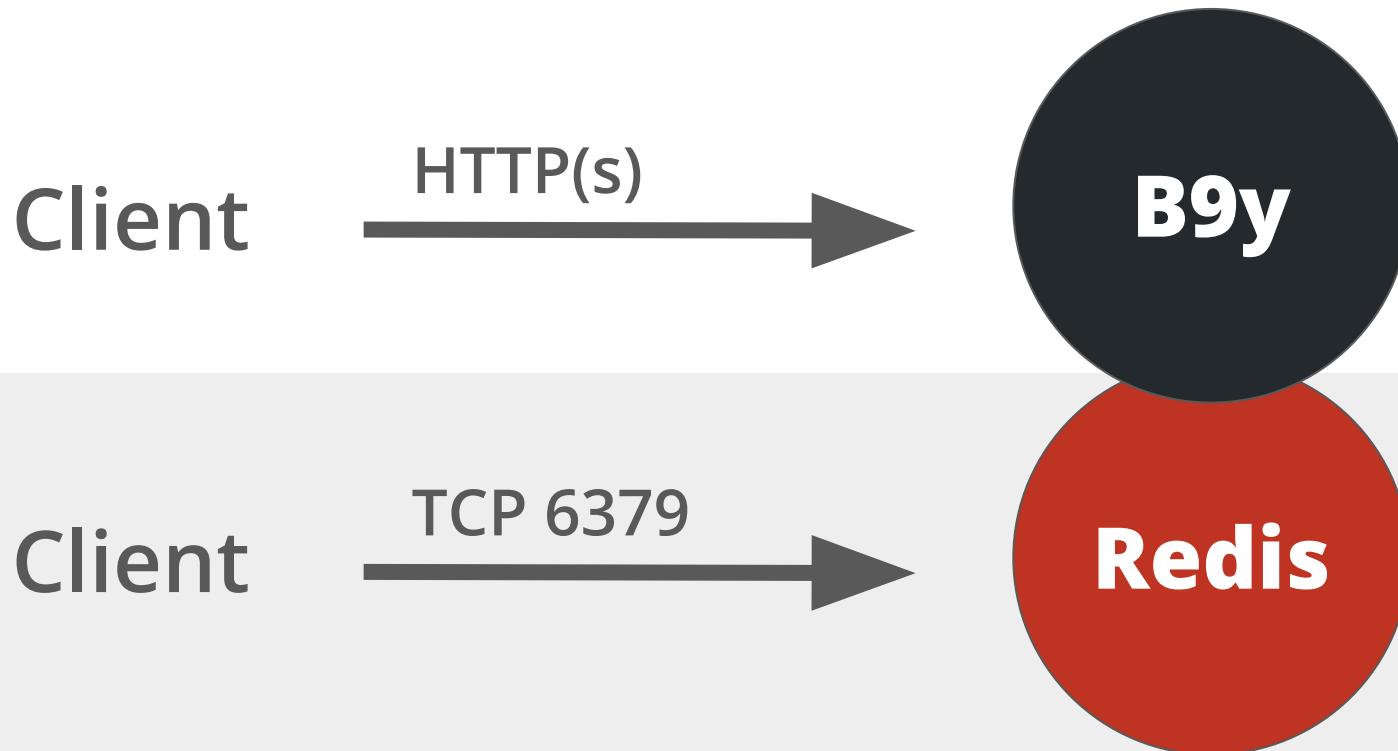
The Bambleweeny
57 Submeson
Brain is part of the
Finite
Improbability
Generator.

Bambleweeny

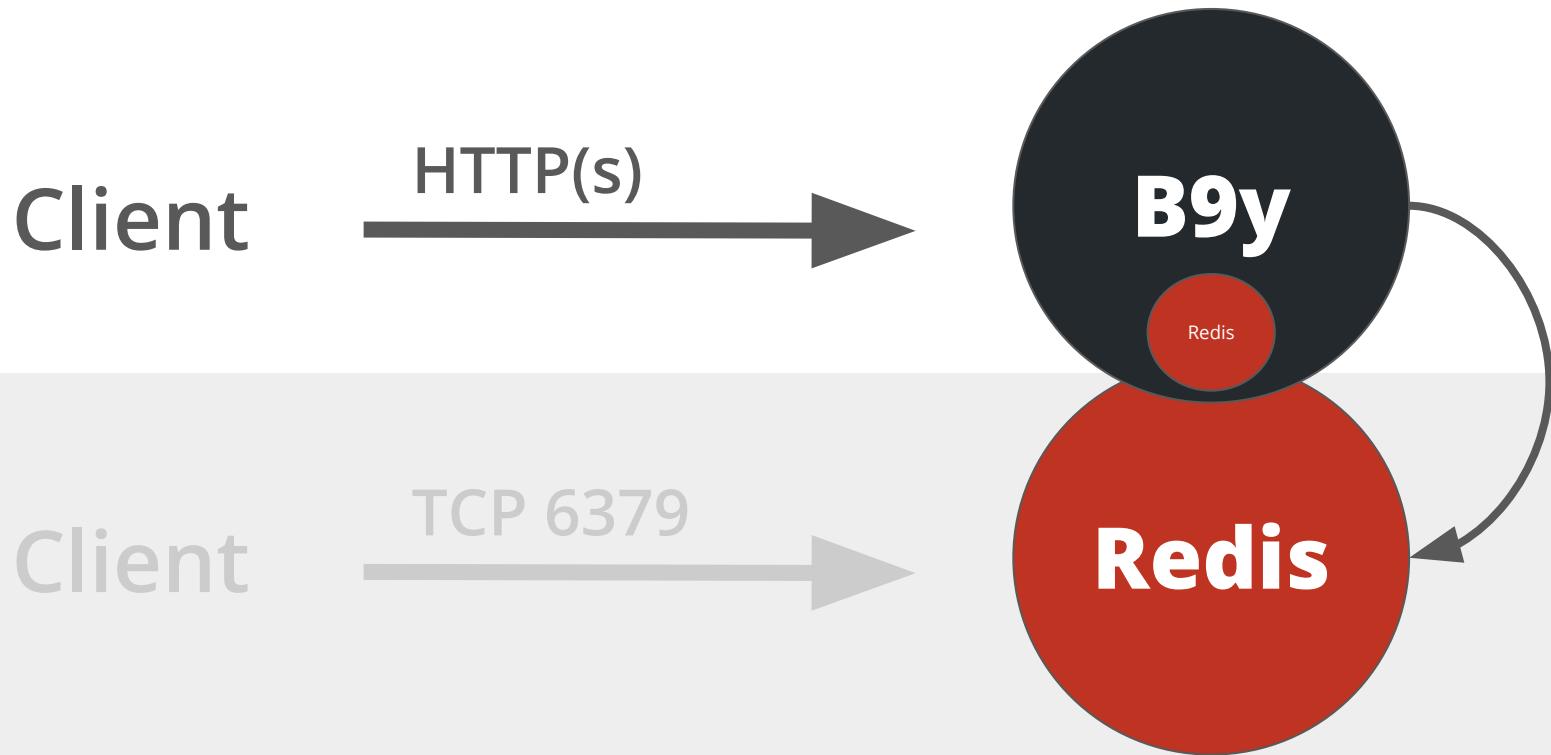
- Lightweight HTTP/REST based Key-Value Store & Message Broker
- Multi-Tenancy → Manage Identities, Access & Quotas
- Fast, Easy to Use & Well-Documented
- Written in Python, using Redis, deployable in a tiny Container

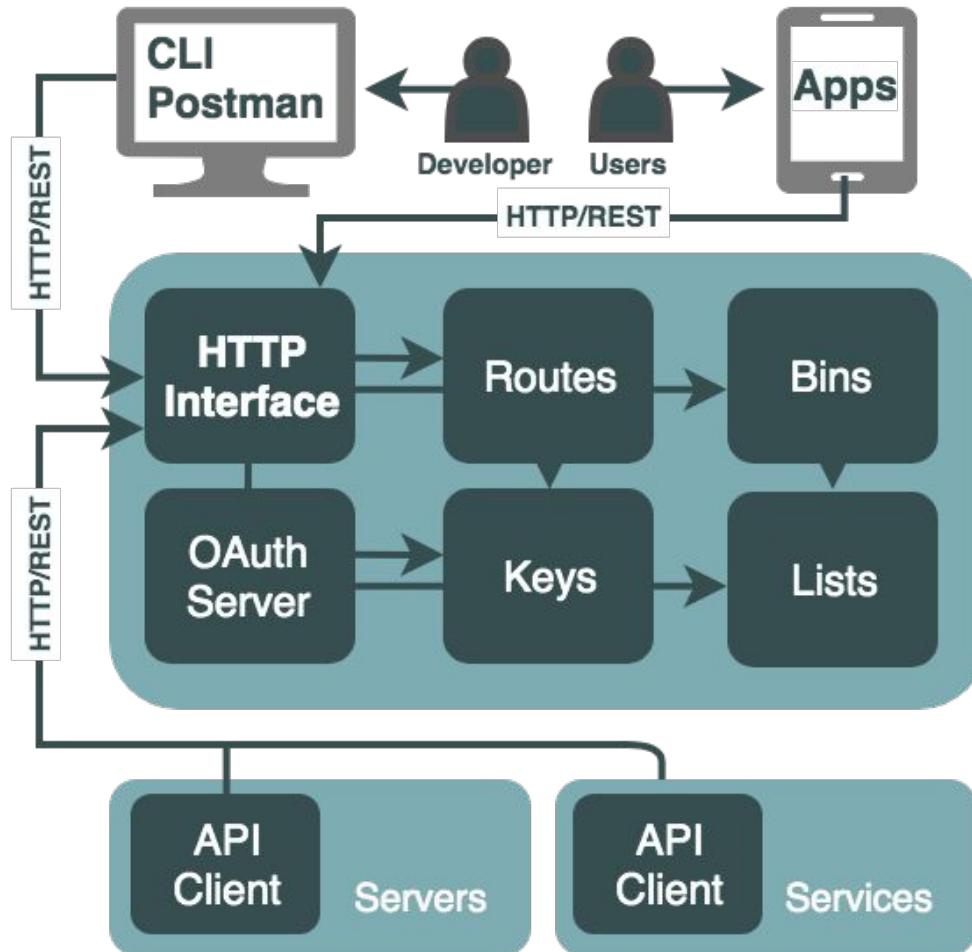


In Simple Terms



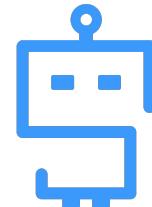
Self Contained or External Redis





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

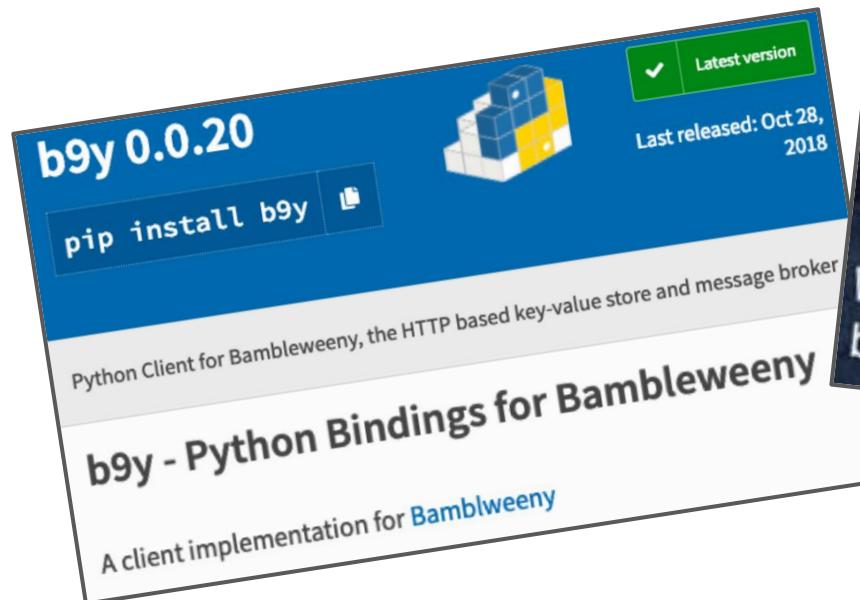
Preview

JSON

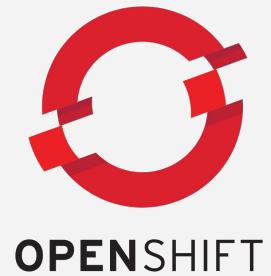
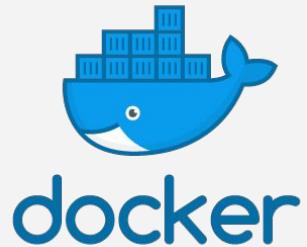


```
1 {  
2   "info": "ok"  
3 }
```

Python Client Library & CLI



Easy to Run

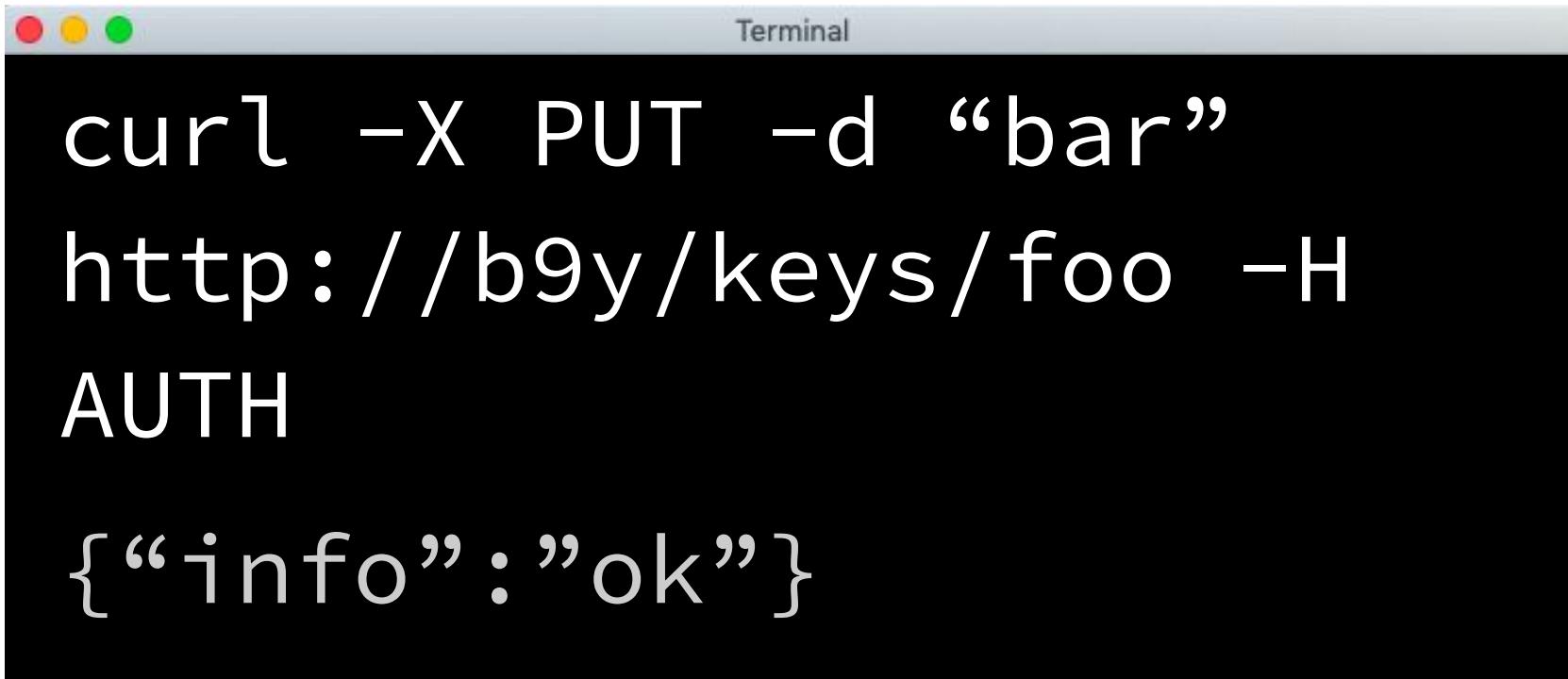


cURL: Get 'foo'

A screenshot of a Mac OS X Terminal window. The window has the standard title bar with red, yellow, and green close buttons, and the word "Terminal". The main area of the window is black and contains white text. The text shows a cURL command being entered. The command consists of "curl http://b9y/keys/foo" on the first line, "-H AUTH" on the second line, and "bar" on the third line.

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

cURL: Set ‘foo’ = ‘bar’

A screenshot of a Mac OS X Terminal window. The title bar says "Terminal". The window contains the following text:

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

The text is white on a black background, matching the terminal's theme.

A red, yellow, and green window control icon is visible in the top-left corner of the terminal window.

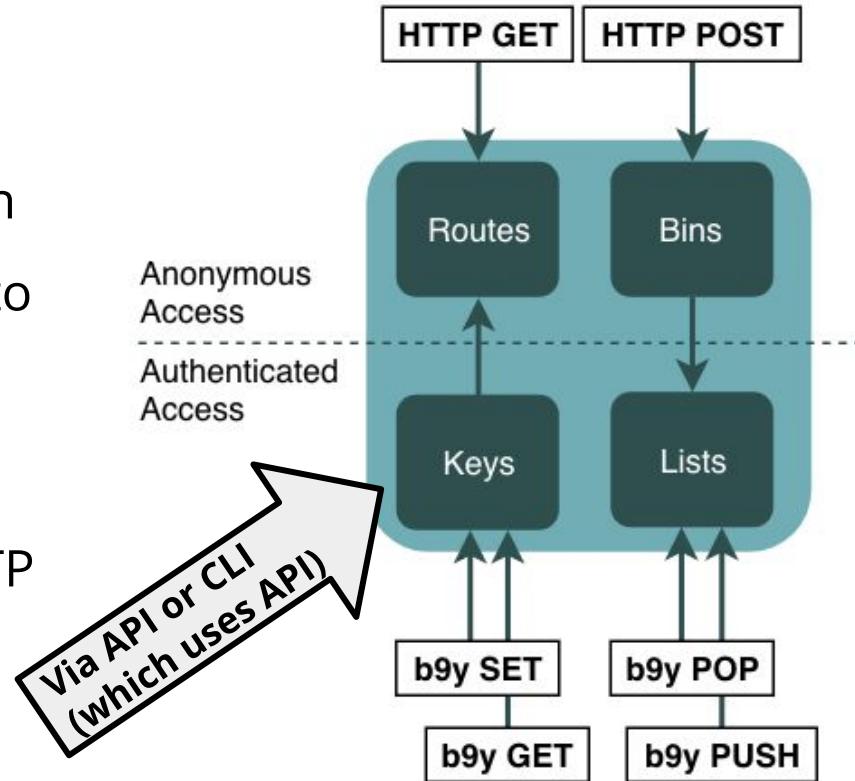
cURL: Allow public access to 'foo'

```
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

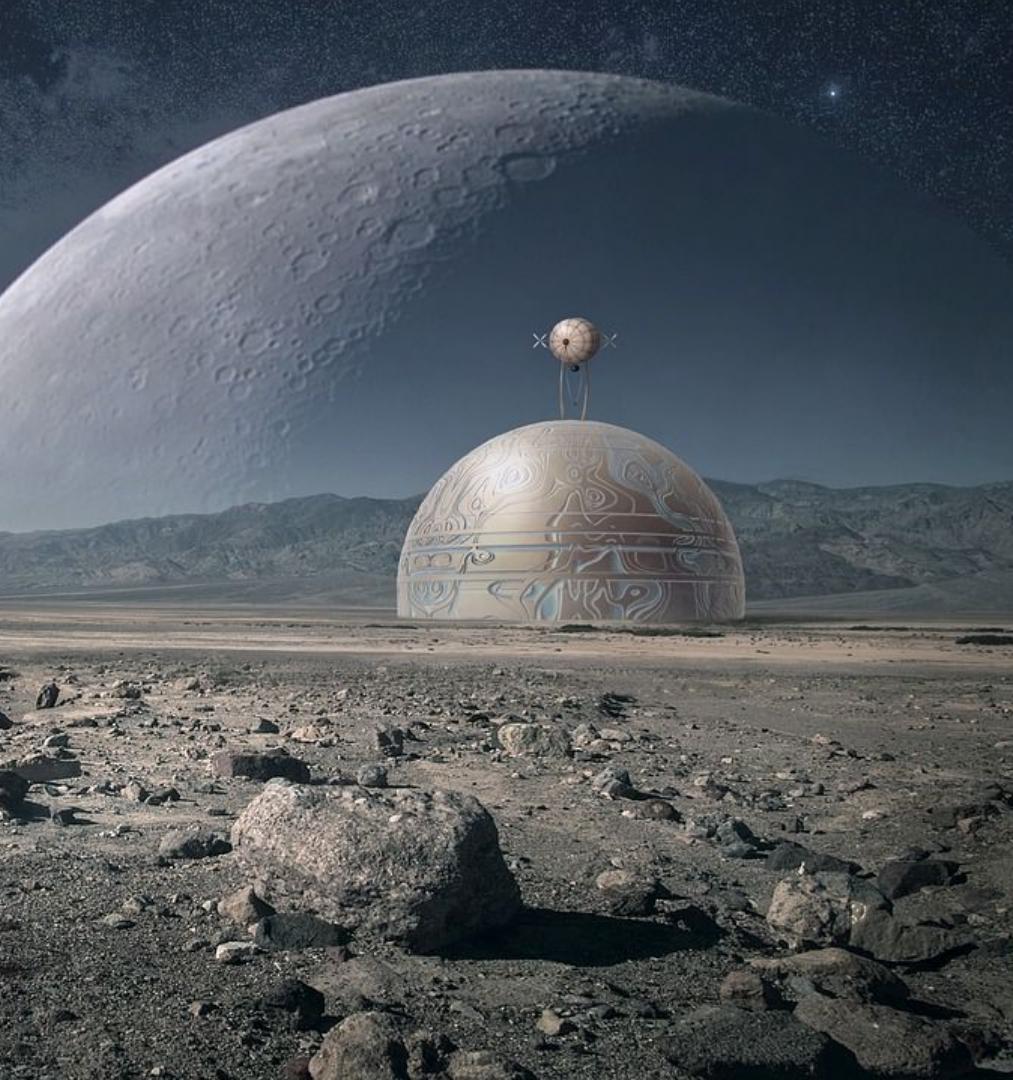
- Keys & Lists are Private
- Bearer Token via /auth/token
- Routes: Public READ Access to Keys via plain HTTP
- Bins: Public WRITE Access (“push”) to Lists via plain HTTP



HTTP(s) is expensive

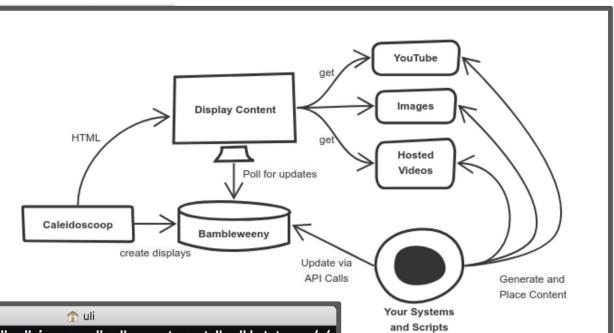
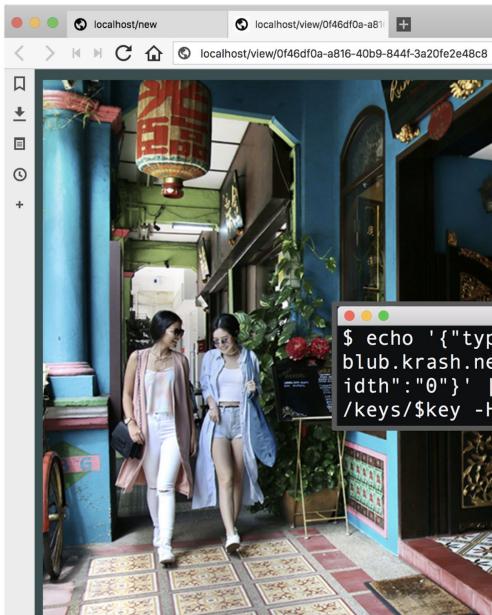
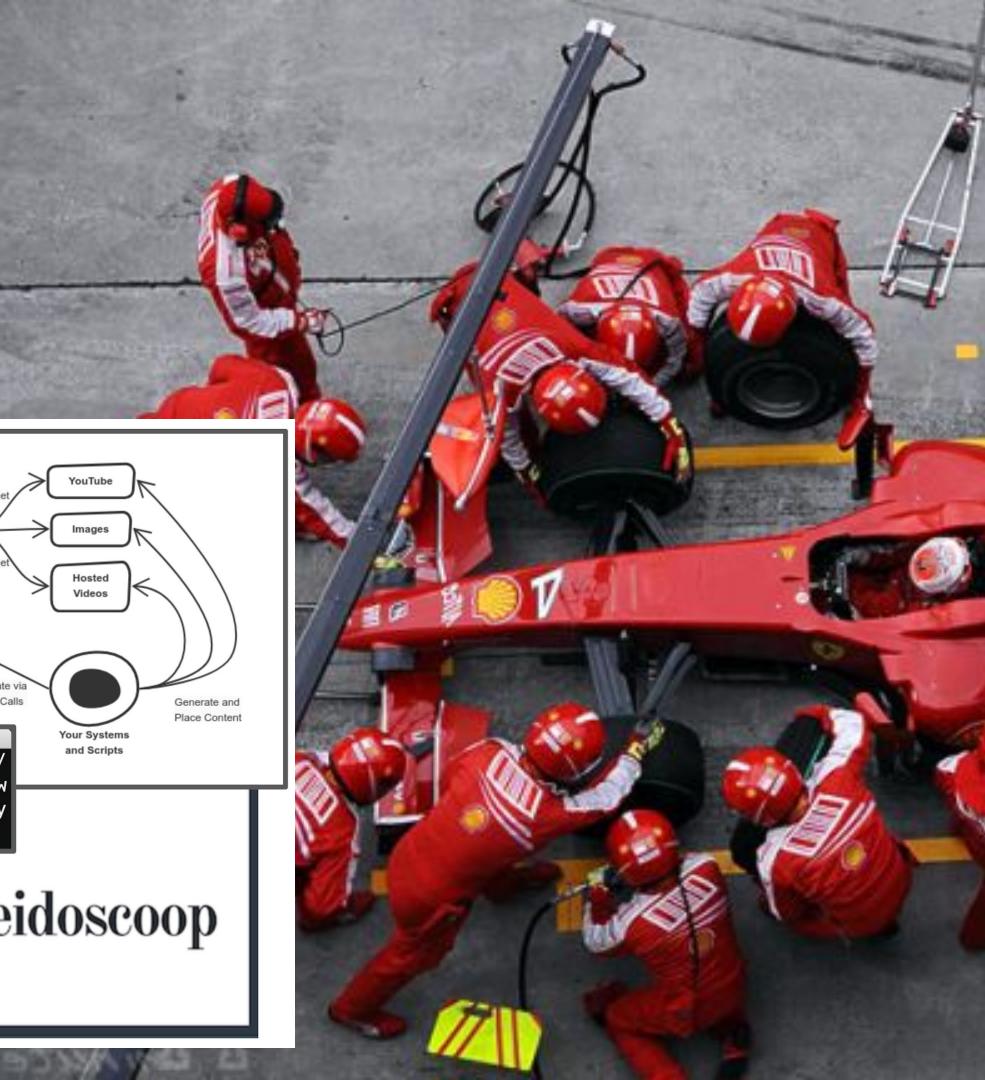
1x vCPU 1 GB RAM (AWS t2.micro)	Reads / second	Writes / second
Redis 4.01	56k	55k
Bambleweeny	540	400





Kubernetes
Microservices
Kafka
Elastic
Zookeeper
MongoDB

Use Case: Fast Prototyping

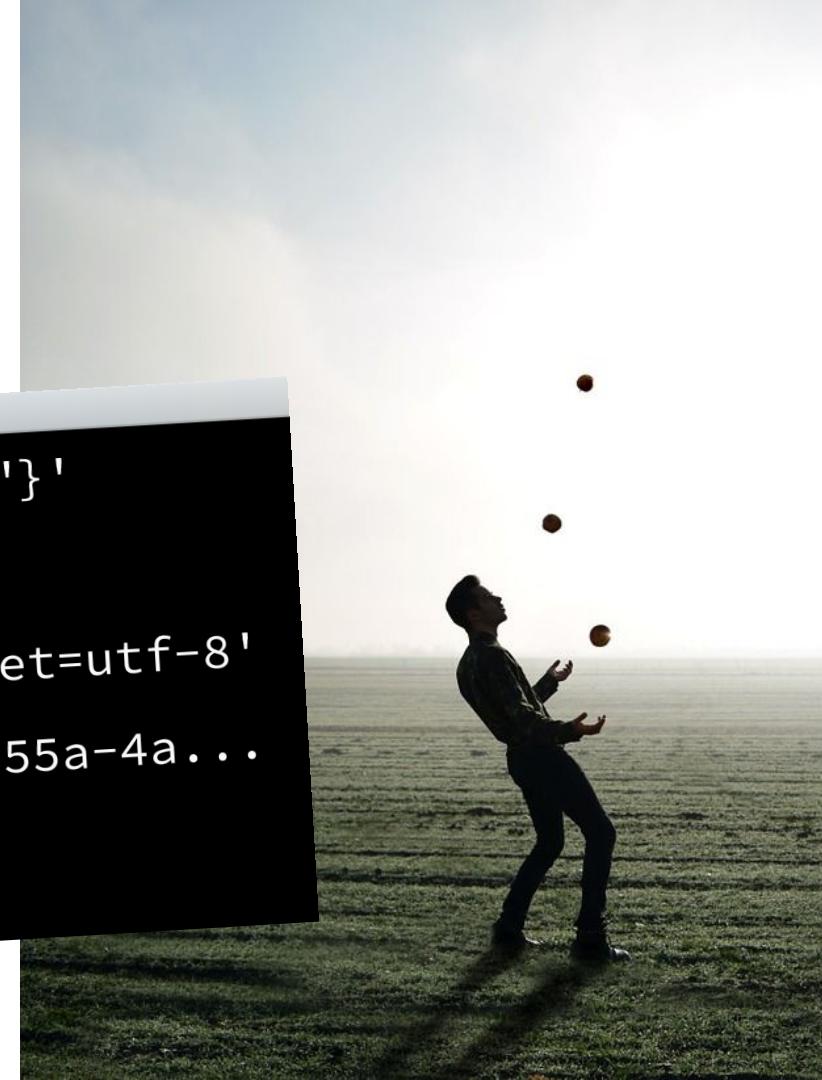


```
$ echo '{"type": "image", "content": "http://blub.krash.net/katong.jpg", "height": "0", "width": "0"}' | curl -X PUT -d @- http://b9y/keys/$key -H $AUTH
```

caleidoscoop

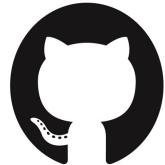
Use Case: POCs & API Mocking

```
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

