API design ?!

best practices to create a successful API

http://bit.ly/1A0LJVk

Nicolas Grenié

Hacker in Residence at 3scale.net
@3scale

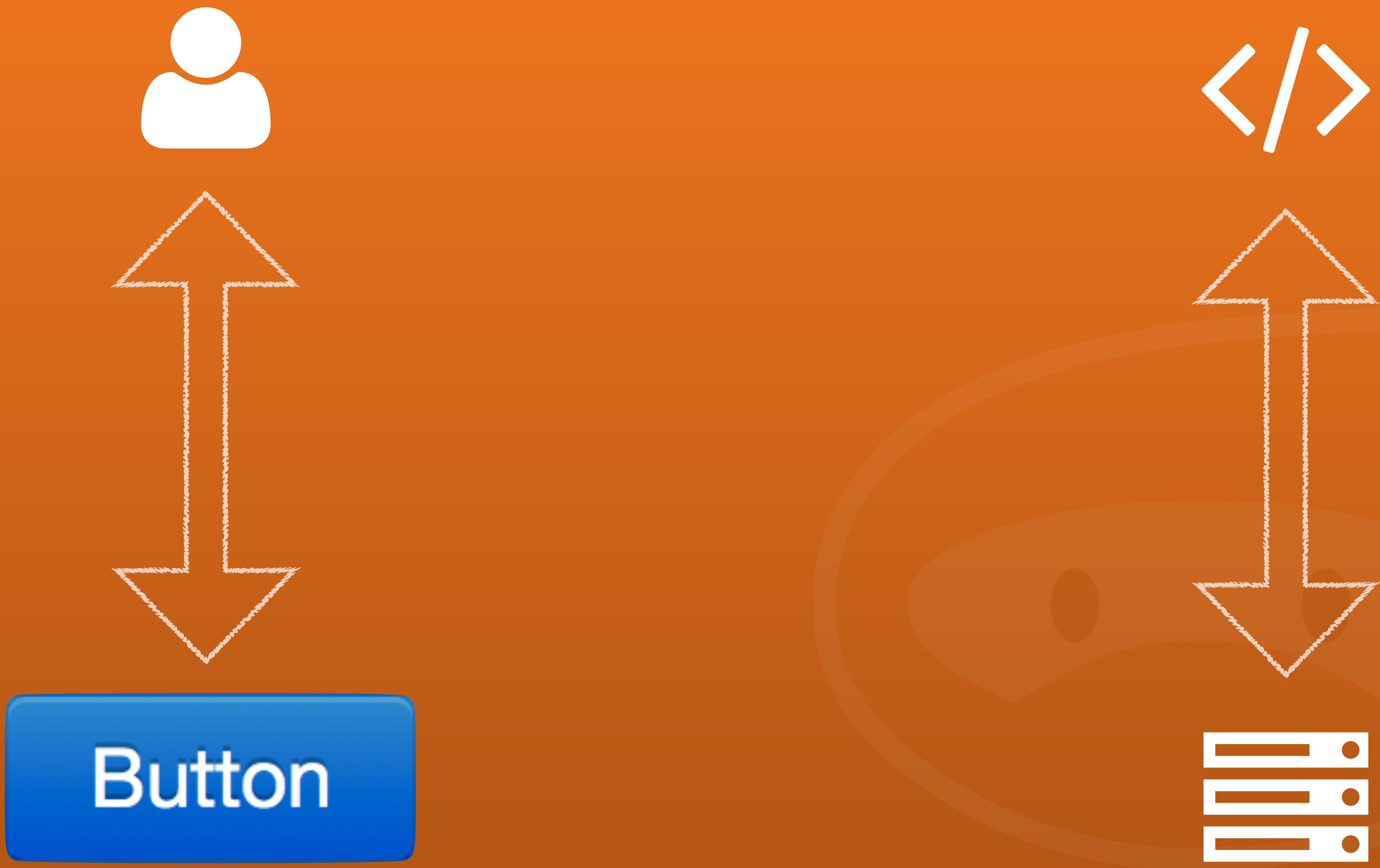
API Workshop at LeWagon
Paris, February 12, 2015

3SCALE

Application
Programming
Interface

UI

API



Examples

stripe

Google maps

mailjet™

twilio™

There is an API for that

You need an API

3SCALE

an API, different goals

- mobile NETFLIX Quora
- customer eco-system amazon webservices
- partner eco-system Spotify
- content distribution IGN Entertainment ESPN
- € API as a Business twilio stripe

SHITY API
SHITY API EVERYWHERE
memegenerator.net

plan and design the API

don't code!

(yet)

Plan

Who are the users? Which industry?

What are the use cases?

- REST, SOAP, RPC
- JSON, XML, CSV...
- fine or coarse-grained

3SCALE

fine or coarse-grained example

https://api.jcdecaux.com/vls/v1/stations?contract_name=Velib&apiKey=461b6a213074fdd841b9c10bf6d5d5010155aee2

Size of JSON response 235kB

- is it valuable?
- how could it be improved?

Design

Versioning

HTTP valid

URLs schema

Hide architecture

Design - Versioning

Following software analogy
API should be versionated v1 , v1.1, v2.0

http://mysite.com/api/v1/books.json

http://mysite.com/api/v1.1/books.json

http://mysite.com/api/books.json?v=1.1

```
curl -X GET -H "X-API-VERSION: 1.1" \
http://mysite.com/api/books.json
```

Design - Versioning

How often are you going to change version?

What about people already using your API?

How many versions will you keep maintaining?

What's the roadmap?

Design - Versioning

- Don't explicitly version your API
- non breaking changes
- no version = flexibility = win

Design - HTTP valid

Respect standards

Know the HTTP Verbs

Return proper HTTP code

Verb	Endpoint	What for
GET	*/book/{book_id}*	Retrieve details of a resource
POST	*/book*	Create resource
DELETE	*/book/{book_id}*	Delete resource
PUT	*/book/{book_id}*	Update or create resource
PATCH	*/book/{book_id}*	Update partial resource
OPTIONS	*any URL*	return methods supported on this URL
TRACE	*any URL*	Echoes
CONNECT	*any URL*	Convert to TCP/IP tunnel
HEAD	*/book/{book_id}*	Same as GET w/o response body

Design - HTTP code

404 PAGE NOT FOUND

THANK YOU MARIO!

BUT OUR PRINCESS IS IN ANOTHER CASTLE!

Design - HTTP code

Design - HTTP code

Create a new ressource with

POST /book

➔ returns *201 created*

Delete a ressource with

DELETE /book/{book_id}

➔ returns *200 success or 204 request processed*

3SCALE

Design - HTTP code

Custom error? Create your own.

➔ it should not be already defined in the spec

➔ 1xx Informational
2xx Success
3xx Redirection
4xx Client error
5xx Server error

➔ document it !!

Design - URLs

Be explicite and intuitive

GET /books - Retrieves a list of books
GET /books/42 - Retrieves a specific book
POST /books - Creates a new book
PUT /books/42 - Updates book #42
PATCH /books/42 - Partially updates book #42
DELETE /books/42 - Deletes book #42

Singular or plural? keep it simple

Design - URLs

Relations ?

GET /books/42/reviews - Retrieves list of reviews for book #42

GET /books/42/reviews/5 - Retrieves review #5 for book #42

GET /books/42/reviews/5/likes - Retrieves likes of review #5 for book #42

➔ GET /reviews/5/likes

3SCALE

Design - URLs

Also for non-CRUD operations

Could be a sub-resource.

PUT /gists/:id/star - Star a gist

DELETE /gists/:id/star - Unstar a gist

Or on it’s own

GET /search

keys of a successful API

intuitive

documented

opinated

*inspired by Kevin Lacker https://www.youtube.com/watch?v=qCdpTji8nxo

3SCALE

Let's code ? :)

Tools

{ grape }

micro-framework for REST APIs in Ruby
https://github.com/intridea/grape

POSTMAN

test REST apis in the browser
http://getpostman.com

Scalingo

easy deploy for apps
http://scalingo.com

3scale

API management
http://scalingo.com

Our API

Basic sentiment analysis on words

Words stored in a text file

```
…
abilities 2
ability    2
aboard 1
absentee     -1
absentees  -1
absolve2
absolved     2
…
```

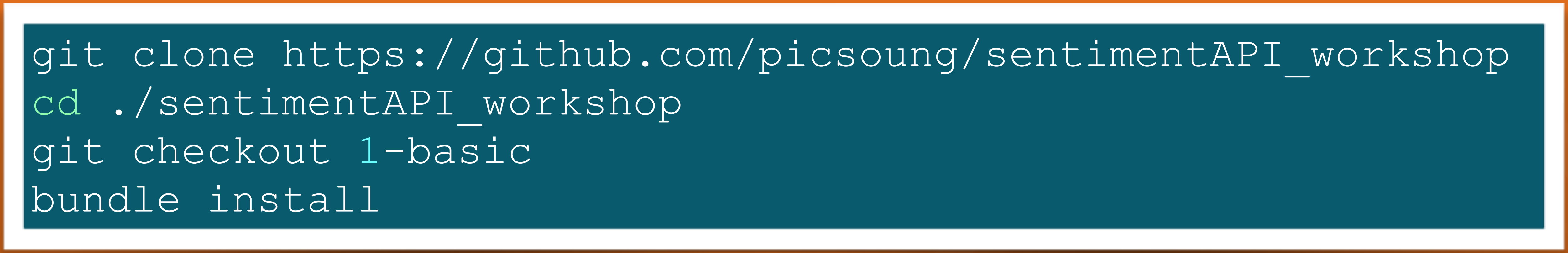
GET sentiment of a word

POST sentiment of a word

Requests authenticated

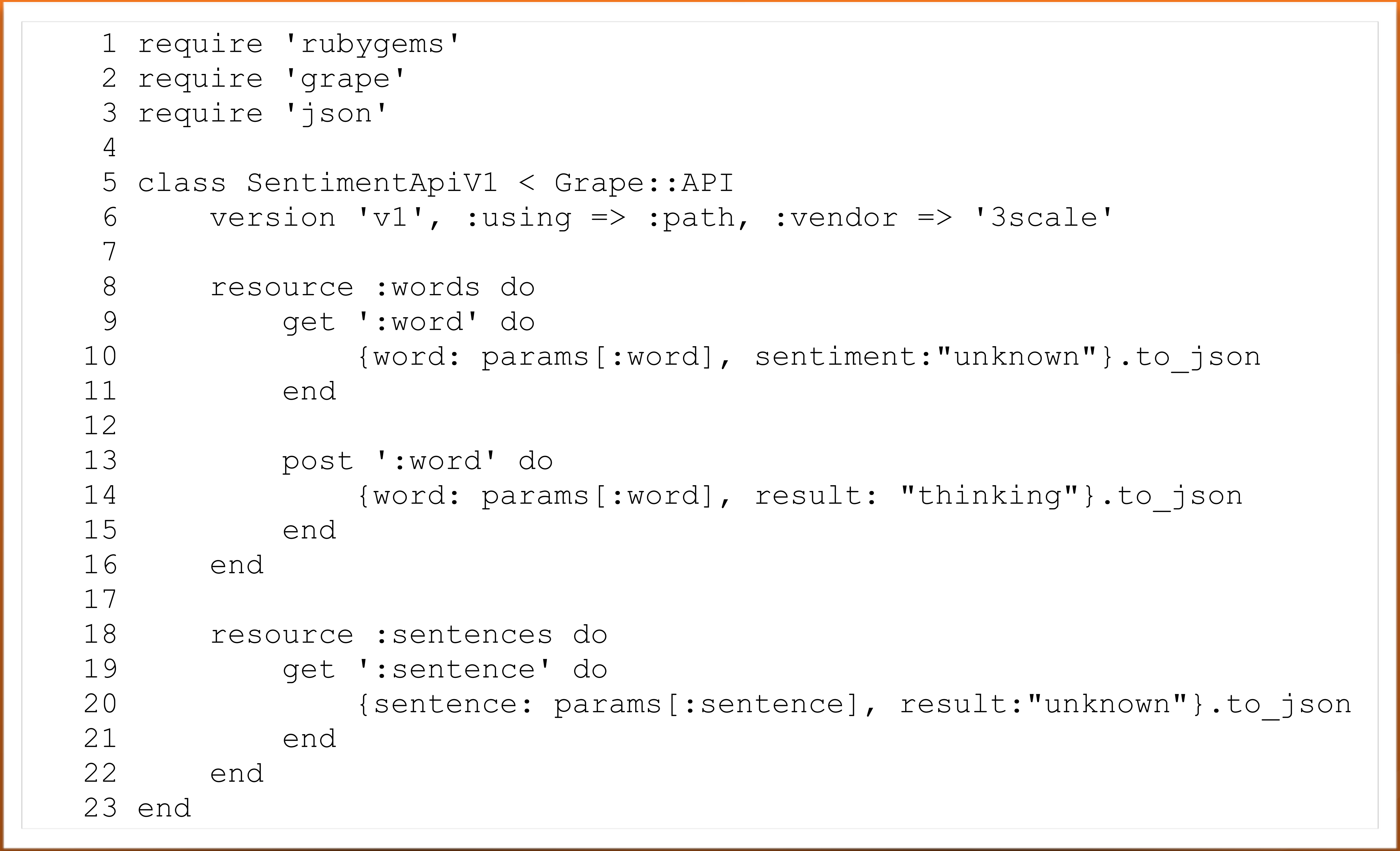
https://github.com/picsoung/sentimentAPI_workshop

Get started



```
git clone https://github.com/picsoung/sentimentAPI_workshop
cd ./sentimentAPI_workshop
git checkout 1-basic
bundle install
```

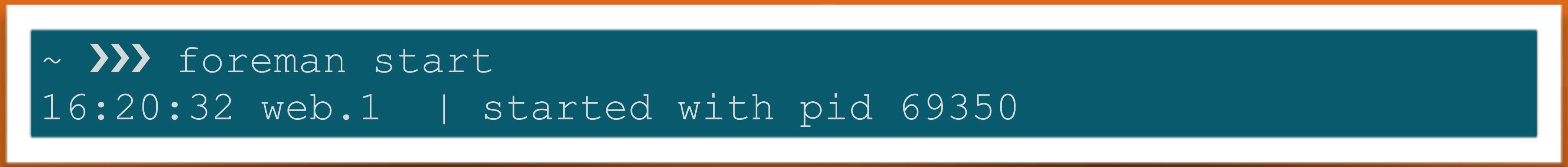
Get started



```ruby
require 'rubygems'
require 'grape'
require 'json'

class SentimentApiV1 < Grape::API
    version 'v1', :using => :path, :vendor => '3scale'

    resource :words do
        get ':word' do
            {word: params[:word], sentiment:"unknown"}.to_json
        end

        post ':word' do
            {word: params[:word], result: "thinking"}.to_json
        end
    end

    resource :sentences do
        get ':sentence' do
            {sentence: params[:sentence], result:"unknown"}.to_json
        end
    end
end
```

Get started

Let's launch it



```
~ ❯❯❯ foreman start
16:20:32 web.1   | started with pid 69350
```

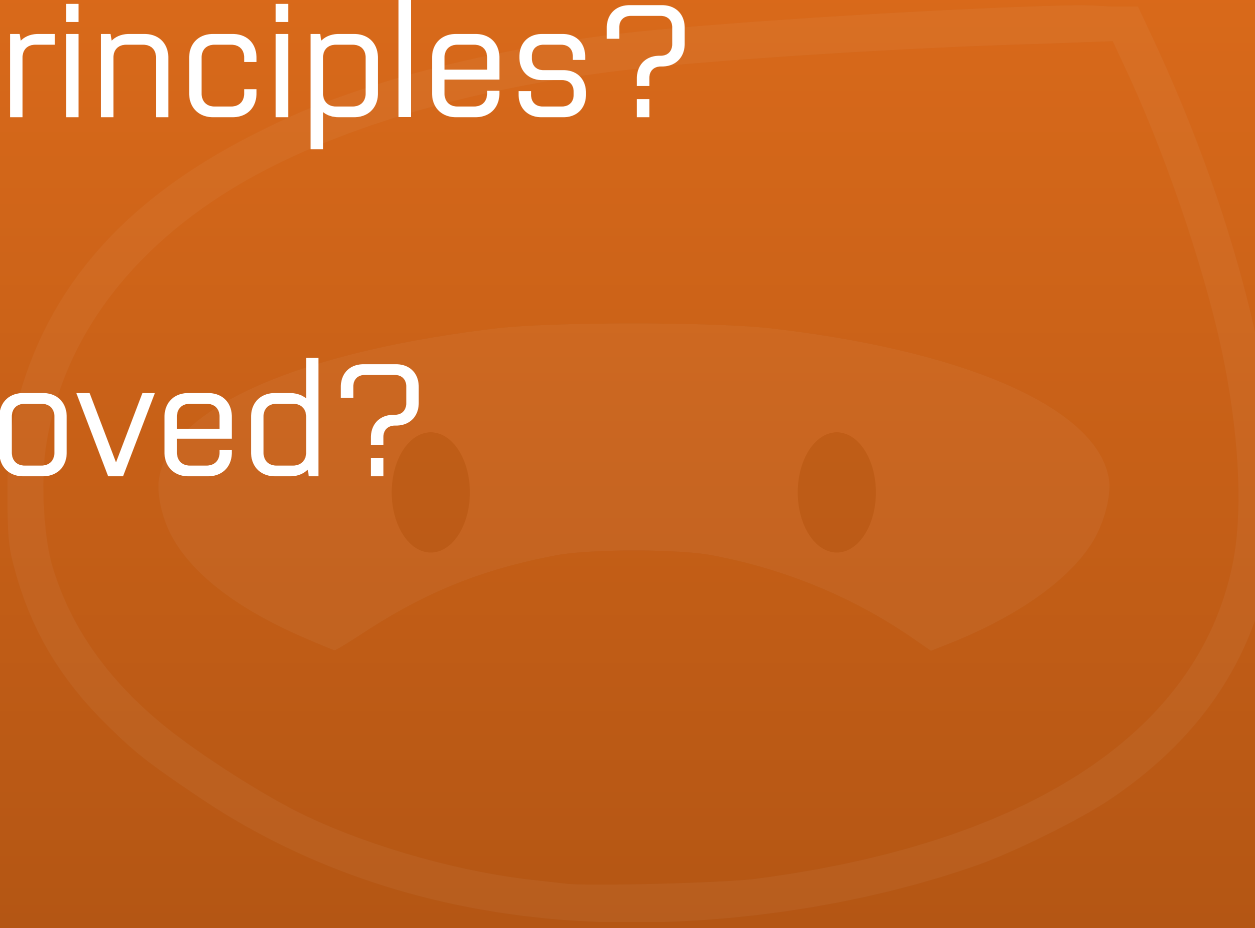
Test the API's endpoints with POSTman

Get started

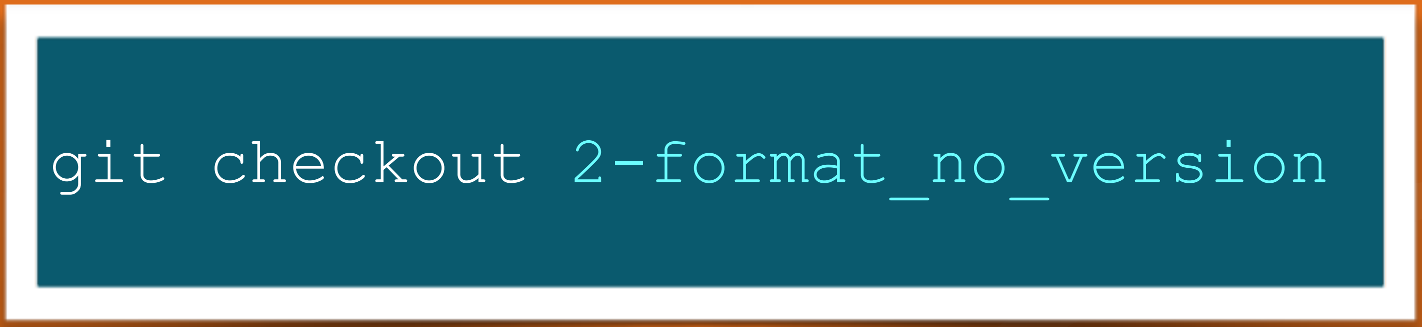
What's wrong?

Does it follow the principles?

What could be improved?



Improve



```
git checkout 2-format_no_version
```

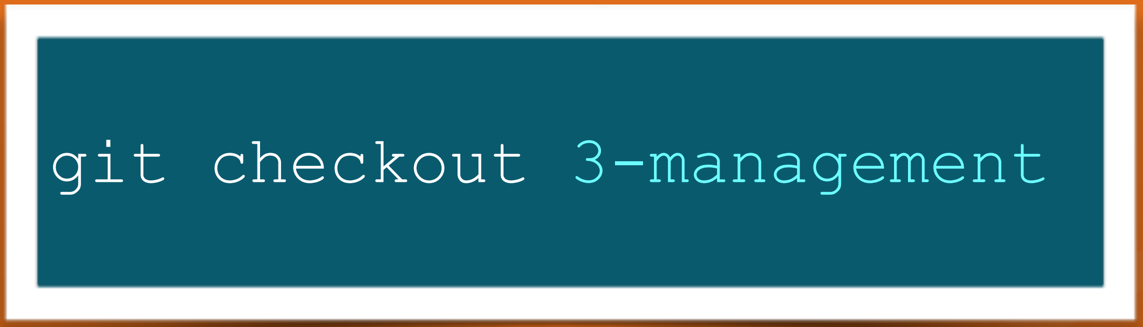
Improve

It works, what is missing?

How do we authenticate users?

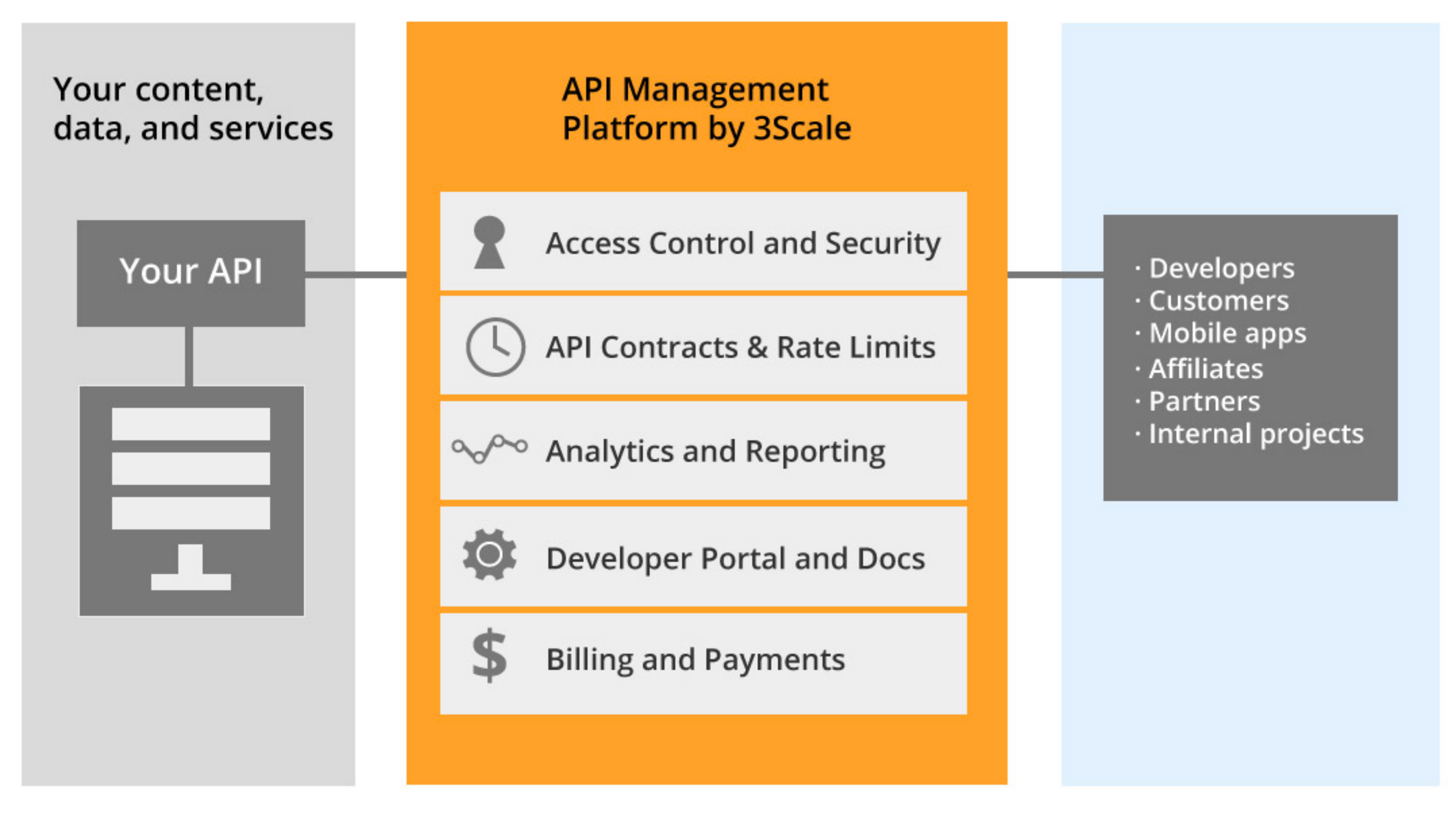
How do I limit the number of calls?

Management



```
git checkout 3-management
```

Management



Your content, data, and services

Your API

API Management Platform by 3Scale

Access Control and Security

API Contracts & Rate Limits

Analytics and Reporting

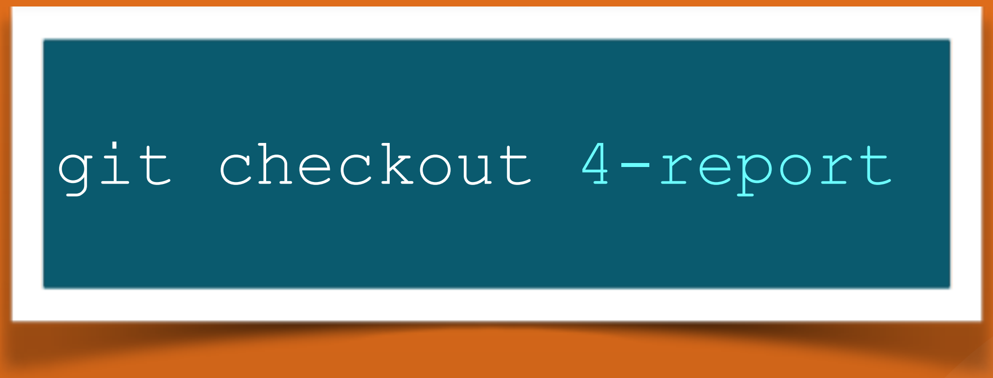
Developer Portal and Docs

Billing and Payments

- Developers
- Customers
- Mobile apps
- Affiliates
- Partners
- Internal projects

3SCALE

Management



```
git checkout 4-report
```

Deploy

Scalingo

easy deploy for apps
http://scalingo.com

```
git remote add scaling git@scalingo.com:YOURAPP.git
git push scaling 4-report:master
```

ONE DOES NOT SIMPLY
RELEASE THEIR API

Next?

Add some real logic and real data

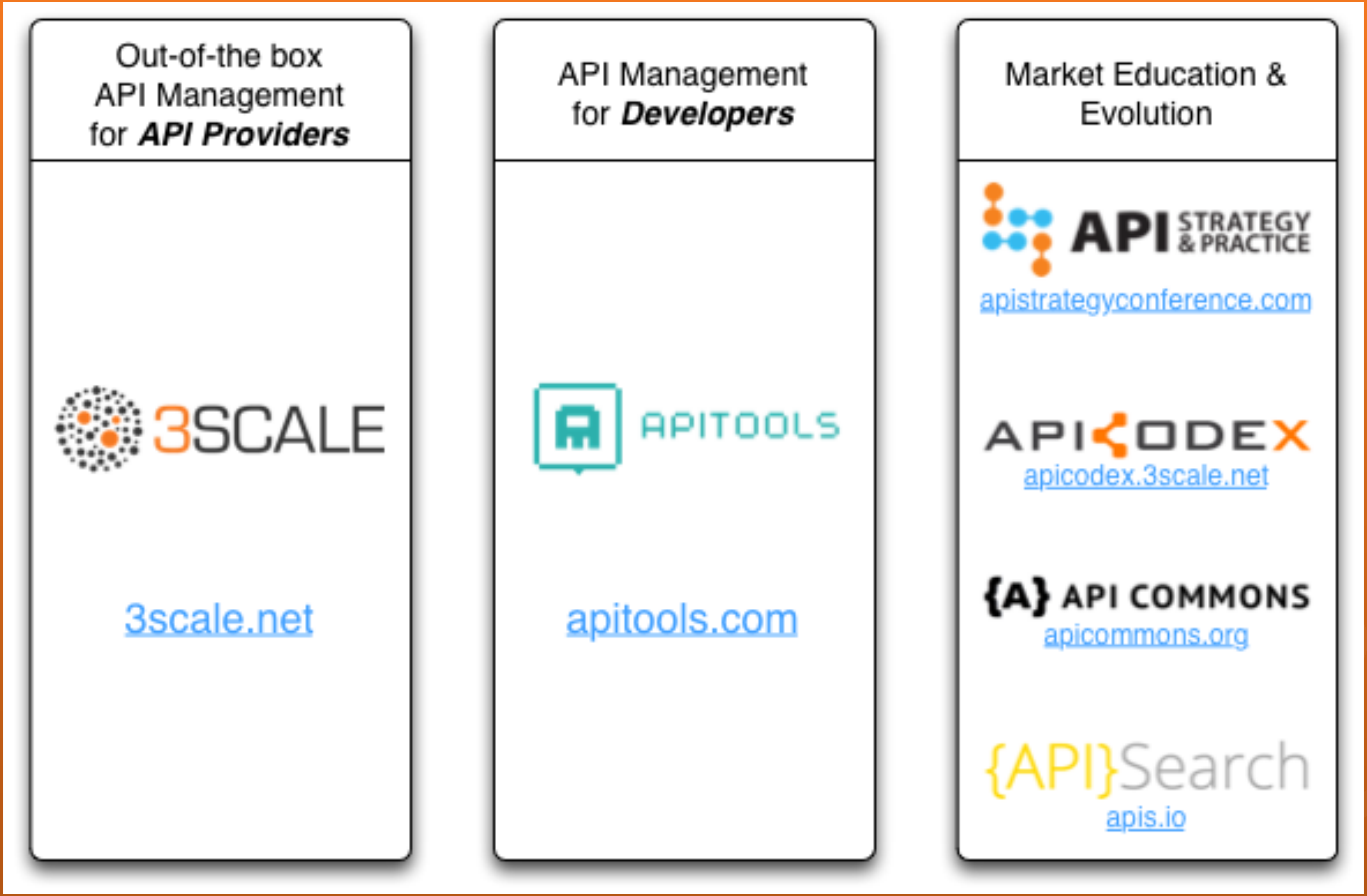
Add tests

Get user

Monetize it

Buy a house in Miami

Above & Beyond



Out-of-the box
API Management
for ***API Providers***

3SCALE

3scale.net

API Management
for ***Developers***

APITOOLS

apitools.com

Market Education &
Evolution

API STRATEGY & PRACTICE

apistrategyconference.com

APICODEX

apicodex.3scale.net

{A} API COMMONS

apicommons.org

{API}Search

apis.io

API design

now you know ;)

Nicolas Grenié

Hacker in Residence at 3scale.net
@3scale

API Workshop at LeWagon
Paris, February 12, 2015

3SCALE