Swagger

Implement Web API documents

Jiang Wu

2015-10-17

Context

About me

- 8 years experience of Ruby on Rails
- Sinatra advocator, minimalism
- Gave a talk at RubyKaigi 2010

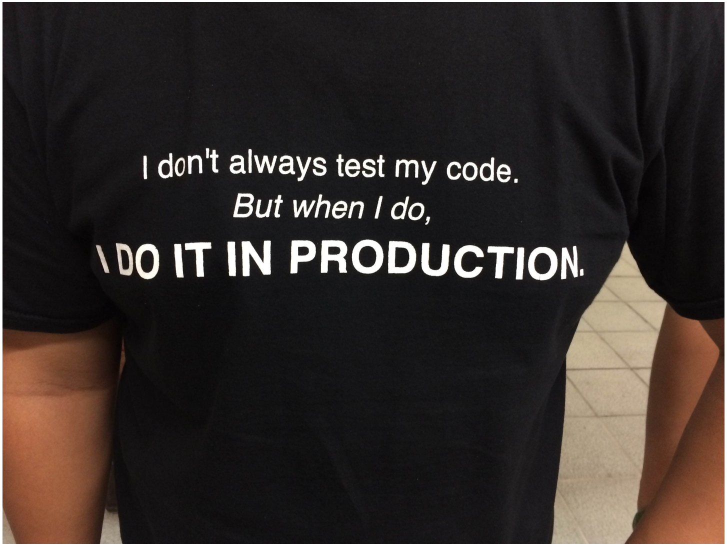
先讲两个笑话给大家提神

程序员最痛恨两件事

- 写文档
- 别人不写文档

Programmers hate 2 things: writing documentation and no documentation.

I don’t always test my code...



Solution

'D'o not 'R'epeat 'Y'ourself Principle

Scenario (Nov. 2014)

- Web 2.0 application start from 2011
- 3 types of API consumers
  - Native application (iOS/Android/PC/Mac)
  - Third party service
  - Javascript: single page application

Constraints 约束

- HTTP/1.1
- JSON data format (for JavaScript)
- REST architecture style

Separate 独立

Separate API from Web application.

- API specific
- ~~Cookie Session~~
- Stateless

API documentations

Must be important, clear and accurate.

- **重要** 跨项目，跨公司
- **清楚** 减少交流成本
- **正确** 和代码保持同步

Write by hand?

- Huge effort
- Not up to date
- CSS?

Context

Demo

https://api.gitcafe.com/apidoc/

Definitions

Swagger

Describe REST services

Swagger UI

Live testable documentation

grape-swagger

Generate API description

Problems solved by Swagger

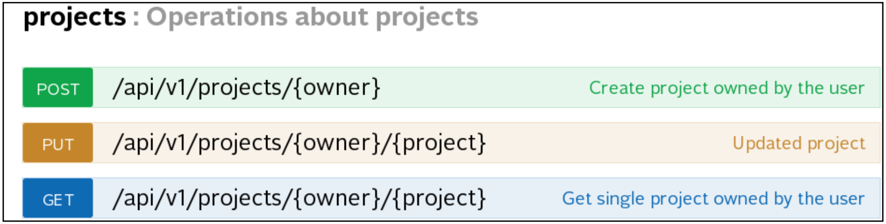
- Communication between different teams/companies
- Synchronization of code and documentation.
- ~~Test~~ Verification of REST services

Describe REST services

- API routes
- Input types
- Output types
- Authorizations

API routes

- namespaces
- paths
- HTTP verbs
- documentations

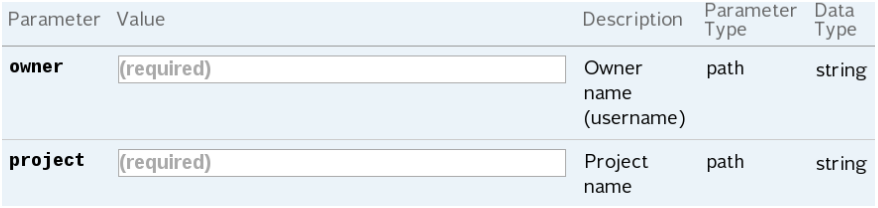


projects : Operations about projects

POST	/api/v1/projects/{owner}	Create project owned by the user
PUT	/api/v1/projects/{owner}/{project}	Updated project
GET	/api/v1/projects/{owner}/{project}	Get single project owned by the user

Input types

- name
- type
- default value
- required/optional
- description



Parameter	Value	Description	Parameter Type	Data Type
owner	(required)	Owner name (username)	path	string
project	(required)	Project name	path	string

Benefit of input types

Code

- Validation of parameters

Documentation

- Description
- Validation of form inputs

Output types

- name
- description
- type
- referencing

Benefit of output types

Code

- Serialization of values

Documentation

- Schema of output

HTTP Response

- code
- message
- output type

Authorizations

Support "basic", "apiKey" and "oauth2".

- scopes
- grant types
- login endpoint
- token endpoint

Short summary of Swagger

- API routes
- Input types
- Output types
- Authorizations

Live Test

Similiar as doctest of Python

```
""" input and output, REPL way
>>> factorial(5)
120
"""
def factorial(n):#Implementation
```

Documentation is both sample and test.

API test before Swagger

- Write it by yourself
- Browser plugins (not quite works)
  RestClient Firefox + Chrome
- Proprietary softwares
  Postman Chrome + NodeJS
  Paw Mac

Swagger clients

- REST clients are not Swagger specific
- Can generate with code generator
- Dynamic code generation for JavaScript/Ruby/Python

```
Swagger.new(endpoint, token).
  ns("account").nick("get_info").
  get(username: "gitcafe")
```

Pitfall

Grape(Ruby) supports wildcard(*) path, while Swagger UI(JS) didn't, and we figured out the bug was in grape-swagger(Ruby).

Context

Swagger and micro services

- Language agnostic
- Testable documentation
- Clients are easy to write
- Service/Documentation hub

Unleash power of database

API	Database
Routes	Tables/Views
Input types	Constraints, Types
Output types	Table columns
Authorizations	Roles

PostgREST

Can we use the declarative information in a relational db schema to mechanically generate an HTTP API?

- *begriffs*

Why PostgREST?

- No server code (but database code)
- Counter attack to NoSQL
- Bare metal speed (written in Haskell)
- HTTP protocol

REST API specifications(1)

- no specification
- WADL
- HATEOAS(Hypermedia as the Engine of Application State)
- HAL

REST API specifications(2)

- Swagger
- RAML
- Blueprint
- ALPS

Apply DRY principle

- Abstract not duplicate
- Select proper tools
- Bring values

Context

Swagger

- Describe REST services
- Provide Live testable documentation with Swagger UI
- Can generate from source code
- Disadvantage: have to investigate across components when debugging

Q&A

My WeChat QR Code