SOA proliferation through specification

James Earl Douglas

MediaWiki Developer Summit 2015

Thesis

Service development and consumption flourishes in a well-specified ecosystem.

Agenda

- Background
- Examples
- Roadmap

Background

- Service-oriented architecture
- Specification
- Specification formats
- Swagger

Service-oriented architecture

Feature stew

See also: the join calculus

Composable units of functionality

- `foo`
- `bar`
- `baz = foo ∘ bar`

Specification

Establish boundaries

- Limit the set of features
- Limit the scope of each feature

Describe the interface

"*To foo a bar, you must provide a baz, and you will receive a qux.*"

Specification formats

- IETF RFC
- MW RFC
- WSDL
- WADL
- RAML
- API Blueprint
- Swagger

Swagger

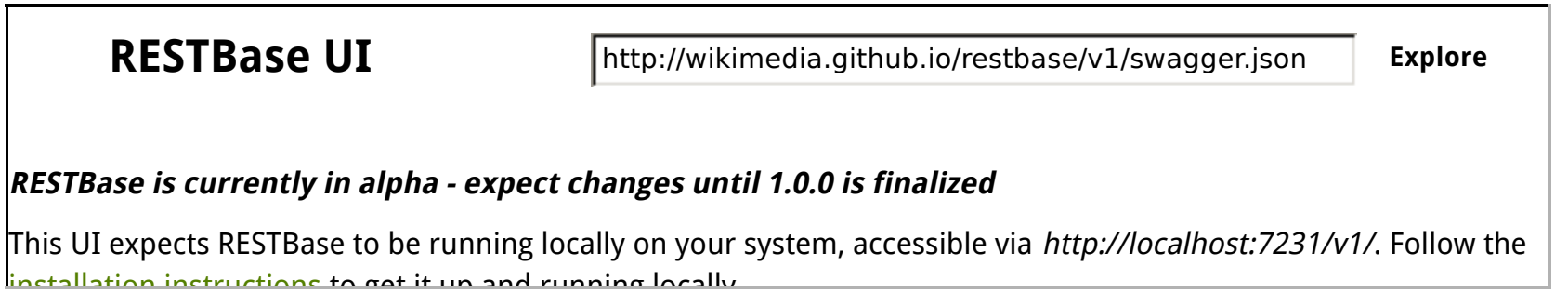

```
{
  "swagger": "2.0",
  "info": {
    "version": "1.0.0",
    "title": "RESTBase",
    "description": "Distributed storage with REST API & dispatcher for backend service",
    "termsOfService": "https://github.com/wikimedia/restbase#restbase",
    "contact": {
      "name": "Services",
      "email": "services@lists.wikimedia.org",
```


http://wikimedia.github.io/restbase/v1/swagger.json

Examples

- Swagger UI
- Swagger static documentation
- Swagger client generation
- Test automation

Swagger UI



http://wikimedia.github.io/restbase/

Swagger static documentation

RESTBase

Distributed storage with REST API & dispatcher for backend service for our Partner
More information: https://www.mediawiki.org/wiki/Services
Contact Info: services@lists.wikimedia.org
GNU Affero
http://opensource.org/licenses/AGPL-3.0

http://wikimedia.github.io/restbase/v1/

Swagger client generation

```
var httpsync = require('httpsync');
var codegen  = require('swagger-js-codegen').CodeGen;
var fs       = require('fs');

var specUrl  = 'http://wikimedia.github.io/restbase/v1/swagger.json';
var response = httpsync.get(specUrl).end();
var swagger  = JSON.parse(response.data.toString());

var clientJs = codegen.getNodeCode({ className: 'RESTBase', swagger: swagger });
fs.writeFileSync('client.js', clientJs);
```

Swagger-generated client

```
var RESTBase = (function() {
    RESTBase.prototype.listRevisions     = function(parameters) { // ...
    RESTBase.prototype.getLatestFormat   = function(parameters) { // ...
    RESTBase.prototype.getFormatRevision = function(parameters) { // ...
    RESTBase.prototype.getFormatRevision = function(parameters) { // ...
    // ...
})();

exports.RESTBase = RESTBase;
```

Test automation

Swagger is extensible

- Augment spec with request/response pairs
- Use them generated documentation
- Use them in automated tests

```
"x-amples": [
  {
    "request": {
      "params": {
        "domain": "en.wikipedia.test.local",
        "title": "Foobar"
      }
    },
    "response": {
      "status": 200,
      "headers": {
        "content-type": "text/html;profile=mediawiki.org/specs/html/1.0.0"
      }
    }
```

```
    }
]
```

Roadmap

- Where we're going
- How we'll get there

Where we're going

Happy users

Features delivered as desired.

Excited developers

Empowered to build and deliver awesomeness.

Rainbows and ponies for all

OMG!! Ponies!!

How we'll get there

Q) How do we stay on track?

A) Specify it.

Spec has to be reliable

- v1 can change within in the philosophy of SemVer
- v2 can change anything it wants

Spec-first vs. spec-last

- Hitting the ground running vs. blocking by backend
- Deliberate vs. accidental

References

- Semantic Versioning
- Design by contract
- Swagger
- swagger-js-codegen
- You are what you document

Discussion

- How do you document your APIs?
- How do you enforce your specifications?
- Does your documentation catch up to your code?
- Does your code catch up to your documentation?
- What are your favorite Web services?
- What are your favorite APIs?