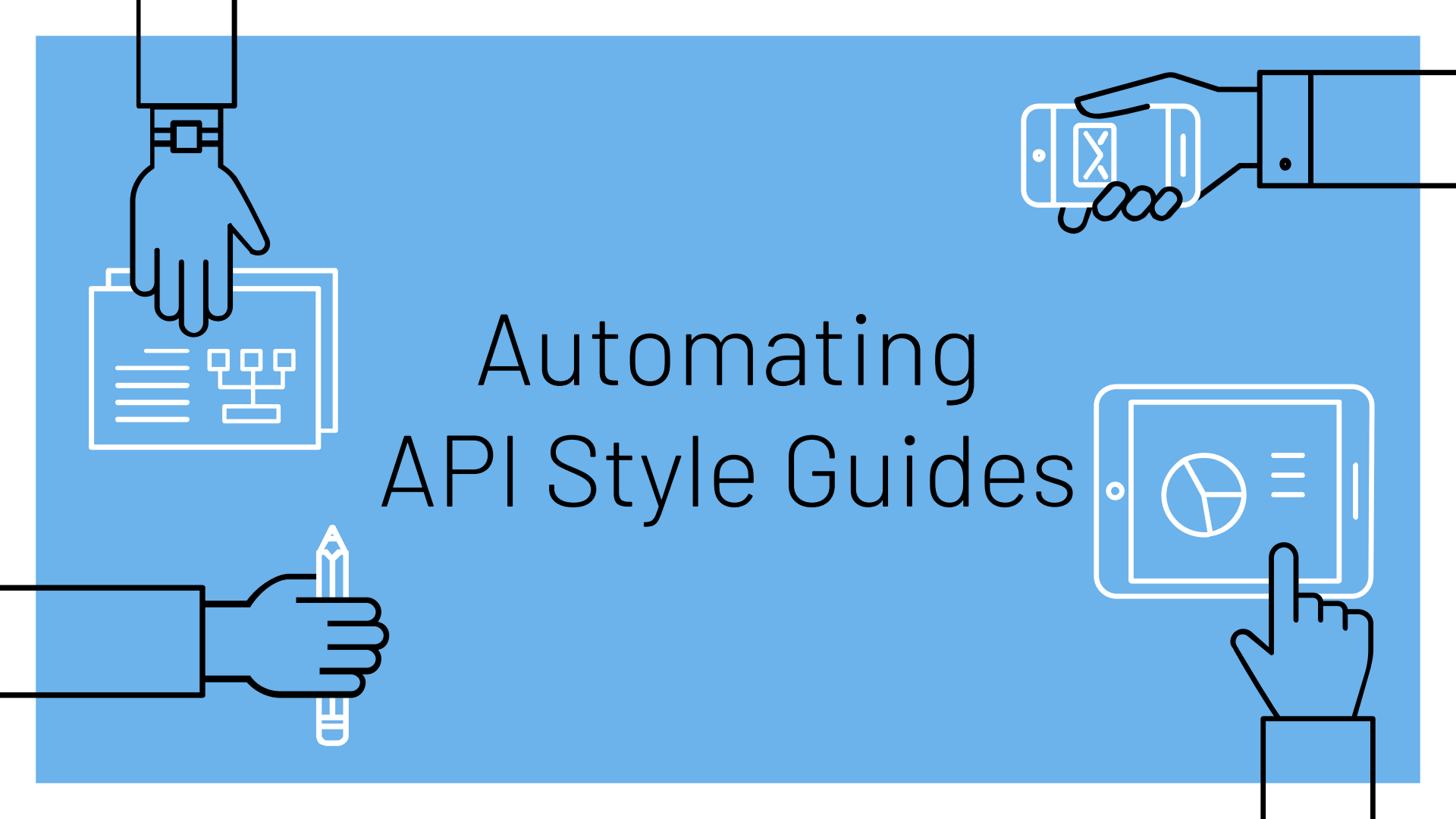
# Automating API Style Guides



# HELLO!

## I am Phil Sturgeon

I love to talk about APIs, crashing bikes, and saving the planet.

You can find me at @philsturgeon

Stoplight

Use Cases Pricing Enterprise Learn Open Source Log in Get Started

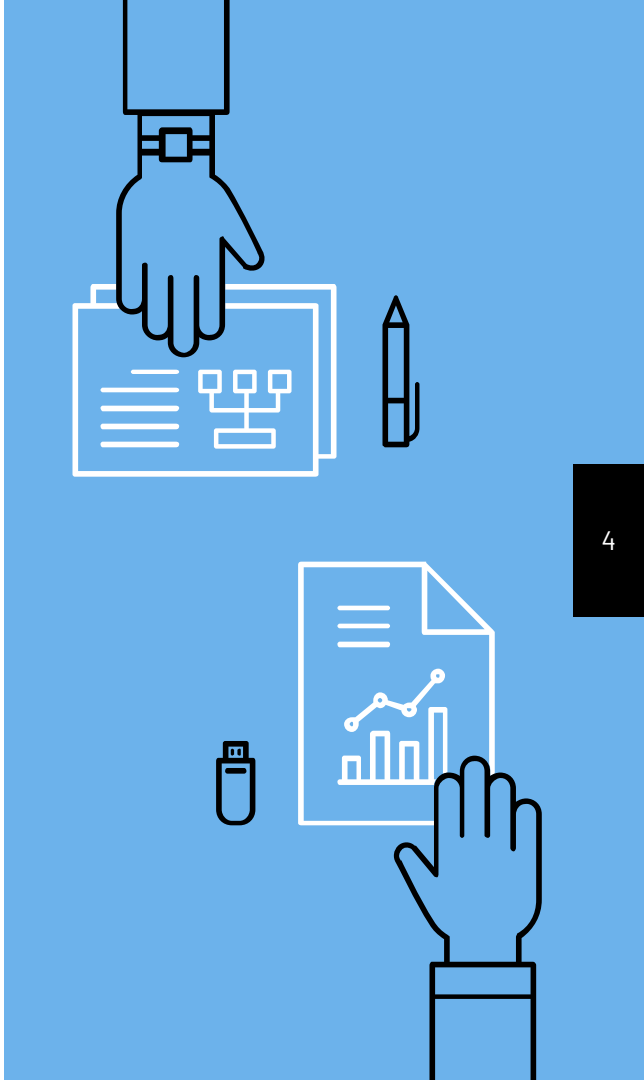
# Build Quality APIs Faster

Supercharge your API lifecycle with best-in-class tooling that integrates seamlessly into your Git workflows.

Get Started For Free

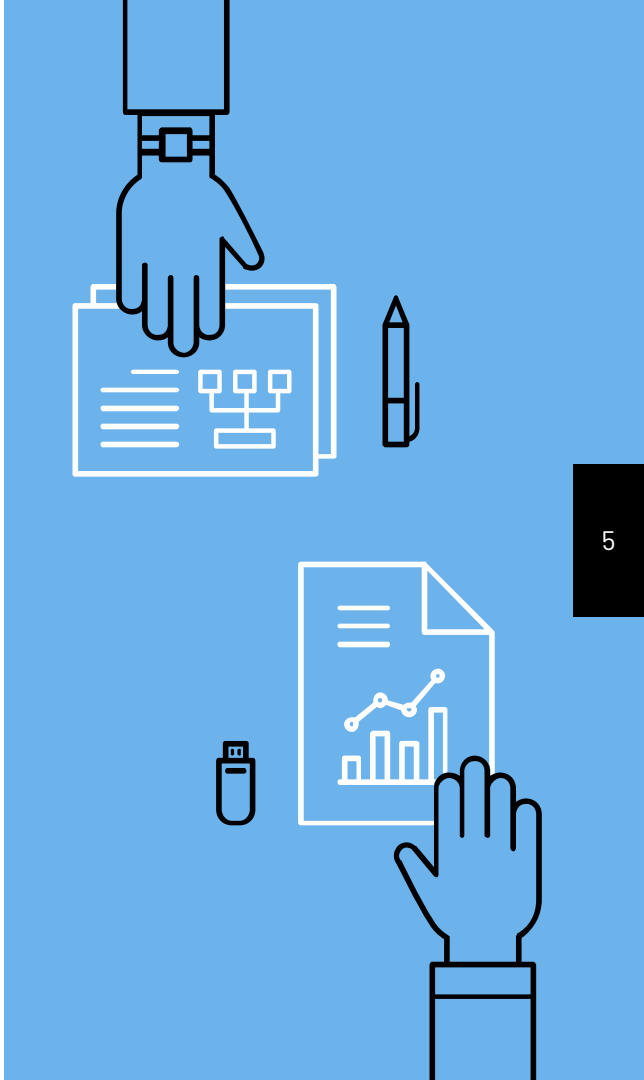
# Inconsistency Isn't Great

1. rando_Naming-conventions
2. Different data formats
    a. HAL
    b. JSON:API
    c. Custom Snowflake
    d. Differently Custom Snowflake
    e. Differently Custom Snowflake v2
3. Different security schemes
4. Unconventional use of status codes



# Inconsistency Wastes Time

1. Assumptions cause mistakes
2. Constantly rechecking docs
3. Cannot reuse generic code
4. Looks silly to external users
5. Bad DevEx in general

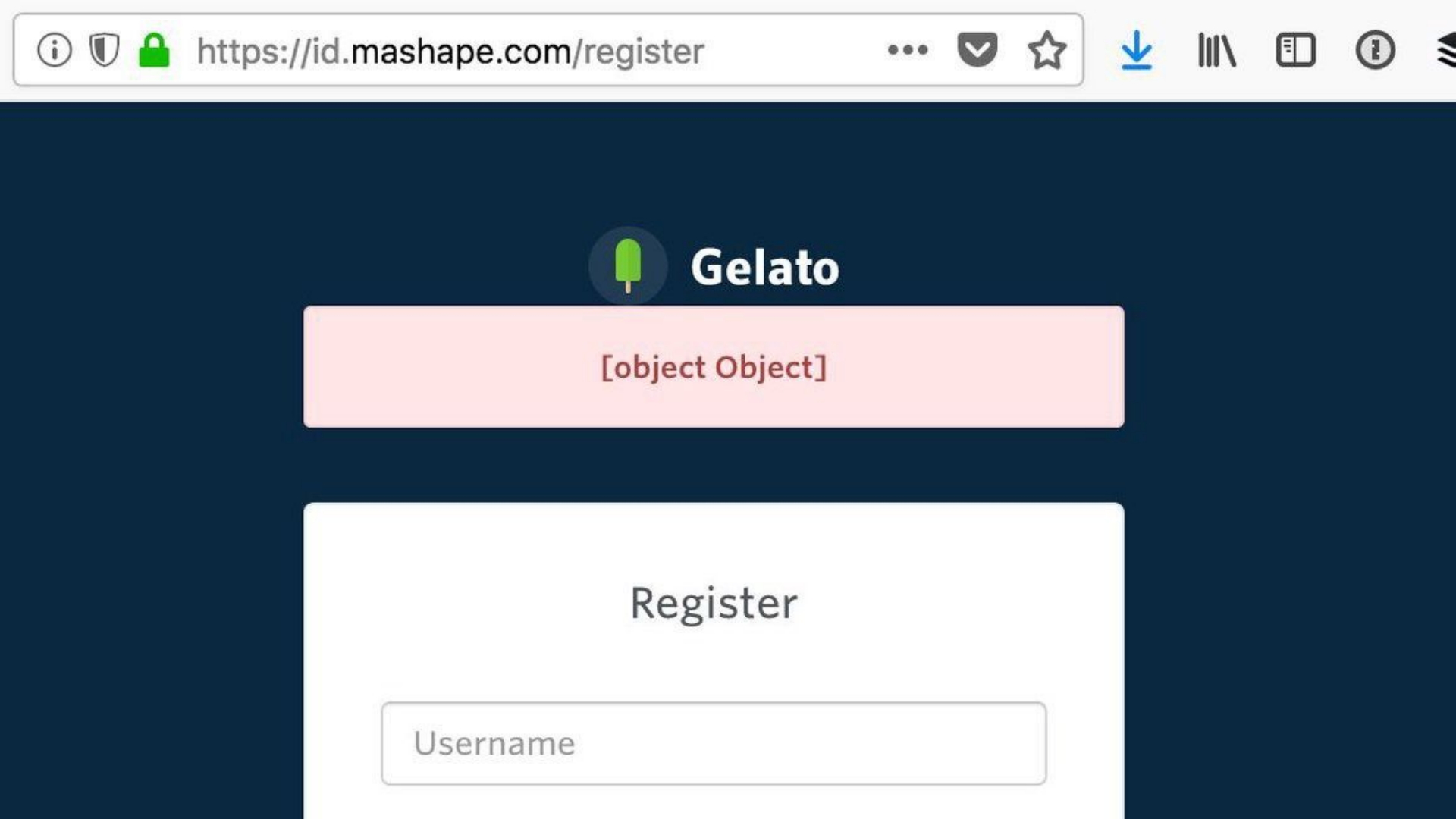


API A

```
{
  "error": "A thing went wrong"
}
```

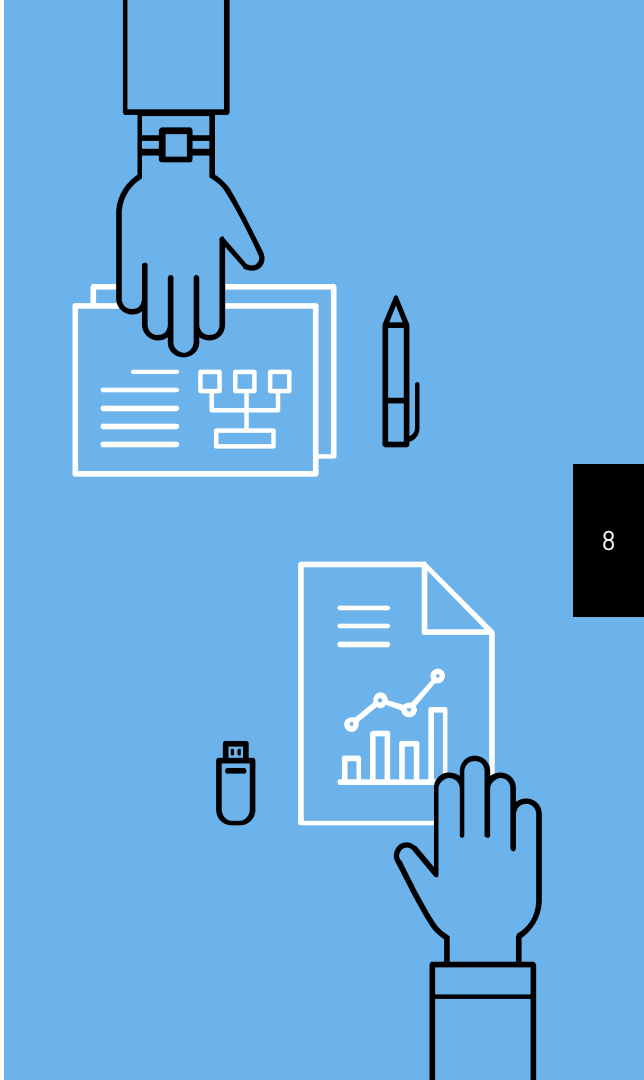
API B

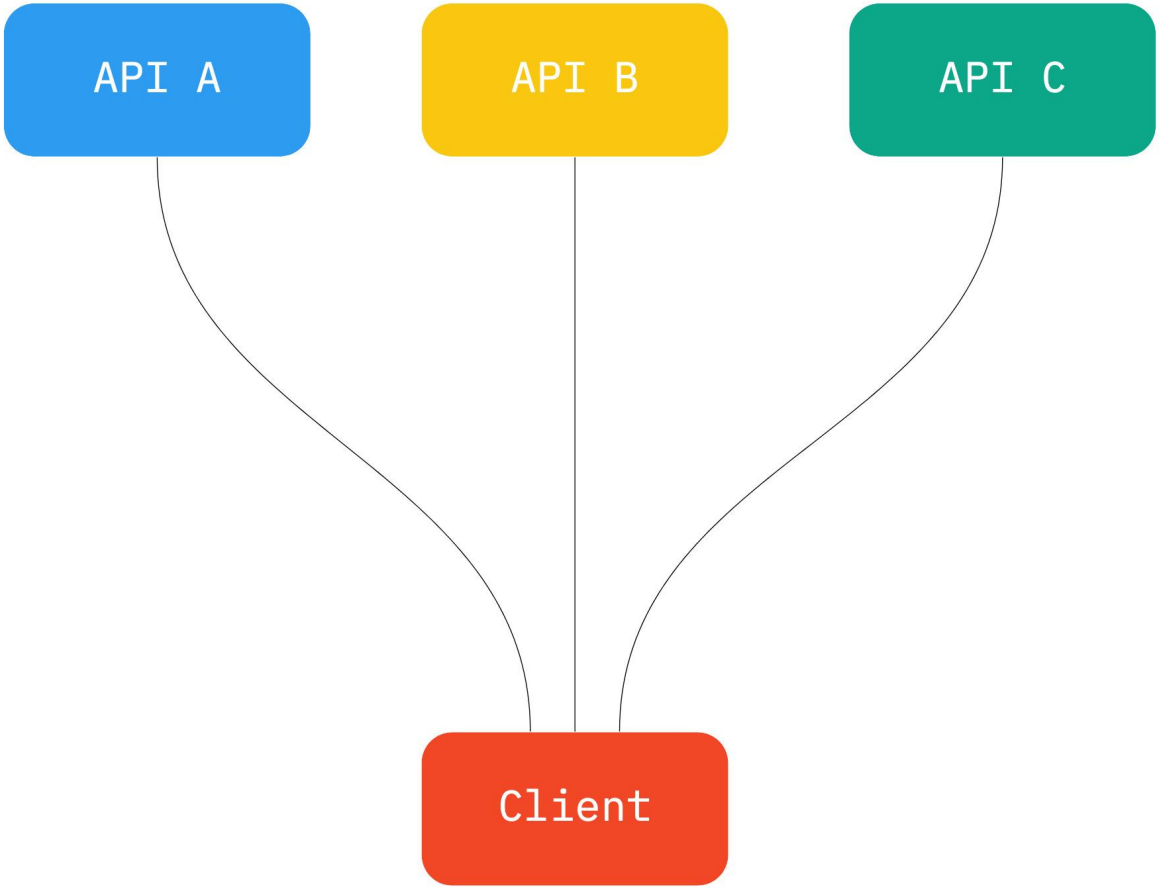
```
{
  "error": {
    "code": "100110",
    "message": "A thing went wrong"
  }
}
```

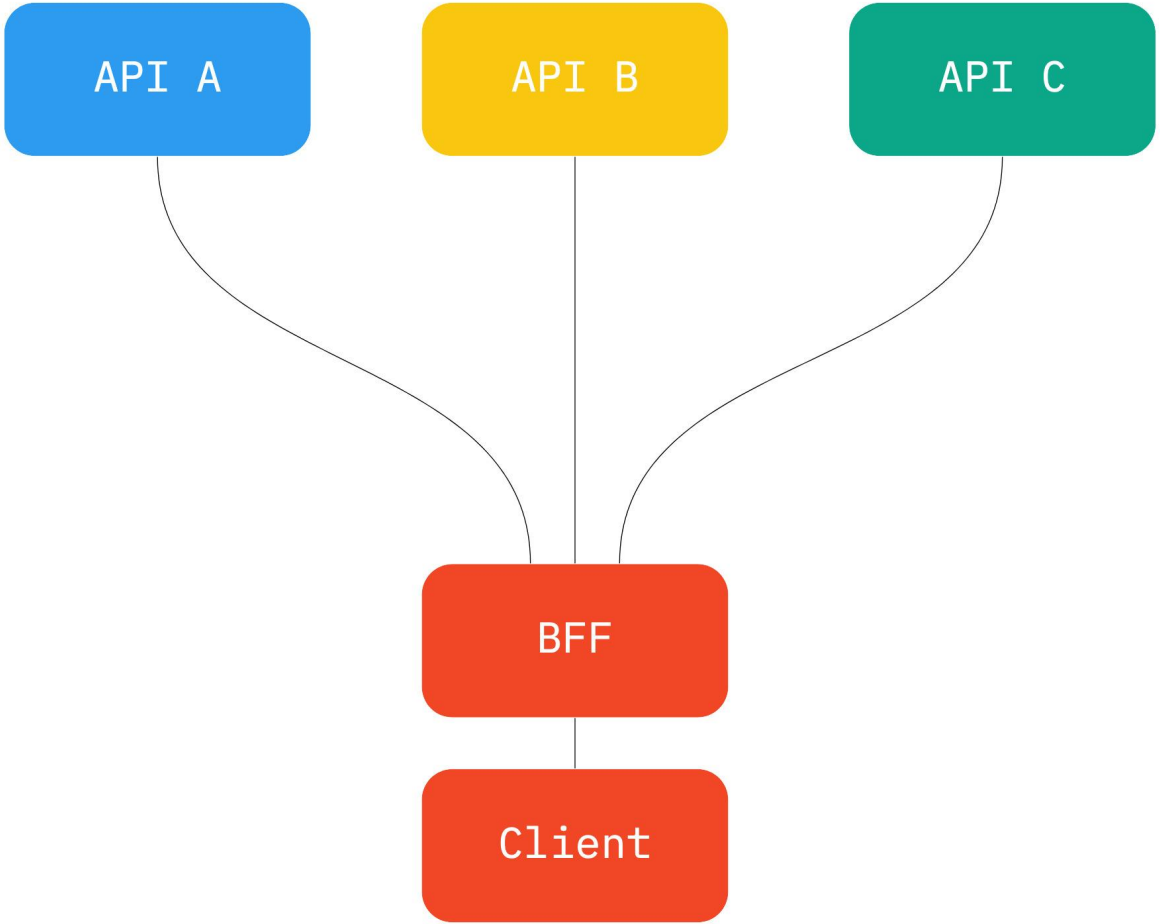

https://id.mashape.com/register
Gelato
[object Object]
Register
Username

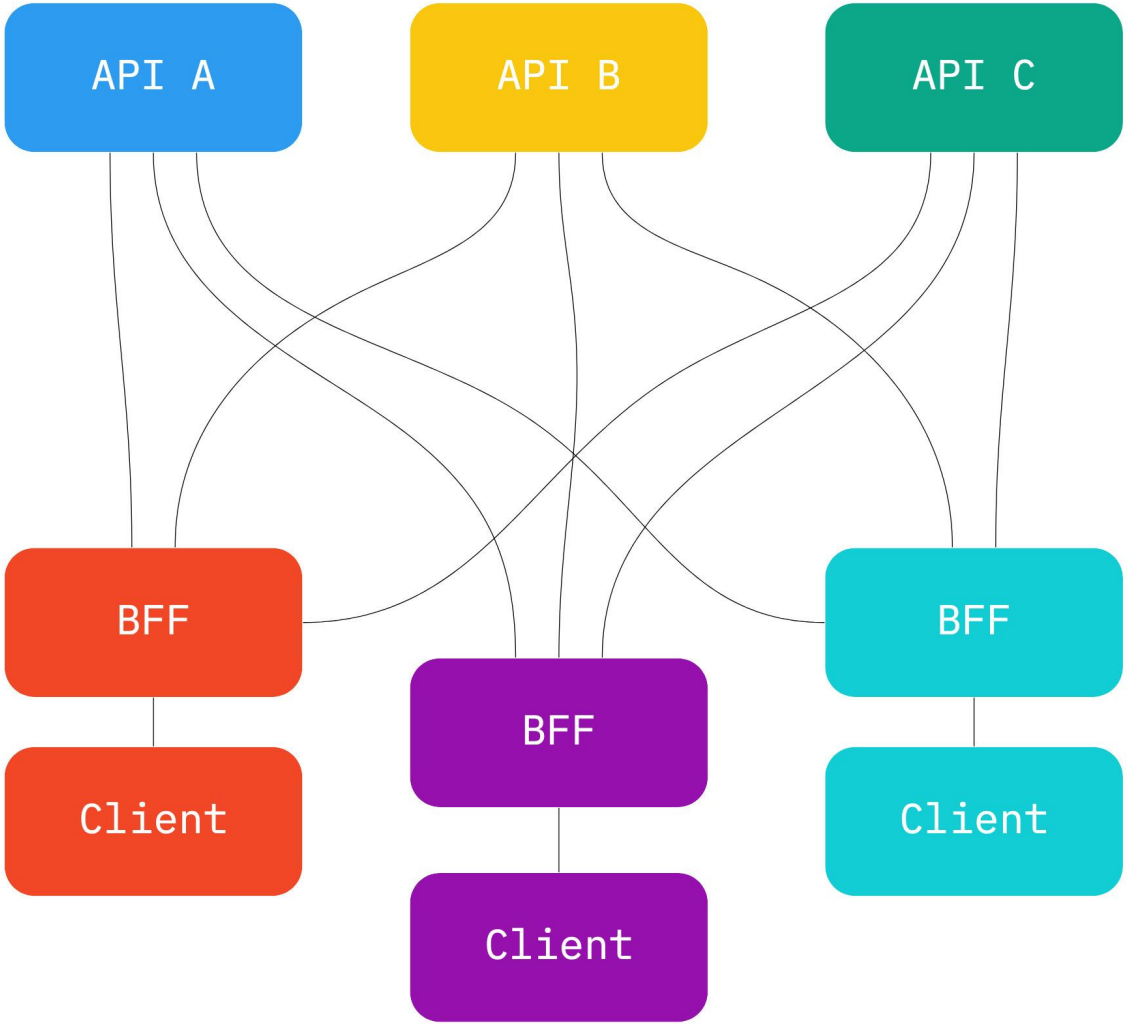
# Standardizing Production

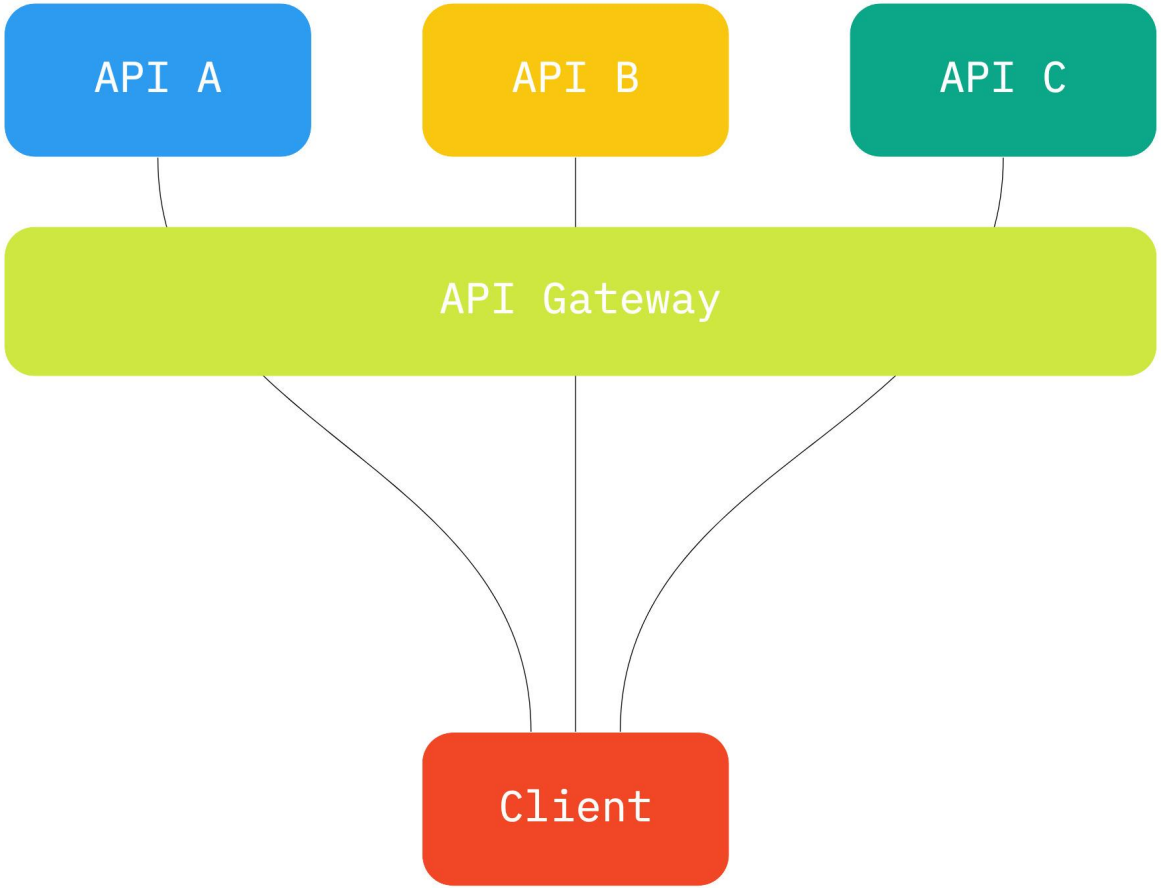
- Make a v2 API?
- Duplicate endpoints via Evolution?
- Build Backends-for-Frontends?
- Start from scratch?
  - GraphQL
  - gRPC
  - Vulcain
  - Twirp









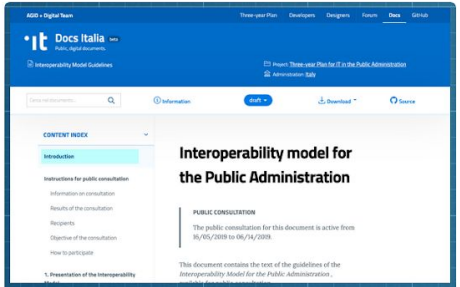


http://apistylebook.com/

# Design Guidelines

Some companies and government agencies share their API Design Guidelines with the community.

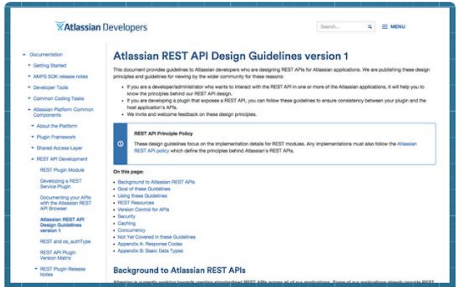
Each document has been analyzed to list covered topics and their references within the document.




## Interoperability model for the Public Administration

Agenzia per l'Italia Digital & Team per la trasformazione digital (Italian administration)

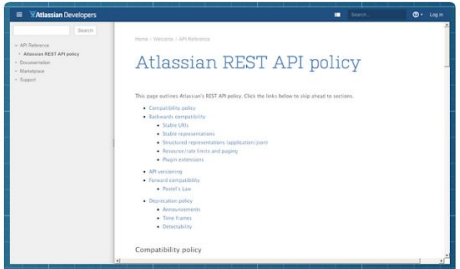
Explore




## Atlassian REST API Design Guidelines version 1

Atlassian

Explore




## Atlassian REST API Policy

Atlassian

Explore

# API Style Guides

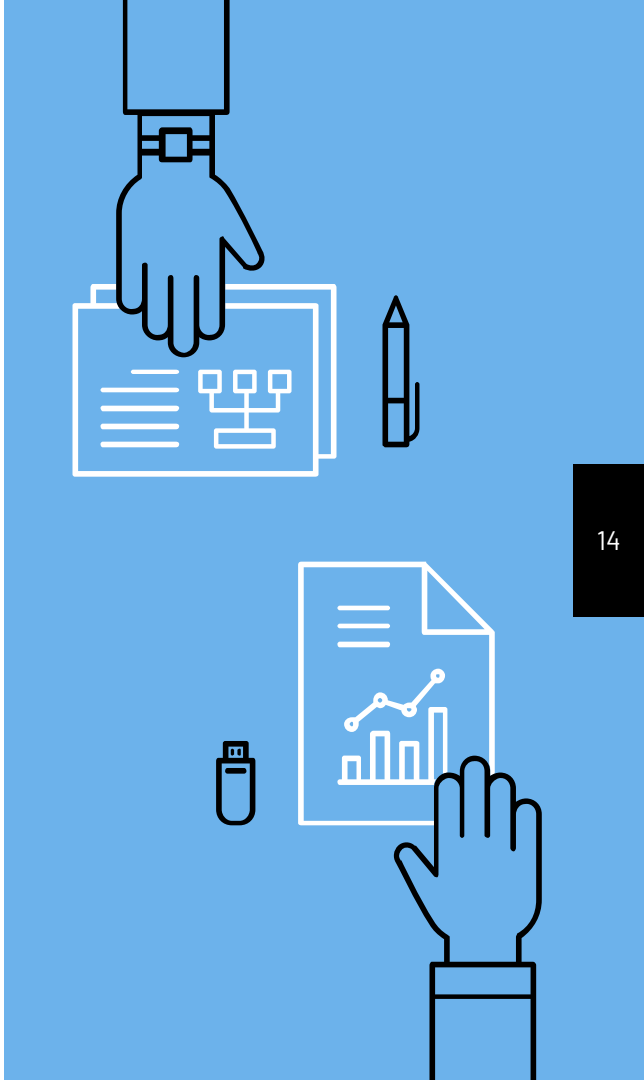
“API Design Guide”

“API Design Guidelines”

“API Style Guide”

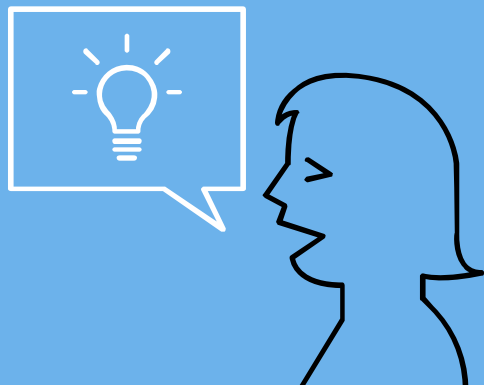
“API Stylebook”

Same thing! 😎



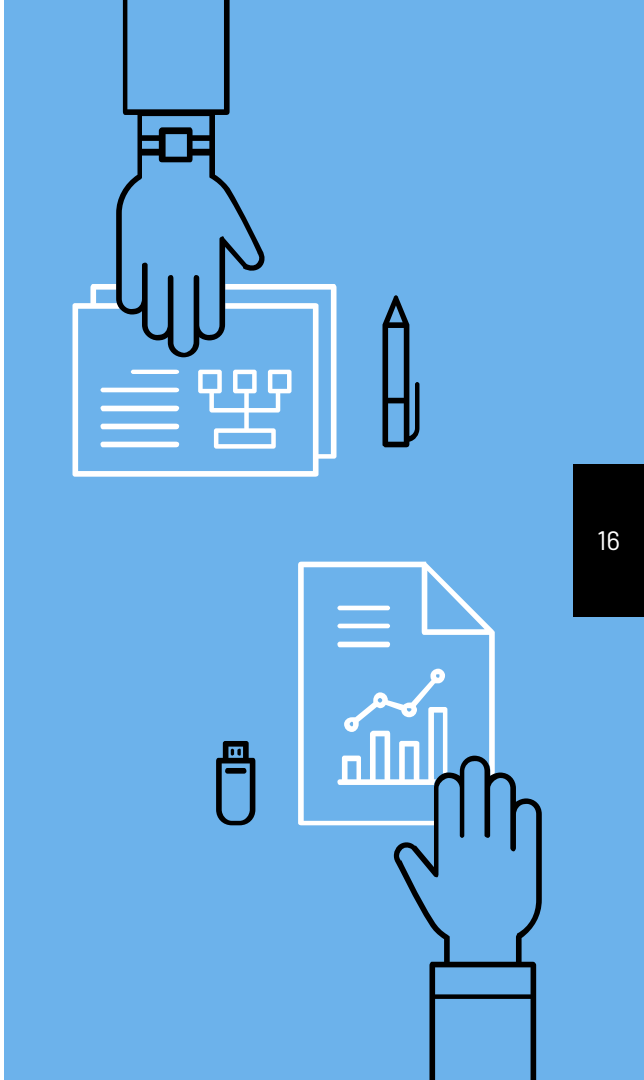
*Your goal should be to advise teams designing APIs toward a more consistent API with other APIs across the organisation.*

***James Higginbottom***



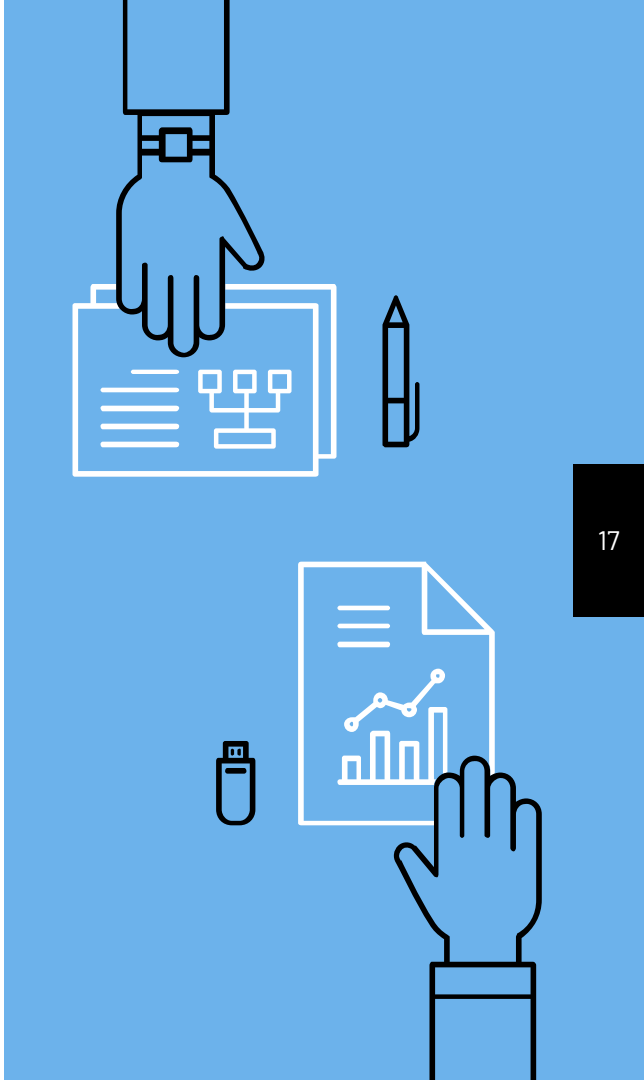
# Ideas for Rules

- Use RFC 7807 for errors
- Use UUID for IDs in URLs or body
- Ban HTTP Basic
- Must have “GET /” for HATEOAS
- Must have “GET /health”
- POST must return HTTP 201 or 202
- No errors returning HTTP 200



# API Style Guide Mediums

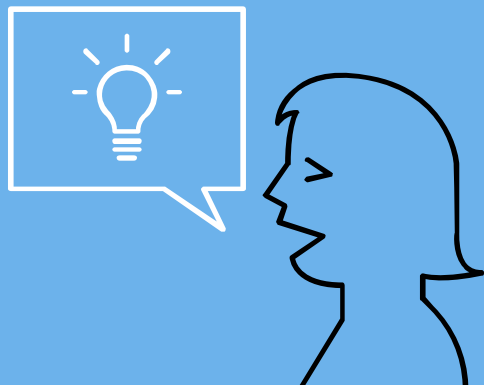
- Static Website / CMS / Wiki
- Google Slides / Keynote
- Random PDFs
- Spreadsheets?!



Most API devs aren't going to read an organization API manifesto.

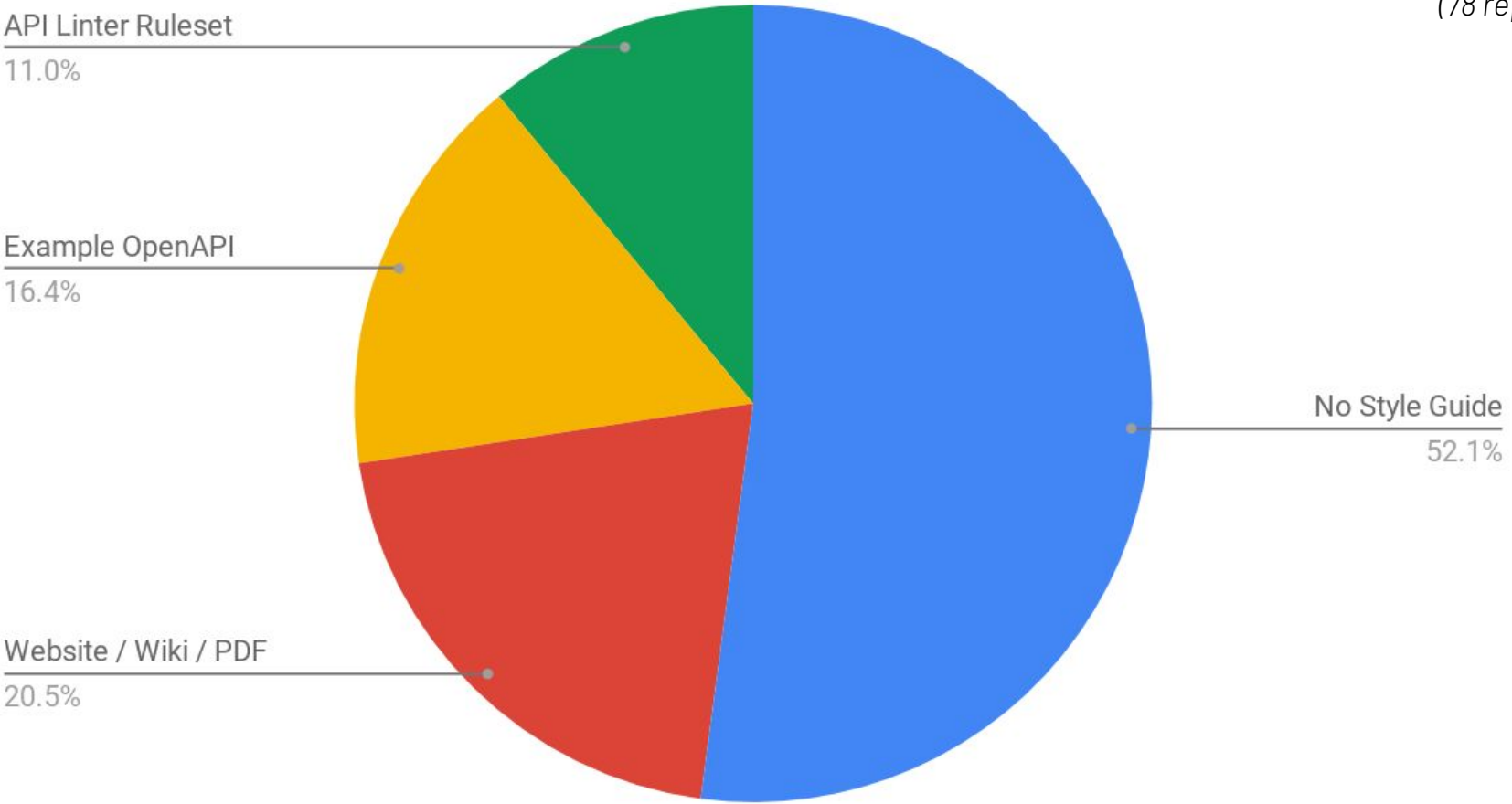
If they do they won't remember it.

If they do they won't re-read it looking for changes.



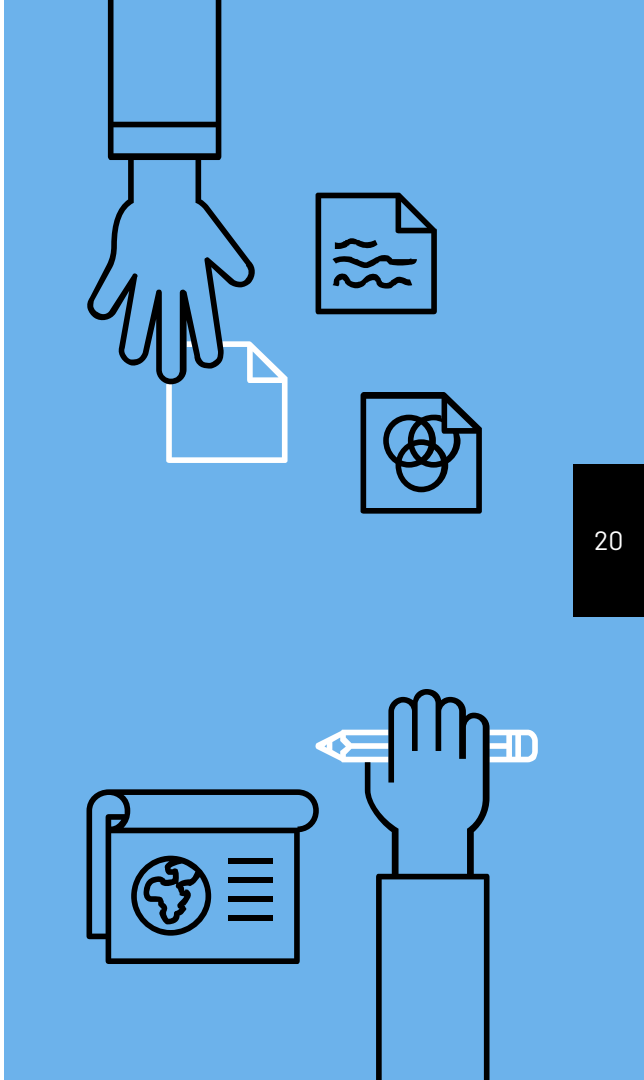
# Do You Have API Style Guides?

*Highly Scientific Twitter Poll*
*July 1st 2020*
*(78 replies)*

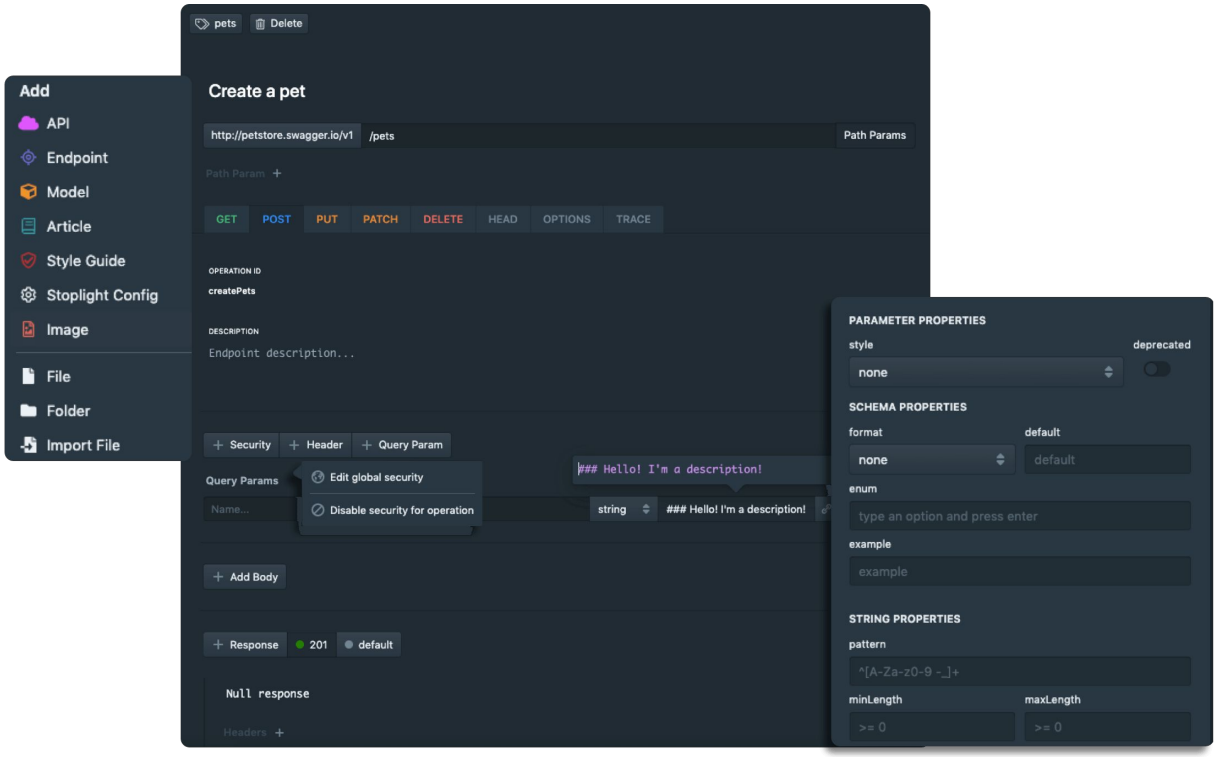


# Types of Rule

| Create Better OpenAPI | Create Better APIs |
| --- | --- |
| Consistent Docs | Consistent APIs |
| Better docs | Naming conventions |
| Better mocks | Security |

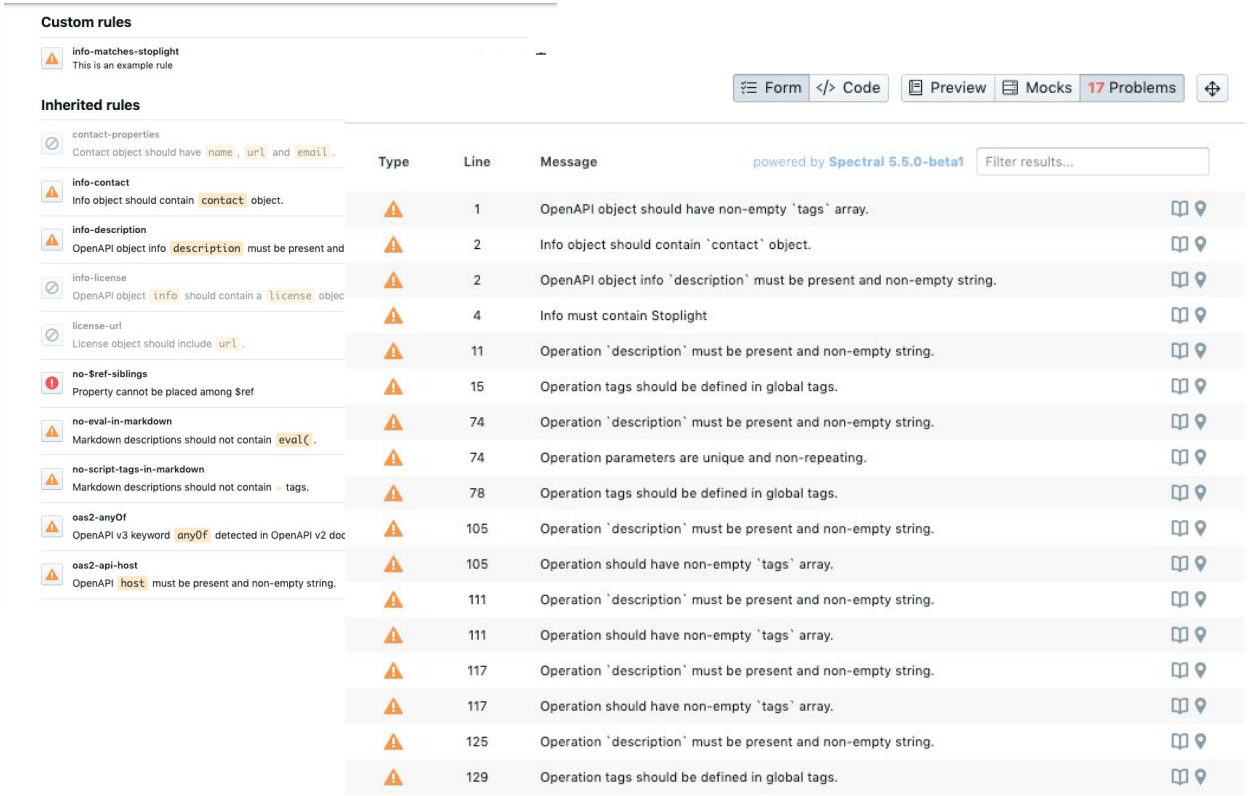


# Stoplight Studio




Avoid messing with OpenAPI YAML/JSON directly with a shiny GUI 👍

# Spectral in Stoplight Studio




Studio offers a generic, tweakable OpenAPI ruleset.

Get feedback on your OpenAPI as you go.

```
api-home:
  description: APIs MUST have a root path defined (`/`).
  severity: error
  given: $.paths
  then:
    field: /
    function: truthy
```

```
api-home-get:
  description: APIs root path MUST have a GET defined.
  severity: error
  given: $.paths[/]
  then:
    field: get
    function: truthy
```

```
paths-kebab-case:
  description: Should paths be kebab-case.
  message: '{{property}} should be kebab-case (lower case and
separated with hyphens)'
  severity: warn
  given: $.paths[*]~
  then:
    function: casing
    functionOptions:
      type: kebab
      separator:
        char: /
        allowLeading: true
```

```
# Authors: Lorna Mitchell and @mheap
semver:
  severity: error
  recommended: true
  message: Please follow semantic versioning. {{value}}
is not a valid version.
  given: $.info.version
  then:
    function: pattern
    functionOptions:
      match: "^([0-9]+.[0-9]+.[0-9]+)$"
```

```
no-http-basic:
  description: "Consider a more secure alternative
to HTTP Basic."
  severity: warn
  given: $.components.securitySchemes[*]
  then:
    field: scheme
    function: pattern
    functionOptions:
      notMatch: basic
```

```
unknown-error-format:
  description: "Errors SHOULD support either RFC 7807 or JSON:API Errors."
  formats:
    - oas3
  severity: warn
  given: $.paths.[*].responses[?(@property.match(/^(4|5)/))].content.*~
  then:
    function: enumeration
    functionOptions:
      values:
      - application/vnd.api+json
      - application/problem+xml
      - application/problem+json
```

```yaml
patch-request-content-type:
  severity: error
  description: '`PATCH` requests cannot use `application/json`'
  given: $.paths.*.[?(@property == 'patch')].requestBody.content[?(@property == 'application/json')]^
  then:
    function: falsy
patch-prefer-merge:
  severity: warn
  description: Prefer `application/merge-patch+json` for `PATCH` requests
  given: $.paths.*.[?(@property == 'patch')].requestBody.content
  then:
    function: contains
    functionOptions:
      match: 'application/merge-patch\+json'
```

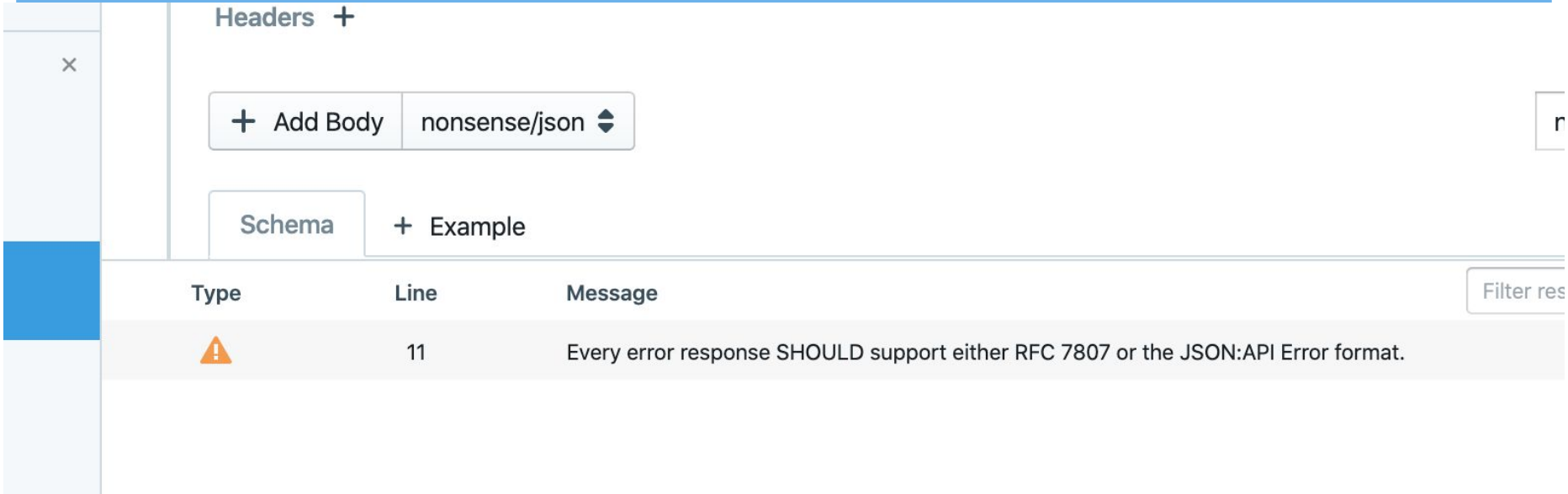
```
const writeGood = require('write-good');

module.exports = function(targetValue, option) {
  const suggestions = writeGood(targetValue || '');

  if (!suggestions.length) {
    return;
  }

  return suggestions.map(s => {
    return {
      message: s.reason
    };
  });
};
```

# Custom Rules in Studio



# Spectral CLI

```
$ spectral lint example.yaml -r apisyouwonthate.yml
OpenAPI 3.x detected

/Users/phil/src/openapi-contrib/style-guides/example.yaml
 19:15  warning  no-http-basic  HTTP Basic is a pretty
insecure way to pass credentials around, please consider
an alternative.

✖ 1 problem (0 errors, 1 warning, 0 infos, 0 hints)
```

# VS Code Spectral

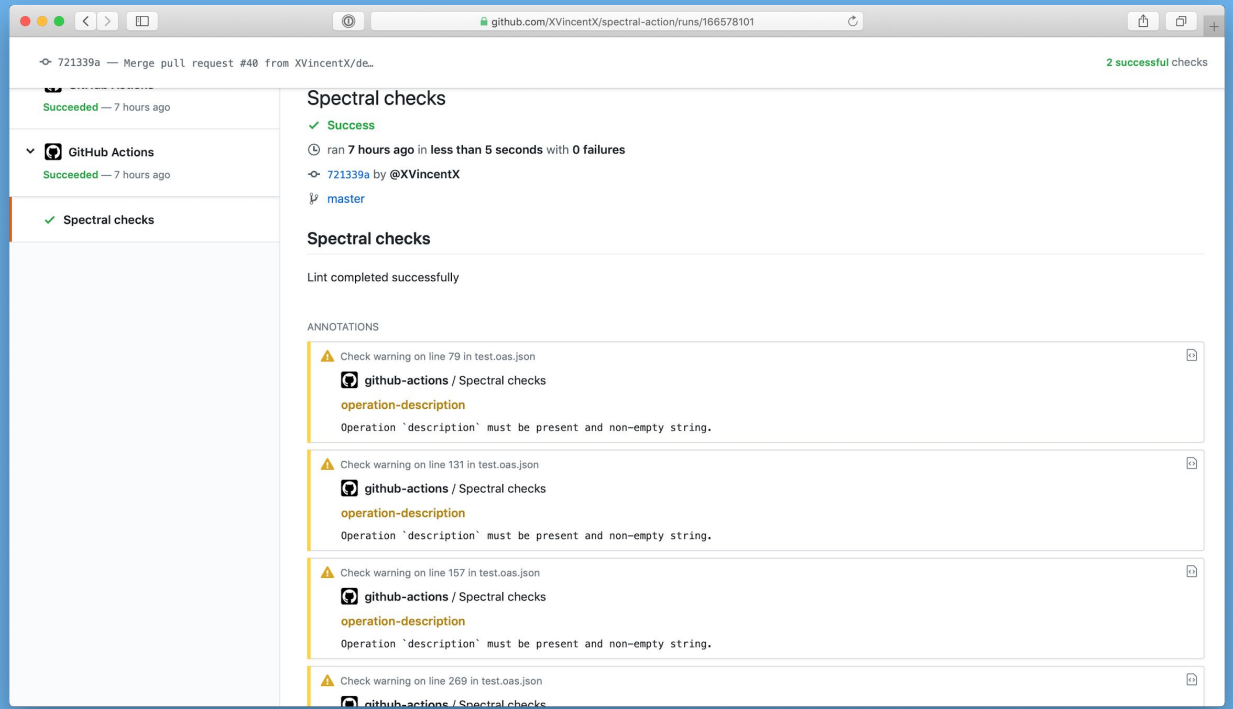
```
7     get:
8       responses:
9         '200':
10          content:
11            nonsense/json:
12              schema:
13                type: string
14                example: 'hello!'
15  components:
16    securitySchemes:
17      please-hack-me:
18        type: http
19        scheme: basic
20
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS 8 Filter. E.g.: text, **/*.ts, !**/node_modules/**

! example.yaml 1

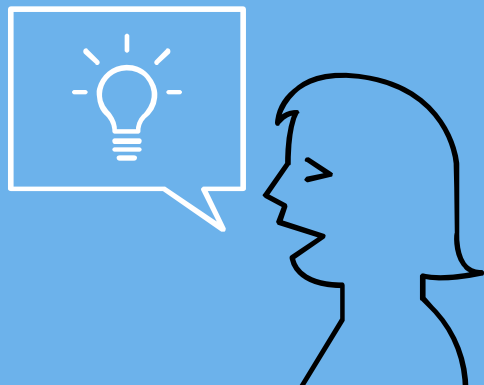
⚠ HTTP Basic is a pretty insecure way to pass credentials around, please consider an alternative. spectral(no-http-basic) [19,

# Spectral GitHub Action



API Design Reviews cannot be completely replaced by automation.

It simplifies the process massively, removing 80% of rejections before reviewers even look.



# THANKS!

## Any questions?

You can find me at:

@philsturgeon

phil@stoplight.io